

## ARM®-based 32-bit Cortex®-M4 MCU with 16 to 64 KB Flash, sLib, 10 timers, ADC, 7 communication interfaces

### Feature

- **Core: ARM®32-bit Cortex®-M4F CPU**
  - 120 MHz maximum frequency, with a Memory Protection Unit (MPU), single-cycle multiplication and hardware division
  - DSP instructions
- **Memories**
  - 16 to 64 KBytes of internal Flash memory
  - 4 Kbytes of boot code area used as a Bootloader or as a general instruction/data memory (one-time-configured)
  - sLib: configurable part of main Flash set as a library area with code executable but secured, non-readable
  - 8 to 16 KBytes of SRAM
- **Clock, reset and power control**
  - 2.4 V ~ 3.6 V application supply and I/Os
  - Power-on reset (POR)/ low-voltage reset (LVR), and power voltage monitor (PVM)
  - 4 to 25 MHz crystal (HEXT)
  - Internal 8 MHz factory-trimmed clock (HICK), accuracy 1% at  $T_A=25\text{ }^{\circ}\text{C}$ , 2 % at  $T_A=-40$  to  $+105\text{ }^{\circ}\text{C}$
  - Internal 40 kHz RC oscillator
  - 32 kHz crystal oscillator (LEXT)
- **Low power**
  - Sleep, Deepsleep, and Standby modes
- **1 x 12-bit A/D converter (up to 15 input channels)**
  - Conversion range: 0 V to 3.6 V
- **1 x COMP, 5 x external input channels and 1 x internal reference voltage channel**
- **2 x operational amplifiers**
- **DMA: 5-channel DMA controller**
  - Peripherals supported: timers, ADC, I<sup>2</sup>S, SPI, I<sup>2</sup>C and USART
- **Debug mode**
  - Serial wire debug (SWD) and JTAG
- **Up to 39 fast GPIOs**
  - All mappable to external interrupt vectors
- Almost 5 V-tolerant
- All fast I/Os, registers accessible with  $f_{AHB}$  speed
- **Up to 10 Timers (TMR)**
  - 1 x 16-bit 7-channel advanced timer, 6-channel PWM output with dead-time generator and emergency stop
  - 5 x 16-bit timers, each with 4 IC/OC/PWM or pulse counter and encoder input
  - 1 x 16-bit basic timer
  - 2 x Watchdog timers (WDT and WWDT)
  - SysTick timer: 24-bit downcounter
- **ERTC: enhanced RTC**
- **Up to 7 communication interfaces**
  - 2 x I<sup>2</sup>C interfaces (SMBus/PMBus support)
  - 2 x USARTs/UART (ISO7816 interface, LIN, IrDA and modem control)
  - 2 x SPIs, both with I<sup>2</sup>S interface multiplexed
  - Infrared transmitter
- **CRC Calculation Unit**
- **96-bit ID (UID)**
- **Packaging**
  - LQFP48 7 x 7 mm
  - LQFP32 7 x 7 mm
  - QFN32 5 x 5 mm
  - QFN32 4 x 4 mm
  - QFN28 4 x 4 mm
  - TSSOP20 6.5 x 4.4 mm
- **List of Models**

Internal Flash	Model
64 KBytes	AT32F421C8T7, AT32F421K8T7 AT32F421K8U7, AT32F421K8U7-4 AT32F421F8P7, AT32F421G8U7, AT32F4212C8T7
32 KBytes	AT32F421C6T7, AT32F421K6T7 AT32F421K6U7, AT32F421K6U7-4 AT32F421F6P7, AT32F421G6U7
16 KBytes	AT32F421C4T7, AT32F421K4T7 AT32F421K4U7, AT32F421K4U7-4 AT32F421F4P7, AT32F421G4U7

## Contents

<b>1</b>	<b>System architecture .....</b>	<b>25</b>
1.1	System overview.....	27
1.1.1	ARM Cortex™-M4 processor.....	27
1.1.2	Bit band.....	27
1.1.3	Interrupt and exception vectors .....	29
1.1.4	System Tick (SysTick) .....	31
1.1.5	Reset .....	31
1.2	List of abbreviations for registers .....	33
1.3	Device characteristics information .....	33
1.3.1	Flash memory size register .....	33
1.3.2	Device electronic signature .....	33
<b>2</b>	<b>Memory resources .....</b>	<b>34</b>
2.1	Internal memory address map .....	34
2.2	Flash memory.....	35
2.3	SRAM memory.....	35
2.4	Peripheral address map.....	36
<b>3</b>	<b>Power control (PWC).....</b>	<b>39</b>
3.1	Introduction .....	39
3.2	Main Features .....	39
3.3	POR/LVR .....	40
3.4	Power voltage monitor (PVM).....	40
3.5	Power domain.....	41
3.6	Power saving modes .....	41
3.7	PWC registers .....	43
3.7.1	Power control register (PWC_CTRL) .....	43
3.7.2	Power control/status register (PWC_CTRLSTS) .....	44
3.7.3	Power control register 2 (PWC_CTRL2) .....	44
<b>4</b>	<b>Clock and reset manage (CRM).....</b>	<b>46</b>
4.1	Clock .....	46

4.1.1	Clock sources .....	46
4.1.2	System clock.....	47
4.1.3	Peripheral clock .....	47
4.1.4	Clock fail detector .....	48
4.1.5	Auto step-by-step system clock switch.....	48
4.1.6	Internal clock output.....	48
4.1.7	Interrupts.....	48
4.2	Reset.....	48
4.2.1	System reset.....	48
4.2.2	Battery powered domain reset.....	49
4.3	CRM registers .....	49
4.3.1	Clock control register (CRM_CTRL).....	50
4.3.2	Clock configuration register (CRM_CFG) .....	51
4.3.3	Clock interrupt register (CRM_CLKINT) .....	52
4.3.4	APB2 peripheral reset register (CRM_APB2RST) .....	53
4.3.5	APB1 peripheral reset register1 (CRM_APB1RST) .....	54
4.3.6	AHB peripheral clock enable register (CRM_AHBEN) .....	54
4.3.7	APB2 peripheral clock enable register (CRM_APB2EN) .....	55
4.3.8	APB1 peripheral clock enable register (CRM_APB1EN) .....	56
4.3.9	Battery powered domain control register (CRM_BPDC).....	57
4.3.10	Control/status register (CRM_CTRLSTS) .....	57
4.3.11	AHB peripheral reset register (CRM_AHBRST) .....	58
4.3.12	PLL configuration register (CRM_PLL).....	58
4.3.13	Additional register (CRM_MISC1) .....	59
4.3.14	Additional register (CRM_MISC2) .....	60
5	<b>Embedded Flash memory controller (FLASH) .....</b>	<b>61</b>
5.1	FLASH introduction.....	61
5.2	Flash memory operation .....	63
5.2.1	Unlock/lock .....	63
5.2.2	Erase operation.....	63
5.2.3	Programming operation.....	65
5.2.4	Read operation .....	66
5.3	Main Flash memory extension area .....	66
5.4	User system data area .....	67

5.4.1	Unlock/lock .....	67
5.4.2	Erase operation.....	67
5.4.3	Programming operation.....	69
5.4.4	Read operation .....	70
5.5	Flash memory protection .....	70
5.5.1	Access protection.....	70
5.5.2	Erase/program protection.....	71
5.6	Read access.....	71
5.7	Special functions .....	71
5.7.1	Security library settings .....	71
5.7.2	Boot memory used as memory extension area .....	72
5.7.3	CRC verify .....	73
5.8	Flash memory registers .....	73
5.8.1	Flash performance select register (FLASH_PSR) .....	74
5.8.2	Flash unlock register (FLASH_UNLOCK) .....	74
5.8.3	Flash user system data unlock register (FLASH_USD_UNLOCK) ...	74
5.8.4	Flash status register (FLASH_STS) .....	75
5.8.5	Flash control register (FLASH_CTRL).....	76
5.8.6	Flash address register (FLASH_ADDR) .....	77
5.8.7	User system data register (FLASH_USD).....	77
5.8.8	Erase/program protection status register (FLASH_EPPS) .....	77
5.8.9	Flash security library status register 0 (SLIB_STS0).....	78
5.8.10	Flash security library status register1 (SLIB_STS1).....	78
5.8.11	Security library password clear register (SLIB_PWD_CLR) .....	79
5.8.12	Security library additional status register (SLIB_MISC_STS).....	79
5.8.13	Flash CRC address register (FLASH_CRC_ARR) .....	79
5.8.14	Flash CRC control register (FLASH_CRC_CTRL) .....	80
5.8.15	Flash CRC check result register (FLASH_CRC_CHK).....	80
5.8.16	Security library password setting register (SLIB_SET_PWD) .....	80
5.8.17	Security library address setting register (SLIB_SET_RANGE) .....	80
5.8.18	Flash extension memory security library setting register (EM_SLIB_SET)	
82		
5.8.19	Boot memory mode setting register (BTM_MODE_SET) .....	82
5.8.20	Security library unlock register (FLASH_UNLOCK) .....	82

<b>6</b>	<b>General-purpose I/Os (GPIOs).....</b>	<b>83</b>
6.1	Introduction .....	83
6.2	Functional overview .....	83
6.2.1	GPIO structure .....	83
6.2.2	GPIO reset status.....	83
6.2.3	General-purpose input configuration.....	84
6.2.4	Analog input/output configuration .....	84
6.2.5	General-purpose output configuration.....	84
6.2.6	GPIO port protection .....	84
6.2.7	IOMUX structure .....	85
6.2.8	Multiplexed function input configuration .....	85
6.2.9	IOMUX function input/output .....	86
6.2.10	Peripheral multiplexed function configuration .....	88
6.2.11	IOMUX map priority .....	88
6.2.12	External interrupt/wake-up lines .....	88
6.3	GPIO registers.....	88
6.3.1	GPIO configuration register (GPIO <sub>x</sub> _CFG) (x=A..H) .....	89
6.3.2	GPIO input mode register (GPIO <sub>x</sub> _OMODE) (x=A..H) .....	89
6.3.3	GPIO drive capability register (GPIO <sub>x</sub> _ODRVR) (x=A..H) .....	89
6.3.4	GPIO pull-up/pull-down register (GPIO <sub>x</sub> _PULL) (x=A..H) .....	89
6.3.5	GPIO input data register (GPIO <sub>x</sub> _IDT) (x=A..H) .....	90
6.3.6	GPIO output data register (GPIO <sub>x</sub> _ODT) (x= A..H) .....	90
6.3.7	GPIO set/clear register (GPIO <sub>x</sub> _SCR) (x=A..H) .....	90
6.3.8	GPIO write protection register (GPIO <sub>x</sub> _WPR) (x=A..H) .....	90
6.3.9	GPIO multiplexed function low register (GPIO <sub>x</sub> _MUXL) (x=A..H) ....	91
6.3.10	GPIO multiplexed function high register (GPIO <sub>x</sub> _MUXH) (x=A..H) ..	91
6.3.11	GPIO bit clear register (GPIO <sub>x</sub> _CLR) (x=A..H) .....	91
6.3.12	GPIO huge current control register (GPIO <sub>x</sub> _HDRV) (x=A..H) .....	91
<b>7</b>	<b>System configuration controller (SCFG) .....</b>	<b>92</b>
7.1	Introduction .....	92
7.2	SCFG registers.....	92
7.2.1	SCFG configuration register1 (SCFG_CFG1) .....	92
7.2.2	SCFG external interrupt configuration register1 (SCFG_EXINTC1)	93
7.2.3	SCFG external interrupt configuration register2 (SCFG_EXINTC2)	94

7.2.4	SCFG external interrupt configuration register3 (SCFG_EXINTC3)	94
7.2.5	SCFG external interrupt configuration register4 (SCFG_EXINTC4)	95
<b>8</b>	<b>External interrupt/Event controller (EXINT) .....</b>	<b>96</b>
8.1	EXINT introduction.....	96
8.2	Function overview and configuration procedure .....	96
8.3	EXINT registers .....	97
8.3.1	Interrupt enable register (EXINT_INTEN).....	97
8.3.2	Event enable register (EXINT_EVTEN) .....	97
8.3.3	Polarity configuration register1 (EXINT_POLCFG1).....	97
8.3.4	Polarity configuration register2 (EXINT_POLCFG2).....	98
8.3.5	Software trigger register (EXINT_SWTRG).....	98
8.3.6	Interrupt status register (EXINT_INTSTS) .....	98
<b>9</b>	<b>DMA controller (DMA) .....</b>	<b>99</b>
9.1	Introduction .....	99
9.2	Main features .....	99
9.3	Functional overview .....	99
9.3.1	DMA configuration.....	99
9.3.2	Handshake mechanism.....	100
9.3.3	Arbiter .....	100
9.3.4	Programmable data transfer width .....	101
9.3.5	Errors .....	102
9.3.6	Interrupts.....	102
9.3.7	Fixed DMA request mapping .....	102
9.4	DMA registers.....	103
9.4.1	DMA interrupt status register (DMA_STS) .....	104
9.4.2	DMA interrupt flag clear register (DMA_CLR) .....	105
9.4.3	DMA channel-x configuration register (DMA_CxCTRL) (x = 1...5)106	
9.4.4	DMA channel-x number of data register (DMA_CxDTCNT) (x = 1...5)107	
9.4.5	DMA channel-x peripheral address register (DMA_CxPADDR) (x = 1...5)	
	107	
9.4.6	DMA channel-x memory address register (DMA_CxMADDR) (x = 1...5)107	
<b>10</b>	<b>CRC calculation unit (CRC).....</b>	<b>108</b>

10.1	CRC introduction .....	108
10.2	CRC registers.....	108
10.2.1	Data register (CRC_DT).....	108
10.2.2	Common data register (CRC_CDT).....	108
10.2.3	Control register (CRC_CTRL).....	109
10.2.4	Initialization register (CRC_IDT) .....	109
<b>11</b>	<b>I<sup>2</sup>C interface .....</b>	<b>110</b>
11.1	I <sup>2</sup> C introduction.....	110
11.2	I <sup>2</sup> C main features .....	110
11.3	I <sup>2</sup> C functional overview.....	110
11.4	I <sup>2</sup> C interface .....	111
11.4.1	I <sup>2</sup> C slave communication flow.....	112
11.4.2	I <sup>2</sup> C master communication flow .....	115
11.4.3	Data transfer using DMA.....	121
11.4.4	SMBus.....	122
11.4.5	I <sup>2</sup> C interrupt requests .....	123
11.4.6	I <sup>2</sup> C debug mode .....	124
11.5	I <sup>2</sup> C registers .....	124
11.5.1	Control register1 (I2C_CTRL1).....	124
11.5.2	Control register2 (I2C_CTRL2).....	125
11.5.3	Own address register1 (I2C_OADDR1) .....	126
11.5.4	Own address register2 (I2C_OADDR2) .....	126
11.5.5	Data register (I2C_DT) .....	127
11.5.6	Status register1 (I2C_STS1) .....	127
11.5.7	Status register2 (I2C_STS2) .....	129
11.5.8	Clock control register (I2C_CLKCTRL) .....	130
11.5.9	Clock rise time register (I2C_TMRISE) .....	130
<b>12</b>	<b>Universal synchronous/asynchronous receiver/transmitter (USART)</b>	<b>131</b>
12.1	USART introduction .....	131
12.2	Full-duplex/half-duplex selector .....	133
12.3	Mode selector.....	133
12.3.1	Introduction.....	133
12.3.2	Configuration procedure .....	133

12.4 USART frame format and configuration .....	137
12.5 DMA transfer introduction .....	138
12.5.1 Transmission using DMA .....	138
12.5.2 Reception using DMA .....	138
12.6 Baud rate generation .....	139
12.6.1 Introduction .....	139
12.6.2 Configuration .....	139
12.7 Transmitter .....	140
12.7.1 Transmitter introduction .....	140
12.7.2 Transmitter configuration .....	140
12.8 Receiver .....	141
12.8.1 Receiver introduction .....	141
12.8.2 Receiver configuration .....	141
12.8.3 Start bit and noise detection .....	142
12.9 Tx/Rx swap .....	143
12.10 Interrupt requests .....	143
12.11 I/O pin control .....	144
12.12 USART registers .....	144
12.12.1 Status register (USART_STS) .....	145
12.12.2 Data register (USART_DT) .....	146
12.12.3 Baud rate register (USART_BAUDR) .....	146
12.12.4 Control register1 (USART_CTRL1) .....	146
12.12.5 Control register2 (USART_CTRL2) .....	147
12.12.6 Control register3 (USART_CTRL3) .....	148
12.12.7 Guard time and divider register (USART_GDIV) .....	149
<b>13 Serial peripheral interface (SPI) .....</b>	<b>150</b>
13.1 SPI introduction .....	150
13.2 Functional overview .....	150
13.2.1 SPI description .....	150
13.2.2 Full-duplex/half-duplex selector .....	151
13.2.3 Chip select controller .....	153
13.2.4 SPI_SCK controller .....	153
13.2.5 CRC .....	153

13.2.6 DMA transfer.....	154
13.2.7 Transmitter .....	155
13.2.8 Receiver .....	155
13.2.9 Motorola mode .....	156
13.2.10 Interrupts .....	158
13.2.11 IO pin control .....	159
13.2.12 Precautions.....	159
<b>13.3 I2S functional description .....</b>	<b>159</b>
13.3.1 I <sup>2</sup> S introduction .....	159
13.3.2 Operation mode selector.....	160
13.3.3 Audio protocol selector .....	161
13.3.4 I2S_CLK controller .....	162
13.3.5 DMA transfer.....	164
13.3.6 Transmitter/Receiver .....	165
13.3.7 I2S communication timings .....	166
13.3.8 Interrupts .....	166
13.3.9 IO pin control .....	166
<b>13.4 SPI registers .....</b>	<b>167</b>
13.4.1 SPI control register1 (SPI_CTRL1) (Not used in I <sup>2</sup> S mode) .....	167
13.4.2 SPI control register2 (SPI_CTRL2) .....	168
13.4.3 SPI status register (SPI_STS) .....	169
13.4.4 SPI data register (SPI_DT) .....	169
13.4.5 SPICRC register (SPI_CPOLY) (Not used in I <sup>2</sup> S mode).....	169
13.4.6 SPIRxCRC register (SPI_RCRC) (Not used in I <sup>2</sup> S mode) .....	170
13.4.7 SPITxCRC register (SPI_TCRC).....	170
13.4.8 SPI_I2S configuration register (SPI_I2SCTRL) .....	170
13.4.9 SPI_I2S prescaler register (SPI_I2SCLKP) .....	171
<b>14 Timer .....</b>	<b>172</b>
14.1 General-purpose timer (TMR6).....	173
14.1.1 TMR6 introduction .....	173
14.1.2 TMR6 main features .....	173
14.1.3 TMR6 functional overview .....	173
14.1.3.1 Count clock .....	173
14.1.3.2 Counting mode .....	173

14.1.3.3 Debug mode .....	174
14.1.4 TMR6 registers .....	175
14.1.4.1 TMR6 control register1 (TMRx_CTRL1) .....	176
14.1.4.2 TMR6 control register2 (TMRx_CTRL2) .....	176
14.1.4.3 TMR6 DMA/interrupt enable register (TMRx_IDEN).....	176
14.1.4.4 TMR6 interrupt status register (TMRxISTS) .....	177
14.1.4.5 TMR6 software event register (TMRx_SWEVT) .....	177
14.1.4.6 TMR6 counter value (TMRx_CVAL) .....	177
14.1.4.7 TMR6 division (TMRx_DIV) .....	177
14.1.4.8 TMR6 period register (TMRx_PR).....	177
14.2 General-purpose timer (TMR3).....	178
14.2.1 TMR3 introduction .....	178
14.2.2 TMR3 main features .....	178
14.2.3 TMR3 functional overview .....	178
14.2.3.1 Count clock .....	178
14.2.3.2 Counting mode .....	182
14.2.3.3 TMR input function.....	185
14.2.3.4 TMR output function.....	187
14.2.3.5 TMR synchronization.....	191
14.2.3.6 Debug mode .....	193
14.2.4 TMR3 registers .....	193
14.2.4.1 Control register1 (TMR3_CTRL1) .....	194
14.2.4.2 Control register2 (TMR3_CTRL2) .....	195
14.2.4.3 Slave timer control register (TMR3_STCTRL) .....	195
14.2.4.4 DMA/interrupt enable register (TMR3_IDEN) .....	196
14.2.4.5 Interrupt status register (TMR3ISTS) .....	197
14.2.4.6 Software event register (TMR3_SWEVT).....	198
14.2.4.7 Channel mode register1 (TMRx_CM1) .....	198
14.2.4.8 Channel mode register2 (TMR3_CM2) .....	200
14.2.4.9 Channel control register (TMR3_CCTRL) .....	201
14.2.4.10 Counter value (TMR3_CVAL) .....	202
14.2.4.11 Division value (TMR3_DIV) .....	202
14.2.4.12 Period register (TMR3_PR) .....	202
14.2.4.13 Channel 1 data register (TMR3_C1DT) .....	202
14.2.4.14 Channel 2 data register (TMR3_C2DT) .....	202
14.2.4.15 Channel 3 data register (TMR3_C3DT) .....	202
14.2.4.16 Channel 4 data register (TMR3_C4DT) .....	203
14.2.4.17 DMA control register (TMR3_DMACTRL).....	203

14.2.4.18 DMA data register (TMR3_DMADT) .....	203
<b>14.3 General-purpose timer (TMR14) .....</b>	<b>204</b>
14.3.1 TMR14 introduction .....	204
14.3.2 TMR14 main features .....	204
14.3.3 TMR14 functional overview .....	204
14.3.3.1 Count clock .....	204
14.3.3.2 Counting mode .....	205
14.3.3.3 TMR input function.....	206
14.3.3.4 TMR output function.....	207
14.3.3.5 Debug mode .....	208
14.3.4 TMR14 registers.....	208
14.3.4.1 Control register1 (TMR14_CTRL1) .....	209
14.3.4.2 Interrupt enable register (TMR14_IDEN) .....	209
14.3.4.3 Interrupt status register (TMR14ISTS) .....	209
14.3.4.4 Software event register (TMR14_SWEVT) .....	210
14.3.4.5 Channel mode register1 (TMR14_CM1) .....	210
14.3.4.6 Channel control register (TMR14_CCTRL) .....	213
14.3.4.7 Counter value (TMR14_CVAL) .....	213
14.3.4.8 Division value (TMR14_DIV) .....	213
14.3.4.9 Period register (TMR14_PR) .....	213
14.3.4.10 Channel 1 data register (TMR14_C1DT) .....	213
14.3.4.11 Channel input remap register (TMR14_RMP).....	215
<b>14.4 General-purpose timer (TMR15) .....</b>	<b>216</b>
14.4.1 TMR15 introduction .....	216
14.4.2 TMR15 main features .....	216
14.4.3 TMR15 functional overview .....	216
14.4.3.1 Count clock .....	216
14.4.3.2 Counting mode .....	218
14.4.3.3 TMR input function.....	220
14.4.3.4 TMR output function.....	222
14.4.3.5 TMR brake function.....	226
14.4.3.6 TMR synchronization.....	227
14.4.3.7 Debug mode .....	228
14.4.4 TMR15 registers.....	229
14.4.4.1 Control register1 (TMR15_CTRL1) .....	229
14.4.4.2 Control register2 (TMR15_CTRL2) .....	230
14.4.4.3 TMR15 slave timer control register (TMR15_STCTRL) .....	230

14.4.4.4 TMR15 DMA/interrupt enable register (TMR15_IDEN) .....	231
14.4.4.5 TMR15 interrupt status register (TMR15ISTS) .....	232
14.4.4.6 TMR15 software event register (TMR15_SWEVT) .....	233
14.4.4.7 TMR15 channel mode register1 (TMR15_CM1) .....	233
14.4.4.8 TMR15 channel control register (TMR15_CCTRL) .....	236
14.4.4.9 TMR15 Counter value (TMR15_CVAL) .....	238
14.4.4.10 TMR15 Division value (TMR15_DIV) .....	238
14.4.4.11 TMR15 period register (TMR15_PR) .....	238
14.4.4.12 TMR15 repetition period register (TMR15_RPR) .....	238
14.4.4.13 TMR15 channel 1 data register (TMR15_C1DT) .....	238
14.4.4.14 TMR15 channel 2 data register (TMR15_C2DT) .....	238
14.4.4.15 TMR15 brake register (TMR15_BRK) .....	239
14.4.4.16 TMR15 DMA control register (TMR15_DMACTRL) .....	240
14.4.4.17 TMR15 DMA data register (TMR15_DMADT) .....	240
<b>14.5 General-purpose timer (TMR16 and TMR17) .....</b>	<b>241</b>
14.5.1 TMR16 and TMR17 introduction .....	241
14.5.2 TMR16 and TMR17 main features .....	241
14.5.3 TMR16 and TMR17 functional overview .....	241
14.5.3.1 Count clock .....	241
14.5.3.2 Counting mode .....	242
14.5.3.3 TMR input function .....	243
14.5.3.4 TMR output function .....	244
14.5.3.5 TMR brake function .....	247
14.5.3.6 Debug mode .....	248
14.5.4 TMR16 and TMR17 registers .....	248
14.5.4.1 TMR16 and TMR17 control register1 (TMRx_CTRL1) .....	249
14.5.4.2 TMR16 and TMR17 control register2 (TMRx_CTRL2) .....	249
14.5.4.3 TMR16 and TMR17 DMA/interrupt enable register (TMRx_IDEN) .....	250
14.5.4.4 TMR16 and TMR17 interrupt status register (TMRxISTS) .....	250
14.5.4.5 TMR16 and TMR17 software event register (TMRx_SWEVT) .....	251
14.5.4.6 TMR16 and TMR17 channel mode register1 (TMRx_CM1) .....	251
14.5.4.7 TMR16 and TMR17 channel control register (TMRx_CCTRL) .....	254
14.5.4.8 TMR16 and TMR17 counter value (TMRx_CVAL) .....	255
14.5.4.9 TMR16 and TMR17 division value (TMRx_DIV) .....	255
14.5.4.10 TMR16 and TMR17 period register (TMRx_PR) .....	255
14.5.4.11 TMR16 and TMR17 repetition period register (TMRx_RPR) .....	255
14.5.4.12 TMR16 and TMR17 channel 1 data register (TMRx_C1DT) .....	255
14.5.4.13 TMR16 and TMR17 brake register (TMRx_BRK) .....	257

14.5.4.14 TMR16 and TMR17 DMA control register (TMRx_DMACTRL)	258
14.5.4.15 TMR16 and TMR17 DMA data register (TMRx_DMADT) .....	258
<b>14.6 Advanced-control timers (TMR1) .....</b>	<b>259</b>
14.6.1 TMR1 introduction .....	259
14.6.2 TMR1 main features .....	259
14.6.3 TMR1 functional overview .....	259
14.6.3.1 Count clock .....	259
14.6.3.2 Counting mode .....	263
14.6.3.3 TMR input function.....	267
14.6.3.4 TMR output function.....	269
14.6.3.5 TMR brake function .....	273
14.6.3.6 TMR synchronization.....	274
14.6.3.7 Debug mode .....	275
14.6.4 TMR1 registers .....	276
14.6.4.1 TMR1 control register1 (TMR1_CTRL1) .....	276
14.6.4.2 TMR1 control register2 (TMR1_CTRL2) .....	278
14.6.4.3 TMR1 slave timer control register (TMR1_STCTRL).....	278
14.6.4.4 TMR1 DMA/interrupt enable register (TMR1_IDEN).....	279
14.6.4.5 TMR1 interrupt status register (TMR1ISTS) .....	281
14.6.4.6 TMR1 software event register (TMR1_SWEVT).....	282
14.6.4.7 TMR1 channel mode register1 (TMR1_CM1) .....	282
14.6.4.8 TMR1 channel mode register2 (TMR1_CM2) .....	284
14.6.4.9 TMR1 Channel control register (TMR1_CCTRL).....	285
14.6.4.10 TMR1 counter value (TMR1_CVAL) .....	287
14.6.4.11 TMR1 division value (TMR1_DIV) .....	287
14.6.4.12 TMR1 period register (TMR1_PR).....	287
14.6.4.13 TMR1 repetition period register (TMR1_RPR).....	287
14.6.4.14 TMR1 channel 1 data register (TMR1_C1DT) .....	287
14.6.4.15 TMR1 channel 2 data register (TMR1_C2DT) .....	287
14.6.4.16 TMR1 channel 3 data register (TMR1_C3DT) .....	289
14.6.4.17 TMR1 channel 4 data register (TMR1_C4DT) .....	289
14.6.4.18 TMR1 brake register (TMR1_BRK).....	289
14.6.4.19 TMR1 DMA control register (TMR1_DMACTRL) .....	290
14.6.4.20 TMR1 DMA data register (TMR1_DMADT) .....	290
<b>15 Window watchdog timer (WWDT) .....</b>	<b>291</b>
15.1 WWDT introduction .....	291
15.2 WWDT main features .....	291

15.3	WWDT functional overview .....	291
15.4	Debug mode .....	292
15.5	WWDT registers .....	292
15.5.1	Control register (WWDT_CTRL) .....	292
15.5.2	Configuration register (WWDT_CFG) .....	293
15.5.3	Status register (WWDT_STS) .....	293
<b>16</b>	<b>Watchdog timer (WDT) .....</b>	<b>294</b>
16.1	WDT introduction .....	294
16.2	WDT main features .....	294
16.3	WDT functional overview .....	294
16.4	Debug mode .....	295
16.5	WDT registers .....	295
16.5.1	Command register (WDT_CMD) .....	295
16.5.2	Divider register (WDT_DIV) .....	295
16.5.3	Reload register (WDT_RLD) .....	296
16.5.4	Status register (WDT_STS) .....	296
<b>17</b>	<b>Enhanced real-time clock (ERTC) .....</b>	<b>297</b>
17.1	ERTC introduction .....	297
17.2	ERTC main features .....	297
17.3	ERTC functional overview .....	297
17.3.1	ERTC clock .....	297
17.3.2	ERTC initialization .....	298
17.3.3	ERTC calibration .....	300
17.3.4	Time stamp .....	300
17.3.5	Tamper detection .....	301
17.3.6	Multiplexed function output .....	301
17.3.7	ERTC wakeup .....	301
17.4	ERTC registers .....	302
17.4.1	ERTC time register (ERTC_TIME) .....	302
17.4.2	ERTC date register (ERTC_DATE) .....	303
17.4.3	ERTC control register (ERTC_CTRL) .....	303
17.4.4	ERTC initialization and status register (ERTC_STS) .....	304
17.4.5	ERTC divider register (ERTC_DIV) .....	305

17.4.6 ERTC alarm clock A register (ERTC_ALA) .....	305
17.4.7 ERTC write protection register (ERTC_WP) .....	306
17.4.8 ERTC subsecond register (ERTC_SBS) .....	306
17.4.9 ERTC time adjustment register (ERTC_TADJ).....	306
17.4.10 ERTC time stamp time register (ERTC_TSTM) .....	306
17.4.11 ERTC time stamp date register (ERTC_TSDT) .....	307
17.4.12 ERTC time stamp subsecond register (ERTC_TSSBS) .....	307
17.4.13 ERTC smooth calibration register (ERTC_SCAL) .....	307
17.4.14 ERTC tamper configuration register (ERTC_TAMP) .....	307
17.4.15 ERTC alarm clock A subsecond register (ERTC_ALASBS) .....	309
17.4.16 ERTC battery powered domain data register (ERTC_BPRx) .....	309
<b>18 Analog-to-digital converter (ADC).....</b>	<b>310</b>
18.1 ADC introduction .....	310
18.2 ADC main features.....	310
18.3 ADC structure.....	310
18.4 ADC functional overview.....	311
18.4.1 Channel management.....	311
18.4.1.1 Internal temperature sensor.....	312
18.4.1.2 Internal reference voltage.....	312
18.4.2 ADC operation process.....	312
18.4.2.1 Power-on and calibration .....	312
18.4.2.2 Trigger.....	313
18.4.2.3 Sampling and conversion sequence.....	314
18.4.3 Conversion sequence management .....	314
18.4.3.1 Sequence mode .....	314
18.4.3.2 Automatic preempted group conversion mode .....	314
18.4.3.3 Repetition mode.....	315
18.4.3.4 Partition mode .....	315
18.4.4 Data management .....	316
18.4.4.1 Data alignment .....	316
18.4.4.2 Data read .....	316
18.4.5 Voltage monitoring .....	317
18.4.6 Status flag and interrupts.....	317
18.5 ADC registers .....	317
18.5.1 ADC status register (ADC_STS) .....	318

18.5.2 ADC control register1 (ADC_CTRL1) .....	318
18.5.3 ADC control register2 (ADC_CTRL2) .....	319
18.5.4 ADC sampling time register 1 (ADC_SPT1).....	321
18.5.5 ADC sampling time register 2 (ADC_SPT2).....	322
18.5.6 ADC preempted channel data offset register x (ADC_PCDTOx) (x=1..4)	
324	
18.5.7 ADC voltage monitor high threshold register (ADC_VVHB).....	324
18.5.8 ADC voltage monitor low threshold register (ADC_VVLB).....	324
18.5.9 ADC ordinary sequence register 1 (ADC_OSQ1) .....	325
18.5.10 ADC ordinary sequence register 2 (ADC_OSQ2) .....	325
18.5.11 ADC ordinary sequence register 3 (ADC_OSQ3) .....	325
18.5.12 ADC preempted sequence register (ADC_PSQ).....	326
18.5.13 ADC preempted data register x (ADC_PDTx) (x=1..4) .....	326
18.5.14 ADC ordinary data register (ADC_ODT) .....	326
<b>19     Comparator (COMP) .....</b>	<b>327</b>
19.1 COMP introduction.....	327
19.2 Main features .....	327
19.3 Interrupt management .....	327
19.4 Design tips .....	328
19.5 Functional overview .....	328
19.5.1 Analog comparator .....	328
19.5.2 Glitch filter.....	329
19.6 CMP registers.....	329
19.6.1 Comparator control and status register 1 (COMP_CTRLSTS) .....	330
19.6.2 Glitch filter enable register (G_FILTER_EN) .....	331
19.6.3 Glitch filter high pulse count (HIGH-PULSE) .....	331
19.6.4 Glitch filter low pulse count (LOW-PULSE) .....	332
<b>20     Operational amplifier (OPA) .....</b>	<b>333</b>
20.1 Introduction .....	333
20.2 Main features .....	333
20.3 Functional description .....	333
<b>21     Infrared timer (IRTMR) .....</b>	<b>335</b>

<b>22</b>	<b>Debug (DEBUG) .....</b>	<b>336</b>
22.1	Debug introduction.....	336
22.2	Debug and trace .....	336
22.3	I/O pin control.....	336
22.4	DEBUG registers .....	336
22.4.1	DEBUG device ID (DEBUG_IDCODE) .....	337
22.4.2	DEBUG control register (DEBUG_CTRL) .....	338
<b>23</b>	<b>Revision history.....</b>	<b>340</b>

## List of figures

Figure 1-1 AT32F421 Series microcontrollers system architecture.....	26
Figure 1-2 Internal block diagram of Cortex®-M4 .....	27
Figure 1-3 Comparison between bit-band region and its alias region: image A .....	27
Figure 1-4 Comparison between bit-band region and its alias region: image B .....	28
Figure 1-5 Reset process .....	31
Figure 1-6 Example of MSP and PC initialization.....	32
Figure 2-1 AT32F421 address mapping.....	34
Figure 3-1 Block diagram of each power supply .....	39
Figure 3-2 Power-on reset/Low voltage reset waveform.....	40
Figure 3-3 PVM threshold and output .....	41
Figure 4-1 AT32F421 clock tree .....	46
Figure 4-2 System reset circuit.....	49
Figure 5-1 Flash memory sector erase process.....	64
Figure 5-2 Flash memory mass erase process .....	65
Figure 5-3 Flash memory programming process .....	66
Figure 5-4 System data area erase process .....	68
Figure 5-5 System data area programming process.....	69
Figure 6-1 GPIO basic structure.....	83
Figure 6-2 IOMUX structure .....	85
Figure 8-1 External interrupt/Event controller block diagram.....	96
Figure 9-1 DMA block diagram .....	99
Figure 9-2 Re-arbitrate after request/acknowledge.....	100
Figure 9-3 PWIDHT: byte, MWIDHT: half-word .....	101
Figure 9-4 PWIDHT: half-word, MWIDHT: word .....	101
Figure 9-5 PWIDHT: word, MWIDHT: byte .....	101
Figure 11-1 I <sup>2</sup> C bus protocol .....	110
Figure 11-2 I <sup>2</sup> C function block diagram .....	111
Figure 11-3 Transfer sequence of slave transmitter .....	113
Figure 11-4 Transfer sequence of slave receiver .....	114
Figure 11-5 Transfer sequence of master transmitter .....	115
Figure 11-6 Transfer sequence of master receiver.....	117
Figure 11-7 Transfer sequence of master receiver when N>2 .....	118
Figure 11-8 Transfer sequence of master receiver when N=2 .....	119
Figure 11-9 Transfer sequence of master receiver when N=1 .....	120
Figure 12-1 USART block diagram.....	131
Figure 12-2 BFF and FERR detection in LIN mode .....	134
Figure 12-3 Smartcard frame format .....	134
Figure 12-4 IrDA DATA(3/16) – normal mode .....	135
Figure 12-5 Hardware flow control .....	135
Figure 12-6 Mute mode using Idle line or Address mark detection.....	136
Figure 12-7 8-bit format USART synchronous mode .....	136
Figure 12-8 Word length .....	137
Figure 12-9 Stop bit configuration .....	138

Figure 12-10 TDC/TDBE behavior when transmitting .....	140
Figure 12-11 Data sampling for noise detection.....	143
Figure 12-12 Tx/Rx swap.....	143
Figure 12-13 USART interrupt map diagram.....	144
Figure 13-1 SPI block diagram .....	150
Figure 13-2 SPI two-wire unidirectional full-duplex connection .....	151
Figure 13-3 Single-wire unidirectional receive only in SPI master mode.....	151
Figure 13-4 Single-wire unidirectional receive only in SPI slave mode .....	152
Figure 13-5 Single-wire bidirectional half-duplex mode .....	152
Figure 13-6 Master full-duplex communications .....	156
Figure 13-7 Slave full-duplex communications.....	157
Figure 13-8 Slave full-duplex communications.....	157
Figure 13-9 Slave half-duplex receive.....	157
Figure 13-10 Slave half-duplex transmit.....	158
Figure 13-11 Master half-duplex receive .....	158
Figure 13-12 SPI interrupts .....	158
Figure 13-13 I <sup>2</sup> S block diagram .....	159
Figure 13-14 I <sup>2</sup> S slave device transmission .....	160
Figure 13-15 I <sup>2</sup> S slave device reception.....	160
Figure 13-16 I <sup>2</sup> S master device transmission.....	161
Figure 13-17 I <sup>2</sup> S master device reception .....	161
Figure 13-18 CK & MCK source in master mode .....	163
Figure 13-19 Audio standard timings.....	166
Figure 13-20 I <sup>2</sup> S interrupts.....	166
Figure 14-1 Basic timer block diagram .....	173
Figure 14-2 Counter timing diagram, CK_INT divided by 1 .....	173
Figure 14-3 Counter structure .....	174
Figure 14-4 Overflow event when PRBEN=0 .....	174
Figure 14-5 Overflow event when PRBEN=1 .....	174
Figure 14-6 Counter timing diagram, internal clock divided by 4 .....	174
Figure 14-7 Block diagram of general-purpose timer.....	178
Figure 14-8 Count clock block diagram.....	179
Figure 14-9 Use CK_INT to drive counter, with TMRx_DIV=0x0 and TMRx_PR=0x16 .....	179
Figure 14-10 Block diagram of external clock mode A.....	180
Figure 14-11 Counting in external clock mode A, with PR=0x32 and DIV=0x0 .....	180
Figure 14-12 Block diagram in external clock mode B .....	180
Figure 14-13 Counting in external clock mode B, with PR=0x32 and DIV=0x0 .....	181
Figure 14-14 Counter timing with prescaler value changing from 1 to 4 .....	181
Figure 14-15 Counter structure .....	182
Figure 14-16 Overflow event when PRBEN=0 .....	182
Figure 14-17 Overflow event when PRBEN=1 .....	183
Figure 14-18 Counter timing diagram, internal clock divided by 4 .....	183
Figure 14-19 Counter timing diagram, internal clock divided by 1, TMRx_PR=0x32 .....	184
Figure 14-20 Encoder mode structure.....	184
Figure 14-21 Example of counter behavior in encoder interface mode (encoder mode C).....	185

Figure 14-22 Input/output channel 1 main circuit.....	186
Figure 14-23 Channel 1 input stage .....	186
Figure 14-24 PWM input mode configuration.....	187
Figure 14-25 PWM input mode.....	187
Figure 14-26 Capture/compare channel output stage (channel 1 to 4) .....	188
Figure 14-27 C1ORAW toggles when counter value matches the C1DT value .....	189
Figure 14-28 Upcounting mode and PWM mode A.....	189
Figure 14-29 Up/down counting mode and PWM mode A.....	189
Figure 14-30 One-pulse mode.....	190
Figure 14-31 Clearing CxORAW(PWM mode A) by EXT input.....	190
Figure 14-32 Example of reset mode .....	191
Figure 14-33 Example of suspend mode .....	191
Figure 14-34 Example of trigger mode .....	191
Figure 14-35 Master/slave timer connection .....	192
Figure 14-36 Using master timer to start slave timer .....	192
Figure 14-37 Starting master and slave timers synchronously by an external trigger.....	193
Figure 14-38 Block diagram of general-purpose TMR14.....	204
Figure 14-39 Count clock.....	204
Figure 14-40 Use CK_INT to drive counter, with TMRx_DIV=0x0 and TMRx_PR=0x16 .....	204
Figure 14-41 Counter timing with prescaler value changing from 1 to 4 .....	205
Figure 14-42 Counter structure .....	205
Figure 14-43 Overflow event when PRBEN=0.....	206
Figure 14-44 Overflow event when PRBEN=1.....	206
Figure 14-45 Input/output channel 1 main circuit.....	206
Figure 14-46 Channel 1 input stage .....	207
Figure 14-47 Capture/compare channel output stage (channel 1) .....	207
Figure 14-48 C1ORAW toggles when counter value matches the C1DT value .....	208
Figure 14-49 Upcounting mode and PWM mode A.....	208
Figure 14-50 Block diagram of general-purpose TMR15.....	216
Figure 14-51 Count clock.....	216
Figure 14-52 Use CK_INT to drive counter, with TMRx_DIV=0x0 and TMRx_PR=0x16 .....	217
Figure 14-53 Block diagram of external clock mode A.....	217
Figure 14-54 Counting in external clock mode A, with PR=0x32 and DIV=0x0.....	218
Figure 14-55 Counter timing with prescaler value changing from 1 to 4 .....	218
Figure 14-56 Counter structure .....	219
Figure 14-57 Overflow event when PRBEN=0.....	219
Figure 14-58 Overflow event when PRBEN=1.....	219
Figure 14-59 OVFIF when RPR=2 .....	220
Figure 14-60 Input/output channel 1 main circuit.....	220
Figure 14-61 Channel 1 input stage .....	221
Figure 14-62 PWM input mode configuration.....	222
Figure 14-63 PWM input mode.....	222
Figure 14-64 Channel 1 output stage.....	222
Figure 14-65 Channel 2 output stage.....	223
Figure 14-66 C1ORAW toggles when counter value matches the C1DT value .....	224

Figure 14-67 Upcounting mode and PWM mode A.....	224
Figure 14-68 One-pulse mode.....	225
Figure 14-69 Complementary output with dead-time insertion .....	226
Figure 14-70 TMR control output.....	227
Figure 14-71 Example of TMR brake function.....	227
Figure 14-72 Example of reset mode .....	228
Figure 14-73 Example of suspend mode .....	228
Figure 14-74 Example of trigger mode.....	228
Figure 14-75 Block diagram of general-purpose TMR16 and TMR17 .....	241
Figure 14-76 Count clock.....	241
Figure 14-77 Use CK_INT to drive counter, with TMRx_DIV=0x0 and TMRx_PR=0x16 .....	242
Figure 14-78 Counter structure .....	242
Figure 14-79 Overflow event when PRBEN=0.....	243
Figure 14-80 Overflow event when PRBEN=1 .....	243
Figure 14-81 OVFIF when RPR=2 .....	243
Figure 14-82 Input/output channel 1 main circuit.....	244
Figure 14-83 Channel 1 input stage .....	244
Figure 14-84 Channel 1 output stage.....	244
Figure 14-85 C1ORAW toggles when counter value matches the C1DT value .....	246
Figure 14-86 Upcounting mode and PWM mode A .....	246
Figure 14-87 One-pulse mode.....	246
Figure 14-88 Complementary output with dead-time insertion .....	247
Figure 14-89 TMR output control.....	248
Figure 14-90 Example of TMR brake function.....	248
Figure 14-91 Block diagram of advanced-control timer .....	259
Figure 14-92 Count clock.....	260
Figure 14-93 Use CK_INT to drive counter, with TMRx_DIV=0x0 and TMRx_PR=0x16 .....	260
Figure 14-94 Block diagram of external clock mode A.....	261
Figure 14-95 Counting in external clock mode A, with PR=0x32 and DIV=0x0 .....	261
Figure 14-96 Block diagram of external clock mode B.....	262
Figure 14-97 Counting in external clock mode B, with PR=0x32 and DIV=0x0 .....	262
Figure 14-98 Counter timing with prescaler value changing from 1 to 4 .....	263
Figure 14-99 Counter structure .....	263
Figure 14-100 Overflow event when PRBEN=0.....	264
Figure 14-101 Overflow event when PRBEN=1 .....	264
Figure 14-102 Counter timing diagram with internal clock divided by 4 .....	264
Figure 14-103 Counter timing diagram with internal clock divided by 1 and TMRx_PR=0x32.....	265
Figure 14-104 OVFIF in upcounting mode and up/down counting mode .....	265
Figure 14-105 Encoder mode structure.....	266
Figure 14-106 Example of encoder interface mode C .....	267
Figure 14-107 Input/output channel 1 main circuit .....	267
Figure 14-108 Channel 1 input stage .....	268
Figure 14-109 PWM input mode configuration.....	269
Figure 14-110 PWM input mode .....	269
Figure 14-111 Output stage for channel 1 to 3 .....	269

Figure 14-112 Output stage for channel 4 .....	270
Figure 14-113 C1ORAW toggles when counter value matches the C1DT value .....	271
Figure 14-114 Upcounting mode and PWM mode A.....	271
Figure 14-115 Up/down counting mode and PWM mode A .....	271
Figure 14-116 One-pulse mode .....	272
Figure 14-117 Clearing CxORAW(PWM mode A) by EXT input .....	272
Figure 14-118 Complementary output with dead-time insertion .....	273
Figure 14-119 TMR output control .....	274
Figure 14-120 Example of TMR brake function.....	274
Figure 14-121 Example of reset mode .....	275
Figure 14-122 Example of suspend mode .....	275
Figure 14-123 Example of trigger mode .....	275
Figure 15-1 Window watchdog block diagram .....	291
Figure 15-2 Window watchdog timing diagram .....	292
Figure 16-1 WDT block diagram.....	294
Figure 17-1 ERTC block diagram .....	297
Figure 18-1 ADC1 block diagram .....	311
Figure 18-2 ADC basic operation process.....	312
Figure 18-3 ADC power-on and calibration .....	313
Figure 18-4 Sequence mode .....	314
Figure 18-5 Preempted group auto conversion mode.....	315
Figure 18-6 Repetition mode .....	315
Figure 18-7 Partition mode .....	316
Figure 18-8 Data alignment .....	316
Figure 19-1 Block Diagram of Comparator.....	327
Figure 19-2 Glitch filter timing when H_PULSE_CNT=1 and L_PULSE_CNT =0 .....	329
Figure 19-3 Glitch filter timing when H_PULSE_CNT=2 and L_PULSE_CNT =1 .....	329
Figure 21-1 IRTMR block diagram .....	335

## List of tables

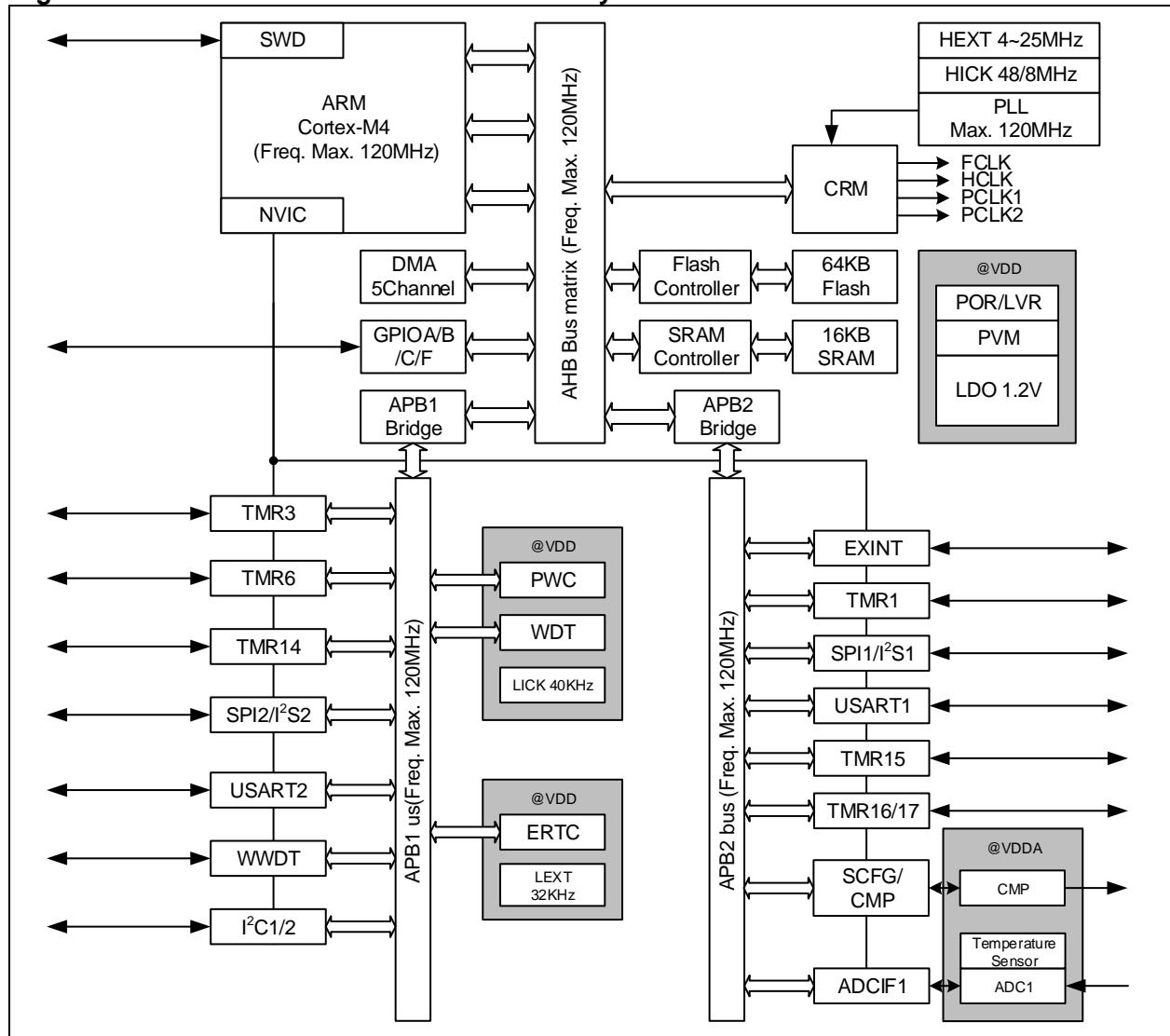
Table 1-1 Bit-band address mapping in SRAM .....	28
Table 1-2 Bit-band address mapping in the peripheral area .....	29
Table 1-3 AT32F421 series vector table .....	29
Table 1-4 List of abbreviations for registers.....	33
Table 1-5 List of abbreviations for registers.....	33
Table 2-1 Flash memory organization (64 KB).....	35
Table 2-2 Flash memory organization (32 KB).....	35
Table 2-3 Flash memory organization (16 KB).....	35
Table 2-4 Peripheral boundary address .....	36
Table 3-1 PW register map and reset values .....	43
Table 4-1 CRM register map and reset values .....	49
Table 5-1 Flash memory architecture(64 K) .....	61
Table 5-2 Flash memory architecture(32 K) .....	61
Table 5-3 Flash memory architecture(16 K) .....	61
Table 5-4 User system data area.....	62
Table 5-5 Flash memory access limit .....	70
Table 5-6 Flash memory interface—Register map and reset value.....	73
Table 6-1 Multiplexed function configuration for port A using GPIO_A MUX* register.....	86
Table 6-2 Multiplexed function configuration for port B using GPIO_B MUX* register.....	87
Table 6-3 Multiplexed function configuration for port F using GPIO_F MUX* register.....	87
Table 6-4 Hardware preemption .....	88
Table 6-5 GPIO register map and reset values .....	88
Table 8-1 External interrupt/Event controller register map and reset value .....	97
Table 9-1 DMA error event.....	102
Table 9-2 DMA interrupt requests .....	102
Table 9-3 DMA requests for each channel .....	102
Table 9-4 DMA register map and reset value .....	103
Table 10-1 CRC register map and reset value .....	108
Table 11-1 I <sup>2</sup> C register map and reset values .....	124
Table 12-1 Baud rate calculation error .....	139
Table 12-2 Data sampling over start bit and noise detection .....	142
Table 12-3 Data sampling over valid data and noise detection.....	142
Table 12-4 USART interrupt request .....	143
Table 12-5 USART register map and reset value .....	144
Table 13-1 Audio frequency precision using system clock.....	163
Table 13-2 SPI register map and reset value .....	167
Table 14-1 TMR functional comparison.....	172
Table 14-2 TMR6 register map and reset value .....	175
Table 14-3 TMR3 internal trigger connection .....	181
Table 14-4 Counting direction versus encoder signals.....	185
Table 14-5 TMR3 register map and reset value .....	193
Table 14-6 Standard CxOUT channel output control bit.....	201
Table 14-7 TMR14 register map and reset value .....	209

Table 14-8 Standard CxOUT channel output control bit.....	213
Table 14-9 TMR15 internal trigger connection .....	218
Table 14-10 TMR15 register map and reset value .....	229
Table 14-11 Complementary output channel CxOUT and CxCOUT control bits with brake function .....	237
Table 14-12 TMR16 and TMR17 register map and reset value .....	248
Table 14-13 Complementary output channel CxOUT and CxCOUT control bits with brake function.....	254
Table 14-14 TMR1 internal trigger connection .....	262
Table 14-15 Counting direction versus encoder signals.....	266
Table 14-16 TMR1 register map and reset value .....	276
Table 14-17 Complementary output channel CxOUT and CxCOUT control bits with brake function.....	286
Table 15-1 Minimum and maximum timeout value when PCLK1=72 MHz .....	292
Table 15-2 WWDT register map and reset value .....	292
Table 16-1 WDT timeout period (LICK=40kHz).....	295
Table 16-2 WDT register and reset value .....	295
Table 17-1 ERTC register configuration .....	298
Table 17-2 ERTC low-power mode wakeup .....	302
Table 17-3 Interrupt control bits .....	302
Table 17-4 ERTC register map and reset values .....	302
Table 18-1 Trigger sources for ADC1 .....	313
Table 18-2 ADC register map and reset values.....	317
Table 19-1 CMP register map and reset values .....	329
Table 22-1 DEBUG register address and reset value .....	336

# 1 System architecture

AT32F421 series microcontrollers incorporates a 32-bit ARM® Cortex®-M4 processor core, multiple 16-bit and 32-bit timers, DMA controller, ERTC, communication interfaces such as SPI, I2C, USART, CMP, 12-bit ADC, programmable voltage monitor (PVM) and other peripherals. Cortex®-M4 processor supports enhanced high-performance DSP instruction set, including extended single-cycle 16-bit/32-bit multiply accumulator (MAC), dual 16-bit MAC instructions, optimized 8-bit/16-bit SIMD operation and saturation operation instructions, as shown in [Figure 1-1](#):

Figure 1-1 AT32F421 Series microcontrollers system architecture



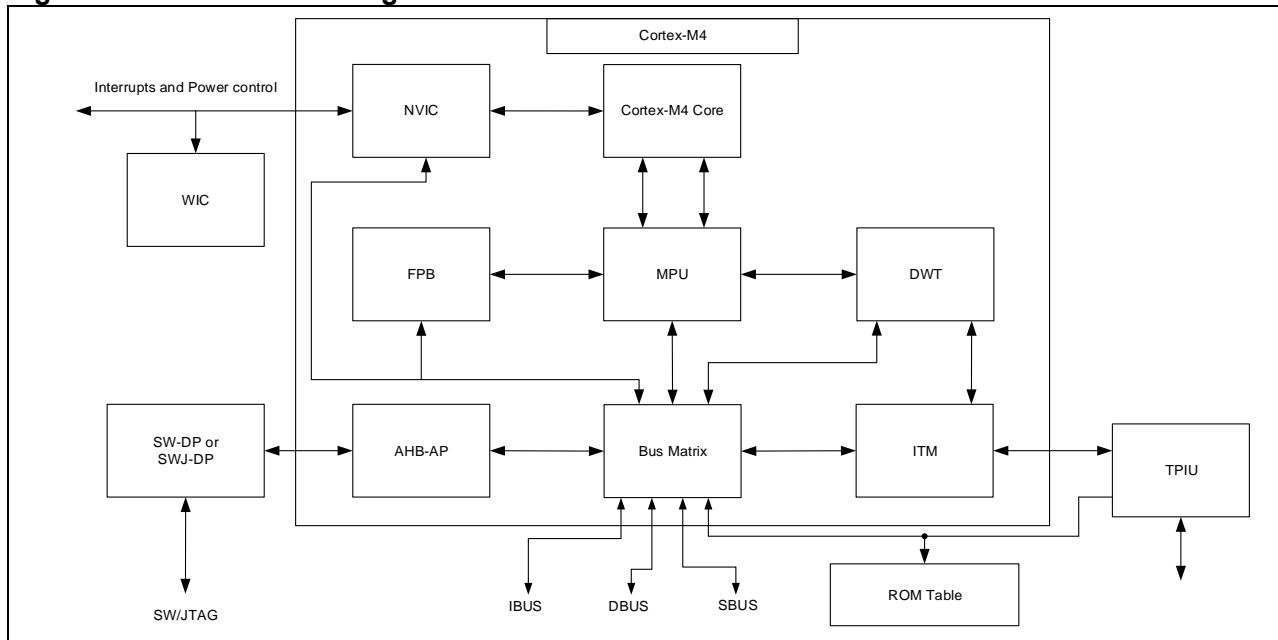
## 1.1 System overview

### 1.1.1 ARM Cortex™-M4 processor

Cortex®-M4 processor is a low-power consumption processor featuring low gate count, low interrupt latency, and low-cost debug. It supports DSP instruction set, particularly applicable to deep-embedded applications that require quicker response to interrupts. Cortex®-M4 processor is based on ARMv7-M architecture, supporting both Thumb instruction set and DSP instruction set.

**Figure 1-2** shows the internal block diagram of Cortex®-M4 processor. Please refer to *ARM Cortex® -M4 Technical Reference Manual* for more information.

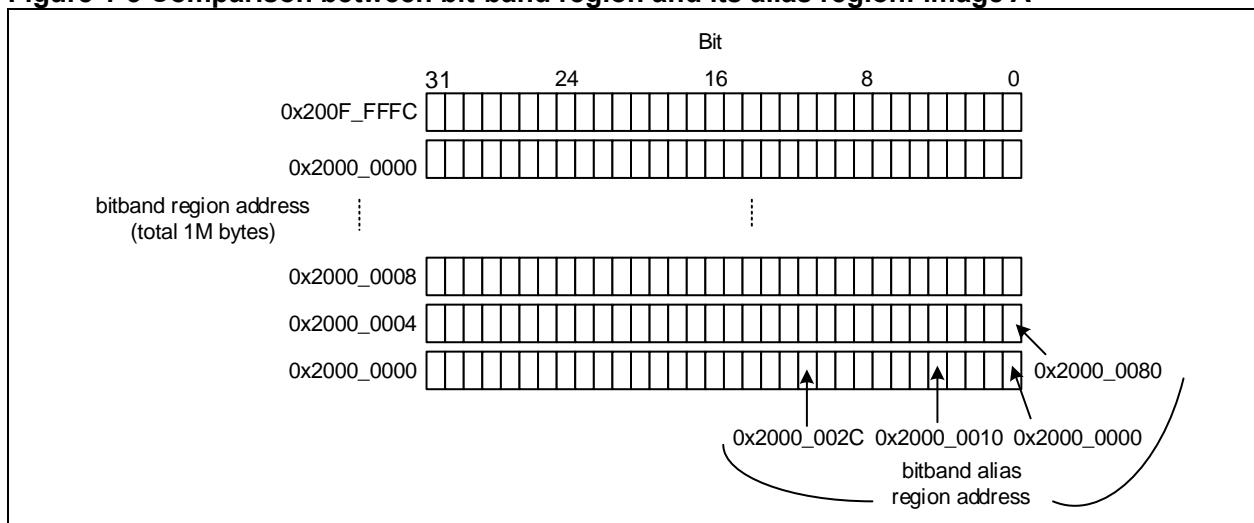
**Figure 1-2 Internal block diagram of Cortex®-M4**

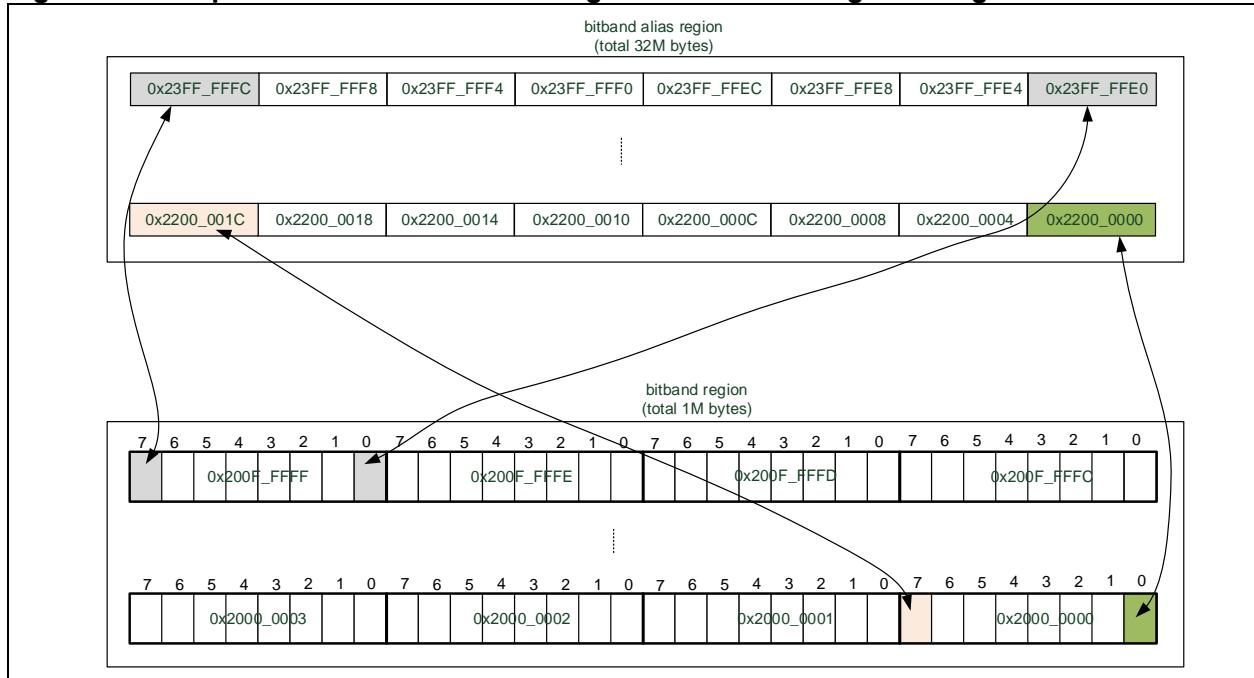


### 1.1.2 Bit band

With the help of bit-band, read and write access to a single bit can be performed in the ordinary way of loading and storing. The Cortex®-M4 memory includes two bit-band regions: the least significant 1M bytes of SRAM and the least significant 1Mbytes of peripherals. In addition to access to bit-band addresses, their respective bit-band alias region can be used to access to any bit in these two bit-band regions. The bit-band alias region transforms each bit in it into a 32-bit word. Thus, accessing to an address in an alias region has the same effect as read-modify-write operation on the targeted bit in a bit-band region.

**Figure 1-3 Comparison between bit-band region and its alias region: image A**



**Figure 1-4 Comparison between bit-band region and its alias region: image B**

Bit-band region: address region for bit-band operations

Bit-band alias region: access to the alias region has the same effect as read-modify-write operation on the bit-band region

Each bit in a bit-band region is mapped into a word (LSB) in an alias region. When accessing to the address in a bit-band alias region, such address is transformed into a bit-band address first. For a read operation, read one word in the bit-band region, and then move the targeted bit to the right to LSB before returning LSB. For a write operation, first move the targeted bit to the left to the corresponding bit number, then perform a read-modify-write operation on bit level.

The address ranges of two memories supporting bit-band operations:

The lowest 1 Mbyte of the SRAM: 0x2000\_0000~0x200F\_FFFF

The lowest 1 Mbyte of the peripherals: 0x4000\_0000~0x400F\_FFFF

For a bit in the SRAM bit-band region, if the byte address is A, the bit number is n ( $0 \leq n \leq 7$ ), then the alias address where the bit is:

$$\text{AliasAddr} = 0x2200\_0000 + (A - 0x2000\_0000) * 32 + n * 4$$

For a bit in the peripheral bit-band region, if the byte address is A, the bit number is n ( $0 \leq n \leq 7$ ), then the alias address where the bit is:

$$\text{AliasAddr} = 0x4200\_0000 + (A - 0x4000\_0000) * 32 + n * 4$$

**Table 1-1** shows the mapping between bit-band region and alias region in SRAM:

**Table 1-1 Bit-band address mapping in SRAM**

Bit-band region	Equivalent alias address
0x2000_0000.0	0x2200_0000.0
0x2000_0000.1	0x2200_0004.0
0x2000_0000.2	0x2200_0008.0
...	...
0x2000_0000.31	0x2200_007C.0
0x2000_0004.0	0x2200_0080.0
0x2000_0004.1	0x2200_0084.0

0x2000_0004.2	0x2200_0088.0
...	...
0x200F_FFFC.31	0x23FF_FFFC.0

**Table 1-2** shows the mapping between bit-band region and alias region in the peripheral area:

**Table 1-2 Bit-band address mapping in the peripheral area**

Bit-band region	Equivalent alias address
0x4000_0000.0	0x4200_0000.0
0x4000_0000.1	0x4200_0004.0
0x4000_0000.2	0x4200_0008.0
...	...
0x4000_0000.31	0x4200_007C.0
0x4000_0004.0	0x4200_0080.0
0x4000_0004.1	0x4200_0084.0
0x4000_0004.2	0x4200_0088.0
...	...
0x400F_FFFC.31	0x43FF_FFFC.0

In addition, bit-band operations can also simplify jump process. When jump operation is based on a bit level, the previous steps are:

- Read the whole register
- Mask the undesired bits
- Compare and jump

For now, you just need do:

- Read the bit status from the bit-band alias region
- Compare and jump

Apart from making code more concise, its important function is also reflected in multi-task environment. When it comes to multiple tasks, it turns the read-modify-write operations into a hardware-supported atomic operation to avoid the scenario where the read-modify-write operation is disrupted, resulting in disorder.

### 1.1.3 Interrupt and exception vectors

**Table 1-3 AT32F421 series vector table**

Pos.	Priority Type	Name	Description	Address
-	-	-	Reserved	0x0000_0000
-3	Fixed	Reset	Reset	0x0000_0004
-2	Fixed	NMI	Non maskable interrupt CRM clock fail detector (CFD) is linked to NMI vector	0x0000_0008
-1	Fixed	HardFault	All class of fault	0x0000_000C
0	Configurable	MemoryManage	Memory management	0x0000_0010
1	Configurable	BusFault	Pre-fetch fault, memory access fault	0x0000_0014

2	Configurable	UsageFault	Undefined instruction or illegal state	0x0000_0018	
-	-	-	Reserved	0x0000_001C ~0x0000_002B	
3	Configurable	SVCall	Call system service via SWI instruction	0x0000_002C	
4	Configurable	DebugLENonitor	Debug monitor	0x0000_0030	
-	-	-	Reserved	0x0000_0034	
5	Configurable	PendSV	Pendable request for system service	0x0000_0038	
6	Configurable	SysTick	System tick timer	0x0000_003C	
0	7	Configurable	WWDT	Window watchdog timer interrupt	0x0000_0040
1	8	Configurable	PVM	PVM interrupt linked to EXINT line 16	0x0000_0044
2	9	Configurable	ERTC	ERTC interrupt linked to EXINT line 16/17	0x0000_0048
3	10	Configurable	FLASH	Flash global interrupt	0x0000_004C
4	11	Configurable	CRM	Clock and Reset manage (CRM) interrupt	0x0000_0054
5	12	Configurable	EXINT1_0	EXINT line 1_0 interrupt	0x0000_0054
6	13	Configurable	EXINT3_2	EXINT line 3_2 interrupt	0x0000_0058
7	14	Configurable	EXINT15_4	EXINT line 15_4 interrupt	0x0000_005C
8	15	Configurable	-	Reserved	0x0000_0060
9	16	Configurable	DMA channel 1	DMA channel 1 global interrupt	0x0000_0064
10	17	Configurable	DMA channel 3_2	DMA channel 3_2 global interrupt	0x0000_0068
11	18	Configurable	DMA channel 5_4	DMA channel 5_4 global interrupt	0x0000_006C
12	19	Configurable	ADC_CMP	ADC and CMP global interrupt	0x0000_0070
13	20	Configurable	TMR1_BRK TMR1_UP TMR1_TRG TMR1_COM	TMR1 interrupt	0x0000_0074
14	21	Configurable	TMR1_CH	TMR1 capture compare interrupt	0x0000_0078
15	22	Configurable	-	Reserved	0x0000_007C
16	23	Configurable	TMR3	TMR3 global interrupt	0x0000_0080
17	24	Configurable	TMR6	TMR6 global interrupt	0x0000_0084
18	25	Configurable	-	Reserved	0x0000_0088
19	26	Configurable	TMR14	TMR14 global interrupt	0x0000_008C
20	27	Configurable	TMR15	TMR15 global interrupt	0x0000_0090
21	28	Configurable	TMR16	TMR16 global interrupt	0x0000_0094
22	29	Configurable	TMR17	TMR17 global interrupt	0x0000_0098
23	30	Configurable	I2C1_EVT	I <sup>2</sup> C1 event interrupt	0x0000_009C
24	31	Configurable	I2C2_EVT	I <sup>2</sup> C2 event interrupt	0x0000_00A0
25	32	Configurable	SPI1	SPI1 global interrupt	0x0000_00A4

26	33	Configurable	SPI2	SPI2 global interrupt	0x0000_00A8
27	34	Configurable	USART1	USART1 global interrupt	0x0000_00AC
28	35	Configurable	USART2	USART2 global interrupt	0x0000_00B0
29	36	Configurable	-	Reserved	0x0000_00B4
30	37	Configurable	-	Reserved	0x0000_00B8
31	38	Configurable	-	Reserved	0x0000_00BC
32	39	Configurable	I2C1_ERR	I <sup>2</sup> C1 error interrupt	0x0000_00C0
33	40	Configurable	-	Reserved	0x0000_00C4
34	41	Configurable	I2C2_ERR	I <sup>2</sup> C2 error interrupt	0x0000_00C8

## 1.1.4 System Tick (SysTick)

The System Tick is a 24-bit downcounter. It will be reloaded with the initial value automatically when it is decremented to zero. It can generate periodic interrupts, so it is often used as multi-task scheduling counter for embedded operating system, and also to call the periodic tasks for non-embedded system.

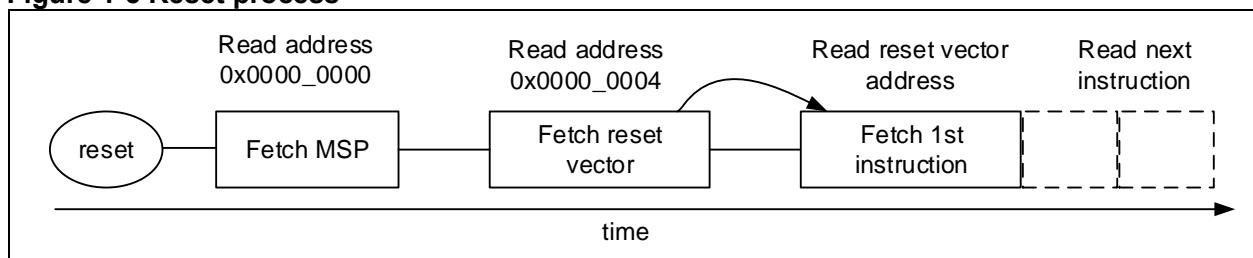
The System Tick calibration value is fixed to 9000, which gives a reference time base of 1 ms when the System Tick clock is set to 9 MHz.

## 1.1.5 Reset

The processor reads the first two words from the CODE memory after a system reset and before program execution.

- Get the initial value of the main stack pointer (MSP) from address 0x0000\_0000
- Get the initial value of the program counter (PC) from address 0x0000\_0004. This value is a reset vector and LSB must be 1. Then take the instructions from the address corresponding to this value.

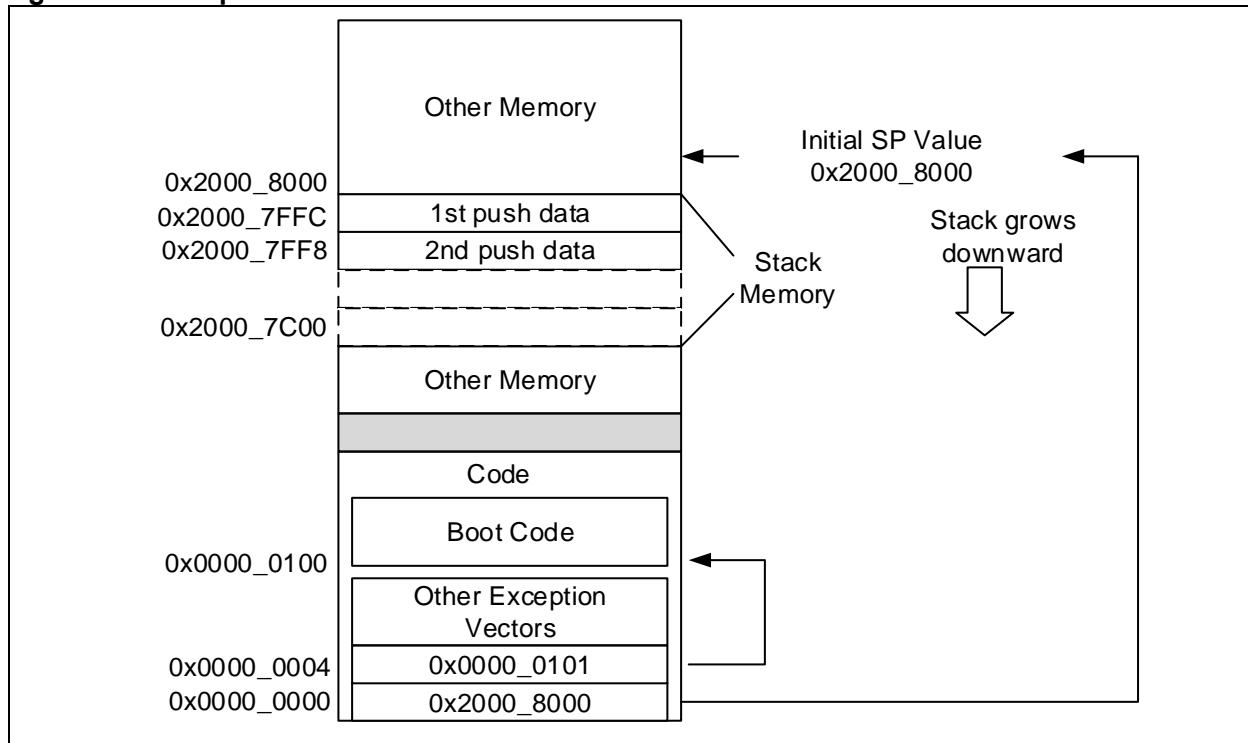
**Figure 1-5 Reset process**



Cortex™-M4 uses a full stack that increases downward, so the initial value of the main stack pointer (MSP) must be the end address of the stack memory plus 1. For example, if the stack area is set between 0x2000\_7C00 and 0x2000\_7FFF, then the initial value of MSP must be defined as 0x2000\_8000.

The vector table starts following the initial value of MSP. Cortex™-M4 operates in Thumb state, and thus each value in the vector table must set the LSB to 1. In [Figure 1-6](#), 0x0000\_0101 is used to represent 0x0000\_0100. Once the instruction at 0x0000\_0100 is executed, the program starts running formally. Before that, it is mandatory to initialize MSP, because the first instruction may be interrupted by NMI or other faults before being executed. After the completion of MSP initialization, it is ready to prepare stack room for its service routines.

Figure 1-6 Example of MSP and PC initialization



In the AT32F421 series, the main Flash memory, Boot code or SRAM can be remapped to the code area between 0x0000\_0000 and 0x07FF\_FFFF. nBOOT1 corresponds to the system configuration byte nBOOT1 of the User System Data area. BOOT0 and nBOOT1 are used to determine the specific memory from which CODE starts.

{BOOT1, BOOT0}=00/10, CODE starts from the main Flash memory

{BOOT1, BOOT0}=11, CODE starts from Boot code

{BOOT1, BOOT0}=01, CODE starts from SRAM

After a system reset or when leaving from Standby mode, the pin values of both nBOOT1 and BOOT0 will be relatched.

When booting from SRAM, BOOT status will be latched. In this case, it is not possible to load a new boot mode after system reset, and it is mandatory to perform a power-on reset in order to select a new boot mode.

Boot code memory contains an embedded boot loader program that allows to reprogram Flash through USART1 or USART2 interface. Besides, it provides extra firmware including communication protocol stacks that can be called by software developers using API.

## 1.2 List of abbreviations for registers

**Table 1-4 List of abbreviations for registers**

Register type	Description
rw	Software can read and write to this bit.
ro	Software can only read this bit.
wo	Software can only write to the bit. Reading it returns its reset value.
rrc	Software can read this bit. Reading this bit automatically clears it.
rw0c	Software can read this bit and clear it by writing 0. Writing 1 has no effect on this bit.
rw1c	Software can read this bit and clear it by writing 1. Writing 0 has no effect on this bit.
rw1s	Software can read this bit and set it by writing 1. Writing 0 has no effect on this bit.
tog	Software can read this bit and toggle it by writing 1. Writing 0 has no effect on this bit.
rwt	Software can read this bit. Writing any value will trigger an event.
resd	Reserved.

## 1.3 Device characteristics information

**Table 1-5 List of abbreviations for registers**

Register abbr.	Base address	Reset value
F_SIZE	0x1FFF F7E0	0xFFFF
UID[31: 0]	0x1FFF F7E8	0xFFFF XXXX
UID[63: 32]	0x1FFF F7EC	0xFFFF XXXX
UID[95: 64]	0x1FFF F7F0	0xFFFF XXXX

### 1.3.1 Flash memory size register

This register contains the information about Flash memory size.

Bit	Abbr.	Reset value	Type	Description
Bit 15: 0	F_SIZE	0xFFFF	ro	Flash size, in terms of KByte For example: 0x0040 = 64 KByte

### 1.3.2 Device electronic signature

The device electronic signature contains the memory size and the unique device ID (96 bits). It is stored in the information block of the Flash memory. The 96-bit ID is unique for any device, and cannot be altered by users. It can be used for the following purposes:

- Serial number: such as USB string serial number
- Part of security keys

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	UID[31: 0]	0xFFFF XXXX	ro	UID for bit 31 to bit 0
Bit 31: 0	UID[63: 32]	0xFFFF XXXX	ro	UID for bit 63 to bit 32

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	UID[95: 64]	0xXXXX XXXX	ro	UID for bit 95 to bit 64

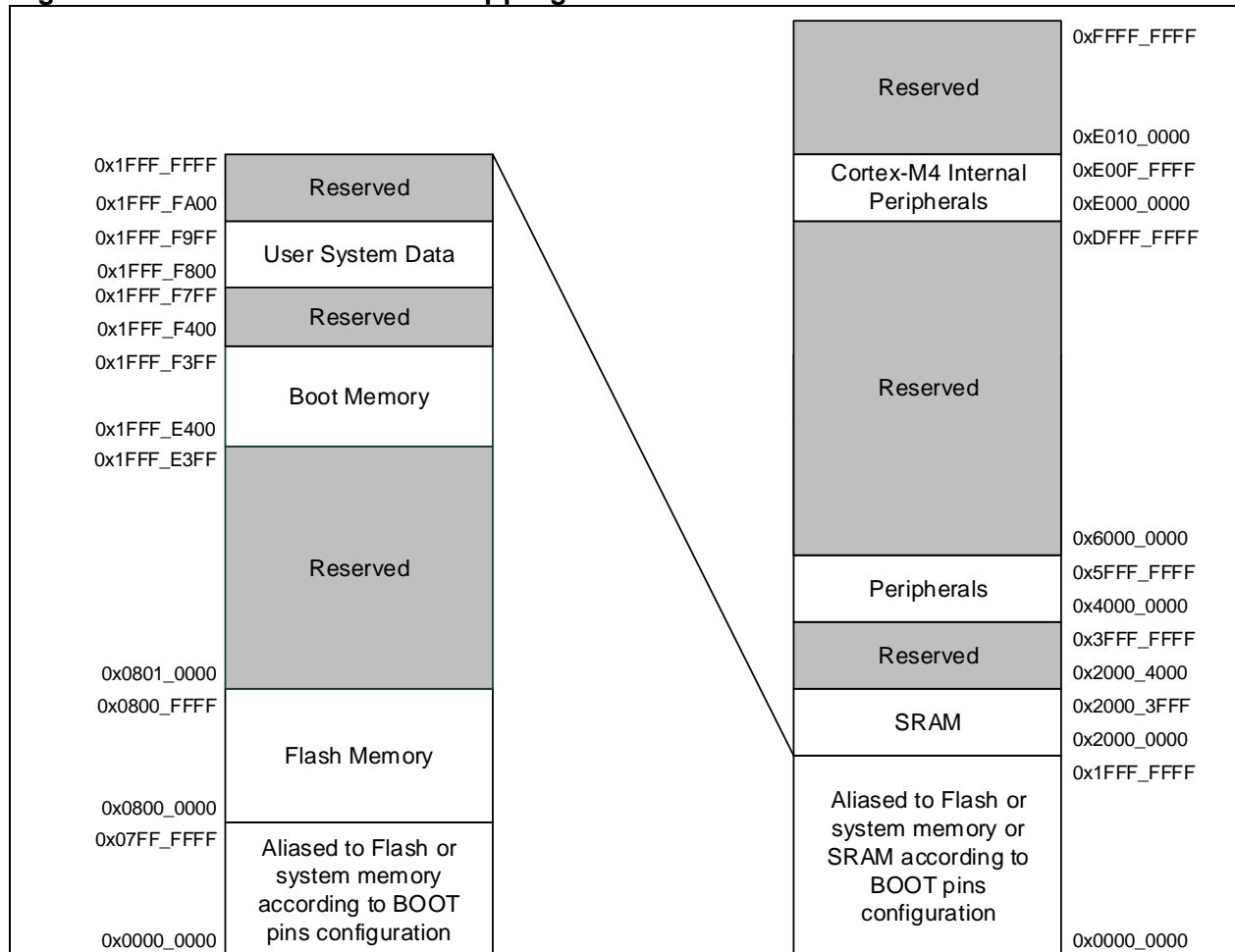
Note: UID[95:88] is series ID, which is 0x09 for AT32F421.

## 2 Memory resources

### 2.1 Internal memory address map

Internal memory contains program memory (Flash), data memory (SRAM), peripheral registers and core registers. Their respective address mapping are shown in [Figure 2-1](#).

**Figure 2-1** AT32F421 address mapping



## 2.2 Flash memory

AT32F421 series provide up to 64 KB of on-chip Flash memory, supporting a single-cycle 32-bit read operation.

Refer to [Chapter 5](#) for more details about Flash memory controller and register configuration.

### Flash memory organization (64 KB)

The main memory contains only bank 1 (64 Kbytes), including 64 sectors, 1 Kbyte per sector.

**Table 2-1 Flash memory organization (64 KB)**

Block	Name	Address range
Main memory Bank1 (64 KB)	Sector 0	0x0800 0000 – 0x0800 03FF
	Sector 1	0x0800 0400 – 0x0800 07FF
	Sector 2	0x0800 0800 – 0x0800 0BFF
	...	...
	Sector 63	0x0800 FC00 – 0x0800 FFFF
	4K Boot loader	0x1FFF E400 – 0x1FFF F3FF
Information block	512B user system data	0x1FFF F800 – 0x1FFF F9FF

### Flash memory organization (32 KB)

The main memory contains only bank 1 (32 Kbytes), including 32 sectors, 1 Kbyte per sector.

**Table 2-2 Flash memory organization (32 KB)**

Block	Name	Address range
Main memory Bank1 (32 KB)	Sector 0	0x0800 0000 – 0x0800 03FF
	Sector 1	0x0800 0400 – 0x0800 07FF
	Sector 2	0x0800 0800 – 0x0800 0BFF
	...	...
	Sector 31	0x0800 7C00 – 0x0800 7FFF
	4K Boot loader	0x1FFF E400 – 0x1FFF F3FF
Information block	512B User system data	0x1FFF F800 – 0x1FFF F9FF

**Table 2-3 Flash memory organization (16 KB)**

Block	Name	Address range
Main memory Bank1 (16 KB)	Sector 0	0x0800 0000 – 0x0800 03FF
	Sector 1	0x0800 0400 – 0x0800 07FF
	Sector 2	0x0800 0800 – 0x0800 0BFF
	...	...
	Sector 15	0x0800 3C00 – 0x0800 3FFF
	4K Boot loader	0x1FFF E400 – 0x1FFF F3FF
Information block	512B User system data	0x1FFF F800 – 0x1FFF F9FF

## 2.3 SRAM memory

The AT32F421 series embeds a 16-KB on-chip SRAM which starts at the address 0x2000\_0000. It can be accessed by bytes, half-words (16 bit) or words (32 bit).

## 2.4 Peripheral address map

**Table 2-4 Peripheral boundary address**

Bus	Boundary address	Peripherals
AHB	0x6000 0000 - 0xFFFF FFFF	Reserved
	0x5004 0000 - 0x5FFF FFFF	Reserved
	0x5000 0000 - 0x5003 FFFF	Reserved
	0x4800 1800 – 0x4FFF FFFF	Reserved
	0x4800 1400 - 0x4800 17FF	GPIOF
	0x4800 0C00 - 0x4800 13FF	Reserved
	0x4800 0800 - 0x4800 0BFF	GPIOC
	0x4800 0400 - 0x4800 07FF	GPIOB
	0x4800 0000 - 0x4800 03FF	GPIOA
	0x4002 8000 - 0x47FF FFFF	Reserved
	0x4002 3400 - 0x4002 7FFF	Reserved
	0x4002 3000 - 0x4002 33FF	CRC
	0x4002 2000 - 0x4002 23FF	Flash memory interface (FLASH)
	0x4002 1400 - 0x4002 1FFF	Reserved
	0x4002 1000 - 0x4002 13FF	Clock and reset manage (CRM)
	0x4002 0800 - 0x4002 0FFF	Reserved
	0x4002 0400 - 0x4002 07FF	Reserved
	0x4002 0000 - 0x4002 03FF	DMA
APB2	0x4001 8400 - 0x4001 7FFF	Reserved
	0x4001 8000 - 0x4001 83FF	Reserved
	0x4001 7C00 - 0x4001 7FFF	Reserved
	0x4001 7800 - 0x4001 7BFF	Reserved
	0x4001 7400 - 0x4001 77FF	Reserved
	0x4001 7000 - 0x4001 73FF	Reserved
	0x4001 6C00 - 0x4001 6FFF	Reserved
	0x4001 6800 - 0x4001 6BFF	Reserved
	0x4001 6400 - 0x4001 67FF	Reserved
	0x4001 6000 - 0x4001 63FF	Reserved
	0x4001 5C00 - 0x4001 5FFF	Reserved
	0x4001 5800 - 0x4001 5BFF	Reserved
	0x4001 5400 - 0x4001 57FF	Reserved
	0x4001 5000 - 0x4001 53FF	Reserved
	0x4001 4C00 - 0x4001 4FFF	Reserved
	0x4001 4800 - 0x4001 4BFF	TMR17 timer
	0x4001 4400 - 0x4001 47FF	TMR16 timer
	0x4001 4000 - 0x4001 43FF	TMR15 timer

Bus	Boundary address	Peripherals
APB1	0x4001 3C00 - 0x4001 4BFF	Reserved
	0x4001 3800 - 0x4001 3BFF	USART1
	0x4001 3400 - 0x4001 37FF	Reserved
	0x4001 3000 - 0x4001 33FF	SPI1/I <sup>2</sup> S1
	0x4001 2C00 - 0x4001 2FFF	TMR1 timer
	0x4001 2800 - 0x4001 2BFF	Reserved
	0x4001 2400 - 0x4001 27FF	ADC
	0x4001 2000 - 0x4001 23FF	Reserved
	0x4001 1C00 - 0x4001 1FFF	Reserved
	0x4001 1800 - 0x4001 1BFF	Reserved
	0x4001 1400 - 0x4001 17FF	Reserved
	0x4001 1000 - 0x4001 13FF	Reserved
	0x4001 0C00 - 0x4001 0FFF	Reserved
	0x4001 0800 - 0x4001 0BFF	Reserved
	0x4001 0400 - 0x4001 07FF	EXINT
	0x4001 0000 - 0x4001 03FF	SCFG/CMP
	0x4000 8000 - 0x4000 FFFF	Reserved
	0x4000 7C00 - 0x4000 7FFF	Reserved
	0x4000 7800 - 0x4000 7BFF	Reserved
	0x4000 7400 - 0x4000 77FF	Reserved
	0x4000 7000 - 0x4000 73FF	Power control (PWC)
	0x4000 6800 - 0x4000 6BFF	Reserved
	0x4000 6400 - 0x4000 67FF	Reserved
	0x4000 6000 - 0x4000 63FF	Reserved
	0x4000 5C00 - 0x4000 5FFF	Reserved
	0x4000 5800 - 0x4000 5BFF	I <sup>2</sup> C2
	0x4000 5400 - 0x4000 57FF	I <sup>2</sup> C1
	0x4000 5000 - 0x4000 53FF	Reserved
	0x4000 4C00 - 0x4000 4FFF	Reserved
	0x4000 4800 - 0x4000 4BFF	Reserved
	0x4000 4400 - 0x4000 47FF	USART2
	0x4000 4000 - 0x4000 43FF	Reserved
	0x4000 3C00 - 0x4000 3FFF	Reserved
	0x4000 3800 - 0x4000 3BFF	SPI2/I <sup>2</sup> S2
	0x4000 3400 - 0x4000 37FF	Reserved
	0x4000 3000 - 0x4000 33FF	Watchdog (WDT)
	0x4000 2C00 - 0x4000 2FFF	Window watchdog (WWDT)
	0x4000 2800 - 0x4000 2BFF	ERTC
	0x4000 2400 - 0x4000 27FF	Reserved
	0x4000 2000 - 0x4000 23FF	TMR14 timer

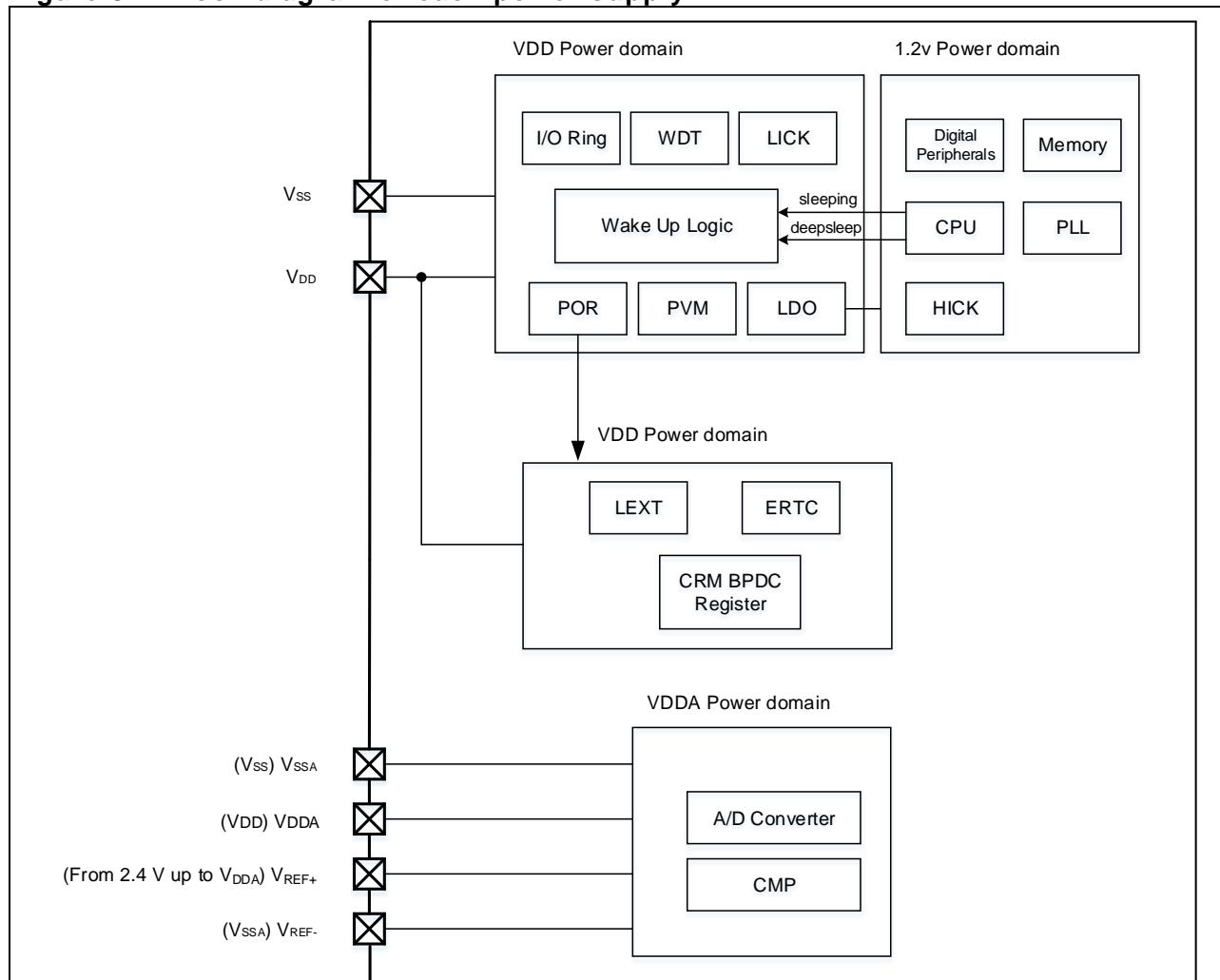
Bus	Boundary address	Peripherals
	0x4000 1C00 - 0x4000 1FFF	Reserved
	0x4000 1800 - 0x4000 1BFF	Reserved
	0x4000 1400 - 0x4000 17FF	Reserved
	0x4000 1000 - 0x4000 13FF	TMR6 timer
	0x4000 0C00 - 0x4000 0FFF	Reserved
	0x4000 0800 - 0x4000 0BFF	Reserved
	0x4000 0400 - 0x4000 07FF	TMR3 timer
	0x4000 0000 - 0x4000 03FF	Reserved

## 3 Power control (PWC)

### 3.1 Introduction

For the AT32F421 series, its operating voltage supply is 2.6 V ~ 3.6 V, with a temperature range of -40~+105 °C. To reduce power consumption, this series provides three types of power saving modes, including Sleep, Deepsleep and Standby modes so as to achieve the best trade-off among the conflicting demands of CPU operating time, speed and power consumption. The AT32F421 series has two power domains—VDD/VDDA domain and 1.2 V domain. The VDD/VDDA domain is supplied directly by external power, while, the 1.2 V domain is powered by an embedded LDO in the VDD/VDDA domain.

**Figure 3-1 Block diagram of each power supply**



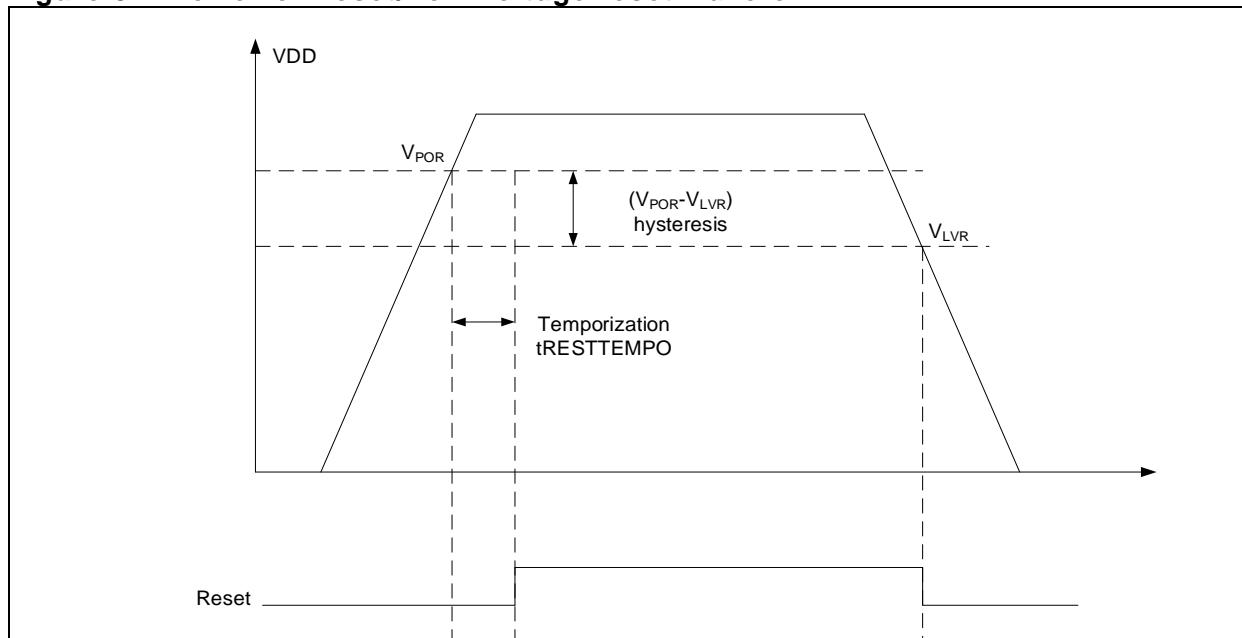
### 3.2 Main Features

- Two power domains: VDD/VDDA domain and 1.2 V core domain
- Three types of power saving modes: Sleep mode, Deepsleep mode, and Standby mode
- Internal voltage regulator supplies 1.2 V voltage for the core domain
- Power voltage detector is provided to generate an interrupt or even when the supply voltage is lower or higher than a programmed threshold
- VDD/VDDA applies separated digital and analog module to reduce noise on external power

### 3.3 POR/LVR

A POR analog module embedded in the VDD/VDDA domain is used to generate a power reset. The power reset signal is released at  $V_{POR}$  when the VDD is increased from 0 V to the operating voltage, or it is triggered at  $V_{LVR}$  when the VDD drops from the operating voltage to 0 V. During the power-on reset period, the reset signal release has certain amount of time delay compared to VDD boost process. At the same time, hysteresis occurs in power-on reset (POR) and low voltage reset (LVR).

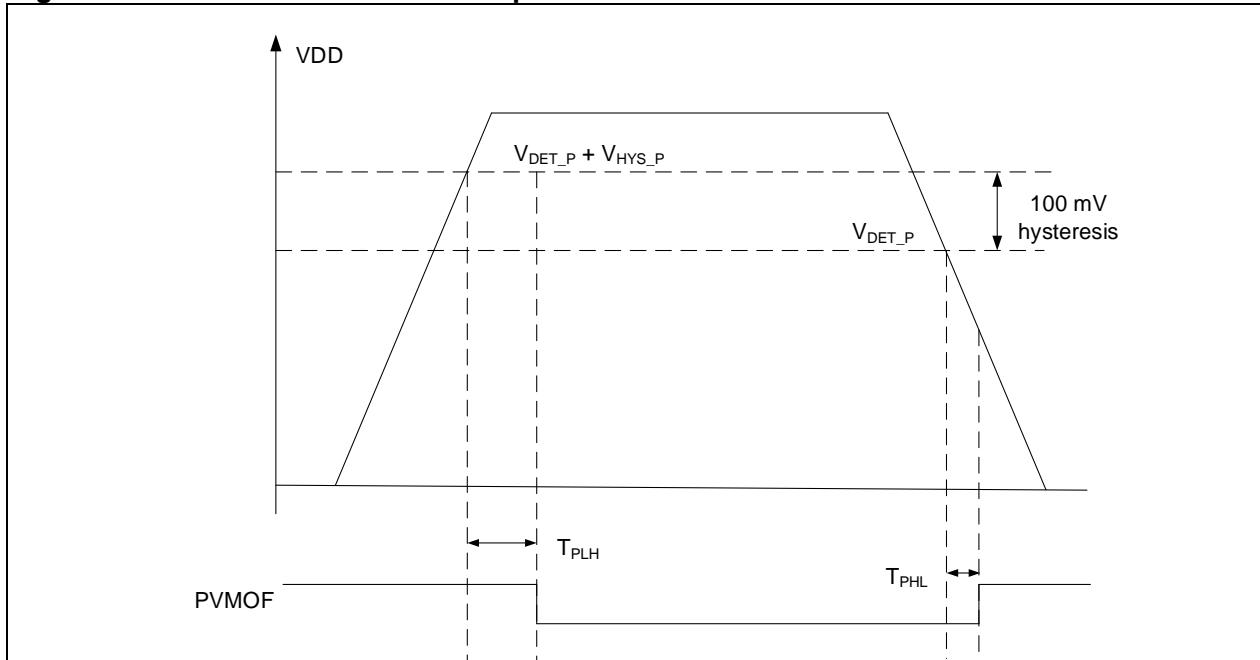
**Figure 3-2 Power-on reset/Low voltage reset waveform**



### 3.4 Power voltage monitor (PVM)

The PVM is used to monitor the power supply variations. It is enabled by setting the PVMEN bit in the power control register (PWC\_CTRL), and the threshold value for voltage monitor is selected with the PVMSEL[2: 0] bit.

After PVM is enabled, the comparison result between VDD and the programmed threshold is indicated by the PVMOF bit in the PWC\_CTRLSTS register, with the hysteresis voltage VHYS\_P being 100 mv. The PVM interrupt will be generated (due to PVMOF level change ) through the EXTI line 16 when VDD rises above the PVM threshold.

**Figure 3-3 PVM threshold and output**

## 3.5 Power domain

### 1.2 V domain

1.2 V core domain includes a CPU core, SRAM, embedded digital peripherals and Phase Locked Loop (PLL). Such power domain is supplied by LDO (voltage regulator).

### VDD/VDDA domain

VDD/VDDA domain includes VDD domain and VDDA domain. The VDD domain contains I/O circuit, power-saving mode wakeup circuit, watchdog timer (WDT), power-on reset/low voltage reset (POR/LVR), LDO, ERTC, LEXT oscillator and all PAD circuits. The VDDA domain contains an ADC (AD converter), temperature sensor and so on.

Typically, to ensure a better accuracy of ADC at a low voltage, the digital circuit is supplied by VDD while the analog circuit is powered by VDDA. The external reference voltage VREF+ and VREF- are connected to the VDDA pin and VSSA pin, respectively.

## 3.6 Power saving modes

When the CPU does not need to be kept running, there are three types of low-power modes available (Sleep mode, Deepsleep mode and Standby mode) to save power. Users can select the mode that gives the best compromise according to the low-power consumption, short startup time, and available wakeup sources. In addition, the power consumption in Run mode can be reduced by slowing down the system clocks or gating the clocks to the APB and AHB peripherals when they are not used.

### Sleep mode

The Sleep mode is entered by executing WFI or WFE instruction. There are two options to select the Sleep mode entry mechanism through the SLEEPONEXIT bit in the Cortex® -M4 system control register.

#### SLEEP-NOW mode:

When SLEEPDEEP=0 and SLEEPONEXIT=0, the MCU enters Sleep mode as soon as WFI or WFE instruction is executed.

When SLEEPDEEP=0 and SLEEPONEXIT=1, the MCU enters Sleep mode as soon as the system exits the lowest-priority interrupt service routine by executing the WFI instruction.

In Sleep mode, all clocks and LDO work normally except CPU clocks (stopped), and all I/O pins keep the same state as in Run mode. The LDO provides a 1.2 V power (for CPU core, memories and embedded peripherals) as it is in normal power consumption mode.

- 1) If the WFI is executed to enter Sleep mode, any peripheral interrupt can wake up the device from Sleep mode.

- 2) If the WFE is executed to enter Sleep mode, the MCU exits Sleep mode as soon as an event occurs. The wakeup event can be generated by the following:
  - Enabling a peripheral interrupt (it is not enabled in the NVIC) and enabling the SEVONPEND bit. When the MCU resumes, the peripheral interrupt pending bit and NVIC channel pending bit must be cleared.
  - Configuring an internal EXINT line as an event mode to generate a wakeup event.

The wakeup time required for a WFE instruction is the shortest, since no time is wasted on interrupt entry/exit.

### Deepsleep Mode

Deepsleep mode is entered by setting the SLEEPDEEP bit in the Cortex™-M4 system control register and clearing the LPSEL bit in the power control register before WFI or WFE instructions.

The LDO status is selected by setting the VRSEL bit in the power control register (PWC\_CTRL). When VRSEL=0, the LDO works in normal mode. When VRSEL=1, the LDO is set in low-power consumption mode.

Additionally, when VRSEL=1 is set in Deepsleep mode, it means that the voltage regulator is in low-power mode. In this case, it is possible to further reduce power consumption in Deepsleep mode by setting the VREXLPE bit (voltage regulator extra low power mode enable).

In Deepsleep mode, all clocks in 1.2 V domain are stopped, and both HICK and HEXT oscillators are disabled. The LDO supplies power to the 1.2 V domain in normal mode or low-power mode. All I/O pins keep the same state as in Run mode. SRAM and register contents are preserved.

- 1) When the Sleep mode is entered by executing a WFI instruction, the interrupt generated on any external interrupt line in Interrupt mode can wake up the system from Deepsleep mode.
- 2) When the Sleep mode is entered by executing a WFE instruction, the interrupt generated on any external interrupt line in Event mode can wake up the system from Deepsleep mode.

When the MCU exits the Deepsleep mode, the HICK RC oscillator is enabled and selected as a system clock after stabilization. When the LDO operates in low-power mode, an additional wakeup delay is incurred for the reason that the LDO must be stabilized before the system is waken from the Deepsleep mode.

### Standby Mode

Standby mode can achieve the lowest power consumption for the device. In this mode, the LDO is disabled. The whole 1.2 V domain, PLL, HICK and HEXT oscillators are also powered off except VDD/VDDA domain. SRAM and register contents are lost.

The Standby mode is entered by the following procedures:

- Set the SLEEPDEE bit in the Cortex™-M4 system control register
- Set the LPSEL bit in the power control register (PWC\_CTRL)
- Clear the SWEF bit in the power control/status register (PWC\_CTRLSTS)
- Execute a WFI/WFE instruction

In Standby mode, all I/O pins remain in a high-impedance state except reset pins, TAMPER pins that are set as anti-tamper or calibration output, and the wakeup pins enabled.

The MCU leaves the Standby mode when an external reset (NRST pin), an WDT reset, ERTC timestamp, ERTC tamper event and a rising edge on the WKUP pin or the rising edge of an ERTC alarm event occurs.

### Debug mode

By default, the debug connection is lost if the MCU enters Deepsleep mode or Standby mode while debugging. The reason is that the Cortex™-M4 core is no longer clocked. However, the software can be debugged even in the low-power mode by setting some configuration bits in the DEBUG register (DEBUG\_CTRL).

## 3.7 PWC registers

The peripheral registers must be accessed by words (32 bit)

**Table 3-1 PW register map and reset values**

Register abbr.	Offset	Reset value
PWC_CTRL	0x00	0x0000 0000
PWC_CTRLSTS	0x04	0x0000 0000
PWC_CTRL2	0x20	0x0000 00XX

### 3.7.1 Power control register (PWC\_CTRL)

Bit	Name	Reset value	Type	Description
Bit 31: 9	Reserved	0x0000 00	resd	Kept at its default value.
Bit 8	BPWEN	0x0	rw	Battery powered domain write enable 0: Disabled 1: Enabled Note: After reset, BPR is write protected. To write, this bit must be set.
Bit 7: 5	PVMSEL	0x0	rw	Power voltage monitoring boundary select 000: Unused, not configurable 001: 2.3 V 010: 2.4 V 011: 2.5 V 100: 2.6 V 101: 2.7 V 110: 2.8 V 111: 2.9 V
Bit 4	PVMEN	0x0	rw	Power voltage monitoring enable 0: Disabled 1: Enabled
Bit 3	CLSEF	0x0	wo	Clear SEF flag 0: No effect 1: Clear the SEF flag Note: This bit is cleared by hardware after clearing the SEF flag. Reading this bit at any time will return all zero.
Bit 2	CLSWEF	0x0	wo	Clear SWEF flag 0: No effect 1: Clear the SWEF flag Note: Clear the SWEF flag after two system clock cycles. This bit is cleared by hardware after clearing the SWEF flag. Reading this bit at any time will return all zero.
Bit 1	LPSEL	0x0	rw	Low power mode select when Cortex™-M4F deepsleeps 0: Enter DEEPSLEEP mode 1: Enter Standby mode
Bit 0	VRSEL	0x0	rw	Voltage regulator status select in Deepsleep mode 0: Enabled 1: Low-power consumption mode

### 3.7.2 Power control/status register (PWC\_CTRLSTS)

Unlike a standard APB read, an additional APB cycles are needed to read this register.

Bit	Name	Reset value	Type	Description
Bit 31: 15	Reserved	0x000000	resd	Kept at its default value.
Bit 14	SWPEN7	0x0	rw	Standby wake-up pin 7 enable 0: Disabled (this pin is used for general-purpose I/O) 1: Enabled (this pin is forced in input pull-down mode, and no longer used for general-purpose I/O) Note: This bit is cleared by hardware after system reset. In Standby mode, this bit is forced to input pull-up mode, regardless of whether it is enabled or not.
Bit 13	SWPEN6	0x0	rw	Standby wake-up pin 6 enable 0: Disabled (this pin is used for general-purpose I/O) 1: Enabled (this pin is forced in input pull-down mode, and no longer used for general-purpose I/O) Note: This bit is cleared by hardware after system reset. In Standby mode, this bit is forced to input pull-up mode, regardless of whether it is enabled or not.
Bit 12: 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	SWPEN2	0x0	rw	Standby wake-up pin 2 enable 0: Disabled (this pin is used for general-purpose I/O) 1: Enabled (this pin is forced in input pull-down mode, and no longer used for general-purpose I/O) Note: This bit is cleared by hardware after system reset. In Standby mode, this bit is forced to input pull-up mode, regardless of whether it is enabled or not.
Bit 8	SWPEN1	0x0	rw	Standby wake-up pin 1 enable 0: Disabled (this pin is used for general-purpose I/O) 1: Enabled (this pin is forced in input pull-down mode, and no longer used for general-purpose I/O) Note: This bit is cleared by hardware after system reset. In Standby mode, this bit is forced to input pull-up mode, regardless of whether it is enabled or not.
Bit 7: 3	Reserved	0x00	resd	Kept at its default value.
Bit 2	PVMOF	0x0	ro	Power voltage monitoring output flag 0: Power voltage is higher than the threshold 1: Power voltage is lower than the threshold Note: The power voltage monitor is stopped in Standby mode.
Bit 1	SEF	0x0	ro	Standby mode entry flag 0: Device is not in Standby mode 1: Device is in Standby mode Note: This bit is set by hardware (enter Standby mode) and cleared by POR/LVR or by setting the CLSEF bit.
Bit 0	SWEF	0x0	ro	Standby wake-up event flag 0: No wakeup event occurred 1: A wakeup event occurred Note: This bit is set by hardware (on a wakeup event), and cleared by POR/LVR or by setting the CLSWEF bit. A wakeup event is generated by one of the following: When the rising edge on the Standby wakeup pin occurs; When the RTC alarm event occurs; If the Standby wakeup pin is enabled when the Standby wakeup pin level is high.

### 3.7.3 Power control register 2 (PWC\_CTRL2)

Bit	Name	Reset value	Type	Description
Bit 31: 6	Reserved	0x000000	resd	Kept at its default value.
Bit 5	VREXLPPEN	0x0	rw	Voltage regulator extra low power mode enable This bit works with the LPSEL and VRSE bits of the PWC_CTRL register. It is applicable only when VRSEL = 1 and the device enters Deepsleep mode.

---

Bit 4: 0	Reserved	0x0	rw	0: Voltage regulator extra low power mode disabled 1: Voltage regulator extra low power mode enabled Note: To enable extra low-power mode, it is mandatory to set LPSEL and VRSEL bits before setting the VREXL PEN.
				V Kept at its default value. Do not change.

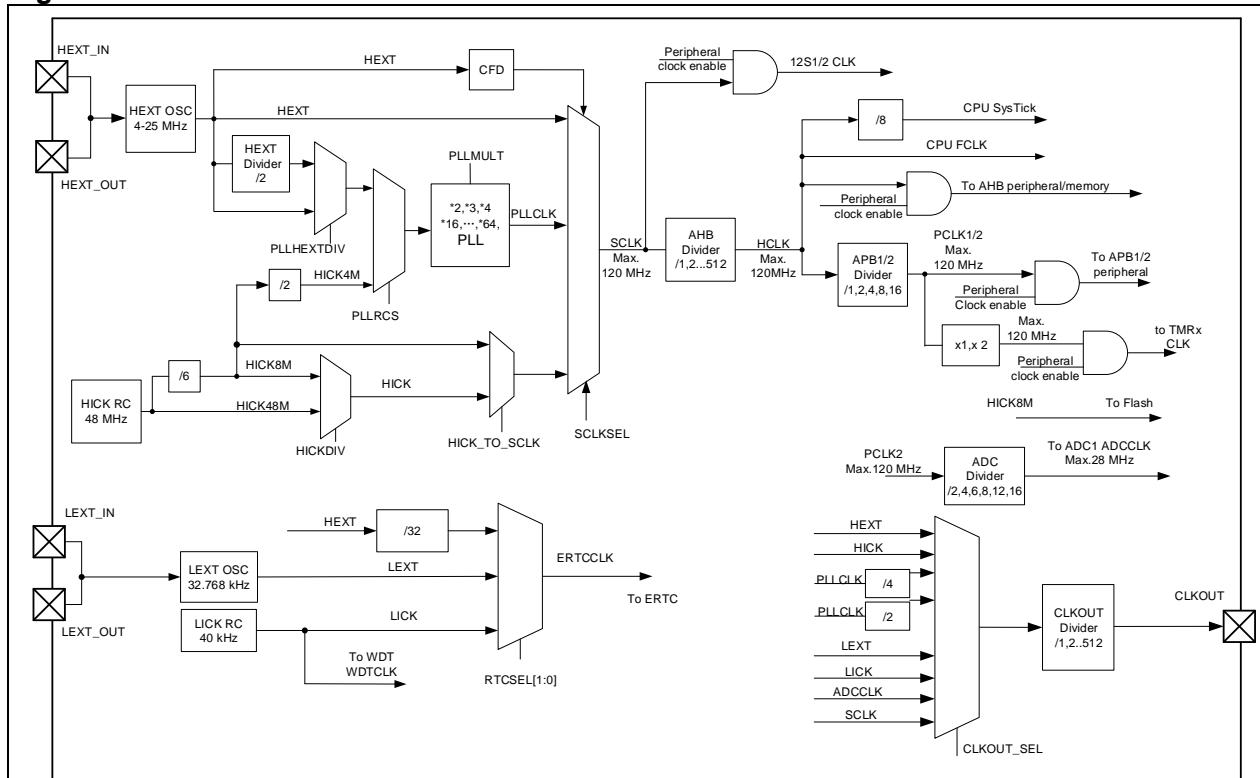
---

## 4 Clock and reset manage (CRM)

### 4.1 Clock

AT32F421 series provide different clock sources: HEXT oscillator clock, HICK oscillator clock, PLL clock, LEXT oscillator and LICK oscillator.

**Figure 4-1 AT32F421 clock tree**



AHB, APB1 and APB2 all support multiple frequency division. The AHB domain has a maximum of 120 MHz, and both APB1 and APB2 are up to 120 MHz.

#### 4.1.1 Clock sources

- High speed external oscillator (HEXT)

The HEXT includes two clock sources: crystal/ceramic resonator and bypass clock.

The HEXT crystal/ceramic resonator is connected externally to a 4~25 MHz HEXT crystal that produces a highly accurate clock for the system. The HEXT clock signal is not released until it becomes stable.

An external clock source can be provided by HEXT bypass. Its frequency can be up to 25 MHz. The external clock signal should be connected to the HEXT\_IN pin while the HEXT\_OUT pin should be left floating.

- High speed internal clock (HICK)

The HICK oscillator is clocked by a high-speed RC in the microcontroller. The internal frequency of the HICK clock is 48 MHz. Although it is less accurate, it takes less time to start than HEXT crystal, and the HICK clock frequency of each device is calibrated to 1% accuracy (25°C) in factory. The factory calibration value is loaded in the HICKCAL[7: 0] bit of the clock control register. The RC oscillator speed may be affected by voltage or temperature variations. Thus the HICK frequency can be trimmed using the HICKTRIM[5: 0] bit in the clock control register.

The HICK clock signal is not released until it becomes stable.

- PLL clock

The HICK or HEXT clock can be used as an input clock source of the PLL. The PLL input clock, after frequency division through pre-scaler inside the PLL, is sent to VCO for frequency multiplication. The VCO frequency is output after being divided by post-scaler. Among them, the pre-scaler clock must be a range of between 2 M and 16 MHz, while the VCO operating frequency must be a range of between

500 MHz and 1000 MHz. Before enabling PLL, it is mandatory to configure PLL parameters for the reason that the configuration parameters cannot be altered once PLL is enabled. The PLL clock signal is not released before it becomes stable.

PLL formula as follows:

PLL output clock = PLL input clock x PLL frequency multiplication factor / (PLL prescaler factor x PLL post-scaler factor)

500MHz <= PLL input clock x PLL frequency multiplication factor / PLL prescaler factor <= 1000MHz

2MHz <= PLL input clock / PLL prescaler factor <= 16MHz

For example, when PLL input clock is 16 MHz, the PLL output frequency can be equal to  $16 \times 96 / (2 \times 8) = 96$  MHz

- Low speed external oscillator (LEXT)

The LEXT oscillator provides two clock sources: LEXT crystal/ceramic resonator and LEXT bypass.

LEXT crystal/ceramic resonator:

The LEXT crystal/ceramic resonator provides a 32.768 KHz low-speed clock source. The LEXT clock signal is not released before it becomes stable.

LEXT bypass clock:

In this mode, an external clock source with a frequency of 32.768 kHz can be provided. The external clock signal should be connected to the LEXT\_IN pin while the LEXT\_OUT must remain floating.

- Low speed internal RC oscillator (LICK)

The LICK oscillator is clocked by an internal low-speed RC oscillator. The clock frequency is between 30 kHz and 60 kHz. It acts as a low-power clock source that can be kept running in Deepsleep mode and Standby mode for watchdog and auto-wakeup unit.

The LICK clock signal is not released before it becomes stable.

## 4.1.2 System clock

After a system reset, the HICK oscillator is selected as system clock. The system clock can make flexible switch among HICK oscillator, HEXT oscillator and PLL clock. However, a switch from one clock source to another occurs only when the target clock source becomes stable. When the HICK oscillator is used directly or indirectly through the PLL as the system clock, it cannot be stopped.

## 4.1.3 Peripheral clock

Most peripherals use HCLK, PCLK1 or PCLK2 clock. The individual peripherals have their dedicated clocks.

System Tick timer (SysTick) is clocked by HCLK or HCLK/8.

ADC is clocked by APB2 divided by 2, 4, 6, 8, 12, 16.

The timers are clocked by APB1/2. In particular, if the APB prescaler is 1, the timer clock frequency is equal to that of APB1/2; otherwise, the timer clock frequency doubles that of the APB1/2 frequency.

ERTC clock sources: HEXT/32 oscillator, LEXT oscillator and LICK oscillator. Once the clock source is selected, it cannot be altered without resetting the battery powered domain. If the LEXT is used as an ERTC clock, the ERTC is not affected when the VDD is powered off. If the HEXT or LICK is selected as an ERTC clock, the ERTC state is not guaranteed when both HEXT and LICK are powered off.

Watchdog is clocked by LICK oscillator. If the watchdog is enabled by either hardware option or software access, the LICK oscillator is forced ON. The clock is provided to the watchdog only after the LICK oscillator temporization.

#### 4.1.4 Clock fail detector

The clock fail detector (CFD) is designed to respond to HEXT clock failure when the HEXT is used as a system clock, directly or indirectly. If a failure is detected on the HEXT clock, a clock failure event is sent to the brake input of TMR1 and an interrupt is generated. This interrupt is directly linked to CPU NMI so that the software can perform rescue operations. The NMI interrupt keeps executing until the CFD interrupt pending bit is cleared. This is why the CFD interrupt has to be cleared in the NMI service routine. The HEXT clock failure will result in a switch of the system clock to the HICK clock, the CFD to be disabled, HEXT clock to be stopped, and even PLL to be disabled if the HEXT clock is selected as the system clock through PLL.

#### 4.1.5 Auto step-by-step system clock switch

The automatic step-by-step frequency switch is designed to ensure a smooth and stable switch of system frequency when the system clock source is switched from others to the PLL or when the AHB prescaler is changed from large to small. When the operational target is larger than 108 MHz, it is recommended to enable the automatic frequency switch. Once it is enabled, the AHB bus is halted by hardware till the completion of the switch. During this switch period, the DMA remain active, and the interrupt events are recorded and then handled by NVIC when the AHB bus resumes.

#### 4.1.6 Internal clock output

The microcontroller allows the internal clock signal to be output to external CLKOUT pins. That is, ADCCLK, SCLK, LICK, LEXT, HICK, HEXT, PLLCLK/2 and PLLCLK/4 can be used as CLKOUT clocks.

#### 4.1.7 Interrupts

The microcontroller specifies a stable flag for each clock source. As a result, when a clock source is enabled, it is possible to determine if the clock is stable by checking the flag pertaining to the clock source. An interrupt request is generated when the interrupt corresponding to the clock source is enabled. If a failure is detected on the HEXT clock, the CFD interrupt is generated. Such interrupt is directly linked to CPU NMI.

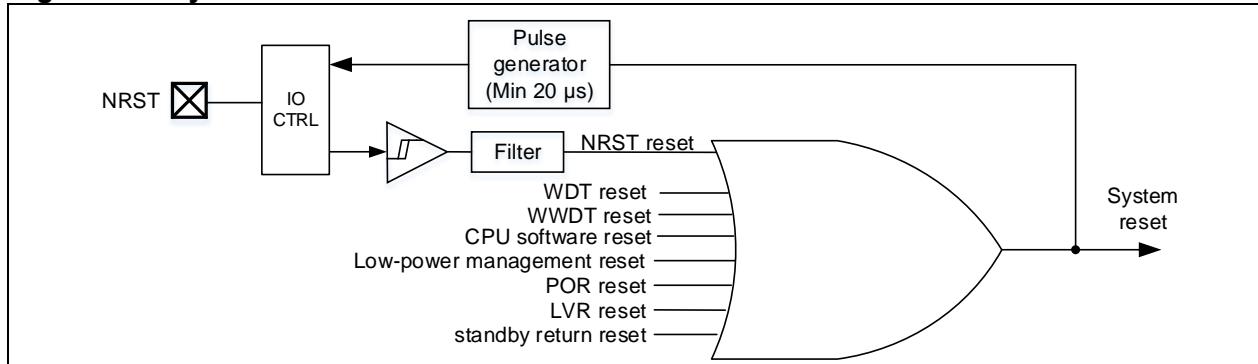
### 4.2 Reset

#### 4.2.1 System reset

AT32F421 series provide the following system reset sources:

- NRST reset: on the external NRST pin
- WDT reset: watchdog overflow
- WWDT reset: window watchdog overflow
- CPU software reset: Cortex™-M4 software reset
- Low-power management reset: This type of reset is enabled when entering Standby mode (by clearing the nSTDBY\_RST bit in the user system data area); this type of reset is also enabled when entering Deepsleep mode (by clearing the nDEPSLP\_RST in the user system data area).
- When exiting Standby mode

NRST reset, WDT reset, WWDT reset, software reset and low-power management reset sets all registers to their reset values except the clock control/status register (CRM\_CTRLSTS) and the battery powered domain; the power-on reset, low-voltage reset or reset generated when exiting Standby mode sets all registers to their reset values except the battery powered domain registers.

**Figure 4-2 System reset circuit**

#### 4.2.2 Battery powered domain reset

Battery powered domain has two specific reset sources:

- Software reset: triggered by setting the BPDRST bit in the battery powered domain control register (CRM\_BPDC)
- VDD power on, if VDD has been powered off.

Software reset affects only the battery powered domain.

### 4.3 CRM registers

These peripheral registers have to be accessed by bytes (8 bits), half words (16 bits) or words (32 bits).

**Table 4-1 CRM register map and reset values**

Register	Offset	Reset value
CRM_CTRL	0x000	0x0000 XX83
CRM_CFG	0x004	0x0000 0000
CRM_CLKINT	0x008	0x0000 0000
CRM_APB2RST	0x00C	0x0000 0000
CRM_APB1RST	0x010	0x0000 0000
CRM_AHBEN	0x014	0x0000 0014
CRM_APB2EN	0x018	0x0000 0000
CRM_APB1EN	0x01C	0x0000 0000
CRM_BPDC	0x020	0x0000 0000
CRM_CTRLSTS	0x024	0xC00 0000
CRM_AHBRST	0x028	0x0000 0000
CRM_PLL	0x2C	0x0000 1F10
CRM_MISC1	0x030	0x0010 0000
CRM_MISC2	0x054	0x0000 000D

### 4.3.1 Clock control register (CRM\_CTRL)

No-wait states, accessible by bytes, half-words or words.

Bit	Name	Reset value	Type	Description
Bit 31: 26	Reserved	0x00	resd	Kept at its default value.
				PLL clock stable
Bit 25	PLLSTBL	0x0	ro	This bit is set by hardware after PLL is ready. 0: PLL clock is not ready. 1: PLL clock is ready.
				PLL enable
Bit 24	PLLEN	0x0	rw	This bit is set and cleared by software. It can also be cleared by hardware when entering Standby or Deepsleep mode. When the PLL clock is used as the system clock, this bit cannot be cleared. 0: PLL is OFF 1: PLL is ON.
Bit 23: 20	Reserved	0x0	resd	Kept at its default value.
				Clock failure detector enable
Bit 19	CFDEN	0x0	rw	0: OFF 1: ON
				High speed external crystal bypass
Bit 18	HEXTBYPSS	0x0	rw	This bit can be written only if the HEXT is disabled. 0: OFF 1: ON
				High speed external crystal stable
Bit 17	HEXTSTBL	0x0	ro	This bit is set by hardware after HEXT becomes stable. 0: HEXT is not ready. 1: HEXT is ready.
				High speed external crystal enable
Bit 16	HEXTEN	0x0	rw	This bit is set and cleared by software. It can also be cleared by hardware when entering Standby or Deepsleep mode. When the HEXT1 clock is used as the system clock, this bit cannot be cleared 0: OFF. 1: ON
				High speed internal clock calibration The default value of this field is the initial factory calibration value.
Bit 15: 8	HICKCAL	0xXX	rw	When the HICK output frequency is 48 MHz, it needs adjust 240 kHz (design value) based on this frequency for each HICKCAL value change; when HICK output frequency is 8 MHz (design value), it needs adjust 40 kHz based on this frequency for each HICKCAL value change. Note: This bit can be written only if the HICKCAL_KEY[7:0] is set as 0x5A.
				High speed internal clock trimming
Bit 7: 2	HICKTRIM	0x20	rw	These bits work with the HICKCAL[7: 0] to determine the HICK oscillator frequency. The default value is 32, which can trim the HICK to be ±1%.
				High speed internal clock stable
Bit 1	HICKSTBL	0x1	ro	This bit is set by hardware after the HICK is ready. 0: Not ready 1: Ready
				High speed internal clock enable
Bit 0	HICKEN	0x1	rw	This bit is set and cleared by software. It can also be set by hardware when exiting Standby or Deepsleep mode. When a HEXT clock failure occurs. This bit can also be set. When the HICK is used as the system clock, this bit cannot be cleared. 0: Disabled 1: Enabled

### 4.3.2 Clock configuration register (CRM\_CFG)

Bit	Name	Reset value	Type	Description																		
Bit 31	Reserved	0x0	resd	Kept at its default value.																		
Bit 26:24	CLKOUT_SEL	0x0	rw	<p>Clock output selection CLKOUT_SEL[3] is the bit 16 of the CRM_MISC1 register.</p> <ul style="list-style-type: none"> <li>0000: None</li> <li>0001: Reserved</li> <li>0010: LICK</li> <li>0011: LEXT</li> <li>0100: SCLK</li> <li>0101: HICK</li> <li>0110: HEXT</li> <li>0111: PLL/2</li> <li>1100: PLL/4</li> <li>1101: USB</li> <li>1110: ADC</li> </ul>																		
Bit 27	Reserved	0x00	resd	Kept at its default value.																		
Bit 23: 22				<p>PLL multiplication factor</p> <table border="0"> <tr> <td>000000: PLL x 2</td> <td>000001: PLL x 3</td> </tr> <tr> <td>000010: PLL x 4</td> <td>000011: PLL x 5</td> </tr> <tr> <td>.....</td> <td></td> </tr> <tr> <td>001100: PLL x 14</td> <td>001101: PLL x 15</td> </tr> <tr> <td>001110: PLL x 16</td> <td>001111: PLL x 16</td> </tr> <tr> <td>010000: PLL x 17</td> <td>010001: PLL x 18</td> </tr> <tr> <td>010010: PLL x 19</td> <td>010011: PLL x 20</td> </tr> <tr> <td>.....</td> <td></td> </tr> <tr> <td>111110: PLL x 63</td> <td>111111: PLL x 64</td> </tr> </table>	000000: PLL x 2	000001: PLL x 3	000010: PLL x 4	000011: PLL x 5	.....		001100: PLL x 14	001101: PLL x 15	001110: PLL x 16	001111: PLL x 16	010000: PLL x 17	010001: PLL x 18	010010: PLL x 19	010011: PLL x 20	.....		111110: PLL x 63	111111: PLL x 64
000000: PLL x 2	000001: PLL x 3																					
000010: PLL x 4	000011: PLL x 5																					
.....																						
001100: PLL x 14	001101: PLL x 15																					
001110: PLL x 16	001111: PLL x 16																					
010000: PLL x 17	010001: PLL x 18																					
010010: PLL x 19	010011: PLL x 20																					
.....																						
111110: PLL x 63	111111: PLL x 64																					
Bit 30: 29	PLLMULT	0x00	rw	<p>HEXT division selection for PLL entry clock</p> <table border="0"> <tr> <td>0: No division</td> <td></td> </tr> <tr> <td>1: HEXT/2</td> <td></td> </tr> </table>	0: No division		1: HEXT/2															
0: No division																						
1: HEXT/2																						
Bit 21: 18				<p>.....</p>																		
Bit 17	PLLHEXTDIV	0x0	rw	<p>PLL reference clock select</p> <table border="0"> <tr> <td>0: HICK-divided clock (4MHz)</td> <td></td> </tr> <tr> <td>1: HEXT clock</td> <td></td> </tr> </table>	0: HICK-divided clock (4MHz)		1: HEXT clock															
0: HICK-divided clock (4MHz)																						
1: HEXT clock																						
Bit 16	PLLRCSC	0x0	rw	<p>ADC division</p> <p>The PCLK divided by the following factors serves the ADC.</p> <table border="0"> <tr> <td>000: PCLK/2</td> <td></td> </tr> <tr> <td>001: PCLK/4</td> <td></td> </tr> <tr> <td>010: PCLK/6</td> <td></td> </tr> <tr> <td>011: PCLK/8</td> <td></td> </tr> <tr> <td>100: PCLK/2</td> <td></td> </tr> <tr> <td>101: PCLK/12</td> <td></td> </tr> <tr> <td>110: PCLK/8</td> <td></td> </tr> <tr> <td>111: PCLK/16</td> <td></td> </tr> </table>	000: PCLK/2		001: PCLK/4		010: PCLK/6		011: PCLK/8		100: PCLK/2		101: PCLK/12		110: PCLK/8		111: PCLK/16			
000: PCLK/2																						
001: PCLK/4																						
010: PCLK/6																						
011: PCLK/8																						
100: PCLK/2																						
101: PCLK/12																						
110: PCLK/8																						
111: PCLK/16																						
Bit 28	ADCDIV	0x0	rw	<p>APB2 division</p> <p>The divided HCLK is used as APB2 clock.</p> <table border="0"> <tr> <td>0xx: not divided</td> <td></td> </tr> <tr> <td>100: divided by 2</td> <td></td> </tr> <tr> <td>101: divided by 4</td> <td></td> </tr> <tr> <td>110: divided by 8</td> <td></td> </tr> <tr> <td>111: divided by 16</td> <td></td> </tr> </table>	0xx: not divided		100: divided by 2		101: divided by 4		110: divided by 8		111: divided by 16									
0xx: not divided																						
100: divided by 2																						
101: divided by 4																						
110: divided by 8																						
111: divided by 16																						
Bit 15: 14				<p>Note: The software must set these bits correctly to ensure that the APB2 clock frequency does not exceed 120 MHz.</p>																		
Bit 13: 11	APB2DIV	0x0	rw	<p>APB1 division</p> <p>The divided HCLK is used as APB1 clock.</p> <table border="0"> <tr> <td>0xx: not divided</td> <td></td> </tr> <tr> <td>100: divided by 2</td> <td></td> </tr> <tr> <td>101: divided by 4</td> <td></td> </tr> <tr> <td>110: divided by 8</td> <td></td> </tr> <tr> <td>111: divided by 16</td> <td></td> </tr> </table>	0xx: not divided		100: divided by 2		101: divided by 4		110: divided by 8		111: divided by 16									
0xx: not divided																						
100: divided by 2																						
101: divided by 4																						
110: divided by 8																						
111: divided by 16																						
Bit 10: 8	APB1DIV	0x0	rw	<p>Note: The software must set these bits correctly to</p>																		

				ensure that the APB1 clock frequency does not exceed 120 MHz
				AHB division The divided SCLK is used as AHB clock. 0xxx: SCLK not divided
Bit 7: 4	AHBDIV	0x0	rw	1000: SCLK divided by 2    1100: SCLK divided by 64 1001: SCLK divided by 4    1101: SCLK divided by 128 1010: SCLK divided by 8    1110: SCLK divided by 256 1011: SCLK divided by 16    1111: SCLK divided by 512
				System clock select status 00: HICK 01: HEXT 10: PLL 11: Reserved. Kept at its default value.
Bit 3: 2	SCLKSTS	0x0	r0	System clock select 00: HICK 01: HEXT 10: PLL 11: Reserved. Kept at its default value.
				System clock select 00: HICK 01: HEXT 10: PLL 11: Reserved. Kept at its default value.
Bit 1: 0	SCLKSEL	0x0	rw	System clock select 00: HICK 01: HEXT 10: PLL 11: Reserved. Kept at its default value.

### 4.3.3 Clock interrupt register (CRM\_CLKINT)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23	CFDFC	0x0	wo	Clock failure detection flag clear Writing 1 by software to clear CFDF. 0: No effect 1: Clear
Bit 22: 21	Reserved	0x0	resd	Kept at its default value.
Bit 20	PLLSTBLFC	0x0	wo	PLL stable flag clear Writing 1 by software to clear PLLSTBLF. 0: No effect 1: Clear
Bit 19	HEXTSTBLFC	0x0	wo	HEXT stable flag clear Writing 1 by software to clear HEXTSTBLF. 0: No effect 1: Clear
Bit 18	HICKSTBLFC	0x0	wo	HICK stable flag clear Writing 1 by software to clear HICKSTBLF. 0: No effect 1: Clear
Bit 17	LEXTSTBLFC	0x0	wo	LEXT stable flag clear Writing 1 by software to clear LEXTSTBLF. 0: No effect 1: Clear
Bit 16	LICKSTBLFC	0x0	wo	LICK stable flag clear Writing 1 by software to clear LICKSTBLF. 0: No effect 1: Clear
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	PLLSTBLIEN	0x0	rw	PLL stable interrupt enable 0: Disabled 1: Enabled
Bit 11	HEXTSTBLIEN	0x0	rw	HEXT stable interrupt enable 0: Disabled 1: Enabled
Bit 10	HICKSTBLIEN	0x0	rw	HICK stable interrupt enable 0: Disabled 1: Enabled
Bit 9	LEXTSTBLIEN	0x0	rw	LEXT stable interrupt enable 0: Disabled 1: Enabled
Bit 8	LICKSTBLIEN	0x0	rw	LICK stable interrupt enable 0: Disabled

				1: Enabled Clock Failure Detection flag This bit is set by hardware when the HEXT clock failure occurs. 0: No clock failure 1: Clock failure
Bit 7	CFDF	0x0	ro	Keep at its default value.
Bit 6: 5	Reserved	0x0	resd	PLL stable flag Set by hardware. 0: PLL is not ready. 1: PLL is ready.
Bit 4	PLLSTBLF	0x0	ro	HEXT stable flag Set by hardware. 0: HEXT is not ready. 1: HEXT is ready.
Bit 3	HEXTSTBLF	0x0	ro	HICK stable flag Set by hardware. 0: HICK is not ready. 1: HICK is ready.
Bit 2	HICKSTBLF	0x0	ro	LEXT stable flag Set by hardware. 0: LEXT is not ready. 1: LEXT is ready.
Bit 1	LEXTSTBLF	0x0	ro	LICK stable interrupt flag Set by hardware. 0: LICK is not ready. 1: LICK is ready.
Bit 0	LICKSTBLF	0x0	ro	

#### 4.3.4 APB2 peripheral reset register (CRM\_APB2RST)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 19	Reserved	0x0000	resd	Kept at its default value.
Bit 18	TMR17ST	0x0	resd	TMR17 reset 0: Does not reset TMR17 1: Reset TMR17
Bit 17	TMR16ST	0x0	rw	TMR16 set 0: Does not reset TMR16 1: Reset TMR16
Bit 16	TMR15ST	0x0	rw	TMR15 reset 0: Does not reset TMR15 1: Reset TMR15
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	USART1RST	0x0	rw	USART1 reset 0: Does not reset USART1 1: Reset USART1
Bit 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	SPI1RST	0x0	rw	SPI1 reset 0: Does not reset SPI1 1: Reset SPI1
Bit 11	TMR1RST	0x0	rw	TMR1 reset 0: Does not reset TMR1 1: Reset TMR1
Bit 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	ADCRST	0x0	rw	ADC reset 0: Does not reset ADC 1: Reset ADC
Bit 8: 2	Reserved	0x00	resd	Kept at its default value.
Bit 1	EXINTRST	0x0	rw	EXINT reset 0: Does not reset EXINT 1: Reset EXINT Note: This bit is always 0 when reading by software.
Bit 0	SCFGCMPRST	0x0	rw	SCFG and CMP reset 0: Does not reset SCFG and CMP

1: Reset SCFG and CMP

### 4.3.5 APB1 peripheral reset register1 (CRM\_APB1RST)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 29	Reserved	0x0	resd	Kept at its default value.
Bit 28	PWCRST	0x0	rw	PWC reset 0: Does not reset PWC 1: Reset PWC
Bit 27: 23	Reserved	0x00	resd	Kept at its default value.
Bit 22	I2C2RST	0x0	rw	I2C2 reset 0: Does not reset I2C2 1: Reset I2C2
Bit 21	I2C1RST	0x0	rw	I2C1 reset 0: Does not reset I2C1 1: Reset I2C1
Bit 20: 18	Reserved	0x0	resd	Kept at its default value.
Bit 17	USART2RST	0x0	rw	USART2 reset 0: Does not reset USART2 1: Reset USART2
Bit 16: 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	SPI2RST	0x0	rw	SPI2 reset 0: Does not reset SPI2 1: Reset SPI2
Bit 13:12	Reserved	0x0	resd	Kept at its default value.
Bit 11	WWDTRST	0x0	rw	WWDT reset 0: Does not reset WWDT 1: Reset WWDT
Bit 10: 9	Reserved	0x0	resd	Kept at its default value.
Bit 8	TMR14RST	0x0	rw	TMR14 reset 0: Does not reset TMR14 1: Reset TMR14
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	TMR6RST	0x0	rw	TMR6 reset 0: Does not reset TMR6 1: Reset TMR6
Bit 3: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	TMR3RST	0x0	rw	TMR3 reset 0: Does not reset TMR3 1: Reset TMR3
Bit 0	Reserved	0x0	resd	Kept at its default value.

### 4.3.6 AHB peripheral clock enable register (CRM\_AHBEN)

Access: by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22	GPIOFEN	0x0	rw	GPIOF clock enable 0: Disabled 1: Enabled
Bit 21: 20	Reserved	0x0	resd	Kept at its default value.
Bit 19	GPIOCEN	0x0	rw	GPIOC clock enable 0: Disabled 1: Enabled
Bit 18	GPIOBEN	0x0	rw	GPIOB clock enable 0: Disabled 1: Enabled
Bit 17	GPIOAEN	0x0	rw	GPIOA clock enable 0: Disabled 1: Enabled
Bit 16:7	Reserved	0x000	resd	Kept at its default value.
Bit 6	CRCEN	0x0	rw	CRC clock enable

				0: Disabled 1: Enabled
Bit 5	Reserved	0x0	resd	Kept at its default value.
				FLASH clock enable
Bit 4	FLASHEN	0x1	rw	This bit is used to enable Flash clock in Sleep or Deepsleep mode. 0: Disabled 1: Enabled
Bit 3	Reserved	0x0	resd	Kept at its default value.
				SRAM clock enable
Bit 2	SRAMEN	0x1	rw	This bit is used to enable SRRM clock in Sleep or Deepsleep mode. 0: Disabled 1: Enabled
Bit 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	DMA1EN	0x0	rw	DMA1 clock enable 0: Disabled 1: Enabled

#### 4.3.7 APB2 peripheral clock enable register (CRM\_APB2EN)

Access: by words, half-words and bytes.

When accessing to peripherals on APB2 bus, wait states are inserted until the completion of the peripheral access on APB2.

Bit	Name	Reset value	Type	Description
Bit 31: 19	Reserved	0x0000	resd	Kept at its default value.
Bit 18	TMR17EN	0x0	rw	TMR17 clock enable 0: Disabled 1: Enabled
Bit 17	TMR16EN	0x0	rw	TMR16 clock enable 0: Disabled 1: Enabled
Bit 16	TMR15EN	0x0	rw	TMR15 clock enable 0: Disabled 1: Enabled
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	USART1EN	0x0	rw	USART1 clock enable 0: Disabled 1: Enabled
Bit 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	SPI1EN	0x0	rw	SPI1 clock enable 0: Disabled 1: Enabled
Bit 11	TMR1EN	0x0	rw	TMR1 clock enable 0: Disabled 1: Enabled
Bit 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	ADC1EN	0x0	rw	ADC1 clock enable 0: Disabled 1: Enabled
Bit 8: 1	Reserved	0x00	resd	Kept at its default value.
Bit 0	SCFGCMPEN	0x0	rw	SCFG and CMP clock enable 0: Disabled 1: Enabled

### 4.3.8 APB1 peripheral clock enable register (CRM\_APB1EN)

Access: by words, half-words and bytes.

No-wait states in most cases. However, when accessing to peripherals on APB1, wait-states are inserted until the end of peripheral access on the APB1 bus.

Bit	Name	Reset value	Type	Description
Bit 31: 29	Reserved	0x0	resd	Kept at its default value.
Bit 28	PWCEN	0x0	rw	PWC clock enable 0: Disabled 1: Enabled
Bit 27: 23	Reserved	0x00	resd	Kept at its default value.
Bit 22	I2C2EN	0x0	rw	I2C2 clock enable 0: Disabled 1: Enabled
Bit 21	I2C1EN	0x0	rw	I2C1 clock enable 0: Disabled 1: Enabled
Bit 20: 18	Reserved	0x0	resd	Kept at its default value.
Bit 17	USART2EN	0x0	rw	USART2 clock enable 0: Disabled 1: Enabled
Bit 16: 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	SPI2EN	0x0	rw	SPI2 clock enable 0: Disabled 1: Enabled
Bit 13: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11	WWDTEN	0	rw	WWDT clock enable 0: Disabled 1: Enabled
Bit 10: 9	Reserved	0x0	resd	Kept at its default value.
Bit 8	TMR14EN	0x0	rw	TMR14 clock enable 0: Disabled 1: Enabled
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	TMR6EN	0x0	rw	TMR6 clock enable 0: Disabled 1: Enabled
Bit 3: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	TMR3EN	0x0	rw	TMR3 clock enable 0: Disabled 1: Enabled
Bit 0	Reserved	0x0	resd	Kept at its default value.

### 4.3.9 Battery powered domain control register (CRM\_BPDC)

Access: 0 to 3 wait states, accessible by words, half-words or bytes. Wait states are inserted in the case of consecutive accesses to this register.

*Note: LEXTEN, LEXTBYPSS, ERTCSEL, and ERTCEN bits of the battery powered domain control register (CRM\_BPDC) are in the battery powered domain. As a result, these bits are write protected after reset, and can only be modified by setting the BPWEN bit in the power control register (PWR\_CTRL). These bits could be reset only by battery powered domain reset. Any internal or external reset does not affect these bits.*

Bit	Name	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	BPDRST	0x0	rw	Battery powered domain software reset 0: Do not reset battery powered domain software 1: Reset battery powered domain software
Bit 15	ERTCEN	0x0	rw	ERTC clock enable Set and cleared by software. 0: Disabled 1: Enabled
Bit 14: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9: 8	ERTCSEL	0x0	rw	ERTC clock selection Once the ERTC clock source is selected, it cannot be changed until the BPDRST bit is reset. 00: No clock 01: LEXT 10: LICK 11: HEXT/128
Bit 7: 3	Reserved	0x00	resd	Kept at its default value.
Bit 2	LEXTBYPSS	0x0	rw	Low speed external crystal bypass 0: Disabled 1: Enabled
Bit 1	LEXTSTBL	0x0	ro	Low speed external oscillator stable Set by hardware after the LEXT is ready. 0: LEXT is not ready. 1: LEXT is ready.
Bit 0	LEXTEN	0x0	rw	External low-speed oscillator enable 0: Disabled 1: Enabled

### 4.3.10 Control/status register (CRM\_CTRLSTS)

Reset flag can only be cleared by power reset or by setting the RSTFC bit, while others are cleared by system reset.

Access: 0 to 3 wait states, accessible by words, half-words or bytes. Wait states are inserted in the case of consecutive accesses to this register.

Bit	Name	Reset value	Type	Description
Bit 31	LPRSTF	0x0	ro	Low-power reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No low-power reset occurs 1: Low-power reset occurs
Bit 30	WWDTRSTF	0x0	ro	Window watchdog timer reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No window watchdog timer reset occurs 1: Window watchdog timer reset occurs
Bit 29	WDTRSTF	0x0	ro	Watchdog timer reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No watchdog timer reset occurs 1: Watchdog timer reset occurs.
Bit 28	SWRSTF	0x0	ro	Software reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No software reset occurs 1: Software reset occurs.

Bit 27	PORRSTF	0x1	ro	POR/LVR reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No POR/LVR reset occurs 1: POR/LVR reset occurs.
Bit 26	NRSTF	0x1	ro	NRST pin reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No NRST pin reset occurs 1: NRST pin reset occurs
Bit 25	Reserved	0x0	resd	Kept at its default value.
Bit 24	RSTFC	0x0	rw	Reset flag clear Cleared by writing 1 through software. 0: No effect 1: Clear the reset flag.
Bit 23: 2	Reserved	0x000000	resd	Kept at its default value.
Bit 1	LICKSTBL	0x0	ro	LICK stable 0: LICK is not ready. 1: LICK is ready.
Bit 0	LICKEN	0x0	rw	LICK enable 0: Disabled 1: Enabled

#### 4.3.11 AHB peripheral reset register (CRM\_AHBRST)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31:23	Reserved	0x000	resd	Kept at its default value.
Bit 22	GPIOFRST	0x0	rw	GPIOF reset This bit is set or cleared by software. 0: Does not reset GPIOF 1: Reset GPIOF
Bit 21: 20	Reserved	0x000	resd	Kept at its default value.
Bit 19	GPIOCRST	0x0	rw	GPIOC reset This bit is set or cleared by software. 0: Does not reset GPIOC 1: Reset GPIOC
Bit 18	GPIOBRST	0x0	rw	GPIOB reset This bit is set or cleared by software. 0: Does not reset GPIOB 1: Reset GPIOB
Bit 17	GPIOARST	0x0	rw	GPIOA reset This bit is set or cleared by software. 0: Does not reset GPIOA 1: Reset GPIOA
Bit 16: 0	Reserved	0x00000	resd	Kept at its default value.

#### 4.3.12 PLL configuration register (CRM\_PLL)

Bit	Name	Reset value	Type	Description
Bit 31	PLLCFGEN	0x0	rw	PLL configuration enable 0: Common integer multiplication mode, which is done by PLL_REF and PLLMULT registers. 1: Flexible configuration mode, which is done by PLL_MS/PLL_NS/PLL_FR registers.
Bit 30: 27	Reserved	0x0	resd	Kept at its default value.
Bit 26: 24	PLL_REF	0x0	rw	PLL input clock selection This field is valid only if PLLCFGGEN=0. 000: 3.9 ~ 5 MHz 001: 5.2 ~ 6.25 MHz 010: 7.8125 ~ 8.33 MHz 011: 8.33 ~ 12.5 MHz 100: 15.625 ~ 20.83 MHz 101: 20.83 ~ 31.25 MHz 110: Reserved 111: Reserved
Bit 23: 17	Reserved	0x00	resd	Kept at its default value.

Bit 16: 8	PLL_NS	0x01F	rw	PLL multiplication factor PLL_NS range (31~500)
Bit 7: 4	PLL_MS	0x1	rw	PLL pre-division PLL_MS range (1~15)
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2: 0	PLL_FR	0x0	rw	PLL post-division factor PLL_FR range (0~5) 000: PLL post-division=1, divided by 1 001: PLL post-division=2, divided by 2 010: PLL post-division=4, divided by 4 011: PLL post-division=8, divided by 8 100: PLL post-division=16, divided by 16 101: PLL post-division=32, divided by 32 Others: Reserved It should be noted the relationship between the PLL-FR values and post-division factors.

#### 4.3.13 Additional register (CRM\_MISC1)

Bit	Name	Reset value	Type	Description
Bit 31: 28	CLKOUTDIV	0x0	rw	Clock output division 0xxx: Clock output 1000: Clock output divided by 2 1001: Clock output divided by 4 1010: Clock output divided by 8 1011: Clock output divided by 16 1100: Clock output divided by 64 1101: Clock output divided by 128 1110: Clock output divided by 256 1111: Clock output divided by 512
Bit 27: 26	Reserved	0x0	resd	Kept its default value.
Bit 25	HICKDIV	0x0	rw	HICK 6 divider selection This bit is used to select HICK or HICK /6. If the HICK/6 is selected, the clock frequency is 8 MHz. Otherwise, the clock frequency is 48 MHz. 0: HICK/6 1: HICK Note: In any case, HICK always input 4 MHz to PLL.
Bit 24: 21	Reserved	0x0	resd	Kept at its default value.
Bit 20: 17	Reserved	0x8	resd	Kept at its default value.
Bit 16	CLKOUT_SEL[3]	0x0	rw	Clock output selection This bit works with the bit [26:24] of the CRM_CFG register.
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7: 0	HICKCAL_KEY	0x00	rw	HICK calibration key The HICKCAL [7:0] can be written only when this field is set 0x5A.

### 4.3.14 Additional register (CRM\_MISC2)

Bit	Name	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Kept at its default value.
Bit 9	HICK_TO_SCLK	0x0	rw	HICK as system clock frequency select When the HICK is selected as the clock source SCLKSEL, the frequency of SCLK is: 0: Fixed 8 MHz, that is, HICK/6 1: 48 MHz or 8 MHz, depending on the HICKDIV
Bit 8: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5: 4	AUTO_STEP_EN	0x0	rw	Auto step-by-step system clock switch enable When the system clock source is switched from others to the PLL or when the AHB prescaler is changed from large to small (system frequency is from small to large), it is recommended to enable the auto step-by-step system clock switch if the operational target is larger than 108 MHz., Once it is enabled, the AHB bus is halted by hardware till the completion of the switch. During this switch period, the DMA remain working, and the interrupt events are recorded and then handled by NVIC when the AHB bus resumes. 00: Disabled 01: Reserved 10: Reserved 11: Enabled. When AHBDIV or SCLKSEL is modified, the auto step-by-step
Bit 3: 0	Reserved	0xd	resd	Kept at its default value.

# 5 Embedded Flash memory controller (FLASH)

## 5.1 FLASH introduction

Flash memory is divided into three parts: main Flash memory, information block and Flash memory registers.

- Main Flash memory is up to 64 KB
- Information block consists of 4 KB boot memory and the user system data area. The boot memory uses USART1 or USART2 for ISP programming.

Main Flash memory contains block 1 only (64 KB), including 64 sectors, 1K per sector.

**Table 5-1 Flash memory architecture(64 K)**

Block	Name	Address range
Main memory	Sector 0	0x0800 0000 – 0x0800 03FF
	Sector 1	0x0800 0400 – 0x0800 07FF
	Sector 2	0x0800 0800 – 0x0800 0BFF
	...	...
	Sector 63	0x0800 FC00 – 0x0800 FFFF
	4KB Boot memory	0x1FFF E400 – 0x1FFF F3FF
Information block	512B user system data	0x1FFF F800 – 0x1FFF F9FF

Main Flash memory contains block 1 only (32 KB), including 32 sectors, 1K per sector.

**Table 5-2 Flash memory architecture(32 K)**

Block	Name	Address range
Main memory	Sector 0	0x0800 0000 – 0x0800 03FF
	Sector 1	0x0800 0400 – 0x0800 07FF
	Sector 2	0x0800 0800 – 0x0800 0BFF
	...	...
	Sector 31	0x0800 7C00 – 0x0800 7FFF
	4KB Boot memory	0x1FFF E400 – 0x1FFF F3FF
Information block	512B User system data	0x1FFF F800 – 0x1FFF F9FF

Main Flash memory contains block 1 only (16 KB), including 16 sectors, 1K per sector.

**Table 5-3 Flash memory architecture(16 K)**

Block	Name	Address range
Main memory	Sector 0	0x0800 0000 – 0x0800 03FF
	Sector 1	0x0800 0400 – 0x0800 07FF
	Sector 2	0x0800 0800 – 0x0800 0BFF
	...	...
	Sector 15	0x0800 3C00 – 0x0800 3FFF
	4KB Boot memory	0x1FFF E400 – 0x1FFF F3FF
Information block	512B User system data	0x1FFF F800 – 0x1FFF F9FF

**User system data area**

The system data will be read from the information block of Flash memory whenever a system reset occurs, and saved in the user system data register (FLASH\_USD) and erase/programming protection status register (FLASH\_EPPS).

Each system data occupies two bytes in which the low bytes represent the system data, and the high bytes represent the inverse code of system data. This is designed to verify the correctness of the selected bit. When the high byte is not equal to the inverse code of the low byte (except when both high and low bytes are all 0xFF), the system data loader will generate a system data error flag (USDERR) forcing the system data and its inverse code to 0xFF.

*Note: The update of the contents in the user system data area becomes effective only after a system reset.*

**Table 5-4 User system data area**

Address	Bit	Description	
0x1FFF_F800	[7: 0]	<b>FAP[7:0]:</b> Flash memory access protection (Access protection enable/disable result is stored in the register FLASH_USD bit [1] and bit [26] 0xA5: Flash access protection disabled 0XCC: High-level Flash access protection enabled Others: Low-level Flash access protection enabled	
	[15: 8]	<b>nFAP[7: 0]:</b> Inverse code of FAP[7: 0]	
	[23: 16]	<b>SSB[7:0]:</b> System setting byte (saved in the bit [9: 2] of the FLASH_USD register)	
		<b>Bit 7: 5</b>	Reserved
		<b>Bit 4 (nBOOT1)</b>	nBOOT1: This bit, along with BOOT0, defines boot mode. When BOOT0 = 1: 0: Boot from SRAM 1: Boot from boot memory
		<b>Bit 3</b>	Reserved
		<b>Bit 2 (nSTDBY_RST)</b>	0: Reset occurred when entering Standby mode 1: No reset occurred when entering Standby mode
		<b>Bit 1 (nDEPSLP_RST)</b>	0: Reset occurred when entering Deepsleep mode 1: No reset occurred when entering Deepsleep mode
	<b>Bit 0 (nWDT_ATO_EN)</b>	0: Watchdog is enabled 1: Watchdog is disabled	
	[31: 24]	<b>nSSB[7: 0]:</b> Inverse code of SSB[7: 0]	
0x1FFF_F804	[7: 0]	<b>Data0[7:0]:</b> User data 0 (It is stored in the bit [17:10] of the FLASH_USD register)	
	[15: 8]	<b>nData0[7: 0]:</b> Inverse code of Data0[7: 0]	
	[23: 16]	<b>Data1[7:0]:</b> User data 1 (It is stored in the bit [25: 18] of the FLASH_USD register)	
	[31: 24]	<b>nData1[7: 0]:</b> Inverse code of Data1[7: 0]	
	0x1FFF_F808	[7: 0]	<b>EPP0[7:0]:</b> Flash erase/write protection byte 0 (stored in the bit [7: 0] of the FLASH_EPPS register) This field is used to protect sector 0~sector 31 of main Flash memory. Each bit takes care of 4 sectors (1 KB per sector) 0: Erase/write protection is enabled 1: Erase/write protection is disabled
[15: 8]		<b>nEPP0[7: 0]:</b> Inverse code of EPP0[7: 0]	
[23: 16]		<b>EPP1[7:0]:</b> Flash erase/write protection byte 1 (stored in the bit [15: 8] of the FLASH_EPPS register) This field is used to protect sector 32~sector 63 of main Flash memory. Each bit takes care of 4 sectors (1 KB per sector) 0: Erase/write protection is enabled 1: Erase/write protection is disabled	
[31: 24]		<b>nEPP1[7: 0]:</b> Inverse code of EPP1[7: 0]	
0x1FFF_F80C		[7: 0]	<b>EPP2[7:0]:</b> Flash erase/write protection byte 2 (stored in the bit [23: 16] of the FLASH_EPPS register) Reserved
	[15: 8]	<b>nEPP2[7: 0]:</b> Inverse code of EPP2[7:0]	

	[23: 16]	EPP3[7:0]: Flash erase/write protection byte 3 (stored in the bit [31:24] of the FLASH_EPPS register) Bit 6:0 are reserved. Bit 7 is used to protect Flash memory extension area 0: Erase/write protection is enabled 1: Erase/write protection is disabled
	[31: 24]	nEPP3[7: 0]: Inverse code of EPP3[7: 0]
0x1FFF_F810	[7: 0]	Data2[7: 0]: User system data 2
	[15: 8]	nData2[7: 0]: Inverse code of Data2[7: 0]
	[23: 16]	Data3[7: 0]: User system data 3
	[31: 24]	nData3[7: 0]: Inverse code of Data3[7: 0]
0x1FFF_F814	[7: 0]	Data4[7: 0]: User system data 4
	[15: 8]	nData4[7: 0]: Inverse code of Data4[7: 0]
	[23: 16]	Data5[7: 0]: User system data 5
	[31: 24]	nData5[7: 0]: Inverse code of Data5[7: 0]
...	...	
...	...	
0x1FFF_F9FC	[7: 0]	Data248[7: 0]: User system data 248
	[15: 8]	nData248[7: 0]: Inverse code of Data248[7: 0]
	[23: 16]	Data249[7: 0]: User system data 249
	[31: 24]	nData249[7: 0]: Inverse code of Data249[7: 0]

## 5.2 Flash memory operation

### 5.2.1 Unlock/lock

After reset, Flash memory is protected, by default. The FLASH\_CTRL register cannot be written. Write and erase operation can be performed only when the Flash memory is unlocked.

#### Unlock procedure:

Flash memory block can be unlocked by writing KEY1 (0x45670123) and KEY2 (0xCDEF89AB) to the FLASH\_UNLOCK register.

*Note: Writing an incorrect key sequence leads to a bus error and locks up the Flash memory until the next reset.*

#### Lock procedure:

Flash memory block can be locked again by setting the OPLK bit in the FLASH\_CTRL register.

### 5.2.2 Erase operation

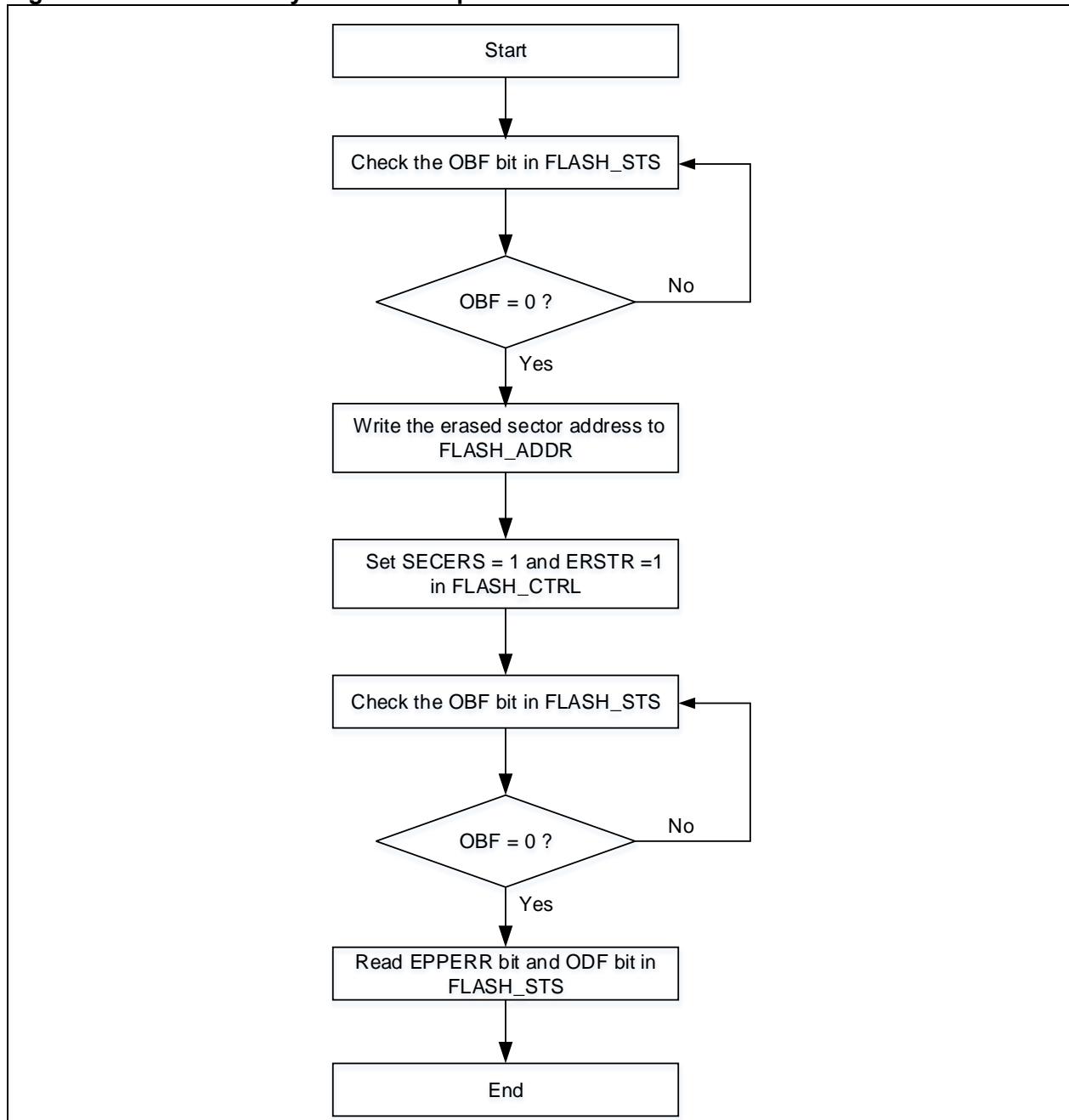
Erase operation must be performed before programming. Flash memory erase includes sector erase and mass erase.

#### Sector erase

Any sector in the Flash memory can be erased with sector erase function independently. To erase a sector, follow the procedures below:

- Check that no Flash memory operation is ongoing by checking the OBF bit of the FLASH\_STS register
- Set the sector to be erased in the FLASH\_ADDR register
- Set the SECERS and ERSTR bits in the FLASH\_CTRL register to enable sector erase
- Wait until the OBF bit becomes “0” in the FLASH\_STS register. Read the EPPER and ODF bits in the FLASH\_STS register to verify.

*Note: When the boot memory is configured as the Flash memory extension area, performing sector-erase operation erases the entire Flash memory extension area.*

**Figure 5-1 Flash memory sector erase process****Mass erase**

Mass erase feature can erase the entire Flash memory.

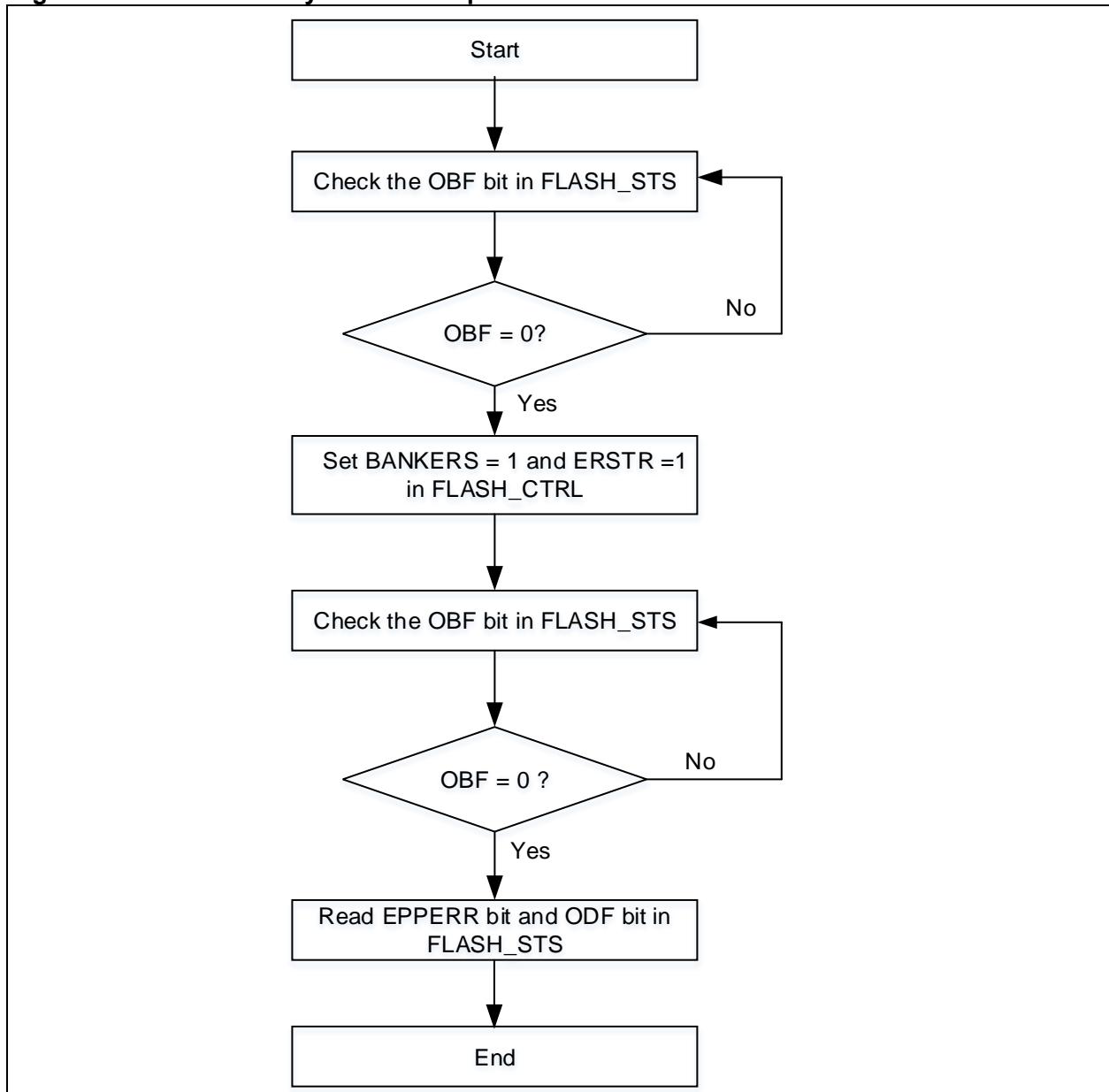
To perform mass erase, follow the procedures below:

- Check that no Flash memory operation is ongoing by checking the OBF bit in the FLASH\_STS register
- Set the BANKERS and ERSTR bits in the FLASH\_CTRL register to enable mass erase;
- Wait until the OBF bit becomes “0” in the FLASH\_STS register. Read the EPPERR bit and ODF bit in the FLASH\_STS register to verify.

*Note:*

- 1) When the boot memory is configured as the Flash memory extension area, performing mass-erase operation erases automatically the entire Flash memory and its extension area.
- 2) Read access during erase operation halts the CPU until the completion of erase.
- 3) Internal HICK must be enabled prior to erase operation.

Figure 5-2 Flash memory mass erase process



### 5.2.3 Programming operation

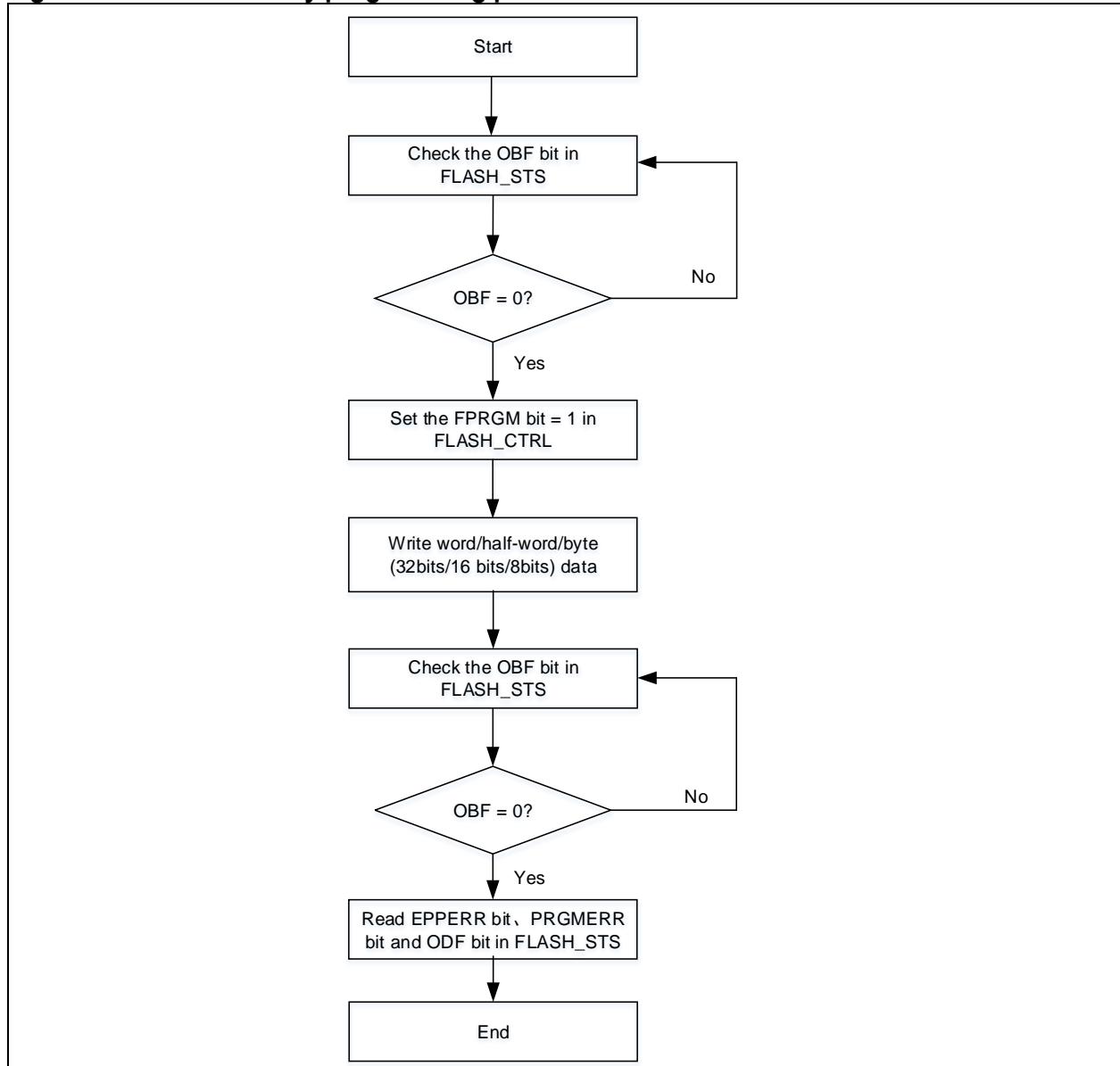
The Flash memory can be programmed with 32-bit, 16-bit or 8-bit data at one time.

The following process is recommended:

- Check that no Flash memory operation is ongoing by checking the OBF bit in the FLASH\_STS register
- Set the FPRGM bit in the FLASH\_CTRL register, so that the Flash memory programming instructions can be received;
- Write the data (word/half-word/byte) to be programmed to the designated address;
- Wait until the OBF bit in the FLASH\_STS register becomes “0”, read the EPPERR, PRGMERR and ODF bits to verify.

*Note:*

1. When the address to be written is not erased in advance, the programming operation is not executed unless the data to be written is all 0. In this case, a programming error is reported by setting the PRGMERR bit in the FLASH\_STS register.
2. Read operation to the Flash memory during programming halts CPU until the completion of programming.
3. Internal HICK must be enabled prior to programming.

**Figure 5-3 Flash memory programming process**

### 5.2.4 Read operation

Flash memory can be accessed through AHB bus of the CPU.

## 5.3 Main Flash memory extension area

Boot memory can also be programmed as the extension area of the main Flash memory to store user-application code. When used as main Flash memory extension area, it behaves like the main Flash memory, including read, unlock, erase and programming operations.

## 5.4 User system data area

### 5.4.1 Unlock/lock

After reset, user system data area is protected, by default. Write and erase operations can be performed only after the Flash memory is unlocked and then the user system data area is unlocked.

#### Unlock procedure:

Flash memory block can be unlocked by writing KEY1 (0x45670123) and KEY2 (0xCDEF89AB) to the FLASH\_UNLOCK register;

Writing KEY1 (0x45670123) and KEY2 (0xCDEF89AB) to the FLASH\_USD\_UNLOCK register, the USDULKS bit in the FLASH\_CTRL register will be automatically set by hardware, indicating that the user system data area can be programmed (write/erase).

*Note: Writing an incorrect key sequence leads to bus error and locks up the Flash memory until the next reset.*

#### Lock procedure:

User system data area is locked again by clearing the USDULKS bit in the FLASH\_CTRL register by software.

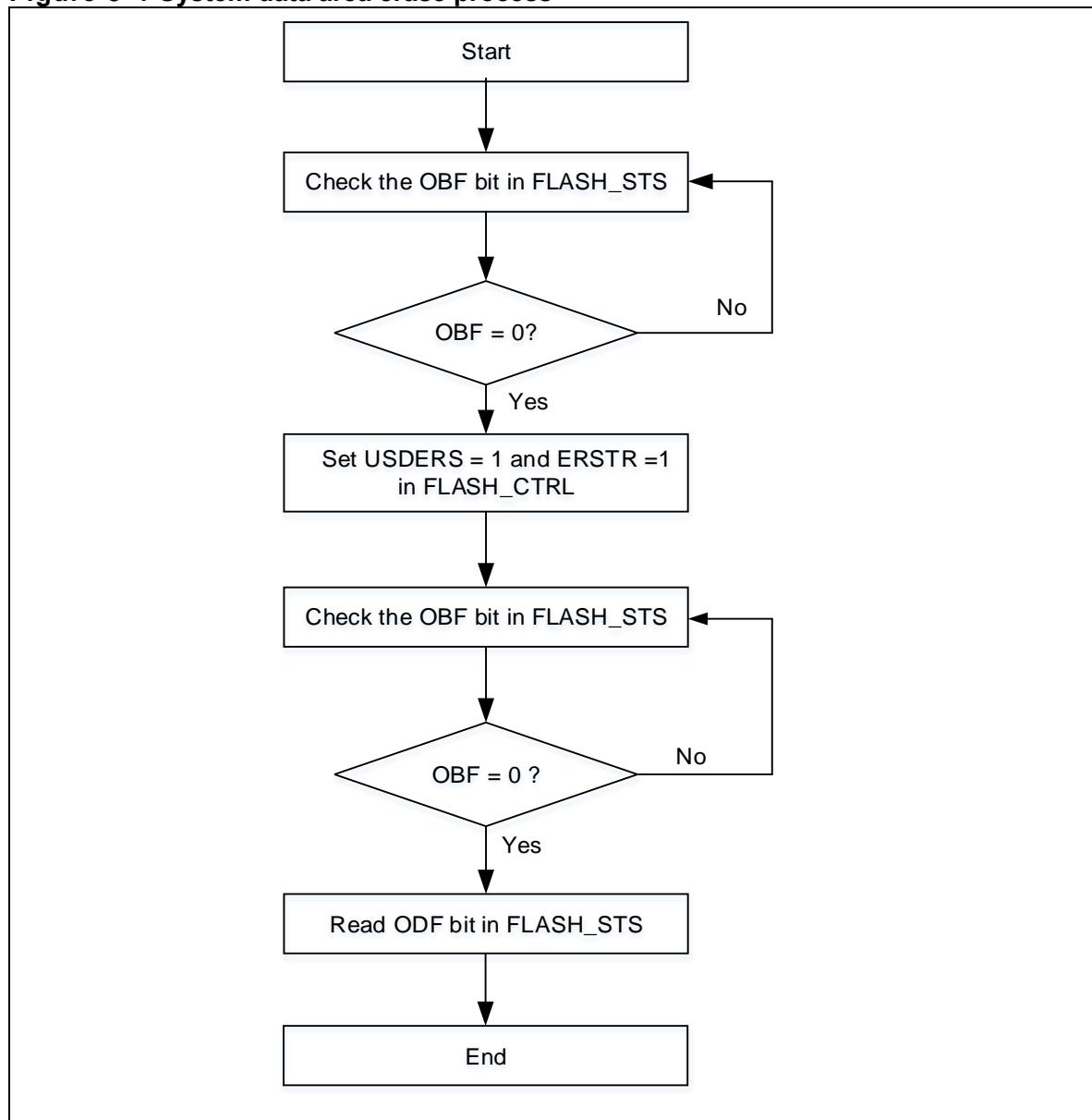
### 5.4.2 Erase operation

Erase operation must be done before programming. User system data area can be erased independently.

To perform erase, following the procedures below:

- Check that no Flash memory operation is ongoing by checking the OBF bit in the FLASH\_STS register
- Set the USDERS and ERSTR bits in the FLASH\_CTRL register to enable erase operation;
- Wait until the OBF bit becomes “0” in the FLASH\_STS register. Read the ODF bit in the FLASH\_STSx register to verify.

*Note: Read operation to the Flash memory during programming halts CPU until the completion of erase. The internal HICK must be enabled prior to erase operation.*

**Figure 5-4 System data area erase process**

### 5.4.3 Programming operation

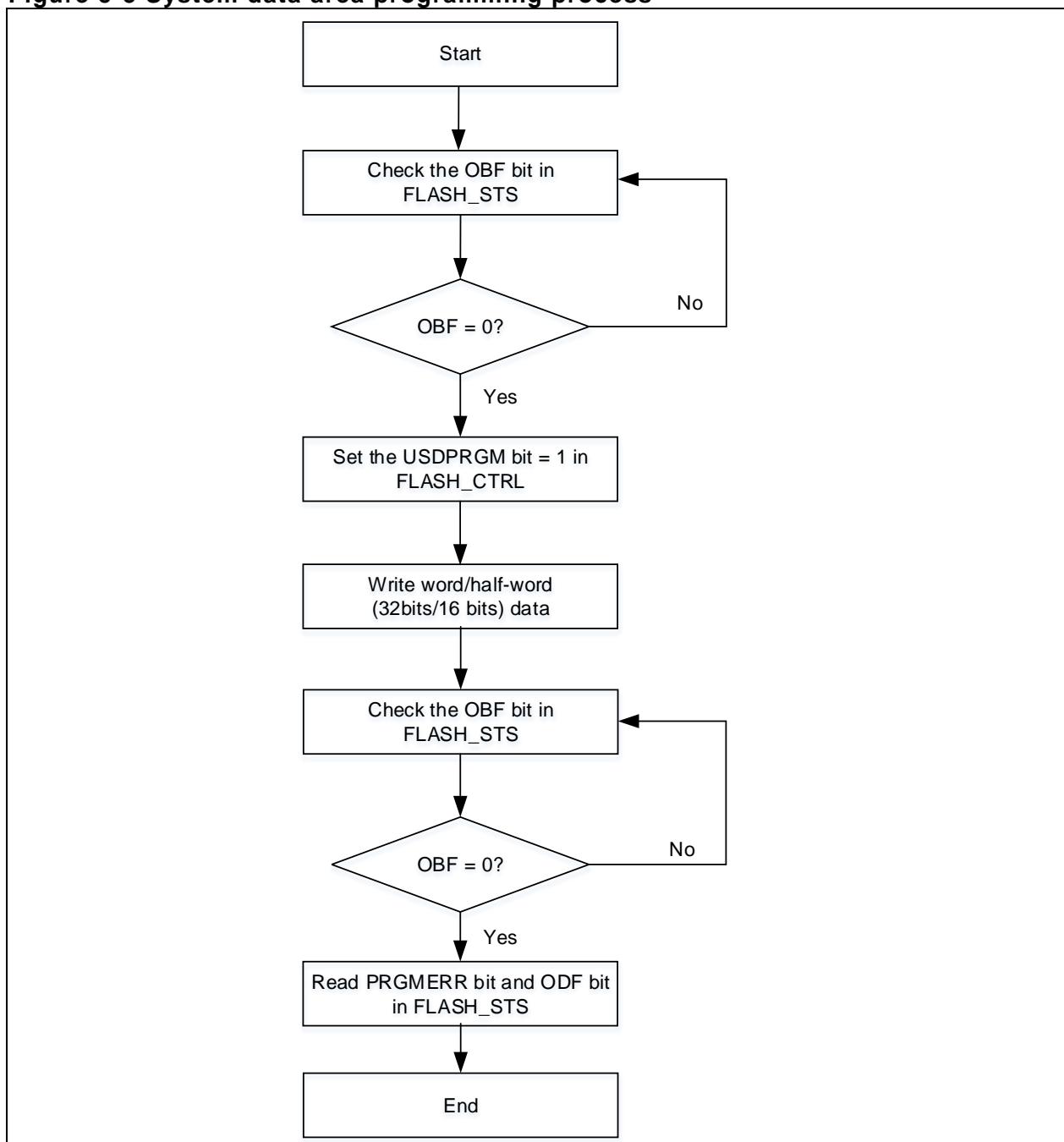
The User system data area can be programmed with 16-bit or 32-bit data at one time.

The following process is recommended:

- Check that no Flash memory operation is ongoing by checking the OBF bit in the FLASH\_STS register
- Set the USDPRGM bit in the FLASH\_CTRL register, so that the programming instructions for the user system data area can be received;
- Write the data (half-word/word) to be programmed to the designated address;
- Wait until the OBF bit in the FLASH\_STS register becomes “0”, read the PRGMERR and ODF bit to verify

*Note: Read operation to the Flash memory during programming halts CPU until the completion of programming. The internal HICK must be enabled prior to programming operation.*

**Figure 5-5 System data area programming process**



## 5.4.4 Read operation

User system data area can be accessed through AHB bus of the CPU.

## 5.5 Flash memory protection

Flash memory includes access and erase/program protection.

### 5.5.1 Access protection

Flash memory access protection is divided into two parts: high-level and low-level.

Once enabled, only the Flash program is allowed to read Flash memory data. This read operation is not permitted in debug mode or by booting from non-Flash memory.

#### Low-level access protection

When the contents in the nFAP and FAP bytes are different from 0x5A and 0xA5, and 0x33 and 0xCC, the low-level Flash memory access protection is enabled after a system reset.

When the Flash access is protected, the user can re-erase the system data area, and unlock Flash access protection (switching from low-level protected to unprotected state will trigger mass erase on the Flash memory and Flash memory extension area automatically) by writing 0xA5 to FAP byte, and then perform a system reset. Subsequently, the system data loader will be reloaded with system data and updated with Flash memory access protection disable state (FAP byte)

#### High-level access protection

When the content in the nFAP is equal to 0x33 , and the content in the FAP byte is equal to 0xCC, the high-level Flash memory access protection is enabled after a system reset.

Once enabled, it is not permissible for users to re-erase and write the system data area. This protection can be unlocked only with hardware by setting the FAP\_HL\_DIS bit.

*Note:*

- 1) *The main memory extension area can also be protected.*
- 2) *If the access protection bit is set in debug mode, then the debug mode has to be cleared by POR instead of system reset in order to resume access to Flash memory data*

**Table 5-5** shows Flash memory access limits when Flash access protection is enabled.

**Table 5-5 Flash memory access limit**

Block	Protection level	Access limits					
		In debug mode or boot from SRAM or boot loader code area			Boot from main Flash memory		
		Read	Write	Erase	Read	Write	Erase
Main Flash memory	Low-level protection	Not allowed		Not allowed <sup>(1)(2)</sup>	Accessible		
	High-level protection <sup>(4)</sup>		Not allowed		Accessible		
User system data area	Low-level protection	Not allowed		Accessible	Accessible		
	High-level protection <sup>(4)</sup>	Not allowed		Not allowed <sup>(3)</sup>	Not allowed		Not allowed <sup>(3)</sup>

(1) Main Flash memory is cleared automatically by hardware when the access protection is disabled;

(2) Only sector erase is forbidden, and mass erase is not affected;

(3) The user system data area can only be cleared by hardware through the FAP\_HL\_DIS bit;

(4) High-level Flash access protection is used to protect Flash memory against access and protect user system data area against erroneous erase operation.

## 5.5.2 Erase/program protection

For 64 K Flash memory and less, erase/program protection is performed on the basis of 4 sectors. This is used to protect the contents in the Flash memory against inadvertent operation when the program crash occurs.

Erase/program operation is not permitted under one of the following events, and the EPPERR bit is set accordingly:

- Attempting to erase/program sectors (in Flash memory and its extended area) which are already protected (erase/program protection enabled)
- Attempting to mass erase Flash memory and its extended area) in which one of the sectors is already protected (erase/program protection enabled)
- When the Flash access protection is enabled, the sector 0~3 in the main Flash memory will be protected against erase/program automatically. In this case, no sector erase/program is allowed.
- When the Flash access protection is enabled, the main Flash memory and its extended area are protected when they are in debug mode or when booting from non-main Flash memory. In this case, no sector erase/program is allowed.

## 5.6 Read access

To increase system clock frequency, it is necessary to program the number of wait states to access to Flash memory through the WTCYC bit in the FLASH\_PSR register.

The Flash read times can be decreased through the PFT\_EN, PFT\_DIS and PFT\_LAT\_DIS bits in the FLASH\_PSR register.

The HFCYC\_EN bit (when it is enabled) is used to save half-cycle system clock for Flash memory access, which is useful for continually reading large amount of constants. But there are some restrictions in system clock frequency. Please refer to the AT32F421 data sheet for more details.

## 5.7 Special functions

### 5.7.1 Security library settings

Security library is a defined area protected by a password in the main memory. This area is only executable but cannot be read (Except for I-Code and D-code buses), written, or deleted, unless a correct code is keyed in. Security library contains instruction security library and data security library.

#### Advantages of security library:

Security library is protected by a password so that solution providers can program core algorithm into this area;

Security library is executable, but cannot be read or deleted (including ISP/IAP/SWD) unless a correct password defined by the solution provider is keyed in;

The remaining blank area can be used for secondary development by solution providers;

Solution providers can sell core algorithm with security library function and do not have to develop a complete set of solutions for every customer.

Security library helps prevent from deliberate damage or changing terminal application codes.

#### Note:

*Security library code must be programmed on the sector level, with its start address being aligned with the main memory address;*

*Only I-Code bus is allowed to read instruction security library;*

*Only I-Code and D-Code bus are allowed to read the read-only area;*

*In an attempt of writing or deleting security library code, a warning message will be issued by setting WRPRFLR =1 in the FLASH\_STS register;*

*Executing mass erase in the main memory will not erase the security library.*

By default, security library setting register is unreadable and write protected. To enable write access to

this register, security library should be unlocked first by writing 0xA35F6D24 to the SLIB\_UNLOCK register. Then check the SLIB\_ULKF bit in the SLIB\_MISC\_STS register to verify if it is unlocked successfully. If successful, write the programmed value into the security library setting register.

To enable security library , follow the steps below:

- Check that no Flash memory operation is ongoing by checking the OBF bit in the FLASH\_STS register
- Write 0xA35F6D24 to the SLIB\_UNLOCK register to unlock security library.
- Check the SLIB\_ULKF bit of SLIB\_MISC\_STS register to verify if it is unlocked successfully.
- If the security library is located in Flash memory, then set the sectors to be protected (including the addresses of instruction and data areas) in the SLIB\_SET\_RANGE register; if the security library is located in the Flash extension area, then set the EM\_SLIB\_SET register
- Wait until the OBF bit becomes “0”
- Set a security library password in the SLIB\_SET\_PWD register
- Wait until the OBF bit becomes “0”
- Program the code to be saved in security library
- Perform a system reset, and then reload security library setting words
- Read the SLIB\_STS0/STS1 register to verify the security library settings

*Note:*

*The main Flash memory and Flash extension area are not intended to be configured as security library at the same time.*

*Security library must be enabled before the Flash access protection is activated.*

To unlock security library, follow the steps below:

- Write the previously set security library password to the SLIB\_PWD\_CLR register
- Wait until the OBF bit becomes “0”
- Perform a system reset, and then reload security library setting word
- Read the SLIB\_STS0 register to check the security library settings

*Note: Deactivating the security library will automatically perform mass erase on the main memory, main memory extension area and security library setting block.*

## 5.7.2 Boot memory used as memory extension area

There is only one chance for users to program the boot memory as the Flash memory extension area. Upon successful configuration, the extension area has the same features as those of Flash memory.

To configure boot memory as a Flash memory extension area, follow the procedure below:

- Check the current mode of the boot memory by reading the bit 0 in the SLIB\_STS0 register
- Write the value 0xA35F6D24 to the SLIB\_UNLOCK register to unlock the boot memory mode settings
- Write non-0xFF to the bit [7: 0] in the BTM\_MODE\_SET register
- Wait until the OBF bit becomes 0
- Perform a system reset, and reload setting words
- Read the SLIB\_STS0 register to verify

*Note: The main Flash extension area are must be set before the Flash access protection is activated.*

### 5.7.3 CRC verify

The sLib code or user code go through optional CRC check on the basis of a sector level.

CRC verify procedure as follows:

- Check that no Flash memory operation is ongoing by checking the OBF bit in the FLASH\_STS register
- Program the start address of the code to be verified in the FLASH\_CRC\_ADDR register
- Program the code count (in terms of sectors) to be verified through bit [15:0] in the FLASH\_CRC\_CTR register
- Enable CRC verify by setting the bit 16 of the FLASH\_CRC\_CTR register
- Wait until the OBF bit becomes 0
- Read the FLASH\_CRC\_CHK register to verify

*Note: The values of the FLASH\_CRC\_ADDR register must be aligned with the start address of the sector; CRC verify must not cross the main Flash memory and its extension area boundary.*

## 5.8 Flash memory registers

These peripheral registers must be accessed by words (32 bits).

**Table 5-6 Flash memory interface—Register map and reset value**

Register	Offset	Reset value
FLASH_PSR	0x00	0x0000 0030
FLASH_UNLOCK	0x04	0xXXXX XXXX
FLASH_USD_UNLOCK	0x08	0xXXXX XXXX
FLASH_STS	0x0C	0x0000 0000
FLASH_CTRL	0x10	0x0002 0080
FLASH_ADDR	0x14	0x0000 0000
FLASH_USD	0x1C	0x03FF FFFC
FLASH_EPPS	0x20	0xFFFF FFFF
SLIB_STS0	0x74	0x00FF 0000
SLIB_STS1	0x78	0xFFFF FFFF
SLIB_PWD_CLR	0x7C	0xFFFF FFFF
SLIB_MISC_STS	0x80	0x0000 0000
FLASH_CRC_ADDR	0x84	0x0000 0000
FLASH_CRC_CTRL	0x88	0x0000 0000
FLASH_CRC_CHK	0x8C	0x0000 0000
SLIB_SET_PWD	0x160	0x0000 0000
SLIB_SET_RANGE	0x164	0x0000 0000
EM_SLIB_SET	0x168	0x0000 0000
BTM_MODE_SET	0x16C	0x0000 0000
SLIB_UNLOCK	0x170	0x0000 0000

### 5.8.1 Flash performance select register (FLASH\_PSR)

Bit	Abbr.	Reset value	Type	Description
Bit 31: 9	Reserved	0x000000	resd	Kept at its default value.
Bit 8	PFT_LAT_DIS	0	rw	Prefetch latency disable 0: Prefetch latency of Flash is enabled, meaning that accessing buffer requires one system clock cycle 1: Prefetch latency of Flash is disabled, meaning that accessing buffer requires no wait state. Note: It is recommended to set this bit to 1 and remain unchanged.
Bit 7	PFT_ENF2	0	ro	Prefetch enable flag 2 When this bit is set, it indicates that the Flash prefetch buffer block 2 is enabled.
Bit 6	PFT_EN2	0	rw	Prefetch enable 2 0: Prefetch buffer block 2 is disabled 1: Prefetch buffer block 2 is enabled. Note: It is recommended to set this bit to 1 and remain unchanged.
Bit 5	PFT_ENF	1	ro	Prefetch enable flag When this bit is set, it indicates that the Flash prefetch buffer is enabled.
Bit 4	PFT_EN	1	rw	Prefetch enable 0: Prefetch is disabled 1: Prefetch is enabled.
Bit 3	HFCYC_EN	0x0	rw	Half cycle accelerated access enable 0: Disabled 1: Enabled Note: There are some limitations in system clock frequency for this feature, refer to the AT32F421 data sheet for details.
Bit 2: 0	WTCYC	0x0	rw	Wait cycle The wait states for Flash access depend on the size of the system clock, and they are in terms of system clocks. 000: Zero wait state when $0\text{MHz} < \text{system clock} \leq 32\text{MHz}$ 001: One wait state when $32\text{MHz} < \text{system clock} \leq 64\text{MHz}$ 010: Two wait states when $64\text{MHz} < \text{system clock} \leq 96\text{MHz}$ 011: Three wait states when $96\text{MHz} < \text{system clock} \leq 120\text{MHz}$

### 5.8.2 Flash unlock register (FLASH\_UNLOCK)

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	UKVAL	0XXXX XXXX	wo	Unlock key value This is used to unlock Flash memory bank and its extension area.

Note: All these bits are write-only, and return 0 when being read.

### 5.8.3 Flash user system data unlock register (FLASH\_USD\_UNLOCK)

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	USD_UKVAL	0XXXX XXXX	wo	User system data Unlock key value

Note: All these bits are write-only, and return 0 when being read.

## 5.8.4 Flash status register (FLASH\_STS)

Bit	Abbr.	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value Operation done flag
Bit 5	ODF	0	rw1c	This bit is set by hardware when Flash memory operations (program/erase) are complete. It is cleared by writing "1". Erase/program protection error
Bit 4	EPPERR	0	rw1c	This bit is set by hardware when programming the erase/program- protected Flash memory address. It is cleared by writing "1".
Bit 3	Reserved	0	resd	Kept at its default value. Programming error
Bit 2	PRGMERR	0	rw1c	When the Flash programming address is in "non-erase" state, this bit is set by hardware. It is cleared by writing "1".
Bit 1	Reserved	0	resd	Kept at its default value. Operation busy flag
Bit 0	OBF	0	ro	When this bit is set, it indicates that Flash memory operation is in progress. It is cleared when operation is complete.

## 5.8.5 Flash control register (FLASH\_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0000	resd	Kept at its default value
				Low power mode enable 0: Low power mode disabled
Bit 17	LPMEN		1	1: Low power mode enabled When this bit is set, the Flash controller lets Flash memory go to low-power mode as soon as MCU enters Deepsleep mode.
				High level Flash access protection disable When this bit is set, the user system data area is automatically cleared by hardware; After a reset, it is unlocked, and low-level access protection is still present. .This bit is automatically cleared by hardware by writing 1 to it.
Bit 16	FAP_HL_DIS	0x0	rw	
				Kept its default value
Bit 15: 13	Reserved	0x0	resd	Operation done flag interrupt enable
Bit 12	ODIFE	0	rw	0: Operation complete interrupt disabled 1: Operation complete interrupt enabled
Bit 11,8,3	Reserved	0	resd	Kept its default value
				Error interrupt enable This bit enables EPPERR or PROGERR interrupt. 0: Interrupt is disabled; 1: Interrupt is enabled.
Bit 10	ERRIE	0	rw	User system data unlock success This bit is set by hardware when the user system data is unlocked successfully, indicating that erase/program operation to the user system data is allowed. This bit is cleared by writing "0", which will re-lock the user system data area.
				Operation lock This bit is set by default, indicating that Flash memory is protected against operations. This bit is cleared by hardware after unlock, indicating that erase/program operation to Flash memory is allowed. Writing "1" can re-lock Flash memory operations.
Bit 7	OPLK	1	rw	
				Erase start An erase operation is triggered when this bit is set by software. This bit is cleared by hardware after the completion of the erase operation.
Bit 6	ERSTR	0	rw	
				User system data erase It indicates the user system data erase.
Bit 5	USDERS	0	rw	
				User system data program It indicates the user system data program.
Bit 4	USDPRGM	0	rw	
				Bank erase It indicates bank erase operation.
Bit 2	BANKERS	0	rw	
				Sector erase It indicates sector erase operation.
Bit 1	SECERS	0	rw	
				Flash program It indicates Flash program operation.
Bit 0	FPRGM	0	rw	

## 5.8.6 Flash address register (FLASH\_ADDR)

Bit	Register	Reset value	Type	Description
Bit 31: 0	FA	0x0000 0000	wo	Flash address This is used to select the address of the sectors to be erased.

## 5.8.7 User system data register (FLASH\_USD)

Bit	Register	Reset value	Type	Description
Bit 31: 27	Reserved	0x00	resd	Kept at its default value
Bit 26	FAP_HL	0	ro	High level Flash access protection The status of the Flash access protection is determined by bit 26 and bit 1. 00: Flash access protection disabled, and FAP=0xA5 01: Low-level Flash access protection enabled, and FAP=non-0xCC and 0xA5. 10: Reserved 11: High-level Flash access protection, and FAP=0xCC
Bit 25: 18	USER_D1	0xFF	ro	User data 1
Bit 17: 10	USER_D0	0xFF	ro	User data 0
Bit 9: 2	SSB	0xFF	ro	System setting byte Includes the system setting bytes in the loaded user system data area Bit 9: 7: Unused Bit 6: nBOOT1 Bit 5: Unused Bit 4: nSTDBY_RST Bit 3: nDEPSLP_RST Bit 2: nWDT_ATO_EN
Bit 1	FAP	0	ro	Flash access protection Access to Flash memory is not allowed when this bit is set.
Bit 0	USDERR	0	ro	User system data error When this bit is set, it indicates that certain byte does not match its inverse code in the user system data area. At this point, this byte and its inverse code will be forced to 0xFF when being read.

## 5.8.8 Erase/program protection status register (FLASH\_EPPS)

Bit	Register	Reset value	Type	Description
Bit 31: 0	EPPS	0xFFFF FFFF	ro	Erase/Program protection status This register reflects the erase/program protection byte status in the loaded user system data.

## 5.8.9 Flash security library status register 0 (SLIB\_STS0)

For Flash memory security library only.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value
				Extension memory sLib instruction start sector 00000000: sector 0 00000001: sector 1 00000010: sector 2 00000011: sector 3 11111111: No instruction sLib Others: invalid
Bit 23: 16	EM_SLIB_INST_SS	0x00	ro	00000011: sector 3 11111111: No instruction sLib Others: invalid
Bit 15: 4	Reserved	0x000	resd	Kept at its default value
				sLib enable flag
Bit 3	SLIB_ENF	0	ro	When this bit is set, it indicates that the main Flash memory is partially or completely (depending on the setting of SLIB_STS1) used as security library code.
				Extension memory sLib enable flag
Bit 2	EM_SLIB_ENF	0	ro	When this bit is set, it indicates that the boot memory is used as the Flash extension area (BTM_AP_ENF is set), and stores security library code.
Bit 1	Reserved	0	resd	Kept at its default value
				Boot memory stores application code enabled flag
Bit 0	BTM_AP_ENF	0	ro	When this bit is set, it indicates that the boot memory can be used as main Flash extension area to store user application code; otherwise, it is only used for storing system boot code.

## 5.8.10 Flash security library status register1 (SLIB\_STS1)

For Flash memory security library only.

Bit	Register	Reset value	Type	Description
				Security library end sector 0000000000: sector 0 0000000001: sector 1 0000000010: sector 2 ...
Bit 31: 22	SLIB_ES	0x3FF	ro	0000001111: sector 15 (the last sector of 16KB main Flash memory) ...
				0000011111: sector 31 (the last sector of 32KB main Flash memory) ...
				0000111111: sector 63 (the last sector of 64KB main Flash memory)
				Security library instruction start sector 0000000000: sector 0 0000000001: sector 1 0000000010: sector 2 ...
Bit 21: 11	SLIB_INST_SS	0x7FF	ro	00000001111: sector 15 (the last sector of 16KB main Flash memory) ...
				00000011111: sector 31 (the last sector of 32KB main Flash memory) ...
				00000111111: sector 63 (the last sector of 64KB main Flash memory) 111111111111: No sLib instruction area
Bit 10: 0	SLIB_SS	0x7FF	ro	Security library start sector 0000000000: sector 0

---

000000000001: sector 1  
000000000010: sector 2  
...  
000000011111: sector 15 (the last sector of 16KB main Flash memory)  
...  
000000111111: sector 31 (the last sector of 32KB main Flash memory)  
...  
000001111111: sector 63 (the last sector of 64KB main Flash memory)

---

### 5.8.11 Security library password clear register (SLIB\_PWD\_CLR)

For Flash memory security library only.

Bit	Register	Reset value	Type	Description
Bit 31: 0	SLIB_PCLR_VAL	0x0000 0000	wo	Security library password clear value This register is used to key in a correct sLib password in order to unlock sLib function. The write status of this register is indicated by bit 0 and bit 1 of the SLIB_MISC_STS register.

---

### 5.8.12 Security library additional status register (SLIB\_MISC\_STS)

For Flash memory security library only.

Bit	Register	Reset value	Type	Description
Bit 31:3	Reserved	0x0000 0000	resd	Kept at its default value
Bit 2	SLIB_ULKF	0	ro	Security library unlock flag When this bit is set, it indicates that sLib-related setting registers can be configured.
Bit 1	SLIB_PWD_OK	0	ro	Security library password ok This bit is set by hardware when the password is correct.
Bit 0	SLIB_PWD_ERR	0	ro	Security library password error This bit is set by hardware when the password is incorrect and the setting value of the password clear register is different from 0xFFFF FFFF. Note: After this bit is set, the hardware will no longer agree to re-program the password clear register until the next reset.

---

### 5.8.13 Flash CRC address register (FLASH\_CRC\_ARR)

For main Flash memory and its extension area only.

Bit	Register	Reset value	Type	Description
Bit 31:0	CRC_ADDR	0x0000 0000	wo	CRC address This register is used to select the start address of a sector to be CRC checked..

---

*Note: All these bits are write-only, and return no response when being read.*

## 5.8.14 Flash CRC control register (FLASH\_CRC\_CTRL)

For main Flash memory and its extension area only.

Bit	Register	Reset value	Type	Description
Bit 31:17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	CRC_STRT	0x0	W0	CRC start This bit is used to enable CRC check for user code or sLib code. It is automatically cleared after enabling CRC by hardware. Note: CRC data ranges from CRC_ADDR to CRC_ADDR+CRC_SN*1
Bit 15: 0	CRC_SN	0x0000	wo	CRC sector number This bit defines the sectors to be CRC checked.

## 5.8.15 Flash CRC check result register (FLASH\_CRC\_CHK)

For Flash memory and its extension area only.

Bit	Register	Reset value	Type	Description
Bit 31: 0	CRC_CHK	0x0000 0000	ro	CRC check result

*Note: All these bits are write-only, and return no response when being read.*

## 5.8.16 Security library password setting register (SLIB\_SET\_PWD)

For Flash security library password setting only.

Bit	Register	Reset value	Type	Description
Bit 31: 0	SLIB_PSET_VAL	0x0000 0000	ro	SLIB password setting value Note: This register can be written only after sLib is unlocked. It is used to set a password of sLib. Writing 0xFFFF_FFFF or 0x0000_0000 has no effect.

*Note: All these bits are write-only, and return 0 when being read.*

## 5.8.17 Security library address setting register (SLIB\_SET\_RANGE)

For Flash security library address setting only.

Bit	Register	Reset value	Type	Description
Bit 31: 22	SLIB_ES_SET	0x000	wo	Security library end sector setting These bits are used to set the security library end sector. 0000000000: Sector 0 0000000001: Sector 1 0000000010: Sector 2 ... 0000001111: Sector 15 (the last sector of 16KB main Flash memory) ... 0000011111: Sector 31 (the last sector of 32KB main Flash memory) ... 0000111111: Sector 63 (the last sector of 64KB main Flash memory)
Bit 21: 11	SLIB_ISS_SET	0x000	wo	Security library instruction start sector setting These bits are used to set the security library instruction start sector. 0000000000: Sector 0 0000000001: Sector 1 0000000010: Sector 2

Bit 10: 0	SLIB_SS_SET	0x000	wo	...
				00000001111: Sector 15 (the last sector of 16KB main Flash memory)
				...
				00000011111: Sector 31 (the last sector of 32KB main Flash memory)
				...
				00000111111: Sector 63 (the last sector of 64KB main Flash memory)
				11111111111: No sLib instruction area
				Security library start sector setting
				These bits are used to set the security library start sector.
				00000000000: Sector 0
				00000000001: Sector 1
				00000000010: Sector 2
				...
				00000001111: Sector 15 (the last sector of 16KB main Flash memory)
				...
				00000011111: Sector 31 (the last sector of 32KB main Flash memory)
				...
				00000111111: Sector 63 (the last sector of 64KB main Flash memory)

---

**Note:**

All these bits are write-only, and return 0 when being read.

This register can be written only after security library is unlocked.

Being out of the Flash address range is an invalid setting.

### 5.8.18 Flash extension memory security library setting register (EM\_SLIB\_SET)

For Flash extension area only.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value
				Extension memory sLib instruction start sector 00000000: Sector 0 00000001: Sector 1 00000010: Sector 2 00000011: Sector 3
Bit 23: 16	EM_SLIB_ISS_SET	0x000	wo	11111111: No sLib instruction area Others: Invalid  Note: When this bit is set to 0xFF, it indicates that the Flash memory extension area from sector 0 to sector 3 is defined as the security library, and the entire security library is read-only.
Bit 15: 0	EM_SLIB_SET	0x000	wo	Extension memory sLib setting By writing 0x5AA5, the extension memory is configured as an area to store security library code.

*Note: All these bits are write-only, and return no response when being read.*

### 5.8.19 Boot memory mode setting register (BTM\_MODE\_SET)

For boot memory only.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0000000	resd	Kept at its default value.
				Boot memory mode setting 0XFF: Boot memory serves as a system area that stores system boot code
Bit 7: 0	BTM_MODE_SET	0x00	wo	Others: Boot memory serves a Flash extension area that stores application code  Note: This register can be set only when Flash access protection is disabled.

*Note: All these bits are write-only, and return no response when being read.*

### 5.8.20 Security library unlock register (FLASH\_UNLOCK)

For security library register unlock only.

Bit	Register	Reset value	Type	Description
Bit 31: 0	SLIB_UKVAL	0x0000 0000	wo	Security library unlock key value The fixed key value is 0xA35F_6D24, which is used for the unlocking of security library setting register.

*Note: All these bits are write-only, and return 0 when being read.*

# 6 General-purpose I/Os (GPIOs)

## 6.1 Introduction

AT32F421 series supports up to 39 bidirectional I/O pins, namely PA0-PA15, PB0-PB15, PC13-PC15, PF0-PF1 and PF6-PF7. Each of the GPIO group supports external communication, with control and data collection feature. In addition, their main features also include:

- Supports general-purpose I/O (GPIO) or multiplexed function I/O (IOMUX)
- Each pin can be configured by software as floating input, pull-up/pull-down input, analog input/output, push-pull/open-drain output, multiplexed push-pull/open-drain output
- Each pin has its respective weak pull-up/pull-down feature
- Each pin's output drive capability is configurable by software
- Each pin can be configured as external interrupt input
- Each pin can be locked

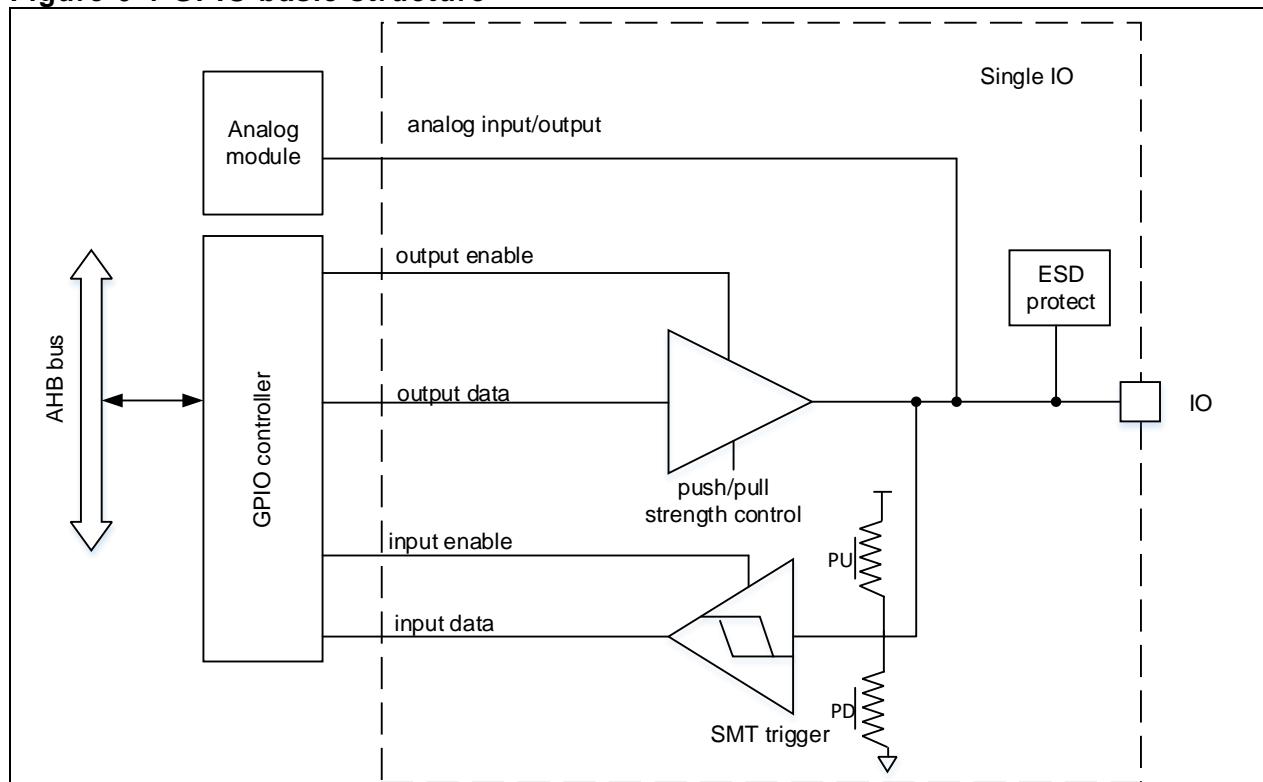
## 6.2 Functional overview

### 6.2.1 GPIO structure

Each of the GPIO pins can be configured by software as four input modes (floating, pull-up/pull-down and analog input) and four output modes (open-drain, push-pull, alternate function push-pull/open-drain output)

Each I/O port bit can be programmed freely. However, I/O port registers can be accessed by bytes, half-bytes or words.

**Figure 6-1 GPIO basic structure**



### 6.2.2 GPIO reset status

After power-on or system reset, all pins are configured as floating input mode except SWD-related pins. SWD-related pins are configured as follows:

- PA13/SWDIO in alternate function pull-up mode;
- PA14/SWCLK in alternate function pull-down mode.

### 6.2.3 General-purpose input configuration

Mode	IOMC	PUPD
Floating input		00
Pull-down input	00	10
Pull-up input		01

When I/O port is configured as input:

- Get I/O states by reading the input data register.
- Support floating input, pull-up/pull-down input configuration.
- Schmitt-trigger input is activated.
- Output is disabled.

*Note: In floating input mode, it is recommended to set the unused pins as analog input mode in order to avoid leakage caused by interference from unused pins in a complex environment.*

### 6.2.4 Analog input/output configuration

Mode	IOMC	PUPD
Analog input/output	11	Unused

When I/O port is configured as analog input:

- Schmitt-trigger input is disabled.
- Digital input/output is disabled.
- Without any pull-up/pull-down resistor.

### 6.2.5 General-purpose output configuration

Mode	IOMC	OM	HDRV	ODRV[1: 0]	PUPD
Push-pull without pull-up/pull-down	01	0	000: Output mode, normal sourcing/sinking strength 001: Output mode, large sourcing/sinking strength 010: Output mode, normal sourcing/sinking strength 011: Output mode, normal sourcing/sinking strength 1xx: Output mode, Maximum sourcing/sinking strength	Any value	
Open-drain without pull-up/pull-down	01	1	000: Output mode, normal sourcing/sinking strength 001: Output mode, large sourcing/sinking strength 010: Output mode, normal sourcing/sinking strength 011: Output mode, normal sourcing/sinking strength 1xx: Output mode, Maximum sourcing/sinking strength	Any value	

When I/O port is configured as output:

- Schmitt-trigger input is enabled
- Output through output register
- Pull-up/pull-down resistors are disabled
- In open-drain mode, forced output 0, and use pull-up resistor to output 1
- In push-pull mode, output 0/1 using output register
- GPIO set/clear register is used to set or clear the corresponding GPIO data output register

*Note: When writing 1 to the IOCB/IOSB bits of the GPIO set/clear register, IOSB has priority over IOCB.*

### 6.2.6 GPIO port protection

Locking mechanism is available to freeze the I/O configuration for the purpose of protection. When LOCK is applied to a port bit, its configuration cannot be modified until the next reset or power on.

## 6.2.7 IOMUX structure

Most of the pins supports output function mapping for multiple peripherals. It is possible to select the peripheral input/output functions for each pin by using the IOMUX input/output checklist described in the section of IOMUX input/output. The multiplexed function of pins is configured using the corresponding GPIO multiplexed register low (GPIO<sub>x</sub>\_MUXL) (for pin 0 ~ pin 7) or the GPIO multiplexed register high (GPIO<sub>x</sub>\_MUXH) (for pin 8 ~ pin 15). A single pin has up to 16 different IOMUX mapping configurations for flexible selection.

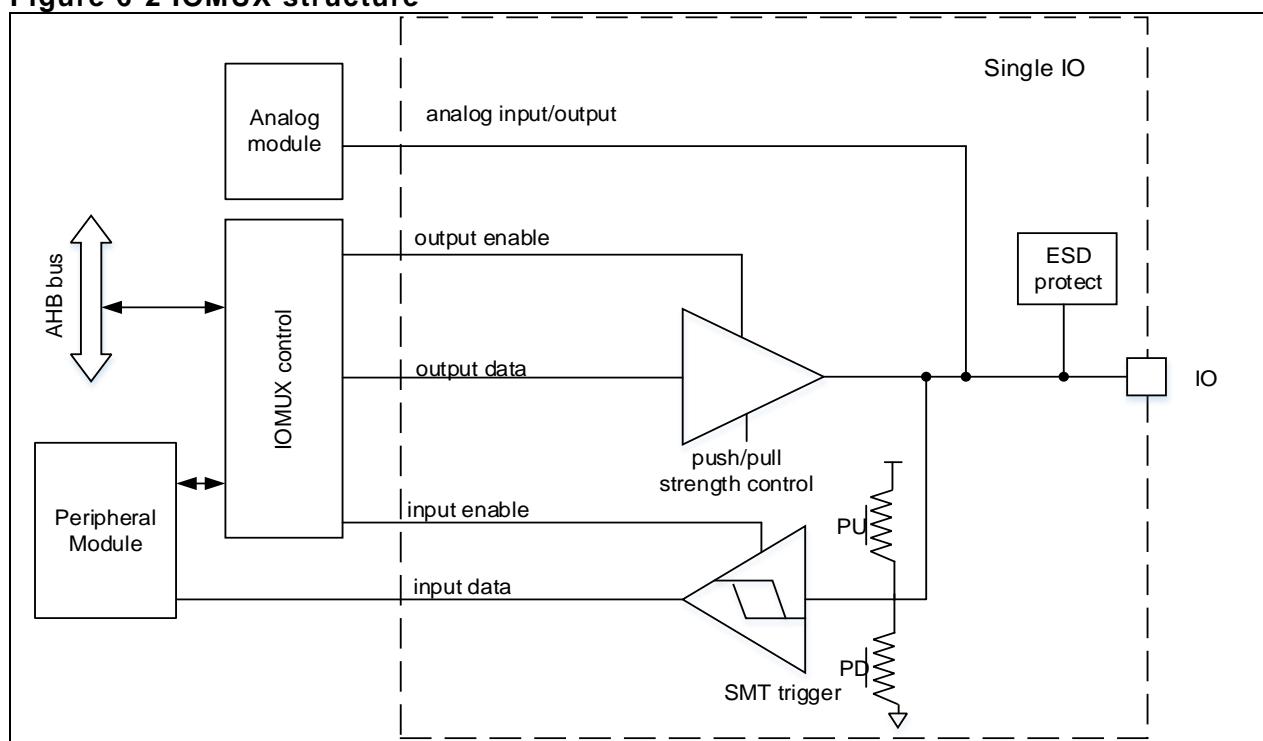
It is only allowed to select one of the peripheral function for each pin by setting the GPIO<sub>x</sub>\_MUXL or GPIO<sub>x</sub>\_MUXH register. Hence, it is not possible that a single pin is occupied by several peripherals at a time.

When used as multiplexed function input, the I/O port should be configured as input modes (floating, pull-up and pull-down), the same as those of general-purpose input functions.

To enable multiplexed function output, the port must be configured as multiplexed function output mode and push-pull or open-drain by setting GPIO<sub>x</sub>\_CFG or GPIO<sub>x</sub>\_OMODE register. In this case, the pin is disconnected from GPIO controller, and controlled by IOMUX controller, instead.

To achieve bidirectional multiplexed function, the port needs to be configured as multiplexed function output modes (push-pull or open-drain), controlled by IOMUX controller.

**Figure 6-2 IOMUX structure**



## 6.2.8 Multiplexed function input configuration

Mode	IOMC	PUPD
Multiplexed floating input		00
Multiplexed pull-down input	10	10
Multiplexed pull-up input		01

When I/O ports are configured as multiplexed function input:

- Get I/O pin state by reading input data registers
- Support floating input, pull-up or pull-down input configuration
- Schmitt-trigger input is activated.
- Pin output is disabled.

## 6.2.9 IOMUX function input/output

The selection of the valid multiplexed functions for each port is done by the GPIOx\_MUXL (for pin 0 to pin 7) or GPIOx\_MUXH (for pin 8 to pin 15) registers.

**Table 6-1 Multiplexed function configuration for port A using GPIO\_A MUX\* register**

Pin name	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PA0		USART2_CTS			I2C2_SC_L	TMR1_E_TR		COMP_OUT
PA1	EVENTOUT	USART2_RTS			I2C2_SD_A	TMR15_CH1N		
PA2	TMR15_CH1	USART2_TX						
PA3	TMR15_CH2	USART2_RX				I2S2_MCLK		
PA4	SPI1_CS/I2S1_WS	USART2_CK			TMR14_CH1			
PA5	SPI1_SCK/I2S1_CK							
PA6	SPI1_MISO/I2S1_MCLK	TMR3_CH1	TMR1_BKI_N	I2S2_MCLK		TMR16_CH1	EVENTOUT	COMP_OUT
PA7	SPI1_MOSI/I2S1_SD	TMR3_CH2	TMR1_CH1N		TMR14_CH1	TMR17_CH1	EVENTOUT	
PA8	CLKOUT	USART1_CK	TMR1_CH1	EVENTOUT	USART2_TX			I2C2_SCL
PA9	TMR15_BKI_N	USART1_TX	TMR1_CH2		I2C1_SC_L	CLKOUT		I2C2_SMB
PA10	TMR17_BKI_N	USART1_RX	TMR1_CH3		I2C1_SD_A			
PA11	EVENTOUT	USART1_CTS	TMR1_CH4		I2C1_SMB	I2C2_SC_L		COMP_OUT
PA12	EVENTOUT	USART1_RTS	TMR1_ETR			I2C2_SD_A		
PA13	SWDIO	IR_OUT					SPI2_MISO/I2S2_MCLK	
PA14	SWCLK	USART2_TX					SPI2_MOSI/I2S2_SD	
PA15	SPI1_CS/I2S1_WS	USART2_RX		EVENTOUT			SPI2_CS_I2S2_WS	

**Table 6-2 Multiplexed function configuration for port B using GPIO\_B MUX\* register**

Pin name	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PB0	EVENTOUT	TMR3_C_H3	TMR1_CH2N	USART2_RX			I2S1_MCLK	
PB1	TMR14_CH1	TMR3_C_H4	TMR1_CH3N				SPI2_SCK/I2S2_CK	
PB2			TMR3_ETR					
PB3	SPI1_SCK/I2S1_CK	EVENTOUT					SPI2_SCK/I2S2_CK	
PB4	SPI1_MISO/I2S1_MCLK	TMR3_C_H1	EVENTOUT			TMR17_BKIN	SPI2_MISO/I2S2_MCLK	I2C2_SDA
PB5	SPI1_MOSI/I2S1_SD	TMR3_C_H2	TMR16_B_KIN	I2C1_SMBA			SPI2_MOSI/I2S2_SD	
PB6	USART1_TX	I2C1_SC_L	TMR16_C_H1N				I2S1_MCLK	
PB7	USART1_RX	I2C1_SD_A	TMR17_C_H1N					
PB8		I2C1_SC_L	TMR16_C_H1					
PB9	IR_OUT	I2C1_SD_A	TMR17_C_H1	EVENTOUT		I2S1_MCLK		SPI2_CS/I2S2_WS
PB10		I2C2_SC_L						SPI2_SCK/I2S2_CK
PB11	EVENTOUT	I2C2_SD_A						
PB12	SPI2_CS/I2S2_WS	EVENTOUT	TMR1_BKI_N			TMR15_BKIN		I2C2_SMBA
PB13	SPI2_SCK/I2S2_CK	-	TMR1_CH1N			I2C2_SCL		
PB14	SPI2_MISO/I2S2_MCLK	TMR15_CH1	TMR1_CH2N			I2C2_SDA		
PB15	SPI2_MOSI/I2S2_SD	TMR15_CH2	TMR1_CH3N	TMR15_CH1N				

**Table 6-3 Multiplexed function configuration for port F using GPIO\_F MUX\* register**

Pin name	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PF0		I2C1_SD_A						
PF1		I2C1_SC_L						
PF6	I2C2_SC_L							
PF7	I2C2_SD_A							

## 6.2.10 Peripheral multiplexed function configuration

When IOMUX multiplexed function is needed:

- To use a peripheral pin as a multiplexed output, the corresponding pin should be configured as multiplexed push-pull/open-drain output.
- To use a peripheral pin as a MUX input, the corresponding pin configured as floating input/pull-up/pull-down input.
- For ADC peripherals, the pin corresponding to analog channel should be configured as analog input/output mode.
- For I2C peripherals that intend to use a pins as bidirectional function, the corresponding pin should be configured as open-drain mode.

## 6.2.11 IOMUX map priority

Almost all peripherals can be configured as corresponding multiplexed functions using GPIOx\_MUXL or GPIOx\_MUXH registers, except for a few pins that might be occupied by hardware directly.

Some of the pins, no matter whatever GPIO mode configuration, are always owned by specific hardware feature directly.

**Table 6-4 Hardware preemption**

Pin	Enable bit	Description
PA0	PWC_CTRLSTS[8] =1	Once enabled, PA0 pin acts as WKUP1 function of PWC.
PB5	PWC_CTRLSTS[13] =1	Once enabled, PB5 pin acts as WKUP6 function of PWC
PB15	PWC_CTRLSTS[14] =1	Once enabled, PB15 pin acts as WKUP7 function of PWC
PC13	PWC_CTRLSTS[9] = 1	Once enabled, PA0 pin acts as WKUP2 function of PWC
PC13	(ERTC_CTRL[23:21] !=3'b000) (ERTC_CTRL[11] !=0)  (ERTC_TAMP[0] != 0)	Once enabled, PC13 pin acts as RTC channel input/output
PC14	CMR_BPDC[0]=1	Once enabled, PC14 pin acts as LEXT channel
PC15	CMR_BPDC[0]=1	Once enabled, PC15 pin acts as LEXT channel
PF0	CMR_CTRL[16]=1	Once enabled, PF0 pin acts as HEXT channel
PF1	CMR_CTRL[16]=1	Once enabled, PF1 pin acts as HEXT channel

## 6.2.12 External interrupt/wake-up lines

Each pin can be used as an external interrupt input and thus the corresponding pin must be configured as input mode.

## 6.3 GPIO registers

**Table 6-5** lists GPIO register map and their reset values. These peripheral registers must be accessed by bytes (8-bit), half-words (16-bit) or words (32 bits).

**Table 6-5 GPIO register map and reset values**

Register	Offset	Reset value
GPIOA_CFGR	0x00	0x2800 0000
GPIOx_CFGR(x = B,C,F)	0x00	0x0000 0000
GPIOx_OMODE	0x04	0x0000 0000
GPIOA_ODRVR	0x08	0x0C00 0000
GPIOx_ODRVR(x = B,C,F)	0x08	0x0000 0000

GPIOA_PULL	0x0C	0x2400 0000
GPIOx_PULL(x = B,C,F)	0x0C	0x0000 0000
GPIOx_IDT	0x10	0x0000 XXXX
GPIOx_ODT	0x14	0x0000 0000
GPIOx_SCR	0x18	0x0000 0000
GPIOx_WPR	0x1C	0x0000 0000
GPIOx_MUXL	0x20	0x0000 0000
GPIOx_MUXH	0x24	0x0000 0000
GPIOx_CLR	0x28	0x0000 0000
GPIOx_HDRV	0x3C	0x0000 0000

### 6.3.1 GPIO configuration register (GPIOx\_CFGR) (x=A...H)

Address offset: 0x00

Reset value: 0x28000000 for port A

0x00000000 for other ports

Bit	Register	Reset value	Type	Description
Bit 2y+1: 2y	IOMCy	0x2800 0000	rw	GPIOx mode configuration (y=0~15) These bits are used to configure the operating modes of GPIOx: 00: Input mode (after reset) 01: General-purpose output 10: Multiplexed function 11: Analog

### 6.3.2 GPIO input mode register (GPIOx\_OMODE) (x=A...H)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Always 0
Bit 15: 0	OM	0x0000	rw	GPIOx output mode configuration (y=0..15) When GPIOx is used as output, there are two output modes for selection: 0: Push-pull (reset state) 1: Open-drain

### 6.3.3 GPIO drive capability register (GPIOx\_ODRVR) (x=A..H)

Address offset: 0x08

Reset values: 0x0C00 0000 for port A 0x00000000 for other ports

Bit	Register	Reset value	Type	Description
Bit 2y+1: 2y	ODRVy	0x0000 0000	rw	GPIOx drive capability (y=0...15) This field is used to configure the IO port drive capability. x0: Normal sourcing/sinking strength 01: Large sourcing/sinking strength 11: Normal sourcing/sinking strength

### 6.3.4 GPIO pull-up/pull-down register (GPIOx\_PULL) (x=A..H)

Address offset: 0x0C

Reset values: 0x2400 0000 for port A 0x00000000 for other ports

Bit	Register	Reset value	Type	Description
Bit 2y+1: 2y	PULLy	0x6400 0000	rw	GPIOx pull-up/pull-down configuration (y=0...15) This field is used to configure the pull-up/pull-down of the IO port. 00: No pull-up, pull-down 01: Pull-up 10: Pull-down

### 6.3.5 GPIO input data register (GPIOx\_IDT) (x=A...H)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Always 0.
Bit 15: 0	IDT	0XXXX	ro	GPIOx input data Indicates the input status of I/O port. Each bit corresponds to an I/O.

### 6.3.6 GPIO output data register (GPIOx\_ODT) (x= A...H)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Always 0.
Bit 15: 0	ODT	0x0000	rw	GPIOx output data Each bit represents an I/O port. It indicates the output status of I/O port. 0: Low 1: High

### 6.3.7 GPIO set/clear register (GPIOx\_SCR) (x=A...H)

Bit	Register	Reset value	Type	Description
Bit 31: 16	IOCB	0x0000	wo	GPIOx clear bit The corresponding ODT register bit is cleared by writing “1” to these bits. Otherwise, the corresponding ODT register bit remains unchanged, which acts as ODT register bit operations. 0: No action to the corresponding ODT bits 1: Clear the corresponding ODT bits
Bit 15: 0	IOSB	0x0000	wo	GPIOx set bit The corresponding ODT register bit is set by writing “1” to these bits. Otherwise, the corresponding ODT register bit remains unchanged, which acts as ODT register bit operations. 0: No action to the corresponding ODT bits 1: Set the corresponding ODT bits

### 6.3.8 GPIO write protection register (GPIOx\_WPR) (x=A...H)

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	WPSEQ	0x0	rw	Write protect sequence Write protect enable sequence bit and WPEN bit must be enabled at the same time to achieve write protection for some I/O bits. Write protect enable bit is executed four times in the order below: write “1” -> write “0” -> write “1” -> read. Note that the value of WPEN bit cannot be modified during this period.
Bit 15: 0	WPEN	0x0000	rw	Write protect enable Each bit corresponds to an I/O port. 0: No effect. 1: Write protect

### 6.3.9 GPIO multiplexed function low register (GPIOx\_MUXL) (x=A..H)

Address offset: 0x20

Reset value: 0x00000000

Bit	Register	Reset value	Type	Description
Bit 4y+3: 4y	MUXLy	0x0	rw	Multiplexed function select for GPIOx pin y (y=0...7) This field is used to configure multiplexed function IOs. 0000: MUX0 0001: MUX1 0010: MUX2 0011: MUX3 0100: MUX4 0101: MUX5 0110: MUX6 0111: MUX7 1xxx: Reserved

### 6.3.10 GPIO multiplexed function high register (GPIOx\_MUXH) (x=A..H)

Bit	Register	Reset value	Type	Description
Bit 4y+3: 4y	MUXHy	0x0	rw	Multiplexed function select for GPIOx pin y (y=8...15) This field is used to configure multiplexed function IOs. 0000: MUX0 0001: MUX1 0010: MUX2 0011: MUX3 0100: MUX4 0101: MUX5 0110: MUX6 0111: MUX7 1xxx: Reserved

### 6.3.11 GPIO bit clear register (GPIOx\_CLR) (x=A...H)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	IOCB	0x0000	wo	GPIOx clear bit The corresponding ODT register bit is cleared by writing “1” to these bits. Otherwise, the corresponding ODT register bit remains unchanged, which acts as ODT register bit operations. 0: No action to the corresponding ODT bits 1: Clear the corresponding ODT bits

### 6.3.12 GPIO huge current control register (GPIOx\_HDRV) (x=A..H)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	HDRV	0x0000	rw	Huge sourcing/sinking strength control 0: Not active 1: GPIO is configured as maximum sourcing/sinking strength

# 7 System configuration controller (SCFG)

## 7.1 Introduction

This device contains a set of system configuration register. The system configuration controller is mainly used to:

- Remap some DMA trigger sources to other DMA channels
- Manage the external interrupts connected to the GPIOs

## 7.2 SCFG registers

**Table 7-1** shows SCFG register map and their reset values.

These peripheral registers must be accessed by words (32 bits).

**Table 7- 1 SCFG register map and reset value**

Register	Offset	Reset value
SCFG_CFG1	0x00	0x0000 000X
SCFG_EXINTC1	0x08	0x0000 0000
SCFG_EXINTC2	0x0C	0x0000 0000
SCFG_EXINTC3	0x10	0x0000 0000
SCFG_EXINTC4	0x14	0x0000 0000

### 7.2.1 SCFG configuration register1 (SCFG\_CFG1)

Bit	Register	Reset value	Type	Description
Bit 31: 13	Reserved	0x00000	resd	Kept at its default value.
Bit 12	TMR17_DMA_RMP	0x0	rw	TMR17 DMA request remap bit This bit is set and cleared by software. 0x0: No remap (DMA requests for TMR17_CH1 and TMR17_OVERFLOW are mapped on DMA channel 1). 0x1: Remap 1 (DMA request for TMR17_CH1 and TMR17_OVERFLOW is mapped on DMA channel 2).
Bit 11	TMR16_DMA_RMP	0x0	rw	TMR16 DMA request remap bit This bit is set and cleared by software. 0x0: No remap (DMA requests for TMR16_CH1 and TMR16_OVERFLOW are remapped on DMA channel 3). 0x1: Remap 1 (DMA requests for TMR16_CH1 and TMR16_OVERFLOW are remapped on DMA channel 4).
Bit 10	USART1_RX_DMA_RMP	0x0	rw	USART1 RX DMA request remap bit This bit is set and cleared by software. It controls the remapping for USART1 RX DMA channel requests. 0: No remap (DMA request for USART1_RX is mapped on DMA channel 3) 1: Remap (DMA request for USART1_RX is mapped on DMA channel 5).
Bit 9	USART1_TX_DMA_RMP	0x0	rw	USART1 TX DMA request remap bit This bit is set and cleared by software. It controls the remapping for USART1 TX DMA channel requests. 0: No remap (DMA request for USART1_TX is mapped on DMA channel 2). 1: Remap (DMA request for USART1_TX is mapped on DMA channel 4).
Bit 8	ADC_DMA_RMP	0x0	rw	ADC DMA request remap bit This bit is set and cleared by software. It controls the remapping for ADC DMA channel requests. 0: No remap (ADC DMA request is remapped on DMA

				channel 1) 1: Remap (ADC DMA request is mapped on DMA channel 2).
				Infrared modulation envelope signal source selection This field is used to select the infrared modulation envelope signal source.
Bit 7: 6	IR_SRC_SEL	0x0	rw	00: TMR16 01: USART1 10: USART2 11: Reserved
Bit 5	IR_POL	0x0	rw	Infrared output polarity selection 0: Infrared output (IR_OUT) is not inverted 1: Infrared output (IR_OUT) is inverted
Bit 4	PA11_12_RMP	0x0	rw	PA11 and PA12 remap on small packages (20 pins) This bit is set and cleared by software. It controls the remapping of PA9/10 or PA11/12 on small packages. 0: No remap (PA9/PA10 corresponds to PA9/PA10) 1: Remap (PA11/PA12 is mapped on PA9/PA10)
Bit 3: 2	Reserved	0xX	resd	Kept at its default value.
Bit 1: 0	MEM_MAP_SEL	0xX	rw	Boot mode status bit These bits are read-only, indicating from which memory to boot after reset. X0: Boot from main memory 01: Boot from bootloader memory 11: Boot from internal SRAM.

## 7.2.2 SCFG external interrupt configuration register1 (SCFG\_EXINTC1)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 12	EXINT3	0x0000	rw	EXINT3 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT3 external interrupt. 0000: PA[3] 0001: PB[3]
Bit 11: 8	EXINT2	0x0000	rw	EXINT2 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT2 external interrupt. 0000: PA[2] 0001: PB[2]
Bit 7: 4	EXINT1	0x0000	rw	EXINT1 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT1 external interrupt. 0000: PA[1] 0001: PB[1] 0101: PF[1]
Bit 3: 0	EXINT0	0x0000	rw	EXINT0 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT0 external interrupt. 0000: PA[0] 0001: PB[0] 0101: PF[0]

### 7.2.3 SCFG external interrupt configuration register2 (SCFG\_EXINTC2)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 12	EXINT7	0x0000	rw	EXINT7 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT7 external interrupt. 0000: PA[7] 0001: PB[7] 0101: PF[7]
Bit 11: 8	EXINT6	0x0000	rw	EXINT6 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT6 external interrupt. 0000: PA[6] 0001: PB[6] 0101: PF[6]
Bit 7: 4	EXINT5	0x0000	rw	EXINT5 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT5 external interrupt. 0000: PA[5] 0001: PB[5]
Bit 3: 0	EXINT4	0x0000	rw	EXINT4 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT4 external interrupt. 0000: PA[4] 0001: PB[4]

### 7.2.4 SCFG external interrupt configuration register3 (SCFG\_EXINTC3)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 12	EXINT11	0x0000	rw	EXINT11 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT11 external interrupt. 0000: PA[11] 0001: PB[11]
Bit 11: 8	EXINT10	0x0000	rw	EXINT10 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT10 external interrupt. 0000: PA[10] 0001: PB[10]
Bit 7: 4	EXINT9	0x0000	rw	EXINT9 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT9 external

---

				interrupt. 0000: PA[9] 0001: PB[9]
Bit 3: 0	EXINT8	0x0000	rw	EXINT8 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT8 external interrupt. 0000: PA[8] 0001: PB[8]

---

## 7.2.5 SCFG external interrupt configuration register4 (SCFG\_EXINTC4)

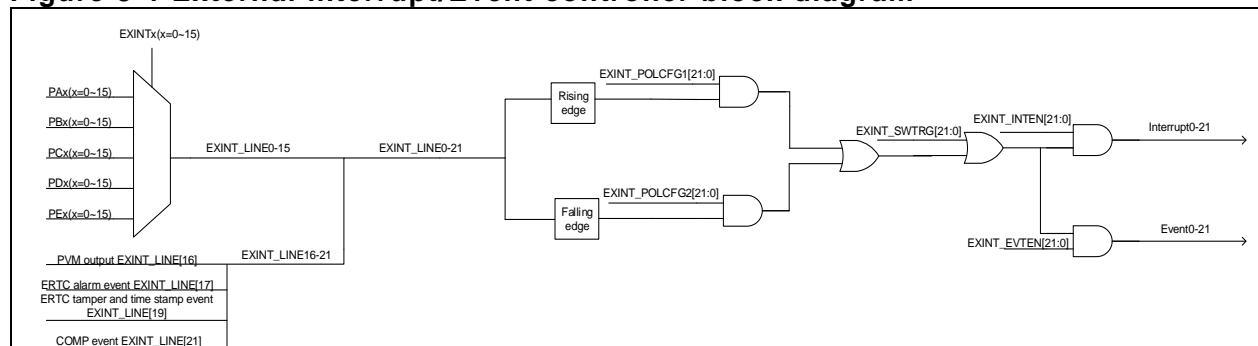
Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 12	EXINT15	0x0000	rw	EXINT15 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT15 external interrupt. 0000: PA[15] 0001: PB[15]
Bit 11: 8	EXINT14	0x0000	rw	EXINT14 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT14 external interrupt. 0000: PA pin14 0001: GPIOB pin14
Bit 7:4	EXINT13	0x0000	rw	EXINT13 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT13 external interrupt. 0000: PA [13] 0001: PB [13]
Bit 3: 0	EXINT12	0x0000	rw	EXINT12 input source configuration These bits can be read/written by software. They are used to select the input source for the EXINT12 external interrupt. 0000: PA [12] 0001: PB [12]

# 8 External interrupt/Event controller (EXINT)

## 8.1 EXINT introduction

EXINT consists of 22 interrupt lines EXINT\_LINE[21:0] (Line 18 and Line 20 are reserved), each of which can generate an interrupt or event by edge detection trigger or software trigger. EXINT can enable or disable an interrupt or event independently through software configuration, and adopt different edge detection modes (rising edge, falling edge or both edges) as well different trigger modes (edge detection, software trigger or both triggers) to respond to the trigger source and generate an interrupt or event.

**Figure 8-1 External interrupt/Event controller block diagram**



### Main features:

- EXINT 0~15 mapping IO can be configured independently
- Independent trigger mode selection on each interrupt line
- Independent enable bit on each interrupt
- Independent enable bit on each event
- Up to 20 software trigger that can be generated and cleared independently
- Independent status bit on each interrupt
- Each interrupt can be cleared independently.

## 8.2 Function overview and configuration procedure

With up to 22 interrupt lines EXINT\_LINE [21:0] (Line 18 and Line 20 are reserved), EXINT can detect not only GPIO external interrupt sources but also internal sources such as PVM output, ERTC alarm, and ERTC tamper and time stamp events. The GPIO interrupt sources can be selected with IOMUX\_EXINTCx register. It should be noted that these input sources are mutually exclusive. For example, EXINT\_LINE0 is allowed to select only one of PA0/PB0/PC0/PD0 pins..., instead of taking both PA0 and PB0 as the input sources at the same time.

EXINT supports multiple edge detection modes, including rising edge, falling edge or both edges, selected by EXINT\_POLCFG1 and EXINT\_POLCFG2 register. Active edge trigger detected on the interrupt line can be used to generate an event or interrupt.

In addition, EXINT supports independent software trigger for the generation of an event or interrupt. This is achieved by setting the corresponding bits in the EXINT\_SWTRG register.

EXINT can enable or disable an interrupt or event individually through software configuration such as EXINT\_INTEN and EXINT\_EVTEN registers, indicating that the corresponding interrupt or event control bit must be enabled in advance.

EXINT also features an independent interrupt status bit. Reading access to EXINT\_INTSTS register can obtain the corresponding interrupt status. The status flag is cleared by writing "1" to this register.

**Interrupt initialization procedure**

- Select an interrupt source by setting IOMUX\_EXINTCx register (This is required if GPIO is used as an interrupt source)
- Select a trigger mode by setting EXINT\_POLCFG1 and EXINT\_POLCFG2 registers
- Enable interrupt or event by setting EXINT\_INTEN and EXINT\_EVTEN registers
- Generate software trigger by setting EXINT\_SWTRG register (This is applied to only software trigger interrupt)

**Interrupt clear procedure**

- Writing “1” to the EXINT\_INTSTS register to clear the interrupts generated, and the corresponding bits in the EXINT\_SWTRG register.

## 8.3 EXINT registers

These peripheral registers must be accessed by words (32 bits).

**Table 8-1** shows EXINT register map and their reset value.

**Table 8-1 External interrupt/Event controller register map and reset value**

Register	Offset	Reset value
EXINT_INTEN	0x00	0x0000 0000
EXINT_EVTEN	0x04	0x0000 0000
EXINT_POLCFG1	0x08	0x0000 0000
EXINT_POLCFG2	0x0C	0x0000 0000
EXINT_SWTRG	0x10	0x0000 0000
EXINT_INTSTS	0x14	0x0000 0000

### 8.3.1 Interrupt enable register (EXINT\_INTEN)

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Forced to 0 by hardware. Interrupt enable or disable on line x 0: Interrupt request is disabled. 1: Interrupt request is enabled. Note: Line 18 and Line 20 are reserved.
Bit 21: 0	INTENx	0x00000	rw	

### 8.3.2 Event enable register (EXINT\_EVTEN)

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Forced to 0 by hardware. Event enable or disable on line x 0: Event request is disabled. 1: Event request is enabled. Note: Line 18 and Line 20 are reserved.
Bit 21: 0	EVTENx	0x00000	rw	

### 8.3.3 Polarity configuration register1 (EXINT\_POLCFG1)

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Forced to 0 by hardware. Rising polarity configuration bit on line x These bits are used to select a rising edge to trigger an interrupt and event on line x. 0: Rising trigger on line x is disabled. 1: Rising trigger on line x is enable. Note: Line 18 and Line 20 are reserved.
Bit 21: 0	RPx	0x00000	rw	

### 8.3.4 Polarity configuration register2 (EXINT\_POLCFG2)

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Forced to be 0 by hardware.
				Falling polarity configuration bit on line x
				These bits are used to select a falling edge to trigger an interrupt and event on line x.
Bit 21: 0	FPx	0x00000	rw	0: Falling trigger on line x is disabled. 1: Falling trigger on line x is enabled. Note: Line 18 and Line 20 are reserved.

### 8.3.5 Software trigger register (EXINT\_SWTRG)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to 0 by hardware.
				Software trigger on line x
				When the corresponding bit in EXINT_INTEN register is 1, if the software writes to this bit, the hardware sets the corresponding bit in the EXINT_INTSTS automatically to generate an interrupt.
Bit 22: 0	SWTx	0x00000	rw	When the corresponding bit in the EXINT_EVTEN register is 1, if the software writes to this bit, the hardware generates an event on the corresponding interrupt line automatically. 0: Default value 1: Software trigger generated Note: This bit is cleared by writing 1 to the corresponding bit in the EXINT_INTSTS register. Note: Line 18 and Line 20 are reserved.

### 8.3.6 Interrupt status register (EXINT\_INTSTS)

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Forced to 0 by hardware.
				Line x status bit
				0: No interrupt occurred. 1: Interrupt occurred. Note: This bit can be cleared by writing "1" to itself. Note: Line 18 and Line 20 are reserved.
Bit 21: 0	LINEx	0x00000	rw	

## 9 DMA controller (DMA)

### 9.1 Introduction

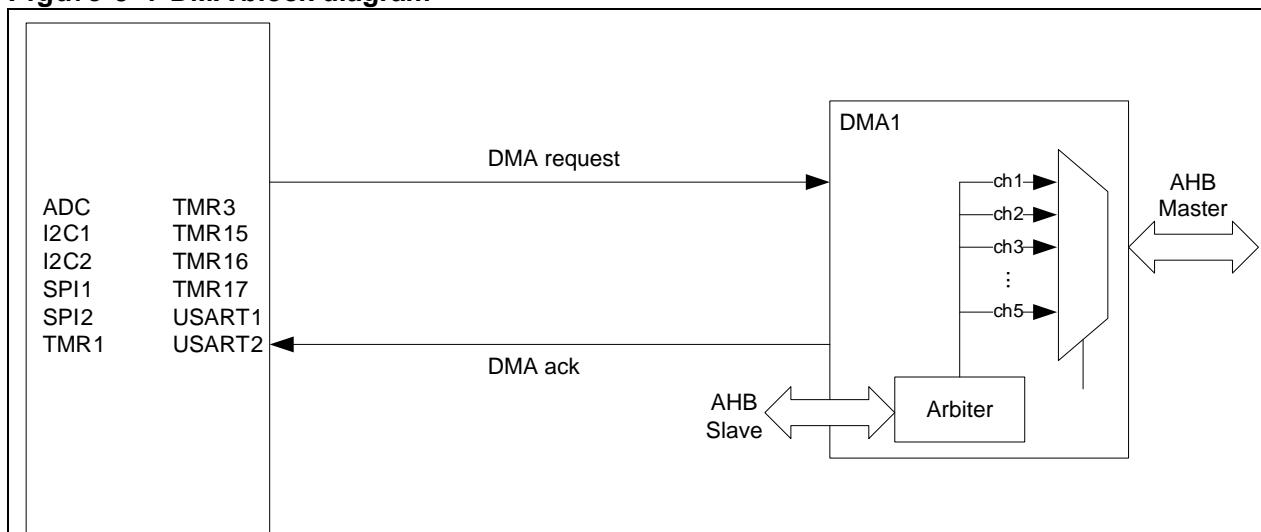
Direct memory access (DMA) controller is designed for 32-bit MCU applications with the aim of enhancing system performance and reducing the generation of interrupts.

The DMA controller contains 5 DMA channels. Each channel manages memory access requests from one or more peripherals. An arbiter is available for coordinating the priority of each DMA request.

### 9.2 Main features

- AMBA compliant (Rev. 2.0)
- Only support AHB OKAY and ERROR responses
- HBUSREQ and HGRANT of AHB master interface are not supported
- Support 5 channels
- Peripheral-to-memory, memory-to-peripheral, and memory-to-memory transfers
- Support hardware handshake
- Support 8-bit, 16-bit and 32-bit data transfers
- Programmable amount of data to be transferred: up to 65535

**Figure 9-1 DMA block diagram**



*Note: The number of DMA peripherals in Figure 9-1 may decrease depending on different models.*

### 9.3 Functional overview

#### 9.3.1 DMA configuration

##### 1. Set the peripheral address in the DMA\_CxPADDR register

The initial peripheral address for data transfer remains unchanged during transmission.

##### 2. Set the memory address in the DMA\_CxMADDR register

The initial memory address for data transfer remains unchanged during transmission.

##### 3. Configure the amount of data to be transferred in the DMA\_CxDTCNT register

Programmable data transfer size is up to 65535. This value is decremented after each data transfer.

##### 4. Configure the channel setting in the DMA\_CxCTRL register

Including channel priority, data transfer direction/width, address incremented mode, circular mode and interrupt mode

##### ● Channel priority (CHPL)

There are four levels, including very high priority, high priority, medium priority and low priority.

If the two channels have the same priority level, then the channel with lower number will get priority over the one with higher number. For example, channel 1 has priority over channel 2.

- **Data transfer direction (DTD)**

Memory-to-peripheral (M2P), peripheral-to-memory (P2M)

- **Address incremented mode (PINCM/MINCM)**

In incremented mode, the subsequent transfer address is the previous address plus transfer width (PWDTH/MWIDTH).

- **Circular mode (LM)**

In circular mode, the contents in the DMA\_CxDTCNT register is automatically reloaded with the initially programmed value after the completion of the last data transfer.

- **Memory-to-memory mode (M2M)**

This mode indicates that DMA channels perform memory-to-memory transfer (works without being triggered by requests from peripherals). Circular mode and memory-to-memory mode cannot be used simultaneously.

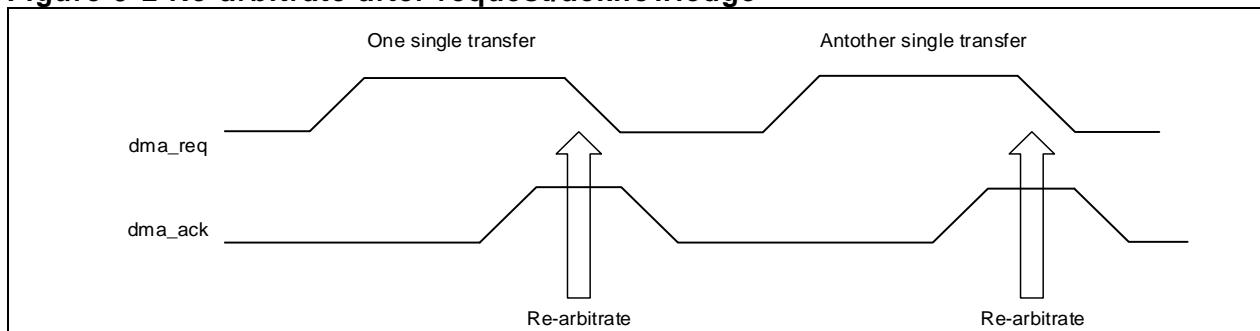
### 9.3.2 Handshake mechanism

In P2M and M2P mode, the peripherals need to send a request signal to the DMA controller. The DMA channel will send a peripheral transfer request (single) until the signal is acknowledged. After the completion of a peripheral transfer, the DMA controller sends an acknowledge signal to the peripheral. The peripheral then releases its request as soon as it receives the acknowledge signal. At the same time, the DMA controller releases the acknowledge signal as well.

### 9.3.3 Arbiter

When several channels are enabled simultaneously, the arbiter will restart arbitration after full data transfer by the master controller. The channel with very high priority waits until the channel of the master controller has completed data transfers before taking control of it. The master controller will re-arbitrate to serve other channels as long as the channel completes a single transfer based on the master controller priority.

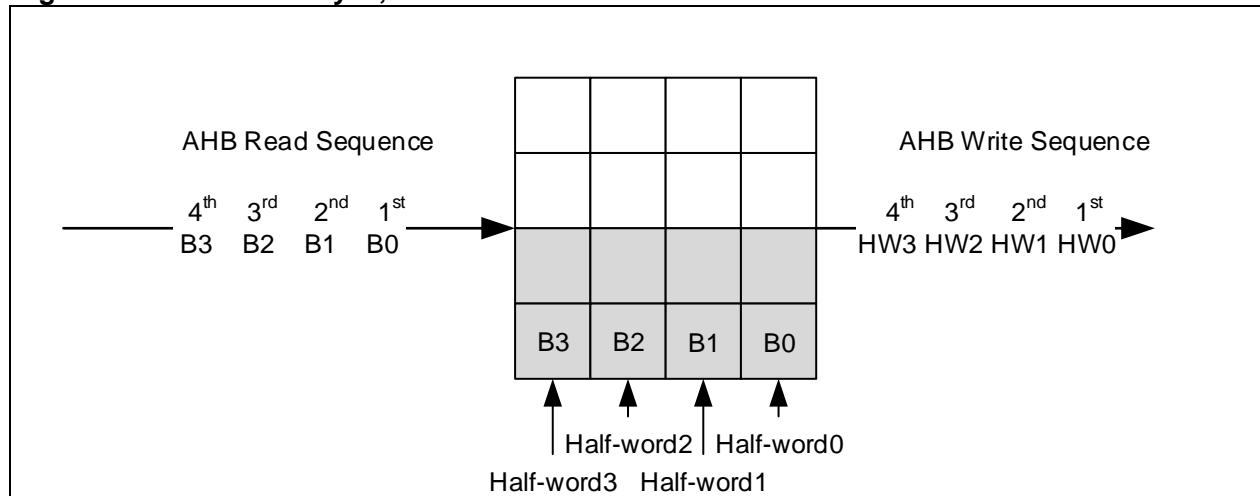
**Figure 9-2 Re-arbitrate after request/acknowledge**



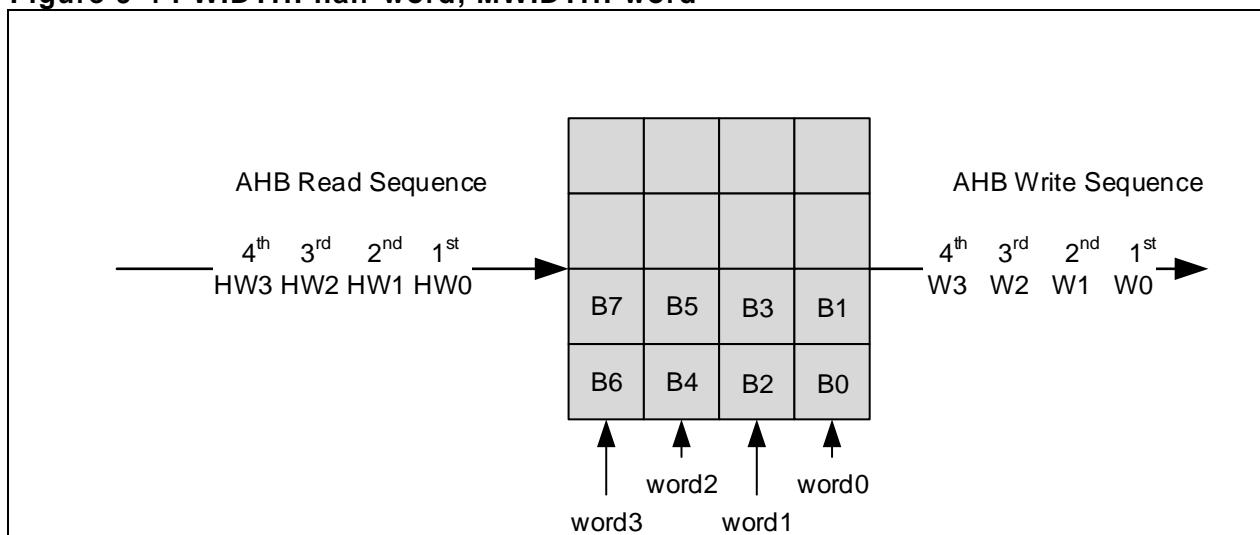
### 9.3.4 Programmable data transfer width

Transfer width of the source data and destination data is programmable through the PWIDHT and MWIDHT bits in the DMA\_CxCTRL register. When PWIDHT is not equal to MWIDHT, number of data to be transferred can be aligned according to the settings of PWIDHT/MWIDHT.

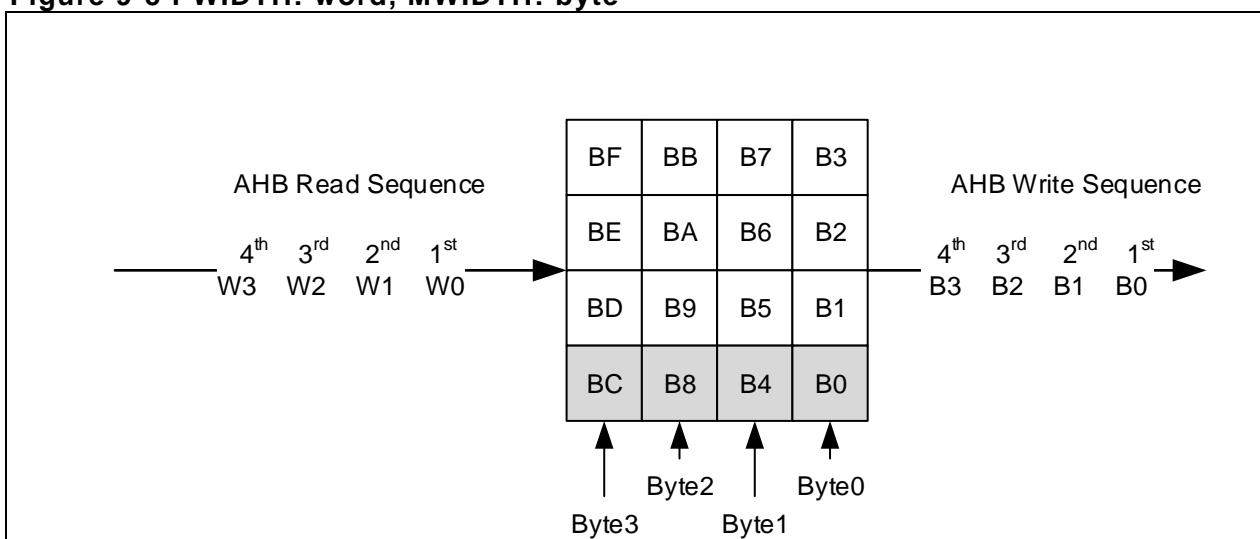
**Figure 9-3 PWIDHT: byte, MWIDHT: half-word**



**Figure 9-4 PWIDHT: half-word, MWIDHT: word**



**Figure 9-5 PWIDHT: word, MWIDHT: byte**



### 9.3.5 Errors

**Table 9-1 DMA error event**

<b>Error event</b>	
Transfer error	AHB response error occurred during DMA read/write access

### 9.3.6 Interrupts

An interrupt can be generated on DMA half-transfer, transfer complete and transfer error. Each channel has its specific interrupt flag, clear and enable bits, as shown in the table below.

**Table 9-2 DMA interrupt requests**

<b>Interrupt event</b>	<b>Event flag bit</b>	<b>Clear control bit</b>	<b>Enable control bit</b>
Half transfer	HDTF	HDTFC	HDTIEN
Transfer completed	FDTF	FDTFC	FDTIEN
Transfer error	DTERRF	DTERRFC	DTERRIEN

*Note: DMA2 channel4/5, channel6/7 interrupts are mapped onto the same interrupt vector.*

### 9.3.7 Fixed DMA request mapping

Several peripheral requests are mapped to the same DMA channel through logic “OR”. This means that only one request can be enabled on a channel at a time.

The peripheral DMA requests can be independently activated/de-activated by setting the control bits in the corresponding peripheral registers.

**Table 9-3 DMA requests for each channel**

<b>Peripherals</b>	<b>Channel 1</b>	<b>Channel 2</b>	<b>Channel 3</b>	<b>Channel 4</b>	<b>Channel 5</b>
ADC	ADC <sup>(1)</sup>	ADC <sup>(2)</sup>			
SPI/I2S		SPI1/I2S1_RX	SPI1/I2S1_TX	SPI2/I2S2_RX	SPI2/I2S2_TX
USART1		USART1_TX <sup>(1)</sup>	USART1_RX <sup>(1)</sup>	USART1_TX <sup>(2)</sup>	USART1_RX <sup>(2)</sup>
USART2				USART2_TX	USART2_RX
I2C		I2C1_TX	I2C1_RX	I2C2_TX	I2C2_RX
TMR1		TMR1_CH1	TMR1_CH2	TMR1_CH4 TMR1_TRIG TMR1_HALL	TMR1_CH3 TMR1_OVERFLOW
MR3		TMR3_CH3	TMR3_CH4 TMR3_OVERFLOW	TMR3_CH1 TMR3_TRIG	
TMR6			TMR6_OVERFLOW		
TMR15					TMR15_CH1 TMR15_OVERFLOW TMR15_TRIG TMR15_HALL TMR15_CH2
TMR16			TMR16_CH1 <sup>(1)</sup> TMR16_OVERFLOW <sup>(1)</sup>	TMR16_CH1 <sup>(2)</sup> TMR16_OVERFLOW <sup>(2)</sup>	
TMR17	TMR17_CH1 <sup>(1)</sup> TMR17_OVERFLOW <sup>(1)</sup>	TMR17_CH1 <sup>(2)</sup> TMR17_OVERFLOW <sup>(2)</sup>			

- When the corresponding remap bit of the SCFG\_CFG1 is 0, the corresponding DMA request is mapped onto the channel.
- When the corresponding remap bit of the SCFG\_CFG1 is 1, the corresponding DMA request is mapped onto the channel.

## 9.4 DMA registers

**Table 9-4** shows DMA register map and their reset values.

These peripheral registers must be accessed by bytes (8 bits), half-words (16 bits) or words (32 bits).

**Table 9-4 DMA register map and reset value**

Register	Offset	Reset value
DMA_STS	0x00	0x0000 0000
DMA_CLR	0x04	0x0000 0000
DMA_C1CTRL	0x08	0x0000 0000
DMA_C1DTCNT	0x0C	0x0000 0000
DMA_C1PADDR	0x10	0x0000 0000
DMA_C1MADDR	0x14	0x0000 0000
DMA_C2CTRL	0x1C	0x0000 0000
DMA_C2DTCNT	0x20	0x0000 0000
DMA_C2PADDR	0x24	0x0000 0000
DMA_C2MADDR	0x28	0x0000 0000
DMA_C3CTRL	0x30	0x0000 0000
DMA_C3DTCNT	0x34	0x0000 0000
DMA_C3PADDR	0x38	0x0000 0000
DMA_C3MADDR	0x3C	0x0000 0000
DMA_C4CTRL	0x44	0x0000 0000
DMA_C4DTCNT	0x48	0x0000 0000
DMA_C4PADDR	0x4C	0x0000 0000
DMA_C4MADDR	0x50	0x0000 0000
DMA_C5CTRL	0x58	0x0000 0000
DMA_C5DTCNT	0x5C	0x0000 0000
DMA_C5PADDR	0x60	0x0000 0000
DMA_C5MADDR	0x64	0x0000 0000

### 9.4.1 DMA interrupt status register (DMA\_STS)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 20	Reserved	0x0	resd	Kept at its default value.
Bit 19	DTERRF5	0x0	ro	Channel 5 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 18	HDTF5	0x0	ro	Channel 5 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 17	FDTF5	0x0	ro	Channel 5 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 16	GF5	0x0	ro	Channel 5 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 15	DTERRF4	0x0	ro	Channel 4 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 14	HDTF4	0x0	ro	Channel 4 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 13	FDTF4	0x0	ro	Channel 4 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 12	GF4	0x0	ro	Channel 4 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 11	DTERRF3	0x0	ro	Channel 3 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 10	HDTF3	0x0	ro	Channel 3 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 9	FDTF3	0x0	ro	Channel 3 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 8	GF3	0x0	ro	Channel 3 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 7	DTERRF2	0x0	ro	Channel 2 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 6	HDTF2	0x0	ro	Channel 2 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 5	FDTF2	0x0	ro	Channel 2 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 4	GF2	0x0	ro	Channel 2 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event

Bit 3	DTERRF1	0x0	ro	Channel 1 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 2	HDTF1	0x0	ro	Channel 1 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 1	FDTF1	0x0	ro	Channel 1 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 0	GF1	0x0	ro	Channel 1 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event

## 9.4.2 DMA interrupt flag clear register (DMA\_CLR)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 20	Reserved	0x0	resd	Kept at its default value.
Bit 19	DTERRFC5	0x0	rw1c	Channel 5 data transfer error flag clear 0: No effect 1: Clear the DTERRF5 flag in the DMA_STS register
Bit 18	HDTFC5	0x0	rw1c	Channel 5 half transfer flag clear 0: No effect 1: Clear the HDTF5 flag in the DMA_STS register
Bit 17	FDTFC5	0x0	rw1c	Channel 5 transfer complete flag clear 0: No effect 1: Clear the FDTF5 flag in the DMA_STS register
Bit 16	GFC5	0x0	rw1c	Channel 5 global interrupt flag clear 0: No effect 1: Clear the DTERRF5, HDTF5, FDTF5 and GF5 in the DMA_STS register
Bit 15	DTERRFC4	0x0	rw1c	Channel 4 data transfer error flag clear 0: No effect 1: Clear the DTERRF4 flag in the DMA_STS register
Bit 14	HDTFC4	0x0	rw1c	Channel 4 half transfer flag clear 0: No effect 1: Clear the HDTF4 flag in the DMA_STS register
Bit 13	FDTFC4	0x0	rw1c	Channel 4 transfer complete flag clear 0: No effect 1: Clear the FDTF4 flag in the DMA_STS register
Bit 12	GFC4	0x0	rw1c	Channel 4 global interrupt flag clear 0: No effect 1: Clear the DTERRF4, HDTF4, FDTF4 and GF4 flag in the DMA_STS register
Bit 11	DTERRFC3	0x0	rw1c	Channel 7 data transfer error flag clear 0: No effect 1: Clear the DTERRF7 flag in the DMA_STS register
Bit 10	HDTFC3	0x0	rw1c	Channel 7 half transfer flag clear 0: No effect 1: Clear the HDTF7 flag in the DMA_STS register
Bit 9	FDTFC3	0x0	rw1c	Channel 3 transfer complete flag clear 0: No effect 1: Clear the FDTF3 flag in the DMA_STS register

Bit 8	GFC3	0x0	rw1c	Channel 3 global interrupt flag clear 0: No effect 1: Clear the DTERRF3, HDTF3, FDTF3 and GF3 flag in the DMA_STS register
Bit 7	DTERRFC2	0x0	rw1c	Channel 2 data transfer error flag clear 0: No effect 1: Clear the DTERRF2 flag in the DMA_STS register
Bit 6	HDTFC2	0x0	rw1c	Channel 2 half transfer flag clear 0: No effect 1: Clear the HDTF2 flag in the DMA_STS register
Bit 5	FDTFC2	0x0	rw1c	Channel 2 transfer complete flag clear 0: No effect 1: Clear the FDTF2 flag in the DMA_STS register
Bit 4	GFC2	0x0	rw1c	Channel 2 global interrupt flag clear 0: No effect 1: Clear the DTERRF2, HDTF2, FDTF2 and GF2 in the DMA_STS register
Bit 3	DTERRFC1	0x0	rw1c	Channel 1 data transfer error flag clear 0: No effect 1: Clear the DTERRF1 flag in the DMA_STS register
Bit 2	HDTFC1	0x0	rw1c	Channel 1 half transfer flag clear 0: No effect 1: Clear the HDTF1 flag in the DMA_STS register
Bit 1	FDTFC1	0x0	rw1c	Channel 1 transfer complete flag clear 0: No effect 1: Clear the FDTF1 flag in the DMA_STS register
Bit 0	GFC1	0x0	rw1c	Channel 1 global interrupt flag clear 0: No effect 1: Clear the DTERRF1, HDTF1, FDTF1 and GF1 in the DMA_STS register

### 9.4.3 DMA channel-x configuration register (DMA\_CxCTRL) (x = 1...5)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 15	Reserved	0x00000	resd	Kept at its default value.
Bit 14	M2M	0x0	rw	Memory to memory mode 0: Disabled 1: Enabled.
Bit 13: 12	CHPL	0x0	rw	Channel priority level 00: Low 01: Medium 10: High 11: Very high
Bit 11: 10	MWIDTH	0x0	rw	Memory data bit width 00: 8 bits 01: 16 bits 10: 32 bits 11: Reserved
Bit 9: 8	PWIDTH	0x0	rw	Peripheral data bit width 00: 8 bits 01: 16 bits 10: 32 bits 1: Reserved
Bit 7	MINCM	0x0	rw	Memory address increment mode 0: Disabled 1: Enabled.
Bit 6	PINCM	0x0	rw	Peripheral address increment mode 0: Disabled

				1: Enabled. Circular mode 0: Disabled
Bit 5	LM	0x0	rw	1: Enabled.
Bit 4	DTD	0x0	rw	Data transfer direction 0: Read from peripherals 1: Read from memory
Bit 3	DTERRIEN	0x0	rw	Data transfer error interrupt enable 0: Disabled 1: Enabled.
Bit 2	HDTIEN	0x0	rw	Half-transfer interrupt enable 0: Disabled 1: Enabled.
Bit 1	FDTIEN	0x0	rw	Transfer complete interrupt enable 0: Disabled 1: Enabled.
Bit 0	CHEN	0x0	rw	Channel enable 0: Disabled 1: Enabled.

#### 9.4.4 DMA channel-x number of data register (DMA\_CxDTCNT) (x = 1...5)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	CNT	0x0000	rw	Number of data to transfer The number of data to transfer is from 0x0 to 0xFFFF. This register can only be written when the CHEN bit in the corresponding channel is set 0. The value is decremented by 1 after each DMA transfer. Note: This register holds the number of data to transfer, instead of transfer size. The transfer size is calculated by data width.

#### 9.4.5 DMA channel-x peripheral address register (DMA\_CxPADDR) (x = 1...5)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	PADDR	0x0000 0000	rw	Peripheral base address Base address of peripheral data register is the source or destination of data transfer. Note: The register can only be written when the CHEN bit in the corresponding channel is set 0.

#### 9.4.6 DMA channel-x memory address register (DMA\_CxMADDR) (x = 1...5)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	MADDR	0x0000 0000	rw	Memory base address Memory address is the source or destination of data transfer. Note: The register can only be written when the CHEN bit in the corresponding channel is set 0.

# 10 CRC calculation unit (CRC)

## 10.1 CRC introduction

The Cyclic Redundancy Check (CRC) is an independent peripheral with CRC check feature. It follows CRC32 standard.

The CRC\_CTRL register is used to select input data toggle (word, REVOD=1) or output data toggle (byte, REVID=01; half-word, REVID=10; word: REVID=11). CRC calculation unit is also equipped with initialization function. After each reset, the value in the CRC\_IDT register is written into the data register (CRC\_DT) by CRC.

Users can write the data to go through CRC check and read the calculated result through CRC\_DT register. Note that the calculation result is the combination of the previous result and the current value to be calculated.

### Main features

- Use CRC-32 code
- 4 HCLK cycles for each CRC calculation
- Support input/output data format toggle
- Perform write/read operation through CRC\_DT register
- Set an initialization value with the CRC\_IDT register. The value is loaded into CRC\_DT register after each CRC reset.

## 10.2 CRC registers

CRC\_DT register can be accessed by bytes (8 bits), half-words (16 bits) or words (32 bits). Other registers have to be accessed by words (32 bits).

**Table 10-1 CRC register map and reset value**

Register	Offset	Reset value
CRC_DT	0x00	0xFFFF FFFF
CRC_CDT	0x04	0x0000 0000
CRC_CTRL	0x08	0x0000 0000
CRC_IDT	0x10	0xFFFF FFFF

### 10.2.1 Data register (CRC\_DT)

Bit	Register	Reset value	Type	Description
Bit 31: 0	DT	0xFFFF FFFF	rw	Data value This field stands for data register bits. Used as input register when writing new data into the CRC calculator. It returns CRC calculation results when it is read.

### 10.2.2 Common data register (CRC\_CDT)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7: 0	CDT	0x00	rw	Common 8-bit data value This field is used to store 1-byte data temporarily. This register is not affected by the CRC reset generated by the RST bit in the CRC_CTRL register.

### 10.2.3 Control register (CRC\_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	REVOD	0x0	resd	Reverse output data Set and cleared by software. This bit is used to control whether or not to reverse output data. 0: No effect 1: Word reverse
Bit 6: 5	REVID	0x0	rw	Reverse input data Set and cleared by software. This bit is used to control how to reverse input data. 00: No effect 01: Byte reverse 10: Half-word reverse 11: Word reverse
Bit 4: 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	RST	0x0	rw	Reset CRC calculation unit Set by software. Cleared by hardware. To reset CRC calculation unit, the data register is set to 0xFFFF FFFF. 0: No effect 1: Reset

### 10.2.4 Initialization register (CRC\_IDT)

Bit	Register	Reset value	Type	Description
Bit 31: 0	IDT	0xFFFF FFFF	rw	Initialization data register When CRC reset is triggered by the RST bit in the CRC_CTRL register, the value in the initialization register is written into the CRC_DT register as an initial value.

# 11 I<sup>2</sup>C interface

## 11.1 I<sup>2</sup>C introduction

I<sup>2</sup>C (inter-integrated circuit) bus interface manages the communication between the microcontroller and serial I<sup>2</sup>C bus. It supports master and slave modes, with up to 400 Kbit/s of communication speed.

## 11.2 I<sup>2</sup>C main features

- I<sup>2</sup>C bus
  - Master and slave modes
  - Multimaster capability
  - Stand speed (100 kHz), fast speed (400 kHz)
  - 7-bit and 10-bit address modes
  - Broadcast call mode
  - Status flag
  - Error flag
  - Clock stretching capability
  - Communication event interrupts
  - Error interrupts
- Support DMA transfer
- Support SMBus2.protocol
  - PEC generation and verification
  - SMBus reminder capability
  - ARP(address resolution protocol)
  - Timeout detection
- PMBus

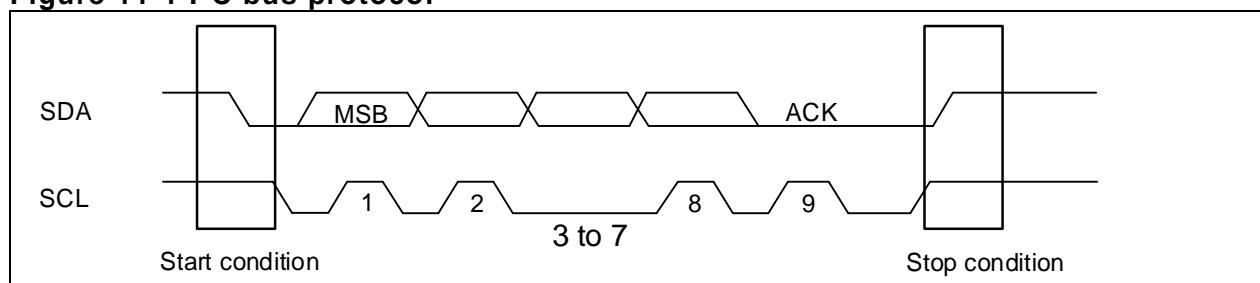
## 11.3 I<sup>2</sup>C functional overview

I<sup>2</sup>C bus consists of a data line (SDA) and clock line (SCL). It can achieve a maximum of 100 kHz speed in standard mode, whereas up to 400kHz in fast mode. A frame of data transfer begins with a Start condition and ends with a Stop condition. The bus is kept in busy state after receiving a Start condition, and becomes idle as long as it receives a Stop condition.

Start condition: SDA switches from high to low when SCL is set high.

Stop condition: SDA switches from low to high when SCL is set high.

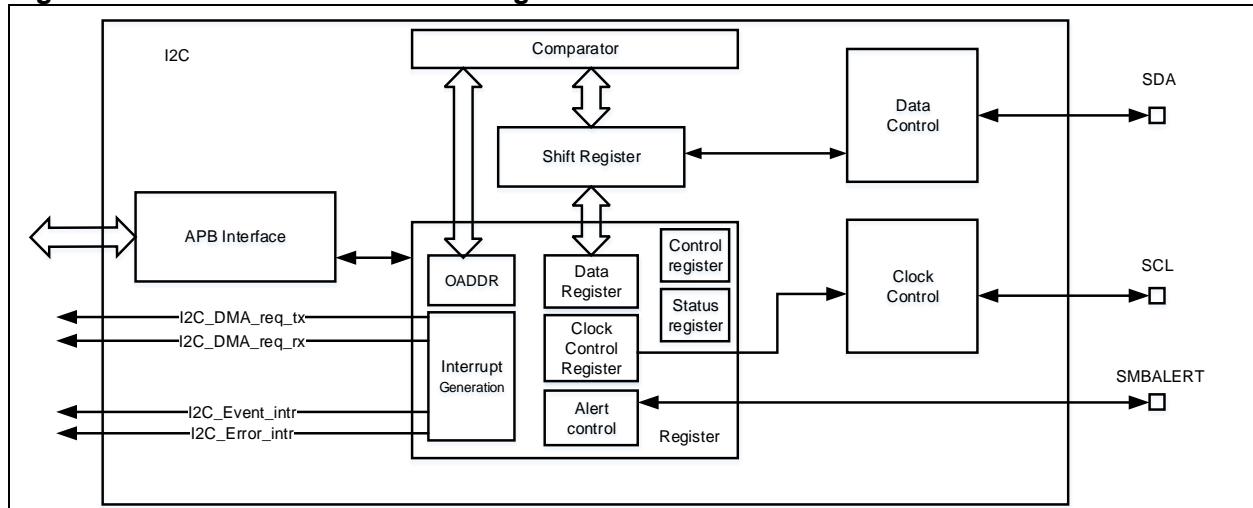
**Figure 11-1 I<sup>2</sup>C bus protocol**



## 11.4 I<sup>2</sup>C interface

[Figure 11-2](#) shows the block diagram of I<sup>2</sup>C function

**Figure 11-2 I<sup>2</sup>C function block diagram**



### 1. I<sup>2</sup>C clock

I<sup>2</sup>C is clocked by either APB1 or APB2. The I<sup>2</sup>C clock division is done by setting the CLKFREQ[7: 0] in the I2C\_CTRL2 register. The minimum clock frequency varies from one mode to another, that is, at least 2 MHz in standard mode, but 4 MHz in fast mode.

### 2. Operating mode

I<sup>2</sup>C bus interface can operate both in master mode and slave mode. Switching from master mode to slave mode, vice versa, is supported as well. By default, the interface operates in slave mode. When GENSTART=1 is set (Start condition is activated), the I<sup>2</sup>C bus interface switches from slave mode to master mode, and returns to slave mode automatically at the end of data transfer (Stop condition is triggered).

- Master transmitter
- Master receiver
- Slave transmitter
- Slave receiver

### 3. Communication process

- Master mode communication:
  1. Start condition generation
  2. Address transmission
  3. Data Tx or Rx
  4. Stop condition generation
  5. End of communication
- Slave mode communication:
  1. Wait until the address is matched.
  2. Data Tx or Rx
  3. Wait for the generation of Stop condition
  4. End of communication

### 4. Address control

Both master and slave support 7-bit and 10-bit addressing modes.

#### Slave address mode:

- In 7-bit mode
  - ADDR2EN=0 stands for a single address mode: only matches OADDR1
  - DUALEN=1 stands for dual address mode: matches OADDR1 and OADDR2

- In 10-bit mode
  - Only matches OADDR1

**Support special slave address:**

- Broadcast call address (0b0000000x): This address is enabled when GCAEN=1.
- SMBus device default address (0b1100001x): This address is enabled for SMBus address resolution protocol in SMBus device mode.
- SMBus master default address (0b0001000x): This address is enabled for SMBus master notification protocol in SMBus master mode.
- SMBus alert address (0b0001100x): This address is enabled for SMBus alert response address protocol in SMBus master mode when SMBALERT = 1

Refer to SMBus2.0 protocol for more information.

**Slave address matching procedure:**

- Receive a Start condition
- Address matching
- The slave sends an ACK if address is matched.
- ADDR7F is set, with DIRF indicating the transmission direction
  - When DIRF =0, slave enters receiver mode, starting receiving data.
  - When DIRF =1, slave enters transmitter mode, starting transmitting data

## 5. Clock stretching capability

Clock stretching is enabled by setting the STRETCH bit in the I2C\_CTRL1 register. Once enabled, when the slave cannot process data in a timely manner on certain conditions, it will pull down SCL line to low level to stop communication in order to prevent data loss.

- Transmitter mode:
  - Clock stretching enable: If no data is written to the I2C\_DT register before the next byte transmission (the first SCL rising edge of the next data), the I<sup>2</sup>C interface will pull down SCL bus and wait until the data is written to the I2C\_DT
  - Clock stretching disable: if no data is written to the I2C\_DT register before the next byte transmission (the first SCL rising edge of next data), an underrun error will happen.
- Receiver mode
  - Clock stretching enable: When the shift register has received another byte before the data in the I2C\_DT register is read, the I<sup>2</sup>C will hold the SCL bus low to wait for the software to read I2C\_DT register
  - Clock stretching disable: The data in the I2C\_DT register is not yet read when the shift register receives another byte. In this case, if another data is received, an overrun error occurs.

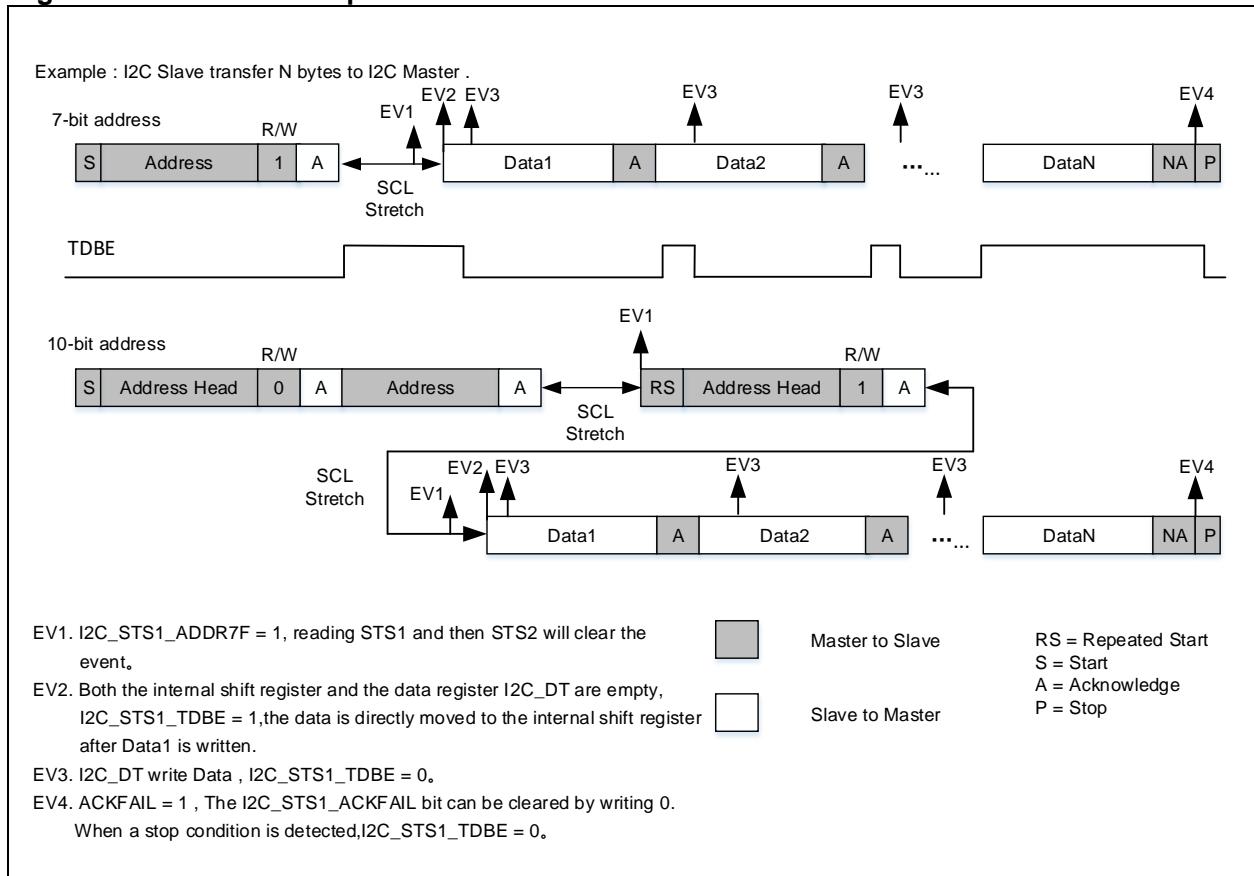
### 11.4.1 I<sup>2</sup>C slave communication flow

**Initialization**

Enable I<sup>2</sup>C peripheral clock, and configure the clock-related bits in the I2C\_CTRL2 register to get correct timings, and then wait for I<sup>2</sup>C master to send a Start condition.

**Transmitter**

Figure 11-3 shows the transfer sequence of slave transmitter.

**Figure 11-3 Transfer sequence of slave transmitter****7-bit address mode:**

1. Wait for the master to send an address
2. EV1: Address is matched (ADDR7F=1), and then the slave pulls the SCL bus low. Reading STS1 and then STS2 by software clears the ADDR7F bit. It then enters transmission stage, and both DT register and internal shift register are now empty. The TDBE bit is set 1 by hardware.
3. EV2: When the data is written to the DT register, it is directly moved to the shift register and the SCL bus is released. The TDBE bit is still set 1 at this time.
4. EV3: The DT register remains empty, but the shift register is not. Writing to the DT register clears the TDBE bit.
5. EV4: After receiving the ACKFAIL event from the master, ACKFAIL=1 is activated. Writing 0 to the ACKFAIL bit clears the event.
6. End of communication.

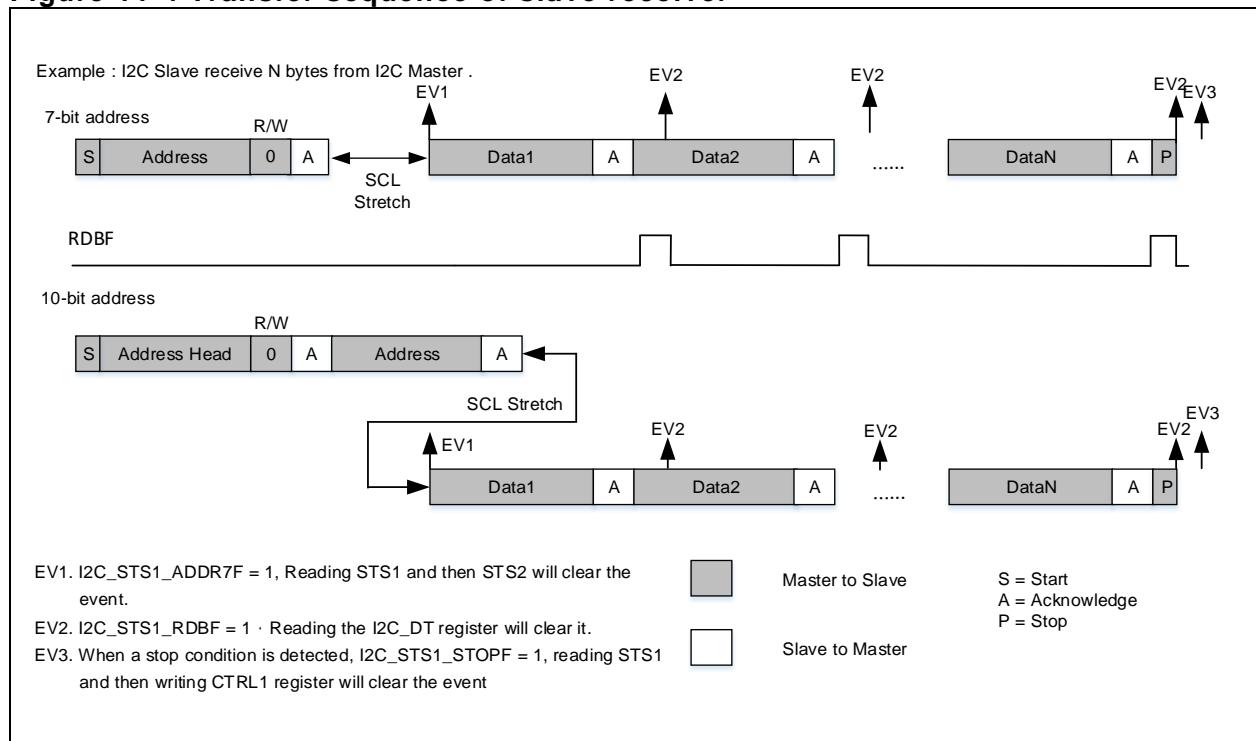
**10-bit address mode:**

1. Wait for the master to send an address
2. EV1: Address is matched (ADDR7F=1), and then the slave pulls the SCL bus low. Reading STS1 and then STS2 by software clears the ADDR7F bit. Wait for the master to re-send Start condition.
3. EV1: Address is matched (ADDR7F=1). Reading STS1 and then STS2 clears the ADDR7F bit once more. It then enters transmission stage. Both DT register and shift register are empty. The TDBE is set 1 by hardware.
4. EV2: When the data is written to DT register, it is directly moved to the shift register, and SCL bus is released. The TDBE is still set 1 at this time.
5. EV3: The DT register remains empty but the shift register is not. Writing to the DT register clears the TDBE bit.
6. EV4: After receiving the ACKFAIL event from the master, ACKFAIL=1 is activated. Writing 0 to the ACKFAIL bit clears the event.
7. End of communication.

**Slave receiver**

Figure 11-4 shows the transfer sequence of slave receiver.

**Figure 11-4 Transfer sequence of slave receiver**

**7-bit address mode:**

1. Wait for the master to send an address.
2. EV1: Address is matched (ADDR7F=1), and the slave pulls the SCL bus low. Reading STS1 and then STS2 by software clears the ADDR7F bit. At this point, the SCL bus is released, and enters receive stage.
3. The internal shift register receives the bus data and stores them to DT register.
4. EV2: After receiving the bytes, the RDBF bit is set to1. Reading the I2C\_DT register clears the RDBF bit.
5. EV3: After receiving the Stop condition from the master, STOPF=1 is activated. Reading STS1 and then writing to CTRL1 register clears the event.
6. End of communication.

**10-bit address mode:**

1. Wait for the master to send an address.
2. EV1: Address is matched (ADDR7F=1). The slave pulls the SCL bus low. Reading STS1 and then STS2 by software clears the ADDR7F bit. At this point, the SCL bus is released, and enters receive stage.
3. The internal shift register receives the bus data and stores them to DT register.
4. EV2: After receiving the byte, the RDBF bit is set to 1. Reading the I2C\_DT register clears the RDBF bit.
5. EV3: After receiving the Stop condition from the master, STOPF=1 is activated. Reading STS1 and then writing to CTRL1 register clears the event.
6. End of communication.

## 11.4.2 I<sup>2</sup>C master communication flow

### Master mode Initialization

1. Program input clocks to generate correct timings through the CLKFREQ bit in the I2C\_CTRL2 register;
2. Program I<sup>2</sup>C communication speed through the I2C\_CLKCTRL bit in the clock control register;
3. Program the maximum rising time of bus through the I2C\_TMRISE register;
4. Program the control register1 I2C\_CTRL1;
5. Enable peripherals, if the GENSTART bit is set, a Start condition is generated on the bus, and the device enters master mode.

### Slave address transmission

Slave address is divided into 7-bit and 10-bit modes. Whether it is transmitter mode or receiver mode depends on the lowest address bit.

- 7-bit address mode:

Transmitter: When the lowest bit of the address sent is 0, the master enters transmitter mode.

Receiver: When the lowest bit of the address sent is 1, the master enters receiver mode.

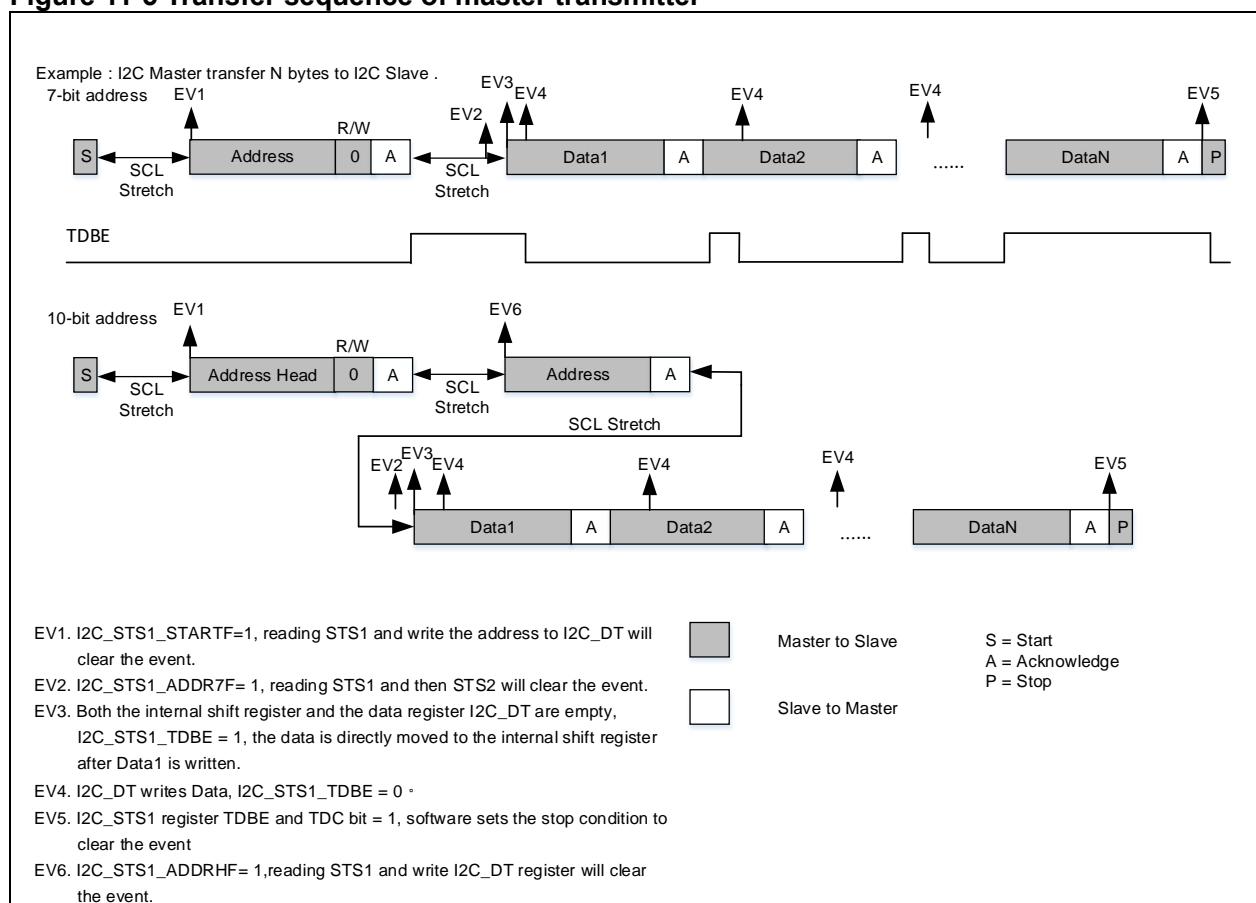
- 10-bit address mode:

Transmitter: First send address head 0b11110xx0 (where xx refers to address [9: 8]), and then slave address [7: 0], the master enters transmitter mode.

Receiver: First send slave address head 0b11110xx0 (where xx refers to address [9:8]) and then address [7: 0], followed by the address head 0b11110xx1 (where xx refers to address [9: 8]), the master enters receiver mode.

### Master transmitter

**Figure 11-5 Transfer sequence of master transmitter**



- **7-bit address mode:**

1. Generate a Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 by software and write the address to DT

register.

3. EV2: Address is matched successfully (ADDR7F=1). Reading STS1 and then STS2 clears the ADDR7F bit. At this point, the master enters transmit stage, and both DT register and internal shift register are empty. The TDBE bit is set 1 by hardware.
4. EV3: When the data is written to the DT register, it is directly moved to the shift register and the SCL bus is released. The TDBE bit is still set 1 at this time.
5. EV4: The DT register remains empty but the shift register is not. Writing to the DT register clears the TDBE bit.
6. The TDBE bit is set only after the second-to-last byte is sent.
7. EV5: TDC=1 indicates that the byte transmission is complete. The master sends a Stop condition (STOPF=1). The TDBE bit and TDC bit are cleared automatically by hardware.
8. End of communication.

- **10-bit address mode:**

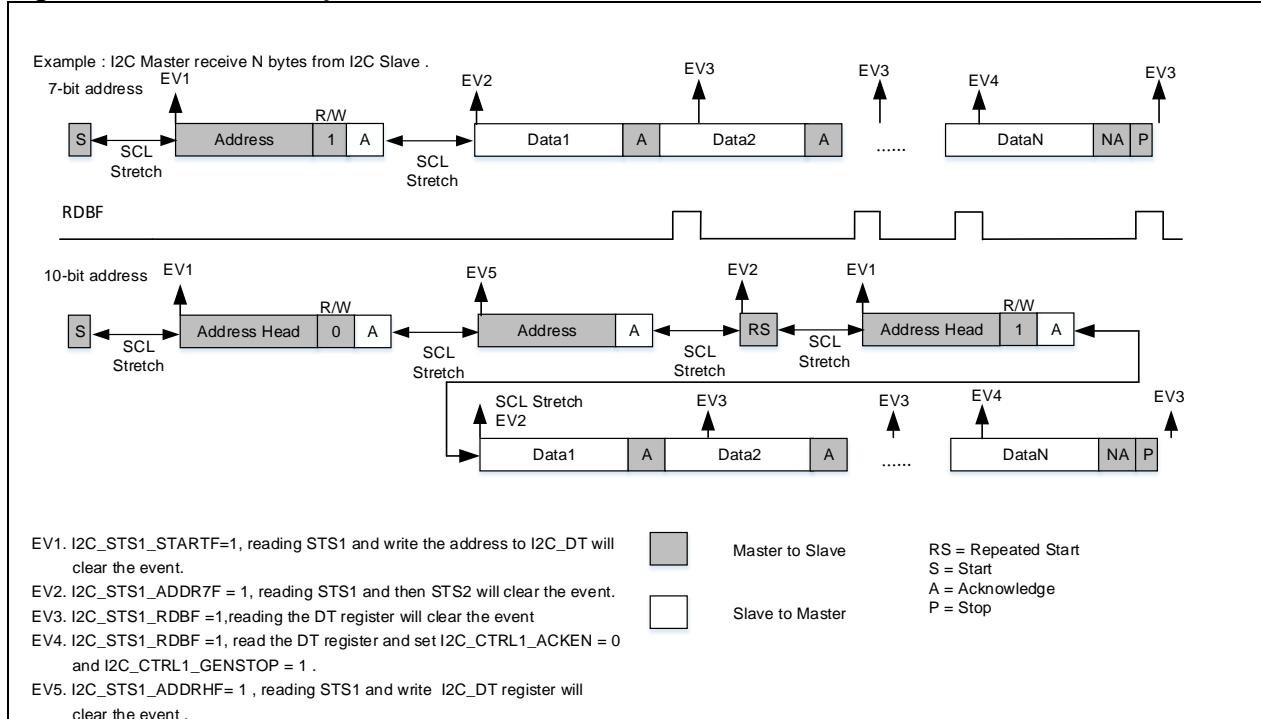
1. Generate Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
3. EV6: 10-bit address head sequence is sent. Reading STS1 and writing to DT register clears the ADDRHF bit.
4. EV2: Address is matched successfully (ADDR7F=1). Reading STS1 and then STS2 clears the ADDR7F bit. In this case, the master enters transmit stage, and both DT register and internal shift register are empty. The TDBE bit is set 1 by hardware.
5. EV3: When the data is written to the DT register, it is directly moved to the shift register and the SCL bus is released. The TDBE bit is still set 1 at this time.
6. EV4: At this point, the DT register remains empty but the shift register is not. Writing to the DT register clears the TDBE bit.
7. The TDBE bit is set only after the second-to-last byte is sent.
8. EV5: TDC=1 indicates that the byte transmission is complete. The master sends Stop condition (STOPF=1). The TDBE bit and TDC bit is cleared by hardware.
9. End of communication.

### Master receiver

Data reception depends on I<sup>2</sup>C interrupt priority:

1. **I<sup>2</sup>C interrupts with very high priority**
- After the second-to-last byte is read, clear the ACKEN bit and set the GENSTOP bit in the I2C\_CTRL1 register to generate Stop condition.
  - If only one byte is to be received, clear the ADDR7F flag and set the ACKEN and GENSTOP bit in the I2C\_CTRL1 register.
  - After a byte is received, the I2C\_STS1\_RDBF bit is set 1 by hardware, and it is cleared after the software read the I2C\_DT register.

Figure 11-6 Transfer sequence of master receiver



### ● 7-bit address mode:

1. Generate a Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
3. EV2: Address is matched successfully (ADDR7F=1). Reading STS1 and then STS2 clears the ADDR7F bit. In this case, the master enters receive stage.
4. EV3: The RDBF bit is set 1 after a byte is received. Reading the I2C\_DT register clears the RDBF.
5. EV4: Once the second-to-last byte is received, the ACKEN bit must be cleared and the GENSTOP must be set by software.
6. EV3: The RDBF bit is set 1 after receiving a byte. Reading the I2C\_DT register clears the RDBF.
7. End of communication.

### ● 10-bit address mode:

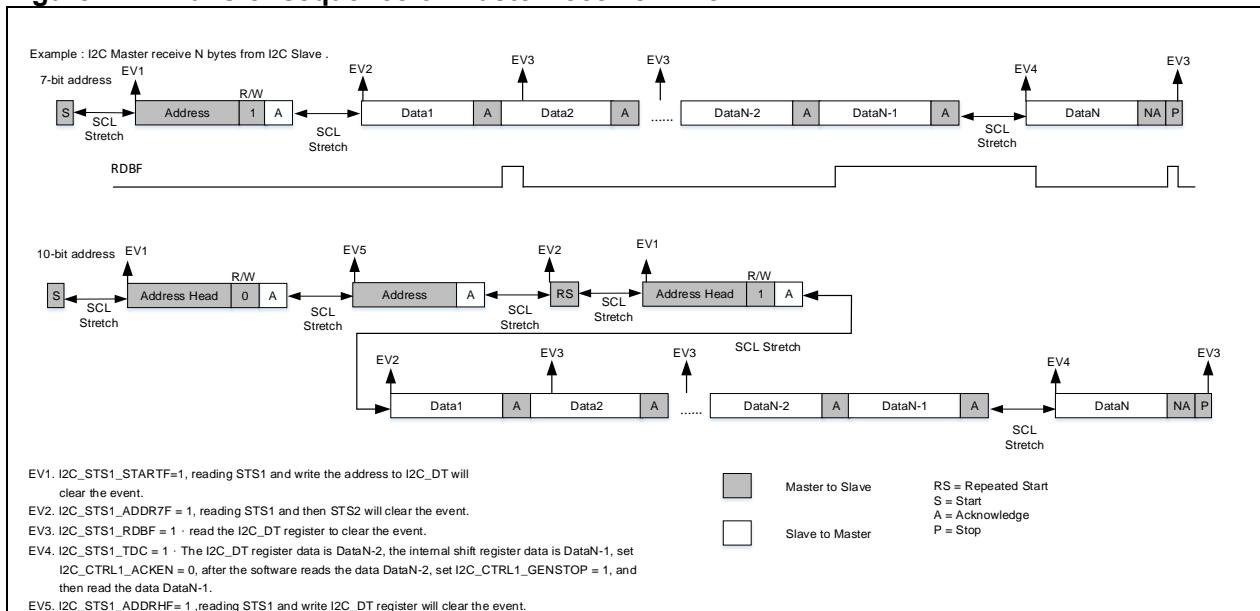
1. Generate Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
3. EV5: 10-bit address head sequence is sent. Reading STS1 and writing to DT register can clear the ADDRHF bit.
4. EV2: Address is matched successfully (ADDR7F=1). Reading STS1 and then STS2 clears the ADDR7F bit, and the master re-send a Start condition (GENSTART=1).
5. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
6. EV2: Address is matched successfully (ADDR7F=1). Reading STS1 and then STS2 clears the ADDR7F bit. The master enters receive stage..
7. EV3: The RDBF bit is set 1 after receiving a byte. Reading the I2C\_DT register clears the RDBF.
8. EV4: Once the second-to-last byte is received, the ACKEN bit must be cleared and the GENSTOP must be set by software.
9. EV3: The RDBF bit is set 1 after receiving the byte. Reading the I2C\_DT register clears the RDBF.
10. End of communication.

### 2. When I<sup>2</sup>C interrupt priority is not very high but the number of bytes to receive is greater than 2

- Do not read the third-to-last byte (N-2) that is received. Clear the ACKEN bit in the I2C\_CTRL1 register after receiving the second-to-last byte (N-1). Then read the third-to-last byte (N-2), set

the GENSTOP bit in the I2C\_CTRL1 register and read the second-to-last byte (N-1). The bus then starts to receive the last one byte.

**Figure 11-7 Transfer sequence of master receiver when N>2**



- **7-bit address mode:**

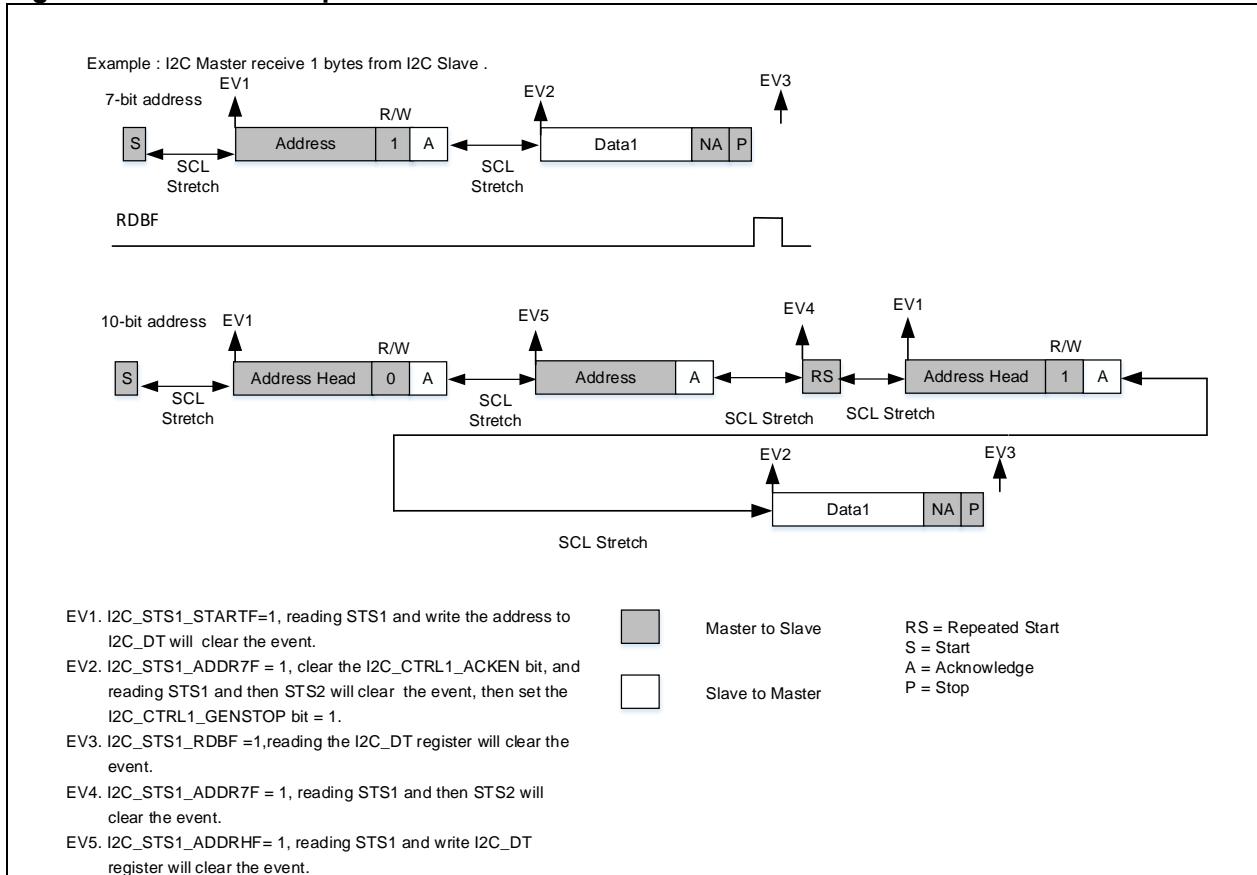
1. Generate a Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
3. EV2: Address is matched successfully (ADDR7F=1). Reading STS1 and then STS2 clears the ADDR7F bit, and the master enters receive stage.
4. EV3: The RDBF bit is set 1 after receiving the byte. Reading the I2C\_DT register clears the RDBF.
5. EV4: TDC=1, the contents in the I2C\_DT is N-2, and that of the shift register is N-1. The ACKEN is set 0 by software and the data N-2 is read, afterwards, the GENSTOP=1, and data N-1 is read.
6. EV3: The RDBF bit is set 1 after receiving the byte. Reading the I2C\_DT register clears the RDBF.
7. End of communication.

- **10-bit address mode:**

1. Generate Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
3. EV5: 10-bit address head sequence is sent. Reading STS1 and writing to DT register can clear the ADDRHF bit.
4. EV2: Address is matched successfully (ADDR7F=1). Reading STS1 and then STS2 clears the ADDR7F bit, and the master re-send Start condition.
5. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to the DT register.
6. EV2: Address is matched successfully (ADDR7F=1). Reading STS1 and then STS2 clears the ADDR7F bit, and the master enters receive stage.
7. EV3: The RDBF bit is set 1 after receiving a byte. Reading the I2C\_DT register clears the RDBF.
8. EV4: TDC=1, the contents in the I2C\_DT is N-2, and that of the shift register is N-1. The ACKEN is set 0 by software and the data N-2 is read, afterwards, the GENSTOP=1, and data N-1 is read.
9. EV3: The RDBF bit is set 1 after receiving a byte. Reading the I2C\_DT register clears the RDBF.

10. End of communication.
3. When I<sup>2</sup>C interrupt priority is not very high but the number of bytes to receive is equal to 2
- Set the MACKCTRL bit in the I2C\_CTRL1 register before data reception. When the address is matched, clear ACKEN bit and then the ADDR7F bit. When the TDC bit is set 1, set the GENSTOP bit in the I2C\_CTRL1 register, and then read the DT register.

**Figure 11-8 Transfer sequence of master receiver when N=2**



● **7-bit address mode:**

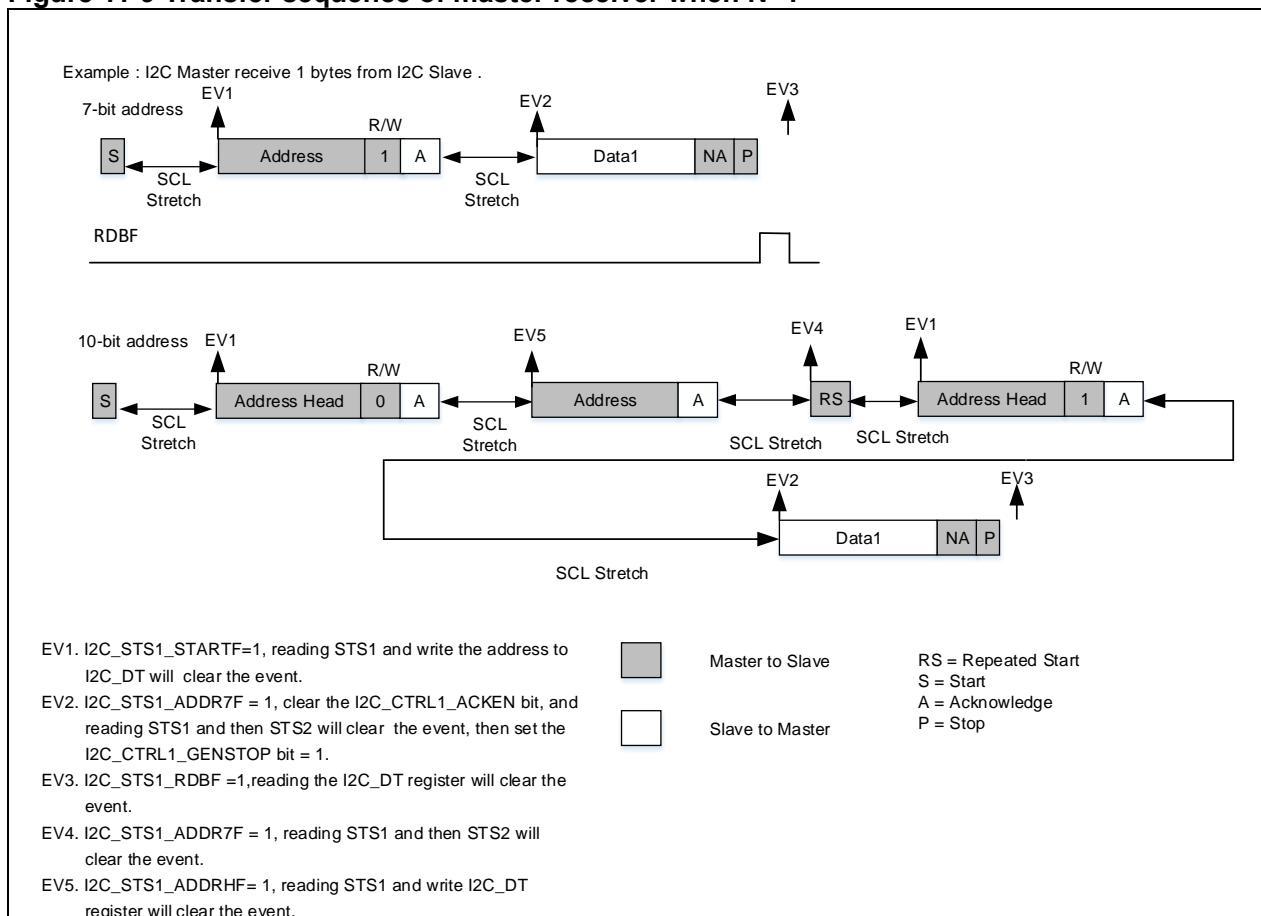
1. Set MACKCTRL=1 in the I2C\_CTRL1 register
2. Generate Start condition (GENSTART=1)
3. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
4. EV2: Address is matched successfully (ADDR7F=1). Clear the ACKEN bit and read STS1 and STS2 clears the ADDR7F bit, the master enters receive state at this time.
5. EV2: TDC=1, set GENSTOP=1, and then read the I2C\_DT register twice.
6. End of communication.

● **10-bit address mode:**

1. Set MACKCTRL=1 in the I2C\_CTRL1 register
2. Generate Start condition (GENSTART=1)
3. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
4. EV4: 10-bit address head is sent. Reading STS1 and writing to DT register can clear the ADDRHF bit.
5. EV2: Address is matched successfully (ADDR7F=1). Reading STS1 and then STS2 clears the ADDR7F bit, and the master re-send Start condition.
6. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to the DT register.
7. EV2: Address is matched successfully (ADDR7F=1). Clear the ACKEN bit and read STS1 and STS2 clears the ADDR7F bit, the master enters receive state at this time.
8. EV3: TDC=1, set GENSTOP=1, and then read the I2C\_DT register twice.

9. End of communication.
4. When I<sup>2</sup>C interrupt priority is not very high but the number of bytes to receive is equal to 1
- After the address is matched, clear the ACKEN bit and then ADDR7F bit, then set the GENSTOP bit in the I2C\_CTRL1 register. Wait until the RDBF bit is set 1 before reading the DT register.

**Figure 11-9 Transfer sequence of master receiver when N=1**



- **7-bit address mode:**

1. Generate a Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
3. EV2: Address is matched successfully (ADDR7F=1). Clear the ACKEN bit, reading STS1 and then STS2 clears the ADDR7F bit. Afterwards, set GENSTOP=1, the master enters receive stage at this time.
4. EV3: RDBF=1. Reading the I2C\_DT register clears the RDBF.
5. End of communication.

- **10-bit address mode:**

1. Generate a Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
3. EV5: 10-bit address head is sent. Read STS1 and write to DT register can clear the ADDRHF bit.
4. EV4: Address is matched successfully (ADDR7F=1). Reading STS1 and STS2 clears the ADDR7F bit, the master re-sends Start condition (GENSTART=1).
5. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to the DT register.
6. EV2: Address is matched successfully (ADDR7F=1). Clearing the ACKEN bit, read STS1 and then STS2 clears the ADDR7F bit. Afterwards, set GENSTOP=1, the master enters receive stage at this time.
7. EV3: RDBF=1. Reading the I2C\_DT register clears the RDBF.
8. End of communication.

### 11.4.3 Data transfer using DMA

I<sup>2</sup>C data transfer can be done using DMA controller. An interrupt is generated by enabling the transfer complete interrupt bit. The DATAIEN bit in the I2C\_CTRL2 register must be set 0 when using DMA for data transfer. The following sequence is for data transfer with DMA.

#### Transmission using DMA

1. Set the peripheral address (DMA\_CxPADDR= I2C\_DT address)
2. Set the memory address (DMA\_CxMADDR=data memory address)
3. The transmission direction is set from memory to peripheral (DTD=1 in the DMA\_CHCTRL register)
4. Configure the total number of bytes to be transferred in the DMA\_CxDTCNT register
5. Configure other parameters such as priority, memory data width, peripheral data width, interrupts, etc., in the DMA\_CHCTRL register
6. Enable a DMA channel by setting CHEN=1 in the DMA\_CxCTRL register
7. Enable I<sup>2</sup>C DMA request by setting DMAEN=1 in the I2C\_CTRL2 register. Once the TDBE bit in the I2C\_STS1 register is set, the data is loaded from the programmed memory to the I2C\_DT register through DMA
8. When the number of data transfers, programmed in the DMA controller, is reached (DMA\_CxDTCNT=0), the data transfer is complete (An interrupt is generated if enabled).
9. Master transmitter: Once the TDC flag is set, the STOP condition is generated, indicating that transfer is complete.

Slave transmitter: Once the ACKFAIL flag is set, clear the ACKFAIL flag, transfer is complete.

#### Reception using DMA

1. Set the peripheral address (DMA\_CxPADDR = I2C\_DT address)
2. Set the memory address (DMA\_CxMADDR = memory address)
3. The transmission directions set from peripheral to memory (DTD=0 in the DMA\_CHCTRL register)
4. Configure the total number of bytes to be transferred in the DMA\_CxDTCNT register
5. Configure other parameters such as priority, memory data width, peripheral data width, interrupts, etc., in the DMA\_CHCTRL register
6. Enable the DMA channel by setting CHEN=1 in the DMA\_CxCTRL register
7. Enable I<sup>2</sup>C DMA request by setting DMAEN=1 in the I2C\_CTRL2 register. Once the RDBE bit in the I2C\_STS1 register is set, the data is loaded from the I2C\_DT register to the programmed memory through DMA
8. When the number of data transfers, programmed in the DMA controller, is reached (DMA\_CxDTCNT=0), the data transfer is complete (An interrupt is generated if enabled).
9. Master receiver: Clear the ACKFAIL flag, the STOP condition is generated, indicating that the transfer is complete (when the number of bytes to be transferred is greater >=2 and DMAEND=1, the NACK signal is generated automatically after transfer complete (DMA\_CxDTCNT=0))

Slave receiver: Wait until the STOPF flag is set, clear the STOPF flag, and the transfer is complete.

## 11.4.4 SMBus

The System Management Bus (SMBus) is a two-wire interface through which various devices can communicate with each other. It is based on I<sup>2</sup>C. With SMBus, the device can provide manufacturer information, tell the system its model/part number, report different types of errors and accept control parameters and so on. For more information, refer to SMBus 2.0 protocol.

### Differences between SMBus and I<sup>2</sup>C

1. SMBus requires a minimum speed of 10 kHz for the purpose of management and monitor. It is quite easy to know whether the bus is in Idle state or not as long as a parameter is input while running on a certain transmission speed, without the need of detecting the STOP signals one after another, or even keeping STOP and other parameter monitor. There is no limit for I<sup>2</sup>C.
2. SMBus transmission speed ranges from 10 kHz to 100 kHz. In contrast, I<sup>2</sup>C has no minimum requirement, and its maximum speed varies from one mode to another, namely, 100 kHz in standard mode and 400 kHz in fast mode.
3. After reset, SMBus needs timeout (35ms), but there is no limit for I<sup>2</sup>C in this regard.

### SMBus applications

1. The I<sup>2</sup>C interface is set in SMBus mode by setting PERMODE=1 in the I2C\_CTRL1 register.
2. Select SMBus mode:  
SMBMODE=1: SMBus host  
SMBMODE=0: SMBus device
3. Other configurations are the same as those of I<sup>2</sup>C.

SMBus protocols are implemented by the user software, while I<sup>2</sup>C interface only provide the address identification of these protocols.

### SMBus address resolution protocol (ARP)

SMBus address conflicts can be resolved by dynamically assigning a new unique address to each device. Refer to SMBus 2.0 protocol for more information about ARP.

Setting the ARPEN bit can enable the I<sup>2</sup>C interface to recognize the default device address (0b1100001x). However, unique device identifier (UDID) and the detailed protocol implementation should be handled by software.

### SMBus host notify protocol

The slave device can send data to the master device through SMBus host notify protocol. For example, the slave can notify the host to implement ARP with this protocol. Refer to SMBus 2.0 protocol for details on SMBus host notify protocol.

When the ARP mode is enabled (ARPEN=1) in host mode (SMBMODE=1), the I<sup>2</sup>C interface is enabled to recognize the 0b0001000x (default host address)

### SMBus Alert

SMBALERT is an optional signal that connects the ALERT pin between the host and the slave. With this signal, the slave notifies the host to access the slave. SMBALERT is a wired-AND signal. For more information about SMBus Alert, refer to SMBus2.0 protocol.

The detailed sequences are as follows:

SMBus host:

1. Enable SMBus Alert mode by setting SMBALERT=1
2. Enable ALERT interrupt if necessary
3. When an alert event occurs on the ALERT pin (ALERT pin changes from high to low)
4. The host will generate ALERT interrupt if enabled
5. The host then processes the interrupt and accesses to all devices through ARA (Alert Response Address 0001100x) so as to get the slave addresses. Only the devices with pulled-down SMBALERT can acknowledge ARA.
6. The host then continues to operate based on the slave addresses available.

SMBus slave:

1. When an alert event occurs and the ALERT pin changes from high to low (SMBALERT=1), the slave responds to ARA (Alert Response Address) address (0001100x)
2. Enable ALERT interrupt if necessary (an interrupt is generated when receiving ARA address)
3. Wait until the host gets the slave addresses through ARA
4. Report its own address, but it continues to wait if the arbitration is lost.
5. Address is reported properly, and the ALERT pin is released (SMBALERT=0).

#### Packet error checking (PEC)

Packet error checking (PEC) is used to guarantee the correctness and integrity of data transfer. This is done by using CRC-8 polynomial:

$$C(x) = x^8 + x^2 + x + 1$$

PEC calculation is enabled when PECEN=1 to check address and data. It becomes invalid when the arbitration is lost.

PEC transmission:

- Common mode: Set PECTRA=1 after the last TDBE event so that PEC is sent after the last byte.
- DMA mode: The PEC is sent automatically after the completion of the last byte transfer. For example, if the number of data to be transferred is 8, then DMA\_TCNTx=8 must be set.

PEC reception:

- Common mode: The PECTRA bit is set after the last RDBF event. The PECTRA must be set before the ACK pulse of the current byte is received.
- DMA mode: The last byte is automatically checked as PECVAL during reception. For instance, if the number of data to be transferred is 8, then DMA\_TCNTx=9 must be set.

In reception mode, the NACK will be generated if PEC check fails.

### 11.4.5 I<sup>2</sup>C interrupt requests

The following table lists all the I<sup>2</sup>C interrupt requests.

Interrupt event	Event flag	Enable control bit
Start condition sent (Host)	STARTF	EVTIEN
Address sent (host) or address matched (slave)	ADDR7F	
10-bit address head sent (host)	ADDRHF	
Data transfer complete	TDC	
Stop condition received (slave)	STOPF	
Transmit data buffer empty	TDBE	EVTIEN and DATAIEN
Receive data buffer full	RDBF	
SMBus alert	ALERTF	
Timeout error	TMOUT	ERRIEN
PEC error	PECERR	
Overload/underload	OUF	
Acknowledge failure	ACKFAIL	
Arbitration lost	ARLOST	
Bus error	BUSERR	

## 11.4.6 I<sup>2</sup>C debug mode

When the microcontroller enters debug mode (Cortex<sup>TM</sup>-M4 halted), the SMBUS timeout feature either continues to work or stops, depending on the I2Cx\_SMBUS\_TIMEOUT configuration bit in the DEBUG module.

## 11.5 I<sup>2</sup>C registers

These peripheral registers must be accessed by words (32 bits).

**Table 11-1 I<sup>2</sup>C register map and reset values**

Register	Offset	Reset value
I2C_CTRL1	0x00	0x0000
I2C_CTRL2	0x04	0x0000
I2C_OADDR1	0x08	0x0000
I2C_OADDR2	0x0C	0x0000
I2C_DT	0x10	0x0000
I2C_STS1	0x14	0x0000
I2C_STS2	0x18	0x0000
I2C_CLKCTRL	0x1C	0x0000
I2C_TMRISE	0x20	0x0002

### 11.5.1 Control register1 (I2C\_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15	RESET	0x0	rw	I <sup>2</sup> C peripheral reset 0: I <sup>2</sup> C peripheral is not at reset state. 1: I <sup>2</sup> C peripheral is at reset state. Note: This bit can be used only when the BUSYF bit is "1", and no Stop condition is detected on the bus.
Bit 14	Reserved	0x0	resd	Kept at its default value.
Bit 13	SMBALERT	0x0	rw	SMBus alert pin set This bit is set or cleared by software. It is cleared by hardware when I2CEN=0. 0: SMBus alert pin high. 1: SMBus alert pin low.
Bit 12	PECTEN	0x0	rw	Request PEC transfer enable This bit is set or cleared by software. It is cleared by hardware after PECTEN is sent, or under Start/Stop condition. 0: No PEC transfer 1: PEC transfer
Bit 11	MACKCTRL	0x0	rw	Master receive mode acknowledge control 0: ACKEN bit controls ACK of the current byte being transferred 1: ACKEN bit controls ACK of the next byte to be transferred. This bit is used only when the number of bytes to receive is equal to 2 so as to ensure that the host responds to ACK in time.
Bit 10	ACKEN	0x0	rw	Acknowledge enable This bit is set or cleared by software. 0: Disabled (no acknowledge sent) 1: Enabled (acknowledge sent)

				Generate stop condition This bit is set or cleared by software. It is cleared when a Stop condition is detected. It is set by hardware when a timeout error is detected. 0: No Stop condition is generated. 1: Stop condition is generate.
Bit 9	GENSTOP	0x0	rw	The slave releases the SCL and SDA lines when this bit is set in slave mode.
Bit 8	GENSTART	0x0	rw	Generate start condition This bit is set or cleared by software. It is cleared when a Start condition is sent. 0: No Start condition is generated. 1: Start condition is generated.
Bit 7	STRETCH	0x0	rw	Clock stretching mode 0: Enabled 1: Disabled Note: This feature applies to slave mode only.
Bit 6	GCAEN	0x0	rw	General call address enable 0: Enabled 1: Disabled
Bit 5	PECEN	0x0	rw	PEC calculation enable 0: Disabled 1: Enabled
Bit 4	ARPEN	0x0	rw	SMBus address resolution protocol enable 0: Disabled 1: Enabled SMBus host: response to host address 0001000x SMBus slave: response to default device address 0001100x
Bit 3	SMBMODE	0x0	rw	SMBus device mode 0: SMBus slave 1: SMBus host
Bit 2	Reserved	0x0	resd	Forced to be 0 by hardware.
Bit 1	PERMODE	0x0	rw	I <sup>2</sup> C peripheral mode 0: I <sup>2</sup> C mode 1: SMBus mode
Bit 0	I2CEN	0x0	rw	I <sup>2</sup> C peripheral enable 0: Disabled 1: Enabled All bits are cleared as I2CEN=0 at the end of the communication. In master mode, this bit must not be cleared before the end of the communication.

Note: When the GENSTART, GENSTP or PECTEN bit is set, the I2C\_CTRL1 cannot be written by software until the corresponding bit has been cleared by hardware, otherwise, a second GENSTART, GENSTP or PECTEN request may be set.

### 11.5.2 Control register2 (I2C\_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Forced to be 0 by hardware.
Bit 12	DMAEND	0x0	rw	End of DMA transfer 0: The next DMA transfer is no the last one. 1: The next DMA transfer is the last one.
Bit 11	DMAEN	0x0	rw	DMA transfer enable 0: Disabled

				1: Enabled
Bit 10	DATAIEN	0x0	rw	Data transfer interrupt enable An interrupt is generated when TDBE =1 or RDBF=1. 0: Disabled 1: Enabled
Bit 9	EVTIEN	0x0	rw	Event interrupt enable 0: Disabled 1: Enabled An interrupt is generated in the following conditions: – STARTF = 1 (Master mode) – ADDR7F = 1 (Master/slave mode) – ADDRHf= 1 (Master mode) – STOPF = 1 (Slave mode) – TDC = 1, but no TDBE or RDBF event – If DATAIEN = 1, the TDBE event is 1. – If DATAIEN = 1, the RDBF event is 1.
Bit 8	ERRIEN	0x0	rw	Error interrupt enable 0: Disabled 1: Enabled An interrupt is generated in the following conditions: – BUSERR = 1 – ARLOST = 1 – ACKFAIL = 1 – OVER = 1 – PECERR = 1 – TMOUT = 1 – ALERTF = 1
Bit 7: 0	CLKFREQ	0x00	rw	I <sup>2</sup> C input clock frequency Correct input clock frequency must be set to generate correct timings. The range allowed is between 2 MHz and 120 MHz. 2: 2MHz 3: 3MHz ..... 120: 120MHz

### 11.5.3 Own address register1 (I2C\_OADDR1)

Bit	Register	Reset value	Type	Description
Bit 15	ADDR1MODE	0x0	rw	Address mode 0: 7-bit address 1: 10-bit address
Bit 14: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9: 0	ADDR1	0x000	rw	Own address1 In 7-bit address mode, bit 0 and bit [9: 8] don't care.

### 11.5.4 Own address register2 (I2C\_OADDR2)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7: 1	ADDR2	0x00	rw	Own address 2 7-bit address
Bit 0	ADDR2EN	0x0	rw	Own address 2 enable 0: In 7-bit address mode, only OADDR1 is recognized. 1: In 7-bit address mode, both OADDR1 and OADDR2 are

---

recognized.

---

### 11.5.5 Data register (I2C\_DT)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value
Bit 7: 0	DT[7: 0]	0x00	rw	<p>This field is used to store data received or to be transferred.</p> <p>Transmitter mode: Data transfer starts automatically when a byte is written to the DT register. Once the transfer starts (TDE=1), I<sup>2</sup>C will keep a continuous data transfer flow if the next data to be transferred is written to the DT register in a timely manner.</p> <p>Receiver mode: Bytes received are copied into the DT register (RDNE=1). A continuous data transfer flow can be maintained if the DT register is read before the next word is received (RDNE=1).</p> <p>Note: If an ARLOST event occurs on ACK pulse, the received byte is not copied into the data register, so it cannot be read.</p>

### 11.5.6 Status register1 (I2C\_STS1)

Bit	Register	Reset value	Type	Description
Bit 15	ALERTF	0x0	rw0c	<p>SMBus alert flag</p> <p>In SMBus host mode:</p> <p>0: No SMBus alert</p> <p>1: SMBus alert event is received.</p> <p>In SMBus slave mode:</p> <p>It indicates the receiving status of the default device address (0001100x)</p> <p>0: Default device address is not received.</p> <p>1: Default device address is received.</p> <p>This bit is cleared by software, or by hardware when I2CEN=0.</p>
Bit 14	TMOUT	0x0	rw0c	<p>SMBus timeout flag</p> <p>0: No timeout error.</p> <p>1: Timeout</p> <p>This bit is cleared by software, or by hardware when I2CEN=0.</p> <p>Note: This function is valid only in SMBUS mode.</p>
Bit 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	PECERR	0x0	rw0c	<p>PEC receive error flag</p> <p>0: No PEC error</p> <p>1: PEC error occurs.</p> <p>This bit is cleared by software.</p>
Bit 11	OUF	0x0	rw0c	<p>Overload / underload flag</p> <p>In transmission mode:</p> <p>0: Normal</p> <p>1: Underload</p> <p>In reception mode:</p> <p>0: Normal</p> <p>1: Overload</p> <p>This bit is cleared by software, or by hardware when I2CEN=0.</p>
Bit 10	ACKFAIL	0x0	rw0c	Acknowledge failure flag 0: No acknowledge failure

				1: Acknowledge failure occurs. Set by hardware when no acknowledge is returned. This bit is cleared by software, or by hardware when I2CEN=0.
Bit 9	ARLOST	0x0	rw0c	Arbitration lost flag 0: No arbitration lost is detected. 1: Arbitration lost is detected. This bit is cleared by software, or by hardware when I2CEN=0. On ARLOST even, the I <sup>2</sup> C interface switches to slave mode automatically.
Bit 8	BUSERR	0x0	rw0c	Bus error flag 0: No Bus error occurs. 1: Bus error occurs. Set by hardware when the interface detects a misplaced Start or Stop condition. This bit is cleared by software, or by hardware when I2CEN=0.
Bit 7	TDBE	0x0	ro	Transmit data buffer empty flag 0: The data is being transferred from the DT register to the shift register (the data is still loaded with the data at this point.) 1: The data has been moved from the DT register to the shift register. The data register is empty now. This flag is set when the DT register is empty, and cleared when writing to the DT register. Note: The TDBE bit is not cleared by writing the first data to be transmitted, or by writing data when the TDC is set, since the data register is still empty at this time.
Bit 6	RDBF	0x0	ro	Receive data buffer full flag 0: Data register is empty. 1: Data register is full (data received) This flag is cleared when the DT register is read. The RDBF bit is not set at ARLOST event.
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	STOPF	0x0	ro	Stop condition generation complete flag 0: No Stop condition is detected. 1: Stop condition is detected. This bit is set by hardware when a Stop condition is detected on the bus by the slave if ACKEN=1. It is cleared by reading STS1 register followed by writing to the CTRL1 register.
Bit 3	ADDRHF	0x0	ro	Master 9~8 bit address head match flag 0: Master 9~8 bit address head mismatch 1: Master 9~8 bit address head match Set by hardware when the first byte is sent by master in 10-bit address mode. Cleared by a write to the CTRL1 register after the STS1 register is read by software, or by hardware when PEN=0. Note: The ADDR10 bit is not set after a NACK reception.
Bit 2	TDC	0x0	ro	Data transfer complete flag 0: Data transfer is not completed yet (the shift register still holds data) 1: Data transfer is completed (shift register is empty) This bit is cleared automatically by read or write access to the DT register, or when a Start or Stop condition is

---

				received. When STRETCH=0 In reception mode, when a new byte (including ACK pulse) is received and the data register is not read yet (RDBF=1) In transmission mode, when a new byte is sent and the data register is not written yet (TDBE=1) The TDC is set under both conditions.
Bit 1	ADDR7F	0x0	ro	0~7 bit address match flag 0: Address is not sent in host mode or received in slave mode 1: Address is sent in host mode or address is received in slave mode. Cleared by read access to STS2 register after the software reads STS1 register. Note: the ADDR7F bit is not set after a NACK reception.
Bit 0	STARTF	0x0	ro	Start condition generation complete flag 0: No Start condition is generated. 1: Start condition is generated. Cleared by write access to the DT register after the software reads the STS1 register.

---

### 11.5.7 Status register2 (I2C\_STS2)

Bit	Register	Reset value	Type	Description
Bit 15: 8	PECVAL	0x00	ro	PEC value Cleared when PECEN is reset.
Bit 7	ADDR2F	0x0	ro	Received address 2 flag 0: Received address matches the contents of OADDR1 1: Received address matches the contents of OADDR2 Cleared when a Stop/Start condition is received, or by hardware when I2CEN=0.
Bit 6	HOSTADDRF	0x0	ro	SMBus host address reception flag 0: SMBus host address is not received. 1: SMBus host address is received. Cleared when a Stop/Start condition is received, or by hardware when I2CEN=0.
Bit 5	DEVADDRF	0x0	ro	SMBus device address reception flag 0: SMBus device address is not received. 1: SMBus device address is received. Cleared when a Stop/Start condition is received, or by hardware when I2CEN=0.
Bit 4	GCADDRF	0x0	ro	General call address reception flag 0: General call address is not received. 1: General call address is received. Cleared when a Stop/Start condition is received, or by hardware when I2CEN=0.
Bit 3	Reserved	0x0	resd	Keep at its default value.
Bit 2	DIRF	0x0	ro	Transmission direction flag 0: Data reception 1: Data transmission Cleared by hardware when a Stop condition is received.
Bit 1	BUSYF	0x0	ro	Bus busy flag transmission mode 0: Bus idle 1: Bus busy Set by hardware on detection of SDA/SCL low, and

---

				cleared by hardware on detection of a Stop condition.
				Transmission mode
				0: Slave mode 1: Master mode
Bit 0	TRMODE	0x0	ro	Set by hardware when the GENSTART is set and a Start condition is sent. Cleared by hardware when a Stop condition is detected.

### 11.5.8 Clock control register (I2C\_CLKCTRL)

Bit	Register	Reset value	Type	Description
Bit 15	SPEEDMODE	0x0	rw	<p>Speed mode selection</p> <p>0: Standard mode (up to 100 kHz) 1: Fast mode (up to 400 kHz)</p> <p>In fast mode, an accurate 400kHz clock is generated when the I<sup>2</sup>C clock frequency is an integer multiple of 10MHz.</p>
Bit 14	DUTYMODE	0x0	rw	<p>Fast mode duty cycle</p> <p>0: The ratio of High to low is 1:2. 1: The ratio of low to high is 9:16.</p>
Bit 13: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11: 0	SPEED	0x000	rw	<p>I<sup>2</sup>C bus speed config</p> <p>In standard mode: High level= SPEED x T<sub>I2C_CLK</sub> Low level= SPEED x T<sub>I2C_CLK</sub></p> <p>In fast mode: DUTYMODE = 0: High level= SPEED x T<sub>I2C_CLK</sub> x 1 Low level= SPEED x T<sub>I2C_CLK</sub> x 2 DUTYMODE = 1: High level= SPEED x T<sub>I2C_CLK</sub> x 9 Low level= SPEED x T<sub>I2C_CLK</sub> x 16</p> <p>The minimum value allowed in standard mode is 4. In fast mode, the minimum value allowed is 1.</p> <p>The CLKCTRL register can be configured only when the I<sup>2</sup>C is disabled (I2CEN=0).</p>

Note: The I<sup>2</sup>C\_CLKCTRL register can be configured only when the I<sup>2</sup>C is disabled (I2CEN=0).

### 11.5.9 Clock rise time register (I2C\_TMRISE)

Bit	Register	Reset value	Type	Description
Bit 15: 6	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 5: 0	RISETIME	0x02	rw	<p>I<sup>2</sup>C bus rise time Time= RISETIME x T<sub>I2C_CLK</sub></p> <p>In standard mode, I<sup>2</sup>C protocol stand is 1000ns, and the formula is:</p> $\text{RISETIME} = F_{I2C\_CLK} + 1$ <p>For example, when I<sup>2</sup>C clock is 48MHz, RISETIME = 48+1</p> <p>In fast mode, I<sup>2</sup>C protocol stand is 300ns, and the formula is:</p> $\text{RISETIME} = F_{I2C\_CLK} \times 0.3 + 1$ <p>For example, when I<sup>2</sup>C clock is 48MHz, RISETIME = 48×0.3+1</p> <p>Note: RISETIME[5:0] can be configured only when I2CEN=0.</p>

# 12 Universal synchronous/asynchronous receiver/transmitter (USART)

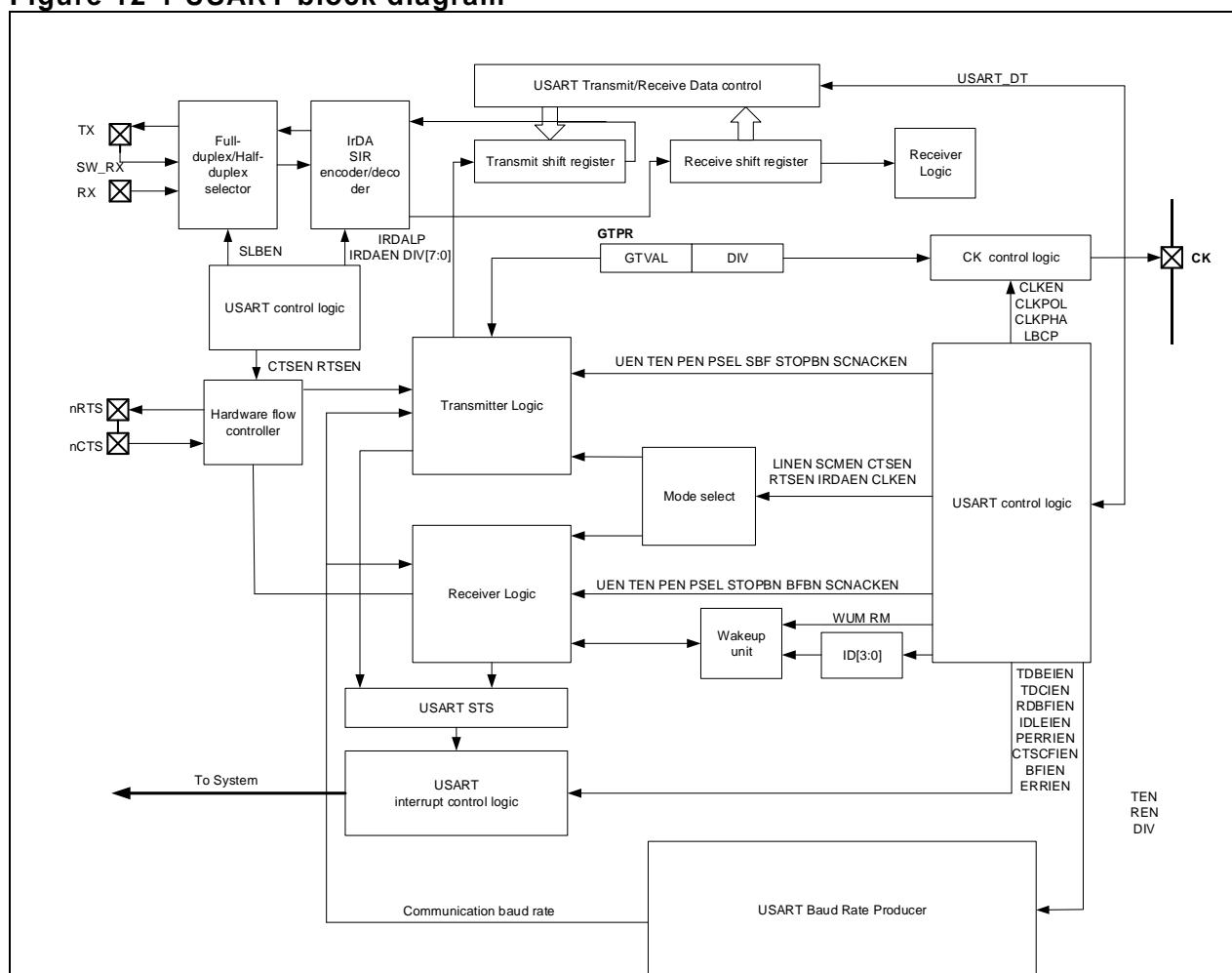
## 12.1 USART introduction

The universal synchronous/asynchronous receiver/transmitter (USART) acts as a general-purpose interface for communication by means of various configurations and peripherals with different data formats. It supports asynchronous full-duplex and half-duplex as well as synchronous transfer. It offers a programmable baud rate generator, generating up to 7.5 Mbits/s of baud rate by setting the system frequency and frequency divider. It is also possible for users to configure the required communication frequency by setting system clocks and frequency division factors.

In addition to standard NRZ asynchronous and synchronous receiver/transmitter communication protocols, USART also supports widely-used serial communication protocols such as LIN (Local Interconnection Network), IrDA (Infrared Data Association) SIRENDEC specification, Asynchronous SmartCard protocol defined in ISO7816-3 standard, and CTS/RTS (Clear To Send/Request To Send) hardware flow operation.

The USART also supports silent mode that allows multi-processor communication and can be woken up by a programmable idle frames or ID matching, so as to build up a USART network. Meanwhile, high-speed communication is possible by using DMA.

**Figure 12-1 USART block diagram**



### USART main features:

- Programmable full-duplex or half-duplex communication
  - Full-duplex, asynchronous communication

- Half-duplex, single communication
- Programmable communication modes
  - NRZ standard format (Mark/Space)
  - LIN (Local Interconnection Network):
    - IrDA SIR:
    - Asynchronous SmartCard protocol defined in ISO7816-3 standard: Support 0.5 or 1.5 stop bits in Smartcard mode
    - RS-232 CTS/RTS (Clear To Send/Request To Send) hardware flow operation
    - Multi-processor communication in silent mode (woken up by a programmable ID matching or bus idle frame)
    - Synchronous mode
  - Programmable baud rate generator
    - Shared by transmission and reception, up to 7.5 Mbits/s
  - Programmable frame format
    - Programmable data word length (8 bits or 9 bits)
    - Programmable stop bits-support 1 or 2 stop bits
    - Programmable parity control: transmitter with parity bit transmission capability, and receiver with received data parity check capability
  - Programmable DMA multi-buffer communication
  - Programmable separate enable bits for transmitter and receiver
  - Programmable output CLK phase, polarity and frequency
  - Detection flags
    - Receive buffer full
    - Transmit buffer empty
    - Transfer complete flag
  - Four error detection flags
    - Overrun error
    - Noise error
    - Framing error
    - Parity error
  - Programmable 10 interrupt sources with flags
    - CTSF changes
    - LIN brake detection
    - Transmit data register empty
    - Transmission complete
    - Receive data register full
    - Idle bus detected
    - Overrun error
    - Framing error
    - Noise error
    - Parity error

## 12.2 Full-duplex/half-duplex selector

The full-duplex and half-duplex selector enables USART to perform data exchanges with peripherals in full-duplex or half-duplex mode, which is achieved by setting the corresponding registers.

In two-wire unidirectional full-duplex mode (by default), TX pin is used for data output, while the RX pin is used for data input. Since the transmitter and receiver are independent of each other, USART is allowed to send/receive data at the same time so as to achieve full-duplex communication.

When the HALFSEL is set 1, the single-wire bidirectional half-duplex mode is selected for communication. In this case, the LINEN, CLKEN, SCMEN and IRDAEN bits must be set 0. RX pin is inactive, while TX and SW\_RX are interconnected inside the USART. For the USART part, TX pins is used for data output, and SW\_RX for data input. For the peripheral part, bidirectional data transfer is executed through IO mapped by TX pin.

## 12.3 Mode selector

### 12.3.1 Introduction

USART mode selector allows USART to work in different operation modes through software configuration so as to enable data exchanges between USART and peripherals with different communication protocols.

USART supports NRZ standard format (Mark/Space), by default. It also supports LIN (Local Interconnection Network), IrDA SIR (Serial Infrared), Asynchronous Smartcard protocol in ISO7816-3 standard, RS-232 CTS/RTS (Clear To Send/Request To Send) hardware flow operation, silent mode and synchronous mode, depending on USART mode selection configuration.

### 12.3.2 Configuration procedure

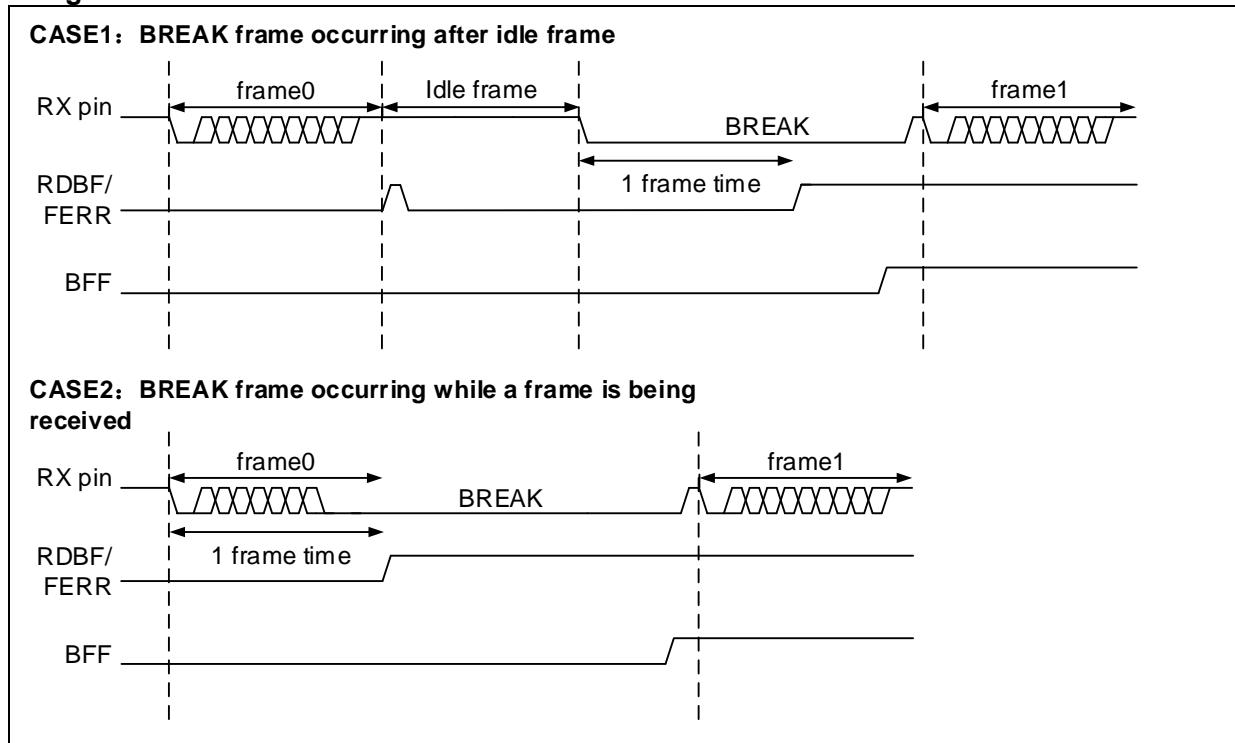
Selection of operation mode is done by following the configuration process listed below. In addition, such configuration method, along with that of receiver and transmitter described in the subsequent sections, are used to make USART initialization configuration.

#### 1. LIN mode:

Parameters configuration: LINEN=1, CLKEN=0, STOPBN[1: 0]=0, SCMEN=0, SLHDEN=0, IRDAEN=0 and DBN=0.

LIN master has brake frame transmission capability, and thus it is able to send 13-bit low-level LIN synchronous brake frame by setting SBF=1.

Similarly, LIN slave has brake frame detection capability, and thus it is able to detect 11-bit or 10-bit brake fame, depending on whether BFBN=1 or BFBN=0.

**Figure 12-2 BFF and FERR detection in LIN mode**

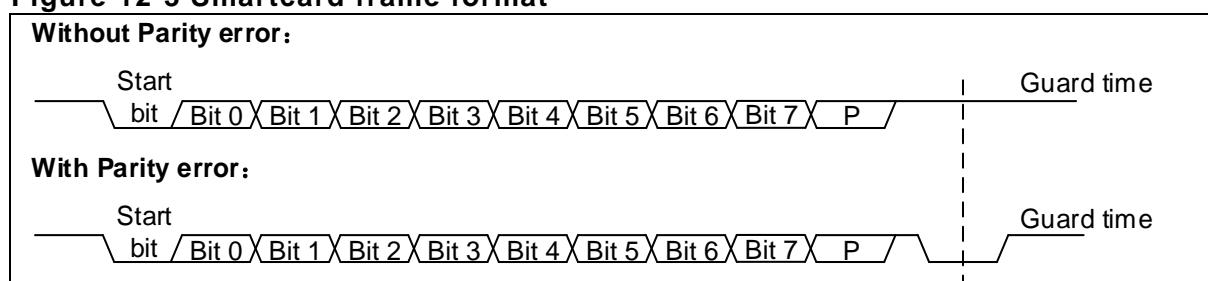
## 2. Smartcard mode:

Parameters configuration: SCMEN=1, LINEN=0, SLHDEN=0, IRDAEN=0, CLKEN=1, DBN=1, PEN=1, and STOPBN[1: 0]=11.

The polarity, phase and pulse number of the clock can be configured by setting the CLKPOL, CLKPHASE and LBCP bits (Refer to Synchronous mode for details).

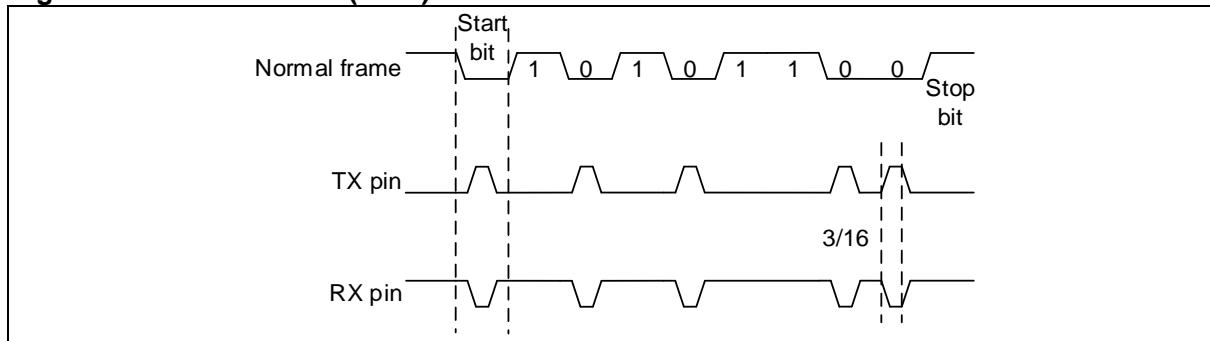
The assertion of the TDC flag can be delayed by setting the SCGT[7: 0] bit (guard time bit). The TDF bit can be asserted high after the guard time counter reaches the value programmed in the SCGT[7: 0] bit.

The Smartcard is a single-wire half duplex communication protocol. The SCNACKEN bit is used to select whether to send NACK when a parity error occurs. This is to indicate to the Smarcard that the data has not been correctly received

**Figure 12-3 Smartcard frame format**

## 3. Infrared mode:

Parameters configuration: IRDAEN=1, CLKEN=0, STOPBN[1: 0]=0, SCMEN=0 and SLHDEN=0. The infrared low-power mode can be enabled by setting IRDALP=1. In normal mode the transmitted pulse width is specified as 3/16 bit. In infrared low-power mode, the pulse width can be configurable. And the ISDIV[7:0] bit can be used to achieve the desired low-power frequency.

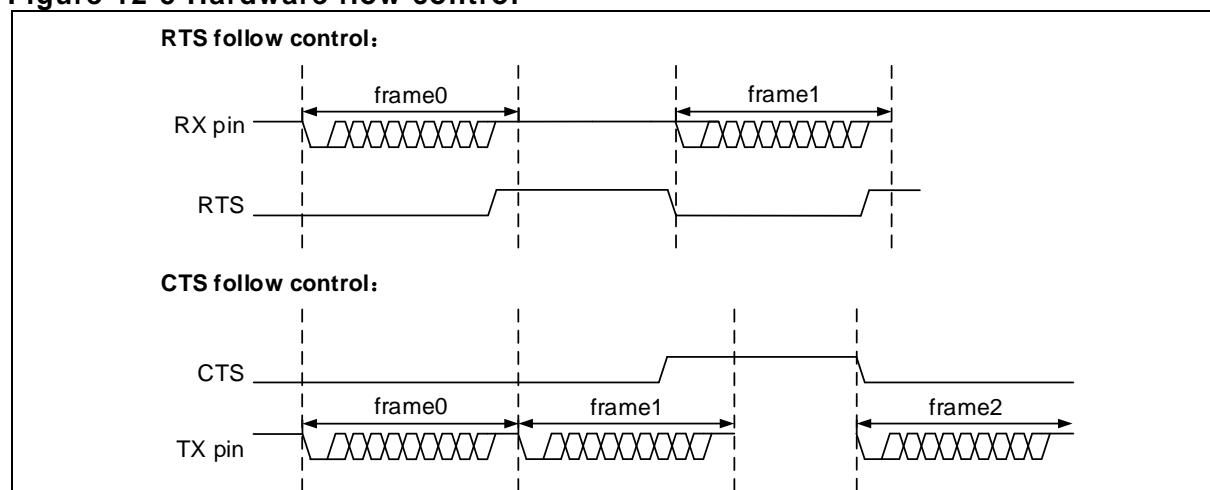
**Figure 12-4 IrDA DATA(3/16) – normal mode**

#### 4. Hardware flow control mode:

RTS and CTS flow control can be enabled by setting RTSEN=1 and CTSEN=1, respectively. This is to control serial data flow between two devices.

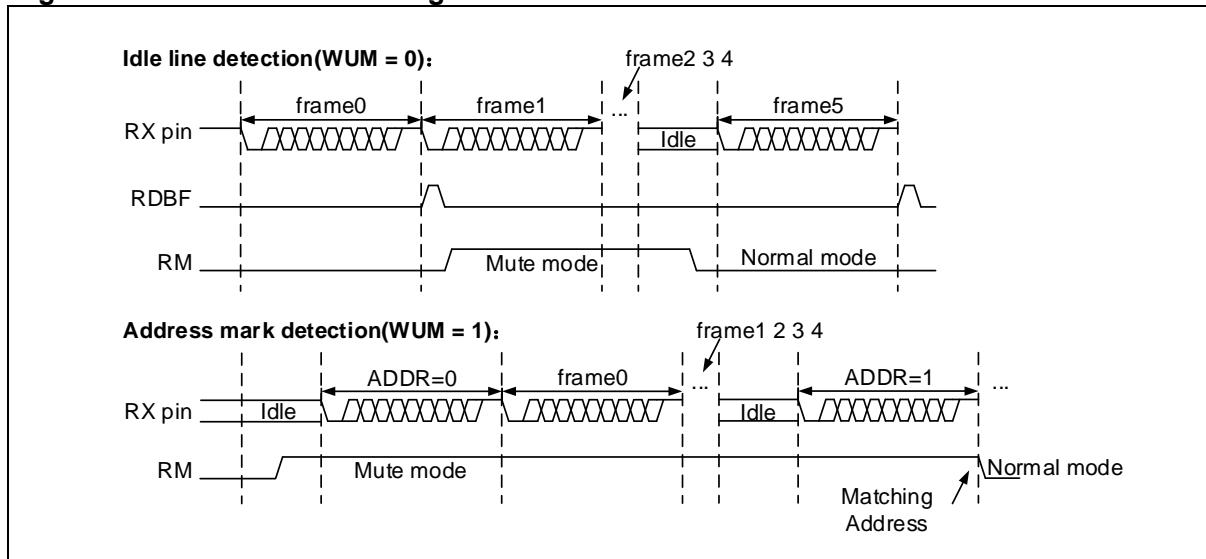
RTS: the RTS becomes active (pull-down means low) as soon as the USART receiver is ready to receive a data. When the data has arrived (starts at each STOP bit) in the receive register, the RTS bit is set, indicating request to stop data transfer at the end of current frame.

CTS: the USART transmitter checks the CTS input before sending next frame. The next data is sent if CTS is active (when low); if CTS becomes inactive (when high) during transmission, it stops sending at the end of current transfer.

**Figure 12-5 Hardware flow control**

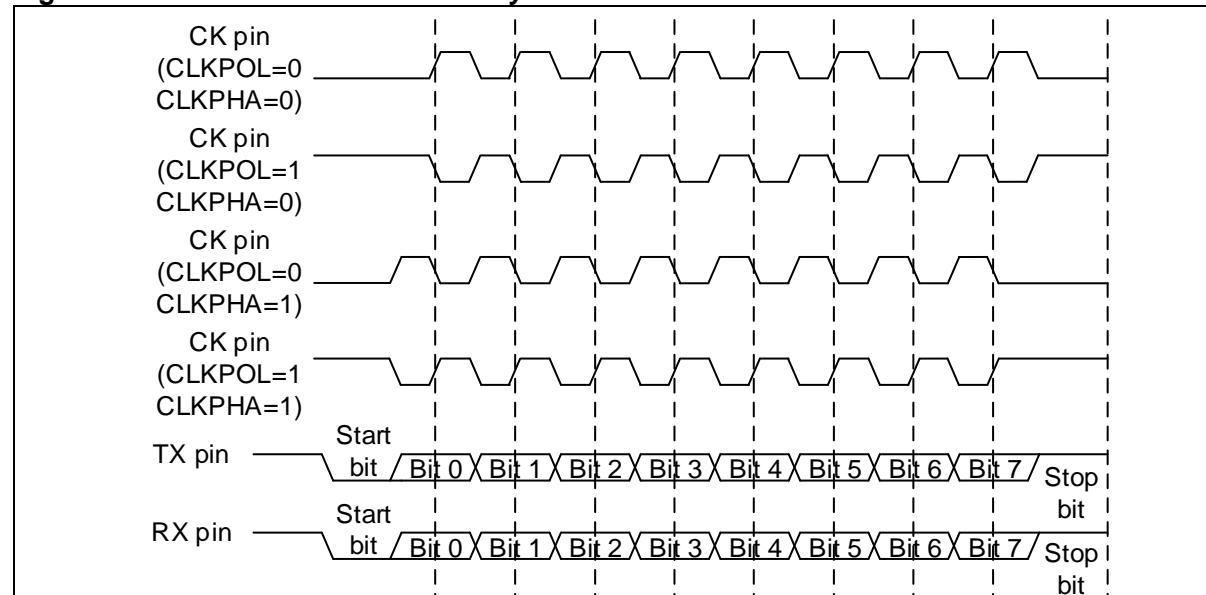
#### 5. Mute mode:

Silent mode can be entered by setting RM=1. It is possible to wake up from silent mode by setting WUM=1 (ID match) and WUM=0 (idle line), respectively. The ID[3: 0] is configurable. When ID match is selected, if the MSB of data bit is set to 1, it indicates that the current data is ID, and the four LSB represent ID value.

**Figure 12-6 Mute mode using Idle line or Address mark detection**

#### 6. Synchronous mode:

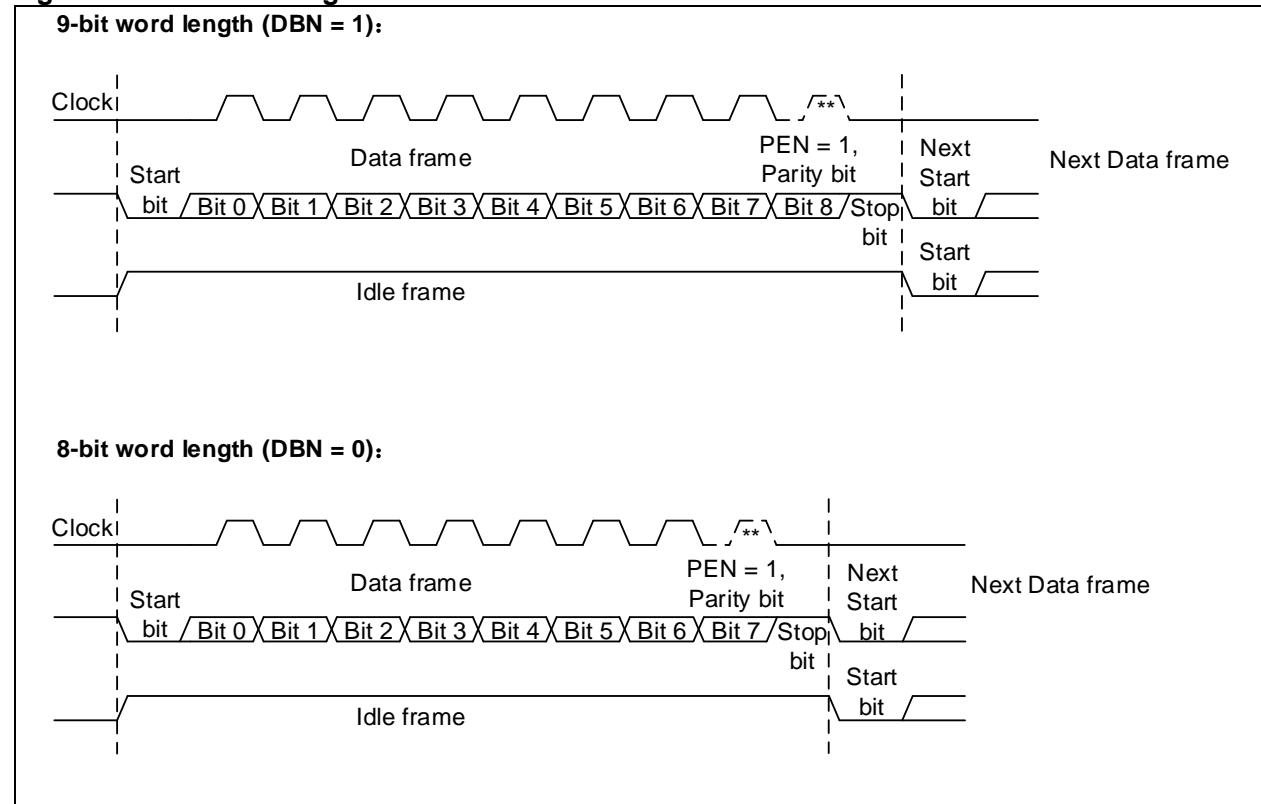
By setting the CLKEN bit to 1, synchronous mode and clock pin output are enabled. Select CK pin high or low in idle state by setting the CLKPOL bit (1 or 0). Whether to sample data on the second or the first edge of the clock depends on the CLKPHA bit (1 or 0). The LBCP bit (1 or 0) is used to select whether to output clock on the last data bit. And the ISDIV[4: 0] is used to select the required clock output frequency.

**Figure 12-7 8-bit format USART synchronous mode**

## 12.4 USART frame format and configuration

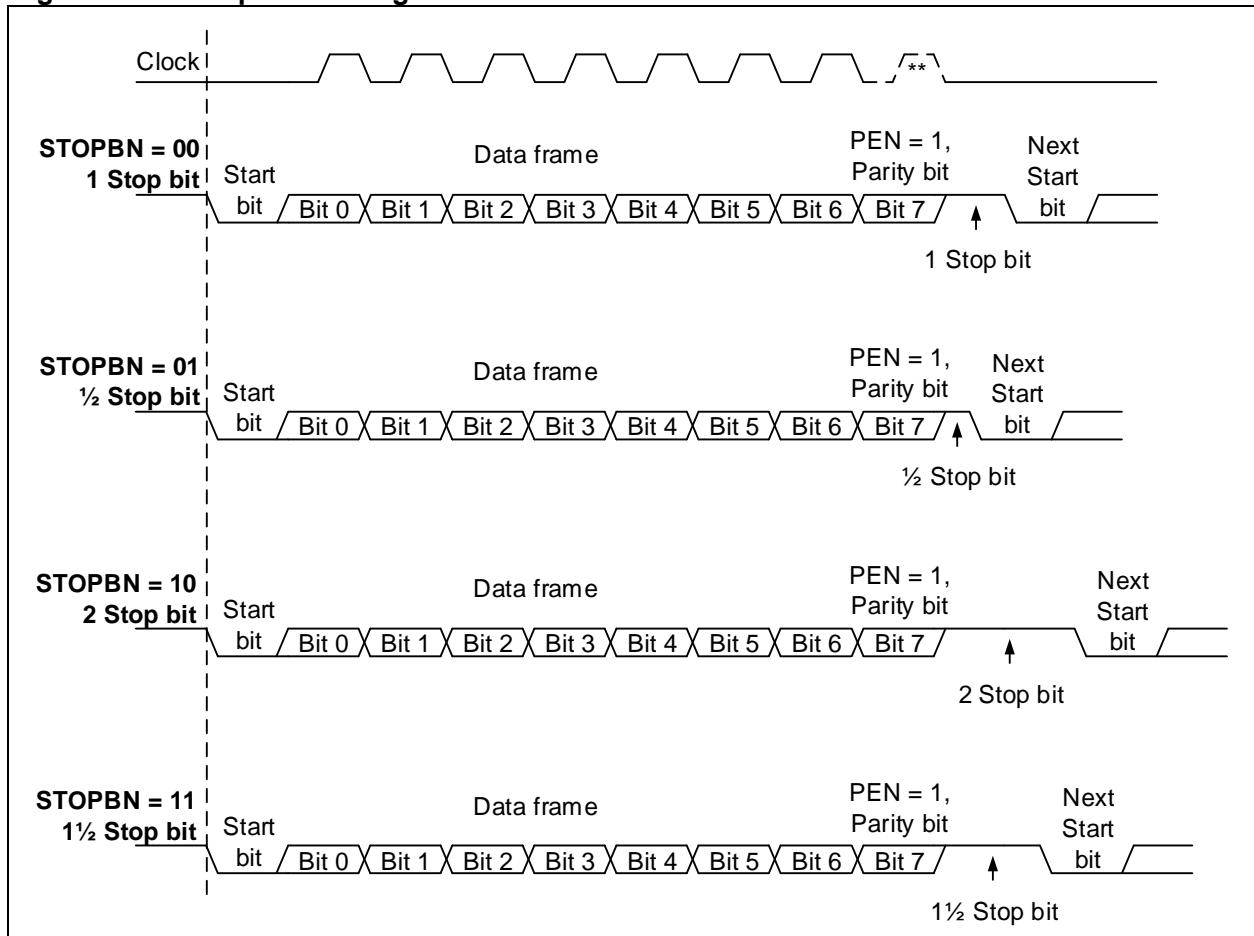
USART data frame consists of start bit, data bit and stop bit, with the last data bit being as a parity bit. USART idle frame size is equal to that of the data frame under current configuration, but all bits are 1. USART brake frame size is the current data frame size plus its stop bit. All bits before the stop bit are 0. The DBN bit is used to program 8-bit (DBN=0) or 9-bit (DBN=1) data bits.

**Figure 12-8 Word length**



The STOPBN bit is used to program one bit (STOPBN=00), 0.5-bit (STOPBN=01), 2-bit (STOPBN=10) and 1.5-bit (STOPBN=11) stop bits.

Set the PEN bit will enable parity control. PSEL=1 indicates Odd parity, while PSEL=0 for Even parity. Once the parity control is enabled, the MSB of the data bit will be replaced with parity bit, that is, the significant data bits are reduced by one bit.

**Figure 12-9 Stop bit configuration**

## 12.5 DMA transfer introduction

Enable transmit data buffer and receive data buffer using DMA to achieve continuous high-speed transmission for USART, which is detailed in subsequent sections. For more information on specific DMA configuration, refer to DMA chapter.

### 12.5.1 Transmission using DMA

1. Select a DMA channel: Select a DMA channel from DMA channel map table described in DMA chapter.
2. Configure the destination of DMA transfer: Configure the USART\_DT register address as the destination address bit of DMA transfer in the DMA control register. Data will be sent to this address after transmit request is received by DMA.
3. Configure the source of DMA transfer: Configure the memory address as the source of DMA transfer in the DMA control register. Data will be loaded into the USART\_DT register from the memory address after transmit request is received by DMA.
4. Configure the total number of bytes to be transferred in the DMA control register (TMRx\_DMACTRL).
5. Configure the channel priority of DMA transfer in the DMA control register (TMRx\_DMACTRL).
6. Configure DMA interrupt generation after half or full transfer in the DMA control register (TMRx\_DMACTRL).
7. Enable DMA transfer channel in the DMA control register (TMRx\_DMACTRL).

### 12.5.2 Reception using DMA

1. Select a DMA transfer channel: Select a DMA channel from DMA channel map table described in DMA chapter.
2. Configure the destination of DMA transfer: Configure the memory address as the destination of DMA

- transfer in the DMA control register. Data will be loaded from the USART\_DT register to the programmed destination after reception request is received by DMA.
3. Configure the source of DMA transfer: Configure the USART\_DT register address as the source of DMA transfer in the DMA control register. Data will be loaded from the USART\_DT register to the programmed destination after reception request is received by DMA.
  4. Configure the total number of bytes to be transferred in the DMA control register (TMRx\_DMACTRL).
  5. Configure the channel priority of DMA transfer in the DMA control register (TMRx\_DMACTRL).
  6. Configure DMA interrupt generation after half or full transfer in the DMA control register (TMRx\_DMACTRL).
  7. Enable a DMA transfer channel in the DMA control register (TMRx\_DMACTRL).

## 12.6 Baud rate generation

### 12.6.1 Introduction

USART baud rate generator uses an internal counter based on PCLK. The DIV (USART\_BAUDR [15:0] register) represents the overflow value of the counter. Each time the counter is full, it denotes one-bit data. Thus each data bit width refers to PCLK cycles x DIV.

The receiver and transmitter of USART share the same baud rate generator, and the receiver splits each data bit into 16 equal parts to achieve oversampling, so the data bit width should not be less than 16 PCLK periods, that is, the DIV value must be greater than 16.

### 12.6.2 Configuration

User can program the desired baud rate by setting different system clocks and writing different values into the USART\_BAUDR register. The calculation format is as follows:

$$\frac{\text{TX}}{\text{RX}} \text{baud rate} = \frac{f_{CK}}{\text{DIV}}$$

Where,  $f_{CK}$  refers to the system clock of USART (i.e. PCLK1/PCLK2)

*Note: 1. The USART\_BAUDR register must be written before UEN. The baud rate register value should not be altered when UEN=1.*

*2. Disabling USART receiver or transmitter will reset the internal counter, and generate a baud rate interrupt.*

**Table 12-1 Baud rate calculation error**

Baud rate fPCLK=36MHz			fPCLK=72MHz				
No.	Kbps	Actual	Value programmed in the baud register	Error %	Actual	Value programmed in the baud register	Error %
1	2.4	2.4	15000	0%	2.4	30000	0%
2	9.6	9.6	3750	0%	9.6	7500	0%
3	19.2	19.2	1875	0%	19.2	3750	0%
4	57.6	57.6	625	0%	57.6	1250	0%
5	115.2	115.384	312	0.15%	115.2	625	0%
6	230. 4	230.769	156	0.16%	230.769	312	0.16%
7	460. 8	461.538	78	0.16%	461.538	156	0.16%
8	921. 6	923.076	39	0.16%	923.076	78	0.16%
9	2250	2250	16	0%	2250	32	0%
10	4500	NA	NA	NA	4500	16	0%

Taking a baud rate of 115.2Kbps as an example, if fPCLK=36MHz, the value in the baud register should be set to 312(0x38). Based on formula, the calculated baud rate (actual) is  $36000000 / 312 = 115384 = 115.384$ Kbps. The % error between the desired and actual value is calculated based on the formula: (Calculated actual result-Desired)/desired baud rate\*100%, that is,  $(115.384 - 115.2) / 115.2 * 100\% = 0.15\%$ .

## 12.7 Transmitter

### 12.7.1 Transmitter introduction

USART transmitter has its individual TEN control bit. The transmitter and receiver share the same baud rate that is programmable. There is a transmit data buffer (TDR) and a transmit shift register in the USART. The TDBE bit is set whenever the TDR is empty, and an interrupt is generated if the TDBEIEN is set.

The data written by software is stored in the TDR register. When the shift register is empty, the data will be moved from the TDR register to the shift register so that the data in the transmit shift register is output on the TX pin in LSB mode. The output format depends on the programmed frame format.

If synchronous transfer or clock output is selected, the clock pulse is output on the CK pin. If the hardware flow control is selected, the control signal is input on the CTS pin.

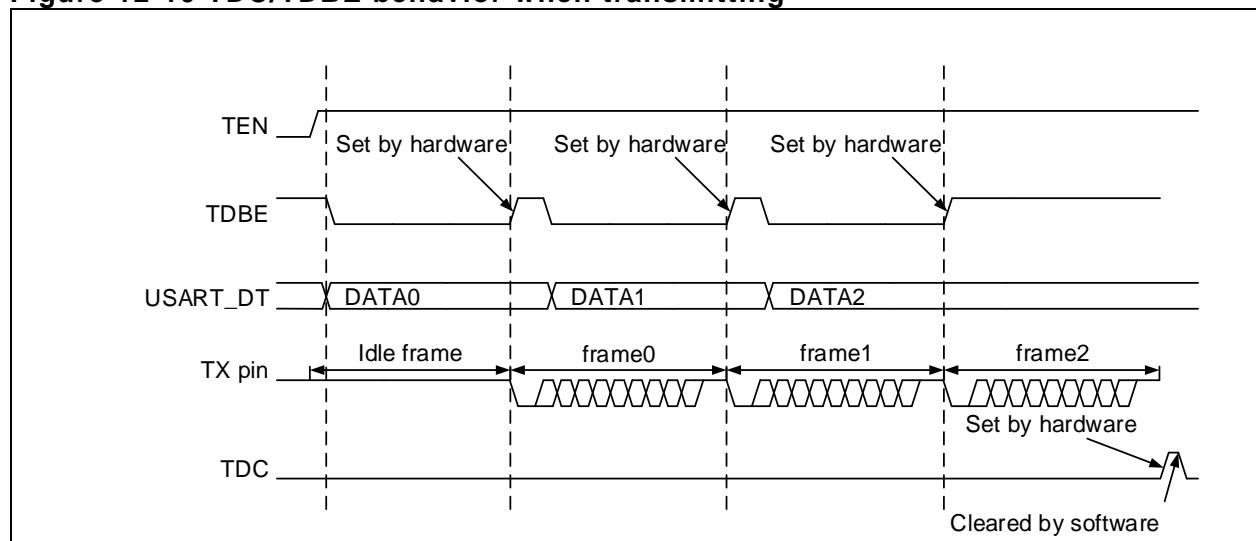
*Note: 1. The TEN bit cannot be reset during data transfer, or the data on the TX pin will be corrupted.*

*2. After the TEN bit is enabled, the USART will automatically send an idle frame.*

### 12.7.2 Transmitter configuration

1. USART enable: Set the UEN bit to 1.
2. Full-duplex/half-duplex configuration: Refer to [12.2 Full-duplex/half-duplex selector](#).
3. Mode configuration: Refer to [12.3 Mode selector](#).
4. Frame format configuration: Refer to [12.4 USART frame format and configuration](#).
5. Interrupt configuration: Refer to [12.10 Interrupt requests](#).
6. DMA transfer configuration: To use DMA mode for data transfer, the DMATEN bit (bit 7 in the USART\_CTRL3 register) should be set, and the DMA register should be configured accordingly.
7. Baud rate configuration: Refer to [12.6 Baud rate generation](#).
8. Transmitter enable: When the TEN bit is set to 1, the USART transmitter will automatically send an idle frame.
9. Write operation: Wait until the TDBE bit is set, and the data to be transferred are written into the USART\_DT register (This operation will clear the TDBE bit). Repeat this step in non-DMA mode.
10. After the last data expected to be transferred is written, wait until the TDC is set, indicating the end of transfer. The USART cannot be disabled before the flag is set, or transfer error will occur.
11. When TDC=1, reading USART\_STS register once and writing to the USART\_DT register once can clear the TDC bit; It can also be cleared by writing "0" to itself, but this is valid only in DMA mode.

**Figure 12-10 TDC/TDBE behavior when transmitting**



## 12.8 Receiver

### 12.8.1 Receiver introduction

USART receiver has its individual REN control bit (bit 2 in the USART\_CTRL1 register). The transmitter and receiver share the same baud rate that is programmable. There is a receive data buffer (RDR) and a receive shift register in the USART.

The data is input on the RX pin of the USART. When a valid start bit is detected, the receiver ports the data received into the receive shift register in LSB mode. After a full data frame is received, the value in the receive shift register is moved to the receive data buffer according to the programmed frame format, and the RDBF is set accordingly. An interrupt is generated if the RDBFIEN is set.

If hardware flow control is selected, the control signal is output on the RTS pin.

During data reception, the USART receiver will detect whether there are errors to occur, including framing error, overrun error, parity check error or noise error, depending on software configuration, and determine whether there are interrupts to generate according to the interrupt enable bits.

### 12.8.2 Receiver configuration

Configuration procedure:

1. USART enable: UEN bit is set.
2. Full-duplex/half-duplex configuration: Refer to [12.2 Full-duplex/half-duplex selector](#).
3. Mode configuration: Refer to [12.3 Mode selector](#).
4. Frame format configuration: Refer to [12.4 USART frame format and configuration](#).
5. Interrupt configuration: Refer to [12.10 Interrupt requests](#).
6. Reception using DMA: To use DMA mode for data reception, the DMAREN bit should be set, and the DMA register should be configured accordingly.
7. Baud rate configuration: Refer to [12.6 Baud rate generation](#).
8. Receiver enable: REN bit is set.

Character reception:

- The RDBF bit is set. It indicates that the content of the shift register is transferred to the RDR (Receiver Data Register). In other words, data is received and can be read (including its associated error flags)
- An interrupt is generated when the RDBFIEN is set.
- The error flag is set when a framing error, noise error or overrun error is detected during reception.
- In DMA mode, the RDNE bit is set after every byte is received, and it is cleared when the data register is read by DMA.
- In non-DMA mode, the RDBF bit is cleared by software reading the USART\_DT register. The RDBF flag can also be cleared by writing 0 to itself. The RDBF bit must be cleared before the end of next frame reception to avoid overrun error.

Brake frame reception:

- Non-LIN mode: It is handled as a framing error, and the FERR is set. An interrupt is generated if the corresponding interrupt bit is enabled. Refer to framing error described below for details.
- LIN mode: It is handled as a brake frame, and the BFF bit is set. An interrupt is generated if the BFLEN is set.

Idle frame reception:

- It is handled as a data frame, and the IDLEF bit is set. An interrupt is generated if the IDLEIEN is set.

When a framing error occurs:

- The FERR bit is set.
- The USART receiver moves the invalid data from the receive shift register to the receive data

buffer.

- In non-DMA mode, both FERR and RDBF are set at the same time. The latter will generate an interrupt. In DMA mode, an interrupt is generated if the ERRIEN.

When an overrun error occurs:

- The ROERR bit is set.
- The data in the receive data buffer is not lost. The previous data is still available when the USART\_DT register is read.
- The content in the receive shift register is overwritten. Afterwards, any data received will be lost.
- An interrupt is generated if the RDBFIEN is set or both ERRIEN and DMAREN are set.
- The ROERR bit is cleared by reading the USART\_STS register and then USART\_DT register.

*Note: If ROERR is set, it indicates that at least one piece of data is lost, with two possibilities:*

*If RDBF=1, it indicates that the last valid data is still stored in the receive data buffer, and can be read.*

*If RDBF=0, it indicates that the last valid data in the receive data buffer has already been read.*

*Note: The REN bit cannot be reset during data reception, or the byte that is currently being received will be lost.*

### 12.8.3 Start bit and noise detection

A start bit detection occurs when the REN bit is set. With the oversampling techniques, the USART receiver samples data on the 3rd, 5th, 7th, 8th, 9th and 10th bits to detect the valid start bit and noise.

**Table 12-2** shows the data sampling over start bit and noise detection.

**Table 12-2 Data sampling over start bit and noise detection**

Sampled value (3·5·7)	Sampled value (8·9·10)	NERR bit	Start bit validity
000	000	0	Valid
001/010/100	001/010/100	1	Valid
001/010/100	000	1	Valid
000	001/010/100	1	Valid
111/110/101/011	Any value	0	Invalid
Any value	111/110/101/011	0	Invalid

*Note: If the sampling values on the 3rd, 5th, 7th, 8th, 9th, and 10th bits do not match the above mentioned requirements, the USART receiver does not think that a correct start bit is received, and thus it will abort the start bit detection and return to idle state waiting for a falling edge.*

The USART receiver has the ability to detect noise. In the non-synchronous mode, the USART receiver samples data on the 7<sup>th</sup>, 8<sup>th</sup> and 9<sup>th</sup> bits, with its oversampling techniques, to distinguish valid data input from noise based on different sampling values, and recover data as well as set NERR flag (Noise Error Flag) bit.

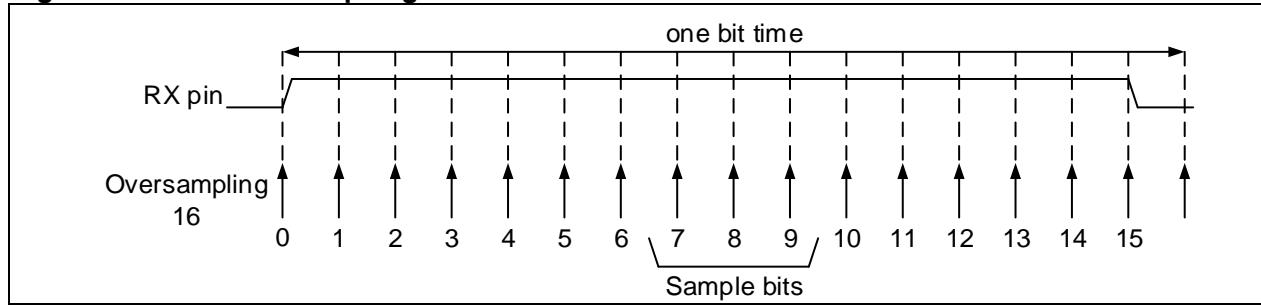
**Table 12-3 Data sampling over valid data and noise detection**

Sampled value	NERR bit	Received bit value	Data validity
000	0	0	Valid
001	1	0	Invalid
010	1	0	Invalid
011	1	1	Invalid
100	1	0	Invalid
101	1	1	Invalid
110	1	1	Invalid
111	0	1	Valid

When noise is detected in a data frame:

- The NERR bit is set at the same time as the RDBF bit
- The invalid data is transferred from the receive shift register to the receive data buffer.
- No interrupt is generated in non-DMA mode. However, since the NERR bit and RDBF bit are set simultaneously, the RDBF bit will generate an interrupt. In DMA mode, an interrupt will be generated if the ERRIEN is set.

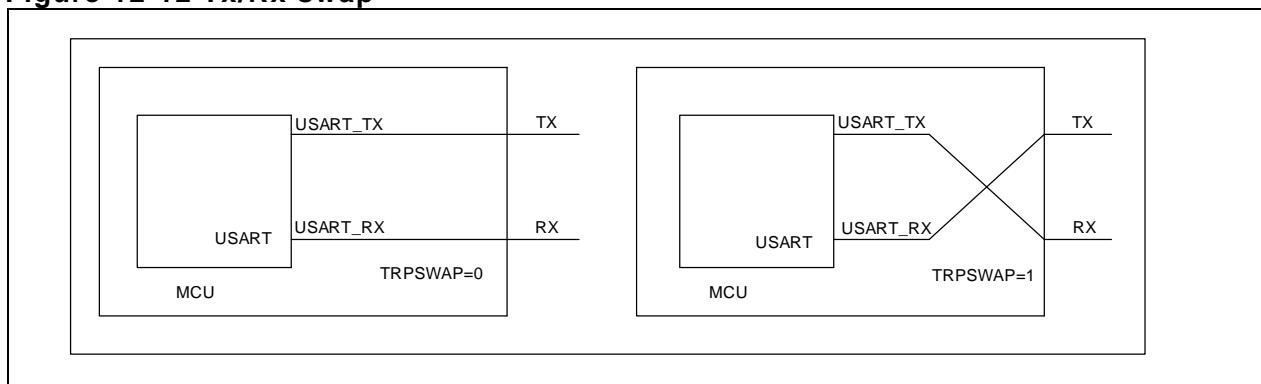
The NERR bit is cleared by reading USART\_STS register and then the USART\_DT register.

**Figure 12-11 Data sampling for noise detection**

## 12.9 Tx/Rx swap

When the TRPSWAP bit (USART\_CTRL2[15]) is set, Tx/Rx pin can be swapped. Two common scenes are listed below:

- If the Tx/Rx were reversed while the user attempts to connect the device externally to a RS-232 chip, it is possible to swap the Tx/Rx through the TRPSWAP bit, without the need of hardware intervention.
- If the user only connected the master Tx to the slave Rx in full-duplex mode, the Tx/Rx can be interchangeable with the TRPSWAP bit, after the master and slave are swapped, without the need of hardware intervention.

**Figure 12-12 Tx/Rx swap**

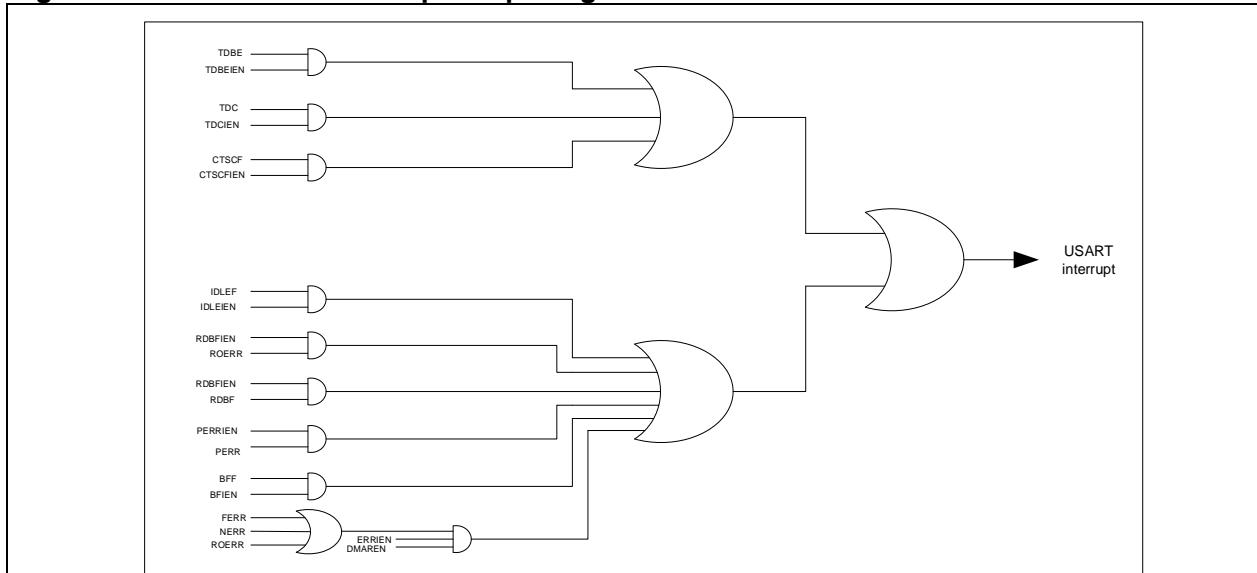
*Note: The SWAP (USART\_CTRL2[15]) can be modified only when the USART is disabled (UEN=0)*

## 12.10 Interrupt requests

USART interrupt generator serves as a control center of USART interrupts. It is used to monitor the interrupt source inside the USART in real time, and to define the generation of interrupts by configuring the corresponding interrupt enable bits. Table 12-4 shows the USART interrupt sources and their interrupt enable bits. An interrupt will be generated over an event when the corresponding interrupt enable bit is set.

**Table 12-4 USART interrupt request**

Interrupt event	Event flag	Enable bit
Transmit data register empty	TDBE	TDBEIEN
CTS flag	CTSCF	CTSCFIEN
Transmit data complete	TDC	TDCIEN
Receive data buffer full	RDBF	RDBFIEN
Receiver overflow error	ROERR	
Idle flag	IDLEF	IDLEIEN
Parity error	PERR	PERRIEN
Brake frame flag	BFF	BFIEN
Noise error, overflow error or framing error	NERR or ROERR or FERR	ERRIEN <sup>(1)</sup>

**Figure 12-13 USART interrupt map diagram**

## 12.11 I/O pin control

The following five interfaces are used for USART communication.

RX: Serial data input.

TX: Serial data output. In single-wire half-duplex and Smartcard mode, the TX pin is used as an I/O for data transmission and reception.

CK: Transmitter clock output. The output CLK phase, polarity and frequency are programmable.

CTS: Transmitter input. Send enable signal in hardware flow control mode.

RTS: Receiver output. Send request signal in hardware flow control mode.

## 12.12 USART registers

These peripheral registers must be accessed by words (32 bits).

**Table 12-5 USART register map and reset value**

Register	Offset	Reset value
USART_STS	0x00	0x0000 00C0
USART_DT	0x04	0x0000 0000
USART_BAUDR	0x08	0x0000 0000
USART_CTRL1	0x0C	0x0000 0000
USART_CTRL2	0x10	0x0000 0000
USART_CTRL3	0x14	0x0000 0000
USART_GDIV	0x18	0x0000 0000

## 12.12.1 Status register (USART\_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Forced 0 by hardware.
Bit 9	CTSCF	0x0	rw0c	<p>CTS change flag This bit is set by hardware when the CTS status line changes. It is cleared by software. 0: No change on the CTS status line 1: A change occurs on the CTS status line.</p>
Bit 8	BFF	0x0	rw0c	<p>Brake frame flag This bit is set by hardware when a brake frame is detected. It is cleared by software. 0: Brake frame is not detected. 1: Brake frame is detected.</p>
Bit 7	TDBE	0x1	ro	<p>Transmit data buffer empty This bit is set by hardware when the transmit data buffer is empty. It is cleared by a USART_DT register write operation. 0: Data is not transferred to the shift register. 1: Data is transferred to the shift register.</p>
Bit 6	TDC	0x1	rw0c	<p>Transmit data complete This bit is set by hardware at the end of transmission. It is cleared by software. (Option 1: read access to USART_STS register followed by a USART_DT write operation; Option 2: Write "0" to this bit) 0: Transmission is not completed. 1: Transmission is completed.</p>
Bit 5	RDBF	0x0	rw0c	<p>Receive data buffer full This bit is set by hardware when the data is transferred from the shift register to the USART_DT register. It is cleared by software. (Option 1: read USART_DT register; Option 2: write "0" to this bit) 0: Data is not received. 1: Data is received.</p>
Bit 4	IDLEF	0x0	ro	<p>Idle flag This bit is set by hardware when an idle line is detected. It is cleared by software. (Read USART_DT register followed by a USART_DT read operation) 0: No idle line is detected. 1: Idle line is detected.</p>
Bit 3	ROERR	0x0	ro	<p>Receiver overflow error This bit is set by hardware when the data is received while the RDNE is still set. It is cleared by software. (Read USART_STS register followed by a USART_DT read operation) 0: No overflow error 1: Overflow error is detected. Note: When this bit is set, the DT register content will not be lost, but the subsequent data will be overwritten.</p>
Bit 2	NERR	0x0	ro	<p>Noise error This bit is set by hardware when noise is detect on a received frame. It is cleared by software. (Read USART_STS register followed by a USART_DT read operation) 0: No noise is detected. 1: Noise is detected.</p>
Bit 1	FERR	0x0	ro	<p>Framing error This bit is set by hardware when a stop bit error (low), excessive noise or brake frame is detected. It is cleared by software. (Read USART_STS register followed by a USART_DT read operation) 0: No framing error is detected. 1: Framing error is detected.</p>
Bit 0	PERR	0x0	ro	Parity error

This bit is set by hardware when parity error occurs. It is cleared by software. USART\_STS register followed by a USART\_DT read operation)  
 0: No parity error occurs.  
 1: Parity error occurs.

## 12.12.2 Data register (USART\_DT)

Bit	Register	Reset value	Type	Description
Bit 31: 9	Reserved	0x000000	resd	Forced 0 by hardware.
Bit 8: 0	DT	0x00	rw	Data value This register provides read and write function. When transmitting with the parity bit enabled, the value written in the MSB bit will be replaced by the parity bit. When receiving with the parity bit enabled, the value in the MSB bit is the received parity bit.

## 12.12.3 Baud rate register (USART\_BAUDR)

*Note: If the TEN and REN bits are both disabled, the baud counter stops counting.*

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Forced 0 by hardware.
Bit 15: 0	DIV	0x0000	rw	Divider This field define the USART divider.

## 12.12.4 Control register1 (USART\_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 31: 14	Reserved	0x000000	resd	Forced 0 by hardware.
Bit 13	UEN	0x0	rw	USART enable 0: USART is disabled. 1: USART is enable.
Bit 12	DBN	0x0	rw	Data bit num This bit is used to program the number of data bits. 0: 8 data bits 1: 9 data bits
Bit 11	WUM	0x0	rw	Wakeup mode This bit determines the way to wake up silent mode. 0: Waken up by idle line 1: Waken up by ID match
Bit 10	PEN	0x0	rw	Parity enable This bit is used to enable hardware parity control (generation of parity bit for transmission; detection of parity bit for reception). When this bit is enabled, the MSB bit of the transmitted data is replaced with the parity bit; Check whether the parity bit of the received data is correct. 0: Parity control is disabled. 1: Parity control is enabled.
Bit 9	PSEL	0x0	rw	Parity selection This bit selects the odd or even parity after the parity control is enabled. 0: Even parity 1: Odd parity
Bit 8	PERRIEN	0x0	rw	PERR interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 7	TDBEIEN	0x0	rw	TDBE interrupt enable 0: Interrupt is disabled.

				1: Interrupt is enabled.
				TDC interrupt enable
Bit 6	TDCIEN	0x0	rw	0: Interrupt is disabled. 1: Interrupt is enabled.
				RDBF interrupt enable
Bit 5	RDBFIEN	0x0	rw	0: Interrupt is disabled. 1: Interrupt is enabled.
				IDLE interrupt enable
Bit 4	IDLEIEN	0x0	rw	0: Interrupt is disabled. 1: Interrupt is enabled.
				Transmitter enable
Bit 3	TEN	0x0	rw	This bit enables the transmitter. 0: Transmitter is disabled. 1: Transmitter is enabled.
				Receiver enable
Bit 2	REN	0x0	rw	This bit enables the receiver. 0: Receiver is disabled. 1: Receiver is enabled.
				Receiver mute
Bit 1	RM	0x0	rw	This bit determines if the receiver is in mute mode or not. It is set or cleared by software. When the idle line is used to wake up from mute mode, this bit is cleared by hardware after wake up. When the address match is used to wake up from mute mode, it is cleared by hardware after wake up. When address mismatches, this bit is set by hardware to enter mute mode again. 0: Receiver is in active mode. 1: Receiver is in mute mode.
				Send brake frame
Bit 0	SBF	0x0	rw	This bit is used to send a brake frame. It can be set or cleared by software. Generally speaking, it is set by software and cleared by hardware at the end of brake frame transmission. 0: No brake frame is transmitted. 1: Brake frame is transmitted.

## 12.12.5 Control register2 (USART\_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x00000	resd	Forced 0 by hardware.
				Transmit/receive pin swap
Bit 15	TRPSWAP	0x0	rw	0: Transmit/receive pin swap disabled 1: Transmit/receive pin swap enabled
				LIN mode enable
Bit 14	LINEN	0x0	rw	0: LIN mode is disabled. 1: LIN mode is enabled.
				STOP bit num
Bit 13: 12	STOPBN	0x0	rw	These bits are used to program the number of stop bits. 00: 1 stop bit 01: 0.5 stop bit 10: 2 stop bits 11: 1.5 stop bits
				Clock enable
Bit 11	CLKEN	0x0	rw	This bit is used to enable the clock pin for synchronous mode or Smartcard mode. 0: Clock is disabled.

				1: Clock is enabled.
				Clock polarity
Bit 10	CLKPOL	0x0	rw	In synchronous mode or Smartcard mode, this bit is used to select the polarity of the clock output on the clock pin in idle state. 0: Clock output low 1: Clock output high
				Clock phase
Bit 9	CLKPHA	0x0	rw	This bit is used to select the phase of the clock output on the clock pin in synchronous mode or Smartcard mode. 0: Data capture is done on the first clock edge. 1: Data capture is done on the second clock edge.
				Last bit clock pulse
Bit 8	LBCP	0x0	rw	This bit is used to select whether the clock pulse of the last data bit transmitted is output on the clock pin in synchronous mode. 0: The clock pulse of the last data bit is no output on the clock pin. 1: The clock pulse of the last data bit is output on the clock pin.
Bit 7	Reserved	0x0	resd	Keep at its default value.
Bit 6	BFIEN	0x0	rw	Brake frame interrupt enable 0: Disabled 1: Enabled
Bit 5	BFBN	0x0	rw	Brake frame bit num This bit is used to select 11-bit or 10-bit brake frame. 0: 10-bit brake frame 1: 11-bit brake frame
Bit 4	Reserved	0x0	resd	Keep at its default value.
Bit 3: 0	ID	0x0	rw	USART identification Configurable USART ID.

*Note: These three bits (CLKPOL, CLKPHA and LBCP) cannot be changed while the transmission is enabled.*

## 12.12.6 Control register3 (USART\_CTRL3)

Bit	Register	Reset value	Type	Description
Bit 31: 11	Reserved	0x000000	resd	Forced 0 by hardware.
Bit 10	CTSCFIEN	0x0	rw	CTSCF interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 9	CTSEN	0x0	rw	CTS enable 0: CTS is disabled. 1: CTS is enabled.
Bit 8	RTSEN	0x0	rw	RTS enable 0: RTS is disabled. 1: RTS is enabled.
Bit 7	DMATEN	0x0	rw	DMA transmitter enable 0: DMA transmitter is disabled. 1: DMA transmitter is enabled.
Bit 6	DMAREN	0x0	rw	DMA receiver enable 0: DMA receiver is disabled. 1: DMA receiver is enabled.
Bit 5	SCMEN	0x0	rw	Smartcard mode enable 0: Smartcard mode is disabled.

				1: Smartcard mode is enabled.
				Smartcard NACK enable
Bit 4	SCNACKEN	0x0	rw	This bit is used to send NACK when parity error occurs. 0: NACK is disabled when parity error occurs. 1: NACK is enabled when parity error occurs.
Bit 3	SLBEN	0x0	rw	Single-wire bidirectional half-duplex enable 0: Single-wire bidirectional half-duplex is disabled. 1: Single-wire bidirectional half-duplex is enabled.
Bit 2	IRDALP	0x0	rw	IrDA low-power mode This bit is used to configure IrDA low-power mode. 0: IrDA low-power mode is disabled. 1: IrDA low-power mode is enabled.
Bit 1	IRDAEN	0x0	rw	IrDA enable 0: IrDA is disabled. 1: IrDA is enabled.
Bit 0	ERRIEN	0x0	rw	Error interrupt enable An interrupt is generated when a framing error, overflow error or noise error occurs. 0: Error interrupt is disabled. 1: Error interrupt is enabled.

## 12.12.7 Guard time and divider register (USART\_GDIV)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Forced 0 by hardware.
Bit 15: 8	SCGT	0x00	rw	Smartcard guard time value This field specifies the guard time value. The transmission complete flag is set after this guard time in smartcard mode.
Bit 7: 0	ISDIV	0x00	rw	IrDA/smartcard division In IrDA mode: 8 bit [7: 0] is valid. It is valid in common mode and must be set to 00000001. In low-power mode, it divides the peripheral clock to serve as the period base of the pulse width; 00000000: Reserved–Do not write. 00000001: Divided by 1 00000010: Divided by 2 ..... Smartcard mode: The lower 5 bit [4: 0] is valid. This division is used to divide the peripheral clock to provide clock for the Smartcard. Configured as follows: 00000: Reserved–Do not write. 00001: Divided by 2 00010: Divided by 4 00011: Divided by 6 .....

# 13 Serial peripheral interface (SPI)

## 13.1 SPI introduction

The SPI interface supports either the SPI protocol or the I<sup>2</sup>S protocol, depending on software configuration. This chapter gives an introduction of the main features and configuration procedure of SPI used as SPI and I<sup>2</sup>S respectively.

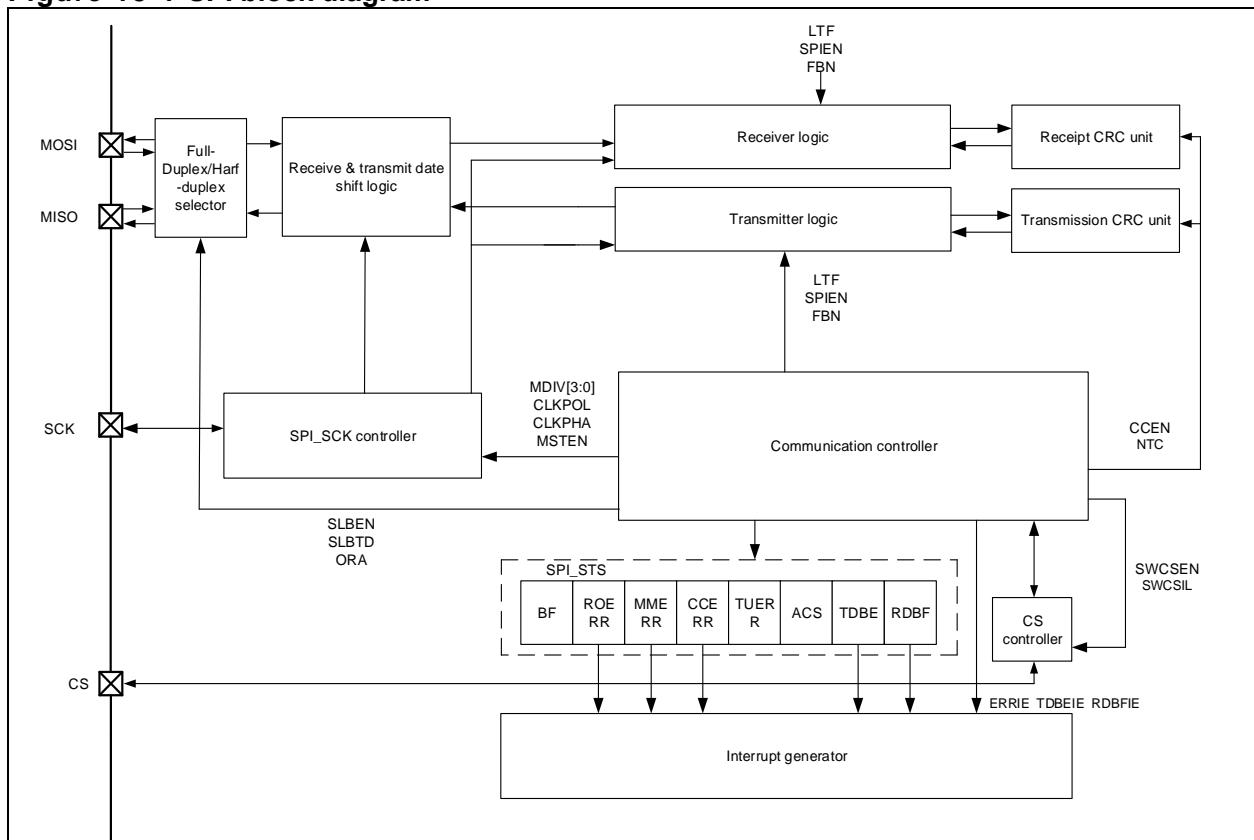
## 13.2 Functional overview

### 13.2.1 SPI description

The SPI can be configured as host or slave based on software configuration, supporting full-duplex, reception-only full-duplex and transmission-only/reception-only half-duplex modes, DMA transfer, and automatic CRC function of SPI internal hardware.

#### SPI block diagram:

**Figure 13-1 SPI block diagram**



#### Main features as SPI:

- Full-duplex or half-duplex communication
  - Full-duplex synchronous communication (supporting reception-only mode to release IO for transmission)
  - Half-duplex synchronous communication (transfer direction is configurable: receive or transmit)
- Master or slave mode
- CS signal processing mode
  - CS signal processing by hardware
  - CS signal processing by software
- 8-bit or 16-bit frame format
- Communication frequency and prescalers (Frequency division factor up to f<sub>PCLK</sub>/2)
- Programmable clock polarity and phase

- Programmable data transfer sequence (MSB-first or LSB-first)
- Programmable error interrupt flags (receiver overflow error, master mode error and CRC error)
- Programmable transmit data buffer empty interrupt and receive data buffer full interrupt
- Support transmission and reception using DMA
- Support hardware CRC transmission and error checking
- Busy status flag

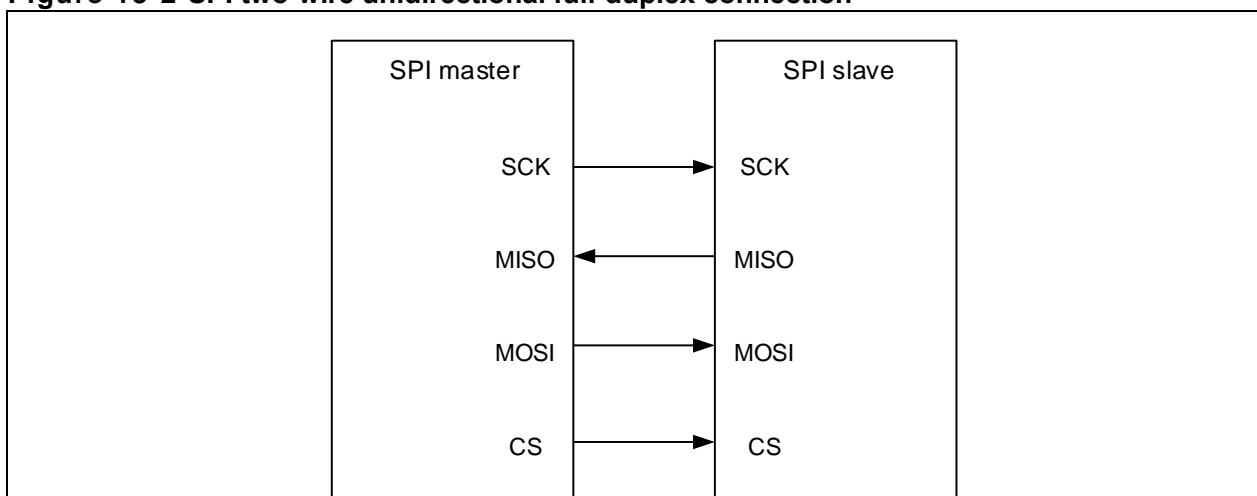
### 13.2.2 Full-duplex/half-duplex selector

When used as an SPI interface, it supports four synchronous modes: two-wire unidirectional full-duplex, single-wire unidirectional receive only, single-wire bidirectional half-duplex transmit and single-wire bidirectional half-duplex receive.

**Figure 13-2 shows the two-wire unidirectional full-duplex mode and SPI IO connection:**

The SPI operates in two-wire unidirectional full-duplex mode when the SLBEN bit and the ORA bit are both 0. In this case, the SPI supports data transmission and reception simultaneously. IO connection is as follows:

**Figure 13-2 SPI two-wire unidirectional full-duplex connection**



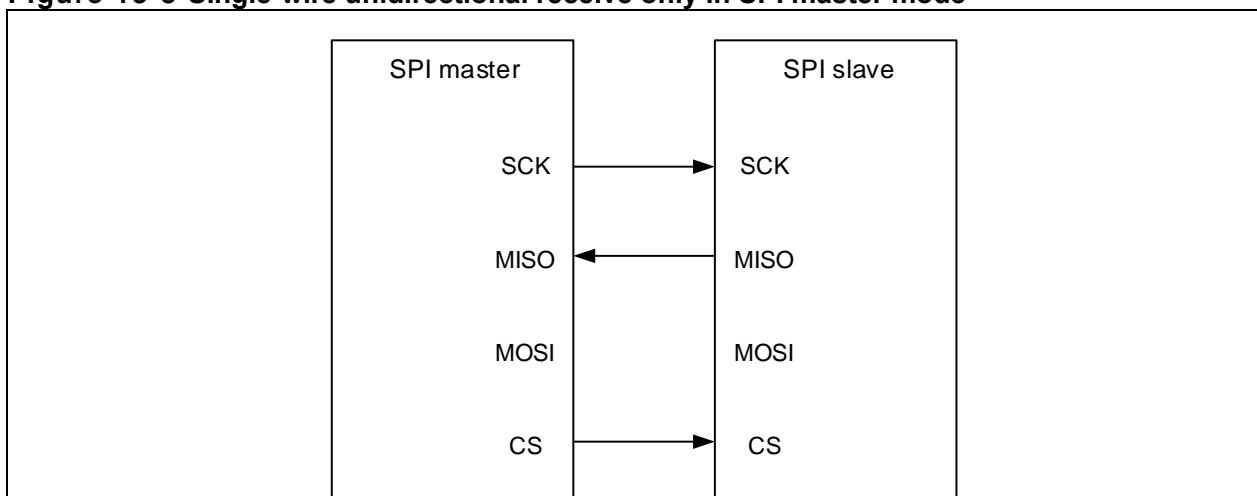
In either master or slave mode, it is necessary to wait until the RDBF bit and TDBE bit is set, as well as BF=0 before disabling SPI or entering power-saving mode (or gating SPI system clock).

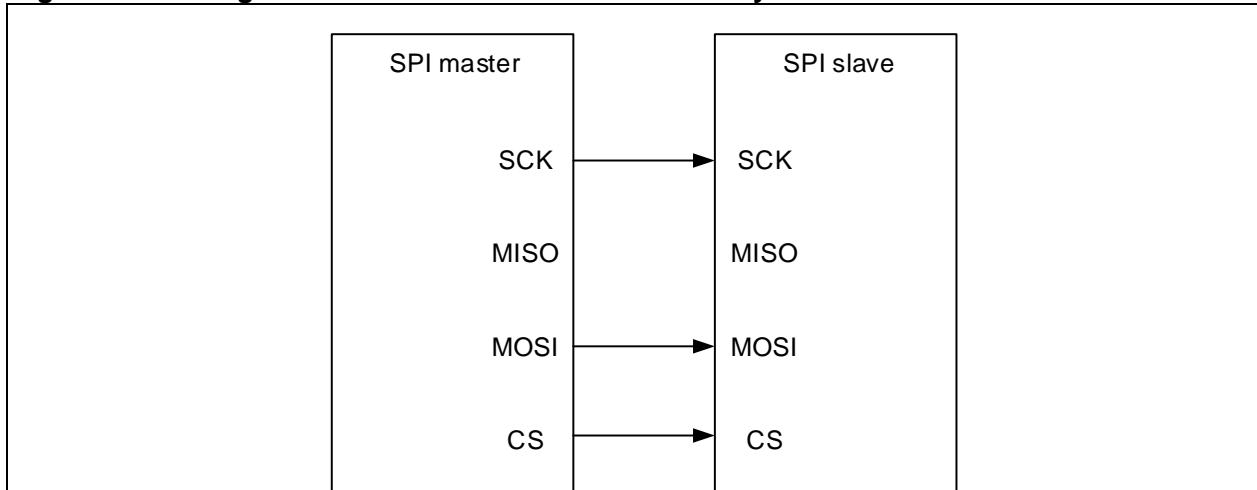
**Figure 13-3 shows the single-wire unidirectional receive-only mode and SPI IO connection**

The SPI operates in single-wire unidirectional receive-only mode when the SLBEN is 0 and the ORA is set to 1. In this case, the SPI can be used only for data reception (transmission is not supported).

In master mode, MISO pin receives data, and the IO mapped by MOSI is released; In slave mode, MOSI receive data, and the IO mapped by MISO is released.

**Figure 13-3 Single-wire unidirectional receive only in SPI master mode**



**Figure 13-4 Single-wire unidirectional receive only in SPI slave mode**

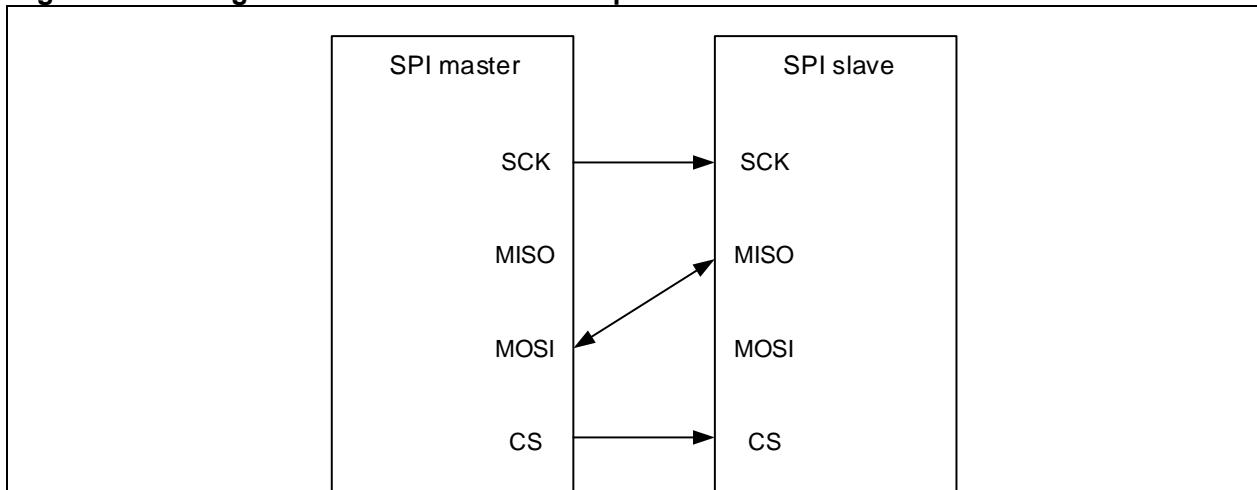
In SPI master mode, it is necessary to wait until the second-to-last RDBF bit is set and then wait another SPI\_CPK cycle before disabling SPI. The last RDBF must be set before entering power-saving mode (or gating SPI system clock).

In slave mode, there is no need to check any flag before disabling SPI. However, it is mandatory to wait until the BF becomes 0 before entering power-saving mode.

#### **Figure 13-5 shows single-wire bidirectional half-duplex mode and SPI IO connection**

When the SLBEN is set, the SPI operates in single-wire bidirectional half-duplex mode. In this case, the SPI supports data reception and transmission alternately. In master mode, the MOSI pin transmits or receives data, while the IO mapped by MISO is released. In slave mode, the MISO pin transmits or receives data, but the IO pin mapped by MOSI is released.

The SLBDT bit is used by software to configure transfer direction. When the SLBDT bit is set, the SPI can be used only for data transmission; when the SLBDT bit is 0, the SPI can be used only for data reception.

**Figure 13-5 Single-wire bidirectional half-duplex mode**

When the SPI is used for data transmission in single-wire bidirectional half-duplex mode (master or slave), the TDBE bit must be set, and the BF must be 0 before disabling SPI. The power-saving mode (or gating SPI system clock) cannot be entered unless the SPI is disabled.

In master mode, when the SPI is used for data reception in single-wire bidirectional half-duplex mode, it is necessary to wait until the second-to-last RDBF is set and then wait another SPI\_SCK cycle before disabling SPI. And the last RDBF must be set before entering power-saving mode (or gating SPI system clock).

In slave mode, when the SPI is used for data reception in single-wire bidirectional half-duplex mode, there is no need to check any flags before disabling SPI. However, the BT must be 0 before entering power-saving mode (or disabling SPI system clock).

### 13.2.3 Chip select controller

The Chip select controller (CS) is used to enable hardware or software control for chip select signals through software configuration. This controller is used to select master/slave device in multi-processor mode, and to avoid conflicts on the data lines by first enabling the SCK signal and then CS signal. The hardware and software configuration procedure is detailed as follows, along with their respective input/output in master and slave mode.

#### CS hardware configuration procedure:

In master mode with CS being as an output, HWCSOE=1, SWCSEN=0, the CS hardware control is enabled. If the SPI is enabled, low level is output on the CS pin. The CS signal is then released after the SPI is disabled and the transmission is complete.

In master mode with CS being as an input, HWCSOE=0, SWCSEN=0, the CS hardware control is enabled. At this point, the SPI is automatically disabled by hardware and enters slave mode as soon as the CS pin low is detected by master SPI. The mode error flag (MMERR bit) is set at the same time. An interrupt is generated if ERRIE=1. When the MMERR is set, the SPIEN and MSTEN cannot be set by software. The MMERR is cleared by read or write access to the SPI\_STS register followed by write operation to the SPI\_CTRL1 register.

In slave mode with CS being as an input, HWCSOE=0, SWCSEN=0, the CS hardware control is enabled. The slave selects whether to transmit / receive data based on the level on the CS pin. The slave is selected for data reception and transmission only when the CS pin is low.

#### CS software configuration procedure:

In master mode with CS being as an input, SWCSEN=1, the CS software control is enabled. When SWCSIL=0, the SPI is automatically disabled by hardware and enters slave mode. The mode error flag (MMERR bit) is set at this time. An interrupt is generated if ERRIE=1. When the MMERR bit is set, the SPIEN and MSTEN bits cannot be set by software. The MMERR bit is cleared by read or write access to the SPI\_STS register followed by write operation to the SPI\_CTRL1 register.

In slave mode with CS being as an input, SWCSEN=1, the CS software control is enabled. The SPI judges the CS signal with the SWCSIL bit, instead of CS pin. When SWCSIL=0, the slave is selected for data reception and transmission.

### 13.2.4 SPI\_SCK controller

The SPI protocol adopts synchronous transmission. In master mode and with the SPI being used as SPI, it is required to generate a communication clock for data reception and transmission on the SPI, and the communication clock should be output to the slave via IO for data reception and transmission. In slave mode, the communication clock is provided by peripherals, and is input to the SPI via IO. In all, the SPI\_SCK controller is used for the generation and distribution of SPI\_SCK, with the configuration procedure detailed as follows:

#### SPI\_SCK controller configuration procedure:

- Clock polarity and clock phase selection: It is selected by setting the CLKPOL and CLKPHA bit.
- Clock prescaler selection: Select the desired PCLK frequency by setting the CRM bit. Select the desired prescaler by setting the MDIV[3: 0] bit.
- Master/slave selection: Select SPI as master or slave by setting the MSTEN bit.

*Note that the clock output is activated after the SPI is enabled in master reception-only mode, and it remains output until when the SPI is disabled and the reception is complete.*

### 13.2.5 CRC

There is an independent transmission and reception CRC calculation unit in the SPI. When used as SPI through software configuration, the SPI enables CRC calculation and CRC check automatically while the user is reading or writing through DMA or CPU. During the transmission, if the received data is not consistent with, detected by hardware, the data in the SPI\_RCRC register, and such data is exactly the CRC value, then the CCERR bit will be set. An interrupt is generated if ERRIE=1.

The CRC function and configuration procedure of the SPI are described as follows.

#### CRC configuration procedure

- CRC calculation polynomial is configured by setting the SPI\_CPOLY register.
- CRC enable: The CRC calculation is enabled by setting the CCEN bit. This operation will reset the SPI\_RCRC and SPI\_TCRC registers.
- Select if or when the NTC bit is set, depending on DMA or CPU data register. See the following descriptions.

#### Transmission using DMA

When DMA is used to write the data to be transmitted, if the CCEN bit is enabled, the hardware calculates the CRC value automatically according to the value in the SPI\_CPOLY register and each transmitted data, and sends the CRC value at the end of the last data transmission. This result is regarded as the value of the SPI\_TCRC register.

#### Reception using DMA

When DMA is used to read the data to be received, if the CCEN bit is enabled, the hardware calculates the CRC value automatically according to the value in the SPI\_CPOLY register and each received data, and waits until the completion of CRC data reception at the end of the last data reception before comparing the received CRC value with the value of the SPI\_RCRC register. If check error occurs, the CCERR flag is set. An interrupt is generated if the ERRIE bit is enabled.

#### Transmission using CPU

Unlike DMA mode, after writing the last data to be transmitted, the CPU mode requires the NTC bit to be set by software before the end of the last data transmission.

#### Reception using CPU

In two-wire unidirectional full-duplex mode, follow CPU transmission mode to operate the NTC bit, the CRC calculation and check in CPU reception mode will be completed automatically.

In single-wire unidirectional reception-only mode and single-wire bidirectional reception-only mode, it is required to set the NTC bit before the software receives the last data when the second-to-last data is received.

### 13.2.6 DMA transfer

The SPI supports write and read operations with DMA. Refer to the following configuration procedure. Special attention should be paid to: when the CRC calculation and check is enabled, the number of data transferred by DMA is configured as the number of the data to be transferred. The number of data read with DMA is configured as the number of the data to be received. In this case, the hardware will send CRC automatically at the end of full transfer, and the receiver will also perform CRC check. Note that the received CRC data will be moved into the SPI\_DT register by hardware, with the RDBF being set, and the DMA read request will be sent if then DAM transfer is enabled. Hence, it is recommended to read the SPI\_DT register to get the CRC value at the end of CRC reception in order to avoid the upcoming transfer error.

#### Transmission with DMA

- Select DMA channel: Select a DMA channel for the current SPI from DMA channel map table described in DMA chapter.
- Configure the destination of DMA transfer: Configure the SPI\_DT register address as the destination address bit of DMA transfer in the DMA control register. Data will be sent to this address after transmit request is received by DMA.
- Configure the source of DMA transfer: Configure the memory address as the source of DMA transfer in the DMA control register. Data will be loaded into the SPI\_DT register from the memory address after transmit request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the channel priority of DMA transfer in the DMA control register.
- Configure DMA interrupt generation after half or full transfer in the DMA control register.
- Enable DMA transfer channel in the DMA control register.

#### Reception with DMA

- Select DMA transfer channel: Select a DMA channel for the current SPI from DMA channel map table described in DMA chapter.

- Configure the destination of DMA transfer: Configure the memory address as the destination of DMA transfer in the DMA control register. Data will be loaded from the SPI\_DT register to the programmed destination after reception request is received by DMA.
- Configure the source of DMA transfer: Configure the SPI\_DT register address as the source of DMA transfer in the DMA control register. Data will be loaded from the SPI\_DT register to the programmed destination after reception request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure DMA interrupt generation after half or full transfer in the DMA control register
- Enable DMA transfer channel in the DMA control register.

### 13.2.7 Transmitter

The SPI transmitter is clocked by SPI\_SCK controller. It can output different data frame formats, depending on software configuration. There is a SPI\_DT register available in the SPI that is used to be written with the data to be transmitted. When the transmitter is clocked, the contents in the SPI\_DT register are copied into the data buffer (Unlike SPI\_DT, it is driven by SPI\_SCK, and controlled by hardware, instead of software), and sent out in order based on the programmed frame format.

Both DMA and CPU can be used for write operation. For DMA transfer, refer to DMA transfer section for more details. For CPU transfer, attention should be paid to the TDBE bit. The reset value of this bit is 1, indicating that the SPI\_DT register is empty. If the TDBeIE bit is set, an interrupt is generated. After the data is written, the TDBE is pulled low until the data is moved to the transmit data buffer before the TDBE is set once again. This means that the user can be allowed to write the data to be transmitted only when the TDBE is set.

After the transmitter is configured and the SPI is enabled, the SPI is ready for data transmission. Before going forward, it is necessary for the users to refer to full-duplex / half-duplex chapter to get detailed configuration information, go to the Chip select controller chapter for specific chip select mode, check the SPI\_SCK controller chapter for information on communication clock, and refer to CRC and DMA transfer chapter to configure CRC and DMA (if necessary). The recommended configuration procedure are as follows.

#### Transmitter configuration procedure:

- Configure full-duplex/half-duplex selector
- Configure chip select controller
- Configure SPI\_SCK controller
- Configure CRC (if necessary)
- Configure DMA transfer (if necessary)
- If the DMA transfer mode is not used, the software will check whether to enable transmit data interrupt (TDBeIE =1) through the TDBE bit.
- Configure frame format: select MSB/LSB mode with the LTF bit, and select 8/16-bit data with the FBN bit
- Enable SPI by setting the SPIEN

### 13.2.8 Receiver

The SPI receiver is clocked by the SPI\_SCK controller. It can output different data frame formats through software configuration. There is a receive data buffer register, driven by the SPI\_SCK, in the SPI receiver.

At the last CLK of each transfer, the data is moved from the shift register to the receive data buffer register. Then the transmitter sets the receive data complete flag to the SPI logic. When the flag is detected by the SPI logic, the data in the receive data buffer is copied into the SPI\_DT register, with the RDBF being set. This means that the data is received, and it is already stored into the SPI\_DT. In this case, read access to the SPI\_DT register will clear the RDBF bit.

Both DMA and CPU can be used for read operation. For DMA transfer, refer to DMA transfer section for more details. For CPU transfer, attention should be paid to the RDBE bit. The reset value of this bit is 0, indicating that the SPI\_DT register is empty. If the data is received and moved into the SPI\_DT, the RDBF is set, meaning that there are some data to be read in the SPI\_DT register. An interrupt is

generated if the RDBFIE bit is set.

When the next received data is ready to be moved to the SPI\_DT register, if the previous received data is still not read (RDBF=1), then the data overflow occurs. The previous receive data is not lost, but the next received data will do. At this point, the ROERR is set. An interrupt is generated if the ERRIE is set. Read SPI\_DT register and then the SPI\_STS register will clear the ROERR bit. The recommended configuration procedure is as follows.

#### Receiver configuration procedure:

- Configure full-duplex/half-duplex selector
- Configure chip select controller
- Configure SPI\_SCK controller
- Configure CRC (if necessary)
- Configure DMA transfer (if necessary)
- If the DMA transfer mode is not used, the software will check whether to enable receive data interrupt (RDBEIE =1) through the RDBE bit.
- Configure frame format: select MSB/LSB mode with the LTF bit, and select 8/16-bit data with the FBN bit
- Enable SPI by setting the SPIEN
- Enable SPI by setting the SPIEN

### 13.2.9 Motorola mode

This section describes the SPI communication timings, which includes full-duplex and half-duplex master/slave timings.

#### Full-duplex communication – master mode

Configured as follows:

MSTEN=1: Master enable

SLBEN=0: Full-duplex mode

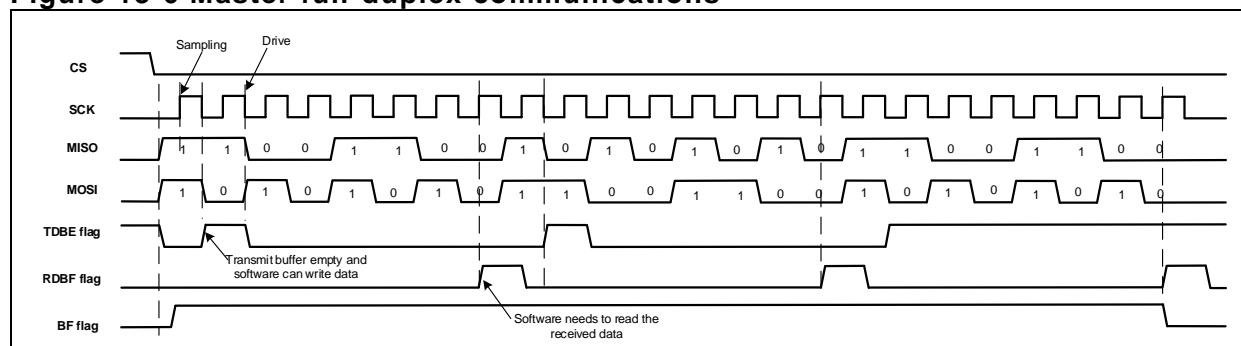
CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Master transmit (MOSI): 0xaa, 0xcc, 0xaa

Slave transmit (MISO): 0xcc, 0xaa, 0xcc

**Figure 13-6 Master full-duplex communications**



#### Full-duplex communication – slave mode

Configured as follows:

MSTEN=0: Slave enable

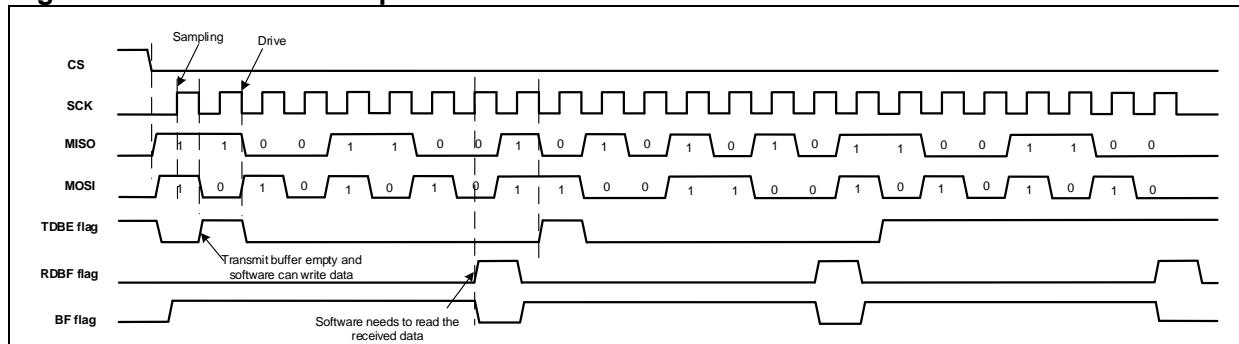
SLBEN=0: Full-duplex mode

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Master transmit (MOSI): 0xaa, 0xcc, 0xaa

Slave transmit (MISO): 0xcc, 0xaa, 0xcc

**Figure 13-7 Slave full-duplex communications****Half-duplex communication – master transmit**

Configured as follows:

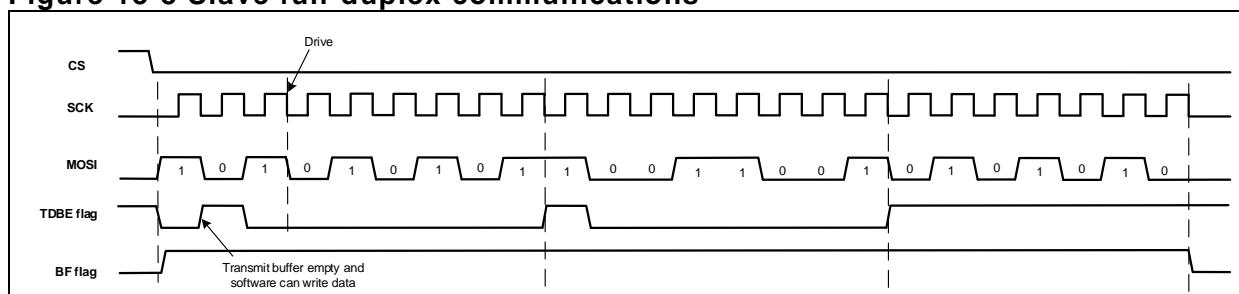
MSTEN=1: Master enable

SLBEN=1: Single line bidirectional mode

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Master transmit (MOSI): 0xaa, 0xcc, 0xaa

**Figure 13-8 Slave full-duplex communications****Half-duplex communication – slave receive**

Configured as follows:

MSTEN=0: Slave enable

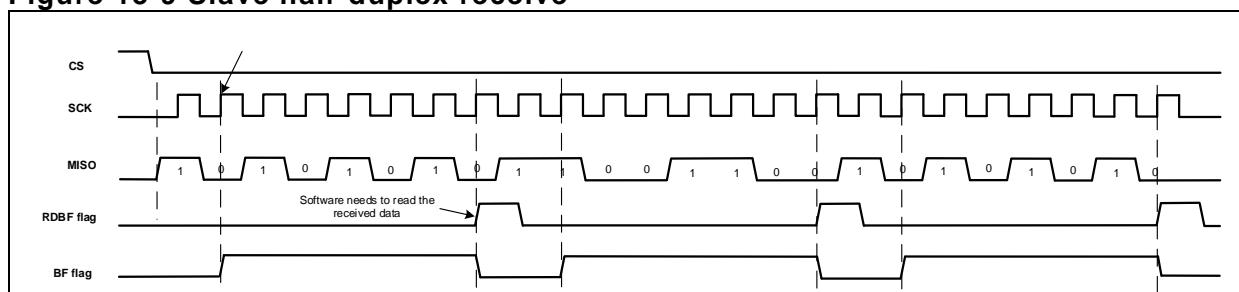
SLBEN=1: Single line bidirectional mode

SLBTD=0: Receive mode

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Slave receive: 0xaa, 0xcc, 0xaa

**Figure 13-9 Slave half-duplex receive****Half-duplex communication – slave transmit**

Configured as follows:

MSTEN=0: Slave enable

SLBEN=1: Single line bidirectional mode

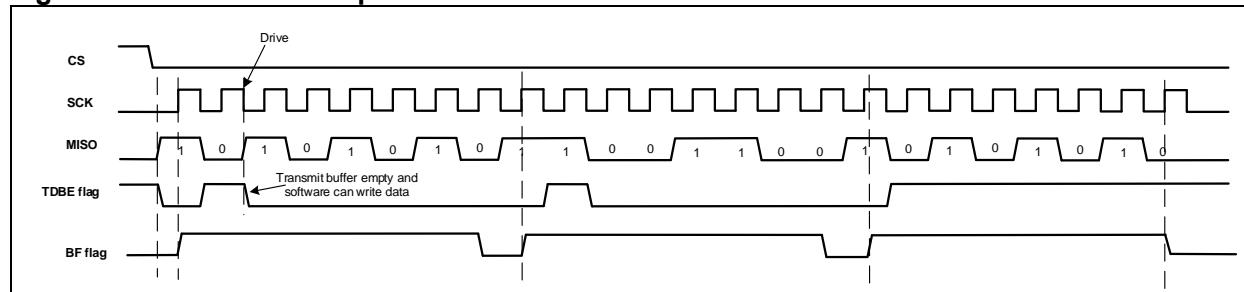
SLBTD=1: Transmit enable

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Slave transmit: 0xaa, 0xcc, 0xaa

**Figure 13-10 Slave half-duplex transmit**



#### Half-duplex communication – master receive

Configured as follows:

MSTEN=1: Master enable

SLBEN=1: Single line bidirectional mode

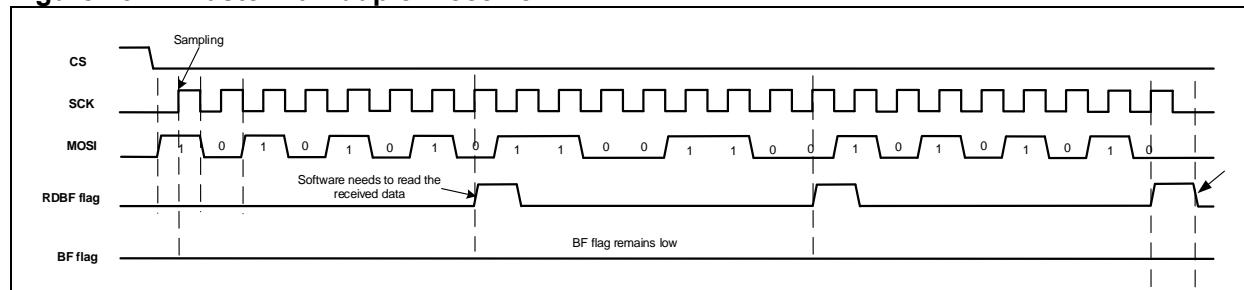
SLBTD=0: Receive enable

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

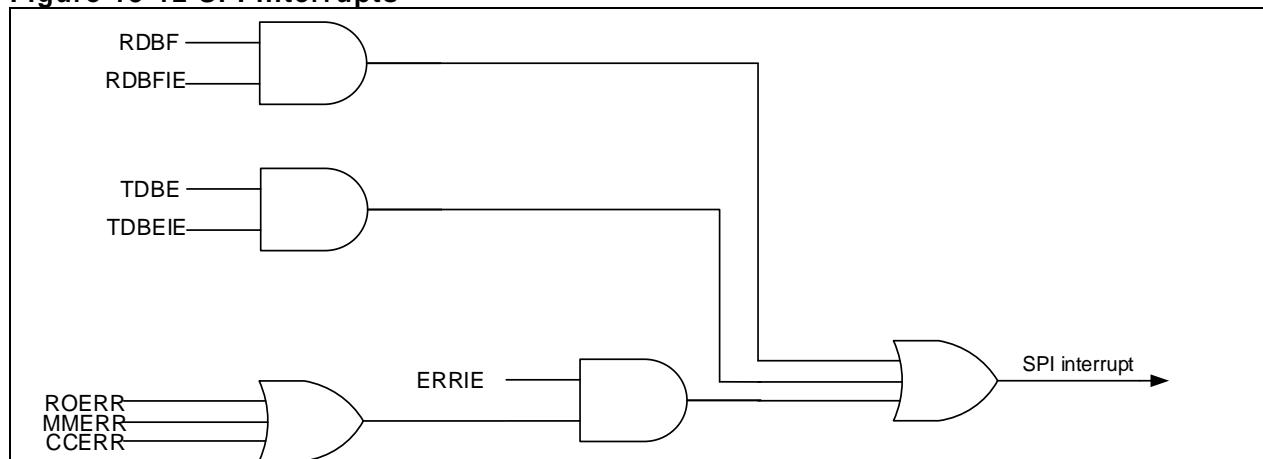
Master receive: 0xaa, 0xcc, 0xaa

**Figure 13-11 Master half-duplex receive**



### 13.2.10 Interrupts

**Figure 13-12 SPI interrupts**



### 13.2.11 IO pin control

When used as SPI, the SPI interface is connected to peripherals through up to four pins. Refer to Section 13.2.2 and Section 13.2.3 for more information on the usage of pins.

- MISO: Master In/Slave Out. The pin receives data in SPI master mode, and transmits data in SPI slave mode.
- MOSI: Master Out/Slave In. The pin transmits data in SPI master mode, and receives data in SPI slave mode.
- SCK: SPI communication clock pin. In SPI master mode, the pin outputs the communication clock to peripherals. In SPI slave mode, the pin inputs the communication clock to SPI interface.
- CS: Chip Select. This is an optional pin which selects master/slave device. Refer to 13.2.3 for more information.

Note: PA13/14 can be used SWDI and SWCK, or as SPI2 MISO/MOSI. Therefore, when one wants to use PA13/14 as SPI2, it is necessary to insert a delay for code download prior to GPIO remap.

### 13.2.12 Precautions

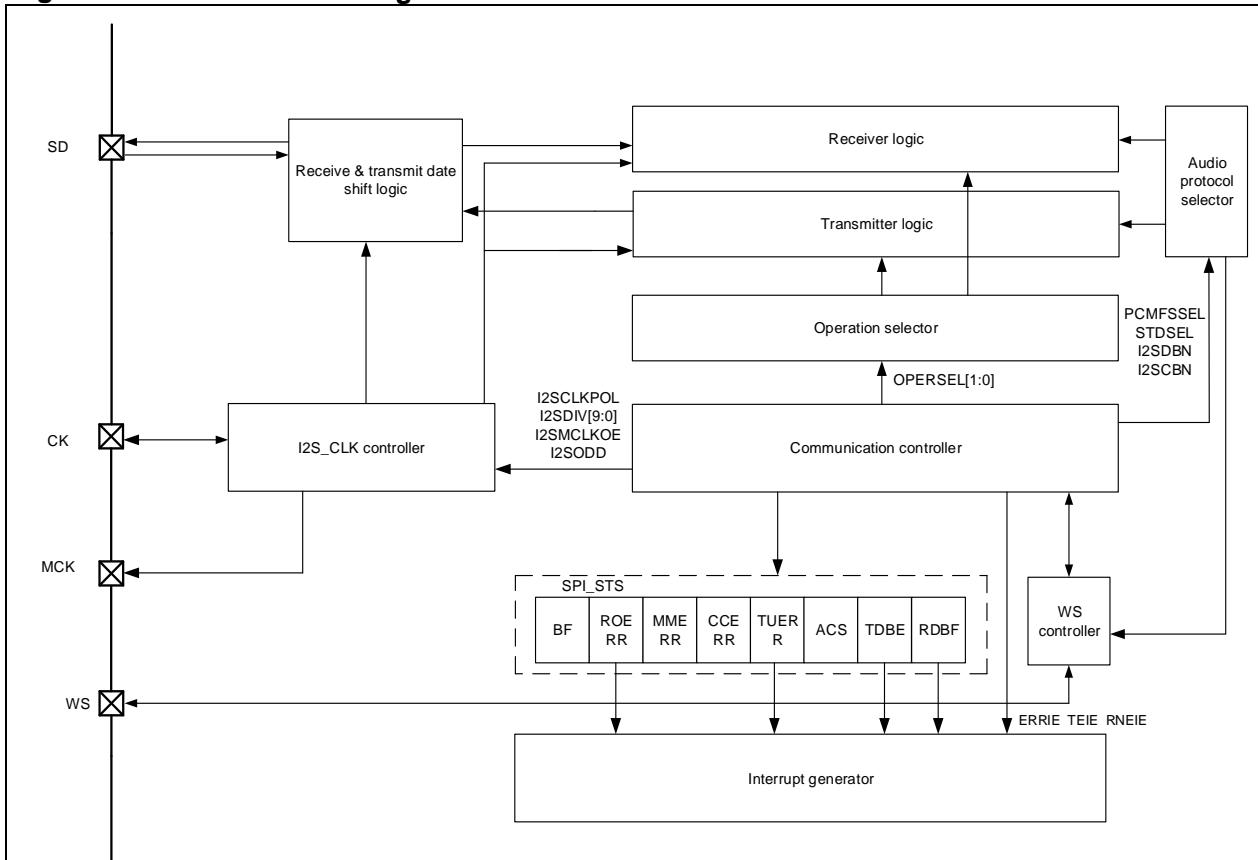
CRC value is obtained by software reading DT register at the end of CRC reception

## 13.3 I<sup>2</sup>S functional description

### 13.3.1 I<sup>2</sup>S introduction

The I<sup>2</sup>S can be configured by software as master reception/transmission, and slave reception/transmission, supporting four kinds of audio protocols including Philips standard, MSB-aligned standard, LSB-aligned standard and PCM standard, respectively. The DMA transfer is also supported.

**Figure 13-13 I<sup>2</sup>S block diagram**



#### Main features when the SPI is used as I<sup>2</sup>S:

- Programmable operation mode
  - Slave device transmission
  - Slave device reception

- Master device transmission
- Master device reception
- Programmable clock polarity
- Programmable clock frequency (8 KHz to 192 KHz)
- Programmable data bits (16 bit, 24 bit, 32 bit)
- Programmable channel bits (16 bit, 32 bit)
- Programmable audio protocol
  - I<sup>2</sup>S Philips standard
  - MSB-aligned standard (left-aligned)
  - LSB-aligned standard (right-aligned)
  - PCM standard (long or short frame)
- DMA transfer
- Main peripheral clock with a fixed frequency of 256x Fs (audio sampling frequency)

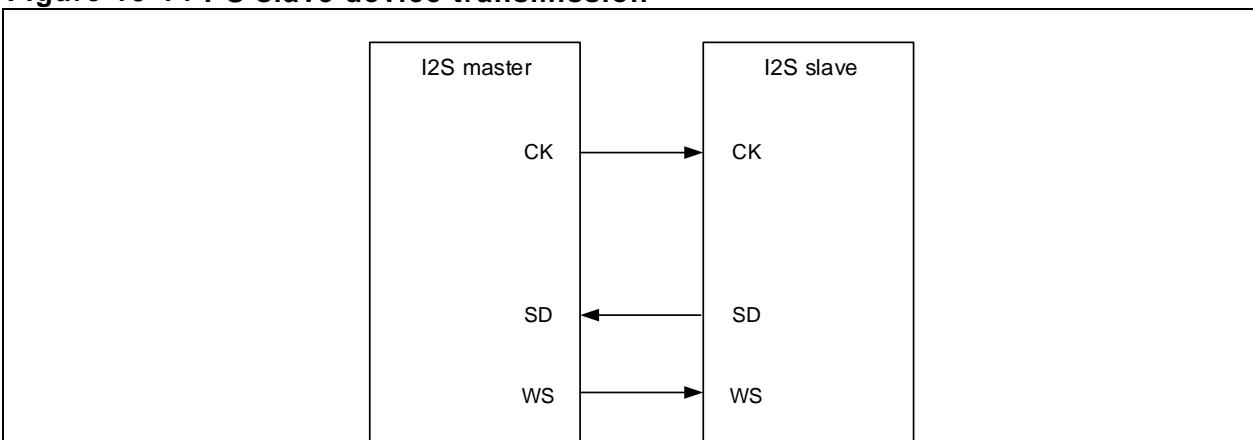
### 13.3.2 Operation mode selector

The SPI, used as I<sup>2</sup>S selector, offers multiple operation modes for selection, namely, slave device transmission, slave device reception, master device transmission and master device reception. This is done by software configuration.

#### Slave device transmission:

Set the I2SMSEL bit, and OPERSEL[1:0]=00, the I<sup>2</sup>S will work in slave device transmission mode.

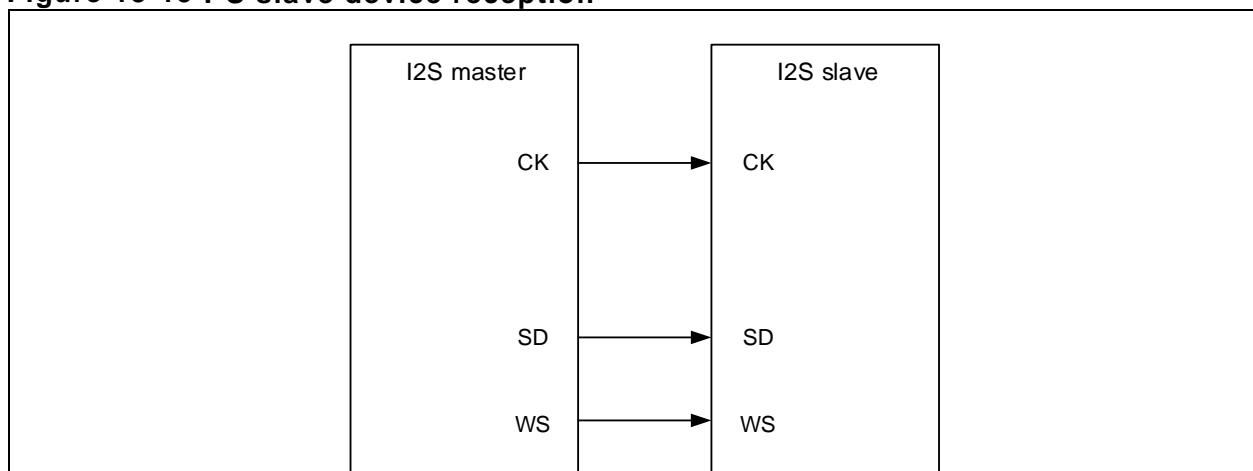
**Figure 13-14 I<sup>2</sup>S slave device transmission**



#### Slave device reception:

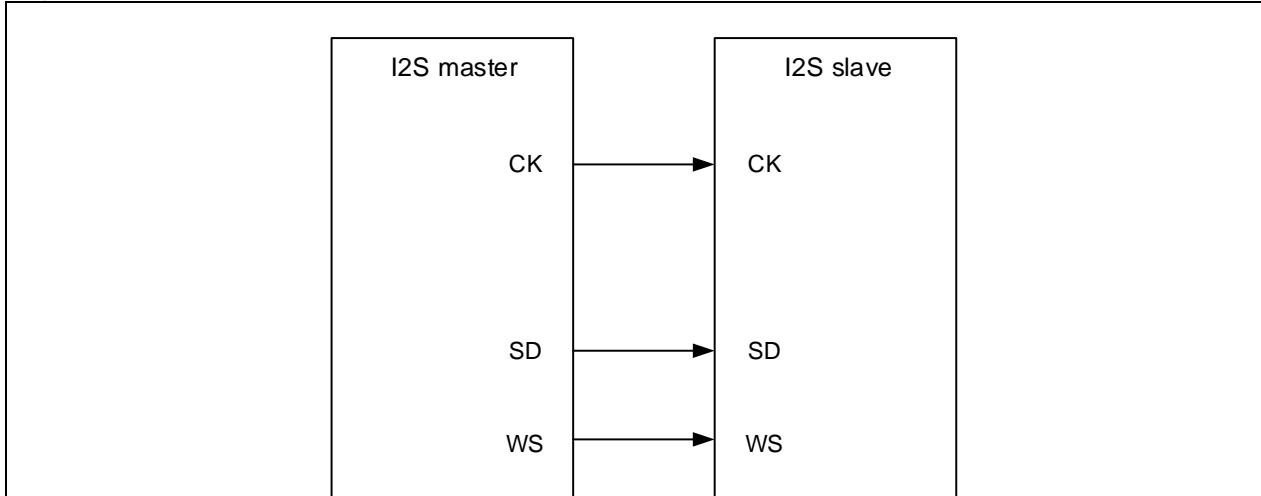
Set the I2SMSEL bit, and OPERSEL[1:0]=01, the I<sup>2</sup>S will work in slave device reception mode.

**Figure 13-15 I<sup>2</sup>S slave device reception**

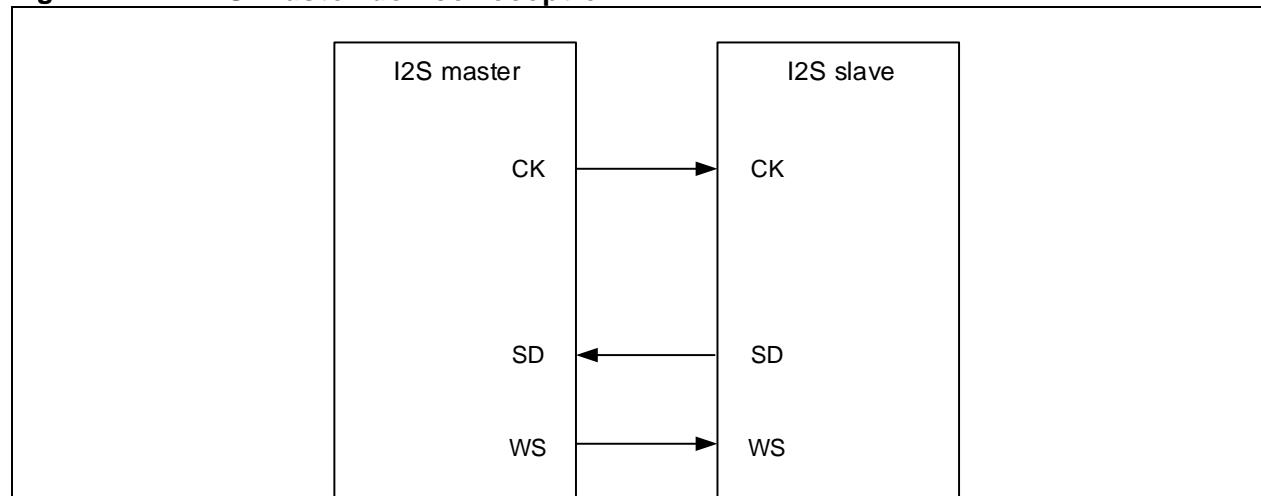


**Master device transmission:**

Set the I2SMSEL bit, and OPERSEL[1:0]=10, the I<sup>2</sup>S will work in master device transmission mode.

**Figure 13-16 I<sup>2</sup>S master device transmission****Master device reception:**

Set the I2SMSEL bit, and OPERSEL[1: 0]=11, the I<sup>2</sup>S will work in master device reception mode.

**Figure 13-17 I<sup>2</sup>S master device reception**

### 13.3.3 Audio protocol selector

While being used as I<sup>2</sup>S, the SPI supports multiple audio protocols. The user can control the audio protocol selector through software configuration to select the desired audio protocol, with the data bits and channel bits being controlled by the audio protocol selector. Besides, the user can also select the data bits and channel bits through software configuration. Meanwhile, the audio protocol selector manages the WS controller, output or detect the WS signal that meets the protocol requirements.

- Select audio protocol by setting the STDSEL bit  
 STDSLE=00: Philips standard  
 STDSLE=01: MSB-aligned standard (left-aligned)  
 STDSLE=10: LSB-aligned standard (right-aligned)  
 STDSLE=11: PCM standard
- Select PCM frame synchronization format: PCMFSEL=1 for PCM long frame synchronization, PCMFSEL=0 for short frame synchronization (this step is required when selecting PCM protocol)
- Select data bits by setting the I2SDBN bit  
 I2SDBN=00: 16 bit  
 I2SDBN =01: 24 bit  
 I2SDBN =10: 32 bit

- Select channel bits by setting the I2SCBN bit  
I2SDBN =0: 16 bit  
I2SDBN =1: 32 bit

*Note: Read/Write operation mode depends on the selected audio protocols, data bits and channel bits. The following lists all possible configuration combinations and their respective read and write operation mode.*

- Philips standard, PCM standard, MSB-aligned or LSB-aligned standard, 16-bit data and 16-bit channel  
The data bit is the same as the channel bit. Each channel requires one read/write operation from/ to the SPI\_DT register, and the number of DMA transfer is 1.
- Philips standard, PCM standard or MSB-aligned standard, 16-bit data and 32-bit channel  
The data bit is different from the channel bit. Each channel requires one read/write operation from/to the SPI\_DT register, and the number of DMA transfer is 1. The first 16 bits (MSB) are the significant bits, and the 16-bit LSB is forced to 0 by hardware.
- Philips standard, PCM standard or MSB-aligned standard, 24-bit data and 32-bit channel  
The data bit is different from the channel bit. Each channel requires two read/write operations from/to the SPI\_DT register, and the number of DMA transfer is 2. The 16-bit MSB transmits and receives the first 16-bit data, the 16-bit LSB transmits and receives the 8-bit MSB data, with 8-bit LSB data being forced to 0 by hardware.
- Philips standard, PCM standard, MSB-aligned or LSB-aligned standard, 32-bit data and 32-bit channel  
The data bit is the same as the channel bit. Each channel requires two read/write operations from/to the SPI\_DT register, and the number of DMA transfer is 2. These 32-bit data are proceeded in two times, with 16-bit data each time.
- LSB-aligned standard, 16-bit data and 32-bit channel  
The data bit is different from the channel bit. Each channel requires one read/write operation from/to the SPI\_DT register, and the number of DMA transfer is 1. The 16 bits (LSB) are the significant bits while the first 16-bit data (MSB) are forced to 0 by hardware.
- LSB-aligned standard, 24-bit data and 32-bit channel  
The data bit is different from the channel bit. Each channel requires two read/write operations from/to the SPI\_DT register, and the number of DMA transfer is 2. For the first 16-bit data, its 8-bit LSB are the significant bits, with the 8-bit MSB forced to 0 by hardware; the subsequent 16 bits transmit and receive the second 16-bit data.

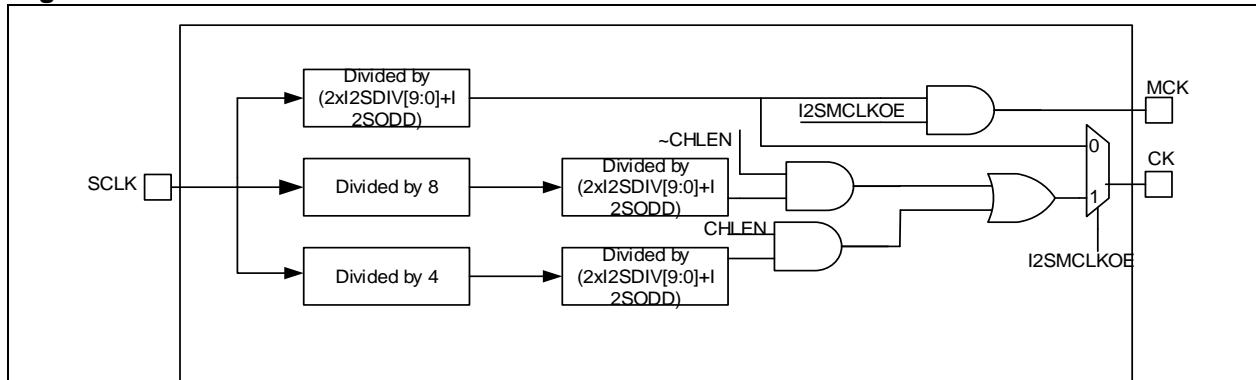
### 13.3.4 I2S\_CLK controller

The audio protocols the SPI supports adopts synchronous transmission. In master mode, it is required to generate a communication clock for data reception and transmission on the SPI, and the communication clock should be output to the slave via IO for data reception and transmission. In slave mode, the communication clock is provided by master, and is input to the SPI via IO. In all, the I2S\_SCK controller is used for the generation and distribution of I2S\_SCK, with the configuration procedure detailed as follows:

When used as I2S master, the SPI can provide communication clock (CK) and main peripheral clock (MCK) shown in Figure 13-13. The CK and MCK are generated by HCLK divider, with the prescaler of the MCK determined by I2SDIV and I2SODD. The calculation formula is seen in Figure 13-12.

The prescaler of the CK depends on whether to provide the main clock for peripherals. To ensure that the main clock is always 256 times larger than the audio sampling frequency, The channel bits should be taken into account. When the main clock is needed, the CK should be divided by 8 (I2SCBN=0) or 4 (I2SCBN=1), then divided again by the same prescaler as that of the MCK, that is the final communication clock; When the main clock is not needed, the prescaler of the CK is determined by I2SDIV and I2SODD, shown in Figure 13-12.

Figure 13-18 CK &amp; MCK source in master mode



Apart from the above-mentioned configuration, the following table lists the values of I2SDIV and I2SODD corresponding to some specific frequencies, as well as their respective error for the users to configure the I2SDIV and I2SODD.

Table 13-1 Audio frequency precision using system clock

SCLK (MHz)	MCL K	Target Fs (Hz)	16bit				32bit			
			I2S DIV	I2S_ODD	RealFs	Error	I2S DIV	I2S_ODD	RealFs	Error
120	No	192000	10	0	187500	2.34%	10	0	187500	2.34%
120	No	96000	19	1	96153	0.16%	10	0	93750	2.34%
120	No	48000	39	0	48076	0.16%	19	1	48076	0.16%
120	No	44100	42	1	44117	0.04%	21	1	43604	1.12%
120	No	32000	58	1	32051	0.16%	29	1	31779	0.69%
120	No	22050	85	0	22058	0.04%	42	1	22058	0.04%
120	No	16000	117	0	16025	0.16%	58	1	16025	0.16%
120	No	11025	170	0	11029	0.04%	85	0	11029	0.04%
120	No	8000	234	1	7995	0.05%	117	1	8012	0.16%
120	Yes	96000	2	1	93750	2.34%	2	1	937500	2.34%
120	Yes	48000	5	0	46875	2.34%	5	0	46875	2.34%
120	Yes	44100	5	1	42613	3.37%	5	1	42613	3.37%
120	Yes	32000	7	1	31250	2.34%	7	1	31250	2.34%
120	Yes	22050	10	1	22321	1.23%	10	1	22321	1.23%
120	Yes	16000	14	1	16163	1.02%	14	1	16163	1.02%
120	Yes	11025	21	1	10901	1.023%	21	1	10901	1.023%
120	Yes	8000	29	1	7944	0.68%	29	1	7944	0.68%
108	No	192000	9	0	187500.0	2.34%	4	1	187500.0	2.34%
108	No	96000	17	1	96428.57	0.446%	9	0	93750	2.34%
108	No	48000	35	0	48214.29	0.446%	17	11	48214.29	0.446%
108	No	44100	38	1	43831.11	0.61%	19	00	44407.89	0.698%
108	No	32000	52	1	32142.86	0.446%	26	11	31839.62	0.501%
108	No	22050	76	1	22058.82	0.04%	38	10	21915.58	0.609%
108	No	16000	110	1	15995.26	0.029%	55	01	16071.43	0.446%
108	No	11025	153	0	11029.41	0.04%	76	11	11029.41	0.04%
108	No	8000	221	0	7997.63	0.029%	105	10	7997.63	0.029%
108	Yes	96000	2	0	105468.8	9.86%	2	0	105468.8	9.86%
108	Yes	48000	4	1	46875.0	2.34%	4	1	46875.0	2.34%
108	Yes	44100	5	0	42187.5	2.34%	5	0	42187.5	2.34%
108	Yes	32000	6	0	32451.92	1.41%	6	0	32451.92	1.41%

108	Yes	22050	9	1	22203.95	0.698%	9	1	22203.95	0.698%
108	Yes	16000	13	0	16225.96	1.41%	13	0	16225.96	1.41%
108	Yes	11025	19	0	11101.97	0.698%	17	1	11101.97	0.698%
108	Yes	8000	26	1	7959.906	0.501%	26	1	7959.906	0.501%
72	No	192000	6	0	187500	2.34%	3	0	187500	2.34%
72	No	96000	11	1	97826.09	1.90%	6	0	93750	2.34%
72	No	48000	32	1	34615.38	27.88%	11	1	48913.04	1.90%
72	No	44100	25	1	44117.65	0.04%	13	0	43269.23	1.88%
72	No	32000	35	0	32142.86	0.45%	17	1	32142.86	0.45%
72	No	22050	51	0	22058.82	0.04%	25	1	22058.82	0.04%
72	No	16000	70	1	15957.45	0.27%	35	0	16071.43	0.45%
72	No	11025	102	0	11029.41	0.04%	51	0	11029.41	0.04%
72	No	8000	140	1	8007.117	0.09%	70	1	7978.723	0.27%
72	Yes	96000	2	0	70312.5	26.76%	2	0	70312.5	26.76%
72	Yes	48000	3	0	46875	2.34%	3	0	46875	2.34%
72	Yes	44100	3	0	46875	6.29%	3	0	46875	6.29%
72	Yes	32000	4	1	31250	2.34%	4	1	31250	2.34%
72	Yes	22050	6	1	21634.62	1.88%	6	1	21634.62	1.88%
72	Yes	16000	9	0	15625	2.34%	9	0	15625	2.34%
72	Yes	11025	13	0	10817.31	1.88%	13	0	10817.31	1.88%
72	Yes	8000	17	1	8035.714	0.45%	17	1	8035.714	0.45%

### 13.3.5 DMA transfer

The SPI supports write and read operations with DMA. Whether used as SPI or I<sup>2</sup>S, read/write request using DMA comes from the same peripheral. As a result, their configuration procedure are the same, described as follows.

#### Transmission with DMA

- Select DMA channel: Select a DMA channel for the current SPI from DMA channel map table described in DMA chapter.
- Configure the destination of DMA transfer: Configure the SPI\_DT register address as the destination address bit of DMA transfer in the DMA control register. Data will be sent to this address after transmit request is received by DMA.
- Configure the source of DMA transfer: Configure the memory address as the source of DMA transfer in the DMA control register. Data will be loaded into the SPI\_DT register from the memory address after transmit request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the channel priority of DMA transfer in the DMA control register.
- Configure DMA interrupt generation after half or full transfer in the DMA control register.
- Enable DMA transfer channel in the DMA control register.

#### Reception with DMA

- Select DMA transfer channel: Select a DMA channel for the current SPI from DMA channel map table described in DMA chapter.
- Configure the destination of DMA transfer: Configure the memory address as the destination of DMA transfer in the DMA control register. Data will be loaded from the SPI\_DT register to the programmed destination after reception request is received by DMA.
- Configure the source of DMA transfer: Configure the SPI\_DT register address as the source of DMA transfer in the DMA control register. Data will be loaded from the SPI\_DT register to the programmed destination after reception request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the total number of bytes to be transferred in the DMA control register.

- Configure DMA interrupt generation after half or full transfer in the DMA control register
- Enable DMA transfer channel in the DMA control register.

### 13.3.6 Transmitter/Receiver

Whether being used as SPI or I<sup>2</sup>S, there is no difference for CPU. The SPI (in whatever mode) shares the same base address, the same SPI\_DT register, the same transmitter and receiver. The SPI transmitter and receiver is responsible for sending and receiving the desired data frame according to the configuration of the communication controller. Thus their status flags such as TDBE, RDBF and ROERR, and their interrupt enable bits including TDBEIE, RDBFIE and ERRIE are identical.

Special attention must be paid to:

- CRC check is not available on the I<sup>2</sup>S. Any operation linked to CRC, including CCERR flag and the corresponding interrupts, is not supported.
- I<sup>2</sup>S protocol needs decode the current channel status. The ACS bit is used to judge whether the current transfer occurs on the left channel (ACS=0) or the right channel (ACS=1).
- TUERR bit indicates whether an underrun occurs. TUERR=1 means an underrun error occurs on the transmitter. An interrupt is generated when the ERRIE is set.
- Read/write operation to the SPI\_DT register is different under different audio protocols, data bits and channel bits. Refer to the audio protocol selector section for more information.
- Pay more attention to the I<sup>2</sup>S disable operation under different configurations, shown as follows:
  - I2SDBN=00, I2SCBN=1, STDSLE=10: wait for the second-to-last RDBF=1 and 17 CK periods before disabling the I<sup>2</sup>S
  - I2SDBN=00, I2SCBN=1, STDSLE=00 or STDSLE=01 or STDSLE=11: wait for the last RDBF=1 and one CK period before the I<sup>2</sup>S
  - I2SDBN, I2SCBN, STDSLE combination: wait for the second-to-last RDBF=1 and one CK period before disabling the I<sup>2</sup>S.

#### I<sup>2</sup>S transmitter configuration procedure:

- Configure operation mode selector
- Configure audio protocol selector
- Configure I2S\_SCK controller
- Configure DMA transfer (if necessary)
- Set the I2SEN bit to enable I<sup>2</sup>S

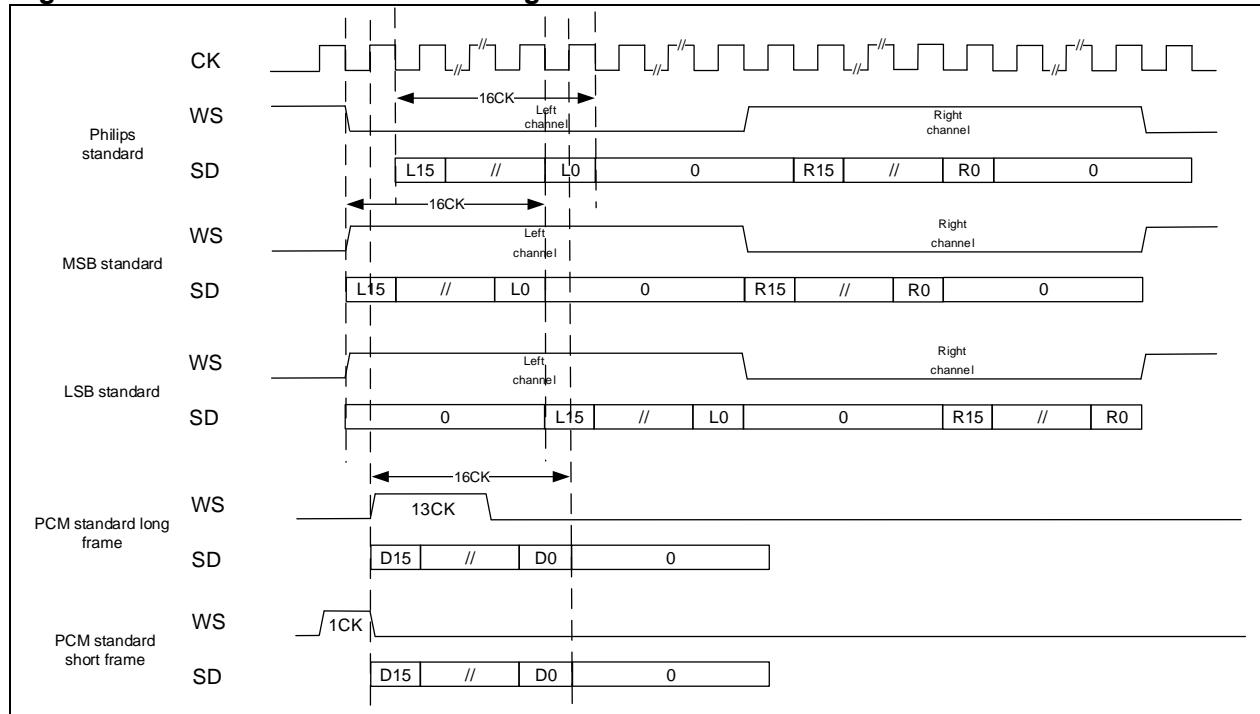
#### I<sup>2</sup>S receiver configuration procedure:

- Configure operation mode selector
- Configure audio protocol selector
- Configure I2S\_SCK controller
- Configure DMA transfer (if necessary)
- Set the I2SEN bit to enable I<sup>2</sup>S

### 13.3.7 I<sup>2</sup>S communication timings

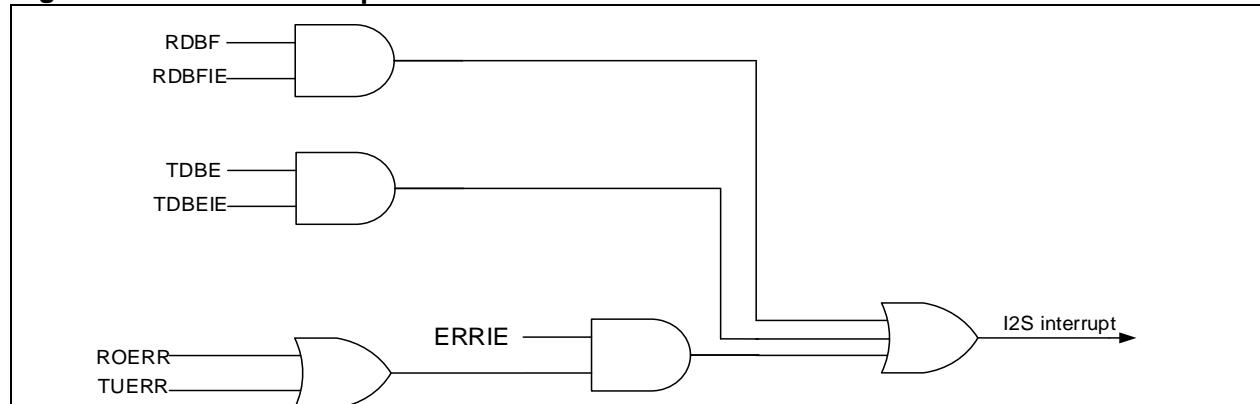
I<sup>2</sup>S supports four different audio standards: Philips standard, the most significant byte (left-aligned) and the least significant byte (right-aligned) standards, and the PCM standard. [Figure 13-19](#) shows their respective timings.

**Figure 13-19 Audio standard timings**



### 13.3.8 Interrupts

**Figure 13-20 I<sup>2</sup>S interrupts**



### 13.3.9 IO pin control

The I<sup>2</sup>S needs three pins for transfer operation, namely, the SD, WS and CK. The MCLK pin is also required if need to provide main clock for peripherals. The I<sup>2</sup>S shares some pins with the SPI, described as follows:

- SD: Serial data (mapped on the MOSI pin) for bidirectional data transmission and reception.
- WS: Word select (mapped on the CS pin) for data control signal output in master mode, and input in slave mode.
- CK: Communication clock (mapped on the SCK pin) as clock signal output in master mode, and input in slave mode.
- MCLK: Master clock (mapped independently) is used to provide main clock for peripherals. The clock frequency output is fixed 256x Fs (audio sampling frequency). Some MCLK and MISO pins share the same GPIO map.

## 13.4 SPI registers

These peripheral registers must be accessed by half-word (16 bits) or word (32 bits).

**Table 13-2 SPI register map and reset value**

Register	Offset	Reset value
SPI_CTRL1	0x00	0x0000
SPI_CTRL2	0x04	0x0000
SPI_STS	0x08	0x0002
SPI_DT	0x0C	0x0000
SPI_CPOLY	0x10	0x0007
SPI_RCRC	0x14	0x0000
SPI_TCRC	0x18	0x0000
SPI_I2SCTRL	0x1C	0x0000
SPI_I2SCLKP	0x20	0x0002

### 13.4.1 SPI control register1 (SPI\_CTRL1) (Not used in I<sup>2</sup>S mode)

Bit	Register	Reset value	Type	Description
Bit 15	SLBEN	0x0	rw	Single line bidirectional half-duplex enable 0: Disabled 1: Enabled
Bit 14	SLBTD	0x0	rw	Single line bidirectional half-duplex transmission direction This bit and the SLBEN bit together determine the data output direction in "Single line bidirectional half-duplex" mode. 0: Receive-only mode 1: Transmit-only mode
Bit 13	CCEN	0x0	rw	RC calculation enable 0: Disabled 1: Enabled
Bit 12	NTC	0x0	rw	Transmit CRC next When this bit is set, it indicates that the next data transferred is CRC value. 0: Next transmitted data is the normal value 1: Next transmitted data is CRC value
Bit 11	FBN	0x0	rw	Frame bit num This bit is used to configure the number of data frame bit for transmission/reception. 0: 8-bit data frame 1: 16-bit data frame
Bit 10	ORA	0x0	rw	Receive-only active In two-wire unidirectional mode, when this bit is set, it indicates that Receive-only is active, but the transmit is not allowed. 0: Transmission and reception 1: Receive-only mode
Bit 9	SWCSEN	0x0	rw	Software CS enable When this bit is set, the CS pin level is determined by the SWCSIL bit. The status of I/O level on the CK pin is invalid. 0: Disabled 1: Enabled
Bit 8	SWCSIL	0x0	rw	Software CS internal level This bit is valid only when the SWCSEN is set. It determines the level on the CS pin. In master mode, this bit must be set. 0: Low level 1: High level

Bit 7	LTF	0x0	rw	LSB transmit first This bit is used to select for MST transfer first or LSB transfer first. 0: MSB 1: LSB
Bit 6	SPIEN	0x0	rw	SPI enable 0: Disabled 1: Enabled
Bit 5: 3	MDIV	0x0	rw	Master clock frequency division In master mode, the peripheral clock divided by the prescaler is used as SPI clock. The MDIV[3] bit is in the SPI_CTRL2 register, MDIV[3: 0]: 0000: Divided by 2 0001: Divided by 4 0010: Divided by 8 0011: Divided by 16 0100: Divided by 32 0101: Divided by 64 0110: Divided by 128 0111: Divided by 256 1000: Divided by 512 1001: Divided by 1024
Bit 2	MSTEN	0x0	rw	Master enable 0: Disabled (Slave) 1: Enabled (Master)
Bit 1	CLKPOL	0x0	rw	Clock polarity Indicates the polarity of clock output in idle state. 0: Low level 1: High level
Bit 0	CLKPHA	0x0	rw	Clock phase 0: Data capture starts from the first clock edge 1: Data capture starts from the second clock edge

Note: The SPI\_CTRL1 register must be 0 in I<sup>2</sup>S mode.

### 13.4.2 SPI control register2 (SPI\_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 9	Reserved	0x00	resd	Forced to be 0 by hardware.
Bit 8	MDIV	0x0	rw	Master clock frequency division Refer to the MDIV[2: 0] of the SPI_CTRL1 register.
Bit 7	TDBEIE	0x0	rw	Transmit data buffer empty interrupt enable 0: Disabled 1: Enabled
Bit 6	RDBFIE	0x0	rw	Receive data buffer full interrupt enable 0: Disabled 1: Enabled
Bit 5	ERRIE	0x0	rw	Error interrupt enable This bit controls interrupt generation when errors occur (CCERR, MMERR, ROERR and TUERR) 0: Disabled 1: Enabled
Bit 4: 3	Reserved	0x0	resd	Kept at its default value
Bit 2	HWCSE	0x0	rw	Hardware CS output enable This bit is valid only in master mode. When this bit is set, the I/O output on the CS pin is low; when this bit is 0, the I/O input on the CS pin must be set high. 0: Disabled 1: Enabled
Bit 1	DMATEN	0x0	rw	DMA transmit enable 0: Disabled 1: Enabled
Bit 0	DMAREN	0x0	rw	DMA receive enable 0: Disabled 1: Enabled

### 13.4.3 SPI status register (SPI\_STS)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Forced to be 0 by hardware
				Busy flag
Bit 7	BF	0x0	ro	0: SPI is not busy. 1: SPI is busy.
				Receiver overflow error
Bit 6	ROERR	0x0	ro	0: No overflow error 1: Overflow error occurs.
				Master mode error
Bit 5	MMERR	0x0	ro	This bit is set by hardware and cleared by software (read/write access to the SPI_STS register, followed by write operation to the SPI_CTRL1 register) 0: No mode error 1: Mode error occurs.
				CRC error
Bit 4	CCERR	0x0	rw0c	Set by hardware, and cleared by software. 0: No CRC error 1: CRC error occurs.
				Transmitter underload error
Bit 3	TUERR	0x0	ro	Set by hardware, and cleared by software (read the SPI_STS register). 0: No underload error 1: Underload error occurs. Note: This bit is only used in I <sup>2</sup> S mode.
				Audio channel state
Bit 2	ACS	0x0	ro	This bit indicates the status of the current audio channel. 0: Left channel 1: Right channel Note: This bit is only used in I <sup>2</sup> S mode.
				Transmit data buffer empty
Bit 1	TDBE	0x1	ro	0: Transmit data buffer is not empty. 1: Transmit data buffer is empty.
				Receive data buffer full
Bit 0	RDBF	0x0	ro	0: Transmit data buffer is not full. 1: Transmit data buffer is full.

### 13.4.4 SPI data register (SPI\_DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DT	0x0000	rw	Data value This register controls read and write operations. When the data bit is set as 8 bit, only the 8-bit LSB [7: 0] is valid.

### 13.4.5 SPICRC register (SPI\_CPOLY) (Not used in I<sup>2</sup>S mode)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CPOLY	0x0007	rw	CRC polynomial This register contains the polynomial used for CRC calculation. Note: This register is valid only in SPI mode.

### 13.4.6 SPIRxCRC register (SPI\_RCRC) (Not used in I<sup>2</sup>S mode)

Bit	Register	Reset value	Type	Description
Bit 15: 0	RCRC	0x0000	ro	<p>Receive CRC</p> <p>When CRC calculation is enabled, this register contains the CRC value computed based on the received data. This register is reset when the CCEN bit in the SPI_CTRL1 register is cleared.</p> <p>When the data frame format is set to 8-bit data, only the 8-bit LSB ([7: 0]) are calculated based on CRC8 standard; when 16-bit data bit is selected, follow CRC16 standard.</p> <p>Note: This register is only used in SPI mode.</p>

### 13.4.7 SPITxCRC register (SPI\_TCRC)

Bit	Register	Reset value	Type	Description
Bit 15: 0	TCRC	0x0000	ro	<p>Transmit CRC</p> <p>When CRC calculation is enabled, this register contains the CRC value computed based on the transmitted data. This register is reset when the CCEN bit in the SPI_CTRL1 register is cleared.</p> <p>When the data frame format is set to 8-bit data, only the 8-bit LSB ([7: 0]) are calculated based on CRC8 standard; when 16-bit data bit is selected, follow CRC16 standard.</p> <p>Note: This register is only used in SPI mode.</p>

### 13.4.8 SPI\_I2S configuration register (SPI\_I2SCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 12	Reserved	0x0	resd	Forced to be 0 by hardware.
Bit 11	I2SMSEL	0x0	rw	<p>I<sup>2</sup>S mode select</p> <p>0: SPI mode 1: I<sup>2</sup>S mode</p>
Bit 10	I2SEN	0x0	rw	<p>I<sup>2</sup>S enable</p> <p>0: Disabled 1: Enabled</p>
Bit 9: 8	OPERSEL	0x0	rw	<p>I<sup>2</sup>S operation mode select</p> <p>00: Slave transmission 01: Slave reception 10: Master transmission 11: Master reception</p>
Bit 7	PCMFSSEL	0x0	rw	<p>PCM frame synchronization</p> <p>This bit is valid only when the PCM standard is used.</p> <p>0: Short frame synchronization 1: Long frame synchronization</p>
Bit 6	Reserved	0x0	resd	Kept at its default value
Bit 5: 4	STDSEL	0x0	rw	<p>I<sup>2</sup>S standard select</p> <p>00: Philips standard 01: MSB-aligned standard (left-aligned) 10: LSB-aligned standard (right-aligned) 11: PCM standard</p>
Bit 3	I2SCLKPOL	0x0	rw	<p>I<sup>2</sup>S clock polarity</p> <p>This bit indicates the clock polarity on the clock pin in idle state.</p> <p>0: Low 1: High</p>
Bit 2: 1	I2SDBN	0x0	rw	I <sup>2</sup> S data bit num

---

				00: 16-bit data length 01: 24-bit data length 10: 32-bit data length 11: Not allowed.
Bit 0	I2SCBN	0x0	rw	I <sup>2</sup> S channel bit num This bit can be configured only when the I <sup>2</sup> S is set to 16-bit data; otherwise, it is fixed to 32-bit by hardware. 0: 16-bit wide 1: 32-bit wide

---

### 13.4.9 SPI\_I2S prescaler register (SPI\_I2SCLKP)

Bit	Register	Reset value	Type	Description
Bit 15: 12	Reserved	0x0	resd	Forced to be 0
Bit 9	I2SMCLKOE	0x0	rw	I <sup>2</sup> S Master clock output enable 0: Disabled 1: Enabled
Bit 8	I2SODD	0x0	rw	Odd factor for I <sup>2</sup> S division 0: Actual divider factor =I2SDIV*2 1: Actual divider factor =(I2SDIV*2)+1
Bit 11: 10 Bit 7: 0	I2SDIV	0x02	rw	I <sup>2</sup> S division It is not allowed to configure I2SDIV[9: 0]=0 or I2SDIV[9: 0]=1

---

## 14 Timer

AT32F421 timers include basic timers, general-purpose timers, and advanced timers.

Please refer to [Section 14.1~ Section 14.6](#) for the detailed function modes. All functions of different timers are shown in the following tables.

**Table 14-1 TMR functional comparison**

Timer type	Timer	Counter bit	Count mode	Repetition	Prescaler	DMA requests	Capture/compare channel	PWM input mode	EXT input	Brake input
Advanced-control timer	TMR1 16		Up Down Up/Down	8-bit	1~65535	O	4	O	O	O
	TMR3 16		Up Down Up/Down	X	1~65535	O	4	O	O	X
	TMR14 16		Up	X	1~65535	X	1	X	X	X
	TMR15 16		Up	O	1~65535	O	2	O	X	O
	TMR16 TMR17 16		Up	O	1~65535	O	1	X	X	O
General-purpose timer	TMR6 16		Up	X	1~65535	O	X	X	X	X
Basic timer	TMR6 16		Up	X	1~65535	O	X	X	X	X
Timer type	Timer	Counter bit	Count mode	PWM output	Single-pulse output	Complementary output	Dead-time	Encoder interface connection	Interfacing with hall sensors	Linkage peripheral
Advanced-control timer	TMR1 16		Up Down Up/Down	O	O	O	O	O	O	Timer synchronization/ADC
	TMR3 16		Up Down Up/Down	O	O	X	X	O	O	Timer synchronization/ADC
	TMR14 16		Up	O	O	X	X	X	X	NA
	TMR15 16		Up	O	O	O	O	X	X	Timer synchronization/ADC
	TMR16 TMR17 16		Up	O	O	X	O	X	X	NA
General-purpose timer	TMR6 16		Up	X	X	X	X	X	X	NA
Basic timer	TMR6 16		Up	X	X	X	X	X	X	NA

## 14.1 General-purpose timer (TMR6)

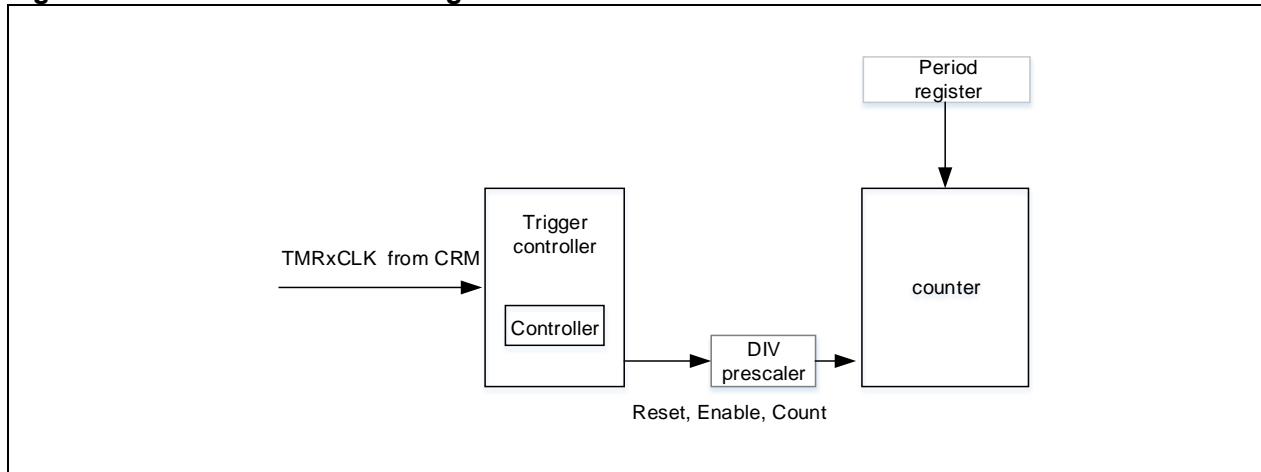
### 14.1.1 TMR6 introduction

The basic timer (TMR6) consists of a 16-bit upcounter and the corresponding control logic, without being connected to external I/Os.

### 14.1.2 TMR6 main features

- Source of counter clock: internal clock
- 16-bit up counter
- Overflow event interrupts and DMA requests supported

**Figure 14-1 Basic timer block diagram**

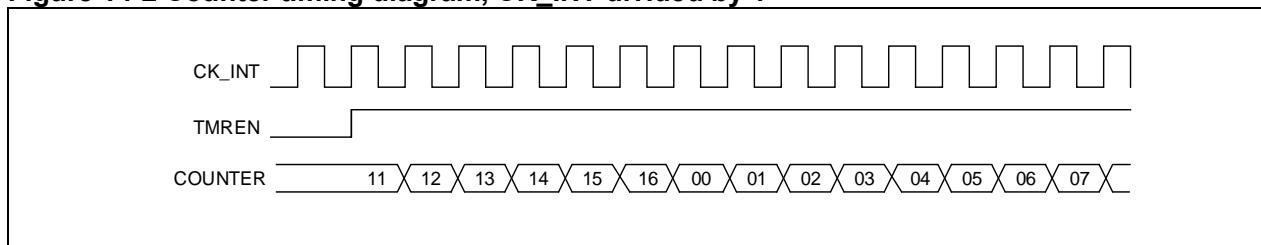


### 14.1.3 TMR6 functional overview

#### 14.1.3.1 Count clock

The counter of TMR6 is clocked by the internal clock source (CK\_INT) divided by prescaler.

**Figure 14-2 Counter timing diagram, CK\_INT divided by 1**



#### 14.1.3.2 Counting mode

The basic timer only supports upcounting mode, and it has an internal 16-bit counter.

The TMRx\_PR register is used to configure the counting period. The value in the TMRx\_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx\_PR register is transferred to the shadow register only at an overflow event.

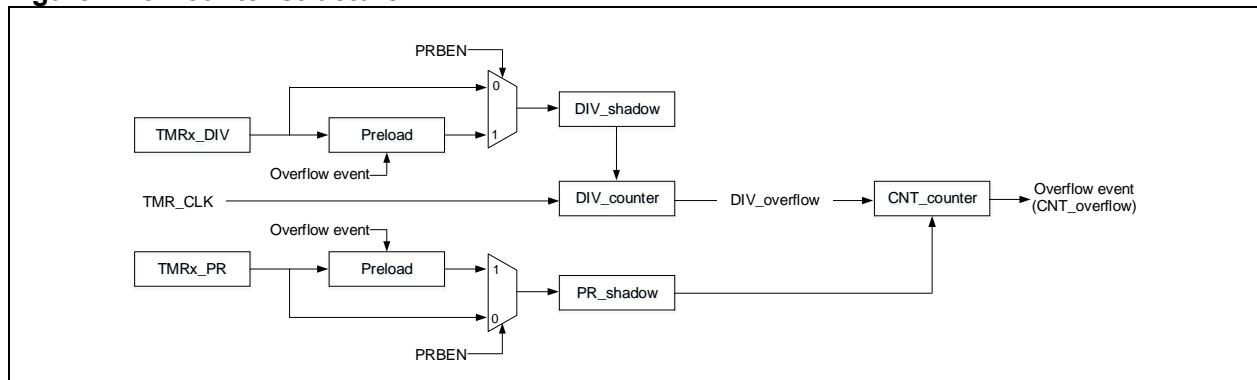
The TMRx\_DIV register is used to configure the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to the TMRx\_PR register, when the periodic buffer is enabled, the value in the TMRx\_DIV register is updated to the shadow register at an overflow event.

Reading the TMRx\_CNT register returns to the current counter value, and writing to the TMRx\_CNT register updates the current counter value to the value being written.

An overflow event is generated by default. Set OVFEN=1 in the TMRx\_CTRL1 to disable generation of update events. The OVFS bit in the TMRx\_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only counter overflow/underflow triggers an overflow event.

Setting the TMREN bit (TMREN=1) enables the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR\_EN is set 1 clock cycle after the TMREN is set.

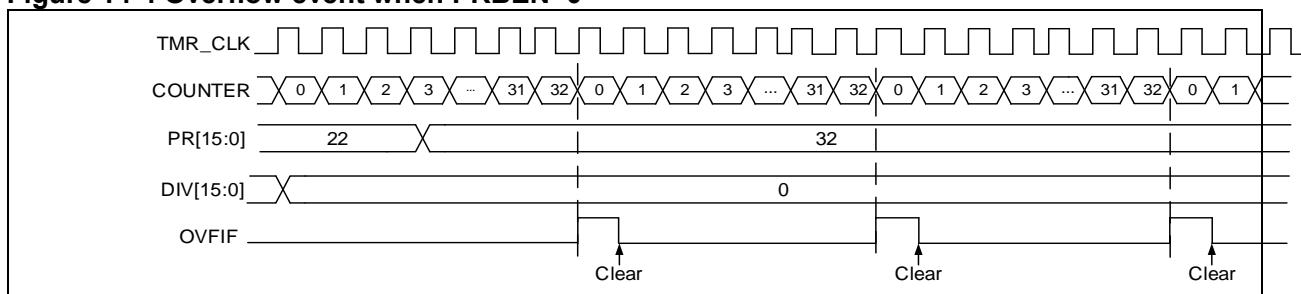
**Figure 14-3 Counter structure**



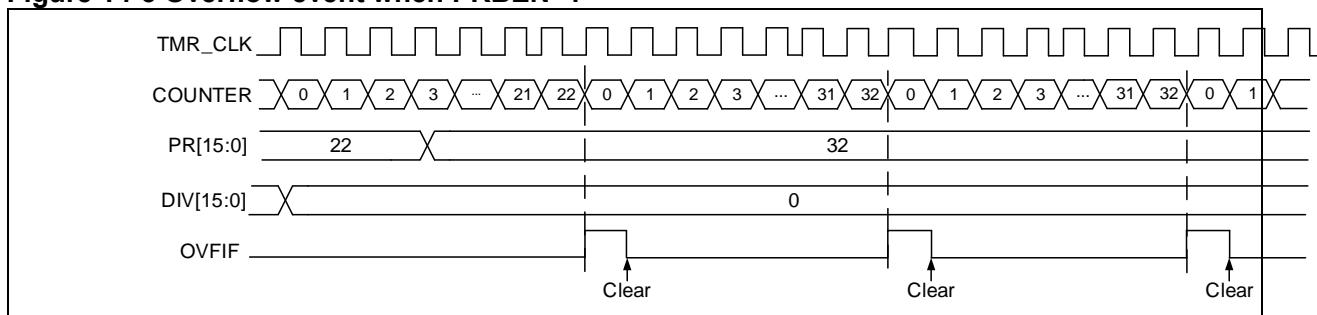
### Upcounting mode

In upcounting mode, the counter counts from 0 to the value programmed in the TMRx\_PR register, restarts from 0, and generates a counter overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the register is no longer reloaded with the preload and re-loaded value after counter overflow occurs, otherwise, the prescaler and re-loaded value will be updated at an overflow event.

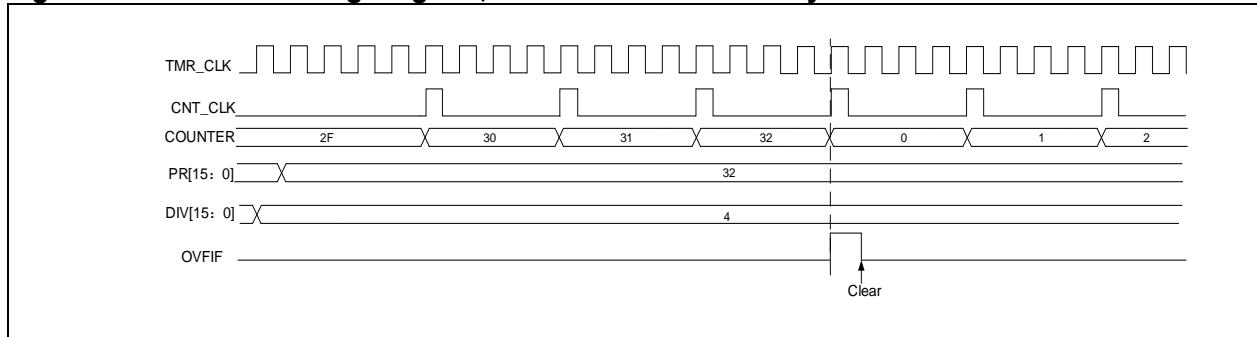
**Figure 14-4 Overflow event when PRBEN=0**



**Figure 14-5 Overflow event when PRBEN=1**



**Figure 14-6 Counter timing diagram, internal clock divided by 4**



### 14.1.3.3 Debug mode

When the microcontroller enters debug mode (Cortex®-M4 core halted), the TMRx counter stops

counting when the TMRx\_PAUSE bit is set.

#### 14.1.4 TMR6 registers

These peripheral registers must be accessed by word (32 bits).

In [Table 14-2](#), all the TMR6 registers are mapped to a 16-bit addressable space.

**Table 14-2 TMR6 register map and reset value**

Register	Offset	Reset value
TMRx_CTRL1	0x00	0x0000
TMRx_CTRL2	0x04	0x0000
TMRx_IDEN	0x0C	0x0000
TMRxISTS	0x10	0x0000
TMRxSWEVT	0x14	0x0000
TMRxCVAL	0x24	0x0000
TMRxDIV	0x28	0x0000
TMRxPR	0x2C	0x0000

#### 14.1.4.1 TMR6 control register1 (TMRx\_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled. 1: Period buffer is enabled.
Bit 6: 4	Reserved	0x0	resd	Kept at its default value.
Bit 3	OCMEN	0x0	rw	One cycle mode enable This bit is used to select whether to stop the counter at an overflow event. 0: Disabled 1: Enabled
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to configure overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated from the slave controller 1: Only counter overflow generates an overflow event.
Bit 1	OVFEN	0x0	rw	Overflow event enable This bit is used to enable or disable OEV event generation. 0: OEV event is enabled. An overflow event is generated by any of the following events: - Counter overflow - Setting the OVFSWTR bit - Overflow event generated from the slave controller 1: OEV event is disabled. If the OVFSWTR bit is set, or a hardware reset is generated from the slave controller, the counter and the prescaler are reinitialized. Note: This bit is set and cleared by software.
Bit 0	TMREN	0x0	rw	TMR enable 0: Disabled 1: Enabled

#### 14.1.4.2 TMR6 control register2 (TMRx\_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6: 4	PTOS	0x0	rw	Master TMR output selection This field is used to select the signals in master mode to be sent to slave timers. 000: Reset 001: Enable 010: Update
Bit 3: 0	Reserved	0x0	resd	Kept at its default value.

#### 14.1.4.3 TMR6 DMA/interrupt enable register (TMRx\_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15: 9	Reserved	0x00	resd	Kept at its default value.
Bit 8	OVFDEN	0x0	rw	Overflow event DMA request enable 0: Disabled 1: Enabled
Bit 7: 1	Reserved	0x00	resd	Kept at its default value.
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

#### 14.1.4.4 TMR6 interrupt status register (TMRx\_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 1	Reserved	0x0000	resd	Kept at its default value.
Bit 0	OVFIF	0x0	rw0c	<p>Overflow interrupt flag This bit is set by hardware at an overflow event. It is cleared by software. 0: No overflow event occurs. 1: Overflow event occurs, and OVFEN=0, and OVFS=0 in the TMRx_CTRL1 register:</p> <ul style="list-style-type: none"> <li>- An overflow event occurs when OVFG=1 in the TMRx_SWEVE register</li> <li>- An overflow event occurs when the counter value (CVAL) is reinitialized by a trigger event.</li> </ul>

#### 14.1.4.5 TMR6 software event register (TMRx\_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 1	Reserved	0x0000	resd	Kept at its default value.
Bit 0	OVFSWTR	0x0	rw0c	<p>Overflow event triggered by software An overflow event is triggered by software. 0: No effect 1: Generate an overflow event by software write operation.</p>

#### 14.1.4.6 TMR6 counter value (TMRx\_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

#### 14.1.4.7 TMR6 division (TMRx\_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	<p>Divider value The counter clock frequency <math>f_{CK\_CNT} = f_{TMR\_CLK} / (DIV[15:0]+1)</math>. At each overflow event, DIV value is sent to the DIV register.</p>

#### 14.1.4.8 TMR6 period register (TMRx\_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0000	rw	<p>Period value This indicates the period value of the TMRx counter. The timer stops working when the period value is 0.</p>

## 14.2 General-purpose timer (TMR3)

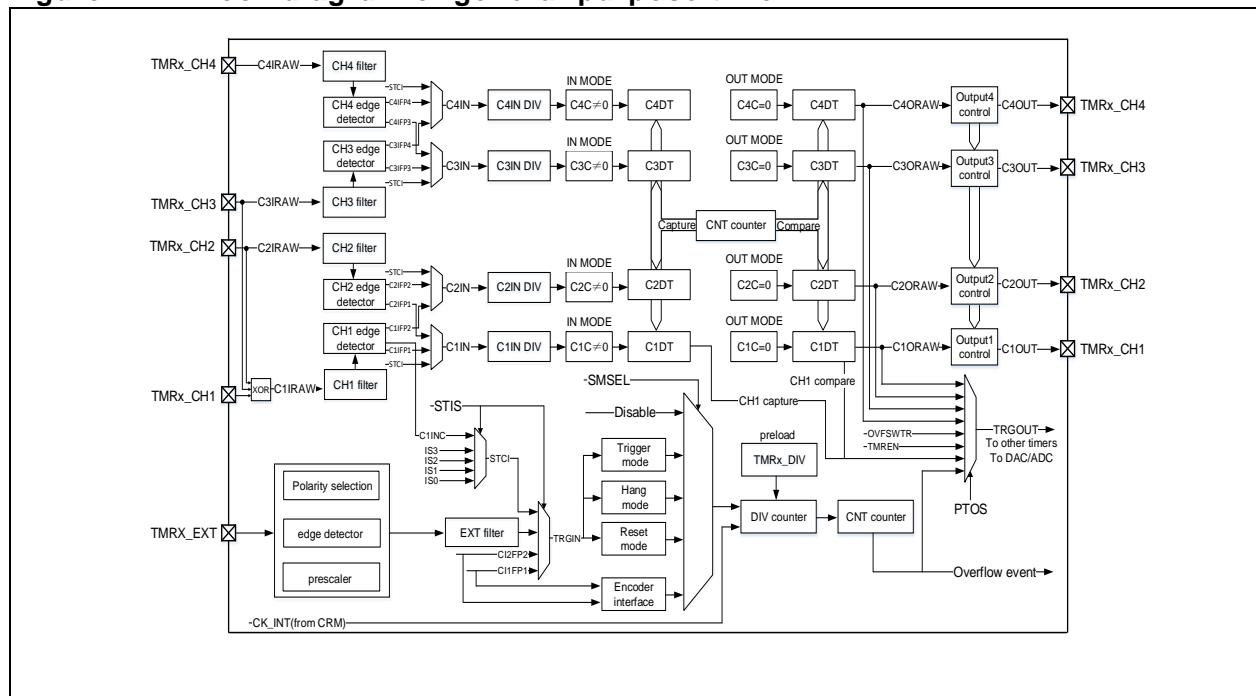
### 14.2.1 TMR3 introduction

The general-purpose timer TMR3 consists of a 16-bit counter supporting up, down, up/down (bidirectional) counting modes, four capture/compare registers, and four independent channels to achieve input capture and programmable PWM output.

### 14.2.2 TMR3 main features

- Source of count clock : internal clock, external clock and internal trigger
- 16-bit up, down, up/down and encoder mode counter
- 4 independent channels for input capture, output compare, PWM generation and one-pulse mode output
- Synchronization control between master and slave timers
- Interrupt/DMA generation on the overflow event, trigger event and channel event
- Support TMR burst DMA transfer

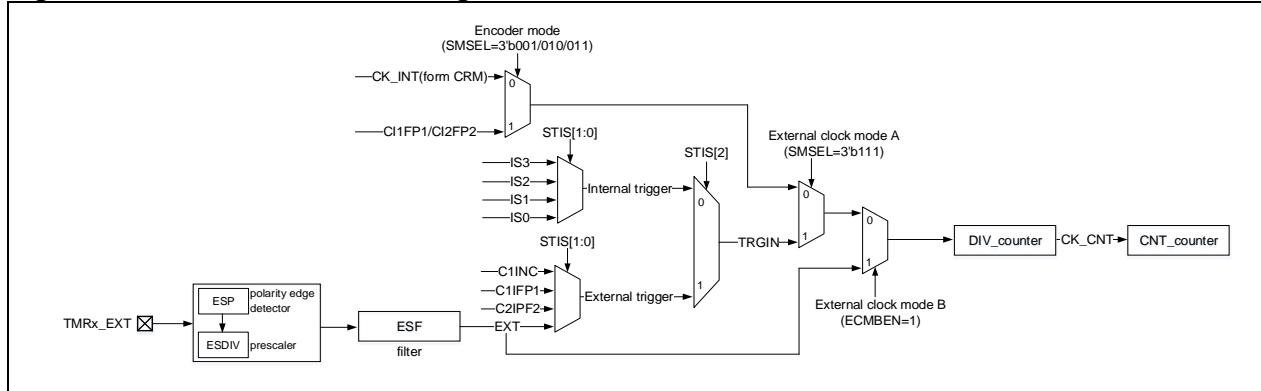
**Figure 14-7 Block diagram of general-purpose timer**



### 14.2.3 TMR3 functional overview

#### 14.2.3.1 Count clock

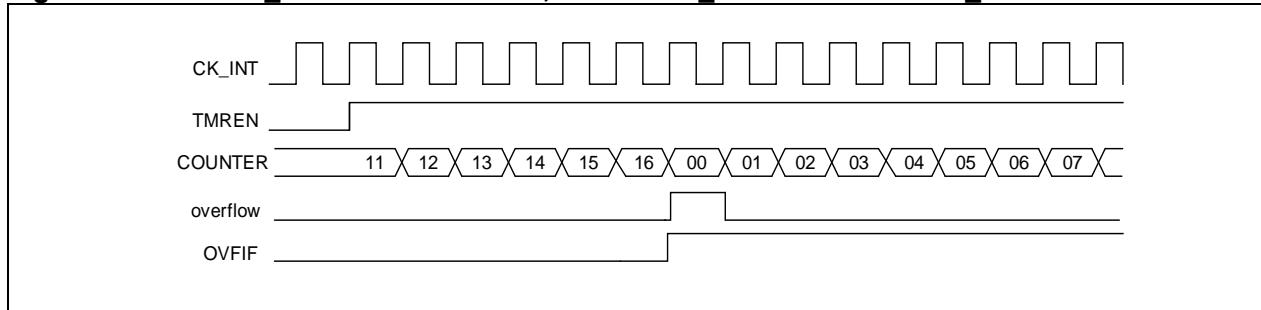
The counter of TMR3 can be clocked by the internal clock (CK\_INT), external clock (external clock mode A and B) and internal trigger input (ISx)

**Figure 14-8 Count clock block diagram**

### Internal clock (CK\_INT)

By default, the CK\_INT divided by a prescaler is used to drive the counter to start counting. The configuration process is as follows:

- Set the TWCMSEL[1:0] bit in the TMRx\_CTRL1 register to select count mode. If the one-way count direction is set, configure OWCDIR bit in the TMRx\_CTRL1 register to select the specific direction.
- Set the TMRx\_DIV register to set counting frequency.
- Set the TMRx\_PR register to set the counting period.
- Set the TMREN bit in the TMRx\_CTRL1 register to enable the counter.

**Figure 14-9 Use CK\_INT to drive counter, with TMRx\_DIV=0x0 and TMRx\_PR=0x16**

### External clock (TRGIN/EXT)

The counter clock can be provided by two external clock sources, namely, TRGIN and EXT signals.

When SMSEL=3'b111, external clock mode A is selected. Set the STIS[2:0] bit to select TRGIN signal to drive the counter to start counting. The external clock sources include: C1INC (STIS=3'b100, channel 1 rising edge and falling edge), C1IFP1 (STIS=3'b101, channel 1 signal with filtering and polarity selection), C2IFP2 (STIS=3'b110, channel 2 signal with filtering and polarity selection) and EXT (STIS=3'b111, external input signal after polarity selection, frequency division and filtering).

When ECMBEN=1, external clock mode B is selected. The counter is driven by the external input signal EXT that has gone through polarity selection, frequency division and filtering. External clock mode B is equivalent to external clock mode A with EXT signal as the TRGIN.

To use external clock mode A, follow the configuration steps as below:

- Configure the external clock source TRGIN.

When TMRx\_CH1 is selected as the TRGIN, configure the channel 1 input filter (by setting the C1DF[3:0] bit in the TMRx\_CM1 register) and channel 1 input polarity (by setting the C1P/C1CP in the TMRx\_CCTRL register).

When TMRx\_CH2 is selected as the TRGIN, configure the channel 2 input filter (by setting the C2DF[3:0] bit in the TMRx\_CM1 register) and channel 1 input polarity (by setting the C2P/C2CP in the TMRx\_CCTRL register).

When TMRx\_EXT is selected as the TRGIN, configure the external signal polarity (by setting the ESP bit in the TMRx\_STCTRL register), external signal division (by setting the ESDIV[1:0] bit in the

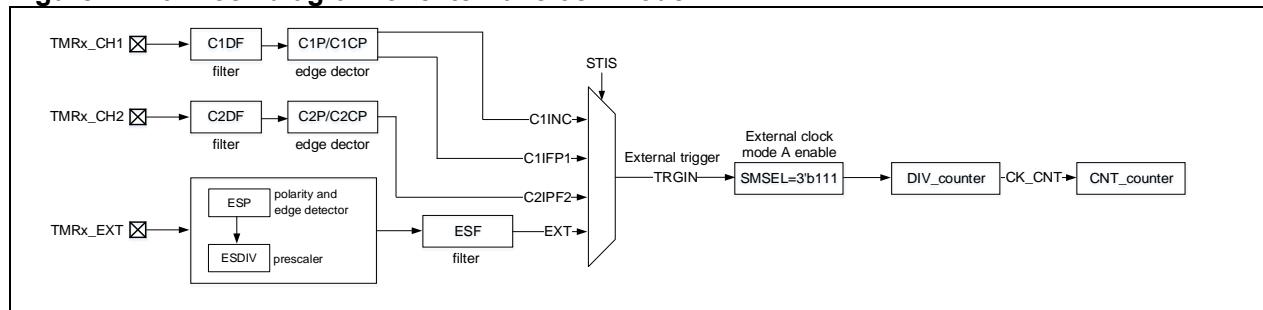
TMRx\_STCTRL register) and external signal filter (by setting the ESF[3:0] bit in the TMRx\_STCTRL register).

- Set the TRGIN signal source by setting the STIS[1:0] bit in the TMRx\_STCTRL register.
- Enable external clock mode A by setting SMSEL=3'b111 in the TMRx\_STCTRL register.
- Set counter counting frequency by setting the DIV[15:0] bit in the TMRx\_DIV register.
- Set counter counting period by setting the PR[15:0] bit in the TMRx\_PR register.
- Enable counter by setting the TMREN bit in the TMRx\_CTRL1 register.

To use external clock mode B, follow the configuration steps as below:

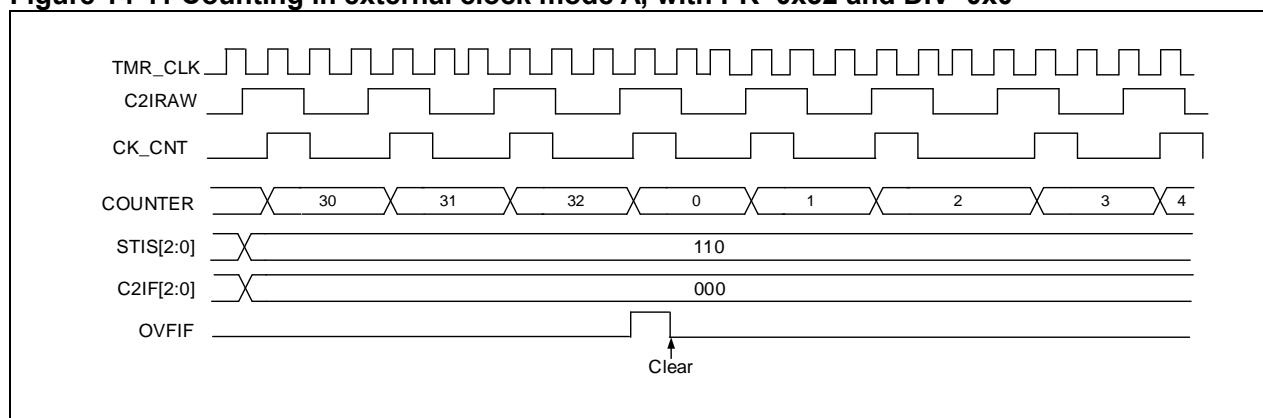
- Set external signal polarity by setting the ESP bit in the TMRx\_STCTRL register.
- Set external signal frequency division by setting the ESDIV[1:0] bit in the TMRx\_STCTRL register.
- Set external signal filter by setting the ESF[3:0] bit in the TMRx\_STCTRL register.
- Enable external clock mode B by setting the ECMBEN bit in the TMRx\_STCTRL register.
- Set counter counting frequency by setting the DIV[15:0] bit in the TMRx\_DIV register.
- Set counter counting period by setting the PR[15:0] bit in the TMRx\_PR register.
- Enable counter by setting the TMREN bit in the TMRx\_CTRL1 register.

**Figure 14-10 Block diagram of external clock mode A**

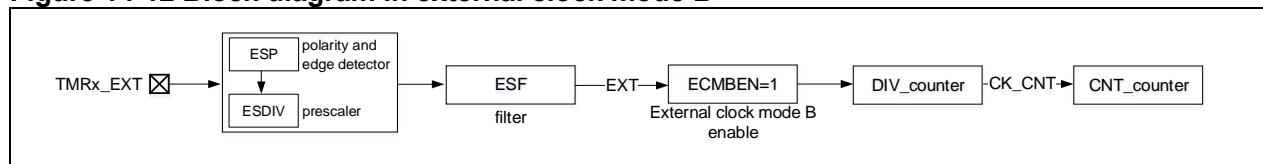


*Note: The delay is present between the input signal and the actual counter clock due to the synchronization circuit.*

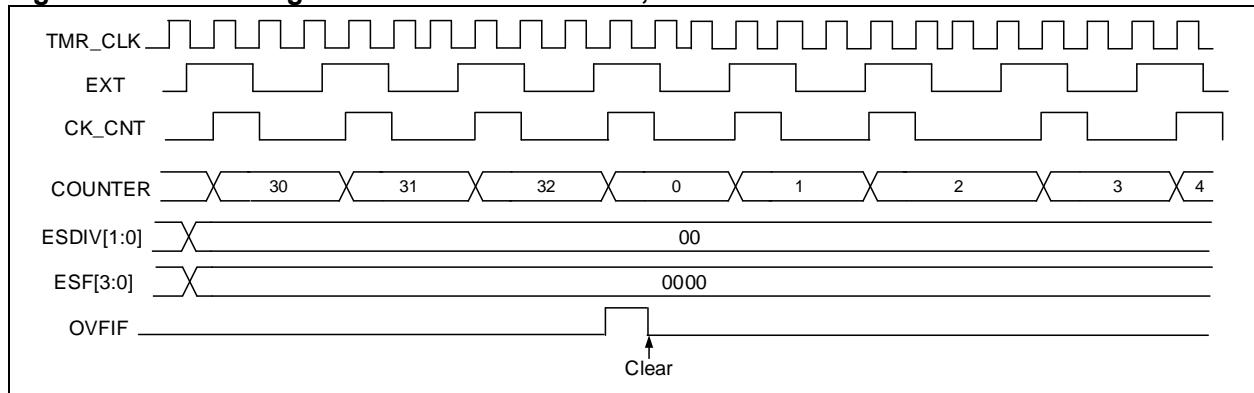
**Figure 14-11 Counting in external clock mode A, with PR=0x32 and DIV=0x0**



**Figure 14-12 Block diagram in external clock mode B**



*Note: The delay is present between the input EXT signal and the actual counter clock due to the synchronization circuit.*

**Figure 14-13 Counting in external clock mode B, with PR=0x32 and DIV=0x0****Internal trigger input (ISx)**

Timer synchronization allows interconnection between several timers. The TMR\_CLK of one timer can be provided by the TRGOUT signal output by another timer. Set the STIS[2: 0] bit to select internal trigger signal to enable counting.

The TMR3 timer consists of a 16-bit prescaler, which is used to generate the CK\_CNT that enables the counter to count. The frequency division relationship between the CK\_CNT and TMR\_CLK can be adjusted by setting the value of the TMR3\_DIV register. The prescaler value can be modified at any time, but it takes effect only when the next overflow event occurs.

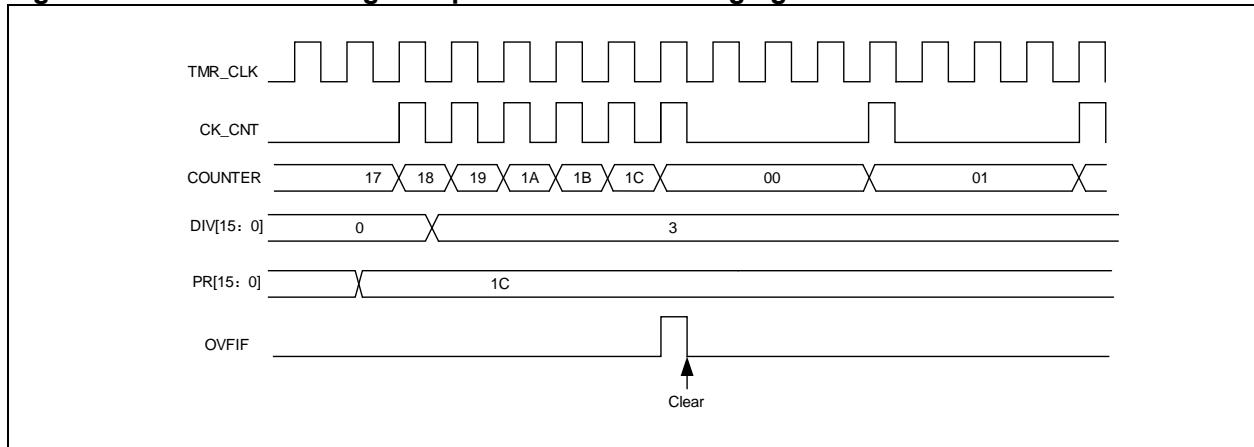
The internal trigger input configuration process is as follows:

- Set the TMRx\_PR register to set counting period.
- Set the TMRx\_DIV register to set counting frequency.
- Set the TWCSEL[1:0] bit in the TMRx\_CTRL1 register to set counting mode.
- Set the STIS[2:0] bit (range: 3'b000~3'b011) in the TMRx\_STCTRL register to select internal trigger.
- Set SMSEL[2:0]=3'b111 in the TMRx\_STCTRL register to select external clock mode A.
- Set the TMREN bit in the TMRx\_CTRL1 register to enable TMRx counter.

**Table 14-3 TMR3 internal trigger connection**

Slave timer	IS0 (STIS = 000)	IS1 (STIS = 001)	IS2 (STIS = 010)	IS3 (STIS = 011)
TMR3	TMR1	-	TMR15	-

*Note 1: If there is no corresponding timer in a device, the corresponding trigger signal ISx is not present.*

**Figure 14-14 Counter timing with prescaler value changing from 1 to 4**

### 14.2.3.2 Counting mode

The TMR3 timer supports several counting modes to meet different application scenarios. It has an internal 16-bit up, down, up/down counter.

The TMRx\_PR register is used to configure the counting period. The value in the TMRx\_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx\_PR register is transferred to the shadow register at an overflow event.

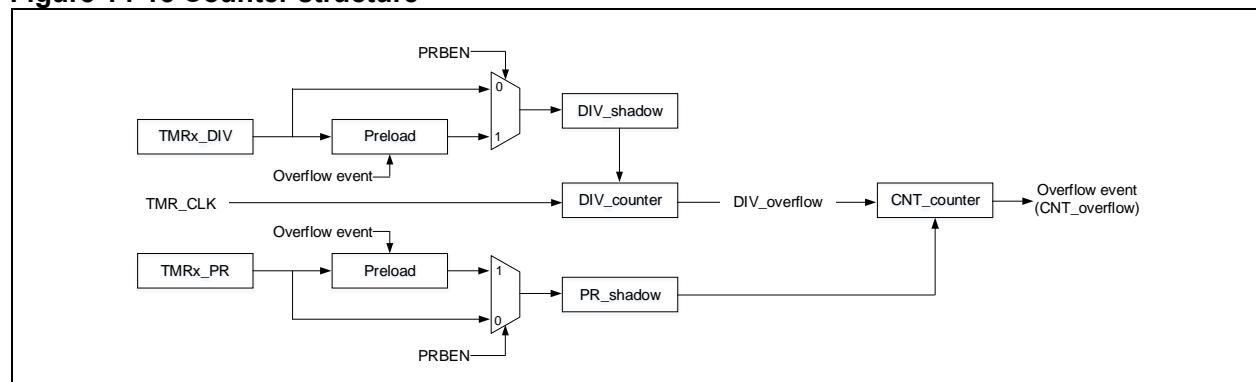
The TMRx\_DIV register is used to configure the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to the TMRx\_PR register, when the periodic buffer is enabled, the value in the TMRx\_DIV register is updated to the shadow register at an overflow event.

Reading the TMRx\_CNT register returns to the current counter value, and writing to the TMRx\_CNT register updates the current counter value to the value being written.

An overflow event is generated by default. Set OVFEN=1 in the TMRx\_CTRL1 to disable generation of update events. The OVFS bit in the TMRx\_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only counter overflow/underflow triggers an overflow event.

Setting the TMREN bit (TMREN=1) enables the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR\_EN is set 1 clock cycle after the TMREN is set.

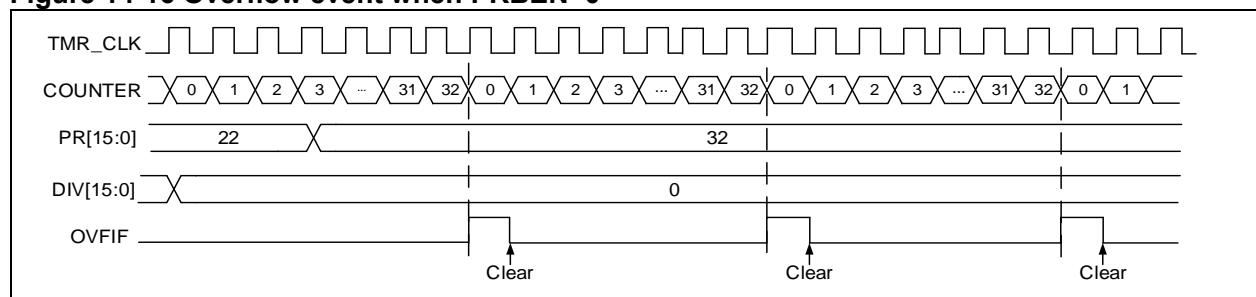
**Figure 14-15 Counter structure**

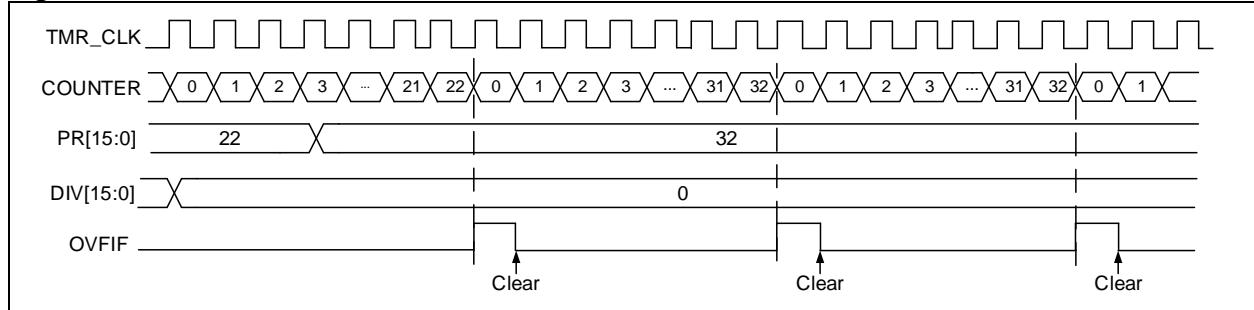


#### Upcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx\_CTRL1 register to enable upcounting mode. In upcounting mode, the counter counts from 0 to the value programmed in the TMRx\_PR register, restarts from 0, and generates a counter overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the register is no longer reloaded with the preload and re-loaded value after counter overflow occurs, otherwise, the prescaler and re-loaded value will be updated at an overflow event.

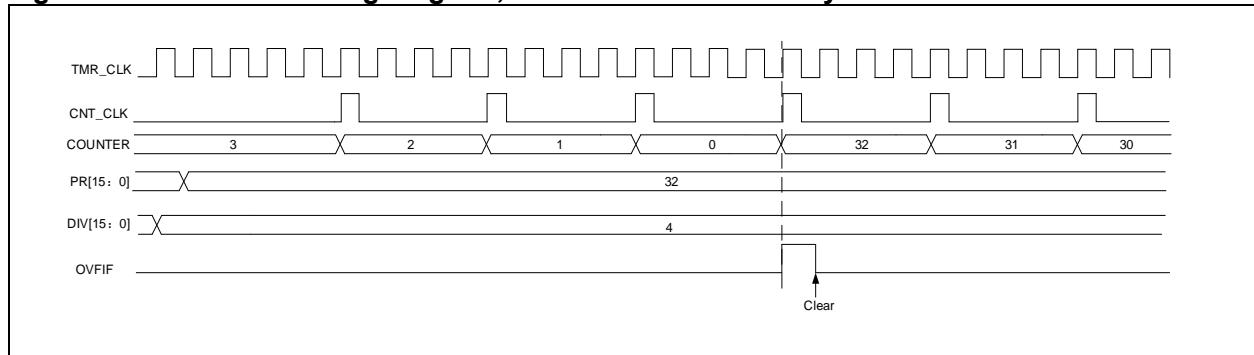
**Figure 14-16 Overflow event when PRBEN=0**



**Figure 14-17 Overflow event when PRBEN=1**

### Downcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b1 in the TMRx\_CTRL1 register to enable downcounting mode. In downcounting mode, the counter counts from the value programmed in the TMRx\_PR register down to 0, and restarts from the value programmed, and generates a counter underflow event.

**Figure 14-18 Counter timing diagram, internal clock divided by 4**

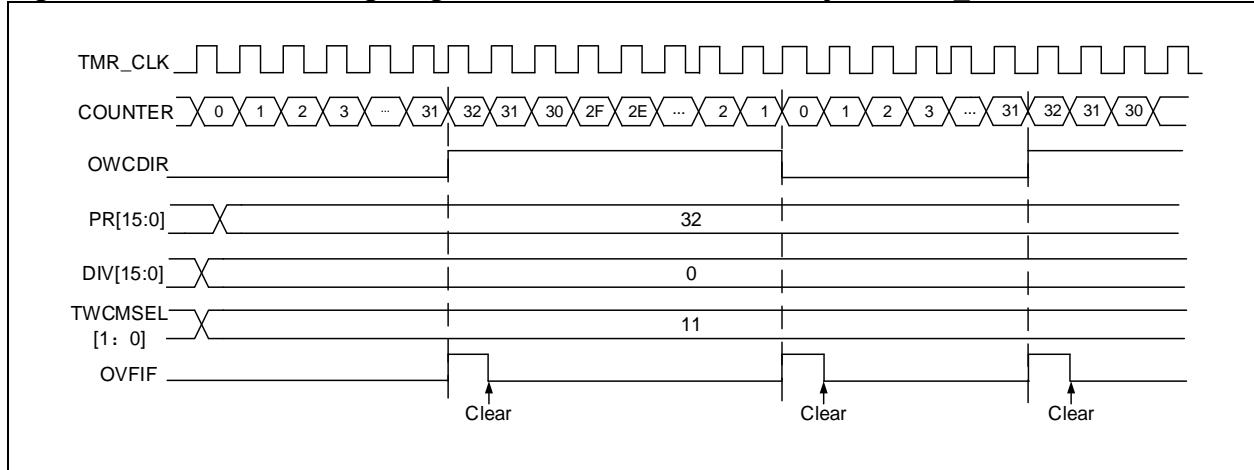
### Up/down counting mode

Set CMSEL[1:0]≠2'b00 in the TMRx\_CTRL1 register to enable up/down counting mode. In up/down counting mode, the counter counts up/down alternatively. When the counter counts from the value programmed in the TMRx\_PR register down to 1, an underflow event is generated, and then restarts counting from 0; When the counter counts from 0 to the value of the TMRx\_PR register -1, an overflow event is generated, and then restarts counting from the value of the TMRx\_PR register. The OWCDIR bit indicates the current counting direction.

The TWCMSEL[1:0] bit in the TMRx\_CTRL1 register is also used to select the CxIF flag setting method in up/down counting mode. In up/down counting mode 1 (TWCMSEL[1:0]=2'b01), CxIF flag can only be set when the counter counts down; in up/down counting mode 2 (TWCMSEL[1:0]=2'b10), CxIF flag can only be set when the counter counts up; in up/down counting mode 3 (TWCMSEL[1:0]=2'b11), CxIF flag can be set when the counter counts up/down.

*Note: The OWCDIR is ready-only in up/down counting mode.*

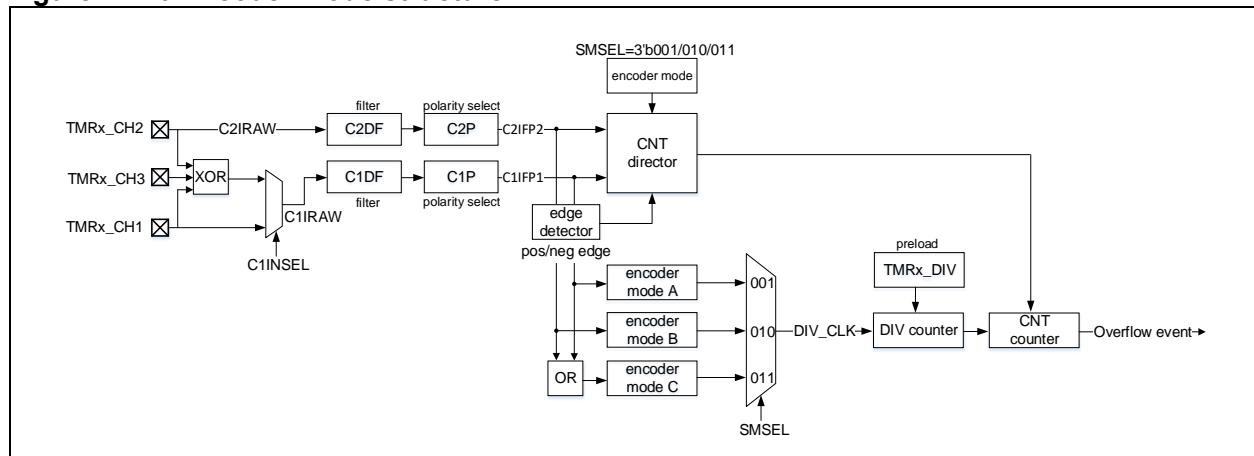
Figure 14-19 Counter timing diagram, internal clock divided by 1, TMRx\_PR=0x32



### Encoder interface mode

To enable the encoder interface mode, write SMSEL[2: 0]= 3'b001/3'b010/3'b011. In this mode, the two inputs (C1IN/C2IN) are required. Depending on the level on one input, the counter counts up or down on the edge of the other input. The OWCDIR bit indicates the direction of the counter, as shown in the table below:

Figure 14-20 Encoder mode structure



Encoder mode A: SMSEL=3'b001. The counter counts on C1IP1 (rising edge and falling edge), and the counting direction is dependent on the edge direction of C1IP1 and the level of C2IP2.

Encoder mode B: SMSEL=3'b010, the counter counts on C2IP2 (rising edge and falling edge), and the counting direction is dependent on the edge direction of C2IP2 and the level of C1IP1.

Encoder mode C: SMSEL=3'b011, the counter counts on C1IP1 and C2IP2 (rising edge and falling edge), and the counting direction is dependent on the C1IP1 edge direction + C2IP2 level, and C2IP2 edge direction + C1IP1 level.

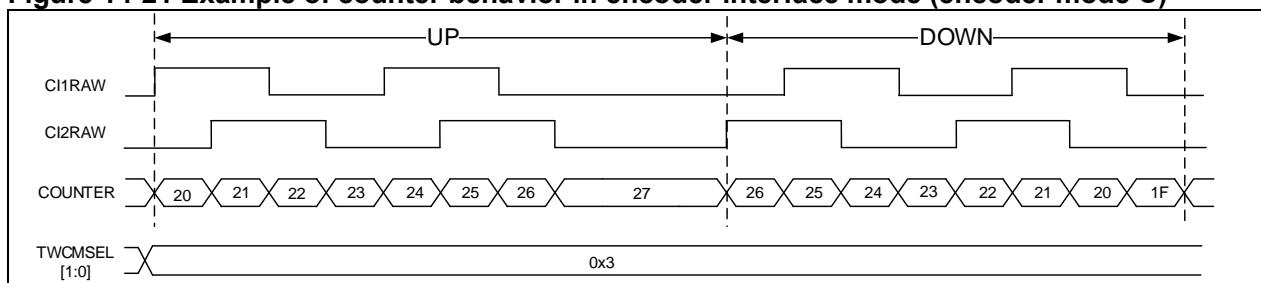
To use the encoder mode, follow the configuration steps as below:

- Set the C1DF[3:0] bit in the TMRx\_CM1 register to set channel 1 input signal filtering; set the C1P bit in the TMRx\_CCTRL register to set channel 1 input signal active level.
- Set the C2DF[3:0] bit in the TMRx\_CM1 register to set channel 2 input signal filtering; set the C2P bit in the TMRx\_CCTRL register to set channel 2 input signal active signal.
- Set the C1C[1:0] bit in the TMRx\_CM1 register to set channel 1 as input mode; set the C2C[1:0] bit in the TMRx\_CM1 register to set channel 2 as input mode.
- Set the SMSEL[2:0] bit in the TMRx\_STCTRL register to select encoder mode A (SMSEL=3'b001), encoder mode B (SMSEL=3'b010) or encoder mode C (SMSEL=3'b011).
- Set the PR[15:0] bit in the TMRx\_PR register to set the counting period.

- Set the DIV[15:0] bit in the TMRx\_DIV register to set the counting frequency.
- Set the IOs corresponding to TMRx\_CH1 and TMRx\_CH2 as multiplexed mode.
- Set the TMREN bit in the TMRx\_CTRL1 register to enable the counter.

**Table 14-4 Counting direction versus encoder signals**

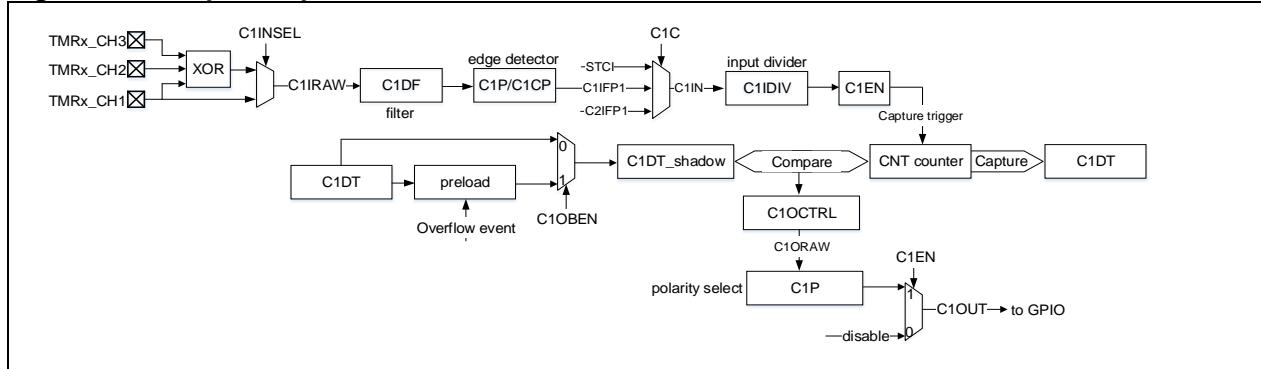
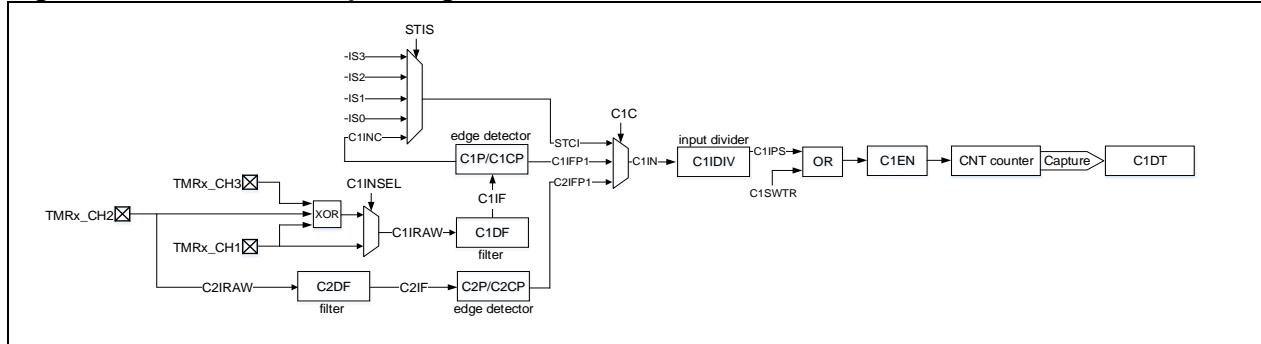
Active edge	Level on opposite signal (C1INFP1 to C2IN, C2INFP2 to C1IN)	C1INFP1 signal		C2INFP2 signal	
		Rising	Falling	Rising	Falling
Count on C1IN only	High	Down	Up	No count	No count
	Low	Up	Down	No count	No count
Count on C2IN only	High	No count	No count	Up	Down
	Low	No count	No count	Down	Up
Count on both C1IN and C2IN	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

**Figure 14-21 Example of counter behavior in encoder interface mode (encoder mode C)**

### 14.2.3.3 TMR input function

The TMR3 has four independent channels, each of which can be configured as input or output. As input, each channel input signal is processed as below:

- TMRx\_CHx outputs the pre-processed CxIRAW. Set the C1INSEL bit to select the source of C1IRAW from TMRx\_CH1 or the XOR-ed TMRx\_CH1, TMRx\_CH2 and TMRx\_CH3, and the sources of C2IRAW, C3IRAW and C4IRAW are TMRx\_CH2, TMRx\_CH3 and TMRx\_CH4, respectively.
- CxIRAW inputs digital filter and outputs a filtered signal CxIF. Set the sampling frequency and sampling times of digital filter by setting the CxDf bit.
- CxIF inputs edge detector and outputs the signal CxIFPx after edge selection. The edge selection is controlled by CxP and CxCP bits, and can be selected as rising edge, falling edge or both edges active.
- CxIFPx inputs capture signal selector and then outputs the signal CxIN after selection. The capture signal selector is controlled by the CxC bits. The source of CxIN can be set as CxIFPx, CyIFPx or STCI. The CyIFPx ( $x \neq y$ ) is the CyIFPy from channel y and handled by channel x edge detector (for example, the C1IFP2 is the C1IFP1 from channel 1 and then handled by channel 2 edge detector), and STCI derives from the slave timer controller, and its source is selected by setting the STIS bit.
- CxIN outputs the signal CxIPS that is divided by the input channel divider. The division factor is set to "No division", "divided by 2", "divided by 4" or "divided by 8" by setting the CxIDIV bit.

**Figure 14-22 Input/output channel 1 main circuit****Figure 14-23 Channel 1 input stage**

### Input mode

In input mode, the TMRx\_CxDT register latch the current counter values after the selected trigger signal is detected, and the capture compare interrupt flag bit (CxIF) is set to 1. An interrupt or a DMA request will be generated if the CxIEN and CxDEN bits are enabled. If the trigger signal is detected when the CxIF is set to 1, a capture overflow event occurs. The TMRx\_CxDT register overwrites the recorded value with the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, following the procedure below:

- Set C1C=01 in the TMR3\_CM1 register to select the C1IN as channel 1 input
- Set the filter bandwidth of C1IN signal (CxDF[3: 0])
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMR3\_CCTRL register
- Program the capture frequency division of C1IN signal (C1DIV[1: 0])
- Enable channel 1 input capture (C1EN=1)
- If needed, enable the relevant interrupt or DMA request by setting the C1IEN bit in the TMR3\_IDEN register or the C1DEN bit in the TMR3\_IDEN register

### Timer Input XOR function

The 3 timer input pins (TMR3\_CH1, TMR3\_CH2 and TMR3\_CH3) are connected to the channel 1 (selected by setting the C1INSEL in the TMR3\_CTRL2 register) through an XOR gate.

The XOR gate can be used to connect Hall sensors. For instance, connect the three XOR inputs to the three Hall sensors respectively so as to calculate the position and speed of the rotation by analyzing three Hall sensor signals.

### PWM input

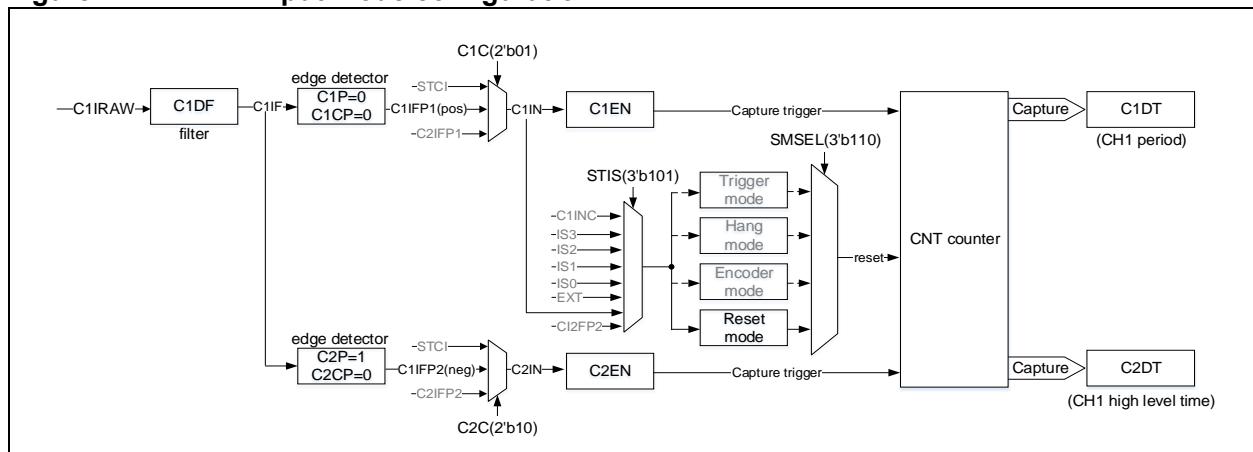
The PWM input mode applies to channel 1 and channel 2. To enable this mode, map the C1IN and C2IN to the same TMRx\_CHx, and configure the CxIFPx of channel 1/2 to trigger slave timer controller reset.

The PWM input mode can be used to measure the period and duty cycle of input signal. The period and duty cycle of channel 1 can be measured as follows:

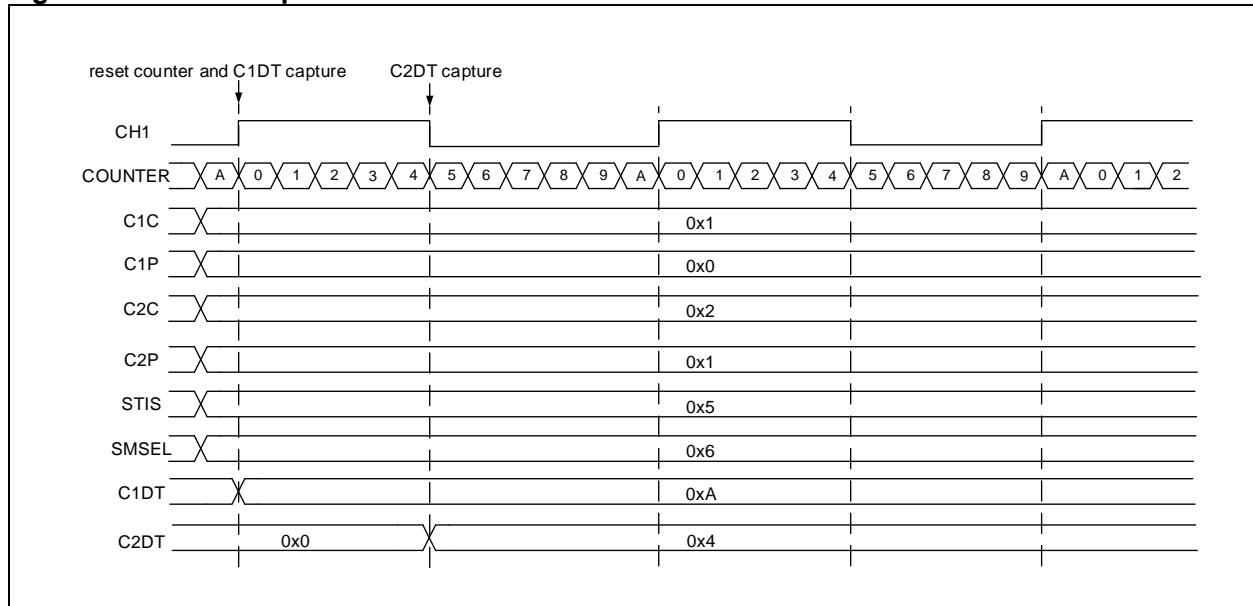
- Set C1C=2'b01 to set C1IN as C1IFP1;
- Set C1P=1'b0 to set C1IFP1 rising edge active;
- Set C2C=2'b10 to set C2IN as C1IFP2;
- Set C2P=1'b1 to set C1IFP2 falling edge active;
- Set STIS=3'b101 to set C1IFP1 as the slave timer trigger signal;
- Set SMSEL=3'b100 to set the slave timer in reset mode;
- Set C1EN=1'b1 and C2EN=1'b1 to enable channel 1 and input capture.

In these configurations, the rising edge of channel 1 input signal triggers capture and saves captured values to the C1DT register, and channel 1 input signal rising edge resets the counter. The falling edge of channel 1 input signal triggers capture and saves captured values to the C2DT register. The period and duty of channel 1 input signal can be calculated through C1DT and C2DT respectively.

**Figure 14-24 PWM input mode configuration**



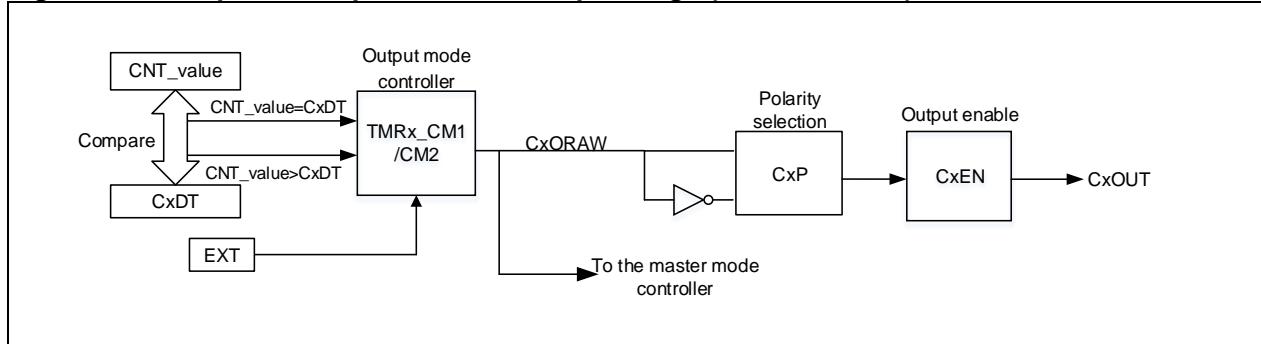
**Figure 14-25 PWM input mode**



#### 14.2.3.4 TMR output function

The TMR output consists of a comparator and an output controller. It is used to program the period, duty cycle and polarity of the output signal.

Figure 14-26 Capture/compare channel output stage (channel 1 to 4)



### Output mode

Write  $CxC[2:0] \neq 2'b00$  to configure the channel as output to implement multiple output modes. In this case, the counter value is compared with the value in the  $TMR3_CxDT$  register, and the intermediate signal  $CxOARAW$  is generated according to the output mode selected by  $CxOCTRL[2:0]$ , which is sent to IO after being processed by the output control circuit. The period of the output signal is configured by the  $TMR3_PR$  register, while the duty cycle by the  $TMR3_CxDT$  register.

Output compare modes include:

- **PWM mode A:** Set  $CxOCTRL=3'b110$  to enable PWM mode A. In upcounting, when  $TMRx_C1DT > TMRx_CVAL$ ,  $C1ORAW$  outputs high; otherwise, outputs low. In downcounting, when  $TMRx_C1DT < TMRx_CVAL$ ,  $C1ORAW$  outputs low; otherwise, outputs high. To set PWM mode A, the following process is recommended:
  - Set the  $TMRx_PR$  register to set PWM period;
  - Set the  $TMRx_CxDT$  register to set PWM duty cycle;
  - Set  $CxOCTRL=3'b110$  in the  $TMRx_CM1/CM2$  register to set output mode as PWM mode A;
  - Set the  $TMRx_DIV$  register to set the counting frequency;
  - Set the  $TWCMSEL[1:0]$  bit in the  $TMRx_CTRL1$  register to set the count mode;
  - Set  $CxP$  bit and  $CxCP$  bit in the  $TMRx_CCTRL$  register to set output polarity;
  - Set  $CxEN$  bit and  $CxCEN$  bit in the  $TMRx_CCTRL$  register to enable channel output;
  - Set the  $OEN$  bit in the  $TMRx_BRK$  register to enable  $TMRx$  output;
  - Set the corresponding GPIO of  $TMR$  output channel as the multiplexed mode;
  - Set the  $TMREN$  bit in the  $TMRx_CTRL1$  register to enable  $TMRx$  counter.
- **PWM mode B:** Set  $CxOCTRL=3'b111$  to enable PWM mode B. In upcounting, when  $TMRx_C1DT > TMRx_CVAL$ ,  $C1ORAW$  outputs low; otherwise, outputs high. In downcounting, when  $TMRx_C1DT < TMRx_CVAL$ ,  $C1ORAW$  outputs high; otherwise, outputs low.
- **Forced output mode:** Set  $CxOCTRL=3'b100/101$  to enable forced output mode. In this case, the  $CxOARAW$  is forced to be the programmed level, irrespective of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.
- **Output compare mode:** Set  $CxOCTRL=3'b001/010/011$  to enable output compare mode. In this case, when the counter value matches the value of the  $CxDT$  register, the  $CxOARAW$  is forced high ( $CxOCTRL=3'b001$ ), low ( $CxOCTRL=3'b010$ ) or toggling ( $CxOCTRL=3'b011$ ).
- **One-pulse mode:** This is a particular case of PWM mode. Set  $OCMEN=1$  to enable one-pulse mode. In this mode, the comparison match is performed in the current counting period. The  $TMREN$  bit is cleared as soon as the current counting is completed. Therefore, only one pulse is output. When configured as in upcounting mode, the configuration must follow the rule:  $CVAL < CxDT \leq PR$ ; in downcounting mode,  $CVAL > CxDT$  is required.
- **Fast output mode:** Set  $CxOIEN=1$  to enable this mode. If enabled, the  $CxOARAW$  signal will not change when the counter value matches the  $CxDT$ , but at the beginning of the current counting period. In other words, the comparison result is advanced, so the comparison result between the counter value and the  $TMR3_CxDT$  register will determine the level of  $CxOARAW$  in advance.

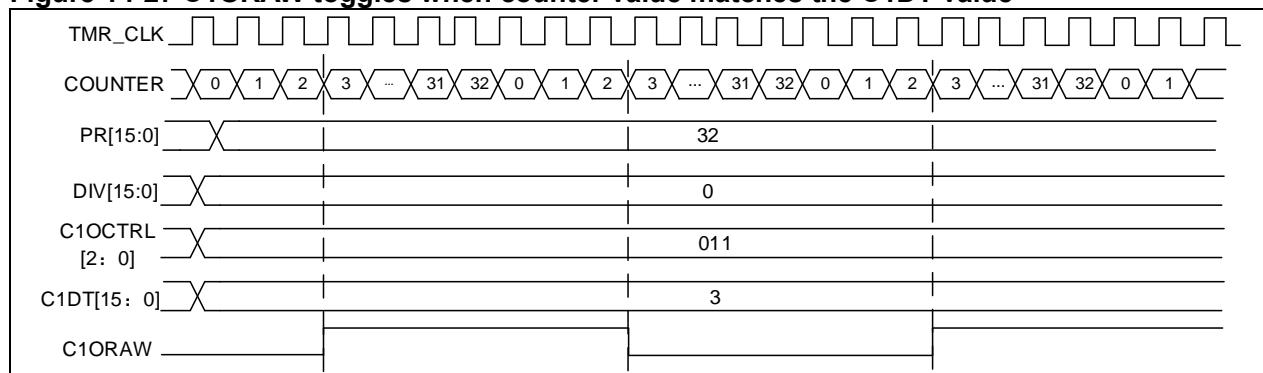
[Figure 14-27](#) gives an example of output compare mode (toggle) with  $C1DT=0x3$ . When the counter value is equal to  $0x3$ ,  $C1OUT$  toggles.

[Figure 14-28](#) gives an example of the combination between upcounting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

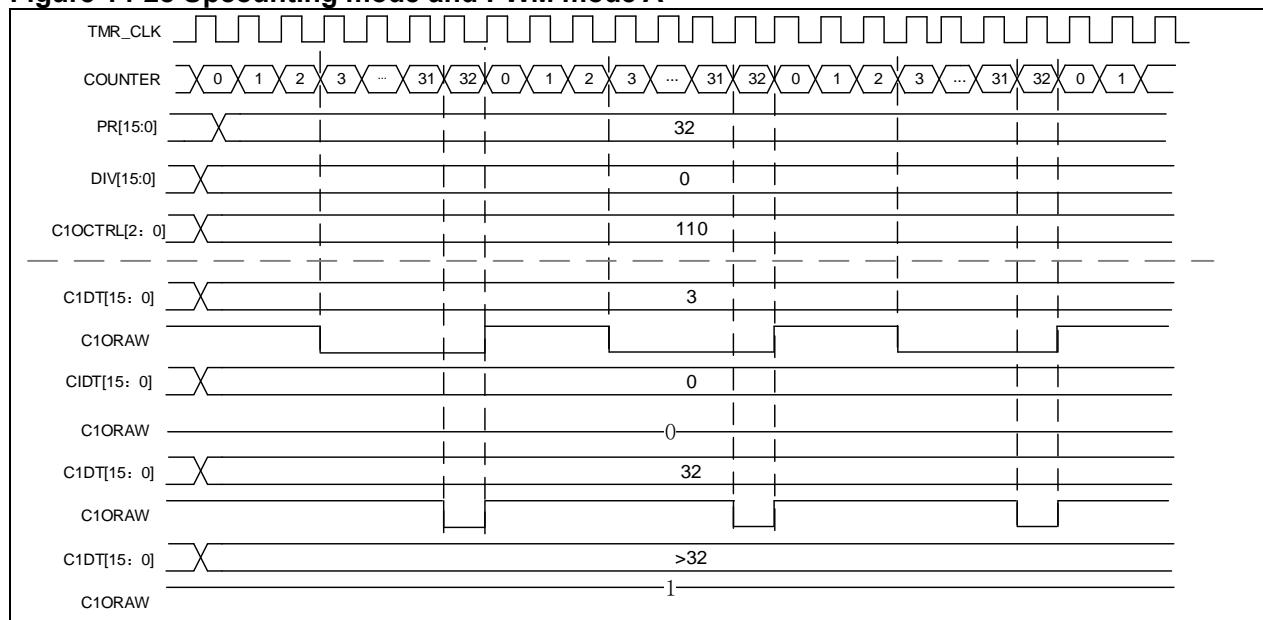
[Figure 14-29](#) gives an example of the combination between up/down counting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

[Figure 14-30](#) gives an example of the combination between upcounting mode and one-pulse PWM mode B. The counter only counts only one cycle, and the output signal sends only one pulse.

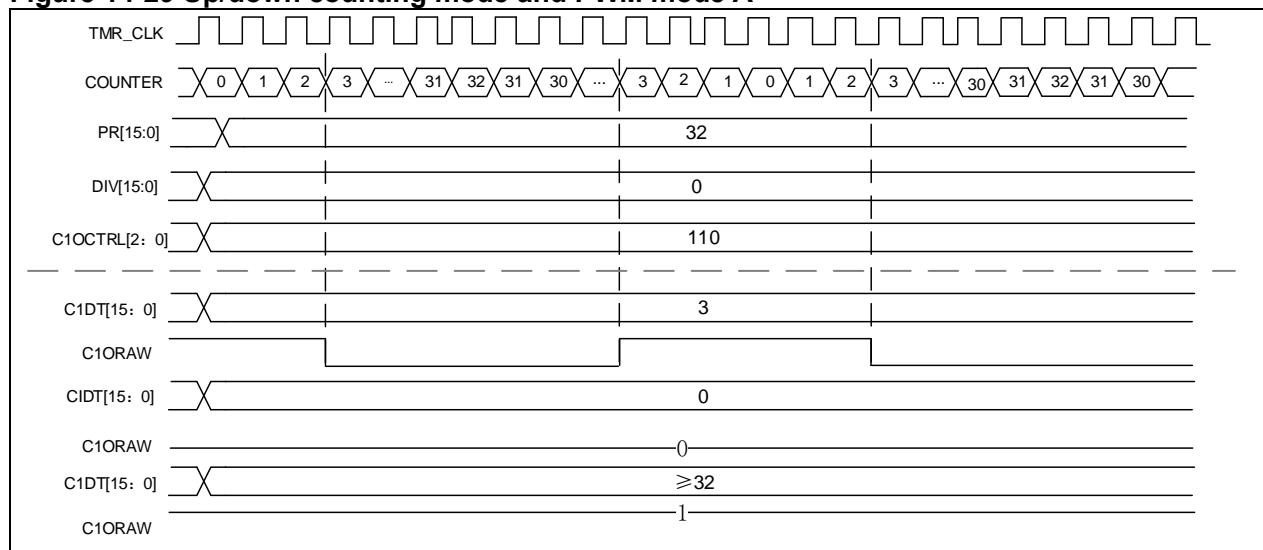
**Figure 14-27 C1ORAW toggles when counter value matches the C1DT value**

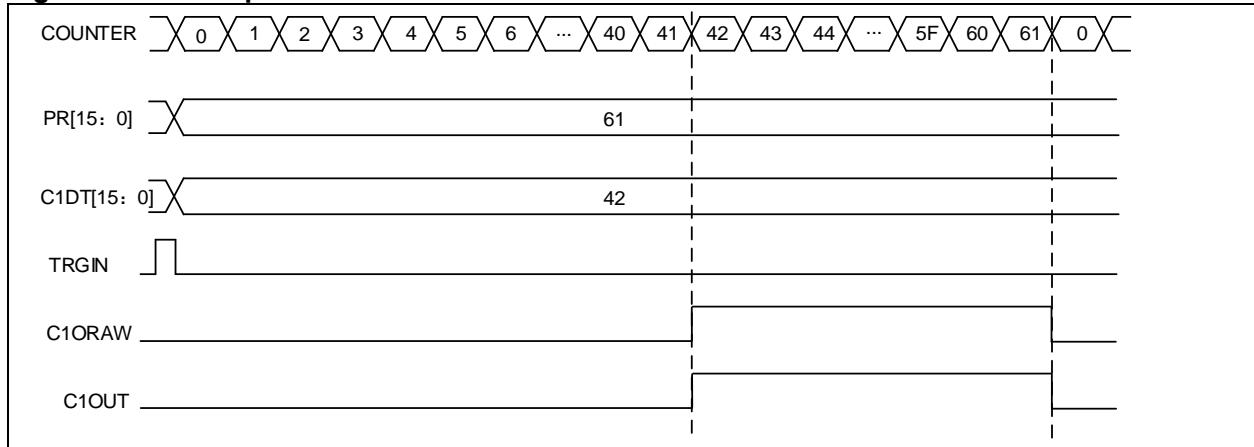


**Figure 14-28 Upcounting mode and PWM mode A**



**Figure 14-29 Up/down counting mode and PWM mode A**



**Figure 14-30 One-pulse mode**

### Master timer event output

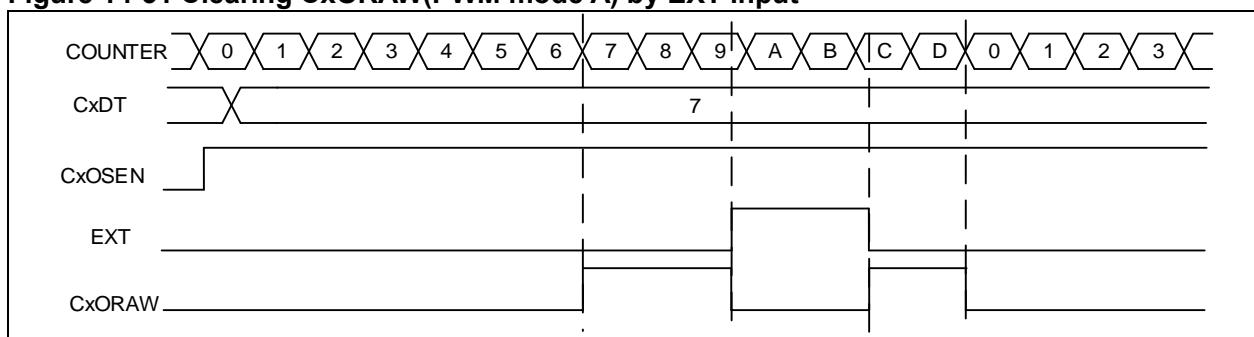
When TMR is selected as the master timer, the following signal sources can be selected as TRGOUT signal to output to the slave timer, by setting the PTOS bit in the TMRx\_CTRL2 register.

- PTOS=3'b000, TRGOUT outputs software overflow event (OVFSWTR bit in the TMRx\_SWEVT register).
- PTOS=3'b001, TRGOUT outputs counter enable signal.
- PTOS=3'b010, TRGOUT outputs counter overflow event.
- PTOS=3'b011, TRGOUT outputs capture and compare event.
- PTOS=3'b100, TRGOUT outputs C1ORAW signal.
- PTOS=3'b101, TRGOUT outputs C2ORAW signal.
- PTOS=3'b110, TRGOUT outputs C3ORAW signal.
- PTOS=3'b111, TRGOUT outputs C4ORAW signal.

### CxORAW clear

When the CxOSEN bit is set to 1, the CxORAW signal for a given channel is cleared by applying a high level to the EXT input. The CxORAW signal remains unchanged until the next overflow event.

This function can only be used in output capture or PWM modes, and does not work in forced output mode. **Figure 14-31** shows the example of clearing CxORAW signal. When the EXT input is high, the CxORAW signal, which was originally high, is driven low; when the EXT is low, the CxORAW signal outputs the corresponding level according to the comparison result between the counter value and CxDT value.

**Figure 14-31 Clearing CxORAW(PWM mode A) by EXT input**

### 14.2.3.5 TMR synchronization

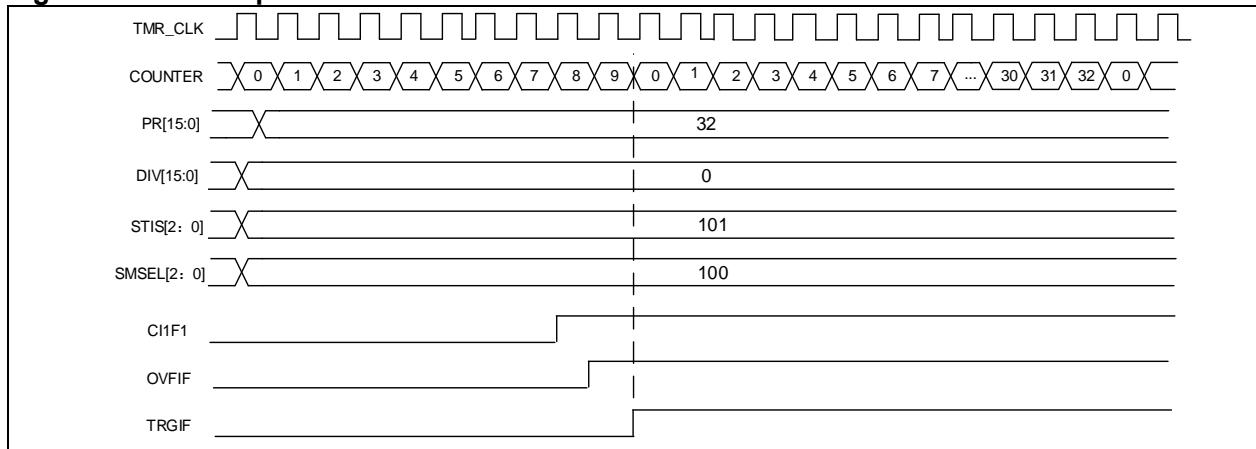
The timers are linked together internally for timer synchronization. Master timer is selected by setting the PTOS[2: 0] bit; Slave timer is selected by setting the SMSEL[2: 0] bit.

Slave mode includes:

#### Slave mode: Reset mode

The counter and its prescaler can be reset by a selected trigger signal. An overflow event is generated when OVFS=0.

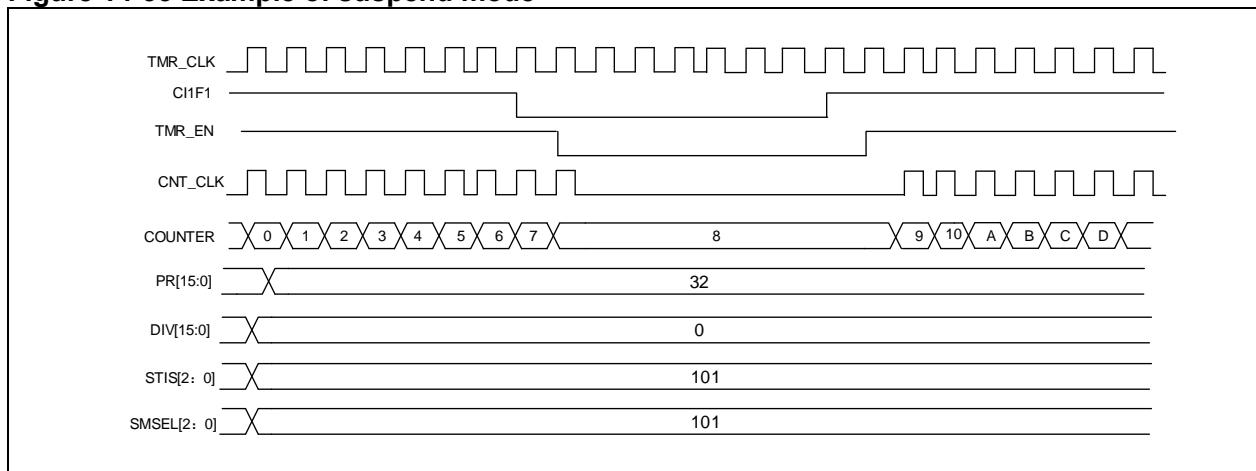
**Figure 14-32 Example of reset mode**



#### Slave mode: Suspend mode

In this mode, the counter is controlled by a selected trigger input. The counter starts counting when the trigger input is high and stops as soon as the trigger input is low.

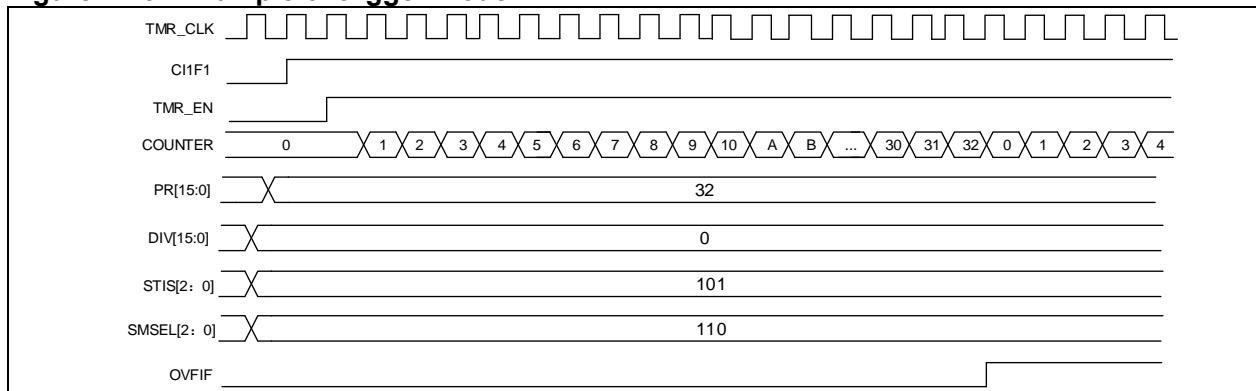
**Figure 14-33 Example of suspend mode**



#### Slave mode: Trigger mode

The counter can start counting on the rising edge of a selected trigger input (TMR\_EN=1)

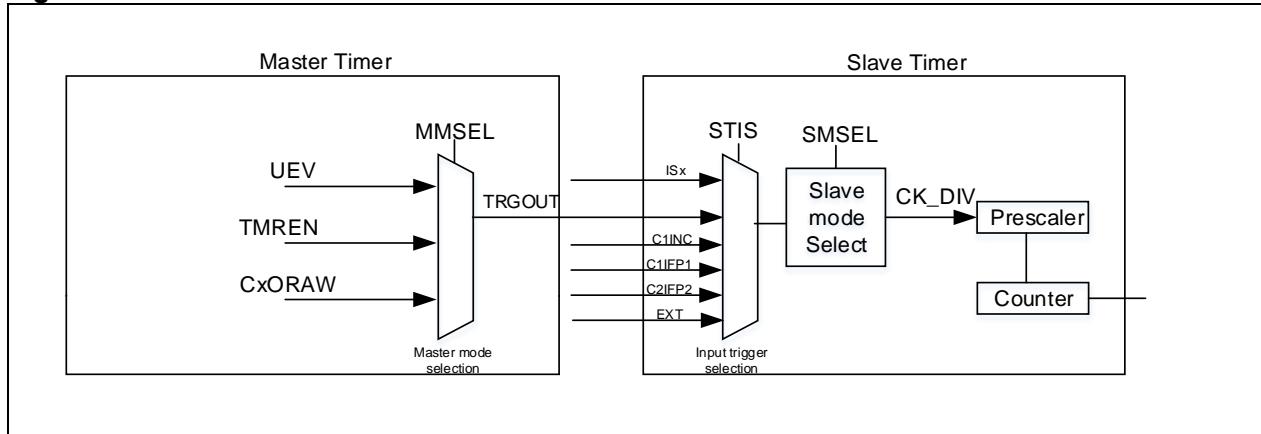
**Figure 14-34 Example of trigger mode**



### Master/slave timer interconnection

Both Master and slave timer can be configured in different master and slave modes respectively. The combination of both them can be used for various purposes. [Figure 14-35](#) provides an example of interconnection between master timer and slave timer.

**Figure 14-35 Master/slave timer connection**



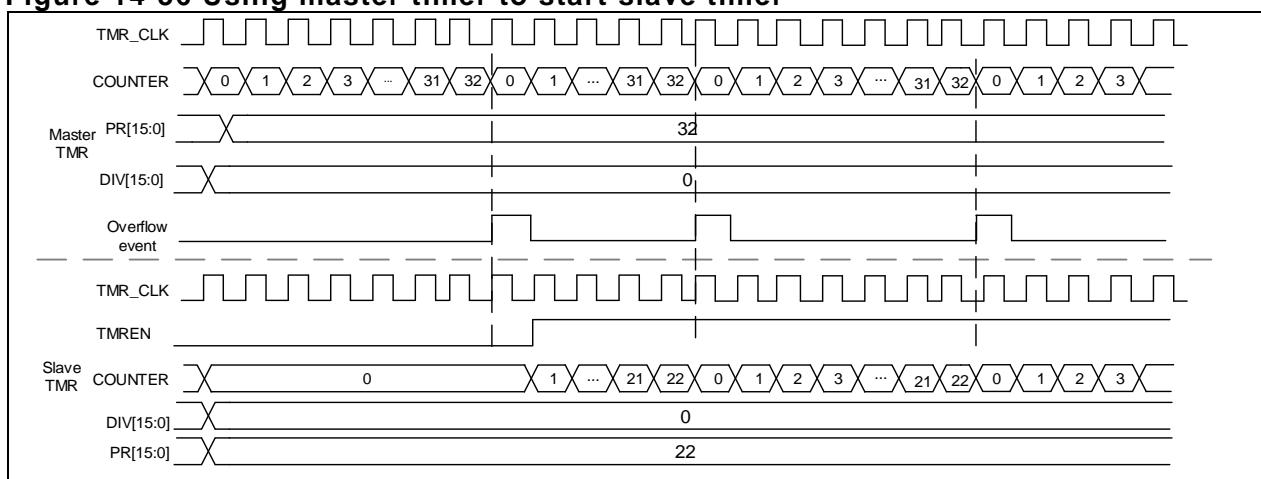
Using master timer to clock the slave timer:

- Configure master timer output signal TRGOUT as an overflow event (PTOS[2: 0]=3'b010). The master timer outputs a pulse signal at each counter overflow event, which is used as the counting clock of the slave timer.
- Configure the master timer counting period (TMRx\_PR register)
- Configure the slave timer trigger input signal TRGIN as master timer output (STIS[2: 0] in the TMRx\_STCTRL register)
- Configure the slave timer to use external clock mode A (SMSEL[2: 0]=3'b111 in the TMRx\_STCTRL register )
- Set TMREN =1 in both master timer and slave timer to enable them

Using master timer to start slave timer:

- Configure master timer output signal TRGOUT as an overflow event (PTOS[2: 0]=3'b010). The master timer outputs a pulse signal at each counter overflow event, which is used as the counting clock of the slave timer.
- Configure master timer counting period (TMRx\_PR register)
- Configure slave timer trigger input signal TRGIN as master timer input
- Configure slave timer as trigger mode (SMSEL=3'b110 in the TMR2\_STCTRL register)
- Set TMREN=1 to enable master timer.

**Figure 14-36 Using master timer to start slave timer**

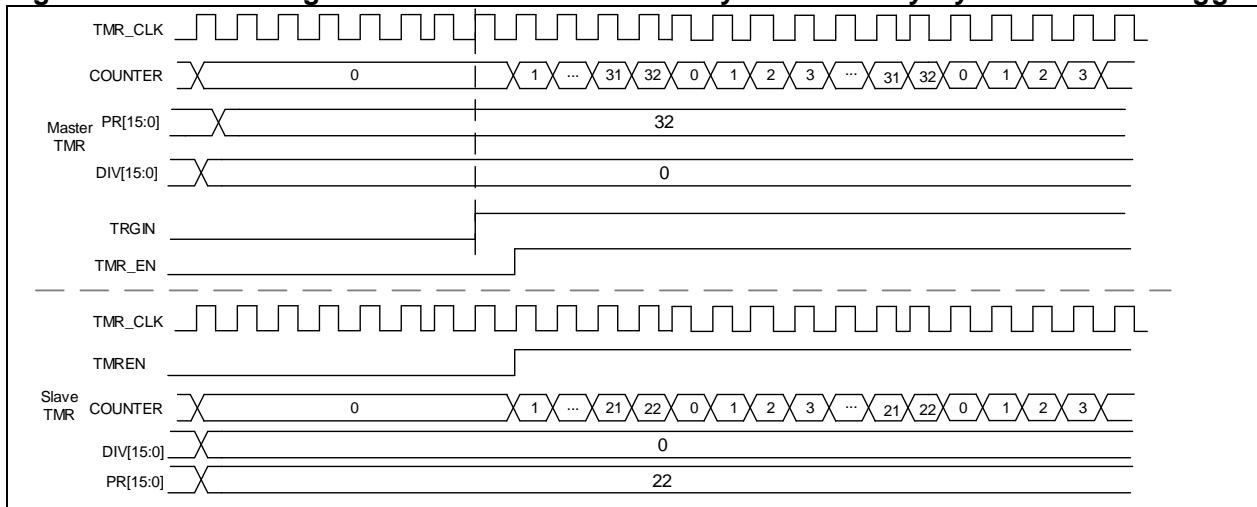


Starting master and slave timers synchronously by an external trigger:

In this example, configure the master timer as master/slave mode synchronously and enable its slave timer synchronization function. This mode is used for synchronization between master timer and slave timer.

- Set the STS bit of the master timer.
- Configure master timer output signal TRGOUT as an overflow event (PTOS[2: 0]=3'b010). The master timer outputs a pulse signal at each counter overflow event, which is used as the counting clock of the slave timer.
- Configure the slave timer mode of the master timer as trigger mode, and select C1IN as trigger source
- Configure slave timer trigger input signal TRGIN as master timer output
- Configure slave timer as trigger mode (SMSEL=3'b110 in the TMR2\_STCTRL register)

**Figure 14-37 Starting master and slave timers synchronously by an external trigger**



#### 14.2.3.6 Debug mode

When the microcontroller enters debug mode (Cortex™-M4 core halted), the TMR3 counter stops counting by setting the TMR3\_PAUSE bit in the DEBUG module.

#### 14.2.4 TMR3 registers

These peripheral registers must be accessed by word (32 bits).

TMR3 registers are mapped into a 16-bit addressable space.

**Table 14-5 TMR3 register map and reset value**

Register	Offset	Reset value
TMR3_CTRL1	0x00	0x0000
TMR3_CTRL2	0x04	0x0000
TMR3_STCTRL	0x08	0x0000
TMR3_IDEN	0x0C	0x0000
TMR3ISTS	0x10	0x0000
TMR3_SWEVT	0x14	0x0000
TMR3_CM1	0x18	0x0000
TMR3_CM2	0x1C	0x0000
TMR3_CCTRL	0x20	0x0000
TMR3_CVAL	0x24	0x0000 0000
TMR3_DIV	0x28	0x0000

TMR3_PR	0x2C	0x0000 0000
TMR3_C1DT	0x34	0x0000 0000
TMR3_C2DT	0x38	0x0000 0000
TMR3_C3DT	0x3C	0x0000 0000
TMR3_C4DT	0x40	0x0000 0000
TMR3_DMACTRL	0x48	0x0000
TMR3_DMADT	0x4C	0x0000

#### 14.2.4.1 Control register1 (TMR3\_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value
				Clock divider 00: Normal, $f_{DTS} = f_{CK\_INT}$ 01: Divided by 2, $f_{DTS} = f_{CK\_INT}/2$ 10: Divided by 4, $f_{DTS} = f_{CK\_INT}/4$ 11: Reserved
Bit 9: 8	CLKDIV	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 7	PRBEN	0x0	rw	Two-way counting mode selection 00: One-way counting mode, depending on the OWCDIR bit 01: Two-way counting mode1, count up and down alternately, the output flag bit is set only when the counter counts down 10: Two-way counting mode2, count up and down alternately, the output flag bit is set only when the counter counts up 11: Two-way counting mode3, count up and down alternately, the output flag bit is set when the counter counts up / down
Bit 6: 5	TWCMSEL	0x0	rw	One-way count direction 0: Up 1: Down
Bit 4	OWCDIR	0x0	rw	One cycle mode enable This bit is use to select whether to stop counting at an overflow event 0: The counter does not stop at an overflow event 1: The counter stops at an overflow event
Bit 3	OCMEN	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 2	OVFS	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 1	OVFEN	0x0	rw	TMR enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	

#### 14.2.4.2 Control register2 (TMR3\_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
				C1IN selection 0: CH1 pin is connected to C1IRAW input 1: The XOR result of CH1, CH2 and CH3 pins is connected to C1IRAW input
Bit 7	C1INSEL	0x0	rw	
				Master TMR output selection This field is used to select the TMRx signal sent to the slave timer. 000: Reset 001: Enable
Bit 6: 4	PTOS	0x0	rw	010: Update 011: Compare pulse 100: C1ORAW signal 101: C2ORAW signal 110: C3ORAW signal 111: C4ORAW signal
Bit 3	DRS	0x0	rw	DMA request source 0: Capture/compare event 1: Overflow event
Bit 2: 0	Reserved	0x0	resd	Kept at its default value.

#### 14.2.4.3 Slave timer control register (TMR3\_STCTRL)

Bit	Register	Reset value	Type	Description
Bit 15	ESP	0x0	rw	External signal polarity 0: High or rising edge 1: Low or falling edge
Bit 14	ECMBEN	0x0	rw	External clock mode B enable This bit is used to enable external clock mode B 0: Disabled 1: Enabled
Bit 13: 12	ESDIV	0x0	rw	External signal divide This field is used to select the frequency division of an external trigger 00: Normal 01: Divided by 2 10: Divided by 4 11: Divided by 8
Bit 11: 8	ESF	0x0	rw	External signal filter This field is used to filter an external signal. The external signal can be sampled only after it has been generated N times 0000: No filter, sampling by $f_{DTS}$ 0001: $f_{SAMPLING} = f_{CK\_INT}$ , N=2 0010: $f_{SAMPLING} = f_{CK\_INT}$ , N=4 0011: $f_{SAMPLING} = f_{CK\_INT}$ , N=8 0100: $f_{SAMPLING} = f_{DTS}/2$ , N=6 0101: $f_{SAMPLING} = f_{DTS}/2$ , N=8 0110: $f_{SAMPLING} = f_{DTS}/4$ , N=6 0111: $f_{SAMPLING} = f_{DTS}/4$ , N=8 1000: $f_{SAMPLING} = f_{DTS}/8$ , N=6 1001: $f_{SAMPLING} = f_{DTS}/8$ , N=8 1010: $f_{SAMPLING} = f_{DTS}/16$ , N=5 1011: $f_{SAMPLING} = f_{DTS}/16$ , N=6 1100: $f_{SAMPLING} = f_{DTS}/16$ , N=8 1101: $f_{SAMPLING} = f_{DTS}/32$ , N=5 1110: $f_{SAMPLING} = f_{DTS}/32$ , N=6 1111: $f_{SAMPLING} = f_{DTS}/32$ , N=8
Bit 7	STS	0x0	rw	Subordinate TMR synchronization If enabled, master and slave timer can be synchronized. 0: Disabled

				1: Enabled Subordinate TMR input selection This field is used to select the subordinate TMR input. 000: Internal selection 0 (IS0) 001: Internal selection 1 (IS1) 010: Internal selection 2 (IS2) 011: Internal selection 3 (IS3) 100: C1IRAW input detector (C1INC) 101: Filtered input 1 (C1IF1) 110: Filtered input 2 (C1IF2) 111: External input (EXT) Please refer to Table 14-3 and 14-5 for more information on ISx for each timer.
Bit 6: 4	STIS	0x0	rw	Kept at its default value Subordinate TMR mode selection 000: Slave mode is disabled 001: Encoder mode A 010: Encoder mode B 011: Encoder mode C 100: Reset mode — Rising edge of the TRGIN input reinitializes the counter 101: Suspend mode — The counter starts counting when the TRGIN is high 110: Trigger mode — A trigger event is generated at the rising edge of the TRGIN input 111: External clock mode A — Rising edge of the TRGIN input clocks the counter Note: Please refer to count mode section for the details on encoder mode A/B/C.
Bit 3	Reserved	0x0	resd	Kept at its default value Subordinate TMR mode selection 000: Slave mode is disabled 001: Encoder mode A 010: Encoder mode B 011: Encoder mode C 100: Reset mode — Rising edge of the TRGIN input reinitializes the counter 101: Suspend mode — The counter starts counting when the TRGIN is high 110: Trigger mode — A trigger event is generated at the rising edge of the TRGIN input 111: External clock mode A — Rising edge of the TRGIN input clocks the counter Note: Please refer to count mode section for the details on encoder mode A/B/C.
Bit 2: 0	SMSEL	0x0	rw	Kept at its default value Subordinate TMR mode selection 000: Slave mode is disabled 001: Encoder mode A 010: Encoder mode B 011: Encoder mode C 100: Reset mode — Rising edge of the TRGIN input reinitializes the counter 101: Suspend mode — The counter starts counting when the TRGIN is high 110: Trigger mode — A trigger event is generated at the rising edge of the TRGIN input 111: External clock mode A — Rising edge of the TRGIN input clocks the counter Note: Please refer to count mode section for the details on encoder mode A/B/C.

#### 14.2.4.4 DMA/interrupt enable register (TMR3\_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at its default value
Bit 14	TDEN	0x0	rw	Trigger DMA request enable 0: Disabled 1: Enabled
Bit 13	Reserved	0x0	resd	Kept at its default value
Bit 12	C4DEN	0x0	rw	Channel 4 DMA request enable 0: Disabled 1: Enabled
Bit 11	C3DEN	0x0	rw	Channel 3 DMA request enable 0: Disabled 1: Enabled
Bit 10	C2DEN	0x0	rw	Channel 2 DMA request enable 0: Disabled 1: Enabled
Bit 9	C1DEN	0x0	rw	Channel 1 DMA request enable 0: Disabled 1: Enabled
Bit 8	OVFDEN	0x0	rw	Overflow event DMA request enable 0: Disabled 1: Enabled
Bit 7	Reserved	0x0	resd	Kept at its default value
Bit 6	TIEN	0x0	rw	Trigger interrupt enable 0: Disabled 1: Enabled
Bit 5	Reserved	0x0	resd	Kept at its default value
Bit 4	C4IEN	0x0	rw	Channel 4 interrupt enable 0: Disabled 1: Enabled
Bit 3	C3IEN	0x0	rw	Channel 3 interrupt enable 0: Disabled 1: Enabled

Bit 2	C2IEN	0x0	rw	Channel 2 interrupt enable 0: Disabled 1: Enabled
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled 1: Enabled
Bit 0	OVFLEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

#### 14.2.4.5 Interrupt status register (TMR3\_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at its default value
Bit 12	C4RF	0x0	rw0c	Channel 4 recapture flag Please refer to C1RF description.
Bit 11	C3RF	0x0	rw0c	Channel 3 recapture flag Please refer to C1RF description.
Bit 10	C2RF	0x0	rw0c	Channel 2 recapture flag Please refer to C1RF description.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing "0". 0: No capture is detected 1: Capture is detected.
Bit 8: 7	Reserved	0x0	resd	Kept at its default value
Bit 6	TRGIF	0x0	rw0c	Trigger interrupt flag This bit is set by hardware on a trigger event. It is cleared by writing "0". 0: No trigger event occurs 1: Trigger event is generated. Trigger event: an active edge is detected on TRGIN input, or any edge in suspend mode.
Bit 5	Reserved	0x0	resd	Kept at its default value
Bit 4	C4IF	0x0	rw0c	Channel 4 interrupt flag Please refer to C1IF description.
Bit 3	C3IF	0x0	rw0c	Channel 3 interrupt flag Please refer to C1IF description.
Bit 2	C2IF	0x0	rw0c	Channel 2 interrupt flag Please refer to C1IF description.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT 0: No capture event occurs 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurs 1: Compare event is generated
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurs 1: Overflow event is generated. If OVFEN=0 and OVFS=0 in the TMRx_CTRL1 register: – An overflow event is generated when OVFG= 1 in the TMRx_SWEVE register; – An overflow event is generated when the counter CVAL is reinitialized by a trigger event.

#### 14.2.4.6 Software event register (TMR3\_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6	TRGSWTR	0x0	rw	Trigger event triggered by software This bit is set by software to generate a trigger event. 0: No effect 1: Generate a trigger event.
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	C4SWTR	0x0	wo	Channel 4 event triggered by software Please refer to C1M description.
Bit 3	C3SWTR	0x0	wo	Channel 3 event triggered by software Please refer to C1M description.
Bit 2	C2SWTR	0x0	wo	Channel 2 event triggered by software Please refer to C1M description
Bit 1	C1SWTR	0x0	wo	Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.
Bit 0	OVFSWTR	0x0	wo	Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.

#### 14.2.4.7 Channel mode register1 (TMRx\_CM1)

##### Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C2OSEN	0x0	rw	Channel 2 output switch enable
Bit 14: 12	C2OCTRL	0x0	rw	Channel 2 output control
Bit 11	C2OBEN	0x0	rw	Channel 2 output buffer enable
Bit 10	C2OIEN	0x0	rw	Channel 2 output enable immediately
				Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN=0: 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS register.
Bit 9: 8	C2C	0x0	rw	Channel 1 output switch enable 0: C1ORAW is not affected by EXT 1: Once high level is detect on EXT input, clear C1ORAW.
Bit 7	C1OSEN	0x0	rw	Channel 1 output control This field defines the behavior of the original signal C1ORAW. 000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMR3_CVAL=TMR3_C1DT 010: C1ORAW is low when TMR3_CVAL=TMR3_C1DT 011: Switch C1ORAW level when TMR3_CVAL=TMR3_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A —OWCDIR=0, C1ORAW is high once TMR3_C1DT>TMR3_CVAL, else low; —OWCDIR=1, C1ORAW is low once TMR3_C1DT < TMR3_CVAL, else high; 111: PWM mode B
Bit 6: 4	C1OCTRL	0x0	rw	

					—OWCDIR=0, C1ORAW is low once TMR3_C1DT >TMR3_CVAL, else high; —OWCDIR=1, C1ORAW is high once TMR3_C1DT <TMR3_CVAL, else low. <i>Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.</i>
Bit 3	C1OBEN	0x0	rw		Channel 1 output buffer enable 0: Buffer function of TMR3_C1DT is disabled. The new value written to the TMR3_C1DT takes effect immediately. 1: Buffer function of TMR3_C1DT is enabled. The value to be written to the TMR3_C1DT is stored in the buffer register, and can be sent to the TMR3_C1DT register only on an overflow event.
Bit 2	C1OEN	0x0	rw		Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw		Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
<b>Input capture mode:</b>					
Bit	Register	Reset value	Type	Description	
Bit 15: 12	C2DF	0x0	rw	Channel 2 digital filter	
Bit 11: 10	C2IDIV	0x0	rw	Channel 2 input divider	
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.	
Bit 7: 4	C1DF	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events. 0000: No filter, sampling is done at $f_{DTS}$ 1000: $f_{SAMPLING} = f_{DTS}/8$ , N=6 0001: $f_{SAMPLING} = f_{CK\_INT}$ , N=2 1001: $f_{SAMPLING} = f_{DTS}/8$ , N=8 0010: $f_{SAMPLING} = f_{CK\_INT}$ , N=4 1010: $f_{SAMPLING} = f_{DTS}/16$ , N=5 0011: $f_{SAMPLING} = f_{CK\_INT}$ , N=8 1011: $f_{SAMPLING} = f_{DTS}/16$ , N=6 0100: $f_{SAMPLING} = f_{DTS}/2$ , N=6 1100: $f_{SAMPLING} = f_{DTS}/16$ , N=8 0101: $f_{SAMPLING} = f_{DTS}/2$ , N=8 1101: $f_{SAMPLING} = f_{DTS}/32$ , N=5 0110: $f_{SAMPLING} = f_{DTS}/4$ , N=6 1110: $f_{SAMPLING} = f_{DTS}/32$ , N=6	

				0111: $f_{SAMPLING} = f_{DTS}/4$ , N=8 1111: $f_{SAMPLING} = f_{DTS}/32$ , N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

#### 14.2.4.8 Channel mode register2 (TMR3\_CM2)

##### Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C4OSEN	0x0	rw	Channel 4 output switch enable
Bit 14: 12	C4OCTRL	0x0	rw	Channel 4 output control
Bit 11	C4OBEN	0x0	rw	Channel 4 output buffer enable
Bit 10	C4OIEN	0x0	rw	Channel 4 output enable immediately
				Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 9: 8	C4C	0x0	rw	00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7	C3OSEN	0x0	rw	Channel 3 output switch enable
Bit 6: 4	C3OCTRL	0x0	rw	Channel 3 output control
Bit 3	C3OBEN	0x0	rw	Channel 3 output buffer enable
Bit 2	C3OIEN	0x0	rw	Channel 3 output enable immediately
				Channel 3 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0': 00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 1: 0	C3C	0x0	rw	00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

##### Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 12	C4DF	0x0	rw	Channel 4 digital filter
Bit 11: 10	C4IDIV	0x0	rw	Channel 4 input divider
				Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 9: 8	C4C	0x0	rw	00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7: 4	C3DF	0x0	rw	Channel 3 digital filter

Bit 3: 2	C3IDIV	0x0	rw	Channel 3 input divider Channel 3 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0':
Bit 1:0	C3C	0x0	rw	00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

#### 14.2.4.9 Channel control register (TMR3\_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 14	Reserved	0x0	resd	Kept at its default value.
Bit 13	C4P	0x0	rw	Channel 4 polarity Please refer to C1P description.
Bit 12	C4EN	0x0	rw	Channel 4 enable Please refer to C1EN description.
Bit 11	C3CP	0x0	rw	Channel 3 complementary polarity Please refer to C1P description.
Bit 10	Reserved	0x0	resd	Default value
Bit 9	C3P	0x0	rw	Channel 3 polarity Please refer to C1P description.
Bit 8	C3EN	0x0	rw	Channel 3 enable Please refer to C1EN description.
Bit 7	C2CP	0x0	rw	Channel 2 complementary polarity Please refer to C1P description.
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	C2P	0x0	rw	Channel 2 polarity Please refer to C1P description.
Bit 4	C2EN	0x0	rw	Channel 2 enable Please refer to C1EN description.
Bit 3	C1CP	0x0	rw	Channel 1 complementary polarity Please refer to C1P description.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured as output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured in input mode: It is C1CP/C1P bits that define the active edge of an input signal.  00: C1IN rising edge is active. When used as external trigger, C1IN is not inverted. 01: C1IN falling edge is active. When used as external trigger, C1IN is inverted. 10: Reserved 11: C1IN rising edge and falling edge are both active. When used as external trigger, C1IN is not inverted.
Bit0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

Table 14-6 Standard CxOUT channel output control bit

CxEN bit	CxOUT output state
0	Output disabled (CxOUT=0, Cx_EN=0)
1	CxOUT = CxORAW + polarity, Cx_EN=1

Note: The state of the external I/O pins connected to the standard CxOUT channel depends on the CxOUT channel state and the GPIO and IOMUX registers.

#### 14.2.4.10 Counter value (TMR3\_CVAL)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	CVAL	0x0000	rw	Counter value

#### 14.2.4.11 Division value (TMR3\_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK\_CNT} = f_{TMR\_CLK} / (\text{DIV}[15:0] + 1)$ . DIV contains the value written at an overflow event.

#### 14.2.4.12 Period register (TMR3\_PR)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	PR	0x0000	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

#### 14.2.4.13 Channel 1 data register (TMR3\_C1DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effect immediately depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

#### 14.2.4.14 Channel 2 data register (TMR3\_C2DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	C2DT	0x0000	rw	Channel 2 data register When the channel 2 is configured as input mode: The C2DT is the CVAL value stored by the last channel 2 input event (C1IN) When the channel 2 is configured as output mode: C2DT is the value to be compared with the CVAL value. Whether the written value takes effect immediately depends on the C2OBEN bit, and the corresponding output is generated on C2OUT as configured.

#### 14.2.4.15 Channel 3 data register (TMR3\_C3DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	C3DT	0x0000	rw	Channel 3 data register When the channel 3 is configured as input mode: The C3DT is the CVAL value stored by the last channel 3 input event (C1IN) When the channel 3 is configured as output mode: C3DT is the value to be compared with the CVAL value. Whether the written value takes effect immediately depends on the C3OBEN bit, and the corresponding output is generated on C3OUT as configured.

#### 14.2.4.16 Channel 4 data register (TMR3\_C4DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value Channel 4 data register When the channel 4 is configured as input mode: The C4DT is the CVAL value stored by the last channel 4 input event (C1IN)
Bit 15: 0	C4DT	0x0000	rw	When the channel 4 is configured as output mode: C4DT is the value to be compared with the CVAL value. Whether the written value takes effect immediately depends on the C4OBEN bit, and the corresponding output is generated on C4OUT as configured.

#### 14.2.4.17 DMA control register (TMR3\_DMACTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12: 8	DTB	0x00	rw	DMA transfer bytes This field defines the number of DMA transfers: 00000: 1 byte      00001: 2 bytes 00010: 3 bytes      00011: 4 bytes ..... 10000: 17 bytes      10001: 18 bytes
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	ADDR	0x00	rw	DMA transfer address offset ADDR is defined as an offset starting from the address of the TMRx_CTRL1 register. 00000: TMRx_CTRL1 00001: TMRx_CTRL2 00010: TMRx_STCTRL .....

#### 14.2.4.18 DMA data register (TMR3\_DMADT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DMADT	0x0000	rw	DMA data register A read or write operation to the DMADT register accesses the TMR registers at the following address: TMRx peripheral address + ADDR*4 to TMRx peripheral address + ADDR*4 + DTB*4.

## 14.3 General-purpose timer (TMR14)

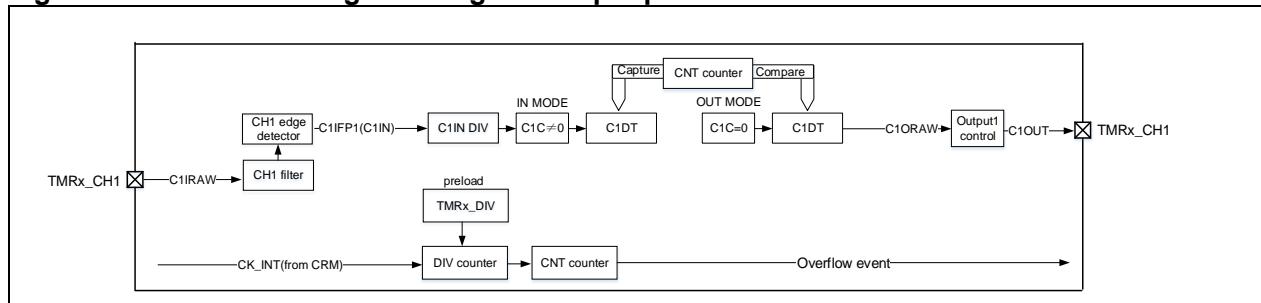
### 14.3.1 TMR14 introduction

The general-purpose timer TMR14 consists of a 16-bit counter supporting upcounting mode. The timer can be synchronized together with other timers.

### 14.3.2 TMR14 main features

- Source of count clock : internal clock
- 16-bit up counter
- 1x independent channels for input capture, output compare and PWM generation
- Synchronization control between master and slave timers
- Interrupt generation on the overflow event and channel event

**Figure 14-38 Block diagram of general-purpose TMR14**



### 14.3.3 TMR14 functional overview

#### 14.3.3.1 Count clock

The counter of TMR14 can be clocked by the internal clock (CK\_INT).

**Figure 14-39 Count clock**

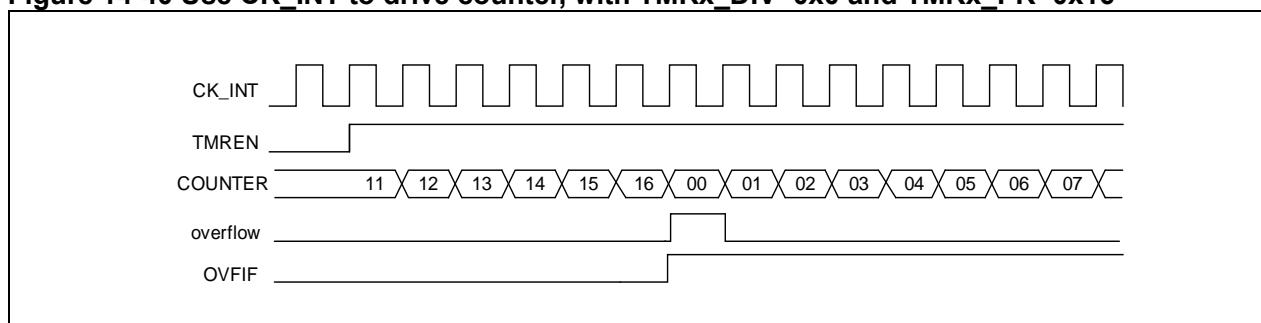


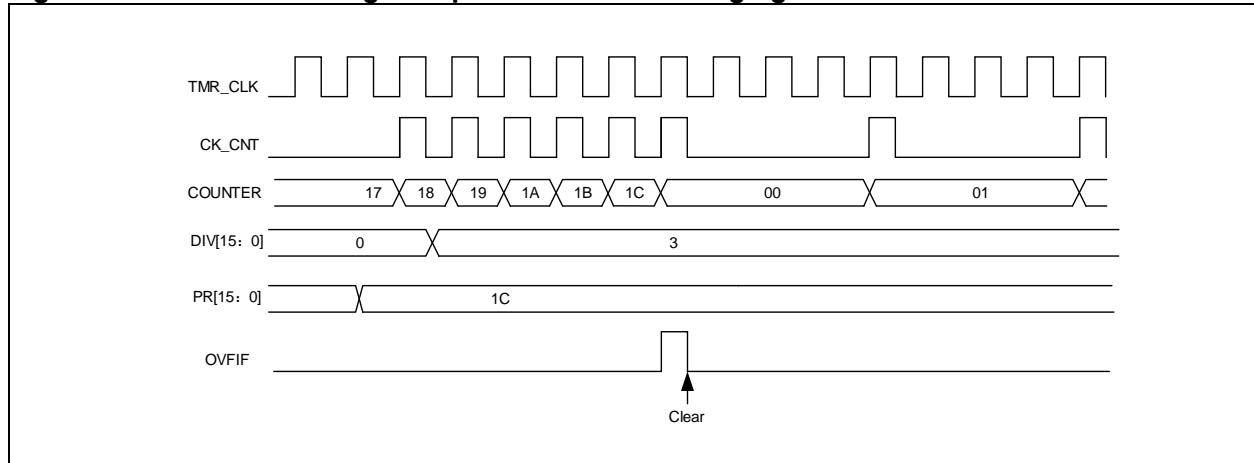
#### Internal clock (CK\_INT)

By default, the CK\_INT divided by a prescaler is used to drive the counter to start counting. The configuration process is as follows:

- Set the TMRx\_DIV register to set the counting frequency;
- Set the TMRx\_PR register to set the counting period;
- Set the TMREN bit in the TMRx\_CTRL1 register to enable the counter.

**Figure 14-40 Use CK\_INT to drive counter, with TMRx\_DIV=0x0 and TMRx\_PR=0x16**



**Figure 14-41 Counter timing with prescaler value changing from 1 to 4**

### 14.3.3.2 Counting mode

The general-purpose timer TMR14 consists of a 16-bit counter supporting upcounting mode.

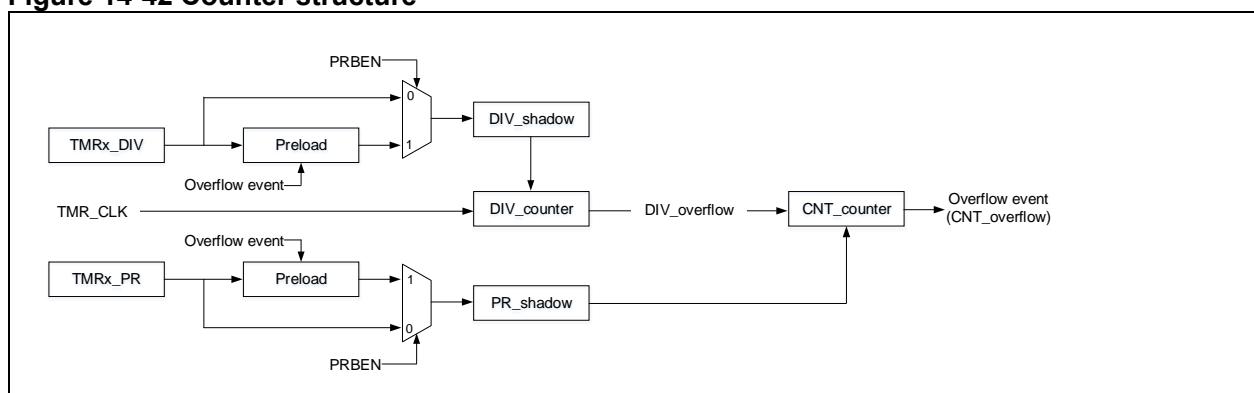
The TMRx\_PR register is used to configure the counting period. The value in the TMRx\_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx\_PR register is transferred to the shadow register at an overflow event.

The TMRx\_DIV register is used to configure the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to the TMRx\_PR register, when the periodic buffer is enabled, the value in the TMRx\_DIV register is updated to the shadow register at an overflow event.

Reading the TMRx\_CNT register returns to the current counter value, and writing to the TMRx\_CNT register updates the current counter value to the value being written.

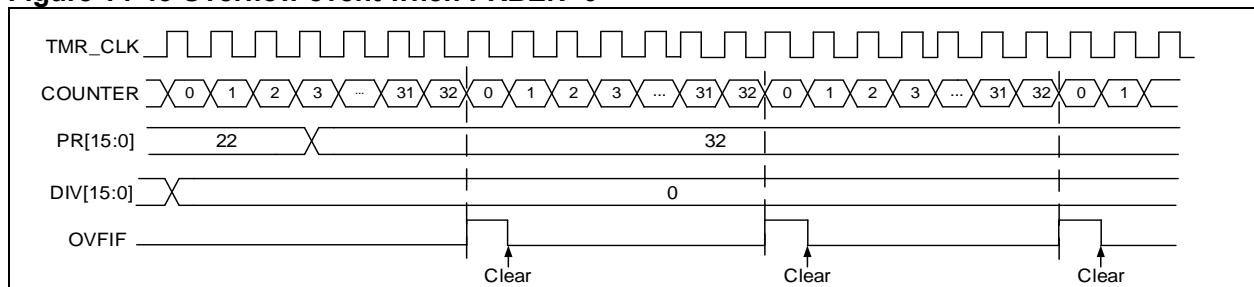
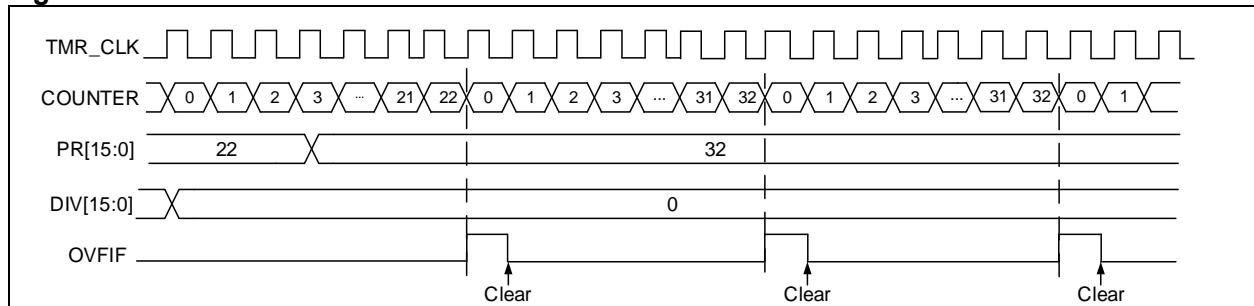
An overflow event is generated by default. Set OVREN=1 in the TMRx\_CTRL1 to disable generation of update events. The OVFS bit in the TMRx\_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only counter overflow/underflow triggers an overflow event.

Setting the TMREN bit (TMREN=1) enables the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR\_EN is set 1 clock cycle after the TMREN is set.

**Figure 14-42 Counter structure**

#### Upcounting mode

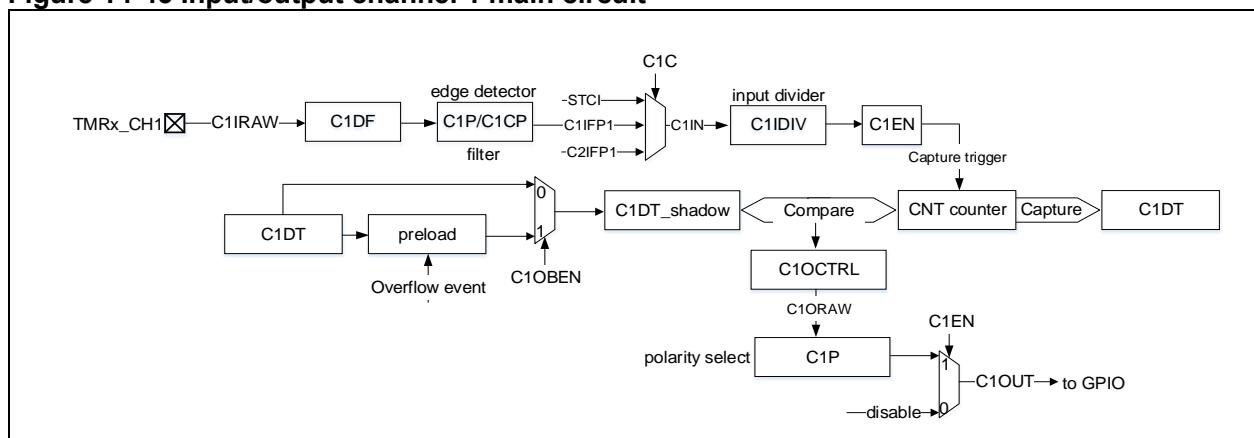
Set CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx\_CTRL1 register to enable upcounting mode. In upcounting mode, the counter counts from 0 to the value programmed in the TMRx\_PR register, restarts from 0, and generates an overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the register is no longer reloaded with the preload and re-loaded value after counter overflow occurs; otherwise, the prescaler and re-loaded value will be updated at an overflow event.

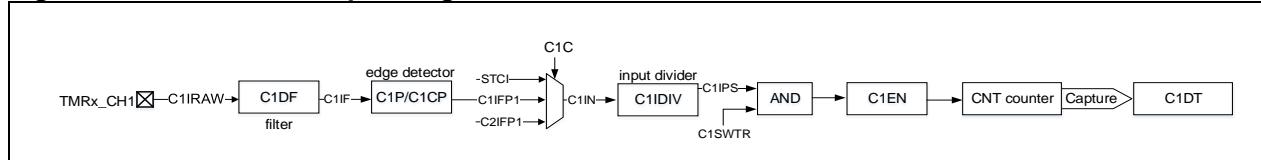
**Figure 14-43 Overflow event when PRBEN=0****Figure 14-44 Overflow event when PRBEN=1**

### 14.3.3.3 TMR input function

The TMR14 timer has one independent channel that can be configured as input or output. As input, each channel input signal is processed as below:

- TMRx\_CHx outputs the pre-processed CxIRAW. Set the C1INSEL bit to select TMRx\_CHx as the source of CxIRAW.
- CxIRAW inputs the digital filter and then outputs a filtered signal CxIF. Set the sampling frequency and sampling times of digital filter by setting the CxDI bit.
- CxIF inputs the edge detector and then outputs the signal CxIFPx after edge selection. The edge selection is controlled by CxP and CxCp bits, and can be selected as rising edge, falling edge or both edges active.
- CxIFPx inputs the capture signal selector and then outputs the signal CxIN after selection. The capture signal selector is controlled by the CxC bits. The source of CxIN can be set as CxIFPx. The CxIFPy (x≠y) is the CxIFPx from channel y and handled by channel x edge detector (for example, the C1IFP2 is the C1IFP1 from channel 1 and then handled by channel 2 edge detector).
- CxIN outputs the signal CxIPS through the input channel divider. The division factor is set to “no division”, “divided by 2”, “divided by 4” or “divided by 8” by setting the CxIDIV bit.

**Figure 14-45 Input/output channel 1 main circuit**

**Figure 14-46 Channel 1 input stage****Input mode**

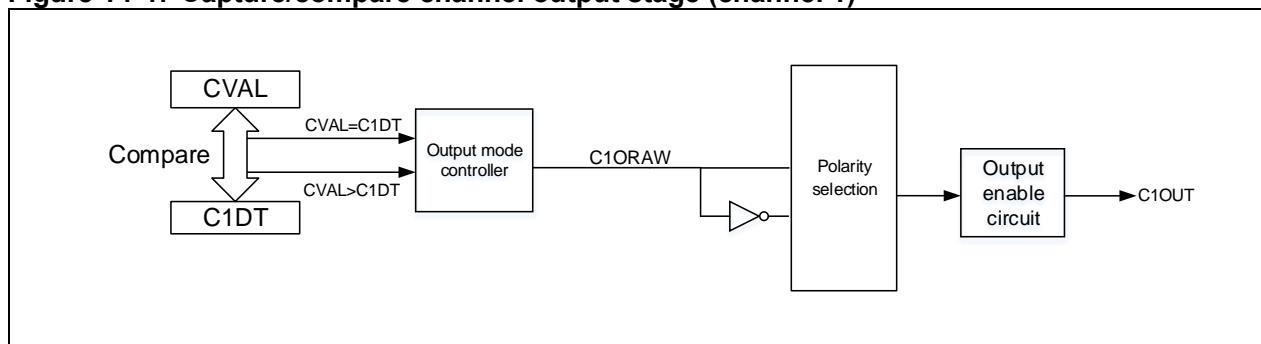
In input mode, the TMRx\_CxDT registers latch the current counter values after the selected trigger signal is detected, and the capture compare interrupt flag bit (CxIF) is set. An interrupt/DMA request will be generated if the CxIEN bit and CxDEN bit are enabled. If the selected trigger signal is detected when the CxIF is set to 1, a capture overflow event occurs. The TMRx\_CxDT register overwrites the recorded value with the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, following the configuration procedure mentioned below:

- Set C1C=01 in the TMR14\_CM1 register to select the C1IN as channel 1 input
- Set the filter bandwidth of C1IN signal (CxDF[3: 0])
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMR14\_CCTR register
- Program the capture frequency division of C1IN signal (C1DIV[1: 0])
- Enable channel 1 input capture (C1EN=1)
- If needed, enable the relevant interrupt by setting the C1IEN bit in the TMR14\_IDEN register

**14.3.3.4 TMR output function**

The TMR output consists of a comparator and an output controller. It is used to program the period, duty cycle and polarity of the output signal.

**Figure 14-47 Capture/compare channel output stage (channel 1)****Output mode**

Write CxC[1: 0]≠2'b00 to configure the channel as output to implement multiple output modes. In this case, the counter value is compared with the value in the TMR14\_CxDT register, and the intermediate signal CxORAW is generated according to the output mode selected by CxOCTRL[2: 0], which is sent to IO after being processed by the output control circuit. The period of the output signal is configured by the TMR14\_PR register, while the duty cycle by the TMR14\_CxDT register.

Output compare modes include:

- **PWM mode A:** Set CxOCTRL=3'b110 to enable PWM mode A. In upcounting, when TMRx\_C1DT>TMRx\_CVAL, C1ORAW outputs high; otherwise, outputs low. In downcounting, when TMRx\_C1DT<TMRx\_CVAL, C1ORAW outputs low; otherwise, outputs high.
- To set PWM mode A, the following process is recommended:

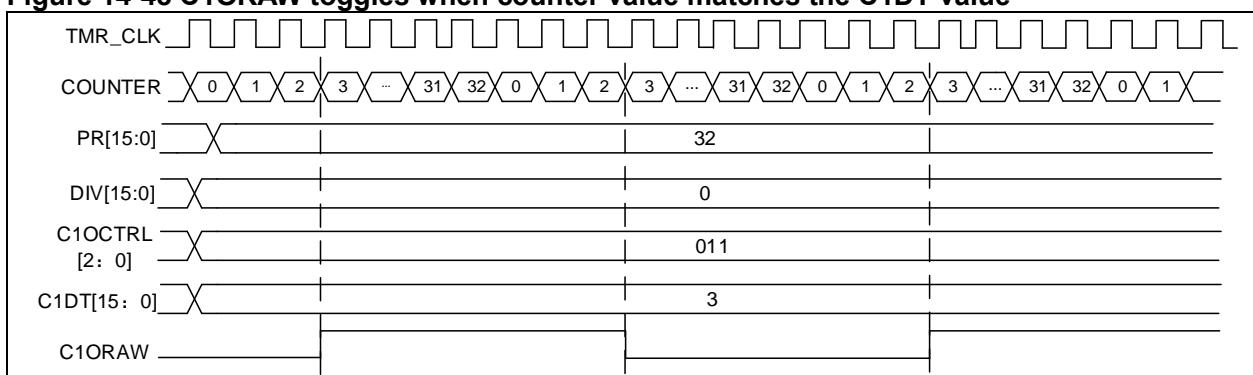
- Set the TMRx\_PR register to set PWM period;
- Set the TMRx\_CxDT register to set PWM duty cycle;
- Set CxOCTRL=3'b110 in the TMRx\_CM1/CM2 register to set output mode as PWM mode A;
- Set the TMRx\_DIV register and set the counting frequency;
- Set the TWCMSEL[1:0] bit in the TMRx\_CTRL1 register to set the count mode;
- Set CxP bit and CxCP bit in the TMRx\_CCTRL register to set output polarity;

- Set CxEN bit and CxCEN bit in the TMRx\_CCTRL register to enable channel output;
- Set the OEN bit in the TMRx\_BRK register to enable TMRx output;
- Set the corresponding GPIO of TMR output channel as the multiplexed mode;
- Set the TMREN bit in the TMRx\_CTRL1 register to enable TMRx counter.
- **PWM mode B:** Set CxOCTRL=3'b111 to enable PWM mode B. In upcounting, when TMRx\_C1DT>TMRx\_CVAL, C1ORAW outputs low; otherwise, outputs high. In downcounting, when TMRx\_C1DT<TMRx\_CVAL, C1ORAW outputs high; otherwise, outputs low.
- **Forced output mode:** Set CxOCTRL=3'b100/101 to enable forced output mode. In this case, the CxORAW is forced to be the programmed level, irrespective of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.
- **Output compare mode:** Set CxOCTRL=3'b001/010/011 to enable output compare mode. In this case, when the counter value matches the value of the CxDT register, the CxORAW is forced high, low or toggling.

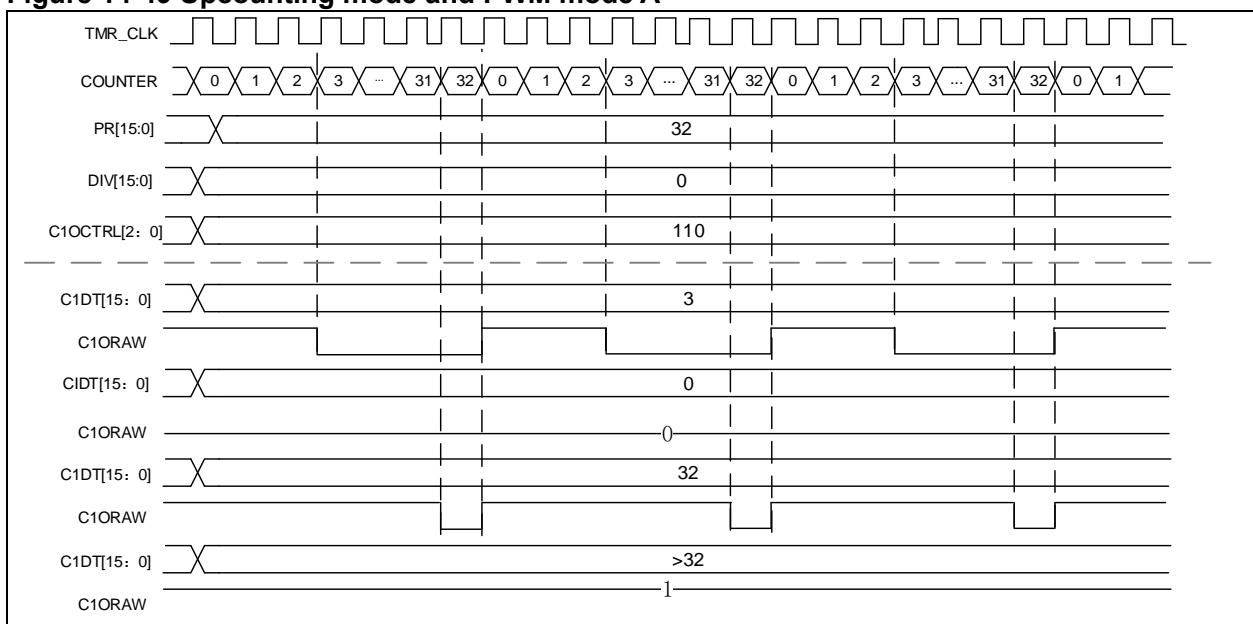
**Figure 14-48** gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

**Figure 14-49** gives an example of the combination between upcounting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

**Figure 14-48 C1ORAW toggles when counter value matches the C1DT value**



**Figure 14-49 Upcounting mode and PWM mode A**



### 14.3.3.5 Debug mode

When the microcontroller enters debug mode (Cortex™-M4 core halted), the TMR14 counter stops counting by setting the TMR14\_PAUSE in the DEBUG module.

### 14.3.4 TMR14 registers

These peripheral registers must be accessed by word (32 bits).  
All TMR14 registers are mapped into a 16-bit addressable space.

**Table 14-7 TMR14 register map and reset value**

Register name	Register	Reset value
TMR14_CTRL1	0x00	0x0000
TMR14_IDEN	0x0C	0x0000
TMR14ISTS	0x10	0x0000
TMR14_SWEVT	0x14	0x0000
TMR14_CM1	0x18	0x0000
TMR14_CCTRL	0x20	0x0000
TMR14_CVAL	0x24	0x0000
TMR14_DIV	0x28	0x0000
TMR14_PR	0x2C	0x0000
TMR14_C1DT	0x34	0x0000

#### 14.3.4.1 Control register1 (TMR14\_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value
Bit 9: 8	CLKDIV	0x0	rw	Clock divider 00: Normal, $f_{DTS}=f_{CK\_INT}$ 01: Divided by 2, $f_{DTS}=f_{CK\_INT}/2$ 10: Divided by 4, $f_{DTS}=f_{CK\_INT}/4$ 11: Reserved
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 6: 3	Reserved	0x0	resd	Kept at its default value
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 1	OVFEN	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	TMR enable 0: Enabled 1: Disabled

#### 14.3.4.2 Interrupt enable register (TMR14\_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15:2	Reserved	0x0000	resd	Kept at its default value.
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled 1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

#### 14.3.4.3 Interrupt status register (TMR14ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag

				This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing "0". 0: No capture is detected 1: Capture is detected.
Bit 8: 2	Reserved	0x00	resd	Kept at its default value.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT 0: No capture event occurs 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurs 1: Compare event is generated
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurs 1: Overflow event is generated. If OVREN=0 and OVFS=0 in the TMR14_CTRL1 register: - An overflow event is generated when OVFG= 1 in the TMR14_SWEVE register; - An overflow event is generated when the counter CVAL is reinitialized by a trigger event.

#### 14.3.4.4 Software event register (TMR14\_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 2	Reserved	0x0000	resd	Kept at its default value.
Bit 1	C1SWTR	0x0	wo	Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.
Bit 0	OVFSWTR	0x0	wo	Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.

#### 14.3.4.5 Channel mode register1 (TMR14\_CM1)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the CxIx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

##### Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6: 4	C1OCTRL	0x0	rw	Channel 1 output control This field defines the behavior of the original signal C1ORAW. 000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMR14_CVAL=TMR14_C1DT 010: C1ORAW is low when TMR14_CVAL=TMR14_C1DT 011: Switch C1ORAW level when TMR14_CVAL=TMR14_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high.

				110: PWM mode A - OWCDIR=0, C1ORAW is high once TMR14_C1DT>TMR14_CVAL, else low; - OWCDIR=1, C1ORAW is low once TMR14_C1DT <TMR14_CVAL, else high; 111: PWM mode B - OWCDIR=0, C1ORAW is low once TMR14_C1DT >TMR14_CVAL, else high; - OWCDIR=1, C1ORAW is high once TMR14_C1DT <TMR14_CVAL, else low. <i>Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.</i>
Bit 3	C1OBEN	0x0	rw	Channel 1 output buffer enable 0: Buffer function of TMR14_C1DT is disabled. The new value written to the TMR14_C1DT takes effect immediately. 1: Buffer function of TMR14_C1DT is enabled. The value to be written to the TMR14_C1DT is stored in the buffer register, and can be sent to the TMR14_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw	Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Reserved 11: Reserved
<b>Input capture mode:</b>				
Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7: 4	C1DF	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events. 0000: No filter, sampling is done at $f_{DTS}$ 1000: $f_{SAMPLING} = f_{DTS}/8$ , N=6 0001: $f_{SAMPLING} = f_{CK\_INT}$ , N=2 1001: $f_{SAMPLING} = f_{DTS}/8$ , N=8 0010: $f_{SAMPLING} = f_{CK\_INT}$ , N=4 1010: $f_{SAMPLING} = f_{DTS}/16$ , N=5 0011: $f_{SAMPLING} = f_{CK\_INT}$ , N=8 1011: $f_{SAMPLING} = f_{DTS}/16$ , N=6 0100: $f_{SAMPLING} = f_{DTS}/2$ , N=6 1100: $f_{SAMPLING} = f_{DTS}/16$ , N=8 0101: $f_{SAMPLING} = f_{DTS}/2$ , N=8 1101: $f_{SAMPLING} = f_{DTS}/32$ , N=5 0110: $f_{SAMPLING} = f_{DTS}/4$ , N=6 1110: $f_{SAMPLING} = f_{DTS}/32$ , N=6 0111: $f_{SAMPLING} = f_{DTS}/4$ , N=8 1111: $f_{SAMPLING} = f_{DTS}/32$ , N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider.

Bit 1: 0 C1C

0x0

rw

00: No divider. An input capture is generated at each active edge.  
01: An input compare is generated every 2 active edges  
10: An input compare is generated every 4 active edges  
11: An input compare is generated every 8 active edges  
Note: the divider is reset once C1EN='0'

---

#### Channel 1 configuration

This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0':  
00: Output  
01: Input, C1IN is mapped on C1IFP1  
10: Reserved  
11: Reserved

---

#### 14.3.4.6 Channel control register (TMR14\_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 4	Reserved	0x0	resd	Kept at its default value.
Bit 3	C1CP	0x0	rw	Channel 1 complementary polarity Please refer to C1P description.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured in output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured in input mode: It is C1CP/C1P bits that define the active edge of an input signal. 00: C1IN rising edge is active. When used as external trigger, C1IN is not inverted. 01: C1IN falling edge is active. When used as external trigger, C1IN is inverted. 10: Reserved 11: C1IN rising edge and falling edge are both active. When used as external trigger, C1IN is not inverted
Bit0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

Table 14-8 Standard CxOUT channel output control bit

CxEN bit	CxOUT output state
0	Output disabled (CxOUT=0)
1	CxOUT = CxORAW + polarity

Note: The state of the external I/O pins connected to the standard CxOUT channel depends on the CxOUT channel state and the GPIO and IOMUX registers.

#### 14.3.4.7 Counter value (TMR14\_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

#### 14.3.4.8 Division value (TMR14\_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK\_CNT} = f_{TMR\_CLK} / (DIV[15:0]+1)$ . DIV contains the value written at an overflow event.

#### 14.3.4.9 Period register (TMR14\_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0000	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

#### 14.3.4.10 Channel 1 data register (TMR14\_C1DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effect immediately

---

depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

---

#### 14.3.4.11 Channel input remap register (TMR14\_RMP)

Bit	Register	Reset value	Type	Description
Bit 15: 0	Reserved	0x00	resd	Kept at its default value.
Bit 1: 0	TMR14_CH1_IRMP	0x0	rw	<p>TMR14 channel 1 input remap 00: TMR14 channel 1 input is connected to GPIO 01: ERTC_CLK 10: HEXT/32 11: CLK_OUT</p>

## 14.4 General-purpose timer (TMR15)

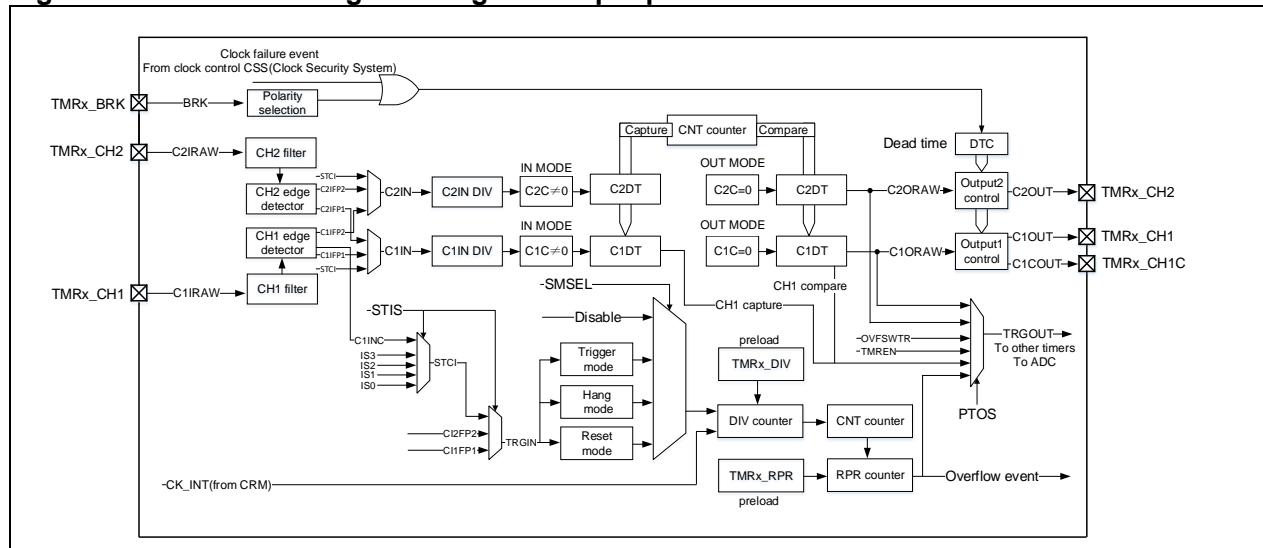
### 14.4.1 TMR15 introduction

The general-purpose timer TMR14 consists of a 16-bit counter supporting upcounting mode. It has two capture/compare registers, and two independent channels to achieve dead-time insert, input capture and programmable PWM output.

### 14.4.2 TMR15 main features

- Source of count clock : internal clock, external clock and internal trigger
- 16-bit upcounter and 8-bit repetition counter
- 2 independent channels for input capture, output compare, PWM generation, one-pulse mode output and dead-time insertion
- 1 x independent channel for complementary output
- TMR brake feature support
- Synchronization control between timers
- Interrupt/DMA generation on the overflow event, trigger event and channel event
- Support TMR burst DMA transfer

**Figure 14-50 Block diagram of general-purpose TMR15**

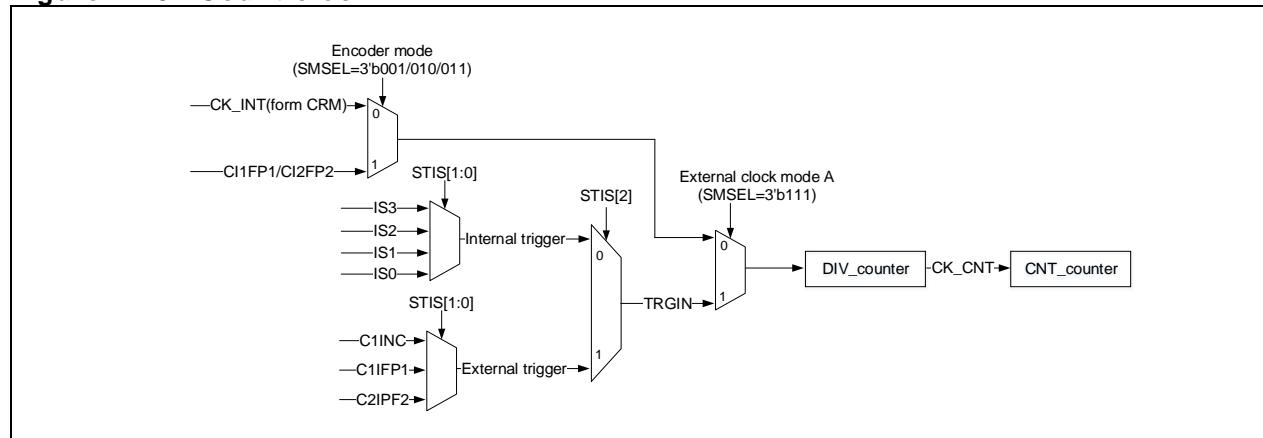


### 14.4.3 TMR15 functional overview

#### 14.4.3.1 Count clock

The counter of TMR15 can be clocked by the internal clock (CK\_INT), external clock (external clock mode A) and internal trigger input (ISx).

**Figure 14-51 Count clock**

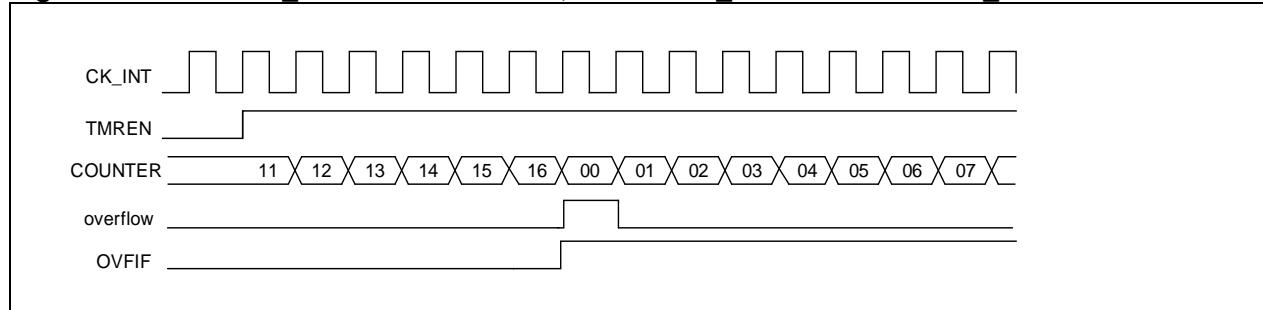


### Internal clock (CK\_INT)

By default, the CK\_INT divided by a prescaler is used to drive the counter to start counting. The configuration process is as follows:

- Set the CLKDIV[1:0] bit in the TMRx\_CTRL1 register to set the CK\_INT frequency;
- Set the TWCMSEL[1:0] bit in the TMRx\_CTRL1 register to select count mode. If the one-way count direction is set, configure OWCDIR bit in the TMRx\_CTRL1 register to select the specific direction.
- Set the TMRx\_DIV register to set the counting frequency;
- Set the TMRx\_PR register to set the counting period;
- Set the TMREN bit in the TMRx\_CTRL1 register to enable the counter.

**Figure 14-52 Use CK\_INT to drive counter, with TMRx\_DIV=0x0 and TMRx\_PR=0x16**



### External clock (TRGIN/EXT)

The counter can be clocked by TRGIN signal.

When SMSEL=3'b111, external clock mode A is selected. Set STIS[2:0] bit to select TRGIN signal to drive the counter to start counting. The external clock sources include: C1INC (STIS=3'b100, channel 1 rising edge and falling edge), C1IFP1 (STIS=3'b101, channel 1 signal after filtering and polarity selection), and C2IFP2 (STIS=3'b110, channel 2 signal after filtering and polarity selection).

To use external clock mode A, follow the configuration steps as below:

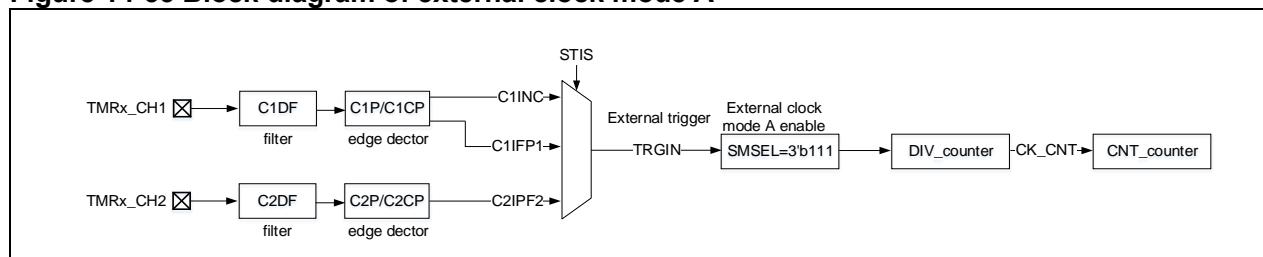
- Configure the external clock source TRGIN.

When TMRx\_CH1 is selected as the TRGIN, configure the channel 1 input filter (by setting the C1DF[3:0] bit in the TMRx\_CM1 register) and channel 1 input polarity (by setting the C1P/C1CP in the TMRx\_CCTRL register).

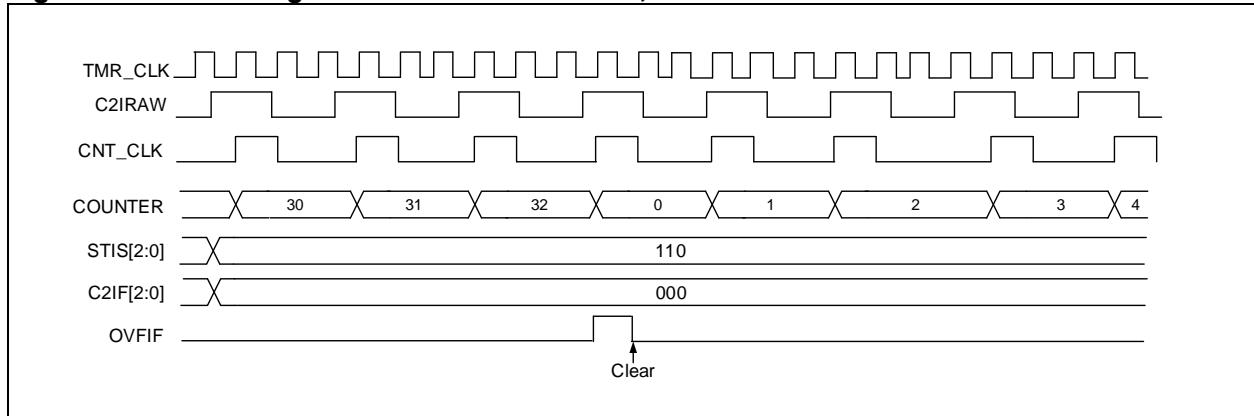
When TMRx\_CH2 is selected as the TRGIN, configure the channel 2 input filter (by setting the C2DF[3:0] bit in the TMRx\_CM1 register) and channel 1 input polarity (by setting the C2P/C2CP in the TMRx\_CCTRL register).

- Set the TRGIN signal source by setting the STIS[1:0] bit in the TMRx\_STCTRL register.
- Enable external clock mode A by setting SMSEL=3'b111 in the TMRx\_STCTRL register.
- Set counter counting frequency by setting the DIV[15:0] bit in the TMRx\_DIV register.
- Set counter counting period by setting the PR[15:0] bit in the TMRx\_PR register.
- Enable counter by setting the TMREN bit in the TMRx\_CTRL1 register.

**Figure 14-53 Block diagram of external clock mode A**



*Note: The delay is present between the signal on the input side and the actual clock of the counter due to the synchronization circuit.*

**Figure 14-54 Counting in external clock mode A, with PR=0x32 and DIV=0x0****Internal trigger input (ISx)**

Timer synchronization allows interconnection between several timers. The TMR\_CLK of one timer can be provided by the TRGOUT signal output by another timer. Set the STIS[2:0] bit to select internal trigger signal to enable counting.

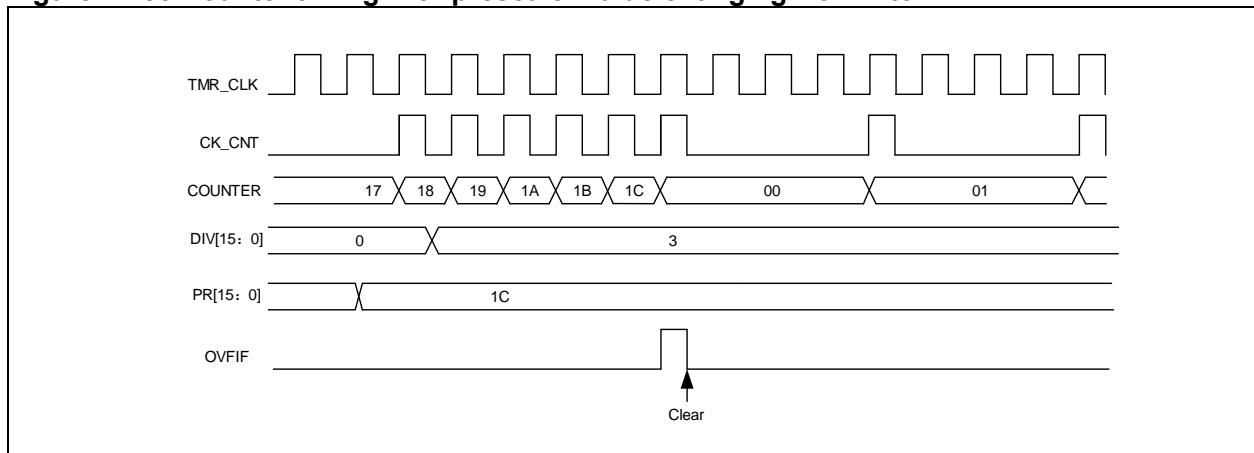
The TMR15 timer consists of a 16-bit prescaler, which is used to generate the CK\_CNT that enables the counter to count. The frequency division relationship between the CK\_CNT and TMR\_CLK can be adjusted by setting the value of the TMR15\_DIV register. The prescaler value can be modified at any time, but it takes effect only when the next overflow event occurs.

The internal trigger input is configured as follows:

- Set the TMRx\_PR register to set counting period;
- Set the TMRx\_DIV register to set counting frequency;
- Set the STIS[2:0] bit (range: 3'b000~3'b011) in the TMRx\_STCTRL register and select internal trigger;
- Set SMSEL[2:0]=3'b111 in the TMRx\_STCTRL register and select external clock mode A;
- Set the TMREN bit in the TMRx\_CTRL1 register to enable TMRx counter.

**Table 14-9 TMR15 internal trigger connection**

Slave timer	IS0 (STIS=000)	IS1 (STIS=001)	IS2 (STIS=010)	IS3 (STIS=011)
TMR15	-	TMR3	TMR16_OC	TMR17_OC

**Figure 14-55 Counter timing with prescaler value changing from 1 to 4****14.4.3.2 Counting mode**

The TMR15 timer supports several counting modes to meet different application scenarios.

The TMRx\_PR register is used to configure the counting period. The value in the TMRx\_PR register is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx\_PR register is transferred to the shadow register at an overflow event.

The TMRx\_DIV register is used to configure the counting frequency. The counter counts once every

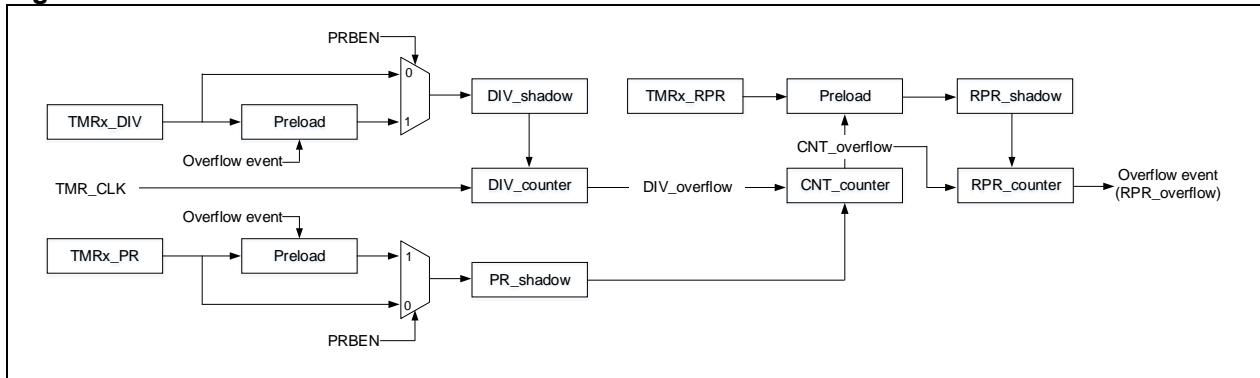
count clock period ( $\text{DIV}[15:0]+1$ ). Similar to the  $\text{TMRx\_PR}$  register, when the periodic buffer is enabled, the value in the  $\text{TMRx\_DIV}$  register is updated to the shadow register at an overflow event.

Reading the  $\text{TMRx\_CNT}$  register returns to the current counter value, and writing to the  $\text{TMRx\_CNT}$  register updates the current counter value to the value being written.

An overflow event is generated by default. Set  $\text{OVFEN}=1$  in the  $\text{TMRx\_CTRL1}$  to disable generation of update events. The  $\text{OVFS}$  bit in the  $\text{TMRx\_CTRL1}$  register is used to select overflow event source. By default, counter overflow/underflow, setting  $\text{OVSFWTR}$  bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the  $\text{OVFS}$  bit is set, only counter overflow/underflow triggers an overflow event.

Setting the  $\text{TMREN}$  bit ( $\text{TMREN}=1$ ) enables the timer to start counting. Base on synchronization logic, however, the actual enable signal  $\text{TMR\_EN}$  is set 1 clock cycle after the  $\text{TMREN}$  is set.

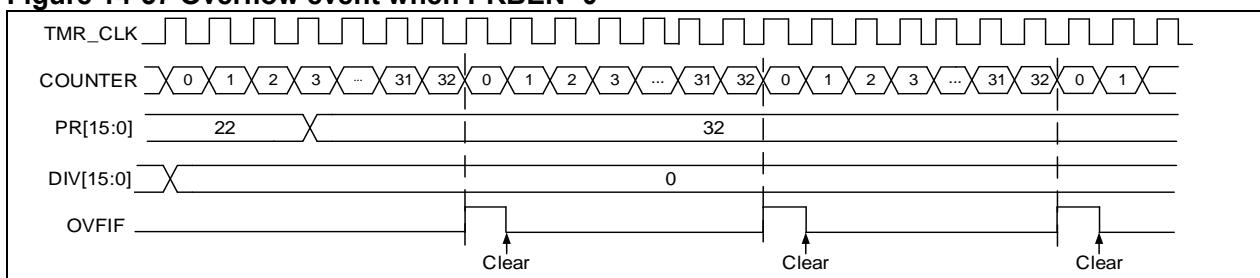
**Figure 14-56 Counter structure**



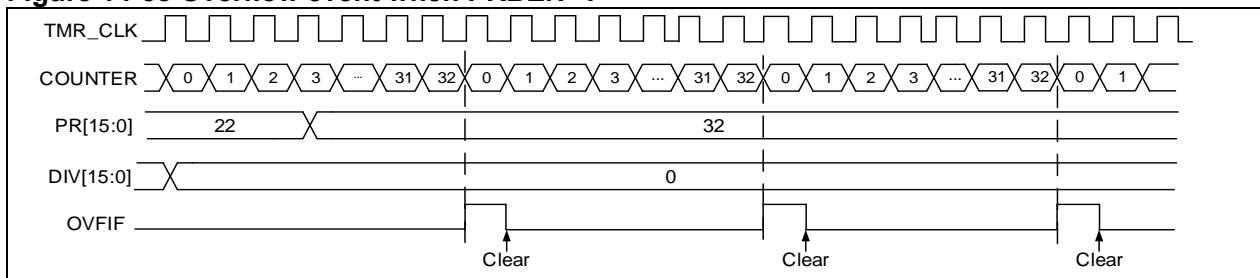
### Upcounting mode

Set  $\text{CMSEL}[1:0]=2'b00$  and  $\text{OWCDIR}=1'b0$  in the  $\text{TMRx\_CTRL1}$  register to enable upcounting mode. In upcounting mode, the counter counts from 0 to the value programmed in the  $\text{TMRx\_PR}$  register, restarts from 0, and generates an overflow event, with the  $\text{OVFIF}$  bit being set to 1. If the overflow event is disabled, the register is no longer reloaded with the preload and re-loaded value after counter overflow occurs; otherwise, the prescaler and re-loaded value will be updated at an overflow event.

**Figure 14-57 Overflow event when PRBEN=0**



**Figure 14-58 Overflow event when PRBEN=1**

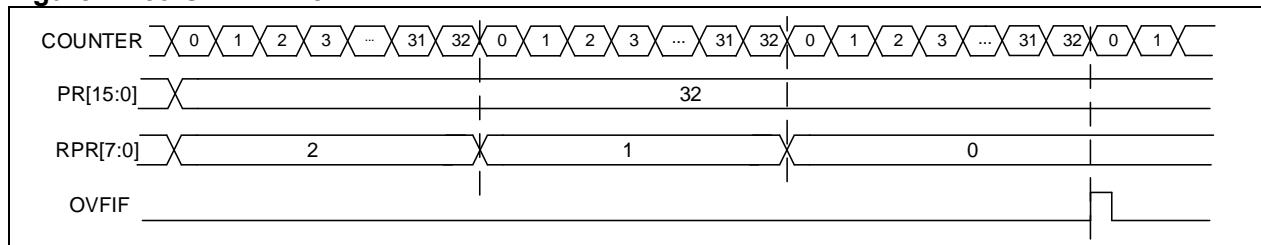


### Repetition counter mode:

The  $\text{TMRx\_RPR}$  register is used to configure the counting period of repetition counter. The repletion counter mode is enabled when the repetition counter value is not equal to 0. In this mode, an overflow event occurs once at every counter overflow ( $\text{RPR}[7:0]+1$ ), and the repetition counter is decremented at

each counter overflow. An overflow event is generated only when the repetition counter reaches 0. The frequency of the overflow event can be adjusted by setting the repetition counter value.

**Figure 14-59 OVIF when RPR=2**

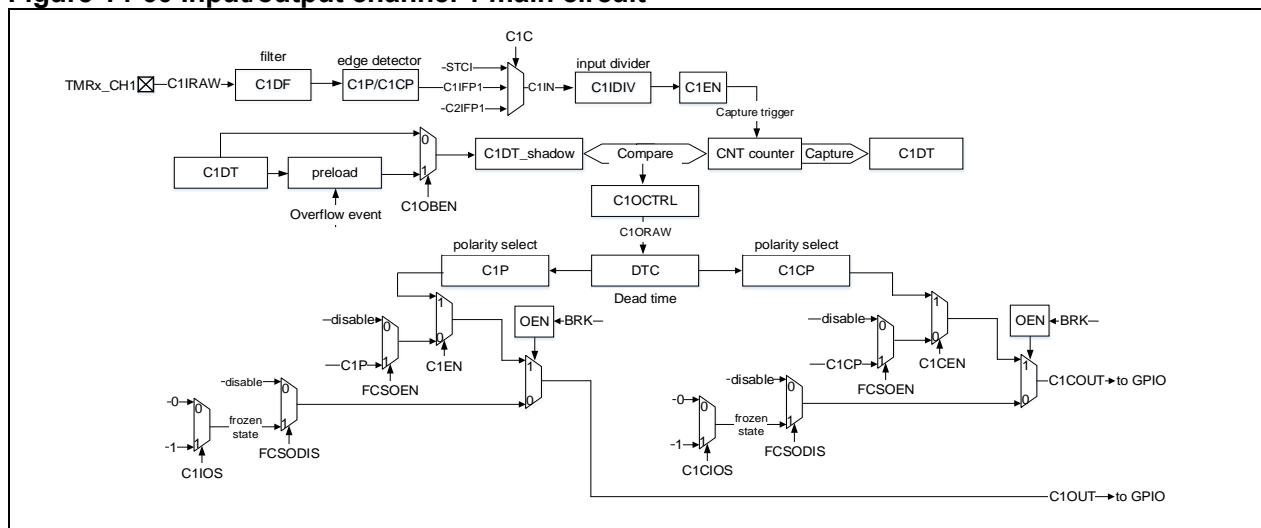


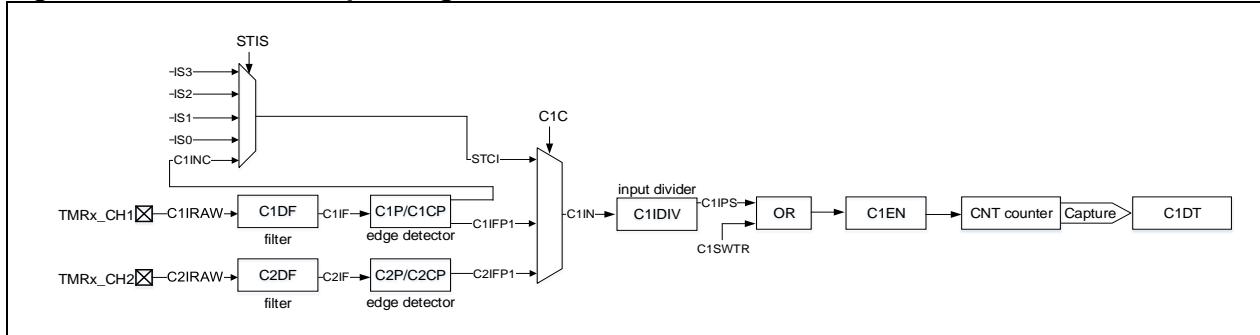
### 14.4.3.3 TMR input function

The TMR15 timer has two independent channels, each of which can be configured as input or output. As input, each channel input signal is processed as below:

- TMRx\_CHx outputs the pre-processed CxIRAW. Set the C1INSEL bit to select TMRx\_CHx as the source of CxIRAW.
- CxIRAW inputs digital filter and outputs a filtered signal CxIF. Set the sampling frequency and times of digital filter by setting the CxDI bit.
- CxIF inputs edge detector and outputs the signal CxIFPx after edge selection. The edge selection is controlled by CxP and CxCP bits, and can be selected as rising edge, falling edge or both edges active.
- CxIFPx inputs capture signal selector and outputs the signal CxIN after selection. The capture signal selector is controlled by the CxC bits. The source of CxIN can be set as CxIFPx, CyIFPx or STCI. The CyIFPx ( $x \neq y$ ) is the CyIFPy from channel y and handled by channel x edge detector (for example, the C1IFP2 is the C1IFP1 from channel 1 and then handled by channel 2 edge detector), and STCI derives from the slave timer controller, and its source is selected by setting the STIS bit.
- CxIN outputs the signal CxIPS through the input channel divider. The division factor is set to “no division”, “divided by 2”, “divided by 4” or “divided by 8” by setting the CxIDIV bit.

**Figure 14-60 Input/output channel 1 main circuit**



**Figure 14-61 Channel 1 input stage**

### **Input mode**

In input mode, the TMRx\_CxDT registers latch the current counter values after the selected trigger signal is detected, and the capture compare interrupt flag bit (CxIF) is set to 1. An interrupt/DMA request will be generated if the CxIEN bit and CxDEN bit are enabled. If the selected trigger signal is detected when the CxIF is set, a capture overflow event occurs. The TMRx\_CxDT register overwrites the recorded value with the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, following the configuration procedure mentioned below:

- Set C1C=01 in the TMR15\_CM1 register to select the C1IN as channel 1 input
- Set the filter bandwidth of C1IN signal (CxDF[3: 0])
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMR15\_CCTR register
- Program the capture frequency division of C1IN signal (C1DIV[1: 0])
- Enable channel 1 input capture (C1EN=1)

If needed, enable the relevant interrupt or DMA request by setting the C1IEN bit in the TMR15\_IDEN register or the C1DEN bit in the TMR15\_IDEN register

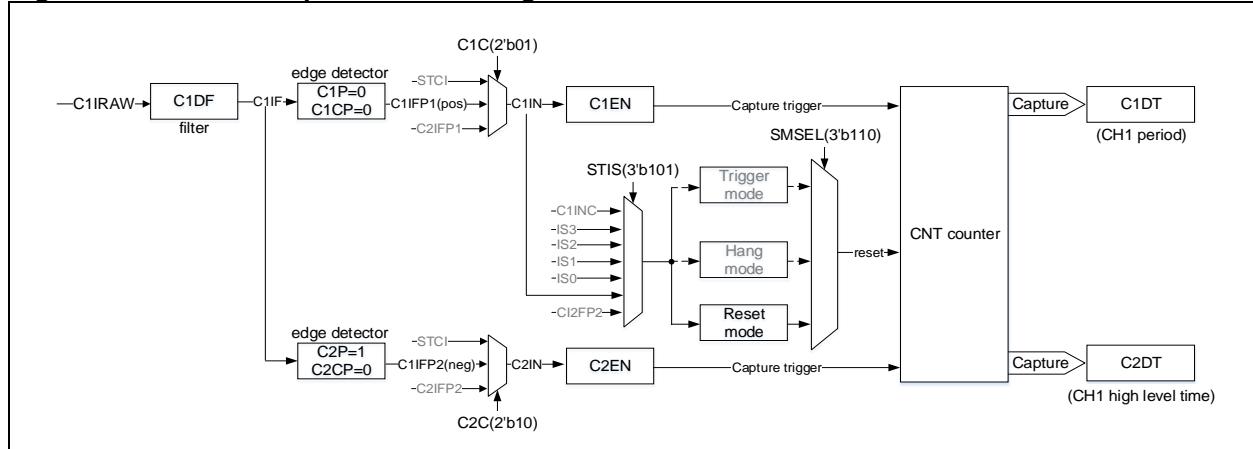
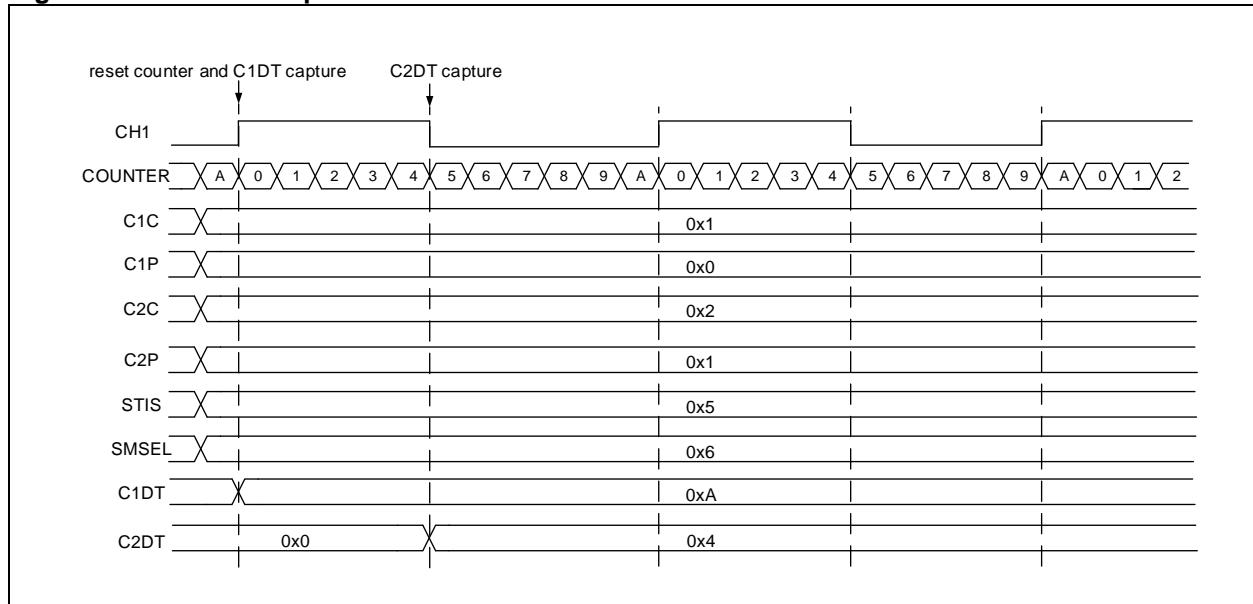
### **PWM input**

The PWM input mode applies to channel 1 and channel 2. To enable this mode, map the C1IN and C2IN to the same TMRx\_CHx, and configure the CxIFPx of channel 1/2 to trigger slave timer controller reset.

The PWM input mode can be used to measure the period and duty cycle of input signal. The period and duty cycle of channel 1 can be measured as follows:

- Set C1C=2'b01 to set C1IN as C1IFP1;
- Set C1P=1'b0 to set C1IFP1 rising edge active;
- Set C2C=2'b10 to set C2IN as C1IFP2;
- Set C2P=1'b1 to set C1IFP2 falling edge active;
- Set STIS=3'b101 to set C1IFP1 as the slave timer trigger signal;
- Set SMSEL=3'b100 to set the slave timer in reset mode;
- Set C1EN=1'b1 and C2EN=1'b1 to enable channel 1 and input capture.

In these configurations, the rising edge of channel 1 input signal triggers capture and saves captured values to the C1DT register, and channel 1 input signal rising edge resets the counter. The falling edge of channel 1 input signal triggers capture and saves captured values to the C2DT register. The period and duty of channel 1 input signal can be calculated through C1DT and C2DT respectively.

**Figure 14-62 PWM input mode configuration****Figure 14-63 PWM input mode**

#### 14.4.3.4 TMR output function

The TMR output consists of a comparator and an output controller. It is used to program the period, duty cycle and polarity of the output signal. The advanced-control timer output function varies from one channel to one channel.

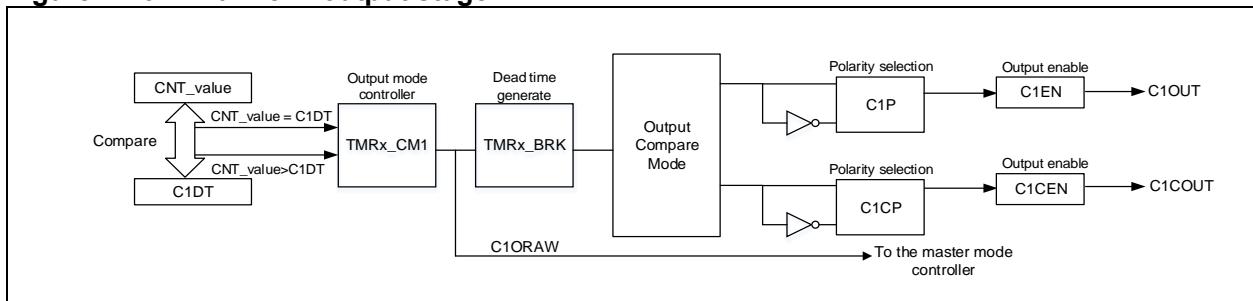
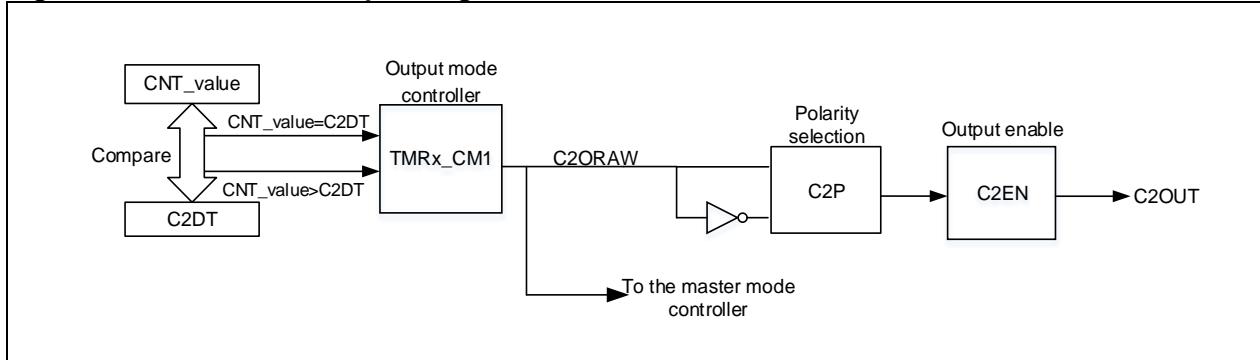
**Figure 14-64 Channel 1 output stage**

Figure 14-65 Channel 2 output stage



### Output mode

Write  $CxC[2:0] \neq 2'b00$  to configure the channel as output to implement multiple output modes. In this case, the counter value is compared with the value in the  $TMRx\_CxDT$  register, and the intermediate signal  $CxORAW$  is generated according to the output mode selected by  $CxOCTRL[2:0]$ , which is sent to IO after being processed by the output control circuit. The period of the output signal is configured by the  $TMRx\_PR$  register, while the duty cycle by the  $TMR15\_CxDT$  register.

Output compare modes include:

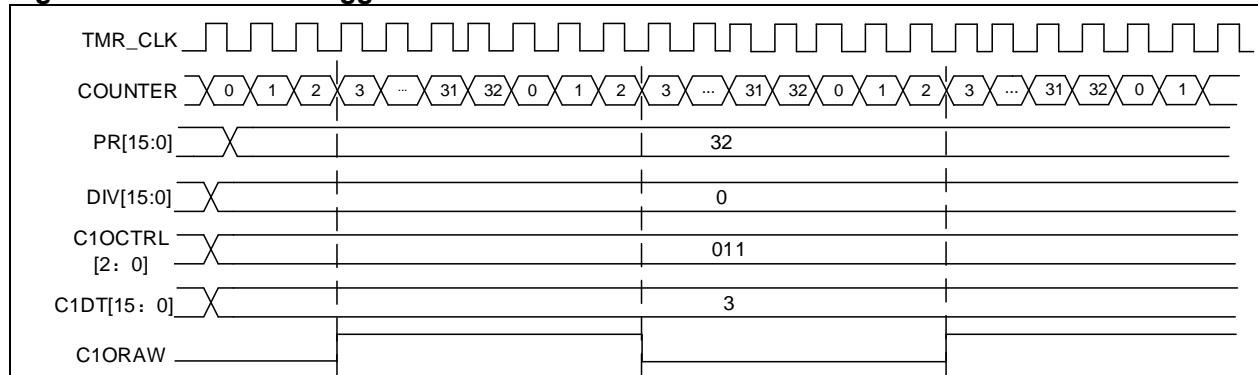
- **PWM mode A:** Set  $CxOCTRL=3'b110$  to enable PWM mode A. In upcounting, when  $TMRx\_C1DT > TMRx\_CVAL$ ,  $C1ORAW$  outputs high; otherwise, outputs low. In downcounting, when  $TMRx\_C1DT < TMRx\_CVAL$ ,  $C1ORAW$  outputs low; otherwise, outputs high. To set PWM mode A, the following process is recommended:
  - Set the  $TMRx\_PR$  register to set PWM period;
  - Set the  $TMRx\_CxDT$  register to set PWM duty cycle;
  - Set  $CxOCTRL=3'b110$  in the  $TMRx\_CM1/CM2$  register and set output mode as PWM mode A;
  - Set the  $TMRx\_DIV$  register to set the counting frequency;
  - Set the  $TWCMSEL[1:0]$  bit in the  $TMRx\_CTRL1$  register to set the count mode;
  - Set  $CxP$  bit and  $CxCP$  bit in the  $TMRx\_CCTRL$  register to set output polarity;
  - Set  $CxEN$  bit and  $CxCEN$  bit in the  $TMRx\_CCTRL$  register to enable channel output;
  - Set the  $OEN$  bit in the  $TMRx\_BRK$  register to enable  $TMRx$  output;
  - Set the corresponding GPIO of  $TMR$  output channel as the multiplexed mode;
  - Set the  $TMREN$  bit in the  $TMRx\_CTRL1$  register to enable  $TMRx$  counter.
- **PWM mode B:** Set  $CxOCTRL=3'b111$  to enable PWM mode B. In upcounting, when  $TMRx\_C1DT > TMRx\_CVAL$ ,  $C1ORAW$  outputs low; otherwise, outputs high. In downcounting, when  $TMRx\_C1DT < TMRx\_CVAL$ ,  $C1ORAW$  outputs high; otherwise, outputs low.
- **Forced output mode:** Set  $CxOCTRL=3'b100/101$  to enable forced output mode. In this case, the  $CxORAW$  is forced to be the programmed level, irrespective of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.
- **Output compare mode:** Set  $CxOCTRL=3'b001/010/011$  to enable output compare mode. In this case, when the counter value matches the value of the  $CxDT$  register, the  $CxORAW$  is forced high ( $CxOCTRL=3'b001$ ), low ( $CxOCTRL=3'b010$ ) or toggling ( $CxOCTRL=3'b011$ ).
- **One-pulse mode:** This is a particular case of PWM mode. Set  $OCMEN=1$  to enable one-pulse mode. In this mode, the comparison match is performed in the current counting period. The  $TMREN$  bit is cleared as soon as the current counting is completed. Therefore, only one pulse is output. When configured as in upcounting mode, the configuration must follow the rule:  $CVAL < CxDT \leq PR$ ; in downcounting mode,  $CVAL > CxDT$  is required.
- **Fast output mode:** Set  $CxOIEN=1$  to enable this mode. If enabled, the  $CxORAW$  signal will not change when the counter value matches the  $CxDT$ , but at the beginning of the current counting period. In other words, the comparison result is advanced, so the comparison result between the counter value and the  $CxDT$  register will determine the level of  $CxORAW$  in advance.

**Figure 14-66** gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

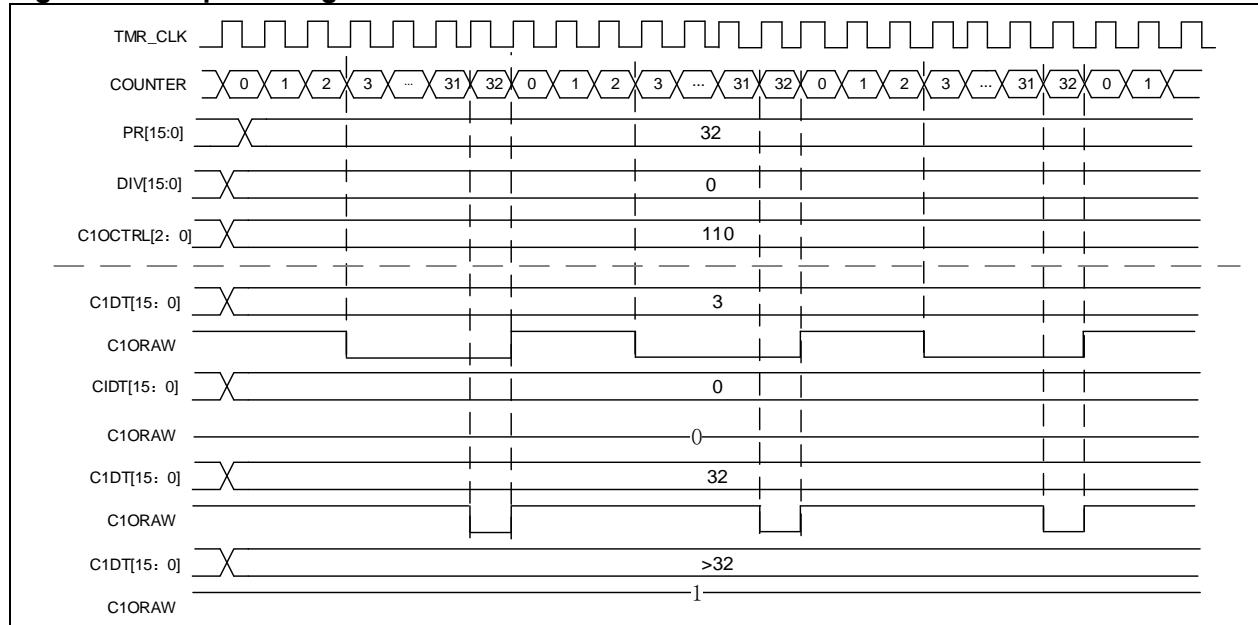
**Figure 14-67** gives an example of the combination between upcounting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

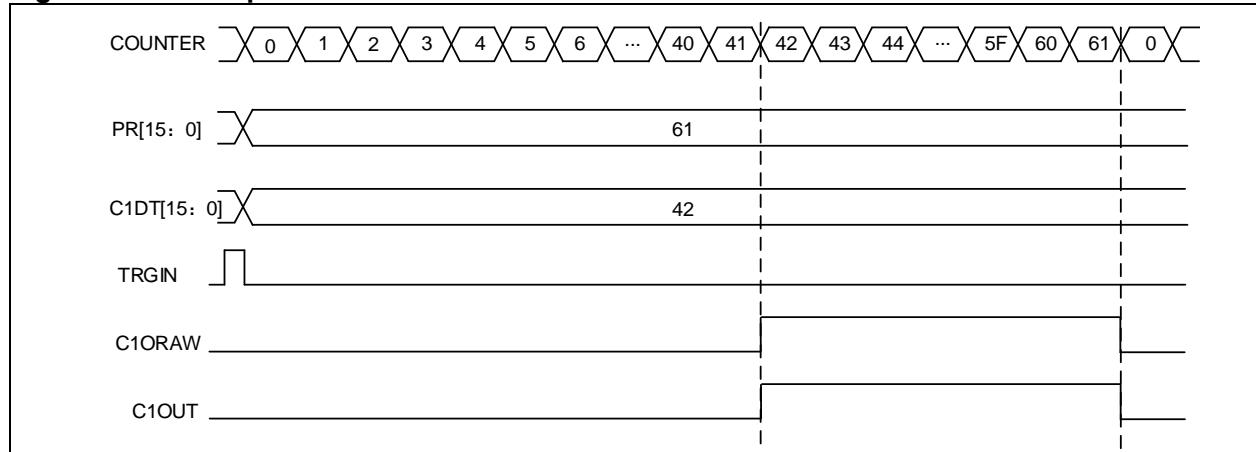
**Figure 14-68** gives an example of the combination between upcounting mode and one-pulse PWM mode B. The counter only counts only one cycle, and the output signal sends only one pulse.

**Figure 14-66 C1ORAW toggles when counter value matches the C1DT value**



**Figure 14-67 Upcounting mode and PWM mode A**



**Figure 14-68 One-pulse mode**

### Master timer event output

When TMR is selected as the master timer, the following signal sources can be selected as TRGOUT signal to output to the slave timer, by setting the PTOS bit in the TMRx\_CTRL2 register.

- PTOS=3'b000, TRGOUT outputs software overflow event (OVFSWTR bit in the TMRx\_SWEVT register).
- PTOS=3'b001, TRGOUT outputs counter enable signal.
- PTOS=3'b010, TRGOUT outputs counter overflow event.
- PTOS=3'b011, TRGOUT outputs capture and compare event.
- PTOS=3'b100, TRGOUT outputs C1ORAW signal.
- PTOS=3'b101, TRGOUT outputs C2ORAW signal.

### Dead-time insertion

The channel 1 of the TMR15 timer contains a set of reverse channel output. This function is enabled by the CxCEN bit and its polarity is defined by CxCP. Refer to Table 14-11 for more information about the output state of CxOUT and CxCOUT.

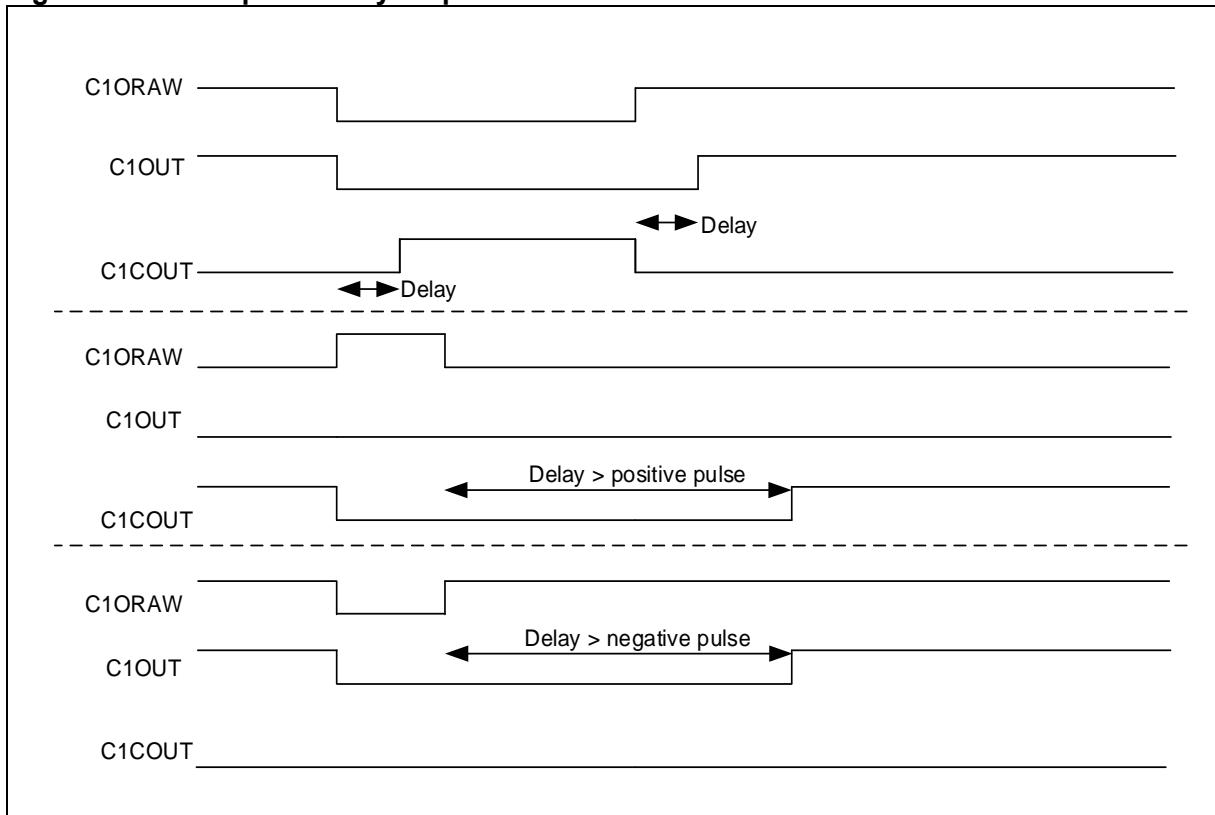
The dead-time is activated when switching to IDLEF state (OEN falling down to 0).

Setting both CxEN and CxCEN bits, and using DTC[7:0] bit to insert dead-time of different durations. After the dead-time insertion, the rising edge of the CxOUT is delayed compared to the rising edge of the reference signal; the rising edge of the CxCOUT is delayed compared to the falling edge of the reference signal.

If the delay is greater than the width of the active output, and if C1OUT and C1COUT do not generate corresponding pulses, the dead-time should be less than the width of the active output.

**Figure 14-69** gives an example of dead-time insertion when CxP=0, CxCP=0, OEN=1, CxEN=1 and CxCEN=1.

Figure 14-69 Complementary output with dead-time insertion



#### 14.4.3.5 TMR brake function

When the brake function is enabled (BRKEN=1), the CxOUT and CxCOUT are jointly controlled by OEN, FCSODIS, FCSOEN, CxIOS and CxCIOS. But, CxOUT and CxCOUT cannot be set both to active level at the same time. Please refer to [Table 14-11](#) for more details.

The brake source can be the brake input pin or a clock failure event. The polarity is controlled by the BRKV bit.

When a brake event occurs, there are the following actions:

- The OEN bit is cleared asynchronously, and the channel output state is selected by setting the FCSODIS bit. This function works even if the MCU oscillator is off.
- Once OEN=0, the channel output level is defined by the CxIOS bit. If FCSODIS=0, the timer output is disabled, otherwise, the output enable remains high.
- When complementary outputs are used:
  - The outputs are first put in reset state, that is, inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
  - If the timer clock is still active, then the dead-time generator is activated. The CxIOS and CxCIOS bits are used to program the level after dead-time. Even in this case, the CxIOS and CxCIOS cannot be driven to their active level at the same time. It should be noted that because of synchronization on OEN, the dead-time duration is usually longer than usual (around 2 clk\_tmr clock cycles)
  - If FCSODIS=0, the timer releases the enable output, otherwise, it keeps the enable output; the enable output becomes high as soon as one of the CxEV and CxCEN bits becomes high.
- If the brake interrupt or DMA request is enabled, the brake status flag is set, and a brake interrupt or DMA request can be generated.
- If AOEN=1, the OEN bit is automatically set again at the next overflow event.

*Note: When the brake input is active, the OEN cannot be set, nor the status flag, BRKIF can be cleared.*

Figure 14-70 TMR control output

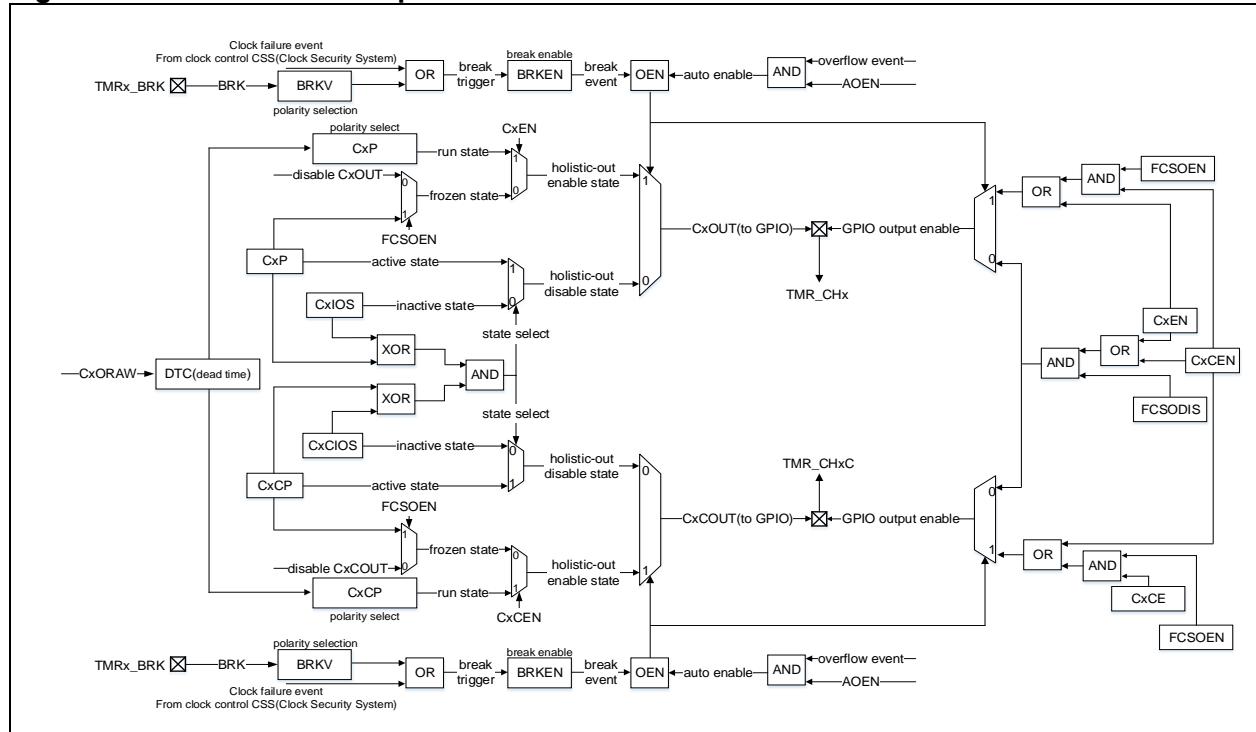
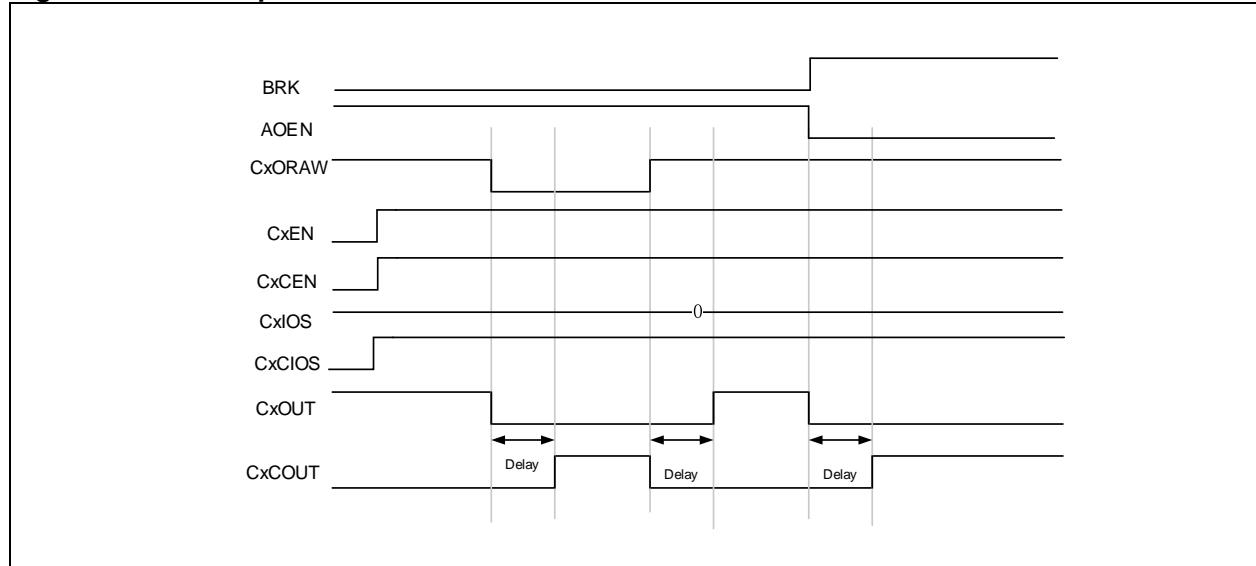


Figure 14-71 Example of TMR brake function



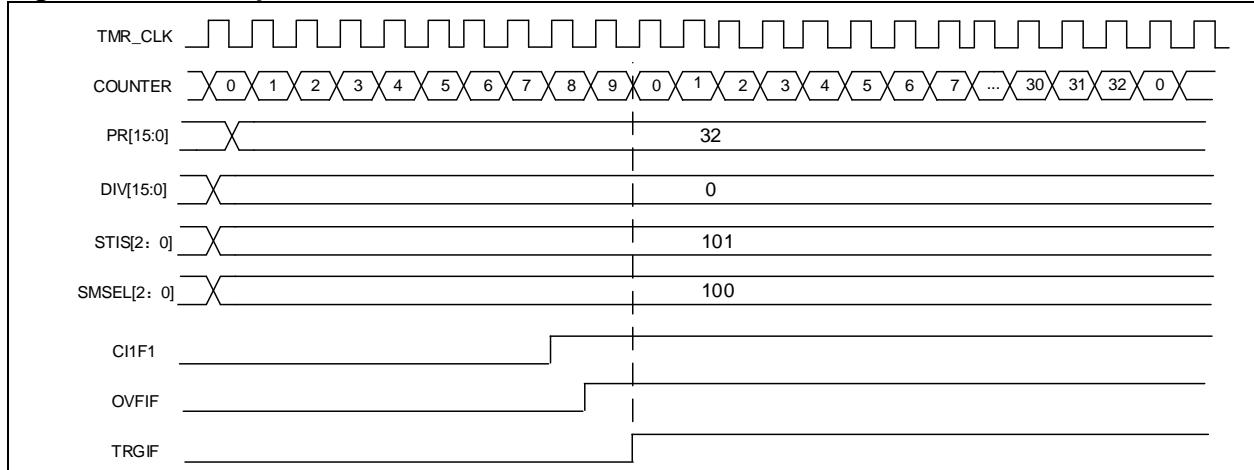
#### 14.4.3.6 TMR synchronization

The timers are linked together internally for timer synchronization. Master timer is selected by setting the PTOS[2: 0] bit; Slave timer is selected by setting the SMSEL[2: 0] bit.

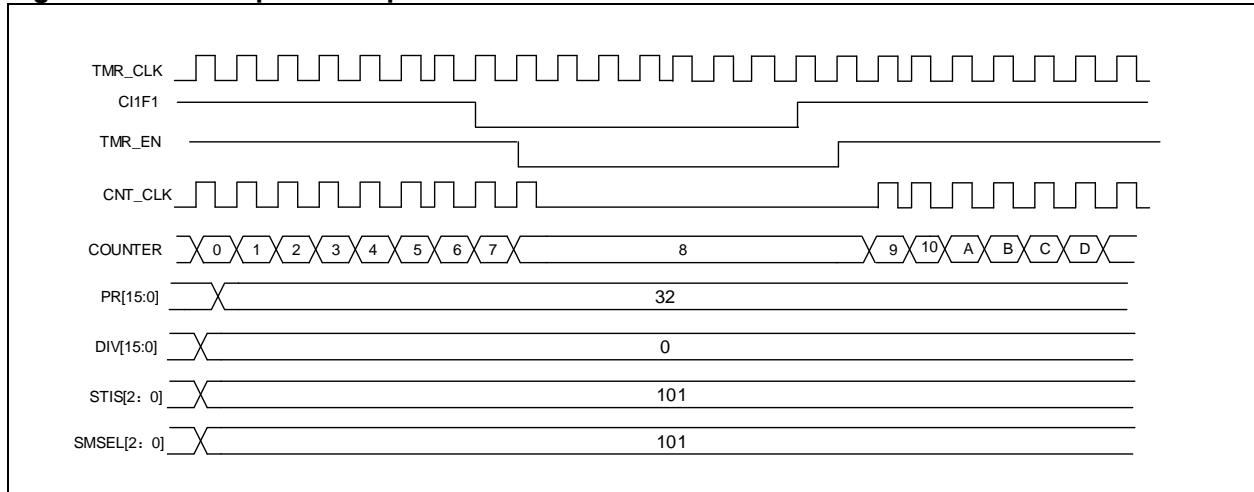
Slave mode include:

##### Slave mode: Reset mode

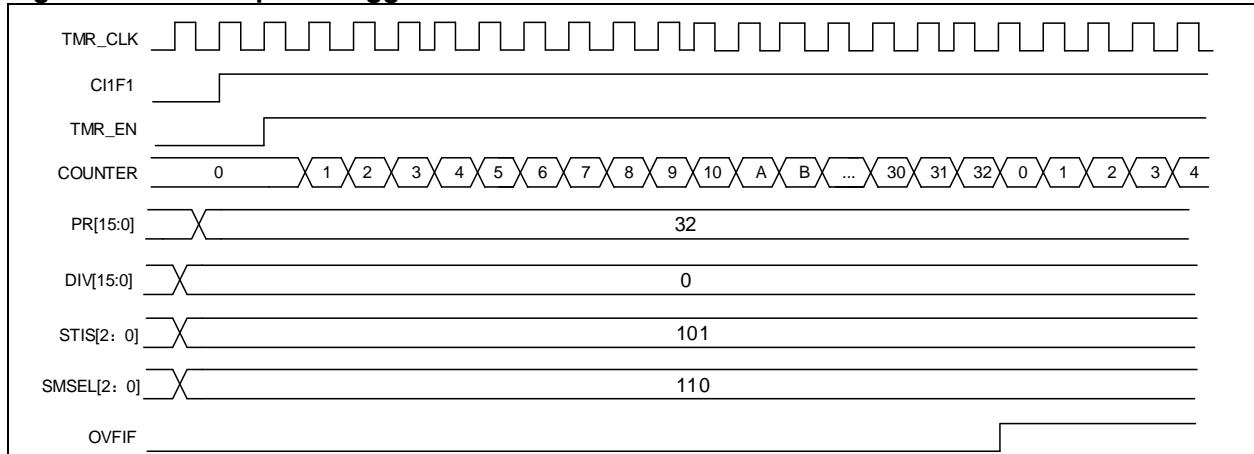
The counter and its prescaler can be reset by a selected trigger signal. An overflow event is generated when OVFS=0.

**Figure 14-72 Example of reset mode****Slave mode: Suspend mode**

In this mode, the counter is controlled by a selected trigger input. The counter starts counting when the trigger input is high and stops as soon as the trigger input is low.

**Figure 14-73 Example of suspend mode****Slave mode: Trigger mode**

The counter can start counting on the rising edge of a selected trigger input (TMR\_EN=1)

**Figure 14-74 Example of trigger mode**

Please refer to [Section 14.2.3.5](#) for more details.

### 14.4.3.7 Debug mode

When the microcontroller enters debug mode (Cortex<sup>TM</sup>-M4 core halted), the TMR15 counter stops counting by setting the TMR15\_PAUSE in the DEBUG module.

#### 14.4.4 TMR15 registers

These peripheral registers must be accessed by word (32 bits).

TMR15 registers are mapped into a 16-bit addressable space.

**Table 14-10 TMR15 register map and reset value**

Register name	Register	Reset value
TMR15_CTRL1	0x00	0x0000
TMR15_CTRL2	0x04	0x0000
TMR15_STCTRL	0x08	0x0000
TMR15_IDEN	0x0C	0x0000
TMR15_ISTS	0x10	0x0000
TMR15_SWEVT	0x14	0x0000
TMR15_CM1	0x18	0x0000
TMR15_CCTRL	0x20	0x0000
TMR15_CVAL	0x24	0x0000
TMR15_DIV	0x28	0x0000
TMR15_PR	0x2C	0x0000
TMR15_RPR	0x30	0x0000
TMR15_C1DT	0x34	0x0000
TMR15_C2DT	0x38	0x0000
TMR15_BRK	0x44	0x0000
TMR15_DMACTRL	0x48	0x0000
TMR15_DMADT	0x4C	0x0000

##### 14.4.4.1 Control register1 (TMR15\_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x0	resd	Kept at its default value
				Clock divider 00: Normal, $f_{DTS}=f_{CK\_INT}$ 01: Divided by 2, $f_{DTS}=f_{CK\_INT}/2$ 10: Divided by 4, $f_{DTS}=f_{CK\_INT}/4$ 11: Reserved
Bit 9: 8	CLKDIV	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 7	PRBEN	0x0	rw	Kept at its default value
Bit 6: 4	Reserved	0x0	resd	One cycle mode enable This bit is use to select whether to stop counting at an update event 0: The counter does not stop at an update event 1: The counter stops at an update event
Bit 3	OCMEN	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 2	OVFS	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 1	OVFEN	0x0	rw	TMR enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	

#### 14.4.4.2 Control register2 (TMR15\_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 31: 11	Reserved	0x0	resd	Kept at its default value.
Bit 10	C2IOS	0x0	rw	Channel 2 idle output state
				Channel 1 complementary idle output state
Bit 9	C1CIOS	0x0	rw	Output OFF (OEN = 0), after dead-timer generation: 0: C1COUT=0 1: C1COUT=1
				Channel 1 idle output state
Bit 8	C1IOS	0x0	rw	Output OFF (OEN = 0), after dead-timer generation 0: C1OUT=0 1: C1OUT=1
Bit 7	Reserved	0x0	resd	Kept at its default value.
				Primary TMR output selection
				This field is used to select the signal that TMR15 outputs to the slave timer.
				000: Reset
Bit 6:4	PTOS	0x0	rw	001: Enable 010: Overflow 011: Compare pulse 100: C1ORAW signal 101: C2ORAW signal
				DMA request source
Bit 3	DRS	0x0	rw	0: Channel event 1: Overflow event
				Channel control bit refresh select
				For channels with complementary output, when the channel control bit has buffer feature:
Bit 2	CCFS	0x0	rw	0: Refresh channel control bit by setting the HALL bit 1: Refresh channel control bit by setting the HALL or with the rising edge of TRGIN.
Bit 1	Reserved	0x0	resd	Kept at its default value.
				Channel buffer control
Bit 0	CBCTRL	0x0	rw	For channels with complementary output: 0: CxEN, CxCEN and CxOCTRL have no buffer feature 1: CxEN, CxCEN and CxOCTRL has buffer feature.

#### 14.4.4.3 TMR15 slave timer control register (TMR15\_STCTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0	resd	Kept at its default value
				Subordinate TMR input selection
				This field is used to select the subordinate TMR input.
				000: Internal selection 0 (IS0) 001: Internal selection 1 (IS1) 010: Internal selection 2 (IS2) 011: Internal selection 3 (IS3)
Bit 7	STIS	0x0	rw	100: C1IRAW input detector (C1INC) 101: Filtered input 1 (C1IF1) 110: Filtered input 2 (C1IF2) 111: Reserved
				Please refer to <a href="#">Table 14-9</a> for more information on ISx for each timer.

Bit 3	Reserved	0x0	resd	Kept at its default value
				Subordinate TMR mode selection 000: Slave mode is disabled 100: Reset mode — Rising edge of the TRGIN input reinitializes the counter
Bit 2: 0	SMSEL	0x0	rw	101: Suspend mode — The counter starts counting when the TRGIN is high 110: Trigger mode — A trigger event is generated at the rising edge of the TRGIN input 111: External clock mode A — Rising edge of the TRGIN input clocks the counter

#### 14.4.4.4 TMR15 DMA/interrupt enable register (TMR15\_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at its default value.
				Trigger DMA request enable
Bit 14	TDEN	0x0	rw	0: Disabled 1: Enabled
Bit 13	HALLDE	0x0	rw	HALL DMA request enable 0: Disabled 1: Enabled
Bit 12: 11	Reserved	0x0	resd	Kept at its default value.
Bit 10	C2DEN	0x0	rw	Channel 2 DMA request enable 0: Disabled 1: Enabled
Bit 9	C1DEN	0x0	rw	Channel 1 DMA request enable 0: Disabled 1: Enabled
Bit 8	OVFDEN	0x0	rw	Overflow event DMA request enable 0: Disabled 1: Enabled
Bit 7	BRKIE	0x0	rw	Brake interrupt enable 0: Disabled 1: Enabled
Bit 6	TIEN	0x0	rw	Trigger interrupt enable 0: Disabled 1: Enabled
Bit 5	HALLIEN	0x0	rw	HALL interrupt enable 0: Disabled 1: Enabled
Bit 4:3	Reserved	0x0	resd	Kept at its default value.
Bit 2	C2IEN	0x0	rw	Channel 2 interrupt enable 0: Disabled 1: Enabled
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled 1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

#### 14.4.4.5 TMR15 interrupt status register (TMR15\_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 11	Reserved	0x0	resd	Kept at its default value.
Bit 10	C2RF	0x0	rw0c	Channel 2 recapture flag Please refer to C1RF description.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing "0". 0: No capture is detected 1: Capture is detected.
Bit 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	BRKIF	0x0	rw0c	Brake interrupt flag This bit is used to indicate whether the brake input level is active or note. It is set by hardware and cleared by writing 0. 0: Inactive 1: Active
Bit 6	TRGIF	0x0	rw0c	Trigger interrupt flag This bit is set by hardware on a trigger event. It is cleared by writing "0". 0: No trigger event occurs 1: Trigger event is generated. Trigger event: an active edge is detected on TRGIN input, or any edge in suspend mode.
Bit 5	HALLIF	0x0	rw0c	HALL interrupt flag This bit is set by hardware and cleared by writing 0 at a trigger event. 0: No HALL event occurred 1: HALL event occurred. HALL event: CxEN, CxCEN and CxOCTRL have been updated.
Bit 4:3	Reserved	0x0	resd	Kept at its default value.
Bit 2	C2IF	0x0	rw0c	Channel 2 interrupt flag Please refer to C1IF description.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMR15_C1DT 0: No capture event occurred 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurred 1: Compare event is generated
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurred 1: Overflow event is generated. When OVFEN=0 and OVFS=0 of TMR15_CTRL1 register: - Overflow event is generated when OVFG=1 of the TMR15_SWEVE register

- Overflow event is generated when the counter value CVAL is re-initiated by a trigger event.

#### 14.4.4.6 TMR15 software event register (TMR15\_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	BRKSWTR	0x0	wo	<p>Brake event triggered by software This bit is set to generate a brake event by software. 0: No effect 1: Generate a brake event.</p>
Bit 6	TRGSWTR	0x0	rw	<p>Trigger event triggered by software This bit is set by software to generate a trigger event. 0: No effect 1: Generate a trigger event.</p>
Bit 5	HALLSWTR	0x0	wo	<p>HALL event triggered by software This bit is set by software to generate a HALL event. 0: No effect 1: Generate a HALL event. Note: This bit is only applicable to the channels with complementary output.</p>
Bit 4:3	Reserved	0x0	resd	Kept at its default value.
Bit 2	C2SWTR	0x0	wo	<p>Channel 2 event triggered by software Please refer to C1M description</p>
Bit 1	C1SWTR	0x0	wo	<p>Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.</p>
Bit 0	OVFSWTR	0x0	wo	<p>Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.</p>

#### 14.4.4.7 TMR15 channel mode register1 (TMR15\_CM1)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the Cxlx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

##### Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C2OSEN	0x0	rw	Channel 2 output switch enable
Bit 14: 12	C2OCTRL	0x0	rw	Channel 2 output control
Bit 11	C2OBEN	0x0	rw	Channel 2 output buffer enable
Bit 10	C2OIEN	0x0	rw	Channel 2 output enable immediately
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS register.

				Channel 1 output switch enable
Bit 7	C1OSEN	0x0	rw	0: C1ORAW is not affected by EXT input 1: When EXT input level is high, C1ORAW is cleared.
				Channel 1 output control This field defines the behavior of the original signal C1ORAW. 000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMRx_CVAL=TMR15_C1DT 010: C1ORAW is low when TMRx_CVAL=TMR15_C1DT 011: Switch C1ORAW level when TMR15_CVAL=TMR15_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high.
Bit 6: 4	C1OCTRL	0x0	rw	110: PWM mode A – OWCDIR=0, C1ORAW is high once TMR15_C1DT>TMR15_CVAL, else low; – OWCDIR=1, C1ORAW is low once TMR15_C1DT < TMR15_CVAL, else high; 111: PWM mode B – OWCDIR=0, C1ORAW is low once TMR15_C1DT > TMR15_CVAL, else high; – OWCDIR=1, C1ORAW is high once TMR15_C1DT < TMR15_CVAL, else low. <i>Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.</i>
Bit 3	C1OBEN	0x0	rw	Channel 1 output buffer enable 0: Buffer function of TMR15_C1DT is disabled. The new value written to the TMR15_C1DT takes effect immediately. 1: Buffer function of TMR15_C1DT is enabled. The value to be written to the TMR15_C1DT is stored in the buffer register, and can be sent to the TMR15_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw	Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

**Input capture mode:**

Bit	Register	Reset value	Type	Description
Bit 15: 12	C2DF	0x0	rw	Channel 2 digital filter
Bit 11: 10	C2IDIV	0x0	rw	Channel 2 input divider
				Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0':
Bit 9: 8	C2C	0x0	rw	00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
				Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events. 0000: No filter, sampling is done at $f_{DTS}$ 1000: $f_{SAMPLING} = f_{DTS}/8$ , N=6 0001: $f_{SAMPLING} = f_{CK\_INT}$ , N=2 1001: $f_{SAMPLING} = f_{DTS}/8$ , N=8 0010: $f_{SAMPLING} = f_{CK\_INT}$ , N=4 1010: $f_{SAMPLING} = f_{DTS}/16$ , N=5 0011: $f_{SAMPLING} = f_{CK\_INT}$ , N=8 1011: $f_{SAMPLING} = f_{DTS}/16$ , N=6 0100: $f_{SAMPLING} = f_{DTS}/2$ , N=6 1100: $f_{SAMPLING} = f_{DTS}/16$ , N=8 0101: $f_{SAMPLING} = f_{DTS}/2$ , N=8 1101: $f_{SAMPLING} = f_{DTS}/32$ , N=5 0110: $f_{SAMPLING} = f_{DTS}/4$ , N=6 1110: $f_{SAMPLING} = f_{DTS}/32$ , N=6 0111: $f_{SAMPLING} = f_{DTS}/4$ , N=8 1111: $f_{SAMPLING} = f_{DTS}/32$ , N=8
Bit 7: 4	C1DF	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
				Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0':
Bit 1: 0	C1C	0x0	rw	00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

#### 14.4.4.8 TMR15 channel control register (TMR15\_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	C2CP	0x0	rw	Channel 2 complementary polarity This bit defines the active edge for input signals. Refer to C1P description.
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	C2P	0x0	rw	Channel 2 polarity Please refer to C1P description.
Bit 4	C2EN	0x0	rw	Channel 2 enable Please refer to C1EN description.
Bit 3	C1CP	0x0	rw	Channel 1 complementary polarity This bit defines the active edge for input signals. Refer to C1P description.
Bit 2	C1CEN	0x0	rw	Channel 1 complementary enable 0: Channel 1 complementary output disabled 1: Channel 1 complementary output enabled
Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured in output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured in input mode: It is C1CP/C1P bits that define the active edge of an input signal. 00: C1IN rising edge is active. When used as external trigger, C1IN is not inverted. 01: C1IN falling edge is active. When used as external trigger, C1IN is inverted. 10: Reserved. 11: C1IN rising edge and falling edge are both active. When used as external trigger, C1IN is not inverted.
Bit0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

**Table 14-11 Complementary output channel CxOUT and CxCOUT control bits with brake function**

		Control bit			Output state (1)	
OEN bit	FCSODIS bit	FCSOEN bit	CxEN bit	CxCEN bit	CxOUT output state	CxCOUT output state
1	X		0	0	Output disabled (no driven by the timer) CxOUT=0, Cx_EN=0	Output disabled (no driven by the timer) CxCOUT=0, CxCEN=0
			0	0	Output disabled (no driven by the timer) CxOUT=0, Cx_EN=0	CxORAW + polarity, CxOUT= CxORAW xor CxCP, CxCEN=1
			0	1	CxORAW+ polarity CxOUT= CxORAW xor CxP, Cx_EN=1	Output disabled (no driven by the timer) CxCOUT=0, CxCEN=0
			0	1	CxORAW+polarity+dead- time, Cx_EN=1	CxORAW inverted+polarity+dead- time, CxCEN=1
			1	0	Output disabled (no driven by the timer) CxOUT=CxP, Cx_EN=0	Output disabled (no driven by the timer) CxOUT=CxP, CxCEN=0
			1	0	Off-state (Output enabled with inactive level) CxOUT=CxP, Cx_EN=1	CxORAW + polarity, CxOUT= CxORAW xor CxP, CxEN=1
			1	1	CxORAW + polarity, CxOUT= CxORAW xor CxP, Cx_EN=1	Off-state (Output enabled with inactive level) CxOUT=CxP, CxCEN=1
			1	1	CxORAW+ polarity+dead- time, Cx_EN=1	CxORAW inverted+polarity+dead- time, CxCEN=1
0	X		0	0	Output disabled (corresponding IO disconnected from timer, and IO floating)	
			0	1	Asynchronously: CxOUT=CxP, Cx_EN=0, CxOUT=CxP, CxCEN=0;	
			0	0	If the clock is present: after a dead-time, CxOUT=CxIOS, CxCOUT=CxCIOS, assuming that CxIOS and CxCIOS do not correspond to CxOUT and CxOUT active level.	
			0	1	CxEN=CxCEN=0: Output disabled (corresponding IO disconnected from timer, and IO floating)	
			1	0	In other cases, Off-state (corresponding channel output inactive level)	
			1	1	Asynchronously: CxOUT =CxP, Cx_EN=1, CxOUT=CxP, CxCEN=1;	
			1	0	If the clock is present: after a dead-time, CxOUT=CxIOS, CxCOUT=CxCIOS, assuming that CxIOS and CxCIOS do not correspond to CxOUT and CxOUT active level.	
			1	1		

*Note: If the two outputs of a channel are not used (CxEN = CxCEN = 0), CxIOS, CxCIOS, CxP and CxCP must be cleared.*

*Note: The state of the external I/O pins connected to the complementary CxOUT and CxCOUT channels depends on the CxOUT and CxCOUT channel state and the GPIO and the IOMUX registers.*

#### 14.4.4.9 TMR15 Counter value (TMR15\_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0	rw	Counter value

#### 14.4.4.10 TMR15 Division value (TMR15\_DIV)

Bit	Register	Reset value	Type	Description
				Divider value
Bit 15: 0	DIV	0x0000	rw	The counter clock frequency $f_{CK\_CNT} = f_{TMR\_CLK} / (\text{DIV}[15:0] + 1)$ . The value of this register is moved to the actual prescaler register when an overflow event occurs.

#### 14.4.4.11 TMR15 period register (TMR15\_PR)

Bit	Register	Reset value	Type	Description
				Period value
Bit 15: 0	PR	0x0000	rw	This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

#### 14.4.4.12 TMR15 repetition period register (TMR15\_RPR)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x0	rw	Kept at its default value.
Bit 7: 0	RPR	0x00	rw	Repetition of period value This field is used to reduce the generation rate of overflow events. An overflow event is generated when the repetition counter reaches 0.

#### 14.4.4.13 TMR15 channel 1 data register (TMR15\_C1DT)

Bit	Register	Reset value	Type	Description
				Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN)
Bit 15: 0	C1DT	0x0000	rw	When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effect immediately depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

#### 14.4.4.14 TMR15 channel 2 data register (TMR15\_C2DT)

Bit	Register	Reset value	Type	Description
				Channel 2 data register When the channel 2 is configured as input mode: The C2DT is the CVAL value stored by the last channel 2 input event (C2IN)
Bit 15: 0	C2DT	0x0000	rw	When the channel 2 is configured as output mode: C2DT is the value to be compared with the CVAL value. Whether the written value takes effect immediately depends on the C2OBEN bit, and the corresponding output is generated on C2OUT as configured.

#### 14.4.4.15 TMR15 brake register (TMR15\_BRK)

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0	resd	Kept at its default value.
Bit 19: 16	BKF	0x0	rw	<p>Brake input filter</p> <p>This field is used to configure the filter for brake input. If the number of filter is N, it indicates that the input edge can pass through the filter only after N sampling events.</p> <ul style="list-style-type: none"> <li>0000: No filter. <math>f_{DTS}</math> (sampling frequency)</li> <li>1000: <math>f_{SAMPLING} = f_{DTS}/8</math>, N=6</li> <li>0001: <math>f_{SAMPLING} = f_{CK\_INT}</math>, N=2</li> <li>1001: <math>f_{SAMPLING} = f_{DTS}/8</math>, N=8</li> <li>0010: <math>f_{SAMPLING} = f_{CK\_INT}</math>, N=4</li> <li>1010: <math>f_{SAMPLING} = f_{DTS}/16</math>, N=5</li> <li>0011: <math>f_{SAMPLING} = f_{CK\_INT}</math>, N=8</li> <li>1011: <math>f_{SAMPLING} = f_{DTS}/16</math>, N=6</li> <li>0100: <math>f_{SAMPLING} = f_{DTS}/2</math>, N=6</li> <li>1100: <math>f_{SAMPLING} = f_{DTS}/16</math>, N=8</li> <li>0101: <math>f_{SAMPLING} = f_{DTS}/2</math>, N=8</li> <li>1101: <math>f_{SAMPLING} = f_{DTS}/32</math>, N=5</li> <li>0110: <math>f_{SAMPLING} = f_{DTS}/4</math>, N=6</li> <li>1110: <math>f_{SAMPLING} = f_{DTS}/32</math>, N=6</li> <li>0111: <math>f_{SAMPLING} = f_{DTS}/4</math>, N=8</li> <li>1111: <math>f_{SAMPLING} = f_{DTS}/32</math>, N=8</li> </ul>
Bit 15	OEN	0x0	rw	<p>Output enable</p> <p>This bit acts on the channels as output. It is used to enable CxOUT and CxCOUT outputs.</p> <ul style="list-style-type: none"> <li>0: Disabled</li> <li>1: Enabled</li> </ul>
Bit 14	AOEN	0x0	rw	<p>Automatic output enable</p> <p>OEN is set automatically at an overflow event.</p> <ul style="list-style-type: none"> <li>0: Disabled</li> <li>1: Enabled</li> </ul>
Bit 13	BRKV	0x0	rw	<p>Brake input validity</p> <p>This bit is used to select the active level of a brake input.</p> <ul style="list-style-type: none"> <li>0: Brake input is active low.</li> <li>1: Brake input is active high.</li> </ul>
Bit 12	BRKEN	0x0	rw	<p>Brake enable</p> <p>This bit is used to enable brake input.</p> <ul style="list-style-type: none"> <li>0: Brake input is disabled.</li> <li>1: Brake input is enabled.</li> </ul>
Bit 11	FCSOEN	0x0	rw	<p>Frozen channel status when holistic output enable</p> <p>This bit acts on the channels that have complementary output. It is used to set the channel state when the timer is inactive and MOEN=1.</p> <ul style="list-style-type: none"> <li>0: CxOUT/CxCOUT outputs are disabled.</li> <li>1: CxOUT/CxCOUT outputs are enabled. Output inactive level.</li> </ul>
Bit 10	FCSODIS	0x0	rw	<p>Frozen channel status when holistic output disable</p> <p>This bit acts on the channels that have complementary output. It is used to set the channel state when the timer is inactive and MOEN=0.</p> <ul style="list-style-type: none"> <li>0: CxOUT/CxCOUT outputs are disabled.</li> <li>1: CxOUT/CxCOUT outputs are enabled. Output idle level.</li> </ul>
Bit 9: 8	WPC	0x0	rw	<p>Write protection configuration</p> <p>This field is used to enable write protection.</p> <ul style="list-style-type: none"> <li>00: Write protection is OFF.</li> </ul>

01: Write protection level 3, and the following bits are write protected:

TMR1\_BRK: DTC, BRKEN, BRKV and AOEN

TMR1\_CTRL2: CxIOS and CxCIOS

10: Write protection level 2. The following bits and all bits in level 3 are write protected:

TMR1\_CCTRL: CxP and CxCP

TMR1\_BRK: FCSODIS and FCSOEN

11: Write protection level 1. The following bits and all bits in level 2 are write protected:

TMR1\_CMx: C2OCTRL and C2OBEN

Note: When WPC>0, its content remains frozen until the next system reset.

Dead-time configuration

This field defines the duration of the dead-time insertion.

The 3-bit MSB of DTC[7: 0] is used for function selection:

0xx: DT = DTC [7: 0] \* TDTS

10x: DT = (64+ DTC [5: 0]) \* TDTS \* 2

110: DT = (32+ DTC [4: 0]) \* TDTS \* 8

111: DT = (32+ DTC [4: 0]) \* TDTS \* 16

Bit 7: 0	DTC	0x00	rw
----------	-----	------	----

*Note: Based on lock configuration, AOEN, BRKV, BRKEN, FCSODIS, FCSOEN and DTC[7:0] can all be write protected. Thus it is necessary to configure write protection when writing to the TMRx\_BRK register for the first time.*

#### 14.4.4.16 TMR15 DMA control register (TMR15\_DMACTRL)

Bit	Register	Reset value	Type	Description
Bit 15:13	Reserved	0x0	resd	Kept at its default value.
Bit 12:8	DTB	0x00	rw	DMA transfer bytes This field defines the number of DMA transfers: 00000: 1 byte      00001: 2 bytes 00010: 3 bytes      00011: 4 bytes ..... 10000: 17 bytes      10001: 18 bytes
Bit 7:5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	ADDR	0x00	rw	DMA transfer address offset This field defines an offset starting from the address of the TMR15_CTRL1 register: 00000: TMR15_CTRL1 00001: TMR15_CTRL2 00010: TMR15_STCTRL .....

#### 14.4.4.17 TMR15 DMA data register (TMR15\_DMADT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DMADT	0x0000	rw	DMA data register A write/read operation to the DMADT register accesses any TMR register located at the following address: TMR15 peripheral address + ADDR*4 to TMR15 peripheral address + ADDR*4 + DTB*4

## 14.5 General-purpose timer (TMR16 and TMR17)

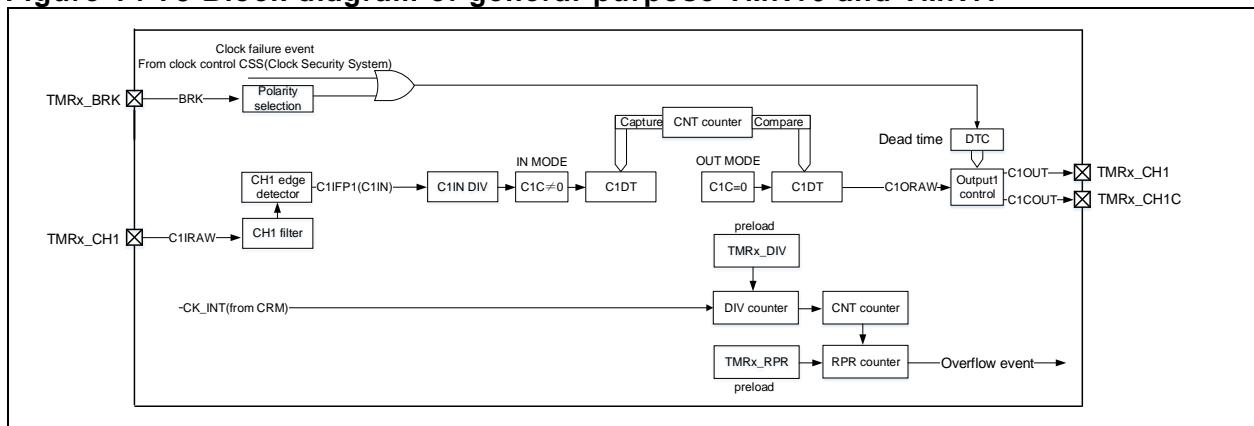
### 14.5.1 TMR16 and TMR17 introduction

The general-purpose timers TMR16 and TMR17 consist of a 16-bit counter supporting upcounting mode. Each of them has a capture/compare register, and an independent channels to achieve dead-time insert, input capture and programmable PWM output.

### 14.5.2 TMR16 and TMR17 main features

- Source of count clock : internal clock, external clock and internal trigger
- 16-bit upcounter and 8-bit repetition counter
- 1 x independent channel for input capture, output compare, PWM generation, one-pulse mode output and dead-time insertion
- 1 x independent channel for complementary output
- TMR brake feature support
- Synchronization control between timers
- Interrupt/DMA generation on the overflow event, trigger event and channel event
- Support TMR burst DMA transfer

**Figure 14-75 Block diagram of general-purpose TMR16 and TMR17**

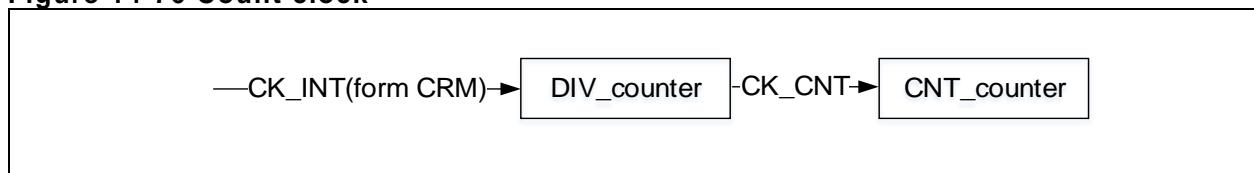


### 14.5.3 TMR16 and TMR17 functional overview

#### 14.5.3.1 Count clock

The counters of TMR16 and TMR17 can be clocked by the internal clock (CK\_INT) only.

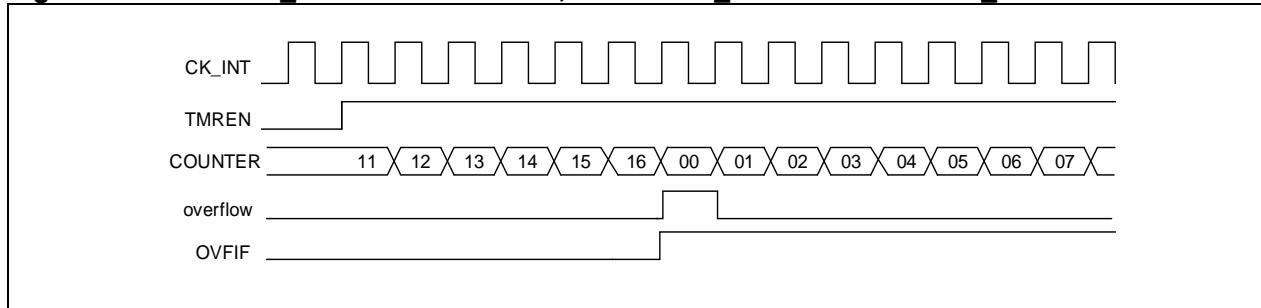
**Figure 14-76 Count clock**



#### Internal clock (CK\_INT)

By default, the CK\_INT divided by a prescaler is used to drive the counter to start counting. The configuration process is as follows:

- Set the TMRx\_DIV register to set the counting frequency;
- Set the TMRx\_PR register to set the counting period;
- Set the TMREN bit in the TMRx\_CTRL1 register to enable the counter.

**Figure 14-77 Use CK\_INT to drive counter, with TMRx\_DIV=0x0 and TMRx\_PR=0x16**

### 14.5.3.2 Counting mode

The timer (TMR16 and TMR17) supports several counting modes to meet different application scenarios. Each timer has an internal 16-bit up counter.

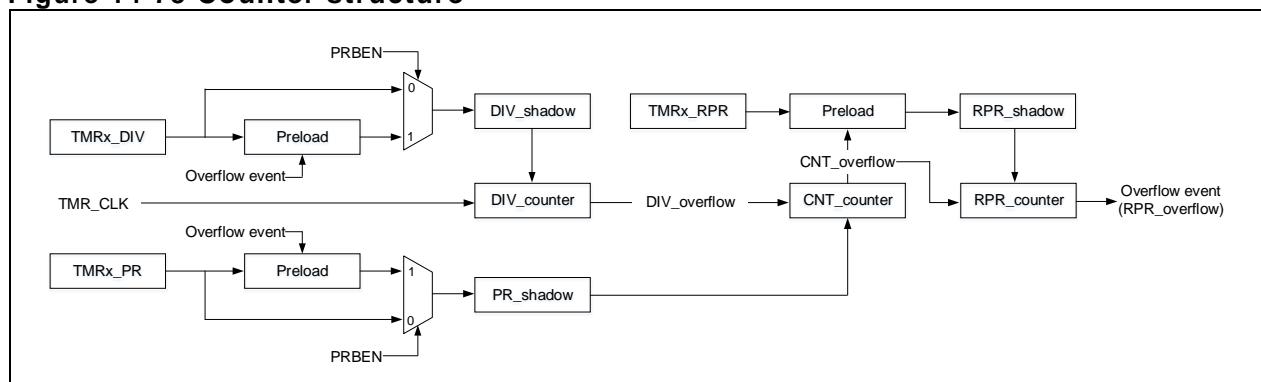
The TMRx\_PR register is used to configure the counting period. The value in the TMRx\_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx\_PR register is transferred to the shadow register only at an overflow event.

The TMRx\_DIV register is used to configure the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to the TMRx\_PR register, when the periodic buffer is enabled, the value in the TMRx\_DIV register is updated to the shadow register at an overflow event.

Reading the TMRx\_CNT register returns to the current counter value, and writing to the TMRx\_CNT register updates the current counter value to the value being written.

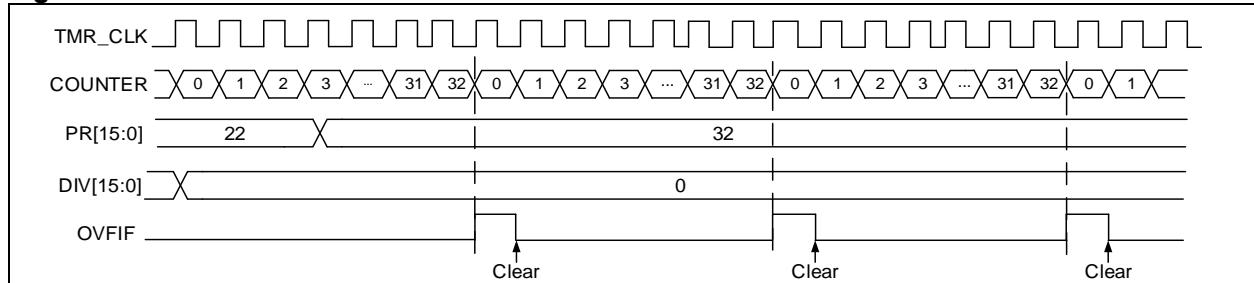
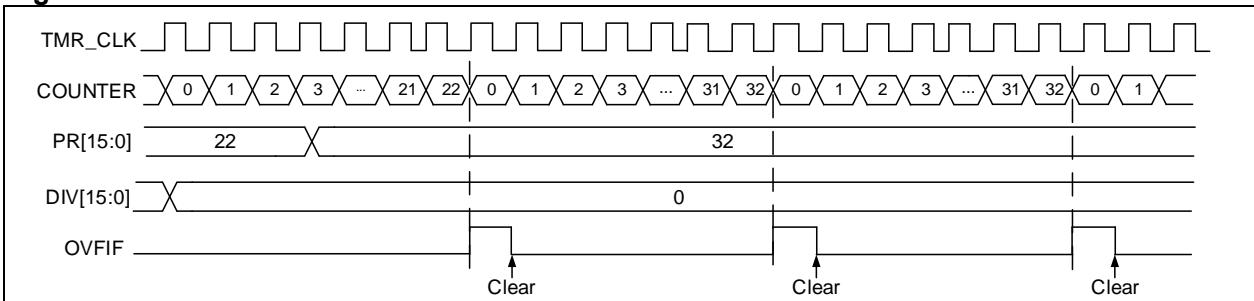
An overflow event is generated by default. Set OVREN=1 in the TMRx\_CTRL1 to disable generation of update events. The OVFS bit in the TMRx\_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only counter overflow/underflow triggers an overflow event.

Setting the TMREN bit (TMREN=1) enables the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR\_EN is set 1 clock cycle after the TMREN is set.

**Figure 14-78 Counter structure**

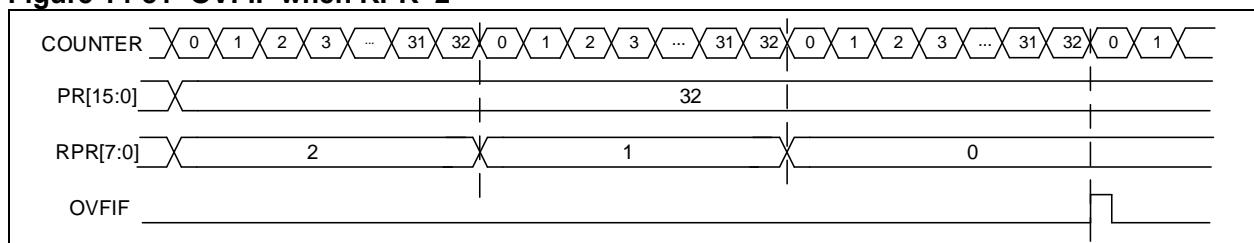
#### Upcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the register to enable upcounting mode. In this mode, the counter counts from 0 to the value programmed in the TMRx\_PR register, restarts from 0, and generates a counter overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the register is no longer reloaded with the preload and re-loaded value after counter overflow occurs, otherwise, the prescaler and re-loaded value will be updated at an overflow event.

**Figure 14-79 Overflow event when PRBEN=0****Figure 14-80 Overflow event when PRBEN=1**

#### Repetition counter mode:

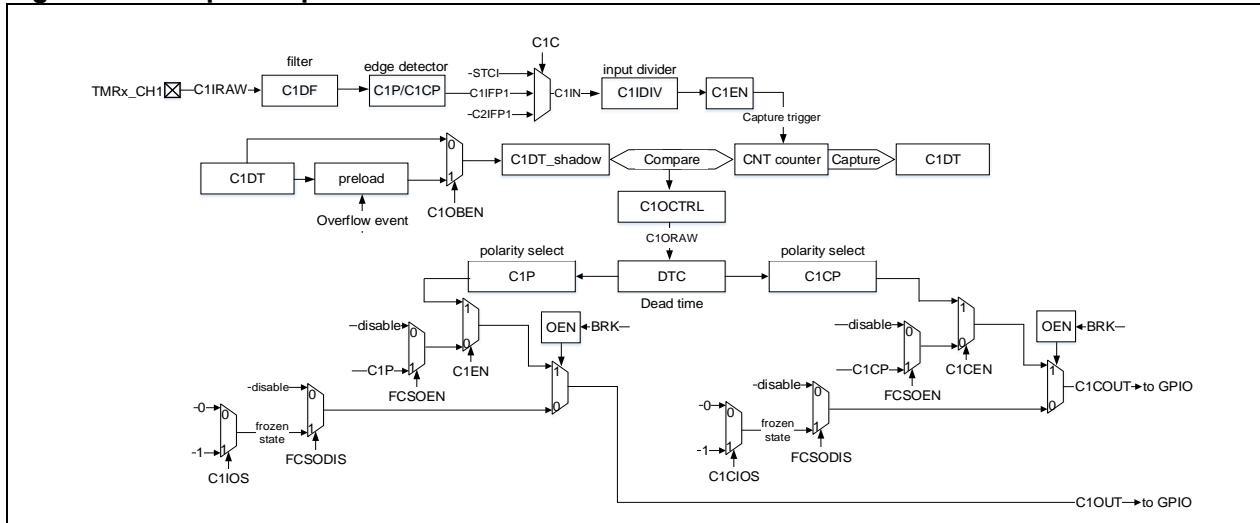
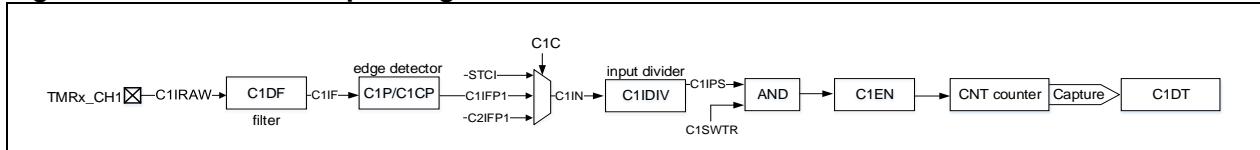
The TMRx\_RPR register is used to configure the counting period of repetition counter. The repletion counter mode is enabled when the repetition counter value is not equal to 0. In this mode, an overflow event occurs once at every counter overflow ( $RPR[7:0]+1$ ), and the repetition counter is decremented at each counter overflow. An overflow event is generated only when the repetition counter reaches 0. The frequency of the overflow event can be adjusted by setting the repetition counter value.

**Figure 14-81 OVFIF when RPR=2**

### 14.5.3.3 TMR input function

Each of timers (TMR16 and TMR17) has one independent channel that can be configured as input or output. As input, each channel input signal is processed as below:

- TMRx\_CHx outputs the pre-processed CxIRAW. Set the C1INSEL bit to select TMRx\_CHx as the source of CxIRAW.
- CxIRAW inputs to the digital filter and outputs a filtered signal CxIF. Set the sampling frequency and sampling times of digital filter by setting the CxDI bit.
- CxIF inputs the edge detector and outputs the signal CxIFPx after edge selection. The edge selection is controlled by CxP and CxCp bits, and can be selected as rising edge, falling edge or both edges active.
- CxIFPx inputs capture signal selector and outputs the signal CxIN after selection. The capture signal selector is controlled by the CxC bits. The source of CxIN can be set as CxIFPx. The CyIFPy from channel y and handled by channel x edge detector (for example, the C1IFP2 is the C1IFP1 from channel 1 and then handled by channel 2 edge detector).
- CxIN outputs the signal CxIPS through the input channel divider. The division factor is set to “no division”, “divided by 2”, “divided by 4” or “divided by 8” by setting the CxIDIV bit.

**Figure 14-82 Input/output channel 1 main circuit****Figure 14-83 Channel 1 input stage**

### Input mode

In input mode, the TMR<sub>x</sub>\_CxDT registers latch the current counter values after the selected trigger signal is detected, and the capture compare interrupt flag bit (CxIF) is set. An interrupt or a DMA request will be generated if the CxIEN and CxDEN bits are enabled. If the selected trigger signal is detected when the CxIF is set to 1, a capture overflow event occurs. The TMR<sub>x</sub>\_CxDT register overwrites the recorded value with the current counter value, and the CxRF is set to 1.

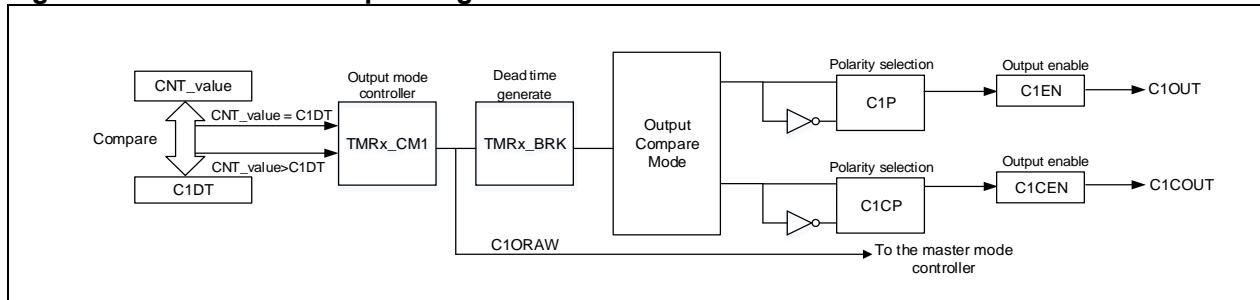
To capture the rising edge of C1IN input, following the configuration procedure mentioned below:

- Set C1C=01 in the TMR<sub>x</sub>\_CM1 register to select the C1IN as channel 1 input
- Set the filter bandwidth of C1IN signal (CxDF[3: 0])
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMR<sub>x</sub>\_CCTRL register
- Program the capture frequency division of C1IN signal (C1DIV[1: 0])
- Enable channel 1 input capture (C1EN=1)

If needed, enable the relevant interrupt or DMA request by setting the C1IEN bit in the TMR<sub>x</sub>\_IDEN register or the C1DEN bit in the TMR<sub>x</sub>\_IDEN register

### 14.5.3.4 TMR output function

The TMR output consists of a comparator and an output controller. It is used to program the period, duty cycle and polarity of the output signal.

**Figure 14-84 Channel 1 output stage**

### Output mode

Write CxC[2: 0]≠2'b00 to configure the channel as output to implement multiple output modes. In this case, the counter value is compared with the value in the TMR<sub>x</sub>\_CxDT register, and the intermediate

signal CxORAW is generated according to the output mode selected by CxOCTRL[2: 0], which is sent to IO after being processed by the output control circuit. The period of the output signal is configured by the TMRx\_PR register, while the duty cycle by the TMRx\_CxDT register.

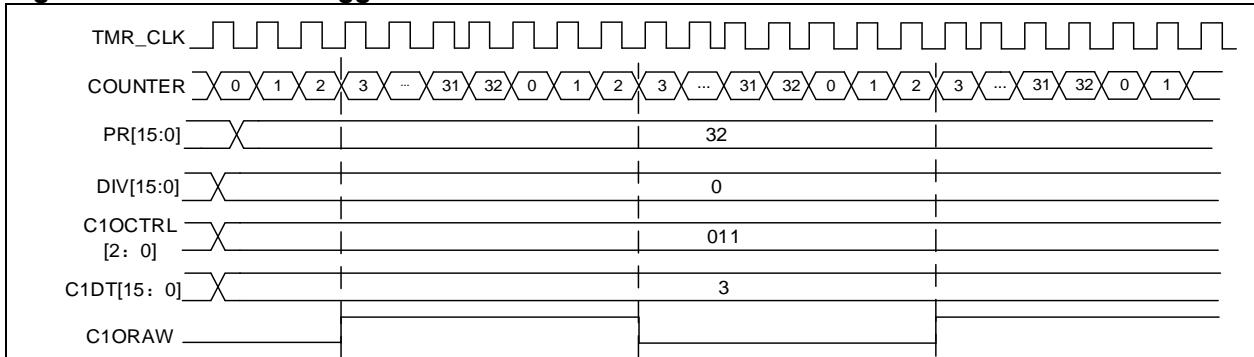
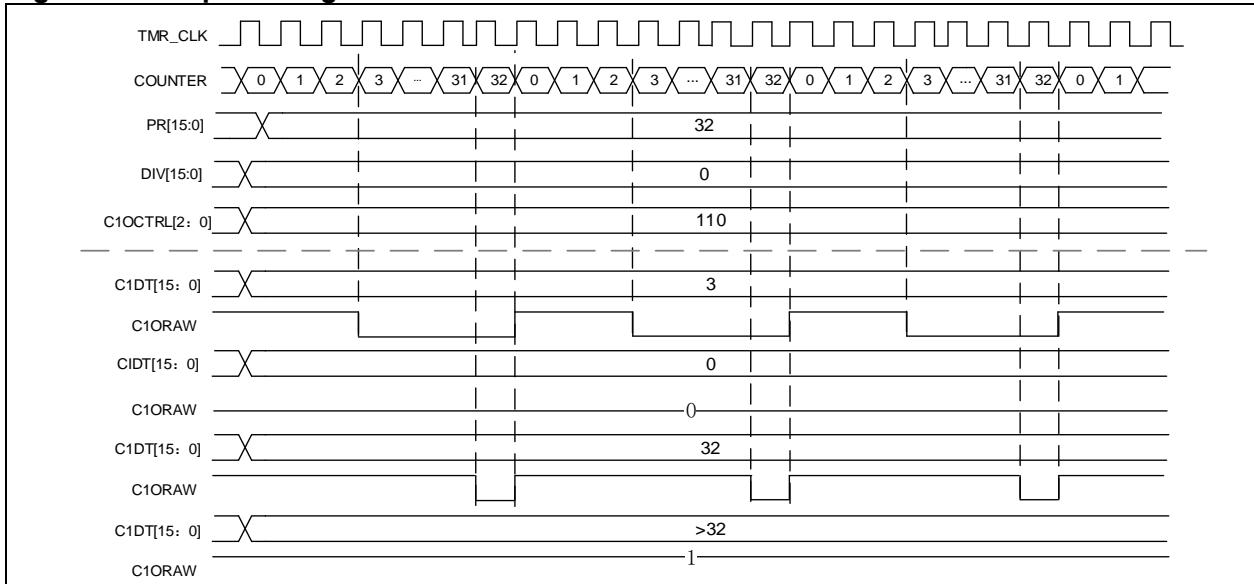
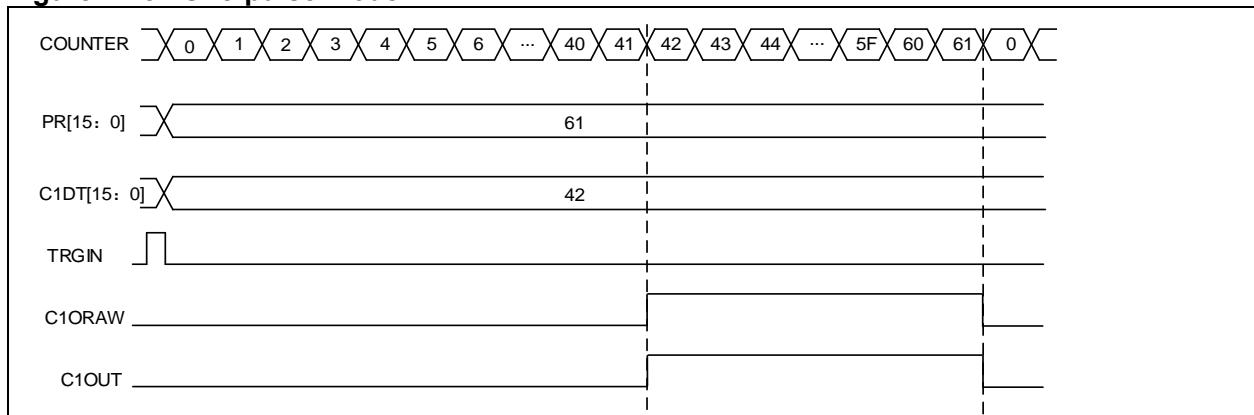
Output compare modes include:

- **PWM mode A:** Set CxOCTRL=3'b110 to enable PWM mode A. In upcounting, when TMRx\_C1DT>TMRx\_CVAL, C1ORAW outputs high; otherwise, outputs low. In downcounting, when TMRx\_C1DT<TMRx\_CVAL, C1ORAW outputs low; otherwise, outputs high. To set PWM mode A, the following process is recommended:
  - Set the TMRx\_PR register to set PWM period;
  - Set the TMRx\_CxDT register to set PWM duty cycle;
  - Set CxOCTRL=3'b110 in the TMRx\_CM1/CM2 register to set output mode as PWM mode A;
  - Set the TMRx\_DIV register to set the counting frequency;
  - Set the TWCMSEL[1:0] bit in the TMRx\_CTRL1 register to set the count mode;
  - Set CxP bit and CxCOP bit in the TMRx\_CCTRL register to set output polarity;
  - Set CxEN bit and CxCEN bit in the TMRx\_CCTRL register to enable channel output;
  - Set the OEN bit in the TMRx\_BRK register to enable TMRx output;
  - Set the corresponding GPIO of TMR output channel as the multiplexed mode;
  - Set the TMREN bit in the TMRx\_CTRL1 register to enable TMRx counter.
- **PWM mode B:** Set CxOCTRL=3'b111 to enable PWM mode B. In upcounting, when TMRx\_C1DT>TMRx\_CVAL, C1ORAW outputs low; otherwise, outputs high. In downcounting, when TMRx\_C1DT<TMRx\_CVAL, C1ORAW outputs high; otherwise, outputs low.
- **Forced output mode:** Set CxOCTRL=3'b100/101 to enable forced output mode. In this case, the CxORAW is forced to be the programmed level, irrespective of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.
- **Output compare mode:** Set CxOCTRL=3'b001/010/011 to enable output compare mode. In this case, when the counter value matches the value of the CxDT register, the CxORAW is forced high (CxOCTRL=3'b001), low (CxOCTRL=3'b010) or toggling (CxOCTRL=3'b011).
- **One-pulse mode:** This is a particular case of PWM mode. Set OCEN=1 to enable one-pulse mode. In this mode, the comparison match is performed in the current counting period. The TMREN bit is cleared as soon as the current counting is completed. Therefore, only one pulse is output. When configured as in upcounting mode, the configuration must follow the rule: CVAL<CxDT≤PR; in downcounting mode, CVAL>CxDT is required.
- **Fast output mode:** Set CXOIEN=1 to enable this mode. If enabled, the CxORAW signal will not change when the counter value matches the CxDT, but at the beginning of the current counting period. In other words, the comparison result is advanced, so the comparison result between the counter value and the TMRx\_CxDT register will determine the level of CxORAW in advance.

**Figure 14-85** gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

**Figure 14-86** gives an example of the combination between upcounting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

**Figure 14-87** gives an example of the combination between upcounting mode and one-pulse PWM mode B. The counter only counts only one cycle, and the output signal sends only one pulse.

**Figure 14-85 C1ORAW toggles when counter value matches the C1DT value****Figure 14-86 Upcounting mode and PWM mode A****Figure 14-87 One-pulse mode**

### Dead-time insertion

The channel 1 of the TMR16 and TMR17 contains a set of reverse channel output. This function is enabled by the CxCEN bit and its polarity is defined by CxCP. Refer to Table 14-13 for more information about the output state of CxOUT and CxCOUT.

The dead-time is activated when switching to IDLE state (OEN falling down to 0).

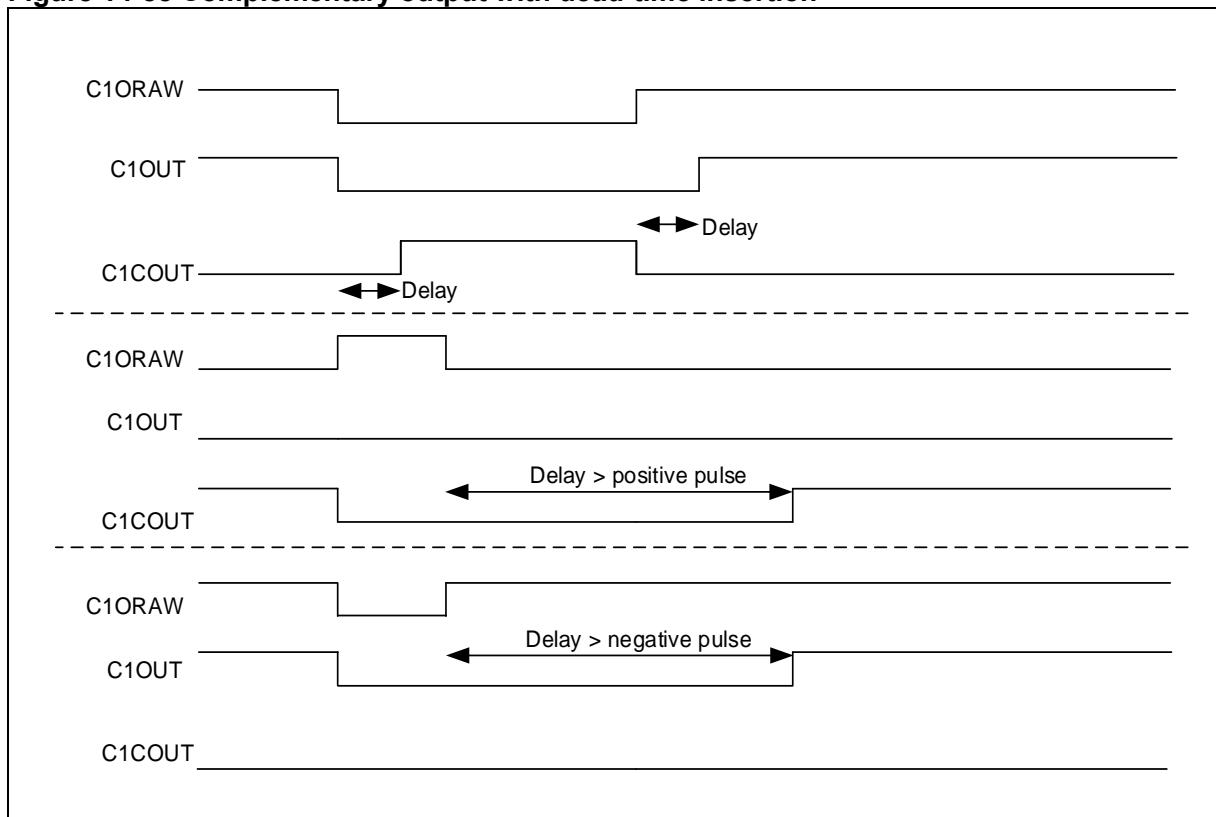
Setting both CxEN and CxCEN bits, and using DTC[7:0] bit to insert dead-time of different durations. After the dead-time insertion, the rising edge of the CxOUT is delayed compared to the rising edge of the reference signal; the rising edge of the CxCOUT is delayed compared to the falling edge of the reference signal.

If the delay is greater than the width of the active output, and if C1OUT and C1COUT do not generate corresponding pulses, the dead-time should be less than the width of the active output.

**Figure 14-88** gives an example of dead-time insertion when CxP=0, CxCP=0, OEN=1, CxEN=1 and

CxCEN=1.

**Figure 14-88 Complementary output with dead-time insertion**



#### 14.5.3.5 TMR brake function

When the brake feature is enabled (BRKEN=1), the CxOUT and CxCOUT are jointly controlled by OEN, FCSODIS, FCSOEN, CxIOS and CxCIOS. But, CxOUT and CxCOUT cannot be set both to active level at the same time. Please refer to [Table 14-13](#) for more details.

The brake sources can be the brake input pin or a clock failure event. The polarity is controlled by the BRKV bit.

When a brake event occurs, there are the following actions:

- The OEN bit is cleared asynchronously, and the channel output state is selected by setting the FCSODIS bit. This function works even if the MCU oscillator is off.
- Once OEN=0, the channel output level is defined by the CxIOS bit. If FCSODIS=0, the timer output is disabled, otherwise, the output enable remains high.
- When complementary outputs are used:
  - The outputs are first put in reset state, that is, inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
  - If the timer clock is still active, then the dead-time generator is activated. The CxIOS and CxCIOS bits are used to program the level after dead-time. Even in this case, the CxIOS and CxCIOS cannot be driven to their active level at the same time. It should be noted that because of synchronization logic on OEN, the dead-time duration is usually longer than usual (around 2 clk\_tmr clock cycles)
  - If FCSODIS=0, the timer releases the enable output, otherwise, it keeps the enable output; the enable output becomes high as soon as one of the CxEN and CxCEN bits becomes high.
- If the brake interrupt or DMA request is enabled, the brake status flag is set, and a brake interrupt or DMA request can be generated.
- If AOEN=1, the OEN bit is automatically set again at the next overflow event.

*Note: When the brake input is active, the OEN cannot be set, nor the status flag, BRKIF can be cleared.*

Figure 14-89 TMR output control

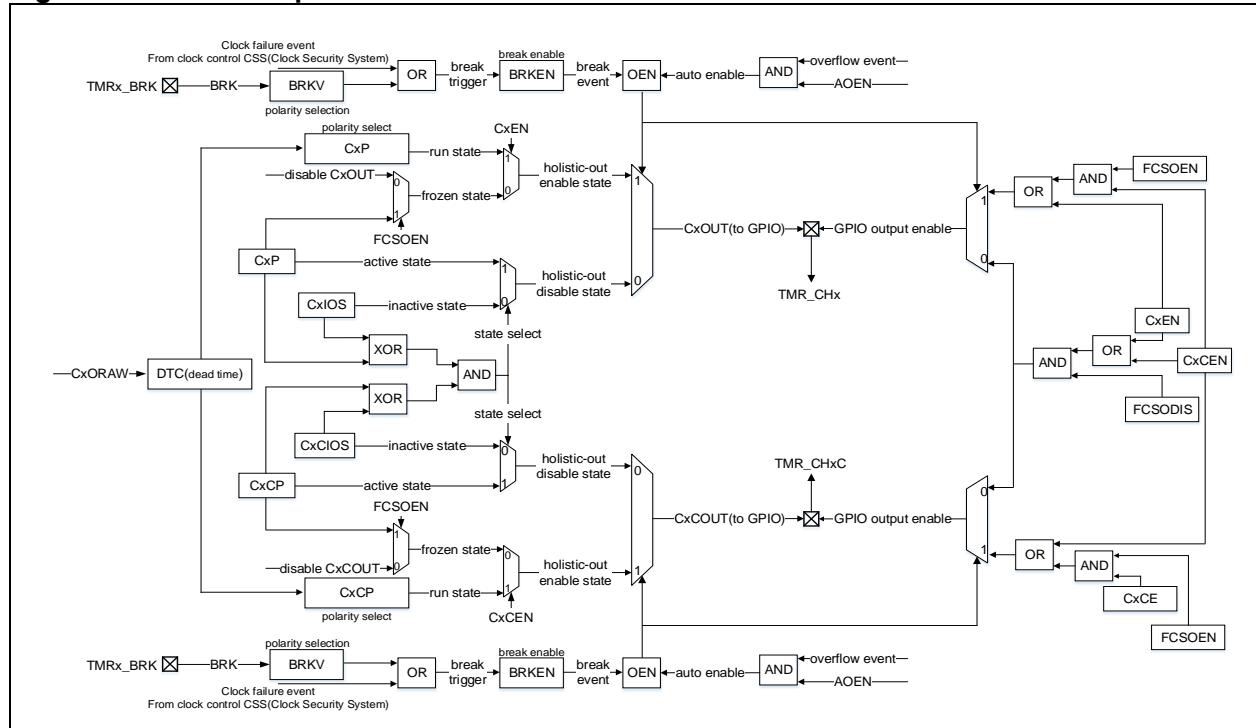
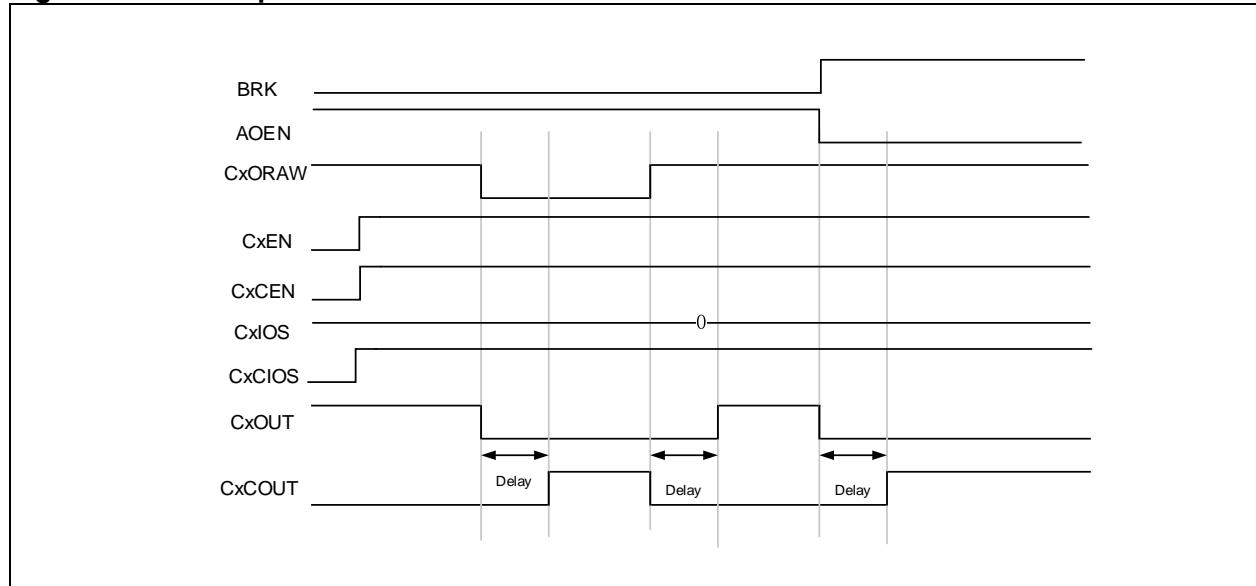


Figure 14-90 Example of TMR brake function



### 14.5.3.6 Debug mode

When the microcontroller enters debug mode (Cortex™-M4 core halted), the TMRx counter stops counting by setting the TMRx\_PAUSE in the DEBUG module.

### 14.5.4 TMR16 and TMR17 registers

These peripheral registers must be accessed by word (32 bits).

TMR16 and TMR17 register are mapped into a 16-bit addressable space.

Table 14-12 TMR16 and TMR17 register map and reset value

Register	Offset	Reset value
TMRx_CTRL1	0x00	0x0000
TMRx_CTRL2	0x04	0x0000

TMRx_IDEN	0x0C	0x0000
TMRxISTS	0x10	0x0000
TMRxSWEVT	0x14	0x0000
TMRxCM1	0x18	0x0000
TMRxCCTRL	0x20	0x0000
TMRxCVAL	0x24	0x0000
TMRxDIV	0x28	0x0000
TMRxPR	0x2C	0x0000
TMRxRPR	0x30	0x0000
TMRxC1DT	0x34	0x0000
TMRxBRK	0x44	0x0000
TMRxDMACTRL	0x48	0x0000
TMRxDMADT	0x4C	0x0000

#### 14.5.4.1 TMR16 and TMR17 control register1 (TMRx\_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x0	resd	Kept at its default value.
Bit 9: 8	CLKDIV	0x0	rw	Clock division 00: Normal, $f_{DTS}=f_{CK\_INT}$ 01: Divided by 2, $f_{DTS}=f_{CK\_INT}/2$ 10: Divided by 4, $f_{DTS}=f_{CK\_INT}/4$ 11: Reserved
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 6: 4	Reserved	0x0	resd	Kept at its default value.
Bit 3	OCMEN	0x0	rw	One cycle mode enable This bit is used to select whether to stop counting at an update event 0: The counter does not stop at an update event 1: The counter stops at an update event
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 1	OVFEN	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	TMR enable 0: Disabled 1: Enabled

#### 14.5.4.2 TMR16 and TMR17 control register2 (TMRx\_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 30: 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	C1CIOS	0x0	rw	Channel 1 complementary idle output state OEN = 0 after dead-time: 0: C1OUTL=0 1: C1OUTL=1
Bit 8	C1IOS	0x0	rw	Channel 1 idle output state OEN = 0 after dead-time: 0: C1OUT=0 1: C1OUT=1
Bit 7: 4	Reserved	0x0	resd	Kept at its default value.

Bit 3	DRS	0x0	rw	DMA request source 0: Capture/compare event 1: Overflow event
Bit 2	CCFS	0x0	rw	Channel control bit flash selection This bit only acts on channels with complementary output. If the channel control bits are buffered: 0: Control bits are updated by setting the HALL bit 1: Control bits are updated by setting the HALL bit or a rising edge on TRGIN.
Bit 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	CBCTRL	0x0	rw	Channel buffer control This bit acts on channels that have complementary output. 0: CxEN, CxCEN and CxOCTRL bits are not buffered. 1: CxEN, CxCEN and CxOCTRL bits are not buffered.

#### 14.5.4.3 TMR16 and TMR17 DMA/interrupt enable register (TMRx\_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	C1DEN	0x0	rw	Channel 1 DMA request enable 0: Disabled 1: Enabled
Bit 8	OVFDEN	0x0	rw	Overflow event DMA request enable 0: Disabled 1: Enabled
Bit 7	BRKIE	0x0	rw	Brake interrupt enable 0: Disabled 1: Enabled
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	HALLIEN	0x0	rw	HALL interrupt enable 0: Disabled 1: Enabled
Bit 4: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled 1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

#### 14.5.4.4 TMR16 and TMR17 interrupt status register (TMRxISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing "0". 0: No capture is detected 1: Capture is detected.
Bit 8	Reserved	0x0	resd	Default value
Bit 7	BRKIF	0x0	rw0c	Brake interrupt flag This bit indicates whether the brake input is active or not. It is set by hardware and cleared by writing "0" 0: Inactive level 1: Active level
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	HALLIF	0x0	rw0c	HALL interrupt flag This bit is set by hardware on HALL event. It is cleared by writing "0". 0: No Hall event occurs. 1: Hall event is detected.

				HALL even: CxEN, CxCEN and CxOCTRL are updated.
Bit 4: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1IF	0x0	rw0c	<p>Channel 1 interrupt flag            If the channel 1 is configured as input mode:            This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT            0: No capture event occurs            1: Capture event is generated</p> <p>If the channel 1 is configured as output mode:            This bit is set by hardware on a compare event. It is cleared by software.            0: No compare event occurs            1: Compare event is generated</p>
Bit 0	OVFIF	0x0	rw0c	<p>Overflow interrupt flag            This bit is set by hardware on an overflow event. It is cleared by software.            0: No overflow event occurs            1: Overflow event is generated. If OVREN=0 and OVFS=0 in the TMRx_CTRL1 register:            - An overflow event is generated when OVFG= 1 in the TMRx_SWEVE register;            - An overflow event is generated when the counter CVAL is reinitialized by a trigger event.</p>

#### 14.5.4.5 TMR16 and TMR17 software event register (TMRx\_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	BRKSWTR	0x0	wo	<p>Brake event triggered by software            This bit is set by software to generate a brake event.            0: No effect            1: Generate a brake event.</p>
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	HALLSWTR	0x0	wo	<p>HALL event triggered by software            This bit is set by software to generate a HALL event.            0: No effect            1: Generate a HALL event.            Note: This bit acts only on channels that have complementary output.</p>
Bit 4: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1SWTR	0x0	wo	<p>Channel 1 event triggered by software            This bit is set by software to generate a channel 1 event.            0: No effect            1: Generate a channel 1 event.</p>
Bit 0	OVFSWTR	0x0	wo	<p>Overflow event triggered by software            This bit is set by software to generate an overflow event.            0: No effect            1: Generate an overflow event.</p>

#### 14.5.4.6 TMR16 and TMR17 channel mode register1 (TMRx\_CM1)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the Cxlx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

##### Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	C1OSEN	0x0	rw	<p>Channel 1 output switch enable            0: C1ORAW is not affected by EXT input.            1: Once a high level is detect on EXT input, clear C1ORAW.</p>
Bit 6: 4	C1OCTRL	0x0	rw	Channel 1 output control

					This field defines the behavior of the original signal C1ORAW. 000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMRx_CVAL=TMRx_C1DT 010: C1ORAW is low when TMRx_CVAL=TMRx_C1DT 011: Switch C1ORAW level when TMRx_CVAL=TMRx_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A — OWCDIR=0, C1ORAW is high once TMRx_C1DT>TMRx_CVAL, else low; — OWCDIR=1, C1ORAW is low once TMRx_C1DT <TMRx_CVAL, else high; 111: PWM mode B — OWCDIR=0, C1ORAW is low once TMRx_C1DT >TMRx_CVAL, else high; — OWCDIR=1, C1ORAW is high once TMRx_C1DT <TMRx_CVAL, else low. <i>Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.</i>
Bit 3	C1OBEN	0x0	rw		Channel 1 output buffer enable 0: Buffer function of TMRx_C1DT is disabled. The new value written to the TMRx_C1DT takes effect immediately. 1: Buffer function of TMRx_C1DT is enabled. The value to be written to the TMRx_C1DT is stored in the buffer register, and can be sent to the TMRx_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw		Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw		Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Reserved 11: Reserved
<b>Input capture mode:</b>					
Bit	Register	Reset value	Type	Description	
Bit 15: 8	Reserved	0x0	resd	Kept at its default value.	
Bit 7: 4	C1DF	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events. 0000: No filter, sampling is done at $f_{DTS}$ 1000: $f_{SAMPLING} = f_{DTS}/8$ , N=6 0001: $f_{SAMPLING} = f_{CK\_INT}$ , N=2 1001: $f_{SAMPLING} = f_{DTS}/8$ , N=8 0010: $f_{SAMPLING} = f_{CK\_INT}$ , N=4 1010: $f_{SAMPLING} = f_{DTS}/16$ , N=5 0011: $f_{SAMPLING} = f_{CK\_INT}$ , N=8 1011: $f_{SAMPLING} = f_{DTS}/16$ , N=6 0100: $f_{SAMPLING} = f_{DTS}/2$ , N=6	

				1100: $f_{SAMPLING}=f_{DTS}/16$ , N=8 0101: $f_{SAMPLING}=f_{DTS}/2$ , N=8 1101: $f_{SAMPLING}=f_{DTS}/32$ , N=5 0110: $f_{SAMPLING}=f_{DTS}/4$ , N=6 1110: $f_{SAMPLING}=f_{DTS}/32$ , N=6 0111: $f_{SAMPLING}=f_{DTS}/4$ , N=8 1111: $f_{SAMPLING}=f_{DTS}/32$ , N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Reserved 11: Reserved

#### 14.5.4.7 TMR16 and TMR17 channel control register (TMRx\_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 4	Reserved	0x0	resd	Kept its default value.
Bit 3	C1CP	0x0	rw	Channel 1 complementary polarity 0: C1COUT is active high. 1: C1COUT is active low.
Bit 2	C1CEN	0x0	rw	Channel 1 complementary enable 0: Output is disabled. 1: Output is enabled.
Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured in output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured in input mode: It is C1CP/C1P bits that define the active edge of an input signal. 00: C1IN rising edge is active. When used as external trigger, C1IN is not inverted. 01: C1IN falling edge is active. When used as external trigger, C1IN is inverted. 10: Reserved. 11: C1IN rising edge and falling edge are both active. When used as external trigger, C1IN is not inverted
Bit 0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

**Table 14-13 Complementary output channel CxOUT and CxCOUT control bits with brake function**

Control bit					Output state <sup>(1)</sup>	
OEN bit	FCSODIS bit	FCSOEN bit	CxEN bit	CxCEN bit	CxOUT output state	CxCOUT output state
1	X	0	0	0	Output disabled (no driven by the timer) CxOUT=0, Cx_EN=0	Output disabled (no driven by the timer) CxCOUT=0, CxCEN=0
		0	0	1	Output disabled (no driven by the timer) CxOUT=0, Cx_EN=0	CxORAW + polarity, CxCOUT= CxORAW xor CxCP, CxCEN=1
		0	1	0	CxORAW+ polarity CxOUT= CxORAW xor CxP, Cx_EN=1	Output disabled (no driven by the timer) CxCOUT=0, CxCEN=0
		0	1	1	CxORAW+polarity+dead-time, Cx_EN=1	CxORAW inverted+polarity+dead-time, CxCEN=1
		1	0	0	Output disabled (no driven by the timer) CxOUT=CxP, Cx_EN=0	Output disabled (no driven by the timer) CxCOUT=CxCP, CxCEN=0
		1	0	1	Off-state (Output enabled with inactive level) CxOUT=CxP, Cx_EN=1	CxORAW + polarity, CxCOUT= CxORAW xor CxCP, CxCEN=1
		1	1	0	CxORAW + polarity, CxOUT= CxORAW xor CxP, Cx_EN=1	Off-state (Output enabled with inactive level) CxCOUT=CxCP, CxCEN=1

			1	1	CxORAW+ polarity+dead-time, Cx_EN=1	CxORAW inverted+polarity+dead-time, CxCEN=1
0	0		0	0	Output disabled (corresponding IO disconnected from timer, and IO floating)	
	0		0	1	Asynchronously: CxOUT=CxP, Cx_EN=0, CxCOUT=CxCP, CxCEN=0;	
	0		1	0	If the clock is present: after a dead-time, CxOUT=CxIOS, CxCOUT=CxCIOS, assuming that CxIOS and CxCIOS do not correspond to CxOUT and CxCOUT active level.	
	0		1	1		
	1	X	0	0	CxEN=CxCEN=0: output disabled (corresponding IO disconnected from timer, and IO floating)	
	1		0	1	In other cases, Off-state (corresponding channel output inactive level)	
	1		1	0	Asynchronously: CxOUT =CxP, Cx_EN=1, CxCOUT=CxCP, CxCEN=1;	
	1		1	1	If the clock is present: after a dead-time, CxOUT=CxIOS, CxCOUT=CxCIOS, assuming that CxIOS and CxCIOS do not correspond to CxOUT and CxCOUT active level.	

Note: If the two outputs of a channel are not used (CxEN = CxCEN = 0), CxIOS, CxCIOS, CxP and CxCP must be cleared.

Note: The state of the external I/O pins connected to the complementary CxOUT and CxCOUT channels depends on the CxOUT and CxCOUT channel state and the GPIO and the IOMUX registers.

#### 14.5.4.8 TMR16 and TMR17 counter value (TMRx\_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0	rw	Counter value

#### 14.5.4.9 TMR16 and TMR17 division value (TMRx\_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0	rw	Divider value The counter clock frequency $f_{CK\_CNT} = f_{TMR\_CLK} / (\text{DIV}[15:0]+1)$ . The value of this register is transferred to the actual prescaler register when an overflow event occurs.

#### 14.5.4.10 TMR16 and TMR17 period register (TMRx\_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

#### 14.5.4.11 TMR16 and TMR17 repetition period register (TMRx\_RPR)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x0	rw	Kept at its default value.
Bit 7: 0	RPR	0x0	rw	Repetition of period value This field is used to reduce the generation rate of overflow events. An overflow event is generated when the repetition counter reaches 0.

#### 14.5.4.12 TMR16 and TMR17 channel 1 data register (TMRx\_C1DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effect immediately

depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

#### 14.5.4.13 TMR16 and TMR17 brake register (TMRx\_BRK)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	OEN	0x0	rw	<p>Output enable This bit is used to enable the output of CxOUT and CxCOUT for those channels that are configured as output.</p> <p>0: Output disabled 1: Output enabled.</p>
Bit 14	AOEN	0x0	rw	<p>Automatic output enable OEN is set automatically at an overflow event.</p> <p>0: Disabled 1: Enabled</p>
Bit 13	BRKV	0x0	rw	<p>Brake input validity This bit is used to select the active level of a brake input.</p> <p>0: Brake input is active low. 1 Brake input is active high.</p>
Bit 12	BRKEN	0x0	rw	<p>Brake enable This bit is used to enable brake input.</p> <p>0: Brake input is disabled. 1: Brake input is enabled.</p>
Bit 11	FCSOEN	0x0	rw	<p>Frozen channel status when holistic output enable This bit acts on the channels that have complementary output. It is used to set the channel state when the timer is inactive and MOEN=1.</p> <p>0: CxOUT/CxCOUT outputs are disabled. 1: CxOUT/CxCOUT outputs are enabled. Output inactive level.</p>
Bit 10	FCSODIS	0x0	rw	<p>Frozen channel status when holistic output disable This bit acts on the channels that have complementary output. It is used to set the channel state when the timer is inactive and MOEN=0.</p> <p>0: CxOUT/CxCOUT outputs are disabled. 1: CxOUT/CxCOUT outputs are enabled. Output idle level.</p>
Bit 9: 8	WPC	0x0	rw	<p>Write protection configuration This field is used to enable write protection.</p> <p>00: Write protection is OFF. 01: Write protection level 3, and the following bits are write protected: TMRx_BRK: DTC, BRKEN, BRKV and AOEN TMRx_CTRL2: CxIOS and CxCIOS 10: Write protection level 2. The following bits and all bits in level 3 are write protected: TMRx_CCTRL: CxP and CxCP TMRx_BRK: FCSODIS and FCSOEN 11: Write protection level 1. The following bits and all bits in level 2 are write protected: TMRx_CMx: C2OCTRL and C2OBEN Note: Once WPC&gt;0, its content remains frozen until the next system reset.</p>
Bit 7: 0	DTC	0x00	rw	<p>Dead-time configuration This field defines the duration of the dead-time insertion. The 3-bit MSB of DTC[7: 0] is used for function selection:</p> <p>0xx: DT = DTC [7: 0] * TDTS 10x: DT = (64+ DTC [5: 0]) * TDTS * 2 110: DT = (32+ DTC [4: 0]) * TDTS * 8 111: DT = (32+ DTC [4: 0]) * TDTS * 16</p>

Note: Based on lock configuration, AOEN, BRKV, BRKEN, FCSODIS, FCSOEN and DTC[7:0] can all be write protected. Thus it is necessary to configure write protection when writing to the TMRx\_BRK register for the first time.

#### 14.5.4.14 TMR16 and TMR17 DMA control register (TMRx\_DMACTRL)

Bit	Register	Reset value	Type	Description
Bit 15:13	Reserved	0x0	resd	Kept at its default value.
Bit 12:8	DTB	0x00	rw	DMA transfer bytes This field defines the number of DMA transfers: 00000: 1 byte      00001: 2 bytes 00010: 3 bytes      00011: 4 bytes ..... 10000: 17 bytes      10001: 18 bytes
Bit 7:5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	ADDR	0x00	rw	DMA transfer address offset ADDR is defined as an offset starting from the address of the TMRx_CTRL1 register: 00000: TMRx_CTRL1 00001: TMRx_CTRL2 00010: TMRx_STCTRL .....

#### 14.5.4.15 TMR16 and TMR17 DMA data register (TMRx\_DMADT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DMADT	0x0000	rw	DMA data register A write/read operation to the DMADT register accesses any TMR register located at the following address: TMRx peripheral address + ADDR*4 to TMRx peripheral address + ADDR*4 + DTB*4

## 14.6 Advanced-control timers (TMR1)

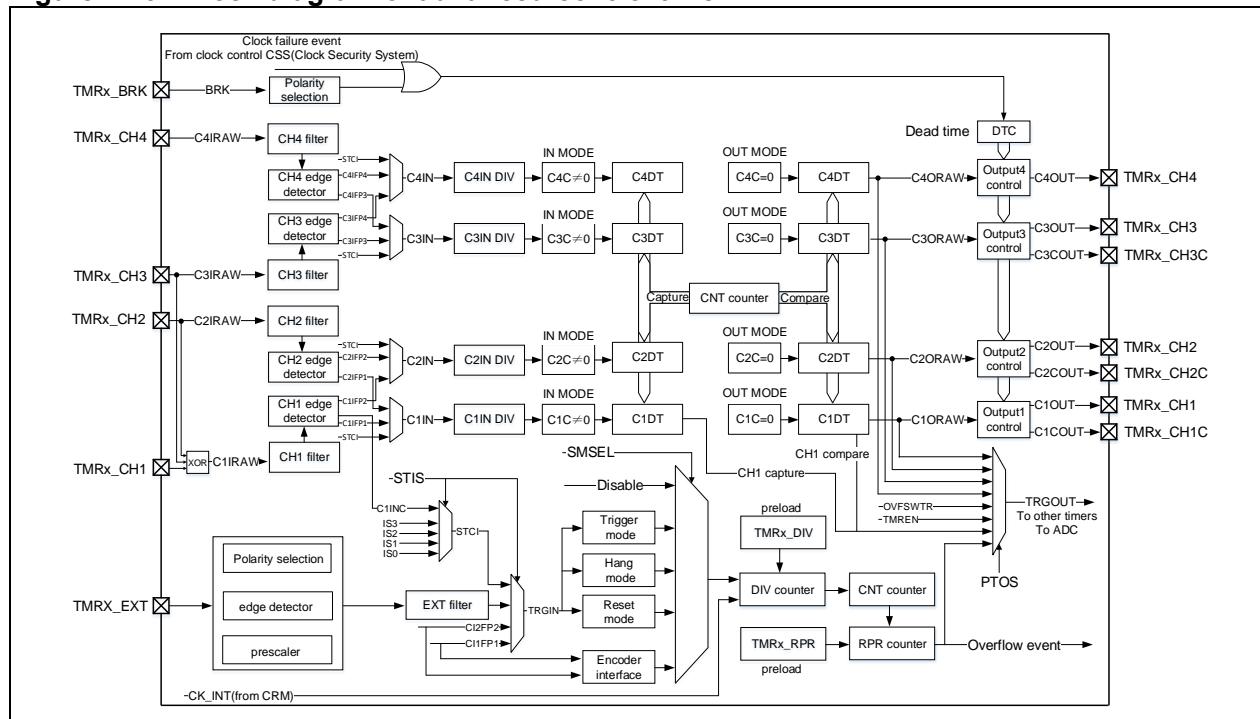
### 14.6.1 TMR1 introduction

The advanced-control timer TMR1 consists of a 16-bit counter supporting up, down or up/down counting modes, four capture/compare registers, and four independent channels to achieve embedded dead-time, input capture and programmable PWM output.

### 14.6.2 TMR1 main features

- Source of counter clock: internal clock, external clock an internal trigger input
- 16-bit up, down, up/down, repetition and encoder mode counter
- Four independent channels for input capture, output compare, PWM generation, one-pulse mode output and embedded dead-time
- Three independent channels for complementary output
- TMR brake function
- Synchronization control between master and slave timers
- Interrupt/DMA is generated at overflow event, trigger event, brake signal input and channel event
- Support TMR burst DMA transfer

**Figure 14-91 Block diagram of advanced-control timer**

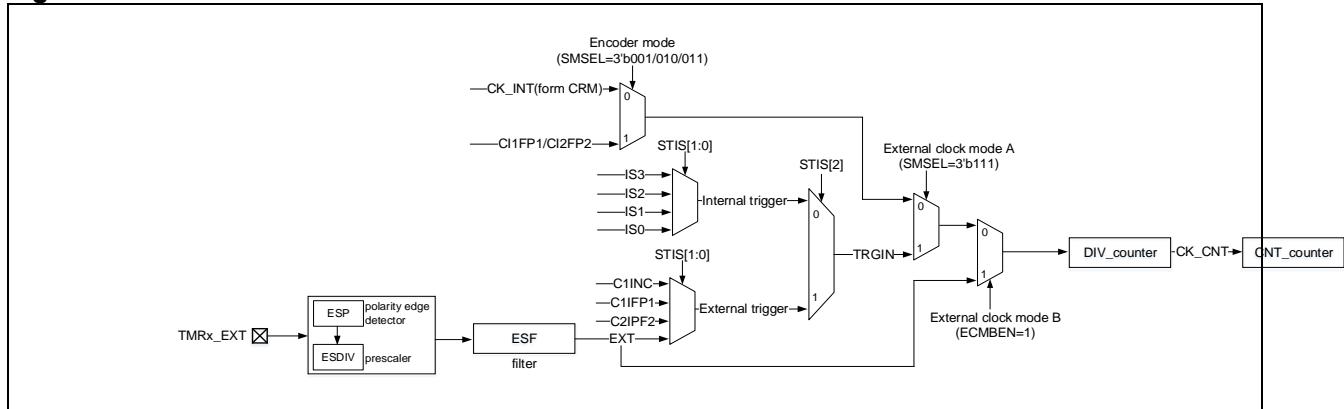


### 14.6.3 TMR1 functional overview

#### 14.6.3.1 Count clock

The counter of TMR1 can be clocked by the internal clock (CK\_INT), external clock (external clock mode A and B) and internal trigger input (ISx).

Figure 14-92 Count clock

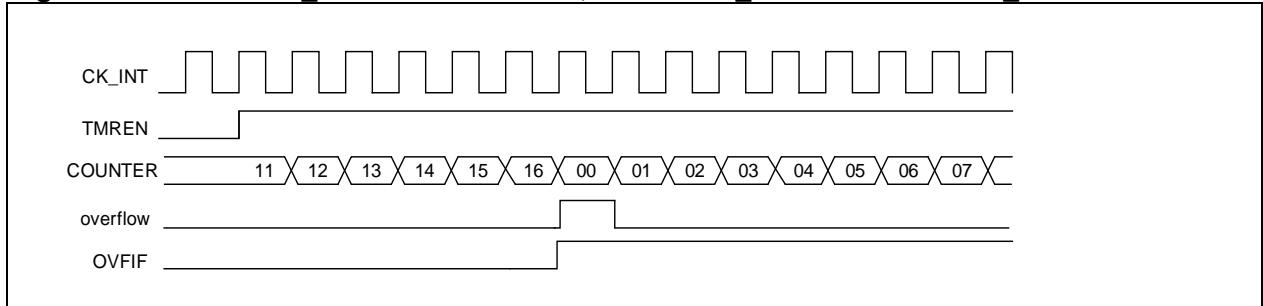


### Internal clock (CK\_INT)

By default, the CK\_INT divided by the prescaler is used to drive the counter to start counting. The configuration process is as follows:

- Set the CLKDIV[1:0] bit in the TMRx\_CTRL1 register to set the CK\_INT frequency;
- Set the TWCMSEL[1:0] bit in the TMRx\_CTRL1 register to select count mode. If the one-way count direction is set, configure OWCDIR bit in the TMRx\_CTRL1 register to select the specific direction.
- Set the TMRx\_DIV register to set the counting frequency;
- Set the TMRx\_PR register to set the counting period;
- Set the TMREN bit in the TMRx\_CTRL1 register to enable the counter.

Figure 14-93 Use CK\_INT to drive counter, with TMRx\_DIV=0x0 and TMRx\_PR=0x16



### External clock (TRGIN/EXT)

The counter clock can be provided by two external clock sources, namely, TRGIN and EXT signals.

When SMSEL=3'b111, external clock mode A is selected. Set the STIS[2:0] bit to select TRGIN signal to drive the counter to start counting. The external clock sources include: C1INC (STIS=3'b100, channel 1 rising edge and falling edge), C1IFP1 (STIS=3'b101, channel 1 signal with filtering and polarity selection), C2IFP2 (STIS=3'b110, channel 2 signal with filtering and polarity selection) and EXT (STIS=3'b111, external input signal after polarity selection, frequency division and filtering).

When ECMBEN=1, external clock mode B is selected. The counter is driven by the external input signal EXT that has gone through polarity selection, frequency division and filtering. External clock mode B is equivalent to external clock mode A with EXT signal as the TRGIN.

To use external clock mode A, follow the configuration steps as below:

- Configure the external clock source TRGIN.

When TMRx\_CH1 is selected as the TRGIN, configure the channel 1 input filter (by setting the C1DF[3:0] bit in the TMRx\_CM1 register) and channel 1 input polarity (by setting the C1P/C1CP in the TMRx\_CCTRL register).

When TMRx\_CH2 is selected as the TRGIN, configure the channel 2 input filter (by setting the C2DF[3:0] bit in the TMRx\_CM1 register) and channel 1 input polarity (by setting the C2P/C2CP in the TMRx\_CCTRL register).

When TMRx\_EXT is selected as the TRGIN, configure the external signal polarity (by setting the ESP bit in the TMRx\_STCTRL register), external signal division (by setting the ESDIV[1:0] bit in the TMRx\_STCTRL register) and external signal filter (by setting the ESF[3:0] bit in the TMRx\_STCTRL register).

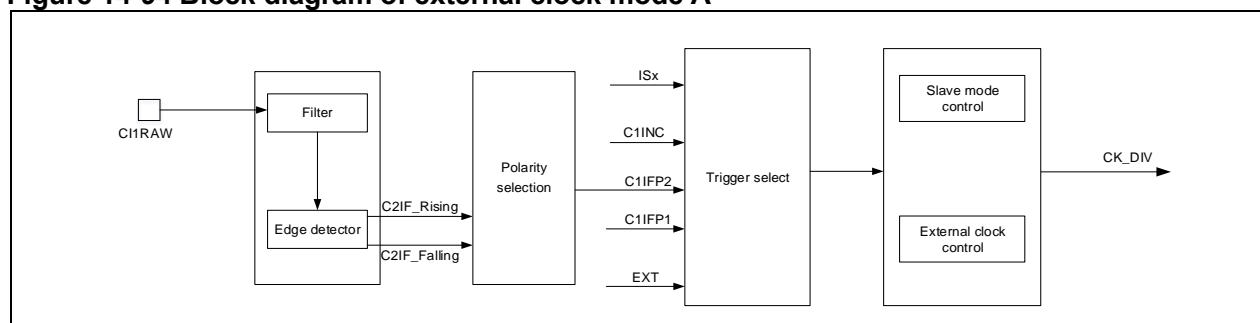
- Set the TRGIN signal source by setting the STIS[1:0] bit in the TMRx\_STCTRL register.
- Enable external clock mode A by setting SMSEL=3'b111 in the TMRx\_STCTRL register.
- Set counter counting frequency by setting the DIV[15:0] bit in the TMRx\_DIV register.
- Set counter counting period by setting the PR[15:0] bit in the TMRx\_PR register.
- Enable counter by setting the TMREN bit in the TMRx\_CTRL1 register.

To use external clock mode B, follow the configuration steps as below:

- Set external signal polarity by setting the ESP bit in the TMRx\_STCTRL register.
- Set external signal frequency division by setting the ESDIV[1:0] bit in the TMRx\_STCTRL register.
- Set external signal filter by setting the ESF[3:0] bit in the TMRx\_STCTRL register.
- Enable external clock mode B by setting the ECMBEN bit in the TMRx\_STCTRL register.
- Set counter counting frequency by setting the DIV[15:0] bit in the TMRx\_DIV register.
- Set counter counting period by setting the PR[15:0] bit in the TMRx\_PR register.

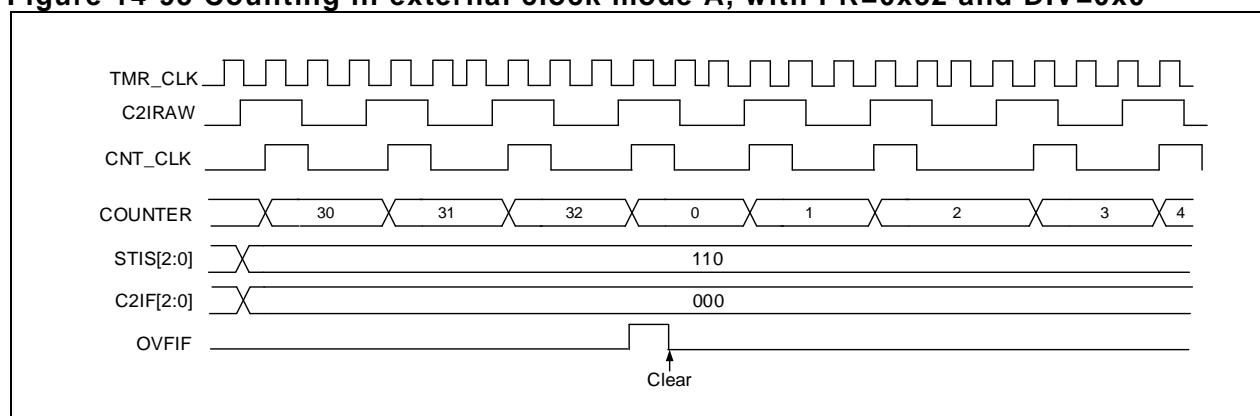
Enable counter by setting the TMREN bit in the TMRx\_CTRL1 register.

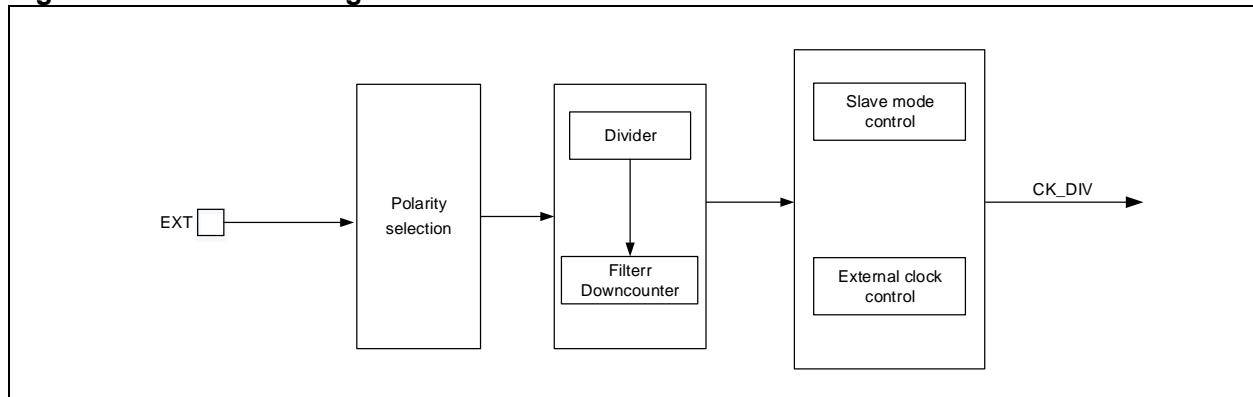
**Figure 14-94 Block diagram of external clock mode A**



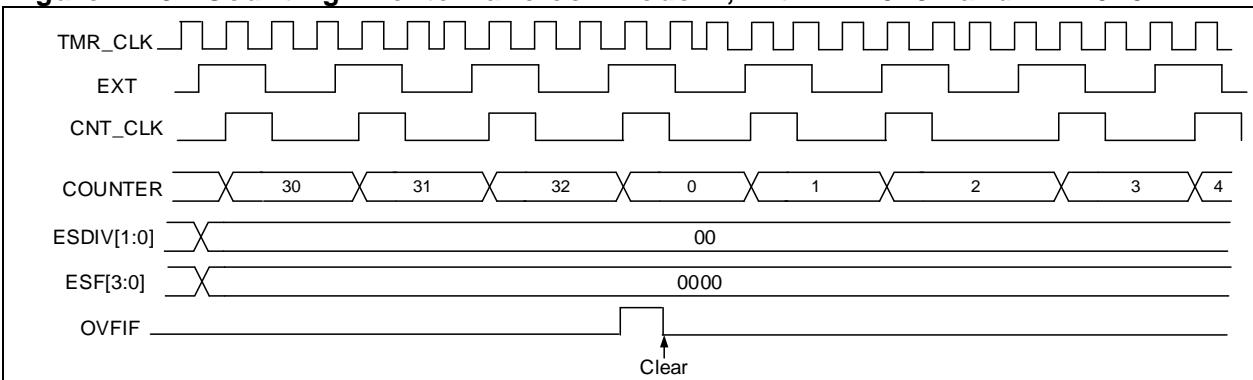
*Note: The delay between the signal on the input side and the actual clock of the counter is due to the synchronization circuit.*

**Figure 14-95 Counting in external clock mode A, with PR=0x32 and DIV=0x0**



**Figure 14-96 Block diagram of external clock mode B**

*Note: The delay is present between the signal on the input side and the actual clock of the counter due to the synchronization circuit.*

**Figure 14-97 Counting in external clock mode B, with PR=0x32 and DIV=0x0**

#### Internal trigger input (ISx)

Timer synchronization allows interconnection between several timers. The TMR\_CLK of one timer can be provided by the TRGOOUT signal output by another timer. Set the STIS[2:0] bit to select internal trigger signal to enable counting.

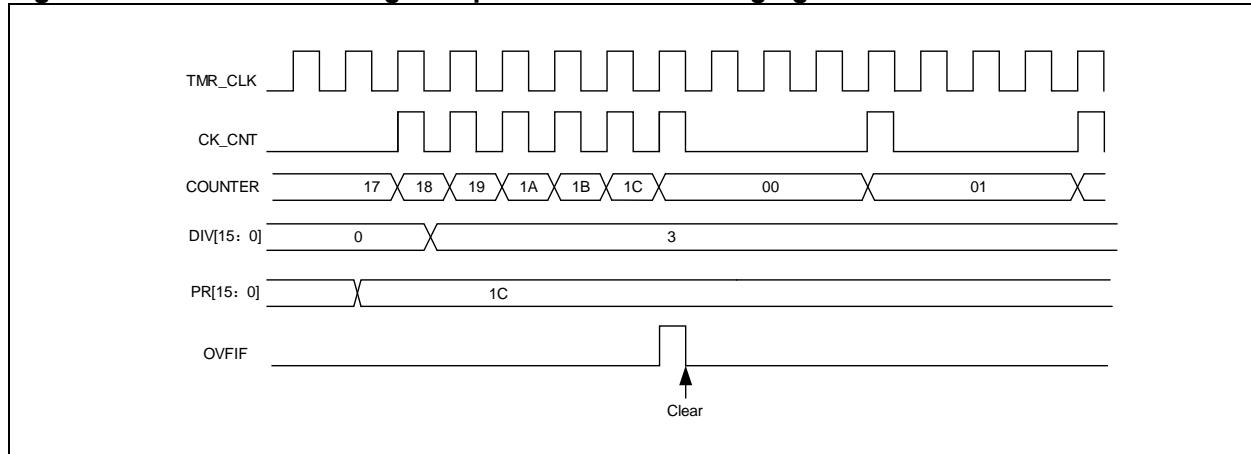
Each timer consists of a 16-bit prescaler, which is used to generate the CK\_CNT that enables the counter to count. The frequency division relationship between the CK\_CNT and TMR\_CLK can be adjusted by setting the value of the TMRx\_DIV register. The prescaler value can be modified at any time, but it takes effect only when the next overflow event occurs.

The internal trigger input is configured as follows:

- Set the TMRx\_PR register to set counting period;
- Set the TMRx\_DIV register to set counting frequency;
- Set the TWCMSEL[1:0] bit in the TMRx\_CTRL1 register to set count mode;
- Set the STIS[2:0] bit (range: 3'b000~3'b011) in the TMRx\_STCTRL register and select internal trigger;
- Set SMSEL[2:0]=3'b111 in the TMRx\_STCTRL register and select external clock mode A;
- Set the TMREN bit in the TMRx\_CTRL1 register to enable TMRx counter.

**Table 14-14 TMR1 internal trigger connection**

Slave timer	IS0 (STIS=000)	IS1 (STIS=001)	IS2 (STIS=010)	IS3 (STIS=011)
TMR1	TMR15	-	TMR3	-

**Figure 14-98 Counter timing with prescaler value changing from 1 to 4**

### 14.6.3.2 Counting mode

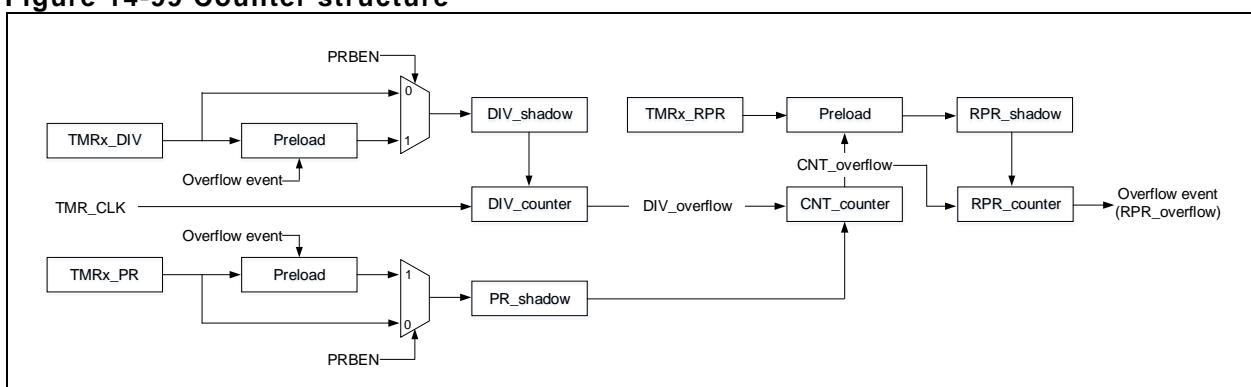
The advanced-control timer consists of a 16-bit counter supporting up, down, up/down counting modes. The TMRx\_PR register is used to configure the counting period. The value in the TMRx\_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx\_PR register is transferred to the shadow register only at an overflow event.

The TMRx\_DIV register is used to configure the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to the TMRx\_PR register, when the periodic buffer is enabled, the value in the TMRx\_DIV register is updated to the shadow register at an overflow event.

Reading the TMRx\_CNT register returns to the current counter value, and writing to the TMRx\_CNT register updates the current counter value to the value being written.

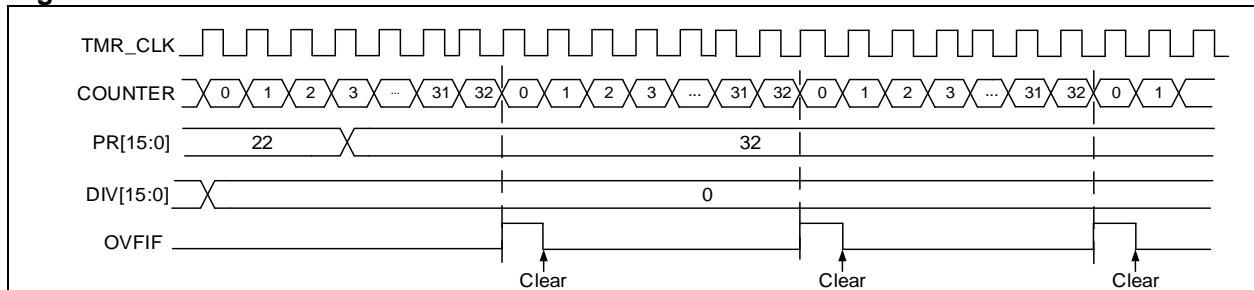
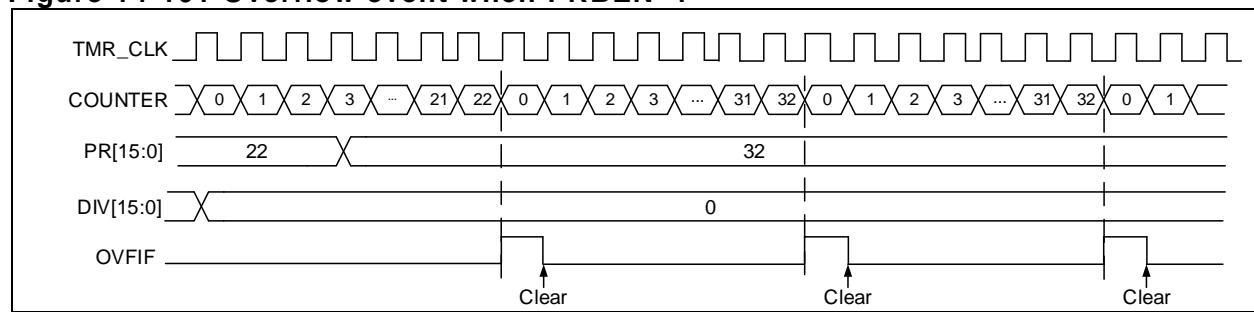
An overflow event is generated by default. Set OVFEN=1 in the TMRx\_CTRL1 to disable generation of update events. The OVFS bit in the TMRx\_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only counter overflow/underflow triggers an overflow event.

Setting the TMREN bit (TMREN=1) enables the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR\_EN is set 1 clock cycle after the TMREN is set.

**Figure 14-99 Counter structure**

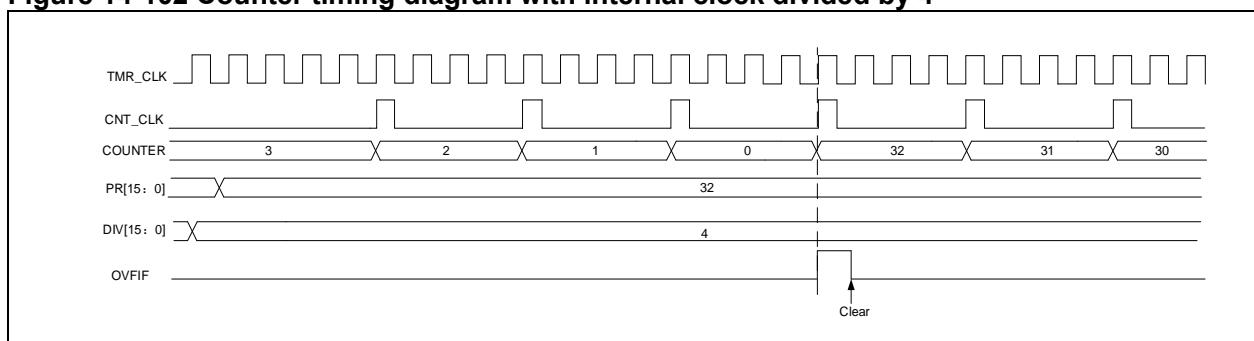
#### Upcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx\_CTRL1 register to enable the upcounting mode. In this mode, the counter counts from 0 to the value programmed in the TMR1\_PR register, restarts from 0, and generates a counter overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the register is no longer reloaded with the preload and re-loaded value after counter overflow occurs, otherwise, the prescaler and re-loaded value will be updated at an overflow event.

**Figure 14-100 Overflow event when PRBEN=0****Figure 14-101 Overflow event when PRBEN=1**

### Downcounting mode

Set **CMSEL[1:0]=2'b00** and **OWCDIR=1'b1** in the **TMRx\_CTRL1** register to enable the downcounting mode. In this mode, the counter counts from the value programmed in the **TMR1\_PR** register down to 0, and restarts from the value programmed in the **TMR1\_PR** register, and generates a counter underflow event.

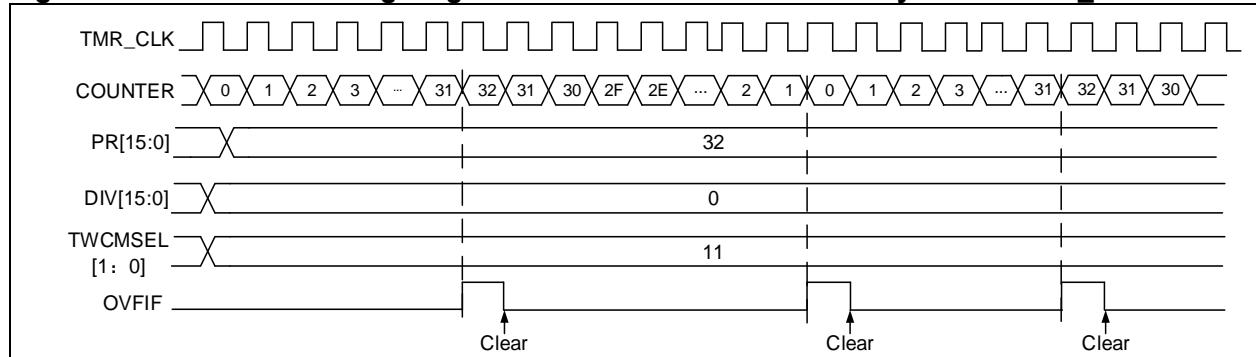
**Figure 14-102 Counter timing diagram with internal clock divided by 4**

### Up/down counting mode

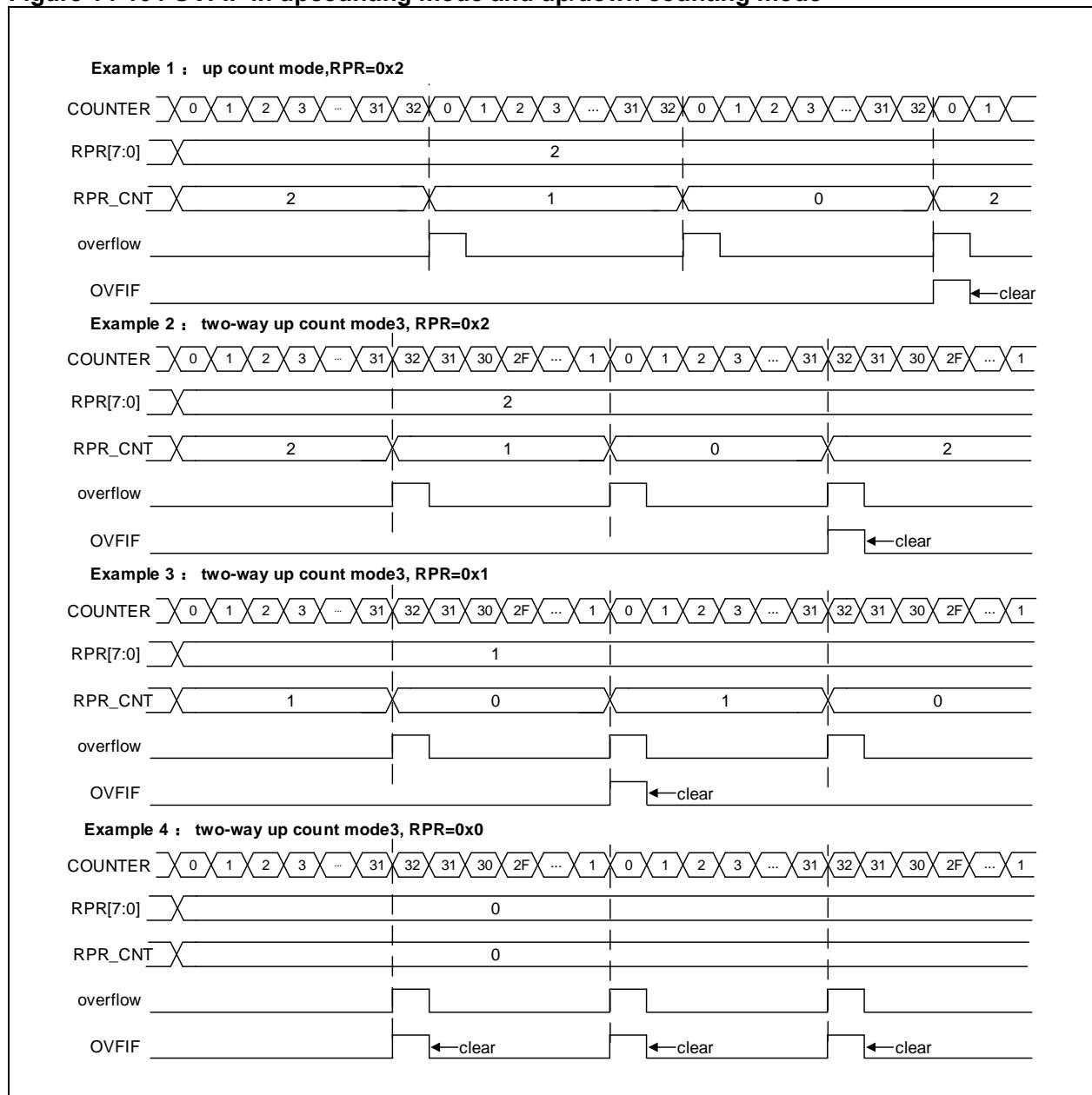
Set **CMSEL[1:0]≠2'b00** in the **TMRx\_CTRL1** register to enable the up/down counting mode. In this mode, the counter counts up/down alternatively. When the counter counts from the value programmed in the **TMR1\_PR** register down to 1, an underflow event is generated, and then restarts counting from 0; when the counter counts from 0 to the value of the **TMR1\_PR** register -1, an overflow event is generated, and then restarts downcounting from the value of the **TMR1\_PR** register. The **OWCDIR** bit indicates the current counting direction.

The **TWCMSEL[1:0]** bit in the **TMRx\_CTRL1** register is also used to select the **CxIF** flag setting method in up/down counting mode. In up/down counting mode 1 (**TWCMSEL[1:0]=2'b01**), **CxIF** flag can only be set when the counter counts down; in up/down counting mode 2 (**TWCMSEL[1:0]=2'b10**), **CxIF** flag can only be set when the counter counts up; in up/down counting mode 3 (**TWCMSEL[1:0]=2'b11**), **CxIF** flag can be set when the counter counts up/down.

*Note: The **OWCDIR** is ready-only in up/down counting mode.*

**Figure 14-103 Counter timing diagram with internal clock divided by 1 and TMRx\_PR=0x32****Repetition counter mode:**

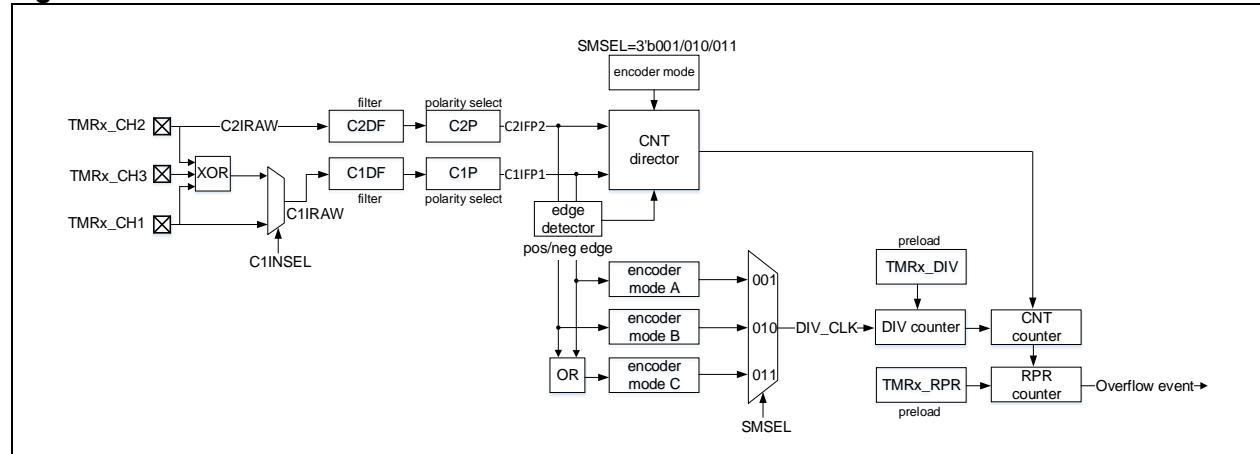
The TMRx\_RPR register is used to configure the counting period of repetition counter. The repletion counter mode is enabled when the repetition counter value is not equal to 0. In this mode, an overflow event occurs once at every counter overflow (RPR[7:0]+1), and the repetition counter is decremented at each counter overflow. An overflow event is generated only when the repetition counter reaches 0. The frequency of the overflow event can be adjusted by setting the repetition counter value.

**Figure 14-104 OVFIF in upcounting mode and up/down counting mode**

### Encoder interface mode

To enable the encoder interface mode, write SMSEL[2: 0]= 3'b001/3'b010/3'b011. In this mode, the two inputs (C1IN/C2IN) are required. Depending on the level on one input, the counter counts up or down on the edge of the other input. The OWCDIR bit indicates the direction of the counter.

**Figure 14-105 Encoder mode structure**



Encoder mode A: SMSEL=3'b001. The counter counts on C1IFP1 (rising edge and falling edge), and the counting direction is dependent on the edge direction of C1IFP1 and the level of C2IFP2.

Encoder mode B: SMSEL=3'b010, the counter counts on C2IFP2 (rising edge and falling edge), and the counting direction is dependent on the edge direction of C2IFP2 and the level of C1IFP1.

Encoder mode C: SMSEL=3'b011, the counter counts on C1IFP1 and C2IFP2 (rising edge and falling edge), and the counting direction is dependent on the C1IFP1 edge direction + C2IFP2 level, and C2IFP2 edge direction + C1IFP1 level.

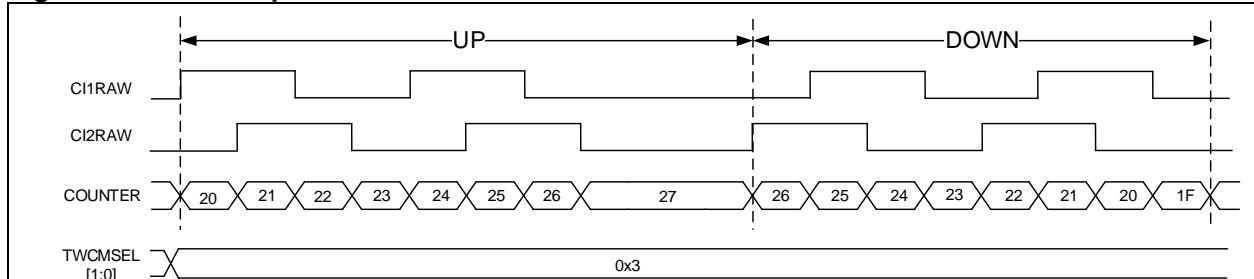
To use the encoder mode, follow the configuration steps as below:

- Set the C1DF[3:0] bit in the TMRx\_CM1 register to set channel 1 input signal filtering; set the C1P bit in the TMRx\_CCTRL register to set channel 1 input signal active level.
- Set the C2DF[3:0] bit in the TMRx\_CM1 register to set channel 2 input signal filtering; set the C2P bit in the TMRx\_CCTRL register to set channel 2 input signal active signal.
- Set the C1C[1:0] bit in the TMRx\_CM1 register to set channel 1 as input mode; set the C2C[1:0] bit in the TMRx\_CM1 register to set channel 2 as input mode.
- Set the SMSEL[2:0] bit in the TMRx\_STCTRL register to select encoder mode A (SMSEL=3'b001), encoder mode B (SMSEL=3'b010) or encoder mode C (SMSEL=3'b011).
- Set the PR[15:0] bit in the TMRx\_PR register to set the counting period.
- Set the DIV[15:0] bit in the TMRx\_DIV register to set the counting frequency.
- Set the IOs corresponding to TMRx\_CH1 and TMRx\_CH2 as multiplexed mode.
- Set the TMREN bit in the TMRx\_CTRL1 register to enable the counter.

**Table 14-15 Counting direction versus encoder signals**

Active edge	Level on opposite signal (C1INFP1 to C2IN, C2INFP2 to C1IN)	C1INFP1 signal		C2INFP2 signal	
		Rising	Falling	Rising	Falling
Count on C1IN only	High	Down	Up	No count	No count
	Low	Up	Down	No count	No count
Count on C2IN only	High	No count	No count	Up	Down
	Low	No count	No count	Down	Up

Count on both C1IN and C2IN	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

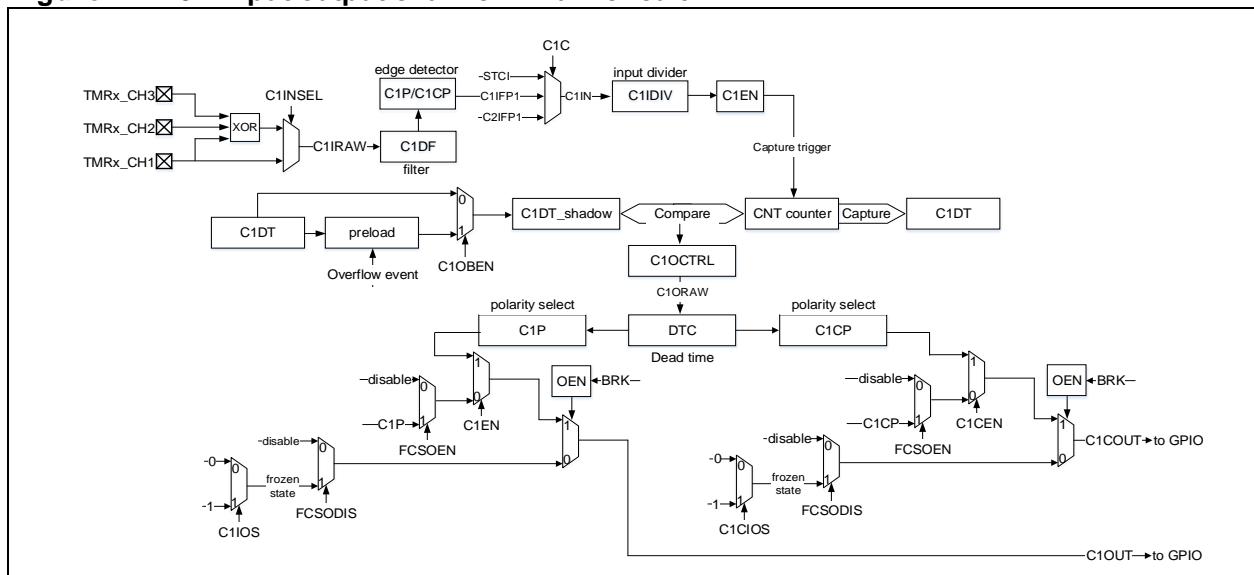
**Figure 14-106 Example of encoder interface mode C**

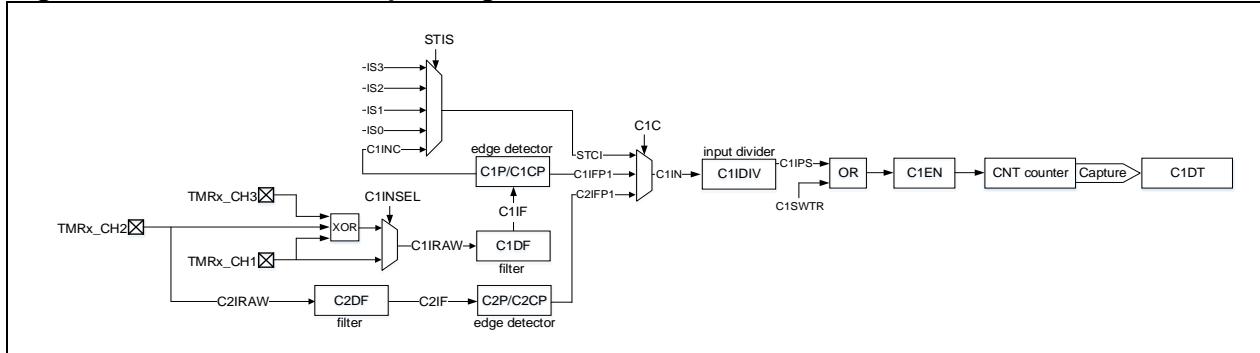
### 14.6.3.3 TMR input function

The TMR1 has four independent channels. Each channel can be configured as input or output.

As input, each channel input signal is processed as below:

- TMRx\_CHx outputs the pre-processed CxIRAW. Set the C1INSEL bit to select the source of C1IRAW from TMRx\_CH1 or the XOR-ed TMRx\_CH1, TMRx\_CH2 and TMRx\_CH3, and the sources of C2IRAW, C3IRAW and C4IRAW are TMRx\_CH2, TMRx\_CH3 and TMRx\_CH4, respectively.
- CxIRAW inputs digital filter and outputs a filtered signal CxIF. Set the sampling frequency and sampling times of digital filter by setting the CxDf bit.
- CxIF inputs edge detector and outputs the signal CxIFPx after edge selection. The edge selection is controlled by CxP and CxCp bits, and can be selected as rising edge, falling edge or both edges active.
- CxIFPx inputs capture signal selector and then outputs the signal CxIN after selection. The capture signal selector is controlled by the CxC bits. The source of CxIN can be set as CxIFPx, CyIFPx or STCI. The CyIFPx ( $x \neq y$ ) is the CyIFPy from channel y and handled by channel x edge detector (for example, the C1IFP2 is the C1IFP1 from channel 1 and then handled by channel 2 edge detector), and STCI derives from the slave timer controller, and its source is selected by setting the STIS bit.
- CxIN outputs the signal CxIPS that is divided by the input channel divider. The division factor is set to "No division", "divided by 2", "divided by 4" or "divided by 8" by setting the CxIDIV bit.

**Figure 14-107 Input/output channel 1 main circuit**

**Figure 14-108 Channel 1 input stage**

### **Input mode**

In input mode, the TMR1\_CxDT registers latch the current counter values after the selected trigger signal is detected, and the capture compare interrupt flag bit (CxIF) is set. An interrupt/DMA request will be generated if the CxIEN bit and CxDEN bit are enabled. If the selected trigger signal is detected when the CxIF is set, a capture overflow event occurs. The TMRx\_CxDT register overwrites the recorded value with the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, following the configuration procedure mentioned below:

- Set C1C=01 in the TMR1\_CM1 register to select the C1IN as channel 1 input
- Set the filter bandwidth of C1IN signal (CxDF[3: 0])
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMR1\_CCTR register
- Program the capture frequency division of C1IN signal (C1DIV[1: 0])
- Enable channel 1 input capture (C1EN=1)
- If needed, enable the relevant interrupt or DMA request by setting the C1IEN bit in the TMR1\_IDEN register or the C1DEN bit in the TMR1\_IDEN register

### **Timer Input XOR function**

The timer input pins (TMR1\_CH1, TMR1\_CH2 and TMR1\_CH3) are connected to the channel 1 (selected by setting the C1INSE in the TMRx\_CTRL2 register) through an XOR gate.

The XOR gate can be used to connect Hall sensors. For example, connect the three XOR inputs to the three Hall sensors respectively so as to calculate the position and speed of the rotation by analyzing three Hall sensor signals.

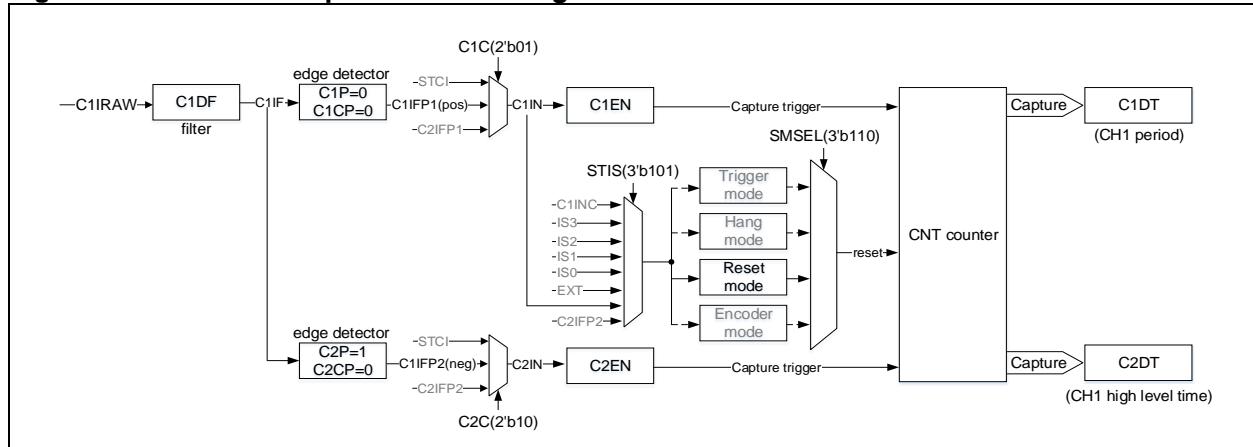
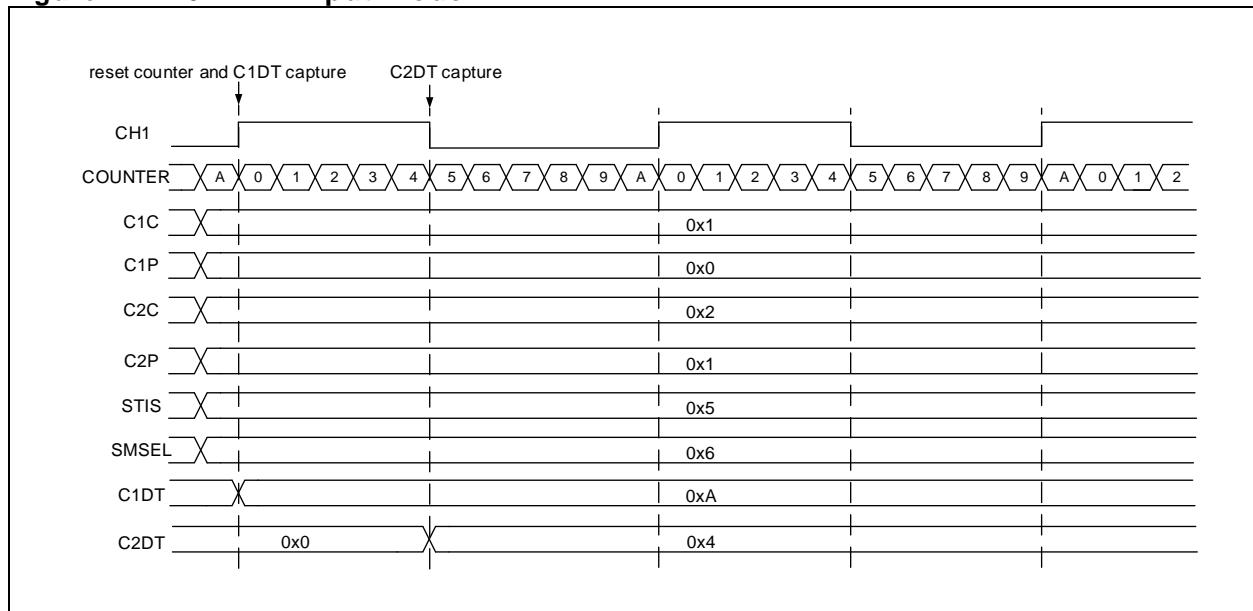
### **PWM input**

The PWM input mode applies to channel 1 and channel 2. To enable this mode, map the C1IN and C2IN to the same TMRx\_CHx, and configure the CxIFPx of channel 1/2 to trigger slave timer controller reset.

The PWM input mode can be used to measure the period and duty cycle of input signal. The period and duty cycle of channel 1 can be measured as follows:

- Set C1C=2'b01 to set C1IN as C1IFP1;
- Set C1P=1'b0 to set C1IFP1 rising edge active
- Set C2C=2'b10 to set C2IN as C1IFP2;
- Set C2P=1'b1 to set C1IFP2 falling edge active;
- Set STIS=3'b101, and set C1IFP1 as the slave timer trigger signal;
- Set SMSEL=3'b100 to set the slave timer in reset mode;
- Set C1EN=1'b1 and C2EN=1'b1 to enable channel 1 and input capture.

In these configurations, the rising edge of channel 1 input signal triggers capture and saves captured values to the C1DT register, and channel 1 input signal rising edge resets the counter. The falling edge of channel 1 input signal triggers capture and saves captured values to the C2DT register. The period and duty of channel 1 input signal can be calculated through C1DT and C2DT respectively.

**Figure 14-109 PWM input mode configuration****Figure 14-110 PWM input mode**

#### 14.6.3.4 TMR output function

The TMR output consists of a comparator and an output controller. It is used to program the period, duty cycle and polarity of the output signal.

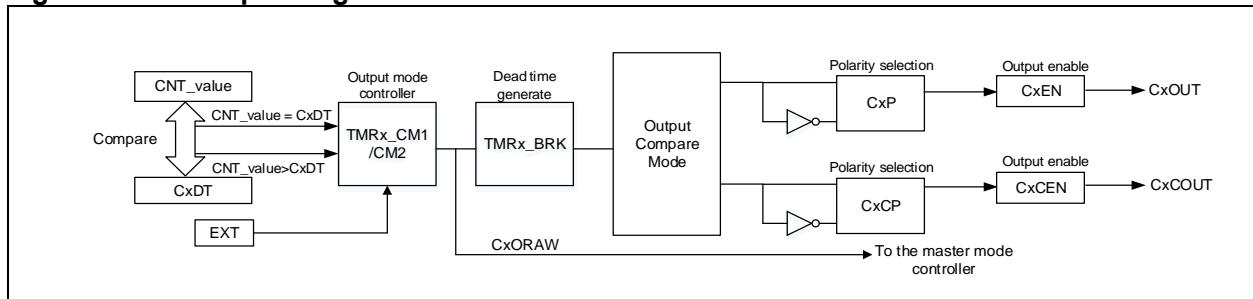
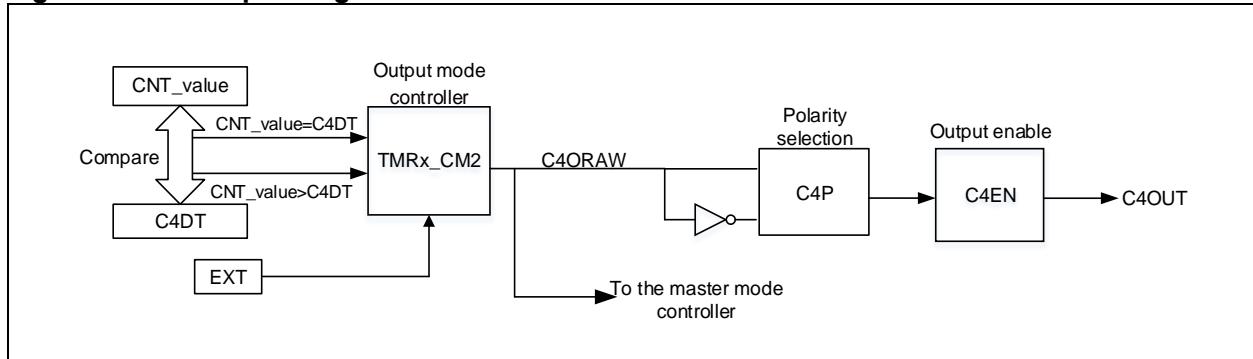
**Figure 14-111 Output stage for channel 1 to 3**

Figure 14-112 Output stage for channel 4



### Output mode

Write  $CxC[2:0] \neq 2'b00$  to configure the channel as output to implement multiple output modes. In this case, the counter value is compared with the value in the  $TMR1_{CxDT}$  register, and the intermediate signal  $CxORAW$  is generated according to the output mode selected by  $CxOCTRL[2:0]$ , which is sent to IO after being processed by the output control circuit. The period of the output signal is configured by the  $TMR1_PR$  register, while the duty cycle by the  $TMR1_{CxDT}$  register.

Output compare modes include:

- **PWM mode A:** Set  $CxOCTRL=3'b110$  to enable PWM mode A. In upcounting, when  $TMRx_{C1DT} > TMRx_{CVAL}$ ,  $C1ORAW$  outputs high; otherwise, outputs low. In downcounting, when  $TMRx_{C1DT} < TMRx_{CVAL}$ ,  $C1ORAW$  outputs low; otherwise, outputs high. To set PWM mode A, the following process is recommended:
  - Set the  $TMRx_PR$  register to set PWM period;
  - Set the  $TMRx_{CxDT}$  register to set PWM duty cycle;
  - Set  $CxOCTRL=3'b110$  in the  $TMRx_{CM1/CM2}$  register to set output mode as PWM mode A;
  - Set the  $TMRx_DIV$  register to set the counting frequency;
  - Set the  $TWCMSEL[1:0]$  bit in the  $TMRx_CTRL1$  register to set the count mode;
  - Set  $CxP$  bit and  $CxCP$  bit in the  $TMRx_CCTRL$  register to set output polarity;
  - Set  $CxEN$  bit and  $CxCEN$  bit in the  $TMRx_CCTRL$  register to enable channel output;
  - Set the  $OEN$  bit in the  $TMRx_BRK$  register to enable  $TMRx$  output;
  - Set the corresponding GPIO of  $TMR$  output channel as the multiplexed mode;
  - Set the  $TMREN$  bit in the  $TMRx_CTRL1$  register to enable  $TMRx$  counter.
- **PWM mode B:** Set  $CxOCTRL=3'b111$  to enable PWM mode B. In upcounting, when  $TMRx_{C1DT} > TMRx_{CVAL}$ ,  $C1ORAW$  outputs low; otherwise, outputs high. In downcounting, when  $TMRx_{C1DT} < TMRx_{CVAL}$ ,  $C1ORAW$  outputs high; otherwise, outputs low.
- **Forced output mode:** Set  $CxOCTRL=3'b100/101$  to enable forced output mode. In this case, the  $CxORAW$  is forced to be the programmed level, irrespective of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.
- **Output compare mode:** Set  $CxOCTRL=3'b001/010/011$  to enable output compare mode. In this case, when the counter value matches the value of the  $CxDT$  register, the  $CxORAW$  is forced high ( $CxOCTRL=3'b001$ ), low ( $CxOCTRL=3'b010$ ) or toggling ( $CxOCTRL=3'b011$ ).
- **One-pulse mode:** This is a particular case of PWM mode. Set  $OCMEN=1$  to enable one-pulse mode. In this mode, the comparison match is performed in the current counting period. The  $TMREN$  bit is cleared as soon as the current counting is completed. Therefore, only one pulse is output. When configured as in upcounting mode, the configuration must follow the rule:  $CVAL < CxDT \leq PR$ ; in downcounting mode,  $CVAL > CxDT$  is required.
- **Fast output mode:** Set  $CxOEN=1$  to enable this mode. If enabled, the  $CxORAW$  signal will not change when the counter value matches the  $CxDT$ , but at the beginning of the current counting period. In other words, the comparison result is advanced, so the comparison result between the counter value and the  $TMRx_{CxDT}$  register will determine the level of  $CxORAW$  in advance.

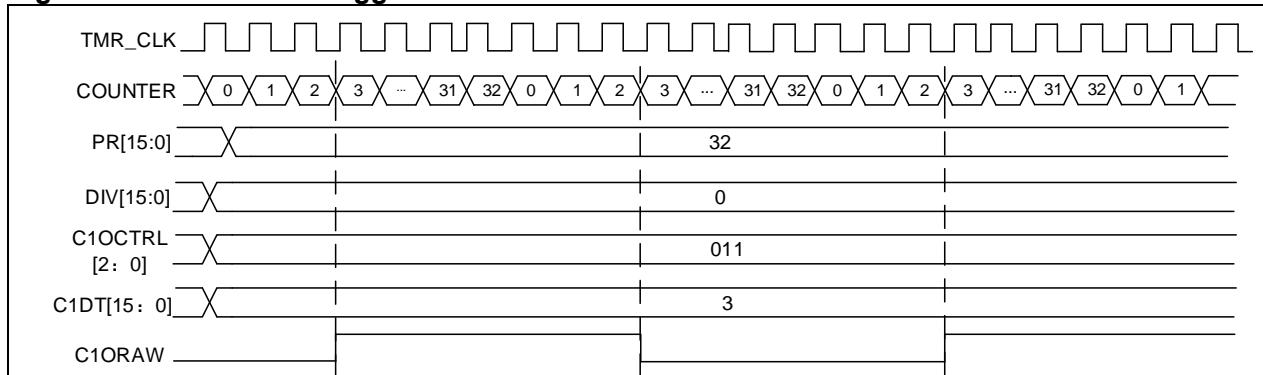
**Figure 14-113** gives an example of output compare mode (toggle) with  $C1DT=0x3$ . When the counter value is equal to  $0x3$ ,  $C1OUT$  toggles.

**Figure 14-114** gives an example of the combination between upcounting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

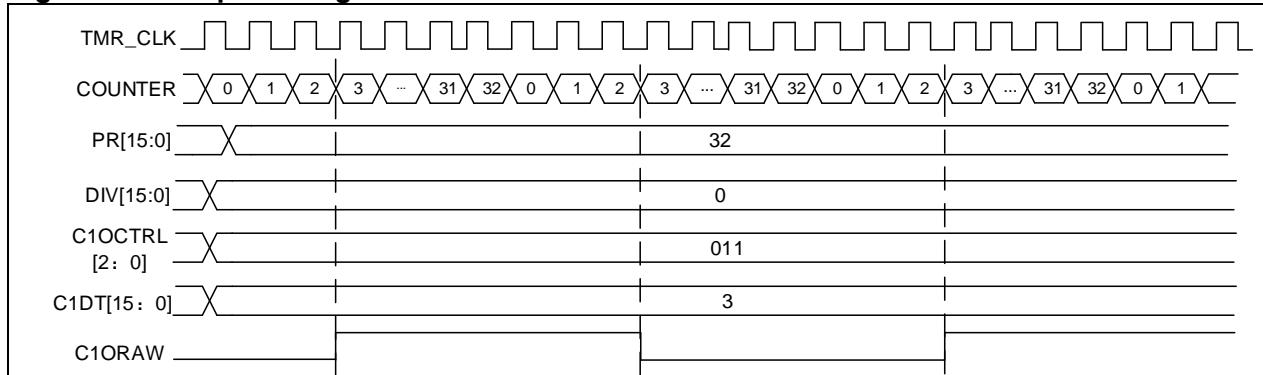
**Figure 14-115** gives an example of the combination between up/down counting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

**Figure 14-116** gives an example of the combination between upcounting mode and one-pulse PWM mode B. The counter only counts only one cycle, and the output signal sends only one pulse.

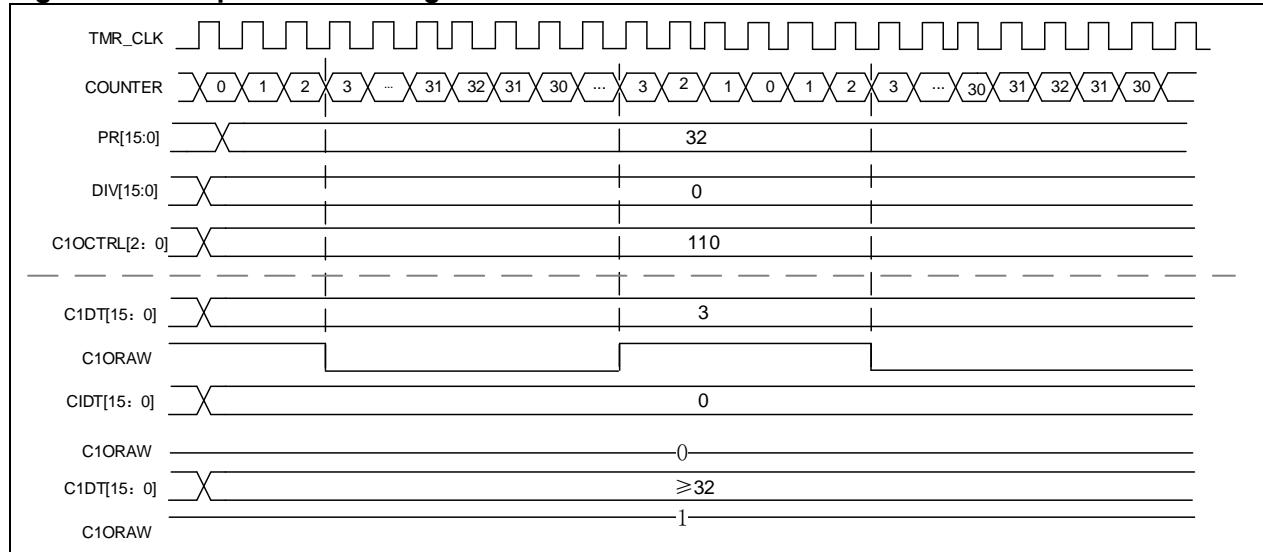
**Figure 14-113 C1ORAW toggles when counter value matches the C1DT value**

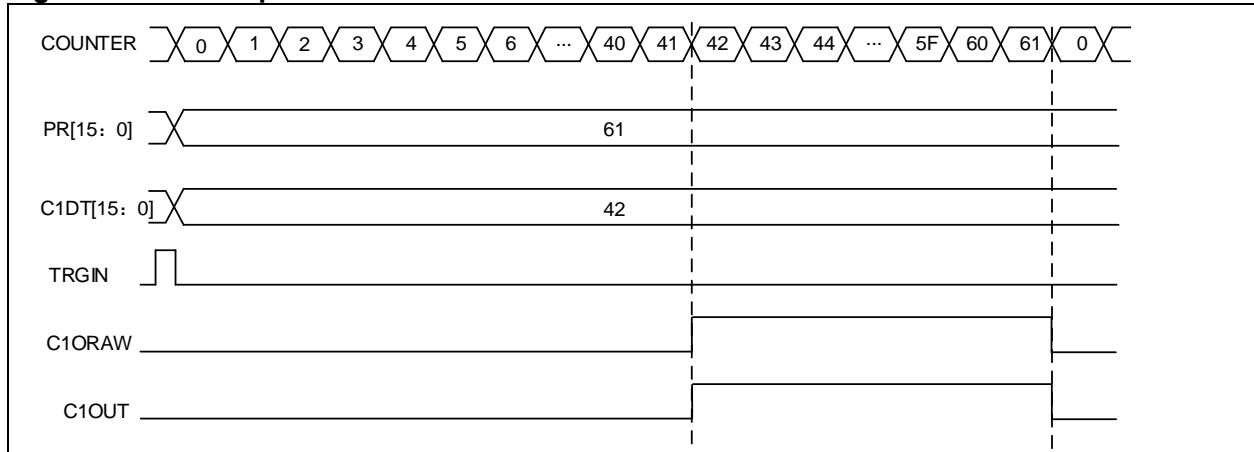


**Figure 14-114 Upcounting mode and PWM mode A**



**Figure 14-115 Up/down counting mode and PWM mode A**



**Figure 14-116 One-pulse mode**

### Master timer event output

When TMR is selected as the master timer, the following signal sources can be selected as TRGOUT signal to output to the slave timer, by setting the PTOS bit in the TMRx\_CTRL2 register.

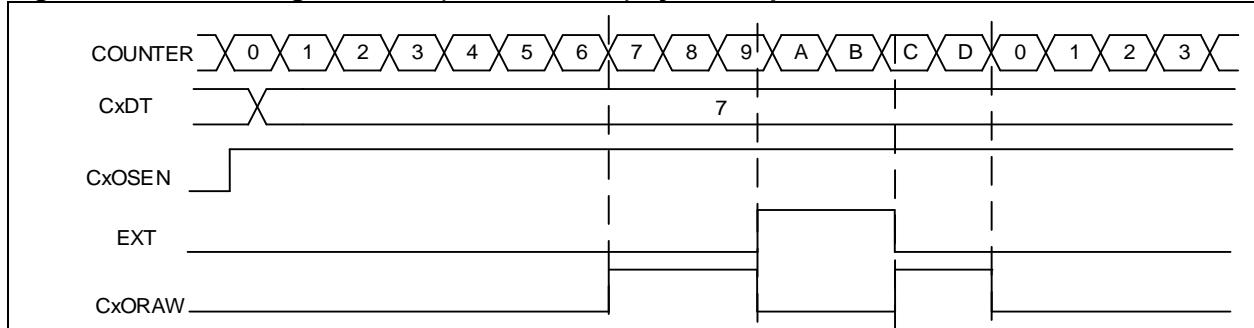
- PTOS=3'b000, TRGOUT outputs software overflow event (OVFSWTR bit in the TMRx\_SWEVT register).
- PTOS=3'b001, TRGOUT outputs counter enable signal.
- PTOS=3'b010, TRGOUT outputs counter overflow event.
- PTOS=3'b011, TRGOUT outputs capture and compare event.
- PTOS=3'b100, TRGOUT outputs C1ORAW signal.
- PTOS=3'b101, TRGOUT outputs C2ORAW signal.
- PTOS=3'b110, TRGOUT outputs C3ORAW signal.
- PTOS=3'b111, TRGOUT outputs C4ORAW signal.

### CxORAW clear

When the CxOSEN bit is set, the CxORAW signal for a given channel is cleared by applying a high level to the EXT input. The CxORAW signal remains unchanged until the next overflow event.

This function can only be used in output capture or PWM modes, and does not work in forced mode.

**Figure 14-117** shows the example of clearing CxORAW. When the EXT input is high, the CxORAW signal, which was originally high, is driven low; when the EXT is low, the CxORAW signal outputs the corresponding level according to the comparison result between the counter value and CxDT value.

**Figure 14-117 Clearing CxORAW(PWM mode A) by EXT input**

### Dead-time insertion

The channel 1 to 3 of the advanced timer contains a set of reverse channel output. This function is enabled by the CxCEN bit and its polarity is defined by CxCP. Refer to Table 14-17 for more information about the output state of CxOUT and CxCOUT.

The dead-time is activated when switching to IDLE state (OEN falling down to 0).

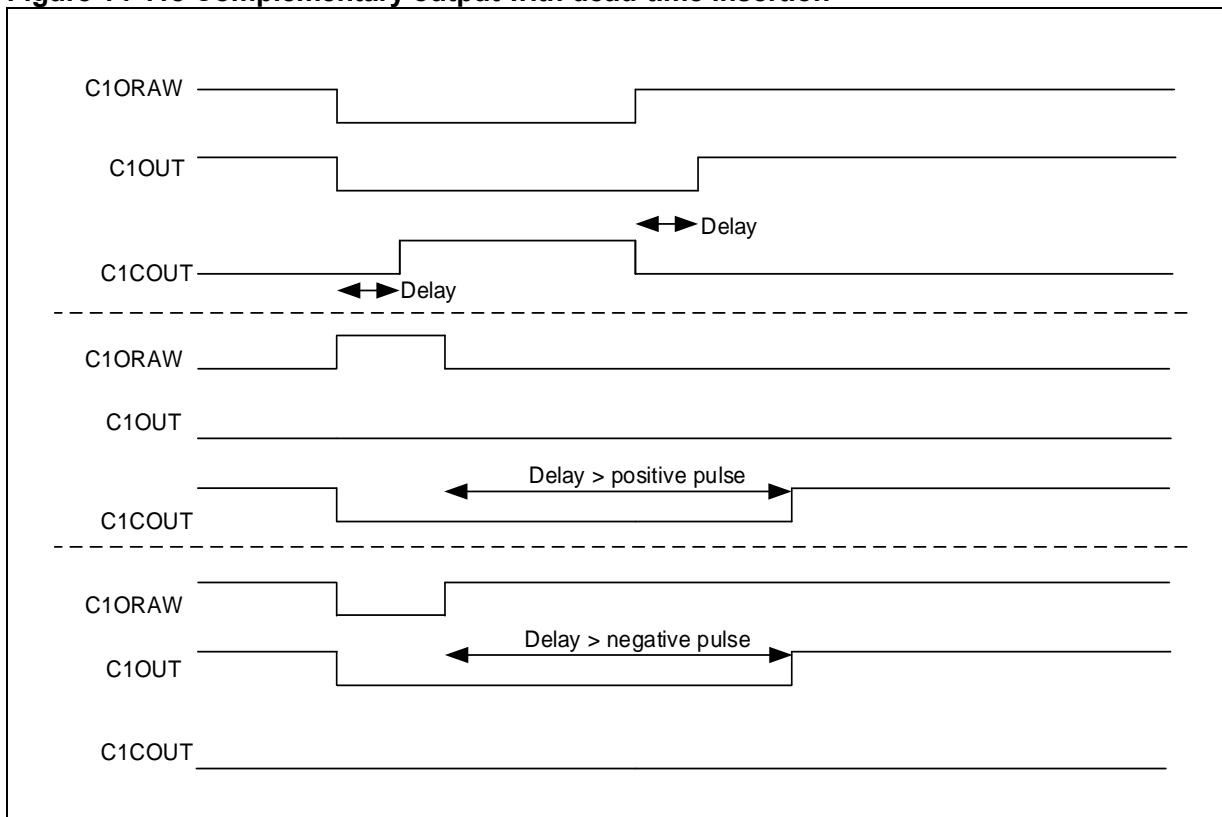
Setting both CxEN and CxCEN bits, and using DTC[7:0] bit to insert dead-time of different durations.

After the dead-time insertion, the rising edge of the CxOUT is delayed compared to the rising edge of the reference signal; the rising edge of the CxCOUT is delayed compared to the falling edge of the reference signal.

If the delay is greater than the width of the active output, and if C1OUT and C1COUT do not generate corresponding pulses, the dead-time should be less than the width of the active output.

**Figure 14-118** gives an example of dead-time insertion when CxP=0, CxCP=0, OEN=1, CxEN=1 and CxCEN=1.

**Figure 14-118 Complementary output with dead-time insertion**



#### 14.6.3.5 TMR brake function

When the brake function is enabled (BRKEN=1), the CxOUT and CxCOUT are jointly controlled by OEN, FCSODIS, FCSOEN, CxIOS and CxCIOS. But, CxOUT and CxCOUT cannot be set both to active level at the same time. Please refer to **Table 14-17** for more details.

The brake source can be the brake input pin or a clock failure event. The polarity is controlled by the BRKV bit.

When a brake event occurs, there are the following actions:

- The OEN bit is cleared asynchronously, and the channel output state is selected by setting the FCSODIS bit. This function works even if the MCU oscillator is off.
- Once OEN=0, the channel output level is defined by the CxIOS bit. If FCSODIS=0, the timer output is disabled, otherwise, the output enable remains high.
- When complementary outputs are used:
  - The outputs are first put in reset state, that is, inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
  - If the timer clock is still active, then the dead-time generator is activated. The CxIOS and CxCIOS bits are used to program the level after dead-time. Even in this case, the CxIOS and CxCIOS cannot be driven to their active level at the same time. It should be noted that because of synchronization on OEN, the dead-time duration is usually longer than usual (around 2 clk\_tmr clock cycles)
  - If FCSODIS=0, the timer releases the enable output, otherwise, it keeps the enable output; the enable output becomes high as soon as one of the CxEN and CxCEN bits becomes high.

- If the brake interrupt or DMA request is enabled, the brake status flag is set, and a brake interrupt or DMA request can be generated.
- If AOEN=1, the OEN bit is automatically set again at the next overflow event.

*Note: When the brake input is active, the OEN cannot be set, nor the status flag, BRKIF can be cleared.*

Figure 14-119 TMR output control

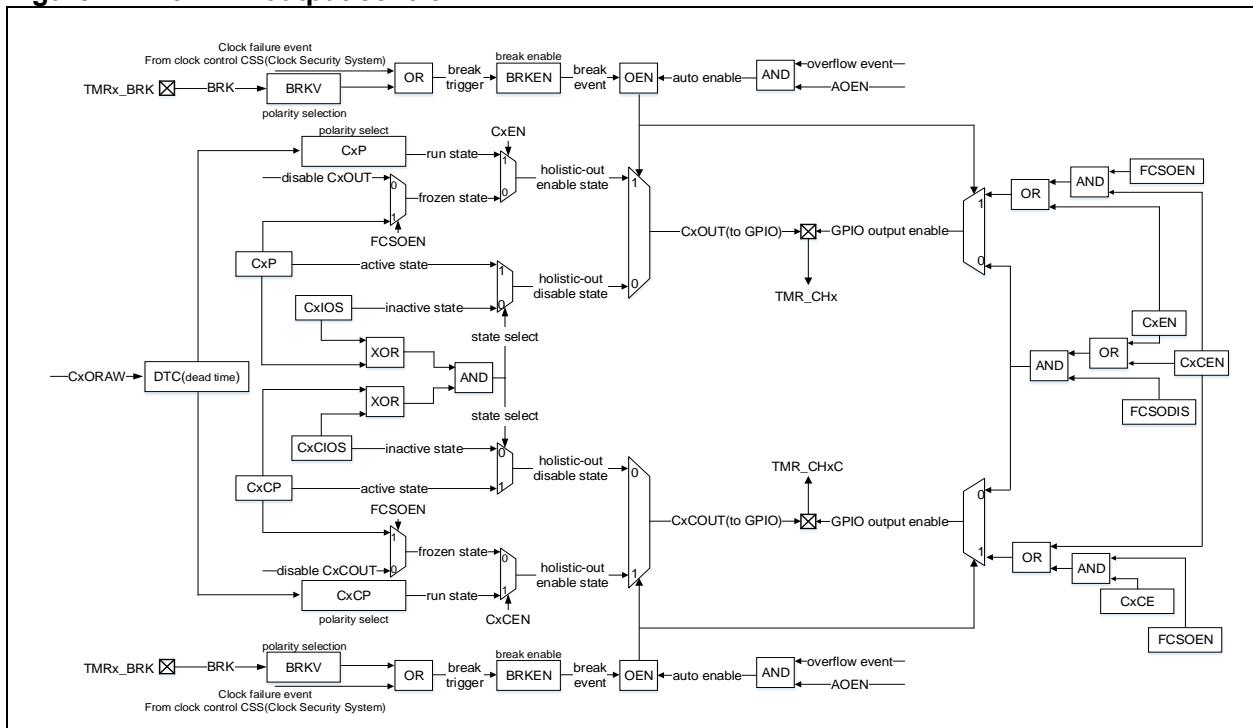
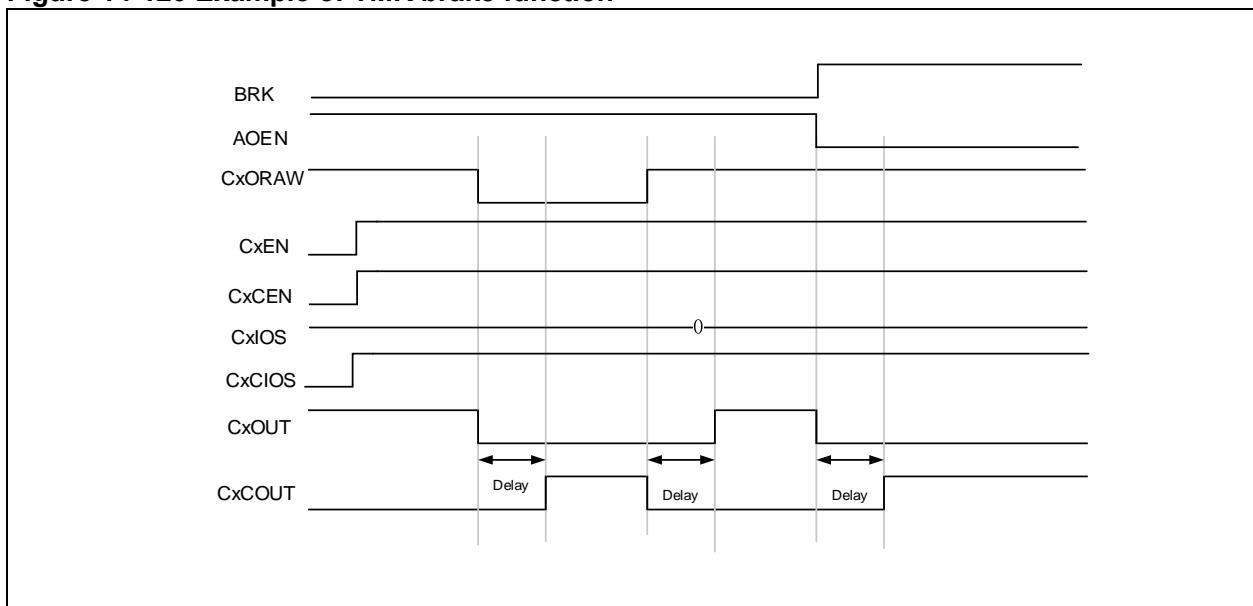


Figure 14-120 Example of TMR brake function



#### 14.6.3.6 TMR synchronization

The timers are linked together internally for timer synchronization. Master timer is selected by setting the PTOS[2: 0] bit; Slave timer is selected by setting the SMSEL[2: 0] bit.

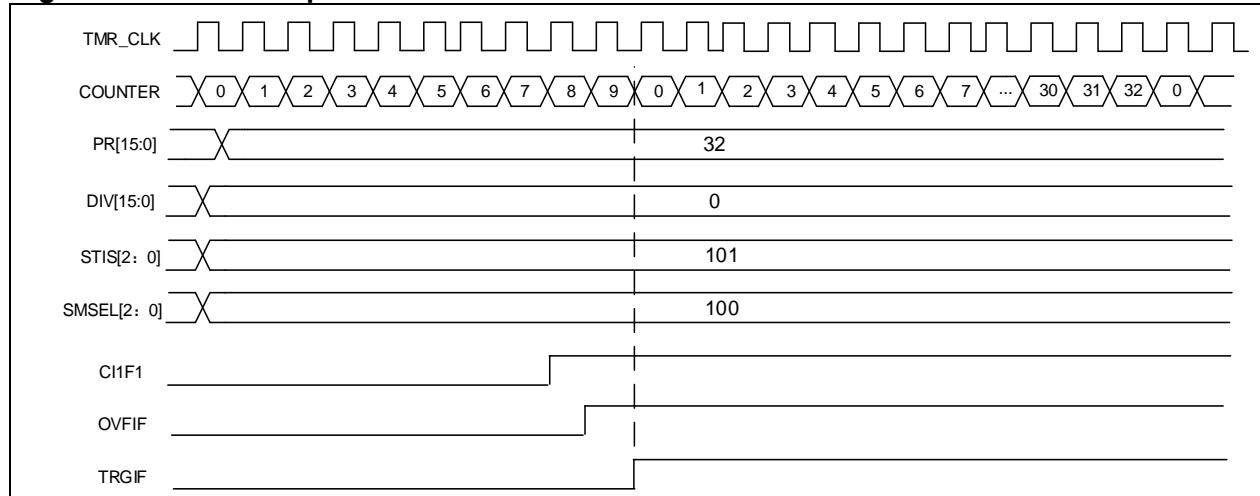
Slave modes include:

##### Slave mode: Reset mode

The counter and its prescaler can be reset by a selected trigger signal. An overflow event

can be generated when OVFS=0.

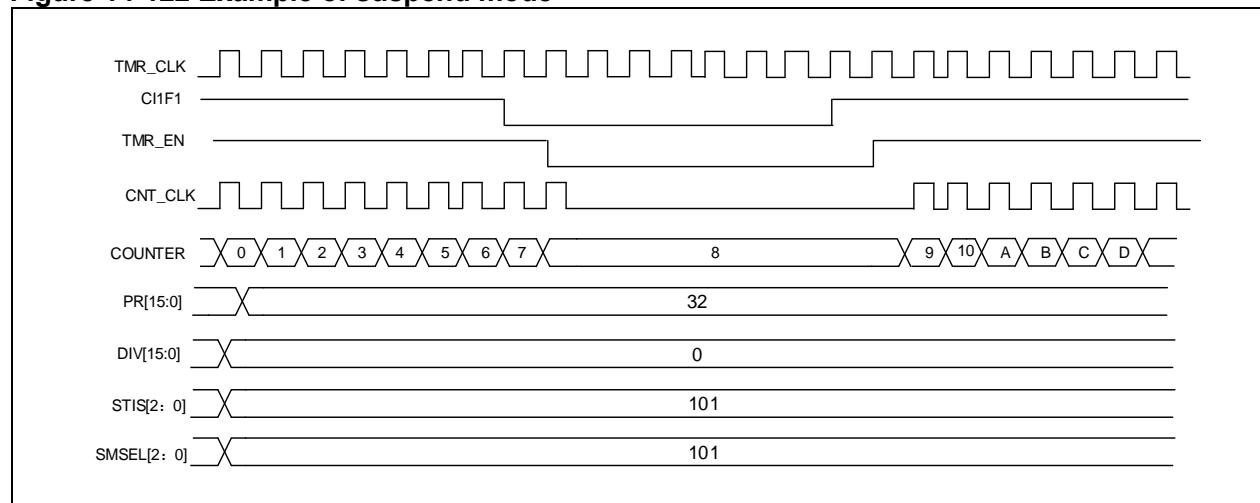
**Figure 14-121 Example of reset mode**



#### Slave mode: Suspend mode

In this mode, the counter is controlled by a selected trigger input. The counter starts counting when the trigger input is high and stops as soon as the trigger input is low.

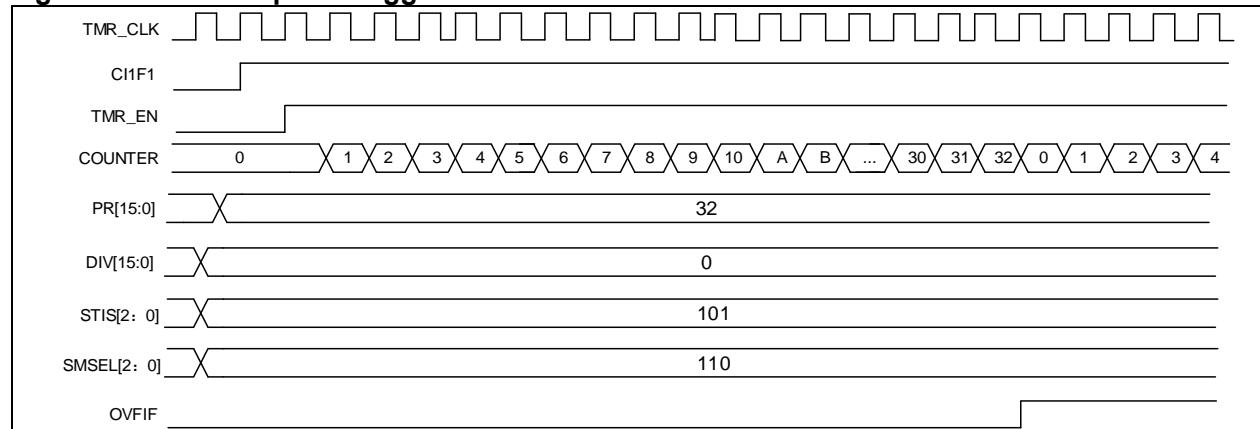
**Figure 14-122 Example of suspend mode**



#### Slave mode: Trigger mode

The counter can start counting on the rising edge of a selected trigger input (TMR\_EN=1)

**Figure 14-123 Example of trigger mode**



Please refer to [Section 14.2.3.5](#) for more details.

#### 14.6.3.7 Debug mode

When the microcontroller enters debug mode (Cortex<sup>TM</sup>-M4 core halted), the TMR1 counter stops

counting by setting the TMR1\_PAUSE in the DEBUG module.

#### 14.6.4 TMR1 registers

These peripheral registers must be accessed by word (32 bits).

TMR1 register are mapped into a 16-bit addressable space.

**Table 14-16 TMR1 register map and reset value**

Register	Offset	Reset value
TMR1_CTRL1	0x00	0x0000
TMR1_CTRL2	0x04	0x0000
TMR1_STCTRL	0x08	0x0000
TMR1_IDEN	0x0C	0x0000
TMR1ISTS	0x10	0x0000
TMR1_SWEVT	0x14	0x0000
TMR1_CM1	0x18	0x0000
TMR1_CM2	0x1C	0x0000
TMR1_CCTRL	0x20	0x0000
TMR1_CVAL	0x24	0x0000
TMR1_DIV	0x28	0x0000
TMR1_PR	0x2C	0x0000
TMR1_RPR	0x30	0x0000
TMR1_C1DT	0x34	0x0000
TMR1_C2DT	0x38	0x0000
TMR1_C3DT	0x3C	0x0000
TMR1_C4DT	0x40	0x0000
TMR1_BRK	0x44	0x0000
TMR1_DMACTRL	0x48	0x0000
TMR1_DMADT	0x4C	0x0000

##### 14.6.4.1 TMR1 control register1 (TMR1\_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9: 8	CLKDIV	0x0	rw	Clock division 00: Normal, $f_{DTS} = f_{CK\_INT}$ 01: Divided by 2, $f_{DTS} = f_{CK\_INT}/2$ 10: Divided by 4, $f_{DTS} = f_{CK\_INT}/4$ 11: Reserved
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 6: 5	TWCMSEL	0x0	rw	Two-way counting mode selection 00: One-way counting mode, depending on the OWCDIR bit 01: Two-way counting mode1, count up and down alternately, the output flag bit is set only when the counter counts down 10: Two-way counting mode2, count up and down alternately, the output flag bit is set only when the counter counts up 11: Two-way counting mode3, count up and down alternately, the output flag bit is set when the counter

				counts up / down
Bit 4	OWCDIR	0x0	rw	One-way count direction 0: Up 1: Down
Bit 3	OCMEN	0x0	rw	One cycle mode enable This bit is use to select whether to stop counting at an update event 0: The counter does not stop at an update event 1: The counter stops at an update event
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 1	OVFEN	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	TMR enable 0: Disabled 1: Enabled

#### 14.6.4.2 TMR1 control register2 (TMR1\_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	C4IOS	0x0	rw	Channel 4 idle output state
Bit 13	C3CIOS	0x0	rw	Channel 3 complementary idle output state
Bit 12	C3IOS	0x0	rw	Channel 3 idle output state
Bit 11	C2CIOS	0x0	rw	Channel 2 complementary idle output state
Bit 10	C2IOS	0x0	rw	Channel 2 idle output state
Bit 9	C1CIOS	0x0	rw	Channel 1 complementary idle output state OEN = 0 after dead-time: 0: C1OUTL=0 1: C1OUTL=1
Bit 8	C1IOS	0x0	rw	Channel 1 idle output state OEN = 0 after dead-time: 0: C1OUT=0 1: C1OUT=1
Bit 7	C1INSEL	0x0	rw	C1IN selection 0: CH1 pin is connected to C1IRAW input 1: The XOR result of CH1, CH2 and CH3 pins is connected to C1IRAW input
Bit 6: 4	PTOS	0x0	rw	Master TMR output selection This field is used to select the TMRx signal sent to the slave timer. 000: Reset 001: Enable 010: Update 011: Compare pulse 100: C1ORAW signal 101: C2ORAW signal 110: C3ORAW signal 111: C4ORAW signal
Bit 3	DRS	0x0	rw	DMA request source 0: Capture/compare event 1: Overflow event
Bit 2	CCFS	0x0	rw	Channel control bit flash selection This bit only acts on channels with complementary output. If the channel control bits are buffered: 0: Control bits are updated by setting the HALL bit 1: Control bits are updated by setting the HALL bit or a rising edge on TRGIN.
Bit 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	CBCTRL	0x0	rw	Channel buffer control This bit acts on channels that have complementary output. 0: CxEN, CxCEN and CxOCTRL bits are not buffered. 1: CxEN, CxCEN and CxOCTRL bits are not buffered.

#### 14.6.4.3 TMR1 slave timer control register (TMR1\_STCTRL)

Bit	Register	Reset value	Type	Description
Bit 15	ESP	0x0	rw	External signal polarity 0: High or rising edge 1: Low or falling edge
Bit 14	ECMBEN	0x0	rw	External clock mode B enable This bit is used to enable external clock mode B 0: Disabled 1: Enabled
Bit 13: 12	ESDIV	0x0	rw	External signal divide This field is used to select the frequency division of an external trigger 00: Normal 01: Divided by 2 10: Divided by 4

				11: Divided by 8
				External signal filter
				This field is used to filter an external signal. The external signal can be sampled only after it has been generated N times
				0000: No filter, sampling by $f_{DTS}$
				0001: $f_{SAMPLING} = f_{CK\_INT}$ , N=2
				0010: $f_{SAMPLING} = f_{CK\_INT}$ , N=4
				0011: $f_{SAMPLING} = f_{CK\_INT}$ , N=8
				0100: $f_{SAMPLING} = f_{DTS}/2$ , N=6
				0101: $f_{SAMPLING} = f_{DTS}/2$ , N=8
				0110: $f_{SAMPLING} = f_{DTS}/4$ , N=6
				0111: $f_{SAMPLING} = f_{DTS}/4$ , N=8
				1000: $f_{SAMPLING} = f_{DTS}/8$ , N=6
				1001: $f_{SAMPLING} = f_{DTS}/8$ , N=8
				1010: $f_{SAMPLING} = f_{DTS}/16$ , N=5
				1011: $f_{SAMPLING} = f_{DTS}/16$ , N=6
				1100: $f_{SAMPLING} = f_{DTS}/16$ , N=8
				1101: $f_{SAMPLING} = f_{DTS}/32$ , N=5
				1110: $f_{SAMPLING} = f_{DTS}/32$ , N=6
				1111: $f_{SAMPLING} = f_{DTS}/32$ , N=8
Bit 11: 8	ESF	0x0	rw	Subordinate TMR synchronization
				If enabled, master and slave timer can be synchronized.
				0: Disabled
				1: Enabled
Bit 7	STS	0x0	rw	Subordinate TMR input selection
				This field is used to select the subordinate TMR input.
				000: Internal selection 0 (IS0)
				001: Internal selection 1 (IS1)
				010: Internal selection 2 (IS2)
				011: Internal selection 3 (IS3)
				100: C1IRAW input detector (C1INC)
				101: Filtered input 1 (C1IF1)
				110: Filtered input 2 (C1IF2)
				111: External input (EXT)
				Refer to <a href="#">Table 14-14</a> for details on ISx for each timer.
Bit 6: 4	STIS	0x0	rw	Channel output switch selection
				This field is used to select the switch source of CxORAW
				0: Select EXT as the switch source of CxORAW
				1: Select CxORAW_OFF as the switch source of CxORAW
Bit 3	COSSEL	0x0	rw	Subordinate TMR mode selection
				000: Slave mode is disabled
				001: Encoder mode A
				010: Encoder mode B
				011: Encoder mode C
				100: Reset mode - Rising edge of the TRGIN input reinitializes the counter
Bit 2: 0	SMSEL	0x0	rw	101: Suspend mode - The counter starts counting when the TRGIN is high
				110: Trigger mode - A trigger event is generated at the rising edge of the TRGIN input
				111: External clock mode A - Rising edge of the TRGIN input clocks the counter
				Note: Refer to count mode section for details on encoder mode A/B/C.

#### 14.6.4.4 TMR1 DMA/interrupt enable register (TMR1\_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	TDEN	0x0	rw	Trigger DMA request enable 0: Disabled 1: Enabled

Bit 13	HALLDE	0x0	rw	HALL DMA request enable 0: Disabled 1: Enabled
Bit 12	C4DEN	0x0	rw	Channel 4 DMA request enable 0: Disabled 1: Enabled
Bit 11	C3DEN	0x0	rw	Channel 3 DMA request enable 0: Disabled 1: Enabled
Bit 10	C2DEN	0x0	rw	Channel 2 DMA request enable 0: Disabled 1: Enabled
Bit 9	C1DEN	0x0	rw	Channel 1 DMA request enable 0: Disabled 1: Enabled
Bit 8	OVFDEN	0x0	rw	Overflow event DMA request enable 0: Disabled 1: Enabled
Bit 7	BRKIE	0x0	rw	Brake interrupt enable 0: Disabled 1: Enabled
Bit 6	TIEN	0x0	rw	Trigger interrupt enable 0: Disabled 1: Enabled
Bit 5	HALLIEN	0x0	rw	HALL interrupt enable 0: Disabled 1: Enabled
Bit 4	C4IEN	0x0	rw	Channel 4 interrupt enable 0: Disabled 1: Enabled
Bit 3	C3IEN	0x0	rw	Channel 3 interrupt enable 0: Disabled 1: Enabled
Bit 2	C2IEN	0x0	rw	Channel 2 interrupt enable 0: Disabled 1: Enabled
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled 1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

#### 14.6.4.5 TMR1 interrupt status register (TMR1\_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	C4RF	0x0	rw0c	Channel 4 recapture flag Please refer to C1RF description.
Bit 11	C3RF	0x0	rw0c	Channel 3 recapture flag Please refer to C1RF description.
Bit 10	C2RF	0x0	rw0c	Channel 2 recapture flag Please refer to C1RF description.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing "0". 0: No capture is detected 1: Capture is detected.
Bit 8	Reserved	0x0	resd	Default value
Bit 7	BRKIF	0x0	rw0c	Brake interrupt flag This bit indicates whether the brake input is active or not. It is set by hardware and cleared by writing "0" 0: Inactive level 1: Active level
Bit 6	TRGIF	0x0	rw0c	Trigger interrupt flag This bit is set by hardware on a trigger event. It is cleared by writing "0". 0: No trigger event occurs 1: Trigger event is generated. Trigger event: an active edge is detected on TRGIN input, or any edge in suspend mode.
Bit 5	HALLIF	0x0	rw0c	HALL interrupt flag This bit is set by hardware on HALL event. It is cleared by writing "0". 0: No Hall event occurs. 1: Hall event is detected. HALL even: CxEN, CxCEN and CxOCTRL are updated.
Bit 4	C4IF	0x0	rw0c	Channel 4 interrupt flag Please refer to C1IF description.
Bit 3	C3IF	0x0	rw0c	Channel 3 interrupt flag Please refer to C1IF description.
Bit 2	C2IF	0x0	rw0c	Channel 2 interrupt flag Please refer to C1IF description.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT 0: No capture event occurs 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurs 1: Compare event is generated
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurs 1: Overflow event is generated. If OVFEN=0 and OVFS=0 in the TMRx_CTRL1 register: - An overflow event is generated when OVFG= 1 in the TMRx_SWEVE register; - An overflow event is generated when the counter CVAL is reinitialized by a trigger event.

#### 14.6.4.6 TMR1 software event register (TMR1\_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7	BRKSWTR	0x0	wo	Brake event triggered by software This bit is set by software to generate a brake event. 0: No effect 1: Generate a brake event.
Bit 6	TRGSWTR	0x0	rw	Trigger event triggered by software This bit is set by software to generate a trigger event. 0: No effect 1: Generate a trigger event.
Bit 5	HALLSWTR	0x0	wo	HALL event triggered by software This bit is set by software to generate a HALL event. 0: No effect 1: Generate a HALL event. Note: This bit acts only on channels that have complementary output.
Bit 4	C4SWTR	0x0	wo	Channel 4 event triggered by software Please refer to C1M description.
Bit 3	C3SWTR	0x0	wo	Channel 3 event triggered by software Please refer to C1M description.
Bit 2	C2SWTR	0x0	wo	Channel 2 event triggered by software Please refer to C1M description
Bit 1	C1SWTR	0x0	wo	Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.
Bit 0	OVFSWTR	0x0	wo	Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.

#### 14.6.4.7 TMR1 channel mode register1 (TMR1\_CM1)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the CxIx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

##### Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C2OSEN	0x0	rw	Channel 2 output switch enable
Bit 14: 12	C2OCTRL	0x0	rw	Channel 2 output control
Bit 11	C2OBEN	0x0	rw	Channel 2 output buffer enable
Bit 10	C2OIEN	0x0	rw	Channel 2 output enable immediately
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS register.
Bit 7	C1OSEN	0x0	rw	Channel 1 output switch enable 0: C1ORAW is not affected by EXT input. 1: Once a high level is detect on EXT input, clear C1ORAW.
Bit 6: 4	C1OCTRL	0x0	rw	Channel 1 output control This field defines the behavior of the original signal C1ORAW.

				000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMR1_CVAL=TMR1_C1DT 010: C1ORAW is low when TMR1_CVAL=TMR1_C1DT 011: Switch C1ORAW level when TMR1_CVAL=TMR1_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A — OWCDIR=0, C1ORAW is high once TMR1_C1DT>TMR1_CVAL, else low; — OWCDIR=1, C1ORAW is low once TMR1_C1DT<TMR1_CVAL, else high; 111: PWM mode B — OWCDIR=0, C1ORAW is low once TMR1_C1DT>TMR1_CVAL, else high; — OWCDIR=1, C1ORAW is high once TMR1_C1DT<TMR1_CVAL, else low. <i>Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.</i>
Bit 3	C1OBEN	0x0	rw	Channel 1 output buffer enable 0: Buffer function of TMRx_C1DT is disabled. The new value written to the TMRx_C1DT takes effect immediately. 1: Buffer function of TMRx_C1DT is enabled. The value to be written to the TMRx_C1DT is stored in the buffer register, and can be sent to the TMRx_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw	Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

**Input capture mode:**

Bit	Register	Reset value	Type	Description
Bit 15: 12	C2DF	0x0	rw	Channel 2 digital filter
Bit 11: 10	C2IDIV	0x0	rw	Channel 2 input divider
				Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 9: 8	C2C	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events.
Bit 7: 4	C1DF	0x0	rw	

				0000: No filter, sampling is done at $f_{DTS}$ 1000: $f_{SAMPLING} = f_{DTS}/8$ , N=6 0001: $f_{SAMPLING} = f_{CK\_INT}$ , N=2 1001: $f_{SAMPLING} = f_{DTS}/8$ , N=8 0010: $f_{SAMPLING} = f_{CK\_INT}$ , N=4 1010: $f_{SAMPLING} = f_{DTS}/16$ , N=5 0011: $f_{SAMPLING} = f_{CK\_INT}$ , N=8 1011: $f_{SAMPLING} = f_{DTS}/16$ , N=6 0100: $f_{SAMPLING} = f_{DTS}/2$ , N=6 1100: $f_{SAMPLING} = f_{DTS}/16$ , N=8 0101: $f_{SAMPLING} = f_{DTS}/2$ , N=8 1101: $f_{SAMPLING} = f_{DTS}/32$ , N=5 0110: $f_{SAMPLING} = f_{DTS}/4$ , N=6 1110: $f_{SAMPLING} = f_{DTS}/32$ , N=6 0111: $f_{SAMPLING} = f_{DTS}/4$ , N=8 1111: $f_{SAMPLING} = f_{DTS}/32$ , N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

#### 14.6.4.8 TMR1 channel mode register2 (TMR1\_CM2)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the Cxlx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

##### Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C4OSEN	0x0	rw	Channel 4 output switch enable
Bit 14: 12	C4OCTRL	0x0	rw	Channel 4 output control
Bit 11	C4OBEN	0x0	rw	Channel 4 output buffer enable
Bit 10	C4OIEN	0x0	rw	Channel 4 output enable immediately
				Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 9: 8	C4C	0x0	rw	Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7	C3OSEN	0x0	rw	Channel 3 output switch enable
Bit 6: 4	C3OCTRL	0x0	rw	Channel 3 output control
Bit 3	C3OBEN	0x0	rw	Channel 3 output buffer enable
Bit 2	C3OIEN	0x0	rw	Channel 3 output enable immediately
Bit 1: 0	C3C	0x0	rw	Channel 3 configuration

This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0':  
 00: Output  
 01: Input, C3IN is mapped on C3IFP3  
 10: Input, C3IN is mapped on C4IFP3  
 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

**Input capture mode:**

Bit	Register	Reset value	Type	Description
Bit 15: 12	C4DF	0x0	rw	Channel 4 digital filter
Bit 11: 10	C4IDIV	0x0	rw	Channel 4 input divider
				Channel 4 configuration
				This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 9: 8	C4C	0x0	rw	
Bit 7: 4	C3DF	0x0	rw	Channel 3 digital filter
Bit 3: 2	C3IDIV	0x0	rw	Channel 3 input divider
				Channel 3 configuration
				This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0': 00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 1: 0	C3C	0x0	rw	

**14.6.4.9 TMR1 Channel control register (TMR1\_CCTRL)**

Bit	Register	Reset value	Type	Description
Bit 15: 14	Reserved	0x0	resd	Kept its default value.
Bit 13	C4P	0x0	rw	Channel 4 polarity Please refer to C1P description.
Bit 12	C4EN	0x0	rw	Channel 4 enable Please refer to C1EN description.
Bit 11	C3CP	0x0	rw	Channel 3 complementary polarity Please refer to C1P description.
Bit 10	C3CEN	0x0	rw	Channel 3 complementary enable Please refer to C1EN description.
Bit 9	C3P	0x0	rw	Channel 3 polarity Please refer to C1P description.
Bit 8	C3EN	0x0	rw	Channel 3 enable Please refer to C1EN description.
Bit 7	C2CP	0x0	rw	Channel 2 complementary polarity Please refer to C1P description.
Bit 6	C2CEN	0x0	rw	Channel 2 complementary enable Please refer to C1EN description.
Bit 5	C2P	0x0	rw	Channel 2 polarity Please refer to C1P description.
Bit 4	C2EN	0x0	rw	Channel 2 enable Please refer to C1EN description.
Bit 3	C1CP	0x0	rw	Channel 1 complementary polarity 0: C1COUT is active high. 1: C1COUT is active low.
Bit 2	C1CEN	0x0	rw	Channel 1 complementary enable 0: Output is disabled. 1: Output is enabled.

Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured in output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured in input mode: It is C1CP/C1P bits that define the active edge of an input signal. 00: C1IN rising edge is active. When used as external trigger, C1IN is not inverted. 01: C1IN falling edge is active. When used as external trigger, C1IN is inverted. 10: Reserved. 11: C1IN rising edge and falling edge are both active. When used as external trigger, C1IN is not inverted
Bit0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

**Table 14-17 Complementary output channel CxOUT and CxCOUT control bits with brake function**

Control bit			Output state <sup>(1)</sup>			
OEN bit	FCSODIS bit	FCSOEN bit	CxEN bit	CxCEN bit	CxOUT output state	CxCOUT output state
1	X	0	0	0	Output disabled (no driven by the timer) CxOUT=0, Cx_EN=0	Output disabled (no driven by the timer) CxOUT=0, CxCEN=0
			0	1	Output disabled (no driven by the timer) CxOUT=0, Cx_EN=0	CxORAW + polarity, CxOUT= CxORAW xor CxCP, CxCEN=1
			1	0	CxORAW+ polarity CxOUT= CxORAW xor CxP, Cx_EN=1	Output disabled (no driven by the timer) CxOUT=0, CxCEN=0
			1	1	CxORAW+polarity+dead- time, Cx_EN=1	CxORAW inverted+polarity+dead- time, CxCEN=1
		1	0	0	Output disabled (no driven by the timer) CxOUT=CxP, Cx_EN=0	Output disabled (no driven by the timer) CxOUT=CxP, CxCEN=0
			0	1	Off-state (Output enabled with inactive level) CxOUT=CxP, Cx_EN=1	CxORAW + polarity, CxOUT= CxORAW xor CxCP, CxCEN=1
			1	0	CxORAW + polarity, CxOUT= CxORAW xor CxP, Cx_EN=1	Off-state (Output enabled with inactive level) CxOUT=CxP, CxCEN=1
			1	1	CxORAW+ polarity+dead- time, Cx_EN=1	CxORAW inverted+polarity+dead- time, CxCEN=1
0	X	0	0	0	Output disabled (corresponding IO disconnected from timer, and IO floating)	
		0	0	1	Asynchronously: CxOUT=CxP, Cx_EN=0, CxOUT=CxP, CxCEN=0;	
		0	1	0	If the clock is present: after a dead-time, CxOUT=CxIOS, CxCOUT=CxCIOS, assuming that CxIOS and CxCIOS do not correspond to CxOUT and	
		0	1	1		

			CxOUT active level.
1	0	0	CxEN=CxCEN=0: output disabled (corresponding IO disconnected from timer, and IO floating)
1	0	1	In other cases, Off-state (corresponding output channel inactive level)
1	1	0	Asynchronously: CxOUT =CxP, Cx_EN=1, CxCOUT=CxCP, CxCEN=1;
1	1	1	If the clock is present: after a dead-time, CxOUT=CxIOS, CxCOUT=CxCIOS, assuming that CxIOS and CxCIOS do not correspond to CxOUT and CxCOUT active level.

Note: If the two outputs of a channel are not used (CxEN = CxCEN = 0), CxIOS, CxCIOS, CxP and CxCP must be cleared.

Note: The state of the external I/O pins connected to the complementary CxOUT and CxCOUT channels depends on the CxOUT and CxCOUT channel state and the GPIO and the IOMUX registers.

#### 14.6.4.10 TMR1 counter value (TMR1\_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

#### 14.6.4.11 TMR1 division value (TMR1\_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK\_CNT} = f_{TMR\_CLK} / (DIV[15:0]+1)$ . The value of this register is transferred to the actual prescaler register when an overflow event occurs.

#### 14.6.4.12 TMR1 period register (TMR1\_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0000	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

#### 14.6.4.13 TMR1 repetition period register (TMR1\_RPR)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	rw	Kept at its default value.
Bit 7: 0	RPR	0x00	rw	Repetition of period value This field is used to reduce the generation rate of overflow events. An overflow event is generated when the repetition counter reaches 0.

#### 14.6.4.14 TMR1 channel 1 data register (TMR1\_C1DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effect immediately depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

#### 14.6.4.15 TMR1 channel 2 data register (TMR1\_C2DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C2DT	0x0000	rw	Channel 2 data register When the channel 2 is configured as input mode: The C2DT is the CVAL value stored by the last channel 2 input event (C1IN)

When the channel 2 is configured as output mode:  
C2DT is the value to be compared with the CVAL value.  
Whether the written value takes effect immediately  
depends on the C2OBEN bit, and the corresponding  
output is generated on C2OUT as configured.

---

#### 14.6.4.16 TMR1 channel 3 data register (TMR1\_C3DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C3DT	0x0000	rw	Channel 3 data register When the channel 3 is configured as input mode: The C3DT is the CVAL value stored by the last channel 3 input event (C1IN) When the channel 3 is configured as output mode: C3DT is the value to be compared with the CVAL value. Whether the written value takes effect immediately depends on the C3OBEN bit, and the corresponding output is generated on C3OUT as configured.

#### 14.6.4.17 TMR1 channel 4 data register (TMR1\_C4DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C4DT	0x0000	rw	Channel 4 data register When the channel 4 is configured as input mode: The C4DT is the CVAL value stored by the last channel 4 input event (C1IN) When the channel 3 is configured as output mode: C4DT is the value to be compared with the CVAL value. Whether the written value takes effect immediately depends on the C4OBEN bit, and the corresponding output is generated on C4OUT as configured.

#### 14.6.4.18 TMR1 brake register (TMR1\_BRK)

Bit	Register	Reset value	Type	Description
Bit 15	OEN	0x0	rw	Output enable This bit acts on the channels as output. It is used to enable CxOUT and CxCOUT outputs. 0: Disabled 1: Enabled
Bit 14	AOEN	0x0	rw	Automatic output enable OEN is set automatically at an overflow event. 0: Disabled 1: Enabled
Bit 13	BRKV	0x0	rw	Brake input validity This bit is used to select the active level of a brake input. 0: Brake input is active low. 1: Brake input is active high.
Bit 12	BRKEN	0x0	rw	Brake enable This bit is used to enable brake input. 0: Brake input is disabled. 1: Brake input is enabled.
Bit 11	FCSOEN	0x0	rw	Frozen channel status when holistic output enable This bit acts on the channels that have complementary output. It is used to set the channel state when the timer is inactive and MOEN=1. 0: CxOUT/CxCOUT outputs are disabled. 1: CxOUT/CxCOUT outputs are enabled. Output inactive level.
Bit 10	FCSODIS	0x0	rw	Frozen channel status when holistic output disable This bit acts on the channels that have complementary output. It is used to set the channel state when the timer is inactive and MOEN=0. 0: CxOUT/CxCOUT outputs are disabled. 1: CxOUT/CxCOUT outputs are enabled. Output idle level.
Bit 9: 8	WPC	0x0	rw	Write protection configuration This field is used to enable write protection. 00: Write protection is OFF. 01: Write protection level 3, and the following bits are write protected:

TMR1\_BRK: DTC, BRKEN, BRKV and AOEN  
 TMR1\_CTRL2: CxIOS and CxCIOS  
 10: Write protection level 2. The following bits and all bits in level 3 are write protected:  
 TMR1\_CCTRL: CxP and CxCP  
 TMR1\_BRK: FCSODIS and FCSOEN  
 11: Write protection level 1. The following bits and all bits in level 2 are write protected:  
 TMR1\_CMx: C2OCTRL and C2OBEN  
 Note: Once WPC>0, its content remains frozen until the next system reset.

Bit 7: 0	DTC	0x00	rw	Dead-time configuration This field defines the duration of the dead-time insertion. The 3-bit MSB of DTC[7: 0] is used for function selection: 0xx: DT = DTC [7: 0] * TDTS 10x: DT = (64+ DTC [5: 0]) * TDTS * 2 110: DT = (32+ DTC [4: 0]) * TDTS * 8 111: DT = (32+ DTC [4: 0]) * TDTS * 16
----------	-----	------	----	--

*Note: Based on lock configuration, AOEN, BRKV, BRKEN, FCSODIS, FCSOEN and DTC[7:0] can all be write protected. Thus it is necessary to configure write protection when writing to the TMRx\_BRK register for the first time.*

#### 14.6.4.19 TMR1 DMA control register (TMR1\_DMACTRL)

Bit	Register	Reset value	Type	Description
Bit 15:13	Reserved	0x0	resd	Kept at its default value.
Bit 12:8	DTB	0x00	rw	DMA transfer bytes This field defines the number of DMA transfers: 00000: 1 byte            00001: 2 bytes 00010: 3 bytes            00011: 4 bytes ..... 10000: 17 bytes            10001: 18 bytes
Bit 7:5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	ADDR	0x00	rw	DMA transfer address offset ADDR is defined as an offset starting from the address of the TMRx_CTRL1 register: 00000: TMRx_CTRL1 00001: TMRx_CTRL2 00010: TMRx_STCTRL .....

#### 14.6.4.20 TMR1 DMA data register (TMR1\_DMADT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DMADT	0x0000	rw	DMA data register A write/read operation to the DMADT register accesses any TMR register located at the following address: TMRx peripheral address + ADDR*4 to TMRx peripheral address + ADDR*4 + DTB*4

# 15 Window watchdog timer (WWDT)

## 15.1 WWDT introduction

The window watchdog downcounter must be reloaded in a limited time window to prevent the watchdog circuits from generating a system reset. The window watch dog is used to detect the occurrence of system malfunctions.

The window watchdog timer is clocked by a divided APB1\_CLK. The precision of the APB1\_CLK enables the window watchdog to take accurate control of the limited window.

## 15.2 WWDT main features

- 7-bit downcounter
- If the watchdog is enabled, a system reset is generated when the value of the downcounter is less than 0x40 or when the downcounter is reloaded outside the window.
- The downcounter can be reloaded by enabling the counter interrupt.

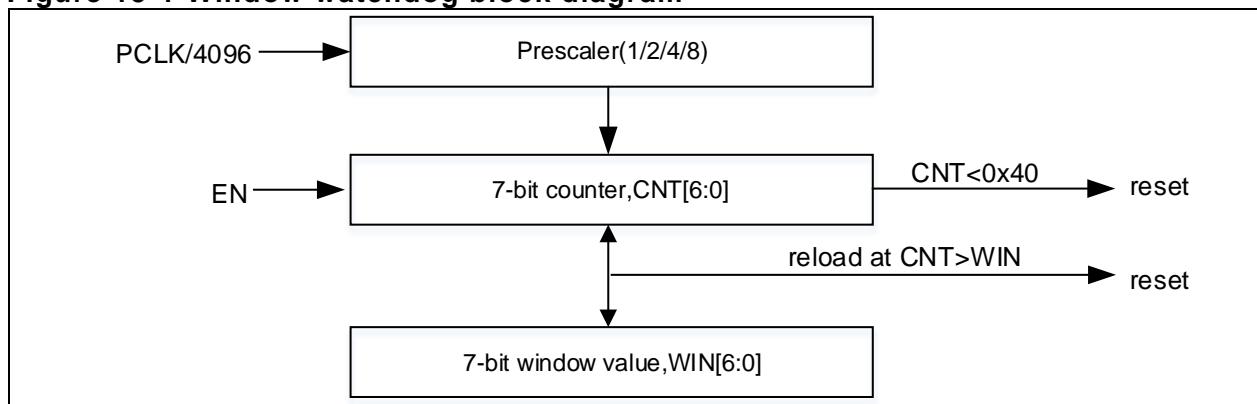
## 15.3 WWDT functional overview

When the watchdog is enabled, a system reset is generated at the following conditions:

When the 7-bit downcounter scrolls from 0x40 to 0x3F;

When the counter is reloaded while the 7-bit downcounter is greater than the value programmed in the window register.

**Figure 15-1 Window watchdog block diagram**



To prevent system reset, the counter must be reloaded only when its value is less than the value stored in the window register and greater than 0x40.

The WWDT counter is clocked by a divided APB1\_CLK, with the division factor being defined by the DIV[1: 0] bit in the WWDT\_CFG register. The counter value determines the maximum counter period before the watchdog generates a reset. The WIN[6: 0] bit can be used to configure the window value.

WWDT offers reload counter interrupt feature. If enabled, the WWDT will set the RLDF flag when the counter value reaches 0x40h, and an interrupt is generated accordingly. The interrupt service routine (ISTS) can be used to reload the counter to prevent a system reset. Note that if CNT[6]=0, setting the WWDTEN bit will generate a system reset, so the CNT[6] bit must be always set (CNT[6]=1) while writing to the WWDT\_CTRL register to prevent the occurrence of an immediate reset once the window watchdog is enabled.

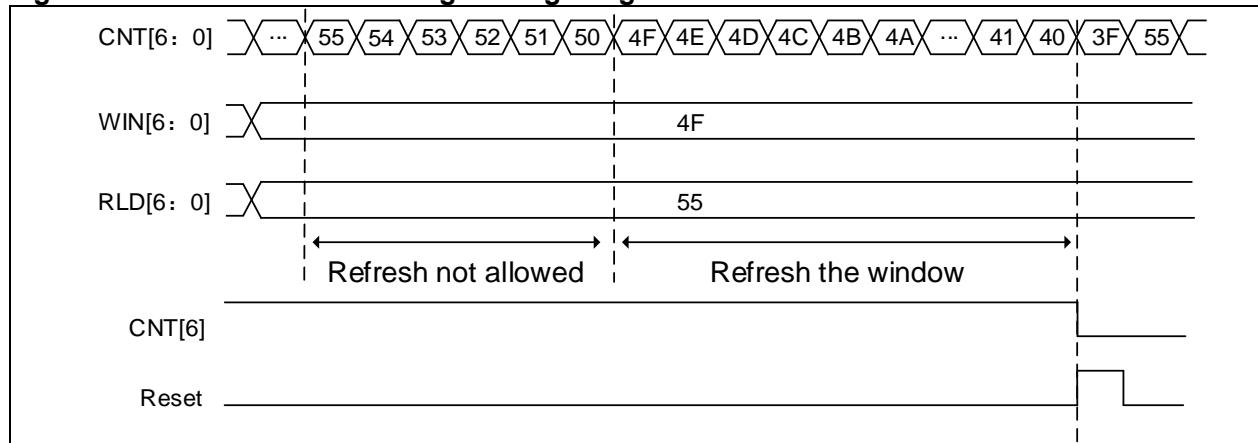
The formula to calculate the window watchdog time out:

$$T_{WWDT} = T_{PCLK1} \times 4096 \times 2^{DIV[1:0]} \times (CNT[5:0] + 1); (\text{ms})$$

Where:  $T_{PCLK1}$  refers to APB1 clock period, in ms.

**Table 15-1 Minimum and maximum timeout value when PCLK1=72 MHz**

Prescaler	Min. Timeout value	Max. Timeout value
0	56.5µs	3.64ms
1	113.5µs	7.28ms
2	227.5µs	14.56ms
3	455µs	29.12ms

**Figure 15-2 Window watchdog timing diagram**

## 15.4 Debug mode

When the microcontroller enters debug mode (Cortex™-M4 core halted), the WWDT counter stops counting by setting the WWDT\_PAUSE in the DEBUG module.

## 15.5 WWDT registers

These peripheral registers must be accessed by word (32 bits).

**Table 15-2 WWDT register map and reset value**

Register name	Offset	Reset value
WWDT_CTRL	0x00	0x7F
WWDT_CFG	0x04	0x7F
WWDT_STS	0x08	0x00

### 15.5.1 Control register (WWDT\_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	WWDTEN	0x0	rw1s	Window watchdog enable 0: Disabled 1: Enabled This bit is set by software, but can be cleared only after reset.
Bit 6: 0	CNT	0x7F	rw	Downcounter When the counter counts down to 0x3F, a reset is generated.

## 15.5.2 Configuration register (WWDT\_CFG)

Bit	Register	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Kept at its default value.
Bit 9	RLDIEN	0x0	rw	Reload counter interrupt 0: Disabled 1: Enabled
Bit 8: 7	DIV	0x0	rw	Clock division value 00: PCLK1 divided by 4096 01: PCLK1 divided by 8192 10: PCLK1 divided by 16384 11: PCLK1 divided by 32768
Bit 6: 0	WIN	0x7F	rw	Window value If the counter is reloaded while its value is greater than the window register value, a reset is generated. The counter must be reloaded between 0x40 and WIN[6: 0].

## 15.5.3 Status register (WWDT\_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 1	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 0	RLDF	0x0	rw0c	Reload counter interrupt flag This flag is set when the downcounter reaches 0x40. This bit is set by hardware and cleared by software.

# 16 Watchdog timer (WDT)

## 16.1 WDT introduction

The WDT is driven by a dedicated low-speed clock (LICK). Due to the lower clock accuracy of LICK, the WDT is best suited to the applications that have lower timing accuracy and can run independently outside the main application.

## 16.2 WDT main features

- 12-bit downcounter
- The counter is clocked by LICK (can work in Stop and Standby modes)
- A system reset is generated when the counter value is decremented to 0

## 16.3 WDT functional overview

### WDT enable:

Both software and hardware operations can be used to enable WDT. In other words, the WDT can be enabled by writing 0xCCCC to the WDT\_CMD register; or when the user enables the hardware watchdog through user system data area, the WDT will be automatically enabled after power-on reset.

### WDT reset:

When the counter value of the WDT counts down to 0, a WDT reset be generated. Thus the WDT\_CMD register must be written with the value 0xAAAA at regular intervals to reload the counter value to avoid the WDT reset.

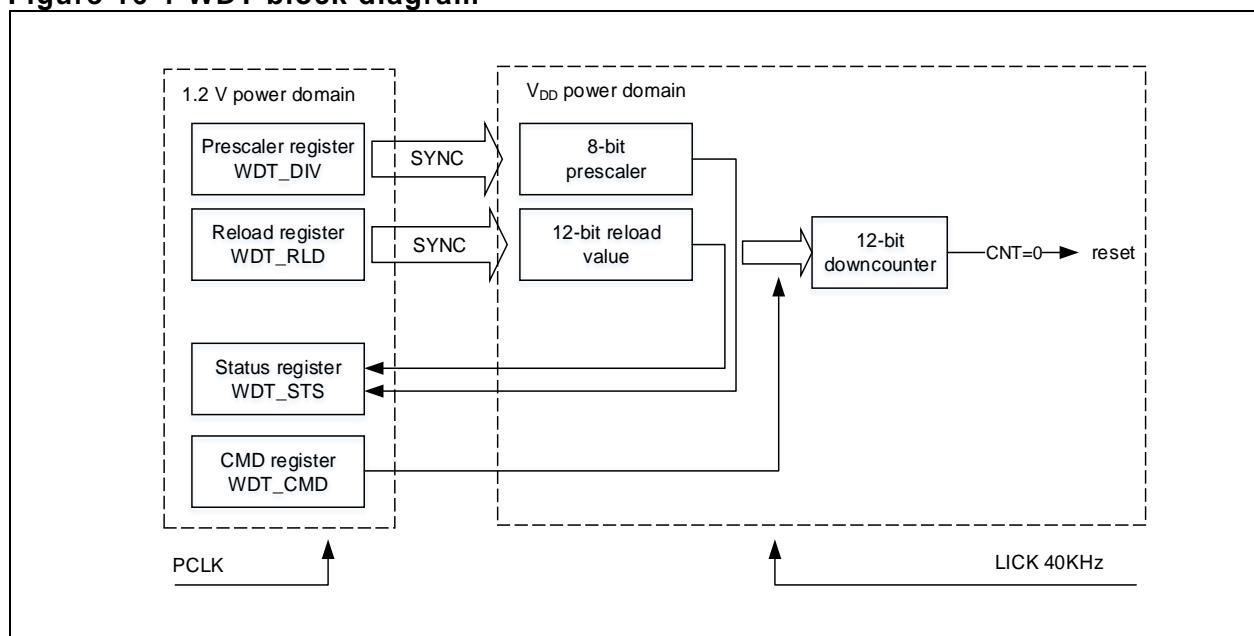
### WDT write-protected:

The WDT\_DIV and WDT\_RLD registers are write-protected. Writing the value 0x5555 to the WDT\_CMD register will unlock write protection. The update status of these two registers are indicated by the DIVF and RLDF bits in the WDT\_STS register. If a different value is written to the WDT\_CMD register, these two registers will be re-protected. Writing the value 0xAAAA to the WDT\_CMD register also enables write protection.

### WDT clock:

The WDT counter is clocked by the LICK. The LICK is an internal RC clock with a typical value of 40kHz, with its range falling between 30kHz and 60kHz. The timeout period is also within a certain range, so a margin should be taken into account when configuring timeout period. The LICK can be calibrated to obtain the WDT timeout with a relatively accuracy. Refer to [Section 4.1.1](#) for details.

**Figure 16-1 WDT block diagram**



**Table 16-1 WDT timeout period (LICK=40kHz)**

Prescaler divider	DIV[2: 0] bits	Min.timeout (ms) RLD[11: 0] = 0x000	Max. timeout (ms) RLD[11: 0] = 0xFFFF
/4	0	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6
/128	5	3.2	13107.2
/256	(6 or 7)	6.4	26214.4

## 16.4 Debug mode

When the microcontroller enters debug mode (Cortex™-M4 core halted), the WDT counter stops counting by setting the WDT\_PAUSE in the DEBUG module.

## 16.5 WDT registers

These peripheral registers must be accessed by words (32 bits).

**Table 16-2 WDT register and reset value**

Register name	Offset	Reset value
WDT_CMD	0x00	0x0000 0000
WDT_DIV	0x04	0x0000 0000
WDT_RLD	0x08	0x0000 0FFF
WDT_STS	0x0C	0x0000 0000

### 16.5.1 Command register (WDT\_CMD)

(Reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	CMD	0x0000	wo	Command register 0xAAAA: Reload counter 0x5555: Unlock write-protected WDT_DIV and WDT_RLD 0xCCCC: Enable WDT. If the hardware watchdog has been enabled, ignore this operation.

### 16.5.2 Divider register (WDT\_DIV)

Bit	Register	Reset value	Type	Description
Bit 31: 3	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 2: 0	DIV	0x0	rw	Clock division value 000: LICK divided by 4 001: LICK divided by 8 010: LICK divided by 16 011: LICK divided by 32 100: LICK divided by 64 101: LICK divided by 128 110: LICK divided by 256 111: LICK divided by 256 The write protection must be unlocked in order to enable write access to the register. The register can be read only when DIVF=0.

### 16.5.3 Reload register (WDT\_RLD)

(Reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11: 0	RLD	0xFFFF	rw	Reload value The write protection must be unlocked in order to enable write access to the register. The register can be read only when RLDF=0.

### 16.5.4 Status register (WDT\_STS)

(Reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 2	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 1	RLDF	0x0	ro	Reload value update complete flag 0: Reload value update complete 1: Reload value update is in process. The reload register WDT_RLD can be written only when RLDF=0
Bit 0	DIVF	0x0	ro	Division value update complete flag 0: Division value update complete 1: Division value update is in process. The divider register WDT_DIV can be written only when DIVF=0

# 17 Enhanced real-time clock (ERTC)

## 17.1 ERTC introduction

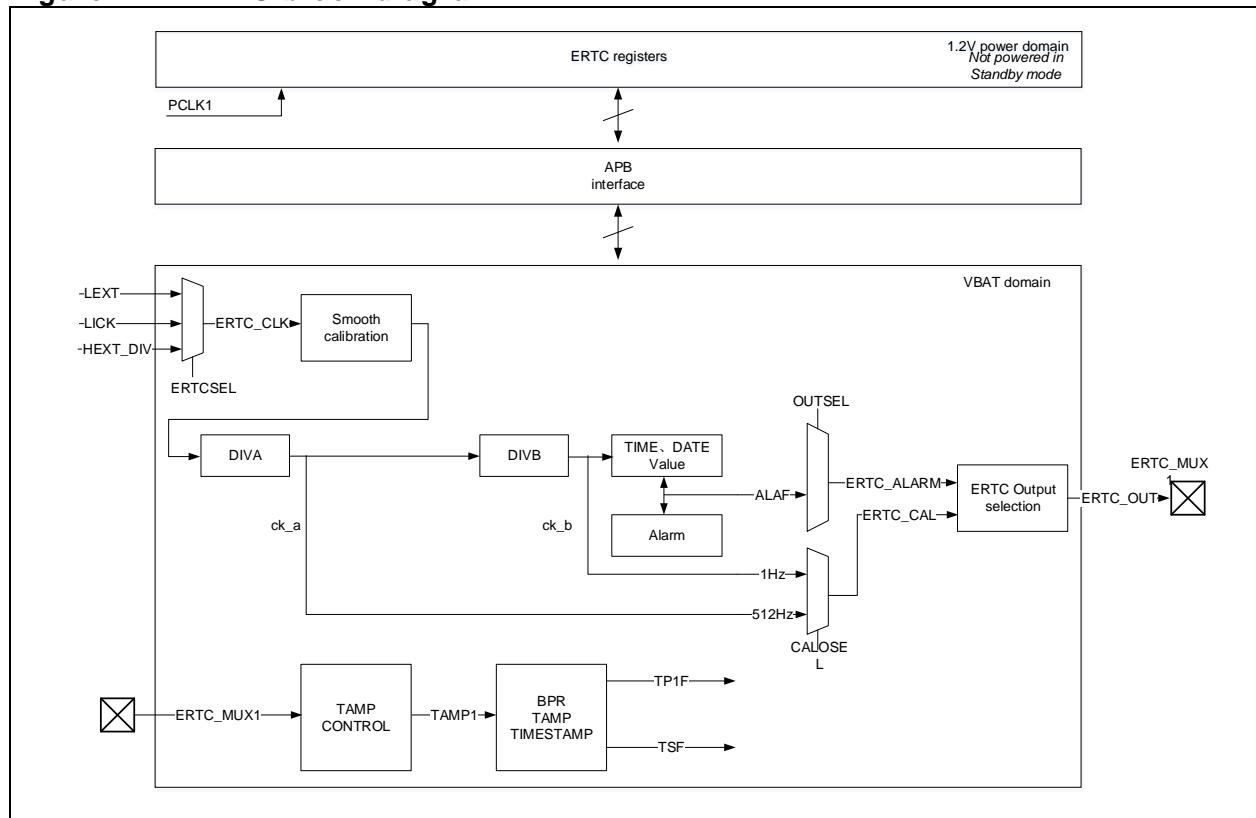
The real-time clock provides a calendar clock function. The time and date can be modified by modifying the ERTC\_TIME and ERTC\_DATE register.

The ERTC module is in battery powered domain, which means that it keeps running and free from the influence of system reset and VDD power off as long as VBAT is powered.

## 17.2 ERTC main features

- Real-time calendar (automatic processing of month days, including 28 (February in a common year), 29 (February in a leap year), 30 (a lunar month of 30 days) and 31 (a solar month of 31 days), where the current register being a multiple of 4 indicates a leap year), a set of alarm
- Reference clock detection
- 1 x programmable tamper detection, supporting time stamp feature
- Fine and coarse calibration support
- 5 x battery powered registers
- 4 x interrupts: alarm A, periodic auto-wakeup, tamper detection and time stamp
- Multiplexed function output, calibration clock output, alarm events or wakeup events
- Multiplexed function input, reference clock input, one-channel tamper detection and time stamp

**Figure 17-1 ERTC block diagram**



## 17.3 ERTC functional overview

### 17.3.1 ERTC clock

ERTC clock source (ERTC\_CLK) is selected via clock controller from a LEXT, LICK, and HEXT/32.

The ERTC embeds two dividers: A and B, programmed by the DIVA[6: 0] and DIVB[14: 0] respectively.

It is recommended that the DIVA is configured to a higher value in order to minimum power consumption.

After being divided by prescaler A and B, the ERTC\_CLK generates ck\_a and ck\_b clocks, respectively.

The ck\_a is used for subsecond update, while the ck\_b is used for calendar update and periodic autowakeup. The clock frequency of ck\_a and ck\_b can be obtained from the following equation:

$$F_{ck\_a} = \frac{f_{ERTC\_CLK}}{DIVA + 1}$$

$$F_{ck\_b} = \frac{f_{ERTC\_CLK}}{(DIVB + 1) \times (DIVA + 1)}$$

To obtain ck\_b with frequency of 1 Hz, DIVA=127, DIVB=255, and 32.768 kHz LEXT should be used. This ck\_b is then used for calendar update.

## 17.3.2 ERTC initialization

### ERTC register write protection

After a power-on reset, all ERTC registers are write protected except ERTC\_STS[14:8], ERTC\_TAMP and ERTC\_BPRx registers. Such protection mechanism is not affected by the system reset. The ERTC registers can be written only after write protection is unlocked.

To unlock the write protection of ERTC registers, the steps below should be respected:

1. Enable power interface clock by setting PWCEN=1 in the CRM\_APB1EN register
2. Unlock write protection of the battery powered domain by setting BPWEN=1 in the PWC\_CTRL register
3. Write 0xCA and 0x53 to the ERTC\_WP register in sequence. Writing an incorrect key will activate the write protection again.

**Table 17-1** lists the ERTC registers that can be configured only after the write protection is unlocked and when the initialization mode is entered.

**Table 17-1 ERTC register configuration**

Register	ERTC_WP enabled	Whether to enter initialization mode	Others
ERTC_TIME	Y	Y	-
ERTC_DATE	Y	Y	-
ERTC_CTRL	Y	Bit 6 and 4 only	-
ERTC_STS	Y, except [14: 8]	-	-
ERTC_DIV	Y	Y	-
ERTC_ALA	Y	N	Configurable when ALAWF =1
ERTC_WP	-	-	-
ERTC_SBS	-	-	-
ERTC_TADJ	Y	N	Configurable when TADJF=0
ERTC_TSTM	-	-	-
ERTC_TSDT	-	-	-
ERTC_TSSBS	-	-	-
ERTC_SCAL	Y	N	Configurable when CALUPDF=0
ERTC_TAMP	N	N	-
ERTC_ALASBS	Y	N	Configurable when ALAWF =1
ERTC_BPRx	N	N	-

### Clock and calendar initialization

After the register write protection is unlocked, follow the procedure below for clock and calendar initialization:

1. Set the IMEN =1 to enter initialization mode
2. Wait until the initialization flag INITF bit is set
3. Configure DIVB and DIVA.
4. Configure the clock and calendar values.
5. Leave the initialization mode by clearing the IMEN bit. Wait until the UPDF bit is set, indicating the completion of the calendar update. The calendar starts counting.

The ERTC also allows the fine-tuning for daylight saving time and clock.

Daylight saving time feature: It is used to increase (ADD1H=1) or decrease (DEC1H=1) one hour in the calendar, without completing the whole initialization process.

Clock calibration: It is used for the fine calibration of the current clock. If only DECSBS[14: 0] is configured, the value will be added to the DIVB counter and a clock latency will be generated. If only ADD1S bit is set, the current clock will increase by one second. If both DECSBS[14: 0] and ADDIS bit are configured, the clock will increase by a fraction of a second.

Time latency (ADD1S=0): DECSBS/(DIVB+1)

Time advance (ADD1S=1): 1-(DECSBS/(DIVB+1))

*Note: To avoid subsecond overflow, SBS[15]=0 must be asserted before setting the ERTC\_TADJ register.*

### Reading the calendar

The ERTC offers two different ways to read the calendar, that is, synchronous read (DREN=0) and asynchronous read (DREN=1).

In the case of DREN=0, the clock and calendar values can be obtained by reading a synchronous shadow register via the PCLK1. The UPDF bit is set each time the shadow register is synchronized with the ERTC calendar value located in the battery powered domain. The synchronization is performed every two ERTC\_CLK. The shadow register is reset by a system reset. To ensure consistency between the 3 values (ERTC\_SBS, ERTC\_TIME and ERTC\_DATE registers), reading lower-order registers will lock the values in the higher-order registers until the ERTC\_DATE register is read. For example, reading the ERTC\_SBS register will lock the values in the ERTC\_TIME and ERTC\_DATE registers.

In the case of DREN=1, the ERTC will perform direct read access to the ERTC clock and calendar located in the battery powered domain with the PCLK1, avoiding the occurrence of errors caused by time synchronization. In this mode, the UPDF flag is cleared by hardware. To ensure the data is correct when reading clock and calendar, the software must read the clock and calendar registers twice, and compare the results of two read operations. If the result is not aligned, read again until that the results of two read accesses are consistent. Besides, it is also possible to compare the least significant bits of the two read operations to determine their consistency.

*Note: In Standby and Deepsleep modes, the current calendar values are not copied into the shadow registers. When waking up from these two modes, UPDF=0 must be asserted, and then wait until UPDF=1, to ensure that the latest calendar value can be read. In synchronous read (DREN=0) mode, the frequency of the PCLK1 must be at least seven times the ERTC\_CLK frequency. In asynchronous read (DREN=1), an additional APB cycle is required to complete the read operations of the calendar register.*

### Alarm clock initialization

The ERTC offers alarm clock A and interrupts.

The alarm clock value is programmed with the ERTC\_ALASBS/ERTC\_ALA. When the programmed alarm value matches the calendar value, an alarm event is generated if an alarm clock is enabled. The MASKx bit can be used to selectively mask calendar fields. The calendar fields, which are masked, are not allocated with an alarm clock.

To configure the alarm clock A the following steps should be respected:

1. Disable alarm clock A (by setting ALAEN=0)
2. Wait until the ALAWF bit is set to enable write access to the alarm clock A
3. Configure alarm clock A register (ERTC\_ALA and ERTC\_ALBSBS)
4. Enable alarm clock A by setting ALAEN=1

*Note: If MASK1=0 in the ERTC\_ALA, the alarm clock can work normally only when*

*the DIVB value is at least equal to 3.*

### 17.3.3 ERTC calibration

#### Smooth digital calibration:

Smooth digital calibration has a higher and well-distributed performance than the coarse digital calibration. The calibration is performed by increasing or decreasing ERTC\_CLK in an evenly manner.

The smooth digital calibration period is around  $2^{20}$  ERTC\_CLK (32 seconds) when the ERTC\_CLK is 32.768 kHz. The DEC[8: 0] bit specifies the number of pulses to be masked during the  $2^{20}$  ERTC\_CLK cycles. A maximum of 511 pulses can be removed. When the ADD is set, 512 pulses can be inserted during the  $2^{20}$  ERTC\_CLK cycles. When DEC[8: 0] and ADD are sued together, a deviation ranging from -511 to +512 ERTC\_CLK cycles can be added during the  $2^{20}$  ERTC\_CLK cycles.

The effective calibrated frequency ( $F_{SCAL}$ ):

$$F_{SCAL} = F_{ERTC\_CLK} \times [1 + \frac{ADD \times 512 - DEC}{2^{20} + DEC - ADD \times 512}]$$

When the divider A is less than 3, the calibration operates as if ADD was equal to 0. The divider B value should be reduced so that each second is accelerated by 8 ERTC\_CLK cycles, which means that 256 ERTC\_CLK cycles are added every 32 seconds. When DEC[8: 0] and ADD are sued together, a deviation ranging from -255 to +256 ERTC\_CLK cycles can be added during the  $2^{20}$  ERTC\_CLK cycles.

At this point, the effective calibrated frequency ( $F_{SCAL}$ )

$$F_{SCAL} = F_{ERTC\_CLK} \times [1 + \frac{256 - DEC}{2^{20} + DEC - 256}]$$

It is also possible to select 8 or 16-second digital calibration period through the CAL8 and CAL16 bits. The 8-second period takes priority over 16-second. In other words, when both 8-second and 16-second are enabled, 8-second calibration period prevails.

The CALUPDF flag in the ERTC indicates the calibration status. During the configuration of ERTC\_SCAL registers, the CALUPDF bit is set, indicating that the calibration value is being updated; Once the calibration value is successfully applied, this bit is cleared automatically, indicating the completion of the calibration value update.

### 17.3.4 Time stamp

When time stamp event is detected on the tamper pin (valid edge is detected), the current calendar value will be stored to the time stamp register.

When a time stamp event occurs, the time stamp flag bit (TSF) in the ERTC\_STS register will be set. If a new time stamp event is detected when time stamp flag (TSF) is already set, then the time stamp overflow flag (TSOF) will be set, but the time stamp registers will remain the result of the last event. By setting the TSIEN bit, an interrupt can be generated when a time stamp event occurs.

Usage of time stamp:

1. How to enable time stamp when a valid edge is detected on a tamper pin
  - Select a rising edge or falling edge to trigger time stamp by setting the TSEDG bit
  - Enable time stamp by setting TSEN=1
2. How to save time stamp on a tamper event
  - Configure tamper detection registers
  - Enable tamper detection time stamp by setting TPTSEN=1

*Note: The TSF bit will be set after two ck\_a cycles following a time stamp event. It is suggested that users poll TSOF bit when the TSF is set.*

### 17.3.5 Tamper detection

The ERTC offers one tamper detection mode: TAMP1. It can be configured as a level detection with filter or edge detection. TAMP1 is mapped onto the tamper pin ERTC\_MUX1.

The TP1F will be set when a valid tamper event is detected. An interrupt will also be generated if a tamper detection interrupt is enabled. If the TPTSEN bit is already set, a time stamp event will be generated accordingly. Once a tamper event occurs, the battery powered registers will be reset so as to ensure data security in the battery powered domain.

#### How to configure edge detection

1. Select edge detection by setting TPFLT=00, and select a valid edge (TP1EDG)
2. According to your needs, configure whether to activate a time stamp on a tamper event (TPTSEN=1)
3. According to your needs, enable a tamper detection interrupt (TPIEN=1)
4. Enable TAMP1 (setting TP1EN=1)

#### How to configure level detection with filter

1. Select level detection with filter, and valid level sampling times (TPFLT≠00)
2. Select tamper detection active level (TP1EDG)
3. Select tamper detection sampling frequency (through the TPFREQ bit )
4. According to your needs, enable tamper detection pull-up (setting TPPU=1). When TPPU=1 is asserted, tamper detection pre-charge time must be configured through the TPPR bit
5. According to your needs, configure whether to activate a time stamp on a tamper event (TPTSEN=1)
6. According to your needs, enable a tamper interrupt (TPIEN=1)
7. Enable TMAP1 by setting TP1EN=1.

While configuring edge detection mode, the following two points deserve our attention:

1. If a rising edge is configured to enable tamper detection, and the tamper detection pin turns to high level before tamper detection is enabled, then a tamper event will be detected right after the tamper detection is enabled;
2. If a falling edge is configured to enable tamper detection, and the tamper detection pin turns to low level before tamper detection is enabled, then a tamper event will be detected right after the tamper detection is enabled.

### 17.3.6 Multiplexed function output

ERTC provides a set of multiplexed function output for the following events:

1. Clocks calibrated (OUTSEL=0 and CALOEN=1)
  - Output 512Hz (CALOSEL=0)
  - Output 1Hz (CALOSEL=1)
2. Alarm clock A (OUTSEL=1)
3. Wakeup events (OUTSEL=3)

When alarm clock or wakeup events are selected (OUTSEL≠0), it is possible to select output type (open-drain or push-pull) with the OUTTYPE bit, and output polarity with the OUTP bit.

### 17.3.7 ERTC wakeup

ERTC can be woken up by alarm clocks, time stamps or tamper events. To enable an ERTC interrupt, configure as follows:

1. Configure the EXINT line corresponding to ERTC interrupts as an interrupt mode and enable it, and select a rising edge
2. Enable a NVIC channel corresponding to an ERTC interrupt
3. Enable an ERTC interrupt

**Table 17-2** lists the ERTC clock sources, events and interrupts that are able to wakeup low-power modes.

**Table 17-2 ERTC low-power mode wakeup**

Clock sources	Events	Wake up Sleep	Wake Deepsleep	up	Wakeup Standby
HEXT	Alarm clock A	✓	✗	✗	✗
	Time stamp	✓	✗	✗	✗
	Tamper event	✓	✗	✗	✗
LICK	Alarm clock A	✓	✓	✓	✓
	Time stamp	✓	✓	✓	✓
	Tamper event	✓	✓	✓	✓
LEXT	Alarm clock A	✓	✓	✓	✓
	Time stamp	✓	✓	✓	✓
	Tamper event	✓	✓	✓	✓

**Table 17-3 Interrupt control bits**

Interrupt events	Event flag	Interrupt enable bit	EXINT line
Alarm clock A	ALAF	ALAIEN	17
Time stamp	TSF	TSIEN	19
Tamper event	TP1F	TPIEN	19

## 17.4 ERTC registers

These peripheral registers have to be accessed by words (32 bits). Writing ERTC registers takes around four ERTC\_CLK clock cycles of APB1, while reading ERTC registers takes around eight APB1\_CLK clock cycles of APB1.

ERTC registers are 16-bit addressable registers.

**Table 17-4 ERTC register map and reset values**

Register name	Offset	Reset value
ERTC_TIME	0x00	0x0000 0000
ERTC_DATE	0x04	0x0000 2101
ERTC_CTRL	0x08	0x0000 0000
ERTC_STS	0x0C	0x0000 0007
ERTC_DIV	0x10	0x007F 00FF
ERTC_ALA	0x1C	0x0000 0000
ERTC_WP	0x24	0x0000 0000
ERTC_SBS	0x28	0x0000 0000
ERTC_TADJ	0x2C	0x0000 0000
ERTC_TSTM	0x30	0x0000 0000
ERTC_TSDT	0x34	0x0000 000D
ERTC_TSSBS	0x38	0x0000 0000
ERTC_SCAL	0x3C	0x0000 0000
ERTC_TAMP	0x40	0x0000 0000
ERTC_ALASBS	0x44	0x0000 0000
ERTC_BPRx	0x50-0x50	0x0000 0000

### 17.4.1 ERTC time register (ERTC\_TIME)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22	AMPM	0x0	rw	AM/PM 0: AM 1: PM Note: This bit is applicable for 12-hr format only. It is 0 for 24-hr format instead.

Bit 21: 20	HT	0x0	rw	Hour tens
Bit 19: 16	HU	0x0	rw	Hour units
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14: 12	MT	0x0	rw	Minute tens
Bit 11: 8	MU	0x0	rw	Minute units
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6: 4	ST	0x0	rw	Second tens
Bit 3: 0	SU	0x0	rw	Second units

### 17.4.2 ERTC date register (ERTC\_DATE)

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23: 20	YT	0x0	rw	Year tens
Bit 19: 16	YU	0x0	rw	Year units
				Week day 0: Forbidden 1: Monday 2: Tuesday 3: Wednesday 4: Thursday 5: Friday 6: Saturday 7: Sunday
Bit 15: 13	WK	0x1	rw	
Bit 12	MT	0x0	rw	Month tens
Bit 11: 8	MU	0x1	rw	Month units
Bit 7: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5: 4	DT	0x0	rw	Date tens
Bit 3: 0	DU	0x1	rw	Date units

### 17.4.3 ERTC control register (ERTC\_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23	CALOEN	0x0	rw	Calibration output enable 0: Calibration output disabled 1: Calibration output enabled
Bit 22: 21	OUTSEL	0x0	rw	Output source selection 00: Output source disabled 01: Alarm clock A 10: Alarm clock B 11: Wakeup events
Bit 20	OUTP	0x0	rw	Output polarity 0: High 1: Low
Bit 19	CALOSEL	0x0	rw	Calibration output selection 0: 512Hz 1: 1Hz
Bit 18	BPR	0x0	rw	Battery powered domain data register This bit in the battery powered domain is not affected by a system reset. It is used to store the daylight saving time change or others that need to be saved permanently.
Bit 17	DEC1H	0x0	wo	Decrease 1 hour 0: No effect 1: Subtract 1 hour Note: This bit is applicable only when the current hour is not 0. The next second takes effect when this bit is set (don't set this bit when the hour is being incremented)
Bit 16	ADD1H	0x0	wo	Add 1 hour

				0: No effect 1: Add 1 hour Note: The next second takes effect when this bit is set (don't set this bit when the hour is being incremented)
Bit 15	TSIEN	0x0	rw	Timestamp interrupt enable 0: Timestamp interrupt disabled 1: Timestamp interrupt enabled
Bit 14: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	ALAIEN	0x0	rw	Alarm A interrupt enable 0: Alarm A interrupt disabled 1: Alarm A interrupt enabled
Bit 11	TSEN	0x0	rw	Timestamp enable 0: Timestamp disabled 1: Timestamp enabled
Bit 10: 9	Reserved	0x0	resd	Kept at its default value.
Bit 8	ALAEN	0x0	rw	Alarm A enable 0: Alarm A disabled 1: Alarm A enabled
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	HM	0x0	rw	Hour mode 0: 24-hour format 1: 12-hour format
Bit 5	DREN	0x0	rw	Date/time register direct read enable 0: Date/time register direct read disabled. ERTC_TIME, ERTC_DATE and ERTC_SBS values are taken from the synchronized registers, which are updated once every two ERTC_CLK cycles 1: Date/time register direct read enabled. ERTC_TIME, ERTC_DATE and ERTC_SBS values are taken from the battery powered domain.
Bit 4	RCDEN	0x0	rw	Reference clock detection enable 0: Reference clock detection disabled 1: Reference clock detection enabled
Bit 3	TSEDG	0x0	rw	Timestamp trigger edge 0: Rising edge 1: Falling edge
Bit 2: 0	Reserved	0x0	resd	Kept at its default value.

#### 17.4.4 ERTC initialization and status register (ERTC\_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	CALUPDF	0x0	ro	Calibration value update complete flag 0: Calibration value update is complete 1: Calibration value update is in progress
Bit 15: 14	Reserved	0x0	resd	This bit is automatically set when software writes to the ERTC_SCAL register. It is automatically cleared when a new calibration value is taking into account. When this bit is set, the write access to the ERTC_SCAL register is not allowed.
Bit 13	TP1F	0x0	rw0c	Tamper detection 1 flag 0: No tamper event 1: Tamper event occurs
Bit 12	TSOF	0x0	rw0c	Timestamp overflow flag 0: No timestamp overflow 1: Timestamp overflow occurs If a new time stamp event is detected when time stamp flag (TSF) is already set, this bit will be set by hardware.
Bit 11	TSF	0x0	rw0c	Timestamp flag 0: No timestamp event 1: Timestamp event occurs It is recommended to double check the TSOF flag after reading a timestamp and clearing the TSF. Otherwise, a new timestamp event may be detected while clearing the

				TSF. <i>Note: The clearing operation of this bit takes effect after two APB_CLK cycles.</i>
Bit 10: 9	Reserved	0x0	resd	Kept at its default value.
Bit 8	ALAF	0x0	rw0c	Alarm clock A flag 0: No alarm clock event 1: Alarm clock event occurs <i>Note: The clearing operation of this bit takes effect after two APB_CLK cycles.</i>
Bit 7	IMEN	0x0	rw	Initialization mode enable 0: Initialization mode disabled 1: Initialization mode enabled When an initialization mode is entered, the calendar stops running.
Bit 6	IMF	0x0	ro	Enter initialization mode flag 0: Initialization mode is not entered 1: Initialization mode is entered The ERTC_TIME, ERTC_DATE and ERTC_DIV registers can be modified only when an initialization mode is enabled (INITEN=1) and entered (INITEF=1).
Bit 5	UPDF	0x0	rw0c	Calendar update flag 0: Calendar update is in progress 1: Calendar update is complete The UPDF bit is set each time the shadow register is synchronized with the ERTC calendar value located in the battery powered domain. The synchronization is performed every two ERTC_CLK.
Bit 4	INITF	0x0	ro	Calendar initialization flag 0: Calendar has not been initialized 1: Calendar has been initialized This bit is set when the calendar year filed (ERTC_DATE) is different from 0. It is cleared when the year is 0.
Bit 3	TADJF	0x0	ro	Time adjustment flag 0: No time adjustment 1: Time adjustment is in progress This bit is automatically set when a write access to the ERTC_TADJ register is performed. It is automatically cleared at the end of time adjustment.
Bit 2: 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	ALAWF	0x1	ro	Alarm A register allows write flag 0: Alarm A register write operation not allowed 1: Alarm A register write operation allowed

#### 17.4.5 ERTC divider register (ERTC\_DIV)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22: 16	DIVA	0x7F	rw	Divider A
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14: 0	DIVB	0x00FF	rw	Divider B Calendar clock = ERTC_CLK/((DIVA+1)x(DIVB+1))

#### 17.4.6 ERTC alarm clock A register (ERTC\_ALA)

Bit	Register	Reset value	Type	Description
Bit 31	MASK4	0x0	rw	Date/week day mask 0: Date/week day is not masked 1: Alarm clock doesn't care about date/week day
Bit 30	WKSEL	0x0	rw	Date/week day select 0: Date 1: Week day (DT[1: 0] is not used)
Bit 29: 28	DT	0x0	rw	Date tens
Bit 27: 24	DU	0x0	rw	Date/week day units
Bit 23	MASK3	0x0	rw	Hour mask

				0: No hour mask 1: Alarm clock doesn't care about hours
Bit 22	AMPM	0x0	rw	AM/PM 0: AM 1: PM <i>Note: This bit is applicable for 12-hour format only. It is 0 for 24-hour format.</i>
Bit 21: 20	HT	0x0	rw	Hour tens
Bit 19: 16	HU	0x0	rw	Hour units
Bit 15	MASK2	0x0	rw	Minute mask 0: No minute mask 1: Alarm clock doesn't care about minutes
Bit 14: 12	MT	0x0	rw	Minute tens
Bit 11: 8	MU	0x0	rw	Minute units
Bit 7	MASK1	0x0	rw	Second mask 0: No second mask 1: Alarm clock doesn't care about seconds
Bit 6: 4	ST	0x0	rw	Second tens
Bit 3: 0	SU	0x0	rw	Second units

#### 17.4.7 ERTC write protection register (ERTC\_WP)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value
Bit 7: 0	CMD	0x00	wo	Command register All ERTC register write protection is unlocked by writing 0xCA and 0x53. Writing any other value will re-activate write protection.

#### 17.4.8 ERTC subsecond register (ERTC\_SBS)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 0	SBS	0x0000	ro	Sub-second value Subsecond is the value in the DIVB counter. Clock frequency = ERTC_CLK/(DIVA+1)

#### 17.4.9 ERTC time adjustment register (ERTC\_TADJ)

Bit	Register	Reset value	Type	Description
Bit 31	ADD1S	0x0	wo	Add 1 second 0: No effect 1: Add one second This bit can be written only when TADJF=0. It is intended to be used with DECSBS in order to fine-tune the time.
Bit 30: 15	Reserved	0x0000	resd	Kept at its default value
Bit 14: 0	DECSBS	0x0000	wo	DECSBS[14: 0]: Decrease sub-second value Delay (ADD1S=0): Delay = DECSBS/(DIVB+1) Advance (ADD1S=1) : Advance =1-(DECSBS/(DIVB+1))

#### 17.4.10 ERTC time stamp time register (ERTC\_TSTM)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value
Bit 22	AMPM	0x0	ro	AM/PM 0: AM 1: PM <i>Note: This bit is applicable for 12-hour format only. It is 0 for 24-hour format.</i>
Bit 21: 20	HT	0x0	ro	Hour tens
Bit 19: 16	HU	0x0	ro	Hour units

Bit 15	Reserved	0x0	resd	Kept at its default value
Bit 14: 12	MT	0x0	ro	Minute tens
Bit 11: 8	MU	0x0	ro	Minute units
Bit 7	Reserved	0x0	resd	Kept at its default value
Bit 6: 4	ST	0x0	ro	Second tens
Bit 3: 0	SU	0x0	ro	Second units

*Note: The content of this register is valid only when the TSF is set in the ERTC\_STS register. It is cleared when TSF bit is reset.*

#### 17.4.11 ERTC time stamp date register (ERTC\_TS DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 13	WK	0x0	ro	Week day
Bit 12	MT	0x0	ro	Month tens
Bit 11: 8	MU	0x0	ro	Month units
Bit 7: 6	Reserved	0x0	resd	Kept at its default value
Bit 5: 4	DT	0x0	ro	Date tens
Bit 3: 0	DU	0x0	ro	Date units

*Note: The content of this register is valid only when the TSF is set in the ERTC\_STS register. It is cleared when TSF bit is reset.*

#### 17.4.12 ERTC time stamp subsecond register (ERTC\_TSSBS)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 0	SBS	0x0000	ro	Sub-second value

*Note: The content of this register is valid only when the TSF is set in the ERTC\_STS register. It is cleared when TSF bit is reset.*

#### 17.4.13 ERTC smooth calibration register (ERTC\_SCAL)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15	ADD	0x0	rw	Add ERTC clock 0: No ERTC clock added 1: One ERTC_CLK is inserted every 211 ERTC_CLK cycles
Bit 14	CAL8	0x0	rw	8-second calibration period 0: No effect 1: 8-second calibration
Bit 13	CAL16	0x0	rw	16 second calibration period 0: No effect 1: 16-second calibration
Bit 12: 9	Reserved	0x0	resd	Kept at its default value
Bit 8: 0	DEC	0x000	rw	Decrease ERTC clock DEC out of ERTC_CLK cycles are masked during the $2^{20}$ ERTC_CLK periods. This bit is usually used with ADD. When the ADD is set, the actual number of ERTC_CLK is equal to $2^{20}+512-DEC$ during the $2^{20}$ ERTC_CLK periods.

#### 17.4.14 ERTC tamper configuration register (ERTC\_TAMP)

Bit	Register	Reset value	Type	Description
Bit 31: 19	Reserved	0x0000	resd	Kept at its default value
Bit 18	OUTTYPE	0x0	rw	Output type 0: Open-drain output

				1: Push-pull output
Bit 17: 16	Reserved	0x0	resd	Kept at its default value
Bit 15	TPPU	0x0	rw	Tamper detection pull-up 0: Tamper detection pull-up enabled 1: Tamper detection pull-up disabled
Bit 14: 13	TPPR	0x0	rw	Tamper detection pre-charge time 0: 1 ERTC_CLK cycle 1: 2 ERTC_CLK cycles 2: 4 ERTC_CLK cycles 3: 8 ERTC_CLK cycles
Bit 12: 11	TPFLT	0x0	rw	Tamper detection filter time 0: No filter 1: Tamper is detected after 2 consecutive samples 2: Tamper is detected after 4 consecutive samples 3: Tamper is detected after 8 consecutive samples
Bit 10: 8	TPFREQ	0x0	rw	Tamper detection frequency 0: ERTC_CLK/32768 1: ERTC_CLK/16384 2: ERTC_CLK/8192 3: ERTC_CLK/4096 4: ERTC_CLK/2048 5: ERTC_CLK/1024 6: ERTC_CLK/512 7: ERTC_CLK/256
Bit 7	TPTSEN	0x0	rw	Tamper detection timestamp enable 0: Tamper detection timestamp disabled 1: Tamper detection timestamp enabled. Save timestamp on a tamper event.
Bit 6: 3	Reserved	0x0	resd	Kept at its default value
Bit 2	TPIEN	0x0	rw	Tamper detection interrupt enable 0: Tamper detection interrupt disabled 1: Tamper detection interrupt enabled
Bit 1	TP1EDG	0x0	rw	Tamper detection 1 valid edge If TPFLT=0: 0: Rising edge 1: Falling edge If TPFLT>0: 0: Low 1: High
Bit 0	TP1EN	0x0	rw	Tamper detection 1 enable 0: Tamper detection 1 disabled 1: Tamper detection 1 enabled

### 17.4.15 ERTC alarm clock A subsecond register (ERTC\_ALASBS)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value
				Sub-second mask 0: No comparison. Alarm A doesn't care about subseconds. 1: SBS[0] is compared 2: SBS[1: 0] are compared 3: SBS[2: 0] are compared ... 14: SBS[13: 0] are compared 15: SBS[14: 0] are compared
Bit 27: 24	SBSMSK	0x0	rw	
Bit 23: 15	Reserved	0x000	rw	Kept at its default value
Bit 14: 0	SBS	0x0000	rw	Sub-second value

### 17.4.16 ERTC battery powered domain data register (ERTC\_BPRx)

Bit	Register	Reset value	Type	Description
Bit 31: 0	DT	0x0000 0000	rw	Battery powered domain data BPR_DTx registers are powered on by $V_{BAT}$ so that they are not reset by a system reset. They are reset on a tamper event or when a battery powered domain is reset.

# 18 Analog-to-digital converter (ADC)

## 18.1 ADC introduction

The ADC is a peripheral that converts an analog input signal into a 12-bit digital signal. Its sampling rate is as high as 2 MSPS. It has up to 18 channels for sampling and conversion.

## 18.2 ADC main features

In terms of analog:

- 12-bit configurable resolution
- Self-calibration time: 154 ADC clock cycles
- ADC conversion time
- ADC conversion time is 0.5  $\mu$ s at 28 MHz (When the system clock is 120 MHz, ADC conversion time is 0.7  $\mu$ s at 20 MHz)
- ADC power supply: Refer to the data sheet for more information
- ADC input range:  $V_{REF-} \leq V_{IN} \leq V_{REF+}$

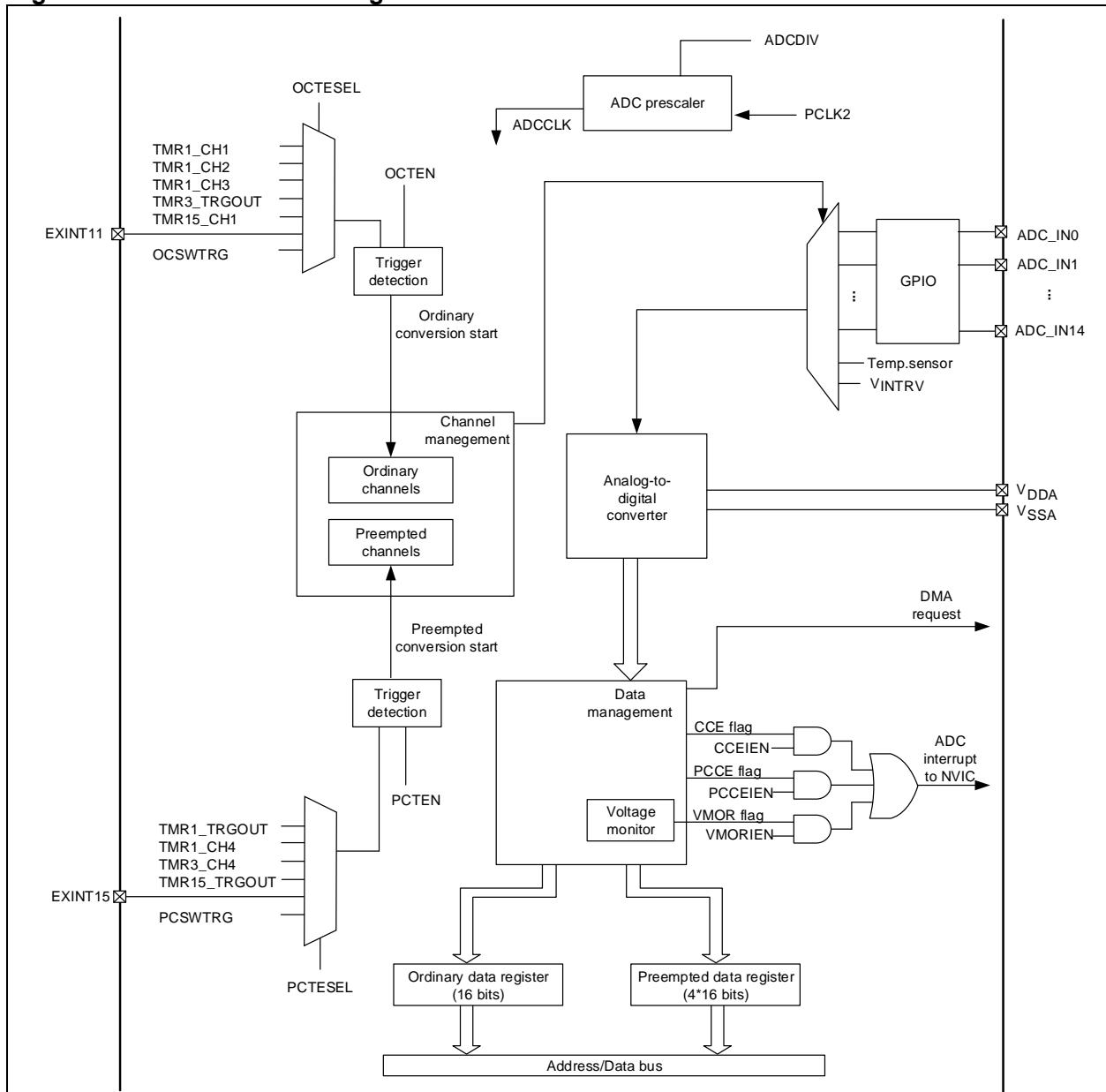
In terms of digital control:

- Regular channels and preempted channels with different priority
- Regular channels and preempted channels both have their own trigger detection circuit
- Each channel can independently define its own sampling time
- Conversion sequence management mechanism supports multiple-channel conversion
- Optional data alignment
- Programmable voltage monitor threshold
- Regular channels with DMA transfer support
- Interrupt generation at one of the following events:
  - End of the conversion of preempted channels
  - End of the conversion of regular channels
  - Voltage outside the programmed threshold

## 18.3 ADC structure

*Figure 18-1* shows the block diagram of ADC1.

Figure 18-1 ADC1 block diagram



#### Input pin description:

- $V_{DDA}$ : Analog supply, ADC analog supply
- $V_{SSA}$ : Analog supply ground, ADC analog supply ground
- $ADCx\_IN$ : Analog input signal channels

Refer to the data sheet for details on the input pins and voltage limits.

## 18.4 ADC functional overview

### 18.4.1 Channel management

#### Analog signal channel input:

There are 18 analog signal channel inputs for each of the ADCs, expressed by  $ADC\_Inx$  ( $x=0$  to  $17$ ).

- $ADC1\_IN0$  to  $ADC1\_IN14$  are referred to as the external analog input,  $ADC1\_IN15$  as  $V_{ss}$ ,  $ADC1\_IN15$  as an internal temperature sensor,  $ADC1\_IN17$  as an internal reference voltage.

#### Channel conversion

The conversions are divided into two groups: ordinary and preempted channels. The preempted group has priority over the ordinary group.

If the preempted channel trigger occurs during the ordinary channel conversion, then the ordinary

channel conversion is interrupted, giving the priority to the preempted channel, and the ordinary channel continues its conversion at the end of the preempted channel conversion. If the ordinary channel trigger occurs during the preempted channel conversion, the ordinary channel conversion won't start until the end of the preempted channel conversion.

Program the channel (ADC\_Inx) into the ordinary channel sequence (ADC\_OSQx) and the preempted channel sequence (ADC\_PSQ), and the same channel can be repeated, the total number of sequences is determined by OCLEN and PCLEN, then it is ready to enable the ordinary channel or preempted channel conversion.

#### 18.4.1.1 Internal temperature sensor

The temperature sensor is connected to ADC1\_IN16. Before the temperature sensor channel conversion, it is mandatory to enable the ITSRVEN bit in the ADC\_CTRL2 register and wait after power-on time.

The converted data of such channel, along with the voltage value at 25°C and Avg\_Slope ,can be used to calculate the temperature.

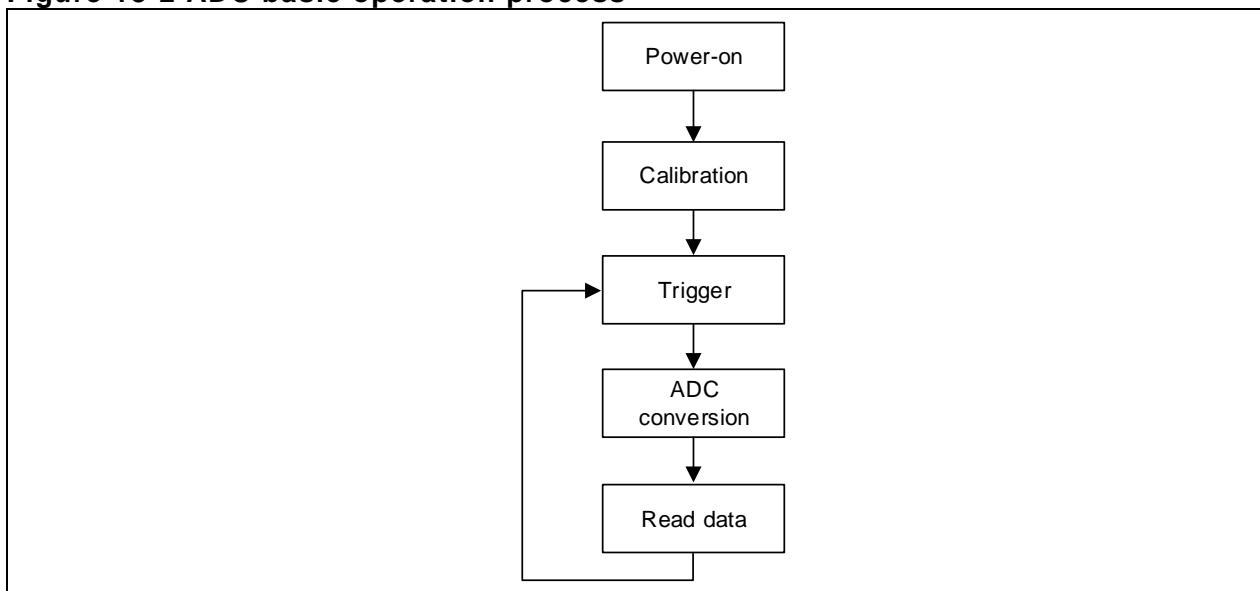
#### 18.4.1.2 Internal reference voltage

The internal reference voltage of the typical value 1.2 V is connected to ADC1\_IN17. It is mandatory to enable the ITSRVEN bit in the ADC\_CTRL2 register before the internal reference channel conversion. The converted data of such channel can be used to calculate the external reference voltage.

### 18.4.2 ADC operation process

**Figure 18-2** shows the basic operation process of the ADC. It is recommended to do the calibration after the initial power-on in order to improve the accuracy of sampling and conversion. After the calibration, trigger is used to enable ADC sampling and conversion. Read data at the end of the conversion.

**Figure 18-2 ADC basic operation process**



#### 18.4.2.1 Power-on and calibration

##### Power-on

Set the ADCxEN bit in the CRM\_APB2EN register to enable ADC clocks: PCLK2 and ADCCLK.

Program the desired ADCCLK frequency by setting the ADCDIV bit in the CRM\_CFG register. The ADCCLK is derived from PCLK2 frequency division.

*Note: ADCCLK must be less than 28 MHz.*

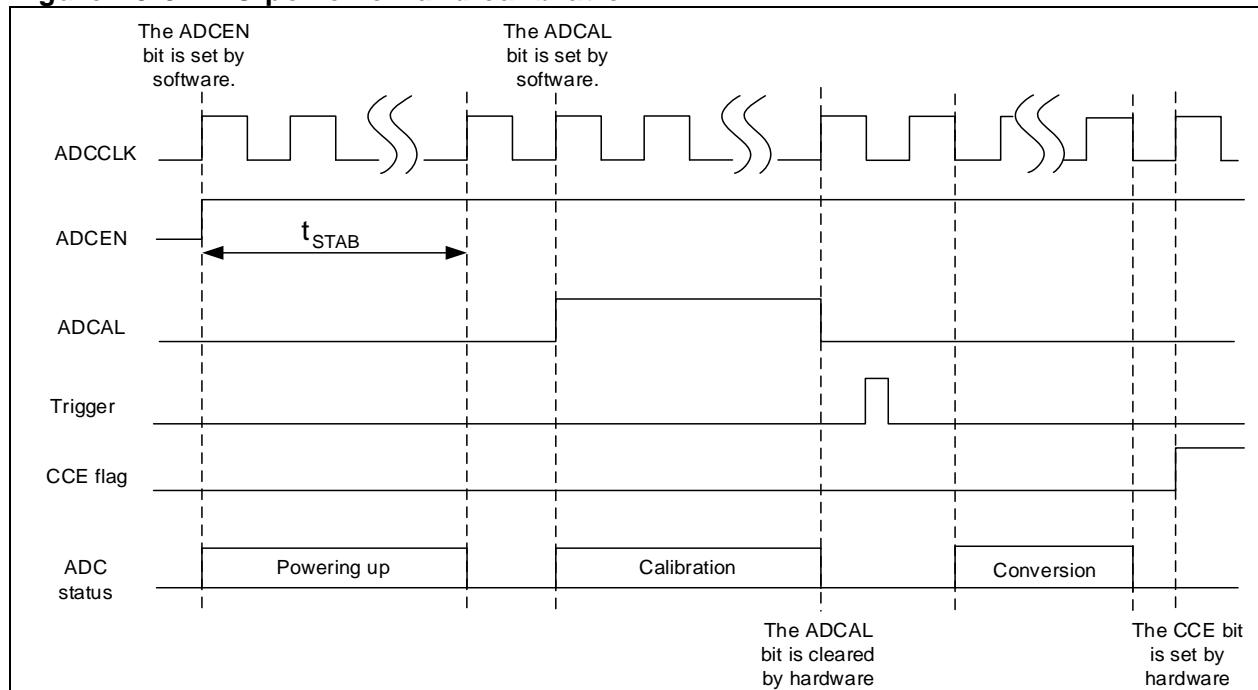
Then set the ADCEN bit in the ADC\_CTRL2 register to power the ADC, and wait until the t<sub>STAB</sub> is ready before starting ADC conversion. Clear the ADCEN bit will halt the ADC conversion and result in a reset. In the meantime, the ADC can be switched off to save power.

## Calibration

After power-on, enable ADC calibration by setting the ADCAL bit in the ADC\_CTRL2 register. When the calibration is complete, the ADCAL bit is cleared by hardware and the conversion is started by software trigger.

After each calibration, the calibration value is stored in ADC\_ODT register, and then value is automatically sent back to the ADC so as to eliminate capacitance errors. The storage of the calibration value will not set the OCCE flag, or generate interrupts or DMA requests.

**Figure 18-3 ADC power-on and calibration**



### 18.4.2.2 Trigger

The ADC triggers contain ordinary channel trigger and preempted channel trigger. The ordinary channel conversion is triggered by ordinary channel triggers while the preempted channel conversion is triggered by preempted ones. After the OCTEN or PCTEN bit is set in the ADC\_CTRL2 register, the ADC starts conversion after a trigger source is detected.

The conversion can be triggered by software writing to the OCSWTRG and PCSWTRG bits in the ADC\_CTRL2 register, or by external events. The external events include timer and pin triggers. The OCTESEL and PCTESEL bits in the ADC\_CTRL2 register are used to select specific trigger sources, as shown in **Table 18-1**.

Besides, the ordinary channel has a special trigger source, which repeatedly enables ADCEN to trigger conversion, without the need of enabling the OCTEN bit in the ADC\_CTRL2 register.

**Table 18-1 Trigger sources for ADC1**

OCTESEL	Trigger source	PCTESEL	Trigger source
000	TMR1_CH1 event	000	TMR1_TRGOUT event
001	TMR1_CH2 event	001	TMR1_CH4 event
010	TMR1_CH3 event	010	Reserved
011	Reserved	011	Reserved
100	TMR3_TRGOUT event	100	TMR3_CH4 event
101	TMR15_CH1 event	101	TMR15_TRGOUT event
110	EXINT line11 external pin	110	EXINT line15 external pin
111	OCSWTRG	111	PCSWTRG

### 18.4.2.3 Sampling and conversion sequence

The sampling period can be configured by setting the CSPTx bit in the ADC\_SPT1 and ADC\_SPT2 registers. A single one conversion time is calculated with the following formula:

$$\text{A single one conversion time (ADCCLK cycle)} = \text{sampling time} + 12.5$$

Example:

If the CSPTx selects 1.5 cycles, then one conversion needs  $1.5+12.5=14$  ADCCLK cycles

If the CSPTx selects 7.5 cycles, then one conversion needs  $7.5+12.5=20$  ADCCLK cycles.

### 18.4.3 Conversion sequence management

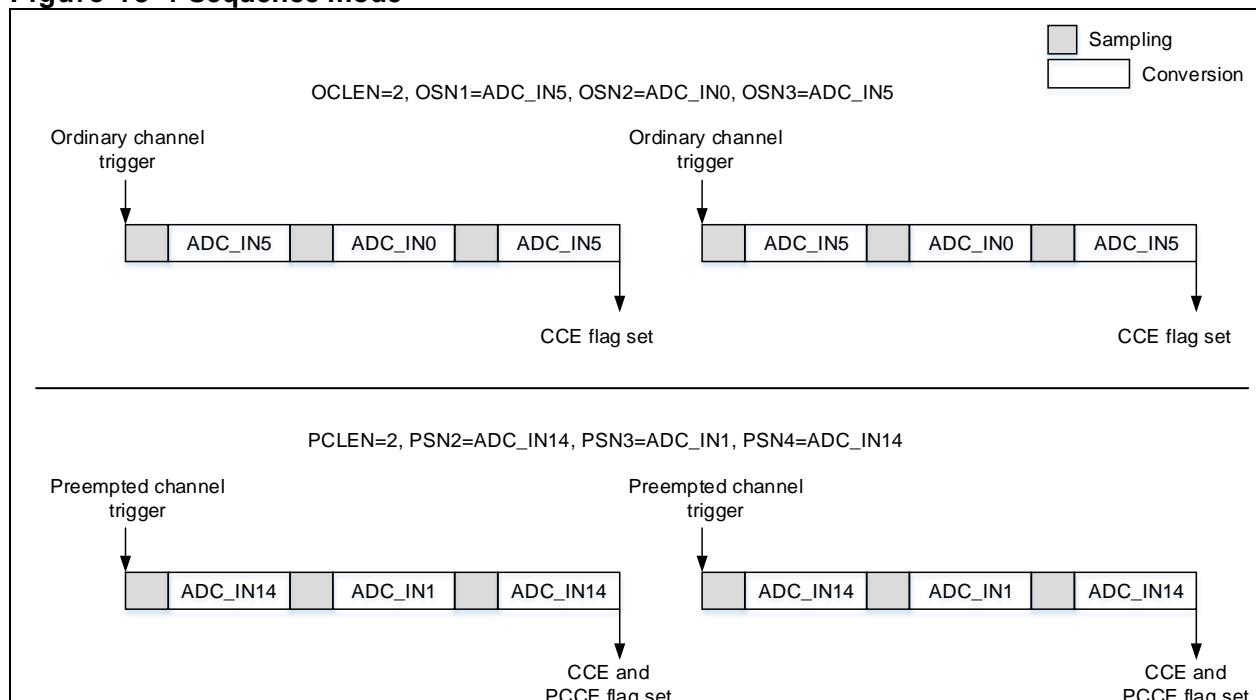
Only one channel is converted at each trigger event by default, that is, either OSN1-defined channel (ordinary trigger) or PSN4-defined channel (preempted trigger).

The detailed conversion sequence modes are described in the following sections. With this, the channels can be converted in a specific order.

#### 18.4.3.1 Sequence mode

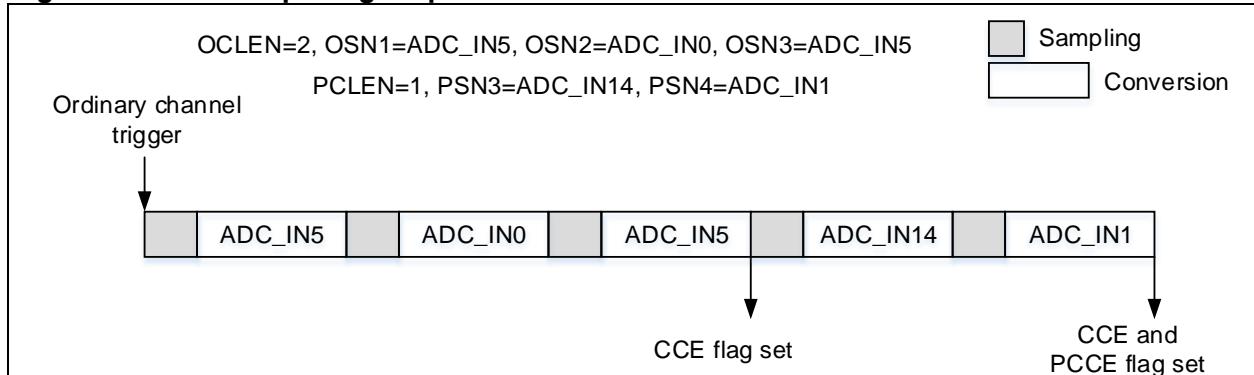
The sequence mode is enabled by setting the SQEN bit in the ADC\_CTRL1 register. The ADC\_OSQx register is used to configure the sequence and the total number of the ordinary channels while the ADC\_PSQ register is used to define the sequence and total number of the preempted channels. After the sequence mode is enabled, a single trigger event enables the conversion of a group of channels in order. The ordinary channels start converting from the QSN1 while the preempted channels starts from the PSNx, where x=4-PCLEN. Figure 18-4 shows an example of the behavior in sequence mode.

**Figure 18-4 Sequence mode**



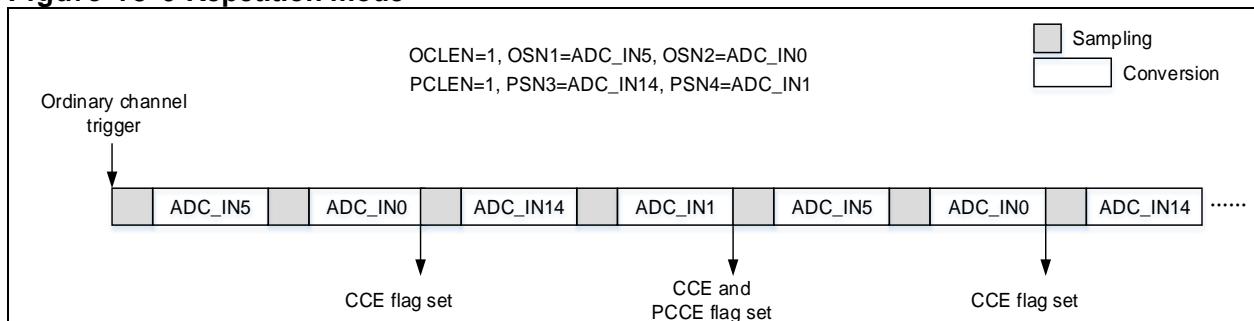
#### 18.4.3.2 Automatic preempted group conversion mode

The automatic preempted group conversion mode is enabled by setting the PCAUTOEN bit in the ADC\_CTRL1 register. Once the ordinary channel conversion is complete, the preempted group will automatically continue its conversion. This mode can work with the sequence mode. The preempted group conversion starts automatically at the end of the conversion of the ordinary group. [Figure 18-5](#) shows an example of the behavior when the automatic preempted group conversion mode works with the ordinary group.

**Figure 18-5 Preempted group auto conversion mode**

### 18.4.3.3 Repetition mode

The repetition mode is enabled by setting the RPEN bit in the ADC\_CTRL2 register. When a trigger signal is detected, the ordinary channels will be converted repeatedly. This mode can work with the ordinary group conversion in the sequence mode so as to enable the repeated conversion of the ordinary group. Such mode can also work with the preempted group auto conversion mode so as to repeatedly convert the ordinary group and preempted group in sequence. Figure 18-6 shows an example of the behavior when the repetition mode works with the sequence mode and preempted group auto conversion mode.

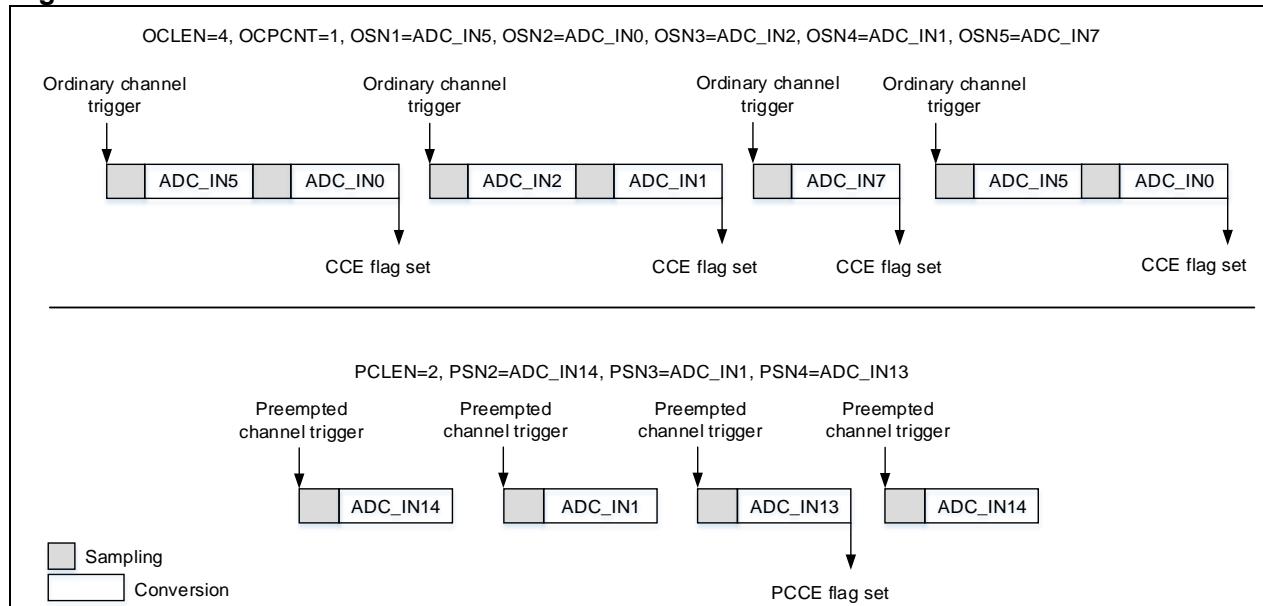
**Figure 18-6 Repetition mode**

### 18.4.3.4 Partition mode

The partition mode of the ordinary group can be enabled by setting the OCPEN bit in the ADC\_CTRL1 register. In this mode, the ordinary group conversion sequence length (OCLEN bit in the ADC\_OSQ1 register) is divided into a smaller sub-group, in which the number of the channels is programmed with the OCPCNT bit in the ADC\_CTRL1 register. A single trigger event will enable the conversion of all the channels in the sub-group. Each trigger event selects different sub-group in order.

Set the PCPEN bit in the ADC\_CTRL1 register will enable the partition mode of the preempted group. In this mode, the conversion sequence length (PCLEN bit in the ADC\_PSQ1 register) is divided into a sub-group with only one channel. A single one trigger event will enable the conversion of the channel in the sub-group. Each trigger event selects different sub-group in order.

The partition mode cannot be used with the repetition mode at the same time. [Figure 18-7](#) shows an example of the behavior in partition mode for ordinary group and preempted group.

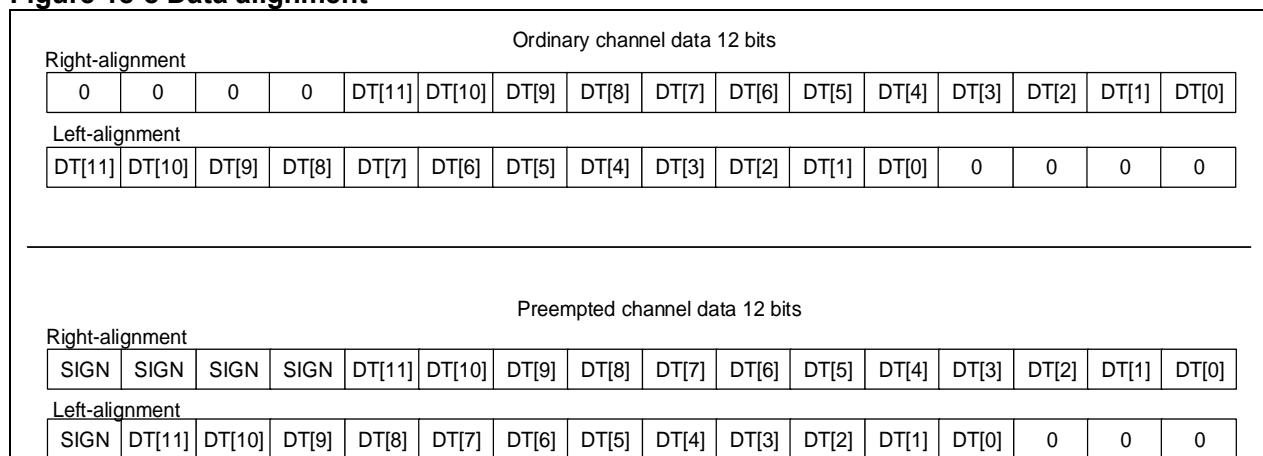
**Figure 18-7 Partition mode**

## 18.4.4 Data management

At the end of the conversion of the ordinary group, the converted value is stored in the ADC\_ODT register. Once the preempted group conversion ends, the converted data of the preempted group is stored in the ADC\_PDTx register.

### 18.4.4.1 Data alignment

DTALIGN bit in the ADC\_CTRL2 register selects the alignment of data (right-aligned or left-aligned). In addition, the converted data of the preempted group is decreased by the offset written in the ADC\_PCDTOx register. Thus the result may be a negative value, marked by SIGN, as shown in [Figure 18-8](#).

**Figure 18-8 Data alignment**

### 18.4.4.2 Data read

Read access to the ADC\_ODT register using CPU or DMA gets the converted data of the ordinary group. Read access to the ADC\_PDTx register using CPU only gets the converted data of the preempted group. When the OCDMAEN bit is set in the ADC\_CTRL2 register, the ADC will issue a DMA request each time the ADC\_ODT register is updated.

## 18.4.5 Voltage monitoring

The OCMEN bit or PCVMEN bit in the ADC\_CTRL1 register is used to enable voltage monitoring based on the converted data.

The VMOR bit will be set if the converted result is outside the high threshold (ADC\_VMHB register) or less than the low threshold (ADC\_VMLB register).

The VMSGGEN bit in the ADC\_CTRL1 register is used to enable voltage monitor on either a single channel or all the channels. The VMCSEL bit is used to select a specific channel that requires voltage monitoring.

Voltage monitoring is based on the comparison result between the original converted data and the 12-bit voltage monitor boundary register, irrespective of the PCDTOx and DTALIGN bits.

## 18.4.6 Status flag and interrupts

Each of the ADCs has its dedicated ADCx\_STS registers, that is, OCCS (ordinary channel conversion start flag), PCCS (preempted channel conversion start flag), PCCE (preempted channel conversion end flag), OCCE (ordinary channel conversion end flag) and VMOR (voltage monitor out of range).

PCCE, CCE and VMOR have their respective interrupt enable bits. Once the interrupt bits are enabled, the corresponding flag is set and an interrupt is sent to CPU.

## 18.5 ADC registers

**Table 18-2** lists ADC register map and their reset values.

These peripheral registers must be accessed by word (32 bits).

**Table 18-2 ADC register map and reset values**

Register name	Offset	Reset value
ADC_STS	0x000	0x0000 0000
ADC_CTRL1	0x004	0x0000 0000
ADC_CTRL2	0x008	0x0000 0000
ADC_SPT1	0x00C	0x0000 0000
ADC_SPT2	0x010	0x0000 0000
ADC_PCDTO1	0x014	0x0000 0000
ADC_PCDTO2	0x018	0x0000 0000
ADC_PCDTO3	0x01C	0x0000 0000
ADC_PCDTO4	0x020	0x0000 0000
ADC_VMHB	0x024	0x0000 0FFF
ADC_VMLB	0x028	0x0000 0000
ADC_OSQ1	0x02C	0x0000 0000
ADC_OSQ2	0x030	0x0000 0000
ADC_OSQ3	0x034	0x0000 0000
ADC_PSQ	0x038	0x0000 0000
ADC_PDT1	0x03C	0x0000 0000
ADC_PDT2	0x040	0x0000 0000
ADC_PDT3	0x044	0x0000 0000
ADC_PDT4	0x048	0x0000 0000
ADC_ODT	0x04C	0x0000 0000

### 18.5.1 ADC status register (ADC\_STS)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 5	Reserved	0x0000000	resd	Kept at its default value.
Bit 4	OCCS	0x0	rw0c	Ordinary channel conversion start flag This bit is set by hardware and cleared by software (writing 0). 0: No ordinary channel conversion started 1: Ordinary channel conversion has started
Bit 3	PCCS	0x0	rw0c	Preempted channel conversion start flag This bit is set by hardware and cleared by software (writing 0). 0: No preempted channel conversion started 1: Preempted channel conversion has started
Bit 2	PCCE	0x0	rw0c	Preempted channel end of conversion flag This bit is set by hardware and cleared by software (writing 0). 0: Conversion is not complete 1: Conversion is complete
Bit 1	OCCE	0x0	rw0c	End of conversion flag This bit is set by hardware. It is cleared by software (writing 0) or by reading the ADC_ODT register. 0: Conversion is not complete 1: Conversion is complete Note: This bit is set at the end of the ordinary or preempted group.
Bit 0	VMOR	0x0	rw0c	Voltage monitoring out of range flag This bit is set by hardware and cleared by software (writing 0). 0: Voltage is within the value programmed 1: Voltage is outside the value programmed

### 18.5.2 ADC control register1 (ADC\_CTRL1)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23	OCVMEN	0x0	rw	Voltage monitoring enable on ordinary channels 0: Voltage monitoring disabled on ordinary channels 1: Voltage monitoring enabled on ordinary channels
Bit 22	PCVMEN	0x0	rw	Voltage monitoring enable on preempted channels 0: Voltage monitoring disabled on preempted channels 1: Voltage monitoring enabled on preempted channels
Bit 21: 16	Reserved	0x0	resd	Kept at its default value.
Bit 15: 13	OCPCNT	0x0	rw	Partitioned mode conversion count of ordinary channels 000: 1 channel 001: 2 channels ..... 111: 8 channels Note: In this mode, the preempted group converts only one channel at each trigger.
Bit 12	PCPEN	0x0	rw	Partitioned mode enable on preempted channels 0: Partitioned mode disabled on preempted channels 1: Partitioned mode enabled on preempted channels
Bit 11	OCPEN	0x0	rw	Partitioned mode enable on ordinary channels This is set and cleared by software to enable or disable

				partitioned mode on ordinary channels. 0: Partitioned mode disabled on ordinary channels 1: Partitioned mode enabled on ordinary channels
Bit 10	PCAUTOEN	0x0	rw	Preempted group automatic conversion enable after ordinary group 0: Preempted group automatic conversion disabled 1: Preempted group automatic conversion enabled
Bit 9	VMSGGEN	0x0	rw	Voltage monitoring enable on a single channel 0: Disabled (Voltage monitoring enabled on all channels) 1: Enabled (Voltage monitoring enabled a single channel)
Bit 8	SQEN	0x0	rw	Sequence mode enable 0: Sequence mode disabled (a single channel is converted) 1: Sequence mode enabled (the selected multiple channels are converted) Note: If this mode is enabled and the CCEIEN/PCCEIEN is set, a CCE or PCCE interrupt is generated only at the end of conversion of the last channel.
Bit 7	PCCEIEN	0x0	rw	Conversion end interrupt enable on Preempted channels 0: Conversion end interrupt disabled on Preempted channels 1: Conversion end interrupt enabled on Preempted channels
Bit 6	VMORIEN	0x0	rw	Voltage monitoring out of range interrupt enable 0: Voltage monitoring out of range interrupt disabled 1: Voltage monitoring out of range interrupt enabled
Bit 5	CCEIEN	0x0	rw	Channel conversion end interrupt enable 0: Channel conversion end interrupt disabled 1: Channel conversion end interrupt enabled
Bit 4: 0	VMCSEL	0x00	rw	Voltage monitoring channel select This field is valid only when the VMSGGEN is enabled. 00000: ADC_IN0 channel 00001: ADC_IN1 channel ..... 01111: ADC_IN15 channel 10000: ADC_IN16 channel 10001: ADC_IN17 channel 10010~11111: Unused, configuration is not allowed.

### 18.5.3 ADC control register2 (ADC\_CTRL2)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 30: 24	Reserved	0x00	resd	Kept at its default value
Bit 23	ITSRVEN	0x0	rw	Internal V <sub>INTRV</sub> enable 0: Internal V <sub>INTRV</sub> disabled 1: Internal V <sub>INTRV</sub> enabled
Bit 22	OCSWTRG	0x0	rw	Conversion of ordinary channels triggered by software 0: Conversion of ordinary channels not triggered 1: Conversion of ordinary channels triggered (This bit is cleared by software or by hardware as soon as the conversion starts)
Bit 21	PCSWTRG	0x0	rw	Conversion of preempted channels triggered by software 0: Conversion of preempted channels not triggered 1: Conversion of preempted channels triggered (This bit is cleared by software or by hardware as soon as the conversion starts)
Bit 20	OCTEN	0x0	rw	Trigger mode enable for ordinary channel conversion 0: Disabled

				1: Enabled
				Trigger event select for ordinary channels conversion 000: Timer 1 CH1 event 001: Timer 1 CH2 event 010: Timer 1 CH3 event 011: Unused. Do not configure. 100: Timer 3 TRGOUT event 101: Timer 15 CH1 event 111: OCSWTRG
Bit 19: 17	OCTESEL	0x0	rw	Kept at its default value
Bit 16	Reserved	0x0	resd	Trigger mode enable for preempted channels conversion
Bit 15	PCTEN	0x0	rw	0: Disabled 1: Enabled
				Trigger event select for preempted channels conversion For ADC1, the trigger events are configured as follows: 000: Timer 1 TRGOUT event 001: Timer 1 CH4 event 010: Unused. Do not configure. 011: Unused. Do not configure. 100: Timer 3 CH4 event 101: Timer 15 TRGOUT event 110: EXINT line 15 111: PCSWTRG
Bit 14: 12	PCTESEL	0x0	rw	Data alignment 0: Right alignment 1: Left alignment
Bit 11	DTALIGN	0x0	rw	Kept at its default value
Bit 10: 9	Reserved	0x0	resd	DMA transfer enable of ordinary channels
Bit 8	OCDMAEN	0x0	rw	0: Disabled 1: Enabled
Bit 7: 4	Reserved	0x0	resd	Kept at its default value.
Bit 3	ADCALINIT	0x0	rw	Initialize A/D calibration This bit is set by software and cleared by hardware. It is cleared after the calibration registers are initialized. 0: No initialization occurred or initialization completed 1: Enable initialization or initializations is ongoing
Bit 2	ADCAL	0x0	rw	A/D Calibration 0: No calibration occurred or calibration completed 1: Enable calibration or calibration is in process
Bit 1	RPEN	0x0	rw	Repetition mode enable 0: Repetition mode disabled When SQEN=0, a single conversion is done each time when a trigger event arrives; when SQEN=1, a group of conversion is done each timer when a trigger event arrives. 1: Repetition mode enabled When SQEN =0, continuous conversion mode on a single channel is enabled at each trigger event; when SQEN =1, continuous conversion mode on a group of channels is enabled at each trigger event.
Bit 0	ADCEN	0x0	rw	A/D converter enable 0: A/D converter disabled (ADC goes to power-down mode) 1: A/D converter enabled

**Note:**

When this bit is in OFF state, write a start command can wake up The ADC from power-down mode.

When this bit is in ON state, write a start command repeatedly while the other bits of the register remain unchanged will start a regular group conversion.

The application should pay attention to the fact that there is a delay of  $t_{STAB}$  between power on and start of conversion.

### 18.5.4 ADC sampling time register 1 (ADC\_SPT1)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23: 21	CSPT17	0x0	rw	Sample time selection of channel ADC_IN17 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 20: 18	CSPT16	0x0	rw	Sample time selection of channel ADC_IN16 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 17: 15	CSPT15	0x0	rw	Sample time selection of channel ADC_IN15 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 14: 12	CSPT14-	0x0	rw	Sample time selection of channel ADC_IN14 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 11: 9	CSPT13	0x0	rw	Sample time selection of channel ADC_IN13 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles

				011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 8: 6	CSPT12	0x0	rw	Sample time selection of channel ADC_IN12 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 5: 3	CSPT11	0x0	rw	Sample time selection of channel ADC_IN11 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 2: 0	CSPT10	0x0	rw	Sample time selection of channel ADC_IN10 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles

### 18.5.5 ADC sampling time register 2 (ADC\_SPT2)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value.
Bit 29: 27	CSPT9	0x0	rw	Sample time selection of channel ADC_IN9 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 26: 24	CSPT8	0x0	rw	Sample time selection of channel ADC_IN8 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles

				110: 71.5 cycles 111: 239.5 cycles
				Sample time selection of channel ADC_IN7 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 23: 21	CSPT7	0x0	rw	Sample time selection of channel ADC_IN6 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 20: 18	CSPT6	0x0	rw	Sample time selection of channel ADC_IN5 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 17: 15	CSPT5	0x0	rw	Sample time selection of channel ADC_IN4 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 14: 12	CSPT4	0x0	rw	Sample time selection of channel ADC_IN3 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 11: 9	CSPT3	0x0	rw	Sample time selection of channel ADC_IN2 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles
Bit 8: 6	CSPT2	0x0	rw	

				101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
				Sample time selection of channel ADC_IN1 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles
Bit 5: 3	CSPT1	0x0	rw	011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
				Sample time selection of channel ADC_IN0 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles
Bit 2: 0	CSPT0	0x0	rw	011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles

### 18.5.6 ADC preempted channel data offset register x (ADC\_PCDTOx) (x=1..4)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value
Bit 11: 0	PCDTOx	0x000	rw	Data offset for Preempted channel x Converted data stored in the ADC_PDTx = Raw converted data – ADC_PCDTOx

### 18.5.7 ADC voltage monitor high threshold register (ADC\_VWHB)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value
Bit 11: 0	VMHB	0xFFFF	rw	Voltage monitoring high boundary

### 18.5.8 ADC voltage monitor low threshold register (ADC\_VWLB)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value
Bit 11: 0	VMLB	0x000	rw	Voltage monitoring low boundary

### 18.5.9 ADC ordinary sequence register 1 (ADC\_OSQ1)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value
				Ordinary conversion sequence length 0000: 1 conversion
Bit 23: 20	OCLEN	0x0	rw	0001: 2 conversions ..... 1111: 16 conversions
Bit 19: 15	OSN16	0x00	rw	Number of 16th conversion in ordinary sequence
Bit 14: 10	OSN15	0x00	rw	Number of 15th conversion in ordinary sequence
Bit 9: 5	OSN14	0x00	rw	Number of 14th conversion in ordinary sequence
				Number of 13th conversion in ordinary sequence
Bit 4: 0	OSN13	0x00	rw	Note: The number can be from 0 to 17. For example, if the number is set to 3, it means that the 13 <sup>th</sup> conversion is ADC_IN3 channel.

### 18.5.10 ADC ordinary sequence register 2 (ADC\_OSQ2)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
Bit 29: 25	OSN12	0x00	rw	Number of 12th conversion in ordinary sequence
Bit 24: 20	OSN11	0x00	rw	Number of 11th conversion in ordinary sequence
Bit 19: 15	OSN10	0x00	rw	Number of 10th conversion in ordinary sequence
Bit 14: 10	OSN9	0x00	rw	Number of 9th conversion in ordinary sequence
Bit 9: 5	OSN8	0x00	rw	Number of 8th conversion in ordinary sequence
				Number of 7th conversion in ordinary sequence
Bit 4: 0	OSN7	0x00	rw	Note: The number can be from 0 to 17. For example, if the number is set to 8, it means that the 7 <sup>th</sup> conversion is ADC_IN8 channel.

### 18.5.11 ADC ordinary sequence register 3 (ADC\_OSQ3)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
Bit 29: 25	OSN6	0x00	rw	Number of 6th conversion in ordinary sequence
Bit 24: 20	OSN5	0x00	rw	Number of 5th conversion in ordinary sequence
Bit 19: 15	OSN4	0x00	rw	Number of 4th conversion in ordinary sequence
Bit 14: 10	OSN3	0x00	rw	Number of 3rd conversion in ordinary sequence
Bit 9: 5	OSN2	0x00	rw	Number of 2nd conversion in ordinary sequence
				Number of 1st conversion in ordinary sequence
Bit 4: 0	OSN1	0x00	rw	Note: The number can be from 0 to 17. For example, if the number is set to 8, it means that the 1st conversion is ADC_IN17 channel.

## 18.5.12 ADC preempted sequence register (ADC\_PSQ)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
				Preempted conversion sequence length 00: 1 conversion 01: 2 conversions 10: 3 conversions 11: 4 conversions
Bit 21: 20	PCLEN	0x0	rw	
Bit 19: 15	PSN4	0x00	rw	Number of 4th conversion in preempted sequence
Bit 14: 10	PSN3	0x00	rw	Number of 3rd conversion in preempted sequence
Bit 9: 5	PSN2	0x00	rw	Number of 2nd conversion in preempted sequence
				Number of 1st conversion in preempted sequence Note: The number can be from 0 to 17. For example, if the number is set to 3, it refers to the ADC_IN3 channel.
Bit 4: 0	PSN1	0x00	rw	If PCLEN is less than 4, the conversion sequence starts from 4-PCLEN. For example, when ADC_PSQ ([21: 0]) =10 00110 00101 00100 00011, it indicates that the scan conversion follows the sequence: 4, 5, 6, not 3, 4,5.

## 18.5.13 ADC preempted data register x (ADC\_PDTx) (x=1..4)

Accessible by words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 0	PDTx	0x0000	rw	Conversion data from preempted channel

## 18.5.14 ADC ordinary data register (ADC\_ODT)

Accessible by words.

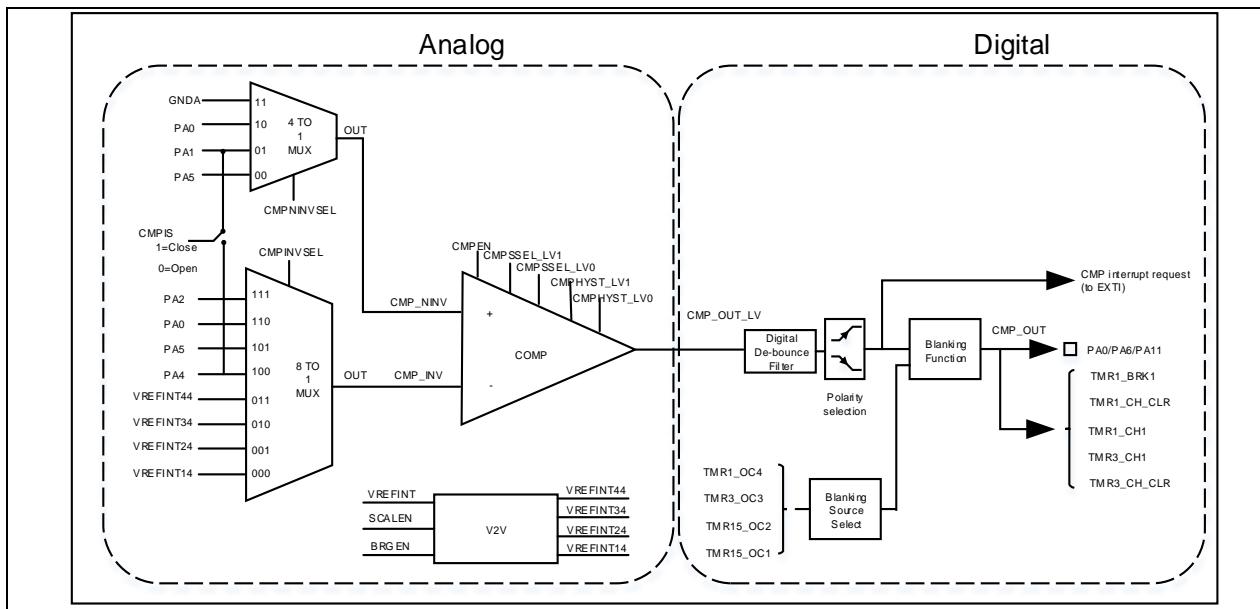
Bit	Register	Reset value	Type	Description
Bit 31: 16	ADC2ODT	0x0000	ro	ADC2 conversion data of ordinary channel Note: These bits are reserved in ADC2 and ADC3. In ADC1, these bits are valid only in master/slave mode, and they contain the conversion result from the ADC2 ordinary channels.
Bit 15: 0	ODT	0x0000	ro	Conversion data of ordinary channel

# 19 Comparator (COMP)

## 19.1 COMP introduction

AT32F421 embeds an ultra-low-power comparator (CMP). They can be used for various purposes, such as, external analog signal monitor/control and wakeup from low-power mode, and working with other timers for pulse width measurement and PWM signal control.

**Figure 19-1 Block Diagram of Comparator**



## 19.2 Main features

- Programmable hysteresis level
- Timer output as comparator blanking source
- Programmable output polarity
- Programmable output speed
- Selectable positive/negative input sources
  - I/O pins
  - Internal reference voltage and three divider values (1/4, 1/2, 3/4)
- Output redirection
  - General-purpose I/O
  - Timer brake input TMR<sub>x</sub>\_BRK
  - Timer input capture TMR\_CH
  - Timer output compare reference value clear TMR\_CH\_CLR
- Wakeup device from low-power mode through EXINT interrupts

## 19.3 Interrupt management

The output of the analog comparator, which is filtered by a digital filter and whose polarity is selected, is input to the EXINT\_21 line to generate an external interrupt or event, which can be used to wake up the system from low-power mode.

For more detailed information, please refer to the section of interrupts and events.

## 19.4 Design tips

The following information can be used for design reference:

- Input/Output configuration

As a comparator input, the I/Os must be configured as an analog mode. The comparator output can be remapped onto external I/Os.

Comparator output configuration:

Multiplexed	GPIOA_MUXL[3:0]=0111	GPIOA_MUXL[27:24]=0111	GPIOA_MUXH[15:12]=0111
Output port	PA0	PA6	PA11

- Locking

The CMP\_CTRLSTS register can be write-protected. After the completion of programming, the corresponding bits in the CMP\_CTRLSTS register can be read-only by setting CMPWP=1, including the CMPWP bit. The write protection can be unlocked only after a system reset. This feature can be used for the applications with specific security requirements.

- Low-power mode

The comparator clock enable control register is shared with the SCFG register. The clock source is PCLK, and uses system reset as its reset signal. The comparator still works in Deepsleep mode, which can be used as an EXINT interrupt source to wakeup device from low-power mode. However, prior to Deepsleep mode entry, it is necessary to disable the digital filtering of the comparator (setting GFE=0 of the G\_FILTER\_EN register).

## 19.5 Functional overview

### 19.5.1 Analog comparator

#### Positive/Negative input selection

Select an I/O or VSSA as a positive input source through the CMPxNINVSEL[1: 0] bit in the CMP\_CTRLSTS register; Select an internal reference voltage, three voltage divider values or an I/O as a negative input source through the CMPxINVSEL[2: 0] bit. The SCALEN bit, along with the BRGEN bit, is used to enable voltage divider values.

#### Hysteresis

The hysteresis feature can be selected through the CMPxHYST[1: 0] bit in the CMP\_CTRLSTS register. This is used to avoid fake signal caused by noise. Hysteresis can be disabled if not needed.

#### Operating mode

The controller can operate in high speed/maximum power consumption, medium speed/medium power consumption, low speed/low power consumption or ultra-low speed/ultra-lower power consumption in order to achieve the best trade-off between performance and power consumption. The operating mode is selected through the CMPxSSEL bit in the CMP\_CTRLSTS register.

#### Output blanking

The CMPBLANKING[2: 0] of the CMP\_CTRLSTS register is used to select the sources of the comparator blanking window. This feature can be used to prevent the generation of peak current at the start of the PVM.

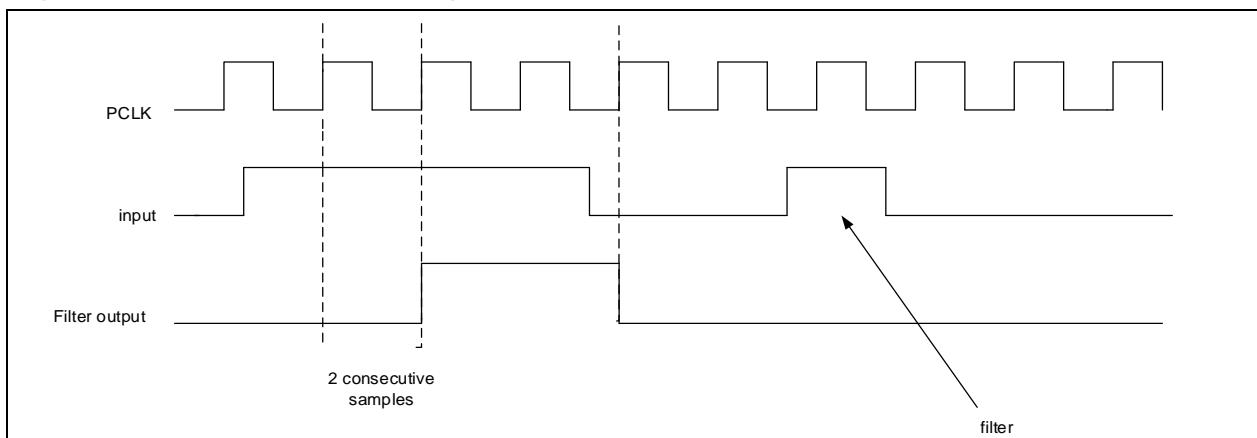
## 19.5.2 Glitch filter

The interference filter can be used to filter glitches and noise.

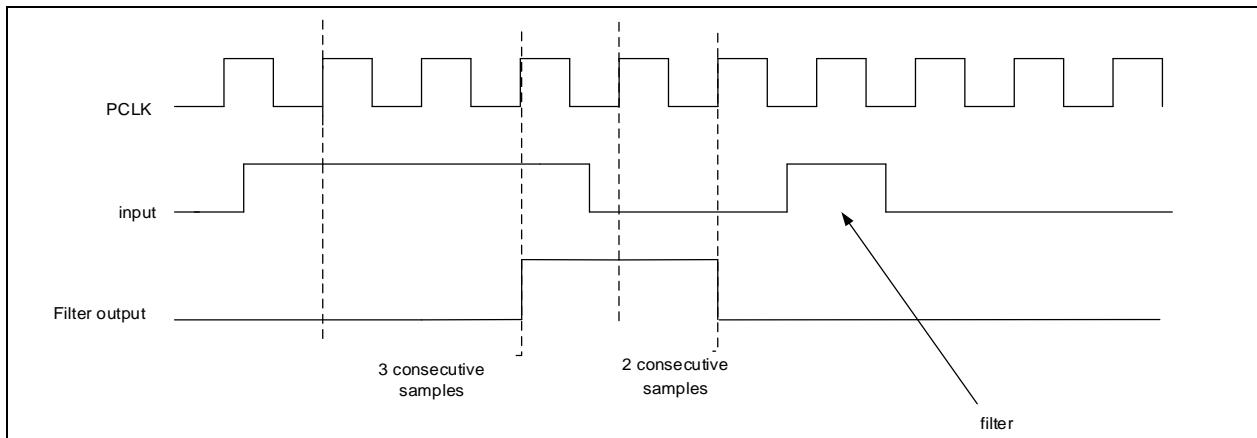
The sensitivity of the filter is controlled by the H\_PULSE\_CNT and L\_PULSE\_CNT bits. The sensitivity of the filter affects the number of the same consecutive sampling. The level change of a certain signal would not be regarded as valid before the consecutive sampling is detected on the filter input.

**Figure 19-2** and **Figure 19-3** shows the timing diagram when the H\_PULSE\_CNT and L\_PULSE\_CNT are with different values.

**Figure 19-2 Glitch filter timing when H\_PULSE\_CNT=1 and L\_PULSE\_CNT =0**



**Figure 19-3 Glitch filter timing when H\_PULSE\_CNT=2 and L\_PULSE\_CNT =1**



## 19.6 CMP registers

These registers must be accessed by words (32 bits).

**Table 19-1 CMP register map and reset values**

Register name	Offset	Reset value
CMP_CTRLSTS1	0x00	0x0000 0080
CMP_CTRLSTS2	0x04	0x0001 0001

## 19.6.1 Comparator control and status register 1 (COMP\_CTRLSTS)

Bit	Register	Reset value	Type	Description
Bit 31	CMPWP	0x0	rw0c	Comparator write protected This bit can be written only once. It is set by software and cleared by system reset. It will latch all the contents in the COMP_CTRLSTS[31:0] register. 0: CMP_CTRLSTS[31:0] can be read/written 1: CMP_CTRLSTS[31:0] is read-only.
Bit 30	CMPVALUE	0x0	ro	Comparator output value This bit is read-only.
Bit 29: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23	SCALEN	0x0	rw	Scale voltage enable 0: Scale voltage disabled (VREFINT44 = VREFINT34 = VREFINT24 = VREFINT14 = 0V) 1: Scale voltage enabled (CMPEN = 1, CMPINVSEL[2] = 0)
Bit 22	BRGEN	0x0	rw	Voltage bridge enable 0: Voltage bridge disabled, VREFINT44 = VREFINT34 = VREFINT24 = VREFINT14 = 1.2V (SCALEN = 1, CMPEN = 1, CMPINVSEL[2] = 0) 1: Voltage bridge enabled, VREFINT44 = 1.2V, VREFINT34 = 0.9V, VREFINT24 = 0.6V, VREFINT14 = 0.3V (SCALEN = 1, CMPEN = 1, CMPINVSEL[2] = 0)
Bit 21	Reserved	0x0	resd	Kept at its default value.
Bit 20:18	CMPBLANKING	0x0	rw	Comparator blanking source 000: No blanking output 001: TMR1 OC4 as a blanking window source 010: Reserved 011: TMR3 OC3 as a blanking window source 100: TMR15 OC2 as a blanking window source 101: Reserved 110: TMR15 OC1 as a blanking window source 111: Reserved
Bit 17:16	CMPHYST	0x0	rw	Comparator hysteresis 00: No hysteresis 01: Low hysteresis 10: Medium hysteresis 11: High hysteresis Refer to the electrical characteristics of hysteresis for details.
Bit 15	CMPP	0x00	rw	Comparator polarity 0: Comparator output value is not inverted 1: Comparator output value is inverted
Bit 14: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12: 10	CMPTAG	0x0	rw	Comparator output target This field defines the COMP output target. 000: No selection 001: Timer 1 brake input 010: Timer 1 input capture 1 011: Timer 1 output compare clear 100: Reserved 101: Reserved 110: Timer 3 input capture 1 111: Timer 3 output compare clear
Bit 9	Reserved	0x0	resd	Kept at its default value.
Bit 8: 7	CMPINVSEL	0x1	rw	Comparator non-inverting selection 00: PA5 01: PA1 (by default) 10: PA0 11: VSSA
Bit 6:4	CMPINVSEL	0x0	rw	Comparator inverting selection 000: 1/4 VREFINT 001: 1/2 VREFINT 010: 3/4 VREFINT 011: VREFINT

				100: PA4 101: PA5 110: PA0 111: PA2
Bit 3: 2	CMPSEL	0x0	rw	Comparator speed selection This bit is used to control the operating mode of the comparator in order to adjust speed and power consumption. 00: High speed/maximum power consumption 01: Medium speed/medium power consumption 10: Low speed/low power consumption 11: Ultra-low speed/ ultra-low power consumption
Bit 1	CMPIS	0x0	rw	Comparator input shift 0: The switch is off. 1: The switch is on. Note: This bit is used to enable the connection between PA1 and PA4 on the comparator inverting input side. It is only used to re-direct signal to high-impedance input, such as the non-inverting input of Comparator (high-impedance switch).
Bit 0	CMPEN	0x0	rw	Comparator enable This bit enables or disables a comparator. 0: Comparator disabled 1: Comparator enabled

## 19.6.2 Glitch filter enable register (G\_FILTER\_EN)

Bit	Register	Reset value	Type	Description
Bit 15: 1	Reserved	0x0000	resd	Kept at its default value.
Bit 0	GFE	0x0	rw	Glitch filter enable 0: No effect 1: Glitch filter enabled

## 19.6.3 Glitch filter high pulse count (HIGH-PULSE)

Bit	Register	Reset value	Type	Description
Bit 15: 6	Reserved	0x000	resd	Kept at its default value.
Bit 5: 0	H_PULSE_CNT	0x0	rw	High pulse Count The level of the filter input signal must wait H_PULSE_CNT+1 cycles before becoming active input, so that the output can turn high level. 0: 1 x pclk 1: 2 x pclk cycles 2: 3 x pclk cycles ..... 62: 63 x pclk cycles 63: 64 x pclk cycles

### 19.6.4 Glitch filter low pulse count (LOW-PULSE)

Bit	Register	Reset value	Type	Description
Bit 15: 6	Reserved	0x000	resd	Kept at its default value. Low pulse Count The level of the filter input signal must wait H_PULSE_CNT+1 cycles before becoming active input, so that the output can turn low level.
Bit 5: 0	L_PULSE_CNT	0x0	rw	0: 1 x pclk 1: 2 x pclk cycles 2: 3 x pclk cycles ..... 62: 63 x pclk cycles 63: 64 x pclk cycles

## 20 Operational amplifier (OPA)

The operational amplifier (OPA) applies to the AT32F421 series.

### 20.1 Introduction

The device embeds two rail-to-rail OPAs with up to 6 MHz bandwidth and an offset less than 4.5mV.

### 20.2 Main features

- Offset voltage: 0.8mV (typical)
- High gain: 97dB (typical)
- High gain bandwidth: 6MHz
- Rail-to-rail input and output
- +2.4 to 3.6V operation voltage
- Quiescent mode: 900uA (single-version OPA) at 3.3V

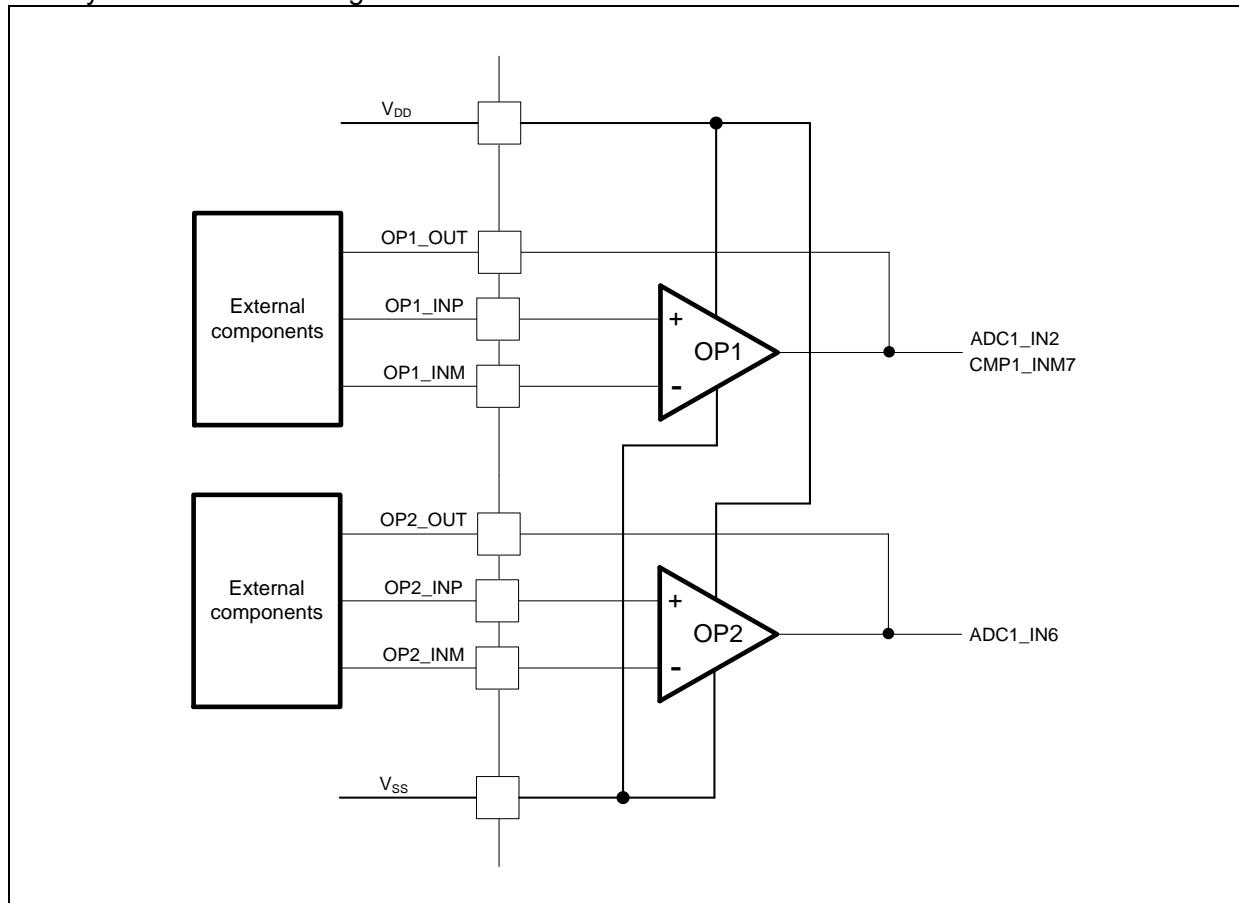
### 20.3 Functional description

For the products with embedded OPA, its OPA is automatically enabled and cannot be switched off. Thus there are some limitations on the use of GPIO pin which is shared with the OPA pin.

The OPx\_INy can still be used as GPIO or its alternate function pin; As OPA input, setting this pin as analog mode is recommended. Note that the input of such pin must be less than VDD+0.3V.

It is also advised to set the OPx\_OUT pin in analog mode. There are a variety of possibilities causing the pin to output voltage because of external circuit and the level of the OP\_IN. Therefore, this pin should not be used as an external input pin. However, the pin is internally connected to ADC1\_INy or CMP1\_INy, which is not affected by this case.

OPA system connection diagram:



Location relationship between device pins and OPA pins

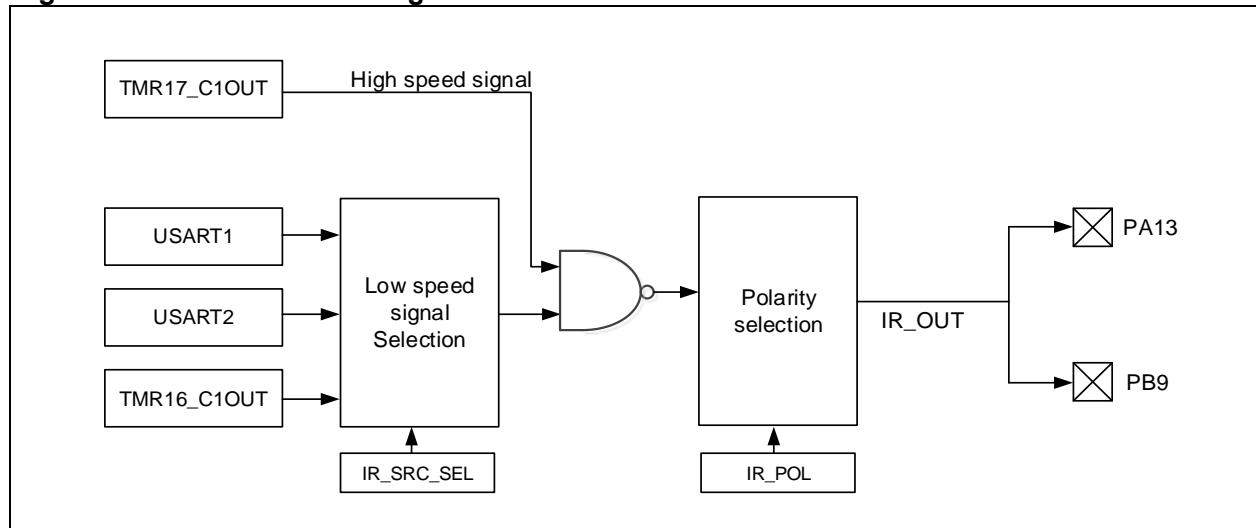
Pin name	OPA pin
PA0	OP1_INP
PA1	OP1_INM
PA2	OP1_OUT
PA4	OP2_INP
PA5	OP2_INM
PA6	OP2_OUT

## 21 Infrared timer (IRTMR)

The IRTMR (Infrared Timer) is used to generate the IR\_OUT signal that drives the infrared LED so as to achieve infrared control.

The IR\_OUT signals consists of a low-frequency modulation envelope and high-frequency carrier signals. The low-frequency modulation envelope signal selects from TMR16\_C1OUT, USART1 and USART2 through the IR\_SRC\_SEL[1: 0] bit in the SCFG\_CFG1 register, while the high-frequency carrier signal is provided by the TMR17\_C1OUT register. The IR\_POL bit in the SCFG\_CFG1 register controls whether the IR\_OUT output is reversed or not. The IR\_OUT signal is output through multiplexed function via PB9 or PA13 (multiplexed mode needs to be configured in advance).

**Figure 21-1 IRTMR block diagram**



## 22 Debug (DEBUG)

### 22.1 Debug introduction

Cortex™-M4 core provides powerful debugging features including halt and single step support, as well as trace function that is used for checking the details of the program execution. The debug features are implemented with a serial wire debug interface.

ARM Cortex™-M4 reference documentation:

- Cortex™-M4 Technical Reference Manual (TRM)
- ARM Debug Interface V5
- ARM CoreSight Design Kit revision r1p0 Technical Reference Manual

### 22.2 Debug and trace

This feature supports debugging for different peripherals, and configures the status of peripherals during debugging. For timers and watchdogs, the user can select whether or not to stop or continue counting during debugging; For CAN, the user can select whether or not to stop or continue updating receive registers during debugging; For I<sup>2</sup>C, the user can select whether or not to stop or continue SMBUS timeout counting.

In addition, code debugging is supported in Low-power mode. In Sleep mode, the clock programmed by code remains active for HCLK and FCLK to continue to work. In Deepsleep mode, HICK oscillator is enabled to feed FCLK and HCLK.

There are several ID codes inside the MCU, which is accessible by the debugger using the DEBUG\_IDCODE at address 0xE0042000. It is part of the DEBUG and is mapped on the external PPB bus. These codes are accessible by the SW debug port or by the user software. They are even accessible while the MCU is under system reset.

Two trace interface modes supported: single-pin mode for serial wire view and multi-pin trace interface.

### 22.3 I/O pin control

The AT32F421 has two general-purpose I/O ports for SW-DP debugging. After a system reset, the SW-DP can be immediately used by the debugger as a default function.

When the user wants to switch to a different debug port or disable debug feature, it is possible to release these dedicated I/O pins by setting GPIO registers. Once a corresponding debug I/O is released by the user, the GPIO controller takes control, and releases these I/Os for general purposes.

### 22.4 DEBUG registers

**Table 22-1** shows DEBUG register map and reset values.

These peripheral registers must be accessed by word (32 bits)

**Table 22-1 DEBUG register address and reset value**

Register name	Offset	Reset value
DEBUG_IDCODE	0xE004 2000	0xFFFF XXXX
DEBUG_CTRL	0xE004 2004	0x0000 0000

## 22.4.1 DEBUG device ID (DEBUG\_IDCODE)

MCU integrates an ID code that is used to identify MCU's revision code. The DEBUG\_IDCODE register is mapped on the external PPB bus at address 0xE0042000. This code is accessible by the JTAG or SW debug port or by the user code.

Bit	Register	Reset value	Type	Description
Bit 31: 0	PID	0xFFFF XXXX ro		PID information
<hr/>				
PID [31: 0]	AT32 part number	FLASH size	Packages	
0x50020100	AT32F421C8T7	64KB	LQFP48	
0x50020101	AT32F421K8T7	64KB	LQFP32	
0x50020102	AT32F421K8U7	64KB	QFN32 (5x5)	
0x50020103	AT32F421K8U7-4	64KB	QFN32 (4x4)	
0x50020104	AT32F421F8U7	64KB	QFN20	
0x50020105	AT32F421F8P7	64KB	TSSOP20	
0x50020086	AT32F421C6T7	32KB	LQFP48	
0x50020087	AT32F421K6T7	32KB	LQFP32	
0x50020088	AT32F421K6U7	32KB	QFN32 (5x5)	
0x50020089	AT32F421K6U7-4	32KB	QFN32 (4x4)	
0x5002008A	AT32F421F6U7	32KB	QFN20	
0x5002008B	AT32F421F6P7	32KB	TSSOP20	
0x5001000C	AT32F421C4T7	16KB	LQFP48	
0x5001000D	AT32F421K4T7	16KB	LQFP32	
0x5001000E	AT32F421K4U7	16KB	QFN32 (5x5)	
0x5001000F	AT32F421K4U7-4	16KB	QFN32 (4x4)	
0x50010010	AT32F421F4U7	16KB	QFN20	
0x50010011	AT32F421F4P7	16KB	TSSOP20	
0x50020112	AT32F421G8U7	64KB	QFN28	
0x50020093	AT32F421G6U7	32KB	QFN28	
0x50010014	AT32F421G4U7	16KB	QFN28	

## 22.4.2 DEBUG control register (DEBUG\_CTRL)

This register is asynchronously reset by POR Reset (not reset by system reset). It can be written by the debugger under reset.

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0000	resd	Kept at its default value.
				TMR14 pause control bit
Bit 27	TMR14_PAUSE	0x0	rw	0: Work normally 1: Timer is disabled
Bit 26: 25	Reserved	0x0	resd	Kept at its default value.
				TMR17 pause control bit
Bit 24	TMR17_PAUSE	0x0	rw	0: Work normally 1: Timer is disabled
				TMR16 pause control bit
Bit 23	TMR16_PAUSE	0x0	rw	0: Work normally 1: Timer is disabled
				TMR15 pause control bit
Bit 22	TMR15_PAUSE	0x0	rw	0: Work normally 1: Timer is disabled
				ERTC512Hz pause control bit 0: Work normally when ERTC512Hz 1: Receive register stops receiving data when ERTC512HZ.
Bit 20	Reserved	0x0	resd	Kept at its default value.
				TMR6 pause control bit
Bit 19	TMR6_PAUSE	0x0	rw	0: Work normally 1: Timer is disabled
Bit 18: 17	Reserved	0x0	resd	Kept at its default value.
				I2C2 pause control bit
Bit 16	I2C2_SMBUS_TIMEOUT	0x0	rw	0: Work normally 1: I2C2 SMBUS timeout control is disabled
				I2C1 pause control bit
Bit 15	I2C1_SMBUS_TIMEOUT	0x0	rw	0: Work normally 1: I2C1 SMBUS timeout control is disabled
				ERTC pause control bit
Bit 14	ERTC_PAUSE	0x0	rw	0: ERTC works normally 1: ERTC receive registers stops receiving data.
Bit 13	Reserved	0x0	resd	Kept at its default value.
				TMR3 pause control bit
Bit 12	TMR3_PAUSE	0x0	rw	0: Work normally 1: Timer is disabled
Bit 11	Reserved	0x0	resd	Kept at its default value.
				TMR1 pause control bit
Bit 10	TMR1_PAUSE	0x0	rw	0: Work normally 1: Timer is disabled
				Window watchdog pause control bit
Bit 9	WWDT_PAUSE	0x0	rw	0: Window watchdog works normally 1: Window watchdog is stopped
				watchdog pause control bit
Bit 8	WDT_PAUSE	0x0	rw	0: Watchdog works normally 1: Watchdog is stopped
Bit 7: 3	Reserved	0x00	resd	Kept at its default value.
Bit 2	STANDBY_DEBUG	0x0	rw	Debug Standby mode control bit 0: The whole 1.2V digital circuit is unpowered in Standby

				mode
				1: The whole 1.2V digital circuit is not unpowered in Standby mode, and the system clock is provided by the internal RC oscillator (HICK)
				Debug Deepsleep mode control bit
				0: In Deepsleep mode, all clocks in the 1.2V domain are disabled. When exiting from Deepsleep mode, the internal RC oscillator (HICK) is enabled, and HICK is used as the system clock source, and the software must reprogram the system clock according to application requirements.
Bit 1	DEEPSLEEP_DEBUG	0x0	rw	1: In Deepsleep mode, system clock is provided by the internal RC oscillator (HICK). When exiting from Deepsleep mode, HICK is used as the system clock source, and the software must reprogram the system clock according to application requirements.
				Debug Sleep mode control bit
Bit 0	SLEEP_DEBUG	0x0	rw	0: When entering Sleep mode, CPU HCLK clock is disabled, but other clocks remain active. When exiting from Sleep mode, it is not necessary to reprogram the clock system.
				1: When entering Sleep mode, all clocks keep running.

## 23 Revision history

Document Revision History

Date	Version	Revision Note
2021.11.17	2.00	Initial release.
2022.06.27	2.01	<ul style="list-style-type: none"><li>1. Updated the descriptions in <a href="#">Section 1.1.5 Reset</a></li><li>2. Updated the descriptions in <a href="#">3.6 Power saving modes</a></li><li>3. Updated the descriptions in <a href="#">4.3.2 Clock configuration register (CRM_CFG)</a></li><li>4. Updated the descriptions in <a href="#">11.5.1 Control register1 (I2C_CTRL1)</a></li><li>5. Updated <a href="#">13 Serial peripheral interface (SPI)</a></li><li>6. Updated the descriptions in <a href="#">18.5.3 ADC control register2 (ADC_CTRL2)</a></li></ul>
2022.11.11	2.02	<ul style="list-style-type: none"><li>1. Updated descriptions of <a href="#">Section 5.8.1</a></li><li>2. Updated descriptions of <a href="#">Chapter 10</a></li><li>3. Updated descriptions of <a href="#">Chapter 14</a></li><li>4. Updated descriptions of <a href="#">Section 17.2</a></li></ul>

**IMPORTANT NOTICE – PLEASE READ CAREFULLY**

Purchasers are solely responsible for the selection and use of ARTERY's products and services; ARTERY assumes no liability for purchasers' selection or use of the products and the relevant services.

No license, express or implied, to any intellectual property right is granted by ARTERY herein regardless of the existence of any previous representation in any forms. If any part of this document involves third party's products or services, it does NOT imply that ARTERY authorizes the use of the third party's products or services, or permits any of the intellectual property, or guarantees any uses of the third party's products or services or intellectual property in any way.

Except as provided in ARTERY's terms and conditions of sale for such products, ARTERY disclaims any express or implied warranty, relating to use and/or sale of the products, including but not restricted to liability or warranties relating to merchantability, fitness for a particular purpose (based on the corresponding legal situation in any judicial districts), or infringement of any patent, copyright, or other intellectual property right.

ARTERY's products are not designed for the following purposes, and thus not intended for the following uses: (A) Applications that have specific requirements on safety, for example: life-support applications, active implant devices, or systems that have specific requirements on product function safety; (B) Aviation applications; (C) Auto-motive application or environment; (D) Aerospace applications or environment, and/or (E) weapons. Since ARTERY products are not intended for the above-mentioned purposes, if purchasers apply ARTERY products to these purposes, purchasers are solely responsible for any consequences or risks caused, even if any written notice is sent to ARTERY by purchasers; in addition, purchasers are solely responsible for the compliance with all statutory and regulatory requirements regarding these uses.

Any inconsistency of the sold ARTERY products with the statement and/or technical features specification described in this document will immediately cause the invalidity of any warranty granted by ARTERY products or services stated in this document by ARTERY, and ARTERY disclaims any responsibility in any form.

© 2022 ARTERY Technology - All Rights Reserved.