



# ASR6601

## Reference Manual

Version: 1.5.0

Date: 2022-08-11

Copyright © 2022 ASR Technology



## About this document

This document provides detailed and complete information on the IoT LPWAN SoC-ASR6601 for application developers.

## Target Audience

This document is mainly intended for the following engineers:

- hardware development engineer
- software engineer
- technical support engineer

## Product numbering

Product models corresponding to this document:

Model	Flash	SRAM	Core	Package	Frequency
ASR6601SE	256 KB	64 KB	32-bit 48 MHz Arm China STAR-MC1	QFN68, 8*8 mm	150 ~ 960 MHz
ASR6601CB	128 KB	16 KB	32-bit 48 MHz Arm China STAR-MC1	QFN48, 6*6 mm	150 ~ 960 MHz

## Copyright Notice

Copyright © 2022 ASR Technology Co., Ltd. All rights reserved. No part or all of this document may be reproduced, transmitted, transcribed, stored or translated in any form or by any means without the written permission of ASR Technology Co., Ltd.

## Trademark Notice



ASR, ASR and other ASR logos are trademarks of ASR Technology Co., Ltd.

All other trade names, trademarks and registered trademarks mentioned in this document are the property of their respective owners and are hereby acknowledged.



## Disclaimer

ASR Technology Co., Ltd. makes no warranty of any kind for the contents of this document and will update the contents of this document or the products described in this document from time to time.

This document is only used as a guide for use, and all contents of this document do not constitute any form of warranty. The information in this document is subject to change without notice.

This document does not assume any responsibility, including responsibility for infringement of any proprietary rights arising from the use of the information in this document.

## ASR Technology Co., Ltd

Address: 9th Floor, Building 10, Zhangjiang Innovation Park, 399 Keyuan Road, Pudong New District, Shanghai Postal Code: 201203

URL : <http://www.asrmicro.com/>

## Document Revision History

Date	Version	Release Notes
2021.03	V1.0.0	Initial Release.
2021.05	V1.1.0	<ul style="list-style-type: none"><li>● Updated Chapter 6 Overview and Table 6-1.</li><li>● Updated parts of descriptions in Sections 16.3, 16.9, and 16.14.1.</li><li>● Corrected the description of LORAC_SR register in Section 12.4.13.</li></ul>
2021.07	V1.2.0	Updated CPU description.
2022.03	V1.3.0	<ul style="list-style-type: none"><li>● Added Chapter 21: DMA and Chapter 22: GPTIMER.</li><li>● Corrected several typos.</li></ul>
2022.05	V1.4.0	Modified RCO4M to RCO3.6M due to crystal frequency adjustment.
2022.08	V1.5.0	<ul style="list-style-type: none"><li>● Updated descriptions of some register bits in Sections 7.5.3, 8.3.3, 8.3.4, 8.3.7, 8.3.12, and 8.3.13.</li><li>● Updated Figure 8-1: Clock network diagram.</li></ul>



# Table of contents

<b>1. Overview .....</b>	<b>1</b>
<b>2. ASR6601 Introduction .....</b>	<b>2</b>
<b>3. Modules and functions .....</b>	<b>3</b>
3.1 ASR6601 SoC Diagram .....	3
3.2 ASR6601 function.....	4
<b>4. Power Management Unit .....</b>	<b>9</b>
4.1 Power Supply pins .....	9
4.2 Power Supply Architecture .....	10
<b>5. Access Control .....</b>	<b>11</b>
5.1 Simple Configuration .....	11
5.1.1 Recoverable Security Configuration .....	11
5.1.2 Unrecoverable Security Configuration .....	11
5.2 Access Control .....	11
5.2.1 Debug Level Rules .....	11
5.2.2 Secure and Non-Secure Operation .....	12
<b>6. Operation Modes .....</b>	<b>13</b>
6.1 Run .....	16
6.1.1 Enter and Exit .....	16
6.1.2 Wakeup Source .....	16
6.2 LpRun .....	16
6.2.1 Enter and Exit .....	16
6.2.2 Wakeup Source .....	16
6.3 Sleep .....	17
6.3.1 Enter and Exit .....	17
6.3.2 Wakeup Source .....	17
6.4 LpSleep.....	17
6.4.1 Enter and Exit .....	17
6.4.2 Wakeup Source .....	17
6.5 Stop0 .....	18
6.5.1 Enter and Exit .....	18
6.5.2 Wakeup Source .....	18
6.6 Stop1 .....	18
6.6.1 Enter and Exit .....	18
6.6.2 Wakeup Source .....	19
6.7 Stop2 .....	19
6.7.1 Enter and Exit .....	19
6.7.2 Wakeup Source .....	19



6.8 Stop3 .....	20
6.8.1 Enter and Exit .....	20
6.8.2 Wakeup Source .....	20
6.9 Standby.....	20
6.9.1 Enter and Exit .....	20
6.9.2 Wakeup Source .....	20
<b>7. System configuration.....</b>	<b>21</b>
7.1 System Architecture .....	21
7.1.1 Arm China STAR-MC1 Processor .....	22
7.1.2 DMAC0 .....	22
7.1.3 DMAC1 .....	22
7.1.4 Master .....	22
7.2 Memory Mapping .....	23
7.2.1 AHB0 SFR .....	24
7.2.2 AHB1 SFR .....	24
7.2.3 APB0 SFR .....	24
7.2.4 APB1 SFR .....	25
7.3 SRAM .....	25
7.4 Boot Modes .....	26
7.5 SYSCFG Registers .....	27
7.5.1 SYSCFG_CR0 .....	28
7.5.2 SYSCFG_CR1 .....	28
7.5.3 SYSCFG_CR2 .....	29
7.5.4 SYSCFG-CR3 .....	32
7.5.5 SYSCFG_CR4 .....	32
7.5.6 SYSCFG_CR5 .....	33
7.5.7 SYSCFG_CR6 .....	33
7.5.8 SYSCFG_CR7 .....	34
7.5.9 SYSCFG_CR8 .....	36
7.5.10 SYSCFG_CR9 .....	36
7.5.11 SYSCFG_CR10 .....	36
7.6 DMA Request MUX .....	38
<b>8. Reset and Clock Control (RCC).....</b>	<b>40</b>
8.1 Reset .....	40
8.1.1 External Reset .....	40
8.1.2 Power-on Reset .....	40
8.1.3 System Reset .....	40
8.1.4 Low-power Reset .....	40
8.2 Clock .....	41
8.2.1 SYS_CLK .....	42
8.2.2 Clocks for the Modules .....	42
8.2.3 MCO Clock output .....	43



8.3 RCC Registers .....	43
8.3.1 RCC_CR0 .....	44
8.3.2 RCC_CR1 .....	46
8.3.3 RCC_CR2 .....	48
8.3.4 RCC_CGR0 .....	50
8.3.5 RCC_CGR1 .....	53
8.3.6 RCC_CGR2 .....	54
8.3.7 RCC_RST0 .....	55
8.3.8 RCC_RST1 .....	58
8.3.9 RCC_RST_SR .....	59
8.3.10 RCC_RST_CR .....	60
8.3.11 RCC_SR .....	61
8.3.12 RCC_SR1 .....	62
8.3.13 RCC_CR3 .....	64
<b>9. Interrupts.....</b>	<b>66</b>
9.1 Main Features .....	66
9.2 SysTick .....	66
9.3 Interrupt Vector Table .....	66
<b>10. Embedded Flash .....</b>	<b>68</b>
10.1 Introduction .....	68
10.2 Main Features .....	68
10.3 Functional Description .....	68
10.3.1 Flash Info Area Division .....	68
10.3.2 EFC_CR Protection .....	69
10.3.3 Read Access Latency .....	69
10.3.4 Accessing Acceleration .....	69
10.3.5 Instruction Prefetch .....	70
10.3.6 Flash Program .....	70
10.3.7 Flash Erase .....	71
10.4 Flash Option Bytes .....	72
10.4.1 Flash Option0 .....	72
10.4.2 Flash Option1 .....	74
10.5 Embedded Flash Registers .....	75
10.5.1 EFC_CR .....	76
10.5.2 EFC_INT_EN .....	78
10.5.3 EFC_SR .....	79
10.5.4 EFC_PROG_DATA0 .....	80
10.5.5 EFC_PROG_DATA1 .....	80
10.5.6 EFC_TIMING_CFG .....	81
10.5.7 EFC_PROTECT_SEQ .....	81
10.5.8 SERIAL_NUM_LOW .....	82
10.5.9 SERIAL_NUM_HIGH .....	82



10.5.10 OPTION_CSR_BYTES .....	82
10.5.11 OPTION_EXE_ONLY_BYTES .....	83
10.5.12 OPTION_WR_PROTECT_BYTES .....	84
10.5.13 OPTION_SECURE_BYT <sub>E</sub> S0 .....	84
10.5.14 OPTION_SECURE_BYT <sub>E</sub> S1 .....	85
<b>11. GPIO.....</b>	<b>86</b>
11.1 Introduction .....	86
11.2 Output Configuration .....	86
11.3 Input Configuration .....	86
11.4 Output Drive Strength .....	87
11.5 GPIO Interrupts .....	87
11.6 Wakeup from Sleep/Stop0~2 Mode .....	87
11.7 Wakeup from Stop3 Mode .....	87
11.8 Alternate Function Configuration .....	87
11.9 Clock and Reset .....	87
11.10 Power Domains .....	88
11.11 Low-power Mode Operation and Wakeup .....	88
11.12 SWD IO.....	88
11.13 BOOT0 Control .....	88
11.14 GPIO Registers .....	89
11.14.1 GPIOx_OER (x=A, B, C, D) .....	90
11.14.2 GPIOx_OTYPER (x=A, B, C, D) .....	90
11.14.3 GPIOx_IER (x=A, B, C, D) .....	90
11.14.4 GPIOx_PER (x=A, B, C, D) .....	91
11.14.5 GPIOx_PSR (x=A, B, C, D) .....	91
11.14.6 GPIOx_IDR (x=A, B, C, D) .....	91
11.14.7 GPIOx_ODR (x=A, B, C, D) .....	92
11.14.8 GPIOx_BRR (x=A, B, C, D) .....	92
11.14.9 GPIOx_BSRR (x=A, B, C, D) .....	92
11.14.10 GPIOx_DSR (x=A, B, C, D) .....	93
11.14.11 GPIOx_INT_CR (x=A, B, C, D) .....	93
11.14.12 GPIOx_FR (x=A, B, C, D) .....	94
11.14.13 GPIOx_WU_EN (x=A, B, C, D) .....	94
11.14.14 GPIOx_WU_LVL (x=A, B, C, D) .....	94
11.14.15 GPIOx_AFRL (x=A, B, C, D) .....	95
11.14.16 GPIOx_AFRH (x=A, B, C) .....	97
11.14.17 GPIOD_AFRH .....	99
11.14.18 GPIOA_STOP3_WU_CR .....	101
11.14.19 GPIOx_STOP3_WU_CR (x=B, C).....	103
11.14.20 GPIOD_STOP3_WU_CR .....	105
<b>12. LoRa Controller (LoRaC) .....</b>	<b>106</b>
12.1 Introduction .....	106



12.2 Main Features .....	106
12.3 Functional Description .....	106
12.3.1 Internal SPI Interface .....	106
12.3.2 Timing Sequence of Power-on .....	107
12.3.3 Interrupts .....	107
12.4 LoRaC Registers .....	108
12.4.1 SSP_CR0 .....	109
12.4.2 SSP_CR1 .....	110
12.4.3 SSP_DR .....	110
12.4.4 SSP_SR .....	111
12.4.5 SSP_CPSR .....	111
12.4.6 SSP_IMSC .....	112
12.4.7 SSP_RIS .....	112
12.4.8 SSP_MIS .....	113
12.4.9 SSP_ICR .....	113
12.4.10 SSP_DMACR .....	113
12.4.11 LORAC_CR0 .....	114
12.4.12 LORAC_CR1 .....	114
12.4.13 LORAC_SR .....	115
12.4.14 LORAC_NSS_CR .....	116
12.4.15 LORAC_SCK_CR .....	116
12.4.16 LORAC_MOSI_CR .....	116
12.4.17 LORAC_MISO_SR .....	117
<b>13. UART .....</b>	<b>118</b>
13.1 Introduction .....	118
13.2 Clock and Reset .....	118
13.3 Reference Clock .....	118
13.4 Baud Rate Generator .....	118
13.5 FIFO .....	119
13.6 UART Operation .....	119
13.6.1 Baud Rate Divider .....	119
13.6.2 Data Transmission .....	119
13.6.3 Data Reception .....	120
13.7 IrDA SIR Operation .....	120
13.7.1 Low-Power Divider .....	120
13.7.2 IrDA SIR IrDA SIR Transmit Encoder .....	121
13.7.3 IrDA SIR IrDA SIR Receive Decoder .....	121
13.8 UART Character Frame .....	121
13.9 IrDA Data Modulation .....	122
13.10 Hardware Flow Control .....	122
13.11 Interrupts .....	122
13.12 DMA .....	122



13.13 UART Registers .....	123
13.13.1 UARTx_DR (x=0, 1, 2, 3) .....	124
13.13.2 UARTx_RSR_ECR (x=0, 1, 2, 3) .....	125
13.13.3 UARTx_FR (x=0, 1, 2, 3) .....	126
13.13.4 UARTx_ILPR (x=0, 1, 2, 3) .....	127
13.13.5 UARTx_IBRD (x=0, 1, 2, 3) .....	127
13.13.6 UARTx_FBRD (x=0, 1, 2, 3) .....	127
13.13.7 UARTx_LCR_H (x=0, 1, 2, 3) .....	128
13.13.8 UARTx_CR (x=0, 1, 2, 3) .....	129
13.13.9 UARTx_IFLS (x=0, 1, 2, 3) .....	130
13.13.10 UARTx_IMSC (x=0, 1, 2, 3) .....	130
13.13.11 UARTx_RIS (x=0, 1, 2, 3) .....	131
13.13.12 UARTx_MIS (x=0, 1, 2, 3) .....	132
13.13.13 UARTx_ICR (x=0, 1, 2, 3) .....	132
13.13.14 UARTx_DMACR (x=0, 1, 2, 3) .....	133
13.13.15 UARTx_ID[8] (x=0, 1, 2, 3) .....	134
<b>14. SSP .....</b>	<b>137</b>
14.1 Introduction .....	137
14.2 Main Features .....	137
14.3 Functional Description .....	137
14.3.1 Basic Information .....	137
14.3.2 Clock Division .....	138
14.3.3 Data Format .....	139
14.3.4 DMA Transfer .....	139
14.3.5 Interrupts .....	140
14.4 SSP Registers .....	140
14.4.1 SSP_CR0.....	141
14.4.2 SSP_CR1.....	142
14.4.3 SSP_DR .....	142
14.4.4 SSP_SR .....	143
14.4.5 SSP_CPSR .....	143
14.4.6 SSP_IMSC .....	144
14.4.7 SSP_RIS .....	144
14.4.8 SSP_MIS .....	145
14.4.9 SSP_ICR.....	145
14.4.10 SSP_DMACR .....	146
<b>15. I2C .....</b>	<b>147</b>
15.1 Introduction .....	147
15.2 Start and Stop Conditions .....	148
15.3 Data Transmission Sequence .....	149
15.4 Data and Addressing .....	150
15.5 Acknowledge .....	151



15.6 Arbitration .....	151
15.7 I2C Master Mode .....	152
15.8 FIFO Mode .....	154
15.9 I2C Slave Mode .....	156
15.10 Clock and Reset .....	157
15.11 Interrupts .....	157
15.12 DMA .....	157
15.13 I2C Registers .....	158
15.13.1 I2Cx_CR (x=0, 1, 2) .....	159
15.13.2 I2Cx_SR (x=0, 1, 2) .....	162
15.13.3 I2Cx_SAR (x=0, 1, 2) .....	163
15.13.4 I2Cx_DBTR (x=0, 1, 2) .....	164
15.13.5 I2Cx_LCR (x=0, 1, 2) .....	164
15.13.6 I2Cx_WCR (x=0, 1, 2) .....	164
15.13.7 I2Cx_RST_CYCL (x=0, 1, 2) .....	165
15.13.8 I2Cx_BMR (x=0, 1, 2) .....	165
15.13.9 I2Cx_WFIFO0 (x=0, 1, 2) .....	165
15.13.10 I2Cx_WFIFO_WPTR (x=0, 1, 2) .....	166
15.13.11 I2Cx_WFIFO_RPTR (x=0, 1, 2) .....	166
15.13.12 I2Cx_RFIFO (x=0, 1, 2) .....	166
15.13.13 I2Cx_RFIFO_WPTR (x=0, 1, 2) .....	167
15.13.14 I2Cx_RFIFO_RPTR (x=0, 1, 2) .....	167
15.13.15 I2Cx_WFIFO_STATUS (x=0, 1, 2) .....	167
15.13.16 I2Cx_RFIFO_STATUS (x=0, 1, 2) .....	168
<b>16. ADC .....</b>	<b>169</b>
16.1 Introduction .....	169
16.2 ADC Input Modes .....	169
16.3 Sampling Channels .....	170
16.4 Trigger Source .....	170
16.5 Low-power Operation .....	171
16.6 Overrun .....	171
16.7 Conversion Modes .....	171
16.8 Voltage Reference .....	171
16.9 Data Buffer .....	172
16.10 DMA .....	172
16.11 Interrupts .....	173
16.12 Wakeup .....	173
16.13 Clock and Reset .....	173
16.14 ADC Registers .....	173
16.14.1 ADC_CR .....	174
16.14.2 ADC_CFGR .....	175
16.14.3 ADC_SEQR0 .....	177



16.14.4 ADC_SEQR1 .....	178
16.14.5 ADC_DIFFSEL .....	179
16.14.6 ADC_ISR .....	179
16.14.7 ADC_IER .....	180
16.14.8 ADC_DR .....	180
<b>17. Basic timer (BSTIM) .....</b>	<b>181</b>
17.1 Introduction .....	181
17.2 Main features .....	181
17.3 Clock source .....	182
17.4 Counter .....	182
17.5 Auto-reload .....	182
17.6 Prescaler .....	182
17.7 DMA .....	183
17.8 Single pulse mode .....	183
17.9 Mode selection .....	183
17.10 Update event management .....	184
17.11 Debug mode control .....	184
17.12 Interrupts .....	184
17.13 BSTIMER registers .....	185
17.13.1 BSTIM_CR1 .....	186
17.13.2 BSTIM_CR2 .....	187
17.13.3 BSTIM_DIER .....	187
17.13.4 BSTIM_SR .....	188
17.13.5 BSTIM_EGR .....	188
17.13.6 BSTIM_CNT .....	188
17.13.7 BSTIM_PSC .....	189
17.13.8 BSTIM_ARR .....	189
<b>18. RTC .....</b>	<b>190</b>
18.1 Introduction .....	190
18.2 Main Features .....	190
18.3 Interface Clock .....	190
18.4 Calendar .....	191
18.4.1 Calendar reading .....	191
18.4.2 Calendar setting .....	191
18.5 RTC PPM Calibration .....	192
18.6 Wake-up from Low-power Mode .....	192
18.7 Tamper/Wakeup IO Detection .....	193
18.7.1 Tamper/Wakeup Initialization and Configuration .....	193
18.7.2 Retention SRAM Erase Operation .....	193
18.8 Periodic Counter .....	193
18.9 RTC Alarms .....	194



18.10 Internal Signal Output through IO .....	195
18.11 RTC Interrupts .....	195
18.12 RTC Registers .....	196
18.12.1 RTC_CR .....	197
18.12.2 RTC_ALARM0 .....	200
18.12.3 RTC_ALARM1 .....	201
18.12.4 RTC_PPMADJUST .....	202
18.12.5 RTC_CALENDAR .....	202
18.12.6 RTC_CALENDAR_H .....	203
18.12.7 RTC_CYC_MAX_VALUE .....	203
18.12.8 RTC_SR .....	204
18.12.9 RTC_ASYNCDATA .....	205
18.12.10 RTC_ASYNCDATA_H .....	205
18.12.11 RTC_CR1 .....	206
18.12.12 RTC_SR1 .....	207
18.12.13 RTC_CR2 .....	208
18.12.14 RTC_SUB_SECOND .....	209
18.12.15 RTC_CYC_CNT_VALUE.....	209
18.12.16 RTC_ALARM0_SUB .....	210
18.12.17 RTC_ALARM1_SUB .....	210
18.12.18 RTC_CALENDAR_R .....	211
18.12.19 RTC_CALENDAR_R_H.....	211
<b>19. Low-power UART (LPUART).....</b>	<b>212</b>
19.1 Introduction .....	212
19.2 Main Features .....	212
19.3 Functional Description .....	212
19.3.1 Data Format .....	212
19.3.2 Baud Rate Generation .....	213
19.3.3 CTS/RTS Flow Control .....	213
19.3.4 DMA .....	214
19.3.5 Interrupts .....	214
19.3.6 CPU Wakeup from Low-power Mode .....	215
19.4 LPUART Registers .....	215
19.4.1 LPUART_CR0 .....	216
19.4.2 LPUART_CR1 .....	217
19.4.3 LPUART_SR0 .....	218
19.4.4 LPUART_SR1 .....	219
19.4.5 LPUART_DATA .....	220
<b>20. Low-power timer (LPTIM) .....</b>	<b>221</b>
20.1 Introduction .....	221
20.2 Main features .....	221
20.3 Interface clock .....	222



20.4 Counter clock selection .....	222
20.5 Counter .....	223
20.6 Counting modes .....	223
20.7 Trigger sources .....	224
20.8 Prescaler .....	224
20.9 PWM .....	225
20.10 Single-pulse, Set-once, Timeout output mode .....	225
20.11 Quadrature encoder mode .....	226
20.12 DEBUG mode control .....	227
20.13 Wake-up signals .....	227
20.14 Interrupts .....	228
20.15 LPTIMER registers .....	228
20.15.1 LPTIM_ISR .....	229
20.15.2 LPTIM_ICR .....	230
20.15.3 LPTIM_IER .....	231
20.15.4 LPTIM_CFGR .....	232
20.15.5 LPTIM_CR .....	234
20.15.6 LPTIM_CMP .....	235
20.15.7 LPTIM_ARR .....	235
20.15.8 LPTIM_CNT .....	235
20.15.9 LPTIM_CSR .....	236
20.15.10 LPTIM_SR1 .....	237
<b>21. DMA .....</b>	<b>238</b>
21.1 Introduction .....	238
21.2 Main features .....	238
21.3 Transfer data length configuration .....	238
21.4 Data trasfer methods .....	239
21.5 LLI .....	241
21.6 Auto-reloading .....	241
21.7 Interrupts .....	242
21.8 DMA registers .....	243
21.8.1 DMA_SARx .....	244
21.8.2 DMA_DARx .....	244
21.8.3 DMA_LLPx .....	244
21.8.4 DMA_CTLx .....	245
21.8.5 DMA_CFGx .....	247
21.8.6 DMA_StatusTfr .....	249
21.8.7 DMA_StatusBlock .....	249
21.8.8 DMA_StatusSrcTran .....	250
21.8.9 DMA_StatusDstTran .....	250
21.8.10 DMA_StatusErr .....	251
21.8.11 DMA_MaskTfr .....	252



21.8.12 DMA_MaskBlock .....	253
21.8.13 DMA_MaskSrcTran .....	254
21.8.14 DMA_MaskDstTran .....	255
21.8.15 DMA_MaskErr .....	256
21.8.16 DMA_ClearTfr .....	257
21.8.17 DMA_ClearBlock .....	257
21.8.18 DMA_ClearSrcTran .....	258
21.8.19 DMA_ClearDstTran .....	258
21.8.20 DMA_ClearErr .....	259
21.8.21 DMA_DmaCfgReg .....	260
21.8.22 DMA_ChEnReg .....	260
<b>22. GPTIMER .....</b>	<b>262</b>
22.1 Introduction .....	262
22.2 Main features .....	262
22.3 Counter .....	264
22.3.1 Counter clock .....	264
22.3.2 Auto-reload .....	265
22.3.3 Up-count .....	265
22.3.4 Down-count .....	266
22.3.5 Center-aligned count .....	266
22.4 Prescaler .....	267
22.5 Capture mode .....	267
22.6 Troughput .....	268
22.6.1 Input capture .....	269
22.6.2 Output compare .....	269
22.7 Trigger input channels .....	272
22.8 Update event management .....	272
22.9 Quadrature encoder mode .....	273
22.10 Slave mode control .....	274
22.11 Master mode control .....	275
22.12 Output control .....	276
22.13 Channels remapping .....	276
22.14 Debug mode control .....	276
22.15 DMA .....	276
22.16 Interrupts .....	277
22.17 GPTIMER registers .....	277
22.17.1 GPTIM_CR1 .....	279
22.17.2 GPTIM_CR2 .....	280
22.17.3 GPTIM_SMCR .....	281
22.17.4 GPTIM_DIER .....	282
22.17.5 GPTIM_SR.....	284
22.17.6 GPTIM_EGR .....	285



22.17.7 GPTIM_CCMR1 .....	286
22.17.8 GPTIM_CCMR2 .....	289
22.17.9 GPTIM_CCER .....	292
22.17.10 GPTIM_CNT .....	294
22.17.11 GPTIM_PSC .....	294
22.17.12 GPTIM_ARR .....	295
22.17.13 GPTIM_CCR0 .....	295
22.17.14 GPTIM_CCR1 .....	295
22.17.15 GPTIM_CCR2 .....	296
22.17.16 GPTIM_CCR3 .....	296
22.17.17 GPTIM_DCR .....	296
22.17.18 GPTIM_DMAR .....	298
22.17.19 GPTIM_OR .....	298

ASR Confidential



# List of Tables

Table 3-1 ASR6601 functions .....	4
Table 6-1 Status of different modules in each operating mode .....	13
Table 7-1 Master bus access range .....	22
Table 7-2 Memory map .....	23
Table 7-3 AHB0 SFR address mapping .....	24
Table 7-4 AHB1 SFR address mapping .....	24
Table 7-5 APB0 SFR address mapping .....	24
Table 7-6 APB1 SFR address mapping .....	25
Table 7-7 ASR6601 Boot Mode Configuration .....	26
Table 7-8 SYSCFG Registers Summary .....	27
Table 7-9 DMA Request MUX .....	38
Table 8-1 RCC Registers Summary .....	43
Table 9-1 Interrupt Vectors .....	66
Table 10-1 Flash Info Area Division .....	68
Table 10-2 Flash Option0 .....	72
Table 10-3 ASR6601 Boot Mode Configuration .....	73
Table 10-4 Flash Option1 .....	74
Table 10-5 Embedded Flash Registers Summary .....	75
Table 11-1 GPIO Registers Summary .....	89
Table 12-1 LORAC Registers Summary .....	108
Table 13-1 Receive FIFO Bit Functions .....	119
Table 13-2 UART Registers Summary .....	123
Table 14-1 SSP Registers Summary .....	140
Table 15-1 Start and Stop Conditions .....	148
Table 15-2 Master Transactions .....	152
Table 15-3 Slave Transactions .....	156
Table 15-4 I2C Registers Summary .....	158
Table 16-1 ADC Sampling Channels .....	170
Table 16-2 ADC Registers Summary .....	173
Table 17-1 BSTIMER interrupts .....	184
Table 17-2 BSTIMER Registers Summary .....	185
Table 18-1 RTC Wakeup Source .....	192
Table 18-2 Bits to Enable Wake-up Signals .....	192
Table 18-3 RTC Interrupts .....	195
Table 18-4 RTC Registers Summary .....	196
Table 19-1 LPUART Registers Summary .....	215
Table 20-1 LPTIMER0 external trigger sources .....	224
Table 20-2 LPTIMER1 external trigger sources .....	224
Quadtature encoder channel signals .....	227
Table 20-4 LPTIMER interrupts .....	228
Table 20-5 LPTIMER Register Summary .....	228



Table 21-1 Handshake Values .....	239
Table 21-2 DMA interrupts .....	242
Table 21-3 DMA Registers Summary .....	243
Table 22-1 GPTIMER module introduction .....	263
Table 22-2 Input channels polarity configuration .....	268
Table 22-3 Input channel mapping .....	268
Table 22-4 Output waveform description .....	270
Table 22-5 Encoder mode .....	273
Table 22-6 GPTIMER internal trigger input mapping .....	274
Table 22-7 GPTIMER interrupts .....	277
Table 22-8 GPTIMER Registers Summary .....	277

ASR Confidential



# List of Figures

Figure 3-1 ASR6601 SoC Diagram .....	3
Figure 4-1 ASR6601 Power Grid .....	9
Figure 4-2 ASR6601 Power Supply Architecture .....	10
Figure 7-1 System Architecture Diagram .....	21
Figure 8-1 Clock Tree .....	41
Figure 12-1 Power-on Timing Sequence .....	107
Figure 13-1 UART Character Frame .....	121
Figure 13-2 IrDA Data Modulation (3/16) .....	122
Figure 14-1 Connection between a SSP Master and a SPI Slave .....	138
Figure 14-2 Connection between a SPI Master and a SSP Slave .....	138
Figure 14-3 MASTER mode clock output calculation .....	138
Figure 15-1 I2C Block Diagram .....	147
Figure 15-2 SDA and SCL Signals During Start and Stop Conditions .....	148
Figure 15-3 FIFO Mode Block Diagram .....	154
Figure 16-1 ADC Diagram .....	169
Figure 17-1 BSTIMER Diagram .....	181
Figure 17-2 Counting and Dividing Waveforms .....	182
Figure 17-3 Single pulse waveform .....	183
Figure 19-1 LPUART Data Format .....	212
Figure 19-2 Connection between Two LPUART Devices .....	213
Figure 20-1 LPTIMER Diagram .....	222
Figure 20-2 Counting mode conversion diagram .....	223
Figure 20-3 Single pulse counting .....	225
Figure 20-4 Set-once counting .....	226
Figure 20-5 Timeout counting .....	226
Figure 21-1 Data transfer .....	238
Figure 21-2 LLI chain table .....	241
Figure 22-1 GPTIMER diagram .....	263
Figure 22-2 External clock mode 1 counting .....	264
Figure 22-3 External clock mode 2 counting .....	264
Figure 22-4 Internal trigger signal for clock counting .....	265
Figure 22-5 Up-counting .....	265
Figure 22-6 Down-counting .....	266
Figure 22-7 Center-aligned counting .....	266
Figure 22-8 Prescaler .....	267
Figure 22-9 Input capture .....	269
Figure 22-10 Output compare mode waveforms .....	271
Figure 22-11 PWM2 edge-aligned counting .....	271
Figure 22-12 PWM2 center-aligned counting .....	271
Figure 22-13 Single pulse output waveform in fast mode .....	272
Figure 22-14 External brake signal trigger .....	272



Figure 22-15 Counting waveform of encoder mode 1.....	274
Figure 22-16 Reset mode waveform in slave mode .....	274
Figure 22-17 Gated mode waveform in slave mode .....	274
Figure 22-18 Trigger mode waveform in slave mode.....	275

ASR Confidential



# 1.

# Overview

ASR6601 is a general LPWAN Wireless Communication SoC chip developed by ASR which supports LoRa modulation. The chip integrates Sub-1G RF transceiver, Arm China STAR-MC1 processor, embedded Flash memory and SRAM, as well as diverse analog modules. ASR6601 is designed for a wide variety of applications, such as smart meters, building automation, smart cities, agricultural sensors, safety and security sensors, supply chain and logistics, etc.

This manual provides detailed and complete information on the IoT LPWAN SoC-ASR6601 for application developers. Together with the API file in SDK, it helps developers solve various problems they may encounter during development. If any further support is needed, please contact us. We will keep this manual updated.

ASR Confidential

## 2.

# ASR6601 Introduction

ASR6601 SoC is a low-power wide area network wireless communication SoC chip that supports LoRa modulation. The ultra-low power transceiver integrated in the ASR6601 chip supports the full frequency band of 150 MHz ~ 960 MHz with the off-chip matching network. In addition to supporting LoRa modulation, it can also support FSK transceiver, MSK transceiver and BPSK transmitter. When powered by 3.3 V power supply, the maximum output power of 22 dBm can be transmitted using the high-power PA. ASR6601 SoC mainly has Run, LpRun, Sleep, LpSleep, Stop0, Stop1, Stop2, Stop3, Standby working modes. Each mode supports different functions, working modules and power consumption. End users can choose the corresponding working mode according to their application scenarios. The two most commonly used low-power modes are Standby mode and Stop3 mode. When powered by 3.3 V, the Standby mode consumes as little as 0.9 uA; the Stop3 mode consumes as little as 1.3 uA (ASR6601CB) and 1.6 uA (ASR6601SE).

ASR6601 SoC uses a 32-bit ARM STAR core with a maximum main frequency of 48 MHz, supports SWD debug interface, supports SysTick, MPU, FPU functions, and supports 37 IRQs with 8 interrupt priorities.

ASR6601 supports UART, I2C, I2S, LPUART, SSP, QSPI and other interfaces. With the peripherals of different types of corresponding interfaces, it can realize rich functions to meet customer needs. In addition to supporting rich number functions, ASR6601 also integrates rich analog functions, including ADC, DAC, OPA and LCD driver.

ASR6601 implements AES encryption through hardware, greatly simplifying the efficiency of encryption and decryption. It also supports national encryption SM2/3/4.

# 3.

# Modules and functions

## 3.1 ASR6601 SoC Diagram

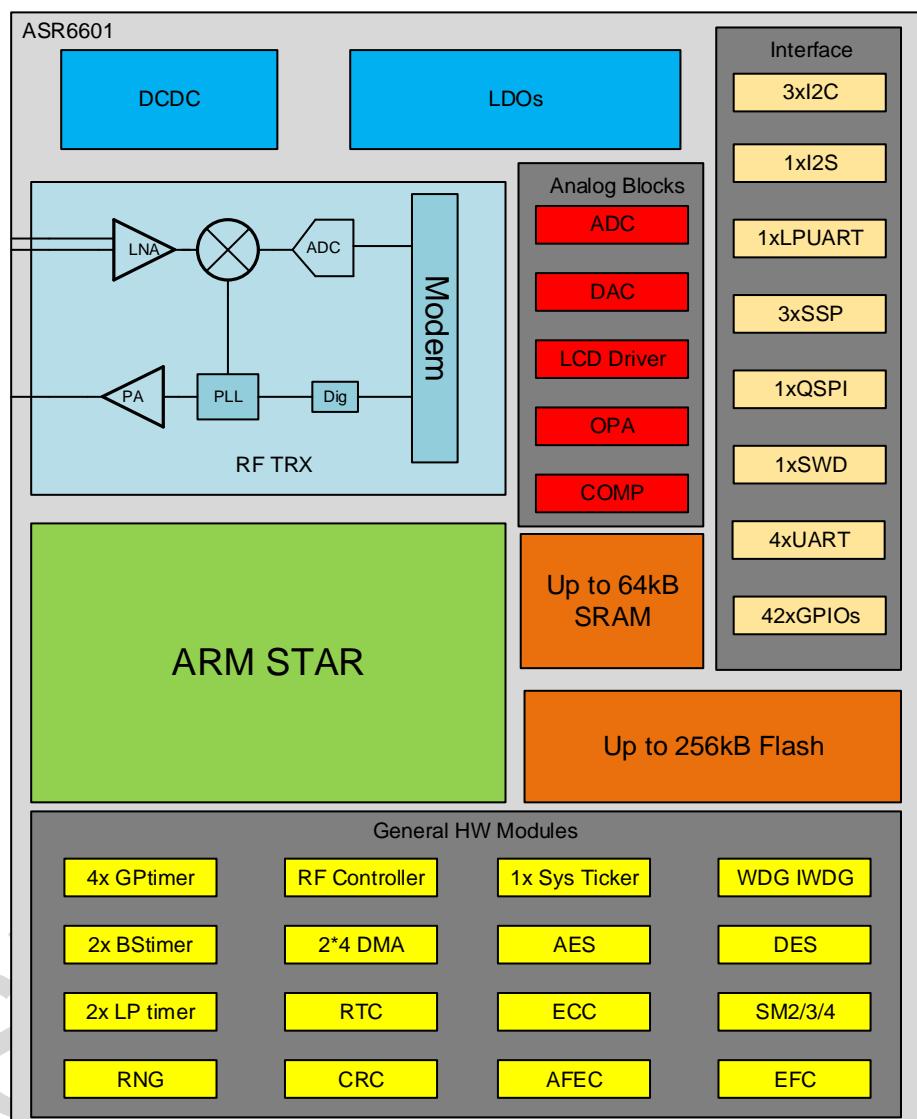


Figure 3-1 ASR6601 SoC Diagram

## 3.2 ASR6601 function

Table 3-1 ASR6601 functions

Module Name	Functions
RCC	Clock and reset control
SYSCFG	System function registers
PWR	<ol style="list-style-type: none"><li>1. Chip low power mode control</li><li>2. Interrupt signal generation</li></ol>
SEC	<ol style="list-style-type: none"><li>1. Security IP Enable</li><li>2. Filtering Security IP alarm signal filtering</li><li>3. Alarm signal processing</li></ol>
CPU	<ol style="list-style-type: none"><li>1. SWD debug interface</li><li>2. Systick function</li><li>3. MPU function</li><li>4. FPU function</li><li>5. 37 IRQs, 8 interrupt priorities</li></ol>
MPU	Access Security control
EFC	<ol style="list-style-type: none"><li>1. Power-on chip mode determination</li><li>2. Flash info area data loaded at power-on</li><li>3. Basic flash operations, including read, program, page erase, mass erase</li><li>4. Flash operation key timing control</li><li>5. Flash instruction prefetch function</li><li>6. Flash program operation supports single and continuous modes</li><li>7. Flash info area option bytes operation</li><li>8. Interrupt signals generating</li></ol>
I2S	<ol style="list-style-type: none"><li>1. Philips I2S serial protocol</li><li>2. Master and Slave modes</li><li>3. 1 RX channel, 1 TX channel, full duplex</li><li>4. Receive FIFO depth is 4</li><li>5. Transmit FIFO depth is 4</li><li>6. Receiver supports 12, 16, 20, 24, 32-bit resolution</li><li>7. Transmitter supports 12, 16, 20, 24, 32-bit resolution</li><li>8. Programmable DMA registers</li><li>9. Programmable FIFO Threshold</li><li>10. 1 interrupt signal generation</li></ol>
UART	<ol style="list-style-type: none"><li>1. IrDA, support 3/16 and low-power (1.41-2.23us) modes</li><li>2. FIFO, 16x8 bits for transmission, 16x10 bits for reception</li><li>3. Transmit and receiving buffers</li><li>4. Baud rate generation, using 16 times oversampling, supports 16-bit integer division and 6-bit fractional division, and supports up to interface clock frequency/16</li></ol>

Module Name	Functions
UART	<ul style="list-style-type: none"><li>5. UART data format configuration</li><li>6. DMA transfer</li><li>7. False start detection</li><li>8. Line break sending and detection</li><li>9. Hardware flow control CTS and RTS</li><li>10. Interrupt signal generation</li></ul>
LPUART	<ul style="list-style-type: none"><li>1. Low power wake-up</li><li>2. Baud rate generation, does not support oversampling, supports 4-bit fractional frequency division and 12-bit integer frequency division, the minimum integer frequency division is 3</li><li>3. UART data format configuration, including 1-2 bits Stop, 0-1 bits parity (odd, even, mark, space, none), 5-8 data bits</li><li>4. Hardware flow control CTS and RTS</li><li>5. DMA transfer</li><li>6. Interrupt signal generation</li></ul>
SSP	<ul style="list-style-type: none"><li>1. Master and Slave modes</li><li>2. Programmable baud rate and prescaler, Master supports up to 1/2 interface clock frequency, Slave supports up to 1/12 interface clock frequency</li><li>3. 8*16 Bit receiving and transmitting FIFO</li><li>4. Configurable data length, 4-16 Bit</li><li>5. DMA transfer</li><li>6. Motorola, Microwire (NS), TI formats</li><li>7. Motorola supports 4 polarity phase bit combinations</li><li>8. Interrupt signal generation</li></ul>
I2C	<ul style="list-style-type: none"><li>1. Master and slave modes, support multi-master arbitration</li><li>2. Multi-host arbitration</li><li>3. Standard Mode and Fast Mode</li><li>4. 7-bit address mode</li><li>5. Clock Stretching</li><li>6. Interrupt signal generation</li><li>7. DMA transfer</li></ul>
AFEC	<ul style="list-style-type: none"><li>1. IP status register</li><li>2. Simulate IP control register</li><li>3. Some registers support Safety lockControl</li><li>4. Interrupt signal generation</li></ul>
LORAC	<ul style="list-style-type: none"><li>1. LORA IP control register</li><li>2. LORA status register</li><li>3. LORA IP SPI interface source, supports ssp master control and reg control</li><li>4. DMA transfer</li><li>5. Interrupt signal generation</li></ul>

Module Name	Functions
RTC	<ol style="list-style-type: none"><li>1. Calendar counting function in BCD format</li><li>2. Ppm adjustment, adjustment step size 0.5ppm, +/-1024 ppm adjustment</li><li>3. Low power wake-up</li><li>4. Tamper/wakeup IO detection function</li><li>5. Cycle counting function, 32-bit counter</li><li>6. Alarm clock function, supports two alarm clocks, supports mask selection</li><li>7. Clear retention sram on Tamper alarm</li><li>8. Some registers support Safety lockControl</li><li>9. Internal signal IO output, including alarm0 matching pulse, alarm1 matching pulse, cycle count configuration pulse, seconds signal output</li><li>10. Calendar count value reading</li><li>11. Sub-second count value reading</li><li>12. Cycle count value reading</li><li>13. Interrupt signal generation</li></ol>
IWDG	<ol style="list-style-type: none"><li>1. Watchdog counting function, down-counting, clock prescaler (4-256)</li><li>2. Watchdog exception status occurs when the count reaches 0 (feeding the dog too late) or when the count value when feeding the dog is greater than the counting window value (feeding the dog too early)</li><li>3. Interrupt signal generation</li><li>4. Dog feeding window configuration</li><li>5. Count value reading</li><li>6. Low power wake-up</li></ol>
QSPI	<ol style="list-style-type: none"><li>1. Master interface only</li><li>2. 1-wire, 2-wire, 4-wire modes</li><li>3. 3 working modes, including indirect access, status query and Memory-mapping</li><li>4. Baud rate division, up to interface clock frequency/2</li><li>5. Interrupt signals generation</li></ol>
CRC	<ol style="list-style-type: none"><li>1. Configurable polynomial bit width: 7, 8, 16, 32 bits</li><li>2. Different hsize accesses, the lower byte is calculated first and can be edited</li><li>3. Programmable crc initial value</li><li>4. Input data reverse, supports byte, halfword and word</li><li>5. Output data reverse, supports word</li></ol>
DMA	<ol style="list-style-type: none"><li>1. 1 master interface AHB bus</li><li>2. AHB interface only supports little-endian structure</li><li>3. Interrupt signal generation</li><li>4. Transfer mode supports M2M, P2M, M2P, P2P</li><li>5. Software triggering handshake signal</li><li>6. 4 sets of hardware handshake signals, including burst and single requests</li><li>7. Hardware handshake signal sources, each group supports 64 source selections</li><li>9. Channel 0 configuration: (1) 8 bytes deep FIFO</li></ol>

Module Name	Functions
DMA	<ul style="list-style-type: none"><li>(2) Maximum burst length is 8</li><li>(3) Maximum transfer length is 2047</li><li>(4) Supports dmac flow control only</li><li>(5) Source address data bit width configurable</li><li>(6) Destination address data bit width configurable</li><li>(7) Address increment, decrement, and unchanged</li><li>(8) Block transfer, including continuous address, automatic loading and linked list</li><li>(9) Scatter and gather</li></ul> <p>10. Channel 1-3 configuration:</p> <ul style="list-style-type: none"><li>(1) 8 bytes deep FIFO</li><li>(2) Maximum burst length is 8</li><li>(3) Maximum transfer length is 2047</li><li>(4) Supports dmac flow control only</li><li>(5) Source address data bit width configurable</li><li>(6) Destination address data bit width configurable</li><li>(7) Address increment, decrement, and unchanged</li><li>(8) Block transfer, including continuous addresses and automatic loading, but does not support linked lists</li><li>(9) Scatter and gather not supported</li></ul>
GPIO	<ul style="list-style-type: none"><li>1. IO output configuration, push-pull, open drain, output high impedance</li><li>2. IO input configuration, floating, input pull-up, input pull-down, analog input</li><li>3. IO pull-up configuration, pull-down configuration, drive capability control</li><li>4. Interrupt signals generation, including rising edge interrupt, falling edge interrupt, and both edges interrupt</li><li>5. Wake-up signals generation, including high level and low level</li></ul>
SAE	<ul style="list-style-type: none"><li>1. AES128/192/256</li><li>2. DES and 3DES</li><li>3. SM2, SM3, SM4 (ASR6601SE)</li><li>4. RSA1024/2048</li><li>5. ECC224/256/384/512</li><li>6. SHA1, SHA-224, SHA256, SHA384, SHA512</li><li>7. Random number generation</li></ul>
BSTIMER	<ul style="list-style-type: none"><li>1. 32bits counter, supports auto-reload, up-counting, down-counting, center-aligned counting</li><li>2. 16-bit counter clock prescaler</li><li>3. Supports DMA requests</li><li>4. Interrupt signals generation</li></ul>
GPTIMER	<ul style="list-style-type: none"><li>1. 32 bits counter, supports auto-reload, up-counting, down-counting, center-aligned counting</li><li>2. 16-bit counter clock prescaler</li><li>3. gptimer0 and gptimer1 supports 4 channels, gptimer2 and gptimer3 supports 2 channels, each channel can support input capture, output comparison, PWM generation, single pulse output</li><li>4. Quadrature decoding</li><li>5. Interrupt signals generation</li><li>6. Supports DMA requests</li></ul>

Module Name	Functions
LPTIMER	<ol style="list-style-type: none"><li>Supports selecting internal clock and IO clock as counting clock</li><li>16 bits counter, up-counting, auto-reload</li><li>Counter clock prescaler</li><li>Quadrature decoding support</li><li>Input capture, output comparison, PWM generation, single pulse output</li><li>Interrupt signals generation</li><li>Supports DMA requests</li></ol>
ADC	<ol style="list-style-type: none"><li>12 bits sampling resolution</li><li>Configurable sampling rate up to 1 MHz</li><li>Single-ended and differential inputs</li><li>Only right data alignment</li><li>8 external channels</li><li>7 internal channels, including DAC output, internal Vref, VDD/3 (battery power), Vts internal temperature sensor), OPA output (3)</li><li>Trigger mode, supports software trigger and hardware trigger</li><li>Sequential, continuous, single, and non-continuous sampling modes</li><li>Analog watchdog function, 3 channels in total, configurable Channel selection and thresholds</li><li>Supports DMA requests</li><li>Interrupt signals generation</li></ol>
DAC	<ol style="list-style-type: none"><li>10 bits output resolution</li><li>Configurable output speed up to 1 MHz</li><li>Right data alignment only</li><li>Special waveform output, supports triangle wave</li><li>Software trigger and hardware trigger</li><li>Supports DMA requests</li><li>Interrupt signals generation</li></ol>
LCDCTRL	<ol style="list-style-type: none"><li>Frame rate control</li><li>Bias control, supports static, 1/2, 1/3, 1/4</li><li>Duty Control, supports static (1comx27seg), 1/2 (2comx26seg), 1/3 (3comx25seg), 1/4 (4comx24seg), 1/8 (8comx20seg)</li><li>Dead frame control, supports dead frame of 0-7 shots, used to adjust contrast</li><li>Blink control, supports the blinking function of 1, 2, 3, 4, 8 or all pixels, configurable blinking frequency</li><li>High and low current selection control, including state machine dynamic control and register static control. During state machine dynamic control, high current can be configured to maintain the number of beats.</li><li>Interrupt signal generation</li></ol>

# 4.

# Power Management Unit

## 4.1 Power Supply pins

ASR6601 has several separated power supply pins. Using separated power supply pins, the interference from digital parts of SoC to RF blocks is reduced.

ASR6601 Power Grid is shown in Figure 4-1:

- **VDD\_IN**: Power supply for the PA in the RF transmitter.
- **VBAT\_RF**: Power supply for the RF TRX, excluding the PA.
- **VDCC\_RF**: Low-power supply for RF TRX, must be connected to VREG pin
- **VBAT\_ESD0**: Digital IO power supply.
- **VBAT\_ESD1**: Digital IO power supply.
- **VBAT\_ESD2**: Digital IO power supply.
- **VBAT\_ESD3**: Digital IO power supply.
- **VBAT\_DCC**: Dedicated power supply for DCDC in analog circuit.
- **VBAT\_ESD\_RTC**: Power supply for IOs in RTC domain.
- **VBAT\_RTC**: Power supply for analog blocks in RTC domain.
- **VBAT\_ANA**: Power supply for analog blocks.

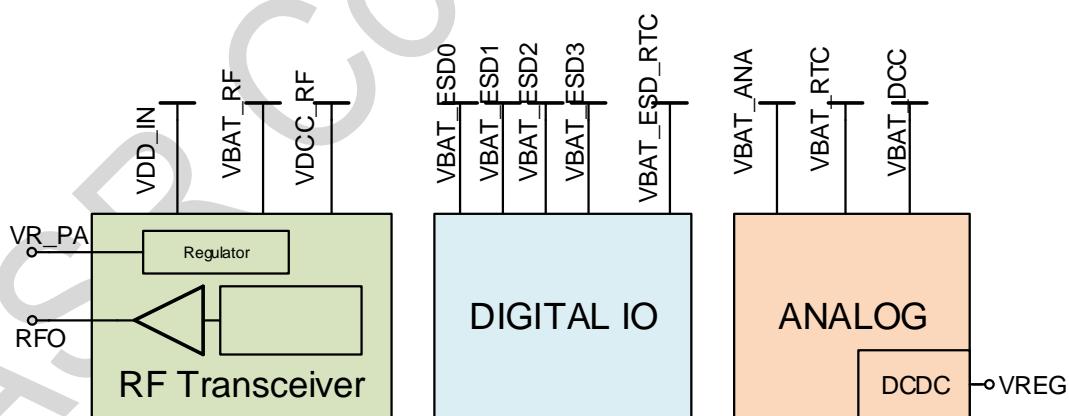


Figure 4-1 ASR6601 Power Grid

## 4.2 Power Supply Architecture

Internal power domains of the chip are mainly divided into **main** domain, **aon** domain and **aonr** domain. Please note that the power domains are divided according to functions, as shown in Figure 4-2.

1. **main** domain contains most of the digital logic circuits of the SoC chip. In the frequently used low-power modes (Standby and Stop3), the power supply of main domain will be turned off.
2. **aon** (**Always On** domain) means that the power supply for this domain is always available, even in low-power mode. Most blocks in aon domain keep running in all power modes.
3. **aonr** (**Always On and Retention**) domain contains the modules that need to keep running in Stop3 mode. These modules will be powered off in Standby mode. When aonr domain modules remain in the current state without power off, the system can quickly recover and continue to execute.

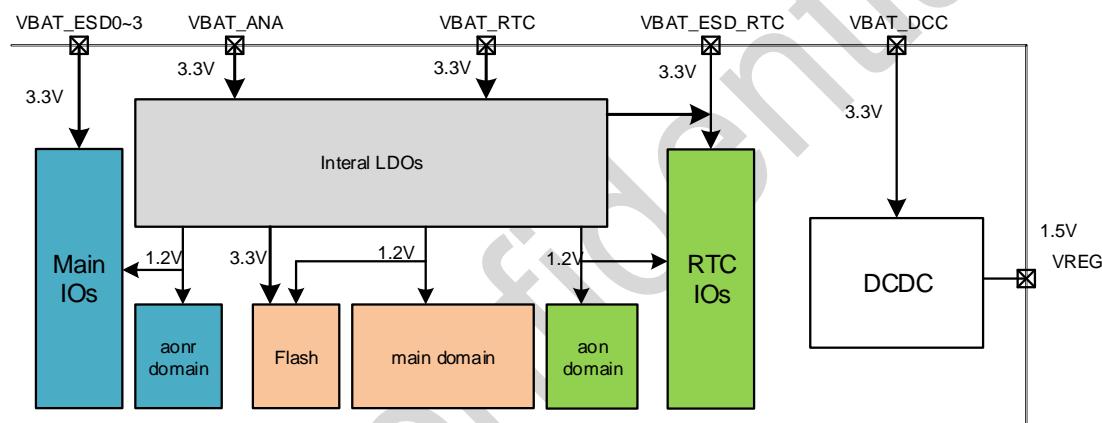


Figure 4-2 ASR6601 Power Supply Architecture

## 5.

# Access Control

## 5.1 Simple Configuration

This section provides customers with commonly used simple configurations to meet their basic security needs.

### 5.1.1 Recoverable Security Configuration

#### ● Enable Security

Configure FlashSecStart to 0 and FlashSecEnd to 0x3F in OPTION1 tab, and set the entire Flash\_main area as a secure area.

Consequently, the code in SWD (Serial Wire Debug) and non-secure area cannot read and write data into Flash\_main to guarantee security. Please note that code in non-secure SRAM area or non-secure DMA will not be able to access Flash\_main.

#### ● Disable Security

Configure FlashSecStart to 0x3F and FlashSecEnd to 0 in OPTION1 tab, and set the entire Flash\_main area as a non-secure area.

The above configurations will erase the entire Flash\_main area, and then the program can be re-downloaded.

### 5.1.2 Unrecoverable Security Configuration

Configure the DebugLevel to 2 in Option0 tab. This operation is irreversible, and the code must be correct and mature.

## 5.2 Access Control

Based on debug level rules, boot startup mode, exe-only access rules, write-protected access rules, info area access rules, and secure area access rules, access rights from the four main interfaces (cpucode, cpusw, dmac0, and dmac1) are controlled.

### 5.2.1 Debug Level Rules

Debug level mainly affects the access rights of cpu\_code (boot from SRAM and boot from bootloader), cpu\_sw, dmac0 and dmac1 to sensitive areas. Sensitive areas include flash\_main, otp area of flash\_info and retention SRAM.

For details see "[ASR6601 Access Control Description](#)".

## 5.2.2 Secure and Non-Secure Operation

- **Secure Operation**

The operations initiated by the code in the secure area include:

- ◆ Operations initiated by DMAC0 configured as a secure area
- ◆ Operations initiated by flash\_main configured as a secure area (CPU\_Code)
- ◆ Operations initiated by system\_sram configured as a secure area (CPU\_Code)

- **Non-secure Operation**

The operations initiated by the code in the non-secure area include:

- ◆ Operations initiated by DMAC0 configured as a non-secure area
- ◆ Operations initiated by DMAC1
- ◆ Operations initiated by Debug Port (CPU\_SW)
- ◆ Operations initiated by Bootloader (CPU\_Code)
- ◆ Operations initiated by flash\_main configured as a non-secure area (CPU\_Code)
- ◆ Operations initiated by system\_sram configured as a non-secure area (CPU\_Code)

For details see "[ASR6601 Access Control Description](#)".

## 6.

# Operation Modes

ASR6601 LPWAN SoC supports Run, LpRun, Sleep, LpSleep, Stop0, Stop1, Stop2, Stop3 and Standby modes. Each mode supports different functions with different working modules and power consumption. The user can choose the appropriate operation mode according to specific application scenarios. All modes are described detailedly in the contents below.

In addition, please note the following points:

1. When entering low power mode, peripherals marked as O (excluding GPIO) are turned off by default. Functions required in low power mode need to be turned on before entering low power mode.
2. When entering low power mode, in order to achieve the corresponding design power consumption value, please pay attention to the following points:
  - (1) Unused GPIOs need to be configured in ANALOG mode (high impedance).
  - (2) If the GPIO used is in input mode, you need to configure the pull-up and pull-down.
  - (3) If the peripheral is in output mode, the pull-up and pull-down of the connected peripheral must be configured according to the output level.
3. RCO48M/2 is used to enter and exit low power mode. If a non-RCO48M/2 clock is used before entering low power mode, it is necessary to switch to RCO48M/2. After exiting low power mode, you can switch to the previously used clock.
4. If the analog functions supported by RCO32K/XO32K and other low power modes are needed in low power mode, they need to be enabled before entering low power mode.
5. The clocks other than RCO48M/RCO32K/XO32K and other analog function modules need to be turned off by software before entering low power consumption.

**Table 6-1 Status of different modules in each operating mode**

	Run	LpRun	Sleep	LpSleep	Stop0	Stop1	Stop2	Stop3	Standby	Standby Wakeup	Stop3 Wakeup	Stop0-2 Wakeup
cpu	Y	Y	NA	NA	NA	NA	NA	NA	NA			
efc	Y	Y	O	O	NA	NA	NA	NA	NA			
sysramc	Y	Y	O	O	NA	NA	NA	NA	NA			
retramc	Y	Y	O	O	NA	NA	NA	NA	NA			
i2s	O	O	O	O	NA	NA	NA	NA	NA			
uart0	O	O	O	O	NA	NA	NA	NA	NA			
uart1	O	O	O	O	NA	NA	NA	NA	NA			
uart2	O	O	O	O	NA	NA	NA	NA	NA			
uart3	O	O	O	O	NA	NA	NA	NA	NA			
ssp0	O	O	O	O	NA	NA	NA	NA	NA			
ssp1	O	O	O	O	NA	NA	NA	NA	NA			
ssp2	O	O	O	O	NA	NA	NA	NA	NA			
qspi	O	O	O	O	NA	NA	NA	NA	NA			
i2c0	O	O	O	O	NA	NA	NA	NA	NA			

	Run	LpRun	Sleep	LpSleep	Stop0	Stop1	Stop2	Stop3	Standby	Standby Wakeup
i2c1	O	O	O	O	NA	NA	NA	NA	NA	
i2c2	O	O	O	O	NA	NA	NA	NA	NA	
adcctrl	O	O	O	O	NA	NA	NA	NA	NA	
dacctrl	O	O	O	O	NA	NA	NA	NA	NA	
gptim0	O	O	O	O	NA	NA	NA	NA	NA	
gptim1	O	O	O	O	NA	NA	NA	NA	NA	
gptim2	O	O	O	O	NA	NA	NA	NA	NA	
gptim3	O	O	O	O	NA	NA	NA	NA	NA	
basictim0	O	O	O	O	NA	NA	NA	NA	NA	
basictim1	O	O	O	O	NA	NA	NA	NA	NA	
wwdg	O	O	O	O	NA	NA	NA	NA	NA	
crc	O	O	O	O	NA	NA	NA	NA	NA	
sec	O	O	O	O	NA	NA	NA	NA	NA	
sac	O	O	O	O	NA	NA	NA	NA	NA	
mpu	O	O	O	O	NA	NA	NA	NA	NA	
dmac0	O	O	O	O	NA	NA	NA	NA	NA	
dmac1	O	O	O	O	NA	NA	NA	NA	NA	
syscfg	O	O	O	O	NA	NA	NA	NA	NA	
afec	O	O	O	O	NA	NA	NA	NA	NA	
lorac	O	O	O	O	NA	NA	NA	NA	NA	
gpio	O	O	O	O	NA	NA	NA	GPIO0~55: Y3 GPIO56~63: Y4	GPIO0~55: NA3 GPIO56~63: Y4	Y
rcc	Y	Y	Y	Y	Y	Y	Y	Y	Y	
pwr	Y	Y	Y	Y	Y	Y	Y	Y	Y	
lpuart	O	O	O	O	O	O	O	O (RX only)	O (RX only)	Y
lcdctrl	O	O	O	O	O	O	O	O	O	
lptim0	O	O	O	O	O	O	O	O	O	Y
lptim1	O	O	O	O	O	O	O	O	O	Y
iwdg	O	O	O	O	O	O	O	O	O	Y1
rtc	O	O	O	O	O	O	O	O	O	Y
ADC	O	O	O	O	NA	NA	NA	NA	NA	
RCO48M	O	O	O	O	NA	NA	NA	NA	NA	
XO24M	O	O	O	O	NA	NA	NA	NA	NA	
PLL48M	O	O	O	O	NA	NA	NA	NA	NA	
RNG	O	O	O	O	NA	NA	NA	NA	NA	
DAC	O	O	O	O	O3	O3	O3	NA	NA	
OPA	O	O	O	O	O	O	O	NA	NA	
COMP	O	O	O	O	O	O	O	O	Y	Y
VD	O	O	O	O	O	O	O	O	Y	Y
RCO3.6M	O	O	O	O	O	O	O	O	O	

	Run	LpRun	Sleep	LpSleep	Stop0	Stop1	Stop2	Stop3	Standby	Standby Wakeup
RCO32K	O	O	O	O	O	O	O	O	O	
XO32K	O	O	O	O	O	O	O	O	O	
LCD	O	O	O	O	O	O	O	O	O	
BOR	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y2
FLASH	Y	Y	Y	Y	SLM	SLM	SLM	PDM	PDM	
SRAM	Y	Y	Y	Y	NA	NA	NA	NA1	NA2	
IO	Y	Y	Y	Y	Y	Y	Y	Y	Y	
RF	O	O	O	O	O	O	O	O	O	Y

Notes and symbol annotations for the table above :

- **Stop0-2:** all GPIOs can be configured to wake up the CPU; all GPIOs retain the previous state in Stop0-2 mode.
- **Stop3:** 56 GPIOs in the main domain can be configured to wake up the CPU; all GPIOs retain the previous state in Stop3 mode.
- **Standby:** 8 GPIOs in the AON domain retain the previous state in Standby mode; 56 GPIOs in the main domain are used as analog functions (such as LCD, COMP) and cannot be used to wake up the CPU. The LPUART only supports RX in Standby/Stop3 mode.
- **Y:** Normal work
- **O:** Optional, configured by software
- **O3:** Data update is not supported, but the output retains current voltage level
- **Y1:** Generate system reset to wake up the system indirectly
- **Y2:** Generate BOR reset to wake up the system indirectly
- **Y3:** Retain the state before entering low-power mode, and can be used to wake up the CPU.
- **Y4:** MUX Function1 of GPIO56~63 is not available and the other alternate functions is available
- **NA1:** Retention and algorithm contents are kept. System content can be configured to be kept or not.
- **NA2:** Retention content is kept
- **NA3:** Analog Output Only

## 6.1 Run

### 6.1.1 Enter and Exit

**Run** mode is the default operation mode after power-on or system reset.

ASR6601 can enter Sleep, LpRun, Stop0, Stop1, Stop2, Stop3 or Standby mode from Run mode.

ASR6601 can return to Run mode from Sleep, LpRun, Stop0, Stop1, Stop2, Stop3 or Standby mode.

For detailed mode switching conditions, please refer to the descriptions of other operation modes.

### 6.1.2 Wakeup Source

N/A

## 6.2 LpRun

### 6.2.1 Enter and Exit

You can enter LpRun from Run. The entry conditions are as follows: the software switches the working state of LDO. Before switching LDO, all high-speed clocks must be turned off and the CPU runs at 32K clock.

**LpRun** config register is used to switch LDO working state:

- 1: Set bits[3:3] of the register (address 0x05) to 1, and the other bits remain unchanged.
- 2: Set bits[21:20] of the register (address 0x06) to 1, and the other bits remain unchanged.

**LpRun** can return to **Run**, and the exit conditions are as follows: the software switches the working state of LDO. The high-speed clock can be turned on only after the switch is completed.

Return to **Run** mode from **LpRun** mode in the following way:

- 1: Clear bits[21:20] of the register (address 0x06) to 0, and the other bits remain unchanged.
- 2: Clear bits[3:3] of the register (address 0x05) to 0, and the other bits remain unchanged.

### 6.2.2 Wakeup Source

N/A

## 6.3 Sleep

### 6.3.1 Enter and Exit

Sleep can be entered from Run. The entry conditions are: CPU executes wfi/wfe instruction (SLEEPDEEP=0), or isr returns (SLEEPONEXIT=1 and SLEEPDEEP=0).

It can return to Run from Sleep, and the exit condition is: if wfi is entered, it supports interrupt wake-up, and if wfe is entered, it supports event wake-up.

**Note:** Because there is no dedicated event wake-up signal, SVONPEND=1 is used and the corresponding NVIC is turned off to achieve it. At this time, the interrupt signal is used for event wake-up.

### 6.3.2 Wakeup Source

Interrupt signals

## 6.4 LpSleep

### 6.4.1 Enter and Exit

LpSleep can be entered from LpRun, and the entry conditions are: CPU executes wfi/wfe instruction (SLEEPDEEP=0), or isr returns (SLEEPONEXIT=1 and SLEEPDEEP=0).

LpSleep can return to LpRun, and the exit conditions are: if wfi enters, interrupt wake-up is supported; if wfe enters, event wake-up is supported.

**Note:** There is no dedicated event wake-up signal. It is implemented by setting SVONPEND=1 and turning off the corresponding NVIC. In this case, the interrupt signal is used for event wake-up.

### 6.4.2 Wakeup Source

Interrupt signals

## 6.5 Stop0

### 6.5.1 Enter and Exit

Stop0 can be entered from Run. The entry conditions are: configure lp\_mode to 2'b00, CPU executes wfi/wfe instruction (SLEEPDEEP=1), or isr returns (SLEEPONEXIT=1 and SLEEPDEEP=1).

Stop0 can be used to return to Run. The exit conditions are: if wfi is entered, interrupt wake-up is supported; if wfe is entered, event wake-up is supported.

The pwr module summarizes the wakeup source status and outputs the pwr\_wakeup\_int interrupt signal and the pwr\_wakeup\_event event signal to the CPU for wakeup.

### 6.5.2 Wakeup Source

- GPIO00-GPIO63 can be used for wake-up. Four IOs form a group. Each IO in a group has a wake-up enable configuration. A group can generate a wake-up signal. Each IO in a group supports the selection of high-level wake-up or low-level wake-up. The wake-up sources other than GPIO are listed below.
- PVM Alarm
- VD Alarm
- TD Alarm
- LD Alarm
- Comparator
- LPTIM0/1
- FD\_32K Alarm
- Wakeup/Tamper IO
- RTC Alarm
- RTC CYC Timer
- LPUART RX Status
- LORA BUSY
- LORA IRQ

## 6.6 Stop1

### 6.6.1 Enter and Exit

You can enter Stop1 from Run. The entry conditions are: configure lp\_mode to 2'b01, CPU executes wfi/wfe instruction (SLEEPDEEP=1), or isr returns (SLEEPONEXIT=1 and SLEEPDEEP=1);

Stop1 can be used to return to Run. The exit conditions are: if wfi is entered, interrupt wake-up is supported; if wfe is entered, event wake-up is supported.

The pwr module summarizes the wakeup source status and outputs the pwr\_wakeup\_int interrupt signal and the pwr\_wakeup\_event event signal to the CPU for wakeup.

## 6.6.2 Wakeup Source

- GPIO00-GPIO63 can be used for wake-up. Four IOs form a group. Each IO in a group has a wake-up enable configuration. A group can generate a wake-up signal. Each IO in a group supports the selection of high-level wake-up or low-level wake-up. The wake-up sources other than GPIO are listed below.
- PVM Alarm
- VD Alarm
- TD Alarm
- LD Alarm
- Comparator
- LPTIM0/1
- FD\_32K Alarm
- Wakeup/Tamper IO
- RTC Alarm
- RTC CYC Timer
- LPUART RX Status
- LORA BUSY
- LORA IRQ

## 6.7 Stop2

### 6.7.1 Enter and Exit

You can enter Stop2 from Run. The entry conditions are: configure lp\_mode to 2'b10, CPU executes wfi/wfe instruction (SLEEPDEEP=1), or isr returns (SLEEPONEXIT=1 and SLEEPDEEP=1);

Stop2 can be used to return to Run. The exit conditions are: if wfi is entered, interrupt wake-up is supported; if wfe is entered, event wake-up is supported;

The pwr module summarizes the wakeup source status and outputs the pwr\_wakeup\_int interrupt signal and the pwr\_wakeup\_event event signal to the CPU for wakeup.

### 6.7.2 Wakeup Source

- GPIO00-GPIO63 can be used for wake-up. Four IOs form a group. Each IO in a group has a wake-up enable configuration. A group can generate a wake-up signal. Each IO in a group supports the selection of high-level wake-up or low-level wake-up. The wake-up sources other than GPIO are listed below.
- PVM Alarm
- VD Alarm
- TD Alarm
- LD Alarm
- Comparator
- LPTIM0/1
- FD\_32K Alarm
- Wakeup/Tamper IO
- RTC Alarm
- RTC CYC Timer
- LPUART RX Status
- LORA BUSY
- LORA IRQ

## 6.8 Stop3

### 6.8.1 Enter and Exit

You can enter Stop3 from Run. The entry conditions are: configure lp\_mode to 2'b11, lp\_mode\_ext to 1'b1, CPU executes wfi/wfe instruction (SLEEPDEEP=1), or isr returns (SLEEPONEXIT=1 and SLEEPDEEP=1);

You can return to Run from Stop3 when a Stop3 wake-up event occurs.

### 6.8.2 Wakeup Source

- GPIO00-GPIO55 can all be used to wake up the CPU, 4 IOs make up a group, and each group can select any of the 4 IOs for wake-up. A group generates a wake-up signal, and any of the IOs can wake up the CPU at high or low level. The wake-up sources other than GPIOs are listed below.
- PVM Alarm
- VD Alarm
- Comparator
- LPTIM0/1
- FD\_32K Alarm
- Wakeup/Tamper IO
- RTC Alarm
- RTC CYC Timer
- LPUART RX Status
- LORA BUSY
- LORA IRQ
- IWDG Timeout

## 6.9 Standby

### 6.9.1 Enter and Exit

The Standby state can be entered from Run state. The entry conditions are: configure lp\_mode to 2'b11, lp\_mode\_ext to 1'b0, CPU executes wfi/wfe instruction (SLEEPDEEP=1), or isr returns (SLEEPONEXIT=1 and SLEEPDEEP=1);

The system can return to Run from Standby. The exit condition is: a Standby wake-up event occurs.

**Note:**

1. When the power supply is switched between DCDC and VBAT, the CPU will return to Run mode immediately after entering Standby mode without any wake-up event.
2. When dbg\_standby=1, the switch between DCDC and VBAT is disabled.

### 6.9.2 Wakeup Source

- PVM Alarm
- VD Alarm
- Comparator
- LPTIM0/1
- FD\_32K Alarm
- Wakeup/Tamper IO
- RTC Alarm
- RTC CYC Timer
- LPUART RX Status
- LORA BUSY
- LORA IRQ
- IWDG Timeout

# 7.

# System configuration

## 7.1 System Architecture

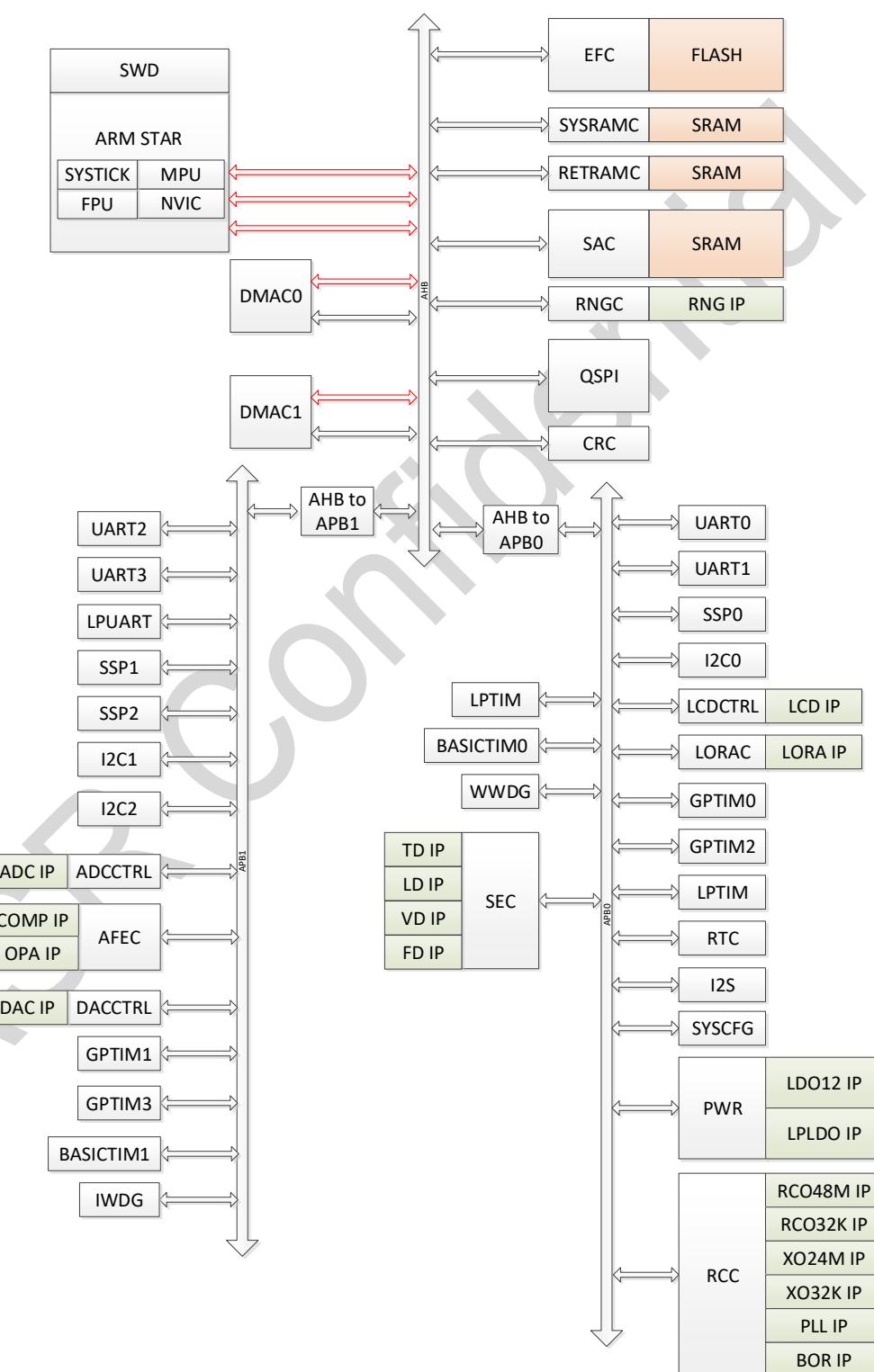


Figure 7-1 System Architecture Diagram

## 7.1.1 Arm China STAR-MC1 Processor

Arm China STAR-MC1 Processor consists of three master buses, including iicode AHB bus, dcode AHB bus and system AHB bus, which are used for program access, data access and register access.

### 7.1.2 DMAC0

DMAC0 has a master bus, which can assist the CPU to transfer data.

### 7.1.3 DMAC1

DMAC1 has a master bus, which can assist the CPU to transfer data.

### 7.1.4 Master

The addresses accessible by each master bus is shown in the table below.

(1) Only accessible when boot from Bootloader.

Table 7-1 Master bus access range

Start Address	End Address	Description	Executable	iicode Access	dcode Access	system Access	DMAC0 Access	DMAC1 Access
0xE0100000	0xFFFFFFFF	Reserved						
0xE0000000	0xE00FFFFF	ARM STAR peripherals						
0xA0000000	0xDFFFFFFF	Reserved						
0x70000000	0x9FFFFFFF	Reserved						
0x60000000	0x6FFFFFFF	Qspi Flash Bank	Y			Y	Y	Y
0x50000000	0x5FFFFFFF	Reserved						
0x40030000	0x4FFFFFFF	AHB1 SFR				Y	Y	Y
0x40020000	0x4002FFFF	AHB0 SFR				Y	Y	Y
0x40010000	0x4001FFFF	APB1 SFR				Y	Y	Y
0x40000000	0x4000FFFF	APB0 SFR				Y	Y	Y
0x30000400	0x3FFFFFFF	Reserved						
0x30000000	0x300003FF	Retention SRAM				Y	Y	Y
0x20010000	0x2FFFFFFF	Reserved						
0x20000000	0x2000FFFF	System SRAM	Y			Y	Y	Y
0x18010000	0x1FFFFFFF	Reserved						
0x18000000	0x1800FFFF	System SRAM	Y	Y	Y			
0x10004000	0x17FFFFFF	Reserved						
0x10003000	0x10003FFF	Option Bytes				Y		
0x10002000	0x10002FFF	Factory Bytes				Y		
0x10001C00	0x10001FFF	OTP				Y		
0x10000000	0x10001BFF	BootLoader		Y <sup>(1)</sup>		Y <sup>(1)</sup>		
0x08040000	0x0FFFFFFF	Reserved						
0x08000000	0x0803FFFF	Main Flash	Y	Y	Y		Y	Y
0x00040000	0x07FFFFFF	Reserved						
0x00000000	0x0003FFFF	Main Flash/BootLoader/ System SRAM <sup>(1)</sup>	Y	Y	Y			

## 7.2 Memory Mapping

The Memory Mapping table is shown below. The bytes are coded in memory in Little Endian format, i.e. the least significant byte is in the lowest address.

**Table 7-2 Memory map**

Category	Start Address	End Address	Description	Size
SYSTEM	0xE0100000	0xFFFFFFFFFF	Reserved	
PPB	0xE0000000	0xE00FFFFF	ARM STAR peripherals	
EXT PERIPHERAL	0xA0000000	0xDFFFFFFF	Reserved	
EXT SRAM	0x70000000	0x9FFFFFFF	Reserved	
	0x60000000	0x6FFFFFFF	Qspi Flash Bank	256MB
PERIPHERAL	0x50000000	0x5FFFFFFF	Reserved	
	0x40030000	0x4FFFFFFF	AHB1 SFR	
	0x40020000	0x4002FFFF	AHB0 SFR	
	0x40010000	0x4001FFFF	APB1 SFR	
	0x40000000	0x4000FFFF	APB0 SFR	
SRAM	0x30000400	0x3FFFFFFF	Reserved	
	0x30000000	0x300003FF	Retention SRAM	1KB
	0x20010000	0x2FFFFFFF	Reserved	
	0x20000000	0x2000FFFF	System SRAM	64KB
CODE	0x18010000	0x1FFFFFFF	Reserved	
	0x18000000	0x1800FFFF	System SRAM	64KB
	0x10004000	0x17FFFFFF	Reserved	
	0x10003000	0x10003FFF	Option Bytes	4KB
	0x10002000	0x10002FFF	Factory Bytes	4KB
	0x10001C00	0x10001FFF	OTP	1KB
	0x10000000	0x10001BFF	BootLoader	7KB
	0x08040000	0x0FFFFFFF	Reserved	
	0x08000000	0x0803FFFF	Flash Main	256KB
	0x00040000	0x07FFFFFF	Reserved	
	0x00000000	0x0003FFFF	Flash Main/BootLoader/ System SRAM <sup>(1)</sup>	256KB

<sup>(1)</sup> The memory corresponding to address 0x00000000 is determined by the boot mode.

## 7.2.1 AHB0 SFR

See the table below for AHB0 SFR Internal Address Mapping.

**Table 7-3 AHB0 SFR address mapping**

Start Address	End Address	Description	Size
0x40025000	0x4002FFFF	Reserved	
0x40024000	0x40024FFF	DMAC1	4KB
0x40023000	0x40023FFF	DMAC0	4KB
0x40022000	0x40022FFF	CRC	4KB
0x40021000	0x40021FFF	QSPI	4KB
0x40020000	0x40020FFF	EFC	4KB

## 7.2.2 AHB1 SFR

See the table below for AHB1 SFR Internal Address Mapping.

**Table 7-4 AHB1 SFR address mapping**

Start Address	End Address	Description	Size
0x40034000	0x4003FFFF	Reserved	
0x40033000	0x40033FFF	RNGC	4KB
0x40030000	0x40032FFF	SAC	12KB <sup>(1)(2)</sup>

<sup>(1)</sup> Low 8KB is ARAM space, and high 4KB is for registers.

<sup>(2)</sup> ARAM space can only be accessed in word.

## 7.2.3 APB0 SFR

See the table below for APB0 SFR Internal Address Mapping.

**Table 7-5 APB0 SFR address mapping**

Start Address	End Address	Description	Size
0x4000F000	0x4000FFFF	SEC	4KB
0x4000E000	0x4000EFFF	RTC	4KB
0x4000D800	0x4000DFFF	LPTIM1	2KB
0x4000D000	0x4000D7FF	LPTIM0	2KB
0x4000C000	0x4000CFFF	BASICTIM0	4KB
0x4000B000	0x4000BFFF	GPTIM2	4KB
0x4000A000	0x4000AFFF	GPTIM0	4KB
0x40009000	0x40009FFF	LORAC	4KB
0x40008000	0x40008FFF	AFEC	4KB

Start Address	End Address	Description	Size
0x40007000	0x40007FFF	I2C0	4KB
0x40006000	0x40006FFF	SSP0	4KB
0x40005000	0x40005FFF	LPUART	4KB
0x40004000	0x40004FFF	UART1	4KB
0x40003000	0x40003FFF	UART0	4KB
0x40002000	0x40002FFF	I2S	4KB
0x40001800	0x40001FFF	PWR	2KB
0x40001000	0x400017FF	SYSCFG	2KB
0x40000000	0x40000FFF	RCC	4KB

## 7.2.4 APB1 SFR

See the table below for APB1 SFR Internal Address Mapping.

**Table 7-6 APB1 SFR address mapping**

Start Address	End Address	Description	Size
0x4001FC00	0x4001FFFF	PortD	1KB
0x4001F800	0x4001FBFF	PortC	1KB
0x4001F400	0x4001F7FF	PortB	1KB
0x4001F000	0x4001F3FF	PortA	1KB
0x4001E000	0x4001EFFF	WWDG	4KB
0x4001D000	0x4001DFFF	IWDG	4KB
0x4001C000	0x4001CFFF	BASICTIM1	4KB
0x4001B000	0x4001BFFF	GPTIM3	4KB
0x4001A000	0x4001AFFF	GPTIM1	4KB
0x40019000	0x40019FFF	DACCTRL	4KB
0x40018000	0x40018FFF	LCDCTRL	4KB
0x40017000	0x40017FFF	ADCCTRL	4KB
0x40016000	0x40016FFF	Reserved	4KB
0x40015000	0x40015FFF	I2C2	4KB
0x40014000	0x40014FFF	I2C1	4KB
0x40013000	0x40013FFF	SSP2	4KB
0x40012000	0x40012FFF	SSP1	4KB
0x40011000	0x40011FFF	UART3	4KB
0x40010000	0x40010FFF	UART2	4KB

## 7.3 SRAM

The SRAM in ASR6601 includes system SRAM, retention SRAM and SAC SRAM. SAC SRAM only supports word access, and system SRAM and retention SRAM support word, halfword, and byte access.

## 7.4 Boot Modes

The boot mode can be configured by the levels of BOOT0 pin (GPIO02) and the data in the Flash.

**Table 7-7 ASR6601 Boot Mode Configuration**

DEBUG_LEVEL	USE_FLASH_BOOT0	FLASH_BOOT0	BOOT0 PIN	FLASH_BOOT1	MAIN_FLASH_EMPTY	Boot Config
2	X	X	X	X	X	Boot from Flash Main
<2	0	X	0	X	0	Boot from Flash Main
<2	0	X	0	X	1	Boot from Flash Bootloader
<2	0	X	1	1	X	Boot from Flash Bootloader
<2	0	X	1	0	X	Boot from System SRAM
<2	1	1	X	X	0	Boot from Flash Main
<2	1	1	X	X	1	Boot from Flash Bootloader
<2	1	0	X	1	X	Boot from Flash Bootloader
<2	1	0	X	0	X	Boot from System SRAM

DebugLevel, UseFlashBoot0, FlashBoot0 and FlashBoot1 is the information area of the Flash, they can be modified according to the application. MainFlashEmpty is determined by the data of address 0 in the Flash Main area. If the data in the address 0 of Flash Main area is 0xFFFFFFFF, the value of MainFlashEmpty is 1, otherwise the value of MainFlashEmpty is 0. BOOT0 pin is GPIO02 in the package.

The boot mode is selected according to the configurations when the system is in these status: first powered up, exit the Standby mode or reset.

## 7.5 SYSCFG Registers

Base Address: 0x40001000

**Table 7-8 SYSCFG Registers Summary**

Register	Offset	Description
SYSCFG_CR0	0x000	Control Register 0, DMA handshake
SYSCFG_CR1	0x004	Control Register 1, DMA handshake
SYSCFG_CR2	0x008	Control Register 2
SYSCFG_CR3	0x00C	Control Register 3, Low power debug connection control
SYSCFG_CR4	0x010	Control Register 4
SYSCFG_CR5	0x014	Control Register 5
SYSCFG_CR6	0x018	Control Register 6,secure lock control
SYSCFG_CR7	0x01C	Control Register 7,secure lock control
SYSCFG_CR8	0x020	Control Register 8, QSPI memory encryption key
SYSCFG_CR9	0x024	Control Register 9, QSPI REMAP
SYSCFG_CR10	0x028	Control Register 10

### 7.5.1 SYSCFG\_CR0

Offset: 0x000

Reset Value: 0x00000000

31-30	29-24	23-22	21-16
RESERVED	DMAC0_HANDSHAKE0_SEL	RESERVED	DMAC0_HANDSHAKE1_SEL
r	r/w	r	r/w
<b>15-14</b>	<b>13-8</b>	<b>7-6</b>	<b>5-0</b>
RESERVED	DMAC0_HANDSHAKE2_SEL	RESERVED	DMAC0_HANDSHAKE3_SEL
r	r	r	r/w

**Bits 31-30 RESERVED:** Must be kept, and can't be modified.

**Bits 29-24 DMAC0\_HANDSHAKE0\_SEL:** DMAC0 HANDSHAKE0 selection. For details, please refer to [Table7-9 DMA Request MUX](#).

**Bits 23-22 RESERVED:** Must be kept, and can't be modified.

**Bits 21-16 DMAC0\_HANDSHAKE1\_SEL:** DMAC0 HANDSHAKE1 selection. For details, please refer to [Table7-9 DMA Request MUX](#).

**Bits 15-14 RESERVED:** Must be kept, and cannot be modified.

**Bits 13-8 DMAC0\_HANDSHAKE2\_SEL:** DMAC0 HANDSHAKE2 selection. For details, please refer to [Table7-9 DMA Request MUX](#).

**Bits 7-6 RESERVED:** Must be kept, and cannot be modified.

**Bits 5-0 DMAC0\_HANDSHAKE3\_SEL:** DMAC0 HANDSHAKE3 selection. For details, please refer to [Table7-9 DMA Request MUX](#).

### 7.5.2 SYSCFG\_CR1

Offset: 0x004

Reset Value: 0x00000000

31-30	29-24	23-22	21-16
RESERVED	DMAC1_HANDSHAKE0_SEL	RESERVED	DMAC1_HANDSHAKE1_SEL
r	r/w	r	r/w
<b>15-14</b>	<b>13-8</b>	<b>7-6</b>	<b>5-0</b>
RESERVED	DMAC1_HANDSHAKE2_SEL	RESERVED	DMAC1_HANDSHAKE3_SEL
r	r	r	r/w

**Bits 31-30 RESERVED:** Must be kept, and cannot be modified.

**Bits 29-24 DMAC1\_HANDSHAKE0\_SEL:** DMAC1 HANDSHAKE0 selection. For details, please refer to [Table7-9 DMA Request MUX](#).

**Bits 23-22 RESERVED:** Must be kept, and cannot be modified.

**Bits 21-16 DMAC1\_HANDSHAKE1\_SEL:** DMAC1 HANDSHAKE1 selection. For details, please

refer to [Table7-9 DMA Request MUX](#).

**Bits 15-14 RESERVED:** Must be kept, and cannot be modified.

**Bits 13-8 DMAC1\_HANDSHAKE2\_SEL:** DMAC1 HANDSHAKE2 selection. For details, please refer to [Table7-9 DMA Request MUX](#).

**Bits 7-6 RESERVED:** Must be kept, and cannot be modified.

**Bits 5-0 DMAC1\_HANDSHAKE3\_SEL:** DMAC1 HANDSHAKE3 selection. For details, please refer to [Table7-9 DMA Request MUX](#).

### 7.5.3 SYSCFG\_CR2

Offset: 0x008

Reset Value: 0x00000000

31	30	29-28	27
RESERVED	SYSCFG_HALTED_IPTI M1_EN	RESERVED	SYSCFG_HALTED_LPT IM0_EN
r	r/w	r	r/w
26	25	24	23
SYSCFG_HALTED_IW DG_EN	SYSCFG_HALTED_WW DG_EN	SYSCFG_HALTED_GP TIM0_EN	SYSCFG_HALTED_GP TIM1_EN
r/w	r/w	r/w	r/w
22	21	20	19
SYSCFG_HALTED_GP TIM2_EN	SYSCFG_HALTED_GP TIM3_EN	SYSCFG_HALTED_BA SICTIM0_EN	SYSCFG_HALTED_BA SICTIM1_EN
r/w	r/w	r/w	r/w
18	17	16-12	
QSPI_MEM_ENCRYPT _EN	QSPI_REMAP_ENABLE	RESERVED	
r/w	r/w	r	
11	10	9-8	
CPU_STCALIB_SKEW	SYSCFG_DBG_SLEEP	RESERVED	
r/w	r/w	r	
7	6	5	4
UART0_DMA_CLR_SEL	UART1_DMA_CLR_SEL	UART2_DMA_CLR_SEL	UART3_DMA_CLR_SEL
r/w	r/w	r/w	r/w
3	2	1	0
SSP0_DMA_CLR_SEL	SSP1_DMA_CLR_SEL	SSP2_DMA_CLR_SEL	SSP_AFEC_DMA_CLR _SEL
r/w	r/w	r/w	r/w

**Bit 31 RESERVED:** Must be kept, and cannot be modified.

**Bit 30 SYSCFG\_HALTED\_LPTIM1\_EN:** Stop LPTIM1 counter if the core is halted

- 0: LPTIM1 counter continues to work normally when the core is halted

- 1: LPTIM1 counter is stopped when the core is halted

**Bits 29-28 RESERVED:** Must be kept, and cannot be modified.

**Bit 27 SYSCFG\_HALTED\_LPTIM0\_EN:** Stop LPTIM0 counter if the core is halted

- 0: LPTIM0 counter continues to work normally when the core is halted
- 1: LPTIM0 counter is stopped when the core is halted

**Bit 26 SYSCFG\_HALTED\_IWDG\_EN:** Stop independent watchdog counter if the core is halted

- 0: the independent watchdog counter continues to work normally when the core is halted
- 1: the independent watchdog counter is stopped when the core is halted

**Bit 25 SYSCFG\_HALTED\_WWDG\_EN:** Stop window watchdog counter if the core is halted

- 0: The window watchdog counter continues to work normally when the core is halted
- 1: The window watchdog counter is stopped when the core is halted

**Bit 24 SYSCFG\_HALTED\_GPTIM0\_EN:** Stop GPTIM0 counter if the core is halted

- 0: GPTIM0 counter continues to work normally when the core is halted
- 1: GPTIM0 counter is stopped when the core is halted

**Bit 23 SYSCFG\_HALTED\_GPTIM1\_EN:** Stop GPTIM1 counter if the core is halted

- 0: GPTIM1 counter continues to work normally when the core is halted
- 1: GPTIM1 counter is stopped when the core is halted

**Bit 22 SYSCFG\_HALTED\_GPTIM2\_EN:** Stop GPTIM2 counter if the core is halted

- 0: GPTIM2 counter continues to work normally when the core is halted
- 1: GPTIM2 counter is stopped when the core is halted

**Bit 21 SYSCFG\_HALTED\_GPTIM3\_EN:** Stop GPTIM3 counter if the core is halted

- 0: GPTIM3 counter continues to work normally when the core is halted
- 1: GPTIM3 counter is stopped when the core is halted

**Bit 20 SYSCFG\_HALTED\_BASICTIM0\_EN:** Stop BASICTIM0 counter if the core is halted

- 0: BASICTIM0 counter continues to work normally when the core is halted
- 1: BASICTIM0 counter is stopped when the core is halted

**Bit 19 SYSCFG\_HALTED\_BASICTIM1\_EN:** Stop BASICTIM1 counter if the core is halted

- 0: BASICTIM1 counter continues to work normally when the core is halted
- 1: BASICTIM1 counter is stopped when the core is halted

**Bit 18 QSPI\_MEM\_ENCRYPT\_EN:** QSPI memory encryption enable

- 0: disabled
- 1: enabled

**Bit 17 QSPI\_REMAP\_ENABLE:** QSPI remap function enable

- 0: disabled
- 1: enabled

**Bits 16-12 RESERVED:** Must be kept, and cannot be modified.

**Bit 11 CPU\_STCALIB\_SKEW:** CPU SysTick skew configuration. Affects STCALIB[24] bit.

- 0: disable
- 1: enable

**Bit 10 SYSCFG\_DBG\_SLEEP:** Allow debug connection in Deepsleep mode

It is only used in debug mode and it will affect the Deepsleep mode.

- 0: not allowed
- 1: allowed

**Bits 9-8 RESERVED:** Must be kept, and cannot be modified.

**Bit 7 UART0\_DMA\_CLR\_SEL:** UART0 DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. UART module uses the synchronized DMA\_CLR signal by default.

- 0: use the DMA\_CLR signal after 2 cycles
- 1: directly use the DMA\_CLR signal output by DMAC

**Bit 6 UART1\_DMA\_CLR\_SEL:** UART1 DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. UART module uses the synchronized DMA\_CLR signal by default.

- 0: use the DMA\_CLR signal after 2 cycles
- 1: directly use the DMA\_CLR signal output by DMAC

**Bit 5 UART2\_DMA\_CLR\_SEL:** UART2 DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. UART module uses the synchronized DMA\_CLR signal by default.

- 0: use the DMA\_CLR signal after 2 cycles
- 1: directly use the DMA\_CLR signal output by DMAC

**Bit 4 UART3\_DMA\_CLR\_SEL:** UART3 DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. UART module uses the synchronized DMA\_CLR signal by default.

- 0: use the DMA\_CLR signal after 2 cycles
- 1: directly use the DMA\_CLR signal output by DMAC

**Bit 3 SSP0\_DMA\_CLR\_SEL:** SSP0 DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. SSP module uses the synchronized DMA\_CLR signal by default.

- 0: use the DMA\_CLR signal after 2 cycles
- 1: directly use the DMA\_CLR signal output by DMAC

**Bit 2 SSP1\_DMA\_CLR\_SEL:** SSP1 DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. SSP module uses the synchronized DMA\_CLR signal by default.

- 0: use the DMA\_CLR signal after 2 cycles
- 1: directly use the DMA\_CLR signal output by DMAC

**Bit 1 SSP2\_DMA\_CLR\_SEL:** SSP2 DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. SSP module uses the synchronized DMA\_CLR signal by default.

- 0: use the DMA\_CLR signal after 2 cycles
- 1: directly use the DMA\_CLR signal output by DMAC

**Bit 0 SSP\_AFEC\_DMA\_CLR\_SEL:** SSP (for afec) DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. SSP module uses the synchronized DMA\_CLR signal by default.

- 0: use the DMA\_CLR signal after 2 cycles
- 1: directly use the DMA\_CLR signal output by DMAC

#### 7.5.4 SYSCFG-CR3

Offset: 0x00C

Reset Value: 0x00000000

This register is in the AON domain.

31-2	1	0
RESERVED	SYSCFG_DBG_STOP	SYSCFG_DBG_STANDBY
r	r/w	r/w

**Bits 31-2 RESERVED:** Must be kept, and cannot be modified.

**Bit 1 SYSCFG\_DBG\_STOP:** Allow a debug connection in Stop mode. It is only used in debug and it will affect Stop mode implementation.

- 0: not allowed
- 1: allowed

**Bit 0 SYSCFG\_DBG\_STANDBY:** Allow a debug connection in Standby mode. It is only used in debug and it will affect Standby mode implementation.

- 0: not allowed
- 1: allowed

#### 7.5.5 SYSCFG\_CR4

Offset: 0x010

Reset Value: 0x00000000

This register is in the AON domain.

31	30-0
SYSCFG_CR4_REG	USER-DEFINED
r/w	r/w

**Bit 31 SYSCFG\_CR4\_REG:** LPTIM1\_IN2 remapping enable

- 0: disabled, LPTIM1\_IN2 is determined by GPIO AFR
- 1: enabled, LPTIM1\_IN2 is derived from LPTIM0\_IN1

**Bits 30-0 USER-DEFINED:** These bits are user-defined and can be used to store a small amount of data by software.

## 7.5.6 SYSCFG\_CR5

Offset: 0x014

Reset Value: 0x00000000

This register is in the AON domain.

31-0
SYSCFG_CR5_REG
r/w

**Bits 31-0 SYSCFG\_CR5\_REG:** These bits are user-defined and can be used to store a small amount of data by software.

## 7.5.7 SYSCFG\_CR6

Offset: 0x018

Reset Value: 0x00000000

31-16	15	14-5	4
RESERVED	RNGC_SECURE_LOCK	ANALOG_MAIN_SECU RE_LOCK	RESERVED
r	r/w	r/w	r
3	2	1	0
SEC_SECURE_LOCK	SAC_SECURE_LOCK	DMAC0_SLAVE_SECU RE_LOCK	DMAC0_MASTER_SEC URE_LOCK
r/w	r/w	r/w	r/w

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bit 15 RNGC\_SECURE\_LOCK:** RNGC security lock

- 0: security lock disabled
- 1: security lock enabled

**Bits 14-5 ANALOG\_MAIN\_SECURE\_LOCK:** Security lock for main domain configuration of AFEC

[5] Correspond to VD

- 0: security lock disabled
- 1: security lock enabled

[6] Correspond to TD

- 0: security lock disabled
- 1: security lock enabled

[7] Correspond to LD

- 0: security lock disabled
- 1: security lock enabled

[8] Correspond to FD24M

- 0: security lock disabled
- 1: security lock enabled

[9] Correspond to FD32M

- 0: security lock disabled
- 1: security lock enabled

[10] Correspond to RNG

- 0: security lock disabled
- 1: security lock enabled

[11] Correspond to TEST

- 0: security lock disabled
- 1: security lock enabled

[14:12]: Unused

- 0: security lock disabled
- 1: security lock enabled

**Bit 4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 SEC\_SECURE\_LOCK:** SEC security lock

- 0: security lock disabled
- 1: security lock enabled

**Bit 3 SAC\_SECURE\_LOCK:** SAC security lock

- 0: security lock disabled
- 1: security lock enabled

**Bit 1 DMAC0\_SLAVE\_SECURE\_LOCK:** DMAC0 slave interface security lock

- 0: security lock disabled
- 1: security lock enabled

**Bit 0 DMAC0\_MASTER\_SECURE\_LOCK:** DMAC0 master interface security lock

- 0: security lock disabled
- 1: security lock enabled

## 7.5.8 SYSCFG\_CR7

Offset: 0x01C

Reset Value: 0x00000000

This register is in the AON domain.

31-15	14-5		4
RESERVED	ANALOG_AON_SECURE_LOCK		RTC_CALENDAR_SECURE_LOCK
r	r/w		r/w
3	2	1	0
RTC_WAKEUP2_SECURE_LOCK	RTC_WAKEUP1_SECU	RTC_WAKEUP0_SECU	RTC_TAMPER_SECURE_LOCK
r/w	r/w	r/w	r/w

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 14-5 ANALOG\_AON\_SECURE\_LOCK:** Security lock for AON domain configuration of AFEC

[5] Correspond to LPLDO

- 0: security lock disabled
- 1: security lock enabled

[6] Correspond to RCO3.6M

- 0: security lock disabled
- 1: security lock enabled

[7] Correspond to PWRSW

- 0: security lock disabled
- 1: security lock enabled

[8] Correspond to RCO32K

- 0: security lock disabled
- 1: security lock enabled

[9] Correspond to XO32K

- 0: security lock disabled
- 1: security lock enabled

[10] Correspond to LDO12

- 0: security lock disabled
- 1: security lock enabled

[11] Correspond to FD32K

- 0: security lock disabled
- 1: security lock enabled

[14:12] Unused

- 0: security lock disabled
- 1: security lock enabled

**Bit 4 RTC\_CALENDAR\_SECURE\_LOCK:** Calendar configuration security lock in RTC

- 0: security lock disabled
- 1: security lock enabled

**Bit 3 RTC\_WAKEUP2\_SECURE\_LOCK:** Wakeup2 configuration security lock in RTC

- 0: security lock disabled
- 1: security lock enabled

**Bit 2 RTC\_WAKEUP1\_SECURE\_LOCK:** Wakeup1 configuration security lock in RTC

- 0: security lock disabled
- 1: security lock enabled

**Bit 1 RTC\_WAKEUP0\_SECURE\_LOCK:** Wakeup0 configuration security lock in RTC

- 0: security lock disabled
- 1: security lock enabled

**Bit 0 RTC\_TAMPER\_SECURE\_LOCK:** Tamper configuration security lock in RTC

- 0: security lock disabled
- 1: security lock enabled

### 7.5.9 SYSCFG\_CR8

Offset: 0x020

Reset Value: 0x00000000

31-0
QSPI_MEM_ENCRYPT_KEY
r/w

**Bits 31-0 QSPI\_MEM\_ENCRYPT\_KEY:** Encryption key for QSPI memory

### 7.5.10 SYSCFG\_CR9

Offset: 0x024

Reset Value: 0x00000000

31-28	27-14	13-0
RESERVED	QSPI_REMAP_SRC_ADDR	QSPI_REMAP_DST_ADDR
r	r/w	r/w

**Bits 31-28 RESERVED:** Must be kept, and cannot be modified.

**Bits 27-14 QSPI\_REMAP\_SRC\_ADDR:** QSPI remap source address, aligned in 1KB

**Bits 13-0 QSPI\_REMAP\_DST\_ADDR:** QSPI remap destination address, aligned in 1KB

### 7.5.11 SYSCFG\_CR10

Offset: 0x028

Reset Value: 0x00000000

31-24	23	22	21-15	14	13-0
RESERVED	I2S_WS_SEL	I2S_WS_EN	I2S_WS_LEN	I2S_MODE_SEL	QSPI_REMAP_SIZE
r	r/w	r/w	r/w	r/w	r/w

**Bits 31-24 RESERVED:** Must be kept, and cannot be modified.

**Bit 23 I2S\_WS\_SEL:** I2S WS output delay enable

- 0: output delay disabled
- 1: output delay enabled

**Note:** This bit can only be configured when the I2S acts as master interface. When enabled, the WS signal is output one cycle later than the data transmission.

**Bit 22 I2S\_WS\_EN:** I2S WS enable

- 0: disabled
- 1: enabled

**Note:** This bit can only be configured when the I2S acts as master interface. When enabled, the WS signal is generated based on the I2S\_WS\_LEN configuration.

**Bits 21-15 I2S\_WS\_LEN:** I2S main interface resolution configuration

N: WS frequency=I2S interface clock frequency/[(N+1)\*2]

The I2S interface clock frequency is jointly determined by the I2S\_CLK\_DIV and I2S\_CLK\_SEL bits in the [RCC\\_CR3](#) and [RCC\\_CR2](#) registers.

**Bit 14 I2S\_MODE\_SEL:** I2S works in master or slave mode

- 0: slave mode
- 1: master mode

**Note:** In addition to this register, it is also necessary to configure the I2S\_CLK\_DIV and I2S\_CLK\_SEL bits in the [RCC\\_CR3](#) and [RCC\\_CR2](#) registers, as well as the alternate functions of GPIOs.

**Bits 13-0 QSPI\_REMAP\_SIZE:** Address space for QSPI remapping, aligned in 1KB

## 7.6 DMA Request MUX

Table 7-9 DMA Request MUX

No.	Source
63	
62	
61	
60	
59	
58	
57	
56	
55	
54	
53	basictim0_up
52	basictim1_up
51	gptim3_up
50	gptim3_trg
49	gptim3_ch0
48	gptim3_ch1
47	gptim2_up
46	gptim2_trg
45	gptim2_ch0
44	gptim2_ch1
43	gptim1_up
42	gptim1_trg
41	gptim1_ch0
40	gptim1_ch1
39	gptim1_ch2
38	gptim1_ch3
37	gptim0_up
36	gptim0_trg
35	gptim0_ch0
34	gptim0_ch1
33	gptim0_ch2
32	gptim0_ch3
31	uart0_rx
30	uart0_tx
29	uart1_rx
28	uart1_tx
27	uart2_rx

No.	Source
26	uart2_tx
25	uart3_rx
24	uart3_tx
23	lpuart_rx
22	lpuart_tx
21	ssp0_rx
20	ssp0_tx
19	ssp1_rx
18	ssp1_tx
17	ssp2_rx
16	ssp2_tx
15	i2c0_rx
14	i2c0_tx
13	i2c1_rx
12	i2c1_tx
11	i2c2_rx
10	i2c2_tx
9	
8	
7	adcctrl
6	dacctrl
5	lorac_rx
4	lorac_tx
3	
2	
1	
0	

# 8.

# Reset and Clock Control (RCC)

## 8.1 Reset

There are four types of reset: external reset, power reset, system reset and low-power reset.

### 8.1.1 External Reset

The external reset is triggered by RSTN IO input (active at low level).

The external reset is used to reset all digital logic.

### 8.1.2 Power-on Reset

The power-on reset is generated by the BOR (Brownout reset) circuitry. The BOR circuitry monitors VBAT to ensure that the internal reset is released when the voltage is greater than 1.8V.

Power-on reset is used to reset all digital logic.

### 8.1.3 System Reset

System reset sources include IWDG Reset, WWDG Reset, Option Byte Load Reset, Software Reset, SEC Reset, Power-on Reset, and External Reset.

- IWDG Reset: generated by the IWDG module for exception recovery.
- WWDG Reset: generated by the WWDG module for exception recovery.
- Option Byte Load Reset: generated by the EFC module and used to start option byte reloading.
- Software Reset: generated by the CPU.
- SEC Reset: generated by the SEC module and used for system reset after security alarm.

System reset is used to reset most of the data logic in the Main domain, but does not affect the reset source status register, which is used to determine which system reset source generates this reset.

### 8.1.4 Low-power Reset

The low-power reset is generated by the low-power state machine and is used to reset the logic of the main domain when the CPU exits Standby or Stop3 mode.

## 8.2 Clock

System clock structure:

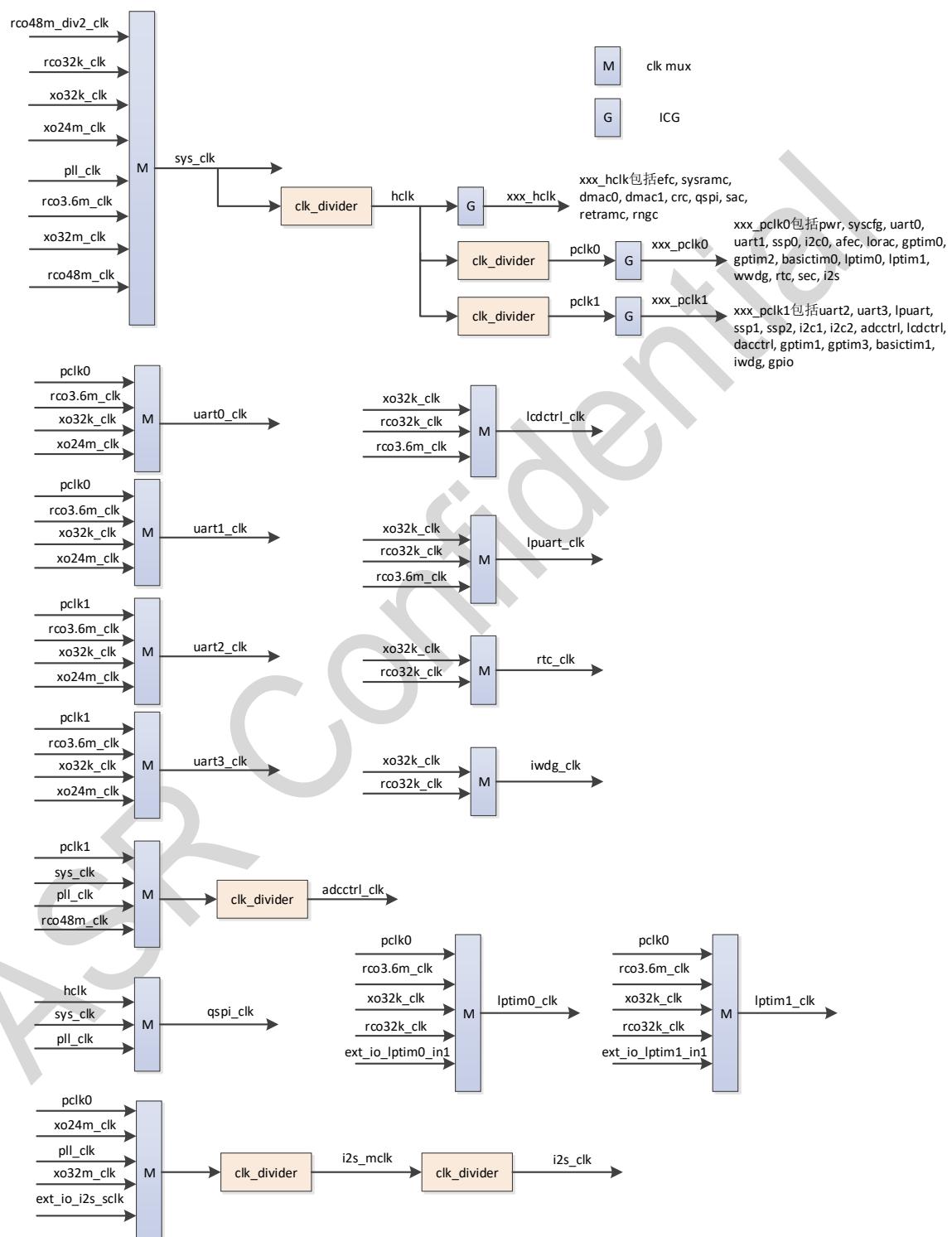


Figure 8-1 Clock Tree

## 8.2.1 System clock SYS\_CLK

The sources of system clock SYS\_CLK include RCO48M divided by 2, RCO32K, XO32K, PLL, XO24M, XO32M, RCO3.6M, RCO48M. The default is RCO48M divided by 2.

- RCO48M (48MHz) is generated from the internal clock circuit.
- RCO32K (32kHz) is generated from the internal clock circuit.
- RCO3.6M (3.6MHz) is generated from the internal clock circuit.
- XO32K (32.768kHz) is generated from an external crystal oscillator.
- XO32M (32MHz) is generated from an external crystal oscillator.
- XO24M (24MHz) is generated from an external crystal oscillator.
- PLL is an internal clock circuit, RCO48M, XO32M, XO24M or RCO3.6M can be selected as PLL clock source, and the PLL clock output supports up to 48MHz.

AHB bus clock HCLK is generated from SYS\_CLK divided by  $2^N$  (N ranges from 0 to 9).

The system includes two APB buses, the APB bus clock PCLK1 and PCLK2 are generated from HCLK divided by  $2^M$  (M in range from 0 to 4). The clock division factor for the two APB buses can be configured independently.

## 8.2.2 Clocks for the Modules

The clocks for the modules consist of bus clocks and interface clocks.

The bus clock is generated by HCLK or PCLK gating and is used for modules to access bus.

In addition to a bus clock, some modules also have an independent interface clock, which is different from the bus clock, and is used to realize the function of the module.

The interface clock source for each module is selectable by software:

- LPTIM: PCLK0, RCO3.6M, XO32K, RCO32K, IO input clock;
- LCDCTRL: XO32K, RCO32K, RCO3.6M;
- LPUART: XO32K, RCO32K, RCO3.6M;
- RTC: XO32K, RCO32K;
- IWDG: XO32K, RCO32K;
- UART: PCLK0/PCLK1, RCO3.6M, XO32K, XO24M;
- ADCCTRL: PCLK1, SYS\_CLK, PLL, RCO48M;
- I2S: PCLK0, XO24M, PLL, XO32M, input clock IO;
- QSPI: HCLK, SYS\_CLK, PLL;

ADCCTRL and I2S also support interface clock division, which is used to generate low frequency interface clocks.

LPTIM, LCDCTRL, LPUART, RTC and IWDG in AON domain and those in Main domain can be enabled or disabled independently.

### 8.2.3 MCO Clock output

The microcontroller clock output (MCO) capability allows the internal clock to be output by IO.

MCO clock source can be RCO32K, XO32K, RCO3.6M, XO24M, XO32M, RCO48M, PLL or SYS\_CLK.

The clock can be output with a frequency divided by software configuration.

## 8.3 RCC Registers

Base Address: 0x40000000

**Table 8-1 RCC Registers Summary**

Register	Offset	Description
RCC_CR0	0x000	Control register 0
RCC_CR1	0x004	Control register 1, interface clock source selection
RCC_CR2	0x008	Control register 2, interface clock source selection
RCC_CGR0	0x00C	Module clock configure register 0
RCC_CGR1	0x010	Module clock configure register 1
RCC_CGR2	0x014	Module clock configure register 2
RCC_RST0	0x018	Module reset control register 0
RCC_RST1	0x01C	Module reset control register 1
RCC_RST_SR	0x020	System reset source Status register
RCC_RST_CR	0x024	System reset source control register
RCC_SR	0x028	Status register, RCC_CGR2 configuration status
RCC_SR1	0x02C	Status register 1, module clock configuration status
RCC_CR3	0x030	Control register 3, interface clock division

### 8.3.1 RCC\_CR0

Offset: 0x000

Reset Value: 0x00000000

31-26	25	24-22	21-19	18
RESERVED	STCLKEN_SEL	MCO_CLK_DIV_NUM	MCO_CLK_SEL	MCO_CLK_OUT_EN
r	r/w	r/w	r/w	r/w
17-15	14-12	11-8	7-5	4-0
PCLK1_DIV	SYS_CLK_SEL	HCLK_DIV	PCLK0_DIV	RESERVED
r/w	r/w	r/w	r/w	r

**Bits 31-26 RESERVED:** Must be kept, and cannot be modified.

**Bit 25 STCLKEN\_SEL:** CPU SysTick clock source selection.

- 0: XO32K
- 1: RCO32K

**Bits 24-22 MCO\_CLK\_DIV\_NUM:** MCO division factor.

- <4: division factor 1
- 4: division factor 2
- 5: division factor 4
- 6: division factor 8
- 7: division factor 16

**Note:** Make sure to configure this bit when MCO\_CLK\_OUT\_EN=0. If the MCO\_CLK\_OUT\_EN bit is enabled, users must disable it by software first, wait for at least 2 current clock cycles or query the [RCC\\_SR1](#) register, and then configure the MCO division factor.

**Bits 21-19 MCO\_CLK\_SEL:** MCO clock source selection.

- 0: RCO32K
- 1: XO32K
- 2: RCO3.6M
- 3: XO24M
- 4: XO32M
- 5: RCO48M
- 6: PLL
- 7: SYS\_CLK

**Note:** Make sure to configure this bit when MCO\_CLK\_OUT\_EN=0. If the MCO\_CLK\_OUT\_EN bit is enabled, users must disable it by software first, wait for at least 2 current clock cycles or query the [RCC\\_SR1](#) register, and then configure the MCO clock source.

**Bit 18 MCO\_CLK\_OUT\_EN:** MCO output enable

- 0: disabled
- 1: enabled

**Bits 17-15 PCLK1\_DIV:** PCLK1 division factor.

- 0: PCLK1 clock frequency = HCLK clock frequency
- 1: PCLK1 clock frequency = 1/2 HCLK clock frequency
- 2: PCLK1 clock frequency = 1/4 HCLK clock frequency
- 3: PCLK1 clock frequency = 1/8 HCLK clock frequency
- >3: PCLK1 clock frequency = 1/16 HCLK clock frequency

**Bits 14-12 SYS\_CLK\_SEL:** SYS\_CLK clock source selection=

- 0: RCO48M divided by 2
- 1: RCO32K
- 2: XO32K
- 3: PLL
- 4: XO24M
- 5: XO32M
- 6: RCO3.6M
- 7: RCO48M

**Bits 11-8 HCLK\_DIV:** HCLK division factor

- 0: HCLK clock frequency = SYS\_CLK clock frequency
- 1: HCLK clock frequency = 1/2 SYS\_CLK clock frequency
- 2: HCLK clock frequency = 1/4 SYS\_CLK clock frequency
- 3: HCLK clock frequency = 1/8 SYS\_CLK clock frequency
- 4: HCLK clock frequency = 1/16 SYS\_CLK clock frequency
- 5: HCLK clock frequency = 1/32 SYS\_CLK clock frequency
- 6: HCLK clock frequency = 1/64 SYS\_CLK clock frequency
- 7: HCLK clock frequency = 1/128 SYS\_CLK clock frequency
- 8: HCLK clock frequency = 1/256 SYS\_CLK clock frequency
- >8: HCLK clock frequency = 1/512 SYS\_CLK clock frequency

**Bits 7-5 PCLK0\_DIV:** PCLK0 division factor

- 0: PCLK0 clock frequency = HCLK clock frequency
- 1: PCLK0 clock frequency = 1/2 HCLK clock frequency
- 2: PCLK0 clock frequency = 1/4 HCLK clock frequency
- 3: PCLK0 clock frequency = 1/8 HCLK clock frequency
- >3: PCLK0 clock frequency = 1/16 HCLK clock frequency

**Bits 4-0 RESERVED:** Must be kept, and cannot be modified.

### 8.3.2 RCC\_CR1

Offset: 0x004

Reset Value: 0x00000000

This register is in the AON domain.

31-12	11	10	9-8	
RESERVED	LPTIM1_EXT_CLK_SEL	LPTIM0_EXT_CLK_SEL	LPTIM1_CLK_SEL	
r	r/w	r/w	r/w	
7-6	5-4	3-2	1	0
LPTIM0_CLK_SEL	LCDCTRL_CLK_SEL	LPUART_CLK_SEL	RTC_CLK_SEL	IWDG_CLK_SEL
r/w	r/w	r/w	r/w	r/w

**Bits 31-12 RESERVED:** Must be kept, and cannot be modified.

**Bit 11 LPTIM1\_EXT\_CLK\_SEL:** LPTIM1 interface clock source selection.

- 0: decided by the LPTIM1\_CLK\_SEL bit
- 1: use external clock from IN1

**Notes:**

1. Make sure to configure this bit when LPTIM1\_CLK\_EN=0. If the LPTIM1\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the [RCC\\_SR1](#) register, and then configure the LPTIM1 interface clock source.
2. This bit and the LPTIM1\_CLK\_SEL bit jointly determine the LPTIM1 interface clock source.

**Bit 10 LPTIM0\_EXT\_CLK\_SEL:** LPTIM0 interface clock source selection.

- 0: decided by the LPTIM0\_CLK\_SEL bit
- 1: use external clock from IN1

**Notes:**

1. Make sure to configure this bit when LPTIM0\_CLK\_EN=0. If the LPTIM0\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the [RCC\\_SR1](#) register, and then configure the LPTIM0 interface clock source.
2. This bit and the LPTIM0\_CLK\_SEL bit jointly determine the LPTIM0 interface clock source.

**Bits 9-8 LPTIM1\_CLK\_SEL:** LPTIM1 interface clock source selection.

- 0: PCLK0
- 1: RCO3.6M
- 2: XO32K
- 3: RCO32K

**Notes:**

1. Make sure to configure this bit when LPTIM1\_CLK\_EN=0. If the LPTIM1\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the [RCC\\_SR1](#) register, and then configure the LPTIM1 interface clock source.
2. This bit and the LPTIM1\_EXT\_CLK\_SEL bit jointly determine the LPTIM1 interface clock source.
3. To select PCLK0 as clock source, the LPTIM1\_INF\_CLK\_EN bit in the [RCC\\_CGR1](#) register must be enabled.

**Bits 7-6 LPTIM0\_CLK\_SEL:** LPTIM0 interface clock source selection.

- 0: PCLK0
- 1: RCO3.6M
- 2: XO32K
- 3: RCO32K

**Notes:**

1. Make sure to configure this bit when *LPTIM0\_CLK\_EN*=0. If the *LPTIM0\_CLK\_EN* bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the *RCC\_SR1* register, and then configure the LPTIM0 interface clock source.
2. This bit and the *LPTIM0\_EXT\_CLK\_SEL* bit jointly determine the LPTIM0 interface clock source.
3. To select PCLK0 as clock source, the *LPTIM0\_INF\_CLK\_EN* bit in the *RCC\_CGR1* register must be enabled.

**Bits 5-4 LCDCTRL\_CLK\_SEL:** LCDCTRL interface clock source selection.

- 0: XO32K
- 1: RCO32K
- >1: RCO3.6M

**Bits 3-2 LPUART\_CLK\_SEL:** LPUART interface clock source selection.

- 0: XO32K
- 1: RCO32K
- >1: RCO3.6M

**Bit 1 RTC\_CLK\_SEL:** RTC interface clock source selection

- 0: XO32K
- 1: RCO32K

**Bit 0 IWDG\_CLK\_SEL:** IWDG interface clock source selection

- 0: XO32K
- 1: RCO32K

### 8.3.3 RCC\_CR2

Offset: 0x008

Reset Value: 0x00000000

31-17	16-15	14-13	12-11	
RESERVED	UART0_CLK_SEL	UART1_CLK_SEL	UART2_CLK_SEL	
r	r/w	r/w	r/w	
10-9	8-7	6-5	4-2	1-0
UART3_CLK_SEL	RESERVED	ADCCTRL_CLK_SEL	I2S_CLK_SEL	QSPI_CLK_SEL
r/w	r	r/w	r/w	r/w

**Bits 31-17 RESERVED:** Must be kept, and cannot be modified.

**Bits 16-15 UART0\_CLK\_SEL:** UART0 interface clock source selection.

- 0: PCLK0
- 1: RCO3.6M
- 2: XO32K
- 3: XO24M

**Note:** Make sure to configure this bit when *UART0\_CLK\_EN=0*. If the *UART0\_CLK\_EN* bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the [RCC\\_SR1](#) register, and then configure the *UART0* interface clock source.

**Bits 14-13 UART1\_CLK\_SEL:** UART1 interface clock source selection.

- 0: PCLK0
- 1: RCO3.6M
- 2: XO32K
- 3: XO24M

**Note:** Make sure to configure this bit when *UART1\_CLK\_EN=0*. If the *UART1\_CLK\_EN* bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the [RCC\\_SR1](#) register, and then configure the *UART1* interface clock source.

**Bits 12-11 UART2\_CLK\_SEL:** UART2 interface clock source selection.

- 0: PCLK1
- 1: RCO3.6M
- 2: XO32K
- 3: XO24M

**Note:** Make sure to configure this bit when *UART2\_CLK\_EN=0*. If the *UART2\_CLK\_EN* bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the [RCC\\_SR1](#) register, and then configure the *UART2* interface clock source.

**Bits 10-9 UART3\_CLK\_SEL:** UART3 interface clock source selection.

- 0: PCLK1
- 1: RCO3.6M
- 2: XO32K
- 3: XO24M

**Note:** Make sure to configure this bit when `UART3_CLK_EN`=0. If the `UART3_CLK_EN` bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the `RCC_SR1` register, and then configure the `UART3` interface clock source.

**Bits 8-7 RESERVED:** Must be kept, and cannot be modified.

**Bits 6-5 ADCCTRL\_CLK\_SEL:** ADCCTRL interface clock source selection.

- 0: PCLK1
- 1: SYS\_CLK
- 2: PLL
- 3: RCO48M

**Note:** Make sure to configure this bit when `ADCCTRL_CLK_EN`=0. If the `ADCCTRL_CLK_EN` bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the `RCC_SR1` register, and then configure the `ADCCTRL` interface clock source.

**Bits 4-2 I2S\_CLK\_SEL:** I2S interface clock source selection

- 0: PCLK0
- 1: XO24M
- 2: PLL
- 3: XO32M
- 3: XO32M

**Notes:**

1. Make sure to configure this bit when `I2S_CLK_EN`=0. If the `I2S_CLK_EN` bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the `RCC_SR1` register, and then configure the `I2S` interface clock source.
2. When `I2S` acts as a slave, the clock source must be configured to external `IOM_I2S_CLK`; when `I2S` acts as a master, the clock source is selected according to functional requirements.

**Bits 1-0 QSPI\_CLK\_SEL:** QSPI interface clock source selection

- 0: HCLK
- 1: SYS\_CLK
- >1: PLL

**Note:** Make sure to configure this bit when `QSPI_CLK_EN`=0. If the `QSPI_CLK_EN` bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the `RCC_SR1` register, and then configure the `QSPI` interface clock source.

## 8.3.4 RCC\_CGR0

Offset: 0x00C

Reset Value: 0x00000000

<b>31</b>	<b>30</b>	<b>29</b>	<b>28</b>	<b>27</b>	<b>26</b>	<b>25</b>	<b>24</b>
PWR_CLK_EN	DMAC0_C_LK_EN	DMAC1_C_LK_EN	CRC_CLK_EN	BASICTIM0_CLK_EN	BASICTIM1_CLK_EN	IOM0_CLK_K_EN	IOM1_CLK_K_EN
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
<b>23</b>	<b>22</b>	<b>21</b>	<b>20</b>	<b>19</b>	<b>18</b>	<b>17</b>	<b>16</b>
IOM2_CLK_EN	IOM3_CL_K_EN	SYSCFG_CLK_EN	UART0_C_LK_EN	UART1_CL_K_EN	UART2_CL_K_EN	UART3_C_LK_EN	LPUART_CLK_EN
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>
SSP0_CLK_EN	SSP1_CL_K_EN	SSP2_CL_K_EN	I2C0_CLK_EN	I2C1_CLK_EN	I2C2_CLK_EN	RESERVE_D	ADCCTRL_CLK_EN
r/w	r/w	r/w	r/w	r/w	r/w	r	r/w
<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
AFEC_CL_K_EN	LCDCTRL_CLK_EN	DACCTRL_CLK_EN	LORAC_C_LK_EN	GPTIM0_C_LK_EN	GPTIM1_C_LK_EN	GPTIM2_CLK_EN	GPTIM3_CLK_EN
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

**Bit 31 PWR\_CLK\_EN:** PWR clock enable.

- 0: disabled
- 1: enabled

**Bit 30 DMAC0\_CLK\_EN:** DMAC0 clock enable.

- 0: disabled
- 1: enabled

**Bit 29 DMAC1\_CLK\_EN:** DMAC1 clock enable.

- 0: disabled
- 1: enabled

**Bit 28 CRC\_CLK\_EN:** CRC clock enable.

- 0: disabled
- 1: enabled

**Bit 27 BASICTIM0\_CLK\_EN:** BASICTIM0 clock enable.

- 0: disabled
- 1: enabled

**Bit 26 BASICTIM1\_CLK\_EN:** BASICTIM1 clock enable

- 0: disabled
- 1: enabled

**Bit 25 IOM0\_CLK\_EN:** IOM0 clock enable.

- 0: disabled
- 1: enabled

**Bit 24 IOM1\_CLK\_EN:** IOM1 clock enable.

- 0: disabled
- 1: enabled

**Bit 23 IOM2\_CLK\_EN:** IOM2 clock enable.

- 0: disabled
- 1: enabled

**Bit 22 IOM3\_CLK\_EN:** IOM3 clock enable.

- 0: disabled
- 1: enabled

**Bit 21 SYSCFG\_CLK\_EN:** SYSCFG clock enable.

- 0: disabled
- 1: enabled

**Bit 20 UART0\_CLK\_EN:** UART0 clock enable.

- 0: disabled
- 1: enabled

**Bit 19 UART1\_CLK\_EN:** UART1 clock enable.

- 0: disabled
- 1: enabled

**Bit 18 UART2\_CLK\_EN:** UART2 clock enable.

- 0: disabled
- 1: enabled

**Bit 17 UART3\_CLK\_EN:** UART3 clock enable.

- 0: disabled
- 1: enabled

**Bit 16 LPUART\_CLK\_EN:** LPUART clock enable.

- 0: disabled
- 1: enabled

**Bit 15 SSP0\_CLK\_EN:** SSP0 clock enable.

- 0: disabled
- 1: enabled

**Bit 14 SSP1\_CLK\_EN:** SSP1 clock enable.

- 0: disabled
- 1: enabled

**Bit 13 SSP2\_CLK\_EN:** SSP2 clock enable.

- 0: disabled
- 1: enabled

**Bit 12 I2C0\_CLK\_EN:** I2C0 clock enable.

- 0: disabled
- 1: enabled

**Bit 11 I2C1\_CLK\_EN:** I2C1 clock enable.

- 0: disabled
- 1: enabled

**Bit 10 I2C2\_CLK\_EN:** I2C2 clock enable.

- 0: disabled
- 1: enabled

**Bit 9 RESERVED:** Must be kept, and cannot be modified.

**Bit 8 ADCCTRL\_CLK\_EN:** ADCCTRL clock enable.

- 0: disabled
- 1: enabled

**Bit 7 AFEC\_CLK\_EN:** AFEC clock enable.

- 0: disabled
- 1: enabled

**Bit 6 LCDCTRL\_CLK\_EN:** LCDCTRL clock enable.

- 0: disabled
- 1: enabled

**Bit 5 DACCTRL\_CLK\_EN:** DACCTRL clock enable.

- 0: disabled
- 1: enabled

**Bit 4 LORAC\_CLK\_EN:** LORAC clock enable.

- 0: disabled
- 1: enabled

**Bit 3 GPTIM0\_CLK\_EN:** GPTIM0 clock enable.

- 0: disabled
- 1: enabled

**Bit 2 GPTIM1\_CLK\_EN:** GPTIM1 clock enable.

- 0: disabled
- 1: enabled

**Bit 1 GPTIM2\_CLK\_EN:** GPTIM2 clock enable.

- 0: disabled
- 1: enabled

**Bit 0 GPTIM3\_CLK\_EN:** GPTIM3 clock enable.

- 0: disabled
- 1: enabled

## 8.3.5

**RCC\_CGR1**

Offset: 0x010

Reset Value: 0x00000000

31-13	12	11	10	9	8	7
RESERVED	LPTIM1_INF_CLK_EN	LPTIM1_CLK_EN	RNGC_CLK_EN	LPTIM0_INF_CLK_EN	I2S_CLK_EN	SAC_CLK_EN
r	r/w	r/w	r/w	r/w	r/w	r/w
<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
WWDG_CNT_CLK_EN	QSPI_CLK_EN	LPTIM0_CLK_EN	IWDG_CLK_EN	WWDG_CLK_EN	RTC_CLK_EN	SEC_CLK_EN
r/w	r/w	r/w	r/w	r/w	r/w	r/w

**Bits 31-13 RESERVED:** Must be kept, and cannot be modified.**Bit 12 LPTIM1\_INF\_CLK\_EN:** LPTIM1 interface PCLK0 clock enable.

- 0: disabled
- 1: enabled

**Bit 11 LPTIM1\_CLK\_EN:** LPTIM1 clock enable.

- 0: disabled
- 1: enabled

**Note:** If PCLK0 is selected as the clock source, the LPTIM1\_INF\_CLK\_EN bit must be enabled before enabling the LPTIM1 clock, while it must be disabled after the LPTIM1 clock is disabled.

**Bit 10 RNGC\_CLK\_EN:** RNGC clock enable.

- 0: disabled
- 1: enabled

**Bit 9 LPTIM0\_INF\_CLK\_EN:** LPTIM0 interface PCLK0 clock enable.

- 0: disabled
- 1: enabled

**Bit 8 I2S\_CLK\_EN:** I2S clock enable.

- 0: disabled
- 1: enabled

**Bit 7 SAC\_CLK\_EN:** SAC clock enable.

- 0: disabled
- 1: enabled

**Bit 6 WWDG\_CNT\_CLK\_EN:** WWDG counter clock enable.

- 0: disabled
- 1: enabled

**Bit 5 QSPI\_CLK\_EN:** QSPI clock enable.

- 0: disabled
- 1: enabled

**Bit 4 LPTIM0\_CLK\_EN:** LPTIM0 clock enable.

- 0: disabled
- 1: enabled

**Note:** If PCLK0 is selected as the clock source, the LPTIM0\_INF\_CLK\_EN bit must be enabled before enabling the LPTIM0 clock, while it must be disabled after the LPTIM0 clock is disabled.

**Bit 3 IWDG\_CLK\_EN:** IWDG clock enable.

- 0: disabled
- 1: enabled

**Bit 2 WWDG\_CLK\_EN:** WWDG clock enable.

- 0: disabled
- 1: enabled

**Bit 1 RTC\_CLK\_EN:** RTC clock enable.

- 0: disabled
- 1: enabled

**Bit 0 SEC\_CLK\_EN:** SEC clock enable.

- 0: disabled
- 1: enabled

### 8.3.6 RCC\_CGR2

Offset:0x014

Reset Value: 0x00000000

This register is in the AON power domain. Read the [RCC\\_SR](#) register before configuring this register.

When the corresponding bit is set in the [RCC\\_SR](#) register, this register can be read; when all the bits are set in the [RCC\\_SR](#) register, this register can be written.

31-6	5	4	3	2	1	0
RESERVED	LPTIM1_AO_N_CLK_EN	LPTIM_AON_CLK_EN	LCDCTRL_AON_CLK_EN	LPUART_AO_N_CLK_EN	RTC_AON_CLK_EN	IWDG_AON_CLK_EN
r	r/w	r/w	r/w	r/w	r/w	r/w

**Bits 31-6 RESERVED:** Must be kept, and cannot be modified.

**Bit 5 LPTIM1\_AON\_CLK\_EN:** Enable the LPTIM1 interface clock in AON domain.

- 0: disabled
- 1: enabled

**Bit 4 LPTIM\_AON\_CLK\_EN:** Enable the LPTIM interface clock in AON domain.

- 0: disabled
- 1: enabled

**Bit 3 LCDCTRL\_AON\_CLK\_EN:** Enable the LCDCTRL interface clock in AON domain.

- 0: disabled
- 1: enabled

**Bit 2 LPUART\_AON\_CLK\_EN:** Enable the LPUART interface clock in AON domain.

- 0: disabled
- 1: enabled

**Bit 1 RTC\_AON\_CLK\_EN:** Enable the RTC interface clock in AON domain.

- 0: disabled
- 1: enabled

**Bit 0 IWDG\_AON\_CLK\_EN:** Enable the IWDG interface clock in AON domain.

- 0: disabled
- 1: enabled

### 8.3.7 RCC\_RST0

Offset: 0x018

Reset Value: 0xffffffff

31	30	29	28	27	26	25	24
UART0_R ST_N	UART1_R ST_N	UART2_R ST_N	UART3_R ST_N	LPUART_ RST_N	SSP0_RS T_N	SSP1_RS T_N	SSP2_RS T_N
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
23	22	21	20	19	18	17	16
QSPI_RST _N	I2C0_RST _N	I2C1_RST _N	I2C2_RST _N	RESERVE D	ADCCTRL _RST_N	AFEC_RS T_N	LCDCTRL _RST_N
r/w	r/w	r/w	r/w	r	r/w	r/w	r/w
15	14	13	12	11	10	9	8
DACCTRL _RST_N	LORAC_R ST_N	IOM_RST _N	GPTIM0_ RST_N	GPTIM1_ RST_N	GPTIM2_ RST_N	GPTIM3_ RST_N	BASICTIM 0_RST_N
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
7	6	5	4	3	2	1	0
BASICTIM 1_RST_N	LPTIM_R ST_N	IWDG_RS T_N	WWDG_R ST_N	RTC_RST _N	CRC_RST _N	SEC_RST _N	SAC_RST _N
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

**Bit 31 UART0\_RST\_N:** UART0 reset control.

- 0: reset
- 1: no action

**Bit 30 UART1\_RST\_N:** UART1 reset control.

- 0: reset
- 1: no action

**Bit 29 UART2\_RST\_N:** UART2 reset control.

- 0: reset
- 1: no action

**Bit 28 UART3\_RST\_N:** UART3 reset control.

- 0: reset
- 1: no action

**Bit 27 LPUART\_RST\_N:** LPUART reset control.

- 0: reset
- 1: no action

**Bit 26 SSP0\_RST\_N:** SSP0 reset control.

- 0: reset
- 1: no action

**Bit 25 SSP1\_RST\_N:** SSP1 reset control.

- 0: reset
- 1: no action

**Bit 24 SSP2\_RST\_N:** SSP2 reset control.

- 0: reset
- 1: no action

**Bit 23 QSPI\_RST\_N:** QSPI reset control.

- 0: reset
- 1: no action

**Bit 22 I2C0\_RST\_N:** I2C0 reset control.

- 0: reset
- 1: no action

**Bit 21 I2C1\_RST\_N:** I2C1 reset control.

- 0: reset
- 1: no action

**Bit 20 I2C2\_RST\_N:** I2C2 reset control.

- 0: reset
- 1: no action

**Bit 19 RESERVED:** Must be kept, and cannot be modified.

**Bit 18 ADCCTRL\_RST\_N:** ADCCTRL reset control.

- 0: reset
- 1: no action

**Bit 17 AFEC\_RST\_N:** AFEC reset control.

- 0: reset
- 1: no action

**Bit 16 LCDCTRL\_RST\_N:** LCDCTRL reset control.

- 0: reset
- 1: no action

**Bit 15 DACCTRL\_RST\_N:** DACCTRL reset control.

- 0: reset

- 1: no action

**Bit 14 LORAC\_RST\_N:** LORAC reset control.

- 0: reset
- 1: no action

**Bit 13 IOM\_RST\_N:** IOM reset control.

- 0: reset
- 1: no action

**Bit 12 GPTIM0\_RST\_N:** GPTIM0 reset control.

- 0: reset
- 1: no action

**Bit 11 GPTIM1\_RST\_N:** GPTIM1 reset control.

- 0: reset
- 1: no action

**Bit 10 GPTIM2\_RST\_N:** GPTIM2 reset control.

- 0: reset
- 1: no action

**Bit 9 GPTIM3\_RST\_N:** GPTIM3 reset control.

- 0: reset
- 1: no action

**Bit 8 BASICTIM0\_RST\_N:** BASICTIM0 reset control.

- 0: reset
- 1: no action

**Bit 7 BASICTIM1\_RST\_N:** BASICTIM1 reset control.

- 0: reset
- 1: no action

**Bit 6 LPTIM0\_RST\_N:** LPTIM0 reset control.

- 0: reset
- 1: no action

**Bit 5 IWDG\_RST\_N:** IWDG reset control.

- 0: reset
- 1: no action

**Bit 4 WWDG\_RST\_N:** WWDG reset control.

- 0: reset
- 1: no action

**Bit 3 RTC\_RST\_N:** RTC reset control.

- 0: reset
- 1: no action

**Bit 2 CRC\_RST\_N:** CRC reset control.

- 0: reset

- 1: no action

**Bit 1 SEC\_RST\_N:** SEC reset control.

- 0: reset
- 1: no action

**Bit 0 SAC\_RST\_N:** SAC reset control.

- 0: reset
- 1: no action

### 8.3.8 RCC\_RST1

Offset: 0x01C

Reset Value: 0x00000001f

31-5	4	3	2	1	0
RESERVED	LPTIM1_RST_N	RNGC_RST_N	I2S_RST_N	DMAC0_RST_N	DMAC1_RST_N
r	r/w	r/w	r/w	r/w	r/w

**Bits 31-5 RESERVED:** Must be kept, and cannot be modified.

**Bit 4 LPTIM1\_RST\_N:** LPTIM1 reset control.

- 0: reset
- 1: no action

**Bit 3 RNGC\_RST\_N:** RNGC reset control.

- 0: reset
- 1: no action

**Bit 2 I2S\_RST\_N:** I2S reset control.

- 0: reset
- 1: no action

**Bit 1 DMAC0\_RST\_N:** DMAC0 reset control.

- 0: reset
- 1: no action

**Bit 0 DMAC1\_RST\_N:** DMAC1 reset control.

- 0: reset
- 1: no action

### 8.3.9 RCC\_RST\_SR

Offset: 0x020

Reset Value: 0x00000040

**Note:** The BOR\_RESET\_SR and STANDBY\_RESET\_SR are in the AON domain.

31-7	6	5	4	3	2	1	0
RESERVED	BOR_RE SET_SR	IWDG_RE SET_SR	WWDG_RE SET_SR	EFC_RE SET_SR	CPU_RE SET_SR	SEC_RE SET_SR	STANDBY_ RESET_SR
r	r/w	r/w	r/w	r/w	r/w	r/w	r/w

**Bits 31-7 RESERVED:** Must be kept, and cannot be modified.

**Bit 6 BOR\_RESET\_SR:** BOR reset status. Set by hardware and cleared by software by writing 1.

- 0: no BOR reset
- 1: BOR reset occurred

**Bit 5 IWDG\_RESET\_SR:** IWDG reset status. Set by hardware and cleared by software by writing 1.

- 0: no IWDG reset
- 1: IWDG reset occurred

**Bit 4 WWDG\_RESET\_SR:** WWDG reset status. Set by hardware and cleared by software by writing 1.

- 0: no WWDG reset
- 1: WWDG reset occurred

**Bit 3 EFC\_RESET\_SR:** EFC reset status. Set by hardware and cleared by software by writing 1.

- 0: no EFC reset
- 1: EFC reset occurred

**Bit 2 CPU\_RESET\_SR:** CPU reset status. Set by hardware and cleared by software by writing 1.

- 0: no CPU reset
- 1: CPU reset occurred

**Bit 1 SEC\_RESET\_SR:** SEC reset status. Set by hardware and cleared by software by writing 1.

- 0: no SEC reset
- 1: SEC reset occurred

**Bit 0 STANDBY\_RESET\_SR:** Standby reset status. Set by hardware and cleared by software by writing 1.

- 0: no MPU reset
- 1: MPU reset occurred

### 8.3.10 RCC\_RST\_CR

Offset: 0x024

Reset Value: 0x00000004

31-6	5	4	3	2	1	0
RESERVED	IWDG_RESE T_REQ_EN	WWDG_RES ET_REQ_EN	EFC_RESE T_REQ_EN	CPU_RESE T_REQ_EN	SEC_RESE T_REQ_EN	RESERVED
r	r/w	r/w	r/w	r/w	r/w	r

**Bits 31-6 RESERVED:** Must be kept, and cannot be modified.

**Bit 5 IWDG\_RESET\_REQ\_EN:** IWDG reset enable.

- 0: disabled
- 1: enabled

**Bit 4 WWDG\_RESET\_REQ\_EN:** WWDG reset enable.

- 0: disabled
- 1: enabled

**Bit 3 EFC\_RESET\_REQ\_EN:** EFC reset enable.

- 0: disabled
- 1: enabled

**Bit 2 CPU\_RESET\_REQ\_EN:** CPU reset enable.

- 0: disabled
- 1: enabled

**Bit 1 SEC\_RESET\_REQ\_EN:** SEC reset enable.

- 0: disabled
- 1: enabled

**Bit 0 RESERVED:** Must be kept, and cannot be modified.

### 8.3.11 RCC\_SR

Offset: 0x028

Reset Value: 0x0000003f

31-6		5	4
RESERVED		SET_LPTIM1_AON_CLK_EN_DONE	SET_LPTIM_AON_CLK_EN_DONE
r		r	r
3	2	1	0
SET_LCDCTRL_AON_CLK_EN_DONE	SET_LPUART_AON_Clk_EN_DONE	SET_RTC_AON_CLK_EN_DONE	SET_IWDG_AON_CLK_EN_DONE
r	r	r	r

**Bits 31-6 RESERVED:** Must be kept, and cannot be modified.

**Bit 5 SET\_LPTIM1\_AON\_CLK\_EN\_DONE:** LPTIM1\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

- 0: configuration in progress
- 1: configuration completed

**Bit 4 SET\_LPTIM0\_AON\_CLK\_EN\_DONE:** LPTIM0\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

- 0: configuration in progress
- 1: configuration completed

**Bit 3 SET\_LCDCTRL\_AON\_CLK\_EN\_DONE:** LCDCTRL\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

- 0: configuration in progress
- 1: configuration completed

**Bit 2 SET\_LPUART\_AON\_CLK\_EN\_DONE:** LPUART\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

- 0: configuration in progress
- 1: configuration completed

**Bit 1 SET\_RTC\_AON\_CLK\_EN\_DONE:** RTC\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

- 0: configuration in progress
- 1: configuration completed

**Bit 0 SET\_IWDG\_AON\_CLK\_EN\_DONE:** IWDG\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

- 0: configuration in progress
- 1: configuration completed

### 8.3.12 RCC\_SR1

Offset: 0x02C

Reset Value: 0x00000000

The clock should be disabled before the clock source is switched or the frequency division changes to avoid glitches. This register is used to determine enable status of the clock.

<b>31-21</b>	<b>20</b>	<b>19</b>	<b>18</b>	<b>17</b>	<b>16</b>
RESERVED	LPTIM1_CLK_EN_SYNC	LPTIM1_AON_C_LK_EN_SYNC	UART0_CLK_EN_SYNC	UART1_CLK_EN_SYNC	UART2_CLK_E_N_SYNC
r	r	r	r	r	r
<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>
UART3_CLK_EN_SYNC	RESERVED	ADCCTRL_CLK_EN_SYNC	LPTIM_CLK_EN_SYNC	QSPI_CLK_E_N_SYNC	LPUART_CLK_EN_SYNC
r	r	r	r	r	r
<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>
LCDCTRL_CL_K_EN_SYNC	IWDG_CLK_EN_SYNC	RTC_CLK_EN_SYNC	MCO_CLK_E_N_SYNC	I2S_CLK_EN_SYNC	LPTIM_AON_C_LK_EN_SYNC
r	r	r	r	r	r
<b>3</b>	<b>2</b>	<b>1</b>			<b>0</b>
LCDCTRL_AON_CLK_EN_SYNC	LPUART_AON_CLK_E_N_SYNC	RTC_AON_CLK_EN_SYNC	IWDG_AON_CLK_EN_SYNC		
r	r	r	r		

**Bits 31-21 RESERVED:** Must be kept, and cannot be modified.

**Bit 20 LPTIM1\_CLK\_EN\_SYNC:** LPTIM1\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 19 LPTIM1\_AON\_CLK\_EN\_SYNC:** LPTIM1\_AON\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 18 UART0\_CLK\_EN\_SYNC:** UART0\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 17 UART1\_CLK\_EN\_SYNC:** UART1\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 16 UART2\_CLK\_EN\_SYNC:** UART2\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 15 UART3\_CLK\_EN\_SYNC:** UART3\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 14 RESERVED:** Must be kept, and cannot be modified.

**Bit 13 ADCCTRL\_CLK\_EN\_SYNC:** ADCCTRL\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 12 LPTIM0\_CLK\_EN\_SYNC:** LPTIM0\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 11 QSPI\_CLK\_EN\_SYNC:** QSPI\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 10 LPUART\_CLK\_EN\_SYNC:** Indicate LPUART\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 9 LCDCTRL\_CLK\_EN\_SYNC:** LCDCTRL\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 8 IWDG\_CLK\_EN\_SYNC:** IWDG\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 7 RTC\_CLK\_EN\_SYNC:** RTC\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 6 MCO\_CLK\_EN\_SYNC:** MCO\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 5 I2S\_CLK\_EN\_SYNC:** I2S\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 4 LPTIM0\_AON\_CLK\_EN\_SYNC:** LPTIM0\_AON\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 3 LCDCTRL\_AON\_CLK\_EN\_SYNC:** LCDCTRL\_AON\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 2 LPUART\_AON\_CLK\_EN\_SYNC:** LPUART\_AON\_CLK\_EN actual status.

- 0: disabled

- 1: enabled

**Bit 1 RTC\_AON\_CLK\_EN\_SYNC:** Indicate RTC\_AON\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

**Bit 0 IWDG\_AON\_CLK\_EN\_SYNC:** IWDG\_AON\_CLK\_EN actual status.

- 0: disabled
- 1: enabled

### 8.3.13 RCC\_CR3

Offset: 0x030

Reset Value: 0x00000000

31-16	15-8	7-0
RESERVED	I2S_MCLK_DIV	I2S_SCLK_DIV
r	r/w	r/w

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-8 I2S\_MCLK\_DIV:** I2S interface clock MCLK frequency division.

- 0: not divided
- 0: not divided
- 2: divided by 2
- 3: divided by 3
- N: divided by N

**Notes:**

1. Make sure to configure I2S\_MCLK\_DIV when I2S\_CLK\_EN=0. If the I2S\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the [RCC\\_SR1](#) register, and then configure I2S\_MCLK\_DIV.
2. When I2S acts as a slave, this bit must be configured to 0 or 1; when I2S acts as a master, this bit is configured according to functional requirements.
3. The duty cycle of the output clock is 50%.

**Bits 7-0 I2S\_SCLK\_DIV:** I2S interface clock SCLK frequency division.

- 0: not divided
- 0: not divided
- 2: divided by 2
- 3: divided by 3
- N: divided by N

**Notes:**

1. Make sure to configure I2S\_SCLK\_DIV when I2S\_CLK\_EN=0. If the I2S\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the [RCC\\_SR1](#) register, and then configure I2S\_SCLK\_DIV.
2. When I2S acts as a slave, this bit must be configured to 0 or 1; when I2S acts as a master, this bit is configured according to functional requirements.
3. The duty cycle of the output clock is 50%.

**ASR**

## 8. Reset and Clock Control (RCC)

ASR6601 Reference Manual



ASR Confidential

# 9.

# Interrupts

## 9.1 Main Features

- Support 37 IRQ interrupts.
- Configurable 0~7 priority levels for each IRQ interrupt.

## 9.2 SysTick

SysTick calibration value is 0x147. Using a 32.768 kHz clock source for SysTick counting gives a reference time base of 10 ms.

## 9.3 Interrupt Vector Table

The interrupt vector Table is as follows:

**Table 9-1 Interrupt Vectors**

Position	Priority	Type of priority	Acronym	Description	Address
-	-	-	-	Reserved	0x0000_0000
-3	fixed		Reset	Reset	0x0000_0004
-2	fixed		NMI_Handler	Secure area check error	0x0000_0008
-1	fixed		HardFault_Handler	fault	0x0000_000C
0	settable		MemManage_Handler	fault	0x0000_0010
1	settable		BusFault_Handler	fault	0x0000_0014
2	settable		UsageFault_Handler	fault	0x0000_0018
-	-	-	-	Reserved	0x0000_001C - 0x0000_002B
	3	settable	SVC_Handler	System service call via SWI instruction	0x0000_002C
	-	-	-	Reserved	0x0000_0030 - 0x0000_0037
	5	settable	PendSV_Handler	Pendable request for system service	0x0000_0038
	6	settable	SysTick_Handler	System tick timer	0x0000_003C
0	7	settable	sec	Include mpu	0x0000_0040
1	8	settable	rtc	Include tamper io, cyc, wakeup io	0x0000_0044



ASR

## 9. Interrupts

ASR6601 Reference Manual



Position	Priority	Type of priority	Acronym	Description	Address
2	9	settable	wwdg		0x0000_0048
3	10	settable	efc		0x0000_004C
4	11	settable	uart3		0x0000_0050
5	12	settable	i2c2		0x0000_0054
6	13	settable	uart0		0x0000_0058
7	14	settable	uart1		0x0000_005C
8	15	settable	uart2		0x0000_0060
9	16	settable	lpuart		0x0000_0064
10	17	settable	ssp0		0x0000_0068
11	18	settable	ssp1		0x0000_006C
12	19	settable	qspi		0x0000_0070
13	20	settable	i2c0		0x0000_0074
14	21	settable	i2c1		0x0000_0078
15	22	settable	-		0x0000_007C
16	23	settable	adcctrl		0x0000_0080
17	24	settable	afec		0x0000_0084
18	25	settable	ssp2		0x0000_0088
19	26	settable	dmac1		0x0000_008C
20	27	settable	dacctrl		0x0000_0090
21	28	settable	lorac		0x0000_0094
22	29	settable	iom		0x0000_0098
23	30	settable	gptim0		0x0000_009C
24	31	settable	gptim1		0x0000_00A0
25	32	settable	gptim2		0x0000_00A4
26	33	settable	gptim3		0x0000_00A8
27	34	settable	basictim0		0x0000_00AC
28	35	settable	basictim1		0x0000_00B0
29	36	settable	lptim0		0x0000_00B4
30	37	settable	sac		0x0000_00B8
31	38	settable	dmac0		0x0000_00BC
32	39	settable	i2s		0x0000_00C0
33	40	settable	lcdctrl		0x0000_00C4
34	41	settable	pwr		0x0000_00C8
35	42	settable	lptim1		0x0000_00CC
36	43	settable	iwdg		0x0000_00D0

# 10.

# Embedded Flash

## 10.1 Introduction

- The whole Flash is divided into Flash info area and Flash main area
- Flash size:
  - ◆ Flash info area: 16 KB
  - ◆ Flash main area: 256 KB for ASR6601SE, 128 KB for ASR6601CB
- • Page erase (4 KB) and Mass erase (all main flash area)

## 10.2 Main Features

- Flash operations include read, program, page erase and mass erase
- Read access latency
- Accessing acceleration
- Instruction prefetch, buffer deep 1
- Flash program operation supports single and continuous modes
- Option bytes in Flash info area
- Interrupt signals generation

## 10.3 Functional Description

### 10.3.1 Flash Info Area Division

The Flash info area is divided into four parts: Option Bytes, Factory Bytes, OTP and BootLoader. See the table below for details.

**Table 10-1 Flash Info Area Division**

Start Address	Description	Size
0x10003000	Option Bytes	4KB
0x10002000	Factory Bytes	4KB
0x10001C00	OTP	1KB
0x10000000	BootLoader	7KB

### 10.3.2 EFC\_CR Protection

By default, the EFC\_CR register cannot be modified, to modify it, the user must configure the protection sequence correctly through the [EFC\\_PROTECT\\_SEQ](#) register in the following order.

If there is an error in the configuration, then the configuration is invalid, and the protection sequence should be reconfigured.

- (1) First write “0x8C9DAEBF” to EFC\_PROTECT\_SEQ register
- (2) Then write “0x13141516” to EFC\_PROTECT\_SEQ register

### 10.3.3 Read Access Latency

In order to improve Flash read performance, the number of wait states (READ\_NUM[19:16]) should be correctly programmed in [EFC\\_TIMING\\_CFG](#) register according to the frequency of SYS\_CLK. The number of wait states (READ\_NUM) equals to (READ\_NUM+1) multiplied by SYS\_CLK clock period. See following details:

- For 48MHz SYS\_CLK frequency READ\_NUM must  $\geq 2$ .
- For 32MHz SYS\_CLK frequency READ\_NUM must  $\geq 1$ .
- For 24MHz SYS\_CLK frequency READ\_NUM must  $\geq 1$ .
- For 3.6MHz SYS\_CLK frequency READ\_NUM must  $\geq 0$ .
- For 32kHz SYS\_CLK frequency READ\_NUM must  $\geq 0$ .

#### Operations to switch to a high-frequency clock source for SYS\_CLK:

- (1) Modify the READ\_NUM value in [EFC\\_TIMING\\_CFG](#) register to match the SYS\_CLK after its clock source is switched.
- (2) Wait for the READ\_NUM\_DONE status bit in [EFC\\_SR](#) register to be set.
- (3) Modify the SYS\_CLK\_SEL field in [RCC\\_CRO](#) register to switch to the target clock source.

#### Operations to switch to a low-frequency clock source for SYS\_CLK:

- (1) Modify the SYS\_CLK\_SEL field in [RCC\\_CRO](#) register to switch to the target clock source.
- (2) Modify the READ\_NUM value in [EFC\\_TIMING\\_CFG](#) register to match the SYS\_CLK after its clock source is switched.
- (3) Wait for the READ\_NUM\_DONE status bit in [EFC\\_SR](#) register to be set.

**Note:** When the user wants to switch to a high-frequency clock source, first increase the READ\_NUM, and then configure the clock source selection bit; otherwise, first configure the clock source selection bit, and then decrease the READ\_NUM.

### 10.3.4 Accessing acceleration

Read acceleration is disabled by default. If  $\text{READ\_NUM} < (2^{\text{HCLK\_DIV}})$ , read acceleration can be enabled to achieve the maximum bus access efficiency. Note that read acceleration must be enabled after READ\_NUM and HCLK\_DIV are configured.

**Note:** Read acceleration and instruction prefetch can't be enabled at the same time.

### 10.3.5 Instruction Prefetch

It is disabled by default. If  $\text{READ\_NUM} \geq (2^{\text{HCLK\_DIV}})$ , read acceleration cannot be enabled. You can choose to enable instruction prefetch to improve access efficiency.

**Note:** Read acceleration and instruction prefetch can't be enabled at the same time.

### 10.3.6 Flash Program

There are two modes for Flash programming:

- **Single Programming Mode**

In single mode, it programs 2 words (8 Bytes) at one time.

- **Continuous Programming Mode**

In continuous mode, it programs a complete word line (512 Bytes) each time.

During continuous programming, Flash cannot be read or executed, so the continuous programming code must be executed in RAM.

#### Steps for single programming:

- (1) Set the PROG\_EN bit in register *EFC\_CR*.
- (2) Write the low 4 Bytes data into register *EFC\_PROG\_DATA0*.
- (3) Write the high 4 Bytes data into register *EFC\_PROG\_DATA1*.
- (4) Write any value to the Flash address to be written into.
- (5) Wait for the OPERATION\_DONE bit in register *EFC\_SR* to be set.
- (6) Write 1 to the OPERATION\_DONE bit in register *EFC\_SR* to clear the flag.

#### Steps for continuous programming:

- (1) Set the PROG\_EN, WRITE\_RELEASE\_EN and PROG\_MODE bits in register *EFC\_CR*.
- (2) Wait for the PROG\_DATA\_WAIT bit in register *EFC\_SR* to be set.
- (3) Write the low 4 Bytes data into register *EFC\_PROG\_DATA0*.
- (4) Write the high 4 Bytes data into register *EFC\_PROG\_DATA1*.
- (5) Write any value to the Flash address to be written into.
- (6) Wait for the PROG\_DATA\_WAIT bit in register *EFC\_SR* to be set.
- (7) Continue to write data to the *EFC\_PROG\_DATA0* and *EFC\_PROG\_DATA1* registers.
- (8) Repeat **Step 6** and **Step 7** until 512 Bytes are written.
- (9) Wait for the OPERATION\_DONE bit in register *EFC\_SR* to be set.
- (10) Write 1 to the OPERATION\_DONE bit in register *EFC\_SR* to clear the flag.

### 10.3.7 Flash Erase

The Flash memory erase operation can be performed at page level (page erase) or on the whole memory (mass erase).

- **Page Erase**

The page erase is measured in 4 Bytes.

- **Mass Erase**

After a mass erase, the entire Flash main area will be 0xFF.

#### Steps for page erase:

- (1) Set the PAGE\_ERASE\_EN bit in register *EFC\_CR*.
- (2) Write any value to the Flash address to be erased.
- (3) Wait for the OPERATION\_DONE bit in register *EFC\_SR* to be set.
- (4) Write 1 to the OPERATION\_DONE bit in register *EFC\_SR* to clear the flag.

#### Steps for mass erase:

- (1) Set the MASS\_ERASE\_EN bit in register *EFC\_CR*.
- (2) Write any value to the Flash address 0x08000000.
- (3) Wait for the OPERATION\_DONE bit in register *EFC\_SR* to be set.
- (4) Write 1 to the OPERATION\_DONE bit in register *EFC\_SR* to clear the flag.

## 10.4 Flash Option Bytes

Flash option bytes is divided into option0 and option1.

### 10.4.1 Flash Option0

Option0 has 64 bits in total, and its format is as follows:

**Table 10-2 Flash Option0**

63-50	49-44	43-38	37-32	31-26	25	24-19
RESERVED	WR_PROT ECT_END	WR_PROTE CT_START	EXE_ONLY2 _END	EXE_ONLY2 _START	EXE_ONLY _KEEP	EXE_ONLY1 _END
<b>18-13</b>	<b>12-5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
EXE_ONLY 1_START	DEBUG_L EVEL	RESERVED	SYS_SRAM _RESET	FLASH_BOO T1	USE_FLAS H_BOOT0	FLASH_BOO T0

**Bits 63-50 RESERVED:** Must be kept, and cannot be modified.

**Bits 49-44 WR\_PROTECT\_END:** Write-protected area end.

When *WR\_PROTECT\_START* > *WR\_PROTECT\_END*, the write-protected area is disabled. It is disabled by default.

**Bits 43-38 WR\_PROTECT\_START:** Write-protected area start.

When *WR\_PROTECT\_START* > *WR\_PROTECT\_END*, the write-protected area is disabled. It is disabled by default.

**Bits 37-32 EXE\_ONLY2\_END:** Exe\_Only2 area end.

When *EXE\_ONLY2\_START* > *EXE\_ONLY2\_END*, the Exe\_Only2 area is disabled. It is disabled by default. Once enabled, this area can only be expanded but can't be disabled or narrowed.

**Bits 31-26 EXE\_ONLY2\_START:** Exe\_Only2 area start.

When *EXE\_ONLY2\_START* > *EXE\_ONLY2\_END*, the Exe\_Only2 area is disabled. It is disabled by default. Once enabled, this area can only be expanded but can't be disabled or narrowed.

**Bit 25 EXE\_ONLY\_KEEP:** Whether Exe\_Only area is kept when the Debug\_Level changes from 1 to 0.

- 0: not keep Exe\_Only area
- 1: keep the Exe\_Only area

This bit can only be set to 0 by software. When Debug\_Level changes from 1 to 0, EXE\_ONLY\_KEEP is set to 1 automatically by hardware.

**Bits 24-19 EXE\_ONLY1\_END:** Exe\_Only1 area end.

When *EXE\_ONLY1\_START* > *EXE\_ONLY1\_END*, the Exe\_Only1 area is disabled. It is disabled by default. Once enabled, this area can only be expanded but can't be disabled or narrowed.

**Bits 18-13 EXE\_ONLY1\_START:** Exe\_Only1 area start.

When *EXE\_ONLY1\_START* > *EXE\_ONLY1\_END*, the Exe\_Only1 area is disabled. It is disabled by default. Once enabled, this area can only be expanded but can't be disabled or narrowed.

**Bits 12-5 DEBUG\_LEVEL:** Debug\_level configuration.

- AA: Level 0
- CC: Level 2
- Others: Level 1

**Bit 4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 SYS\_SRAM\_RESET:** Whether to clear system SRAM during system startup after its reset

- 1: clear system SRAM
- 0: not clear system SRAM

**Bit 2 FLASH\_BOOT1:** This bit can be used to identify the boot mode.

**Bit 1 USE\_FLASH\_BOOT0:** This bit can be used to identify the boot mode.

**Bit 0 FLASH\_BOOT0:** This bit can be used to identify the boot mode.

See table below for the boot mode configuration summary:

**Table 10-3 ASR6601 Boot Mode Configuration**

DEBUG_LEVEL	USE_FLASH_BOOT0	FLASH_BOOT0	BOOT0_PIN	FLASH_BOOT1	MAIN_FLASH_EMPTY	Boot Config
2	X	X	X	X	X	Boot from Flash Main
<2	0	X	0	X	0	Boot from Flash Main
<2	0	X	0	X	1	Boot from Flash Bootloader
<2	0	X	1	1	X	Boot from Flash Bootloader
<2	0	X	1	0	X	Boot from System SRAM
<2	1	1	X	X	0	Boot from Flash Main
<2	1	1	X	X	1	Boot from Flash Bootloader
<2	1	0	X	1	X	Boot from Flash Bootloader
<2	1	0	X	0	X	Boot from System SRAM

## 10.4.2 Flash Option1

Option1 has 64 bits in total, and its format is as follows:

**Table 10-4 Flash Option1**

63-56	55	54-49	48	47-42	41-37
RESERVED	SYSRAM_HID E_EN	SYSRAM_HID E_START	FLASH_HIDE_ EN	FLASH_HIDE_ _START	RETRAM_SEC URE_END
<b>36-32</b>	<b>31-24</b>	<b>23-18</b>	<b>17-12</b>	<b>11-6</b>	<b>5-0</b>
RETRAM_SEC URE_START	RESERVED	SYSRAM_SEC URE_END	SYSRAM_SEC URE_START	FLASH_SEC URE_END	FLASH_SECU RE_START

**Bits 63-56 RESERVED:** Must be kept, and cannot be modified.

**Bit 55 SYSRAM\_HIDE\_EN:** SysRamHide area enable control.

- 0: SysRamHide area enabled
- 1: SysRamHide area disabled

Only valid if FlashSecure area is enabled.

**Bits 54-49 SYSRAM\_HIDE\_START:** SysRamHide area start.

The configuration is only valid when the SysRamHide area is within the SysRamSecure area and the FlashSecure area is enabled by bits[11:0]. The SysRamHide area is from SysRamHideStart to SysRamSecureEnd.

**Bit 48 FLASH\_HIDE\_EN:** FlashHide area enable control.

- 0: FlashHide area enabled
- 1: FlashHide area disabled

Only valid if FlashSecure area is enabled.

**Bits 47-42 FLASH\_HIDE\_START:** FlashHide area start.

The configuration is only valid when the FlashHide area is within the FlashSecure area and the FlashSecure area is enabled by bits[11:0]. The FlashHide area is from FlashHideStart to FlashSecureEnd.

**Bits 41-37 RETRAM\_SECURE\_END:** RetRam Secure area end.

When *RETRAM\_SECURE\_START > RETRAM\_SECURE\_END*, the RetRam Secure area is disabled.  
The configuration is only valid when the FlashSecure area is enabled by bits[11:0].

**Bits 36-32 RETRAM\_SECURE\_START:** RetRam Secure area start.

When *RETRAM\_SECURE\_START > RETRAM\_SECURE\_END*, the RetRam Secure area is disabled.  
The configuration is only valid when the FlashSecure area is enabled by bits[11:0].

**Bits 31-24 RESERVED:** Must be kept, and cannot be modified.

**Bits 23-18 SYSRAM\_SECURE\_END:** SysRam Secure area end.

When *SYSRAM\_SECURE\_START > SYSRAM\_SECURE\_END*, the SysRam Secure area is disabled.  
The configuration is only valid when the FlashSecure area is enabled by bits[11:0].

**Bits 17-12 SYSRAM\_SECURE\_START:** SysRam Secure area start.

When *SYSRAM\_SECURE\_START > SYSRAM\_SECURE\_END*, the SysRam Secure area is disabled.  
The configuration is only valid when the FlashSecure area is enabled by bits[11:0].

**Bits 11-6 FLASH\_SECURE\_END:** Flash Secure area end.

When *FLASH\_SECURE\_START > FLASH\_SECURE\_END*, the Flash Secure area is disabled. The Flash Secure area enable is the master switch for enabling other secure areas. When the Flash Secure area is disabled, the erase operation is triggered.

**Bits 5-0 FLASH\_SECURE\_START:** Flash Secure area start.

When *FLASH\_SECURE\_START > FLASH\_SECURE\_END*, the Flash Secure area is disabled. The Flash Secure area enable is the master switch for enabling other secure areas. When the Flash Secure area is disabled, the erase operation is triggered.

## 10.5 Embedded Flash Registers

Base Address: 0x40020000

**Table 10-5 Embedded Flash Registers Summary**

Register	Offset	Description
EFC_CR	0x000	Control Register
EFC_INT_EN	0x004	Interrupt enable register
EFC_SR	0x008	Status Register
EFC_PROG_DATA0	0x00C	Program Data 0
EFC_PROG_DATA1	0x010	Program Data 1
EFC_TIMING_CFG	0x014	Timing configuration register
EFC_PROTECT_SEQ	0x018	Protection Sequence
RESERVED	0x01C-0x028	Reserved
SERIAL_NUM_LOW	0x02C	Less Significant 32 bits of the Chip Serial Number
SERIAL_NUM_HIGH	0x030	More Significant 32 bits of the Chip Serial Number
RESERVED	0x034-0x038	Reserved
OPTION_CSR_BYTES	0x03C	OPTION control and status data
OPTION_EXE_ONLY_BYTES	0x040	OPTION Execution-only data
OPTION_WR_PROTECT_BYTES	0x044	OPTION Write-protection data
OPTION_SECURE_BYTES0	0x048	OPTION Secure Data 0
OPTION_SECURE_BYTES1	0x04C	OPTION Secure Data 1

### 10.5.1 EFC\_CR

Offset: 0x000

Reset Value: 0x00000000

31	30-10	9	8	7	6
INFO_BYTE_LO AD	RESERVED	ECC_DIS	OPTION_OPR _EN	RESERVED	WRITE_RELEASE SE_EN
w	r	r/w	r/w	r	r/w
5	4	3	2	1	0
PREFETCH_EN	READ_ACC_EN	PROG_MODE	PROG_EN	PAGE_ERASE SE_EN	MASS_ERASE _EN
r/w	r/w	r/w	r/w	r/w	r/w

**Bit 31 INFO\_BYTE\_LOAD:** Info byte load reset request.

- write 0: no action
- write 1: system will reset, and reload the information in the Flash info area, such as options. This bit is automatically cleared by hardware.

**Bits 30-10 RESERVED:** Must be kept, and cannot be modified.

**Bit 9 ECC\_DIS:** ECC encoding disable.

**Bit 8 OPTION\_OPR\_EN:** Option operation enable.

- 0: Option operation disabled
- 1: Option operation enabled

**Notes:**

1. Any two of **OPTION\_OPR\_EN**, **PROG\_EN** and **PAGE\_ERASE\_EN** cannot be enabled at the same time.
2. After each option operation is performed, the system should be reset for the configuration to take effect.

**Bit 7 RESERVED:** Must be kept, and cannot be modified.

**Bit 6 WRITE\_RELEASE\_EN:** When the system executes Flash program, erase (including Mass) and option operations, the AHB bus mode should be selected.

- 0: hold mode
- 1: release mode

**Note:** Once configured in the release mode, the Flash cannot be read or executed during programming/erasing operation, otherwise, the **FLASHBUSY\_ERR** error flag will be set. But you can access the **EFC\_SR** register and wait the operation to be completed.

**Bit 5 PREFETCH\_EN:** Flash instruction prefetch enable.

- 0: prefetch disabled
- 1: prefetch enabled

**Note:** Read acceleration and instruction prefetch can't be enabled at the same time.

**Bit 4 READ\_ACC\_EN:** Flash read acceleration enable.

- 0: read acceleration disabled (hold mode)
- 1: read acceleration enabled (release mode)

**Notes:**

1. When *READ\_NUM < (2^HCLK\_DIV)*, the read acceleration can be enabled. And it must be enabled after *READ\_NUM* and *HCLK\_DIV* configurations are completed.
2. Read acceleration and instruction prefetch can't be enabled at the same time.

**Bit 3 PROG\_MODE:** flash program mode selection.

- 0: single programming mode. In this mode, the data in the *EFC\_PROG\_DATA1* and *EFC\_PROG\_DATA0* registers are written to the specified address in each program.
- 1: WL continuous programming mode. In this mode, a word line (512 Bytes) is programmed to the continuous address of the Flash memory automatically. During the procedure, the software checks the *PROG\_DATA\_WAIT* flag to determine whether to write new data into the *EFC\_PROG\_DATA1* and *EFC\_PROG\_DATA0* registers.

**Notes:**

1. The ECC encoding format in Flash is 64+8, so an even number of words are programmed each time.
2. In WL continuous programming mode, the *WRITE\_RELEASE\_EN* bit should be set to 1. During the programming process, only the *EFC\_SR*, *EFC\_PROG\_DATA1* and *EFC\_PROG\_DATA0* registers can be read or written, the Flash cannot be read or executed.

**Bit 2 PROG\_EN:** Flash programming enable.

- 0: write to the Flash memory does not trigger Flash programming operation
- 1: write to the Flash memory triggers Flash programming operation

**Notes:**

1. In single programming mode, the programming is started by writing data to the 8-Byte aligned Flash address. The data of register *EFC\_PROG\_DATA0* will be written into the low 4-Byte address space, and the data of register *EFC\_PROG\_DATA1* will be written into the high 4-Byte address space.
2. In WL continuous programming mode, programming is started by writing data to the Flash address, and the programming address is accumulated by 8 Bytes until the end of a WL programming.

**Bit 1 PAGE\_ERASE\_EN:** Flash page erasing enable.

- 0: write to the Flash memory does not trigger Flash page erasing operation
- 1: write to the Flash memory triggers Flash page erasing operation

**Bit 0 MASS\_ERASE\_EN:** Flash mass erasing enable

- 0: write to the Flash memory does not trigger Flash mass erasing operation
- 1: write to the Flash memory triggers Flash mass erasing operation

**Notes:**

1. When the bit is set, if there is a write to the address belonging to the Flash main area, mass erase is only performed on the main area; if there is a write to the address belonging to the Flash info area, mass erase is performed on both the main and info areas.
2. **Do not** perform mass erase on the Flash info area, otherwise the chip will be destroyed.

## 10.5.2 EFC\_INT\_EN

Offset: 0x004

Reset Value: 0x00000000

31-9	8	7	6	5
RESERVED	TWO_BIT_ERROR_INT_EN	ONE_BIT_CORRECT_INT_EN	PROG_ERR_INT_EN	PAGE_ERASE_ERR_INT_EN
r	r/w	r/w	r/w	r/w
4	3	2	1	0
OPTION_WR_ERR_INT_EN	FLASHBUSY_ERR_INT_EN	PROG_DATA_WAIT_INT_EN	RESERVED	OPERATION_DONE_INT_EN
r/w	r/w	r/w	r	r/w

**Bits 31-9 RESERVED:** Must be kept, and cannot be modified.

**Bit 8 TWO\_BIT\_ERROR\_INT\_EN:** ECC TWO\_BIT\_ERROR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 7 ONE\_BIT\_CORRECT\_INT\_EN:** ECC ONE\_BIT\_CORRECT interrupt enable.

- 0: disabled
- 1: enabled

**Bit 6 PROG\_ERR\_INT\_EN:** PROG\_ERR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 5 PAGE\_ERASE\_ERR\_INT\_EN:** PAGE\_ERASE\_ERR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 4 OPTION\_WR\_ERR\_INT\_EN:** OPTION\_WR\_ERR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 3 FLASHBUSY\_ERR\_INT\_EN:** FLASHBUSY\_ERR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 2 PROG\_DATA\_WAIT\_INT\_EN:** PROG\_DATA\_WAIT interrupt enable.

- 0: disabled
- 1: enabled

**Bit 1 RESERVED:** Must be kept, and cannot be modified.

**Bit 0 OPERATION\_DONE\_INT\_EN:** OPERATION\_DONE interrupt enable.

- 0: disabled
- 1: enabled

### 10.5.3 EFC\_SR

Offset: 0x008

Reset Value: 0x00000006

<b>31-9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>
RESERVED	TWO_BIT_ERROR	ONE_BIT_CORRECT	PROG_ERR	PAGE_ERASE_ERR
r	r/w	r/w	r/w	r/w
<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
OPTION_WR_ERR	FLASHBUSY_ERR	PROG_DATA_WAIT	READ_NUM_DONE	OPERATION_DONE
r/w	r/w	r/w	r	r/w

**Bits 31-9 RESERVED:** Must be kept, and cannot be modified.

**Bit 8 TWO\_BIT\_ERROR:** TWO\_BIT\_ERROR flag is set when the Flash memory is read.

- 0: no two-bit error occurred
- 1: 1: two-bit error occurred when reading the Flash memory and ECC did not correct

**Bit 7 ONE\_BIT\_CORRECT:** ONE\_BIT\_CORRECT flag is set when the Flash memory is read.

- 0: no one-bit error occurred
- 1: one-bit error occurred when reading the Flash memory and ECC corrected it

**Bit 6 PROG\_ERR:** Some partitions within the Flash info area don't support programming operation(PROG\_EN). Programming operation to these partitions will be blocked, and this bit will be set by hardware and cleared by software writing 1 to it.

- 0: no programming error occurred
- 1: programming error occurred

**Note:** The option area cannot be written by direct program operations. The bootloader area cannot be programmed.

**Bit 5 PAGE\_ERASE\_ERR:** The Flash info area don't support erasing operation. Erasing operation to the info area will be blocked, and this bit will be set by hardware and cleared by software writing 1 to it.

- 0: no page erase error occurred
- 1: a page erase error occurred

**Bit 4 OPTION\_WR\_ERR:** The Option area should be configured with the limitations respected, or the configuration is invalid and this bit is set by hardware. It is cleared by software writing 1 to it.

- 0: no write permission error on Option byte
- 1: a write permission error on Option byte occurred

The configuration for the Option area must respect the following limitations:

1. Flash EXE\_Only1/EXE\_Only2 area can't be disabled or narrowed once it is enabled.
2. Bit EXE\_ONLY\_KEEP can't be modified from 0 to 1.
3. When SECURE\_AREA\_EN=1, operations initiated by non-secure areas only act on the FLASH\_SECURE\_END/FLASH\_SECURE\_START bits in Option bytes to clear the secure\_area\_en status bit.

**Bit 3 FLASHBUSY\_ERR:** When Flash is performing programming, erasing (including mass), and option operations, the read operation by the software will be blocked, the data returned by the bus is uncertain, it is an abnormal state, this bit will be set by hardware and cleared by software writing 1 to it.

- 0: no error occurred
- 1: read error occurred during a Flash operation

**Bit 2 PROG\_DATA\_WAIT:** Waiting for data to be written to the Flash memory in WL continuous programming mode. This bit is set by hardware and is cleared automatically by hardware when the software writes new data to the *FC\_PROG\_DATA0* and *EFC\_PROG\_DATA1* registers. It can also be cleared by software writing 1 to it.

- 0: the value of registers *EFC\_PROG\_DATA0* and *EFC\_PROG\_DATA1* has been written to the Flash memory
- 1: wait for the value of registers *EFC\_PROG\_DATA0* and *EFC\_PROG\_DATA1* to be written to the Flash memory

**Bit 1 READ\_NUM\_DONE:** *READ\_NUM* configuration status flag, it indicates whether the *READ\_NUM* configuration is complete. This bit is set and cleared by hardware.

- 0: in progress
- 1: complete

**Bit 0 OPERATION\_DONE:** Flash operation status flag, it indicates whether Flash mass erase/page erase/program-option operation is complete. This bit is set by hardware and cleared by software writing 1 to it.

- 0: in progress
- 1: complete

#### 10.5.4 EFC\_PROG\_DATA0

Offset: 0x00C

Reset Value: 0x00000000

31-0
PROG_DATA0
r/w

**Bits 31-0 PROG\_DATA0:** programming data 0.

**Note:** When programming, write data to register *EFC\_PROG\_DATA0* first.

#### 10.5.5 EFC\_PROG\_DATA1

Offset: 0x010

Reset Value: 0x00000000

31-0
PROG_DATA1
r/w

**Bits 31-0 PROG\_DATA1:** programming data 1.

**Note:** When programming, write data to register *EFC\_PROG\_DATA0* first.

### 10.5.6 EFC\_TIMING\_CFG

Offset: 0x014

Reset Value: 0x00031D1D

31-20	19-16	15-0
RESERVED	READ_NUM	RESERVED
r	r/w	r

**Bits 31-20 RESERVED:** Must be kept, and cannot be modified.

**Bit 19-16 READ\_NUM:** Flash read operation read wait count control, the read wait count is equal to (READ\_NUM+1) SYS\_CLK clock cycles.

- For 48 MHz SYS\_CLK frequency, READ\_NUM must  $\geq 2$ .
- For 32 MHz SYS\_CLK frequency, READ\_NUM must  $\geq 1$ .
- For 24 MHz SYS\_CLK frequency, READ\_NUM must  $\geq 1$ .
- For 4 MHz SYS\_CLK frequency, READ\_NUM must  $\geq 0$ .
- For 32 kHz SYS\_CLK frequency, READ\_NUM must  $\geq 0$ .

**Note:** When changing the SYS\_CLK clock source in register [RCC\\_CR0](#), pay attention to the sequence of operations. If you intend to switch to a faster clock source, first increase the READ\_NUM, and then configure the clock source selection bit; otherwise, first configure the clock source selection bit, and then decrease the READ\_NUM.

**Bits 15-0 RESERVED:** Must be kept, and cannot be modified.

### 10.5.7 EFC\_PROTECT\_SEQ

Offset: 0x018

Reset Value: 0x00000000

31-0
PROTECT_SEQ
w

**Bits 31-0 PROTECT\_SEQ:** Protection sequence for the configuration of register [EFC\\_CR](#). By default, the EFC\_CR register cannot be modified, to modify it, the user must configure the protection sequence correctly through the FC\_PROTECT\_SEQ register in the following order. If there is an error in the configuration, then the configuration is invalid, and the protection sequence should be reconfigured.

1. Write 0x8C9DAEBF.
2. Write 0x13141516.
3. You can operate with EFC\_CR

### 10.5.8 SERIAL\_NUM\_LOW

Offset: 0x02C

31-0
SERIAL_NUM_LOW
r

**Bits 31-0 SERIAL\_NUM\_LOW:** Less significant 32 bits of the chip serial number.

### 10.5.9 SERIAL\_NUM\_HIGH

Offset: 0x030

31-0
SERIAL_NUM_HIGH
r

**Bits 31-0 SERIAL\_NUM\_HIGH:** Most significant 32 bits of the chip serial number.

### 10.5.10 OPTION\_CSR\_BYTES

Offset: 0x03C

Reset Value: 0x000000BD

31-7	6-5	4	3	2	1	0
RESERVED	DEBUG_LEVEL	SECURE_AREA_EN	SYS_SRAM_RST	FLASH_BOOT1	USE_FLASH_BOOT0	FLASH_BOOT0
r	r	r	r	r	r	r

**Bits 31-7 RESERVED:** Must be kept, and cannot be modified.

**Bits 6-5 DEBUG\_LEVEL:** Debug level setting.

- 0: Level 0 ● 1: Level 1 ●
- 2: Level 2

**Bit 4 SECURE\_AREA\_EN:** Flash secure area status flag.

- 0: secure area disabled
- 1: secure area enabled

**Bit 3 SYS\_SRAM\_RST:** Clear system SRAM during system startup after its reset.

- 0: do not clear system SRAM
- 1: clear system SRAM

**Bit 2 FLASH\_BOOT1:** This bit can be used for boot mode identification. See [Table 7-7](#) for more details.

- 0: System SRAM boot
- 1: BootLoader

**Bit 1 USE\_FLASH\_BOOT0:** This bit can be used for boot mode identification. See *Table 7-7* for more details.

- 0: Use BOOT0 pin
- 1: Use FLASH\_BOOT0 option bit

**Bit 0 FLASH\_BOOT0:** This bit can be used for boot mode identification, and the configuration is only valid when USE\_FLASH\_BOOT0=1. See *Table 7-7* for more details.

- 0: FLASH\_BOOT1 controls the startup mode
- 1: Boot from Main Flash

### 10.5.11 OPTION\_EXE\_ONLY\_BYTES

Offset: 0x040

Reset Value: 0x00FC0FC0

31-25	24	23-18	17-12	11-6	5-0
RESERVED	EXE_ONLY_K EEP	EXE_ONLY2_ END	EXE_ONLY2_ START	EXE_ONLY1_ END	EXE_ONLY1_ START
r	r	r	r	r	r

**Bits 31-25 RESERVED:** Must be kept, and cannot be modified.

**Bit 24 EXE\_ONLY\_KEEP:** Keep Exe\_Only area when the Debug\_Level changes from 1 to 0.

- 0: erase ExeOnly area
- 1: keep the ExeOnly area

This bit can only be cleared by software.

**Bits 23-18 EXE\_ONLY2\_END:** Exe\_Only2 area end offset.

If *EXEONLY2\_START > EXEONLY2\_END*, the ExeOnly2 area is disabled.

**Bits 17-12 EXE\_ONLY2\_START:** Exe\_Only2 area start offset.

If *EXEONLY2\_START > EXEONLY2\_END*, the ExeOnly2 area is disabled. Once enabled, this area can only be expanded but can't be disabled or narrowed.

**Bits 11-6 EXE\_ONLY1\_END:** Exe\_Only1 area end offset.

If *EXEONLY1\_START > EXEONLY1\_END*, the ExeOnly1 area is disabled.

**Bits 5-0 EXE\_ONLY1\_START:** Exe\_Only1 area start offset.

If *EXEONLY1\_START > EXEONLY1\_END*, the ExeOnly1 area is disabled. Once enabled, this area can only be expanded but can't be disabled or narrowed.

### 10.5.12 OPTION\_WR\_PROTECT\_BYTES

Offset: 0x044

Reset Value: 0x0003F03F

31-12	11-6	5-0
RESERVED	WRPROTECT_END	WRPROTECT_START
r	r	r

**Bits 31-12 RESERVED:** Must be kept, and cannot be modified.

**Bits 11-6 WRPROTECT\_END:** Write-protected area end offset.

If *WRPROTECT\_START* > *WRPROTECT\_END*, the write-protected area is disabled.

**Bits 5-0 WRPROTECT\_START:** Write-protected area start offset.

If *WRPROTECT\_START* > *WRPROTECT\_END*, the write-protected area is disabled

### 10.5.13 OPTION\_SECURE\_BYTOS0

Offset: 0x048

Reset Value: 0x00FC0FC0

31-24	23-18	17-12	11-6	5-0
RESERVED	SYSRAM_SECURE_END	SYSRAM_SECURE_START	FLASH_SECURE_END	FLASH_SECURE_START
r	r	r	r	r

**Bits 31-24 RESERVED:** Must be kept, and can't be modified.

**Bits 23-18 SYSRAM\_SECURE\_END:** SysRam Secure area end.

If *SYSRAM\_SECURE\_START* > *SYSRAM\_SECURE\_END*, the SysRam Secure area is disabled.

The configuration is only valid when *SECURE\_AREA\_EN*=1.

**Bits 17-12 SYSRAM\_SECURE\_START:** SysRam Secure area start.

If *SYSRAM\_SECURE\_START* > *SYSRAM\_SECURE\_END*, the SysRam Secure area is disabled.

The configuration is only valid when *SECURE\_AREA\_EN*=1.

**Bits 11-6 Flash Secure area end.**

If *FLASH\_SECURE\_START* > *FLASH\_SECURE\_END*, the Flash Secure area is disabled.

**Bits 5-0 FLASH\_SECURE\_START:** Flash Secure area start.

If *FLASH\_SECURE\_START* > *FLASH\_SECURE\_END*, the Flash Secure area is disabled.

The Flash Secure area enable is the master switch for enabling other secure areas.

If the Flash Secure area is enabled, the *SECURE\_AREA\_EN* bit is set, which means all the other secure areas can be enabled.

If the Flash Secure area is disabled, the *SECURE\_AREA\_EN* bit is cleared, which triggers the erase operation.

### 10.5.14 OPTION\_SECURE\_BYTES1

Offset: 0x04C

Reset Value: 0x008103E0

31-24	23		22-17
RESERVED	SYSRAM_HIDE_ENABLE		SYSRAM_HIDE_START
r	r		r
16	15-10	9-5	4-0
FLASH_HIDE_ENABLE	FLASH_HIDE_START	RETRAM_SECURE_END	RETRAM_SECURE_START
r	r	r	r

**Bits 31-24 RESERVED:** Must be kept, and can't be modified.

**Bit 23 SYSRAM\_HIDE\_ENABLE:** SysRamHide area enable control.

- 0: SysRamHide area enabled
- 1: SysRamHide area disabled

The configuration is only valid when SECURE\_AREA\_EN=1.

**Bits 22-17 SYSRAM\_HIDE\_START:** SysRamHide area start.

The configuration is only valid when the SysRamHide area is within the SysRamSecure area and when SECURE\_AREA\_EN=1.

The SysRamHide area is from SYSRAM\_HIDE\_START to SYSRAM\_SECURE\_END.

**Bit 16 FLASH\_HIDE\_ENABLE:** FlashHide area enable control.

- 0: FlashHide area enabled
- 1: FlashHide area disabled

The configuration is only valid when SECURE\_AREA\_EN=1.

**Bits 15-10 FLASH\_HIDE\_START:** FlashHide area start.

The configuration is only valid when the FlashHide area is within the FlashSecure area and when SECURE\_AREA\_EN=1.

The FlashHide area is from FLASH\_HIDE\_START to FLASH\_SECURE\_END.

**Bits 9-5 RETRAM\_SECURE\_END:** RetRam Secure area end.

If RETRAM\_SECURE\_START > RETRAM\_SECURE\_END, the RetRam Secure area is disabled.

The configuration is only valid when SECURE\_AREA\_EN=1.

**Bits 4-0 RETRAM\_SECURE\_START:** RetRam Secure area start.

If RETRAM\_SECURE\_START > RETRAM\_SECURE\_END, the RetRam Secure area is disabled.

The configuration is only valid when SECURE\_AREA\_EN=1.

# 11.

# GPIO

## 11.1 Introduction

ASR6601 GPIOs are divided into four groups: Ports A, B, C, and D. The SFR registers of each group are allocated the same, and they are distinguished by different base addresses. PortD Pin8 ~ Pin15 are located in the AON domain, and the other IOs are located in the Main domain.

All GPIOs support input and output, pull-up and pull-down, push-pull output and open-drain output. The output drive current can be configured as 4mA or 8mA. All GPIOs can generate interrupts, which can be triggered by rising edge, falling edge or both edges. In Sleep/Stop0~2 mode, all GPIOs can be used for wake-up; while in Stop3 mode, only some GPIOs can be used to wake-up MCU. All GPIOs support alternate functions.

## 11.2 Output Configuration

GPIO data output is configured by the *GPIOx\_OER* and *GPIOx\_ODR* registers.

GPIO output can be set or cleared. Writing 1 to bits[15:0] in register *GPIOx\_BRR* or writing 1 to bits[31-16] in register *GPIOx\_BSRR* can **clear** the corresponding bit in register *GPIOx\_ODR*. And writing 1 to bits[15:0] in register *GPIOx\_BSRR* can **set** the corresponding bit in register *GPIOx\_ODR*.

GPIO port is configured as **push-pull** output through register *GPIOx\_OTYPER*. As to output in **open-drain** mode, for PortD Pin8 ~ PortD Pin15, it is enabled by configuring the *GPIOx\_IER*, *GPIOx\_OER*, *GPIOx\_ODR* and *GPIOx\_PSR* registers, and for other IO ports, it is enabled by configuring the *GPIOx\_OER*, *GPIOx\_IER*, *GPIOx\_ODR* and *GPIOx\_OTYPER* registers. Not implementing a real open drain structure, the open drain function is achieved by control of the *GPIOx\_OER* and *GPIOx\_ODR* registers.

GPIO can be configured as analog output.

## 11.3 Input Configuration

GPIO data input is enabled by configuring register *GPIOx\_IER*, and you can read register *GPIOx\_IDR* to get the input status.

Input floating mode is realized by configuring register *GPIOx\_PER* to disable pull-up and pull-down.

Pull-up or pull-down is enabled by configuring register *GPIOx\_PER*, and register *GPIOx\_PSR* is used for pull-up or pull-down selection.

GPIO can be configured as analog input.

## 11.4 Output Drive Strength

High (8 mA) or low (4 mA) output drive strength is configured by [GPIOx\\_DSR](#) register.

## 11.5 GPIO Interrupts

All GPIOs support interrupts, which can be triggered by rising edge, falling edge or both edges.

Interrupts are enabled by configuring [GPIOx\\_INT\\_CR](#) register.

## 11.6 Wakeup from Sleep/Stop0~2 Mode

In Sleep or Stop 0/1/2 mode, MCU can be woken up at high level or low level, and the output wake-up signal is high level. GPIO00-GPIO63 can all be used for wakeup, four IOs make up a group. A group can generate a wakeup signal, and each IO in a group can wake up MCU at high level or low level. In Sleep/Stop0~2 mode, the wakeup function is enabled by configuring the [GPIOx\\_WU\\_EN](#) register, and the high-level or low-level wakeup is selected by configuring the [GPIOx\\_WU\\_LVL](#) register.

## 11.7 Wakeup from Stop3 Mode

GPIO00~GPIO55 of the Main domain, every 4 IO MUXs output a wake-up signal, a total of 14 wake-up signals. Support high-level wake-up or low-level wake-up and wake-up enable control, which is achieved by configuring the Stop3 wake-up enable register [GPIOx\\_STOP3\\_WU\\_CR](#).

## 11.8 Alternate Function Configuration

GPIO can be used as general I/O or configured as alternate function. GPIO input/output is enabled or disabled by the [GPIOx\\_OER](#) and [GPIOx\\_IER](#) registers, while the alternate function input/output is enabled or disabled by alternate peripherals. The I/O pull-up or pull-down is configured by the [GPIOx\\_PER](#) and [GPIOx\\_PSR](#) registers.

As to alternate function control, 3-bit for each pin among PortD Pin8~Pin15, and 4-bit for each of the other pins. By default, PortA Pin6 and Pin7 are configured as SWD pins, and the other IOs are configured as GPIO.

The function of Portx Pin[7:0] is configured through the lower 8 Pin function MUX selection register [GPIOx\\_AFRL](#), and the function of Portx Pin[15:8] is configured through the upper 8 Pin function MUX selection register [GPIOx\\_AFRH](#).

## 11.9 Clock and Reset

There are four groups of APB bus clock and APB bus reset, each group has an independent bus clock and bus reset.

## 11.10 Power Domains

### Main Domain:

Except for PortD Pin8~PortD Pin15, the corresponding pads are all in the Main domain.

### AON (always-on) Domain:

The PADs corresponds to PortD Pin8~Pin15 are in the AlwaysOn domain. If they are configured as alternate function, they are directly controlled by the peripherals. Otherwise, they will be controlled by the GPIO registers in the AlwaysOn domain.

## 11.11 Low-power Mode Operation and Wakeup

1. In Sleep mode, all GPIOs can work and output wake-up signal.
2. In Stop0/Stop1/Stop2 mode, all GPIOs can work and output wake-up signal.
3. In Stop3 mode, GPIO00~GPIO55 can retain the state, and can be configured as wake-up signal.
4. In Stop3 mode, PortD Pin8~Pin15 in AlwaysOn domain can retain the state, CPU can also be woken up through RTC.
5. In Standby mode, PortD Pin8~PortD Pin15 can work, while the other IOs can't work.

## 11.12 SWD IO

**Default Control:** The GPIO alternate function low register selects SWD by default, and SWCLK pull-down (PortA Pin7) and SWDIO pull-up (PortA Pin6) are default.

**Sealing control:** After power-on, the IO status is controlled by the default state of the register until the DebugLevel judgment is completed. If sealing is found to be necessary, permanent sealing is performed; otherwise, it continues to be controlled by the register.

**Software configuration:** During software operation, the SWD can be disabled by controlling the multiplexing register. Note that it is a one-way seal, that is, it can only be disabled, and cannot be disabled and then enabled.

## 11.13 BOOT0 Control

**Default Control:** Since all IOs except the SWCLK and SWDIO IOs are analog IOs by default, the BOOT0, SWCLK and SWDIO pins require special control at power-on.

**BOOT0 (GPIO02):** BOOT0 is in input pull-down status before io\_lock. After EFC is locked, it switches to GPIO mode.

## 11.14 GPIO Registers

GPIO Port A Base Address: 0x4001F000

GPIO Port B Base Address: 0x4001F400

GPIO Port C Base Address: 0x4001F800

GPIO Port D Base Address: 0x4001FC00

**Table 11-1 GPIO Registers Summary**

Register	Offset	Description
GPIOx_OER	0x000	General output enable register
GPIOx_OTYPER	0x004	General output type control register
GPIOx_IER	0x008	General input enable register
GPIOx_PER	0x00C	Pull-up/pull-down enable register
GPIOx_PSR	0x010	Pull-up/pull-down selection register
GPIOx_IDR	0x014	Input data register
GPIOx_ODR	0x018	Output data register
GPIOx_BRR	0x01C	Bit reset register
GPIOx_BSRR	0x020	Bit set or reset register
GPIOx_DSR	0x024	Output drive strength register
GPIOx_INT_CR	0x028	Interrupt enable register
GPIOx_FR	0x02C	Interrupt edge flag register
GPIOx_WU_EN	0x030	Wake-up from Sleep/Stop0~2 mode enable register
GPIOx_WU_LVL	0x034	Wake-up level control register for Sleep/Stop0~2 mode
GPIOx_AFRL	0x038	GPIO alternate function low register
GPIOx_AFRH	0x03C	GPIO alternate function high register
GPIOx_STOP3_WU_CR	0x040	Wake-up from Stop3 mode enable/control register

### 11.14.1 GPIOx\_OER (x=A, B, C, D)

Offset: 0x000

Reset Value: 0x0000FFFF

31-16	15-0
RESERVED	OEN
r-0h	rw-ffffh

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 OEN:** Portx Pin[15:0] output enable.

- 0: output enabled
- 1: output disabled

### 11.14.2 GPIOx\_OTYPER (x=A, B, C, D)

Offset: 0x004

Reset Value: 0x00000000

31-16	15-0
RESERVED	OTYPE
r-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 OTYPE:** Portx Pin[15:0] output type control.

- 0: push-pull
- 1: open-drain

**Note:** The output type of the pads in the AON domain (PortD\_Pin[15:8]) is controlled by the [GPIOx\\_IER](#), [GPIOx\\_OER](#), [GPIOx\\_ODR](#) and [GPIOx\\_PSR](#) registers instead of this register. For the other pins, the open drain mode is enabled through the [GPIOx\\_IER](#), [GPIOx\\_OER](#), [GPIOx\\_ODR](#) and [GPIOx\\_OTYPER](#) registers.

### 11.14.3 GPIOx\_IER (x=A, B, C, D)

Offset: 0x008

Reset Value: 0x00000000

31-16	15-0
RESERVED	IE
r-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 IE:** Portx Pin[15:0] input enable.

- 0: input disabled

- 1: input enabled

#### 11.14.4 GPIOx\_PER (x=A, B, C, D)

Offset: 0x0C

Reset Value: 0x00000000

31-16	15-0
RESERVED	PE
r-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 PE:** Portx Pin[15:0] pull-up/pull-down enable.

- 0: pull-up/pull-down disabled
- 1: pull-up/pull-down enabled

GPIO pull-up and pull-down is selected by the *GPIOx\_PSR* register. By default, pull-up/pull-down is disabled, and all the IOs except PortA\_Pin[7:6] are in analog mode. PortA\_Pin[7:6] are used as SWD function.

#### 11.14.5 GPIOx\_PSR (x=A, B, C, D)

Offset: 0x010

Reset Value: 0x00000000

31-16	15-0
RESERVED	PS
r-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 PS:** Portx Pin[15:0] pull-up/pull-down selection.

- 0: pull-down
- 1: pull-up

#### 11.14.6 GPIOx\_IDR (x=A, B, C, D)

Offset: 0x014

Reset Value: 0x00000000

31-16	15-0
RESERVED	ID
r-0h	r-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 ID:** Portx Pin[15:0] input.

- 0: low level

- 1: high level

#### 11.14.7 GPIOx\_ODR (x=A, B, C, D)

Offset: 0x018

Reset Value: 0x00000000

31-16	15-0
RESERVED	OD
r-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 OD:** Portx Pin[15:0] output.

- 0: low level
- 1: high level

#### 11.14.8 GPIOx\_BRR (x=A, B, C, D)

Offset: 0x01C

Reset Value: 0x00000000

31-16	15-0
RESERVED	BR
r-0h	w-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 BR:** Portx Pin[15:0] output data clear.

- 0: no action
- 1: clear the corresponding bit of the GPIOx\_ODR register

#### 11.14.9 GPIOx\_BSRR (x=A, B, C, D)

Offset: 0x020

Reset Value: 0x00000000

31-16	15-0
BR	BSR
w-0h	w-0h

**Bits 31-16 BR:** Portx Pin[15:0] output data clear.

- 0: no action
- 1: clear the corresponding bit of the GPIOx\_ODR register

**Note:** If BSR and BR are both valid, BSR has higher priority.

**Bits 15-0 BSR:** Portx Pin[15:0] output data set.

- 0: no action
- 1: set the corresponding bit of the GPIOx\_ODR register

**Note:** If BSR and BR are both valid, BSR has higher priority.

#### 11.14.10 GPIOx\_DSR (x=A, B, C, D)

Offset: 0x024

Reset Value: 0x00000000

31-16	15-0
RESERVED	DS
r-0h	w-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 DS:** Portx Pin[15:0] output drive strength configuration.

- 0: low drive strength (4 mA)
- 1: high drive strength (8 mA)

#### 11.14.11 GPIOx\_INT\_CR (x=A, B, C, D)

Offset: 0x028

Reset Value: 0x00000000

2*n + 1	2*n
NEG_INT_EN	POS_INT_EN
rw-0h	rw-0h

**Bits 2\*n + 1 NEG\_INT\_EN:** Portx Pin[15:0] enable interrupt triggered by falling edge.

- 0: interrupt triggered by falling edge disabled
- 1: interrupt triggered by falling edge enabled

**Bits 2\*n POS\_INT\_EN:** Portx Pin[15:0] enable interrupt triggered by rising edge.

- 0: interrupt triggered by rising edge disabled
- 1: interrupt triggered by rising edge enabled

### 11.14.12 GPIOx\_FR (x=A, B, C, D)

Offset: 0x02C

Reset Value: 0x00000000

<b>2*n + 1</b>	<b>2*n</b>
NEG_F	POS_F
rw1c-0h	rw1c-0h

**Bits 2\*n + 1 NEG\_INT\_F:** Portx Pin[15:0] interrupt flag (falling edge)

- 0: no interrupt triggered by falling edge occurred
- 1: interrupt triggered by falling edge occurred

**Bits 2\*n POS\_INT\_F:** Portx Pin[15:0] interrupt flag (rising edge)

- 0: no interrupt triggered by rising edge occurred
- 1: interrupt triggered by rising edge occurred

### 11.14.13 GPIOx\_WU\_EN (x=A, B, C, D)

Offset: 0x030

Reset Value: 0x00000000

<b>31-16</b>	<b>15-0</b>
RESERVED	WU_EN
r-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 WU\_EN:** Enable/disable Portx Pin[15:0] to wake-up CPU from Sleep/Stop0~2 modes.

- 0: disable Sleep/Stop0~2 wakeup
- 1: enable Sleep/Stop0~2 wakeup

### 11.14.14 GPIOx\_WU\_LVL (x=A, B, C, D)

Offset: 0x034

Reset Value: 0x00000000

<b>31-16</b>	<b>15-0</b>
RESERVED	WU_LVL
r-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 WU\_LVL:** Configure Portx Pin[15:0] CPU Sleep/Stop0~2 mode wakeup to high or low level.

- 0: wake-up at low level
- 1: wake-up at high level

### 11.14.15 GPIOx\_AFRL (x=A, B, C, D)

Offset: 0x038

Reset Value: 0x00000000

31-28	27-24	23-20	19-16	15-12	11-8	7-4	3-0
AF7	AF6	AF5	AF4	AF3	AF2	AF1	AF0
rw-0h							

**Bits 31-28 AF7:** Portx Pin7 function selection

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 27-24 AF6:** Portx Pin6 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 23-20 AF5:** Portx Pin5 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 19-16 AF4:** Portx Pin4 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2

- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 15-12 AF3:** Portx Pin3 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 11-8 AF2:** Portx Pin2 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 7-4 AF1:** Portx Pin1 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 3-0 AF0:** Portx Pin0 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5

- 0110: Function6
- 0111: Function7
- others: Reserved

### 11.14.16 GPIOx\_AFRH (x=A, B, C)

Offset: 0x03C

Reset Value: 0x00000000

31-28	27-24	23-20	19-16	15-12	11-8	7-4	3-0
AF15	AF14	AF13	AF12	AF11	AF10	AF9	AF8
rw-0h							

**Bits 31-28 AF15:** Portx Pin15 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 27-24 AF14:** Portx Pin14 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 23-20 AF13:** Portx Pin13 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 19-16 AF12:** Portx Pin12 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 15-12 AF11:** Portx Pin11 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 11-8 AF10:** Portx Pin10 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 7-4 AF9:** Portx Pin9 function selection.

- 0000: Function0
- 0001: Function1
- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

**Bits 3-0 AF8:** Portx Pin8 function selection.

- 0000: Function0
- 0001: Function1

- 0010: Function2
- 0011: Function3
- 0100: Function4
- 0101: Function5
- 0110: Function6
- 0111: Function7
- others: Reserved

### 11.14.17 GPIOD\_AFRH

Offset: 0x03C

Reset Value: 0x00000000

31-24	23-21	20-18	17-15	14-12	11-9	8-6	5-3	2-0
RESERVED	AF15	AF14	AF13	AF12	AF11	AF10	AF9	AF8
r-0h	rw-0h							

**Bits 31-24 RESERVED:** Must be kept, and cannot be modified.

- 001: Function1
- 010: Function2
- 011: Function3
- 100: Function4
- 101: Function5
- 110: Function6
- 111: Function7

**Bits 23-21 AF15:** PortD Pin15 function selection.

- 001: Function1
- 010: Function2
- 011: Function3
- 100: Function4
- 101: Function5
- 110: Function6
- 111: Function7

**Bits 20-18 AF14:** PortD Pin14 function selection.

- 001: Function1
- 010: Function2
- 011: Function3
- 100: Function4
- 101: Function5
- 110: Function6
- 111: Function7

**Bits 17-15 AF13:** PortD Pin13 function selection.

- 001: Function1

- 010: Function2
- 011: Function3
- 100: Function4
- 101: Function5
- 110: Function6
- 111: Function7

**Bits 14-12 AF12:** PortD Pin12 function selection.

- 001: Function1
- 010: Function2
- 011: Function3
- 100: Function4
- 101: Function5
- 110: Function6
- 111: Function7

**Bits 11-9 AF11:** Pin11 function selection.

- 001: Function1
- 010: Function2
- 011: Function3
- 100: Function4
- 101: Function5
- 110: Function6
- 111: Function7

**Bits 8-6 AF10:** PortD Pin10 function selection.

- 001: Function1
- 010: Function2
- 011: Function3
- 100: Function4
- 101: Function5
- 110: Function6
- 111: Function7

**Bits 5-3 AF9:** PortD Pin9 function selection.

- 001: Function1
- 010: Function2
- 011: Function3
- 100: Function4
- 101: Function5
- 110: Function6
- 111: Function7

**Bits 2-0 AF8:** PortD Pin8 function selection.

- 001: Function1
- 010: Function2
- 011: Function3

- 100: Function4
- 101: Function5
- 110: Function6
- 111: Function7

### 11.14.18 GPIOA\_STOP3\_WU\_CR

Offset: 0x040

Reset Value: 0x00000000

31-16	15	14	13-12	11
RESERVED	STOP3_WU_EN_G1	STOP3_WU_LVL_G3	STOP3_WU_SEL_G3	STOP3_WU_EN_G2
r-0h	rw-0h	rw-0h	rw-0h	rw-0h
10	9-8	7	6	
STOP3_WU_LVL_G2	STOP3_WU_SEL_G2	STOP3_WU_EN_G1	STOP3_WU_LVL_G1	
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h
5-4	3	2	1-0	
STOP3_WU_SEL_G1	STOP3_WU_EN_G0	STOP3_WU_LVL_G0	STOP3_WU_SEL_G0	
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bit 15 STOP3\_WU\_EN\_G3:** PortA Group3 wake-up pin enable control in Stop3 mode.

- 0: disabled
- 1: enabled

**Bit 14 STOP3\_WU\_LVL\_G3:** PortA Group3 wake-up pin level selection in Stop3 mode.

- 0: low level
- 1: high level

**Bits 13-12 STOP3\_WU\_SEL\_G3:** PortA Pin Group3 wake-up source selection in Stop3 mode.

- 00: PortA Pin6
- 01: PortA Pin7
- 10: PortA Pin14
- 11: PortA Pin15

**Bit 11 STOP3\_WU\_EN\_G2:** PortA Group2 wake-up pin enable control in Stop3 mode.

- 0: disabled
- 1: enabled

**Bit 10 STOP3\_WU\_LVL\_G2:** PortA Group2 wake-up pin level selection in Stop3 mode.

- 0: low level
- 1: high level

**Bits 9-8 STOP3\_WU\_SEL\_G2:** PortA Pin Group2 wake-up source selection in Stop3 mode.

- 00: PortA Pin8
- 01: PortA Pin9
- 10: PortA Pin10
- 11: PortA Pin11

**Bit 7 STOP3\_WU\_EN\_G1:** PortA Group1 wake-up pin enable control in Stop3 mode.

- 0: disabled
- 1: enabled

**Bit 6 STOP3\_WU\_LVL\_G1:** PortA Group1 wake-up pin level selection in Stop3 mode.

- 0: low level
- 1: high level

**Bits 5-4 STOP3\_WU\_SEL\_G1:** PortA Pin Group1 wake-up source selection in Stop3 mode.

- 00: PortA Pin4
- 01: PortA Pin5
- 10: PortA Pin12
- 11: PortA Pin13

**Bit 3 STOP3\_WU\_EN\_G0:** PortA Group0 wake-up pin enable control in Stop3 mode.

- 0: disabled
- 1: enabled

**Bit 2 STOP3\_WU\_LVL\_G0:** PortA Group0 wake-up pin level selection in Stop3 mode.

- 0: low level
- 1: high level

**Bits 1-0 STOP3\_WU\_SEL\_G0:** PortA Pin Group0 wake-up source selection in Stop3 mode.

- 00: PortA Pin0
- 01: PortA Pin1
- 10: PortA Pin2
- 11: PortA Pin3

### 11.14.19 GPIOx\_STOP3\_WU\_CR (x=B, C)

Offset: 0x040

Reset Value: 0x00000000

31-16	15	14	13-12	11
RESERVED	STOP3_WU_EN_G3	STOP3_WU_LVL_G3	STOP3_WU_SEL_G3	STOP3_WU_EN_G2
r-0h	rw-0h	rw-0h	rw-0h	rw-0h
10	9-8	7	6	
STOP3_WU_LVL_G2	STOP3_WU_SEL_G2	STOP3_WU_EN_G1	STOP3_WU_LVL_G1	
rw-0h	rw-0h	rw-0h	rw-0h	
5-4	3	2	1-0	
STOP3_WU_SEL_G1	STOP3_WU_EN_G0	STOP3_WU_LVL_G0	STOP3_WU_SEL_G0	
rw-0h	rw-0h	rw-0h	rw-0h	

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bit 15 STOP3\_WU\_EN\_G3:** Portx Pin Group3 wake-up enable control in Stop3 mode.

- 0: disabled
- 1: enabled

**Bit 14 STOP3\_WU\_LVL\_G3:** Portx Pin Group3 wake-up level selection in Stop3 mode.

- 0: low level
- 1: high level

**Bits 13-12 STOP3\_WU\_SEL\_G3:** Portx Pin Group3 wake-up source selection in Stop3 mode.

- 00: Portx Pin12
- 01: Portx Pin13
- 10: Portx Pin14
- 11: Portx Pin15

**Bit 11 STOP3\_WU\_EN\_G2:** Portx Pin Group2 wake-up enable control in Stop3 mode.

- 0: disabled
- 1: enabled

**Bit 10 STOP3\_WU\_LVL\_G2:** Portx Pin Group2 wake-up level selection in Stop3 mode.

- 0: low level
- 1: high level

**Bits 9-8 STOP3\_WU\_SEL\_G2:** Portx Pin Group2 wake-up source selection in Stop3 mode.

- 00: Portx Pin8
- 01: Portx Pin9
- 10: Portx Pin10
- 11: Portx Pin11

**Bit 7 STOP3\_WU\_EN\_G1:** Portx Pin Group1 wake-up enable control in Stop3 mode.

- 0: disabled
- 1: enabled

**Bit 6 STOP3\_WU\_LVL\_G1:** Portx Pin Group1 wake-up level selection in Stop3 mode.

- 0: low level
- 1: high level

**Bits 5-4 STOP3\_WU\_SEL\_G1:** Portx Pin Group1 wake-up source selection in Stop3 mode.

- 00: Portx Pin4
- 01: Portx Pin5
- 10: Portx Pin6
- 11: Portx Pin7

**Bit 3 STOP3\_WU\_EN\_G0:** Portx Pin Group0 wake-up enable control in Stop3 mode.

- 0: disabled
- 1: enabled

**Bit 2 STOP3\_WU\_LVL\_G0:** Portx Pin Group0 wake-up level selection in Stop3 mode.

- 0: low level
- 1: high level

**Bits 1-0 STOP3\_WU\_SEL\_G0:** Portx Pin Group0 wake-up source selection in Stop3 mode.

- 00: Portx Pin0
- 01: Portx Pin1
- 10: Portx Pin2
- 11: Portx Pin3

### 11.14.20 GPIOD\_STOP3\_WU\_CR

Offset: 0x040

Reset Value: 0x00000000

31-8		7	6
RESERVED		STOP3_WU_EN_G1	STOP3_WU_LVL_G1
r-0h		rw-0h	rw-0h
5-4	3	2	1-0
STOP3_WU_SEL_G1	STOP3_WU_EN_G0	STOP3_WU_LVL_G0	STOP3_WU_SEL_G0
rw-0h	rw-0h	rw-0h	rw-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bit 7 STOP3\_WU\_EN\_G1:** PortD Pin Group1 wake-up enable control in Stop3 mode.

- 0: disabled
- 1: enabled

**Bit 6 STOP3\_WU\_LVL\_G1:** PortD Pin Group1 wake-up level selection in Stop3 mode.

- 0: low level
- 1: high level

**Bits 5-4 STOP3\_WU\_SEL\_G1:** PortD Pin Group1 wake-up source selection in Stop3 mode.

- 00: PortD Pin4
- 01: PortD Pin5
- 10: PortD Pin6
- 11: PortD Pin7

**Bit 3 STOP3\_WU\_EN\_G0:** PortD Pin Group0 Stop3 wake-up enable control in Stop3 mode.

- 0: disabled
- 1: enabled

**Bit 2 STOP3\_WU\_LVL\_G0:** PortD Pin Group0 Stop3 wake-up level selection in Stop3 mode.

- 0: low level
- 1: high level

**Bits 1-0 STOP3\_WU\_SEL\_G0:** PortD Pin Group0 Stop3 wake-up source selection in Stop3 mode.

- 00: PortD Pin0
- 01: PortD Pin1
- 10: PortD Pin2
- 11: PortD Pin3

# 12.

# LoRa Controller (LoRaC)

## 12.1 Introduction

LoRa Controller is used to control the internal RF TRX to transmit and reception LoRa signals.

## 12.2 Main Features

- SPI interface for RF TRX control
- Interrupt signal generation

## 12.3 Functional Description

### 12.3.1 Internal SPI Interface

There is an internal SPI interface in the LoRa Controller, which allows the LoRa Controller to directly control RF TRX through registers. The communication between the MCU and RF TRX is as follows:

- (1) Initialize the internal SSP in LoRa Controller.
- (2) Check the BUSY\_DIG\_SR bit in register [LORAC\\_SR](#) is 0, if it is 0, it means that RF TRX is currently free for communication.
- (3) Clear REG\_NSS bit in register [LORAC\\_NSS\\_CR](#).
- (4) Write data into register [SSP\\_DR](#) which belonging to the internal SSP of LoRa Controller.
- (5) Wait for the transmission to be completed.
- (6) Read back the data through register [SSP\\_DR](#).
- (7) Repeat Steps 4 ~ Step 6 as required.
- (8) Set REG\_NSS bit in register [LORAC\\_NSS\\_CR](#).

### 12.3.2 Timing Sequence of Power-on

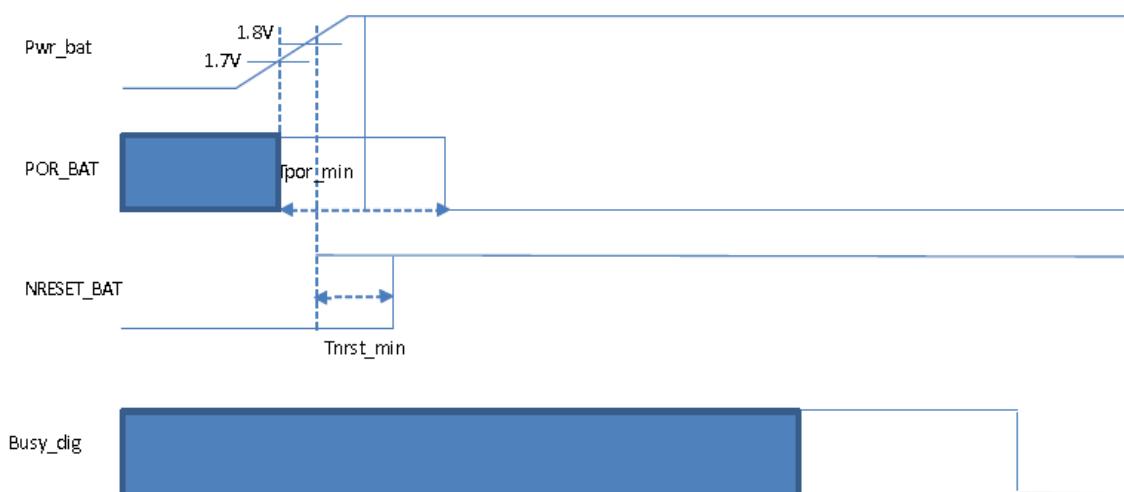


Figure 12-1 Power-on Timing Sequence

As shown in the figure above, the process of power-on is:

- (1) Set NRESET\_BAT bit in register *LORAC\_CR1*.
- (2) Clear POR\_BAT bit in register *LORAC\_CR1*.
- (3) Wait for the BUSY\_DIG\_SR bit in register *LORAC\_SR* to be cleared.

T<sub>por\_min</sub> is 100  $\mu$ s and T<sub>nrst\_min</sub> is 50  $\mu$ s.

### 12.3.3 Interrupts

The LoRa Controller transparently transmits the RF TRX interrupt request, and generates the interrupt signal. Note that once the interrupt request of the LoRa Controller is triggered, software must send the *ClearIrqStatus* command to the RF TRX to clear the interrupt, otherwise the interrupt request will be triggered again.

## 12.4 LoRaC Registers

LORAC Base Address: 0x40009000

**Table 12-1 LORAC Registers Summary**

Register	Offset	Description
SSP_CRO	0x000	LORAC Internal SSP Control Register 0
SSP_CR1	0x004	LORAC Internal SSP Control Register 1
SSP_DR	0x008	LORAC Internal SSP Data Register
SSP_SR	0x00C	LORAC Internal SSP Status Register
SSP_CPSR	0x010	LORAC Internal SSP Clock Prescaler Register
SSP_IMSC	0x014	LORAC Internal SSP Interrupt Mask Set/Clear Register
SSP_RIS	0x018	LORAC Internal SSP Raw Interrupt Status register
SSP_MIS	0x01C	LORAC Internal SSP Masked Interrupt Status register
SSP_ICR	0x020	LORAC Internal SSP Interrupt Clear Register
SSP_DMACR	0x024	LORAC Internal SSP DMA Control Register
RESERVED	0x028-0x0FC	Must be kept, and cannot be modified
LORAC_CR0	0x100	LORAC Control Register 0
LORAC_CR1	0x104	LORAC Control Register 1
LORAC_SR	0x108	LORAC Status Register
LORAC_NSS_CR	0x10C	LORAC NSS Control Register
LORAC_SCK_CR	0x110	LORAC SCK Control Register
LORAC_MOSI_CR	0x114	LORAC MOSI Control Register
LORAC_MISO_SR	0x118	LORAC MISO Status Register

### 12.4.1 SSP\_CR0

Offset: 0x000

Reset Value: 0x00000000

31-16	15-8	7	6	5-4	3-0
RESERVED	SCR	SPH	SPO	FRF	DSS
r	r/w	r/w	r/w	r/w	r/w

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-8 SCR:** Serial clock rate, used to set the SSP transfer rate.

$$F_{SSPCLKOUT} = \frac{F_{SSPCLK}}{CPSDVR \times (1+SCR)}$$

The formula to calculate the SSP transfer rate is as above, where CPSDVR is an even number ranging from 2 to 254.

**Bit 7 SPH:** SSP phase setting, only applied in Motorola SPI format.

**Bit 6 SPO:** SSP polarity setting, only applied in Motorola SPI format.

**Bits 5-4 FRF:** SSP frame formats setting.

- 0: Motorola SPI
- 1: Texas Instruments SPI
- 2: National Semiconductor Microwire
- 3: reserved

**Bits 3-0 DSS:** Data width setting.

- 0: reserved
- 1: reserved
- 2: reserved
- 3: 4 bits
- 4: 5 bits
- 5: 6 bits
- 6: 7 bits
- 7: 8 bits
- 8: 9 bits
- 9: 10 bits
- 10: 11 bits
- 11: 12 bits
- 12: 13 bits
- 13: 14 bits
- 14: 15 bits
- 15: 16 bits

### 12.4.2 SSP\_CR1

Offset: 0x004

Reset Value: 0x00000000

31-4	3	2	1	0
RESERVED	SOD	MS	SSE	LBM
r	r/w	r/w	r/w	r/w

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 SOD:** SSP output disable in slave mode.

- 0: SSP output enabled in slave mode
- 1: SSP output disabled in slave mode

**Bit 2 MS:** Master/slave mode selection.

- 0: master mode
- 1: slave mode

**Bit 1 SSE:** SSP enable.

- 0: disabled
- 1: enabled

**Bit 0 LBM:** loopback mode.

- 0: normal mode
- 1: loopback mode

### 12.4.3 SSP\_DR

Offset: 0x008

Reset Value: 0x00000000

31-16	15-0
RESERVED	DATA
r	r/w

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 DATA:** SSP TX/RX data.

#### 12.4.4 SSP\_SR

Offset: 0x00C

Reset Value: 0x00000003

31-5	4	3	2	1	0
RESERVED	BSY	RFF	RNE	TNF	TFE
r	r	r	r	r	r

**Bits 31-5 RESERVED:** Must be kept, and cannot be modified.

**Bit 4 BSY:** SSP busy flag.

- 0: SSP is idle
- 1: SSP transfer in progress

**Bit 3 RFF:** RX FIFO full flag.

- 0: RX FIFO is not full
- 1: RX FIFO is full

**Bit 2 RNE:** RX FIFO not empty flag.

- 0: RX FIFO is empty
- 1: RX FIFO is not empty

**Bit 1 TNF:** TX FIFO not full flag.

- 0: TX FIFO is full
- 1: TX FIFO is not full

**Bit 0 TFE:** TX FIFO empty flag.

- 0: TX FIFO is not empty
- 1: TX FIFO is empty

#### 12.4.5 SSP\_CPSR

Offset: 0x010

Reset Value: 0x00000000

31-8	7-0
RESERVED	CPSDVS
r	r/w

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 CPSDVS:** Clock prescaler divider, must be an even number between 2~254.

## 12.4.6 SSP\_IMSC

Offset: 0x014

Reset Value: 0x00000000

31-4	3	2	1	0
RESERVED	TXIM	RXIM	RTIM	RORIM
r	r/w	r/w	r/w	r/w

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 TXIM:** TX interrupt mask bit.

- 0: TX interrupt is masked
- 1: TX interrupt is not masked

**Bit 2 RXIM:** RX interrupt mask bit.

- 0: RX interrupt is masked
- 1: RX interrupt is not masked

**Bit 1 RTIM:** RX timeout interrupt mask bit.

- 0: RX timeout interrupt is masked
- 1: RX timeout interrupt is not masked

**Bit 0 RORIM:** RX overrun interrupt mask bit.

- 0: RX overrun interrupt is masked
- 1: RX overrun interrupt is not masked

## 12.4.7 SSP\_RIS

Offset: 0x018

Reset Value: 0x00000008

31-4	3	2	1	0
RESERVED	TXRIS	RXRIS	RTRIS	RORRIS
r	r	r	r	r

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 TXRIS:** TX raw interrupt status.

**Bit 2 RXRIS:** RX raw interrupt status.

**Bit 1 RTRIS:** RX timeout raw interrupt status.

**Bit 0 RORRIS:** RX overrun raw interrupt status.

### 12.4.8 SSP\_MIS

Offset: 0x01C

Reset Value: 0x00000000

31-4	3	2	1	0
RESERVED	TXMIS	RXMIS	RTMIS	RORMIS
r	r	r	r	r

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 TXMIS:** TX masked interrupt status.

**Bit 2 RXMIS:** RX masked interrupt status.

**Bit 1 RTMIS:** RX timeout masked interrupt status.

**Bit 0 RORMIS:** RX overrun masked interrupt status.

### 12.4.9 SSP\_ICR

Offset: 0x020

Reset Value: 0x00000000

31-2	1	0
RESERVED	RTIC	RORIC
r	w	w

**Bits 31-2 RESERVED:** Must be kept, and cannot be modified.

**Bit 1 RTIC:** RX timeout interrupt clear. This bit is cleared by software writing 1 to it, writing 0 has no effect.

**Bit 0 RORIC:** RX overrun interrupt clear. This bit is cleared by software writing 1 to it, writing 0 has no effect.

### 12.4.10 SSP\_DMACR

Offset: 0x024

Reset Value: 0x00000000

31-2	1	0
RESERVED	TXDMAE	RXDMAE
r	r/w	r/w

**Bits 31-2 RESERVED:** Must be kept, and cannot be modified.

**Bit 1 TXDMAE:** DMA TX enable.

- 0: DMA TX disabled
- 1: DMA TX enabled

**Bit 0 RXDMAE:** DMA RX enable.

- 0: DMA RX disabled
- 1: DMA RX enabled

### 12.4.11 LORAC\_CR0

Offset: 0x100

Reset Value: 0x00000000

31-11	10	9	8	7-5	4-0
RESERVED	NSS_SEL	SCK_MOSI_SEL	RESERVED	IRQ_DIG_INT_EN	RESERVED
r	r/w	r/w	r	r/w	r

**Bits 31-11 RESERVED:** Must be kept, and cannot be modified.

**Bit 10 NSS\_SEL:** NSS source selection for RF TRX.

- 0: from register LORAC\_NSS\_CR
- 1: from internal SSP of LORAC

**Bit 9 SCK\_MOSI\_SEL:** SCK/MOSI/MISO source selection for RF TRX.

- 0: from LORAC\_SCK\_CR, LORAC\_MOSI\_CR and LORA\_MISO\_SR
- 1: from internal SSP of LORAC

**Bit 8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-5 IRQ\_DIG\_INT\_EN:** IRQ\_DIG\_INT high level interrupt enable.

Bit[5] corresponds to IRQ\_DIG[0], bit[6] corresponds to IRQ\_DIG[1] and bit[7] corresponds to IRQ\_DIG[2].

- 0: disabled
- 1: enabled

**Bits 4-0 RESERVED:** Must be kept, and cannot be modified.

### 12.4.12 LORAC\_CR1

Offset: 0x104

Reset Value: 0x00000080

31-8	7	6	5
RESERVED	POR_BAT	RESERVED	NRESET_BAT
r	r/w	r	r/w
4-3	2	1	0
RESERVED	CLK_32M_EN_BAT	TCXO_EN_BAT	PWRTCXO_EN_BAT
r	r/w	r/w	r/w

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bit 7 POR\_BAT:** POR\_BAT control.

- 0: no reset
- 1: reset

**Bit 6 RESERVED:** Must be kept, and cannot be modified.

**Bit 5 NRESET\_BAT:** NRESET\_BAT control.

- 0: reset
- 1: no reset

**Bits 4-3 RESERVED:** Must be kept, and cannot be modified.

**Bit 2 CLK\_32M\_EN\_BAT:** CLK\_32M\_EN\_BAT control.

- 0: disabled
- 1: enabled

**Bit 1 TCXO\_EN\_BAT:** TCXO\_EN\_BAT control.

- 0: disabled
- 1: enabled

**Bit 0 PWRTCXO\_EN\_BAT:** PWRTCXO\_EN\_BAT control.

- 0: disabled
- 1: enabled

#### 12.4.13 LORAC\_SR

Offset: 0x108

Reset Value: 0x00000100

31-9	8	7-5	4-2	1	0
RESERVED	BUSY_DIG_SR	IRQ_DIG_SR	RESERVED	CLK_32M_RDY_BAT_SR	RESERVED
r	r	r	r	r	r

**Bits 31-9 RESERVED:** Must be kept, and cannot be modified.

**Bit 8 BUSY\_DIG\_SR:** BUSY\_DIG status flag, it indicates whether the RF TRX is busy with processing commands. This bit is set and cleared by hardware.

- 0: RF TRX is not busy
- 1: RF TRX is busy with processing commands

**Bits 7-5 IRQ\_DIG\_SR:** IRQ\_DIG flag, it indicates the RF TRX interrupt request. This bit is set and cleared by hardware. Noted that once the interrupt request is triggered, software must send the *ClearIrqStatus* command to the RF TRX to clear the interrupt, otherwise the interrupt request will be triggered again.

- 0: no interrupt
- 1: interrupt occurred

**Bits 4-2 RESERVED:** Must be kept, and cannot be modified.

**Bit 1 CLK\_32M\_RDY\_BAT\_SR:** CLK\_32M\_RDY\_BAT status flag, it indicates whether the XO32M clock for RF TRX is ready. This bit is set and cleared by hardware.

- 0: not ready
- 1: ready

**Bit 0 RESERVED:** Must be kept, and cannot be modified.。

#### 12.4.14 LORAC\_NSS\_CR

Offset: 0x10C

Reset Value: 0x00000001

31-1	0
RESERVED	REG_NSS
r	r/w

**Bits 31-1 RESERVED:** Must be kept, and cannot be modified.

**Bit 0 REG\_NSS:** NSS control bit.

- 0: pull down NSS pin
- 1: pull up NSS pin

#### 12.4.15 LORAC\_SCK\_CR

Offset: 0x110

Reset Value: 0x00000000

31-1	0
RESERVED	REG_SCK
r	r/w

**Bits 31-1 RESERVED:** Must be kept, and cannot be modified.

**Bit 0 REG\_SCK:** SCK control bit.

- 0: pull down SCK pin
- 1: pull up SCK pin

#### 12.4.16 LORAC\_MOSI\_CR

Offset: 0x114

Reset Value: 0x00000000

31-1	0
RESERVED	REG_MOSI
r	r/w

**Bits 31-1 RESERVED:** Must be kept, and cannot be modified.

**Bit 0 REG\_MOSI:** MOSI control bit.

- 0: pull down MOSI pin
- 1: pull up MOSI pin

### 12.4.17 LORAC\_MISO\_SR

Offset: 0x118

Reset Value: 0x00000000

31-1	0
RESERVED	REG_MISO
r	r

**Bits 31-1 RESERVED:** Must be kept, and cannot be modified.

**Bit 0 REG\_MISO:** MISO status flag, it indicates the status of MISO (RF TRX output pin). This bit is set and cleared by hardware.

- 0: low
- 1: high

# 13.

# UART

## 13.1 Introduction

ASR6601 UART unit supports UART and IrDA modes.

The transmit and receive FIFOs are independent, with a depth of 16. The FIFO threshold can be configured to 1/8, 1/4, 1/2, 3/4 and 7/8. Disabling the FIFO is equivalent to a depth of 1 character.

16-bit baud rate divisor integer part and 6-bit baud rate divisor fractional part. Standard asynchronous communication bit, support 5, 6, 7 and 8-bit data, support parity check, support 1 or 2 stop bits. Support DMA, support false start bit detection, support Line Break generation and detection, support hardware flow control.

The maximum bit rate in IrDA mode is 460800, and the maximum bit rate in Low-Power IrDA mode is 115200, half-duplex. Supports 3/16 and LowPower (1.41~2.23μs) bit lengths. In Low-Power IrDA mode, the bit length is approximated by dividing the UARTCLK frequency.

Each UART port can be uniquely identified by the ID register.

## 13.2 Clock and Reset

Each UART has independent APB bus clock and independent APB bus reset.

## 13.3 Reference Clock

UARTCLK frequency must meet the requirements of baud rate generation:

$$F_{UARTCLK(min)} \geq 16 \times baudrate_{(max)}$$

$$F_{UARTCLK(max)} \leq 16 \times 65535 \times baudrate_{(min)}$$

For example, to generate baud rates from 110 bps to 460800 bps, the UARTCLK frequency must be between 7.3728 MHz to 115.34 MHz.

In the meantime, the UARTCLK frequency cannot be greater than **5 /3** times the frequency of PCLK :

$$F_{UARTCLK} \leq \frac{5}{3} * F_{PCLK}$$

For example, in UART mode, when UARTCLK is 14.7456 MHz, to generate 921600 baud, PCLK must be greater than or equal to 8.85276 MHz. This ensures that the UART has enough time to write the received data into the receive FIFO.

## 13.4 Baud Rate Generator

The baud rate generator contains free-running counters that generate the internal  $\times 16$  clocks, *Baud16* and *IrLPBaud16*. *Baud16* provides timing information for UART transmission and reception control. *Baud16* is a pulse stream with a width of one UARTCLK clock cycle and a frequency of 16 times the baud rate. *IrLPBaud16* provides timing information to generate the pulse width of the IrDA encoded transmit bit stream in low-power IrDA mode.

## 13.5 FIFO

The transmit FIFO and receive FIFO are independent, and they are enabled or disabled by the FEN bit in the UART Line Control Register ([UARTx\\_LCR\\_H](#)). The transmit FIFO is an 8-bit wide and 16 deep FIFO memory buffer. The receive FIFO is a 12-bit wide and 16 deep FIFO memory buffer, and it has four extra bits per character for status information. You can program the watermark level to  $1/8$ ,  $1/4$ ,  $1/2$ ,  $3/4$  or  $7/8$  for each FIFO through the Interrupt FIFO Level Selection Register ([UARTx\\_IFLS](#)). When FIFO is disabled, the depth is 1 byte. The FIFO status can be read from the Flag Register ([UARTx\\_FR](#)).

Bits[10:8] of the receive FIFO are error bits indicating associated errors. Bit[11] of the receive FIFO serves as an overrun indicator.

**Table 13-1 Receive FIFO Bit Functions**

FIFO Bit	Function
11	Overrun error
10	Break error
9	Parity error
8	Framing error
7:0	Received data

## 13.6 UART Operation

### 13.6.1 Baud Rate Divider

The baud rate divisor consists of a 16-bit integer and a 6-bit fractional part. The 16-bit integer is written to register [UARTx\\_IBRD](#). The 6-bit fractional part is written to register [UARTx\\_FBRD](#). The fractional baud rate divider enables the use of any clock with a frequency  $>3.6864$  MHz to act as UARTCLK, while it is still possible to generate all the standard baud rates. The Baud Rate Divisor has the following relationship to UARTCLK:

$$\text{Baud Rate Divisor} = \text{UARTCLK} / (16 \times \text{BautRate}) = \text{BRD}_I + \text{BRD}_F$$

BRD<sub>I</sub> is the integer part and BRD<sub>F</sub> is the fractional part separated by a decimal point:

16-bit Integer Part . 6-bit Fractional Part

The 6-bit number can be calculated by taking the fractional part of the required baud rate divisor and multiplying it by 64 (that is,  $2^n$ , where n is the effective width of the [UARTx\\_FBRD](#) register) and adding 0.5 to account for rounding errors:

$$\text{Fractional Part} = \text{BRD}_F \times 2^n + 0.5$$

### 13.6.2 Data Transmission

Data received or transmitted is stored in two 16-Byte FIFOs, and the receive FIFO has four extra bits per character for status information.

For transmission, data is written into the TX FIFO through the Data Register ([UARTx\\_DR](#)). Enable the UART through the UARLEN bit in the Data Register ([UARTx\\_CR](#)), then data starts transmitting with the data bit, stop bits, parity bit and other parameters indicated in the Line Control Register ([UARTx\\_LCR\\_H](#)) until the TX FIFO is empty.

Once data is written into the TX FIFO, the BUSY signal goes high and remains high while data is being transmitted.

Only when the TX FIFO is empty and the stop bits included in the last character have been transmitted from the shift register, the BUSY signal will go low. Even though the UART is no longer enabled, the BUSY signal is still high.

### 13.6.3 Data Reception

Enable the UART through the UARTEN bit in the Control Register (*UARTx\_CR*) and configure the data width, stop bits, parity bit and other parameters by the Line Control Register (*UARTx\_LCR\_H*).

When the receiver is idle, UARTRXD is pulled low, Baud16 enables the receive counter to start running, and data is sampled on the 8th cycle of that counter in UART mode or the 4th cycle of the counter in IrDA mode to allow for the shorter logic 0 pulses.

If UARTRXD remains low on the 8th cycle of Baud16, then a valid start bit is detected, otherwise a false start bit is detected and is ignored.

If the start bit is valid, then data sampling is performed every 16th cycle of Baud16 according to the length configured by the WLEN bit in register *UARTx\_LCR\_H*. If parity mode is enabled, the parity bit will be checked.

Finally, if UARTRXD is high, a valid stop bit is confirmed, otherwise a framing error is occurred. The full character received is stored in the RX FIFO along with the associated error bits.

## 13.7 IrDA SIR Operation

The IrDA SIR ENDEC provides the function of converting between an UART data stream and half-duplex serial SIR interface. The role of the SIR ENDEC is to provide a digital encoded output, and decoded input to the UART. There are two modes of operation:

- **In IrDA mode**, a zero logic level is transmitted as high pulse, and the pulse width is specified as  $3/16$  of the selected baud rate bit period on the nSIROUT signal, while logic one levels are transmitted as a LOW signal.
- **In low-power IrDA mode**, the width of the transmitted infrared pulse is set to three times the period of the internally generated IrLPBaud16 signal (1.63  $\mu$ s, assuming a nominal frequency of 1.842 MHz).

The IrDA SIR physical layer specifies a half-duplex communication link, with a minimum 10ms delay between transmission and reception. This delay must be generated by software because it is not supported by the UART. The delay is required because the infrared receiver circuits might become biased.

### 13.7.1 Low-Power Divider

The IrLPBAUD16 signal is generated by dividing the UARTCLK signal according to the low power divider value configured by the ILPDVSR bit in register *UARTx\_ILPR*.

$$\text{Low-Power Divider} = (F_{\text{UARTCLK}} / F_{\text{IrLPBAUD16}})$$

$F_{\text{IrLPBAUD16}}$  is nominally 1.8432 MHz, which meets the requirement of **1.42MHz <  $F_{\text{IrLPBAUD16}}$  < 2.12MHz**.

### 13.7.2 IrDA SIR Transmit Encoder

The SIR transmit encoder modulates the NRZ (Non Return-to-Zero) transmit bit stream output from the UART. The IrDA SIR physical layer specifies use of a RZI (Return to Zero, Inverted) modulation scheme, which represents logic 0 as an infrared light pulse. The modulated output pulse stream is transmitted to an external output driver and infrared LED.

In IrDA mode the transmitted pulse width is specified as three times the period of the internal  $\times 16$  clock (Baud16), that is,  $3/16$  of a bit period.

In low-power IrDA mode the transmit pulse width is specified as  $3/16$  of a 115200 bps bit period. This is implemented as three times the period of a nominal 1.8432 MHz clock (IrLPBaud16). In normal and low-power IrDA modes, when the fractional baud rate divider is used, the transmitted SIR pulse stream includes more jitter. This is because the Baud16 pulses cannot be generated at regular intervals when fractional division is used. That is, the Baud16 cycles have a different number of UARTCLK cycles. The worst case jitter in the SIR pulse stream can be up to three UARTCLK cycles. Provided that the UARTCLK is  $> 3.6864$  MHz and the baud rate used for IrDA mode is  $\leq 115200$  bps, the jitter is less than 9%. This is within the limits of the SIR IrDA Specification where the maximum amount of jitter permitted is 13%.

### 13.7.3 IrDA SIR Receive Decoder

The SIR receive decoder demodulates the Return-to-Zero bit stream from the infrared detector and outputs the received NRZ serial bit stream to the UART received data input. The decoder input is normally HIGH in the idle state. The transmit encoder output has the opposite polarity to the decoder input.

START bit is detected when the decoder input is LOW.

To prevent the UART from responding to glitches on the received data input, SIRIN pulses less than  $3/16$  of Baud16 will be ignored in IrDA mode; and SIRIN pulses less than  $3/16$  of IrLPBaud16 will be ignored in low-power IrDA mode.

## 13.8 UART Character Frame

The UART character frame is shown below:

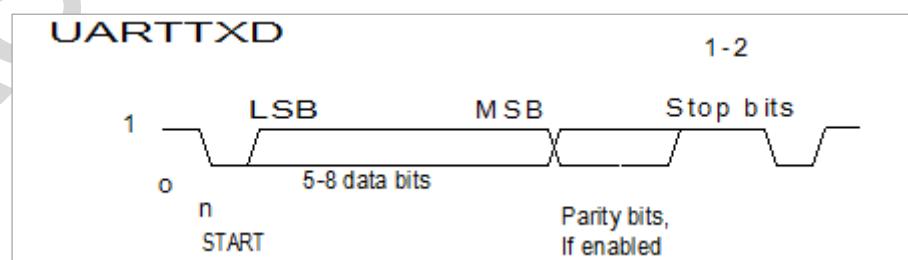


Figure 13-1 UART Character Frame

## 13.9 IrDA Data Modulation

The IrDA  $\frac{3}{16}$  data modulation is shown below.

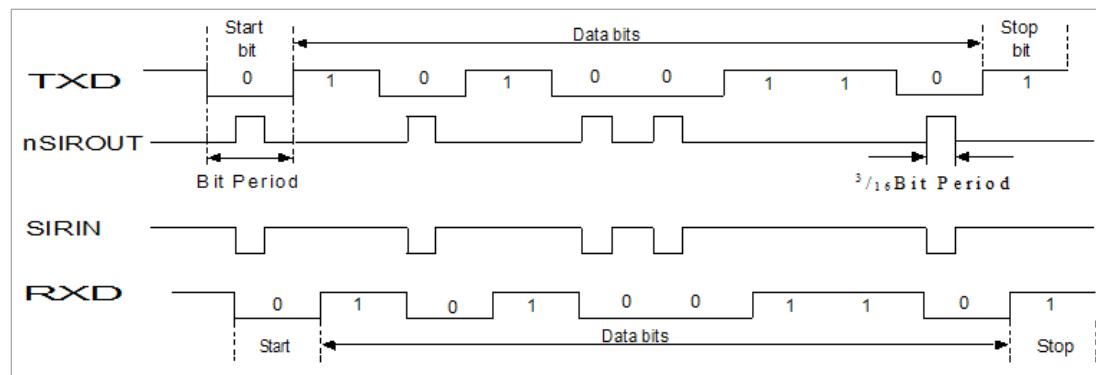


Figure 13-2 IrDA Data Modulation (3/16)

## 13.10 Hardware Flow Control

The hardware flow control is selectable using the CTSEn and RTSEn bits in the [UARTx\\_CR](#) Register.

When RTS flow control is enabled, the nUARTRTS signal is asserted until the receive FIFO is filled up to the watermark level.

When the CTS flow control is enabled, the transmitter can only transfer data when nUARTCTS signal is asserted and the transmit FIFO is not empty.

## 13.11 Interrupts

The UART supports the generation of Tx Done, Rx Done, Rx Timeout, Frame Error, Break Error, Parity Error and Overrun Error interrupts. The individual interrupts can be enabled or disabled by configuring the mask bits in the Interrupt Mask Set/Clear Register ([UARTx\\_IMSC](#)). The status of all interrupt signals, including the interrupt bits that are disabled, can be read from the Raw Interrupt Status Register ([UARTx\\_RIS](#)). The status of the enabled interrupt signals can be read from the Masked Interrupt Status Register ([UARTx\\_MIS](#)). The interrupt is cleared by writing "1" to the corresponding bit in the Interrupt Clear Register ([UARTx\\_ICR](#)).

## 13.12 DMA

The UART module supports DMA transmission and reception, which is configured by register [UARTx\\_DMACR](#).

## 13.13 UART Registers

UART0 Base Address: 0x40003000

UART1 Base Address: 0x40004000

UART2 Base Address: 0x40010000

UART3 Base Address: 0x40011000

**Table 13-2 UART Registers Summary**

Register	Offset	Description
UARTx_DR	0x000	Data Register
UARTx_RSR_ECR	0x004	Receive Status Register/Error Clear Register
UARTx_RSV0[4]	0x008	4 x 4 Bytes reserved
UARTx_FR	0x018	Flag Register
UARTx_RSV1	0x01C	4 Bytes reserved
UARTx_ILPR	0x020	IrDA Low-Power Counter Register
UARTx_IBRD	0x024	Integer Baud Rate Register
UARTx_FBRD	0x028	Fractional Baud Rate Register
UARTx_LCR_H	0x02C	Line Control Register
UARTx_CR	0x030	Control Register
UARTx_IFLS	0x034	Interrupt FIFO Level Selection Register
UARTx_IMSC	0x038	Interrupt Mask Set/Clear Register
UARTx_RIS	0x03C	Raw Interrupt Status Register
UARTx_MIS	0x040	Masked Interrupt Status Register
UARTx_ICR	0x044	Interrupt Clear Register
UARTx_DMACR	0x048	DMA Control Register
UARTx_RSV2[997]	0x04C	4 x 997 Bytes reserved

### 13.13.1 UARTx\_DR ( $x=0, 1, 2, 3$ )

Offset: 0x000

Reset Value: 0x00000000

31-12	11	10	9	8	7-0
RESERVED	OE	BE	PE	FE	DATA
r-0h	r-0h	r-0h	r-0h	r-0h	rw-0h

**Bits 31-12 RESERVED:** Must be kept, and cannot be modified.

**Bit 11 OE:** Overrun error flag.

- 0: no overrun error
- 1: overrun occurred

**Bit 10 BE:** Break error flag.

- 0: no break error
- 1: break error occurred

When this bit is set, it indicates that the received data input was held LOW for longer than a full-word (defined as start, data, parity and stop bits) transmission time.

In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO.

**Bit 9 PE:** Parity error flag.

- 0: no parity error
- 1: parity error occurred

When this bit is set, it indicates that the parity of the received data character does not match the configuration of the EPS bit in the *UARTx\_LCR\_H* Register.

In FIFO mode, this error is associated with the character at the top of the FIFO.

**Bit 8 FE:** Framing error flag.

- 0: no framing error
- 1: framing error occurred

When this bit is set, it indicates that the received character did not have a valid stop bit.

In FIFO mode, this error is associated with the character at the top of the FIFO.

**Bits 7-0 DATA:** Transmit data character/Receive data character.

### 13.13.2 **UARTx\_RSR\_ECR** (x=0, 1, 2, 3)

Offset: 0x004

Reset Value: 0x00000000

31-4	3	2	1	0
RESERVED	OE	BE	PE	FE
r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 OE:** Overrun error flag.

- 0: no overrun error
- 1: overrun occurred

**Bit 2 BE:** Break error flag.

- 0: no break error
- 1: break error occurred

When this bit is set, it indicates that the received data input was held LOW for longer than a full-word (defined as start, data, parity and stop bits) transmission time.

In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO.

**Bit 1 PE:** Parity error flag.

- 0: no parity error
- 1: parity error occurred

When this bit is set, it indicates that the parity of the received data character does not match the configuration of the EPS bit in the [UARTx\\_LCR\\_H Register](#).

In FIFO mode, this error is associated with the character at the top of the FIFO.

**Bit 0 FE:** Framing error flag.

- 0: no framing error
- 1: framing error occurred

When this bit is set, it indicates that the received character did not have a valid stop bit.

In FIFO mode, this error is associated with the character at the top of the FIFO.

### 13.13.3 UARTx\_FR (x=0, 1, 2, 3)

Offset: 0x018

Reset Value: 0x00000000

31-8	7	6	5	4	3	2-0
RESERVED	TXFE	RXFF	TXFF	RXFE	BUSY	RESERVED
r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bit 7 TXFE:** Transmit FIFO empty.

- 0: transmit FIFO/UART\_DR register not empty
- 1: transmit FIFO/UART\_DR empty

This bit is associated with the FEN bit in the *UARTx\_LCR\_H* register.

This bit does not indicate if there is data in the transmit shift register.

**Bit 6 RXFF:** Receive FIFO full.

- 0: receive FIFO/UART\_DR not full
- 1: receive FIFO/UART\_DR full

This bit is associated with the FEN bit in the *UARTx\_LCR\_H* register.

**Bit 5 TXFF:** Transmit FIFO full.

- 0: transmit FIFO/UART\_DR not full
- 1: transmit FIFO/UART\_DR full

This bit is associated with the FEN bit in the *UARTx\_LCR\_H* register.

**Bit 4 RXFE:** Receive FIFO empty.

- 0: receive FIFO/UART\_DR not empty
- 1: receive FIFO/UART\_DR empty

This bit is associated with the FEN bit in the *UARTx\_LCR\_H* register.

**Bit 3 BUSY:** UART busy.

- 0: no transfer
- 1: transfer in progress

This bit is set to 1 as soon as the transmit FIFO becomes non-empty, irrespective of whether the UART is enabled or not.

**Bits 2-0 RESERVED:** Must be kept, and cannot be modified.

#### 13.13.4 UARTx\_ILPR (x=0, 1, 2, 3)

Offset: 0x020

Reset Value: 0x00000000

31-8	7-0
RESERVED	ILPDVSR
r-0h	rw-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 ILPDVSR:** low-power divisor value. Zero is an illegal value. Writing "0" results in no generation of IrLPBaud16 pulses.

#### 13.13.5 UARTx\_IBRD (x=0, 1, 2, 3)

Offset: 0x024

Reset Value: 0x00000000

31-16	15-0
RESERVED	BAUD_DIVINT
r-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 BAUD\_DIVINT:** The integer baud rate divisor.

#### 13.13.6 UARTx\_FBRD (x=0, 1, 2, 3)

Offset: 0x028

Reset Value: 0x00000000

31-6	5-0
RESERVED	BAUD_DIVFRAC
r-0h	rw-0h

**Bits 31-6 RESERVED:** Must be kept, and cannot be modified.

**Bits 5-0 BAUD\_DIVFRAC:** The fractional baud rate divisor.

### 13.13.7 UARTx\_LCR\_H (x=0, 1, 2, 3)

Offset: 0x02C

Reset Value: 0x00000000

31-7	6-5	4	3	2	1	0
RESERVED	WLEN	FEN	STP2	EPS	PEN	BRK
r-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 31-7 RESERVED:** Must be kept, and cannot be modified.

**Bits 6-5 WLEN:** Word length.

- 00: 5 bits
- 01: 6 bits
- 10: 7 bits
- 11: 8 bits

**Bit 4 FEN:** FIFO enable.

- 0: FIFO disabled
- 1: FIFO enabled

**Bit 3 STP2:** Stop bits selection.

- 0: 1 stop bit
- 1: 2 stop bits

**Bit 2 EPS:** Parity selection.

- 0: odd parity
- 1: even parity

This bit has no effect if the PEN bit is 0.

**Bit 1 PEN:** Parity enable.

- 0: parity checking disabled
- 1: parity checking enabled.

**Bit 0 BRK:** Send break.

- write 0: end the Break command
- write 1: a low-level is continually output on the UART\_TXD pin, after completing transmission of the current character.

For the proper execution of the Break command, the software must set this bit for at least two complete frames.

### 13.13.8 UARTx\_CR (x=0, 1, 2, 3)

Offset: 0x030

Reset Value: 0x00000000

31-24		23-16	15	14	13-10
RESERVED		RESERVED	CTSEn	RTSEn	RESERVED
r-0h		r-0h	rw-0h	rw-0h	r-0h
<b>9</b>	<b>8</b>	<b>7-3</b>	<b>2</b>	<b>1</b>	<b>0</b>
RXE	TXE	RESERVED	SIRLP	SIREN	UARTEN
rw-0h	rw-0h	r-0h	rw-0h	rw-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bit 15 CTSEn:** CTS hardware flow control enable.

- 0: CTS hardware flow control disabled
- 1: CTS hardware flow control enabled

**Bit 14 RTSEn:** RTS hardware flow control enable.

- 0: RTS hardware flow control disabled
- 1: RTS hardware flow control enabled

**Bits 13-10 RESERVED:** Must be kept, and cannot be modified.

**Bit 9 RXE:** Receiver enable.

- 0: receiver disabled. If the UART is disabled during reception, then it completes the current character before stopping.
- 1: receiver enabled

**Bit 8 TXE:** Transmitter enable.

- 0: transmitter disabled. If the UART is disabled during transmission, then it completes the current character before stopping.
- 1: transmitter enabled

**Bits 7-3 RESERVED:** Must be kept, and cannot be modified.

**Bit 2 SIRLP:** Low-power IrDA SIR encoding mode selection.

- 0: low-level bits are transmitted with a pulse width of **3 /16** of the bit period.
- 1: low-level bits are transmitted with a pulse width of 3 times the period of the IrLPBaud16 input signal. Setting this bit helps reduce power consumption, but might reduce transmission distances.

**Bit 1 SIRE:** IrDA SIR enable.

- 0: IrDA SIR ENDEC is disabled. SIR\_OUT remains LOW, and signal transitions on SIR\_IN are ignored. Data is transmitted and received on UART\_TXD and UART\_RXD.
- 1: IrDA SIR ENDEC is enabled. UARTRXD remains HIGH, and signal transitions on UARTRXD are ignored. Data is transmitted and received on SIR\_OUT and SIR\_IN.

This bit has no effect if the UARTEN bit is 0.

**Bit 0 UARTEN:** UART enable.

- 0: UART disabled. If the UART is disabled during transmission or reception, then it completes the current character before stopping.
- 1: UART enabled

### 13.13.9 UARTx\_IFLS (x=0, 1, 2, 3)

Offset: 0x034

Reset Value: 0x00000000

31-6	5-3	2-0
RESERVED	RXIFLSEL	TXIFLSEL
r-0h	rw-0h	rw-0h

**Bits 31-6 RESERVED:** Must be kept, and cannot be modified.

**Bits 5-3 RXIFLSEL:** Receive interrupt FIFO level selection.

- 000: receive FIFO becomes **≥ 1/8 full**
- 001: receive FIFO becomes **≥ 1/4 full**
- 010: receive FIFO becomes **≥ 1/2 full**
- 011: receive FIFO becomes **≥ 3/4 full**
- 100: receive FIFO becomes **≥ 7/8 full**
- 101~111: reserved

**Bits 2-0 TXIFLSEL:** Transmit interrupt FIFO level selection.

- 000: transmit FIFO becomes **≥ 1/8 full**
- 001: transmit FIFO becomes **≥ 1/4 full**
- 010: transmit FIFO becomes **≥ 1/2 full**
- 011: transmit FIFO becomes **≥ 3/4 full**
- 100: transmit FIFO becomes **≥ 7/8 full**
- 101~111: reserved

### 13.13.10 UARTx\_IMSC (x=0, 1, 2, 3)

Offset: 0x038

Reset Value: 0x00000000

31-16	15-11	10	9	8	7	6	5	4	3-0
RESERVED	RESERVED	OEIM	BEIM	PEIM	FEIM	RTIM	TXIM	RXIM	RESERVED
r-0h	r-0h	rw-0h	r-0h						

**Bits 31-11 RESERVED:** Must be kept, and cannot be modified.

**Bit 10 OEIM:** Overrun error interrupt mask bit.

- 0: overrun error interrupt disabled
- 1: overrun error interrupt enabled

**Bit 9 BEIM:** Break error interrupt mask bit.

- 0: break error interrupt disabled
- 1: break error interrupt enabled

**Bit 8 PEIM:** Parity error interrupt mask bit.

- 0: parity error interrupt disabled

- 1: parity error interrupt enabled

**Bit 7 FEIM:** Framing error interrupt mask bit.

- 0: framing error interrupt disabled
- 1: framing error interrupt enabled

**Bit 6 RTIM:** Receive timeout interrupt mask bit.

- 0: receive timeout interrupt disabled
- 1: receive timeout interrupt enabled

**Bit 5 TXIM:** Transmission completion interrupt mask bit.

- 0: transmission completion interrupt disabled
- 1: transmission completion interrupt enabled

**Bit 4 RXIM:** Reception completion interrupt mask bit.

- 0: reception completion interrupt disabled
- 1: transmission completion interrupt enabled

**Bits 3-0 RESERVED:** Must be kept, and cannot be modified.

### 13.13.11 UARTx\_RIS (x=0, 1, 2, 3)

Offset: 0x03C

Reset Value: 0x00000000

31-16	15-11	10	9	8	7	6	5	4	3-0
RESERVED	RESERVED	OERIS	BERIS	PERIS	FERIS	RTRIS	TXRIS	RXRIS	RESERVED
r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 31-11 RESERVED:** Must be kept, and cannot be modified.

**Bit 10 OERIS:** Overrun error raw interrupt status.

**Bit 9 BERIS:** Break error raw interrupt status.

**Bit 8 PERIS:** Parity error raw interrupt status.

**Bit 7 FERIS:** Framing error raw interrupt status.

**Bit 6 RTRIS:** Receive timeout raw interrupt status.

**Bit 5 TXRIS:** Transmission completion raw interrupt status.

**Bit 4 RXRIS:** Reception completion raw interrupt status.

**Bits 3-0 RESERVED:** Must be kept, and cannot be modified.

### 13.13.12 UARTx\_MIS (x=0, 1, 2, 3)

Offset: 0x040

Reset Value: 0x00000000

31-16	15-11	10	9	8	7	6	5	4	3-0
RESERVED	RESERVED	OEMIS	BEMIS	PEMIS	FEMIS	RTMIS	TXMIS	RXMIS	RESERVED
r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 31-11 RESERVED:** Must be kept, and cannot be modified.

**Bit 10 OEMIS:** Overrun error masked interrupt status.

**Bit 9 BEMIS:** Break error masked interrupt status.

**Bit 8 PEMIS:** Parity error masked interrupt status.

**Bit 7 FEMIS:** Framing error masked interrupt status.

**Bit 6 RTMIS:** Receive timeout masked interrupt status.

**Bit 5 TXMIS:** Transmission completion masked interrupt status.

**Bit 4 RXMIS:** Reception completion masked interrupt status.

**Bits 3-0 RESERVED:** Must be kept, and cannot be modified.

### 13.13.13 UARTx\_ICR (x=0, 1, 2, 3)

Offset: 0x044

Reset Value: 0x00000000

31-16	15-11	10	9	8	7	6	5	4	3-0
RESERVED	RESERVED	OEIC	BEIC	PEIC	FEIC	RTIC	TXIC	RXIC	RESERVED
r-0h	r-0h	w-0h	r-0h						

**Bits 31-11 RESERVED:** Must be kept, and cannot be modified.

**Bit 10 OEIC:** Clear Overrun Error interrupt.

- 0: no action
- 1: clear overrun error interrupt

**Bit 9 BEIC:** Clear Break Error interrupt.

- 0: no action
- 1: clear break error interrupt

**Bit 8 PEIC:** Clear Parity Error interrupt.

- 0: no action
- 1: clear parity error interrupt

**Bit 7 FEIC:** Clear Framing Error interrupt.

- 0: no action
- 1: clear framing error interrupt

**Bit 6 RTIC:** Clear Receive Timeout interrupt.

- 0: no action
- 1: clear receive timeout interrupt

**Bit 5 TXIC:** Clear Transmission Completion interrupt.

- 0: no action
- 1: clear transmission completion interrupt

**Bit 4 RXIC:** Clear Reception Completion interrupt.

- 0: no action
- 1: clear reception completion interrupt

**Bits 3-0 RESERVED:** Must be kept, and cannot be modified.

### 13.13.14 UARTx\_DMACR (x=0, 1, 2, 3)

Offset: 0x048

Reset Value: 0x00000000

31-3	2	1	0
RESERVED	DMAONERR	TXDMAE	RXDMAE
r-0h	rw-0h	rw-0h	rw-0h

**Bits 31-3 RESERVED:** Must be kept, and cannot be modified.

**Bit 2 DMAONERR:** DMA on error.

**Bit 1 TXDMAE:** Transmit DMA enable.

- 0: disabled
- 1: enabled

**Bit 0 RXDMAE:** Receive DMA enable.

- 0: disabled
- 1: enabled

### 13.13.15 UARTx\_ID[8] (x=0, 1, 2, 3)

#### 13.13.15.1 PeriphID0

Offset: 0x0FE0

Reset Value: 0x00000000

31-8	7-0
RESERVED	PARTNUMBER0
r-0h	r-11h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 PARTNUMBER0:** 0x11.

#### 13.13.15.2 PeriphID1

Offset: 0x0FE4

Reset Value: 0x00000000

31-8	7-4	3-0
RESERVED	DESIGNER0	PARTNUMBER1
r-0h	r-1h	r-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-4 DESIGNER0:** 0x1.

**Bits 3-0 PARTNUMBER1:** 0x0.

#### 13.13.15.3 PeriphID2

Offset: 0x0FE8

Reset Value: 0x00000000

31-8	7-4	3-0
RESERVED	REVISION0	DESIGNER1
r-0h	r-xh	r-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-4 REVISION0:**

- 0x0: r1p0
- 0x1: r1p1
- 0x2: r1p3/r1p4
- 0x3: r1p5

**Bits 3-0 DESIGNER1:** 0x0.

#### 13.13.15.4 PeriphID3

Offset: 0x0FEC

Reset Value: 0x00000000

31-8	7-0
RESERVED	CONFIGURATION
r-0h	r-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 CONFIGURATION:** 0x00.

#### 13.13.15.5 PCellID0

Offset: 0x0FD0

Reset Value: 0x00000000

31-8	7-0
RESERVED	CellID0
r-0h	r-dh

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 CellID0:** 0x0d.

#### 13.13.15.6 PCellID1

Offset: 0x0FD4

Reset Value: 0x00000000

31-8	7-0
RESERVED	CellID1
r-0h	r-f0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 CellID1:** 0xf0.

### 13.13.15.7 PCellID2

Offset: 0x0FD8

Reset Value: 0x00000000

31-8	7-0
RESERVED	CellID2
r-0h	r-5h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 CellID2:** 0x05.

### 13.13.15.8 PCellID3

Offset: 0x0FDC

Reset Value: 0x00000000

31-8	7-0
RESERVED	CellID3
r-0h	r-b1h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 CellID3:** 0xb1.

# 14.

# SSP

## 14.1 Introduction

All three SSP (synchronous serial port) can be configured as a master or slave device.

SSP support multiple frame formats, with configurable data width and transfer rate.

## 14.2 Main Features

- Master or slave operation
- Up to 16 MHz output
- 16-bit wide TX/RX FIFO with a depth of 8
- Multiple frame formats
- 4-bit to 16-bit data width
- DMA
- Interrupt signal generation

## 14.3 Functional Description

### 14.3.1 Basic Information

Four I/O pins (SSP\_NSS, SSP\_CLK, SSP\_TX, SSP\_RX) are dedicated to SSP communication with external devices.

#### 1. **SSP\_NSS**

The chip select pin is active at low level.

#### 2. **SSP\_CLK**

SSP clock pin acts as clock output in master mode and as the clock input in slave mode.

#### 3. **SSP\_TX**

SSP TX pin is used to transmit data in both master and slave modes.

#### 4. **SSP\_RX**

SSP RX pin is used to receive data in both master and slave modes.

The connection between SSP and SPI device is shown in the figure below. Note the difference between SSP\_TX/SSP\_RX and SPI\_MOSI/SPI\_MISO.

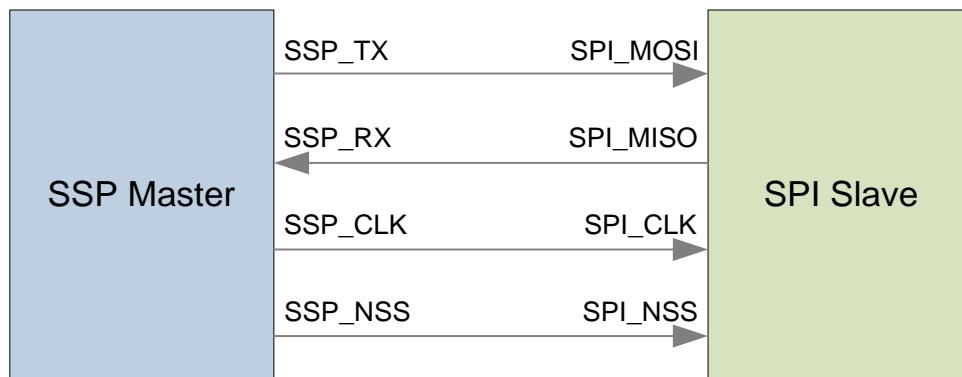


Figure 14-1 Connection between a SSP Master and a SPI Slave

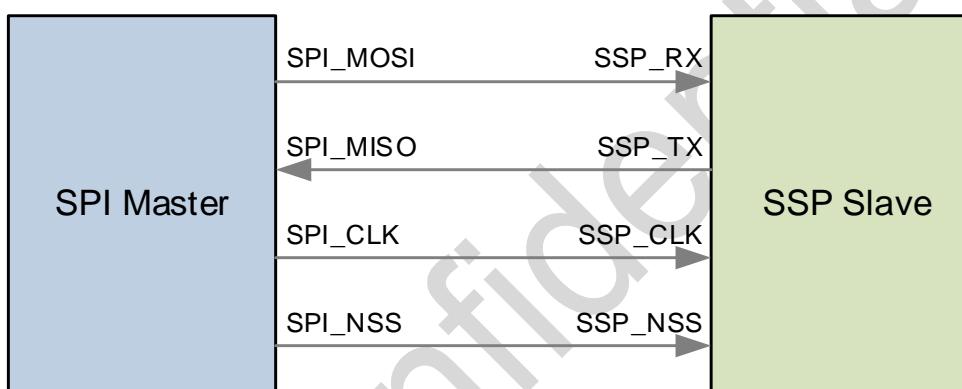


Figure 14-2 Connection between a SPI Master and a SSP Slave

### 14.3.2 Clock Division

SSP clock should meet below requirements:

- (1) Up to 16 MHz output clock
  - (2) The clock frequency in master mode is at most 1/2 of PCLK clock frequency
  - (3) The clock frequency in slave mode is at most 1/12 of PCLK clock frequency
- The formula to calculate clock output in master mode is below:

$$F_{SSPCLKOUT} = \frac{F_{SSPCLK}}{CPSDVR \times (1+SCR)}$$

Figure 14-3 MASTER mode clock output calculation

SSPCLK is the interface clock for SSP, and SSPCLKOUT is the output clock. For example, SSPCLK is 24MHz by default and 1MHz SSPCLKOUT is required, then the user should set bit CPSDVR in register [SSP\\_CPSR](#) to 2, and set bit SCR in register [SSP\\_CR0](#) to 11.

### 14.3.3 Data Format

SSP supports three frame formats:

- Motorola SPI
- Texas Instruments SPI
- National Semiconductor Microwire

### 14.3.4 DMA Transfer

#### SSP DMA Transfer Process:

- (1) Enable the TXDMAE bit in register *SSP\_DMACR*.
- (2) Configure *SSP\_DR* register as the destination address of DMA.
- (3) Configure the memory address of the data to be sent as the source address of DMA.
- (4) Configure the data width of DMA transfer to 8 bits by configuring the SRC\_TR\_WIDTH and DES\_TR\_WIDTH bits to 0 in the *DMA\_CTLx* register.
- (5) Configure the DMA burst length to 4 by configuring the SRC\_MSIZE and DEST\_MSIZE bits to 1 in the *DMA\_CTLx* register.
- (6) Configure the total length of DMA data transfer.
- (7) Configure DMA handshake type to the corresponding SSP TX type (for example, for SSP0, configure it to DMA\_HANDSHAKE\_SSP\_0\_TX).
- (8) Activate the DMA.

When the DMA transfer is completed, the CH\_EN\_x bit in the DMA\_CHENREG register is cleared.

#### SSP DMA Reception Process:

- (1) Enable the RXDMAE bit in register *SSP\_DMACR*.
- (2) Configure register *SSP\_DR* as the source address of DMA.
- (3) Configure the memory address of the data to be received as the destination address of DMA.
- (4) Configure the data width of DMA transfer to 8 bits by configuring the SRC\_TR\_WIDTH and DES\_TR\_WIDTH bits to 0 in the *DMA\_CTLx* register.
- (5) Configure the DMA burst length to 4 by configuring the SRC\_MSIZE and DEST\_MSIZE bits to 1 in the *DMA\_CTLx* register.
- (6) Configure the total length of DMA data transfer.
- (7) Configure DMA handshake type to the corresponding SSP RX type (for example, for SSP0, configure it to DMA\_HANDSHAKE\_SSP\_0\_RX).
- (8) Activate the DMA.

When the DMA transfer is completed, the CH\_EN\_x bit in the DMA\_CHENREG register is cleared.

### 14.3.5 Interrupts

SSP has four interrupts: SSP RX interrupt, SSP TX interrupt, SSP RX OVERRUN interrupt and SSP RX TIMEOUT.

#### 1. SSP RX Interrupt

SSP RX interrupt is triggered when there are 4 or more locations in SSP RX FIFO.

#### 2. SSP TX Interrupt

SSP TX interrupt is triggered when there are 4 or less locations in SSP TX FIFO.

#### 3. SSP RX Overrun Interrupt

SSP RX overrun interrupt is triggered when the SSP RX FIFO is full and continues to receive data.

#### 4. SSP RX Timeout Interrupt

SSP RX timeout interrupt is triggered when the SSP RX FIFO is not empty but SSP has not received any new data for the duration time of 32-bit data transfer.

## 14.4 SSP Registers

SSP0 Base Address: 0x40006000

SSP1 Base Address: 0x40012000

SSP2 Base Address: 0x40013000

Table 14-1 SSP Registers Summary

Register	Offset	Description
SSP_CR0	0x00	Control register 0
SSP_CR1	0x04	Control register 1
SSP_DR	0x08	Data register
SSP_SR	0x0C	Status register
SSP_CPSR	0x10	Clock Prescaler Register
SSP_IMSC	0x14	Interrupt Mask Set/Clear Register
SSP_RIS	0x18	Raw Interrupt Status register
SSP_MIS	0x1C	Masked Interrupt Status register
SSP_ICR	0x20	Interrupt Clear Register
SSP_DMACR	0x24	DMA Control Register

### 14.4.1 SSP\_CR0

Offset: 0x000

Reset Value: 0x00000000

31-16	15-8	7	6	5-4	3-0
RESERVED	SCR	SPH	SPO	FRF	DSS
r	r/w	r/w	r/w	r/w	r/w

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-8 SCR:** Serial clock rate, used to set the SSP transfer rate.

$$F_{SSPCLKOUT} = \frac{F_{SSPCLK}}{CPSDVR \times (1+SCR)}$$

The formula to calculate the SSP transfer rate is as above, where CPSDVR is an even number ranging from 2 to 254.

**Bit 7 SPH:** SSP phase setting, only applied in Motorola SPI format.

**Bit 6 SPO:** SSP polarity setting, only applied in Motorola SPI format.

**Bits 5-4 FRF:** SSP frame formats setting.

- 0: Motorola SPI
- 1: Texas Instruments SPI
- 2: National Semiconductor Microwire
- 3: reserved

**Bits 3-0 DSS:** Data width setting.

- 0: reserved
- 1: reserved
- 2: reserved
- 3: 4 bits
- 4: 5 bits
- 5: 6 bits
- 6: 7 bits
- 7: 8 bits
- 8: 9 bits
- 9: 10 bits
- 10: 11 bits
- 11: 12 bits
- 12: 13 bits
- 13: 14 bits
- 14: 15 bits
- 15: 16 bits

### 14.4.2 SSP\_CR1

Offset: 0x004

Reset Value: 0x00000000

31-4	3	2	1	0
RESERVED	SOD	MS	SSE	LBM
r	r/w	r/w	r/w	r/w

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 SOD:** SSP output disable in slave mode.

- 0: SSP output enabled in slave mode
- 1: SSP output disabled in slave mode

**Bit 2 MS:** Master/slave mode selection.

- 0: master mode
- 1: slave mode

**Bit 1 SSE:** SSP enable.

- 0: disabled
- 1: enabled

**Bit 0 LBM:** loopback mode.

- 0: normal mode
- 1: loopback mode

### 14.4.3 SSP\_DR

Offset: 0x008

Reset Value: 0x00000000

31-16	15-0
RESERVED	DATA
r	r/w

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 DATA:** SSP TX/RX data.

#### 14.4.4 SSP\_SR

Offset: 0x00C

Reset Value: 0x00000003

31-5	4	3	2	1	0
RESERVED	BSY	RFF	RNE	TNF	TFE
r	r	r	r	r	r

**Bits 31-5 RESERVED:** Must be kept, and cannot be modified.

**Bit 4 BSY:** SSP busy flag.

- 0: SSP is idle
- 1: SSP transfer in progress

**Bit 3 RFF:** RX FIFO full flag.

- 0: RX FIFO is not full
- 1: RX FIFO is full

**Bit 2 RNE:** RX FIFO not empty flag.

- 0: RX FIFO is empty
- 1: RX FIFO is not empty

**Bit 1 TNF:** TX FIFO not full flag.

- 0: TX FIFO is full
- 1: TX FIFO is not full

**Bit 0 TFE:** TX FIFO empty flag.

- 0: TX FIFO is not empty
- 1: TX FIFO is empty

#### 14.4.5 SSP\_CPSR

Offset: 0x010

Reset Value: 0x00000000

31-8	7-0
RESERVED	CPSDVS
r	r/w

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 CPSDVS:** Clock prescaler divider, must be an even number between 2~254.

#### 14.4.6 SSP\_IMSC

Offset: 0x014

Reset Value: 0x00000000

31-4	3	2	1	0
RESERVED	TXIM	RXIM	RTIM	RORIM
r	r/w	r/w	r/w	r/w

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 TXIM:** TX interrupt mask bit.

- 0: TX interrupt is masked
- 1: TX interrupt is not masked

**Bit 2 RXIM:** RX interrupt mask bit.

- 0: RX interrupt is masked
- 1: RX interrupt is not masked

**Bit 1 RTIM:** RX timeout interrupt mask bit.

- 0: RX timeout interrupt is masked
- 1: RX timeout interrupt is not masked

**Bit 0 RORIM:** RX overrun interrupt mask bit.

- 0: RX overrun interrupt is masked
- 1: RX overrun interrupt is not masked

#### 14.4.7 SSP\_RIS

Offset: 0x018

Reset Value: 0x00000008

31-4	3	2	1	0
RESERVED	TXRIS	RXRIS	RTRIS	RORRIS
r	r	r	r	r

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 TXRIS:** TX raw interrupt status.

**Bit 2 RXRIS:** RX raw interrupt status.

**Bit 1 RTRIS:** RX timeout raw interrupt status.

**Bit 0 RORRIS:** RX overrun raw interrupt status.

#### 14.4.8 SSP\_MIS

Offset: 0x01C

Reset Value: 0x00000000

31-4	3	2	1	0
RESERVED	TXMIS	RXMIS	RTMIS	RORMIS
r	r	r	r	r

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 TXMIS:** TX masked interrupt status.

**Bit 2 RXMIS:** RX masked interrupt status.

**Bit 1 RTMIS:** RX timeout masked interrupt status.

**Bit 0 RORMIS:** RX overrun masked interrupt status.

#### 14.4.9 SSP\_ICR

Offset: 0x020

Reset Value: 0x00000000

31-2	1	0
RESERVED	RTIC	RORIC
r	w	w

**Bits 31-2 RESERVED:** Must be kept, and cannot be modified.

**Bit 1 RTIC:** RX timeout interrupt clear. This bit is cleared by software writing 1 to it, writing 0 has no effect.

**Bit 0 RORIC:** RX overrun interrupt clear. This bit is cleared by software writing 1 to it, writing 0 has no effect.

#### 14.4.10 SSP\_DMACR

Offset: 0x024

Reset Value: 0x00000000

31-2	1	0
RESERVED	TXDMAE	RXDMAE
r	r/w	r/w

**Bits 31-2 RESERVED:** Must be kept, and cannot be modified.

**Bit 1 TXDMAE:** DMA TX enable.

- 0: DMA TX disabled
- 1: DMA TX enabled

**Bit 0 RXDMAE:** DMA RX enable.

- 0: DMA RX disabled
- 1: DMA RX enabled

# 15.

# I2C

## 15.1 Introduction

The I2C bus interface unit supports master mode and slave mode. SDA is the data transmission line, and SCL is the interface clock line. It supports multi-host and bus arbitration functions. It supports 100Kbps standard rate mode and 400Kbps fast mode. It supports FIFO mode, the transmit FIFO depth is 8, the receive FIFO depth is 16, and the read and write pointers of FIFO are configurable.

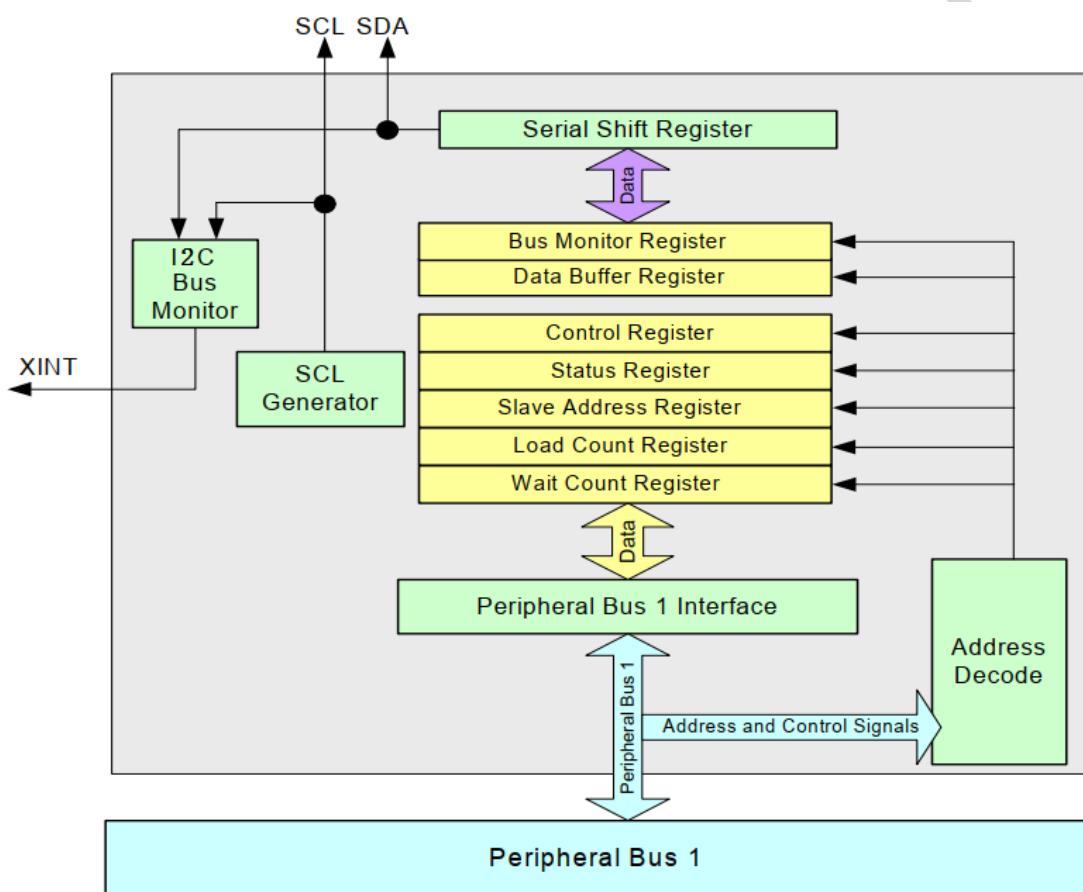


Figure 15-1 I2C Block Diagram

## 15.2 Start and Stop Conditions

**Start condition:** When SCL level is high, SDA level changes from high to low, thus generating a Start condition.

**Stop condition:** When SCL level is high, SDA level changes from low to high, thus generating a Stop condition.

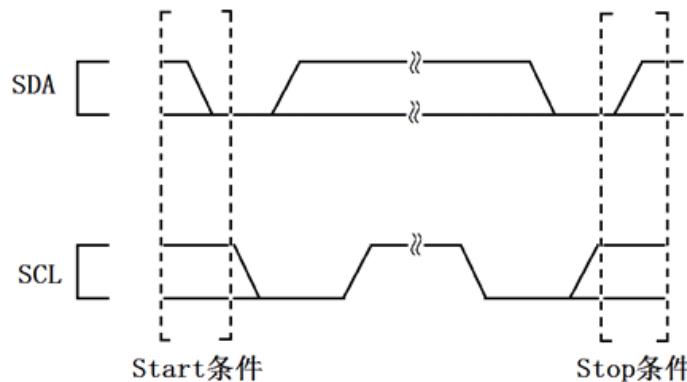


Figure 15-2 SDA and SCL Signals During Start and Stop Conditions

Start a byte transmission or generate Start, Repeated Start or Stop conditions by configuring the START and STOP bits in register *I2Cx\_CR*.

Table 15-1 Start and Stop Conditions

Start bit	Stop bit	Condition Type	Description
0	0	No Start or Stop	When multiple data bytes are to be transmitted, I2C will not send a Start or Stop condition.
0	1	Start or Repeated Start	I2C sends a Start condition and then transmits the 8-bit data in the <i>I2Cx_DB</i> register. Before a Start condition is sent, register <i>I2Cx_DB</i> must contain the 7-bit slave address and the R/nW bit. For a Repeated Start, the <i>I2Cx_DB</i> register must contain the target slave address and the R/nW bit, which allows a master to perform multiple transfers without freeing the bus. The interface stays in master transmit mode for writes, and switches to master receive mode for reads.
1	x	Stop	In master transmit mode, a Stop condition is sent on the I2C bus after the 8-bit data in the <i>I2Cx_DB</i> register has been transferred. In master receive mode, bit ACKNAK in the <i>I2Cx_CR</i> register must be set to send a NAK. The received data is stored in the <i>I2Cx_DB</i> register, and then a Stop condition is sent on the I2C bus.

### 1. Start Condition

The Start condition and the data in register *I2Cx\_DB* will be sent after bit TB in register *I2Cx\_CR* is set. I2C bus stays in master transmit mode for write requests and stays in master receive mode for read requests.

For a Repeated Start, a change in Read, Write or the target slave address, the register *I2Cx\_DB*R contains the updated target slave address and the R/nW bit.

The START condition will not be cleared by the I2C. If I2C loses bus arbitration when it starts to send a Start condition, it will try to resend the Start condition when the bus is freed.

## 2. No Start or Stop Condition

When I2C is transmitting multiple data bytes, the START and STOP bits in the *I2Cx\_CR* register are set to 0, there is no Start or Stop condition. Software writes the data byte, and I2C sets bit ITE in register *I2Cx\_SR* and clears bit TB in register *I2Cx\_CR*. Then software writes a new byte to register *I2Cx\_DB*R and sets bit TB in register *I2Cx\_CR*, which starts the new byte transmission. This process continues until the START or STOP bit in register *I2Cx\_CR* is set by software. After a Start, Stop or Repeated Start condition is transmitted, the START and STOP bits in register *I2Cx\_CR* are not cleared automatically by I2C.

After each byte and ACK/NAK are transferred, I2C holds the SCL line low and waits until the TB bit in register *I2Cx\_CR* is set.

## 3. Stop Condition

A Stop condition terminates a data transfer. In master transmit mode, the STOP and TB bits in register *I2Cx\_CR* must be set to start the transmission of the last byte. In master receive mode, the ACKNAK, STOP and TB bits in register *I2Cx\_CR* must be set to start the reception of the last byte. After a Stop condition is transmitted, software must clear the STOP bit in register *I2Cx\_CR*.

## 15.3 Data Transfer Sequence

I2C transmits data in 1-byte increments and by following sequence:

1. Start
2. 7-bit slave address
3. R/nW bit
4. Acknowledge pulse
5. 8 data bits
6. Acknowledge pulse
7. Repeat of Step 5 and Step 6
8. Repeated Start (repeat Step 1) or Stop

## 15.4 Data and Addressing

The I2C Data Buffer Register (*I2Cx\_DBR*) and the I2C Slave Address Register (*I2Cx\_SAR*) manage data and slave addressing. *I2Cx\_DBR* contains 1 byte of data or a 7-bit target slave address and the R/nW bit. *I2Cx\_SAR* contains the ASR6601 I2C slave address when the I2C is in slave mode. After I2C receives a full byte of data and an ACK, it stores the data in register *I2Cx\_DBR*. To transmit data, the CPU writes to the *I2Cx\_DBR* register, and the I2C transmits the data to the serial bus when the TB bit in register *I2Cx\_CR* is set.

### 1. I2C is in Master/Slave Transmit mode:

- (1) Software writes data to the *I2Cx\_DBR* register, which makes the I2C to start a master transaction or to send the next data byte after the ITE bit in register *I2Cx\_SR* is set.
- (2) When bit TB in register *I2Cx\_CR* is set, the data in register *I2Cx\_DBR* is transmitted.
- (3) If the ITEIE bit in register *I2Cx\_CR* is set, an *I2Cx\_DBR* transmit-empty interrupt is triggered after a byte and an ACK is transferred.
- (4) When the I2C is ready to send the next byte before the CPU writes to the *I2Cx\_DBR* register and there is no Stop condition, the I2C is in a wait state until the CPU writes to the *I2Cx\_DBR* register and sets the TB bit in the *I2Cx\_CR* register.

**Note:** In FIFO mode, software writes to the TX FIFO instead of the *I2Cx\_DBR* register.

### 2. I2C is in Master/Slave Receive mode:

- (1) When a full byte of data is received (if the DRFIE bit in register *I2Cx\_CR* is set), the *I2Cx\_DBR* receive-full interrupt is generated and the IRF bit in register *I2Cx\_SR* is set, the CPU then reads the *I2Cx\_DBR* register to retrieve the data.
- (2) After the ACK cycle is completed, I2C transfers data from the shift register to the *I2Cx\_DBR* register.
- (3) I2C is in wait state until the *I2Cx\_DBR* register is read by the CPU.
- (4) After the CPU reads the *I2Cx\_DBR* register, the I2C updates the ACKNAK and TB bits in register *I2Cx\_CR* to allow the transmission of the next byte.

**Note:** In FIFO mode, software reads from the RX FIFO instead of the *I2Cx\_DBR* register.

### 3. Addressing a Slave Device:

As a master device, the I2C must form and send the first byte of a transaction. This byte consists of a 7-bit slave address and a R/nW bit. After the first byte is transmitted, the I2C must receive an ACK from the slave device. When it is a Write transaction, the I2C remains in master transmit mode, and the slave device remains in slave receive mode. When it is a Read transaction, the I2C switches to master receive mode immediately after receiving an ACK, and the slave device switches to slave transmit mode. When a NAK is returned, the I2C automatically sends a Stop condition and sets the BED bit in register *I2Cx\_SR* to abort the current transaction.

## 15.5 Acknowledge

Each byte transmission must be accompanied by an ACK generated by the master or slave receiver. The transmitter must release the SDA line for the receiver to transmit the ACK.

In master transmit mode, if the target slave receiver does not generate an ACK, the SDA line remains high, which indicates a NAK. The lack of an ACK causes I2C to set the BED bit in register *I2Cx\_SR* and generate an interrupt. I2C automatically generates a Stop condition and aborts the transmission.

In master receive mode, I2C sends a NAK to notify the slave transmitter to stop sending data. The ACKNAK bit in the *I2Cx\_CR* register controls the generation of ACK/NAK on the bus. According to the I2C protocol, the BED bit in the *I2Cx\_SR* register is not set for a master receive mode NAK. I2C automatically sends the ACK every time it receives a byte from the bus. Before the master receiver receives the last byte, software must set the ACKNAK bit in the *I2Cx\_CR* register to generate a NAK. The NAK is sent after the last byte to indicate that the last byte has been sent.

In slave receive mode, I2C automatically acknowledges its own slave address, irrespective of whether the ACKNAK bit in the *I2Cx\_CR* register is set. In slave mode, I2C automatically sends the ACK after receiving a byte, irrespective of whether the ACKNAK bit in the *I2Cx\_CR* register is set.

In slave transmit mode, receiving a NAK indicates that the last byte has been transferred. The master then sends a Stop condition or Repeated Start condition. The UB bit in register *I2Cx\_SR* remains set until a Stop condition or Repeated Start condition is received.

## 15.6 Arbitration

I2C bus arbitration is required by a multi-master capability. Bus arbitration is used when two or more masters simultaneously generate a Start condition within the minimum time of a Start condition.

Arbitration can last for a long time. If the slave address and the R/nW bit are the same, the arbitration moves to the data. Due to the Wired-And nature of the I2C bus, no data is lost if two or all masters are outputting the same bus state. If the address, the R/nW bit, or the data are different, the master that transitioned to the high state (the master data is different from the SDA line) loses arbitration and aborts the data transfer. The I2C bus sets bit ALD in register *I2Cx\_SR*, then returns to the idle state.

In FIFO mode, software must empty the FIFOs once arbitration is lost. This can be done by clearing the read and write pointer registers for TxFIFO and RxFIFO.

## 15.7 I2C Master Mode

When software starts a read or write operation on the I2C bus, the I2C switches from the default slave receive mode to master transmit mode. The Start condition is followed by the 7-bit slave address and the R/nW bit.

After receiving an ACK, the I2C enters one of the two master modes:

- Master transmit: I2C writes data
- Master receive: I2C reads data

The CPU writes to register *I2Cx\_CR* to start a master transaction.

**Table 15-2 Master Transactions**

I2C Master Action	Op. mode	Definition
Generate clock output	Master transmit Master receive	<ul style="list-style-type: none"> <li>● The master drives the SCL line.</li> <li>● The SCLE and UE bits in the <i>I2Cx_CR</i> register must be set.</li> </ul>
Write slave address to <i>I2Cx_DB</i>	Master transmit Master receive	<ul style="list-style-type: none"> <li>● CPU writes to bits[7:1] in the <i>I2Cx_DB</i> register before a Start condition is enabled.</li> <li>● The first 7 bits are sent after Start.</li> </ul>
Write R/nW bit to <i>I2Cx_DB</i>	Master transmit Master receive	<ul style="list-style-type: none"> <li>● CPU writes the R/nW control bit to the least significant bit in register <i>I2Cx_DB</i>.</li> <li>● If the R/nW bit is low, master remains a master transmitter, if the R/nW bit is high, master switches to a master receiver</li> </ul>
Send Start condition	Master transmit Master receive	<p>After the 7-bit target slave address and the R/nW bit are written into the <i>I2Cx_DB</i> register,</p> <ul style="list-style-type: none"> <li>● Software sets the START bit in register <i>I2Cx_CR</i>.</li> <li>● Software sets the TB bit in register <i>I2Cx_CR</i> to initiate the Start condition.</li> </ul>
Initiate first data byte transmission	Master transmit Master receive	<ul style="list-style-type: none"> <li>● CPU writes one data byte to the <i>I2Cx_DB</i> register.</li> <li>● Software sets the TB bit in register <i>I2Cx_CR</i> and I2C starts the transmission of the Byte.</li> <li>● The TB bit in register <i>I2Cx_CR</i> is cleared and the ITE bit in register <i>I2Cx_SR</i> is set when the transfer is complete.</li> </ul>
Arbitrate for I2C bus	Master transmit Master receive	<p>If 2 or more masters send a Start condition within the same clock period, then bus arbitration must occur.</p> <ul style="list-style-type: none"> <li>● I2C arbitrates as long as there is a need. Bus arbitration takes place during the transmission of target slave address, R/nW bit or data, and it continues until all masters except one master lose the bus. No data is lost.</li> <li>● If I2C loses arbitration, the ALD bit in register <i>I2Cx_SR</i> is set, and I2C switches to slave receive mode.</li> <li>● If I2C loses arbitration when it starts to send the target slave address, it will try to resend the address when the bus is freed.</li> </ul>
Write one data byte to <i>I2Cx_DB</i>	Master transmit only	<ul style="list-style-type: none"> <li>● When the ITE bit in the <i>I2Cx_SR</i> register is set and the TB bit in the <i>I2Cx_CR</i> register is cleared, if enabled, the <i>I2Cx_DB</i> transmit-empty interrupt is generated.</li> <li>● The CPU writes a data byte to the <i>I2Cx_DB</i> register, sets the appropriate Start/Stop condition combination as required,</li> </ul>

		<p>and sets the TB bit in register <i>I2Cx_CR</i> to send data. The 8 bits of data are transferred from the shift register to the serial bus. If the STOP bit in register <i>I2Cx_CR</i> is set before the transfer, then the 8 bits of data is followed by a Stop condition.</p>
Wait for ACK from slave receiver	Master transmit only	<p>As a master transmitter, the I2C generates the ACK clock, and releases the SDA line for the slave receiver to transmit the ACK.</p>
Read one byte from <i>I2Cx_DB</i> R	Master receive only	<p>After the ACKNAK bit in register <i>I2Cx_CR</i> is read, the 8 bits of data in the shift register is transferred to the <i>I2Cx_DB</i>R register:</p> <ul style="list-style-type: none"> <li>The CPU reads the <i>I2Cx_DB</i>R register when the IRF bit in register <i>I2Cx_SR</i> is set and the TB bit in register <i>I2Cx_CR</i> is cleared. If the <i>I2Cx_DB</i>R receive-full interrupt is enabled, it is signalled to the CPU.</li> <li>When the <i>I2Cx_DB</i>R register is read, if the ACKNAK bit in register <i>I2Cx_SR</i> is cleared (indicating ACK), the software must clear the ACKNAK bit and set the TB bit in register <i>I2Cx_CR</i> to start the next byte Read.</li> <li>If the ACKNAK bit in <i>I2Cx_SR</i> is set (indicating NAK), the TB bit in <i>I2Cx_CR</i> is cleared, the STOP bit in <i>I2Cx_CR</i> is set, and the UB bit in <i>I2Cx_SR</i> is set, then the last byte has been read from the <i>I2Cx_DB</i>R register, and the I2C is sending the Stop.</li> <li>If the ACKNAK bit in <i>I2Cx_SR</i> is set (indicating NAK) and the TB bit in <i>I2Cx_CR</i> is cleared, but the STOP bit in <i>I2Cx_CR</i> is cleared, then the software has two options:             <ol style="list-style-type: none"> <li>Set the START bit in <i>I2Cx_CR</i>, write a new target slave address to the <i>I2Cx_DB</i>R register, set the TB bit in <i>I2Cx_CR</i>, and send a Repeated Start.</li> <li>Set the MA bit in <i>I2Cx_CR</i> and keep the TB bit as 0 in <i>I2Cx_CR</i>, and only send a Stop.</li> </ol> </li> </ul>
Transmit ACK to slave transmitter	Master receive only	<ul style="list-style-type: none"> <li>As a master receiver, the I2C generates the ACK clock and drives the SDA line during the ACK cycle.</li> <li>If the next data byte is the last transaction, the user software sets the ACKNAK bit in register <i>I2Cx_CR</i> to generate NAK.</li> </ul>
Generate a Repeated Start	Master transmit Master receive	<p>Use a Repeated Start condition instead of a Stop condition to initiate a new transaction without releasing the bus.</p> <ul style="list-style-type: none"> <li>The Repeated Start is generated after the last data byte has been transmitted.</li> <li>Software must write the next 7-bit target slave address and the R/nW bit to the <i>I2Cx_DB</i>R register, set the START bit in register <i>I2Cx_CR</i>, and set the TB bit in register <i>I2Cx_CR</i>.</li> </ul>
Generate Stop	Master transmit Master receive	<ul style="list-style-type: none"> <li>A Stop is generated after the last data byte has been transmitted.</li> <li>The STOP bit in register <i>I2Cx_CR</i> must be set before the transmission of the last byte.</li> </ul>

## 15.8 FIFO Mode

The FIFO mode can only be used when the I2C is in *Master Mode*.

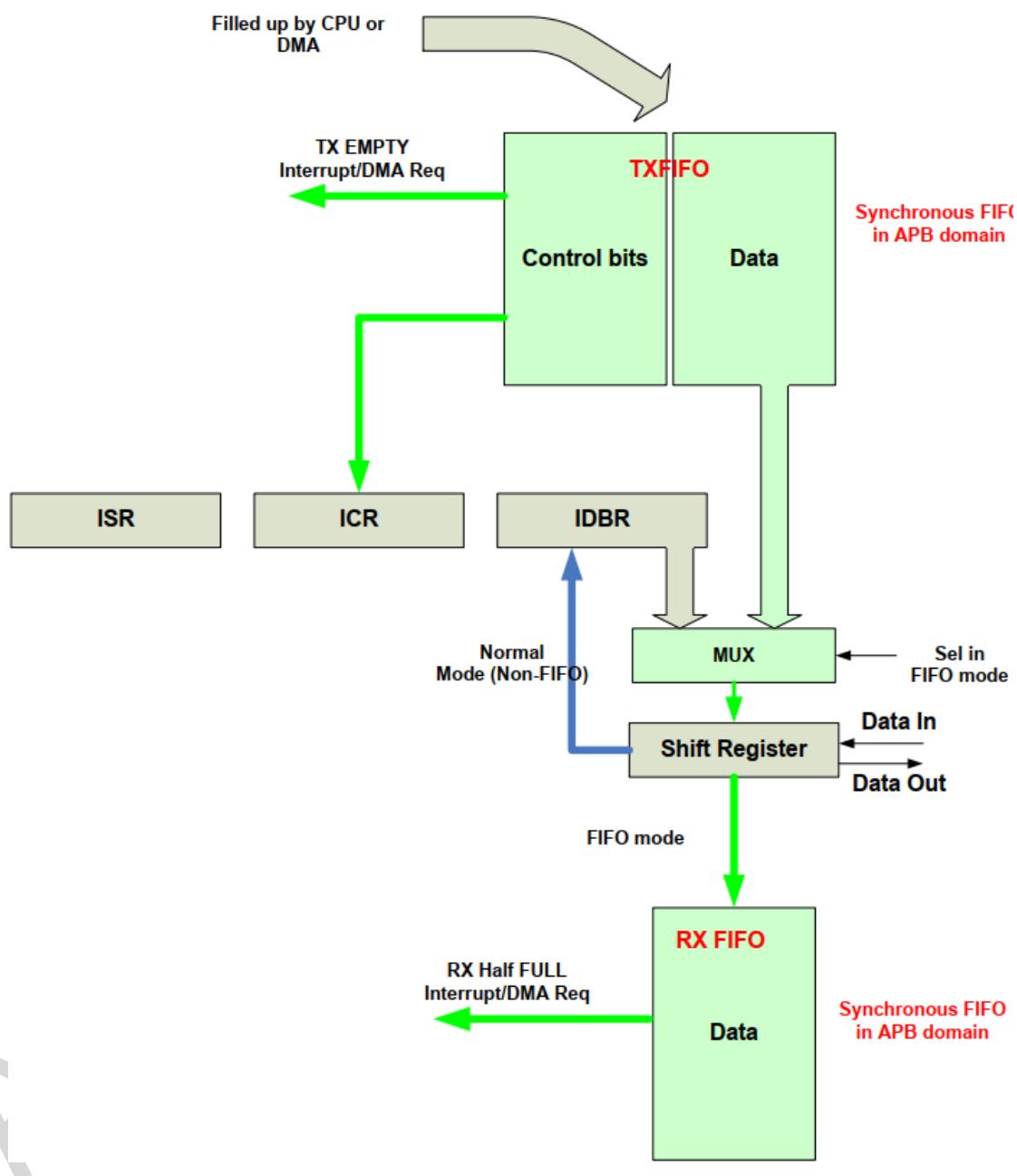


Figure 15-3 FIFO Mode Block Diagram

FIFOs can be used for both transmission and reception to help reducing the empty and full interrupts of register I2Cx\_DBR. The FIFOs allow reading and writing multiple bytes without interrupting the CPU after each byte.

DMA is used to improve I2C transactions (typically more than 8 Bytes). The entire transaction can be completed by DMA without generating multiple FIFO interrupts.

The FIFO mode is backward compatible, and it is disabled by clearing the FIFO\_EN bit in the *I2Cx\_CR* register.

Transmit FIFO has a width of 12 (4 control bits and 8 data bits) and a depth of 8. The 4 control bits are bits[3:0] in register *I2Cx\_CR*, which are required for each data byte transmission. After a byte is transmitted, the next byte is copied from the TX FIFO into the shift register, and the control bits are copied into the *I2Cx\_CR* register. This byte is now transferred, and it continues like that until the Stop bit is set.

Receive FIFO has a width of 8 (8 data bits) and a depth of 16, which is used to store the received data. The control bits for each byte and dummy data are put in the corresponding position in the TX FIFO. When the RX FIFO is half full, an interrupt or DMA request is sent out for the data in the RX FIFO to be read out.

In order to support the FIFO mode and fully utilize its capabilities, the following status and control bits need to be configured:

- (1) Set the FIFO\_EN bit in register *I2Cx\_CR* to enable the FIFO mode.
- (2) Set the TXBEGIN bit in register *I2Cx\_CR* to start a transaction.
- (3) Bits[31:28] in register *I2Cx\_CR* enables or disables all the FIFO related interrupts, and bits [31:28] in register *I2Cx\_SR* is used to inquire the interrupt status.
- (4) TXDONE interrupt is generated when each transaction is completed (that is, a Stop condition is sent).
- (5) The DMA\_EN bit in register *I2Cx\_CR* is used to enable/disable DMA mode.

In DMA mode, all the FIFO related interrupts must be disabled in register *I2Cx\_CR* (bits[31:28]), and the DMA\_EN bit in this register must be set. In this way, all DMA requests are sent to the DMA without interrupting the CPU. The TXDONE\_IE bit in the *I2Cx\_CR* register needs to be set in both FIFO and DMA modes to generate an interrupt to the CPU after each transaction is completed.

## 15.9 I2C Slave Mode

**Table 15-3 Slave Transactions**

Operation	Op.mode	Description
Slave receive mode (default)	Slave receive only	<ul style="list-style-type: none"> <li>The I2C monitors all slave address transactions.</li> <li>The UE bit in register <i>I2Cx_CR</i> must be set.</li> <li>I2C monitors the Start conditions on the bus. When a Start condition is detected, the interface reads the first 8 bits of data and compares the most significant 7 bits with those in the <i>I2Cx_SAR</i> register. If they match, the I2C sends an ACK.</li> <li>If the 8th bit (R/nW bit) of the first byte is low, then I2C stays in slave receive mode, and the SAD bit in register <i>I2Cx_SR</i> is cleared. If the R/nW bit is high, I2C switches to slave transmit mode and sets the SAD bit in register <i>I2Cx_SR</i>.</li> </ul>
Set the slave address detection bit	Slave receive Slave transmit	<ul style="list-style-type: none"> <li>Indicates that the interface has detected the matching slave address.</li> <li>If enabled, an slave address detection interrupt is generated after the matching slave address is received and acknowledged, and the SAD bit in register <i>I2Cx_SR</i> is set.</li> </ul>
Read one byte from <i>I2Cx_DB</i>	Slave receive only	<ul style="list-style-type: none"> <li>8 bits are read from the serial bus into the shift register. When a full byte has been received and the ACK/NAK bit is completed, the byte in the shift register is transferred to the <i>I2Cx_DB</i> register.</li> <li>When the IRF bit in register <i>I2Cx_SR</i> is set, and the TB bit in register <i>I2Cx_CR</i> is cleared, if enabled, the <i>I2Cx_DB</i> receive-full interrupt is generated.</li> <li>Software reads one data byte from the <i>I2Cx_DB</i> register, then configures the ACKNAK bit in register <i>I2Cx_CR</i> as required and sets the TB bit in register <i>I2Cx_CR</i>. This makes the I2C exit from the wait state and continue to receive data from the master transmitter.</li> </ul>
Transmit ACK to master transmitter	Slave receive only	<ul style="list-style-type: none"> <li>As a slave receiver, the I2C pulls the SDA line low to generate the ACK when SCL is high.</li> <li>ACK/NAK is controlled by bit ACKNAK in register <i>I2Cx_CR</i>.</li> </ul>
Write one byte to <i>I2Cx_DB</i>	Slave transmit only	<ul style="list-style-type: none"> <li>When the ITE bit in register <i>I2Cx_SR</i> is set and the TB bit in register <i>I2Cx_CR</i> is cleared, if enabled, the <i>I2Cx_DB</i> transmit-empty interrupt is generated.</li> <li>Software writes a byte into register <i>I2Cx_DB</i> and then sets the TB bit in register <i>I2Cx_CR</i> to start the transmission.</li> </ul>
Waiting for ACK from master receiver	Slave transmit only	As a slave transmitter, the I2C releases the SDA line for the master receiver to pull the line low to transmit the ACK.

## 15.10 Clock and Reset

Each I2C interface has independent APB bus clock and independent APB bus reset. Software must ensure that the I2C unit is disabled (*I2Cx\_CR*[UE]=0) before reset, and ensure that the I2C bus is idle (*I2Cx\_SR*[IBB]=0) when the unit is enabled after reset. When reset, all registers except the *I2Cx\_SAR*, return to the default reset condition. *I2Cx\_SAR* is not affected by a reset.

Steps for I2C clock reset:

1. Set the UR bit in the *I2Cx\_CR* register, and clear the remaining bits of this register.
2. Clear the *I2Cx\_SR* register.
3. Clear the UR bit in the *I2Cx\_CR* register.

## 15.11 Interrupts

I2C interrupts are configured by register *I2Cx\_CR*, and the interrupt status can be obtained by querying the corresponding bit in register *I2Cx\_SR*.

## 15.12 DMA

DMA (Direct Memory Access) is enabled by setting the DMA\_EN bit in register *I2Cx\_CR* to support transmission and reception.

## 15.13 I2C Registers

I2C0 Base Address: 0x40007000

I2C1 Base Address: 0x40014000

I2C2 Base Address: 0x40015000

**Table 15-4 I2C Registers Summary**

Register	Offset	Description
I2Cx_CR	0x000	Control Register
I2Cx_SR	0x004	Status Register
I2Cx_SAR	0x008	Slave Address Register
I2Cx_DBR	0x00C	Data Buffer Register
I2Cx_LCR	0x010	Load Count Register
I2Cx_WCR	0x014	Wait Count Register
I2Cx_RST_CYCL	0x018	Reset SCL Cycle
I2Cx_BMR	0x01C	Bus Monitor Register
I2Cx_WFIFO	0x020	Write FIFO Register
I2Cx_WFIFO_WPTR	0x024	Write FIFO Write Pointer Register
I2Cx_WFIFO_RPTR	0x028	Write FIFO Read Pointer Register
I2Cx_RFIFO	0x02C	Read FIFO Register
I2Cx_RFIFO_WPTR	0x030	Read FIFO Write Pointer Register
I2Cx_RFIFO_RPTR	0x034	Read FIFO Read Pointer Register
I2Cx_RESV[2]	0x038	Reserved
I2Cx_WFIFO_STATUS	0x040	Write FIFO Status Register
I2Cx_RFIFO_STATUS	0x044	Read FIFO Status Register

### 15.13.1 I2Cx\_CR (x=0, 1, 2)

Offset: 0x000

Reset Value: 0x00000200

31	30	29	28	27	26	25	24
RXOV_IE	RXF_IE	RXHF_IE	TXE_IE	TXDONE_IE	MSDE	MSDIE	SSDIE
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h
23	22	21	20	19	18	17-16	
SADIE	BEIE	RESERVED	DRFIE	ITEIE	ALDIE	RESERVED	
rw-0h	rw-0h	r-0h	rw-0h	rw-0h	rw-0h	r-0h	
15	14	13	12	11	10	9-8	
RESERVED	UE	SCLE	MA	IBRR	UR	MODE	
r-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-2h	
7	6	5	4	3	2	1	0
DMA_EN	RESERVED	FIFOEN	TXBEGIN	TB	ACKNAK	STOP	START
rw-0h	r-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bit 31 RXOV\_IE:** Receive FIFO overrun interrupt enable.

- 0: receive FIFO overrun interrupt disabled
- 1: receive FIFO overrun interrupt enabled

**Bit 30 RXF\_IE:** Receive FIFO full interrupt enable.

- 0: receive FIFO full interrupt disabled
- 1: receive FIFO full interrupt enabled

**Bit 29 RXHF\_IE:** Receive FIFO half full interrupt enable.

- 0: Receive FIFO half full interrupt disabled
- 1: Receive FIFO half full interrupt enabled

**Bit 28 TXE\_IE:** Transmit FIFO empty interrupt enable.

- 0: Transmit FIFO empty interrupt disabled
- 1: Transmit FIFO empty interrupt enabled

**Bit 27 TXDONE\_IE:** Transaction done interrupt enable.

- 0: transaction done interrupt disabled
- 1: transaction done interrupt enabled

**Bit 26 MSDE:** Master Stop detection enable.

- 0: Master Stop detection disabled
- 1: Master Stop detection enabled

**Bit 25 MSDIE:** Master Stop detection interrupt enable.

- 0: Master Stop detection interrupt disabled
- 1: Master Stop detection interrupt enabled

**Bit 24 SSDIE:** Slave Stop detection interrupt enable.

- 0: Slave Stop detection interrupt disabled
- 1: Slave Stop detection interrupt enabled

**Bit 23 SADIE:** Slave address detection interrupt enable.

- 0: Slave address detection interrupt disabled
- 1: Slave address detection interrupt enabled

**Bit 22 BEIE:** Bus error interrupt enable.

- 0: Bus error interrupt disabled
- 1: Bus error interrupt enabled

**Bit 21 RESERVED:** Must be kept, and cannot be modified.

**Bit 20 DRFIE:** I2Cx\_DBR receive-full interrupt enable.

- 0: I2Cx\_DBR receive-full interrupt disabled
- 1: I2Cx\_DBR receive-full interrupt enabled

**Bit 19 ITEIE:** I2Cx\_DBR transmit-empty interrupt enable.

- 0: I2Cx\_DBR transmit-empty interrupt disabled
- 1: I2Cx\_DBR transmit-empty interrupt enabled

**Bit 18 ALDIE:** Arbitration loss detection interrupt enable.

- 0: Arbitration loss detection interrupt disabled
- 1: Arbitration loss detection interrupt enabled

**Bits 17-15 RESERVED:** Must be kept, and cannot be modified.

**Bit 14 UE:** I2C unit enable.

- 0: I2C unit disabled
- 1: I2C unit enabled (the default is slave receive mode)

Software must ensure that the I2C bus is idle before enabling the I2C unit, and ensure that the internal I2C clock is enabled before setting or clearing this bit.

**Bit 13 SCLE:** SCL enable.

- 0: disable the I2C from driving the SCL line
- 1: enable the I2C clock output for master-mode operation

**Bit 12 MA:** Master abort.

This bit is used for the I2C to generate a Stop condition in master mode.

- 0: Stop condition is generated when the STOP bit in this register is set
- 1: Stop condition is generated without data transmission

In master transmit mode, after a data byte is transmitted, the TB bit in this register is cleared and the ITE bit in register *I2Cx\_SR* is set. When no more data bytes need to be sent, setting the MA bit to generate a Stop condition to free the bus. In master receive mode, when a NAK is sent with the STOP bit=0 and without a Repeated Start condition followed, setting the MA bit to generate a Stop condition to free the bus. The TB bit in this register must remain clear.

**Bit 11 IBRR:** I2C bus reset request.

- 0: no action
- 1: I2C bus reset, and this bit is cleared automatically

**Bit 10 UR:** Unit reset.

- 0: no action
- 1: reset the I2C unit

**Bits 9-8 MODE:** Bus clock mode for the master.

- 00: standard mode – 100 Kbps
- 01: fast mode – 400 Kbps

**Bit 7 DMA\_EN:** DMA enable.

- 0: DMA requests disabled
- 1: DMA requests enabled

**Bit 6 RESERVED:** Must be kept, and cannot be modified.

**Bit 5 FIFOEN:** FIFO mode enable.

- 0: FIFO mode disabled
- 1: FIFO mode enabled

**Bit 4 TXBEGIN:** Transaction begin.

- 0: no action
- 1: new transaction begins

This bit is cleared by hardware after a Stop condition is generated, and the software needs to set it again to start a new transaction.

**Bit 3 TB:** Transfer byte, used to send or receive a byte on the I2C bus.

- 0: cleared by I2C when one byte is sent or received
- 1: send or receive a byte

The I2C unit monitors this bit to determine whether the byte transfer has completed. In master or slave mode, after each byte including the ACK is transferred, I2C holds the SCL line low until the TB bit is set.

**Bit 2 ACKNAK:** The positive/negative acknowledge (ACK/NAK) control bit in master receive mode.

- 0: send ACK after receiving a data byte
- 1: send NAK after receiving a data byte

In slave mode, when the I2C responds to its slave address or the reception is complete, it automatically sends an ACK, regardless of whether the ACKNAK bit is set.

**Bit 1 STOP:** Generate a Stop condition.

- 0: no action
- 1: generate Stop condition

This bit is used to generate a Stop condition on the I2C bus after the transmission of the next data byte in master mode. In master receive mode, the ACKNAK bit and the STOP bit must be set to 1 at the same time.

**Bit 0 START:** Generate a Start condition.

- 0: no action
- 1: generate a Start condition

This bit is used to generate a Start condition on the I2C bus in master mode.

### 15.13.2 I2Cx\_SR (x=0, 1, 2)

Offset: 0x004

Reset Value: 0x00000000

31	30	29	28	27	26	25
RXOV	RXF	RXHF	TXE	TXDONE	MSD	RESERVED
rw1c-0h	rw1c-0h	rw1c-0h	rw1c-0h	rw1c-h	r1ch	r-0h
24	23	22	21	20	19	18
SSD	SAD	BED	RESERVED	IRF	ITE	ALD
rw1c-0h	rw1c-0h	rw1c-0h	r-0h	rw1c-0h	rw1c-0h	rw1c-0h
17	16	15	14	13-8	7-0	
RESERVED	IBB	UB	ACKNAK	RESERVED	RESERVED	
r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	

**Bit 31 RXOV:** Receive FIFO overrun flag.

- 0: no receive FIFO overrun occurred
- 1: receive FIFO overrun occurred, and it is cleared by software writing 1 to it

**Bit 30 RXF:** Receive FIFO full flag.

- 0: receive FIFO is not full
- 1: receive FIFO is full, and it is cleared by software writing 1 to it

**Bit 29 RXHF:** Receive FIFO half-full flag.

- 0: receive FIFO is not half full
- 1: receive FIFO is half full, and it is cleared by software writing 1 to it

**Bit 28 TXE:** Transmit FIFO empty flag.

- 0: transmit FIFO is not empty
- 1: transmit FIFO is empty, and it is cleared by software writing 1 to it

**Bit 27 TXDONE:** Transaction done flag (used in FIFO mode).

- 0: transaction in progress
- 1: transaction is done, and it is cleared by software writing 1 to it

**Bit 26 MSD:** Master Stop detection flag (only valid in master mode).

- 0: no master Stop detected
- 1: master Stop detected, and it is cleared by software writing 1 to it

**Bit 25 RESERVED:** Must be kept, and cannot be modified.

**Bit 24 SSDIE:** Slave Stop detection flag.

- 0: no slave Stop detected
- 1: slave Stop was detected, and it is cleared by software writing 1 to it

**Bit 23 SAD:** Slave address detection flag.

- 0: no matching slave address detected
- 1: matching slave address detected, and it is cleared by software writing 1 to it

**Bit 22 BED:** Bus error detection flag.

- 0: no bus error detected
  - 1: bus error detected, and it is cleared by software writing 1 to it
- This bit is set in two cases:
- As a master transmitter, the I2C did not receive an ACK after sending a byte.
  - As a slave receiver, the I2C generates a NAK.

**Bit 21 RESERVED:** Must be kept, and cannot be modified.

**Bit 20 IRF:** I2Cx\_DB<sub>R</sub> receive full flag.

- 0: the I2Cx\_DB<sub>R</sub> register has not received a new data byte or the I2C bus is idle
- 1: the I2Cx\_DB<sub>R</sub> register received a new data byte, and it is cleared by software writing 1 to it

**Bit 19 ITE:** I2Cx\_DB<sub>R</sub> transmit empty flag.

- 0: the data byte transmit in progress
- the I2C has finished transmitting a byte on the I2C bus, and it is cleared by software writing 1 to it

**Bit 18 ALD:** Arbitration loss detection flag, used in multi-master scenarios.

- 0: I2C wins the arbitration or no arbitration took place
- 1: I2C loses the arbitration, and it is cleared by software writing 1 to it

**Bit 17 RESERVED:** Must be kept, and cannot be modified.

**Bit 16 IBB:** I2C bus busy flag.

- 0: Bus is idle or the bus is being used by the I2C interface
- 1: Bus is busy but not used by the I2C interface

**Bit 15 UB:** I2C unit busy flag.

- 0: I2C unit is idle
- 1: I2C unit is busy

**Bit 14 ACKNAK:** ACK/NAK status flag.

- 0: I2C received or sent an ACK
- 1: I2C received or sent a NAK

In slave transmit mode, this bit is used to determine whether the byte transmitted is the last one. This bit is updated after each byte and ACK/NAK information is received.

**Bits 13-0 RESERVED:** Must be kept, and cannot be modified.

### 15.13.3 I2Cx\_SAR (x=0, 1, 2)

Offset: 0x008

Reset Value: 0x00000000

31-7	6-0
RESERVED	SLAVE_ADDRESS
r-0h	rw-0h

**Bits 31-7 RESERVED:** Must be kept, and cannot be modified.

**Bits 6-0 SLAVE\_ADDRESS:** The ASR6601 I2C slave address used in slave mode.

#### 15.13.4 I2Cx\_DBR (x=0, 1, 2)

Offset: 0x00C

Reset Value: 0x00000000

31-8	7-0
RESERVED	DATA_BUFFER
r-0h	rw-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 DATA\_BUFFER:** Buffer for I2C bus transmit/receive data.

#### 15.13.5 I2Cx\_LCR (x=0, 1, 2)

Offset: 0x010

Reset Value: 0x18183a7e

31-18	17-9	8-0
RESERVED	FLV	SLV
r-1818h	rw-1dh	rw-7eh

**Bits 31-18 RESERVED:** Must be kept, and cannot be modified.

**Bits 17-9 FLV:** Phase decrementer load value for fast mode SCL in master mode.

**Bits 8-0 SLV:** Phase decrementer load value for standard mode SCL in master mode.

#### 15.13.6 I2Cx\_WCR (x=0, 1, 2)

Offset: 0x14

Reset Value: 0x0000143a

31-5	4-0
RESERVED	COUNT
r-a1h	rw-1ah

**Bits 31-5 RESERVED:** Must be kept, and cannot be modified.

**Bits 4-0 COUNT:** Counter values for defining the setup and hold times in standard and fast modes.

### 15.13.7 I2Cx\_RST\_CYCL (x=0, 1, 2)

Offset: 0x018

Reset Value: 0x00000000

31-4	3-0
RESERVED	RST_CYC
r-0h	rw-0h

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bits 3-0 RST\_CYC:** Serial bus reset SCL cycle count.

### 15.13.8 I2Cx\_BMR (x=0, 1, 2)

Offset: 0x01C

Reset Value: 0x00000003

31-2	1	0
RESERVED	SCL	SDA
r-0h	r-1h	r-1h

**Bits 31-2 RESERVED:** Must be kept, and cannot be modified.

**Bit 1 SCL:** SCL pin state.

**Bit 0 SDA:** SDA pin state.

### 15.13.9 I2Cx\_WFIFO (x=0, 1, 2)

Offset: 0x020

Reset Value: 0x00000000

31-12	11-8	7-0
RESERVED	CONTROL	DATA
r-0h	w-0h	w-0h

**Bits 31-12 RESERVED:** Must be kept, and cannot be modified.

**Bits 11-8 CONTROL:** I2C bus transmit/receive data control bits.

**Bits 7-0 DATA:** I2C bus send data for write transactions and dummy data for read transactions.

### 15.13.10 I2Cx\_WFIFO\_WPTR (x=0, 1, 2)

Offset: 0x024

Reset Value: 0x00000000

31-4	3-0
RESERVED	DATA
r-0h	rw-0h

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bits 3-0 DATA:** The position in the Transmit FIFO where the software will write the next entry.

### 15.13.11 I2Cx\_WFIFO\_RPTR (x=0, 1, 2)

Offset: 0x028

Reset Value: 0x00000000

31-4	3-0
RESERVED	DATA
r-0h	rw-0h

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bits 3-0 DATA:** The position in the Transmit FIFO where the hardware will read the next entry.

### 15.13.12 I2Cx\_RFIFO (x=0, 1, 2)

Offset: 0x02C

Reset Value: 0x00000000

31-8	7-0
RESERVED	DATA
r-0h	r-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 DATA:** I2C bus receive data for read transactions.

### 15.13.13 I2Cx\_RFIFO\_WPTR (x=0, 1, 2)

Offset: 0x030

Reset Value: 0x00000000

31-4	3-0
RESERVED	DATA
r-0h	r-0h

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bits 3-0 DATA:** The position in the Receive FIFO where the hardware will write the next entry.

### 15.13.14 I2Cx\_RFIFO\_RPTR (x=0, 1, 2)

Offset: 0x034

Reset Value: 0x00000000

31-4	3-0
RESERVED	DATA
r-0h	r-0h

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bits 3-0 DATA:** The position in the Receive FIFO where the software will read the next entry.

### 15.13.15 I2Cx\_WFIFO\_STATUS (x=0, 1, 2)

Offset: 0x040

Reset Value: 0x00000000

31-16	15-9	8-1	0
RESERVED	WFIFO_SIZE	WFIFO_EMPTY	WFIFO_FULL
r-0h	r-0h	r-0h	r-0h

**Bits 31-6 RESERVED:** Must be kept, and cannot be modified.

**Bits 5-2 WFIFO\_SIZE:** The Transmit FIFO size.

**Bit 1 WFIFO\_EMPTY:** Transmit FIFO empty.

**Bit 0 WFIFO\_FULL:** Transmit FIFO full.

### 15.13.16 I2Cx\_RFIFO\_STATUS (x=0, 1, 2)

Offset: 0x044

Reset Value: 0x00000000

31-24	23-16	15-8	7-4
RESERVED	RESERVED	RESERVED	RFIFO_SIZE
r-0h	r-0h	r-0h	r-0h
<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
RFIFO_EMPTY	RFIFO_FULL	RFIFO_HALFFULL	RFIFO_OVERRUN
r-0h	r-0h	r-0h	r-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-4 RFIFO\_SIZE:** The Receive FIFO size.

**Bit 3 RFIFO\_EMPTY:** Receive FIFO empty.

**Bit 2 RFIFO\_FULL:** Receive FIFO full.

**Bit 1 RFIFO\_HALFFULL:** Receive FIFO half full.

**Bit 0 RFIFO\_OVERRUN:** Receive FIFO overrun.

# 16.

# ADC

## 16.1 Introduction

The 12-bit ADC (Analog to Digital Converter) has 8 external channels and 7 internal channels for measuring signals with up to 1M sampling rate. The internal VBAT/3 channel allows the ADC to measure the VBAT/3 signal. ADC analog input channels can be configured in single ended (range: 0.1V~1.1V) or differential mode (range: -1.0~1.0V). The ADC conversion supports a programmable channel sequence with a length between 1 to 16 in continuous, single, or discontinuous sampling modes. The conversion can be initiated by software or hardware configurable trigger sources. In addition, the ADC supports DMA request and interrupt generation.

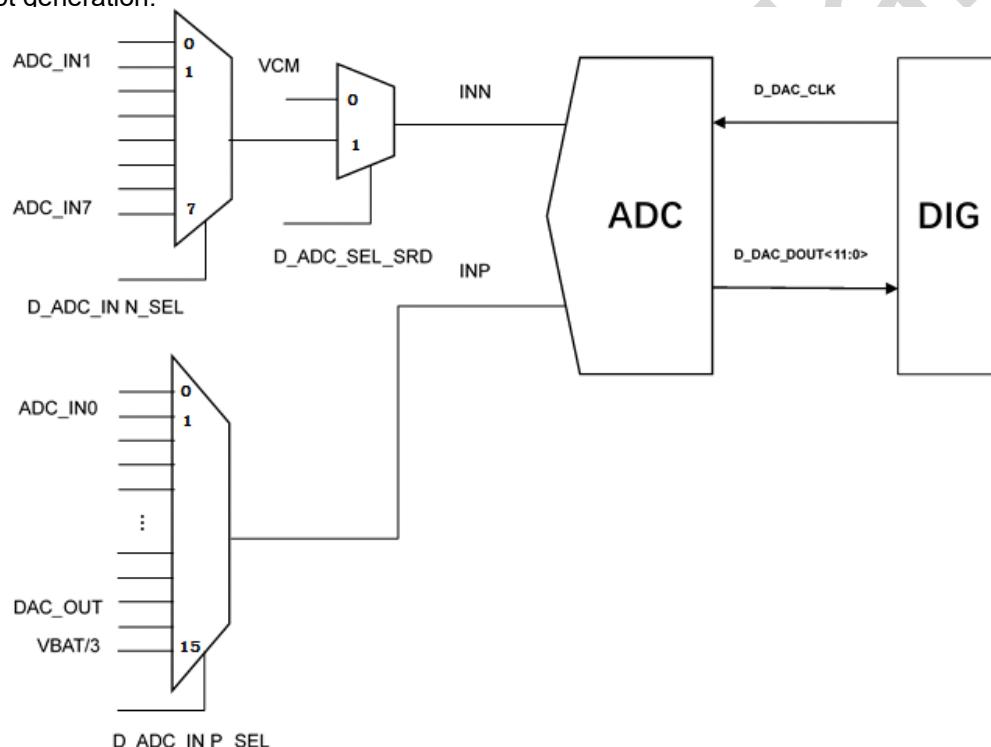


Figure 16-1 ADC Diagram

## 16.2 ADC Input Mode

Channels can be configured to be either single-ended input or differential input through the differential mode selection register ([ADC\\_DIFFSEL](#)). External channels support both single ended and differential modes, and internal channels only support single-ended mode. A fixed combination is required in differential mode, channel 0 and 1 is a differential group, channel 2 and 3 is a differential group, channel 4 and 5 is a differential group, and channel 6 and 7 is a differential group. The width of the last data result of a conversion stored in the data buffer is 12-bit, where the MSB is a sign bit and the other 11 bits are data bits in differential input mode, but no sign bit is presented in single-ended mode, all 12 bits are data bits.

## 16.3 Sampling Channels

- **8 External Channels:** In single-ended mode, each channel is independent. In differential mode, every two channels form a group and cannot be split.
- **7 Internal Channels:** include DAC output, internal VRef, VBAT/3 (battery voltage), Vts (internal temperature sensor) and a channel dedicated for internal tests. The internal channels do not support differential mode.

Table 16-1 ADC Sampling Channels

Channel No.	Signal	Source
1	ADC_PAD_IN<0>	gpio11
2	ADC_PAD_IN<1>	gpio08
3	ADC_PAD_IN<2>	gpio05
4	ADC_PAD_IN<3>	gpio04
5	ADC_PAD_IN<4>	gpio50
6	ADC_PAD_IN<5>	gpio49
7	ADC_PAD_IN<6>	gpio48
8	ADC_PAD_IN<7>	gpio47
9	OPA0_ADC_OUT	
10	OPA1_ADC_OUT	
11	OPA2_ADC_OUT	
12	DCTEST_OUT	
13	TD_OUT_TEST	
14	DAC_CORE_AOUT	
15	VBAT31	

To generate VBAT31 signal, it is necessary to enable VBAT/3 voltage division circuit by setting the D\_VBAT\_DIV3\_EN bit in the RESV1 register of the analog part. This channel is nominally 1/3 of VBAT and it is 1/3.06 of VBAT actually.

## 16.4 Trigger Source

- **Software trigger:** The conversion starts immediately when a rising edge on the START bit of ADC\_CR is detected.
- **Hardware trigger:** The conversion is triggered by Timer or IO, containing 10 selectable trigger sources with a configurable level.

## 16.5 Low-power Operation

A new trigger request can only be received after the [ADC\\_DR](#) register has been read or the EOC flag is cleared, which can prevent overrun but might bypass some trigger requests.

## 16.6 Overrun

Configure the ADC\_DR register to hold old data or update with new data when an overrun occurs.

## 16.7 Conversion Modes

The sampling mode is configured by the CONV\_MODE bit in register [ADC\\_CFGR](#): The ADC conversion supports a programmable channel sequence with a length between 1 to 16, and the channels can be configured in single-ended or differential mode. In differential mode, only the P input of the channels in the sequence need to be configured. A channel can be selected more than once in the sequence, and thus the conversion of the same channel will be performed multiple times in each sequence. The conversion sequence is configured through the [ADC\\_SEQR0](#) and [ADC\\_SEQR1](#) registers, and every 4 bits configures one channel number. The two 32-bit registers have 64 bits in total, and thus up to 16 channels can be configured to be converted.

- **Continuous Mode:** When a software or hardware trigger event occurs, the ADC performs a sequence of conversions. After the conversions are completed, the ADC automatically restarts and continuously performs the same sequence of conversions until a STOP command is issued by software.
- **Single Mode:** When a software or hardware trigger event occurs, the ADC performs a single sequence of conversions and then stops automatically after the conversions are completed.
- **Discontinuous Mode:** Each conversion defined in the sequence requires a hardware or software trigger event. When a sequence of conversions is completed, a new trigger event restarts the conversion of the first channel defined in the sequence. While in continuous and single modes, the complete sequence is converted upon a single trigger event.

## 16.8 Voltage Reference

The reference voltage is configured through the D\_ADC\_SEL\_VREF bit in the RST register of the analog part. The external or internal reference voltage is configured by clearing or setting this bit, and the default value is 1.

- **Internal Voltage Reference:** VRef, 1.2V.
- **External Voltage Reference:** VREFP/3, VREFP≤3.6V. VREFP is connected to VDDA in the ASR6601CB (48-pin) chip.

## 16.9 Data Buffer

For the 12-bit data buffer, the most significant bit is the sign bit in differential input mode.

ADC Value	Definition (differential)	Definition (single-ended)
1111_1111_1111	+Vref <sup>(1)</sup>	+Vref <sup>(1)</sup>
...	...	...
...	...	...
...	...	...
...	...	...
...	...	...
1000_0000_0001	+Vref/2048 <sup>(1)</sup>	+Vref/2+Vref/4096 <sup>(1)</sup>
1000_0000_0000	0	+Vref/2 <sup>(1)</sup>
0111_1111_1111	-Vref/2048 <sup>(1)</sup>	+Vref/2-Vref/4096 <sup>(1)</sup> ...
...	...	...
...	...	...
...	...	...
...	...	...
0000_0000_0000	-Vref <sup>(1)</sup>	0

<sup>(1)</sup> This value should be calibrated by software to correct error on the ADC hardware.

The measure range in differential mode is -1.0~1.0V, and the measure range in single-ended mode is 0.1~1.1V. In order to correct the error on the ADC hardware, ASR6601 is calibrated before leaving the factory. The calibration data (Offset and Gain) are stored in Flash. The user needs to convert the data read from register [ADC\\_DR](#) to get the final AD value. The formula is as follows:

$$V = (V_{out} - Offset) / Gain$$

where **V<sub>out</sub>** is the value readed from the data buffer.

## 16.10 DMA

When the 12-bit data buffer is full, the DMA request is generated if the DMA\_EN bit in register ADC\_CFGR is set. DMA request is disabled by writing '0' to the DMA\_EN bit.

## 16.11 Interrupts

The interrupt sources include the end of conversion (EOC), end of a sequence of conversions (EOS), and a data overrun (OVERRUN). The interrupts are enabled through register [ADC\\_IER](#), and the interrupt status is inquired through the [ADC\\_ISR](#) register.

## 16.12 Wakeup

The MCU wakes up from the Sleep mode if an interrupt or event is generated.

## 16.13 Clock and Reset

The ADC bus reset and the ADC clock reset are independent. The ADC module supports the APB bus clock. The ADC interface clock source can be one of the following sources (divided or not): sys\_clk, apb\_x\_pclk, pll\_clk or rco48m\_clk.

## 16.14 ADC Registers

Base Address: 0x40017000

**Table 16-2 ADC Registers Summary**

Register	Offset	Description
ADC_CR	0x000	ADC Control Register
ADC_CFGR	0x004	ADC Configuration Register
ADC_SEQR0	0x008	ADC Sequence Register 0
ADC_SEQR1	0x00C	ADC Sequence Register 1
ADC_DIFFSEL	0x010	ADC Differential Mode Selection Register
ADC_ISR	0x014	ADC Interrupt and Status Register
ADC_IER	0x018	ADC Interrupt Enable Register
ADC_DR	0x01C	ADC Data Register

### 16.14.1 ADC\_CR

Offset: 0x000

Reset Value: 0x00000000

31-4	3	2	1	0
RESERVED	STOP	START	DIS	EN
r-0h	rw-0h	rw-0h	w-0h	rw-0h

**Bits 31-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 STOP:** ADC stop conversion command.

- 0: no action
- 1: Stop the ADC. Read 1 means that the STOP command is in progress

**Notes:**

1. Software writes 1 to this bit to stop and discard an ongoing conversion, thus the conversion sequence is reset; this bit is cleared automatically by hardware.
2. After the START bit is cleared, software must wait for 3 ADCCLK ticks before reconfigure the START bit; or wait for 1 CLK\_DIV (set in register [ADC\\_CFGR](#)) tick to set the DIS bit to disable the ADC.
3. Software is allowed to set this bit only when START=1 and STOP=0.
4. Before setting the STOP bit, it is recommended to disable the trigger source first, or keep the trigger level in an invalid state.

**Bit 2 START:** ADC start conversion command.

- 0: no action
- 1: start the ADC. Read 1 means that the ADC conversion is being performed

This bit is set by software to start the ADC conversion. Software is allowed to set this bit only when EN=1 and DIS=0. Whether an ADC conversion starts immediately (software trigger mode) or won't start until a hardware trigger event occurs depends on the TRIG\_SEL[18:17] configuration bits in register [ADC\\_CFGR](#).

This bit is automatically cleared by hardware in the following the following circumstances:

1. In single conversion mode, if software trigger is selected (TRIG\_SEL=00 in register [ADC\\_CFGR](#)), the START bit is cleared when the EOS flag in register [ADC\\_ISR](#) is set.
2. In discontinuous conversion mode, if software trigger is selected, the START bit is cleared when the EOC flag in register [ADC\\_ISR](#) is set.
3. In any case, after the execution of the STOP command, the START and STOP bits are cleared by hardware at the same time.

**Bit 1 DIS:** ADC disable.

- 0: no action
- 1: disable the ADC

Software is allowed to set this bit only when EN=1 and START=0 (no conversion in progress).

**Bit 0 EN:** ADC enable.

- 0: no action
- 1: enable the ADC. Read 1 means that the ADC is enabled

This bit is set by software to enable the ADC module. The software is allowed to set this bit only when all bits of register [ADC\\_CR](#) equal 0. Reading this bit reflects whether the ADC is enabled or not. The software must wait at least 100us for the ADC analog circuit to stabilize after initialization before it enables the ADC conversion.

## 16.14.2 ADC\_CFGR

Offset: 0x004

Reset Value: 0x00000002

31-24	23	22	21-20	19	18-17
RESERVED	RESERVED	WAIT_MODE	CONV_MODE	OVERRUN_MODE	TRIG_SEL
r-0h	r-0h	r-0h	r-0h	r-0h	rw-0h
<b>16</b>	<b>15-13</b>	<b>12</b>	<b>11-8</b>	<b>7-0</b>	
EXT_TRIG_SEL[3]	EXT_TRIG_SEL[2:0]	DMA_EN	CLK_DIV[11:8]	CLK_DIV[7:0]	
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-2h

**Bits 31-23 RESERVED:** Must be kept, and cannot be modified.

**Bit 22 WAIT\_MODE:** Wait conversion mode.

- 0: wait conversion mode disabled
- 1: wait conversion mode enabled

Wait for the conversion mode, that is, a new trigger request can only be received after register ADC\_DR has been read or the EOC flag (in ADC\_ISR) is cleared, which can prevent overrun but may bypass some trigger requests.

Software is allowed to write this bit only when START=0 (in register [ADC\\_CR](#)).

**Bits 21-20 CONV\_MODE:** ADC conversion mode selection.

- 00: single conversion mode
- 01: continuous conversion mode
- 1x: discontinuous conversion mode

Software is allowed to write this bit only when START=0 (in register [ADC\\_CR](#)).

**Notes:**

1. In single conversion mode, when a software or hardware trigger event occurs, the ADC performs a single sequence of conversions (set by ADC\_SEQR0/1). After the conversions are completed, the ADC stops until a new trigger occurs.
2. In continuous conversion mode, when a software or hardware trigger event occurs, the ADC performs a sequence of conversions (set by ADC\_SEQR0/1). After the conversions are completed, the ADC automatically re-starts and continuously performs the same sequence of conversions until a STOP command is issued by software.
3. In discontinuous conversion mode, each conversion defined in the sequence (set by ADC\_SEQR0/1) requires a hardware or software trigger event. When a sequence of conversions is completed, a new trigger event restarts the conversion of the first channel defined in the sequence.

**Bit 19 OVERRUN\_MODE:** Overrun management mode.

- 0: the old data in the ADC\_DR register is hold when an overrun is occurred
- 1: the ADC\_DR register is overwritten with the newly converted data when an overrun is occurred

Software is allowed to write this bit only when START=0 (in register [ADC\\_CR](#)).

**Bits 18-17 TRIG\_SEL:** Trigger mode and polarity selection.

- 00: software trigger. The conversion starts immediately when a rising edge on the START bit of ADC\_CR is detected
- 01: hardware trigger detection on the rising edge
- 10: hardware trigger detection on the falling edge
- 10: hardware trigger detection on the falling edge

Software is allowed to write this bit only when START=0 (in register [ADC\\_CR](#)).

When a hardware trigger is selected, after the START bit is configured, software must wait for 3 ADCCLK ticks before receiving the trigger signal.

**Bits 16-13 EXT\_TRIG\_SEL:** External trigger selection for the start of ADC conversion.

- 0000~0100: reserved
- 0101: GPIO47
- 0110: GPIO31
- 0111: GPIO19
- 1000: GPIO10
- 1001: GPTIM1\_TRGO
- 1010: GPTIM0\_CH2\_OUT
- 1011: GPTIM3\_TRGO
- 1100: GPTIM0\_CH3\_OUT
- 1101: GPTIM0\_TRGO
- 1110: GPTIM2\_CH1\_OUT
- 1111: reserved

**Notes:**

1. Software is allowed to write this bit only when START=0 (in register [ADC\\_CR](#)).
2. If the TRGO signal of GPTIMx is used as the trigger source for the ADC conversion, the MMS bit in the GPTIM0\_CR2 and GPTIM1\_CR2 registers can only be configured as 0b100 (OC0REF), 0b101 OC1REF), 0b110 (OC2REF) or 0b111 (OC3REF). For GPTIM2\_CR2 and GPTIM3\_CR2, only 0b100 (OC0REF) or 0b101 (OC1REF) can be selected.
3. To achieve timed trigger or periodic trigger, you need to configure the selected channel as output mode, select the corresponding output mode, and configure the corresponding GPTIMx\_ARR and GPTIMx\_CCRx according to the required time.

**Bit 12 DMA\_EN:** DMA enable.

- 0: DMA disabled
- 1: DMA enabled

**Bits 11-0 CLK\_DIV:** ADCCLK prescale.

- 000: not divided
- 001: not divided
- n: ADC\_IP\_CLK=ADCCLK/n, 50% duty cycle

**Notes:**

1. This bit can only be configured when all bits of the [ADC\\_CR](#) register are 0; the clock source selection for ADCCLK is configured in the [RCC\\_CR2](#) register.
2. The clock division and clock source selection need to consider the data readout speed. The ADC samples every 16 ADC clock cycles, if a high-speed ADC clock source is chosen, the converted data cannot be read in time by the software or the DMA, which may cause overflow.

### 16.14.3 ADC\_SEQR0

Offset: 0x008

Reset Value: 0x00000000

**Note:** Software is allowed to configure this register only when START=0 and EN=0 (in [ADC\\_CR](#)).

31-28	27-24	23-20	19-16	15-12	11-8	7-4	3-0
SEL7	SEL6	SEL5	SEL4	SEL3	SEL2	SEL1	SEL0
rw-0h							

**Bits 31-28 SEL7:** Select the channel number from 1 to 15 as the 7th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 27-24 SEL6:** Select the channel number from 1 to 15 as the 6th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 23-20 SEL5:** Select the channel number from 1 to 15 as the 5th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 19-16 SEL4:** Select the channel number from 1 to 15 as the 4th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 15-12 SEL3:** Select the channel number from 1 to 15 as the 3th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 11-8 SEL2:** Select the channel number from 1 to 15 as the 2nd in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 7-4 SEL1:** Select the channel number from 1 to 15 as the 1st in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 3-0 SEL0:** Select the channel number from 1 to 15 as the 0th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

#### 16.14.4 ADC\_SEQR1

Offset: 0x00C

Reset Value: 0x00000000

**Note:** Software is allowed to configure this register only when START=0 and EN=0 (in [ADC\\_CR](#)).

31-28	27-24	23-20	19-16	15-12	11-8	7-4	3-0
SEL15	SEL14	SEL13	SEL12	SEL11	SEL10	SEL9	SEL8
rw-0h							

**Bits 31-28 SEL15:** Select the channel number from 1 to 15 as the 15th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 27-24 SEL14:** Select the channel number from 1 to 15 as the 14th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 23-20 SEL13:** Select the channel number from 1 to 15 as the 13th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 19-16 SEL12:** Select the channel number from 1 to 15 as the 12th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 15-12 SEL11:** Select the channel number from 1 to 15 as the 11th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 11-8 SEL10:** Select the channel number from 1 to 15 as the 10th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 7-4 SEL9:** Select the channel number from 1 to 15 as the 9th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

**Bits 3-0 SEL8:** Select the channel number from 1 to 15 as the 8th in the conversion sequence.

If 0 rather than 1 to 15 is configured, it marks the end of the sequence and itself cannot be converted. If the channel numbers selected by bits SELx are the same, the conversion of the same channel in a sequence will be performed multiple times.

In differential input mode, only the channel number of the positive input needs to be configured by software, and the channel number of the negative input is selected automatically by hardware according to register [ADC\\_DIFFSEL](#).

### 16.14.5 ADC\_DIFFSEL

Offset: 0x010

Reset Value: 0x00000000

**Note:** Software is allowed to configure this register only when START=0 and EN=0 (in [ADC\\_CR](#)).

31-16	15-9	8-1	0
RESERVED	SEL1	SEL0	RESERVED
r-0h	r-0h	rw-0h	r-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-9 SEL1:** Channels 9 to 15 are internal channels.

These channels can only be configured in single-ended mode. These bits are read-only.

**Bits 8-1 SEL0:** Differential or single-ended mode selection for Channels 1 to 8.

Each bit controls a channel with the same number as it:

- 0: channel x is configured in single-ended mode
- 1: channel x is configured in differential mode

In Differential mode, a group consists of two adjacent external channels, such as Group 1 consisting of Channel 2 and Channel 3, so the corresponding two control bits of this register should be set to 1 at the same time.

**Bit 0 RESERVED:** Must be kept, and cannot be modified.

### 16.14.6 ADC\_ISR

Offset: 0x014

Reset Value: 0x00000000

**Note:** It is recommended to clear all bits of this register before software sets the START bit (in register [ADC\\_CR](#)).

31-3	2	1	0
RESERVED	OVERRUN	EOS	EOC
r-0h	rw1c-0h	rw1c-0h	rw1c-0h

**Bits 31-3 RESERVED:** Must be kept, and cannot be modified.

**Bit 2 OVERRUN:** ADC conversion overrun flag.

- 0: no overrun
- 1: overrun occurred

This bit is set by hardware when an overrun occurs and a new conversion is completed when the EOC flag was already set, but the [ADC\\_DR](#) register has not been read or software writing 1 to clear this bit was not configured.

It is cleared by software writing 1 to it.

**Bit 1 EOS:** End of sequence of conversions flag.

- 0: conversion sequence in progress
- 1: conversion sequence completed

This bit is set by hardware when a sequence of conversions (set by [ADC\\_SEQR0/1](#)) is completed.

It is cleared by software writing 1 to it.

**Bit 0 EOC:** End of conversion flag.

- 0: channel conversion in progress
- 1: channel conversion completed

This flag is set by hardware at the end of each conversion of a channel (when the newly converted data is stored in the *ADC\_DR* register).

It is cleared by software writing 1 to it or by reading the *ADC\_DR* register.

### 16.14.7 ADC\_IER

Offset: 0x018

Reset Value: 0x00000000

31-3	2	1	0
RESERVED	OVERRUN_INT_EN	EOS_INT_EN	EOC_INT_EN
r-0h	rw-0h	rw-0h	rw-0h

**Bits 31-3 RESERVED:** Must be kept, and cannot be modified.

**Bit 2 OVERRUN\_INT\_EN:** ADC conversion overrun interrupt enable.

- 0: overrun interrupt disabled
- 1: overrun interrupt enabled

**Bit 1 EOS\_INT\_EN:** End of conversion sequence interrupt enable.

- 0: end of conversion sequence interrupt disabled
- 1: end of conversion sequence interrupt enabled

**Bit 0 EOC\_INT\_EN:** End of conversion interrupt enable.

- 0: end of conversion interrupt disabled
- 1: end of conversion interrupt enabled

### 16.14.8 ADC\_DR

Offset: 0x01C

Reset Value: 0x00000000

31-12	11-0
RESERVED	DATA
r-0h	r-0h

**Bits 31-12 RESERVED:** Must be kept, and cannot be modified.

**Bits 11-0 DATA:** ADC converted data. In differential mode, bit[11] is the sign bit.

# 17.

# Basic timer (BSTIM)

## 17.1 Introduction

BSTIMER (Basic Timer) contains a 16bits counter, supports auto-reloading function, and supports up to 16-bits programmable frequency division counter. There are two BSTIMERs, named BSTIMER0 and BSTIMER1.

## 17.2 Main features

BSTIMER includes the following functions:

- 16bits counter, up-counting, auto-reloading
- Prescaler
- DMA control
- Single pulse mode
- Master mode
- Update event management
- Debug mode control
- Interrupt signal generation

BSTIMER structure diagram:

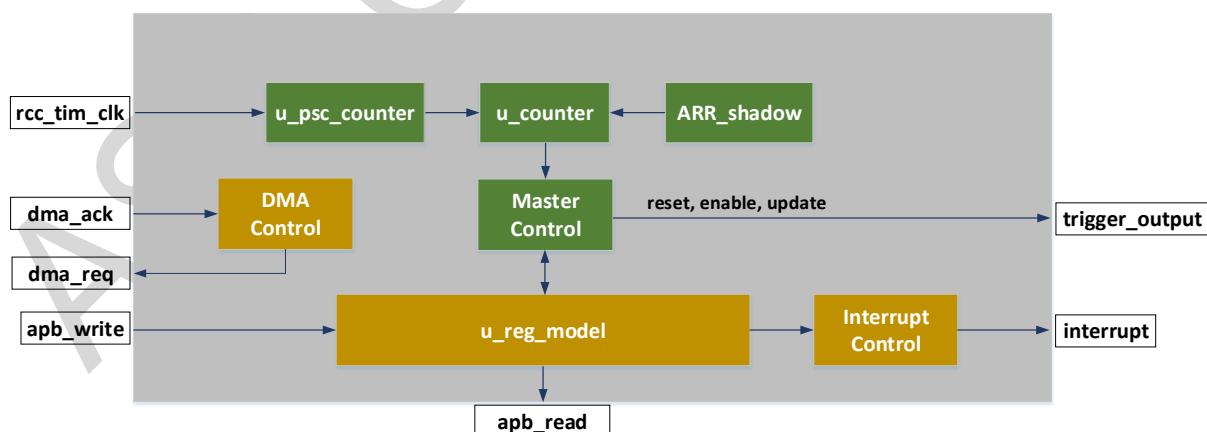


Figure 17-1 BSTIMER Diagram

- **rcc\_tim\_clk**: BSTIMER clock
- **dma\_ack**: DMA ACK
- **dma\_req**: BSTIMER DMA Request
- **apb\_write**: APB bus write
- **apb\_read**: APB bus read
- **trigger\_output**: **trigger\_output**: BSTIMER TRGO output
- **interrupt**: BSTIMER interrupt

## 17.3 Clock source

The clock source of the BSTIMER interface is PCLK and cannot be set to other clock sources. Please refer to the RCC chapter for clock enable and complex bits configuration.

## 17.4 Counter

The counter only supports upward counting. When counting to ARR, the counter value will change from ARR to 0, and then continue counting. At the same time, the status flag UIF is set. If the update event interrupt request is enabled, that is, UIE is set, an interrupt will also be generated, indicating that a counting cycle is completed. In the next counting cycle, the counter continues to count from 0, and so on.

## 17.5 Auto-reload

The ARPE Bits of Register BSTIM\_CR1 can be configured by software to set whether to enable the ARR Shadow Register. If ARPE=0, the Shadow Register is disabled. The value written by the software is directly updated to the ARR for counter use. If ARPE=1, the value written by the software is the value will not take effect immediately, until the update event arrives, the value will be updated to ARR for counter use.

## 17.6 Prescaler

BSTIMER supports 16-bit (1~65535) programmable prescaler, which is implemented by the frequency division counter BSTIM\_PSC. The interface clock is used as the clock of the prescaler, and the CEN of the register BSTIM\_CR1 is used as the count enable of the prescaler. When the prescaler counts to the pre-loaded frequency division value, it outputs a pulse as the count enable of the next level counter, and then the prescaler returns to zero and counts again, and so on.

The division value of the division counter uses the shadow register by default, that is, the software write operation will not take effect immediately, but the new division value will be written into the shadow register until the update event (UG event is set, count overflow) arrives, at which time the division value will officially take effect. The software read operation reads the written register value, not the shadow register. If there are multiple write operations before the update event arrives, the previously written value will be overwritten. The counting and division waveforms are as follows:

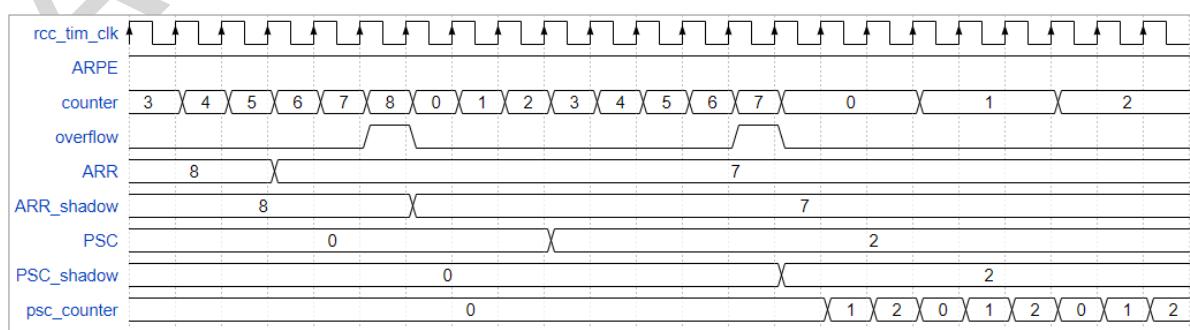


Figure 17-2 Counting and Dividing Waveforms

## 17.7 DMA

BSTIMER supports the DMA function. After enabling the DMA function, all its Registers except BSTIM\_SR and BSTIM\_EGR can transfer data to each other and memory. BSTIM\_SR can only read data, and BSTIM\_EGR can only write data. Enable DMA through UDE Bits of Register BSTIM\_DIER , when there is an update event, a DMA Request will be generated. DMA 返回的 The ACK signal returned by DMA clears the module's DMA Request signal.

## 17.8 Single pulse mode

BSTIMER supports single pulse counting mode, which can be enabled by setting the OPM bit of register BSTIM\_CR1. In this mode, when the counter counts to the ARR value, it will return to zero and stop counting (CEN hardware automatically clears to zero), and will not count again unless it is initialized again, as shown in the following figure:

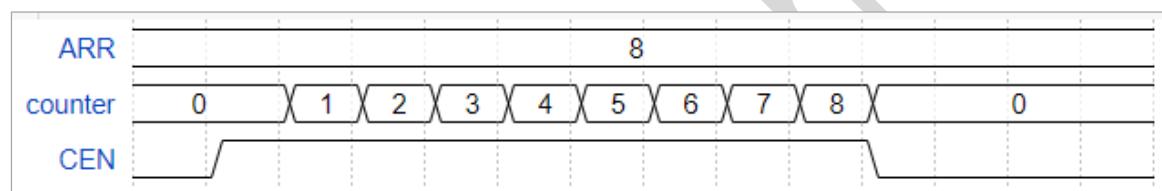


Figure 17-3 Single pulse waveform

## 17.9 Mode selection

BSTIMER can be cascaded with other internal modules and used as a host to control the DAC by generating a trigger output signal (TRGO). The source of the TRGO signal can be selected by software configuration of the MMS bit of the BSTIM\_CR2 register, as follows:

- MMS=3'b000 : Reset mode, in which case the UG flag will be output as a TRGO signal to the external slave.
- MMS=3'b001 : Enable mode, in which case the counter's count enable CEN will be output as the TRGO signal to the external slave.
- MMS=3'b010 : Update mode, in which case the update event is output to the external slave as the TRGO signal.
- Other values of MMS are reserved.

## 17.10 Update event management

Update events include the following event sources:

1. Counter overflow event (overflow), that is, the value of counter changes from ARR to 0.
2. UG Set Bits (software Set Bits), that is, configure the UG Bits of Register BSTIM\_EGR.

The control signals related to update event management are URS and UDIS in BSTIM\_CR1 register . The specific controls are as follows:

- If UDIS=0, URS=0, the overflow and UG bits setting will initialize the counter and prescaler. If the shadow register is enabled, the update event will update the written value to the shadow register (ARR depends on ARPE). UIF will set. If interrupt or DMA is enabled, interrupt or DMA Request will be generated.
- If UDIS=0, URS=1, the overflow and UG bits setting will initialize the counter and prescaler. If the shadow register is enabled, the update event will update the written value to the shadow register (ARR depends on ARPE). UIF will only set Bits in the case of overflow. If interrupt or DMA is enabled, interrupt or DMA Request will be generated.
- If UDIS=1 (URS ignored), only UG bit setting will still initialize counter and prescaler, but the shadow register will not be updated, and UIF will not set, so no corresponding interrupt or DMA request will be generated.

## 17.11 Debug mode control

BSTIMER can be configured by software whether to stop counting under debug. The DEBUG mode counting control of BSTIMER0 and BSTIMER1 is implemented through the CR2 Register of SYSCFG. If this function is enabled, BSTIMER will stop counting when entering the system debug mode (the counter will not be initialized).

## 17.12 Interrupts

The interrupt signal of BSTIMER is as follows:

**Table 17-1 BSTIMER interrupts**

Interrupt	Description
Update event interrupt	Counter overflow and UG bit setting can generate update event interrupt.

The interrupts above are enabled by configuring the UIE bit in the BSTIM\_DIER register, and the interrupt status can be obtained through the BSTIM\_SR register.

## 17.13 BSTIMER registers

BSTIMER0 Base Address: 0x4000C000

BSTIMER1 Base Address: 0x4001C000

**Table 17-2 BSTIMER Registers Summary**

Register	Offset	Description
BSTIM_CR1	0x000	Control Register 1
BSTIM_CR2	0x004	Control Register 2
BSTIM_DIER	0x00C	DMA/Interrupt Enable Register
BSTIM_SR	0x010	Status Register
BSTIM_EGR	0x014	Event Register
BSTIM_CNT	0x024	Counter Register
BSTIM_PSC	0x028	Prescaler Register
BSTIM_ARR	0x02C	Auto-Reload Register

### 17.13.1 BSTIM\_CR1

Offset: 0x000

Reset Value: 0x00000000

31-8	7	6-4	3	2	1	0
RESERVED	ARPE	RESERVED	OPM	URS	UDIS	CEN
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bit 7 ARPE:** Reload shadow Register enable.

- 0: BSTIM\_ARR Shadow Register disabled
- 1: BSTIM\_ARR Shadow Register enabled

**Bits 6-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 OPM:** Single pulse mode enable.

- 0: single pulse mode disabled
- 1: single pulse mode Enabled, counter stops counting at the next update event

**Bit 2 URS:** Update event source selection, this bit only affects the interrupt (UIF) and DMA flag bits, and does not affect the internal logic.

- 0: Counter overflow and UG bit can set UIF bit
- 1: Only counter overflow events can set UIF bit

**Bit 1 UDIS:** Update events are disable.

- 0: Update event enabled, update events can be generated.
- 1: The update event is disabled, the shadow Register and UIF will not be updated, but at this time the counter and prescaler can still be initialized by the UG bit Set event.

**Bit 0 CEN:** Counter enable, CEN is cleared by hardware in single pulse mode.

- 0: Counter disabled
- 1: Counter enabled

### 17.13.2 BSTIM\_CR2

Offset: 0x004

Reset Value: 0x00000000

31-7	6-4	3-0
RESERVED	MMS	RESERVED
rw-0h	rw-0h	rw-0h

**Bits 31-7 RESERVED:** Must be kept, and cannot be modified.

**Bits 6-4 MMS:** Master mode selection, TRGO output can be configured.

- 000: Reset mode, UG will be output as TRGO signal
- 001: Enable mode, CEN will be output as TRGO signal
- 010: Update mode, update events (internal signals) will be output as TRGO signals
- Other values: Reserved

**Bits 3-0 RESERVED:** Must be kept, and cannot be modified.

### 17.13.3 BSTIM\_DIER

Offset: 0x00C

Reset Value: 0x00000000

31-9	8	7-1	0
RESERVED	UDE	RESERVED	UIE
rw-0h	rw-0h	rw-0h	rw-0h

**Bits 31-9 RESERVED:** Must be kept, and cannot be modified.

**Bit 8 UDE:** Update event DMA Request Enable.

- 0: disable update event DMA Request
- 1: enable update event DMA Request

**Bit 0 UIE:** Update event Interrupt Request enable.

**Bit 0 UIE:** Update event Interrupt Request enable.

- 0: Update event Interrupt Request disabled
- 1: Update event Interrupt Request enabled

#### 17.13.4 BSTIM\_SR

Offset: 0x010

Reset Value: 0x00000000

31-1	0
RESERVED	UIF
r-0h	r-0h

**Bits 31-1 RESERVED:** Must be kept, and cannot be modified.

**Bit 0 UIF:** Update event flag.

- 0: no update event
- 1: update event occurs

#### 17.13.5 BSTIM\_EGR

Offset: 0x014

Reset Value: 0x00000000

31-1	0
RESERVED	UG
w-0h	w-0h

**Bits 31-1 RESERVED:** Must be kept, and cannot be modified.

**Bit 0 UG:** Update event generation enable.

- 0: update event generation disabled
- 1: update event generation enabled

#### 17.13.6 BSTIM\_CNT

Offset: 0x024

Reset Value: 0x00000000

31-16	15-0
RESERVED	CNT
rw-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 CNT:** Counter count value.

### 17.13.7 BSTIM\_PSC

Offset: 0x028

Reset Value: 0x00000000

31-16	15-0
RESERVED	PSC
rw-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 PSC:** Clock division value is PSC+1.

### 17.13.8 BSTIM\_ARR

Offset: 0x02C

Reset Value: 0x0000FFFF

31-16	15-0
RESERVED	ARR
rw-0h	rw-ffffh

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 ARR:** Counter auto-reload value.

# 18.

# RTC

## 18.1 Introduction

The Real-time Clock is an independent BCD timer/counter. It has two 32-bit registers, which contain the seconds, minutes, hours (12-hour or 24-hour format), day of week, date of month, month, and year, expressed in binary coded decimal format (BCD). In addition, there is a 32-bit register used to indicate sub-seconds value. The RTC supports operation under low-power mode.

## 18.2 Main Features

The features of RTC are as follows:

- Calendar with the seconds, minutes, hours (12-hour or 24-hour format), day of week, date of month, month, and year, expressed in binary coded decimal format (BCD).
- RTC frequency calibration with a resolution of about 0.5 ppm with a range from -1024 ppm to +1024 ppm.
- Wake-up from low-power modes.
- Tamper/wakeup IO detection at high or low level with configurable filter.
- 32-bit counter for periodic count.
- 2 Alarms, support calendar matching.
- Clear retention SRAM on tamper/wakeup alarm.
- Internal signal output by GPIO, including Alarm0 pulse, Alarm1 pulse, periodic counter pulse and second signal.
- Calendar values reading.
- Sub-seconds value reading.
- Period counting value reading.
- Interrupt signal generation.

## 18.3 Interface Clock

Both XO32K and RCO32K can be RTC clock source and XO32K accuracy is higher than RCO32K.

See [RCC](#) Chapter for clock configuration details.

## 18.4 Calendar

The RTC calendar time and date are accessed through two types of registers, which are the asynchronous registers and the synchronous registers.

- *RTC\_SYNCDATA* indicates the seconds, minutes and hours; *RTC\_SYNCDATA\_H* indicates the day of week, date of month, month and year.
- *RTC\_CALENDAR\_R* and *RTC\_CALENDAR\_R\_H* are **synchronous registers**.

*RTC\_CALENDAR\_R* indicates the seconds, minutes and hours; *RTC\_CALENDAR\_R\_H* indicates the day of week, date of month, month and year.

### 18.4.1 Calendar reading

This document only introduces reading the RTC calendar values by **synchronous registers**. The synchronous registers should be read several times with the same result obtained to ensure that the data is correct. Follow below steps to read the calendar:

- (1) Read the *RTC\_SUB\_SECOND* register to get the subsecond\_count value.
- (2) Read the value of the *RTC\_CALENDAR\_R* register for two consecutive times, if the values read are different, then continue reading until the values read for two consecutive times are the same.
- (3) Read the value of the *RTC\_CALENDAR\_R\_H* register for two consecutive times, if the values read are different, then continue reading until the values read for two consecutive times are the same.
- (4) Read the value of the *RTC\_SUB\_SECOND* register again, if the value is not equal to the value in Step 1, then software will restart reading the calendar from Step 1.
- (5) When the subseconds downcounter reaches 0, the value of the *RTC\_CALENDAR\_R* or *RTC\_CALENDAR\_R\_H* register may have no change, so if the subsecond\_count value is 0, then the software will restart reading the calendar from Step 1; if subsecond\_count value is not 0, then it indicates that the complete calendar time has been read correctly.

For converting subsecond\_count to sub-second (unit: microsecond), first obtain the frequency of the RTC interface clock (fRTCCLK) through the *RTC\_CLK\_SEL* bit in the *RCC\_CR1* register, then use the formula below to calculate the sub-second:

$$\text{sub-second} = (1000000 * \text{SUBSECOND\_COUNT}) / \text{fRTCCLK}$$

### 18.4.2 Calendar setting

The *RTC\_CALENDAR\_H* and *RTC\_CALENDAR* registers are used to set the calendar. *RTC\_CALENDAR\_H* sets the year, month, date of month and day of week. *RTC\_CALENDAR* sets the hours, minutes and seconds. Since the *RTC\_SUB\_SECOND* register is read-only, the subsecond is read-only. Follow below steps to set the calendar:

- (1) Read the *RTC\_SR1* register, and wait for all *WRITE\_XXX\_DONE* bits and the *SECOND\_SR* bit (bits[11:1]) to be set. After that, writing to the *RTC\_CALENDAR\_H* and *RTC\_CALENDAR* registers is allowed.
- (2) Configure the year, month, date of month and day of week in the *RTC\_CALENDAR\_H* register.

- (3) Read the *RTC\_SR1* register, and wait for all WRITE\_XXX\_DONE bits and the SECOND\_SR bit (bits[11:1]) to be set. After that, writing to the *RTC\_CALENDAR\_H* and *RTC\_CALENDAR* registers is allowed.
- (4) Configure the hours, minutes and seconds in the *RTC\_CALENDAR* register.

## 18.5 RTC PPM Calibration

The RTC frequency can be calibrated with a resolution of about 0.5 ppm with a range from -1024 ppm to +1024 ppm. Configure register *RTC\_PPMADJUST* to set the adjustment value. When the value is set to 0x7FFF, it means to adjust 0 ppm, that is, no adjustment is required. Follow below steps to conduct the PPM calibration:

- (1) Read the *RTC\_SR1* register, and wait for all WRITE\_XXX\_DONE bits and the SECOND\_SR bit (bits[11:1]) to be set. After that, writing to the *RTC\_PPMADJUST* register is allowed.
- (2) Configure the adjustment value in the *RTC\_PPMADJUST* register.

## 18.6 Wake-up from Low-power Mode

RTC can wake up the MCU from Sleep, Stop or Standby mode through an interrupt or wakeup signal.

**Table 18-1 RTC Wakeup Source**

Mode	Description
Sleep	RTC interrupts can wake up the device from the Sleep mode.
Stop0/Stop1/Stop2/Stop3	RTC wakeup, RTC tamper event, RTC alarm, and periodic count signal can wake up the device from Stop mode.
Standby	RTC wakeup, RTC tamper event, RTC alarm, and periodic count signal can wake up the device from Standby mode.

Enable the wakeup/tamper IO, RTC alarm and periodic count signal for wake-up through the corresponding bit in register *RTC\_CR* :

**Table 18-2 Bits to Enable Wake-up Signals**

Function	Bit in Register RTC_CR
WAKEUP_IO0	WAKEUP0_WKEN1
WAKEUP_IO1	WAKEUP1_WKEN1
WAKEUP_IO2	WAKEUP2_WKEN1
TAMPER	TAMPER_WKEN1
ALARM0	RTC_ALARM0_WKEN
ALARM1	RTC_ALARM1_WKEN
CYC	CYC_WKEN

## 18.7 Tamper/Wakeup IO Detection

The tamper/wakeup IO input events can be configured for edge detection or level detection with filtering. Edge detection means to detect the rising or falling edge of GPIO, while level detection means to detect the high or low level of GPIO. If GPIO is active at high level, a tamper detection event is generated when a high level is detected on GPIO input; if GPIO is active at low level, a tamper detection event is generated when a high level is detected on GPIO input. When an input event is detected, the following actions can be conducted:

- Erase the retention SRAM.
- Generate an interrupt, capable to wakeup from Sleep mode.
- Generate a wakeup signal (WAKEUP\_IO0/WAKEUP\_IO1/WAKEUP\_IO2/TAMPER), capable to wakeup from Stop and Standby modes.

### 18.7.1 Tamper/Wakeup Initialization and Configuration

Before Tamper/Wakeup initialization, the corresponding GPIO should be configured as tamper/wakeup function. In addition, if it is level detection, GPIO should be configured with pull-up or pull-down. If GPIO is active at high level, pull down GPIO; if GPIO is active at low level, pull up GPIO.

Taking Tamper as an example, the initialization and configuration process is as follows:

- (1) If it is level detection, set the filter length by configuring bit TAMPER\_FILTER\_CFG in the [RTC\\_CR](#) register, configure the active level by bit TAMPER\_LEVEL\_SEL, and then enable the tamper pin level wakeup by the TAMPER\_WKEN0 bit. **If it is edge detection, ignore this step.**
- (2) Configure bit TAMPER\_WKEN1 in register [RTC\\_CR](#) to enable TAMPER\_SR to wake up the MCU from Stop or Standby mode. **If no such need, ignore this step.**
- (3) Set the TAMPER\_EN bit in the [RTC\\_CR](#) register to enable tamper detection.

### 18.7.2 Retention SRAM Erase Operation

When the tamper/wakeup IO input event is detected, the hardware can erase the retention SRAM. This is configured by setting the corresponding bit of RTC\_RET\_SRAM\_ERASE\_EN in register [RTC\\_CR2](#). Bit0 corresponds to wakeup IO0, bit1 corresponds to wakeup IO1, bit2 corresponds to wakeup IO2, and bit3 corresponds to tamper function.

## 18.8 Periodic Counter

The periodic counter generates interrupts or wakeup events at regular intervals. The regular interval is set according to the configured CYC\_MAX\_VALUE in the [RTC\\_CYC\\_MAX\\_VALUE](#) register. Obtain the RTC interface clock frequency (fRTCCLK) through the RTC\_CLK\_SEL bit in the [RCC\\_CR1](#) register, and then use the formula below to calculate the regular interval (in microseconds):

$$\text{Regular interval} = (1000000 * \text{CYC\_MAX\_VALUE}) / \text{fRTCCLK}$$

During the periodic count, the number of elapsed cycles is read from the CYC\_CNT\_VALUE bits in the [RTC\\_CYC\\_CNT\\_VALUE](#) register. On this basis, the interval (in microseconds) from the start of the ongoing counting to the current moment can be calculated by the formula below:

$$\text{Interval} = (1000000 * \text{CYC_CNT\_VALUE}) / f_{\text{RTCCLK}}$$

Follow below steps to configure the periodic count:

- (1) When the regular interval is known, calculate the CYC\_MAX\_VALUE according to the above formula, and configure this value to register [RTC\\_CYC\\_MAX\\_VALUE](#).
- (2) Configure bit CYC\_WKEN in register [RTC\\_CR](#) to enable CYC\_SR to wake up CPU from Stop or Standby mode. **If no such need, ignore this step.**
- (3) Set bit CYC\_START\_COUNTER in register [RTC\\_CR](#) to enable periodic counter.

## 18.9 RTC Alarms

RTC provides two alarms: Alarm 0 and Alarm 1. Both support mask selection and matching with calendar. With Mask configuration, each calendar field (sub-seconds, seconds, minutes, hours, date or day of week) can be independently selected to match the values programmed in the alarm registers. Note that for the date and the day of week, we can only choose one of them for the match.

If bit ALARMx\_WEEK\_SEL (Alarmx means Alarm 0 or Alarm 1, similarly hereinafter) is 0 in the register [RTC\\_ALARMx](#), the date is selected for the match; if bit ALARMx\_WEEK\_SEL is 1, the day of week is selected for the match.

If the sub-seconds and seconds are not involved but the minutes are involved in Alarmx comparison, when Alarmx values match with those of the RTC Calendar, 60 interrupts or/and 60 wake-up events are generated at a one-second interval in one minute. If the sub-seconds, seconds, and minutes are not involved but the hours are involved in Alarmx comparison, when Alarmx values match with those of the RTC Calendar, 3600 interrupts or/and 3600 wake-up events are generated at a one-second interval in one hour. Whether the interrupts or/and wake-up events are generated depends on whether the alarm interrupt or/and the alarm wake-up is enabled.

The seconds, minutes, hours, date or day mask are configured through the ALARMx\_MASK bit field in the [RTC\\_ALARMx](#) register, and the sub-seconds mask is configured through the [RTC\\_ALARMx\\_SUB\\_MASK](#) bit field in the [RTC\\_ALARMx\\_SUB](#) register. The sub-seconds value is set by the [RTC\\_ALARMx\\_SUB\\_VALUE](#) bit field in the [RTC\\_ALARMx\\_SUB](#) register.

The [RTC\\_ALARMx\\_SUB\\_VALUE](#) indicates RTC clock cycles, and the formula for converting clock cycles to time is the same as that in periodic count.

Take Alarm 0 as an example to describe the alarm configuration process as follows:

- (1) Set the calendar.
- (2) Configure the Alarm 0 values (including the hours, minutes, seconds, date or day) by the [ALARM0\\_VALUE](#) bit field in the [RTC\\_ALARM0](#) register.
- (3) Configure the sub-seconds value for Alarm 0 through the [RTC\\_ALARM0\\_SUB\\_VALUE](#) bit field in the [RTC\\_ALARM0\\_SUB](#) register.
- (4) Configure the seconds, minutes, hours, date or day mask for Alarm 0.
- (5) Configure the sub-seconds mask for Alarm 0.
- (6) Whether the [ALARM0\\_SR](#) interrupt or [ALARM0\\_SR](#) wake-up is enabled depends on the specific needs. They are configured through the [ALARM0\\_SR\\_INT\\_EN](#) bit in register [RTC\\_CR1](#) and the [RTC\\_ALARM0\\_WKEN](#) bit in register [RTC\\_CR](#).
- (7) Enable the Alarm 0 through the [ALARM0\\_EN](#) bit in register [RTC\\_ALARM0](#).
- (8) Enable the calendar by setting the [RTC\\_START\\_RTC](#) bit in register [RTC\\_CR](#).

## 18.10 Internal Signal Output through IO

The internal signals that can be output through IO include Alarm 0 pulse, Alarm 1 pulse, periodic counter pulse, and the second signal. The alarm pulse and periodic counter pulse are pulses with a width of one RTC clock cycle. The Alarm pulse is output when the programmed values match with the Calendar. The periodic counter pulse is output every time the programmed count value is reached. The second signal is a square wave with a duty cycle of 50% and the frequency is 1 Hz. The RTC IO can output inverted levels. When the RTC\_OUT\_POL bit of the *RTC\_CR2* register is 0, it means that the level is non-inverted, and when this bit is 1, it means that the level is inverted. Configure the RTC\_OUT\_SEL bit of the *RTC\_CR2* register to select the RTC IO output signal.

## 18.11 Interrupts

The interrupt signals of RTC are as follows:

**Table 18-3 RTC Interrupts**

Interrupt	Description
Alarm 0 interrupt	Interrupt is generated when the interval set by Alarm 0 is reached.
Alarm 1 interrupt	Interrupt is generated when the interval set by Alarm 1 is reached.
Periodic wakeup interrupt	Interrupt is generated at regular intervals.
Tamper interrupt	Interrupt is generated when an input event is detected by tamper IO.
IO0 Wakeup interrupt	Interrupt is generated when an input event is detected by Wakeup IO0 IO.
IO1 Wakeup interrupt	Interrupt is generated when an input event is detected by Wakeup IO1 IO.
IO2 Wakeup interrupt	Interrupt is generated when an input event is detected by Wakeup IO2 IO.
Second signal interrupt	Interrupt is generated by the second signal every second.

The above interrupts are enabled by configuring the *RTC\_CR1* register. The second signal interrupt status is indicated by the SECOND\_SR bit in the *RTC\_SR1* register, and the other interrupts' status is indicated by the *RTC\_SR* register.

## 18.12 RTC Registers

Base Address: 0x4000E000

**Table 18-4 RTC Registers Summary**

Register	Offset	Description
RTC_CR	0x000	RTC Control Register 1
RTC_ALARM0	0x004	RTC Alarm 0 Register
RTC_ALARM1	0x008	RTC Alarm 1 Register
RTC_PPMADJUST	0x00C	RTC PPMADJUST Register
RTC_CALENDAR	0x010	RTC Calendar Configuration Register (second, minute, hour)
RTC_CALENDAR_H	0x014	RTC Calendar Configuration Register (date/day of week, month, year)
RTC_CYC_MAX_VALUE	0x018	RTC Periodic Counter Value Configuration Register
RTC_SR	0x01C	RTC Status Register
RTC_ASYNCDATA	0x020	RTC Calendar ASYNDATA Register (second, minute, hour)
RTC_ASYNCDATA_H	0x024	RTC Calendar ASYNDATA Register (date/day of week, month, year)
RTC_CR1	0x028	RTC Control register 1 (interrupt enable)
RTC_SR1	0x02C	RTC Status Register 1
RTC_CR2	0x030	RTC Control Register 2
RTC_SUB_SECOND	0x034	RTC Sub-second Register
RTC_CYC_CNT_VALUE	0x038	RTC Periodic Counter Value Register (read-only)
RTC_ALARM0_SUB	0x03C	RTC Alarm 0 Sub-second Register
RTC_ALARM1_SUB	0x040	RTC Alarm 1 Sub-second Register
RTC_CALENDAR_R	0x044	RTC Calendar SYNDATA Register (second, minute, hour)
RTC_CALENDAR_R_H	0x048	RTC Calendar SYNDATA Register (date/day of week, month, year)

### 18.12.1 RTC\_CR

Offset: 0x000

Reset Value: 0x00000000

31-29	28	27	26	25
RESERVED	RTC_START_RT C	RTC_ALARM0_W KEN	RTC_ALARM1_W KEN	CYC_WKEN
r-0h	rw-0h	rw-0h	rw-0h	rw-0h
24	23	22	21	20
CYC_START_CO UNTER	TAMPER_EN	TAMPER_LEVEL_ SEL	TAMPER_WKEN0	TAMPER_WKEN1
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h
19-18	17	16	15	14
TAMPER_FILTER _CFG	WAKEUP0_EN	WAKEUP0_LEVE L_SEL	WAKEUP0_WKE N0	WAKEUP0_WKE N1
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h
13-12	11	10	9	8
WAKEUP0_FILTE R_CFG	WAKEUP1_EN	WAKEUP1_LEVE L_SEL	WAKEUP1_WKE N0	WAKEUP1_WKE N1
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h
7-6	5	4	3	2
WAKEUP1_FI LTER_CFG	WAKEUP2_E N	WAKEUP2_LE VEL_SEL	WAKEUP2_W KEN0	WAKEUP2_W KEN1
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h
1-0				

**Bits 31-29 RESERVED:** Must be kept, and cannot be modified.

**Bit 28 RTC\_START\_RTC:** RTC calendar enable.

- 0: disabled
- 1: enabled

**Bit 27 RTC\_ALARM0\_WKEN:** ALARM0\_SR wake-up enable.

- 0: disabled
- 1: enabled

**Bit 26 RTC\_ALARM1\_WKEN:** ALARM1\_SR wake-up enable.

- 0: disabled
- 1: enabled

**Bit 25 CYC\_WKEN:** CYC\_SR wake-up enable.

- 0: disabled
- 1: enabled

**Bit 24 CYC\_START\_COUNTER:** Periodic counter enable.

- 0: disabled
- 1: enabled

**Bit 23 TAMPER\_EN:** Tamper enable.

- 0: disabled
- 1: enabled

**Bit 22 TAMPER\_LEVEL\_SEL:** Tamper active level selection.

- 0: low level
- 1: high level

**Bit 21 TAMPER\_WKEN0:** Tamper level wake-up enable.

- 0: disabled
- 1: enabled

When TAMPER\_EN is set to 0, this configuration is still valid.

**Bit 20 TAMPER\_WKEN1:** TAMPER\_SR wake-up enable.

- 0: disabled
- 1: enabled

**Bits 19-18 TAMPER\_FILTER\_CFG:** Tamper filter control.

- 0: none
- 1: filter length is 1 RTC interface clock cycle
- 2: filter length is 3 RTC interface clock cycles
- 3: filter length is 7 RTC interface clock cycles

**Bit 17 WAKEUP0\_EN:** WAKEUP0 enable.

- 0: disabled
- 1: enabled

**Bit 16 WAKEUP0\_LEVEL\_SEL:** WAKEUP0 active level selection.

- 0: low level
- 1: high level

**Bit 15 WAKEUP0\_WKEN0:** WAKEUP0 level wake-up enable.

- 0: disabled
- 1: enabled

When WAKEUP0\_EN is set to 0, this configuration is still valid.

**Bit 14 WAKEUP0\_WKEN1:** WAKEUP0\_SR wake-up enable.

- 0: disabled
- 1: enabled

**Bits 13-12 WAKEUP0\_FILTER\_CFG:** WAKEUP0 filter control.

- 0: none
- 1: filter length is 1 RTC interface clock cycle
- 2: filter length is 3 RTC interface clock cycles
- 3: filter length is 7 RTC interface clock cycles

**Bit 11 WAKEUP1\_EN:** WAKEUP1 enable.

- 0: disabled
- 1: enabled

**Bit 10 WAKEUP1\_LEVEL\_SEL:** WAKEUP1 active level selection.

- 0: low level
- 1: high level

**Bit 9 WAKEUP1\_WKEN0:** WAKEUP1 level wake-up enable.

- 0: disabled
- 1: enabled

When WAKEUP1\_EN is set to 0, this configuration is still valid.

**Bit 8 WAKEUP1\_WKEN1:** WAKEUP1\_SR wake-up enable.

- 0: disabled
- 1: enabled

**Bits 7-6 WAKEUP1\_FILTER\_CFG:** WAKEUP1 filter control.

- 0: none
- 1: filter length is 1 RTC interface clock cycle
- 2: filter length is 3 RTC interface clock cycles
- 3: filter length is 7 RTC interface clock cycles

**Bit 5 WAKEUP2\_EN:** WAKEUP2 enable.

- 0: disabled
- 1: enabled

**Bit 4 WAKEUP2\_LEVEL\_SEL:** WAKEUP2 active level selection.

- 0: low level
- 1: high level

**Bit 3 WAKEUP2\_WKEN0:** WAKEUP2 level wake-up enable.

- 0: disabled
- 1: enabled

When WAKEUP2\_EN is set to 0, this configuration is still valid.

**Bit 2 WAKEUP2\_WKEN1:** WAKEUP2\_SR wake-up enable.

- 0: disabled
- 1: enabled

**Bits 1-0 WAKEUP2\_FILTER\_CFG:** WAKEUP2 filter control.

- 0: none
- 1: filter length is 1 RTC interface clock cycle
- 2: filter length is 3 RTC interface clock cycles
- 3: filter length is 7 RTC interface clock cycles

## 18.12.2 RTC\_ALARM0

Offset: 0x004

Reset Value: 0x00000000

31	30	29-26	25-0
ALARM0_EN	ALARM0_WEEK_SEL	ALARM0_MASK	ALARM0_VALUE
rw-0h	rw-0h	rw-0h	rw-0h

**Bit 31 ALARM0\_EN:** Alarm 0 enable.

- 0: disabled
- 1: enabled

**Bit 30 ALARM0\_WEEK\_SEL:** Date or day of week selection for Alarm 0.

- 0: match the date
- 1: match the day of week

**Bits 29-26 ALARM0\_MASK:** Alarm 0 mask configuration.

[26] Alarm 0 seconds mask

- 0: match the seconds
- 1: seconds are not involved in Alarm 0 comparison

[27] Alarm 0 minutes mask

- 0: match the minutes
- 1: minutes are not involved in Alarm 0 comparison

[28] Alarm 0 hours mask

- 0: match the hours
- 1: hours are not involved in Alarm 0 comparison

[29] Alarm 0 date or day of week mask

- 0: match the date or day of week
- date or day of week is not involved in Alarm 0 comparison

**Bits 25-0 ALARM0\_VALUE:** Alarm 0 value configuration. When the calendar sub-seconds, seconds, minutes, hours, date or day of week match the values programmed in this register and the [RTC\\_ALARM0\\_SUB](#) register, the ALARM0\_SR bit is set.

[3:0]: second units

[6:4]: second tens

[10:7]: minute units

[13:11]: minute tens

[17:14]: hour units

[19:18]: hour tens

[23:20]: bits[23:20] configure date units or bits[22:20] configure day of week

[25:24]: date tens

### 18.12.3 RTC\_ALARM1

Offset: 0x008

Reset Value: 0x00000000

31	30	29-26	25-0
ALARM1_EN	ALARM1_WEEK_SEL	ALARM1_MASK	ALARM1_VALUE
rw-0h	rw-0h	rw-0h	rw-0h

**Bit 31 ALARM1\_EN:** Alarm 1 enable.

- 0: disabled
- 1: enabled

**Bit 30 ALARM1\_WEEK\_SEL:** Date or day of week selection for Alarm 1.

- 0: match the date
- 1: match the day of week

**Bits 29-26 ALARM1\_MASK:** Alarm 1 mask configuration.

[26] Alarm 1 seconds  
mask

- 0: match the seconds
- 1: seconds are not involved in Alarm 1 comparison.

[27] Alarm 1 minutes mask

- 0: match the minutes
- 1: minutes are not involved in Alarm 1 comparison.

[28] Alarm 1 hours mask

- 0: match the hours
- 1: hours are not involved in Alarm 1 comparison.

[29] Alarm 1 date or day of week mask

- 0: match the date or day of week
- 1: date or day of week is not involved in Alarm 1 comparison

**Bits 25:0 ALARM1\_VALUE:** Alarm 1 value configuration. When the calendar sub-seconds, seconds, minutes, hours, date or day of week match the values programmed in this register and the [RTC\\_ALARM1\\_SUB](#) register, the ALARM1\_SR bit is set.

[3:0]: second units

[6:4]: second tens

[10:7]: minute units

[13:11]: minute tens

[17:14]: hour units

[19:18]: hour tens

[23:20]: bits[23:20] configure date units or bits[22:20] configure day of week

[25:24]: date tens

#### 18.12.4 RTC\_PPMAJUST

Offset: 0x00C

Reset Value: 0x00007FFF

31-16	15-0
RESERVED	PPMADJUST_VALUE
r-0h	rw-7ffffh

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 PPMADJUST\_VALUE:** The RTC clock frequency can be calibrated with a resolution of about 0.5 ppm with a range from -1024 ppm to +1024 ppm.

- 77ff: increase frequency of RTC by 1024 ppm
- 7800: increase frequency of RTC by 1023.5 ppm
- ...
- 7ffd: increase frequency of RTC by 1 ppm
- 7ffe: increase frequency of RTC by 0.5 ppm
- 7fff: no adjustment
- 8000: decrease frequency of RTC by 0.5 ppm
- 8001: decrease frequency of RTC by 1 ppm
- ...
- 87fe: decrease frequency of RTC by 1023.5 ppm
- 87ff: decrease frequency of RTC by 1024 ppm

#### 18.12.5 RTC\_CALENDAR

Offset: 0x010

Reset Value: 0x00000000

31-20	19-0
RESERVED	CALENDAR_VALUE
r-0h	w-0h

**Bits 31-20 RESERVED:** Must be kept, and cannot be modified.

**Bits 19-0 CALENDAR\_VALUE:** RTC calendar time values.

- [3:0]: second units
- [6:4]: second tens
- [10:7]: minute units
- [13:11]: minute tens
- [17:14]: hour units
- [19:18]: hour tens

### 18.12.6 RTC\_CALENDAR\_H

Offset: 0x014

Reset Value: 0x00000841

31-22	21-0
RESERVED	CALENDAR_H_VALUE
r-0h	w-841h

**Bits 31-22 RESERVED:** Must be kept, and cannot be modified.

**Bits 21-0 CALENDAR\_H\_VALUE:** RTC calendar date values.

- [3:0]: date units
- [5:4]: date tens
- [9:6]: month units
- [10]: month tens
- [13:11]: week day units
- [17:14]: year units
- [17:14]: year units

### 18.12.7 RTC\_CYC\_MAX\_VALUE

Offset: 0x018

Reset Value: 0x00008000

31-0
CYC_MAX_VALUE
rw-8000h

**Bits 31-0 CYC\_MAX\_VALUE:** The programmed count value for the periodic counter to reach. When the periodic counter reaches the CYC\_MAX\_VALUE, the periodic counter status flag (bit CYC\_SR) is set. The periodic counter is clocked by the RTC interface clock.

## 18.12.8 RTC\_SR

Offset: 0x01C

Reset Value: 0x00000000

31-7	6	5	4
RESERVED	ALARM0_SR	ALARM1_SR	CYC_SR
r-0h	rw-0h	rw-0h	rw-0h
<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
TAMPER_SR	WAKEUP0_SR	WAKEUP1_SR	WAKEUP2_SR
rw-0h	rw-0h	rw-0h	rw-0h

**Bits 31-7 RESERVED:** Must be kept, and cannot be modified.

**Bit 6 ALARM0\_SR:** Alarm 0 flag.

This flag is set by hardware and cleared by software writing 1 to it.

- 0: Alarm 0 values doesn't match the Calendar
- 1: Alarm 0 values match the Calendar

**Bit 5 ALARM1\_SR:** Alarm 1 flag.

This flag is set by hardware and cleared by software writing 1 to it.

- 0: Alarm 1 values doesn't match the Calendar
- 1: Alarm 1 values match the Calendar

**Bit 4 CYC\_SR:** Periodic counter flag.

This flag is set by hardware and cleared by software writing 1 to it.

- 0: CYC\_MAX\_VALUE is not reached
- 1: CYC\_MAX\_VALUE is reached

**Bit 3 TAMPER\_SR:** Tamper flag.

This flag is set by hardware and cleared by software writing 1 to it.

- 0: tamper pin active level is not detected
- 1: tamper pin active level is detected

**Bit 2 WAKEUP0\_SR:** Wakeup0 flag.

This flag is set by hardware and cleared by software writing 1 to it.

- 0: the Wakeup0 active level is not detected
- 1: the Wakeup0 active level is detected

**Bit 1 WAKEUP1\_SR:** Wakeup1 flag.

This flag is set by hardware and cleared by software writing 1 to it.

- 0: Wakeup1 active level is not detected
- 1: Wakeup1 active level is detected

**Bit 0 WAKEUP2\_SR:** Wakeup2 flag.

This flag is set by hardware and cleared by software writing 1 to it.

- 0: Wakeup2 active level is not detected
- 1: Wakeup2 active level is detected

### 18.12.9 RTC\_ASYNCDATA

Offset: 0x020

Reset Value: 0x00000000

31-20	19-0
RESERVED	SYN_DATA
r-0h	r-0h

**Bits 31-20 RESERVED:** Must be kept, and cannot be modified.

**Bits 19-0 SYN\_DATA:** RTC calendar time values. This register is read-only by software.

- [3:0]: second units
- [6:4]: second tens
- [10:7]: minute units
- [13:11]: minute tens
- [17:14]: hour units
- [19:18]: hour tens

### 18.12.10 RTC\_ASYNCDATA\_H

Offset: 0x024

Reset Value: 0x00000000

31-22	21-0
RESERVED	SYN_DATA_H
r-0h	r-0h

**Bits 31-22 RESERVED:** Must be kept, and cannot be modified.

**Bits 21-0 SYN\_DATA\_H:** RTC calendar date values. This register is read-only by software.

- [3:0]: date units
- [5:4]: date tens
- [9:6]: month units
- [10]: month tens
- [13:11]: week day units
- [17:14]: year units
- [21:18]: year tens

### 18.12.11 RTC\_CR1

Offset: 0x028

Reset Value: 0x00000000

31-8	7	6
RESERVED	SECOND_SR_INT_EN	ALARM0_SR_INT_EN
r-0h	rw-0h	rw-0h
5	4	3
ALARM1_SR_INT_EN	CYC_SR_INT_EN	TAMPER_SR_INT_EN
rw-0h	rw-0h	rw-0h
2	1	0
WAKEUP0_SR_INT_EN	WAKEUP1_SR_INT_EN	WAKEUP2_SR_INT_EN
rw-0h	rw-0h	rw-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bit 7 SECOND\_SR\_INT\_EN:** SECOND\_SR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 6 ALARM0\_SR\_INT\_EN:** ALARM0\_SR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 5 ALARM1\_SR\_INT\_EN:** ALARM1\_SR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 4 CYC\_SR\_INT\_EN:** CYC\_SR (periodic counter) interrupt enable.

- 0: disabled
- 1: enabled

**Bit 3 TAMPER\_SR\_INT\_EN:** TAMPER\_SR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 2 WAKEUP0\_SR\_INT\_EN:** WAKEUP0\_SR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 1 WAKEUP1\_SR\_INT\_EN:** WAKEUP1\_SR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 0 WAKEUP2\_SR\_INT\_EN:** WAKEUP2\_SR interrupt enable.

- 0: disabled
- 1: enabled

### 18.12.12 RTC\_SR1

Offset: 0x02C

Reset Value: 0x00000DFF

31-12	11	10	9
RESERVED	WRITE_ALARM0_SUB_DONE	WRITE_ALARM1_SUB_DONE	SECOND_SR
r-0h	r-1h	r-1h	rw-0h
8	7	6	
WRITE_RTCCR2_DONE	WRITE_RTCCR_DONE	WRITE_ALARM0_DONE	
r-1h	r-1h	r-1h	
5	4	3	
WRITE_ALARM1_DONE	WRITE_PPMADJUST_DONE	WRITE_CALENDAR_DONE	
r-1h	r-1h	r-1h	
2	1	0	
WRITE_CYC_MAX_VALUE_DONE	WRITE_RTCSR_DONE	READ_CALENDAR_DONE	
r-1h	r-1h	r-1h	

**Bits 31-12 RESERVED:** Must be kept, and cannot be modified.

**Bit 11 WRITE\_ALARM0\_SUB\_DONE:** The complete flag of the write operation to register [RTC\\_ALARM0\\_SUB](#). This bit is set and cleared by hardware.

- 0: write operation in progress
- 1: write operation completed

**Bit 10 WRITE\_ALARM1\_SUB\_DONE:** The complete flag of the write operation to register [RTC\\_ALARM1\\_SUB](#). This bit is set and cleared by hardware.

- 0: write operation in progress
- 1: write operation completed

**Bit 9 SECOND\_SR:** Second signal interrupt status. This bit is set by hardware and cleared by software writing 1 to it.

- 0: no second signal interrupt generated
- 1: second signal interrupt generated

**Bit 8 WRITE\_RTCCR2\_DONE:** The complete flag of the write operation to register [RTC\\_CR2](#). This bit is set and cleared by hardware.

- 0: write operation in progress
- 1: write operation completed

**Bit 7 WRITE\_RTCCR\_DONE:** The complete flag of the write operation to register [RTC\\_CR](#). This bit is set and cleared by hardware.

- 0: write operation in progress
- 1: write operation completed

**Bit 6 WRITE\_ALARM0\_DONE:** The complete flag of the write operation to register [RTC\\_ALARM0](#). This bit is set and cleared by hardware.

- 0: write operation in progress
- 1: write operation completed

**Bit 5 WRITE\_ALARM1\_DONE:** The complete flag of the write operation to register [RTC\\_ALARM1](#). This bit is set and cleared by hardware.

- 0: write operation in progress
- 1: write operation completed

**Bit 4 WRITE\_PPMADJUST\_DONE:** The complete flag of the write operation to register [RTC\\_PPMADJUST](#). This bit is set and cleared by hardware.

- 0: write operation in progress
- 1: write operation completed

**Bit 3 WRITE\_CALENDAR\_DONE:** The complete flag of the write operations to registers [RTC\\_CALENDAR](#) and [RTC\\_CALENDAR\\_H](#). This bit is set and cleared by hardware.

- 0: write operation in progress
- 1: write operation completed

**Bit 2 WRITE\_CYC\_MAX\_VALUE\_DONE:** The complete flag of the write operation to register [RTC\\_CYC\\_MAX\\_VALUE](#). This bit is set and cleared by hardware.

- 0: write operation in progress
- 1: write operation completed

**Bit 1 WRITE\_RTCSR\_DONE:** The complete flag of the write operation to register [RTC\\_SR](#). This bit is set and cleared by hardware.

- 0: write operation in progress
- 1: write operation completed

**Bit 0 READ\_CALENDAR\_DONE:** The complete flag of the read operations to registers [RTC\\_CALENDAR\\_R](#) and [RTC\\_CALENDAR\\_R\\_H](#). This bit is set and cleared by hardware.

- 0: write operation in progress
- 1: write operation completed

### 18.12.13 RTC\_CR2

Offset: 0x030

Reset Value: 0x00000000

31-8	7	6-4	3-0
RESERVED	RTC_OUT_POL	RTC_OUT_SEL	RTC_RET_SRAM_ERASE_EN
r-0h	rw-0h	rw-0h	rw-0h

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bit 7 RTC\_OUT\_POL:** RTC IO output polarity.

- 0: RTC IO output level is non-inverted
- 1: RTC IO output level is inverted

**Bits 6-4 RTC\_OUT\_SEL:** RTC IO output selection.

- 0-3: no output
- 4: alarm 0 pulse
- 5: alarm 1 pulse
- 6: cyc pulse
- 7: second signal (50% duty cycle)

**Bits 3-0 RTC\_RET\_SRAM\_ERASE\_EN:** If enabled, the Retention SRAM is erased upon a tamper or a wakeup event. [0]: wakeup0, [1]: wakeup1, [2]: wakeup2, [3]: tamper.

- 0: disabled
- 1: enabled

#### 18.12.14 RTC\_SUB\_SECOND

Offset: 0x034

Reset Value: 0x00000000

31-15	14-0
RESERVED	RTC_SUB_SECOND_VALUE
r-0h	r-0h

**Bits 31-15 RESERVED:** Must be kept, and cannot be modified.

**Bits 14-0 RTC\_SUB\_SECOND\_VALUE:** The subsecond count value of the RTC calendar counter. This register should be read several times with the same result obtained to ensure that the data is correct.

#### 18.12.15 RTC\_CYC\_CNT\_VALUE

Offset: 0x038

Reset Value: 0x00000000

31-0
CYC_CNT_VALUE
r-0h

**Bits 31-0 CYC\_CNT\_VALUE:** Periodic counter value. This register should be read several times with the same result obtained to ensure that the data is correct.

### 18.12.16 RTC\_ALARM0\_SUB

Offset: 0x03C

Reset Value: 0x00000000

31-20	19-16	15	14-0
RESERVED	RTC_ALARM0_SUB_MASK	RESERVED	RTC_ALARM0_SUB_VALUE
r-0h	rw-0h	r-0h	rw-0h

**Bits 31-20 RESERVED:** Must be kept, and cannot be modified.

**Bits 19-16 RTC\_ALARM0\_SUB\_MASK:** Alarm 0 sub-second mask configuration. If sub-seconds are used in Alarm 0, it is recommended not to perform RTC PPM calibration.

- 0: No comparison on sub-seconds for Alarm 0.
- 1: RTC\_ALARM0\_SUB\_VALUE [14:1] are not involved in Alarm 0 comparison. Only bit0 is compared.
- 2: RTC\_ALARM0\_SUB\_VALUE [14:2] are not involved in Alarm 0 comparison. Only bits[1:0] are compared.
- N: [N-1:0] matches of the sub-second count, and other sub-second count bits are ineffective.

**Bit 15 RESERVED:** Must be kept, and cannot be modified.

**Bits 14-0 RTC\_ALARM0\_SUB\_VALUE:** Alarm 0 sub-seconds value. When the calendar subseconds, seconds, minutes, hours, date or day of week match the values programmed in this register and the [RTC\\_ALARM0](#) register, the ALARM0\_SR bit is set.

### 18.12.17 RTC\_ALARM1\_SUB

Offset: 0x040

Reset Value: 0x00000000

31-20	19-16	15	14-0
RESERVED	RTC_ALARM1_SUB_MASK	RESERVED	RTC_ALARM1_SUB_VALUE
r-0h	rw-0h	r-0h	rw-0h

**Bits 31-20 RESERVED:** Must be kept, and cannot be modified.

**Bits 19-16 RTC\_ALARM1\_SUB\_MASK:** Alarm 1 sub-second mask configuration. If sub-seconds are used in Alarm 1, it is recommended not to perform RTC PPM calibration.

- 0: No comparison on sub-seconds for Alarm 1.
- 1: RTC\_ALARM1\_SUB\_VALUE [14:1] are not involved in Alarm 1 comparison. Only bit0 is compared.
- 2: RTC\_ALARM1\_SUB\_VALUE [14:2] are not involved in Alarm 1 comparison. Only bits[1:0] are compared.
- N: [N-1:0] matches of the sub-second count, and other sub-second count bits are ineffective.

**Bit 15 RESERVED:** Must be kept, and cannot be modified.

**Bits 14-0 RTC\_ALARM1\_SUB\_VALUE:** Alarm 1 sub-seconds value. When the calendar subseconds, seconds, minutes, hours, date or day of week match the values programmed in this register and the [RTC\\_ALARM1](#) register, the ALARM1\_SR bit is set.

### 18.12.18 RTC\_CALENDAR\_R

Offset: 0x044

Reset Value: 0x00000000

31-20	19-0
RESERVED	CALENDAR_SYNC
r-0h	r-0h

**Bits 31-20 RESERVED:** Must be kept, and cannot be modified.。

**Bits 19-0 CALENDAR\_SYNC:** RTC\_CALENDAR\_R register values (seconds, minutes and hours). This register should be read several times with the same result obtained to ensure that the data is correct.

### 18.12.19 RTC\_CALENDAR\_R\_H

Offset: 0x048

Reset Value: 0x00000841

31-22	21-0
RESERVED	CALENDAR_H_SYNC
r-0h	r-841h

**Bits 31-22 RESERVED:** Must be kept, and cannot be modified.

**Bits 21-0 CALENDAR\_H\_SYNC:** RTC\_CALENDAR\_R\_H register values (date or day of week, month and year). This register should be read several times with the same result obtained to ensure that the data is correct.

# 19.

# Low-power UART (LPUART)

## 19.1 Introduction

LPUART (Low-power Universal Asynchronous Receiver/Transmitter) is a low-power serial port peripheral. When the 32K clock is used, the LPUART communications can be up to 9600 baud/s. Even in Deepsleep mode, the LPUART can be woken up by received data.

LPUART supports CTS (Clear To Send)/RTS (Require To Send) flow control.

DMA (direct memory access) can be used for data transmission and reception.

## 19.2 Main Features

- Programmable baud rate
- Programmable data format (support 5, 6, 7 or 8 data bits, 1 or 2 stop bits and 1 or no parity bit)
- DMA capability
- 1 byte deep TX FIFO/RX FIFO
- CTS/RTS flow control
- Interrupt generation
- Wakeup CPU from low-power modes

## 19.3 Functional Description

### 19.3.1 Data Format

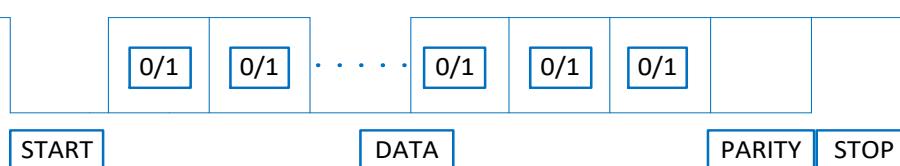


Figure 19-1 LPUART Data Format

When LPUART is idle, its data line should be kept at high level.

For data transmission, the start bit (START), data bits (DATA), parity bit (PARITY) and stop bits (STOP) are sequentially transmitted. The meaning of each bit is as follows:

- (1) **Start Bit:** 0 signal is sent first to indicate the start of data transmission.
- (2) **Data Bits:** 5, 6, 7 or 8 data bits are transmitted in sequence.
- (3) **Parity Bit:** After the data bits, the parity bit is transmitted, or it can be configured as no parity bit.
- (4) **Stop Bit:** 1 or 2 stop bits mark the end of data transmission.

### 19.3.2 Baud Rate Generation

The LPUART baud rate divisor consists of an integer part and a fractional part. This is mainly configured through the LPUART\_BAUD\_RATE\_INT and LPUART\_BAUD\_RATE\_FRA bits in the [LPUART\\_CRO](#) register.

Taking an LPUART interface clock frequency of 32.768kHz and 9600 baud/s as an example, the baud rate divisor is  $32768/9600=3.413$ . Thus, set the integer part of the baud rate divisor to **3** through the LPUART\_BAUD\_RATE\_INT bit, and set the fractional part of the baud rate divisor to **7** ( $0.413*16=6.608$ , rounded to 7) through the LPUART\_BAUD\_RATE\_FRA bit.

### 19.3.3 CTS/RTS Flow Control

The connection between two LPUART devices is shown in the following figure:

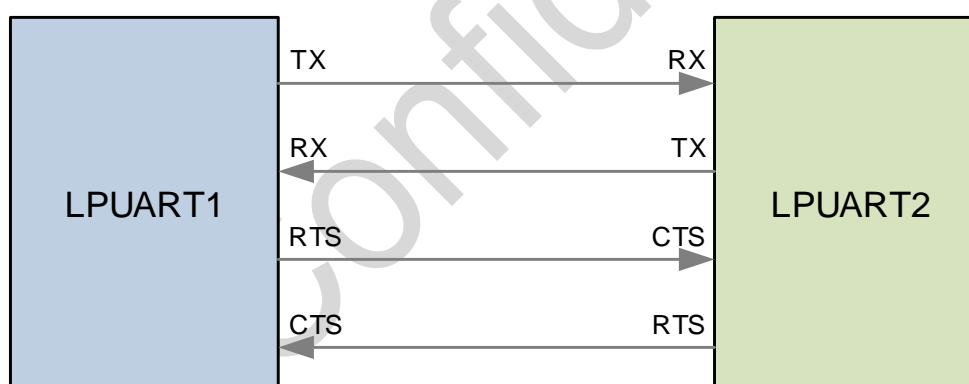


Figure 19-2 Connection between Two LPUART Devices

**RTS (Require to Send)** is an output signal used to determine whether the device is ready to receive data. It is active low, so the low level indicates that the device is ready for data reception.

**CTS (Clear to Send)** is an input signal used to determine whether the device can send data to the other. It is active low, so the low level indicates that the device can send data to the other.

### 19.3.4 DMA

#### LPUART DMA Transmit Process:

- (1) Enable the DMA\_TX\_EN bit in register *LPUART\_CR1*.
- (2) Configure register *LPUART\_DATA* as the destination address of DMA.
- (3) Configure the memory address of the data to be sent as the source address of DMA.
- (4) Configure the data width of DMA transfer to 8 bits by configuring the SRC\_TR\_WIDTH and DES\_TR\_WIDTH bits to 0 in the *DMA\_CTLx* register.
- (5) Configure the DMA burst length to 1 by configuring the SRC\_MSIZE and DEST\_MSIZE bits to 0 in the *DMA\_CTLx* register.
- (6) Configure the total length of DMA data transfer.
- (7) Configure DMA handshake type to DMA\_HANDSHAKE\_LPUART\_TX.
- (8) Activate the DMA.

When the DMA transfer is completed, the CH\_EN\_x bit in the DMA\_CHENREG register is cleared.

#### LPUART DMA Receive Process:

- (1) Enable the DMA\_RX\_EN bit in register *LPUART\_CR1*.
- (2) Configure register *LPUART\_DATA* as the source address of DMA.
- (3) Configure the memory address of the data to be received as the destination address of DMA.
- (4) Configure the data width of DMA transfer to 8 bits by configuring the SRC\_TR\_WIDTH and DES\_TR\_WIDTH bits to 0 in the *DMA\_CTLx* register.
- (5) Configure the DMA burst length to 1 by configuring the SRC\_MSIZE and DEST\_MSIZE bits to 0 in the *DMA\_CTLx* register.
- (6) Configure the total length of DMA data transfer.
- (7) Configure DMA handshake type to DMA\_HANDSHAKE\_LPUART\_RX.
- (8) Activate the DMA.

When the DMA transfer is completed, the CH\_EN\_x bit in the DMA\_CHENREG register is cleared.

### 19.3.5 Interrupts

#### LPUART interrupt signals:

- TX\_DONE interrupt
- TXFIFO\_EMPTY interrupt
- RXFIFO\_NOT\_EMPTY interrupt
- RX\_OVERFLOW interrupt
- STOP\_ERR interrupt
- PARITY\_ERR interrupt
- START\_INVALID interrupt
- RX\_DONE interrupt
- START\_VALID interrupt

### 19.3.6 CPU Wakeup from Low-power Modes

RX low-level, START\_VALID and RX\_DONE signals can be used to wakeup the CPU from low power modes.

LPUART wakeup is enabled by configuring the LPUART\_WAKEUP\_EN[[24:22] bits in register [LPUART\\_CR0](#).

## 19.4 LPUART Registers

Base Address:0x40005000

**Table 19-1 LPUART Registers Summary**

Register	Offset	Description
LPUART_CR0	0x00	Control Register 0
LPUART_CR1	0x04	Control Register 1
LPUART_SR0	0x08	Status Register 0
LPUART_SR1	0x0C	Status Register 1
LPUART_DATA	0x10	Data Register

### 19.4.1 LPUART\_CR0

Offset: 0x00

Reset Value: 0x00000E13

31-27	26	25	24-22	21-10
RESERVED	LPUART_RTS_EN	LPUART_RX_EN	LPUART_WAKEUP_EN	LPUART_BAUD_RA TE_INT
r	r/w	r/w	r/w	r/w
9-6	5	4-2	1-0	
LPUART_BAUD_RATE_FRA	LPUART_STOP_LEN	LPUART_PARITY_CFG	LPUART_DATA_LEN	
r/w	r/w	r/w	r/w	

**Bits 31-27 RESERVED:** Must be kept, and cannot be modified.

**Bit 26 LPUART\_RTS\_EN:** LPUART RTS flow control enable.

- 0: disabled
- 1: enabled

**Bit 25 LPUART\_RX\_EN:** LPUART reception enable.

- 0: disabled
- 1: enabled

**Bits 24-22 LPUART\_WAKEUP\_EN:** LPUART wakeup enable.

[22] Enable RX low-level signal as a wakeup source

- 0: disabled
- 1: enabled

[23] Enable START\_VALID signal as a wakeup source

- 0: disabled
- 1: enabled

[24] Enable RX\_DONE signal as a wakeup source

- 0: disabled
- 1: enabled

**Bits 21-10 LPUART\_BAUD\_RATE\_INT:** The integer part of the baud rate divisor.

The frequency division factor is equal to the UART interface clock frequency/baud rate.

Take the UART interface clock frequency as 32.768KHz and the baud rate as 9600 as an example, the frequency division factor is  $32768/9600=3.413$ , lpuart\_baud\_rate\_int is configured as 3, and lpuart\_baud\_rate\_fra is configured as  $0.413*16=6$  or 7.

**Bits 9-6 LPUART\_BAUD\_RATE\_FRA:** The fractional part of the baud rate divisor.

**Bit 5 LPUART\_STOP\_LEN:** LPUART STOP bits configuration.

- 0: 1 stop bit
- 1: 2 stop bits

**Bits 4-2 LPUART\_PARITY\_CFG:** LPUART parity bit configuration.

- 0: even parity
- 1: odd parity
- 2: parity bit is 0
- 3: parity bit is 1
- >3: no parity

**Bits 1-0 LPUART\_DATA\_LEN:** LPUART data length.

Data width=LPUART\_DATA\_LEN+5

#### 19.4.2 LPUART\_CR1

Offset: 0x004

Reset Value: 0x00000000

31-13	12	11	10	9
RESERVED	LPUART_CTS_EN	DMA_TX_EN	DMA_RX_EN	LPUART_TX_EN
r	r/w	r/w	r/w	r/w
8	7	6	5	4
TX_DONE_INT_EN	TXFIFO_EMPTY_INT_EN	RXFIFO_NOT_EMPTY_INT_EN	RX_OVERFLOW_INT_EN	STOP_ERR_INT_EN
r/w	r/w	r/w	r/w	r/w
3	2	1	0	
PARITY_ERR_INT_EN	START_INVALID_INT_EN	RX_DONE_INT_EN	START_VALID_INT_EN	
r/w	r/w	r/w	r/w	

**Bits 31:13 RESERVED:** Must be kept, and cannot be modified.

**Bit 12 LPUART\_CTS\_EN:** LPUART CTS flow control enable.

- 0: disabled
- 1: enabled

**Bit 11 DMA\_TX\_EN:** DMA transmission requests enable.

- 0: disabled
- 1: enabled

**Bit 10 DMA\_RX\_EN:** DMA reception requests enable.

- 0: disabled
- 1: enabled

**Bit 9 LPUART\_TX\_EN:** LPUART transmission enable.

- 0: disabled
- 1: enabled

**Bit 8 TX\_DONE\_INT\_EN:** TX\_DONE interrupt enable.

- 0: disabled
- 1: enabled

**Bit 7 TXFIFO\_EMPTY\_INT\_EN:** TXFIFO\_EMPTY interrupt enable.

- 0: disabled
- 1: enabled

**Bit 6 RXFIFO\_NOT\_EMPTY\_INT\_EN:** RXFIFO\_NOT\_EMPTY interrupt enable.

- 0: disabled
- 1: enabled

**Bit 5 RX\_OVERFLOW\_INT\_EN:** RX\_OVERFLOW interrupt enable.

- 0: disabled
- 1: enabled

**Bit 4 STOP\_ERR\_INT\_EN:** STOP\_ERR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 3 PARITY\_ERR\_INT\_EN:** PARITY\_ERR interrupt enable.

- 0: disabled
- 1: enabled

**Bit 2 START\_INVALID\_INT\_EN:** START\_INVALID interrupt enable.

- 0: disabled
- 1: enabled

**Bit 1 RX\_DONE\_INT\_EN:** RX\_DONE interrupt enable.

- 0: disabled
- 1: enabled

**Bit 0 START\_VALID\_INT\_EN:** START\_VALID interrupt enable.

- 0: disabled
- 1: enabled

#### 19.4.3 LPUART\_SR0

Offset: 0x008

Reset Value: 0x00000000

31-6		5	4
RESERVED		RX_OVERFLOW_SR	STOP_ERR_SR
r		r/w	r/w
3	2	1	0
PARITY_ERR_SR	START_INVALID_SR	RX_DONE_SR	START_VALID_SR
r/w	r/w	r/w	r/w

**Bits 31-6 RESERVED:** Must be kept, and cannot be modified.

**Bit 5 RX\_OVERFLOW\_SR:** RX\_OVERFLOW flag is used to indicate whether a RX buffer overflow has occurred. This bit is set by hardware and cleared by software writing 1 to it.

- 0: no RX buffer overflow

- 1: RX buffer overflow occurred

**Bit 4 STOP\_ERR\_SR:** STOP\_ERR flag is used to indicate whether a Stop error has occurred. This bit is set by hardware and cleared by software writing 1 to it.

- 0: no Stop error
- 1: Stop error occurred

**Bit 3 PARITY\_ERR\_SR:** PARITY\_ERR flag is used to indicate whether a parity error has occurred. This bit is set by hardware and cleared by software writing 1 to it.

- 0: no parity error
- 1: parity error occurred

**Bit 2 START\_INVALID\_SR:** START\_INVALID flag is used to indicate whether an invalid Start bit has been received. This bit is set by hardware and cleared by software writing 1 to it.

- 0: no invalid Start
- 1: invalid Start bit has been received

**Bit 1 RX\_DONE\_SR:** RX\_DONE flag is used to indicate whether the data reception is completed. This bit is set by hardware and cleared by software writing 1 to it.

- 0: data reception not completed
- 1: data reception completed

**Bit 0 START\_VALID\_SR:** START\_VALID flag is used to indicate whether a valid Start bit has been received. This bit is set by hardware and cleared by software writing 1 to it.

- 0: no valid Start
- 1: valid Start bit has been received

#### 19.4.4 LPUART\_SR1

Offset: 0x00C

Reset Value: 0x00000016

31-6		5	4
RESERVED		TX_DONE	TXFIFO_EMPTY
r		r/w	r
3	2	1	0
RXFIFO_NOT_EMPTY	WRITE_CR0_DONE	WRITE_SR0_DONE	RESERVED
r	r	r	r

**Bits 31-6 RESERVED:** Must be kept, and cannot be modified.

**Bit 5 TX\_DONE:** TX\_DONE flag. This bit is set by hardware and cleared by software writing 1 to it.

- 0: data transmission in progress
- 1: data transmission completed

**Bit 4 TXFIFO\_EMPTY:** TXFIFO\_EMPTY flag. This bit is set by hardware and cleared by software writing to the [LPUART\\_DATA](#) register.

- 0: non-empty
- 1: empty

**Bit 3 RXFIFO\_NOT\_EMPTY:** RXFIFO\_NOT\_EMPTY flag. This bit is set by hardware and cleared by software reading the [LPUART\\_DATA](#) register.

- 0: empty
- 1: non-empty

**Bit 2 WRITE\_CR0\_DONE:** The status of a write operation to the [LPUART\\_CR0](#) register. This bit is set and cleared by hardware.

- 0: write operation to the [LPUART\\_CR0](#) register in progress
- 1: write operation to the [LPUART\\_CR0](#) register completed

**Bit 1 WRITE\_SR0\_DONE:** The status of a write operation to the [LPUART\\_SR0](#) register. This bit is set and cleared by hardware.

- 0: write operation to the [LPUART\\_SR0](#) register in progress
- 1: write operation to the [LPUART\\_SR0](#) register completed

**Bit 0 RESERVED:** Must be kept, and cannot be modified.

#### 19.4.5 LPUART\_DATA

Offset: 0x010

Reset Value: 0x00000000

31-8	7-0
RESERVED	LPUART_DATA
r	r/w

**Bits 31-8 RESERVED:** Must be kept, and cannot be modified.

**Bits 7-0 LPUART\_DATA:** LPUART TX/RX data.

**Notes:**

1. If the data width is less than 8 bits, the less significant bits of the LPUART\_DATA register is valid.
2. Before reading the LPUART\_DATA register, check the RXFIFO\_NOT\_EMPTY flag to ensure that there is data in RXFIFO; before writing to the LPUART\_DATA register, check the TXFIFO\_EMPTY flag to ensure that the TXFIFO can be written.

# 20.

# Low-power timer (LPTIM)

## 20.1 Introduction

LPTIMER (Low Power Timer) is a 16-bit timer. Due to multiple clock sources, LPTIMER can run in all operating modes except standby mode and supports wake-up from all low-power operating modes. There are two separate LPTIMER, respectively LPTIMER0 and LPTIMER1.

## 20.2 Main features

LPTIMER includes the following functions:

- Selecting internal clock and external clock as counting clock
- 16 bits counter, up-counting, auto-reload ● Two counting modes, single counting and continuous counting
- Software trigger and external trigger source
- Prescaler
- PWM generation
- Single pulse, Set-once, and Timeout mode output
- DEBUG mode control
- Supports generating channel output events, match events, overflow events, trigger events, DOWN events, and UP events as wake-up signal outputs
- Quadrature decoding
- Interrupt signal generation

LPTIMER block diagram:

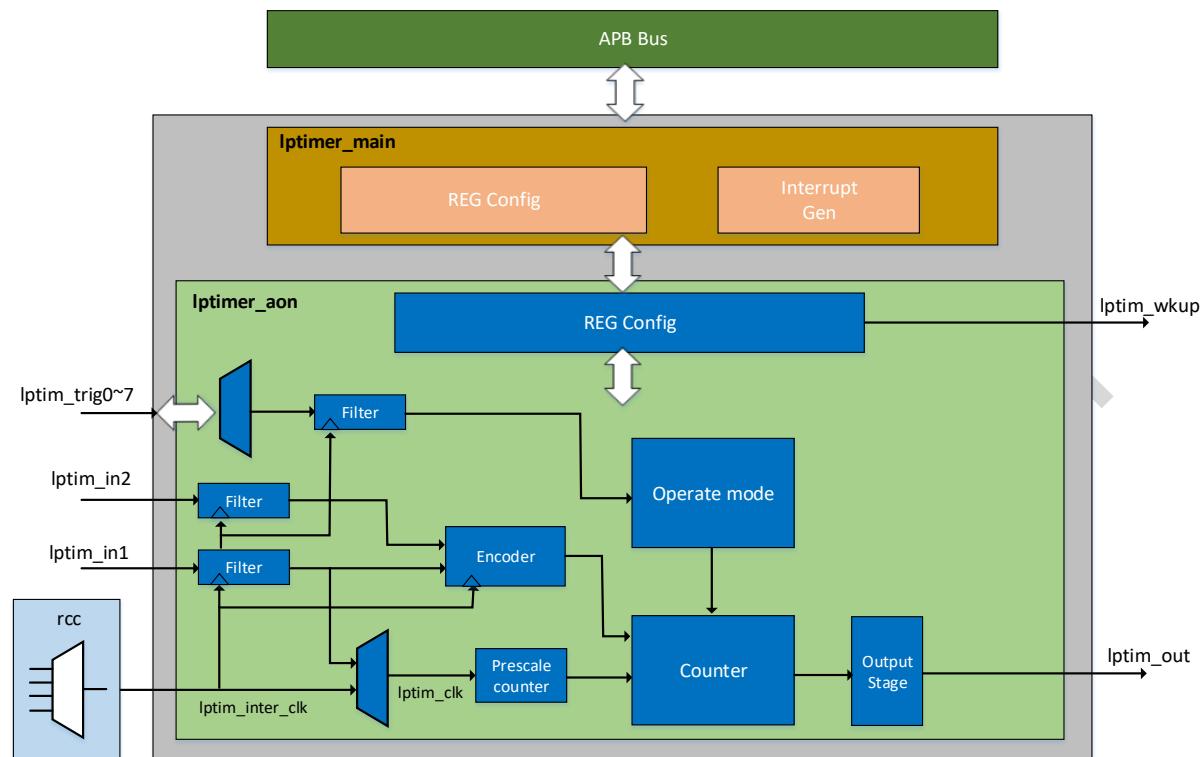


Figure 20-1 LPTIMER Diagram

- **Iptim\_trig0~7:** External trigger source of LPTIMER
- **Iptim\_in2:** LPTIMER's IN2 pin
- **Iptim\_in1:** LPTIMER's IN1 pin
- **Iptim\_wkup:** LPTIMER wake-up signal
- **Iptim\_out:** LPTIMER's OUT pin

## 20.3 Interface clock

LPTIMER interface clock source can be internal clock and external clock. The internal clock source includes PCLK0, RCO3.6M, XO32K, and RCO32K. The external clock source is input through the GPIO of IN1. For clock configuration and selection, please refer to the RCC chapter.

## 20.4 Counter clock selection

In addition to the internal and external interface clocks, the LPTIMER count clocks are also divided into internal and external clock sources. The internal and external clock sources are consistent with the interface clock. The register bit that controls the count clock selection is the COUNTMODE of the register LPTIM\_CFGR. A value of 0 indicates that the counter is controlled by the internal clock, and a value of 1 indicates that the counter is controlled by the external clock.

If the LPTIM1\_EXT\_CLK\_SEL bit or LPTIM\_EXT\_CLK\_SEL bit of Register RCC\_CR1 or the RCC module is 0, it means that the interface clock of LPTIMER0 or LPTIMER1 is the internal clock, then COUNTMODE The value can be 0 or 1, that is, the counting clock can be either an internal clock or an external clock; if LPTIM1\_EXT\_CLK\_SEL bit or LPTIM\_EXT\_CLK\_SEL bit is 1, then the COUNTMODE bit of Register LPTIM\_CFGR of LPTIMER0 or LPTIMER1 can only be set to 0, then 0 It does not mean that the counting clock is an internal clock, but it means that the COUNTMODE value needs to be cleared, and the counting clock can only be an external clock.

## 20.5 Counter

Except for the encoding mode, the counter only supports up-counting. When counting to ARR, an ARRM interrupt is generated, and the counter returns to 0 and starts counting again. If the timeout mode is enabled, in addition to clearing the counter when the counter value increases to ARR, the trigger signal can also clear the counter to start counting again. If the encoding mode is enabled, the counting direction of the counter is controlled by hardware. When counting up to ARR, an ARRM event is generated and the counter is cleared. When counting down to 0, ARR is reloaded to the counter.

## 20.6 Counting modes

LPTIMER supports two counting modes, single counting and continuous counting. In single counting mode, the first trigger signal (hardware or software) arriving during the counter stop phase will trigger the counter to start counting, and the trigger signal during the counting process will be ignored, the counter will stop counting when it reaches ARR, and will not start counting again until the next trigger signal arrives, and so on. In the continuous counting mode, once triggered (hardware or software), the counter will keep counting, from 0 to ARR, then return to 0 counts again, and so on.

The two counting modes can be switched at any time (provided that enable is set to Bits). For example, if LPTIMER is configured as single counting mode, if CNTSTRT of BitsRegister LPTIM\_CR is set, the counter will not stop counting when it reaches the ARR value; LPTIMER is configured as continuous. In counting mode, if the SNGSTRT of BitsRegister LPTIM\_CR is set, the counter will stop counting when it reaches ARR until the next trigger signal arrives. Therefore, the status figure is as follows:

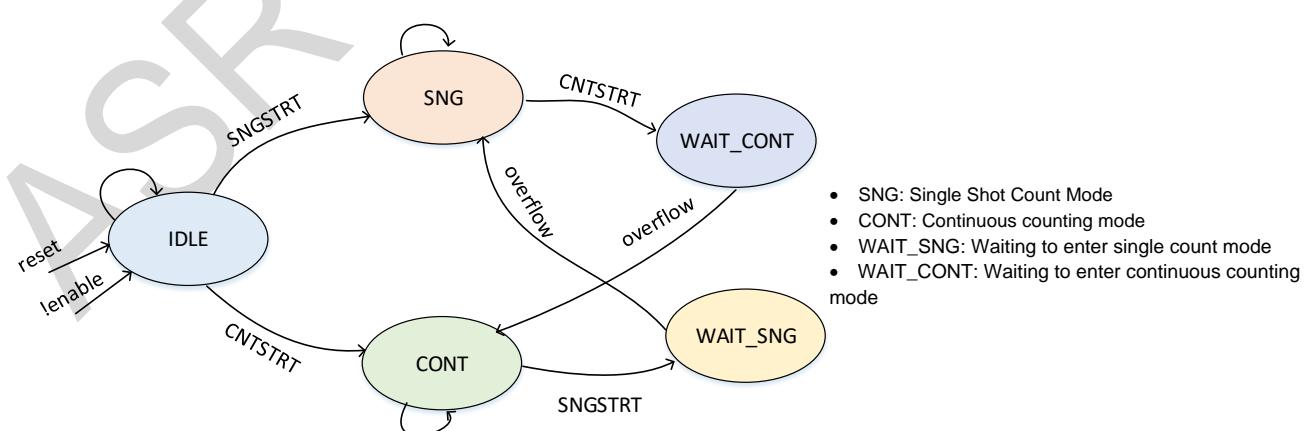


Figure 20-2 Counting mode conversion Diagram

## 20.7 Trigger sources

There are two ways to trigger LPTIMER counting, one is software trigger, and the other is trigger from external trigger source. It is controlled by the TRIGEN bits section of Register LPTIM\_CFGR. When the value is 0, it is software trigger, and when it is non-zero, it is external trigger. When it is an external trigger, you can set the external trigger signal to be valid on the rising edge, valid on the falling edge or valid on both edges. LPTIMER has 8 trigger input sources that can be selected. The external trigger sources of LPTIMER0 are as follows in Table:

**Table 20-1 LPTIMER0 external trigger source**

TRIGSEL	External Trigger	Comment
lptim_ext_trig0	lptim_etr	Lptimer etr pin input
lptim_ext_trig1	comp0	Comp0 output
lptim_ext_trig2	comp1	Comp1 output
lptim_ext_trig3	rtc_cyc_counter	RTC cyc counter output pulse
lptim_ext_trig4	rtc_alarm0	RTC alarm0 output pulse
lptim_ext_trig5	rtc_alarm1	RTC alarm1 output pulse
lptim_ext_trig6	gpio	GPIO58
lptim_ext_trig7	gpio	GPIO59

LPTIMER1 external trigger sources are as follows in Table:

**Table 20-2 LPTIMER1 external trigger source**

TRIGSEL	External Trigger	Comment
lptim_ext_trig0	lptim_etr	Lptimer etr pin input
lptim_ext_trig1	comp0	Comp0 output
lptim_ext_trig2	comp1	Comp1 output
lptim_ext_trig3	rtc_cyc_counter	RTC cyc counter output pulse
lptim_ext_trig4	rtc_alarm0	RTC alarm0 output pulse
lptim_ext_trig5	rtc_alarm1	RTC alarm1 output pulse
lptim_ext_trig6	gpio	GPIO60
lptim_ext_trig7	gpio	GPIO61

## 20.8 Prescaler

The Count Enable signal can be divided by software configuration, supporting 1, 2, 4, 8, 16, 32, 64, and 128 frequency division. The prescaler is configured by configuring the PRESC Bits section of Register LPTIM\_CFGR. The prescaler is implemented through counter, that is counting enable signal generated by the upper stage circuit will be used as the counting enable of the prescaler. When the frequency division counter counts to the preloaded frequency division value, a pulse is output as the counting enable of the next level counter, and then the frequency division counter resets to zero and counts again, and so on.

## 20.9 PWM

LPTIMER can generate PWM waveforms. The polarity of the waveform can be controlled by the WAVPOL bit of Register LPTIM\_CFGR, and the duty cycle can be controlled by the values of Register LPTIM\_CMP and LPTIM\_ARR. Taking software triggering and internal clock counting as an example, the process of configuring PWM is as follows:

- (1) Configure the COUNTMODE of Register LPTIM\_CFGR to 0, that is, set the internal clock count.
- (2) The PRESC of Register LPTIM\_CFGR is the default value, that is, the prescaler is not set.
- (3) The value of PRELOAD in the configuration register LPTIM\_CFGR is 0, which means that the cache function of the registers LPTIM\_CMP and LPTIM\_ARR is not enabled. It can be enabled if necessary.
- (4) Caching function of LPTIM\_ARR can also be enabled if needed.
- (5) Configure WAVPOL of Register LPTIM\_CFGR to 0, that is, the waveform output is not inverted.
- (6) Configure the WAVE of Register LPTIM\_CFGR to 0.
- (7) To enable LPTIMER, set the ENABLE bit in the LPTIM\_CR register.
- (8) Set the values of registers LPTIM\_ARR and LPTIM\_CMP.
- (9) Enable the continuous counting function by setting CNTSTRT in register LPTIM\_CR.

## 20.10 Single-pulse, Set-once, Timeout output mode

In single pulse mode, when the counter is not counting, and the first trigger signal is detected, the counting Enable will set the Bits. If the ARR or enable is cleared or the module resets the Bits, the counting Enable will be cleared. The trigger signal during the counting process will be ignored, as shown in following Figure:

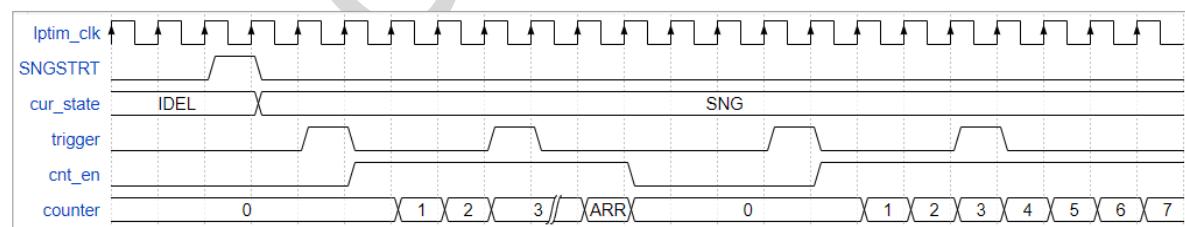


Figure 20-3 Single pulse counting

Single pulse mode is achieved by configuring the WAVE of Register LPTIM\_CFGR to 0 and the SNGSTRT of Register LPTIM\_CR to 1.

In Set-once mode, after the first trigger signal is detected, the counting Enable is set to Bits. If ARR is counted, the counting Enable is cleared. The trigger signal during the counting process will be masked. The masking signal is implemented through mask, that is, detection After reaching the first trigger signal, the mask is valid and all subsequent trigger signals are masked, as shown in the following figure:

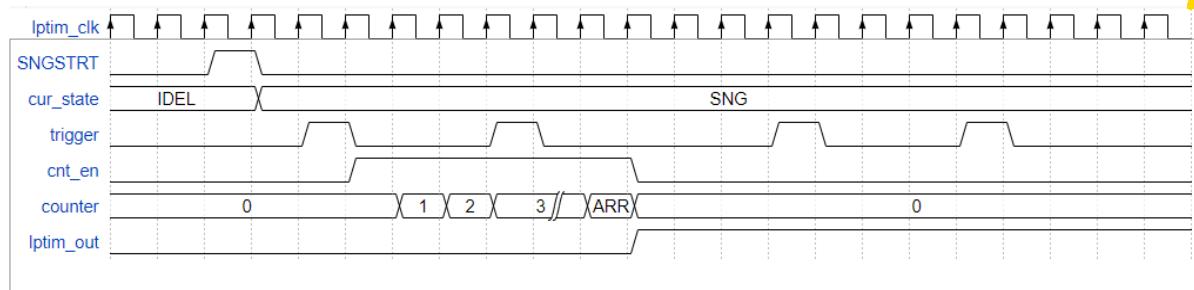


Figure 20-4 Set-once counting

Set-once mode is configured by configuring Register LPTIM\_CFGR's WAVE to 1 and Register LPTIM\_CR's SNGSTRT is implemented as 1.

Timeout mode is similar to continuous counting mode. Once triggered, counting Enable is always valid. The difference is that the trigger signal during the counting process will cause the counter to count again from 0, and the output waveform will also be cleared, as shown in the following figure:

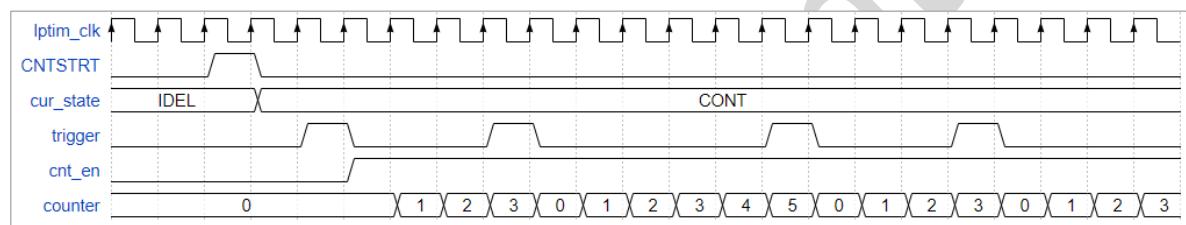


Figure 20-5 Timeout counting

Timeout mode is implemented by configuring the WAVE of Register LPTIM\_CFGR to 0 and the CNTSTRT of Register LPTIM\_CR to 1.

## 20.11 Quadrature encoder mode

LPTIMER supports quadrature encoding counting function, which can input quadrature signals through IN1 and IN2 for counting and direction detection. There are three encoding modes, counting only on rising edges, counting on falling edges and counting on both edges. Enable the encoding mode through Register The ENC of LPTIM\_CFGR is controlled, and the edge control of the encoding mode is realized through the CKPOL of Register LPTIM\_CFGR. Under this function, the two channel inputs can be configured with digital filtering functions. The filtering Enable is controlled by CKFLT\_ENABLE of Register LPTIM\_CFGR, and the filter value is controlled by CKFLT of Register LPTIM\_CFGR. Configuration. Through the combination of two channel signals, the counting Enable and direction control signals can be generated to control the addition and subtraction of counter. The specific combination method is shown in the Table below:

**Table 20-3 Quadrature encoder channel signals**

Encoding mode	IN1/IN2 level	IN1		IN2	
		rising edge	falling edge	rising edge	falling edge
Rising edge count	high level	count down	-	count up	-
	low level	count up	-	count down	-
Falling edge count	high level	-	count up	-	count down
	low level	-	count down	-	count up
Both edges count	high level	count down	count up	count up	count down
	low level	count up	count down	count down	count up

The IN1 and IN2 input signal frequencies must be less than 1/4 of the LPTIMER clock frequency.

## 20.12 DEBUG mode control

LPTIMER can be configured by software whether to stop counting under debug. The DEBUG mode counting control of LPTIMER0 and LPTIMER1 is implemented through the CR2 Register of SYSCFG. If this function is enabled, LPTIMER stops counting when entering the system debug mode (the counter will not be initialized).

## 20.13 Wake-up signals

LPTIMER has 6 wake-up signal outputs.

- **Channel output signal**, at this time the channel output will be output as a wake-up signal.
- **Matching event (CMPM)**, at this time, the matching event between counter and Register LPTIM\_CMP will be output as a wake-up signal.
- **Overflow event (ARRM)**, at this time the overflow event will be output as a wake-up signal.
- **Trigger event (EXTTRIG)**, the valid trigger event at this time will be output as a wake-up signal.
- **DOWN event**, if the counting direction changes from up counting to down counting, the DOWN event bit will set. At this time, the DOWN event will be output as a wake-up signal.
- **UP event**, if the counting direction changes from down counting to up counting, the UP event bit will set. At this time, the UP event will be output as a wake-up signal.

In addition to the channel output signal, the above wake-up signals are all flag bits of the LPTIM\_ISR Register, and have independent EnableBits. The EnableBits are the OUT\_WKUP\_EN, CMPM\_WKUP\_EN, ARRM\_WKUP\_EN, EXTTRIG\_WKUP\_EN, DOWN\_WKUP\_EN, and UP\_WKUP\_EN bits of Register LPTIM\_CFGR respectively. The wake-up signal is ANDed with the corresponding enable bits. The relationship between each Wakeup Source is OR.

## 20.14 Interrupts

Interrupt signals of LPTIMER:

**Table 20-4 LPTIMER interrupts**

Interrupt	Description
DOWN interrupt	In encoding mode, counting direction changes from upward to downward
UP interrupt	In encoding mode, counting direction changes from downward to upward
ARROK interrupt	ARR value loading is completed
CMPOK interrupt	ARR value loading is completed
EXTTRIG interrupt	Valid trigger edge is detected
ARRM interrupt	Counter value reaches ARR
CMPM interrupt	Counter value matches CMP

The above interrupt is enabled by configuring Register LPTIM\_IER. The status of all interrupts can be obtained through LPTIM\_SR1 Register.

## 20.15 LPTIMER registers

LPTIMER0 Base Address: 0x4000D000

LPTIMER1 Base Address: 0x4000D800

**Table 20-5 LPTIMER Registers Summary**

Register	Offset	Description
LPTIM_ISR	0x000	Status Register
LPTIM_ICR	0x004	Status Clear Register
LPTIM_IER	0x008	Interrupt Enable Register
LPTIM_CFGR	0x00C	Configuration Register, which needs to be modified when ENABLE of LPTIM_CR Register is cleared
LPTIM_CR	0x010	Control Register
LPTIM_CMP	0x014	Compare Register
LPTIM_ARR	0x018	Auto-reload Register
LPTIM_CNT	0x01C	Counter Register
LPTIM_CSR	0x020	Status flag clear Register. Table indicates whether to clear the completion flag when using Register LPTIM_ICR to clear LPTIM_ISR certain status bits.
LPTIM_SR1	0x024	Interrupt flag Register, interrupt flag bits will be cleared immediately by LPTIM_ICR Register

## 20.15.1 LPTIM\_ISR

Offset: 0x000

Reset Value: 0x000000180

31-9	8	7	6	5	4	3	2	1	0
RESERVED	CROK	CFGROK	DOWN	UP	ARROK	CMPOK	EXTTRIG	ARRM	CMPM
r-0h	r-1h	r-1h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 31-9 RESERVED:** Must be kept, and cannot be modified.

**Bit 8 CROK:** The status of the last write operation to Register LPTIM\_CR. This bit is controlled by hardware, and you need to check whether the last write operation is completed before writing.

- 0: Writing operation in progress
- 1: The last write operation to LPTIM\_CR has been completed

**Bit 7 CFGROK:** The status of the last write operation to LPTIM\_CFGR. This bit is controlled by hardware, and you need to check whether the last write operation is completed before writing.

- **Bit 7 CFGROK:** The status of the last write operation to LPTIM\_CFGR. This bit is controlled by hardware, and you need to check whether the last write operation is completed before writing.
- 1: The last write operation to LPTIM\_CFGR has been completed

**Bit 6 DOWN:** The counting direction changes from upward to downward in encoding mode.

- 0: The counting direction does not change from top to bottom.
- 1: Counting direction changes from upward to downward

It can be cleared by writing to LPTIM\_ICR Register, but it requires a time synchronization clearing pulse, so it cannot be cleared immediately.

**Bit 5 UP:** The counting direction changes from downward to upward in encoding mode.

- 0: The counting direction does not change from bottom to upward.
- 1: Counting direction changes from downward to upward

It can be cleared by writing to LPTIM\_ICR Register, but it requires a time synchronization clearing pulse, so it cannot be cleared immediately.

**Bit 4 ARROK:** ARR value loading status.

- 0: Not loaded yet
- 1: Loading completed

It can be cleared by writing to LPTIM\_ICR Register, but it requires a time synchronization clearing pulse, so it cannot be cleared immediately.

**Bit 3 CMPOK:** CMP value loading status.

- 0: Not loaded yet
- 1: Loading completed

It can be cleared by writing to LPTIM\_ICR Register, but it requires a time synchronization clearing pulse, so it cannot be cleared immediately.

**Bit 2 EXTTRIG:** Whether a valid trigger edge is detected.

- 0: No valid trigger edge detected
- 1: Valid trigger edge detected

It can be cleared by writing to LPTIM\_ICR Register, but it requires a time synchronization clearing pulse, so it cannot be cleared immediately.

**Bit 1 ARRM:** Whether the counter value reaches the ARR value.

- 0: counter value has not reached ARR
- 1: counter value reaches ARR

It can be cleared by writing to LPTIM\_ICR Register, but it requires a time synchronization clearing pulse, so it cannot be cleared immediately.

**Bit 0 CMPM:** counter value matches CMP value.

- 0: counter value does not match CMP value
- 1: counter value matches CMP value

It can be cleared by writing to LPTIM\_ICR Register, but it requires a time synchronization clearing pulse, so it cannot be cleared immediately.

## 20.15.2 LPTIM\_ICR

Offset: 0x004

Reset Value: 0x00000000

31-7	6	5	4	3	2	1	0
RESERVED	DOWNCF	UPCF	ARROKCF	CMPOKCF	EXTTRIGCF	ARRMCF	CMPMCF
w-0h	w-0h	w-0h	w-0h	w-0h	w-0h	w-0h	w-0h

**Bits 31-7 RESERVED:** Must be kept, and cannot be modified.

**Bit 6 DOWNCF:** Clear the DOWN flag. Software writes 1 to clear the flag bit, and the bit is cleared by hardware.

- 0: no action
- 1: clear flag

**Bit 5 UPCF:** Clear the UP flag. Software writes 1 to clear the flag bit, and the bit is cleared by hardware.

- 0: no action
- 1: clear flag

**Bit 4 ARROKCF:** Clear the ARROK flag. Software writes 1 to clear the flag bit, and the bit is cleared by hardware.

- 0: no action
- 1: clear flag

**Bit 3 CMPOKCF:** Clear the CMPOK flag. Software writes 1 to clear the flag bit, and the bit is cleared by hardware.

- 0: no action
- 1: clear flag

**Bit 2 EXTTRIGCF:** Clear the EXTTRIG flag. Software writes 1 to clear the flag bit, and the bit is cleared by hardware.

- 0: no action
- 1: clear flag

**Bit 1 ARRMCF:** Clear the ARRM flag. Software writes 1 to clear the flag bit, and the bit is cleared by hardware.

- 0: no action
- 1: clear flag

**Bit 0 CMPMCF:** Clear the CMPM flag. Software writes 1 to clear the flag Bits, which are cleared by hardware.

- 0: no action
- 1: clear flag

### 20.15.3 LPTIM\_IER

Offset: 0x008

Reset Value: 0x00000000

31-7	6	5	4	3	2	1	0
RESERVED	DOWNIE	UPIE	ARROKIE	CMPOKIE	EXTTRIGIE	ARRMIE	CMPMIE
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 31-7 RESERVED:** Must be kept, and cannot be modified.

**Bit 6 DOWNIE:** DOWN interrupt enable.

- 0: disable
- 1: enable

**Bit 5 UPIE:** UP interrupt enable.

- 0: disable
- 1: enable

**Bit 4 ARROKIE:** ARROK interrupt enable.

- 0: disable
- 1: enable

**Bit 3 CMPOKIE:** CMPOK interrupt enable.

- 0: disable
- 1: enable

**Bit 2 EXTTRIGIE:** EXTTRIG interrupt enable.

- 0: disable
- 1: enable

**Bit 1 ARRMIE:** ARRM interrupt enable.

- 0: disable
- 1: enable

**Bit 0 CMPMIE:** CMPM interrupt enable.

- 0: disable
- 1: enable

## 20.15.4 LPTIM\_CFGR

Offset: 0x00c

Reset Value: 0x00000000

31	30	29	28	27	26
RESERVED	OUT_WKUP_EN	DOWN_WKUP_EN	UP_WKUP_E_N	EXTTRIG_WK_UP_EN	ARRM_WKUP_EN
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h
25	24	23	22	21	20
CMPM_WKUP_EN	ENC	COUNTMODE	PRELOAD	WAVPOL	WAVE
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h
19	18-17	16	15-13	12	11-9
TIMEOUT	TRIGEN	RESERVED	TRIGSEL	RESERVED	PRES
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h
8	7-6	5	4-3	2-1	0
TRGLT_ENAB <sub>LE</sub>	TRGFLT	CKFLT_ENAB <sub>LE</sub>	CKFLT	CKPOL	RESERVED
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bit 31 RESERVED:** Must be kept, and cannot be modified.

**Bit 30 OUT\_WKUP\_EN:** LPTIM\_OUT wake-up Enable.

- 0: LPTIM\_OUT cannot trigger wake-up signal
- 1: LPTIM\_OUT can trigger the wake-up signal

**Bit 29 DOWN\_WKUP\_EN:** DOWN event wake-up Enable.

- 0: DOWN event cannot trigger wake-up signal
- 1: DOWN event can trigger wake-up signal

**Bit 28 UP\_WKUP\_EN:** UP event wake-up Enable.

- 0: UP event cannot trigger wake-up signal
- 1: UP event can trigger wake-up signal

**Bit 27 EXTTRIG\_WKUP\_EN:** External trigger event wake-up Enable.

- 0: External trigger events cannot trigger wake-up signal
- 1: External trigger events can trigger wake-up signal

**Bit 26 ARRM\_WKUP\_EN:** Count overflow event wake-up Enable (except ENC mode).

- 0: Count overflow cannot trigger wake-up signal
- 1: Count overflow event triggers wake-up signal

**Bit 25 CMPM\_WKUP\_EN:** Count matching event wake-up Enable.

- 0: Count match cannot trigger wake-up signal
- 1: Count matching event triggers wake-up signal

**Bit 24 ENC:** Encoding mode Enable.

- 0: Disable encoding mode

- 0: Disable encoding mode

**Bit 23 COUNTMODE:** Counting mode selection.

- 0: counter is controlled by internal clock
- 1: counter is controlled by external clock

**Bit 22 PRELOAD:** Register cache Enable.

- 0: ARR and CMP are directly operated by software
- 1: ARR and CMP are updated by update events.

**Bit 21 WAVPOL:** Output waveform polarity.

- 0: normal polarity
- 1: inverted polarity

**Bit 20 WAVE:** Waveform shape.

- 0: disable Set-once, select PWM or single pulse mode
- 1: enable Set-once mode

**Bit 19 TIMEOUT:** Timeout mode Enable.

- 0: disable Timeout mode
- 1: enable Timeout mode

**Bits 18-17 TRIGEN:** External trigger enable and polarity selection.

- 00: Software trigger
- 01: External trigger rising edge
- 10: External trigger falling edge
- 11: External trigger double-edge

**Bit 16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-13 TRIGSEL:** External trigger source selection.

- 000: lptim\_ext\_trig0
- 001: lptim\_ext\_trig1
- 010: lptim\_ext\_trig2
- 011: lptim\_ext\_trig3
- 100: lptim\_ext\_trig4
- 101: lptim\_ext\_trig5
- 110: lptim\_ext\_trig6
- 111: lptim\_ext\_trig7

**Bit 12 RESERVED:** Must be kept, and cannot be modified.

**Bits 11-9 PRESC:** clock frequency division.

- 000: /1
- 001: /2
- 010: /4
- 011: /8
- 100: /16
- 101: /32
- 110: /64
- 111: /128

**Bit 8 TRGLT\_ENABLE:** To enable the trigger input filter, the filter length must be configured first and then enabled.

- 0: disable trigger input filter
- 1: enable trigger input filter

**Bits 7-6 TRGFLT:** Trigger input filter configuration.

- 00: no action
- 01: Enable filter, filter length N=2
- 10: Enable filter, filter length N=4
- 11: Enable filter, filter length N=8

**Bit5 CKFLT\_ENABLE:** Enable the external clock filter. The filter length must be configured first and then enabled.

- 0: disable external clock filter
- 1: enable external clock filter

**Bits 4-3 CKFLT:** External clock filter configuration.

- 00: no action
- 01: enable filter, filter length N=2
- 10: enable filter, filter length N=4
- 11: enable filter, filter length N=8

**Bits 2-1 CKPOL:** Encoder mode control.

- 00: select Encoder mode 1, rising edge counting
- 01: select Encoder mode 2, falling edge counting
- 10: select Encoder mode 3, double-edge counting
- 11: reserved

**Bit 0 RESERVED:** Must be kept, and cannot be modified.

## 20.15.5 LPTIM\_CR

Offset: 0x010

Reset Value: 0x00000000

31-3	2	1	0
RESERVED	CNTSTRT	SNGSTRT	ENABLE
rw-0h	rw-0h	rw-0h	rw-0h

**Bits 31-3 RESERVED:** Must be kept, and cannot be modified.

**Bit 2 CNTSTRT:** Continuous counting mode Enable.

- 0: disable
- 1: Enable continuous counting mode. Write 1 to start continuous counting mode. If SNGSTRT is set during continuous counting mode, the counting will stop when ARR is counted next time (switch to single counting mode). This bit needs to be modified after ENABLE is set.

**Bit 1 SNGSTRT:** Single counting mode Enable.

- 0: disable
- 1: Enable single count mode. Write 1 to start single count mode. If CNTSTRT is set during single count mode, it will continue counting when ARR is counted next time (switch to continuous count mode). This bit needs to be modified after ENABLE is set.

**Bit 0 ENABLE:** LPTIMER Enable.

- 0: disable LPTIMER
- 1: enable LPTIMER

### 20.15.6 LPTIM\_CMP

Offset: 0x014

Reset Value: 0x00000000

31-16	15-0
RESERVED	CMP
rw-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 CMP:** Comparison value, which can only be modified after setting Bits in ENABLE of Register LPTIM\_CR.

### 20.15.7 LPTIM\_ARR

Offset: 0x018

Reset Value: 0x00000001

31-16	15-0
RESERVED	ARR
rw-0h	rw-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 ARR:** Reload value, which can only be modified after the ENABLE bits of Register LPTIM\_CR are set.

### 20.15.8 LPTIM\_CNT

Offset: 0x01c

Reset Value: 0x00000000

31-16	15-0
RESERVED	CNT
r-0h	r-0h

**Bits 31-16 RESERVED:** Must be kept, and cannot be modified.

**Bits 15-0 CNT:** The counting result; is valid only when the reading results are consistent twice.

## 20.15.9 LPTIM\_CSR

Offset: 0x020

Reset Value 0x00000001

31-5	4	3	2	1	0
RESERVED	DOWN_CLR_DONE	UP_CLR_DONE	EXTTRIG_CLR_DONE	ARRM_CLR_DONE	CMPM_CLR_DONE
r-0h	r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 31-5 RESERVED:** Must be kept, and cannot be modified.

**Bit 4 DOWN\_CLR\_DONE:** DOWN clearing completed.

- 0: clearing DOWN flag
- 1: clear successfully

**Bit 3 UP\_CLR\_DONE:** UP clearing completed.

- 0: clearing UP flag
- 1: clear successfully

**Bit 2 EXTTRIG\_CLR\_DONE:** EXTTRIG clearing completed.

- 0: clearing EXTTRIG flag
- 1: clear successfully

**Bit 1 ARRM\_CLR\_DONE:** ARRM clearing completed.

- 0: clearing ARRM flag
- 1: clear successfully

**Bit 0 CMPM\_CLR\_DONE:** CMPM clearing completed.

- 0: clearing CMPM flag
- 1: clear successfully

## 20.15.10 LPTIM\_SR1

Offset: 0x024

Reset Value: 0x00000000

31-7	6	5	4	3	2	1	0
RESERVED	DOWN	UP	ARROK	CMPOK	EXTTRIG	ARRM	CMPM
r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 31-7 RESERVED:** Must be kept, and cannot be modified.

**Bit 6 DOWN:** The counting direction changes from upward to downward in encoding mode.

- 0: counting direction does not change upward top to downward
- 1: counting direction changes from upward to downward

**Bit 5 UP:** The counting direction changes from downward to upward in encoding mode.

- 0: counting direction does not change from downward to upward
- 1: counting direction changes from downward to upward

**Bit 4 ARROK:** ARR value loading status.

- 0: not loaded
- 1: loading completed

**Bit 3 CMPOK:** CMP value loading status.

- 0: not loaded
- 1: loading completed

**Bit 2 EXTTRIG:** Valid trigger edge detection.

- 0: no valid trigger edge detected
- 1: valid trigger edge is detected.

**Bit 1 ARRM:** Counter value reaching ARR value detection.

- 0: counter value has not reached ARR
- 1: 1: counter value reaches ARR

**Bit 0 CMPM:** counter value matching CMP value detection.

- 0: counter value does not match CMP value
- 1: counter value matches CMP value

# 21.

# DMA

## 21.1 Introduction

DMA supports four data transfer methods: peripheral to peripheral, peripheral to memory, memory to peripheral, and memory to memory. It supports data width of 8 bits, 16 bits or 32 bits, and supports data auto-reloading and data chain table (LLI). There are two DMAs, named DMA0 and DMA1. Each DMA has 4 channels. Two DMAs are independent of each other and can work simultaneously. 4 channels in each DMA are also independent of each other and can run simultaneously.

## 21.2 Main features

- Configurable transfer data length
- Configurable data transfer methods
- Auto-reloading
- LLI

## 21.3 Transfer data length configuration

DMA can transmit data of multiple blocks. When transmitting the data of each block, it is first transmitted in burst mode. If there is data that is not long enough for burst, it is then sent in single mode. The data transmission of peripherals is as follows:

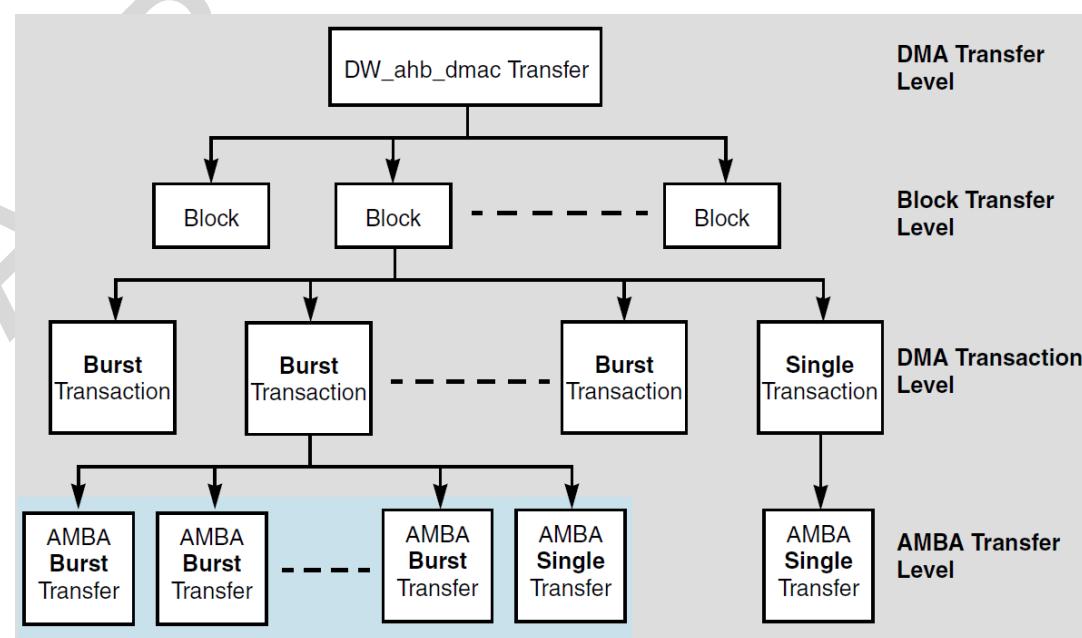


Figure 21-1 Data transfer

The source and destination data widths of DMA are configured through the SRC\_TR\_WIDTH and DST\_TR\_WIDTH bits of the DMA\_CTLx register (x is 0, 1, 2, or 3). A value of 000 for this bit field indicates 8 bits, 001 for 16 bits, and 002 for 32 bits.

The source and destination burst data lengths of DMA are configured through the SRC\_MSIZEx and DEST\_MSIZEx bits of the DMA\_CTLx register. When the bit value is 000, it means 1, 001 means 4, and 002 means 8. Then the conversion to Bytes is SRC\_MSIZEx (DEST\_MSIZEx) \* (number of bits of data width / 8). The burst data length Bytes of DMA needs to be consistent with the input or output FIFO length of the peripheral, otherwise data loss may occur.

The block size of DMA is configured by the BLOCK\_TS bit of the DMA\_CTLx register, which is up to 12 bits. The maximum block size is 4095, which is converted to Bytes as BLOCK\_TS \* (number of bits of data width / 8).

## 21.4 Data transfer methods

DMA supports four data transfer modes: peripheral to peripheral, peripheral to memory, memory to peripheral, and memory to memory. Peripheral to peripheral means that both the source and destination of the data are peripherals; peripheral to memory means that the source is the peripheral and the destination is memory; memory to peripheral means that the source is memory and the destination is the peripheral; memory to memory means that both the source and destination are memory.

The data transfer mode is configured through the TT\_FC bit field of the DMA\_CTLx register. Except for the memory to memory transfer mode, the other modes need to configure the handshake signal between the peripheral and DMA. The handshake value of the peripheral is shown in the following table:

**Table 21-1 Handshake value**

Handshake value	Peripheral signal	Peripheral signal description
4	lorac_tx	LORA tx
5	lorac_rx	LORA rx
6	daccctrl	DAC
7	adccctrl	ADC
10	i2c2_tx	I2C2 tx
11	i2c2_rx	I2C2 rx
12	i2c1_tx	I2C1 tx
13	i2c1_rx	I2C1 rx
14	i2c0_tx	I2C0 tx
15	i2c0_rx	I2C0 rx
16	ssp2_tx	SSP2 tx
17	ssp2_rx	SSP2 rx
18	ssp1_tx	SSP1 tx
19	ssp1_rx	SSP1 rx
20	ssp0_tx	SSP0 tx
21	ssp0_rx	SSP0 rx
22	lpuart_tx	LPUART tx
23	lpuart_rx	LPUART rx

Handshake value	Peripheral signal	Peripheral signal description
24	uart3_tx	UART3 tx
25	uart3_rx	UART3 rx
26	uart2_tx	UART2 tx
27	uart2_rx	UART2 rx
28	uart1_tx	UART1 tx
29	uart1_rx	UART1 rx
30	uart0_tx	UART0 tx
31	uart0_rx	UART0 rx
32	gptim0_ch3	GPTIMER0 channel 3
33	gptim0_ch2	GPTIMER0 channel 2
34	gptim0_ch1	GPTIMER0 channel 1
35	gptim0_ch0	GPTIMER0 channel 0
36	gptim0_trg	GPTIMER0 trigger
37	gptim0_up	GPTIMER0 update
38	gptim1_ch3	GPTIMER1 channel 3
39	gptim1_ch2	GPTIMER1 channel 2
40	gptim1_ch1	GPTIMER1 channel 1
41	gptim1_ch0	GPTIMER1 channel 0
42	gptim1_trg	GPTIMER1 trigger
43	gptim1_up	GPTIMER1 update
44	gptim2_ch1	GPTIMER2 channel 1
45	gptim2_ch0	GPTIMER2 channel 0
46	gptim2_trg	GPTIMER2 trigger
47	gptim2_up	GPTIMER2 update
48	gptim3_ch1	GPTIMER3 channel 1
49	gptim3_ch0	GPTIMER3 channel 0
50	gptim3_trg	GPTIMER3 trigger
51	gptim3_up	GPTIMER3 update
52	basictim1_up	BSTIMER1 update
53	basictim0_up	BSTIMER0 update

## 21.5 LLI

When there are multiple pieces of discontinuous memory data that need to be moved to peripherals or memory, LLI (chain table method) can be used, as shown in the following figure:

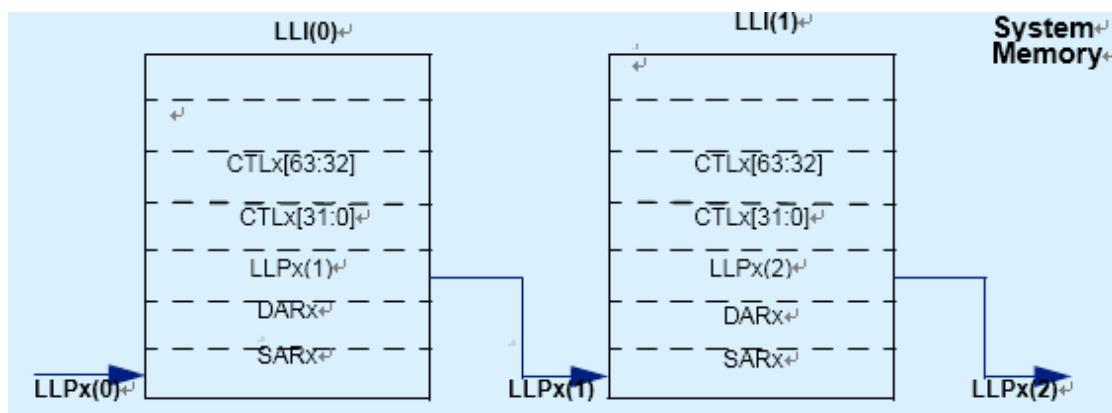


Figure 21-2 LLI chain table

LLI(0), LLI(1) Table indicates the information of configuring block0 and block1, including source and destination address, data bits width, burst length and block length. LLPx Table indicates that the current block points to the address of the next block, and the address of the first block The LLP points to the address of the second block, which is the first address of LLI(1), and so on. The LLP of the last block is 0. The length of each block can be different, and the first address of the memory is also different.

## 21.6 Auto-reloading

Auto-reloading means that after all the data in the memory in the block is moved or written, the data is moved or written again from the starting address of the memory, and the cycle repeats until the DMA channel used is disabled. The Auto-reloading function can be used for both the source and destination of DMA, as long as it is memory.

## 21.7 Interrupts

DMA interrupt signals:

**Table 21-2 DMA interrupts**

Interrupt	Description
DMA block transfer completed interrupt	Interrupt generated after completion of DMA block transfer
DMA destination processing completed interrupt	Interrupt generated after completion of DMA destination processing
DMA source processing completed interrupt	Interrupt generated after DMA source processing is completed
DMA transfer error interrupt	Interrupt generated if error occurs during DMA transfer
DMA full transfer completed interrupt	Interrupt generated after DMA completes transfer

Enable the above interrupt by configuring *DMA\_MaskBlock*, *DMA\_MaskDstTran*, *DMA\_MaskSrcTran*, *DMA\_MaskErr* and *DMA\_MaskTfr* Register.

The status of all interrupts can be obtained through *DMA\_StatusBlock*, *DMA\_StatusDstTran*, *DMA\_StatusSrcTran*, *DMA\_StatusErr* and *DMA\_StatusTfr* Register.

Clear the interrupt status by configuring *DMA\_ClearBlock*, *DMA\_ClearDstTran*, *DMA\_ClearSrcTran*, *DMA\_ClearErr* and *DMA\_ClearTfr* Register.

## 21.8 DMA registers

DMA0 Base Address: 0x40023000

DMA1 Base Address: 0x40024000

**Table 21-3 DMA Registers Summary**

Register	Offset	Description
DMA_SARx	0x000	Source address Register, x represents channel 0, 1, 2, 3, correspond to offset 0x000, 0x058, 0x0B0, 0x108 respectively
DMA_DARx	0x008	Destination address Register, x represents channel 0, 1, 2, 3, correspond to offset 0x008, 0x060, 0x0B8, 0x110 respectively
DMA_LLPx	0x010	Chain Table pointer Register, x represents channel 0, 1, 2, 3, correspond to offset 0x010, 0x068, 0x0C0, 0x118 respectively
DMA_CTLx	0x018	Channel Control Register, x represents channel 0, 1, 2, 3, correspond to offset 0x018, 0x070, 0x0C8, 0x120 respectively
DMA_CFGx	0x040	Channel configuration register, x represents channel 0, 1, 2, 3, correspond to offset 0x040, 0x098, 0x0F0, 0x148 respectively
DMA_StatusTfr	0x2E8	DMA complete transfer completed Interrupt Status Register
DMA_StatusBlock	0x2F0	DMA block transfer completed Interrupt Status Register
DMA_StatusSrcTran	0x2F8	DMA source processing completed Interrupt Status Register
DMA_MaskDstTran	0x300	DMA destination processing completed Interrupt Status Register
DMA_StatusErr	0x308	DMA transfer error Interrupt Status Register
DMA_MaskTfr	0x310	DMA full transfer completed Interrupt Enable Register
DMA_MaskBlock	0x318	DMA block transfer completed Interrupt Enable Register
DMA_MaskSrcTran	0x320	DMA source processing completed Interrupt Enable Register
DMA_MaskDstTran	0x328	DMA destination processing completed Interrupt Enable Register
DMA_MaskErr	0x330	DMA transfer error Interrupt Enable Register
DMA_ClearTfr	0x338	DMA complete transfer completed Interrupt Status Clear Register
DMA_ClearBlock	0x340	DMA block transfer completed Interrupt Status Clear Register
DMA_ClearSrcTran	0x348	DMA source processing completes interrupt Status Clear Register
DMA_ClearDstTran	0x350	DMA destination processing completes interrupt Status Clear Register
DMA_ClearErr	0x358	DMA transfer error interrupt Status Clear Register
DMA_DmaCfgReg	0x398	DMA Enable Register
DMA_ChEnReg	0x3A0	DMA channel Enable Register

### 21.8.1 DMA\_SARx

Offset: 0x000, 0x058, 0x0B0, 0x108

Reset Value: 0x0000000000000000

63-32	31-0
RESERVED	SAR
r-0h	rw-0h

**Bits 63-32 RESERVED:** Must be kept, and cannot be modified.

**Bits 31-0 SAR:** DMA source address.

### 21.8.2 DMA\_DARx

Offset: 0x008, 0x060, 0x0B8, 0x110

Reset Value: 0x0000000000000000

63-32	31-0
RESERVED	DAR
r-0h	rw-0h

**Bits 63-32 RESERVED:** Must be kept, and cannot be modified.

**Bits 31-0 DAR:** DMA destination address.

### 21.8.3 DMA\_LLPx

Offset: 0x010, 0x068, 0x0C0, 0x118

Reset Value: 0x0000000000000000

63-32	31-0
RESERVED	LOC
r-0h	rw-0h

**Bits 63-32 RESERVED:** Must be kept, and cannot be modified.

**Bits 31-0 LOC:** The first address of the next LLI chain Table.

### 21.8.4 DMA\_CTLx

Offset: 0x018, 0x070, 0x0C8, 0x120

Reset Value: 0x0000000200308801

63-45	44	43-32	31-29	28
RESERVED	DONE	BLOCK_TS	RESERVED	LLP_SRC_EN
r-0h	rw-0h	rw-2h	rw-0h	rw-0h
<b>27</b>	<b>26-25</b>	<b>24-23</b>	<b>22-20</b>	<b>19</b>
LLP_DST_EN	SMS	DMS	TT_FC	RESERVED
rw-0h	rw-0h	rw-0h	rw-3h	rw-0h
<b>18</b>	<b>17</b>	<b>16-14</b>	<b>13-11</b>	<b>10-9</b>
DST_SCATTER_EN	SRC_GATHER_EN	SRC_MSIZE	DEST_MSIZE	SINC
rw-0h	rw-0h	rw-1h	rw-1h	rw-0h
<b>8-7</b>	<b>6-4</b>	<b>3-1</b>		<b>0</b>
DINC	SRC_TR_WIDTH	DST_TR_WIDTH		INT_EN
rw-0h	rw-0h	rw-0h		rw-1h

**Bits 63-45 RESERVED:** Must be kept, and cannot be modified.

**Bit 44 DONE:** LLI chain Table block transfer status.

- 0: completed
- 1: uncompleted

**Bits 43-32 BLOCK\_TS:** block length.

**Bits 31-29 RESERVED:** Must be kept, and cannot be modified.

**Bit 28 LLP\_SRC\_EN:** DMA source LLI chain Table enable.

- 0: disabled
- 1: enabled

**Bit 27 LLP\_DST\_EN:** DMA destination LLI chain Table enable.

- 0: disabled
- 1: enabled

**Bits 26-25 SMS:** DMA source AHB master selection.

- 00: AHB master 1
- 01: AHB master 2
- 10: AHB master 3
- 11: AHB master 4

**Bits 24-23 SMS:** DMA destination AHB master selection.

- 00: AHB master 1
- 01: AHB master 2
- 10: AHB master 3
- 11: AHB master 4

**Bits 22-20 TT\_FC:** DMA data transfer mode selection.

- 000: memory to memory
- 001: memory to peripheral
- 010: peripheral to memory
- 011: peripheral-to-peripheral
- other values: invalid

**Bit 19 RESERVED:** Must be kept, and cannot be modified.

**Bit 18 DST\_SCATTER\_EN:** DMA destination scatter enable.

- 0: disabled
- 1: enabled

**Bit 17 SRC\_GATHER\_EN:** DMA source gather enable.

- 0: disabled
- 1: enabled

**Bits 16-14 SRC\_MSIZEx:** DMA source burst length.

- 000: 1
- 001: 4
- 010: 8
- other values: invalid

**Bits 13-11 DEST\_MSIZEx:** DMA destination burst length.

- 000: 1
- 001: 4
- 010: 8
- other values: invalid

**Bits 10-9 SINC:** DMA source address control.

- 00: increment
- 01: decrement
- 10: no change
- 10: no change

**Bits 8-7 DINC:** DMA destination address control.

- 00: increment
- 01: decrement
- 10: no change
- 10: no change

**Bits 6-4 SRC\_TR\_WIDTH:** DMA source data width configuration.

- 000: 8 bits
- 001: 16 bits
- 010: 32 bits
- other values: invalid

**Bits 3-1 DST\_TR\_WIDTH:** DMA destination data width configuration.

- 000: 8 bits
- 001: 16 bits

- 010: 32 bits
- other values: invalid

**Bit 0 INT\_EN:** DMA interrupt enable.

- 0: disable
- 1: enable

### 21.8.5 DMA\_CFGx

Offset: 0x040, 0x098, 0x0F0, 0x148

Reset Value: 0x0000000400020E00

63-47	46-43	42-39	38	37	36-34
RESERVED	DEST_PER	SRC_PER	SS_UPD_EN	DS_UPD_EN	PROTCTL
r-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-1h
<b>33</b>	<b>32</b>	<b>31</b>	<b>30</b>	<b>29-20</b>	<b>19</b>
FIFO_MODE	FCMODE	RELOAD_DST	RELOAD_SRC	RESERVED	SRC_HS_POL
rw-0h	rw-0h	rw-0h	rw-0h	r-0h	rw-0h
<b>18</b>	<b>17</b>	<b>16</b>	<b>15-14</b>	<b>13-12</b>	<b>11</b>
DST_HS_POL	LOCK_B	LOCK_CH	LOCK_B_L	LOCK_CH_L	HS_SEL_SRC
rw-0h	rw-1h	rw-0h	rw-0h	rw-0h	rw-1h
<b>10</b>	<b>9</b>	<b>8</b>	<b>7-5</b>		<b>4-0</b>
HS_SEL_DST	FIFO_EMPTY	CH_SUSP	CH_PRIOR		RESERVED
rw-1h	r-1h	rw-0h	rw-0h		r-0h

**Bits 63-47 RESERVED:** Must be kept, and cannot be modified.

**Bits 46-43 DEST\_PER:** DMA destination handshake interface, valid values are 0 to 3.

**Bits 42-39 SRC\_PER:** DMA source handshake interface, valid values are 0 to 3.

**Bit 38 SS\_UPD\_EN:** DMA source status update enable.

- 0: disabled
- 1: enabled

**Bit 37 DS\_UPD\_EN:** DMA destination status update enable.

- 0: disabled
- 1: enabled

**Bits 36-34 PROTCTL:** Protection control.

**Bit 33 FIFO\_MODE:** FIFO mode selection.

- 0: full FIFO can be obtained
- 1: only half of the FIFO can be obtained

**Bit 32 FCMODE:** Source flow control mode selection.

- 0: request from the source is processed as soon as it is issued
- 1: request from the source will not be processed until there is a request from the destination

**Bit 31 RELOAD\_DST:** DMA destination auto-reloading enable.

- 0: disabled
- 1: enabled

**Bit 30 RELOAD\_SRC:** DMA source auto-reloading enable.

- 0: disabled
- 1: enabled

**Bits 29-20 RESERVED:** Must be kept, and cannot be modified.

**Bit 18 SRC\_HS\_POL:** DMA destination handshake interface information polarity.

- 0: active high
- 1: active low

**Bit 18 DST\_HS\_POL:** DMA destination handshake interface information polarity.

- 0: active high
- 1: active low

**Bit 17 LOCK\_B:** Bus lock control.

- 0: unlocked
- 1: locked

**Bit 16 LOCK\_CH:** DMA channel lock control.

- 0: unlocked
- 1: locked

**Bits 15-14 LOCK\_B\_L:** Bus lock delay.

- 00: wait until the DMA transfer is completed
- 01: wait until the block transfer is completed
- 10: wait until DMA processing is completed

**Bits 13-12 LOCK\_CH\_L:** DMA channel lock delay.

- 00: wait until the DMA transfer is completed
- 01: wait until the block transfer is completed
- 10: wait until DMA processing is completed

**Bit 11 HS\_SEL\_SRC:** DMA source handshake signal selection.

- 0: hardware handshake
- 1: software handshake

**Bit 9 FIFO\_EMPTY:** DMA Channel FIFO empty indication.

- 0: not empty
- 1: empty

**Bit 8 CH\_SUSP:** DMA channel FIFO suspend indication.

- 0: not suspended
- 1: suspended

**Bits 7-5 CH\_PRIOR:** DMA channel priority configuration, valid values are 0 to 3, 0 is the lowest priority, 3 is the highest priority.

**Bits 4-0 RESERVED:** Must be kept, and cannot be modified.

### 21.8.6 DMA\_StatusTfr

Offset: 0x2E8

Reset Value: 0x0000000000000000

63-4	3	2	1	0
RESERVED	CHAN3_STATUS	CHAN2_STATUS	CHAN1_STATUS	CHAN0_STATUS
r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 63-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 CHAN3\_STATUS:** Transfer completion status of DMA channel 3.

- 0: not completed
- 1: completed

**Bit 2 CHAN2\_STATUS:** Transfer completion status of DMA channel 2.

- 0: not completed
- 1: completed

**Bit 1 CHAN1\_STATUS:** Transfer completion status of DMA channel 1.

- 0: not completed
- 1: completed

**Bit 0 CHAN0\_STATUS:** Transfer completion status of DMA channel 0.

- 0: not completed
- 1: completed

### 21.8.7 DMA\_StatusBlock

Offset: 0x2F0

Reset Value: 0x0000000000000000

63-4	3	2	1	0
RESERVED	CHAN3_STATUS	CHAN2_STATUS	CHAN1_STATUS	CHAN0_STATUS
r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 63-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 CHAN3\_STATUS:** Block transfer completion status of DMA channel 3.

- 0: not completed
- 1: completed

**Bit 2 CHAN2\_STATUS:** Block transfer completion status of DMA channel 2.

- 0: not completed
- 1: completed

**Bit 1 CHAN1\_STATUS:** Block transfer completion status of DMA channel 1.

- 0: not completed
- 1: completed

**Bit 0 CHAN0\_STATUS:** Block transfer completion status of DMA channel 0.

- 0: not completed
- 1: completed

### 21.8.8 DMA\_StatusSrcTran

Offset: 0x2F8

Reset Value: 0x0000000000000000

63-4	3	2	1	0
RESERVED	CHAN3_STATUS	CHAN2_STATUS	CHAN1_STATUS	CHAN0_STATUS
r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 63-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 CHAN3\_STATUS:** Source transfer completion status of DMA channel 3.

- 0: not completed
- 1: completed

**Bit 2 CHAN2\_STATUS:** Source transfer completion status of DMA channel 2.

- 0: not completed
- 1: completed

**Bit 1 CHAN1\_STATUS:** Source transfer completion status of DMA channel 1.

- 0: not completed
- 1: completed

**Bit 0 CHAN0\_STATUS:** Source transfer completion status of DMA channel 0.

- 0: not completed
- 1: completed

### 21.8.9 DMA\_StatusDstTran

Offset: 0x300

Reset Value: 0x0000000000000000

63-4	3	2	1	0
RESERVED	CHAN3_STATUS	CHAN2_STATUS	CHAN1_STATUS	CHAN0_STATUS
r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 63-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 CHAN3\_STATUS:** Destination transfer completion status of DMA channel 3.

- 0: not completed
- 1: completed

**Bit 2 CHAN2\_STATUS:** Destination transfer completion status of DMA channel 2.

- 0: not completed
- 1: completed

**Bit 1 CHAN1\_STATUS:** Destination transfer completion status of DMA channel 1.

- 0: not completed
- 1: completed

**Bit 0 CHAN0\_STATUS:** Destination transfer completion status of DMA channel 0.

- 0: not completed
- 1: completed

### 21.8.10 DMA\_StatusErr

Offset: 0x308

Reset Value: 0x0000000000000000

63-4	3	2	1	0
RESERVED	CHAN3_STATUS	CHAN2_STATUS	CHAN1_STATUS	CHAN0_STATUS
r-0h	r-0h	r-0h	r-0h	r-0h

**Bits 63-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 CHAN3\_STATUS:** Transmission error status of DMA channel 3.

- 0: no error
- 1: error

**Bit 2 CHAN2\_STATUS:** Transmission error status of DMA channel 2.

- 0: no error
- 1: error

**Bit 1 CHAN1\_STATUS:** Transmission error status of DMA channel 1.

- 0: no error
- 1: error

**Bit 0 CHAN0\_STATUS:** Transmission error status of DMA channel 0.

- 0: no error
- 1: error

### 21.8.11 DMA\_MaskTfr

Offset:0x310

Reset Value: 0x0000000000000000

63-12	11	10	9	8
RESERVED	INT_MASK_WE_3	INT_MASK_WE_2	INT_MASK_WE_1	INT_MASK_WE_0
r-0h	w-0h	w-0h	w-0h	w-0h
7-4	3	2	1	0
RESERVED	INT_MASK_3	INT_MASK_2	INT_MASK_1	INT_MASK_0
r-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 63-12 RESERVED:** Must be kept, and cannot be modified.

**Bit 11 INT\_MASK\_WE\_3:** DMA channel 3 transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 10 INT\_MASK\_WE\_2:** DMA channel 2 transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 8 INT\_MASK\_WE\_1:** DMA channel 1 transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 8 INT\_MASK\_WE\_0:** DMA channel 0 transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bits 7-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 INT\_MASK\_3:** DMA channel 3 transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 2 INT\_MASK\_2:** DMA channel 2 transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 1 INT\_MASK\_1:** DMA channel 1 transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 0 INT\_MASK\_0:** DMA channel 0 transfer completion interrupt enable.

- 0: disabled
- 1: enabled

## 21.8.12 DMA\_MaskBlock

Offset: 0x318

Reset Value: 0x0000000000000000

63-12	11	10	9	8
RESERVED	INT_MASK_WE_3	INT_MASK_WE_2	INT_MASK_WE_1	INT_MASK_WE_0
r-0h	w-0h	w-0h	w-0h	w-0h
7-4	3	2	1	0
RESERVED	INT_MASK_3	INT_MASK_2	INT_MASK_1	INT_MASK_0
r-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 63-12 RESERVED:** Must be kept, and cannot be modified.

**Bit 11 INT\_MASK\_WE\_3:** DMA channel 3 block transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 10 INT\_MASK\_WE\_2:** DMA channel 2 block transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 9 INT\_MASK\_WE\_1:** DMA channel 1 block transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 8 INT\_MASK\_WE\_0:** DMA channel 0 block transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bits 7-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 INT\_MASK\_3:** DMA channel 3 block transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 2 INT\_MASK\_2:** DMA channel 2 block transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 1 INT\_MASK\_1:** DMA channel 1 block transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 0 INT\_MASK\_0:** DMA channel 0 block transfer completion interrupt enable.

- 0: disabled
- 1: enabled

### 21.8.13 DMA\_MaskSrcTran

Offset:0x320

Reset Value: 0x0000000000000000

63-12	11	10	9	8
RESERVED	INT_MASK_WE_3	INT_MASK_WE_2	INT_MASK_WE_1	INT_MASK_WE_0
r-0h	w-0h	w-0h	w-0h	w-0h
7-4	3	2	1	0
RESERVED	INT_MASK_3	INT_MASK_2	INT_MASK_1	INT_MASK_0
r-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 63-12 RESERVED:** Must be kept, and cannot be modified.

**Bit 11 INT\_MASK\_WE\_3:** DMA channel 3 destination transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 10 INT\_MASK\_WE\_2:** DMA channel 2 destination transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 9 INT\_MASK\_WE\_1:** DMA channel 1 destination transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 8 INT\_MASK\_WE\_0:** DMA channel 0 destination transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bits 7-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 INT\_MASK\_3:** DMA channel 3 destination transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 2 INT\_MASK\_2:** DMA channel 2 destination transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 1 INT\_MASK\_1:** DMA channel 1 destination transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 0 INT\_MASK\_0:** DMA channel 0 destination transfer completion interrupt enable.

- 0: disabled
- 1: enabled

### 21.8.14 DMA\_MaskDstTran

Offset: 0x328

Reset Value: 0x0000000000000000

63-12	11	10	9	8
RESERVED	INT_MASK_WE_3	INT_MASK_WE_2	INT_MASK_WE_1	INT_MASK_WE_0
r-0h	w-0h	w-0h	w-0h	w-0h
7-4	3	2	1	0
RESERVED	INT_MASK_3	INT_MASK_2	INT_MASK_1	INT_MASK_0
r-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 63-12 RESERVED:** Must be kept, and cannot be modified.

**Bit 11 INT\_MASK\_WE\_3:** DMA channel 3 destination transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 10 INT\_MASK\_WE\_2:** DMA channel 2 destination transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 9 INT\_MASK\_WE\_1:** DMA channel 1 destination transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 8 INT\_MASK\_WE\_0:** DMA channel 0 destination transfer completion interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bits 7-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 INT\_MASK\_3:** DMA channel 3 destination transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 2 INT\_MASK\_2:** DMA channel 2 destination transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 1 INT\_MASK\_1:** DMA channel 1 destination transfer completion interrupt enable.

- 0: disabled
- 1: enabled

**Bit 0 INT\_MASK\_0:** DMA channel 0 destination transfer completion interrupt enable.

- 0: disabled
- 1: enabled

### 21.8.15 DMA\_MaskErr

Offset:0x330

Reset Value: 0x0000000000000000

63-12	11	10	9	8
RESERVED	INT_MASK_WE_3	INT_MASK_WE_2	INT_MASK_WE_1	INT_MASK_WE_0
r-0h	w-0h	w-0h	w-0h	w-0h
7-4	3	2	1	0
RESERVED	INT_MASK_3	INT_MASK_2	INT_MASK_1	INT_MASK_0
r-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 63-12 RESERVED:** Must be kept, and cannot be modified.

**位 11 INT\_MASK\_WE\_3:** Channel 3 transfer error interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 10 INT\_MASK\_WE\_2:** Channel 2 transfer error interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 9 INT\_MASK\_WE\_1:** Channel 1 transfer error interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bit 8 INT\_MASK\_WE\_0:** Channel 0 transfer error interrupt mask write enable.

- 0: disabled
- 1: enabled

**Bits 7-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 INT\_MASK\_3:** Channel 3 transfer error interrupt enable.

- 0: disabled
- 1: enabled

**Bit 2 INT\_MASK\_2:** Channel 2 transfer error interrupt enable.

- 0: disabled
- 1: enabled

**Bit 1 INT\_MASK\_1:** Channel 1 transfer error interrupt enable.

- 0: disabled
- 1: enabled

**Bit 0 INT\_MASK\_0:** Channel 0 transfer error interrupt enable.

- 0: disabled
- 1: enabled

### 21.8.16 DMA\_ClearTfr

Offset: 0x338

Reset Value: 0x0000000000000000

63-4	3	2	1	0
RESERVED	CHAN3_CLEAR	CHAN2_CLEAR	CHAN1_CLEAR	CHAN0_CLEAR
r-0h	w-0h	w-0h	w-0h	w-0h

**Bits 63-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 CHAN3\_CLEAR:** Clear DMA channel 3 transfer completion status.

- 0: no action
- 1: clear

**Bit 2 CHAN2\_CLEAR:** Clear DMA channel 2 transfer completion status.

- 0: no action
- 1: clear

**Bit 1 CHAN1\_CLEAR:** Clear DMA channel 1 transfer completion status.

- 0: no action
- 1: clear

**Bit 0 CHAN0\_CLEAR:** Clear DMA channel 0 transfer completion status.

- 0: no action
- 1: clear

### 21.8.17 DMA\_ClearBlock

Offset: 0x340

Reset Value: 0x0000000000000000

63-4	3	2	1	0
RESERVED	CHAN3_CLEAR	CHAN2_CLEAR	CHAN1_CLEAR	CHAN0_CLEAR
r-0h	w-0h	w-0h	w-0h	w-0h

**Bits 63-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 CHAN3\_CLEAR:** Clear DMA channel 3 block transfer completion status.

- 0: no action
- 1: clear

**Bit 2 CHAN2\_CLEAR:** Clear DMA channel 2 block transfer completion status.

- 0: no action
- 1: clear

**Bit 1 CHAN1\_CLEAR:** Clear DMA channel 1 block transfer completion status.

- 0: no action
- 1: clear

**Bit 0 CHAN0\_CLEAR:** Clear DMA channel 0 block transfer completion status.

- 0: no action
- 1: clear

### 21.8.18 DMA\_ClearSrcTran

Offset: 0x348

Reset Value: 0x0000000000000000

63-4	3	2	1	0
RESERVED	CHAN3_CLEAR	CHAN2_CLEAR	CHAN1_CLEAR	CHAN0_CLEAR
r-0h	w-0h	w-0h	w-0h	w-0h

**Bits 63-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 CHAN3\_CLEAR:** Clear DMA channel 3 source transfer completion status.

- 0: no action
- 1: clear

**Bit 2 CHAN2\_CLEAR:** Clear DMA channel 2 source transfer completion status.

- 0: no action
- 1: clear

**Bit 1 CHAN1\_CLEAR:** Clear DMA channel 1 source transfer completion status.

- 0: no action
- 1: clear

**Bit 0 CHAN0\_CLEAR:** Clear DMA channel 0 source transfer completion status.

- 0: no action
- 1: clear

### 21.8.19 DMA\_ClearDstTran

Offset: 0x350

Reset Value: 0x0000000000000000

63-4	3	2	1	0
RESERVED	CHAN3_CLEAR	CHAN2_CLEAR	CHAN1_CLEAR	CHAN0_CLEAR
r-0h	w-0h	w-0h	w-0h	w-0h

**Bits 63-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 CHAN3\_CLEAR:** Clear DMA channel 3 destination transfer completion status.

- 0: no action
- 1: clear

**Bit 2 CHAN2\_CLEAR:** Clear DMA channel 2 destination transfer completion status.

- 0: no action

- 1: clear

**Bit 1 CHAN1\_CLEAR:** Clear DMA channel 1 destination transfer completion status.

- 0: no action
- 1: clear

**Bit 0 CHAN0\_CLEAR:** Clear DMA channel 0 destination transfer completion status.

- 0: no action
- 1: clear

### 21.8.20 DMA\_ClearErr

Offset: 0x358

Reset Value: 0x0000000000000000

63-4	3	2	1	0
RESERVED	CHAN3_CLEAR	CHAN2_CLEAR	CHAN1_CLEAR	CHAN0_CLEAR
r-0h	w-0h	w-0h	w-0h	w-0h

**Bits 63-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 CHAN3\_CLEAR:** Clear DMA channel 3 transfer error status.

- 0: no action
- 1: clear

**Bit 2 CHAN2\_CLEAR:** Clear DMA channel 2 transfer error status.

- 0: no action
- 1: clear

**Bit 1 CHAN1\_CLEAR:** Clear DMA channel 1 transfer error status.

- 0: no action
- 1: clear

**Bit 0 CHAN0\_CLEAR:** Clear DMA channel 0 transfer error status.

- 0: no action
- 1: clear

### 21.8.21 DMA\_DmaCfgReg

Offset: 0x398

Reset Value: 0x0000000000000000

63-1	0
RESERVED	DMA_EN
r-0h	rw-0h

**Bits 63-1 RESERVED:** Must be kept, and cannot be modified.

**Bit 0 DMA\_EN:** DMA enable control.

- 0: disabled
- 1: enabled

### 21.8.22 DMA\_ChEnReg

Offset: 0x3A0

Reset Value: 0x0000000000000000

63-12	11	10	9	8
RESERVED	CH_EN_WE_3	CH_EN_WE_2	CH_EN_WE_1	CH_EN_WE_0
r-0h	w-0h	w-0h	w-0h	w-0h
7-4	3	2	1	0
RESERVED	CH_EN_3	CH_EN_2	CH_EN_1	CH_EN_0
r-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 63-12 RESERVED:** Must be kept, and cannot be modified.

**Bit 11 CH\_EN\_WE\_3:** DMA channel 3 enable control information write enable.

- 0: disabled
- 1: enabled

**Bit 10 CH\_EN\_WE\_2:** DMA channel 2 enable control information write enable.

- 0: disabled
- 1: enabled

**Bit 9 CH\_EN\_WE\_1:** DMA channel 1 enable control information write enable.

- 0: disabled
- 1: enabled

**Bit 8 CH\_EN\_WE\_0:** DMA channel 0 enable control information write enable.

- 0: disabled
- 1: enabled

**Bits 7-4 RESERVED:** Must be kept, and cannot be modified.

**Bit 3 CH\_EN\_3:** DMA channel 3 enable control. When the DMA transfer is completed, the hardware automatically disables this channel.

- 0: disabled

- 1: enabled

**Bit 2 CH\_EN\_2:** DMA channel 2 enable control. When the DMA transfer is completed, the hardware automatically disables this channel.

- 0: disabled
- 1: enabled

**Bit 1 CH\_EN\_1:** DMA channel 1 enable control. When the DMA transfer is completed, the hardware automatically disables this channel.

- 0: disabled
- 1: enabled

**Bit 0 CH\_EN\_0:** DMA channel 0 enable control. When the DMA transfer is completed, the hardware automatically disables this channel.

- 0: disabled
- 1: enabled

# 22.

# GPTIMER

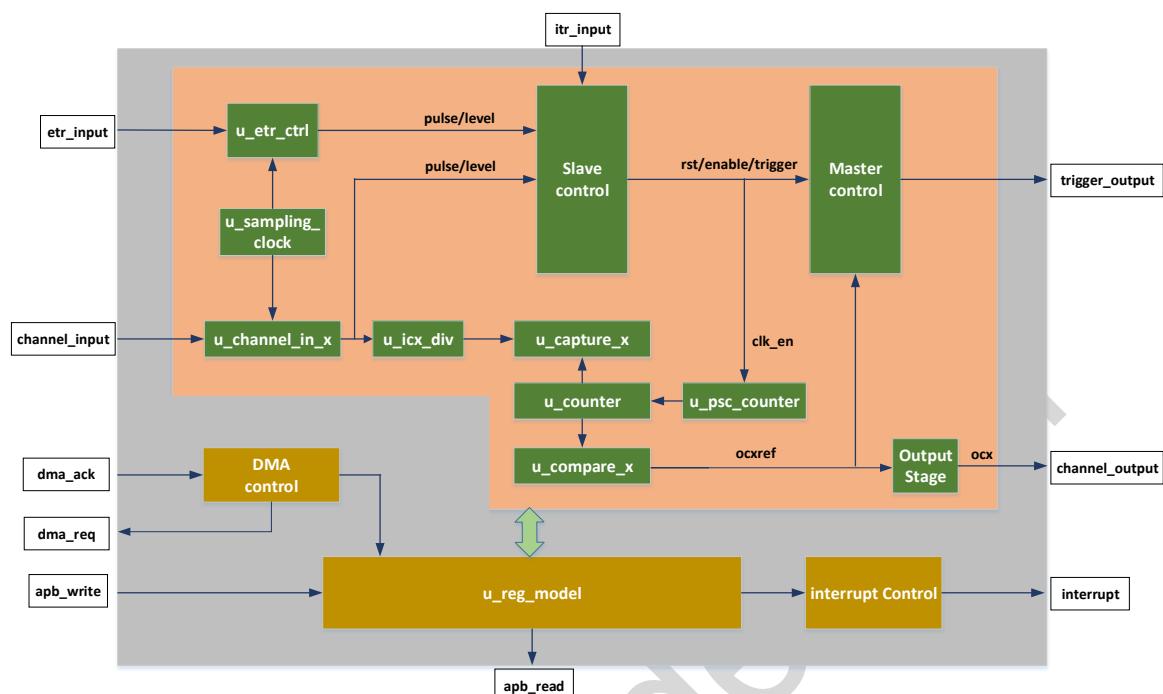
## 22.1 Introduction

ASR6601 has a total of 4 general-purpose timers (GPTIMER), of which GPTIMER0 and GPTIMER1 have 4 channels. GPTIMER2 and GPTIMER3 have 2 channels, that is, GPTIMER2 and GPTIMER3 do not have channels 2 and 3.

GPTIMER contains a 16-bit counter, supports auto-reloading function, and supports up to 16-bit programmable prescaler. The 4 channels can be independently configured as input or output. It supports input capture, output comparison and other functions, counting clock and counting mode. Software configurable, supports connection to Hall devices, supports encoding mode (only applicable to GPTIMER0 and GPTIMER1), supports DMA configuration, has independent interrupt output, supports encoding functions, etc. Based on rich channel configurations and functions, the GPTIMER can be used for timing counting, measuring input pulse width (us-ms level), generating PWM waveforms and other applications.

## 22.2 Main features

- 16-bit counter, supports automatic reloading, configurable edge-aligned (up, down) counting and center-aligned (up/down) counting.
- 16-bit programmable prescaler (division coefficient 1-65535), which can be configured during the counting process.
- Up to 4 independent channels, which can complete input capture, output comparison, PWM waveform output, and single pulse waveform output.
- Supports channel output polarity selection and input edge configuration.
- Supports synchronization with external input or other modules (GPTIMER, ADC, DAC).
- Independent DMA channel, up to 6 groups of DMA Request, including update events, trigger events and 4 groups of channel events (capture, comparison).
- Supports quadrature encoding function.
- Supports external trigger channel input clock for counting and supports external trigger channel input trigger signal, supports channel input clock for counting.
- Supports channel remapping, that is, mapping GPIO signals or internal signals of other modules to channels or external channels.

**GPTIMER structure diagram:**

**Figure 22-1 GPTIMER diagram**
**Table 22-1 GPTIMER module introduction**

Module name	Description
slave control	Slave mode controller
master control	Master mode controller
u_etrx_ctrl	ETR channel control, including polarity, prescaler, filtering and other configurations
u_channel_in_x	Input channel x controls including polarity, filtering and edge configuration
u_icx_div	Input channel x event divider
u_sampling_clock	The sampling clock that generates the filter
u_capture_x	Input channel x capture function
u_compare_x	Output channel x compare function
u_psc_counter	16-bit prescaler
u_counter	16-bit counter
u_reg_model	Register related configuration
output stage	Output control
interrupt control	Interrupt control
dma control	DMA capability
itr_input	Internal trigger channel input
etr_input	External trigger channel input
channel_input	Channel input
dma_ack	DMA acknowledge
dma_req	DMA request
apb_write	APB bus write
apb_read	APB bus read

Module name	Description
apb_read	APB bus read
trigger_output	Master mode signal output is an internal signal and will not be output to the outside
channel_output	Channel output

## 22.3 Counter

GPTIMER's counter has a total of 16-bit, supports upward, downward, and center-aligned counting. The counting clock is optional. Counting can be enabled or disabled by software configuration. The software can read and write at any time (it is recommended not to write during the counting process to avoid unknown errors).

### 22.3.1 Counter clock

GP Timer has four counting clock sources, namely internal clock, external clock mode 1, external clock mode 2 and internal trigger signal control counting. Among them, the internal clock is the default mode ( $SMS==3'b000$ ), the clock comes from RCC, as long as CEN is set, the prescaler and the counter will start counting, the other three cases use the corresponding signal as the counting enable, not as the real clock.

External clock mode 1 ( $SMS==3'b111$ ,  $TS==3'b100/101/110$ ), in this mode, the counter is controlled by the rising edge or falling edge or both edges of the selected channel input as the counter control, for example, if the rising edge of channel 0 is selected to control counting, then each rising edge will increase the counter by 1 (upward counting, no frequency division), and the waveform is as follows:

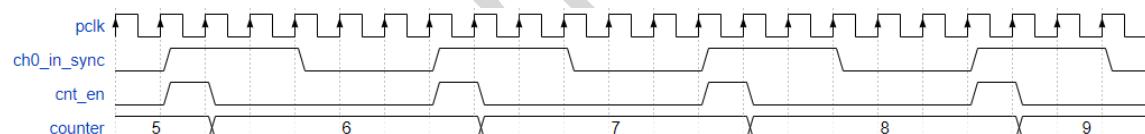


Figure 22-2 External clock mode 1 counting

In external clock mode 2 ( $ECE==1$ ) the counter is controlled by the rising edge or falling edge of ETR as the counting enable of the counter. For example, if the rising edge of ETR is configured, the waveform is as shown in the figure below.

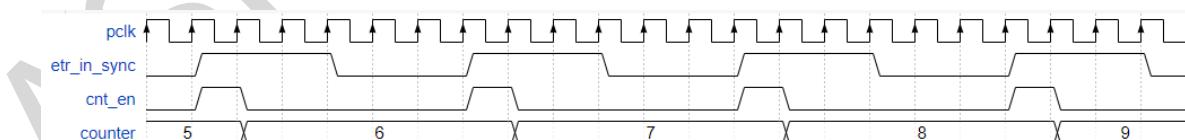
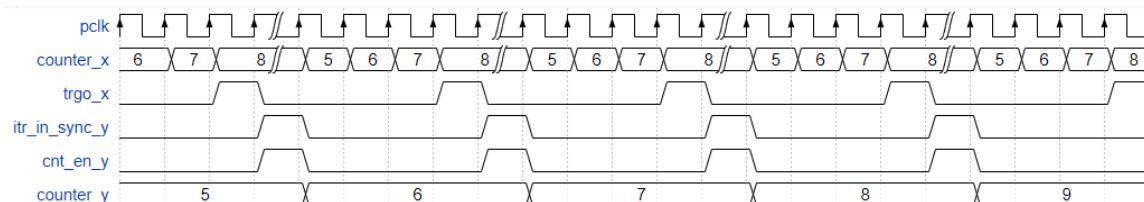


Figure 22-3 External clock mode 2 counting

GPTIMER can also choose the internal trigger signal to control counting ( $SMS==3'b111$ ,  $TS==3'b001/010/011$ ), that is, the trigger output signal of the upper level GPTIMER can be used as the counting clock of the GPTIMER, thereby realizing GPTIMER cascade. In this case, the upper level GPTIMER is equivalent to a frequency division counter, and the waveform is as shown in the figure below.



**Figure 22-4 Internal trigger signal for clock counting**

When ETR is used as the counting clock input, there are two ways to implement it. One is external clock mode 1, configure  $SMS==3'b111$ ,  $TS==3'b111$ , and the other is external clock mode 2, configure  $ECE==1$ .

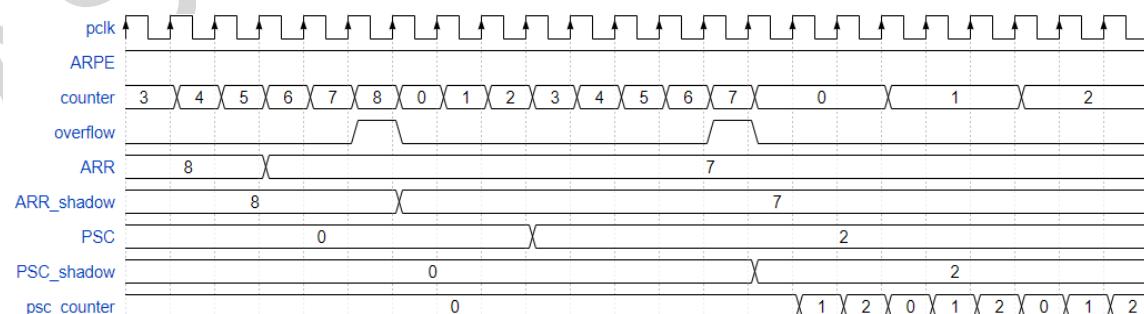
### 22.3.2 Auto-reload

GPTIMER supports the automatic reloading function. When counting upwards, after counting to the reload value (ARR), it will return to zero and start counting again. When counting downwards, it will start counting from ARR. After counting to 0, it will return to ARR and restart counting. When center-aligned counting, the counter starts counting from 0 to ARR-1, and then counts from ARR to 0.

ARR is software configurable (ARPE) whether to enable the shadow Register. If ARPE=0, the Shadow Register is disabled and the value written by the software is synchronously updated to ARR for use by the counter. If ARPE=1, the value written by the software will not be immediately effective, until the update event arrives, the value will be updated to the shadow Register for counter use.

### 22.3.3 Up-count

If configured in up-counting mode, the counter will start incrementing from 0 to ARR after it is enabled and has a counting clock, generating an overflow event (overflow), and then reset to zero and restart counting. If UG is set (software or hardware) during the counting process, the counter including the prescaler will be initialized (reset to zero). In terms of timing, the overflow flag will be generated during the last count value. If the shadow register is enabled, the ARR, PSC, CCRx and other registers will be updated to the corresponding shadow registers at the beginning of the next round of counting. The waveform is shown in the figure below.



**Figure 22-5 Up-counting**

### 22.3.4 Down-count

If configured in down-counting mode, the counter will start to count down from ARR to 0 after it is enabled and there is a counting clock, generating an underflow event (underflow), and then return to ARR to restart counting. If UG is set (software or hardware) during the counting process, the counter including the frequency division counter will be initialized (the counter returns to ARR, and the prescaler returns to zero). In terms of timing, the underflow flag will be generated during the last count value ( $CNT=0$ ), but please note that if the shadow register is enabled, the ARR register will be updated to the corresponding shadow register before the next round of counting starts ( $CNT=0$ ) to ensure that the next round of counting can use the latest load value and frequency division value. PSC and CCRx are the same as before and will be updated to the shadow register at the next clock of underflow. The waveform is shown in the figure below.

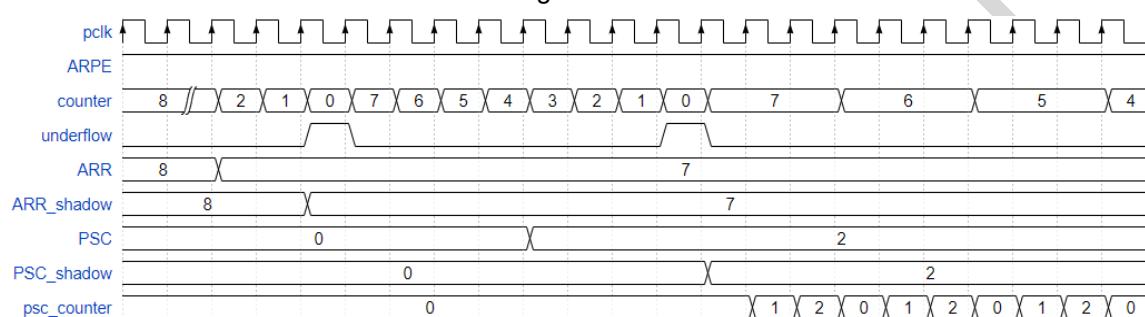


Figure 22-6 Down-counting

### 22.3.5 Center-aligned count

If configured as the center-aligned counting mode, the counter will start incrementing from 0 to ARR-1 after it is enabled and there is a counting clock, generating an overflow event, and then decrementing from ARR to 1, generating an underflow event, and then restarting the counting from 0. If UG is set (software or hardware) during the counting process, the counter including the prescaler will be initialized (reset to zero). Please note that if the shadow register is enabled, the ARR and PSC registers will be updated to the corresponding shadow registers when counting up to the old ARR-1 to ensure that the new ARR and new PSC can be used when counting down, and the update of CCRx is the same as before. When counting down, the shadow registers of ARR, PSC and CCRx will be updated after underflow occurs. In this mode, the counting direction is controlled by hardware, and the software configuration is invalid. See figure below:

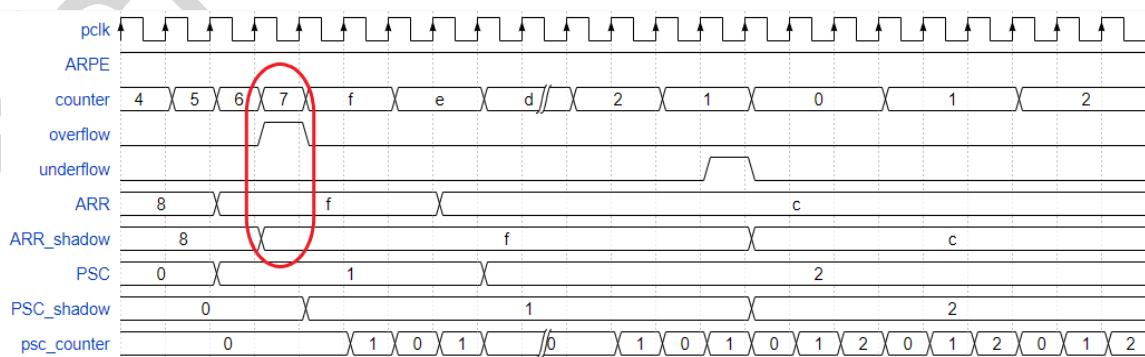


Figure 22-7 Center-aligned counting

## 22.4 Prescaler

GPTIMER supports 16-bit (1~65535) programmable frequency division, which is realized by the prescaler. The count enable signal generated by the upper circuit will be used as the enable control count of the frequency division counter. When the frequency division counter counts to the pre-loaded frequency division value, it outputs a pulse as the count enable of the next level counter, and then the frequency division counter returns to zero and counts again, and so on.

The division value of the prescaler uses the shadow register by default, that is, the software write operation will not take effect immediately, but the new division value will be written to the shadow register until the update event (UG is set, count overflow) arrives, at which time the division value will take effect. The software read operation reads the written register value, not the shadow register. If there are multiple write operations before the update event arrives, the previously written value will be overwritten.

Take an example to illustrate the prescaler. If it is configured to be divided by 4, then 4 high levels must be input to output a valid pulse. The waveform is as shown in the figure below (channel 0, no filtering, selected the rising edge of channel 0 as the valid pulse, and configure ic0 to be divided by 4).

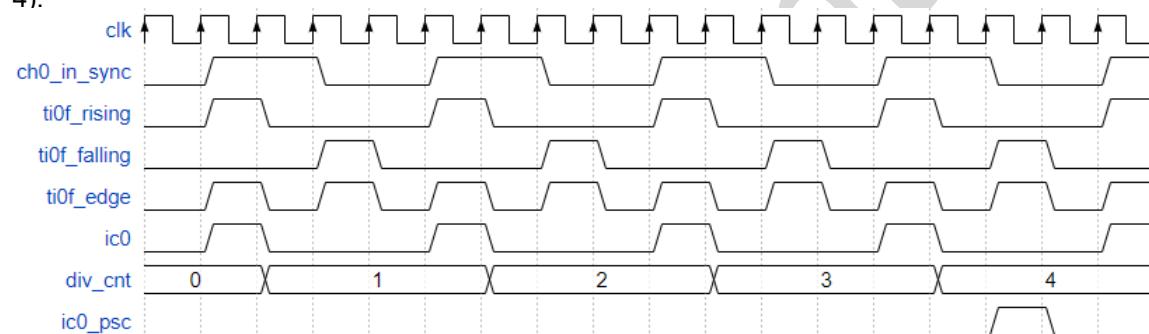


Figure 22-8 Prescaler

## 22.5 Capture mode

Each input channel and external trigger channel can select the digital filtering function, which samples the input signal by using a high-frequency sampling clock (the frequency is at least 4 times that of the input signal). The clock of all flip-flops inside GPTIMER is provided by pclk. The software can configure the frequency of the sampling clock (CKD is pclk, pclk/2, pclk/4 respectively), and the frequency division is achieved by using a counter. For example, if the sampling frequency is configured to be 4 times the frequency of pclk, the counter is controlled by pclk to count, and a pulse (with a width of one cycle of pclk) is generated every 4 pclk cycles, which is used as the enable signal of the subsequent counter. In each channel, the user can also configure the sampling clock division of the digital filter again, that is, configure the value of ETF, and the filtering principle is the same as above.

## 22.6 Troughput

Each channel of each GPTIMER has multiple sources. These signal sources are asynchronous with the GPTIMER, so synchronization processing is required inside the module. The synchronized channel input signal can be filtered according to the software configuration. The sampling frequency and window length of the filter can be configured by software (ICxF). The filtered signal generates an edge signal by an edge detector. The effective level (or effective edge) can be configured by software. The processed channel signal can be used as the control signal of the slave mode controller, the encoding mode input signal, or the input capture enable signal (configurable frequency division). Each input channel can be mapped to the current channel, the adjacent channel or the internal trigger signal TRC (CCxS[1:0] configuration). The specific scheme is shown in the table (taking channel 0 as an example), where ti0fp0 is the input signal mapped to channel 0, and ti1fp0 is the input signal mapped to channel 1.

**Table 22-2 Input channels polarity configuration**

{CCONP, CCOP}	有效脉冲（应用于输入捕获、复位模式、触发模式、外部时钟模式）		有效电平（应用于 Gate 模式、编码模式）
	ti0fp0	ti1fp0	
2'b00	通道 0 上升沿	通道 1 上升沿	通道 0 高电平
2'b01	通道 0 下降沿	通道 1 下降沿	通道 0 低电平
2'b10	保留	保留	保留
2'b11	通道 0 双沿	通道 1 双沿	通道 0 高电平

**Table 22-3 Input channel mapping**

CCxS	i cx 映射
2'b01	tixfpx (x 代表当前通道)
2'b10	tiyfpx (y 代表相邻通道)
2'b11	trc (仅适用于 TS=3'b000、3'b001、3'b010、3'b100)

In addition, channel 0 is different from other channels. Channel 0 can be connected to the XOR output of channel 0, channel 1 and channel 2 through software configuration (TI0S is set). At this time, other functions of the channel are still valid. This function is only applicable to GPTIMER0 and GPTIMER1.

## 22.6.1 Input capture

Input capture is activated only when the channel is configured as input mode and CCxE is set. The capture behavior can be triggered by software (CCxG) or hardware (current channel, adjacent channel or internal interconnect signal). When a valid capture trigger signal is generated, GPTIMER will latch the current counter value into the corresponding CCRx register and set the CCxIF flag bit. If the corresponding interrupt or DMA mask bit is enabled, an interrupt signal or DMA request will be generated. If more than one capture behavior occurs when CCxIF is set (not cleared by software), CCxOF is set to indicate that a capture overflow event has occurred. Reading the CCxR register (or writing 0 to the corresponding bit of the SR register) can clear CCxIF and CCxOF. The waveform is shown in the figure below.

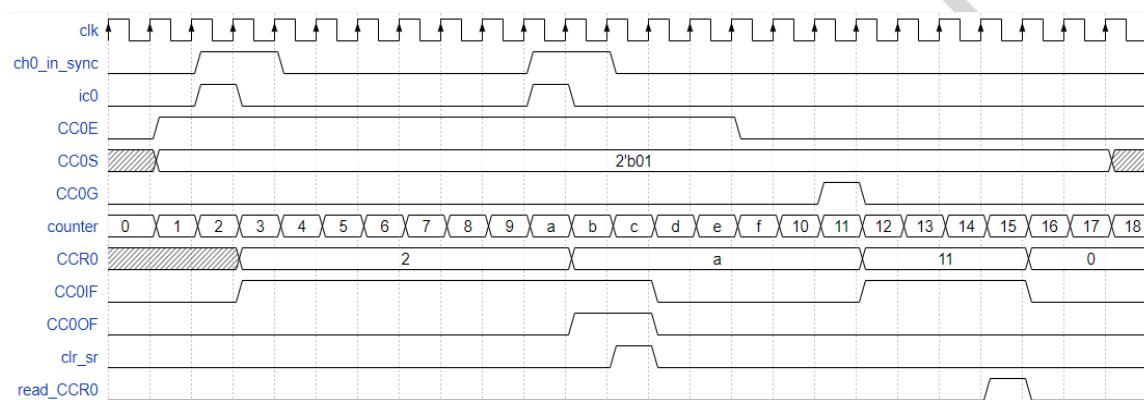


Figure 22-9 Input capture

## 22.6.2 Output compare

The output compare function is activated only when the channel is configured as output mode and CCxE is set. This function controls the channel output high and low flip by comparing the counter value with the CCRx value, thereby outputting a specific waveform.

### 22.6.2.1 CCRx preload

There are two ways to write to the CCRx register. If CCxPE is set, the CCRx value written by the software will not be used directly. The shadow register is the real buffer until the update event occurs. The CCRx value will be updated to the shadow register; if OCxPE is reset, the CCRx value written by the software will be used directly, and the shadow register is disabled.

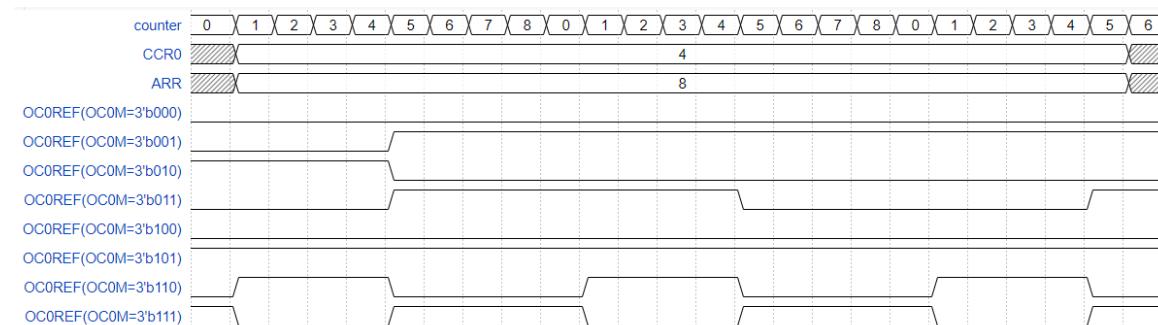
## 22.6.2.2 Output compare modes

When a match (CNT==CCR) occurs, the channel output will flip according to the configured mode, and the CCxIF flag will be set. If the corresponding interrupt or DMA mask bit is enabled, an interrupt or DMA request will be generated. The specific mode control is shown in the following table.

Table 22-4 Output waveform description

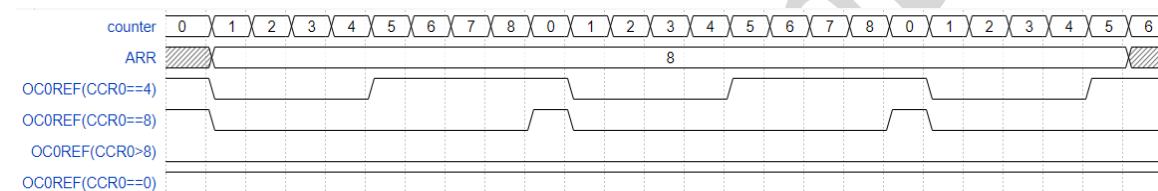
比较模式	计数模式	输出波形
冻结模式	Any	无论 CNT 如何变化, 输出维持不变
SET 模式	Any	在 CNT==CCR 后, 输出高电平
RESET 模式	Any	在 CNT==CCR 后, 输出低电平
TOGGLE 模式	Any	在 CNT==CCR 时, 翻转当前电平
强制 RESET 模式	Any	选择该模式后, 直接输出低电平, 忽略比较结果
强制 SET 模式	Any	选择该模式后, 直接输出高电平, 忽略比较结果
PWM1 模式	向上计数 (边沿对齐 pwm)	CNT<CCR 时, 输出高电平, CNT>=CCR 时, 输出低电平。如果 CCR>ARR, 则输出一直为高电平 (100%PWM), 如果 CCR==0, 则输出一直为低电平 (0%PWM)。
	向下计数 (边沿对齐 pwm)	CNT<=CCR 时, 输出高电平, CNT>CCR 时, 输出低电平。如果 CCR>ARR, 则输出一直为高电平 (100%PWM)。注意: 0%PWM 模式在该情况下不支持。
	中间计数 (中间对齐 pwm)	相当于向上计数与向下计数相结合。如果 CCR>=ARR, 则输出一直为高电平 (100%PWM), 如果 CCR==0, 则输出一直为低电平 (0%PWM)。
PWM2 模式	向上计数 (边沿对齐 pwm)	CNT<CCR 时, 输出低电平, CNT>=CCR 时, 输出高电平。如果 CCR>ARR, 则输出一直为低电平 (0%PWM), 如果 CCR==0, 则输出一直为高电平 (100%PWM)。
	向下计数 (边沿对齐 pwm)	CNT<=CCR 时, 输出低电平, CNT>CCR 时, 输出高电平。如果 CCR>ARR, 则输出一直为低电平 (0%PWM)。注意: 100%PWM 模式在该情况下不支持。
	中间计数 (中间对齐 pwm)	相当于向上计数与向下计数相结合。如果 CCR>=ARR, 则输出一直为低电平 (0%PWM), 如果 CCR==0, 则输出一直为高电平 (100%PWM)。

The output waveforms in each mode are as follows (taking upward counting as an example).



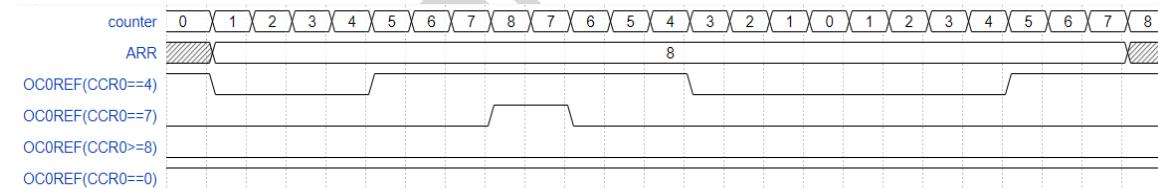
**Figure 22-10 Output compare mode waveforms**

Among them, the PWM mode also supports the output of 0% and 100% waveforms by configuring ARR and CCR. The PWM2 waveform of edge-aligned counting is shown in the figure below.



**Figure 22-11 PWM2 edge-aligned counting**

The PWM2 waveform of center-aligned count is shown in the figure below.



**Figure 22-12 PWM2 center-aligned counting**

### 22.6.2.3 Single pulse fast output

In single pulse mode (OPM is set), the two PWM modes can be configured as fast output mode (OCxFE is set). After enabling fast mode, the output waveform will ignore the comparison results of CNT and CCR, and the level flip will be controlled by the rising edge of the trigger signal (selected according to TS). The output signal level is equivalent to the level after the match event occurs. For example, configure GPTIMER channel 0 as output mode, select PWM1 mode, and select ETR input as the trigger signal. When ETR inputs a high level, channel 0 immediately outputs a high level (when OCxP=0). This function can effectively reduce the delay from the edge of the trigger signal to the waveform output. The single pulse output waveform when fast mode is enabled is as shown below:

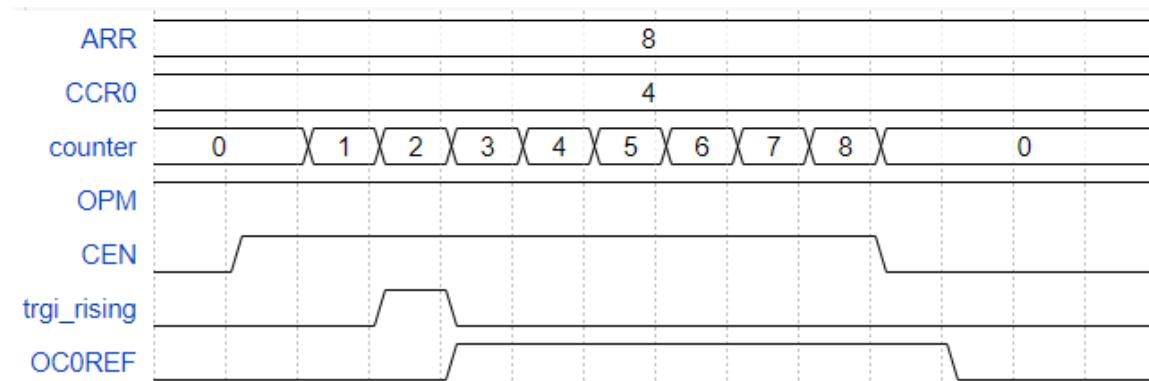


Figure 22-13 Single pulse output waveform in fast mode

#### 22.6.2.4 Brake signal

In addition to being affected by the count value, the output waveform can also be cleared by hardware through an external trigger signal (ETR). To use this function, you need to enable the OCxCE bit in advance, and at the same time ensure that ETR disables frequency division (ETP=2'b00), and ETR cannot be used as a count clock. After enabling this function (OCxCE=1), when the level of ETR is valid (default high level), the channel output will be cleared. When changing the active level of ETR, it is achieved by configuring ETP. After turning off this function (OCxCE=0), the channel output will not be restored immediately, but will wait until the next count cycle starts to restore normal output. The comparison waveform of turning on and off the external trigger signal to clear the channel output function is as follows:

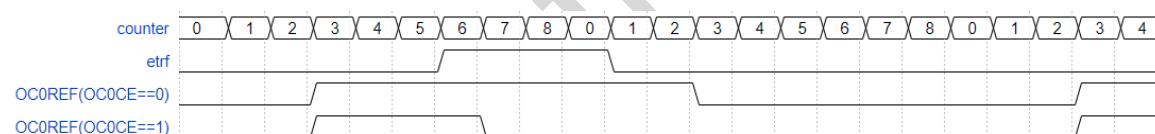


Figure 22-14 External brake signal trigger

## 22.7 Trigger input channels

Each GPTIMER's ETR has multiple sources, and one channel is selected to be input into the module through MUX. These signal sources are asynchronous with the GPTIMER, so synchronization processing is required inside the module. The synchronized ETR signal can select the effective level (or effective edge), configure the frequency division (1, 2, 4, 8) and filter processing according to the software configuration. The sampling frequency and window length of the filter can be configured by software (ETF).

## 22.8 Update event management

Update events have the following event sources:

1. Counter overflow events (overflow and underflow)
2. UG set

The control signals related to update event management are mainly URS and UDIS, and the specific controls are as follows:

- If UDIS=0, URS=0, underflow, overflow, and UG are set to initialize the counter and prescale counter (in center-aligned mode, the counter will not be cleared by overflow, nor will ARR be loaded by underflow). If the shadow register is enabled, the update event will update the written value to the shadow register (ARR depends on ARPE, CCRx depends on OCxPE), UIF will be set, and if the interrupt or DMA mask bit is enabled, an interrupt or DMA request will be generated.
- If UDIS=0, URS=1, underflow, overflow, and UG setting will initialize the counter and prescale counter (in center-aligned mode, the counter will not be cleared by overflow, nor will ARR be loaded by underflow). If the shadow register is enabled, the update event will update the written value to the shadow register (ARR depends on ARPE, CCRx depends on OCxPE). UIF will only be set in the case of overflow or underflow. If the interrupt or DMA mask bit is enabled, an interrupt or DMA request will be generated. This configuration can effectively avoid the situation where the capture interrupt and update interrupt are generated at the same time when UG is set to initialize the counter in input capture mode.
- If UDIS=1 (ignore URS), the underflow, overflow, and UG settings will initialize the counter and pre-scale counter (the counter will not be cleared by overflow in center-aligned mode, nor will it be loaded with ARR by underflow), but the shadow register will not be updated and UIF will not be set, so no corresponding interrupt or DMA request will be generated.

## 22.9 Quadrature encoder mode

This GPTIMER supports the quadrature encoding counting function. Quadrature signals can be input through channel 0 and channel 1 for counting and direction detection. There are three encoding modes: counting only on the edge of channel 0, counting only on the edge of channel 1, and counting on the edges of channel 1 and channel 2. Under this function, the two channel inputs can be configured with digital filtering functions, and the polarity configuration and frequency division configuration are invalid. Through the combination of the two channel signals, the counting enable and direction control signals can be generated to control the counter addition and subtraction (if CEN is enabled), so in this mode, the software configuration of the counting direction is invalid. See the table below for specific combinations.

Table 22-5 Encoder mode

编码模式	通道 0/1 电平	通道 0 边沿		通道 1 电平	
		上升沿	下降沿	上升沿	下降沿
编码模式 1 (在通道 1 边沿计数)	高电平	-	-	向上计数	向下计数
	低电平	-	-	向下计数	向上计数
编码模式 2 (在通道 0 边沿计数)	高电平	向下计数	向上计数	-	-
	低电平	向上计数	向下计数	-	-
编码模式 3 (在所有通道边沿计数)	高电平	向下计数	向上计数	向上计数	向下计数
	低电平	向上计数	向下计数	向下计数	向上计数

In encoding mode, the counter also counts between 0-ARR. When counting up to ARR, overflow occurs, and then it returns to 0 to count again. When counting down to 0, underflow occurs, and then it returns to ARR to count again. In addition, in this mode, input capture (channel 2 and channel 3), output comparison, frequency division, and trigger output functions are still applicable. The counting waveform of encoding mode 1 is as follows:

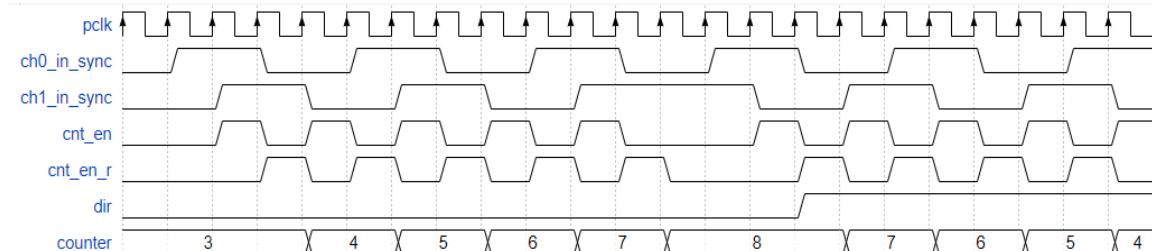


Figure 22-15 Counting waveform of encoder mode 1

## 22.10 Slave Mode Control

GPTIMER supports cascade operation as a slave of external or internal modules. The trigger input signal TRGI of slave mode has multiple sources, which are selected by TS[2:0]. The structure is shown in the figure above, where ITRx comes from the trigger output signal (TRGO) of other internal GPTIMERS. The specific mapping relationship is shown in the table below.

Table 22-6 GPTIMER internal trigger input mapping

从机 GPTIMER	ITR0	ITR1	ITR2
GPTIMER0	GPTIMER2	GPTIMER3	GPTIMER1
GPTIMER1	GPTIMER0	GPTIMER3	GPTIMER2
GPTIMER2	GPTIMER3	GPTIMER0	-
GPTIMER3	GPTIMER1	GPTIMER2	-

There are four main ways of slave mode control:

1. **Reset mode:** The rising edge of TRGI will initialize the counter and frequency divider counter, and can update the shadow register (when UDIS=0). The reset mode waveform in slave mode is as follows:

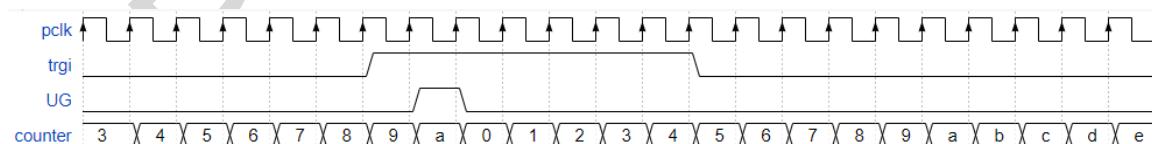


Figure 22-16 Reset mode waveform in slave mode

2. **Gated mode:** TRGI level can control the operation and stop of the counter. Under the default active level, the counter counts when the level is high, and stops counting (not reset) when the level is low. In this mode, CEN needs to be set by software. The waveform of the gated mode in slave mode is as follows:

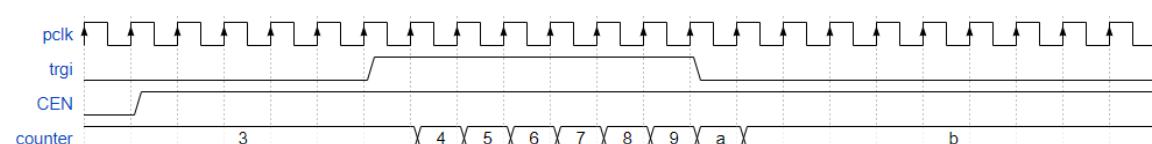


Figure 22-17 Gated mode waveform in slave mode

3. **Trigger mode:** The rising edge of TRGI can control the counter to start counting, but cannot control whether the counter stops. In this mode, CEN does not need to be set by software.

The waveform of the trigger mode in slave mode is as follows:

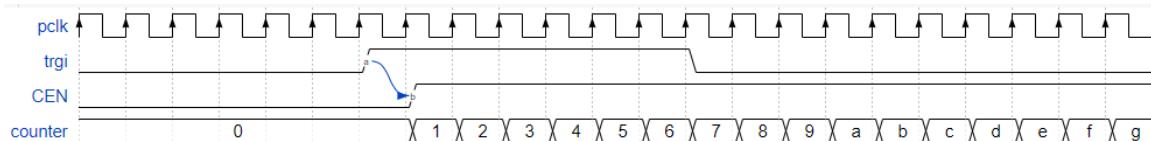


Figure 22-18 Trigger mode waveform in slave mode

4. **Clock mode** (i.e. external clock mode 1): The rising edge of TRGI is used as the count enable of the counter to control the count. At this time, the frequency division circuit is still valid.

In slave mode, the rising edge of TRGI will set the TIF flag bit, and if the corresponding interrupt or DMA mask bit is enabled, an interrupt or DMA request will be generated. However, the gated mode is somewhat special. In this mode, in addition to the rising edge, the falling edge can also set TIF.

In addition, when using GPTIMER in cascade, it is necessary to ensure that the master and slave clocks have the same frequency and phase, otherwise unknown errors will occur.

## 22.11 Master mode control

GPTIMER can also be used as a master mode to control other GPTIMERS or ADC and DAC by generating a trigger output signal (TRGO). The source of the TRGO signal can be configured by software as follows:

- MMS=3'b000: Reset mode, in which case the UG flag will be output as a TRGO signal to the external slave.
- MMS=3'b001: Enable mode, the count enable of the counter will be output as the TRGO signal to the external slave. If the current GPTIMER is also in the slave gating mode, the signal is the gating signal, otherwise CEN is directly output as the TRGO signal.
- MMS=3'b010: Update mode, in which case the update event is output as the TRGO signal.
- MMS=3'b011: Channel 0 compare pulse mode. If CC0IF is set, a pulse is output as the TRGO signal, regardless of whether CC0IF is already set.
- MMS=3'b100: Comparison mode 1, in this case OC0REF is output as the TRGO signal.
- MMS=3'b101: Comparison mode 2, in this case OC1REF is output as the TRGO signal.
- MMS=3'b110: Comparison mode 3, in this case OC2REF is output as the TRGO signal.
- MMS=3'b111: Comparison mode 4, in this case OC3REF is output as the TRGO signal.

**Note:** The signal OCxREF output by the last 4 modes is not the final channel output, but an internal signal.

When GPTIMER is configured as the master enable mode, there is a special application, which is to start the master and slave counters synchronously. However, because the master's CEN is output as TRGO to the slave and enables the slave counter, it takes two clock delays (assuming that the master and slave clocks are of the same frequency and phase), so when using this function, the host's CEN signal will be delayed by 2 clock cycles using two-level registers to ensure synchronization. This function can be enabled by software configuration (MSM).

## 22.12 Output control

GPTIMER0 and GPTIMER1 have 4 output channels, GPTIMER2 and GPTIMER3 have 2 output channels, and have corresponding output enable signals. Channel output is only valid when CCxE is set. At this time, the output polarity can be controlled by CCxP. The output polarity refers to whether the output valid level is high or low. The output enable signal is high-effective, that is, it is effective when CCxE is set. At the same time, it is necessary to ensure that the channel is correctly configured as the output mode, and the output mode is configured by CCxS.

## 22.13 Channels remapping

Channel remapping is to map the input signal of GPTIMER channel or external trigger channel ETR from other external or internal signals. GPTIMER0's ETR channel, channel 0 and channel 3 support remapping, GPTIMER1's channel 2 supports remapping, GPTIMER2's ETR channel, channel 0 and channel 1 support remapping, GPTIMER3's ETR channel and channel 0 support remapping.

## 22.14 Debug mode control

GPTIMER can be configured by software to stop counting in debug mode. If this function is enabled, GPTIMER will be stopped (the counter will not be initialized) when entering the system debug mode.

## 22.15 DMA

GPTIMER has a total of 6 DMA request sources, named update event (UIF), 4 channel events (capture event, compare Match) (CCxIF) and trigger event (TIF), whether to enable the corresponding DMA request can be configured by independent mask bits. For channel event DMA, the DMA request source of the channel can be configured by software (CCDS bit). If CCDS=0, each channel DMA requests come from events of each channel, such as capture and comparison match events; if CCDS=1, then DMA requests of each channel All come from update events, and channel events will be blocked.

Each DMA request is set only when there is no corresponding acknowledge signal, DMA enable is turned on and a DMA event occurs. When the DMA request is set, the acknowledge signal can clear the DMA request, otherwise the DMA request will remain set.

In addition to conventional DMA operations, GPTIMER also supports burst functions, that is, a DMA request can continuously read and write multiple internal registers. The DBL bit can select the burst length, up to 18, and the DBA can select the starting address of the burst. The address of the DMAR register can be used as the target address or source address of the DMA (DMA does not need to be set to increment each time). When a DMA request is set, GPTIMER calculates the actual address of each read and write operation based on the values of DBL and DBA. The actual address calculation method is: CR1 + (DBA + index) × 4, where the value of index is 0 to DBL.

**Note:** There is a reserved register address in the middle of the register group, which will also be included in the burst operation of DMA. When actually using it, please pay attention to the configured length. For example, if the starting address is the ARR register (0x2C), and the DBL is configured as 5'b00010 (3 bursts), the three registers actually operated by DMA are 0x2C, 0x30, and 0x34, of which 0x30 is a reserved register, so it cannot be written and the read value is always 0.

## 22.16 Interrupts

GPTIMER has 6 interrupt sources, named update event (UIF), 4-channel event (capture event, compare match) (CCxIF) and trigger event (TIF). Each interrupt can be enabled or not by an independent interrupt mask bit. The interrupt flag bit and the corresponding mask bit are in an AND relationship, and the interrupts are in an OR relationship. The interrupt signals of GPTIMER are as follows:

**Table 22-7 GPTIMER interrupts**

Interrupt	Description
触发事件中断	触发源产生事件时的中断
通道 3 事件中断	通道 3 产生捕获或比较事件时的中断
通道 2 事件中断	通道 2 产生捕获或比较事件时的中断
通道 1 事件中断	通道 1 产生捕获或比较事件时的中断
通道 0 事件中断	通道 0 产生捕获或比较事件时的中断
更新事件中断	产生更新事件时的中断

The above interrupts are enabled by configuring the TIE, CC3IE, CC2IE, CC1IE, CC0IE, and UIE bits of the DIER register respectively.

## 22.17 GPTIMER registers

GPTIMER0 Base Address: 0x4000A000

GPTIMER1 Base Address: 0x4001A000

GPTIMER2 Base Address: 0x4000B000

GPTIMER3 Base Address: 0x4001B000

**Table 22-8 GPTIMER Registers Summary**

Register	Offset	Description
GPTIM_CR1	0x00	控制寄存器 1
GPTIM_CR2	0x04	控制寄存器 2
GPTIM_SMCR	0x08	从模式控制寄存器
GPTIM_DIER	0x0C	DMA/中断使能寄存器
GPTIM_SR	0x10	状态寄存器
GPTIM_EGR	0x14	事件寄存器
GPTIM_CCMR1	0x18	捕获比较模式寄存器 1
GPTIM_CCMR2	0x1C	捕获比较模式寄存器 2
GPTIM_CCER	0x20	捕获比较使能寄存器
GPTIM_CNT	0x24	计数寄存器



ASR

22. GPTIMER

ASR6601 Reference Manual



GPTIM_PSC	0x28	计数器分频值寄存器
GPTIM_ARR	0x2C	计数器重装载值寄存器
GPTIM_CCR0	0x34	通道 0 捕获比较寄存器
GPTIM_CCR1	0x38	通道 1 捕获比较寄存器
GPTIM_CCR2	0x3C	通道 2 捕获比较寄存器
GPTIM_CCR3	0x40	通道 3 捕获比较寄存器
GPTIM_DCR	0x48	DMA 控制寄存器
GPTIM_DMAR	0x4C	DMA 地址寄存器
GPTIM_OR	0x50	通道重映射寄存器

## 22.17.1 GPTIM\_CR1

Offset: 0x000

Reset Value: 0x0000

<b>15-10</b>	<b>9-8</b>	<b>7</b>	<b>6-5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
RESERVED	CKD	ARPE	CMS	DIR	OPM	URS	UDIS	CEN
r-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 15-10 RESERVED:** Must be kept, and cannot be modified.

**Bits 9-8 CKD:** Sample clock divider.

- 00: fDTS = fpclk
- 01: fDTS = fpclk
- 10: fDTS = fpclk
- 11: fDTS = reserved

**Bit 7 ARPE:** Reload shadow register enable.

- 0: ARR shadow register disabled
- 1: ARR shadow register enabled

**Bits 6-5 CMS:** Intermediate counting mode selection.

- 00: edge-aligned counting mode, DIR controls up or down counting
- 01: center alignment mode 1. The output compare interrupt flag is only set during down counting
- 10: center alignment mode 2. The output compare interrupt flag is only set during upward counting
- 11: center alignment mode 3. The output compare interrupt flag is set during both up and down counting processes

**Bit 4 DIR:** Counting direction selection. Center alignment mode and encoding mode, this bit is controlled by hardware.

- 0: count up
- 1: count down

**Bit 3 OPM:** Single Pulse Mode Enable.

- 0: single pulse mode disabled
- 1: single pulse mode enabled, the counter stops counting at the next update event

**Bit 2 URS:** Update event source selection. This bit only affects the interrupt and DMA flag bit (UIF) and does not affect the internal logic.

- 0: counter overflow, UG bit, and trigger in slave reset mode can all set UIF
- 1: only counter overflow events can set UIF

**Bit 1 UDIS:** Update events disable.

- 0: Update event enabled, middle alignment mode 1. The output compare interrupt flag can generate updates only if it is set during the downward counting process event
- 1: The update event is disabled, the shadow register and UIF will not be updated, but at this time the counter and divider counter can still be set by UG Bit event initialization

**Bit 0 CEN:** Counter enable, CEN is set by hardware in trigger mode, and CEN is cleared by hardware in single pulse mode.

- 0: counter disabled
- 1: counter enabled

## 22.17.2 GPTIM\_CR2

Offset: 0x004

Reset Value: 0x0000

15-8	7	6-4	3	2-0
RESERVED	TIOS	MMS	CCDS	RESERVED
r-0h	rw-0h	rw-0h	rw-0h	r-0h

**Bits 15-8 RESERVED:** Must be kept, and cannot be modified.

**Bit 7 TIOS:** Channel 1 source XOR selection (this feature is only supported by timer0 and timer1).

- 0: channel 0 is mapped to channel 0 input
- 1: channel 0 is the XOR output of channels 0, 1, and 2

**Bits 6-4 MMS:** Master mode selection, configurable TRGO output.

- 000: reset mode, UG will be output as TRGO signal
- 001: enable mode, CNT\_EN (not CEN) will be output as TRGO signal
- 010: update mode, update events (internal signals) will be output as TRGO signals
- 011: compare pulse mode, TRGO will output a pulse every time CC0IF is about to be set, even if CC0IF has been set
- 100: compare mode, OC0REF (internal signal) is output as TRGO signal
- 101: compare mode, OC1REF (internal signal) is output as TRGO signal
- 110: compare mode, OC2REF (internal signal) is output as TRGO signal
- 111: compare mode, OC3REF (internal signal) is output as TRGO signal

**Bit 3 CCDS:** Channel DMA request source selection (this feature is only supported by gtimer0 and gtimer1).

- 0: DMA requests of each channel (excluding update event requests and trigger event requests) are generated by channel events (capture, comparison)
- 1: DMA requests of each channel (excluding update event requests and trigger event requests) are generated by update events

**Bits 2-0 RESERVED:** Must be kept, and cannot be modified.

### 22.17.3 GPTIM\_SMCR

Offset: 0x008

Reset Value: 0x0000

15	14	13-12	11-8	7	6-4	3	2-0
ETP	ECE	ETPS	ETF	MSM	TS	RESERVED	SMS
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	r-0h	rw-0h

**Bit 15 ETP:** External trigger polarity selection (it is best not to select the mode (SMS) when configuring the polarity, in order to prevent the internal signal from flipping and triggering unknown errors).

- 0: external trigger input not inverted
- 1: external trigger input inverted

**Bit 14 ECE:** External clock mode 2 enable.

- 0: external clock mode 2 disabled
- 1: external clock mode 2 enabled

**Bit 13-12 ETPS:** External trigger input frequency division (this frequency division is mainly used for 50% duty cycle frequency reduction, such as 24M signal divided by 2 to 12M, the level is doubled).

- 00: no frequency division
- 01: frequency divided by 2
- 10: frequency divided by 4
- 11: frequency divided by 8

**Bits 11-8 ETF:** External trigger input filter configuration.

- 0000: filter disabled
- 0001: filter sampling frequency fsampling=fclk, filter length N=2
- 0010: filter sampling frequency fsampling=fclk, filter length N=4
- 0011: filter sampling frequency fsampling=fclk, filter length N=8
- 0100: filter sampling frequency fsampling=fDTS/2, filter length N=6
- 0101: filter sampling frequency fsampling=fDTS/2, filter length N=8
- 0110: filter sampling frequency fsampling=fDTS/4, filter length N=6
- 0111: Filter sampling frequency fsampling=fDTS/4, filter length N=8
- 1000: filter sampling frequency fsampling=fDTS/8, filter length N=6
- 1001: filter sampling frequency fsampling=fDTS/8, filter length N=8
- 1010: filter sampling frequency fsampling=fDTS/16, filter length N=5
- 1011: Filter sampling frequency fsampling=fDTS/16, filter length N=6
- 1100: filter sampling frequency fsampling=fDTS/16, filter length N=8
- 1101: filter sampling frequency fsampling=fDTS/32, filter length N=5
- 1110: filter sampling frequency fsampling=fDTS/32, filter length N=6
- 1111: filter sampling frequency fsampling=fDTS/32, filter length N=8

**Bit 7 MSM:** Master-slave mode synchronization (when using this function, it is necessary to ensure that the clocks of the two timers are in the same frequency and phase).

- 0: no action
- 1: the TRGI trigger input will be delayed to start counting at the same time as the slave counter.

**Bit 6-4 TS:** Trigger source selection, select the source of TRGI (SMS must be cleared when configuring this bit).

- 000: ITR0
- 001: ITR1
- 010: ITR2 (timer2 and timer3 do not have this channel)
- 011: reserved
- 100: channel 0 edge detection output
- 101: channel 0 filter output
- 110: channel 1 filter output
- 111: channel trigger input

**Bit 3 RESERVED:** Must be kept, and cannot be modified.

**Bit 2-0 SMS:** Slave mode selection (it is best to configure the channel parameters before selecting the mode to prevent internal signal flipping from triggering unknown errors).

- 000: slave mode disabled
- 001: coding mode 1, the counter only counts on the edge of channel 1
- 010: coding mode 2, the counter only counts on the edge of channel 0
- 011: coding mode 3, the counter counts on the edges of channels 0 and 1
- 100: reset mode, the rising edge of TRGI will reset the counter
- 101: gating mode, the counter only counts during TRGI high level
- 110: trigger mode, the counter will start counting on the rising edge of TRGI. This mode only controls the start of counting
- 111: external clock mode 1, the rising edge of TRGI is used as the counter counting clock

#### 22.17.4 GPTIM\_DIER

Offset: 0x00C

Reset Value: 0x0000

15	14	13	12	11	10	9	8
RESERVED	TDE	RESERVED	CC3DE	CC2DE	CC1DE	CC0DE	UDE
r-0h	rw-0h	r-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h
7	6	5	4	3	2	1	0
RESERVED	TIE	RESERVED	CC3IE	CC2IE	CC1IE	CC0IE	UIE
r-0h	rw-0h	r-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bit 15 RESERVED:** Must be kept, and cannot be modified.

**Bit 14 TDE:** Trigger event DMA request enable.

- 0: trigger event DMA request disabled
- 1: trigger event DMA request enabled

**Bit 13 RESERVED:** Must be kept, and cannot be modified.

**Bit 12 CC3DE:** Channel 3 event DMA request enable.

- 0: channel 3 event DMA request disabled
- 1: channel 3 event DMA request enabled

**Bit 11 CC2DE:** Channel 2 event DMA request enable.

- 0: channel 2 event DMA request disabled
- 1: channel 2 event DMA request enabled

**Bit 10 CC1DE:** Channel 1 event DMA request enable.

- 0: channel 1 event DMA request disabled
- 1: channel 1 event DMA request enabled

**Bit 9 CC0DE:** Channel 0 event DMA request enable.

- 0: channel 0 event DMA request disabled
- 1: channel 0 event DMA request enabled

**Bit 8 UDE:** Update event DMA request enable.

- 0: update event DMA request disabled
- 1: update event DMA request enabled

**Bit 7 RESERVED:** Must be kept, and cannot be modified.

**Bit 6 TIE:** Trigger event interrupt enable.

- 0: trigger event interrupt disabled
- 1: trigger event interrupt enabled

**Bit 5 RESERVED:** Must be kept, and cannot be modified.

**Bit 4 CC3IE:** Channel 3 event interrupt enable.

- 0: channel 3 event interrupt disabled
- 1: channel 3 event interrupt enabled

**Bit 3 CC2IE:** Channel 2 event interrupt enable.

- 0: channel 2 event interrupt disabled
- 1: channel 2 event interrupt enabled

**Bit 2 CC1IE:** Channel 1 event interrupt request enable.

- 0: channel 1 event interrupt disabled
- 1: channel 1 event interrupt enabled

**Bit 1 CC0IE:** Channel 0 event interrupt enable.

- 0: channel 0 event interrupt disabled
- 1: channel 0 event interrupt enabled

**Bit 0 UIE:** Update event interrupt enable.

- 0: update event interrupt disabled
- 1: update event interrupt enabled

## 22.17.5 GPTIM\_SR

Offset: 0x010

Reset Value: 0x0000

15-13		12	11	10	9	8-7
RESERVED		CC3OF	CC2OF	CC1OF	CC0OF	RESERVED
r-0h		rw-0h	rw-0h	rw-0h	rw-0h	r-0h
6	5	4	3	2	1	0
TIF	RESERVED	CC3IF	CC2IF	CC1IF	CC0IF	UIF
rw-0h	r-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bits 15-13 RESERVED:** Must be kept, and cannot be modified.

**Bit 12 CC3OF:** Channel 3 overcapture flag (cleared by writing 0).

- 0: no overcapture
- 1: at least 1 overcapture occurred

**Bit 11 CC2OF:** Channel 2 overcapture flag (cleared by writing 0).

- 0: no overcapture
- 1: at least 1 overcapture occurred

**Bit 10 CC1OF:** Channel 1 overcapture flag (cleared by writing 0).

- 0: no overcapture
- 1: at least 1 overcapture occurred

**Bit 9 CC0OF:** Channel 0 overcapture flag (cleared by writing 0).

- 0: no overcapture
- 1: at least 1 overcapture occurred

**Bits 8-7 RESERVED:** Must be kept, and cannot be modified.

**Bit 6 TIF:** Trigger event interrupt flag (cleared by writing 0).

- 0: no event
- 1: trigger event occurs

**Bit 5 RESERVED:** Must be kept, and cannot be modified.

**Bit 4 CC3IF:** Channel 3 Capture/Compare Event Flag (Compare mode: write 0 to clear; Capture mode: read the ccrx register or write 0 both can be cleared).

- 0: no event
- 1: capture or compare event occurs

**Bit 3 CC2IF:** Channel 2 Capture/Compare Event Flag (Compare mode: write 0 to clear; Capture mode: read the ccrx register or write 0 both can be cleared).

- 0: no event
- 1: capture or compare event occurs

**Bit 2 CC1IF:** Channel 1 Capture/Compare Event Flag (Compare mode: write 0 to clear; Capture mode: read the ccrx register or write 0 both can be cleared).

- 0: no event

- 1: capture or compare event occurs

**Bit 1 CC0IF:** Channel 0 Capture/Compare Event Flag (Compare mode: write 0 to clear; Capture mode: read the ccrx register or write 0 both can be cleared).

- 0: no event
- 1: capture or compare event occurs

**Bit 0 UIF:** Update event flag (read SR or write 0 to clear this bit).

- 0: no event
- 1: update event occurs

## 22.17.6 GPTIM\_EGR

Offset: 0x014

Reset Value: 0x0000

15-7	6	5	4	3	2	1	0
RESERVED	TG	RESERVED	CC3G	CC2G	CC1G	CC0G	UG
r-0h	w-0h	r-0h	w-0h	w-0h	w-0h	w-0h	w-0h

**Bits 15-7 RESERVED:** Must be kept, and cannot be modified.

**Bit 6 TG:** generate trigger.

- 0: no action
- 1: generate a trigger event, TIF is set

**Bit 5 RESERVED:** Must be kept, and cannot be modified.

**Bit 4 CC3G:** generate Channel 3 event

- 0: no action
- 1: generate capture action in input mode and comparison action in output mode. CC3IF is set in both modes.

**Bit 3 CC2G:** generate Channel 2 event

- 0: no action
- 1: generate capture action in input mode and comparison action in output mode. CC2IF is set in both modes.

**Bit 2 CC1G:** generate Channel 1 event

- 0: no action
- 1: generate capture action in input mode and comparison action in output mode. CC1IF is set in both modes.

**Bit 1 CC0G:** generate Channel 0 event

- 0: no action
- 1: generate capture action in input mode and comparison action in output mode. CC0IF is set in both modes.

**Bit 0 UG:** generate update event

- 0: no action
- 1: generate update event

## 22.17.7 GPTIM\_CCMR1

Offset: 0x018

Reset Value: 0x0000

### Output mode structure:

15	14-12	11	10	9-8	7	6-4	3	2	1-0
OC1CE	OC1M	OC1PE	OC1FE	CC1S	OC0CE	OC0M	OC0PE	OC0FE	CC0S
rw-0h									

#### Bit 15 OC1CE: Channel 1 Output Compare Clear Enable.

- 0: clear function disabled
- 1: clear function enabled, ETRF high level can clear the channel output

#### Bits 12-14 OC1M: Channel 1 output compare mode selection.

- 000: freeze mode, channel output does not change with the comparison result
- 001: valid mode, channel output valid level after matching
- 010: failure mode, channel output failure level after matching
- 011: flip mode, the channel output will be flipped after matching
- 100: forced valid mode. After selecting this mode, the valid level will be output directly
- 101: forced failure mode. After selecting this mode, the failure level will be output directly
- 110: PWM1 mode. In this mode, when counting upward, the channel outputs a valid level when CNT<CCR, otherwise the output level is invalid; When counting down, the channel outputs an invalid level when CNT>CCR, otherwise it outputs a valid level (when counting up, if CCRx>ARR, OCxREF always outputs high level. If CCRx==0, OCxREF always outputs low level; when counting down, if CCRx>ARR, OCxREF always outputs high level, and 0% PWM is not supported at this time)
- 111: PWM2 mode. In this mode, when counting upward, when CNT<CCR, the channel outputs an invalid level, otherwise it outputs a valid level; when counting down, when CNT>CCR, the channel outputs a valid level, otherwise it outputs an invalid level (0% and 100% waveforms are the same as PWM1)

#### Bit 11 OC1PE: Channel 1 output compare shadow register enable.

- 0: shadow register disabled
- 1: shadow register enabled

#### Bit 10 OC1FE: Channel 1 fast output enable.

- 0: fast mode disabled, the output only changes when matching
- 1: fast mode enabled. The trigger input is equivalent to a matching event, which directly affects the channel output and is not affected by the comparison between the counter and CCR

#### Bits 9-8 CC1S: Capture/Compare Select.

- 00: channel is configured in output mode
- 01: channel is configured in input mode, and the capture channel input is mapped to channel 1
- 10: channel is configured in input mode, and the capture channel input is mapped to channel 0
- 11: channel is configured in input mode, and the capture channel input is mapped to the trigger input TRC

#### Bit 7 OC0CE: Channel 1 output compare clear enable.

- 0: clear function disabled
- 1: clear function enabled, ETRF high level can clear the channel output

**Bits 6-4 OC0M:** Channel 0 output compare mode selection.

- 000: freeze mode, channel output does not change with the comparison result
- 001: valid mode, channel output valid level after matching
- 010: failure mode, channel output failure level after matching
- 011: flip mode, the channel output will be flipped after matching
- 100: forced valid mode. After selecting this mode, the valid level will be output directly
- 101: forced failure mode. After selecting this mode, the failure level will be output directly
- 110: PWM1 mode. In this mode, when counting upward, the channel outputs a valid level when CNT<CCR, otherwise the output level is invalid; When counting down, the channel outputs an invalid level when CNT>CCR, otherwise it outputs a valid level (when counting up, if CCRx>ARR, OCxREF always outputs high level. If CCRx==0, OCxREF always outputs low level; when counting down, if CCRx>ARR, OCxREF always outputs high level, and 0% PWM is not supported at this time)
- 111: PWM2 mode. In this mode, when counting upward, when CNT<CCR, the channel outputs an invalid level, otherwise it outputs a valid level; when counting down, when CNT>CCR, the channel outputs a valid level, otherwise it outputs an invalid level (0% and 100% waveforms are the same as PWM1)

**Bit 3 OC0PE:** Channel 0 output compare shadow register enable.

- 0: shadow register disabled
- 1: shadow register enabled

**Bit 2 OC0FE:** Channel 0 fast output enable.

- 0: fast mode disabled, the output only changes when matching
- 1: fast mode enabled. The trigger input is equivalent to a matching event, which directly affects the channel output and is not affected by the comparison between the counter and CCR

**Bits 1-0 CC0S:** Capture/Compare Select.

- 00: channel is configured in output mode
- 01: channel is configured in input mode, and the capture channel input is mapped to channel 1
- 10: channel is configured in input mode, and the capture channel input is mapped to channel 0
- 11: channel is configured in input mode, and the capture channel input is mapped to the trigger input TRC

**Input mode structure:**

15-12	11-10	9-8	7-4	3-2	1-0
IC1F	IC1PSC	CC1S	IC0F	IC0PSC	CC0S
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bit 15-12 IC1F:** Channel 1 input filter configuration (CCxS ! = 0x0 needs to be configured for this function to take effect).

- 0000: filter disabled
- 0001: filter sampling frequency fsampling=fclk, filter length N=2
- 0010: filter sampling frequency fsampling=fclk, filter length N=4
- 0011: filter sampling frequency fsampling=fclk, filter length N=8
- 0100: filter sampling frequency fsampling=fDTS/2, filter length N=6
- 0101: filter sampling frequency fsampling=fDTS/2, filter length N=8
- 0110: filter sampling frequency fsampling=fDTS/4, filter length N=6
- 0111: filter sampling frequency fsampling=fDTS/4, filter length N=8
- 1000: filter sampling frequency fsampling=fDTS/8, filter length N=6
- 1001: filter sampling frequency fsampling=fDTS/8, filter length N=8
- 1010: filter sampling frequency fsampling=fDTS/16, filter length N=5

- 1011: filter sampling frequency fsampling=fDTS/16, filter length N=6
- 1100: filter sampling frequency fsampling=fDTS/16, filter length N=8
- 1101: filter sampling frequency fsampling=fDTS/32, filter length N=5
- 1110: filter sampling frequency fsampling=fDTS/32, filter length N=6
- 1111: filter sampling frequency fsampling=fDTS/32, filter length N=8

**Bits 11-10 IC1PSC:** Channel 1 frequency division (CCxS! =0x0 needs to be configured for this function to take effect).

- 00: no frequency division
- 01: frequency divided by 2
- 10: frequency divided by 4
- 11: frequency divided by 8

**Bits 9-8 CC1S:** Capture/Compare Select.

- 00: channel is configured in output mode
- 01: channel is configured in input mode, and the capture channel input is mapped to channel 1
- 10: channel is configured in input mode, and the capture channel input is mapped to channel 0
- 11: channel is configured in input mode, and the capture channel input is mapped to the trigger input TRC

**Bit 7-4 IC1F:** Channel 0 input filter configuration (CCxS ! = 0x0 needs to be configured for this function to take effect).

- 0000: filter disabled
- 0001: filter sampling frequency fsampling=fpclk, filter length N=2
- 0010: filter sampling frequency fsampling=fpclk, filter length N=4
- 0011: filter sampling frequency fsampling=fpclk, filter length N=8
- 0100: filter sampling frequency fsampling=fDTS/2, filter length N=6
- 0101: filter sampling frequency fsampling=fDTS/2, filter length N=8
- 0110: filter sampling frequency fsampling=fDTS/4, filter length N=6
- 0111: filter sampling frequency fsampling=fDTS/4, filter length N=8
- 1000: filter sampling frequency fsampling=fDTS/8, filter length N=6
- 1001: filter sampling frequency fsampling=fDTS/8, filter length N=8
- 1010: filter sampling frequency fsampling=fDTS/16, filter length N=5
- 1011: filter sampling frequency fsampling=fDTS/16, filter length N=6
- 1100: filter sampling frequency fsampling=fDTS/16, filter length N=8
- 1101: filter sampling frequency fsampling=fDTS/32, filter length N=5
- 1110: filter sampling frequency fsampling=fDTS/32, filter length N=6
- 1111: filter sampling frequency fsampling=fDTS/32, filter length N=8

**Bits 3-2 IC0PSC:** Channel 0 frequency division (CCxS! =0x0 needs to be configured for this function to take effect).

- 00: no frequency division
- 01: frequency divided by 2
- 10: frequency divided by 4
- 11: frequency divided by 8

**Bits 1-0 CC0S:** Capture/Compare Select.

- 00: channel is configured in output mode
- 01: channel is configured in input mode, and the capture channel input is mapped to channel 1
- 10: channel is configured in input mode, and the capture channel input is mapped to channel 0
- 11: channel is configured in input mode, and the capture channel input is mapped to the trigger input TRC

## 22.17.8 GPTIM\_CCMR2

Offset: 0x01C

Reset Value: 0x0000

### Output mode structure:

15	14-12	11	10	9-8	7	6-4	3	2	1-0
OC3CE	OC3M	OC3PE	OC3FE	CC3S	OC2CE	OC2M	OC2PE	OC2FE	CC2S
rw-0h									

**Bit 15 OC3CE:** Channel 3 Output Compare Clear Enable.

- 0: clear function disabled
- 1: clear function enabled, ETRF high level can clear the channel output

**Bits 12-14 OC3M:** Channel 3 output compare mode selection.

- 000: freeze mode, channel output does not change with the comparison result
- 001: valid mode, channel output valid level after matching
- 010: failure mode, channel output failure level after matching
- 011: flip mode, the channel output will be flipped after matching
- 100: forced valid mode. After selecting this mode, the valid level will be output directly
- 101: forced failure mode. After selecting this mode, the failure level will be output directly
- 110: PWM1 mode. In this mode, when counting upward, the channel outputs a valid level when CNT<CCR, otherwise the output level is invalid; When counting down, the channel outputs an invalid level when CNT>CCR, otherwise it outputs a valid level (when counting up, if CCRx>ARR, OCxREF always outputs high level. If CCRx==0, OCxREF always outputs low level; when counting down, if CCRx>ARR, OCxREF always outputs high level, and 0% PWM is not supported at this time)
- 111: PWM2 mode. In this mode, when counting upward, when CNT<CCR, the channel outputs an invalid level, otherwise it outputs a valid level; when counting down, when CNT>CCR, the channel outputs a valid level, otherwise it outputs an invalid level (0% and 100% waveforms are the same as PWM1)

**Bit 11 OC3PE:** Channel 3 output compare shadow register enable.

- 0: shadow register disabled
- 1: shadow register enabled

**Bit 10 OC3FE:** Channel 3 fast output enable.

- 0: fast mode disabled, the output only changes when matching
- 1: fast mode enabled. The trigger input is equivalent to a matching event, which directly affects the channel output and is not affected by the comparison between the counter and CCR

**Bits 9-8 CC3S:** Capture/Compare Select.

- 00: channel is configured in output mode
- 01: channel is configured in input mode, and the capture channel input is mapped to channel 1
- 10: channel is configured in input mode, and the capture channel input is mapped to channel 0
- 11: channel is configured in input mode, and the capture channel input is mapped to the trigger input TRC

**Bit 7 OC2CE:** Channel 2 output compare clear enable.

- 0: clear function disabled
- 1: clear function enabled, ETRF high level can clear the channel output

**Bits 6-4 OC2M:** Channel 2 output compare mode selection.

- 000: freeze mode, channel output does not change with the comparison result
- 001: valid mode, channel output valid level after matching
- 010: failure mode, channel output failure level after matching
- 011: flip mode, the channel output will be flipped after matching
- 100: forced valid mode. After selecting this mode, the valid level will be output directly
- 101: forced failure mode. After selecting this mode, the failure level will be output directly
- 110: PWM1 mode. In this mode, when counting upward, the channel outputs a valid level when CNT<CCR, otherwise the output level is invalid; When counting down, the channel outputs an invalid level when CNT>CCR, otherwise it outputs a valid level (when counting up, if CCRx>ARR, OCxREF always outputs high level. If CCRx==0, OCxREF always outputs low level; when counting down, if CCRx>ARR, OCxREF always outputs high level, and 0% PWM is not supported at this time)
- 111: PWM2 mode. In this mode, when counting upward, when CNT<CCR, the channel outputs an invalid level, otherwise it outputs a valid level; when counting down, when CNT>CCR, the channel outputs a valid level, otherwise it outputs an invalid level (0% and 100% waveforms are the same as PWM1)

**Bit 3 OC2PE:** Channel 2 output compare shadow register enable.

- 0: shadow register disabled
- 1: shadow register enabled

**Bit 2 OC2FE:** Channel 2 fast output enable.

- 0: fast mode disabled, the output only changes when matching
- 1: fast mode enabled. The trigger input is equivalent to a matching event, which directly affects the channel output and is not affected by the comparison between the counter and CCR

**Bits 1-0 CC2S:** Capture/Compare Select.

- 00: channel is configured in output mode
- 01: channel is configured in input mode, and the capture channel input is mapped to channel 1
- 10: channel is configured in input mode, and the capture channel input is mapped to channel 0
- 11: channel is configured in input mode, and the capture channel input is mapped to the trigger input TRC

**Input mode structure:**

15-12	11-10	9-8	7-4	3-2	1-0
IC3F	IC3PSC	CC3S	IC2F	IC2PSC	CC2S
rw-0h	rw-0h	rw-0h	rw-0h	rw-0h	rw-0h

**Bit 15-12 IC3F:** Channel 3 input filter configuration (CCxS != 0x0 needs to be configured for this function to take effect).

- 0000: filter disabled
- 0001: filter sampling frequency fsampling=fclk, filter length N=2
- 0010: filter sampling frequency fsampling=fclk, filter length N=4
- 0011: filter sampling frequency fsampling=fclk, filter length N=8
- 0100: filter sampling frequency fsampling=fDTS/2, filter length N=6
- 0101: filter sampling frequency fsampling=fDTS/2, filter length N=8
- 0110: filter sampling frequency fsampling=fDTS/4, filter length N=6
- 0111: filter sampling frequency fsampling=fDTS/4, filter length N=8
- 1000: filter sampling frequency fsampling=fDTS/8, filter length N=6
- 1001: filter sampling frequency fsampling=fDTS/8, filter length N=8
- 1010: filter sampling frequency fsampling=fDTS/16, filter length N=5

- 1011: filter sampling frequency fsampling=fDTS/16, filter length N=6
- 1100: filter sampling frequency fsampling=fDTS/16, filter length N=8
- 1101: filter sampling frequency fsampling=fDTS/32, filter length N=5
- 1110: filter sampling frequency fsampling=fDTS/32, filter length N=6
- 1111: filter sampling frequency fsampling=fDTS/32, filter length N=8

**Bits 11-10 IC3PSC:** Channel 3 frequency division (CCxS! =0x0 needs to be configured for this function to take effect).

- 00: no frequency division
- 01: frequency divided by 2
- 10: frequency divided by 4
- 11: frequency divided by 8

**Bits 9-8 CC3S:** Capture/Compare Select.

- 00: channel is configured in output mode
- 01: channel is configured in input mode, and the capture channel input is mapped to channel 1
- 10: channel is configured in input mode, and the capture channel input is mapped to channel 0
- 11: channel is configured in input mode, and the capture channel input is mapped to the trigger input TRC

**Bit 7-4 IC2F:** Channel 0 input filter configuration (CCxS ! = 0x0 needs to be configured for this function to take effect).

- 0000: filter disabled
- 0001: filter sampling frequency fsampling=fpclk, filter length N=2
- 0010: filter sampling frequency fsampling=fpclk, filter length N=4
- 0011: filter sampling frequency fsampling=fpclk, filter length N=8
- 0100: filter sampling frequency fsampling=fDTS/2, filter length N=6
- 0101: filter sampling frequency fsampling=fDTS/2, filter length N=8
- 0110: filter sampling frequency fsampling=fDTS/4, filter length N=6
- 0111: filter sampling frequency fsampling=fDTS/4, filter length N=8
- 1000: filter sampling frequency fsampling=fDTS/8, filter length N=6
- 1001: filter sampling frequency fsampling=fDTS/8, filter length N=8
- 1010: filter sampling frequency fsampling=fDTS/16, filter length N=5
- 1011: filter sampling frequency fsampling=fDTS/16, filter length N=6
- 1100: filter sampling frequency fsampling=fDTS/16, filter length N=8
- 1101: filter sampling frequency fsampling=fDTS/32, filter length N=5
- 1110: filter sampling frequency fsampling=fDTS/32, filter length N=6
- 1111: filter sampling frequency fsampling=fDTS/32, filter length N=8

**Bits 3-2 IC2PSC:** Channel 2 frequency division (CCxS! =0x0 needs to be configured for this function to take effect).

- 00: no frequency division
- 01: frequency divided by 2
- 10: frequency divided by 4
- 11: frequency divided by 8

**Bits 1-0 CC2S:** Capture/Compare Select.

- 00: channel is configured in output mode
- 01: channel is configured in input mode, and the capture channel input is mapped to channel 1
- 10: channel is configured in input mode, and the capture channel input is mapped to channel 0
- 11: channel is configured in input mode, and the capture channel input is mapped to the trigger input TRC

## 22.17.9 GPTIM\_CCER

Offset: 0x020

Reset Value: 0x0000

15	14	13	12	11	10	9	8
CC3NP	RESERVED	CC3P	CC3E	CC2NP	RESERVED	CC2P	CC2E
rw-0h	r-0h	rw-0h	rw-0h	rw-0h	r-0h	rw-0h	rw-0h
7	6	5	4	3	2	1	0
CC1NP	RESERVED	CC1P	CC1E	CC0NP	RESERVED	CC0P	CC0E
rw-0h	r-0h	rw-0h	rw-0h	rw-0h	r-0h	rw-0h	rw-0h

**Bit 15 CC3NP:** Output reverse polarity, this bit must be set to 0 in output mode, see CC3P in input mode.

**Bit 14 RESERVED:** Must be kept, and cannot be modified.

**Bit 13 CC3P:** Output polarity, must work together with CC3NP (it is best to configure the polarity before mode selection to prevent internal signal inversion. Redirect triggers an unknown error).

**Output mode:**

- 0: high level output polarity
- 1: low level output polarity

**Input mode, {CC3NP, CC3P}:**

- 00: channel input active rising edge is active (capture, trigger, reset, clock mode), high level is active (gating, encoding mode)
- 01: channel input falling edge is active (capture, trigger, reset, clock mode), low level is active (gating, encoding mode)
- 10: reserved
- 11: both rising and falling edges of channel input are active (capture, trigger, reset, clock mode), high level is active (gating, encoding mode)

**Bit 12 CC3E:** Channel 3 enable.

**Input mode:**

- 0: capture disabled
- 1: capture enabled

**Output mode:**

- 0: output disabled
- 1: output enabled

**Bit 11 CC2NP:** Output reverse polarity, this bit must be set to 0 in output mode, see CC2P in input mode.

**Bit 10 RESERVED:** Must be kept, and cannot be modified.

**Bit 9 CC2P:** Output polarity, must work together with CC2NP (it is best to configure the polarity before mode selection to prevent internal signal inversion. Redirect triggers an unknown error).

**Output mode:**

- 0: high level output polarity
- 1: low level output polarity

**Input mode, {CC2NP, CC2P}:**

- 00: channel input active rising edge is active (capture, trigger, reset, clock mode), high level is active (gating, encoding mode)
- 01: channel input falling edge is active (capture, trigger, reset, clock mode), low level is active (gating, encoding mode)
- 10: reserved
- 11: both rising and falling edges of channel input are active (capture, trigger, reset, clock mode), high level is active (gating, encoding mode)

**Bit 8 CC2E:** Channel 2 enable.

**Input mode:**

- 0: capture disabled
- 1: capture enabled

**Output mode:**

- 0: output disabled
- 1: output enabled

**Bit 7 CC1NP:** Output reverse polarity, this bit must be set to 0 in output mode, see CC1P in input mode.

**Bit 6 RESERVED:** Must be kept, and cannot be modified.

**Bit 5 CC1P:** Output polarity, must work together with CC1NP (it is best to configure the polarity before mode selection to prevent internal signal inversion. Redirect triggers an unknown error).

**Output mode:**

- 0: high level output polarity
- 1: low level output polarity

**Input mode, {CC1NP, CC1P}:**

- 00: channel input active rising edge is active (capture, trigger, reset, clock mode), high level is active (gating, encoding mode)
- 01: channel input falling edge is active (capture, trigger, reset, clock mode), low level is active (gating, encoding mode)
- 10: reserved
- 11: both rising and falling edges of channel input are active (capture, trigger, reset, clock mode), high level is active (gating, encoding mode)

**Bit 4 CC1E:** Channel 1 enable.

**Input mode:**

- 0: capture disabled
- 1: capture enabled

**Output mode:**

- 0: output disabled
- 1: output enabled

**Bit 3 CC0NP:** Output reverse polarity, this bit must be set to 0 in output mode, see CC0P in input mode.

**Bit 2 RESERVED:** Must be kept, and cannot be modified.

**Bit 1 CC0P:** Output polarity, must work together with CC0NP (it is best to configure the polarity before mode selection to prevent internal signal inversion. Redirect triggers an unknown error).

**Output mode:**

- 0: high level output polarity
- 1: low level output polarity

**Input mode, {CC0NP, CC0P}:**

- 00: channel input active rising edge is active (capture, trigger, reset, clock mode), high level is active (gating, encoding mode)
- 01: channel input falling edge is active (capture, trigger, reset, clock mode), low level is active (gating, encoding mode)
- 10: reserved
- 11: both rising and falling edges of channel input are active (capture, trigger, reset, clock mode), high level is active (gating, encoding mode)

**Bit 0 CC0E:** Channel 0 enable.

**Input mode:**

- 0: capture disabled
- 1: capture enabled

**Output mode:**

- 0: output disabled
- 1: output enabled

## 22.17.10 GPTIM\_CNT

Offset: 0x024

Reset Value: 0x0000

15-0
CNT
rw-0h

**Bits 15-0 CNT:** counter count value.

## 22.17.11 GPTIM\_PSC

Offset: 0x028

Reset Value: 0x0000

15-0
PSC
rw-0h

**Bits 15-0 PSC:** Prescaler value is PSC+1.

### 22.17.12 GPTIM\_ARR

Offset: 0x02C

Reset Value: 0xFFFF

15-0
PSC
rw-FFFFh

**Bits 15-0 ARR:** counter reload value.

### 22.17.13 GPTIM\_CCR0

Offset: 0x034

Reset Value: 0x0000

15-0
CCR0
rw-0h

**Bits 15-0 CCR0:** In output mode this register saves the comparison value written by the user for comparison with CNT; in input mode this register holds the captured value and is read-only.

### 22.17.14 GPTIM\_CCR1

Offset: 0x038

Reset Value: 0x0000

15-0
CCR1
rw-0h

**Bits 15-0 CCR1:** In output mode this register saves the comparison value written by the user for comparison with CNT; in input mode this register holds the captured value and is read-only.

### 22.17.15 GPTIM\_CCR2

Offset: 0x03C

Reset Value: 0x0000

15-0
CCR2
rw-0h

**Bits 15-0 CCR2:** In output mode this register saves the comparison value written by the user for comparison with CNT; in input mode this register holds the captured value and is read-only.

### 22.17.16 GPTIM\_CCR3

Offset: 0x040

Reset Value: 0x0000

15-0
CCR3
rw-0h

**Bits 15-0 CCR3:** In output mode this register saves the comparison value written by the user for comparison with CNT; in input mode this register holds the captured value and is read-only.

### 22.17.17 GPTIM\_DCR

Offset: 0x048

Reset Value: 0x0000

15-13	12-8	7-5	4-0
RESERVED	DBL	RESERVED	DBA
r-0h	rw-0h	r-0h	rw-0h

**Bits 15-13 RESERVED:** Must be kept, and cannot be modified.

**Bits 12-8 DBL:** DMA continuous read and write length.

- 00000: 1 transfer
- 00001: 2 transfers
- 00010: 3 transfers
- 00011: 4 transfers
- 00100: 5 transfers
- 00101: 6 transfers
- 00110: 7 transfers
- 00111: 8 transfers
- 01000: 9 transfers

- 01001: 10 transfers
- 01010: 11 transfers
- 01011: 12 transfers
- 01100: 13 transfers
- 01101: 14 transfers
- 01110: 15 transfers
- 01111: 16 transfers
- 10000: 17 transfers
- 10001: 18 transfers

**Bits 7-5 RESERVED:** Must be kept, and cannot be modified.

**Bits 4-0 DBA:** DMA base address for continuous reading and writing.

- 00000: CR1 register
- 00001: CR2 register
- 00010: SMCR register
- 00011: DIER register
- 00100: SR register
- 00101: EGR register
- 00110: CCMR1 register
- 00111: CCMR2 register
- 01000: CCER register
- 01001: CNT register
- 01010: PSC register
- 01011: ARR register
- 01100: reserved register with offset 0x30
- 01101: CCR0 register
- 01110: CCR1 register
- 01111: CCR2 register
- 10000: CCR3 register
- 10001: reserved register with offset 0x44
- 10010: DCR register
- 10011: DMAR register
- 10100: OR register
- 10101: reserved
- 10110: reserved
- 10111: reserved
- 11000: reserved
- 11001: reserved
- 11010: reserved
- 11011: reserved
- 11100: reserved
- 11101: reserved
- 11110: reserved
- 11111: reserved

### 22.17.18 GPTIM\_DMAR

Offset: 0x04C

Reset Value: 0x0000

15-0
DMAR
rw-0h

**Bits 15-0 DMAR:** This Register saves the value of the Register of the current DMA operation. For example, the current DMA needs to operate the TIM\_CR2 Register, then directly operating the address is equivalent to operating the TIM\_CR2 Register. Please refer to the specific Register of the Table. Values for DSTEP, DBL and DBA.

### 22.17.19 GPTIM\_OR

Offset: 0x050

Reset Value: 0x0000

#### GPTIMER0 Register structure:

15-11	10-7	6-4	3-0
RESERVED	ETR_RMP	TI3_RMP	TI0_RMP
r-0h	rw-0h	rw-0h	rw-0h

**Bits 15-11 RESERVED:** Must be kept, and cannot be modified.

**Bits 10-7 ETR\_RMP:** ETR remapping.

- 0000 : iom
- 0001: comp0
- 0010: comp1
- 0011: xo32k
- 0100: rco48m
- 0101: adcctrl\_awd0
- 0110: adcctrl\_awd1
- 0111: adcctrl\_awd2
- 1000: uart\_rx[0]
- 1001: uart\_rx[1]
- 1010: uart\_rx[2]
- 1011: uart\_rx[3]
- 1100: uart\_rx[4]
- 1101: reserved
- 1110: reserved
- 1111: reserved

**Bits 6-4 TI3\_RMP:** Channel 3 remapping.

- 000 : iom
- 001:comp0
- 010: comp1
- 011: reserved
- 100: reserved
- 101: reserved
- 110: reserved
- 111: reserved

**Bits 3-0 TI0\_RMP:** Channel 0 remapping.

- 0000 : iom
- 0001: uart\_rx[0]
- 0010: uart\_rx[1]
- 0011: uart\_rx[2]
- 0100: uart\_rx[3]
- 0101: uart\_rx[4]
- 0110: reserved
- 0111: reserved
- 1000: reserved
- 1001: reserved
- 1010: reserved
- 1011: reserved
- 1100: reserved
- 1101: reserved
- 1110: reserved
- 1111: reserved

#### GPTIMER1 Register structure:

15-2	1-0
RESERVED	TI2_RMP
r-0h	rw-0h

**Bit 15-2 RESERVED:** Must be kept, and cannot be modified.

**Bits 1-0 TI2\_RMP:** Channel 2 remapping.

- 00: iom
- 01: TIM3\_CH1
- 10: reserved
- 11: reserved

### GPTIMER2 Register structure:

15-10	9-7	6-5	4-0
RESERVED	ETR_RMP	TI1_RMP	TI0_RMP
r-0h	rw-0h	rw-0h	rw-0h

**Bits 15-11 RESERVED:** Must be kept, and cannot be modified.

**Bits 10-7 ETR\_RMP:** ETR remapping.

- 0000: ion
- 0001: comp0
- 0010: comp1
- 0011: xo32k
- 0100: reserved
- 0101: reserved
- 0110: reserved
- 0111: reserved
- 1000: reserved
- 1001: reserved
- 1010: reserved
- 1011: reserved
- 1100: reserved
- 1101: reserved
- 1110: reserved
- 1111: reserved

**Bits 6-5 TI1\_RMP:** Channel 1 remapping.

- 00: iom
- 01: comp1
- 10: reserved
- 11: reserved

**Bits 4-0 TI0\_RMP:** Channel 0 remapping.

- 00000: iom
- 00001: xo24m
- 00010: xo32m
- 00011: rco48m
- 00100: xo32k
- 00101: rco32k
- 00110: mco
- 00111: comp0
- 01000: rco3.6m
- 01001: rtc\_alarm1\_happen\_pulse
- 01010: rtc\_alarm0\_happen\_pulse
- 01011: rtc\_cyc\_counter\_pulse
- 01100: reserved
- 01101: reserved

- 01110: reserved
- 01111: reserved
- 10000: reserved
- 10001: reserved
- 10010: reserved
- 10011: reserved
- 10100: reserved
- 10101: reserved
- 10110: reserved
- 10111: reserved
- 11000: reserved
- 11001: reserved
- 11010: reserved
- 11011: reserved
- 11100: reserved
- 11101: reserved
- 11110: reserved
- 11111: reserved

#### GPTIMER3 Register structure:

15-7	6-3	2-0
RESERVED	ETR_RMP	TIO_RMP
r-0h	rw-0h	rw-0h

**Bits 15-7 RESERVED:** Must be kept, and cannot be modified.

**Bits 6-3 ETR\_RMP:** ETR remapping.

- 0000 : iom
- 0001: comp0
- 0010: comp1
- 0011: xo32k
- 0100: uart\_rx[0]
- 0101: uart\_rx[1]
- 0110: uart\_rx[2]
- 0111: uart\_rx[3]
- 1000: uart\_rx[4]
- 1001: reserved
- 1010: reserved
- 1011: reserved
- 1100: reserved
- 1101: reserved
- 1110: reserved
- 1111: reserved

**Bits 2-0 TI0\_RMP:** Channel 0 remapping.

- 000: iom
- 001: comp0
- 010: comp1
- 011: uart\_rx[0]
- 100: uart\_rx[1]
- 101: uart\_rx[2]
- 110: uart\_rx[3]
- 111: uart\_rx[4]