

HC32L021 Series

32-bit ARM[®] Cortex[®]-M0+ Microcontroller

Reference Manual

Rev1.20
August 2026

Statement

Copyright ©2026 Xiaohua Semiconductor Co., Ltd. All Rights Reserved.

All intellectual property rights related to the information contained in this document and its annexes belongs to Xiaohua Semiconductor Co., Ltd. (hereinafter referred to as "XHSC"). The customer only has the right to use the information contained in this document and its annexes. Without the written permission of XHSC, it is forbidden to extract, copy, modify, use or spread any part or all of this document in any form.

Trademark Statement

XHSC XIAOHUA SEMICONDUCTOR, **XHSC**, and other trademarks are the trademarks of XHSC. All other products or services shown on XHSC products are the property of their respective owners.

Precautions

- XHSC reserves the right to change, correct, enhance and modify the product and/or this document at any time without notice. Users can get the latest information before placing orders. XHSC products are sold in accordance with the sales terms and conditions stated in the basic purchase and sale contract.
- It is the customer's responsibility to select the appropriate XHSC product for your application and to design, validate and test your application to ensure that your application meets the appropriate standards and any safety, security or other requirements. The customer shall be solely responsible for this.
- XHSC hereby acknowledges that no intellectual property license has been granted by express or implied means.
- The resale of XHSC Products shall be invalidated by any warranty commitment of XHSC to such Products if its terms are different from those set forth herein.
- **Important Notice:** Before using this product, be sure to consult the latest official Errata Sheet to fully understand the product's known corrections and avoid usage risks.
- Information in this notification replaces information in previous versions.

Xiaohua Semiconductor Co., Ltd.

Address:	F4, Block A, No.1867 Zhongke Road, Pudong New Area, Shanghai, China
URL:	https://www.xhsc.com.cn
E-mail:	XHSC_MCU@xhsc.com.cn (Business Consulting) XHSC_FAE@xhsc.com.cn (Technical Consulting)

Preface

Bit/Bit Field

A register contains a bit (one bit wide) or a bit field (multiple bit width), and the bit field is usually described as [y:x], starting from the x bit and ending at the y bit. Unless otherwise specified, decimal numbers are used in the bit field.

It is not recommended to use unspecified fields; otherwise, unpredictable consequences may be caused.

Register Address

The address of each register can be obtained by the base address of the system address division and the offset address of the register (see the "Register List" of each module for details).

Access Type

The following abbreviations are used for register description:

Access Type	Description
RW	Readable and writable.
RO	Read only. Writes are ignored.
WO	Write only. Read value undefined.
RWA	Readable and writable. Writable only when unlocked or in safe mode.
RWC	Readable and writable. Writing any value clears this bit.
RC	Read only. Writes are ignored. This bit is cleared automatically after reading.
RS	Read only. Writes are ignored. This bit is set automatically after reading.

Number Format

- 0x prefix is used for hexadecimal numbers
- 0b prefix is used for binary numbers
- Numbers without prefix are in decimal representation

Reserved (Res.)

Registers and bit fields marked with *Reserved* are reserved for future use. These should be written to their reset value unless otherwise stated in the Register Description. The read value of Reserved bit may be different in different devices or features.

Reset Value (Reset)

The reset value indicates the value after the register is reset. Registers need to be initialized and reset before use; otherwise, unpredictable consequences may be caused.

Peripheral Availability Description

For HC32L021 Series 32-bit ARM® Cortex®-M0+ Microcontroller supported peripherals and their numbers, refer to the corresponding "Datasheet".

Security Statement

There may be potential security problems due to the use of a certain function or protocol, which need to be declared to remind users and avoid security risks.

Contents

Preface	iii
1 Product Introduction	20
2 Storage and Architectures	21
2.1 Overview	21
2.2 System Address Division.....	23
2.3 Memory and Module Address Assignment.....	24
3 Operating Mode	25
3.1 Overview.....	25
3.2 Working Mode Switching.....	25
3.3 Active Mode.....	26
3.4 Sleep Mode.....	26
3.5 DeepSleep Mode.....	27
3.6 Functional/Module Working Status.....	28
3.7 Register Description.....	29
3.7.1 System Control Register (SCB_SCR) (Cortex-M0+ Core System Control Register).....	29
4 FLASH	30
4.1 Introduction.....	30
4.2 Functional Description.....	30
4.2.1 Capacity Division.....	30
4.2.2 Read Wait Period.....	30
4.2.3 Flash Security Protection.....	30
4.3 Operation Example.....	32
4.3.1 Sector Erase.....	32
4.3.2 Chip Erase.....	32
4.3.3 Program.....	33
4.3.4 Cont-Program.....	34
4.3.5 Read.....	36
4.4 Register Description.....	38
4.4.1 Register List.....	38
4.4.2 CR Register (FLASH_CR).....	39
4.4.3 IFR Register (FLASH_IFR).....	41
4.4.4 ICLR Register (FLASH_ICLR).....	42
4.4.5 BYPASS Register (FLASH_BYPASS).....	43
4.4.6 SLOCK0 Register (FLASH_SLOCK0).....	43
4.4.7 Protection Status Register (FLASH_LOCK_ST).....	44
5 RAM	45
5.1 Introduction.....	45
5.2 Functional Description.....	45
5.2.1 RAM Address Range.....	45
5.2.2 Read and Write Bit Width.....	45
6 SYSCTRL	46
6.1 System Clock Introduction.....	46

6.1.1 Clock Architecture Diagram.....	46
6.1.2 Clock Types.....	47
6.1.3 Clock Start Process.....	48
6.2 System Clock Switching.....	49
6.2.1 System Clock Switching Introduction.....	49
6.2.2 Switching Process.....	49
6.2.3 Examples of Switching.....	50
6.2.4 Clock Switching Diagram.....	51
6.3 Clock Calibration Module.....	51
6.4 Interrupt Wakeup Control.....	52
6.4.1 Executing Interrupt Service after Waking from DeepSleep mode.....	52
6.4.2 Interrupt Service Not Executed after Waking from DeepSleep mode.....	52
6.4.3 Using Sleep-on-exit Feature.....	53
6.5 Reset Control.....	53
6.5.1 Reset Controller Overview.....	53
6.5.2 Reset Function.....	54
6.6 Register Description.....	56
6.6.1 Register List.....	56
6.6.2 System Control Register 0 (SYSCTRL0).....	57
6.6.3 System Control Register 1 (SYSCTRL1).....	59
6.6.4 System Control Register 2 (SYSCTRL2).....	60
6.6.5 RC48M Control Register (RC48M_CR).....	61
6.6.6 RCL Control Register (RCL_CR).....	62
6.6.7 XTL Control Register (XTL_CR).....	64
6.6.8 Reset Flag Register (RESET_FLAG).....	66
6.6.9 Peripheral Module Clock Control Register 0 (PERI_CLKEN0).....	68
6.6.10 Peripheral Module Clock Control Register 1 (PERI_CLKEN1).....	70
6.6.11 Peripheral Module Reset Control Register 0 (PERI_RESET0).....	71
6.6.12 Peripheral Module Reset Control Register 1 (PERI_RESET1).....	73
6.6.13 Peripheral Debug Function Enable Register (DEBUG_ACTIVE).....	74
7 GPIO.....	75
7.1 Introduction.....	75
7.2 Features.....	75
7.3 Functional Description.....	76
7.3.1 Port Configuration Function.....	76
7.3.2 Port Write.....	77
7.3.3 Port Read.....	77
7.3.4 Port Digital Multiplexing Function.....	79
7.3.5 Port Interrupt Function.....	80
7.3.6 Port Input Filter Function.....	80
7.3.7 Port Reset Status.....	80
7.3.8 HSI2C Communication Level Selection Function.....	80
7.4 Operation Example.....	81
7.4.1 Port Multiplexing Configured as Analog Port Operation Flow.....	81
7.4.2 Port Multiplexing Configured as Digital Universal Port Operation Flow.....	81

7.4.3 Port Multiplexing Configured as Digital Function Port Operation Flow.....	81
7.4.4 Port Multiplexing Configured as Debug Test Port Operation Flow.....	81
7.4.5 Port Multiplexing Configured as Infrared Output Signal Operation Flow.....	81
7.4.6 Port High Level Interrupt Operation Flow.....	82
7.4.7 Port Low Level Interrupt Operation Flow.....	82
7.4.8 Port Rising Edge Interrupt Operation Flow.....	82
7.4.9 Port Falling Edge Interrupt Operation Flow.....	82
7.4.10 Port Pull-Up Enable Configuration Operation Flow.....	82
7.4.11 Port Open-Drain Output Configuration Operation Flow.....	82
7.4.12 Port Bit Setting Operation Flow.....	83
7.4.13 Port Bit Clearing Operation Flow.....	83
7.4.14 Port Bit Setting and Clearing Operation Flow.....	83
7.4.15 Port Input Filtering Operation Flow.....	83
7.5 Register Description.....	84
7.5.1 Register List.....	84
7.5.2 Port General Register.....	86
7.5.3 Port Auxiliary Register.....	99
7.5.4 Port Function Selection Register.....	104
8 NVIC.....	105
8.1 Introduction.....	105
8.2 Functional Description.....	105
8.2.1 Interrupt Priority.....	105
8.2.2 Interrupt Digraph.....	106
8.2.3 Interrupt Input and Pending Behavior.....	107
8.2.4 Interrupt Waiting.....	109
8.2.5 Interrupt Source.....	109
8.2.6 Interrupt Structure Diagram.....	111
8.3 Operation Example.....	113
8.3.1 External Interrupt Enable.....	113
8.3.2 NVIC Interrupt Enable and Clear Enable.....	113
8.3.3 NVIC Interrupt Pending and Clear Pending.....	113
8.3.4 NVIC Interrupt Priority.....	113
8.3.5 NVIC Interrupt Mask.....	113
8.4 Register Description.....	115
8.4.1 Register List.....	115
8.4.2 Interrupt Enable Set Register (NVIC_ISER).....	116
8.4.3 Interrupt Enable Clear Register (NVIC_ICER).....	116
8.4.4 Interrupt Pending State Set Register (NVIC_ISPR).....	117
8.4.5 Interrupt Pending State Clear Register (NVIC_ICPR).....	117
8.4.6 Interrupt Priority Register (NVIC_IPR0).....	118
8.4.7 Interrupt Priority Register (NVIC_IPR1).....	118
8.4.8 Interrupt Priority Register (NVIC_IPR2).....	119
8.4.9 Interrupt Priority Register (NVIC_IPR3).....	119
8.4.10 Interrupt Priority Register (NVIC_IPR4).....	120
8.4.11 Interrupt Priority Register (NVIC_IPR5).....	120

8.4.12 Interrupt Priority Register (NVIC_IPR6).....	121
8.4.13 Interrupt Priority Register (NVIC_IPR7).....	121
8.4.14 Interrupt Mask Special Register (PRIMASK).....	122
9 SYSTICK.....	123
9.1 Introduction.....	123
9.2 Operation Example.....	123
9.2.1 Set SysTick.....	123
9.3 Register Description.....	124
9.3.1 Register List.....	124
9.3.2 SysTick Control and Status Register (CTRL).....	125
9.3.3 SysTick Reload Register (LOAD).....	126
9.3.4 SysTick Current Value Register (VAL).....	127
9.3.5 SysTick Calibration Value Register (CALIB).....	127
10 CTRIM.....	128
10.1 Introduction.....	128
10.2 Features.....	128
10.3 Functional Description.....	128
10.3.1 RC48M Clock Calibration Mode.....	128
10.3.2 RCL Clock Calibration Mode.....	129
10.3.3 Low Power Timer Mode.....	130
10.4 Operation Example.....	130
10.4.1 RC48M Clock Automatic Calibration Example.....	130
10.4.2 RCL Clock Automatic Calibration Example.....	131
10.5 Register Description.....	132
10.5.1 Register List.....	132
10.5.2 Autoload Register (CTRIM_ARR).....	133
10.5.3 Error Counter Register (CTRIM_CNT).....	133
10.5.4 Control Register 0 (CTRIM_CR0).....	134
10.5.5 Control Register 1 (CTRIM_CR1).....	136
10.5.6 Interrupt Enable Register (CTRIM_IER).....	138
10.5.7 Interrupt and Status Register (CTRIM_ISR).....	139
10.5.8 Interrupt Status Flag Register (CTRIM_ICR).....	140
10.5.9 Error Result Register (CTRIM_FCAP).....	141
10.5.10 TRIM Result Register (CTRIM_TVAL).....	142
10.5.11 Error Upper Limit Register (CTRIM_FLIM).....	142
11 CTIM.....	143
11.1 Introduction.....	143
11.2 Features.....	144
11.2.1 BTIM Features.....	144
11.2.2 GTIM Features.....	144
11.3 Functional Description.....	145
11.3.1 BTIM Functional Description.....	145
11.3.2 GTIM Functional Description.....	150
11.4 Register Description.....	158
11.4.1 Register List.....	158

11.4.2 BTIM Register.....	160
11.4.3 GTIM Register.....	166
12 ATIM3.....	175
12.1 Introduction.....	175
12.2 Features.....	175
12.3 Functional Description.....	176
12.3.1 Timer Clock.....	176
12.3.2 Timer Counter.....	176
12.3.3 Timer Prescaler.....	176
12.3.4 Mode 0 Count Timer Function.....	177
12.3.5 Mode 1 Pulse Width Measurement PWC.....	179
12.3.6 Mode2/3 Compare Capture Mode.....	182
12.3.7 Mode2/3 Slave Mode.....	200
12.3.8 Quadrature Encoder Counting Function.....	202
12.3.9 Timer Triggers ADC.....	203
12.3.10 Brake Control.....	204
12.3.11 Interrupt.....	204
12.3.12 Timer Interconnection.....	205
12.3.13 Input Interconnection.....	205
12.4 Operation Example.....	205
12.4.1 Mode0 Counting Timer Function Configuration Example.....	205
12.4.2 Mode1 Pulse Width Measurement PWC Configuration Example.....	205
12.4.3 Mode2/3 Compare Capture Mode Configuration Example.....	206
12.4.4 Complementary PWM Output B Compare as ADC Trigger Function.....	207
12.5 Register Description.....	209
12.5.1 Register List.....	209
12.5.2 Mode0 Count Timer Register.....	211
12.5.3 Mode1 Pulse Width Measurement PWC Register.....	217
12.5.4 Mode2/3 Compare Capture Mode Register.....	225
13 RTC.....	245
13.1 Introduction.....	245
13.2 Features.....	245
13.3 Functional Description.....	246
13.3.1 Function Block Diagram.....	247
13.3.2 Power-on Setting.....	247
13.3.3 Clock Error Compensation.....	247
13.3.4 Timestamp Function.....	249
13.3.5 Wake-up Timer.....	249
13.3.6 RTC Interrupt.....	250
13.4 Operation Example.....	250
13.4.1 RTC Count Start Setting.....	250
13.4.2 Read Count Register.....	251
13.4.3 Write Count Register.....	251
13.4.4 Alarm Clock Setting.....	252
13.4.5 1Hz Output.....	252

13.5 Register Description.....	254
13.5.1 Register List.....	254
13.5.2 Control Register 0 (RTC_CR0).....	255
13.5.3 Control Register 1 (RTC_CR1).....	257
13.5.4 Second Count Register (RTC_SEC).....	258
13.5.5 Minute Count Register (RTC_MIN).....	258
13.5.6 Hour Count Register (RTC_HOUR).....	259
13.5.7 Day Count Register (RTC_DAY).....	261
13.5.8 Week Count Register (RTC_WEEK).....	262
13.5.9 Month Count Register (RTC_MON).....	263
13.5.10 Year Count Register (RTC_YEAR).....	264
13.5.11 Clock Error Compensation Register (RTC_COMPEN).....	265
13.5.12 High-Speed Compensation Accuracy Parameter Register (RTC_HCOMP).....	267
13.5.13 Control Register 2 (RTC_CR2).....	268
13.5.14 Alarm A Register (RTC_ALMA).....	269
13.5.15 Alarm B Register (RTC_ALMB).....	271
13.5.16 Timestamp Time Register (RTC_TSTR).....	273
13.5.17 Timestamp Date Register (RTC_TSDR).....	274
13.5.18 Wake-Up Timer Register (RTC_WUTR).....	275
13.5.19 Interrupt Enable Register (RTC_IER).....	275
13.5.20 Interrupt Flag Register (RTC_IFR).....	277
13.5.21 Interrupt Clear Register (RTC_ICR).....	278
13.5.22 Protection Register (RTC_WPR).....	279
14 IWDT.....	280
14.1 Introduction.....	280
14.2 Features.....	280
14.3 Functional Description.....	280
14.3.1 Function Block Diagram.....	280
14.3.2 Window Watchdog Mode.....	280
14.3.3 Independent Watchdog Mode.....	281
14.3.4 Watchdog Overflow Handling.....	281
14.3.5 Watchdog Pause.....	281
14.3.6 Watchdog Stop and Run Again.....	281
14.3.7 Register Write Protection.....	281
14.4 Operation Example.....	281
14.4.1 Configure IWDT as Window Watchdog.....	281
14.4.2 Configure IWDT as Independent Watchdog.....	282
14.4.3 Watchdog Timeout Time (10kHz Independent Clock).....	282
14.5 Register Description.....	283
14.5.1 Register List.....	283
14.5.2 Key Value Register (IWDT_KR).....	284
14.5.3 Control Register (IWDT_CR).....	285
14.5.4 Reload Register (IWDT_ARR).....	286
14.5.5 Status Register (IWDT_SR).....	287
14.5.6 Window Register (IWDT_WINR).....	288

15 LPUART.....	289
15.1 Introduction.....	289
15.2 Features.....	289
15.3 Functional Description.....	290
15.3.1 Function Block Diagram.....	290
15.3.2 Clock Description.....	290
15.3.3 Working Mode.....	290
15.3.4 Baud Rate Generation.....	296
15.3.5 Frame Error Detection.....	299
15.3.6 Multi-Machine Communication.....	299
15.3.7 Hardware Flow Control.....	300
15.3.8 Receive/Transmit Buffer.....	301
15.4 Operation Example.....	302
15.4.1 Transmit Data Example.....	302
15.4.2 Receive Data Example.....	302
15.5 Register Description.....	303
15.5.1 Register List.....	303
15.5.2 Data Register (LPUART_SBUF).....	304
15.5.3 Control Register (LPUART_SCON).....	305
15.5.4 Address Register (LPUART_SADDR).....	308
15.5.5 Address Mask Register (LPUART_SADEN).....	308
15.5.6 Interrupt Flag Bit Register (LPUART_ISR).....	309
15.5.7 Interrupt Flag Bit Clear Register (LPUART_ICR).....	310
15.5.8 Baud Rate Register (LPUART_SCNT).....	310
15.5.9 Baud Rate Generate Register (LPUART_BSEL).....	311
15.5.10 9600/4800 Baud Rate Modulator Register (LPUART_MODU).....	312
16 SPI.....	314
16.1 Introduction.....	314
16.2 Features.....	314
16.3 Functional Description.....	314
16.3.1 Function Block Diagram.....	314
16.3.2 Communication Format and Timing.....	315
16.3.3 Slave Select Pin Configuration.....	316
16.3.4 Full-duplex Communication.....	317
16.3.5 Single-wire Half-duplex Communication.....	318
16.3.6 Simplex Communication.....	318
16.3.7 Multi-machine Communication.....	319
16.3.8 Master Mode Delayed Sampling.....	320
16.3.9 Status Flags and Error Flags.....	321
16.3.10 Interrupt Generation and Clearing.....	324
16.4 Operation Example.....	324
16.4.1 Example of Sending and Receiving Data in Full-duplex by the Master.....	324
16.4.2 Example of Sending and Receiving Data in Full-duplex by the Slave.....	325
16.4.3 Single-wire Half-duplex Sending and Receiving Data Example.....	325
16.5 Register Description.....	327

16.5.1 Register List.....	327
16.5.2 SPI Control Register 0 (SPI_CR0).....	328
16.5.3 SPI Control Register 1 (SPI_CR1).....	330
16.5.4 SPI Single-wire Half-duplex Output Enable Register (SPI_HDOE).....	331
16.5.5 SPI Internal Slave Select Register (SPI_SSI).....	331
16.5.6 SPI Status Register (SPI_SR).....	332
16.5.7 SPI Interrupt Clear Register (SPI_ICR).....	334
16.5.8 SPI Data Register (SPI_DR).....	335
17 HSI2C.....	336
17.1 Introduction.....	336
17.2 Features.....	336
17.3 Functional Description.....	337
17.3.1 System Block Diagram.....	337
17.3.2 Protocol Description.....	337
17.3.3 Master Mode.....	338
17.3.4 Slave Mode.....	342
17.3.5 Communication Level Selection Function.....	345
17.4 Operation Example.....	345
17.4.1 Master Transmission Mode.....	345
17.4.2 Master Receiving Mode.....	345
17.4.3 Slave Transmission Mode.....	346
17.4.4 Slave Receiving Mode.....	346
17.4.5 Master High-Speed Mode Switching.....	347
17.5 Register Description.....	348
17.5.1 Register List.....	348
17.5.2 Master Control Register (I2C_MCR).....	349
17.5.3 Master Status Register (I2C_MSR).....	350
17.5.4 Master Interrupt Enable Register (I2C_MIER).....	352
17.5.5 Master Status Clear Register (I2C_MSCR).....	353
17.5.6 Master Configuration 1 Register (I2C_MCFGR1).....	354
17.5.7 Master Configuration 2 Register (I2C_MCFGR2).....	356
17.5.8 Master Configuration 3 Register (I2C_MCFGR3).....	358
17.5.9 Master Clock Configuration Register 0 (I2C_MCCR0).....	359
17.5.10 Master Clock Configuration Register 1 (I2C_MCCR1).....	361
17.5.11 Master Transmit Data Register (I2C_MTDR).....	363
17.5.12 Master Receive Data Register (I2C_MRDR).....	364
17.5.13 Master Data Configuration Register (I2C_MDMR).....	365
17.5.14 Slave Control Register 0 (I2C_SCR0).....	366
17.5.15 Slave Status Register (I2C_SSR).....	367
17.5.16 Slave Interrupt Enable Register (I2C_SIER).....	370
17.5.17 Slave Status Clear Register (I2C_SSCR).....	372
17.5.18 Slave Control 1 Register (I2C_SCR1).....	373
17.5.19 Slave Control 2 Register (I2C_SCR2).....	376
17.5.20 Slave Transmit ACK Register (I2C_STAR).....	378
17.5.21 Slave Transmit Data Register (I2C_STDR).....	379

17.5.22 Slave Receive Data Register (I2C_SRDR).....	380
17.5.23 Slave Address Register (I2C_SADDR).....	381
17.5.24 Slave Address Status Register (I2C_SASR).....	382
18 ADC.....	383
18.1 Introduction.....	383
18.2 Features.....	383
18.3 Functional Description.....	383
18.3.1 ADC Block Diagram.....	383
18.3.2 Conversion Timing and Conversion Speed.....	383
18.3.3 Scan Conversion Mode.....	384
18.3.4 ADC Conversion External Trigger Source.....	385
18.3.5 ADC Interrupt.....	385
18.4 Operation Example.....	385
18.4.1 Start the ADC SQR Scan Conversion Operation Process through the ADC_SQRSTART.START Bit.....	385
18.4.2 External Trigger Start ADC SQR Scan Conversion Operation Flow.....	386
18.5 Register Description.....	387
18.5.1 Register List.....	387
18.5.2 ADC Basic Configuration Register 0 (ADC_CR0).....	388
18.5.3 ADC SQR Scan Conversion Channel Configuration Register 0 (ADC_SQR0).....	390
18.5.4 ADC SQR Scan Conversion Channel Configuration Register 1 (ADC_SQR1).....	391
18.5.5 ADC SQR Scan Conversion Channel x Result Register (ADC_SQRRESULTx) (x=0-3).....	392
18.5.6 ADC Interrupt Flag Register (ADC_IFR).....	393
18.5.7 ADC Interrupt Clear Register (ADC_ICR).....	394
18.5.8 ADC SQR Scan Conversion External Interrupt Trigger Source Configuration Register (ADC_EXTTRIGGER).....	395
18.5.9 ADC SQR Scan Conversion Start Control Register (ADC_SQRSTART).....	396
19 VC.....	397
19.1 Introduction.....	397
19.2 Features.....	397
19.3 Functional Description.....	397
19.3.1 Function Block Diagram.....	397
19.3.2 Setup/Response Time.....	398
19.3.3 Digital Filtering.....	398
19.3.4 Hysteresis Function.....	399
19.3.5 Configuration of Comparison Result Polarity.....	399
19.3.6 Window Comparison Function.....	399
19.3.7 Blank Window.....	399
19.4 Register Description.....	400
19.4.1 Register List.....	400
19.4.2 Control Register 0 (VCx_CR0) (x=0-1).....	401
19.4.3 Control Register 1 (VCx_CR1) (x=0-1).....	403
19.4.4 Control Register 2 (VCx_CR2) (x=0-1).....	405
19.4.5 Status Register (VCx_SR) (x=0-1).....	406
19.4.6 Offset Trim Register (VC_TRIM).....	407
20 LVD.....	408
20.1 Introduction.....	408

20.2 Features.....	408
20.3 Functional Description.....	408
20.3.1 LVD Block Diagram.....	408
20.3.2 Digital Filtering.....	408
20.3.3 Hysteresis Function.....	409
20.4 Operation Example.....	410
20.4.1 LVD Configured as Low Voltage Reset.....	410
20.4.2 LVD Configured as Voltage Change Interrupt.....	410
20.5 Register Description.....	411
20.5.1 Register List.....	411
20.5.2 LVD Configuration Register (LVD_CR).....	412
20.5.3 LVD Status Register (LVD_SR).....	415
21 TRNG.....	416
21.1 Introduction.....	416
21.2 Functional Description.....	416
21.2.1 True Random Number Generation Principle.....	416
21.3 Operation Example.....	416
21.3.1 Generating 64bits True Random Number (The First Time After Power-On).....	416
21.3.2 Generating 64bits True Random Number (Not Generated for the First Time After Power-on).....	417
21.4 Register Description.....	418
21.4.1 Register List.....	418
21.4.2 Control Register (TRNG_CR).....	419
21.4.3 Mode Register (TRNG_MODE).....	420
21.4.4 Data Register 0 (TRNG_DATA0).....	421
21.4.5 Data Register 1 (TRNG_DATA1).....	421
22 SWD Debug Interface.....	422
22.1 Introduction.....	422
22.2 Functional Description.....	422
22.2.1 Debug System Diagram.....	422
22.2.2 ARM® Reference Documentation.....	423
22.2.3 Debug Port Pins.....	423
22.2.4 SWD Port.....	424
22.2.5 Core Debugging.....	427
22.2.6 BPU (Breakpoint Unit).....	427
22.2.7 DWT (Data Watchpoint).....	428
22.2.8 MCU Debug Component (DBG).....	428
23 Device Electronic Signature.....	429
23.1 Introduction.....	429
23.2 Functional Description.....	429
23.2.1 Product Unique Identifier (UID).....	429
23.3 Register Description.....	430
23.3.1 Product Unique Identifier (UID) Register (80 bits).....	430
23.3.2 Product Model Register.....	430
23.3.3 FLASH Capacity Register.....	431

23.3.4 RAM Capacity Register.....	431
23.3.5 Pin Count/Flash Sector Size Configuration Register.....	432
A Glossary.....	433
Revision History.....	434

List of Figures

Figure 2-1 System Architecture Schematic Diagram.....	22
Figure 3-1 Working Mode State Transition Diagram.....	25
Figure 6-1 Clock Architecture Diagram.....	47
Figure 6-2 Crystal Oscillator Clock Startup Diagram.....	49
Figure 6-3 Clock Switching Diagram.....	51
Figure 6-4 Reset Source Diagram.....	54
Figure 7-1 Port Circuit Diagram.....	76
Figure 7-2 AHB Bus Port Changes with System Clock.....	77
Figure 7-3 Port Pin Data Read Synchronization Diagram.....	78
Figure 7-4 GPIO Port Input Filter Function Timing Diagram.....	80
Figure 8-1 Interrupt Vector Table.....	107
Figure 8-2 Interrupt Active and Pending States.....	108
Figure 8-3 Interrupt Pending State is Cleared and Then Reasserted.....	108
Figure 8-4 When the Interrupt Exits, If the Interrupt Request Remains High, It Will Cause the Interrupt Processing to Be Executed Again.....	108
Figure 8-5 Interrupt Pending Interrupts Generated During Interrupt Processing Can Also Be Acknowledged.....	109
Figure 8-6 Interrupt Structure Diagram.....	112
Figure 10-1 RC48M Clock Calibration Operational Block Diagram.....	128
Figure 10-2 RCL Clock Calibration Operational Block Diagram.....	129
Figure 10-3 General Low Power Timer Scenario Operational Block Diagram.....	130
Figure 11-1 BTIM Block Diagram.....	145
Figure 11-2 BTIM Filter Unit Diagram.....	145
Figure 11-3 BTIM Counter Waveform.....	146
Figure 11-4 BTIM Timing Mode Block Diagram.....	146
Figure 11-5 BTIM Counter Mode Block Diagram.....	147
Figure 11-6 BTIM Trigger Start Mode Block Diagram.....	147
Figure 11-7 BTIM Trigger Start Mode Counting Diagram.....	148
Figure 11-8 BTIM Gating Mode Block Diagram.....	148
Figure 11-9 BTIM Gating Mode Counting Diagram.....	149
Figure 11-10 BTIM Timer Cascade Diagram.....	149
Figure 11-11 GTIM Block Diagram.....	150
Figure 11-12 GTIM Filter Unit Diagram.....	150
Figure 11-13 GTIM Counter Waveform.....	151
Figure 11-14 GTIM Timing Mode Block Diagram.....	152
Figure 11-15 GTIM Counter Mode Block Diagram.....	152
Figure 11-16 GTIM Trigger Start Mode Block Diagram.....	153
Figure 11-17 GTIM Trigger Start Mode Counting Diagram.....	153
Figure 11-18 GTIM Gating Mode Block Diagram.....	154
Figure 11-19 GTIM Gating Mode Counting Diagram.....	154
Figure 11-20 GTIM Capture Function Diagram.....	155
Figure 11-21 GTIM Compare Function Diagram.....	156
Figure 12-1 ATIM3 Block Diagram.....	176
Figure 12-2 Mode0 Count Timer Free Counting Block Diagram.....	177
Figure 12-3 Mode0 Count Timer Reload Counting Block Diagram.....	177
Figure 12-4 Mode0 Count Timer 16-Bit Reload Counting Waveform.....	178
Figure 12-5 Mode0 Count Timer 32-Bit Free Counting Waveform.....	178
Figure 12-6 Mode0 Count Timer Free Counting Timing Diagram.....	179
Figure 12-7 Mode0 Count Timer Reload Counting Timing Diagram (Prescaler=2).....	179
Figure 12-8 Mode1 Pulse Width Measurement PWC Function Block Diagram.....	180
Figure 12-9 Mode1: Pulse Width Measurement PWC High-Level Pulse Width Measurement.....	181

Figure 12-10	Mode1: Pulse Width Measurement PWC Falling Edge to Falling Edge Period Measurement.....	181
Figure 12-11	Mode1: Pulse Width Measurement PWC Rising Edge to Rising Edge Period Measurement.....	181
Figure 12-12	Mode1: Pulse Width Measurement PWC Rising Edge to Rising Edge Period Measurement Single Mode.....	182
Figure 12-13	Mode 2/3 Compare Capture Mode Counter Block Diagram.....	182
Figure 12-14	Mode 2 Compare Capture Mode Up-Counting Without Prescaler.....	183
Figure 12-15	Mode 2 Compare Capture Mode Up-Counting With Prescaler.....	183
Figure 12-16	Mode 2 Compare Capture Mode Down-Counting Without Prescaler.....	184
Figure 12-17	Mode 2 Compare Capture Mode Down-Counting With Prescaler.....	184
Figure 12-18	Mode2 Edge-Aligned Timer Waveform (M23CR.DIR=0).....	184
Figure 12-19	Mode2 Edge-Aligned Timer Waveform (M23CR.DIR=1).....	184
Figure 12-20	Mode3 Compare Capture Mode Up/Down-Counting with Prescaler.....	185
Figure 12-21	Mode3 Center-Aligned Counter Waveform.....	185
Figure 12-22	Repetition Counter Generates Update Timing (RCR.OV=0, RCR.UND=0).....	186
Figure 12-23	Update Timing Generated by Repetition Counter when Overflow Is Masked (RCR.OV=1, RCR.UND=0).....	186
Figure 12-24	Update Timing Generated by Repetition Counter when Underflow Is Masked (RCR.OV=0, RCR.UND=1).....	186
Figure 12-25	Mode 3 Triangular Wave Mode with Cache Enable.....	187
Figure 12-26	Mode 3 Triangular Wave Mode with Cache Invalid.....	187
Figure 12-27	Mode 2 Sawtooth Wave Mode Up-Counting with Cache Enable.....	188
Figure 12-28	Mode 2 Sawtooth Wave Mode Up-Counting with Cache Invalid.....	188
Figure 12-29	Mode 2 Sawtooth Wave Mode Down-Counting with Cache Enable.....	188
Figure 12-30	Mode 2 Sawtooth Wave Mode Down-Counting with Cache Invalid.....	189
Figure 12-31	Mode2 Sawtooth Wave Down-Counting Mode with Compare Cache Enable.....	189
Figure 12-32	OCREF Output Block Diagram.....	190
Figure 12-33	Sawtooth Counting Single Point Comparison OCREF Output Waveform (OCMyx=0b111).....	191
Figure 12-34	Triangular Counting Single Point Comparison OCREF Output Waveform (OCMyx=0b111).....	191
Figure 12-35	Sawtooth Counting Dual Point Comparison OCREF Output Waveform (OCMyx=0b111).....	191
Figure 12-36	Triangular Counting Dual Point Comparison OCREF Output Waveform (OCMyx=0b111).....	192
Figure 12-37	Independent PWM Output Block Diagram.....	193
Figure 12-38	PWM Output Waveform When CCPy=0.....	193
Figure 12-39	PWM Output Waveform When CCPy=1.....	193
Figure 12-40	Complementary PWM Output Block Diagram.....	194
Figure 12-41	Complementary PWM Output Waveform.....	194
Figure 12-42	Complementary PWM Output Waveform (with Dead Time).....	195
Figure 12-43	Dead Time.....	195
Figure 12-44	Triangular Wave Mode Single Pulse Counting.....	196
Figure 12-45	Sawtooth Up-Counting Single Pulse Mode.....	196
Figure 12-46	Sawtooth Down-Counting Single Pulse Mode.....	197
Figure 12-47	Compare Interrupt Diagram.....	198
Figure 12-48	Capture Function Block Diagram.....	198
Figure 12-49	Capture Timing Diagram.....	198
Figure 12-50	ATIM3 CH0A Capture Input Port Selection.....	199
Figure 12-51	CH0B Capture Input Port Selection.....	199
Figure 12-52	Mode2/3 Slave Mode Diagram.....	200
Figure 12-53	Mode2/3 Slave Mode Gating Function.....	201
Figure 12-54	Mode2/3 Slave Mode Trigger and Reset Function.....	202
Figure 12-56	Quadrature Encoder Counting Mode3 Counting Diagram.....	203
Figure 12-57	Timer Triggers ADC.....	204
Figure 13-1	RTC Block Diagram.....	247
Figure 14-1	IWDT Function Block Diagram.....	280
Figure 15-1	LPUART Function Block Diagram.....	290
Figure 15-2	Mode0 Data Transmitting.....	293
Figure 15-3	Mode0 Data Receiving.....	293

Figure 15-4	Mode1 Data Transmitting.....	294
Figure 15-5	Mode1 Data Receiving.....	294
Figure 15-6	Mode2/Mode3 Data Transmitting.....	294
Figure 15-7	Mode2/Mode3 Data Receiving.....	295
Figure 15-8	LPUART Hardware Flow Control.....	300
Figure 15-9	RTS Hardware Flow Control Signal.....	300
Figure 15-10	CTS Hardware Flow Control Signal.....	301
Figure 15-11	Receive Buffer.....	301
Figure 15-12	Transmit Buffer.....	301
Figure 16-1	SPI Block Diagram.....	314
Figure 16-2	SPI Communication Format Timing Diagram.....	316
Figure 16-3	Slave Select NSS Pin Configuration.....	317
Figure 16-4	SPI Full-duplex Communication.....	318
Figure 16-5	SPI Single-wire Half-duplex Communication.....	318
Figure 16-6	SPI Simplex Transmit Communication Diagram.....	319
Figure 16-7	SPI Simplex Receive Communication Diagram.....	319
Figure 16-8	SPI Multi-machine System.....	320
Figure 16-9	Explanation Diagram of General Flags in SPI Master Mode.....	323
Figure 16-10	SPI Slave Mode General Flag Bit Description Diagram.....	323
Figure 17-1	I2C System Block Diagram.....	337
Figure 17-2	I2C Bus Timing Diagram.....	338
Figure 17-3	I2C Bus Timing Parameters Diagram.....	339
Figure 18-1	ADC Block Diagram.....	383
Figure 18-2	ADC Conversion Timing Diagram.....	384
Figure 18-3	Examples of ADC SQR Scan Conversion Process.....	384
Figure 19-1	VC Function Block Diagram.....	397
Figure 19-2	VC Filter Response Time.....	398
Figure 19-3	VC Hysteresis Function.....	399
Figure 19-4	VC Blank Window.....	399
Figure 20-1	LVD Block Diagram.....	408
Figure 20-2	LVD Filter Output.....	409
Figure 20-3	LVD Hysteresis Response.....	410
Figure 21-1	True Random Number Generator Function Block Diagram.....	416
Figure 22-1	Debug Support Block Diagram.....	422

List of Tables

Table 2-1	Address Division Table.....	24
Table 3-1	Reset Type Response Capabilities in Different Operational Mode.....	25
Table 3-2	Peripheral Working Status Table in Each Mode.....	28
Table 4-1	Flash Capacity Division Table.....	30
Table 4-3	Flash Controller (FLASH) Register Offset.....	38
Table 5-1	RAM Address Mapping.....	45
Table 6-3	System Controller (SYSCTRL) Register Offset.....	56
Table 7-1	Port Status Truth Table.....	77
Table 7-2	Port Multiplexing Table.....	79
Table 7-3	General-Purpose I/O (GPIO) Register Offset.....	84
Table 8-1	Cortex-M0+ Processor Interrupt List.....	105
Table 8-4	Interrupt Controller (NVIC) Register Offset Address.....	115
Table 9-2	SysTick Timer Register Offset.....	124
Table 10-1	Clock Trim Module (CTRM) Register Offset.....	132
Table 11-1	Comparison Table of Pin Function and Timer Function.....	143
Table 11-3	GTIM TRS Trigger Source List.....	156
Table 11-5	BTIM Register Offset.....	158
Table 11-6	GTIM Register Offset.....	158
Table 12-7	CH0A Capture Input Signal.....	199
Table 12-12	ATIM3 Interrupt List.....	204
Table 12-13	Timer Interconnection Signal List.....	205
Table 12-14	Mode0 Timer Register Offset.....	209
Table 12-15	Mode1 Pulse Width Measurement (PWM) Register Offset.....	209
Table 12-16	Mode2/3 Compare-Capture Mode Register Offset.....	209
Table 13-1	Basic Features of RTC.....	245
Table 13-2	Real Time Counter (RTC) Register Offset.....	254
Table 14-1	IWDT Timeout Time (10kHz Independent Clock).....	282
Table 14-2	Independent Watchdog (IWDT) Register Offset.....	283
Table 15-1	LPUART Mode0/1/2/3 Data Structure.....	291
Table 15-2	B8 Data Meaning.....	291
Table 15-3	SCLK = 4MHz Baud Rate Calculation Table.....	296
Table 15-4	SCLK = 8MHz Baud Rate Calculation Table.....	297
Table 15-5	SCLK = 16MHz Baud Rate Calculation Table.....	297
Table 15-6	SCLK = 24MHz Baud Rate Calculation Table.....	297
Table 15-7	SCLK = 32MHz Baud Rate Calculation Table.....	298
Table 15-8	SCLK = 48MHz Baud Rate Calculation Table.....	298
Table 15-9	Low Power Universal Asynchronous Receiver/Transmitter (LPUART) Register Offset.....	303
Table 16-1	SPI Interrupt.....	324
Table 16-2	Serial Peripheral Interface (SPI) Register Offset.....	327
Table 17-1	Host Timing Parameters.....	340
Table 17-2	Synchronization Delay.....	340
Table 17-3	I2C Timing Parameter Limitations.....	340
Table 17-4	I2C Example Timing Configuration.....	341
Table 17-7	High Speed Integrated Circuit Bus (HSI2C) Register Offset.....	348
Table 18-1	ADC Interrupt.....	385
Table 18-2	Analog-to-Digital Converter (ADC) Register Offset.....	387
Table 19-1	Analog Voltage Comparator (VC) Register Offset.....	400
Table 20-1	Low Voltage Detector (LVD) Register Offset.....	411
Table 21-1	True Random Number Generator (TRNG) Register Offset.....	418
Table 22-3	Host Transmitted Data Packet Request.....	424

Table 22-4	Target Transmitted Acknowledgment Response.....	425
Table 22-5	Data Transfers Transmitted by Host or Target.....	425

1 Product Introduction

HC32L021 Series is a basic 32-bit ultra-low power MCU with a wide operating voltage range. The chip integrates a 12-bit 1Msps SARADC, high-performance PWM timer, LPUART, SPI, HSI2C, and other peripherals, featuring ultra-low power consumption, high reliability, and high integration. This series adopts an ARM Cortex-M0+ core with a maximum frequency of 48MHz, compatible with mature mainstream development and debugging software such as Keil, IAR, and Eclipse, and supports development using C language and assembly language.

Typical Application

- Motor control, battery management
- Smart home, medical devices
- Security alarms, intelligent transportation
- Sensor modules, wireless modules, electronic shelf labels

About This Manual

This manual describes the chip's functionalities, operational guidelines, and usage methods. For detailed chip specifications, refer to the corresponding "Datasheet".

2 Storage and Architectures

2.1 Overview

System consists of the following parts:

- 1 AHB bus Master:
 - ▶ Cortex-M0+
- Multiple AHB bus Slaves:
 - ▶ Flash Memory
 - ▶ SRAM Memory
 - ▶ AHB to APB Bridge 0 includes all APB0 interface peripherals
 - ▶ AHB to APB Bridge 1 includes all APB1 interface peripherals
 - ▶ AHB peripherals

The entire system bus structure is realized by multi-level AHB-lite bus interconnection.

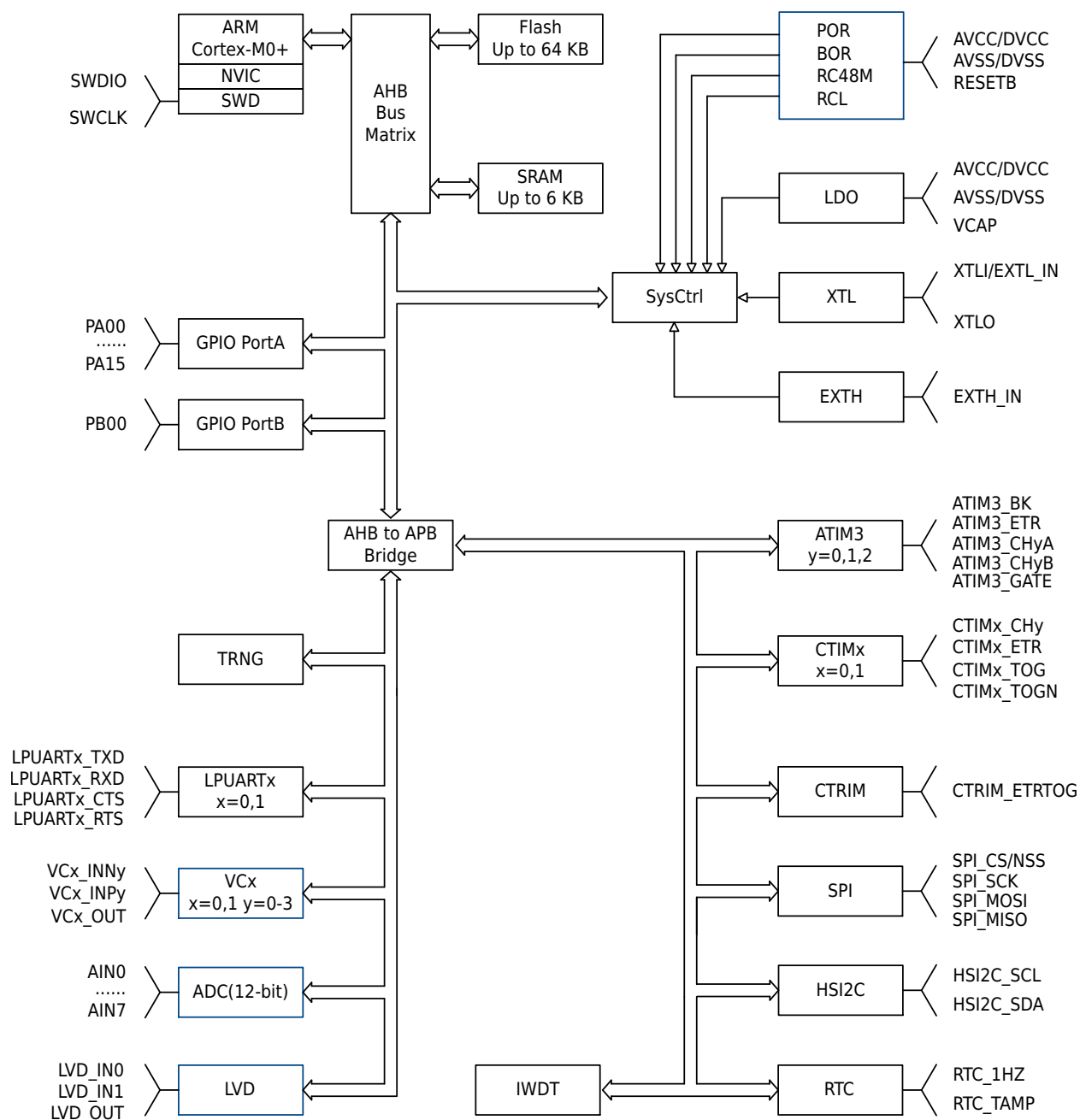
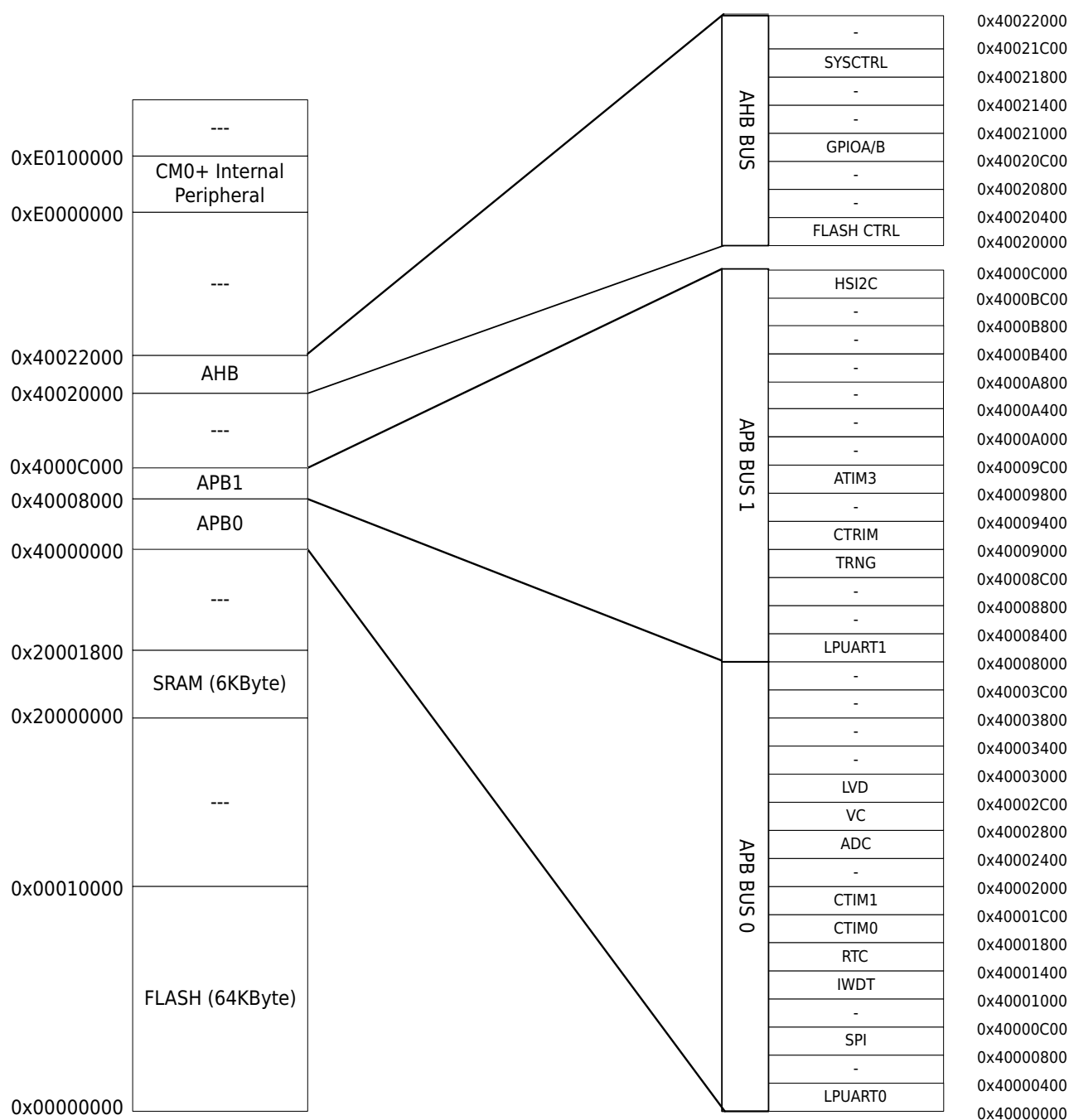


Figure 2-1 System Architecture Schematic Diagram

2.2 System Address Division

System of address region division:



2.3 Memory and Module Address Assignment

Table 2-1 Address Division Table

Categories	Boundary Address	Size	Memory Area
Memory	0x00000000-0x0000FFFF	64KByte	FLASH
	0x20000000-0x200017FF	6KByte	SRAM
APB0 peripheral	0x40000000-0x400003FF	1KByte	LPUART0
	0x40000800-0x40000BFF	1KByte	SPI
	0x40001000-0x400013FF	1KByte	IWDG
	0x40001400-0x400017FF	1KByte	RTC
	0x40001800-0x40001BFF	1KByte	CTIM0
	0x40001C00-0x40001FFF	1KByte	CTIM1
	0x40002400-0x400027FF	1KByte	ADC
	0x40002800-0x40002BFF	1KByte	VC
	0x40002C00-0x40002FFF	1KByte	LVD
APB1 peripherals	0x40008000-0x400083FF	1KByte	LPUART1
	0x40008C00-0x40008FFF	1KByte	TRNG
	0x40009000-0x400093FF	1KByte	CTIM
	0x40009800-0x40009BFF	1KByte	ATIM3
	0x4000BC00-0x4000BFFF	1KByte	HSI2C
AHB peripherals	0x40020000-0x400203FF	1KByte	FLASH CTRL
	0x40020C00-0x40020FFF	1KByte	GPIOA/B
	0x40021800-0x40021BFF	1KByte	SYSCTRL

3 Operating Mode

3.1 Overview

This product can realize 3 working modes under the control of the power management module. The operable modules and their power consumption differ across working modes.

This product has the following working modes:

1. Active Mode: CPU running, peripheral functional modules running.
2. Sleep mode: CPU halted, peripheral functional modules running.
3. DeepSleep mode: CPU halted, high-speed clock modules disabled.

3.2 Working Mode Switching

The product can be freely switched among 3 working modes when it is working. In Active mode, executing specific instructions can transition the device into Sleep mode or DeepSleep mode. Wake-up from Sleep or DeepSleep mode via an interrupt and return to Active mode.

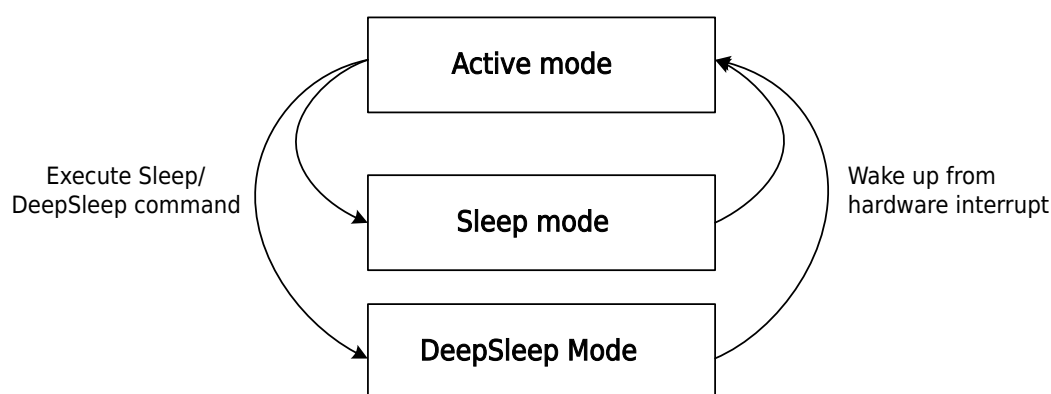


Figure 3-1 Working Mode State Transition Diagram

The interrupts that the CPU can respond in various working modes, please refer to [Interrupt Source](#) in "Interrupt Controller (NVIC)" Chapter.

In different working mode, the reset types system can respond following to:

Table 3-1 Reset Type Response Capabilities in Different Operational Mode

Reset Vector Index Reference	Reset Source	Active Mode	Sleep Mode	DeepSleep Mode
[0]	Power-on Reset (POR)/Power-down Reset (BOR)	✓	✓	✓
[1]	Hardware Reset RESETB Pin	✓	✓	✓
[2]	Hardware Reset LVD	✓	✓	✓
[3]	Hardware Reset IWDG	✓	✓	✓
[4]	Hardware Reset Cortex-M0+ LOCKUP	✓	×	×
[5]	Software Reset Cortex-M0+ SYS-RESETREQ	✓	×	×

3.3 Active Mode

The MCU is in the Active mode after system is reset at power-on, or after waking up from various low power consumptions. Various low-power modes are available to conserve power when the CPU is not required to continue running, such as while waiting for an external event. Users need to select an optimal low-power mode based on the lowest power consumption, fastest startup time, and available wake-up sources.

In Active mode, several ways to reduce chip power consumption in run mode:

1. In Active mode, the speed of system clock (HCLK) can be reduced by programming the prescaler registers (SYSCTRL0.HCLK_PRS).
2. In Active mode, disable the clocks of unused peripherals (PERI_CLKENx) to reduce power consumption.
3. Turn off the clocks of unused peripheral clock (PERI_CLKENx) before executing WFI instructions.
4. In certain scenarios, low-power mode can be used instead of Active mode, as this product features extremely short wake-up time that can still meet system real-time response requirements.

3.4 Sleep Mode

Executing the WFI instruction enters Sleep mode. In Sleep mode, the CPU halted, but the clock module, system clock, NVIC interrupt processing and peripheral functional modules can still work.

When the system enters the sleep state, the port state will not be changed. Before entering the sleep state, the IO state can be changed as needed.

Enter Sleep Mode

Enter sleep state by executing WFI instruction. There are two options available for selecting the Sleep mode entry mechanism based on value of the SLEEPONEXIT bit in the system control register of Cortex-M0+:

- SLEEP-NOW: If the SLEEPONEXIT bit is cleared, the MCU enters Sleep mode immediately when WFE is executed.
- SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the MCU enters Sleep mode immediately when the system exits from the lowest priority interrupt handler.

Exit Sleep Mode

If the WFI instruction is executed to enter the Sleep mode, any peripheral interrupt responded by the the high-priority nested vector interrupt controller can wake up the system from the Sleep mode.

Precautions

- The position of SLEEPONEXIT is 1, and the interrupt will automatically enter Sleep after execution, and the program does not need to write `__wfi()`;
- SLEEPONEXIT bit is cleared to 0, `main()` executes `__wfi()` and enters Sleep, interrupt triggers and executes the interrupt program and returns to `main()`, then executes WFI instruction and enters Sleep. Wait for a subsequent interrupt to fire.
- The SLEEPONEXIT bit does not affect the execution of the `__wfi()` instruction. SLEEPONEXIT = 0: `main()` executes `__wfi()` and enters Sleep, interrupt triggers and executes the interrupt program and returns to `main()`, and then continues to execute;
- If Sleep is entered in an interrupt, only interrupts with a priority higher than this interrupt can wake up, and the high priority is executed first, and then the low priority is executed; interrupts with a priority lower than or equal to this interrupt cannot wake up.

- For details on entering and waking up from Sleep mode via WFE (Wait For Event) instruction, refer to the Cortex-M0+ relevant technical manual.

3.5 DeepSleep Mode

Using SLEEPDEEP with the WFI instruction enables entry into DeepSleep mode. In DeepSleep mode, the CPU halted, the high-speed clock is disabled, the low-speed clock can be configured to remain active or not, certain low-power peripheral modules may be selectively enabled, while the NVIC interrupt handling remains functional.

- When the system (Using high-speed clock) enters DeepSleep mode, the high-speed clock is automatically disabled, and the low-speed clock remains in the state before entering DeepSleep.
- The system (Using low-speed clock) enters the DeepSleep mode, the low-speed clock is not automatically disabled and maintains its operation. Only ARM Cortex-M0+ is not running, other modules are running.
- When the system clock is switched, all clocks will not be automatically disabled, and the corresponding clocks need to be disabled or enabled by software according to power consumption and system requirements.
- When the system enters the DeepSleep mode, it will not change the port state. Before entering the DeepSleep mode, change the IO state as needed.

Enter DeepSleep Mode

First set the SLEEPDEEP bit of Cortex-M0+ in the system control register, and enter the DeepSleep mode by executing the WFI instruction. Depending on the value of the SLEEPONEXIT bit in the system control register, there are two options for selecting the DeepSleep mode entry mechanism:

- SLEEP-NOW: If the SLEEPONEXIT bit is cleared, the MCU enters DeepSleep mode immediately when WFI is executed.
- SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the MCU enters DeepSleep mode immediately when the system exits from the lowest priority interrupt handler.

Exit DeepSleep Mode

If the WFI instruction is executed to enter DeepSleep mode, any peripheral interrupt that is responded to by the high priority nested vector interrupt controller (operable under DeepSleep peripheral module interrupt) can wake up the system from DeepSleep mode.

For wake-up configuration, refer to the "Interrupt Wakeup Control" section in the "System Controller (SYSCRTL)" chapter.

Precautions

- The position of SLEEPONEXIT is 1, and the interrupt will automatically enter DeepSleep after execution, and the program does not need to write `__wfi()`;
- SLEEPONEXIT bit is cleared to 0, `main()` executes `__wfi()` and enters DeepSleep, interrupt triggers and executes the interrupt program and returns to `main()`, then executes WFI instruction and enters DeepSleep. Wait for a subsequent interrupt to fire.
- The SLEEPONEXIT bit does not affect the execution of the `__wfi()` instruction. SLEEPONEXIT = 0: `main()` executes `__wfi()` and enters DeepSleep, interrupt triggers and executes the interrupt program and returns to `main()`, and then continues to execute.
- If DeepSleep is entered in an interrupt, only interrupts with a priority higher than this interrupt can wake up, and the high priority is executed first, and then the low priority is executed; interrupts with a priority lower than or equal to this interrupt cannot wake up.

- For details on entering and waking up from DeepSleep mode via WFE (Wait For Event) instruction, refer to the Cortex-M0+ relevant technical manual.

3.6 Functional/Module Working Status

The comparison of the working status of the functional/modules in the three working modes is shown in the table below:

Table 3-2 Peripheral Working Status Table in Each Mode

Functional/Module	Active Mode	Sleep mode	DeepSleep mode
Cortex-M0+	✓	×	×
XTL	✓	✓	✓
RC48M	✓	✓	×
RCL	✓	✓	✓
RESET	✓	✓	✓
GPIO	✓	✓	✓
HSI2C	✓	✓	✓ ⁽³⁾
SPI	✓	✓	×
CTIM	✓	✓	✓ ⁽²⁾
FLASH	✓	✓	×
RAM	✓	✓	×
CTIM	✓	✓	×
ATIM3	✓	✓	×
RTC	✓	✓	✓
IWDT	✓	✓	✓
LPUART	✓	✓	✓ ⁽²⁾
TRNG	✓	✓	×
ADC	✓	✓	×
VC	✓	✓	✓ ⁽¹⁾
LVD	✓	✓	✓
SWD	✓	✓	✓



Notice

1. If you use the VC wake-up function in DeepSleep mode, the VC filter clock cannot select PCLK.
2. When the working clock is PCLK, wakeup from DeepSleep mode is not supported.
3. Only slave mode supports wakeup from DeepSleep.
4. Before executing the WFI command to enter DeepSleep mode, all interrupts not used for MCU wakeup must be disabled.

3.7 Register Description

3.7.1 System Control Register (SCB_SCR) (Cortex-M0+ Core System Control Register)

Base address: 0xE000ED10

Offset	Bit Position																																					
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved																												SEVONPEND		RESERVED		SLEEPDEEP		SLEEPONEXIT		RESERVED	
Access																													RW				RW		RW			

Bit	Name	Functional Description	Access
31:5	Reserved	-	-
4	SEVONPEND	Event is Sent when Interrupt Pending 0b1: If WFE is used to enter Sleep or DeepSleep mode, any newly pending event can wake up the processor	RW
3	Reserved	-	-
2	SLEEPDEEP	Sleep Mode Selection 0b0: Sleep mode 0b1: DeepSleep mode	RW
1	SLEEPONEXIT	Sleep-on-Exit Feature 0b0: Disable sleep-on-exit feature 0b1: Enable Sleep-on-Exit feature. When exiting exception handling and returning to thread mode, the processor automatically enters Sleep or DeepSleep mode (WFI)	RW
0	Reserved	-	-

4 FLASH

4.1 Introduction

This system includes a Flash memory with a capacity of 64KB, which is divided into 128 sectors, and the capacity of each sector is 512 bytes.

This controller supports erasing, programming and reading operations on Flash memory, also supports erasing and writing protection of Flash memory, and writing protection of control registers.

This controller supports data read and write operations of three-bit widths of byte (8 bits), half-word (16 bits) and word (32 bits) of Flash. Note: The target address of the half-word operation must be half-word aligned (the lowest bit of the address is 0b0), and the address of the word operation must be word-aligned (the lowest two bits of the address are 0b00). If the target address is not aligned according to the bit width, the operation is invalid and the CPU will enter the Hard Fault interrupt.

4.2 Functional Description

4.2.1 Capacity Division

The correspondence between the capacity division addresses and sequence numbers is as follows:

Table 4-1 Flash Capacity Division Table

Address	Serial Number	...	Address	Serial Number
0x0E00-0x0FFF	Sector7	...	0x0FE000x0FFFF	Sector127
0x0C00-0x0DFF	Sector6	...	0x0FC000x0FDFF	Sector126
0x0A00-0x0BFF	Sector5	...	0x0FA000x0FBFF	Sector125
0x0800-0x09FF	Sector4	...	0x0F8000x0F9FF	Sector124
0x0600-0x07FF	Sector3	...	0x0F6000x0F7FF	Sector123
0x0400-0x05FF	Sector2	...	0x0F4000x0F5FF	Sector122
0x0200-0x03FF	Sector1	...	0x0F2000x0F3FF	Sector121
0x0000-0x01FF	Sector0	...	0x0F0000x0F1FF	Sector120

4.2.2 Read Wait Period

The fastest instruction fetch frequency supported by the built-in Flash of this device is 24MHz. When the HCLK frequency exceeds 24MHz and is less than or equal to 48MHz, waiting period must be inserted for the Flash read time, that is, set FLASH_CR.WAIT to 1. When the wait cycle is inserted, Flash will complete a read operation every two periods.

4.2.3 Flash Security Protection

Full Chip Erase/Write Protection

By setting FLASH_CR.RO to 1, the entire Flash can be configured as read-only, meaning it cannot be programmed or erased.

Sector Erase/Write Protection

The entire 64KB Flash memory is divided into 128 sectors, and every 4 sectors share an erase/write protection bit. When the page is protected, the erasing and writing operations on the sector are invalid and an alarm flag and interrupt signal are generated. When any sector in the Flash memory is protected, the whole-chip erasing of the Flash is invalid, and an alarm flag and an interrupt signal are generated.

PC Address Erase and Write Protection

The CPU runs the program in Flash, if the current PC pointer just falls within the address range of the page to be erased, then the erase operation is invalid and an alarm flag and an interrupt signal are generated.

Register Write Protection

The important controller of this module shields ordinary write operations, and must be modified by writing sequence.

- **Registers Can Be Changed After a Write Sequence Are As Follows:**

FLASH_CR, FLASH_SLOCK0

- **Registers Can Be Changed Without a Write Sequence Are As Follows:**

FLASH_ICLR, FLASH_BYPASS

The specific operation steps to modify the register value by writing sequence are as follows:

Step1. Write 0x5A5A to the FLASH_BYPASS register.

Step2. Write 0xA5A5 to the FLASH_BYPASS register.

Step3. Write the target value to the register to be modified.

Step4. Verify whether the current value of the register to be modified is the same as the target, if not, jump to Step1.

Step5. Perform other operations.



Notice

Write 0x5A5A, 0xA5A5, and write the target register. Do not insert any write operations (write Flash, RAM, REG, etc.) between these three steps of writing operations, otherwise the value of the target register cannot be rewritten. If rewriting fails, you need to perform these three steps again.

Data Readout Protection

The chip's Flash supports 4 -level read protection, and the current protection level can be read through the FLASH_LOCK_ST register. When programming the chip with a programmer, the read protection level of Flash can be configured.

The functions of the chip under each protection level are shown in the table below.

Protection Level	Chip Function
Level0	Flash can be read and written through ISP and SWD interfaces, if the level is downgraded to this level through other levels, the Flash content will be forcibly erased
Level1	Data cannot be read Date can be downgraded to Level 0 via ISP and SWD interfaces.

Protection Level	Chip Function
Level2	Data cannot be read Data can be downgraded The SWD interfaces is disabled
Level3	Data cannot be read The functions of the ISP and SWD interfaces being disabled, that is, the chip can only be programmed once

4.3 Operation Example



Notice

- Before performing erase or write operations on FLASH, it is necessary to set FLASH_CR.RO to 0 and configure FLASH as readable, programmable and erasable.
- After the FLASH erase or write operation is completed, it is recommended to set FLASH_CR.RO to 1 and configure FLASH as read-only, non-programmable or non-erasable.

4.3.1 Sector Erase

Context

Sector erase allows the user to erase one user-specified sector at a time. After the erase operation is completed, the data in the sector are all 0xFF. If the erase operation is performed from Flash, the CPU will stop fetching instructions, and the hardware will automatically wait for the operation to complete (FLASH_CR.BUSY becomes 0); if the erase operation is performed from RAM, the CPU will not stop fetching instructions, user software should wait for the operation to complete (FLASH_CR.BUSY becomes 0).

Procedure

- Step1.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step2.** Configure FLASH_CR.OP to 2, set the Flash operation mode to Sector Erase.
- Step3.** Check if FLASH_CR.OP is 2, if not, jump to Step1.
- Step4.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step5.** Set the corresponding bit of FLASH_SLOCK to 1 to remove the sector 's erasure protection.
- Step6.** Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, jump to Step4.
- Step7.** Write arbitrary data to any address in the Sector to be erased, triggering Sector erasure.

```
*((unsigned char *)0x00000200) = 0x00
```

- Step8.** Wait for FLASH_CR.BUSY to become 0, and the Sector erase operation is completed.
- Step9.** To erase other Sectors, repeat Step4-Step8.

4.3.2 Chip Erase

Context

Chip erase can erase all sector at one time. After the erase operation is completed, the data in all sector are 0xFF. If the erase operation is performed from the Flash, the operation will be prohibited. Because this operation will erase the program segment where the current PC is located. If this happens, the FLASH_IFR.RC will be set; if the erase operation is performed from RAM, the CPU

will not stop fetching instructions, and the user software should wait for the operation to complete (FLASH_CR.BUSY becomes 0).

Procedure

- Step1.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step2.** Configure FLASH_CR.OP to 3, set the Flash operation mode to Chip erase.
- Step3.** Check if FLASH_CR.OP is 3, if not, jump to Step1.
- Step4.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step5.** Set FLASH_SLOCK0 to 0xFFFFFFFF, remove the erase protection of Sector0-127.
- Step6.** Check whether FLASH_SLOCK0 is 0xFFFFFFFF, if not 0xFFFFFFFF, jump to Step4.
- Step7.** Write any address in the Chip to be erased to trigger Chip erasure.

```
*((unsigned char *)0x00000000) = 0x00
```

- Step8.** Wait for FLASH_CR.BUSY to become 0, and the Chip erase operation is completed.

4.3.3 Program

Context

The program can only change the bit data in Flash from 1 to 0, so before writing data, make sure that the data in the address to be written is 0xFF. Support 3 write lengths: Byte (8bits), Half-word (16bits), Word (32bits), the written data is stored in Flash in little-endian mode, that is, the low byte of the data is stored in the low address. If the program is executed from Flash, the CPU will stop fetching instructions, and the hardware will automatically wait for the operation to complete (for FLASH_CR.BUSY to becomes 0); if the program is executed from RAM, the CPU will not stop fetching instructions, and the user Software should wait for the operation to complete (for FLASH_CR.BUSY to goes to 0).

Procedure

● Byte Program

- Step1.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step2.** Configure FLASH_CR.OP to 1, set the Flash operation mode to write.
- Step3.** Check if FLASH_CR.OP is 1, if not, jump to Step1.
- Step4.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step5.** Set the corresponding bit of FLASH_SLOCK to 1 to remove the erase protection.
- Step6.** Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, jump to Step4.
- Step7.** Perform Byte program on the target address to be written to trigger the program.

```
*((unsigned char *)0x00001231) = 0x5A
```

- Step8.** Wait for for FLASH_CR.BUSY to to become 0, and the program is completed.
- Step9.** To write Byte to other addresses, repeat Step7-Step8.

● Half-word Program

- Step1.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step2.** Configure FLASH_CR.OP to 1, set the Flash operation mode to write.

- Step3.** Check if FLASH_CR.OP is 1, if not, jump to Step1.
- Step4.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step5.** Set the corresponding bit of FLASH_SLOCK to 1 to remove the erase protection.
- Step6.** Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, jump to Step4.
- Step7.** Perform a Half-word program on the target address to be written to trigger the program.

```
*((unsigned short int *)0x00001232) = 0xABCD
```

- Step8.** Wait for for FLASH_CR.BUSY to to become 0, and the program is completed.
- Step9.** To write Half-word to other addresses, repeat Step7-Step8.

● Word Program

- Step1.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step2.** Configure FLASH_CR.OP to 1, set the Flash operation mode to write.
- Step3.** Check if FLASH_CR.OP is 1, if not, jump to Step1.
- Step4.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step5.** Set the corresponding bit of FLASH_SLOCK to 1 to remove the erase protection.
- Step6.** Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, jump to Step4.
- Step7.** Perform a Word program on the target address to be written to trigger the program.

```
*((unsigned int *)0x00001234) = 0x55667788
```

- Step8.** Wait for for FLASH_CR.BUSY to to become 0, and the program is completed.
- Step9.** To write Word to other addressed, repeat Step7-Step8.

4.3.4 Cont-Program

Context

The program can only change the bit data in Flash from 1 to 0, so before writing data, make sure that the data in the address to be written is 0xFF. Support 3 write lengths: Byte (8bits), Half-word (16bits), Word (32bits), The written data is stored in Flash in little-endian mode, that is, the low byte of the data is stored in the low address. Cont-program can only be performed from RAM, and "read " on Flash are prohibited. User software should wait for the operation to complete (for FLASH_CR.BUSY to becomes 0).

It is recommended that HCLK not be lower than 2 MHz during continuous programming.

Procedure

● Byte Program

- Step1.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step2.** Configure FLASH_CR.OP value at 0.
- Step3.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step4.** Configure FLASH_CR.CONTP to 1, set the Flash operation mode to continuous write.
- Step5.** Check whether FLASH_CR.OP and FLASH_CR.CONTP are 0 and 1 respectively, if not, jump to Step1.

- Step6.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step7.** Set the corresponding bit of FLASH_SLOCK to 1 to remove the erase protection.
- Step8.** Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, jump to Step6.
- Step9.** Perform Byte program on the target address to be written to trigger the program.

```
*((unsigned char *)0x00001232) = 0xAB
```

- Step10.** Wait for for FLASH_CR.BUSY to to become 0, and the program is completed.
- Step11.** To write Byte to other addresses, repeat Step9-Step10.
- Step12.** If all the writing operation is completed, proceed to the next step.
- Step13.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step14.** Configure FLASH_CR.OP at 1.
- Step15.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step16.** Configure FLASH_CR.CONTP to 0.
- Step17.** Check whether FLASH_CR.OP and FLASH_CR.CONTP are 1 and 0 respectively, if not, jump to Step13.
- Step18.** Wait FLASH_CR.BUSY to become 0.
- Step19.** Proceed with the following, FLASH_CR.OP can be changed to the value of other modes complete other functional operations.

● Half-word Program

- Step1.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step2.** Configure FLASH_CR.OP value at 0.
- Step3.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step4.** Configure FLASH_CR.CONTP to 1, set the Flash operation mode to continuous write.
- Step5.** Check whether FLASH_CR.OP and FLASH_CR.CONTP are 0 and 1 respectively, if not, jump to Step1.
- Step6.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step7.** Set the corresponding bit of FLASH_SLOCK to 1 to remove the erase protection.
- Step8.** Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, jump to Step6.
- Step9.** Perform a Half-word program on the target address to be written to trigger the program.

```
*((unsigned short int *)0x00001232) = 0xABCD
```

- Step10.** Wait for for FLASH_CR.BUSY to to become 0, and the program is completed.
- Step11.** To write Half-word to other addresses, repeat Step9-Step10.
- Step12.** If all the writing operation is completed, proceed to the next step.
- Step13.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step14.** Configure FLASH_CR.OP at 1.

- Step15.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step16.** Configure FLASH_CR.CONTP to 0.
- Step17.** Check whether FLASH_CR.OP and FLASH_CR.CONTP are 1 and 0 respectively, if not, jump to Step13.
- Step18.** Wait FLASH_CR.BUSY to become 0.
- Step19.** Proceed with the following, FLASH_CR.OP can be changed to the value of other modes complete other functional operations.

- **Word Program**

- Step1.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step2.** Configure FLASH_CR.OP value at 0.
- Step3.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step4.** Configure FLASH_CR.CONTP to 1, set the Flash operation mode to continuous write.
- Step5.** Check whether FLASH_CR.OP and FLASH_CR.CONTP are 0 and 1 respectively, if not, jump to Step1.
- Step6.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step7.** Set the corresponding bit of FLASH_SLOCK to 1 to remove the erase protection.
- Step8.** Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, jump to Step6.
- Step9.** Perform a Word program on the target address to be written to trigger the program.

```
*((unsigned int *)0x00001234) = 0xABCDABCD
```

- Step10.** Wait for FLASH_CR.BUSY to become 0, and the write operation is completed.
- Step11.** To write Word to other addressed, repeat Step9-Step10.
- Step12.** If all the writing operation is completed, proceed to the next step.
- Step13.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step14.** Configure FLASH_CR.OP to 1, set the Flash operation mode to write.
- Step15.** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step16.** Configure FLASH_CR.CONTP to 0.
- Step17.** Check whether FLASH_CR.OP and FLASH_CR.CONTP are 1 and 0 respectively, if not, jump to Step13.
- Step18.** Wait FLASH_CR.BUSY to become 0.
- Step19.** Proceed with the following, FLASH_CR.OP can be changed to the value of other modes complete other functional operations.

4.3.5 Read

Context

Support to 3 kinds read: Byte(8bits), Half-word(16bits), Word(32bits), the read data is little-endian mode, that is, the low byte of the data is stored in the low address. Read need no prior steps, the data in the FLASH can be read at any time.

Procedure

- **Byte Read**

```
temp = *((unsigned char *)0x00001231)
```

- **Half-word Read**

```
temp = *((unsigned short int *)0x00001232)
```

- **Word Read**

```
temp = *((unsigned int *)0x00001234)
```

4.4 Register Description

4.4.1 Register List

Base address: 0x40020000

Table 4-3 Flash Controller (FLASH) Register Offset

Offset	Register	Description
0x20	FLASH_CR	Control Register
0x24	FLASH_IFR	Interrupt Flag Register
0x28	FLASH_ICLR	Interrupt Flag Clear Register
0x2C	FLASH_BYPASS	0x5A5A-0xA5A5 Bypass Sequence Register
0x30	FLASH_SLOCK0	Sector0-127 Erase and Write Protection Register
0x6C	FLASH_LOCK_ST	Protection Status Register

4.4.2 CR Register (FLASH_CR)

Offset	Bit Position															
0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0b00000000 00000000 0000X010 00000000															
Name	Reserved										RO	CONTP	DPSTB_EN	Reserved	IE	BUSY
Access											RW	RW	RW		RW	RO
															RW	OP

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11	RO	Read-only enable bit 0b0: Readable, programmable, erasable 0b1: Read only, not programmable or erasable	RW
10	CONTP	Continuous programming 0b0: Single programming 0b1: Continuous programming	RW
9	DPSTB_EN	FLASH dpstb enable mask bit 0b0: When the system enters DeepSleep mode, FLASH does not enter low power consumption mode 0b1: When the system enters DeepSleep mode, FLASH enters low power consumption mode	RW
8:7	Reserved	-	-
6:5	IE	FLASH interrupt enable IE[6]: FLASH erase/write protected address interrupt enable IE[5]: FLASH erase PC Value interrupt enable 0b0: Disable 0b1: Enable	RW
4	BUSY	Idle/Busy flag bit 0b0: Idle State 0b1: Busy State	RO
3:2	WAIT	Read FLASH period 0b00: 1 period, 0-24MHz 0b01: 2 periods, 24-48MHz 0b10: Setting prohibited 0b11: Setting prohibited	RW

Bit	Name	Functional Description	Access
1:0	OP	FLASH operation ● 0b00: Read ● 0b01: Program ● 0b10: Sector erase ● 0b11: Chip erase  Note ● During Flash erase/write operations, modifying the RC48M clock configuration of RC48M_CR.FSEL is prohibited, as it may cause operation failure. ● Before entering DeepSleep mode, the OP field must be configured as 0b00; otherwise, the system cannot transition to DeepSleep.	RW

4.4.3 IFR Register (FLASH_IFR)

Offset	Bit Position																															
0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000010																															
Name	Reserved																														PROT	
Access																															RO	PC

Bit	Name	Functional Description	Access
31:2	Reserved	-	-
1	PROT	FLASH erase/write protection alarm interrupt flag bit	RO
0	PC	FLASH erase/write PC address alarm interrupt flag bit	RO

4.4.4 ICLR Register (FLASH_ICLR)

Offset	Bit Position																																
0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x0000000F																																
Name	Reserved																														PROT		PC
Access																															WO		

Bit	Name	Functional Description	Access
31:2	Reserved	-	-
1	PROT	Clear protection alarm interrupt flag bit Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
0	PC	Clear the PC address alarm interrupt flag bit Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

4.4.5 BYPASS Register (FLASH_BYPASS)

Offset	Bit Position																															
0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																BYSEQ															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	BYSEQ	FLASH module protection control register Write 0x5A5A first to FLASH_BYPASS, and then write 0xA5A5 to start the program to FLASH module registers. As long as the program is done to module register, this protection bit automatically recovers to the protection state. Subsequent program to the FLASH module registers require repeating the previous procedures.	RW

4.4.6 SLOCK0 Register (FLASH_SLOCK0)

Offset	Bit Position																															
0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0X00000000 (Peripheral reset is invalid)																															
Name	SLOCK																															
Access	RW																															

Bit	Name	Functional Description	Access														
31:0	SLOCK	Sector Erase/Write Protection Bit ● 0b0: Disable erase/write ● 0b1: Enable erase/write	RW														
		<table><tr><th>Bit</th><th>Serial Number</th></tr><tr><td>0</td><td>Sector0-3</td></tr><tr><td>1</td><td>Sector4-7</td></tr><tr><td>2</td><td>Sector8-11</td></tr><tr><td>3</td><td>Sector12-15</td></tr><tr><td>...</td><td>...</td></tr><tr><td>31</td><td>Sector124-127</td></tr></table>		Bit	Serial Number	0	Sector0-3	1	Sector4-7	2	Sector8-11	3	Sector12-15	...	...	31	Sector124-127
		Bit		Serial Number													
		0		Sector0-3													
		1		Sector4-7													
		2		Sector8-11													
		3		Sector12-15													
		...		...													
31	Sector124-127																

4.4.7 Protection Status Register (FLASH_LOCK_ST)

Offset	Bit Position																															
0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															LOCK_ST
Access																																RO

Bit	Name	Functional Description	Access
31:2	Reserved	-	-
1:0	LOCK_ST	FLASH Protection Status ● 0b00: Level0, ISP can read and write, SWD can read and write ● 0b01: Level1, ISP can be downgraded, SWD can be downgraded; data cannot be read out ● 0b10: Level2, ISP can be downgraded, SWD has no function; data cannot be read out ● 0b11: Level3, ISP has no function, SWD has no function; data cannot be read out	RO

5 RAM

5.1 Introduction

This system includes an SRAM block with a capacity of 6KB, supporting read and write operations in byte (8 bits), halfword (16 bits), and word (32 bits) formats. Read and write operations can be performed at the system clock frequency without waiting for cycles.

5.2 Functional Description

5.2.1 RAM Address Range

RAM in the system map is shown in the table below:

Table 5-1 RAM Address Mapping

Address Range	Size	Memory Type
0x20000000-0x200017FF	6KByte	SRAM

5.2.2 Read and Write Bit Width

This controller supports byte (8 bits), halfword (16 bits), word (32 bits) read and write operations with three-bit widths.

- Addresses for byte operations must be byte aligned
- The target address for halfword operations must be halfword-aligned (the least significant bit of the address must be 0b0).
- The address for word operations must be word-aligned (the least significant two bits of the address must be 0b00).

If the target address of the read and write operation is not aligned according to the bit width, the operation is invalid, and the system will generate a Hard Fault error interrupt.

6 SYSCTRL

6.1 System Clock Introduction

The clock control module primarily provides system clock, peripheral clock, and peripheral reset control functions.

This product provides the following different clock sources as the system clock:

- RC48M - Internal high-speed RC clock
- RCL - Internal low-speed RC clock
- EXTH - External high-speed clock source is used as input
- XTL - External low-speed crystal oscillator: supports external low-speed crystal/ceramic resonator or external low-speed clock source as input (high-precision 32.768kHz).

The clock input pin assignment is detailed in the table below:

Clock pin	Corresponding pin
XTLI/EXTL_IN	PA07
XTLO	PA06
EXTH_IN	PA11



Note

1. When switching the system clock source, please follow the procedure strictly. See the [System Clock Switching](#) section for details.
2. When XTL using an active clock, the clock signal is input directly from the EXTL_IN pin. External high-speed clock signal is input directly from the EXTH_IN pin.

This product also contains the following two auxiliary clocks:

- RC10K - Internal 10kHz clock, for IWDG modules
- RC256K - Internal 256kHz clock, for LVD and VC modules.

6.1.1 Clock Architecture Diagram

The clock architecture of this product is shown as follows:

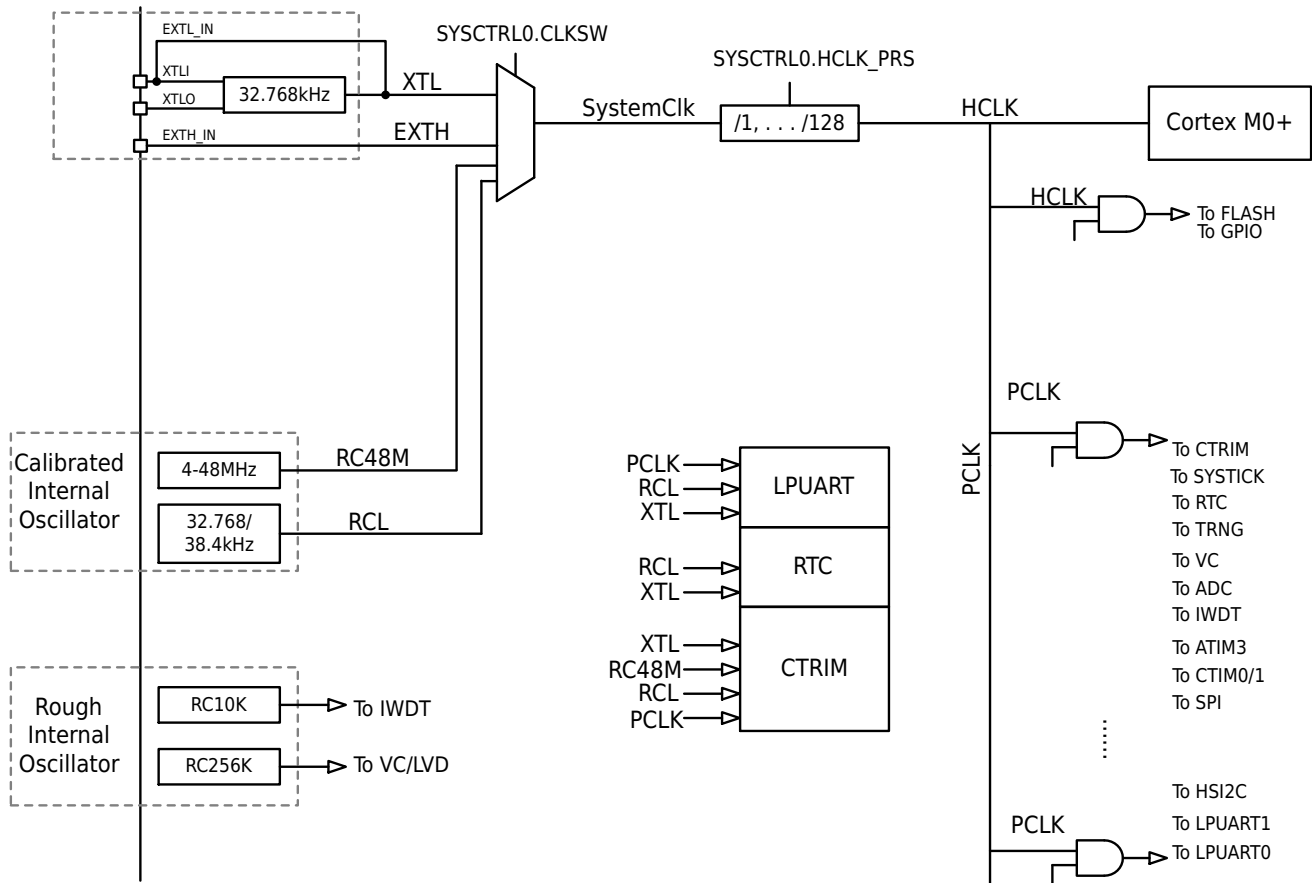


Figure 6-1 Clock Architecture Diagram

6.1.2 Clock Types

6.1.2.1 RC48M

RC48M is an internal high-speed clock oscillator with configurable and calibratable frequency, and it is also the default clock source used after chip power-up or partial reset with a default frequency of 4MHz. The clock frequency can be configured through the RC48M_CR register. Four preset calibrated frequency configuration values—4MHz, 6MHz, 32MHz and 48MHz before leaving the factory can be directly loaded for use; it can also adjust the calibration value in this register to change to other frequencies or achieve higher-precision calibration according to actual needs.

When the system enters DeepSleep mode, this high-speed clock will be automatically turned off.

For instructions on changing the RC48M clock source frequency, please refer to the chapter [System Clock Switching](#).

The internal high-speed clock requires a short time from start up to stabilization. In order to respond to interrupts quickly in DeepSleep mode, it is recommended to switch the system clock source to RC48M before entering DeepSleep mode.

6.1.2.2 RCL

RCL is an internal low-speed clock with configurable and calibratable frequency.

The clock frequency can be configured through RCL_CR register, two preset calibrated frequency configuration values before leaving the factory: 32.768kHz and 38.4kHz, which can be directly loaded for use; it can also adjust the calibration value in this register to change to other frequencies or achieve higher-precision calibration according to actual needs.

When the system enters DeepSleep mode, this clock will not be automatically turned off, and the ultra-low power peripheral module can choose RCL as its clock.

6.1.2.3 EXTH

EXTH supports external active clock as input but does not support external crystal connection.

The clock signal is directly input from the EXTH_IN pin when in use. The method for inputting an active clock signal from EXTH_IN is as follows:

- Configure the EXTH_IN pin as a GPIO input;
- Set SYSCTRL1.EXTH_EN to 1.



Notice

The external clock source signal must comply with the relevant requirements for EXTH specified in the Electrical Specifications chapter of the Datasheet.

6.1.2.4 XTL

XTL external low-speed crystal oscillator clock requires an external 32.768kHz low-power crystal oscillator, which has ultra-high precision and ultra-low power. When the system enters DeepSleep mode, this clock will not shut down automatically. Peripheral modules operating in ultra-low power mode can select XTL as their clock.

XTL supports an external active clock as input, allowing the input clock signal directly from the EXTL_IN pin without connecting a crystal. The method for inputting an active clock signal from EXTL_IN is as follows:

- Configure the EXTL_IN pin as a GPIO input;
- Configure the XTL stable time XTL_CR.STARTUP;
- Set SYSCTRL1.EXTL_EN to 1;
- Wait for the XTL stable flag to be set XTL_CR.STABLE.



Notice

The crystal, its matching capacitors, and the external clock source signal must comply with the requirements for the low-speed external clock XTL specified in the Electrical Specifications chapter of the Datasheet.

6.1.3 Clock Start Process

All of the above clock sources require a startup stabilization time. Taking XTL as example to illustrate the process of the clock startup and stabilization.

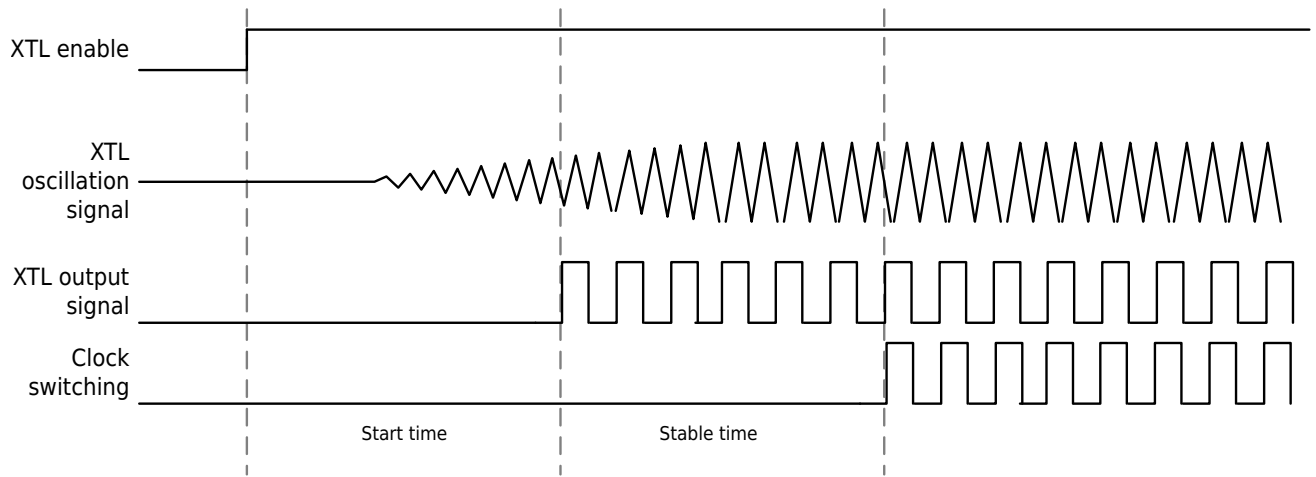


Figure 6-2 Crystal Oscillator Clock Startup Diagram

6.2 System Clock Switching

6.2.1 System Clock Switching Introduction

The clock source of the system clock can be switched among RCL, EXTH, XTL, and RC48M through SYSCTRL0.CLKSW. When switching, any two of the clock sources RCL, EXTH, XTL, and RC48M can be switched between each other. The clock switching operation must be performed according to clock switching processes described below, otherwise abnormalities may occur.

The system clock source can be used as HCLK after system division, an appropriate frequency division value can be configured via SYSCTRL0.HCLK_PRSC.

When switching the clock, FLASH_CR.WAIT must be configured accordingly based on the current and target HCLK frequencies.

- When $HCLK \leq 24MHz$, FLASH_CR.WAIT should be set to 0.
- When $HCLK > 24MHz$, FLASH_CR.WAIT should be set to 1;



Note

For detailed instructions on setting FLASH_CR.WAIT, please refer to the chapter "FLASH Controller (FLASH)".

6.2.2 Switching Process

6.2.2.1 Standard Clock Switching Process

Procedure

- Step1.** If the new clock source requires an external pin, set the pin to the appropriate mode.



Note

An analog pin is required when connecting to an external crystal oscillator; GPIO input is required and the external clock input is enabled when connecting to an external clock input.

- Step2.** Configure the oscillation parameters of the new clock source.
- Step3.** Enable the oscillator of the new clock source.
- Step4.** If the frequency of the new clock source is higher than that of the previous one, configure the FLASH_CR.WAIT before switching, based on the new clock source frequency, following the procedure described in the chapter FLASH Controller.
- Step5.** Wait for the new clock source to output a stable frequency.

- Step6.** Configure SYCTRL0.CLKSW to select the source of the system clock as the new clock source.
- Step7.** If the frequency of the new clock source is lower than that of the previous one, configure the FLASH_CR.WAIT after switching, based on the new clock source frequency, following the procedure described in the chapter FLASH Controller.
- Step8.** Turn off clock sources that are no longer in use.

6.2.2.2 RC48M Switching Process Between Different Oscillation Frequencies

Context

Choose one of the following methods according to the usage scenario:

Procedure

- Load according to the target frequency: directly load data from the factory preset target frequency calibration value storage address into the RC48M_CR.
- Set directly according to actual needs: write the calibration value to the RC48M_CR, adjust the target output frequency of the RC48M.

6.2.3 Examples of Switching

6.2.3.1 Example of Switching from Other Clocks to XTL

Procedure

- Step1.** Set XTLI and XTLO pin's ADS Bit to 1, configure XTLI/XTLO pins as analog ports.
- Step2.** Configure the parameters in the XTL_CR register according to the features of the crystal and the system.
- Step3.** Write 0x5A5A and 0xA5A5 to SYCTRL2 register in sequence to enable register rewriting.
- Step4.** Set SYCTRL0.XTL_EN to 1 to enable the crystal oscillator circuit.
- Step5.** Query and wait for the XTL_CR.STABLE flag to become 1, and the crystal oscillator outputs a stable clock.
- Step6.** Write 0x5A5A and 0xA5A5 to SYCTRL2 register in sequence to enable register rewriting.
- Step7.** Set SYCTRL0.CLKSW to 3, switch the system clock to XTL.
- Step8.** See the chapter FLASH Controller to set FLASH_CR.WAIT to 0.
- Step9.** Write 0x5A5A and 0xA5A5 to SYCTRL2 register in sequence to enable register rewriting.
- Step10.** Configure the control bits in SYCTRL0 or SYCTRL1. Turn off the clock source that is no longer used.

6.2.3.2 Example of Switching from Other Clocks to RCL

Procedure

- Step1.** Configure RCL_CR.TRIM.
- Step2.** Write 0x5A5A and 0xA5A5 to SYCTRL2 register in sequence to enable register rewriting.
- Step3.** Set SYCTRL0.RCL_EN to 1 to enable the RCL oscillation circuit.
- Step4.** Query and wait for RCL_CR.STABLE flag to become 1, and the RCL outputs a stable clock.
- Step5.** Write 0x5A5A and 0xA5A5 to SYCTRL2 register in sequence to enable register rewriting.
- Step6.** Set SYCTRL0.CLKSW to 2, switch the system clock to RCL.
- Step7.** See the chapter FLASH Controller to set FLASH_CR.WAIT to 0.
- Step8.** Write 0x5A5A and 0xA5A5 to SYCTRL2 register in sequence to enable register rewriting.
- Step9.** Configure the control bits in SYCTRL0 or SYCTRL1. Turn off the clock source that is no longer used.

6.2.3.3 Example of Switching from Other Clocks to RC48M

Procedure

- Step1.** According to the higher frequency between the current clock and the RC48M clock to be configured, configure FLASH_CR.WAIT following the procedure described in chapter FLASH Controller.
- Step2.** Write the preset value to the RC48M_CR register to configure the desired clock frequency.
- Step3.** Write 0x5A5A and 0xA5A5 to SYCTRL2 register in sequence to enable register rewriting.
- Step4.** Set SYCTRL0.RC48M_EN to 1, enable RC48M oscillation circuit.
- Step5.** Query and wait for RC48M_CR.STABLE flag to become 1, RC48M outputs a stable clock.
- Step6.** Write 0x5A5A and 0xA5A5 to SYCTRL2 register in sequence to enable register rewriting.
- Step7.** Set SYCTRL0.CLKSW to 0, switch the system clock to RC48M.
- Step8.** According to the frequency of RC48M, configure FLASH_CR.WAIT according to the procedures in chapter Flash Controller.
- Step9.** Write 0x5A5A and 0xA5A5 to SYCTRL2 register in sequence to enable register rewriting.
- Step10.** Configure the control bits in SYCTRL0 or SYCTRL1. Turn off the clock source that is no longer used.

6.2.4 Clock Switching Diagram

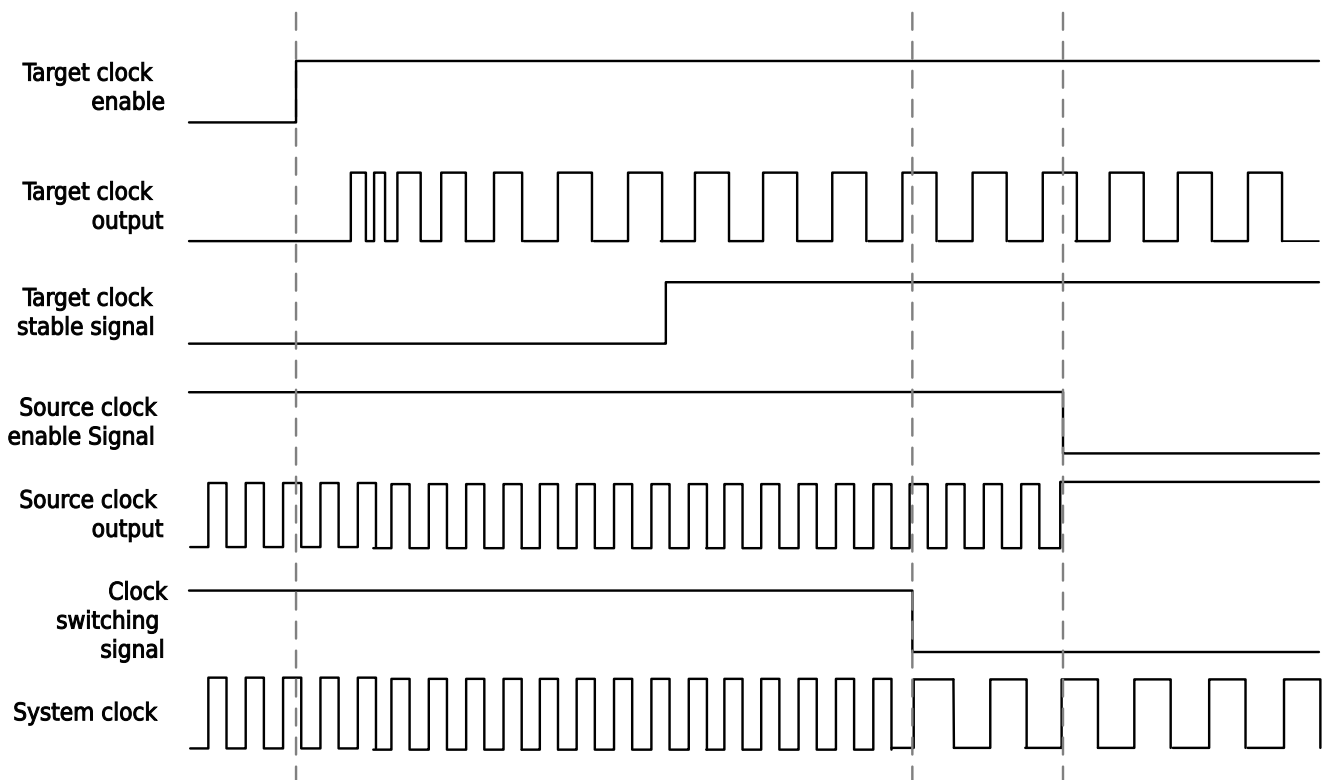


Figure 6-3 Clock Switching Diagram

6.3 Clock Calibration Module

RC48M and RCL support calibration using an external higher-precision clock source. This product integrates an internal clock calibration circuit CTRIM modules. For more information, see the relevant chapter descriptions.

6.4 Interrupt Wakeup Control

When the processor executes the WFI instruction to enter the Sleep/DeepSleep state, it will stop executing instructions. When an interrupt request (higher priority) occurs during Sleep/DeepSleep and needs to be serviced, the processor is woken up.

The behavior of the processor in the Sleep/DeepSleep state when receiving an interrupt request is shown in the following table:

PRIMASK Status	WFI Behavior	Wakeup	ISR Execution
0	IRQ Priority > Current Class	Y	Y
0	IRQ Priority ≤ Current Class	N	N
1	IRQ Priority > Current Class	Y	N
1	IRQ Priority ≤ Current Class	N	N

6.4.1 Executing Interrupt Service after Waking from DeepSleep mode

Procedure

- Step1.** Enable the NVIC corresponding to the module that needs to wake up the processor.
- Step2.** Enable the interrupt corresponding to the module that needs to wake up the processor.
- Step3.** Set SCB->SCR.SLEEPDEEP to 1.
- Step4.** Execute WFI instruction enter the DeepSleep mode.
- Step5.** The system enters the DeepSleep mode and waits for the interrupt to wake up, and executes the interrupt service program after waking up.

Examples

```
SCB->SCR |= 0x00000004u;
while(1)
{
    __asm("WFI");
}
```

6.4.2 Interrupt Service Not Executed after Waking from DeepSleep mode

Procedure

- Step1.** Enable the NVIC corresponding to the module that needs to wake up the processor.
- Step2.** Enable the interrupt corresponding to the module that needs to wake up the processor.
- Step3.** Set PRIMASK to 1.
- Step4.** Set SCB->SCR.SLEEPDEEP to 1.
- Step5.** Execute WFI instruction enter the DeepSleep mode.
- Step6.** The system enters the DeepSleep mode and waits for the interrupt to wake up, and executes the next instruction after waking up;
- Step7.** Clear interrupt flag, clear interrupt pending status.

Step8. Execute user-defined operations.

Examples

```
__asm("CPSID I");//Set PRIMASK
SCB->SCR |= 0x00000004u;
while(1)
{
    __asm("WFI");
    ...//Clear the interrupt flag of the wake-up interrupt
    ...//Clear the interrupt pending status of the wake-up interrupt
    ...//Perform user-defined operation
}
```

6.4.3 Using Sleep-on-exit Feature

Context

Sleep-on-exit is ideal for interrupt-driven applications. When this feature is enabled, the processor enters Sleep/DeepSleep mode whenever exception handling is completed and returns to thread mode. With the sleep-on-exit feature, the processor can be in Sleep/DeepSleep mode as much as possible.

Cortex-M0+ take use of the sleep-on-exit feature to enter Sleep/DeepSleep. This situation is similar to the effect of executing WFI immediately after executing an abnormal exit. Attention: in order to avoid the need to push the stack when entering an exception next time, the processor will not perform the process of popping the stack.

Procedure

- Step1.** Enable the NVIC corresponding to the module that needs to wake up the processor.
- Step2.** Enable the interrupt corresponding to the module that needs to wake up the processor.
- Step3.** Set SCB->SCR.SLEEPDEEP to 1.
- Step4.** Set SCB->SCR.SLEEPONEXIT to 1.
- Step5.** Execute WFI instruction enter the DeepSleep mode.
- Step6.** The system enters the DeepSleep mode and waits for the interrupt to wake up, and executes the interrupt service subroutine after waking up.
- Step7.** Automatically enters DeepSleep mode when exiting interrupt service.

Examples

```
SCB->SCR |= 0x00000004u;
SCB->SCR |= 0x00000002u;
while(1)
{
    __asm("WFI");
}
```

6.5 Reset Control

6.5.1 Reset Controller Overview

This product has 6 reset signal sources, each of which can make the CPU run again. Most registers will be reset to reset values, and the program will start executing from the reset vector.

- Digital area power-on and power-off reset POR
- External Reset PAD, low level is reset signal
- IWDT reset
- LVD low voltage reset
- Cortex-M0+ SYSRESETREQ software reset
- Cortex-M0+ LOCKUP hardware reset

Each reset source is indicated by a corresponding reset flag. Reset flags are set by hardware and need to be cleared by user software. When the chip is reset, if RESET_FLAG.POR15V or RESET_FLAG.POR5V is 1, it is a **power-on reset**. When **power-on a reset**, the user program should clear the RESET_FLAG register to zero. Then, the reset source can be determined by the relevant bits of RESET_FLAG during the next reset.

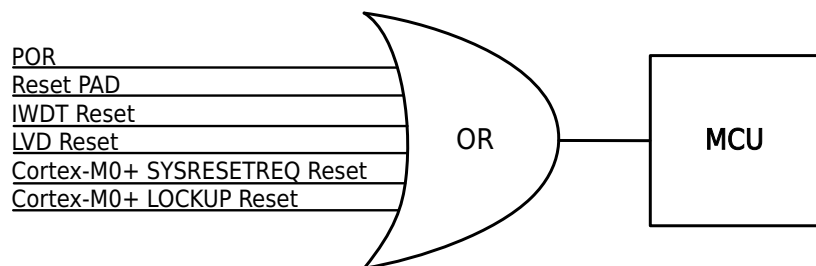


Figure 6-4 Reset Source Diagram

6.5.2 Reset Function

6.5.2.1 Power-on and Power-off Reset POR

This product has two power supply areas: VCC area and VCAP area. All analog modules and IO work in the VCC area; other modules work in the VCAP area.

- When the VCC area is powered on, when the VCC voltage is lower than the POR threshold voltage, POR5V signal is generated; when the VCC area is powered off, when the VCC voltage is lower than the BOR threshold voltage, POR5V signal is generated.
- When the VCAP area is powered on, when the VCAP voltage is lower than the POR threshold voltage, POR15V signal is generated; when the VCAP area is powered off, when the VCAP voltage is lower than the BOR threshold voltage, POR15V signal is generated.

Both the POR5V signal and the POR15V signal will reset the chip's registers to the initialization state.

6.5.2.2 External Reset Pin Reset

A system reset is generated when the external reset pin detects a low level. The reset pin has a built-in pull-up resistor and integrates a glitch filter circuit. The glitch filter circuit filters out glitches shorter than a certain time width. Therefore, the low-level signal applied to the reset pin must exceed this time width to ensure a reliable reset of the chip. For specific timing parameters, see the RESETB Pin Features section under the chapter "Electrical Specifications" in the Datasheet.

6.5.2.3 Watchdog Reset

For watchdog reset, refer to the relevant introduction in the Independent Watchdog (IWDT) chapter.

6.5.2.4 LVD Low Voltage Reset

For LVD low voltage reset, refer to the relevant introduction in the "Low Voltage Detector (LVD)" chapter.

6.5.2.5 Cortex-M0+ SYSRESETREQ Reset

Cortex-M0+ software reset.

6.5.2.6 Cortex-M0+ LOCKUP Reset

When Cortex-M0+ encounters a serious exception, it will stop its own PC pointer at the current address and lock it, and reset the entire CORE area after a delay of several clock cycles. This reset function is enabled when SYCTRL1.LOCKUP_EN is set to 1.

6.6 Register Description

6.6.1 Register List

Base address: 0x40021800

Table 6-3 System Controller (SYCTRL) Register Offset

Offset	Register	Description	Width
0x00	SYCTRL0	System Control Register 0	32
0x04	SYCTRL1	System Control Register 1	32
0x08	SYCTRL2	System Control Register 2	32
0x0C	RC48M_CR	RC48M Control Register	32
0x14	RCL_CR	RCL Control Register	32
0x18	XTL_CR	XTL Control Register	32
0x1C	RESET_FLAG	Reset Flag Register	32
0x20	PERI_CLKEN0	Peripheral Module Clock Control Register 0	32
0x24	PERI_CLKEN1	Peripheral Module Clock Control Register 1	32
0x28	PERI_RESETO	Peripheral Module Reset Control Register 0	32
0x2C	PERI_RESET1	Peripheral Module Reset Control Register 1	32
0x38	DEBUG_ACTIVE	Peripheral Debug Function Enable Register	32

6.6.2 System Control Register 0 (SYSCTRL0)

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00008001																															
Name	Reserved																WAKEUP_BYRC48M	Reserved				HCLK_PRS		CLKSW		Reserved	XTL_EN	RCL_EN	Reserved	RC48M_EN		
Access																	RW					RW		RW			RW	RW		RW		

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15	WAKE-UP_BYRC48M	System clock source control bit after wake-up 0b0: After waking up from DeepSleep mode, the system clock source remains unchanged. 0b1: After waking up from DeepSleep mode, the system clock source switches to RC48M, and the original clock source remains enabled.  Note When the debugger is connected, the chip will not enter DeepSleep mode.	RW
14:11	Reserved	-	-
10:8	HCLK_PRS	HCLK system frequency division selection 0b000: SystemClk 0b001: SystemClk/2 0b010: SystemClk/4 0b011: SystemClk/8 0b100: SystemClk/16 0b101: SystemClk/32 0b110: SystemClk/64 0b111: SystemClk/128	RW
7:5	CLKSW	SystemClk clock source selection 0b000: RC48M 0b001: EXTH 0b010: RCL 0b011: XTL - Others: Setting prohibited	RW
4	Reserved	-	-

Bit	Name	Functional Description	Access
3	XTL_EN	External low-speed crystal oscillator XTL enable control ● 0b0: Disable ● 0b1: Enable	RW
2	RCL_EN	Internal low-speed clock RCL enable control ● 0b0: Disable ● 0b1: Enable	RW
1	<i>Reserved</i>	-	-
0	RC48M_EN	Internal high-speed clock RC48M enable control ● 0b0: Disable ● 0b1: Enable	RW

6.6.3 System Control Register 1 (SYSCTRL1)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000008																															
Name	Reserved																					CTIMER1_FUN_SEL	CTIMER0_FUN_SEL	SWD_USE_IO	SPI_PCTRL	LOCKUP_EN	Reserved	CLOCK_FAULT_EN	XTL_ALWAYS_ON	EXTL_EN	EXTH_EN	RESET_USE_IO
Access																						RW	RW	RW	RW	RW		RW	RW	RW	RW	RW

Bit	Name	Functional Description	Access
31:11	Reserved	-	-
10	CTIMER1_FUN_SEL	CTIM1 function selection 0b0: CTIM1 function as a General Timer (GTIM). 0b1: CTIM1 function as a Basic Timer (BTIM).	RW
9	CTIMER0_FUN_SEL	CTIM0 function selection 0b0: CTIM0 function as a General Timer (GTIM). 0b1: CTIM0 function as a Basic Timer (BTIM).	RW
8	SWD_USE_IO	SWD port function configuration 0b0: SWD port 0b1: GPIO port	RW
7	SPI_PCTRL	SPI port control selection 0b0: SPI control 0b1: GPIO control	RW
6	LOCKUP_EN	Cortex-M0+ LockUp enable 0b0: Disable 0b1: Enable	RW
5	Reserved	-	-
4	CLOCK_FAULT_EN	Clock error detection enable 0b0: Disable 0b1: Enable	RW
3	XTL_ALWAYS_ON	XTL enable permission control 0b0: SYSCTRL0.XTL_EN can be set and cleared. 0b1: SYSCTRL0.XTL_EN can only be set.	RW

Bit	Name	Functional Description	Access
2	EXTL_EN	External XTL clock source control 0b0: XTL is generated by a crystal. 0b1: XTL clock source is input from EXTL_IN, when using this clock as the system clock, EXTL_EN must not be disabled (the external clock source must comply with the requirements for the low-speed external clock XTL specified in the Electrical Specifications chapter of the Datasheet).	RW
1	EXTH_EN	EXTH clock enable control 0b0: The EXTH clock is disabled. 0b1: EXTH clock enable, and EXTH clock source is input from EXTH_IN  Note When EXTH is used as the system clock source, it cannot directly enter DeepSleep mode. To enter DeepSleep mode, the system clock must be switched to RC48M. When using this clock as the system clock, EXTH_EN must not be disabled.	RW
0	RE-SET_USE_IO	RESET PAD function configuration 0b0: as PADRSTB function 0b1: as GPIO	

6.6.4 System Control Register 2 (SYCTRL2)

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																SYCTRL2															
Access																	WO															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	SYCTRL2	SYCTRL0 register and SYCTRL1 protection control register Write 0x5A5A first to SYCTRL2, and then write 0xA5A5 to start the write operation to SYCTRL0 and SYCTRL1 register. As long as the write operation is performed to SYCTRL0 or SYCTRL1 register, this protection bit automatically recovery to the protection state. Subsequent write operations to SYCTRL0 or SYCTRL1 require repeating the previous steps.	WO

6.6.5 RC48M Control Register (RC48M_CR)

Offset	Bit Position															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0x0X00XXXX (The power-on reset value is the value stored at the 4MHz calibration value address)															
Name	STABLE	Reserved								FSEL	TRIMC				TRIMF	Reserved
Access	RO									RW	RW				RW	

Bit	Name	Functional Description	Access
31	STABLE	RC48M clock stable flag bit 0b0: RC48M is not stable and cannot be used by internal circuits. 0b1: RC48M is stable and can be used by internal circuits.	RO
30:18	Reserved	-	-
17:16	FSEL	Operating frequency selection This bit does not require independent configuration. For parameter selection, refer to the instructions. 0b00: 4MHz 0b01: 6MHz 0b10: 32MHz 0b11: 48MHz  Note Switching FSEL is not allowed during Flash programming and erasing.	RW
15:8	TRIMC	Coarse adjustment of clock frequency Parameter selection details are provided in the instructions.	RW
7:2	TRIMF	Clock frequency fine tuning Parameter selection details are provided in the instructions.	RW
1:0	Reserved	-	-

**Note**

RC48M clock frequency adjustment: changing the value of this register allows you to adjust the output frequency of RC48M.

The Flash already contains several sets of calibration values and configuration parameters. By reading all the data from the specified addresses in Flash and writing them into the entire RC48M_CR register, you can obtain an accurate frequency.

- 48MHz calibration value address: 0x001007B0 - 0x001007B3
- 32MHz calibration value address: 0x001007B4 - 0x001007B7
- 6MHz calibration value address: 0x001007B8 - 0x001007BB
- 4MHz calibration value address: 0x001007BC - 0x001007BF

6.6.6 RCL Control Register (RCL_CR)

Offset	Bit Position																																			
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x0X000XXX (The TRIM value after power-on reset is the value stored at the 32.768kHz calibration value address)																																			
Name	Reserved																MSEL	Reserved			STABLE	Reserved			TRIM											
Access																	RW				RO				RW											

Bit	Name	Functional Description	Access
31:17	Reserved	-	-
16	MSEL	Pattern selection 0b0: High performance mode (HPM) 0b1: Low power mode (LPM)  Note This bit does not support independent configuration. For configuration details, refer to the description of TRIM bit.	RW
15:13	Reserved	-	-
12	STABLE	Internal low-speed clock RCL stable flag 0b0: RCL is not stable and cannot be used by internal circuits. 0b1: RCL is stable and can be used by internal circuits.	RO
11:10	Reserved	-	-

Bit	Name	Functional Description	Access
9:0	TRIM	Internal low-speed clock frequency adjustment 2 sets of frequency calibration values are stored in Flash. ● 38.4kHz calibration value address: 0x001007AC-0x001007AF ● 32.768kHz calibration value address: 0x001007A8-0x001007AB To achieve precise frequency output, read the calibration values from Flash and write them to the RCL_CR register after performing the following operations: ● 38.4kHz High-Performance Mode: <pre>*((unsigned int *)0x001007AC)</pre> ● 38.4kHz Low-Power Mode: <pre>*((unsigned int *)0x001007AC) 0x10000</pre> ● 32.768kHz High-Performance Mode: <pre>*((unsigned int *)0x001007A8)</pre> ● 32.768kHz Low-Power Mode: <pre>*((unsigned int *)0x001007A8) 0x10000</pre>  Note The interval between loading calibration values must be greater than 6 RCL clock cycles.	RW

6.6.7 XTL Control Register (XTL_CR)

Offset	Bit Position																															
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000029																															
Name	Reserved																					FAULT_STATE	DETEN	DETIME	DETIE	STABLE	STARTUP	AMP_SEL	DRIVER			
Access																						RW	RW	RW	RW	RO	RW	RW	RW			

Bit	Name	Functional Description	Access
31:11	Reserved	-	-
10	FAULT_STATE	XTL work state flag Writing 0 clears the flag, writing 1 has no effect Read: 0b0: Works fine 0b1: Fails to work	RW
9	DETEN	Oscillation stop detection enable 0b0: Disable oscillator stop detection 0b1: Enable oscillator stop detection  Note To enable oscillator stop detection when an external low-speed crystal oscillator is connected, SYSCTRL0.XTH_EN must be set to 1, and XTH_CR.STABLE must read as 1.	RW
8	DETIME	Crystal failure detection interval selection 0b0: Gear 1, 2ms detection interval 0b1: Gear 2, 8ms detection interval  Note This bit must not be modified when SYSCTRL0.XTL_EN is set to 1.	RW
7	DETIE	Failure detection and operational failure detection interrupt enable control 0b0: Disable 0b1: Enable	RW
6	STABLE	External low-speed crystal oscillator stable flag bit 0b0: XTL is not stable and cannot be used by internal circuits. 0b1: XTL is stable and can be used by internal circuits.	RO
5:4	STARTUP	External low-speed crystal oscillator stable time selection 0b00: 4096 cycles 0b01: 8192 cycles 0b10: 16384 cycles (Recommended value) 0b11: 32768 cycles	RW

Bit	Name	Functional Description	Access
3:2	AMP_SEL	XTL crystal oscillation amplitude adjustment ● 0b00: minimum amplitude ● 0b01: normal amplitude ● 0b10: larger amplitude (Recommended value) ● 0b11: maximum amplitude	RW
1:0	DRIVER	XTL crystal oscillator driving capability selection ● 0b00: Weakest driving capability ● 0b01: Weaker driving capability (Recommended value) ● 0b10: Stronger driving capability ● 0b11: Strongest driving capability  Note It is necessary to select an appropriate driving capability according to the characteristics of the crystal oscillator, the load capacitance and the parasitic parameters of the circuit board. The greater the driving capability, the greater the power consumption; the weaker the driving capability, the smaller the power consumption.	RW

6.6.8 Reset Flag Register (RESET_FLAG)



Note

The reset value is 0x03 at power-on reset. After the program determines the reset source, it should clear the register so that it can determine the reset source the next time a reset occurs.

Offset	Bit Position																																					
0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000003																																					
Name	Reserved																								RSTB	SYSREQ	LOCKUP	Reserved	IWDT	LVD	POR15V	POR5V						
Access																									RW	RW	RW		RW	RW	RW	RW						

Bit	Name	Functional Description	Access
31:8	Reserved	-	-
7	RSTB	RESETB port reset flag Write 0 to clear flag, write 1 has no effect Read: 0b0: No reset occurred 0b1: Reset occurred	RW
6	SYSREQ	Cortex-M0+ CPU software reset flag Write 0 to clear flag, write 1 has no effect Read: 0b0: No reset occurred 0b1: Reset occurred	RW
5	LOCKUP	Cortex-M0+ CPU Lockup reset flag Write 0 to clear flag, write 1 has no effect Read: 0b0: No reset occurred 0b1: Reset occurred	RW
4	Reserved	-	-
3	IWDT	IWDT reset flag Write 0 to clear flag, write 1 has no effect Read: 0b0: No reset occurred 0b1: Reset occurred	RW
2	LVD	LVD reset flag Write 0 to clear flag, write 1 has no effect Read: 0b0: No reset occurred 0b1: Reset occurred	RW

Bit	Name	Functional Description	Access
1	POR15V	VCAP domain reset flag Write 0 to clear flag, write 1 has no effect Read: ● 0b0: No reset occurred ● 0b1: Reset occurred	RW
0	POR5V	VCC power domain reset flag Write 0 to clear flag, write 1 has no effect Read: ● 0b0: No reset occurred ● 0b1: Reset occurred	RW

6.6.9 Peripheral Module Clock Control Register 0 (PERI_CLKEN0)

Offset	Bit Position																															
0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x80800000																															
Name	FLASH	HSI2C	Reserved	GPIO	Reserved								RTC	Reserved	TRNG	VC	ADC	IWDT	Reserved				ATIM3	Reserved				SPI	Reserved	LPUART1	LPUART0	Reserved
Access	RW	RW			RW									RW		RW	RW	RW	RW					RW					RW		RW	RW

Bit	Name	Functional Description	Access
31	FLASH	FLASH controller module clock enable After closing, the FLASH configuration registers cannot be written, and the programs in the FLASH can still run. 0b0: Disable 0b1: Enable	RW
30	HSI2C	HSI2C module clock enable 0b0: Disable 0b1: Enable	RW
29	Reserved	-	-
28	GPIO	GPIO module clock enable 0b0: Disable 0b1: Enable	RW
27:21	Reserved	-	-
20	RTC	RTC module clock enable 0b0: Disable 0b1: Enable	RW
19	Reserved	-	-
18	TRNG	TRNG module clock enable 0b0: Disable 0b1: Enable	RW
17	VC	VC, LVD module clock enable 0b0: Disable 0b1: Enable	RW
16	ADC	ADC module clock enable 0b0: Disable 0b1: Enable	RW
15	IWDT	IWDT module clock enable 0b0: Disable 0b1: Enable	RW
14:12	Reserved	-	-

Bit	Name	Functional Description	Access
11	ATIM3	ATIM3 module clock enable ● 0b0: Disable ● 0b1: Enable	RW
10:7	Reserved	-	-
6	SPI	SPI module clock enable ● 0b0: Disable ● 0b1: Enable	RW
5:4	Reserved	-	-
3	LPUART1	LPUART1 module clock enable ● 0b0: Disable ● 0b1: Enable	RW
2	LPUART0	LPUART0 module clock enable ● 0b0: Disable ● 0b1: Enable	RW
1:0	Reserved	-	-

**Note**

The above peripherals must have their corresponding control bits set to enable before use.

6.6.10 Peripheral Module Clock Control Register 1 (PERI_CLKEN1)

Offset	Bit Position																																			
0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																														CTRIM		CTIM1		CTIM0	
Access																															RW		RW		RW	

Bit	Name	Functional Description	Access
31:3	Reserved	-	-
2	CTRIM	CTRIM module clock enable 0b0: Disable 0b1: Enable	RW
1	CTIM1	CTIM1 module clock enable 0b0: Disable 0b1: Enable	RW
0	CTIM0	CTIM0 module clock enable 0b0: Disable 0b1: Enable	RW

**Note**

The above peripherals must have their corresponding control bits set to enable before use.

6.6.11 Peripheral Module Reset Control Register 0 (PERI_RESET0)

Offset	Bit Position																																						
0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
Reset	0x5017084C																																						
Name	Reserved		Reserved		Reserved									Reserved																									
	HSI2C												RTC		TRNG	VC	ADC	Reserved								ATIM3													
Access		RW		RW									RW		RW	RW	RW									RW													

Bit	Name	Functional Description	Access
31	Reserved	-	-
30	HSI2C	HSI2C block reset enable 0b0: In reset state 0b1: Normal working status	RW
29	Reserved	-	-
28	GPIO	GPIO module reset enable 0b0: In reset state 0b1: Normal working status	RW
27:21	Reserved	-	-
20	RTC	RTC module reset enable 0b0: In reset state 0b1: Normal working status	RW
19	Reserved	-	-
18	TRNG	TRNG module reset enable 0b0: In reset state 0b1: Normal working status	RW
17	VC	VC module reset enable 0b0: In reset state 0b1: Normal working status	RW
16	ADC	ADC module reset enable 0b0: In reset state 0b1: Normal working status	RW
15:12	Reserved	-	-
11	ATIM3	ATIM3 module reset enable 0b0: In reset state 0b1: Normal working status	RW
10:7	Reserved	-	-
6	SPI	SPI module reset enable 0b0: In reset state 0b1: Normal working status	RW

Bit	Name	Functional Description	Access
5:4	Reserved	-	-
3	LPUART1	LPUART1 module reset enable ● 0b0: In reset state ● 0b1: Normal working status	RW
2	LPUART0	LPUART0 module reset enable ● 0b0: In reset state ● 0b1: Normal working status	RW
1:0	Reserved	-	-

6.6.12 Peripheral Module Reset Control Register 1 (PERI_RESET1)

Offset	Bit Position																															
0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000007																															
Name	Reserved																										CTRIM		CTIM1		CTIM0	
Access																											RW		RW		RW	

Bit	Name	Functional Description	Access
31:3	Reserved	-	-
2	CTRIM	CTRIM module reset enable 0b0: In reset state 0b1: Normal working status	RW
1	CTIM1	CTIM1 module reset enable 0b0: In reset state 0b1: Normal working status	RW
0	CTIM0	CTIM0 module reset enable 0b0: In reset state 0b1: Normal working status	RW

6.6.13 Peripheral Debug Function Enable Register (DEBUG_ACTIVE)

**Note**

This register setting is effective only in SWD debug mode.

Offset	Bit Position																															
0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved														CTIM1	CTIM0	Reserved				ATIM3	Reserved	RTC	IWDT	Reserved							
Access															RW	RW					RW		RW	RW								

Bit	Name	Functional Description	Access
31:18	Reserved	-	-
17	CTIM1	Configuration of CTIM1 counting function under SWD debug interface 0b0: Normal counting 0b1: Stop counting	RW
16	CTIM0	Configuration of CTIM0 counting function under SWD debug interface 0b0: Normal counting 0b1: Stop counting	RW
15:12	Reserved	-	-
11	ATIM3	Configuration of ATIM3 counting function under SWD debug interface 0b0: Normal counting 0b1: Stop counting	RW
10	Reserved	-	-
9	RTC	Configuration of RTC counting function under SWD debug interface 0b0: Normal counting 0b1: Stop counting	RW
8	IWDT	Configuration of IWDT counting function under SWD debug interface 0b0: Normal counting 0b1: Stop counting	RW
7:0	Reserved	-	-

7 GPIO

7.1 Introduction

This series supports up to 16 digital general-purpose input/output ports PA00-PA15 and a digital input port PB00 (RESET PIN reassign). The input and output signals of the analog module ADC/VC/LVD, and the input and output signals of various functional modules (such as SPI/LPUART/HSI2C/Timer, etc.) can be multiplexed with digital general-purpose input and output ports.

Each port can be configured as internal pull-up input, floating input, push-pull output, open-drain output. After the digital port is configured as an analog port, the digital function is isolated, "1" and "0" cannot be output, and the result of reading the port input value register by the CPU is "0".

When each digital port is configured as an input, it can provide an external interrupt. The interrupt type can be configured as high-level trigger, low-level trigger, rising edge trigger, and falling edge trigger. Query the interrupt flag bit of Px_STAT[n] can know the corresponding interrupt trigger port. In addition, each digital port interrupt can wake up the chip from Sleep mode/DeepSleep mode to Active mode.

7.2 Features

The general-purpose I/O supports the following features:

- Registers support AHB bus interface read and write
- Analog function pins/digital common pins/digital function pins are multiplexed
- Support pull-up/open drain output function selection
- Support interrupt in Active mode/Sleep mode/DeepSleep mode
- Support high level/low level/rising edge/falling edge trigger interrupt
- Support bit set, bit clear, bit set clear function

7.3 Functional Description

7.3.1 Port Configuration Function

Each port can be configured as an analog port or a digital port through the configuration register (PxADS) according to system requirements. When PxADS is 1, the port is configured as an analog port, and when PxADS is 0, the port is configured as a digital port.

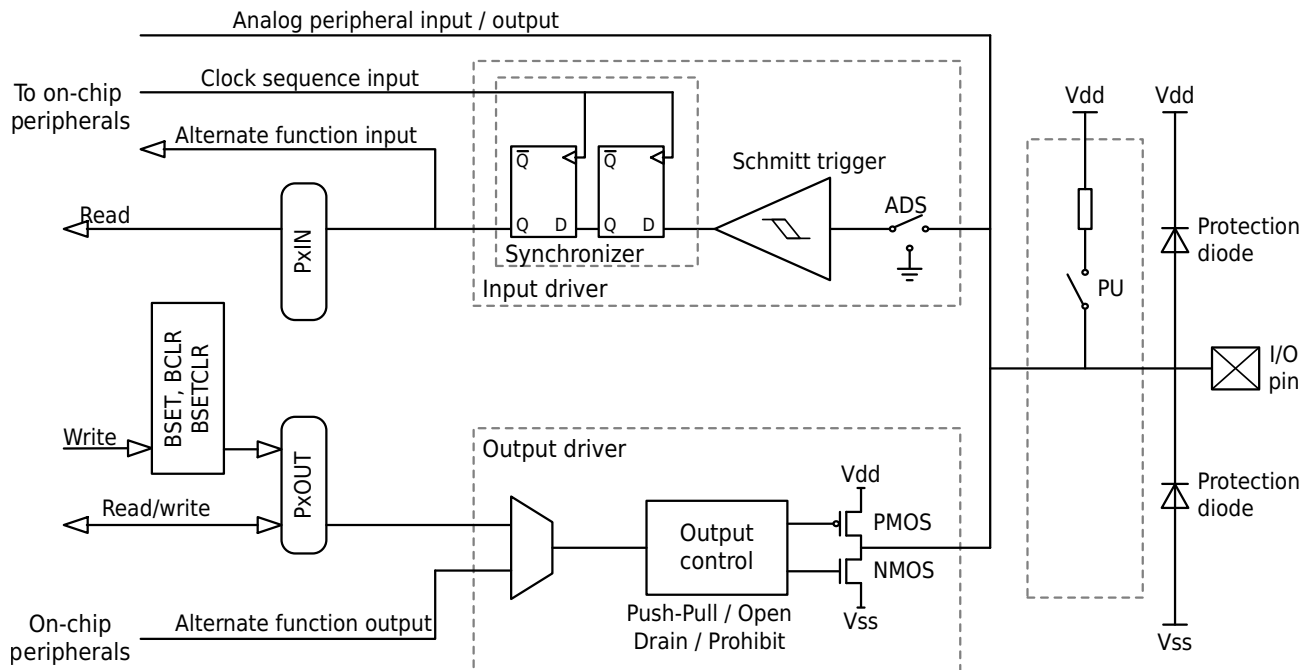


Figure 7-1 Port Circuit Diagram

When the port is configured as a digital port, it can accept digital general-purpose input and output signals (set the Pxy_SEL register to 0), and accept the input and output signals of various functional modules through the configuration register Pxy_SEL, for details, refer to the port digital multiplexing function module.

The corresponding registers can also be configured to achieve the following features:

- Internal pull-up (PxPU)

The pull-up register (PxPU) correspond to the port pull-up enable. When the corresponding bit is 1, set the corresponding bit pin pull-up enable, When it is 0, the pull-up of the corresponding bit pin is prohibited.
- Open-drain output (PxOD)

Set the pin output state through the PxOD register. When PxOD is '1', the port open-drain output is enabled, and when it is '0', the port push-pull output is enabled. When the open-drain pin is not connected to an external pull-up resistor, it can only output low level. If it needs to output high level at the same time, a pull-up resistor is required.
- Direction selection (PxDIR)

Used to set the direction of the port pin. When PxDIR is 0, the port is output, and when PxDIR is 1, the port is input.
- Input level status (PxIN)

The synchronized pin level can be obtained by reading the PxIN register. When PxIN is 1, the pin is high level, and when PxIN is 0, the pin is low level.

- Output high and low level selection (PxOUT)

When the port pin is configured as an output, if PxOUT is 1, the port pin output is high level, if PxOUT is 0, the output is low level.

- Bit set (PxBSET), bit cleared (PxBCLR), bit set cleared (PxBSETCLR)

Bit set and bit cleared are applicable to setting the bit that the user wants to change to obtain the corresponding value without changing the value of other bits. When a bit is set, the corresponding value is set to 1, and when a bit is cleared, the corresponding value is set to 0.



Note

The above features are invalid when configured as an analog port. PB00 is fixed in digital input pull-up state.

The relationship between port states and register configuration is as follows:

Table 7-1 Port Status Truth Table

IO state	IO Direction	PxADS	PxDIR	PxOUT	PxIN	PxBSET	PxBCLR	PxBSETCLR	PxPU	PxOD	Pxy_SEL
Simulation	Input/Output	1	W	W	0	W	W	W	W	W	W
Floating	Input	0	1	W	X	W	W	W	0	W	0
Pull-up	Input	0	1	W	1	W	W	W	1	W	0
1	Input	0	1	W	1	W	W	W	W	W	0
0	Input	0	1	W	0	W	W	W	W	W	0
1	Output	0	0	1	1	0	0	0	W	0	0
0	Output	0	0	0	0	0	0	0	W	0	0
1	Output	0	0	W	1	1	0	0	W	0	0
0	Output	0	0	W	0	0	1	0	W	0	0
1 (SET)	Output	0	0	W	1 (SET)	0	0	1	W	0	0
0 (CLR)	Output	0	0	W	0 (CLR)	0	0	1	W	0	0
0	Output	0	0	0	0	0	0	0	W	1	0
Z	Output	0	0	1	X	0	0	0	0	1	0
1	Output	0	0	1	1	0	0	0	1	1	0

0-Logic low, 1-Logic high, W-Whatever 0 or 1, X-unknown state, Z-high impedance

7.3.2 Port Write

The port input value/output value register (PxIN/PxOUT) supports AHB bus reading and writing. The following figure shows the fastest timing of bus port flipping:

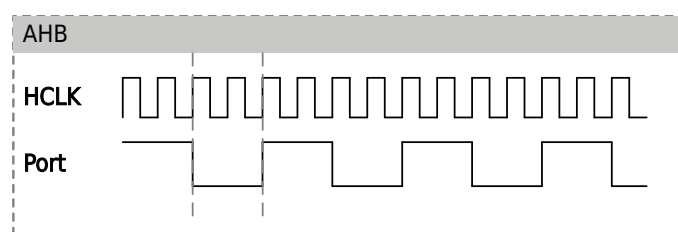


Figure 7-2 AHB Bus Port Changes with System Clock

For the AHB bus, the IO is flipped every two HCLK cycles.

7.3.3 Port Read

Each port can get the port pin level by reading the PxIN register. As [Port Circuit Diagram](#) shown, each bit of the PxIN register and the latch in front of it form a synchronizer to avoid the signal instability caused

by the pin level change in a short time when the system clock state changes, but it also introduces delay. The synchronization diagram for reading port pin data is as follows:

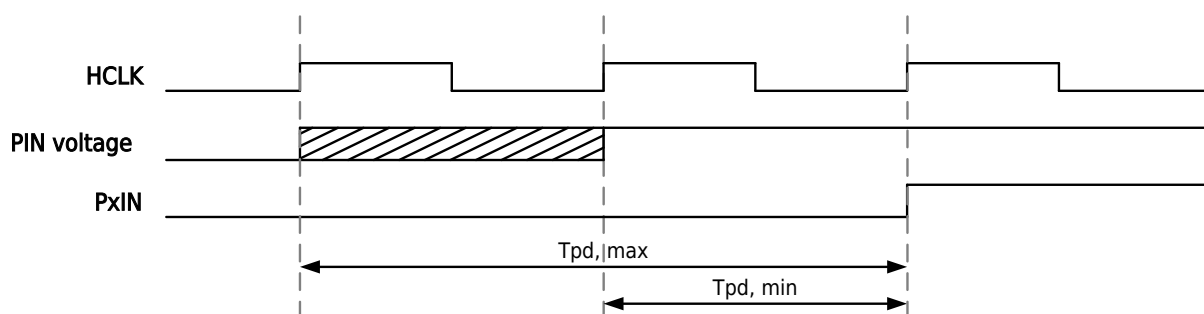


Figure 7-3 Port Pin Data Read Synchronization Diagram

During the clock period after the rising edge of the system clock, the pin level signal will be latched in the internal register, as shown in the shaded part, after the next rising edge of the system clock, the stable pin level signal can be read. Then, when the rising edge of the system clock, the data is latched into the PxIN register. The signal replacement delay T_{pd} is 1-2 system clocks.



Note

Handling of unconnected pins: If there are unconnected pins during use, in order to avoid floating current consumption caused by pins not having a certain level in other digital input enable modes, it is recommended to assign a certain level to the pin.

7.3.4 Port Digital Multiplexing Function

Port digital multiplexing is one of the main functions of the general-purpose I/O. Through the configuration register, the port can be flexibly configured as a digital general-purpose port/digital function port.

The Pxy_SEL register is used for digital general-purpose port/digital function port switching, and each port can be independently configured as a functional port required by the system.

Table 7-2 Port Multiplexing Table

AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
PA00	CTIM1_CH0	LPUART1_TXD	-	-	ATIM3_CH0A	RTC_1HZ	ATIM3_GATE
PA01	LPUART1_TXD	-	-	RTC_TAMP	ATIM3_CH0B	SPI_MISO	HSI2C_SCL
PA02	LPUART1_RXD	ATIM3_CH2A	-	CTIM0_CH0	ATIM3_CH1A	SPI_MOSI	HSI2C_SDA
PA03	LPUART0_RXD	HSI2C_SDA	-	CTIM1_TOG	ATIM3_CH1B	SPI_SCK	-
PA04	LPUART0_TXD	HSI2C_SCL	-	CTIM1_TOGN	ATIM3_CH2A	SPI_CS	ATIM3_BK
PA05	CTIM1_CH3	SPI_CS	ATIM3_CH2B	-	RTC_1HZ	CTIM0_ETR	ATIM3_ETR
PA06	HSI2C_SDA	LPUART1_CTS	ATIM3_CH0B	-	SPI_SCK	LPUART0_RXD	LVD_OUT
PA07	HSI2C_SCL	LPUART1_RTS	CTIM1_ETR	-	SPI_CS	LPUART0_TXD	ATIM3_GATE
PA08	-	ATIM3_CH2B	-	CTIM0_CH0	SPI_MISO	LPUART1_TXD	IR_OUT
PA09	-	ATIM3_CH1B	-	CTIM0_CH1	SPI_MOSI	LPUART1_RXD	-
PA10	SPI_SCK	CTIM0_CH3	-	-	LPUART0_RXD	HSI2C_SDA	-
PA11	SPI_MOSI	ATIM3_CH0A	-	CTIM0_CH2	LPUART0_TXD	HSI2C_SCL	CTIM1_ETRTOG
PA12	SPI_CS	-	ATIM3_CH1A	CTIM0_CH1	LPUART1_RXD	-	LPUART0_RTS
PA13	SPI_MISO	-	ATIM3_CH2A	CTIM1_CH1	LPUART0_RXD	CTIM0_TOG	ATIM3_BK
PA14	CTIM1_ETRTOG	CTIM1_ETR	-	VC0_OUT	LPUART0_TXD	CTIM0_TOGN	TCLK_OUT
PA15	LPUART0_CTS	CTIM1_CH2	-	VC1_OUT	LPUART1_TXD	CTIM0_ETR	RTC_1HZ
PB00	-	-	-	-	CTIM0_ETR	ATIM3_ETR	ATIM3_BK

7.3.5 Port Interrupt Function

Each digital general-purpose port can be interrupted by an external signal source. The external signal source can be four types of signals: high level/low level/rising edge/falling edge, and the corresponding interrupt enable registers are High Level Interrupts Enable Register (PxHIE)/Low Level Interrupt Enable Register (PxLIE)/Rising Edge Interrupt Enable Register (PxRIE)/Falling Edge Interrupt Enable Register (PxFIE). When an interrupt is triggered, it can be determined which port triggered the interrupt by querying the Interrupt Status Register (Px_STAT), and the corresponding interrupt status flag bit can be cleared by clearing the Interrupt Clear Register (Px_ICLR).

7.3.6 Port Input Filter Function

PA00, PA08, PB00 support digital input filter function. If the filter function is enabled, the MCU receives the filtered PAD input signal; otherwise, the MCU still receives the original PAD input signal. Both the filter clock and division factor can be configured via the Filter Configuration Register (GPIO_FLTCFG). The timing diagram of the filtering function:

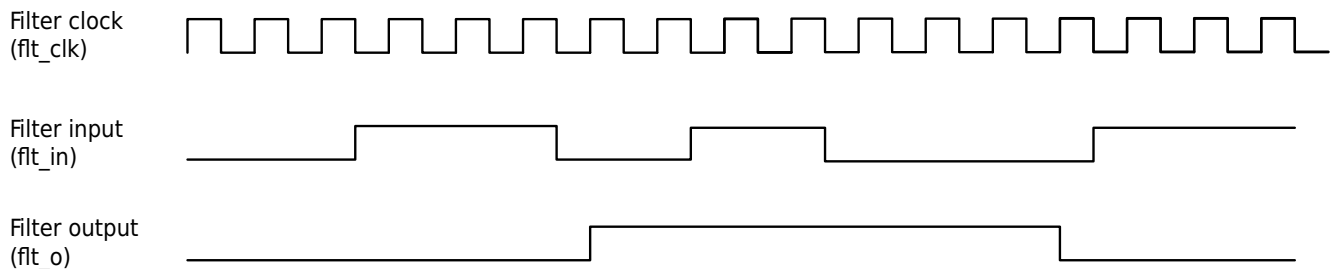


Figure 7-4 GPIO Port Input Filter Function Timing Diagram



Notice

If low-power wakeup is required, disable the filter function or switch the filter clock to RC256K.

7.3.7 Port Reset Status

During reset and before GPIO registers are configured after reset completion, except for the following I/O ports, all other ports are configured in analog port state; the status of special pins is as follows:

- SWCLK (PA14): Digital input pull-up
- SWDIO (PA13): Digital input pull-up
- PA10: Digital input pull-up
- PA11: Digital input pull-up
- RESET (PB00): Digital input pull-up

7.3.8 HSI2C Communication Level Selection Function

The HSI2C interface supports level signal communication at different voltages, that is, low voltage level recognition. And only when PA02 and PA11 selects the HSI2C function, this function is supported. Communication level voltage recognition selection is achieved by configuring GPIO_CTRL1.HSI2C_LSEL. When GPIO_CTRL1.HSI2C_LSEL is set to 0, HSI2C and GPIO input levels are recognized normally; When GPIO_CTRL1.HSI2C_LSEL is set to 1, HSI2C and GPIO input low-voltage levels are recognized.

7.4 Operation Example

7.4.1 Port Multiplexing Configured as Analog Port Operation Flow

Procedure

Set the register PxADS[n] to 1.

7.4.2 Port Multiplexing Configured as Digital Universal Port Operation Flow

Procedure

- Step1.** Set the register PxADS[n] to 0.
- Step2.** Set the register Pxy_SEL to 0.
- Step3.** Set the register PxDIR[n] to 1: the port direction is input, and the CPU can read the port status PxIN[n].
- Step4.** Set the register PxDIR[n] to 0: the port direction is output.
- Step5.** Set the register PxOUT[n] to 1: the port outputs a high level.
- Step6.** Set the register PxOUT[n] to 0: the port outputs a low level.

7.4.3 Port Multiplexing Configured as Digital Function Port Operation Flow

Procedure

- Step1.** Set the register PxADS[n] to 0.
- Step2.** According to the system requirement, set the register Pxy_SEL to 1-7, for specific configurations, refer to [the Port Multiplexing Table](#).
- Step3.** According to the system requirement, set the register PxDIR[n].
- Step4.** According to the system requirement, set the register PxPU[n]/PxOD[n].

7.4.4 Port Multiplexing Configured as Debug Test Port Operation Flow

Context

For specific operations, please refer to the relevant introduction in the "SWD Debugging Interface" chapter.

7.4.5 Port Multiplexing Configured as Infrared Output Signal Operation Flow

Context

Ports PA08 can modulate the internal clock signal with a frequency of 38.4kHz into infrared signal output.



Note

Carrier output is only generated when operating with the 38.4kHz RCL clock enabled.

Procedure

- Step1.** Set register PAADS[8] to 0.
- Step2.** Set register PA08_SEL = 7.
- Step3.** Set register PADIR[8] = 0, port direction is output.
- Step4.** Set the register GPIO_CTRL1[14] to select the output polarity of the infrared signal.
- Step5.** Set register PAOUT[8] = 1, select the gated infrared signal output.

7.4.6 Port High Level Interrupt Operation Flow

Procedure

- Step1.** Set the register PxADS[n] to 0.
- Step2.** Set the register Pxy_SEL to 0.
- Step3.** Set the register PxDIR[n] to 1.
- Step4.** Set the register PxHIE[n] to 1.
- Step5.** Read the register Px_STAT[n] after the interrupt is triggered.
- Step6.** Set the register Px_ICLR[n] to 0 to clear the register Px_STAT[n].

7.4.7 Port Low Level Interrupt Operation Flow

Procedure

- Step1.** Set the register PxADS[n] to 0.
- Step2.** Set the register Pxy_SEL to 0.
- Step3.** Set the register PxDIR[n] to 1.
- Step4.** Set the register PxLIE[n] to 1.
- Step5.** Read the register Px_STAT[n] after the interrupt is triggered.
- Step6.** Set the register Px_ICLR[n] to 0 to clear the register Px_STAT[n].

7.4.8 Port Rising Edge Interrupt Operation Flow

Procedure

- Step1.** Set the register PxADS[n] to 0.
- Step2.** Set the register Pxy_SEL to 0.
- Step3.** Set the register PxDIR[n] to 1.
- Step4.** Set the register PxRIE[n] to 1.
- Step5.** Read the register Px_STAT[n] after the interrupt is triggered.
- Step6.** Set the register Px_ICLR[n] to 0 to clear the register Px_STAT[n].

7.4.9 Port Falling Edge Interrupt Operation Flow

Procedure

- Step1.** Set the register PxADS[n] to 0.
- Step2.** Set the register Pxy_SEL to 0.
- Step3.** Set the register PxDIR[n] to 1.
- Step4.** Set the register PxFIE[n] to 1.
- Step5.** Read the register Px_STAT[n] after the interrupt is triggered.
- Step6.** Set the register Px_ICLR[n] to 0 to clear the register Px_STAT[n].

7.4.10 Port Pull-Up Enable Configuration Operation Flow

Procedure

Set the register PxPU[n] to 1.

7.4.11 Port Open-Drain Output Configuration Operation Flow

Procedure

Set the register PxOD[n] to 1.

7.4.12 Port Bit Setting Operation Flow

Procedure

Set the register PxBSET[n] to 1.

7.4.13 Port Bit Clearing Operation Flow

Procedure

Set the register PxBCLR[n] to 1.

7.4.14 Port Bit Setting and Clearing Operation Flow

Procedure

Set the register PxBSETCLR[n] to 1.

7.4.15 Port Input Filtering Operation Flow

Procedure

- Step1.** Set the register GPIO_FLTCFG.FCLK_SEL to 0 to select HCLK as the filter clock.
- Step2.** Set the register GPIO_FLTCFG.FCLK_DIV to 0b01 to select the filter clock frequency division factor to 4.
- Step3.** Set the register GPIO_FLTCFG.PA0FLT_EN to 1 to enable the filtering function for PA00.

7.5 Register Description

7.5.1 Register List

Base address: 0x40020C00

Table 7-3 General-Purpose I/O (GPIO) Register Offset

Offset	Register	Description
0x00+0x4*y	PAy_SEL	Port PAy Function Configuration Register (y=0-15)
0x40+0x4*y	PBy_SEL	Port PBy Function Configuration Register (y=0)
0x100	PADIR	Port PA Input/Output Configuration Register
0x104	PAIN	Port PA Input Value Register
0x108	PAOUT	Port PA Output Value Configuration Register
0x10C	PAADS	Port PA Digital-to-Analog Configuration Register
0x110	PABSET	Port PA Bit Set Register
0x114	PABCLR	Port PA Bit Clear Register
0x118	PABSETCLR	Port PA Bit Set Clear Register
0x120	PAPU	Port PA Pull-Up Enable Configuration Register
0x12C	PAOD	Port PA Open-Drain Output Configuration Register
0x130	PAHIE	Port PA High Level Interrupt Enable Configuration Register
0x134	PALIE	Port PA Low Level Interrupt Enable Configuration Register
0x138	PARIE	Port PA Rising Edge Interrupt Enable Configuration Register
0x13C	PAFIE	Port PA Falling Edge Interrupt Enable Configuration Register
0x200	PA_STAT	Port PA Interrupt Status Register
0x210	PA_ICLR	Port PA Interrupt Clear Register
0x144	PBIN	Port PB Input Value Register
0x170	PBHIE	Port PB High Level Interrupt Enable Configuration Register
0x174	PBLIE	Port PB Low Level Interrupt Enable Configuration Register
0x178	PBRIE	Port PB Rising Edge Interrupt Enable Configuration Register
0x17C	PBFIE	Port PB Falling Edge Interrupt Enable Configuration Register

Offset	Register	Description
0x240	PB_STAT	Port PB Interrupt Status Register
0x250	PB_ICLR	Port PB Interrupt Clear Register
0x304	GPIO_CTRL1	Port Alternate Function Configuration Register 1
0x308	GPIO_CTRL2	Port Alternate Function Configuration Register 2
0x310	GPIO_TIMES	Port Auxiliary Function Timer ETR selection
0x314	GPIO_TIMCPS	Port Auxiliary Function Timer Capture Input selection
0x330	GPIO_FLTCFG	Port Filter Configuration Register

7.5.2 Port General Register

7.5.2.1 Port Px Input/Output Configuration Register (PxDIR) (x =A)

Offset: 0x100 (PA)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																PxDIR															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxDIR	Port Px input/output configuration register Corresponding to Px15-Px00: 0b0: Configured as output 0b1: Configured as input  Note Each bit corresponds to one port, please refrain from configuring non-existent ports, for example: PxDIR[15] corresponds to port Px15.	RW

7.5.2.2 Port Px Input Value Register (PxIN) (x =A/B)

Offset: 0x104 (PA), 0x144 (PB)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	-																															
Name	Reserved																PxIN															
Access																																

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxIN	Port Px input value register Corresponding to Px15-Px00: ● 0b0: Input low level ● 0b1: Input high level  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	RO

7.5.2.3 Port Px Output Value Configuration Register (PxOUT) (x=A)

Offset: 0x108 (PA)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0XXXXXXXXX																															
Name	Reserved																PxOUT															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxOUT	Port Px output value register Corresponding to Px15-Px00: 0b0: Output low level 0b1: Output high level, if it is configured as an open-drain output, an external pull-up resistor is required to pull it high  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	RW

7.5.2.4 Port Px Digital-to-Analog Configuration Register (PxADS) (x=A)

Offset: 0x10C (PA)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000093FF																															
Name	Reserved																PxADS															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxADS	Port Px digital-to-analog configuration register Corresponding to Px15-Px00: 0b0: Configured as a digital port 0b1: Configured as an analog port  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	RW

7.5.2.5 Port Px Bit Set Register (PxBSET) (x=A)

Offset: 0x110 (PA)

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																PxBSET															
Access																	WO															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxBSET	Port Px bit set register Corresponding to Px15-Px00, writing 1 sets the bit, writing 0 has no effect on the port state.  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	WO

7.5.2.6 Port Px Bit Clear Register (PxBCLR) (x=A)

Offset: 0x114 (PA)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																PxBCLR															
Access																	WO															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxBCLR	Port Px bit clear register Corresponding to Px15-Px00, writing 1 clears the bit, writing 0 has no effect on the port state.  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	WO

7.5.2.7 Port Px Bit Set Clear Register (PxBSETCLR) (x=A)

Offset: 0x118 (PA)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	PxBSET																PxBCLR															
Access	WO																WO															

Bit	Name	Functional Description	Access
31:16	PxBSET	Port Px bit set register Corresponding to Px15-Px00, writing 1 sets the bit, writing 0 has no effect on the port state.  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	WO
15:0	PxBCLR	Port Px bit clear register Corresponding to Px15-Px00, writing 1 clears the bit, writing 0 has no effect on the port state.  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	WO

**Note**

When the same bit of PxBSET and PxBCLR is both set to 1, PxBCLR has higher priority, the port is cleared.

7.5.2.8 Port Px Pull-Up Enable Configuration Register (PxPU) (x=A)

Offset: 0x120 (PA)

Offset	Bit Position																																					
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00006C00																																					
Name	Reserved																PxPU																					
Access																																						

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxPU	Port Px pull-up enable configuration register Corresponding to Px15-Px00: ● 0b0: Disable ● 0b1: Enable  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	RW

7.5.2.9 Port Px Open-Drain Output Configuration Register (PxOD) (x=A)

Offset: 0x12C (PA)

xOffset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																PxOD															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxOD	Port Px open-drain output configuration register Corresponding to Px15-Px00: 0b0: Set the port output mode to push-pull output 0b1: Set the port output mode to open-drain output  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	RW

7.5.2.10 Port Px High Level Interrupt Enable Configuration Register (PxHIE) (x=A/B)

Offset: 0x130 (PA), 0x170 (PB)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																PxHIE															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxHIE	Port Px high level interrupt enable configuration register Corresponding to Px15-Px00: ● 0b0: Disable ● 0b1: Enable  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	RW

7.5.2.11 Port Px Low Level Interrupt Enable Configuration Register (PxLIE) (x=A/B)

Offset: 0x134 (PA), 0x174 (PB)

Offset	Bit Position																																					
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved																PxLIE																					
Access																																						

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxLIE	Port Px low level interrupt enable configuration register Corresponding to Px15-Px00: 0b0: Disable 0b1: Enable  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	RW

7.5.2.12 Port Px Rising Edge Interrupt Enable Configuration Register (PxRIE) (x=A/B)

Offset: 0x138 (PA), 0x178 (PB)

Offset	Bit Position																																					
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved																PxRIE																					
Access																																						

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxRIE	Port Px rising edge interrupt enable configuration register Corresponding to Px15-Px00: 0b0: Disable 0b1: Enable  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	RW

7.5.2.13 Port Px Falling Edge Interrupt Enable Configuration Register (PxFIE) (x=A/B)

Offset: 0x13C (PA), 0x17C (PB)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																PxFlE															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	PxFIE	Port Px falling edge interrupt enable configuration register Corresponding to Px15-Px00: 0b0: Disable 0b1: Enable  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	RW

7.5.2.14 Port Px Interrupt Status Register (Px_STAT) (x=A/B)

Offset: 0x200 (PA), 0x240 (PB)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																Px_STAT															
Access																	RO															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	Px_STAT	Port Px interrupt status register Corresponding to Px15-Px00: 0b0: No interrupt trigger 0b1: Interrupt trigger  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	RO

7.5.2.15 Port Px Interrupt Clear Register (Px_ICLR) (x=A/B)

Offset: 0x210 (PA), 0x250 (PB)

Offset	Bit Position																																					
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x0000FFFF																																					
Name	Reserved																Px_ICLR																					
Access																	WO																					

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	Px_ICLR	Port Px interrupt clear register Corresponding to Px15-Px00, write 0 to clear flag, write 1 has no effect; read always returns 1.  Note Each bit corresponds to one port, please refrain from configuring non-existent ports.	WO

7.5.3 Port Auxiliary Register

7.5.3.1 Port Auxiliary Function Configuration Register 1 (GPIO_CTRL1)

Offset	Bit Position																																
0x304	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																HSI2C_LSEL	IR_POL	Reserved														
Access																	RW	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15	HSI2C_LSEL	HSI2C I/O signal active level selection 0b0: Input level recognized normally 0b1: Input level low voltage recognition  Note For specific values, refer to the "Datasheet".	RW
14	IR_POL	IR output polarity selection 0b0: Forward output 0b1: Inverted output	RW
13:0	Reserved	-	-

7.5.3.2 Port Auxiliary Function Configuration Register 2 (GPIO_CTRL2)

Offset	Bit Position																															
0x308	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								TCLK_DIV		TCLK_SEL		Reserved			
Access																									RW		RW					

Bit	Name	Functional Description	Access
31:8	Reserved	-	-
7:6	TCLK_DIV	TCLK output frequency division selection 0b00: TCLK 0b01: TCLK/2 0b10: TCLK/4 0b11: TCLK/8	RW
5:4	TCLK_SEL	TCLK output signal source selection 0b00: RC48M 0b01: RCL 0b10: HCLK 0b11: XTL	RW
3:0	Reserved	-	-

7.5.3.3 Port Auxiliary Function Timer ETR Selection (GPIO_TIMES)

Offset	Bit Position																															
0x310	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				ATIM3_E		Reserved		CTIM1_E		Reserved	CTIM0_E				
Access																					RW				RW			RW				

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11:9	ATIM3_E	ATIM3 timer ETR input selection ● 0b000: Pxy_SEL ● 0b001: LPUART0_RXD ● 0b010: LPUART1_RXD ● 0b011: VCO_OUT ● 0b100: LVD_OUT ● Others: Setting prohibited	RW
8:5	Reserved	-	-
4:3	CTIM1_E	CTIM1 timer ETR input selection ● 0b00: Pxy_SEL ● 0b01: LPUART1_RXD ● 0b10: VC1_OUT ● 0b11: Setting prohibited	RW
2	Reserved	-	-
1:0	CTIM0_E	CTIM0 timer ETR input selection ● 0b00: Pxy_SEL ● 0b01: LPUART0_RXD ● 0b10: VCO_OUT ● 0b11: LVD_OUT	RW

7.5.3.4 Port Auxiliary Function Timer Capture Input Selection (GPIO_TIMCPS)

Offset	Bit Position																															
0x314	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				ATIM3_C0A		Reserved				CTIM1_C0		Reserved		CTIM0_C0	
Access																					RW						RW				RW	

Bit	Name	Functional Description	Access
31:11	Reserved	-	-
10:9	ATIM3_C0A	ATIM3 timer CH0A input selection 0b00: Pxy_SEL 0b01: LPUART0_RXD 0b10: LPUART1_RXD 0b11: VC1_OUT	RW
8:5	Reserved	-	-
4:3	CTIM1_C0	CTIM1 timer CH0 input selection 0b00: Pxy_SEL 0b01: LPUART0_RXD 0b10: VCO_OUT 0b11: Setting prohibited	RW
2	Reserved	-	-
1:0	CTIM0_C0	CTIM0 timer CH0 input selection 0b00: Pxy_SEL 0b01: LPUART1_RXD 0b10: VC1_OUT 0b11: LVD_OUT	RW

7.5.3.5 Port Filter Configuration Register (GPIO_FLTCFG)

Offset	Bit Position																																	
0x330	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																					FCLK_SEL	FCLK_DIV		PA0FLT_EN	PA8FLT_EN	PB0FLT_EN	Reserved						
Access																						RW	RW		RW	RW	RW							

Bit	Name	Functional Description	Access
31:11	Reserved	-	-
10	FCLK_SEL	Filter clock source selection 0b0: HCLK 0b1: RC256K	RW
9:8	FCLK_DIV	Filter clock frequency division configuration 0b00: FCLK 0b01: FCLK/4 0b10: FCLK/16 0b11: FCLK/64	RW
7	PA0FLT_EN	PA00 filter enable 0b0: Filter disable 0b1: Filter enable	RW
6	PA8FLT_EN	PA08 filter enable 0b0: Filter disable 0b1: Filter enable	RW
5	PB0FLT_EN	PB00 filter enable 0b0: Filter disable 0b1: Filter enable	RW
4:0	Reserved	-	-

7.5.4 Port Function Selection Register

7.5.4.1 Port Pxy Function Configuration Register (Pxy_SEL) (x =A/B)

Offset: 0x00+0x4*y (PA) (y= 0-15);

0x40+0x4*y (PB)(y=0);

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																														SEL	
Access																																

Bit	Name	Functional Description	Access
31:3	Reserved	-	-
2:0	SEL	Port Pxy function selection ● 0b000: GPIO ● 0b001: AF1 ● 0b010: AF2 ● 0b011: AF3 ● 0b100: AF4 ● 0b101: AF5 ● 0b110: AF6 ● 0b111: AF7  Note For the specific functions of AF1-AF7 please refer to the "Port Digital Multiplexing Function".	RW

8 NVIC

8.1 Introduction

The Cortex-M0+ processor is internally equipped with a Nested Vector Interrupt Controller (NVIC), which supports up to 32 Interrupt Request (IRQ) inputs and 1 Non Maskable Interrupt (NMI) input (not used in this system). Meanwhile, the processor supports multiple internal exceptions.

Each exception source has a separate exception number, each exception type is assigned a corresponding priority. Some exception priorities are fixed, while others are programmable, as detailed in the table below.

Table 8-1 Cortex-M0+ Processor Interrupt List

Exception Number	Exception Type	Priority	Description
1	Reset	-3 (highest)	Reset
2	NMI	-2	Non-maskable interrupt (not used in this system)
3	Hardware error	-1	Error handling exception
4-10	Reserved	-	-
11	SVC	Programmable	Invoke the hypervisor through the SVC command
12-13	Reserved	-	-
14	PendSV	Programmable	Pendable requests for system services
15	SysTick	Programmable	SysTick timer
16	Interrupt #0	Programmable	External interrupt #0
17	Interrupt #1	Programmable	External interrupt #1
...	...	...	...
47	Interrupt #31	Programmable	External interrupt #31

This chapter only introduces the processor's 32 external interrupt requests (interrupt #0-interrupt #31). For internal exceptions, see the relevant core documentation and reference guides. In addition, this chapter only discusses the interrupt handling mechanism of the NVIC in the processor core, and does not address the interrupt generation mechanisms of peripheral modules.

8.2 Functional Description

8.2.1 Interrupt Priority

Each external interrupt corresponds to a priority register, each priority is 2 bits wide, and uses the highest two bits of the interrupt priority register, and each register occupies 1 byte (8 bits). Under this setting, the available priorities are 0x00 (highest), 0x40, 0x80 and 0xC0 (lowest).

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Used		Not used, read as 0					

When multiple interrupts occur during processor execution:

- Preemption occurs if the processor is servicing an interrupt and a higher-priority interrupt is triggered. The running interrupt processing will be pended and the new interrupt will be executed instead. This process is usually called interrupt nesting. After the new interrupt is executed, the previous interrupt processing will continue and return to the program thread after it finishes.
- If the priority of the currently executing interrupt is higher than or equal to that of the new interrupt, the new interrupt will wait and enter the pending state. Once the current interrupt handler completes execution, the processor resumes by servicing the pending interrupt.
- If two interrupts occur at the same time and they have the same priority, the interrupt with the lower interrupt number will be serviced first. For example, if interrupt #0 and interrupt #1 are both enabled, have the same priority, and are triggered at the same time, interrupt #0 will be serviced first.

8.2.2 Interrupt Digraph

When the Cortex-M0+ processor is about to handle an interrupt service request, it must first determine the starting address of the exception handler. The required information is called the vector table, as illustrated in the figure below. The vector table is stored at the beginning of the memory space and contains the exception (interrupt) vectors of the exceptions (interrupts) available in the system, as well as the initial value of the main stack pointer (MSP).

Memory address		Exception number
0x000000BC	Interrupt vector #31	47
...	...	...
0x0000004C	Interrupt vector #3	19
0x00000048	Interrupt vector #2	18
0x00000044	Interrupt vector #1	17
0x00000040	Interrupt vector #0	16
0x0000003C	SysTick vector	15
0x00000038	PendSV vector	14
0x00000034	Reserved	13
0x00000030	Reserved	12
0x0000002C	SVC vector	11
0x00000028	Reserved	10
0x00000024	Reserved	9
0x00000020	Reserved	8
0x0000001C	Reserved	7
0x00000018	Reserved	6
0x00000014	Reserved	5
0x00000010	Reserved	4
0x0000000C	Hardware Fault Exception	3
0x00000008	NMI vector	2
0x00000004	Reset vector	1
0x00000000	Initial Value of MSP	0

Figure 8-1 Interrupt Vector Table

The storage order of the interrupt vector is consistent with the interrupt number. Since each vector is 1 word (4 bytes), the address of the interrupt vector is the interrupt number multiplied by 4, and each interrupt vector is the starting address of the interrupt processing.

8.2.3 Interrupt Input and Pending Behavior

In the NVIC module of the Cortex-M0+ processor, each interrupt input corresponds to a pending state register, and each register has only 1 bit, which is used to save the interrupt request, regardless of whether the request has been confirmed. When the processor starts servicing the interrupt, the hardware will automatically clear the pending status bit.

The peripherals of this system use level-triggered interrupt output. When an interrupt event occurs, the interrupt signal will be confirmed because the peripheral is connected to the NVIC. This signal remains high until the processor executes the interrupt service and clears the peripheral's interrupt signal. Inside the NVIC, when an interrupt is detected, the pending state of the interrupt will be set, and when the

processor receives the interrupt and starts executing the interrupt service routine, the pending state will be cleared. The handling process is illustrated in the following figure:

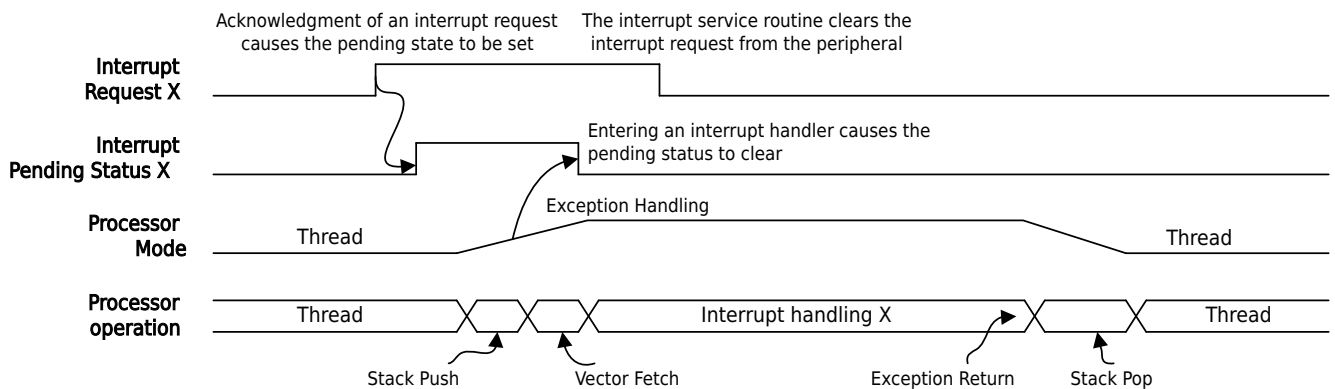


Figure 8-2 Interrupt Active and Pending States

If the interrupt request is not executed immediately and is cleared by software before being acknowledged, the processor will ignore the request and does not perform interrupt processing. When configuring a peripheral, the pending interrupt state can be cleared by writing to the NVIC_ICPR register, thereby removing any interrupt requests generated by the peripheral prior to its configuration.

If the peripheral is still holding an interrupt request when software clears the pending state, the pending state is also generated immediately. The handling process is illustrated in the following figure:

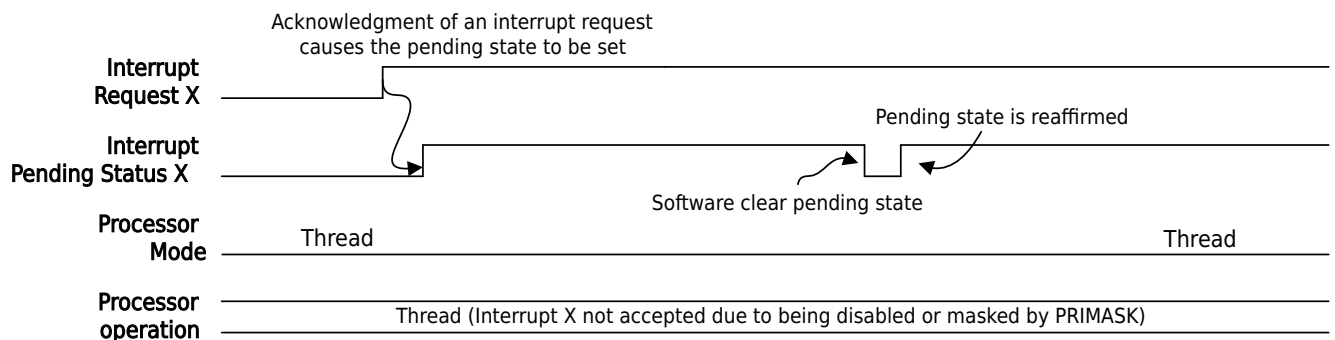


Figure 8-3 Interrupt Pending State is Cleared and Then Reasserted

If the interrupt request generated by the peripheral is not cleared when the exception is handled, the pending state will be activated again after the exception returns, so that the interrupt service routine will be executed again. The handling process is illustrated in the following figure:

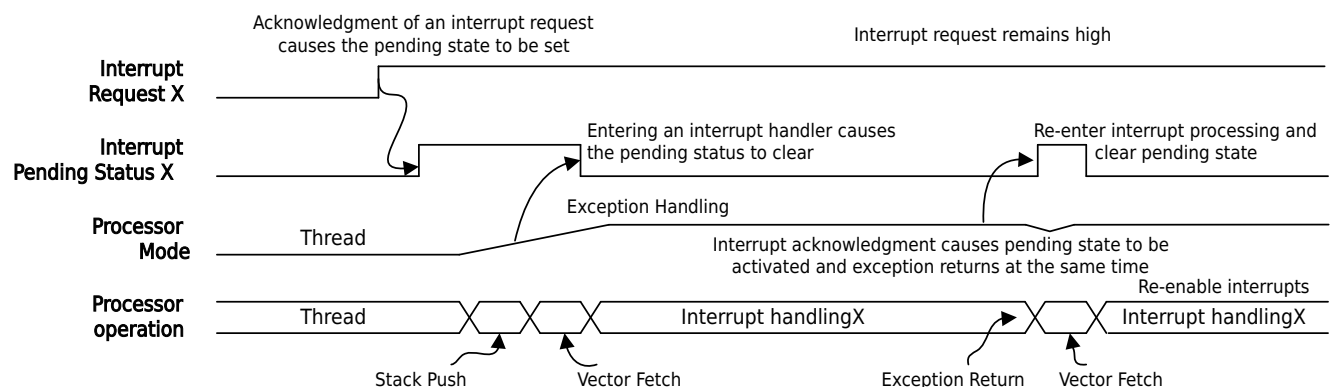


Figure 8-4 When the Interrupt Exits, If the Interrupt Request Remains High, It Will Cause the Interrupt Processing to Be Executed Again

If a peripheral interrupt request occurs during the execution of the interrupt service routine, the request will be regarded as a new interrupt request, and after this interrupt exits, the interrupt service program will be executed again. The handling process is illustrated in the following figure:

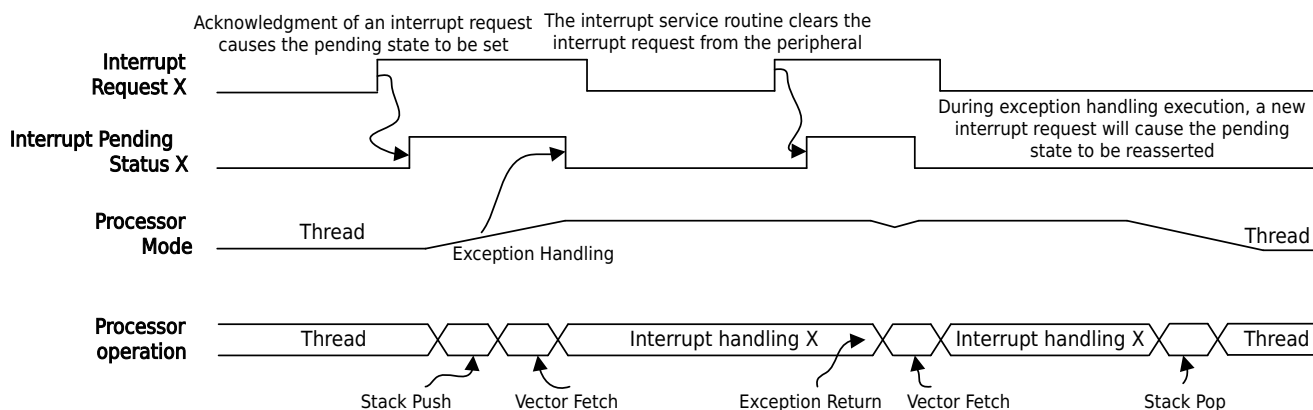


Figure 8-5 Interrupt Pending Interrupts Generated During Interrupt Processing Can Also Be Acknowledged

8.2.4 Interrupt Waiting

Typically, the interrupt waiting of the NVIC is 16 cycles. This wait time starts from the processor clock cycle when the interrupt is acknowledged, and ends when the interrupt handler starts executing. Computing interrupt waiting requires the following prerequisites:

- The interrupt is enabled and not masked by PRIMASK or any other currently executing exception handler.
- The memory system does not have any wait states. Bus transfers are used for interrupt processing, stack push, vector fetches, or instruction fetches at the start of interrupt processing. If the memory system needs to wait, the wait states generated when bus transfers occur may delay interrupts.

The following situations may cause different interrupt waiting:

- The end of the interrupt is chained. If another interrupt request is generated when the interrupt returns, the processor will skip the process of popping and pushing the stack, thus reducing the interrupt waiting time.
- Delayed arrival. If an interrupt occurs, another low-priority interrupt is being pushed to the stack. Due to the existence of the delayed arrival mechanism, the high-priority interrupt will be executed first, which will also reduce the waiting time of the high-priority interrupt.

8.2.5 Interrupt Source

Because the NVIC of the Cortex-M0+ processor supports up to 32 external interrupts, and in this system, there are 17 external interrupt sources, some external interrupts are multiplexed on the same NVIC interrupt input, and NMI (Non-Maskable Interrupt) did not use. The corresponding relationship between all external interrupt sources of this system and NVIC interrupt input is shown in the following table:

NVIC Interrupt Input	External Interrupt Source	Active Mode	Sleep mode	DeepSleep mode
Interrupt #0	GPIOA	✓	✓	✓
Interrupt #1	GPIOB	✓	✓	✓
Interrupt #2	-	-	-	-

NVIC Interrupt Input	External Interrupt Source	Active Mode	Sleep mode	DeepSleep mode
Interrupt #3	-	-	-	-
Interrupt #4	-	-	-	-
Interrupt #5	ATIM3	✓	✓	×
Interrupt #6	-	-	-	-
Interrupt #7	-	-	-	-
Interrupt #8	LPUART0	✓	✓	✓
Interrupt #9	LPUART1	✓	✓	✓
Interrupt #10	SPI	✓	✓	×
Interrupt #11	-	-	-	-
Interrupt #12	-	-	-	-
Interrupt #13	-	-	-	-
Interrupt #14	CTIM0	✓	✓	×
Interrupt #15	CTIM1	✓	✓	×
Interrupt #16	-	-	-	-
Interrupt #17	-	-	-	-
Interrupt #18	HSI2C	✓	✓	✓ ⁽²⁾
Interrupt #19	-	-	-	-
Interrupt #20	-	-	-	-
Interrupt #21	-	-	-	-
Interrupt #22	IWDT	✓	✓	✓
Interrupt #23	RTC	✓	✓	✓
Interrupt #24	ADC	✓	✓	×
Interrupt #25	-	-	-	-
Interrupt #26	VC0	✓	✓	✓
Interrupt #27	VC1	✓	✓	✓
Interrupt #28	LVD	✓	✓	✓
Interrupt #29	-	-	-	-
Interrupt #30	FLASH	✓	✓	×
Interrupt #31	CTRIM/CLKDET	✓	✓	✓

**Notice**

1. Since some module interrupts are multiplexed for the same IRQ interrupt source, when the CPU enters the interrupt operation, it must first determine which module generated the interrupt, and then perform the corresponding interrupt operation.
2. Only slave mode supports wakeup from DeepSleep.

8.2.6 Interrupt Structure Diagram

The interrupt structure diagram of this system is shown in the block diagram below. Please note the following:

- The respective interrupt enables of the peripheral interrupt sources are not marked in the figure, and only the logic block diagram of the interrupt signal after the peripheral interrupt is generated is included here.
- IRQ has more than one peripheral interrupt input multiplexing, and the interrupt flag bits of these peripherals must be read separately to determine which peripheral interrupt it is.
- If a peripheral interrupt source is asserted with a high-level signal, the corresponding bit in the NVIC interrupt pending state set register (NVIC_ISPR) will be set regardless of whether the interrupt is enabled in the NVIC interrupt enable set register (NVIC_ISER), indicating that an interrupt has been generated by the peripheral.
- Only when the interrupt enable register NVIC_ISER is set, the corresponding interrupt IRQ will respond to the processor and execute the corresponding interrupt program.
- In the interrupt program, the high-level interrupt signal of the peripheral interrupt source must be cleared, and the interrupt pending register NVIC_ISPR is automatically cleared by the hardware.
- The interrupt priority registers NVIC_IPR0-NVIC_IPR7 set the priority of 32 interrupt sources, 0b00 has the highest priority, and 0b11 has the lowest priority. When the priority is the same, the priority is determined by the interrupt number, and the smaller the number, the higher the priority.

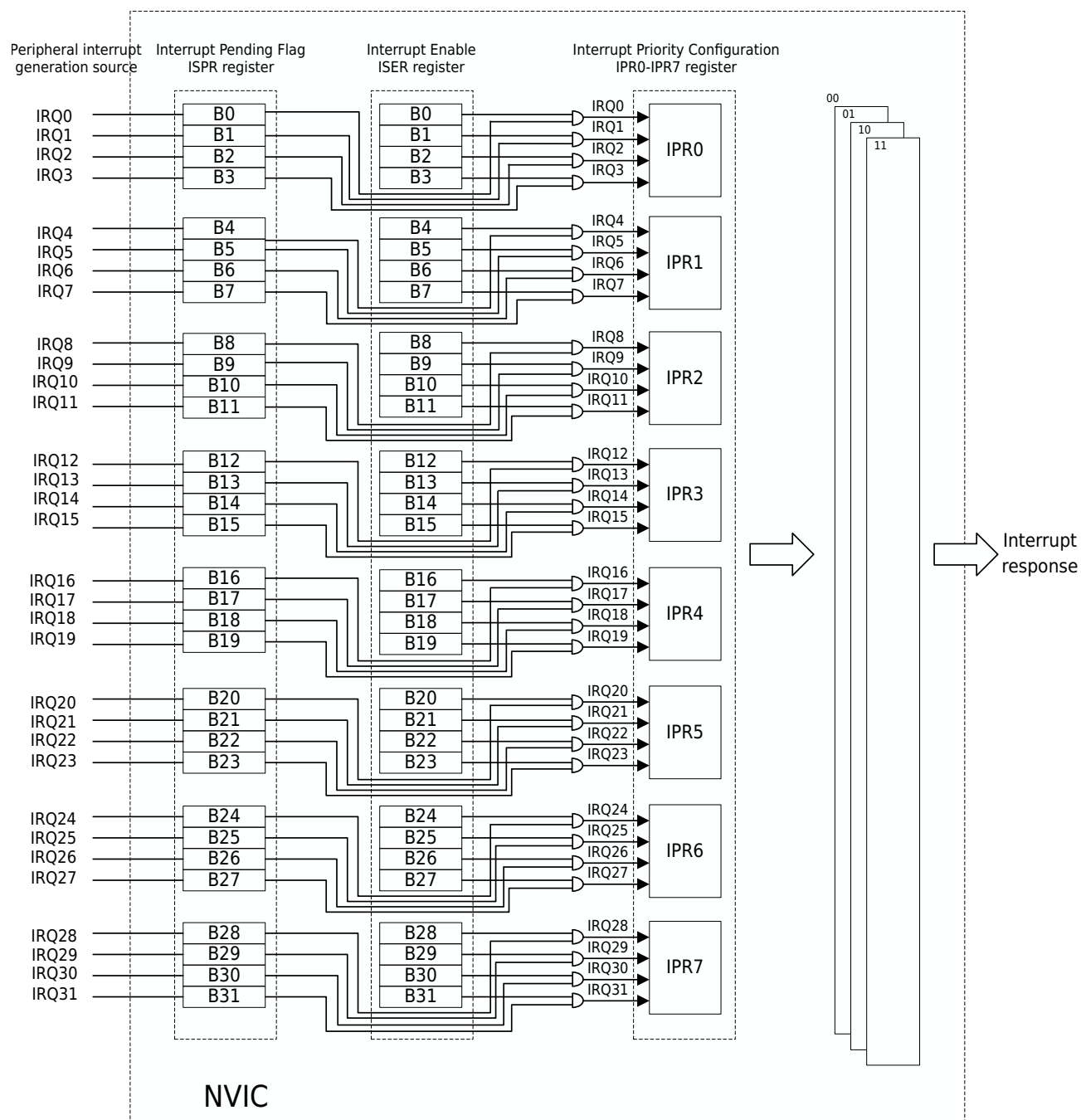


Figure 8-6 Interrupt Structure Diagram

8.3 Operation Example

8.3.1 External Interrupt Enable

Each peripheral module has its own interrupt enable register. When an interrupt operation is required, the peripheral's own interrupt enable must be enabled first. The operation of this enable bit is not discussed in this chapter, please refer to the chapter descriptions of the corresponding peripheral modules.

8.3.2 NVIC Interrupt Enable and Clear Enable

The Cortex-M0+ processor supports up to 32 interrupt sources, and each interrupt source corresponds to an interrupt enable bit and a clear-enable bit, forming a 32-bit interrupt enable set register (NVIC_ISER) and a 32-bit interrupt enable clear register (NVIC_ICER). If you want to enable a particular interrupt, set the corresponding bit in the NVIC_ISER register to 1. If you want to clear a particular interrupt, set the corresponding bit in the NVIC_ICER register to 1.

The interrupt enable and clear-enable operations mentioned here apply only to the NVIC processor. Whether an interrupt is generated by each peripheral is determined by the peripheral's own interrupt control registers and is unrelated to NVIC_ISER and NVIC_ICER.

8.3.3 NVIC Interrupt Pending and Clear Pending

If an interrupt occurs but cannot be handled immediately, the interrupt request will be pending. The pending state is kept in a register and will remain valid as long as the processor's current priority has not been lowered enough to handle the pending request and the pending state has not been cleared manually.

When the processor starts to enter the interrupt service, the hardware will automatically cause the clearing of the pending state.

You can access or modify the interrupt pending state by manipulating the interrupt pending state set register (NVIC_ISPR) and interrupt pending state clear register (NVIC_ICPR). The interrupt pending state register allows software to trigger interrupts.

8.3.4 NVIC Interrupt Priority

Setting the NVIC_IPR0-NVIC_IPR7 registers determines the priority of IRQ0-IRQ31. The programming of the interrupt priority register should be done before the interrupt is enabled, which is usually done at the beginning of the program. Changing the interrupt priority after the interrupt is enabled should be avoided, the results of this situation are unpredictable and are not supported by the Cortex-M0+ processor.

8.3.5 NVIC Interrupt Mask

Some time-sensitive applications need to disable all interrupts in a short period of time, which can be realized by using the interrupt mask register PRIMASK. Only 1 bit of the special register PRIMASK is valid, and it defaults to 0 after reset. When this register is 0, all interrupts and exceptions are enabled; when set to 1, only NMI (not supported by this system) and hardware error exceptions are enabled. In fact, when PRIMASK is set to 1, the current priority of the processor is reduced to 0 (highest programmable priority).

PRIMASK register can be programmed in a variety of ways. Using assembly language, the MSR instruction can be used to set and clear the PRIMASK register. If using C language and CMSIS device driver library, users can use the following functions to set and clear PRIMASK.

```
void_enable_irq(void); //Clear PRIMASK  
void_disable_irq(void); //Set PRIMASK
```

8.4 Register Description

8.4.1 Register List

Base address: 0xE000E000

Table 8-4 Interrupt Controller (NVIC) Register Offset Address

Offset	Register	Description	Width
0x100	NVIC_ISER	Interrupt Enable Set Register	32
0x180	NVIC_ICER	Interrupt Enable Clear Register	32
0x200	NVIC_ISPR	Interrupt Pending State Set Register	32
0x280	NVIC_ICPR	Interrupt Pending State Clear Register	32
0x400	NVIC_IPR0	Interrupt #0- Interrupt #3 Priority Registers	32
0x404	NVIC_IPR1	Interrupt #4- Interrupt #7 Priority Registers	32
0x408	NVIC_IPR2	Interrupt #8- Interrupt #11 Priority Registers	32
0x40C	NVIC_IPR3	Interrupt #12- Interrupt #15 Priority Registers	32
0x410	NVIC_IPR4	Interrupt #16- Interrupt #19 Priority Registers	32
0x414	NVIC_IPR5	Interrupt #20- Interrupt #23 Priority Registers	32
0x418	NVIC_IPR6	Interrupt #24- Interrupt #27 Priority Registers	32
0x41C	NVIC_IPR7	Interrupt #28- Interrupt #31 Priority Registers	32
-	PRIMASK	Interrupt Mask Special Register	32

8.4.2 Interrupt Enable Set Register (NVIC_ISER)

Offset	Bit Position																															
0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	SETENA																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	SETENA	Interrupt Enable Set Flag Set enable interrupt #0 to interrupt #31 ● bit 0: IRQ0 ● bit 1: IRQ1 ● bit 2: IRQ2 ● ... ● bit 31: IRQ31  Note Each bit: writing 1 is set, writing 0 has no effect.	RW

8.4.3 Interrupt Enable Clear Register (NVIC_ICER)

Offset	Bit Position																															
0x180	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CLRENA																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	CLRENA	Interrupt Clear-Enable Clear-enable interrupt #0 to interrupt #31 ● bit 0: IRQ0 ● bit 1: IRQ1 ● bit 2: IRQ2 ● ... ● bit 31: IRQ31  Note Each bit: write 1 to clear flag, write 0 has no effect. .	RW

8.4.4 Interrupt Pending State Set Register (NVIC_ISPR)

Offset	Bit Position																															
0x200	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	SETPEND																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	SETPEND	Interrupt Pending State Set Flag Set the pending state of interrupt #0 to interrupt #31 ● bit 0: IRQ0 ● bit 1: IRQ1 ● bit 2: IRQ2 ● ... ● bit 31: IRQ31  Note Each bit: write 1 is set, writing 0 has no effect.	RW

8.4.5 Interrupt Pending State Clear Register (NVIC_ICPR)

Offset	Bit Position																															
0x280	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CLRPEND																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	CLRPEND	Interrupt Pending State Clear Clear the pending state of interrupt #0 to interrupt #31 ● bit 0: IRQ0 ● bit 1: IRQ1 ● bit 2: IRQ2 ● ... ● bit 31: IRQ31  Note Each bit: write 1 to clear flag, write 0 has no effect.	RW

8.4.6 Interrupt Priority Register (NVIC_IPR0)

Offset	Bit Position																															
0x400	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	IPRO																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	IPR0	Priority of Interrupt #0 to Interrupt #3 The corresponding bits for each interrupt priority are as follows: ● [31:30]: Priority of interrupt #3 ● [23:22]: Priority of interrupt #2 ● [15:14]: Priority of interrupt #1 ● [7:6]: Priority of interrupt #0 In the above configuration, a setting of 0b00 indicates the highest priority, whereas 0b11 indicates the lowest priority. Since the other bits are unused, configuring them has no effect and read always returns 0.	RW

8.4.7 Interrupt Priority Register (NVIC_IPR1)

Offset	Bit Position																															
0x404	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	IPR1																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	IPR1	Priority of Interrupt #4 to Interrupt #7 The corresponding bits for each interrupt priority are as follows: ● [31:30]: Priority of interrupt #7 ● [23:22]: Priority of interrupt #6 ● [15:14]: Priority of interrupt #5 ● [7:6]: Priority of interrupt #4 In the above configuration, a setting of 0b00 indicates the highest priority, whereas 0b11 indicates the lowest priority. Since the other bits are unused, configuring them has no effect and read always returns 0.	RW

8.4.8 Interrupt Priority Register (NVIC_IPR2)

Offset	Bit Position																															
0x408	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	IPR2																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	IPR2	Priority of Interrupt #8 to Interrupt #11 The corresponding bits for each interrupt priority are as follows: ● [31:30]: Priority of interrupt #11 ● [23:22]: Priority of interrupt #10 ● [15:14]: Priority of interrupt #9 ● [7:6]: Priority of interrupt #8 In the above configuration, a setting of 0b00 indicates the highest priority, whereas 0b11 indicates the lowest priority. Since the other bits are unused, configuring them has no effect and read always returns 0.	RW

8.4.9 Interrupt Priority Register (NVIC_IPR3)

Offset	Bit Position																															
0x40C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	IPR3																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	IPR3	Priority of Interrupt #12 to Interrupt #15 The corresponding bits for each interrupt priority are as follows: ● [31:30]: Priority of interrupt #15 ● [23:22]: Priority of interrupt #14 ● [15:14]: Priority of interrupt #13 ● [7:6]: Priority of interrupt #12 In the above configuration, a setting of 0b00 indicates the highest priority, whereas 0b11 indicates the lowest priority. Since the other bits are unused, configuring them has no effect and read always returns 0.	RW

8.4.10 Interrupt Priority Register (NVIC_IPR4)

Offset	Bit Position																															
0x410	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	IPR4																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	IPR4	Priority of Interrupt #16 to Interrupt #19 The corresponding bits for each interrupt priority are as follows: ● [31:30]: Priority of interrupt #19 ● [23:22]: Priority of interrupt #18 ● [15:14]: Priority of interrupt #17 ● [7:6]: Priority of interrupt #16 In the above configuration, a setting of 0b00 indicates the highest priority, whereas 0b11 indicates the lowest priority. Since the other bits are unused, configuring them has no effect and read always returns 0.	RW

8.4.11 Interrupt Priority Register (NVIC_IPR5)

Offset	Bit Position																															
0x414	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	IPR5																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	IPR5	Priority of Interrupt #20 to Interrupt #23 The corresponding bits for each interrupt priority are as follows: ● [31:30]: Priority of interrupt #23 ● [23:22]: Priority of interrupt #22 ● [15:14]: Priority of interrupt #21 ● [7:6]: Priority of interrupt #20 In the above configuration, a setting of 0b00 indicates the highest priority, whereas 0b11 indicates the lowest priority. Since the other bits are unused, configuring them has no effect and read always returns 0.	RW

8.4.12 Interrupt Priority Register (NVIC_IPR6)

Offset	Bit Position																															
0x418	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	IPR6																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	IPR6	Priority of Interrupt #24 to Interrupt #27 The corresponding bits for each interrupt priority are as follows: ● [31:30]: Priority of interrupt #27 ● [23:22]: Priority of interrupt #26 ● [15:14]: Priority of interrupt #25 ● [7:6]: Priority of interrupt #24 In the above configuration, a setting of 0b00 indicates the highest priority, whereas 0b11 indicates the lowest priority. Since the other bits are unused, configuring them has no effect and read always returns 0.	RW

8.4.13 Interrupt Priority Register (NVIC_IPR7)

Offset	Bit Position																															
0x41C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	IPR7																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	IPR7	Priority of Interrupt #28 to Interrupt #31 The corresponding bits for each interrupt priority are as follows: ● [31:30]: Priority of interrupt #31 ● [23:22]: Priority of interrupt #30 ● [15:14]: Priority of interrupt #29 ● [7:6]: Priority of interrupt #28 In the above configuration, a setting of 0b00 indicates the highest priority, whereas 0b11 indicates the lowest priority. Since the other bits are unused, configuring them has no effect and read always returns 0.	RW

8.4.14 Interrupt Mask Special Register (PRIMASK)

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															PRIMASK
Access																																RW

Bit	Name	Functional Description	Access
31:1	Reserved	-	-
0	PRIMASK	Interrupt mask flag Typically used in time-critical application scenarios. The special register can be accessed through MSR and MRS special register operation instructions, and can also be accessed by changing the processor state instruction CPS. 0b0: All interrupts and exceptions are in the enabled state. 0b1: Only the NMI and hardware error exceptions are enabled.	RW

9 SYSTICK

9.1 Introduction

If the OS wants to support multitasking, it needs to perform context switching periodically, so hardware resources such as timers are needed to interrupt program execution. When the timer interrupt is generated, the processor will perform OS task scheduling in exception handling, and will also perform OS maintenance work at the same time. The Cortex-M0+ processor includes a simple timer called SysTick, which is used to generate periodic interrupt requests.

SysTick is a 24-bit timer and counts down. After the count of the timer is reduced to 0, a programmable value will be reloaded, and a SysTick exception (exception number 15) will be generated at the same time. This abnormal event will cause the execution of SysTick exception handling, which is a part of the OS.

For systems that do not require an OS, the SysTick timer can also be used for other purposes, such as timing, time-counting, or providing an interrupt source for tasks that need to be executed periodically. The generation of SysTick exception is controllable. If the exception is disabled, the SysTick timer can still be used by polling, such as checking the current count value or polling the count flag.

9.2 Operation Example

9.2.1 Set SysTick

Context

Since the reload value and current value of the SysTick timer are undefined at reset, in order to prevent abnormal results, the configuration of SysTick needs to follow a certain process.

Procedure

- Step1.** Configure SysTick->CTRL.ENABLE to 0, disable SysTick.
 - Step2.** Configure SysTick->CTRL.CLKSOURCE to select the clock source of SysTick.
 - Step3.** Configure SysTick->LOAD, select the overflow period of SysTick.
 - Step4.** Writing any value to SysTick->VAL, clear both SysTick->VAL and the SysTick->CTRL.COUNT-FLAG.
 - Step5.** Configure SysTick->CTRL.TICKINT to 1, enable SysTick interrupt.
 - Step6.** Configure SysTick->CTRL.ENABLE to 1, enable SysTick.
 - Step7.** Read SysTick->CTRL in the interrupt service routine to clear the overflow flag.
- The SysTick overflow period is SysTick->LOAD + 1. The configuration example is as follows:

Clock Source	SysTick->LOAD	Overflow Period
HCLK 4MHz	3999	1ms
XTL 32.768kHz	327	10.01ms

9.3 Register Description

9.3.1 Register List

Base address: 0xE000E010

Table 9-2 SysTick Timer Register Offset

Offset	Register	CMSIS Symbol	Description	Width
0x00	SYS_CSR	SysTick->CTRL	SysTick Control and Status Register	32
0x04	SYS_RVR	SysTick->LOAD	SysTick Reload Register	32
0x08	SYS_CVR	SysTick->VAL	SysTick Current Value Register	32
0x0C	SYS_CALIB	SysTick->CALIB	SysTick Calibration Value Register	32

9.3.2 SysTick Control and Status Register (CTRL)

Offset	Bit Position																																	
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved															COUNTFLAG	Reserved															CLKSOURCE	TICKINT	ENABLE
Access																RO																RW	RW	RW

Bit	Name	Functional Description	Access
31:17	Reserved	-	-
16	COUNTFLAG	SysTick timer overflow flag Reading this register can clear COUNTFLAG flag 0b0: SysTick timer has not overflowed 0b1: SysTick timer has underflowed	RO
15:3	Reserved	-	-
2	CLKSOURCE	SysTick clock source selection 0b0: Use reference clock (XTL) 0b1: Use core clock (HCLK)  Note When the reference clock is selected as the SysTick clock source, the XTL clock source must be enabled synchronously.	RW
1	TICKINT	SysTick interrupt enable 0b0: Disable 0b1: Enable	RW
0	ENABLE	SysTick timer enable 0b0: Disable 0b1: Enable	RW

9.3.3 SysTick Reload Register (LOAD)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved								RELOAD																							
Access									RW																							

Bit	Name	Functional Description	Access
31:24	Reserved	-	-
23:0	RELOAD	SysTick timer reload value	RW

9.3.4 SysTick Current Value Register (VAL)

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved								CURRENT																							
Access									RW																							

Bit	Name	Functional Description	Access
31:24	Reserved	-	-
23:0	CURRENT	Current count value of the SysTick timer Read the register to obtain the current count value of the SysTick timer. Writing any value to the register will clear both the register itself and the COUNTFLAG.	RW

9.3.5 SysTick Calibration Value Register (CALIB)

Offset	Bit Position																																	
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x40000148																																	
Name	NOREF	SKEW	Reserved								TENMS																							
Access	RO	RO									RO																							

Bit	Name	Functional Description	Access
31	NOREF	Whether the SysTick count clock supports the reference clock flag 0b0: Support reference clock (XTL) 0b1: Does not support reference clock (XTL)  Note When this bit is set to 1, CTRL.CLKSOURCE must be set to 1.	RO
30	SKEW	TENMS accuracy indication 0b0: TENMS value represents an exact 10ms 0b1: TENMS value represents an approximate 10ms	RO
29:24	Reserved	-	-
23:0	TENMS	10 millisecond calibration value	RO

10 CTRIM

10.1 Introduction

When the module works in calibration mode, it automatically performs real-time calibration of the RC48M/RCL output frequency, making the accuracy of the RC48M/RCL output frequency is no longer affected by environmental variations. When the module works in timer mode, it has general timing function and can still work normally under DeepSleep mode.

10.2 Features

The clock calibration supports the following features:

- Calibration mode: Supports RC48M/RCL automatic real-time calibration/single calibration
- Calibration mode: Multiple calibration steps, quickly approaching the target
- Timing mode: Can work in DeepSleep mode
- Timing mode: Supports automatic wake-up with a period range of 60μs to 65536s
- Timing mode: multiple clock signals available for selection
- Timing mode: Toggle output

10.3 Functional Description

10.3.1 RC48M Clock Calibration Mode

When CTRIM_CR1.MD is configured as 0b00, the module operates in RC48M calibration mode. This mode enables automatic real-time calibration of the output frequency of RC48M, ensuring the accuracy of RC48M remains unaffected by environmental variations. During calibrating RC48M, the module requires a precise low-frequency clock signal, which can be either the on-chip PCLK or an external input via the ETR pin.



Note

The TCLK frequency must be greater than twice the RCLK frequency.

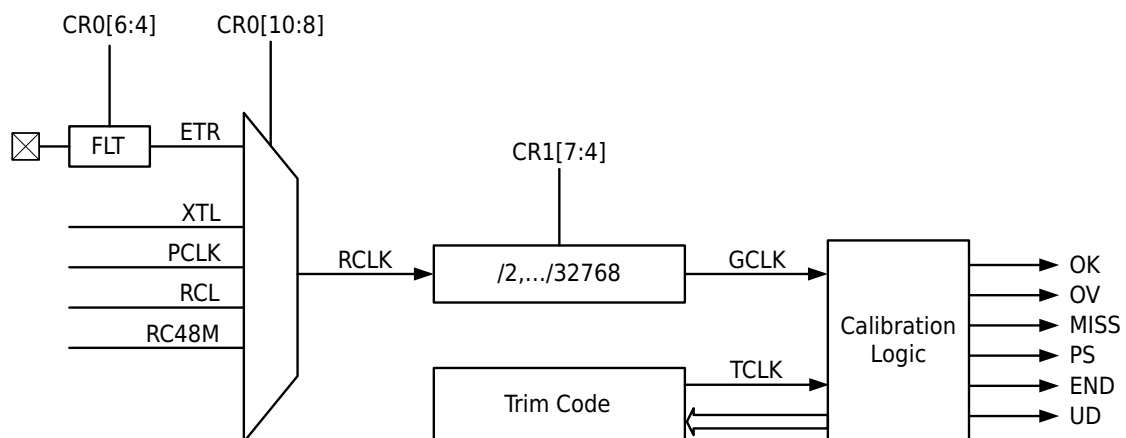


Figure 10-1 RC48M Clock Calibration Operational Block Diagram

When automatic calibration begins, the error counter performs down-counting on the TCLK signal (RC48M oscillation frequency) starting from CTRIM_ARR at each GCLK cycle, then switches to up-counting upon reaching zero. If the error counter value within a GCLK cycle is less than the CTRIM_FLIM

register value, CTRIM_ISR.OK is set to flag that the output frequency of RC48M meets expectations. If the error counter value within a GCLK cycle is greater than the CTRIM_FLIM register value, the TrimCode of RC48M is automatically adjusted for the next counting cycle. The auto-adjustment rules are: If counting direction is down-counting, increase the TrimCode of RC48M by CTRIM_CR0.STEP; If counting direction is up-counting, decrease the TrimCode of RC48M by CTRIM_CR0.STEP. The calibration logic automatically reduces the value of CTRIM_CR0.STEP to achieve gradual convergence of the output frequency of RC48M toward the target value.

UD event is generated when the error counter count decrements to zero; MISS event is triggered when the error counter count increments to CTRIM_ARR, stopping further counting. Whenever the rising edge of GCLK arrives, the error counter starts counting again.

● RC48M Single Calibration

Configure appropriate STEP, initial value, and FLIM using the same method as above. Set CTRIM_CR1.OST=1 to perform single calibration of RC48M, which completes when STEP becomes 1.

10.3.2 RCL Clock Calibration Mode

When CTRIM_CR1.MD is set to 0b01, the module operates in RCL calibration mode. This mode can automatically calibrate the output frequency of RCL in real time, so that the accuracy of RCL output frequency will no longer be affected by environmental changes. When calibrating RCL, the module requires a precise high-frequency clock signal, which can be either the on-chip PCLK or an external input via the ETR pin.



Note

The TCLK frequency must be greater than twice the RCLK frequency.

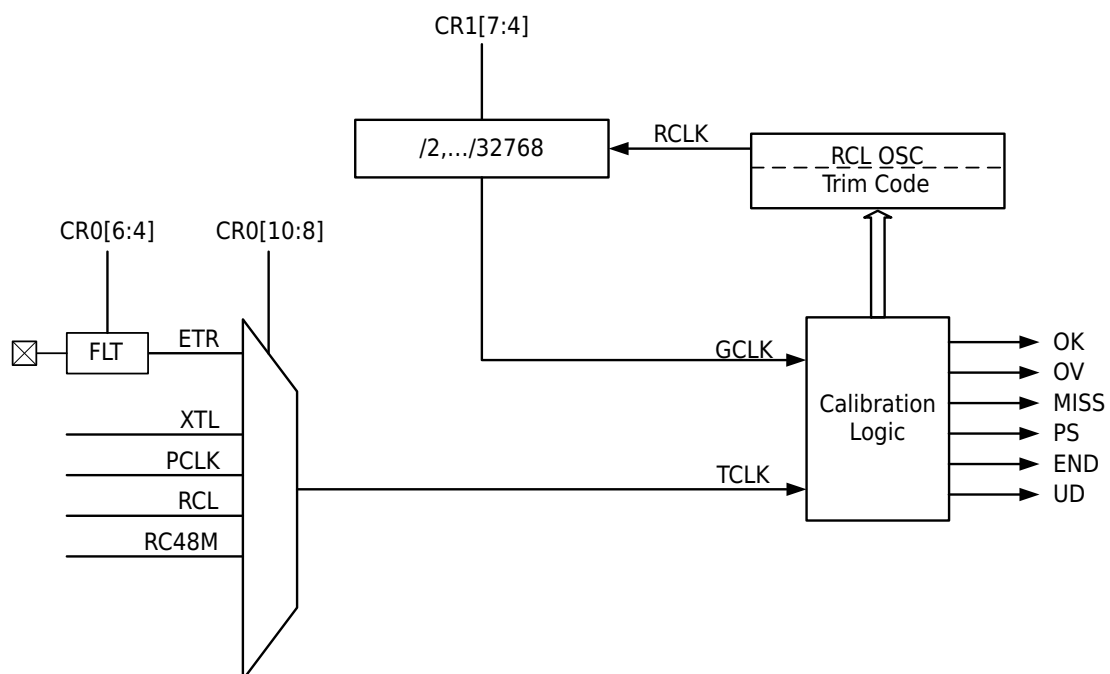


Figure 10-2 RCL Clock Calibration Operational Block Diagram

When automatic calibration begins, the error counter performs down-counting on the TCLK clock signal starting from CTRIM_ARR at each GCLK (the oscillation frequency of RCL after frequency division) cycle, then switches to up-counting upon reaching zero. If the error counter value within a GCLK cycle is less than the value of CTRIM_FLIM register, CTRIM_ISR.OK is set to flag that the RCL output frequency meets

expectations. If the error counter value within a GCLK cycle is greater than the value of CTRIM_FLIM register, the TrimCode of RCL is automatically adjusted for the next counting cycle. The auto-adjustment rules are: If counting direction is down-counting, decrease TrimCode of RCL by CTRIM_CR0.STEP; If counting direction is up-counting, increase TrimCode of RCL by CTRIM_CR0.STEP. The calibration logic automatically reduces the value of CTRIM_CR0.STEP to achieve gradual convergence of the RCL output frequency toward the target value.

UD event is generated when the error counter decrements to zero; stopping further counting and MISS event is triggered when the counter increments to CTRIM_ARR. Whenever the rising edge of GCLK arrives, the error counter starts counting again.

10.3.3 Low Power Timer Mode

When CTRIM_CR1.MD is set to 0b11, the module operates in general low power timer mode. In this mode, CTRIM_CR0.SRC must be configured to a clock available in low power mode. In this mode, CTRIM has timing and interrupt functions of a general timer. If the timer clock is RCL, it operates normally in DeepSleep mode, supporting automatic wake-up with a period range of 60μs-65536s.

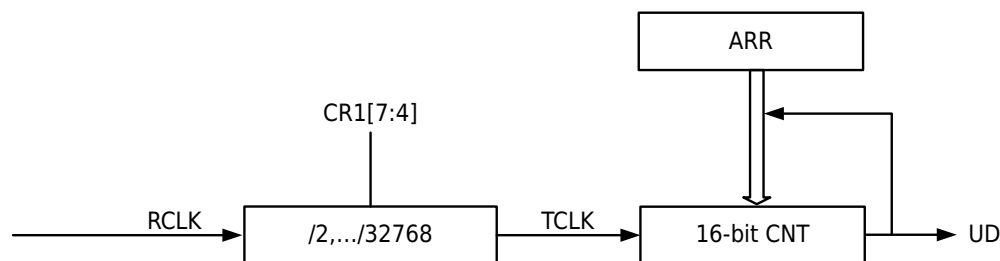


Figure 10-3 General Low Power Timer Scenario Operational Block Diagram

The timer counts down from ARR, and when it reaches 0, an underflow is generated. If the interrupt is enabled, underflow interrupt is generated, the counting period is $ARR + 1$.

TOG output level flips once every time the timer overflows, the TOG signal can serve as the trigger source for the ADC; the user can output the TOG signal to the pin through the port multiplexing function of the GPIO to drive an external circuit.

10.4 Operation Example

10.4.1 RC48M Clock Automatic Calibration Example

Procedure

- Step1.** Refer to the "General-Purpose I/O (GPIO)" chapter for related description of pin multiplexing configuration, set the relevant pin function to CTRIM.ETR, and supply it with a 2kHz square wave.
- Step2.** Set CTRIM_CR1.MD to 0 to select RC48M clock calibration mode.
- Step3.** Set CTRIM_CR0.SRC to 0, and the external ETR pin input signal is used as RCLK.
- Step4.** Set CTRIM_CR1.PRS to 1, perform 2 frequency division on RCLK, and GCLK frequency is 1kHz.
- Step5.** Configure CTRIM_ARR to 47999, and the target count value in a GCLK count cycle is 48000.



Note

The ETR pin input signal is a 2kHz square wave, the period of GCLK is 1ms. The theoretical count value of the error counter for the RC48M output frequency within 1ms should be 48000.

- Step6.** Set CTRIM_FLIM to 240, select RC48M calibration accuracy of 0.5%.

$$48000 \times 0.005 = 240.$$

- Step7.** Set CTRIM_CR0.STEP to 0, and select the initial calibration step to be 16.
- Step8.** Set CTRIM_CR1.AUTO to 1 to enable automatic calibration.
- Step9.** Set CTRIM_CR1.EN to 1 to start clock calibration.
- Step10.** Wait for CTRIM_ISR.END to be 1, and the automatic calibration is completed.

10.4.2 RCL Clock Automatic Calibration Example

Procedure

- Step1.** Refer to the "General-Purpose I/O (GPIO)" chapter for related description of pin multiplexing configuration, set the relevant pin function to CTRIM.ETR, and supply it with a 1MHz square wave.
- Step2.** Set CTRIM_CR1.MD to 1, select RCL clock calibration mode.
- Step3.** Set CTRIM_CR0.SRC to 0, use the external ETR pin input signal as TCLK.
- Step4.** Set CTRIM_CR1.PRS to 0x0A, perform 1024 frequency division on RCLK (RCL output frequency is), and the target frequency of GCLK is 32Hz.
- Step5.** Configure CTRIM_ARR to 31249, and the target count value in a GCLK count cycle is 31250.



Note

The target period of GCLK is 31.25ms, during which the theoretical count value of the error counter for TCLK(1MHz) should be 31250.

- Step6.** Set CTRIM_FLIM to 156, select RCL correction accuracy to 0.5%.
 $31250 \times 0.005 = 156.$
- Step7.** Set CTRIM_CR0.STEP to 0, and select the initial calibration step to be 16.
- Step8.** Set CTRIM_CR1.AUTO to 1 to enable automatic calibration.
- Step9.** Set CTRIM_CR1.EN to 1 to start clock calibration.
- Step10.** Wait for CTRIM_ISR.END to be 1, and the automatic calibration is completed.

10.5 Register Description

10.5.1 Register List

Base address: 0x40009000

Table 10-1 Clock Trim Module (CTRM) Register Offset

Offset	Register	Description	Width
0x00	CTRM_ARR	Autoload Register	32
0x04	CTRM_CNT	Error Counter Register	32
0x0C	CTRM_CR0	Control Register 0	32
0x10	CTRM_CR1	Control Register 1	32
0x14	CTRM_IER	Interrupt Enable Register	32
0x18	CTRM_ISR	Interrupt and Status Register	32
0x1C	CTRM_ICR	Interrupt Flag Clear Register	32
0x20	CTRM_FCAP	Error Result Register, Counter Value at the End of the GCLK Cycle	32
0x24	CTRM_TVAL	TRIM Result Register	32
0x28	CTRM_FLIM	Error Upper Limit Register	32

10.5.2 Autoload Register (CTRIM_ARR)

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																ARR															
Access																																

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	ARR	Error counter reload value	RW

10.5.3 Error Counter Register (CTRIM_CNT)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CNT															
Access																																

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	CNT	Error counter count value	RO

10.5.4 Control Register 0 (CTRIM_CR0)

**Note**

This register can only be changed when CTRIM_CR1.EN is 0.

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				SRC		Reserved	ETRFLT		Reserved	STEP					
Access																					RW			RW			RW					

Bit	Name	Functional Description	Access
31:11	Reserved	-	-
10:8	SRC	Clock source selection ● 0b000: ETR ● 0b001: Setting prohibited ● 0b010: XTL ● 0b011: PCLK ● 0b100: RCL ● 0b101: RC48M ● 0b110: Setting prohibited ● 0b111: Setting prohibited	RW
7	Reserved	-	-
6:4	ETRFLT	External input pin (ETR) filter configuration ● 0b000: No filter ● 0b001: $F_{\text{sample}} = \text{PCLK}, N=2$ ● 0b010: $F_{\text{sample}} = \text{PCLK}, N=4$ ● 0b011: $F_{\text{sample}} = \text{PCLK}, N=6$ ● 0b100: $F_{\text{sample}} = \text{PCLK}/4, N=4$ ● 0b101: $F_{\text{sample}} = \text{PCLK}/4, N=6$ ● 0b110: $F_{\text{sample}} = \text{PCLK}/8, N=4$ ● 0b111: $F_{\text{sample}} = \text{PCLK}/8, N=6$	RW
3	Reserved	-	-

Bit	Name	Functional Description	Access
2:0	STEP	TRIM initial step configuration ● 0b000: 0x10 ● 0b001: 0x02 ● 0b010: 0x08 ● 0b011: 0x20 ● 0b100: 0x80 ● 0b101: 0x100 ● 0b110: 0x200 ● 0b111: 0x400	RW

10.5.5 Control Register 1 (CTRIM_CR1)

**Note**

This register can only be changed when CTRIM_CR1.EN is 0.

Offset	Bit Position																																	
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																													AUTO		MD		EN
Access																								RW		RW				RW		RW		RW

Bit	Name	Functional Description	Access
31:9	Reserved	-	-
8	OST	Calibration mode configuration 0b0: Real-time calibration mode, the TRIM process continues 0b1: Single calibration mode, end when STEP becomes 1	RW
7:4	PRS	Reference clock prescaler configuration 0b0000: Setting prohibited 0b0001: Divide by 2 0b0010: Divide by 4 0b0011: Divide by 8 0b0100: Divide by 16 0b0101: Divide by 32 0b0110: Divide by 64 0b0111: Divide by 128 0b1000: Divide by 256 0b1001: Divide by 512 0b1010: Divide by 1024 0b1011: Divide by 2048 0b1100: Divide by 4096 0b1101: Divide by 8192 0b1110: Divide by 16384 0b1111: Divide by 32768	RW
3	AUTO	RC48M/RCL automatic TRIM enable configuration 0b0: Disable 0b1: Enable Note In real-time calibration mode, this bit is prohibited from being configured as 0.	RW

Bit	Name	Functional Description	Access
2:1	MD	Operating mode configuration ● 0b00: RC48M calibration mode, CR0.SRC needs to select a low-speed precision clock ● 0b01: RCL calibration mode, CR0.SRC needs to select a high-speed precision clock ● 0b10: Setting prohibited ● 0b11: Timer mode	RW
0	EN	Timer enable control ● 0b0: Disable ● 0b1: Enable	RW

10.5.6 Interrupt Enable Register (CTRIM_IER)

Offset	Bit Position																																	
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																																	
Access																											RW		RW		RW		RW	

Bit	Name	Functional Description	Access
31:6	Reserved	-	-
5	OK	Auto Trim results in range interrupt enable 0b0: Disable 0b1: Enable	RW
4	OV	TrimCode overflow interrupt enable 0b0: Disable 0b1: Enable	RW
3	MISS	Count failure interrupt enable 0b0: Disable 0b1: Enable	RW
2	PS	Error counter completion interrupt enable 0b0: Disable 0b1: Enable	RW
1	END	Auto Trim end interrupt enable 0b0: Disable 0b1: Enable	RW
0	UD	Error counter underflow interrupt enable 0b0: Disable 0b1: Enable	RW

10.5.7 Interrupt and Status Register (CTRIM_ISR)

Offset	Bit Position																																							
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
Reset	0x00000000																																							
Name	Reserved																										DIR		OK		OV		MISS		PS		END		UD	
Access																											RO		RO		RO		RO		RO		RO		RO	

Bit	Name	Functional Description	Access
31:7	Reserved	-	-
6	DIR	Error counter counting direction 0b0: Counting up, RC48M output frequency is higher than the target frequency, RCL output frequency is lower than the target frequency 0b1: Counting down, RC48M output frequency is lower than the target frequency, RCL output frequency is higher than the target frequency	RO
5	OK	Trim accuracy flag 0b0: Accuracy does not meet FLIM requirements when Trim is completed 0b1: Accuracy meets FLIM requirements when Trim is completed	RO
4	OV	TrimCode overflow flag 0b0: The TrimCode of RC48M/RCL is not 0x000 or 0x1FF 0b1: The TrimCode of RC48M/RCL is adjusted to 0x000 or 0x1FF	RO
3	MISS	Counting failure flag 0b0: The error counter does not count up to ARR during one cycle of GCLK 0b1: The error counter counts up to ARR during one cycle of GCLK	RO
2	PS	Error counter completion flag 0b0: Error counter is counting 0b1: Error counter has completed one counting cycle and stopped	RO
1	END	Auto Trim end flag 0b0: Auto Trim is in progress 0b1: Auto Trim is completed	RO
0	UD	Error counter underflow flag 0b0: The error counter has not counted to 0 0b1: The error counter has counted to 0	RO

10.5.8 Interrupt Status Flag Register (CTRIM_ICR)

Offset	Bit Position																																							
0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
Reset	0x0000003F																																							
Name	Reserved																												OK		OV		MISS		PS		END		UD	
Access																																								

Bit	Name	Functional Description	Access
31:6	Reserved	-	-
5	OK	Clear the calibration success OK flag Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
4	OV	Clear TrimCode overflow flag Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
3	MISS	Clear count failure flag Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
2	PS	Clear error counter completion flag Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
1	END	Clear auto Trim end flag Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
0	UD	Clear error counter underflow flag Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

10.5.9 Error Result Register (CTRIM_FCAP)

Offset	Bit Position																															
0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																FCAP															
Access																																

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	FCAP	Frequency error capture value	RO

10.5.10 TRIM Result Register (CTRIM_TVAL)

Offset	Bit Position																															
0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000100																															
Name	Reserved																TVAL															
Access																	RO															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	TVAL	Auto TRIM result  Note 1. After auto TRIM completes, write the TVAL value to the corresponding TRIM register. 2. If reading during TRIM operation, two identical reads are required for validity.	RO

10.5.11 Error Upper Limit Register (CTRIM_FLIM)

Offset	Bit Position																															
0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000010																															
Name	Reserved																				FLIM											
Access																					RW											

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11:0	FLIM	Error tolerance	RW

11 CTIM

11.1 Introduction

CTIM is a composite timer, which can be configured as a general timer GTIM with four groups of comparison capture functions, or as a group (three) of basic timers BTIM with timing functions only. Controlled by SYSCTRL1.CTIMERx_FUN_SEL. When this bit is 1, BTIM is active; when 0, GTIM is active. Each CTIM in this product can be independently and flexibly configured as either GTIM or BTIM.

- BTIM is a group of timing counters with basic timing functions, each group contains three 16-bit timers with reload function. This product can be configured with up to six basic timers: BTIM0-5.
- GTIM is a 16-bit timer with 4-way compare capture function. This product can be configured with up to two basic timers: BTIM0-1.

Table 11-1 Comparison Table of Pin Function and Timer Function

CTIM (x=0, 1)	GTIM (x=0, 1)	BTIM (y=0, 3)
CTIM _x _ETR	GTIM _x _ETR	BTIM _y _ETR
CTIM _x _TOG	GTIM _x _TOG	BTIM _{y+0} _TOG
CTIM _x _TOGN	GTIM _x _TOGN	BTIM _{y+0} _TOGN
CTIM _x _CH0	GTIM _x _CH0	BTIM _{y+1} _TOG
CTIM _x _CH1	GTIM _x _CH1	BTIM _{y+1} _TOGN
CTIM _x _CH2	GTIM _x _CH2	BTIM _{y+2} _TOG
CTIM _x _CH3	GTIM _x _CH3	BTIM _{y+2} _TOGN



Note

- BTIM0/1/2 share CTIM0_ETR; BTIM3/4/5 share CTIM1_ETR.
- When x=0, y=0; when x=1, y=3.

11.2 Features

11.2.1 BTIM Features

- Three 16-bit timers
- 2 toggle outputs per timer
- Single pulse mode
- External input counting function
- Gating function
- External trigger function
- Cascade function

11.2.2 GTIM Features

- 16-bit timer
- Two toggle outputs
- Single pulse mode
- External input counting function
- Gating function
- External trigger function
- Four independent PWM outputs, adjustment range 0%-100%
- 4-way independent capture
- Cascade function

11.3 Functional Description

11.3.1 BTIM Functional Description

11.3.1.1 Function Block Diagram

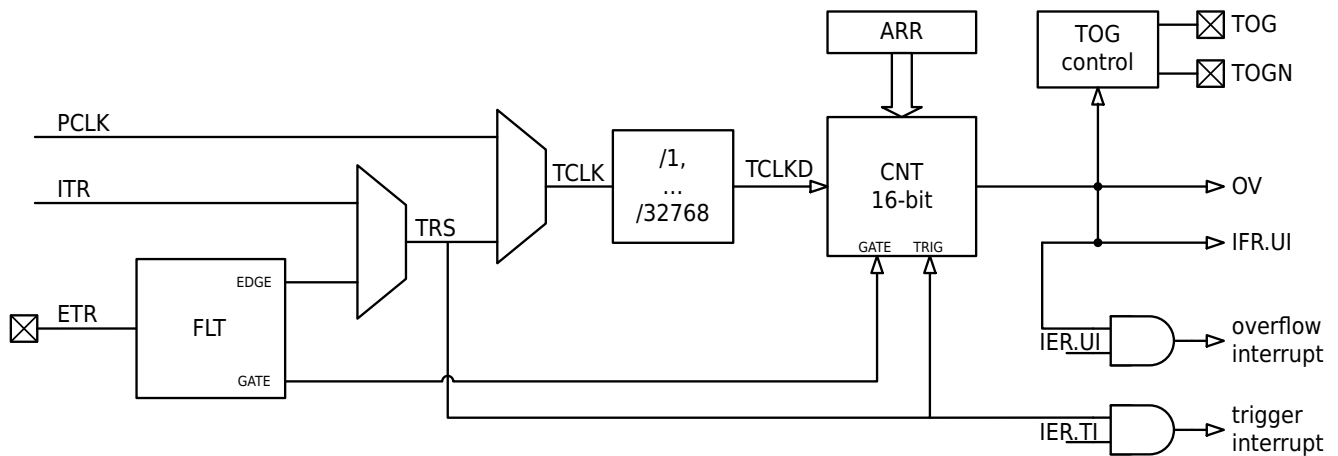


Figure 11-1 BTIM Block Diagram



Note

- A group of 3 BTIM share one ETR.
- For more information on the ITR signal sources, see the section "[Timer Cascade](#)".

11.3.1.2 Filter Unit

ETR signal input by the external pin is filtered, polarity selected, and edge detected, the gating signal and edge signal are output for use by the counting unit and the triggering unit. Since the ETR signal uses PCLK for synchronous filtering, the frequency of the input ETR signal should be lower than 1/2 of the frequency of PCLK.

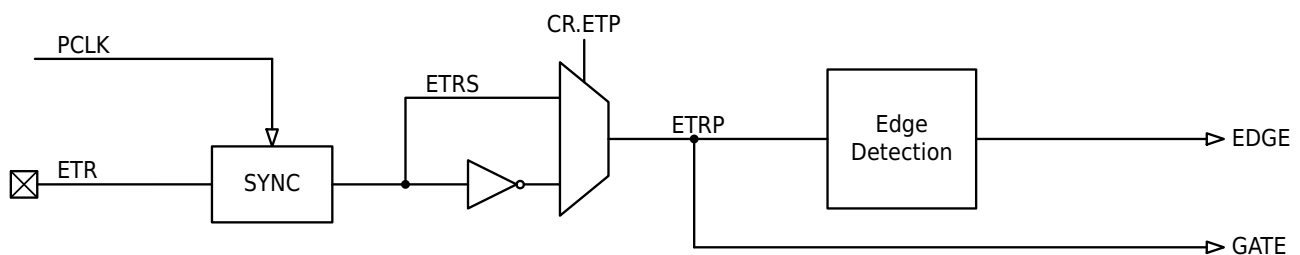


Figure 11-2 BTIM Filter Unit Diagram

11.3.1.3 Counting Unit

On each rising edge of the TCLKD signal, the internal counter CNT is automatically incremented by 1. Counting to ARR, an overflow signal OV (width of one PCLK) is generated, and the internal count value is reset to 0 and counting up starts again. The counting range of the counter is 0 to ARR, and the counting period of the counting unit is $ARR+1$.

The timer supports 2 counting modes, One-Shot mode and continuous counting mode.

- When BTIMx_CR.OST is set to 1, the timer works in One-Shot mode; the timer stops counting immediately after an overflow occurs, and the hardware automatically clears CEN.

- When BTIMx_CR.OST is set to 0, the timer works in continuous counting mode; as long as BTIMx_CR.CEN is 1, the timer will keep counting.

Every time there is an overflow, BTIMx_IFR.UI will be set to 1 by hardware. If the corresponding interrupt is enabled, an overflow interrupt can be generated. The user should clear the overflow flag in the interrupt service routine.

If BTIMx_CR.TOGEN is set to 1, the TOG/TOGN output toggles once on each overflow. The TOG/TOGN signal can be output to a pin via GPIO multiplexing function to drive external circuits.

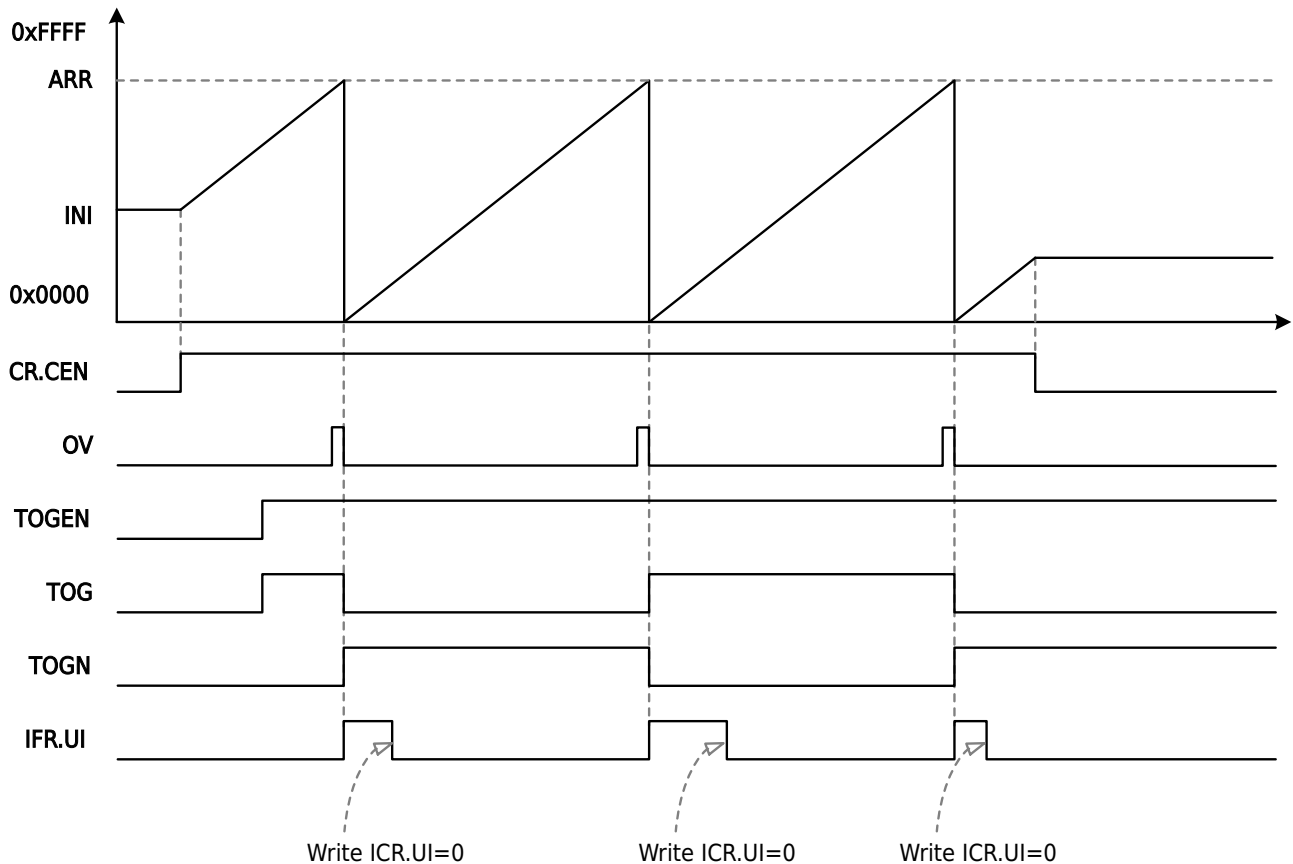


Figure 11-3 BTIM Counter Waveform

11.3.1.4 Timer Mode

When setting BTIMx_CR.MD to 0x0, BTIMx works in timer mode. In this mode, the counting clock is the clock TCLKD after PCLK frequency division. When BTIMx_CR.CEN is set to 1, the timer starts counting up from the initial value, and overflows after counting to ARR and restarts counting from 0.

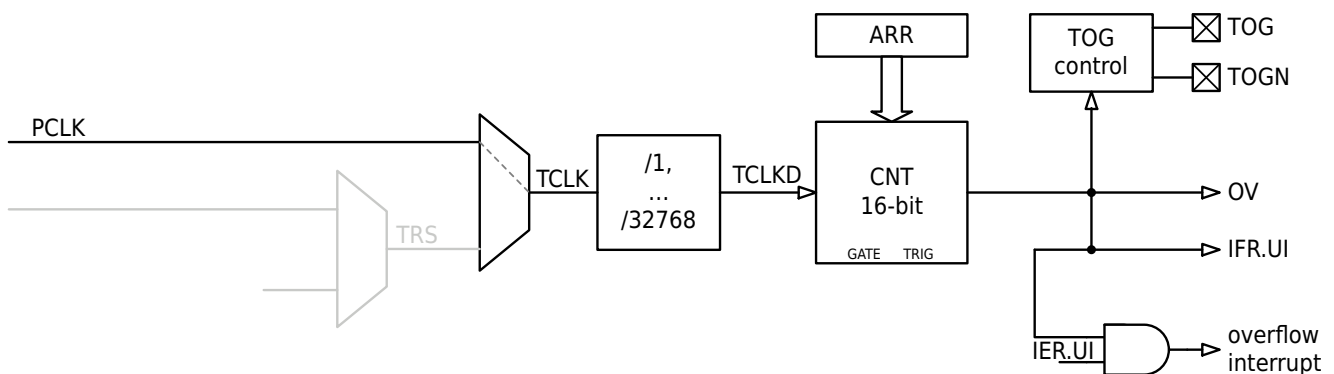


Figure 11-4 BTIM Timing Mode Block Diagram

11.3.1.5 Counter Mode

When setting BTIMx_CR.MD to 0x1, BTIMx works in counter mode. In this mode, the counting clock is the signal input by the ITR or ETR pin after frequency division. When BTIMx_CR.CEN is set to 1, the counter starts counting up from the initial value, and overflows after counting to ARR, and then starts counting from 0 again.

The counting clock source can be selected as ITR or ETR through BTIMx_CR.TRS; through BTIMx_CR.ETP, the rising edge or falling edge of the ETR pin input signal can be selected to be counted.

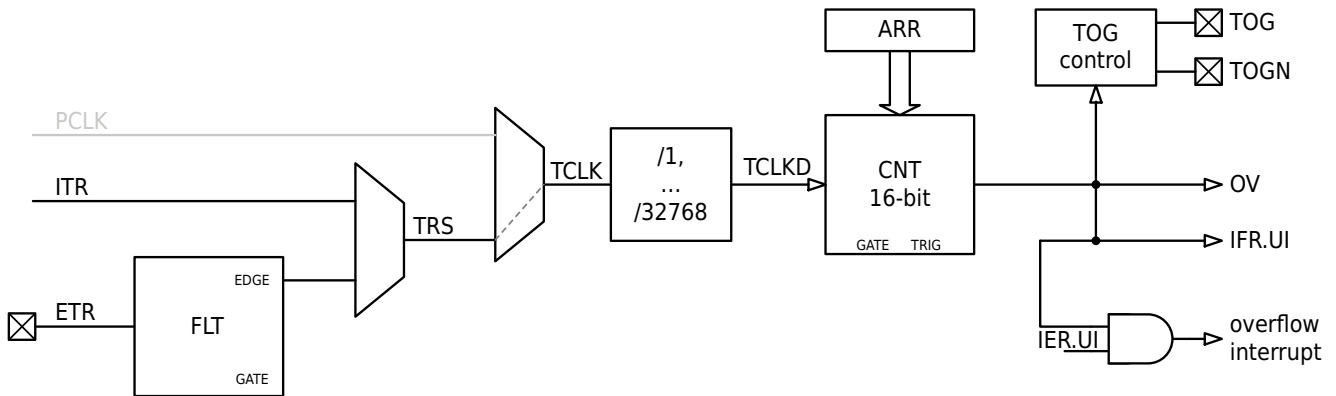


Figure 11-5 BTIM Counter Mode Block Diagram

11.3.1.6 Trigger Start Mode

When BTIMx_CR.MD is set to 0x2, BTIMx works in the trigger start mode, and the counting clock is the clock TCLKD after PCLK frequency division. In this mode, the timer has two start methods: set BTIMx_CR.CEN to 1 or there is an expected level change on the TRS signal.

After starting, the counter starts counting up from the initial value, overflows after counting to ARR and restarts counting from 0; set BTIMx_CR.CEN to 0 at any time, you can stop the timer.

Select TRS signal source as ETR or ITR through BTIMx_CR.TRS.

The rising edge or falling edge of the ETR pin input signal can be selected to start the timer through BTIMx_CR.ETP.

Every time an expected trigger signal is detected, BTIMx_IFR.TI will be set to 1 by hardware. If the corresponding interrupt is enabled, a trigger interrupt can be generated. The user should clear the trigger flag in the interrupt service routine.

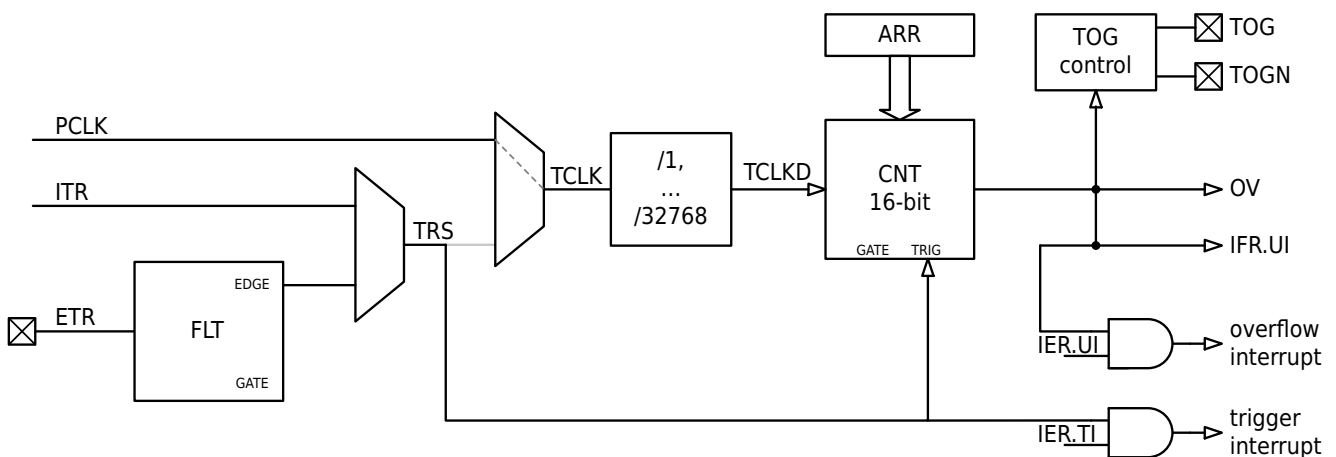


Figure 11-6 BTIM Trigger Start Mode Block Diagram

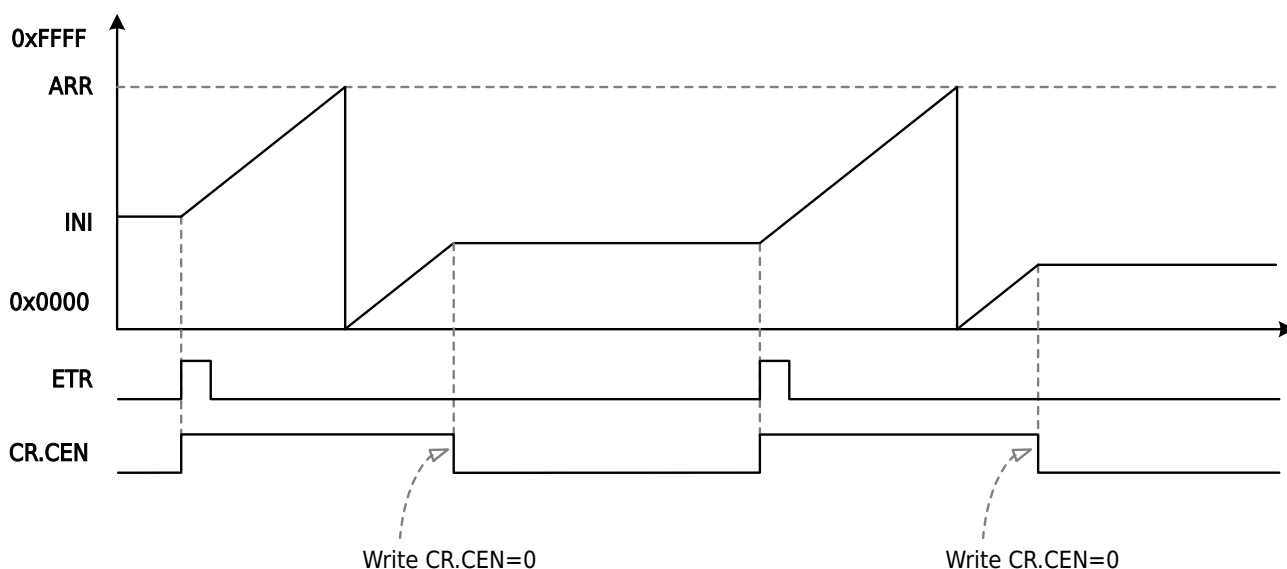


Figure 11-7 BTIM Trigger Start Mode Counting Diagram

11.3.1.7 Gating Mode

When BTIMx_CR.MD is set to 0x3, BTIMx works in gating mode, and the counting clock is the clock TCLKD after PCLK frequency division. In this mode, the timer needs to meet two conditions before it can count: BTIMx_CR.CEN is 1, and the level on the ETR pin meets expectations. When the timer is running, the counter starts counting up from the initial value, overflows after counting to ARR and starts counting from 0 again.

The high or low level of the ETR pin input signal can be selected as the counting level through BTIMx_CR.ETP.

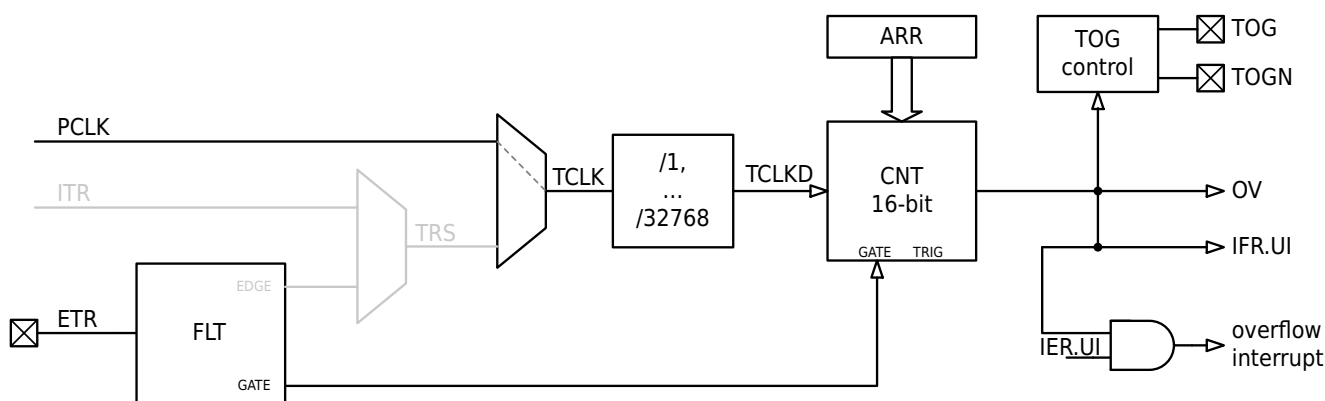


Figure 11-8 BTIM Gating Mode Block Diagram

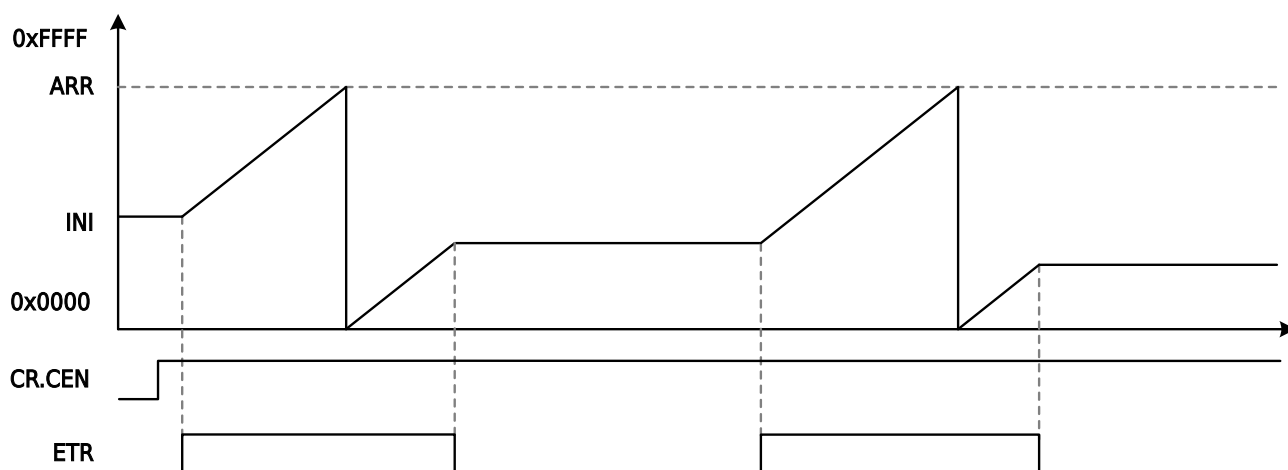


Figure 11-9 BTIM Gating Mode Counting Diagram

11.3.1.8 Timer Cascade

BTIM can be used in cascade through appropriate configuration, such as combining two 16-bit timers into a 32-bit timer. When used in cascade, the count clock of the first stage timer can be PCLK or the input signal from the ETR pin, and the count clock of the subsequent stage timer is the overflow signal of the previous stage timer. The signal cascading has been solidified in the chip, and its cascading method is as follows.

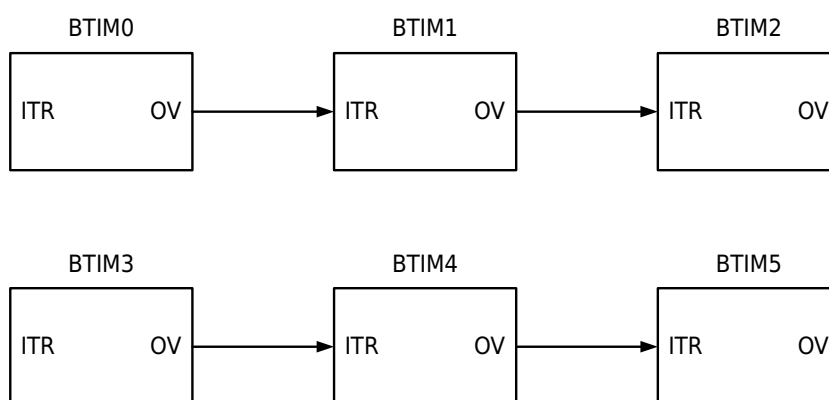


Figure 11-10 BTIM Timer Cascade Diagram

11.3.1.9 Interrupt

BTIM supports trigger interrupts and overflow interrupts. The specific interrupt sources and control methods are shown in the following table.

Interrupt Source	Description	Interrupt Enable	Interrupt Clear Method	NVIC External Interrupt Source
BTIM0/1/2_IFR.TI	Trigger interrupt	BTIM0/1/2_IER.TI	Write 0 to BTIM0/1/2_ICR.TI	CTIM0
BTIM0/1/2_IFR.UI	Overflow interrupt	BTIM0/1/2_IER.UI	Write 0 to BTIM0/1/2_ICR.UI	
BTIM4/5/6_IFR.TI	Trigger interrupt	BTIM4/5/6_IER.TI	Write 0 to BTIM4/5/6_ICR.TI	CTIM1
BTIM4/5/6_IFR.UI	Overflow interrupt	BTIM4/5/6_IER.UI	Write 0 to BTIM4/5/6_ICR.UI	

11.3.2 GTIM Functional Description

11.3.2.1 Function Block Diagram

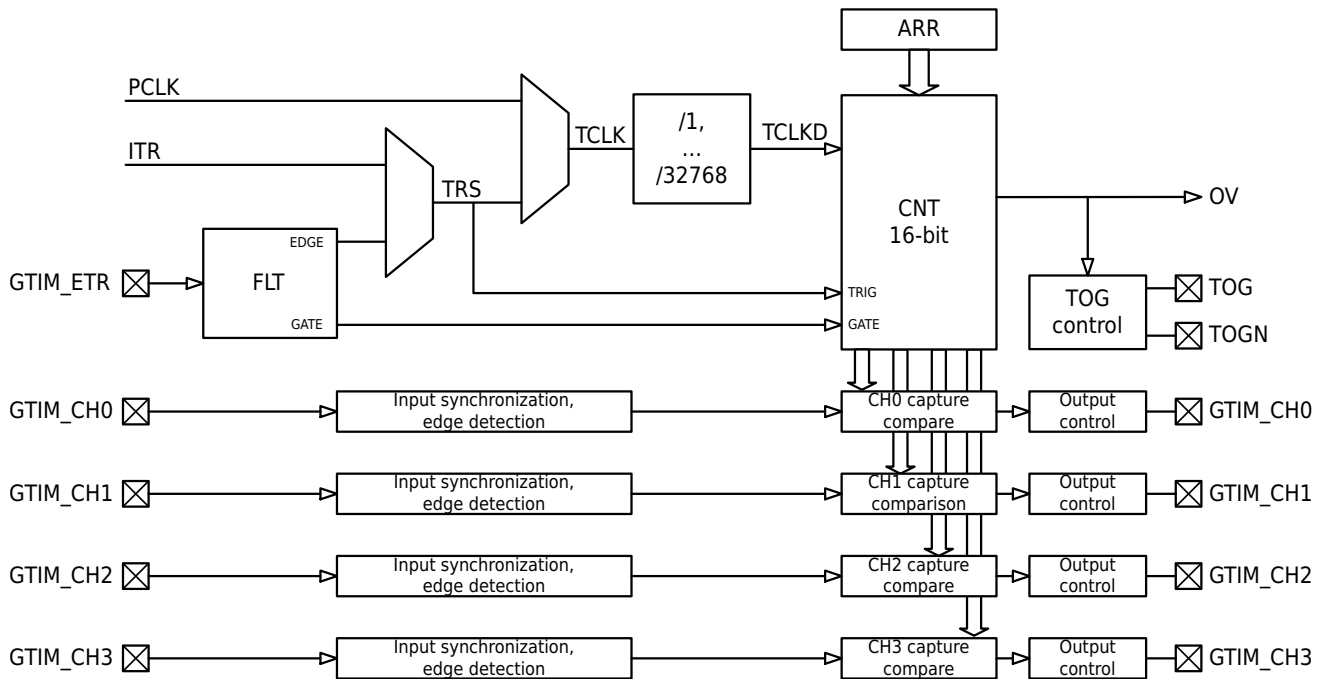


Figure 11-11 GTIM Block Diagram



Note

The source of the ITR signal is detailed in the "On-Chip Peripheral Interconnection" section.

11.3.2.2 Filter Unit

After synchronous filtering, polarity selection, and edge detection of the ETR signal input by the external pin, the gating signal and edge signal are output for use by the counting unit and the triggering unit. Since the ETR signal uses a filter clock for synchronous filtering, the frequency of the input ETR signal should be lower than one-half of the frequency of the filter clock.

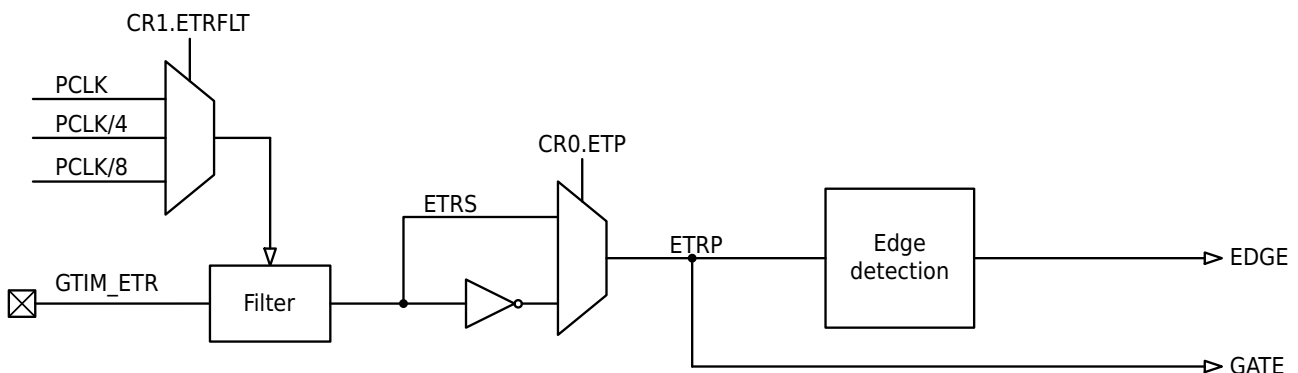


Figure 11-12 GTIM Filter Unit Diagram

11.3.2.3 Counting Unit

On each rising edge of the TCLKD signal, the internal counter CNT is automatically incremented by 1. counting to ARR, an overflow signal OV (width of one PCLK) is generated, and the internal count value is

reset to 0 and counting up starts again. The counting range of the counter is 0 to ARR, and the counting period of the counting unit is ARR+1.

The timer supports two counting modes: One-Shot mode and continuous counting mode.

- When GTIM_CR0.OST is set to 1, the timer works in One-Shot mode; the timer stops counting immediately after an overflow occurs, and the hardware automatically clears CEN.
- When setting GTIM_CR0.OST to 0, the timer works in continuous counting mode; as long as GTIM_CR0.CEN is 1, the timer will keep counting.

Every time there is an overflow, GTIM_IFR.UI will be set to 1 by hardware. If the corresponding interrupt is enabled, an overflow interrupt can be generated. The user should clear the overflow flag in the interrupt service routine.

If GTIM_CR0.TOGEN is 1, the TOG/TOGN output level toggles once per overflow. Users can output the TOG/TOGN signal to a pin through GPIO multiplexing function to drive external circuits.

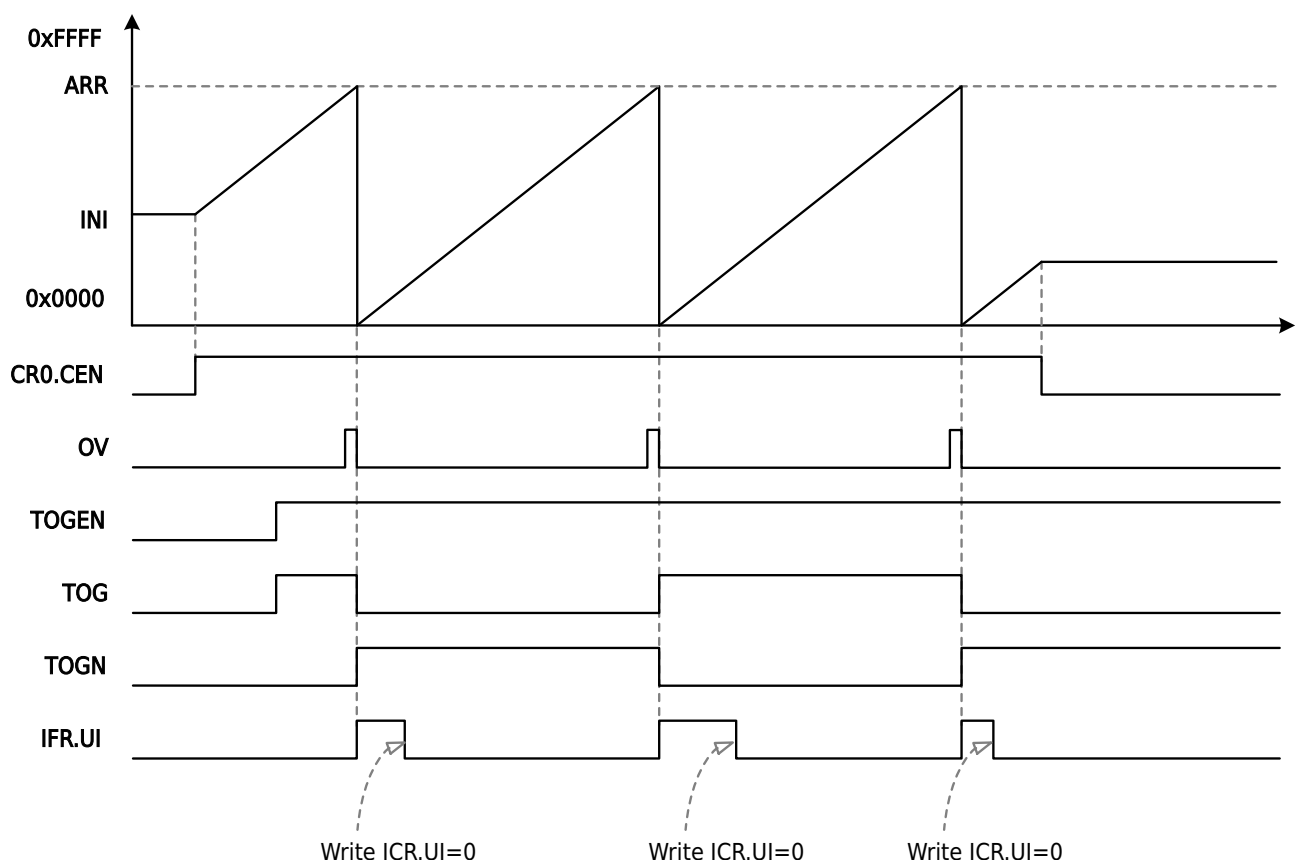


Figure 11-13 GTIM Counter Waveform

11.3.2.4 Timer Mode

GTIM_CR0.MD is set to 0x0, GTIM works in timer mode. In this mode, the counting clock is TCLKD derived from the divided PCLK. When GTIM_CR0.CEN is set to 1, the timer starts incrementing from the initial value. Upon reaching the ARR value, an overflow occurs and the counter restarts from 0.

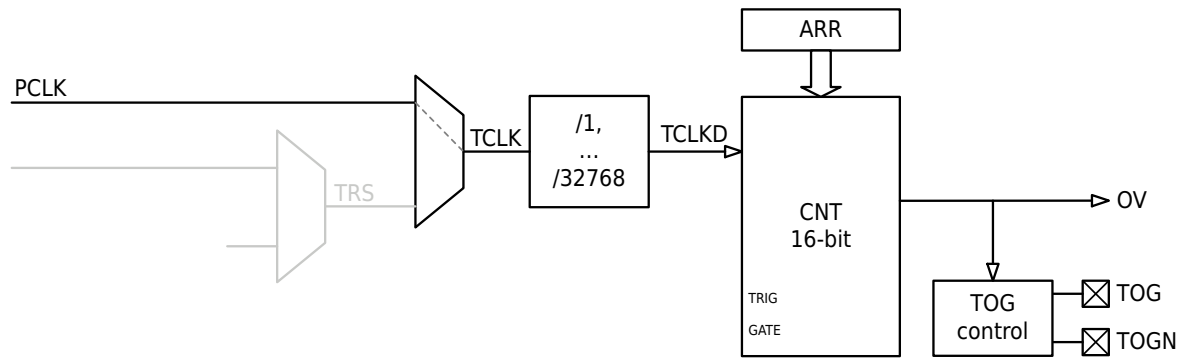


Figure 11-14 GTIM Timing Mode Block Diagram

11.3.2.5 Counter Mode

When GTIM_CR0.MD is set to 0x1, GTIM works in counter mode. In this mode, the counting clock is the signal input by the ITR or ETR pin after frequency division. When GTIM_CR0.CEN is set to 1, the counter begins incrementing from the initial value. Upon reaching the ARR value, an overflow occurs and the counting restarts from 0.

Through GTIM_CR0.TRS, you can select the counting clock source as ITR or ETR; through GTIM_CR0.ETP, you can choose to count the rising edge or falling edge of the ETR pin input signal.

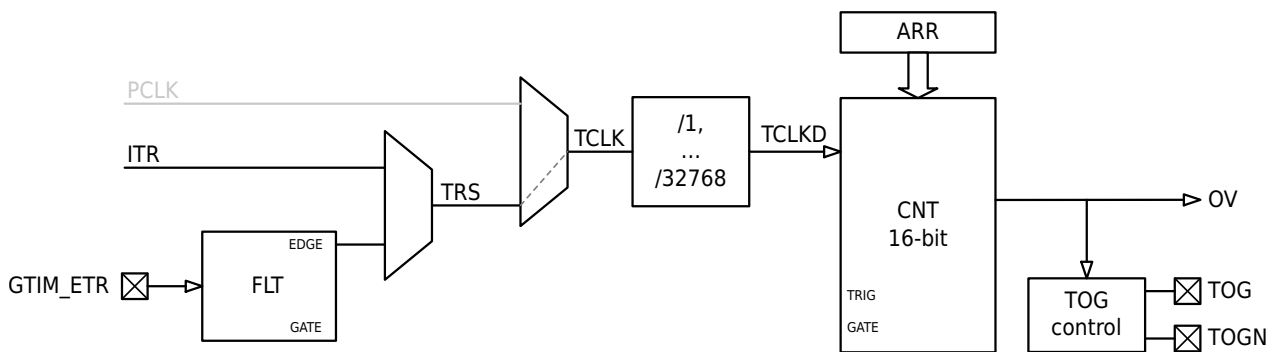


Figure 11-15 GTIM Counter Mode Block Diagram

11.3.2.6 Trigger Start Mode

When GTIM_CR0.MD is set to 0x2, GTIM works in trigger start mode, and the count clock is TCLKD after PCLK frequency division. In this mode, the timer has two startup methods: setting GTIM_CR0.CEN to 1 or detecting the expected level change on the TRS signal.

After starting, the counter starts counting up from the initial value, overflows after counting to ARR, and restarts counting from 0; set GTIM_CR0.CEN to 0 at any time, you can stop the timer.

Select the TRS signal source as ETR or ITR through GTIM_CR0.TRS.

The rising edge or falling edge of the ETR pin input signal can be selected to start the timer through GTIM_CR0.ETP.

Every time an expected trigger signal is detected, GTIM_IFR.TI will be set to 1 by hardware. If the corresponding interrupt is enabled, a trigger interrupt can be generated. The user should clear the trigger flag in the interrupt service routine.

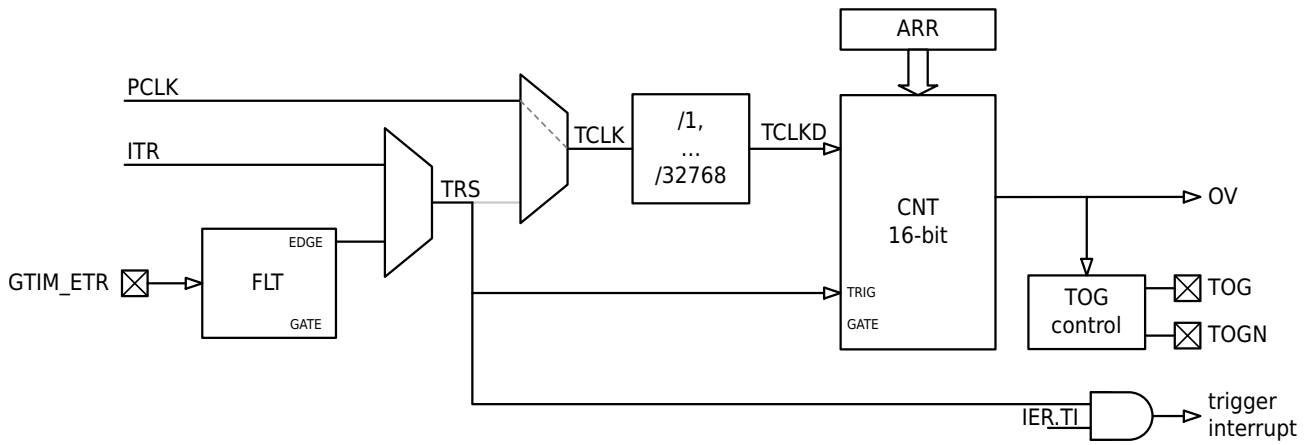


Figure 11-16 GTIM Trigger Start Mode Block Diagram

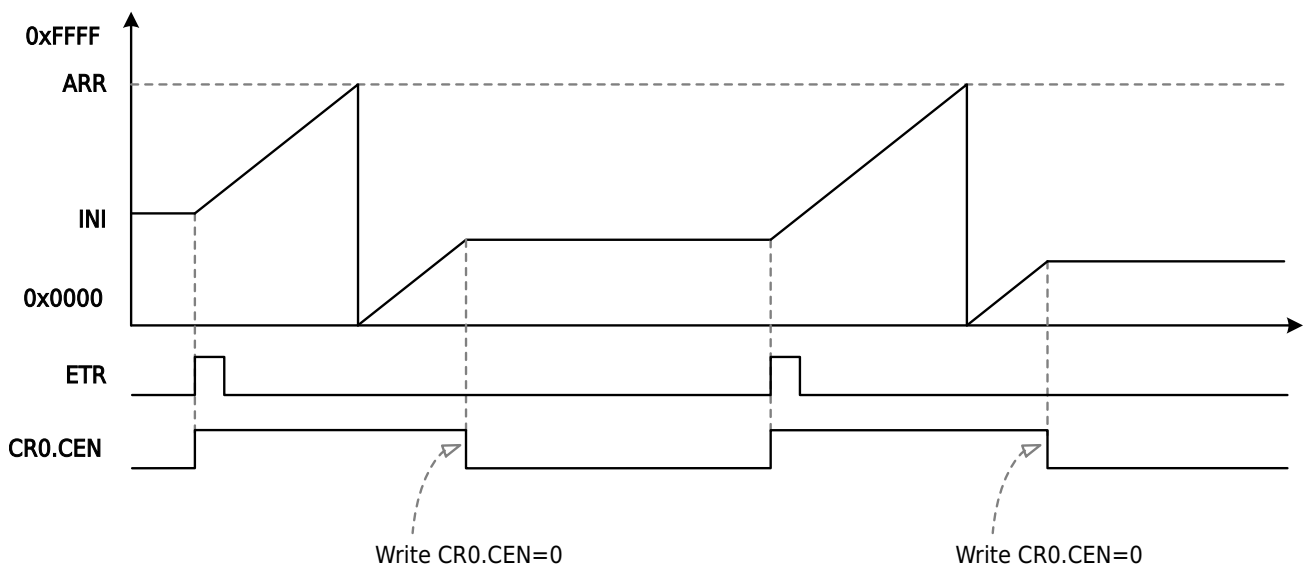


Figure 11-17 GTIM Trigger Start Mode Counting Diagram

11.3.2.7 Gating Mode

When GTIM_CR0.MD is set to 0x3, GTIM works in gating mode, and the count clock is TCLKD after PCLK frequency division. In this mode, the timer needs to meet two conditions before it can count: GTIM_CR0.CEN is 1, and the level on the ETR pin meets expectations. When the timer is running, the counter starts counting up from the initial value, overflows after counting to ARR and starts counting from 0 again.

The high or low level for the ETR pin input signal can be selected as the counting level via GTIM_CR0.ETP.

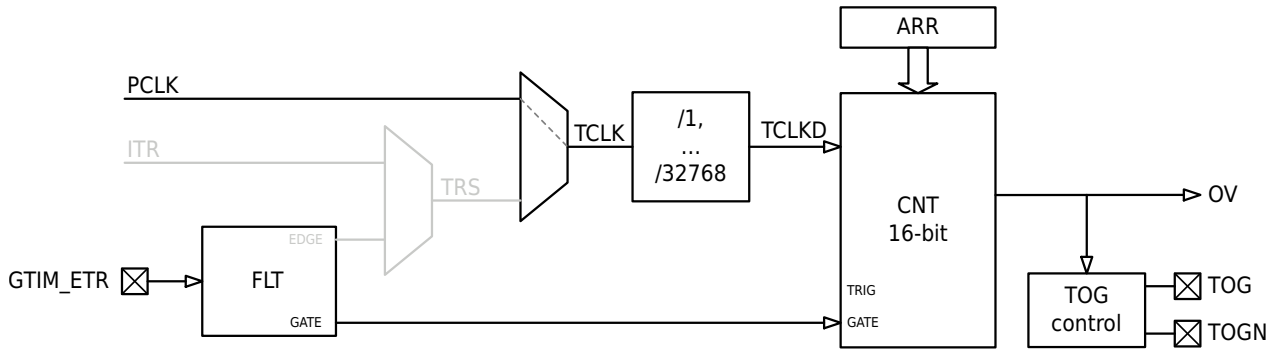


Figure 11-18 GTIM Gating Mode Block Diagram

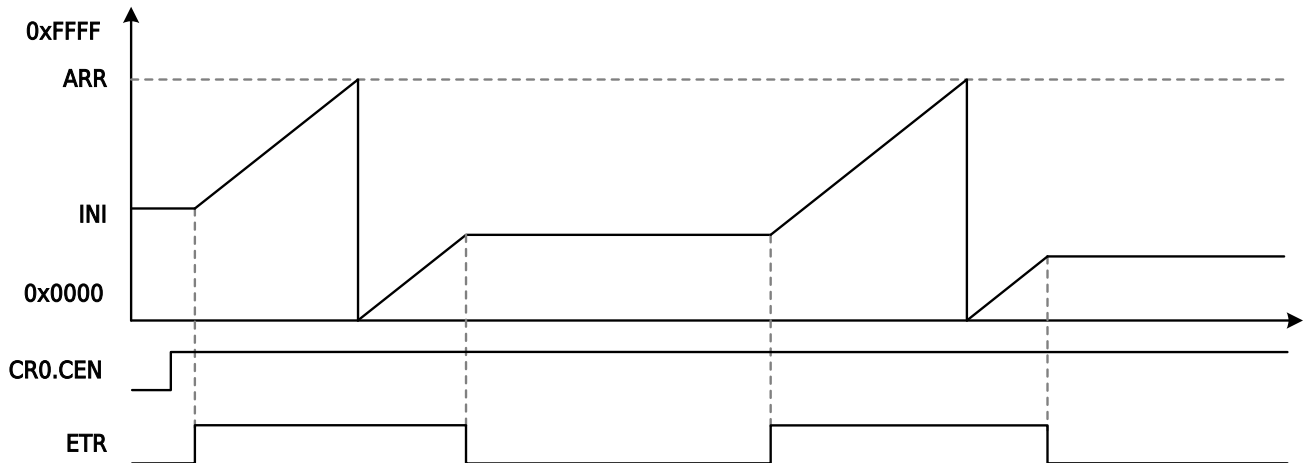


Figure 11-19 GTIM Gating Mode Counting Diagram

11.3.2.8 Compare Capture Function

The GTIM features four compare capture channels: CH0-CH3.

11.3.2.8.1 Capture Function

When CMMR.CCyM (y=0-3) is configured to 1-3, CHy operates in capture mode and supports three capture methods:

- 0b001: Rising edge capture
- 0b010: Falling edge capture
- 0b011: Capture on both rising and falling edges

When the desired pulse edge is detected on CHy channel, the hardware automatically writes the current counter value into the CCRy register and sets IFR.CCy. If the corresponding channel interrupt is enabled, a compare capture interrupt will be generated. The user should clear this interrupt flag in the interrupt service routine.

The figure below shows the schematic of two channels for input capture. Among them, CH0 is captured on the rising edge; CH1 is captured on the rising and falling edges at the same time.

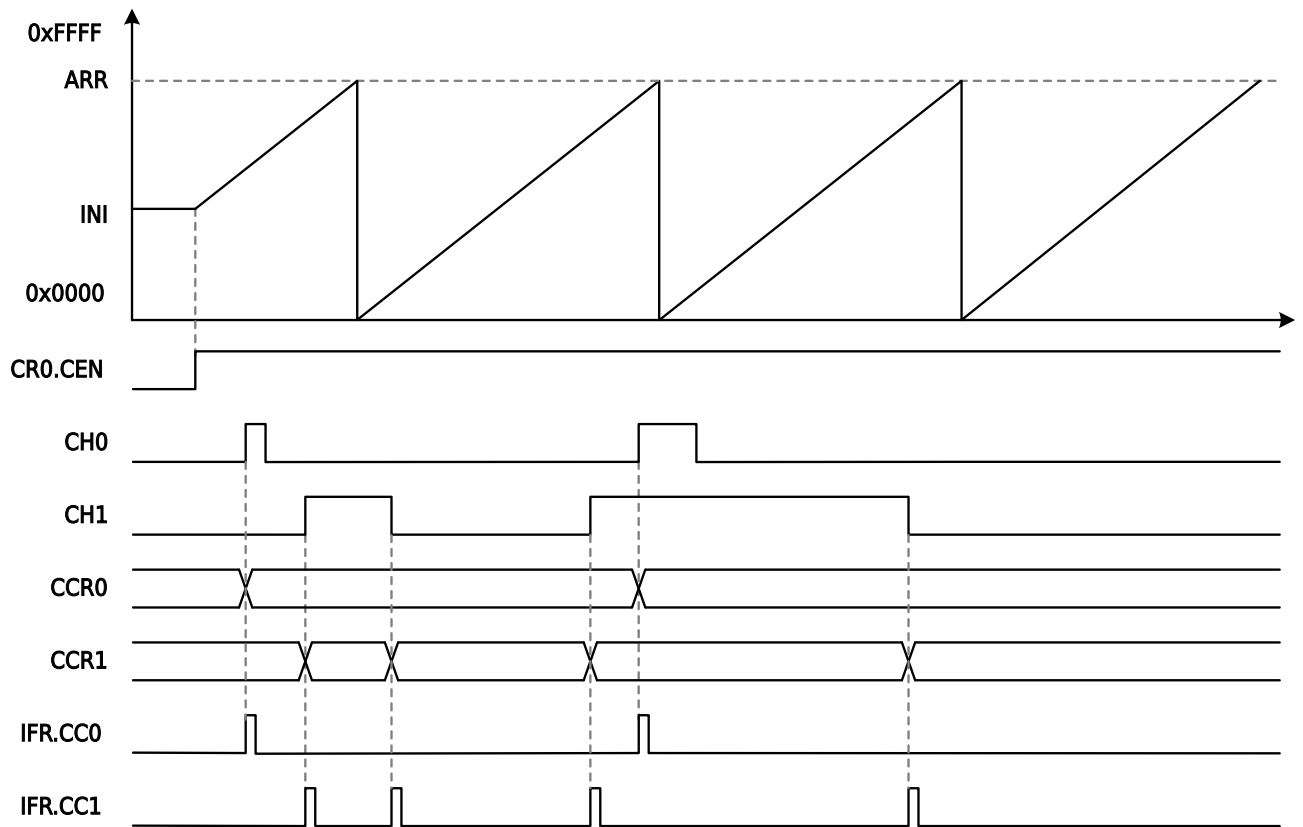


Figure 11-20 GTIM Capture Function Diagram

11.3.2.8.2 Compare Function

When CMMR.CCyM ($y=0-3$) is configured to 4-7, CHy operates in compare mode and can generate four types of waveforms:

- 0b100: Force output low
- 0b101: Force output high
- 0b110: PWM non-inverted output (outputs high level when $CNT \geq CCRy$)
- 0b111: PWM inverted output (outputs high level when $CNT < CCRy$)



Note

When the ARR value is not 0xFFFF, the PWM adjustment range can be achieved from 0% to 100%.

For example, in non-inverted output mode when ARR is 99, the duty cycle corresponding to CCR values from 0-100 is 100%-0%.

The figure below shows the diagram of the comparison output of two channels. Among them, CH0 adopts PWM forward output; CH1 adopts PWM reverse output.

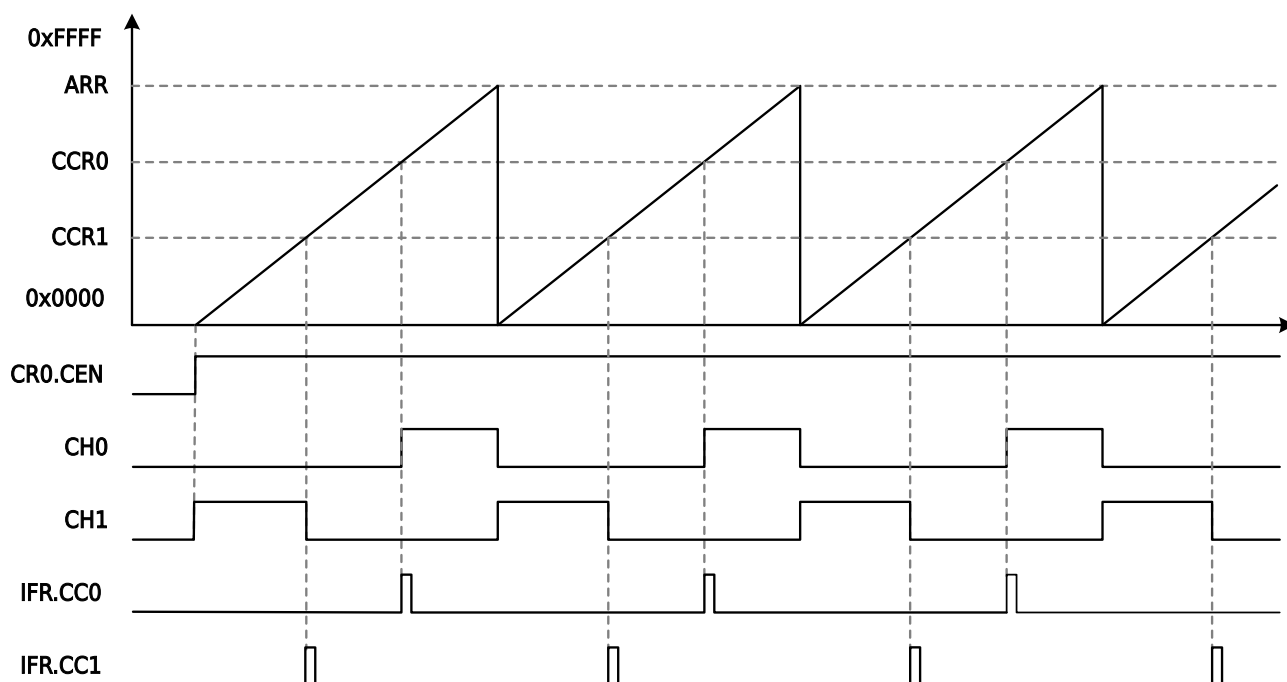


Figure 11-21 GTIM Compare Function Diagram

11.3.2.9 Timer Triggers ADC

In GTIM mode, both count overflow and CH0 compare/capture events can generate trigger signals to initiate automatic ADC conversion. For detailed information on the compare/capture function, please refer to [Compare Capture Function](#). The corresponding signal triggers for ADC are controlled via GTIM_IER.OVTRAD (for overflow triggers) and GTIM_IER.C0TRAD (for CH0 compare/capture triggers), respectively.

11.3.2.10 On-Chip Peripheral Interconnection

Trigger interconnection information as below:

Table 11-3 GTIM TRS Trigger Source List

CR0.TRS	0b00	0b01	0b10	0b11
GTIM0	GTIM0_ETR	GTIM1_OV	-	ATIM3_TRGO
GTIM1	GTIM1_ETR	GTIM0_OV	-	ATIM3_TRGO

The CH0 input of GTIM0 can be sourced from port inputs, or alternatively connected to other modules/ports through the port function register selection. For specific configurations, refer to the description of register "Port Auxiliary Function Timer Capture Input Selection (GPIO_TIMCPS)" in the "General-Purpose I/O (GPIO)" chapter.

The ETR input of GTIM0 can be sourced from port inputs, or alternatively connected to other modules/ports through the port function register selection. For specific configurations, refer to the description of register "Port Auxiliary Function Timer ETR Selection (GPIO_TIMES)" in the "General-Purpose I/O (GPIO)" chapter.

11.3.2.11 Interrupt

GTIM supports trigger interrupts, overflow interrupts, and channel capture/compare interrupts. The specific interrupt sources and control methods are shown in the table below.

Interrupt Source	Description	Interrupt Enable	Interrupt Clear Method	NVIC External Interrupt Source
GTIM0_IFR.TI	Trigger interrupt	GTIM0_IER.TI	Write 0 to GTIM0_ICR.TI	CTIM0
GTIM0_IFR.UI	Overflow interrupt	GTIM0_IER.UI	Write 0 to GTIM0_ICR.UI	
GTIM0_IFR.CCx	Channel capture/compare interrupt	GTIM0_IER.CCx	Write 0 to GTIM0_ICR.CCx	
GTIM1_IFR.TI	Trigger interrupt	GTIM1_IER.TI	Write 0 to GTIM1_ICR.TI	CTIM1
GTIM1_IFR.UI	Overflow interrupt	GTIM1_IER.UI	Write 0 to GTIM1_ICR.UI	
GTIM1_IFR.CCx	Channel capture/compare interrupt	GTIM1_IER.CCx	Write 0 to GTIM1_ICR.CCx	

11.4 Register Description

11.4.1 Register List

- BTIM0 base address: 0x40001800
- BTIM1 base address: 0x40001900
- BTIM2 base address: 0x40001A00
- BTIM3 base address: 0x40001C00
- BTIM4 base address: 0x40001D00
- BTIM5 base address: 0x40001E00



Note

BTIM is enabled when SYSCTRL1.CTIMERx_FUN_SEL is set to 1.

Table 11-5 BTIM Register Offset

Offset	Register	Description	Width
0x00	BTIM_ARR	Reload Register	32
0x04	BTIM_CNT	Count Register	32
0x10	BTIM_CR	Control Register	32
0x14	BTIM_IER	Interrupt Enable Control Register	32
0x18	BTIM_IFR	Interrupt Flag Register	32
0x1C	BTIM_ICR	Interrupt Flag Clear Register	32
0x48	BTIM_AIFR	Composite Interrupt Flag Register	32
0x4C	BTIM_AICR	Composite Interrupt Flag Clear Register	32

- GTIM0 base address: 0x40001800
- GTIM1 base address: 0x40001C00



Note

GTIM is enabled when SYSCTRL1.CTIMERx_FUN_SEL is set to 0.

Table 11-6 GTIM Register Offset

Offset	Register	Description	Width
0x300	GTIM_ARR	Reload Register	32
0x304	GTIM_CNT	Count Register	32
0x308	GTIM_CMMR	Compare Capture Control Register	32
0x30C	GTIM_CR1	Control Register 1	32
0x310	GTIM_CR0	Control Register 0	32
0x314	GTIM_IER	Interrupt Enable Control Register	32

Offset	Register	Description	Width
0x318	GTIM_IFR	Interrupt Flag Register	32
0x31C	GTIM_ICR	Interrupt Flag Clear Register	32
0x320+0x4*y	GTIM_CCRy	CHy Compare Capture Register (y=0-3)	32

11.4.2 BTIM Register

11.4.2.1 Reload Register (BTIM_ARR)

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																ARR															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	ARR	Timer reload value	RW

11.4.2.2 Count Register (BTIM_CNT)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CNT															
Access																																
																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	CNT	Timer count value	RW

11.4.2.3 Control Register (BTIM_CR)

Offset	Bit Position																																													
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0														
Reset	0x00000000																																													
Name	Reserved																				ETP	TR	S	PR	TOGEN	MD	CEN																			
Access																					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11	ETP	ETR signal polarity selection for external pin input 0b0: ETR forward (rising edge active in trigger mode, high level active in gating mode) 0b1: ETR reverse (falling edge active in trigger mode, low level active in gating mode)	RW
10:9	TRRS	Trigger source selection 0b00: Signal from ETR pin input 0b01: ITR, see the section "Timer Cascade" - Others: Setting prohibited  Note BTIM0-2 share one ETR pin; BTIM3-5 share another ETR pin.	RW
8	OST	Single/continuous counting mode control 0b0: Continuous counting mode 0b1: Single counting mode	RW
7:4	PRS	Prescaler division ratio configuration 0b0000: DIV1 0b0001: DIV2 0b0010: DIV4 0b0011: DIV8 0b0100: DIV16 0b0101: DIV32 0b0110: DIV64 0b0111: DIV128 0b1000: DIV256 0b1001: DIV512 0b1010: DIV1024 0b1011: DIV2048 0b1100: DIV4096 0b1101: DIV8192 0b1110: DIV16384 0b1111: DIV32768	RW

Bit	Name	Functional Description	Access
3	TOGEN	TOG pin output enable control 0b0: TOG, TOGN output levels are both 0 0b1: TOG, TOGN output signals with opposite levels	RW
2:1	MD	Timer working mode configuration 0b00: Timer mode, counting clock is PCLK 0b01: Counter mode, counting clock is TRS signal 0b10: Trigger start mode, counting clock is PCLK; TRS signal triggers counter start 0b11: Gating mode, counting clock is PCLK; ETR pin input signal used as gate	RW
0	CEN	Timer enable control 0b0: Timer stopped 0b1: Timer enabled  Note After trigger start mode is triggered, CEN is set to 1 by hardware.	RW

11.4.2.4 Interrupt Enable Control Register (BTIM_IER)

Offset	Bit Position																																
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																														TI		UI
Access																															RW		RW

Bit	Name	Functional Description	Access
31:2	Reserved	-	-
1	TI	Trigger interrupt enable control 0b0: Trigger interrupt disabled 0b1: Trigger interrupt enabled	RW
0	UI	Overflow interrupt enable control 0b0: Overflow interrupt disabled 0b1: Overflow interrupt enabled	RW

11.4.2.5 Interrupt Flag Register (BTIM_IFR)

Offset	Bit Position																																
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																																
Access																															RO	TI	RO

Bit	Name	Description	Access
31:2	Reserved	-	-
1	TI	Trigger flag 0b0: No trigger event occurred 0b1: A trigger event occurred	RO
0	UI	Counter overflow flag 0b0: Counter has not overflowed 0b1: Counter overflowed	RO

11.4.2.6 Interrupt Flag Clear Register (BTIM_ICR)

Offset	Bit Position																																
0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000003																																
Name	Reserved																																
Access																																	

Bit	Name	Functional Description	Access
31:2	Reserved	-	-
1	TI	Trigger flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
0	UI	Counter overflow flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

11.4.2.7 Composite Interrupt Flag Register (BTIM_AIFR)

Offset	Bit Position																																					
0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved																										TI25		UI25		TI14		UI14		TI03		UI03	
Access																											RO		RO		RO		RO		RO		RO	

Bit	Name	Functional Description	Access
31:6	Reserved	-	-
5	TI25	BTIM2/5 trigger flag 0b0: No trigger event occurred 0b1: A trigger event occurred	RO
4	UI25	BTIM2/5 overflow flag 0b0: Counter has not overflowed 0b1: Counter overflowed	RO
3	TI14	BTIM1/4 trigger flag 0b0: No trigger event occurred 0b1: A trigger event occurred	RO
2	UI14	BTIM1/4 overflow flag 0b0: Counter has not overflowed 0b1: Counter overflowed	RO
1	TI03	BTIM0/3 trigger flag 0b0: No trigger event occurred 0b1: A trigger event occurred	RO
0	UI03	BTIM0/3 overflow flag 0b0: Counter has not overflowed 0b1: Counter overflowed	RO

11.4.2.8 Composite Interrupt Flag Clear Register (BTIM_AICR)

Offset	Bit Position																																					
0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x0000003F																																					
Name	Reserved																											TI25		UI25		TI14		UI14		TI03		UI03
Access																											WO	TI25	WO	UI25	WO	TI14	WO	UI14	WO	TI03	WO	UI03

Bit	Name	Functional Description	Access
31:6	Reserved	-	-
5	TI25	BTIM2/5 trigger flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
4	UI25	BTIM2/5 overflow flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
3	TI14	BTIM1/4 trigger flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
2	UI14	BTIM1/4 overflow flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
1	TI03	BTIM0/3 trigger flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
0	UI03	BTIM0/3 overflow flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

11.4.3 GTIM Register

11.4.3.1 Reload Register (GTIM_ARR)

Offset	Bit Position																															
0x300	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF																															
Name	Reserved																ARR															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	ARR	Timer reload value	RW

11.4.3.2 Count Register (GTIM_CNT)

Offset	Bit Position																															
0x304	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CNT															
Access																																
																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	CNT	Timer count value	RW

11.4.3.3 Control Register 1 (GTIM_CR1)

Offset	Bit Position																															
0x30C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								ETRFLT		Reserved					
Access																									RW							

Bit	Name	Functional Description	Access
31:9	Reserved	-	-
8:7	Reserved	All bits read as 0, Write as 0	-
6:4	ETRFLT	ETR pin input signal filter configuration The sampling clock is PCLK or PCLK frequency division, and the number of sampling points is N 0b000: No filter 0b001: $F_{\text{sample}} = \text{PCLK}$, $N=2$ 0b010: $F_{\text{sample}} = \text{PCLK}$, $N=4$ 0b011: $F_{\text{sample}} = \text{PCLK}$, $N=6$ 0b100: $F_{\text{sample}} = \text{PCLK}/4$, $N=4$ 0b101: $F_{\text{sample}} = \text{PCLK}/4$, $N=6$ 0b110: $F_{\text{sample}} = \text{PCLK}/8$, $N=4$ 0b111: $F_{\text{sample}} = \text{PCLK}/8$, $N=6$	RW
3:0	Reserved	-	-

11.4.3.4 Control Register 0 (GTIM_CR0)

Offset	Bit Position																															
0x310	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				ETP	TRS		OST	PRS				TOGEN	MD		CEN
Access																					RW	RW		RW	RW				RW	RW		RW

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11	ETP	ETR signal polarity selection for external pin input 0b0: ETR forward (rising edge active in trigger mode, high level active in gating mode) 0b1: ETR reverse (falling edge active in trigger mode, low level active in gating mode)	RW
10:9	TRRS	Trigger source selection 0b00: Signal from ETR pin input 0b01: Internal interconnection 1 0b10: Internal interconnection 2 0b11: Internal interconnection 3  Note Internal interconnection details can be found in the On-chip Peripheral Interconnection section.	RW
8	OST	Single/continuous counting mode control 0b0: Continuous counting mode 0b1: Single counting mode	RW
7:4	PRS	Prescaler division ratio configuration 0b0000: DIV1 0b0001: DIV2 0b0010: DIV4 0b0011: DIV8 0b0100: DIV16 0b0101: DIV32 0b0110: DIV64 0b0111: DIV128 0b1000: DIV256 0b1001: DIV512 0b1010: DIV1024 0b1011: DIV2048 0b1100: DIV4096 0b1101: DIV8192 0b1110: DIV16384	RW

Bit	Name	Functional Description	Access
7:4	PRS	● 0b1111: DIV32768	RW
3	TOGEN	TOG pin output enable control ● 0b0: TOG, TOGN output levels are both 0 ● 0b1: TOG, TOGN output signals with opposite levels	RW
2:1	MD	Timer working mode configuration ● 0b00: Timer mode, counting clock is PCLK ● 0b01: Counter mode, counting clock is from TRS selection signal ● 0b10: Trigger start mode, counting clock is PCLK; TRS signal triggers counter start ● 0b11: Gating mode, counting clock is PCLK, ETR pin input signal as gate control	RW
0	CEN	Timer enable control ● 0b0: Timer stopped ● 0b1: Timer enabled  Note After trigger start mode is triggered, CEN is set to 1 by hardware.	RW

11.4.3.5 Interrupt Enable Control Register (GTIM_IER)

Offset	Bit Position																															
0x314	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				C0TRAD	OVTRAD	Reserved				CC3	CC2	CC1	CC0	TI	UI
Access																					RW	RW					RW	RW	RW	RW	RW	RW

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11	C0TRAD	Compare capture 0 triggers ADC enable control 0b0: Disable 0b1: Enable	RW
10	OVTRAD	Overflow triggers ADC enable control 0b0: Disable 0b1: Enable	RW
9:6	Reserved	-	-
5	CC3	CH3 capture compare interrupt enable control 0b0: Disable 0b1: Enable	RW
4	CC2	CH2 capture compare interrupt enable control 0b0: Disable 0b1: Enable	RW
3	CC1	CH1 capture compare interrupt enable control 0b0: Disable 0b1: Enable	RW
2	CC0	CH0 capture compare interrupt enable control 0b0: Disable 0b1: Enable	RW
1	TI	Trigger interrupt enable control 0b0: Disable 0b1: Enable	RW
0	UI	Overflow interrupt enable control 0b0: Disable 0b1: Enable	RW

11.4.3.6 Interrupt Flag Register (GTIM_IFR)

Offset	Bit Position																																																	
0x318	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																		
Reset	0x00000000																																																	
Name	Reserved																																																	
Access																											R0		CC3		R0		CC2		R0		CC1		R0		CC0		R0		TI		R0		UI	

Bit	Name	Functional Description	Access
31:6	Reserved	-	-
5	CC3	CH3 compare capture interrupt flag 0b0: No compare capture event occurred 0b1: A compare capture event occurred	RO
4	CC2	CH2 compare capture interrupt flag 0b0: No compare capture event occurred 0b1: A compare capture event occurred	RO
3	CC1	CH1 compare capture interrupt flag 0b0: No compare capture event occurred 0b1: A compare capture event occurred	RO
2	CC0	CH0 compare capture interrupt flag 0b0: No compare capture event occurred 0b1: A compare capture event occurred	RO
1	TI	Trigger interrupt flag 0b0: No trigger event occurred 0b1: A trigger event occurred	RO
0	UI	Overflow interrupt flag 0b0: Counter not overflowed 0b1: Counter overflowed	RO

11.4.3.7 Interrupt Flag Clear Register (GTIM_ICR)

Offset	Bit Position																																					
0x31C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x0000003F																																					
Name	Reserved																										CC3		CC2		CC1		CC0		TI		UI	
Access																											WO		WO		WO		WO		WO		WO	
																											WO		WO		WO		WO		WO		WO	
																											WO		WO		WO		WO		WO		WO	

Bit	Name	Functional Description	Access
31:6	Reserved	-	-
5	CC3	CH3 compare capture interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
4	CC2	CH2 compare capture interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
3	CC1	CH1 compare capture interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
2	CC0	CH0 compare capture interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
1	TI	Trigger interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
0	UI	Overflow interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

11.4.3.8 Compare Capture Control Register (GTIM_CMMR)

Offset	Bit Position																															
0x308	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CC3M		Reserved	CC2M		Reserved	CC1M		Reserved	CC0M						
Access																	RW			RW			RW			RW						

Bit	Name	Functional Description	Access
31:15	Reserved	-	-
14:12	CC3M	CH3 capture compare mode configuration CC0M for function description	RW
11	Reserved	-	-
10:8	CC2M	CH2 capture compare mode configuration CC0M for function description	RW
7	Reserved	-	-
6:4	CC1M	CH1 capture compare mode configuration CC0M for function description	RW
3	Reserved	-	-
2:0	CC0M	CH0 capture compare mode configuration 0b000: No function 0b001: Rising edge capture 0b010: Falling edge capture 0b011: Capture on both rising and falling edges 0b100: Force output low 0b101: Force output high 0b110: PWM non-inverted output (outputs high level when $CNT \geq CCR0$) 0b111: PWM inverted output (outputs high level when $CNT < CCR0$)	RW

11.4.3.9 Compare Capture Register (GTIM_CCRy) (y=0-3)

Offset: 0x320+0x4*y

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF(GTIM_CCR0/GTIM_CCR2); 0x00000000(GTIM_CCR1/GTIM_CCR3)																															
Name	Reserved																CCR															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	CCR	16-bit compare capture register	RW

12 ATIM3

12.1 Introduction

ATIM3 is a timer composed of 1 counting unit and 6 comparison units, supporting 6 independent PWM outputs or 3 pairs of complementary PWM outputs, with support for 6 capture inputs.

Using timer prescaler, system prescaler and system clock selection, the pulse width and waveform period can be flexibly adjusted; the pulse width can be measured conveniently.

12.2 Features

- 6 independent PWM outputs CH0A, CH0B, CH1A, CH1B, CH2A, CH2B
- 3 complementary PWM outputs (CHxA, CHxB)
- Up to 6 capture inputs
- Pulse width measurement
- Quadrature encoder counting function
- Single pulse mode
- External input counting function
- Dead-time control
- Braking control
- Edge alignment, symmetric center alignment and asymmetric center alignment PWM output

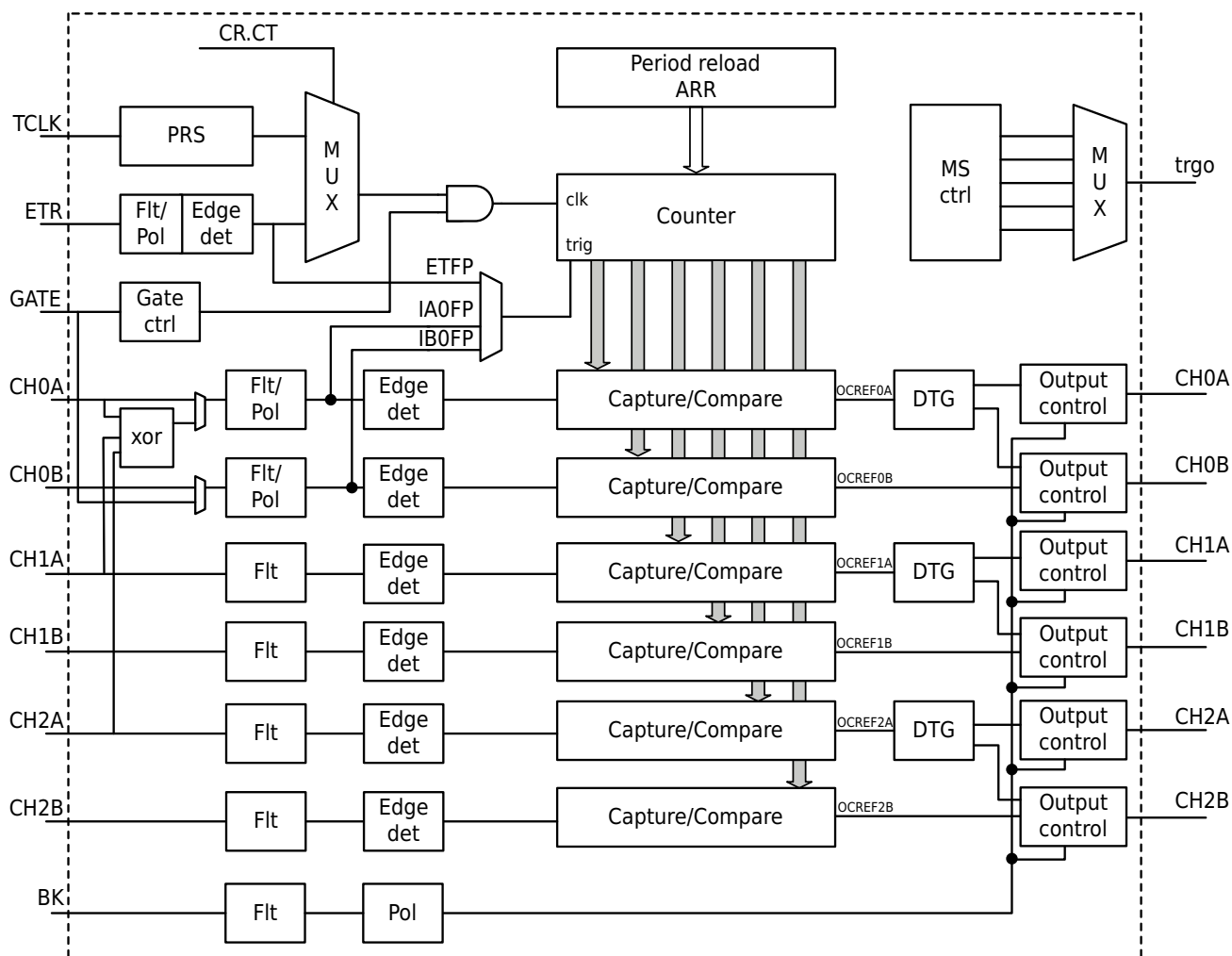


Figure 12-1 ATIM3 Block Diagram

12.3 Functional Description

12.3.1 Timer Clock

The timer uses PCLK as the timer clock, and the timer clock is marked with TCLK below.

12.3.2 Timer Counter

The main module of advanced timer consists of a 16-bit counter and its associated auto-reload register. The counter can count up, count down, or alternately count up and down. The counter clock can be divided by a prescaler.

The counter, auto-reload register and prescaler register can be read and written by software. Read and write operations can be performed even when the counter is running.

12.3.3 Timer Prescaler

When using TCLK as the timer clock, the prescaler can be used. The prescaler settings are as follows:

PRS	0b000	0b001	0b010	0b011	0b100	0b101	0b110	0b111
Frequen- cy divi- sion ratio	1	2	4	8	16	32	64	256

The prescaler does not have a preloaded buffer, so any changes to the prescaler will take effect immediately.

12.3.4 Mode 0 Count Timer Function

The counter counts up in this mode. The counter supports two counting modes, reload mode and free counting mode, and you can choose external clock ETR counting or system clock counting. The gate can control the count mask. Counting to the maximum overflow generates an interrupt. The inverted output signals TOG/TOGN are output through ports CHA/CHB.

The counting range of the reload mode counts up from ARR to 0xFFFF overflow, and then starts counting from ARR, and the counting period is 0xFFFF-ARR+1; the free counting mode overflows after counting from the set count value to 0xFFFFFFFF, and the count value restarts count from 0x0 after overflow.

Function Block Diagram

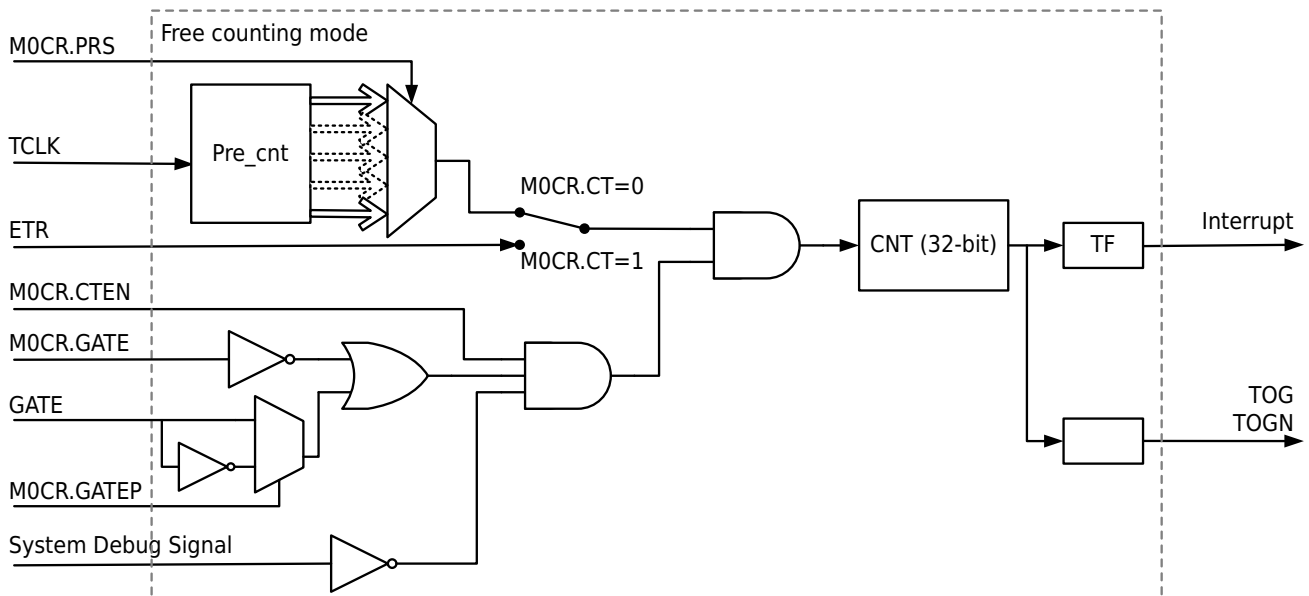


Figure 12-2 Mode0 Count Timer Free Counting Block Diagram

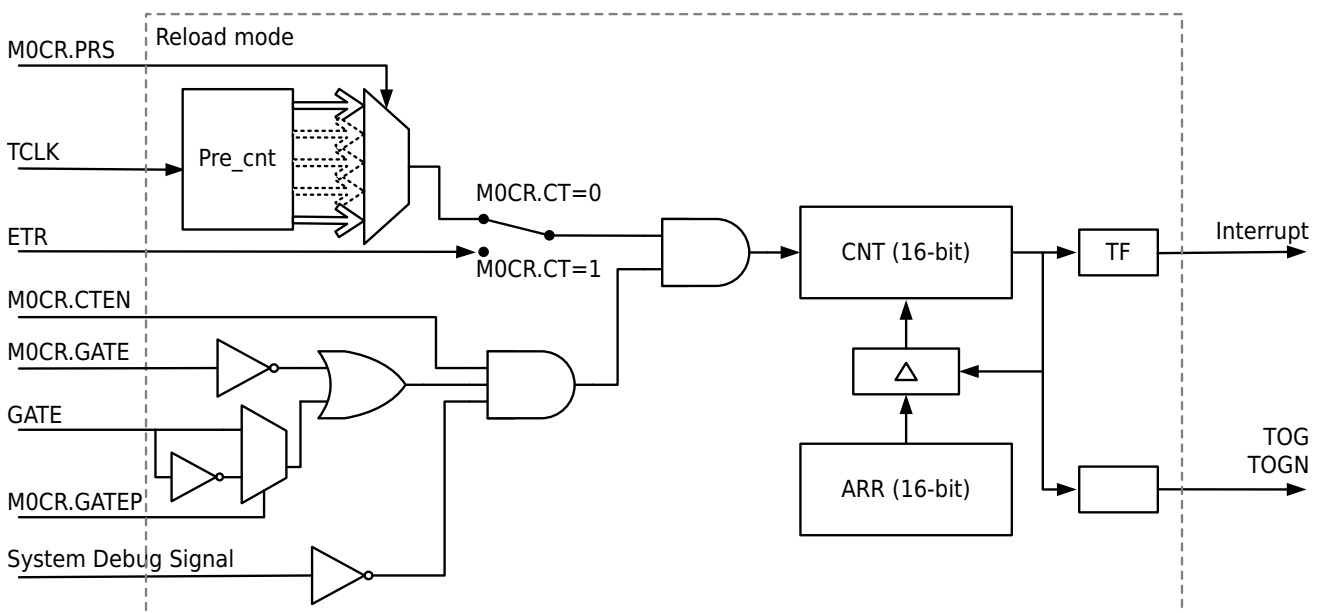


Figure 12-3 Mode0 Count Timer Reload Counting Block Diagram

**Note**

The "System Debug Signal" in the block diagram originates from DEBUG_ACTIVE.

Count Waveform

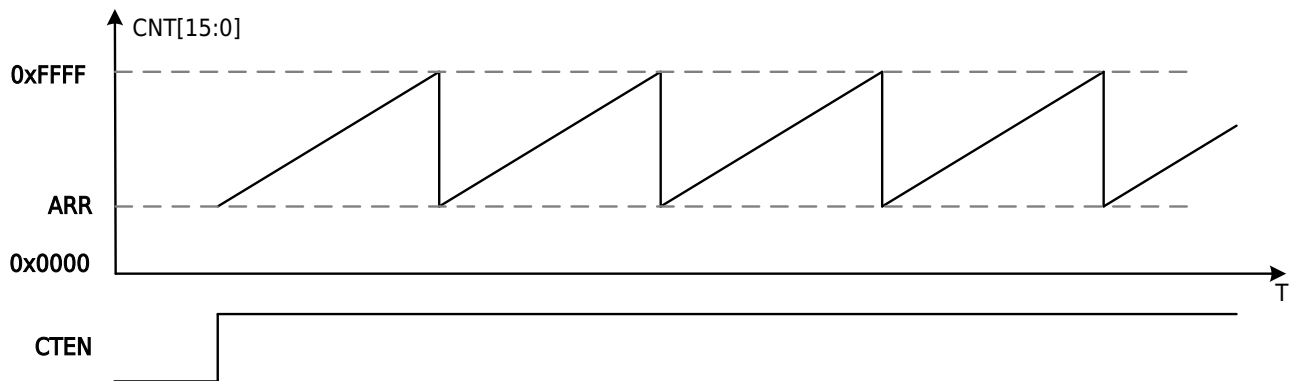


Figure 12-4 Mode0 Count Timer 16-Bit Reload Counting Waveform

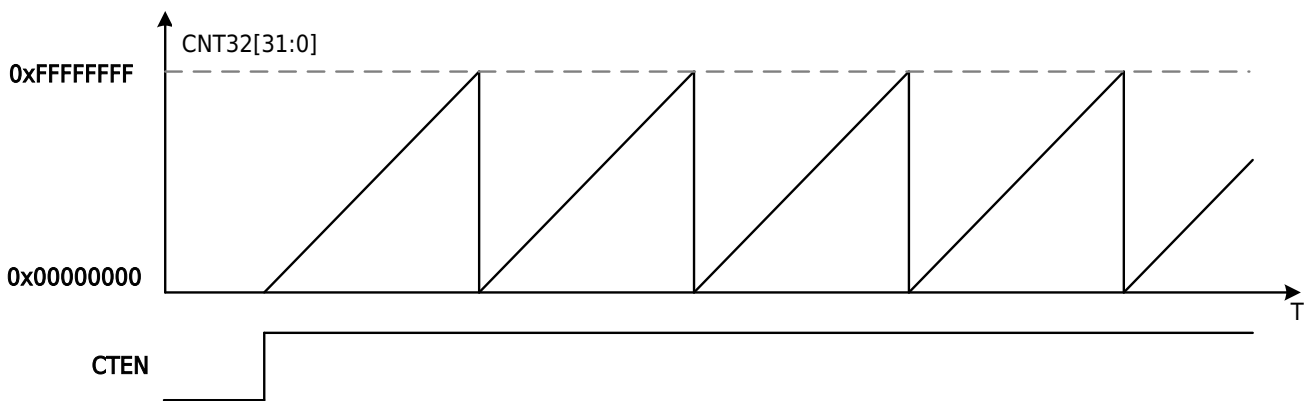


Figure 12-5 Mode0 Count Timer 32-Bit Free Counting Waveform

Counting Function

Counting function are used to determine the number of times an event occurs. In the counting function, the counter increments every falling edge of the corresponding input clock. The input signal is sampled by the internal TCLK. Therefore, the external input signal must remain stable for a duration longer than one TCLK clock cycle. Counting to the maximum value will overflow and generate an interrupt. The interrupt flag needs to be cleared by software.

**Notice**

When the ETR input serves as the count clock of the counter, the ETR clock frequency must not exceed 1/2 of the TCLK clock.

Timing Function

The timing function is used to generate interval timing. In the timing function, the timer has a pre-divided frequency, and the timer accumulates once per clock of each pre-divided frequency. When counting to the maximum value, it will overflow and generate an interrupt. The interrupt flag needs to be cleared by software.

Timing Diagram

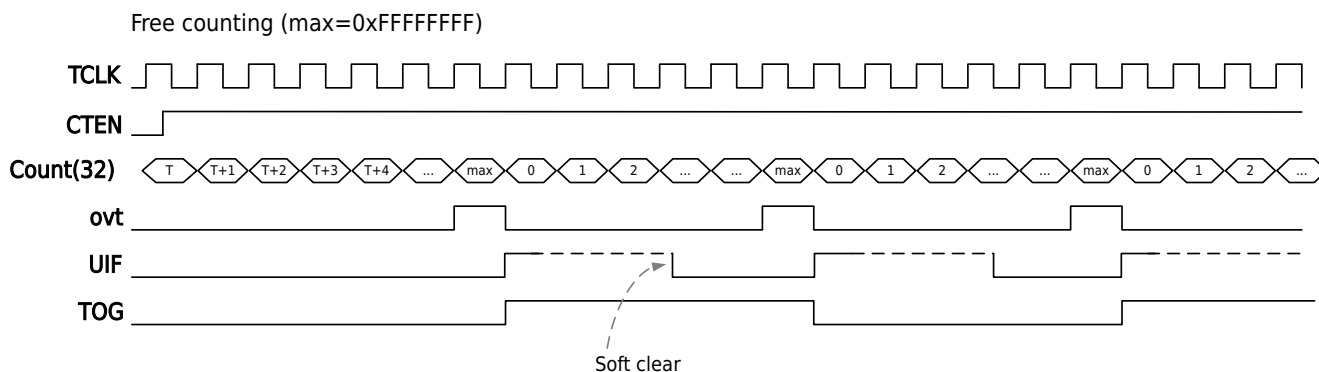


Figure 12-6 Mode0 Count Timer Free Counting Timing Diagram

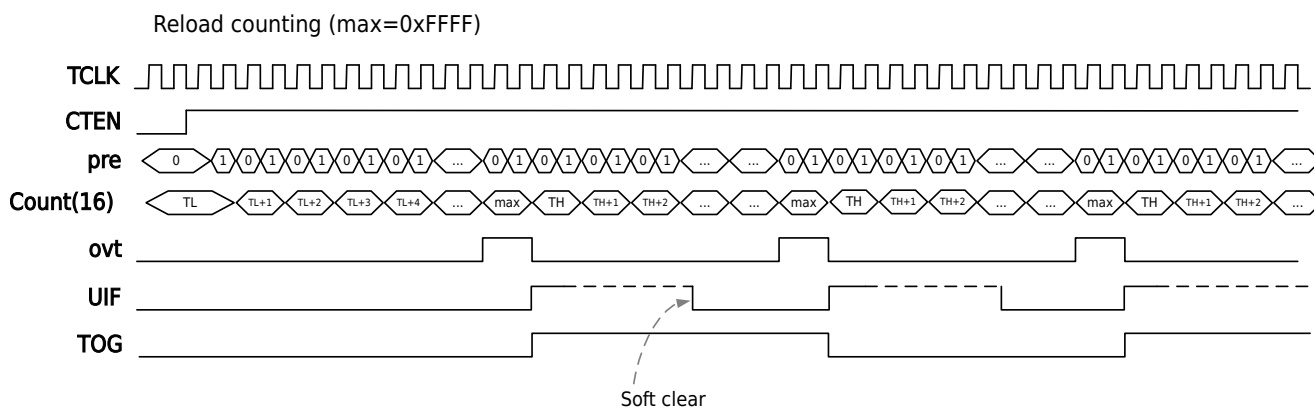


Figure 12-7 Mode0 Count Timer Reload Counting Timing Diagram (Prescaler=2)

Buzzer Function

The Buzzer function can be realized through the toggling output function of the timer. To use toggle output, the DTR.MOE control bit must be enabled. Setting M0CR.TOGEN to 0 simultaneously sets both CHA and CHB outputs to 0. The timer reload mode configurations for Buzzer output at different frequencies under a 4MHz counting clock are as follows:

Buzzer Frequency	Counter Period	Counter Count Value	Counter Reload Value	CNT Initial Value	ARR Reload Value
1000Hz	0.5ms	2000	63536	0xF830	0xF830
2000Hz	0.25ms	1000	64536	0xFC18	0xFC18
4000Hz	0.125ms	500	65036	0xFE0C	0xFE0C

12.3.5 Mode 1 Pulse Width Measurement PWC

In this mode, the high level and low level or period width of the input pulse can be automatically measured.

The first valid edge counter initializes to 0x0001, the second valid edge will stop counting, store the current count value in CCR0A, generate a capture interrupt CA0F, and if the counter overflows, an overflow flag will be raised. Setting overflow interrupt enable will generate overflow interrupt.

M1CR.EDG1ST	0	0	1	1
M1CR.EDG2ND	0	1	0	1

Pulse width measurement	Upper edge-upper edge Cycle width	Upper edge-lower edge High level width	Lower edge-upper edge Low level width	Lower edge-lower edge Cycle width
-------------------------	--------------------------------------	---	--	--------------------------------------

During period measurement, a period is measured at intervals of one period.

PWC Function Block Diagram

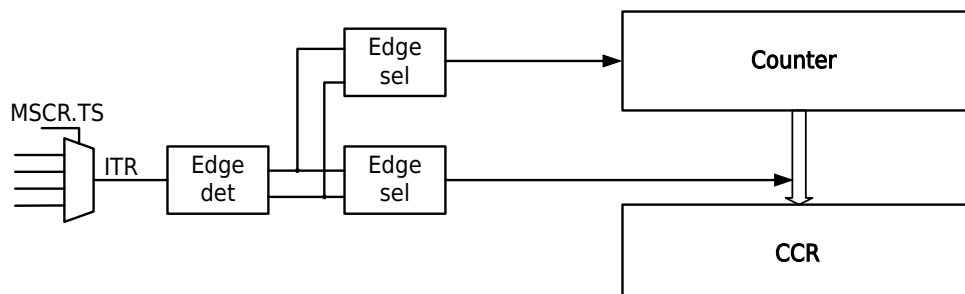


Figure 12-8 Mode1 Pulse Width Measurement PWC Function Block Diagram

Select the measurement signal source by registering MSCR.TS.

Trigger Selection	Value	Signal Source
MSCR.TS	0b000	ETFP: ETR filtered external input phase selection signal, selectable between external filtered and input inverted phase
	0b001	ITR0: Timer internal interconnection signal 0, TRGO output from other timers
	0b010	ITR1: Timer internal interconnection signal 1, TRGO output from other timers
	0b011	ITR2: Timer internal interconnection signal 2, TRGO output from other timers
	0b100	ITR3: Timer internal interconnection signal 3, TRGO output from other timers
	0b101	IA0ED: Invalid
	0b110	IAFP: CH0A filtered external input phase selection signal, selectable between external filtered and input inverted phase (polarity selection invalid in pulse width measurement mode)
	0b111	IBFP: CH0B filtered external input phase selection signal, selectable between external filtered and input inverted phase (polarity selection invalid in pulse width measurement mode)

For TRGO output, refer to the register MSCR description.

Timer	ITR0	ITR1	ITR2	ITR3
ATIM3	-	-	GTIM_OV	-

PWC Waveform Measurement Timing Diagram

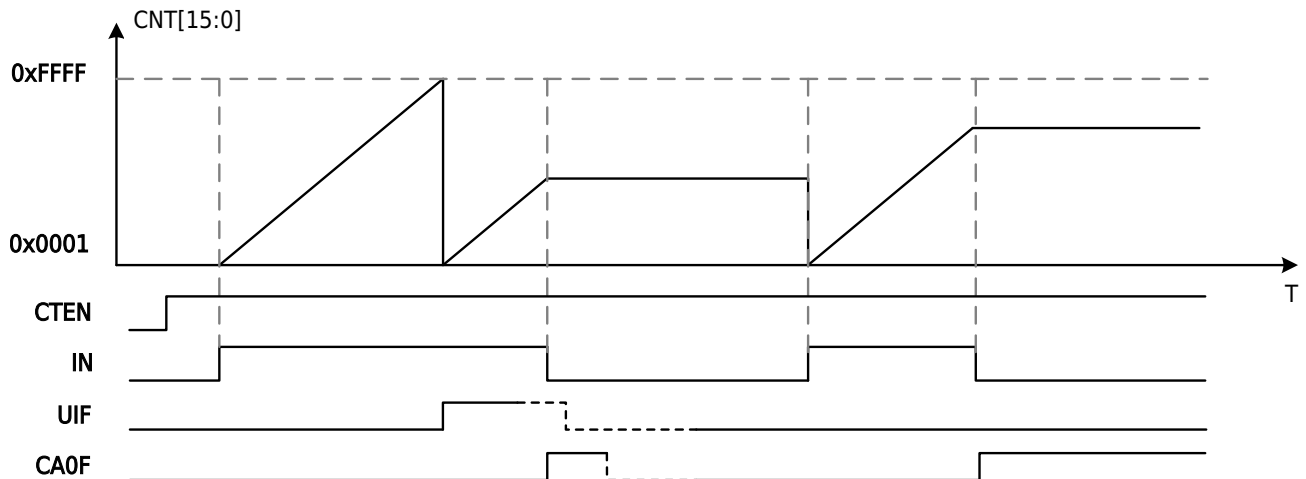


Figure 12-9 Mode1: Pulse Width Measurement PWC High-Level Pulse Width Measurement

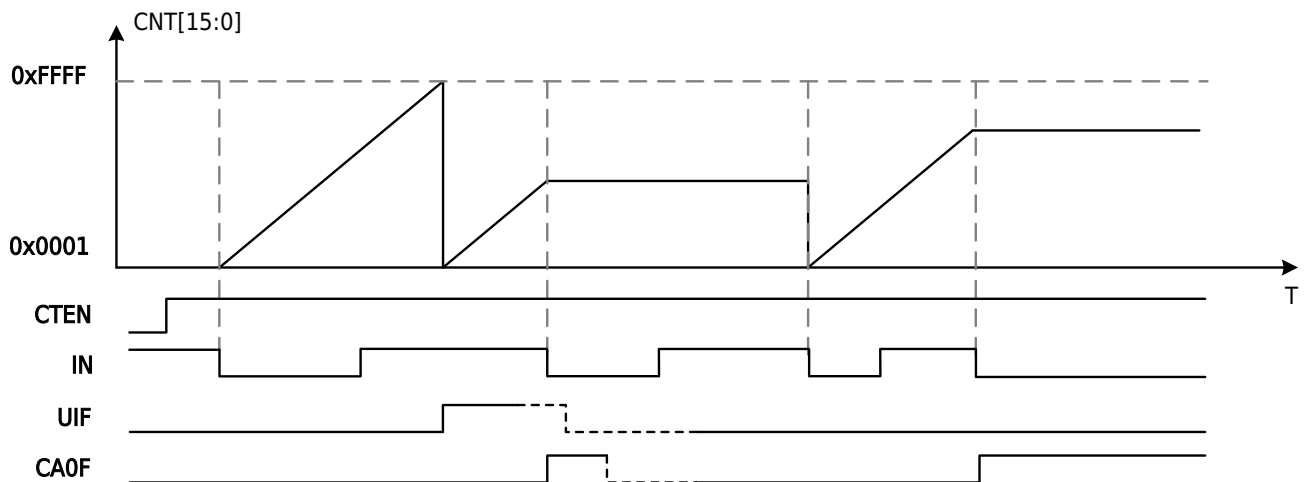


Figure 12-10 Mode1: Pulse Width Measurement PWC Falling Edge to Falling Edge Period Measurement

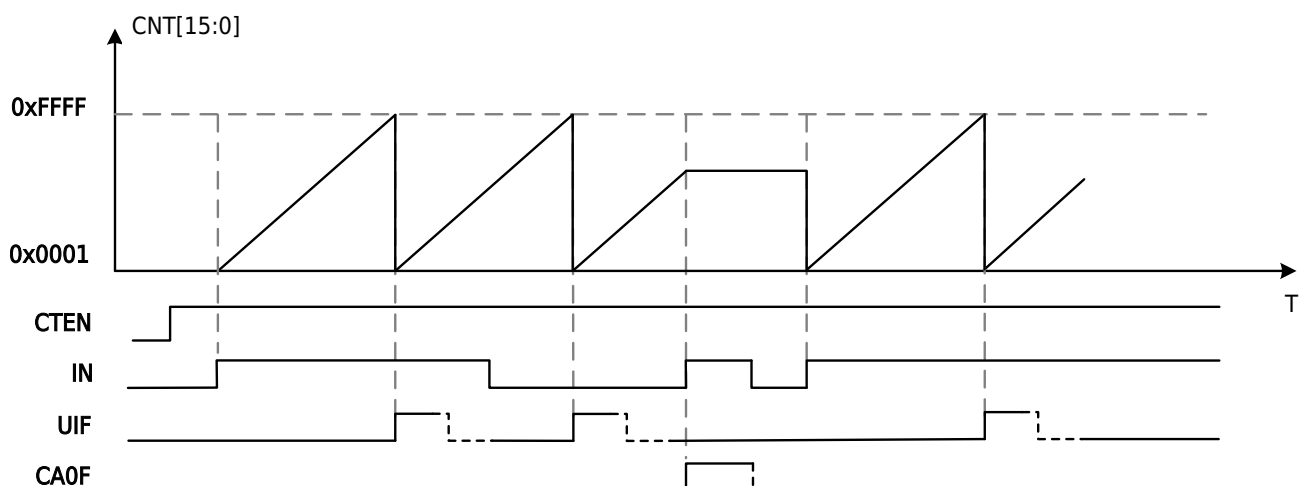


Figure 12-11 Mode1: Pulse Width Measurement PWC Rising Edge to Rising Edge Period Measurement

PWC Single trigger Mode

Set M1CR.ONESHOT=1 to set PWC single measurement, and CTEN will be cleared after the measurement is completed.

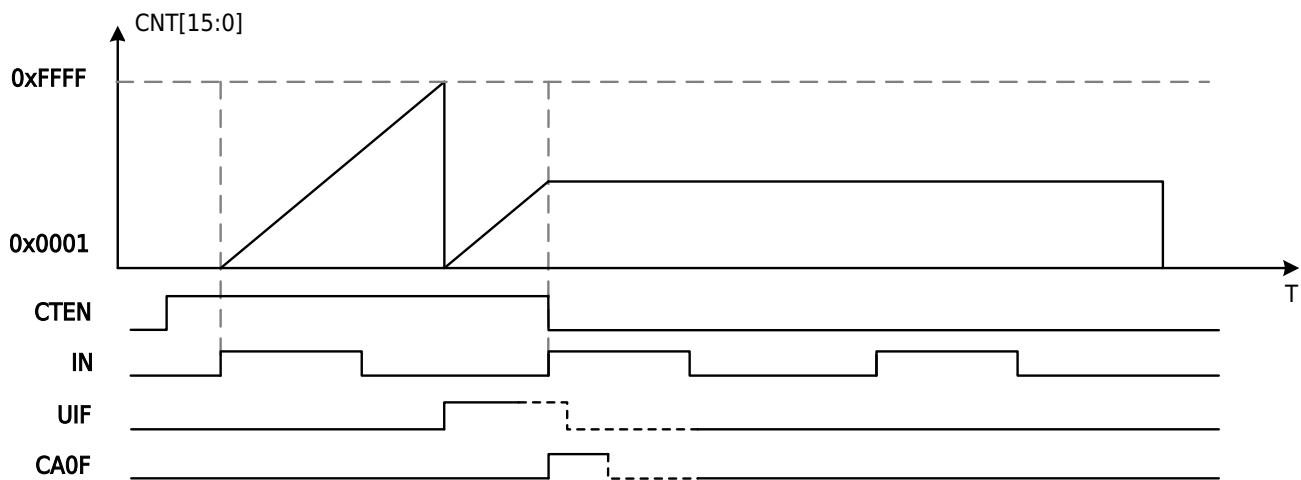


Figure 12-12 Model1: Pulse Width Measurement PWC Rising Edge to Rising Edge Period Measurement Single Mode

12.3.6 Mode2/3 Compare Capture Mode

12.3.6.1 Counter

The main part of the counter is a 16-bit counter with associated auto-loaded register. This counter can count up (mode 2), count down (mode 2) or count up and down in both directions (mode 3). The clock of the counter can be divided by the prescaler PRS, or ETR can be selected to input an external clock or the external input signal and internal interconnection signal selected by MSCR.TS.

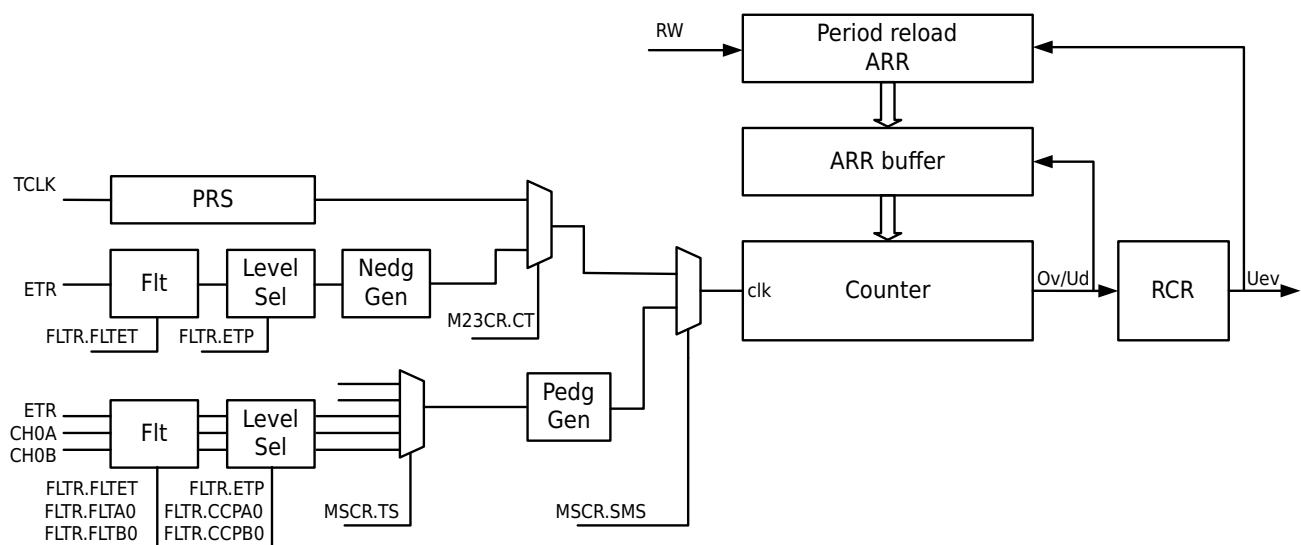


Figure 12-13 Mode 2/3 Compare Capture Mode Counter Block Diagram

The basic counter unit includes:

- Counter register CNT
- Prescaler register M23CR.PRS
- Auto-reload register ARR
- Repeat count register RCR
- Clock selection control registers FLTR, M23CR, MSCR

The auto-loaded register has a cache function, and the reload value is updated from the buffer register to the counter after the counter generates an event update. When the counter is in the stopped state or the cache function is disabled, the auto-reload register is immediately updated to the buffer register. When the timer is in the running state and the cache function is enabled, the value written to the

auto-reload register is not immediately updated to the buffer register. The auto-reload register updates to the buffer register only after an event update occurs.

For clock selection and gating functions, trigger functions, and reset functions, refer to the relevant descriptions in the [Mode2/3 Slave Mode](#) chapter.

12.3.6.2 Counter Waveform

Mode 2 Sawtooth Counting Waveform

Mode 2 operates as a sawtooth counting waveform, and the counting direction can be changed by configuring M23CR.DIR.

When M23CR.DIR is set to 0, the counter operates in up-counting mode. In this mode, the counter increments from 0 to the auto-reload value (ATIM3_ARR), then start counting from 0 again and generates a counter overflow event. If the repeat counter is used, an update event (UEV) is generated when the number of up-counting repetitions reaches the programmed value in the repeat counter register plus one (ATIM3_RCR+1). Otherwise, an update event is generated on every counter overflow.

An update event is also generated when M23CR.UG is set to 1 (either by software or using the slave mode controller).

When the update event occurs, all registers will be updated and the update flag (ATIM3_IFR.UIF bit) will be set to 1 (depending on the M23CR.URS bit):

- The auto-reload cache value will be updated with the ARR register value
- The compare buffer values will be updated with the compare register CCRxy.

The following diagrams illustrate the counter waveforms for different counting directions when ARR = 0x2C. The number of count overflow period clocks is ARR+1.

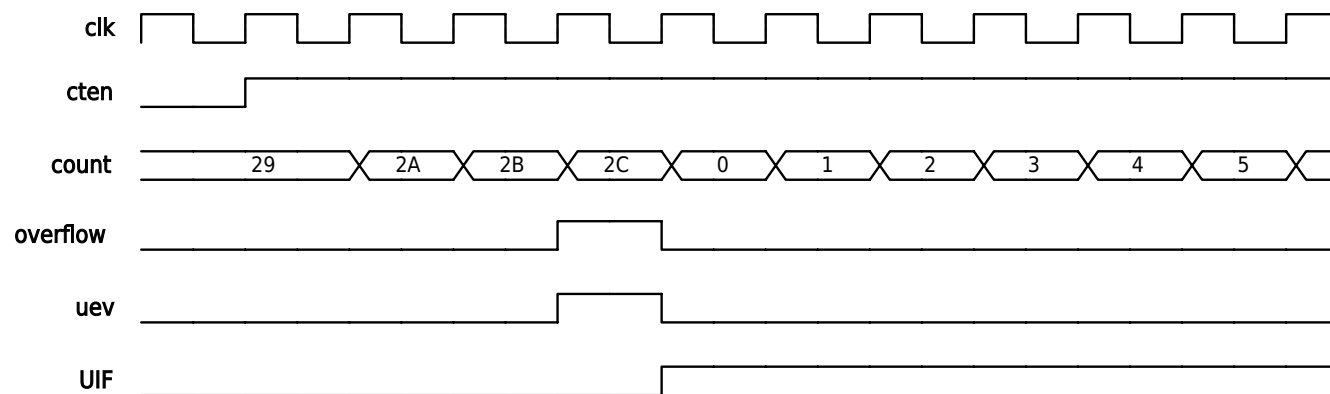


Figure 12-14 Mode 2 Compare Capture Mode Up-Counting Without Prescaler

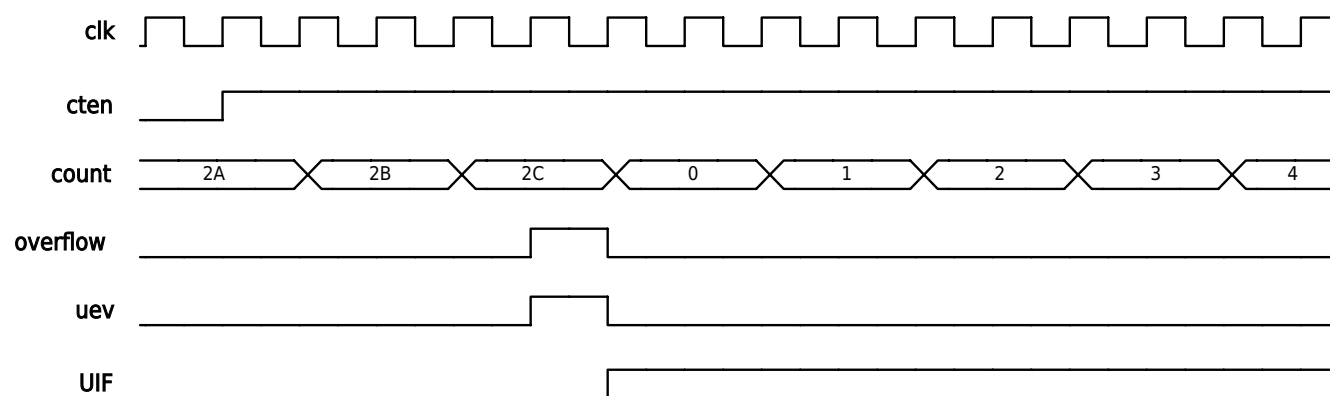


Figure 12-15 Mode 2 Compare Capture Mode Up-Counting With Prescaler

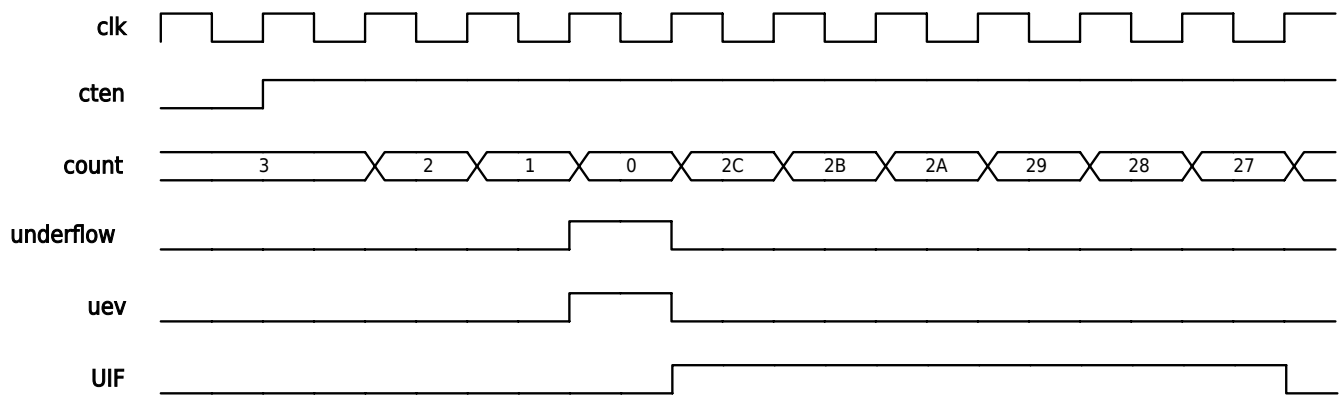


Figure 12-16 Mode 2 Compare Capture Mode Down-Counting Without Prescaler

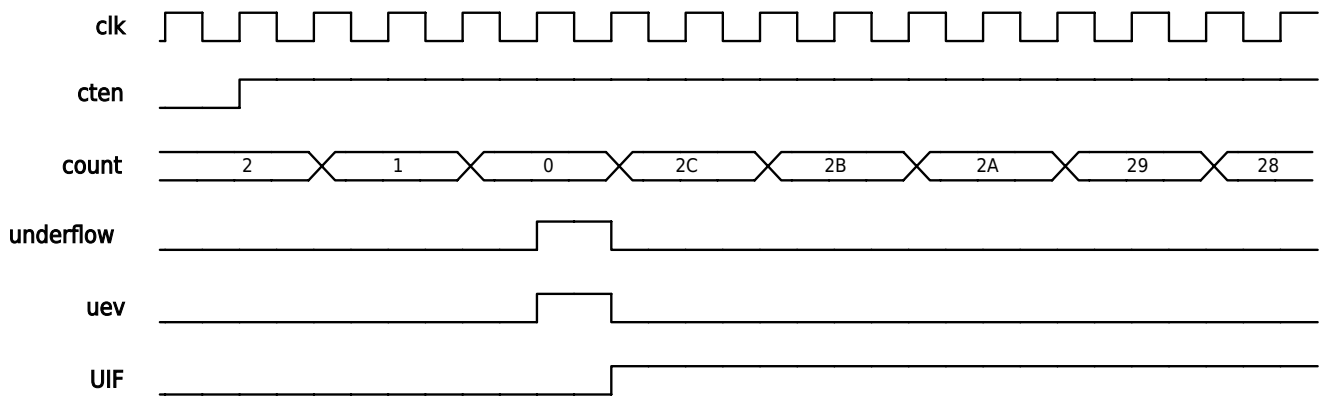


Figure 12-17 Mode 2 Compare Capture Mode Down-Counting With Prescaler

The counting waveforms for mode 2 are shown in the following diagrams:

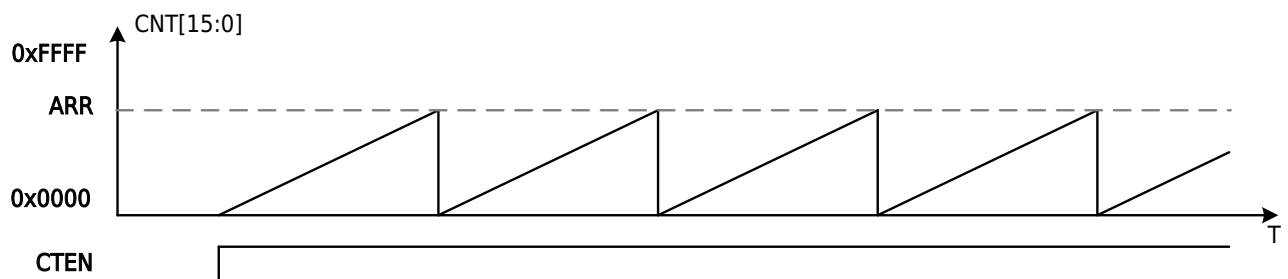


Figure 12-18 Mode2 Edge-Aligned Timer Waveform (M23CR.DIR=0)

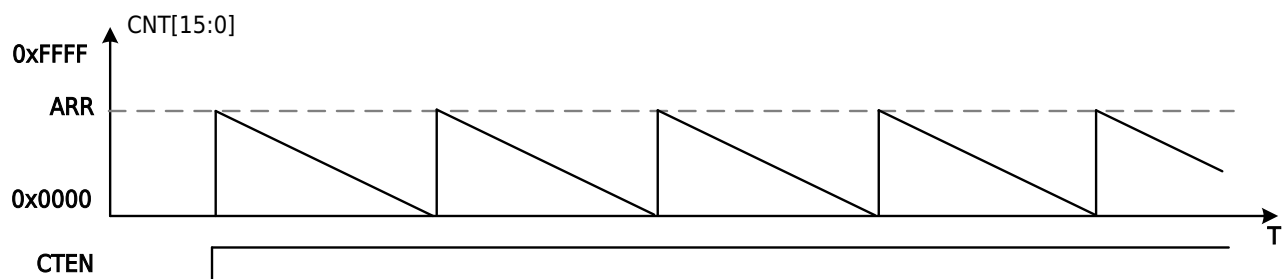


Figure 12-19 Mode2 Edge-Aligned Timer Waveform (M23CR.DIR=1)

Mode3 Triangular Counting Waveform

Mode 3 is a triangular counting waveform, the counting direction control bit is read-only, and the counting direction cannot be changed.

In center-aligned (triangular wave) mode, the M23CR.DIR direction bit is read-only, and writing to it has no effect. Switching from other modes to center-aligned mode DIR is automatically cleared to 0. DIR is automatically cleared in reset mode by software event update and external trigger in slave mode.

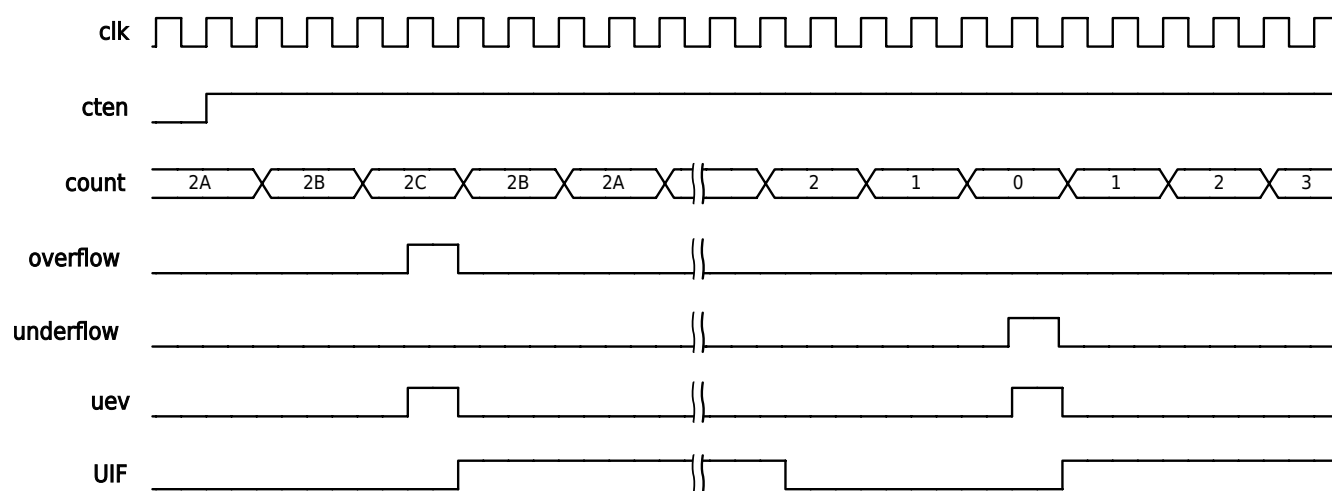


Figure 12-20 Mode3 Compare Capture Mode Up/Down-Counting with Prescaler

The counting waveforms for mode 3 are shown in the following diagrams:

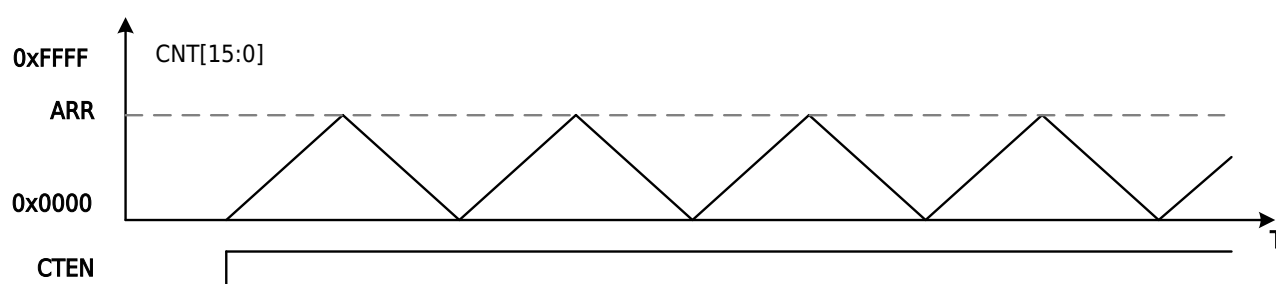


Figure 12-21 Mode3 Center-Aligned Counter Waveform

12.3.6.3 Repeat Counting

The repeat counter performs down-counting using counter overflows. When counting reaches 0, that is, when the counter overflows by the value set in the repeat register plus one. If buffer register is enabled, the period reload register updates to the period buffer register. In compare mode, the compare register value updates to the compare buffer register.

The repetition counter decrements when the following conditions are met:

- In up-counting mode: upon each counter overflow
- In down-counting mode: upon each counter underflow
- In triangular wave mode: upon each overflow and underflow

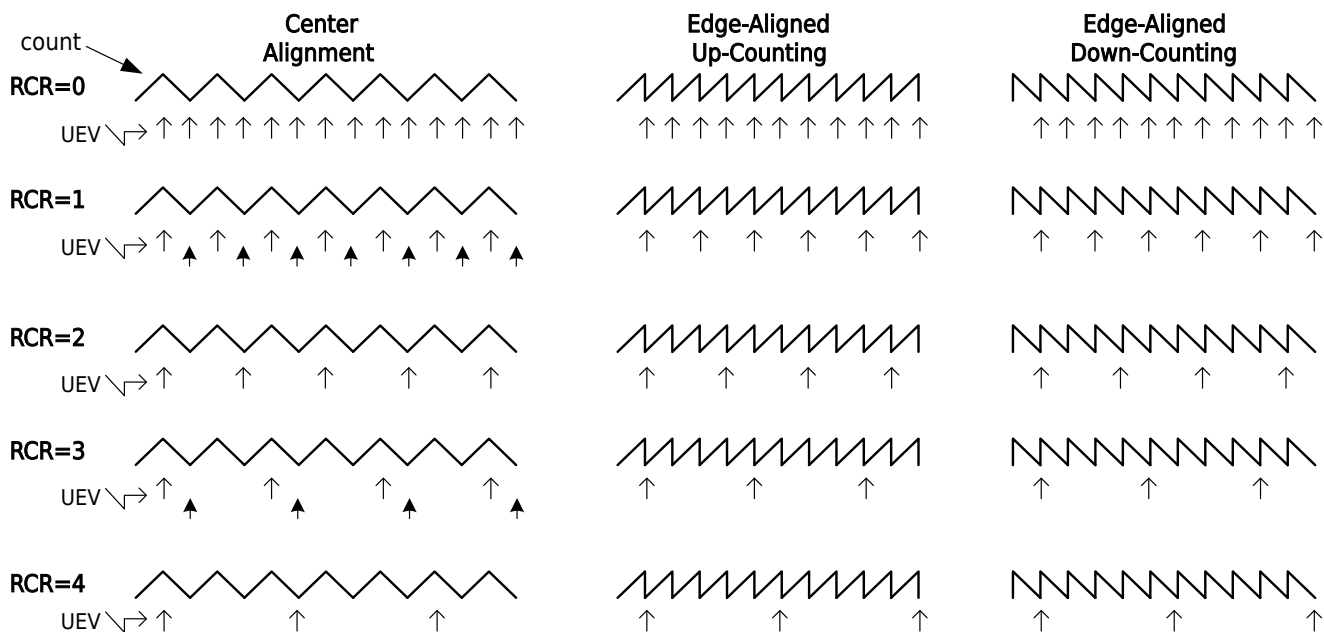


Figure 12-22 Repetition Counter Generates Update Timing (RCR.OV=0, RCR.UND=0)

RCR.OV is the overflow mask control bit for the repeat counter. When RCR.OV=1, overflow events are ignored during repeat counting.

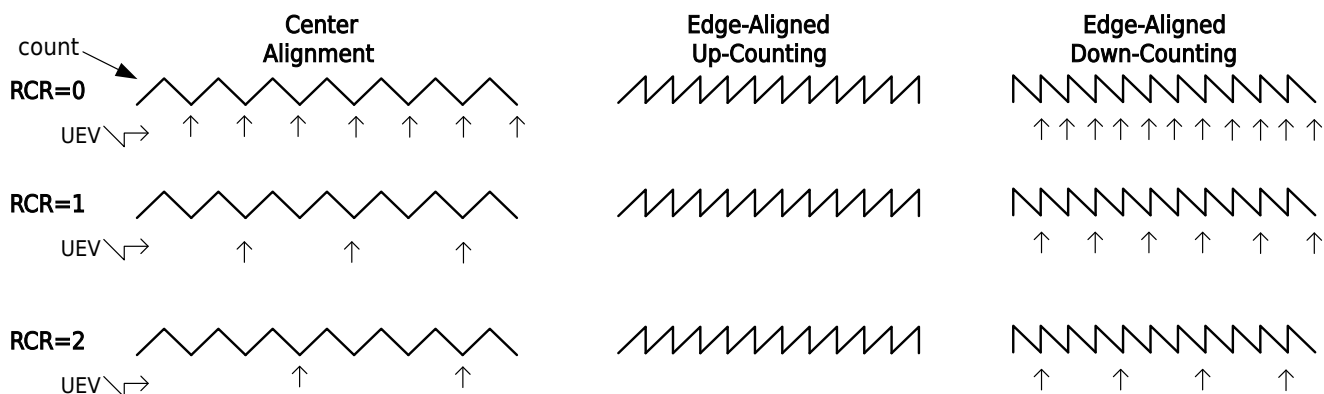


Figure 12-23 Update Timing Generated by Repetition Counter when Overflow Is Masked (RCR.OV=1, RCR.UND=0)

RCR.UND is the underflow mask control bit for the repetition counter. If RCR.UND is 1, underflow events will be ignored during repetition counting.

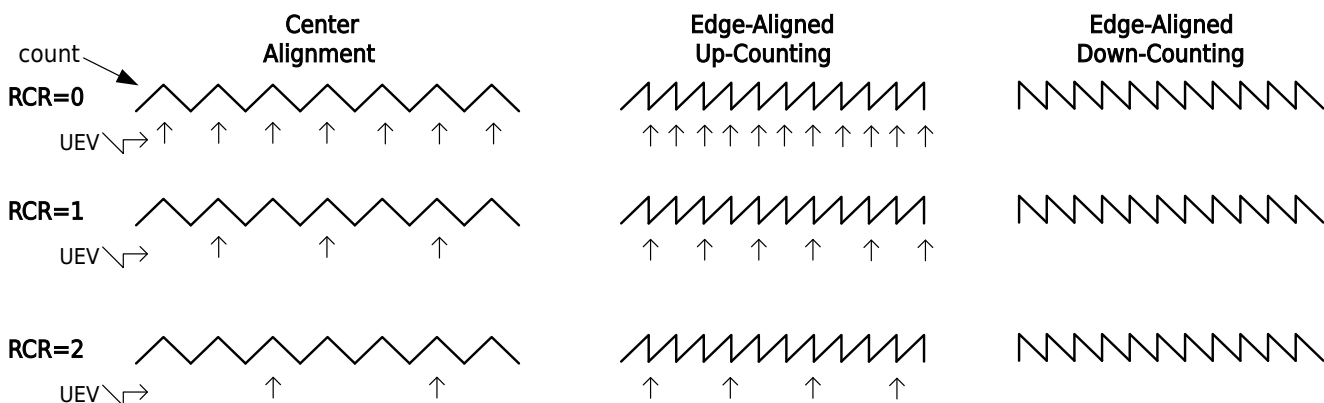


Figure 12-24 Update Timing Generated by Repetition Counter when Underflow Is Masked (RCR.OV=0, RCR.UND=1)

In addition to events update (UEV) generated by overflow/underflow through the repeat counter, software-triggered and slave-mode reset events update (UEV) can also be generated by writing to register M23CR.UG. This operation requires configuration of M23CR.URS.

12.3.6.4 Data Buffer

Both the auto-reload data (ARR) and compare registers (CCRxy) can be configured with buffering function. When buffering function is enabled, the written period value (ARR) and compare value (CCRxy) only take effect upon a UEV event update.

The update timing diagrams for the auto-reload value under different counting modes are shown below:

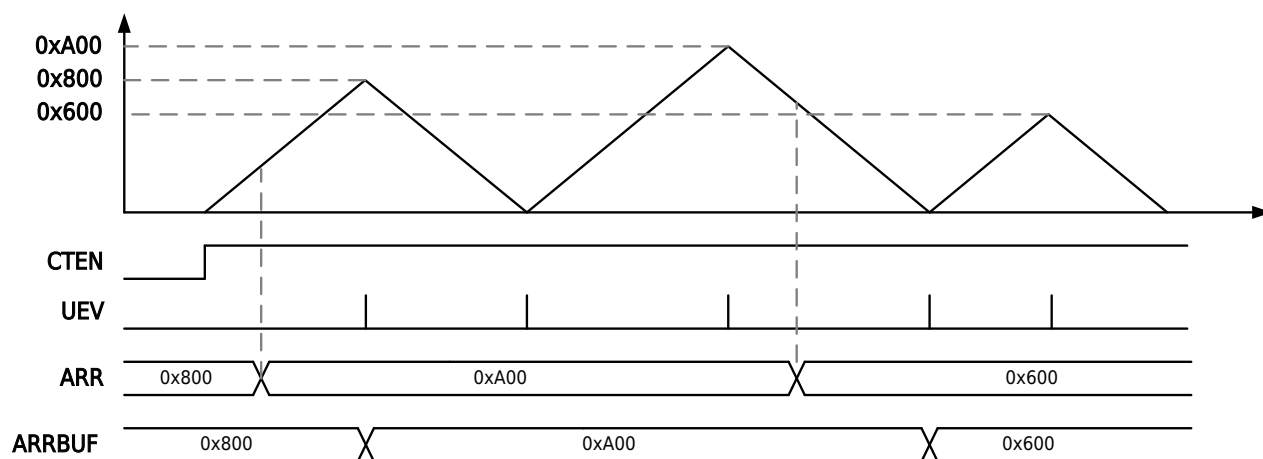


Figure 12-25 Mode 3 Triangular Wave Mode with Cache Enable

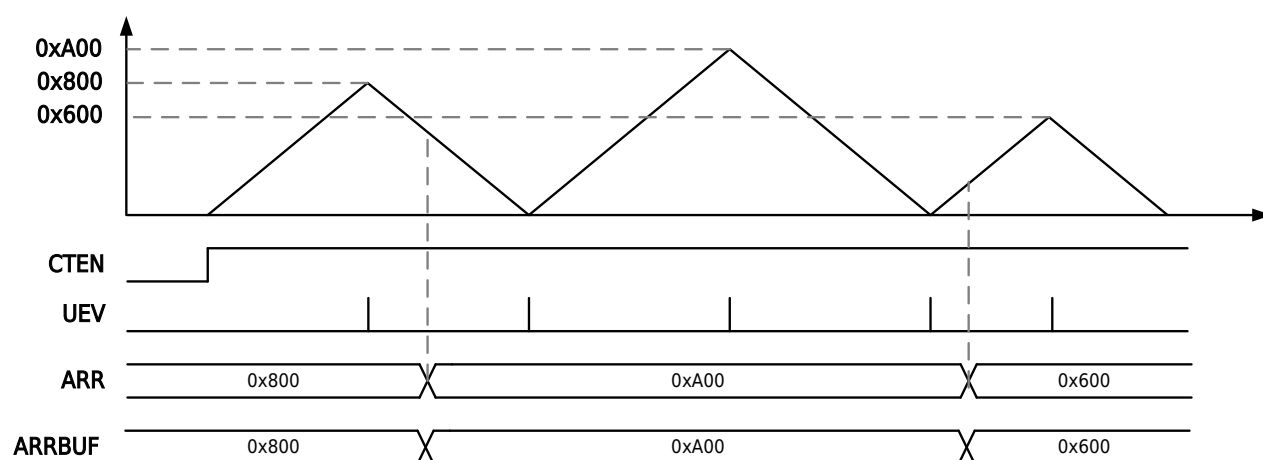


Figure 12-26 Mode 3 Triangular Wave Mode with Cache Invalid

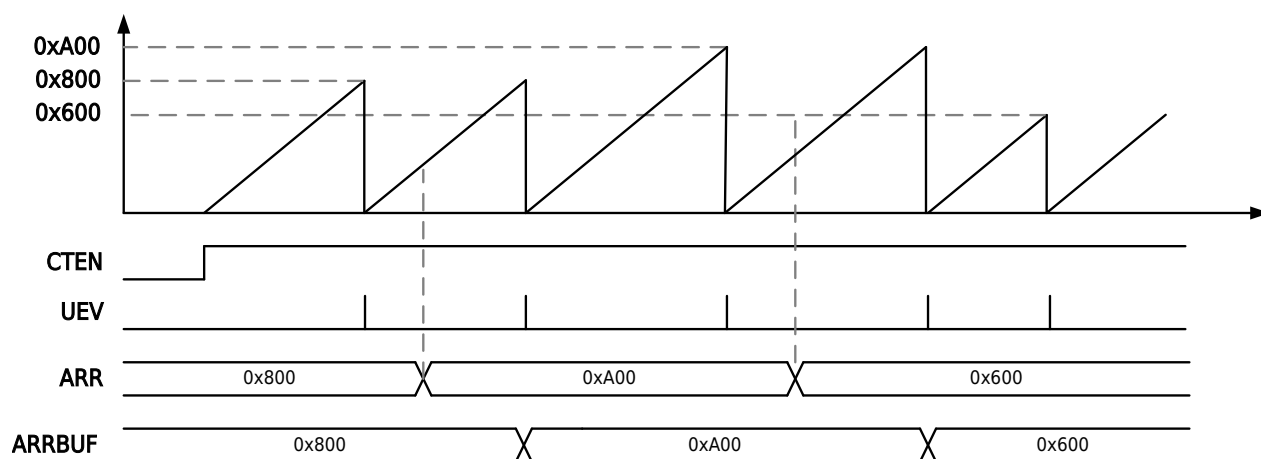


Figure 12-27 Mode 2 Sawtooth Wave Mode Up-Counting with Cache Enable

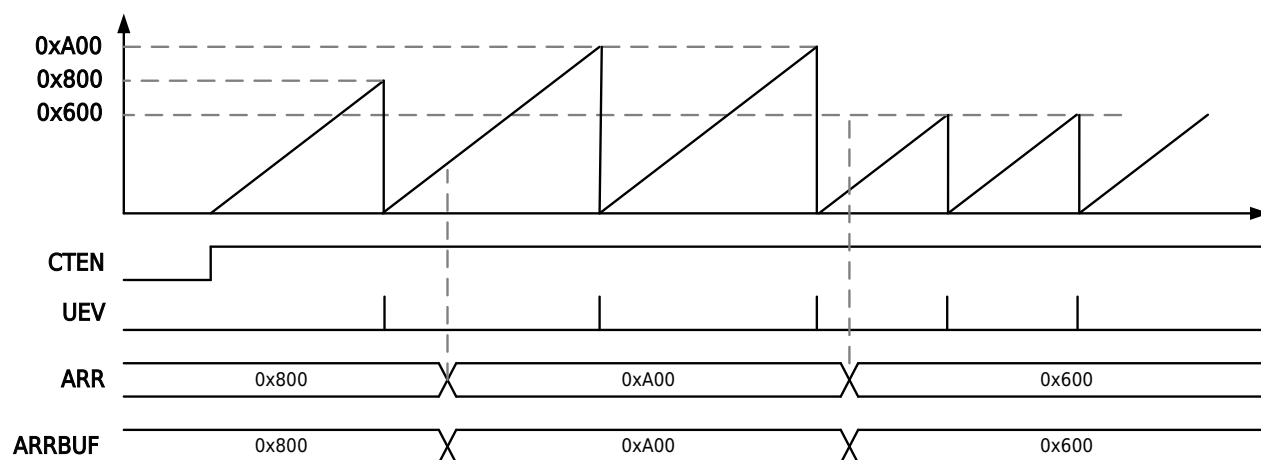


Figure 12-28 Mode 2 Sawtooth Wave Mode Up-Counting with Cache Invalid

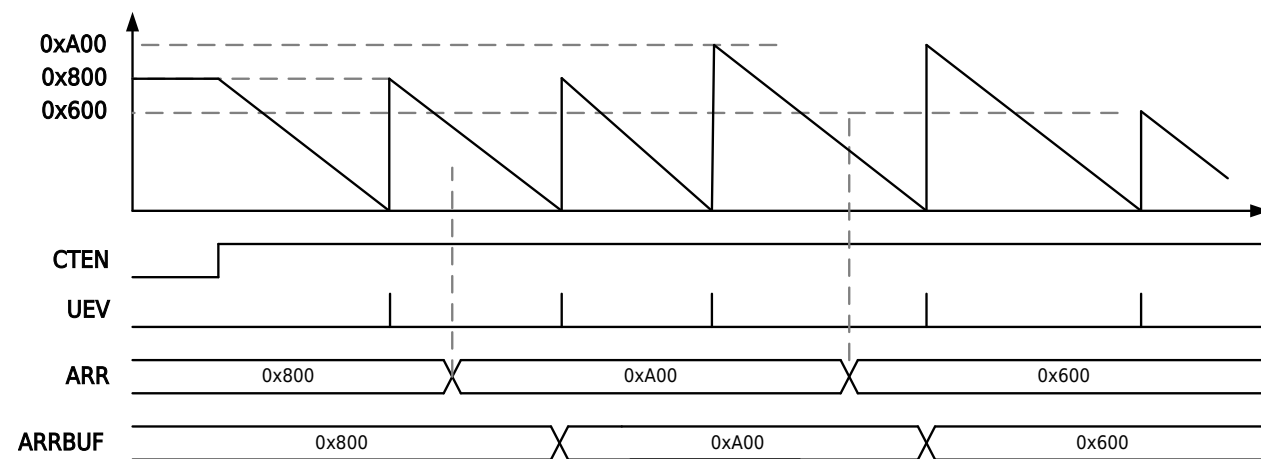


Figure 12-29 Mode 2 Sawtooth Wave Mode Down-Counting with Cache Enable

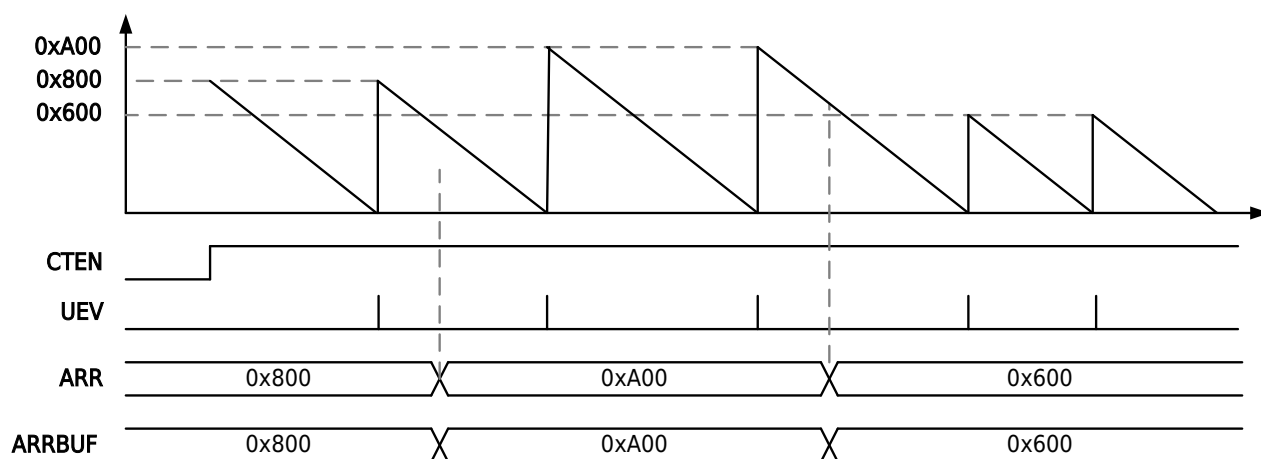


Figure 12-30 Mode 2 Sawtooth Wave Mode Down-Counting with Cache Invalid

**Note**

In triangular wave mode and sawtooth up-counting mode, when cache is disabled and the ARR value is modified, the current counter value must be less than the ARR period value to be changed; otherwise, the current cycle will count up to 0xFFFF.

The update behavior of the compare cache and period cache remains consistent, therefore their timing diagrams are not listed individually here.

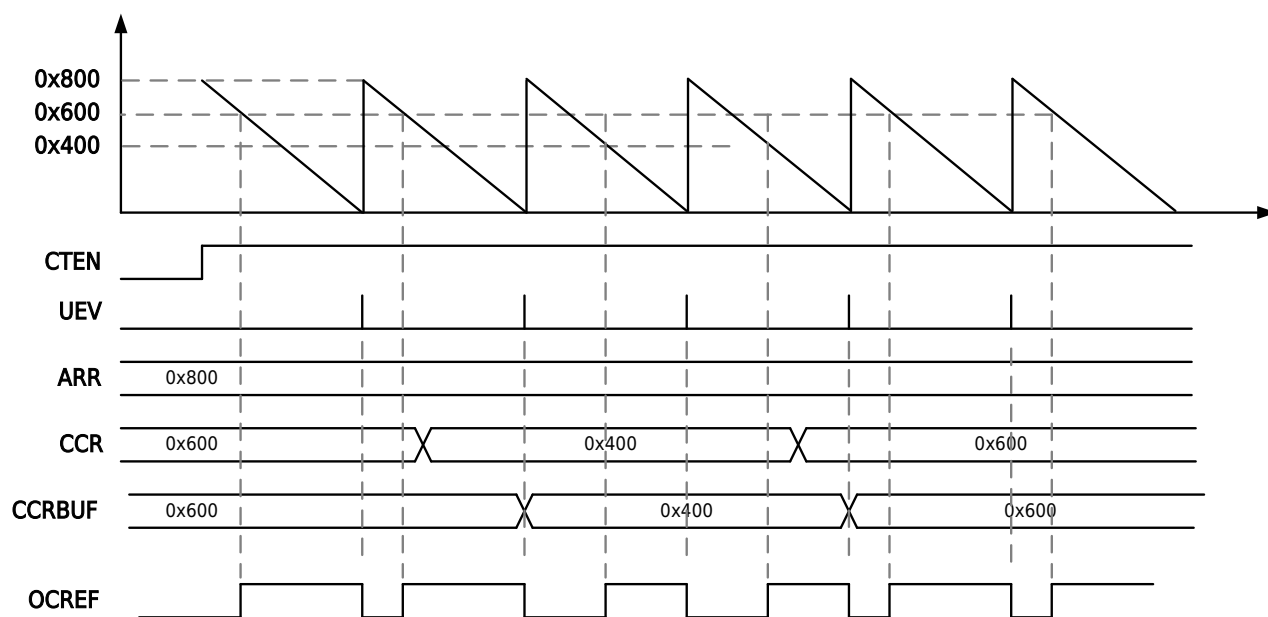


Figure 12-31 Mode2 Sawtooth Wave Down-Counting Mode with Compare Cache Enable

12.3.6.5 Compare Output OCREF

The compare output OCREFA can be configured for single point comparison, where the OCREFA output is controlled by compare register CCRA. Alternatively, OCREFA compare output can be configured for dual point comparison, where its output is jointly controlled by compare registers CCRA and CCRB.

OCREFB can only use single point comparison, and the comparison register CCRB is used to control the comparison output of OCREFB.

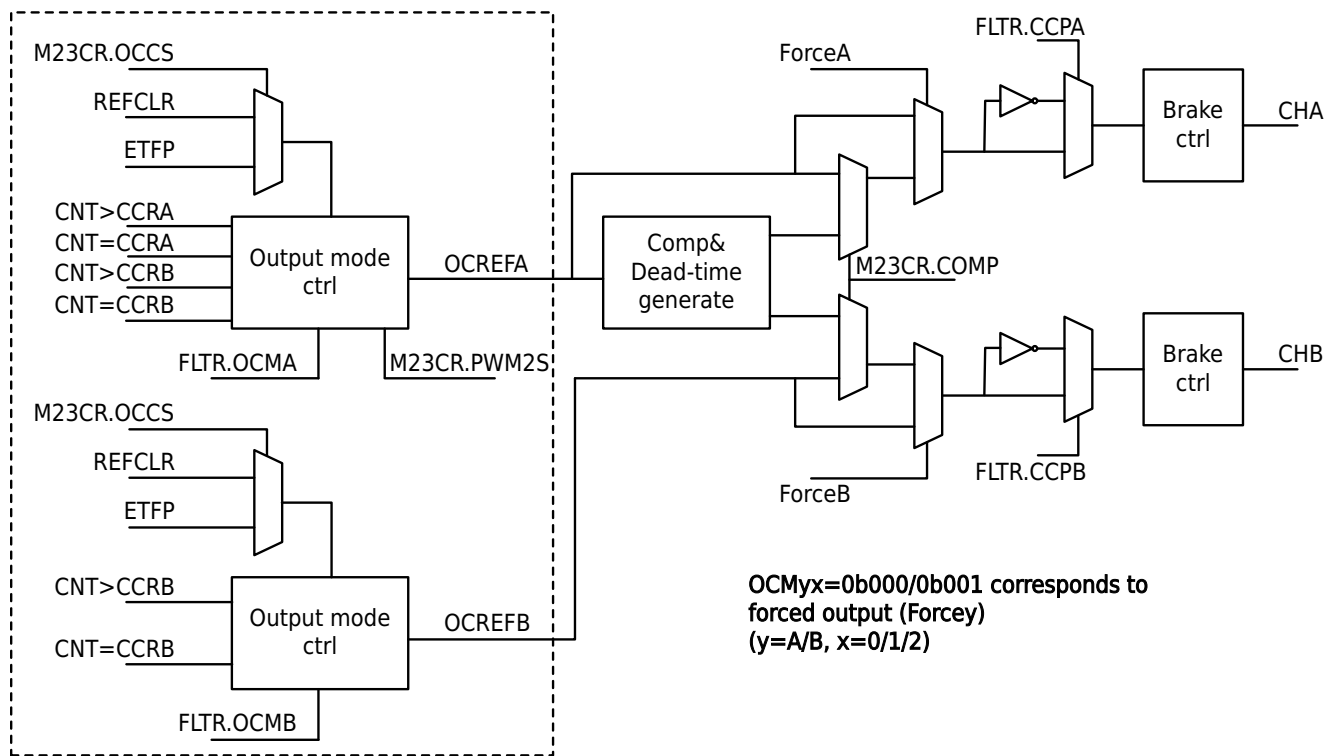


Figure 12-32 OCREF Output Block Diagram

OCREF output uses ATIM3_FLTR.OCMyx selection:

- 0b000: Force to 0
- 0b001: Force to 1
- 0b010: Force to 0 on compare match
- 0b011: Force to 1 on compare match
- 0b100: Toggle on compare match
- 0b101: Output high level for one clock cycle on compare match
- 0b110: PWM mode 1
 - ▶ Single point comparison

During up-counting, the output is high level when $CNT < CCR_{xy}$; during down-counting, the output is low level when $CNT > CCR_{xy}$.
 - ▶ Dual point comparison
 - ◆ For sawtooth wave up-counting, the output is low level when $CCR_{xA} < CNT \leq CCR_{xB}$
 - ◆ For sawtooth wave down-counting, the output is high level when $CCR_{xA} < CNT \leq CCR_{xB}$
 - ◆ For triangular wave up-counting, the output is high level when $CNT < CCR_{xA}$; for down-counting, the output is low level when $CNT > CCR_{xB}$
- 0b111: PWM mode 2
 - ▶ Single point comparison

During up-counting, the output is low level when $CNT < CCR_{xy}$; during down-counting, the output is high level when $CNT > CCR_{xy}$
 - ▶ Dual point comparison
 - ◆ For sawtooth wave up-counting, the output is high level when $CCR_{xA} \leq CNT < CCR_{xB}$
 - ◆ For sawtooth wave down-counting, the output is low level when $CCR_{xA} \leq CNT < CCR_{xB}$
 - ◆ For triangular wave up-counting, the output is low level when $CNT < CCR_{xA}$; for down-counting, the output is high level when $CNT > CCR_{xB}$

**Note**

Forced output has higher priority. When forced output is active, complementary output control becomes invalid.

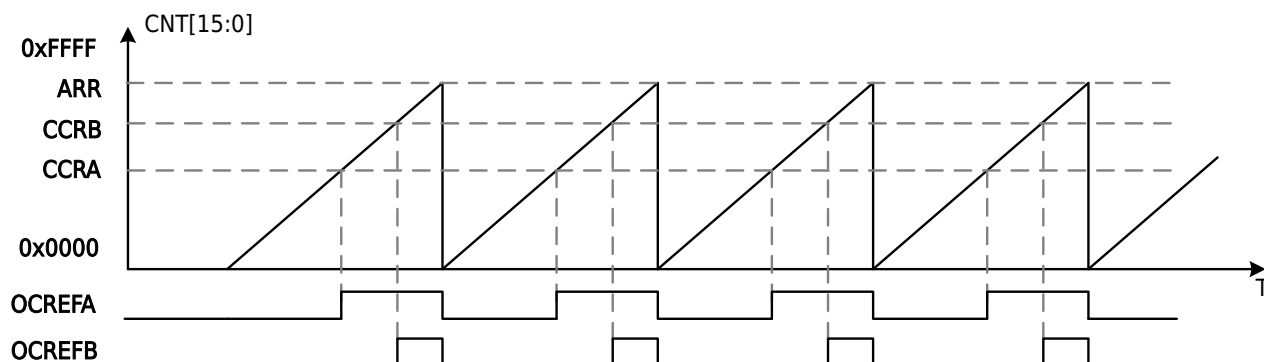


Figure 12-33 Sawtooth Counting Single Point Comparison OCREF Output Waveform (OCMyx=0b111)

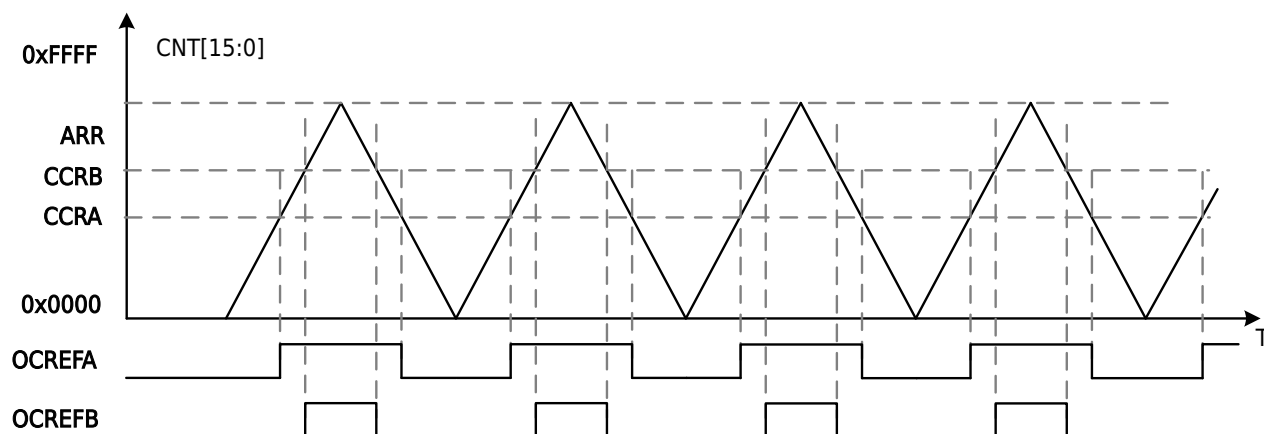


Figure 12-34 Triangular Counting Single Point Comparison OCREF Output Waveform (OCMyx=0b111)

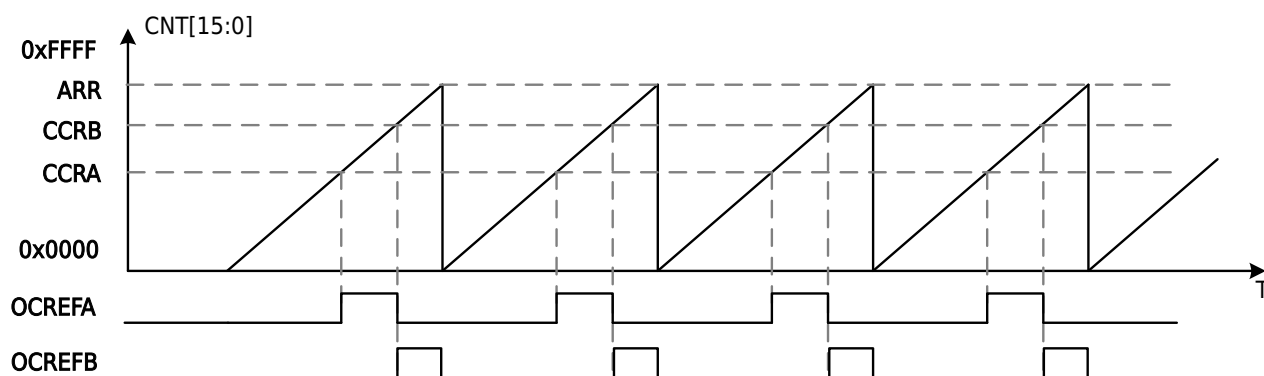


Figure 12-35 Sawtooth Counting Dual Point Comparison OCREF Output Waveform (OCMyx=0b111)

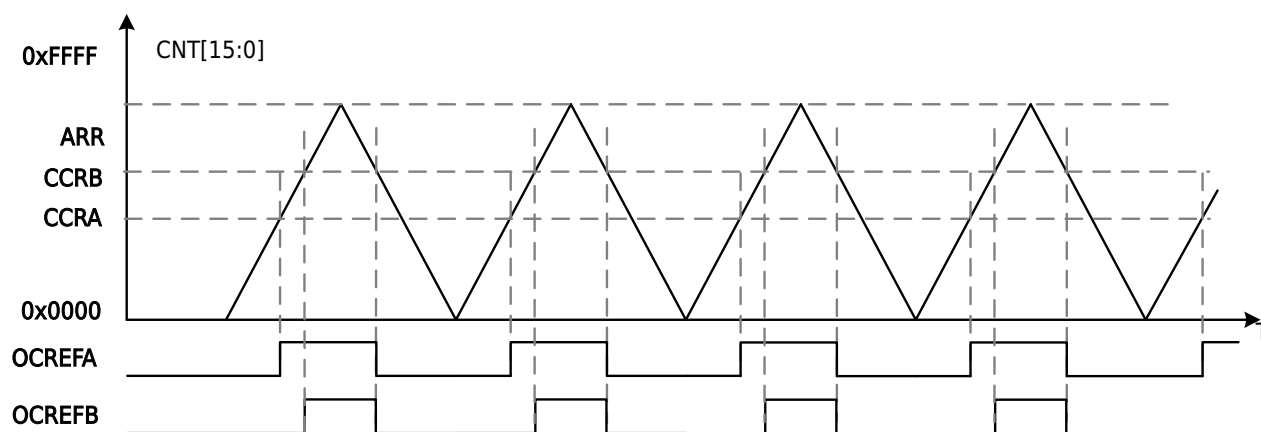


Figure 12-36 Triangular Counting Dual Point Comparison OCREF Output Waveform (OCMyx=0b111)

12.3.6.6 Independent PWM Output

The output of CHA is controlled by OCREFA, and the output of CHB is controlled by OCREFB. The inversion of CHA and CHB outputs can be controlled through FLTR.CCPA and FLTR.CCPB respectively.

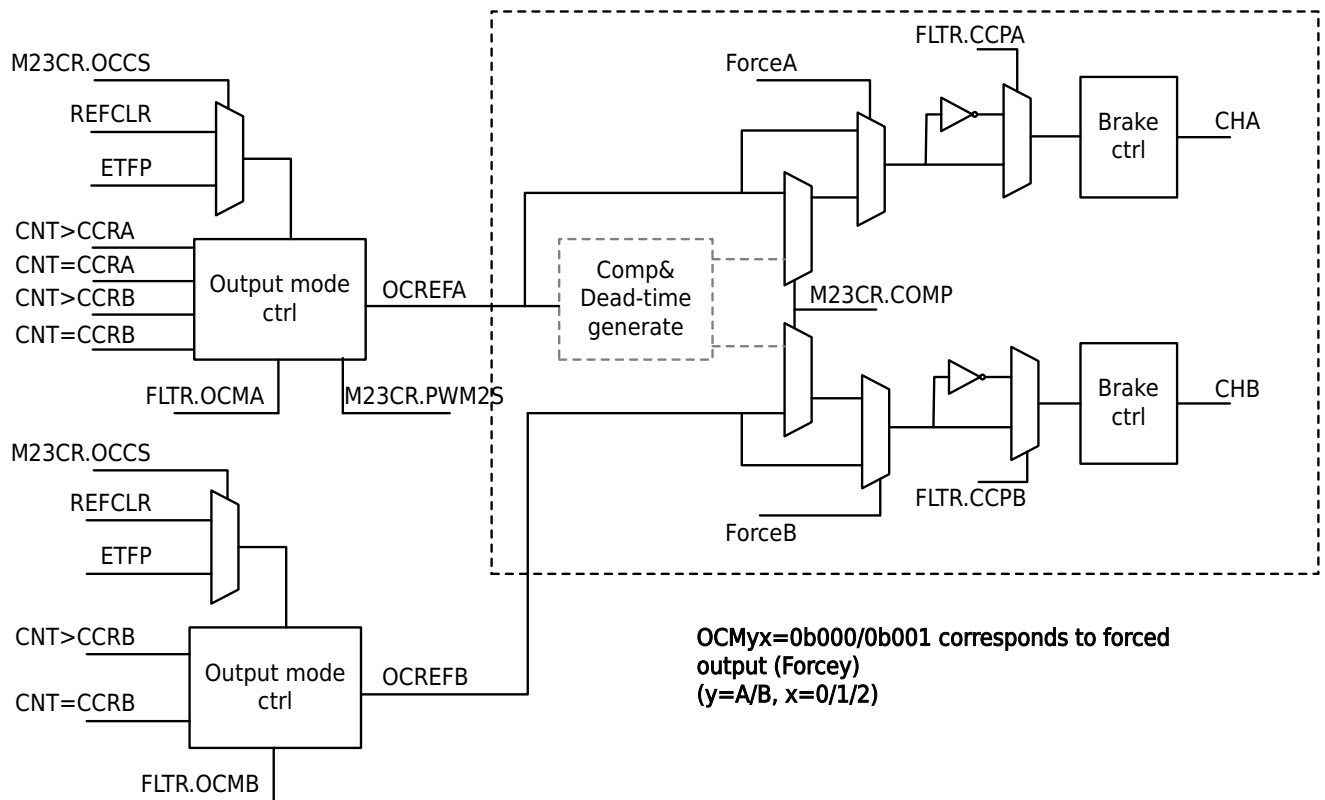


Figure 12-37 Independent PWM Output Block Diagram

PWM output and OCREF relationship, as shown in the following diagram:

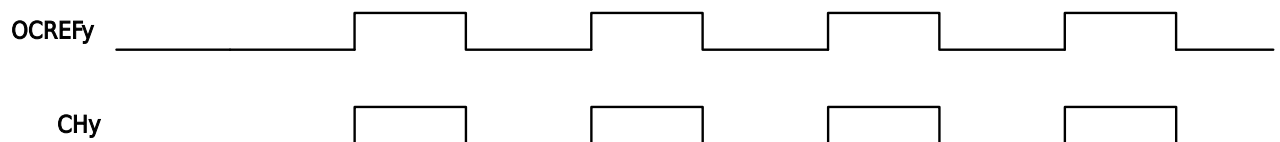


Figure 12-38 PWM Output Waveform When CCPy=0

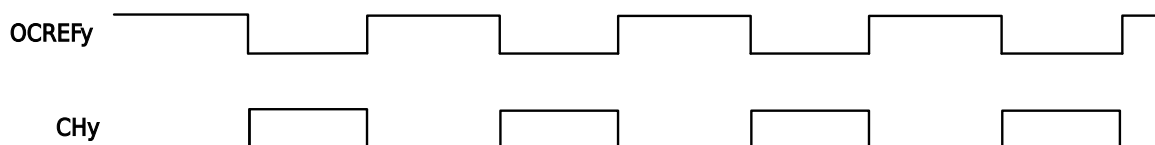


Figure 12-39 PWM Output Waveform When CCPy=1

12.3.6.7 Complementary PWM Output

CHA is controlled by OCREFA, and OCREFA controls the output of CHB at the same time. The compare register CCRxB can be used as a dedicated compare control ADC trigger.

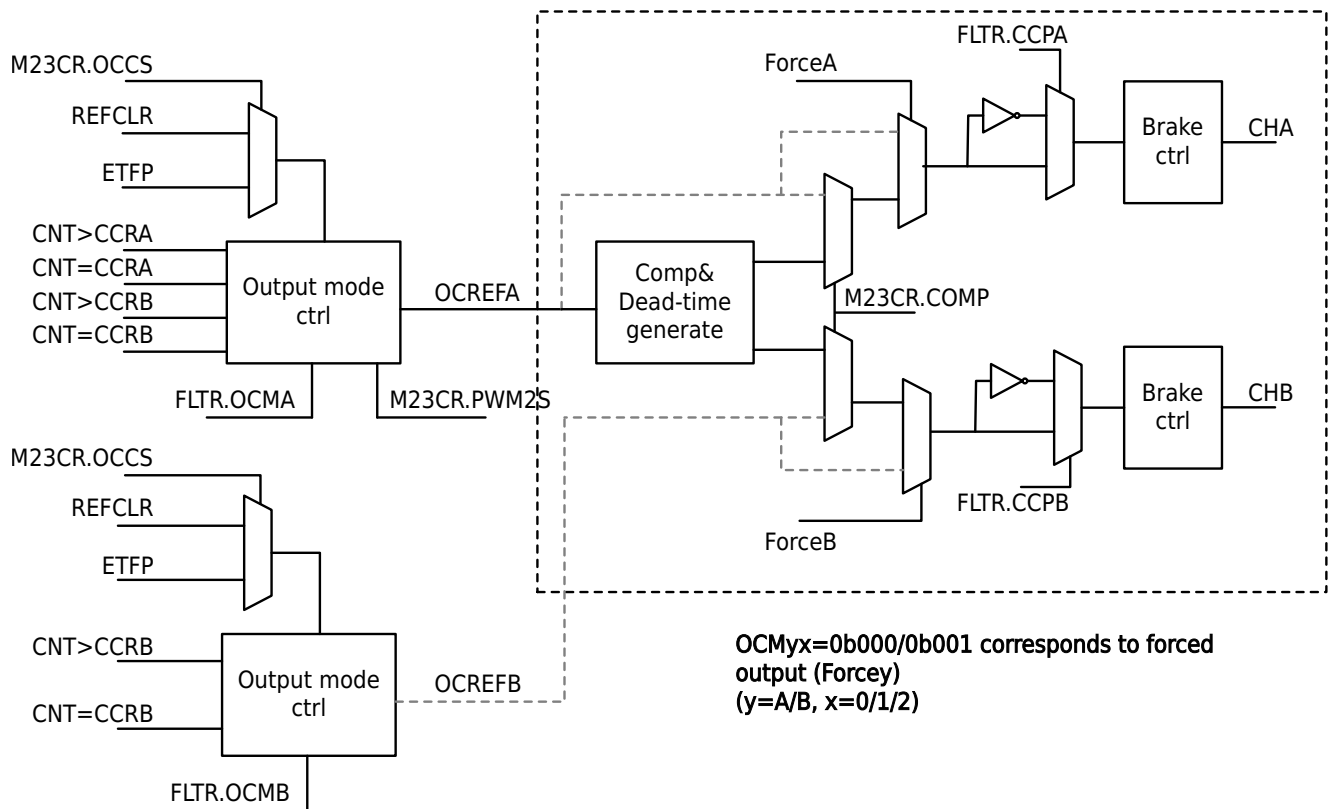


Figure 12-40 Complementary PWM Output Block Diagram

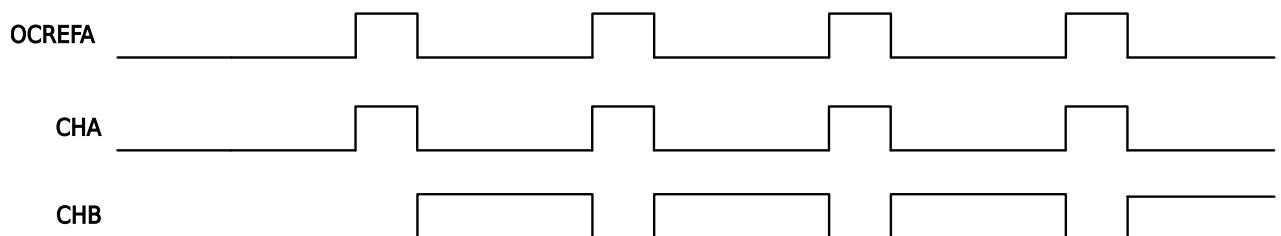


Figure 12-41 Complementary PWM Output Waveform

12.3.6.8 PWM Output with Dead Time

The dead time function can be set in complementary PWM output mode.

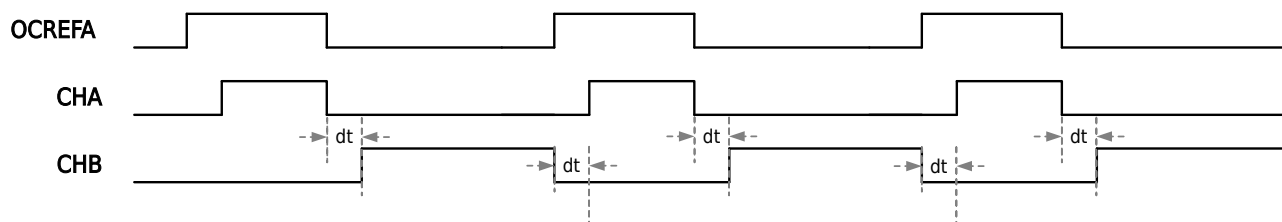


Figure 12-42 Complementary PWM Output Waveform (with Dead Time)

The dead time duration is controlled by 8-bit DTR. The relationship between dead time dt and DTR is as follows:

DTR	T	dt	step
DTR[7]=0b0	$DTR[6:0] + 2$	2-129	1
DTR[7:6]=0b10	$\{DTR[5:0] + 64\} * 2 + 2$	130-256	2
DTR[7:5]=0b110	$\{DTR[4:0] + 32\} * 8 + 2$	258-506	8
DTR[7:5]=0b111	$\{DTR[4:0] + 32\} * 16 + 2$	514-1010	16

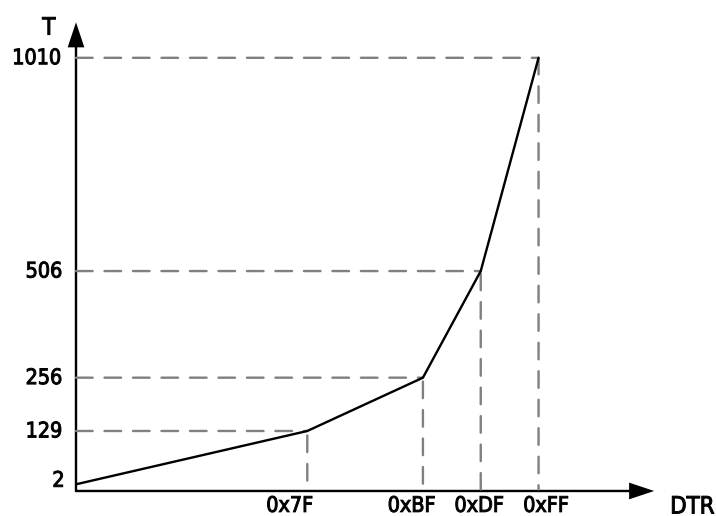


Figure 12-43 Dead Time



Note

When TCLK is 48MHz and M23CR.PRS is set to 0, the dead time is adjusted from 5.33μs to 21.04μs.

12.3.6.9 Single Pulse Output

Single pulse mode (ONESHOT) is a special case of the aforementioned PWM modes. In this mode, the counter can be triggered to start by an excitation signal and generate a programmable width pulse, as shown in the following diagram.

The counter can be started by the slave mode controller. Waveforms can be generated in either output compare mode or PWM mode. Setting the ATIM3_M23CR.ONESHOT register to 1 enables single pulse mode. Thus, when the next update event (UEV) occurs, the counter will automatically stop.

Single pulse mode in sawtooth wave down-counting mode, the counter initial value should not be set to 0, and the count value in up-counting mode should not be set to a value greater than or equal to ARR.

Output pulse can only be correctly generated when the compare value differs from the counter initial value. Before starting (when the timer is waiting for a trigger), the following configurations must be made:

- For increment counting: $CNT < CCR_{xy} \leq ARR$ (special attention: $CCR_{xy} > 0$)
- For decrement counting: $CNT > CCR_{xy}$

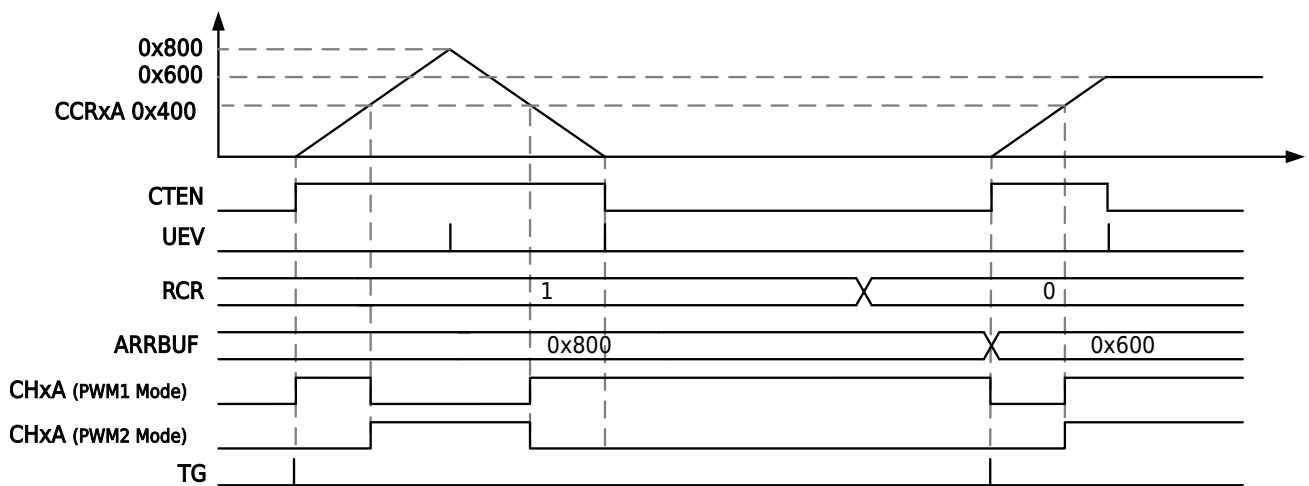


Figure 12-44 Triangular Wave Mode Single Pulse Counting

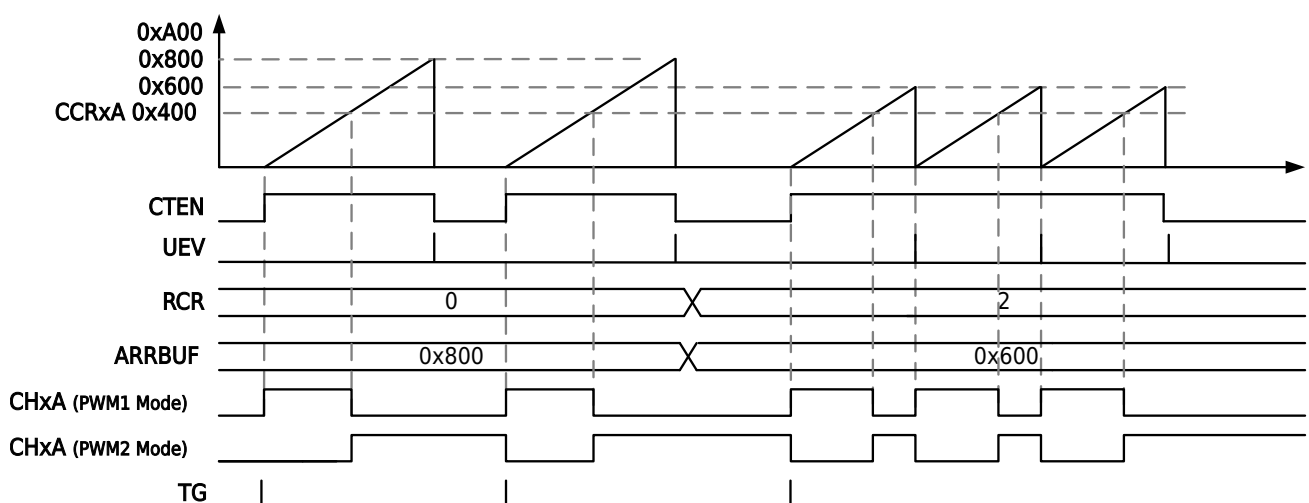


Figure 12-45 Sawtooth Up-Counting Single Pulse Mode

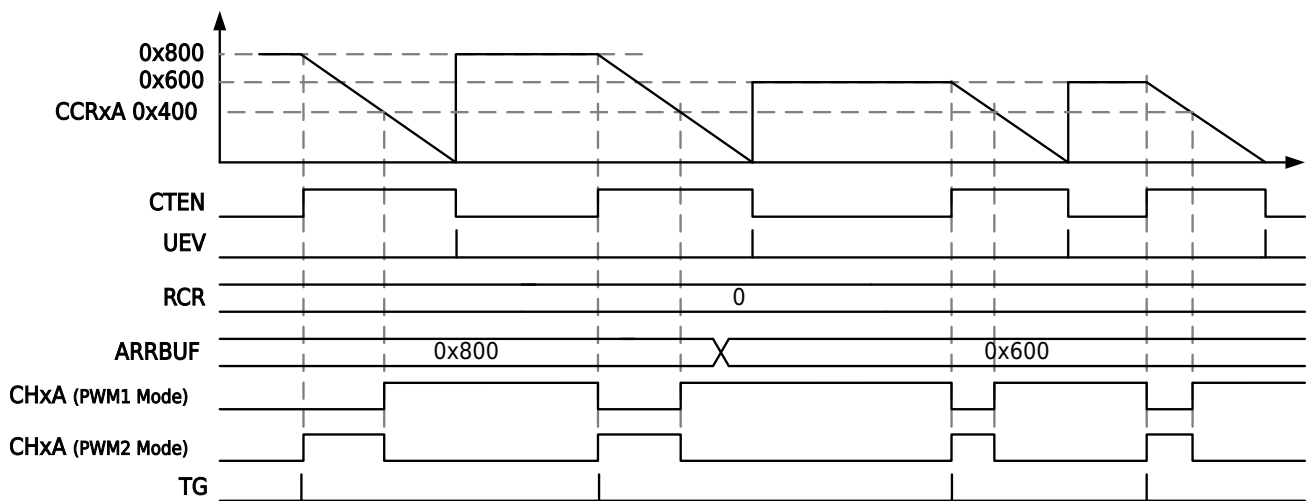


Figure 12-46 Sawtooth Down-Counting Single Pulse Mode

**Notice**

In up-counting, when configured as PWM2 mode, since the channel initial output is low level, the first trigger will generate a rising edge and maintain a high level. Upon the second trigger, a programmable width low level pulse can be output. In down-counting, when configured as PWM1 mode, since the channel initial output is low level, the first trigger will generate a rising edge and maintain a high level. Upon the second trigger, a programmable width low level pulse can be output.

12.3.6.10 Compare Interrupt

Sawtooth wave compare match will set the corresponding flag in the IFR register to 1. If the interrupt enables CRCHx.CIEy (x=0/1/2; y=A/B), an interrupt will be triggered.

For triangular wave compare match, it can independently select rising count compare match, falling count compare match, or both.

For compare A match, when the count value equals CCRxA, it is uniformly controlled by M23CR.CIS:

- M23CR.CIS=0b00: no output on compare match
- M23CR.CIS=0b01: compare match during up-counting
- M23CR.CIS=0b10: compare match during down-counting
- M23CR.CIS=0b11: compare match during both up-counting and down-counting

For compare B match, when the count value equals CCRxB, each channel can be individually controlled by CRCHx.CISB:

- When CRCHx.CISB=0b00, channel x has no output on compare match.
- When CRCHx.CISB=0b01, channel x compare match during up-counting.
- When CRCHx.CISB=0b10, channel x compare match during down-counting.
- When CRCHx.CISB=0b11, channel x compare match during both up-counting and down-counting.

The independent control of channel B compare match enables more flexible ADC triggering. For more details, refer to the [Timer Trigger ADC](#) section in the documentation.

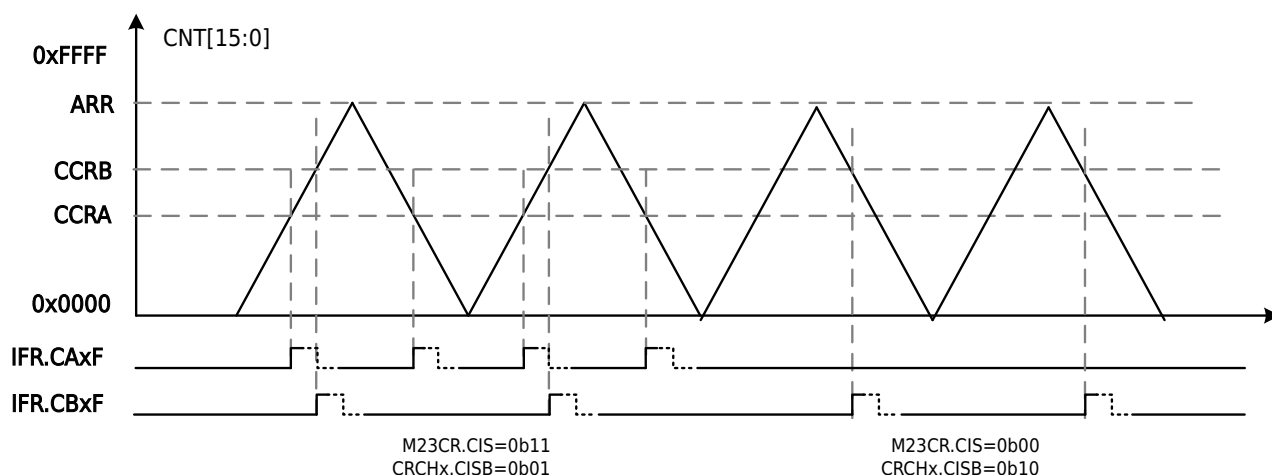


Figure 12-47 Compare Interrupt Diagram

12.3.6.11 Capture Input

The capture function (M23CR.MODE=2/3) can be configured in both triangular wave counting mode and sawtooth wave counting mode. The capture level edge can be set, and when a capture occurs, the captured value is stored in the compare capture register and generates a capture interrupt.

The compare capture function for each channel can be individually configured and selected through the CRCHx.CSA/CSB register. The capture edge for each channel can be independently set and via the CRCHx.CFy/CRY (x=0/1/2; y=A/B) register to select the triggering edge.

When the capture action occurs again before the capture flag is cleared after the capture occurs, the capture data overwrite flag will be generated. When the timer is not started, if there is a valid capture edge, the capture flag and capture action will also be generated.

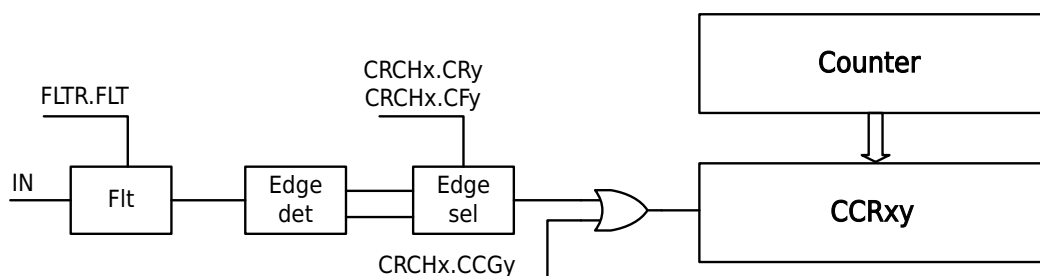


Figure 12-48 Capture Function Block Diagram

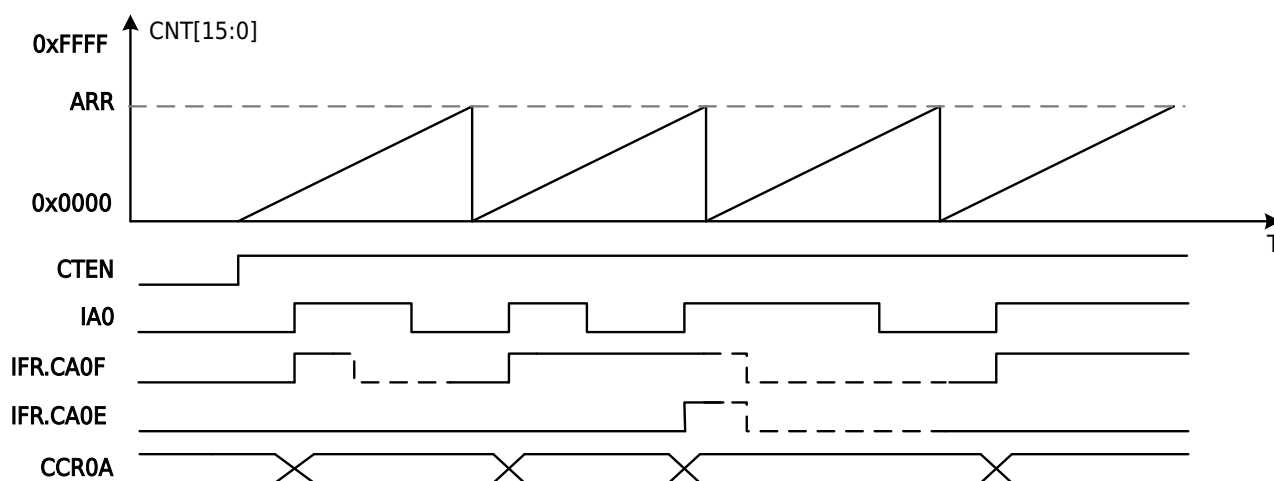


Figure 12-49 Capture Timing Diagram

CH0A Capture Input

The capture input signal for CH0A is as shown in the table below.

Table 12-7 CH0A Capture Input Signal

MSCR.IA0S	ATIM3
0b0	CH0A
0b1	CH0A CH1A CH2A XOR Input

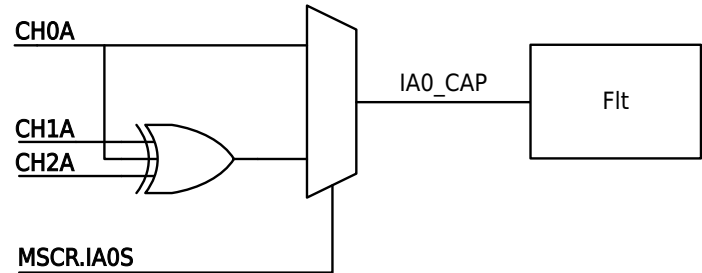


Figure 12-50 ATIM3 CH0A Capture Input Port Selection

CH0B Capture Input

The capture input signal for CH0B is as shown in the table below.

MSCR.IB0S	ATIM3																		
0b0	CH0B																		
0b1	Internal trigger MSCR.TS select signal <table><tr><th>MSCR.TS Value</th><th>Channel B Capture Input</th></tr><tr><td>0b000</td><td>ETR filtered phase selection output signal, filter phase selection configurable</td></tr><tr><td>0b001</td><td>Internal interconnection ITR0</td></tr><tr><td>0b010</td><td>Internal interconnection ITR1</td></tr><tr><td>0b011</td><td>Internal interconnection ITR2</td></tr><tr><td>0b100</td><td>Internal interconnection ITR3</td></tr><tr><td>0b101</td><td>CH0A edge</td></tr><tr><td>0b110</td><td>CH0A filtered output signal, filter function configurable</td></tr><tr><td>0b111</td><td>Reserved</td></tr></table>	MSCR.TS Value	Channel B Capture Input	0b000	ETR filtered phase selection output signal, filter phase selection configurable	0b001	Internal interconnection ITR0	0b010	Internal interconnection ITR1	0b011	Internal interconnection ITR2	0b100	Internal interconnection ITR3	0b101	CH0A edge	0b110	CH0A filtered output signal, filter function configurable	0b111	Reserved
MSCR.TS Value	Channel B Capture Input																		
0b000	ETR filtered phase selection output signal, filter phase selection configurable																		
0b001	Internal interconnection ITR0																		
0b010	Internal interconnection ITR1																		
0b011	Internal interconnection ITR2																		
0b100	Internal interconnection ITR3																		
0b101	CH0A edge																		
0b110	CH0A filtered output signal, filter function configurable																		
0b111	Reserved																		

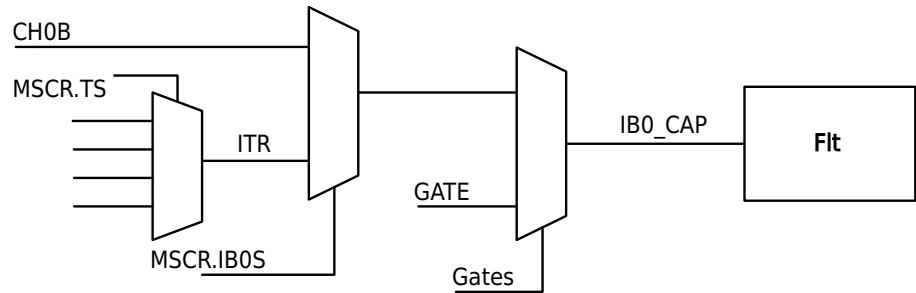


Figure 12-51 CH0B Capture Input Port Selection

The Gates signal is generated for selecting the PWM complementary output function. When PWM is the single point comparison complementary output, the compare capture register CCR0B is unused, and automatically switches to using GATE as the capture compare output.



Notice

When GATE is configured for compare output, its output mode only supports being set to force the output to 0 or 1 and being set to PWM mode valid (that is, OCMB0 is configured as 0/1/6/7).

12.3.7 Mode2/3 Slave Mode

The timer can be synchronized to an external trigger in several modes: reset mode, gated mode and triggered mode.

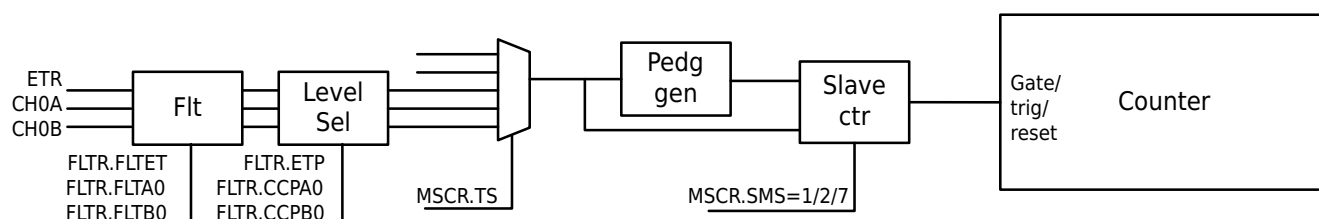


Figure 12-52 Mode2/3 Slave Mode Diagram

The slave mode function is selected via MSCR.SMS:

	Value	Description
MSCR.SMS	0b001	Slave mode function selection: reset mode
	0b010	Slave mode function selection: trigger mode
	0b111	Slave mode function selection: gating mode
MSCR.TS	0b000	The filtered phase selected signal ETP from port ETR features configurable filtering function
	0b001	Internal interconnect signal ITR0
	0b010	Internal interconnect signal ITR1
	0b011	Internal interconnect signal ITR2
	0b100	Internal interconnect signal ITR3
	0b101	Port CH0A edge signal
	0b110	The filtered phase selected signal IAFP from port CH0A features configurable filtering function
	0b111	The filtered phase selected signal IBFP from port CH0B features configurable filtering function

Gated Counting

Enable the counter according to the selected input level. In the following example, the counter only counts up when CH0A is low:

- Configure channel CH0A to detect a low level on CH0A. Configure the input filter bandwidth (filtering is not required in this case, therefore maintain FLTR.FLTA0=0b000). Set FLTR.CCPA0=1 to determine the polarity (only detect low level).

- Set MSCR.SMS=0b111 to configure the timer in gating mode; set MSCR.TS=0b110 to select CH0A as the input source.
- Set the M23CR.CTEN=1 to start the counter. In gating mode, if CTEN=0, the counter cannot be started regardless of the trigger input level. As long as CH0A is low, the counter starts counting according to the internal clock, and stops counting once CH0A becomes high.

**Note**

The gating input must not be a pulse signal. When gated counting, TS cannot be configured as 0b101; when TS is configured as 0b001/0b010/0b011/0b100, attention must be paid to whether ITR is a pulse signal.

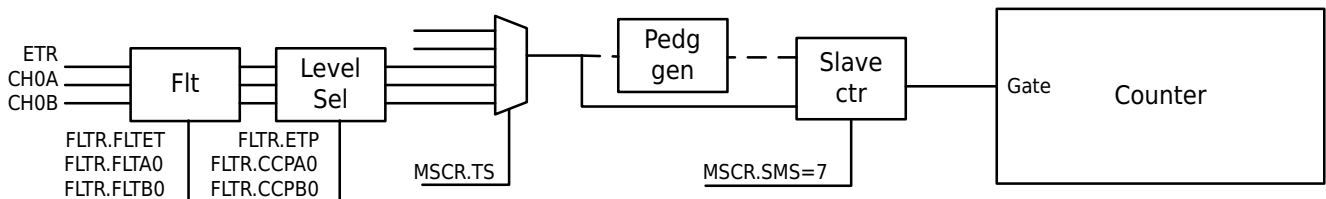


Figure 12-53 Mode2/3 Slave Mode Gating Function

Trigger Function

Using external triggers (CH0A, CH0B, ETR) enables synchronous timer startup. Alternatively, timer synchronization can be configured using the timer internal interconnect signals combined with MSCR.MSM. The trigger signal is the rising edge of the input signal. Alternatively, software trigger function can be initiated by writing to M23CR.TG.

A selected event on the input enables the counter. In the example below, the counter starts up counting on the rising edge of the CH0B input:

- Configure channel CH0B to detect the rising edge of CH0B. Configure the input filter bandwidth (in this example, no filter is required, keep FLTR.FLTB0=0b000). Set FLTR.CCPB0=0 to determine polarity (no inversion).
- Set MSCR.SMS=0b010 to configure the timer in trigger mode; set MSCR.TS=0b111 to select CH0B as the input source. When a rising edge appears on CH0B, the counter starts counting driven by the internal clock and sets the TIF flag at the same time. The delay between the rising edge of CH0B and the start of the counter depends on the synchronization circuit at the CH0B input.

**Note**

When using falling edge trigger, first select the trigger polarity and then choose the mode; otherwise, false triggering may occur.

Reset Counting

When a trigger input event occurs, both the counter and its prescaler can be reinitialized; Simultaneously, if the M23CR.URS is low, an update event UEV is generated; Subsequently, all preload registers (ARR, CCRxy) are updated.

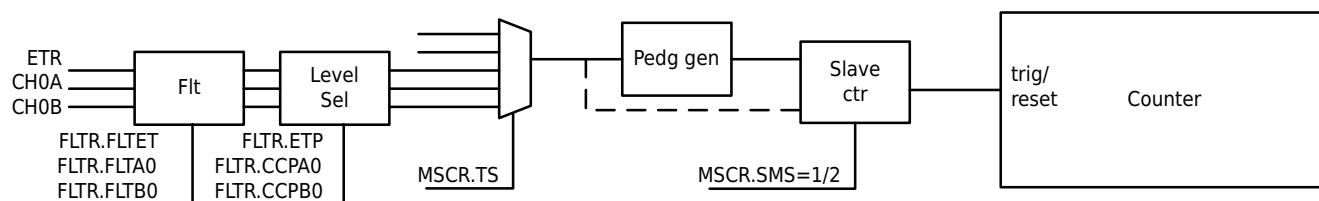


Figure 12-54 Mode2/3 Slave Mode Trigger and Reset Function

12.3.8 Quadrature Encoder Counting Function

MSCR.SMS=4/5/6, corresponding to the mode 1/2/3 of quadrature encoder mode. At this point, the counter performs encoder counting based on the phase relationship between IAFP and IBFP. IAFP and IBFP are the filtered phase selection signals for port inputs CH0A and CH0B.

Mode 1 uses the edge count of CH0A. Mode 2 uses the edge count of CH0B. Mode 3 uses both the edges count of CH0A and CH0B.

In order to ensure the correctness of the counting phase, it is necessary to ensure that the phase of the A/B input is greater than the phase difference of one pulse width, and the A/B input pulse width needs to be greater than two pulse widths.

As shown in the table below, the edge condition of IAFP corresponds to the level signal of IBFP, while the edge condition of IBFP corresponds to the level signal of IAFP. For example, in MOD1, when IBFP is at a high level and a rising edge is detected on IAFP, the counter counts down; when IBFP is at a high level and a falling edge is detected on IAFP, the counter counts up.

Mode	Quadrature Signal Level Condition		Quadrature Signal IAFP Edge Condition		Quadrature Signal IBFP Edge Condition	
	IBFP	IAFP	Rising Edge	Falling Edge	Rising Edge	Falling Edge
MOD1	High level	-	Down-counting	Up-counting	-	-
	Low level	-	Up-counting	Down-counting	-	-
MOD2	-	High level	-	-	Up-counting	Down-counting
	-	Low level	-	-	Down-counting	Up-counting
MOD3	High level	-	Down-counting	Up-counting	-	-
	Low level	-	Up-counting	Down-counting	-	-
	-	High level	-	-	Up-counting	Down-counting
	-	Low level	-	-	Down-counting	Up-counting

CHAF/CHBF is the signal filtered by port CH0A/CH0B, and IAFP/IBFP is the signal after port filter phase selection.

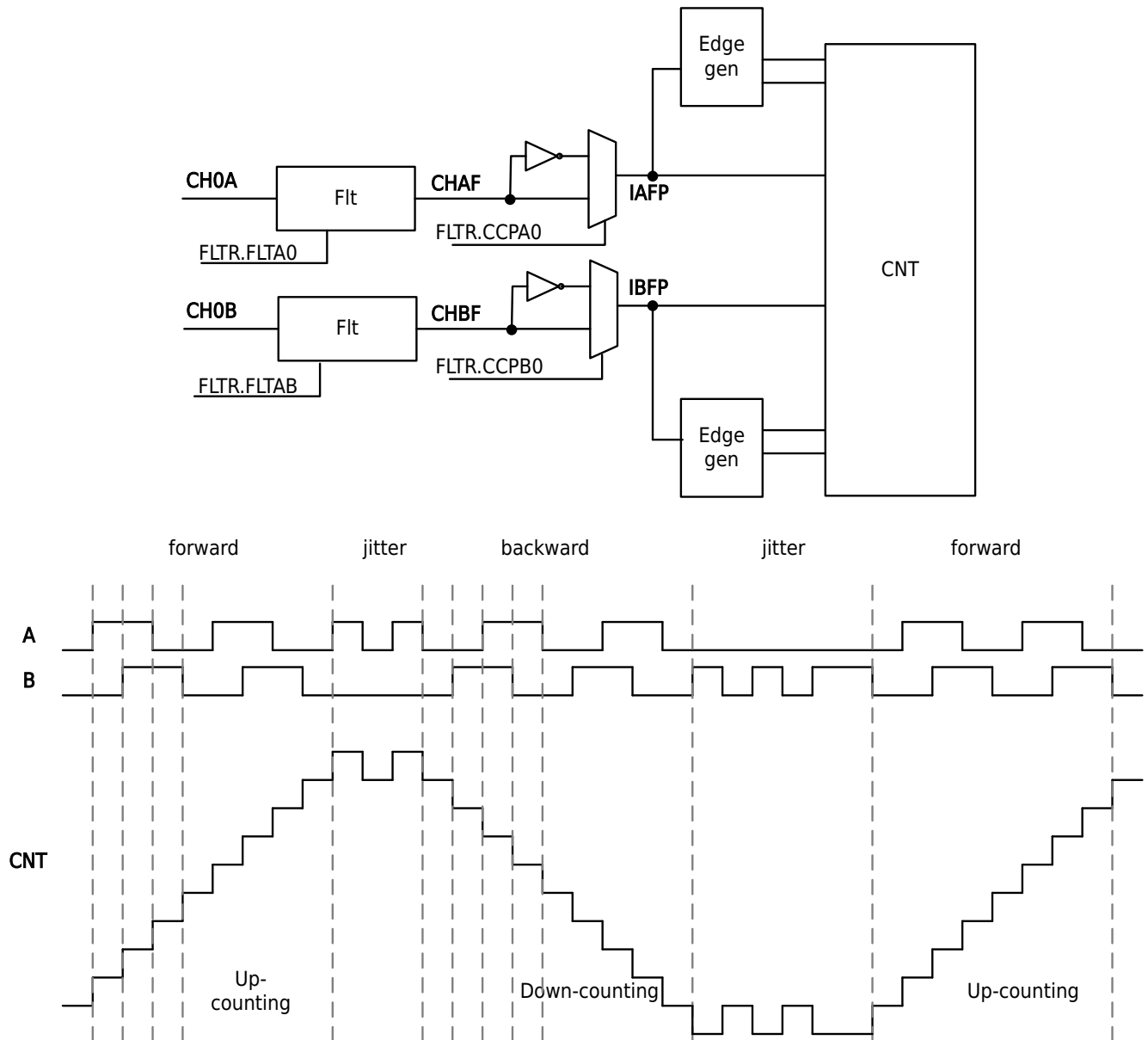


Figure 12-56 Quadrature Encoder Counting Mode3 Counting Diagram

12.3.9 Timer Triggers ADC

UEV Triggers ADC

When a update event (UEV) is generated and both ADTR.UEVE and ADTR.ADTE are enabled, a trigger request can be output to the ADC.

Compare Match Triggers ADC

CCR0A, CCR1A, and CCR2A compare matches can be configured to trigger the ADC.

- Mode3: The comparison match conditions for CHxA channels are uniformly controlled through the M23CR.CIS control register. You can select to trigger up-counting comparison match, down-counting comparison match, or both up-counting and down-counting comparison matches.
- Mode2: No need to configure M23CR.CIS.

CCR0B, CCR1B, and CCR2B compare matches can be configured to trigger the ADC.

- Mode 3: The comparison match conditions for CHxB channels are individually controlled via the CRCHx.CISB register. Options include triggering up-counting comparison match, down-counting comparison match, or both up-counting and down-counting comparison matches.

- Mode2: No configuration of CRCHx.CISB is required.

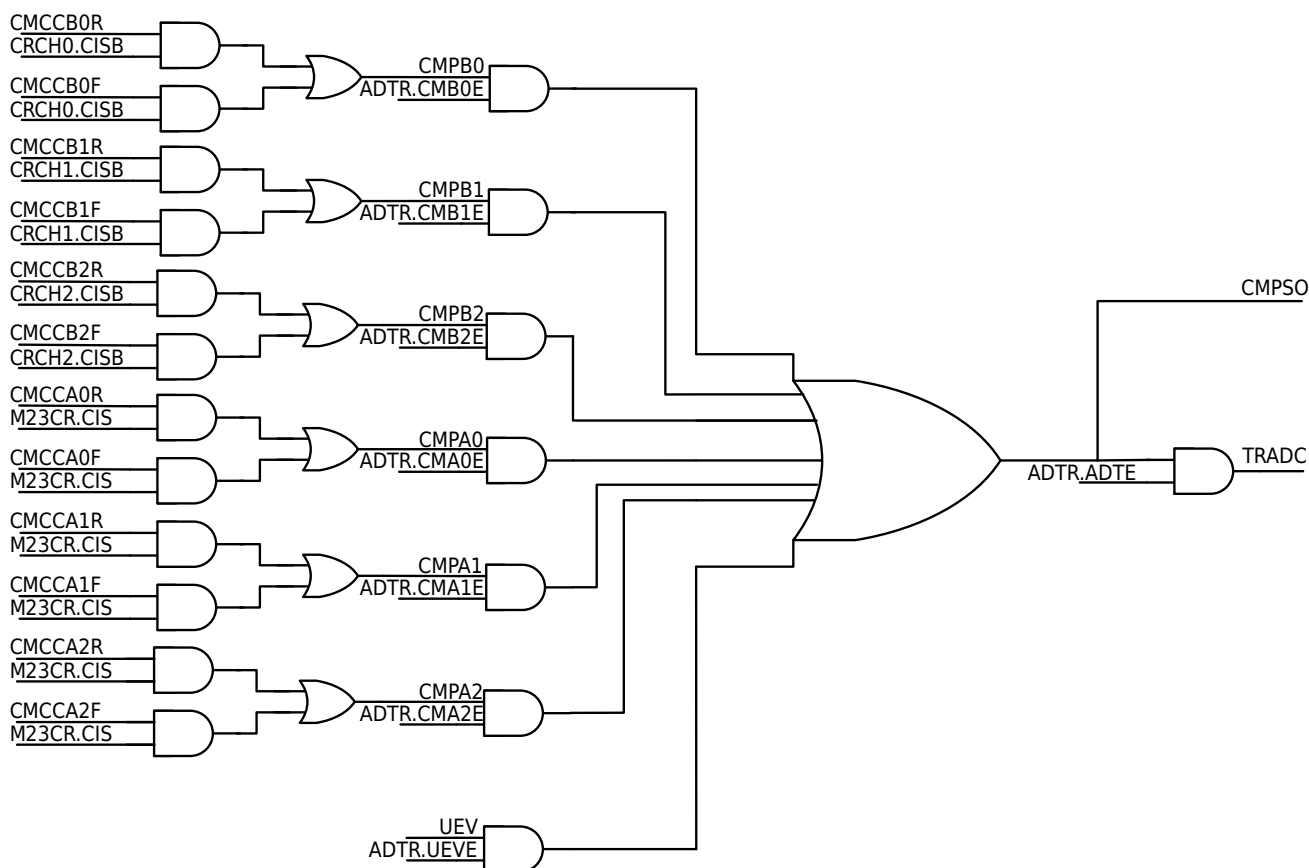


Figure 12-57 Timer Triggers ADC

12.3.10 Brake Control

The external BK port input can control the brake, and the system fail can control the brake function. The software brake function can be implemented through M23CR.BG to control the output port to the set state.

12.3.11 Interrupt

Table 12-12 ATIM3 Interrupt List

Interrupt Control Register	Interrupt Name
MxCR.UIE (x=0/1/23)	Overflow interrupt/event update interrupt
CRCHx.CIEA (x=0/1/2)	Channel (0/1/2) A compare capture interrupt
CRCHx.CIEB (x=0/1/2)	Channel (0/1/2) B compare capture interrupt
M23CR.BIE	Brake interrupt
M23CR.TIE	Trigger interrupt
M23CR.OVE	Overflow interrupt
M23CR.UNDE	Underflow interrupt

12.3.12 Timer Interconnection

The TRGO output signal can be connected to the ITR signal of other timers. The connection relationship is as follows:

Table 12-13 Timer Interconnection Signal List

	ITR0	ITR1	ITR2	ITR3
CTIM0	-	CTIM1_OV	-	ATIM3_TRGO
CTIM1	-	CTIM0_OV	-	ATIM3_TRGO
ATIM3	CTIM0_OV	CTIM1_OV	CTIM_TOG	-



Note

CTIM0_OV and CTIM1_OV are single cycle signals and cannot be used as inputs for gating mode.

12.3.13 Input Interconnection

The ETR input of ATIM3 can be input from a port or selected via the port function register to connect to other modules or ports. For specific configurations, refer to the register "Port Auxiliary Function Timer ETR Selection (GPIO_TIMES)" description in the "General-Purpose I/O (GPIO)" chapter.

The CH0A input of ATIM3 can be input from a port or selected via the port function register to connect to other modules or ports. For specific configurations, refer to the register "Port Auxiliary Function Timer ETR Selection (GPIO_TIMCPS)" description in the "General-Purpose I/O (GPIO)" chapter.

12.4 Operation Example

12.4.1 Mode0 Counting Timer Function Configuration Example

Procedure

- **Reload Timer Setting**
 - Step1.** Configure timer mode M0CR.MODE=0;
 - Step2.** Set load value ARR;
 - Step3.** Set counter initial value CNT;
 - Step4.** Clear interrupt flag;
 - Step5.** Enable interrupt M0CR.UIE;
 - Step6.** Enable reload mode M0CR.MD;
 - Step7.** Start timer M0CR.CTEN.
- **BUZZER Output Control**
 - Step1.** Set appropriate ARR value based on output frequency;
 - Step2.** Configure timer as reload mode, refer to [Reload Timer Settings](#);
 - Step3.** Enable output enable DTR.MOE;
 - Step4.** Start another timer to control M0CR.TOGEN for implementing interval frequency output.

12.4.2 Mode1 Pulse Width Measurement PWC Configuration Example

Procedure

- **Pulse Low Level Measurement Setting**
 - Step1.** Configure pulse measurement mode M1CR.MODE=1;

- Step2.** Set MSCR.TS to select measurement signal, then configure MSCR.IA0S/IB0S (IB0S cannot be set to 1) based on MSCR.TS setting to select measurement channel;
- Step3.** Set M1CR.EDG2ND=0 and M1CR.EDG1ST=1 to measure low level;
- Step4.** Clear interrupt flag;
- Step5.** Enable overflow interrupt M1CR.UIE;
- Step6.** Enable measurement completion interrupt CR0.CIEA;
- Step7.** Enable timer M1CR.CTEN;
- Step8.** In interrupt service routine, read CCR0A and overflow count then clear interrupt flag;
- Step9.** Wait for next measurement.

- **Pulse High Level Single Measurement Setting**

- Step1.** Configure pulse measurement mode M1CR.MODE=1;
- Step2.** Set MSCR.TS to select measurement signal, then configure MSCR.IA0S/IB0S (IB0S cannot be set to 1) based on MSCR.TS setting to select measurement channel;
- Step3.** Set pulse single-shot measurement mode M1CR.ONESHOT=1;
- Step4.** Configure M1CR.EDG2ND=1 and M1CR.EDG1ST=0 to measure high level;
- Step5.** Clear interrupt flag;
- Step6.** Enable overflow interrupt M1CR.UIE;
- Step7.** Enable measurement completion interrupt CR0.CIEA;
- Step8.** Enable timer M1CR.CTEN;
- Step9.** In interrupt service routine, read CCR0A and overflow count then clear interrupt flag;
- Step10.** Measurement is completed.

12.4.3 Mode2/3 Compare Capture Mode Configuration Example

Procedure

- **Edge-aligned Independent PWM Output Setting**

- Step1.** Set mode M23CR.MODE=2;
- Step2.** Configure sawtooth wave counting direction M23CR.DIR;
- Step3.** Set PWM period value ARR;
- Step4.** Set counter initial value (must be smaller than period value);
- Step5.** Configure PWM compare value CCRxA/CCRxB;
- Step6.** Set PWM output compare modes FLTR.OCMAx and FLTR.OCMBx to 6 or 7;
- Step7.** Clear relevant interrupt flags;
- Step8.** Enable corresponding interrupts;
- Step9.** Configure output polarities FLTR.CCPAx and FLTR.CCPBx;
- Step10.** Enable output DTR.MOE;
- Step11.** Enable timer M23CR.CTEN.

- **Center-Aligned Complementary PWM Output Setting**

- Step1.** Set mode M23CR.MODE=3;
- Step2.** Configure complementary output M23CR.COMP;
- Step3.** Set PWM period value ARR;
- Step4.** Set counter initial value (must be smaller than period value);
- Step5.** Set PWM compare value CCRxA (CCRxB does not need to be set as PWM output is independent of this register);
- Step6.** Set PWM output compare modes FLTR.OCMAx and FLTR.OCMBx to 6 or 7;

- Step7.** Clear relevant interrupt flags;
- Step8.** Enable corresponding interrupts;
- Step9.** Configure output polarities FLTR.CCPAx and FLTR.CCPBx;
- Step10.** Enable output DTR.MOE;
- Step11.** Enable timer M23CR.CTEN.

- **Asymmetric Center-Aligned Complementary PWM Output with Dead-Time Setting**

- Step1.** Set mode M23CR.MODE=3;
- Step2.** Configure complementary output M23CR.COMP;
- Step3.** Enable dual-point comparison enable M23CR.PWM2S;
- Step4.** Set PWM period value ARR;
- Step5.** Set counter initial value (must be smaller than period value);
- Step6.** Set PWM compare values CCRxA and CCRxB (up-counting comparison point is CCRxA, down-counting comparison point is CCRxB);
- Step7.** Set PWM output compare modes FLTR.OCMAx and FLTR.OCMBx to 6 or 7;
- Step8.** Clear relevant interrupt flags;
- Step9.** Enable corresponding interrupts;
- Step10.** Configure output polarities FLTR.CCPAx and FLTR.CCPBx;
- Step11.** Set dead time enable DTR.DTEN;
- Step12.** Set dead time duration DTR.DTR;
- Step13.** Enable output DTR.MOE;
- Step14.** Enable timer M23CR.CTEN.

- **Capture Function Configuration**, setting CH0B for rising edge capture function

- Step1.** Select counter counting mode, set M23CR.MODE=2;
- Step2.** Configure CH0B as capture mode CRCH0.CSB=1;
- Step3.** Select input filter FLTR.FLTB0 as needed;
- Step4.** Select capture edge CRCH0.CRB=1;
- Step5.** Set ARR to change the period value;
- Step6.** Clear relevant interrupt flags;
- Step7.** Clear the capture flag and enable the corresponding interrupt;
- Step8.** Start timer M23CR.CTEN;
- Step9.** After detecting the capture flag, read CCR0B to obtain the captured value.


Note

When the timer is not started, if there is a valid capture edge, the capture flag and capture action will also be generated.

12.4.4 Complementary PWM Output B Compare as ADC Trigger Function

Procedure

- Step1.** Configure as PWM complementary output mode, with setup steps referring to [Center-Aligned Complementary PWM Output Setting](#)
- Step2.** Set CCRxB to set ADC trigger comparison point
- Step3.** Set CRCHx.CISB to select up-counting and down-counting comparison matching (sawtooth wave does not need to be selected)
- Step4.** Select the source ADTR for ADC trigger comparison

Step5. Enable ADC triggers ADTR.ADTE

12.5 Register Description

12.5.1 Register List

ATIM3 base address: 0x40009800

Table 12-14 Mode0 Timer Register Offset

Offset	Register	Description	Width
0x000	ATIM3_ARR	16-Bit Mode Reload Register/Period	32
0x004	ATIM3_CNT	16-Bit Mode Count Register	32
0x008	ATIM3_CNT32	32-Bit Mode Count Register  Note ATIM3_CNT32 is the remapping register for ATIM3_ARR and ATIM3_CNT. In order to facilitate the 32-bit timer write operation.	32
0x00C	ATIM3_M0CR	Mode0 Control Register	32
0x010	ATIM3_IFR	Interrupt Flag Register	32
0x014	ATIM3_ICLR	Interrupt Flag Clear Register	32
0x030	ATIM3_DTR	Dead Time Register	32

Table 12-15 Mode1 Pulse Width Measurement (PWM) Register Offset

Offset	Register	Description	Width
0x004	ATIM3_CNT	16-Bit Mode Count Register	32
0x00C	ATIM3_M1CR	Mode1 Control Register	32
0x010	ATIM3_IFR	Interrupt Flag Register	32
0x014	ATIM3_ICLR	Interrupt Flag Clear Register	32
0x018	ATIM3_MSCR	Master-Slave Mode Control Register	32
0x01C	ATIM3_FLTR	Input Filter Register	32
0x024	ATIM3_CR0	Control Register	32
0x03C	ATIM3_CCR0A	Compare Capture Register	32

Table 12-16 Mode2/3 Compare-Capture Mode Register Offset

Offset	Register	Description	Width
0x000	ATIM3_ARR	16-Bit Mode Reload Register/Period	32
0x004	ATIM3_CNT	16-Bit Mode Count Register	32
0x00C	ATIM3_M23CR	Mode2/3 Control Register	32

Offset	Register	Description	Width
0x010	ATIM3_IFR	Interrupt Flag Register	32
0x014	ATIM3_ICLR	Interrupt Flag Clear Register	32
0x018	ATIM3_MSCR	Master-Slave Mode Control Register	32
0x01C	ATIM3_FLTR	Output Control/Input Filter Register	32
0x020	ATIM3_ADTR	ADC Trigger Control Register	32
0x024	ATIM3_CRCH0	Channel 0 Control Register	32
0x028	ATIM3_CRCH1	Channel 1 Control Register	32
0x02C	ATIM3_CRCH2	Channel 2 Control Register	32
0x030	ATIM3_DTR	Dead Time Register	32
0x034	ATIM3_RCR	Repeat Cycle Setting Value Register	32
0x03C	ATIM3_CCR0A	Channel 0A Compare/Capture Register	32
0x040	ATIM3_CCR0B	Channel 0B Compare/Capture Register	32
0x044	ATIM3_CCR1A	Channel 1A Compare/Capture Register	32
0x048	ATIM3_CCR1B	Channel 1B Compare/Capture Register	32
0x04C	ATIM3_CCR2A	Channel 2A Compare/Capture Register	32
0x050	ATIM3_CCR2B	Channel 2B Compare/Capture Register	32

12.5.2 Mode0 Count Timer Register

12.5.2.1 16-Bit Mode Reload Register (ATIM3_ARR)

Offset	Bit Position																															
0x000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																ARR															
Access																																

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	ARR	16-bit reload timer reload value register	RW

12.5.2.2 16-Bit Mode Count Register (ATIM3_CNT)

Offset	Bit Position																															
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CNT															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	CNT	16-bit reload timer count value register	RW

12.5.2.3 32-Bit Mode Count Register (ATIM3_CNT32)

Offset	Bit Position																															
0x008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	CNT32																															
Access	RW																															

Bit	Name	Functional Description	Access
31:0	CNT32	32-bit timer count value register Valid only in Mode 0 32-bit timer free-counting mode, and writing to this register disables in other modes.	RW

12.5.2.4 Mode0 Control Register (ATIM3_M0CR)

Offset	Bit Position																															
0x00C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00600008																															
Name	Reserved																MODE	Reserved	UIE	GATEP	GATE	Reserved	PRS				TOGEN	CT	MD	CTEN		
Access																	RW		RW	RW	RW			RW					RW	RW	RW	RW

Bit	Name	Functional Description	Access
31:14	Reserved	-	-
13:12	MODE	Working mode 0b00: Timer mode 0 0b01: PWC mode 0b10: Sawtooth wave mode 0b11: Triangle wave mode	RW
11	Reserved	-	-
10	UIE	Interrupt enable control 0b0: Disable 0b1: Enable	RW
9	GATEP	Port GATE polarity control 0b0: Port GATE high level is valid 0b1: Port GATE low level is valid	RW
8	GATE	Timer gating enable 0b0: No gating, timer works when CTEN=1 0b1: Gating enabled, timer only operates when GATE port is valid and CTEN=1	RW
7	Reserved	-	-
6:4	PRS	Internal clock frequency division selection 0b000: 1 0b001: 2 0b010: 4 0b011: 8 0b100: 16 0b101: 32 0b110: 64 0b111: 256	RW
3	TOGEN	Toggle output enable in mode 0 0b0: Toggle output disabled, CHA and CHB outputs remain low level 0b1: Toggle output enabled	RW

Bit	Name	Functional Description	Access
2	CT	Count clock selection ● 0b0: Internal count clock TCLK ● 0b1: External count clock ETR  Note When configured in external clock mode, this clock selection is invalid.	RW
1	MD	Mode selection 32 timing/16 timing mode selection ● 0b0: 32-bit free counting ● 0b1: 16-bit reload counting	RW
0	CTEN	Timer enable ● 0b0: Disable ● 0b1: Enable	RW

12.5.2.5 Interrupt Flag Register (ATIM3_IFR)

Offset	Bit Position																															
0x010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															UIF
Access																																RO

Bit	Name	Functional Description	Access
31:1	Reserved	-	-
0	UIF	Overflow interrupt 0b0: Counter is not overflowed 0b1: Counter overflowed	RO

12.5.2.6 Interrupt Flag Clear Register (ATIM3_ICLR)

Offset	Bit Position																															
0x014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000FFFFFFF																															
Name	Reserved																															UIF
Access																																WO

Bit	Name	Functional Description	Access
31:1	Reserved	-	-
0	UIF	Overflow interrupt clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

12.5.2.7 Dead Time Register (ATIM3_DTR)

Offset	Bit Position																															
0x030	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				MOE	Reserved										
Access																					RW											

Bit	Name	Functional Description	Access
31:13	Reserved	-	-
12	MOE	Toggle output enable 0b0: Toggle output in input status 0b1: Toggle port in output status	RW
11:0	Reserved	-	-

12.5.3 Mode1 Pulse Width Measurement PWC Register

12.5.3.1 16-Bit Mode Count Register (ATIM3_CNT)

Offset	Bit Position																															
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CNT															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	CNT	Count value register	RW

12.5.3.2 Mode1 Control Register (ATIM3_M1CR)

Offset	Bit Position																																		
0x00C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00600008																																		
Name	Reserved																	ONESHOT	MODE		Reserved	UIE		EDG2ND	EDG1ST	Reserved	PRS		Reserved	CT	Reserved	CTEN			
Access																		RW	RW			RW		RW	RW		RW			RW			RW		

Bit	Name	Functional Description	Access															
31:15	Reserved	-	-															
14	ONESHOT	Single trigger mode selection 0b0: Cycle measurement 0b1: Automatically terminates after completing one pulse measurement, re-enable CTEN is required for subsequent measurements	RW															
13:12	MODE	Working mode 0b00: Timer mode 0 0b01: PWC mode 0b10: Sawtooth wave mode 0b11: Triangle wave mode	RW															
11	Reserved	-	-															
10	UIE	Interrupt enable control Counting to 0xFFFF will overflow and generate an overflow flag 0b0: Disable 0b1: Enable	RW															
9	EDG2ND	Pulse width measurement end edge selection Refer to the EDG1ST function description for details	RW															
8	EDG1ST	Pulse width measurement start edge selection	RW															
		<table><tr><th>EDG2ND</th><th>EDG1ST</th><th>Measurement</th></tr><tr><td>0b0</td><td>0b0</td><td>Up edge-up edge period</td></tr><tr><td>0b0</td><td>0b1</td><td>Low level width</td></tr><tr><td>0b1</td><td>0b0</td><td>High level width</td></tr><tr><td>0b1</td><td>0b1</td><td>Falling edge-falling edge period</td></tr></table>		EDG2ND	EDG1ST	Measurement	0b0	0b0	Up edge-up edge period	0b0	0b1	Low level width	0b1	0b0	High level width	0b1	0b1	Falling edge-falling edge period
		EDG2ND		EDG1ST	Measurement													
		0b0		0b0	Up edge-up edge period													
		0b0		0b1	Low level width													
		0b1		0b0	High level width													
0b1	0b1	Falling edge-falling edge period																
7	Reserved	-	-															

Bit	Name	Functional Description	Access
6:4	PRS	Internal clock frequency division selection ● 0b000: 1 ● 0b001: 2 ● 0b010: 4 ● 0b011: 8 ● 0b100: 16 ● 0b101: 32 ● 0b110: 64 ● 0b111: 256	RW
3	Reserved	-	-
2	CT	Count clock selection ● 0b0: Internal count clock TCLK ● 0b1: External count clock ETR	RW
1	Reserved	-	-
0	CTEN	Pulse width measurement enable ● 0b0: Disable ● 0b1: Enable	RW

12.5.3.3 Interrupt Flag Register (ATIM3_IFR)

Offset	Bit Position																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
0x010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
Reset	0x00000000																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
Name	Reserved																										CA0F		Reserved		UIF																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
Access																											RO				RO																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					</

Bit	Name	Functional Description	Access
31:3	Reserved	-	-
2	CA0F	Pulse width measurement interrupt flag ● 0b0: Pulse width measurement is not completed ● 0b1: Pulse width measurement completed	RO
1	Reserved	-	-
0	UIF	Overflow interrupt flag ● 0b0: Counter is not overflowed ● 0b1: Counter overflowed	RO

12.5.3.4 Interrupt Flag Clear Register (ATIM3_ICLR)

Offset	Bit Position																																	
0x014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x000FFFFF																																	
Name	Reserved																										CA0F		Reserved		UIF			
Access																											WO				WO			

Bit	Name	Functional Description	Access
31:3	Reserved	-	-
2	CA0F	Pulse width measurement interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
1	Reserved	-	-
0	UIF	Overflow interrupt clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

12.5.3.5 Master-Slave Mode Control Register (ATIM3_MSCR)

Offset	Bit Position																																	
0x018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																				IB0S	IA0S	Reserved			TS		Reserved						
Access																					RW	RW				RW								

Bit	Name	Functional Description	Access
31:13	Reserved	-	-
12	IB0S	CH0B input selection 0b0: CH0B 0b1: Internal trigger TS selection signal	RW
11	IA0S	IA0 input selection 0b0: CH0A 0b1: CH0A CH1A CH2A XOR  Note Setting to 1 will cause input changes upon any port transition.	RW
10:8	Reserved	-	-
7:5	TS	Trigger selection 0b000: Signal ETRFP after port ETR filtered phase selection 0b001: Internal interconnect signal ITR0 0b010: Internal interconnect signal ITR1 0b011: Internal interconnect signal ITR2 0b100: Internal interconnect signal ITR3 0b101: Setting prohibited 0b110: Signal IAFP after port CH0A filtered phase selection 0b111: Signal IBFP after port CH0B filtered phase selection	RW
4:0	Reserved	-	-

12.5.3.6 Input Filter Register (ATIM3_FLTR)

Offset	Bit Position																															
0x01C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	ETP	FLTET	Reserved																				FLTBO	Reserved	FLTA0							
Access			RW	RW																						RW		RW				

Bit	Name	Functional Description	Access
31	ETP	ETR input phase selection 0b0: Same phase 0b1: Reverse input	RW
30:28	FLTET	ETR filter control The sampling clock is PCLK or PCLK frequency division, and the number of sampling points is N 0b000: No filter 0b001: $F_{\text{sample}} = \text{PCLK}$, $N=2$ 0b010: $F_{\text{sample}} = \text{PCLK}$, $N=4$ 0b011: $F_{\text{sample}} = \text{PCLK}$, $N=6$ 0b100: $F_{\text{sample}} = \text{PCLK}/4$, $N=4$ 0b101: $F_{\text{sample}} = \text{PCLK}/4$, $N=6$ 0b110: $F_{\text{sample}} = \text{PCLK}/8$, $N=4$ 0b111: $F_{\text{sample}} = \text{PCLK}/8$, $N=6$	RW
27:7	Reserved	-	-
6:4	FLTBO	CHB input filter control The sampling clock is PCLK or PCLK frequency division, and the number of sampling points is N 0b000: No filter 0b001: $F_{\text{sample}} = \text{PCLK}$, $N=2$ 0b010: $F_{\text{sample}} = \text{PCLK}$, $N=4$ 0b011: $F_{\text{sample}} = \text{PCLK}$, $N=6$ 0b100: $F_{\text{sample}} = \text{PCLK}/4$, $N=4$ 0b101: $F_{\text{sample}} = \text{PCLK}/4$, $N=6$ 0b110: $F_{\text{sample}} = \text{PCLK}/8$, $N=4$ 0b111: $F_{\text{sample}} = \text{PCLK}/8$, $N=6$  Note CR0.CSB needs to be configured as 1.	RW
3	Reserved	-	-
2:0	FLTA0	CHA input filter control	RW

Bit	Name	Functional Description	Access
2:0	FLTA0	The sampling clock is PCLK or PCLK frequency division, and the number of sampling points is N ● 0b000: No filter ● 0b001: $F_{\text{sample}} = \text{PCLK}$, N=2 ● 0b010: $F_{\text{sample}} = \text{PCLK}$, N=4 ● 0b011: $F_{\text{sample}} = \text{PCLK}$, N=6 ● 0b100: $F_{\text{sample}} = \text{PCLK}/4$, N=4 ● 0b101: $F_{\text{sample}} = \text{PCLK}/4$, N=6 ● 0b110: $F_{\text{sample}} = \text{PCLK}/8$, N=4 ● 0b111: $F_{\text{sample}} = \text{PCLK}/8$, N=6  Note CR0.CSA needs to be configured as 1.	RW

12.5.3.7 Control Register (ATIM3_CR0)

Offset	Bit Position																																				
0x024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00003000																																				
Name	Reserved																								CIEA		Reserved		CSB		CSA		Reserved				
Access																									RW				RW		RW						

Bit	Name	Functional Description	Access
31:9	Reserved	-	-
8	CIEA	Pulse width measurement complete interrupt enable 0b0: Disable 0b1: Enable	RW
7:6	Reserved	-	-
5	CSB	B channel capture/compare function selection When pulse width measurement uses channel B filtering, CSB must be set to 1 0b0: Compare mode 0b1: Capture mode	RW
4	CSA	A channel capture/compare function selection When pulse width measurement uses channel A filtering, CSA must be set to 1 0b0: Compare mode 0b1: Capture mode	RW
3:0	Reserved	-	-

12.5.3.8 Compare Capture Register (ATIM3_CCR0A)

Offset	Bit Position																															
0x03C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CCR0A															
Access																	RO															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	CCR0A	Pulse width measurement result	RO

12.5.4 Mode2/3 Compare Capture Mode Register

12.5.4.1 16-Bit Mode Reload Register (ATIM3_ARR)

Offset	Bit Position																															
0x000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																ARR															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	ARR	Reload register/period register With cache function, when the counter is not enabled or the cache is not enabled, the cache register can be updated immediately	RW

12.5.4.2 16-Bit Mode Count Register (ATIM3_CNT)

Offset	Bit Position																															
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CNT															
Access																																
																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	CNT	Reload timer count value register When using PWM compare output, triangular wave mode or saw-tooth wave mode up-counting, the initialized CNT value must be less than the ARR value.	RW

12.5.4.3 Mode2/3 Control Register (ATIM3_M23CR)

Offset	Bit Position																																		
0x00C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00600008																																		
Name	Reserved	UNDE	OVE	DIR	BG	UG	TG	OCCE	CIS	BIE	TIE	TDE	URS	OCCS	CSG	ONESHOT	MODE	UDE	UIE	CFG	CRG	BUFPEN	PRS	PWM2S	CT	COMP	CTEN								
Access		RW	RW	RW/RO	WO	WO	WO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		

Bit	Name	Functional Description	Access
31:30	Reserved	-	-
29	UNDE	Underflow interrupt enable 0b0: Disable 0b1: Enable	RW
28	OVE	Overflow interrupt enable 0b0: Disable 0b1: Enable	RW
27	DIR	Count direction Write only in sawtooth wave mode. Read only in other modes, write invalid 0b0: Up-counting 0b1: Down-counting  Note DIR is automatically cleared when switching from other modes to center-aligned mode. In the center-aligned mode, DIR is automatically cleared in reset mode by software event update and external trigger in slave mode.	RW/RO
26	BG	Software brake Auto-clear. Write 1 to generate a software brake, write 0 has no effect.	WO
25	UG	Software update Auto-clear. Write 1 to generate a software update, write 0 has no effect. Initializes the counter and updates the buffer register to the corresponding registers (buffer enable), the PRS bit will also be cleared.	WO
24	TG	Software trigger Auto-clear. Write 1 to generate a software trigger, write 0 has no effect. Triggering is available in trigger mode SMS=2 and mode=2/3.	WO

Bit	Name	Functional Description	Access
23	OCCE	OCREF clear enable ● 0b0: OCREF output unaffected by OCREF_CLR ● 0b1: OCREF_CLR signal can clear OCREF output	RW
22:21	CIS	Channel A compare match mode ● 0b00: No match ● 0b01: Compare match during up-counting ● 0b10: Compare match during down-counting ● 0b11: Compare match during both up/down counting  Note Channel B compare match mode is controlled by the CISB bit in the CRCHx register.	RW
20	BIE	Brake interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
19	TIE	Trigger interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
18	TDE	<i>Reserved</i>	RW
17	URS	Update source ● 0b0: Overflow/underflow/software update UG/slave mode reset ● 0b1: Overflow/underflow	RW
16	OCES	OCREF_CLR signal source selection ● 0b0: Voltage comparator VC output, VC selection is configured in the VCx_CR2 register. ● 0b1: Signal after ETR port filtered phase selection When OCCE is enabled, OCREF_CLR can clear the OCREF compare output signal (valid when OCM _{yx} >1), and the compare output resumes after the next UEV event.	RW
15	CSG	GATE capture/compare selection in PWM complementary mode Valid only during PWM complementary output, using CCR0B as the GATE compare or capture channel ● 0b0: Compare ● 0b1: Capture	RW
14	ONESHOT	Single trigger mode selection ● 0b0: Cycle count ● 0b1: Timer stops after an event update occurs	RW
13:12	MODE	Working mode ● 0b00: Timer mode 0 ● 0b01: PWC mode ● 0b10: Sawtooth wave mode ● 0b11: Triangle wave mode	RW
11	UDE	<i>Reserved</i>	RW

Bit	Name	Functional Description	Access
10	UIE	Update interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
9	CFG	Falling edge capture valid control when GATE is used as capture input Only valid when PWM complementary output ● 0b0: Lower edge capture is invalid ● 0b1: Lower edge capture is valid	RW
8	CRG	Rising edge capture valid control when GATE is used as capture input Only valid when PWM complementary output ● 0b0: Upper edge capture is invalid ● 0b1: Upper edge capture is valid	RW
7	BUFPEN	Auto-reload ARR cache enable ● 0b0: Period cache disabled, written value immediately affects the period value ● 0b1: Period cache enabled, written value affects the period value after the next event update	RW
6:4	PRS	Internal clock frequency division selection ● 0b000: 1 ● 0b001: 2 ● 0b010: 4 ● 0b011: 8 ● 0b100: 16 ● 0b101: 32 ● 0b110: 64 ● 0b111: 256	RW
3	PWM2S	OCREFA dual-point compare selection ● 0b0: Dual-point compare enabled, uses CCRA and CCRB compares to control OCREFA output ● 0b1: Single-point compare enabled, only uses CCRA compare to control OCREFA output (default value)  Note OCREFB remains unaffected and still uses CCRB to control OCREFB output.	RW
2	CT	Count clock selection ● 0b0: Internal count clock TCLK ● 0b1: External count clock ETR	RW
1	COMP	PWM complementary output mode selection ● 0b0: Independent PWM output ● 0b1: Complementary PWM output	RW

Bit	Name	Functional Description	Access
0	CTEN	Timer enable ● 0b0: Disable ● 0b1: Enable It can be enabled by external trigger, and this bit is automatically cleared at the end of ONESHOT mode	RW

12.5.4.4 Interrupt Flag Register (ATIM3_IFR)

Offset	Bit Position																																													
0x010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0														
Reset	0x00000000																																													
Name	Reserved														UNDF	OVF	TIF	BIF	CB2E	CB1E	CB0E	CA2E	CA1E	CA0E	CB2F	CB1F	CB0F	CA2F	CA1F	CA0F	Reserved	UIF														
Access															RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO		

Bit	Name	Functional Description	Access
31:18	Reserved	-	-
17	UNDF	Underflow interrupt flag ● 0b0: Not occurred ● 0b1: Occurred	RO
16	OVF	Overflow interrupt flag ● 0b0: Not occurred ● 0b1: Occurred	RO
15	TIF	Trigger interrupt flag ● 0b0: Not occurred ● 0b1: Occurred	RO
14	BIF	Brake interrupt flag ● 0b0: Not occurred ● 0b1: Occurred	RO
13	CB2E	Channel CH2B capture data loss flag ● 0b0: No data loss ● 0b1: Data loss	RO
12	CB1E	Channel CH1B capture data loss flag ● 0b0: No data loss ● 0b1: Data loss	RO
11	CB0E	Channel CH0B capture data loss flag ● 0b0: No data loss ● 0b1: Data loss	RO
10	CA2E	Channel CH2A capture data loss flag ● 0b0: No data loss ● 0b1: Data loss	RO
9	CA1E	Channel CH1A capture data loss flag ● 0b0: No data loss ● 0b1: Data loss	RO
8	CA0E	Channel CH0A capture data loss flag ● 0b0: No data loss ● 0b1: Data loss	RO

Bit	Name	Functional Description	Access
7	CB2F	Channel CH2B capture/compare match flag ● 0b0: Not occurred ● 0b1: Occurred	RO
6	CB1F	Channel CH1B capture/compare match flag ● 0b0: Not occurred ● 0b1: Occurred	RO
5	CB0F	Channel CH0B capture/compare match flag ● 0b0: Not occurred ● 0b1: Occurred	RO
4	CA2F	Channel CH2A capture/compare match flag ● 0b0: Not occurred ● 0b1: Occurred	RO
3	CA1F	Channel CH1A capture/compare match flag ● 0b0: Not occurred ● 0b1: Occurred	RO
2	CA0F	Channel CH0A capture/compare match flag ● 0b0: Not occurred ● 0b1: Occurred	RO
1	<i>Reserved</i>	-	-
0	UIF	Event update interrupt flag ● 0b0: Not occurred ● 0b1: Occurred	RO

12.5.4.5 Interrupt Flag Clear Register (ATIM3_ICLR)

Offset	Bit Position																															
0x014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x000FFFFFFF																															
Name	Reserved														UNDF	OVF	TIF	BIF	CB2E	CB1E	CB0E	CA2E	CA1E	CA0E	CB2F	CB1F	CB0F	CA2F	CA1F	CA0F	Reserved	UIF
Access															WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

Bit	Name	Functional Description	Access
31:18	Reserved	-	-
17	UNDF	Underflow interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
16	OVF	Overflow interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
15	TIF	Trigger interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
14	BIF	Brake interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
13	CB2E	Channel CH2B capture data loss flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
12	CB1E	Channel CH1B capture data loss flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
11	CB0E	Channel CH0B capture data loss flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
10	CA2E	Channel CH2A capture data loss flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
9	CA1E	Channel CH1A capture data loss flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
8	CA0E	Channel CH0A capture data loss flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
7	CB2F	Channel CH2B capture/compare match flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
6	CB1F	Channel CH1B capture/compare match flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
5	CB0F	Channel CH0B capture/compare match flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
4	CA2F	Channel CH2A capture/compare match flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

Bit	Name	Functional Description	Access
3	CA1F	Channel CH1A capture/compare match flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
2	CA0F	Channel CH0A capture/compare match flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
1	<i>Reserved</i>	-	-
0	UIF	Event update interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

12.5.4.6 Master-Slave Mode Control Register (ATIM3_MSCR)

Offset	Bit Position																																	
0x018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																				IB0S		IA0S		SMS		TS		MSM		CCDS		MMS	
Access																					RW		RW		RW		RW		RW		RW		RW	

Bit	Name	Functional Description	Access
31:13	Reserved	-	-
12	IB0S	CH0B input selection 0b0: CH0B 0b1: Internal trigger TS selection signal	RW
11	IA0S	IA0 input selection 0b0: CH0A 0b1: CH0A CH1A CH2A XOR  Note Setting to 1 will cause input changes upon any port transition.	RW
10:8	SMS	Slave mode function selection 0b000: Use internal clock 0b001: Reset function 0b010: Trigger mode 0b011: External clock mode 0b100: Quadrature encoder counting mode1 0b101: Quadrature encoder counting mode2 0b110: Quadrature encoder counting mode3 0b111: Gating function	RW
7:5	TS	Trigger selection 0b000: Signal ETRFP after port ETR filtered phase selection 0b001: Internal interconnect signal ITR0 0b010: Internal interconnect signal ITR1 0b011: Internal interconnect signal ITR2 0b100: Internal interconnect signal ITR3 0b101: Edge signal of port CH0A 0b110: Signal IAFP after port CH0A filtered phase selection 0b111: Signal IBFP after port CH0B filtered phase selection	RW

Bit	Name	Functional Description	Access
4	MSM	Master-slave selection ● 0b0: No delay ● 0b1: Delay enabled, synchronously starts both master and slave counters  Note When using the trigger mode, the slave mode is set to 0, and the master mode is set to 1, so that the master and slave counts can be started at the same time.	RW
3	CCDS	<i>Reserved</i>	RW
2:0	MMS	Master mode input selection For internal interconnection, connected to ITRx of other timers ● 0b000: Software update UG, write M23CR.UG; software update TG, write M23CR.TG ● 0b001: Timer enable CTEN ● 0b010: Timer event update UEV ● 0b011: Compare match select output CMPSO ● 0b100: Timer compare parameter output OCREFOA ● 0b101: Timer compare parameter output OCREFOA ● 0b110: Timer compare parameter output OCREFOA ● 0b111: Timer compare parameter output OCREFOB  Note When MMS is configured as 0b000/0b010/0b011, the output is a pulse signal.	RW

12.5.4.7 Output Control/Input Filter Register (ATIM3_FLTR)

Offset	Bit Position																															
0x01C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	ETP	FLTET		BKP		FLTBK		CCPB2		OCMB2/FLTB2		CCPA2		OCMA2/FLTA2		CCPB1		OCMB1/FLTB1		CCPA1		OCMA1/FLTA1		CCPB0/CCPB0		OCMB0/FLTB0		CCPA0/CCPA0		OCMA0/FLTA0		
Access	RW	RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		RW		

Bit	Name	Functional Description	Access
31	ETP	ETR input phase selection 0b0: Same phase 0b1: Inverted input	RW
30:28	FLTET	ETR filter control The sampling clock is PCLK or PCLK frequency division, and the number of sampling points is N 0b000: No filter 0b001: $F_{\text{sample}} = \text{PCLK}$, $N=2$ 0b010: $F_{\text{sample}} = \text{PCLK}$, $N=4$ 0b011: $F_{\text{sample}} = \text{PCLK}$, $N=6$ 0b100: $F_{\text{sample}} = \text{PCLK}/4$, $N=4$ 0b101: $F_{\text{sample}} = \text{PCLK}/4$, $N=6$ 0b110: $F_{\text{sample}} = \text{PCLK}/8$, $N=4$ 0b111: $F_{\text{sample}} = \text{PCLK}/8$, $N=6$	RW
27	BKP	Brake BK input phase selection 0b0: Same phase 0b1: Inverted input	RW
26:24	FLTBK	Brake input filter control The sampling clock is PCLK or PCLK frequency division, and the number of sampling points is N 0b000: No filter 0b001: $F_{\text{sample}} = \text{PCLK}$, $N=2$ 0b010: $F_{\text{sample}} = \text{PCLK}$, $N=4$ 0b011: $F_{\text{sample}} = \text{PCLK}$, $N=6$ 0b100: $F_{\text{sample}} = \text{PCLK}/4$, $N=4$ 0b101: $F_{\text{sample}} = \text{PCLK}/4$, $N=6$ 0b110: $F_{\text{sample}} = \text{PCLK}/8$, $N=4$ 0b111: $F_{\text{sample}} = \text{PCLK}/8$, $N=6$	RW

Bit	Name	Functional Description	Access
23	CCPB2	Compare function: CH2B channel compare output phase control ● 0b0: Normal output ● 0b1: Inverted output	RW
22:20	OCMB2	Compare function: CH2B channel compare control Refer to OCMB0 setting	RW
	FLTB2	Capture function: CH2B input channel filter setup Refer to FLTBK setting	RW
19	CCPA2	Compare function: CH2A channel compare output phase control ● 0b0: Normal output ● 0b1: Inverted output	RW
18:16	OCMA2	Compare function: CH2A channel compare control Refer to OCMB0 setting	RW
	FLTA2	Capture function: CH2A input channel filter setup Refer to FLTBK setting	RW
15	CCPB1	Compare function: CH1B channel compare output phase control ● 0b0: Normal output ● 0b1: Inverted output	RW
14:12	OCMB1	Compare function: CH1B channel compare control Refer to OCMB0 setting	RW
	FLTB1	Capture function: CH1B input channel filter setup Refer to FLTBK setting	RW
11	CCPA1	Compare function: CH1A channel compare output phase control ● 0b0: Normal output ● 0b1: Inverted output	RW
10:8	OCMA1	Compare function: CH1A channel compare control Refer to OCMB0 setting	RW
	FLTA1	Capture function: CH1A input channel filter setup Refer to FLTBK setting	RW
7	CCPB0	Compare function: output compare mode CCPBx compare output CHBx port polarity control ● 0b0: Normal output ● 0b1: Inverted output	RW
	CCPB0	Encode count and slave mode gating function: input phase control CCPB0 slave mode gating, reset, external trigger, external clock using CH0B port input polarity control ● 0b0: Normal input ● 0b1: Inverted input	RW
6:4	OCMB0	Compare function: CH0B channel compare control ● 0b000: Force to 0 ● 0b001: Force to 1 ● 0b010: Force to 0 on compare match ● 0b011: Force to 1 on compare match	RW

Bit	Name	Functional Description	Access
6:4	OCMB0	● 0b100: Toggle on compare match ● 0b101: Output high level for one clock cycle on compare match ● 0b110: PWM mode 1 ▶ Single point comparison During up-counting, the output is high level when $CNT < CCRxy$; during down-counting, the output is low level when $CNT > CCRxy$. ▶ Dual point comparison ◆ For sawtooth wave up-counting, the output is low level when $CCRxA < CNT \leq CCRxB$ ◆ For sawtooth wave down-counting, the output is high level when $CCRxA < CNT \leq CCRxB$ ◆ For triangular wave up-counting, the output is high level when $CNT < CCRxA$; for down-counting, the output is low level when $CNT > CCRxB$ ● 0b111: PWM mode 2 ▶ Single point comparison During up-counting, the output is low level when $CNT < CCRxy$; during down-counting, the output is high level when $CNT > CCRxy$ ▶ Dual point comparison ◆ For sawtooth wave up-counting, the output is high level when $CCRxA \leq CNT < CCRxB$ ◆ For sawtooth wave down-counting, the output is low level when $CCRxA \leq CNT < CCRxB$ ◆ For triangular wave up-counting, the output is low level when $CNT < CCRxA$; for down-counting, the output is high level when $CNT > CCRxB$	
	FLTBO	Capture function: CH0B input channel filter setup Refer to FLTBK setting	RW
3	CCPA0	Compare function: output compare mode CCPAx compare output CHAx port polarity control ● 0b0: Normal output ● 0b1: Inverted output	RW
	CCPA0	Encode count and slave mode gating function: input phase control CCPA0 slave mode gating, reset, external trigger, external clock using CH0A port input polarity control ● 0b0: Normal input ● 0b1: Inverted input	RW
2:0	OCMA0	Compare function: channel A compare control Refer to OCMB0 setting	RW
	FLTA0	Capture function: CH0A input channel filter setup Refer to FLTBK setting	RW

12.5.4.8 ADC Trigger Control Register (ATIM3_ADTR)

Offset	Bit Position																																							
0x020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
Reset	0x00000000																																							
Name	Reserved																								ADTE		CMB2E		CMB1E		CMB0E		CMA2E		CMA1E		CMA0E		UEVE	
Access																									RW		RW		RW		RW		RW		RW		RW		RW	

Bit	Name	Functional Description	Access
31:8	Reserved	-	-
7	ADTE	Enable ADC trigger global control 0b0: Disable 0b1: Enable	RW
6	CMB2E	Channel 2B compare match trigger ADC enable 0b0: Disable 0b1: Enable	RW
5	CMB1E	Channel 1B compare match trigger ADC enable 0b0: Disable 0b1: Enable	RW
4	CMB0E	Channel 0B compare match trigger ADC enable 0b0: Disable 0b1: Enable	RW
3	CMA2E	Channel 2A compare match trigger ADC enable 0b0: Disable 0b1: Enable	RW
2	CMA1E	Channel 1A compare match trigger ADC enable 0b0: Disable 0b1: Enable	RW
1	CMA0E	Channel 0A compare match trigger ADC enable 0b0: Disable 0b1: Enable	RW
0	UEVE	Event update trigger ADC enable 0b0: Disable 0b1: Enable	RW

12.5.4.9 Channel x Control Register (ATIM3_CRCHx) (x=0/1/2)

Offset: 0x024 (CRCH0), 0x028 (CRCH1), 0x02C (CRCH2)

Offset	Bit Position																																																
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																	
Reset	0x00003000																																																
Name	Reserved																																																
Access																	WO	WO	RW																														

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15	CCGB	Capture compare B software trigger Hardware auto-clear, writing 1 triggers the following actions, writing 0 has no effect: ● In compare mode, generates interrupt only ● In capture mode, generates interrupt and captures counter value to capture register	WO
14	CCGA	Capture compare A software trigger Hardware auto-clear, writing 1 triggers the following actions, writing 0 has no effect: ● In compare mode, generates interrupt only ● In capture mode, generates interrupt and captures counter value to capture register	WO
13:12	CISB	B channel compare match setting ● 0b00: No match ● 0b01: Compare match during up-counting ● 0b10: Compare match during down-counting ● 0b11: Compare match during both up/down counting	RW
11:10	Reserved	-	-
9	CIEB	B capture compare trigger interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
8	CIEA	A capture compare trigger interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
7	BUFEB	Compare function: B compare cache enable control ● 0b0: Disable ● 0b1: Enable	RW

Bit	Name	Functional Description	Access
6	BUFEA	Compare function: A compare cache enable control ● 0b0: Disable ● 0b1: Enable	RW
5	CSB	B channel capture/compare function selection ● 0b0: Compare mode ● 0b1: Capture mode	RW
4	CSA	A channel capture/compare function selection ● 0b0: Compare mode ● 0b1: Capture mode	RW
3:2	CFB_CRB	Capture function: channel B rising falling edge capture enable ● 0b00: Disable ● 0b01: Rising edge capture ● 0b10: Falling edge capture ● 0b11: Both rising and falling edge capture	RW
	BKSB	Compare function: channel B output brake level control ● 0b00: High-impedance output ● 0b01: No effect on output ● 0b10: Force output low level ● 0b11: Force output high level  Note No DEBUG_STOP mode.	RW
1:0	CFA_CRA	Capture function: channel A rising falling edge capture enable ● 0b00: Disable ● 0b01: Rising edge capture ● 0b10: Falling edge capture ● 0b11: Both rising and falling edge capture	RW
	BKSA	Compare function: channel A output brake level control ● 0b00: High-impedance output ● 0b01: No effect on output ● 0b10: Force output low level ● 0b11: Force output high level  Note In DEBUG_STOP mode, the brake level control for both channel A and channel B is governed by BKSA.	RW

12.5.4.10 Dead Time Register (ATIM3_DTR)

Offset	Bit Position																															
0x030	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																		VCE	SAFEEN	MOE	AOE	BKE	DTEN	BKSEL	DTR						
Access																			RW	RW	RW	RW	RW	RW	RW							RW

Bit	Name	Functional Description	Access																				
31:15	Reserved	-	-																				
14	VCE	VC brake enable 0b0: Disable 0b1: Enable	RW																				
13	SAFEEN	Safety brake enable (lockup) 0b0: Disable 0b1: Enable	RW																				
12	MOE	PWM output enable 0b0: Disable 0b1: Enable	RW																				
11	AOE	PWM output automatically enable 0b0: Disable 0b1: Enable	RW																				
10	BKE	Port brake enable 0b0: Disable 0b1: Enable	RW																				
9	DTEN	Dead time control enable 0b0: Disable 0b1: Enable	RW																				
8	BKSEL	Reserved	RW																				
7:0	DTR	Dead time <table border="1"> <thead> <tr> <th>DTR</th><th>T</th><th>dt</th><th>step</th></tr> </thead> <tbody> <tr> <td>DTR[7]=0b0</td><td>DTR[6:0] + 2</td><td>2-129</td><td>1</td></tr> <tr> <td>DTR[7:6]=0b10</td><td>{DTR[5:0] + 64} * 2 + 2</td><td>130-256</td><td>2</td></tr> <tr> <td>DTR[7:5]=0b110</td><td>{DTR[4:0] + 32} * 8 + 2</td><td>258-506</td><td>8</td></tr> <tr> <td>DTR[7:5]=0b111</td><td>{DTR[4:0] + 32} * 16 + 2</td><td>514-1010</td><td>16</td></tr> </tbody> </table>	DTR	T	dt	step	DTR[7]=0b0	DTR[6:0] + 2	2-129	1	DTR[7:6]=0b10	{DTR[5:0] + 64} * 2 + 2	130-256	2	DTR[7:5]=0b110	{DTR[4:0] + 32} * 8 + 2	258-506	8	DTR[7:5]=0b111	{DTR[4:0] + 32} * 16 + 2	514-1010	16	RW
DTR	T	dt	step																				
DTR[7]=0b0	DTR[6:0] + 2	2-129	1																				
DTR[7:6]=0b10	{DTR[5:0] + 64} * 2 + 2	130-256	2																				
DTR[7:5]=0b110	{DTR[4:0] + 32} * 8 + 2	258-506	8																				
DTR[7:5]=0b111	{DTR[4:0] + 32} * 16 + 2	514-1010	16																				

12.5.4.11 Repeat Cycle Setting Value Register (ATIM3_RCR)

Offset	Bit Position																																	
0x034	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																						UND		OV		RCR							
Access																							RW		RW		RW							

Bit	Name	Functional Description	Access
31:10	Reserved	-	-
9	UND	Underflow mask control for repeating counting 0b0: Down-counting overflow enable 0b1: Down-count overflow mask	RW
8	OV	Overflow mask control for repetiting counting 0b0: Up-counting overflow enable 0b1: Up-counting overflow mask	RW
7:0	RCR	Repetition period count value Sets RCR+1 periods. The overflow controlled by [9:8] generates an update event. When the counter overflows up or down, the internal RCR_CNT decrements by 1. When counting reaches zero, RCR_CNT reloads the RCR value and generates an update event UEV signal.	RW

12.5.4.12 Channel x Compare Capture Register (ATIM3_CCRxy) (x=0/1/2, y=A/B)

Offset: 0x03C (CCR0A), 0x040 (CCR0B), 0x044 (CCR1A), 0x048 (CCR1B), 0x04C (CCR2A), 0x050 (CCR2B)

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																CCRxy															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	CCRxy	Compare capture register Comparison with cache function	RW

13 RTC

13.1 Introduction

The Real Time Counter provides seconds, minutes, hours, day, week, month, and year information, and the number of days in a month and leap year can be automatically adjusted. Clock operation can be done in 24 or 12 hour format via the RTC_CR0. AMPM register bit.

13.2 Features

Table 13-1 Basic Features of RTC

Basic Features	Description
Clock source	Off-chip low-speed crystal oscillator XTL (32.768kHz) On-chip low-speed oscillator RCL
Basic function	Can calculate seconds, minutes, hours, days, weeks, months, years between 00 and 99
	Automatic leap year adjustment
	Configurable to 24 or 12 hour format
	Programmable start or stop
	Provides two sets of alarm functions
	With high-precision 1Hz square wave signal output
	Configurable wake-up timer
	Timestamp function
Interrupt	Supports configurable periodic interrupts with a period ranging from 0.5 to 32 seconds.
	Alarm interrupt A/B with down-to-the-second precision
	60μs-145Hour wake-up timer interrupt
	Timestamp interrupt

13.3 Functional Description

The clock source of the Real Time Counter can be configured as an external low-speed crystal oscillator, or the internal low-speed clock RCL; by default, the external low-speed crystal oscillator is used. The RTC_SEC, RTC_MIN, RTC_HOUR, RTC_DAY, RTC_WEEK, RTC_MON, RTC_YEAR, RTC_ALMA, and RTC_ALMB registers are only affected by power-on reset. Their states are undefined after power-on and require initialization. Other registers are only controlled by power-on reset and module reset, and other reset sources cannot reset these registers.

All date and time values written to or read from software are in BCD format and do not require hexadecimal-to-decimal conversion.

Any invalid date time will not be able to be written, such as 32nd day, 25th hour, 70th second, month B, etc.

13.3.1 Function Block Diagram

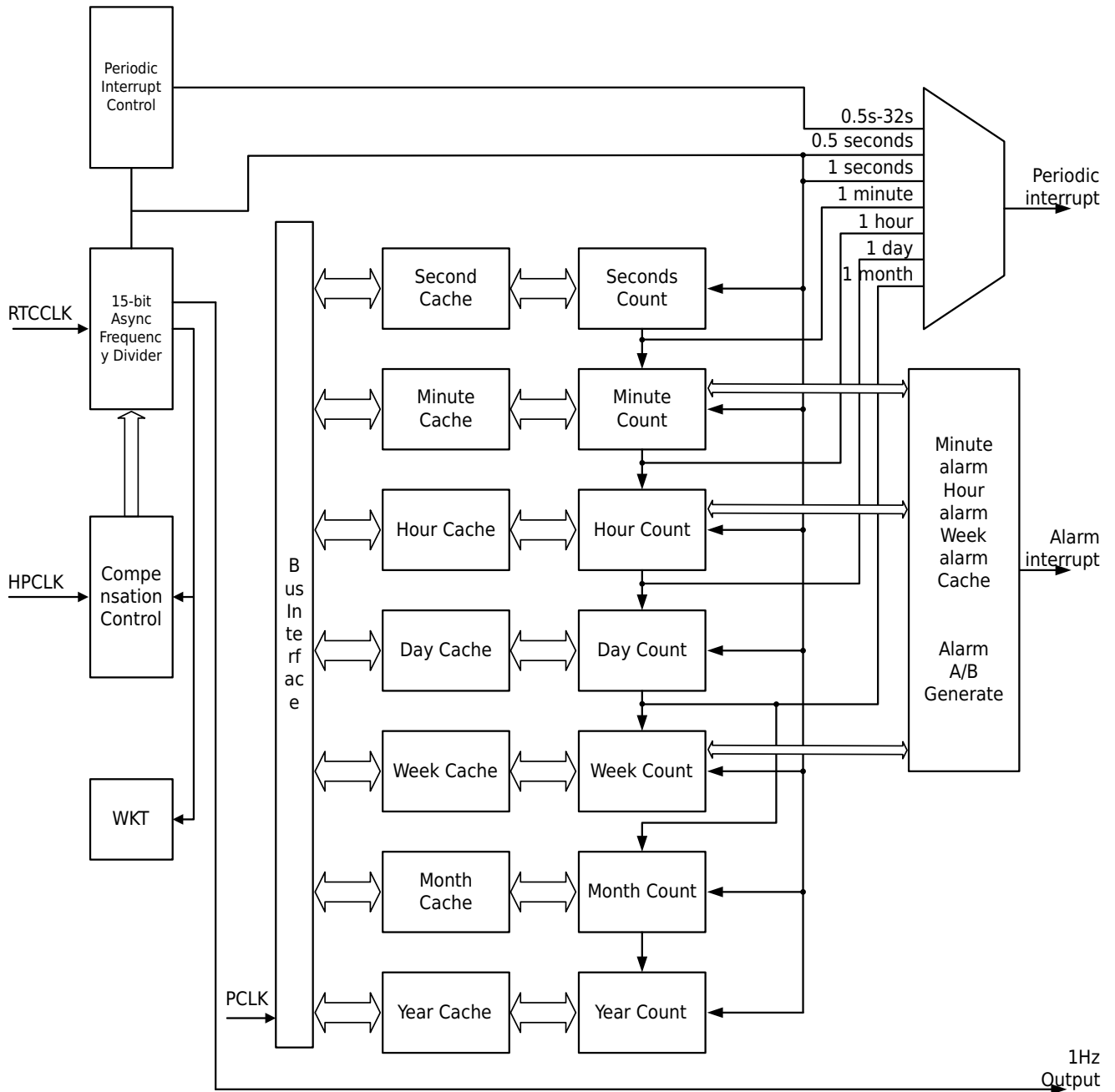


Figure 13-1 RTC Block Diagram

13.3.2 Power-on Setting

The RTC is reset once after power-on. When the system remains powered, no external reset requests can reset the RTC. The RTC will continue counting without interruption. After power-on, configure the initial calendar values, alarm settings, error compensation, and interrupts, and then start the RTC.

13.3.3 Clock Error Compensation

Due to the inherent error in the external crystal oscillator, compensation is required to achieve high-precision counting results. Two compensation methods are available: one based on the internal clock **RTCCLK** error compensation (accuracy of 0.96ppm or 0.06ppm); and the other based on high-speed clock **HPCLK** error compensation.

Principle and Calculation of Internal Clock Error Compensation (0.96ppm)

Since the counter uses a 32.768kHz clock to count, if it is necessary to compensate the precision per second, it can only be compensated according to the integer period of 32.768kHz, and the minimum unit of compensation per second is $(1/32768) \times 10^6 = 30.5\text{ppm}$, unable to meet the high precision requirements.

Then, when realizing high-precision clock compensation under the counting clock of 32.768kHz, it is necessary to adjust the algorithm to expand the maximum compensation period by 32 times. Then, when the minimum unit that can only be compensated is 30.5ppm, the average compensation unit per second becomes $30.5\text{ppm}/32 = 0.96\text{ppm}$. To meet the requirements of higher precision clock compensation and ensure the compensation is distributed evenly within each 32-second interval, this register introduces a setting with 5 fractional bits.

For example, when the 1Hz clock is directly output in the default state, the compensation target value is calculated by measuring the accuracy of the clock.

Assuming that the actual measured value is 1.0000203Hz, then:

Actual Vibration Frequency = $32768 \times 1.0000203 \approx 32768.6651904$ (actual oscillation frequency needs to be accurate to at least three decimal places)

$$\begin{aligned}\text{Compensation Target Value} &= (\text{Actual Vibration Frequency} - \text{Target Frequency}) / \text{Target Frequency} \times 10^6 \\ &= (32768.6651904 - 32768) / 32768 \times 10^6 \\ &= 20.3\text{ppm}\end{aligned}$$

Based on the formula:

$$\text{CR}[8:0] = \left(\frac{\text{Compensation Target Value [ppm]} \times 2^{15}}{10^6} \right)_{\text{Take 2's complement}} + 0001.00000B$$

If the compensation target value is +20.3ppm, calculate the corresponding register value as follows:

$$\begin{aligned}\text{CR}[8:0] &= (20.3 \times 2^{15}/10^6)_{\text{Take 2's complement}} + 0001.00000B \\ &= (0.6651904)_{\text{Take 2's complement}} + 0001.00000B \\ &= 0000.10101B + 0001.00000B \\ &= 0001.10101B\end{aligned}$$

If the compensation target value is -20.3ppm, calculate the corresponding register value as follows:

$$\begin{aligned}\text{CR}[8:0] &= (-20.3 \times 2^{15}/10^6)_{\text{Take 2's complement}} + 0001.00000B \\ &= (-0.6651904)_{\text{Take 2's complement}} + 0001.00000B \\ &= 1111.01011B + 0001.00000B \\ &= 0000.01011B\end{aligned}$$

Principle and Calculation of Internal Clock Error Compensation (0.06ppm)

Achieving a compensation accuracy of 0.06ppm requires extending the maximum compensation period by a factor of 512. Then, when the minimum unit that can only be compensated is 30.5ppm, the average compensation unit per second becomes $30.5\text{ppm}/512 = 0.06\text{ppm}$. It meets the clock compensation requirement with higher precision. And the compensation occurs in a relatively uniform range every 512 seconds. Hence, 9 decimal places is introduced in this register.

For example, when the 1Hz clock is directly output in the default state, the compensation target value is calculated by measuring the accuracy of the clock.

Assuming that the actual measured value is 0.9999888Hz, then:

Actual Vibration Frequency = $32768 \times 0.9999888 \approx 32767.6329984$ (actual oscillation frequency needs to be accurate to at least five decimal places)

$$\begin{aligned}\text{Compensation Target Value} &= (\text{Actual Vibration Frequency} - \text{Target Frequency}) / \text{Target Frequency} \times 10^6 \\ &= (32767.6329984 - 32768) / 32768 \times 10^6 \\ &= -11.29\text{ppm}\end{aligned}$$

Based on the formula:

$$\text{CR}[12:0] = \left(\frac{\text{Compensation Target Value [ppm]} \times 2^{15}}{10^6} \right)_{\text{Take 2's complement}} + 0001.000000000B$$

If the compensation target value is -11.29ppm, calculate the corresponding register value as follows:

$$\begin{aligned}\text{CR}[12:0] &= (-11.29 \times 2^{15} / 10^6)_{\text{Take 2's complement}} + 0001.000000000B \\ &= (-0.37)_{\text{Take 2's complement}} + 0001.000000000B \\ &= 1111.101000011B + 0001.000000000B \\ &= 0000.101000011B\end{aligned}$$

Principle and Calculation of Error Compensation Based on High-Speed Clock:

The calculation method of this method is the same as the error compensation based on its own clock. With the introduction of the high-speed clock HPCLK (PCLK), the 1/32768-second error that originally needed to be accumulated within a maximum of 32 seconds can be dispersed to every 1 second. By applying compensation every 1 second, a high-precision 1Hz clock output is achieved on average.

The correct high-speed compensation parameter value RTC_HCOMP.PARA must be set according to the frequency.

$$\text{RTC_HCOMP.PARA} = f/32768/32$$

f is the compensation frequency, and it is recommended to configure it as an integer multiple of 32.768 kHz.

- 0.06ppm compensation accuracy: the valid range for f is 16 to 48MHz.
- 0.96ppm compensation accuracy: the valid range for f is 4 to 48MHz.

13.3.4 Timestamp Function

Timestamp is enabled by setting RTC_CR2.TSE to 1.

When a timestamp event is detected on the RTC_TS pin, the calendar is saved to the timestamp registers (RTC_TSTR and RTC_TSDR).

When a timestamp event occurs, the RTC_IFR.TS (timestamp flag) is set to 1.

An interrupt can be generated when a timestamp detection event occurs by setting RTC_IER.TS to 1.

If a new timestamp event is detected while the timestamp flag (TS) is already set to 1, the timestamp overflow flag (TSOV) will be set to 1 and the timestamp registers (RTC_TSTR and RTC_TSDR) will hold the results of the previous event.

13.3.5 Wake-up Timer

Periodic wake-up flags are generated by a 16-bit programmable auto-reload down counter. This wake-up function can be enabled through the RTC_CR2.WUTE register.

The clock input to the wake-up timer can be:

- RTC clock (RTCCLK) divided by 2, 4, 8 or 16.

- Second pulse divided by 1, 2, 4 or 8.

After initialization is completed, the timer starts counting down. When the wake-up function is enabled in low power mode, the down count remains active. Additionally, when the counter reaches zero, the RTC_IFR.WU register is set to 1, and the wake-up register automatically reloads using the value of the RTC_WUTR register.

The WU flag must then be cleared in software.

The pause wake-up timer will not generate a reload action. If retiming is required, a write action must be generated to the RTC_WUTR register.

When the periodic wake-up interrupt is enabled by setting the RTC_IER.WU register to 1, it takes the device out of low power mode.

System resets and low power mode (Sleep, DeepSleep) have no effect on the wake-up timer.

After the RTC counting has started, if the system needs to enter a low power mode, it is necessary to ensure that the register information has been synchronized to the RTC. Please perform any of the following confirmations before switching modes.

- After configuring the RTC registers, wait for at least two RTC count clocks before switching modes.
- After configuring the RTC registers, set RTC_CR1.WAIT=1 and check that RTC_CR1.WAITF=1. Then set RTC_CR1.WAIT=0, query RTC_CR1.WAITF=0 and then switch modes.

13.3.6 RTC Interrupt

RTC supports 4 interrupt types. Alarm A/B interrupt, periodic interrupt, timestamp interrupt, and wake-up timer interrupt.

Alarm A/B Interrupt

When RTC_IER.ALMx=1, if the current calendar time is equal to the alarm register, the alarm interrupt is triggered. By setting the MSK related bits, you can shield the comparison of the corresponding alarm time bits.

Periodic Interrupt

When RTC_IER.PR=1, a period interrupt is triggered after the selected period occurs. Since the interrupt is shared, it is distinguished by flag register bits.

Wake-up Timer Interrupt

When RTC_IER.WU=1, an interrupt is triggered when the wake-up timer counter reaches zero. Since the interrupt is shared, it is distinguished by flag register bits.

Timestamp Interrupt

When RTC_IER.TS=1, an interrupt is generated upon a timestamp trigger. Since the interrupt line is shared, it is distinguished by flag register bits.

13.4 Operation Example

13.4.1 RTC Count Start Setting

Procedure

- Step1.** Set RTC_CR0.START=0 to stop counting.
- Step2.** Set RTC_CR0.AMPM and RTC_CR0.PRDS; RTC_CR0.PRDX configures the time format and interrupt period.
- Step3.** Set RTC_CR1.CKSEL to select the RTC timing clock.

- Step4.** Set the calendar counter registers for seconds, minutes, hours, week, day, month, and year.
- Step5.** When clock error compensation is required, set the counting clock error compensation register RTC_COMPEN.
- Step6.** Clear the interrupt flag bits register RTC_IFR, and enable the interrupts.
- Step7.** Set RTC_CR0.START=1 to start counting.

13.4.2 Read Count Register

Context

There are three methods to read the count register.

Procedure

- **Read Method 1 at Any Time**

- Step1.** Set RTC_CR1.WAIT=1 to stop the calendar register counting and enter read/write mode.
- Step2.** Query until RTC_CR1.WAITF=1.
- Step3.** Read the values of the second, minute, hour, week, day, month, and year count registers.
- Step4.** Set RTC_CR1.WAIT=0 to start the counter counting.
- Step5.** Query until RTC_CR1.WAITF=0.

- **Read Method 2 at Any Time**

- Step1.** Read minutes, hours, weeks, days, months, and year count register values.
- Step2.** Read out the second count register value.
- Step3.** Read the second count register value again.
- Step4.** Judging whether the reading values of the two seconds are the same, if they are different, start from the first step again, if the same, end read.

- **Interrupt Read Method**

Read the second, minute, hour, week, day, month, year count register value in the RTC cycle interrupt service. Because after the interrupt occurs, it will take at least 0.5s for the next data change.

13.4.3 Write Count Register

Context

Changing the seconds register in the counting mode will reset the seconds counting, writing minutes, hours, weeks, days, months, years count register values will not affect the RTC counting.



Note

The RTC register is write protected. Please refer to the RTC_WPR register to remove the write protection before writing.

WAIT is not required to write seconds, minutes, hours, weeks, days, months, and year count registers in RTC disabled mode.

Procedure

- Step1.** Set RTC_CR1.WAIT=1, stop the calendar register counting, and enter the read and write mode.
- Step2.** Query until RTC_CR1.WAITF=1.
- Step3.** Write the second, minute, hour, week, day, month, year count register value.
- Step4.** Set RTC_CR1.WAIT=0, the counter starts counting again.



Notice

All write operations must be completed within 1 second.

Step5. Query until RTC_CR1.WAITF=0.

13.4.4 Alarm Clock Setting

Procedure

- Step1.** Set RTC_CR2.ALMxE=0 to disable the alarm.
- Step2.** Set RTC_IER.ALMx=1 to enable the alarm interrupt.
- Step3.** Alarm RTC_ALMx setting, configure the alarm time as needed;
- Step4.** Set RTC_CR2.ALMxE=1 to enable the alarm interrupt.
- Step5.** Wait for an interrupt to occur.
- Step6.** Since the alarm clock interrupt and the fixed-period interrupt share the interrupt request signal, when RTC_IFR.ALMx=1, enter the alarm clock interrupt processing; otherwise, enter the fixed-period interrupt processing.

13.4.5 1Hz Output

Context

RTC can choose to output three types of 1Hz clocks: general precision, higher precision and high precision. When the clock error compensation function is valid, it outputs a high-precision 1Hz clock; when using HPCLK with different frequencies, it outputs a high-precision 1Hz clock. The system control register must be configured according to the HPCLK frequency. The three methods for 1Hz output are as follows:

Procedure

- **General Precision 1Hz Output (without Clock Compensation)**
 - Step1.** Set RTC_CR0.START=0 to stop counting;
 - Step2.** RTC output pin setting.
 - Step3.** Set RTC_CR0.HZ1OE=1 to enable clock output.
 - Step4.** Set RTC_CR0.START=1, counting starts.
 - Step5.** Wait for more than 2 count cycles.
 - Step6.** 1Hz output starts.
- **Higher Precision 1Hz Output (Low-speed Compensation)**
 - Step1.** Set RTC_CR0.START=0 to stop counting;
 - Step2.** RTC output pin setting.
 - Step3.** Set RTC_CR0.HZ1OE=1 to enable clock output.
 - Step4.** Configure RTC_COMPEN.CR to set the compensation value.
 - Step5.** Set RTC_COMPEN.EN=1 to enable error compensation.
 - Step6.** Set RTC_CR0.START=1, counting starts.
 - Step7.** Wait for more than 2 count cycles.
 - Step8.** 1Hz output starts.
- **High Precision 1 Hz Output (Based on Higher Precision Output, Providing the RTC with a High-Speed Compensation Clock)**
 - Step1.** Set RTC_CR0.START=0 to stop counting;
 - Step2.** RTC output pin setting.
 - Step3.** Set RTC_CR0.HZ1OE=1 to enable clock output.
 - Step4.** Set RTC_CR0.HZ1SEL=1, choose to output high-precision 1Hz clock.
 - Step5.** Set the high-speed compensation parameter RTC_HCOMP.PARA according to the configured frequency.

- Step6.** Configure RTC_COMPEN.CR[12:0] to set the compensation.
- Step7.** Set RTC_COMPEN.EN=1 to enable error compensation.
- Step8.** Set RTC_COMPEN.CSTEPS=0 to enable 0.96ppm accuracy compensation; or set RTC_COMPEN.CSTEPS=1 to enable 0.06ppm accuracy compensation.
- Step9.** Set RTC_CR0.START=1, counting starts.
- Step10.** Wait for more than 2 count cycles.
- Step11.** 1Hz output starts.

13.5 Register Description

13.5.1 Register List

Base address: 0x40001400

Table 13-2 Real Time Counter (RTC) Register Offset

Offset	Register	Description	Width
0x000	RTC_CR0	Control Register 0	32
0x004	RTC_CR1	Control Register 1	32
0x008	RTC_SEC	Second Count Register	32
0x00C	RTC_MIN	Minute Count Register	32
0x010	RTC_HOUR	Hour Count Register	32
0x014	RTC_WEEK	Week Count Register	32
0x018	RTC_DAY	Day Count Register	32
0x01C	RTC_MON	Month Count Register	32
0x020	RTC_YEAR	Year Count Register	32
0x030	RTC_COMPEN	Clock Error Compensation Register	32
0x034	RTC_HCOMP	High-Speed Compensation Accuracy Parameter Register	32
0x03C	RTC_CR2	Control Register 2	32
0x040	RTC_ALMA	Alarm A Register	32
0x044	RTC_ALMB	Alarm B Register	32
0x048	RTC_TSTR	Timestamp Time Register	32
0x04C	RTC_TSDR	Timestamp Date Register	32
0x050	RTC_WUTR	Wake-Up Timer Register	32
0x054	RTC_IER	Interrupt Enable Register	32
0x058	RTC_IFR	Interrupt Flag Register	32
0x05C	RTC_ICR	Interrupt Clear Register	32
0x060	RTC_WPR	Protection Register	32

13.5.2 Control Register 0 (RTC_CR0)

Offset	Bit Position															
0x000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0x00000000 (Only power-on reset and module reset are effective for resetting this register)															
Name	Reserved								PRDSEL	PRDX				START	HZ1SEL	HZ1OE
Access									RW	RW				RW	RW	RW

Bit	Name	Functional Description	Access
31:15	Reserved	-	-
14	PRDSEL	Periodic interrupt source selection 0b0: Use the periodic interrupt time interval set by PRDS 0b1: Use the periodic interrupt time interval set by PRDX	RW
13:8	PRDX	Set the time interval for generating periodic interrupts The settable range is from 0.5 seconds to 32 seconds, and the step is 0.5 seconds. 0b000000: 0.5 seconds 0b000001: 1 second 0b111110: 31.5 seconds 0b111111: 32 seconds	RW
7	START	RTC count enable 0b0: Stop the RTC count 0b1: Enable the RTC count	RW
6	HZ1SEL	1Hz output selection 0b0: Normal precision 1Hz output 0b1: High precision 1Hz output	RW
5	HZ1OE	1Hz output enable 0b0: Disabled 1Hz output 0b1: Enable 1Hz output	RW
4	Reserved	-	-
3	AMPM	Time format selection 0b0: 12-hour time format 0b1: 24-hour time format	RW

Bit	Name	Functional Description	Access
2:0	PRDS	Set the time interval for generating interrupts ● 0b000: No periodic interrupt generated ● 0b001: 0.5 seconds ● 0b010: 1 second ● 0b011: 1 minute (At 00 seconds every minute) ● 0b100: 1 hour (At 00:00 every hour) ● 0b101: 1 day (At 00:00:00 every day) ● 0b110/0b111: 1 month (At 00:00:00 on the first day of every month)  Notice If you need to write and change the time interval of periodic interrupt when START=1, the operation steps are as follows: 1. Turn off the RTC interrupt in the NVIC; 2. Change the time interval of periodic interrupt; 3. Clear the RTC interrupt flag; 4. Enable RTC interrupt.	RW

13.5.3 Control Register 1 (RTC_CR1)

Offset	Bit Position																																
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000 (Only power-on reset and module reset are effective for this register reset)																																
Name	Reserved																						CKSEL		Reserved					WAITF		WAIT	
Access																							RW							RO		RW	

Bit	Name	Functional Description	Access
31:11	Reserved	-	-
10:8	CKSEL	RTC clock selection 0b000/0b001: XTL 32.768k 0b010/0b011: RCL 32k - Others: Setting prohibited	RW
7:2	Reserved	-	-
1	WAITF	Read and write wait valid flag 0b0: Not write/read status 0b1: Write/read status  Notice Flag indicating whether the WAIT bit setting is effective. Please confirm that this bit is set to "1" before writing or reading. During the counting process, the bit is cleared to "0" after the WAIT bit is cleared to "0" and waits for the writing to be completed.	RO
0	WAIT	Read and write wait setting 0b0: Normal Counting Mode 0b1: Write/read mode  Notice Please set this bit to "1" when writing/reading. Since the counter is counting continuously, please complete the writing/reading operation within 1 second and clear this bit to "0".	RW

13.5.4 Second Count Register (RTC_SEC)

Offset	Bit Position																															
0x008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved																								SECH				SECL			
Access																									RW				RW			

Bit	Name	Functional Description	Access
31:7	Reserved	-	-
6:4	SECH	Second count tens digit BCD code of second tens digit count value	RW
3:0	SECL	Second count units digit BCD code of second units digit count value	RW

Indicates 0-59 seconds, using decimal counting. Please write the BCD code in decimal 0 to 59. Values written outside this range will be ignored.

13.5.5 Minute Count Register (RTC_MIN)

Offset	Bit Position																															
0x00C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved																								MINH				MINL			
Access																									RW				RW			

Bit	Name	Functional Description	Access
31:7	Reserved	-	-
6:4	MINH	Minute count tens digit BCD code of minute tens digit count value	RW
3:0	MINL	Minute count units digit BCD code of minute units digit count value	RW

Indicates 0-59 points, using decimal counting. Please write the BCD code in decimal 0 to 59. Values written outside this range will be ignored.

13.5.6 Hour Count Register (RTC_HOUR)

Offset	Bit Position																			
0x010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12
Reset	0xFFFFFFFF																			
Name	Reserved																HOURH		HOURL	
Access																	RW		RW	

Bit	Name	Functional Description	Access
31:6	Reserved	-	-
5:4	HOURH	Hour count tens digit BCD code of hour tens digit count value	RW
3:0	HOURL	Hour count units digit BCD code of hour units digit count value	RW

In 24-hour format, it means 0-23 hours.

In 12-hour format, it means 0-12 hours. Bit5=0 means AM, then 01-12 means morning; Bit5=1 means PM, then 01-12 means afternoon.

Please set the correct decimal BCD code according to the value of the control bit RTC_CR0.AMPM. The value written out of range will be ignored.

For specific time values, please refer to the following table:

24-Hour Clock Format (AMPM=1)		12-Hour Clock Format (AMPM=0)	
Time	Register Value	Time	Register Value
00 o'clock	0x00	AM 12 o'clock	0x12
01 o'clock	0x01	AM 01 o'clock	0x01
02 o'clock	0x02	AM 02 o'clock	0x02
03 o'clock	0x03	AM 03 o'clock	0x03
04 o'clock	0x04	AM 04 o'clock	0x04
05 o'clock	0x05	AM 05 o'clock	0x05
06 o'clock	0x06	AM 06 o'clock	0x06
07 o'clock	0x07	AM 07 o'clock	0x07
08 o'clock	0x08	AM 08 o'clock	0x08
09 o'clock	0x09	AM 09 o'clock	0x09
10 o'clock	0x10	AM 10 o'clock	0x10
11 o'clock	0x11	AM 11 o'clock	0x11
12 o'clock	0x12	PM 12 o'clock	0x32

24-Hour Clock Format (AMPM=1)		12-Hour Clock Format (AMPM=0)	
Time	Register Value	Time	Register Value
13 o'clock	0x13	PM 01 o'clock	0x21
14 o'clock	0x14	PM 02 o'clock	0x22
15 o'clock	0x15	PM 03 o'clock	0x23
16 o'clock	0x16	PM 04 o'clock	0x24
17 o'clock	0x17	PM 05 o'clock	0x25
18 o'clock	0x18	PM 06 o'clock	0x26
19 o'clock	0x19	PM 07 o'clock	0x27
20 o'clock	0x20	PM 08 o'clock	0x28
21 o'clock	0x21	PM 09 o'clock	0x29
22 o'clock	0x22	PM 10 o'clock	0x30
23 o'clock	0x23	PM 11 o'clock	0x31

13.5.7 Day Count Register (RTC_DAY)

Offset	Bit Position																															
0x018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0XXXXXXXXX																															
Name	Reserved																										DAYH		DAYL			
Access																											RW		RW			

Bit	Name	Functional Description	Access
31:6	Reserved	-	-
5:4	DAYH	Day count tens digit value BCD code of day tens digit count value	RW
3:0	DAYL	Day count units digit BCD code of day units digit count value	RW

Means 1 to 31 days, uses decimal counting, automatically calculates leap year and month. Please write the BCD code in decimal 1 to 31. Values written outside the range will be ignored.

Refer to the table below for the specific range of day values.

Month	Day Count Value
February (normal year)	01-28
February (leap year)	01-29
April, June, September, November	01-30
January, March, May, July, August, October , December	01-31

13.5.8 Week Count Register (RTC_WEEK)

Offset	Bit Position																															
0x014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0XXXXXXXXX																															
Name	Reserved																															WEEK
Access																																RW

Bit	Name	Functional Description	Access
31:3	Reserved	-	-
2:0	WEEK	Week count value BCD code of week count value	RW

Decimal 0-6 means Sunday-Saturday. Please write the BCD code in decimal 0 to 6. Values written outside the range will be ignored.

For specific week count values, please refer to the following table:

Week	Week Count Value
Sunday	0x00
Monday	0x01
Tuesday	0x02
Wednesday	0x03
Thursday	0x04
Friday	0x05
Saturday	0x06

13.5.9 Month Count Register (RTC_MON)

Offset	Bit Position																															
0x01C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0XXXXXXXXX																															
Name	Reserved																								MON							
Access																									RW							

Bit	Name	Functional Description	Access
31:5	Reserved	-	-
4:0	MON	Month count value BCD code of month count value	RW

Indicates 1-12 months, using decimal counting. Please write the BCD code in decimal 1 to 12. Values written outside the range will be ignored.

13.5.10 Year Count Register (RTC_YEAR)

Offset	Bit Position																															
0x020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0XXXXXXXXXX																															
Name	Reserved																								YEARH				YEARL			
Access																									RW				RW			

Bit	Name	Functional Description	Access
31:8	Reserved	-	-
7:4	YEARH	Year conut tens digit BCD code of year tens digit count value	RW
3:0	YEARL	Year conut units digit BCD code of year units digit count value	RW

Indicates 0 to 99 years, using decimal counting, counting according to the month carry. Automatically calculate leap years such as: 00, 04, 08, ..., 92, 96, etc. Please write the BCD code in decimal 0-99. Values written outside the range will be ignored.

13.5.11 Clock Error Compensation Register (RTC_COMPEN)

Offset	Bit Position																																		
0x030	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00000020 (Only power-on reset and module reset are effective for this register reset)																																		
Name	Reserved																		EN	CSTEPS	Reserved	CR													
Access																																			

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15	EN	Compensation enable 0b0: Disable clock error compensation 0b1: Enable clock error compensation	RW
14	CSTEPS	Compensation accuracy 0b0: Compensation accuracy 0.96 ppm 0b1: Compensation accuracy 0.06 ppm	RW
13	Reserved	-	-
12:0	CR	Compensation value CSTEPS=0b0: By setting the compensation value, an accuracy compensation of $\pm 0.96\text{ppm}$ can be performed per second. The compensation value is 9-bit 2's complement code with a decimal point, and the last 5 bits are the decimal part. Compensation range: 274.6ppm to 212.6ppm, with a minimum differential error of $\pm 0.48\text{ppm}$ and a minimum resolution of 0.96ppm. CSTEPS=0b1: By setting the compensation value, an accuracy compensation of $\pm 0.05960\text{ppm}$ can be performed per second. The compensation value is 13-bit 2's complement code with a decimal point, and the last 9 bits are the decimal part. Compensation range: 274.65820ppm to 213.56344ppm, with a minimum differential error of $\pm 0.0298\text{ppm}$ and a minimum resolution of 0.05960ppm.	RW

For 0.96ppm more compensation accuracy, see the following table:

Compensation Value Setting										Compensation Value
EN	CR[8:0]									
1	1	0	0	0	0	0	0	0	0	-274.6ppm
	1	0	0	0	0	0	0	0	1	-273.7ppm
	...									...
	0	0	0	0	1	1	1	1	1	-0.96ppm
	0	0	0	1	0	0	0	0	0	0ppm
	...									...
	0	1	1	1	1	1	1	1	0	+211.7ppm
	0	1	1	1	1	1	1	1	1	+212.6ppm
0	X	X	X	X	X	X	X	X	X	Uncompensated

For 0.06ppm more compensation accuracy, see the following table:

Compensation Value Setting														Compensation Value
EN&CSTEPS	CR[12:0]													
1	1	0	0	0	0	0	0	0	0	0	0	0	0	-274.65820ppm
	1	0	0	0	0	0	0	0	0	0	0	0	1	-274.59859ppm
	...													...
	0	0	0	0	1	1	1	1	1	1	1	1	1	-0.059605ppm
	0	0	0	1	0	0	0	0	0	0	0	0	0	0ppm
	...													...
	0	1	1	1	1	1	1	1	1	1	1	1	0	+213.50383ppm
	0	1	1	1	1	1	1	1	1	1	1	1	1	+213.56344ppm
0	X	X	X	X	X	X	X	X	X	X	X	X	X	Uncompensated

13.5.12 High-Speed Compensation Accuracy Parameter Register (RTC_HCOMP)

Offset	Bit Position																															
0x034	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000 (Only power-on reset and module reset are effective for this register reset)																															
Name	Reserved																								PARA							
Access																									RW							

Bit	Name	Functional Description	Access
31:6	Reserved	-	-
5:0	PARA	Compensation parameters Set the corresponding high-speed compensation parameter according to the frequency. $RTC_HCOMP.PARA = f/32768/32$ f is the compensation frequency, and it is recommended to configure it as an integer multiple of 32.768 kHz. 0.06ppm compensation accuracy: the valid range for f is 16 to 48MHz. 0.96ppm compensation accuracy: the valid range for f is 4 to 48MHz.	RW

Please refer to the following table for the specific high-speed compensation parameters:

Compensation Frequency f (MHz)	Compensation Parameter PARA
4	0b000100
6	0b000110
8	0b001000
12	0b001100
16	0b001111
20	0b010011
24	0b010111
32	0b011111
48	0b101110

13.5.13 Control Register 2 (RTC_CR2)

Offset	Bit Position																															
0x03C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000 (Only power-on reset and module reset are effective for this register reset)																															
Name	Reserved																				ALMBE	ALMAE	Reserved	WUTE	TSE	Reserved	TSEDG	WUCKSEL				
Access																					RW	RW		RW	RW		RW	RW				

Bit	Name	Functional Description	Access
31:11	Reserved	-	-
10	ALMBE	Alarm B enable 0b0: Disable 0b1: Enable	RW
9	ALMAE	Alarm A enable 0b0: Disable 0b1: Enable	RW
8	Reserved	-	-
7	WUTE	Wake-up timer enable 0b0: Disable 0b1: Enable	RW
6	TSE	Timestamp enable 0b0: Disable 0b1: Enable	RW
5:4	Reserved	-	-
3	TSEDG	Timestamp edge selection 0b0: Rising edge 0b1: Falling Edge	RW
2:0	WUCKSEL	Wake-up timer clock source selection 0b000: RTC/2 0b001: RTC/4 0b010: RTC/8 0b011: RTC/16 0b100: RTC_Sec/1 0b101: RTC_Sec/2 0b110: RTC_Sec/4 0b111: RTC_Sec/8	RW

13.5.14 Alarm A Register (RTC_ALMA)

Offset	Bit Position																															
0x040	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved	WEEK							MSKH	Reserved	HOURH			HOURL			MSKM	MINH			MINL			MSKS	SECH			SECL				
Access		RW							RW		RW			RW			RW	RW			RW			RW	RW			RW				

Bit	Name	Functional Description	Access
31	Reserved	-	-
30:24	WEEK	Alarm weekday bit Bit24-Bit30 correspond to Sunday-Saturday respectively: When the corresponding bit is set to 1, the alarm is active on that day of the week. For example: - Setting it to 0b0000001 indicates that the Sunday alarm is enabled - Setting it to 0b0000011 means the alarm is enabled for Sunday and Monday - Setting it to 0b1000000 indicates that the Saturday alarm is enabled	RW
23	MSKH	Alarm hour bit mask bit 0b0: Alarm A will be generated only when the hour bit match 0b1: The hour bit is invalid in alarm A	RW
22	Reserved	-	-
21:20	HOURH	Alarm hour tens digit BCD code of the alarm hour tens digit	RW
19:16	HOURL	Alarm hour units digit BCD code of the alarm hour units digit	RW
15	MSKM	Alarm minute bit mask bit 0b0: Alarm A will be generated only when the minute bit match 0b1: The minute bit is invalid in alarm A	RW
14:12	MINH	Alarm minute tens digit BCD code of the alarm minute tens digit	RW
11:8	MINL	Alarm minute units digit BCD code of the alarm minute units digit	RW
7	MSKS	Alarm second bit mask bit 0b0: Alarm A will be generated only when the second bit match 0b1: The second bit is invalid in alarm A	RW
6:4	SECH	Alarm seconds tens digit BCD code of the alarm second tens digit	RW

Bit	Name	Functional Description	Access
3:0	SECL	Alarm seconds units digit BCD code of the alarm second units digit	RW

**Note**

When performing bit-field operations, note that when configuring the 12-hour format, the HOUURL register must not be written with 0.

13.5.15 Alarm B Register (RTC_ALMB)

Offset	Bit Position																															
0x044	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	Reserved	WEEK							MSKH	Reserved	HOURH			HOURL			MSKM	MINH			MINL			MSKS	SECH			SECL				
Access		RW							RW		RW			RW			RW	RW			RW			RW	RW			RW				

Bit	Name	Functional Description	Access
31	Reserved	-	-
30:24	WEEK	Alarm weekday bit Bit24-Bit30 correspond to Sunday-Saturday, respectively: When the corresponding bit is set to 1, the alarm is active on that day of the week. For example: - Setting it to 0b0000001 indicates that the Sunday alarm is enabled - Setting it to 0b0000011 means the alarm is enabled for Sunday and Monday - Setting it to 0b1000000 indicates that the Saturday alarm is enabled	RW
23	MSKH	Alarm hour mask bit 0b0: Alarm B will be generated only when the hour bit matches 0b1: The hour bit is invalid in alarm B	RW
22	Reserved	-	-
21:20	HOURH	Alarm hour tens digit BCD code of the alarm hour tens digit	RW
19:16	HOURL	Alarm hour units digit BCD code of the alarm hour units digit	RW
15	MSKM	Alarm minute bit shielding bit 0b0: Alarm B will be generated only when the minute bit match 0b1: The minute bit is invalid in alarm B	RW
14:12	MINH	Alarm minute tens digit BCD code of the alarm minute tens digit	RW
11:8	MINL	Alarm minute units digit BCD code of the alarm minute units digit	RW
7	MSKS	Alarm seconds digit mask bit 0b0: Alarm B will be generated only when the second bit match 0b1: The second bit is invalid in alarm B	RW
6:4	SECH	Alarm seconds tens digit BCD code of the alarm second tens digit	RW

Bit	Name	Functional Description	Access
3:0	SECL	Alarm seconds units digit BCD code of the alarm second units digit	RW

**Note**

When performing bit-field operations, note that when configuring the 12-hour format, the HOUURL register must not be written with a value of 0.

13.5.16 Timestamp Time Register (RTC_TSTR)

Offset	Bit Position																																
0x048	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000 (Only power-on reset and module reset are effective for this register reset)																																
Name	Reserved										HOURH	HOURL				Reserved	MINH				MINL				Reserved	SECH				SECL			
Access											RO	RO					RO				RO					RO				RO			

Bit	Name	Functional Description	Access
31:22	Reserved	-	-
21:20	HOURH	Hour count tens digit BCD code of hour tens digit count value  Note This bit uses 24-hour format and is not affected by RTC_CR0.AMPM.	RO
19:16	HOURL	Hour count units digit BCD code of hour units digit count value  Note This bit uses 24-hour format and is not affected by RTC_CR0.AMPM.	RO
15	Reserved	-	-
14:12	MINH	Minute count tens digit BCD code of minute tens digit count value	RO
11:8	MINL	Minute count units digit BCD code of minute units digit count value	RO
7	Reserved	-	-
6:4	SECH	Second count tens digit BCD code of second tens digit count value	RO
3:0	SECL	Second count units digit BCD code of second units digit count value	RO

13.5.17 Timestamp Date Register (RTC_TSDR)

Offset	Bit Position																															
0x04C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000 (Only power-on reset and module reset are effective for this register)																															
Name	Reserved																WEEK	MON		Reserved	DAYH	DAYL										
Access																	RO	RO			RO	RO										

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:13	WEEK	Week count value Week count value, the format is the same as RTC_WEEK	RO
12:8	MON	Month count value Month count value, the format is the same as RTC_MON	RO
7:6	Reserved	-	-
5:4	DAYH	Day count tens digit BCD code of day tens digit count value	RO
3:0	DAYL	Day count units digit BCD code of day units digit count value	RO

13.5.18 Wake-Up Timer Register (RTC_WUTR)

Offset	Bit Position																															
0x050	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x0000FFFF (Only power-on reset and module reset are effective for this register reset)																															
Name	Reserved																ARR															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	ARR	Wake-up timer period value	RW

13.5.19 Interrupt Enable Register (RTC_IER)

Offset	Bit Position																															
0x054	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000 (Only power-on reset and module reset are effective for this register reset)																															
Name	Reserved																								PRD	Reserved	TS	WU	ALMB	ALMA		
Access																									RW		RW	RW	RW	RW		

Bit	Name	Functional Description	Access
31:7	Reserved	-	-
6	PRD	Periodic interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
5:4	Reserved	-	-
3	TS	Timestamp interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
2	WU	Timing interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
1	ALMB	Alarm B interrupt enable ● 0b0: Disable ● 0b1: Enable	RW

Bit	Name	Functional Description	Access
0	ALMA	Alarm A interrupt enable ● 0b0: Disable ● 0b1: Enable	RW

13.5.20 Interrupt Flag Register (RTC_IFR)

Offset	Bit Position																																						
0x058	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
Reset	0x00000000 (Only power-on reset and module reset are effective for this register reset)																																						
Name	Reserved																										PRD		Reserved	TSOV		TS		WU		ALMB		ALMA	
Access																											RO			RO		RO		RO		RO		RO	

Bit	Name	Functional Description	Access
31:7	Reserved	-	-
6	PRD	Periodic interrupt flag 0b0: No interrupt occurred 0b1: Interrupt is occurred	RO
5	Reserved	-	-
4	TSOV	Timestamp overflow flag 0b0: No overflow occurred 0b1: Overflow is occurred	RO
3	TS	Timestamp interrupt flag 0b0: No interrupt occurred 0b1: Interrupt is occurred	RO
2	WU	Timer interrupt flag 0b0: No interrupt occurred 0b1: Interrupt is occurred	RO
1	ALMB	Alarm B interrupt flag 0b0: No interrupt occurred 0b1: Interrupt is occurred	RO
0	ALMA	Alarm A interrupt flag 0b0: No interrupt occurred 0b1: Interrupt is occurred	RO

13.5.21 Interrupt Clear Register (RTC_ICR)

Offset	Bit Position																																						
0x05C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
Reset	0x000003FF																																						
Name	Reserved																										PRD		Reserved	TSOV		TS		WU		ALMB		ALMA	
Access																											WO			WO		WO		WO		WO		WO	

Bit	Name	Functional Description	Access
31:7	Reserved	-	-
6	PRD	Periodic interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
5	Reserved	-	-
4	TSOV	Timestamp overflow flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
3	TS	Timestamp interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
2	WU	Timing interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
1	ALMB	Alarm B interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
0	ALMA	Alarm A interrupt flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

13.5.22 Protection Register (RTC_WPR)

Offset	Bit Position																															
0x060	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								WPR							
Access																									WO							

Bit	Name	Functional Description	Access
31:8	Reserved	-	-
7:0	WPR	Protection sequence To disable RTC register write protection, write 0xCA followed by 0x53 to WPR; to enable RTC register write protection, write 0xCA followed by 0xFF (not equal to 0x53) to WPR. Except for WPR and ICR, all RTC registers are write-protected by default, and cannot be written upon power-on.	WO

14 IWDT

14.1 Introduction

This product features an embedded independent watchdog peripheral for high security and flexibility of use. This independent watchdog detects and resolves failures caused by software errors, triggering an interrupt or reset when its count reaches a programmed timeout value. The watchdog is driven by the IWDTCLK (clock source is RC10K), and it can still work normally even if the main clock fails.

14.2 Features

- Independent RC oscillator
- Trigger interrupt or reset on overflow
- A running watchdog can be stopped by a specific write sequence
- During DeepSleep mode, it can be configured to automatically pause counting

14.3 Functional Description

14.3.1 Function Block Diagram

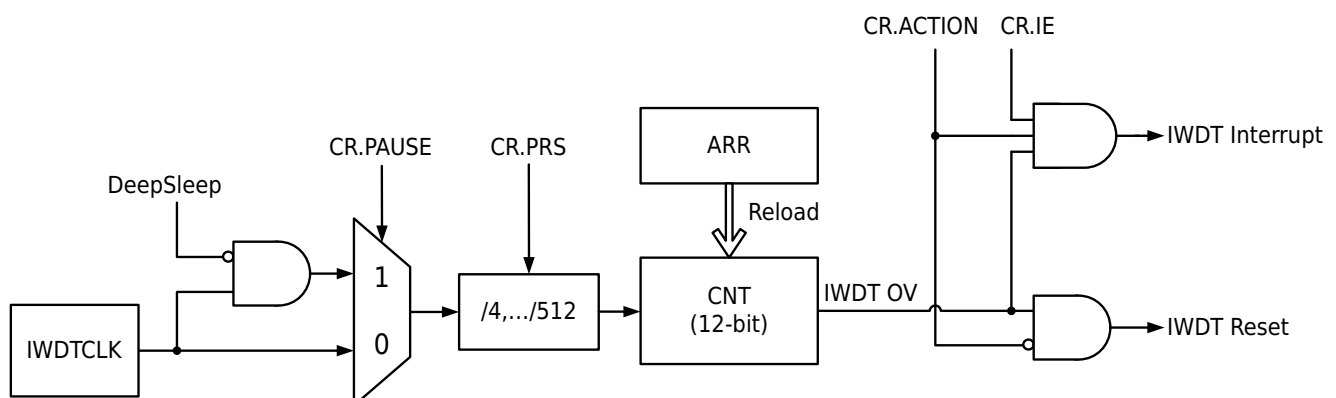


Figure 14-1 IWDT Function Block Diagram

After starting the watchdog, the counter starts counting down from 0xFFFF. When 0xAAAA is written to the IWDT_KR register, the value of IWDT_ARR is reloaded to the counter. When the count value is decremented to 0x000, the watchdog will generate an overflow signal; when the counter is reloaded and the count value is greater than the value of the IWDT_WINR register, an overflow signal will also be generated. Configuring the corresponding control bit of IWDT_CR enables the watchdog to generate reset or interrupt after overflow. Configuring the corresponding control bit of IWDT_CR, the watchdog can automatically suspend counting during DeepSleep.

14.3.2 Window Watchdog Mode

When the value of the IWDT_WINR register is configured to be less than IWDT_ARR, IWDT works in the window watchdog mode. In this mode, there are two conditions to trigger the watchdog overflow: the count value is decremented to 0x000, and the watchdog is reloaded when the count value is greater than the window value. When changing the IWDT_WINR register value, the hardware will automatically reload the watchdog and reset the down counter value to IWDT_ARR. The user program should reload

the watchdog before the count value is less than or equal to the value of the IWDT_WINR register and the count value is decremented to 0x000.

14.3.3 Independent Watchdog Mode

When the value of the IWDT_WINR register is configured to be greater than or equal to IWDT_ARR, IWDT works in the independent watchdog mode. In this mode, there is only one condition that triggers the watchdog overflow: the count value is decremented to 0x000. The user program should reload the watchdog before the count value is decremented to 0x000.

14.3.4 Watchdog Overflow Handling

The overflow signal of the watchdog can trigger an interrupt or a reset. When IWDT_CR.ACTION is set to 0, the watchdog overflow signal will reset the MCU; when IWDT_CR.ACTION is set to 1 and IWDT_CR.IE is 1, the watchdog overflow signal will trigger an interrupt, by writing 0 to the IWDT_SR register, the IWDT_SR.OV flag can be cleared to 0.

14.3.5 Watchdog Pause

If IWDT_CR.PAUSE is set to 1, the watchdog automatically pauses when the MCU enters DeepSleep; when the MCU exits DeepSleep, the watchdog continues to run. Using this feature, the MCU does not need to wake up frequently to feed the dog when it is in a low-power consumption state, and can achieve lower standby power consumption.

14.3.6 Watchdog Stop and Run Again

Write 0x1234, 0x5678 to the IWDT_KR register in sequence can stop the running watchdog.

Write 0xCCCC to the IWDT_KR register will enable the watchdog to run again with its configuration unchanged.

14.3.7 Register Write Protection

The IWDT_CR, IWDT_ARR, and IWDT_WINR registers are write-protected. If you need to modify these three registers, you should write 0x5555 to the IWDT_KR register to release the write protection of these registers after the watchdog starting; writing other values to the IWDT_KR register will enable the write protection of these registers.

14.4 Operation Example

14.4.1 Configure IWDT as Window Watchdog

- Step1.** Write 0xCCCC to the IWDT_KR register to enable the IWDT.
- Step2.** Write 0x5555 to the IWDT_KR register to release the write protection of the register.
- Step3.** Configure IWDT_CR, select the counting clock frequency of the watchdog, the action after overflow, and the DeepSleep feature.
- Step4.** Configure IWDT_ARR to select the overflow period of the watchdog.
- Step5.** Wait for IWDT_SR.ARRF to be 0.
- Step6.** Write the required value to IWDT_WINR to configure the window size.
- Step7.** Wait for IWDT_SR.WINRF to become 0, and automatically refresh the counter value to IWDT_ARR.
- Step8.** Before the count value is less than or equal to the IWDT_WINR register value and the count value is decremented to 0, write 0xAAAA to the IWDT_KR register to reload IWDT.

14.4.2 Configure IWDT as Independent Watchdog

- Step1.** Write 0xCCCC to the IWDT_KR register to enable the IWDT.
- Step2.** Write 0x5555 to the IWDT_KR register to release the write protection of the register.
- Step3.** Configure IWDT_CR, select the counting clock frequency of the watchdog, the action after overflow, and the DeepSleep feature.
- Step4.** Configure IWDT_ARR to select the overflow period of the watchdog.
- Step5.** Wait for IWDT_SR.ARRF to be 0
- Step6.** Refresh the counter value to the value of IWDT_ARR(IWDT_KR=0x0000AAAA).
- Step7.** Before the count value is decremented to 0, write 0xAAAA to the IWDT_KR register to reload the IWDT.

14.4.3 Watchdog Timeout Time (10kHz Independent Clock)

Table 14-1 IWDT Timeout Time (10kHz Independent Clock)

Prescaler Factor	PRS[2:0] Bits	Minimum Time (ms) ARR[11:0] = 0x000	Maximum Time (ms) ARR[11:0] = 0xFFF
4 frequency division	0b000	$0.1 * 4 = 0.4$	$0.1 * 4 * 4096 = 1638.4$
8 frequency division	0b001	$0.1 * 8 = 0.8$	$0.1 * 8 * 4096 = 3276.8$
16 frequency division	0b010	$0.1 * 16 = 1.6$	$0.1 * 16 * 4096 = 6553.6$
32 frequency division	0b011	$0.1 * 32 = 3.2$	$0.1 * 32 * 4096 = 13107.2$
64 frequency division	0b100	$0.1 * 64 = 6.4$	$0.1 * 64 * 4096 = 26214.4$
128 frequency division	0b101	$0.1 * 128 = 12.8$	$0.1 * 128 * 4096 = 52428.8$
256 frequency division	0b110	$0.1 * 256 = 25.6$	$0.1 * 256 * 4096 = 104857.6$
512 frequency division	0b111	$0.1 * 512 = 51.2$	$0.1 * 512 * 4096 = 209715.2$

14.5 Register Description

14.5.1 Register List

Base address: 0x40001000

Table 14-2 Independent Watchdog (IWDT) Register Offset

Offset	Register	Description	Width
0x80	IWDT_KR	Key Value Register	32
0x84	IWDT_CR	Control Register	32
0x88	IWDT_ARR	Reload Register	32
0x8C	IWDT_SR	Status Register	32
0x90	IWDT_WINR	Window Register	32

14.5.2 Key Value Register (IWDT_KR)

Offset	Bit Position																															
0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																KR															
Access																	WO															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	KR	IWDT key value ● Write 0xCCCC: Start IWDT counter ● Write 0xAAAA: Reload IWDT count value ● Write 0x1234, 0x5678 in sequence: Stop watchdog ● Write 0x5555: Release the write protection of IWDT_ARR, IWDT_WINR, IWDT_CR ● Write non-0x5555: Enable the write protection of IWDT_ARR, IWDT_WINR, IWDT_CR	WO

14.5.3 Control Register (IWDT_CR)


Note

- This register can only be modified after writing 0x5555 to the IWDT_KR register.
- This register can only be modified when IWDT_SR.CRF is 0.

Offset	Bit Position																															
0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																										PAUSE	ACTION	IE	Reserved	PRS	
Access																											RW	RW	RW		RW	

Bit	Name	Functional Description	Access
31:7	Reserved	-	-
6	PAUSE	WDT pause control in DeepSleep mode ● 0b0: IWDT continues to run in DeepSleep mode ● 0b1: IWDT automatically pauses in DeepSleep mode	RW
5	ACTION	Action configuration after IWDT overflow ● 0b0: Reset is generated after watchdog overflow ● 0b1: Interrupt is generated after watchdog overflow	RW
4	IE	IWDT interrupt enable control ● 0b0: Disable watchdog interrupt ● 0b1: Enable watchdog interrupt	RW
3	Reserved	-	-
2:0	PRS	Configure the watchdog count clock and the frequency division ratio of RC10K oscillator ● 0b000: 4-frequency division ● 0b001: 8-frequency division ● 0b010: 16-frequency division ● 0b011: 32-frequency division ● 0b100: 64-frequency division ● 0b101: 128-frequency division ● 0b110: 256-frequency division ● 0b111: 512-frequency division	RW

14.5.4 Reload Register (IWDT_ARR)


Note

- This register can only be modified after writing 0x5555 to the IWDT_KR register.
- This register can only be modified when IWDT_SR.ARRF is 0.

Offset	Bit Position																															
0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000FFF																															
Name	Reserved																ARR															
Access																																

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11:0	ARR	IWDT reload value	RW

14.5.5 Status Register (IWDT_SR)


Note

Write 0 to this register can clear the IWDT_SR.OV flag bit.

Offset	Bit Position																																			
0x8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																																			
Access																																	RO	RUN	RW	OV

Bit	Name	Functional Description	Access
31:5	Reserved	-	-
4	RUN	WDT running flag 0b0: WDT is not running 0b1: WDT is running	RO
3	OV	WDT overflow flag Write 0 to clear the overflow flag, write 1 has no effect. Read: 0b0: WDT overflow has not occurred 0b1: WDT overflow has occurred	RW
2	WINRF	WINR register update flag 0b0: WINR register update completed 0b1: WINR register is being updated Notice The WINR register can only be updated when the WINRF is 0.	RO
1	ARRF	ARR register update flag 0b0: ARR register update completed 0b1: ARR register is being updated Notice The ARR register can only be updated when the ARRF is 0.	RO
0	CRF	CR register update flag 0b0: CR register update completed 0b1: CR register is being updated Notice The CR register can only be updated when the CRF is 0.	RO

14.5.6 Window Register (IWDT_WINR)



Note

- This register can only be modified after writing 0x5555 to the IWDT_KR register.
- This register can only be modified when IWDT_SR.WINRF is 0.

Offset	Bit Position																															
0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000FFF																															
Name	Reserved												WINR																			
Access																			RW													

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11:0	WINR	Window comparator comparison value	RW

15 LPUART

15.1 Introduction

The Low Power Universal Asynchronous Receiver/Transmitter (LPUART) is a UART that allows full-duplex communication in the low power mode of MCU. When the MCU is in DeepSleep mode, LPUART can also receive UART data frames normally and wake up the MCU through the reception completion interrupt.

15.2 Features

- Configuration clock PCLK
- Transmission clock SCLK (SCLK can select XTL, RCL and PCLK)
- Support synchronous half-duplex, asynchronous full-duplex and single-wire half-duplex transmission.
- Programmable serial communication function
 - ▶ 2 character lengths: 8 bits, 9 bits
 - ▶ 3 check methods: no check, odd check, even check
 - ▶ 3 stop lengths: 1 bit, 1.5 bits, 2 bits
- Supports transmitting and receiving data in low power mode
- 16-bit baud rate counter
- Support hardware flow control (RTS, CTS)
- Support multi-machine communication and automatic address recognition

15.3 Functional Description

15.3.1 Function Block Diagram

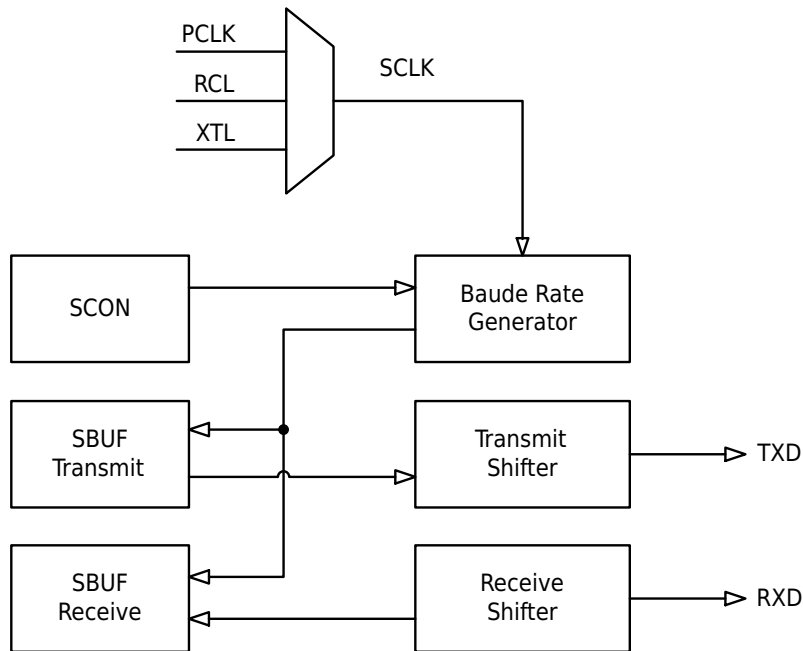


Figure 15-1 LPUART Function Block Diagram

15.3.2 Clock Description

The LPUART module has two clocks:

- Configuration Clock PCLK

The configuration clock is used for register configuration of the LPUART module by the system APB bus, and the configuration clock is fixed as the APB bus clock PCLK. When the system enters DeepSleep mode, the PCLK clock will stop.

- Transmission Clock SCLK

The transmission clock is used for LPUART data transceiver logic work, and the clock source can be selected from the external low-speed crystal oscillator clock (XTL), the internal low-speed clock (RCL), and the PCLK clock.

When the system enters DeepSleep mode, if SCLK selects XTL or RCL, the LPUART can still perform normal data transmission and reception without being affected by the system's DeepSleep mode.

15.3.3 Working Mode

LPUART supports multiple working modes: synchronous half-duplex mode, asynchronous full-duplex mode, and single-wire half-duplex mode.

By combining LPUART_SCON.SM with LPUART_SCON.HDSEL, various required working modes can be configured.

15.3.3.1 Mode0-Mode3 Function Comparison

Configure LPUART_SCON.SM to select different transmission modes: Mode0-Mode3. The main function comparison of these four working modes is shown in the table below:

Table 15-1 LPUART Mode0/1/2/3 Data Structure

Working Mode		Transmission Bit Width	Data Composition	Baud Rate
Mode0	Synchronous half-duplex mode	8-bit	Data (8bits)	$BaudRate = \frac{f_{SCLK}}{12}$
Mode1	Asynchronous full-duplex mode	10-11-bit	Start (1bit) + Data (8bits) + Stop (1-2bits)	$BaudRate = \frac{f_{SCLK}}{OVER \times SCNT}$
Mode2	Asynchronous full-duplex mode	11-12-bit	Start (1bit) + Data (8bits) + B8 (1bit) + Stop (1-2bits)	$BaudRate = \frac{f_{SCLK}}{OVER}$
Mode3	Asynchronous full-duplex mode	11-12-bit	Start (1bit) + Data (8bits) + B8 (1bit) + Stop (1-2bits)	$BaudRate = \frac{f_{SCLK}}{OVER \times SCNT}$

- When SCLK is selected as XTL (with the frequency of 32.768 kHz), Mode 1/2/3 can support baud rates of 9600/4800. The generation of the baud rate is detailed in the registers LPUART_BSEL/ LPUART_MODU.
- Mode0 can only be used as a master to transmit LPUART synchronous shift clock, and cannot be used as a slave to receive LPUART synchronous shift clock input from the outside.
- f_{SCLK} is the clock frequency of SCLK.
- OVER value:
 - ▶ When LPUART_SCON.OVER=0b00, the value of OVER in Mode1/3 mode is 16; and the value of OVER in Mode2 mode is 32.
 - ▶ When LPUART_SCON.OVER=0b01, the value of OVER in Mode1/3 mode is 8; and the value of OVER in Mode2 mode is 16.
 - ▶ When LPUART_SCON.OVER=0b10, the value of OVER in Mode1/3 mode is 4; and the value of OVER in Mode2 mode is 8.
- The definition of SCNT is detailed in register LPUART_SCNT introduction.
- B8 data bit is special and has different meanings in different applications, please refer to the following table:

Table 15-2 B8 Data Meaning

Application Scenario	LPUART_SCON.ADRDET	LPUART_SCON.B8CONT[1:0]	B8 Data Meaning
Parity check	-	0b01/0b10	When receiving, B8 is the parity check bit of the 8bits of data received. When transmitting, B8 is the parity check bit of the 8bits of data transmitted.
Multi-machine communication	0b1	-	B8=1, it means that it is currently an address frame. B8=0, it means that it is currently a data frame.

Application Scenario	LPUART_SCON. ADRDET	LPUART_SCON. B8CONT[1:0]	B8 Data Meaning
Other	0b0	0b00/0b11	Bit8 receiving/transmitting data.

When the multi-machine communication mode is turned on, the parity check of the received data is automatically closed; the parity check of the transmitted data is still controlled by B8CONT.

15.3.3.2 Mode0 Data Transmitting and Reception

● Mode0 Data Transmitting

When LPUART_SCON.REN set to 0, the LPUART operates in the transmission mode. The data to be transmitted is written into the LPUART_SBUF register, and then the data will be output from the RXD (lower bits first and higher bits later), while the synchronous shift clock is output from the TXD.

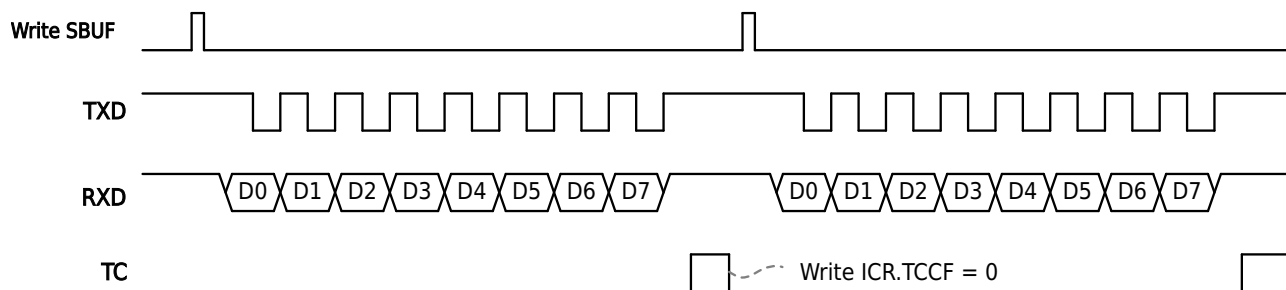


Figure 15-2 Mode0 Data Transmitting

● Mode0 Data Receiving

When LPUART_SCON.REN is set to 1, the LPUART is in the receiving state. When the LPUART_ISR.RC flag is 0, the TXD pin will output 8 synchronous shift clocks, and data will be input from RXD (lower bits first and higher bits later), each time the LPUART_ISR.RC flag is cleared, the next byte can be received.

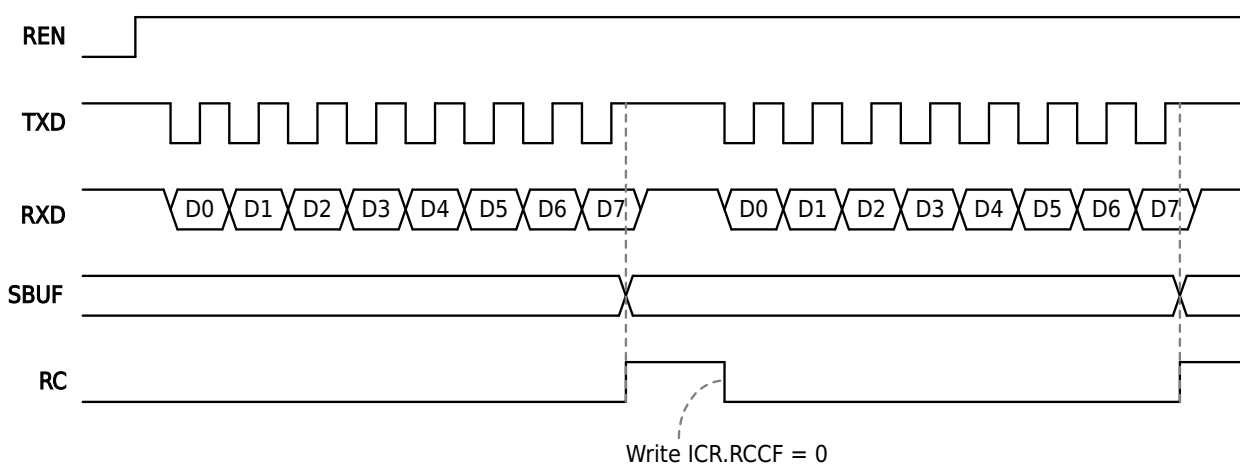


Figure 15-3 Mode0 Data Receiving

15.3.3.3 Mode1 Data Transmitting and Reception

● Mode1 Data Transmitting

Whether LPUART_SCON.REN is 0 or 1, data can be transmitted normally.

Write data to LPUART_SBUF, the hardware automatically loads the data into the transmit shift register and shifts the output from the TXD pin (lower bits first and higher bits later), and the LPUART_ISR.TXE flag is set to 1 by the hardware.

When the user queries that the LPUART_ISR.TXE flag is 1, the next data to be transmitted can be written to the LPUART_SBUF register.

Whenever the TXD pin completes transmitting a frame of data, the LPUART_ISR.TC flag will be set to 1 by hardware to indicate the current transmitting progress.

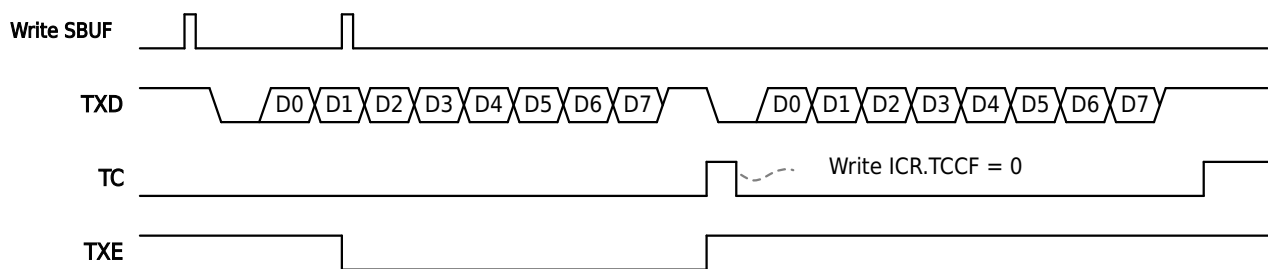


Figure 15-4 Mode1 Data Transmitting

- Mode1 Data Receiving

When receiving data, set LPUART_SCON.REN to 1, and clear the LPUART_ISR.RC bit to 0.

When a frame of data reception is completed, LPUART_ISR.RC flag will be set to 1 by hardware, and the received data can be read from LPUART_SBUF register.

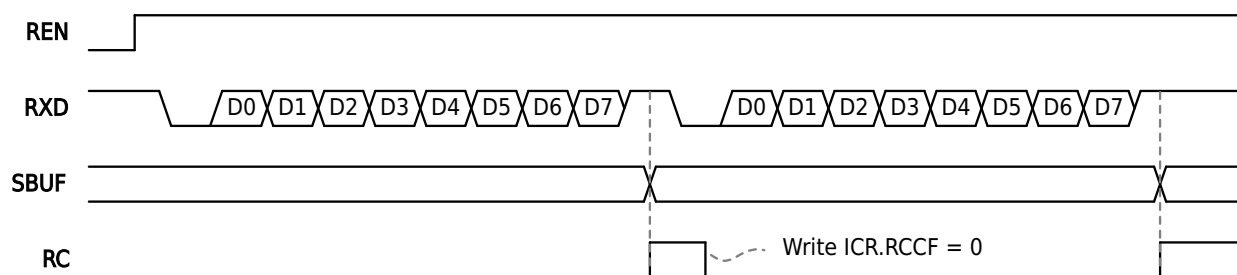


Figure 15-5 Mode1 Data Receiving

15.3.3.4 Mode2/Mode3 Data Transmitting and Reception

- Mode2/3 Data Transmitting

Whether LPUART_SCON.REN is 0 or 1, data can be transmitted normally.

Write data to LPUART_SBUF, the hardware automatically loads the data into the transmit shift register and shifts the output from the TXD pin (lower bits first and higher bits later), and the LPUART_ISR.TXE flag is set to 1 by the hardware.

When the user queries that the LPUART_ISR.TXE flag is 1, the next data to be transmitted can be written to the LPUART_SBUF register.

Whenever the TXD pin completes transmitting a frame of data, the LPUART_ISR.TC flag will be set to 1 by hardware to indicate the current transmitting progress.

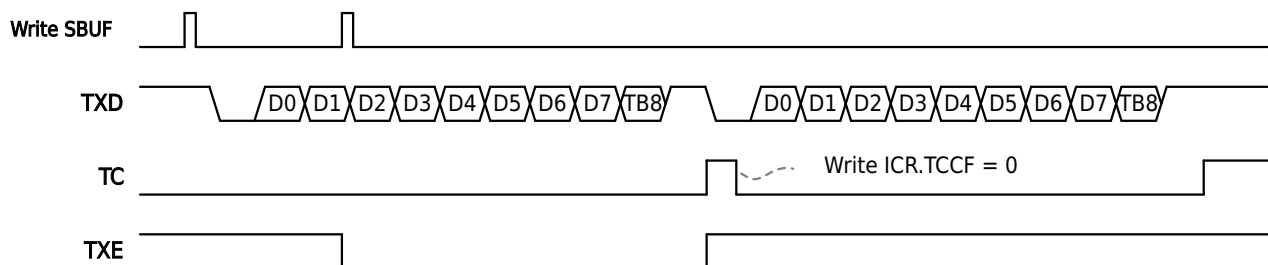


Figure 15-6 Mode2/Mode3 Data Transmitting

- Mode2/3 Data Receiving

When receiving data, set LPUART_SCON.REN to 1, and clear the LPUART_ISR.RC bit to 0.

When a frame of data reception is completed, LPUART_ISR.RC flag will be set to 1 by hardware, and the received data can be read from LPUART_SBUF register.

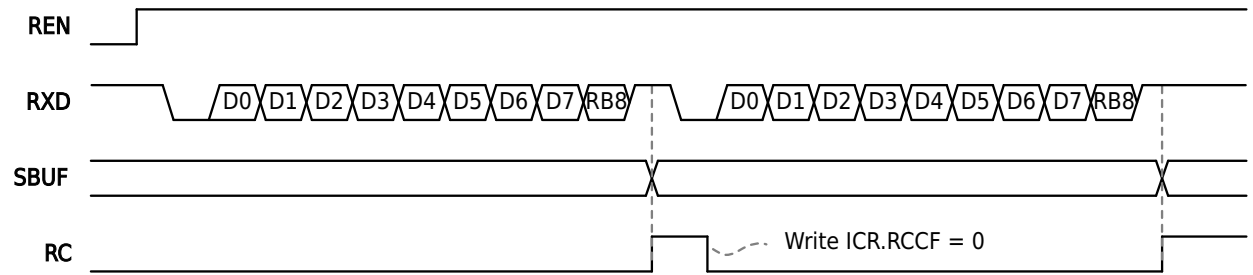


Figure 15-7 Mode2/Mode3 Data Receiving

15.3.3.5 Single-Wire Half-Duplex Data Transmitting and Reception

When LPUART_SCON.HDSEL=0b1, it can enter the single-wire half-duplex transmission mode:

- The TX and RX signal lines are connected inside the module.
- The RX signal is no longer used, and the received and transmitted data are all completed through the TX signal. TX signal is done by hardware logic without software control.
- After configuring LPUART_SCON.HDSEL to 0b1, must wait for the transmission duration of one data bit (i.e., after 1/BaudRate seconds) before normal reception can occur.
- When the transmit buffer is empty, the TX signal is always input (receive state). Once a data is filled into the transmit buffer, the TX signal becomes output (transmit state). When the transmission is complete, the transmit buffer becomes empty and the TX signal returns to input (receive state).
- When there is no data transmission, the TXD pin is always in the input state. The user program needs to configure this pin as an open-drain output and connect an external pull-up resistor.



Notice

1. Even if the module is in the process of receiving, as long as the transmit buffer is filled with data, the TX signal will immediately become output (transmit state). The receiving process will be interrupted and a wrong received data will be obtained.
2. Similarities and differences between single-wire half-duplex mode and Mode0 mode:
 - The same: both use a signal to time-division multiplex to complete data transmission and reception.
 - Different: Mode0 is synchronous transmission, with synchronous clock. The single-wire half-duplex mode is asynchronous transmission, and all baud rate calculations still follow the baud rate formula of Mode1-Mode3, only the transceiver function is multiplexed on the TX signal.

Single-wire half-duplex transmission supports baud rates of 9600/4800. However, it is important to note that when receiving data in single-wire half-duplex mode at baud rates of 9600/4800, the following steps must be followed:

- Step1.** Refer to the relevant descriptions of pin digital multiplexing functions in the "General-Purpose I/O (GPIO)" chapter to map TXD to the required pin; configure TXD as an open-drain output and enable the pull-up resistor.
- Step2.** Configure SYCTRL module register PERI_CLKEN0.LPUARTx to 1 to enable LPUART configuration clock and working clock.
- Step3.** Configure LPUART_SCON.SCLKSEL to select the transmission clock source for LPUART.
- Step4.** Set LPUART_SCON.HDSEL to 0b1 to enable single-wire mode.
- Step5.** Add a delay of at least 500 μ s and initialize TXD to the receiving state.
- Step6.** Configure the LPUART_BSEL and LPUART_MODU registers to enable 9600/4800 baud rate modulation.
- Step7.** For subsequent steps, refer to the ["Receive Data Example"](#) chapter.

For other scenarios, such as transmitting data in single-wire half-duplex mode at baud rates of 9600/4800, or modulating the baud rate by configuring the LPUART_SCNT register, simply follow the steps outlined in the "Transmit Data Example" or "Receive Data Example" chapters for normal configuration.

15.3.4 Baud Rate Generation

The formula for generating the baud rate in Mode0-Mode3 is as follows:

- Mode0 baud rate generation formula: $\text{BaudRate} = \frac{f_{SCLK}}{12}$
- Mode1 baud rate generation formula: $\text{BaudRate} = \frac{f_{SCLK}}{\text{OVER} \times \text{SCNT}}$
- Mode2 baud rate generation formula: $\text{BaudRate} = \frac{f_{SCLK}}{\text{OVER}}$
- Mode3 baud rate generation formula: $\text{BaudRate} = \frac{f_{SCLK}}{\text{OVER} \times \text{SCNT}}$



Note

- f_{SCLK} is the clock frequency of SCLK.
- OVER value:
 - ▶ When LPUART_SCON.OVER = 0b00, the value of OVER in Mode1/3 mode is 16; and the value of OVER in Mode2 mode is 32.
 - ▶ When LPUART_SCON.OVER = 0b01, the value of OVER in Mode1/3 mode is 8; and the value of OVER in Mode2 mode is 16.
 - ▶ When LPUART_SCON.OVER = 0b10, the value of OVER in Mode1/3 mode is 4; and the value of OVER in Mode2 mode is 8.
- The definition of SCNT is detailed in register LPUART_SCNT introduction.

Mode1/Mode3 Baud Rate Setting Example

Table 15-3 SCLK = 4MHz Baud Rate Calculation Table

Baud Rate	SCLK=4MHz								
	OVER=4			OVER=8			OVER=16		
	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%
2400	417	2398.08	-0.08%	208	2403.85	0.16%	104	2403.85	0.16%
4800	208	4807.69	0.16%	104	4807.69	0.16%	52	4807.69	0.16%
9600	104	9615.38	0.16%	52	9615.38	0.16%	26	9615.38	0.16%
19200	52	19230.77	0.16%	26	19230.77	0.16%	13	19230.77	0.16%
38400	26	38461.54	0.16%	13	38461.54	0.16%	7	35714.29	-6.99%
57600	17	58823.53	2.12%	9	55555.56	-3.55%	4	62500.00	8.51%
76800	13	76923.08	0.16%	7	71428.57	-6.99%	3	83333.33	8.51%
115200	9	111111.11	-3.55%	4	125000.00	8.51%	2	125000.00	8.51%
128000	8	125000.00	-2.34%	4	125000.00	-2.34%	2	125000.00	-2.34%
256000	4	250000.00	-2.34%	2	250000.00	-2.34%	1	250000.00	-2.34%
1000000	1	1000000.00	0.00%	1	/	/	0	/	/

Table 15-4 SCLK = 8MHz Baud Rate Calculation Table

Baud Rate	SCLK=8MHz								
	OVER=4			OVER=8			OVER=16		
	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%
2400	833	2400.96	0.04%	417	2398.08	-0.08%	208	2403.85	0.16%
4800	417	4796.16	-0.08%	208	4807.69	0.16%	104	4807.69	0.16%
9600	208	9615.38	0.16%	104	9615.38	0.16%	52	9615.38	0.16%
19200	104	19230.77	0.16%	52	19230.77	0.16%	26	19230.77	0.16%
38400	52	38461.54	0.16%	26	38461.54	0.16%	13	38461.54	0.16%
57600	35	57142.86	-0.79%	17	58823.53	2.12%	9	55555.56	-3.55%
76800	26	76923.08	0.16%	13	76923.08	0.16%	7	71428.57	-6.99%
115200	17	117647.06	2.12%	9	111111.11	-3.55%	4	125000.00	8.51%
128000	16	125000.00	-2.34%	8	125000.00	-2.34%	4	125000.00	-2.34%
256000	8	250000.00	-2.34%	4	250000.00	-2.34%	2	250000.00	-2.34%
1000000	2	1000000.00	0.00%	1	1000000.00	0.00%	1	/	/
2000000	1	2000000.00	0.00%	1	/	/	0	/	/

Table 15-5 SCLK = 16MHz Baud Rate Calculation Table

Baud Rate	SCLK=16MHz								
	OVER=4			OVER=8			OVER=16		
	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%
2400	1667	2399.52	-0.02%	833	2400.96	0.04%	417	2398.08	-0.08%
4800	833	4801.92	0.04%	417	4796.16	-0.08%	208	4807.69	0.16%
9600	417	9592.33	-0.08%	208	9615.38	0.16%	104	9615.38	0.16%
19200	208	19230.77	0.16%	104	19230.77	0.16%	52	19230.77	0.16%
38400	104	38461.54	0.16%	52	38461.54	0.16%	26	38461.54	0.16%
57600	69	57971.01	0.64%	35	57142.86	-0.79%	17	58823.53	2.12%
76800	52	76923.08	0.16%	26	76923.08	0.16%	13	76923.08	0.16%
115200	35	114285.71	-0.79%	17	117647.06	2.12%	9	111111.11	-3.55%
128000	31	129032.26	0.81%	16	125000.00	-2.34%	8	125000.00	-2.34%
256000	16	250000.00	-2.34%	8	250000.00	-2.34%	4	250000.00	-2.34%
1000000	4	1000000.00	0.00%	2	1000000.00	0.00%	1	1000000.00	0.00%
2000000	2	2000000.00	0.00%	1	2000000.00	0.00%	1	/	/
4000000	1	4000000.00	0.00%	1	/	/	0	/	/

Table 15-6 SCLK = 24MHz Baud Rate Calculation Table

Baud Rate	SCLK=24MHz								
	OVER=4			OVER=8			OVER=16		
	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%
2400	2500	2400.00	0.00%	1250	2400.00	0.00%	625	2400.00	0.00%
4800	1250	4800.00	0.00%	625	4800.00	0.00%	313	4792.33	-0.16%
9600	625	9600.00	0.00%	313	9584.66	-0.16%	156	9615.38	0.16%
19200	313	19169.33	-0.16%	156	19230.77	0.16%	78	19230.77	0.16%
38400	156	38461.54	0.16%	78	38461.54	0.16%	39	38461.54	0.16%
57600	104	57692.31	0.16%	52	57692.31	0.16%	26	57692.31	0.16%

Baud Rate	SCLK=24MHz								
	OVER=4			OVER=8			OVER=16		
	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%
76800	78	76923.08	0.16%	39	76923.08	0.16%	20	75000.00	-2.34%
115200	52	115384.62	0.16%	26	115384.62	0.16%	13	115384.62	0.16%
128000	47	127659.57	-0.27%	23	130434.78	1.90%	12	125000.00	-2.34%
256000	23	260869.57	1.90%	12	250000.00	-2.34%	6	250000.00	-2.34%
1000000	6	1000000.00	0.00%	3	1000000.00	0.00%	2	/	/
2000000	3	2000000.00	0.00%	2	/	/	1	/	/
6000000	1	6000000.00	0.00%	1	/	/	0	/	/

Table 15-7 SCLK = 32MHz Baud Rate Calculation Table

Baud Rate	SCLK=32MHz								
	OVER=4			OVER=8			OVER=16		
	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%
2400	3333	2400.24	0.01%	1667	2399.52	-0.02%	833	2400.96	0.04%
4800	1667	4799.04	-0.02%	833	4801.92	0.04%	417	4796.16	-0.08%
9600	833	9603.84	0.04%	417	9592.33	-0.08%	208	9615.38	0.16%
19200	417	19184.65	-0.08%	208	19230.77	0.16%	104	19230.77	0.16%
38400	208	38461.54	0.16%	104	38461.54	0.16%	52	38461.54	0.16%
57600	139	57553.96	-0.08%	69	57971.01	0.64%	35	57142.86	-0.79%
76800	104	76923.08	0.16%	52	76923.08	0.16%	26	76923.08	0.16%
115200	69	115942.03	0.64%	35	114285.71	-0.79%	17	117647.06	2.12%
128000	63	126984.13	-0.79%	31	129032.26	0.81%	16	125000.00	-2.34%
256000	31	258064.52	0.81%	16	250000.00	-2.34%	8	250000.00	-2.34%
1000000	8	1000000.00	0.00%	4	1000000.00	0.00%	2	1000000.00	0.00%
2000000	4	2000000.00	0.00%	2	2000000.00	0.00%	1	2000000.00	0.00%
4000000	2	4000000.00	0.00%	1	4000000.00	0.00%	1	/	/
8000000	1	8000000.00	0.00%	1	/	/	0	/	/

Table 15-8 SCLK = 48MHz Baud Rate Calculation Table

Baud Rate	SCLK=48MHz								
	OVER=4			OVER=8			OVER=16		
	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%
2400	5000	2400.00	0.00%	2500	2400.00	0.00%	1250	2400.00	0.00%
4800	2500	4800.00	0.00%	1250	4800.00	0.00%	625	4800.00	0.00%
9600	1250	9600.00	0.00%	625	9600.00	0.00%	313	9584.66	-0.16%
19200	625	19200.00	0.00%	313	19169.33	-0.16%	156	19230.77	0.16%
38400	313	38338.66	-0.16%	156	38461.54	0.16%	78	38461.54	0.16%
57600	208	57692.31	0.16%	104	57692.31	0.16%	52	57692.31	0.16%
76800	156	76923.08	0.16%	78	76923.08	0.16%	39	76923.08	0.16%
115200	104	115384.62	0.16%	52	115384.62	0.16%	26	115384.62	0.16%
128000	94	127659.57	-0.27%	47	127659.57	-0.27%	23	130434.78	1.90%
256000	47	255319.15	-0.27%	23	260869.57	1.90%	12	250000.00	-2.34%
1000000	12	1000000.00	0.00%	6	1000000.00	0.00%	3	1000000.00	0.00%

Baud Rate	SCLK=48MHz								
	OVER=4			OVER=8			OVER=16		
	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%	SCNT	Actual Baud Rate	Error%
2000000	6	2000000.00	0.00%	3	2000000.00	0.00%	2	1500000.00	-25.00%
4000000	4	3000000.00	0.00%	2	3000000.00	0.00%	1	3000000.00	0.00%
6000000	3	4000000.00	0.00%	2	/	/	1	/	/
12000000	1	12000000.00	0.00%	1	/	/	0	/	/

15.3.5 Frame Error Detection

When LPUART works in Mode1/2/3, it has frame error detection function:

- When receiving data, if the stop bit is not recognized within the expected time, it is a frame error.
- When a frame error is detected, LPUART_ISR.FE will be set to 1 by hardware, and the user program should clear this flag in time.

15.3.6 Multi-Machine Communication

When working in Mode2/3, set the LPUART_SCON.ADRDET to 1 to enable the multi-machine communication address automatic identification function. The master can transmit address frames and data frames, and the slave can receive data frames only after receiving matching address frames.

Master Transmitting Address and Data Frame

When LPUART_SBUF.DATA8 is 1, the address frame is transmitted, LPUART_SBUF.DATA is the address of the target slave.

When LPUART_SBUF.DATA8 is 0, the data frame is transmitted, LPUART_SBUF.DATA is the data to be transmitted.

Slave Receiving Address and Data Frame

When LPUART_SCON.ADRDET is 1, the slave only receives address frames that match its own address, ignoring data frames and unmatched address frames. When the slave receives the address frame from RXD, the hardware automatically compares the address in the address frame with the local address. Only when the two match, the received address is stored in the LPUART_SBUF register and LPUART_ISR.RC is set.

When LPUART_SCON.ADRDET is 0, the slave only receives data frames and ignores address frames. The user program should set LPUART_SCON.ADRDET to 0 after the address matches to receive subsequent data frames transmitted by the master.

Slave Address Configuration

The slave address consists of the broadcast address and the user configuration address:

- Broadcast Address: LPUART_SADDR | LPUART_SADEN
- User Configuration Address: LPUART_SADDR & LPUART_SADEN

When an address frame is received, if the address received from the bus, after being ANDed with the value of LPUART_SADEN, matches the user-configured address, or if the received address, after being ANDed with the broadcast address, matches the broadcast address, it is considered that an address frame matching the slave address has been received.

Example: LPUART_SADDR=0x55, LPUART_SADEN=0xF0, then the slave can match addresses 0xF5, 0xF7, 0xFD, 0xFF (broadcast address) and 0x50~0x5F (user-configured address).

15.3.7 Hardware Flow Control

The hardware flow control function can be realized by adding CTS and RTS signals, the LPUART module will automatically control the transmitting and receiving of data based on the high and low levels of CTS and RTS without the need for software judgment.

The hardware flow control diagram between two LPUART modules is as follows:

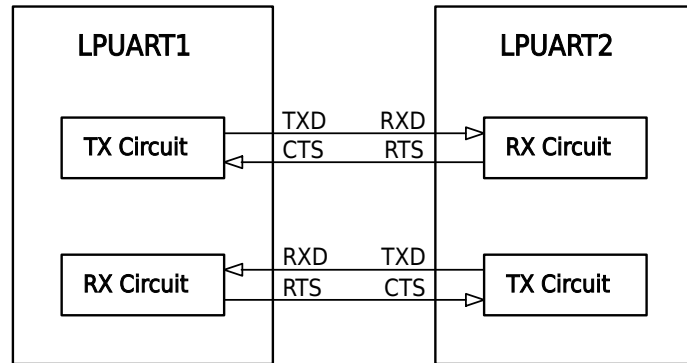


Figure 15-8 LPUART Hardware Flow Control

RTS Flow Control

When RTS flow control is enabled (LPUART_SCON.RTSEN is 1), the RTS pin indicates the receiving status of this module:

- When the receive buffer is empty, the RTS pin turns low level to notify the transmitting end to send the next frame of data.
- When the receive buffer is full, the RTS pin turns to high level, notifying the transmitting end to pause sending the next frame of data.

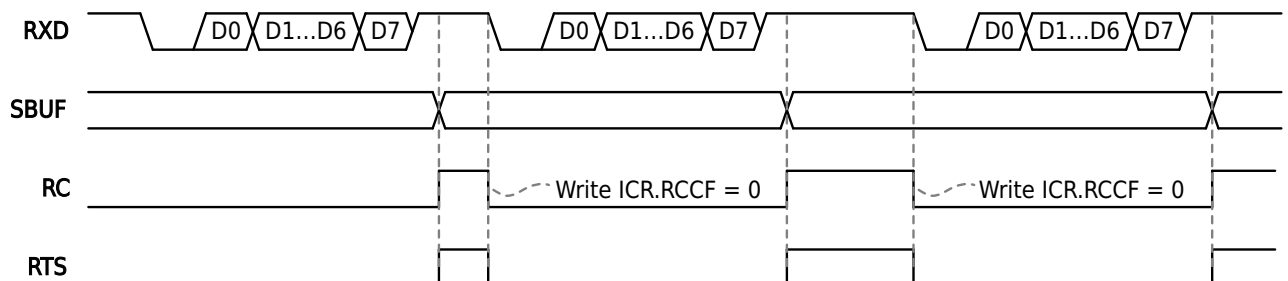


Figure 15-9 RTS Hardware Flow Control Signal

CTS Flow Control

When CTS flow control is enabled (LPUART_SCON.CTSEN is set to 1), the CTS pin instructs this module to transmit the next frame of data:

- If CTS is low level, LPUART transmits the next frame of data.
- If CTS is high level, after LPUART finishes transmitting the current frame, it waits for CTS to become low level before sending the next frame.

When the CTS level changes, LPUART_ISR.CTSIF is automatically set to 1. Read LPUART_ISR.CTS to obtain the current CTS pins level value.

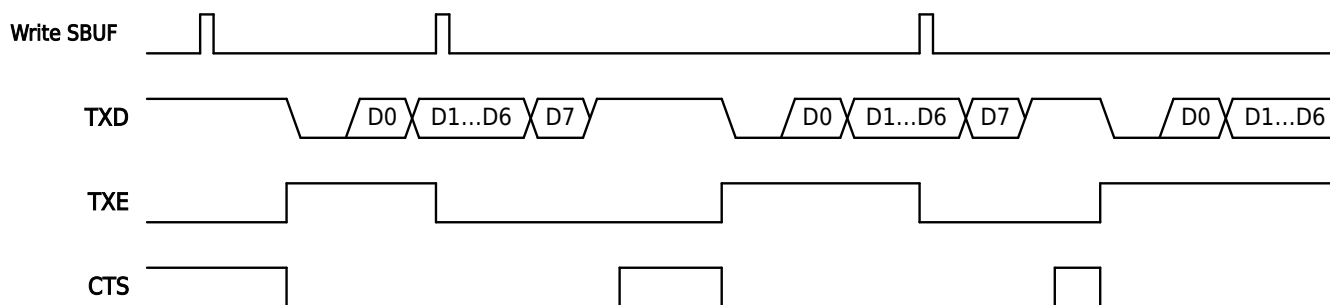


Figure 15-10 CTS Hardware Flow Control Signal

15.3.8 Receive/Transmit Buffer

Receive Buffer

The receive end of the LPUART module has a frame (8bits/9bits) receive buffer, which retains the received data frame until the Stop bit of the next data frame is received before updating the current data frame.

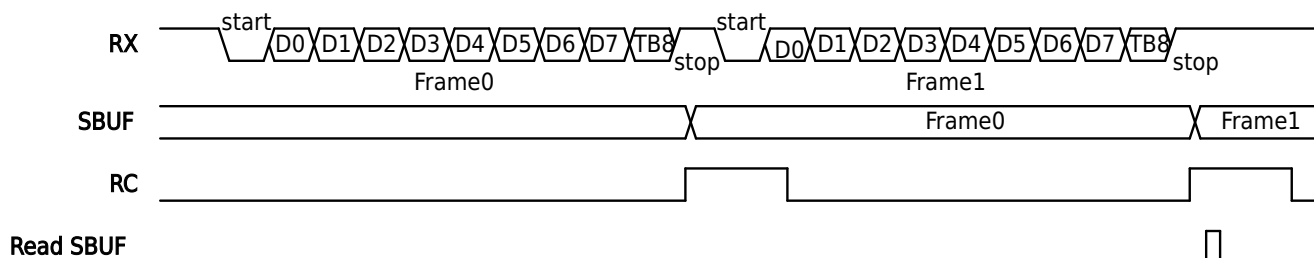


Figure 15-11 Receive Buffer

Transmit Buffer

The transmit end of LPUART module has a frame (8bits/9bits) transmit buffer. When the LPUART is transmitting the current frame, the software for the next transmitting data will be written into LPUART_SBUF.

- When LPUART_ISR.TXE = 0, it indicates that the current transmit buffer is full, LPUART_SBUF cannot write the next transmit data.
- When LPUART_ISR.TXE = 1, it indicates that the current transmit buffer is empty, LPUART_SBUF can write the next transmit data. After the current data transmission is completed, the hardware automatically loads the data in the transmit buffer into the shift register and sends it out.

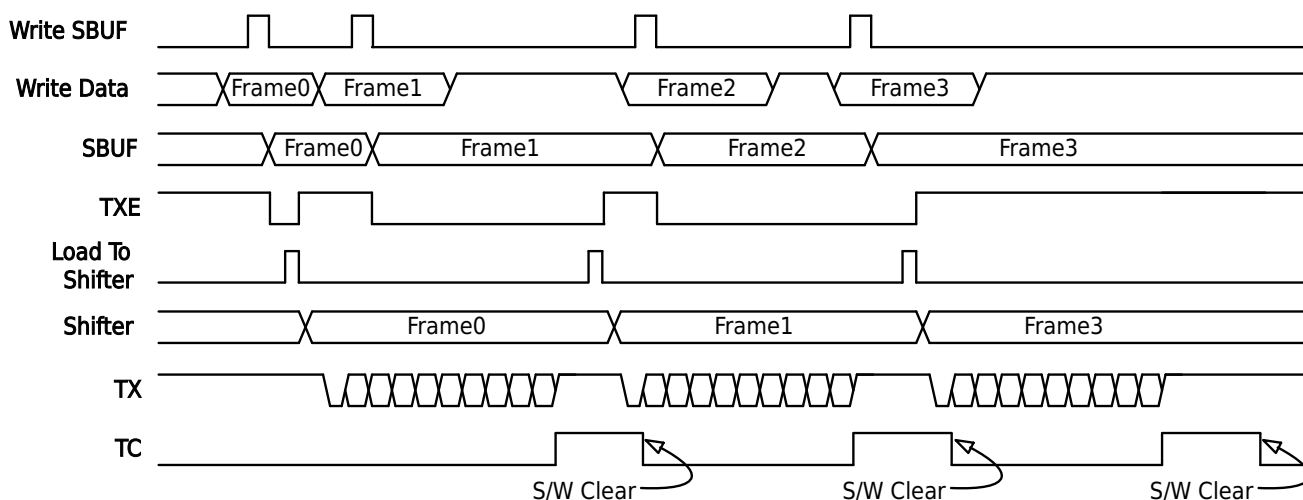


Figure 15-12 Transmit Buffer

15.4 Operation Example

15.4.1 Transmit Data Example

Procedure

- Step1.** Refer to the relevant description of the port digital multiplexing function in the "General-Purpose I/O (GPIO)" chapter, map TXD to the required pin; and configure TXD to CMOS output.
- Step2.** Configure SYSCTRL module register PERI_CLKEN0.LPUARTx to 1 to enable LPUART configuration clock and working clock.
- Step3.** Configure LPUART_SCON.SCLKSEL to select the transmission clock source for LPUART.
- Step4.** Configure LPUART_SCON.SM to select the working mode of LPUART.
- Step5.** Configure LPUART_SCON.OVER and LPUART_SCNT register to configure the communication baud rate.
- Step6.** Configure LPUART_SCON.STOPBIT, LPUART_SCON.B8CONT, to configure the STOP length and parity mode of LPUART.
- Step7.** Write 0x00 to the ICR register to clear all status flags.
- Step8.** When the value of ISR.TXE is queried for 1, write a frame of data to be transmitted to the LPUART_SBUF register.
- Step9.** The query waits for ISR.TC to become 1 and the data frame written this time is sent.
- Step10.** Write 0x00 to ICR.TCCF to clear the transmitting completion flag.
- Step11.** If the data to be transmitted is not completed, then jump to step8.

15.4.2 Receive Data Example

Procedure

- Step1.** Refer to the relevant description of the port digital multiplexing function in the "General-Purpose I/O (GPIO)" chapter. Map the RXD to the required pin; and configure RXD as an input and enable the pull-up resistor.
- Step2.** Configure SYSCTRL module register PERI_CLKEN0.LPUARTx to 1 to enable LPUART configuration clock and working clock.
- Step3.** Configure LPUART_SCON.SCLKSEL to select the transmission clock source for LPUART.
- Step4.** Configure LPUART_SCON.SM to select the working mode of LPUART.
- Step5.** Configure LPUART_SCON.OVER and LPUART_SCNT register to configure the communication baud rate.
- Step6.** Configure LPUART_SCON.STOPBIT, LPUART_SCON.B8CONT, to configure the STOP length and parity mode of LPUART.
- Step7.** Write 0x00 to the ICR register to clear all status flags.
- Step8.** Set LPUART_SCON.REN to 1 to enable RXD data reception.
- Step9.** The query waits for ISR.RC to become 1, and one frame of data is received.
- Step10.** Query the ISR.PE and ISR.FE to confirm whether the received data frame is valid. If it is invalid, perform error handling, if it is valid, read the data from the LPUART_SBUF register and save it .
- Step11.** Write 0x00 to ICR.RCCF to clear the reception completion flag.
- Step12.** If the data to be received is not completed, jump to step9.

15.5 Register Description

15.5.1 Register List

- LPUART0 base address: 0x40000000
- LPUART1 base address: 0x40008000

Table 15-9 Low Power Universal Asynchronous Receiver/Transmitter (LPUART) Register Offset

Offset	Register	Description	Width
0x00	LPUART_SBUF	Data Register	32
0x04	LPUART_SCON	Control Register	32
0x08	LPUART_SADDR	Address Register	32
0x0C	LPUART_SADEN	Address Mask Register	32
0x10	LPUART_ISR	Interrupt Flag Bit Register	32
0x14	LPUART_ICR	Interrupt Flag Bit Clear Register	32
0x18	LPUART_SCNT	Baud Rate Register	32
0x1C	LPUART_BSEL	Baud Rate Generate Register	32
0x20	LPUART_MODU	9600/4800 Baud Rate Modulator Register	32

15.5.2 Data Register (LPUART_SBUF)

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																							DATA8		DATA						
Access																								RW		RW						

Bit	Name	Functional Description	Access
31:9	Reserved	-	-
8	DATA8	Bit8 Data Bit ● In Mode 0/1, read this bit is 0, write this bit is invalid ● Under Mode 2/3, there are such situations: ▶ When the hardware parity bit is turned on, this bit is the parity bit of the received data when receiving, and the verification is performed by the hardware. If the verification error occurs, the verification error flag bit PE is set to 1; this bit is invalid when transmitting, the parity bit of the transmitted data is calculated and transmitted by the hardware. ▶ When the hardware parity bit is turned off, this bit is received data Bit8 when receiving; this bit is sent data Bit8 when transmitting.  Note When the multi-machine communication mode is turned on, the parity check of the received data is automatically closed; the parity check of the transmitted data is still controlled by B8CONT.	RW
7:0	DATA	Data When transmitting data, write the data to be transmitted into this register; when receiving data, read the received data from this register.	RW

15.5.3 Control Register (LPUART_SCON)

Offset	Bit Position																																	
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved									HDSEL	FEIE	CTSIE	CTSEN	RTSEN	Reserved	STOPBIT	PEIE	SCLKSEL	OVER	TXEIE	SM	ADRDET	REN	B8CONT	TCIE	RCIE								
Access										RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Functional Description	Access
31:23	Reserved	-	-
22	HDSEL	Single-wire half-duplex function enable 0b0: Disable 0b1: Enable	RW
21	FEIE	Frame error interrupt enable 0b0: Disable 0b1: Enable	RW
20	CTSIE	CTS signal flip interrupt enable 0b0: Disable 0b1: Enable	RW
19	CTSEN	CTS signal enable control 0b0: Disable 0b1: Enable	RW
18	RTSEN	RTS signal enable control 0b0: Disable 0b1: Enable	RW
17:16	Reserved	-	-
15:14	STOPBIT	STOPBIT duration configuration 0b00: 1-bit 0b01: 1.5-bit 0b10: 2-bit 0b11: Setting prohibited  Notice In Mode 0, this bit must be set to 0b00.	RW
13	PEIE	Parity error interrupt enable 0b0: Disable 0b1: Enable	RW

Bit	Name	Functional Description	Access
12:11	SCLKSEL	Transmission clock SCLK source selection ● 0b00: PCLK ● 0b01: PCLK ● 0b10: XTL ● 0b11: RCL	RW
10:9	OVER	Oversampling control ● Mode0: Invalid ● Mode1/3: ▶ 0b00: 16 sampling frequency division ▶ 0b01: 8 sampling frequency division ▶ 0b10: 4 sampling frequency division ▶ 0b11: Setting prohibited ● Mode2: ▶ 0b00: 32 sampling frequency division ▶ 0b01: 16 sampling frequency division ▶ 0b10: 8 sampling frequency division ▶ 0b11: Setting prohibited	RW
8	TXEIE	TX Buf empty interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
7:6	SM	Working mode ● 0b00: Mode0, synchronous half-duplex, Data (8bits) ● 0b01: Mode1, asynchronous full-duplex, Start (1bit) + Data (8bits) + Stop (1-2bits) ● 0b10: Mode2, asynchronous full-duplex, Start (1bit) + Data (8bits) + B8 (1bit) + Stop (1-2bits) ● 0b11: Mode3, asynchronous full-duplex, Start (1bit) + Data (8bits) + B8 (1bit) + Stop (1-2bits)	RW
5	ADDRDET	Multi-machine address automatic recognition enable ● 0b0: Disabled ● 0b1: Enabled	RW
4	REN	Receive enable ● Mode0: ▶ 0b0: Transmit ▶ 0b1: Receivie ● Mode1/2/3: ▶ 0b0: Transmit ▶ 0b1: Receive/Transmit	RW
3:2	B8CONT	DATA8 data bit function configuration ● 0b00: Deciding by software read/write LPUART_SBUF.DATA8 ● 0b01: Hardware even parity check ● 0b10: Hardware odd parity check ● 0b11: Setting prohibited	RW

Bit	Name	Functional Description	Access
1	TCIE	Transmit complete interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
0	RCIE	Receive complete interrupt enable ● 0b0: Disable ● 0b1: Enable	RW

15.5.4 Address Register (LPUART_SADDR)

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								ADDR							
Access																									RW							

Bit	Name	Functional Description	Access
31:8	Reserved	-	-
7:0	ADDR	Slave device address register	RW

15.5.5 Address Mask Register (LPUART_SADEN)

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								MASK							
Access																									RW							

Bit	Name	Functional Description	Access
31:8	Reserved	-	-
7:0	MASK	Slave device address mask register Each bit corresponds to an address bit: 0b0: Does not participate in address matching 0b1: Participates in address matching	RW

15.5.6 Interrupt Flag Bit Register (LPUART_ISR)

Offset	Bit Position																																					
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000008																																					
Name	Reserved																																					
Access																											RO	CTS	RO	CTSIF	RO	PE	RO	TXE	RO	FE	RO	TC

Bit	Name	Functional Description	Access
31:7	Reserved	-	-
6	CTS	CTS signal flag Can only be cleared through hardware 0b0: CTS signal is low level 0b1: CTS signal is high level	RO
5	CTSIF	CTS signal flip flag 0b0: CTS signal does not flip 0b1: CTS signal flipped	RO
4	PE	Parity check error flag 0b0: No parity errors occurred 0b1: A parity error occurred	RO
3	TXE	Tx Buffer empty flag Can only be cleared through hardware 0b0: Tx Buffer is not empty 0b1: Tx Buffer is empty	RO
2	FE	Frame error flag 0b0: No frame error occurred 0b1: A frame error occurred	RO
1	TC	Transmit completion flag 0b0: Transmission not completed 0b1: Transmission completed	RO
0	RC	Receive completion flag 0b0: Reception not completed 0b1: Reception completed	RO

15.5.7 Interrupt Flag Bit Clear Register (LPUART_ICR)

Offset	Bit Position																																					
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000037																																					
Name	Reserved																										CTSIFCF		PECF		Reserved		FECF		TCCF		RCCF	
Access																											WO		WO				WO		WO		WO	

Bit	Name	Functional Description	Access
31:6	Reserved	-	-
5	CTSIFCF	ISR.CTSIF flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
4	PECF	ISR.PE flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
3	Reserved	-	-
2	FECF	ISR.FE flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
1	TCCF	ISR.TC flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
0	RCCF	ISR.RC flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

15.5.8 Baud Rate Register (LPUART_SCNT)

Offset	Bit Position																															
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																SCNT															
Access																	RW															

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	SCNT	Baud rate counter	RW

15.5.9 Baud Rate Generate Register (LPUART_BSEL)

Offset	Bit Position																															
0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															BSEL
Access																																RW

Bit	Name	Functional Description	Access
31:2	Reserved	-	-
1:0	BSEL	Baud rate generation way ● 0b00-0b01: Generate baud rate using OVER and SCNT (the same as before) ● 0b10: When SCLK is selected as XTL (LPUART_SCON.SCLKSEL=0b10), generate a baud rate of 9600 using modulation method ● 0b11: When SCLK is selected as XTL (LPUART_SCON.SCLKSEL=0b10), generate a baud rate of 4800 using modulation method	RW

15.5.10 9600/4800 Baud Rate Modulator Register (LPUART_MODU)

Offset	Bit Position															
0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0x0000054A															
Name	Reserved												MODU			
Access													RW			

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11:0	MODU	9600/4800 baud rate modulator When SCLK is selected as XTL (LPUART_SCON.SCLKSEL=0b10) and BSEL.BSEL = 0b10/0b11, this register is valid: - When BSEL.BSEL = 0b10, the modulation settings required for a 9600 baud rate are generated, and the value to be set in this register should be 0x54A. - When BSEL.BSEL = 0b11, the modulation settings required for a 4800 baud rate are generated, and the value to be set in this register should be 0xEF8.	RW

Principle of Generating 9600 Baud Rate

When SCLK is selected as XTL (LPUART_SCON.SCLKSEL=0b10), and the SCLK frequency is 32768. The baud rate of LPUART is 9600 (BSEL.BSEL = 0b10). Each bit requires $32768/9600 = 3.413$ SCLKs. If each bit uses 3 SCLKs, the error of each bit is -0.413 SCLKs. If each bit uses 4 SCLKs, the error of each bit is +0.587 SCLKs.

Bit	Name	Number of SCLK	LPUART_MODU	Final Inaccuracy (Number of SCLK)
0	Start	3	0	-0.413
1	DATA[0]	4	1	+0.174(=-0.413+0.587)
2	DATA[1]	3	0	-0.239(=+0.174-0.413)
3	DATA[2]	4	1	+0.348(=-0.239+0.587)
4	DATA[3]	3	0	-0.065(=+0.348-0.413)
5	DATA[4]	3	0	-0.478(=-0.065-0.413)
6	DATA[5]	4	1	+0.109(=-0.478+0.587)
7	DATA[6]	3	0	-0.304(=+0.109-0.413)
8	DATA[7]	4	1	+0.283(=-0.304+0.587)
9	Parity/stop1	3	0	-0.130(=+0.283-0.413)
10	Stop1/stop2	4	1	+0.457(=-0.13+0.587)
11	Stop2	3	0	+0.044(=+0.457-0.413)

Principle of Generating 4800 Baud Rate

When SCLK is selected as XTL (LPUART_SCON.SCLKSEL=0b10), and the SCLK frequency is 32768. The baud rate of LPUART is 4800 (BSEL.BSEL = 0b11). Each bit requires $32768/4800 = 6.827$ SCLKs. If each bit uses 6 SCLKs, the error of each bit is -0.827 SCLKs. If each bit uses 7 SCLKs, the error of each bit is +0.173 SCLKs.

Bit	Name	Number of SCLK	LPUART_MODU	Final Inaccuracy (Number of SCLK)
0	Start	7	1	+0.173
1	DATA[0]	7	1	+0.346(=+0.173+0.173)
2	DATA[1]	6	0	-0.481(=+0.346-0.827)
3	DATA[2]	7	1	-0.308(=-0.481+0.173)
4	DATA[3]	7	1	-0.135(=-0.308+0.173)
5	DATA[4]	7	1	+0.038(=-0.135+0.173)
6	DATA[5]	7	1	+0.211(=+0.038+0.173)
7	DATA[6]	7	1	+0.384(=+0.211+0.173)
8	DATA[7]	6	0	-0.443(=+0.384-0.827)
9	Parity/stop1	7	1	-0.270(=-0.443+0.173)
10	Stop1/stop2	7	1	-0.097(=-0.270+0.173)
11	Stop2	7	1	+0.076(=-0.097+0.173)

16 SPI

16.1 Introduction

The serial peripheral interface (SPI) provides data transmission function conforming to the SPI communication protocol in the master-slave mode, which can be used for synchronous serial communication between the MCU and external devices. This module supports full-duplex and single-wire half-duplex and simplex communication.

16.2 Features

- Can be configured as master or slave, support multi-machine mode
- The maximum division factor in master mode is $PCLK/2$
- The maximum division factor in slave mode is $PCLK/4$
- Multiple communication modes: full-duplex, single-wire half-duplex, simplex
- 2 transmission orders: MSB first or LSB first
- Various data frame lengths: 4bits-16bits
- 2 NSS methods: hardware control, software control
- Configurable serial clock polarity and phase
- Support master mode delayed sampling

16.3 Functional Description

16.3.1 Function Block Diagram

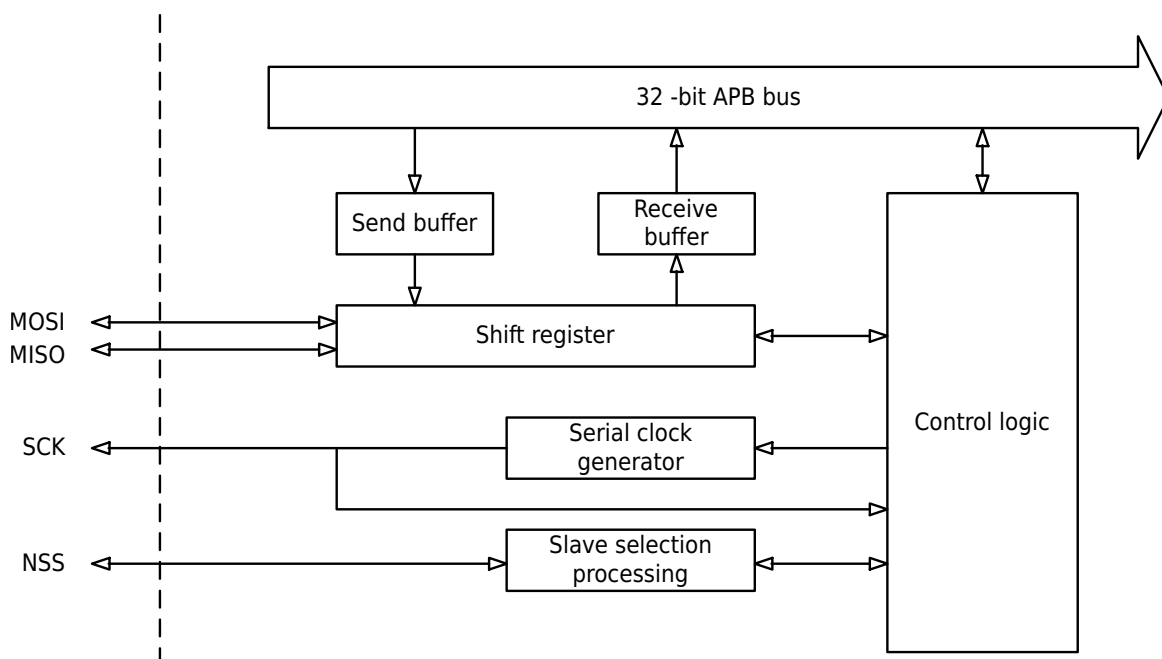


Figure 16-1 SPI Block Diagram

16.3.2 Communication Format and Timing

SPI serial frame format depends on serial clock polarity CPOL, serial clock phase CPHA and data frame format. The normal communication of the SPI interface requires that all masters and slaves must be configured to the same communication format.

The clock polarity CPOL refers to the level state of the serial clock in the idle state where the SPI does not transmit data. When SPI_CR0.CPOL is 0, the serial clock SCK is low when idle; when SPI_CR0.CPOL is 1, the serial clock SCK is high when idle.

The clock phase CPHA refers to the sequence in which the master and slave of the SPI sample and shift according to the serial clock SCK. When SPI_CR0.CPHA is 0, the master and slave sample data on the front edge and shift data on the back edge; When SPI_CR0.CPHA is 1, the master and slave shift data on the front edge, and sample data on the back edge.

By setting SPI_CR0.WIDTH, the width of each frame of data can be configured from 4bits to 16bits.

By setting SPI_CR0.LSBF, the transmission sequence of each frame of data can be configured as LSB first or MSB first.

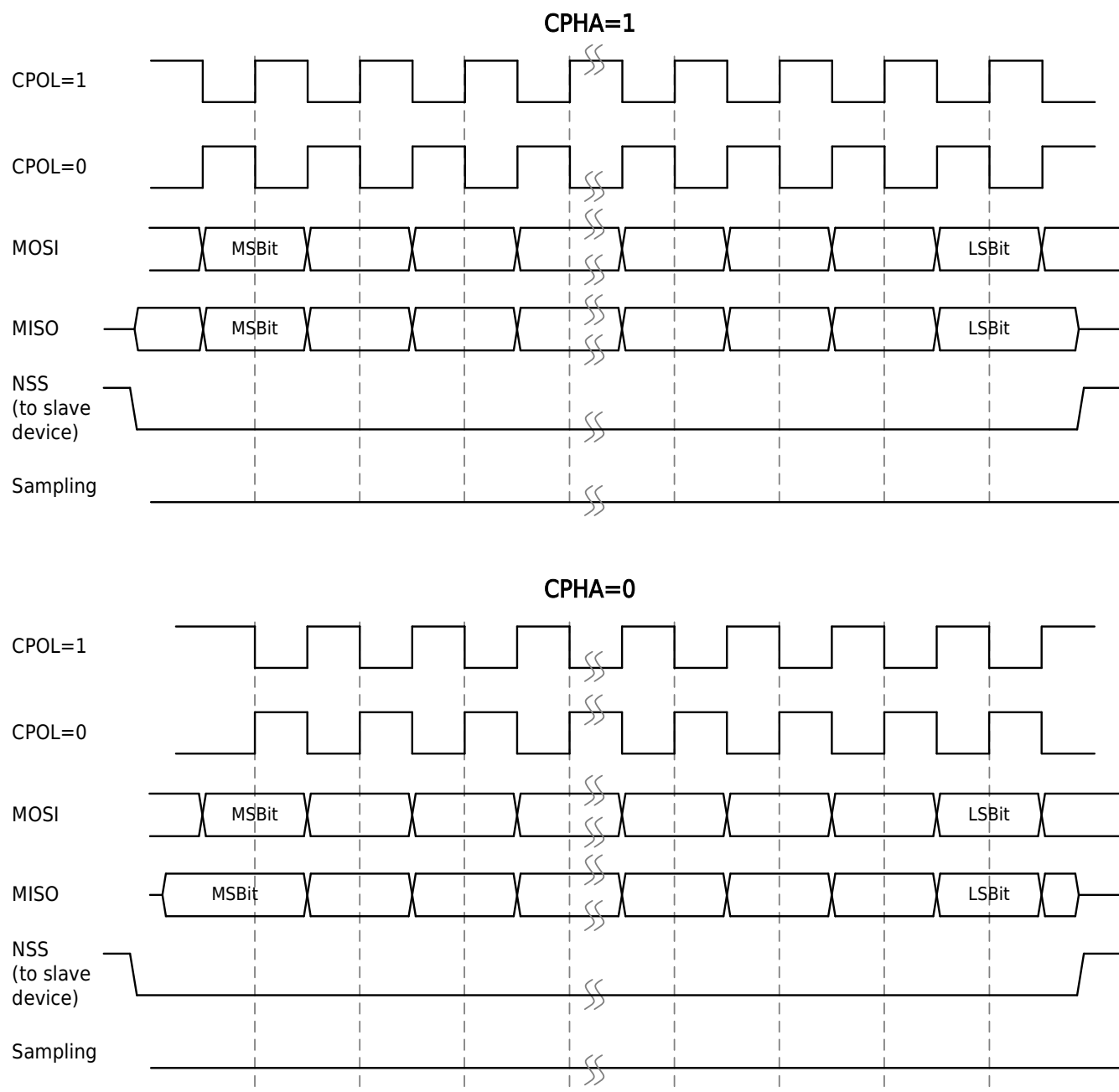


Figure 16-2 SPI Communication Format Timing Diagram

16.3.3 Slave Select Pin Configuration

This module provides a variety of slave selection methods to adapt to different application scenarios.

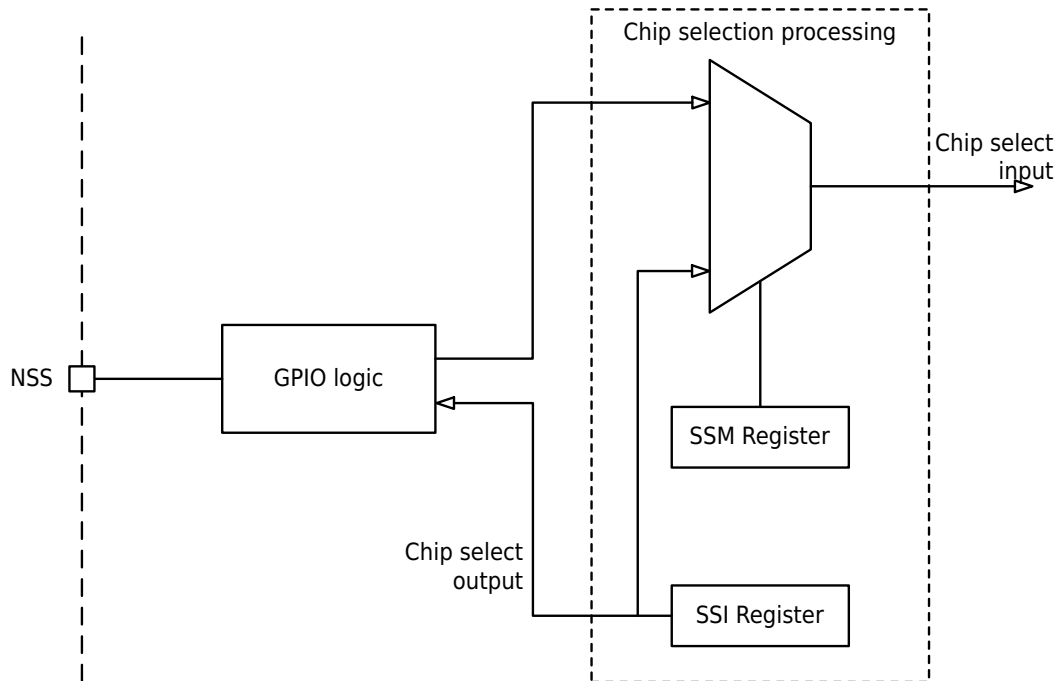


Figure 16-3 Slave Select NSS Pin Configuration

- When the SPI works in master mode and SPI_CR0.SSM is 0, the slave selects NSS as an input, and its state is determined by the level of the NSS pin.
- When the SPI works in master mode and SPI_CR0.SSM is 1, the slave selects NSS as an output, and the output level of the NSS pin is determined by the internal register SPI_SSI.SSI. When SPI_SSI.SSI is 0, the NSS pin outputs a low level; when SPI_SSI.SSI is 1, the NSS pin outputs a high level.
- When SPI works in slave mode and SPI_CR0.SSM is 0, the low level of NSS pin can select this module.
- When SPI operates in slave mode and SPI_CR0.SSM is set to 1, the NSS pin no longer functions as a chip select. The module can be selected when the SPI_SSI.SSI register is set to 0.

16.3.4 Full-duplex Communication

The full-duplex communication method allows the master and slave to perform two-way communication of sending and receiving at the same time relying on two unidirectional data lines. In this mode, the master provides a clock signal through the SCK pin to synchronize data input and output. The master and slave MOSI pins are connected to each other, and the MISO pins are connected to each other. The master sends data to the slave through the MOSI pin, and the slave returns the data through the MISO pin. Whenever the user writes data to the SPI_DR register, the SPI master will send data to the SPI slave through SCK and MOSI; and receive data from the slave through SCK and MISO.

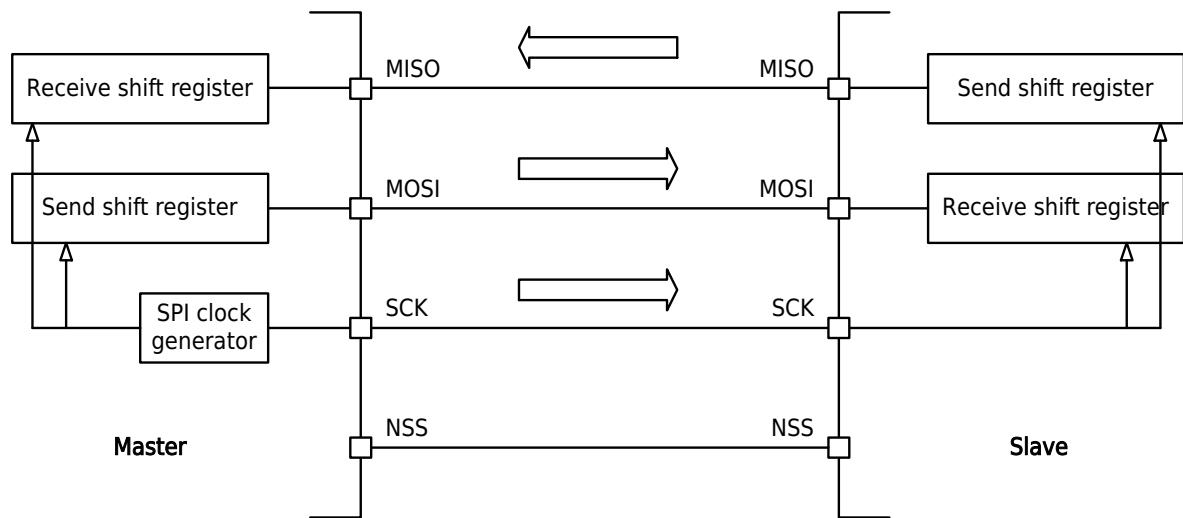


Figure 16-4 SPI Full-duplex Communication

16.3.5 Single-wire Half-duplex Communication

The single-wire half-duplex communication method allows the master and slave to perform time-sharing two-way communication based on a two-way data line. The pin used as a master is MOSI, and the pin used as a slave is MISO. The data transfer direction is determined by SPI_HDOE.HDOE. When SPI_HDOE.HDOE is set to 1, the data line transmits data. Unused pins do not participate in SPI communication and can be used by other functions. When the master operates in receive mode, arbitrary data must be written to the SPI_DR register to trigger the SPI master to send SCK for data transmission.

When the master-slave performs single-wire half-duplex communication, since the switching of sending and receiving may not be synchronized, they may have a driving conflict for the bidirectional shared data line. In order to avoid possible damage to the device due to such situations, it is recommended to insert a current-limiting resistor in series on the data line to protect chip pins.

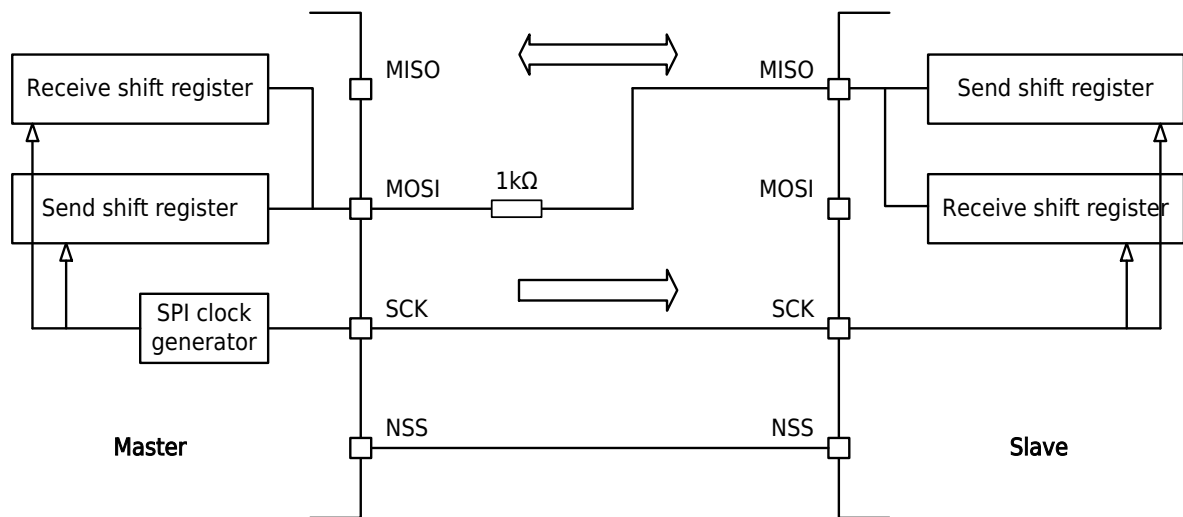


Figure 16-5 SPI Single-wire Half-duplex Communication

16.3.6 Simplex Communication

The simplex communication method allows the master and slave to rely on a one-way data line to perform one-way communication of fixed reception or fixed transmission. By setting SPI_CR0.CM, select the simplex only send or simplex only receive mode. Unused pins do not participate in SPI communication and can be used by other functions.

When the master works in the sending state and the slave works in the receiving state, the master writes data to the SPI_DR register to trigger the SPI master to send SCK for data transmission.

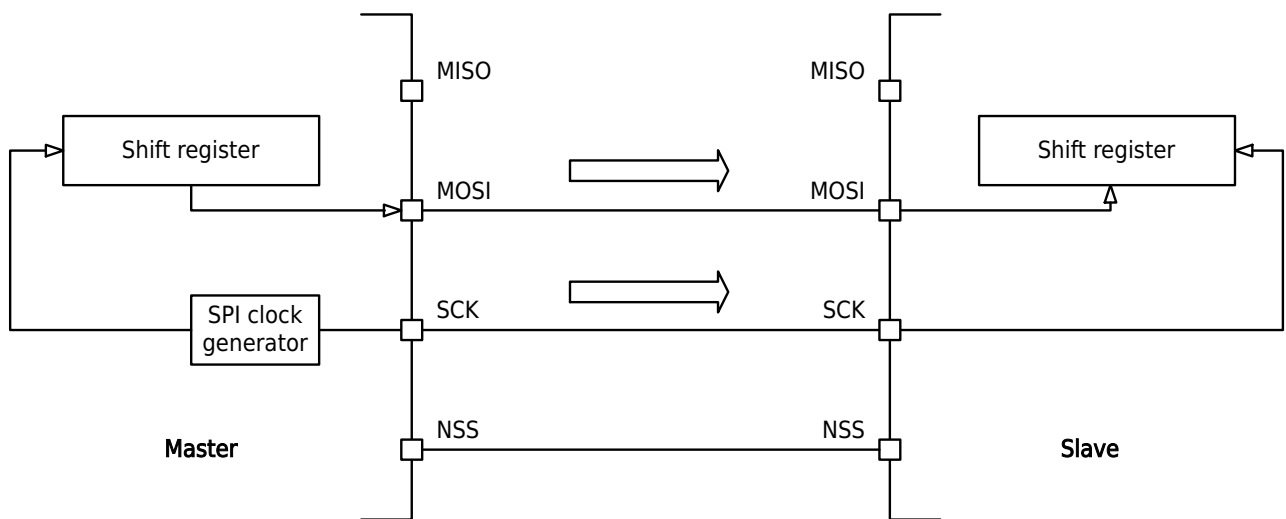


Figure 16-6 SPI Simplex Transmit Communication Diagram

When the master works in the receiving state and the slave works in the sending state, the master also needs to write arbitrary data to the SPI_DR register to trigger the SPI master to send SCK for data transmission.

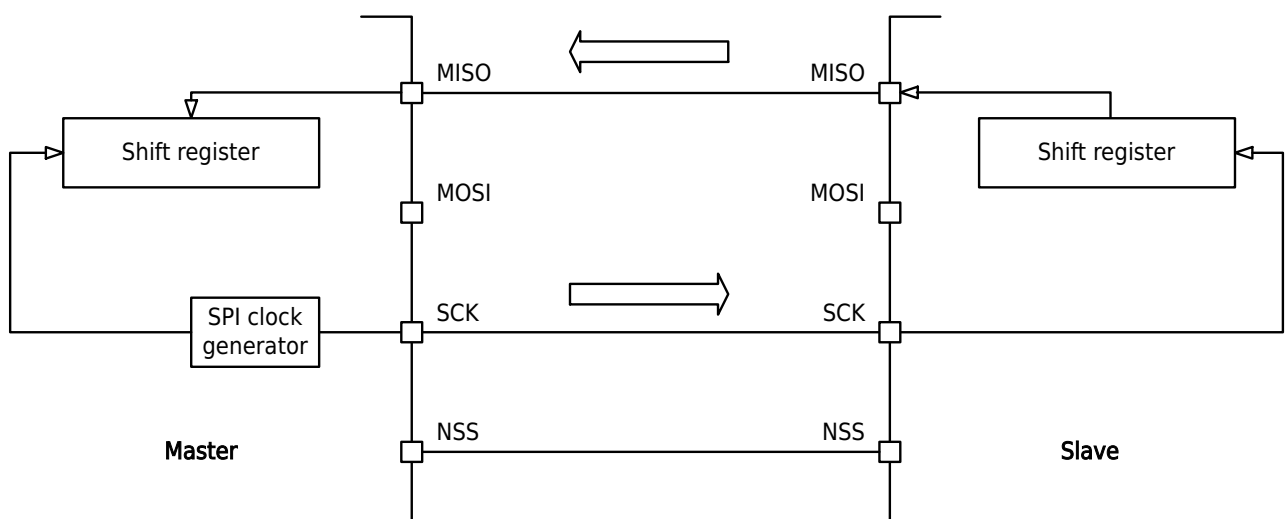


Figure 16-7 SPI Simplex Receive Communication Diagram



Note

- NSS pin can be used to provide hardware control flow between master and slave devices. Peripherals can also choose not to use these pins. After that, the hardware control flow must be handled internally for the master and slave.
- In this configuration, two idle pins (MOSI or MISO) can be used as GPIO.

16.3.7 Multi-machine Communication

In a multi-machine communication system, the slave selection NSS pin of the master is configured as a hardware input (SPI_CR0.SSM is 0), and the GPIO is responsible for outputting the slave selection signal, and the NSS input of the master is driven by the GPIO of another master. Once the slave selection NSS pin of the master is pulled low by the GPIO of other masters, a mode error MODF will be generated, which will cause the SPI of this module to be forced to return to the disabled state (that is, SPI_CR0.EN

is cleared), and the master-slave mode selection register (SPI_CR0.MSTR) will also be cleared (that is, switched to slave mode). When the system needs to use SPI communication again, it needs to reset and initialize the module again. Therefore, when the master tries to obtain the SPI bus control right again, it needs to detect whether the NSS input is high. If it is high and there is no MODF, the bus control right can be successfully obtained. In a system with multiple slaves, each slave shares the serial clock and data lines, but requires its own independent slave selection signal, which requires the master to output the corresponding slave selection signal to each slave through different GPIOs.

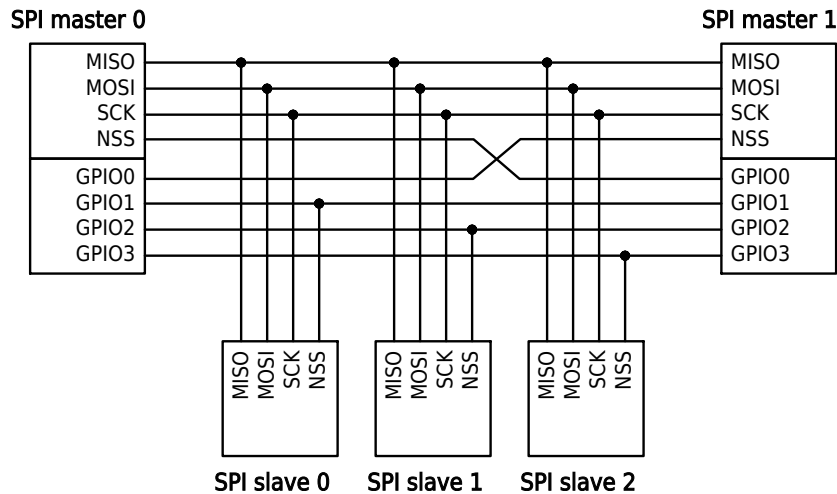


Figure 16-8 SPI Multi-machine System

16.3.8 Master Mode Delayed Sampling

In SPI communication, due to the following factors:

- The time required for the slave to send data from the shift edge of the serial clock SCK to MISO;
- Propagation time for data from slave data output MISO to master data input MISO.

When the master receives valid data, there is a certain delay compared to the shift edge of the serial clock. In the traditional SPI protocol, the master needs to sample the MISO data at the sampling edge of half an SCK cycle after the shift edge, which requires that the above-mentioned delay must be less than half an SCK cycle, and the difference between half an SCK cycle and this delay must meet the requirement of MISO setup time. Therefore, when the above-mentioned delay is obvious, the rate of SPI communication will be restricted, and data errors will occur when the rate is high.

This module provides the function of delayed sampling in the master mode, which can effectively alleviate the communication speed bottleneck problem caused by the above problems. By setting the register SPI_CR0.SMP to 1, enable the master mode to delay sampling. At this time, as the SPI master, this module will take the sampling edge of the traditional SPI standard as the reference, and delay half the SCK cycle for sampling. This not only provides a higher tolerance to delay for the entire system, but also does not affect the data transmission of the next bit, thereby achieving a higher communication rate.



Note

1. This function is only valid in master mode, not in slave mode.
2. SPI_CR0.SMP does not affect the configuration of the communication format of the master and slave — whether this function is enabled or not, the master and slave must be configured with the same CPOL and CPHA according to the requirements of conventional SPI.

16.3.9 Status Flags and Error Flags

SPI provides a variety of status flags to indicate the current working status, including general status flags and error flags.

Slave Select Input (NSS) State

The slave selection input status indicates the level of the currently input slave selection signal, which corresponds to the register SPI_SR.SSLVL. When SPI_SR.SSLVL is 0, the slave selection input is low; when SPI_SR.SSLVL is 1, the slave selection input is high.

Bus Busy Flag Bit

The bus busy flag indicates whether the current SPI interface is busy or idle, corresponding to the register SPI_SR.BUSY. When SPI_SR.BUSY is 0, the SPI interface is in an idle state, and there is no data transmission currently; when SPI_SR.BUSY is 1, the SPI interface is in a busy state, and data transmission is currently being prepared or in progress.

Slave Select Input Rising Edge Flag Bit

The slave selection input rising edge flag indicates whether the current input slave selection signal has a rising edge, corresponding to the register SPI_SR.SSR. When SPI_SR.SSR is 0, there is no rising edge on the slave selection input; when SPI_SR.SSR is 1, there is a rising edge on the slave selection input.

Slave Select Input Falling Edge Flag Bit

The slave selection input falling edge flag indicates whether the current input slave selection signal has a falling edge, corresponding to the register SPI_SR.SSF. When SPI_SR.SSF is 0, there is no falling edge on the slave select input; when SPI_SR.SSF is 1, there is a falling edge on the slave select input.

Transmit Buffer Empty Flag Bit

The send buffer empty flag indicates whether the current SPI send buffer is loaded with data to be sent, corresponding to the register SPI_SR.TXE. When SPI_SR.TXE is 0, the SPI send buffer has been loaded with data to be sent; when SPI_SR.TXE is 1, the SPI send buffer is empty, and the next data to be sent can be written.

Receive Buffer Not Empty Flag Bit

The receiving buffer non-empty flag indicates whether there is received data in the receiving buffer of the current SPI, corresponding to the register SPI_SR.RXNE. When SPI_SR.RXNE is 0, the SPI receive buffer has not been loaded with valid received data; when SPI_SR.RXNE is 1, the SPI receive buffer is loaded with received data, which is available for reading.

Slave Selection Error Flag Bit in Slave Mode

When the SPI is working in slave mode and valid data transmission is in progress, the input slave selection signal should remain selected (low level) to ensure normal communication. If the slave selection is pulled high during this period, a slave selection error occurs, the corresponding register SPI_SR.SSERR will be set.

If a slave selection error occurs, the corresponding data transmission error will occur, and the SPI interface will also enter the unselected state due to the slave selection being pulled high. After a slave selection error occurs, the corresponding error flag is set and does not prevent subsequent normal data transmission, but the error flag will remain set until the user program clears it or the SPI is reset by the system control module.

It should be noted that the slave selection error SSERR is an error that only occurs in the slave mode, and it will only be triggered when the slave selection is pulled high when valid data is still being

transmitted. When there is no valid data transmission between frames, a normal slave select pulling high does not trigger this error.

Receive Buffer Overflow Error Flag Bit

The size of the SPI receive buffer is one frame. When the data in the receive buffer has not been read out and the data of the new frame has been received, a receive buffer overflow error occurs, and the corresponding register SPI_SR.OVF will be set.

When a receive buffer overflow error occurs, the original data in the buffer will be overwritten by the newly received data, resulting in data loss. After a receive buffer overflow error occurs, the corresponding error flag bit does not prevent subsequent normal data transmission, but the error flag bit will remain set until the user program clears it or the SPI is reset by the system control module.

Transmit Buffer Underflow Error Flag Bit in Slave Mode

SPI send buffer is one frame. When the send buffer is empty in slave mode and the transmission of a new frame starts, a send buffer underflow error occurs, and the corresponding register SPI_SR.UDF will be set.

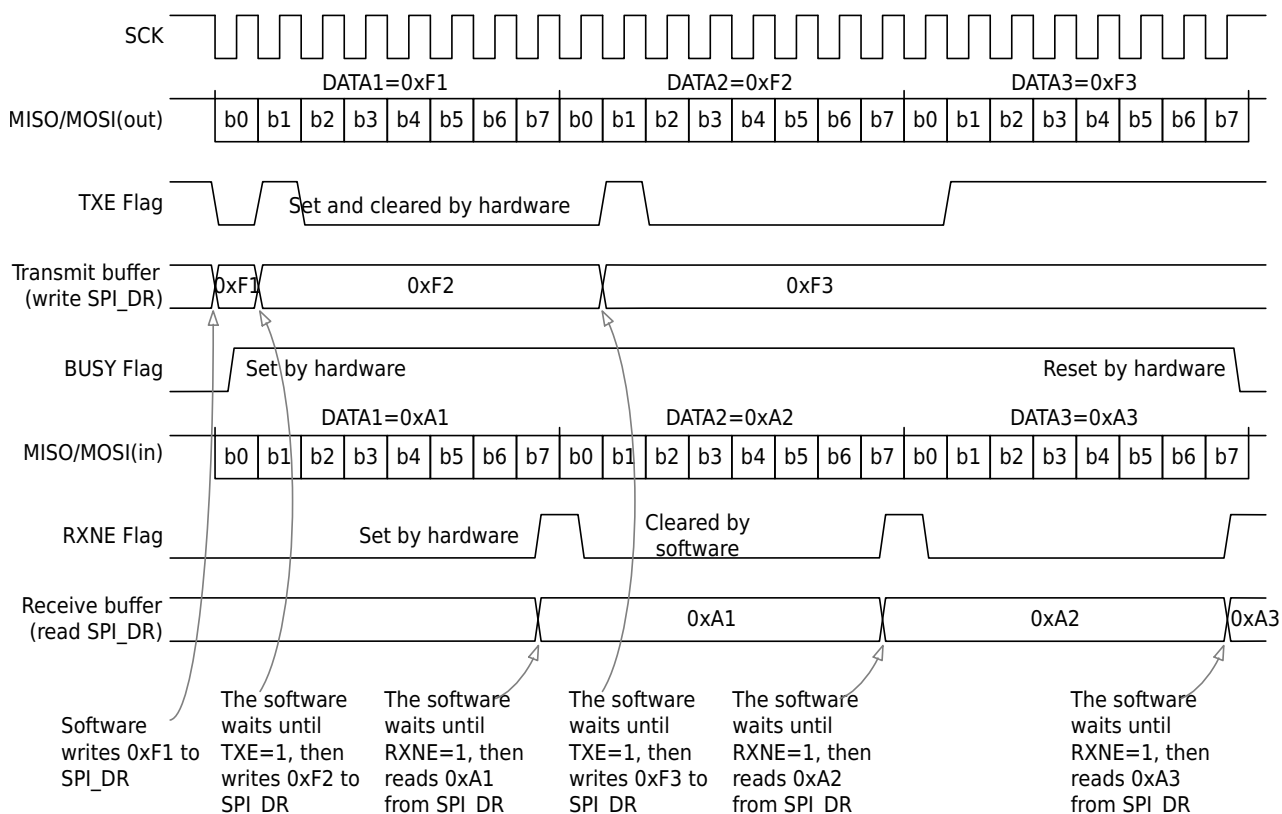
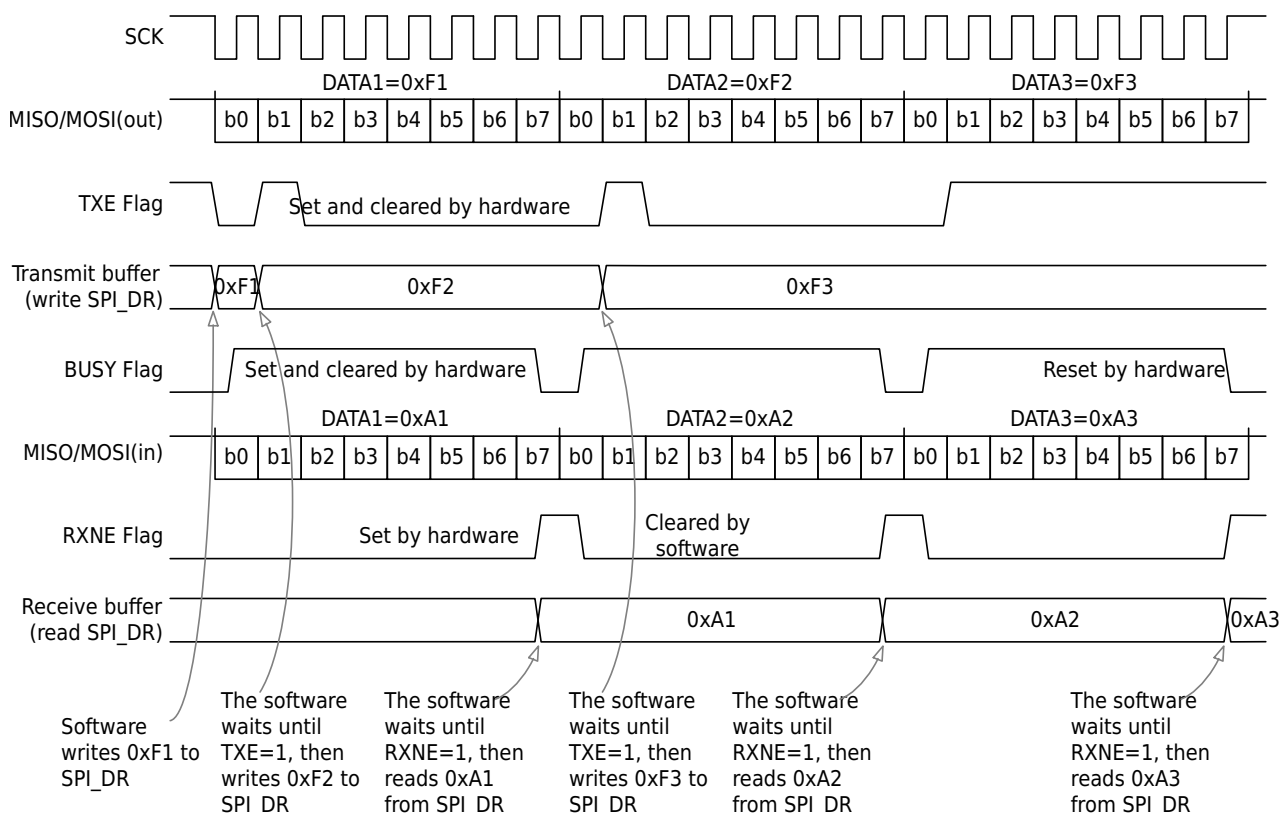
The occurrence of a send buffer underflow error in slave mode means that the send buffer is not filled with data in time, and the data sent to the master is missed. After a buffer underflow error occurs in the slave mode, the corresponding error flag bit does not prevent subsequent normal data transmission, but the error flag bit will remain set until the user program clears it or the SPI is reset by the system control module.

Mode Error Flag Bit in Master mode

When the SPI works in master mode and the slave selection pin is configured as an input, the input slave selection signal must remain unselected (high level) to ensure normal communication. If the slave selection is pulled low, a mode error will occur and the corresponding register SPI_SR.MODF will be set.

A mode error will cause the SPI to be forced to return to the disabled state (that is, SPI_CR0.EN is cleared), and the master-slave mode selection register (SPI_CR0.MSTR) will also be cleared (that is, switch to slave mode). After a mode error occurs, setting the corresponding error flag does not prevent the user program from re-enabling the SPI and performing normal data transmission, but the error flag will remain set until the user program clears it or the SPI is reset by the system control module.

It should be noted that the mode error MODF is an error that only appears in the specific configuration of the host mode. It will only be triggered when the slave selection is configured as an input and pulled low, and the error will not be triggered in other cases.

**Master Mode Example with
CPOL=1 and CPHA=1****Figure 16-9 Explanation Diagram of General Flags in SPI Master Mode****Slave mode example with
CPOL=1 and CPHA=1****Figure 16-10 SPI Slave Mode General Flag Bit Description Diagram**

16.3.10 Interrupt Generation and Clearing

SPI supports generating interrupt requests from different events, the specific information is as follows:

Table 16-1 SPI Interrupt

Interrupt Event	Corresponding Flag Bit (SPI_SR)	Corresponding Interrupt Enable (SPI_CR1)	Method to Clear the Interrupt Flag
Slave select input rising edge	SSR	SSRIE	Write 0 to SPI_ICR.SSR
Slave select input falling edge	SSF	SSFIE	Write 0 to SPI_ICR.SSF
Send buffer empty	TXE	TXEIE	Write data to the data register DR
Receive buffer is not empty	RXNE	RXNEIE	Read data from the data register DR or write 0 to SPI_ICR.RXNE.
Slave selection error in slave mode	SSERR	SSERRIE	Write 0 to SPI_ICR.SSERR
Receive buffer overflow error	OVF	OVFIE	Write 0 to SPI_ICR.OVF
Slave mode transmit buffer underflow error	UDF	UDFIE	Write 0 to SPI_ICR.UDF
Mode error in master mode	MODF	MODFIE	Write 0 to SPI_ICR.MODF

As shown in the above table, each interrupt event has an independent interrupt enable control, and the interrupt generated by the error event also has an independent interrupt clear register. For the above interrupt event (or request), no matter whether the interrupt is enabled or not, the method described in the table can be used to clear the interrupt flag.

16.4 Operation Example

16.4.1 Example of Sending and Receiving Data in Full-duplex by the Master

- Step1.** According to the relevant description of the pin digital multiplexing function in the chapter "General-Purpose I/O (GPIO)", map NSS, SCK, MOSI, and MISO to the required pins.
- Step2.** Configure SYCTRL module register PERI_CLKEN0.SPI to 1 to enable SPI configuration clock and working clock.
- Step3.** Set CR0.MSTR to 1, select the master mode.
- Step4.** Set CR0.BRR to configure the communication baud rate.
- Step5.** Set CR0.CM to 0, select full-duplex communication mode.
- Step6.** Set CR0.SSM to 1 to configure the slave selection mode.
- Step7.** Set CR0.CPOL, CR0.CPHA, select serial clock polarity and phase.
- Step8.** Set CR0.WIDTH to configure the data width of each frame.
- Step9.** Set CR0.LSBF to configure the data sequence order when sending and receiving data.
- Step10.** Set CR0.EN to 1 to enable the SPI module.
- Step11.** Set SSI.SSI to 0, select the corresponding slave.
- Step12.** Write the data to be sent into the DR register, and the SPI automatically starts data transmission.

- Step13.** Query and wait for SR.RXNE to become 1 to complete the sending and receiving of a frame of data.
- Step14.** Read the received data from the DR register and save it.
- Step15.** If the data to be sent and received has not been completed, jump to step12 to continue execution.

16.4.2 Example of Sending and Receiving Data in Full-duplex by the Slave

- Step1.** According to the relevant description of the pin digital multiplexing function in the chapter "General-Purpose I/O (GPIO)", map NSS, SCK, MOSI, and MISO to the required pins.
- Step2.** Configure SYSCTRL module register PERI_CLKEN0.SPI to 1 to enable SPI configuration clock and working clock.
- Step3.** Set CR0.MSTR to 0, select slave mode.
- Step4.** Set CR0.CM to 0, select full-duplex communication mode.
- Step5.** Set CR0.SSM to configure the slave selection mode.
- Step6.** Set CR0.CPOL, CR0.CPHA, select serial clock polarity and phase.
- Step7.** Set CR0.WIDTH to configure the data width of each frame.
- Step8.** Set CR0.LSBF to configure the data sequence order when sending and receiving data.
- Step9.** Set CR0.EN to 1 to enable the SPI module.
- Step10.** Set ICR.TXE to 0 to clear the send buffer.
- Step11.** The query waits for SR.SSLVL to become 0, and the SPI master selects this module.
- Step12.** Write the data to be sent into the DR register, and prepare the data to be sent.
- Step13.** Query and wait for SR.RXNE to become 1 to complete the sending and receiving of a frame of data.
- Step14.** Read the received data from the DR register and save it.
- Step15.** If the data to be sent and received has not been completed, jump to step12 to continue execution.
- Step16.** Wait for SR.SSLVL to become 1 to complete this communication.

16.4.3 Single-wire Half-duplex Sending and Receiving Data Example

- Step1.** According to the relevant description of the pin digital multiplexing function in the chapter "General-Purpose I/O (GPIO)", map NSS, SCK, MOSI of the master and NSS, SCK, MISO of the slave to the required pins .
- Step2.** Configure SYSCTRL module register PERI_CLKEN0.SPI to 1 to enable SPI configuration clock and working clock.
- Step3.** The master sets CR0.MSTR to 1, and the slave sets CR0.MSTR to 0.
- Step4.** The master sets CR0.BRR to configure the communication baud rate.
- Step5.** Set CR0.CM to 0b11, select single-line half-duplex communication mode.
- Step6.** Configure CR0.SSM to configure the slave selection mode.
- Step7.** Configure CR0.CPOL, CR0.CPHA, select serial clock polarity and phase.
- Step8.** Configure CR0.WIDTH to configure the data width of each frame.
- Step9.** Configure CR0.LSBF to configure the data sequence order when sending data.
- Step10.** Set CR0.EN to 1 to enable the SPI module.
- Step11.** The master sets SSI.SSI to 0 and selects the corresponding slave; the slave waits for SR.SSLVL to become 0 to select this module.

- Step12.** The master sets HDOE.HDOE to 1 and switches to the sending mode; the slave sets HDOE.HDOE to 0 and switches to the receiving mode.
- Step13.** The master writes the data to be sent into the DR register, and the SPI automatically starts data transmission.
- Step14.** The master query waits for SR.BUSY to become 0 to complete the transmission of a frame of data; the slave query waits for SR.RXNE to become 1 to complete the reception of a frame of data.
- Step15.** The slave reads the received data from the DR register and saves it.
- Step16.** If the data to be sent by the master is not completed, then jump to step13 to continue execution.
- Step17.** The master sets HDOE.HDOE to 0 and switches to the receiving mode; the slave sets HDOE.HDOE to 1 and switches to the sending mode.
- Step18.** The slave query waits for SR.TXE to become 1, and the sending buffer is idle.
- Step19.** If the data to be sent by the slave is not completed, write a frame of data to the DR register.
- Step20.** Writes arbitrary data to the DR register, and the SPI master sends SCK for data transmission.
- Step21.** The master query waits for SR.RXNE to become 1, and completes the reception of a frame of data.
- Step22.** The master reads the received data from the DR register and saves it.
- Step23.** If the data to be sent by the slave is not completed, then jump to step18 to continue execution.
- Step24.** The slave waits for SR.SSLVL to become 1, sets HDOE.HDOE to 0, and completes this communication.

16.5 Register Description

16.5.1 Register List

SPI base address: 0x40000800

Table 16-2 Serial Peripheral Interface (SPI) Register Offset

Offset	Register	Description	Width
0x00	SPI_CR0	SPI Control Register 0	32
0x04	SPI_CR1	SPI Control Register 1	32
0x08	SPI_HDOE	SPI Single-wire Half-duplex Output Enable Register	32
0x0C	SPI_SSI	SPI Internal Slave Select Register	32
0x10	SPI_SR	SPI Status Register	32
0x14	SPI_ICR	SPI Interrupt Clear Register	32
0x18	SPI_DR	SPI Data Register	32

16.5.2 SPI Control Register 0 (SPI_CR0)

Offset	Bit Position															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0x00000700															
Name	Reserved												SMP	CM	SSM	WIDTH
Access													RW	RW	RW	RW

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15	SMP	Master mode delayed sample bit 0b0: In traditional SPI mode, the data received by the master is sampled according to the settings of CPOL and CPHA 0b1: Delayed sampling method, the sampling time is delayed by half an SCK cycle compared with the traditional method	RW
14:13	CM	Communication mode configuration 0b00: full-duplex bidirectional 0b01: simplex transmit only 0b10: simplex receive only 0b11: single-wire half-duplex	RW
12	SSM	Slave selection configuration Master mode: 0b0: NSS pin determines the slave select input value 0b1: SSI register determines the slave select output value Slave mode: 0b0: NSS pin determines the slave select input value 0b1: SSI register determines the slave select input value	RW
11:8	WIDTH	Frame data width configuration 0b0011: 4-bit 0b0100: 5-bit ... 0b1110: 15-bit 0b1111: 16-bit - Others: Setting prohibited	RW
7	LSBF	Data frame high and low order selection 0b0: Most Significant Bit (MSB) transmitted and received first 0b1: Least Significant Bit (LSB) transmitted and received first	RW
6	MSTR	Operating mode configuration 0b0: Slave mode 0b1: Master mode	RW

Bit	Name	Functional Description	Access
5	CPOL	Serial clock polarity configuration ● 0b0: Low level during standby ● 0b1: High level during standby	RW
4	CPHA	Serial clock phase configuration ● 0b0: Front edge sampling/back edge shift ● 0b1: Front edge shift/back edge sampling	RW
3:1	BRR	Master mode baud rate configuration ● 0b000: PCLK/2 ● 0b001: PCLK/4 ● 0b010: PCLK/8 ● 0b011: PCLK/16 ● 0b100: PCLK/32 ● 0b101: PCLK/64 ● 0b110: PCLK/128 ● 0b111: Setting prohibited	RW
0	EN	SPI module enable control ● 0b0: SPI module disabled ● 0b1: SPI module enabled	RW

16.5.3 SPI Control Register 1 (SPI_CR1)

Offset	Bit Position																																	
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																								MODFIE	SSERRIE	OVFIE	UDFIE	SSRIE	SSFIE	RXNEIE	TXEIE		
Access																									RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Functional Description	Access
31:8	Reserved	-	-
7	MODFIE	Mode error interrupt enable control 0b0: Disable 0b1: Enable	RW
6	SSERRIE	Slave select error interrupt enable control in slave mode 0b0: Disable 0b1: Enable	RW
5	OVFIE	Receive buffer overflow error interrupt enable control 0b0: Disable 0b1: Enable	RW
4	UDFIE	Transmit buffer underflow error interrupt enable control in slave mode 0b0: Disable 0b1: Enable	RW
3	SSRIE	Slave select input rising edge interrupt enable control 0b0: Disable 0b1: Enable	RW
2	SSFIE	Slave select input falling edge interrupt enable control 0b0: Disable 0b1: Enable	RW
1	RXNEIE	Receive buffer non-empty interrupt enable control 0b0: Disable 0b1: Enable	RW
0	TXEIE	Transmit buffer empty interrupt enable control 0b0: Disable 0b1: Enable	RW

16.5.4 SPI Single-wire Half-duplex Output Enable Register (SPI_HDOE)

Offset	Bit Position																															
0x08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															HDOE
Access																																RW

Bit	Name	Functional Description	Access
31:1	Reserved	-	-
0	HDOE	MOSI/MISO transceiver switching control 0b0: Receive only 0b1: Transmit only  Note Only valid in single-wire half-duplex communication.	RW

16.5.5 SPI Internal Slave Select Register (SPI_SSI)

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000001																															
Name	Reserved																															SSI
Access																																RW

Bit	Name	Functional Description	Access
31:1	Reserved	-	-
0	SSI	Internal slave select Slave select output value is determined by the register in master mode. Slave select input value is determined by the register in slave mode.  Note Valid when the SPI_CR0.SSM register is 1.	RW

16.5.6 SPI Status Register (SPI_SR)

Offset	Bit Position																															
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000201																															
Name	Reserved																						SSLVL	BUSY	MODF	SSERR	OVF	UDF	SSR	SSF	RXNE	TXE
Access																							RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Functional Description	Access
31:10	Reserved	-	-
9	SSLVL	State of slave select input 0b0: Slave select input is low level 0b1: Slave select input is high level  Note This bit setting is invalid when in master mode and SPI_CR0.SSM=1.	RO
8	BUSY	Bus busy flag bit 0b0: Bus idle 0b1: Bus busy	RO
7	MODF	Mode error flag bit 0b0: Normal 0b1: Error  Note MODF is triggered when the slave select input is low in master mode.	RO
6	SSERR	Slave selection error flag in slave mode 0b0: Normal 0b1: Error	RO
5	OVF	Receive buffer overflow error flag bit 0b0: Normal 0b1: Error	RO
4	UDF	Transmit buffer underflow error flag in slave mode 0b0: Normal 0b1: Error	RO
3	SSR	Slave select input rising edge flag 0b0: No rising edge detected 0b1: Rising edge detected  Note This bit setting is invalid when in master mode and SPI_CR0.SSM=1.	RO

Bit	Name	Functional Description	Access
2	SSF	Slave select input falling edge flag ● 0b0: No falling edge detected ● 0b1: Falling edge detected  Note This bit setting is invalid when in master mode and SPI_CR0.SSM=1.	RO
1	RXNE	Receive buffer not empty flag ● 0b0: Receive buffer empty ● 0b1: Receive buffer not empty	RO
0	TXE	Send buffer empty flag ● 0b0: Transmit buffer not empty ● 0b1: Transmit buffer empty	RO

16.5.7 SPI Interrupt Clear Register (SPI_ICR)

Offset	Bit Position																																							
0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
Reset	0x000000FF																																							
Name	Reserved																								MODF		SSERR		OVF		UDF		SSR		SSF		RXNE		TXE	
Access																									WO		WO		WO		WO		WO		WO		WO		WO	

Bit	Name	Functional Description	Access
31:8	Reserved	-	-
7	MODF	SR.MODF flag clear bit Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
6	SSERR	SR.SSERR flag clear bit Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
5	OVF	SR.OVF flag clear bit Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
4	UDF	SR.UDF flag clear bit Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
3	SSR	SR.SSR flag clear bit Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
2	SSF	SR.SSF flag clear bit Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
1	RXNE	SR.RXNE flag clear bit Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO
0	TXE	SR.TXE clear bit Write 0 to set the SR.TXE flag to 1, write 1 has no effect; read always returns 1.	WO

16.5.8 SPI Data Register (SPI_DR)

Offset	Bit Position																															
0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																DR															
Access																																

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:0	DR	SPI data bits When sending data, write the data to be sent into this register When receiving data, read the received data from this register	RW

17 HSI2C

17.1 Introduction

I2C is a two-wire bidirectional synchronous serial bus, which uses a clock line and a data line to transmit information between two devices connected to the bus, providing a simple and efficient method for data exchange between devices. Each device connected to the bus has a unique address, and any device can be used as both a master and a slave, but only one master is allowed at the same time. I2C standard is a real multi-master bus with a conflict detection mechanism and an arbitration mechanism. It can use the arbitration mechanism to avoid data conflicts and protect data when multiple masters request control of the bus at the same time.

This I2C module complies with the System Management Bus specification (Version 3). SMBus is a single-ended, simple two-wire bus typically used for low-bandwidth communications.

17.2 Features

HSI2C supports:

- Supports standard mode, fast mode, fast enhanced mode, and high-speed mode.
- Hs-mode in slave mode.
- Supports multi-master, including synchronization and arbitration. Multi-master means any number of masters can be present. In addition, the master and slave roles can be changed during the transmission gap (after a STOP is sent).
- Supports 7-bit and 10-bit addressing.
- Supports software reset, START byte, and device ID (software support is also required).

HSI2C master supports:

- One word of command/transmit buffer (8-bit transmit data + 3-bit command), one word of receive buffer (8-bit receive data).
- The command buffer will wait for the I2C bus to be idle before starting the transmission.
- The command buffer can initialize (repeated) START and STOP conditions, as well as one or more master receive transmissions.
- The STOP condition can be generated from the command buffer or automatically when the transmit buffer is empty.
- Flexible receive data matching function can generate interrupts when data matching and/or discard unwanted data.
- Flags and optional interrupts to indicate Repeated START conditions, STOP conditions, arbitration loss, NACK, and command word errors.
- Supports configurable bus idle timeout and pin-stuck-low timeout.

HSI2C slave supports:

- Separate I2C slave registers to minimize software overhead for switching between master/slave.
- Supports SMBus alert address and broadcast address.
- Supports clock stretching.
- Supports transmit data register interrupt and receive data register interrupt.
- Software-controllable ACK or NACK, configurable ACK/NACK bit clock stretching. This is to avoid transmit buffer underrun and receive buffer overflow errors.

- Supports flags and interrupts for end of packet, STOP condition, or bit error detection.
- Support Ultra low power wake-up mode.

17.3 Functional Description

17.3.1 System Block Diagram

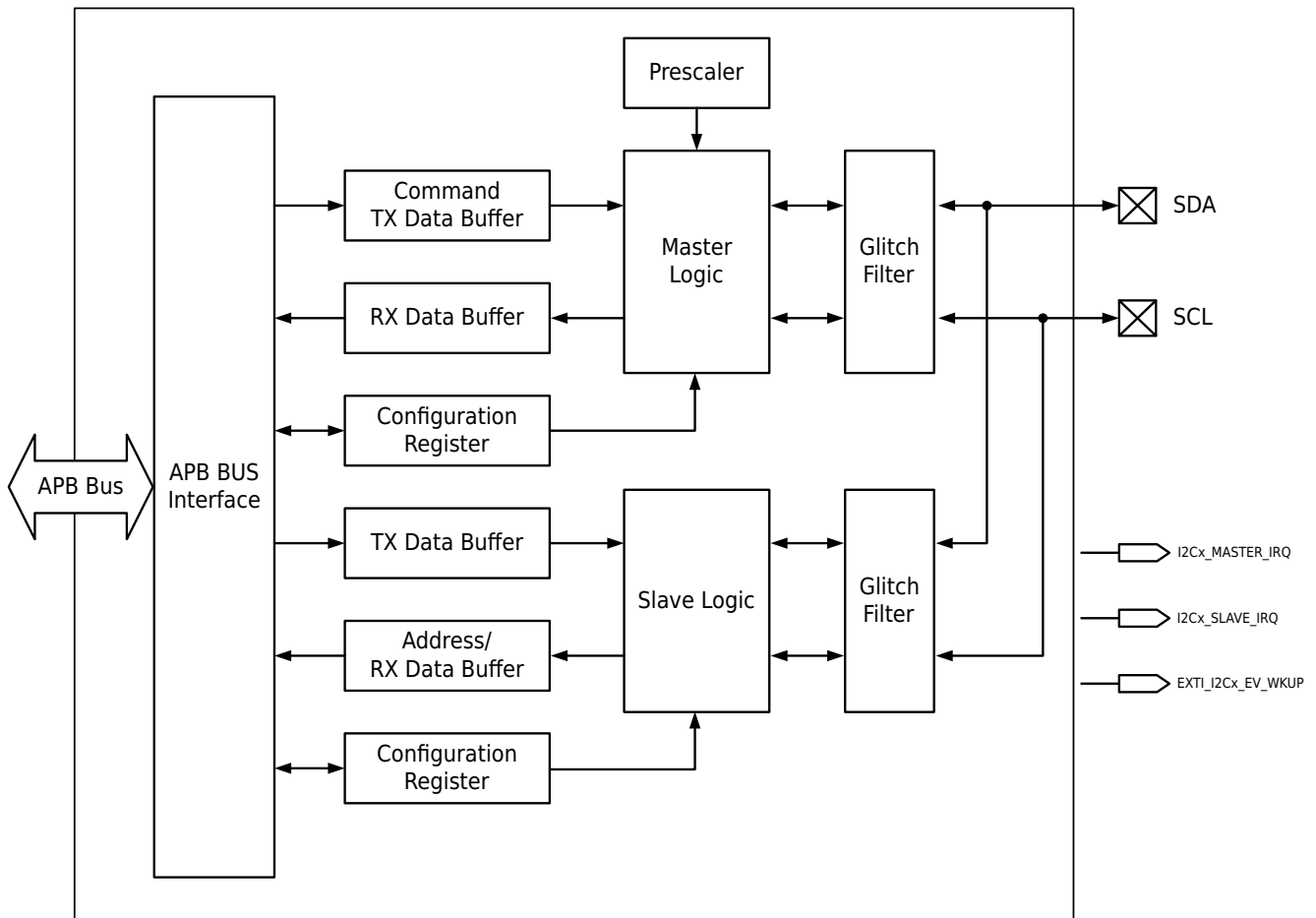


Figure 17-1 I2C System Block Diagram

17.3.2 Protocol Description

The I2C bus consists of a clock line (SCL) and a data line (SDA). All connection devices must be drain open output. The SCL and SDA lines are connected to external pull-up resistors, and the resistance value depends on the system application.

In general, a complete communication process consists of the following four parts:

1. START condition
2. Address transmission
3. Data transmission
4. STOP condition

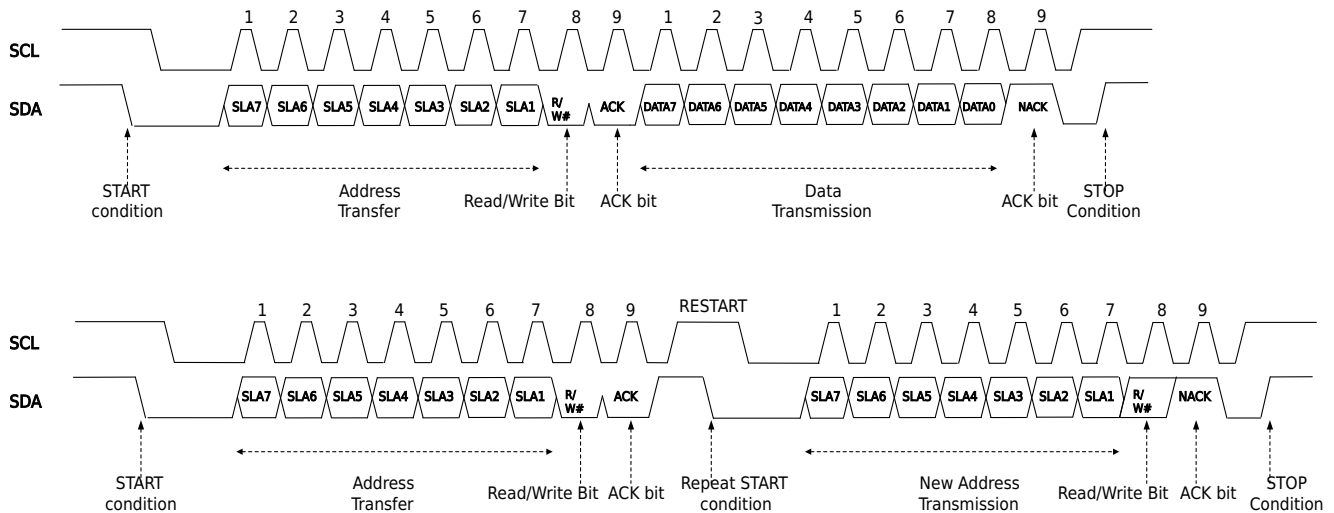


Figure 17-2 I2C Bus Timing Diagram

17.3.3 Master Mode

17.3.3.1 Transmit and Command Buffer Commands

The transmit buffer stores command data to initiate various I2C operations. As long as I2C is enabled (MCR.MEN bit is set), the following operations can be initiated by commands in the transmit buffer:

- Address byte and START or STOP condition expecting ACK or NACK.
- Transmit data (this is the default behavior for zero-extended byte writes to the transmit buffer).
- Receive a byte of data (can also be configured to discard the receive data and not store it in the receive buffer).
- STOP condition (can also be configured to send a STOP condition when the transmit buffer is empty).

Multiple transmit and receive commands can be inserted between start and stop conditions; transmit and receive commands must not be interleaved. Receive data commands, receive data, and discard commands can be interleaved to ensure that only the desired receive data is stored in the receive buffer.

The I2C master automatically sends a NACK on the last byte of a receive data command unless the next command in the buffer is also a receive data command. A NACK is also automatically sent if the transmit buffer is empty when a receive data command completes.

The I2C master enable (MCR.MEN) should be turned off after all transmits are complete.

17.3.3.2 Master Operation

Whenever I2C is started, it monitors the I2C bus to detect when the I2C bus is idle (MSR.BBF). The I2C bus is no longer considered idle if SCL or SDA is low; the I2C bus becomes idle if a STOP condition is detected or the bus idle timeout (MCFGR2.BUSIDLE configuration) occurs. After the I2C bus is idle, the transmit buffer is not empty, and then the I2C master initiates a transfer on the I2C bus. Refer to the following steps:

- The bus idle wait time is equal to $(MCCR0.CLKLO + 1)$ multiplied by the prescaler (MCFGR1.PRE-SCALE).
- The START condition and address byte are sent using the timing configuration in the Master Clock Configuration 0 (MCCR0); if configured for high-speed mode, the timing configuration of the Master Clock Configuration 1 (MCCR1) is used.

According to the configuration of the transmit buffer, either a master transmit transfer or a master receive transfer is performed.

- A NACK is sent on the last byte of a master receive transfer, unless the next command in the transmit buffer is also a receive data command and the transmit buffer is not empty.
- The sending of a repeated START or STOP condition is configured by the transmit buffer and/or MCFGR1.AUTOSTOP. The repeated START condition can change the timing configuration register used.

When the I2C master is disabled (MCR.MEN is cleared or mode is entered), the I2C continues to empty the transmit buffer until a STOP condition is sent. However, the I2C will no longer pause the I2C bus waiting for the transmit or receive buffer, and the I2C automatically generates a STOP condition after the transmit buffer is empty.

The I2C master can stop the I2C bus under the following specific conditions; this causes SCL to be pulled low continuously on the first bit of the byte until the condition is removed:

- The I2C master is enabled and busy, the transmit buffer is empty, and MCFGR1.AUTOSTOP is cleared.
- The I2C master is enabled and receiving data, the receive data has not been discarded (due to a command or receive data match), and the receive buffer is full.

17.3.3.3 Receive Buffer and Data Matching

The receive buffer is used to store receive data during the master receive transmission. The receive data can also be configured to discard the receive data instead of storing it in the receive buffer; this is configured by the command word in the transmit buffer.

The receive data supports the receive data matching function, which can match the received data with one of the two bytes or the mask data byte. The data match function can also be configured to compare only the previous one or two received data words since the last (repeated) START condition. Received data that is discarded due to a command word will not cause a data match, and the match on the first received data word is delayed until after the discarded data is received. The receiver match function can also be configured to discard all received data until a data match is detected, using the MCFGR1.RDMO control bit. Clear the MCFGR1.RDMO control bit after data match and clear MCFGR1.RDMO before clearing MSR.DMF to allow all subsequent data to be received.

17.3.3.4 Timing Parameters

The I2C master can configure the following timing parameters. The parameters for high-speed mode (Master Clock Configuration 1 (MCCR1)) and other modes (Master Clock Configuration 0 (MCCR0)) can be configured separately. This allows high-speed mode master code to be transmitted using standard timing parameters, then switching to high-speed mode timing (after the repeated START), until the next STOP condition occurs.

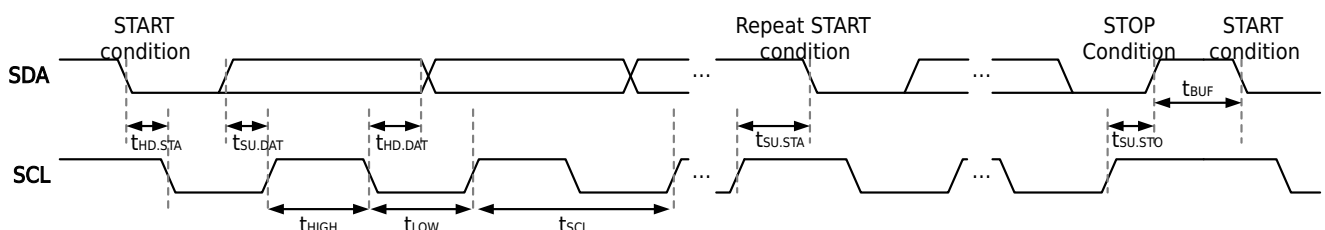


Figure 17-3 I2C Bus Timing Parameters Diagram

The timing parameters of the I2C master are usually related to the I2C function clock cycle, and the parameter configuration needs to meet the specifications required by the corresponding mode.

Table 17-1 Host Timing Parameters

Parameters	Symbol	Timing Parameters (I2C function clock cycle)
SCL clock cycle	t_{SCL}	$(CLKHI + CLKLO + 2 + SCL_LATENCY) \times 2^{PRESCALE}$
SCL clock low time	t_{LOW}	$(CLKLO + 1) \times 2^{PRESCALE}$
SCL clock high time	t_{HIGH}	$(CLKHI + 1 + SCL_LATENCY) \times 2^{PRESCALE}$
(Repeated) START condition hold time	$t_{HD:STA}$	$(SETHOLD + 1) \times 2^{PRESCALE}$
Data setup time	$t_{SU:DAT}$	$(SDA_LATENCY + 1) \times 2^{PRESCALE}$
Data hold time	$t_{HD:DAT}$	$(DATAVD + 1) \times 2^{PRESCALE}$
Repeated START condition setup time	$t_{SU:STA}$	$(SETHOLD + 1 + SCL_LATENCY) \times 2^{PRESCALE}$
STOP condition setup time	$t_{SU:STO}$	$(SETHOLD + 1 + SCL_LATENCY) \times 2^{PRESCALE}$
Bus Idle (STOP condition to START condition)	t_{BUF}	$(CLKLO + 1 + SDA_LATENCY) \times 2^{PRESCALE}$
Data validity period	$t_{VD:DAT}$	$(DATAVD + 1) \times 2^{PRESCALE}$
Data response valid time	$t_{VD:ACK}$	$(DATAVD + 1) \times 2^{PRESCALE}$

The following table defines the delay parameters, which assume that the rise time is less than one I2C function clock cycle. The rise time depends on several factors, including I/O propagation delay, I2C bus loading, and the value of the external pull-up resistor. A larger rise time increases the number of clock cycles required for the signal to pass through the synchronizer (and glitch filter), thus increasing latency.

Table 17-2 Synchronization Delay

Parameters	Definition
SCL_LATENCY	$\text{ROUNDDOWN}((2 + \text{FILTSC} + \text{SCL_RISETIME})/2^{PRESCALE})$
SDA_LATENCY	$\text{ROUNDDOWN}((2 + \text{FILTS} + \text{SDA_RISETIME})/2^{PRESCALE})$

The I2C configuration needs to follow the following timing restrictions to avoid unexpected START or STOP conditions on the I2C bus and to prevent the I2C master from detecting unexpected START or STOP conditions. SDA cannot change while SCL is high outside of sending a (repeated) START or STOP condition.

Table 17-3 I2C Timing Parameter Limitations

Parameters	Minimum	Maximum	Description
CLKLO	0x03	-	Satisfy $CLKLO > SCL_LATENCY$ at the same time
CLKHI	0x01	-	Configure CLKHI to meet the duty cycle requirements in the I2C specification
SETHOLD	0x02	-	Satisfy both $SETHOLD > SDA_LATENCY$
DATAVD	0x01	$CLKLO - SDA_LATENCY - 1$	Configure DATAVD to meet the data retention requirements in the I2C specification

Parameters	Minimum	Maximum	Description
FILTSL	0x00	$[\text{CLKLO} \times 2^{\text{PRESCALE}}] - 3$	FILTCSL and FILTSDA are the only parameters that are not multiplied by 2^{PRESCALE}
FILTSDA	FILTSL	$[\text{CLKLO} \times 2^{\text{PRESCALE}}] - 3$	Configuring FILTSDA greater than FILTSL can delay the SDA input to compensate for board-level offsets
BUSIDLE	$(\text{CLKLO} + \text{SETHOLD} + 2) \times 2$	-	Satisfy $\text{BUSIDLE} > \text{CLKHI} + 1$ at the same time

The timing parameters must be configured to meet the requirements of the I2C specification; this depends on the supported modes and the I2C function clock frequency. When switching between two modes (for example, fast and high-speed modes) using different clock configuration registers, the PRESCALE factor must remain constant between the modes. Some example timing configurations are provided below.

Table 17-4 I2C Example Timing Configuration

I2C Mode	Clock Frequency	Baud Rate	PRE-SCALE	FILTSL/ FILTSDA	SETHOLD	CLKO	CLKHI	DATAVD
Standard	8MHz	100kbps	0x0	0xD/0xF	0x37	0x29	0x16	0x0D
Fast	8MHz	400kbps	0x0	0x0/0x0	0x04	0x0B	0x05	0x02
Fast+	8MHz	1Mbps	0x0	0x0/0x0	0x02	0x03	0x01	0x01
Standard	48MHz	100kbps	0x2	0x2/0xE	0x3A	0x3B	0x39	0x0C
Fast	48MHz	400kbps	0x0	0x1/0x1	0x1D	0x3E	0x35	0x0F
Fast	48MHz	400kbps	0x2	0x1/0x1	0x07	0x11	0x0B	0x03
Fast+	48MHz	1Mbps	0x2	0x1/0x1	0x03	0x06	0x04	0x04
HS Mode	48MHz	3.2Mbps	0x0	0x0/0x0	0x07	0x08	0x03	0x01

17.3.3.5 Error Condition Description

The I2C master monitors for errors while active. The following conditions will generate error flags and block the issuance of a new START condition until the flags are cleared by software:

- A START or STOP condition is detected that was not generated by the I2C master (MSR.ALF is set). Software must respond to the MSR.ALF flag by terminating the current command (by setting MCR.RST).
- A different value is received on SDA than the data being transmitted (MSR.ALF is set).
- A NACK is detected during data transmission and MCFGR1.IGNACK is cleared (MSR.NDF is set).
- A NACK is detected when an ACK was expected for the address byte, and MCFGR1.IGNACK is cleared (MSR.NDF is set).
- An ACK is detected when a NACK was expected for the address byte, and MCFGR1.IGNACK is cleared (MSR.NDF is set).
- A data transmission or reception is attempted by the transmit buffer without a preceding START condition (MSR.FEF is set).

- SCL (or SDA if MCFGR1.TIMECFG is set) remains low for (MCFGR3.PINLOW × 256) prescaler cycles without a pin state change (MSR.PLTF is set).

Software must respond to the MSR.PLTF flag by terminating the current command (by setting MCR.RST).

The MCFGR2.BUSIDLE field can be configured to treat the I2C bus as idle only if both SCL and SDA remain high for (MCFGR2.BUSIDLE + 1) prescaler cycles. Normally, the I2C bus is considered idle when the master is first enabled. However, when MCFGR2.BUSIDLE is configured to a non-zero value, the bus will only be considered idle after both SCL and SDA remain high for (MCFGR2.BUSIDLE + 1) prescaler cycles.

17.3.3.6 Interrupts

The table below lists the master mode flags that can generate I2C master interrupts.

Flag	Description	Interrupt	Wakeup
TDF	Data can be written to the transmit buffer.	✓	-
RDF	Data can be read from the receive buffer.	✓	-
EPF	The master has issued a repeated START or STOP condition.	✓	-
SDF	The master has issued a STOP condition.	✓	-
NDF	● A NACK was detected on the address byte when the master expected an ACK. ● An ACK was detected on the address byte when the master expected a NACK. ● A NACK was detected by the host during transmission of a data byte.	✓	-
ALF	● A START or STOP condition was detected at an invalid time, and the master lost arbitration. ● The master was transmitting data, but the received data did not match the transmitted data.	✓	-
FEF	The master expected a START condition in the command buffer, but the next entry in the command buffer was not a START condition	✓	-
PLTF	Pin low timeout is enabled and SCL (or SDA if configured) is held low for longer than the configured timeout period	✓	-
DMF	The received data matches the configured data, and the received and discarded data configured by MTDR.CMD will not set DMF	✓	-
MBF	The I2C master is busy sending/receiving data	-	-
BBF	The I2C master is enabled and activity is detected on the I2C bus, but no STOP condition is detected and no bus idle timeout occurs (if enabled)	-	-

17.3.4 Slave Mode

To perform all slave mode transfers on the I2C bus, the I2C slave logic runs independently of the I2C master logic.

17.3.4.1 Address Matching

The I2C slave can be configured:

- Matches one of two addresses in 7-bit or 10-bit addressing mode.
- Matches a range of addresses in 7-bit or 10-bit addressing mode.
- Matches a broadcast address and generates the corresponding flag.
- Matches an SMBus alert address and generates the corresponding flag.
- Detects high-speed mode master code, disables digital filters and output valid delay time until the next STOP condition is detected.

After matching a valid address, the I2C slave automatically performs a slave transmit transfer or a slave receive transfer until:

- A NACK is detected (unless SCR1.IGNACK is set)
- A bit error is detected (the I2C slave is driving SDA, but a different value is sampled)
- A (repeated) START or STOP condition is detected

17.3.4.2 Transmit and Receive Data

- The transmit and receive data registers are double buffered and updated only during slave transmit transfers and slave receive transfers respectively.
- The received slave address can be configured to be read from the Slave Receive Data Register (SRDR) or the Slave Address Status Register (SASR).
- The Transmit Data Register can be configured to request data only after a slave transmit transfer is detected, or to request new data when the Transmit Data Register is empty.
- The Slave Transmit Data Register (STDR) can be written only when the Transmit Data Flag (SSR.TDF) is set high.
- The Slave Receive Data Register (SRDR) can be read only when the Receive Data Flag (SSR.RDF) is set high (or the Address Valid Flag is set high and SCR1.RxCFG=1).
- The Slave Address Status Register (SASR) can be read only when the Address Valid Flag (SSR.AVF) is set high.

17.3.4.3 Clock Stretching

The I2C slave supports clock stretching. The following conditions can be configured to perform clock stretching:

- The Address Valid Flag (SSR.AVF) is set high during the 9th clock pulse of the address byte.
- The Transmit Data Flag (SSR.TDF) is set high during the 9th clock pulse of the slave transmit transfer.
- The Receive Data Flag (SSR.RDF) is set high during the 9th clock pulse of the slave receive transfer.
- The Transmit ACK Flag (SSR.TAF) is set high during the 8th clock pulse of the address byte or slave receive transfer. In high-speed mode, this configuration is disabled.
- Clock stretching can also be extended to the SCR2.CLKHOLD period to allow additional setup time to support external sampling of the SDA pin. In high-speed mode, this configuration is disabled.

Unless extended by the SCR2.CLKHOLD configuration, clock stretching will extend one peripheral bus clock cycle after SDA is updated when clock stretching is enabled.

17.3.4.4 Timing Parameters

The I2C slave can configure the following timing parameters. These parameters are disabled when SCR2.FILTEN is cleared or when the I2C slave detects high-speed mode. When disabled, the I2C slave

is clocked directly from the I2C bus and may not meet all timing requirements of the I2C specification (such as SDA minimum hold time in standard/fast mode).

- SDA data valid time from SCL pull low to SDA update
- SCL hold time when clock stretching is enabled to increase setup time when sampling external SDA
- SCL glitch filter time
- SDA glitch filter time

The I2C slave has the following restrictions on timing parameters:

- SCR2.FILTSDA must be configured to be greater than or equal to SCR2.FILTSCS (unless compensating for board level offsets between SDA and SCL).
- The configuration of SCR2.DATAVD must be less than the minimum SCL low period.

17.3.4.5 Error Condition Description

An I2C slave can detect the following error conditions:

- When the I2C slave drives SDA and the sampled value is different from the expected value, the bit error flag (SSR.BEF) will be set high.
- When the transmit data is underrun or the receive data is overflowed, the buffer error flag (SSR.FEF) will be set high. To avoid underrun and overflow, turn on the clock stretching function.
- When SCR1.RxCFG is set high, if the address overflows, the buffer error flag (SSR.FEF) will be set high, otherwise the address overflow will not be marked. To avoid overflow, the clock stretching function can be turned on.

The I2C slave will not implement time-out detection due to SCL and/or SDA being pulled low. If detection of this condition is required, the I2C master logic should be used so that software can reset the I2C slave if such a condition is detected.

17.3.4.6 Interrupts

The following table lists the slave mode flags that can generate I2C slave interrupts.

Flag	Description	Interrupt	Wakeup
TDF	Data can be written to the slave transmit data register (STDR)	✓	-
RDF	Data can be read from the slave receive data register (SRDR)	✓	-
AVF	The address can be read from the Slave Address Status Register (SASR)	✓	✓
TAF	ACK/NACK can be written to the slave transmit ACK register (STAR)	✓	-
RSF	The slave detects an address match followed by a repeated START condition	✓	-
SDF	The slave detects an address match and a STOP condition follows	✓	-
BEF	The slave is sending data, but the data received is different from the data sent	✓	-
FEF	● Sending data underload ● Receive data overflow ● Address status overflow (when receiving data configuration SCR1.RxCFG = 1) The FEF flag can only be set when clock stretching is disabled.	✓	-
AM0F	The slave detects an address match with SADDR.ADDR0	✓	-

Flag	Description	Interrupt	Wakeup
AM1F	The slave detects an address match with SADDR.ADDR1 or uses an address range	✓	-
GCF	The slave detects an address that matches the broadcast address	✓	-
SARF	The slave detects an address that matches the SMBus alert address	✓	-
SBF	The I2C slave is busy receiving an address byte or sending/receiving data.	-	-
BBF	The I2C slave is enabled and a START condition is detected on the I2C bus, but a STOP condition is not detected	-	-

17.3.5 Communication Level Selection Function

The HSI2C interface supports signal communication at different voltage levels. For functional details, refer to the corresponding descriptions in "[HSI2C Communication Level Selection Function](#)" of "General-Purpose I/O (GPIO)" chapter.

17.4 Operation Example

17.4.1 Master Transmission Mode

The operation flow in the master transmission mode is as follows:

- Step1.** Enable the I2C module clock and reset the I2C module.
- Step2.** Configure the I2C module SCL and SDA pins.
- Step3.** Configure the I2C register according to the transmission baud rate so that the frequency of the sent SCL is consistent with the transmission baud rate:
 - MCFGR1.PRESCALE
 - MCCR0.CLKLO
 - MCCR0.CLKHI
 - MCCR0.DATAVD
 - MCCR0.ETHOLD
- Step4.** Set MCR.MEN to 1 to enable the I2C module.
- Step5.** Check whether MSR.BBF is 0. If it is 0, proceed to the next step.
- Step6.** Write a START instruction and address + write operation to the MTDR register, for example 0x4AA.
- Step7.** After waiting for MSR.TDF to be 1, write the data to be sent into MTDR, continue to query MSR.TDF, and wait for MSR.TDF to be 1 before writing the next data to be sent into MTDR. This step is repeated until all the data to be sent are written into the MTDR.
- Step8.** Continue to query MSR.TDF, wait for MSR.TDF to be 1, and then write a STOP command to MTDR.CMD, for example, 0x2. Check whether the I2C master MSR.SDF is 1. If it is 1, it means that the STOP condition has been issued and the sending operation is completed.

17.4.2 Master Receiving Mode

The operation flow in the master receiving mode is as follows:

- Step1.** Enable the I2C module clock and reset the I2C module.
- Step2.** Configure the I2C module SCL and SDA pins.

- Step3.** Configure the I2C register according to the transmission baud rate so that the frequency of the sent SCL is consistent with the transmission baud rate:
- MCFGR1.PRESCALE
 - MCCR0.CLKLO
 - MCCR0.CLKHI
 - MCCR0.DATAVD
 - MCCR0.SETHOLD
- Step4.** Set MCR.MEN to 1 to enable the I2C module.
- Step5.** Check whether MSR.BBF is 0. If it is 0, proceed to the next step.
- Step6.** Write the START instruction and address + read operation to the MTDR register, for example, 0x4AB.
- Step7.** Wait for MSR.TDF to be 1 and then write the receive data instruction + data quantity (the data quantity is the actual received data quantity minus 1) to MTDR, for example, 0x103, indicating that 4 data are received.
- Step8.** Wait for MSR.RDF to be 1 and then read the received data from MRDR, continue to query MSR.RDF, and wait for MSR.RDF to be 1 and then read the next received data from MRDR. Repeat this step until all received data are read from MRDR.
- Step9.** Continue to query MSR.TDF, wait for MSR.TDF to be 1, and then write a STOP command to MTDR.CMD, for example, 0x2. Check whether the I2C master MSR.SDF is 1. If it is 1, it means that the STOP condition has been issued and the receiving operation is completed.

17.4.3 Slave Transmission Mode

The operation flow in slave transmission mode is as follows:

- Step1.** Enable the I2C module clock and reset the I2C module.
- Step2.** Configure the I2C module SCL and SDA pins.
- Step3.** Configure I2C slave control 1 register and slave address register:
- SCR1.ADDRMODE
 - SCR1.TXCFG
 - SADDR.ADDR0
- Step4.** Set SCR0.SEN.EN to 1 to enable the I2C module.
- Step5.** Query whether SSR.AVF is 1. If it is 1, proceed to the next step.
- Step6.** Read SASR.RADDR[0]:
- 0 indicates master write operation and the slave is in receive mode;
 - 1 indicates master read operation and the slave is in transmit mode.
- Step7.** Wait for SSR.TDF to be 1 before writing the data to be sent to STDR. Continue to query SSR.TDF. Wait for SSR.TDF to be 1 before writing the next data to be sent to STDR. Repeat this step until all data to be sent are written to STDR.
- Step8.** Query whether SSR.AVF is 0. If it is 0, it means that the I2C address status register SASR has been read. Query the I2C slave SSR.SDF to see if it is 1. If it is 1, it means that the STOP condition has been detected and the sending operation is finished.

17.4.4 Slave Receiving Mode

The operation flow in slave receiving mode is as follows:

- Step1.** Enable the I2C module clock and reset the I2C module.

- Step2.** Configure the I2C module SCL and SDA pins.
- Step3.** Configure I2C slave control 1 register and slave address register:
- SCR1.ADDRMODE
 - SADDR.ADDR0
- Step4.** Set SCR0.SEN.EN to 1 to enable the I2C module.
- Step5.** Query whether SSR.AVF is 1. If it is 1, proceed to the next step.
- Step6.** Read SASR.RADDR[0]:
- 0 indicates master write operation and the slave is in receive mode;
 - 1 indicates master read operation and the slave is in transmit mode.
- Step7.** Wait for SSR.RDF to be 1 before reading the received data from SRDR, continue to query SSR.RDF, wait for SSR.RDF to be 1 before reading the next received data from SRDR. Loop this step until all received data is read from SRDR.
- Step8.** Query whether SSR.AVF is 0, which means that the I2C address status register SASR has been read, query whether the I2C slave SSR.SDF is 1, which means that the STOP condition has been detected and the receiving operation is over.

17.4.5 Master High-Speed Mode Switching

After a reset, the master operates in fast mode. To switch to high-speed mode, the master must complete the following sequence of operations:

- Step1.** Enable the I2C module clock and reset the I2C module.
- Step2.** Configure the I2C module SCL and SDA pins.
- Step3.** Configure the I2C register according to the transmission baud rate so that the frequency of the sent SCL is consistent with the transmission baud rate:
- MCFGR1.PRESCALE
 - MCCR0.CLKLO
 - MCCR0.CLKHI
 - MCCR0.DATAVD
 - MCCR0.ETHOLD
- Step4.** Configure the I2C registers according to the baud rate of high-speed mode to ensure that the frequency of the SCL pulses transmitted after switching matches the baud rate of high-speed mode.
- MCCR1.CLKLO
 - MCCR1.CLKHI
 - MCCR1.DATAVD
 - MCCR1.ETHOLD
- Step5.** Set MCR.MEN to 1 to enable the I2C module.
- Step6.** Check whether MSR.BBF is 0. If it is 0, proceed to the next step.
- Step7.** Write the START command and control code (0x00001xxx) to the MTDR register, for example, 0x509. After the master expects a NACK response, switch to high-speed mode.
- Step8.** Wait until MSR.TDF=1, then write the START command and address + read/write operation to MTDR register, for example: 0x6AB/0x6AA.
- Step9.** Subsequent operation:
- Read operations: refer to the [Host Receive Mode](#) steps7-9.
 - Write operations: refer to the [Host Transmit Mode](#) steps7-8.

17.5 Register Description

17.5.1 Register List

HSI2C base address: 0x4000BC00

Table 17-7 High Speed Integrated Circuit Bus (HSI2C) Register Offset

Offset	Register	Description	Width
0x010	I2C_MCR	Master control register	32
0x014	I2C_MSR	Master status register	32
0x018	I2C_MIER	Master interrupt enable register	32
0x01C	I2C_MSCR	Master status clear register	32
0x024	I2C_MCFGR1	Master configuration 1 register	32
0x028	I2C_MCFGR2	Master configuration 2 register	32
0x02C	I2C_MCFGR3	Master configuration 3 register	32
0x030	I2C_MCCR0	Master clock configuration 0 register	32
0x034	I2C_MCCR1	Master clock configuration 1 register	32
0x038	I2C_MTDR	Master transmit data register	32
0x03C	I2C_MRDR	Master receive data register	32
0x040	I2C_MDMR	Master data configuration register	32
0x110	I2C_SCR0	Slave control 0 register	32
0x114	I2C_SSR	Slave status register	32
0x118	I2C_SIER	Slave interrupt enable register	32
0x11C	I2C_SSCR	Slave status clear register	32
0x120	I2C_SCR1	Slave control 1 register	32
0x124	I2C_SCR2	Slave control 2 register	32
0x128	I2C_STAR	Slave transmit ACK register	32
0x12C	I2C_STDR	Slave transmit data register	32
0x130	I2C_SRDR	Slave receive data register	32
0x140	I2C_SADDR	Slave address register	32
0x150	I2C_SASR	Slave address status register	32

17.5.2 Master Control Register (I2C_MCR)

Offset	Bit Position																																				
0x010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Reset	0x00000000																																				
Name	Reserved																														DBGEN		Reserved	RST		MEN	
Access																															RW			RW		RW	

Bit	Name	Functional Description	Access
31:4	Reserved	-	-
3	DBGEN	Debug Enable Enable the master in debug mode. 0b0: Disable the master in debug mode 0b1: Enable the master in debug mode	RW
2	Reserved	-	-
1	RST	Software Reset 0b0: Master logic is not reset 0b1: Master logic reset Resets all internal master logic and registers, except the Master Control Register (MCR). RST remains asserted until cleared by software. The reset takes effect immediately and remains set until cleared by software. No minimum delay is required before clearing the software reset.	RW
0	MEN	Master Enable 0b0: Disable master mode 0b1: Enable master mode	RW

17.5.3 Master Status Register (I2C_MSR)

Offset	Bit Position																																			
0x014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000001																																			
Name	Reserved						BBF	MBF	Reserved							DMF	PLTF		FEF		ALF		NDF		SDF		EPF		Reserved						RDF	TDF
Access							RO	RO								RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO								RO	RO	

Bit	Name	Functional Description	Access
31:26	Reserved	-	-
25	BBF	Bus busy flag 0b0: I2C bus idle 0b1: I2C bus busy	RO
24	MBF	Master busy flag 0b0: I2C master idle 0b1: I2C master busy	RO
23:15	Reserved	-	-
14	DMF	Data match flag 0b0: Matching data has not been received 0b1: Matching data has been received Indicates that the received data has matched the MDMR.MATCH0 and/or MDMR.MATCH1 fields (configured by MCFGR1.MATCFG). Received data that is discarded due to the MTDR.CMD field does not cause the DMF to be set.	RO
13	PLTF	Pin low timeout flag 0b0: Pin Low Timeout has not occurred or is disabled 0b1: Pin Low Timeout has occurred Set when the SCL and/or SDA inputs are low for longer than the pin low period (pin low timeout, MCFGR3.PINLOW), even when the I2C master is idle. 1. Software is responsible for resolving pin low conditions 2. PLTF must not be cleared as long as the pin low timeout persists 3. PLTF must be cleared before the I2C can initiate a START condition.	RO
12	FEF	Buffer error flag 0b0: No error. 0b1: Master sending or receiving data without a START condition. Detects an attempt to send or receive data without first generating a (repeated) START condition. This can occur if the transmit buffer underflows when MCFGR1.AUTOSTOP=1. When FEF is set, the I2C master sends a STOP condition (if busy) and will not initiate a new START condition until FEF is cleared.	RO

Bit	Name	Functional Description	Access
11	ALF	Arbitration lost flag 0b0: Master did not lose arbitration 0b1: Master lost arbitration Set if any of the following conditions exist: - The I2C master sends a logic 1 and detects a logic 0 on the I2C bus - The I2C master detects a START or STOP condition while sending data When ALF is set, the I2C master releases the I2C bus (becomes idle), and the I2C master will not initiate a new START condition until ALF is cleared.	RO
10	NDF	NACK detection flag 0b0: NACK not expected does not detected 0b1: NACK not expected detected Set if the I2C master detects a NACK that it did not expect when sending an address or data. When asserted, the master will not initiate a new START condition until NDF is cleared. If a NACK is required for a given address (configured by the command word), NDF is set if a NACK is not generated. When NDF is set, the I2C master automatically sends a STOP condition if MCFGR1.AUTOSTOP = 1 or the transmit buffer is not empty.	RO
9	SDF	Stop detection sign 0b0: The master did not generate a STOP condition 0b1: The master has generated a STOP condition Set when the I2C master generates a STOP condition.	RO
8	EPF	End of packet flag 0b0: The master did not generate a STOP or repeated START condition 0b1: The master has generated a STOP and a repeated START condition Asserted when the I2C master generates a repeated START condition or a STOP condition. Not asserted the first time the master generates a START condition.	RO
7:2	Reserved	-	-
1	RDF	Receive data flag 0b0: Not ready to receive data 0b1: Ready to receive data Whenever the receive buffer is full, the receive data flag RDF is set.	RO
0	TDF	Send data flag 0b0: Send data request not generated 0b1: Send data request has been generated TDF is set when the transmit buffer is empty.	RO

17.5.4 Master Interrupt Enable Register (I2C_MIER)

Offset	Bit Position																																														
0x018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0															
Reset	0x00000000																																														
Name	Reserved																			DMIE		PLTIE		FEIE		ALIE		NDIE		SDIE		EPIE	Reserved								RDIE		TDIE				
Access																				RW		RW		RW		RW		RW		RW		RW		RW		RW									RW		RW

Bit	Name	Functional Description	Access
31:15	Reserved	-	-
14	DMIE	Data match interrupt enable 0b0: Disable 0b1: Enable	RW
13	PLTIE	Pin low timeout interrupt enable 0b0: Disable 0b1: Enable	RW
12	FEIE	Buffer error interrupt enable 0b0: Disable 0b1: Enable	RW
11	ALIE	Arbitration lost interrupt enable 0b0: Disable 0b1: Enable	RW
10	NDIE	NACK detection interrupt enable 0b0: Disable 0b1: Enable	RW
9	SDIE	STOP detection interrupt enable 0b0: Disable 0b1: Enable	RW
8	EPIE	End of packet interrupt enable 0b0: Disable 0b1: Enable	RW
7:2	Reserved	-	-
1	RDIE	Receive data interrupt enable 0b0: Disable 0b1: Enable	RW
0	TDIE	Transmit data interrupt enable 0b0: Disable 0b1: Enable	RW

17.5.5 Master Status Clear Register (I2C_MSCR)

Offset	Bit Position																																								
0x01C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0									
Reset	0x00000000																																								
Name	Reserved																			DMC		PLTC		FEC		ALC		NDC		SDC		EPC		Reserved							
Access																				WO		WO		WO		WO		WO		WO		WO		WO							

Bit	Name	Functional Description	Access
31:15	Reserved	-	-
14	DMC	Data match flag cleared Write 1 to clear flag, write 0 has no effect.	WO
13	PLTC	Pin low timeout flag cleared Write 1 to clear flag, write 0 has no effect.	WO
12	FEC	Buffer error flag cleared Write 1 to clear flag, write 0 has no effect.	WO
11	ALC	Arbitration lost flag cleared Write 1 to clear flag, write 0 has no effect.	WO
10	NDC	NACK detection flag clear Write 1 to clear flag, write 0 has no effect.	WO
9	SDC	STOP detection flag clear Write 1 to clear flag, write 0 has no effect.	WO
8	EPC	End packet flag cleared Write 1 to clear flag, write 0 has no effect.	WO
7:0	Reserved	-	-

17.5.6 Master Configuration 1 Register (I2C_MCFGR1)


Note

This register can be modified only when the I2C master is disabled.

Offset	Bit Position																															
0x024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																MATCFG		RDMO	Reserved	TIMECFG		IGNACK	AUTOSTOP	Reserved						PRESCALE	
Access																	RW		RW		RW		RW	RW							RW	

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15:13	MATCFG	Match configuration Configuring the following values will cause MSR.DMF to be set. Refer to the Master Data Configuration Register (MDMR). 0b000: Match disabled 0b001: Setting prohibited 0b010: Match enabled (first data word equals MDMR.MATCH0 or MDMR.MATCH1) 0b011: Match enabled (any data word equals MDMR.MATCH0 or MDMR.MATCH1) 0b100: Match enabled (first data word equals MDMR.MATCH0 and second data word equals MDMR.MATCH1) 0b101: Match enabled (any data word equals MDMR.MATCH0 and next data word equals MDMR.MATCH1) 0b110: Match enabled (first data word logical and MDMR.MATCH1 equals MDMR.MATCH0 logical and MDMR.MATCH1) 0b111: Match enabled (any data word logical and MDMR.MATCH1 equals MDMR.MATCH0 logical and MDMR.MATCH1)	RW
12	RDMO	Receive data match only 0b0: Received data is stored in the receive buffer 0b1: Received data is discarded unless the data match flag (MSR.DMF) is set When enabled, all received data that does not cause the data match flag (MSR.DMF) to be set is discarded. When MSR.DMF is set, the RDMO configuration is ignored. When RDMO is disabled, clear RDMO before clearing MSR.DMF to ensure that no received data is lost.	RW
11	Reserved	-	-

Bit	Name	Functional Description	Access
10	TIMECFG	Timeout configuration ● 0b0: MSR.PLTF set if SCL low level duration exceeds configured timeout ● 0b1: MSR.PLTF set if SCL or SDA low level duration exceeds configured timeout TIMECFG configures the pin low-level timeout flag field (MSR.PLTF) timeout.	RW
9	IGNACK	IGNACK configuration ● 0b0: I2C master receives ACK/NACK normally ● 0b1: The I2C master treats the received NACK as an ACK When set, a received NACK field is ignored and assumed to be an ACK. The IGNACK bit needs to be set in fast-mode plus.	RW
8	AUTOSTOP	Automatic STOP generation ● 0b0: No effect ● 0b1: Automatically generate a STOP condition whenever the transmit buffer is empty and the I2C master is busy When enabled, a STOP condition is generated whenever the I2C master is busy and the transmit buffer is empty; a STOP condition can also be generated using the transmit buffer command.	RW
7:3	<i>Reserved</i>	-	-
2:0	PRESCALE	Prescaler configuration Configures the clock prescaler used for all I2C masters, except the digital glitch filter. ● 0b000: PCLK clock ● 0b001: PCLK clock divided by 2 ● 0b010: PCLK clock divided by 4 ● 0b011: PCLK clock divided by 8 ● 0b100: PCLK clock divided by 16 ● 0b101: PCLK clock divided by 32 ● 0b110: PCLK clock divided by 64 ● 0b111: PCLK clock divided by 128	RW

17.5.7 Master Configuration 2 Register (I2C_MCFGR2)

**Note**

This register can be modified only when the I2C master is disabled.

Offset	Bit Position																															
0x028	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x80800000																															
Name	FILTBPSDA	Reserved		FILTSDA		FILTBPSCL		Reserved		FILTSCL		Reserved		BUSIDLE																		
Access	RW			RW		RW				RW				RW																		

Bit	Name	Functional Description	Access
31	FILTBPSDA	Analog filter SDA bypass 0b0: SDA passes through the analog filter 0b1: SDA bypasses the analog filter	RW
30:28	Reserved	-	-
27:24	FILTSDA	Glitch filter SDA Configure the I2C master digital glitch filter for the SDA input. 1. Configuring to 0 disables the glitch filter. 2. Glitches equal to or shorter than the FILTSDA period are filtered out and ignored. 3. The delay through the glitch filter is equal to the FILTSDA period, and it must be configured to be shorter than the minimum SCL low or high period. 4. The glitch filter cycle count is not affected by the PRESCALE configuration and is automatically bypassed in high-speed mode.	RW
23	FILTBPSCL	Analog filter SCL bypass 0b0: SCL passes through the analog filter 0b1: SCL bypasses the analog filter	RW
22:20	Reserved	-	-
19:16	FILTSCCL	Glitch filter SCL Configure the I2C master digital glitch filter for the SCL input. 1. Configuring to 0 disables the glitch filter. 2. Glitches equal to or shorter than the FILTSCCL period are filtered out and ignored. The FILTSCCL period is based on the function clock. 3. The delay through the glitch filter is equal to the FILTSCCL period and must be configured to be shorter than the minimum SCL low or high period. 4. The glitch filter cycle count is not affected by the PRESCALE configuration and is automatically bypassed in high-speed mode.	RW
15:12	Reserved	-	-

Bit	Name	Functional Description	Access
11:0	BUSIDLE	Bus idle timeout Configure the bus idle timeout period in clock cycles. 1. If the high level of both SCL and SDA persists for longer than the BUSIDLE period, the I2C bus is considered idle, and the master can generate a START condition. 2. When the bus idle timeout is set to zero, the bus idle timeout is disabled.	RW

17.5.8 Master Configuration 3 Register (I2C_MCFGR3)

**Note**

This register can be modified only when the I2C master is disabled.

Offset	Bit Position																															
0x02C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved												PINLOW												Reserved							
Access													RW																			

Bit	Name	Functional Description	Access
31:20	Reserved	-	-
19:8	PINLOW	Pin low timeout Configure the pin low timeout flag in clock cycles. 1. If the low level of either SCL or SCL/SDA persists for longer than (PINLOW * 256) cycles, the MSR.PLTF flag is set. 2. When the pin low timeout is set to zero, the pin low timeout function is disabled.	RW
7:0	Reserved	-	-

17.5.9 Master Clock Configuration Register 0 (I2C_MCCR0)



Note

This register cannot be modified when the I2C master is enabled and is used for standard, fast, fast mode plus, and ultra-fast transmission.

Offset	Bit Position																																					
0x030	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved		DATAVD						Reserved		SETHOLD						Reserved		CLKHI						Reserved		CLKLO											
Access			RW								RW								RW								RW											

Bit	Name	Functional Description	Access
31:30	Reserved	-	-
29:24	DATAVD	Data valid delay Used as the minimum number of cycles (minus 1) for SDA data hold time. The configured time must be shorter than the minimum SCL low period.	RW
23:22	Reserved	-	-
21:16	SETHOLD	Setup hold delay The minimum number of cycles (minus 1) used by the master under the following conditions: 1. Hold time for the START condition 2. Setup and hold time for the repeated START condition 3. Setup time for the STOP condition The setup time will be extended by the time required to detect the rising edge on the external SCL pin. Any additional board delay caused by external load is ignored, and this time is equal to $(2 + \text{FILTSCL})/2^{\text{PRESCALE}}$ cycles.	RW
15:14	Reserved	-	-
13:8	CLKHI	Clock high period The minimum number of cycles (minus 1) for which the SCL clock is driven high by the master. The SCL high time is extended by the time required to detect the rising edge on the external SCL pin. Any additional board delay caused by external load is ignored, and this time is equal to $(2 + \text{FILTSCL})/2^{\text{PRESCALE}}$ cycles.	RW
7:6	Reserved	-	-

Bit	Name	Functional Description	Access
5:0	CLKLO	Clock low period The minimum number of cycles (minus 1) for which the SCL clock is driven low by the master. The clock low period value is also used for the minimum bus idle time between the STOP and START conditions; this is extended by the time required to detect the rising edge on the external SCL pin. Any additional board delay caused by external load is ignored, and this time is equal to $(2 + \text{FILTSC})/2^{\text{PRESCALE}}$ cycles.	RW

17.5.10 Master Clock Configuration Register 1 (I2C_MCCR1)



Note

When the I2C master is enabled and used for high-speed mode transmission, this register cannot be modified. The separate clock configuration for high-speed mode allows arbitration to occur in fast mode (timing configured by the Master Clock Configuration 0 register, MCCR0) before switching to high-speed mode (timing configured by MCCR1).

Offset	Bit Position																															
0x034	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved	DATAVD						Reserved	SETHOLD						Reserved	CLKHI						Reserved	CLKLO									
Access		RW							RW							RW							RW									

Bit	Name	Functional Description	Access
31:30	Reserved	-	-
29:24	DATAVD	Data valid delay Used as the minimum number of cycles (minus 1) for SDA data hold time. The configured time must be shorter than the minimum SCL low period.	RW
23:22	Reserved	-	-
21:16	SETHOLD	Setup hold delay The minimum number of cycles (minus 1) used by the master under the following conditions: 1. Hold time for the START condition 2. Setup and hold time for the repeated START condition 3. Setup time for the STOP condition The setup time will be extended by the time required to detect the rising edge on the external SCL pin. Any additional board delay caused by external load is ignored, and this time is equal to $(2 + \text{FILTSCl})/2^{\text{PRESCALE}}$ cycles.	RW
15:14	Reserved	-	-
13:8	CLKHI	Clock high period The minimum number of cycles (minus 1) for which the SCL clock is driven high by the master. The SCL high time is extended by the time required to detect the rising edge on the external SCL pin. Any additional board delay caused by external load is ignored, and this time is equal to $(2 + \text{FILTSCl})/2^{\text{PRESCALE}}$ cycles.	RW
7:6	Reserved	-	-

Bit	Name	Functional Description	Access
5:0	CLKLO	Clock low period The minimum number of cycles (minus 1) for which the SCL clock is driven low by the master. The clock low period value is also used for the minimum bus idle time between the STOP and START conditions; this is extended by the time required to detect the rising edge on the external SCL pin. Any additional board delay caused by external load is ignored, and this time is equal to $(2 + \text{FILTSCLE})/2^{\text{PRESCALE}}$ cycles.	RW

17.5.11 Master Transmit Data Register (I2C_MTDR)

Offset	Bit Position																															
0x038	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				CMD		DATA									
Access																					WO		WO									

Bit	Name	Functional Description	Access
31:11	Reserved	-	-
10:8	CMD	Command data ● 0b000: Send DATA[7:0] ● 0b001: Receive (DATA[7:0] + 1) bytes ● 0b010: Generate STOP condition ● 0b011: Receive and discard (DATA[7:0] + 1) bytes ● 0b100: Generate (repeated) START and send address in DATA[7:0] ● 0b101: Generate (repeated) START and send address in DATA[7:0]. This transmission expects to return NACK ● 0b110: Generate (repeated) START and send address in DATA[7:0] in high-speed mode ● 0b111: Generate (repeated) START and send address in DATA[7:0] in high-speed mode. This transmission expects to return NACK	WO
7:0	DATA	Transmit data Perform 8-bit write to data with zero extension of the CMD field	WO

17.5.12 Master Receive Data Register (I2C_MRDR)

**Note**

Reading the Receive Data Register returns the data that the I2C master has received but has not yet discarded.

Offset	Bit Position																																
0x03C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00004000																																
Name	Reserved																		RxEMPTY	Reserved							DATA						
Access																			RO								RO						

Bit	Name	Functional Description	Access
31:15	Reserved	-	-
14	RxEMPTY	RX empty status RxEMPTY indicates the empty status of the master receive data buffer 0b0: The receive buffer is not empty 0b1: The receive buffer is empty	RO
13:8	Reserved	-	-
7:0	DATA	Receiving data Received data can be discarded due to the CMD field, or the master can be configured to discard unmatched data.	RO

17.5.13 Master Data Configuration Register (I2C_MDMR)


Note

This register can only be modified when the I2C master is disabled or idle.

Offset	Bit Position																															
0x040	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved								MATCH1								Reserved								MATCH0							
Access									RW																RW							

Bit	Name	Functional Description	Access
31:24	Reserved	-	-
23:16	MATCH1	Match value 1 When enabled, the received data is compared with the match value.	RW
15:8	Reserved	-	-
7:0	MATCH0	Match value 0 When enabled, the received data is compared with the match value.	RW

17.5.14 Slave Control Register 0 (I2C_SCR0)

Offset	Bit Position																																	
0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved																																	
Access																															RW	RST	RW	SEN

Bit	Name	Functional Description	Access
31:2	Reserved	-	-
1	RST	Software reset 0b0: Slave mode logic not reset 0b1: Slave mode logic reset The reset takes effect immediately and remains set until cleared by software. No minimum delay is required before clearing the software reset.	RW
0	SEN	Enable I2C slave mode 0b0: Disable slave mode 0b1: Enable slave mode	RW

17.5.15 Slave Status Register (I2C_SSR)

Offset	Bit Position																																					
0x114	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x00000000																																					
Name	Reserved						BBF	SBF	Reserved					SARF	GCF	AM1F	AM0F	FEF	BEF	SDF	RSF		Reserved					TAF	AVF	RDF	TDF							
Access							RO	RO						RO	RO	RO	AM1F	RO	AM0F	RO	FEF	RO	BEF	RO	SDF	RO	RSF	RO			RO	TAF	RO	AVF	RO	RDF	RO	TDF

Bit	Name	Functional Description	Access
13	AM1F	Address match 1 flag Indicates that the received address matches the ADDR1 field or the ADDR0 to ADDR1 range configured in the Slave Address Register (SADDR). 0b0: No address matching ADDR1 or ADDR0/ADDR1 range has been received 0b1: Received an address that matches ADDR1.ADDR0/ADDR1 range  Note - The address match 1 flag can be cleared by reading the Slave Address Status Register (SASR). - The address match 1 flag cannot generate an asynchronous wake-up. When SCR1.ADDRMODE is configured to 0x6 or 0x7, the address matching uses the ADDR0/ADDR1 range. If ADDR0 > ADDR1, AM1F will be 0.	RO
12	AM0F	Address match 0 flag Indicates that the received address matches the ADDR0 field configured in SCR1.ADDRMODE. 0b0: No ADDR0 matching address has been received 0b1: Received an address that matches ADDR0  Note - The AM0F flag can be cleared by reading the Slave Address Status Register (SASR). - The AM0F flag cannot generate an asynchronous wake-up.	RO
11	FEF	Buffer error flag FEF can only be set when clock stretching is disabled. 0b0: No buffer underflow or overflow has been detected 0b1: Buffer underflow or overflow has been detected	RO
10	BEF	Bit error flag 0b0: Slave has not detected a bit error 0b1: Slave has detected a bit error  Note If the I2C slave sends a logical 1 but detects a logical 0 on the I2C bus, the Bit Error Flag (BEF) is set. The slave will ignore the rest of the transmission until the next (repeated) START condition.	RO
9	SDF	STOP detection flag 0b0: Slave has not detected STOP 0b1: Slave has detected STOP The SDF is set when the I2C slave detects a STOP condition and the I2C slave matches the last address byte.	RO

Bit	Name	Functional Description	Access
8	RSF	Repeated START flag (RSF) 0b0: Slave has not detected a repeated START condition 0b1: Slave has detected a repeated START condition The RSF (Repeated START Flag) is set when the I2C slave detects a repeated START condition and the I2C slave matches the last address byte. The RSF is not set when the slave detects the first START condition.	RO
7:4	Reserved	-	-
3	TAF	Send ACK flag 0b0: No ACK/NACK is required to be sent 0b1: ACK/NACK is required to be sent The TAF (Transmit ACK Flag) is cleared by writing to the Slave Transmit ACK Register (STAR).	RO
2	AVF	Address valid flag (AVF) 0b0: Address status register disabled 0b1: Address status register enabled The address valid flag AVF is cleared by reading the Slave Address Status Register (SASR). When SCR1.RxCFG = 1, SSR.AVF is also cleared by reading the Slave Receive Data Register (SRDR).	RO
1	RDF	Receive data flag 0b0: Receive data not ready 0b1: Ready to receive data The receive data flag RDF is cleared by reading the Slave Receive Data Register (SRDR). When SCR1.RxCFG = 1, if SSR.AVF is set, RDF will not be cleared when reading the Slave Receive Data Register (SRDR).	RO
0	TDF	Send data flag 0b0: No request to send data 0b1: Request to send data The transmit data flag TDF is cleared by writing to the Slave Transmit Data Register (STDR). When SCR1.TxCFG = 0, TDF is also cleared if a NACK or repeated START or STOP condition is detected.	RO

17.5.16 Slave Interrupt Enable Register (I2C_SIER)

Offset	Bit Position																																													
0x118	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0														
Reset	0x00000000																																													
Name	Reserved																	SARIE		GCIE		AM1IE		AM0IE		FEIE		BEIE		SDIE		RSIE	Reserved					TAIE		AVIE		RDIE		TDIE		
Access																		RW		RW		RW		RW		RW		RW		RW		RW		RW						RW		RW		RW		RW

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15	SARIE	SMBus alarm response interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
14	GCIE	Broadcast interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
13	AM1IE	Address match 1 interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
12	AM0IE	Address match 0 interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
11	FEIE	Buffer error interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
10	BEIE	Bit error interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
9	SDIE	STOP detection interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
8	RSIE	Repeat START interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
7:4	Reserved	-	-
3	TAIE	Send ACK interrupt enable ● 0b0: Disable ● 0b1: Enable	RW

Bit	Name	Functional Description	Access
2	AVIE	Address valid interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
1	RDIE	Receive data interrupt enable ● 0b0: Disable ● 0b1: Enable	RW
0	TDIE	Transmit data interrupt enable ● 0b0: Disable ● 0b1: Enable	RW

17.5.17 Slave Status Clear Register (I2C_SSCR)

Offset	Bit Position																																			
0x11C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
Reset	0x00000000																																			
Name	Reserved																				FEC		BEC		SDC		RSC		Reserved							
Access																					WO		WO		WO		WO									

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11	FEC	Buffer error flag cleared Write 1 to clear flag, write 0 has no effect.	WO
10	BEC	Bit error flag clear Write 1 to clear flag, write 0 has no effect.	WO
9	SDC	STOP detection flag clear Write 1 to clear flag, write 0 has no effect.	WO
8	RSC	Repeat START flag clear Write 1 to clear flag, write 0 has no effect.	WO
7:0	Reserved	-	-

17.5.18 Slave Control 1 Register (I2C_SCR1)

**Note**

This register can be modified only when the I2C slave is disabled.

Offset	Bit Position																																
0x120	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved														ADDRMODE		Reserved		HSMEN	IGNACK	RxCFG	TxCFG	SAEN	GCEN	Reserved					ACKSTALL	TxDSTALL	RXSTALL	ADRSTALL
Access															RW				RW	RW	RW	RW	RW	RW						RW	RW	RW	RW

Bit	Name	Functional Description	Access
31:19	Reserved	-	-
18:16	ADDRMODE	Address mode Select the condition that causes an address match 0b000: Address match 0 (7-bit) 0b001: Address match 0 (10-bit) 0b010: Address match 0 (7-bit) or Address match 1 (7-bit) 0b011: Address match 0 (10-bit) or Address match 1 (10-bit) 0b100: Address match 0 (7-bit) or Address match 1 (10-bit) 0b101: Address match 0 (10-bit) or Address match 1 (7-bit) 0b110: From address match 0 (7-bit) to address match 1 (7-bit) 0b111: From address match 0 (10-bit) to address match 1 (10-bit)	RW
15:14	Reserved	-	-
13	HSMEN	High-speed mode enable 0b0: Disable detection of HS-mode master code 0b1: Enable detection of HS-mode master code Detection of a high-speed mode master code at slave address 0000_1XX, but does not result in an address match on that code. When set and any HS-mode master code is detected, the SCR2.FILTEN and SCR1.ACKSTALL bits are ignored until the next STOP condition is detected.	RW
12	IGNACK	Ignore NACK 0b0: The slave ends the transmission when NACK is detected 0b1: The slave does not end the transmission when NACK is detected When Ignore NACK is set, the I2C slave continues to transmit after detecting NACK. The Ignore NACK bit needs to be set in fast plus mode.	RW
11	RXCFCG	Receive data configuration 0b0: Read the slave receive data register to return the received data and clear the receive data flag (SSR.RDF)	RW

Bit	Name	Functional Description	Access
11	RXCFG	0b1: Read the slave receive data register when the address valid flag (SSR.AVF) is set, return the slave address status register and clear the address valid flag. Read the slave receive data register when the address valid flag is cleared, return the received data and clear the receive data flag (SSR.RDF)	RW
10	TXCFG	Transmit flag configuration 0b0: The Transmit Data Flag is set only when the Slave Transmit Data Register is empty during a slave transmit 0b1: The Transmit Data Flag is set whenever the Slave Transmit Data Register is empty The Transmit Data Flag is always set before a NACK is detected at the end of a slave transmit. This may cause an extra word to be written to the Transmit Data Buffer. - When TxCFG = 0, the Slave Transmit Data Register is automatically cleared when a slave transmit is detected. This causes the Transmit Data Flag to be set whenever a slave transmit is detected and causes the Transmit Data Flag to flip at the end of a slave transmit. - When TxCFG = 1, the Transmit Data Flag is set whenever the Slave Transmit Data Register is empty and flips when the Transmit Data Register is full. This allows the Transmit Data Register to be filled before a slave transmit transfer is detected, but can cause the Transmit Data Register to be written before a NACK is detected on the last byte of a slave transmit transfer.	RW
9	SAEN	SMBus alert enable 0b0: Disable SMBus alert match 0b1: Enable SMBus alert match	RW
8	GCEN	Broadcast enabled 0b0: Broadcast address disabled 0b1: Broadcast address enabled	RW
7:4	Reserved	-	-
3	ACKSTALL	ACK SCL pause 0b0: Clock stretching disabled 0b1: Clock stretching enabled Enable SCL clock stretching during the slave transmit address byte and the slave receive address and data bytes so that software can write to the Slave Transmit ACK Register (STAR) before sending an ACK or NACK. Clock stretching occurs when the 9th bit is transmitted, so it is not compatible with high-speed mode. When ACKSTALL is enabled, it is not necessary to set SCR1.RxSTALL or SCR1.ADRSTALL.	RW
2	TxDSTALL	TX Data SCL pause 0b0: Clock stretching disabled 0b1: Clock stretching enabled When SSR.TDF = 1, SCL clock stretching is enabled during slave transmit operations. Clock stretching occurs after the 9th bit, making it compatible with high-speed mode.	RW

Bit	Name	Functional Description	Access
1	RXSTALL	RX SCL pause ● 0b0: Clock stretching disabled ● 0b1: Clock stretching enabled When SSR.RDF = 1, SCL clock stretching is enabled during slave receive operations. Clock stretching occurs after the 9th bit, making it compatible with High-Speed mode.	RW
0	ADRSTALL	Address SCL pause ● 0b0: Clock stretching disabled ● 0b1: Clock stretching enabled When SSR.AVF = 1, SCL clock stretching is enabled. Clock stretching occurs only after the 9th bit, making it compatible with High-Speed mode.	RW

17.5.19 Slave Control 2 Register (I2C_SCR2)

**Note**

This register can be modified only when the I2C slave is disabled.

Offset	Bit Position																																	
0x124	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000000																																	
Name	Reserved								FILTSDA				FILTSCS				Reserved	FILTEN	DATAVD								Reserved				CLKHOLD			
Access									RW				RW					RW	RW												RW			

Bit	Name	Functional Description	Access
31:24	Reserved	-	-
23:20	FILTSDA	Glitch filter SDA Configures the I2C slave digital glitch filter for the SDA input. 1. Configuring to 0 disables the glitch filter. 2. Glitches equal to or shorter than the FILTSDA period are filtered out and ignored. 3. The delay introduced by the glitch filter is equal to FILTSDA+2 cycles and must be configured to be less than the minimum SCL low or high period. 4. The glitch filter cycle count is not affected by the PRESCALE configuration and is disabled in high-speed mode.	RW
19:16	FILTSCS	Glitch filter SCL Configures the I2C slave digital glitch filter for the SCL input. 1. Configuring to 0 disables the glitch filter. 2. Glitches equal to or shorter than the FILTSCS period are filtered out and ignored. 3. The delay introduced by the glitch filter is equal to FILTSCS+2 cycles and must be configured to be less than the minimum SCL low or high period. 4. The glitch filter cycle count is not affected by the PRESCALE configuration and is disabled in high-speed mode.	RW
15	Reserved	-	-
14	FILTEN	Filter enable 0b0: Disables the digital filter and output delay counter in slave mode 0b1: Enables the digital filter and output delay counter in slave mode FILTEN should only be updated when the I2C slave is disabled.	RW

Bit	Name	Functional Description	Access
13:8	DATAVD	Data valid delay Configures the SDA data valid delay time for the I2C slave, which equals $FILTSCL + DATAVD + 2$ cycles. 1. The data valid delay must be configured to be less than the minimum SCL low period. 2. The I2C slave data valid delay is not affected by the PRESCALE configuration and is disabled in high-speed mode.	RW
7:4	<i>Reserved</i>	-	-
3:0	CLKHOLD	Clock hold time Configures the minimum clock hold time for the I2C slave when clock stretching is enabled. 1. The minimum hold time is equal to $CLKHOLD + 3$ cycles. 2. The I2C slave clock hold time is not affected by the PRESCALE configuration and is disabled in high-speed mode.	RW

17.5.20 Slave Transmit ACK Register (I2C_STAR)

Offset	Bit Position																															
0x128	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															TXNACK
Access																																RW

Bit	Name	Functional Description	Access
31:1	Reserved	-	-
0	TXNACK	Transmit NACK 0b0: Write ACK for each received word 0b1: Write NACK for each received word After receiving each word, software can send ACK (logic 0) or NACK (logic 1); sending NACK chooses whether to use ACK or NACK. - When SCR1.ACKSTALL is set, a NACK must be written for each matching address byte and each received word. ACKSTALL must be set because it pauses data transmission until software reads the received word (and decides whether to respond with ACK or NACK). - To configure the default ACK/NACK, NACK can also be written when the I2C slave is disabled or idle.	RW

**Note**

The STAR register can only be written when the ACK SCL pause position bit (SCR1.ACKSTALL) is set. The SCR1.ACKSTALL bit enables clock stretching during the ACK/NACK period, during which software can write to the STAR register. This logic ensures that the clock stretching lasts at least 1 bus clock cycle after the STAR register is updated. The clock hold time field (SCR2.CLKHOLD) can be used to further extend the clock stretching duration.

17.5.21 Slave Transmit Data Register (I2C_STDR)

Offset	Bit Position																															
0x12C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																								DATA							
Access																									WO							

Bit	Name	Functional Description	Access
31:8	Reserved	-	-
7:0	DATA	Transmit data Writing to the Slave Transmit Data Register (STDR) stores the I2C slave transmit data in the slave transmit data register.	WO

**Note**

Clock stretching (enabled or disabled) affects the transmission timing of data. The SCR1.TXDSTALL bit enables clock stretching during the first data bit of the slave transmit transfer.

1. If clock stretching is enabled (SCR1.TXDSTALL=1), the data transmission is paused until the STDR is updated. After the STDR is updated, the clock stretching is extended for at least 1 bus clock cycle, and the clock hold time field (SCR2.CLKHOLD) can be used to extend the duration.
2. If clock stretching is disabled (SCR1.TXDSTALL=0), data should be written to the slave transmit register before the slave transmission begins; otherwise, if data is not written before the transmission starts, the buffer error flag (SSR.FEF) will be set.

17.5.22 Slave Receive Data Register (I2C_SRDR)

Offset	Bit Position																																	
0x130	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00004000																																	
Name	Reserved																		Reserved								DATA							
																	RO	SOF	REMP															
Access																	RO	RO									RO							

Bit	Name	Functional Description	Access
31:16	Reserved	-	-
15	SOF	Frame start Indicates whether this is the first data word. 0b0: This is not the first data word since a (repeated) START or STOP condition 0b1: This is the first data word since a (repeated) START or STOP condition	RO
14	REMP	RX empty Indicates the empty state of the slave receive data register. 0b0: The receive data register is not empty 0b1: The receive data register is empty	RO
13:8	Reserved	-	-
7:0	DATA	Receiving data I2C slave received data	RO

17.5.23 Slave Address Register (I2C_SADDR)

**Note**

This register can be modified only when the I2C slave is disabled.

Offset	Bit Position																															
0x140	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved				ADDR1								Reserved				ADDR0								Reserved							
Access					RW												RW															

Bit	Name	Functional Description	Access
31:27	Reserved	-	-
26:17	ADDR1	Address value 1 Compares with the received address to detect the slave address. - In 10-bit mode, the first address byte is compared with {11110, ADDR1[26:25]}, and the second address byte is compared with ADDR1[24:17]. - In 7-bit mode, the address is compared with ADDR1[23:17].	RW
16:11	Reserved	-	-
10:1	ADDR0	Address value 0 Compares with the received address to detect the slave address. - In 10-bit mode, the first address byte is compared with {11110, ADDR0[10:9]}, and the second address byte is compared with ADDR0[8:1]. - In 7-bit mode, the address is compared with ADDR0[7:1].	RW
0	Reserved	-	-

17.5.24 Slave Address Status Register (I2C_SASR)

Offset	Bit Position																																		
0x150	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reset	0x00004000																																		
Name	Reserved																		ANV	Reserved	RADDR														
Access																			RO		RO														

Bit	Name	Functional Description	Access
31:15	Reserved	-	-
14	ANV	Address disabled 0b0: Receive address (RADDR) enabled 0b1: Receive address (RADDR) disabled	RO
13:11	Reserved	-	-
10:0	RADDR	Received address Each time the AMF is set, the received address is updated; the AMF can be cleared by reading the slave address status register. - In 7-bit mode, the address byte is stored in RADDR[7:0]. - In 10-bit mode, the first address byte is {11110, RADDR[10:9], RADDR[0]}, and the second address byte is RADDR[8:1]. Thus, the R/W bit is always stored in RADDR[0].	RO

18 ADC

18.1 Introduction

External analog signals need to be converted into digital signals to be further processed by the MCU. This product integrates an 12-bit high-precision, high-speed successive approximation register (SAR ADC) module.

18.2 Features

- 12-bit conversion accuracy
- 1Msps conversion speed
- Up to 10 input channels are supported, including 8 external pin inputs, 2 internal input channels (1/3 AVCC voltage, VCAP)
- 2 reference sources: AVCC voltage, EXVREF pins
- ADC voltage input range: 0-Vref
- Support SQR scan conversion
- Software can configure ADC conversion rate
- Support on-chip peripherals to automatically trigger ADC conversion, effectively reducing chip power consumption and improving real-time conversion

18.3 Functional Description

18.3.1 ADC Block Diagram

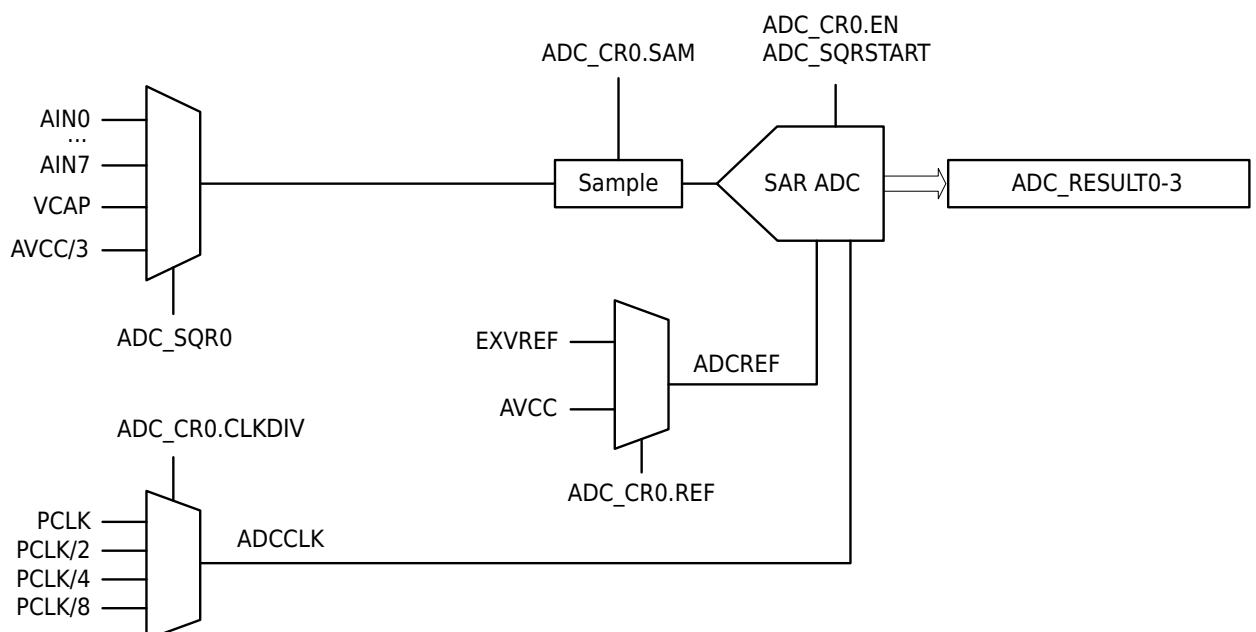


Figure 18-1 ADC Block Diagram

18.3.2 Conversion Timing and Conversion Speed

The ADC conversion timing is shown in the figure below: a complete ADC conversion consists of a sampling process and a successive comparison process. The sampling process requires 2-127 ADCCLK

cycles, configured by ADC_CR0.SAM; the successive comparison process requires 12 ADCCLK cycles. Therefore, an ADC conversion requires a total of 14-139 ADCCLK cycles.

The ADC conversion speed is measured in sps (samples per second), indicating how many ADC conversions are performed per second. The ADC conversion speed is: the frequency of ADCCLK/the number of ADCCLKs required for one ADC conversion.

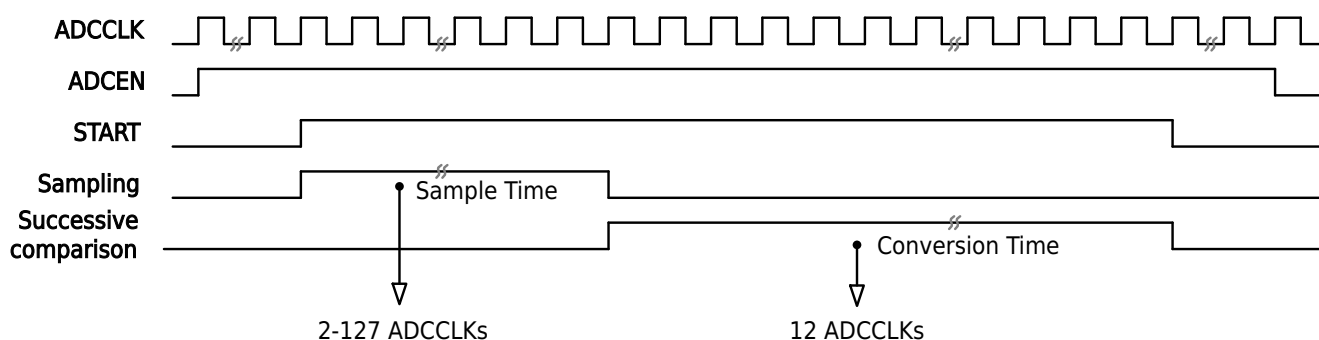


Figure 18-2 ADC Conversion Timing Diagram

ADC conversion speed is related to the ADC reference voltage and AVCC voltage. For more information, see the "Electrical Specifications" chapter in "Datasheet".

18.3.3 Scan Conversion Mode

SQR Scan Conversion Mode

SQR scan conversion mode supports up to 4 consecutive conversions. The total number of conversions is configured by ADC_SQR1.CNT, and all channels can be configured for conversion through ADC_SQR0.CHxMUX. This mode can be initiated either by setting the ADC_SQRSTART.START bit or by setting the external trigger of ADC_EXTTRIGGER. After starting the conversion, the ADC module converts the channels configured in CHxMUX-CH0MUX in sequence until the total number of conversions is completed. After the ADC module completes the total number of conversions, the ADC_IFR.SQRIF bit will be automatically set to 1, and the conversion result will be saved in the ADC_SQRRESULTx-ADC_SQRRESULT0 register corresponding to the conversion channel. The following diagram illustrates the SQR scan conversion process for performing 4 conversions on AIN0, AIN5, AIN1, and AIN2. In this configuration, SQR scan conversion channel 3 is set to AIN0, channel 2 to AIN5, channel 1 to AIN1, and channel 0 to AIN2. After ADC_SQRSTART.START is set to 1, the ADC module will sequentially convert SQR scan channels 3 to 0.

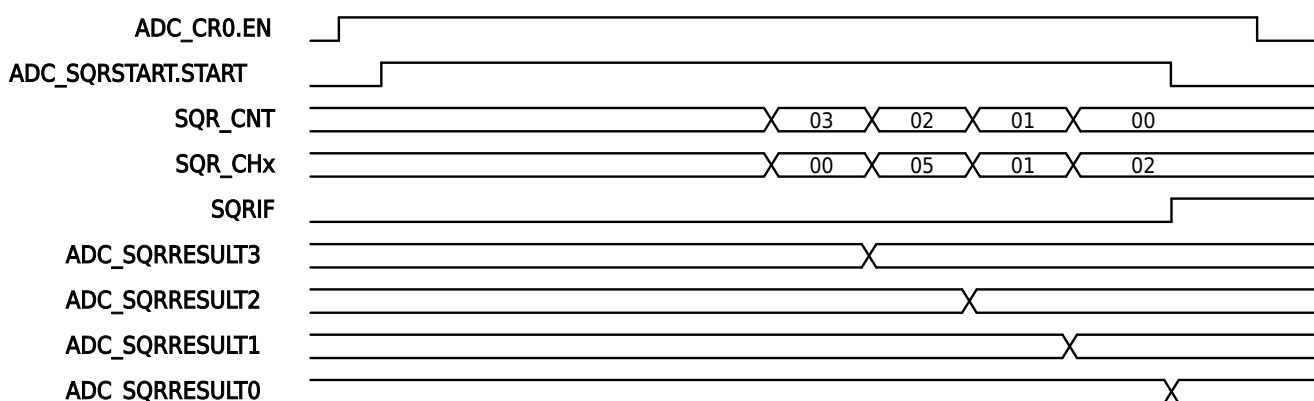


Figure 18-3 Examples of ADC SQR Scan Conversion Process

18.3.4 ADC Conversion External Trigger Source

ADC conversions can be initiated either by software configuration or by an external trigger.

The ADC_EXTTRIGGER registers can be configured to set the external trigger source for ADC conversions.

For more information, see the "[ADC SQR Scan Conversion External Interrupt Trigger Source Configuration Register \(ADC_EXTTRIGGER\)](#)".

18.3.5 ADC Interrupt

The ADC interrupt request is shown in the table below:

Table 18-1 ADC Interrupt

Interrupt Source	Interrupt Flag	Interrupt Enable
ADC SQR scan conversion complete	ADC_IFR.SQRIF	ADC_CR0.IE

18.4 Operation Example

18.4.1 Start the ADC SQR Scan Conversion Operation Process through the ADC_SQRSTART.START Bit

Procedure

- Step1.** Configure the corresponding bits of PAADS, and configure the ADC channel to be converted as an analog port.
- Step2.** Set PAADS[2] to 1 to configure the ADC external reference voltage pin as an analog port.



Note

If the ADC reference voltage does not select an external reference voltage pin, this step can be skipped.

- Step3.** Set ADC_CR0.EN to 1 to enable the ADC module.
- Step4.** Delay for the longer of $T_{ADCSTART}$ (refer to the "Electrical Characteristics/Operating Condition-ADC Characteristics" section in the "Datasheet" for specific values), and wait for the ADC module to complete startup.
- Step5.** Configure ADC_CR0.REF and select the reference voltage of the ADC.
- Step6.** Configure ADC_CR0.SAM and ADC_CR0.CLKDIV to set the conversion speed of ADC.
- Step7.** Configure ADC_SQR0.CHxMUX, and select SQR scan conversion channels.
- Step8.** Configure ADC_SQR1.CNT, and select the total number of conversions for SQR scan conversions.
The total number of conversions needs to be consistent with the number of conversion channels configured in step7.
- Step9.** Set ADC_ICR.SQRIC to 0, clear ADC_IFR.SQRIF flag.
- Step10.** Set ADC_SQRSTART.START to 1 to start ADC SQR scan conversion.
- Step11.** Wait for ADC_IFR.SQRIF to become 1, read ADC_SQRRESULTx-ADC_SQRRESULT0 registers to obtain the conversion result of the corresponding channel.
- Step12.** To convert other channels, repeat steps7-11.
- Step13.** Set ADC_CR0.EN to 0, turn off ADC module.

18.4.2 External Trigger Start ADC SQR Scan Conversion Operation Flow

Procedure

- Step1.** Configure the corresponding bits of PAADS, and configure the ADC channel to be converted as an analog port.
- Step2.** Set PAADS[2] to 1 to configure the ADC external reference voltage pin as an analog port.

**Note**

If the ADC reference voltage does not select an external reference voltage pin, this step can be skipped.

- Step3.** Set ADC_CR0.EN to 1 to enable the ADC module.
- Step4.** Delay for the longer of $T_{ADCSTART}$ (refer to the "Electrical Characteristics/Operating Condition-ADC Characteristics" section in the "Datasheet" for specific values), and wait for the ADC module to complete startup.
- Step5.** Set ADC_CR0.IE to 1 to enable ADC interrupt.
- Step6.** Enable the ADC interrupt in the NVIC interrupt vector table.
- Step7.** Configure ADC_CR0.REF and select the reference voltage of the ADC.
- Step8.** Configure ADC_CR0.SAM and ADC_CR0.CLKDIV to set the conversion speed of ADC.
- Step9.** Configure ADC_SQR0.CHxMUX, and select SQR scan conversion channels.
- Step10.** Configure ADC_SQR1.CNT, and select the total number of conversions for SQR scan conversions.
The total number of conversions needs to be consistent with the number of conversion channels configured in step9.
- Step11.** Set ADC_ICR to 0x0, clear the ADC interrupt flag.
- Step12.** Configure ADC_EXTTRIGGER to select the external trigger condition.
- Step13.** When an external trigger condition triggers the ADC to complete the conversion, the ADC module will generate an interrupt. Users can read ADC_SQRRESULTx-ADC_SQRRESULT0 registers in the ADC interrupt service routine to get the conversion result of the corresponding channel.
- Step14.** To convert other channels, repeat steps9-13.
- Step15.** Set ADC_CR0.EN to 0, turn off ADC module.

18.5 Register Description

18.5.1 Register List

Base address: 0x40002400

Table 18-2 Analog-to-Digital Converter (ADC) Register Offset

Offset	Register	Description	Width
0x004	ADC_CR0	ADC Basic Configuration Register 0	32
0x040	ADC_SQR0	ADC SQR Scan Conversion Channel Configuration Register 0	32
0x048	ADC_SQR1	ADC SQR Scan Conversion Channel Configuration Register 1	32
0x050	ADC_SQRRESULT0	ADC SQR Scan Conversion Channel Configuration Register 0	32
0x054	ADC_SQRRESULT1	ADC SQR Scan Conversion Channel Configuration Register 1	32
0x058	ADC_SQRRESULT2	ADC SQR Scan Conversion Channel Configuration Register 2	32
0x05C	ADC_SQRRESULT3	ADC SQR Scan Conversion Channel Configuration Register 3	32
0x0B0	ADC_IFR	ADC Interrupt Flag Register	32
0x0B4	ADC_ICR	ADC Interrupt Clear Register	32
0x0B8	ADC_EXTTRIGGER	ADC SQR Scan Conversion External Interrupt Trigger Source Configuration Register	32
0x0C4	ADC_SQRSTART	ADC SQR Scan Conversion Start Control Register	32

18.5.2 ADC Basic Configuration Register 0 (ADC_CR0)

Offset	Bit Position																															
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00042000																															
Name	Reserved												SAM		Reserved	IE	Reserved	IBSEL		Reserved	REF	Reserved				CLKDIV		Reserved	EN			
Access													RW			RW		RW			RW						RW			RW		

Bit	Name	Functional Description	Access
31:21	Reserved	-	-
20:17	SAM	ADC sampling period selection 0b0000: 2 ADCCLK cycles 0b0001: 3 ADCCLK cycles 0b0010: 4 ADCCLK cycles 0b0011: 6 ADCCLK cycles 0b0100: 8 ADCCLK cycles 0b0101: 10 ADCCLK cycles 0b0110: 12 ADCCLK cycles 0b0111: 16 ADCCLK cycles 0b1000: 20 ADCCLK cycles 0b1001: 28 ADCCLK cycles 0b1010: 36 ADCCLK cycles 0b1011: 48 ADCCLK cycles 0b1100: 64 ADCCLK cycles 0b1101: 72 ADCCLK cycles 0b1110: 96 ADCCLK cycles 0b1111: 127 ADCCLK cycles	RW
16	Reserved	-	-
15	IE	ADC interrupt control 0b0: Disable 0b1: Enable	RW
14	Reserved	-	-
13:12	IBSEL	IBAS current selection 0b00: Lowest power consumption (Recommended) 0b10: Lower power consumption - Others: Setting prohibited	RW
11:10	Reserved	-	-
9	REF	ADC reference voltage selection 0b0: AVCC voltage 0b1: External reference voltage EXVREF (PA02)	RW
8:4	Reserved	-	-

Bit	Name	Functional Description	Access
3:2	CLKDIV	ADC clock selection ● 0b00: PCLK clock ● 0b01: PCLK clock divided by 2 ● 0b10: PCLK clock divided by 4 ● 0b11: PCLK clock divided by 8  Notice ● When the system clock division selected by HCLK is greater than 1, the CLKDIV division must be set to greater than 1. ● Can only be modified when ADC is idle.	RW
1	<i>Reserved</i>	-	-
0	EN	ADC enable control ● 0b0: Disable ● 0b1: Enable  Note It is recommended to disable the ADC when it is idle to avoid unexpected behavior.	RW

18.5.3 ADC SQR Scan Conversion Channel Configuration Register 0 (ADC_SQR0)

Offset	Bit Position																															
0x040	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved														CH3MUX	Reserved	CH2MUX	Reserved	CH1MUX	Reserved	CH0MUX											
Access															RW		RW		RW		RW											

Bit	Name	Functional Description	Access
31:19	Reserved	-	-
18:15	CH3MUX	SQR scan conversion channel 3 selection For more information, see CH0MUX	RW
14	Reserved	-	-
13:10	CH2MUX	SQR scan conversion channel 2 selection For more information, see CH0MUX	RW
9	Reserved	-	-
8:5	CH1MUX	SQR scan conversion channel 1 selection For more information, see CH0MUX	RW
4	Reserved	-	-
3:0	CH0MUX	SQR scan conversion channel 0 selection ● 0b0000: AIN0 (PA00) ● 0b0001: AIN1(PA02) ● 0b0010: AIN2(PA03) ● 0b0011: AIN3(PA05) ● 0b0100: AIN4(PA09) ● 0b0101: AIN5(PA10) ● 0b0110: AIN6(PA11) ● 0b0111: AIN7(PA12) ● 0b1000: VCAP(1.6V) ● 0b1001: AVCC/3 ● Others: Setting prohibited	RW

18.5.4 ADC SQR Scan Conversion Channel Configuration Register 1 (ADC_SQR1)

Offset	Bit Position																															
0x048	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																															CNT
Access																																RW

Bit	Name	Functional Description	Access
31:2	Reserved	-	-
1:0	CNT	Number of SQR scan conversions The actual number of conversions is CNT + 1, with a maximum valid setting of 0x3 (i.e., 4 conversions). Channels selected by CHxMUX-CH0MUX (x=CNT) are scanned by SQR in sequence.	RW

18.5.5 ADC SQR Scan Conversion Channel x Result Register (ADC_SQRRESULTx) (x=0-3)

Offset: 0x050 (SQRRESULT0), 0x054 (SQRRESULT1), 0x058 (SQRRESULT2), 0x05C (SQRRESULT3)

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				RESULT											
Access																					RO											

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11:0	RESULT	ADC SQR scan conversion channelx conversion result	RO

18.5.6 ADC Interrupt Flag Register (ADC_IFR)

Offset	Bit Position																																
0x0B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																															RO	SORIF
Access																																RO	

Bit	Name	Functional Description	Access
31:1	Reserved	-	-
0	SQRIF	ADC SQR scan conversion complete flag 0b0: Conversion not complete 0b1: Conversion complete	RO

18.5.7 ADC Interrupt Clear Register (ADC_ICR)

Offset	Bit Position																															
0x0B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000001																															
Name	Reserved																															WO SORIC
Access																																WO

Bit	Name	Functional Description	Access
31:1	Reserved	-	-
0	SQRIC	ADC SQR scan conversion complete flag clear Write 0 to clear flag, write 1 has no effect; read always returns 1.	WO

18.5.8 ADC SQR Scan Conversion External Interrupt Trigger Source Configuration Register (ADC_EXTTRIGGER)

Offset	Bit Position																															
0x0B8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																												EXT_TRIG_EN			
Access																													RW			

Bit	Name	Functional Description	Access
31:4	Reserved	-	-
3:0	EXT_TRIG_EN	External interrupt trigger source selection ● 0b0001: CTIM0 ADC trigger source rising edge ● 0b0010: CTIM1 ADC trigger source rising edge ● 0b0011: ATIM3 ADC trigger source rising edge ● 0b0100: CTRIM_TOG rising edge ● 0b0101: LPUART0 interrupt request ● 0b0110: VC0 interrupt request ● 0b0111: PA03 interrupt request ● 0b1000: PA07 interrupt request ● Others: No selection	RW



Note

- To enable CTIM to trigger automatic ADC conversion, the corresponding trigger enable register of CTIM must be configured. For detailed operation steps, please refer to the "[Function Description - Timer Trigger ADC](#)" section in the "Composite Timer (CTIM)".
- ATIM3 triggers ADC automatic conversion without the need to enable the corresponding interrupt. Simply configure the corresponding trigger enable register of the TIM. For detailed operation, please refer to the "[Function Description - Timer Trigger ADC](#)" section in the "Advanced Timer (ATIM3)".
- CTRIM triggers ADC automatic conversion without the need to enable the corresponding interrupt. For detailed information, please refer to the "[Low Power Timer Mode](#)" section in the "Clock Trim Module (CTRM)".
- Except for ATIM3, CTIM0/1, and CTRIM, other peripherals trigger the ADC using the rising edge of their respective interrupt flags. If repeated triggering is required, the interrupt flag needs to be cleared. If you do not need to enter the interrupt service routine, please do not enable the interrupt enable of the NVIC.

18.5.9 ADC SQR Scan Conversion Start Control Register (ADC_SQRSTART)

Offset	Bit Position																																
0x0C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																																START
Access																																	RW

Bit	Name	Functional Description	Access
31:1	Reserved	-	-
0	START	ADC SQR scan conversion start control Write 1 to start ADC SQR scan conversion; write 0 is prohibited. The hardware automatically clears this bit after the SQR scan completes. Read: 0b0: ADC is in idle state 0b1: SQR conversion in progress  Note - Write 1 is invalid when START is 1. - Writing 1 is effective only when the ADC is in idle state.	RW

19 VC

19.1 Introduction

The analog voltage comparator VC is used to compare the size of two input analog voltages, and output high/low level according to the comparison result.

- When the "+" input voltage is higher than the "-" input voltage, the voltage comparator output is high;
- When the "+" input voltage is lower than the "-" input voltage, the voltage comparator output is low.

19.2 Features

- Support 64-stage voltage divider reference
- Support 4 external inputs and 2 on-chip input signals
- Support 2 output polarities
- Support window compare function
- A variety of filter time configurations can effectively enhance the anti-interference ability of the chip
- Support 3 interrupt triggering methods: high level/rising edge/falling edge
- The output of the voltage comparator can be used as the input of the timer
- Support working in DeepSleep mode

19.3 Functional Description

19.3.1 Function Block Diagram

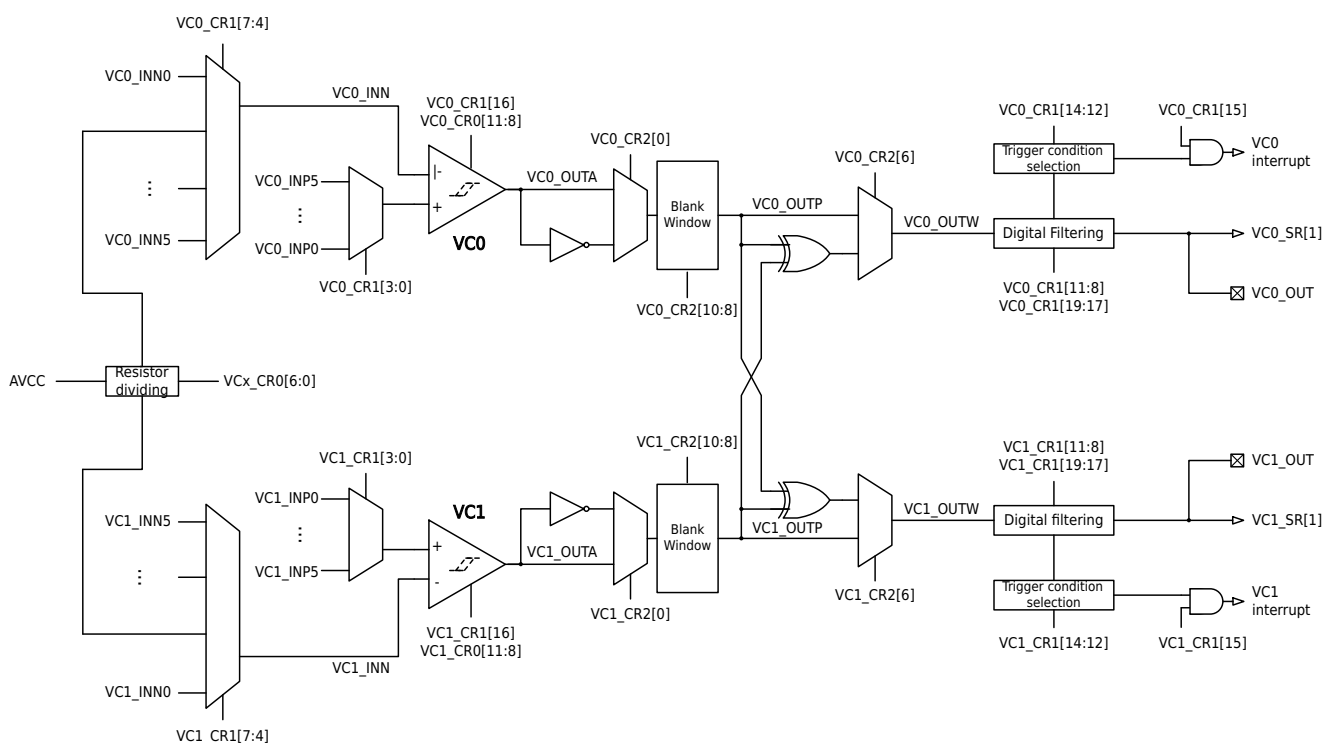


Figure 19-1 VC Function Block Diagram

19.3.2 Setup/Response Time

When using a voltage comparator, the time from when VC is enabled or when the input voltage at both ends of VC changes to when correct result is output is determined by the VCx_CR0.BIAS bit. The larger the current, the faster the VC response. For the specific time, see the description in the "Electrical Specifications-Operating Conditions-VC Features" chapter of the "Datasheet".

When the DAC is enabled (VCx_CR0.DIVEN = 0b1), it is necessary to wait for the DAC to stabilize before the comparator can output normally. For the DAC start-up time, please refer to the description in the "Electrical Specifications" chapter of the Datasheet.

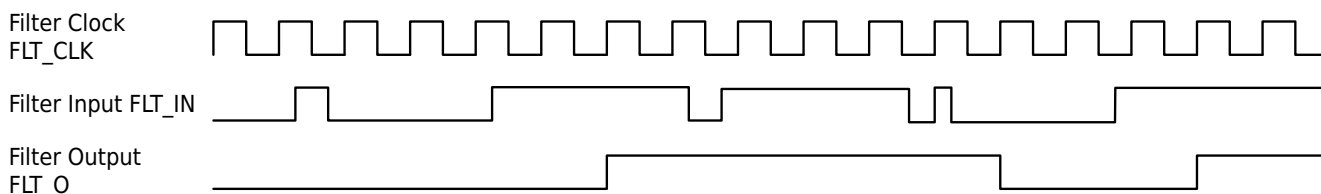
19.3.3 Digital Filtering

In addition to the inherent setup/response time of the voltage comparator, the user can set a longer filter time to filter out system noise, such as high current noise when the motor stops.

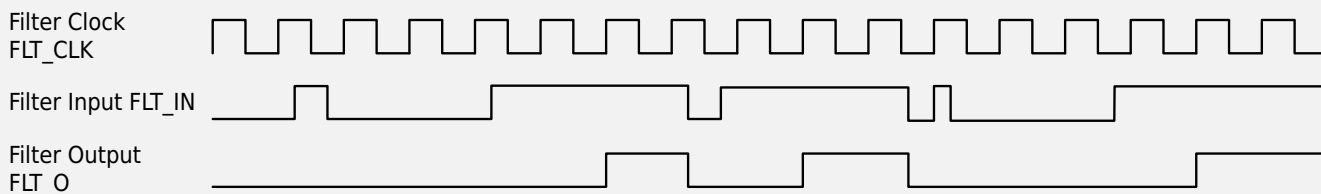
The digital filtering signal level can be read from the register VCx_SR[1]; when the GPIO function is configured as VCx_OUT, the digital filtering signal can be output from the GPIO for easy measurement.

Enable the digital filter module and select the filter time as 2 filter clock cycles, the filter illustration is as follows:

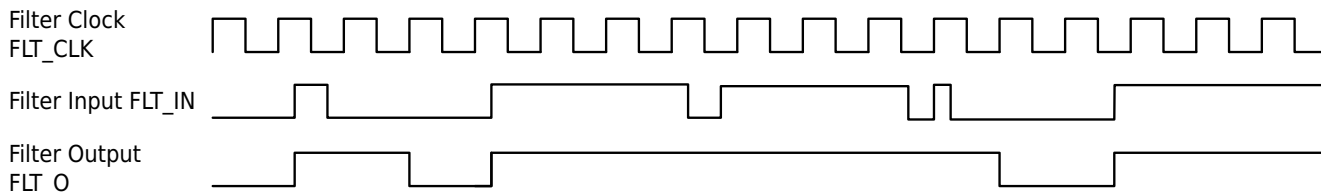
Normal Filter Mode



Peak Filter Mode



Valley Filter Mode



Spike Filter Mode

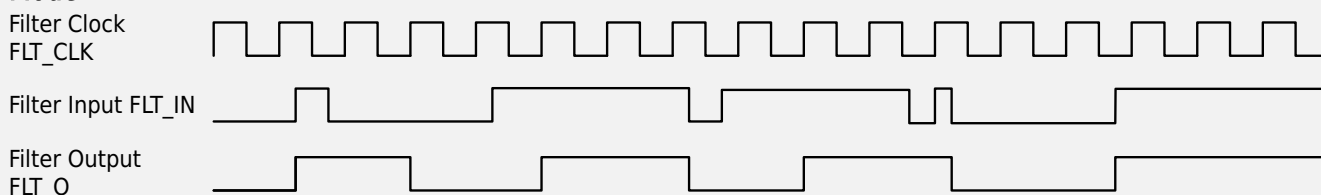


Figure 19-2 VC Filter Response Time

19.3.4 Hysteresis Function

The hysteresis function can be selected for the voltage comparator, and the diagram after the hysteresis function is enabled is as follows:

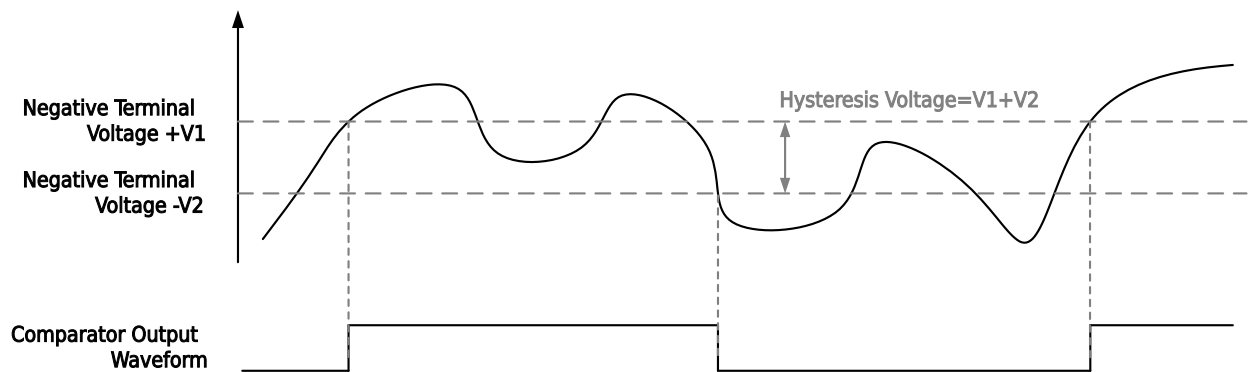


Figure 19-3 VC Hysteresis Function

19.3.5 Configuration of Comparison Result Polarity

When VCx_CR2.POL is configured as 0, the level of VCx_OUTA and VCx_OUTP is the same.

When VCx_CR2.POL is configured as 1, the level of VCx_OUTA and VCx_OUTP is opposite.

19.3.6 Window Comparison Function

When VCx_CR2.WINDOW is configured to 0, the VCx_OUTW signal is the same as the VCx_OUTP signal.

When VCx_CR2.WINDOW is configured to 1, the VCx_OUTW signal is the XOR value of VC0_OUTP and VC1_OUTP.

19.3.7 Blank Window

The PWM signals output by timer ATIM3 can be used as blank window signal to control the output of the comparator. Configure VCx_CR2.BLANK to enable the blank window function or select the corresponding PWM blank window signal.

The purpose of the blank window is to prevent abnormal VC comparison results caused by current glitches that occur at the start of the PWM cycle. Blank window function diagram is as follows:

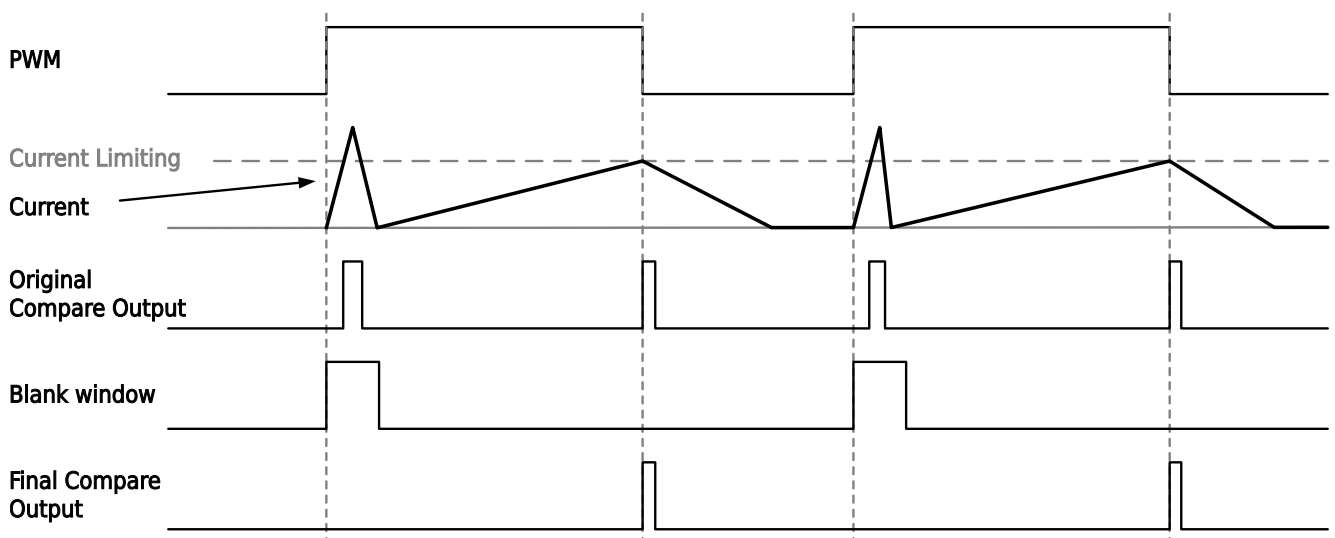


Figure 19-4 VC Blank Window

19.4 Register Description

19.4.1 Register List

Base address: 0x40002800

Table 19-1 Analog Voltage Comparator (VC) Register Offset

Offset	Register	Description	Width
0x00	VC0_CR0	VC0 Control Register 0	32
0x04	VC0_CR1	VC0 Control Register 1	32
0x08	VC0_CR2	VC0 Control Register 2	32
0x0C	VC0_SR	VC0 Status Register	32
0x10	VC1_CR0	VC1 Control Register 0	32
0x14	VC1_CR1	VC1 Control Register 1	32
0x18	VC1_CR2	VC1 Control Register 2	32
0x1C	VC1_SR	VC1 Status Register	32
0x20	VC_TRIM	VC Offset Trim Register	32

19.4.2 Control Register 0 (VCx_CR0) (x=0-1)

Offset: 0x00 (VC0_CR0), 0x10 (VC1_CR0)

Offset	Bit Position																																	
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reset	0x00000020																																	
Name	Reserved																				HYS		BIAS		Reserved	DIVEN	DIV							
Access																					RW		RW			RW								

Bit	Name	Functional Description	Access
31:12	Reserved	-	-
11:10	HYS	Hysteresis function configuration 0b00: No hysteresis 0b01: Hysteresis voltage approximately 10mV 0b10: Hysteresis voltage approximately 20mV 0b11: Hysteresis voltage approximately 30mV	RW
9:8	BIAS	Power selection 0b00: Low power consumption level 0b01: Medium and low power consumption level 0b10: Medium power consumption level 0b11: High power consumption level  Note The larger the power consumption level, the faster the response speed. For the the specifications of specific gears, see the description in "Electrical Specifications-VC Features" section of the "Datasheet".	RW
7	Reserved	-	-
6	DIVEN	Resistor voltage divider circuit enable control 0b0: Disable 0b1: Enable  Note VC0/VC1 share this bit setting.	RW
5:0	DIV	Resistor voltage divider circuit output voltage configuration 0b000000: 0/64 Vref 0b000001: 1/64 Vref ... 0b111110: 62/64 Vref 0b111111: 63/64 Vref  Note VC0/VC1 share this bit setting, Vref source is AVCC.	RW

**Note**

When VC is enabled, if you modify the VC control register in the above table, you need to wait for the T_{setup} time before the VC can work normally. For the value of T_{setup} , detailed see the "Datasheet".

19.4.3 Control Register 1 (VCx_CR1) (x=0-1)

Offset: 0x04 (VC0_CR1), 0x14 (VC1_CR1)

Offset	Bit Position																																					
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reset	0x000000FF																																					
Name	Reserved												FLT_MODE		FLTCLK		EN		IE		LEVEL		RISING		FALLING		FLTTIME				NSEL				PSEL			
Access													RW		RW		RW		RW		RW		RW		RW		RW				RW				RW			

Bit	Name	Functional Description	Access
31:20	Reserved	-	-
19:18	FLT_MODE	Filter pattern selection 0b00: Standard Filtering Mode. Filter output toggles only if input remains stable for a defined hold time after transition. 0b01: Peak Filtering Mode. Output sets to logic 1 only after the input holds logic 1 for a stable duration; sets to 0 immediately on input set to 0. The purpose of this mode is to filter out input peaks that are not long enough. 0b10: Valley Filtering Mode. Output sets to logic 0 only after input holds logic 0 for a stable duration; sets to 1 immediately on input set to 1. The purpose of this mode is to filter out input valleys that are not long enough. 0b11: Glitch Filtering Mode. Output toggles immediately on input transition, then ignores subsequent input changes for a hold time. In this mode, the filter output will closely follow the input, and at the same time, it will prevent the output signal from having glitches.	RW
17	FLTCLK	Filter clock selection 0b0: RC256K 0b1: PCLK	RW
16	EN	VC enable control 0b0: Disable 0b1: Enable	RW
15	IE	VC interrupt enable control 0b0: Disable 0b1: Enable	RW
14	LEVEL	Filter signal high level trigger interrupt enable control 0b0: Disable 0b1: Enable	RW
13	RISING	Filter signal rising edge trigger interrupt enable control 0b0: Disable 0b1: Enable	RW

Bit	Name	Functional Description	Access
12	FALLING	Filter signal falling edge trigger interrupt enable control ● 0b0: Disable ● 0b1: Enable	RW
11:8	FLTTIME	Filter time configuration N represents filter clock cycles ● 0x0: Digital filter disabled ● 0x1: N = 2 ● 0x2: N = 4 ● 0x3: N = 8 ● 0x4: N = 16 ● 0x5: N = 32 ● 0x6: N = 64 ● 0x7: N = 128 ● 0x8: N = 256 ● 0x9: N = 512 ● 0xA: N = 1024 ● 0xB: N = 2048 ● Other: N = 4096	RW
7:4	NSEL	Voltage comparator negative input signal selection ● 0b0000: VCx_INN0 (x=0, PA00; x=1, PA01) ● 0b0001: VCx_INN1 (x=0, PA02; x=1, PA03) ● 0b0010: VCx_INN2 (x=0, PA08; x=1, PA10) ● 0b0011: VCx_INN3 (x=0, PA15; x=1, PA12) ● 0b0100: VCx_INN4 (Built-in resistor voltage divider output voltage) ● 0b0101: VCx_INN5 (core voltage) ● Others: Setting prohibited	RW
3:0	PSEL	Voltage comparator positive input signal selection ● 0b0000: VCx_INP0 (x=0, PA00; x=1, PA01) ● 0b0001: VCx_INP1 (x=0, PA02; x=1, PA03) ● 0b0010: VCx_INP2 (x=0, PA08; x=1, PA10) ● 0b0011: VCx_INP3 (x=0, PA15; x=1, PA12) ● 0b0100: VCx_INP4 (Built-in resistor voltage divider output voltage) ● 0b0101: VCx_INP5 (core voltage) ● Others: Setting prohibited	RW

**Note**

When VC is enabled, if you modify NSEL/PSEL, you need to wait for T_{setup} before VC can work normally. For the value of T_{setup} , details see the "Datasheet".

19.4.4 Control Register 2 (VCx_CR2) (x=0-1)

Offset: 0x08 (VC0_CR2), 0x18 (VC1_CR2)

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																				BLANK		Reserved	WINDOW	ATIMBK	ATIM3CLR	Reserved		POL			
Access																					RW			RW	RW	RW			RW			

Bit	Name	Functional Description	Access
31:11	Reserved	-	-
10:8	BLANK	VCx blank window source selection 0b000: No blank window 0bxx1: ATIM3 OCREF0B 0bx1x: ATIM3 OCREF1B 0b1xx: ATIM3 OCREF2B	RW
7	Reserved	Reads as 0, Write as 0	-
6	WINDOW	Window comparison function configuration 0b0: Disable 0b1: Enable	RW
5	ATIMBK	VCx filtered signal output to the ATIM3 brake enable control 0b0: Disable 0b1: Enable	RW
4	ATIM3CLR	VCx filtered signal output to the ATIM3 REFCLR enable control 0b0: Disable 0b1: Enable	RW
3:1	Reserved	-	-
0	POL	VCx_OUTP signal polarity configuration 0b0: VCx_OUTP is high when the positive terminal is greater than the negative terminal 0b1: VCx_OUTP is low when the positive terminal is greater than the negative terminal	RW

19.4.5 Status Register (VCx_SR) (x=0-1)

Offset: 0x0C (VC0_SR), 0x1C (VC1_SR)

Offset	Bit Position																																
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																														FLTV		INTF
Access																															RO	RW	

Bit	Name	Functional Description	Access
31:2	Reserved	-	-
1	FLTV	Filtered signal level 0b0: Low level 0b1: High level	RO
0	INTF	Interrupt flag Write 0 to clear the flag, write 1 has no effect Read: 0b0: No interrupt occurred 0b1: Interrupt has occurred	RW

19.4.6 Offset Trim Register (VC_TRIM)

Offset	Bit Position																															
0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000 (Peripheral reset is invalid)																															
Name	Reserved																								TRIM1				TRIM0			
Access																									RW				RW			

Bit	Name	Functional Description	Access
31:8	Reserved	-	-
7:4	TRIM1	VC1 Trim Value 2 sets of trim values are stored in Flash. Select the appropriate Flash address based on the actual situation, read the trim value stored in that address, and then write it into VC_TRIM.TRIM1. - When the reference voltage at the positive terminal or negative terminal of the VC is less than or equal to 1/2 AVCC: 0x001007C2 - When the reference voltage at the positive terminal or negative terminal of the VC is greater than 1/2 AVCC: 0x001007C3	RW
3:0	TRIM0	VC0 Trim Value 2 sets of trim values are stored in Flash. Select the appropriate Flash address based on the actual situation, read the trim value stored in that address, and then write it into VC_TRIM.TRIM0. - When the reference voltage at the positive terminal or negative terminal of the VC is less than or equal to 1/2 AVCC: 0x001007C0 - When the reference voltage at the positive terminal or negative terminal of the VC is greater than 1/2 AVCC: 0x001007C1	RW

20 LVD

20.1 Introduction

LVD can be used to monitor the voltage of DVCC and chip pins. When the comparison result of the monitored voltage and the LVD threshold meets the trigger condition, the LVD will generate an interrupt or reset signal, and the user can perform some urgent tasks according to the signal.

20.2 Features

- 3 monitoring sources: DVCC, PA03, PA10
- 16-stage threshold voltage, 1.8-4.8V optional
- 3 interrupt triggering methods: high level/rising edge/falling edge
- 2 trigger results: reset and interrupt
- 12-stage filter configuration to prevent false triggering
- With hysteresis function, strong anti-interference

20.3 Functional Description

20.3.1 LVD Block Diagram

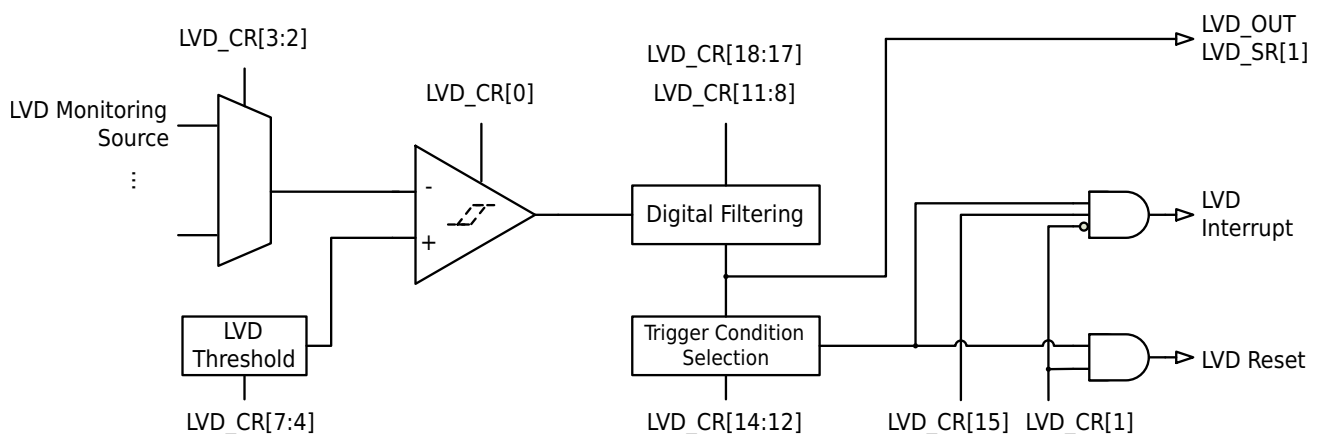


Figure 20-1 LVD Block Diagram



Note

For the selection of LVD monitoring sources, please refer to the descriptions of the SOURCE bit in "[LVD Configuration Register \(LVD_CR\)](#)".

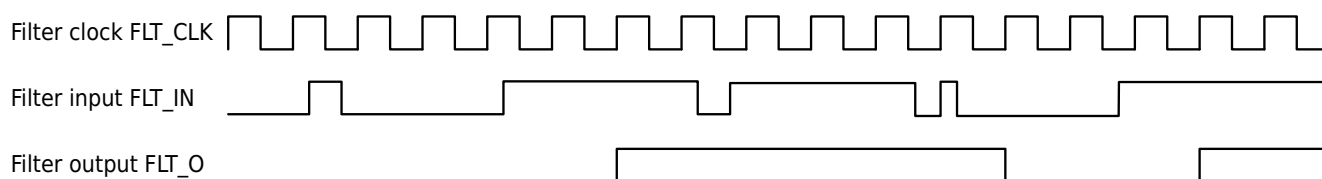
20.3.2 Digital Filtering

If the working environment of the chip is bad, the output of the hysteresis comparator will appear noise signal. The noise signal whose pulse width is less than LVD_CR.DEBOUNCE_TIME in the output waveform of the hysteresis comparator can be filtered out. If the digital filter module is disabled, the input and output signals of the digital filter module are the same. The filter clock used by the digital filter module is a built-in RC oscillator with an oscillation frequency of about 256kHz.

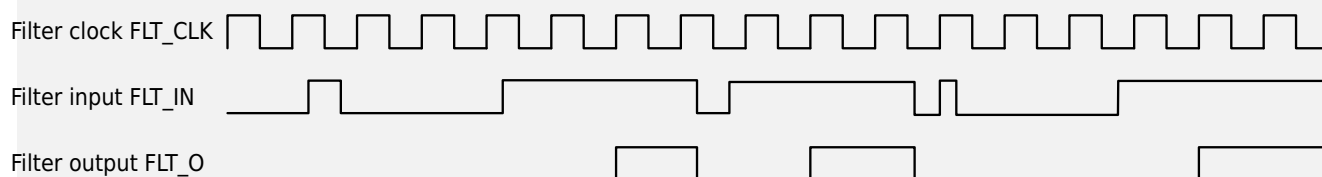
The digitally filtered signal level can be read from the register LVD_SR[1]; when the GPIO function is configured as LVD_OUT, the digitally filtered signal can be output from the GPIO for easy measurement.

Enable the digital filter module, and select the filter time as 2 filter clock cycles, the filter illustration is as follows:

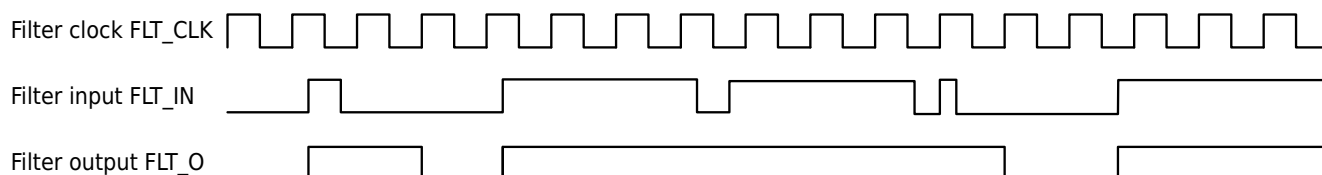
Normal Filter Mode



Peak Filter Mode



Valley Filter Mode



Spike Filter Mode

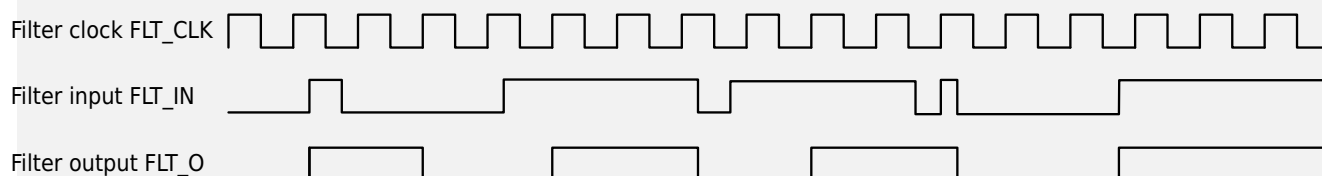


Figure 20-2 LVD Filter Output

20.3.3 Hysteresis Function

The built-in voltage comparator of LVD has a hysteresis function, which can enhance the anti-interference ability of the chip. The output signal of LVD will not flip until the input signal is higher or lower than the threshold voltage by the hysteresis voltage/2. The input and output signal diagram of LVD is as follows:

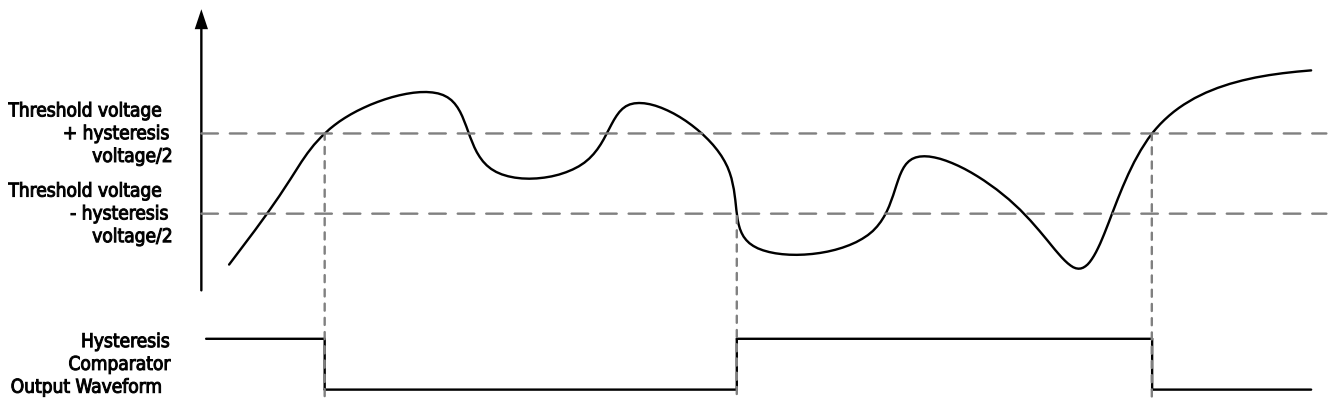


Figure 20-3 LVD Hysteresis Response

20.4 Operation Example

20.4.1 LVD Configured as Low Voltage Reset

Context

In this mode, the MCU is reset when the monitored voltage is lower than the threshold voltage.

Procedure

- Step1.** Configure LVD_CR.SOURCE to select the voltage source to be monitored.
- Step2.** Configure LVD_CR.VTDS to select the threshold voltage of LVD.
- Step3.** Configure LVD_CR.DEBOUNCE_TIME to select the LVD filter time.
- Step4.** Configure LVD_CR.FLT_MODE to select the filter mode.
- Step5.** Set LVD_CR.HTEN to 1, select high level to trigger LVD action.
- Step6.** Set LVD_CR.ACT to 1, select LVD action as reset.
- Step7.** Set LVD_CR.EN to 1 to enable LVD.

20.4.2 LVD Configured as Voltage Change Interrupt

Context

In this mode, an interrupt is generated when the monitored voltage goes above or below the threshold voltage.

Procedure

- Step1.** Configure LVD_CR.SOURCE to select the voltage source to be monitored.
- Step2.** Configure LVD_CR.VTDS to select the threshold voltage of LVD.
- Step3.** Configure LVD_CR.DEBOUNCE_TIME to select the LVD filter time.
- Step4.** Configure LVD_CR.FLT_MODE to select the filter mode.
- Step5.** Set LVD_CR.RTEN and LVD_CR.FTEN to 1, select level change to trigger LVD action.
- Step6.** Set LVD_CR.ACT to 0, select LVD action as interrupt.
- Step7.** Set LVD_CR.IE to 1 to enable LVD interrupt.
- Step8.** Enable the LVD interrupt in the NVIC interrupt vector table.
- Step9.** Set LVD_CR.EN to 1 to enable LVD.
- Step10.** Execute the operations required by the user in the LVD interrupt service routine; write 0x00 to LVD_SR to clear the interrupt flag before exiting the interrupt service routine.

20.5 Register Description

20.5.1 Register List

Base address: 0x40002C00

Table 20-1 Low Voltage Detector (LVD) Register Offset

Offset	Register	Description	Width
0x000	LVD_CR	LVD Configuration Register	32
0x004	LVD_SR	LVD Status Register	32

20.5.2 LVD Configuration Register (LVD_CR)

Offset	Bit Position																																												
0x000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0													
Reset	0x00000000																																												
Name	Reserved														FLT_MODE	Reserved	IE	HTEN	RTEN	FTEN	DEBOUNCE_TIME				VTDS				SOURCE		ACT	EN													
Access															RW		RW	RW	RW	RW	RW				RW				RW		RW		RW	RW	RW										

Bit	Name	Functional Description	Access
31:19	Reserved	-	-
18:17	FLT_MODE	Filter pattern selection 0b00: Standard Filtering Mode. Filter output toggles only if input remains stable for a defined hold time after transition. 0b01: Peak Filtering Mode. Output sets to logic 1 only after input holds logic 1 for a stable duration; sets to 0 immediately on input 0. The purpose of this mode is to filter out input peaks that are not long enough. 0b10: Valley Filtering Mode. Output sets to logic 0 only after input holds logic 0 for a stable duration; sets to 1 immediately on input 1. The purpose of this mode is to filter out input valleys that are not long enough. 0b11: Glitch Filtering Mode. Output toggles immediately on input transition, then ignores subsequent input changes for a hold time. In this mode, the filter output will closely follow the input, and at the same time, it will prevent the output signal from having spikes.	RW
16	Reserved	-	-
15	IE	LVD interrupt enable 0b0: Disable 0b1: Enable	RW
14	HTEN	High level trigger enable The monitored voltage is lower than the threshold voltage 0b0: Disable 0b1: Enable	RW
13	RTEN	Rising edge trigger enable The monitored voltage changes from above the threshold voltage to below the threshold voltage 0b0: Disable 0b1: Enable	RW

Bit	Name	Functional Description	Access
12	FTEN	Falling edge trigger enable The monitored voltage changes from below the threshold voltage to above the threshold voltage 0b0: Disable 0b1: Enable	RW
11:8	DE-BOUNCE_TIME	Digital filter time configuration N represents filter clock cycles 0x0: Digital filter disabled 0x1: N = 2 0x2: N = 4 0x3: N = 8 0x4: N = 16 0x5: N = 32 0x6: N = 64 0x7: N = 128 0x8: N = 256 0x9: N = 512 0xA: N = 1024 0xB: N = 2048 - Other: N = 4096	RW
7:4	VTDS	LVD threshold voltage selection 0b0000: 1.8V 0b0001: 2.0V 0b0010: 2.2V 0b0011: 2.4V 0b0100: 2.6V 0b0101: 2.8V 0b0110: 3.0V 0b0111: 3.2V 0b1000: 3.4V 0b1001: 3.6V 0b1010: 3.8V 0b1011: 4.0V 0b1100: 4.2V 0b1101: 4.4V 0b1110: 4.6V 0b1111: 4.8V	RW
3:2	SOURCE	LVD monitoring source selection 0b00: DVCC power voltage 0b01: PA03 port input voltage 0b10: PA10 port input voltage 0b11: DVCC power voltage	RW
1	ACT	LVD trigger action selection 0b0: NVIC interrupt 0b1: System reset	RW

Bit	Name	Functional Description	Access
0	EN	LVD enable control ● 0b0: Disable ● 0b1: Enable	RW

20.5.3 LVD Status Register (LVD_SR)

Offset	Bit Position																																
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reset	0x00000000																																
Name	Reserved																														FLTV		INTF
Access																															RO		RW

Bit	Name	Functional Description	Access
31:2	Reserved	-	-
1	FLTV	Level value of digital filter output 0b0: Digital filter output low level 0b1: Digital filter output high level	RO
0	INTF	LVD interrupt flag Write 0 to clear the interrupt flag, write 1 has no effect. Read: 0b0: No interrupt occurred 0b1: Interrupt occurred	RW

21 TRNG

21.1 Introduction

The true random number module generates 64-bit true random numbers.

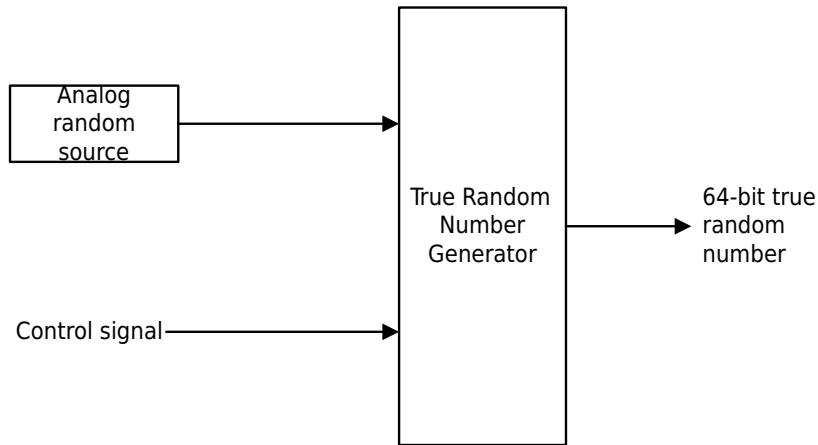


Figure 21-1 True Random Number Generator Function Block Diagram

21.2 Functional Description

21.2.1 True Random Number Generation Principle

This module uses an internal analog random source, which can generate 64bits true random numbers every time it is started. In addition, software configuration can be performed on the way of true random number generation. For details, refer to the Register Description section. The generated 64-bit true random numbers are stored in DATA0 and DATA1 registers respectively.

21.3 Operation Example

21.3.1 Generating 64bits True Random Number (The First Time After Power-On)

Procedure

- Step1.** Enable the random source circuit: Set TRNG_CR.TRNGCIR_EN to 0b1 to start the random source circuit, which begins outputting serial true random numbers.
- Step2.** Select reload initial value: Set TRNG_MODE.LOAD to 0b1 so that the initial value of newly generated true random numbers is obtained from the random source.
- Step3.** Select direct feedback mode for PRNG64: Set TRNG_MODE.FDBK to 0b1 to XOR the feedback signal with the random source before inputting it into the PRNG.
- Step4.** Select the shift count for PRNG64: Set TRNG_MODE.CNT to 0b110 to choose 256 shifts.
- Step5.** Generate true random numbers: Set TRNG_CR.TRNG_RUN to 0b1, and the hardware performs operations based on the true random number generation configuration. Upon completion, the hardware automatically clears TRNG_CR.TRNG_RUN.
- Step6.** Select not to reload the initial value: Set TRNG_MODE.LOAD to 0b0.
- Step7.** Select direct feedback mode for PRNG64: Set TRNG_MODE.FDBK to 0b0 to directly input the feedback signal into the PRNG.
- Step8.** Select the shift count for PRNG64: Set TRNG_MODE.CNT to 0b100 to choose 64 shifts.

- Step9.** Generate true random numbers: Set TRNG_CR.TRNG_RUN to 0b1, and the hardware performs operations based on the true random number generation configuration. Upon completion, the hardware automatically clears TRNG_CR.TRNG_RUN.
- Step10.** Read true random numbers: After detecting that TRNG_CR.TRNG_RUN has changed to "0," the software reads TRNG_DATA0 and TRNG_DATA1 to obtain a 64-bit true random number.
- Step11.** After completing the generation of true random numbers, it is recommended to disable the random source circuit to save power: Set TRNG_CR.TRNGCIR_EN to 0b0 to turn off the random source circuit.

21.3.2 Generating 64bits True Random Number (Not Generated for the First Time After Power-on)

Procedure

- Step1.** Enable the random source circuit: Set TRNG_CR.TRNGCIR_EN to 0b1 to start the random source circuit, which begins outputting serial true random numbers.
- Step2.** Select not to reload the initial value: Set TRNG_MODE.LOAD to 0b0.
- Step3.** Select direct feedback mode for PRNG64: Set TRNG_MODE.FDBK to 0b1 to XOR the feedback signal with the random source before inputting it into the PRNG.
- Step4.** Select the shift count for PRNG64: Set TRNG_MODE.CNT to 0b110 to choose 256 shifts.
- Step5.** Generate true random numbers: Set TRNG_CR.TRNG_RUN to 0b1, and the hardware performs operations based on the true random number generation configuration. Upon completion, the hardware automatically clears TRNG_CR.TRNG_RUN.
- Step6.** Select direct feedback mode for PRNG64: Set TRNG_MODE.FDBK to 0b0 to directly input the feedback signal into the PRNG.
- Step7.** Select the shift count for PRNG64: Set TRNG_MODE.CNT to 0b100 to choose 64 shifts.
- Step8.** Read true random numbers: After detecting that TRNG_CR.TRNG_RUN has changed to "0," the software reads TRNG_DATA0 and TRNG_DATA1 to obtain a 64-bit true random number. If new true random numbers need to be generated continuously, proceed to step 2 until the requirements are met.
- Step9.** After completing the generation of true random numbers, it is recommended to disable the random source circuit to save power: Set TRNG_CR.TRNGCIR_EN to 0b0 to turn off the random source circuit.

21.4 Register Description

21.4.1 Register List

Base address: 0x40008C00

Table 21-1 True Random Number Generator (TRNG) Register Offset

Offset	Register	Description	Width
0x00	TRNG_CR	Control register	32
0x04	TRNG_MODE	Mode Register	32
0x0C	TRNG_DATA0	Data Register 0	32
0x10	TRNG_DATA1	Data Register 1	32

21.4.2 Control Register (TRNG_CR)

Offset	Bit Position																															
0x00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000000																															
Name	Reserved																														TRNG_RUN	TRNGCIR_EN
Access																															RW	RW

Bit	Name	Functional Description	Access
31:2	Reserved	-	-
1	TRNG_RUN	True random number generation control The software writes "1" and starts to generate a new 64bits random number; after the operation is completed, the hardware is cleared; ● 0b0: The random number generation is completed ● 0b1: Write 1 to start random number generation, read 1 to indicate that random number is being generated	RW
0	TRNGCIR_EN	Random source circuit enable ● 0b0: Turn off the random source ● 0b1: Turn on the random source	RW

21.4.3 Mode Register (TRNG_MODE)

Offset	Bit Position																															
0x04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0x00000010																															
Name	Reserved																												CNT		FDBK	
Access																													RW		RW	
																													RW		LOAD	

Bit	Name	Functional Description	Access
31:5	Reserved	-	-
4:2	CNT	Feedback shift times of 64-bit TRNG ● 0b000: Shift 0 times (that is, output the sampling value of the random source) ● 0b001: Shift 8 times ● 0b010: Shift 16 times ● 0b011: Shift 32 times ● 0b100: Shift 64 times ● 0b101: Shift 128 times ● 0b110: Shift 256 times ● 0b111: Setting prohibited	RW
1	FDBK	During the shift operation, whether the feedback signal of the 64-bit TRNG is XORed with the random source ● 0b0: Do not perform XOR operation ● 0b1: Perform XOR operation	RW
0	LOAD	When generating a new random number, whether the 64-bit TRNG obtains a new initial value from the random source ● 0b0: Do not load new initial values (generating pseudo-random numbers) ● 0b1: Load a new initial value (generating true random numbers)	RW

21.4.4 Data Register 0 (TRNG_DATA0)

Offset	Bit Position																															
0x0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	DATA0																															
Access	RO																															

Bit	Name	Functional Description	Access
31:0	DATA0	Low 32-bit random number	RO

21.4.5 Data Register 1 (TRNG_DATA1)

Offset	Bit Position																															
0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xFFFFFFFF																															
Name	DATA1																															
Access	RO																															

Bit	Name	Functional Description	Access
31:0	DATA1	High 32-bit random number	RO

22 SWD Debug Interface

22.1 Introduction

This series of products uses an ARM Cortex-M0+ core, which features a hardware debugging module SWD that supports complex debugging operations. The hardware debug module allows the core to stop when fetching instructions (instruction breakpoints) or accessing data (data breakpoints). When the core is stopped, both the internal state of the core and the external state of the system can be queried in the IDE. After the query is complete, the core and peripherals can be restored and program execution continues. When this series of MCUs are connected to a debugger and start debugging, the debugger will use the hardware debugging module of the core to perform debugging operations.

22.2 Functional Description

22.2.1 Debug System Diagram

This product uses an ARM Cortex-M0+ core and includes hardware extensions for advanced debugging function; therefore, the debugging function of this product are consistent with Cortex-M0+. Debug extensions allow the core to halt the core when fetching instructions (instruction breakpoints) or fetching accessing data (data breakpoints). When the core stops, you can query the internal state of the core and the external state of the system. After querying, the core and system are resumed, and program execution is restored.

When the debug host is connected to the MCU and debugged, the debug function will be used.

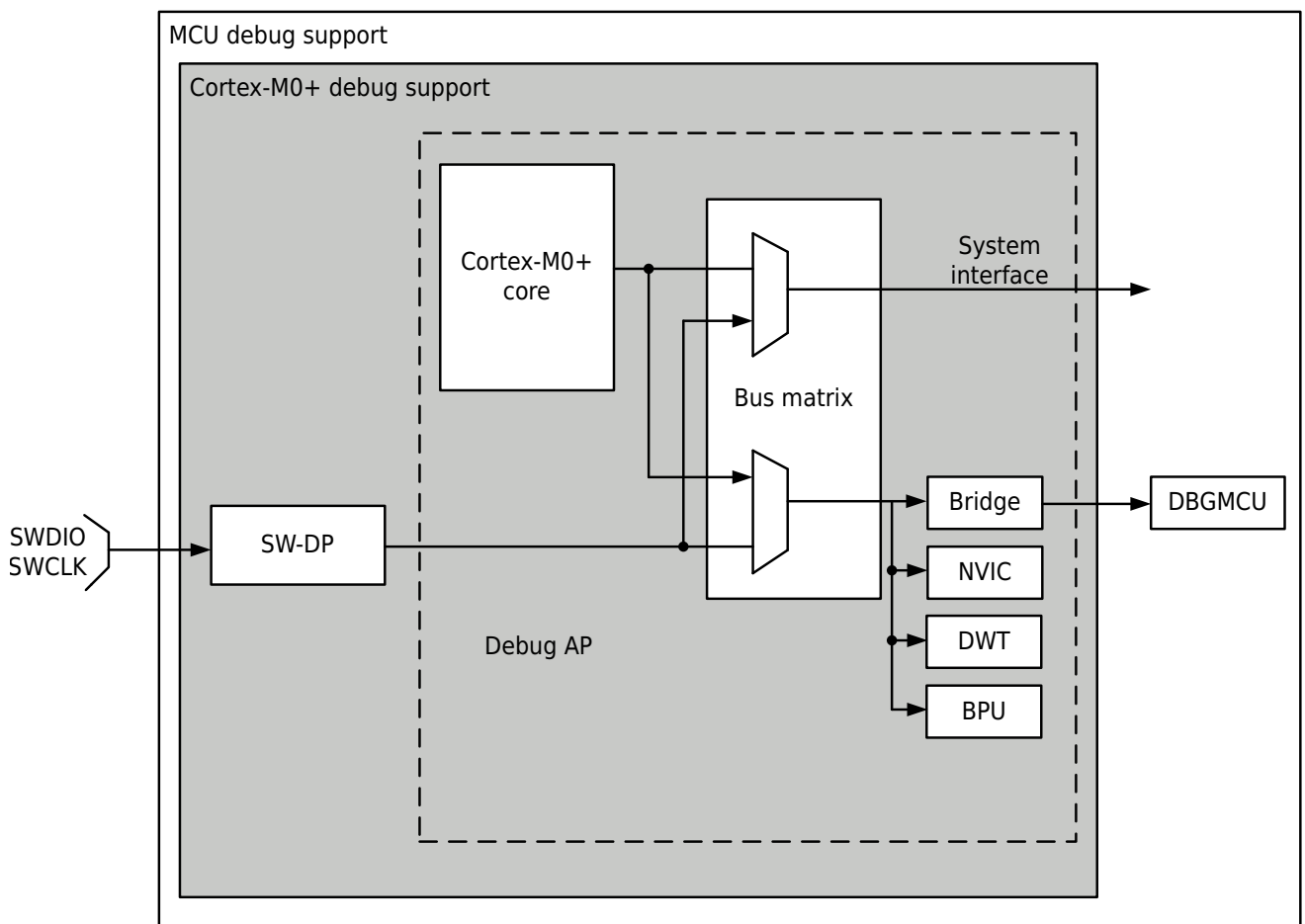


Figure 22-1 Debug Support Block Diagram

The built-in debugging function in the Cortex-M0+ core are part of the ARM® CoreSight design suite.

ARM® Cortex-M0+ core provides integrated on-chip debugging support, including:

- SW-DP: Serial line
- BPU: Breakpoint unit
- DWT: Data watchpoint trigger



Note

For detailed information on the debugging function supported by the ARM® Cortex-M0+ core, see "Cortex-M0+ Technical Reference Manual".

22.2.2 ARM® Reference Documentation

- Cortex-M0+ Technical Reference Manual (TRM)
Available at developer.arm.com.
- ARM® debug interface V5
- ARM® CoreSight Design Suite Version r1p1 Technical Reference Manual
- ARMv6-M Architecture Reference Manual

22.2.3 Debug Port Pins

SWD Port Pins

The SWD interface of this series requires 2 pins, as shown in the table below.

SWD Port Name	Debugging Function	Pin Assignment
SWCLK	Serial clock	PA14
SWDIO	Serial data input/output	PA13

SW-DP Pin Assignment

If the [Encryption Chip] option is enabled when programming the program, the SWD debugging function will be disabled after power-on. If the [Encryption Chip] option is not enabled during programming, the PA13/PA14 pins will be initialized as dedicated pins for debugger use after power-on. Users can set SYSCTRL1.SWD_USE_IO register to disable the SWD pin debugging function, and the SWD pin will be released to be used as a normal GPIO. The configuration and function of SWD pins are summarized in the following table:

[Encryption Chip] Option	SWD_USE_IO Configuration	PA13/PA14 Function
Encryption	0	-
Encryption	1	GPIO
No encryption	0	SWD
No encryption	1	GPIO

Internal Pull-Up on SWD Pins

After the user software releases the SW I/O, the GPIO controller takes control of these pins. The reset state of the GPIO control register will put the I/O into the equivalent state:

- SWDIO: input pull-up
- SWCLK: input pull-up

22.2.4 SWD Port

22.2.4.1 Introduction to SWD Protocol

This synchronous serial protocol uses two pins:

- SWCLK: Clock from host to target
- SWDIO: Bidirectional

Using this protocol, two register sets (DPACC register set and APACC register set) can be read and written simultaneously. LSB is ahead when transferring data.

For SWDIO bidirectional management, the line must be pulled up on the circuit board (ARM® recommends using 100k). These pull-up resistors can be configured internally, so external pull-up resistors are not required.

Every time the SWDIO direction is changed in the protocol, the conversion time is inserted, and the line is not driven by the host or the target. By default, this conversion time is one bit time, but can be adjusted by configuring the SWCLK frequency.

22.2.4.2 SWD Protocol Sequence

Each sequence consists of three phases:

1. Host transmitted data packet request (8-bit)
2. Target transmitted acknowledgment response (3-bit)
3. Data transfer phase transmitted by host or target (33-bit)

Table 22-3 Host Transmitted Data Packet Request

Bit	Name	Description
0	Start	Must be 0b1
1	APnDP	● 0b0: DP access ● 0b1: AP access
2	RnW	● 0b0: Write request ● 0b1: Read request
4:3	A[3:2]	Address field of DP or AP register
5	Parity check	Unit parity for the first few bits
6	Stop	0b0
7	Reside	Not driven by the host. Due to the presence of the pull-up, the target must read it as 0b1.

For detailed information on the DPACC and APACC registers, please refer to "Cortex-M0+ Technical Reference Manual".

The packet request is always followed by a conversion time (1 bit by default), at which point neither the host nor the target will drive.

Table 22-4 Target Transmitted Acknowledgment Response

Bit	Name	Description
2:0	ACK	● 0b001: FAULT ● 0b010: WAIT ● 0b100: OK

The ACK response must be followed by conversion time only when a READ transaction occurs or a WAIT or FAULT acknowledgment is received.

Table 22-5 Data Transfers Transmitted by Host or Target

Bit	Name	Description
31:0	WDATA or RDATA	Write or read data
32	Parity check	Single parity for 32 data bits

The conversion time must be after the DATA transfer only when a READ transaction occurs.

22.2.4.3 SW-DP State Machine (Reset, Idle State, ID Code)

The SW-DP state machine includes an internal ID code to identify the SW-DP. This code complies with the JEP-106 standard and is set to the default ARM® code **0x0BC11477** (which corresponds to the Cortex®-M0+).



Note

The SW-DP state machine is inactive until the target reads the ID code.

- The SW-DP state machine is in reset state after power-on reset or after the line has been at high level for more than 50 cycles.
- If the line is low for at least two cycles after the reset state, the SW-DP state machine is in the idle state.
- After the reset state, the state machine must first enter the idle state and then perform a read access to the DP-SW ID CODE register. FAULT acknowledgment response on another transaction.

For more detailed information about the SW-DP state machine, please refer to "Cortex-M0+ Technical Reference Manual" and "ARM® CoreSight Design Suite Version r1p1 Technical Reference Manual".

22.2.4.4 DP and AP Read/Write Access

- Read access to DP is not delayed: the target response can be sent immediately (if ACK=OK) or delayed (if ACK=WAIT).
- Delay read access to AP. This means that access results will be returned on the next transfer. If the next access to be performed is not an AP access, the DP-RDBUFF register must be read to obtain the result.
- Every time an AP read access or RDBUFF read request is made, the READOK flag of the DP-CTRL/STAT register will be updated to know whether the AP read access is successful.
- SW-DP has a write buffer (for DP or AP writes) so that write operations can be accepted even while other operations are still pending. If the write buffer is full, the target acknowledgment response is WAIT. Except for IDCODE read, CTRL/STAT read or ABORT write, these operations will also be accepted when the write buffer is full.

- Due to the asynchronous clock domains SWCLK and HCLK, two additional SWCLK cycles are required after the write operation (after the parity bit) for the write operation to take effect internally. These cycles should be applied while the line is being driven low (idle state).

This is especially important when writing to the CTRL/STAT register to initiate a power-on request. Otherwise the next operation (one that is only valid after the core is powered on) will be executed immediately, which will cause a failure.

22.2.4.5 SW-DP Register

These registers can be accessed when APnDP=0:

A[3:2]	RW	CTRLSEL Bit of the SELECT Register	Register	Description
0b00	Read	-	IDCODE	The manufacturer code is set to the default ARM® code for Cortex®-M0+. 0x0BC11477 (identifies SW-DP)
0b00	Write	-	ABORT	-
0b01	Read/write	0	DP-CTRL/STAT	Purpose: ● Request system or debug power-on ● Configure transport operations for AP access ● Control compare and verify operations ● Read some status flags (overflow and power-on acknowledgment)
0b01	Read/write	1	WIRE CONTROL	Used to configure the physical serial port protocol (such as switching time duration)
0b10	Read	-	READ RESEND	Allow recovery of reads data from corrupted debug software transfers without repeating the original AP transmission.
0b10	Write	-	SELECT	4-word register window used to select the currently accessed port and active
0b11	Read/write	-	READ BUFFER	Since an AP access has been issued, this read buffer is very useful (it provides the result of the AP read request during the execution of the next AP transaction). This read buffer captures data from the AP and displays the result of the previous read without requiring a new operation to be initiated

22.2.4.6 SW-AP Register

These registers can be accessed when APnDP=1:

There are multiple AP registers, which are addressed in the following combinations:

- Shift value A[3:2]
- Current value of DP SELECT register

Address	A[3:2] Value	Description
0x0	0b00	Reserved, must be kept at the reset value.

Address	A[3:2] Value	Description
0x4	0b01	DP CTRL/STAT register. Used for: ● Request system or debug power-on ● Configure transport operations for AP access ● Control compare and verify operations ● Read some status flags (overflow and power-on acknowledgment)
0x8	0b10	DP SELECT register: used to select the currently accessed port and active 4-word register window. ● Bit[31:24]: APSEL, select the current AP ● Bit[23:8]: Reserved ● Bit[7:4]: APBANKSEL, select the active 4-word register window within the current AP ● Bit[3:0]: Reserved
0xC	0b11	DP RDBUFF Register: Used by the debugger to retrieve the final result after performing a series of operations (without initiating a new JTAG-DP request).

22.2.5 Core Debugging

Debug the core through the core debug registers. Debug access to these registers is through the debug access port. It consists of four registers:

Register	Description
DHCSR	32-bit debug stop control and status register This register provides information about the state of the processor, enables the core to enter a debug stop state, and provides processor stepping capabilities.
DCRSR	17-bit debug core register selector registers This register selects the processor register to be read or written.
DCRDR	32-bit debug core register data registers This register holds data read and written between the register and the processor selected by the DCRSR (selector) register.
DEMCR	32-bit debug exception and monitor control registers This register provides vector capture and debug monitor control.

These registers are not reset on system reset. They can only be reset by a power-on reset. For more information, see "Cortex-M0+ Technical Reference Manual".

In order to put the core into the debug stop state immediately after reset, it is necessary to:

- Enable Debug and Exception Monitoring Control Register bit 0 (VC_CORRESET)
- Enable Debug Halt Control and Status Register bit 0 (C_DEBUGEN)

22.2.6 BPU (Breakpoint Unit)

Cortex-M0+ BPU implementation provides 4 breakpoint registers.

Processor breakpoint implements PC-based breakpoint functionality.

For more information about the BPU CoreSight identification registers, including their addresses and access types, please see "ARMv6-M Architecture Reference Manual" and "ARM® CoreSight Component Technical Reference Manual".

22.2.7 DWT (Data Watchpoint)

The Cortex-M0+ DWT implementation provides 1 watchpoint register groups.

DWT Function

The processor watchpoint implements data address and PC-based watchpoint functions (i.e., PC sampling registers) and includes comparator address masking. For detailed information, please refer to the "ARMv6-M Architecture Reference Manual".

DWT Program Counter Sampling Register

Processors implementing the data watchpoint unit also implement the ARMv6-M optional DWT Program Counter Sampling Register (DWT_PCSR). This register allows the debugger to periodically sample the PC without stopping the processor, enabling coarse-grained analysis. For more information, see "ARMv6-M Architecture Reference Manual".

Cortex-M0+ DWT_PCSR records instructions that pass condition codes as well as instructions that do not pass condition codes.

22.2.8 MCU Debug Component (DBG)

MCU debug component helps the debugger provide support for:

- Low power mode
- Timer and watchdog clock control during breakpoints

Debug Support for Low Power Mode

To enter low power mode, the instruction WFI or WFE must be executed.

The MCU supports several low power modes that disable the CPU clock or reduce CPU power consumption.

The core does not allow FCLK or HCLK to be turned off during a debugging session. They must remain active as they are required for debug connections during debugging. The MCU incorporates special methods that allow users to debug software in low power mode.

Debug Support for Timer and Watchdog

During a breakpoint, the behavior of the counters of the timer and watchdog must be selected:

- The counter continues to count while the breakpoint is generated. This method is required when using PWM to control motors.
- The counter stops to count while the breakpoint is generated. This method is required when using the watchdog.

23 Device Electronic Signature

23.1 Introduction

The electronic signature is stored in the system storage area of the flash memory module and can be read by SWD or CPU. The chip identification information is programmed during manufacturing. User firmware or external devices can read the electronic signature to automatically match MCUs with different configurations.

23.2 Functional Description

23.2.1 Product Unique Identifier (UID)

Typical application scenarios for unique identifier (UID):

- Used as a serial number
- UID is used as a security key when used in conjunction with software cryptographic primitives and protocols to increase the security of code in Flash prior to programming the internal Flash
- Activation of the secure bootstrap process, etc.

The UID provides a reference number that is unique to any device and in any context. The user can never change these bits. The UID can also be read in different ways like single byte/halfword/word and then concatenated using a custom algorithm.

23.3 Register Description

23.3.1 Product Unique Identifier (UID) Register (80 bits)

Base address: 0x001008B4

Offset	Description	UID Bits (80bits)							
		7	6	5	4	3	2	1	0
0	Y Coordinate on the wafer	UID[7:0]							
1	X Coordinate on the wafer	UID[15:8]							
2	Wafer Number	UID[23:16]							
3	Lot Number	UID[31:24]							
4		UID[39:32]							
5		UID[47:40]							
6		UID[55:48]							
7		UID[63:56]							
8		UID[71:64]							
9		UID[79:72]							

23.3.2 Product Model Register

0x00100740-0x0010074F store the ASCII code representing the product model. If the product model is less than 16 bytes, fill it with 0x00.

Example: The product model represented by 484333324C3032314338504200000000 is HC32L021C8PB.

23.3.3 FLASH Capacity Register

Base address: 0x00100760

Offset	Bit Position																															
'	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	'																															
Name	FLASHSIZE																															
Access	RO																															

Bit	Name	Functional Description	Access
31:0	FLASHSIZE	Product built-in Flash capacity In unit of Byte 0x00010000 means the Flash capacity is 64K Byte	RO

23.3.4 RAM Capacity Register

Base address: 0x00100764

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	-																															
Name	RAMSIZE																															
Access	RO																															

Bit	Name	Functional Description	Access
31:0	RAMSIZE	Product built-in RAM capacity In unit of Byte 0x00001800 means the RAM capacity is 6K Byte	RO

23.3.5 Pin Count/Flash Sector Size Configuration Register

Base address: 0x00100768

Offset	Bit Position																															
-	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	-																															
Name	PINCOUNT																FLASHSECTORSIZE															
Access	RO																RO															

Bit	Name	Functional Description	Access
31:16	PINCOUNT	Product pins count 0x0014 means the number of product pins is 20	RO
15:0	FLASHSEC-TORSIZE	Flash sector size In unit of Byte 0x0200 represents 512Byte.	RO

A Glossary

Terminology	Full Name	description
Byte	-	Bytes, 8-bit data
Half Word	-	Half word, 16-bit data
Word	-	Words, 32-bit data
Double Word	-	Double word, 64 -bit data.
IAP	-	It refers to reprogram the MCU's Flash during the running of a user program (programming in the application).
ICP	-	It means that the MCU's Flash of the MCU can be programmed by using JTAG protocol, SWD protocol or bootstrap program when the device is installed on the user application circuit board (in-circuit/online programming).
AHB	Advanced High-performance Bus	Advanced High-Performance Bus
APB	Advanced Peripheral Bus	Advanced peripheral bus
DMA	Direct Memory Access	Direct memory access
TIM/TMR	-	Timer

Revision History

Revision/Date	Changes
Rev1.20 Aug. 28, 2026	1. RTC: In the "Timestamp Time Register (RTC_TSTR)" chapter, add note "This bit uses 24-hour format and is not affected by RTC_CR0.AMPM" to HOURH/HOURL bit. 2. LPUART: In the "Multi-Machine Communication" chapter, the description of Slave Address Configuration is optimized. 3. ATIM3: In the "Mode2/3 Compare Capture Mode - Capture Input" chapter, the capture input description of CH0B is optimized. When MSCR.IB0S is configured to 1, MSCR.TS does not support selecting the CH0B filter signal. 4. SWD Debug Interface: The internal ID of SW-DP is modified to "0x0BC11477".
Rev1.10 Nov. 21, 2025	1. FLASH: In the "FLASH Security Protection" chapter, the read-only FLASH_LOCK_ST has been removed from the list of "registers that can only be modified through write sequence operations". 2. SYSCTRL: In the "System Clock Introduction" chapter, the description format is optimized and add the correspondence relationship between clock inputs and pins; In the "Cortex-M0+ LOCKUP Reset" chapter, the description "This reset function is enabled when SYSCTRL1.LOCKUP_EN is set to 1" is added; In the "RCL Control Register (RCL_CR)" chapter, the MSEL bit is added, the address for the calibration value of Bit[9:0] is expanded to a 32-bit width, and the description is optimized; In the "XTL Control Register (XTL_CR)" chapter, the description "when an external low-speed crystal oscillator is connected" is added to the Bit9 explanation, the recommended value for the DRIVER bit is changed from "0b10" to "0b01"; In the "Reset Flag Register (RESET_FLAG)" chapter, add note "The reset value is 0x03 at power-on reset. After the program determines the reset source, it should clear the register so that it can determine the reset source the next time a reset occurs", simultaneously optimize the functional descriptions of each bit. 3. CTRIM: In the "Low Power Timer Mode" chapter, the description "the TOG signal can serve as the trigger source for the ADC" is added; 4. SYSTICK: In the "SysTick Calibration Value Register (CALIB)" chapter, the reset value is changed from "0xFFFFFFFF" to "0x40000148". 5. CTIM: A new Chapter "Timer Triggers ADC" is added. 6. ATIM3: In the "Mode 0 Counting Timer Function" chapter, the GATEP polarity in the functional block diagram has been modified, counting function description optimized, and ETR clock frequency constraints added; In the "Mode 2/3 Compare Capture Mode - Single Pulse Output" chapter, the pulse counting timing diagram details have been optimized with additional relevant explanations; In the "Mode 2/3 Compare Capture Mode - Capture Input" chapter, the GATE signal has been added to the CH0B capture input port selection diagram, and descriptions and explanations related to GATE have been supplemented simultaneously; In the "Mode 2/3 Compare Capture Mode Registers - Master/Slave Mode Control Register (TIM_MSCR)" chapter, the Bit[2:0] function description for MMS=0 now specifies ATIM3 supports "software update TG by writing M23CR.TG";

Revision/Date	Changes
Rev1.10 Nov. 21, 2025	7. RTC: In the "Control Register 0 (RTC_CR0)" chapter, supplement the specific time descriptions for setting PRDS to 1 minute or 1 hour or 1 day or 1 month; In the "Interrupt Enable Register (RTC_IER)" chapter, TSOV is deleted; in the "Interrupt Flag Register (RTC_IFR)" and "Interrupt Clear Register (RTC_ICR)" chapters, Bit4 name is optimized with the description "interrupt" removed. 8. IWDG: In the "Control Register (IWDG_CR)" chapter, the PRS bit explanation is moved to the section overview with description changed to "This register can only be modified when IWDG_SR.CRF=0"; in the "Status Register (IWDG_SR)" chapter, Bit0 name is changed from "PRSF" to "CRF", function is changed from "IWDG_CR.PRS update flag" to "CR register update flag" with corresponding modifications to the functional description values, and the explanation is updated to "CR register can only be updated when CRF is 0"; 9. LPUART: In the "Single-Wire Half-Duplex Data Transmitting and Reception" chapter, the description "After configuring LPUART_SCON.HDSEL to 0b1, must wait for the transmission duration of one data bit (i.e., after 1/BaudRate seconds) before normal reception can occur" is added, Additionally, add configuration guidance for receiving data in single-wire half-duplex mode at baud rates of 9600/4800; In the "Address Register (LPUART_SADDR)" chapter, Bit[7:0] name is changed from "SADDR" to "ADDR"; In the "Address Mask Register (LPUART_SADEN)" chapter, Bit[7:0] name is changed from "SADEN" to "MASK", the description of Bit[7:0] is added. 10. HSI2C: The "Operation Example" chapter has added "Master High-Speed Mode Switching". 11. ADC: In the "Conversion Timing and Conversion Speed" chapter, indicate the information of ADC conversion speed refer to Datasheet; In the "ADC Basic Configuration Register 0 (ADC_CR0)" chapter, the Bit[13:12] value is changed from "0b00, 0b11" to "0b00, 0b10", the Bit[3:2] function description has been supplemented with the note "Modifiable only when ADC is in idle state"; In the "ADC_EXTTRIGGER Register" chapter, the functional description is optimized, the note "To enable CTIM to trigger automatic ADC conversion, the corresponding trigger enable register of CTIM must be configured. For detailed operation steps, please refer to the Function Description - Timer Trigger ADC" section in the "Composite Timer (CTIM)" and "CTIM triggers ADC automatic conversion without the need to enable the corresponding interrupt. For detailed information, please refer to the Low Power Timer Mode" section in the "Clock Trim Module (CTRIM)" are added. 12. VC: In the "Setup/Response Time" chapter, usage description when enabling the DAC is added; In the "Control Register 0 (VCx_CR0)" chapter, Bit[9:8] note "When 0b10 and 0b11 are selected, the BGR auto turn-on" is deleted; the new register "Offset Trim Register (VC_TRIM)" is added. 13. LVD: Modify all AVCC descriptions in this chapter to DVCC.
Rev1.00 Apr. 14, 2025	First official release.