

HC32L110 Series

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

Reference Manual

Rev2.50 August 2026

Statement

- ★ Xiaohua Semiconductor Co., Ltd. (hereinafter referred to as "XHSC") reserves the right to change, correct, enhance, modify Xiaohua Semiconductor products and / or this document at any time without prior notice. Users can get the latest information before placing orders. XHSC products are sold in accordance with the terms and conditions of sale set forth in the Basic Contract for Purchase and Sales.
- ★ 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.
- ★ Any graphics or words marked “®” or “™” are trademarks of XHSC. All other products or services shown on XHSC products are the property of their respective owners.
- ★ 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 and replaces information in previous versions.

©2026 Xiaohua Semiconductor Co., Ltd. All rights reserved

Contents

Statement	2
Contents	3
Table Index.....	16
Graph Index	17
Overview.....	21
1 System Structure.....	22
1.1 Overview	22
1.2 System address division.....	23
1.3 Memory and Module Address Assignment.....	25
2 Operating mode.....	26
2.1 Operating mode	28
2.2 Sleep mode	29
2.3 Deep sleep mode	30
3 System Controller (SYSCTRL).....	33
3.1 Clock source introduction	33
3.1.1 Internal high speed RC clock RCH.....	34
3.1.2 Internal low speed RC clock RCL	34
3.1.3 External low-speed crystal oscillator clock XTL.....	35
3.1.4 External high-speed crystal oscillator clock XTH	35
3.1.5 Clock start process	35
3.2 System Clock Switching	36
3.2.1 Standard Clock Switching Process	36
3.2.2 Switch from RCH to XTL example.....	37
3.2.3 Switch from RCH to XTH example.....	37
3.2.4 Example of switching from RCL to XTH	38
3.2.5 Switch from RCH to RCL example	39
3.2.6 Example of switching from RCL to RCH.....	39
3.2.7 Switch between different oscillation frequencies of RCH	39
3.3 Clock Calibration Module	40
3.4 Interrupt Wakeup Control	41
3.4.1 Enable the NVIC corresponding to the module that needs to wake up the processor	41
3.4.2 Method not to execute interrupt service routine after wake-up from deep-sleep mode	42
3.4.3 Using exit hibernation feature	42
3.5 Register	44

3.5.1	System Control Register 0 (SYSCTRL0)	45
3.5.2	System Control Register 1 (SYSCTRL1)	47
3.5.3	System Control Register 2 (SYSCTRL2)	48
3.5.4	RCH Control Register (RCH_CR)	48
3.5.5	Oscillation XTH Control Register (XTH_CR)	49
3.5.6	RCL Control Register (RCL_CR)	50
3.5.7	XTL Control Register (XTL_CR)	51
3.5.8	Peripheral Module Clock Control Register (PERI_CLKEN)	52
3.5.9	Systick Clock Control (SYSTICK_CR)	54
4	Reset Controller (RESET)	55
4.1	Reset Controller Introduction	55
4.1.1	Power-on and power-off reset POR/BOR	55
4.1.2	External reset pin reset	56
4.1.3	WDT reset	56
4.1.4	PCA reset	56
4.1.5	LVD low voltage reset	56
4.1.6	Cortex-M0+ SYSRESETREQ reset	56
4.1.7	Cortex-M0+ LOCKUP reset	56
4.2	Register	57
4.2.1	Reset flag register (Reset_flag)	57
4.2.2	Peripheral module reset control register (PERI_RESET)	58
5	Interrupt Controller (NVIC)	60
5.1	Overview	60
5.2	Interrupt priority	60
5.3	Interrupt digraph	61
5.4	Interrupt Input and Suspend Behavior	62
5.5	Interrupt wait	64
5.6	Interrupt source	64
5.7	Interrupt structure diagram	66
5.8	Register	68
5.8.1	Interrupt Enable Setting Register (SCS_SETENA)	68
5.8.2	Interrupt Enable Clear Register (SCS_CLRENA)	69
5.8.3	Interrupt Pending Status Set Register (SCS_SETPEND)	69
5.8.4	Interrupt Pending Status Clear Register (SCS_CLRPEND)	70
5.8.5	Interrupt Priority Register (SCS_IPR0)	70
5.8.6	Interrupt Priority Register (SCS_IPR1)	71
5.8.7	Interrupt Priority Register (SCS_IPR2)	71
5.8.8	Interrupt Priority Register (SCS_IPR3)	72

5.8.9	Interrupt Priority Register (SCS_IPR4)	72
5.8.10	Interrupt Priority Register (SCS_IPR5)	73
5.8.11	Interrupt Priority Register (SCS_IPR6)	73
5.8.12	Interrupt Priority Register (SCS_IPR7)	74
5.8.13	Interrupt Mask Special Register (SCS_PRIMASK)	75
5.9	Basic operation of the software	76
5.9.1	External interrupt enable	76
5.9.2	NVIC interrupt enable and clear enable	76
5.9.3	NVIC interrupt pending and clear pending	76
5.9.4	NVIC interrupt priority	76
5.9.5	NVIC interrupt mask	77
6	Port Controller (GPIO)	78
6.1	Introduction to Port Controllers	78
6.2	Port Controller Main Features	78
6.3	Port Controller Functional Description	79
6.3.1	Port configuration function	79
6.3.2	Port write	81
6.3.3	Port read	82
6.3.4	Port multiplexing function	83
6.3.5	Port interrupt function	83
6.4	Port Configuration Operations	84
6.4.1	Port multiplexing operation process	84
6.4.2	Port Interrupt Operation Flow	84
6.4.3	Port configuration operation flow	85
6.5	Port Controller Register Description	86
6.5.1	Port P0	89
6.5.2	Port P1	98
6.5.3	Port P2	106
6.5.4	Port P3	122
6.5.5	Port auxiliary Function	139
7	FLASH controller (FLASH)	144
7.1	Overview	144
7.2	Structural block diagram	144
7.3	Functional description	144
7.3.1	Sector Erase (Sector Erase)	145
7.3.2	Full Chip Erase (Chip Erase)	145
7.3.3	Write operation (Program)	146
7.3.4	Read operation	148

7.4	Erase Timing.....	148
7.5	Read wait period	149
7.6	Erase and write protection	150
7.6.1	Erase and write protection bit.....	150
7.6.2	PC address erase and write protection	150
7.7	Register write protection	150
7.8	Register	151
7.8.1	TNVS parameter register (FLASH_TNVS).....	151
7.8.2	TPGS parameter register (FLASH_TPGS)	152
7.8.3	TPROG parameter register (FLASH_TPROG).....	152
7.8.4	TSERASE register (FLASH_TSERASE)	153
7.8.5	TMERASE parameter register (FLASH_TMERASE)	153
7.8.6	TPRCV parameter register (FLASH_TPRCV)	154
7.8.7	TSRCV parameter register (FLASH_TSRCV)	154
7.8.8	TMRCV parameter register (FLASH_TMRCV)	155
7.8.9	CR register (FLASH_CR)	155
7.8.10	IFR register (FLASH_IFR)	156
7.8.11	ICLR register (FLASH_ICLR)	156
7.8.12	BYPASS register (FLASH_BYPASS).....	157
7.8.13	SLOCK register (FLASH_SLOCK)	157
8	RAM controller(RAM).....	158
8.1	Overview	158
8.2	Functional description	158
8.3	Register	159
8.3.1	Control Register (RAM_CR).....	159
8.3.2	Parity Error Address Register (RAM_ERRADDR)	160
8.3.3	Error Interrupt Flag Register (RAM_IFR)	160
8.3.4	Error Interrupt Flag Clear Register (RAM_ICLR)	161
9	Basic Timer (TIM0/1/2)	162
9.1	Introduction to Basic Timers.....	162
9.2	Base Timer Function Description	162
9.2.1	Counting function	164
9.2.2	Buzzer function.....	165
9.3	Base Timer Interconnection.....	165
9.3.1	GATE interconnection.....	165
9.3.2	Toggle Output Interconnects.....	166
9.4	Base Timer register description.....	167
9.4.1	16 -bit Mode Reload Register (TIMx_ARR)	167

9.4.2	16 -bit Mode Count Register (TIMx_CNT)	168
9.4.3	32 -bit mode count register (TIMx_CNT32)	168
9.4.4	Control Register (TIMx_CR)	169
9.4.5	Interrupt Flag Register (TIMx_IFR).....	169
9.4.6	Interrupt Flag Clear Register (TIMx_ICLR)	170
10	Low Power Timer (LPTIM)	171
10.1	Introduction to LPTimer	171
10.2	LPTimer Function Description	171
10.2.1	Counting function	173
10.2.2	Timing function	173
10.3	LPTimer Interconnect.....	173
10.3.1	GATE interconnection.....	173
10.3.2	EXT Interconnection.....	173
10.3.3	Toggle Output Interconnects.....	173
10.4	LPTimer register description.....	174
10.4.1	Counter Count Value Register (LPTIM_CNT)	174
10.4.2	Reload Register (LPTIM_ARR)	175
10.4.3	Control Register (LPTIM_CR)	175
10.4.4	Interrupt Flag Register (LPTIM_IFR).....	176
10.4.5	Interrupt Flag Clear Register (LPTIM_ICLR)	176
11	Programmable Count Array (PCA)	177
11.1	Introduction to PCA.....	177
11.2	PCA function description	178
11.2.1	PCA Timer / Counter.....	178
11.2.2	PCA capture function	180
11.2.3	PCA comparison function.....	182
11.3	Interconnection and control of PCA module and other modules.....	187
11.3.1	ECI interconnection.....	187
11.3.2	PCACAP0	187
11.3.3	PCACAP1	187
11.3.4	PCACAP [4:2]	187
11.4	PCA register description	188
11.4.1	Control Register (PCA_CCON).....	189
11.4.2	Mode Register (PCA_CMOD).....	190
11.4.3	Count register (PCA_CNT)	191
11.4.4	Interrupt Clear Register (PCA_ICLR)	191
11.4.5	Compare Capture Mode Register (PCA_CCAPM0~4)	192
11.4.6	Compare the upper 8 bits of the capture data register (PCA_CCAP0~4H).....	193

11.4.7	Compare the lower 8 bits of the capture data register (PCA_CCAP0~4L)	193
11.4.8	Compare capture 16 -bit register (PCA_CCAP0~4)	194
11.4.9	Compare High Speed Output Flag Register (PCA_CCAP0)	194
12	Advanced Timer (TIM4/5/6)	195
12.1	Introduction to Advanced Timers.....	195
12.2	Advanced Timer Function Description	197
12.2.1	Basic action	197
12.2.2	Timer selection	199
12.2.3	Counting direction	200
12.2.4	Digital filtering	200
12.2.5	Software synchronization.....	201
12.2.6	Hardware synchronization	203
12.2.7	Cache function.....	205
12.2.8	General PWM output.....	207
12.2.9	Orthogonal coding count	212
12.2.10	Periodic interval response.....	217
12.2.11	Protection mechanism	217
12.2.12	Interrupt Description.....	218
12.2.13	Brake protection	219
12.2.14	Interconnection.....	221
12.3	Register description.....	224
12.3.1	Common Count Reference Register (TIMx_CNTER)	226
12.3.2	Universal Period Reference Register (TIMx_PERAR)	226
12.3.3	General purpose period buffer register (TIMx_PERBR).....	227
12.3.4	Universal Compare Base Registers (TIMx_GCMAR-GCMDR)	227
12.3.5	DEAD TIME REFERENCE REGISTER (TIMx_DTUAR-DTDAR)	228
12.3.6	General Control Register (TIMx_GCONR).....	229
12.3.7	Interrupt Control Register (TIMx_ICONR).....	230
12.3.8	Port Control Register (TIMx_PCONR)	231
12.3.9	Buffer Control Register (TIMx_BCONR).....	232
12.3.10	Dead Time Control Register (TIMx_DCONR)	233
12.3.11	Filter Control Register (TIMx_FCONR).....	234
12.3.12	Valid Period Register (TIMx_VPERR)	235
12.3.13	Status Flag Register (TIMx_STFLR).....	236
12.3.14	Hardware Start Event Select Register (TIMx_HSTAR)	237
12.3.15	Hardware Stop Event Select Register (TIMx_HSTPR)	239
12.3.16	Hardware Clear Event Select Register (TIMx_HCELR)	241
12.3.17	Hardware Capture A Event Select Register (TIMx_HCPAR)	243

12.3.18	Hardware Capture B Event Select Register (TIMx_HCPBR).....	244
12.3.19	Hardware Increment Event Select Register (TIMx_HCUPR)	245
12.3.20	Hardware Decrement Event Select Register (TIMx_HCDOR)	247
12.3.21	Software Synchronization Start Register (TIMx_SSTAR)	249
12.3.22	Software Sync Stop Register (TIMx_SSTPR)	249
12.3.23	Software Synchronous Clear Register (TIMx_SCLRR)	250
12.3.24	Interrupt Flag Register (TIMx_IFR).....	251
12.3.25	Interrupt Flag Clear Register (TIMx_ICLR)	252
12.3.26	Spread spectrum and interrupt trigger selection (TIMx_CR)	253
12.3.27	AOS Selection Control Register (TIMx_AOSSR).....	254
12.3.28	AOS Selection Control Register Flag Clear (TIMx_AOSCL)	255
12.3.29	Port Brake Control Register (TIMx_PTBKS)	256
12.3.30	Port Trigger Control Register (TIMx_TTRIG)	257
12.3.31	AOS Trigger Control Register (TIMx_ITRIG).....	258
12.3.32	Port Brake Polarity Control Register (TIMx_PTBKP).....	259
13	Real Time Clock (RTC)	260
13.1	Introduction to Real Time Clocks	260
13.2	Real Time Clock Functional Description	261
13.2.1	Power-on setting	261
13.2.2	RTC count start setting	262
13.2.3	System low power mode switching.....	262
13.2.4	read count register	262
13.2.5	write count register	263
13.2.6	Alarm clock setting	263
13.2.7	1Hz output.....	263
13.2.8	Clock Error Compensation	264
13.3	RTC Interrupt	266
13.3.1	RTC alarm interrupt.....	266
13.3.2	RTC cycle interrupt.....	266
13.4	RTC register description.....	267
13.4.1	Control Register 0 (RTC_CR0).....	268
13.4.2	Control Register 1 (RTC_CR1).....	269
13.4.3	Second Count Register (RTC_SEC)	270
13.4.4	Minute Count Register (RTC_MIN)	270
13.4.5	Hour count register (RTC_HOUR).....	271
13.4.6	Day Count Register (RTC_DAY)	273
13.4.7	Week Count Register (RTC_WEEK)	274
13.4.8	Month Count Register (RTC_MON).....	275

13.4.9	Year Count Register (RTC_YEAR)	275
13.4.10	Minute Alarm Register (RTC_ALMMIN)	276
13.4.11	Hour Alarm register (RTC_ALMHOUR).....	276
13.4.12	Week Alarm Register (RTC_ALMWEEK)	277
13.4.13	Clock Error Compensation Register (RTC_COMPEN)	278
14	Watchdog Timer (WDT)	280
14.1	Introduction to WDT	280
14.2	WDT Functional Description	280
14.2.1	Interrupt generated after WDT overflow	280
14.2.2	Reset after WDT overflow	281
14.3	WDT register description	282
14.3.1	WDT Clear Control Register (WDT_RST).....	282
14.3.2	WDT_CON register	283
15	General Synchronous Asynchronous Transceiver (UART).....	284
15.1	Overview	284
15.2	Structural block diagram	284
15.3	Main characteristics	284
15.4	Functional description	285
15.4.1	Operating mode.....	285
15.4.2	Baud rate generation.....	290
15.5	Frame error detection.....	295
15.6	Multi-machine communication	295
15.7	Automatic Address Recognition.....	295
15.8	Transceiver buffer.....	296
15.8.1	Receive buffer.....	296
15.8.2	Send cache	296
15.9	Register	297
15.9.1	Data Register (UARTx_SBUF)	297
15.9.2	Control Register (UARTx_SCON).....	298
15.9.3	Address Register (UARTx_SADDR)	299
15.9.4	Address Mask Register (UARTx_SADEN).....	299
15.9.5	Flag Register (UARTx_ISR).....	300
15.9.6	Flag Clear Register (UARTx_ICR)	300
16	Low Power Synchronous Asynchronous Transceiver (LPUART)	301
16.1	Overview	301
16.2	Structural block diagram	301
16.3	Main characteristics	302
16.4	Functional description	302

16.4.1	Configuration Clock and Transmit Clock.....	302
16.4.2	Operating mode.....	302
16.4.3	Baud rate generation.....	307
16.5	Frame error detection.....	307
16.6	Multi-machine communication	307
16.7	Automatic Address Recognition.....	308
16.8	Transceiver buffer.....	309
16.8.1	Receive buffer.....	309
16.8.2	Send cache	309
16.9	Register.....	310
16.9.1	Data Register (LPUART_SBUF).....	310
16.9.2	Control Register (LPUART_SCON)	311
16.9.3	Address Register (LPUART_SADDR).....	312
16.9.4	Address Mask Register (LPUART_SADEN).....	312
16.9.5	Interrupt Flag Bit Register (LPUART_ISR).....	313
16.9.6	Interrupt Flag Bit Clear Register (LPUART_ICR)	313
17	I2C -bus (I2C).....	314
17.1	Introduction.....	314
17.2	Main characteristics	314
17.3	Protocol description.....	314
17.3.1	Data transmission on the I2C bus.....	314
17.3.2	Acknowledgment on the I2C bus	316
17.3.3	Arbitration on the I2C bus.....	316
17.4	Functional description	318
17.4.1	Serial clock generator.....	319
17.4.2	Input filter	319
17.4.3	Address comparator.....	319
17.4.4	Response flag	320
17.4.5	Interrupt generator	320
17.4.6	Operating mode.....	320
17.4.7	Status code expression.....	326
17.5	Programming example	328
17.5.1	Master sending example.....	328
17.5.2	Master receive example.....	328
17.5.3	Slave receiving example.....	329
17.5.4	Slave sending example.....	330
17.6	Register description.....	331
17.6.1	I2C Baud Rate Counter Enable Register (I2C_TMRUN)	331

17.6.2	I2C Baud Rate Counter Configuration Register (I2C_TM)	332
17.6.3	I2C Configuration Register (I2C_CR)	332
17.6.4	I2C data register (I2C_DATA)	333
17.6.5	I2C Address Register (I2C_ADDR)	333
17.6.6	I2C Status Register (I2C_STAT)	334
18	Serial Peripheral Interface (SPI)	335
18.1	Introduction to SPI	335
18.2	Main Features of SPI	335
18.3	SPI Functional Description	335
18.3.1	SPI master mode	335
18.3.2	SPI slave mode	336
18.3.3	SPI data frame format	337
18.3.4	SPI Status Flags and Interrupts	339
18.3.5	SPI multi-machine system configuration instructions	339
18.3.6	SPI pin configuration description	341
18.4	SPI programming example	341
18.4.1	SPI master sending example	341
18.4.2	SPI master receive example	342
18.4.3	SPI slave sending example	343
18.4.4	SPI Slave Receive Example	343
18.5	SPI register description	344
18.5.1	SPI Configuration Register (SPI_CR)	345
18.5.2	SPI Chip Select Configuration Register (SPI_SSN)	346
18.5.3	SPI Status Register (SPI_STAT)	346
18.5.4	SPI data register (SPI_DATA)	347
19	Clock Trim Module (CLKTRIM)	348
19.1	Introduction to CLK_TRIM	348
19.2	CLK_TRIM main features	348
19.3	CLK_TRIM function description	349
19.3.1	CLK_TRIM calibration mode	349
19.3.2	CLK_TRIM monitoring mode	350
19.4	CLK_TRIM register description	351
19.4.1	Configuration Register (CLKTRIM_CR)	352
19.4.2	Reference counter initial value configuration register (CLKTRIM_REFCON)	352
19.4.3	Reference Counter Value Register (CLKTRIM_REFCNT)	353
19.4.4	Calibration Counter Value Register (CLKTRIM_CALCNT)	353
19.4.5	Interrupt Flag Register (CLKTRIM_IFR)	354
19.4.6	Interrupt flag clear register (CLKTRIM_ICLR)	354

19.4.7	Calibration Counter Overflow Value Configuration Register (CLKTRIM_CALCON) ...	355
20	Cyclic redundancy check (CRC)	356
20.1	Overview	356
20.2	Main characteristics	356
20.3	Functional description	356
20.3.1	Operating mode.....	356
20.3.2	Encoding.....	356
20.3.3	Write bit width	356
20.4	Programming example	357
20.4.1	CRC-16 encoding mode	357
20.4.2	CRC-16 check mode.....	357
20.5	Registerdescription.....	358
20.5.1	Register list.....	358
20.5.2	Results register (CRC_RESULT)	358
20.5.3	Data register (CRC_DATA)	359
21	Analog-to-digital converter (ADC)	360
21.1	Module Introduction.....	360
21.2	ADC block diagram.....	360
21.3	Conversion Timing and Conversion Speed	361
21.4	Single conversion mode	361
21.5	Continuous Conversion Mode	363
21.6	Continuous Conversion Accumulation Mode.....	364
21.7	ADC conversion result comparison.....	366
21.8	ADC Interrupt	367
21.9	Measure ambient temperature using a temperature sensor	367
21.10	ADC module registers.....	369
21.10.1	ADC Configuration Register 0 (ADC_CR0)	370
21.10.2	ADC Configuration Register 1 (ADC_CR1)	371
21.10.3	ADC Configuration Register 2 (ADC_CR2)	373
21.10.4	ADC channel 0 conversion result (ADC_result0)	374
21.10.5	ADC channel 1 conversion result (ADC_result1)	374
21.10.6	ADC channel 2 conversion result (ADC_result2)	375
21.10.7	ADC channel 3 conversion result (ADC_result3)	375
21.10.8	ADC channel 4 conversion result (ADC_result4)	376
21.10.9	ADC channel 5 conversion result (ADC_result5)	376
21.10.10	ADC channel 6 conversion result (ADC_result6)	377
21.10.11	ADC channel 7 conversion result (ADC_result7)	377
21.10.12	ADC channel 8 conversion result (ADC_result8)	378

21.10.13	Accumulated value of ADC conversion result (ADC_result_acc).....	378
21.10.14	ADC Compare Upper Threshold (ADC_HT)	379
21.10.15	ADC Compare Lower Threshold (ADC_LT)	379
21.10.16	ADC Interrupt Flag Register (ADC_IFR)	380
21.10.17	ADC Interrupt Clear Register (ADC_ICLR).....	380
21.10.18	ADC result (ADC_result)	381
22	Analog Comparator (VC).....	382
22.1	Introduction to Analog Voltage Comparator VC	382
22.2	Voltage Comparator Block Diagram	383
22.3	Setup / Response Time	383
22.4	Filter time	384
22.5	Hysteresis function.....	384
22.6	VC register.....	385
22.6.1	VC Configuration Register (VC_CR)	386
22.6.2	VC0 Configuration Register (VC0_CR)	387
22.6.3	VC1 Configuration Register (VC1_CR)	388
22.6.4	VC0 Output Configuration Register (VC0_OUT_CFG).....	389
22.6.5	VC1 Output Configuration Register (VC1_OUT_CFG).....	390
22.6.6	VC Interrupt Register (VC_IFR)	391
23	Low Voltage Detector (LVD)	392
23.1	Introduction to LVD	392
23.2	LVD block diagram.....	392
23.3	Digital filtering.....	393
23.4	Hysteresis function	393
23.5	Configuration example	394
23.5.1	LVD configured as low voltage reset.....	394
23.5.2	LVD configured as voltage change interrupt.....	394
23.6	LVD register	395
23.6.1	LVD configuration register (LVD_CR)	396
23.6.2	LVD Interrupt Register (LVD_IFR).....	398
24	Simulate other registers	399
24.1	BGR Configuration Register (BGR_CR).....	399
25	SWD debug interface	400
25.1	SWD debugging additional functions.....	400
25.2	ARM® Reference Documentation.....	401
25.3	Debug port pins.....	402
25.3.1	SWD port pin.....	402
25.3.2	SW-DP pin assignment.....	402

25.3.3	Internal pull-up on SWD pin	402
25.4	SWD port	402
25.4.1	Introduction to SWD Protocol	402
25.4.2	SWD protocol sequence	403
25.4.3	SW-DP state machine (reset, idle state, ID code)	404
25.4.4	DP and AP read / write access	404
25.4.5	SW-DP register	405
25.4.6	SW-AP Register	405
25.5	Kernel debugging	406
25.6	BPU (Breakpoint Unit)	406
25.6.1	BPU function	406
25.7	DWT (Data Watchpoint)	406
25.7.1	DWT function	406
25.7.2	DWT Program Counter Sample Register	406
25.8	MCU debug group (DBG)	407
25.8.1	Debug support for low power modes	407
25.8.2	Debug support for timers, watchdog	407
25.9	Debug mode module working state control (DEBUG_ACTIVE)	408
26	Device electronic signature	409
26.1	Product Unique Identifier (UID) Register (80 bits)	409
26.2	Product Model Register	409
26.3	FLASH capacity register	410
26.4	RAM capacity register	410
26.5	Pin Count Register	410
27	Appendix A SysTick Timer	411
27.1	Introduction to SysTick Timers	411
27.2	Set SysTick	411
27.3	SysTick register	412
27.3.1	SysTick Control and Status Register (CTRL)	412
27.3.2	SysTick reload register (LOAD)	412
27.3.3	SysTick Current Value Register (VAL)	412
28	Appendix B Document Conventions	413
28.1	List of register-related abbreviations	413
28.2	Glossary	413
	Version revision history	414

Table Index

Table 2-1	Runnable module diagram in run mode	28
Table 2-2	Runnable Module Diagram in Sleep Mode	30
Table 2-3	Runnable module diagram in deep sleep mode	31
Table 5-1	Cortex-M0+ Processor Exception List.....	60
Table 5-2	Correspondence between external interrupt and NVIC interrupt input	64
Table 6-1	Port State Truth Table	80
Table 6-2	Port multiplexing table	83
Table 7-1	FLASH erasing time parameters at different frequencies.....	148
Table 9-1	Base TimerRegister List.....	167
Table 10-1	LPTimer register list	174
Table 11-1	PCA comparison capture function module setup.....	186
Table 11-2	PCA register list.....	188
Table 12-1	Basic Features of Advanced Timer	195
Table 12-2	Advanced Timer port list	195
Table 12-3	AOS source selection.....	221
Table 12-4	Port trigger selection.....	222
Table 12-5	Advanced Timer register list.....	224
Table 13-1	Basic characteristics of RTC.....	260
Table 13-2	RTC register list	267
Table 14-1	WDT register list.....	282
Table 15-1	Mode0/1/2/3 Data Structure	285
Table 16-1	Mode0/1/2/3 Data Structure	303
Table 17-1	I2C clock signal baud rate	319
Table 17-2	I2C status code expression	326
Table 17-3	Register List	331
Table 18-1	SPI pin configuration description table.....	341
Table 18-2	SPI register list	344
Table 18-3	Master Mode Baud Rate Selection.....	345
Table 19-1	Register List	351
Table 21-1	ADC register	369
Table 22-1	VC register	385
Table 23-1	LVD register.....	395

Graph Index

Figure 1-1	System Architecture Diagram.....	22
Figure 1-2	Schematic Diagram of Address Area division	24
Figure 2-1	Control Mode Block Diagram	26
Figure 3-1	Clock Control Module Block Diagram	34
Figure 3-2	Schematic diagram of crystal oscillator clock startup	36
Figure 3-3	Schematic diagram of clock switching.....	38
Figure 3-4	Clock Calibration Schematic	41
Figure 4-1	Reset Source Schematic	55
Figure 5-1	Only the upper two bits of the priority register are used	60
Figure 5-2	Interrupt Vector Table	61
Figure 5-3	Interrupt active and pending states.....	62
Figure 5-4	Interrupt pending status is cleared and then reasserted	63
Figure 5-5	When the interrupt exits, if the interrupt request remains high, it will cause the interrupt processing to be executed again	63
Figure 5-6	Interrupt pending interrupts generated during interrupt processing can also be acknowledged.....	63
Figure 5-7	Interrupt Structure Diagram	66
Figure 6-1	Port Circuit Diagram	81
Figure 6-2	AHB BusPort Changes With System Clock	81
Figure 6-3	Read port pin data synchronization diagram	82
Figure 7-1	Memory Sector Division	144
Figure 9-1	Base Timer Block Diagram.....	162
Figure 9-2	Timer Mode 1 Block Diagram.....	163
Figure 9-3	Timer Mode 2 Block Diagram.....	164
Figure 9-4	Mode 1 Timing Diagram.....	164
Figure 9-5	Mode 2 Timing Diagram (prescaler set to 2).....	165
Figure 10-1	LPTimer block diagram	171
Figure 10-2	LPTimer Mode 1	172
Figure 10-3	LPTimer Mode 2	172
Figure 11-1	Overall block diagram of PCA	177
Figure 11-2	PCA Counter Block Diagram	179
Figure 11-3	PCA Capture Functional Block Diagram	181
Figure 11-4	PCA Comparison Functional Block Diagram	183
Figure 11-5	PCA WDT Functional Block Diagram	184
Figure 11-6	PCA PWM Functional Block Diagram	185
Figure 11-7	PCA PWM Output Waveform	186

Figure 12-1	Advanced Timer Block Diagram.....	196
Figure 12-2	Sawtooth Waveform (Counting Up).....	197
Figure 12-3	Triangle Waveform	197
Figure 12-4	Compare output action	198
Figure 12-5	Capturing Input Actions	199
Figure 12-6	Filter function of the capture input port	201
Figure 12-7	Software Synchronization Action	202
Figure 12-8	Hardware Synchronous Action	204
Figure 12-9	Single-buffer mode comparison output timing	205
Figure 12-10	PWM Spread Spectrum Output Schematic.....	207
Figure 12-11	CHA output PWM wave	207
Figure 12-12	Software setting GCMBR Complementary PWM wave output in triangular wave A mode	208
Figure 12-13	Triangular wave B mode hardware setting GCMBR Complementary PWM wave output (symmetrical dead zone)	209
Figure 12-14	6-phase PWM wave.....	210
Figure 12-15	Three-phase complementary PWM wave output with dead time in triangular wave A mode.....	211
Figure 12-16	Basic counting action in position mode	212
Figure 12-17	Phase difference counting action setting in position mode (1 time).....	213
Figure 12-18	Phase difference counting action setting in position mode (2 time).....	213
Figure 12-19	Phase difference counting action setting in position mode (4 time).....	213
Figure 12-20	Direction counting action in position mode	214
Figure 12-21	Phase Z counting action in revolution mode.....	214
Figure 12-22	Position counter output counting action in revolution mode.....	215
Figure 12-23	Z-phase counting and position counter output mixed counting action in revolution mode	215
Figure 12-24	Revolution counting mode - Mixed counting Z-phase shielding action example 1.....	216
Figure 12-25	Revolution counting mode - Mixed counting Z-phase shielding action example 2.....	217
Figure 12-26	Cycle Interval Valid Request Signal Action.....	217
Figure 12-27	Schematic diagram of port braking and software braking	219
Figure 12-28	Output same high and same low brake schematic diagram	220
Figure 12-29	VC brake control diagram	220
Figure 12-30	Timer4/5/6 interrupt selection	221
Figure 13-1	RTC Block Diagram	261
Figure 14-1	Overall block diagram of WDT	280
Figure 15-1	UART Block Diagram	284
Figure 15-2	Mode0 sending data	286

Figure 15-3	Mode0 receiving data	286
Figure 15-4	Mode1 sending data	287
Figure 15-5	Mode1 receiving data	287
Figure 15-6	Mode2 sending data	288
Figure 15-7	Mode2 receiving data	288
Figure 15-8	Mode3 Send Data	289
Figure 15-9	Mode3 Receive Data	289
Figure 15-10	Receive Cache	296
Figure 16-1	LPUART Block Diagram	301
Figure 16-2	Mode0 sending data	304
Figure 16-3	Mode0 receiving data	304
Figure 16-4	Mode1 sending data	305
Figure 16-5	Mode1 receiving data	305
Figure 16-6	Mode2 sending data	306
Figure 16-7	Mode2 receiving data	306
Figure 16-8	Mode3 Send Data	306
Figure 16-9	Mode3 Receive Data	307
Figure 16-10	Receive Cache	309
Figure 16-11	Send Cache.....	309
Figure 17-1	I2C transfer protocol	315
Figure 17-2	START and STOP conditions	315
Figure 17-3	Bit transfer on the I2C bus.....	316
Figure 17-4	Acknowledgment signal on I2C bus	316
Figure 17-5	Arbitration on the I2C bus.....	317
Figure 17-6	I2C function block diagram.....	318
Figure 17-7	Data synchronization diagram of master sending mode	320
Figure 17-8	I2C master sending state diagram.....	321
Figure 17-9	Data synchronization diagram of main receiving mode.....	322
Figure 17-10	I2C master receiving state diagram.....	322
Figure 17-11	Slave Receive Mode Data Synchronization Diagram.....	323
Figure 17-12	Slave receiving state diagram	323
Figure 17-13	Slave mode data synchronization diagram.....	324
Figure 17-14	I2C Slave Transmit State Diagram	324
Figure 17-15	I2C General Call State Diagram	325
Figure 18-1	Slave receiving diagram	337
Figure 18-2	Slave sending diagram	337
Figure 18-3	Host Mode Frame Format.....	338
Figure 18-4	Data frame format when slave CPHA is 0	338

Figure 18-5	Data frame format when slave CHPA is 1	338
Figure 18-6	Schematic diagram of SPI multi-master/multi-slave system.....	340
Figure 21-1	ADC block diagram	360
Figure 21-2	ADC conversion timing diagram	361
Figure 21-3	Example Of ADC Continuous Conversion Process	363
Figure 21-4	ADC Continuous Conversion Accumulation Process Example	365
Figure 22-1	VC Frame Diagram.....	383
Figure 22-2	VC Filter Response Time	384
Figure 22-3	VC hysteresis function	384
Figure 23-1	LVD Block Diagram	392
Figure 23-2	LVD Filtered Output.....	393
Figure 23-3	LVD Hysteretic Response	393
Figure 25-1	Debug Support Block Diagram	401

Overview

The HC32L110 series is an ultra-low power consumption, Low Pin Count, wide voltage operating range MCU designed to extend battery life in portable measurement systems. Integrating 12-bit 1Msps high-precision SARADC and integrating rich communication peripherals such as comparator, multi-channel UART, SPI, I2C, etc., it has the characteristics of high integration, high anti-interference, high reliability and ultra-low power consumption. The core of this product adopts the Cortex-M0+ core, cooperates with mature Keil & IAR debugging and development software, supports C language, assembly language, and assembly instructions.

Ultra-low power MCU typical applications

- Sensor applications, IoT applications
- Smart Transportation, Smart City, Smart Home
- Fire detectors, smart door locks, wireless monitoring and other smart sensor applications
- All kinds of portable devices that are battery-powered and power-hungry, etc.

About this manual

This manual mainly introduces the function, operation and usage of the chip. For chip specifications, please refer to the corresponding "Datasheet".

1 System Structure

1.1 Overview

This product system consists of the following parts:

- 1 AHB bus Master:
 - Cortex-M0+
- 4 AHB bus slaves:
 - FLASH memory
 - SRAM memory
 - AHB0, AHB to APB Bridge, including all APB interface peripherals
 - AHB1, contains all AHB interface peripherals

The entire system bus structure is realized by multi-level AHB-lite bus interconnection. As shown below:

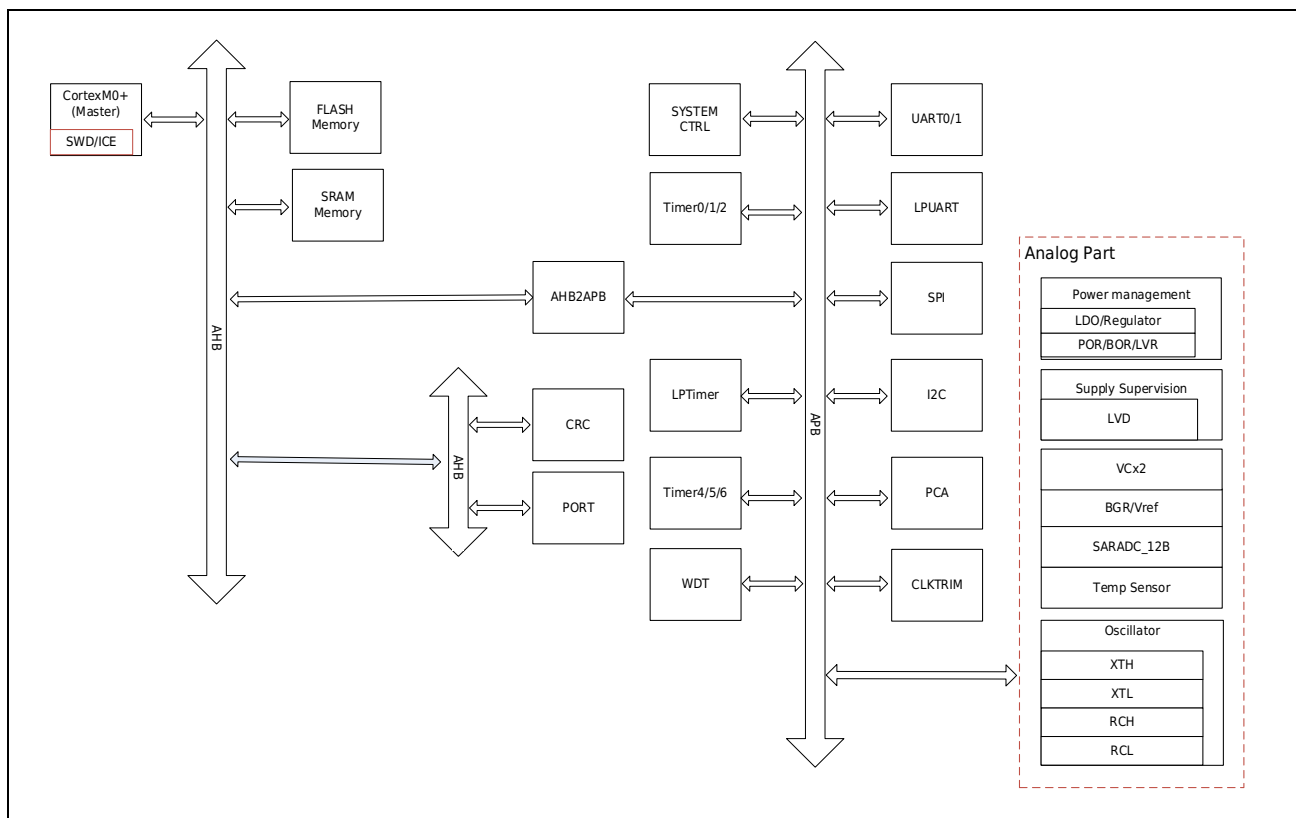
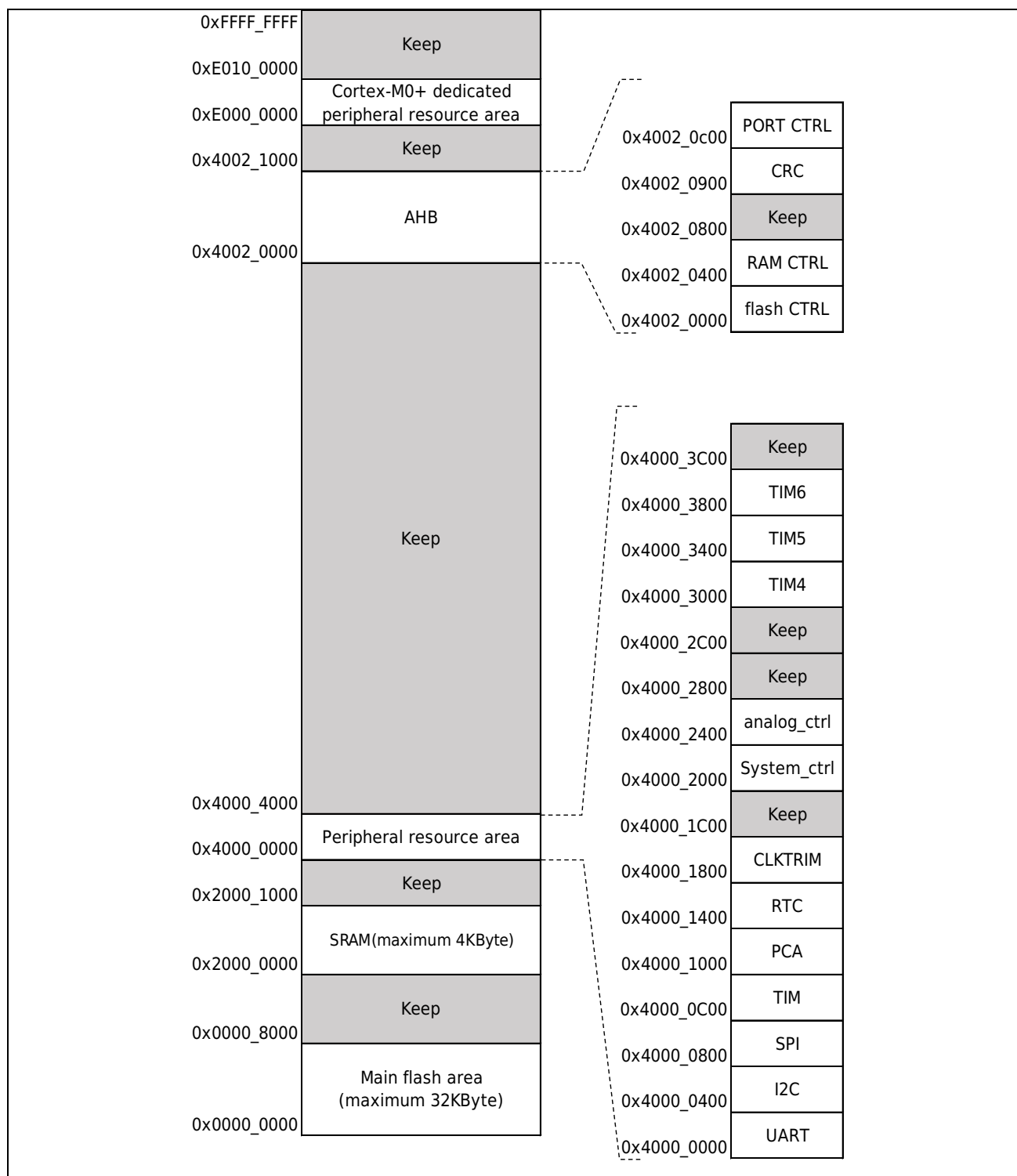


Figure 1-1 System Architecture Diagram

1.2 System address division

The address area division of the entire HC32L110 system is shown in the figure below:



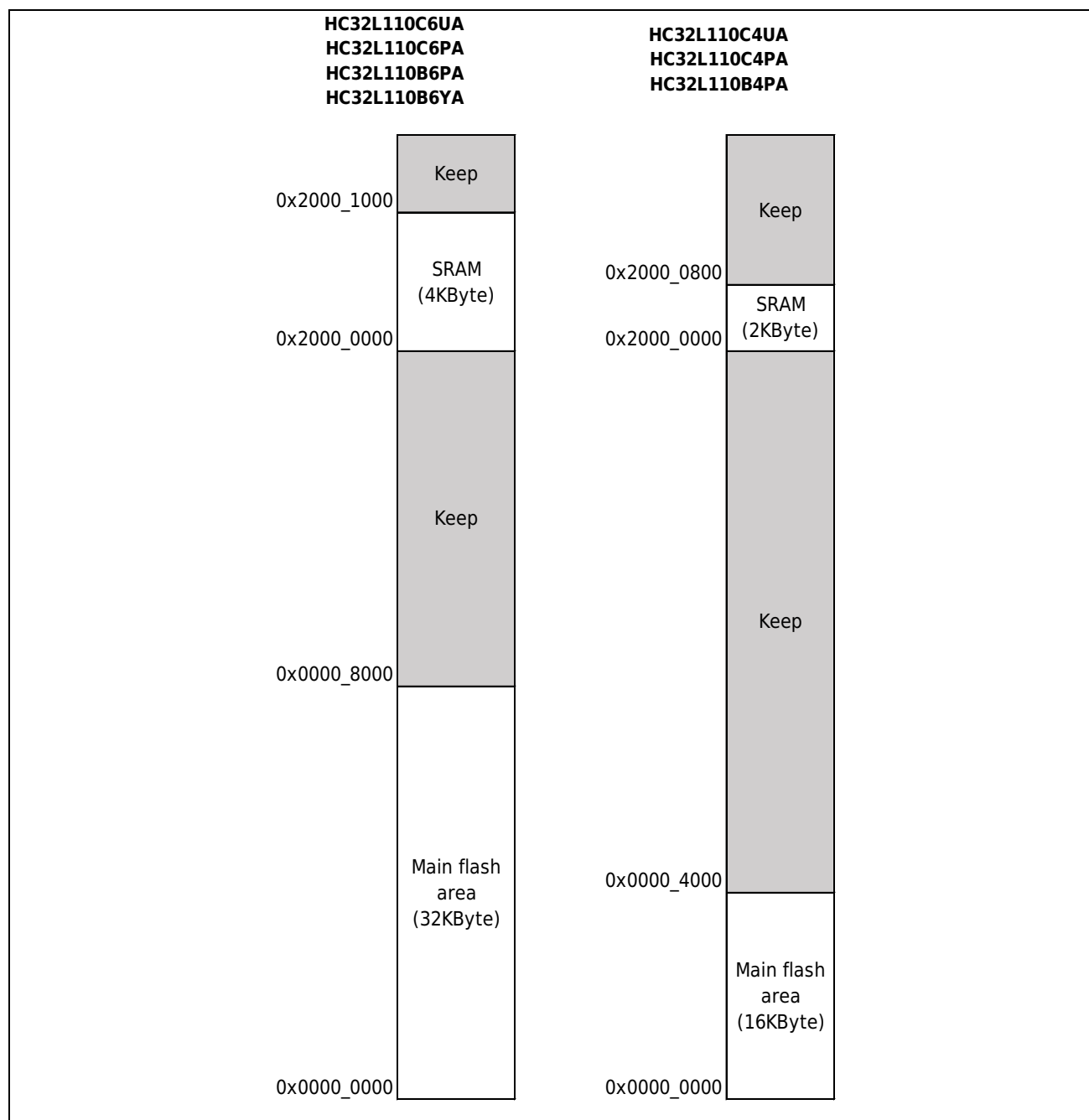


Figure 1-2 Schematic Diagram of Address Area division

1.3 Memory and Module Address Assignment

Boundary Address	Size	Memory Area	Description
0x0000_0000 - 0x0000_7FFF	32kByte	FLASH Memory	
0x0000_8000 - 0x0010_0BFF	-	Reserved	
0x0010_0C00 - 0x0010_0C3B	60Byte	Trim Data	
0x0010_0C3C - 0x0010_0E6F	-	Reserved	
0x0010_0E70 - 0x0010_0E7F	16Byte	UID	
0x0010_0E80 - 0x1FFF_FFFF	-	Reserved	
0x2000_0000 - 0x2000_0FFF	4kByte	SRAM Memory	
0x2000_1000 - 0x3FFF_FFFF	-	Reserved	
0x4000_0000 - 0x4000_00FF	256Byte	UART0	
0x4000_0100 - 0x4000_01FF	256Byte	UART1	
0x4000_0200 - 0x4000_02FF	256Byte	LPUART	
0x4000_0300 - 0x4000_03FF	-	Reserved	
0x4000_0400 - 0x4000_07FF	1kByte	I2C	
0x4000_0800 - 0x4000_0BFF	1kByte	SPI	
0x4000_0C00 - 0x4000_0FFF	1kByte	Timer0/1/2/WDT/LPTimer	
0x4000_1000 - 0x4000_13FF	1kByte	PCA	
0x4000_1400 - 0x4000_17FF	1kByte	RTC	
0x4000_1800 - 0x4000_1BFF	1kByte	CLKTRIM	
0x4000_1C00 - 0x4000_1FFF	-	Reserved	
0x4000_2000 - 0x4000_23FF	1kByte	SYSTEMCTRL	
0x4000_2400 - 0x4000_27FF	1kByte	ANALOGCTRL	
0x4000_2800 - 0x4000_2FFF	-	Reserved	
0x4000_3000 - 0x4000_33FF	1kByte	Timer4	
0x4000_3400 - 0x4000_37FF	1kByte	Timer5	
0x4000_3800 - 0x4000_3BFF	1kByte	Timer6	
0x4000_3C00 - 0x4001_FFFF	-	Reserved	
0x4002_0000 - 0x4002_03FF	1kByte	FLASH CTRL	
0x4002_0400 - 0x4002_07FF	1kByte	RAM CTRL	
0x4002_0800 - 0x4002_08FF	256Byte	Reserved	
0x4002_0900 - 0x4002_0BFF	768Byte	CRC	
0x4002_0C00 - 0x4002_0FFF	1kByte	PORT CTRL	

2 Operating mode

The power management module of this product is responsible for managing the switching between various working modes of this product. And control the working state of each functional module in each working mode. VCC) of this product is 1.8 ~ 5.5V.

This product has the following working modes:

- (1) Active Mode: CPU running, peripheral function modules running.
- (2) Sleep mode: The CPU stops running, and the peripheral function modules run.
- (3) Deep sleep mode: The CPU stops running, and the high-speed clock stops running.

From run mode, other low-power modes can be entered by executing a software program. From various other low-power modes, it is possible to return to run mode through an interrupt trigger.

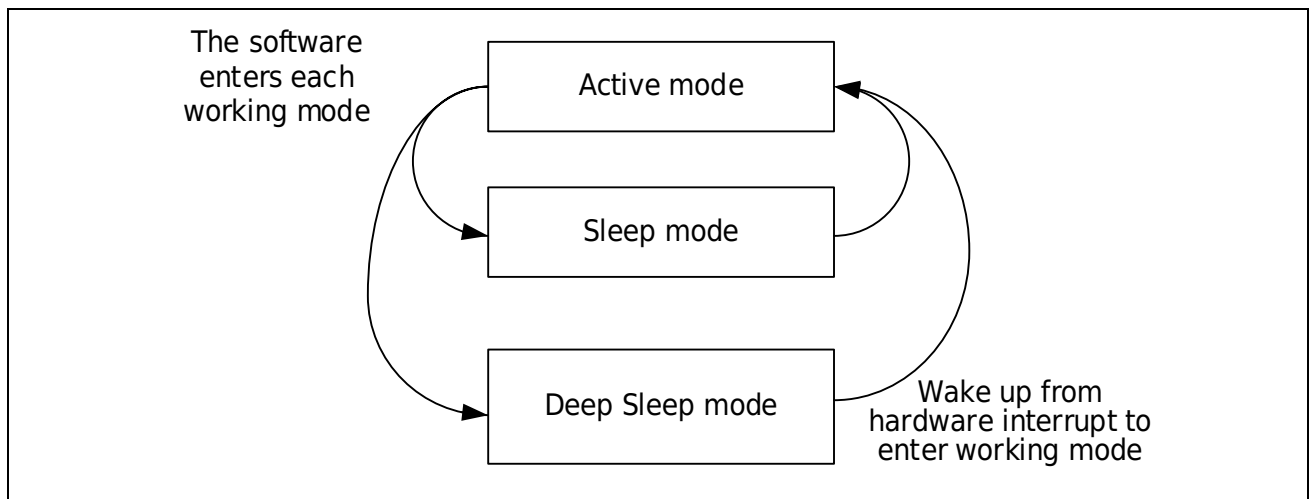


Figure 2-1 Control Mode Block Diagram

In each mode, the CPU can respond to all interrupt types.

	Interrupt source	Operating mode	Sleep mode	Deep sleep mode
[0]	GPIO_P0	✓	✓	✓
[1]	GPIO_P1	✓	✓	✓
[2]	GPIO_P2	✓	✓	✓
[3]	GPIO_P3	✓	✓	✓
[6]	UART0	✓	✓	
[7]	UART1	✓	✓	
[8]	LPUART	✓	✓	✓
[10]	SPI	✓	✓	
[12]	I2C	✓	✓	
[14]	Timer0	✓	✓	
[15]	Timer1	✓	✓	
[16]	Timer2	✓	✓	
[17]	LPTimer	✓	✓	✓
[18]	Timer4	✓	✓	
[19]	Timer5	✓	✓	
[20]	Timer6	✓	✓	
[21]	PCA	✓	✓	
[22]	WDT	✓	✓	✓
[23]	RTC	✓	✓	✓
[24]	ADC	✓	✓	
[26]	VC0	✓	✓	✓
[27]	VC1	✓	✓	✓
[28]	LVD	✓	✓	✓
[30]	RAM FLASH	✓	✓	
[31]	CLKTRIM	✓	✓	✓

In each mode, the product responds to all reset types.

	reset source	Operating mode	Sleep mode	Deep sleep mode
[0]	Power-on brown-out reset POR	✓	✓	✓
[1]	External Reset Pin reset	✓	✓	✓
[2]	LVD reset	✓	✓	✓
[3]	WDT reset	✓	✓	✓
[4]	PCA reset	✓	✓	
[5]	Cortex-M0+ LOCKUP hardware reset	✓		
[6]	Cortex-M0+ SYSRESETREQ software reset	✓		

2.1 Operating mode

The active mode of this product:

The microcontroller MCU is in the running state after the 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.

Table 2-1 Runnable module diagram in run mode

Active Mode(Active Mode):		
Cortex-M0+	SWD	XTH
FLASH	UART0-1	RCH
RAM	SPI	ADC
TIM0-2	I2C	RESET
TIM4-6	CRC	POR/BOR
PCA	XTL	LVD
LPUART	RCL	VC
LPTIM	RTC	CLKTRIM
GPIO	WDT OSC	WDT

Several ways to reduce chip power consumption in run mode:

- 1) (HCLK, PCLK) can be slowed down by programming the prescaler registers (SYSCLK0.AHB_CLK_DIV, SYSCLK0.APB_CLK_DIV). The prescaler can also be used to slow down the clocks to peripherals before entering Sleep mode.
- 2) PERI_CLKx) of unused peripherals to reduce power consumption.
- 3) In run mode, turn off unused peripheral clocks (PERI_CLKx) to reduce power consumption, and put the system into sleep mode to reduce power consumption even more, and turn off unused peripheral clocks (PERI_CLKx) before executing WFI instructions.
- 4) Use low-power mode instead of sleep mode, because the wake-up time of this product is extremely short (~4uS), which can also meet the real-time response requirements of the system.

2.2 Sleep mode

This product sleep mode

Use the WFI command to enter the sleep mode. In the sleep mode, the CPU stops running, but the clock module, system clock, NVIC interrupt processing and peripheral functional modules can still work.

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

How to enter sleep mode:

Enter sleep state by executing WFI instruction. Depending on the value of the SLEEPONEXIT bit in the Cortex™-M0+ System Control Register, there are two options for selecting the sleep mode entry mechanism:

SLEEP-NOW: If the SLEEPONEXIT bit is cleared, the microcontroller enters sleep mode immediately when WFI or WFE is executed.

SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the microcontroller enters sleep mode immediately when the system exits from the lowest priority interrupt handler.

How to exit sleep mode:

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

Note:

- The position of SLEEP-ON-EXIT is 1, and the interrupt will automatically enter sleep after execution, and the program does not need to write `__wfi()`;
- **SLEEP-ON-EXIT** This bit is cleared to 0, `main()` executes `__wfi()` and enters sleeping, interrupt triggers and executes the interrupt program and returns to `main()`, then executes WFI instruction and enters sleeping. Wait for a subsequent interrupt to fire.
- The SLEEP-ON-EXIT bit does not affect the execution of the `__wfi()` instruction. **SLEEP-ON-EXIT = 0:** `main()` enters sleeping after executing `wfi()`, interrupt triggers and returns to `main()` after executing the interrupt program, 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 be woken up.

Table 2-2 Runnable Module Diagram in Sleep Mode

Sleep Mode		
Cortex-M0+	SWD	XTH
FLASH	UART0-1	RCH
RAM	SPI	ADC
TIM0-2	I2C	RESET
TIM4-6	CRC	POR/BOR
PCA	XTL	LVD
LPUART	RCL	VC
LPTIM	RTC	CLKTRIM
GPIO	WDT OSC	WDT

Modules that are grayed out are not working in their current state.

2.3 Deep sleep mode

This product deep sleep mode

Use SLEEPDEEP with the WFI command to enter deep sleep mode. In deep sleep mode, the CPU stops running, the high-speed clock is turned off, the low-speed clock can be configured to run or not, and some low-power peripheral modules can be configured to allow or not. NVIC interrupt processing can still work.

- When the system enters deep sleep mode from the high-speed clock, the high-speed clock is automatically turned off, and the low-speed clock remains in the state before entering deep sleep.
- The system enters the deep sleep mode from the low-speed clock, and since the low-speed clock will not be automatically shut down, it keeps running and enters the sleep mode. Only ARM Cortex-M0+ is not running, other modules are running.
- When the system clock is switched, all clocks will not be automatically turned off, and the corresponding clocks need to be turned off and turned on by software according to power consumption and system requirements.
- When the system enters the deep sleep state, it will not change the port state. Before entering the sleep state, change the IO state to the sleep state as needed. Unused IO pins and unlead IO pins of the package need to be set as inputs and enable pull-up.

How to enter deep sleep mode:

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

SLEEP-NOW: If the SLEEPONEXIT bit is cleared, the microcontroller enters sleep mode immediately when WFI or WFE is executed.

SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the microcontroller enters sleep mode immediately when the system exits from the lowest priority interrupt handler.

How to exit deep sleep mode:

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

For wake-up settings, refer to 3.4 Interrupt Wake-Up Control WIC.

Table 2-3 Runnable module diagram in deep sleep mode

Deep Sleep Mode		
Cortex-M0+	SWD	XTH
FLASH	UART0-1	RCH
RAM	SPI	ADC
TIM0-2	I2C	RESET
TIM4-6	CRC	POR/BOR
PCA	XTL	LVD
LPUART	RCL	VC
LPTIM	RTC	CLKTRIM
GPIO	WDT OSC	WDT

Modules that are **grayed** out are not working in their current state.

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

Address: 0xE000ED10

Reset value: 0x0000 0000

Bit	Marking	Functional description	Read and write
31:5	RESERVED	Keep	
4	SEVONPEND	When set to 1, an event is generated every time a new interrupt is pending, which can be used to wake up the processor if WFE sleep is used	RW
3	RESERVED	Keep	
2	SLEEPDEEP	When set to 1, implement WFI to enter deep sleep, and this product enters Deep sleep mode When set to 0, execute WFI to enter sleep mode, and this product enters sleep/Idle mode	RW
1	SLEEPONEXIT	When set to 1, the processor automatically enters sleep mode (WFI) when exiting exception handling and returning to the program thread When set to 0, the feature is automatically disabled	RW
0	RESERVED	Keep	

After entering deep sleep, there are two options for the system clock after waking up. The clock entering deep sleep is used by default. After the configuration register SYSCTRL0.wakeup_byRCH is set to 1, no matter what clock is before entering deep sleep, the internal high-speed clock RCH will be used after waking up. If you use an external crystal oscillator, this setting can speed up the wake-up of the system.

3 System Controller (SYSCTRL)

3.1 Clock source introduction

The clock control module mainly controls the system clock and the peripheral clock. Different clock sources can be configured as the system clock, different frequency divisions of the system clock can be configured, and peripheral clocks can be enabled or disabled. In addition, in order to ensure the accuracy of the oscillator, the internal clock has a calibration function.

This product supports the following four different clock sources as the system clock:

- Internal high-speed RC clock RCH (output frequency is 4~24MHz)
- Internal low-speed RC clock RCL (38.4K and 32.768K configurable)
- External high-speed crystal oscillator clock XTH
- External low-speed crystal oscillator clock XTL

Note:

- When switching the clock source of the system clock, please strictly follow the operation steps to switch, see chapter 3.2 for details.
- XTL can directly input 32.768KHz clock signal from P14 pin without crystal oscillator. XTH can directly input 4~32MHz clock signal from P01 pin without crystal oscillator. When using XTH external input, P02 pin is prohibited from being used as GPIO, and P02 port needs to be configured in analog state; When using XTL external input, pin P15 is prohibited from being used as GPIO, and port P15 needs to be configured in analog mode.

This product also contains the following two auxiliary clocks:

- Internal low-speed 10K clock; only used by watchdog and CLKTRIM modules.
- Internal 150K clock: only for LVD and VC modules.

The following figure shows the clock architecture of this product:

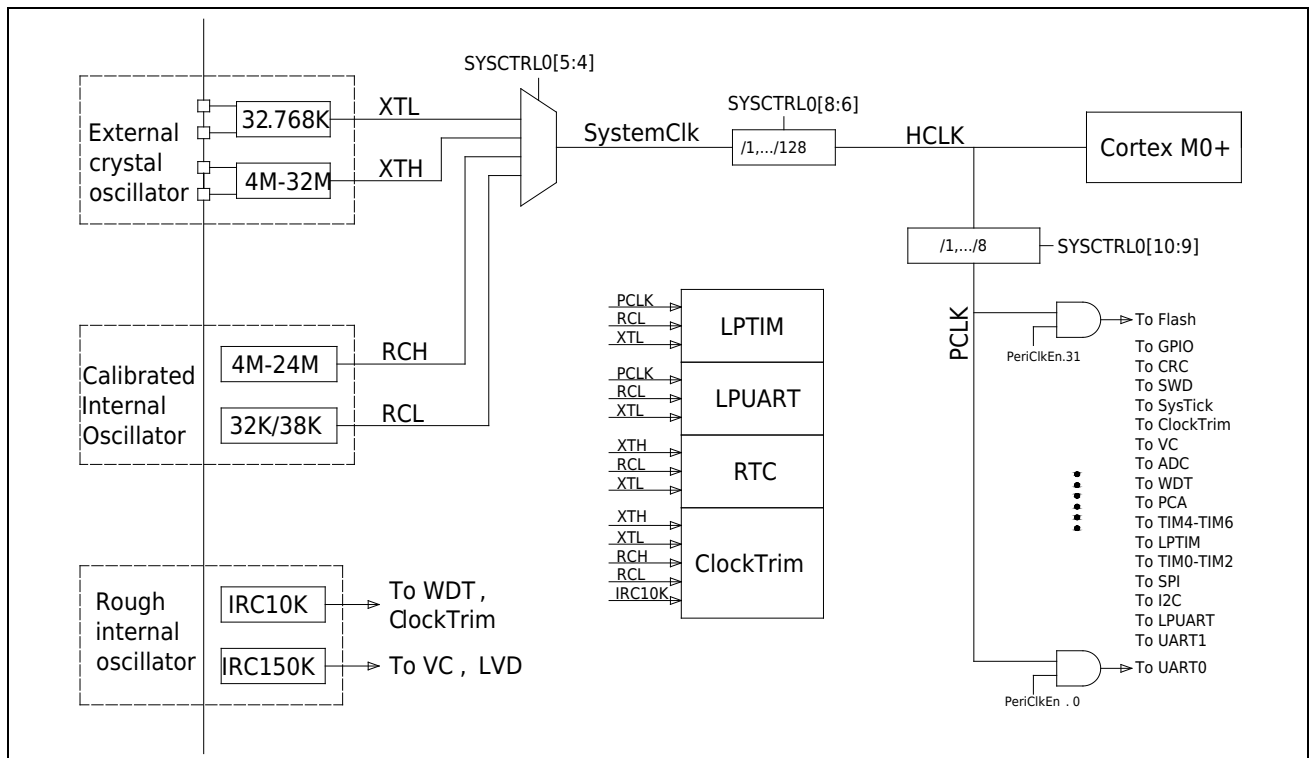


Figure 3-1 Clock Control Module Block Diagram

3.1.1 Internal high speed RC clock RCH

The default clock source after the chip is powered on or reset is the internal high-speed clock with a frequency of 4MHz; when the system enters Deep Sleep, this high-speed clock will be automatically turned off.

The output frequency of RCH can be adjusted by changing the value of the register RCH_CR[10:0]. Every time the register value increases by 1, the output frequency of RCH will increase by about 0.2%, and the total adjustment range is 4~24MHz.

5 frequencies 4 MHz, 8 MHz, 16 MHz, 22.12 MHz, and 24 MHz have been pre-adjusted before leaving the factory; if other frequencies are required, please manually adjust the value of this register.

Changing the RCH output frequency needs to follow a specific change timing, see the chapter on system clock switching for details.

4us from startup to stabilization. In order to respond to interrupts quickly in deep sleep mode, it is recommended to switch the system clock to RCH before entering deep sleep mode.

3.1.2 Internal low speed RC clock RCL

The output frequency of the internal low-speed clock can be selected through the register RCL_CR[9:0]. The available frequencies are 38.4KHz and 32.768KHz. When the system enters Deep Sleep, this low-speed clock will not be automatically turned off, and the ultra-low-power peripheral module can choose RCL as its clock.

3.1.3 External low-speed crystal oscillator clock XTL

The external low-speed crystal oscillator clock requires an external 32.768KHz low-power crystal oscillator, which has ultra-high precision and ultra-low power consumption. When the system enters Deep Sleep, the low-speed clock will not be turned off automatically. Peripheral modules operating in ultra-low power mode can select XTL as their clock.

XTL can also not be connected to the crystal oscillator, and directly input the 32.768KHz clock signal from the P14 pin. The method of inputting the clock signal from P14 is: configure the P14 pin as GPIO input; set SYSCTRL1. EXTL_EN to 1; set SYSCTRL0. XTL_EN to 1.

Note:

- The crystal and its matching devices must meet the relevant requirements of the **low-speed external clock XTL in the electrical characteristics of the data sheet.**

3.1.4 External high-speed crystal oscillator clock XTH

The external high-speed crystal oscillator clock needs an external high-speed crystal oscillator of 4 MHz ~32MHz. When the system enters Deep Sleep, this high-speed clock will be automatically turned off.

XTH can also not be connected to the crystal oscillator, and directly input a 4MHz ~ 32MHz clock signal from the P01 pin. The method of inputting clock signal from P01 is: configure P01 pin as GPIO input; set SYSCTRL1. EXTH_EN to 1; set SYSCTRL0. XTH_EN to 1.

Note:

- The crystal and its matching devices must meet the relevant requirements of the **high-speed external clock XTH in the electrical characteristics of the data sheet.**

3.1.5 Clock start process

The above four clock sources all need a start-up stabilization time. The following figure uses an external XTH as an example to illustrate the start-up stabilization process of the clock.

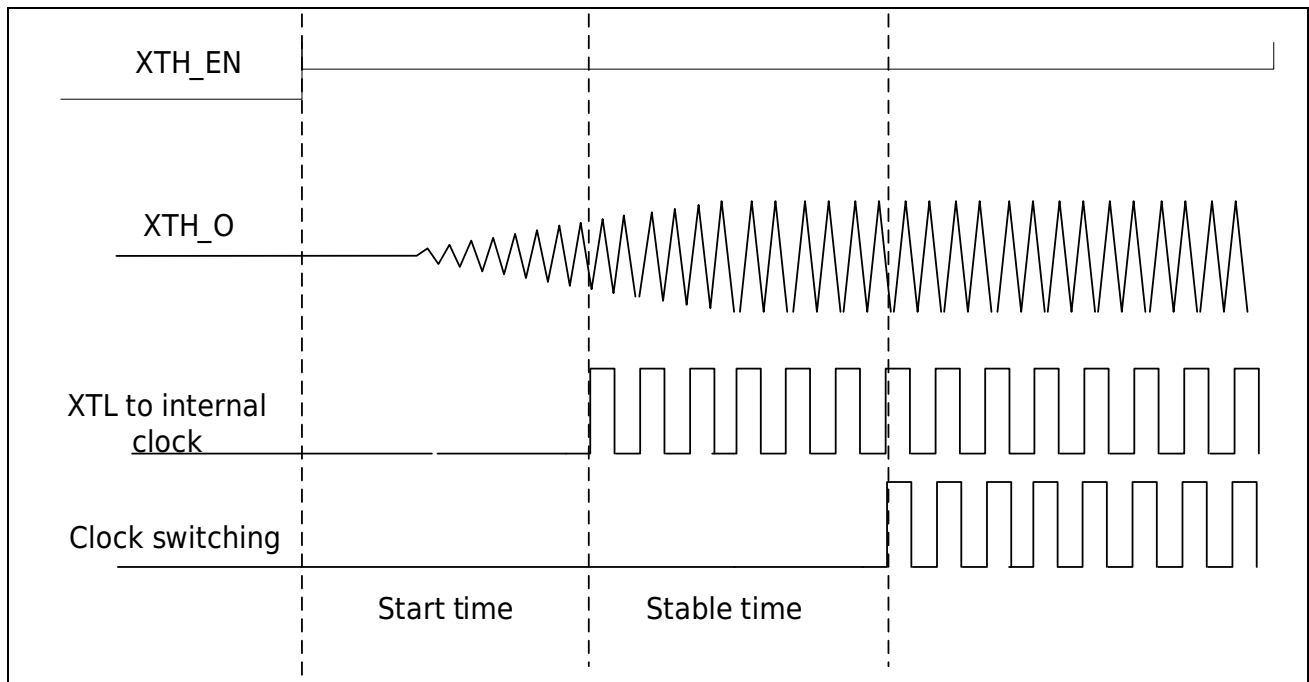


Figure 3-2 Schematic diagram of crystal oscillator clock startup

3.2 System Clock Switching

The clock source of the system clock can be switched among RCH, RCL, XTH, and XTL through SYSCTRL0[5:4]. The clock switching operation must be performed according to the standard clock switching process, otherwise abnormalities may occur. If the frequency of the new clock source is greater than 24MHz, you need to set FLASH_CR.WAIT to 1.

Note: To rewrite the value of FLASH_CR.WAIT, you need to write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence, and then assign a value to FLASH_CR.WAIT. For details, see the FLASH controller chapter.

3.2.1 Standard Clock Switching Process

The operation process is as follows:

Step1: If the new clock source requires an external pin, set the pin to an 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: According to the higher frequency of the current clock source and the new clock source, configure FLASH_CR.WAIT according to the procedures in the Flash controller chapter.

Step5: Wait for the new clock source to output a stable frequency.

Step6: Configure SYCTRL0. Clk_sw4_sel, select the source of the system clock as the new clock source.

Step7: According to the frequency of the new clock source, configure FLASH_CR. WAIT according to the procedures in the Flash controller chapter.

Step8: Turn off the clock source that is no longer used.

3.2.2 Switch from RCH to XTL example

The operation process is as follows:

Step1: Set P1ADS. P1ADS4 and P1ADS. P1ADS4 to 1, and configure P14/P15 pins as analog ports.

Step2: According to the characteristics of crystal oscillator, configure XTL_CR.Driver and XTL_CR.Startup.

Step3: Write 0x5A5A and 0xA5A5 to the 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 the SYCTRL2 register in sequence to enable register rewriting.

Step7: Set SYCTRL0. Clk_sw4_sel to 3, switch the system clock to XTL.

Step8: Write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting.

Step9: Set SYCTRL0. RCH_EN to 0, turn off the RCH oscillator.

3.2.3 Switch from RCH to XTH example

The operation process is as follows:

Step1: Set P0ADS. P0ADS1 and P0ADS. P0ADS1 to 1, and configure P01/P02 pins as analog ports.

Step2: According to the characteristics of crystal oscillator, configure XTH_CR.Driver.

Step3: Set XTH_CR. Startup to 3, choose the longest crystal oscillator stabilization time.

Step4: Write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting.

Step5: Set SYCTRL0. XTH_EN to 1 to enable the crystal oscillator circuit.

Step6: Configure FLASH_CR. WAIT according to crystal oscillator frequency.

Step7: Query and wait for the XTH_CR. Stable flag to become 1, the software delays more than 10ms, and the crystal oscillator outputs a stable clock.

Step8: Write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting.

Step9: Set SYCTRL0. Clk_sw4_sel to 1, switch the system clock to XTH.

Step10: Write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting.

Step11: Set SYCTRL0. RCH_EN to 0, turn off the RCH oscillator.

The following figure is the clock switching timing diagram:

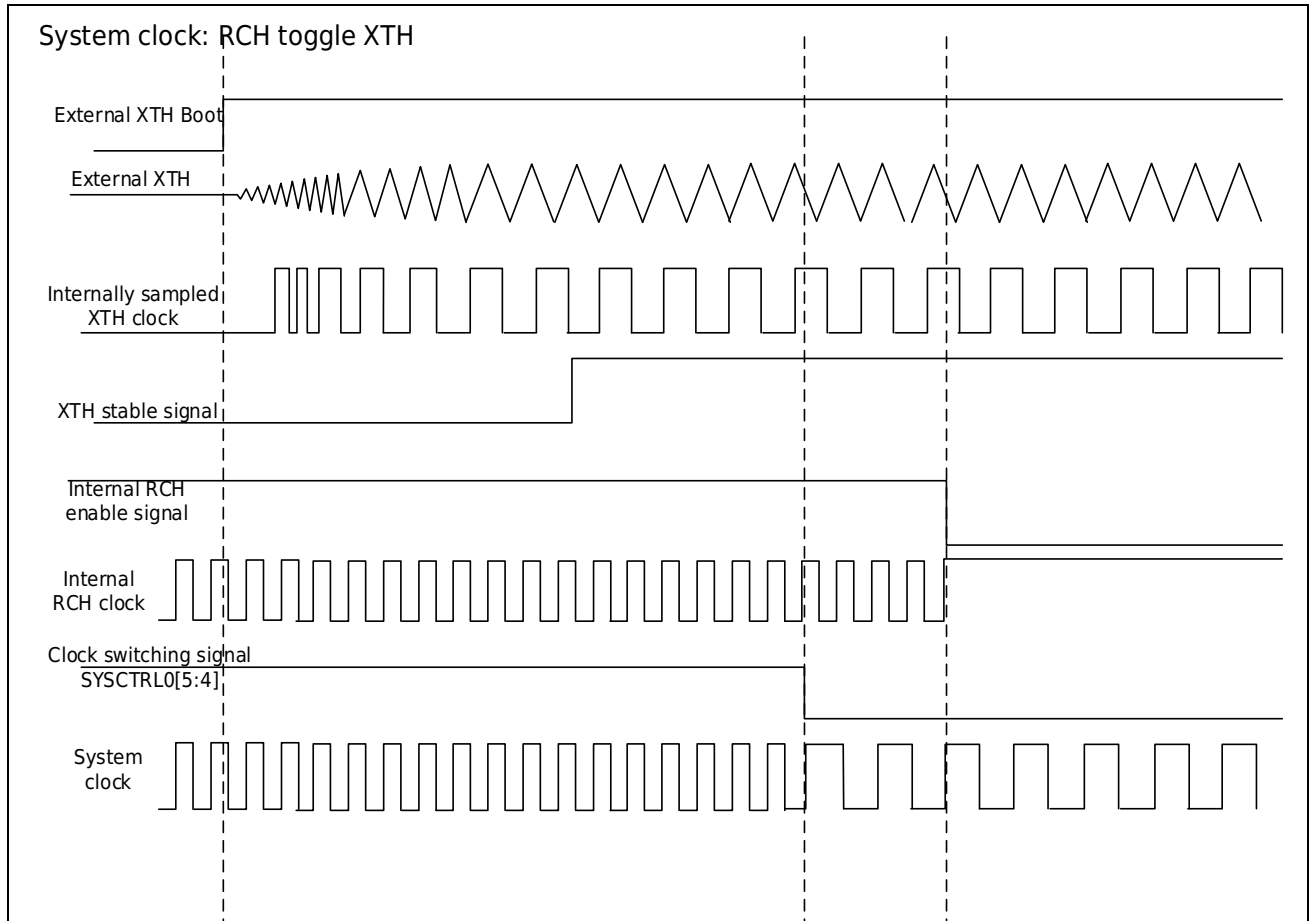


Figure 3-3 Schematic diagram of clock switching

3.2.4 Example of switching from RCL to XTH

The operation process is as follows:

Step1: Set P0ADS. P0ADS1 and P0ADS. P0ADS1 to 1, and configure P01/P02 pins as analog ports.

Step2: According to the characteristics of crystal oscillator, configure XTH_CR.Driver.

Step3: Set XTH_CR. Startup to 3, choose the longest crystal oscillator stabilization time.

Step4: Write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting.

Step5: Set SYCTRL0. XTH_EN to 1 to enable the crystal oscillator circuit.

Step6: Configure FLASH_CR. WAIT according to crystal oscillator frequency.

Step7: Query and wait for the XTH_CR. Stable flag to become 1, the software delays more than 10ms, and the crystal oscillator outputs a stable clock.

Step8: Write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting.

Step9: Set SYCTRL0. Clk_sw4_sel to 1, switch the system clock to XTH.

Step10: Write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting.

Step11: Set SYCTRL0. RCL_EN to 0, turn off the RCL oscillator.

3.2.5 Switch from RCH to RCL example

The operation process is as follows:

Step1: Configure RCL_CR.TRIM and RCL_CR.Startup.

Step2: Write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting.

Step3: Set SYCTRL0. RCL_EN to 1 to enable the RCL oscillator circuit.

Step4: The query waits for the RCL_CR. Stable flag to become 1, and the RCL outputs a stable clock.

Step5: Write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting.

Step6: Set SYCTRL0. Clk_sw4_sel to 2, switch the system clock to RCL.

Step7: If you need to turn off the RCH oscillator, perform subsequent operations: write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting; set SYCTRL0. RCH_EN to 0 to turn off the RCH oscillator.

3.2.6 Example of switching from RCL to RCH

The operation process is as follows:

Step1: Configure RCH_CR.TRIM.

Step2: Write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting.

Step3: Set SYCTRL0. RCH_EN to 1 to enable the RCH oscillator circuit.

Step4: The query waits for the RCH_CR. Stable flag to become 1, and the RCH outputs a stable clock.

Step5: Write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting.

Step6: Set SYCTRL0. Clk_sw4_sel to 0, switch the system clock to RCH.

Step7: If you need to turn off the RCL oscillator, perform subsequent operations: write 0x5A5A and 0xA5A5 to the SYCTRL2 register in sequence to enable register rewriting; set SYCTRL0. RCL_EN to 0 to turn off the RCL oscillator.

3.2.7 Switch between different oscillation frequencies of RCH

There are two schemes for switching between different oscillation frequencies of the RCH.

Scheme 1:

Step1: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step2: Set SYSCTRL0.HCLK_PRS to 0x7.

Step3: Adjust the output frequency of RCH step by step up or down, 4M -> 8M -> 16M -> 24M/22.12M or 24M/22.12M -> 16M -> 8M -> 4M.

Step4: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step 5: Set SYSCTRL0.HCLK_PRS is 0x0.

The example code for switching from 4M to 24M is as follows:

```
MOP_SystemCtrl->SYSCTRL2 = 0X5A5A;
MOP_SystemCtrl->SYSCTRL2 = 0XA5A5;
MOP_SystemCtrl->SYSCTRL0_f.HCLK_PRS = 7;
MOP_SystemCtrl->RCH_CR = *((uint16*)( 0X00100C08 )); //4M
MOP_SystemCtrl->RCH_CR = *((uint16*)( 0X00100C06 )); //8M
MOP_SystemCtrl->RCH_CR = *((uint16*)( 0X00100C04 )); //16M
MOP_SystemCtrl->RCH_CR = *((uint16*)( 0X00100C00 )); //24M
MOP_SystemCtrl->SYSCTRL2 = 0X5A5A;
MOP_SystemCtrl->SYSCTRL2 = 0XA5A5;
MOP_SystemCtrl->SYSCTRL0_f.HCLK_PRS = 0;
```

Scheme 2:

Step1: Switch the system clock to RCL, see 3.2.5 Example of switching from RCH to RCL.

Step2: Change the value of RCH_CR.TRIM to change the frequency of RCH oscillation.

Step3: Switch the system clock to RCH, see 3.2.6 Switch from RCL internal low speed to RCH example.

3.3 Clock Calibration Module

This product has a built-in clock calibration circuit. As shown in the figure below, the four sources of the system clock can be calibrated to each other. After selecting the reference clock and the clock to be calibrated, set the value of the register REFCNT and set cali.start to start the clock calibration circuit. At this time, the two 32-bit counters (increment and decrement) work at the same time. When the decrement counter is equal to 0, cali.finish is set, indicating that the calibration is over. At this time, the software can read the CALCNT value, so that it is easy to get the reference clock and The frequency relationship between the clocks being calibrated.

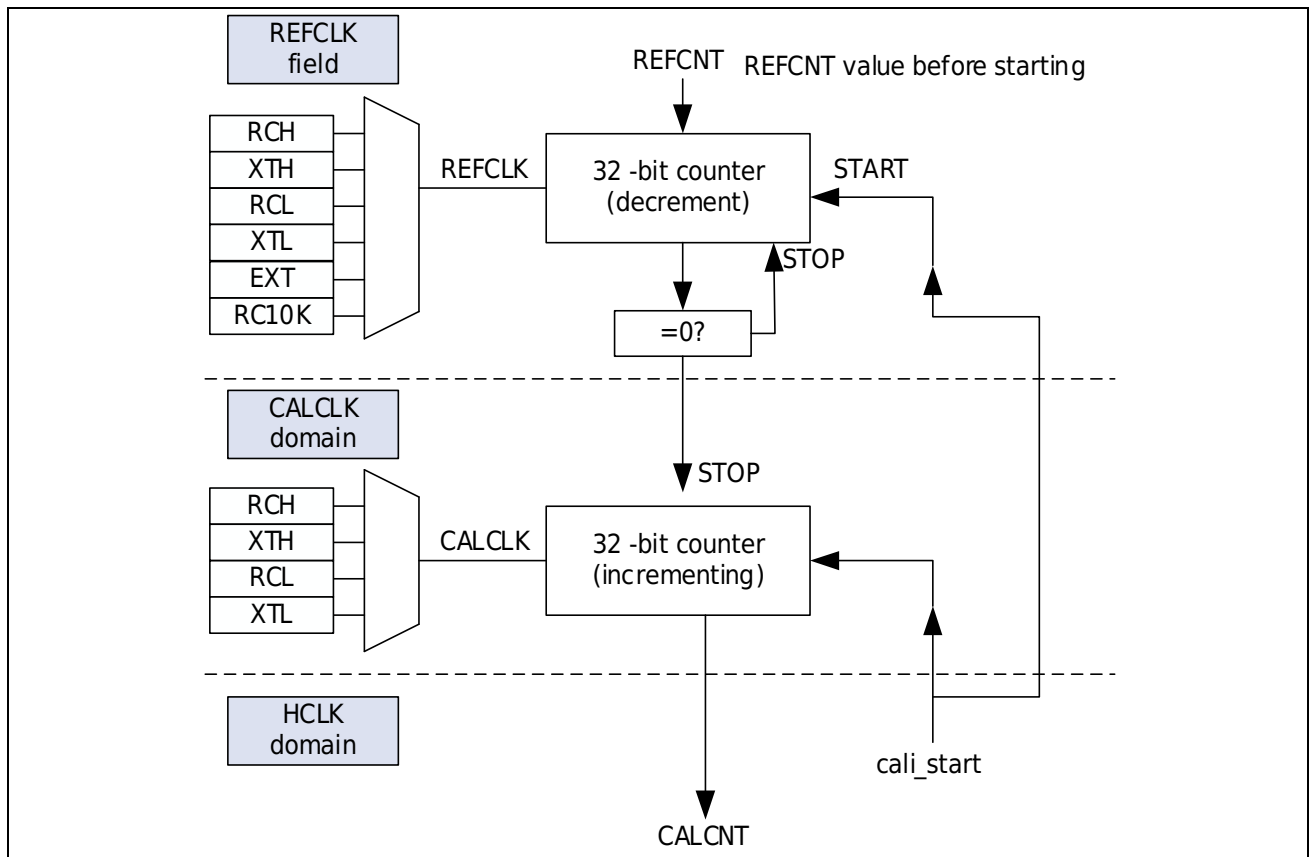


Figure 3-4 Clock Calibration Schematic

3.4 Interrupt Wakeup Control

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

The behavior of the processor in the sleep 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 $\leq$ current level	N	N
1	IRQ Priority > Current Class	Y	N
1	IRQ priority $\leq$ current level	N	N

3.4.1 Enable the NVIC corresponding to the module that needs to wake up the processor

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set SCB->SCR.SLEEPDEEP to 1
4. Execute WFI instruction to enter deep sleep mode

5. The system enters the deep sleep mode and waits for the interrupt to wake up, and executes the interrupt service program after waking up

Routine:

```
SCB_SCR |= 0x00000004u;
while(1)
{
    _asm("WFI");
}
```

3.4.2 Method not to execute interrupt service routine after wake-up from deep-sleep mode

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set PRIMASK to 1
4. Set SCB->SCR.SLEEPDEEP to 1
5. Execute WFI instruction to enter deep sleep mode
6. The system enters the deep sleep mode and waits for the interrupt to wake up, and executes the next instruction after waking up
7. Clear interrupt flag, clear interrupt pending status

Routine:

```
_asm("CPSID I"); //Set PRIMASK
SCB_SCR |= 0x00000004u;
while(1)
{
    _asm("WFI");
    BTIMERLP_REG->TFCR_f.TFC=0; //Clear Int Flag
    NVIC_ClearPendingIRQ(BASE_TIMER3_IRQn); //Clear Pending Flag
}
```

3.4.3 Using exit hibernation feature

Exiting hibernation (sleep-on-exit) is ideal for interrupt-driven applications. When this feature is enabled, the processor enters sleep mode whenever exception handling is complete and returns to thread mode. With the exit-from-sleep feature, the processor can be in sleep mode as much as possible.

Cortex-M0 uses the exit dormancy feature to enter dormancy. This situation is similar to the effect of executing WFI immediately after executing an abnormal exit. However, 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.

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set SCB->SCR.SLEEPDEEP to 1
4. Set SCB->SCR.SLEEPONEXIT to 1

5. Execute WFI instruction to enter deep sleep mode
6. The system enters the deep sleep mode and waits for the interrupt to wake up, and executes the interrupt service subroutine after waking up
7. Automatically enters sleep mode when exiting interrupt service

Routine:

```
SCB_SCR |= 0x00000004u;  
SCB_SCR |= 0x00000002u;  
while(1)  
{  
    asm("WFI");  
}
```

3.5 Register

Table 3-1 System Control Register Table

Base address 0x40002000

Register	Offset address	Description
SYSCTRL0	0x000	System Control Register 0
SYSCTRL1	0x004	System Control Register 1
SYSCTRL2	0x008	System Control Register 2
RCH_CR	0x00C	RCH Control Register
XTH_CR	0x010	XTH Control Register
RCL_CR	0x014	RCL control register
XTL_CR	0x018	XTL Control Register
PERI_CLKEN	0x020	Peripheral Module Clock Control Register
SYSTICK_CR	0x034	Systick clock control

3.5.1 System Control Register 0 (SYSCTRL0)

Offset address: 0x000

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Wakeup_b yRCH	Reserved				PCLK_PRS	HCLK_PRS			clk_sw4_sel	XTL_EN	RCL_EN	XTH_EN	RCH_EN		
R/W					R/W	R/W			R/W	R/W	R/W	R/W	R/W		

Bit	Marking	Functional description
31:16	Reserved	Keep
15	wakeup_byRCH	1: After waking up from Deep Sleep, the system clock source is RCH, and the original clock continues to be enabled. 0: After waking up from Deep Sleep, do not change the system clock source.
14:11	Reserved	Keep
10:9	PCLK_PRS	PCLK frequency division selection 00: HCLK 01: HCLK/2 10: HCLK/4 11: HCLK/8
8:6	HCLK_PRS	HCLK frequency division selection 000: SystemClk 001: SystemClk/2 010: SystemClk/4 011: SystemClk/8 100: SystemClk/16 101: SystemClk/32 110: SystemClk/64 111: SystemClk/128
5:4	Clk_sw4_sel	System clock source selection 00: Internal high-speed clock RCH 01: External high-speed crystal oscillator XTH 10: Internal low-speed clock RCL 11: External low-speed crystal oscillator XTL
3	XTL_EN	External low-speed crystal oscillator XTL enable control 0: OFF 1: Enable Note: P14 and P15 need to be set as analog ports.
2	RCL_EN	Internal low-speed clock RCL enable control 0: OFF 1: Enable
1	XTH_EN	External high-speed crystal oscillator XTH enable control 0: OFF 1: Enable Note: When the system enters DeepSleep, this high-speed clock will be automatically turned off.
0	RCH_EN	Internal high-speed clock RCH enable signal. 0: OFF 1: Enable Note: When the system enters DeepSleep, this high-speed clock will be automatically turned off.

Note:

- Every time you rewrite the value of SYCTRL0 and SYCTRL1, you need to write 0x5A5 and 0xA5A5 to SYCTRL2 in sequence. Such steps can effectively prevent misoperation of SYCTRL0 and SYCTRL1 registers.

3.5.2 System Control Register 1 (SYSCTRL1)

Offset address: 0x004

Reset value: 0x00000008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				RTC_FREQ_ADJUST		SWD_UIO	RES_UIO	LOCK_EN	RTC_LPW	Res	XTL_alwayson	EXTL_EN	EXTH_EN	Res	
				R/W		R/W	R/W	R/W	R/W		R/W	R/W	R/W		
Bit	Marking			Functional description											
31:12	Reserved														
11:9	RTC_FREQ_ADJUST			RTC high speed clock compensation clock frequency selection 000 4M; 001 6M; 010 8M;011 12M 100 16M; 101 20M; 110 24M; 111 32M;											
8	SWD_USE_IO			SWD port function configuration 0: SWD port 1: GPIO port											
7	RESET_USE_IO			RESET port function configuration 0: RESET port 1: GPIO port											
6	LOCKUP_EN			Cortex-M0+ LockUp function configuration 0: off 1: Enable Note: After enabling, the CPU will reset the MCU when it reads an invalid command, which can enhance system reliability.											
5	RTC_LPW			RTC module low power control 1: Low power mode enabled 0: Low power mode disabled Note: After enabling, the RTC module enters a low power consumption state, and its registers cannot be read or written.											
4	Reserved														
3	XTL_ALWAYS_ON			XTL Advanced Enable Control 1: SYSCTRL0.XTL_EN can only be set. 0: SYSCTRL0.XTL_EN can be set and cleared.											
2	EXTL_EN			External XTL clock input control 1: XTL output clock input from P14. 0: XTL output clock is generated by crystal oscillator. Note: When using P14 input clock, it is necessary to set SYSCTRL0.XTL_EN to 1.											
1	EXTH_EN			External XTH input control 1: XTH output clock is input from P01. 0: XTH output clock is generated by crystal oscillator. Note: When using P01 to input clock, you need to set SYSCTRL0.XTH_EN to 1.											
0	Reserved														

Note:

- Every time you rewrite the value of SYSCTRL0 and SYSCTRL1, you need to write 0x5A5 and 0xA5A5 to SYSCTRL2 in sequence. Such steps can effectively prevent misoperation of SYSCTRL0 and SYSCTRL1 registers.

3.5.3 System Control Register 2 (SYSCTRL2)

Offset address: 0x008

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYSCTRL2															
WO															

Bit	Marking	Functional description
31:16	Reserved	
15:0	SYSCTRL2	Registers SYSCTRL0 and SYSCTRL1 protect the series control registers, write 0x5A5A to SYSCTRL2 first, and then write 0xA5A5 to start the write operation to registers SYSCTRL0 and SYSCTRL1, as long as the registers SYSCTRL0 and SYSCTRL1 are written, this protection bit will automatically return to the protection state and needs to be rewritten Enter the series to open the protection.

3.5.4 RCH Control Register (RCH_CR)

Offset address: 0x00C

Reset value: 0x00000126

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				Stable	TRIM										
				RO	R/W										

Bit	Marking	Functional description
31:12	Reserved	
11	stable	Internal high-speed clock RCH stable flag. 1: It means that RCH is stable and can be used by internal circuits. 0: It means that RCH is not stable and cannot be used by internal circuits.
10:0	TRIM	Clock frequency adjustment, changing the value of this register can adjust the output frequency of RCH. Every time the register value increases by 1, the output frequency of RCH will increase by about 0.2%, and the total adjustment range is 4~24MHz. 5 groups of frequency calibration values have been saved in Flash, and the precise frequency can be obtained by reading out the calibration values in Flash and writing them into RCH_CR.TRIM. 24M calibration value address: 0x00100C00 - 0x00100C01 22.12M calibration value address: 0x00100C02 - 0x00100C03 16M calibration value address: 0x00100C04 - 0x00100C05 8M calibration value address: 0x00100C06 - 0x00100C07 4M calibration value address: 0x00100C08 - 0x00100C09 The change of RCH output frequency needs to follow a specific sequence, see section 3.2.7 of System Clock Switching for details.

3.5.5 Oscillation XTH Control Register (XTH_CR)

Offset address: 0x010

Reset value: 0x00000022

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Stable	Startup	Driver				
									RO	R/W	R/W				

Bit	Marking	Functional description
31:6	Reserved	
6	stable	External high-speed clock XTH stable flag bit. 1: It means that XTH is stable and can be used by internal circuits. 0: It means that XTH is not stable and cannot be used by internal circuits. Note: In order to increase system reliability, after querying this flag, the software needs to delay more than 10ms before switching the system clock to XTH.
5:4	Startup	External high-speed clock XTH stabilization time selection 00: 256 cycles; 01: 1024 cycles; 10: 4096 cycles; 11: 16384 cycles; Note: It is highly recommended to set the stabilization time of XTH to 11. If the XTH stabilization time is insufficient, the system will not work stably when performing clock switching or waking up from deep sleep.
3:0	Driver	3:2 Freq selects the operating frequency of the crystal oscillator 11: 24M~32M 10: 16M~24M 01: 8M~16M 00: 4M~8M 1:0 Driver selects the drive capability of the crystal oscillator 11: The strongest driving ability 10: Default drive capacity (recommended value) 01: Weak driving ability 00: Weakest drive capability Note: It is necessary to select the 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.

3.5.6 RCL Control Register (RCL_CR)

Offset address: 0x014

Reset value: 0x0000033Fh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			Stable	Startup			TRIM								
			RO	R/W			R/W R/W								

Bit	Marking	Functional description
31:13	Reserved	
12	stable	Internal low-speed clock RCL stable flag. 1: It means that RCL is stable and can be used by internal circuits. 0: It means that RCL is not stable and cannot be used by internal circuits.
11:10	Startup	Internal low-speed clock RCL stabilization time selection 11: 256 cycles; 10: 64 cycles; 01: 16 cycles; 00: 4 cycles;
9:0	TRIM	Internal low-speed clock frequency adjustment, 2 sets of frequency calibration values are stored in Flash. The precise frequency can be obtained by reading out the calibration value in Flash and writing it into RCL_CR.TRIM. 38.4K calibration value address: 0x00100C20 - 0x00100C21 32.768K calibration value address: 0x00100C22 - 0x00100C23

3.5.7 XTL Control Register (XTL_CR)

Offset address: 0x018

Reset value: 0x00000021

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Stable	Startup	Driver				
									RO	R/W	R/W				

Bit	Marking	Functional description
31:6	Reserved	
6	stable	External low-speed crystal oscillator XTL stable flag bit. 1: It means that XTL is stable and can be used by internal circuits. 0: It means that XTL is not stable and cannot be used by internal circuits.
5:4	Startup	External low-speed crystal oscillator XTL stabilization time selection 00: 256 cycles; 01: 1024 cycles; 10: 4096 cycles; 11: 16384 cycles;
3:0	Driver	External low-speed crystal oscillator XTL driver selection 1111: maximum drive 0000: minimum drive 3:2 Amp_control Fine adjustment of XTL oscillation amplitude. 11: Maximum amplitude 10: Larger amplitude (recommended value) 01: Normal amplitude 00: Minimum amplitude 1:0 Driver external crystal oscillator drive capability selection 11: The strongest driving ability 10: Strong driving ability 01: Default drive capacity (recommended value) 00: Weakest drive capability, Note: It is necessary to select the 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. 00: Weakest drive capability

3.5.8 Peripheral Module Clock Control Register (PERI_CLKEN)

Reset value: 0xC080_0000

Offset address: 0x020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Flash	Res.		GPIO	Res.	CRC	Res.	TICK	Res.		Trim	RTC	Res.		VC	ADC
R/W			R/W		R/W		R/W			R/W	R/W			R/W	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDT	PCA	Res.			ADV TIM	LP TIM	BASE TIM	Res.	SPI	Res.	I2C	Res	LPUART	UART 1	UART 0
R/W	R/W				R/W	R/W	R/W		R/W	R/W	R/W		R/W	R/W	R/W

Bit	Marking	Functional description
31	flash	The clock of the Flash controller module is enabled, and the clock needs to be enabled when operating the flash register 1: Enable; 0: Disable Note: This bit is not affected when the program is executed from flash.
30:29	Res.	Reserved bit
28	GPIO	GPIO module clock enable. 1: Enable; 0: Disable
27	Res.	Reserved bit
26	CRC	CRC module clock enable. 1: Enable; 0: Disable
25	Res.	Reserved bit
24	TICK	SysTick timer reference clock enable. 1: Enable; 0: Disable
23:22	Res.	Reserved bit
21	Trim	CLKTRIM module clock enable. 1: Enable; 0: Disable
20	RTC	RTC module clock enable. 1: Enable; 0: Disable
19:18	Res.	Reserved bit
17	VC	VC, LVD, module clock enable. 1: Enable; 0: Disable
16	ADC	ADC module clock enable. 1: Enable; 0: Disable
15	WDT	WDT module clock enable. 1: Enable; 0: Disable
14	PCA	PCA module clock enable. 1: Enable; 0: Disable
13:11	Res.	Reserved bit
10	ADVTIM	Timer456 module clock enable. 1: Enable; 0: Disable
9	LPTIM	LPTimer module clock enable. 1: Enable; 0: Disable
8	BASETIM	Timer012 module clock enable. 1: Enable; 0: Disable
7	Res.	Reserved bit
6	SPI	SPI module clock enable. 1: Enable; 0: Disable
5	Res.	Reserved bit
4	I2C	I2C module clock enable. 1: Enable; 0: Disable
3	Res.	Reserved bit
2	LPUART	LPUART module clock enable. 1: Enable; 0: Disable

1	UART1	UART1 module clock enable. 1: Enable; 0: Disable
0	UART0	UART0 module clock enable. 1: Enable; 0: Disable

3.5.9 SysTick Clock Control (SYSTICK_CR)

Reset value: 0x0100_0147

Offset address: 0x034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				CLK_SEL		NO REF	SKEW	STCALIB[23: 16]							
				R/W		R/W	R/W	R/W							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STCALIB[15: 0]															
R/W															

Bit	Marking	Functional description
31-28	Reserved	
27-26	CLK_SEL	SysTick external reference clock selection 00: External low-speed clock XTL 01: Internal low-speed clock RCL 10: System clock divided by 8 SystemClk/8 11: External high-speed clock XTH
25	NOREF	SysTick clock source selection 1: Internal high frequency clock HCLK 0: external reference clock, selected by SYSTICK_CR[27:26] Note: When using an external reference clock, the reference clock frequency is not allowed to be higher than HCLK
24	SKEW	STCALIB Accuracy Indication 1: STCALIB value represents roughly 10ms 0: STCALIB value represents exactly 10ms
23:0	STCALIB	When the reference clock is XTL, the 10 ms calibration value

4 Reset Controller (RESET)

4.1 Reset Controller Introduction

This product has 7 reset signal sources, each reset signal can make the CPU run again, most of the registers will be reset to the reset value, and the program counter PC will be reset to point to 00000000.

- POR/BOR reset (VCC domain and Vcore domain)
- External Reset PAD reset
- WDT reset
- PCA reset
- LVD 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. The user program should clear the register Reset_flag to 0 during power-on reset, and then the reset source can be judged by the relevant bits of Reset_flag at the next reset.

The figure below describes the reset sources for each area.

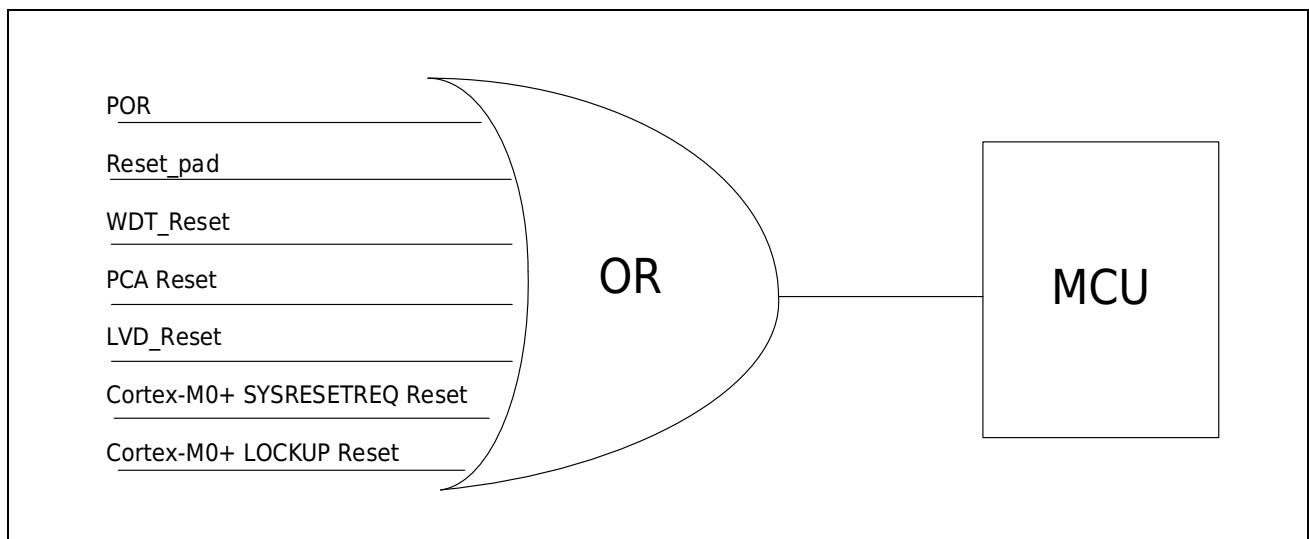


Figure 4-1 Reset Source Schematic

4.1.1 Power-on and power-off reset POR/BOR

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

When the VCC area is powered on, when the VCC voltage is lower than the POR threshold voltage (typically 1.65V), a POR5V signal will be generated; when the VCC area is powered off, when the VCC voltage is lower than the BOR threshold voltage (typically 1.5V), will generate POR5V signal.

When the Vcore area is powered on, when the Vcore voltage is lower than the POR threshold voltage, a POR15V signal will be generated; when the Vcore area is powered off, when the Vcore voltage is lower than the BOR threshold voltage, a POR15V signal will be generated.

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

4.1.2 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 will filter the glitch signal less than 20us (typical value), therefore, the low level signal added to the reset pin must be greater than 20us, in order to ensure reliable reset of the chip.

4.1.3 WDT reset

Watchdog reset, please refer to chapter WDT.

4.1.4 PCA reset

PCA reset, please refer to the chapter PCA.

4.1.5 LVD low voltage reset

LVD reset, please refer to chapter LVD.

4.1.6 Cortex-M0+ SYSRESETREQ reset

Cortex-M0+ software reset

4.1.7 Cortex-M0+ LOCKUP reset

When Cortex-M0+ encounters a serious exception, it will stop its PC pointer at the current address, lock itself, and reset the entire CORE area after a few clock cycle delays.

4.2 Register

4.2.1 Reset flag register (Reset_flag)

Reset value: 00000000_00000000_00000000_xxxxxx11b

Address: 0x4000201C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RSTB	sysreq	lockup	PCA	WDT	LVD	Por15	Por5v
								RW0	RW0	RW0	RW0	RW0	RW0	RW0	RW0

Bit	Marking	Functional description
31:8	Reserved	
7	RSTB	RESETB port reset flag, needs software initialization and clearing, power-on status is uncertain 1: A port reset occurred 0: No port reset occurs
6	Sysreq	Cortex-M0+ CPU software reset flag, needs software initialization and clearing, power-on status is uncertain 1: Cortex-M0+ CPU software reset occurred 0: No Cortex-M0+ CPU software reset occurs
5	Lockup	Cortex-M0+ CPU Lockup reset flag, needs software initialization and clearing, power-on status is uncertain 1: A Cortex-M0+ CPU Lockup reset occurred 0: No Cortex-M0+ CPU Lockup reset occurs
4	PCA	PCA reset flag, needs software initialization and clearing, power-on status is uncertain 1: PCA reset occurs 0: No PCA reset occurs
3	WDT	WDT reset flag, needs software initialization and clearing, power-on status is uncertain 1: WDT reset occurs 0: No WDT reset occurs
2	LVD	LVD reset flag, needs software initialization and clearing, power-on status is uncertain 1: LVD reset occurs 0: No LVD reset occurs
1	POR15V	Vcore domain reset flag 1: The Vcore domain is reset 0: Vcore domain does not reset
0	POR5V	VCC power domain reset flag 1: VCC power domain reset occurred 0: No reset occurs in VCC power domain

4.2.2 Peripheral module reset control register (PERI_RESET)

Reset value: 0xD7B3C757

Address: 0x40002028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.			GPIO	Res.	CRC	Res.	TICK	Res.	TRIM	RTC	Res.	VC	ADC		
			R/W		R/W		R/W		R/W	R/W		R/W			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	PCA	Res.			ADV TIM	LP TIM	BASE TIM TIM	Res.	SPI	Res.	I2C	Res.	LPUART	UART 1	UART 0
	R/W				R/W	R/W	R/W		R/W		R/W		R/W	R/W	R/W

Bit	Marking	Functional description
31:29	Res.	Reserved bit
28	GPIO	GPIO module reset enable. 1: normal operation; 0: module is in reset state
27	Res.	Reserved bit
26	CRC	CRC module reset enable. 1: normal operation; 0: module is in reset state
25	Res.	Reserved bit
24	TICK	SYSTICK module reset enable. 1: normal operation; 0: module is in reset state
23:22	Res.	Reserved bit
21	TRIM	CLKTRIM module reset enable. 1: normal operation; 0: module is in reset state
20	RTC	RTC module reset enable. 1: normal operation; 0: module is in reset state
19:18	Res.	Reserved bit
17	VC	VC, LVD, module reset enable. 1: normal operation; 0: module is in reset state
16	ADC	ADC module reset enable. 1: normal operation; 0: module is in reset state
15	Res.	Reserved bit
14	PCA	PCA module reset enable. 1: normal operation; 0: module is in reset state
13:11	Res.	Reserved bit
10	ADVTIM	Timer456 module reset enable. 1: normal operation; 0: module is in reset state
9	LPTIM	LPTimer module reset enable. 1: normal operation; 0: module is in reset state
8	BASETIM	Timer012 module reset enable. 1: normal operation; 0: module is in reset state
7	Res.	Reserved bit
6	SPI	SPI module reset enable. 1: normal operation; 0: module is in reset state
5	Res.	Reserved bit
4	I2C	I2C block reset enable. 1: normal operation; 0: module is in reset state
3	Res.	Reserved bit
2	LPUART	LPUART module reset enable. 1: normal operation; 0: module is in reset state
1	UART1	UART1 module reset enable. 1: normal operation; 0: module is in reset state

0	UART0	UART0 module reset enable. 1: normal operation; 0: module is in reset state
---	-------	--

5 Interrupt Controller (NVIC)

5.1 Overview

The Cortex-M0+ processor has a built-in nested vectored interrupt controller (NVIC), which supports up to 32 interrupt request (IRQ) inputs and 1 non-maskable interrupt (NMI) input (not used in this product system). In addition, the processor supports several internal exceptions.

Each exception source has a separate exception number, each exception type has a corresponding priority, some exceptions have a fixed priority, while others are programmable. The details are shown in the following table:

Table 5-1 Cortex-M0+ Processor Exception List

Exception number	Exception type	Priority	Description
1	Reduction	-3 (highest)	Reduction
2	NMI	-2	Non-maskable interrupt (not used in this system)
3	Hardware error	-1	Error handling exception
4-10	Keep	NA	...
11	SVC	Programmable	Invoke the hypervisor through the SVC command
12-13	Keep	NA	...
14	PendSV	Programmable	Suspendable 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 32 external interrupt requests of the processor (interrupt #0 to interrupt #31) in detail, and the specific situation of the internal exception of the processor can refer to other related documents. At the same time, this chapter only discusses the interrupt handling mechanism of the NVIC in the processor core, and the interrupt generation mechanism of the peripheral module itself is not discussed here.

5.2 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					

Figure 5-1 Only the upper two bits of the priority register are used

Preemption occurs when the processor is already running another interrupt handler and the new interrupt has a higher priority than the one currently being executed. The running interrupt processing will be suspended 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 processor is running another interrupt handler with the same or higher priority, the new interrupt will wait and enter the pending state. Pending interrupts will wait until the current interrupt level changes, for example, after the currently running interrupt handler finishes returning, the current priority is lowered to be lower than 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 executed first. For example, if interrupt #0 and interrupt #1 are enabled and have the same priority, when they are triggered at the same time, interrupt #0 will be executed first.

5.3 Interrupt digraph

When the Cortex-M0+ processor processes an interrupt service request, it needs to first determine the starting address of the exception handling. The required information is called a vector table, as shown in Figure 5-2. 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
0x0000004C	Interrupt #3 vector	19
0x00000048	Interrupt #2 vector	18
0x00000044	Interrupt #1 vector	17
0x00000040	Interrupt #0 vector	16
0x0000003C	SysTick vector	15
0x00000038	PendSV vector	14
0x00000034	unused	13
0x00000030	unused	12
0x0000002C	SVC vector	11
0x00000028	unused	10
0x00000024	unused	9
0x00000020	unused	8
0x0000001C	unused	7
0x00000018	unused	6
0x00000014	unused	5
0x00000010	unused	4
0x0000000C	hardware error exception	3
0x00000008	NMI vector	2
0x00000004	reset vector	1
0x00000000	MSP Initial value	0

Figure 5-2 Interrupt Vector Table

Among them, 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.

5.4 Interrupt Input and Suspend Behavior

NVIC module of the Cortex-M0+ processor, each interrupt input corresponds to a pending status 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 status of the interrupt will be set, and when the processor receives the interrupt and starts executing the interrupt service routine, the pending status will be cleared. The process is shown in Figure 5-3:

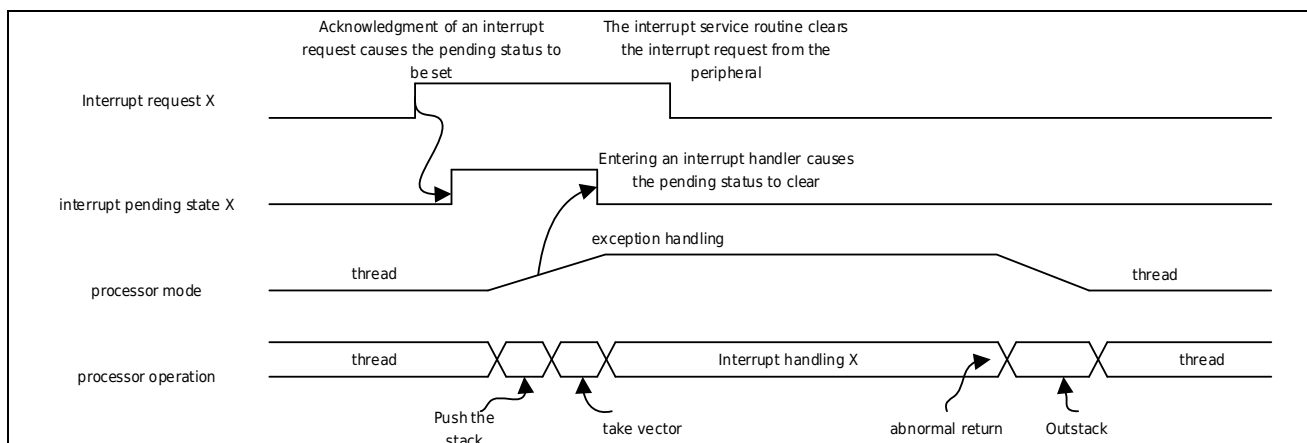


Figure 5-3 Interrupt active and pending states

If the interrupt request is not executed immediately and is cleared by software before being acknowledged, the processor ignores the request and does not perform interrupt processing. Interrupt pending status can be cleared by writing to the NVIC_CLRPEND register, which is useful when setting up a peripheral that may have generated an interrupt request before setting it.

If the peripheral is still holding an interrupt request when software clears the pending state, the pending state is also generated immediately. The process is shown in Figure 5-4:

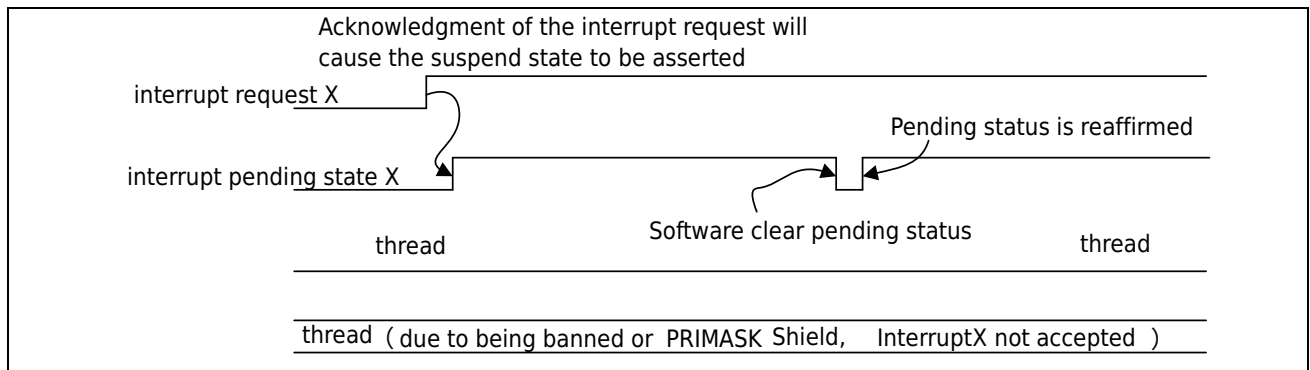


Figure 5-4 Interrupt pending status is cleared and then reasserted

If the interrupt request generated by the peripheral is not cleared when the exception is handled, the suspended state will be activated again after the exception returns, so that the interrupt service routine will be executed again. The process is shown in Figure 5-5:

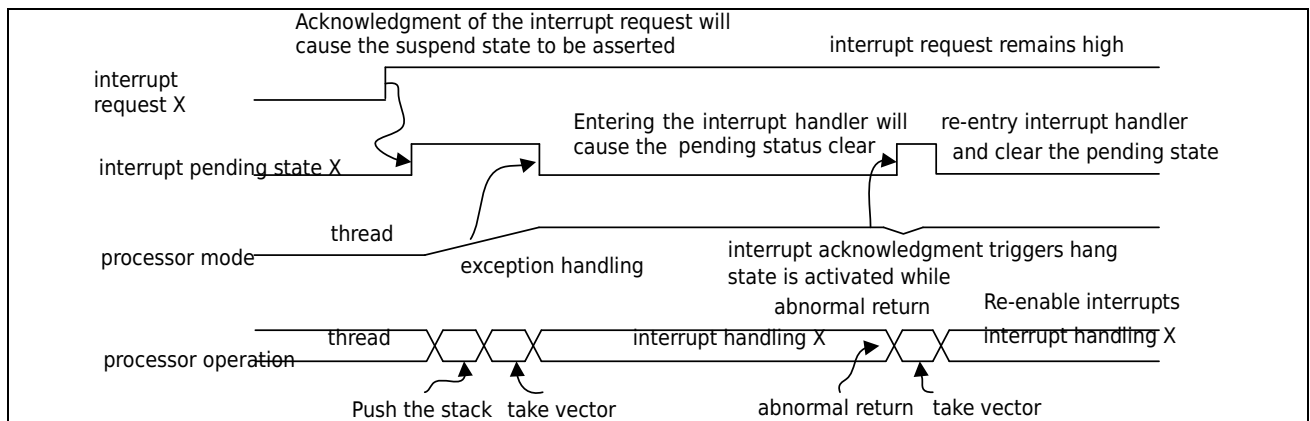


Figure 5-5 When the interrupt exits, if the interrupt request remains high, it will cause the interrupt processing to be executed again

If a peripheral interrupt request is generated during the execution of the terminal service program, the request will be regarded as a new interrupt request, and after this interrupt exits, the interrupt service program will be executed again. The process is shown in Figure 5-6:

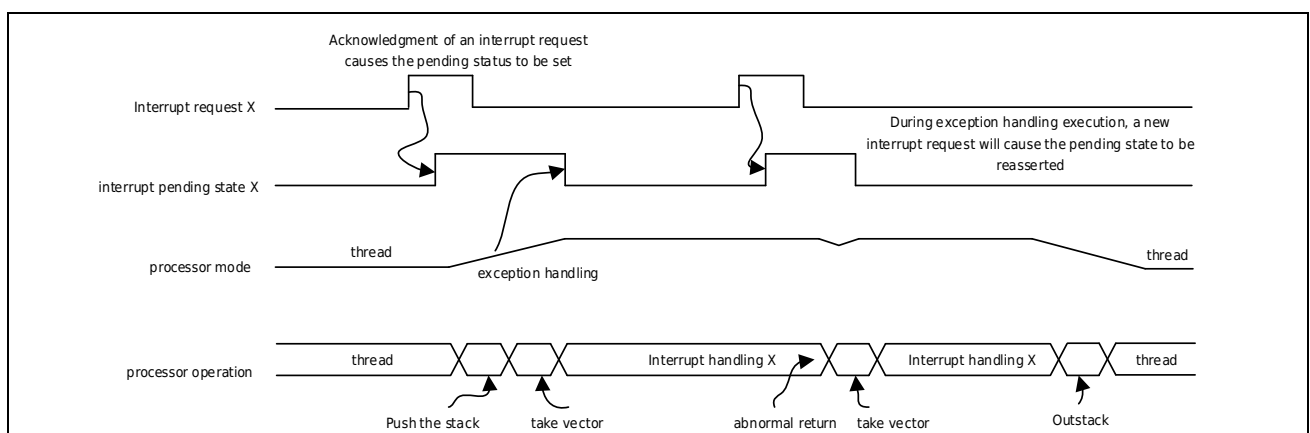


Figure 5-6 Interrupt pending interrupts generated during interrupt processing can also be acknowledged

5.5 Interrupt wait

Typically, the interrupt latency 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 SCS_PRIMASK or other exception handling in progress.
- 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 waits:

- 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.

5.6 Interrupt source

Because the NVIC of the Cortex-M0+ processor supports up to 32 external interrupts, and in this system, there are more than 32 external interrupt sources, so some external interrupts are multiplexed on the same NVIC interrupt input, and NMI (non-maskable interrupt) and did not use. all external interrupt sources of this system and NVIC interrupt input is shown in the following table:

Table 5-2 Correspondence between external interrupt and NVIC interrupt input

NVIC interrupt input	External interrupt source	Active mode	Sleep mode	DeepSleep mode
Interrupt #0	PORT0	v	v	v
Interrupt #1	PORT1	v	v	v
Interrupt #2	PORT2	v	v	v
Interrupt #3	PORT3	v	v	v
Interrupt #4	Reserved	-	-	-
Interrupt #5	Reserved	-	-	-
Interrupt #6	UART0	v	v	-
Interrupt #7	UART1	v	v	-
Interrupt #8	LPUART	v	v	v
Interrupt #9	Reserved	-	-	-
Interrupt #10	SPI	v	v	-

NVIC interrupt input	External interrupt source	Active mode	Sleep mode	DeepSleep mode
Interrupt #11	Reserved	-	-	-
Interrupt #12	I2C	v	v	-
Interrupt #13	Reserved	-	-	-
Interrupt #14	TIM0	v	v	-
Interrupt #15	TIM1	v	v	-
Interrupt #16	TIM2	v	v	-
Interrupt #17	LPTIM	v	v	v
Interrupt #18	TIM4	v	v	-
Interrupt #19	TIM5	v	v	-
Interrupt #20	TIM6		v	-
Interrupt #21	PCA	v	v	-
Interrupt #22	WDT	v	v	v
Interrupt #23	RTC	v	v	v
Interrupt #24	ADC	v	v	-
Interrupt #25	Reserved	-	-	-
Interrupt #26	VC0	v	v	v
Interrupt #27	VC1	v	v	v
Interrupt #28	LVD	v	v	v
Interrupt #29	Reserved	-	-	-
Interrupt #30	EFCTRL/RAMCTRL	v	v	-
Interrupt #31	CLK_TRIM	v	v	v

Note:

- Because some module interrupts are reused 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.

5.7 Interrupt structure diagram

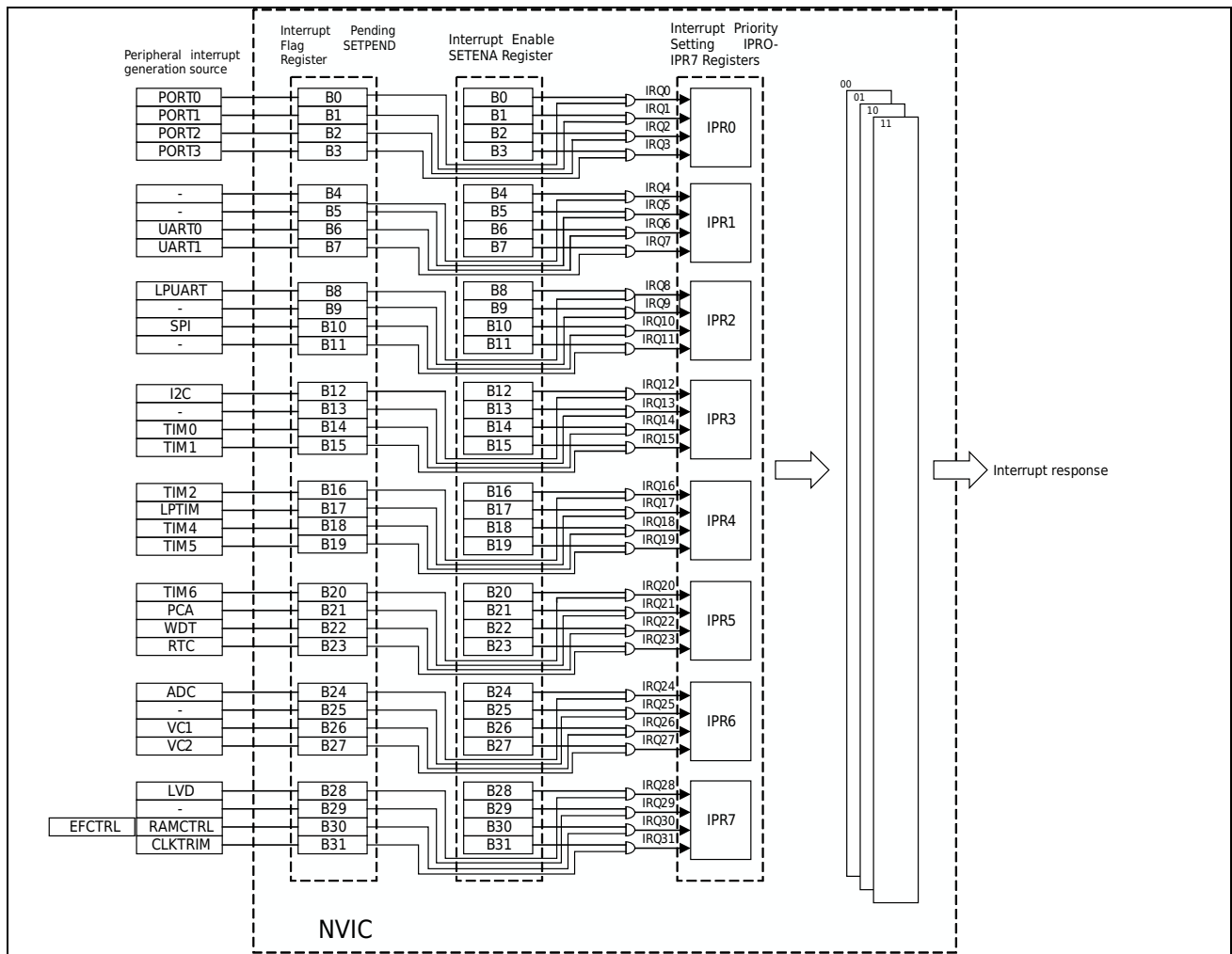


Figure 5-7 Interrupt Structure Diagram

The interrupt structure block diagram of this system is shown in Figure 5-7. A few points to note:

- 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.
- IRQ30 has 2 peripheral interrupt inputs multiplexed, and the interrupt flag bits of these 2 peripherals must be read separately to determine which peripheral interrupt it is.
- If the peripheral interrupt source has a high level, regardless of whether the NVIC interrupt enable register SCS_SETENA is set or not, the interrupt pending register SCS_SEPEND will be set, indicating that the corresponding peripheral interrupt source has an interrupt.
- Only when the interrupt enable register SCS_SETENA 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 SCS_SEPEND is automatically cleared by hardware.

- The interrupt priority registers SCS_IPR0-SCS_IPR7 set the priority of 32 interrupt sources, 00 has the highest priority and 11 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.

5.8 Register

Base address: 0xE000 E000

Register	Offset address	Description
SCS_SETENA	0x100	Interrupt Request Enable Register
SCS_CLRENA	0x180	Interrupt Request Clear Enable Register
SCS_SETPEND	0x200	Interrupt Set Pending Register
SCS_CLRPEND	0x280	Interrupt Clear Pending Register
SCS_IPR0	0x400	Interrupt #0- Interrupt #3 Priority Registers
SCS_IPR1	0x404	Interrupt #4- Interrupt #7 Priority Registers
SCS_IPR2	0x408	Interrupt #8- Interrupt #11 Priority Registers
SCS_IPR3	0x40C	Interrupt #12- Interrupt #15 Priority Registers
SCS_IPR4	0x410	Interrupt #16- Interrupt #19 Priority Registers
SCS_IPR5	0x414	Interrupt #20- Interrupt #23 Priority Registers
SCS_IPR6	0x418	Interrupt #24- Interrupt #27 Priority Registers
SCS_IPR7	0x41C	Interrupt #28- Interrupt #31 Priority Registers
SCS_PRIMASK	-	Interrupt Mask Special Register

5.8.1 Interrupt Enable Setting Register (SCS_SETENA)

Offset address: 0x100

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SETENA[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETENA[15:0]															
RW															

Bit	Marking	Functional description
31:0	SETENA [31:0]	Set enable interrupt #0 to interrupt #31; write "1" set, write "0" invalid [0]:IRQ0 [1]:IRQ1 [2]:IRQ2 [31]:IRQ31

5.8.2 Interrupt Enable Clear Register (SCS_CLRENA)

Offset address: 0x180

Reset value: 0x0000 0000

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

Bit	Marking	Description
31:0	CLRENA	Clear enable interrupt #0 to interrupt #31; write "1" to clear, write "0" to be invalid

5.8.3 Interrupt Pending Status Set Register (SCS_SETPEND)

Offset address: 0x200

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SETPEND[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETPEND[15:0]															
RW															

Bit	Marking	Functional description
31:0	SETPEND	Set the pending status of interrupt #0 to interrupt #31; write "1" to set, write "0" to have no effect [0]:IRQ0 [1]:IRQ1 [2]:IRQ2 [31]:IRQ31

5.8.4 Interrupt Pending Status Clear Register (SCS_CLRPEND)

Offset address: 0x280

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLRPEND[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRPEND[15:0]															
RW															

Bit	Marking	Description
31:0	CLRPEND	Clear pending status of Interrupt #0 to Interrupt #31; write "1" to clear, write "0" to have no effect [0]:IRQ0 [1]:IRQ1 [2]:IRQ2 [31]:IRQ31

5.8.5 Interrupt Priority Register (SCS_IPR0)

Offset address: 0x400

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR0[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR0[15:0]															
RW															

Bit	Marking	Functional description
31:0	IPR0[31:0]	Priority of interrupt #0 to interrupt #3; [31:30]: Priority of interrupt #3 [23:22]: Priority of interrupt #2 [15:14]: Priority of interrupt #1 [7:6]: Priority of Interrupt #0 Among them, 00 has the highest priority and 11 has the lowest priority

5.8.6 Interrupt Priority Register (SCS_IPR1)

Offset address: 0x404

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR1[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR1[15:0]															
RW															

Bit	Marking	Functional description
31:0	IPR1[31:0]	Priority of interrupt #4 to interrupt #7; [31:30]: Priority of interrupt #7 [23:22]: Priority of interrupt #6 [15:14]: Priority of interrupt #5 [7:6]: Priority of Interrupt #4 Among them, 00 has the highest priority and 11 has the lowest priority

5.8.7 Interrupt Priority Register (SCS_IPR2)

Offset address: 0x408

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR2[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR2[15:0]															
RW															

Bit	Marking	Functional description
31:0	IPR2[31:0]	Priority of interrupt #8 to interrupt #11; [31:30]: Priority of interrupt #11 [23:22]: Priority of interrupt #10 [15:14]: Priority of interrupt #9 [7:6]: Priority of Interrupt #8 Among them, 00 has the highest priority and 11 has the lowest priority

5.8.8 Interrupt Priority Register (SCS_IPR3)

Offset address: 0x40C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR3[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR3[15:0]]															
RW															

Bit	Marking	Functional description
31:0	IPR3[31:0]	Priority of interrupt #12 to interrupt #15; [31:30]: Priority of interrupt #15 [23:22]: Priority of interrupt #14 [15:14]: Priority of interrupt #13 [7:6]: Priority of Interrupt #12 Among them, 00 has the highest priority and 11 has the lowest priority

5.8.9 Interrupt Priority Register (SCS_IPR4)

Offset address: 0x410

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR4[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR4[15:0]]															
RW															

Bit	Marking	Functional description
31:0	IPR4[31:0]	Priority of interrupt #16 to interrupt #19; [31:30]: Priority of interrupt #19 [23:22]: Priority of interrupt #18 [15:14]: Priority of interrupt #17 [7:6]: Priority of Interrupt #16 Among them, 00 has the highest priority and 11 has the lowest priority

5.8.10 Interrupt Priority Register (SCS_IPR5)

Offset address: 0x414

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR5[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR5[15:0]															
RW															

Bit	Marking	Functional description
31:0	IPR5[31:0]	Priority of interrupt #20 to interrupt #23; [31:30]: Priority of interrupt #23 [23:22]: Priority of interrupt #22 [15:14]: Priority of interrupt #21 [7:6]: Priority of Interrupt #20 Among them, 00 has the highest priority and 11 has the lowest priority

5.8.11 Interrupt Priority Register (SCS_IPR6)

Offset address: 0x418

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR6[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR6[15:0]															
RW															

Bit	Marking	Functional description
31:0	IPR6[31:0]	Priority of interrupt #24 to interrupt #27; [31:30]: Priority of interrupt #27 [23:22]: Priority of interrupt #26 [15:14]: Priority of interrupt #25 [7:6]: Priority of Interrupt #24 Among them, 00 has the highest priority and 11 has the lowest priority

5.8.12 Interrupt Priority Register (SCS_IPR7)

Offset address: 0x41C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR7[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR7[15:0]															
RW															

Bit	Marking	Functional description
31:0	IPR7[31:0]	Priority of interrupt #28 to interrupt #31; [31:30]: Priority of interrupt #31 [23:22]: Priority of interrupt #30 [15:14]: Priority of interrupt #29 [7:6]: Priority of Interrupt #28 Among them, 00 has the highest priority and 11 has the lowest priority

5.8.13 Interrupt Mask Special Register (SCS_PRIMASK)

Offset address: ---

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PRIMASK
R															RW

SCS_PRIMASK															
Reset value: 0x0000_0000															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															PRIMASK
RO															RW
BIT	Symbol	Description													
31:1	Reserved														
0	PRIMASK	When set, all interrupts except NMI and hardware error exceptions will be masked After clearing, all exceptions and interrupts will not be masked The special register needs to be accessed through MSR and MRS special register operation instructions, and can also be accessed by changing the processor state instruction CPS. When dealing with time-sensitive applications, it is necessary to manipulate the PRIMASK register.													

5.9 Basic operation of the software

5.9.1 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 respective chapter descriptions of the peripheral modules.

5.9.2 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. In this way, there are 32-bit interrupt enable register `SCS_SETENA` and 32-bit clear enable register `SCS_CLRENA`. If you want to enable an interrupt, set the corresponding bit of the `SCS_SETENA` register to 1. If you want to clear an interrupt, set the corresponding bit in the `SCS_CLRENA` register to 1.

Note that the interrupt enable mentioned here is only for the processor NVIC. Whether the interrupt of each peripheral is generated or not is determined by the interrupt control register of the peripheral, and has nothing to do with `SCS_SETENA` and `SCS_CLRENA`.

5.9.3 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 suspended state.

The interrupt pending status can be accessed or modified by operating the interrupt pending set `SCS_SETPEND` and interrupt pending clear `SCS_CLRPEND` registers. The Interrupt Pending Status Register allows software to trigger interrupts.

5.9.4 NVIC interrupt priority

Setting the `SCS_IPR0-SCS_IPR7` registers determines the priority of `SCS_IRQ0-SCS_IRQ32`. 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 result of this situation is unpredictable and is not supported by the Cortex-M0+ processor.

5.9.5 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 SCS_PRIMASK. Only 1 bit of the special register SCS_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 SCS_PRIMASK is set to 1, the current priority of the processor is reduced to 0 (the highest priority of the summable value).

SCS_PRIMASK register can be programmed in a variety of ways. Using assembly language, the MSR instruction can be used to set and clear the SCS_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
```

6 Port Controller (GPIO)

6.1 Introduction to Port Controllers

This product has 16 digital general input and output ports P01-P03, P14-P15, P23-P27, P31-P36 and 1 digital general input port P00. The input and output signals of analog signal ADC/VC/LVD, the input and output signals of various functional modules (such as SPI, UART, I2C, Timer, etc.), and the input and output signals of test and debugging functions can be multiplexed with digital ports.

Each port can be configured as internal pull-up (pull up)/ pull-down (pull down) input, high-impedance input (floating input), push-pull output (CMOS output), open drain output (open drain output), two levels drive capability output. In order to prevent abnormal actions of external devices when the chip is abnormally reset, the port is configured as a high-impedance input after the chip is reset. However, in order to avoid the leakage caused by high-impedance input, the user should configure the port accordingly after the chip is started (configured as internal pull-up/ pull -down input or output).

"1" and "0" cannot be output, and the result of the CPU reading the port is "0".

All ports can provide external interrupts, and each interrupt can be configured into four types: high-level trigger, low-level trigger, rising edge trigger, and falling edge trigger. You can check the interrupt flag bit of Px_STAT[n] The corresponding interrupt trigger port. In addition, each port interrupt can wake up the chip from sleep mode / deep sleep mode to work mode.

6.2 Port Controller Main Features

The port controller supports the following features:

- Port multiplexing function
 - Analog function pin multiplexing
 - Debug pin multiplexing
 - Digital common pin multiplexing
 - Digital function pin multiplexing
- Configuration Features
 - Support pull-up / pull-down
 - High level/low level
 - Push-pull output
 - Open-Drain Output
- External interrupt source
 - High level / low level
 - Rising edge / falling edge
- Support interrupt in working mode/sleep mode/deep sleep mode

6.3 Port Controller Functional Description

6.3.1 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 configured as a digital port, you can also configure the corresponding registers to achieve the following features:

1. Internal pull-up (PxPU) / pull-down (PxPD)

The pull-up register (PxPU) and the pull-down register (PxPD) correspond to the port pull-up enable and port pull-down enable respectively. When the corresponding bit is '1', set the corresponding bit pin pull-up / pull-down enable to '0', disable the pull-up / pull-down of the corresponding bit pin.

2. Two-speed drive output (PxDR)

The driving ability can be changed through the PxDR register. When PxDR is '1', it is low driving ability, and when PxDR is '0', it is high driving ability.

3. 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 open-drain output is disabled. 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.

4. Direction selection (PxDIR)

Used to set the direction of the port pin. PxDIR is '0', the port is output, and when PxDIR is '1', the port is input.

5. Output high and low level selection (PxOUT), accessible through AHB bus

When the port pin is configured as an output, if PxOUT is '1', the output of the port pin is high level, if it is configured as an open-drain output, an external pull-up resistor is required to pull it high. If PxOUT is '0', output low level.

6. Input level status (PxIN) can be accessed through the AHB bus

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

Note: The above features are invalid when configured as an analog port.

Table 6-1 Port State Truth Table

80/415

The port circuit structure is shown in the figure below:

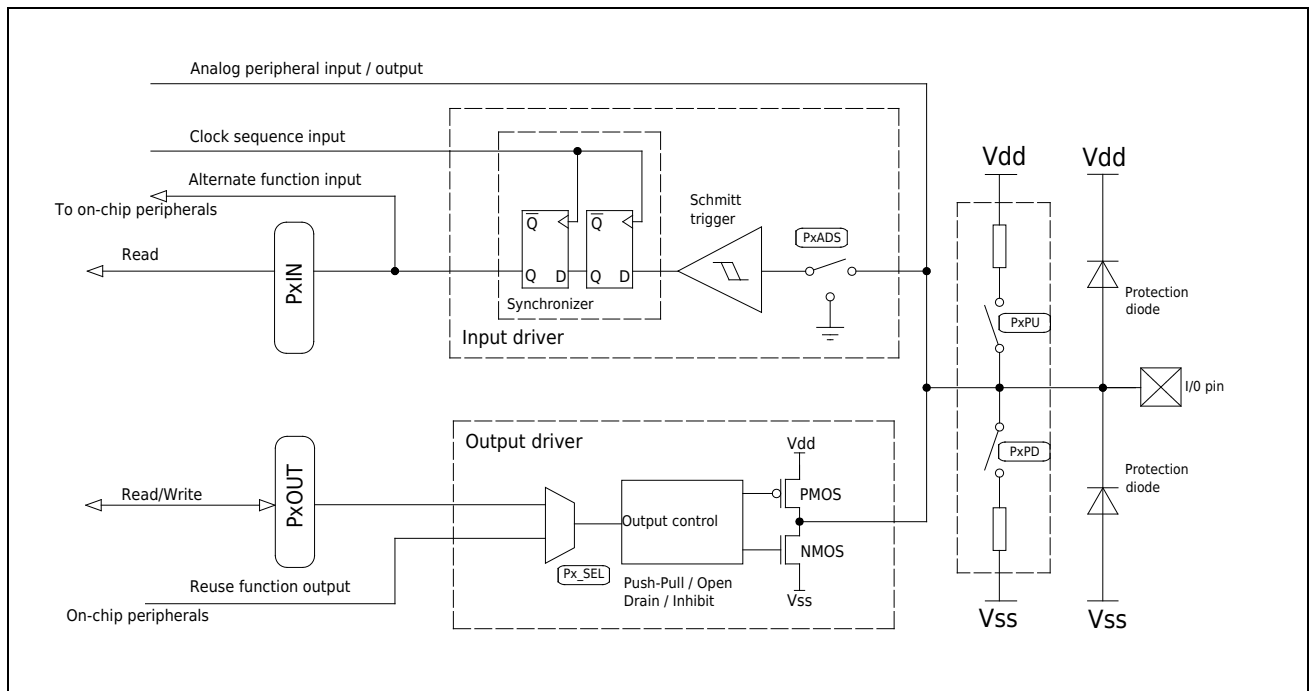


Figure 6-1 Port Circuit Diagram

6.3.2 Port write

The port input value / output value register (PxIN/PxOUT) supports AHB bus read and write. For the AHB bus, the processing cycle of the system clock (HCLK) is different from that of other buses, and the IO is flipped every two HCLK cycles. The following figure shows the fastest timing of AHB bus port flipping:

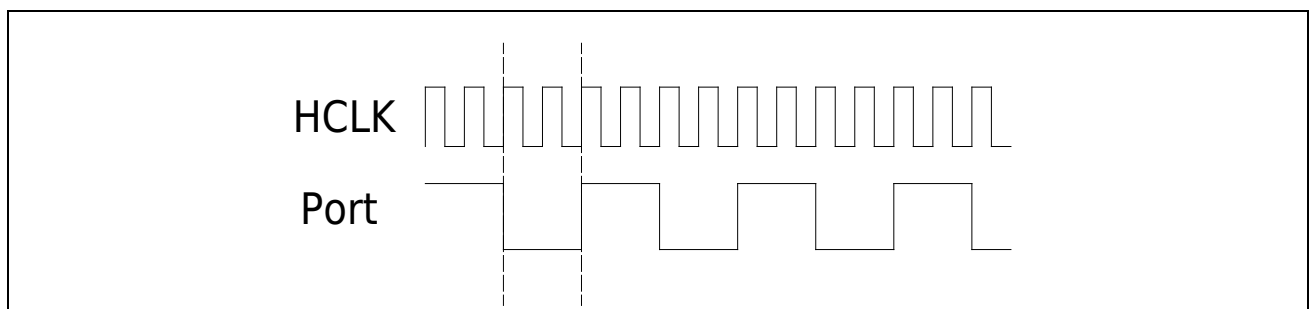


Figure 6-2 AHB BusPort Changes With System Clock

6.3.3 Port read

Each port can get the port pin level by reading the PxIN register. PxIN register and the latch in front of it form a synchronizer, thereby avoiding the signal instability caused by the pin level change in a short time when the system clock state changes, but it also introduces a delay. The synchronization diagram for reading port pin data is as follows:

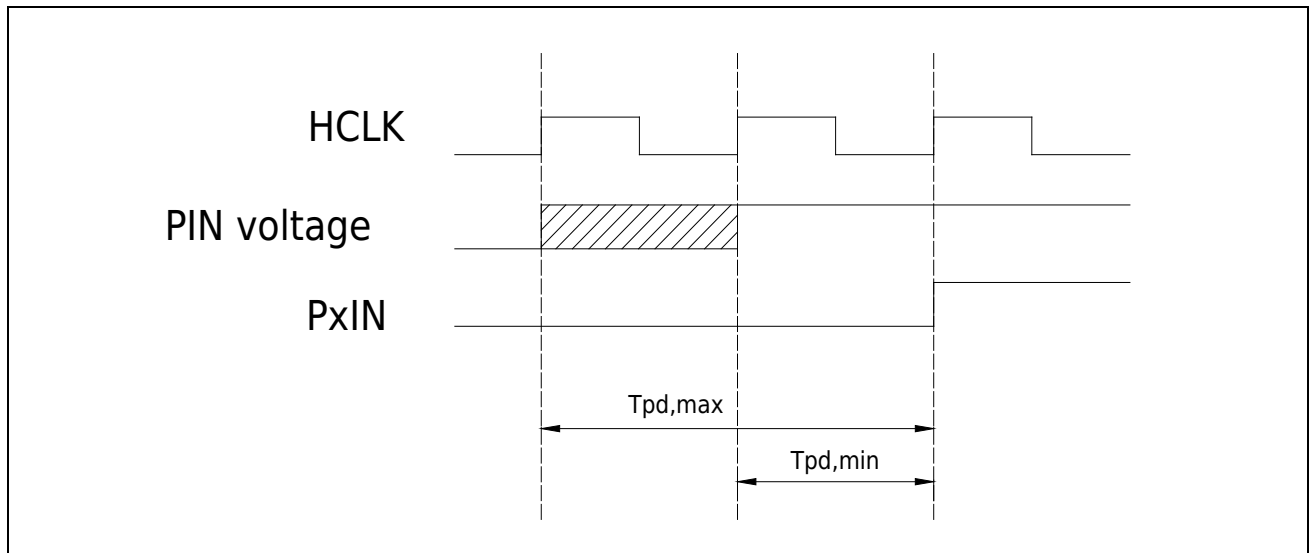


Figure 6-3 Read port pin data synchronization diagram

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 system clock rises, the data is latched into the PxIN register. The signal replacement delay Tpd is 1-2 system clocks.

Note:

- Handling of unconnected pins:
If there are unconnected pins during use, in order to avoid 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.

6.3.4 Port multiplexing function

Port multiplexing is one of the main functions of a port controller. Through the configuration register, the port can be flexibly configured as an analog port / test and debug port / digital general port / digital function port.

The PxADS register is used for digital port / analog port switching. When PxADS is '1', the port is configured as an analog port. At this time, the digital function is isolated and the digital "1" and "0" cannot be output. The result of the CPU reading the port is "0". When PxADS is '0', the port is configured as a digital port. At this time, the digital general-purpose port / digital function port is switched by configuring the Px_sel register. Each port can be independently configured as a functional port required by the system. For configuration information about testing and debugging ports, please refer to relevant chapters.

Table 6-2 Port multiplexing table

Simulation		Number							
PxADS=1		PxADS=0							
		Px_sel=0	Px_sel=1	Px_sel=2	Px_sel=3	Px_sel=4	Px_sel=5	Px_sel=6	Px_sel=7
AIN4	VCIN4	P34	PCA_CH0	LPUART_TXD	TIM5_CHA	TIM0_EXT	TIM4_CHA	RTC_1Hz	TIM1_TOG
AIN5	VCIN5	P35	UART1_TXD	TIM6_CHB	UART0_TXD	TIM0_GATE	TIM4_CHB	SPI_MISO	I2C_SDA
AIN6 ADC_VREF	VCIN6	P36	UART1_RXD	TIM6_CHA	UART0_RXD	PCA_CH4	TIM5_CHA	SPI_MOSI	I2C_SCL
AIN7 XTHI	VCIN7	P01	UART0_RXD	I2C_SDA	UART1_TXD	TIM0_TOG	TIM5_CHB	SPI_SCK	TIM2_EXT
AIN8 XTHO		P02	UART0_TXD	I2C_SCL	UART1_RXD	TIM0_TOGN	TIM6_CHA	SPI_CS	TIM2_GATE
LVDIN1		P03	PCA_CH3	SPI_CS	TIM6_CHB	LPTIM_EXT	RTC_1Hz	PCA_ECI	VC0_OUT
XTLO		P15	I2C_SDA	TIM2_TOG	TIM4_CHB	LPTIM_GATE	SPI_SCK	UART0_RXD	LVD_OUT
XTLI		P14	I2C_SCL	TIM2_TOGN	PCA_ECI	ADC_RDY	SPI_CS	UART0_TXD	NC
LVDIN2	VCIN0	P23	TIM6_CHA	TIM4_CHB	TIM4_CHA	PCA_CH0	SPI_MISO	UART1_TXD	IR_OUT
AIN0		P24	TIM4_CHB	TIM5_CHB	HCLK_OUT	PCA_CH1	SPI_MOSI	UART1_RXD	VC1_OUT
LVDIN3	VCIN1	P25	SPI_SCK	PCA_CH0	TIM5_CHA	LVD_OUT	LPUART_RXD	I2C_SDA	TIM1_GATE
AIN1		P26	SPI_MOSI	TIM4_CHA	TIM5_CHB	PCA_CH2	LPUART_TXD	I2C_SCL	TIM1_EXT
		P27/SWDIO	SPI_MISO	TIM5_CHA	TIM6_CHA	PCA_CH3	UART0_RXD	RCH_OUT	XTH_OUT
		P31/SWCLK	LPTIM_TOG	PCA_ECI	PCLK_OUT	VC0_OUT	UART0_TXD	RCL_OUT	HCLK_OUT
AIN2	VCIN2	P32	LPTIM_TOGN	PCA_CH2	TIM6_CHB	VC1_OUT	UART1_TXD	PCA_CH4	RTC_1Hz
AIN3	VCIN3	P33	LPUART_RXD	PCA_CH1	TIM5_CHB	PCA_ECI	UART1_RXD	XTL_OUT	TIM1_TOGN
		P00 Reset							

6.3.5 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 / low level

interrupt enable register / rising edge interrupt enable register / falling edge interrupt enable register.

When an interrupt is triggered, it can be determined which port triggered the interrupt by querying the interrupt status register, and the corresponding interrupt status flag bit can be cleared by clearing the interrupt clear register.

6.4 Port Configuration Operations

6.4.1 Port multiplexing operation process

Port multiplexing configured as an analog port

Step1: Set register Px_ADS to 1

Port multiplexing configured as a digital universal port

- a) Set register Px_ADS to 0
- b) Set register Px_sel to 0
- c) Set the register PxDIR to 1: the port direction is input, and the CPU can read the port status PxIN
- d) Set register PxDIR to 0: port direction is output
- e) Set the register PxOUT to 1: port output high level
- f) Set the register PxOUT to 0: port output low level

Port multiplexing configured as a digital function port

- a) Set register Px_ADS to 0
- b) Set the register Px_sel to 1~7 (according to system requirements, refer to the port multiplexing table)
- c) Set the register PxDIR (according to system requirements)

Port multiplexing is configured as a debug test port

Refer to the chapters related to testing and debugging.

Port multiplexing configured as infrared output signal

Port P23 can be configured as an infrared output signal with a frequency of 38K.

- a) Set register P23_ADS to 0
- b) Set register P23_sel to 7
- c) Set register P2DIR[3] to 0: port direction is output
- d) Set the bit14 of the register GPIO_CTRL1 to select the output polarity of the infrared signal
- e) Set register P2OUT[3] to gate the output of infrared signal

6.4.2 Port Interrupt Operation Flow

High level interrupt

- a) Set register Px_ADS to 0

- b) Set register Px_sel to 0
- c) Set register PxDIR to 1
- d) Set register PxHIE to 1
- e) Read the interrupt status register Px_STAT after the interrupt is triggered
- f) Set the register Px_ICLR to 0 to clear the interrupt status register Px_STAT

Low level interrupt

- a) Set register Px_ADS to 0
- b) Set register Px_sel to 0
- c) Set register PxDIR to 1
- d) Set register PxLIE to 1
- e) Read the interrupt status register Px_STAT after the interrupt is triggered
- f) Set the register Px_ICLR to 0 to clear the interrupt status register Px_STAT

Rising edge interrupt

- a) Set register Px_ADS to 0
- b) Set register Px_sel to 0
- c) Set register PxDIR to 1
- d) Set register PxRIE to 1
- e) Read the interrupt status register Px_STAT after the interrupt is triggered
- f) Set the register Px_ICLR to 0 to clear the interrupt status register Px_STAT

Falling edge interrupt

- a) Set register Px_ADS to 0
- b) Set register Px_sel to 0
- c) Set register PxDIR to 1
- d) Set register PxFIE to 1
- e) Read the interrupt status register Px_STAT after the interrupt is triggered
- f) Set the register Px_ICLR to 0 to clear the interrupt status register Px_STAT

6.4.3 Port configuration operation flow

Pull-up enable

- a) Set register PxPU to 1

Pull down enable

- a) Set register PxPU to 0
- b) Set register PxPD to 1

High drive capacity

- a) Set register PxDR to 0

Open-Drain Output

- a) Set register PxOD to 1

6.5 Port Controller Register Description

Register list

Base address: 0x40020C00

Offset	Register name	Access	Register description
0x04	P01_SEL	RW	Port P01 function configuration register
0x08	P02_SEL	RW	Port P02 Function Configuration Register
0x0c	P03_SEL	RW	Port P03 Function Configuration Register
0x50	P14_SEL	RW	Port P14 Function Configuration Register
0x54	P15_SEL	RW	Port P15 Function Configuration Register
0x8c	P23_SEL	RW	Port P23 Function Configuration Register
0x90	P24_SEL	RW	Port P24 Function Configuration Register
0x94	P25_SEL	RW	Port P25 Function Configuration Register
0x98	P26_SEL	RW	Port P26 Function Configuration Register
0x9c	P27_SEL	RW	Port P27 Function Configuration Register
0xc4	P31_SEL	RW	Port P31 Function Configuration Register
0xc8	P32_SEL	RW	Port P32 Function Configuration Register
0xcc	P33_SEL	RW	Port P33 Function Configuration Register
0xd0	P34_SEL	RW	Port P34 Function Configuration Register
0xd4	P35_SEL	RW	Port P35 Function Configuration Register
0xd8	P36_SEL	RW	Port P36 Function Configuration Register
0x100	P0DIR	RW	Port P0 input and output configuration register
0x104	P0IN	RO	Port P0 Input Value Register
0x108	P0OUT	RW	Port P0 output value configuration register
0x10c	P0ADS	RW	Port P0 digital-analog configuration register
0x11c	P0DR	RW	Port P0 Drive Capability Configuration Register
0x120	P0PU	RW	Port P0 pull-up enable configuration register
0x124	P0PD	RW	Port P0 pull-down enable configuration register
0x12c	P0OD	RW	Port P0 open-drain output configuration register
0x130	P0HIE	RW	Port P0 High Level Interrupt Enable Configuration Register
0x134	P0LIE	RW	Port P0 Low Level Interrupt Enable Configuration Register
0x138	P0RIE	RW	Port P0 rising edge interrupt enable configuration register
0x13c	P0FIE	RW	Port P0 falling edge interrupt enable configuration register
0x200	P0_STAT	RO	Port P0 Interrupt Status Register
0x210	P0_ICLR	RW	Port P0 Interrupt Clear Register
0x140	P1DIR	RW	Port P1 input and output configuration register
0x144	P1IN	RO	Port P1 Input Value Register
0x148	P1OUT	RW	Port P1 output value configuration register
0x14c	P1ADS	RW	Port P1 digital-analog configuration register

Offset	Register name	Access	Register description
0x15c	P1DR	RW	Port P1 Drive Capability Configuration Register
0x160	P1PU	RW	Port P1 pull-up enable configuration register
0x164	P1PD	RW	Port P1 pull-down enable configuration register
0x16c	P1OD	RW	Port P1 Open-Drain Output Configuration Register
0x170	P1HIE	RW	Port P1 High Level Interrupt Enable Configuration Register
0x174	P1LIE	RW	Port P1 Low Level Interrupt Enable Configuration Register
0x178	P1RIE	RW	Port P1 Rising Edge Interrupt Enable Configuration Register
0x17c	P1FIE	RW	Port P1 falling edge interrupt enable configuration register
0x240	P1_STAT	RO	Port P1 Interrupt Status Register
0x250	P1_ICLR	RW	Port P1 Interrupt Clear Register
0x180	P2DIR	RW	Port P2 input and output configuration register
0x184	P2IN	RO	Port P2 Input Value Register
0x188	P2OUT	RW	Port P2 output value configuration register
0x18c	P2ADS	RW	Port P2 digital-analog configuration register
0x19c	P2DR	RW	Port P2 Drive Capability Configuration Register
0x1a0	P2PU	RW	Port P2 pull-up enable configuration register
0x1a4	P2PD	RW	Port P2 pull-down enable configuration register
0x1ac	P2OD	RW	Port P2 Open-Drain Output Configuration Register
0x1b0	P2HIE	RW	Port P2 High Level Interrupt Enable Configuration Register
0x1b4	P2LIE	RW	Port P2 Low Level Interrupt Enable Configuration Register
0x1b8	P2RIE	RW	Port P2 Rising Edge Interrupt Enable Configuration Register
0x1bc	P2FIE	RW	Port P2 falling edge interrupt enable configuration register
0x280	P2_STAT	RO	Port P2 Interrupt Status Register
0x290	P2_ICLR	RW	Port P2 Interrupt Clear Register
0x1c0	P3DIR	RW	Port P3 input and output configuration register
0x1c4	P3IN	RO	Port P3 Input Value Register
0x1c8	P3OUT	RW	Port P3 output value configuration register
0x1cc	P3ADS	RW	Port P3 digital-analog configuration register
0x1dc	P3DR	RW	Port P3 Drive Capability Configuration Register
0x1e0	P3PU	RW	Port P3 pull-up enable configuration register
0x1e4	P3PD	RW	Port P3 pull-down enable configuration register
0x1ec	P3OD	RW	Port P3 Open-Drain Output Configuration Register
0x1f0	P3HIE	RW	Port P3 High Level Interrupt Enable Configuration Register
0x1f4	P3LIE	RW	Port P3 Low Level Interrupt Enable Configuration Register
0x1f8	P3RIE	RW	Port P3 Rising Edge Interrupt Enable Configuration Register
0x1fc	P3FIE	RW	Port P3 falling edge interrupt enable configuration register
0x2c0	P3_STAT	RO	Port P3 Interrupt Status Register
0x2d0	P3_ICLR	RW	Port P3 Interrupt Clear Register

Offset	Register name	Access	Register description
0x304	GPIO_CTRL1	RW	Port Miscellaneous Function Configuration Register 1
0x308	GPIO_CTRL2	RW	Port Miscellaneous Function configuration register 2
0x30c	GPIO_CTRL3	RW	Port auxiliary function configuration register 3
0x310	GPIO_CTRL4	RW	Port Miscellaneous Function Configuration Register 4

6.5.1 Port P0

6.5.1.1 Port P01 function configuration register (P01_SEL)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P01_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P01_sel	Port P01 function selection. 000: GPIO P01 001: UART0_RXD UART0 module RXD signal 010: I2C_SDA I2C module data signal 011: UART1_TXD UART1 module TXD signal 100: TIM0_TOG Timer0 module flip signal 101: TIM5_CHB Advanced Timer module channel 1 B signal 110: SPI_SCK SPI module clock signal 111: TIM2_EXT Timer2 module external clock input signal

6.5.1.2 Port P02 function configuration register (P02_SEL)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P02_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P02_sel	Port P02 function selection. 000: GPIO P02 001: UART0_TXD UART0 module TXD signal 010: I2C_SCL I2C module clock signal 011: UART1_RXD UART1 module RXD signal 100: TIM0_TOGN Timer0 module flip signal reverse signal 101: TIM6_CHA Advanced Timer module channel 2 A signal 110: SPI_CS SP I module master mode chip select signal 111: TIM2_GATE Timer2 module gate control signal

6.5.1.3 Port P03 function configuration register (P03_SEL)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P03_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P03_sel	Port P03 function selection. 000: GPIO P03 001: PCA_CH3 PCA module channel 3 capture / compare signal 010: SPI_CS SPI module master mode chip select signal 011: TIM6_CHB Advanced Timer module channel 2 B signal 100: LPTIM_EXT Timer3 module external clock input signal 101: RTC_1Hz RTC module 1Hz output signal 110: PCA_ECI PCA module external clock input signal 111: VC0_OUT VC0 module output

6.5.1.4 Port P0 input and output configuration register (P0DIR)

Offset address: 0x100

Reset value: 0xffff ffff

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												P0DI R3	P0DI R2	P0DI R1	Res
												RW	RW	RW	

Bit	Marking	Functional description
31:4	Reserved	
3	P0DIR3	Port P03 input and output configuration register 1: configured as an input 0: configured as an Output
2	P0DIR2	Port P02 input and output configuration register 1: configured as an input 0: configured as an Output
1	P0DIR1	Port P01 input and output configuration register 1: configured as an input 0: configured as an Output
0	Reserved	

6.5.1.5 Port P0 Input Value Register (P0IN)

Offset address: 0x104

Reset value: NA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												P0IN3	P0IN2	P0IN1	P0IN0
												RO	RO	RO	RO

Bit	Marking	Functional description
31:4	Reserved	
3	P0IN3	Port P03 input value register 1: Input is high level 0: Input is low level
2	P0IN2	Port P02 Input Value Register 1: Input is high level 0: Input is low level
1	P0IN1	Port P01 Input Value Register 1: Input is high level 0: Input is low level
0	P0IN0	Port P00 Input Value Register 1: Input is high level 0: Input is low level

6.5.1.6 Port P0 output value configuration register (P0OUT)

Offset address: 0x108

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												P0OUT3	P0OUT2	P0OUT1	Res
												RW	RW	RW	

Reset value: NA

Bit	Marking	Functional description
31:4	Reserved	
3	P0OUT3	Port P03 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
2	P0OUT2	Port P02 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
1	P0OUT1	Port P01 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
0	Reserved	

6.5.1.7 Port P0 digital-analog configuration register (P0ADS)

Offset address: 0x10C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												P0ADS3	P0ADS2	P0ADS1	Res
												RW	RW	RW	

Bit	Marking	Functional description
31:4	Reserved	
3	P0ADS3	Port P03 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
2	P0ADS2	Port P02 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
1	P0ADS1	Port P01 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
0	Reserved	

6.5.1.8 Port P0 Drive Capability Configuration Register (P0DR)

Offset address: 0x11C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												P0DR3	P0DR2	P0DR1	Res
												RW	RW	RW	

Bit	Marking	Functional description
31:4	Reserved	
3	P0DR3	Port P03 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
2	P0DR2	Port P02 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
1	P0DR1	Port P01 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
0	Reserved	

6.5.1.9 Port P0 pull-up enable configuration register (P0PU)

Offset address: 0x120

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												P0PU 3	P0PU 2	P0PU 1	Res
												RW	RW	RW	

Bit	Marking	Functional description
31:4	Reserved	
3	P0PU3	Port P03 pull-up enable configuration register 1: Enable 0: Prohibited
2	P0PU2	Port P02 pull-up enable configuration register 1: Enable 0: Prohibited
1	P0PU1	Port P01 pull-up enable configuration register 1: Enable 0: Prohibited
0	Reserved	

6.5.1.10 Port P0 pull-down enable configuration register (P0PD)

Offset address: 0x124

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												P0PD 3	P0PD 2	P0PD 1	Res
												RW	RW	RW	

Bit	Marking	Functional description
31:4	Reserved	
3	P0PD3	Port P03 pull-down enable configuration register 1: Enable 0: Prohibited
2	P0PD2	Port P02 pull-down enable configuration register 1: Enable 0: Prohibited
1	P0PD1	Port P01 pull-down enable configuration register 1: Enable 0: Prohibited
0	Reserved	

6.5.1.11 Port P0 open-drain output configuration register (P0OD)

Offset address: 0x12C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												P0OD 3	P0OD 2	P0OD 1	Res
												RW	RW	RW	

Bit	Marking	Functional description
31:4	Reserved	
3	P0OD3	Port P03 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
2	P0OD2	Port P02 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
1	P0OD1	Port P01 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
0	Reserved	

6.5.1.12 Port P0 High Level Interrupt Enable Configuration Register (P0HIE)

Offset address: 0x130

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												P0HIE 3	P0HIE 2	P0HIE 1	P0HIE 0
												RW	RW	RW	RW

Bit	Marking	Functional description
31:4	Reserved	
3	P0HIE3	Port P03 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
2	P0HIE2	Port P02 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
1	P0HIE1	Port P01 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
0	P0HIE0	Port P00 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited

6.5.1.13 Port P0 Low Level Interrupt Enable Configuration Register (POLIE)

Offset address: 0x134

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												POLIE 3	POLIE 2	POLIE 1	POLIE 0
												RW	RW	RW	RW

Bit	Marking	Functional description
31:4	Reserved	
3	POLIE3	Port P03 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
2	POLIE2	Port P02 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
1	POLIE1	Port P01 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
0	POLIE0	Port P00 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited

6.5.1.14 Port P0 Rising Edge Interrupt Enable Configuration Register (PORIE)

Offset address: 0x138

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												PORIE 3	PORIE 2	PORIE 1	PORIE 0
												RW	RW	RW	RW

Bit	Marking	Functional description
31:4	Reserved	
3	PORIE3	Port P03 Rising Edge Interrupt Enable Configuration Register 1: Enable 0: Prohibited
2	PORIE2	Port P02 Rising Edge Interrupt Enable Configuration Register 1: Enable 0: Prohibited
1	PORIE1	Port P01 Rising Edge Interrupt Enable Configuration Register 1: Enable 0: Prohibited
0	PORIE0	Port P00 Rising Edge Interrupt Enable Configuration Register 1: Enable 0: Prohibited

6.5.1.15 Port P0 Falling Edge Interrupt Enable Configuration Register (POFIE)

Offset address: 0x13C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												POFIE 3	POFIE 2	POFIE 1	POFIE 0
												RW	RW	RW	RW

Reset value: 0x0000 0000

Bit	Marking	Functional description
31:4	Reserved	
3	POFIE3	Port P03 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
2	POFIE2	Port P02 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
1	POFIE1	Port P01 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
0	POFIE0	Port P00 falling edge interrupt enable configuration register 1: Enable 0: Prohibited

6.5.1.16 Port P0 Interrupt Status Register (P0_STAT)

Offset address: 0x200

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												POST A3	POST A2	POST A1	POST A0
												RO	RO	RO	RO

Bit	Marking	Functional description
31:4	Reserved	
3	POSTA3	Port P03 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
2	POSTA2	Port P02 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
1	POSTA1	Port P01 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
0	POSTA0	Port P00 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger

6.5.1.17 Port P0 Interrupt Clear Register (P0_ICLR)

Offset address: 0x210

Reset value: 0xffff ffff

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												P0CL R3	P0CL R2	P0CL R1	P0CL R0
												RW	RW	RW	RW

Bit	Marking	Functional description
31:4	Reserved	
3	P0CLR3	Port P03 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
2	P0CLR2	Port P02 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
1	P0CLR1	Port P01 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
0	P0CLR0	Port P00 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit

6.5.2 Port P1

6.5.2.1 Port P14 function configuration register (P14_SEL)

Offset address: 0x50

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P14_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P14_sel	Port P14 function selection. 000: GPIO P14 001: I2C_SCL I2C module clock signal 010: TIM2_TOGN The reverse signal of the Timer2 module flip signal 011: PCA_ECI PCA module external clock input signal 100: ADC_RDY ADC module RDY signal 101: SPI_CS SPI module master mode chip select signal 110: UART0_TXD UART0 module TXD signal 111: NC

6.5.2.2 Port P15 function configuration register (P15_SEL)

Offset address: 0x54

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P15_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P15_sel	Port P15 function selection. 000: GPIO P15 001: I2C_SDA I2C module data signal 010: TIM2_TOG Timer2 module flip signal 011: TIM4_CHB Advanced Timer module channel 0 B signal 100: LPTIM_GATE Timer3 module gate control signal 101: SPI_SCK SPI module clock signal 110: UART0_RXD UART0 module RXD signal 111: LVD_OUT LVD module output signal

6.5.2.3 Port P1 input and output configuration register (P1DIR)

Offset address: 0x140

Reset value: 0xffff ffff

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1DIR5	P1DIR4	Reserved			
										RW	RW				

Bit	Marking	Functional description
31:6	Reserved	
5	P1DIR5	Port P15 input and output configuration register 1: configured as an input 0: configured as an Output
4	P1DIR4	Port P14 input and output configuration register 1: configured as an input 0: configured as an Output
3:0	Reserved	

6.5.2.4 Port P1 Input Value Register (P1IN)

Offset address: 0x144

Reset value: NA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1IN5	P1IN4	Reserved			
										RO	RO				

Bit	Marking	Functional description
31:6	Reserved	
5	P1IN5	Port P15 input value register 1: Input is high level 0: Input is low level
4	P1IN4	Port P14 Input Value Register 1: Input is high level 0: Input is low level
3:0	Reserved	

6.5.2.5 Port P1 output value configuration register (P1OUT)

Offset address: 0x148

Reset value: NA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1OUT5	P1OUT4	Reserved			
										RW	RW				

Bit	Marking	Functional description
31:6	Reserved	
5	P1OUT5	Port P15 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
4	P1OUT4	Port P14 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
3:0	Reserved	

6.5.2.6 Port P1 digital-analog configuration register (P1ADS)

Offset address: 0x14C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1ADS5	P1ADS4	Reserved			
										RW	RW				

Bit	Marking	Functional description
31:6	Reserved	
5	P1ADS5	Port P15 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
4	P1ADS4	Port P14 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
3:0	Reserved	

6.5.2.7 Port P1 Drive Capability Configuration Register (P1DR)

Offset address: 0x15C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1DR 5	P1DR 4	Reserved			
										RW	RW				

Reset value: 0x0000 0000

Bit	Marking	Functional description
31:6	Reserved	
5	P1DR5	Port P15 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
4	P1DR4	Port P14 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
3:0	Reserved	

6.5.2.8 Port P1 pull-up enable configuration register (P1PU)

Offset address: 0x160

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1PU 5	P1PU 4	Reserved			
										RW	RW				

Bit	Marking	Functional description
31:6	Reserved	
5	P1PU5	Port P15 pull-up enable configuration register 1: Enable 0: Prohibited
4	P1PU4	Port P14 pull-up enable configuration register 1: Enable 0: Prohibited
3:0	Reserved	

6.5.2.9 Port P1 pull-down enable configuration register (P1PD)

Offset address: 0x164

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1PD 5	P1PD 4	Reserved			
										RW	RW				

Bit	Marking	Functional description
31:6	Reserved	
5	P1PD5	Port P15 pull-down enable configuration register 1: Enable 0: Prohibited
4	P1PD4	Port P14 pull-down enable configuration register 1: Enable 0: Prohibited
3:0	Reserved	

6.5.2.10 Port P1 open-drain output configuration register (P1OD)

Offset address: 0x16C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1OD 5	P1OD 4	Reserved			
										RW	RW				

Bit	Marking	Functional description
31:6	Reserved	
5	P1OD5	Port P15 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
4	P1OD4	Port P14 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
3:0	Reserved	

6.5.2.11 Port P1 High Level Interrupt Enable Configuration Register (P1HIE)

Offset address: 0x170

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1HIE 5	P1HIE 4	Reserved			
										RW	RW				

Bit	Marking	Functional description
31:6	Reserved	
5	P1HIE5	Port P15 High Level interrupt enable configuration register 1: Enable 0: Prohibited
4	P1HIE4	Port P14 high level interrupt enable configuration register 1: Enable 0: Prohibited
3:0	Reserved	

6.5.2.12 Port P1 Low Level Interrupt Enable Configuration Register (P1LIE)

Offset address: 0x174

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1LIE 5	P1LIE 4	Reserved			
										RW	RW				

Reset value: 0x0000 0000

Bit	Marking	Functional description
31:6	Reserved	
5	P1LIE5	Port P15 low level interrupt enable configuration register 1: Enable 0: Prohibited
4	P1LIE4	Port P14 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
3:0	Reserved	

6.5.2.13 Port P1 Rising Edge Interrupt Enable Configuration Register (P1RIE)

Offset address: 0x178

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1RIE 5	P1RIE 4	Reserved			
										RW	RW				

Bit	Marking	Functional description
31:6	Reserved	
5	P1RIE5	Port P15 rising edge interrupt enable configuration register 1: Enable 0: Prohibited
4	P1RIE4	Port P14 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
3:0	Reserved	

6.5.2.14 Port P1 Falling Edge Interrupt Enable Configuration Register (P1FIE)

Offset address: 0x17C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1FIE 5	P1FIE 4	Reserved			
										RW	RW				

Bit	Marking	Functional description
31:6	Reserved	
5	P1FIE5	Port P15 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
4	P1FIE4	Port P14 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
3:0	Reserved	

6.5.2.15 Port P1 Interrupt Status Register (P1_STAT)

Offset address: 0x240

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1ST A5	P1ST A4	Reserved			
										RO	RO				

Bit	Marking	Functional description
31:6	Reserved	
5	P1STA5	Port P15 Interrupt status register 1: interrupt trigger 0: no interrupt trigger
4	P1STA4	Port P14 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
3:0	Reserved	

6.5.2.16 Port P1 Interrupt Clear Register (P1_ICLR)

Offset address: 0x250

Reset value: 0xffff ffff

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										P1CL R5	P1CL R4	Reserved			
										RW	RW				

Bit	Marking	Functional description
31:6	Reserved	
5	P1CLR5	Port P15 Interrupt clear register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
4	P1CLR4	Port P14 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
3:0	Reserved	

6.5.3 Port P2

6.5.3.1 Port P23 function configuration register (P23_SEL)

Offset address: 0x8C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P23_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P23_sel	Port P23 function selection. 000: GPIO P23 001: TIM6_CHA Advanced Timer module channel 2 A signal 010: TIM4_CHB Advanced Timer module channel 0 B signal 011: TIM4_CHA Advanced Timer module channel 0 A signal 100: PCA_CH0 PCA module channel 0 capture / compare signal 101: SPI_MOSI SPI module master input slave output data signal 110: UART1_TXD UART1 module TXD signal 111: IR_OUT infrared output signal

6.5.3.2 Port P24 function configuration register (P24_SEL)

Offset address: 0x90

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P24_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P24_sel	Port P24 function selection. 000: GPIO P24 001: TIM4_CHB Advanced Timer module channel 0 B signal 010: TIM5_CHB Advanced Timer module channel 1 B signal 011: HCLK_OUT AHB bus clock output signal 100: PCA_CH1 PCA module channel 1 capture / comparison signal 101: SPI_MOSI SPI module master output slave input data signal 110: UART1_RXD UART1 module RXD signal 111: VC1_OUT VC1 module output

6.5.3.3 Port P25 function configuration register (P25_SEL)

Offset address: 0x94

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P25_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P25_sel	Port P25 function selection. 000: GPIO P25 001: SPI_SCK SPI module clock signal 010: PCA_CH0 PCA module channel 0 capture / compare signal 011: TIM5_CHA Advanced Timer module channel 1 A signal 100: LVD_OUT LVD module output signal 101: LPUART_RXD LPUART module RXD signal 110: I2C_SDA I2C module data signal 111: TIM1_GATE Timer1 module gate control signal

6.5.3.4 Port P26 function configuration register (P26_SEL)

Offset address: 0x98

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P26_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P26_sel	Port P26 function selection. 000: GPIO P26 001: SPI_MOSI SPI module master output slave input data signal 010: TIM4_CHA Advanced Timer module channel 0 A signal 011: TIM5_CHB Advanced Timer module channel 1 B signal 100: PCA_CH2 PCA module channel 2 capture / comparison signal 101: LPUART_TXD LPUART module TXD signal 110: I2C_SCL I2C module clock signal 111: TIM1_EXT Timer1 module external clock input signal

6.5.3.5 Port P27 function configuration register (P27_SEL)

Offset address: 0x9C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P27_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P27_sel	Port P27 function selection. 000: GPIO P27 001: SPI_MOSI SP I module master input slave output data signal 010: TIM5_CHA Advanced Timer module channel 1 A signal 011: TIM6_CHA Advanced Timer module channel 2 A signal 100: PCA_CH3 PCA module channel 3 capture / compare signal 101: UART0_RXD UART0 module RXD signal 110: RCH_OUT internal 24M RC clock output signal 111: XTH_OUT external 32M crystal oscillator output signal

6.5.3.6 Port P2 input and output configuration register (P2DIR)

Offset address: 0x180

Reset value: 0xffff ffff

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2DI R7	P2DI R6	P2DI R5	P2DI R4	P2DI R3	Reserved		
								RW	RW	RW	RW	RW			

Bit	Marking	Functional description
31:8	Reserved	
7	P2DIR7	Port P27 input and output configuration register 1: configured as an input 0: configured as an Output
6	P2DIR6	Port P26 input and output configuration register 1: configured as an input 0: configured as an Output
5	P2DIR5	Port P25 input and output configuration register 1: configured as an input 0: configured as an Output
4	P2DIR4	Port P24 input and output configuration register 1: configured as an input 0: configured as an Output
3	P2DIR3	Port P23 input and output configuration register 1: configured as an input 0: configured as an Output
2:0	Reserved	

6.5.3.7 Port P2 Input Value Register (P2IN)

Offset address: 0x184

Reset value: NA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2IN7	P2IN6	P2IN5	P2IN4	P2IN3	Reserved		
								RO	RO	RO	RO	RO			

Bit	Marking	Functional description
31:8	Reserved	
7	P2IN7	Port P27 Input Value Register 1: Input is high level 0: Input is low level
6	P2IN6	Port P26 Input Value Register 1: Input is high level 0: Input is low level
5	P2IN5	Port P25 Input Value Register 1: Input is high level 0: Input is low level
4	P2IN4	Port P24 Input Value Register 1: Input is high level 0: Input is low level
3	P2IN3	Port P23 Input Value Register 1: Input is high level 0: Input is low level
2:0	Reserved	

6.5.3.8 Port P2 output value configuration register (P2OUT)

Offset address: 0x188

Reset value: NA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2OU T7	P2OU T6	P2OU T5	P2OU T4	P2OU T3	Reserved		
								RW	RW	RW	RW	RW			

Bit	Marking	Functional description
31:8	Reserved	
7	P2OUT7	Port P27 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
6	P2OUT6	Port P26 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
5	P2OUT5	Port P25 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
4	P2OUT4	Port P24 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
3	P2OUT3	Port P23 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
2:0	Reserved	

6.5.3.9 Port P2 digital-analog configuration register (P2ADS)

Offset address: 0x18C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2AD S7	P2AD S6	P2AD S5	P2AD S4	P2AD S3	Reserved		
								RW	RW	RW	RW	RW			

Bit	Marking	Functional description
31:8	Reserved	
7	P2ADS7	Port P27 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
6	P2ADS6	Port P26 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
5	P2ADS5	Port P25 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
4	P2ADS4	Port P24 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
3	P2ADS3	Port P23 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
2:0	Reserved	

6.5.3.10 Port P2 Drive Capability Configuration Register (P2DR)

Offset address: 0x19C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2DR 7	P2DR 6	P2DR 5	P2DR 4	P2DR 3	Reserved		
								RW	RW	RW	RW	RW			

Bit	Marking	Functional description
31:8	Reserved	
7	P2DR7	Port P27 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
6	P2DR6	Port P26 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
5	P2DR5	Port P25 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
4	P2DR4	Port P24 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
3	P2DR3	Port P23 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
2:0	Reserved	

6.5.3.11 Port P2 pull-up enable configuration register (P2PU)

Offset address: 0x1A0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2PU 7	P2PU 6	P2PU 5	P2PU 4	P2PU 3	Reserved		
								RW	RW	RW	RW	RW			

Bit	Marking	Functional description
31:8	Reserved	
7	P2PU7	Port P27 pull-up enable configuration register 1: Enable 0: Prohibited
6	P2PU6	Port P26 pull-up enable configuration register 1: Enable 0: Prohibited
5	P2PU5	Port P25 pull-up enable configuration register 1: Enable 0: Prohibited
4	P2PU4	Port P24 pull-up enable configuration register 1: Enable 0: Prohibited
3	P2PU3	Port P23 pull-up enable configuration register 1: Enable 0: Prohibited
2:0	Reserved	

6.5.3.12 Port P2 pull-down enable configuration register (P2PD)

Offset address: 0x1A4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2PD 7	P2PD 6	P2PD 5	P2PD 4	P2PD 3	Reserved		
								RW	RW	RW	RW	RW			

Bit	Marking	Functional description
31:8	Reserved	
7	P2PD7	Port P27 pull-down enable configuration register 1: Enable 0: Prohibited
6	P2PD6	Port P26 pull-down enable configuration register 1: Enable 0: Prohibited
5	P2PD5	Port P25 pull-down enable configuration register 1: Enable 0: Prohibited
4	P2PD4	Port P24 pull-down enable configuration register 1: Enable 0: Prohibited
3	P2PD3	Port P23 pull-down enable configuration register 1: Enable 0: Prohibited
2:0	Reserved	

6.5.3.13 Port P2 open-drain output configuration register (P2OD)

Offset address: 0x1AC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2OD 7	P2OD 6	P2OD 5	P2OD 4	P2OD 3	Reserved		
								RW	RW	RW	RW	RW			

Bit	Marking	Functional description
31:8	Reserved	
7	P2OD7	Port P27 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
6	P2OD6	Port P26 Open Drain Output Configuration Register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
5	P2OD5	Port P25 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
4	P2OD4	Port P24 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
3	P2OD3	Port P23 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
2:0	Reserved	

6.5.3.14 Port P2 High Level Interrupt Enable Configuration Register (P2HIE)

Offset address: 0x1B0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2HIE 7	P2HIE 6	P2HIE 5	P2HIE 4	P2HIE 3	Reserved		
								RW	RW	RW	RW	RW			

Bit	Marking	Functional description
31:8	Reserved	
7	P2HIE7	Port P27 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
6	P2HIE6	Port P26 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
5	P2HIE5	Port P25 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
4	P2HIE4	Port P24 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
3	P2HIE3	Port P23 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
2:0	Reserved	

6.5.3.15 Port P2 Low Level Interrupt Enable Configuration Register (P2LIE)

Offset address: 0x1B4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2LIE 7	P2LIE 6	P2LIE 5	P2LIE 4	P2LIE 3	Reserved		
								RW	RW	RW	RW	RW			

Bit	Marking	Functional description
31:8	Reserved	
7	P2LIE7	Port P27 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
6	P2LIE6	Port P26 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
5	P2LIE5	Port P25 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
4	P2LIE4	Port P24 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
3	P2LIE3	Port P23 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
2:0	Reserved	

6.5.3.16 Port P2 Rising Edge Interrupt Enable Configuration Register (P2RIE)

Offset address: 0x1B8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2RIE 7	P2RIE 6	P2RIE 5	P2RIE 4	P2RIE 3	Reserved		
								RW	RW	RW	RW	RW			

Bit	Marking	Functional description
31:8	Reserved	
7	P2RIE7	Port P27 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
6	P2RIE6	Port P26 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
5	P2RIE5	Port P25 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
4	P2RIE4	Port P24 Rising Edge Interrupt Enable Configuration Register 1: Enable 0: Prohibited
3	P2RIE3	Port P23 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
2:0	Reserved	

6.5.3.17 Port P2 Falling Edge Interrupt Enable Configuration Register (P2FIE)

Offset address: 0x1BC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2FIE 7	P2FIE 6	P2FIE 5	P2FIE 4	P2FIE 3	Reserved		
								RW	RW	RW	RW	RW			

Reset value: 0x0000 0000

Bit	Marking	Functional description
31:8	Reserved	
7	P2FIE7	Port P27 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
6	P2FIE6	Port P26 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
5	P2FIE5	Port P25 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
4	P2FIE4	Port P24 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
3	P2FIE3	Port P23 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
2:0	Reserved	

6.5.3.18 Port P2 Interrupt Status Register (P2_STAT)

Offset address: 0x280

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2ST A7	P2ST A6	P2ST A5	P2ST A4	P2ST A3	Reserved		
								RO	RO	RO	RO	RO			

Bit	Marking	Functional description
31:8	Reserved	
7	P2STA7	Port P27 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
6	P2STA6	Port P26 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
5	P2STA5	Port P25 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
4	P2STA4	Port P24 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
3	P2STA3	Port P23 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
2:0	Reserved	

6.5.3.19 Port P2 Interrupt Clear Register (P2_ICLR)

Offset address: 0x290

Reset value: 0xffff ffff

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P2CL R7	P2CL R6	P2CL R5	P2CL R4	P2CL R3	Reserved		
								RW	RW	RW	RW	RW			

Bit	Marking	Functional description
31:8	Reserved	
7	P2CLR7	Port P27 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
6	P2CLR6	Port P26 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
5	P2CLR5	Port P25 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
4	P2CLR4	Port P24 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit

3	P2CLR3	Port P23 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
2:0	Reserved	

6.5.4 Port P3

6.5.4.1 Port P31 function configuration register (P31_SEL)

Offset address: 0xC4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P31_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P31_sel	Port P31 function selection. 000: GPIO P31 001: LPTIM_TOG Timer3 module toggle signal 010: PCA_ECI PCA module external clock input signal 011: PCLK_OUT APB bus clock output signal 100: VCO_OUT VCO module output 101: UART0_TXD UART0 module TXD signal 110: RCL_OUT internal 38K RC clock output signal 111: HCLK_OUT AHB bus clock output signal

6.5.4.2 Port P32 function configuration register (P32_SEL)

Offset address: 0xC8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P32_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P32_sel	Port P32 function selection. 000: GPIO P32 001: LPTIM_TOGN The reverse signal of the Timer3 module flip signal 010: PCA_CH2 PCA module channel 2 capture/comparison signal 011: TIM6_CHB Advanced Timer module channel 2 B signal 100: VC1_OUT VC1 module output 101: UART1_TXD UART1 module TXD signal 110: PCA_CH4 PCA module channel 4 capture/comparison signal 111: RTC_1Hz RTC module 1Hz output signal

6.5.4.3 Port P33 function configuration register (P33_SEL)

Offset address: 0xCC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P33_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P33_sel	Port P33 function selection. 000: GPIO P33 001: LPUART_RXD LPUART module RXD signal 010: PCA_CH1 PCA module channel 1 capture / comparison signal 011: TIM5_CHB Advanced Timer module channel 1 B signal 100: PCA_ECI PCA module external clock input signal 101: UART1_RXD UART1 module RXD signal 110: XTL_OUT external 32K crystal oscillator output signal 111: TIM1_TOGN The reverse signal of the Timer1 module flip signal

6.5.4.4 Port P34 function configuration register (P34_SEL)

Offset address: 0xD0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P34_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P34_sel	Port P34 function selection. 000: GPIO P34 001: PCA_CH0 PCA module channel 0 capture / compare signal 010: LPUART_TXD LPUART module TXD signal 011: TIM5_CHA Advanced Timer module channel 1 A signal 100: TIM0_EXT Timer0 module external clock input signal 101: TIM4_CHA Advanced Timer module channel 0 A signal 110: RTC_1Hz RTC module 1Hz output signal 111: TIM1_TOG Timer1 module toggle signal

6.5.4.5 Port P35 function configuration register (P35_SEL)

Offset address: 0xD4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P35_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P35_sel	Port P35 function selection. 000: GPIO P35 001: UART1_TXD UART1 module TXD signal 010: TIM6_CHB Advanced Timer module channel 2 B signal 011: UART0_TXD UART0 module TXD signal 100: TIM0_GATE Timer0 module gate control signal 101: TIM4_CHB Advanced Timer module channel 0 B signal 110: SPI_MOSI SPI module master input slave output data signal 111: I2C_SDA I2C module data signal

6.5.4.6 Port P36 function configuration register (P36_SEL)

Offset address: 0xD8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													P36_sel		
													RW		

Bit	Marking	Functional description
31:3	Reserved	
2:0	P36_sel	Port P36 function selection. 000: GPIO P36 001: UART1_RXD UART1 module RXD signal 010: TIM6_CHA Advanced Timer module channel 2 A signal 011: UART0_RXD UART0 module RXD signal 100: PCA_CH4 PCA module channel 4 capture / compare signal 101: TIM5_CHA Advanced Timer module channel 1 A signal 110: SPI_MOSI SPI module master output slave input data signal 111: I2C_SCL I2C module clock signal

6.5.4.7 Port P3 input and output configuration register (P3DIR)

Offset address: 0x1C0

Reset value: 0xffff ffff

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3DI R6	P3DI R5	P3DI R4	P3DI R3	P3DI R2	P3DI R1	Res
									RW	RW	RW	RW	RW	RW	

Bit	Marking	Functional description
31:7	Reserved	
6	P3DIR6	Port P36 input and output configuration register 1: configured as an input 0: configured as an Output
5	P3DIR5	Port P35 input and output configuration register 1: configured as an input 0: configured as an Output
4	P3DIR4	Port P34 input and output configuration register 1: configured as an input 0: configured as an Output
3	P3DIR3	Port P33 input and output configuration register 1: configured as an input 0: configured as an Output
2	P3DIR2	Port P32 input and output configuration register 1: configured as an input 0: configured as an Output
1	P3DIR1	Port P31 input and output configuration register 1: configured as an input 0: configured as an Output
0	Reserved	

6.5.4.8 Port P3 Input Value Register (P3IN)

Offset address: 0x1C4

Reset value: NA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3IN6	P3IN5	P3IN4	P3IN3	P3IN2	P3IN1	Res
									RO	RO	RO	RO	RO	RO	

Bit	Marking	Functional description
31:7	Reserved	
6	P3IN6	Port P36 Input Value Register 1: Input is high level 0: Input is low level
5	P3IN5	Port P35 Input Value Register 1: Input is high level 0: Input is low level
4	P3IN4	Port P34 Input Value Register 1: Input is high level 0: Input is low level
3	P3IN3	Port P33 Input Value Register 1: Input is high level 0: Input is low level
2	P3IN2	Port P32 Input Value Register 1: Input is high level 0: Input is low level
1	P3IN1	Port P31 Input Value Register 1: Input is high level 0: Input is low level
0	Reserved	

6.5.4.9 Port P3 output value configuration register (P3OUT)

Offset address: 0x1C8

Reset value: NA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3OUT6	P3OUT5	P3OUT4	P3OUT3	P3OUT2	P3OUT1	Res
									RW	RW	RW	RW	RW	RW	

Bit	Marking	Functional description
31:7	Reserved	
6	P3OUT6	Port P36 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
5	P3OUT5	Port P35 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
4	P3OUT4	Port P34 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
3	P3OUT3	Port P33 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
2	P3OUT2	Port P32 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
1	P3OUT1	Port P31 output value configuration register 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level
0	Reserved	

6.5.4.10 Port P3 digital-analog configuration register (P3ADS)

Offset address: 0x1CC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3AD S6	P3AD S5	P3AD S4	P3AD S3	P3AD S2	P3AD S1	Res
									RW	RW	RW	RW	RW	RW	

Bit	Marking	Functional description
31:7	Reserved	
6	P3ADS6	Port P36 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
5	P3ADS5	Port P35 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
4	P3ADS4	Port P34 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
3	P3ADS3	Port P33 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
2	P3ADS2	Port P32 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
1	P3ADS1	Port P31 digital-analog configuration register 1: configured as an analog port 0: configured as a digital port
0	Reserved	

6.5.4.11 Port P3 Drive Capability Configuration Register (P3DR)

Offset address: 0x1DC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3DR 6	P3DR 5	P3DR 4	P3DR 3	P3DR 2	P3DR 1	Res
									RW	RW	RW	RW	RW	RW	

Reset value: 0x0000 0000

Bit	Marking	Functional description
31:7	Reserved	
6	P3DR6	Port P36 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
5	P3DR5	Port P35 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
4	P3DR4	Port P34 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
3	P3DR3	Port P33 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
2	P3DR2	Port P32 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
1	P3DR1	Port P31 Drive Capability Configuration Register 1: Low drive capability 0: High drive capability
0	Reserved	

6.5.4.12 Port P3 pull-up enable configuration register (P3PU)

Offset address: 0x1E0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3PU 6	P3PU 5	P3PU 4	P3PU 3	P3PU 2	P3PU 1	Res
									RW	RW	RW	RW	RW	RW	

Bit	Marking	Functional description
31:7	Reserved	
6	P3PU6	Port P36 pull-up enable configuration register 1: Enable 0: Prohibited
5	P3PU5	Port P35 pull-up enable configuration register 1: Enable 0: Prohibited
4	P3PU4	Port P34 pull-up enable configuration register 1: Enable 0: Prohibited
3	P3PU3	Port P33 pull-up enable configuration register 1: Enable 0: Prohibited
2	P3PU2	Port P32 pull-up enable configuration register 1: Enable 0: Prohibited
1	P3PU1	Port P31 pull-up enable configuration register 1: Enable 0: Prohibited
0	Reserved	

6.5.4.13 Port P3 pull-down enable configuration register (P3PD)

Offset address: 0x1E4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3PD 6	P3PD 5	P3PD 4	P3PD 3	P3PD 2	P3PD 1	Res
									RW	RW	RW	RW	RW	RW	

Bit	Marking	Functional description
31:7	Reserved	
6	P3PD6	Port P36 pull-down enable configuration register 1: Enable 0: Prohibited
5	P3PD5	Port P35 pull-down enable configuration register 1: Enable 0: Prohibited
4	P3PD4	Port P34 pull-down enable configuration register 1: Enable 0: Prohibited
3	P3PD3	Port P33 pull-down enable configuration register 1: Enable 0: Prohibited
2	P3PD2	Port P32 pull-down enable configuration register 1: Enable 0: Prohibited
1	P3PD1	Port P31 pull-down enable configuration register 1: Enable 0: Prohibited
0	Reserved	

6.5.4.14 Port P3 open-drain output configuration register (P3OD)

Offset address: 0x1EC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3OD 6	P3OD 5	P3OD 4	P3OD 3	P3OD 2	P3OD 1	Res
									RW	RW	RW	RW	RW	RW	

Bit	Marking	Functional description
31:7	Reserved	
6	P3OD6	Port P36 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
5	P3OD5	Port P35 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
4	P3OD4	Port P34 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
3	P3OD3	Port P33 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
2	P3OD2	Port P32 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
1	P3OD1	Port P31 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
0	Reserved	

6.5.4.15 Port P3 High Level Interrupt Enable Configuration Register (P3HIE)

Offset address: 0x1F0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3HIE 6	P3HIE 5	P3HIE 4	P3HIE 3	P3HIE 2	P3HIE 1	Res
									RW	RW	RW	RW	RW	RW	

Bit	Marking	Functional description
31:7	Reserved	
6	P3HIE6	Port P36 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
5	P3HIE5	Port P35 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
4	P3HIE4	Port P34 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
3	P3HIE3	Port P33 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
2	P3HIE2	Port P32 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
1	P3HIE1	Port P31 High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
0	Reserved	

6.5.4.16 Port P3 Low Level Interrupt Enable Configuration Register (P3LIE)

Offset address: 0x1F4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3LIE 6	P3LIE 5	P3LIE 4	P3LIE 3	P3LIE 2	P3LIE 1	Res
									RW	RW	RW	RW	RW	RW	

Bit	Marking	Functional description
31:7	Reserved	
6	P3LIE6	Port P36 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
5	P3LIE5	Port P35 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
4	P3LIE4	Port P34 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
3	P3LIE3	Port P33 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
2	P3LIE2	Port P32 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
1	P3LIE1	Port P31 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
0	Reserved	

6.5.4.17 Port P3 Rising Edge Interrupt Enable Configuration Register (P3RIE)

Offset address: 0x1F8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3RIE 6	P3RIE 5	P3RIE 4	P3RIE 3	P3RIE 2	P3RIE 1	Res
									RW	RW	RW	RW	RW	RW	

Bit	Marking	Functional description
31:7	Reserved	
6	P3RIE6	Port P36 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
5	P3RIE5	Port P35 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
4	P3RIE4	Port P34 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
3	P3RIE3	Port P33 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
2	P3RIE2	Port P32 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
1	P3RIE1	Port P31 Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited
0	Reserved	

6.5.4.18 Port P3 Falling Edge Interrupt Enable Configuration Register (P3FIE)

Offset address: 0x1FC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3FIE 6	P3FIE 5	P3FIE 4	P3FIE 3	P3FIE 2	P3FIE 1	Res
									RW	RW	RW	RW	RW	RW	

Bit	Marking	Functional description
31:7	Reserved	
6	P3FIE6	Port P36 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
5	P3FIE5	Port P35 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
4	P3FIE4	Port P34 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
3	P3FIE3	Port P33 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
2	P3FIE2	Port P32 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
1	P3FIE1	Port P31 falling edge interrupt enable configuration register 1: Enable 0: Prohibited
0	Reserved	

6.5.4.19 Port P3 Interrupt Status Register (P3_STAT)

Offset address: 0x2C0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3ST A6	P3ST A5	P3ST A4	P3ST A3	P3ST A2	P3ST A1	Res
									RO	RO	RO	RO	RO	RO	

Bit	Marking	Functional description
31:7	Reserved	
6	P3STA6	Port P36 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
5	P3STA5	Port P35 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
4	P3STA4	Port P34 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
3	P3STA3	Port P33 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
2	P3STA2	Port P32 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
1	P3STA1	Port P31 Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger
0	Reserved	

6.5.4.20 Port P3 Interrupt Clear Register (P3_ICLR)

Offset address: 0x2D0

Reset value: 0xffff ffff

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									P3CL R6	P3CL R5	P3CL R4	P3CL R3	P3CL R2	P3CL R1	Res
									RW	RW	RW	RW	RW	RW	

Bit	Marking	Functional description
31:7	Reserved	
6	P3CLR6	Port P36 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
5	P3CLR5	Port P35 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
4	P3CLR4	Port P34 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
3	P3CLR3	Port P33 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
2	P3CLR2	Port P32 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
1	P3CLR1	Port P31 Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit
0	Reserved	

6.5.5 Port auxiliary Function

6.5.5.1 Port auxiliary Function configuration register 1 (GPIO_CTRL1)

Offset address: 0x304

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	ir_pol	hclk_en	pclk_en	hclk_sel		pclk_sel		ssn_sel				ext_clk_sel			
	RW	RW	RW	RW		RW		RW				RW			

Bit	Marking	Functional description
31:15	Reserved	
14	ir_pol	IR output polarity selection. 0 - positive output 1 - Reverse output
13	hclk_en	hclk output control. 0 - outputs 0 1 - output hclk
12	pclk_en	pclk output gating. 0 - outputs 0 1 - output pclk
11:10	hclk_sel	hclk output frequency division selection 00: hclk 01: hclk/2 10: hclk/4 11: hclk/8
9:8	pclk_sel	pclk output frequency division selection 00: hclk 01: hclk/2 10: hclk/4 11: hclk/8
7:4	ssn_sel	SPI SSN signal source selection 0000: high level 1000: P23 0001: P35 1001: P24 0010: P36 1010: P25 0011: P01 1011: P26 0100: P02 1100: P27 0101: P03 1101: P31 0110: P15 1110: P32 0111: P14 1111: P33
3:0	ext_clk_sel	External clock signal source selection 0000: high level 1000: P23 0001: P35 1001: P24 0010: P36 1010: P25 0011: P01 1011: P26 0100: P02 1100: P27 0101: P03 1101: P31 0110: P15 1110: P32 0111: P14 1111: P33

--	--	--

6.5.5.2 Port auxiliary function configuration register 2 (GPIO_CTRL2)

Offset address: 0x308

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						pca_cap4_sel		pca_cap3_sel		pca_cap2_sel		pca_cap1_sel		pca_cap0_sel	
						RW		RW		RW		RW		RW	

Bit	Marking	Functional description
31:10	Reserved	
9:8	pca_cap4_sel	PCA capture channel 4 signal source selection 00: PCA_CH4 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
7:6	pca_cap3_sel	PCA capture channel 3 signal source selection 00: PCA_CH3 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
5:4	pca_cap2_sel	PCA capture channel 2 signal source selection 00: PCA_CH2 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
3:2	pca_cap1_sel	PCA capture channel 1 signal source selection 00: PCA_CH1 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
1:0	pca_cap0_sel	PCA capture channel 0 signal source selection 00: PCA_CH0 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD

6.5.5.3 Port auxiliary Function configuration register 3 (GPIO_CTRL3)

Offset address: 0x30C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				tm6_a_sel		tm5_a_sel		tm4_a_sel		tm6_b_sel		tm5_b_sel		tm4_b_sel	
				RW		RW		RW		RW		RW		RW	

Bit	Marking	Functional description
31:12	Reserved	
11:10	tm6_a_sel	Timer6 A channel signal source selection 00: TIM6_CHA 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
9:8	tm5_a_sel	Timer5 A channel signal source selection 00: TIM5_CHA 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
7:6	tm4_a_sel	Timer4 A channel signal source selection 00: TIM4_CHA 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
5:4	tm6_b_sel	Timer6 B channel signal source selection 00: TIM6_CHB 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
3:2	tm5_b_sel	Timer5 B channel signal source selection 00: TIM5_CHB 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
1:0	tm4_b_sel	Timer4 B channel signal source selection 00: TIM4_CHB 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD

6.5.5.4 Port auxiliary Function configuration register 4 (GPIO_CTRL4)

Offset address: 0x310

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								tm3_gate_sel	tm2_gate_sel	tm1_gate_sel	tm0_gate_sel				
								RW	RW	RW	RW				

Bit	Marking	Functional description
31:8	Reserved	
7:6	tm3_gate_sel	Timer3 gate input signal source selection 00: LPTIM_GATE 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
5:4	tm2_gate_sel	Timer2 gate input signal source selection 00: TIM2_GATE 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
3:2	tm1_gate_sel	Timer1 gate input signal source selection 00: TIM1_GATE 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD
1:0	tm0_gate_sel	Timer0 gate input signal source selection 00: TIM0_GATE 01: UART0_RXD 10: UART1_RXD 11: LPUART_RXD

7 FLASH controller (FLASH)

7.1 Overview

This device contains a FLASH memory with a capacity of 32kByte, which is divided into 64 Sectors, and each Sector has a capacity of 512Byte. This module supports erase, program and read operations to this memory. In addition, this module supports the protection of FLASH memory erase and write, and the write protection of control registers.

7.2 Structural block diagram

Address range	Sector serial number		Address range	Sector serial number
0x1E00-0x1FFF	Sector15		0x7E00-0x7FFF	Sector63
0x1C00-0x1DFF	Sector14		0x7C00-0x7DFF	Sector62
0x1A00-0x1BFF	Sector13		0x7A00-0x7BFF	Sector61
0x1800-0x19FF	Sector12		0x7800-0x79FF	Sector60
0x1600-0x17FF	Sector11		0x7600-0x77FF	Sector59
0x1400-0x15FF	Sector10		0x7400-0x75FF	Sector58
0x1200-0x13FF	Sector9		0x7200-0x73FF	Sector57
0x1000-0x11FF	Sector8		0x7000-0x71FF	Sector56
0x0E00-0x0FFF	Sector7		0x6E00-0x6FFF	Sector55
0x0C00-0x0DFF	Sector6		0x6C00-0x6DFF	Sector54
0x0A00-0x0BFF	Sector5		0x6A00-0x6BFF	Sector53
0x0800-0x09FF	Sector4		0x6800-0x69FF	Sector52
0x0600-0x07FF	Sector3		0x6600-0x67FF	Sector51
0x0400-0x05FF	Sector2		0x6400-0x65FF	Sector50
0x0200-0x03FF	Sector1		0x6200-0x63FF	Sector49
0x0000-0x01FF	Sector0		0x6000-0x61FF	Sector48

Figure 7-1 Memory Sector Division

7.3 Functional description

This controller supports three bit width mode FLASH read and write: Byte (8bits), Half-word (16bits), Word (32bits). Note that the address of the Byte operation must be Byte- aligned, the target address of the Half-word operation must be aligned by Half-word (the lowest bit of the address is 0), the address of the Word operation must be aligned by Word (the lowest two bits of the address are 0). If the address of the read and write operation is not aligned according to the above method, the system will enter the Hard Fault error interrupt.

7.3.1 Sector Erase (Sector Erase)

Page erase can erase a page (sector) specified by the user at a time. After the erase operation is completed, the data in the page (Sector) is 0xFF. If the erase operation is executed 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 executed from RAM, the CPU will not stop fetching means, user software should wait for the operation to complete (FLASH_CR.BUSY becomes 0).

Page (Sector) erase operation steps are as follows:

Step1: Configure FLASH erasing parameters, see the erasing timing chapter for details.

Step2: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step3: Configure FLASH_CR. OP to 2, and set the Flash operation mode to Sector Erase.

Step4: Check whether FLASH_CR.OP is 2, if not, jump to Step2.

Step5: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step6: Set the corresponding bit of FLASH_SLOCK to 1 to remove the sector 's erasure protection.

Step7: Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, jump to Step5.

Step8: Write arbitrary data to any address in the Sector to be erased, triggering Sector erasure.

*Example: * ((unsigned char *) 0x00000200) = 0x00.*

Step9: Wait for FLASH_CR. BUSY to become 0, and the Sector erase operation is completed.

Step10: To erase other sectors, repeat Step5 – Step9.

7.3.2 Full Chip Erase (Chip Erase)

Full chip erase can erase all pages (Sector) at one time. After the erase operation is completed, the data in all pages (Sector) are 0xFF. If the erase operation is executed 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 executed from RAM, the CPU will not stop fetching means, user software should wait for the operation to complete (FLASH_CR.BUSY becomes 0).

The full chip erase operation steps are as follows:

Step1: Configure FLASH erasing parameters, see the erasing timing chapter for details.

Step2: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step3: Configure FLASH_CR. OP to 3, and set the Flash operation mode to Chip erase.

Step4: Check whether FLASH_CR. OP is 3, if not, jump to Step2.

Step5: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step6: Set FLASH_SLOCK to 0xFFFF to remove the erase protection of all Sectors.

Step7: Check if FLASH_SLOCK is 0xFFFF, if it is not 0xFFFF, jump to Step5.

Step8: Write any address in the Chip to be erased to trigger Chip erasure.

*Example: *((unsigned char *) 0x00000000) = 0x00.*

Step9: Wait for FLASH_CR. BUSY to become 0, and the Chip erase operation is completed.

7.3.3 Write operation (Program)

The write operation 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 writing 3 data 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 write operation is executed 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 write operation is executed from RAM, the CPU will not stop fetching instructions, and the user Software should wait for the operation to complete (FLASH_CR. BUSY goes to 0).

Byte write operation steps are as follows:

Step1: Configure FLASH erasing parameters, see the erasing timing chapter for details.

Step2: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step3: Configure FLASH_CR. OP to 1, and set the Flash operation mode to write.

Step4: Check whether FLASH_CR.OP is 1, if not, jump to Step2.

Step5: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step6: Set the corresponding bit of FLASH_SLOCK to 1 to remove the erase protection.

Step7: Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, jump to Step5.

Step8: Perform Byte write operation on the target address to be written to trigger the write operation.

*Example: *((unsigned char *) 0x00001231) = 0x5A.*

Step9: Wait for FLASH_CR. BUSY to become 0, and the write operation is completed.

Step10: If you need to write Byte to other addresses that have removed the erase protection, repeat Step8 - Step9.

Half-word write operation steps are as follows:

- Step1:** Configure the FLASH erasing time, see the erasing timing chapter for details.
- Step2:** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step3:** Configure FLASH_CR.OP to 1, and set the Flash operation mode to write.
- Step4:** Check whether FLASH_CR.OP is 1, if not, jump to Step2.
- Step5:** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step6:** Set the corresponding bit of FLASH_SLOCK to 1 to remove the erase protection.
- Step7:** Check whether the corresponding bit of FLASH_SLOCK is 1, if not, jump to Step5
- Step8:** Perform a Half-word write operation on the target address to be written to trigger the write operation.
*Example: *((unsigned short *)0x00001232) = 0xABCD.*
- Step9:** Wait for FLASH_CR.BUSY to become 0, and the write operation is completed.
- Step10:** If you need to write Half-word to other addresses that have removed the erase protection, repeat Step8 – Step9.

Word write operation steps are as follows:

- Step1:** Configure FLASH erasing parameters, see the erasing timing chapter for details.
- Step2:** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step3:** Configure FLASH_CR.OP to 1, and set the Flash operation mode to write.
- Step4:** Check whether FLASH_CR.OP is 1, if not, jump to Step2.
- Step5:** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step6:** Set the corresponding bit of FLASH_SLOCK to 1 to remove the erase protection.
- Step7:** Check whether the corresponding bit of FLASH_SLOCK is 1, if not, jump to Step5
- Step8:** Perform a Word write operation on the target address to be written to trigger the write operation.
*Example: *((unsigned long *)0x00001234) = 0x55667788.*
- Step9:** Wait for FLASH_CR.BUSY to become 0, and the write operation is completed.
- Step10:** If you need to write Word to other addresses that have removed the erase protection, repeat Step8 – Step9.

7.3.4 Read operation

Support to read 3 data lengths: Byte (8bits), Half-word (16bits), Word (32bits), the read data is little-endian mode, that is, the low byte of the data stored in the low address. the data in the FLASH can be read at any time.

Byte read operation example: temp = * ((unsigned char *) 0x00001231)
Example of half-word read operation temp = * ((unsigned int *) 0x00001232)
Word read operation example temp = * ((unsigned long *) 0x00001234)

7.4 Erase Timing

FLASH memory has strict time requirements for the control signals of the erasing and programming operations, and failure of the timing of the control signals will cause the erasing and programming operations to fail. When powering on, the erasing parameters when HCLK is 4MHz are loaded by default; if the HCLK frequency is not 4MHz when erasing Flash, the user program should load the erasing parameters corresponding to the HCLK frequency. When operating on FLASH, the frequency range of HCLK is required to be 1MHz ~ 32MHz.

The registers related to the erasing timing parameters are: FLASH_TNVS, FLASH_TPGS, FLASH_TPROG, FLASH_TSERASE, FLASH_TMERASE, FLASH_TPRCV, FLASH_TSRCV, FLASH_TMRCV. If the HCLK is increased from the default 4MHz to 8MHz, the value of the above FLASH_Tx register should be set to twice the default value, that is, keep the current result of $T_{sysclk} * FLASH_Tx$ equal to the default value.

The following table lists the corresponding FLASH programming timing parameters at different frequencies:

Table 7-1 FLASH erasing time parameters at different frequencies

	4M	8M	16M	24M	32M
FLASH_TNVS	0x20	0x40	0x80	0xC0	0x100
FLASH_TPGS	0x17	0x2E	0x5C	0x8A	0xB8
FLASH_TPROG	0x1B	0x36	0x6C	0xA2	0xD8
FLASH_TSERAS	0x4650	0x8CA0	0x11940	0x1A5E0	0x23280
FLASH_TMERASE	0x222E0	0x445C0	0x88B80	0xCD140	0x111700
FLASH_TPRCV	0x18	0x30	0x60	0x90	0xC0
FLASH_TSRCV	0xF0	0x1E0	0x3C0	0x5A0	0x780
FLASH_TMRCV	0x3E8	0x7D0	0xFA0	0x1770	0x1F40

The operation steps to configure the erasing and writing parameters when the system frequency is 8MHz are as follows:

Step1: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

- Step2:** Write 0x40 to the FLASH_TNVS register, if the read value of this register is not 0x40, then jump to the previous step.
- Step3:** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step4:** Write 0x2E to the FLASH_TPGS register, if the read value of this register is not 0x2E, then jump to the previous step.
- Step5:** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step6:** Write 0x36 to the FLASH_TPROG register, if the read value of this register is not 0x36, then jump to the previous step.
- Step7:** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step8:** Write 0x8CA0 to the FLASH_TSERASE register, if the read value of this register is not 0x8CA0, then jump to the previous step.
- Step9:** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step10:** Write 0x445C0 to the FLASH_TMERASE register, if the read value of this register is not 0x445C0, then jump to the previous step.
- Step11:** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step12:** Write 0x30 to the FLASH_TPRCV register, if the read value of this register is not 0x30, then jump to the previous step.
- Step13:** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step14:** Write 0x1E0 to the FLASH_TSRCV register, if the read value of this register is not 0x1E0, then jump to the previous step.
- Step15:** Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.
- Step16:** Write 0x7D0 to the FLASH_TMRCV register, if the read value of this register is not 0x7D0, then jump to the previous step.

7.5 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, a wait cycle must be inserted for CPU fetching, that is, set FLASH_CR.WAIT to 1. When the wait cycle is inserted, the CPU only reads the instruction code in the FLASH memory every two cycles.

7.6 Erase and write protection

7.6.1 Erase and write protection bit

The entire 32kByte FLASH memory is divided into 64 Sectors, and every 4 Sectors share an erase and write protection bit. When the Sector is protected, the erasing and writing operations on the Sector are invalid and an alarm flag and an interrupt signal are generated. When any Sector in the FLASH memory is protected, the Chip erase and write of the FLASH is invalid, and an alarm flag and an interrupt signal are generated.

7.6.2 PC address erase and write protection

When the CPU runs the program in FLASH, if the current PC pointer falls within the address range of the Sector to be erased, the erase operation is invalid and an alarm flag and an interrupt signal are generated.

7.7 Register write protection

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

Registers that require a write sequence to be changed are as follows:

FLASH_TNVS, FLASH_TPGS, FLASH_TPROG, FLASH_TSERASE, FLASH_TMERASE, FLASH_TPRCV, FLASH_TSRCV, FLASH_TMRCV, FLASH_CR, FLASH_SLOCK.

Registers that 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 value, if not, jump to Step1.

Step5: Perform other operations.

Note:

- No write operation can be inserted between the two steps of writing 0x5a5a and writing 0xa5a5, and it cannot be interrupted by interruption, otherwise the Bypass sequence will be invalid, and the 0x5a5a-0xa5a5 sequence needs to be rewritten.

7.8 Register

Base address: 0x4002 0000

Register	Offset address	Description
FLASH_TNVS	0x00	Tnvs time parameter
FLASH_TPGS	0x04	Tpgs time parameter
FLASH_TPROG	0x08	Tprog time parameter
FLASH_TSERASE	0x0C	Tserase time parameter
FLASH_TMERASE	0x10	Tmerase time parameter
FLASH_TPRCV	0x14	Tprcv time parameter
FLASH_TSRCV	0x18	Tsrcv time parameters
FLASH_TMRCV	0x1C	Tmrcv time parameter
FLASH_CR	0x20	control register
FLASH_IFR	0x24	Interrupt Flag Register
FLASH_ICLR	0x28	Interrupt Flag Clear Register
FLASH_BYPASS	0x2C	0x5a5a-0xa5a5 Bypass sequence register
FLASH_SLOCK	0x30	Sector Erase and Write Protection Register

7.8.1 TNVS parameter register (FLASH_TNVS)

Offset address: 0x00

Reset value: 0x0000 0020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TNVS							
R								RW							

Bit	Marking	Functional description
31:9	Reserved	
8:0	TNVS	Calculation formula: $TNVS = 8 \times HCLK$, the unit of HCLK is MHz. For details on how to modify the value of this register, see 7.7. 4MHz example: $TNVS = 8 \times 4 = 32$.

7.8.2 TPGS parameter register (FLASH_TPGS)

Offset address: 0x04

Reset value: 0x0000 0017

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TPGS							
R								RW							

Bit	Marking	Functional description
31:8	RESERVED	
7:0	TPGS	Calculation formula: $TPGS = 5.75 \times HCLK$, the unit of HCLK is MHz. For details on how to modify the value of this register, see 7.7. 4MHz example: $TPGS = 5.75 \times 4 = 23$.

7.8.3 TPROG parameter register (FLASH_TPROG)

Offset address: 0x08

Reset value: 0x0000 001B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TPROG							
R								RW							

Bit	Marking	Functional description
31:8	Reserved	
7:0	TPROG	Calculation formula: $TPROG = 6.75 \times HCLK$, the unit of HCLK is MHz. For details on how to modify the value of this register, see 7.7. 4MHz example: $TPROG = 6.75 \times 4 = 27$.

7.8.4 TSERASE register (FLASH_TSERASE)

Offset address: 0x0C

Reset value: 0x0000 4650

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														TSERASE	
														RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSERASE															
RW															

Bit	Marking	Functional description
31:18	Reserved	
17:0	TSERASE	Calculation formula: $TSERASE = 4500 * HCLK$, the unit of HCLK is MHz. For details on how to modify the value of this register, see 7.7. 4MHz example: $TSERASE = 4500 * 4 = 18000$.

7.8.5 TMERASE parameter register (FLASH_TMERASE)

Offset address: 0x10

Reset value: 0x000222E0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											TMERASE				
R											RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMERASE															
RW															

Bit	Marking	Functional description
31:21	Reserved	
20:0	TMERASE	Calculation formula: $TMERASE = 35000 * HCLK$, the unit of HCLK is MHz. For details on how to modify the value of this register, see 7.7. 4MHz example: $TMERASE = 35000 * 4 = 140000$.

7.8.6 TPRCV parameter register (FLASH_TPRCV)

Offset address: 0x14

Reset value: 0x0000 0018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				TPRCV											
R				RW											

Bit	Marking	Functional description
31:12	Reserved	
11:0	TPRCV	Calculation formula: $TPRCV = 6 * HCLK$, the unit of HCLK is MHz. For details on how to modify the value of this register, see 7.7. 4MHz example: $TPRCV = 6 * 4 = 24$.

7.8.7 TSRCV parameter register (FLASH_TSRCV)

Offset address: 0x18

Reset value: 0x0000 00F0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				TSRCV											
R				RW											

Bit	Marking	Functional description
31:12	Reserved	
11:0	TSRCV	Calculation formula: $TSRCV = 60 * HCLK$, the unit of HCLK is MHz. For details on how to modify the value of this register, see 7.7. 4MHz example: $TSRCV = 60 * 4 = 240$.

7.8.8 TMRCV parameter register (FLASH_TMRCV)

Offset address: 0x1C

Reset value: 0x0000 03E8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				TMRCV											
R				RW											

Bit	Marking	Functional description
31:13	RESERVED	
12:0	TMRCV	Calculation formula: $TMRCV = 250 \times HCLK$, the unit of HCLK is MHz. For details on how to modify the value of this register, see 7.7. 4MHz example: $TMRCV = 250 \times 4 = 1000$.

7.8.9 CR register (FLASH_CR)

Offset address: 0x20

Reset value: 0x0000 0200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									IE	BUSY	Res	WAIT	OP		
R									RW	RO	RO	RW	RW		

Bit	Marking	Functional description
31:7	Reserved	
6:5	IE	IE[6]: FLASH erase and write protected address interrupt enable; 0: Disable; 1: Enable IE[5]: FLASH erase PC value interrupt enable; 0: Disable; 1: Enable
4	BUSY	Idle / busy flag; 0: idle state; 1: busy state;
3	Reserved	
2	WAIT	Read FLASH waiting period; 0: 0 waiting period; 1: 1 waiting period
1:0	OP	FLASH operation; 00: read; 01: program; 10: sector erase; 11: chip erase

For details on how to modify the value of this register, see 7.7.

7.8.10 IFR register (FLASH_IFR)

Offset address: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														IF1	IF0
R														RO	RO

Bit	Marking	Description
31:2	Reserved	
1	IF1	Erase and write protection alarm interrupt flag bit
0	IF0	Erase and write PC address alarm interrupt flag bit

7.8.11 ICLR register (FLASH_ICLR)

Offset address: 0x28

Reset value: 0x0000 000F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														ICLR1	ICLR0
R														RW0	RW0

Bit	Marking	Functional description
31:4	Reserved	
3:2	Reserved	Write invalid, read as 0x3
1	ICLR1	Clear protection alarm interrupt flag; write 0 to clear; write 1 to be invalid;
0	ICLR0	Clear the PC address alarm interrupt flag; write 0 to clear; write 1 to be invalid;

7.8.12 BYPASS register (FLASH_BYPASS)

Offset address: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BYSEQ															
WO															

Bit	Marking	Description
31:16	Reserved	
15:0	BYSEQ	Before modifying the registers of this module, the 0x5a5a-0xa5a5 sequence must be written to the BYSEQ[15:0] register. each write to the Bypass sequence, the register can only be modified once. If you need to modify the register again, you must enter the 0x5a5a-0xa5a5 sequence again. See 7.7 for details.

7.8.13 SLOCK register (FLASH_SLOCK)

Offset address: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLOCK															
RW															

Bit	Marking	Description
31:16	Reserved	
15:0	SLOCK	Sector erase and write protection bit; 0: not allowed to erase and write; 1: allowed to erase and write SLOCK[0] corresponds to: Sector0-1-2-3 SLOCK[1] corresponds to: Sector4-5-6-7 SLOCK[2] corresponds to: Sector8-9-10-11 SLOCK[3] corresponds to: Sector12-13-14-15 SLOCK[15] corresponds to: Sector60-61-62-63

8 RAM controller(RAM)

8.1 Overview

This product contains a static SRAM with a capacity of 2K/4K byte, supporting byte (8bits), half-word (16bits), word (32bits) Three read and write operations. Read and write operations can be performed at the HCLK frequency without waiting for cycles. In addition, this controller also supports parity check, which can perform parity check on the SRAM data of each byte, and generate a parity check error interrupt.

8.2 Functional description

Read and write bit width

This controller supports Byte (8bits), Half-word (16bits), Word (32bits) Read and write operations with three bit widths. Byte operation must be aligned by Byte, the target address of Half-word operation must be aligned by Half-word (the lowest bit of the address is 0), and the address of Word operation must be aligned by Word (the lowest two bits of the address are 0). 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.

Parity

This controller supports parity check of SRAM data. After reset, this function is disabled by default. When the parity check function is turned on, when writing data to the SRAM, the parity check is performed on each Byte of data, and the 1bit check value and 8bits data are stored in the SRAM together. When reading data from SRAM, the controller will read 8bits data and 1bit check value, and perform parity check. If the check is wrong, the parity check error flag will be set. When the interrupt is enabled, a Error interrupt.

Note:

- When the parity check is enabled, the SRAM must be initialized before reading the SRAM data, otherwise the parity check alarm flag or interrupt may be triggered by mistake.

8.3 Register

Base address: 0x4002 0400

Register	Offset address	Description
RAM_CR	0x00	control register
RAM_ERRADDR	0x04	Error Address Register
RAM_IFR	0x08	Error Interrupt Flag Register
RAM_ICLR	0x0C	Error Interrupt Flag Clear Register

8.3.1 Control Register (RAM_CR)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													IE	CHKE N	
R													RW	RW	

Bit	Marking	Functional description
31:2	RESERVED	
1	IE	Error alarm interrupt enable signal; 1: enable alarm interrupt, 0: disable alarm interrupt;
0	CHKEN	Parity check enable signal; 1: enable parity check, 0: close parity check;

8.3.2 Parity Error Address Register (RAM_ERRADDR)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				ERRADDR											
R				RO											

Bit	Marking	Functional description
31:12	Reserved	
11:0	ERRADDR	12bits parity error byte address

8.3.3 Error Interrupt Flag Register (RAM_IFR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ERR
R															RO

Bit	Marking	Functional description
31:1	RESERVED	
0	ERR	Parity error flag

8.3.4 Error Interrupt Flag Clear Register (RAM_ICLR)

Offset address: 0x0C

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ERR CLR
R															RW0

Bit	Marking	Functional description
31:1	Reserved	
0	ERRCLR	Error interrupt flag clear bit; write 1: invalid, write 0: clear

9 Basic Timer (TIM0/1/2)

9.1 Introduction to Basic Timers

The basic timer contains three timers Timer0/1/2. Timer0/1/2 have exactly the same function. Timer0/1/2 is a synchronous timer/counter, which can be used as a 16-bit timer/counter with automatic reloading function, or as a 32-bit timer/counter without reloading function. Timer0/1/2 can count external pulses or implement system timing.

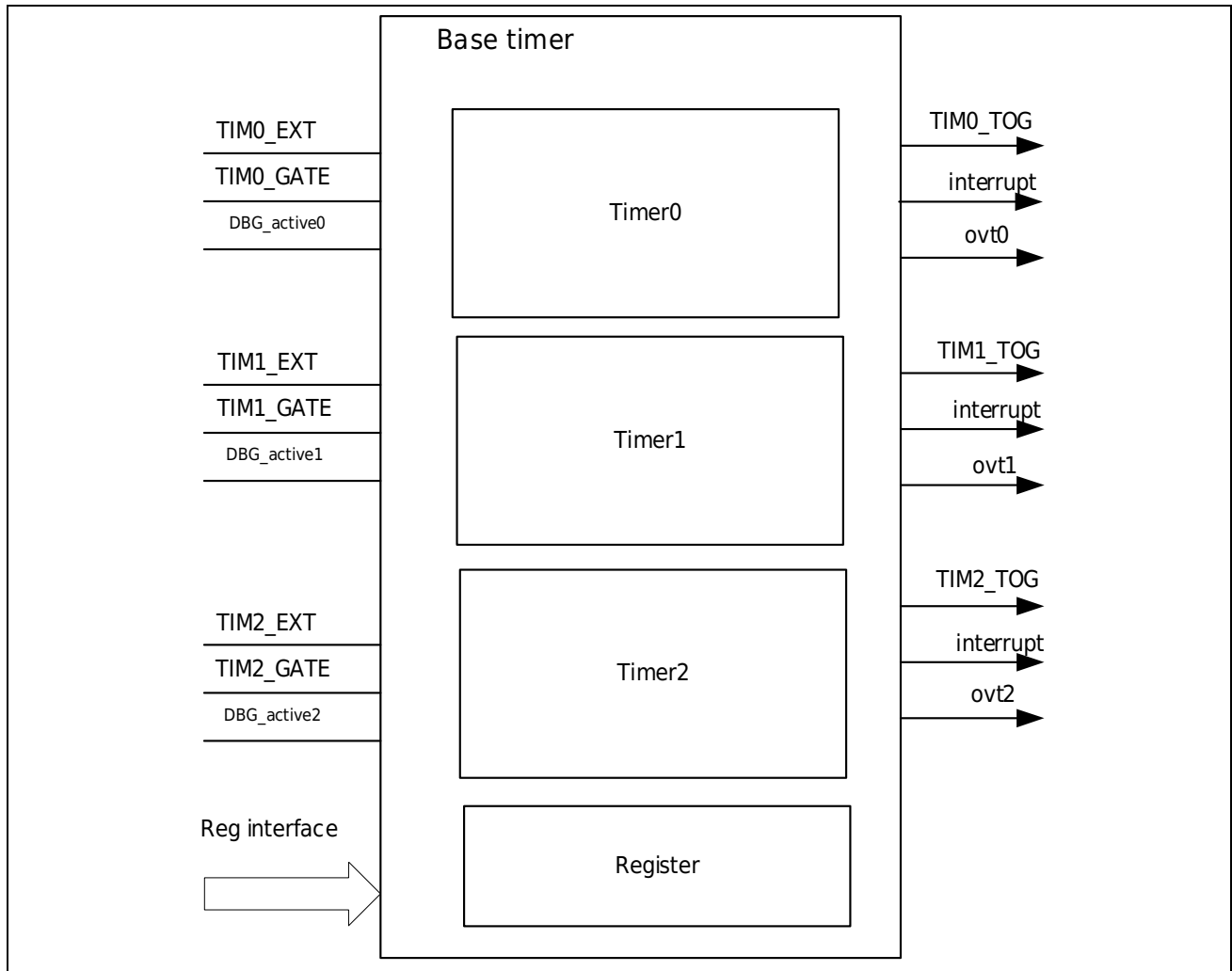


Figure 9-1 Base Timer Block Diagram

9.2 Base Timer Function Description

Each timer / counter of Timer0/1/2 has an independent control start signal, as well as an external input clock and gate control signal.

Timer0/1/2 uses EXT and GATE to perform the counting function, EXT is used for the external input clock signal of the counter, and GATE is used for the active level counting enable signal. When the gating function is enabled, the counter will count only when the external input GATE level is valid, otherwise the counter is in a hold state. Gate enable is controlled using CR.gate. The default gating

function is off. Gate level selection is controlled by CR.GATE_P. The default high level is the effective level of gate control; after setting to 1, the low level of gate control is the active level.

TIM0/1/2 uses PCLK, GATE for timing function, PCLK is used for internal input clock signal of timer, GATE can be used for active level timing enable signal. When the gating function is enabled, the timer will count only when the external input GATE level is valid, otherwise the timer is in the stop state of the timing counter. Gate enable is controlled using CR.Gate. The default gating function is off. Gate level selection is controlled by CR.Gate_P. The default high level is the effective level of gate control; after setting to 1, the active level of gate control is low level. Timing function can be configured with pre-divider. CR.PRS controls the divider ratio.

PRS [2:0]	000	001	010	011	100	101	110	111
Frequency division ratio	1	2	4	8	16	32	64	256

The timer / counter of TIM0/1/2 supports two working modes, and the working mode is selected by setting MD in the timer control register (CR). Mode 1 is a 32 -bit free counting mode. Mode 2 is a 16 -bit reload mode.

In 32-bit free counting mode, an interrupt is generated after counting to the maximum 0xFFFFFFFF overflow, and the timer/counter becomes 0X00000000; and then continues to count; in 16-bit heavy load mode, an interrupt is generated after counting to the maximum 0xFFFF, and the value of the timer/counter is loaded as ARR value, and then continue to count up.

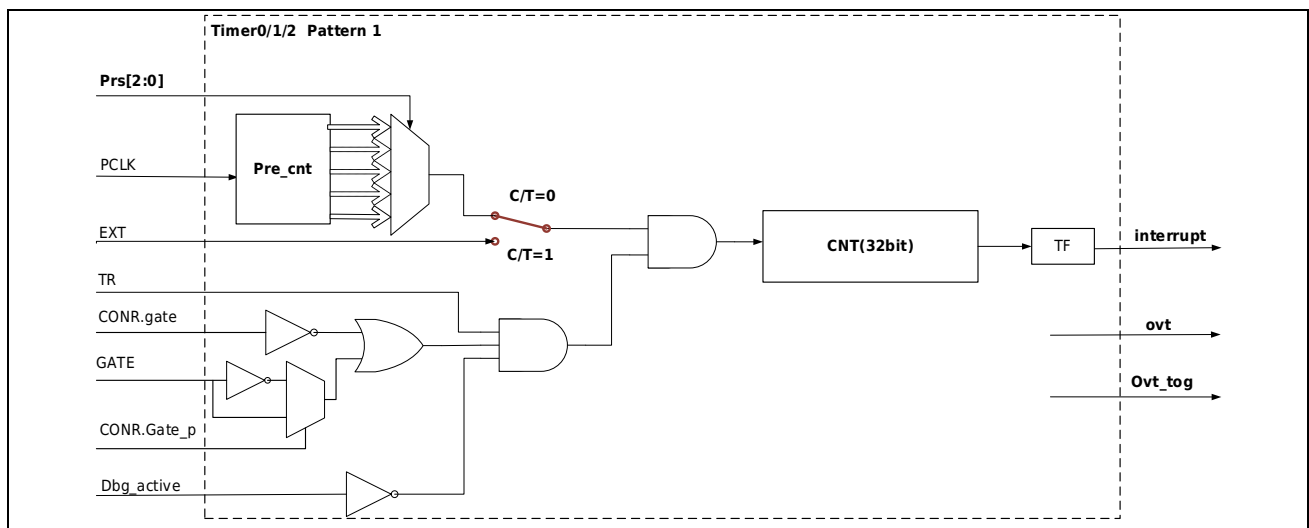


Figure 9-2 Timer Mode 1 Block Diagram

In the mode 2 heavy load mode, the software processing speed needs to be considered when the timing time is set small, otherwise the interrupt will be too late to process and the interrupt will be lost.

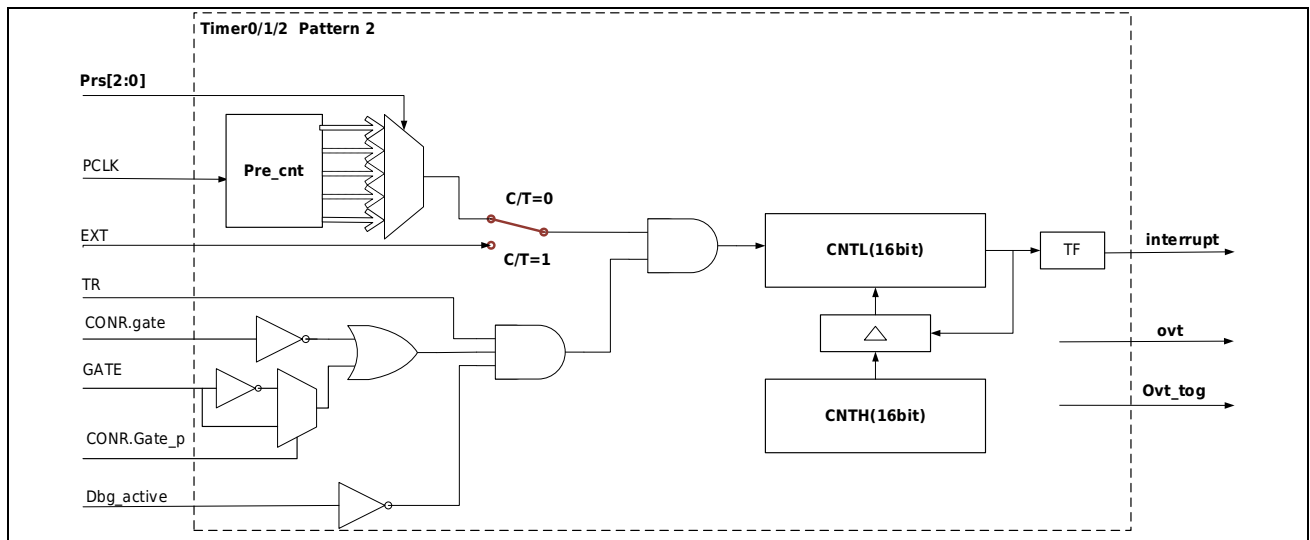


Figure 9-3 Timer Mode 2 Block Diagram

When the corresponding timer TR is set to 1, the timer starts to run. Mode 1 is a 32-bit timer / counter. After startup, it starts counting from the initial value T set by the register, and generates an overflow interrupt after counting to the maximum. Then continue counting from 0. Mode 2 is a 16-bit reload timer / counter. After startup, it starts counting up from the initial value of the register CNT. After counting to the maximum value of 0xFFFF, an interrupt is generated and the value of the reload register ARR is transferred to the counter CNT, and continues to count up.

9.2.1 Counting function

Counting functions are used to determine the number of times an event occurs. In the count function, the counter increments every falling edge of the corresponding input clock. The input signal is sampled by the internal PCLK, so the external input clock frequency cannot exceed the system Pclk clock. Counting to the maximum value will overflow and generate an interrupt. The interrupt flag needs to be cleared by software.

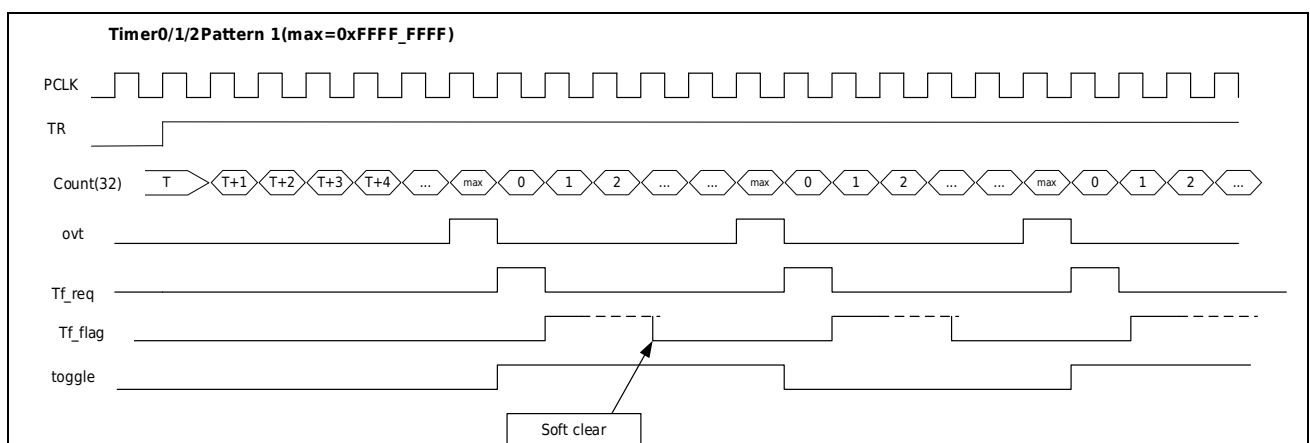


Figure 9-4 Mode 1 Timing Diagram

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.

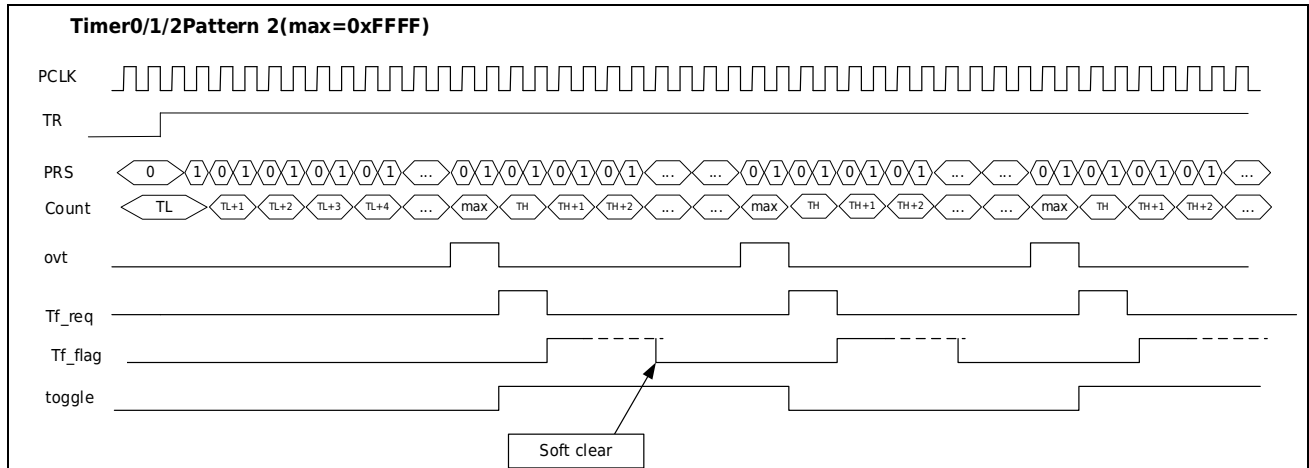


Figure 9-5 Mode 2 Timing Diagram (prescaler set to 2)

9.2.2 Buzzer function

The function of driving the Buzzer can be realized through the flipping output function of the timer. CR.TOG_EN is 1, TOG, TOGN output reverse. Setting CR.TOG_EN to 0 can set port TOG, TOGN output to 0 at the same time. When the counting clock is 4M, the Timer reload mode configuration of the Buzzer outputting different frequencies is 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

9.3 Base Timer Interconnection

9.3.1 GATE interconnection

The GATE input can be directly input from the port, or the RX signal of the UART; it can also be configured as the input of the VC as the GATE signal. The GATE of Timer0/1/2 can be configured.

Through the internal interconnection configuration, the automatic identification of the UART baud rate can be realized, the pulse width of the VC comparison output can be measured, and the external control count can be realized.

The configuration selection RX input is controlled by the GPIO_CTRL register, and the VC control is controlled by the VC control register. Port selection, UART input selection and VC input selection can only be selected as a gating input to be valid.

9.3.2 Toggle Output Interconnects

The tog0 output of TIM0 is connected to the internal module UART0 to control the baud rate of UART0; the tog1 output of TIM1 is connected to the internal module UART1 to control the baud rate of UART1; the toggle output of TIM0/1/2 is also output to the port, which can drive Buzzer realizes the control of the buzzer.

9.4 Base Timer register description

Table 9-1 Base TimerRegister List

x=0,1,2;

Base Timer base address 0X40000C00

Timer	Offset address	Description
TIM0	0x00	TIM0 offset address
TIM1	0x20	TIM1 offset address
TIM2	0x40	TIM2 offset address

Register	Offset address	Description
TIMx_ARR	0X000	TIM0/1/2 reload register
TIMx_CNT	0X004	TIM0/1/2 16 -bit mode count register
TIMx_CNT32	0X008	TIM0/1/2 32 -bit mode count register
TIMx_CR	0X00C	TIM0/1/2 Control Register
TIMx_IFR	0X010	TIM0/1/2 interrupt flag
TIMx_ICLR	0X014	TIM0/1/2 Interrupt Clear Register

9.4.1 16 -bit Mode Reload Register (TIMx_ARR)

Offset address: 0x000

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
R/W															

Bit	Symbol	Description
31:16	Reserved	Reserved bit, read as 0
15:0	ARR	Timer Mode 2 Reload Register

9.4.2 16-bit Mode Count Register (TIMx_CNT)

Offset address: 0x004

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
R/W															

Bit	Symbol	Description
31:16	Reserved	Reserved bit, read as 0
15:0	CNT	Timer Mode 2 Count Value Register

9.4.3 32-bit mode count register (TIMx_CNT32)

Offset address: 0x008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT32[31:16]															
R/W															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT32[15:0]															
R/W															

Bit	Symbol	Description
31:0	CNT32	Timer Mode 1 Count Value Register

9.4.4 Control Register (TIMx_CR)

Offset address: 0x00C

Reset value: 0x0000 0008

31:11	10	9	8	7	6:4	3	2	1	0
Reserved	IE	GATE_P	GATE	Reserved	PRS	TOG_EN	CT	MD	TR
	RW	R/W	R/W		R/W	R/W	R/W	R/W	R/W

Bit	Symbol	Description
31:11	Reserved	Reserved bit, read as 0
10	IE	Interrupt enable control, enable interrupt after writing 1
9	GATE_P	Port GATE polarity control, the default high level gate is valid, set to 1 and low level is valid
8	GATE	Timer Gating 0: No gate control, the timer works when TR=1; 1: Only work when the port GATE is valid and TR=1;
7	Reserved	Reserved bit, read as 0
6:4	PRS	TIM prescaler selection. 000:1; 001:2; 010:4; 011:8; 100:16; 101:32; 110:64; 111:256;
3	TOG_EN	TOG output enable 0: TOG, TOGN output 0 at the same time 1: TOG, TOGN output signals with opposite phases. Available for buzzers.
2	CT	Counter / timer function selection 0: Timer function, the timer is counted by PCLK. 1: Counter function, the counter counts by the falling edge of the external input. The external input is sampled by PCLK, and the external input clock frequency is lower than 1/2 sampling clock.
1	MD	Timer working mode 0: Mode 1 32 -bit counter / timer 1: Mode 2 auto-reload 16 -bit counter / timer
0	TR	Timer run control 0: timer stopped 1: Timer running

9.4.5 Interrupt Flag Register (TIMx_IFR)

Offset address: 0x010

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved								TF
								RO

Bit	Symbol	Description
31:1	REV	Reserved bit, read as 0
0	TF	Interrupt flag, set by hardware. write clear register clear

9.4.6 Interrupt Flag Clear Register (TIMx_ICLR)

Offset address: 0x014

Reset value: 0x0000 0001

31:8 7 6 5 4 3 2 1 0

Reserved										TFC
										W0

Bit	Symbol	Description
31:1	Reserved	Reserved bit, read as 0
0	TFC	Interrupt flag is cleared, write 0 to clear, write 1 to be invalid

10 Low Power Timer (LPTIM)

10.1 Introduction to LPTimer

LPTimer is an asynchronous 16-bit timer / counter, which can still time / count through internal low-speed RC or external low-speed crystal oscillator after the system clock is turned off. Wake up the system in low-power mode through interrupts.

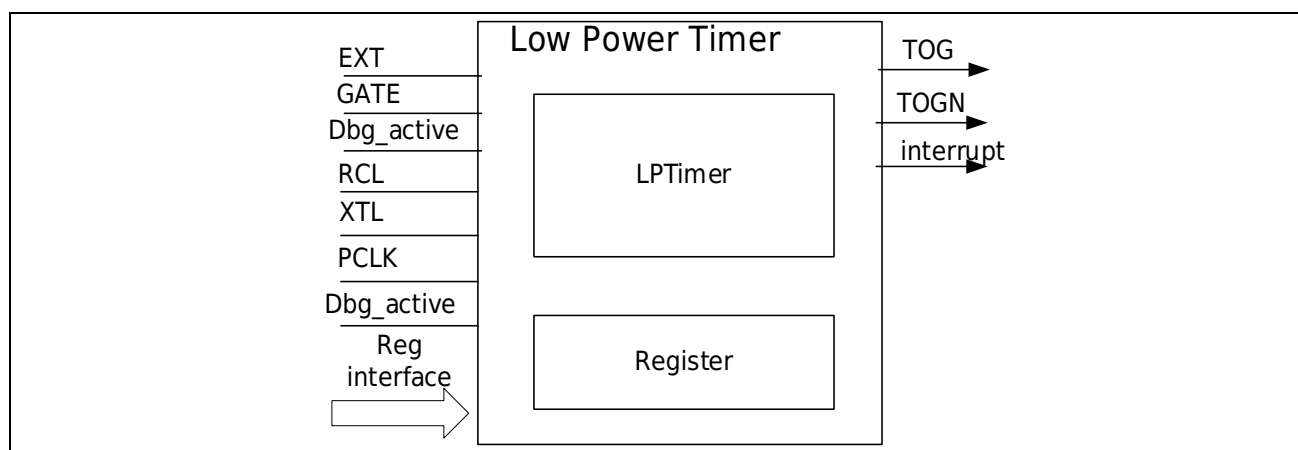


Figure 10-1 LPTimer block diagram

10.2 LPTimer Function Description

LPTimer supports 2 working modes, each timer / counter has an independent control start signal, as well as external input clock and gating signal.

LPTimer uses EXT and GATE to perform the counting function, EXT is used for the external input clock signal of the counter, and Gate is used for the active level counting enable signal.

LPTimer supports two working modes, and the working mode is selected by setting MD in the timer control register (CR). Mode 1 is a 16-bit free counting mode. Mode 2 is a 16-bit reload mode.

When LPTimer starts, it will automatically load the value of the reload register ARR into the counter.

LPTimer can choose three clocks as the timer clock, which are selected by the control register CR.TCK_SEL. PCLK is selected by default. The clock selection is as shown in the table:

TCK_SEL	00	01	10	11
Timer clock	PCLK	PCLK	XTL	RCL
Read timer count value	read synchronized	no sync	read synchronized	read synchronized

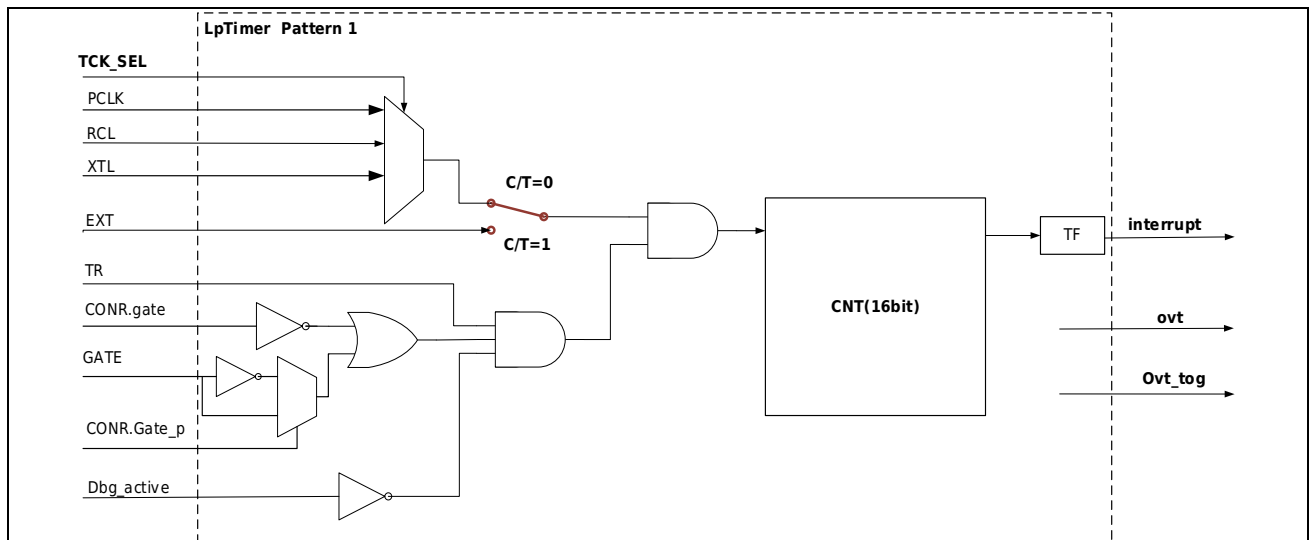


Figure 10-2 LPTimer Mode 1

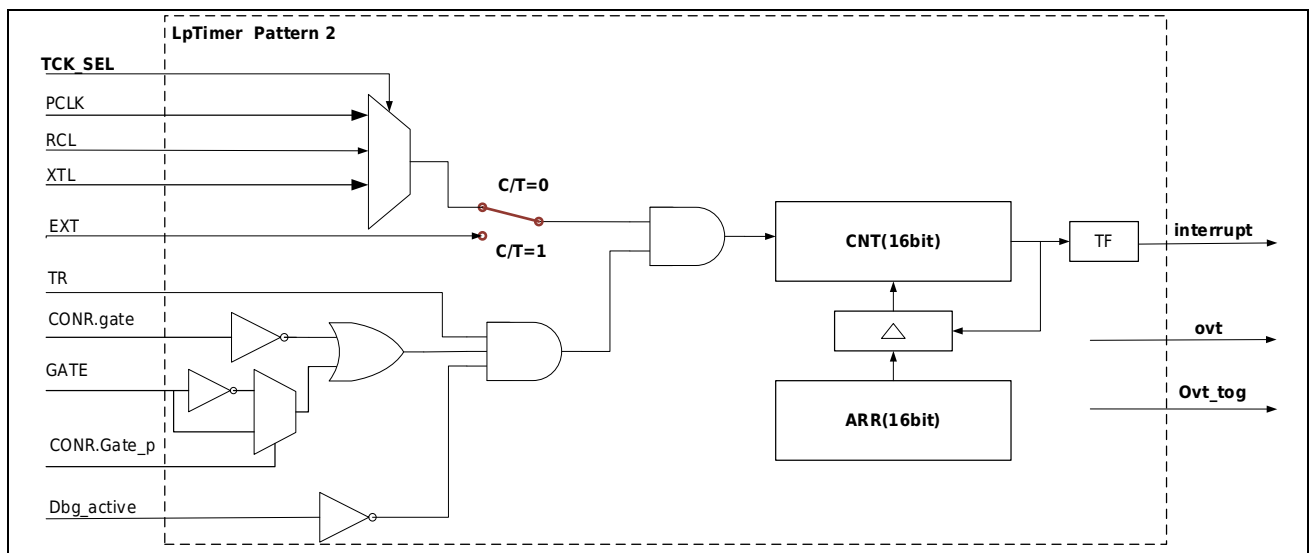


Figure 10-3 LPTimer Mode 2

When the corresponding timer TR is set to 1, the timer starts to run. The counter starts counting from the set value, and generates an overflow interrupt after counting to a maximum of 0xFFFF. After Mode 1 generates an interrupt, the counter is cleared and continues to count up. After Mode 2 generates an interrupt, the value of the reload register ARR is transferred to the counter and counts up. The initial value of the counter CNT is automatically loaded by ARR when the timer is started.

Since LPTimer is an asynchronous timer, the timer interrupt is synchronous from the Timer clock domain to the pclk pre-set. When the timer value in reload mode is set greater than 0xFFFFb, the interrupt will be lost. It is recommended to use the interrupt function if the value of the reload register is less than 0xFFFFC. Not using interrupts does not have this restriction.

10.2.1 Counting function

Counting functions are used to determine the number of times an event occurs. In the count function, the counter increments every falling edge of the corresponding input clock. The input signal is sampled by the internal count clock, so the external input clock frequency cannot exceed the system count clock. Counting to the maximum value will overflow and generate an interrupt.

10.2.2 Timing function

The timing function is used to generate interval timing. In the timing function, the timer accumulates once per clock, and when the count reaches the maximum value, it will overflow and generate an interrupt.

10.3 LPTimer Interconnect

10.3.1 GATE interconnection

The GATE input can be directly input from the port, or the RX signal of the UART; it can also be configured as the input of the VC as the GATE signal. The GATE of LPTIM can be configured as follows.

Through the internal interconnection configuration, the automatic identification of the UART baud rate can be realized, the pulse width of the VC comparison output can be measured, and the external control count can be realized.

The UART selection control register is GPIO_CTRL4 in the port control register, and the VC output control register is in the VC control module.

10.3.2 EXT Interconnection

The EXT input can be directly input from the port, or it can be configured as the input of the VC as the EXT signal. The EXT of LPTIM can be configured as follows.

With the interconnect configuration, VC pulse counts can be measured. The VC output control register is in the VC control block.

10.3.3 Toggle Output Interconnects

LPTimer is output to the port, which can drive the Buzzer to realize the control of the buzzer.

10.4 LPTimer register description

Table 10-1 LPTimer register list

Base address 0X40000C00

Register	Offset address	Description
CNT	0X060	LPTimer count value read-only register
ARR	0X064	LPTimer reload register
CR	0X06C	LPTimer Control Register
IFR	0X070	LPTimer interrupt flag
ICLR	0X074	LPTimer Interrupt Clear Register

10.4.1 Counter Count Value Register (LPTIM_CNT)

Offset address: 0x060

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
RO															

Bit	Symbol	Description
31:16	Reserved	Reserved bit, read as 0
15:0	CNT	Low-power timer count value read-only register. When starting the timer, the initial value of CNT is automatically loaded by ARR.

10.4.2 Reload Register (LPTIM_ARR)

Offset address: 0x064

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
R/W															

Bit	Symbol	Description
31:16	Reserved	Reserved bit, read as 0
15:0	ARR	Low Power Timer Reload Register CR.WT_FLAG needs to be read before writing ARR, and data can only be written when WT_FLAG is 1. WT2_FLAG will go low after writing the ARR register.

10.4.3 Control Register (LPTIM_CR)

Offset address: 0x06C

Reset value: 0x0000 0008

31:11	10	9	8	7	6	4:5	3	2	1	0
Reserved	IE	GATE_P	GATE	WT_FLAG	Reserved	TCK_SEL	TOG_EN	CT	MD	TR
	RW	R/W	R/W	RO		R/W	R/W	R/W	R/W	R/W

Bit	Symbol	Description
31:11	Reserved	Reserved bit, read as 0
10	IE	Interrupt enable control, enable interrupt after writing 1
9	GATE_P	GATE polarity control, the default high level gate is valid, after setting to 1, the low level is valid
8	GATE	Timer Gating 0: No gate control, the timer works when TR=1; 1: works when the gate is 1 and TR=1;
7	WT_FLAG	WT, write synchronization flag 0: Synchronizing, writing ARR is invalid at this time 1: Synchronization is complete, ARR can be changed at this time
6	Reserved	Reserved bit, read as 0
5:4	TCK_SEL	LPTimer clock selection 00:PCLK; 10:XTL; 11:RCL
3	TOG_EN	TOG output enable 0: TOG, TOGN output 0 at the same time 1: TOG, TOGN output signals with opposite phases. Available for buzzers.
2	CT	Counter / timer function selection 0: Timer function, the timer uses the clock selected by TCK_SEL to count. 1: Counter function, the counter uses the falling edge of the external input to count. The sampling clock uses the clock selected by TCK_SEL, and the external input clock is lower than 1/2 sampling clock.
1	MD	Timer working mode 0: mode 1 no reload mode 16-bit counter / timer 1: Mode 2 auto-reload 16-bit counter / timer
0	TR	Timer run control bit 0: timer stopped 1: Timer running

10.4.4 Interrupt Flag Register (LPTIM_IFR)

Offset address: 0x070

Reset value: 0x0000 0000

31:8 7 6 5 4 3 2 1 0

Reserved									TF
									RO

Bit	Symbol	Description
31:1	Reserved	Reserved bit, read as 0
0	TF	Interrupt flag, set by hardware. write clear register clear

10.4.5 Interrupt Flag Clear Register (LPTIM_ICLR)

Offset address: 0x074

Reset value: 0x0000 0001

31:8 7 6 5 4 3 2 1 0

Reserved									TFC
									W0

Bit	Symbol	Description
31:1	Reserved	Reserved bit, read as 0
0	TFC	Interrupt flag is cleared, write 0 to clear, write 1 to be invalid

11 Programmable Count Array (PCA)

11.1 Introduction to PCA

PCA (Programmable Counter Array) supports up to 5 16-bit capture/compare modules. The timer/counter can be used as a common clock count/event counter capture/compare function. Each module of PCA can be independently programmed to provide input capture, output comparison or pulse width modulation. In addition, module 4 has an additional watchdog timer mode.

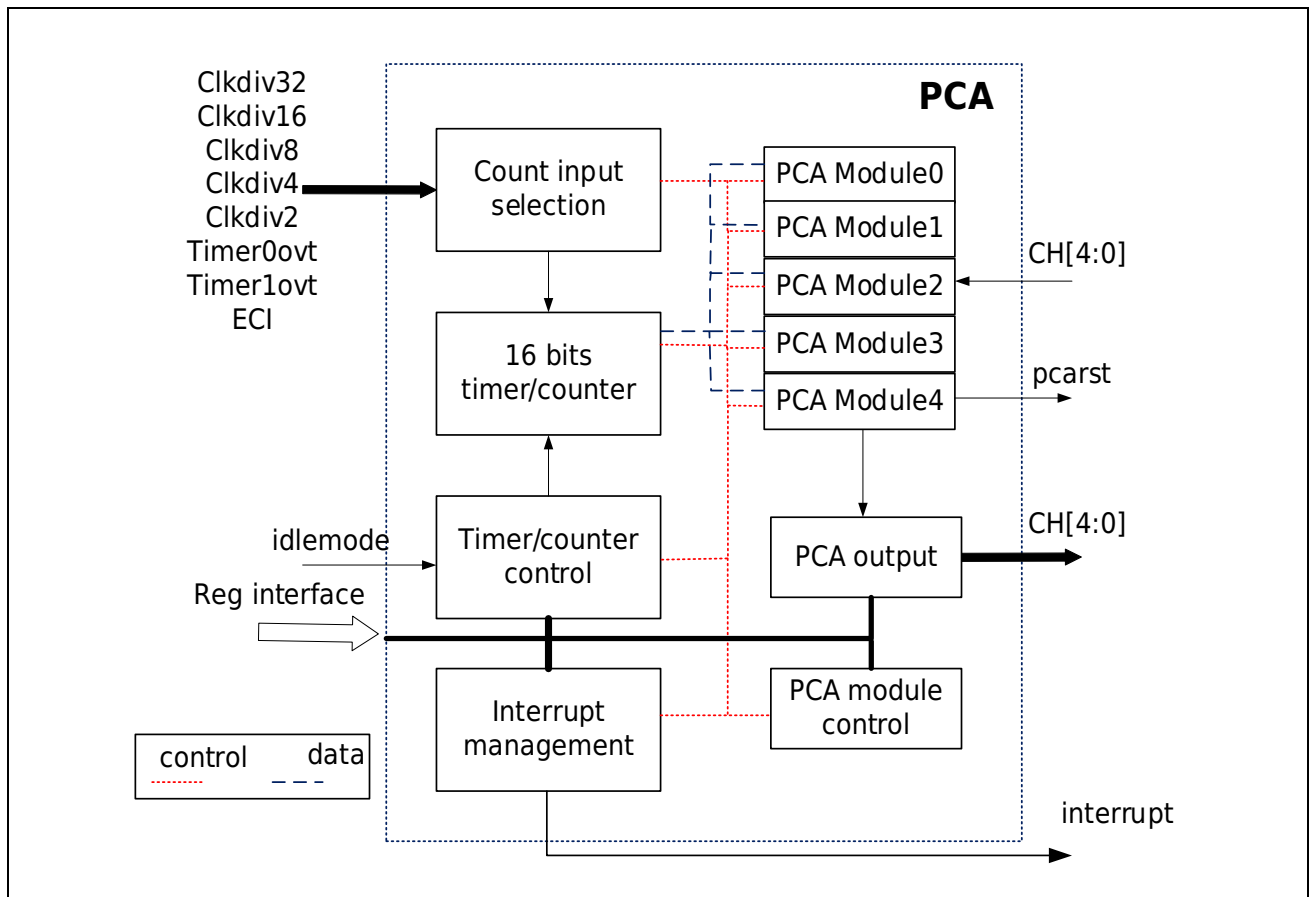


Figure 11-1 Overall block diagram of PCA

11.2 PCA function description

Each module can be configured to work independently, and there are three working modes: edge-triggered capture, output comparison, and 8-bit pulse width modulation. Each module has its own function registers in the system controller, and these registers are used to configure the working mode of the module and exchange data with the module.

Each comparison / capture module is composed of a comparison / capture register group (CCAPx), a 16-bit comparator, and various logic gate controls. The register group is used to store time or times, for external trigger capture conditions, or internal trigger comparison conditions. In PWM mode, the register (CCAPxL) is used to control the duty cycle of the output waveform.

Each module can be independently programmed to operate in any of the following modes:

- 16-bit capture mode.
- Compare mode: 16-bit software timer, 16-bit high-speed output, 16-bit watchdog timer (module 4) or 8-bit pulse width modulation.
- Have not started.

The Compare / Capture Module Mode Registers (CCAPMx) determine the corresponding operating mode. When programming the compare / capture modules, they are based on a common time count. The timer / counter is turned on and off through the CCON.CR bit to control the operation of the PCA timer / counter. On a compare / capture module capture, the software timer, high-speed output, sets the module's compare / capture flag (CCON.CCFx), and generates a PCA interrupt request if the corresponding enable bit is set in the CCAPMx register. The CPU can read and write the CCAPx registers at any time.

11.2.1 PCA Timer / Counter

This group of special function registers of CNT can be used as a 16-bit timer / counter. This is a 16-bit up counter. If the CMOD.CFIE bit is set to "1", when the CNT overflows, the hardware automatically sets the PCA overflow flag (CCON.CF) and generates a PCA interrupt request. CMOD.CPS[2:0] Three bits select eight signals to input to timer/counter.

- The system clock PCLK is divided by 32.
- The system clock PCLK is divided by 16.
- The system clock PCLK is divided by 8.
- The system clock PCLK is divided by 4.
- The system clock PCLK is divided by 2.
- Timer 0 overflow. CNT is incremented each time Timer 0 overflows, thus providing a variable programming frequency input to the PCA.
- Timer 1 overflow. CNT is incremented each time Timer 1 overflows, thus providing a variable programming frequency input to the PCA.

- ECI. The CPU samples the PCA ECI every 4 PCLK clock cycles. When the sampling result changes from high to low each time, CL is automatically increased by 1. Therefore, the highest ECI input frequency cannot be higher than 1/8 of the system clock PCLK to meet Sampling requirements. Set the run controller (CCON.CR) to start the PCA timer / counter. When CMOD.CIDL is set to "1", the PCA timer / counter can continue to run in idle mode. The CPU can read the value of CNT at any time, but when the count is started (CCON.CR=1), in order to prevent counting errors, CNT is prohibited from being written.

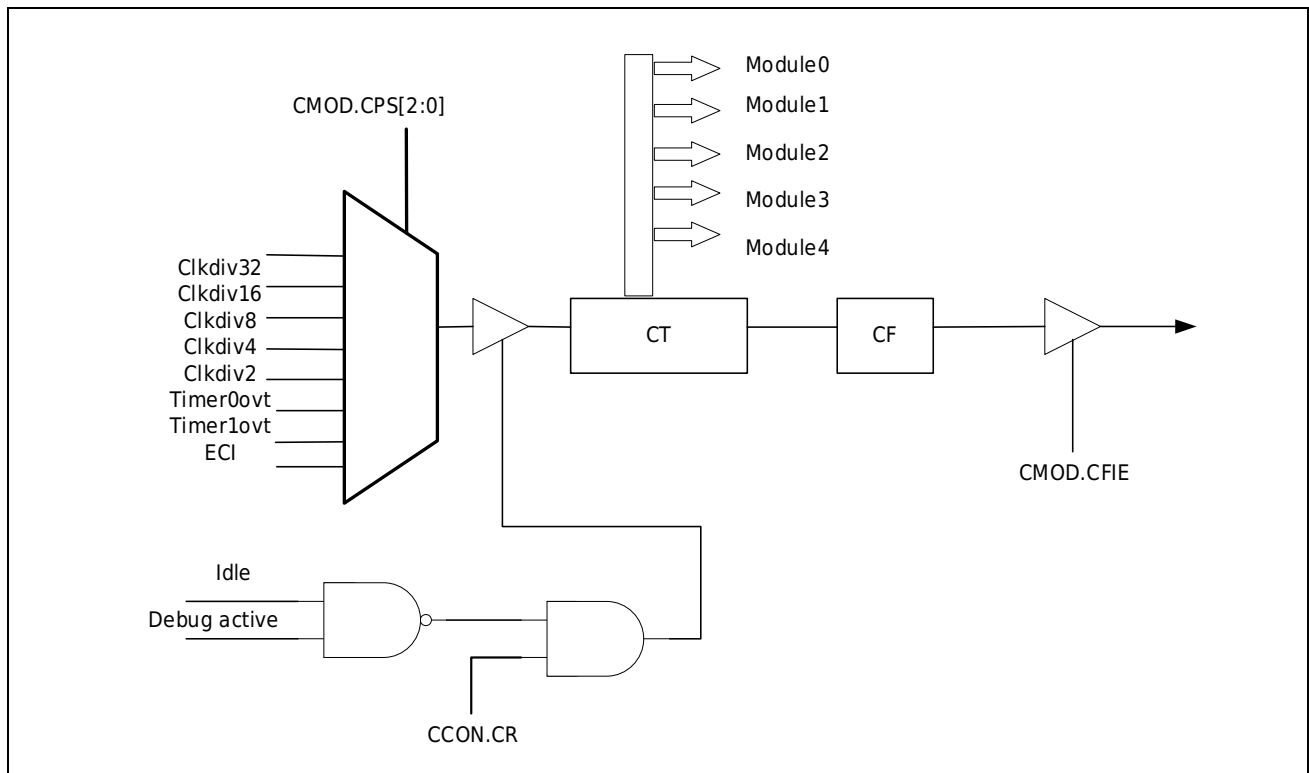


Figure 11-2 PCA Counter Block Diagram

11.2.2 PCA capture function

PCA capture mode provides 5- way PCA to measure pulse period, pulse width, duty cycle and phase difference.

A level transition on the pin causes the PCA to capture the value of the PCA counter / timer and load it into the corresponding module's 16 -bit capture / compare register (CCAPx). The CCAPMx.CAPP and CCAPMx.CAPN bits are used to select the type of level change that triggers the capture: low to high (positive edge), high to low (negative edge), or any change (positive or negative edge) along). When a capture occurs, the Capture / Compare Flag (CCFn) in CCON is set to logic '1' and an interrupt request is generated (if CCF interrupts are enabled). When the CPU turns to the interrupt service routine, the CCFn bit cannot be automatically cleared by hardware, and the user software can write the INTCL register to clear this flag bit. It is necessary to capture the upper and lower edges at the same time. The suggested process: first enable one kind of edge capture, enable another kind of capture in the interrupt service routine after the capture occurs, and turn off the former capture interrupt. Switch in turn.

The resolution is equal to the clock of the timer / counter. 2 clock cycles during the high level or low level to ensure that the input signal can be recognized by the hardware.

The CPU can read or write the CCAPx registers at any time.

Capture settings:

- When capturing on an external rising edge is required, CCAPMx.CAPP = "1" and CCAPMx.CAPN = "0"
- When capturing on an external falling edge is required, CCAPMx.CAPP = "0" and CCAPMx.CAPN = "1"
- When capture is required on external rising and falling edges, CCAPMx.CAPP = "1" and CCAPMx.CAPN = "1"
- When the external rising and falling edges need to be captured at the same time and the captured edge needs to be known, first set CCAPMx.CAPP = "1", after the rising edge capture occurs, switch to the falling edge capture in the interrupt service subroutine and set CCAPMx.CAPN = "1", and clear CCAPMx.CAPP = "0". After the next capture, set it as rising edge capture, and switch the captured edge in turn.

Note:

- Subsequent captures by the same module override existing captured values. In order to keep the captured value, save it in RAM in the interrupt service routine, this operation must be completed before the next event occurs, otherwise the previous captured sampling value will be lost.

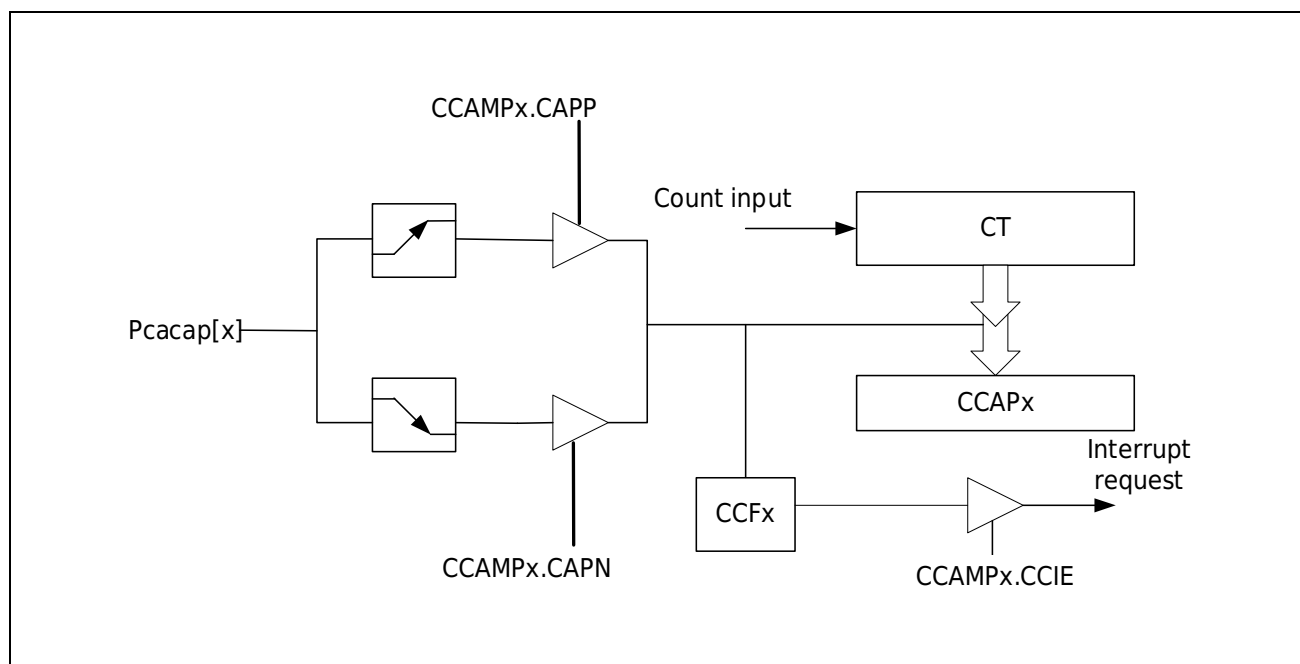


Figure 11-3 PCA Capture Functional Block Diagram

11.2.3 PCA comparison function

The compare function provides five modules as following functions, timer, event counter, pulse width modulation. Four modes use the compare function: 16 -bit software timer mode, high-speed output mode, WDT mode and PWM mode. In the first three functions, the compare / capture module compares the value of the 16- bit PCA timer / counter with the 16- bit value preloaded into the module's CCAPx register. In PWM mode, the PCA module continuously compares the PCA Timer / Counter Low Byte Register (CNT) with an 8 -bit value in the CCAPxL module register. The comparison is done every 4 clock cycles, which matches the clock rate of the fastest PCA timer/counter.

Setting the CCAPMx.ECOM bit selects the compare function for this module.

To use modules in compare mode correctly, follow the general procedure below:

- Select the operating mode of the PCA module
- Selects the input signal for the PCA timer / counter.
- The compare value is loaded into the module's compare / capture register pair.
- Set the PCA timer / counter run control bit.
- An interrupt is generated after a match, and the compare/capture flag of the module is cleared.

11.2.3.1 16 -bit software counting mode

To set a compare / capture module in 16 -bit software timer mode, the CCAPMx.ECOM and CCAPMx.MAT bits need to be set. Once a match occurs between the PCA timer / counter and the compare / capture register (CCAPx), this sets the module's compare / capture flag (CCON.CCFx). This will generate an interrupt request if the corresponding interrupt enable bit (CCAPMx.CCIE) is set. Since the compare/capture flag is not cleared by hardware when interrupt processing is performed, the user must clear the flag in software. In the interrupt service routine, a new 16 -bit compare value can be written to the compare / capture register (CCAPx).

Note: To prevent invalid matches when updating these registers, user software should write CCAPxL first, followed by CCAPxH. Writing to CCAPxL will clear the ECOMx bit which disables the compare function, while writing to CCAPxH will simultaneously set the ECOMx bit and re-enable the compare function. That is, when writing a 16-bit value to the capture/compare register of PCA0, the low byte should be written first.

11.2.3.2 High speed output mode

In the high-speed output mode, whenever the value in the PCA counter matches the 16 -bit capture / compare register (CCAPx) of the module, the logic level on the CAP/CMP[x] pin of the module PCA will change. This can provide higher precision than switching IO output, because this high-speed output will not be interrupted to respond and affect the output frequency. If the CPU is used to switch the IO output, power consumption and precision are lacking.

To set a compare / capture module's high-speed output mode, set the CCAPMx.ECOM, CCAPMx.MAT, and CCAPMx.TOG bits. A match between the PCA timer / counter and the compare / capture register (CCAPx) toggles the PCA 's CAP/CMP[x] signal and sets the module's compare / capture flag (CCON.CCFx). By software setting or clearing the CAP/CMP[x] signal of the PCA, the user can choose to match the switching signal from low to high or high to low.

The user can also choose to generate an interrupt request, by setting the corresponding interrupt enable bit (CCAPMx.CCIE), when a match occurs, an interrupt request can be generated. Since the compare/capture flag interrupt cannot be cleared by hardware, the user must clear this flag in software. If the user does not change the compare/capture register in the interrupt program, PCA will re-count the compare value, and if it matches, the next rollover will occur. In the interrupt service routine, a new 16 -bit compare value can be written to the compare / capture register (CCAPx).

Note: To prevent invalid matches while updating these registers, user software should write CCAPxL first and then CCAPxH. Writing to CCAPxL clears the ECOM bit which disables the compare function, while writing to CCAPxH sets the ECOM bit and re-enables the compare function.

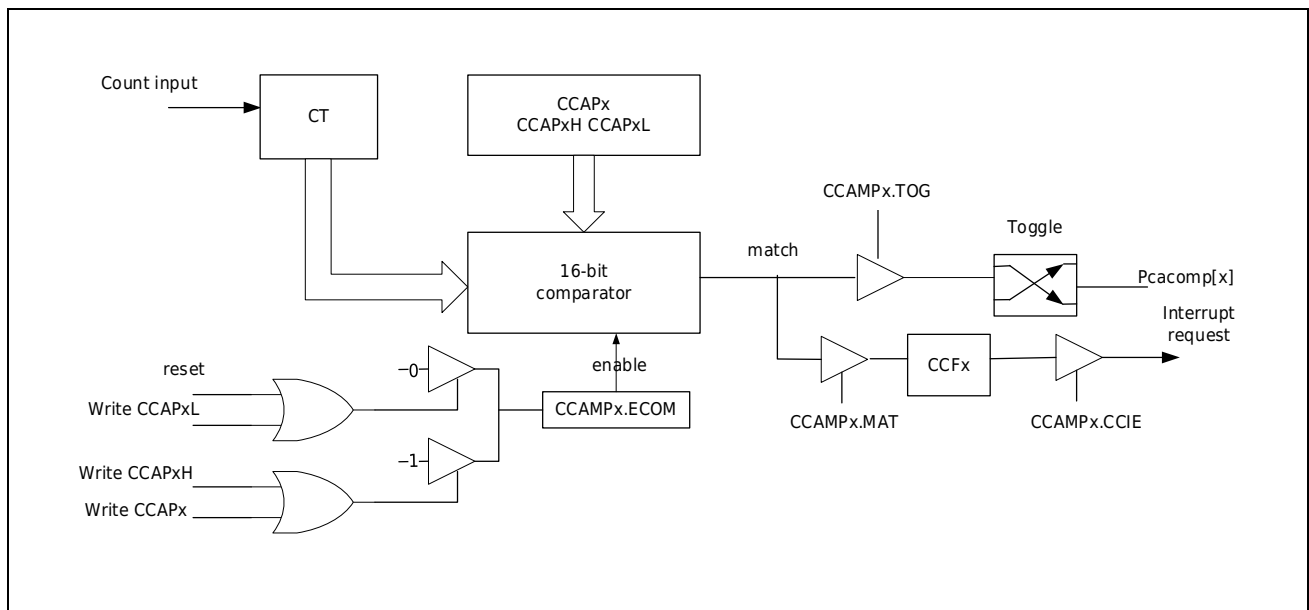


Figure 11-4 PCA Comparison Functional Block Diagram

11.2.3.3 WDT function of PCA module 4

In addition to a WDT hardware module, PCA module 4, this product also provides a 16-bit WDT with programmable frequency. This mode generates a reset signal when the count value of the PCA timer/counter matches the value stored in the module 4 compare/capture register (CCAP4). The WDT reset signal of PCA is used as an independent reset signal. Combined with external reset (RST), hardware watchdog reset (WDTRST) and LVD low voltage reset, POR power-on and power-off reset. Users are free to combine them or use them individually. Module 4 is the one with the unique PCA of the WDT pattern. When not set as WDT, it can be used independently in other modes.

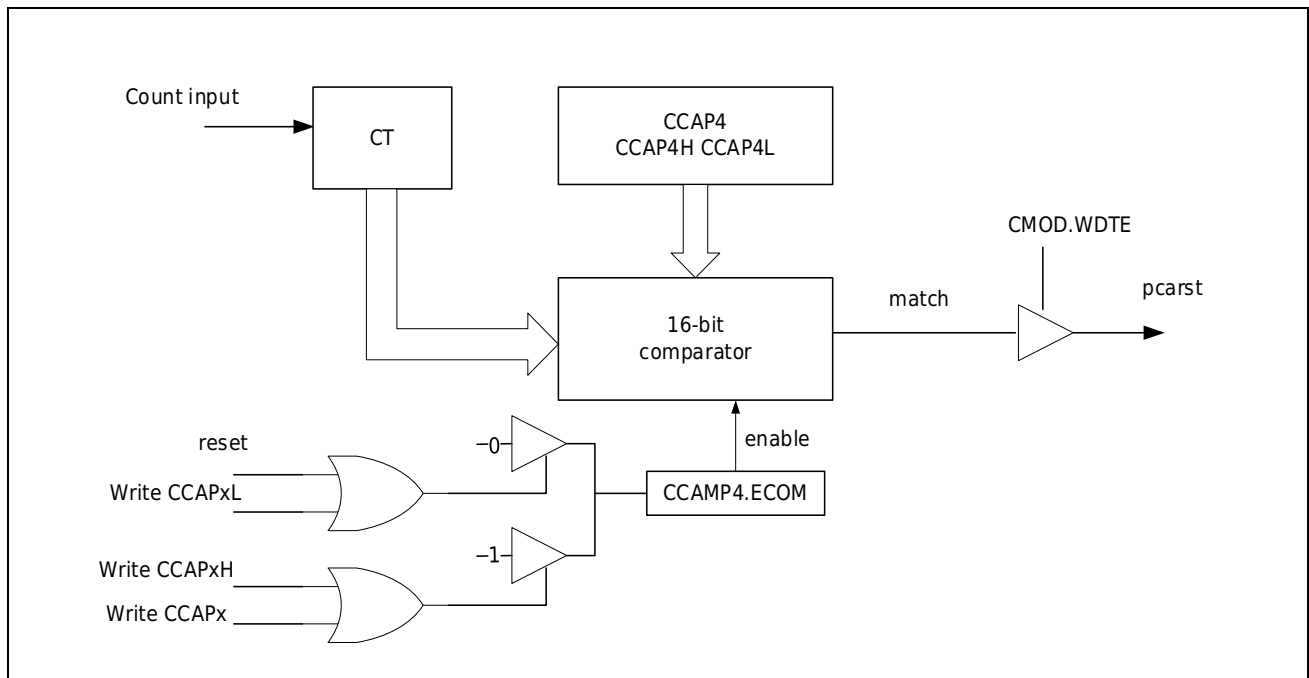


Figure 11-5 PCA WDT Functional Block Diagram

When PCA module 4 is used as WDT, CCAPM4.ECOM4, CCAPM4.MAT4 and CMOD.WDTE must be set. In addition, the PCA timer / counter can set CMOD.CPS to select different input counting frequencies.

Input a 16-bit compare value in the compare / capture register (CCAP4). In the PCA timer / counter (CNT), enter a 16-bit initial value or use a reset value (0x0000). these values multiplied by the PCA input pulse rate determines the WDT match run time. Setting the Timer / Counter Run Control bit (CCON.CR) starts the PCA WDT. The WDT of the PCA generates a reset signal every time there is a match. To prevent a PCA WDT from resetting, the user has three options.

- Periodically the comparison value CCAP4 changes, so a match never occurs.
- Periodically change the PCA timer / counter value (CNT) so a match never happens.
- Disable the module reset output signal by clearing the CMOD.WDTE bit before a match, and re-enable it later.

The first two options are more reliable because the WDT is not disabled in the third option.

The second option is not recommended if other PCA modules are used, since the five modules share a common time base. Therefore, the first option is best in most applications.

PCA WDT configuration process:

1. Configure WDT compare / capture register PCA_CCAP4
2. Configure the PCA count register PCA_CNT
3. Configure PCA_CCAMP4 to select the compare and match function
4. Configure PCA_CM0D to select the input clock and enable the WDT function
5. Start PCA

6. Select clear PCA WDT clear mode to clear PCA WDT before PCA WDT reset

11.2.3.4 PCA 8-bit PWM function

Pulse Width Modulation is a technique that uses a program to control the duty cycle, period, and phase of a waveform. Each of the five PCA modules can be independently used to generate a pulse width modulated (PWM) output at the CAP/CMP[x] pin of the corresponding PCA, with a pulse width of 8-bit resolution. The frequency of the PWM output depends on the time base of the PCA counter/timer. Use the capture / compare register CCAPxL of the module to change the duty cycle of the PWM output signal. When the low byte (CNTL) of the PCA counter / timer is equal to the value in CCAPxL, the output on the CAP/CMP[x] pin of the PCA is set to "1"; when the count value in CNTL overflows, the PCA The CAP/CMP[x] outputs are reset to "0". When the low byte CNTL of the counter / timer overflows (from 0xFF to 0x00), the value stored in CCAPxH is automatically loaded into CCAPxL without software intervention.

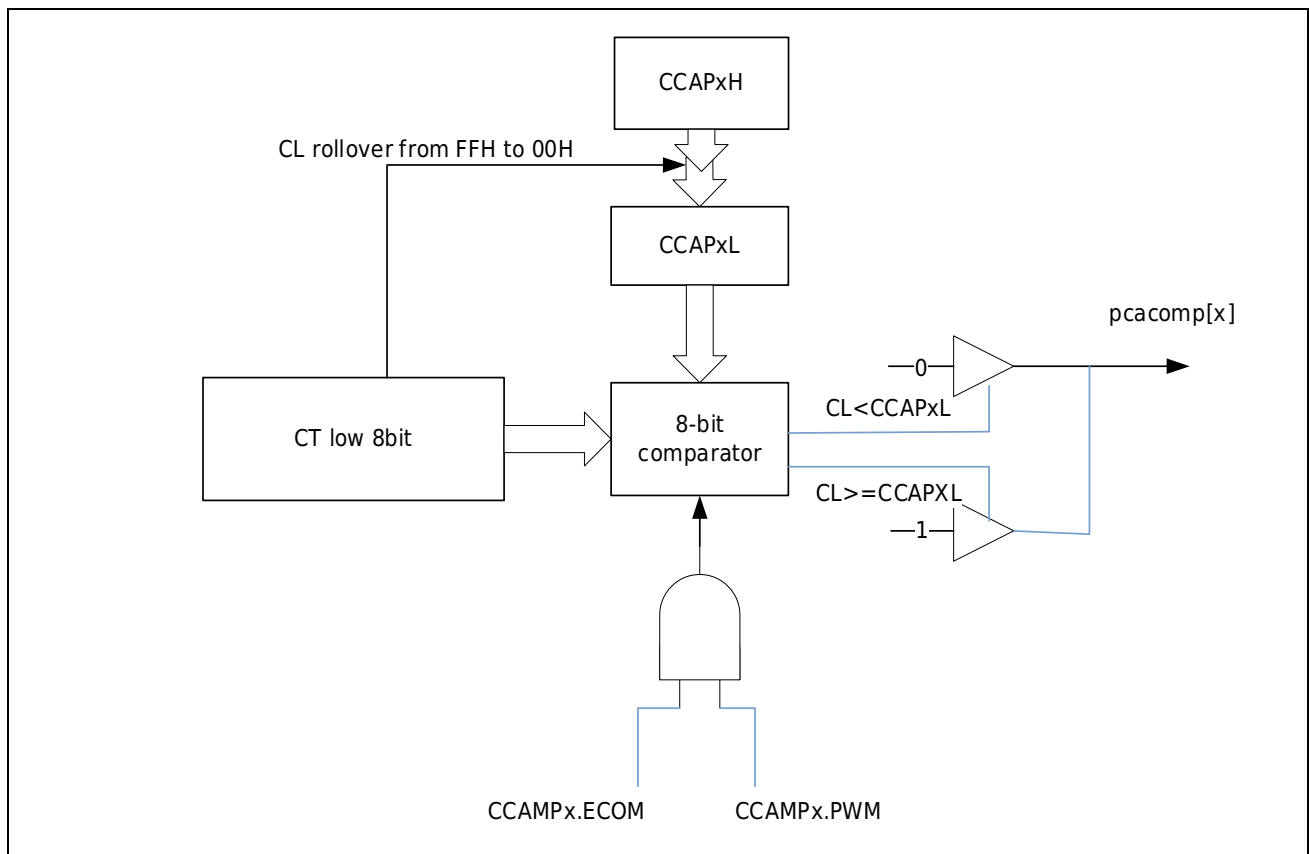


Figure 11-6 PCA PWM Functional Block Diagram

In this mode, the value of the low byte (CNTL) of the PCA timer / counter is constantly compared with the value of the low byte (CCAPxL) of the compare / capture register. When $CNTL < CCAPxL$, the output waveform is low; when $CNTL \geq CCAPxL$, the output waveform is high. When CNTL overflows, the system automatically loads the value of CCAPxH into CCAPxL and starts a new counting cycle.

CCAPxL determines the duty cycle of the current cycle, and the value of CCAPxH determines the duty cycle of the next cycle. Changing the value in CCAPxL can change the duty cycle of PWM. As shown in the figure below, adjusting the value of CCAPxL from 0 to 255 can achieve a duty cycle of 100 % to 0.4 %. In order to prevent glitches, it is recommended not to change the value of CCAPxL directly, but to change the value of CCAPxH to be automatically loaded to CCAPxL by hardware in the next cycle.

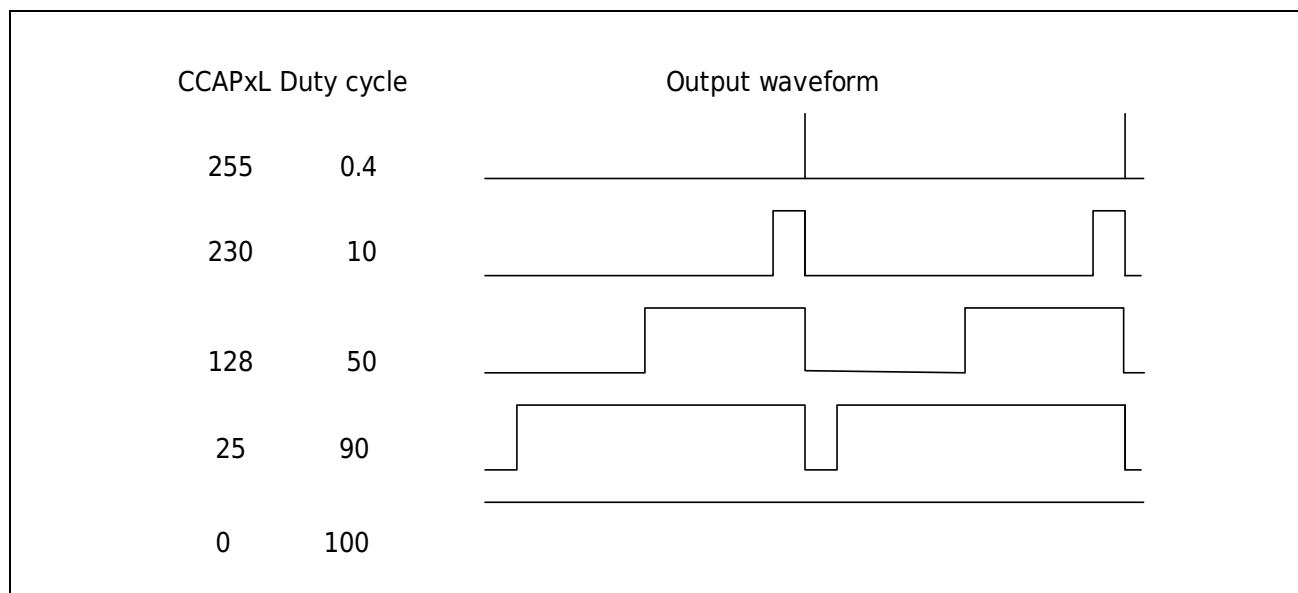


Figure 11-7 PCA PWM Output Waveform

To set a compare / capture module in PWM mode, the CCAPMx.ECOM and CCAPMx.PWM bits need to be set. In addition, the PCA timer / counter can select the input count signal frequency by programming CMOD.CSP[2:0]. Enter an 8 -bit value in CCAPxL to specify the duty cycle of the first PWM waveform. Entering an 8 -bit value in CCAPxH specifies the duty cycle of the second PWM waveform. Set the timer / counter run control bit (CCON.CR) to start the PCA timer / counter.

Table 11-1 PCA comparison capture function module setup

ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	Way of working
X	1	0	0	0	0	X	Capture with positive edge trigger
X	0	1	0	0	0	X	Trigger capture with negative edge
X	1	1	0	0	0	X	Capture with edge trigger
1	0	0	1	0	0	X	Software timer
1	0	0	1	1	0	X	High speed output
1	0	0	0	0	1	X	8 -bit pulse width modulator

11.3 Interconnection and control of PCA module and other modules

11.3.1 ECI interconnection

The ECI input can be a different input port selected externally through the IO MUX, or it can be the filtered output of the comparison of the internal VC. The VC output control register is in the VC control block.

11.3.2 PCACAP0

The capture input for channel 0 can be:

- Input port of external IO MUX
- MUX input for RX of external UART
- The compared filtered output of the internal VC1

The UART selection control register is GPIO_CTRL2 in the port control register, and the VC output control register is in the VC control module.

11.3.3 PCACAP1

The capture input for channel 1 can be:

- Input port of external IO MUX
- MUX input for RX of external UART
- The compared filtered output of the internal VC2

The UART selection control register is GPIO_CTRL2 in the port control register, and the VC output control register is in the VC control module.

11.3.4 PCACAP [4:2]

The capture input for channel 2,3,4 can be:

- Input port of external IO MUX
- MUX input for RX of external UART

UART selection control register is in the port control register GPIO_CTRL2.

11.4 PCA register description

Table 11-2 PCA register list

Base address 0X40001000

Register	Offset address	Description
CCON	0X000	PCA Control Register
CMOD	0X004	PCA Mode Register
CNT	0X008	PCA count register
ICLR	0X00C	PCA Interrupt Clear Register
CCAPM0	0x010	PCA Compare/Capture Module 0 Mode Register
CCAPM1	0x014	PCA Compare/Capture Module 1 Mode Register
CCAPM2	0x018	PCA Compare/Capture Module 2 Mode Register
CCAPM3	0x01C	PCA compare / capture module 3 mode register
CCAPM4	0x020	PCA Compare/Capture Module 4 Mode Register
CCAP0H	0X024	PCA compare/capture module 0 high 8-bit register
CCAP0L	0X028	PCA compare/capture module 0 lower 8-bit register
CCAP1H	0X02C	PCA compare/capture module 1 high 8-bit register
CCAP1L	0X030	PCA compare/capture module 1 lower 8-bit register
CCAP2H	0X034	PCA compare/capture module 2 high 8-bit register
CCAP2L	0X038	PCA compare/capture module 2 lower 8-bit register
CCAP3H	0X03C	PCA compare/capture module 3 high 8-bit register
CCAP3L	0X040	PCA comparison / capture module 3 lower 8-bit registers
CCAP4H	0X044	PCA compare/capture module 4 high 8-bit register
CCAP4L	0X048	PCA compare/capture module 4 lower 8-bit register
CCAPO	0X04C	PCA PWM and high-speed output flag register
CCAP0	0X050	16-bit register for PCA compare / capture module 0
CCAP1	0X054	16-bit register for PCA compare / capture module 1
CCAP2	0X058	16-bit register for PCA compare / capture module 2
CCAP3	0X05C	16-bit register for PCA compare / capture module 3
CCAP4	0X060	16-bit register for PCA compare / capture module 4

11.4.1 Control Register (PCA_CCON)

Offset address: 0x000

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved	CF	CR	Reserved	CCF4	CCF3	CCF2	CCF1	CCF0
	RO	R/W		RO	RO	RO	RO	RO

Bit	Symbol	Description
31:8	Reserved	Reserved bit
7	CF	PCA counter overflow flag (write invalid) When the PCA count overflows, CF is set by hardware, if the CFIE bit of the CMOD register is 1, the CF flag can generate an interrupt 1: counter overflow occurs; 0: no overflow;
6	CR	PCA counter run control bit 1: Start PCA counter counting 0: Disable PCA counter counting
5	Reserved	Reserved bit
4	CCF4	PCA counter module 4 compare / capture flag bit This bit is set by hardware when a match or capture occurs. (write invalid) When CCAPM4.CCIE is set, this flag will generate a PCA interrupt
3	CCF3	PCA counter module 3 compare / capture flag bit This bit is set by hardware when a match or capture occurs. (write invalid) When CCAPM3.CCIE is set, this flag will generate a PCA interrupt
2	CCF2	PCA counter module 2 compare / capture flag bit This bit is set by hardware when a match or capture occurs. (write invalid) When CCAPM2.CCIE is set, this flag will generate a PCA interrupt
1	CCF1	PCA counter module 1 compare / capture flag bit This bit is set by hardware when a match or capture occurs. (write invalid) When CCAPM1.CCIE is set, this flag will generate a PCA interrupt
0	CCF0	PCA counter module 0 compare / capture flag bit This bit is set by hardware when a match or capture occurs. (write invalid) When CCAPM0.CCIE is set, this flag will generate a PCA interrupt

11.4.2 Mode Register (PCA_CM0D)

Offset address: 0x004

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved	CIDL	WDTE	Reserved		CPS		CFIE	
	R/W	R/W			R/W		R/W	

Bit	Symbol	Description
31:8	Reserved	Reserved bit
7	CIDL	PCA stop working in idle mode IDLE ? 1: In sleep mode (sleep), PCA stops working 0: In sleep mode (sleep), PCA continues to work
6	WDTE	PCA WDT function enable control bit 1: Start PCA module 4 WDT function 0: Disable PCA module 4 WDT function
5:4	Reserved	Reserved bit
3:1	CPS	Clock frequency division selection and clock source selection 000: PCLK/32 001: PCLK/16 010: PCLK/8 011: PCLK/4 100: PCLK/2 101: Timer0 overflow 110: Timer1 overflow 111: ECI external clock, clock PCLK four-frequency sampling
0	CFIE	PCA counter interrupt enable control signal 1: enable interrupt 0: disable interrupt

11.4.3 Count register (PCA_CNT)

Offset address: 0x008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT															
R/W															

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15:0	CNT	The value of the timer counter CNT can only be written in the PCA stop state, otherwise the write is invalid

11.4.4 Interrupt Clear Register (PCA_ICLR)

Offset address: 0x00C

Reset value: 0x0000/ 009Fh

31:8	7	6	5	4	3	2	1	0
Reserved	CF	Reserved		CCF4	CCF3	CCF2	CCF1	CCF0
	W0			W0	W0	W0	W0	W0

Bit	Symbol	Description
31:8	Reserved	Reserved bit
7	CF	The PCA counter overflow flag is cleared (software writes 0 to clear, writes 1 to be invalid), and the read value is 1
6:5	Res.	Reserved bit
4	CCF4	PCA counter module 4 compare / capture flag clear (software write 0 to clear, write 1 is invalid), the read value is 1
3	CCF3	PCA counter module 3 compare / capture flag clear (software write 0 to clear, write 1 is invalid), the read value is 1
2	CCF2	PCA counter module 2 compare / capture flag clear (software write 0 to clear, write 1 is invalid), the read value is 1
1	CCF1	PCA counter module 1 compare / capture flag clear (software write 0 to clear, write 1 is invalid), the read value is 1
0	CCF0	PCA counter module 0 compare / capture flag clear (software write 0 to clear, write 1 is invalid), the read value is 1

11.4.5 Compare Capture Mode Register (PCA_CCAPM0~4)

Offset address

CCAPM0: 0x010; CCAPM1: 0x014; CCAPM2: 0x018;

CCAPM3: 0x01C; CCAPM4: 0x020;

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved		ECOM	CAPP	CAPN	MAT	TOG	PWM	CCIE
		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Symbol	Description
31:7	Reserved	Reserved bit
6	ECOM	Enable comparator function control bit 1: Enable the comparator function; 0: Disable the comparator function; When PCA is used for software counter, high-speed output, PWM mode, WDT mode, set ECOM Writing to the CCAMPHx or CCAMPx registers automatically sets the ECOM bit; writing to the CCAMPLx registers automatically clears the ECOM bit
5	CAPP	Positive edge capture control bit 1: Enable rising edge capture; 0: Disable rising edge capture
4	CAPN	Negative edge capture control bit 1: Enable falling edge capture; 0: Disable falling edge capture
3	MAT	Allow match control bits 1: Once the PCA count value matches the value of the compare / capture register of the module, the interrupt flag CCFx (x=0-4) registered in CCON will be set 0: Disable match function
2	TOG	Toggle control bit 1: Work in the PCA high-speed output mode, once the value of the PCA counter matches the value of the compare / capture register of the module, the CCPx pin will flip 0: disable flip function
1	PWM	PWM Control Bits 1: Enable CCPx pin as PWM output 0: Disable PWM pulse width modulation function PWM function is valid only when CCAPMx[6:0]=100_0010
0	CCIE	PCA enable interrupt 1: Enable compare / capture interrupt 0: PCA compare / capture function interrupt disabled

11.4.6 Compare the upper 8 bits of the capture data register (PCA_CCAP0~4H)

Offset address

CCAP0H: 0x024; CCAP1H: 0x02C; CCAP2H: 0x034;

CCAP3H: 0x03C; CCAP4H: 0x044;

Reset value: 0x0000 0000

31:8 7 6 5 4 3 2 1 0

Reserved	CCAPx[15: 8]							
	R/W							

Bit	Symbol	Description
31:8	Reserved	Reserved bit
7:0	CCAPx[15:8]	Compare / Capture Mode High 8-bit Register When the PCA mode is used in compare / capture mode, it is used to save the upper 8 bits of the 16-bit capture count value; writing the CCAPxH register will automatically set the ECOM bit of the register CCAPMx. When PCA mode is used in PWM mode, it is used to control the output duty ratio load register. When the lower 8 bits of the counter overflow, the load register will be automatically updated to the PWM comparison register.

11.4.7 Compare the lower 8 bits of the capture data register (PCA_CCAP0~4L)

Offset address

CCAP0L: 0x028; CCAP1L: 0x030; CCAP2L: 0x038;

CCAP3L: 0x040; CCAP4L: 0x048;

Reset value: 0x0000 0000

31:8 7 6 5 4 3 2 1 0

Reserved	CCAPx[7: 0]							
	R/W							

Bit	Symbol	Description
31:8	Reserved	Reserved bit
7:0	CCAPx[7:0]	Compare / capture mode lower 8-bit register When the PCA mode is used in compare / capture mode, it is used to save the lower 8 bits of the 16-bit capture count value; writing the CCAPxH register will automatically clear the ECOM bit of the register CCAPMx. When the PCA mode is used in the PWM mode, it is used to control the output duty ratio comparison register. In the PWM mode, the value of the lower 8 bits of the counter is less than the value of CCAPx[7:0] and the PWM output is low, otherwise the PWM output is high. flat.

11.4.8 Compare capture 16 -bit register (PCA_CCAP0~4)

Offset address

CCAP0: 0x050; CCAP1: 0x054; CCAP2: 0x058;

CCAP3: 0x05C; CCAP4: 0x060;

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCAPx[15: 0]															
R/W															

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15:0	CCAPx	Compare / capture mode 16 -bit register When the PCA mode is used in compare / capture mode, it is used to save the 16 -bit capture count value; writing the CCAPx register will set the ECOM bit of the register CCAPMx. Writing the CCAPX register is equivalent to writing the two 8 -bit registers CCAPxL and CCAPxH. This register can be read and written directly in compare / capture mode. In PWM mode, use the CCAPxL and CCAPxH registers

11.4.9 Compare High Speed Output Flag Register (PCA_CCAPO)

Offset address: 0x04C

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved				CCAPO4	CCAPO3	CCAPO2	CCAPO1	CCAPO0
				R/W	R/W	R/W	R/W	R/W

Bit	Symbol	Description
31:5	Reserved	Reserved bit
4	CCAPO4	Compare the output value of module 4
3	CCAPO3	Compare the output value of module 3
2	CCAPO2	Compare the output value of module 2
1	CCAPO1	Compare the output value of module 1
0	CCAPO0	Compare the output value of module 0

12 Advanced Timer (TIM4/5/6)

12.1 Introduction to Advanced Timers

Timer4/5/6 which contains three timers. Timer4/5/6 are high-performance counters with the same function, which can be used to count and generate different forms of clock waveforms. One timer can generate a complementary pair of PWM or independent 2- way PWM output, which can capture external input for pulse width or period Measurement.

The basic functions and characteristics of advanced timers are shown in the table.

Table 12-1 Basic Features of Advanced Timer

Waveform mode	Sawtooth wave, triangular wave
Basic functions	● Up and down counting direction
	● Software synchronization
	● Hardware synchronization
	● Cache function
	● Orthogonal code count
	● General purpose PWM output
	● Brake protection
	● AOS related action
Interrupt type	Count comparison match interrupt
	Count cycle match interrupt
	Dead time error interrupt

Table 12-2 Advanced Timer port list

Port name	Direction	Function
TIMx_CHA	Input/Output	Quadrature encoding count clock input port or capture input port or comparison output port (x=4~6) 2) Hardware start, stop, clear condition input port
TIMx_CHB		
TIMTRIA	Input	Hardware count clock input port or capture input port Hardware start, stop, clear condition input port
TIMTRIB		
TIMTRIC		
TIMTRID		

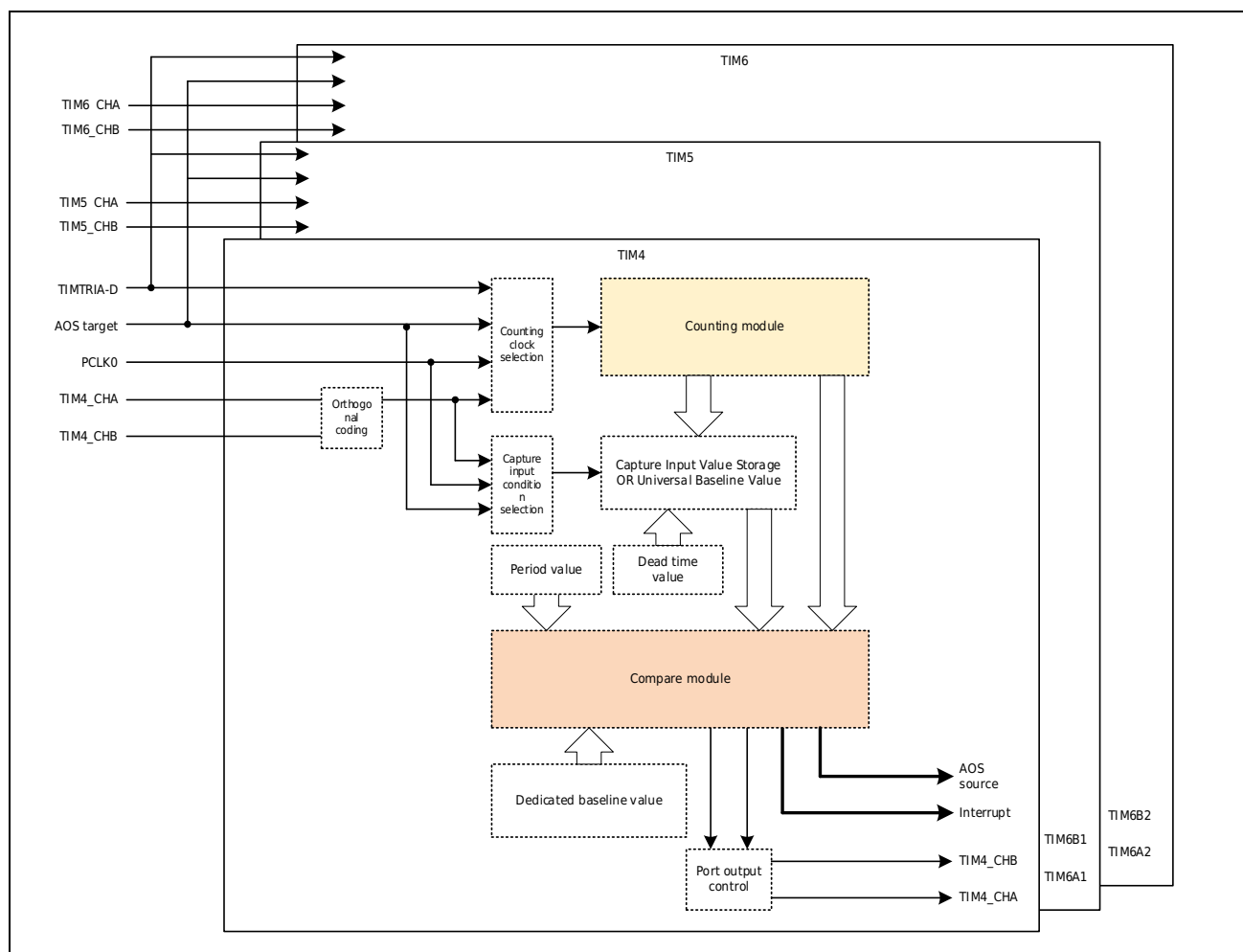


Figure 12-1 Advanced Timer Block Diagram

12.2 Advanced Timer Function Description

12.2.1 Basic action

12.2.1.1 Basic Waveform Mode

Timer4/5/6 has 2 basic counting waveform modes, sawtooth wave mode and triangle wave mode. The waveform mode is subdivided due to different internal counting actions. The triangular wave mode is divided into triangular wave A mode and triangular wave B mode. The basic waveforms of sawtooth and triangle waves are shown in Figure 12-2 and Figure 12-3. The difference between the triangular wave A mode and the triangular wave B mode is that there is a difference in the buffer transfer. The triangular wave A mode only has one buffer transfer (valley point) in one cycle, while the triangular wave B mode has two buffer transfers (peak point and valley point) in one cycle.

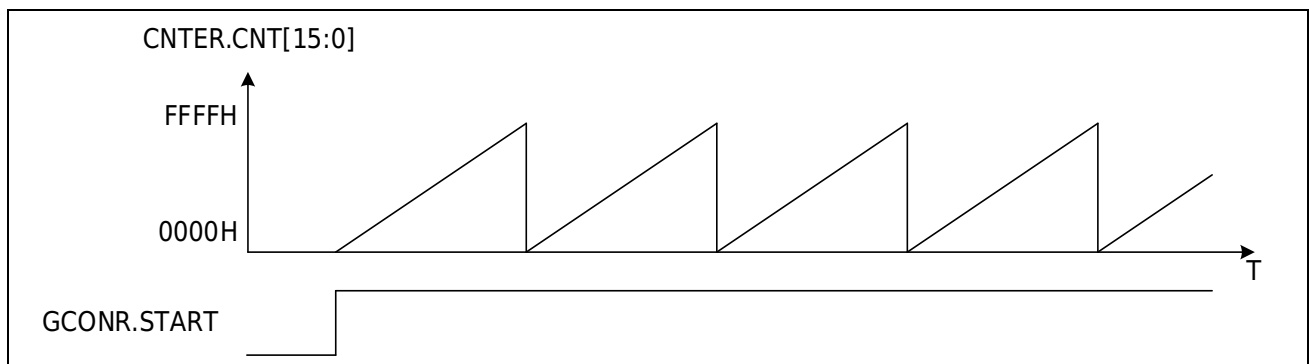


Figure 12-2 Sawtooth Waveform (Counting Up)

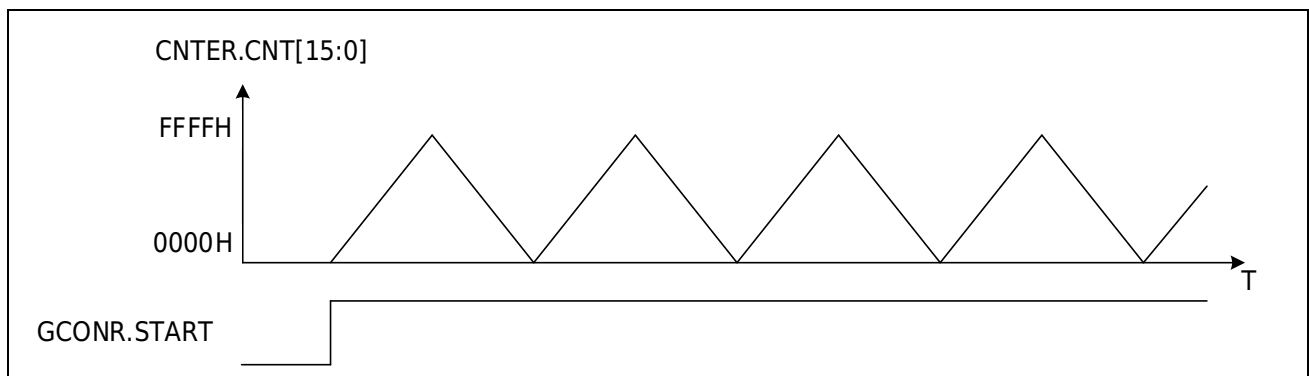


Figure 12-3 Triangle Waveform

12.2.1.2 Comparison output

Timer4/5/6 A timer has 2 comparison output ports (CHxA, CHxB), which can output a specified level when the count value matches the count reference value. The GCMAR and GCMBR registers correspond to the count comparison reference values of CHxA and CHxB respectively. When the count value of the counter is equal to GCMAR, the CHxA port outputs the specified level; when the count value of the counter is equal to GCMBR, the CHxB port outputs the specified level.

The counting start level, stop level, and counting comparison match level of CHxA and CHxB ports can be set by PCONR.STACA, PCONR.STPCA, PCONR.STASTPSA, PCONR.CMPCA [1:0 of the port control register (PCONR)], PCONR.PERCA[1:0] and PCONR.STACB, PCONR.STPCB, PCONR.STASTPSB, PCONR.CMPCB[1:0], PCONR.PERCB[1:0] bit settings. Figure 12-4 is an example of the comparison output operation.

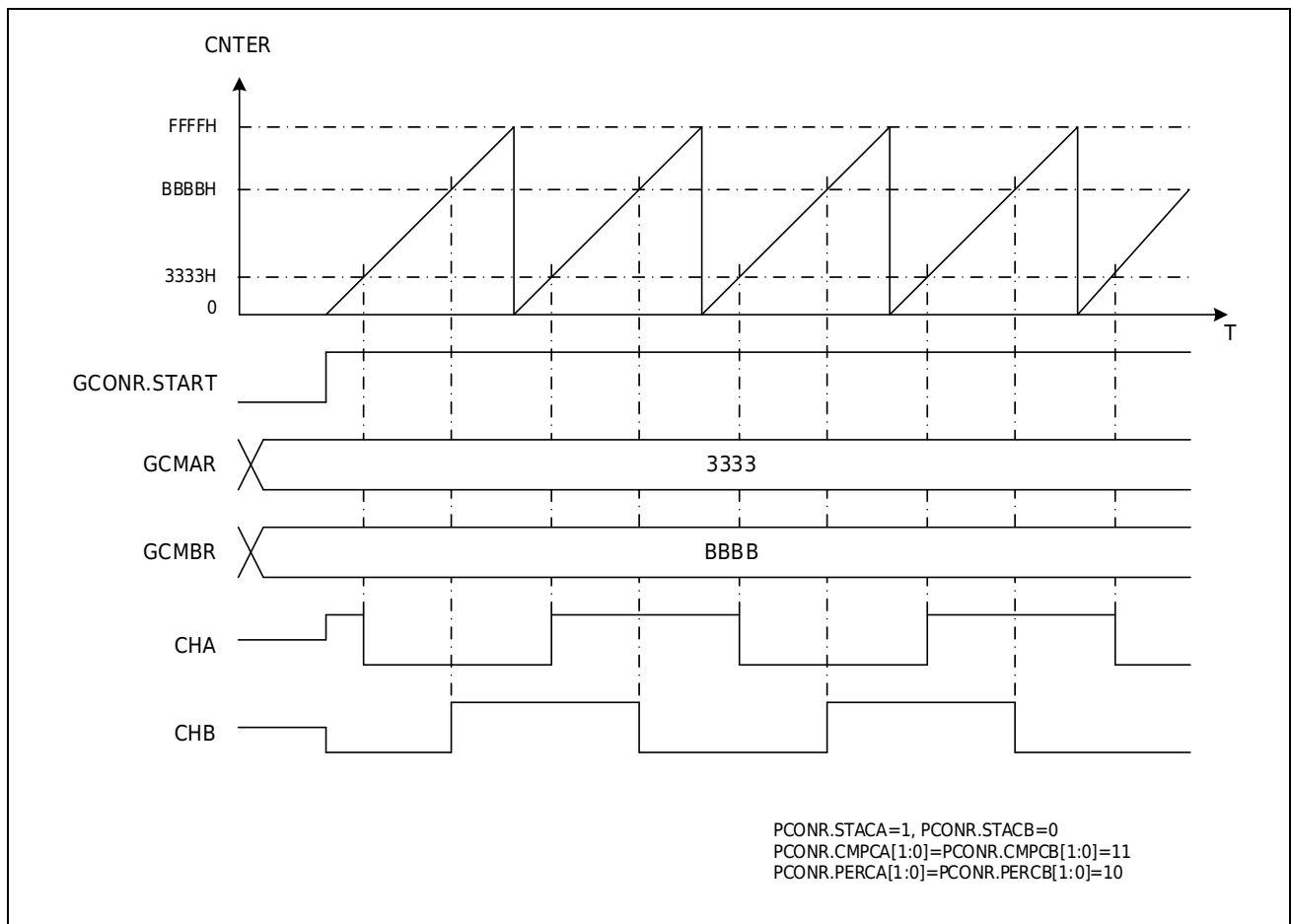


Figure 12-4 Compare output action

12.2.1.3 Capture input

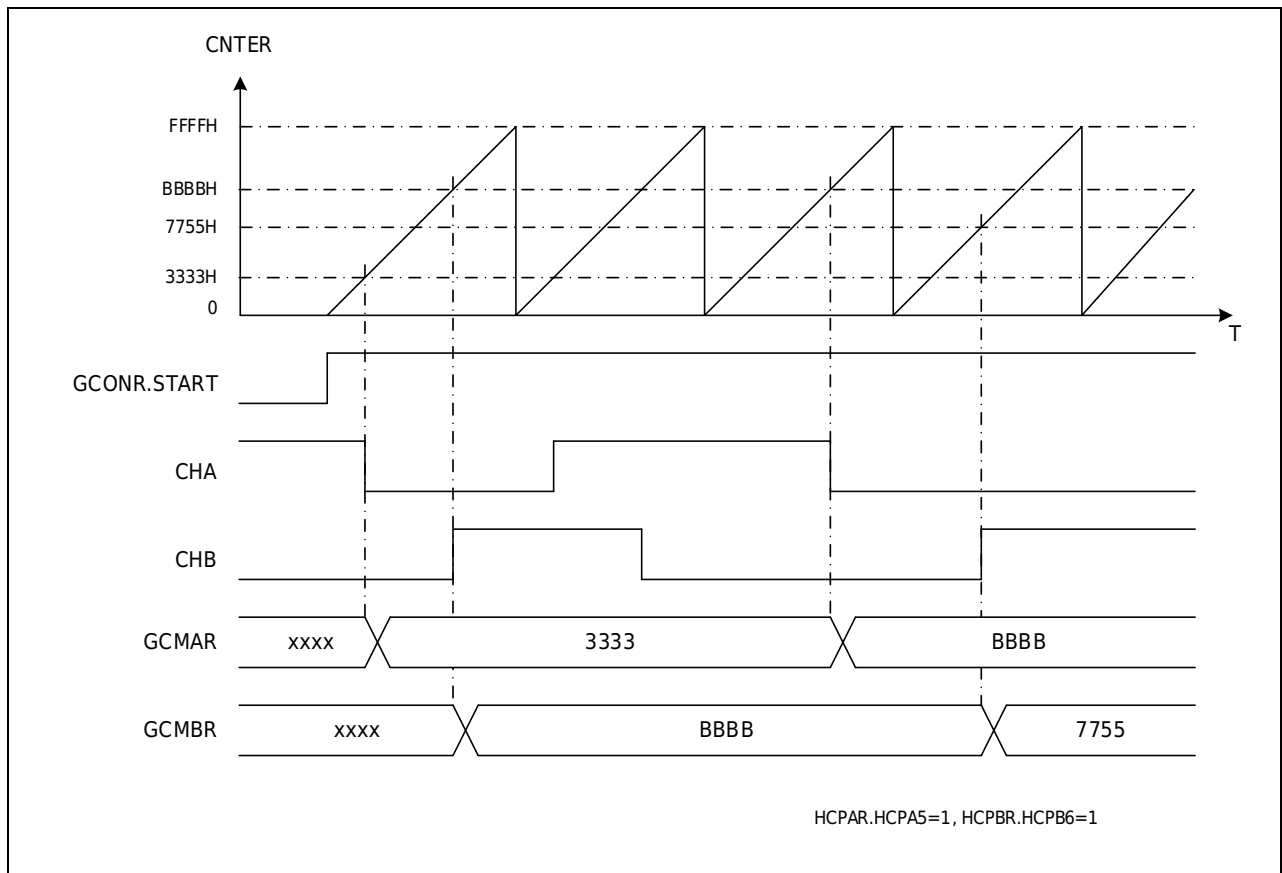


Figure 12-5 Capturing Input Actions

Timer4/5/6 all have a capture input function, with 2 sets of capture input registers (GCMAR, GCMBR), used to save the captured count value. Set the PCONR.CAPCA and PCONR.CAPCB bits of the port control register (PCONR) to 1, and the capture input function is valid. When the corresponding capture input condition is set and the condition is valid, the current count value is saved to the corresponding register (GCMAR, GCMBR).

The condition of each capture input can be AOS event trigger, TIMTRIA-TIMTRID input, CHxA or CHxB input, etc. The specific condition selection can be set through the hardware capture event selection register (HCPAR, HCPBR). Figure 12-5 is an example of an action to capture input.

12.2.2 Timer selection

Timer4/5/6 can have the following options:

- PCLK and 2, 4, 8, 16, 64, 256, 1024 frequency division of PCLK (GCONR.CKDIV[2:0] setting)
- AOS event trigger input (set by HCUPR.HCUP[19:16] or HCDOR.HCDO[19:16])
- CHxA and CHxB (set by HCUPR.HCUP[7:0] or HCDOR.HCDO[7:0])
- TIMTRIA-TIMTRID (HCUPR.HCUP [15:8] or HCDOR.HCDO [15:8] setting)

It can be seen from the above description that clocks b, c, and d are independent of each other, and can be set to be valid or invalid respectively, and when clocks b, c, and d are selected, clock a is automatically invalid.

12.2.3 Counting direction

Timer4/5/6 can be changed by software. The method of changing the counting direction is slightly different in different waveform modes.

12.2.3.1 Sawtooth counting direction

In sawtooth wave mode, the counting direction can be set while the counter is counting or stopped.

When counting up, set GCONR.DIR=0 (count down), then the counter will change to down count mode after counting to overflow; when counting down, set GCONR.DIR=1 (count up), the counter will change to count up mode after counting to underflow.

When counting is stopped, the GCONR.DIR bit is set. GCONR.DIR will not be reflected in the counting until the counting starts until overflow or underflow.

12.2.3.2 Triangular wave counting direction

In triangle wave mode, the counting direction can only be set when the counter is stopped. It is invalid to set the counting direction in counting.

When counting is stopped, the GCONR.DIR bit is set. GCONR.DIR will not be reflected in the counting until the counting starts until overflow or underflow.

12.2.4 Digital filtering

The CHxA, CHxB, TIMTRIA~D port inputs of Timer4/5/6 all have digital filter function. The filter function of the corresponding port can be enabled by setting the relevant enable bit of the filter control register (FCONR). The reference clock used for filtering is also set by the filter control register (FCONR).

When the filtered sampling reference clock is sampled to the same level three times on the port, the level is transferred to the module as an effective level. A level less than three times consistent will be filtered out as external interference and not transmitted to the module. Its operation is shown in Figure 12-6, for example.

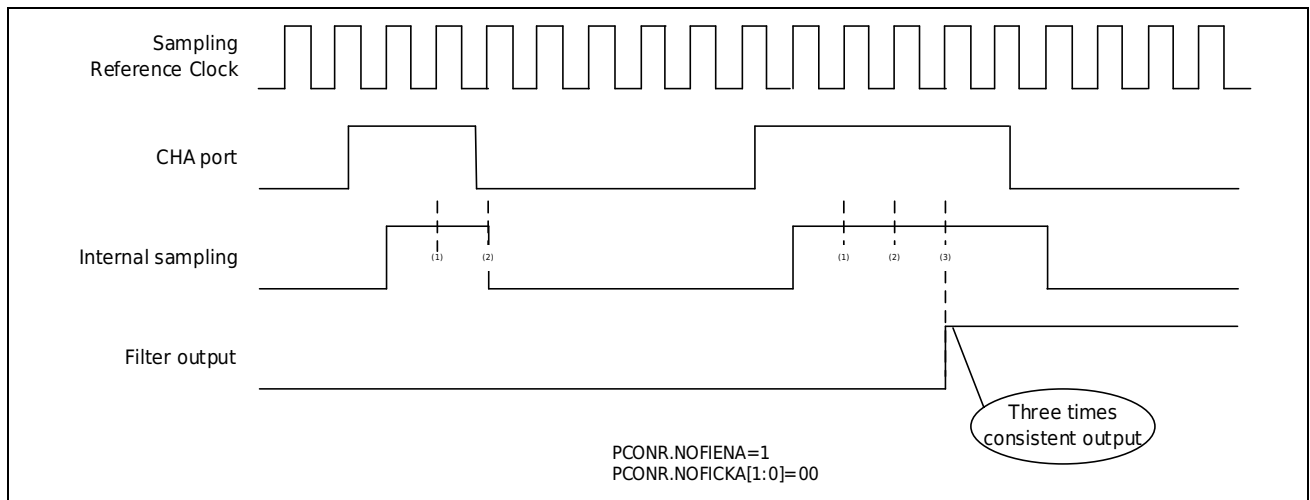


Figure 12-6 Filter function of the capture input port

TIMTRIA~D ports are a group of ports shared by Timer4/5/6. The digital filtering function of this group of ports is only implemented on Timer4, and the digital filtering function of other timers Timer5/6 is invalid for this group of ports.

12.2.5 Software synchronization

12.2.5.1 Software synchronization start

Timer4/5/6 can realize the synchronous start of the target Timer4/5/6 by setting the relevant bits of the software synchronous start register (SSTAR).

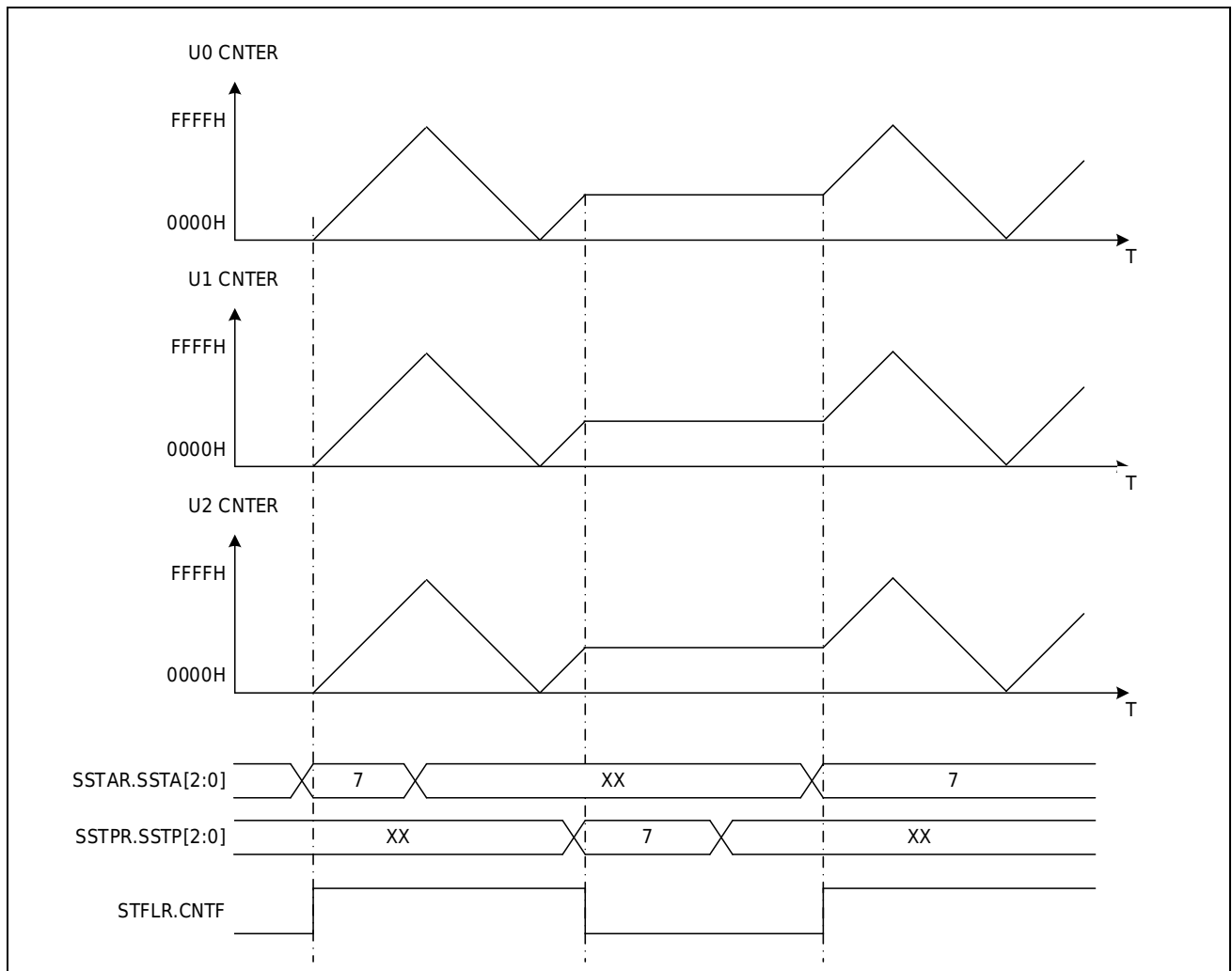


Figure 12-7 Software Synchronization Action

12.2.5.2 Software co-stop

Timer4/5/6 can realize the synchronous stop of the target Timer4/5/6 by setting the relevant bits of the software synchronous stop register (SSTPR).

12.2.5.3 Software synchronous clear

Timer4/5/6 can realize the synchronous clear of the target Timer4/5/6 by setting the relevant bits of the software synchronous clear register (SCLRR).

As shown in Figure 12-7, if SSTAR. SSTA0=SSTAR. SSTA1=SSTAR. SSTA2 is set for Timer4, software synchronization startup for Timer4/5/6 can be achieved.

Software synchronous action related registers (SSTAR, SSTPR, SCLRR) are a group of registers that are independent of Timer4/5/6 and shared between each TIM. Each bit of this group of registers is only valid when writing 1, and writing 0 is invalid. When reading the SSTAR register, the counter status of each timer will be read, and when reading SSTPR or SCLRR, 0 will be read.

12.2.6 Hardware synchronization

In addition to having 2 general-purpose input ports (CHxA, CHxB) independently, each timer also has 4 external general-purpose input ports (TIMTRIA, TIMTRIB, TIMTRIC, TIMTRID) and 4 AOS targets, which can realize the hardware between timers Synchronized actions.

12.2.6.1 Hardware sync start

Each Timer4/5/6 can choose to use hardware to start the counter, select the timer with the same hardware start condition to realize synchronous start when the start condition is valid. The specific hardware start condition is determined by the setting of the hardware start event selection register (HSTAR).

12.2.6.2 Hardware co-stop

Each Timer4/5/6 can choose to stop the counter by hardware, and the timer with the same hardware stop condition can realize synchronous stop when the stop condition is valid. The specific hardware stop condition is determined by the setting of the hardware stop event selection register (HSTPR).

12.2.6.3 Hardware synchronous clear

Each Timer4/5/6 can choose to clear the counter by hardware, and select the timer with the same hardware clearing condition to realize synchronous clearing when the clearing condition is valid. The specific hardware clear condition is determined by the setting of the hardware clear event select register (HCLRR).

12.2.6.4 Hardware Synchronous Capture Input

Each Timer4/5/6 can choose to use hardware to realize the capture input function, and select the timer with the same capture input function condition to realize synchronous capture input when the capture input function condition is valid. The specific hardware capture input function conditions are determined by the settings of the hardware capture event selection registers (HCPAR, HCPBR).

12.2.6.5 Hardware sync count

Timer4/5/6 can choose to use the hardware input as CLOCK to count, and select the timer with the same hardware counting condition to realize synchronous counting when the hardware counting CLOCK is valid. The specific hardware counting conditions are determined by the settings of the hardware increment event selection register (HCUPR) and the hardware decrement event selection register (HCDOR).

When the hardware synchronous counting function is selected, only the external input clock source is selected, which does not affect the start, stop, and reset actions of the counter. The start, stop, and reset of the counter also need to be set separately.

Figure 12-8 shows an example of the hardware synchronous operation of Timer4/5/6.

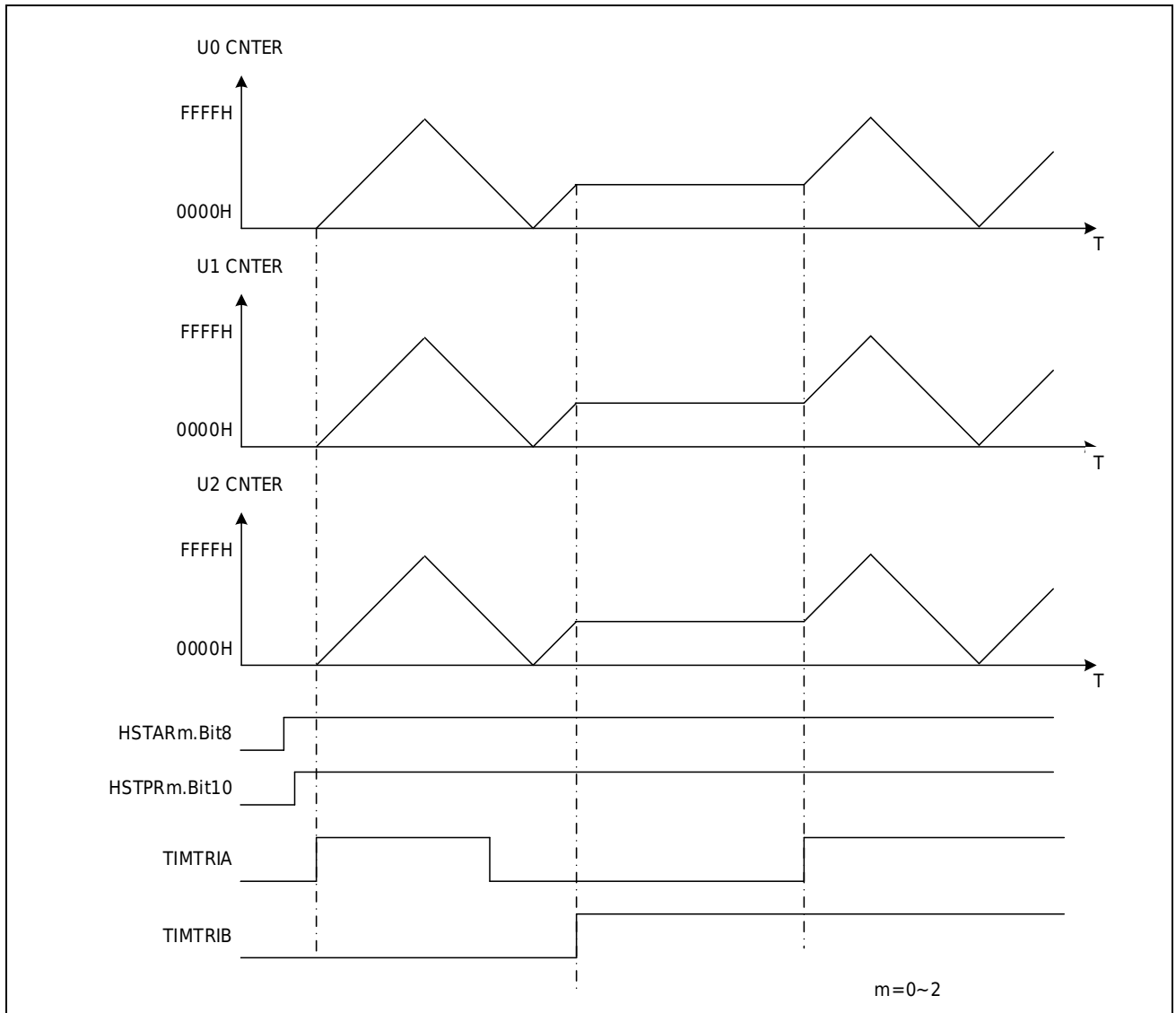


Figure 12-8 Hardware Synchronous Action

12.2.7 Cache function

Cache action means that by setting the cache control register (BCONR), at the time point of cache transmission, the following events are selected:

- The value of the general period reference buffer register (PERBR) is automatically transferred to the general period reference register (PERAR)
- The value of the general comparison reference value buffer register (GCMCR, GCMDR) is automatically transferred to the general comparison reference value register (GCMAR, GCMBR) (when comparing output)
- The value of the general comparison reference value register (GCMAR, GCMBR) is automatically transferred to the general comparison reference value buffer register (GCMCR, GCMDR) (when capturing input)

Figure 12-9 shows the timing chart of the single cache method of the universal comparison reference value register when comparing output actions. It can be seen from the figure that changing the value of the general comparison reference value register (GCMAR) during counting can adjust the output duty cycle, and changing the value of the general period reference value register (PERAR) can adjust the output cycle.

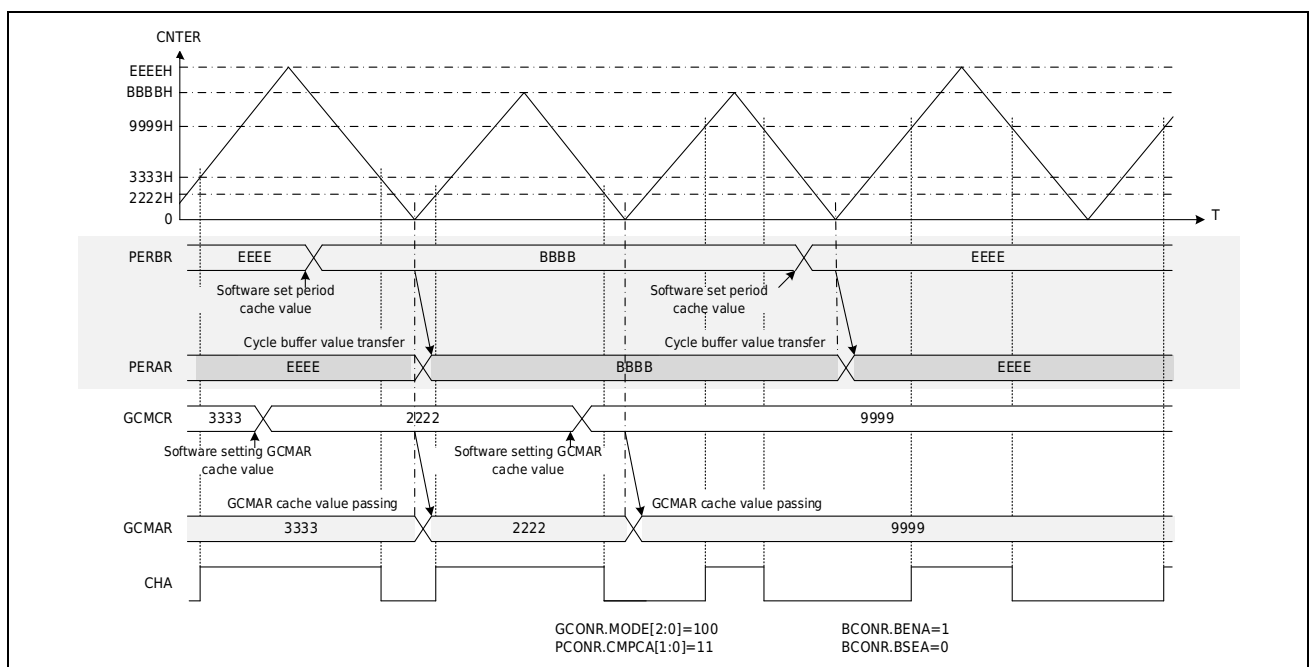


Figure 12-9 Single-buffer mode comparison output timing

12.2.7.1 Cache delivery time point

12.2.7.2 General cycle reference value cache transmission time point

The transmission time point of the periodic reference value buffer is the counting overflow point or the counting down point when the sawtooth wave occurs, and the counting valley point when the triangular wave occurs.

12.2.7.3 Universal baseline value cache delivery time point

In sawtooth wave mode, set BCONR.BENA=1 or BCONR.BNEB=1, and the cache operation is valid. Buffer transfers occur at overflow or underflow points.

In triangular wave A mode, set BCONR.BENA=1 or BCONR.BNEB=1, the cache operation is valid. Buffer transfers occur at count valley points.

In triangular wave B mode, set BCONR.BENA=1 or BCONR.BNEB=1, the cache operation is valid. Buffer transfers occur at count troughs and count peaks.

12.2.7.4 Capture input value buffer transfer time point

The capture input action cache transmission time point is when the capture input action is performed.

12.2.7.5 Buffer transfer during clear action

In the sawtooth wave counting mode or hardware counting mode, if there is a clearing action during the normal comparison output operation, the general cycle reference value, general comparison reference value, etc. will be buffered once according to the corresponding buffer operation setting status.

12.2.8 General PWM output

12.2.8.1 PWM spread spectrum output

In order to reduce the external interference of the PWM output, there is a spread spectrum configuration in the PWM output stage. Each PWM output cycle fine-tunes the phase of the PWM output.

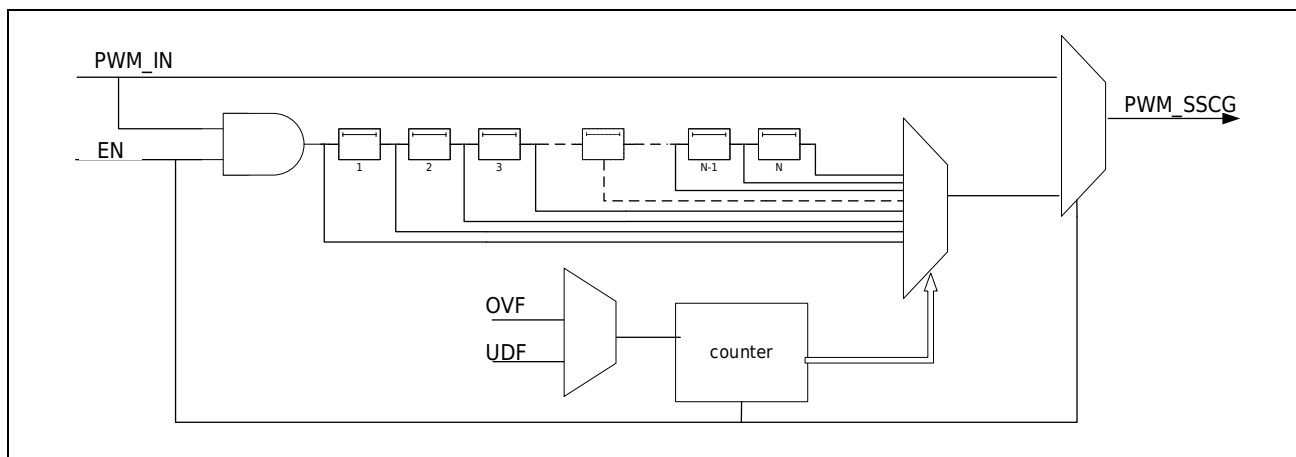


Figure 12-10 PWM Spread Spectrum Output Schematic

12.2.8.2 Independent PWM output

The 2 ports CHxA and CHxB of each timer can independently output PWM waves. As shown in Figure 12-11, the CHA port of Timer6 outputs PWM wave.

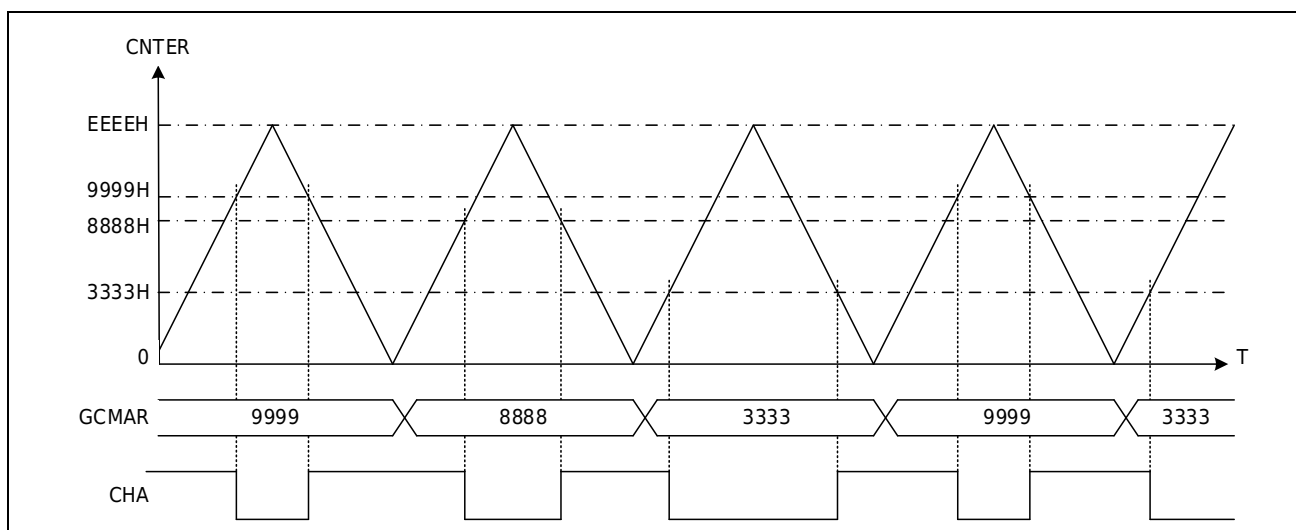


Figure 12-11 CHA output PWM wave

12.2.8.3 Complementary PWM output

The CHxA port and the CHxB port can be combined to output complementary PWM waveforms in different modes.

12.2.8.3.1 Software setting GCMBR complementary PWM output

Software setting GCMBR complementary PWM output means that in sawtooth wave mode, triangular wave A mode, and triangular wave B mode, the value of the general comparison reference value register (GCMBR) used for CHxB port waveform output is directly set by the register, and the general comparison reference The value of the value register (GCMAR) is not directly related.

Figure 12-12 shows an example of software setting GCMBR complementary PWM wave output.

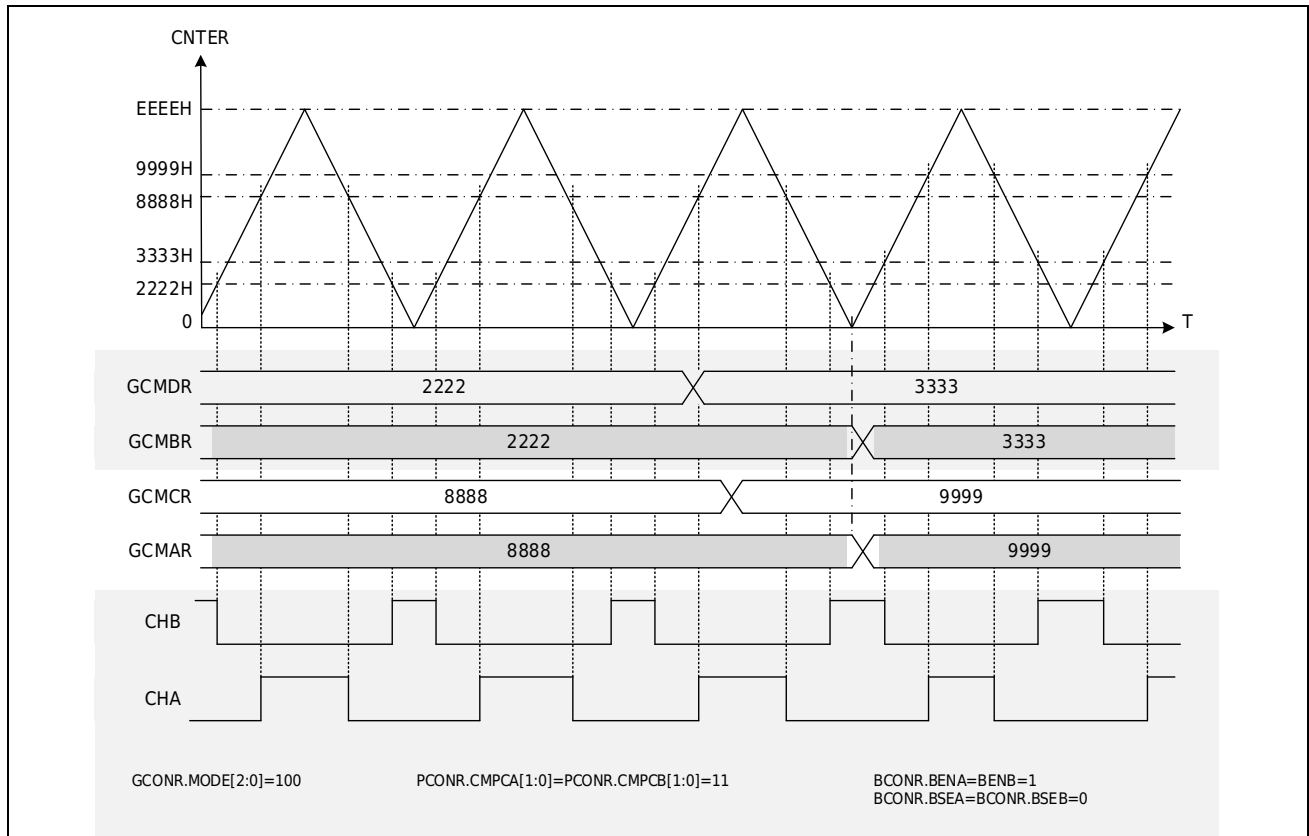


Figure 12-12 Software setting GCMBR Complementary PWM wave output in triangular wave A mode

12.2.8.3.2 Hardware setting GCMBR complementary PWM output

Hardware setting GCMBR Complementary PWM output means that in triangular wave A mode and triangular wave B mode, the value of the general comparison reference value register (GCMBR) used for CHxB port waveform output is determined by the general comparison reference value register (GCMAR) and the dead time reference value register (DTUAR, DTDAR) is determined.

Figure 12-13 is an example of hardware setting GCMBR complementary PWM wave output.

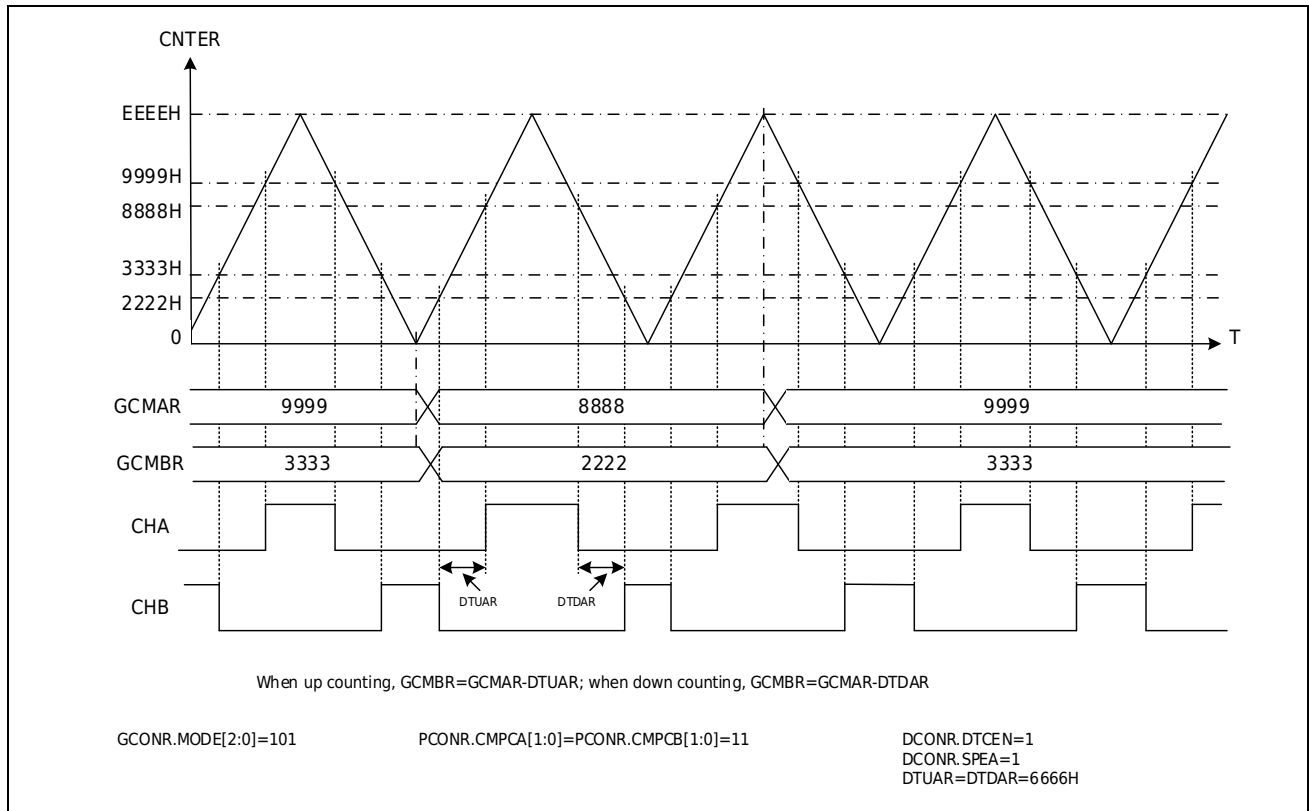


Figure 12-13 Triangular wave B mode hardware setting GCMBR Complementary PWM wave output (symmetrical dead zone)

12.2.8.4 Multi-phase PWM output

The CHxA and CHxB ports of each timer can output 2 -phase independent PWM waves or a group of complementary PWM waves. Multiple timers can be combined and combined with software and hardware synchronous actions to realize multi-phase PWM wave output. As shown in Figure 12-14, the combination of Timer4, Timer5, and Timer6 outputs 6-phase PWM waves; as shown in Figure 12-15, the combination of Timer4, Timer5, and Timer6 outputs 3 complementary PWM waves.

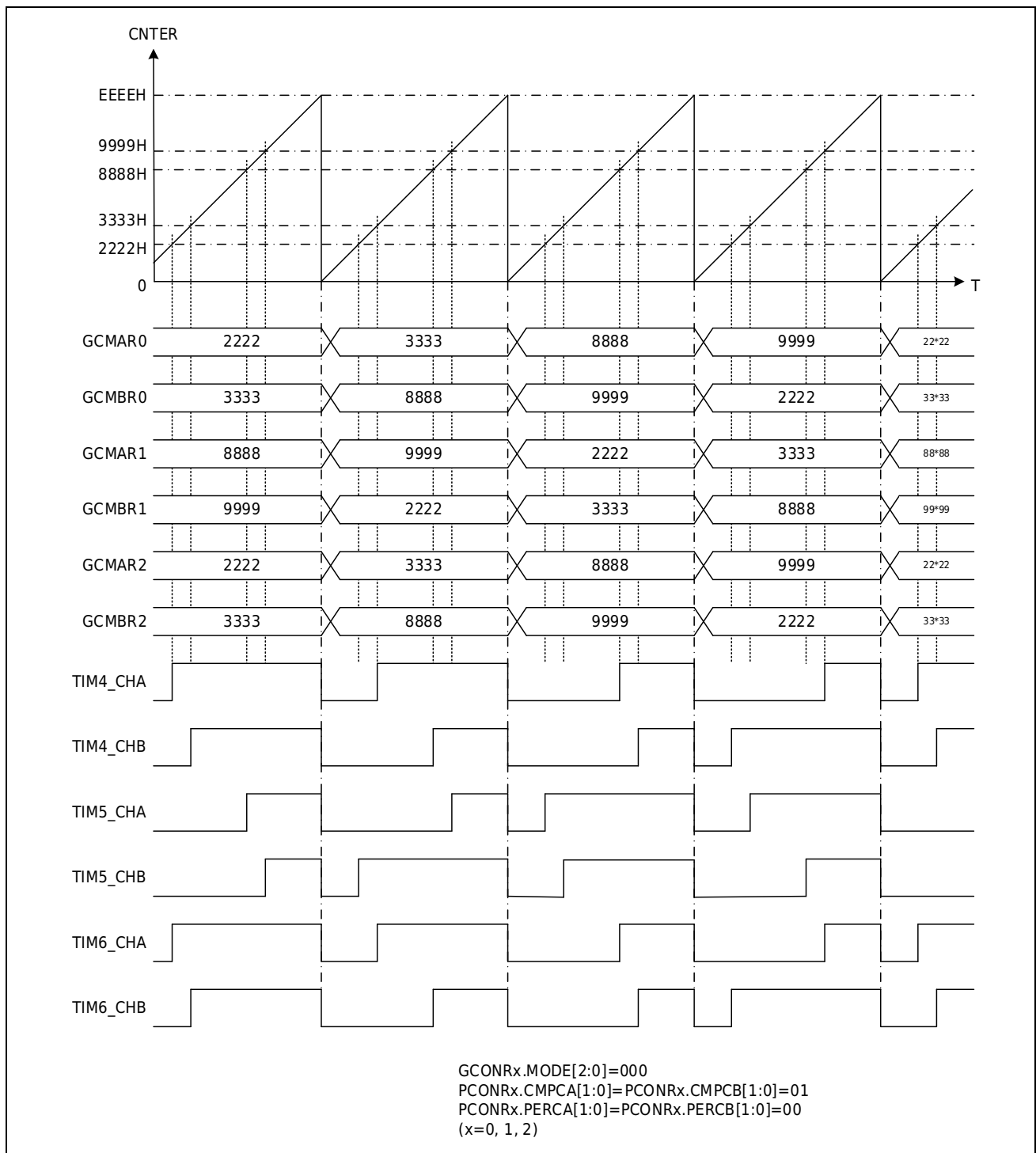


Figure 12-14 6-phase PWM wave

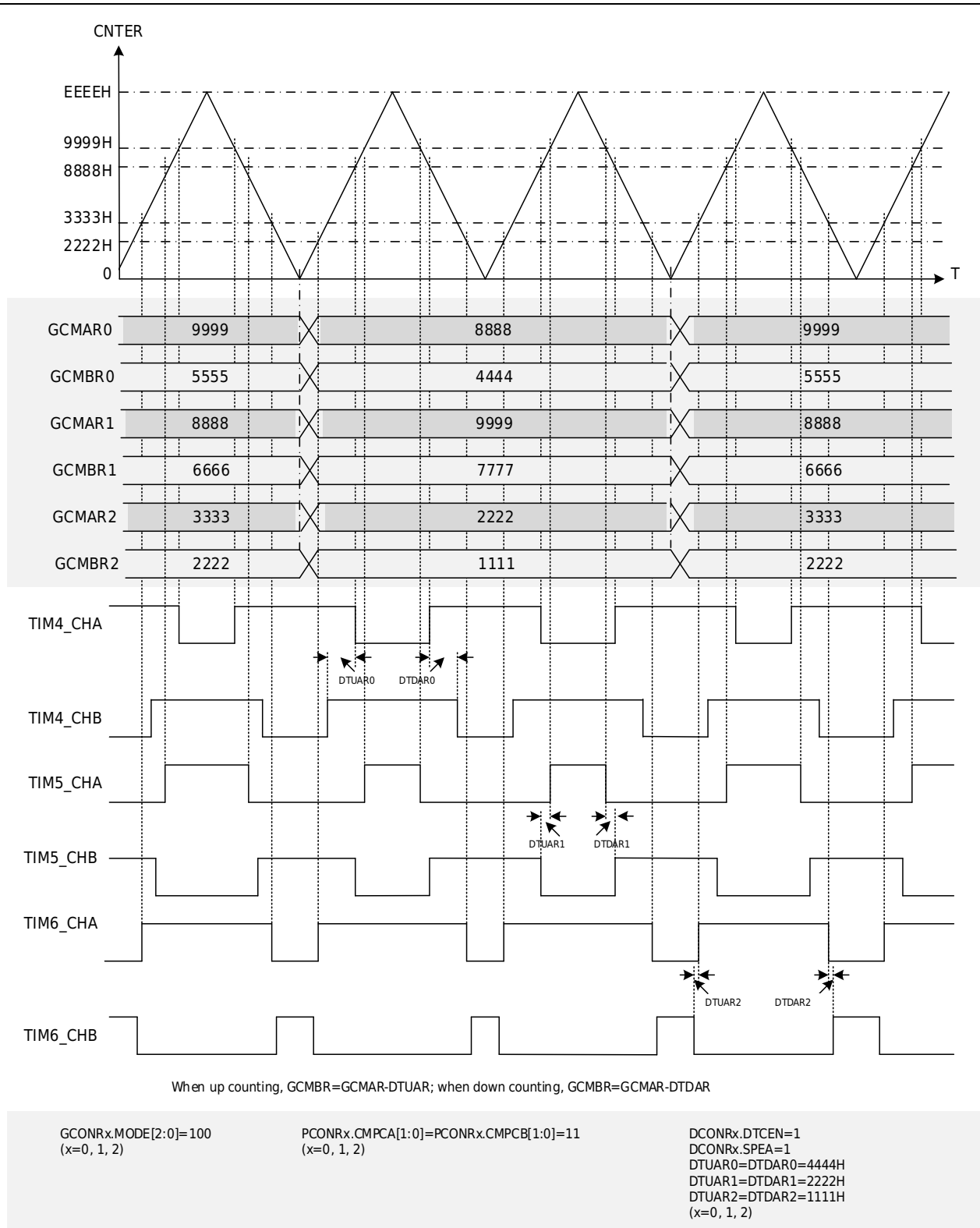


Figure 12-15 Three-phase complementary PWM wave output with dead time in triangular wave A mode

12.2.9 Orthogonal coding count

Treating CHxA input as AIN input, CHxB input as BIN input, and any input in TIMTRIA-D as ZIN input, the Advanced Timer can realize the quadrature encoding count of three inputs.

AIN and BIN of one timer can realize the position counting mode; the combined action of AIN, BIN and ZIN of two timers can realize the revolution counting mode, one timer is used for position counting, and the other timer is used for revolution counting.

In revolution counting mode, every combination of two timers (combination of timers 4 and 5, timer 4 as a position counting unit, timer 5 as a revolution counting unit) realizes position counting and revolution counting respectively.

AIN and BIN are realized by setting the orthogonal relationship between CHxA and CHxB in the hardware increment event selection register (HCUPR) and the hardware decrement event selection register (HCDOR); the input action of ZIN is cleared by setting the hardware of the position unit. The event selection register (HCLRR) realizes clearing the position counter of the position counting unit, and realizes the counting of the revolution counter of the revolution counting unit by setting the hardware increment event selection register (HCUPR) of the revolution unit.

12.2.9.1 Position counting mode

The quadrature encoding position mode refers to the realization of basic counting function, phase difference counting function and direction counting function according to the input of AIN and BIN.

12.2.9.1.1 Basic count

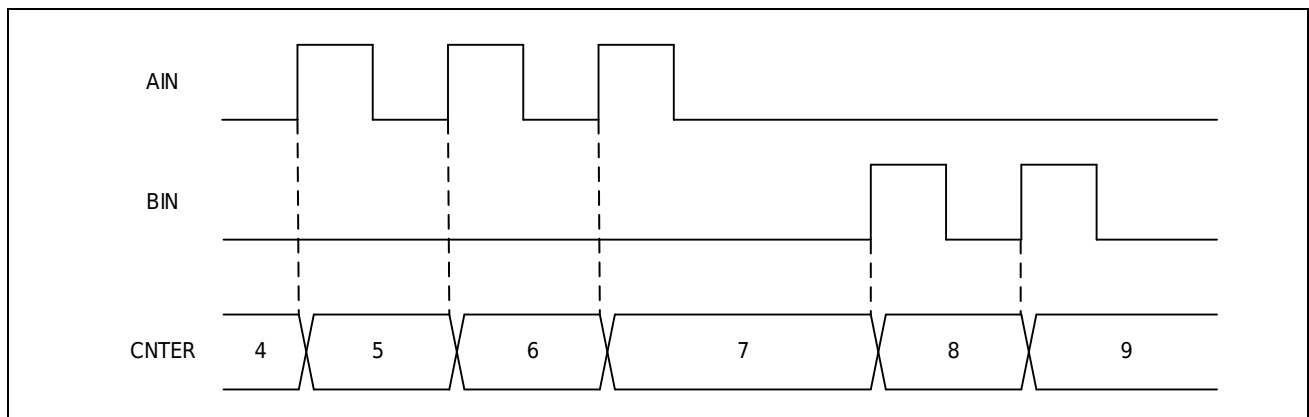


Figure 12-16 Basic counting action in position mode

By setting the HCUPR and HCDOR registers, various ways of phase difference counting can be flexibly realized.

12.2.9.1.2 Phase difference count

Phase difference counts are counted according to the phase relationship between AIN and BIN. According to different settings, 1-fold counting, 2-fold counting, 4-fold counting, etc. can be realized, as shown in Figure 12-17~ Figure 12-19 in the following figure.

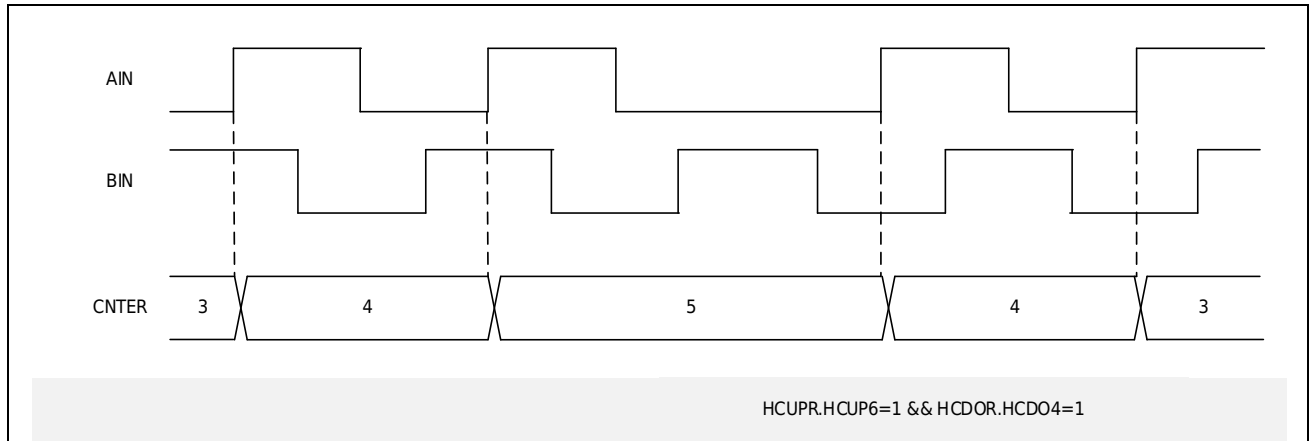


Figure 12-17 Phase difference counting action setting in position mode (1 time)

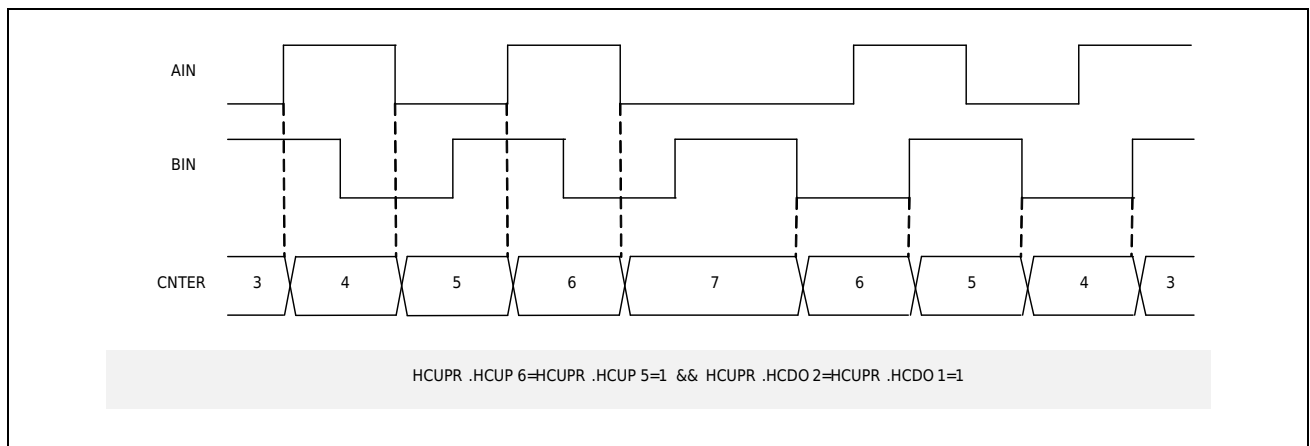


Figure 12-18 Phase difference counting action setting in position mode (2 time)

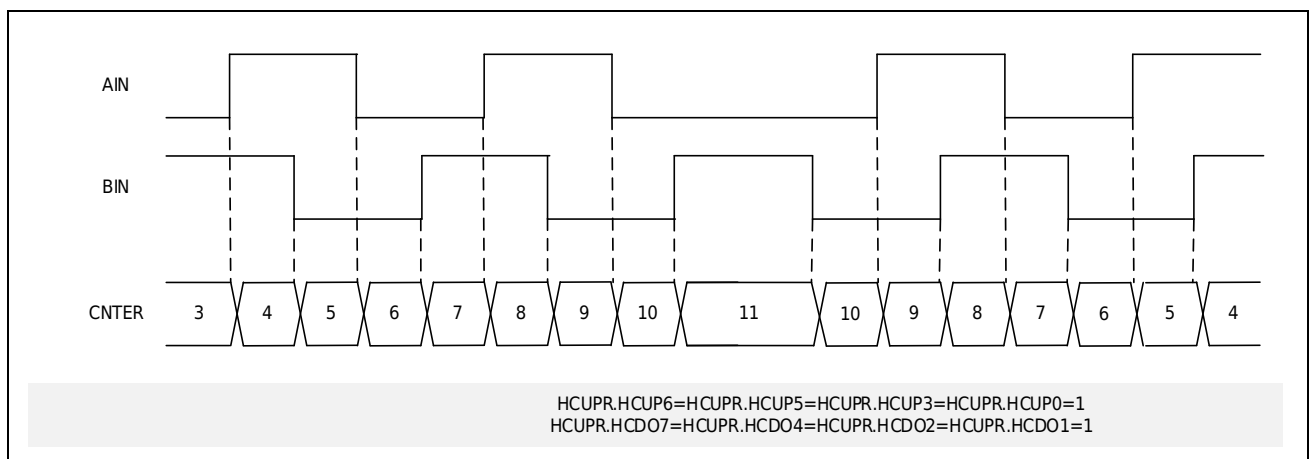


Figure 12-19 Phase difference counting action setting in position mode (4 time)

12.2.9.1.3 Direction count

Direction counting refers to setting the input state of AIN as direction control, and using the input of BIN as clock counting, as shown in Figure 12-20.

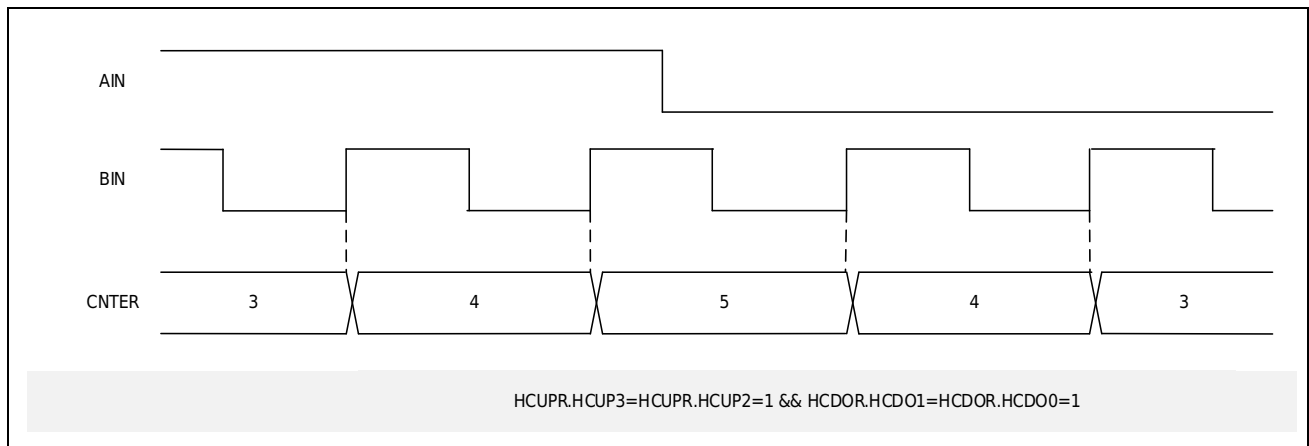


Figure 12-20 Direction counting action in position mode

12.2.9.2 Revolution mode

Orthogonal encoding revolution mode refers to the addition of ZIN input events on the basis of AIN and BIN counts to realize the judgment of the number of revolutions, etc. In the revolution mode, according to the counting method of the revolution counter, the Z phase counting function, the position counter output counting function and the Z phase counting and position counter output mixed counting function can be realized. That is to use two Advanced Timers to realize this function.

12.2.9.2.1 Z phase count

Z-phase counting refers to the counting action that the revolution counting unit counts and the position counting unit is cleared according to the input of ZIN.

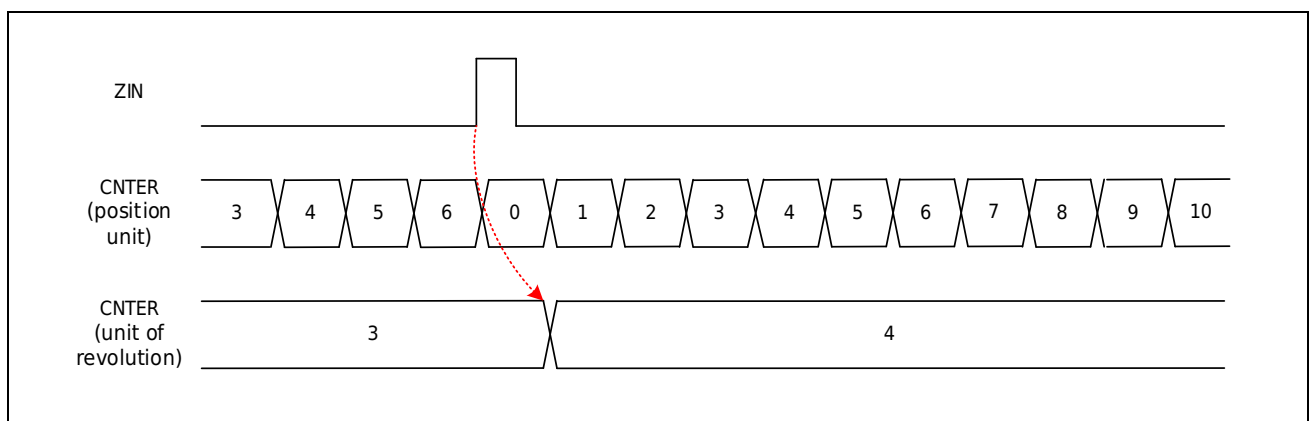


Figure 12-21 Phase Z counting action in revolution mode

12.2.9.2.2 Position overflow count

Position overflow counting means that when the counting of the position counting unit overflows or underflows, an overflow event is generated, thereby triggering the counter of the revolution counting unit to count once (in this counting mode, the input of ZIN does not perform the counting action of the revolution counting unit and clearing action of the position counting unit).

The overflow event of the position counting unit realizes the counting of the revolution counting unit through the linkage gating of the AOS module, and the position overflow counting can be realized. The increment (decrement) event selection register (HCUPR or HCDOR) of the hardware increment (decrement) event selection register (HCUPR or HCDOR) of the revolution counting unit selects 1 bit in Bit16:it19, and the AOS module sets the event source of the corresponding increment (decrement) event. It is the count overflow event of the position counting unit, please refer to the AOS chapter for details. As shown in Figure 12-22.

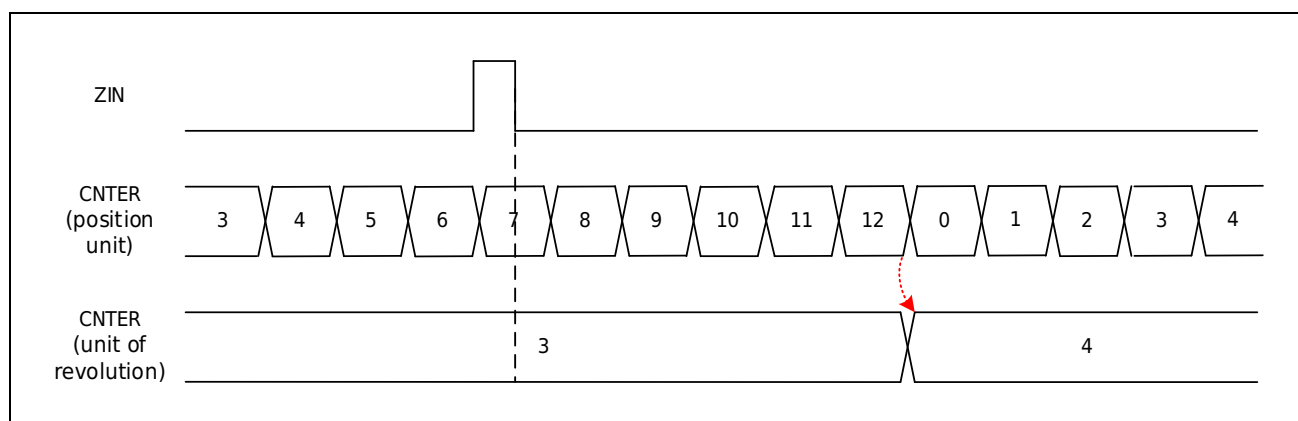


Figure 12-22 Position counter output counting action in revolution mode

12.2.9.2.3 Mixed count

Mixed counting refers to the counting action that combines the above two counting methods of Z-phase counting and position overflow counting, and its realization method is also a combination of the above two counting methods. As shown in Figure 12-23.

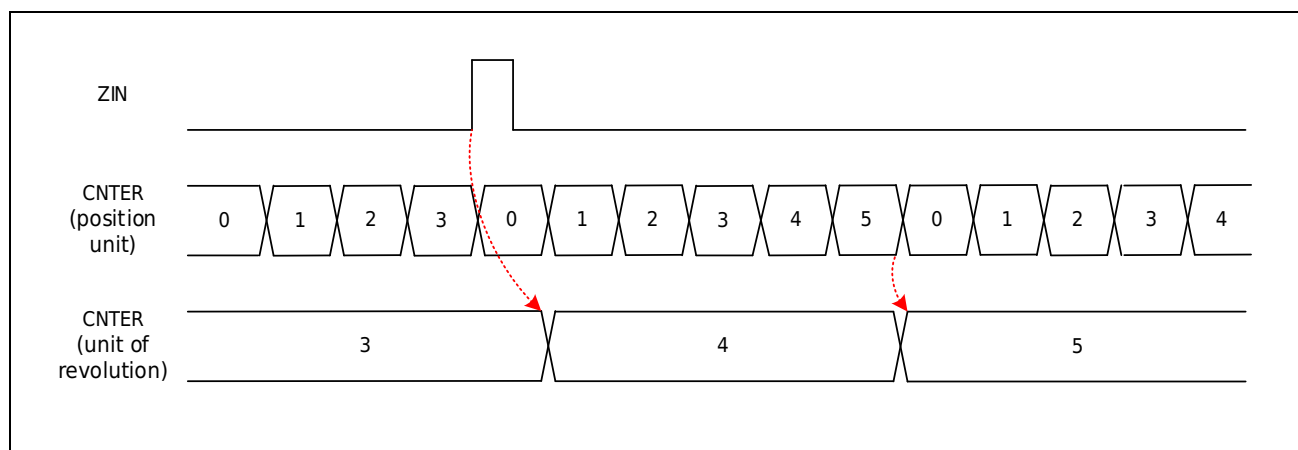


Figure 12-23 Z-phase counting and position counter output mixed counting action in revolution mode

12.2.9.2.4 Z phase motion shielding

In the Z-phase counting function or mixed counting function of the revolution counting mode, it can be set within a few cycles after the overflow point or underflow point of the position counter (GCONR.ZMSK[0:1] setting), ZIN The effective input mask of the input is disabled, and the counting of the revolution counting unit and the clearing of the position counting unit are not performed.

When the GCONR.ZMSKPOS of the general control register (GCONR) of the position counting unit is 1, the Z-phase masking function of the position counting unit is enabled, and the number of cycles of Z-phase masking is set by GCONR.ZMSK; the general control register of the revolution counting unit (When GCONR.ZMSKREV of GCONR) is 1, the Z-phase shielding function of the revolution counting unit is enabled.

Figure 12-24 is when the revolution counting mode is mixed counting, when there is a ZIN phase input within 4 counting cycles after the counting overflow of the position counting unit, the action of the ZIN phase input is invalid, that is, the revolution counting unit does not count, and the position counting unit does not reset ;The subsequent ZIN phase input works normally.

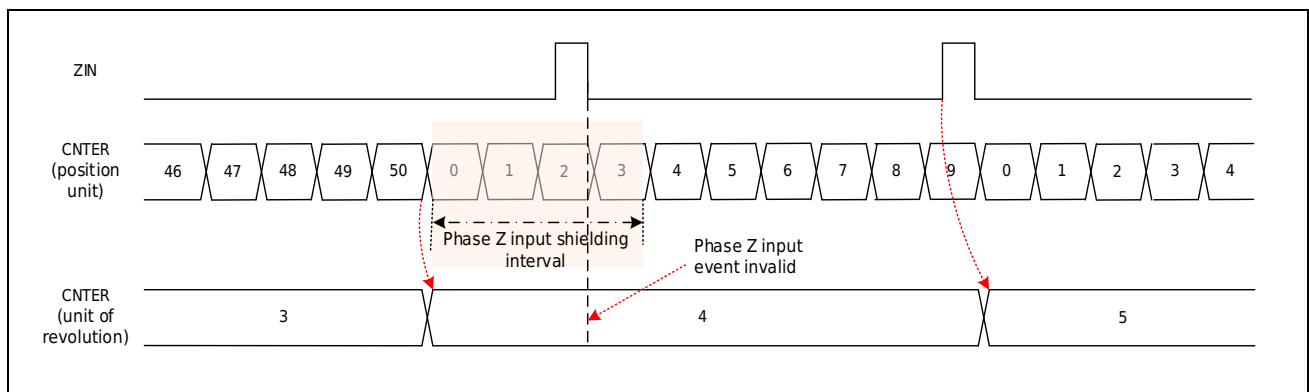


Figure 12-24 Revolution counting mode - Mixed counting Z-phase shielding action example 1

Figure 12-25 shows that when the revolution counting mode is mixed counting, the counting direction changes in the third cycle after the position counting unit overflows, and the masking cycle of the 4 cycles set at this time becomes invalid (the actual ZIN phase masking function remains up to 3 cycles), start counting down. After the counting underflow occurs in the position counting unit, the ZIN phase shielding function is re-opened and becomes invalid after 4 cycles. During the masking period of the ZIN phase, the input function of the ZIN phase is invalid, that is, the revolution counting unit does not count, and the position counting unit does not clear; the subsequent ZIN phase input operates normally.

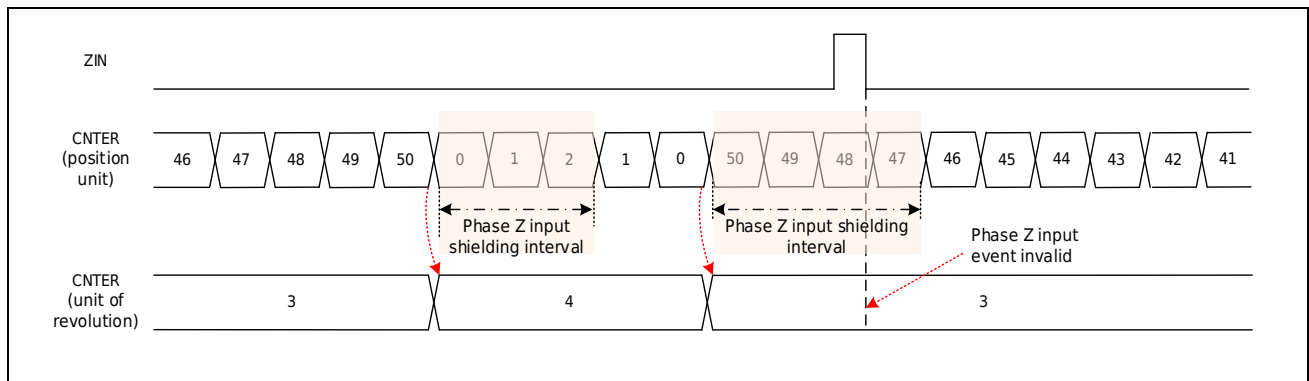


Figure 12-25 Revolution counting mode - Mixed counting Z-phase shielding action example 2

12.2.10 Periodic interval response

The general comparison reference value registers (GCMAR~GCMDR) of Timer4/5/6 can respectively generate special valid request signals when the counting comparison matches, and send them to the AOS module for associated actions with other modules.

The request signal can generate an effective request signal every several cycles. By setting the VPERR.PCNTS bit of the valid period register (VPERR) to specify how many cycles the request signal is valid once, even if the count value is equal to the value of the comparison reference value register GCMAR or GCMBR in other cycles, no valid signal will be output request signal. Figure 12-26 shows an example of the operation of the periodic interval valid request signal.

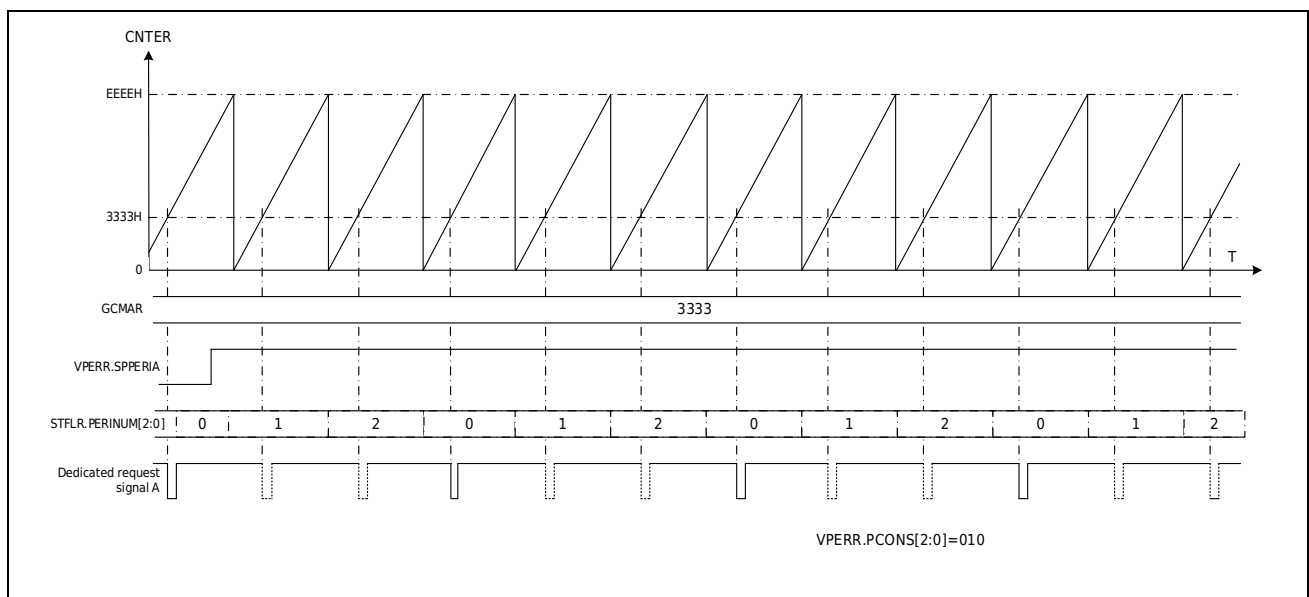


Figure 12-26 Cycle Interval Valid Request Signal Action

12.2.11 Protection mechanism

Advanced Timer can protect and control the output state of the port.

Advanced Timer has 4 shared ports to output invalid event interfaces, these 4 interfaces are connected to 4 groups of brake events output by the brake control module. The abnormal condition

events gated on each interface can be set from the brake control, and when abnormal conditions are detected on these interfaces, the control of the general PWM output can be realized.

During the normal output period of the port, if a braking event from the brake control is detected, the output state of the port can be changed to a preset state. When an abnormal brake control event occurs at the general-purpose PWM output port, the state of the port can change to output high-impedance state, output low level or output high level (determined by the settings of PCONR.DISVALA and PCONR.DISVALB).

For example, if PCONR.DISSELA[1:0]=01&PCONR.DISVALA=01 is set, then during the normal output period of CHxA port, if a brake event occurs on output invalid condition 1, the output of CHxA port will become a high-impedance state.

12.2.12 Interrupt Description

Timer4/5/6 each contain 3 types of 9 interrupts in total. They are 4 common count comparison match interrupts (including 2 capture input interrupts), 2 count cycle match interrupts, and 1 dead zone time error interrupt.

12.2.12.1 Count comparison match interrupt

4 general -purpose comparison reference registers (GCMAR-GCMDR), which can be compared with the count value to generate a comparison match valid signal. When the count compare matches, the STFLR.CMAF~STFLR.CMDF bits in the status flag register (STFLR) will be set to 1 respectively. At this time, if the corresponding bit in ICONR.INTENA~ICONR.INTEND of the interrupt control register (ICONR) is set to 1 to enable the interrupt, the corresponding interrupt request will also be triggered.

The capture input action occurs when the capture input valid condition selected by the hardware capture event selection register (HCPAR, HCPBR) occurs. At this time, if the ICONR.INTENA or ICONR.INTENB bit of the interrupt control register (ICONR) is set to 1 to enable the interrupt, the corresponding interrupt request will be triggered.

12.2.12.2 Count cycle match interrupt

When the sawtooth wave counts up to the overflow point, the sawtooth wave counts down to the underflow point, the triangle wave counts to the valley point or the triangle wave counts to the peak point, the STFLR. 1. At this time, if the ICONR.INTENOVF bit and ICONR.INTENUDF bit of the interrupt control register (ICONR) are set to enable the interrupt, the counting cycle match interrupt can be triggered at the corresponding time point.

12.2.12.3 Dead time error interrupt

the dead time reference register (DTUAR, DTDAR) is loaded into the general comparison reference register (GCMBR), if the cycle limit is exceeded, a dead time error will be generated, and the

STFLR.DTEF of the status flag register (STFLR) bit will be set to 1. At this time, if the ICONR.INTENDE bit of the interrupt control register (ICONR) is set to enable the interrupt, the dead time error interrupt will be triggered at this moment.

12.2.13 Brake protection

When invalid conditions 0~3 can be set, configure PCONR.DISVALA, PCONR.DISVALB. When the invalid condition is valid, the hardware automatically changes the port state to the preset state (high level, low level, high impedance state, and maintains normal output).

12.2.13.1 Port brake and software brake

After the port is controlled by polarity selection and effectively enabled, it is digitally filtered and synchronized to generate a port brake flag; the port brake flag is used as the invalid condition of the Advanced Timer 3. The port brake flag needs to be cleared by software.

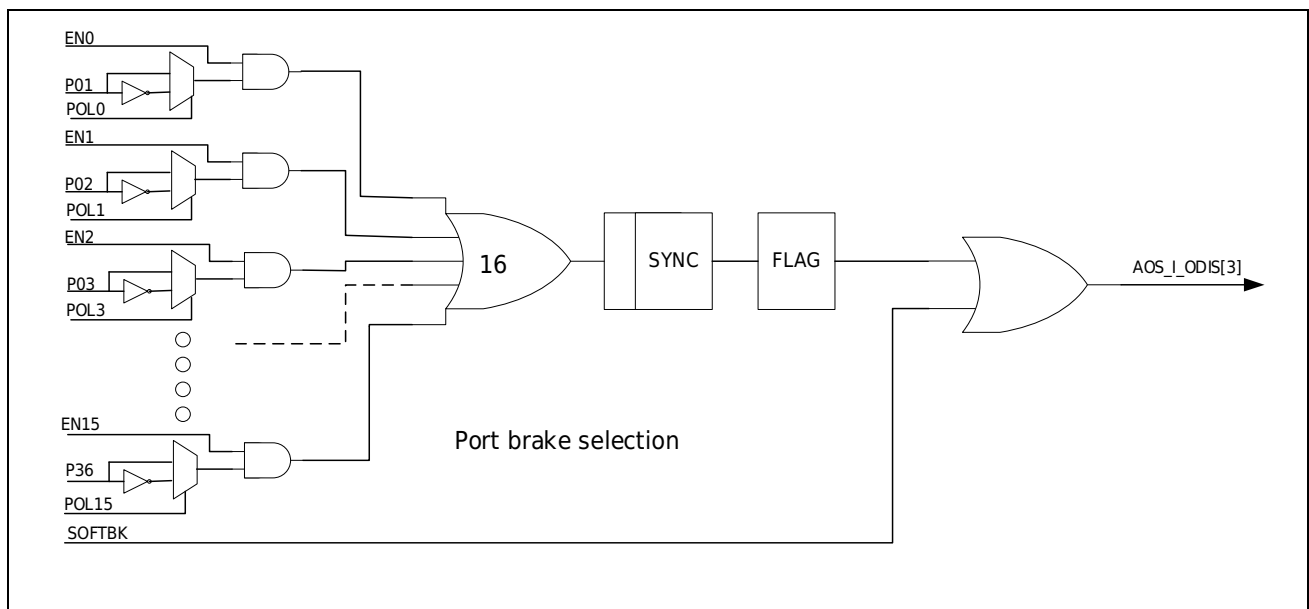


Figure 12-27 Schematic diagram of port braking and software braking

12.2.13.2 Automatic braking in low power mode

The system enters the low power consumption mode, and the PWM will not work normally after the clock stops. Low power mode controls PWM brake as invalid condition 2 of Advanced Timer.

12.2.13.3 Output level same high and same low brake

The output level is monitored by the level, and after it is effectively enabled, it is synchronized to generate the same high and low brake flag; the port brake flag is used as the invalid condition 1 of the Advanced Timer. The same high and same low brake flag needs to be cleared by software.

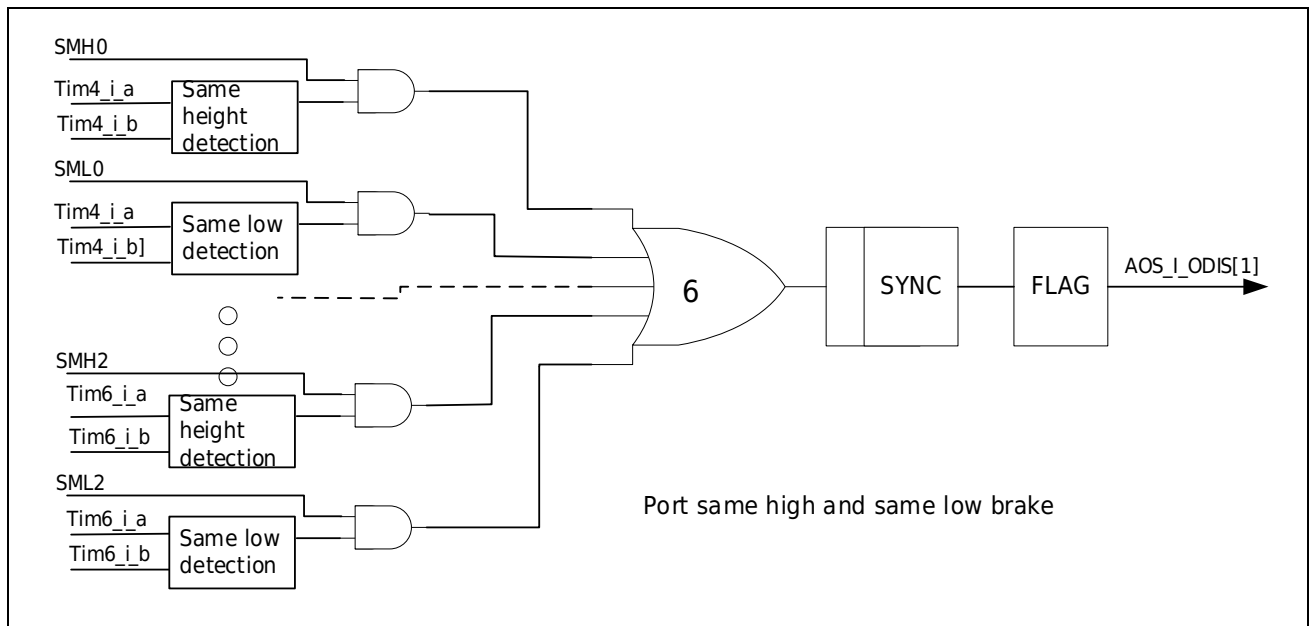


Figure 12-28 Output same high and same low brake schematic diagram

12.2.13.4 VC brake

VC1, VC2 interrupt flags are enabled as the invalid condition 0 of the Advanced Timer.

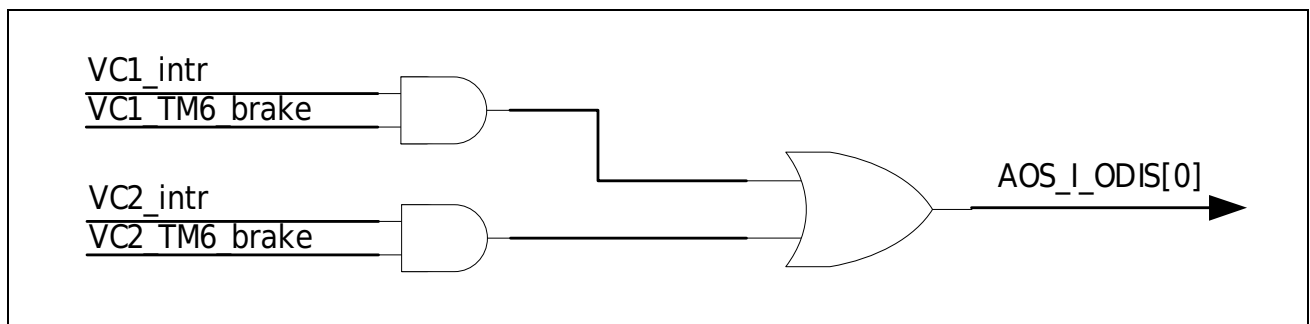


Figure 12-29 VC brake control diagram

12.2.14 Interconnection

12.2.14.1 Interrupt trigger output

Because one interrupt of Timer4/5/6 contains multiple interrupt sources. The interrupt signals for controlling the trigger ADC and controlling the AOS have separate control, and different sources can be selected. You can choose overflow, underflow, and any interrupt source of 4 compare matches with a total of 6 TIMx interrupt sources as the trigger condition.

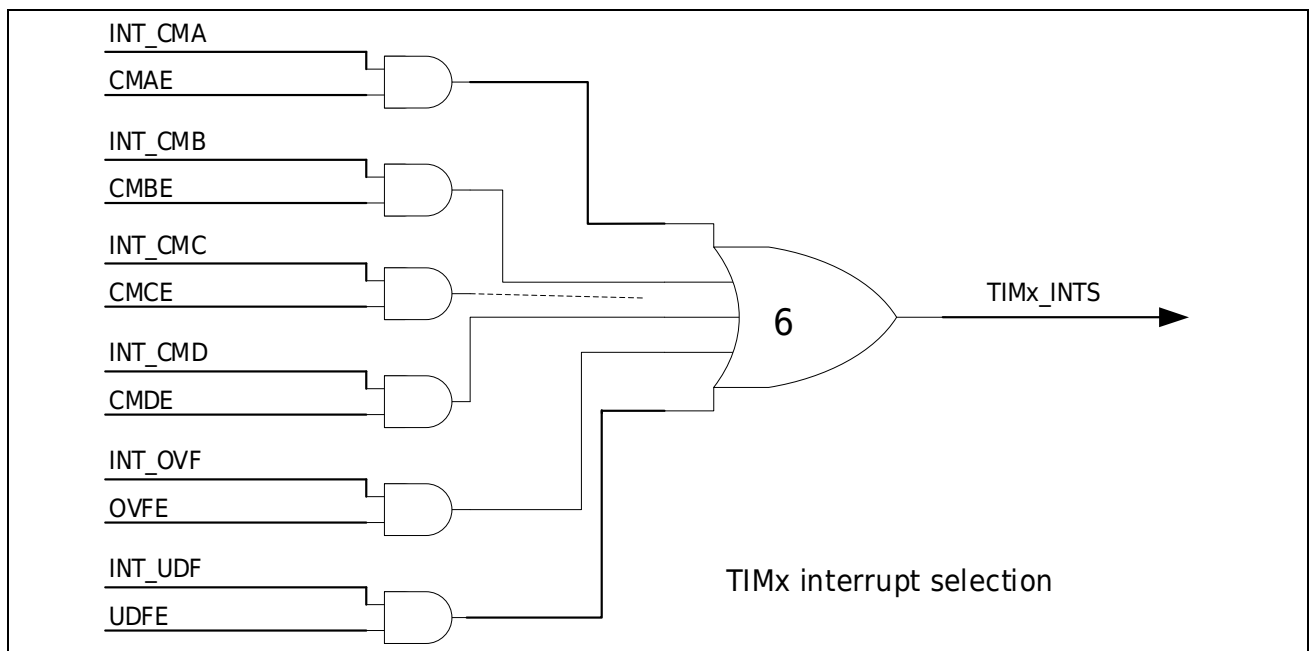


Figure 12-30 Timer4/5/6 interrupt selection

12.2.14.2 AOS trigger

AOS is the internal signal of the system, which can trigger the Advanced Timer 's counter to start, stop, clear, add 1, subtract 1 and other functions after selection control. Advanced Timer has 4 AOS triggers, and each trigger can select the interrupt source of different modules. The selected signal generates a single pulse trigger input to the Advanced Timer to control the start, stop, and clearing of the counter of the Advanced Timer.

Timer4/5/6 internally uses registers to select different AOS_I_TRIG as its own trigger signal. If you can use the HSTAR register, you can use an interrupt to trigger the hardware start of the corresponding timer.

Table 12-3 AOS source selection

	AOS_i_trig0	AOS_i_trig1	AOS_i_trig2	AOS_i_trig3
Select control signal	ITRIG.IAOS0S	ITRIG.IAOS1S	ITRIG.IAOS2S	ITRIG.IAOS3S
0000	TIM0_INT	TIM0_INT	TIM0_INT	TIM0_INT
0001	TIM1_INT	TIM1_INT	TIM1_INT	TIM1_INT
0010	TIM2_INT	TIM2_INT	TIM2_INT	TIM2_INT

	AOS_i_trig0	AOS_i_trig1	AOS_i_trig2	AOS_i_trig3
0011	LPTIMER_INT	LPTIMER_INT	LPTIMER_INT	LPTIMER_INT
0100	TIM4_INTS	TIM4_INTS	TIM4_INTS	TIM4_INTS
0101	TIM5_INTS	TIM5_INTS	TIM5_INTS	TIM5_INTS
0110	TIM6_INTS	TIM6_INTS	TIM6_INTS	TIM6_INTS
0111	UART0_INT	UART0_INT	UART0_INT	UART0_INT
1000	UART1_INT	UART1_INT	UART1_INT	UART1_INT
1001	LPUART_INT	LPUART_INT	LPUART_INT	LPUART_INT
1010	VC1_INT	VC1_INT	VC1_INT	VC1_INT
1011	VC2_INT	VC2_INT	VC2_INT	VC2_INT
1100	RTC_INT	RTC_INT	RTC_INT	RTC_INT
1101	PCA_INT	PCA_INT	PCA_INT	PCA_INT
1110	SPI_INT	SPI_INT	SPI_INT	SPI_INT
1111	ADC_INT	ADC_INT	ADC_INT	ADC_INT

12.2.14.3 Port Trigger TRIGA-TRIGD

The port trigger can control the hardware start, stop, clear, capture, counter plus and minus counting functions of the Advanced Timer, and the digital filter function is optional, and the port can be configured as any port of the chip.

Table 12-4 Port trigger selection

Select control signals to control independently	TRIGA	TRIGB	TRIGC	TRIGD
0000	P01	P01	P01	P01
0001	P02	P02	P02	P02
0010	P03	P03	P03	P03
0011	P15	P15	P15	P15
0100	P14	P14	P14	P14
0101	P23	P23	P23	P23
0110	P24	P24	P24	P24
0111	P25	P25	P25	P25
1000	P26	P26	P26	P26
1001	P27	P27	P27	P27
1010	P31	P31	P31	P31
1011	P32	P32	P32	P32
1100	P33	P33	P33	P33
1101	P34	P34	P34	P34
1110	P35	P35	P35	P35
1111	P36	P36	P36	P36

12.2.14.4 The comparison output VC is interconnected with Advanced Timer

The VC can be interconnected to the capture input of the Advanced Timer, which can capture the edge of the VC output;

12.2.14.5 Interconnection between UART and Advanced Timer

UARTx_RX / LPUART_RX can be connected to BaseTimer, LPTimer, PCA and Advanced Timer through internal interconnection. The automatic identification of baud rate can be realized by software.

The UART selection control register is GPIO_CTRL3 in the port control register, and the VC output control register is in the VC control module.

12.3 Register description

CH0 base address 0x40003000

CH1 base address 0x40003400

CH2 base address 0x40003800

Table 12-5 Advanced Timer register list

Register	Offset address	Description
TIMx_CNTER	0x000	General purpose count reference register
TIMx_PERAR	0x004	General purpose period reference register
TIMx_PERBR	0x008	General purpose cycle reference buffer register
TIMx_GCMAR	0x010	General purpose compare A reference value register
TIMx_GCMBR	0x014	General purpose compare B reference value register
TIMx_GCMCR	0x018	General purpose compare C reference value register
TIMx_GCMDR	0x01C	General purpose compare D reference value register
TIMx_DTUAR	0x040	DEAD TIME REFERENCE REGISTER
TIMx_DTDAR	0x044	DEAD TIME REFERENCE REGISTER
TIMx_GCONR	0x050	General control register
TIMx_ICONR	0x054	Interrupt control register
TIMx_PCONR	0x058	Port control register
TIMx_BCONR	0x05C	Cache control register
TIMx_DCONR	0x060	Dead Band Control Register
TIMx_FCONR	0x068	Filter control register
TIMx_VPERR	0x06C	Valid period register
TIMx_STFLR	0x070	Status flag register
TIMx_HSTAR	0x074	Hardware Boot Event Select Register
TIMx_HSTPR	0x078	Hardware Stop Event Select Register
TIMx_HCELR	0x07C	Hardware clear event select register
TIMx_HCPAR	0x080	Hardware Capture Event Select Register
TIMx_HCPBR	0x084	Hardware Capture Event Select Register
TIMx_HCUPR	0x088	Hardware Decrease Event Selection Register
TIMx_HCDOR	0x08C	Hardware Decrease Event Selection Register
TIMx_IFR	0x100	Interrupt Flag Register
TIMx_ICLR	0x104	Interrupt Clear Register
TIMx_CR	0x108	Spread spectrum and interrupt trigger selection register
TIMx_AOSSR	0x110	AOS selection register, shared by three channels
TIMx_AOSCL	0x114	AOS brake flag clear register, shared by three channels
TIMx_PTBKS	0x118	Port brake control register, shared by three channels
TIMx_TTRIG	0x11C	Port trigger control register, shared by three channels

Register	Offset address	Description
TIMx_ITRIG	0x120	AOS trigger control register, shared by three channels
TIMx_PTBKP	0x124	Port brake polarity control register, shared by three channels
TIMx_SSTAR	0x3F4	Software Synchronization Enable Register
TIMx_SSTPR	0x3F8	Software Synchronization Stop Register
TIMx_SCLRR	0x3FC	Software synchronous clear register

12.3.1 Common Count Reference Register (TIMx_CNTER)

Address offset: 0x000

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
R/W															

Bit	Symbol	Functional description
31:16	Reserved	-
15:0	CNT[15:0]	The count value of the current counter

12.3.2 Universal Period Reference Register (TIMx_PERAR)

Address offset: 0x004

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PERA[15:0]															
R/W															

Bit	Symbol	Functional description
31:16	Reserved	-
15:0	PERA[15:0]	Counting period value, set the counting period value of each round of counting

12.3.3 General purpose period buffer register (TIMx_PERBR)

Address offset: 0x008

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PERB[15:0]															
R/W															

Bit	Symbol	Functional description
31:16	Reserved	-
15:0	PERB[15:0]	Buffer count cycle value, cache value of count cycle

12.3.4 Universal Compare Base Registers (TIMx_GCMAR-GCMDR)

Address offset: 0x0010, 0x0014, 0x0018, 0x001C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GCMA-D [15:0]															
R/W															

Bit	Symbol	Functional description
31:16	Reserved	-
15:0	GCMA-D [15:0]	Counting comparison reference value, comparison reference value setting, matching signal is valid when it is equal to the count value

12.3.5 DEAD TIME REFERENCE REGISTER (TIMx_DTUAR-DTDAR)

Address offset: 0x040, 0x044

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DTUA/DTDA [15:0]															
R/W															

Bit	Symbol	Functional description
31:16	Reserved	-
15:0	DTUA/DA [15:0]	Dead time value, dead time set value

12.3.6 General Control Register (TIMx_GCONR)

Address offset: 0x050

Reset value: 0x00000100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												ZMSK	ZMSK POS	ZMSK REV	
												R/W	R/W	R/W	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DIR	Res.	CKDIV			MODE		START	
							R/W		R/W			R/W		R/W	

Bit	Marking	Functional description
31:20	Reserved	-
19:18	ZMSK	Phase Z input muting cycle number Quadrature encoding phase Z input masked count period value 00: Phase Z input shielding function is invalid 01: The Z-phase input is masked within 4 counting cycles after the position count overflows or underflows 10: The Z-phase input is masked within 8 count cycles after the position count overflows or underflows 11: The Z-phase input is masked within 16 count cycles after the position count overflows or underflows
17	ZMSKPOS	Z phase input position counter selection 0: When the Z phase is input, the timer is used as a position counter, and the position counter clearing function works normally during the masking period 1: When the Z phase is input, the timer is used as a position counter, and the function of clearing the position counter is masked during the masking period
16	ZMSKREV	Z phase input revolution counter selection 0: When the Z phase is input, the timer is used as a revolution counter, and the revolution counter counting function works normally during the shielding period 1: When the Z phase is input, the timer is used as a revolution counter, and the counting function of the revolution counter is shielded during the shielding period
15:9	Reserved	-
8	DIR	Counting direction 0: count down; 1: count up
7	Reserved	-
6:4	CKDIV	Counting clock selection 000: PCLK0 001: PCLK0/2 010: PCLK0/4 011: PCLK0/8 100: PCLK0/16 101: PCLK0/64 110: PCLK0/256 111: PCLK0/1024
3:1	MODE	Counting mode 000: sawtooth wave A mode 100: triangular wave A mode 101: triangular wave B mode Please do not set other values
0	START	Counter start 0: counter off; 1: counter start <i>Note: This bit will automatically become 0 when the software stop condition or hardware stop condition is valid</i>

12.3.7 Interrupt Control Register (TIMx_ICONR)

Address offset: 0x054

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTEN DE	INTEN UDF	INTEN OVF	Reserved		INTEN D	INTEN C	INTEN B	INTEN A
							R/W	R/W	R/W			R/W	R/W	R/W	R/W

Bit	Marking	Function
31:9	Reserved	-
8	INTENDE	Dead time error interrupt enable 0: When the dead time is wrong, the interrupt is invalid 1: When the dead time is wrong, the interrupt is enabled
7	INTENUDF	Underflow interrupt enable 0: When an overflow occurs during a sawtooth wave or counts to the valley point during a triangular wave, the interrupt is invalid 1: When an underflow occurs during a sawtooth waveform or counts to a valley point during a triangular waveform, the interrupt is enabled
6	INTENOVF	Overflow interrupt enable 0: When an overflow occurs during a sawtooth wave or counts to the peak point during a triangular wave, the interrupt is invalid 1: When an overflow occurs in a sawtooth wave or counts to the peak point in a triangular wave, the interrupt is enabled
5:4	Reserved	-
3	INTEND	Count match interrupt enable D 0: When the GCMDR register is equal to the count value, the interrupt is invalid 1: When the GCMDR register is equal to the count value, the interrupt is enabled
2	INTENC	Count match interrupt enable C 0: When the GCMCR register is equal to the count value, the interrupt is invalid 1: When the GCMCR register is equal to the count value, the interrupt is enabled
1	INTENB	Count match interrupt enable B 0: When the GCMBR register is equal to the count value, or when a capture input event occurs, the interrupt is invalid 1: The interrupt is enabled when the GCMBR register is equal to the count value, or when a capture input event occurs
0	INTENA	Count Match interrupt enable A 0: When the GCMAR register is equal to the count value, or when a capture input event occurs, the interrupt is invalid 1: The interrupt is enabled when the GCMAR register is equal to the count value, or when a capture input event occurs

12.3.8 Port Control Register (TIMx_PCONR)

Address offset: 0x058

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			DISVALB	DISSELB		OUTENB		PERCB		CMPCB		STASTPS B	STPC B	STAC B	CAPC B
			R/W	R/W		R/W		R/W		R/W		R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DISVALA	DISSELA		OUTENA		PERCA		CMPCA		STASTPS A	STPC A	STAC A	CAPC A
			R/W	R/W		R/W		R/W		R/W		R/W	R/W	R/W	R/W

Bit	Marking	Function
31:29	Reserved	-
28:27	DISVALB	CHxB output state control 00: When the condition selected among the forced output invalid conditions 0~3 is met, the CHxB port outputs normally 01: When the selected condition among the forced output invalid conditions 0~3 is met, the CHxB port outputs a high-impedance state 10: When the forced output invalid condition 0~3 is selected, the CHxB port outputs a low level 11: When the selected condition among the forced output invalid conditions 0~3 is met, the CHxB port outputs a high level
26:25	DISSELB	Force output invalid condition selection B 00: select forced output invalid condition 0; 01: select forced output invalid condition 1 10: Select forced output invalid condition 2; 11: Select forced output invalid condition 3
24	OUTENB	Output enable B 0: CHxB port output is invalid when the Advanced Timer function is enabled 1: The CHxB port output is valid when the Advanced Timer function is active
23:22	PERCB	Port status setting when period values matchB 00: When the count value of the counter is equal to the period value, the CHxB port output remains low 01: When the count value of the counter is equal to the period value, the CHxB port output is set to high level 10: When the counter count value is equal to the period value, the CHxB port output is set to the previous state 11: When the count value of the counter is equal to the period value, the CHxB port output is set to the inversion level
21:20	CMPCB	Port status setting when comparison values matchB 00: When the counter count value is equal to GCMBR, the output of CHxB port remains low 01: When the counter count value is equal to GCMBR, the CHxB port output is set to high level 10: When the counter count value is equal to GCMBR, the CHxB port output is set to the previous state 11: When the count value of the counter is equal to GCMBR, the CHxB port output is set to the inverted level
19	STASTPSB	Count start stop port state selection B 0: When counting starts or stops, CHxB port output is determined by STACB, STPCB 1: When counting starts or stops, the CHxB port output is set to the previous state <i>Note: The counting start here refers to the initial counting start or stop and restart; the counting stop refers to the initial stop or counting start and then stop</i>
18	STPCB	Count stop port status setting B 0: When counting stops, CHxB port output is set to low level 1: When counting stops, CHxB port output is set to high level
17	STACB	Count start port status setting B 0: When counting starts, CHxB port output is set to low level 1: When counting starts, CHxB port output is set to high level
16	CAPCB	Function mode selection B 0: comparison output function; 1: capture input function
15:13	Reserved	-
12:11	DISVALA	CHxA output state control 00: When the condition selected among the forced output invalid conditions 0~3 is met, the CHxA port outputs normally 01: When the condition selected in the forced output invalid condition 0~3 is met, the CHxA port outputs a high-impedance state 10: When the condition selected in the forced output invalid condition 0~3 is met, the CHxA port outputs a low level 11: When the selected condition among the forced output invalid conditions 0~3 is met, the CHxA port

		outputs a high level
10:9	DISSELA	Force output invalid condition selection A 00: select forced output invalid condition 0; 01: select forced output invalid condition 1 10: Select forced output invalid condition 2; 11: Select forced output invalid condition 3
8	OUTENA	Output enable A 0: CHxA port output is invalid when the Advanced Timer function is active 1: CHxA port output is valid when the Advanced Timer function is active
7:6	PERCA	Port status setting when period values matchA 00: When the count value of the counter is equal to the period value, the CHxA port output remains low 01: When the count value of the counter is equal to the period value, the CHxA port output is set to high level 10: When the counter count value is equal to the period value, the CHxA port output is set to the previous state 11: When the count value of the counter is equal to the period value, the output of the CHxA port is set as an inversion level
5:4	CMPCA	Port State Setting A When Comparing Values Match 00: When the counter count value is equal to GCMAR, the CHxA port output remains low 01: When the counter count value is equal to GCMAR, the CHxA port output is set to high level 10: When the counter count value is equal to GCMAR, the CHxA port output is set to the previous state 11: When the count value of the counter is equal to GCMAR, the CHxA port output is set to the inverted level
3	STASTPSA	Count start stop port state selection A 0: When counting starts or stops, CHxA port output is determined by STACA, STPCA 1: When counting starts or stops, the CHxA port output is set to the previous state <i>Note: The counting start here refers to the initial counting start or stop and restart; the counting stop refers to the initial stop or counting start and then stop</i>
2	STPCA	Count stop port state setting A 0: When counting stops, the CHxA port output is set to low level; 1: When counting stops, CHxA port output is set to high level
1	STACA	Count start port status setting A 0: When counting starts, CHxA port output is set to low level 1: When counting starts, CHxA port output is set to high level
0	CAPCA	Functional mode selection A 0: comparison output function; 1: capture input function

12.3.9 Buffer Control Register (TIMx_BCONR)

Address offset: 0x05C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							BENP	Reserved					BENB	Res.	BENA
							R/W						R/W		R/W

Bit	Marking	Function
31:9	Reserved	-
8	BENP	Period value buffer transfer 0: Buffer transmission is invalid 1: Buffer transfer enable (PERBR->PERAR)
7:3	Reserved	-
2	BENB	General comparison value buffer transfer B 0: Buffer transmission is invalid 1: Buffer transfer enable When comparing output functions: (GCMDB->GCMDBR); when capturing input functions: (GCMDBR->GCMDBR)
1	Reserved	-
0	BENA	General comparison value buffer transfer A 0: Buffer transmission is invalid 1: Buffer transfer enable When comparing output functions: (GCMDBR->GCMDBR); when capturing input functions: (GCMDBR->GCMDBR)

12.3.10 Dead Time Control Register (TIMx_DCONR)

Address offset: 0x060

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							SEPA	Reserved						DTCEN	
							R/W							R/W	

Bit	Marking	Function
31:9	Reserved	-
8	SEPA	Separate settings 0: DTUAR and DTDAR are set separately 1: The value of DTDAR is automatically equal to the value of DTUAR
7:1	Reserved	-
0	DTCEN	Dead zone function 0: Dead zone function is invalid 1: The dead zone function is valid

12.3.11 Filter Control Register (TIMx_FCONR)

Address offset: 0x068

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	NOFICKTD		NOFI ENTD	Res.	NOFICKTC		NOFI ENTC	Res.	NOFICKTB		NOFI ENTB	Res.	NOFICKTA		NOFI ENTA
	R/W		R/W		R/W		R/W		R/W		R/W		R/W		R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									NOFICKGB	NOFI ENGB	Res.	NOFICKGA	NOFI ENGA		
									R/W	R/W		R/W	R/W		

Bit	Marking	Function
31	Reserved	-
30:29	NOFICKTD[1:0]	TRID port filter sampling reference clock selection 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
28	NOFIENDT	TRID port capture input filtering enable, 0 is invalid; 1 is enabled
27	Reserved	-
26:25	NOFICKTC[1:0]	TRIC port filter sampling reference clock selection 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
24	NOFIENTC	TRIC port capture input filtering enable, 0 is invalid; 1 is enabled
23	Reserved	-
22:21	NOFICKTB[1:0]	TRIB port filter sampling reference clock selection 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
20	NOFIENB	TRIB port capture input filtering enable, 0 is invalid; 1 is enabled
19	Reserved	-
18:17	NOFICKTA[1:0]	TRIA port filter sampling reference clock selection 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
16	NOFIENTA	TRIA port capture input filtering enable, 0 is invalid; 1 is enabled
15:7	Reserved	-
6:5	NOFICKGB[1:0]	CHxIB port filter sampling reference clock selection 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
4	NOFIENB	CHxIB port capture input filter enable, 0 is invalid; 1 is enabled
3	Reserved	-
2:1	NOFICKGA[1:0]	CHxIA port filter sampling reference clock selection 00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64
0	NOFIENGA	CHxIA port capture input filtering enable, 0 is invalid; 1 is enabled

Note:

- TRIGA-D filter setting is only valid in TIM4, and invalid in Timer5/6.

12.3.12 Valid Period Register (TIMx_VPERR)

Address offset: 0x06C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											PCNTS		PCNTE		
											R/W		R/W		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												GE PERID	GE PERIC	GE PERIB	GE PERIA
												R/W	R/W	R/W	R/W

Bit	Marking	Function
31:21	Reserved	-
20:18	PCNTS	Valid period selection 000: Valid cycle selection function is invalid 001: Valid every 1 cycle 010: Valid every 2 cycles 011: Valid every 3 cycle 100: Valid every 4 cycles 101: Valid every 5 cycle 110: Valid every 6 cycles 111: Valid every 7 cycle
17:16	PCNTE	Active cycle count condition selection 00: Valid period selection function is invalid 01: The sawtooth wave counts the upper and lower overflow points or the triangular wave trough as the counting condition 10: The sawtooth wave counts the upper and lower overflow points or the triangular wave peak as the counting condition 11: The sawtooth wave counts the upper and lower overflow points or the triangular wave trough and peak as the counting condition
15:4	Reserved	-
3	GEPERID	General signal effective period selection D 0: Valid period selection function is invalid; 1: Valid period selection function is enabled
2	GEPERIC	General signal effective period selection C 0: Valid period selection function is invalid; 1: Valid period selection function is enabled
1	GEPERIB	General signal effective period selection B 0: Valid period selection function is invalid; 1: Valid period selection function is enabled
0	GEPERIA	General signal effective period selection A 0: Valid period selection function is invalid; 1: Valid period selection function is enabled

12.3.13 Status Flag Register (TIMx_STFLR)

Address offset: 0x070

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
DIRF	Reserved						VPERNUM		Reserved					
R							R							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DTEF	UDFF	OVFF	Reserved		CMDF	CMCF	CMBF	CMAF
							R/W	R/W	R/W			R/W	R/W	R/W	R/W

Bit	Marking	Function
31	DIRF	Counting direction 0: count down 1: count up
30:24	Reserved	-
23:21	VPERNUM	Number of cycles When the effective cycle selection function is enabled, the number of cycles after counting
20:9	Reserved	-
8	DTEF	Dead time error 0: no dead time error occurred; 1: dead time error occurred
7	UDFF	Underflow match 0: Sawtooth underflow does not occur or triangle wave counts to the valley point 1: A sawtooth wave underflow occurs or a triangular wave counts to a valley point
6	OVFF	Overflow match 0: No sawtooth overflow or triangular wave counting to peak 1: Sawtooth overflow occurs or triangular wave counts to peak
5:4	Reserved	-
3	CMDF	Count match D 0: The value of the GCMR register is not equal to the count value; 1: The value of the GCMR register is equal to the count value
2	CMCF	Count match C 0: The value of the GCMR register is not equal to the count value; 1: The value of the GCMR register is equal to the count value
1	CMBF	Count match B 0: The value of the GCMR register is not equal to the count value, and the CHxB capture completion action has not occurred 1: The value of the GCMR register is equal to the count value, or a CHxB capture complete action occurs
0	CMAF	Count Match A 0: The value of the GCMR register is not equal to the count value, and the CHxA capture completion action has not occurred 1: The value of the GCMR register is equal to the count value, or a CHxA capture complete action occurs

12.3.14 Hardware Start Event Select Register (TIMx_HSTAR)

Address offset: 0x074

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STARTS	Reserved														
R/W															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSTA15	HSTA14	HSTA13	HSTA12	HSTA11	HSTA10	HSTA9	HSTA8	HSTA7	HSTA6	HSTA5	HSTA4	HSTA3	HSTA2	HSTA1	HSTA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Function
31	STARTS	Hardware enable 0: hardware startup is invalid 1: Hardware startup is valid <i>Note: When the hardware boot is valid, the setting of SSTAR is invalid</i>
30:16	Reserved	-
15	HSTA15	Hardware Start Condition 15: Sampled on falling edge on TIMTRID port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
14	HSTA14	Hardware Start Condition 14: Rising edge sampled on TIMTRID port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
13	HSTA13	Hardware Start Condition 13: Sampled on falling edge on TIMTRIC port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
12	HSTA12	Hardware Start Condition 12: Rising edge sampled on TIMTRIC port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
11	HSTA11	Hardware Start Condition 11: Sampled on falling edge on TIMTRIB port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
10	HSTA10	Hardware Start Condition 10: Rising edge sampled on TIMTRIB port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
9	HSTA9	Hardware start condition 9: TIMTRIA port sampled to falling edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
8	HSTA8	Hardware start condition 8: Sampling to rising edge on TIMTRIA port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
7	HSTA7	Hardware start condition 7: CHxB port sampled to falling edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
6	HSTA6	Hardware start condition 6: Sampled on CHxB port to rising edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
5	HSTA5	Hardware start condition 5: CHxA port sampled to falling edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
4	HSTA4	Hardware start condition 4: CHxA port sampled to rising edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
3	HSTA3	Hardware start condition 3: Event trigger 3 from AOS is valid 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
2	HSTA2	Hardware start condition 2: Event trigger 2 from AOS is valid 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
1	HSTA1	Hardware start condition 1: Event trigger 1 from AOS is valid 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
0	HSTA0	Hardware start condition 0: Event trigger 0 from AOS is valid 0: Hardware startup is invalid when conditions match

		1: Hardware startup is valid when conditions match
--	--	--

12.3.15 Hardware Stop Event Select Register (TIMx_HSTPR)

Address offset: 0x078

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STOPS	Reserved														
R/W															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSTP15	HSTA14	HSTP13	HSTP12	HSTP11	HSTP10	HSTP9	HSTP8	HSTP7	HSTP6	HSTP5	HSTP4	HSTP3	HSTP2	HSTP1	HSTP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Function
31	STOPS	Hardware disable 0: hardware stop is invalid 1: Hardware stop is valid <i>Note: When the hardware stop is valid, the software stop setting is invalid</i>
30:16	Reserved	-
15	HSTP15	Hardware Stop Condition 15: Falling edge sampled on TIMTRID port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
14	HSTP14	Hardware Stop Condition 14: Rising edge sampled on TIMTRID port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
13	HSTP13	Hardware Stop Condition 13: Falling edge sampled on TIMTRIC port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
12	HSTP12	Hardware Stop Condition 12: Rising edge sampled on TIMTRIC port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
11	HSTP11	Hardware Stop Condition 11: Falling edge sampled on TIMTRIB port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
10	HSTP10	Hardware Stop Condition 10: Rising edge sampled on TIMTRIB port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
9	HSTP9	Hardware Stop Condition 9: Falling edge sampled on TIMTRIA port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
8	HSTP8	Hardware Stop Condition 8: Rising edge sampled on TIMTRIA port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
7	HSTP7	Hardware Stop Condition 7: Sampled on CHxB port to falling edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
6	HSTP6	Hardware Stop Condition 6: Sampled to rising edge on CHxB port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
5	HSTP5	Hardware stop condition 5: CHxA port sampled to falling edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
4	HSTP4	Hardware Stop Condition 4: Sampled to rising edge on CHxA port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
3	HSTP3	Hardware stop condition 3: Event trigger 3 from AOS is valid 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
2	HSTP2	Hardware stop condition 2: Event trigger 2 from AOS is valid 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
1	HSTP1	Hardware stop condition 1: Event trigger 1 from AOS is valid 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
0	HSTP0	Hardware stop condition 0: Event trigger 0 from AOS is valid 0: Hardware stop invalid when condition matches

		1: Hardware stops working when conditions match
--	--	---

12.3.16 Hardware Clear Event Select Register (TIMx_HCELR)

Address offset: 0x07C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLEARs	Reserved														
R/W															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCELR15	HCELR14	HCELR13	HCELR12	HCELR11	HCELR10	HCELR9	HCELR8	HCELR7	HCELR6	HCELR5	HCELR4	HCELR3	HCELR2	HCELR1	HCELR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Function
31	STARTS	Hardware clear enable 0: Hardware reset is invalid 1: Hardware clear is valid <i>Note: When the hardware clear is valid, the setting of software clear is invalid</i>
30:16	Reserved	-
15	HCELR15	Hardware clear Condition 15: Sampled on falling edge on TIMTRID port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
14	HCELR14	Hardware clear Condition 14: Sampled on rising edge on TIMTRID port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
13	HCELR13	Hardware clear Condition 13: Sampled on falling edge on TIMTRIC port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
12	HCELR12	Hardware clear condition 12: Sampled on rising edge on TIMTRIC port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
11	HCELR11	Hardware clear condition 11: TIMTRIB port sampled on falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
10	HCELR10	Hardware clear condition 10: sampled on rising edge on TIMTRIB port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
9	HCELR9	Hardware clear condition 9: TIMTRIA port sampled to falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
8	HCELR8	Hardware clear condition 8: sampled on rising edge on TIMTRIA port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
7	HCELR7	Hardware clear condition 7: CHxB port sampled to falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
6	HCELR6	Hardware clear condition 6: CHxB port sampled to rising edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
5	HCELR5	Hardware clear condition 5: CHxA port sampled to falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
4	HCELR4	Hardware clear condition 4: CHxA port sampled to rising edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
3	HCELR3	Hardware clear condition 3: Event trigger 3 from AOS is valid 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
2	HCELR2	Hardware clear condition 2: Event trigger 2 from AOS is valid 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
1	HCELR1	Hardware clear condition 1: Event trigger 1 from AOS is valid 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
0	HCELR0	Hardware clear condition 0: Event trigger 0 from AOS is valid 0: Hardware zeroing is invalid when conditions match

		1: Hardware zeroing is valid when conditions match
--	--	--

12.3.17 Hardware Capture A Event Select Register (TIMx_HCPAR)

Address offset: 0x080

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCPA 15	HCPA 14	HCPA 13	HCPA 12	HCPA 11	HCPA 10	HCPA 9	HCPA 8	HCPA 7	HCPA 6	HCPA 5	HCPA 4	HCPA 3	HCPA 2	HCPA 1	HCPA 0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Function
31:16	Reserved	-
15	HCPA15	Hardware Capture A Condition 15: Sampled to falling edge on TIMTRID port 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
14	HCPA14	Hardware Capture A Condition 14: Sample to rising edge on TIMTRID port 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
13	HCPA13	Hardware Capture A Condition 13: Sample to falling edge on TIMTRIC port 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
12	HCPA12	Hardware Capture A Condition 12: Sample to rising edge on TIMTRIC port 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
11	HCPA11	Hardware Capture A Condition 11: Sampled to falling edge on TIMTRIB port 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
10	HCPA10	Hardware Capture A Condition 10: Sampling on TIMTRIB port to rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
9	HCPA9	Hardware capture A condition 9: TIMTRIA port sampled to falling edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
8	HCPA8	Hardware Capture A Condition 8: Sampling on TIMTRIA port to rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
7	HCPA7	Hardware Capture A Condition 7: Sampled on CHxB port to falling edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
6	HCPA6	Hardware Capture A Condition 6: Sampled on CHxB port to rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
5	HCPA5	Hardware Capture A Condition 5: Sampled on CHxA port to falling edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
4	HCPA4	Hardware Capture A Condition 4: Sampled on CHxA port to rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
3	HCPA3	Hardware capture A condition 3: Event trigger 3 from AOS is valid 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
2	HCPA2	Hardware capture A condition 2: Event trigger 2 from AOS is valid 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
1	HCPA1	Hardware capture A condition 1: Event trigger 1 from AOS is valid 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
0	HCPA0	Hardware capture A condition 0: Event trigger 0 from AOS is valid 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid

12.3.18 Hardware Capture B Event Select Register (TIMx_HCPBR)

Address offset: 0x084

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCPB15	HCPB14	HCPB13	HCPB12	HCPB11	HCPB10	HCPB9	HCPB8	HCPB7	HCPB6	HCPB5	HCPB4	HCPB3	HCPB2	HCPB1	HCPB0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Function
31:16	Reserved	-
15	HCPB15	Hardware Capture B Condition 15: Sample to falling edge on TIMTRID port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
14	HCPB14	Hardware Capture B Condition 14: Sampled to rising edge on TIMTRID port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
13	HCPB13	Hardware Capture B Condition 13: Sample to falling edge on TIMTRIC port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
12	HCPB12	Hardware Capture B Condition 12: Sampled to rising edge on TIMTRIC port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
11	HCPB11	Hardware Capture B Condition 11: Sampled to falling edge on TIMTRIB port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
10	HCPB10	Hardware Capture B Condition 10: Sampled to rising edge on TIMTRIB port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
9	HCPB9	Hardware capture B condition 9: TIMTRIA port sampled to falling edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
8	HCPB8	Hardware Capture B Condition 8: Sampling on TIMTRIA port to rising edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
7	HCPB7	Hardware Capture B Condition 7: Sampled on CHxB port to falling edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
6	HCPB6	Hardware Capture B Condition 6: Sampled to rising edge on CHxB port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
5	HCPB5	Hardware Capture B Condition 5: Sampled on CHxA port to falling edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
4	HCPB4	Hardware Capture B Condition 4: Sampled on CHxA port to rising edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
3	HCPB3	Hardware capture B condition 3: Event trigger 3 from AOS is valid 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
2	HCPB2	Hardware capture B condition 2: Event trigger 2 from AOS is valid 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
1	HCPB1	Hardware capture B condition 1: Event trigger 1 from AOS is valid 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
0	HCPB0	Hardware capture B condition 0: Event trigger 0 from AOS is valid 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid

12.3.19 Hardware Increment Event Select Register (TIMx_HCUPR)

Address offset: 0x088

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												HCUP 19	HCUP 18	HCUP 17	HCUP 16
												R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCUP 15	HCUP 14	HCUP 13	HCUP 12	HCUP 11	HCUP 10	HCUP 9	HCUP 8	HCUP 7	HCUP 6	HCUP 5	HCUP 4	HCUP 3	HCUP 2	HCUP 1	HCUP 0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Function
31:20	Reserved	-
19	HCUP19	Hardware increment condition: Event trigger 3 from AOS is valid 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
18	HCUP18	Hardware increment condition: Event trigger 2 from AOS is valid 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
17	HCUP17	Hardware increment condition: Event trigger 1 from AOS is valid 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
16	HCUP16	Hardware increment condition: the event from AOS triggers 0 to be valid 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
15	HCUP15	Hardware increment condition: TIMTRID port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
14	HCUP14	Hardware increment condition: TIMTRID port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
13	HCUP13	Hardware increment condition: TIMTRIC port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
12	HCUP12	Hardware increment condition: TIMTRIC port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
11	HCUP11	Hardware increment condition: TIMTRIB port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
10	HCUP10	Hardware increment condition: TIMTRIB port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
9	HCUP9	Hardware increment condition: TIMTRIA port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
8	HCUP8	Hardware increment condition: TIMTRIA port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
7	HCUP7	Hardware increment condition: When the CHxB port is high level, the CHxA port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
6	HCUP6	Hardware increment condition: When the CHxB port is high level, the CHxA port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
5	HCUP5	Hardware increment condition: When the CHxB port is low level, the CHxA port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
4	HCUP4	Hardware increment condition: When the CHxB port is low level, the CHxA port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
3	HCUP3	Hardware increment condition: When the CHxA port is high level, the CHxB port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches

2	HCUP2	Hardware increment condition: When the CHxA port is high level, the CHxB port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
1	HCUP1	Hardware increment condition: When the CHxA port is low level, the CHxB port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
0	HCUP0	Hardware increment condition: When the CHxA port is low level, the CHxB port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches

12.3.20 Hardware Decrement Event Select Register (TIMx_HCDOR)

Address offset: 0x08C

Reset value: 0x0000 0000

31	30	29	28	27 26	26	25	24	23	22	21	20	19	18	17	16
Reserved												HCD 0 19	HCD 0 18	HCD 0 17	HCD 0 16
												R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCD 0 15	HCD 0 14	HCD 0 13	HCD 0 12	HCD 0 11	HCD 0 10	HCD 0 9	HCD 0 8	HCD 0 7	HCD 0 6	HCD 0 5	HCD 0 4	HCD 0 3	HCD 0 2	HCD 0 1	HCD 0 0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Function
31:20	Reserved	-
19	HCD019	Hardware decrement condition: Event trigger 3 from AOS is valid 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
18	HCD018	Hardware decrement condition: Event trigger 2 from AOS is valid 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
17	HCD017	Hardware decrement condition: Event trigger 1 from AOS is valid 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
16	HCD016	Hardware decrement condition: the event from AOS triggers 0 to be valid 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
15	HCD015	Hardware decrement condition: TIMTRID port sampled to falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
14	HCD014	Hardware decrement condition: Sampled to rising edge on TIMTRID port 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
13	HCD013	Hardware decrement condition: Sampled to falling edge on TIMTRIC port 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
12	HCD012	Hardware decrement condition: Sampled on TIMTRIC port to rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
11	HCD011	Hardware decrement condition: TIMTRIB port sampled to falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
10	HCD010	Hardware decrement condition: Sampled on TIMTRIB port to rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
9	HCD09	Hardware decrement condition: TIMTRIA port sampled to falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
8	HCD08	Hardware decrement condition: Sampled to rising edge on TIMTRIA port 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
7	HCD07	Hardware decrement condition: When CHxB port is high level, CHxA port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
6	HCD06	Hardware decrement condition: When CHxB port is high level, CHxA port is sampled to a rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
5	HCD05	Hardware decrement condition: When the CHxB port is low, the CHxA port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
4	HCD04	Hardware decrement condition: When CHxB port is low level, CHxA port is sampled to a rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match

3	HCD03	Hardware decrement condition: When the CHxA port is high, the CHxB port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
2	HCD02	Hardware decrement condition: When CHxA port is high level, CHxB port is sampled to a rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
1	HCD01	Hardware decrement condition: When the CHxA port is low, the CHxB port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
0	HCD00	Hardware decrement condition: When CHxA port is low level, CHxB port is sampled to a rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match

12.3.21 Software Synchronization Start Register (TIMx_SSTAR)

Address offset: 0x3F4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													SSTA 2	SSTA 1	SSTA 0
													R/W	R/W	R/W

Bit	Marking	Function
31:3	Reserved	
2	SSTA2	Timer6 software start 0: Software startup is invalid 1: Software startup enable
1	SSTA1	Timer5 software start 0: Software startup is invalid 1: Software startup enable
0	SSTA0	Timer4 software start 0: Software startup is invalid 1: Software startup enable

12.3.22 Software Sync Stop Register (TIMx_SSTPR)

Address offset: 0x3F8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													SSTP 2	SSTP 1	SSTP 0
													R/W	R/W	R/W

Bit	Marking	Function
31:3	Reserved	
2	SSTP2	Timer6 software stop 0: Software stop is invalid 1: Software stop enable
1	SSTP1	Timer5 software stop 0: Software stop is invalid 1: Software stop enable
0	SSTP0	Timer4 software stop 0: Software stop is invalid 1: Software stop enable

12.3.23 Software Synchronous Clear Register (TIMx_SCLRR)

Address offset: 0x3FC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													SCLR 2	SCLR 1	SCLR 0
													R/W	R/W	R/W

Bit	Marking	Function
31:3	Reserved	
2	SCLR2	Timer6 software reset 0: Software reset is invalid 1: Software clear enable
1	SCLR1	Timer5 software reset 0: Software reset is invalid 1: Software clear enable
0	SCLR0	Timer4 software reset 0: Software reset is invalid 1: Software clear enable

12.3.24 Interrupt Flag Register (TIMx_IFR)

Address offset: 0x100

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAMH F	SAML F	Reserved					DTEF	UDFF	OVFF	Reserved		CMDF	CMC F	CMB F	CMA F
RO	RO						RO	RO	RO			RO	RO	RO	RO

Bit	Marking	Place name	Function
31:16	Reserved	-	
15	SAMHF	CHxA/B port high state interrupt flag	0: No simultaneous high level on CHxA and CHxB ports 1: High level on both CHxA and CHxB ports
14	SAMLF	CHxA/B port low state interrupt flag	0: No simultaneous low on CHxA and CHxB ports 1: Low level on both CHxA and CHxB ports
13:9	Reserved	-	
8	DTEF	Dead Time Error Interrupt Flag	0: no dead time error occurred; 1: dead time error occurred
7	UDFF	Underflow match interrupt flag	0: Sawtooth underflow does not occur or triangle wave counts to the valley point 1: A sawtooth wave underflow occurs or a triangular wave counts to a valley point
6	OVFF	Overflow match interrupt flag	0: No sawtooth overflow or triangular wave counting to peak 1: Sawtooth overflow occurs or triangular wave counts to peak
5:4	Reserved	-	
3	CMDF	Count match D interrupt flag	0: The value of the GCMDR register is not equal to the count value; 1: The value of the GCMDR register is equal to the count value
2	CMCF	Count match C interrupt flag	0: The value of the GCMCR register is not equal to the count value; 1: The value of the GCMCR register is equal to the count value
1	CMBF	count match B interrupt flag	0: The value of the GCMBR register is not equal to the count value, and the CHxB capture completion action has not occurred 1: The value of the GCMBR register is equal to the count value, or a CHxB capture complete action occurs
0	CMAF	count match A interrupt flag	0: The value of the GCMAF register is not equal to the count value, and the CHxA capture completion action has not occurred 1: The value of the GCMAF register is equal to the count value, or a CHxA capture complete action occurs

12.3.25 Interrupt Flag Clear Register (TIMx_ICLR)

Address offset: 0x104

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAMH C	SAML C	Reserved -					DTEC	UDFC	OVFC	Reserved	CMDC	CMC C	CMB C	CMA C	
W0	W0						W0	W0	W0		W0	W0			

Bit	Marking	Function
31:16	Reserved	-
15	SAMHC	CHxA/B port high status interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
14	SAMLC	CHxA/B port low state interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
13:9	Reserved	-
8	DTEC	The dead time error interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
7	UDFC	The underflow match interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
6	OVFC	The overflow match interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
5:4	Reserved	-
3	CMDC	Count match D interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
2	CMCC	The count matches the C interrupt flag to clear, writing 1 is invalid, writing 0 to clear the corresponding interrupt
1	CMBC	Count match B interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
0	CMAC	Count match A interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt

12.3.26 Spread spectrum and interrupt trigger selection (TIMx_CR)

Address offset: 0x108

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					DITENS	DITENB	DITENA	UDFE	OVFE	Reserved		CMD E	CMCE	CMBE	CMAE
					R/W	R/W	R/W	R/W	R/W			R/W	R/W	R/W	R/W

Bit	Marking	Function
31:11	Reserved	-
10	DITENS	PWM spread spectrum count selection 0: select underflow, 1: select overflow
9	DITENB	PWM channel B spread spectrum enable 0: Enable invalid, 1: Enable valid, change PWM output delay every cycle
9	DITENA	PWM channel A spread spectrum enable 0: Enable invalid, 1: Enable valid, change PWM output delay every cycle
7	UDFE	Underflow match enables ADC trigger 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS i_tirg
6	OVFE	Overflow match enable to trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS i_tirg
5:4	Reserved	-
3	CMDE	Count match D enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS i_tirg
2	CMCE	Count Match C Enable Trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS i_tirg
1	CMBE	Count match B enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS i_tirg
0	CMAE	Count match A enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS i_tirg

12.3.27 AOS Selection Control Register (TIMx_AOSSR)

Address offset: 0x110

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SMH 2	SMH 1	SMH0	SML2	SML1	SML0	SOFTB K	Reserved		BFILTE N	BFILTS		FSAME	FBRAK E
		R/W	R/W	R/W	R/W	R/W	R/W	R/W			R/W	R/W	R	R	

Bit	Marking	Function
31:14	Reserved	-
13	SMH2	Same height selection for channel 2 0: The selection is invalid, 1: The selection is valid, AOS _i odis[1] appears when the same high
12	SMH1	Channel 1 same height selection 0: The selection is invalid, 1: The selection is valid, AOS _i odis[1] appears when the same high
11	SMH0	Channel 0 same high selection 0: The selection is invalid, 1: The selection is valid, AOS _i odis[1] appears when the same high
10	SML2	Channel 2 with low selection 0: The selection is invalid, 1: The selection is valid, AOS _i odis[1] appears when the same low
9	SML1	Channel 1 with low selection 0: The selection is invalid, 1: The selection is valid, AOS _i odis[1] appears when the same low
8	SML0	Channel 0 with low selection 0: The selection is invalid, 1: The selection is valid, AOS _i odis[1] appears when the same low
7	SOFTBK	Software brake: Write 1 to realize software brake
13	Reserved	-
4	BFILTEN	Port brake filter enable
3:2	BFILTS	Port brake filter clock selection
1	FSAME	Same high and same low brake flag, read only
0	FBRAKE	Port brake flag, read only

12.3.28 AOS Selection Control Register Flag Clear (TIMx_AOSCL)

Address offset: 0x114

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FSAME	FBRAKE	
													R	R	

Bit	Marking	Function
31:2	Reserved	-
1	FSAME	Same high and same low brake flag to clear, write 0 to clear, write 1 to be invalid, read constant to 1
0	FBRAKE	The port brake flag is cleared, write 0 to clear, write 1 to be invalid, read 1

12.3.29 Port Brake Control Register (TIMx_PTBKS)

Address offset: 0x118

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Function
31:16	Reserved	-
15	EN15	P36 Brake port enable: 1 selection, 0 invalid
14	EN14	P35 brake port enable: 1 selected, 0 invalid
13	EN13	P34 brake port enable: 1 selected, 0 invalid
12	EN12	P33 brake port enable: 1 selected, 0 invalid
11	EN11	P32 brake port enable: 1 selected, 0 invalid
10	EN10	P31 brake port enable: 1 selected, 0 invalid
9	EN9	P27 brake port enable: 1 selected, 0 invalid
8	EN8	P26 brake port enable: 1 selected, 0 invalid
7	EN7	P26 brake port enable: 1 selected, 0 invalid
6	EN6	P24 brake port enable: 1 selected, 0 invalid
5	EN5	P23 brake port enable: 1 selected, 0 invalid
4	EN4	P15 brake port enable: 1 selected, 0 invalid
3	EN3	P14 brake port enable: 1 selected, 0 invalid
2	EN2	P03 brake port enable: 1 selected, 0 invalid
1	EN1	P02 brake port enable: 1 selected, 0 invalid
0	EN0	P01 brake port enable: 1 selected, 0 invalid

12.3.30 Port Trigger Control Register (TIMx_TTRIG)

Address offset: 0x11C

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

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
TRIGDS				TRIGCS				TRIGBS				TRIGAS			
R/W				R/W				R/W				R/W			
Bit		Marking		Function											
31:16		Reserved		-											
15:12		TRIGDS		TIMx trigger D port selection											
11:8		TRIGCS		TIMx trigger C port selection											
7:4		TRIGBS		TIMx trigger B port selection											
3:0		TRIGAS		TIMx trigger A port selection											

The control signal and port selection are as follows

0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
P01	P02	P03	P15	P14	P23	P24	P25	P26	P27	P31	P32	P33	P34	P35	P36

12.3.31 AOS Trigger Control Register (TIMx_ITRIG)

Address offset: 0x120

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IAOS3S				IAOS2S				IAOS1S				IAOS0S			
R/W				R/W				R/W				R/W			
Bit		Marking		Function											
31:16		Reserved		-											
15:12		IAOS3S		TIMx AOS3 trigger source selection											
11:8		IAOS2S		TIMx AOS2 trigger source selection											
7:4		IAOS1S		TIMx AOS1 trigger source selection											
3:0		IAOS0S		TIMx AOS0 trigger source selection											

The control signal (IAOSxS) and interrupt source selection are as follows (x=0,1,2,3)

0000	0001	0010	0011	0100	0101	0110	0111
TIM0_INT	TIM1_INT	TIM2_INT	LPTIMER_INT	TIM4_INTS	TIM5_INTS	TIM6_INTS	UART0_INT
1000	1001	1010	1011	1100	1101	1110	1111
UART1_INT	LPUART_INT	VC1_INT	VC2_INT	RTC_INT	PCA_INT	SPI_INT	ADC_INT

12.3.32 Port Brake Polarity Control Register (TIMx_PTBKP)

Address offset: 0x124

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POL15	POL14	POL13	POL12	POL11	POL10	POL9	POL8	POL7	POL6	POL5	POL4	POL3	POL2	POL1	POL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Function
31:16	Reserved	-
15	POL15	P36 brake port: 1 is active at low level, 0 is active at high level
14	POL14	Polarity selection of P35 brake port: 1 is active at low level, 0 is active at high level
13	POL13	Polarity selection of P34 brake port: 1 is active at low level, 0 is active at high level
12	POL12	Polarity selection of P33 brake port: 1 is active at low level, 0 is active at high level
11	POL11	Polarity selection of P32 brake port: 1 is active at low level, 0 is active at high level
10	POL10	Polarity selection of P31 brake port: 1 is active at low level, 0 is active at high level
9	POL9	Polarity selection of P27 brake port: 1 is active at low level, 0 is active at high level
8	POL8	Polarity selection of P26 brake port: 1 is active at low level, 0 is active at high level
7	POL7	Polarity selection of P26 brake port: 1 is active at low level, 0 is active at high level
6	POL6	Polarity selection of P24 brake port: 1 is active at low level, 0 is active at high level
5	POL5	Polarity selection of P23 brake port: 1 is active at low level, 0 is active at high level
4	POL4	Polarity selection of P15 brake port: 1 is active at low level, 0 is active at high level
3	POL3	Polarity selection of P14 brake port: 1 is active at low level, 0 is active at high level
2	POL2	Polarity selection of P03 brake port: 1 is active at low level, 0 is active at high level
1	POL1	Polarity selection of P02 brake port: 1 is active at low level, 0 is active at high level
0	POL0	Polarity selection of P01 brake port: 1 is active at low level, 0 is active at high level

13 Real Time Clock (RTC)

13.1 Introduction to Real Time Clocks

The real-time clock / calendar 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 AM/PM register bits. Table 14-1 Shows Its Basic Characteristics.

Table 13-1 Basic characteristics of RTC

Timer	Off-chip low-speed crystal oscillator XTL (32.768kHz) On-chip low-speed oscillator RCL (32kHz, 1% precision) Off-chip high-speed crystal oscillator XTH
Basic functions	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
	With alarm clock function
	With high-precision 1Hz square wave signal output
Interrupt	With periodic interrupt
	With alarm interrupt

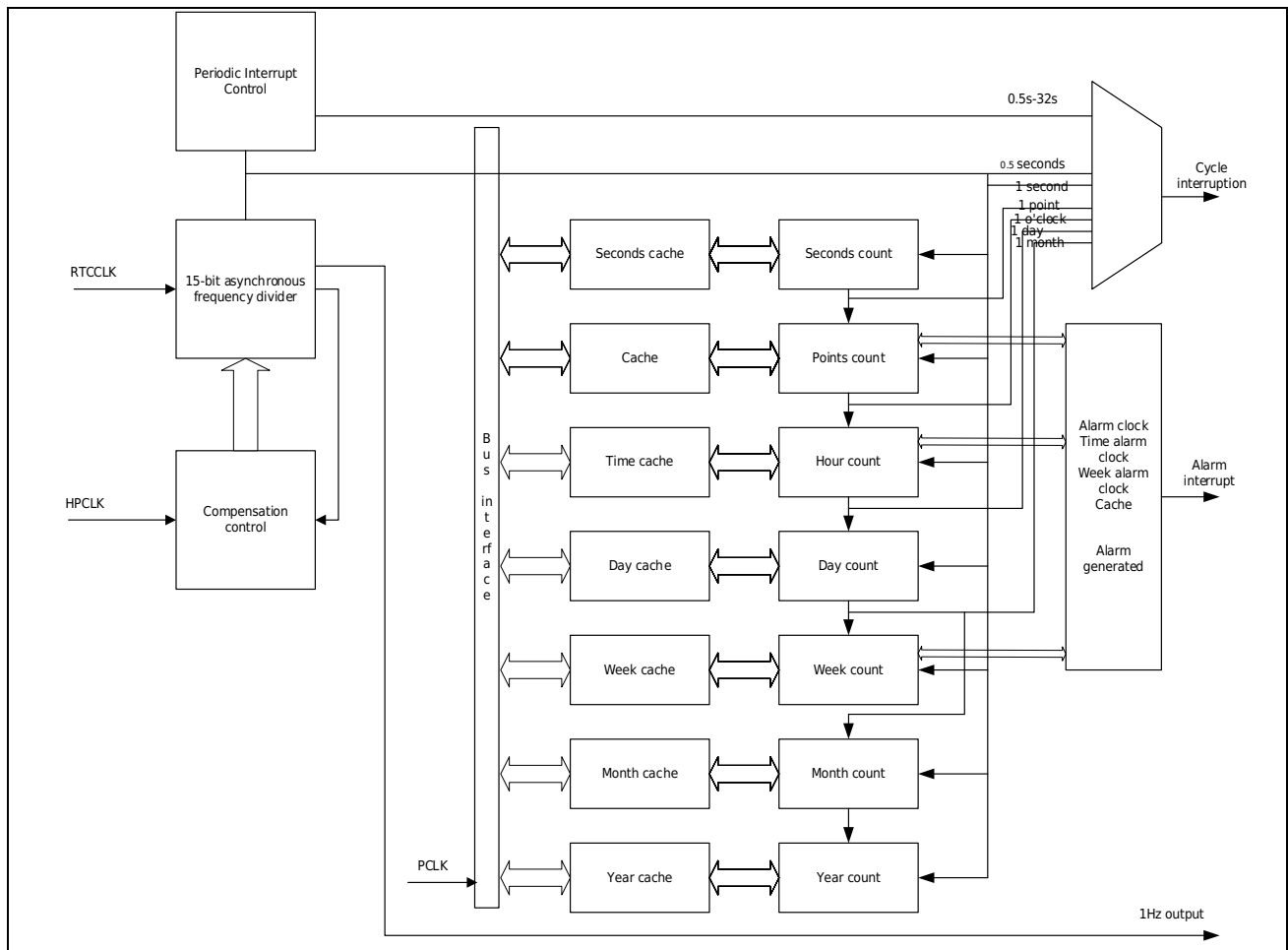


Figure 13-1 RTC Block Diagram

13.2 Real Time Clock Functional Description

The clock source of the real-time clock can be configured as an external low-speed crystal oscillator, an external high-speed crystal oscillator, or an internal low-speed RC; the external low-speed crystal oscillator is used by default. The control registers CR0, CR1 and COMPEN are only controlled by power-on reset, and other reset sources cannot reset these three control registers. The power-on status of other data registers is uncertain, and needs to be initialized after power-on, and will not be affected by any reset.

All the date and time values written and read by the software are BCD codes, and there is no need to convert hexadecimal to decimal.

Any invalid date time will not be able to be written, such as 32nd, 25th, 70th second, month B, etc.

13.2.1 Power-on setting

The RTC is reset once after power-on. When the system is not powered off, various external reset requests cannot reset the RTC, and the RTC will always be in the counting state. After power-on, after setting the calendar initial value, alarm clock setting, error compensation, interrupt, etc., start the RTC.

13.2.2 RTC count start setting

1. Set CR0.START=0, the counting stops;
2. Set CR0.AMPM and CR0.PRDS, CR0.PRDX set time and interrupt cycle;
3. Set CR1.CKSEL to select the timing clock of RTC;
4. Set the calendar counting registers for seconds, minutes, hours, weeks, days, months, and years;
5. When clock error compensation is required, set the counting clock error compensation register COMPEN;
6. Clear the interrupt flag bits CR1.ALMF, CR1.PRDF, and enable the interrupt;
7. Set CR0.START=1 to start counting.

13.2.3 System low power mode switching

After the RTC counting starts, if the system switches to the low power consumption mode immediately, please perform any of the following confirmations before switching the mode.

The control register is in the system control register SYCTRL1.RTC_LPW

1. After setting CR0.START=1, switch modes after more than 2 RTC count clocks.
2. After setting CR0.START=1, set CR1.WAIT=1 and query CR1.WAITF=1. Then set CR1.WAIT=0, query CR1.WAITF=0 and then switch modes.

In RTC low power mode, RTC registers cannot be read or written. In low power mode, the RTC consumes less current.

Switching low power mode while RTC is running does not need to wait.

13.2.4 read count register

There are three ways to read the count register:

- Mode 1: Read mode 1 at any time
 1. Set CR1.WAIT=1, stop the calendar register counting, and enter the read and write mode;
 2. Query until CR1.WAITF=1;
 3. Read the second, minute, hour, week, day, month, year count register value;
 4. Set CR1.WAIT=0, the counter counts;
 5. Query until CR1.WAITF=0.
- Mode 2: Read mode 2 at any time
 1. Read minutes, hours, weeks, days, months, and year counting register values;
 2. Read out the second count register value;
 3. Read the second count register value again;
 4. Judging whether the reading values of the two seconds are the same, if they are different, start from the first step again, and end the same reading.
- Mode 3: Interrupt read mode

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.

13.2.5 write count register

1. Set CR1.WAIT=1, stop the calendar register counting, and enter the read and write mode;
2. Query until CR1.WAITF=1;
3. Write the second, minute, hour, week, day, month, year count register value;
4. Set CR1.WAIT=0, the counter starts counting again. Note that all write operations must be completed within 1 second;
5. Query until CR1.WAITF=0.

WAIT to write seconds, minutes, hours, weeks, days, months, and year count registers in RTC disabled mode.

Note:

- Changing the seconds register in the counting mode will reset the seconds counting, writing minutes, hours, weeks, days, months, years counting register values will not affect the RTC counting.

13.2.6 Alarm clock setting

1. Set CR1.ALMEN=0, the alarm clock is disabled;
2. Set CR1.ALMIE=1, the alarm clock interruption is allowed;
3. Minute alarm clock ALMMIN, hour alarm clock ALMHOUR, week alarm clock ALMWEAK setting;
4. Set CR1.ALMEN=1, the alarm clock is allowed;
5. Wait for an interrupt to occur;
6. Since the alarm clock interrupt and the fixed-period interrupt share the interrupt request signal, when CR1.ALMF=1, enter the alarm clock interrupt processing; otherwise, enter the fixed-period interrupt processing.

13.2.7 1Hz output

RTC can choose to output three kinds 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 PCLK with different frequencies, it outputs a high-precision 1Hz clock. The system control register needs to be configured according to the PCLK frequency, where,

- 1Hz output of general precision is set as follows: (no clock compensation)
 1. Set CR0.START=0, the counting stops;
 2. RTC output pin setting;
 3. CR0.1HZOE=1, clock output permission;

4. Set CR0.START=1 to start counting;
 5. Wait for more than 2 count cycles;
 6. 1Hz output starts.
- The higher precision 1Hz output setting is as follows: (low speed compensation)
 1. Set CR0.START=0, the counting stops;
 2. RTC output pin setting;
 3. CR0.1HZOE=1, clock output permission;
 4. Clock error compensation register COMPEN.CR compensation number setting;
 5. Clock error compensation register COMPEN.EN=1, error compensation is valid;
 6. Set CR0.START=1 to start counting;
 7. Wait for more than 2 count cycles;
 8. 1Hz output starts.
 - When the high-precision 1Hz output is required, it is necessary to provide 4M, 6M, 8M, 12M, 16M, 20M, 24M, 32MHz high-speed PCLK clocks for the RTC on the basis of higher-precision output. The output settings are as follows:
 1. Set CR0.START=0, the counting stops;
 2. RTC output pin setting;
 3. CR0.1HZOE=1, clock output permission;
 4. CR0.1HZSEL=1, choose to output high-precision 1Hz clock;
 5. Configure high-speed clock compensation clock SYSCTRL1.RTC_FREQ_ADJUST
 6. Clock error compensation register COMPEN.CR[8:0] compensation number setting;
 7. Clock error compensation register COMPEN.EN=1, accuracy compensation is valid;
 8. Set CR0.START=1 to start counting;
 9. Wait for more than 2 count cycles;
 10. 1Hz output starts.

13.2.8 Clock Error Compensation

Because there is an error in the external crystal oscillator, when it is necessary to obtain a high-precision counting result, the error needs to be compensated. There are two types of compensation methods: the first, error compensation based on its own clock; the second, error compensation based on a high-speed clock.

The principle and calculation of error compensation based on its own clock:

Since the counter uses a 32.768KHz clock to count, if it is necessary to compensate the accuracy per second, it can only be compensated according to the integer cycle of 32.768KHz, and the minimum unit of compensation per second is $(1/32768)*10^6=30.5\text{ppm}$, which cannot be satisfied 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 $30.5\text{ppm}/32=0.96\text{ppm}$. It meets the clock compensation requirement with high precision. And the compensation occurs in a relatively uniform range every 32 seconds. 5 decimal places is introduced in this register.

Example 1:

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 oscillation frequency = $32768 \times 0.9999888 \approx 32767.63$

Compensation target value = $(\text{actual oscillation frequency} - \text{target frequency})/\text{target frequency} \times 10^6$

$$= (32767.96 - 32768) / 32768 \times 10^6$$

$$= -11.29\text{ppm}$$

Basis

$$\text{CR}[8:0] = \left(\frac{\text{Compensation target value}[\text{ppm}] \times 2^{15}}{10^6} \right)_{\text{Take 2's complement}} + 0001.00000\text{B}$$

If the compensation target value is -11.29ppm, calculate the corresponding register value as follows:

$$\text{CR}[8:0] = (-11.29 \times 2^{15}/10^6) \text{ 2's complement} + 0001.00000\text{B}$$

$$= (-0.37) \text{ 2's complement} + 0001.00000\text{B}$$

$$= 1111.10101\text{B} + 0001.00000\text{B}$$

$$= 0000.10101\text{B}$$

Principle and calculation of error compensation based on high-speed 24MHz clock:

The calculation method of this method is the same as the error compensation based on its own clock. Due to the introduction of a 4M-32MHz high-speed clock, the $1/32768$ second error that originally needs to be accumulated within a maximum of 32 seconds can be dispersed to every 1 second, and a minimum 0.96ppm (23 24MHz clock cycles) compensation is performed for each 1 second to achieve an average High-precision 1Hz clock output per second.

13.3 RTC Interrupt

The RTC supports two types of interrupts. Alarm clock interruption, regular period interruption. The alarm clock interrupt and the periodic interrupt share an interrupt signal.

13.3.1 RTC alarm interrupt

When CR1.ALMIE=1, if the current calendar time is equal to the minute alarm clock register (ALMMIN), hour alarm clock register (ALMHOUR), and weekly alarm clock register (ALMWEEK), an alarm clock interrupt is triggered.

13.3.2 RTC cycle interrupt

ALMIE=1 of the control register 1 (CR1), after the selected period occurs, a fixed-period wake-up interrupt is triggered. Since the alarm clock and the fixed-period shared interrupt, it is distinguished by the flag register bit.

13.4 RTC register description

Table 13-2 RTC register list

Base address 0X40001400

Register	Offset address	Description
RTC_CR0	0X000	Control register 0
RTC_CR1	0X004	Control register 1
RTC_SEC	0X008	Second Count Register
RTC_MIN	0X00C	Minute counter register
RTC_HOUR	0X010	Time counter register
RTC_WEEK	0X014	Week Count Register
RTC_DAY	0X018	Day Count Register
RTC_MON	0X01C	Month Count Register
RTC_YEAR	0X020	Year Count Register
RTC_ALMMIN	0X024	Minute alarm register
RTC_ALMHOUR	0X028	Time alarm register
RTC_ALMWEEK	0X02C	Weekly Alarm Register
RTC_COMPEN	0X030	Clock Error Compensation Register

13.4.1 Control Register 0 (RTC_CR0)

* Only power-on reset is valid for this register

Address offset: 0x000

Reset value 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13:8	7	6	5	4	3	2:0
Res.	PRDSEL	PRDX	START	HZ1SEL	HZ1OE	Res.	AMPM	PRDS
	R/W	R/W	R/W	R/W	R/W		R/W	R/W

Bit	Symbol	Functional description
31:15	Reserved	-
14	PRDSEL	0: Use the periodic interrupt time interval set by PRDS 1: Use the periodic interrupt time interval set by PRDX
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. 000000: 0.5 seconds 000001: 1 second 111110: 31.5 seconds 111111: 32 seconds
7	START	0: stop RTC counter 1: Enable RTC counter
6	1HZSEL	0: Normal precision 1Hz output 1: High precision 1Hz output
5	1HZOE	0: disable 1Hz output 1: Enable 1Hz output
4	Reserved	-
3	AMPM	0: 12 hours 1: 24 hours
2:0	PRDS	Set the interval at which interrupts are generated: 000: no periodic interrupt 001: 0.5 seconds 010: 1 second 011: 1 minute 100: 1 hour 101: 1 day 11x: January _ Note: If you need to write and change the time interval of cycle interruption when START=1, the operation steps are as follows: step1, turn off the RTC interrupt in the NVIC; step2, change the time interval of periodic interruption; step3, clear the RTC interrupt flag; step4, enable RTC interrupt.

13.4.2 Control Register 1 (RTC_CR1)

* Only power-on reset is valid for this register

Address offset: 0x004

Reset value 0X00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15:11	10:8	7	6	5	4	3	2	1	0
Reserved	CKSEL	ALMEN	ALMIE	Res.	ALMF	PRDF	Res.	WAITF	WAIT
	R/W	R/W	R/W		RO	RO		R/W	R/W

Bit	Symbol	Functional description
31:11	Reserved	-
10:8	CKSEL	RTC clock selection 00x: XTL 32.768k 01x: RCL 32k 100: XTH/128 (choose this item when the crystal oscillator is 4M) 101: XTH/256 (choose this item when the crystal oscillator is 8M) 110: XTH/512 (choose this item when the crystal oscillator is 16M) 111: XTH/1024 (select this item when the crystal oscillator is 32M)
7	ALMEN	0: disable alarm clock 1: enable alarm clock Note: When ALMEN is enabled during START=1 calendar counting process and ALMIE=1 interrupt permission, please disable the system interrupt to prevent malfunction. After enabling, please clear the ALMF flag bit.
6	ALMIE	0: disable alarm interrupt 1: enable alarm clock interrupt
5	Reserved	-
4	ALMF	0: No alarm interrupt occurred 1: Alarm interrupt has occurred Note: This bit is only valid when ALMEN=1. When the alarm clock matches, a clock of 32.768KHz is set to 1. Writing 0 clears the flag, writing 1 has no effect.
3	PRDF	0: Periodic interrupt does not occur 1: A cycle interrupt has occurred to 1 after a periodic interrupt occurs. Writing 0 clears this flag, writing 1 has no effect.
2	Reserved	-
1	WAITF	0: Not writing / reading status 1: Write / read status Note: WAIT bit setting is valid flag. "1" before writing / 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.
0	WAIT	0: normal counting mode 1: write / read mode Note: 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".

13.4.3 Second Count Register (RTC_SEC)

Address offset: 0x008

Reset value: Indefinite

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									SECH		SECL				
									R/W		R/W				

Bit	Symbol	Functional description
31:7	Reserved	-
6:4	SECH	Tens of seconds value
3:0	SECL	Second count unit value

Represents 0-59 seconds, counted in decimal. Please write the BCD code of decimal 0-59, when writing wrong value, the written value will be ignored.

13.4.4 Minute Count Register (RTC_MIN)

Address offset: 0x00C

Reset value: Indefinite

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									MINH		MINL				
									R/W		R/W				

Bit	Symbol	Functional description
31:7	Reserved	-
6:4	MINH	Score tens digit value
3:0	MINL	Sub-count unit value

Represents 0-59 points, using decimal counting. Please write the BCD code of decimal 0-59, when writing wrong value, the written value will be ignored.

13.4.5 Hour count register (RTC_HOUR)

Address offset: 0x010

Reset value: Indefinite

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										HOURH		HOURL			
										R/W		R/W			

Bit	Symbol	Functional description
31:6	Reserved	-
5:4	HOURH	Tens of digits
3:0	HOURL	Hour counter value

In 24-hour format, it means 0-23 hours. 12- hour format, b5=0 means AM, then 01:12 means morning; b5=1 means PM, then 21:32 means afternoon.

Please set the correct decimal 0:23 or 01:12, 21:32 BCD code according to the value of AMPM. Writing a value out of range will be ignored.

Please refer to the following table for the specific time:

24 hour clock	AMPM=1	12 hour clock	AMPM=0
Time	Register representation	Time	Register representation
00 HOUR	00H	AM 12:00	12H
01 HOUR	01H	AM 1:00	01H
02 HOUR	02H	AM 2:00	02H
03 HOUR	03H	AM 3:00	03H
04 HOUR	04H	AM 4:00	04H
05 HOUR	05H	AM 5:00	05H
06 HOUR	06H	AM 6:00	06H
07 HOUR	07H	AM 7:00	07H
08 HOUR	08H	AM 8:00	08H
09 HOUR	09H	AM 9:00	09H
10 HOUR	10H	AM 10:00	10H
11 HOUR	11H	AM 11:00	11H
12 HOUR	12H	PM 12:00	32H
13 HOUR	13H	PM 1:00	21H
14 HOUR	14H	PM 2:00	22H
15 HOUR	15H	PM 3:00	23H
16 HOUR	16H	PM 4:00	24H
17 HOUR	17H	PM 5:00	25H
18 HOUR	18H	PM 6:00	26H
19 HOUR	19H	PM 7:00	27H
20 HOUR	20H	PM 8:00	28H
21 HOUR	21H	PM 9:00	29H
22 HOUR	22H	PM 10:00	30H
23 HOUR	23H	PM 11:00	31H

13.4.6 Day Count Register (RTC_DAY)

Address offset: 0x018

Reset value: Indefinite

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										DAYH		DAYL			
										R/W		R/W			

Bit	Symbol	Functional description
31:6	Reserved	-
5:4	DAYH	Dozens of days
3:0	DAYL	Day count unit value

Decimal means 1:31 day, automatically calculates leap year and month. The specific expression is as follows:

Month	Day count display
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

13.4.7 Week Count Register (RTC_WEEK)

Address offset: 0x014

Reset value: Indefinite

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													WEEK		
													R/W		

Bit	Symbol	Functional description
31:3	Reserved	-
2:0	WEEK	Week count value

Decimal 0:6 means Sunday:Saturday. Please write the correct decimal 0:6 BCD code, other values will be ignored. The corresponding relationship between the week count value is as follows:

Week	Week count
Sunday	00H
Monday	01H
Tuesday	02H
Wednesday	03H
Thursday	04H
Friday	05H
Saturday	06H

13.4.8 Month Count Register (RTC_MON)

Address offset: 0x01C

Reset value: Indefinite

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											MON				
											R/W				

Bit	Symbol	Functional description
31:5	Reserved	-
4:0	MON	Month count value

Decimal 1:12 represents 1:12 months. Please write in the correct decimal 1:12 BCD code, other values will be ignored.

13.4.9 Year Count Register (RTC_YEAR)

Address offset: 0x020

Reset value: Indefinite

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								YEARH				YEARL			
								R/W				R/W			

Bit	Symbol	Functional description
31:8	Reserved	-
7:4	YEARH	Year tens digit value
3:0	YEARL	Year unit count value

Decimal 0:99 means 0:99 years. Count according to the monthly round. Automatically calculate leap years such as: 00, 04, 08, ..., 92, 96, etc. Please write correct decimal year count value, wrong value will be ignored.

13.4.10 Minute Alarm Register (RTC_ALMMIN)

Address offset: 0x024

Reset value: Indefinite

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ALMMINH		ALMMINL				
									R/W		R/W				

Bit	Symbol	Functional description
31:6	Reserved	-
5:4	ALMMINH	Ten digits of minute alarm matching value
3:0	ALMMINL	Minute alarm clock matching value unit digit

Please set the BCD code of decimal 0:59. Write other values, no alarm match will occur.

13.4.11 Hour Alarm register (RTC_ALMHOUR)

Address offset: 0x028

Reset value: Indefinite

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ALMHOURH		ALMHOURL				
									R/W		R/W				

Bit	Symbol	Functional description
31:6	Reserved	-
5:4	ALMHOURH	Time alarm clock ten digit matching value
3:0	ALMHOURL	Hour alarm ones match value

Please set the correct alarm clock matching value according to the time system, otherwise the alarm clock matching will not happen.

13.4.12 Week Alarm Register (RTC_ALMWEAK)

Address offset: 0x02C

Reset value: Indefinite

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ALMWEAK						
									R/W						

Bit	Symbol	Functional description
31:7	Reserved	-
6:0	ALMWEAK	Week alarm match value. b0:b6 correspond to Sunday:Saturday respectively, when corresponding to "1", it means that the alarm clock is valid on that day of the week. For example, b0=1, b5=1 means that the Sunday and Friday alarm settings are valid.

Please set the correct alarm clock matching value according to the time system, otherwise the alarm clock matching will not happen.

13.4.13 Clock Error Compensation Register (RTC_COMPEN)

Address offset: 0x030

* Only power-on reset is valid for this register, reset value: 0x00000020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	Reserved						CR								
R/W							R/W								

Bit	Symbol	Functional description																																																																																																																	
31:16	Reserved	-																																																																																																																	
15	EN	Compensation enable 0: disable clock error compensation 1: Enable clock error compensation																																																																																																																	
14:b	Reserved	-																																																																																																																	
8:0	CR	Compensation value Through the compensation value setting, the accuracy compensation of +/-0.96ppm per second can be performed. The compensation value is 9-bit 2's complement code with a decimal point, and the last 5 bits are the decimal part. Compensable range 274.6ppm: 212.6ppm. Minimum differential error +/-0.48ppm. Minimum resolution 0.96ppm. Please refer to the following table for specific compensation accuracy:																																																																																																																	
		<table><tr><th colspan="10">Compensation value setting</th><th>Compensation</th></tr><tr><th>EN</th><th colspan="9">CR[8:0]</th><th></th></tr><tr><td rowspan="8">1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>-274.6ppm</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>-273.7ppm</td></tr><tr><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>-0.95ppm</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0ppm</td></tr><tr><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td><td>:</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>+211.7ppm</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>+212.6ppm</td></tr><tr><td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>Uncompensated</td></tr></table>	Compensation value setting										Compensation	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	-0.95ppm	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	Uncompensated
		Compensation value setting										Compensation																																																																																																							
		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	-0.95ppm																																																																																																							
			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	Uncompensated																																																																																																										

Explanation and calculation of compensation principle:

Since the counter uses a 32.768KHz clock to count, if it is necessary to compensate the accuracy per second, it can only be compensated according to the integer cycle of 32.768KHz, and the minimum unit of compensation per second is $(1/32768)*10^6=30.5\text{ppm}$, which cannot be satisfied 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 $30.5\text{ppm}/32=0.96\text{ppm}$. It meets the clock compensation requirement with high precision. And the compensation occurs in a relatively uniform range every 32 seconds. 5 decimal places is introduced in this register.

The setpoint is calculated as follows:

$$CR[8:0] = \left(\frac{\text{Compensation target value[ppm]} \times 2^{15}}{10^6} \right) \text{Take 2's complement} + 0001.00000B$$

If the compensation target value is +20.6ppm, calculate the corresponding register value as follows:

$$\begin{aligned} CR[8:0] &= (20.3 \times 2^{15}/10^6) \text{ 2's complement} + 0001.00000B \\ &= (0.6651904) \text{ 2's complement} + 0001.00000B \\ &= 0000.10101B + 0001.00000B \\ &= 0001.10101B \end{aligned}$$

If the compensation target value is -20.6ppm, calculate the corresponding register value as follows:

$$\begin{aligned} CR[8:0] &= (-20.3 \times 2^{15}/10^6) \text{ 2's complement} + 0001.00000B \\ &= (-0.6651904) \text{ 2's complement} + 0001.00000B \\ &= 1111.01011B + 0001.00000B \\ &= 0000.01011B \end{aligned}$$

14 Watchdog Timer (WDT)

14.1 Introduction to WDT

WDT can be used to detect and resolve failures caused by software errors. When the WDT counter reaches the set overflow time, it will trigger an interrupt or generate a system reset. WDT is driven by a dedicated 10KHz on-chip oscillator.

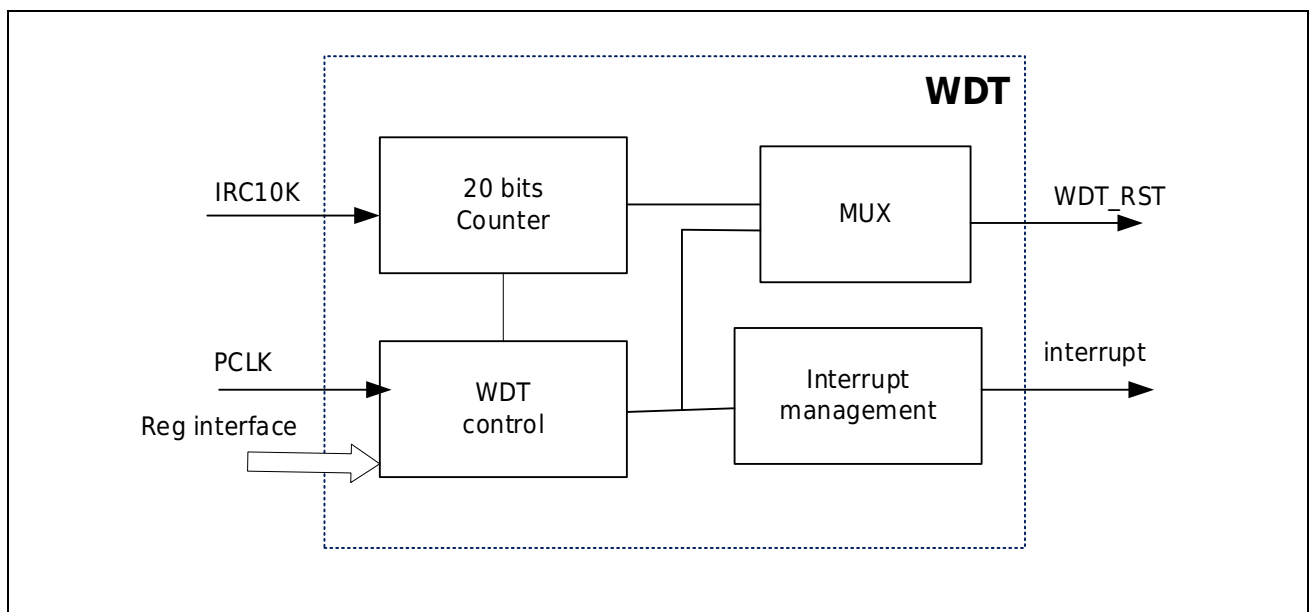


Figure 14-1 Overall block diagram of WDT

14.2 WDT Functional Description

- 20Bit free-running up counter, the overflow time can be configured from 1.6ms to 50s.
- Action after overflow can be configured as interrupt or reset.
- The WDT clock is provided by an independent RC oscillator, which can work in Sleep and DeepSleep modes.
- WDTCON register can only be modified when the WDT is not started to prevent unintentional modification of the WDT configuration after startup.

14.2.1 Interrupt generated after WDT overflow

In this mode, WDT will generate interrupt periodically according to the set time. WDT interrupt flag needs to be cleared in the interrupt service routine.

The configuration method is as follows:

Step1: Configure WDT_CON. WOV, select WDT timer overflow time.

Step2: Set WDT_CON. WINT_EN to 1, select WDT to generate an interrupt after overflow.

Step3: Enable the WDT interrupt in the NVIC interrupt vector table.

Step4: Write 0x1E and 0xE1 to the WDT_RST register in sequence to start the WDT timer.

Step5: Write 0x1E and 0xE1 to the WDT_RST register in the interrupt service routine to clear the interrupt flag.

14.2.2 Reset after WDT overflow

In this mode, the Reset signal will be generated after the WDT counter overflows, which will reset the MCU. User program needs to clear the WDT counter before WDT overflow, so as to avoid WDT reset.

The configuration method is as follows:

Step1: Configure WDT_CON. WOV, select WDT counter overflow time.

Step2: Set WDT_CON. WINT_EN to 0, select WDT to generate reset after overflow.

Step3: Write 0x1E and 0xE1 to the WDT_RST register in sequence to start the WDT timer.

Step4: Before the WDT overflows, write 0x1E and 0xE1 to the WDT_RST register to clear the WDT counter.

Note: Since the WDT oscillator is a low-precision RC oscillator, it is strongly recommended to clear the WDT before the WDT counter reaches half of the overflow value.

14.3 WDT register description

Table 14-1 WDT register list

Base address 0X40000C00

Register	Offset address	Description
WDT_RST	0X080	WDT Clear Control Register
WDT_CON	0X084	WDT control register

14.3.1 WDT Clear Control Register (WDT_RST)

Offset address: 0x080

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved	WDTRST							
	WO							

Bit	Symbol	Description
31:8	Reserved	Reserved bit, read as 0
7:0	WDTRST	Watchdog start / clear control When the watchdog is not started, write 0x1E and 0xE1 to this register in turn to start the WDT timer. When the watchdog is enabled, write 0x1E and 0xE1 to this register in turn to clear the WDT timer and interrupt flag.

14.3.2 WDT_CON register

Offset address: 0x084

Reset value: 0x0000/ 000F

Note:

- This register can only be written when WDT is not running.

31:16	15:8	7	6	5	4	3	2	1	0
Reserved	WCNTL	WDINT	Res.	WINT_EN	WDTR	WOV			
	RO	RO		R/W	RO	R/W			

Bit	Symbol	Description
31:16	Reserved	Reserved bit, read as 0
15:8	WCNTL	WDT counter low 8 bits
7	WDINT	WDT interrupt flag 1: WDT interrupt has occurred, write 0x1E, 0xE1 to the WDT_RST register to clear the interrupt flag. 0: No WDT interrupt occurs.
5	WINT_EN	Action configuration after WDT overflow 1: Generate interrupt after WDT overflow. 0: A reset is generated after WDT overflows.
4	WDTR	WDT running flag 1: WDT is running 0: WDT stop
3:0	WOV[3:0]	WDT timeout time configuration 0000: 1.6ms 1000: 500ms 0001: 3.2ms 1001: 820ms 0010: 6.4ms 1010: 1.64s 0011: 13ms 1011: 3.28s 0100: 26ms 1100: 6.55s 0101: 51ms 1101: 13.1s 0110: 102ms 1110: 26.2s 0111: 205ms 1111: 52.4s

15 General Synchronous Asynchronous Transceiver (UART)

15.1 Overview

This product has 2 general-purpose UART modules (UART0/1). The universal synchronous asynchronous transceiver (UART) can flexibly perform full-duplex data exchange with external devices. It supports synchronous one-way communication and multi-processor communication. It is often used in short-distance, low-speed serial communication. The UART provides a variety of baud rates through a programmable baud rate generator. The baud rate of UART0 is generated by TIMER0, and the baud rate of UART1 is generated by TIMER1. UART supports multiple working modes.

15.2 Structural block diagram

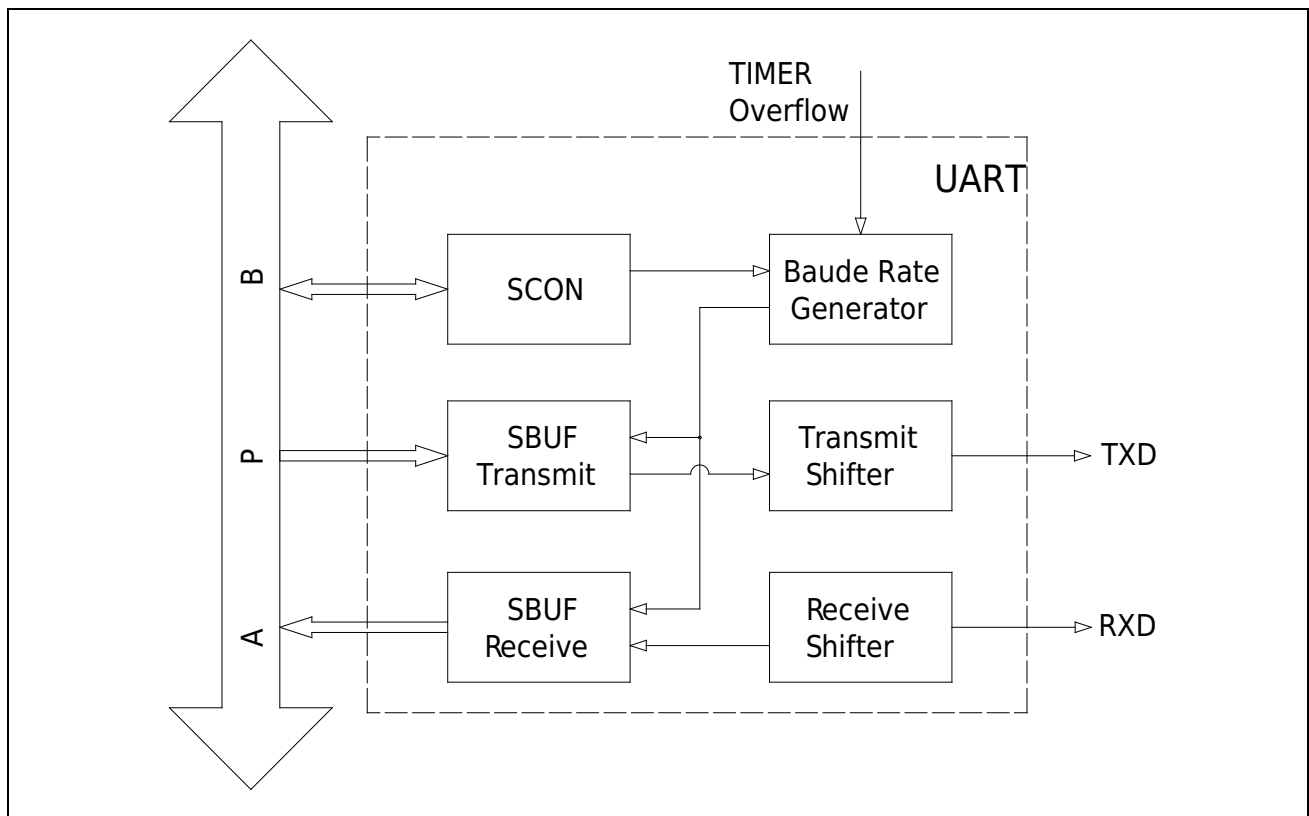


Figure 15-1 UART Block Diagram

15.3 Main characteristics

General purpose UART module supports the following basic functions:

- Full-duplex transmission, half-duplex transmission
- Programmable serial communication function
 - Two character lengths: 8 bits, 9 bits
 - Mode0/1/2/3 four transmission modes
- 16-bit baud rate generator

- Multi-machine communication
- Automatic Address Recognition

15.4 Functional description

15.4.1 Operating mode

UART supports multiple working modes: synchronous half-duplex mode and asynchronous full-duplex mode. Through the combination of UARTx_SCON.SM0 and UARTx_SCON.SM1, various working modes can be configured.

15.4.1.1 Mode0~Mode3 function comparison

Configure UARTx_SCON.SM to select different transmission modes: Mode0~Mode3. The main function pairs of these four working modes are shown in the table below:

Table 15-1 Mode0/1/2/3 Data Structure

Operating mode		Transmission bit width	Data composition	Baud rate
Mode0	synchronous mode Half duplex	8bit	Data(8bit)	$BaudRate = \frac{f_{PCLK}}{12}$
Mode1	Asynchronous mode Full duplex	10bit	Start (1bit) + Data(8bit) + Stop(1bit)	$BaudRate = \frac{(DBAUD + 1)f_{PCLK}}{32 * (65536 - TM)}$
Mode2	Asynchronous mode Full duplex	11bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop(1bit)	$BaudRate = \frac{(DBAUD + 1)f_{PCLK}}{64}$
Mode3	Asynchronous mode Full duplex	11bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop(1bit)	$BaudRate = \frac{(DBAUD + 1)f_{PCLK}}{32 * (65536 - TM)}$

Note:

- Mode0 can only be used as a master to send UART synchronous shift clock, and cannot be used as a slave to receive UART synchronous shift clock input from the outside.
- f_{PCLK} represents the frequency of the current PCLK.
- The definition of DBAUD is detailed in UARTx_SCON.
- TM is the count value of TIMER. Note that TIMER must be configured as a 16-bit auto-reload mode, and both the count register and the reload register must be written with the TM value.

15.4.1.2 Mode0 (synchronous mode, half-duplex)

When working in Mode0, UART works in synchronous mode, and its baud rate is 1/12 of the fixed PCLK clock. UART receives data through RXD input, UART sends data through RXD output, and RXD is the input and output port at this time. The UART synchronous shift clock is output by TXD, and TXD is an output port at this time. Note that this mode can only be used as a master to send a synchronous shift clock, and cannot be used as a slave to receive the clock from the outside. In this mode, the transmitted data bit width can only be 8 bits, without start bit and stop bit.

Clear UARTx_SCON.SM0 and UARTx_SCON.SM1 to enter Mode0 working mode.

When sending data, clear the UARTx_SCON.REN bit and write the data to the UARTx_SBUF register. At this time, the sending data will be output from RXD (low bit first, high bit later), and the synchronous shift clock will be output from TXD.

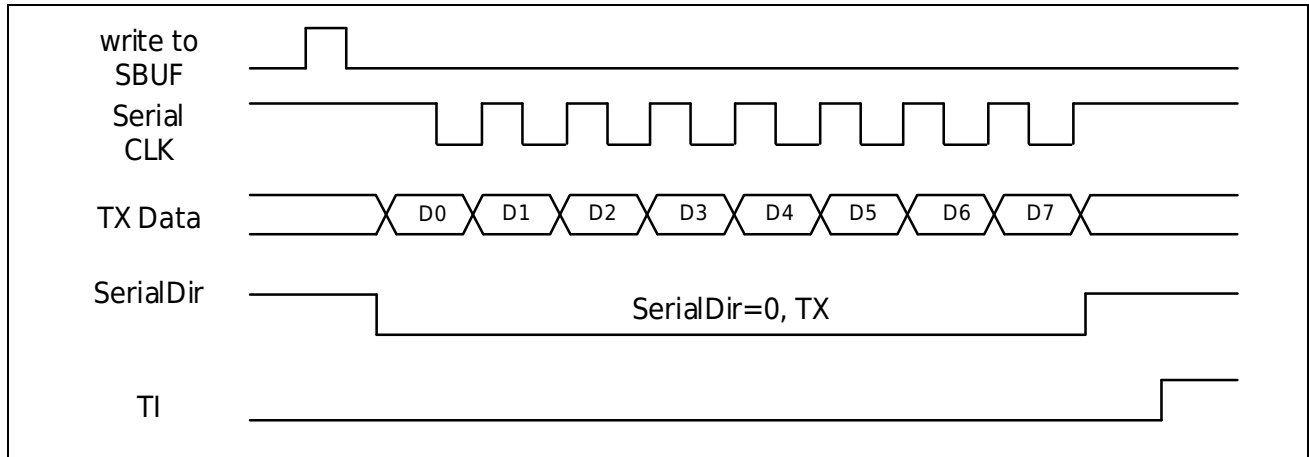


Figure 15-2 Mode0 sending data

When receiving data, set the UARTx_SCON.REN bit and clear the UARTx_ISR.RI bit. When the reception is finished, the data can be read from the UARTx_SBUF register. At this time, the received data is input from RXD (low bit first, high bit later), and the synchronous shift clock is output from TXD.

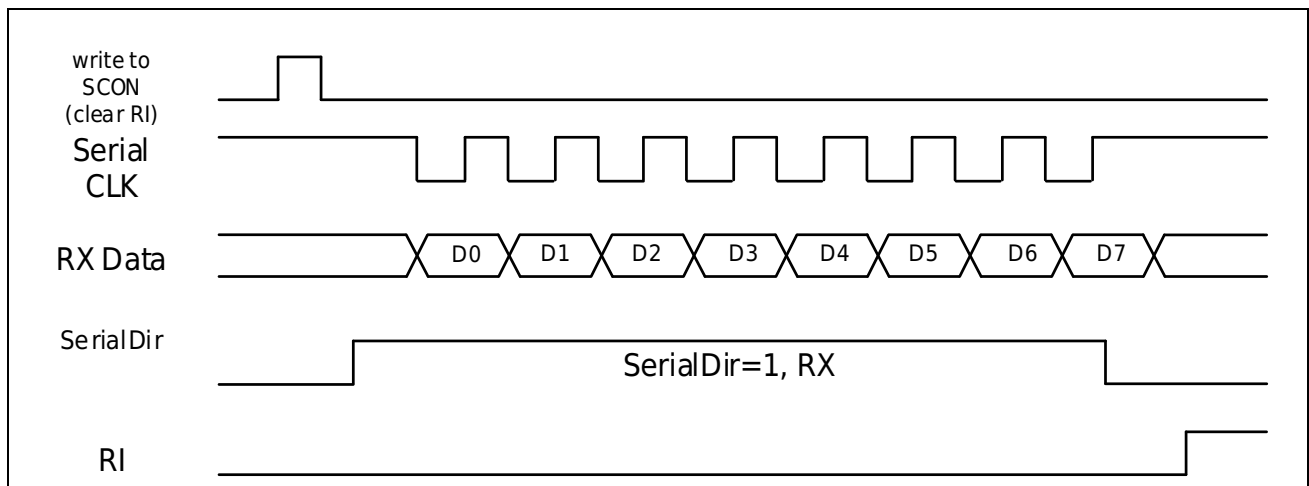


Figure 15-3 Mode0 receiving data

15.4.1.3 Mode1 (asynchronous mode, full duplex)

When working in Mode1, the sending data is sent through TXD, and the receiving data is received through RXD. The data consists of 10 bits: start bit "0", followed by 8 data bits (low bit first, high bit later), and finally end bit "1".

In this mode, the baud rate is generated by the timer module and is programmable. The baud rate of UART0 is generated by TIMER0, and the baud rate of UART1 is generated by TIMER1.

Clear UARTx_SCON.SM0 to 0 and set UARTx_SCON.SM1 to 1 to enter Mode1 working mode.

When sending data, it has nothing to do with the value of UARTx_SCON.REN, write the sent data into the UARTx_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

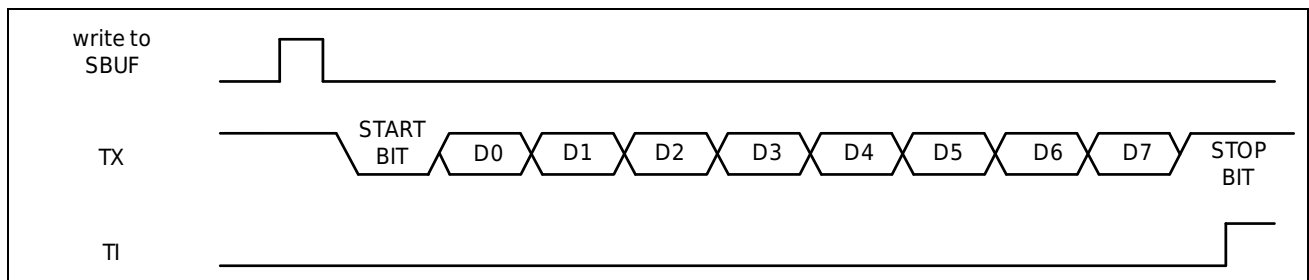


Figure 15-4 Mode1 sending data

When receiving data, you need to set the UARTx_SCON.REN bit to 1, and clear the UARTx_ISR.RI bit to 0. Start to receive the data on RXD (low bit first, high bit later), when the reception is completed, it can be read from the UARTx_SBUF register.

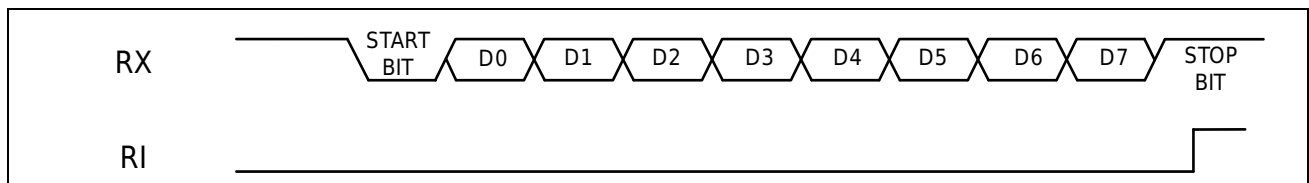


Figure 15-5 Mode1 receiving data

15.4.1.4 Mode2 (asynchronous mode, full duplex)

When working in Mode2, the sending data is sent through TXD, and the receiving data is received through RXD. The data consists of 11 bits: start bit "0", followed by 8 data bits, 1 TB8 bit and stop bit. The extra TB8 bit is used in a multi-machine communication environment. When TB8=1, it indicates that the address frame is received; when TB8=0, it indicates that the received data frame. When multi-machine communication is not required, this bit can also be used as a parity bit.

In this mode, the baud rate can be generated independently without an external TIMER.

Set UARTx_SCON.SM0 to 1 and UARTx_SCON.SM1 to 0 to enter Mode2 working mode.

When sending data, it has nothing to do with the value of UARTx_SCON.REN, and write the sent data into the UARTx_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

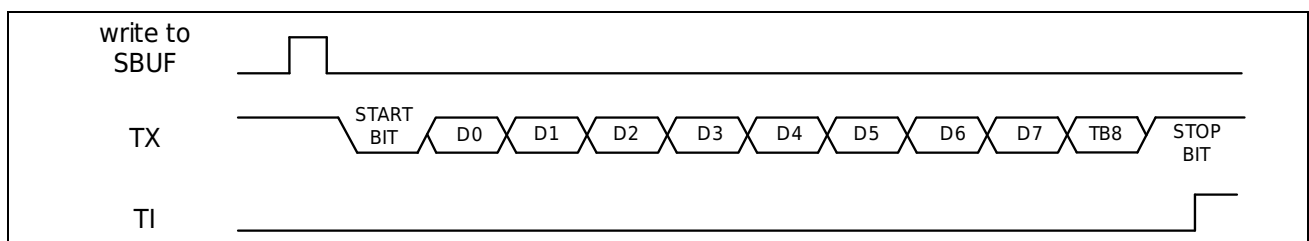


Figure 15-6 Mode2 sending data

When receiving data, you need to set the UARTx_SCON.REN bit to 1, and clear the UARTx_ISR.RI bit to 0. Start to receive the data on RXD (low bit first, high bit later), when the reception is completed, it can be read from the UARTx_SBUF register.

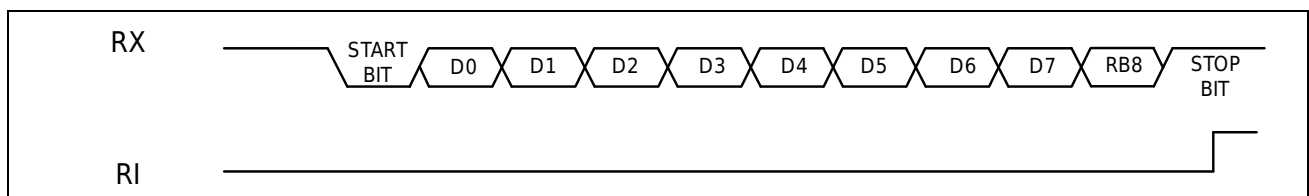


Figure 15-7 Mode2 receiving data

15.4.1.5 Mode3 (asynchronous mode, full duplex)

The data format, transmission timing and operation mode of Mode3 are the same as Mode2. The only difference is that the baud rate of Mode3 is generated by TIMER instead of independently generated by the device itself like Mode2. Mode3 is programmable, and the baud rate generation method is the same as that of Mode1. In this product, the baud rate of UART0 is generated by TIMER0, and the baud rate of UART1 is generated by TIMER1.

Set UARTx_SCON.SM0 to 1 and UARTx_SCON.SM1 to 1 to enter Mode3 working mode.

When sending data, it has nothing to do with the value of UARTx_SCON.REN, and write the sent data into the UARTx_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

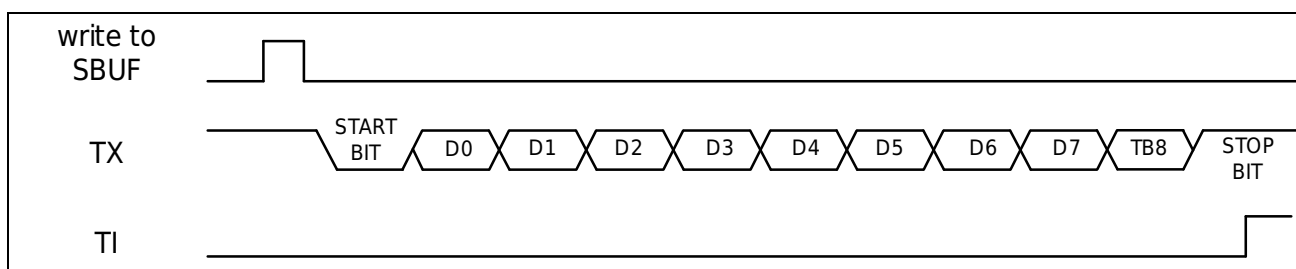


Figure 15-8 Mode3 Send Data

When receiving data, you need to set the UARTx_SCON.REN bit to 1, and clear the UARTx_ISR.RI bit to 0. Start to receive the data on RXD (low bit first, high bit later), when the reception is completed, it can be read from the UARTx_SBUF register.

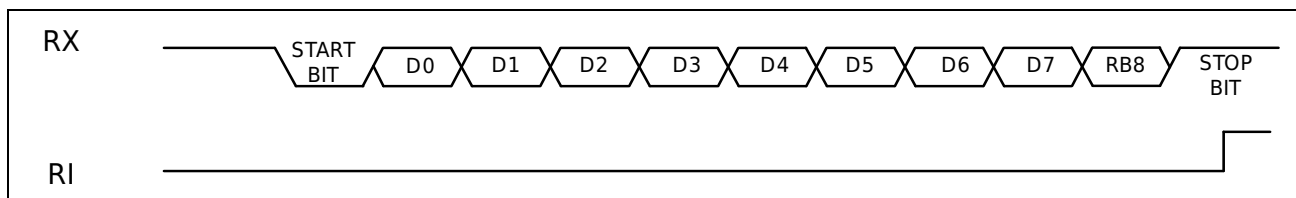


Figure 15-9 Mode3 Receive Data

15.4.2 Baud rate generation

Mode0~Mode3 are different, as shown below for details:

Mode0 baud rate generation formula: $BaudRate = \frac{f_{PCLK}}{12}$

Mode1 baud rate generation formula: $BaudRate = \frac{(DBAUD+1)f_{PCLK}}{32*(65536-TM)}$

Mode2 baud rate generation formula: $BaudRate = \frac{(DBAUD+1)f_{PCLK}}{64}$

Mode3 baud rate generation formula: $BaudRate = \frac{(DBAUD+1)f_{PCLK}}{32*(65536-TM)}$

Note:

- f_{PCLK} represents the frequency of the current PCLK.
- The definition of DBAUD is detailed in UARTx_SCON.
- TM is the count value of TIMER. Note that TIMER must be configured as a 16 -bit auto-reload mode, and both the count register and the reload register must be written with the TM value.

15.4.2.1 Mode1/Mode3 baud rate setting example

Baud rate	PCLK = 1 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	26	2403.85	0.16%	13	2403.85	0.16%
4800	13	4807.69	0.16%	7	4464.29	-6.99%
9600	7	8928.57	-6.99%	3	10416.67	8.51%
19200	3	20833.33	8.51%	2	15625.00	-18.62%
38400	2	31250.00	-18.62%	1	31250.00	-18.62%
57600	1	62500.00	8.51%	1	31250.00	-45.75%
76800	1	62500.00	-18.62%	0	#DIV/0!	#DIV/0!
115200	1	62500.00	-45.75%	0	#DIV/0!	#DIV/0!

Baud rate	PCLK = 4 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	104	2403.85	0.16%	52	2403.85	0.16%
4800	52	4807.69	0.16%	26	4807.69	0.16%
9600	26	9615.38	0.16%	13	9615.38	0.16%
19200	13	19230.77	0.16%	7	17857.14	-6.99%
38400	7	35714.29	-6.99%	3	41666.67	8.51%
57600	4	62500.00	8.51%	2	62500.00	8.51%
76800	3	83333.33	8.51%	2	62500.00	-18.62%
115200	2	125000.00	8.51%	1	125000.00	8.51%

Baud rate	PCLK = 10 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	260	2403.85	0.16%	130	2403.85	0.16%
4800	130	4807.69	0.16%	65	4807.69	0.16%
9600	65	9615.38	0.16%	33	9469.70	-1.36%
19200	33	18939.39	-1.36%	16	19531.25	1.73%
38400	16	39062.50	1.73%	8	39062.50	1.73%
57600	11	56818.18	-1.36%	5	62500.00	8.51%
76800	8	78125.00	1.73%	4	78125.00	1.73%
115200	5	125000.00	8.51%	3	104166.67	-9.58%

Baud rate	PCLK = 14 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	365	2397.26	-0.11%	182	2403.85	0.16%
4800	182	4807.69	0.16%	91	4807.69	0.16%
9600	91	9615.38	0.16%	46	9510.87	-0.93%
19200	46	19021.74	-0.93%	23	19021.74	-0.93%
38400	23	38043.48	-0.93%	11	39772.73	3.57%
57600	15	58333.33	1.27%	8	54687.50	-5.06%
76800	11	79545.45	3.57%	6	72916.67	-5.06%
115200	8	109375.00	-5.06%	4	109375.00	-5.06%

Baud rate	PCLK = 20 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	521	2399.23	-0.03%	260	2403.85	0.16%
4800	260	4807.69	0.16%	130	4807.69	0.16%
9600	130	9615.38	0.16%	65	9615.38	0.16%
19200	65	19230.77	0.16%	33	18939.39	-1.36%
38400	33	37878.79	-1.36%	16	39062.50	1.73%
57600	22	56818.18	-1.36%	11	56818.18	-1.36%
76800	16	78125.00	1.73%	8	78125.00	1.73%
115200	11	113636.36	-1.36%	5	125000.00	8.51%

Baud rate	PCLK = 24 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	625	2400.00	0.00%	313	2396.17	-0.16%
4800	313	4792.33	-0.16%	156	4807.69	0.16%
9600	156	9615.38	0.16%	78	9615.38	0.16%
19200	78	19230.77	0.16%	39	19230.77	0.16%
38400	39	38461.54	0.16%	20	37500.00	-2.34%
57600	26	57692.31	0.16%	13	57692.31	0.16%
76800	20	75000.00	-2.34%	10	75000.00	-2.34%
115200	13	115384.62	0.16%	7	107142.86	-6.99%

Baud rate	PCLK = 2 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	52	2403.85	0.16%	26	2403.85	0.16%
4800	26	4807.69	0.16%	13	4807.69	0.16%
9600	13	9615.38	0.16%	7	8928.57	-6.99%
19200	7	17857.14	-6.99%	3	20833.33	8.51%
38400	3	41666.67	8.51%	2	31250.00	-18.62%
57600	2	62500.00	8.51%	1	62500.00	8.51%
76800	2	62500.00	-18.62%	1	62500.00	-18.62%
115200	1	125000.00	8.51%	1	62500.00	-45.75%

Baud rate	PCLK = 8 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	208	2403.85	0.16%	104	2403.85	0.16%
4800	104	4807.69	0.16%	52	4807.69	0.16%
9600	52	9615.38	0.16%	26	9615.38	0.16%
19200	26	19230.77	0.16%	13	19230.77	0.16%
38400	13	38461.54	0.16%	7	35714.29	-6.99%
57600	9	55555.56	-3.55%	4	62500.00	8.51%
76800	7	71428.57	-6.99%	3	83333.33	8.51%
115200	4	125000.00	8.51%	2	125000.00	8.51%

Baud rate	PCLK = 11.0592 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	288	2400.00	0.00%	144	2400.00	0.00%
4800	144	4800.00	0.00%	72	4800.00	0.00%
9600	72	9600.00	0.00%	36	9600.00	0.00%
19200	36	19200.00	0.00%	18	19200.00	0.00%
38400	18	38400.00	0.00%	9	38400.00	0.00%
57600	12	57600.00	0.00%	6	57600.00	0.00%
76800	9	76800.00	0.00%	5	69120.00	-10.00%
115200	6	115200.00	0.00%	3	115200.00	0.00%

Baud rate	PCLK = 16 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	417	2398.08	-0.08%	208	2403.85	0.16%
4800	208	4807.69	0.16%	104	4807.69	0.16%
9600	104	9615.38	0.16%	52	9615.38	0.16%
19200	52	19230.77	0.16%	26	19230.77	0.16%
38400	26	38461.54	0.16%	13	38461.54	0.16%
57600	17	58823.53	2.12%	9	55555.56	-3.55%
76800	13	76923.08	0.16%	7	71428.57	-6.99%
115200	9	111111.11	-3.55%	4	125000.00	8.51%

Baud rate	PCLK = 32 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	833	2400.96	0.04%	417	2398.08	-0.08%
4800	417	4796.16	-0.08%	208	4807.69	0.16%
9600	208	9615.38	0.16%	104	9615.38	0.16%
19200	104	19230.77	0.16%	52	19230.77	0.16%
38400	52	38461.54	0.16%	26	38461.54	0.16%
57600	35	57142.86	-0.79%	17	58823.53	2.12%
76800	26	76923.08	0.16%	13	76923.08	0.16%
115200	17	117647.06	2.12%	9	111111.11	-3.55%

Baud rate	PCLK = 22.12 MHz					
	Dual baud rate			Single baud rate		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	576	2400.17	0.01%	288	2400.17	0.01%
4800	288	4800.35	0.01%	144	4800.35	0.01%
9600	144	9600.69	0.01%	72	9600.69	0.01%
19200	72	19201.39	0.01%	36	19201.39	0.01%
38400	36	38402.78	0.01%	18	38402.78	0.01%
57600	24	57604.17	0.01%	12	57604.17	0.01%
76800	18	76805.56	0.01%	9	76805.56	0.01%
115200	12	115208.33	0.01%	6	115208.33	0.01%

15.5 Frame error detection

When working in Mode1/2/3, UART has a frame error detection function, and the hardware will automatically detect whether the received frame data has a valid Stop bit. Stop bit as expected, resulting in synchronization failure or excessive noise, then UARTx_ISR.FE is set to 1. The UARTx_ISR.FE bit is set to 1 by hardware and cleared to 0 by software. If the software does not clear it to 0 in time, the UARTx_ISR.FE flag will not be cleared to 0 even if the subsequent received data has a valid Stop bit.

15.6 Multi-machine communication

Mode2/3 has multi-machine communication function, for which a bit TB8/RB8 is added to its frame format. Set UARTx_SCON.SM2 to "1" to enable the multi-device communication bit.

When the multi-machine communication bit is turned on, when sending data, the master can use UARTx_SCON.TB8 to distinguish whether the current frame is an address frame (UARTx_SCON.TB8=1) or a data frame (UARTx_SCON.TB8=0). When receiving data, the slave will ignore the current received frame whose RB8 bit (9th bit) is "0".

- When it is a data frame, the frame data will not be stored in the UARTx_SBUF register of the slave, and the slave will not generate a receive interrupt.
- When it is an address frame, since the automatic address recognition function in the multi-machine communication has been turned on, the slave can detect whether the received address matches its own address.
 - If the addresses match, the slave will set UARTx_SCON.RB8 to "1" and UARTx_ISR.RI to "1". UARTx_SCON.RB8=1 and UARTx_ISR.RI=1, the slave software clears the UARTx_SCON.SM2 bit to "0" and accepts the data frame.
 - If the address does not match, it means that the master is not addressing the slave, the slave hardware keeps UARTx_SCON.RB8 and UARTx_ISR.RI as "0", the software keeps UARTx_SCON.SM2 as "1", and continues to be in the address monitoring state.

Note: If necessary, the multi-machine communication bit can also be turned on in Mode1, and the TB8 bit is replaced by the stop bit. When the slave receives a matching address frame and a valid stop bit, UARTx_ISR.RI will be set to "1".

15.7 Automatic Address Recognition

When the multi-machine communication bit is turned on (UARTx_SCON.SM2 is set to "1"), the automatic address recognition function will also be turned on. This function is implemented by hardware, so that the slave can detect each address frame received, and if the address matches the address of the slave, the receiving end will give the UARTx_ISR.RI receiving flag. If the addresses do not match, the receiving end will not give any acceptance flag.

Slave Address Configuration

The slave address consists of the broadcast address and the user configuration address:

- Broadcast Address: UART_SADDR | UART_SADEN
- User Configuration Address: UART_SADDR & UART_SADEN

When an address frame is received, if the address received from the bus, after being ANDed with the value of UART_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: UART_SADDR=0x55, UART_SADEN=0xF0, then the slave can match addresses 0xF5, 0xF7, 0xFD, 0xFF (broadcast address) and 0x50~0x5F (user-configured address).

15.8 Transceiver buffer

15.8.1 Receive buffer

The general UART (UART0/1) receiving end has a frame length (8/9bits) receiving buffer, that is to say, when a frame of data is received, the data in the receiving buffer will be kept until the Stop of the next frame of data. After the bit is received, the receive buffer will be updated with a new frame of data.

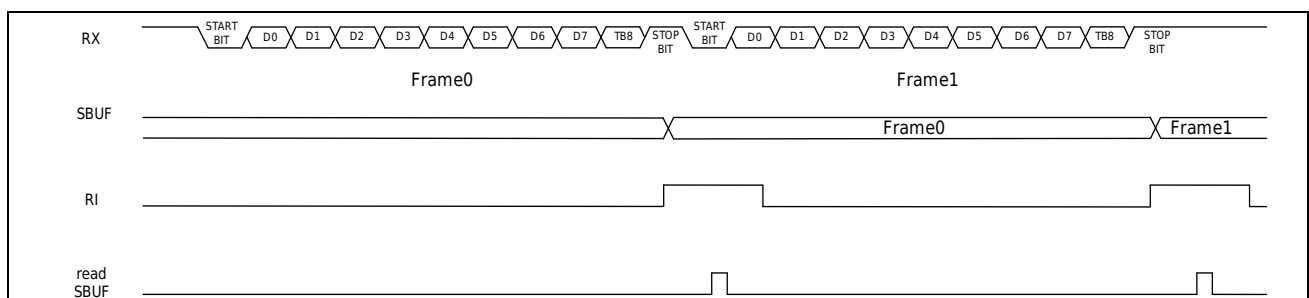


Figure 15-10 Receive Cache

15.8.2 Send cache

Universal UART (UART0/1) transmitters do not support transmit buffering. UARTx_SBUF register is filled during the process of sending data, it will destroy the data currently being sent. Software should avoid this operation.

15.9 Register

UART0 base address: 0x4000 0000

UART1 base address: 0x4000 0100

Register	Offset address	Description
UARTx_SBUF	0x00	Data register
UARTx_SCON	0x04	control register
UARTx_SADDR	0x08	Address register
UARTx_SADEN	0x0C	Address mask register
UARTx_ISR	0x10	interrupt flag register
UARTx_ICR	0x14	Interrupt flag bit clear register

15.9.1 Data Register (UARTx_SBUF)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SBUF							
R								RW							

Bit	Marking	Functional description
31:8	Reserved	
7:0	SBUF	When sending data, the data to be sent is written into this register; When receiving data, read the received data from this register; Note that the value read to this register is actually the value in RxBuffer, and the value written to this register is actually written to TXShifter.

15.9.2 Control Register (UARTx_SCON)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						DBAUD	Reserv ed	SM01	SM2	REN	TB8	RB8	TIEN	RIEN	
R						RW	R	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:10	Reserved	
9	DBAUD	Baud rate setting 0: single baud rate; 1: double baud rate
8	Reserved	
7:6	SM01	Working mode configuration 00: mode0; 01: mode1; 10: mode2; 11: mode3
5	SM2	Multi-machine communication enable control 0: Disable the multi-device communication function 1: Enable multi-machine communication function
4	REN	Receive Enable Control Mode0: 0: send, 1: receive Other: 0: send, 1: receive / send
3	TB8	TB8 bits to be sent when sending data
2	RB8	RB8 bit received when receiving data
1	TIEN	Transmit complete interrupt enable 0: Disable transmit complete interrupt 1: Enable transmit complete interrupt
0	RIEN	Receive complete interrupt enable 0: Disable receive complete interrupt 1: Enable receive complete interrupt

15.9.3 Address Register (UARTx_SADDR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SADDR							
R								RW							

Bit	Marking	Functional description
31:8	Reserved	
7:0	SADDR	Slave device address

15.9.4 Address Mask Register (UARTx_SADEN)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SADEN							
R								RW							

Bit	Marking	Functional description
31:8	Reserved	
7:0	SADEN	Slave device address mask

15.9.5 Flag Register (UARTx_ISR)

Offset address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FE	TI	RI
R													R	R	R

Bit	Symbol	Description
31:3	Reserved	
2	FE	Received frame error flag; set by hardware, cleared by software 1: FE interrupt flag is active 0: FE interrupt flag is invalid
1	TI	Transmit complete interrupt flag; set by hardware, cleared by software 1: TI interrupt flag is valid 0: TI interrupt flag is invalid
0	RI	Receive completion interrupt flag; set by hardware, cleared by software 1: RI interrupt flag is valid 0: RI interrupt flag is invalid

15.9.6 Flag Clear Register (UARTx_ICR)

Offset address: 0x14

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FE CLR	TI CLR	RI CLR
R													W	W	W

Reset value: 0x0000 0000

Bit	Marking	Functional description
31:3	RESERVED	
2	FECLR	Receive frame error flag cleared Write 0 to clear, write 1 to be invalid
1	TICLR	Transmit completed interrupt flag cleared Write 0 to clear, write 1 to be invalid
0	RICLR	Receive completion interrupt flag cleared Write 0 to clear, write 1 to be invalid

16 Low Power Synchronous Asynchronous Transceiver (LPUART)

16.1 Overview

This product has a LPUART module, LPUART contains all the necessary hardware support to enable asynchronous serial communication with minimum power consumption; the baud rate can be generated by the external TIMER2 or by the internal logic of the module.

In order to support low power consumption applications, LPUART adds a SCLK clock in addition to the original PCLK clock. The internal register configuration logic of the LPUART module works in the PCLK clock domain, and the data transceiver logic works in the SCLK clock domain. When the system enters the low power consumption mode, turn off the high-frequency PCLK clock and turn on the low-frequency SCLK clock, and the LPUART can still perform normal data transmission and reception.

SCLK clock source can be selected: PCLK, external low-speed clock (XTL), internal low-speed clock (RCL). When LPMODE=1, the SCLK clock also supports 1/2/4/8/16/32/64/128 times prescaler.

Note that when LPMODE=0, LPUART receives the Toggle output signal of the TIMER2 clock instead of the Overflow signal, so the Toggle output of TIMER2 must be enabled.

16.2 Structural block diagram

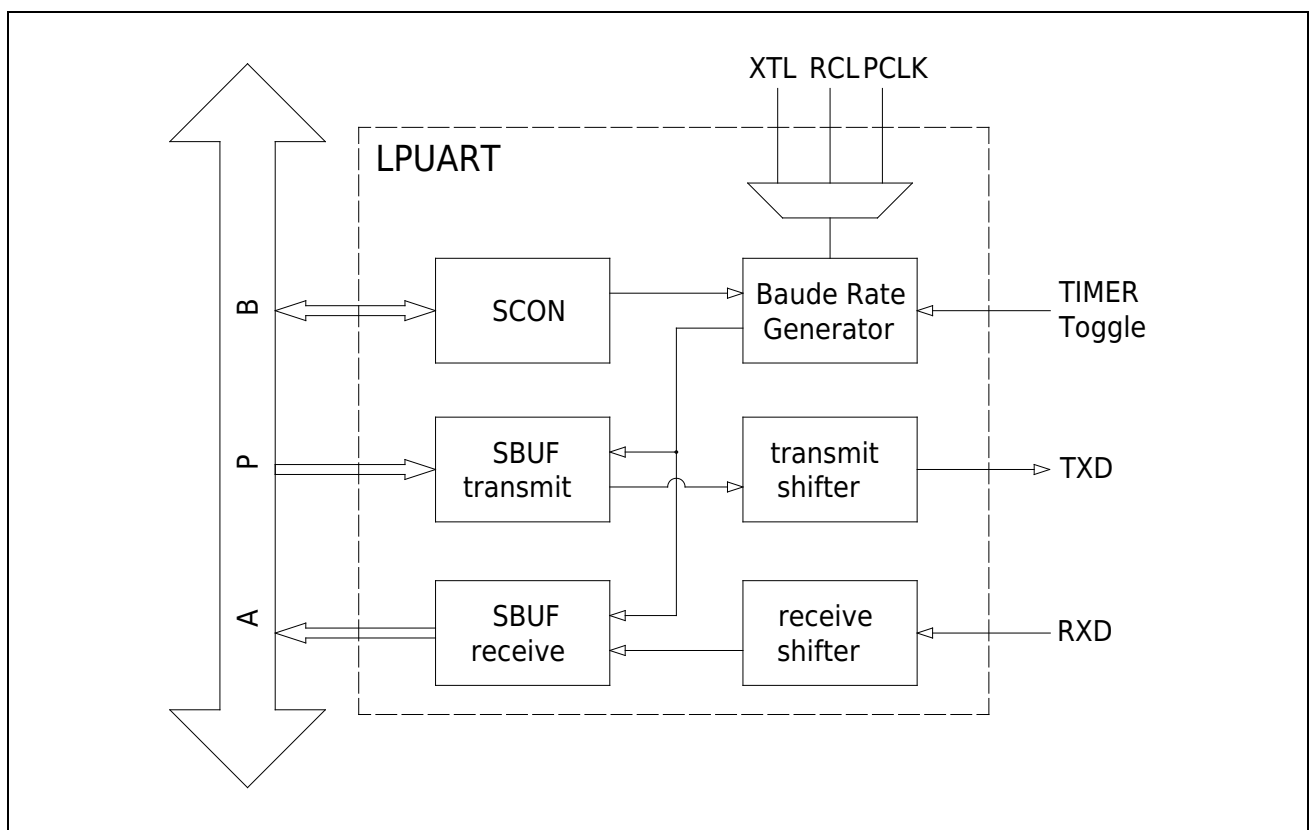


Figure 16-1 LPUART Block Diagram

16.3 Main characteristics

LPUART module supports the following basic functions:

- Configuration clock PCLK
- Transmission clock SCLK (SCLK can choose XTL, RCL and PCLK)
- Send and receive data in system low power mode
- Full-duplex transmission, half-duplex transmission
- Programmable serial communication function
 - Two character lengths: 8 bits, 9 bits
 - Mode0/1/2/3 four transmission modes
- 16-bit baud rate counter
- Multi-machine communication
- Automatic Address Recognition

16.4 Functional description

16.4.1 Configuration Clock and Transmit Clock

The LPUART module has two clocks: configuration clock PCLK and transfer clock SCLK.

- configure clock
The configuration clock is used for the system APB bus to configure the registers of the LPUART module, and it is fixed as PCLK.
- Transmission clock
The transmission clock is used for LPUART data transceiver logic work, and XTL, RCL and PCLK can be selected. When SCLK is XTL or RCL and the system enters DeepSleep, LPUART can still send and receive data normally.

16.4.2 Operating mode

LPUART supports multiple working modes: synchronous half-duplex mode and asynchronous full-duplex mode. By configuring the value of LPUARTx_SCON.SM, the required working mode can be obtained.

LPUART adds a LPMODE control bit to UART. When this bit is "1", only Mode1/3 working mode is supported, and the baud rate generation method will also change.

16.4.2.1 Mode0~Mode3 function comparison

When LPMODE=0:

Configure LPUARTx_SCON.SM to choose 4 transmission modes, and the function comparison of each mode is as follows:

Table 16-1 Mode0/1/2/3 Data Structure

Operating mode		Transmission bit width	Data composition	Baud rate
Mode0	Synchronous mode Half duplex	8bit	Data(8bit)	$BaudRate = \frac{f_{SCLK}}{12}$
Mode1	Asynchronous mode Full duplex	10bit	Start (1bit) + Data(8bit) + Stop(1bit)	$BaudRate = \frac{(DBAUD + 1)f_{SCLK}}{32 * (65536 - TM)}$
Mode2	Asynchronous mode Full duplex	11bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop(1bit)	$BaudRate = \frac{(DBAUD + 1)f_{SCLK}}{64}$
Mode3	Asynchronous mode Full duplex	11bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop(1bit)	$BaudRate = \frac{(DBAUD + 1)f_{SCLK}}{32 * (65536 - TM)}$

When LPMODE=1:

Mode2	Asynchronous mode Full duplex	11bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop(1bit)	$BaudRate = \frac{f_{SCLK}}{PreScale * 4}$
-------	----------------------------------	-------	---	--

Note:

- Mode0 can only send LPUART synchronous shift clock as a master, and cannot accept externally input LPUART synchronous shift clock as a slave.
- f_{SCLK} represents the frequency of the current SCLK.
- LPUARTx_SCON for the definition of DBAUD.
- TM is the count value of TIMER. Note that TIMER must be configured as a 16 -bit auto-reload mode, and both the count register and the reload register must be written with the TM value.
- PreScale is the prescaler coefficient.

16.4.2.2 Mode0 (synchronous mode, half-duplex) data sending and receiving instructions

Set LPUART_SCON.SM to 0, LpUart works in Mode0, clock and data are input and output synchronously, and the baud rate is fixed at SCLK/12. The transmitted data width is fixed at 8 bits, without start and end bits. The receiving and sending of data are realized through the RXD pin; the synchronous shift clock is output from the TXD pin. Note that the TXD pin does not accept an input clock.

When LPMODE=1, Mode0 working mode is not supported.

When sending data, set LPUART_SCON.REN to 0, and write the data to be sent into the SBUF register. At this time, the sending data will be output from RXD (low bit first, high bit later), and the synchronous shift clock will be output from TXD.

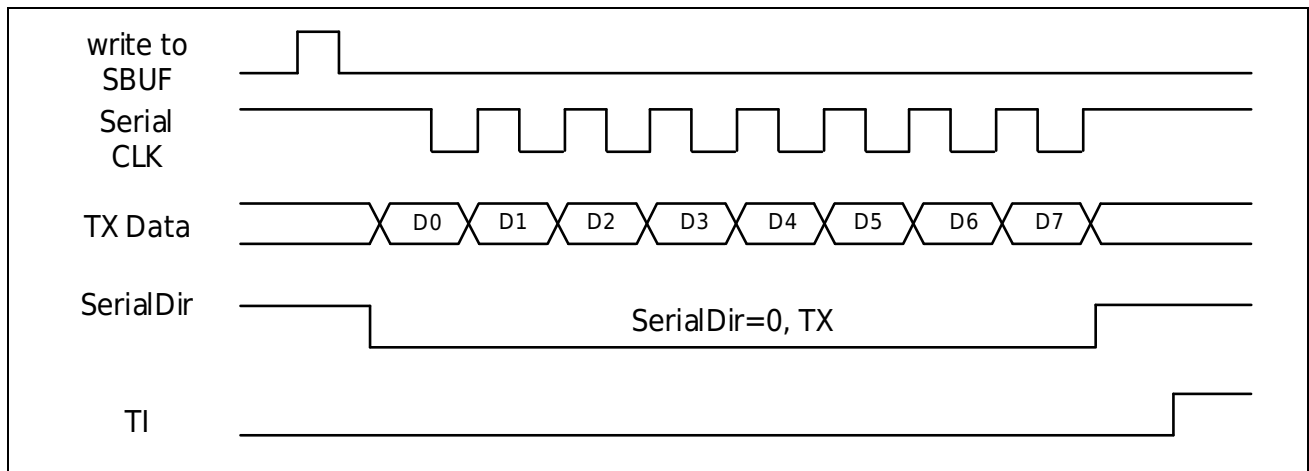


Figure 16-2 Mode0 sending data

When receiving data, set LPUART_SCON.REN to 1 and clear the LPUART_ISR.RI bit. When the reception is finished, the data can be read from the LPUART_SBUF register. At this time, the received data is input from RXD (low bit first, high bit later), and the synchronous shift clock is output from TXD.

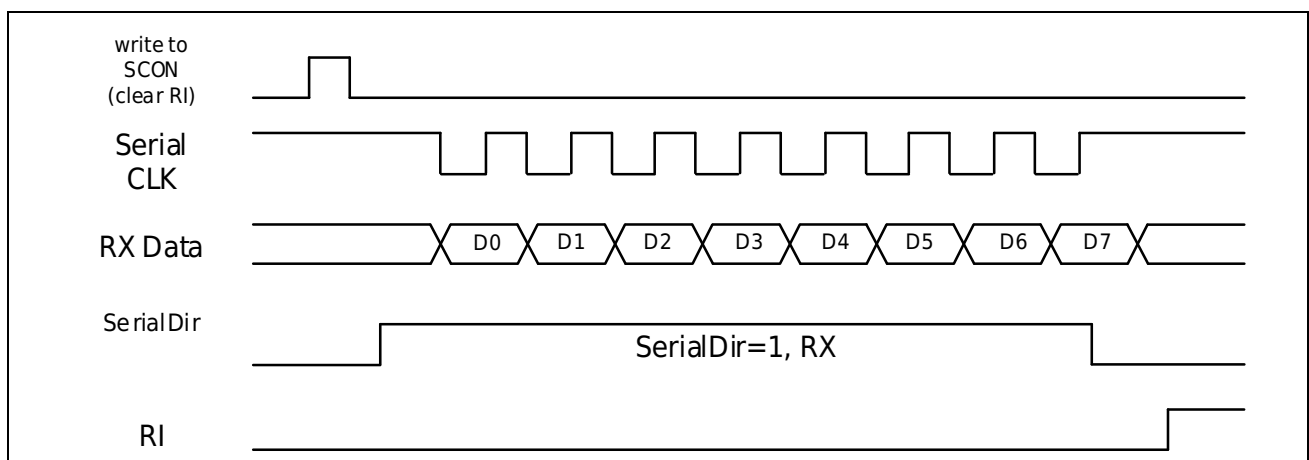


Figure 16-3 Mode0 receiving data

16.4.2.3 Mode1 (asynchronous mode, full duplex) data sending and receiving instructions

When working in Mode1, the sending data is sent through TXD, and the receiving data is received through RXD. The data consists of 10 bits: start bit "0", followed by 8 data bits (low bit first, high bit later), and finally end bit "1".

In this mode, the baud rate of LPUART is generated by the timer TIMER2 module and is programmable.

Clear LPUART_SCON.SM0 to 0 and set LPUART_SCON.SM1 to 1 to enter Mode1 working mode.

When LPMODE=1, the Mode1 working mode is supported, but the baud rate calculation method is changed, please refer to the baud rate generation chapter for details.

When sending data, it has nothing to do with the value of LPUART_SCON.REN, write the sent data into the LPUART_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

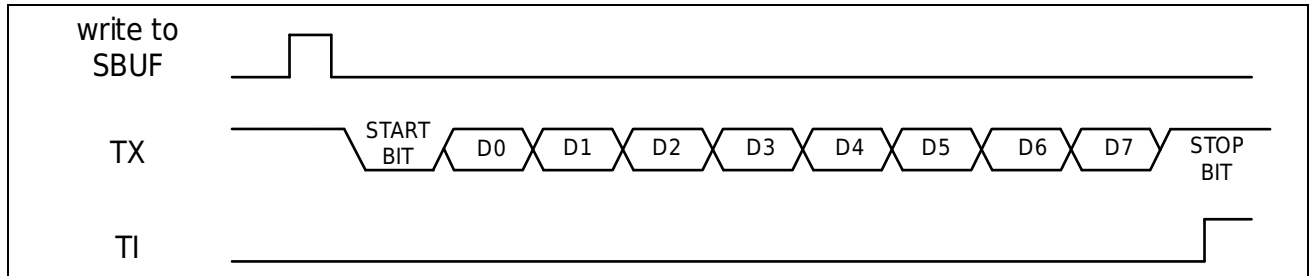


Figure 16-4 Mode1 sending data

When receiving data, you need to set the LPUART_SCON.REN bit to 1, and clear the LPUART_ISR.RI bit to 0. Start to receive the data on RXD (low bit first, high bit later), when the reception is completed, it can be read from the LPUART_SBUF register.

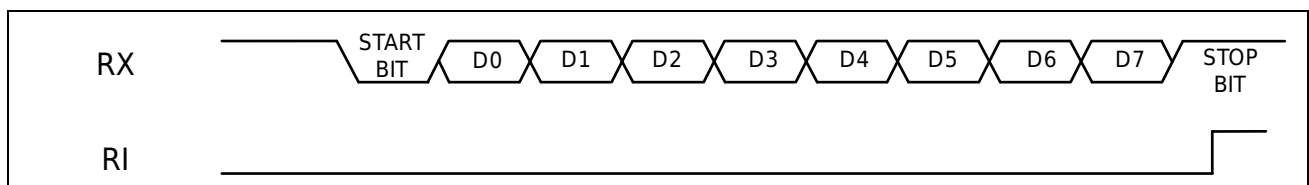


Figure 16-5 Mode1 receiving data

16.4.2.4 Mode2 (asynchronous mode, full duplex) data sending and receiving instructions

When working in Mode2, the sending data is sent through TXD, and the receiving data is received through RXD. The data consists of 11 bits: start bit "0", followed by 8 data bits, 1 TB8 bit and stop bit. The extra TB8 bit is used in a multi-machine communication environment. When TB8=1, it indicates that the address frame is received; when TB8=0, it indicates that the received data frame. When multi-machine communication is not required, this bit can also be used as a parity bit.

In this mode, the baud rate can be generated independently without an external TIMER.

Set LPUART_SCON.SM0 to 1 and clear LPUART_SCON.SM1 to enter Mode2.

When LPMODE=1, Mode2 working mode is not supported.

When sending data, it has nothing to do with the value of LPUART_SCON.REN, and write the sent data into the LPUART_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

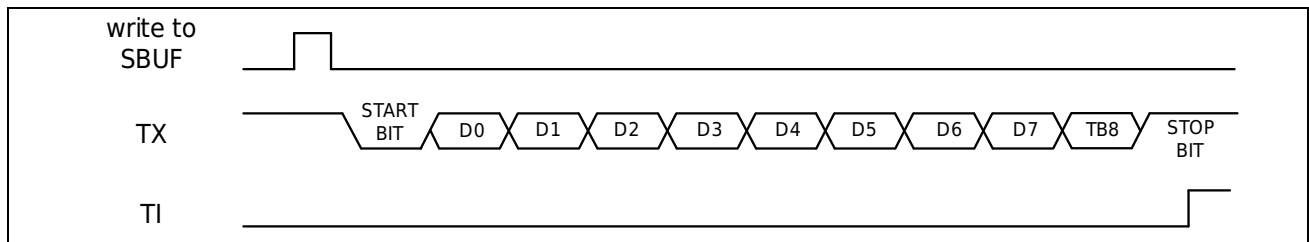


Figure 16-6 Mode2 sending data

When receiving data, you need to set the LPUART_SCON.REN bit to 1, and clear the LPUART_ISR.RI bit to 0. Start to receive the data on RXD (low bit first, high bit later), when the reception is completed, it can be read from the LPUART_SBUF register.

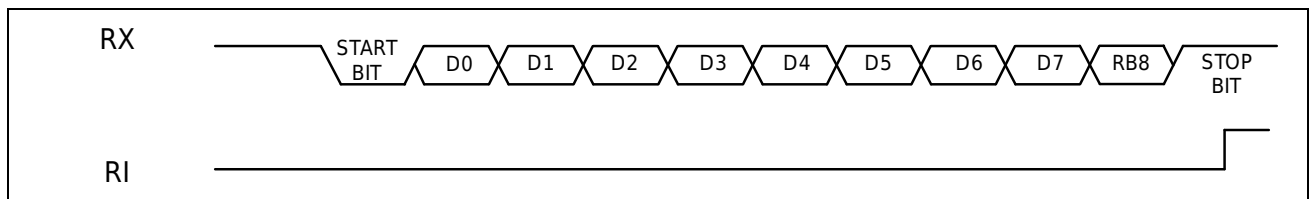


Figure 16-7 Mode2 receiving data

16.4.2.5 Mode3 (asynchronous mode, full duplex) data sending and receiving instructions

The data format, transmission timing and operation mode of Mode3 are the same as Mode2. The only difference is that the baud rate of Mode3 is generated by TIMER instead of independently generated by the device itself like Mode2. Mode3 is programmable, and the baud rate generation method is the same as that of Mode1.

Set LPUART_SCON.SM0 to 1 and LPUART_SCON.SM1 to 1 to enter Mode3 working mode.

When LPMODE=1, it supports Mode3 working mode. However, the calculation method of the baud rate has changed. For details, refer to the chapter on baud rate generation.

When sending data, it has nothing to do with the value of LPUART_SCON.REN, and write the sent data into the LPUART_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

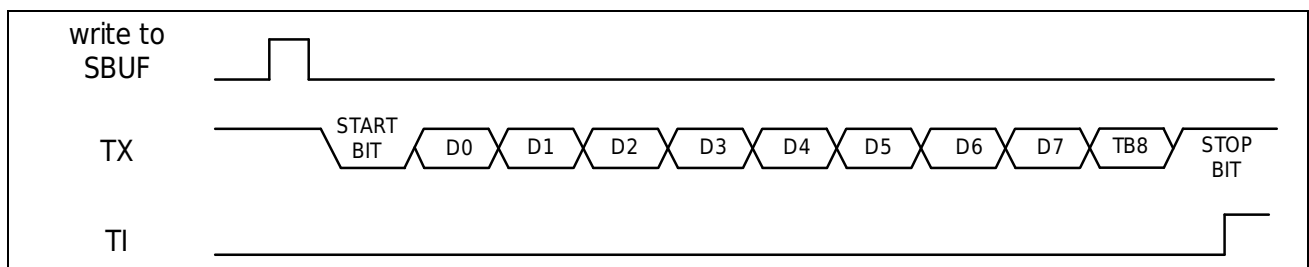


Figure 16-8 Mode3 Send Data

When receiving data, you need to set the LPUART_SCON.REN bit to 1, and clear the LPUART_ISR.RI bit to 0. Start to receive the data on RXD (low bit first, high bit later), when the reception is completed, it can be read from the LPUART_SBUF register.

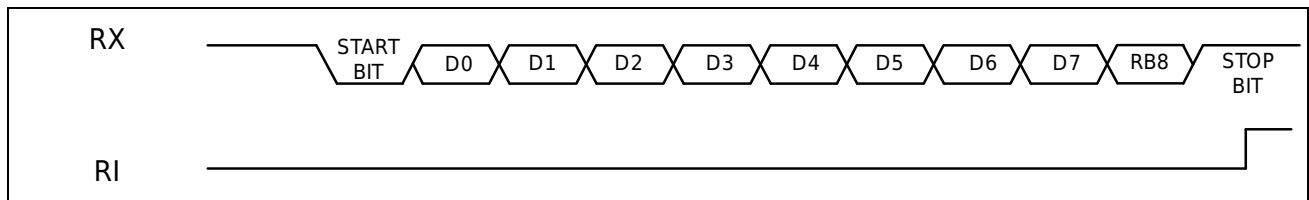


Figure 16-9 Mode3 Receive Data

16.4.3 Baud rate generation

LPMODE=0

The formula for generating the baud rate in Mode0~Mode3 is as follows:

$$\begin{aligned} \text{Mode0 baud rate generation formula: } BaudRate &= \frac{f_{SCLK}}{12} \\ \text{Mode1 baud rate generation formula: } BaudRate &= \frac{(DBAUD+1)f_{SCLK}}{32*(65536-TM)} \\ \text{Mode2 baud rate generation formula: } BaudRate &= \frac{(DBAUD+1)f_{SCLK}}{64} \\ \text{Mode3 baud rate generation formula: } BaudRate &= \frac{(DBAUD+1)f_{SCLK}}{32*(65536-TM)} \end{aligned}$$

LPMODE=1

When LPMODE=1, Mode0 and Mode2 are not supported.

$$\text{Mode1, Mode3 baud rate generation formula: } BaudRate = \frac{f_{SCLK}}{PreScale*4}$$

Note:

- f_{SCLK} represents the frequency of the current SCLK.
- LPUARTx_SCON for the definition of DBAUD.
- TM is the count value of TIMER. Note that TIMER must be configured as a 16-bit auto-reload mode, and both the count register and the reload register must be written with the TM value.
- PreScale is the prescaler coefficient.

16.5 Frame error detection

When working in Mode1/2/3, LPUART has a frame error detection function, and the hardware will automatically detect whether the received frame data has a valid Stop bit. The LPUART_ISR.FE bit is set to 1 if a valid Stop bit is not received within the expected time when receiving data, resulting in a synchronization failure or excessive noise. The LPUART_ISR.FE bit is set to 1 by hardware and cleared to 0 by software. If the software does not clear it to 0 in time, the subsequent received data will not clear the LPUART_ISR.FE flag even if it has a valid Stop bit.

16.6 Multi-machine communication

Mode2/3 has multi-machine communication function, for which a bit TB8/RB8 is added to its frame format. Set SCON.SM2 to "1" to enable the multi-machine communication bit.

When the multi-machine communication bit is turned on, when sending data, the master can use LPUART_SCON.TB8 to distinguish whether the current frame is an address frame (LPUART_SCON.TB8=1) or a data frame (LPUART_SCON.TB8=0).

- When it is a data frame, the frame data will not be stored in the LPUARTx_SBUF register of the slave, and the slave will not generate a receive interrupt.
- When it is an address frame, since the automatic address recognition function in the multi-machine communication has been turned on, the slave can detect whether the received address matches its own address.
 - If the address matches, the slave will set "1" to LPUARTx_ISR.RI and set "1" to LPUARTx_SCON.RB. LPUARTx_SCON.RB8=1 and LPUARTx_ISR.RI=1, the slave software clears the LPUARTx_SCON.SM2 bit to "0" and accepts the data frame.
 - If the address does not match, it means that the master is not addressing the slave, the slave hardware keeps LPUARTx_RB8 and LPUARTx_ISR.RI as "0", the software keeps LPUARTx_SCON.SM2 as "1", and the slave continues to be in the address monitoring state.

Note:

- If necessary, the multi-device communication bit can also be turned on in Mode1, and the TB8 bit is replaced by the stop bit. When the slave receives a matching address frame and a valid stop bit, LPUARTx_ISR.RC will be set to "1".

16.7 Automatic Address Recognition

When the multi-machine communication bit is turned on (LPUART_SCON.SM2 is set to "1"), the automatic address recognition function will also be turned on. This function is implemented by hardware, so that the slave can detect each address frame received, and if the address matches the slave address, the receiving end will give the LPUART_ISR.RI receiving flag. If the addresses do not match, the receiving end will not give any acceptance flag.

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).

16.8 Transceiver buffer

16.8.1 Receive buffer

The LPUART receiving end has a receiving buffer with a frame length (8/9bits), which means that when a frame of data is received, the data in the receiving buffer will be kept until the Stop bit of the next frame of data is received. The cache will be updated with a new frame of data.

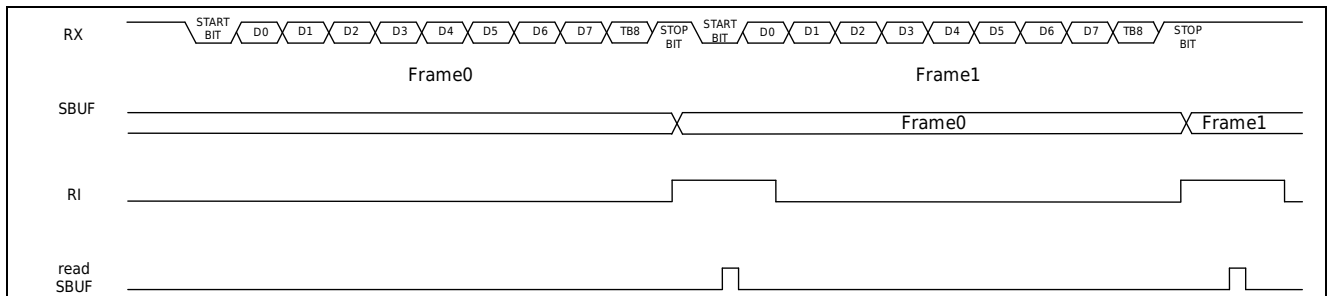


Figure 16-10 Receive Cache

16.8.2 Send cache

LPUART sender has a send buffer with a frame length (8/9bits). When the send shift register is sending the current frame data, the CPU can write the data to be sent in the next frame to the send buffer.

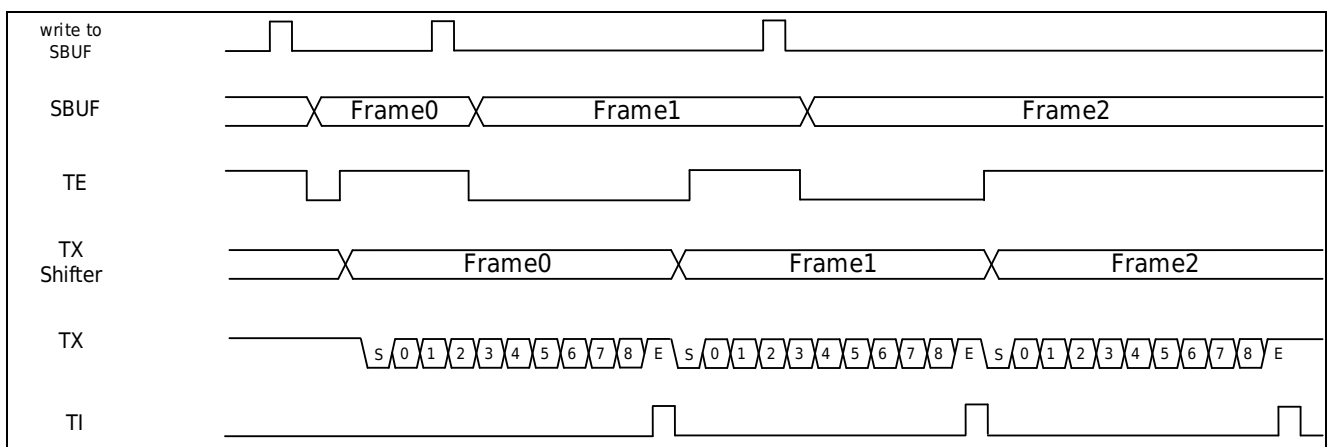


Figure 16-11 Send Cache

Among them, the register bit LPUART_ISR.TE is the send buffer empty flag bit. The LPUART module only contains one frame (8/9bits) send buffer, so when the LPUART_ISR.TE bit is "1", the hardware will automatically shield the software from writing to the LPUART_SBUF register until the LPUART_ISR.TE bit becomes "0". Before the software fills the sending data into the LPUART_SBUF register, it must judge the status of the LPUART_ISR.TE bit "0" and "1", otherwise the sending data will be lost.

16.9 Register

LPUART base address: 0x4000 0200

Register name	Offset address	Description
LPUART_SBUF	0x00	Data register
LPUART_SCON	0x04	control register
LPUART_SADDR	0x08	Address register
LPUART_SADEN	0x0C	Address mask register
LPUART_ISR	0x10	interrupt flag register
LPUART_ICR	0x14	Interrupt flag bit clear register

16.9.1 Data Register (LPUART_SBUF)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SBUF							
R								RW							

Bit	Marking	Functional description
31:8	Reserved	
7:0	SBUF	When sending data, write the sending data into this register; when receiving data, read it from this register after receiving the data. Note that the value read to this register is actually the value in RXBuffer, and the value written to this register is actually written to TXShifter.

16.9.2 Control Register (LPUART_SCON)

Offset address: 0x04

Reset value: 0x0000 E000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRS		SCLKSEL		LPMODE	DBAUD	TEEN	SM01		SM2	REN	TB8	RB8	TIEN	RIEN	
RW		RW					RW		RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:16	Reserved	
15:13	PRS	Transmission clock SCLK prescaler selection; 000: div128; 001: div64; 010: div32; 011: div16; 100: div8; 101: div4; 110: div2; 111: div1 PRS[2:0] is valid only when LPMODE=1; when LPMODE=0, PRS[2:0] will not prescale SCLK.
12:11	SCLKSEL	Transmission clock SCLK selection; 00: PCLK; 01: PCLK; 10: XTL; 11: RCL
10	LPMODE	Low power mode; 0: normal working mode; 1: Low power consumption mode
9	DBAUD	Double baud rate; 0: single baud rate; 1: double baud rate
8	TEEN	Transmit buffer empty interrupt enable; 0: disable; 1: enable
7:6	SM01	Operating mode; 00: mode0; 01: mode1; 10: mode2; 11: mode3
5	SM2	Multi-host communication; 0: disable, 1: enable
4	REN	Receive enable; mode0: 0: send, 1: receive Other: 0: send, 1: receive / send
3	TB8	Send TB8 bits
2	RB8	Receive RB8 bit
1	TIEN	Transmit complete interrupt enable; 0: disable; 1: enable
0	RIEN	Receive complete interrupt enable; 0: disable; 1: enable

16.9.3 Address Register (LPUART_SADDR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SADDR							
R								RW							

Bit	Marking	Functional description
31:8	Reserved	
7:0	SADDR	Slave Device Address Register

16.9.4 Address Mask Register (LPUART_SADEN)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SADEN							
R								RW							

Bit	Symbol	Functional description
31:8	Reserved	
7:0	SADEN	Slave Device Address Mask Register

16.9.5 Interrupt Flag Bit Register (LPUART_ISR)

Offset address: 0x10

Reset value: 0x0000 0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												TE	FE	TI	RI
R												R	R	R	R

Bit	Marking	Functional description
31:4	Reserved	
3	TE	Transmit buffer empty interrupt flag, set by hardware, cleared by hardware. Note: When the value of this bit is "0", the hardware automatically shields the software from writing to SBUF. 1: TE interrupt is valid 0: TE interrupt is invalid
2	FE	Receive frame error flag, set by hardware, cleared by software 1: FE interrupt is valid 0: FE interrupt is invalid
1	TI	Transmit complete interrupt flag, set by hardware, cleared by software 1: TI interrupt is valid 0: TI interrupt is invalid
0	RI	Receive completion interrupt flag, set by hardware, cleared by software 1: RI interrupt is valid 0: RI interrupt is invalid

16.9.6 Interrupt Flag Bit Clear Register (LPUART_ICR)

Offset address: 0x14

Reset value: 0x0000 0007

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FE CLR	TI CLR	RI CLR	
R												W	W	W	

Bit	Symbol	Description
31:3	Reserved	
2	FECLR	Clear the receive frame error flag; Write 0 to clear, write 1 to be invalid
1	TICLR	Clear the sending completion interrupt flag; Write 0 to clear, write 1 to be invalid
0	RICLR	Clear the receive completion interrupt flag; write 0 to clear, write 1 to be invalid

17 I2C -bus (I2C)

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.

I2C bus controller can meet various specifications of the I2C bus and support all transmission modes communicating with the I2C bus. The I2C logic can handle byte transfers autonomously. It can keep track of the serial transfer, and there is a status register (I2C_STAT) that can reflect the status of the I2C bus controller and the I2C bus.

17.2 Main characteristics

I2C controller supports the following features:

- Support four working modes of master sending/receiving and slave sending/receiving
- Support standard (100Kbps) / fast (400Kbps) / high speed (1Mbps) three working rates
- Support 7-bit addressing function
- Support noise filtering function
- Support broadcast address
- Support interrupt status query function

17.3 Protocol description

The I2C bus uses "SCL" (serial clock bus) and "SDA" (serial data bus) to connect devices to transfer information. The Master computer outputs a serial clock signal on the SCL line, and the data is transmitted on the SDA line, and each byte is transmitted (the highest bit MSB starts to be transmitted), followed by an acknowledge bit. One SCL clock pulse transfers one data bit.

17.3.1 Data transmission on the I2C bus

Usually the standard I2C transmission protocol consists of four parts: start (S) or repeated start signal (Sr), slave address and read and write bits, transmission data, and stop signal (P).

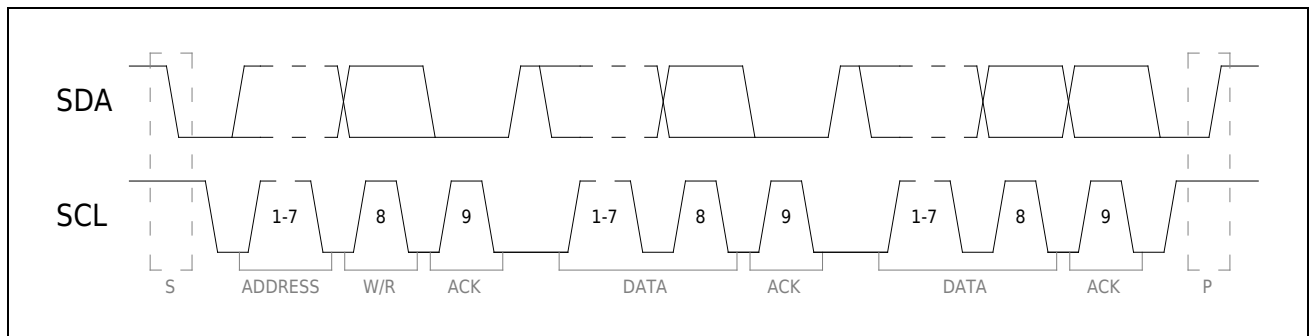


Figure 17-1 I2C transfer protocol

- Start signal, repeat start signal, stop signal

When the bus is in an idle state (the SCL and SDA lines are high at the same time), a signal from high to low appears on the SDA line, indicating that a start signal is generated on the bus. A repeated start signal occurs when there is no stop signal between two start signals. This method is used by a master to communicate with another slave or the same slave with a different transfer direction (for example: from writing to the device to reading from the device) without releasing the bus.

When the SCL line is high, a low-to-high signal appears on the SDA line, which is defined as a stop signal. The master sends a stop signal to the bus to end the data transfer.

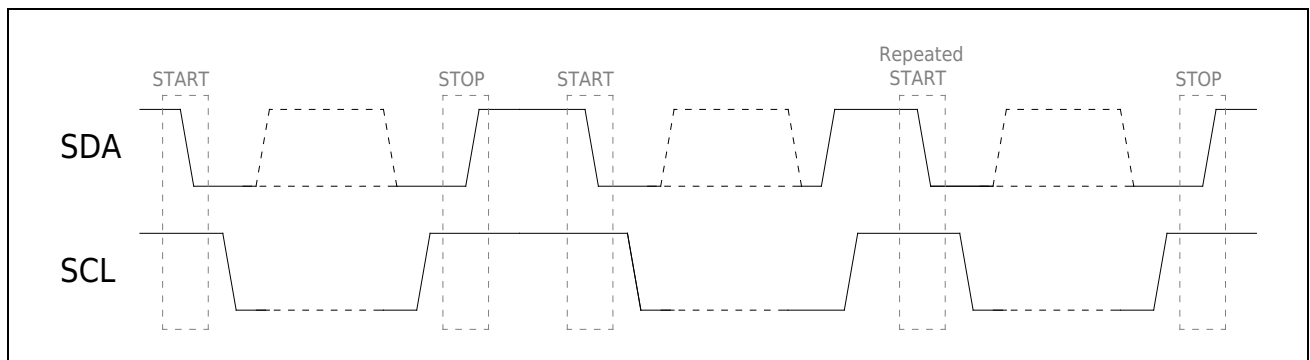


Figure 17-2 START and STOP conditions

- Slave address and read/write bits

When the start signal is generated, the Master immediately transmits the first byte of data: 7-bit slave address + read and write bits, the read and write bits control the data transmission direction of the slave (0: write; 1: read). A slave addressed by the master will acknowledge by pulling SDA low on the ninth SCL clock cycle.

- Transfer data

During data transfer, one SCL clock pulse transfers one data bit, and the SDA line can only change when SCL is low.

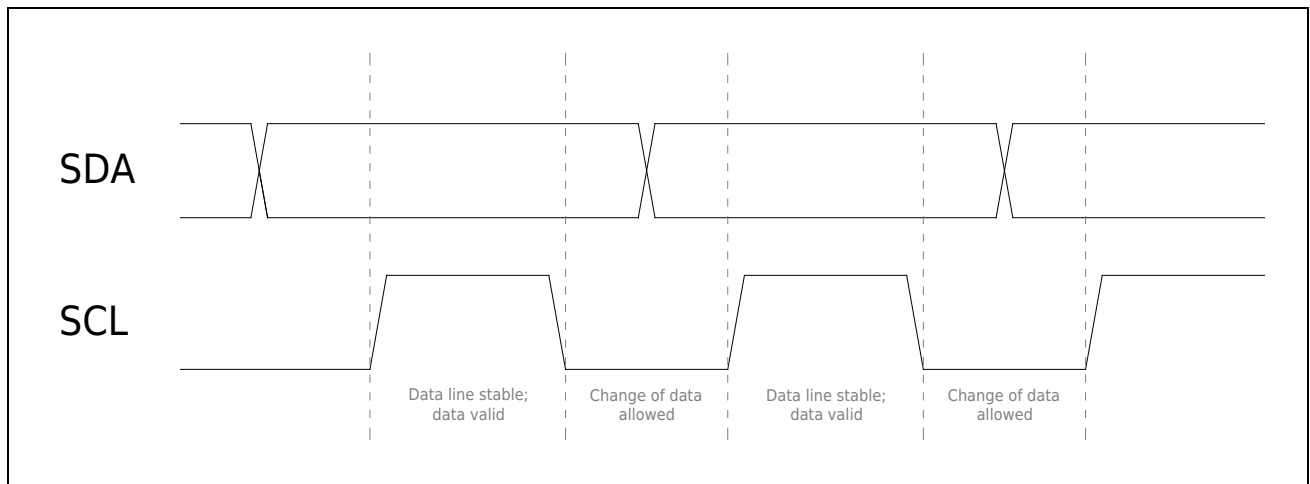


Figure 17-3 Bit transfer on the I2C bus

17.3.2 Acknowledgment on the I2C bus

Each byte transferred is followed by an acknowledge bit. By pulling the SDA line low, the receiver is allowed to echo the transmitter. ACK is a low-level signal. When the clock signal is high, SDA remains low to indicate that the receiving end has successfully received the data from the sending end.

(NACK) is generated on the slave, the Master can generate a stop signal to exit data transmission, or generate a repeated start signal to start a new round of data transmission. When the Master is used as a receiving device, a non-response signal (NACK) occurs, and the slave releases the SDA line, causing the Master to generate a stop signal or a repeated start signal.

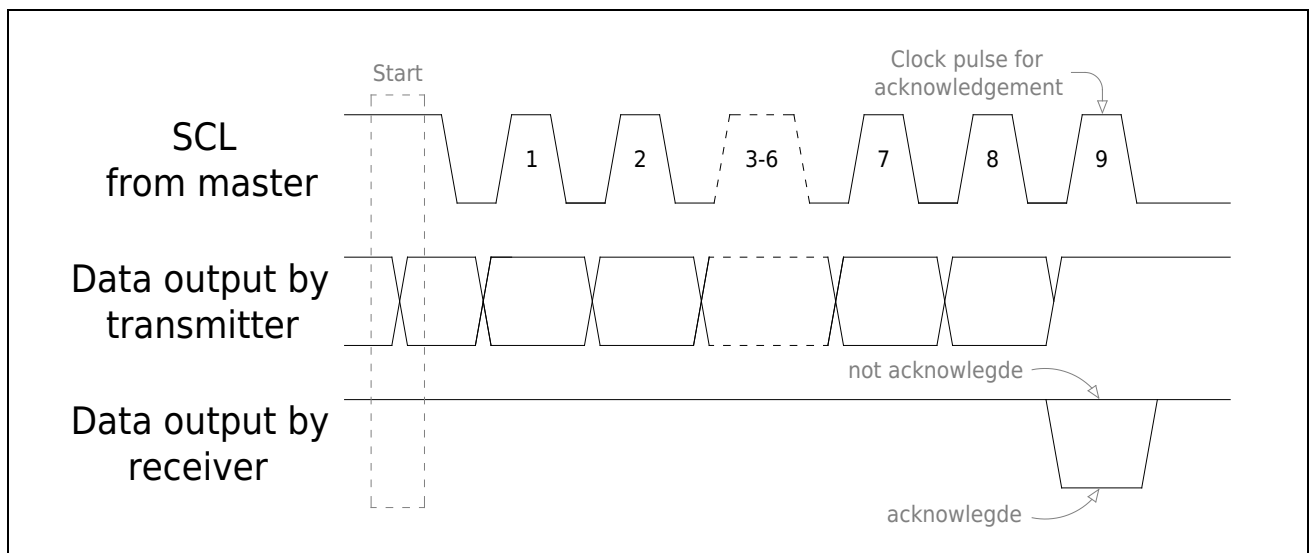


Figure 17-4 Acknowledgment signal on I2C bus

17.3.3 Arbitration on the I2C bus

Arbitration on the I2C bus is divided into two parts: synchronization on the SCL line and arbitration on the SDA line

■ Synchronization on the SCL line (clock synchronization)

Since the I2C bus has the logic function of "AND", as long as one node on the SCL line sends a low level, the bus will show a low level. The bus can only behave as a high level when all nodes are sending a high level. Therefore, the clock low time is determined by the device with the longest clock level period, and the clock high time is determined by the device with the shortest clock high period. Due to the characteristics of I2C, when multiple Masters send clock signals at the same time, a unified clock signal is represented on the bus. If the slave wants the Master to reduce the transmission speed, it can notify the Master by actively pulling SCL low to extend its low level time. When the Master finds that the SCL level is pulled low when preparing for the next transmission, it waits until the slave completes the operation. And release control of the SCL line.

■ Arbitration on SDA Online

The arbitration on the SDA line is also due to the logical function of the "AND" of the I2C bus. After the master sends data, it decides whether to exit the competition by comparing the data on the bus. The master that loses the arbitration immediately switches to the unaddressed slave state to ensure that it can be addressed by the master that wins the arbitration. A master that loses arbitration continues to output clock pulses (on SCL) until the current serial byte has been sent. This principle can ensure that the I2C bus will not lose data when multiple Masters attempt to control the bus.

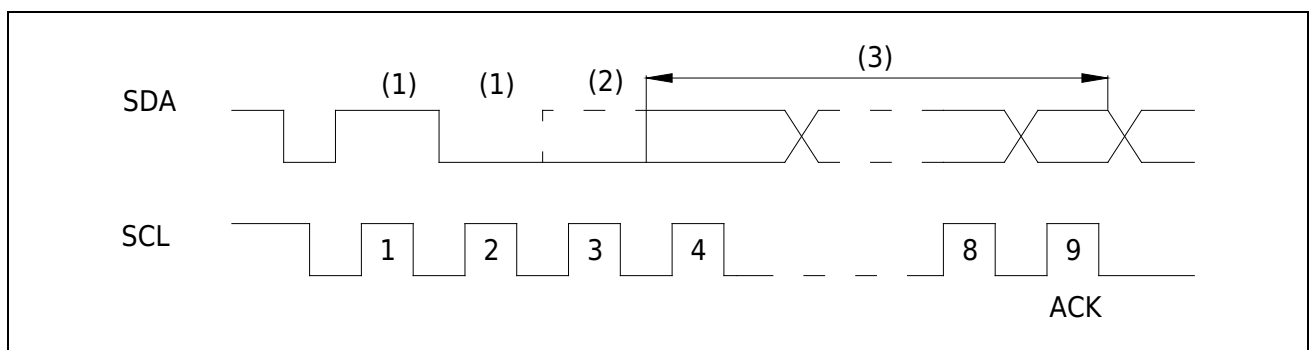


Figure 17-5 Arbitration on the I2C bus

- Another device sends serial data;
- Another device first negates a logic 1 sent by the I2C master by pulling SDA low (dotted line). Arbitration is lost, I2C enters slave receive mode;
- At this time, I2C is in the slave receiving mode, but still generates clock pulses until the current byte is sent. I2C will not generate a clock pulse for the next byte transfer. Once arbitration is won, the data transfer on SDA is initiated by the new master.

17.4 Functional description

The I2C bus uses two wires to transfer information between devices connected to the bus "SCL" (Serial Clock Line) and "SDA" (Serial Data Line). Filtering logic can filter glitches on the data bus to protect data integrity. Since there are only non-directional ports, the I2C component requires the use of open-drain buffers to the pins. Each device connected to the bus can be addressed by a specific address using software. The I2C standard is a true multi-master bus with a collision detection mechanism and an arbitration mechanism. It prevents data collisions when two or more Masters start transmitting data at the same time. I2C bus can be queried in the status register.

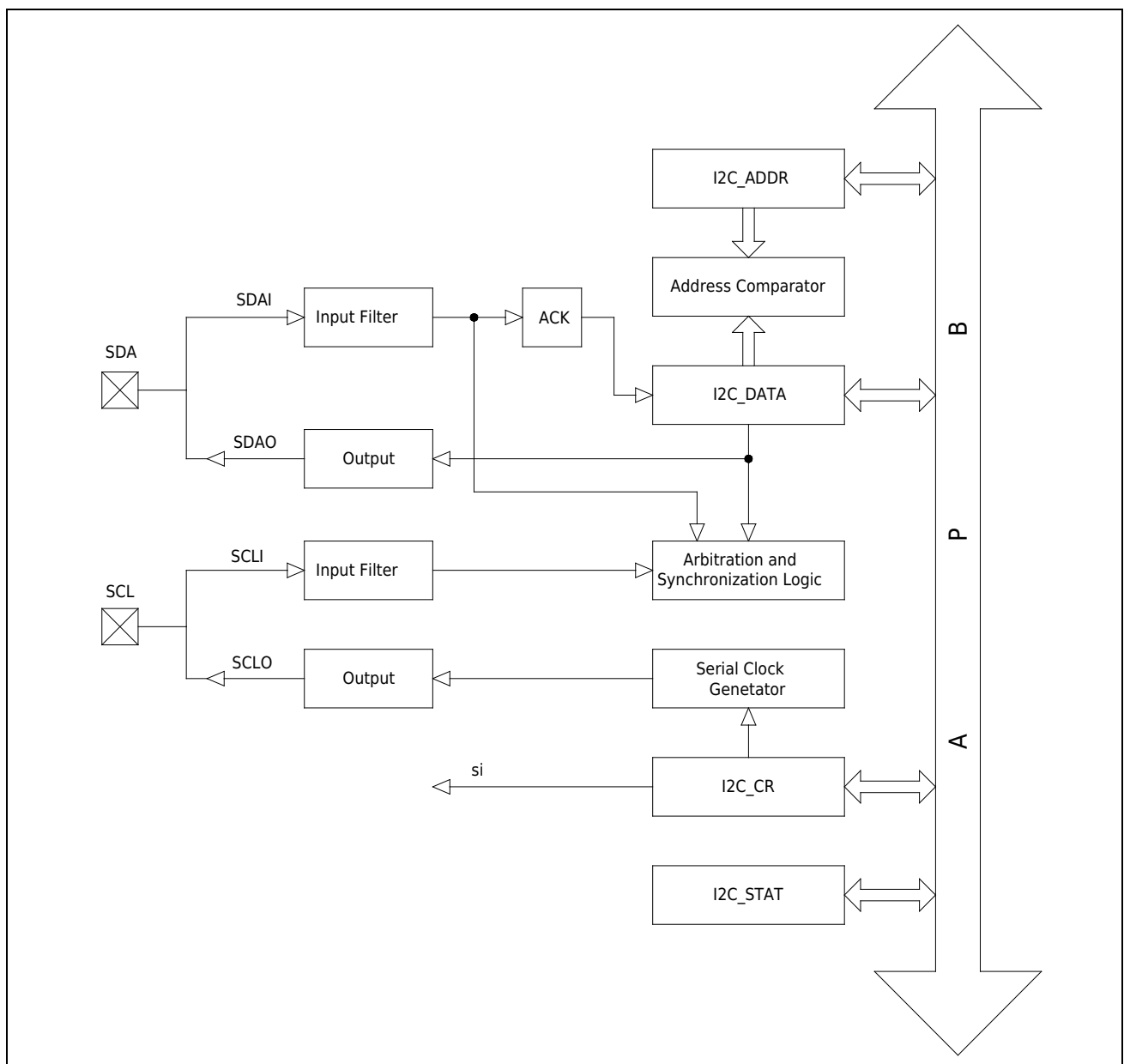


Figure 17-6 I2C function block diagram

17.4.1 Serial clock generator

The serial clock generator uses an 8 -bit counter as the baud rate generator. The frequency relationship between the SCL signal and the PCLK signal is $F_{scl} = F_{pclk} / 8 / (I2C_TM.tn+1)$, where $I2C_TM.tn$ should be greater than 0.

The following table lists the output frequency value of SCL signal when PCLK frequency is combined with $I2C_TM.tn$.

Table 17-1 I2C clock signal baud rate

PCLK (KHz)	I2C_TM.tn						
	1	2	3	4	5	6	7
1000	62	41	31	25	20	17	15
2000	125	83	62	50	41	35	31
4000	250	166	125	100	83	71	62
6000	375	250	187	150	125	107	93
8000	500	333	250	200	166	142	125
10000	625	416	312	250	208	178	156
12000	750	500	375	300	250	214	187
14000	875	583	437	350	291	250	218
16000	1000	666	500	400	333	285	250

17.4.2 Input filter

The input signal is synchronous with PCLK, and the spike pulse signal lower than the period of PCLK will be filtered out.

When this module is used as the Master, if the value of $I2C_TM$ is less than or equal to 9, $I2C_CR.H1M$ should be set to 1; if the value of $I2C_TM$ is greater than 9, $I2C_CR.H1M$ should be set to 0.

When this module is used as a slave, if the ratio of PCLK to SCL frequency is less than or equal to 30, $I2C_CR.H1M$ should be set to 1; if the ratio of PCLK to SCL frequency is greater than 30, $I2C_CR.H1M$ should be set to 0.

17.4.3 Address comparator

I2C comparator compares its own slave address with the received 7 -bit slave address. It can program its own slave address using the "I2C_ADDR" register. And it will be compared with the first received octet or broadcast address (0x00) according to the "i2cadr" bit of the "I2C_ADDR" register. If any one is the same, the "si" bit of the "I2C_CR" register will be set to 1 and an interrupt request will be generated.

17.4.4 Response flag

"aa" flag in the "I2C_CR" register is the answer flag. When the "aa" bit is 1, the I2C module responds with an acknowledgment bit after receiving the data, and when the "aa" bit is 0, the I2C module returns a non-acknowledgement bit after receiving the data.

17.4.5 Interrupt generator

"si" flag in the "I2C_CR" register is an interrupt flag. "si" flag is set to 1 whenever the value of the status register (I2C_STAT) changes (except for 0xF8). When an interrupt is generated, the status of the I2C bus can be obtained by querying the status register (I2C_STAT) to determine the actual source of the interrupt. In order to proceed to the next step, the "si" flag must be cleared by software.

17.4.6 Operating mode

The I2C component can realize 8-bit bidirectional data transmission, the transmission rate can reach 100Kbps in standard mode, 400Kbps in high-speed mode, and 1Mbps in ultra-high-speed mode, and can work in four modes: Master send mode, Master receiving mode, slave receiving mode, slave sending mode. There is also a special mode, general call mode, which operates in a similar way to slave receive mode.

■ Host send mode

The master sends multiple bytes to the slave, and the master generates a clock, so it is necessary to fill in the set value in I2C_TM. I2C_CR.sta needs to be set to 1 in master send mode. When the bus is free, the master initiates a start bit START. I2C_CR.si is set to 1 if successful. Next, write the slave address and write bit (SLA+W) into I2C_DATA, and after clearing the "si" bit, SLA+W is sent on the bus.

Next, write the slave address and write bit (SLA+W) into I2Cx_DATA, and after clearing the "si" bit, SLA+W is sent on the bus. The data is then sent according to the user-defined format. After all the data is sent, set I2C_CR.sto to 1, clear the "si" bit and send a STOP signal, or send a repeated start signal for a new round of data transmission.

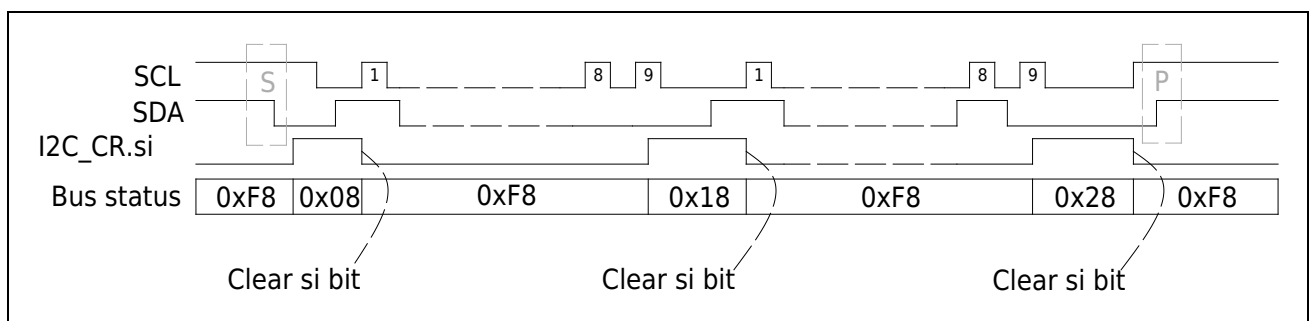


Figure 17-7 Data synchronization diagram of master sending mode

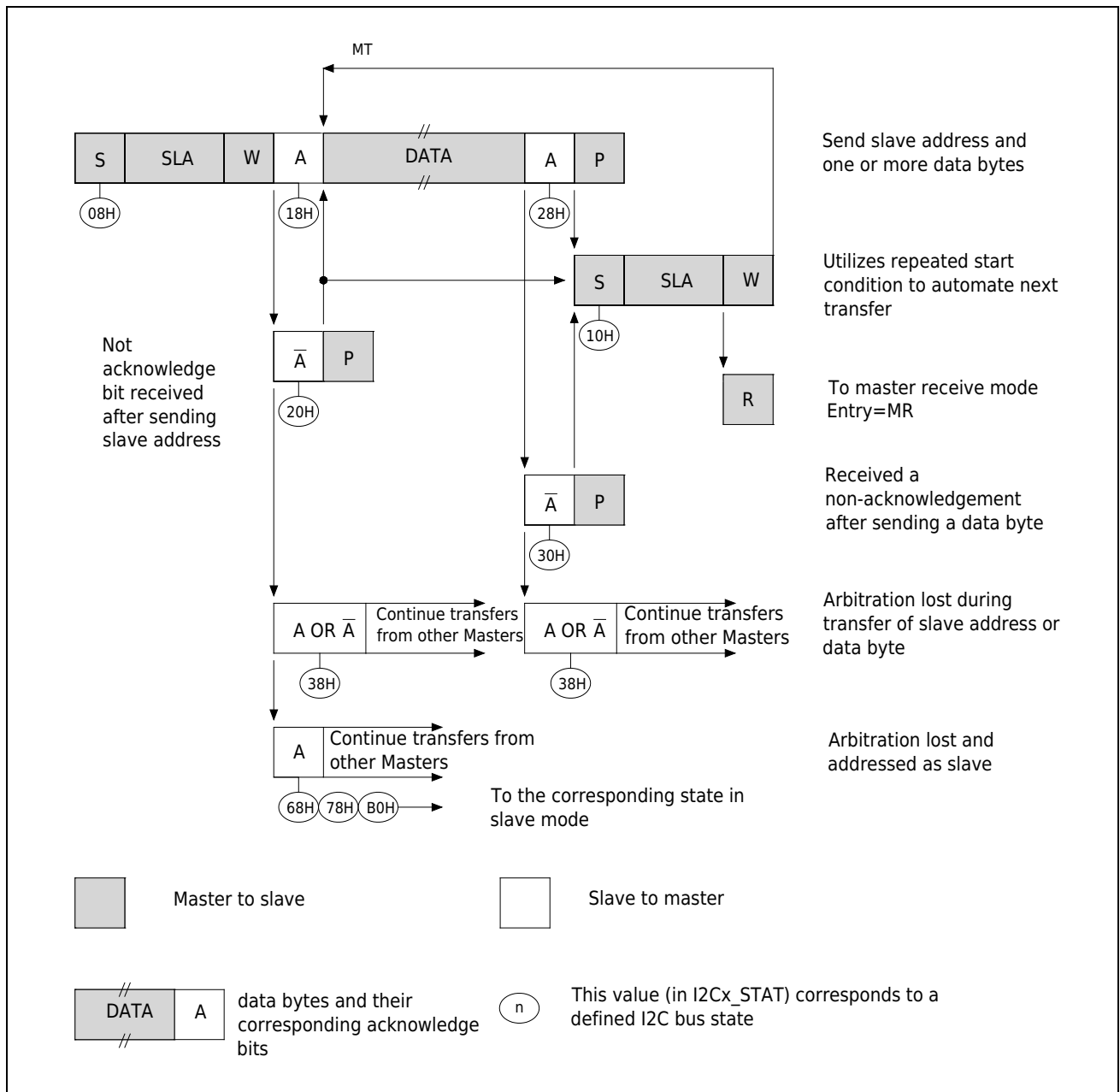


Figure 17-8 I2C master sending state diagram

■ Host receiving mode

The master receives the mode, and the data is transmitted by the slave. The initialization setting is the same as the master sending mode, after the master sends the start bit, I2C_DATA should be written into the slave address and " read bit " (SLA+R). I2C_CR.si is set to 1 after receiving the slave acknowledge bit ACK. " After si" is cleared to 0, it starts to receive data from the slave. If I2C_CR.aa is 1, the master responds with an acknowledgment bit after receiving the data; if it is 0, the master does not respond with a NACK after receiving the data.. Then the Master can send a stop signal or repeat the start signal to start the next round of data transmission.

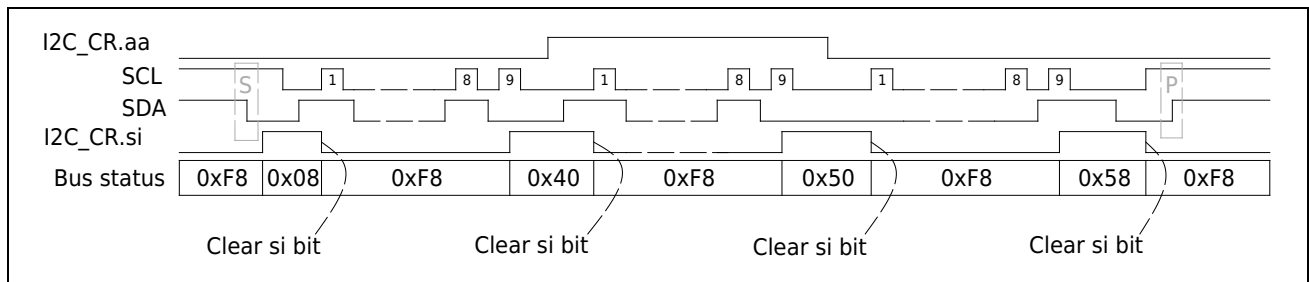


Figure 17-9 Data synchronization diagram of main receiving mode

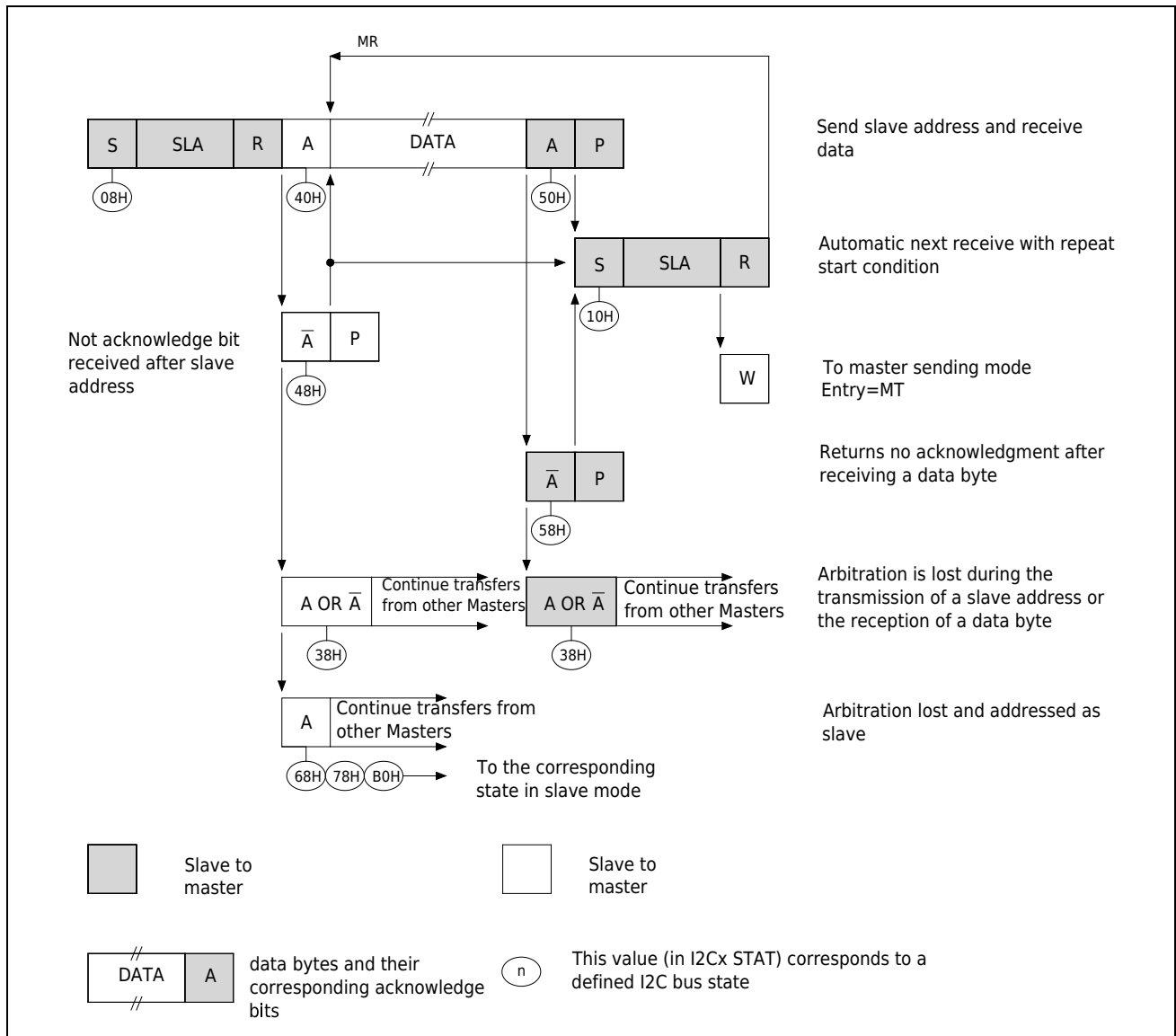


Figure 17-10 I2C master receiving state diagram

■ Slave receiving mode

In slave receiving mode, the slave receives data from the master. Before the transmission starts, I2C_ADDR should be written to the slave address, and I2C_CR.aa is set to 1 to respond to the addressing of the master. After the above initialization, the slave enters idle mode and waits for a "write" signal (SLA+W). If the master fails to arbitrate, it will also directly enter the slave

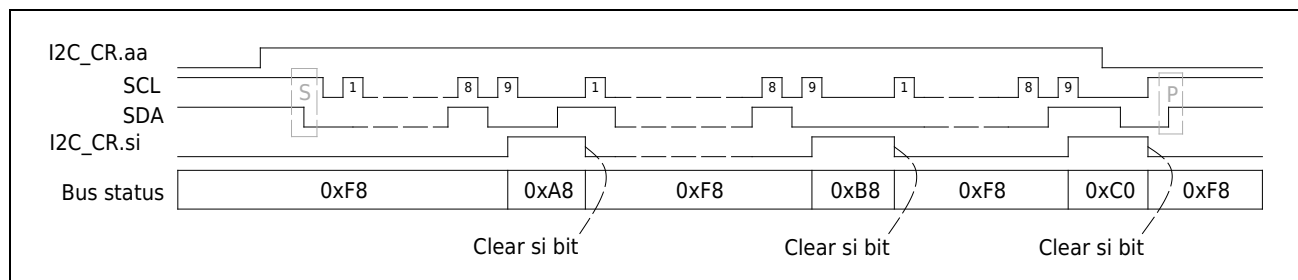


Figure 17-13 Slave mode data synchronization diagram

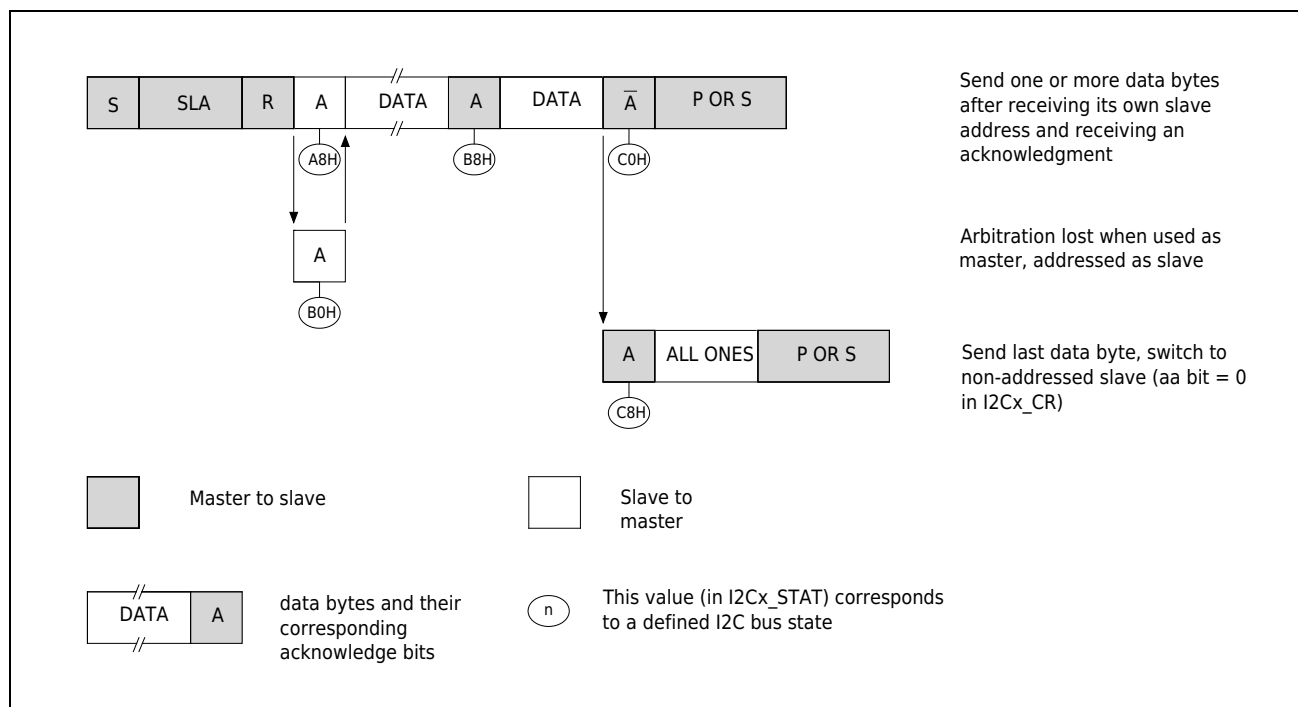


Figure 17-14 I2C Slave Transmit State Diagram

17.4.7 Status code expression

There are two special states in the I2C status register: F8H and 00H.

F8H: This status code indicates that no relevant information is available, because the serial interrupt flag "si" has not been set. This condition occurs between other states and before the I2C module has started to perform a serial transfer.

00H: This status code indicates that a bus error occurred during I2C serial transmission. A bus error occurs when a START or STOP condition occurs at an illegal position in a format frame. These illegal locations refer to address bytes, data bytes, or acknowledge bits during serial transfers. when external disturbances affect the internal I2C block signals. "si" is set when a bus error occurs.

Table 17-2 I2C status code expression

Status code	Description
Master sending mode	
08H	Start condition sent
10H	Repeated start condition sent
18H	SLA+W sent, ACK received
20H	SLA+W sent, not ACK received
28H	Data in I2C_DATA has been sent, ACK has been received
30H	Data in I2C_DATA has been sent, non-ACK has been received
38H	Loss of Arbitration when SLA+ reads or writes data bytes
Main receiving mode	
08H	Start condition sent
10H	Repeated start condition sent
38H	Arbitration lost in non- ACK
40H	SLA+R sent, ACK received
48H	SLA+R sent, not ACK received
50H	Data byte has been received, ACK has been returned
58H	Data bytes received, non- ACK returned
Slave receive mode	
60H	Has received its own SLA+W and returned ACK
68H	When the master is in SLA+ read and write lost arbitration, has received its own SLA+W, and has returned ACK
80H	The previous addressing used its own slave address, received data bytes, and returned ACK
88H	The previous addressing used its own slave address, received data bytes, and returned non- ACK
A0H	When statically addressed, a STOP condition or a repeated START condition is received
Slave send mode	
A8H	Received its own SLA+R and returned ACK
B0H	When the Master loses arbitration, has received its own SLA+R, and has returned ACK

Status code	Description
B8H	ACK received
C0H	Data bytes sent, not ACK received
C8H	Loaded data bytes have been sent, ACK has been received
general call mode	
70H	Received broadcast address (0x00), returned ACK
78H	When the master is in SLA+ read and write lost arbitration, the broadcast address has been received, and ACK has been returned
90H	The previous addressing used the broadcast address, data bytes have been received, and ACK has been returned
98H	The previous addressing used the broadcast address, the data byte has been received, and a non- ACK has been returned
A0H	When statically addressed, a STOP condition or a repeated START condition is received
Other miscellaneous status	
F8H	No relevant status information available, si=0
00H	A bus error occurs during transmission, or external interference causes I2C to enter an undefined state

17.5 Programming example

17.5.1 Master sending example

Step1: Set PERI_CLKEN.I2C to 1 to enable the I2C module clock.

Step2: Write 0 and 1 to PERI_RESET.I2C in sequence to reset the I2C module.

Step3: Configure the ports used by SCL and SDA of I2C as open-drain mode, and configure the appropriate port direction according to the I2C mode (for example, if the port direction is configured as output, the corresponding output register needs to be initialized before the corresponding bit of DIR is cleared. 1) and finally map SCL and SDA to corresponding pins.

Step4: Configure I2C_TM so that the clock rate of SCL meets the application requirements.

Step5: Set I2C_TMRUN to 1 to enable the SCL clock generator.

Step6: Set I2C_CR.ens to 1 to enable the I2C module.

Step7: Set I2C_CR.sta to 1, the bus tries to send the Start signal.

Step8: Wait for I2C_CR.si to become 1, and the Start signal has been sent to the bus.

Step9: Query I2C_STAT, if the value of this register is 0x08 or 0x10, proceed to the next step, otherwise perform error handling.

Step10: Write SLA+W to I2C_DATA, set I2C_CR.sta to 0, set I2C_CR.si to 0, and send SLA+W.

Step11: Wait for I2C_CR.si to become 1, SLA+W has been sent to the bus.

Step12: Query I2C_STAT, if the register value is 0x18, proceed to the next step. Otherwise, perform error handling.

Step13: Write the data to be sent to I2C_DATA, set I2C_CR.si to 0, and send the data.

Step14: Wait for I2C_CR.si to become 1, the data has been sent to the bus.

Step15: Query I2C_STAT, if the register value is 0x28, proceed to the next step. Otherwise, perform error handling.

Step16: If the data to be sent is not completed, then jump to Step13 to continue execution.

Step17: Set I2C_CR.sto to 1, set I2C_CR.si to 0, and the bus tries to send a Stop signal.

17.5.2 Master receive example

Step1: Set PERI_CLKEN.I2C to 1 to enable the I2C module clock.

Step2: Write 0 and 1 to PERI_RESET.I2C in sequence to reset the I2C module.

Step3: Configure the ports used by SCL and SDA of I2C as open-drain mode, and configure the appropriate port direction according to the I2C mode (for example, if the port direction is configured

as output, the corresponding output register needs to be initialized before the corresponding bit of DIR is cleared. 1) and finally map SCL and SDA to corresponding pins.

Step4: Configure I2C_TM so that the clock rate of SCL meets the application requirements.

Step5: Set I2C_TMRUN to 1 to enable the SCL clock generator.

Step6: Set I2C_CR.ens to 1 to enable the I2C module.

Step7: Set I2C_CR.sta to 1, the bus tries to send the Start signal.

Step8: Wait for I2C_CR.si to become 1, and the Start signal has been sent to the bus.

Step9: Query I2C_STAT, if the register value is 0x08 or 0x10, proceed to the next step, otherwise, perform error handling.

Step10: Write SLA+R to I2C_DATA, set I2C_CR.sta to 0, set I2C_CR.si to 0, and send SLA+R.

Step11: Wait for I2C_CR.si to become 1, SLA+R has been sent to the bus.

Step12: Query I2C_STAT, if the register value is 0x40, proceed to the next step, otherwise, proceed with error handling.

Step13: Set I2C_CR.aa to 1 to enable the response flag.

Step14: Set I2C_CR.si to 0, the slave sends data, and the master sends ACK or NACK according to I2C_CR.aa.

Step15: Wait for I2C_CR.si to become 1, read the received data from I2C_DATA.

Step16: Query I2C_STAT, if the value of this register is 0x50 or 0x58, proceed to the next step, otherwise, perform error handling.

Step17: If the data to be received is only the last byte, set I2C_CR.aa to 0 and enable the non-response flag.

Step18: If the data to be received is not completed, then jump to Step14 to continue execution.

Step19: Set I2C_CR.sto to 1, set I2C_CR.si to 0, and the bus tries to send a Stop signal.

17.5.3 Slave receiving example

Step1: Set PERI_CLKEN.I2C to 1 to enable the I2C module clock.

Step2: Write 0 and 1 to PERI_RESET.I2C in sequence to reset the I2C module.

Step3: Configure the ports used by SCL and SDA of I2C as open-drain mode, and configure the appropriate port direction according to the I2C mode (for example, if the port direction is configured as output, the corresponding output register needs to be initialized before the corresponding bit of DIR is cleared. 1) and finally map SCL and SDA to corresponding pins.

Step4: Set I2C_CR.ens to 1 to enable the I2C module.

Step5: Configure I2C_ADDR as the slave address.

Step6: Set I2C_CR.aa to 1 to enable the response flag.

Step7: Wait for I2C_CR.si to become 1 and be addressed by SLA+W.

Step8: Query I2C_STAT, if the value of this register is 0x60, proceed to the next step, otherwise perform error handling.

Step9: Set I2C_CR.si to 0, the master sends data, and the slave returns ACK or NACK according to I2C_CR.aa.

Step10: Wait for I2C_CR.si to become 1, read the received data from I2C_DATA.

Step11: Query I2C_STAT, if the value of this register is 0x80, continue to the next step, otherwise perform error handling

Step12: If the data to be received is not completed, then jump to Step9 to continue execution.

Step13: Set I2C_CR.aa to 0, and set I2C_CR.si to 0.

17.5.4 Slave sending example

Step1: Set PERI_CLKEN.I2C to 1 to enable the I2C module clock.

Step2: Write 0 and 1 to PERI_RESET.I2C in sequence to reset the I2C module.

Step3: Configure the ports used by SCL and SDA of I2C as open-drain mode, and configure the appropriate port direction according to the I2C mode (for example, if the port direction is configured as output, the corresponding output register needs to be initialized before the corresponding bit of DIR is cleared. 1) and finally map SCL and SDA to corresponding pins.

Step4: Set I2C_CR.ens to 1 to enable the I2C module.

Step5: Configure I2C_ADDR as the slave address.

Step6: Set I2C_CR.aa to 1 to enable the response flag.

Step7: Wait for I2C_CR.si to become 1 and be addressed by SLA+R.

Step8: Query I2C_STAT, if the value of this register is 0xA8, proceed to the next step, otherwise, perform error handling.

Step9: Write the data to be sent to I2C_DATA, set I2C_CR.si to 0, and send the data.

Step10: Wait for I2C_CR.si to become 1, the data has been sent to the bus.

Step11: Query I2C_STAT, if the value of this register is 0xB8 or 0xC0, proceed to the next step, otherwise, perform error handling.

Step12: If the data to be sent is not completed, then jump to Step9 to continue execution.

Step13: Set I2C_CR.aa to 0, and set I2C_CR.si to 0.

17.6 Register description

Register list

Table 17-3 Register List

I2C base address: 0x40000400

Offset	Register name	Access	Register description
0x00	I2C_TMRUN	RW	I2C baud rate counter enable register.
0x04	I2C_TM	RW	I2C baud rate counter configuration register.
0x08	I2C_CR	RW	I2C configuration register.
0x0c	I2C_DATA	RW	I2C data register.
0x10	I2C_ADDR	RW	I2C address register.
0x14	I2C_STAT	RO	I2C state register.

17.6.1 I2C Baud Rate Counter Enable Register (I2C_TMRUN)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														tme	
														RW	

Bit	Marking	Functional description
31: 1	Reserved	
0	tme	Baud rate counter enable. 0 - disable 1 - enable

17.6.2 I2C Baud Rate Counter Configuration Register (I2C_TM)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								tm							
								RW							

Bit	Marking	Functional description
31:8	Reserved	
7:0	tm	tm: Baud rate counter configuration value. $F_{scl} = F_{pclk} / 8 / (tm+1)$ where $tm > 0$

17.6.3 I2C Configuration Register (I2C_CR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								ens	sta	sto	si	aa	Res	h1m	
								RW	RW	RW	RW	RW		RW	

Bit	Marking	Functional description
31:7	Reserved	
6	ens	I2C module enable control 0 - disabled 1 - enable
5	sta	I2C bus control 0 - no function 1 - send START to the bus
4	sto	I2C bus control 0 - no function 1 - Send STOP to the bus
3	si	I2C interrupt flag reads 1, an I2C interrupt has occurred Write 0, I2C performs next operation
2	aa	Acknowledgment control bit 0 - send NAK 1 - send ACK
1	Reserved	
0	h1m	I2C filter parameter configuration 0 - advanced filtering, higher anti-interference performance 1 - Simple filtering, faster communication rate Note: See the [Input Filter] chapter for details.

17.6.4 I2C data register (I2C_DATA)

Address offset: 0x0c

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								i2cdat							
								RW							

Bit	Marking	Functional description
31:8	Reserved	
7:0	i2cdat	I2C data register In I2C send mode, write the data to be sent In I2C receive mode, read received data

17.6.5 I2C Address Register (I2C_ADDR)

Address offset: 0x10

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								i2cadr							GC
								RW							RW

Bit	Marking	Functional description
31:8	Reserved	
7:1	i2cadr	I2C slave mode address.
0	GC	Broadcast Address Acknowledgment Enable 0 - disabled 1 - enable

17.6.6 I2C Status Register (I2C_STAT)

Address offset: 0x14

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								i2csta							
								RO							

Bit	Marking	Functional description
31:8	Reserved	
7:0	i2csta	I2C state register. For the specific definition of the status value, see the chapter [State Code Expression]

18 Serial Peripheral Interface (SPI)

18.1 Introduction to SPI

The SPI interface is a synchronous serial data communication interface working in full-duplex mode, using 4 pins for communication: MISO, MOSI, SCK, CS/SSN. When SPI is used as the master, output CS and SCK signals to control the communication process. When SPI acts as a slave, it communicates under the control of SSN and SCK signals.

18.2 Main Features of SPI

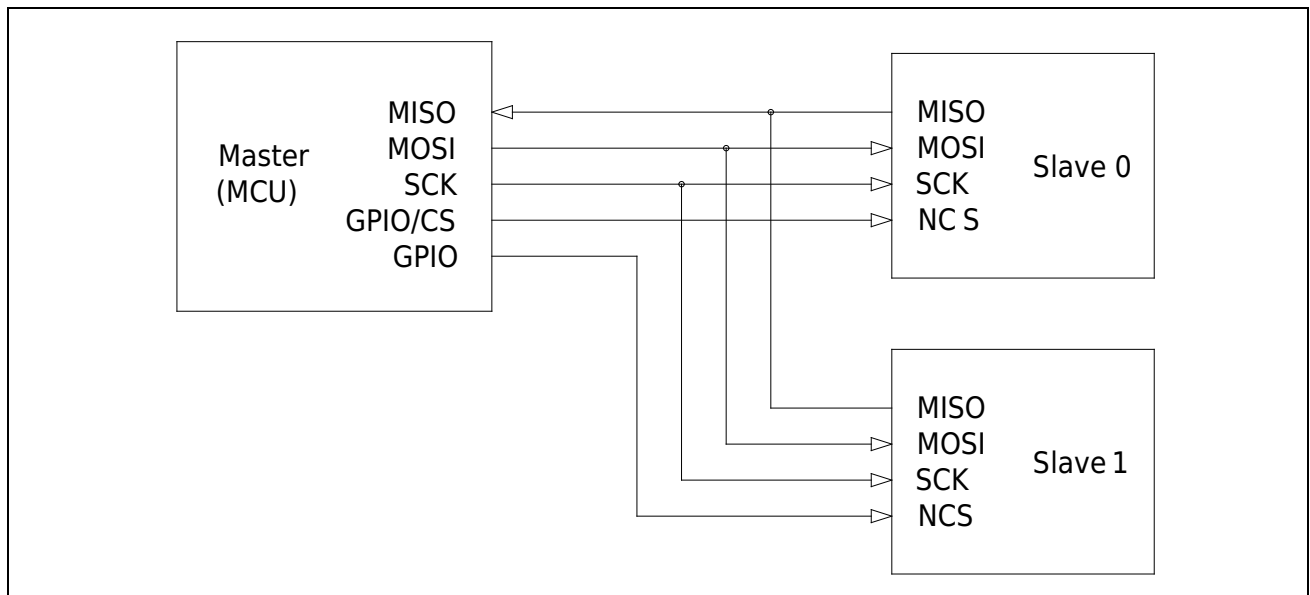
- Support SPI master mode, SPI slave mode
- Support standard four-wire full-duplex communication
- Supports configuration of serial clock polarity and phase
- Master mode supports 7 communication speeds
- The maximum frequency division factor of the host mode is $PCLK/2$
- The maximum frequency division factor of slave mode is $PCLK/4$
- The frame length is fixed at 8 bits, and the MSB is transmitted first

18.3 SPI Functional Description

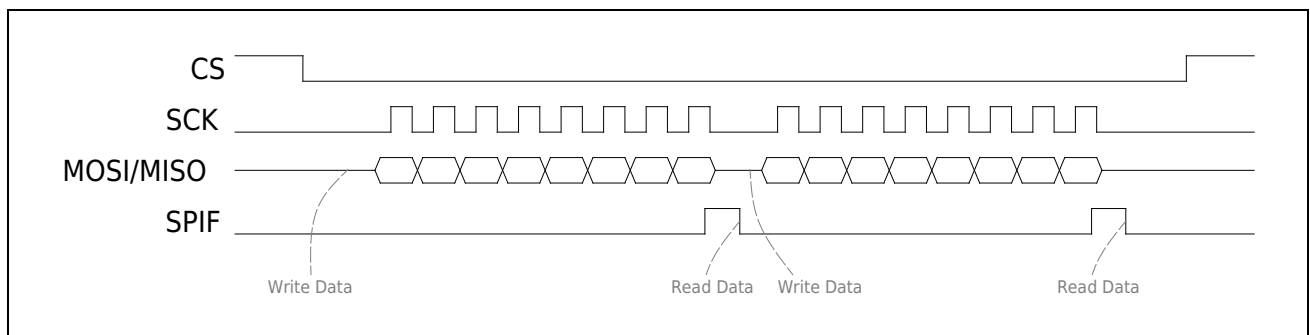
18.3.1 SPI master mode

The length of each data frame is fixed at 8 bits, and the first bit of sent data is fixed at MSB. Set SPI_CR.mstr to 1, the SPI interface works in master mode. The SPI transfer is started by writing data to the SPI_Data register. The SCK pin automatically generates a serial clock; on the edge of the serial clock, the data in the shift register is sent to the MOSI pin, and the data on the MISO pin is received into the shift register. Every time the transmission of a data frame is completed, the SPIF will be set to 1 by the hardware, and the SPIF flag can be cleared by reading SPI_DATA. SCK pin output clock is controlled by SPI_CR[spr2:spr0], and the output frequency range is $PCLK/2 \sim PCLK/128$; the output level of the CS pin is controlled by SPI_SSN.ssn, and the output of the GPIO pin The level is controlled by GPIO related registers.

A typical application block diagram of master mode is shown below.

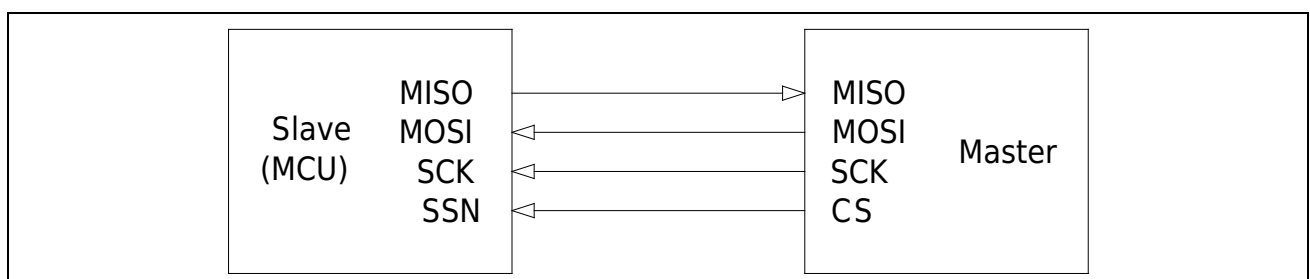


The communication diagram of the master mode is shown in the figure below, where CPOL=0, CPHA=0.



18.3.2 SPI slave mode

The length of each data frame is fixed at 8 bits, and the first bit of received data is fixed at MSB. Set SPI_CR.mstr to 0, the SPI interface works in slave mode. In this mode, the SCK pin is used as an input pin and the serial clock comes from an external Master; the SSN pin is used as an input pin and the chip select signal comes from an external Master or is fixed at low level. GPIO chapter for details on the SSN pins. A typical application block diagram of the slave mode is shown below.



When the SPI slave receives a data frame from the SPI master, the SPIF bit is set high; the user program should read the received data as soon as possible. The communication timing of receiving data in slave mode is as follows, where CPOL=0, CPHA=0.

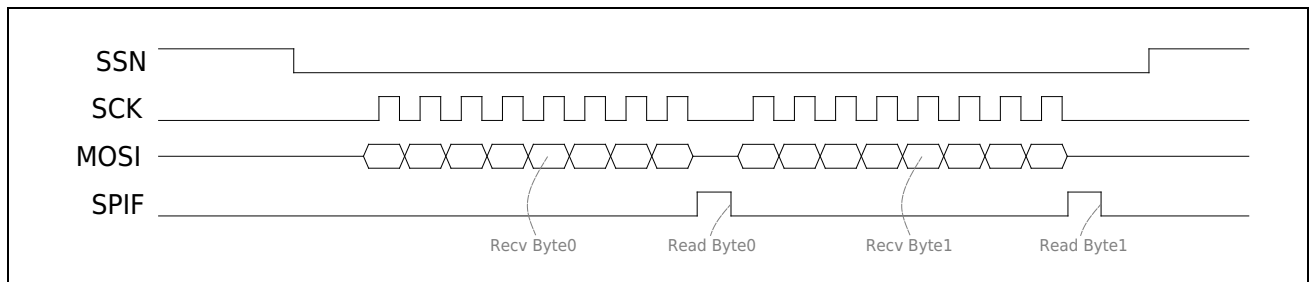


Figure 18-1 Slave receiving diagram

When the SPI slave needs to send data to the master, it should write the first byte of data to be sent to the SPI_DATA register as soon as possible after the master pulls down NSS; whenever the SPIF flag is 1, it should read SPI_DATA as soon as possible. Clear the SPIF flag and write the subsequent data to be sent to the SPI_DATA register. The communication timing of sending data in slave mode is as follows, where CPOL=0, CPHA=0.

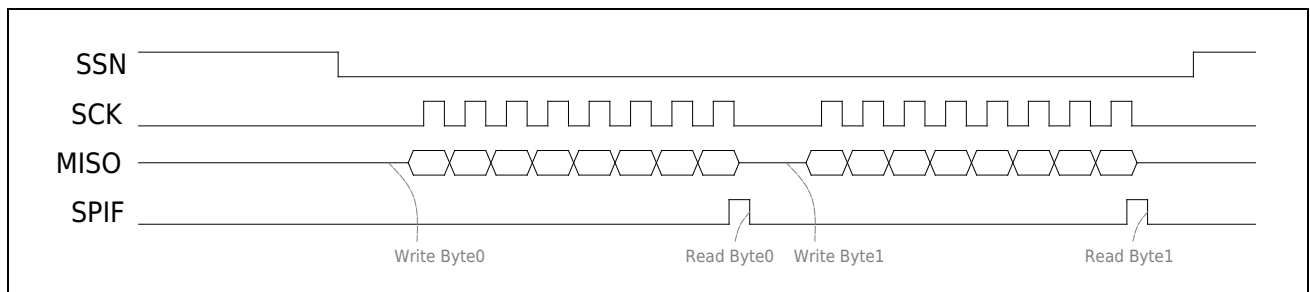


Figure 18-2 Slave sending diagram

18.3.3 SPI data frame format

The SPI interface frame format depends on the configuration of clock polarity bit CPOL and clock phase bit CPHA.

When CPOL is 0, the idle state of SCK line is low. When CPOL is 1, the idle state of SCK line is high level. When CPHA is 0, the data will be sampled when the first SCK clock transition signal jumps. When CPHA is 1, the data will be sampled when the second SCK clock signal jumps.

The frame format of the SPI interface master is shown in the figure below.

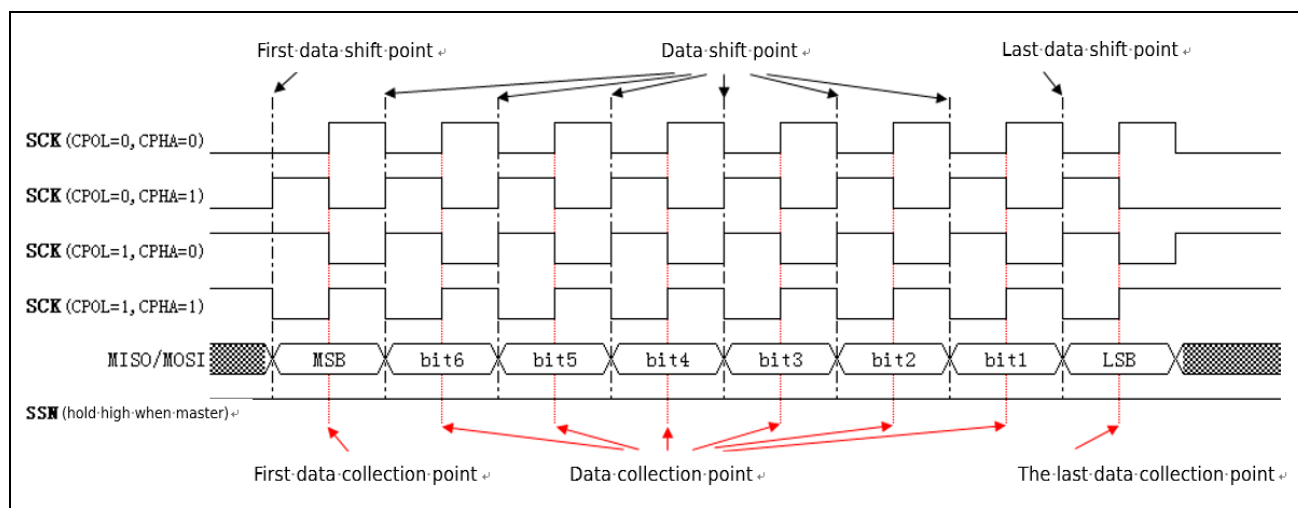


Figure 18-3 Host Mode Frame Format

The SPI interface slave frame format is shown in the figure below.

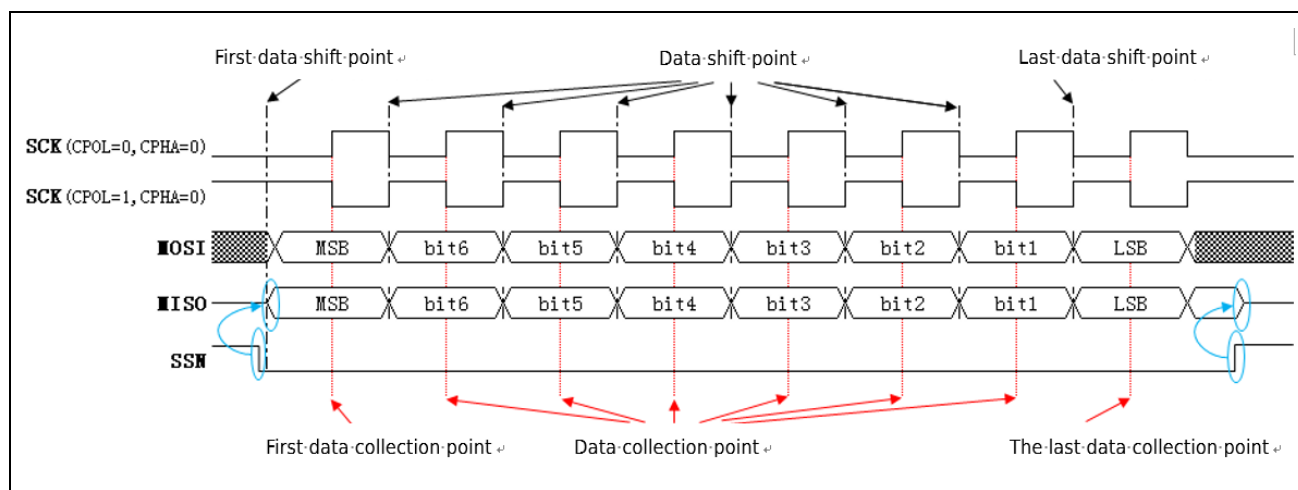


Figure 18-4 Data frame format when slave CPHA is 0

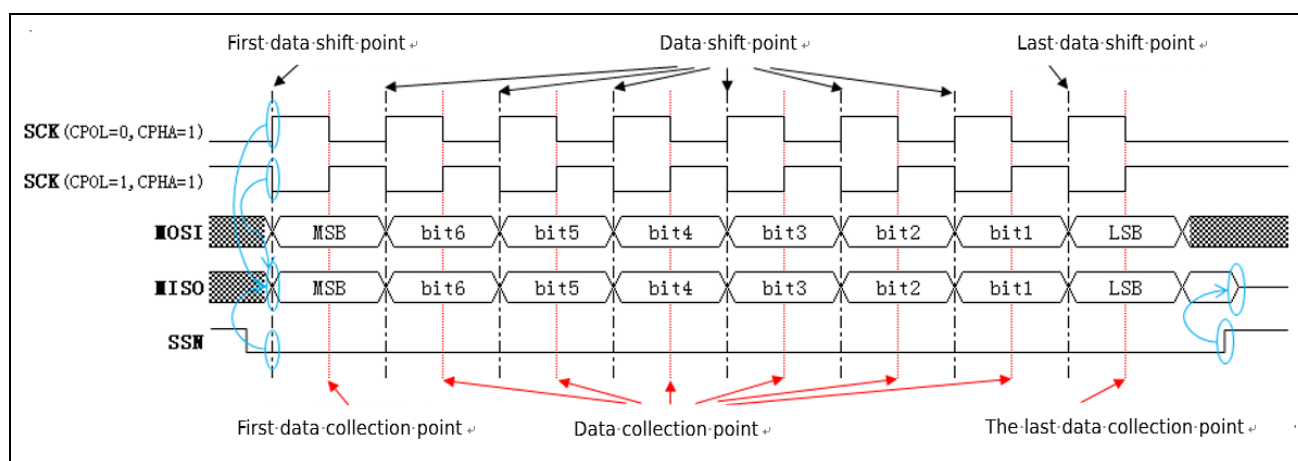


Figure 18-5 Data frame format when slave CPHA is 1

18.3.4 SPI Status Flags and Interrupts

SPI will generate the following three status flags during work, and the generation conditions and clearing methods are as follows.

- When the transmission of a data frame is completed, SPIF will be set by hardware. This flag can be cleared by reading the SPI_DATA register.
- When SPI works in master mode and the external SSN input is low level, SPI_STAT.mdf will be set by hardware, indicating that other SPI masters are occupying the bus. When the SSN input is high level, SPI_STAT.mdf will be automatically cleared by hardware.
- When the SPI works in slave mode, if the SSN pin is pulled high during data transmission, SPI_STAT.sserr will be set. Set SPI_CR.spen to 0 to clear this flag.

If the SPI interrupt vector is enabled, an interrupt can be generated in either of the following cases:

- SPI transmission is complete, that is, SPI_STAT.SPIF is 1
- SPI is wrong, that is, SPI_STAT.mdf is 1

18.3.5 SPI multi-machine system configuration instructions

- When the SPI module works as a master and works in a single-master system, the slave can be controlled through the SPI_CS pin or GPIO pin. SPI_CS pin is selected as the chip select signal of the slave, set SPI_SSN.ssn to 0 to select the slave, and set SPI_SSN.ssn to 1 to release the slave. GPIO pin is selected as the chip select signal of the slave, set the corresponding bit of the GPIOx_OUT register to 0 to select the slave, and set the corresponding bit of the GPIOx_OUT register to 1 to release the slave.
- When the SPI module is a slave, configure the source of SPI_SSN as needed (see GPIO port auxiliary controller for details). When the SSN is low, the slave machine can be selected for communication; when the SSN is high, the machine is in an unselected state.
- When the SPI mode works on multi-master and multi-slave, all slave chip select signals are connected through GPIO pins, and the master must also be connected to the SSN signals of other masters through GPIO pins to monitor whether the bus is occupied. Master0 needs to communicate as shown in the figure below is: wait for Master0.SSN to become high; output low from GPIO0 to notify Master1 to release the SPI bus; output low from GPIO2 to select Slave1; communicate with Slave1; output high from GPIO2 to release Slave1; output high from GPIO0 to release Master1.



Figure 18-6 Schematic diagram of SPI multi-master/multi-slave system

18.3.6 SPI pin configuration description

SPI can maintain some or all functions under some special pin configurations.

The specific situation is as follows ("√" means that the pin is configured and used, and blank means that the pin is not configured):

Table 18-1 SPI pin configuration description table

	SPI_CS(Master) / SPI_SSN (slave)	SCK	MOSI	MISO	Functional description
Host mode	√	√	√	√	general configuration All Masters function normally
		√	√	√	All Masters function normally
	√	√	√		Master sending function is normal
	√	√		√	Master receiving function is normal
		√	√		Master sending function is normal
		√		√	Master receiving function is normal
Slave mode	√	√	√	√	general configuration All slaves function normally
	√	√	√		Slave receiving function is normal
	√	√		√	Slave sending function is normal
	√ fixed low level	√	√	√	All slaves function normally
	√ fixed low level	√	√		Slave receiving function is normal
	√ fixed low level	√		√	Slave sending function is normal

Note:

- Conditions not listed in the table are currently not supported.
- In master mode, even if the SPI_CS chip select output is not used, SPI.SSN needs to be set to 1 before sending data, and SPI.SSN needs to be set to 0 after sending data.
- In slave mode and chip select input is fixed at low level, in order to maintain normal function, SPI_CR.cpha=1 must be satisfied.

18.4 SPI programming example

18.4.1 SPI master sending example

Step1: Map CS/SCK/MISO/MOSI to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter; configure the CS/SCK/MOSI pins as output mode, and configure the MISO pin as input mode.

Step2: Set SPI_CR.mstr to 1 to make SPI work in master mode.

Step3: Configure SPI_CR[spr2:spr0] to make the clock rate of SCK output meet the application requirements.

Step4: Configure SPI_CR.cpol and SPI_CR.cpha to make the data frame format meet the application requirements.

Step5: Set SPI_CR.spen to 1 to enable the SPI interface.

Step6: Set SPI_SSN.ssn to 0, make the CS pin output low level to select the slave.

Step7: Write the data to be sent into SPI_DATA and wait for SPIF to become 1.

Step8: Read SPI_DATA to clear SPIF flag.

Step9: If the data to be sent is not completed, then jump to Step7 to continue execution.

Step10: Set SPI_SSN.ssn to 1, make the CS pin output high level to release the slave.

Note:

- GPIO can be used instead of CS to realize chip select output, which is mostly used in multi-machine communication system.
- SPI_SSN.ssn must be set to 0 during the transmission process, and SPI_SSN.ssn must be set to 1 after the transmission is completed.

18.4.2 SPI master receive example

Step1: Map CS/SCK/MISO/MOSI to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter; configure the CS/SCK/MOSI pins as output mode, and configure the MISO pin as input mode.

Step2: Set SPI_CR.mstr to 1 to make SPI work in master mode.

Step3: Configure SPI_CR[spr2:spr0] to make the clock rate of SCK output meet the application requirements.

Step4: Configure SPI_CR.cpol and SPI_CR.cpha to make the data frame format meet the application requirements.

Step5: Set SPI_CR.spen to 1 to enable the SPI interface.

Step6: Set SPI_SSN.ssn to 0, make the CS pin output low level to select the slave.

Step7: Write arbitrary data to SPI_DATA to trigger the master to send SCK.

Step8: Query and wait for SPI_STAT.SPIF to become 1, and the data sent by the slave has been received.

Step9: Read the received data from SPI_DATA.

Step10: If the data to be received is not completed, then jump to Step7 to continue execution.

Step11: Set SPI_SSN.ssn to 1, make the CS pin output high level to release the slave.

Note:

- GPIO can be used instead of CS to realize chip select output, which is mostly used in multi-machine communication system.
- SPI_SSN.ssn must be set to 0 during the transmission process, and SPI_SSN.ssn must be set to 1 after the transmission is completed.

18.4.3 SPI slave sending example

- Step1: Map SSN/SCK/MISO/MOSI to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter; and configure the SSN/SCK/MOSI pins as input mode, and configure the MISO pin as output mode. SSN pins, see GPIO Port Auxiliary Controller.
- Step2: Set SPI_CR.mstr to 0 to make SPI work in slave mode.
- Step3: Configure SPI_CR.cpol and SPI_CR.cpha to make the data frame format meet the application requirements.
- Step4: Set SPI_CR.spen to 1 to enable the SPI interface.
- Step5: The query waits for the SSN pin to be pulled low, and the master selects the SPI slave.
- Step6: Write the data to be sent into SPI_DATA and wait for SPIF to become 1.
- Step7: Read SPI_DATA to clear SPIF flag.
- Step8: When it is found that the SSN pin is low and the data to be sent has not been completed, jump to Step6.
- Step9: The query waits for the SSN pin to be pulled high, and the master releases the SPI slave.

18.4.4 SPI Slave Receive Example

- Step1: Map SSN/SCK/MISO/MOSI to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter; and configure the SSN/SCK/MOSI pins as input mode, and configure the MISO pin as output mode. SSN pins, see GPIO Port Auxiliary Controller.
- Step2: Set SPI_CR.mstr to 0 to make SPI work in slave mode.
- Step3: Configure SPI_CR.cpol and SPI_CR.cpha to make the data frame format meet the application requirements.
- Step4: Set SPI_CR.spen to 1 to enable the SPI interface.
- Step5: The query waits for the SSN pin to be pulled low, and the master selects the SPI slave.
- Step6: The query waits for SPI_STAT.SPIF to become 1, and the data sent by the master has been received.
- Step7: Read the received data from SPI_DATA.
- Step8: If the data to be received is not completed, then jump to Step6 to continue execution.
- Step9: The query waits for the SSN pin to be pulled high, and the master releases the SPI slave.

18.5 SPI register description

Register list

Table 18-2 SPI register list

SPI base address: 0x40000800

Offset	Register name	Access	Register description
0x00	SPI_CR	RW	SPI configuration register
0x04	SPI_SSN	RW	SPI Chip Select Configuration Register
0x08	SPI_STAT	RO	SPI status register
0x0c	SPI_DATA	RW	SPI data register

18.5.1 SPI Configuration Register (SPI_CR)

Address offset: 0x00

Reset value: 0x0000 0014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								spr2	spen	Res	mstr	cpol	cpha	spr1	spr0
								RW	RW		RW	RW	RW	RW	RW

Bit	Marking	Functional description			
31:8	Reserved				
7	spr2	Baud rate selection bit 2 Refer to spr0.			
6	spen	SPI module enable control 0 - disabled 1 - enable			
5	Reserved				
4	mstr	SPI working mode configuration 0 - Slave mode 1 - Host mode			
3	cpol	SCK line idle state configuration 0 - Low level 1 - High level			
2	cpha	Clock Phase Configuration 0 - first edge 1 - second edge			
1	spr1	Baud rate selection bit 1 Reference spr0			
0	spr0	Baud rate selection bit 0			
		Table 18-3 Master Mode Baud Rate Selection			
		spr2	spr1	spr0	SCK Rate
		0	0	0	PCLK /2
		0	0	1	PCLK /4
		0	1	0	PCLK /8
		0	1	1	PCLK /16
		1	0	0	PCLK /32
		1	0	1	PCLK /64
		1	1	0	PCLK /128
1	1	1	Reserved		

18.5.2 SPI Chip Select Configuration Register (SPI_SSN)

Address offset: 0x04
Reset value: 0x000000FF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														ssn	
															RW

Bit	Marking	Functional description
31:1	Reserved	
0	ssn	SPI_CS output level configuration in master mode 0: SPI_CS port output low level 1: SPI_CS port outputs high level

18.5.3 SPI Status Register (SPI_STAT)

Address offset: 0x08
Reset value: 0x00000004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								spif	Res.	sserr	mdf	Reserved			
								RO		RO	RO				

Bit	Marking	Functional description
31:8	Reserved	
7	spif	Transfer complete flag 1: The SPI bus has completed a byte transfer 0: SPI bus is transmitting
6	Reserved	
5	sserr	SSN error flag in slave mode
4	mdf	In Master mode, the conflict flag 1: SSN pin level is low 0: SSN pin level is high
3:0	Reserved	

18.5.4 SPI data register (SPI_DATA)

Address offset: 0x0c

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								spdat							
								RW							

Bit	Marking	Functional description
31:8	Reserved	
7:0	spdat	Data register In send mode, write the byte to be sent to this register; In receive mode, the received byte is read from this register;

19 Clock Trim Module (CLKTRIM)

19.1 Introduction to CLK_TRIM

The CLK_TRIM (Clock Trimming) module is a circuit specially used to calibrate/monitor the clock. In the calibration mode, select an accurate clock source to calibrate the inaccurate clock source, repeat the calibration, and adjust the parameters of the inaccurate clock source until the frequency of the calibrated clock source meets the accuracy requirements. In the calibration mode, the count value will have a certain error, but it is within the allowable precision error range. In the monitoring mode, select a stable clock source to monitor the system's working clock. Under the set monitoring period, monitor whether the system's working clock is invalid and generate an interrupt. In calibration mode and monitor mode, the required clock sources must be initialized and enabled. For the specific configuration process, please refer to Chapter 4 System Controller.

19.2 CLK_TRIM main features

CLK_TRIM supports the following features:

- Calibration mode
- Monitoring mode
- 32-bit reference clock counter can be loaded with initial value
- 32-bit clock counter to be calibrated with configurable overflow value
- 6 reference clock sources
- 4 clock sources to be calibrated
- Support interrupt mode

19.3 CLK_TRIM function description

19.3.1 CLK_TRIM calibration mode

The calibration mode is mainly used to select an accurate clock source as a reference clock to calibrate an inaccurate clock source to be calibrated.

The software repeatedly calibrates according to the following operation process, and adjusts the parameters of the clock source to be calibrated until the clock source to be calibrated meets the frequency accuracy requirements.

19.3.1.1 Operating procedures

1. Set the CLKTRIM_CR.refclk_sel register to select the reference clock.
2. Set the CLKTRIM_CR.calclk_sel register to select the clock to be calibrated.
3. Set the CLKTRIM_REFCON.rcntval register to the calibration time.
4. Set CLKTRIM_CR.IE register to enable interrupt.
5. Set the CLKTRIM_CR.trim_start register to start the trim.
6. The reference clock counter and the clock to be calibrated counter start counting.
7. When the reference clock counter counts down from the initial value to 0, CLKTRIM_IFR.stop is set to 1 and an interrupt is triggered.
8. The interrupt service subroutine judges that CLKTRIM_IFR.stop is 1, reads the values of registers CLKTRIM_REFCNT and CLKTRIM_CALCNT,
9. Clear the CLKTRIM_CR.trim_start register to end the trim.

Note:

- In the calibration mode, it is possible that the calibration time is set too long, and the clock counter to be calibrated overflows before CLKTRIM_IFR.stop is set to 1, and CLKTRIM_IFR.calcnt_of is set to 1, triggering an interrupt. When the interrupt service subroutine finds that CLKTRIM_IFR.calcnt_of is set to 1, clear the CLKTRIM_CR.trim_start register to end the calibration.

In this case, the calibration cannot be performed correctly, and the calibration time must be adjusted and re-calibrated.

The specific steps are:

Set the CLKTRIM_REFCON.rcntval register to adjust the calibration time.

Set the CLKTRIM_CR.trim_start register to restart the trim.

19.3.2 CLK_TRIM monitoring mode

The monitoring mode is mainly used to select a stable clock source as the reference clock, and monitor the abnormal state of the system working clock in the set time period. In monitoring mode, only external XTH clock or external XTL clock can be selected as the monitored clock.

19.3.2.1 Operating procedures

1. Set the CLKTRIM_CR.refclk_sel register to select the reference clock.
2. Set the CLKTRIM_CR.calclk_sel register to select the monitored clock.
3. Set the CLKTRIM_REFCON.rcntval register to monitor interval time.
4. Set the CLKTRIM_CALCON.ccntval register to the overflow time of the monitored clock counter.
5. Set the CLKTRIM_CR.mon_en register to enable the monitor function.
6. Set CLKTRIM_CR.IE register to enable interrupt.
7. Set the CLKTRIM_CR.trim_start register to start monitoring.
8. The reference clock counter and the monitored clock counter start counting.
9. When the counting of the reference clock counter reaches the monitoring interval time, it is judged whether the monitored clock counter overflows. If it overflows, it means that the monitored clock is working normally. If there is no overflow, it means that the monitored clock fails, CLKTRIM_IFR.xtal32k_fault/xtal32m_fault is set to 1, and an interrupt is triggered.
10. Process the interrupt service subroutine, clear the interrupt flag bit CLKTRIM_IFR.xtal32k_fault/xtal32m_fault, and clear the CLKTRIM_CR.trim_start register to end monitoring.

19.4CLK_TRIM register description

Register list

Table 19-1 Register List

Base address: 0x40001800

Offset	Register name	Access	Register description
0x00	CLKTRIM_CR	RW	Configuration register.
0x04	CLKTRIM_REFCON	RW	Reference counter initial value configuration register.
0x08	CLKTRIM_REFCNT	RO	Reference counter value register.
0x0c	CLKTRIM_CALCNT	RO	Calibration counter value register.
0x10	CLKTRIM_IFR	RO	Interrupt flag bit register.
0x14	CLKTRIM_ICLR	RW	Interrupt flag bit clear register
0x18	CLKTRIM_CALCON	RW	Calibration Counter Overflow Value Configuration Register

19.4.1 Configuration Register (CLKTRIM_CR)

Offset address:0x00

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								IE	mon_en	calclk_sel	refclk_sel		trim_start		
								RW	RW	RW	RW		RW		

Bit	Marking	Functional description
31:8	Reserved	
7	IE	Interrupt Enable Register 0 - Disable 1 - Enable
6	mon_en	Monitor Mode Enable Register 0 - Disable 1 - Enable
5:4	calclk_sel	To-Calibrate / Monitor Clock Select Register 00 ---- RCH 01 ---- XTH 10 ---- RCL 11 ---- XTL
3:1	refclk_sel	Reference Clock Select Register 000 ---- RCH 001 ---- XTH 010 ---- RCL 011 ---- XTL 100 ---- IRC10K 101 ---- EXT_CLK_IN
0	trim_start	Calibration / Monitoring Start Register 0 - Stop 1 - Start

19.4.2 Reference counter initial value configuration register (CLKTRIM_REFCON)

Offset address:0x04

Reset value:0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rcntval[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rcntval[15:0]															
RW															

Bit	Marking	Functional description
31:0	rcntval	Reference counter initial value

19.4.3 Reference Counter Value Register (CLKTRIM_REFCNT)

Offset address: 0x08

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
refcnt[31:16]															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
refcnt[15:0]															
RO															

Bit	Marking	Functional description
31:0	refcnt	Reference counter value

19.4.4 Calibration Counter Value Register (CLKTRIM_CALCNT)

Offset address: 0x0c

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
calcnt[31:16]															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
calcnt[15:0]															
RO															

Bit	Marking	Functional description
31:0	calcnt	Calibration counter value

19.4.5 Interrupt Flag Register (CLKTRIM_IFR)

Offset address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												xth_fault	xtl_fault	calcnt_of	stop
												RO	RO	RO	RO

Bit	Marking	Functional description
31:4	Reserved	
3	xth_fault	XTH failure flag. CLKTRIM_ICLR.xth_fault_clr write zero to clear this flag
2	xtl_fault	XTL failure flag. CLKTRIM_ICLR.xtl_fault_clr write zero to clear this flag
1	calcnt_of	Calibration counter overflow flag. CLKTRIM_CR.start write zero to clear this flag
0	stop	Reference counter stop flag. CLKTRIM_CR.start write zero to clear this flag

19.4.6 Interrupt flag clear register (CLKTRIM_ICLR)

Offset address: 0x14

Reset value: 0xf

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												xth_fault_clr	xtl_fault_clr	Reserved	
												RW	RW		

Bit	Marking	Functional description
31:4	Reserved	
3	xth_fault_clr	Clear XTH failure flag, write zero to clear.
2	xtl_fault_clr	Clear the XTL failure flag, write zero to clear.
1:0	Reserved	

19.4.7 Calibration Counter Overflow Value Configuration Register
(CLKTRIM_CALCON)

Offset address: 0x18

Reset value: 0xffff ffff

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ccntval[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ccntval[15:0]															
RW															

Bit	Marking	Functional description
31:0	ccntval	Calibration counter overflow value

20 Cyclic redundancy check (CRC)

20.1 Overview

A cyclic redundancy check (CRC) calculation unit takes a data stream or data block as input and generates an output number under the control of a generator polynomial. This output number is often used to verify the correctness and integrity of data transmission or storage. This module supports calculating CRC value and checking CRC value.

20.2 Main characteristics

- One implementation standard: ISO/IEC13239
- One encoding method: CRC-16, $x^{16} + x^{12} + x^5 + 1$
- Three write bit widths: 8bit, 16bit, 32bit
- Two working modes: CRC encoding mode, CRC check mode

20.3 Functional description

20.3.1 Operating mode

This module supports two working modes: CRC encoding mode and CRC checking mode.

The CRC encoding mode is to input a certain amount of raw data to the CRC module and obtain the output value (CRC_RESULT.RESULT) generated by the CRC module. The CRC check mode is to input a certain amount of original data + CRC check value to the CRC module, and verify whether the original data matches the CRC check value (CRC_RESULT.FLAG).

20.3.2 Encoding

This module supports CRC-16 encoding, the calculation result is 16 bits, and the generating polynomial is $x^{16} + x^{12} + x^5 + 1$.

20.3.3 Write bit width

This module supports three write bit widths: 8bit, 16bit, 32bit. The writing of different bit widths needs to comply with the principle of "consistent bit width, low first and then high", that is, "every time data is written, it must be written into a register equal to the effective data bit width of this time, and the lower bit Data is written before higher bit data".

The following shows how the same sequence of data is written using three bit widths, and the output results are the same.

- 8bit bit width write: 0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77
- 16bit bit width write: 0x1100, 0x3322, 0x5544, 0x7766
- 32bit bit width write: 0x33221100, 0x77665544

20.4 Programming example

20.4.1 CRC-16 encoding mode

Step 1: Write 0xFFFF to CRC_RESULT to initialize CRC calculation.

Step 2: Write the original data to be encoded into the CRC_DATA register in sequence, and the write bit width can be selected from 8bit, 16bit, and 32bit.

Step 3: Read CRC_RESULT.RESULT to get the CRC value.

20.4.2 CRC-16 check mode

Step 1: Write 0xFFFF to CRC_RESULT to initialize CRC calculation.

Step 2: Write the encoded data sequence into the CRC_DATA register in sequence, and the write bit width can be selected from 8bit, 16bit, and 32bit.

Step 3: CRC_RESULT.FLAG determines whether the encoded data sequence has been tampered with.

20.5 Register description

20.5.1 Register list

Base address: 0x4002 0900

Register	Offset address	Description
CRC_RESULT	0x04	CRC Result Register
CRC_DATA	0x80	CRC data register

20.5.2 Results register (CRC_RESULT)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															FLAG
															RO
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESULT															
RW															

Bit	Symbol	Description
16	FLAG	CRC check result 0: The current CRC check error 1: The current CRC check is correct
15:0	RESULT	CRC calculation result Read this register to get the CRC calculation result 0xFFFF to this register initializes the CRC calculation

20.5.3 Data register (CRC_DATA)

Offset address: 0x80

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA[31:16]															
WO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA[15:0]															
WO															

Bit	Symbol	Functional description
31:0	DATA	This register is used to write data that needs to be calculated, and supports 3 write bit widths 8bit writing method: * ((uint8_t *)0x40020980) = 0xXX 16bit writing method: * ((uint16_t *)0x40020980) = 0xFFFF 32bit writing method: * ((uint32_t *)0x40020980) = 0xFFFFFFFF

21 Analog-to-digital converter (ADC)

21.1 Module Introduction

External analog signals need to be converted into digital signals to be further processed by the MCU. This series integrates a 12-bit successive approximation analog-to-digital converter (SAR ADC) module with high precision and high conversion rate. Has the following properties:

- 12-bit conversion accuracy;
- 1MSPS conversion speed ($VCC > 2.7V$);
- 11 conversion channels: 9 pin channels, built-in temperature sensor, 1/3 power supply voltage;
- 4 kinds of reference sources: power supply voltage, ExRef pin, built-in 1.5V reference voltage, built-in 2.5V reference voltage;
- ADC voltage input range: $0 \sim V_{ref}$;
- 3 conversion modes: single conversion, continuous conversion, cumulative conversion;
- Software can configure ADC conversion rate;
- Built-in signal amplifier, can convert high impedance signal;
- Support on-chip peripherals to automatically trigger ADC conversion, effectively reducing chip power consumption and improving real-time conversion.

21.2 ADC block diagram

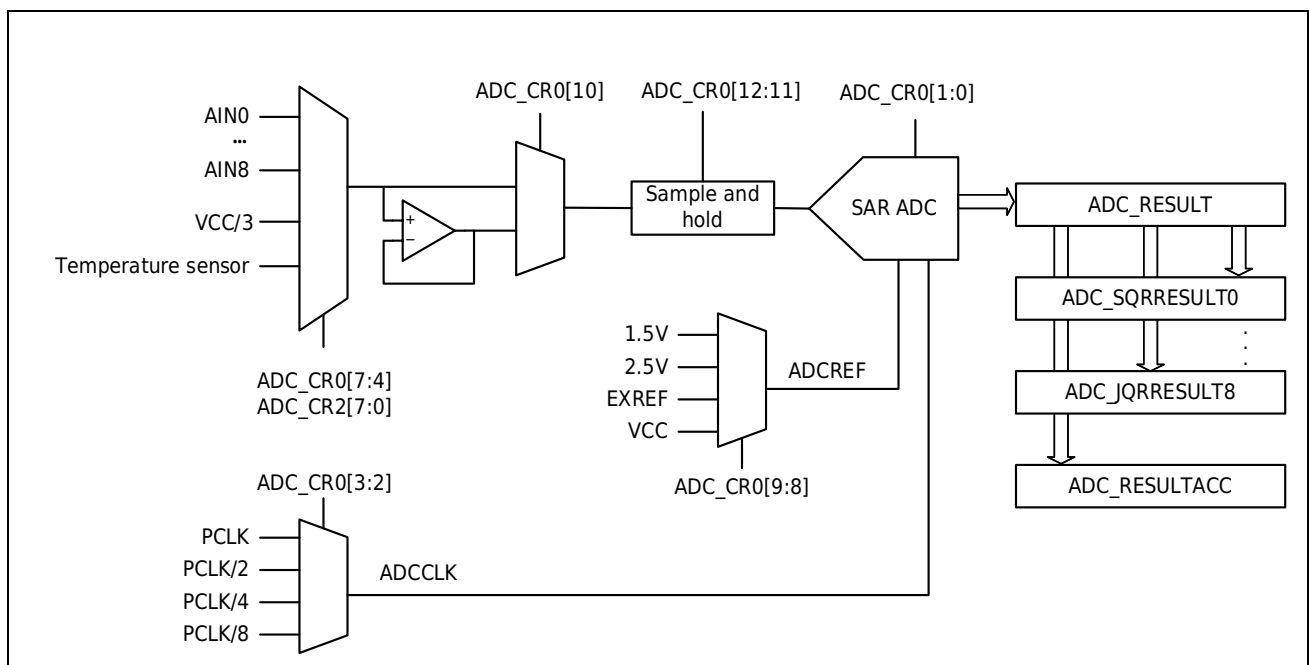


Figure 21-1 ADC block diagram

21.3 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. Among them, the sampling process requires 4~12 AdcClks, which are configured by ADC_CR0.SAM; the successive comparison process requires 16 AdcClks. Therefore, an ADC conversion requires a total of 20~ 28 AdcClk.

ADC conversion speed is sps, that is, how many ADC conversions are performed per second. The calculation method of the ADC conversion speed is: the frequency of AdcClk / the number of AdcClk required for one ADC conversion.

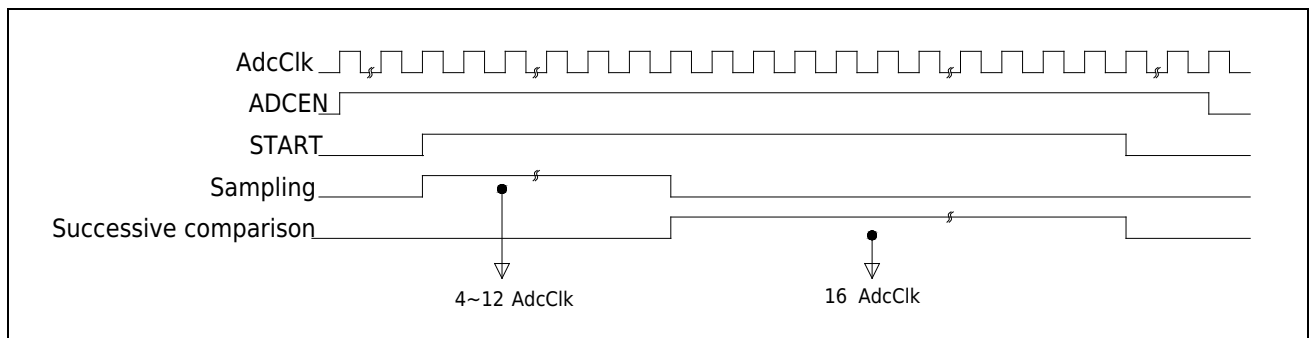


Figure 21-2 ADC conversion timing diagram

The ADC conversion speed is related to the ADC reference voltage and VCC voltage. The maximum conversion speed is shown in the table below:

ADC reference voltage	VCC voltage	Maximum conversion speed	Maximum AdcClk Frequency
Internal 1.5V	1.8~5.5V	200Ksps	4MHz
Internal 2.5V	2.8~5.5V	200Ksps	4MHz
VCC / ExRef	1.8~2.4V	200Ksps	4MHz
VCC / ExRef	2.4~2.7V	500Ksps	16MHz
VCC / ExRef	2.7~5.5V	1Msps	24MHz

21.4 Single conversion mode

In the single-conversion mode, only one conversion is performed after the ADC starts, and all 12 ADC channels can be converted. This mode can be started by setting the ADC_CR0.START bit or by setting the external trigger of ADC_CR1[9:0]. Once the ADC conversion of the selected channel is completed, the ADC_CR0.START bit is automatically cleared and the conversion result is saved in the ADC_result register.

Start the ADC single conversion operation process through the START bit:

Step1: Configure the corresponding bits of P0ADS~P3ADS, and configure the ADC channel to be converted as an analog port.

Step2: Set P3ADS.6 to 1, and 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 BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.ADCEN to 1 to enable the ADC module.

Step5: Delay 20uS, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.CT to 0, select single conversion mode.

Step7: Configure ADC_CR0. SREF to select the reference voltage of ADC.

Step8: Configure ADC_CR0. SAM and ADC_CR0. CLKSEL to set the conversion speed of ADC.

Step9: Configure ADC_CR0. SEL to select the channel to be converted.

Step10: Set ADC_CR0.START to 1 to start ADC single conversion.

Step11: Wait for ADC_CR0.START to become 0, read the ADC_result register to get the ADC conversion result.

Step12: To convert other channels, repeat Step9~Step11.

Step13: Set ADC_CR0.ADCEN and BGR_CR.BGR_EN to 0, turn off the ADC module and BGR module.

ADC single conversion operation process through an external trigger:

Step1: Configure the corresponding bits of P0ADS~P3ADS, and configure the ADC channel to be converted as an analog port.

Step2: Set P3ADS.6 to 1, and 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 BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.ADCEN to 1 to enable the ADC module.

Step5: Delay 20uS, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.CT to 0, select single conversion mode.

Step7: Set ADC_HT to 0x00.

Step8: Set ADC_CR1. HtCmp to 1 to enable ADC high threshold comparison function.

Step9: Set ADC_CR0.IE to 1 to enable ADC interrupt.

Step10: Enable the ADC interrupt in the NVIC interrupt vector table.

Step11: Configure ADC_CR0. SREF to select the reference voltage of ADC.

Step12: Configure ADC_CR0. SAM and ADC_CR0. CLKSEL to set the conversion speed of ADC.

Step13: Configure ADC_CR0. SEL to select the channel to be converted.

Step14: Set ADC_IFR to 0x00, clear the ADC interrupt flag.

Step15: Configure ADC_CR1. TRIGS1 and ADC_CR1. TRIGS0 to select the external trigger condition.

Step16: When the external trigger condition triggers the ADC to complete the conversion, the ADC module will generate an interrupt. The user can read the ADC_result register in the ADC interrupt service program to obtain the ADC conversion result.

Step17: To convert other channels, repeat Step13~Step16.

Step18: Set ADC_CR0.ADCEN and BGR_CR.BGR_EN to 0, turn off the ADC module and BGR module.

21.5 Continuous Conversion Mode

In the continuous conversion mode, starting the ADC once can perform multiple conversions on multiple channels in sequence; the ADC channels that can be converted are AIN0~AIN7. ADC conversions is configured by ADCCR2.ADCCNT; the channel to be converted is configured by ADC_CR2[7:0]. This mode can be started by setting the ADC_CR0.START bit or by setting the external trigger of ADC_CR2[9:0]. After starting the continuous conversion, the ADC module converts the channels to be converted in AIN0~AIN7 in turn until the total number of conversions is completed. After the ADC module completes the total number of conversions, the ADC_IFR.CONT_INTF bit will be automatically set to 1, and the conversion results will be stored in the ADC_result0~ADC_result7 registers corresponding to the conversion channels. ADC channels to be converted, only the last conversion result is saved in the ADC_result0~ADC_result7 registers.

The figure below demonstrates the process of 10 consecutive conversions on AIN0, AIN1, AIN5. After setting START to 1 through the register, the state machine inside the ADC will convert AIN0, AIN1, and AIN5 in sequence until the count value of ADCCNT becomes 0.

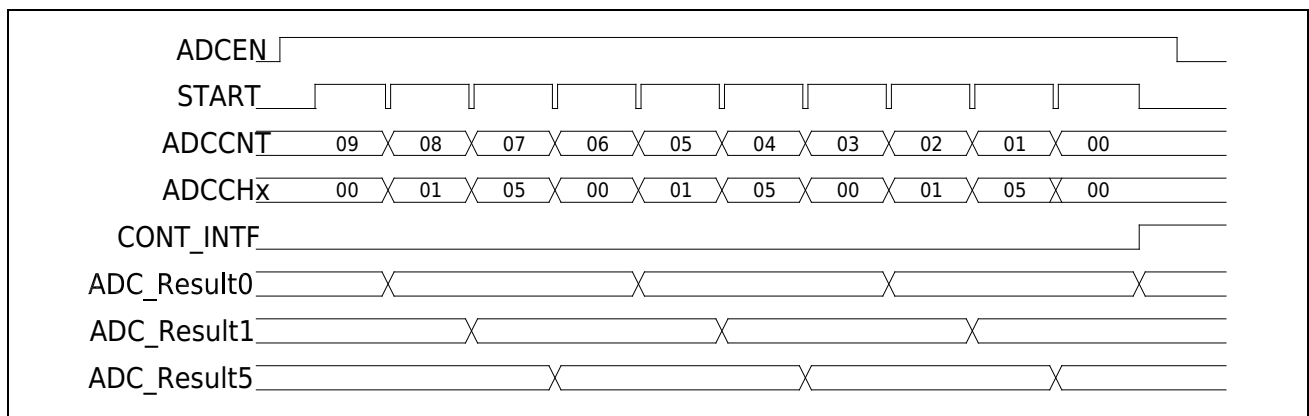


Figure 21-3 Example Of ADC Continuous Conversion Process

Start the ADC continuous conversion operation process through the START bit:

Step1: Configure the corresponding bits of P0ADS~P3ADS, and configure the ADC channel to be converted as an analog port.

Step2: Set P3ADS.6 to 1, and 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 BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.ADCEN to 1 to enable the ADC module.

Step5: Delay 20uS, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.CT to 1, select continuous conversion mode.

Step7: Set ADC_CR1[14:12] to 0, and turn off the conversion result comparison function.

Step8: Configure ADC_CR2. ADCCNT to select the total number of conversions for continuous conversion.

Step9: Configure ADC_CR0. SREF to select the reference voltage of ADC.

Step10: Configure ADC_CR0. SAM and ADC_CR0. CLKSEL to set the conversion speed of ADC.

Step11: Configure ADC_CR2[7:0] to select the channel to be converted.

Step12: Set ADC_ICLR. CONT_INTC to 0, and clear the ADC_IFR. CONT_INTF flag.

Step13: Set ADC_CR0. StateRst to 1, reset the continuous conversion state.

Step14: Set ADC_CR0.START to 1 to start ADC continuous conversion.

Step15: Wait for ADC_IFR.CONT_INTF to become 1, read ADC_result0~ADC_result7 registers to obtain the conversion result of the corresponding channel.

Step16: To convert other channels, repeat Step11~Step15.

Step17: Set ADC_CR0.ADCEN and BGR_CR.BGR_EN to 0, turn off the ADC module and BGR module.

21.6 Continuous Conversion Accumulation Mode

In the continuous conversion and accumulation mode, starting the ADC once can perform multiple conversions on multiple channels and accumulate the results of each conversion; the ADC channels that can be converted are AIN0~AIN7. ADC conversions is configured by ADCCR2.ADCCNT; the channel to be converted is configured by ADC_CR2[7:0]. This mode can be started by setting the ADC_CR0.START bit or by setting the external trigger of ADC_CR2[9:0]. After starting the continuous conversion, the ADC module converts the channels to be converted in AIN0~AIN7 in turn until the total number of conversions is completed. After the ADC module completes the total number of conversions, the ADC_IFR. CONT_INTF bit will be automatically set to 1, and the accumulated value of the conversion result will be saved in the ADC_result_acc register.

The figure below demonstrates the process of accumulating 10 consecutive conversions of AIN0, AIN1, and AIN5. After setting START to 1 through the register, the state machine inside the ADC will convert AIN0, AIN1, and AIN5 in sequence until the count value of ADCCNT becomes 0.

ADC_result_acc register is automatically accumulated each time a conversion is complete. The conversion results of AIN0, AIN1, and AIN5 given in the figure are 0x010, 0x020, and 0x040 in turn.

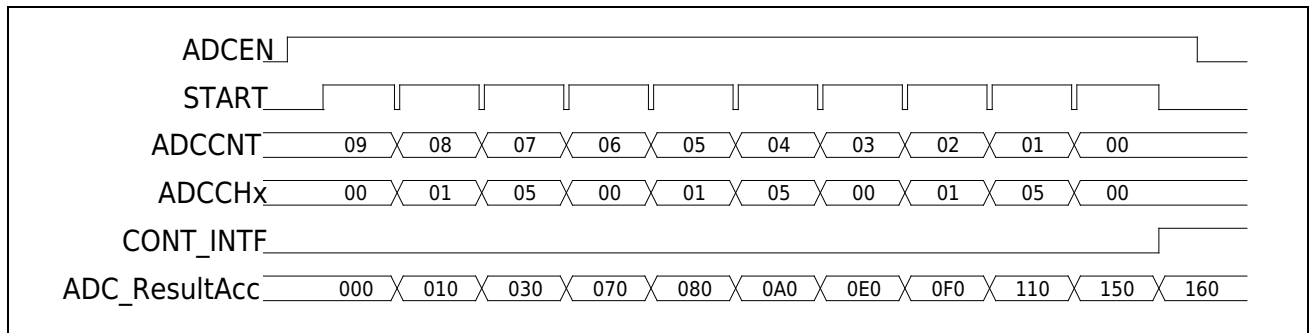


Figure 21-4 ADC Continuous Conversion Accumulation Process Example

Start the ADC continuous conversion and accumulation operation process through the START bit:

Step1: Configure the corresponding bits of P0ADS~P3ADS, and configure the ADC channel to be converted as an analog port.

Step2: Set P3ADS.6 to 1, and 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 BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.ADCEN to 1 to enable the ADC module.

Step5: Delay 20uS, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.CT to 1, select continuous conversion mode.

Step7: Set ADC_CR1[14:12] to 0, and turn off the conversion result comparison function.

Step8: Set ADC_CR1. RACC_EN to 1 to enable the automatic accumulation function of ADC conversion.

Step9: Configure ADC_CR2. ADCCNT to select the total number of conversions for continuous conversion.

Step10: Configure ADC_CR0. SREF to select the reference voltage of ADC.

Step11: Configure ADC_CR0. SAM and ADC_CR0. CLKSEL to set the conversion speed of ADC.

Step12: Configure ADC_CR2[7:0] to select the channel to be converted.

Step13: Set ADC_ICLR. CONT_INTC to 0, and clear the ADC_IFR. CONT_INTF flag.

Step14: Set ADC_CR1. RACC_CLR to 0, clear the ADC_result_acc register.

Step15: Set ADC_CR0. StateRst to 1, reset the continuous conversion state.

Step16: Set ADC_CR0.START to 1 to start ADC continuous conversion.

Step17: Wait for ADC_IFR. CONT_INTF to become 1, read the ADC_result_acc register to obtain the accumulated value of the conversion result.

Step18: To convert other channels, repeat Step12~Step17.

Step19: Set ADC_CR0.ADCEN and BGR_CR.BGR_EN to 0, turn off the ADC module and BGR module.

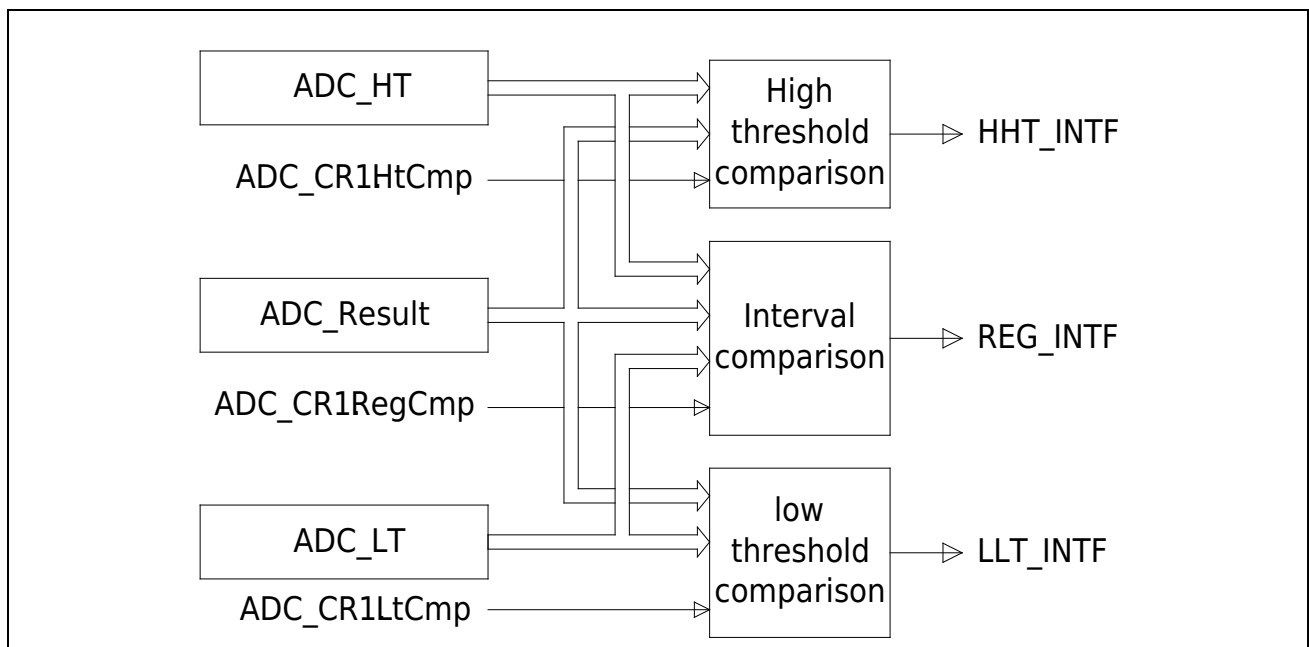
21.7 ADC conversion result comparison

When the ADC conversion is completed, the ADC conversion result can be compared with the threshold set by the user, supporting upper threshold comparison, lower threshold comparison, and interval value comparison. This function needs to set the corresponding control bits HtCmp, LtCmp, RegCmp to 1. This function can realize the automatic monitoring of the analog quantity, until the ADC conversion result meets the user's expectations, an interrupt is generated to apply for user program interface entry.

Upper threshold comparison: When the ADC conversion result is within the range [ADC_HT, 4095], ADC_IFR.HHT_INTF is set to 1; writing 0 to ADC_ICLR.HHT_INTC clears ADC_IFR.HHT_INTF.

Lower threshold comparison: when the ADC conversion result is in the range [0, ADC_LT), ADC_IFR.LLT_INTF is set to 1; writing 0 to ADC_ICLR.LLT_INTC clears ADC_IFR.LLT_INTF.

Interval value comparison: When the ADC conversion result is within the [ADC_LT, ADC_HT) interval, ADC_IFR.REG_INTF is set to 1; writing 0 to ADC_ICLR.REG_INTC clears ADC_IFR.REG_INTF.



21.8 ADC Interrupt

The ADC interrupt request is shown in the table below:

Interrupt source	Interrupt logo	Interrupt enable
ADC continuous conversion completed	ADC_IFR.CONT_INTF	ADC_CR0.IE
The ADC conversion result is in the interval value area	ADC_IFR.REG_INTF	
The ADC conversion result is in the upper threshold region	ADC_IFR.HHT_INTF	
ADC conversion result comparison lower threshold area	ADC_IFR.LLT_INTF	

21.9 Measure ambient temperature using a temperature sensor

The output voltage of the temperature sensor will change with the change of the ambient temperature, so the corresponding ambient temperature can be calculated according to the output voltage of the temperature sensor. When the measurement channel of the ADC module selects the output voltage of the temperature sensor, the ambient temperature can be measured.

Calculated as follows:

$$\text{Ambient Temperature} = 25 + 0.0839 \times V_{\text{ref}} \times (\text{AdcValue} - \text{Trim})$$

Among them: V_{ref} is the reference voltage of the current ADC module, and the value is 1.5 or 2.5.

AdcValue is the result of measuring the output voltage of the temperature sensor by the ADC module, and the value is 0~4095.

Trim is a 16-bit calibration value, which needs to be read from the Flash memory during calculation, and its storage address is detailed in the table below.

ADC reference voltage	Calibration value storage address	Calibration value accuracy
Internal 1.5V	0x00100C34	±5°C
Internal 2.5V	0x00100C36	±5°C

An example calculation is as follows:

Condition 1: $V_{\text{ref}}=2.5$, $\text{AdcValue}=0x7E5$, $\text{Trim}=0x76C$:

Temperature 1: $25 + 0.0839 \times 2.5 \times (0x7E5 - 0x76C) = 50^{\circ}\text{C}$.

Condition 2: $V_{\text{ref}}=1.5$, $\text{AdcValue}=0x72D$, $\text{Trim}=0x76C$:

Temperature 3: $25 + 0.0839 \times 1.5 \times (0x72D - 0x76C) = 17^{\circ}\text{C}$.

Operation process of measuring ambient temperature through ADC:

Step1: Set BGR_CR.BGR_EN to 3, enable BGR module and temperature sensor module.

Step2: Set ADC_CR0.ADCEN to 1 to enable the ADC module.

Step3: Delay 20uS, wait for the ADC and BGR module to start up.

Step4: Set ADC_CR1.CT to 0, select single conversion mode.

Step5: Configure ADC_CR0. SREF, select the reference voltage of ADC as internal 1.5V or internal 2.5V.

Step6: Configure ADC_CR0. SAM and ADC_CR0. CLKSEL to set the conversion speed of ADC.

Step7: Set ADC_CR0. SEL to 0x0A, select the channel to be converted as the output of the temperature sensor.

Step8: Set ADC_CR0. BUFEN to 1 to enable the input signal amplifier.

Step9: Set ADC_CR0.START to 1 to start ADC single conversion.

Step10: Wait for ADC_CR0.START to become 0, read the ADC_result register to get the ADC conversion result.

Step11: Set ADC_CR0.ADCEN and BGR_CR.BGR_EN to 0, turn off the ADC module and BGR module.

Step12: Read the calibration value of the temperature sensor and calculate the current ambient temperature according to the formula.

21.10 ADC module registers

Table 21-1 ADC register

Base address 0x40002400

Register	Offset address	Description
ADC_CR0	0x004	ADC Configuration Register 0
ADC_CR1	0x008	ADC Configuration Register 1
ADC_CR2	0x00C	ADC Configuration Register 2
ADC_result0	0x030	ADC channel 0 conversion result
ADC_result1	0x034	ADC channel 1 conversion result
ADC_result2	0x038	ADC channel 2 conversion result
ADC_result3	0x03C	ADC channel 3 conversion result
ADC_result4	0x040	ADC channel 4 conversion result
ADC_result5	0x044	ADC channel 5 conversion result
ADC_result6	0x048	ADC channel 6 conversion result
ADC_result7	0x04C	ADC channel 7 conversion result
ADC_result 8	0x050	ADC channel 8 conversion result
ADC_result_acc	0x054	Accumulated value of ADC conversion result
ADC_HT	0x058	ADC comparison upper threshold
ADC_LT	0x05C	ADC comparison lower threshold
ADC_IFR	0x060	ADC Interrupt Flag Register
ADC_ICLR	0x064	ADC Interrupt Clear Register
ADC_result	0x068	ADC conversion result

21.10.1 ADC Configuration Register 0 (ADC_CR0)

Offset address 0x004

Reset value 0x000013F0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
State Rst	IE	Res.	SAM		BUFE N	SREF		SEL			CLKSEL		STAR T	ADCE N	
R/W	R/W		R/W		R/W	R/W		R/W			R/W		R/W	R/W	

Bit	Marking	Functional description
31:16	Reserved	Keep
15	StateRst	ADC continuous conversion state control 1: Reset ADC continuous conversion status 0: invalid
14	IE	ADC interrupt control 1: enable interrupt 0: disable interrupt
13	Reserved	Keep
12:11	SAM	ADC sampling period selection 00: 4 sampling periods 01: 6 sampling periods 10: 8 sampling periods 11: 12 sampling periods
10	BUFEN	ADC input signal amplifier control 0: Turn off the amplifier, and the external input signal is directly connected to the ADC. 1: Turn on the amplifier, the external input signal is amplified by the amplifier and then connected to the ADC for high-impedance signals.
9:8	SREF	ADC reference voltage selection 00: Internal 1.5V 01: Internal 2.5V 10: External reference voltage ExRef (P3.6) 11: Supply voltage
7:4	SEL	ADC conversion channel selection (single conversion mode) 0000: select channel 0 input P2.4 0001: Select channel 1 input P2.6 0010: Select channel 2 input P3.2 0011: Select channel 3 input P3.3 0100: Select channel 4 input P3.4 0101: Select channel 5 input P3.5 0110: Select channel 6 input P3.6 0111: Select channel 7 input P0.1 1000: Select channel 8 input P0.2 1001: VCC/3

21.10.2 ADC Configuration Register 1 (ADC_CR1)

Offset address 0x008

Reset value 0x00007000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RACC CLR	RegC mp	HtCm p	LtCm p	HtCm p	CT	TRIGS1					TRIGS0				
R/W	R/W	R/W	R/W	R/W	R/W	R/W					R/W				

Bit	Marking	Functional description
31:16	Reserved	Keep
15	RACC_CLR	ADC conversion result accumulation register is cleared 1: no effect; 0: ADC conversion result accumulation register (ADC_result_acc) is cleared. Note: This bit is read as 0, so special attention should be paid to the value of this bit when operating this register to prevent misoperation.
14	RegCmp	ADC interval comparison control 1: enable range comparison 0: interval comparison disabled
13	HtCmp	ADC high threshold compare control 1: enable high threshold compare 0: disable high threshold comparison
12	LtCmp	ADC low threshold compare control 1: enable low threshold compare 0: disable low threshold comparison
11	RACC_EN	ADC conversion result automatic accumulation control 1: enable the automatic accumulation function of ADC conversion results 0: disable the automatic accumulation function of ADC conversion results
10	CT	ADC conversion mode selection 1: Continuous conversion mode 0: single conversion mode
9:5	TRIGS1	ADC conversion automatic trigger selection 1: 00000: Disable automatic triggering of ADC conversions 00001: Timer0 interrupt, trigger ADC conversion automatically 00010: Timer1 interrupt, automatically trigger ADC conversion 00011: Timer2 interrupt, automatically trigger ADC conversion 00100: LPTimer interrupt, trigger ADC conversion automatically 00101: Timer4 interrupt, automatically trigger ADC conversion 00110: Timer5 interrupt, automatically trigger ADC conversion 00111: Timer6 interrupt, automatically trigger ADC conversion 01000: UART0 interrupt, automatically trigger ADC conversion 01001: UART1 interrupt, automatically trigger ADC conversion 01010: LPUART interrupt, trigger ADC conversion automatically 01011: VC0 interrupt, automatically trigger ADC conversion 01100: VC1 interrupt, automatically trigger ADC conversion 01101: RTC interrupt, automatically trigger ADC conversion 01110: PCA interrupt, automatically trigger ADC conversion 01111: SPI interrupt, automatically trigger ADC conversion 10000: P01 interrupt, automatically trigger ADC conversion 10001: P02 interrupt, automatically trigger ADC conversion 10010: P03 interrupt, automatically trigger ADC conversion 10011: P14 interrupt, automatically trigger ADC conversion 10100: P15 interrupt, automatically trigger ADC conversion 10101: P23 interrupt, automatically trigger ADC conversion 10110: P24 interrupt, automatically trigger ADC conversion 10111: P25 interrupt, automatically trigger ADC conversion 11000: P26 interrupt, automatically trigger ADC conversion 11001: P27 interrupt, automatically trigger ADC conversion 11010: P31 interrupt, automatically trigger ADC conversion 11011: P32 interrupt, automatically trigger ADC conversion 11100: P33 interrupt, automatically trigger ADC conversion 11101: P34 interrupt, automatically trigger ADC conversion 11110: P35 interrupt, automatically trigger ADC conversion 11111: P36 interrupt, automatically trigger ADC conversion

		Note: 1) TIM4/5/6 interrupt triggers ADC automatic conversion. In addition to enabling the corresponding interrupt of TIM4/5/6, it is also necessary to configure TIMx_CR to select the interrupt source that can trigger the ADC. 2) The ADC is triggered by the rising edge of each interrupt flag. 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.
4:0	TRIGS0	ADC conversion automatic trigger selection 0: 00000: Disable automatic triggering of ADC conversions 00001: Select Timer0 interrupt to automatically trigger ADC conversion 00010: Select Timer1 interrupt to automatically trigger ADC conversion 00011: Select Timer2 interrupt to automatically trigger ADC conversion 00100: Select LPTimer interrupt to automatically trigger ADC conversion 00101: Select Timer4 interrupt to automatically trigger ADC conversion 00110: Select Timer5 interrupt to automatically trigger ADC conversion 00111: Select Timer6 interrupt to automatically trigger ADC conversion 01000: Select UART0 interrupt to automatically trigger ADC conversion 01001: Select UART1 interrupt to automatically trigger ADC conversion 01010: Select LPUART interrupt to trigger ADC conversion automatically 01011: Select VC0 interrupt to automatically trigger ADC conversion 01100: Select VC1 interrupt to automatically trigger ADC conversion 01101: Select RTC interrupt, automatically trigger ADC conversion 01110: Select PCA interrupt to automatically trigger ADC conversion 01111: Select SPI interrupt to automatically trigger ADC conversion 10000: select P01 interrupt, automatically trigger ADC conversion 10001: Select P02 interrupt, automatically trigger ADC conversion 10010: Select P03 interrupt to automatically trigger ADC conversion 10011: Select P14 interrupt to automatically trigger ADC conversion 10100: Select P15 interrupt to automatically trigger ADC conversion 10101: Select P23 interrupt to automatically trigger ADC conversion 10110: Select P24 interrupt to automatically trigger ADC conversion 10111: Select P25 interrupt to automatically trigger ADC conversion 11000: Select P26 interrupt to automatically trigger ADC conversion 11001: Select P27 interrupt to automatically trigger ADC conversion 11010: Select P31 interrupt to automatically trigger ADC conversion 11011: Select P32 interrupt to automatically trigger ADC conversion 11100: Select P33 interrupt to automatically trigger ADC conversion 11101: Select P34 interrupt to automatically trigger ADC conversion 11110: Select P35 interrupt to automatically trigger ADC conversion 11111: Select P36 interrupt to automatically trigger ADC conversion Note: 1) TIM4/5/6 interrupt triggers ADC automatic conversion. In addition to enabling the corresponding interrupt of TIM4/5/6, it is also necessary to configure TIMx_CR to select the interrupt source that can trigger the ADC. 2) The ADC is triggered by the rising edge of each interrupt flag. 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.

21.10.3 ADC Configuration Register 2 (ADC_CR2)

Offset address 0x00C

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCCNT								CH7E N	CH6E N	CH5E N	CH4E N	CH3E N	CH2E N	CH1E N	CH0E N
R/W								R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Functional description
31:16	Reserved	Keep
15:8	ADCCNT	ADC continuous conversion times configuration 0: Continuous conversion once 1: Continuous conversion 2 times 255: Continuous conversion 256 times
7	CH7EN	ADC continuous conversion channel 7 enable 1: Enable 0: Prohibited
6	CH6EN	ADC continuous conversion channel 6 enable 1: Enable 0: Prohibited
5	CH5EN	ADC continuous conversion channel 5 enable 1: Enable 0: Prohibited
4	CH4EN	ADC continuous conversion channel 4 enable 1: Enable 0: Prohibited
3	CH3EN	ADC continuous conversion channel 3 enable 1: Enable 0: Prohibited
2	CH2EN	ADC continuous conversion channel 2 enable 1: Enable 0: Prohibited
1	CH1EN	ADC continuous conversion channel 1 enable 1: Enable 0: Prohibited
0	CH0EN	ADC continuous conversion channel 0 enable 1: Enable 0: Prohibited

21.10.4 ADC channel 0 conversion result (ADC_result0)

Offset address 0x030

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				result0											
				RO											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	result0	ADC channel 0 conversion result

21.10.5 ADC channel 1 conversion result (ADC_result1)

Offset address 0x034

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				result1											
				RO											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	result1	ADC channel 1 conversion result

21.10.6 ADC channel 2 conversion result (ADC_result2)

Offset address 0x038

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				result2											
				RO											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	result2	ADC channel 2 conversion result

21.10.7 ADC channel 3 conversion result (ADC_result3)

Offset address 0x03C

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				result3											
				RO											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	result3	ADC channel 3 conversion result

21.10.8 ADC channel 4 conversion result (ADC_result4)

Offset address 0x040

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				result4											
				RO											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	result4	ADC channel 4 conversion result

21.10.9 ADC channel 5 conversion result (ADC_result5)

Offset address 0x044

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				result5											
				RO											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	result5	ADC channel 5 conversion result

21.10.10 ADC channel 6 conversion result (ADC_result6)

Offset address 0x048

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				result6											
				RO											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	result6	ADC channel 6 conversion result

21.10.11 ADC channel 7 conversion result (ADC_result7)

Offset address 0x04C

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				result7											
				RO											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	result7	ADC channel 7 conversion result

21.10.12 ADC channel 8 conversion result (ADC_result8)

Offset address 0x050

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				result8											
				RO											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	result8	ADC channel 8 conversion result

21.10.13 Accumulated value of ADC conversion result (ADC_result_acc)

Offset address 0x054

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												Result_acc[19:16]			
												RO			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Result_acc[15:0]															
RO															

Bit	Marking	Functional description
31:20	Reserved	Keep
19:0	Result_acc	ADC conversion accumulated value

21.10.14 ADC Compare Upper Threshold (ADC_HT)

Offset address 0x058

Reset value 0x00000FFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				HT											
				R/W											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	HT	ADC conversion result comparison upper threshold

21.10.15 ADC Compare Lower Threshold (ADC_LT)

Offset address 0x05C

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				LT											
				R/W											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	LT	ADC conversion result comparison lower threshold

21.10.16 ADC Interrupt Flag Register (ADC_IFR)

Offset address 0x060

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												CONT INTF	REG INTF	HHT INTF	LLT INTF
												RO	RO	RO	RO

Bit	Marking	Functional description
31:4	Reserved	Keep
3	CONT_INTF	Continuous Conversion Complete Flag 1: ADC continuous conversion completed 0: ADC continuous conversion not completed
2	REG_INTF	ADC conversion result comparison interval flag 1: The ADC conversion result is within the range of [ADC_LT, ADC_HT) 0: The ADC conversion result is outside the [ADC_LT, ADC_HT) range
1	HHT_INTF	ADC conversion result comparison upper threshold flag 1: The ADC conversion result is within the range [ADC_HT, 4095] 0: ADC conversion result is outside [ADC_HT, 4095] range
0	LLT_INTF	ADC conversion result comparison lower threshold flag 1: The ADC conversion result is in the range [0, ADC_LT) 0: The ADC conversion result is outside the interval [0, ADC_LT)

21.10.17 ADC Interrupt Clear Register (ADC_ICLR)

Offset address 0x064

Reset value 0x00000004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												CONT INTC	REG INTC	HHT INTC	LLT INTC
												W0	W0	W0	W0

Bit	Marking	Functional description
31:4	Reserved	Keep
3	CONT_INTC	Write 0 to clear the continuous conversion complete flag Write 1 has no effect
2	REG_INTC	Write 0 to clear the ADC conversion result comparison interval flag Write 1 has no effect
1	HHT_INTC	Write 0 to clear ADC conversion result comparison upper threshold Write 1 has no effect
0	LLT_INTC	Write 0 to clear ADC conversion result comparison lower threshold flag Write 1 has no effect

21.10.18 ADC result (ADC_result)

Offset address 0x068

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				result											
				RO											
Bit	Marking			Functional description											
31:12	Reserved			Keep											
11:0	result			ADC conversion results											

22 Analog Comparator (VC)

22.1 Introduction to Analog Voltage Comparator VC

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 flat. The analog voltage comparator VC integrated in this product has the following characteristics:

- Support voltage comparison function;
- Support internal 64- stage VCC voltage divider (use the voltage divider source voltage to be greater than 1.8V);
- Support 8 external input ports as the input of the voltage comparator;
- Support three software-configurable interrupt trigger modes: high level trigger / rising edge trigger / falling edge trigger;
- The output of the voltage comparator can be used as the input of the gate control port of Base Timer and LPTimer;
- The output of the voltage comparator can be used as the brake input or capture input of the Advanced Timer;
- Support working in ultra-low power consumption mode, the interrupt output of the voltage comparator can wake up the chip from ultra-low power consumption mode;
- Provide software configurable filter time to enhance the anti-interference ability of the chip.

22.2 Voltage Comparator Block Diagram

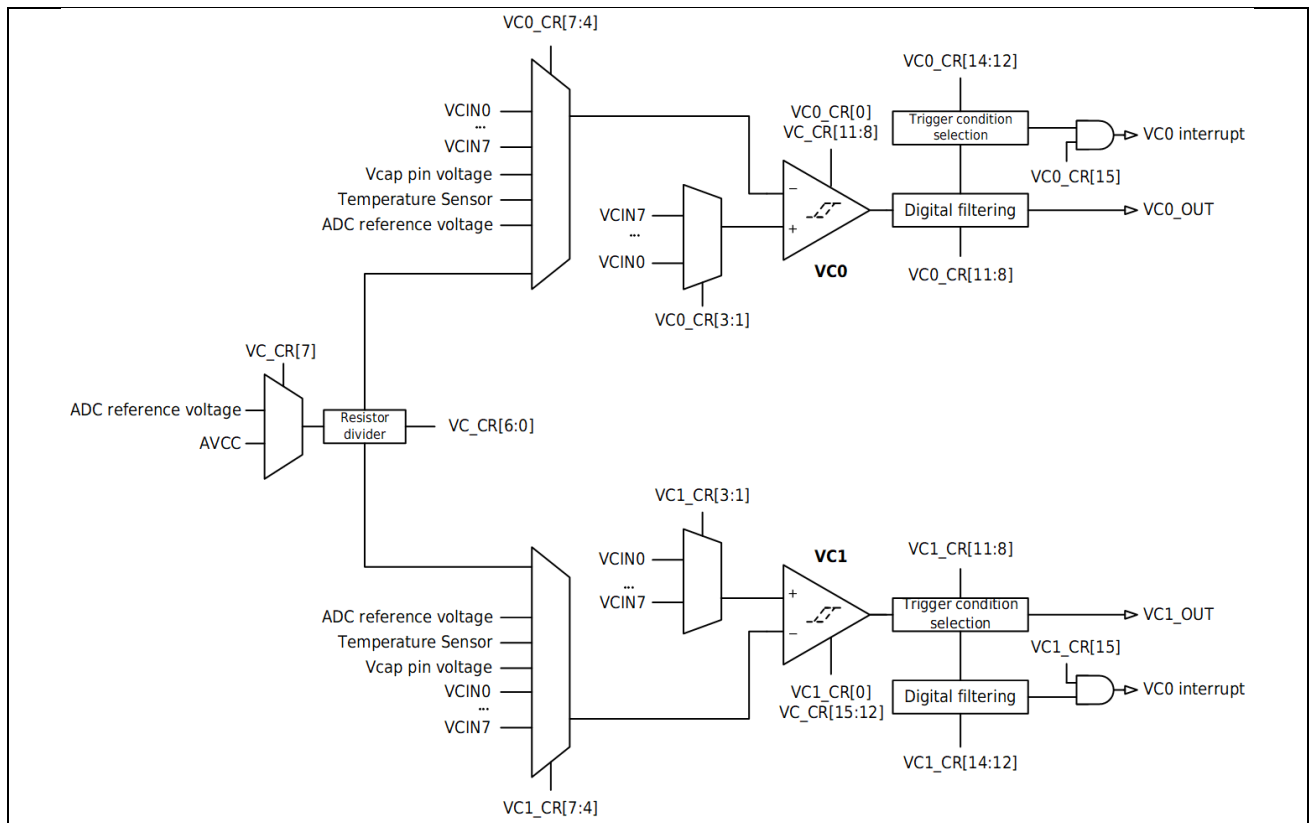


Figure 22-1 VC Frame Diagram

22.3 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 the correct result is output is determined by the BIAS_SEL control bit in the VC control register (VC_CR).

If the temperature sensor, and ADC module reference voltage are selected as the terminal input of the comparator, the internal BGR module needs to be turned on. The start-up time of the internal BGR is about 20us, and the voltage comparator needs to wait for the internal BGR to stabilize before outputting normally.

22.4 Filter time

In addition to the inherent settling/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 digitally filtered signal level can be read from the register VC_IFR.VCx_Filter; when the GPIO function is configured as VCx_OUT, the digitally filtered signal can be output from the GPIO for easy measurement.

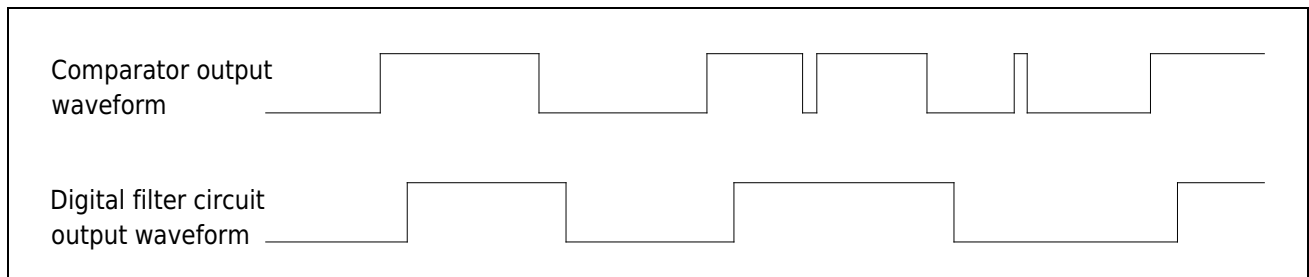


Figure 22-2 VC Filter Response Time

22.5 Hysteresis function

The hysteresis function can be selected for the voltage comparator, and the diagram after the hysteresis function is enabled is as follows.

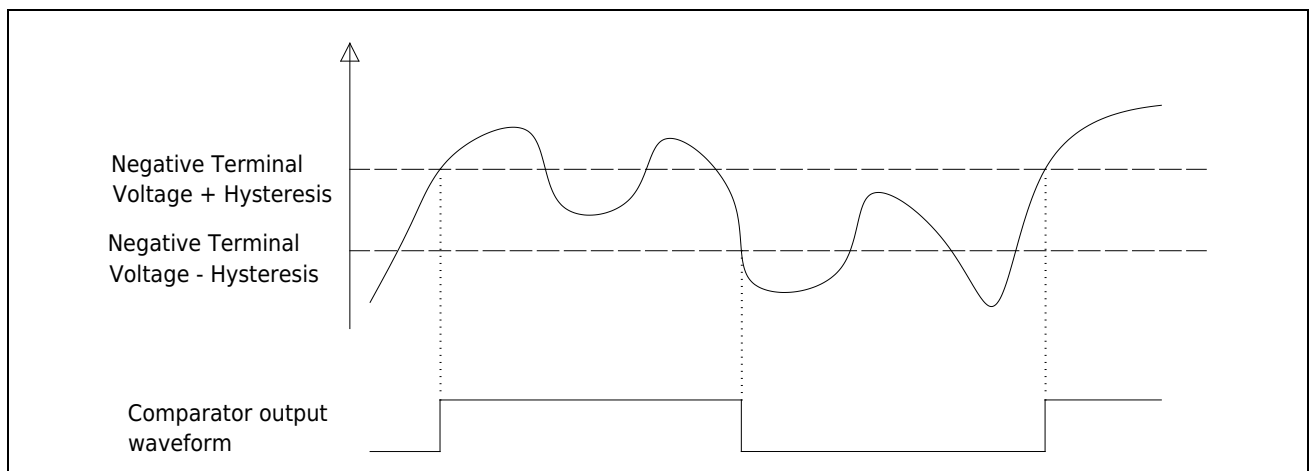


Figure 22-3 VC hysteresis function

22.6VC register

Table 22-1 VC register

Base address 0x40002400

Register	Offset address	Description
VC_CR	0x010	VC0/1 Configuration Register 0
VC0_CR	0x014	VC0 configuration register
VC1_CR	0x018	VC1 configuration register
VC0_OUT_CFG	0x01C	VC0 output configuration register
VC1_OUT_CFG	0x020	VC1 Output Configuration Register
VC_IFR	0x024	VC interrupt register

22.6.1 VC Configuration Register (VC_CR)

Offset address 0x010

Reset value 0x00000020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VC1_HYS_SEL		VC1_BIAS_SEL		VC0_HYS_SEL		VC0_BIAS_SEL		VC_REF2P5_SEL	VC_DIV_EN	VC_DIV					
R/W		R/W		R/W		R/W		R/W	R/W	R/W					

Bit	Marking	Functional description
31:16	Reserved	Keep
15:14	VC1_HYS_SEL	VC1 hysteresis selection: 00: no hysteresis 01: Hysteresis voltage about 10mV 10: The hysteresis voltage is about 20mV 11: The hysteresis voltage is about 30mV
13:12	VC1_BIAS_SEL	VC1 power consumption selection (the greater the power consumption, the faster the response speed) 00:300nA 01:1.2uA 10:10uA (need power supply voltage not less than 2.8V, need to manually turn on BGR) 11:20uA (need power supply voltage not less than 2.8V, need to manually open BGR) Note: BGR startup time is about 30us
11:9	VC0_HYS_SEL	VC0 hysteresis selection: 00: no hysteresis 01: Hysteresis voltage about 10mV 10: The hysteresis voltage is about 20mV 11: The hysteresis voltage is about 30mV
9:8	VC0_BIAS_SEL	VC0 power consumption selection (the greater the power consumption, the faster the response speed) 00:300nA 01:1.2uA 10:10uA (need power supply voltage not less than 2.8V, need to manually turn on BGR) 11:20uA (need power supply voltage not less than 2.8V, need to manually open BGR) Note: BGR startup time is about 30us
7	VC_REF2P5_SEL	VC_DIV reference voltage Vref selection 0:VCC 1: The reference voltage selected by ADC_CR0.SREF
6	VC_DIV_EN	6-bit DAC enable 1: Enable
5:0	VC_DIV	6-bit DAC configuration 000000: 1/64 Vref 000001:2/64 Vref 000010:3/64 Vref 000011:4/64 Vref ... 111110:63/64 Vref 111111: Vref

22.6.2 VC0 Configuration Register (VC0_CR)

Offset address 0x014

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	level	rising	falling	debounce_time			FLTEN	n_sel				p_sel		EN	
R/W	R/W	R/W	R/W	R/W			R/W	R/W				R/W		R/W	

Bit	Marking	Functional description
31:16	Reserved	Keep
15	IE	VC interrupt enable 1: enable; 0: disable
14	level	VC output signal high level triggers INT flag
13	rising	VC output signal rising edge triggers INT flag
12	Falling	VC output signal falling edge triggers INT flag
11:9	debounce_time	VC output filter time configuration 111: The filtering time is about 28.8ms 110: The filtering time is about 7.2ms 101: The filtering time is about 1.8ms 100: The filtering time is about 450us 011: The filtering time is about 112us 010: The filtering time is about 28us 001: The filtering time is about 14us 000: The filtering time is about 7us Note: The filter time configuration is only valid when FLTEN=1.
8	FLTEN	1: start VC filter 0: VC without filter
7:4	N_SEL	Voltage comparator "-" terminal input selection 0000: select channel 0 input P2.3 0001: select channel 1 input P2.5 0010: select channel 2 input P3.2 0011: select channel 3 input P3.3 0100: select channel 4 input P3.4 0101: select channel 5 input P3.5 0110: select channel 6 input P3.6 0111: select channel 7 input P0.1 1000: Resistor divider output voltage 1001: Built-in temperature sensor output voltage 1010: Reserved 1011: Reference voltage of ADC module 1100: VCAP pin voltage
3:1	P_SEL	Voltage comparator "+" Terminal input selection 000: select channel 0 input P2.3 001: select channel 1 input P2.5 010: select channel 2 input P3.2 011: select channel 3 input P3.3 100: select channel 4 input P3.4 101: select channel 5 input P3.5 110: select channel 6 input P3.6 111: select channel 7 input P0.1
0	EN	Comparator Enable 1: enable voltage comparator 0: disable voltage comparator

22.6.3 VC1 Configuration Register (VC1_CR)

Offset address 0x018

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	level	rising	falling	debounce_time			FLTEN	n_sel				p_sel		EN	
R/W	R/W	R/W	R/W	R/W			R/W	R/W				R/W		R/W	

Bit	Marking	Functional description
31:16	Reserved	Keep
15	IE	VC interrupt enable 1: enable; 0: disable
14	level	VC output signal triggers interrupt selection 1: enable high level trigger INT flag 0: disable high level triggering INT flag
13	rising	VC output signal triggers interrupt selection 1: enable rising edge to trigger INT flag 0: disable rising edge triggering INT flag
12	falling	VC output signal triggers interrupt selection 1: enable falling edge trigger INT flag 0: disable falling edge triggering INT flag
11:9	debounce_time	VC output filter time configuration 111: The filtering time is about 28.8ms 110: The filtering time is about 7.2ms 101: The filtering time is about 1.8ms 100: The filtering time is about 450us 011: The filtering time is about 112us 010: The filtering time is about 28us 001: The filtering time is about 14us 000: The filtering time is about 7us Note: The filter time configuration is only valid when FLTEN=1.
8	FLTEN	1: start VC filter 0: VC without filter
7:4	N_SEL	Voltage comparator "-" terminal input selection 0000: select channel 0 input P2.3 0001: select channel 1 input P2.5 0010: select channel 2 input P3.2 0011: select channel 3 input P3.3 0100: select channel 4 input P3.4 0101: select channel 5 input P3.5 0110: select channel 6 input P3.6 0111: select channel 7 input P0.1 1000: Resistor divider output voltage 1001: Built-in temperature sensor output voltage 1010: Reserved 1011: Reference voltage of ADC module 1100: VCAP pin voltage
3:1	P_SEL	Voltage comparator "+" Terminal input selection 000: select channel 0 input P2.3 001: select channel 1 input P2.5 010: select channel 2 input P3.2 011: select channel 3 input P3.3 100: select channel 4 input P3.4 101: select channel 5 input P3.5 110: select channel 6 input P3.6 111: select channel 7 input P0.1
0	EN	Comparator Enable 1: enable voltage comparator 0: disable voltage comparator

22.6.4 VC0 Output Configuration Register (VC0_OUT_CFG)

Offset address 0x01C

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
brake	TIM6	INV_T imer 6	TIM5	INV_T imer 5	TIM4	INV_T imer4	PCAE CI	PCAC AP0	INV_P CA	TM3E CLK	LPTI MG	TIM2 G	TIM1 G	TIM0 G	INV_T imer
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Functional description
31:16	Reserved	Keep
15	brake	VC0 as Advanced Timer brake control 1: enable; 0: disable.
14	TIM6	VC0 filter result output to TIM6 capture input enable 1: enable; 0: disable.
13	INV_TIM6	VC0 filter result output to TIM6 reverse enable 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.
12	TIM5	VC0 filter result output to TIM5 capture input enable 1: enable; 0: disable.
11	INV_TIM5	VC0 filter result output to TIM5 reverse enable 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.
10	TIM4	VC0 filter result output to TIM4 capture input enable 1: enable; 0: disable.
9	INV_TIM4	VC0 filter result output to TIM4 reverse enable 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.
8	PCAECI	VC0 filter result output to PCA external clock enable control 1: enable; 0: disable.
7	PCACAP0	VC0 filter result output to PCA capture 0 enable control 1: enable; 0: disable.
6	INV_PCA	VC0 filter result output negative to PCA 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.
5	LPTIMECLK	VC0 filter result output to LPTIMER external clock enable control 1: enable; 0: disable.
4	LPTIMG	VC0 filter result output to LPTIMER3 GATE enable control 1: enable; 0: disable.
3	TIM2G	VC0 filter result output to TIM2 GATE enable control 1: enable; 0: disable.
2	TIM1G	VC0 filter result output to TIM1 GATE enable control 1: enable; 0: disable.
1	TIM0G	VC0 filter result output to TIM0 GATE enable control 1: enable; 0: disable.
0	INV_Timer	VC0 filter result output is negative to each TIM0/1/2, LPTimer 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.

22.6.5 VC1 Output Configuration Register (VC1_OUT_CFG)

Offset address 0x020

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
brake	TIM6	INV_T imer 6	TIM5	INV_T imer 5	TIM4	INV_T imer4	PCAE CI	PCAC AP1	INV_P CA	TM3E CLK	LPTI MG	TIM2 G	TIM1 G	TIM0 G	INV_T imer
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Functional description
31:16	Reserved	Keep
15	brake	VC1 as Advanced Timer brake control 1: enable; 0: disable.
14	TIM6	VC1 filter result output to TIM6 capture input enable 1: enable; 0: disable.
13	INV_TIM6	VC1 filter result output to TIM6 reverse enable 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.
12	TIM5	VC1 filter result output to TIM5 capture input enable 1: enable; 0: disable.
11	INV_TIM5	VC1 filter result output to TIM5 reverse enable 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.
10	TIM4	VC1 filter result output to TIM4 capture input enable 1: enable; 0: disable.
9	INV_TIM4	VC1 filter result output to TIM4 reverse enable 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.
8	PCAECI	VC1 filter output to PCA external clock enable control 1: enable; 0: disable.
7	PCACAP1	VC1 filter result output to PCA capture 1 enable control 1: enable; 0: disable.
6	INV_PCA	VC1 filter result output negative to PCA 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.
5	LPTIMECLK	VC1 filter result output to LPTIMER external clock enable control 1: enable; 0: disable.
4	LPTIMG	VC1 filter result output to LPTIMER3 GATE enable control 1: enable; 0: disable.
3	TIM2G	VC1 filter result output to TIM2 GATE enable control 1: enable; 0: disable.
2	TIM1G	VC1 filter result output to TIM1 GATE enable control 1: enable; 0: disable.
1	TIM0G	VC1 filter result output to TIM0 GATE enable control 1: enable; 0: disable.
0	INV_Timer	VC1 filter result output is negative to each TIM0/1/2, LPTimer 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.

22.6.6 VC Interrupt Register (VC_IFR)

Offset address 0x024

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												VC1_Filter	VC0_Filter	VC1_INTF	VC0_INTF
												RO	RO	R/W	R/W

Bit	Marking	Functional description
31:4	Reserved	Keep
3	VC1_Filter	Status after VC1 Filter
2	VC0_Filter	Status after VC0 Filter
1	VC1_INTF	VC1 interrupt flag, 1 VC1 interrupt occurs; 0 no interrupt occurs; write 0 to clear the interrupt flag, write 1 is invalid
0	VC0_INTF	VC0 interrupt flag, 1 VC0 interrupt occurs; 0 no interrupt occurs; writing 0 clears the interrupt flag, writing 1 is invalid

23 Low Voltage Detector (LVD)

23.1 Introduction to LVD

LVD can be used to monitor the voltage of VCC 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.

LVD has the following characteristics:

- 4 monitoring sources, VCC, P03, P23, P25;
- 16-level threshold voltage, flexible and multi-purpose;
- 8 trigger conditions, combinations of high level, rising edge and falling edge;
- 2 trigger results, reset and interrupt;
- 8-stage filter configuration to prevent false triggering;
- With hysteresis function, strong anti-interference.

23.2 LVD block diagram

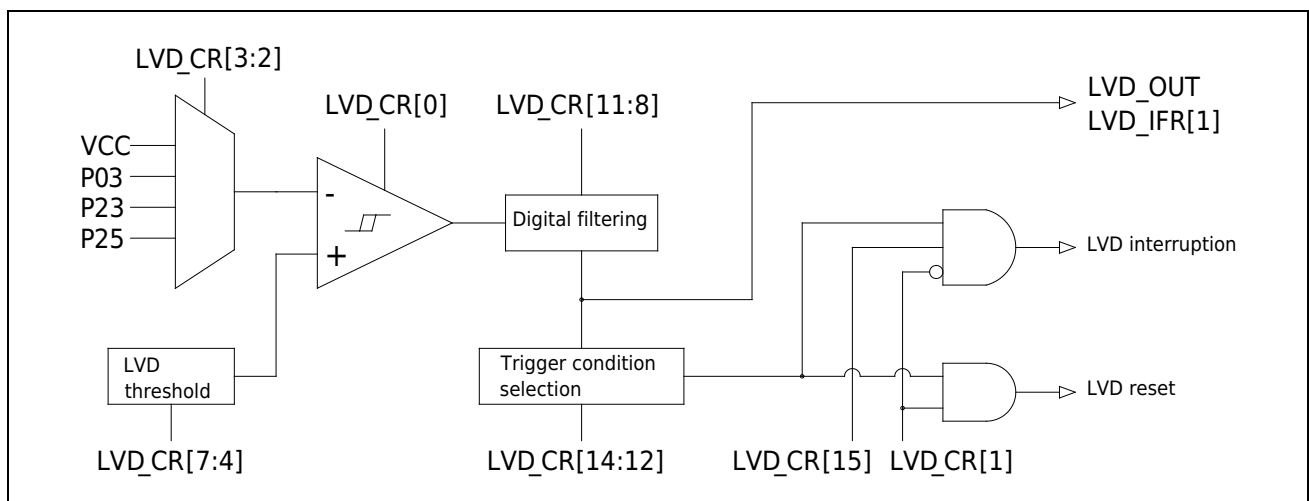


Figure 23-1 LVD Block Diagram

23.3 Digital filtering

If the working environment of the chip is bad, the output of the hysteresis comparator will appear noise signal. When the digital filtering module is enabled, 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 digitally filtered signal level can be read from the register LVD_IFR[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 filtering module, and the filtering diagram is as follows:

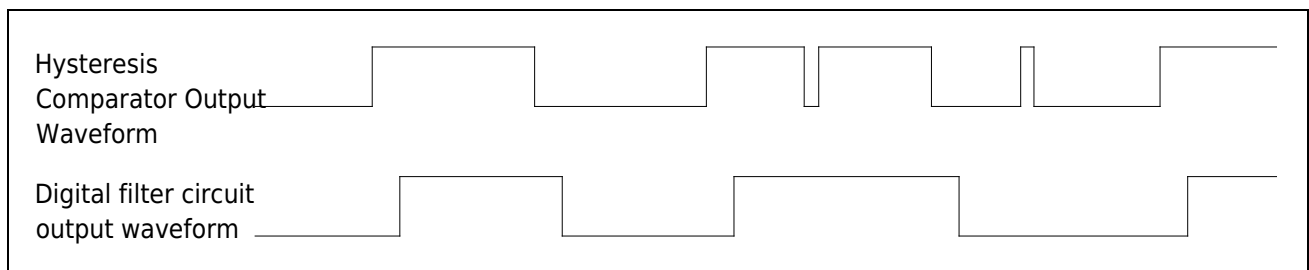


Figure 23-2 LVD Filtered Output

23.4 Hysteresis function

The built-in voltage comparator of LVD has a hysteresis function, and its output signal will not flip until the input signal is 20mV higher or lower than the threshold voltage. The hysteresis function can enhance the anti-interference ability of the chip, as shown in the figure below.

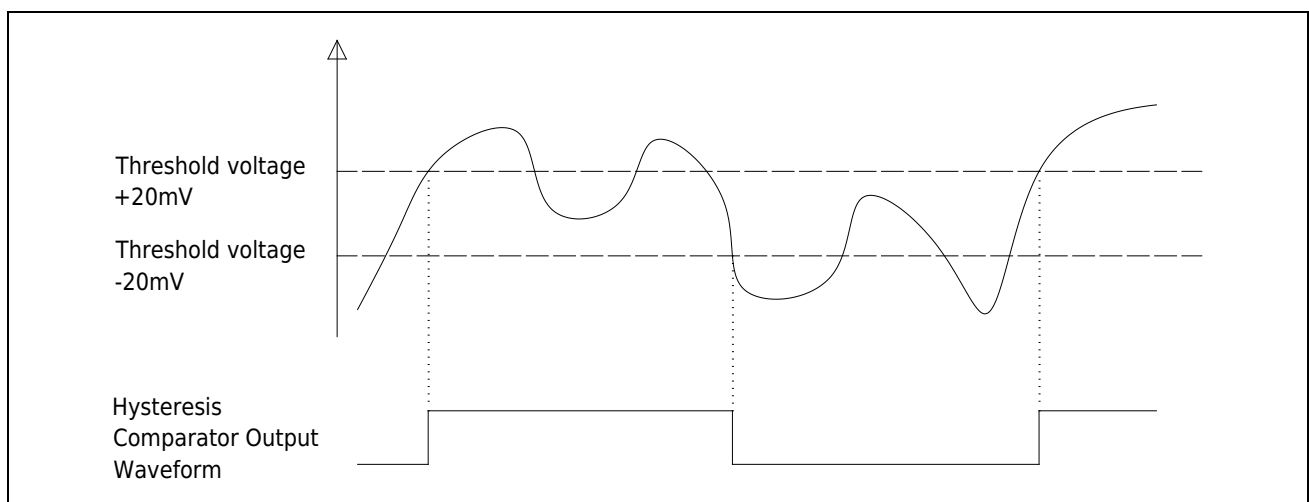


Figure 23-3 LVD Hysteretic Response

23.5 Configuration example

23.5.1 LVD configured as low voltage reset

In this mode, the MCU is reset when the monitored voltage is lower than the threshold voltage.

The configuration method is as follows:

Step1: Configure LVD_CR.Source_sel to select the voltage source to be monitored.

Step2: Configure LVD_CR.VTDS, select the threshold voltage of LVD.

Step3: Configure LVD_CR.Debounce_time, select LVD filter time.

Step4: Configure LVD_CR.FLTEN to enable LVD filtering.

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.LVDEN to 1 to enable LVD.

23.5.2 LVD configured as voltage change interrupt

In this mode, an interrupt is generated when the monitored voltage goes above or below the threshold voltage.

The configuration method is as follows:

Step1: Configure LVD_CR.Source_sel to select the voltage source to be monitored.

Step2: Configure LVD_CR.VTDS, select the threshold voltage of LVD.

Step3: Configure LVD_CR.Debounce_time, select LVD filter time.

Step4: Configure LVD_CR.FLTEN to enable LVD filtering.

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.LVDEN to 1 to enable LVD.

Step10: Write 0x00 to LVD_IFR in the interrupt service routine to clear the interrupt flag.

23.6 LVD register

Table 23-1 LVD register

Base address 0x40002400

Register	Offset address	Description
LVD_CR	0x028	LVD configuration register
LVD_IFR	0x02C	LVD Interrupt Flag Register

23.6.1 LVD configuration register (LVD_CR)

Offset address 0x028

Reset value 0x00000100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	HTEN	RTEN	FTEN	Debounce_time			FLT EN	VTDS			Source_sel		ACT	LVD EN	
R/W	R/W	R/W	R/W	R/W			R/W	R/W			R/W		R/W	R/W	

Bit	Marking	Functional description
31:16	Reserved	Keep
15	IE	LVD interrupt enable 1: Enable; 0: Disabled.
14	HTEN	High level trigger enable (the monitored voltage is lower than the threshold voltage) 1: Enable; 0: Disabled.
13	RTEN	Rising edge trigger enable (the monitored voltage changes from higher than the threshold voltage to lower than the threshold voltage) 1: Enable; 0: Disabled.
12	FTEN	Falling edge trigger enable (the monitored voltage changes from lower than the threshold voltage to higher than the threshold voltage) 1: Enable; 0: Disabled.
11:9	Debounce_time	Digital filter time configuration 111: The filtering time is about 28.8ms 110: The filtering time is about 7.2ms 101: The filtering time is about 1.8ms 100: The filtering time is about 450us 011: The filtering time is about 112us 010: The filtering time is about 28us 001: The filtering time is about 14us 000: The filtering time is about 7us Note: The filter time is only valid when FLTEN is 1.
8	FLTEN	Digital filter function configuration 1: Enable digital filtering 0: disable digital filtering
7:4	VTDS	LVD monitoring voltage selection 1111: 3.3v 1110: 3.2v 1101: 3.1v 1100: 3.0v 1011: 2.9v 1010: 2.8v 1001: 2.7v 1000: 2.6v 0111: 2.5v 0110: 2.4v 0101: 2.3v 0100: 2.2v 0011: 2.1v 0010: 2.0v 0001: 1.9v

		0000: 1.8v
3:2	Source_sel	LVD monitoring source selection 11: P2.5 port input voltage 10: P2.3 port input voltage 01: P0.3 port input voltage 00: VCC supply voltage
1	ACT	LVD trigger action selection 1: system reset 0: NVIC interrupt
0	LVDEN	LVD control 1: Enable LVD 0: disable LVD

23.6.2 LVD Interrupt Register (LVD_IFR)

Offset address 0x02C

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														LVD_Filte	INTF
														RO	R/W

Bit	Marking	Functional description
31:2	Reserved	Keep
1	LVD_Filte	Status after LVD Filter
0	INTF	LVD interrupt flags: 1: LVD interrupt occurred; 0: No interrupt occurred; Writing 0 clears the interrupt flag, writing 1 has no effect.

24 Simulate other registers

Base address 0x40002400

Register	Offset address	Description
BGR_option	0x078	BGR Control Register

24.1 BGR Configuration Register (BGR_CR)

Offset address 0x000

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														TS_E N	BGR_ EN
														R/W	R/W

Bit	Marking	Functional description
31:2	Reserved	Keep
1	TS_EN	Internal temperature sensor enable 1: enable internal temperature sensor 0: disable internal temperature sensor Note: TS needs about 20us startup time to be stable.
0	BGR_EN	BGR enable control 1: enable BGR 0: disable BGR Note: 1) This register can only be operated when PERI_CLKEN.ADC is 1. 2) After BGR is enabled for 20us, a stable high-precision reference voltage can be output. After the BGR is stable, it can be used by other modules, so the steps of waiting for the BGR to be stable should be added in the user operation. 3) When using ADC, BGR must be enabled. 4) When using VC, it is necessary to decide whether to enable BGR according to the configuration of the VC register.

25 SWD debug interface

This series uses the ARM Cortex-M0+ core, which has a hardware debug 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 kernel is stopped, both the internal state of the kernel 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 the HC32L110 microcontroller is connected to the debugger and starts debugging, the debugger will use the kernel's hardware debugging module for debugging.

Note:

- SWD can't work in DeepSleep mode, please debug in Active and Sleep mode.

25.1 SWD debugging additional functions

This product uses the ARM Cortex-M0+ CPU, which includes hardware extensions for advanced debugging functions, so this product has the same debugging functions as Cortex-M0+. Debug extensions allow the kernel to halt the kernel when fetching instructions (instruction breakpoints) or fetching data (data breakpoints). When the kernel stops, you can query the internal state of the kernel and the external state of the system. After the query completes, the kernel and system are restored and program execution is restored.

When the debug host is connected to the MCU and debugged, the debug function will be used.

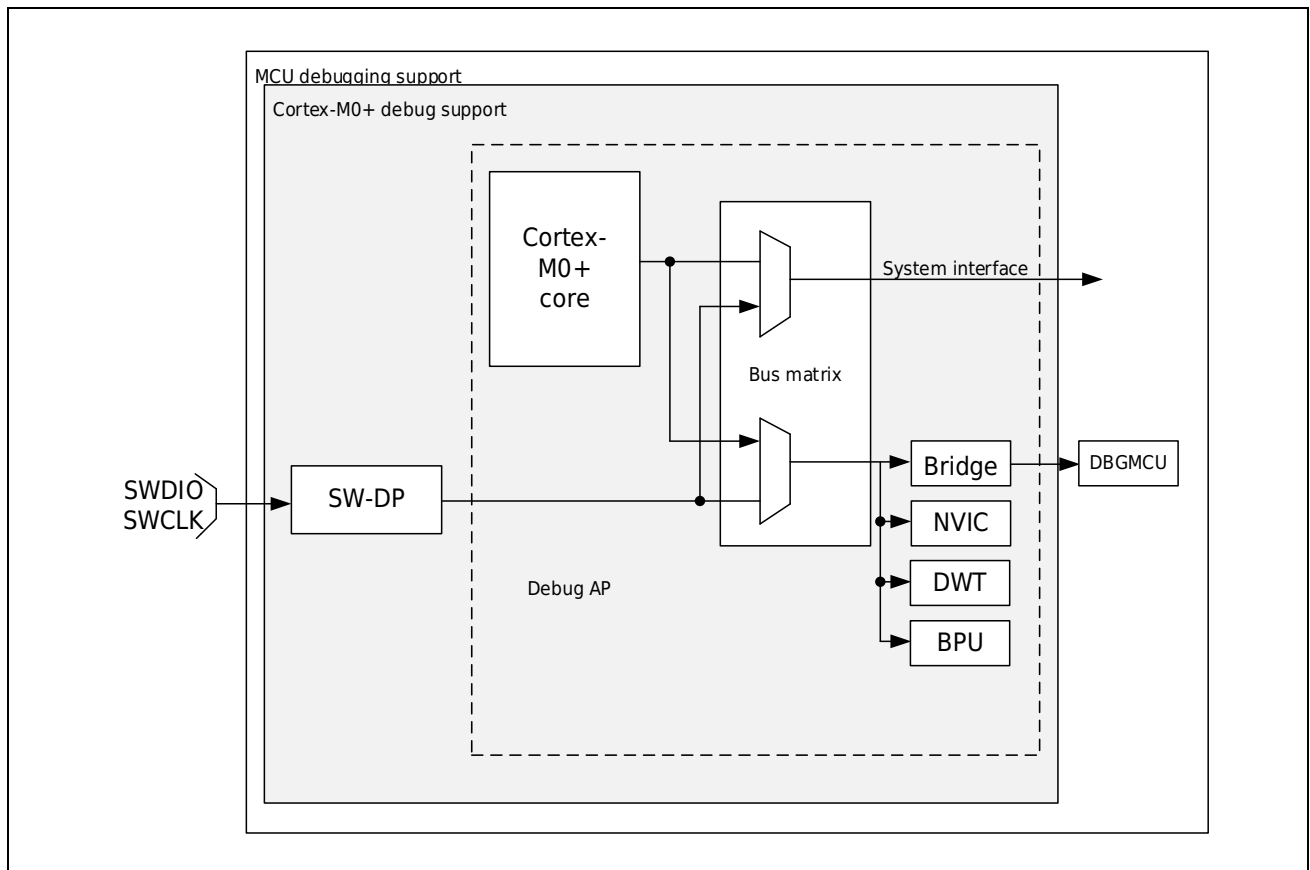


Figure 25-1 Debug Support Block Diagram

Debug features built into the Cortex®-M0+ core are part of the ARM® CoreSight Design Suite.

The ARM® Cortex®-M0+ core provides integrated on-chip debugging support. It includes:

- SW-DP: serial line
- BPU: Breakpoint unit
- DWT: data watchpoint trigger

Note:

- Details on the debug features supported by the ARM® Cortex®-M0+ core can be found in the Cortex® -M0+ Technical Reference Manual.

25.2 ARM® Reference Documentation

- Cortex®-M0+ Technical Reference Manual (TRM)
Available from www.infocenter.arm.com.
- ARM® Debug Interface V5
- ARM® CoreSight Design Suite Version r1p1 Technical Reference Manual

25.3 Debug port pins

25.3.1 SWD port pin

The SWD interface of this series requires 2 pins, as shown in the table below.

SWD port name	Debug function	Pin assignment
SWCLK	Serial clock	P31
SWDIO	Serial data input / output	P27

25.3.2 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 [encrypted chip] option is not enabled when programming the program, the P27/P31 pins will be initialized as dedicated pins that can be used by the debugger after power-on. Users can set the SYSCTRL1.SWD_USE_IO register to disable the SWD pin debug function, and the SWD pin will be released to be used as a normal GPIO. SWD pins are summarized in the following table:

[Encryption chip] option	SWD_USE_IO configuration	P27/P31 function
Encryption	0	NA
Encryption	1	GPIO
No encryption	0	SWD
No encryption	1	GPIO

25.3.3 Internal pull-up on SWD pin

After the user software releases the SW I/O, the GPIO controller takes control of these pins. GPIO control register puts the I/O into the equivalent state:

- SWDIO: input pull-up
- SWCLK: input pull-up

No external resistors need to be added due to built-in pull-up and pull-down resistors.

25.4 SWD port

25.4.1 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. When transferring data, the LSB comes first.

For SWDIO bidirectional management, the line must be pulled up on the board (ARM® recommends 100 k). These pull-up resistors are internally configurable. No external pull-up resistors are 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 it can be adjusted by configuring the SWCLK frequency.

25.4.2 SWD protocol sequence

Each sequence consists of three phases:

1. The host sends a packet request (8 bits)
2. Acknowledgment response sent by target (3 digits)
3. The host or target sends the data transfer phase (33 bits)

Bit	Name	Note
0	Start up	Must be 1
1	APnDP	0: DP access; 1: AP access
2	RnW	0: write request; 1: read request
4:3	A[3:2]	Address field of DP or AP register
5	Parity	Unit parity for the first few bits
6	Stop	0
7	Reside	Not driven by the host. 1 by target due to pull-up

For a detailed description of the DPACC and APACC registers, refer to the Cortex®-M0+ TRM.

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.

Bit	Name	Note
0	ACK	001: FAULT 010: WAIT 100: OK

The ACK response must be followed by transition time only when a READ transaction occurs or a WAIT or FAULT acknowledgment is received.

Bit	Name	Note
0:31	WDATA or RDATA	Write or Read data
32	Parity	Single parity for 32 data bits

The transfer time must be after the DATA transfer only when a READ transaction occurs.

25.4.3 SW-DP state machine (reset, idle state, ID code)

SW-DP has an internal ID code for identifying SW-DP. The code is JEP-106 compliant. This ID code is the default ARM® code, set to 0x0BC11477 (equivalent to 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.
- Cortex®-M0+ TRM and CoreSight Design Suite r1p0TRM for more details on the SW-DP state machine.*

25.4.4 DP and AP read / write access

- 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

(operations that are only valid after the core is powered on) are executed immediately, which will cause a failure.

25.4.5 SW-DP register

These registers can be accessed when APnDP=0:

A[3:2]	RW	CTRLSEL bit of the SELECT register	Register	Note
00	Read		IDCODE	Manufacturer code set to default ARM® for Cortex®-M0+ Code. 0x0BC11477 (identifies SW-DP)
00	Write		ABORT	
01	Read / Write	0	DP-CTRL/STAT	Purpose: - Request system or debug power up - Configure transport operations for AP access - Control compare and verify operations - read some status flags (overflow and power-on acknowledgment)
01	Read / Write	1	WIRE CONTROL	Used to configure the physical serial port protocol (such as switching time duration)
10	Read		READ RESEND	Allow recovery of reads from corrupted debug software transfers data without repeating the original AP transmission.
10	Write		SELECT	4 -word register used to select the currently accessed port and active window
11	Read / Write		READ BUFFER	Since the AP access has been issued, this read buffer is very useful Used (provide read AP request when executing the next AP transaction the result of). This read buffer captures data from the AP, shown as

The result of a previous read, without initiating a new operation.

25.4.6 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	Note
0x0	00	Reserved, must remain at reset value.
0x4	01	DP CTRL/STAT register. Used for: - Request system or debug power up - Configure transport operations for AP access - Control compare and verify operations - read some status flags (overflow and power-on acknowledgment)
0x8	10	DP SELECT Register: Used to select the currently accessed port and active 4 - word register window. - Bits 31:24: APSEL: Select the current AP (select the current AP) - Bits 23:8: Reserved - Bits 7:4: APBANKSEL: Select the active 4- word register window on the current AP - Bits 3:0: Reserved
0xC	11	DP RDBUFF register: used to obtain the final result after performing a series of operations through the debugger (No need to request new JTAG-DP operations)

25.5 Kernel debugging

Debug the kernel through the kernel debug registers. Debug access to these registers is through the debug access port. It consists of four registers:

Register	Note
DHCSR	<i>32-bit debug stop control and status register</i> 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	<i>17-bit debug core register selector registers:</i> This register selects the processor register to be read or written.
DCRDR	<i>32-bit Debug Core Registers Data Registers:</i> This register holds the data read and written between the register and the processor selected by the DCRSR (selector) register.
DEMCR	<i>32-bit Debug Exception and Watchdog Control Register:</i> 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. See Cortex®-M0+ TRM for more details.

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)

25.6 BPU (Breakpoint Unit)

The Cortex®-M0+ BPU implementation provides four breakpoint registers.

25.6.1 BPU function

Processor breakpoint implements PC- based breakpoint functionality.

See the ARMv6-M ARM® and ARM® CoreSight Components Technical Reference Manual for more information on the BPU CoreSight Identification Registers, their addresses and access types.

25.7 DWT (Data Watchpoint)

The Cortex®-M0+ DWT implementation provides two watchpoint register sets.

25.7.1 DWT function

Processor watchpoints implement data address and PC-based watchpoint functionality (i.e. PC sampling registers) and support for comparator address masks as described in ARMv6-M ARM®.

25.7.2 DWT Program Counter Sample 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 halting the processor. This provides a rough analysis. See ARMv6-M ARM® for more information.

Cortex®-M0+ DWT_PCSR records passing condition codes and instructions and instructions failing condition codes.

25.8 MCU debug group (DBG)

MCU Debug Component helps the debugger provide support for:

- Low power mode
- Timer during breakpoint, clock control of watchdog

25.8.1 Debug support for low power modes

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.

FCLK or HCLK during a debug 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 modes.

25.8.2 Debug support for timers, 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. For example, this is often required when PWM is controlling a motor.
- When a breakpoint is generated, the counter stops counting. This is required when used with a watchdog.

25.9 Debug mode module working state control (DEBUG_ACTIVE)

Reset value 0x00000FFF (this register setting has an effect only in SWD debug mode)

Offset address: 0x038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				LPTIM	Res.	RTC	WDT	PCA	TIM6	TIM5	TIM4	LPTIM	TIM2	TIM1	TIM0
				RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:12	Reserved	Keep
11	LPTIM	When debugging, Timer3 count function configuration 1: In the SWD debugging interface, the Timer3 counting function is suspended 0: In the SWD debugging interface, Timer3 counts normally
10	Reserved	Keep
9	RTC	When debugging, the RTC count function configuration 1: In the SWD debugging interface, suspend the RTC counting function 0: In the SWD debugging interface, the RTC counts normally
8	WDT	When debugging, the WDT count function configures the 1: In the SWD debugging interface, suspend the WDT counting function 0: In the SWD debugging interface, WDT counts normally
7	PCA	When debugging, the PCA counting function configures 1: In the SWD debugging interface, the PCA counting function is suspended 0: In the SWD debugging interface, PCA counts normally
6	TIM6	When debugging, Timer6 counting function configuration 1: In the SWD debugging interface, the Timer counting function is suspended 0: In the SWD debugging interface, the Timer counts normally
5	TIM5	When debugging, Timer5 count function configuration 1: In the SWD debugging interface, the Timer counting function is suspended 0: In the SWD debugging interface, the Timer counts normally
4	TIM4	When debugging, Timer4 count function configuration 1: In the SWD debugging interface, the Timer counting function is suspended 0: In the SWD debugging interface, Timer counts normally
3	LPTIM	When debugging, the LpTimer count function configures the 1: In the SWD debugging interface, the Timer counting function is suspended 0: In the SWD debugging interface, Timer counts normally
2	TIM2	When debugging, Timer2 count function configuration 1: In the SWD debugging interface, the Timer counting function is suspended 0: In the SWD debugging interface, the Timer counts normally
1	TIM1	When debugging, the Timer1 count function configures the 1: In the SWD debugging interface, the Timer counting function is suspended 0: In the SWD debugging interface, the Timer counts normally
0	TIM0	When debugging, Timer0 count function configuration 1: In the SWD debugging interface, the Timer counting function is suspended 0: In the SWD debugging interface, the Timer counts normally

26 Device electronic signature

The electronic signature is stored in the system storage area of the flash memory module and can be read by SWD or CPU. HC32Fxxx / HC32Lxxx microcontrollers with different configurations.

26.1 Product Unique Identifier (UID) Register (80 bits)

Typical application scenarios for unique identifiers:

- 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 80-bit unique device identifier provides a reference number that is unique to any device and in any context. The user can never change these bits. The 80-bit UID can also be read in different ways like single byte/halfword/word and then concatenated using a custom algorithm.

Base address: 0x0010 0E74

Offset address	Description	UID Bit(80bit)							
		7	6	5	4	3	2	1	0
0	X Coordinate on the wafer	UID[7:0]							
1	Rev ID	UID[15:8]							
2	Fixed value FFH	Reserved							
3	Fixed value FFH	Reserved							
4	Fixed value 00H	UID[23:16]							
5	Fixed value 00H	UID[31:24]							
6	Wafer Number	UID[39:32]							
7	Y Coordinate on the wafer	UID[47:40]							
8	Wafer Lot Number	UID[55:48]							
9		UID[63:56]							
10		UID[71:64]							
11		UID[79:72]							

26.2 Product Model Register

0x0010 0C60 ~ 0x0010 0C6F stores the ASCII code of the product model. If the product model is less than 16 bytes, fill it with 0x00.

Example: The product model represented by 484333324C3133364B38544100000000 is HC32L136K8TA.

26.3 FLASH capacity register

Base address: 0x0010 0C70

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FlashSize[31:16]															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FlashSize[15:0]															
R															
Bit	Marking	Functional description													
31:0	FlashSize	The capacity of the product's built-in Flash, in bytes 0x00008000 means the Flash capacity is 32K Byte													

26.4 RAM capacity register

Base address: 0x0010 0C74

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RamSize[31:16]															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RamSize[15:0]															
R															
Bit	Marking	Functional description													
31:0	RamSize	The capacity of the product's built-in RAM, in bytes 0x00000800 means the RAM capacity is 2K Byte													

26.5 Pin Count Register

Base address: 0x0010 0C7A

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PinCount[15:0]															
R															
Bit	Marking	Functional description													
15:0	PinCount	The number of product pins, in unit 0x0020 means that the number of product pins is 32													

27 Appendix A SysTick Timer

27.1 Introduction to SysTick Timers

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. There is a simple timer called SysTick in the Cortex-M0 processor, 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, timing, or providing an interrupt source for tasks that need to be executed periodically. SysTick exceptions is controllable. If the exceptions are disabled, the SysTick timer can still be used by polling, such as checking the current count value or polling the overflow flag.

27.2 Set SysTick

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:

Step1: Configure SysTick->CTRL. ENABLE is 0, disable SysTick.

Step2: Configure SysTick->CTRL. CLKSOURCE and SYSTICK_CR[27:25] to select the clock source of SysTick.

Step3: Configure SysTick->LOAD, select the overflow period of SysTick.

Step4: Write any value to SysTick->VAL, clear SysTick->VAL and SysTick->CTRL. COUNTFLAG.

Step5: Configure SysTick->CTRL. TICKINT is 1, enable SysTick interrupt.

Step6: Set Sys Tick->CTRL. ENABLE to 1 to enable SysTick.

Step7: Read SysTick->CTRL in the interrupt service routine to clear the overflow flag.

Note: The SysTick overflow period is SysTick->LOAD+1, the configuration example is as follows:

Timer	SysTick->LOAD	Overflow cycle
RCH 4M	3999	1ms
XTL 32.768K	327	10.01ms

27.3 SysTick register

Address	Name	CMSIS symbol	Full name
0XE000E010	SYS_CSR	SysTick->CTRL	SysTick Control and Status Register
0XE000E014	SYS_RVR	SysTick->LOAD	SysTick reload value register
0XE000E018	SYS_CVR	SysTick->VAL	SysTick current value register

27.3.1 SysTick Control and Status Register (CTRL)

Bit	Symbol	Functional description	Types of	Reset value
31:17	Reserved	-	-	-
16	COUNTFLAG	Systick timer overflow flag 1: Systick timer underflow occurred. 0: Systick timer overflow has not occurred. Reading this register clears the COUNTFLAG flag	RO	0
15:3	Reserved	-	-	-
2	CLKSOURCE	SysTick clock source selection 1: Core clock (HCLK) 0: Reference clock, determined by SYSTICK_CR[27:25]	R/W	0
1	TICKINT	SysTick interrupt enable 1: enable interrupt 0: disable interrupt	R/W	0
0	ENABLE	SysTick timer enable 1: enable SysTick 0: disable SysTick	R/W	0

27.3.2 SysTick reload register (LOAD)

Bit	Symbol	Functional description	Types of	Reset value
31:24	Reserved	-	-	-
23:0	RELOAD	SysTick timer reload value	RW	-

27.3.3 SysTick Current Value Register (VAL)

Bit	Symbol	Functional description	Types of	Reset value
31:24	Reserved	-	-	-
23:0	CURRENT	Read this register to get the current count value of the SysTick timer Write any value to this register, clear this register and COUNTFLAG	RW	-

28 Appendix B Document Conventions

28.1 List of register-related abbreviations

The following abbreviations are used in register descriptions:

RW: Read and write, software can read and write these bits.

RO: Read only, software can only read these bits.

WO: Write only, software can only write to this bit. Reading this bit will return invalid data.

W1: Only write 1, the hardware automatically clears 0, writing 0 is invalid

R0W1: software reads this bit as 0, writes 1 to clear this bit. Writing a 0 has no effect on the value of this bit.

RW0: Software can read and write this bit, writing 1 is invalid, writing 0 clears

R1W0: Software reads this bit as 1, writes 0 to clear this bit. Writing a 1 has no effect on the value of this bit.

RC: Software can read this bit. When this bit is read, it is automatically cleared.
Writing "0" has no effect on the value of this bit.

Res, Reserved: Reserved bit, must remain at reset value.

28.2 Glossary

This section provides brief definitions of acronyms and abbreviations used in this document:

Word: 32 -bit data.

Half Word: 16 -bit data.

Byte: 8 -bit data.

IAP (In-Application Programming): IAP means that the microcontroller's Flash can be reprogrammed while the user program is running.

ICP (In-Circuit Programming): ICP means that the Flash of the microcontroller can be programmed using the JTAG protocol, SWD protocol or bootloader when the device is installed on the user application circuit board.

AHB: Advanced High Performance Bus.

APB: Low-speed peripheral bus.

DMA: Direct Memory Access.

TIM: timer

Version revision history

version number	Revision Date	modify the content
Rev1.00	2018/01/23	Released the first edition of the HC32L110 Series Reference Manual.
Rev1.10	2018/05/04	Version updated, corrected the Flash data, and modified part of the description in Chapter 4.
Rev1.20	2018/05/23	Modified the clock switching process, and added the PCA comparison capture function module setting.
Rev1.30	2018/11/01	Supplement the description of functional modules in Chapter 1, added Chapter 2 to describe the pin configuration and functions, supplement Chapter 9 FLASH operation description, modified Chapter 28 electrical characteristic parameters, and added Chapters 29 and 31.
Rev1.40	2018/11/26	Modified the name: UART2→LPUART. Added "Note" to Sections 2.1 and 2.2.
Rev1.50	2019/02/22	Corrected the following data: ①ADC characteristics ②ESD characteristics ③ECFLASH minimum value in memory characteristics ④Increased package size ⑤Added AVCC/AVSS in the pin configuration diagram.
Rev1.60	2019/07/05	Corrected the following data: ①Corrected the UID address ②Corrected the programming mode ③Updated the QFN pin configuration diagram style.
Rev1.70	2019/12/13	Corrected the following data: ①Typical application circuit diagram ②LVD block diagram ③ADC characteristic unit ④Voltage comparator frame diagram ⑤Flag bit register (UARTx_ISR) ⑥XTH and XTL diagram and precautions in external clock source characteristics ⑦Updated I2C bus (I2C), serial Peripheral Interface (SPI), Universal Synchronous Asynchronous Receiver Receiver (UART) and Low Power Synchronous Asynchronous Receiver Receiver (LPUART) section description.
Rev1.80	2020/01/17	Corrected the following data: ① Added CSP16 package ②Device electronic signature ③Appendix A SysTick timer ④Cyclic redundancy check (CRC) ⑤System control register 1 (SYSCTRL1) add note ⑥CR register (FLASH_CR) reset value ⑦I2C configuration register (I2C_CR) of 0 bits.
Rev1.90	2020/03/06	Corrected the following data: ①Added attention items in programming mode ②Step8 of 17.5.1 and 17.5.2.
Rev2.00	2020/04/30	Corrected the following data: ①Increased VCC/3 accuracy in ADC characteristics ②Corrected typos in external clock source characteristics ③RCL oscillator accuracy in internal clock source characteristics.
Rev2.10	2020/05/29	Corrected the following data: ①Added Step2 and Step3 in 17.5 ②Changed I2Cx to I2C.
Rev2.20	2020/07/31	Corrected the following data: ① Added TIM timer characteristics and communication interface section ② EFT level ③ Internal AHB/APB clock frequency in common working conditions ④ Corrected typos ⑤ Input characteristics in port characteristics - ports P0, P1, P2, P3, V _{IH} in RESET and the value of V _{IL} .
Rev2.30	2020/09/30	Corrected the following data: ①7.8.4 Updated typos ②Clock system description ③RCH oscillator accuracy in internal clock source characteristics ④V _{IL} and V _{IH} of RESETB pin characteristics ⑤Added SPI characteristics.
Rev2.31	2020/12/31	Deleted product features, pin configuration, packaging information, etc. (please refer to the latest datasheet for related information), and revised the statement.
Rev2.32	2022/03/09	The company logo is updated.
Rev2.33	2022/08/13	"I2C Bus" chapter, I2C operation process errata.
Rev2.40	2024/05/23	Corrected the following data: ① Added notes to 3.1 Clock Source Description; ② Deleted Px_DIR in Figure 6-1 port circuit diagram; ③ 17.5

version number	Revision Date	modify the content
		Programming Example Procedure Updated; ④ Modified the TRIGSx bit description in 21.10.2 ADC_CR1 register; ⑤ Deleted the 1.2V related content of the analog-to-digital converter (ADC) section; ⑥ Deleted the reference voltage related to 1.2V and on-chip BGR output in the section of analog Voltage Comparator (VC).
Rev2.41	2025/03/28	Deleted the description of the maximum communication rate between master and slave in Section 18.2 Main Features of SPI.
Rev2.50	2026/08/19	Corrected the following data: ① In the sections "15.7 Automatic Address Recognition" and "16.7 Automatic Address Recognition", modified the description of the broadcast address detected by the slave address detection; ② In the section 25.4 "SWD Port", changed "SW-DP core ID 0x0BB11477" to "0x0BC11477".