



## HC32L110 series

**32 -bit ARM® Cortex®-M0+ microcontroller**

User Manual



## statement

ÿ Huada Semiconductor Co., Ltd. (hereinafter referred to as "HDSC") reserves the right to change, correct, enhance, modify Huada Semiconductor products and/or at any time

rights to this document without notice. Users can obtain the latest relevant information before placing an order. HDSC products are purchased and sold according to the basic contract  
the stated terms and conditions of sale.

ÿ Customers should select the appropriate HDSC product for your application, and design, validate and test your application to ensure that your application meets the relevant requirements.

Compliance with standards and any safety, security or other requirements. The customer is solely responsible for this.

ÿ HSC hereby acknowledges that no license to any intellectual property rights, express or implied, is granted.

ÿ Resale of HDSC products on terms different from those set forth herein voids any HDSC warranty for such products.

ÿ Any graphics or words marked with “®” or “™” are trademarks of HDSC. All other products or services displayed on HDC products

Names are the property of their respective owners.

ÿ The information in this notice supersedes and replaces the information in previous editions.

©2020 Huada Semiconductor Co., Ltd. - All Rights Reserved

## content

statement.....	2
content.....	3
Introduction .....	.twenty three
1 system structure.....	.twenty four
1.1 Overview .....	.twenty four
1.2 System Addressing .....	25
1.3 Memory and module address assignments.....	27
2 Operating mode.....	28
2.1 Operating Mode .....	30
2.2 Sleep Mode .....	31
2.3 Deep Sleep Mode.....	33
3 System Controller (SYSCTRL) .....	36
3.1 Introduction to the clock source.....	36
3.1.1 Internal high-speed RC clock RCH.....	37
3.1.2 Internal low-speed RC clock RCL.....	37
3.1.3 External low-speed crystal clock XTL 3.1.4 .....	38
External high-speed crystal clock XTH.....	38
3.1.5 Clock Startup Procedure .....	39
3.2 System Clock Switching .....	40
3.2.1 Standard Clock Switching Flow .....	40
3.2.2 Example of switching from RCH to XTL.....	40
3.2.3 Example of switching from RCH to XTH.....	41
3.2.4 Example of switching from RCL to XTH.....	42
3.2.5 Example of switching from RCH to RCL.....	43
3.2.6 Example of switching from RCL to RCH.....	43
3.2.7 Switching between different oscillation frequencies of RCH.....	44
3.3 Clock Calibration Module.....	45
3.4 Interrupt wake-up control.....	46
3.4.1 The method of executing the interrupt service routine after waking up from deep sleep mode.....	46
3.4.2 The method of not executing the interrupt service routine after waking up from deep sleep mode.....	46
3.4.3 Using the Exit Hibernation feature.....	47
3.5 Registers.....	49
3.5.1 System Control Register 0 (SYSCTRL0).....	50
3.5.2 System Control Register 1 (SYSCTRL1).....	52
3.5.3 System Control Register 2 (SYSCTRL2).....	54
3.5.4 RCH Control Register (RCH_CR) .....	55
Register (XTH_CR) .....	3.5.6
RCL Control Register (RCL_CR) .....	57
3.5.7 XTL Control Register (XTL_CR) .....	Peripheral Module Clock 58
3.5.8 Control Register (PERI_CLKEN) .....	59



3.5.9	Systick Clock Control (SYSTICK_CR) .....	61
4	Reset the controller (RESET).....	62
4.1	Introducing the Reset Controller .....	62
4.1.1	Power-on and power-off reset POR/BOR.....	63
4.1.2	External reset pin reset.....	63
4.1.3	WDT reset.....	63
4.1.4	PCA reset.....	63
4.1.5	LVD Low Voltage Reset .....	63
4.1.6	Cortex-M0+ SYSRESETREQ reset.....	63
4.1.7	Cortex-M0+ LOCKUP reset.....	63
4.2	register.....	65
4.2.1	Reset Flag Register (Reset_flag) .....	4.2.2 Peripheral 65
	Modules Reset Control Register (PERI_RESET) .....	Interrupt Controller 66
5	(NVIC) .....	68
5.1	Overview .....	68
5.2	Interrupt Priority .....	68
5.3	Interrupt Vector Table.....	69
5.4	Interrupt Input and Suspend Behavior .....	70
5.5	Interrupt wait .....	73
5.6	Interrupt Sources .....	73
5.7	Interrupt Block Diagram .....	75
5.8	register.....	77
5.8.1	Interrupt Enable Setting Register (SCS_SETENA).....	77
5.8.2	Interrupt Enable Clear Register (SCS_CLRENA).....	78
5.8.3	Interrupt Pending Status Setting Register (SCS_SETPEND).....	78
5.8.4	Interrupt Pending Status Clear Register (SCS_CLRPEND).....	79
5.8.5	Interrupt Priority Register (SCS_IPR0).....	80
5.8.6	Interrupt Priority Register (SCS_IPR1).....	81
5.8.7	Interrupt Priority Register (SCS_IPR2).....	82
5.8.8	Interrupt Priority Register (SCS_IPR3).....	83
5.8.9	Interrupt Priority Register (SCS_IPR4).....	84
5.8.10	Interrupt Priority Register (SCS_IPR5).....	85
5.8.11	Interrupt Priority Register (SCS_IPR6) .....	86
5.8.12	Interrupt Priority Register (SCS_IPR7).....	87
5.8.13	Interrupt Mask Special Register (SCS_PRIMASK) .....	88
5.9	Basic software operation.....	89
5.9.1	External Interrupt Enable.....	89
5.9.2	NVIC Interrupt Enable and Clear Enable.....	89
5.9.3	NVIC Interrupt Pending and Clear Pending .....	89
5.9.4	NVIC Interrupt Priority .....	89
5.9.5	NVIC Interrupt Mask.....	90
6	Port Controller (GPIO) .....	91



6.1	Introduction to Port Controllers.....	91
6.2	Port Controller Key Features .....	92
6.3	Port Controller Functional Description .....	93
6.3.1	Port Configuration Functions.....	93
6.3.2	Writing to the port.....	95
6.3.3	Reading of Ports .....	96
6.3.4	Port multiplexing function.....	97
6.3.5	Port Interrupt Function .....	98
6.4	Port Configuration Operations .....	99
6.4.1	Port Multiplexing Operation Flow .....	99
6.4.2	Port Interrupt Operation Flow .....	100
6.4.3	Port Configuration Operation Flow .....	101
6.5	Port Controller Register Descriptions .....	102
6.5.1	Port P0.....	105
6.5.1.1	Port P01 function configuration register (P01_SEL).....	105
6.5.1.2	Port P02 function configuration register (P02_SEL).....	106
6.5.1.3	Port P03 function configuration register (P03_SEL).....	107
6.5.1.4	Port P0 I/O Configuration Register (P0DIR).....	108
6.5.1.5	Port P0 Input Value Register (P0IN) .....	109
6.5.1.6	Port P0 output value configuration register (P0OUT) .....	110
6.5.1.7	Port P0 digital-analog configuration register (P0ADS).....	111
6.5.1.8	Port P0 Drive Capability Configuration Register (P0DR).....	112
6.5.1.9	Port P0 Pull-Up Enable Configuration Register (P0PU).....	113
6.5.1.10	Port P0 pull-down enable configuration register (P0PD).....	114
6.5.1.11	Port P0 Open-Drain Output Configuration Register (P0OD).....	115
6.5.1.12	Port P0 High Interrupt Enable Configuration Register (P0HIE).....	116
6.5.1.13	Port P0 Low Level Interrupt Enable Configuration Register (P0LIE).....	117
6.5.1.14	Port P0 Rising Edge Interrupt Enable Configuration Register (P0RIE) .....	118
6.5.1.15	Port P0 Falling Edge Interrupt Enable Configuration Register (P0FIE).....	119
6.5.1.16	Port P0 Interrupt Status Register (P0_STAT) .....	120
6.5.1.17	Port P0 Interrupt Clear Register (P0_ICLR).....	121
6.5.2	Port P1.....	122
6.5.2.1	Port P14 function configuration register (P14_SEL).....	122
6.5.2.2	Port P15 function configuration register (P15_SEL).....	123
6.5.2.3	Port P1 I/O Configuration Register (P1DIR).....	124
6.5.2.4	Port P1 Input Value Register (P1IN) .....	125
6.5.2.5	Port P1 Output Value .....	125
6.5.2.6	Configuration Register (P1OUT) .....	126
6.5.2.7	Port P1 digital-analog configuration register (P1ADS).....	127
6.5.2.8	Port P1 Drive Capability Configuration Register (P1DR).....	128
6.5.2.9	Port P1 Pull-Up Enable Configuration Register (P1PU).....	129
6.5.2.10	Port P1 pull-down enable configuration register (P1PD).....	130
	Port P1 Open-Drain Output Configuration Register (P1OD).....	131



6.5.2.11 Port P1 High Interrupt Enable Configuration Register (P1HIE).....	132
6.5.2.12 Port P1 Low Level Interrupt Enable Configuration Register (P1LIE).....	
133 6.5.2.13 Port P1 Rising Edge Interrupt Enable Configuration Register (P1RIE) .....	134
6.5.2.14 Port P1 Falling Edge Interrupt Enable Configuration Register (P1FIE).....	
135 6.5 .2.15 Port P1 Interrupt Status Register (P1_STAT) .....	136
6.5.2.16 Port P1 Interrupt Clear Register (P1_ICLR).....	137 6.5.3
Port P2 .....	138 6.5.3.1 Port P23
function configuration register (P23_SEL).....	138 6.5.3.2 Port P24
function configuration register (P24_SEL). ....	139 6.5.3.3 Port
P25 function configuration register (P25_SEL).....	140 6.5.3.4 Port P26
function configuration register (P26_SEL).....	141 6.5. 3.5 Port P27
function configuration register (P27_SEL).....	142 6.5.3.6 Port P2 I/
O Configuration Register (P2DIR).ý... .....	143 6.5.3.7 Port P2 Input Value
Register (P2IN) .....	144 6.5.3.8 Port P2 output value
configuration register (P2OUT) .....	6.5.3.9 Port P2 digital- 145
analog configuration register (P2ADS).....	146 6.5.3.10 Port P2 Drive
Capability Configuration Register (P2DR).....	147 6.5.3.11
Port P2 Pull-Up Enable Configuration Register (P2PU).....	148 6.5.3.12
Port P2 Pull-down Enable Configuration Register (P2PD) .....	
149 6.5.3.13 Port P2 Open-Drain Output Configuration Register (P2OD).. .....	
150 6.5.3.14 Port P2 High Interrupt Enable Configuration Register (P2HIE).....	
151 6.5.3.15 Port P2 Low Level Interrupt Enable Configuration Register (P2LIE).....	
152 6.5.3.16 Port P2 Rising Edge Interrupt Enable Configuration Register (P2RIE) .....	153
6.5.3.17 Port P2 down Edge Interrupt Enable Configuration Register (P2FIE).....	154 6.5.3.18 Port P2 Interrupt Status Register
(P2_STAT) .....	6.5.3.19 Port P2 Interrupt Clear Register
(P2_ICLR).....	156 6.5.4 Port
P3.....	157 6.5.4.1 Port
P31 function configuration register (P31_SEL).....	157 6.5.4.2 Port
P32 function configuration register (P32_SEL).....	158 6.5.4.3 Port P33
function configuration register (P33_SEL).....	159 6.5.4.4 Port P34
function configuration register (P34_SEL).....	160 6.5.4.5 Port
P35 function configuration register (P35_SEL).....	161 6.5.4.6 Port P36
function configuration register ( P36_SEL).....	162 6.5.4.7 Port P3 I/O
Configuration Register (P3DIR).....	163 6.5.4.8 Port P3 Input Value
Register (P3IN) .....	165 6.5.4.9 Port P3 output value
configuration register (P3OUT) .....	167 6.5.4.10 Ports P3 Digital-to-
Analog Configuration Register (P3ADS).....	169 6.5.4.11 Port P3 drive
capability configuration register (P3 DR) .....	171 6.5.4.12
Port P3 Pull-Up Enable Configuration Register (P3PU).....	173
6.5.4.13 Port P3 Pull-down Enable Configuration Register (P3PD).....	
175 6.5.4.14 Port P3 Open-Drain Output Configuration Register (P3OD).....	177 6.5.4.15 Port



6.5.4.16 Port P3 Low Level Interrupt Enable Configuration Register (P3LIE).....	181
6.5.4.17 Port P3 Rising Edge Interrupt Enable Configuration Register (P3RIE) .....	189
Port P3 Falling Edge Interrupt Enable Configuration Register (P3FIE).....	185
6.5.4.19 Port P3 Interrupt Status Register (P3_STAT) .....	187
6.5.4.20 Port P3 Interrupt Clear Register (P3_ICLR).....	189
6.5.5       Port Accessibility .....	191
6.5.5.1 Port Auxiliary Function Configuration Register 1 (GPIO_CTRL1).....	191
6.5.5.2 Port Auxiliary Function Configuration Register 2 (GPIO_CTRL2).....	193
6.5.5.3 Port Auxiliary Function Configuration Register 3 (GPIO_CTRL3).....	195
6.5.5.4 Port Auxiliary Function Configuration Register 4 (GPIO_CTRL4).....	197
7 FLASH Controller (FLASH).....	199
7.1       Overview .....	199
7.2       Structure diagram.....	199
7.3       Functional Description.....	200
7.3.1       Page Erase (Sector Erase).....	200
7.3.2       Chip Erase .....	Write operation
7.3.3       (Program) .....	201
7.3.4       Read operation .....	203
7.4       Erase Timing .....	204
7.5       Read Wait Cycle .....	206
7.6       Erase and write protection.....	206
7.6.1       Erase and write protection bits .....	206
7.6.2       PC address erasure protection.....	206
7.7       Register Write Protection .....	207
7.8       register.....	208
7.8.1       TNVS parameter register (FLASH_TNVS).....	208
7.8.2       TPGS parameter register (FLASH_TPGS).....	209
7.8.3       TPROG parameter register (FLASH_TPROG) .....	209
7.8.4       TSEERASE register (FLASH_TSEERASE).....	210
7.8.5       TMERASE parameter register (FLASH_TMERASE) .....	210
7.8.6       TPRCV parameter register (FLASH_TPRCV).....	211
7.8.7       TSRCV parameter register (FLASH_TSRCV).....	211
7.8.8       TMRCV parameter register (FLASH_TMRCV).....	212
7.8.9       CR register (FLASH_CR).....	212
7.8.10       IFR register (FLASH_IFR).....	213
7.8.11       ICLR register (FLASH_ICLR) .....	213
7.8.12       BYPASS register (FLASH_BYPASS).....	214
7.8.13       SLOCK register (FLASH_SLOCK) .....	215
8 RAM Controller (RAM).....	216
8.1       Overview .....	216
8.2       Functional Description.....	216
8.3       register.....	217



8.3.1	Control Register (RAM_CR) .....	217
8.3.2	Parity Error Address Register (RAM_ERRADDR).....	218
8.3.3	Error Interrupt Flag Register (RAM_IFR) .....	218
8.3.4	Error Interrupt Flag Clear Register (RAM_ICLR).....	219
9 Basic timer (TIM0/1/2) .....		220
9.1	Introduction to Basic Timers.....	220
9.2	Base Timer function description.....	221
9.2.1	Counting function.....	223
9.2.2	Timing function.....	223
9.2.3	Buzzer features.....	224
9.3	Base Timer Interconnect .....	225
9.3.1	GATE interconnection.....	225
9.3.2	Toggle Output Interconnect .....	225
9.4	Base Timer Register Description.....	226
9.4.1	16-bit Mode Reload Register (TIMx_ARR).....	226
9.4.2	16-bit Mode Count Register (TIMx_CNT).....	227
9.4.3	32-bit Mode Count Register (TIMx_CNT32).....	227
9.4.4	Control Register (TIMx_CR) .....	228
9.4.5	Interrupt Flag Register (TIMx_IFR).....	229
9.4.6	Interrupt Flag Clear Register (TIMx_ICLR).....	229
10 Low Power Timer (LPTIM).....		230
10.1	Introduction to LPTimer.....	230
10.2	LPTimer function description.....	231
10.2.1	Counting function.....	232
10.2.2	Timing function.....	232
10.3	LPTimer Interconnect .....	233
10.3.1	GATE Interconnect .....	233
10.3.2	EXT Interconnect .....	233
10.3.3	Toggle Output Interconnect .....	233
10.4	LPTimer register description.....	234
10.4.1	Counter count value register (LPTIM_CNT) .....	235
10.4.2	Reload Register (LPTIM_ARR).....	235
10.4.3	Control Register (LPTIM_CR).....	236
10.4.4	Interrupt Flag Register (LPTIM_IFR).....	237
10.4.5	Interrupt Flag Clear Register (LPTIM_ICLR).....	237
11 Programmable Counting Array (PCA) .....		238
11.1	Introduction to PCA.....	238
11.2	PCA function description.....	239
11.2.1	PCA Timer/Counter .....	239
11.2.2	PCA capture function.....	241
11.2.3	PCA compare function.....	243
11.2.3.1	16-bit software counter mode.....	243



11.2.3.2 High-speed output mode.....	244
11.2.3.3 WDT function of PCA module 4.....	245
11.2.3.4PCA 8-bit PWM function.....	247
11.3 Interconnection and control of PCA module and other modules.....	249
11.3.1 ECI Interconnect .....	249
11.3.2 PCACAP0 .....	249
11.3.3 PCACAP1 .....	249
11.3.4 PCACAP[4:2].....	249
11.4 PCA register description.....	250
11.4.1 Control Register (PCA_CCON).....	251
11.4.2 Mode Register (PCA_CMOD) .....	252
11.4.3 Count register (PCA_CNT) .....	253
(PCA_ICLR) .....	11.4.4 Interrupt Clear Register
(PCA_CCAPM0~4) .....	253
11.4.5 Compare capture mode registers .....	254
11.4.6 The upper 8 bits of the compare capture data register (PCA_CCAP0~4H).....	255
11.4.7 Compare capture data register lower 8 bits (PCA_CCAP0~4L) .....	11.4.8 Compare
capture 16-bit registers (PCA_CCAP0~4) .....	255
11.4.9 Compare High Speed Output Flag Register (PCA_CCAPO).....	256
12 Advanced Timer (TIM4/5/6) .....	257
12.1 Introduction to Advanced Timers.....	257
12.2 Advanced Timer function description.....	259
12.2.1 Basic actions.....	259
12.2.1.1 Basic Waveform Mode.....	259
12.2.1.2 Compare output.....	260
12.2.1.3 Capture Input .....	261
12.2.2 Clock Source Selection .....	262
12.2.3 Counting direction .....	263
12.2.3.1 Ramp counting direction.....	263
12.2.3.2 Triangle wave count direction.....	263
12.2.4 Digital Filtering .....	264
12.2.5 Software Synchronization .....	265
12.2.5.1 Software synchronization startup.....	265
12.2.5.2 Software Synchronization Stop .....	265
12.2.5.3 Software Synchronous Clear .....	265
12.2.6 Hardware Synchronization .....	267
12.2.6.1 Hardware synchronization startup.....	267
12.2.6.2 Hardware Synchronization Stop .....	267
12.2.6.3 Hardware synchronous clearing.....	267
12.2.6.4 Hardware Sync Capture Input.....	267
12.2.6.5 Hardware Synchronization Count.....	267
12.2.7 Cache function.....	269
12.2.7.1 Buffer transfer time point.....	270

12.2.7.2 Generic cycle reference value buffer transfer time point.....	270
12.2.7.3 Generic Baseline Value Cache Delivery time point .....	270
12.2.7.4 Capture input value buffer transfer time point.....	270
12.2.7.5 Buffer transfer when clearing action.....	
270 12.2.8 General purpose PWM output. ....	271
12.2.8.1 PWM Spread Spectrum Output .....	
271 12.2.8.2 Independent PWM output .....	
271 12.2.8.3 Complementary PWM output.....	
271 12.2.8.3.1 Software set GCMBR complement PWM output.....	272
12.2.8.3.2 GCMBR Complementary PWM Output by Hardware .....	
273 12.2.8.4 Multiphase PWM Output .....	
12.2.9       274 Quadrature Code Count .....	
276 12.2.9.1 Position count mode .....	
276 12.2.9.1.1 Basic count.....	276
12.2.9.1.2 Phase Difference Count.....	277
12.2.9.1.3 Direction Count .....	
278 12.2.9.2 Revolution model.....	
278 12.2.9.2.1 Z-phase count .....	
278 12.2.9.2.2 Position Overflow Count .....	
279 12.2.9.2.3 Mixed counts.....	279
12.2.9.2.4 Z-phase action shield .....	280
12.2.10 Periodic Interval Response .....	
12.2.11       282 Protection Mechanisms .....	
12.2.12       283 Interrupt Description .....	
284 12.2.12.1 Counting Compare Match Interrupt .....	
284 12.2.12.2 Count Period Match Interrupt.....	284
12.2.12.3 Dead-time error interrupt .....	284
12.2.13       Brake Protection .....	285
12.2.13.1 Port Brake and Software Brake .....	285
12.2.13.2 Auto Brake in Low Power Mode .....	285
12.2.13.3 The output level is the same as the high and the low brake.....	
286 12.2.13.4VC Brakes .....	286
12.2.14       Internal interconnection.....	
287 12.2.14.1 Interrupt trigger output .....	
287 12.2.14.2AOS trigger .....	287
Port Trigger TRIGA-TRIGD.....	288
Interconnecting the compare output VC with the Advanced Timer .....	
289 12.2.14.5 UART and Advanced Timer Interconnect .....	
12.3       289 Register Descriptions .....	290
12.3.1 General purpose count reference register (TIMx_CNTER) .....	
12.3.2       292 Universal Period Reference Register (TIMx_PERAR).....	
292 12.3.3 General Periodic Buffer Register (TIMx_PERBR) .....	293



12.3.4 General Purpose Compare Reference Registers (TIMx_GCMAR-GCMDR) .....	293
12.3.5 Dead Time Reference Register (TIMx_DTUAR-DTDAR).....	294
12.3.6 General Purpose Control Register (TIMx_GCONR).....	295
12.3.7 Interrupt Control Register (TIMx_ICONR) .....	297
12.3.8 Port Control Register (TIMx_PCONR).....	298
12.3.9 Buffer Control Register (TIMx_BCONR).....	301
12.3.10 Dead Time Control Register (TIMx_DCONR).....	302
12.3.11 Filter Control Register (TIMx_FCONR).....	303
12.3.12 Valid Period Register (TIMx_VPERR).....	305
12.3.13 Status Flag Register (TIMx_STFLR).....	306
12.3.14 Hardware Start Event Select Register (TIMx_HSTAR).....	308
12.3.15 Hardware Stop Event Select Register (TIMx_HSTPR) .....	12.3.16 Hardware Clear 310
Event Select Register (TIMx_HCELR) .....	312
12.3.17 Hardware Capture A Event Select Register (TIMx_HCPAR).....	314
12.3.18 Hardware Capture B Event Select Register (TIMx_HCPBR).....	316
12.3.19 Hardware Increment Event Select Register (TIMx_HCUPR).....	318
12.3.20 Hardware Decrement Event Select Register (TIMx_HCDOR).....	320
12.3.21 Software Synchronization Start Register (TIMx_SSTAR) .....	322
12.3.22 Software Synchronization Stop Register (TIMx_SSTPR).....	323
12.3.23 Software Synchronous Clear Register (TIMx_SCLRR) .....	12.3.24 324
Interrupt Flag Register (TIMx_IFR).....	325
12.3.25 Interrupt Flag Clear Register (TIMx_ICLR).....	327
12.3.26 Spread spectrum and interrupt trigger selection (TIMx_CR).....	328
12.3.27 AOS selection control register (TIMx_AOSSR).....	329
12.3.28 AOS Select Control Register Flag Clear (TIMx_AOSCL).....	330
12.3.29 Port Brake Control Register (TIMx_PTAKS) .....	12.3.30 Port Trigger 331
Control Register (TIMx_TTRIG).....	332
12.3.31 AOS Trigger Control Register (TIMx_ITRIG) .....	12.3.32 Port Brake 333
Polarity Control Register (TIMx_PTAKP) .....	13 Real Time Clock 334
(RTC) .....	335
13.1 Introduction to Real Time Clocks.....	335
13.2 Real-time clock function description.....	336
13.2.1 Power-on settings.....	336
13.2.2 RTC count start setting.....	336
13.2.3 System Low Power Mode Switching .....	336
13.2.4 Reading the count register.....	337
13.2.5 Write Count Register .....	337
13.2.6 Alarm setting .....	338
13.2.7 1Hz output.....	338
13.2.8 Clock Error Compensation .....	339
13.3 RTC Interrupt.....	341
13.3.1 RTC Alarm Interrupt.....	341



13.3.2 RTC Cycle Interrupt.....	341
13.4 RTC register description.....	342
13.4.1 Control Register 0 (RTC_CR0) .....	13.4.2 Control Register 1 343
(RTC_CR1) .....	345
13.4.3 Second Count Register (RTC_SEC) .....	.347
13.4.4 Minute Count Register (RTC_MIN) .....	347
13.4.5 Hour count register (RTC_HOUR) .....	13.4.6 Day Count Register 348
(RTC_DAY) .....	350
13.4.7 Week Count Register (RTC_WEEK).....	351
13.4.8 Month Count Register (RTC_MON) .....	352
13.4.9 Year Count Register (RTC_YEAR).....	352
13.4.10 Minute Alarm Register (RTC_ALMMIN).....	353
13.4.11 Time Alarm Register (RTC_ALMHOUR) .....	13.4.12 Weekly Alarm 353
Register (RTC_ALM WEEK)....	354
13.4.13 Clock Error Compensation Register (RTC_COMPEN) .....	355
14 Watchdog Timer (WDT).....	357
14.1 Introduction to WDT.....	357
14.2 WDT function description.....	358
14.2.1 Interrupt after WDT overflow.....	358
14.2.2 Reset after WDT overflow.....	358
14.3 WDT register description.....	359
14.3.1 WDT Clear Control Register (WDT_RST).....	359
14.3.2 WDT_CON register.....	15 Universal Synchronous 360
Asynchronous Receiver/Transmitter (UART).....	361
15.1 Overview .....	361
15.2 Structure diagram.....	361
15.3 Main Features .....	362
15.4 Functional Description.....	363
15.4.1 Operating mode.....	363
15.4.1.1 Mode0~Mode3 function comparison.....	363
15.4.1.2 Mode0 (Sync Mode, Half Duplex) .....	363
15.4.1.3 Mode1 (asynchronous mode, full duplex) .....	365
15.4.1.4 Mode2 (asynchronous mode, full duplex) .....	365
15.4.1.5 Mode3 (asynchronous mode, full duplex) .....	366
15.4.2 Baud Rate Generation .....	368
15.4.2.1 Mode1/Mode3 baud rate setting example.....	369
15.5 Framing Error Detection.....	373
15.6 Multi-machine communication.....	374
15.7 Automatic Address Recognition.....	375
15.7.1 Given address.....	375
15.7.1.1 Broadcast address.....	375
15.7.1.2 Examples .....	375



15.8	Transceiver Buffer .....	376
15.8.1	Receive buffer.....	376
15.8.2	Send Cache .....	376
15.9	register.....	377
15.9.1	Data Register (UARTx_SBUF) .....	377
15.9.2	Control Register (UARTx_SCON) .....	378
15.9.3	Address Register (UARTx_SADDR).....	379
15.9.4	Address Mask Register (UARTx_SADEN) .....	379
	15.9.5 Flags Register (UARTx_ISR) .....	380
15.9.6	Flag Clear Register (UARTx_ICR).....	381
16	Low-Power Synchronous Asynchronous Receiver/Transmitter (LPUART) .....	382
16.1	Overview .....	382
16.2	Structure diagram.....	383
16.3	Main Features .....	384
16.4	Functional Description.....	385
16.4.1	Configure Clock and Transmit Clock .....	385
16.4.2	Operating mode.....	385
	16.4.2.1 Mode0~Mode3 function comparison.....	386
	16.4.2.2Mode0 (synchronous mode, half-duplex) data sending and receiving instructions.....	387
	16.4.2.3Mode1 (asynchronous mode, full duplex) data sending and receiving instructions.....	388
	16.4.2.4Mode2 (asynchronous mode, full duplex) data sending and receiving instructions.....	389
	16.4.2.5Mode3 (asynchronous mode, full duplex) data sending and receiving instructions.....	390
16.4.3	Baud Rate Generation .....	391
16.5	Framing Error Detection.....	391
16.6	Multi-machine communication.....	392
16.7	Automatic Address Recognition.....	393
16.7.1	Given address.....	393
16.7.2	Broadcast address.....	393
16.7.3	Examples .....	393
16.8	Transceiver Buffer .....	394
16.8.1	Receive buffer.....	394
16.8.2	Send Cache .....	394
16.9	register.....	395
16.9.1	Data Register (LPUART_SBUF) .....	395
16.9.2	Control Register (LPUART_SCON).....	396
	16.9.3 Address Register (LPUART_SADDR) .....	398
	16.9.4 Address Mask Register (LPUART_SADEN).....	398
	16.9.5 Interrupt Flag Register (LPUART_ISR).....	399
	16.9.6 Interrupt Flag Clear Register (LPUART_ICR) .....	400
17	I2C bus (I2C).....	401
17.1	Introduction .....	401
17.2	Main Features .....	401



17.3	Protocol Description.....	401
17.3.1	Data transfer on the I2C bus.....	402
17.3.2	Acknowledgments on the I2C bus.....	403
17.3.3	Arbitration on the I2C bus.....	404
17.4	Functional Description.....	406
17.4.1	Serial Clock Generator .....	407
17.4.2	Input filter.....	407
17.4.3	Address Comparator.....	407
17.4.4	Acknowledgment flag bit .....	408
17.4.5	Interrupt Generator .....	408
17.4.6	Operating mode.....	408
17.4.7	Status code representation.....	415
17.5	Programming Examples .....	417
17.5.1	Host Send Example .....	417
17.5.2	Host Receive Example .....	418
17.5.3	Slave Receive Example .....	419
17.5.4	Slave Send Example .....	420
17.6	Register Descriptions.....	421
17.6.1	I2C Baud Rate Counter Enable Register (I2C_TMRUN) .....	421
17.6.2	I2C Baud Rate Counter Configuration Register (I2C_TM).....	422
17.6.3	I2C Configuration Register (I2C_CR).....	423
17.6.4	I2C Data Register (I2C_DATA) .....	425
17.6.5	I2C Address Register (I2C_ADDR).....	426
17.6.6	I2C Status Register (I2C_STAT) .....	427
18	Serial Peripheral Interface (SPI) .....	428
18.1	Introduction to SPI.....	428
18.2	SPI main features.....	428
18.3	SPI function description.....	429
18.3.1	SPI Master Mode.....	429
18.3.2	SPI Slave Mode.....	429
18.3.3	SPI data frame format.....	431
18.3.4	SPI Status Flags and Interrupts.....	432
18.3.5	SPI Multi-machine System Configuration Instructions.....	432
18.3.6	SPI pin configuration description.....	434
18.4	SPI programming example.....	435
18.4.1	SPI Master Transmit Example.....	435
18.4.2	SPI Master Receive Example.....	435
18.4.3	SPI Slave Transmit Example .....	436
18.4.4	SPI Slave Receive Example .....	436
18.6	SPI Register Descriptions.....	438
18.6.1	SPI Configuration Register (SPI_CR).....	439
18.6.2	SPI Chip Select Configuration Register (SPI_SSN).....	441



18.6.3	SPI Status Register (SPI_STAT).....	442
18.6.4	SPI Data Register (SPI_DATA).....	443
19	Clock Trim Module (CLKTRIM).....	444
19.1	Introduction to CLK_TRIM.....	444
19.2	CLK_TRIM main features.....	444
19.3	CLK_TRIM function description.....	445
19.3.1	CLK_TRIM calibration mode.....	445
19.3.1.1	Operational Flow .....	445
19.3.2	CLK_TRIM monitor mode.....	446
19.3.2.1	Operational Flow .....	446
19.4	CLK_TRIM register description.....	447
19.4.1	Configuration Register (CLKTRIM_CR) .....	448
19.4.2	Reference Counter Initial Value Configuration Register (CLKTRIM_REFCON).....	449
19.4.3	Reference Counter Value Register (CLKTRIM_REFCNT) .....	449
19.4.4	Calibration Counter Value Register (CLKTRIM_CALCNT).....	450
19.4.5	Interrupt Flag Register (CLKTRIM_IFR) .....	451
	(CLKTRIM_ICLR ) .....	452
19.4.7	Calibration Counter Overflow Value Configuration Register (CLKTRIM_CALCON) .....	20 Cyclic 453
Redundancy Check (CRC)	.....	.454
20.1	Overview .....	.454
20.2	Main Features .....	.454
20.3	Functional Description.....	.454
20.3.1	Operating mode.....	.454
20.3.2	Encoding.....	.454
20.3.3	Write Bit Width .....	.454
20.4	Programming Examples .....	.455
20.4.1	CRC-16 Encoding Mode.....	.455
20.4.2	CRC-16 Check Mode .....	.455
20.5	Register Descriptions.....	.456
20.5.1	Register List .....	.456
20.5.2	Result Register (CRC_RESULT) .....	.457
20.5.3	Data Register (CRC_DATA).....	.457
21	Analog-to-Digital Converter (ADC) .....	.458
21.1	Module Introduction.....	.458
21.2	ADC Block Diagram .....	.458
21.3	Conversion Timing and Conversion Speed .....	.459
21.4	Single Conversion Mode .....	.460
21.5	Continuous Conversion Mode .....	.462
21.6	Continuous Conversion Accumulation Mode .....	.464
21.7	Comparison of ADC conversion results.....	.466
21.8	ADC Interrupts.....	.467
21.9	Measuring the ambient temperature with a temperature sensor.....	.467



21.10 ADC Module Registers.....	469
21.10.1 ADC Configuration Register 0 (ADC_CR0).....	470
21.10.2 ADC Configuration Register 1 (ADC_CR1).....	472
21.10.3 ADC Configuration Register 2 (ADC_CR2).....	475
21.10.4 ADC channel 0 conversion result (ADC_result0) .....	477
21.10.5 ADC channel 1 conversion result (ADC_result1) .....	477
21.10.6 ADC channel 2 conversion result (ADC_result2) .....	478
21.10.7 ADC channel 3 conversion result (ADC_result3) .....	478
21.10.8 ADC channel 4 conversion result (ADC_result4) .....	479
21.10.9 ADC channel 5 conversion result (ADC_result5) .....	479
21.10.10 ADC channel 6 conversion result (ADC_result6) .....	480
21.10.11 ADC channel 7 conversion result (ADC_result7) .....	480
21.10.12 ADC channel 8 conversion result (ADC_result8) .....	481
21.10.13 ADC conversion result accumulation value (ADC_result_acc).....	481
21.10.14 ADC Compare Upper Threshold (ADC_HT).....	482
21.10.15 ADC Compare Lower Threshold (ADC_LT).....	482
21.10.16 ADC Interrupt Flag Register (ADC_IFR).....	483
21.10.17 ADC Interrupt Clear Register (ADC_ICLR) .....	484
21.10.18 ADC result (ADC_result) .....	22 Analog Comparator 484
(VC) .....	485
22.1 Introduction to Analog Voltage Comparator VC.....	485
22.2     Voltage Comparator Block Diagram .....	486
22.3 Setup/Response Time .....	486
22.4     Filter time .....	487
22.5     Hysteresis function.....	487
22.6 VC register.....	488
22.6.1 VC Configuration Register (VC_CR).....	489
22.6.2 VC0 Configuration Register (VC0_CR).....	491
22.6.3 VC1 Configuration Register (VC1_CR).....	493
22.6.4 VC0 Output Configuration Register (VC0_OUT_CFG).....	495
22.6.5 VC1 Output Configuration Register (VC1_OUT_CFG).....	497
22.6.6 VC Interrupt Register (VC_IFR).....	499
23 Low Voltage Detector (LVD).....	500
23.1 Introduction to LVD.....	500
23.2 LVD Block Diagram .....	500
23.3     Digital Filtering .....	501
23.4     Hysteresis function.....	501
23.5     Configuration example.....	502
23.5.1 LVD Configured for Low Voltage Reset.....	502
23.5.2 LVD Configured as Interrupt on Voltage Change.....	502
23.6 LVD register.....	503
23.6.1 LVD Configuration Register (LVD_CR).....	503



23.6.2 LVD Interrupt Register (LVD_IFR) .....	505
24 Simulate other registers.....	.506
24.1 BGR Configuration Register (BGR_CR).....	.506
25 SWD debug interface.....	.507
25.1 SWD commissioning additional functions.....	.507
25.2 ARM® Reference Documentation.....	.508
25.3      Debug Port Pins .....	.509
25.3.1 SWD port pins.....	.509
25.3.2 SW-DP pin assignment.....	.509
25.3.3 Internal pull-up on the SWD pin.....	.509
25.4 SWD port.....	.510
25.4.1 Introduction to SWD protocol.....	.510
25.4.2 SWD protocol sequence .....	.510
25.4.3 SW-DP state machine (reset, idle state, ID code).....	.511
25.4.4 DP and AP read/write access .....	.511
25.4.5 SW-DP register.....	.512
25.4.6 SW-AP register.....	.513
25.5      Kernel debugging.....	.514
25.6      BPU (Breakpoint Unit) .....	.514
25.6.1 BPU function.....	.514
25.7      DWT (Data Observation Point) .....	.515
25.7.1 DWT function.....	.515
25.7.2 DWT Program Counter Sampling Register.....	.515
25.8 MCU Debug Component (DBG) .....	.516
25.8.1      Debug support for low-power modes.....	.516
25.8.2      Debug support for timers, watchdogs.....	.516
25.9      Debug mode module working state control (DEBUG_ACTIVE).....	.517
26 Device Electronic Signature .....	.519
26.1 Product Unique Identification (UID) Register (80-bit).....	.519
26.2      Product Model Register .....	.520
26.3 FLASH size register.....	.520
26.4 RAM Size Register .....	.521
26.5      Pin Count Register .....	.521
27 Appendix A SysTick Timer.....	.522
27.1      Introduction to SysTick timers.....	.522
27.2 Setting SysTick.....	.522
27.3 SysTick Register.....	.523
27.3.1      SysTick Control and Status Register (CTRL) .....	.523
27.3.2      SysTick Reload Register (LOAD) .....	.523
27.3.3      SysTick Current Value Register (VAL) .....	.524
28 APPENDIX B DOCUMENT CONVENTIONS .....	.525
28.1      List of Register-Related Abbreviations .....	.525



---

28.2      Glossary.....	525
Version History & Contact .....	526



## table directory

Table 2-1 Operational module diagram in run mode.....	
30 Table 2-2 Operational block diagrams in sleep mode .....	
32 Table 2-3 Operational Module Diagram in Deep Sleep Mode .....	
34 Table 5-1 Cortex-M0+ processor exception list .....	68
Table 5-2 Correspondence between external interrupt and NVIC interrupt input.....	74 Table 6-1 Port Status Truth
Table .....	94 Table 6-2 Port
Multiplexing Table .....	97 Table
7-1 Different frequencies Next FLASH erasing time parameters.....	
204 Table 9-1 Base Timer Register List .....	226 Table
10-1 LPTimer register list. ....	234 Table 11-1
PCA Compare Capture Function Block Settings .....	234 .....
248 Table 11-2 PCA Register List .....	250 Table
12-1 Basic Features of Advanced Timer .....	257
Table 12-2 Advanced Timer Port List .....	257 Table 12-3 AOS
source selection.....	288 Table 12-4 Port Trigger
Selection .....	288 Table 12-5 Advanced Timer
Register List.....	291 Table 13-1 Basic characteristics of
RTC .....	335 Table 13- 2 RTC register
list.....	342 Table 14-1 WDT register
list.....	359 Table 15-1 Mode0/1/2/3 Data
Structure .....	363 Table 16-1 Mode0/ 1/2/3
Data Structures .....	386 Table 17-1 I2C Clock Signal
Baud Rate .....	407 Table 17-2 I2C Status Code
Description .....	416 Table 17-3 Register
List .....	421 Table 18-1 SPI pin
configuration description table.....	434 Table 18-2 SPI Register
List... .....	438 Table 18-3 Host Mode Baud Rate
Selection .....	440 Table 19-1 Register
List.. .....	447 Table 21-1 ADC
Registers.....	469 Table 22-1 VC
Registers .....	488 Table 23-1 LVD Register .....



## Figure Catalog

Figure 1-1 Schematic diagram of system architecture.....	.....
24 Figure 1-2 Address Area Partitioning Diagram .....	..... 26
Figure 2-1 Block Diagram of Control Mode .....	.....
28 Figure 3-1 Block Diagram of Clock Control Module .....	.....
28 .....	..... 37 Figure 3-2 Crystal oscillator clock startup
diagram .....	..... 39 Figure 3-3 Clock Switching
Diagram .....	..... 42 Figure 3-4 Schematic diagram
of clock calibration.....	..... 45 Figures 4-1
Schematic diagram of reset source.....	..... 62 Figure
5-1 used only High-order two bits of the priority register.....	..... 68 Figure
5-2 Interrupt Vector Table .....	..... 69 Figure 5-3
Interrupt Active and Pending States .....	..... 70 Figure
5-4 Interrupt pending state is cleared and then re-acknowledged.....	..... 71
Figure 5-5 If the interrupt request is held high when the interrupt exits, it will cause the interrupt processing to be executed again.....	..... 71 Figure 5 -6 Interrupt pending during interrupt processing can also be acknowledged.....
.....	..... 72 Figure 5-7 Interrupt
Structure .....	..... 75 Figure
6-1 Port Circuit Diagram .....	.....
95 Figure 6-2 AHB bus port variation with system clock .....	..... 95
Figure 6-3 Read Port Boot Foot Data Synchronization Diagram .....	.....
96 Figure 7-1 Memory Sector Division .....	..... 199
Figure 9-1 Base Timer block diagram .....	..... 220 Figure 9-2
Block Diagram of Timer Mode 1.....	..... 222 Figure 9-3
Block Diagram of Timer Mode 2 .....	..... 222 Figures 9-4 Mode 1
Timing Chart .....	..... 223 Figure 9-5 Mode 2 Timing
Diagram (prescaler set to 2).....	..... 223 Figure 10-1 LPTimer Block
Diagram .....	..... 230 Figure 10-2 LPTimer Mode
1 .....	..... 231 Figure 10-3 LPTimer Mode
2.....	..... 232 Figure 11-1 Overall block diagram of
PCA.....	..... 238 Figure 11-2 PCA Counter Block
Diagram .....	..... 240 Figure 11-3 PCA Capture
Functional Block Diagram.....	..... 242 Figure 11-4 PCA Compare Function
Block Diagram .....	..... 244 Figure 11-5 PCA WDT functional
block diagram .....	..... 245 Figure 11-6 PCA PWM
functional block diagram .....	..... 247 Figure 11-7 PCA PWM output
waveform.....	..... 248 Figure 12-1 Advanced Timer
Block Diagram .....	..... 258 Figure 12-2 Ramp
waveform (count up) .....	..... 259 Figure 12-3 Triangle
Waveform .....	..... 259 Figure 12-4
Comparison output action.....	..... 260 Figure 12-5 Capture Input Action



Figure 12-7 Software synchronization action.....	
265 Figure 12-8 Hardware synchronization action.....	
268 Figure 12-9 Compare output timing in single buffer mode.. .....	
269 Figure 12-10 Schematic diagram of PWM spread spectrum output.....	
271 Figure 12-11 CHA output PWM Wave .....	271
Figure 12-12 Software setting of GCMBR complementary PWM wave output in triangle wave A mode..... .....	272
Figure 12-13 GCMBR Complementary PWM Wave Output (Symmetrical Dead Zone) by Hardware Setting in Triangle Wave B Mode .....	
273 Figure 12-14 6-Phase PWM Wave.....	
274 Figure 12-15 Three-phase complementary PWM wave output with dead time in triangular wave A mode..... .....	
275 Figure 12-16 Basic counting action in position mode.....	
276 Fig. 12-17 Phase difference count action setting in position mode (1 times)..... .....	
277 Figure 12-18 Phase difference counting action in position mode Setting (2x) .....	
277 Figure 12-19 Phase difference count action setting in position mode (4 times)..... .....	
277 Fig. 12-20 Direction counting action in position mode..... .....	
278 Figure 12-21 Z-phase counting action in revolution mode..... .....	
278 Fig. 12-22 Position counter output counting action in revolution mode..... .....	
279 Figure 12-23 Z-phase count and position counter output mixed count action in revolution mode..... .....	
279 Fig. 12-24 Revolution count mode - mixed count Z-phase shielding action example 1 .....	
280 Fig. 12-25 Revolution count mode - mixed count Z-phase mask action example 2 .....	
281 Figure 12-26 Periodic interval valid request signal action .....	
282 Figure 12-27 Port brake and software brake diagram..... .....	285
Fig. 12-28 Schematic diagram of output same high and low braking.....	
286 Fig. 12-29 VC brake control diagram..... .....	286 Figure
12-30 Timer4/5/6 Interrupt Selection .....	287 Figure
13-1 RTC block diagram .....	335
Figure 14-1 Overall block diagram of the WDT .....	357
Figure 15-1 Block diagram of UART structure..... .....	361 Figure
15-2 Mode0 Transmit data..... .....	364 Figure 15-3 Mode0
Receive Data .....	364 Figure 15-4 Mode1
send data .....	365 Figure 15-5 Mode1
Receive Data .....	365 Figure 15-6
Mode2 Sending Data .....	366 Figure 15-7
Mode2 receive data .....	366 Figure 16-1 LPUART block diagram..... .....
diagram..... .....	383 Figure 16-2 Mode0
Sending Data .....	387 Figure 16-3 Mode0
Receive Data .....	387 Figure 16-4 Mode1
Sending Data .....	388 Figure 16-5 Mode1
Receive data..... .....	388 Figure 16-6 Mode2 Sending Data .....
..... .....	389 Figure 16-7 Mode2
Receive Data .....	389 Figure 17-1 I2C transfer protocol .....
..... .....	402 Figure 17-2 START and STOP conditions



Figure 17-3 I2C bus upper transmission.....	403 Figure 17-4
Acknowledge signal on the I2C bus.....	404 Figure 17-5 Arbitration
on the I2C bus.....	405 Figure 17-6 I2C function block
diagram .....	406 Figure 17-7 Data synchronization diagram
in master sending mode.....	409 Figure 17-8 I2C Master
Transmit State Diagram.....	409 Figure 17-9 Data synchronization
diagram in master receive mode .....	410 Figure
17-10 I2C host receiving state diagram.....	411 Figure
17-11 Data synchronization diagram in slave receive mode.....	412 Figure
17-12 Slave Receive State Diagram .....	412 Figure
17-13 Data synchronization diagram in slave transmit mode.....	
413 Figure 17-14 I2C Slave Transmit State Diagram.....	413 Figure
17-15 I2C General Call State Diagram.....	414 Figure 18-1
Schematic diagram of slave receiving.....	430 Figure
18-2 Slave Sending Diagram .....	430 .....
430 Figure 18-3 Host Mode Frame Format .....	430 .....
431 Figure 18-4 Data frame format when the slave CPHA is 0.....	431 .....
Figure 18-5 Data when slave CHPA is 1 Frame format.....	432 Figure
18-6 Schematic diagram of a SPI multi-master/multi-slave system.....	433 .....
Figure 21-1 ADC Block Diagram .....	
433 .....	458 Figure 21-2 ADC Conversion Timing
Diagram .....	459 Figure 21-3 ADC Continuous Conversion
Process Example .....	462 Figure 21-4 Example of
ADC Continuous Conversion Accumulation Process .....	464 Figure 22-1 VC
frame diagram.. .....	486 Figure 22-2 VC filter
response time.. .....	487 Figure 22-3 VC Hysteresis
Function .....	487 Figure 23-1 LVD Block
Diagram .....	
500 Figure 23-2 LVD Filtered Output .....	501 Figure 23-3 LVD hysteresis



## Introduction

The HC32L110 series is an ultra-low power, Low Pin Count,

MCU with wide voltage operating range. Integrated 12-bit 1Msps high precision SARADC and integrated comparator, multiplexer

Rich communication peripherals such as UART, SPI, and I2C have the characteristics of high integration, high anti-interference, high reliability and ultra-low power consumption.

The core of this product adopts Cortex-M0+ core, cooperates with mature Keil & IAR debugging and development software, supports C language and programming language, assembly instructions.

## Typical application of ultra-low power MCU

- ÿ Sensor applications, IoT applications
- ÿ Smart Transportation, Smart City, Smart Home
- ÿ Fire alarm probes, smart door locks, wireless monitoring and other smart sensor applications
- ÿ Various portable devices that are battery-powered and power-hungry

## About this manual

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

## 1 System structure

### 1.1 Overview

This product system consists of the following parts:

ÿ 1 AHB bus Master:

- Cortex-M0+

ÿ 4 AHB bus Slaves:

- FLASH memory

- SRAM memory

- AHB0, AHB to APB Bridge, contains all APB interface peripherals

- AHB1, contains all AHB interface peripherals

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

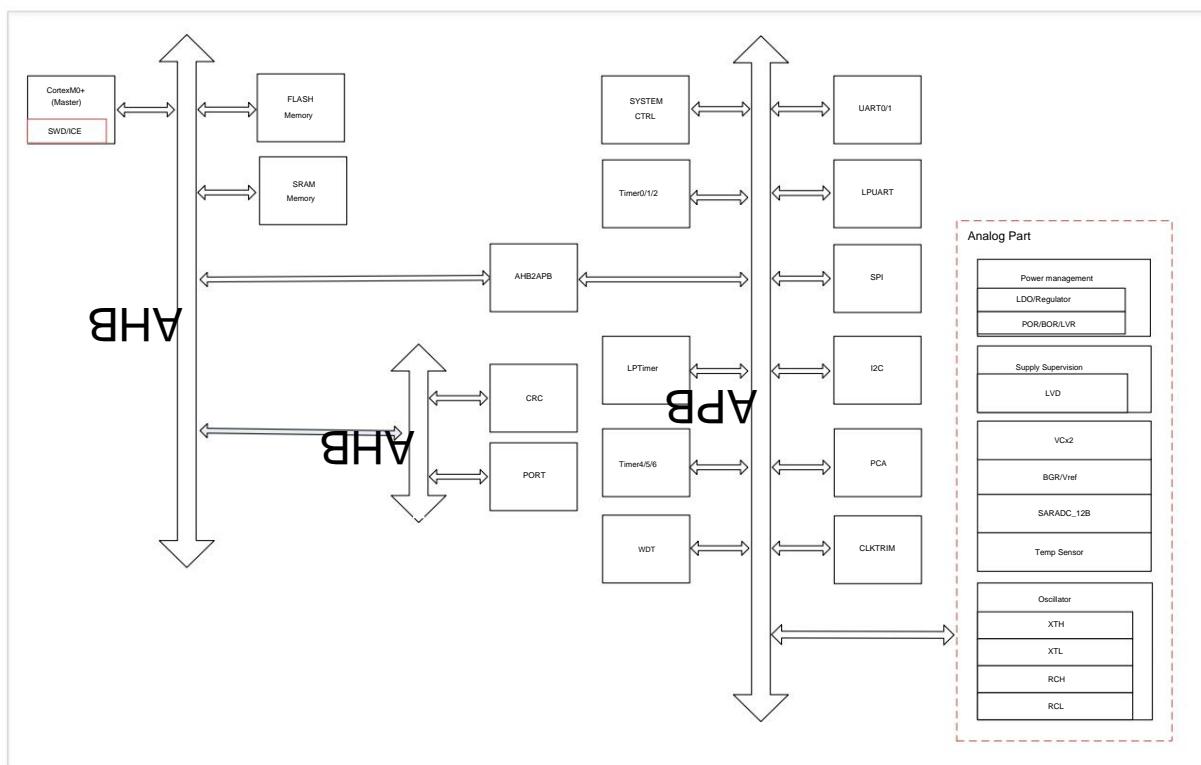
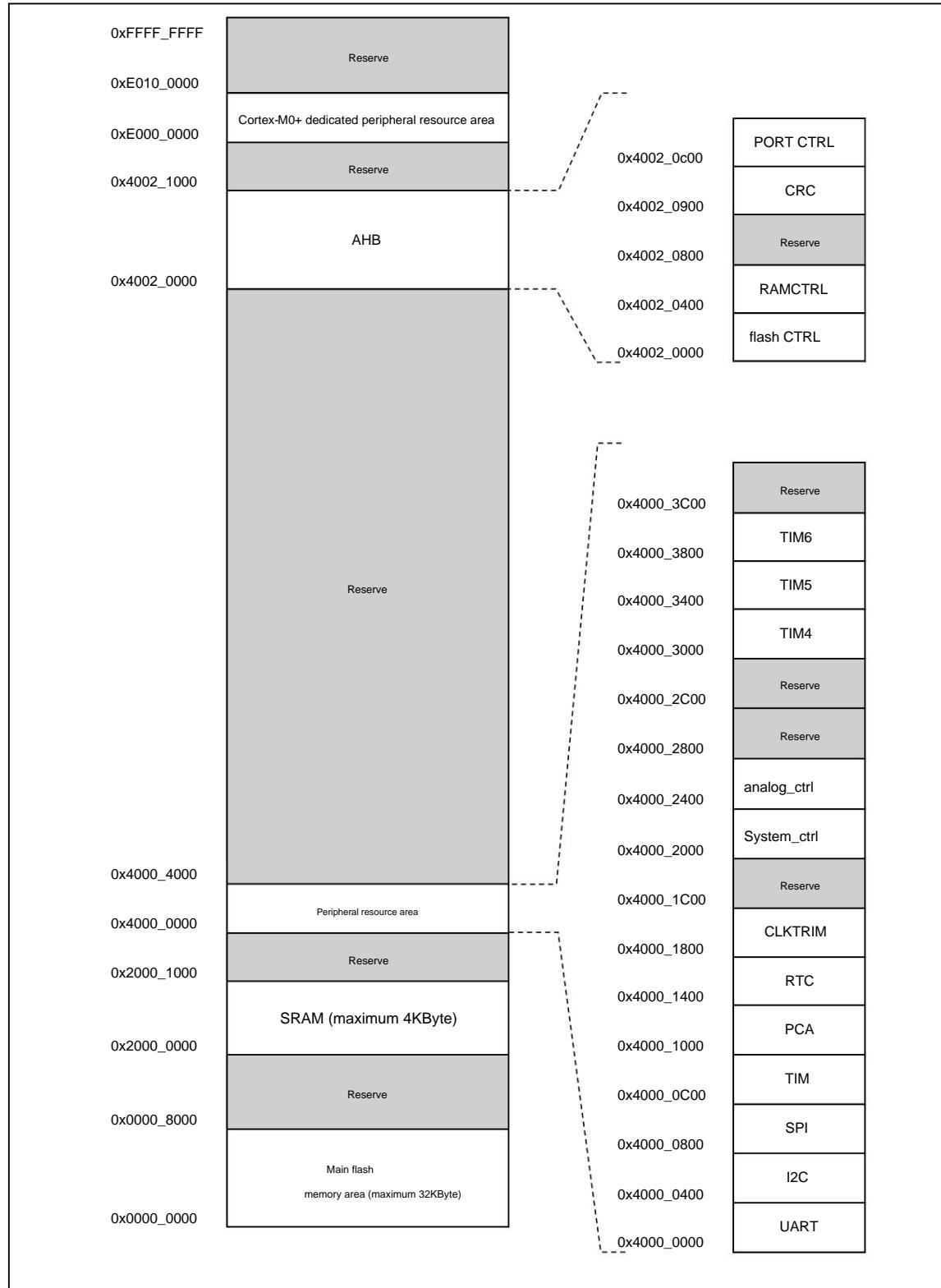


Figure 1-1 System Architecture Diagram

## 1.2 System Address Division

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



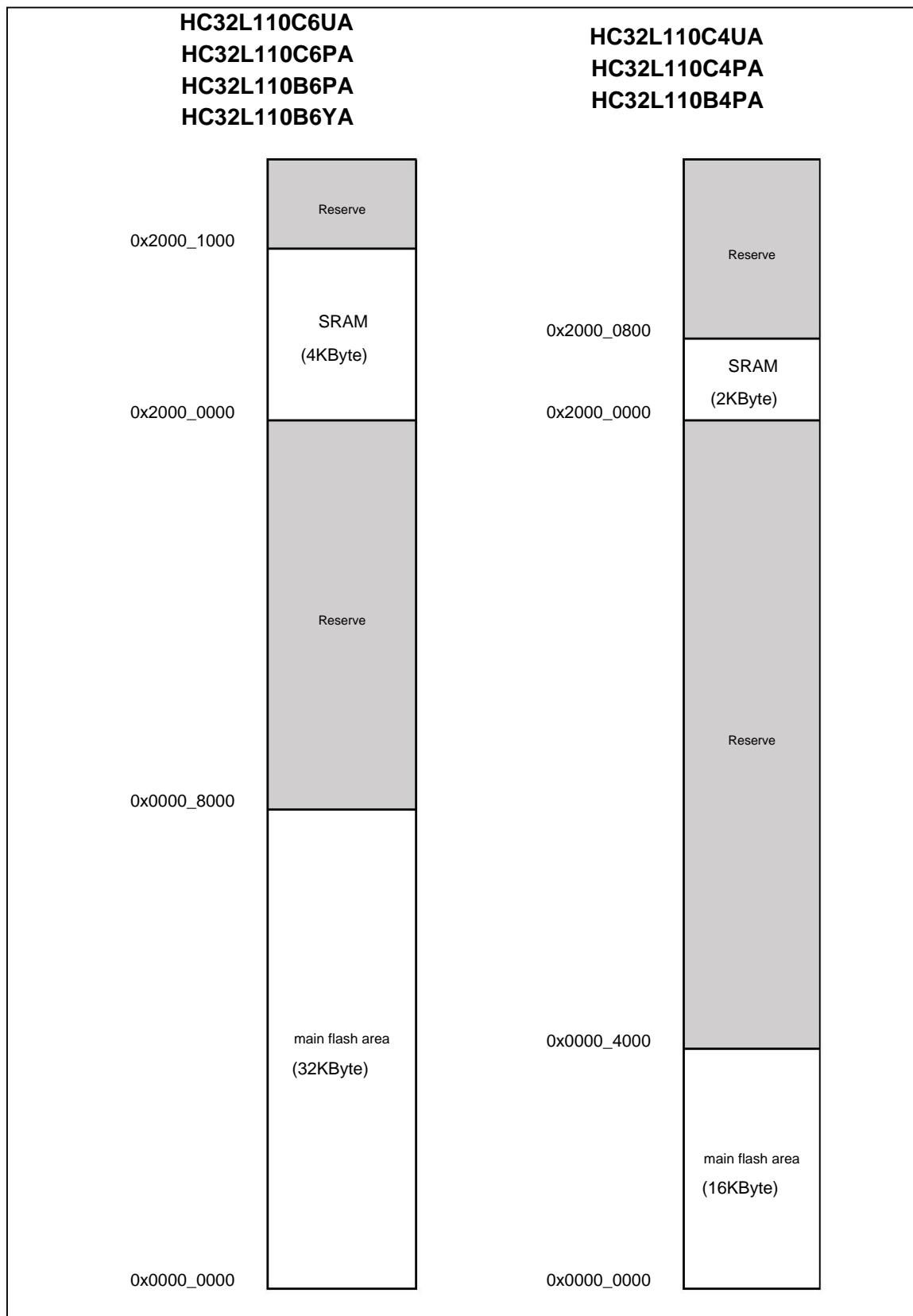


Figure 1-2 Schematic diagram of address area division

### 1.3 Memory and module address allocation

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 USART0		
0x4000_0100 – 0x4000_01FF	256Byte USART1		
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 – 1kByte	1kByte	Reserved	
0x4000_2000 – 0x4000_23FF		SYSTEMCTRL	
0x4000_2400 – 0x4000_27FF		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 working modes

The power management module of this product is responsible for managing the switching between various working modes of this product, and controlling each working mode

The working status of each functional module below. The operating voltage (VCC) of this product is 1.8 ~ 5.5V.

This product has the following working modes:

- (1) Running mode: CPU is running, peripheral function modules are running.
- (2) Sleep mode: CPU stops running and 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, through

Interrupt trigger to return to run mode.

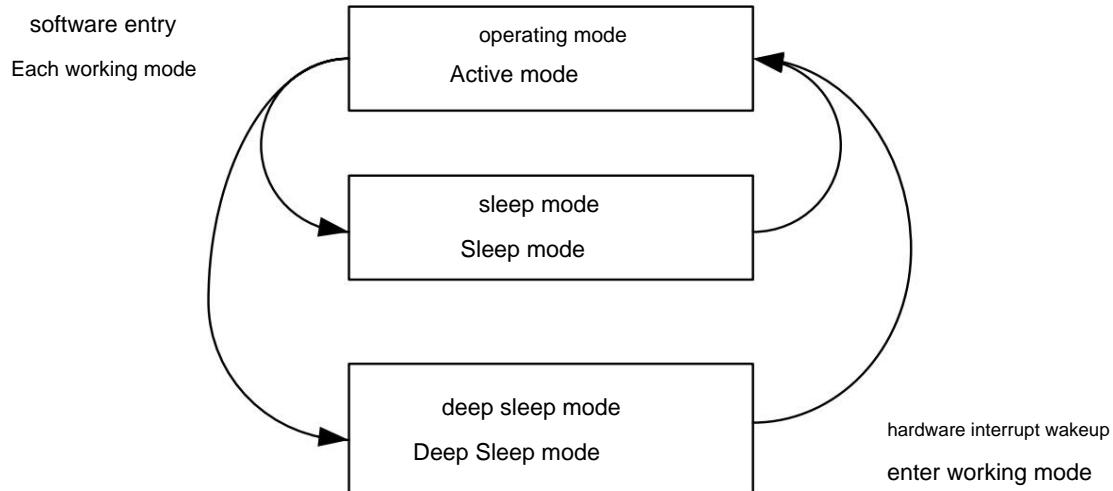


Figure 2-1 Control Mode Block Diagram

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

	interrupt source	Run Mode	Sleep Mode	Deep Sleep Mode	
[0]	GPIO_P0	ÿ	ÿ	ÿ	
[1]	GPIO_P1	ÿ	ÿ	ÿ	
[2]	GPIO_P2	ÿ	ÿ	ÿ	
[3]	GPIO_P3	ÿ	ÿ	ÿ	
[6]	UART0	ÿ	ÿ		
[7]	UART1	ÿ	ÿ		
[8]	LPUART	ÿ	ÿ	ÿ	
[10]	SPI	ÿ	ÿ		
[12]	I2C	ÿ	ÿ		

[14]	Timer0	ÿ	ÿ	
[15]	Timer1	ÿ	ÿ	
[16]	Timer2	ÿ	ÿ	
[17]	LPTimer	ÿ	ÿ	ÿ
[18]	Timer4	ÿ	ÿ	
[19]	Timer5	ÿ	ÿ	
[20]	Timer6	ÿ	ÿ	
[twenty one]	PCA	ÿ	ÿ	
[twenty two]	WDT	ÿ	ÿ	ÿ
[twenty three]	RTC	ÿ	ÿ	ÿ
[twenty four]	ADC	ÿ	ÿ	
[26]	VC0	ÿ	ÿ	ÿ
[27]	VC1	ÿ	ÿ	ÿ
[28]	LVD	ÿ	ÿ	ÿ
[30]	RAMFLASH	ÿ	ÿ	
[31]	CLKTRIM	ÿ	ÿ	ÿ

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

	Reset	Run Mode	Sleep Mode	Deep Sleep Mode
[0]	source power-on power-off reset	ÿ	ÿ	ÿ
[1]	POR ÿ External Reset Pin reset ÿ	ÿ	ÿ	ÿ
[2]	LVD reset	ÿ	ÿ	ÿ
[3]	WDT reset	ÿ	ÿ	ÿ
[4]	PCA reset	ÿ	ÿ	
[5]	Cortex-M0+ LOCKUP hardware reset	ÿ		
[6]	Cortex-M0+ SYSRESETREQ software reset	ÿ		

## 2.1 Operation Mode

The operating mode of this product (Active Mode):

The microcontroller MCU is in the running state after the system is reset at power-on, or after wake-up from various low power consumptions. when the CPU

Various low-power modes can be used to save energy when it is not necessary to continue running, such as while waiting for an external event. user needs

Choose an optimal low-power mode based on conditions such as lowest power consumption, fastest startup time, available wake-up sources, etc.

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

Table 2-1 Operational Module Diagram in Run Mode

Several ways to reduce chip power consumption in run mode:

1) In run mode, by changing the prescaler register ( SYSCLK0.AHB\_CLK\_DIV,

SYSCLK0.APB\_CLK\_DIV) can be programmed to reduce any system clock (HCLK, PCLK)

speed. A prescaler can also be used to reduce the clocks to peripherals before entering Sleep mode.

2) In run mode, turn off the clocks (PERI\_CLKx) of peripherals not in use to reduce power consumption.

3) In run mode, turn off the clocks (PERI\_CLKx) of peripherals that are not used to reduce power consumption, and let the system enter the sleep mode.

Reduce power consumption even more in sleep mode and turn off clocks to peripherals not in use before executing WFI instructions

(PERI\_CLKx).

4) Use low-power mode instead of sleep mode, because the wake-up time of this product is very short (~4uS), it can also meet the needs of the system

real-time response needs.



## 2.2 Sleep Mode

This product sleep mode (Sleep Mode)

Use the WFI command to enter the sleep mode. In sleep mode, the CPU stops running, but the clock module, the system

clock, NVIC interrupt processing and peripheral function modules can still work.

The system enters the sleep state and will not change the port state. Before entering the sleep state, change the IO state to sleep state as needed.

status.

How to enter hibernate mode:

The sleep state is entered by executing the WFI instruction. According to Cortex

<sup>TM</sup>-M0+ in the system control register

The value of the SLEEPONEXIT bit, there are two options for selecting the sleep mode entry mechanism:

SLEEP-NOW: If the SLEEPONEXIT bit is cleared, when WFI or WFE is executed, the microcontroller

Go to sleep mode immediately.

SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the system starts from the lowest priority interrupt handler

Upon exit, the microcontroller immediately enters sleep mode.

How to exit hibernate mode:

If the WFI instruction is executed to enter sleep mode, any high-priority nested vectored interrupt controller responds to the peripheral

Interruptions can wake up the system from sleep mode.

Use Note:

- SLEEP-ON-EXIT This bit is set to 1. After executing the interrupt, it automatically enters sleep, and the program does not need to write `__wfi();`

- SLEEP-ON-EXIT This bit is cleared to 0, `main()` enters sleeping after executing `__wfi()`, the interrupt is triggered and the execution is completed

After the interrupt program returns to `main()`, it executes the WFI instruction and enters sleeping. Wait for subsequent interrupts to trigger.

- The SLEEP-ON-EXIT bit does not affect the execution of the `__wfi()` instruction. SLEEP-ON-EXIT =0: `main()` executes

After `wfi()`, enter sleeping, interrupt is triggered and after executing the interrupt program return to `main()`, continue to execute;

- If sleep is entered in an interrupt, only the interrupt with a priority higher than this interrupt can wake up, execute the high priority first,

Execute low priority again; interrupts with priority lower than or equal to this interrupt cannot wake up.

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

Grayed-out modules do not work in their current state.

Table 2-2 Operational module diagram in sleep mode



## 2.3 Deep Sleep Mode

This product deep sleep mode (Deep Sleep Mode)

Use SLEEPDEEP with the WFI command to enter deep sleep mode. In deep sleep mode, the CPU stops

stop running, high-speed clock is off, low-speed clock can be configured to run, some low-power peripheral modules can be configured to allow

Yes, NVIC interrupt handling still works.

ÿ The system enters deep sleep mode from high-speed clock, the high-speed clock is automatically turned off, and the low-speed clock keeps entering deep sleep mode

previous state.

ÿ The system enters the deep sleep mode from the low-speed clock. Since the low-speed clock will not automatically shut down, it keeps running and enters the sleep mode.

sleep mode. Only the ARM Cortex-M0+ does not run, all other modules run.

ÿ When the system clock is switched, all clocks will not be automatically turned off, and the software needs to be turned off and turned on according to the power consumption and system requirements

corresponding clock.

ÿ The system enters the deep sleep state and will not change the port state. Before entering the sleep state, change the IO state to

state in hibernation. Unused IO pins and IO pins that are not brought out by the package need to be set as input and enabled.

pull.

How to enter deep sleep mode:

First set the SLEEPDEEP bit in the Cortex-M0+ system control register, enter by executing the WFI instruction

sleep state. Depending on the value of the SLEEPONEXIT bit in the Cortex™-M0+ system control register, there are two options

Items can be used to select the deep sleep mode entry mechanism:

SLEEP-NOW: If the SLEEPONEXIT bit is cleared, when WFI or WFE is executed, the microcontroller

Go to sleep mode immediately.

SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the system starts from the lowest priority interrupt handler

Upon exit, the microcontroller immediately enters sleep mode.

How to exit deep sleep mode:

If the WFI instruction is executed to enter sleep mode, any peripheral interrupt that is serviced by the nested vectored interrupt controller

(Peripheral module interrupts that can be run under Deep Sleep) can wake up the system from sleep mode.

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

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

Grayed-out modules do not work in their current state.

Table 2-3 Operational module diagram in deep sleep mode



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

Address: 0xE000ED10

Reset value: 0x0000 0000

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

After entering deep sleep, there are two options for the system clock after waking up. By default, the clock entering deep sleep is used, and the configuration register

After the device SYSCTRL0.wakeup\_byRCH is 1, no matter what clock was before entering deep sleep, it will be used after waking up.

Internal high-speed clock RCH. This setting can speed up the wake-up of the system if an external crystal oscillator is used.

## 3 System Controller (SYSCTRL)

### 3.1 Introduction to Clock Sources

The clock control module mainly controls the system clock and peripheral clock, and can configure different clock sources as the system clock,

Different system clock divisions can be configured and peripheral clocks can be enabled or disabled. In addition, in order to ensure the accuracy of the oscillator,

Internal clocks are calibrated.

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

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

Notice:

– When switching the clock source of the system clock, strictly follow the operation steps to switch, see Chapter 3.2 for details.

– XTL can directly input the 32.768KHz clock signal from the P14 pin without the crystal oscillator. XTH can not be connected

Crystal oscillator, input 4~32MHz clock signal directly from P01 pin.

This product also contains the following two secondary clocks:

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

The following figure shows the clock architecture of this product:

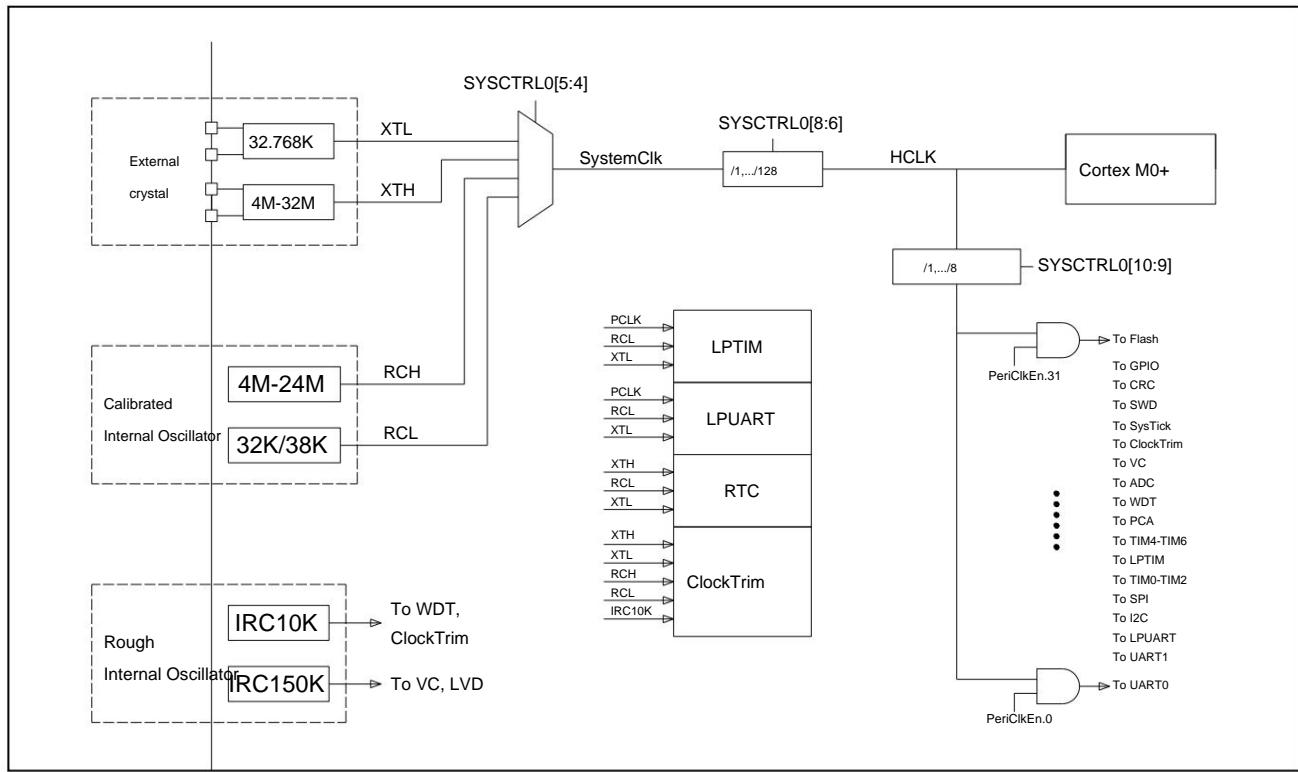


Figure 3-1 Block diagram of clock control module

### 3.1.1 Internal high-speed RC clock RCH

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

The output frequency of RCH can be adjusted by changing the value of register RCH\_CR[10:0]. Register value increases by 1 each time

The output frequency of RCH is increased by about 0.2%, and the total adjustment range is 4~24MHz.

5 preset frequencies 4MHz, 8MHz, 16MHz, 22.12MHz, 24MHz;

Please manually adjust the value of this register.

Changing the RCH output frequency requires a specific change timing, see the System Clock Switching section for details.

The internal high-speed clock only needs 4us from startup to stable. In order to quickly respond to interrupts 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, 32.768KHz. When the system enters Deep Sleep, this low-speed clock will not be automatically turned off, and the ultra-low power consumption

The device module can select RCL as its clock.



### 3.1.3 External low-speed crystal clock XTL

The external low-speed crystal oscillator clock needs to be connected to a 32.768KHz low-power crystal oscillator with ultra-high precision and ultra-low power consumption.

When the system enters Deep Sleep, this low-speed clock will not automatically shut down. Peripheral modules operating in ultra-low power modes can to select XTL as its clock.

XTL can also input 32.768KHz clock signal directly from the P14 pin without the crystal oscillator. Input clock from P14

The signal method is: configure the P14 pin as GPIO input; set SYSCTRL1.EXTL\_EN to 1; set  
SYSCTRL0.XTL\_EN is 1.

Notice:

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

### 3.1.4 External high-speed crystal clock XTH

The external high-speed crystal oscillator clock needs to be connected to a high-speed crystal oscillator of 4 MHz to 32 MHz. When the system enters Deep Sleep,

This high-speed clock is automatically turned off.

XTH can also not connect to the crystal oscillator, and directly input the clock signal of 4MHz ~ 32MHz from the P01 pin. Input from P01

The method of clock signal is: configure P01 pin as GPIO input; set SYSCTRL1.EXTH\_EN to 1; set  
Set SYSCTRL0.XTH\_EN to 1.

Notice:

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

### 3.1.5 Clock startup process

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

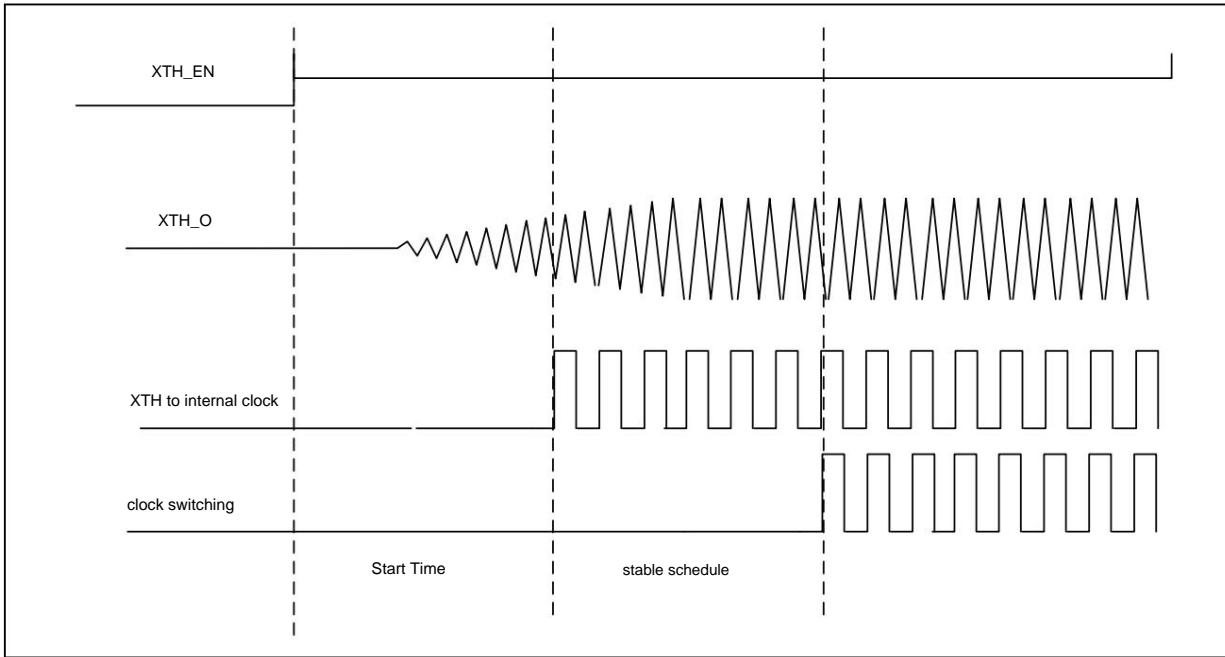


Figure 3-2 Schematic diagram of crystal oscillator clock startup



## 3.2 System Clock Switching

The clock source of the system clock can be switched between RCH, RCL, XTH, XTL through SYSCTRL0[5:4].

The clock switching operation must be performed according to the standard clock switching process, otherwise an exception may occur. If the new clock source

If the frequency is greater than 24MHz, you need to set FLASH\_CR.WAIT to 1.

Note: rewrite `FLASH_CR.WAIT` value, it is necessary to first `FLASH_BYPASS` Registers are written in sequence `0x5A5A, 0xA5A5` Then

Right again `FLASH_CR.WAIT` assign, see `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 the appropriate mode.

Note: An analog pin is required when connecting to an external crystal oscillator; it is required when connecting an external clock input `GPIO` Input and enable 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 according to the flow of the Flash Controller chapter

`FLASH_CR.WAIT`.

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

Step6: Configure `SYSCTRL0.Clk_sw4_sel`, and 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 procedure in the Flash Controller chapter.

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

### 3.2.2 Example of switching from **RCH** to **XTL**

The operation process is as follows:

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

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

Step3: Write `0x5A5A` and `0xA5A5` to the `SYSCTRL2` register in turn to enable register rewriting.

Step4: Set `SYSCTRL0.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 `SYSCTRL2` register in turn to enable register rewriting.



Step7: Set SYSCTRL0. Clk\_sw4\_sel to 3, and switch the system clock to XTL.

Step8: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in turn to enable register rewriting.

Step9: Set SYSCTRL0.RCH\_EN to 0, turn off the RCH oscillator.

### 3.2.3 Example of switching from **RCH** 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: Configure XTH\_CR.Driver according to the crystal oscillator characteristics.

Step3: Set XTH\_CR. Startup to 3, and select the longest crystal oscillator stabilization time.

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

Step5: Set SYSCTRL0.XTH\_EN to 1 to enable the crystal oscillator circuit.

Step6: According to the crystal frequency, configure FLASH\_CR.WAIT.

Step7: After the query waits for the XTH\_CR. Stable flag to become 1, the software delays more than 10ms, and the crystal oscillator output is stable clock.

Step8: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in turn to enable register rewriting.

Step9: Set SYSCTRL0. Clk\_sw4\_sel to 1, and switch the system clock to XTH.

Step10: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in turn to enable register rewriting.

Step11: Set SYSCTRL0.RCH\_EN to 0 and turn off the RCH oscillator.

The following figure shows the clock switching timing diagram:

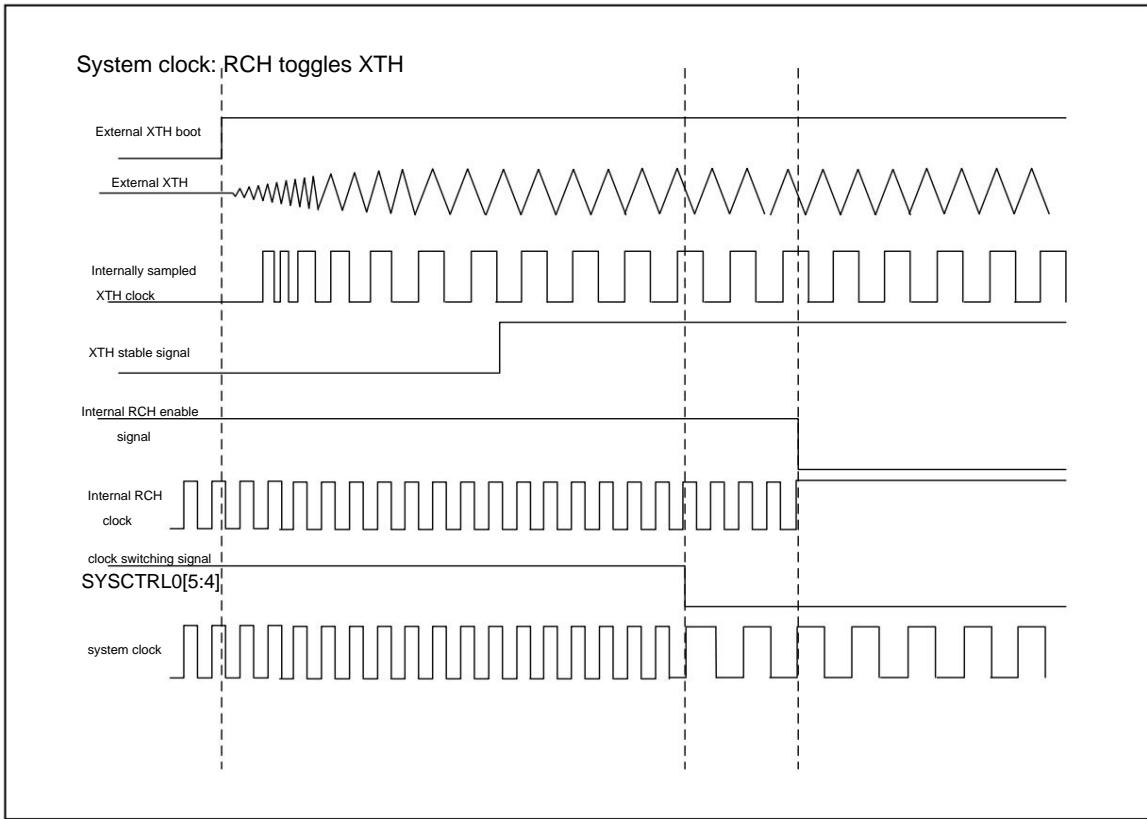


Figure 3-3 Clock switching diagram

### 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: Configure XTH\_CR.Driver according to the crystal oscillator characteristics.

Step3: Set XTH\_CR. Startup to 3, and select the longest crystal oscillator stabilization time.

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

Step5: Set SYSCTRL0.XTH\_EN to 1 to enable the crystal oscillator circuit.

Step6: According to the crystal frequency, configure FLASH\_CR.WAIT.

Step7: After the query waits for the XTH\_CR. Stable flag to become 1, the software delays more than 10ms, and the crystal oscillator output is stable clock.

Step8: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in turn to enable register rewriting.

Step9: Set SYSCTRL0. Clk\_sw4\_sel to 1, and switch the system clock to XTH.



Step10: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in turn to enable register rewriting.

Step11: Set SYSCTRL0.RCL\_EN to 0 and turn off the RCL oscillator.

### 3.2.5 Example of switching from **RCH** to **RCL**

The operation process is as follows:

Step1: Configure RCL\_CR.TRIM and RCL\_CR. Startup.

Step2: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in turn to enable register rewriting.

Step3: Set SYSCTRL0. RCL\_EN to 1 to enable the RCL oscillator circuit.

Step4: Query and wait for the RCL\_CR. Stable flag to become 1, and the RCL outputs a stable clock.

Step5: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in turn to enable register rewriting.

Step6: Set SYSCTRL0. Clk\_sw4\_sel to 2, and switch the system clock to RCL.

Step7: To turn off the RCH oscillator, perform the following operations: write 0x5A5A,

0xA5A5, enable register rewriting; set SYSCTRL0. RCH\_EN to 0, 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 SYSCTRL2 register in turn to enable register rewriting.

Step3: Set SYSCTRL0. RCH\_EN to 1 to enable the RCH oscillator circuit.

Step4: Query and wait for the RCH\_CR. Stable flag to become 1, and the RCH outputs a stable clock.

Step5: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in turn to enable register rewriting.

Step6: Set SYSCTRL0. Clk\_sw4\_sel to 0, and switch the system clock to RCH.

Step7: To turn off the RCL oscillator, perform the following operations: write 0x5A5A,

0xA5A5, enable register rewriting; set SYSCTRL0.RCL\_EN to 0, turn off the RCL oscillator.

### 3.2.7 Switching between different oscillation frequencies of RCH

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

Scenario 1:

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

Step5: Set SYSCTRL0.HCLK\_PRS to 0x0.

Example code to switch from 4M to 24M is shown below:

```
M0P_SystemCtrl->SYSCTRL2 = 0X5A5A;  
M0P_SystemCtrl->SYSCTRL2 = 0XA5A5;  
M0P_SystemCtrl->SYSCTRL0_f.HCLK_PRS = 7;  
M0P_SystemCtrl->RCH_CR = *((uint16 *) (0X00100C08) ); //4M  
M0P_SystemCtrl->RCH_CR = *((uint16 *) (0X00100C06) ); //8M  
M0P_SystemCtrl->RCH_CR = *((uint16 *) (0X00100C04) ); //16M  
M0P_SystemCtrl->RCH_CR = *((uint16 *) (0X00100C00) ); //24M  
M0P_SystemCtrl->SYSCTRL2 = 0X5A5A;  
M0P_SystemCtrl->SYSCTRL2 = 0XA5A5;  
M0P_SystemCtrl->SYSCTRL0_f.HCLK_PRS = 0;
```

Scenario 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 and change the frequency of RCH oscillation.

Step3: Switch the system clock to RCH, see 3.2.6 Example of switching from RCL to RCH at low speed.

### 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 with each other.

After the reference clock and the calibrated clock, set the register REFCNT value, set cali.start to start the clock calibration power

way, at this time two 32-bit counters (increment, decrement) work at the same time, when the down counter is equal to 0, cali.finish

is set to indicate that the calibration is over, at this time the software can read the CALCNT value, so it is easy to get the reference clock

Frequency relationship to the clock being calibrated.

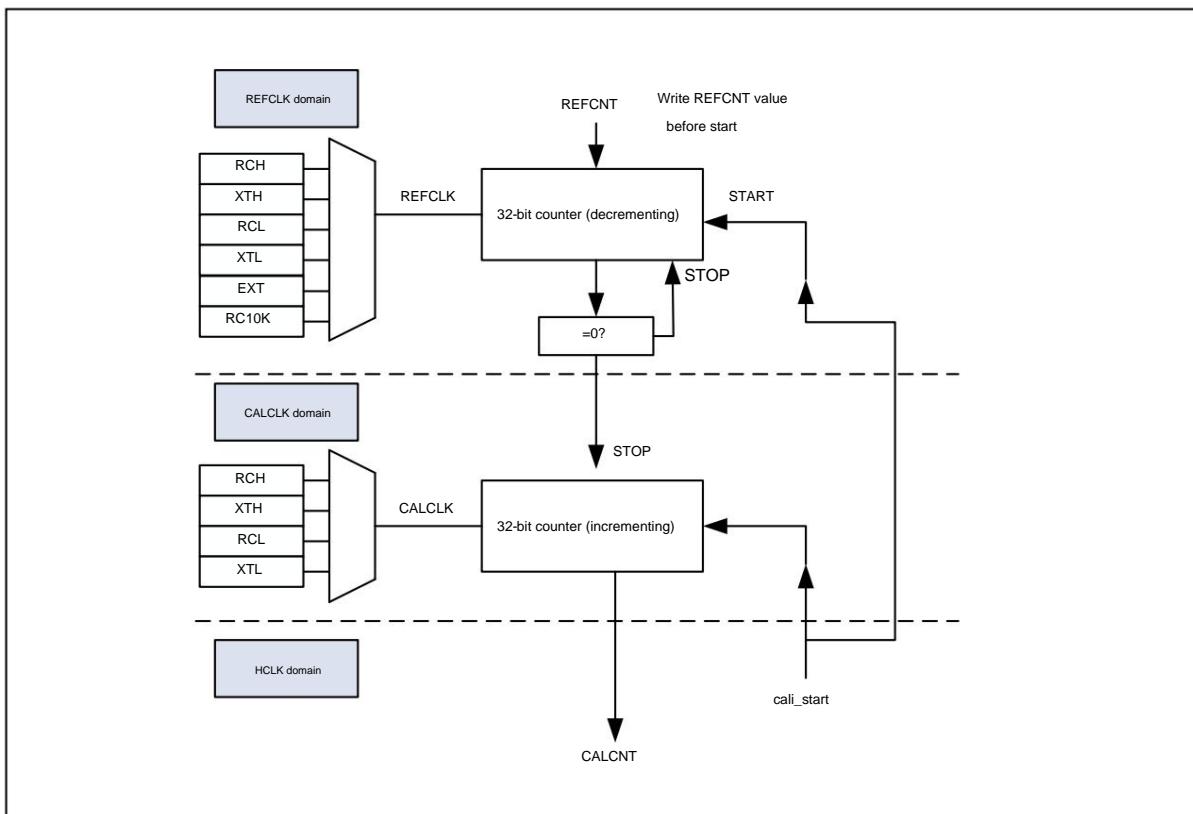


Figure 3-4 Schematic diagram of clock calibration

### 3.4 Interrupt wake-up control

When the processor executes a WFI instruction and goes to sleep, it stops executing instructions. An interrupt occurred during sleep

The processor wakes up when a request (higher priority) needs to be processed.

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

PRIMASK status	WFI behavior	wake	ISR implementation
0	IRQ priority > current level	Y	Y
0	IRQ priority = current level	N	N
1	IRQ priority > current level	Y	N
1	IRQ priority = current level	N	N

#### 3.4.1 The method of executing the interrupt service routine after waking 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 SCB->SCR.SLEEPDEEP to 1
4. Execute WFI instruction to enter deep sleep mode
5. The system enters deep sleep mode and waits for the interrupt to wake up, and then executes the interrupt service routine after waking up

Routine:

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

#### 3.4.2 The method of not executing the interrupt service routine after waking 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 the WFI instruction to enter deep sleep mode
6. The system enters deep sleep mode and waits for the interrupt to wake up, and executes the next instruction after wake up
7. Clear interrupt flag, clear interrupt pending status

Routine:

```

__asm("CPSID I");           //Set PRIMASK

SCB_SCR |= 0x00000004u;

while(1)

{

    __asm("WFI");

    BTIMERLP_REG->TCR_f.TFC=0;           //Clear Int Flag

    NVIC_ClearPendingIRQ(BASE_TIMER3_IRQn); //Clear Pending Flag

}

```

### 3.4.3 Using the Exit Hibernation feature

Sleep-on-exit is ideal for interrupt-driven applications. When this feature is enabled, as long as the

After processing and returning to thread mode, the processor will enter sleep mode. Using the exit sleep feature, the processor

Can be in sleep mode as much as possible.

Cortex-M0 uses the exit sleep feature to enter sleep, which is the same as executing WFI immediately after executing abnormal exit.

The effect is about the same. However, in order to enter the exception next time, there is no need to push the stack, the processor will not execute the stack process.

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

Routine:



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

### 3.5 Register

base address 0x40002000

register	offset address	describe
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

Table 3-1 System Control Register Table

### 3.5.1 System Control Register 0 (SYSCTRL0)

Offset address: 0x000

Reset value: 0x0000 0001

31	30	29	28	27	26	25	twenty four	twenty three	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_byRCH	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	Marker function description	
31:16	Reserved	
15	wakeup_byRCH	1: After waking up from Deep Sleep, the source of the system clock is RCH, and the original clock continues to be enabled. 0: Do not change the system clock source after waking up from Deep Sleep.
14:11	Reserved	
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: Disable 1: Enable external high-speed crystal oscillator XTH Enable control 0: Disable 1:
1	XTH_EN	Enable Note: When the system enters DeepSleep, this high-speed clock will be automatically turned off. Internal high-speed clock RCH enable signal. 0: Disable 1: Enable Note: When the system enters DeepSleep, this high-speed clock will automatically shut down.
0	RCH_EN	

Notice:

- Every time you rewrite the values of SYSCTRL0 and SYSCTRL1, you need to write 0x5A5, 0xA5A5. Such steps can effectively prevent misoperation of the SYSCTRL0 and SYSCTRL1 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_always_on	EXTL_EN	EXTH_EN	Res	
				R/W		R/WR	WR/WR	WR/W			R/WR/W/R/W				

bit mark		Function 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 instruction, which can enhance the 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 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: The XTL output clock is input from P14. 0: XTL output clock is generated by crystal oscillator.  Note: When using P14 input clock, set SYSCTRL0.XTL_EN to 1.



1	EXTH_EN	External XTH input control 1: The XTH output clock is input from P01. 0: XTH output clock is generated by crystal oscillator. Note: When using P01 to input the clock, set SYSCTRL0.XTH_EN to 1.
0	Reserved	

Notice:

- Every time you rewrite the values of SYSCTRL0 and SYSCTRL1, you need to write 0x5A5, 0xA5A5. Such steps can effectively prevent misoperation of the 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 mark		Function description
31:16	Reserved	
15:0	SYSCTRL2	<p>Register SYSCTL0, SYSCTRL1 protect the serial control register, write to SYSCTRL2 first 0x5A5A, then write 0xA5A5 to start the write operation for the registers SYSCTL0 and SYSCTRL1, only To write to the registers SYSCTL0, SYSCTRL1, this protection bit automatically restores the protection state, Need to rewrite series to open protection.</p>

### 3.5.4 RCH Control Register (RCH\_CR)

Offset address: 0x00C

Reset value: 0x00000126

31	30	29	28	27	26	25								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 mark		Function description
31:12	Reserved	
11	stable	<p>Internal high-speed clock RCH stable flag.</p> <p>1: Indicates that the RCH is stable and can be used by the internal circuit.</p> <p>0: It means that RCH is not stable and cannot be used by the internal circuit.</p>
10:0	TRIM	<p>Clock frequency adjustment, changing the value of this register can adjust the output frequency of RCH. register value</p> <p>For each increase of 1, the output frequency of the RCH increases by about 0.2%, and the total adjustment range is 4~24MHz.</p> <p>The calibration values of 5 groups of frequencies have been saved in the Flash, read and write the calibration values in the Flash</p> <p>RCH_CR.TRIM to get the exact frequency.</p> <p>24M calibration value address: 0x00100C00 - 0x00100C01</p> <p>22.12M Calibration value address: 0x00100C02 - 0x00100C03</p> <p>16M calibration value address: 0x00100C04 - 0x00100C05</p> <p>8M calibration value address: 0x00100C06 - 0x00100C07</p> <p>4M calibration value address: 0x00100C08 - 0x00100C09</p> <p>The change of the RCH output frequency needs to follow a specific timing. For details, please refer to chapter 3.2.7 for system clock switching.</p>

### 3.5.5 Oscillation XTH Control Register (XTH\_CR)

Offset address: 0x010

Reset value: 0x00000022

31	30	29	28	27	26	25								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 mark		Function description
31:6	Reserved	
6	stable	<p>External high-speed clock XTH stable flag.</p> <p>1: Indicates that XTH is stable and can be used by the internal circuit.</p> <p>0: Indicates that XTH is not stable and cannot be used by the internal circuit.</p> <p>Note: In order to increase the reliability of the system, after the flag is queried, the software needs to delay more than 10ms. Only then can the system clock be switched to XTH.</p>
5:4	Startup	<p>External high-speed clock XTH stabilization time selection</p> <p>00: 256 cycles;</p> <p>01: 1024 cycles;</p> <p>10: 4096 cycles;</p> <p>11: 16384 cycles;</p> <p>Note: It is strongly recommended to set the stabilization time of XTH to 11. If the XTH stabilization time is insufficient, the system does not work stably during clock switching or waking up from deep sleep.</p>
3:0	Driver	<p>3:2 Freq selects the operating frequency of the crystal oscillator</p> <p>11: 24M~32M</p> <p>10: 16M~24M</p> <p>01: 8M~16M</p> <p>00: 4M~8M</p> <p>1:0 Driver Select the drive capability of the crystal oscillator</p> <p>11: The strongest driving ability</p> <p>10: Default drive capability (recommended value)</p> <p>01: Weak drive capability</p> <p>00: Weakest drive capability</p> <p>Note: It is necessary to select the appropriate driving energy according to the characteristics of the crystal oscillator, the load capacitance and the parasitic parameters of the circuit board. force. The greater the driving ability, the greater the power consumption; the weaker the driving ability, the less the power consumption.</p>

### 3.5.6 RCL Control Register (RCL\_CR)

Offset address: 0x014

Reset value: 0x0000033Fh

31	30	29	28	27	26 25 24					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 mark		Function description
31:13	Reserved	
12	stable	Internal low-speed clock RCL stable flag. 1: Indicates that the RCL is stable and can be used by the internal circuit. 0: It means that RCL is not stable and cannot be used by the internal circuit.
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	The internal low-speed clock frequency is adjusted, and the calibration values of 2 groups of frequencies are saved in the Flash. Read and write the calibration value in Flash to RCL_CR.TRIM to get the accurate frequency. 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								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 mark		Function description
31:6	Reserved	
6	stable	External low-speed crystal oscillator XTL stable flag. 1: Indicates that XTL is stable and can be used by internal circuits. 0: Indicates 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 drive selection 1111: Maximum drive 0000: Minimum drive 3:2 Amp_control Minor 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 capability (recommended value) 00: Weakest drive capability, <small>Note: It is necessary to select the appropriate driving energy according to the characteristics of the crystal oscillator, the load capacitance and the parasitic parameters of the circuit board. force. The greater the driving ability, the greater the power consumption; the weaker the driving ability, the less the power consumption. 00: Weakest drive energy force</small>

### 3.5.8 Peripheral module clock control register (PERI\_CLKEN)

Reset value: 0xC080\_0000

Offset address: 0x020

31	30	29	28	27	26	25	twenty four	seventy five	seventy six	seventy seven	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		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 UART1	UART0	R/W	R/W	R/W	R/W	R/W
R/W				R/W		R/W									

bit mark		Function 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: Executing programs from flash is not affected by this bit.
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
twenty four	TICK	SysTick timer reference clock enable. 1: Enable; 0: Disable
23:22	Res.	reserved bit
seventy one	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	twenty four	Accurate time	Accurate time	Accurate time	20 19	18	17	16
Reserved			CLK_SEL	NO REF	SKE W	STCALIB [23:16]							
			R/W	R/WR/W		R/W							
15	14	13	12	11	10	9	8	7	6	5	4	3	2
STCALIB[15:0]													0
R/W													

bit mark		Function 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] <small>Note: When using an external reference clock, the reference clock frequency is not allowed to be higher than HCLK</small>
23:0	SKEW	STCALIB Accuracy Indication 1: STCALIB value represents roughly 10ms 0: STCALIB value represents exact 10ms
23:0	STCALIB	When the reference clock is XTL, the calibration value of 10 ms

## 4 Reset controller (RESET)

### 4.1 Introduction of reset controller

This product has 7 reset signal sources, each reset signal can make the CPU run again, most registers

The counter will be reset to the reset value and the program counter PC will be reset to point to 00000000.

þ POR/BOR reset (VCC domain and Vcore domain)

þ External Reset PAD reset

þ WDT reset

þ PCA reset

þ LVD reset

þ Cortex-M0+ SYSRESETREQ software reset

þ Cortex-M0+ LOCKUP hardware reset

Each reset source is indicated by a corresponding reset flag. The reset flags are all 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 during power-on reset, then the reset\_flag can be passed in the next reset

The relevant bits of , determine the reset source.

The following figure describes the reset source of each area.

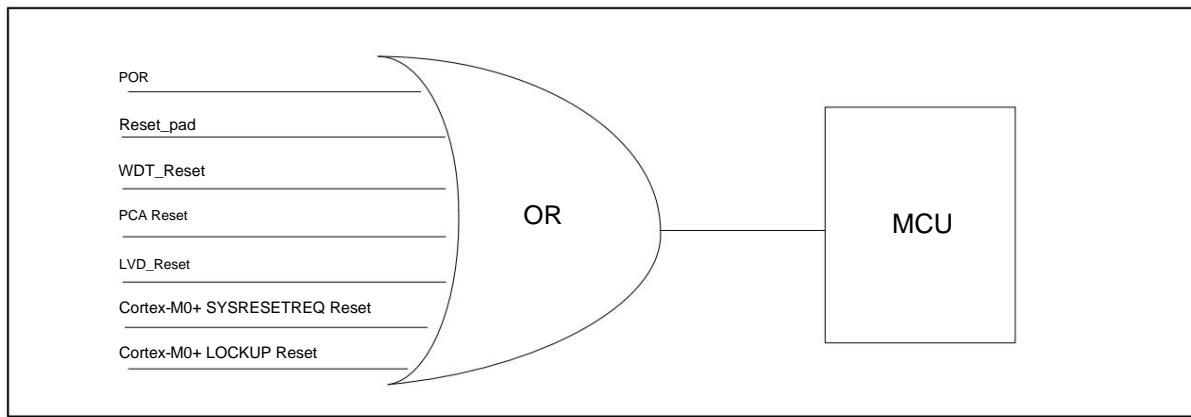


Figure 4-1 Schematic diagram of reset source



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

This product has two power supply areas: **VCC area**, **Vcore** area. All analog modules and **IO** work on **VCC**

region; other modules work in the **Vcore** region.

When the VCC region is powered on, when the VCC voltage is lower than the POR threshold voltage (typically 1.65V), a POR5V will be generated

Signal; when the VCC region is powered off, when the VCC voltage is lower than the BOR threshold voltage (typically 1.5V), it will generate

POR5V signal.

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

When the region is powered down, the POR15V signal is generated when the Vcore voltage is lower than the BOR threshold voltage.

Both the POR5V signal and the POR15V signal will reset the chip's registers 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 built-in pull-up resistors and

A glitch filter circuit is integrated. The glitch filter circuit will filter the glitch signal less than 20us (typical value), because

Therefore, the low-level signal added to the reset pin must be greater than 20us to ensure that the chip is reset reliably.

### 4.1.3 WDT reset

Watchdog reset, please refer to the WDT chapter.

#### 4.1.4 PCA reset

PCA reset, please refer to the PCA chapter for instructions.

#### 4.1.5 LVD low voltage reset

For LVD reset, please refer to the LVD chapter.

#### 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 and lock itself.



---

and resets the entire CORE region after a delay of several clock cycles.

## 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	Twenty four	Twenty three	Twenty two	Twenty one	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	RW0

bit mark		Function description
31:8	Reserved	
7	RSTB	RESETB port reset flag, software initialization and clearing are required, the power-on state is uncertain 1: A port reset occurred 0: No port reset occurred
6	Sysreq	Cortex-M0+ CPU software reset flag, software initialization and clearing are required, and the power-on state is uncertain 1: Cortex-M0+ CPU software reset occurred 0: No Cortex-M0+ CPU software reset has occurred
5	Lockup	Cortex-M0+ CPU Lockup reset flag, software initialization and clearing are required, the power-on state is uncertain 1: A Cortex-M0+ CPU Lockup reset occurred 0: No Cortex-M0+ CPU Lockup reset occurred
4	PCA	PCA reset flag, software initialization and clearing are required, the power-on state is uncertain 1: PCA reset occurred 0: No PCA reset occurred
3	WDT	WDT reset flag, software initialization and clearing are required, the power-on state is uncertain 1: WDT reset occurred 0: No WDT reset occurred
2	LVD	LVD reset flag, software initialization and clearing are required, the power-on state is uncertain 1: LVD reset occurred 0: No LVD reset occurred
1	POR15V	Vcore domain reset flag 1: The Vcore domain is reset 0: No reset occurs in the Vcore domain
0	POR5V	VCC power domain reset flag 1: A reset occurs in the VCC power domain 0: No reset occurs in the 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					20 19	18	17	16

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

Bit tag	function	description												
31:29	Res. Reserved	bit												
28	GPIO	GPIO module reset enable. 1: normal work; 0: the module is in reset state												
27	Res. Reserved	bit												
26	CRC	CRC module reset enable. 1: normal work; 0: the module is in reset state												
25	Res. Reserved	bit												
	TICK	SYSTICK module reset enable. 1: normal work; 0: the module is in reset state												
23:22	Res. Reserved	bit												
	TRIM	CLKTRIM module reset enable. 1: normal work; 0: the module is in reset state												
20	RTC	RTC module reset enable. 1: normal work; 0: the module is in reset state												
19-18	Res. Reserved	bit												
17	VC	VC, LVD, module reset enable. 1: normal work; 0: the module is in reset state												
16	ADC	ADC module reset enable. 1: normal work; 0: the module is in reset state												
15	Res. Reserved	bit												
14	PCA	PCA module reset enable. 1: normal work; 0: the module is in reset state												
13:11	Res. Reserved	bit												
10	ADVTIM	Timer456 module reset enable. 1: normal work; 0: the module is in reset state												



9	LPTIM	LPTimer module reset enable. 1: normal work; 0: the module is in reset state
8	BASETIM	Timer012 module reset enable. 1: normal work; 0: the module is in reset state
7	Res. Reserved	bit
6	SPI	SPI module reset enable. 1: normal work; 0: the module is in reset state
5	Res. Reserved	bit
4	I2C	I2C module reset enable. 1: normal work; 0: the module is in reset state
3	Res. Reserved	bit
2	LPUART	LPUART module reset enable. 1: normal work; 0: the module is in reset state
1	UART1	UART1 module reset enable. 1: normal work; 0: the module is in reset state
0	UART0	UART0 module reset enable. 1: normal work; 0: the 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) that supports up to 32 Interrupt Requests (IRQs) input, and 1 non-maskable interrupt (NMI) input (not used in this product system). In addition, processing

The processor also supports multiple inner exceptions.

Each exception source has a separate exception number, and each exception type has a corresponding priority.

Advances are fixed, while some are programmable. The details are shown in the following table:

exception number	exception type	priority	describe
1	reset	-3 (highest)	reset
2	NMI	-2	Non-maskable interrupts (not used in this system)
3	hardware error	-1 reserved	error handling exception
4-10		NA	...
11	SVC	programmable	Invoke the hypervisor through the SVC instruction
12-13 Reserved		NA	...
14	PendSV	programmable	suspendable requests for system services
15	SysTick	programmable	SysTick timer
16	Interrupt	programmable	External Interrupt #0
17	#0 Interrupt #1	programmable	External Interrupt #1
...	...	...	...
47	Break #31		External Interrupt #31

Programmable table 5-1 Cortex-M0+ processor exception list

This chapter only introduces the processor's 32 external interrupt requests (interrupt #0 to interrupt #31) in detail.

For details of exceptions, please refer to other related documents. At the same time, this chapter only discusses the interrupts of the NVIC in the processor core

Processing mechanism, 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 interrupt priority

The highest two bits of the level register, each register occupies 1 byte (8 bits). Under this setting, the available priority

The levels 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 Priority register using only the upper two bits

If the processor is already running another interrupt handler, and the new interrupt has a higher priority than the one being executed, then

Preemption will occur. Running interrupt processing is suspended in favor of a new interrupt, a process commonly referred to as

Break nesting. After the new interrupt is executed, the previous interrupt processing continues and returns to the

in the program thread.

If the processor is running another interrupt handler with the same or higher priority, the new interrupt will wait and

and enter the suspended state. Pending interrupts will wait until the current interrupt level changes, e.g. currently running interrupts

After processing is complete and returns, the current priority is reduced to less 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, interrupt #0 will

Execute first.

### 5.3 Interrupt Vector Table

When the Cortex-M0+ processor wants to process an interrupt service request, it needs to first determine the starting address of the exception handling, so

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 system

The exception (interrupt) vector of the exceptions (interrupts) available in the system, and the initial value of the main stack pointer (MSP).

memory address		exception number
0x0000004C		19
0x00000048	Interrupt #3 Vector	18
0x00000044	Interrupt #2 Vector	17
0x00000040	Interrupt #1 Vector	16
0x0000003C	Interrupt #0 Vector	15
0x00000038	SysTick vector	14
0x00000034	PendSV vector	13
0x00000030	not used not used	12
0x0000002C	SVC vector	11
0x00000028	not used	10
0x00000024	not used	9
0x00000020	not used	8
0x0000001C	not used	7
0x00000018	not used	6
0x00000014	not used	5
0x00000010	not used	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 interrupt

The address of the vector is the interrupt number multiplied by 4, and each interrupt vector is the starting address of interrupt processing.

## 5.4 Interrupt Input and Suspend Behavior

In the NVIC module of the Cortex-M0+ processor, each interrupt input corresponds to a pending status register, and

Each register has only 1 bit to hold an interrupt request, regardless of whether the request is acknowledged or not. When the processor

Hardware will automatically clear the pending status bit when it starts processing this interrupt.

The peripherals of this system use level-triggered interrupt output. When an interrupt event occurs, since the peripherals are connected to the NVIC,

The interrupt signal is acknowledged. Until the processor services the interrupt and clears the peripheral's interrupt signal, this signal will

keep it high. Inside the NVIC, when an interrupt is detected, the pending state of the interrupt will be set.

The pending state is cleared when the processor receives the interrupt and begins executing the interrupt service routine. The process is shown in Figure 5-3

shown:

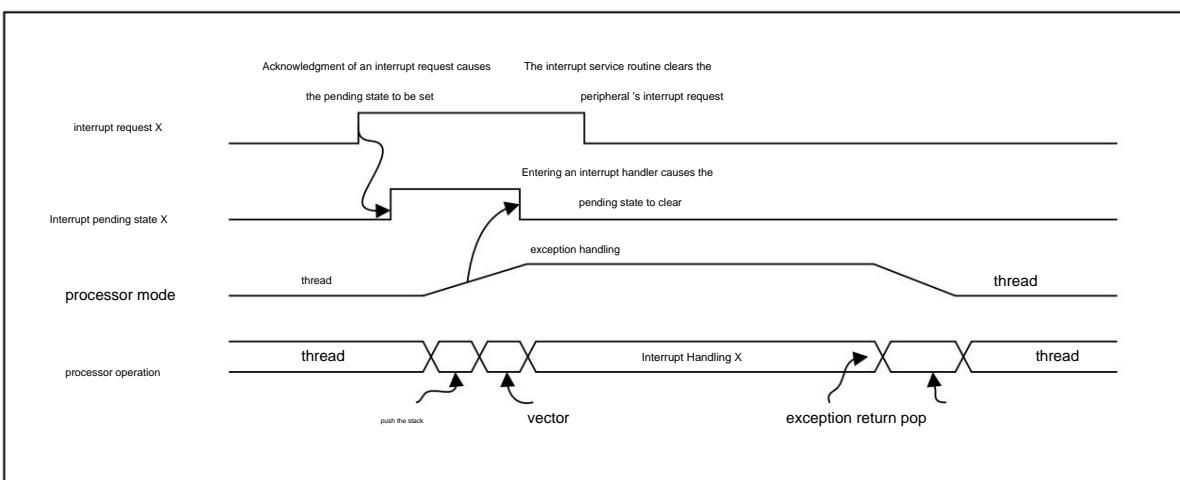


Figure 5-3 Interrupt Active and Pending States

If the interrupt request is not executed immediately, and is cleared by software before acknowledging, the processor will ignore this

request, and no interrupt handling is performed. The interrupt pending status can be cleared by writing to the NVIC\_CLRPEND register.

state, which is useful when setting up a peripheral that may have generated an intermediate

interrupt request.

If the peripheral still holds an interrupt request when the suspend state is cleared by software, the suspend state is also generated immediately. It should have

The process is shown in Figure 5-4:

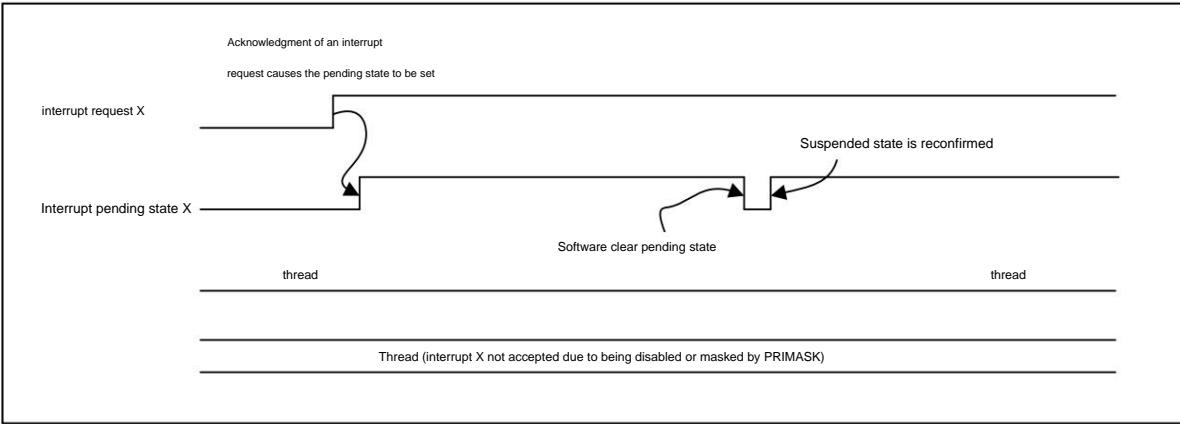


Figure 5-4 Interrupt pending state is cleared and then re-acknowledged

If the interrupt request generated by the peripheral is not cleared when the exception is handled, the pending state will be reset again after the exception returns.

Activate so that the interrupt service routine will execute again. The process is shown in Figure 5-5:

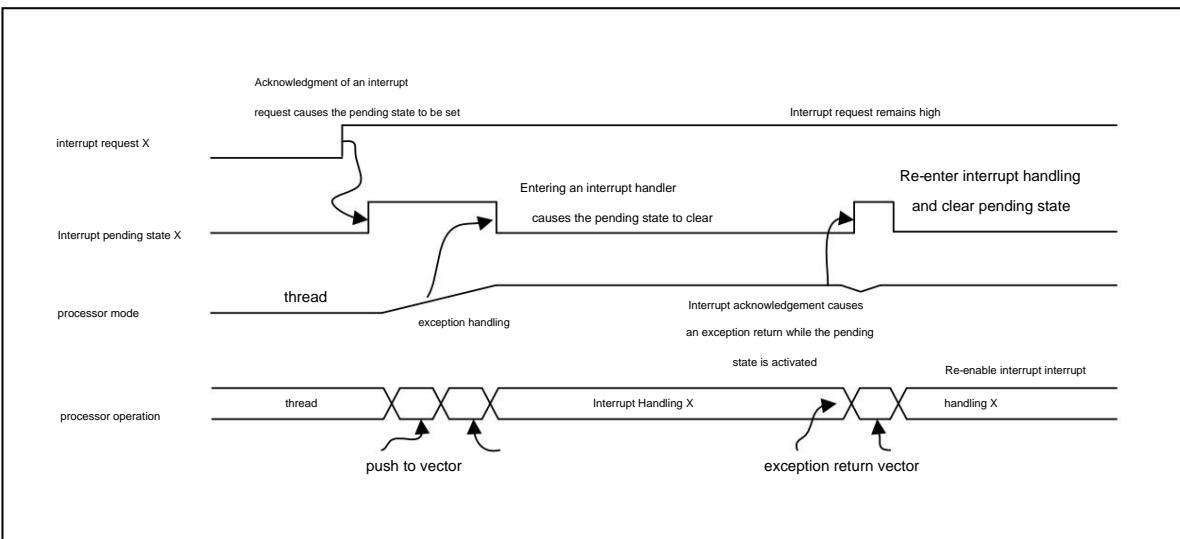


Figure 5-5 If the interrupt request remains high when the interrupt exits, it will cause the interrupt processing to be executed again

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, it will also cause the interrupt service routine to be executed again. The process is shown in Figure 5-6:

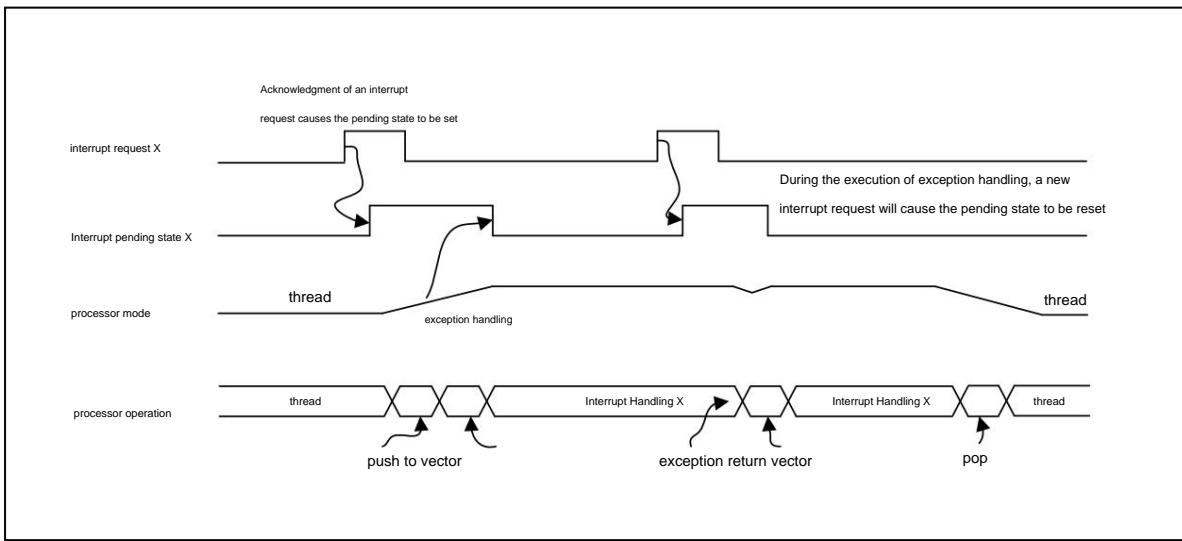


Figure 5-6 Interrupt pending during interrupt processing can also be acknowledged

## 5.5 Interrupt wait

Typically, the NVIC's interrupt latency is 16 cycles. This wait time from the interrupt acknowledgement processor when

The clock cycle starts until the interrupt processing starts and ends. The following prerequisites are required to calculate the interrupt wait:

ÿ The interrupt is enabled and not masked by SCS\_PRIMASK or other exception handling in progress.

ÿ The memory system does not have any wait states, and is fetched at the beginning of interrupt processing, stacking, vector or interrupt processing.

Refers to the bus transfer will be used, if the memory system needs to wait, then the wait state generated when the bus transfer occurs

state may delay interrupts.

The following situations may cause different interrupt waits:

ÿ The end of the interrupt chain, if another interrupt request is generated when the interrupt returns, the processor will skip the pop and

The process of pushing the stack, thus reducing the interrupt waiting time.

ÿ Delayed arrival, if another low-priority interrupt is being pushed to the stack when the interrupt occurs, due to the delay

The existence of the arrival mechanism, the high-priority interrupt will be executed first, which will also lead to the waiting time of the high-priority interrupt

time decreases.

## 5.6 Interrupt Sources

Because the NVIC of the Cortex-M0+ processor supports up to 32 external interrupts, and in this system, the external interrupt source

More than 32, so some external interrupts are multiplexed on the same NVIC interrupt input, and NMI (non-maskable

interrupt) is not used. The corresponding relationship between all external interrupt sources and NVIC interrupt input in this system is shown in the following table:

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	-
Interrupt #11	Reserved	-	-	-

Break#12	I2C	v	v	-
Break#13	Reserved	-	-	-
Break#14	TIM0	v	v	-
Break#15	TIM1	v	v	-
Break#16	TIM2	v	v	-
Break#17	LPTIM	v	v	v
Break#18	TIM4	v	v	-
Break#19	TIM5	v	v	-
Break#20	TIM6		v	-
Break#21	PCA	v	v	-
Break#22	WDT	v	v	v
Break#23	RTC	v	v	v
Break#24	ADC	v	v	-
Break#25	Reserved	-	-	-
Break#26	VC0	v	v	v
Break#27	VC1	v	v	v
Break#28	LVD	v	v	v
Interrupt #29	Reserved	-	-	-
Break #30	EFCTRL/RAMCTRL v		v	-
Break #31	CLK_TRIM Table	v	v	v

5-2 Correspondence between external interrupt and NVIC interrupt input

## Notice:

- Since some module interrupts are multiplexed to 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

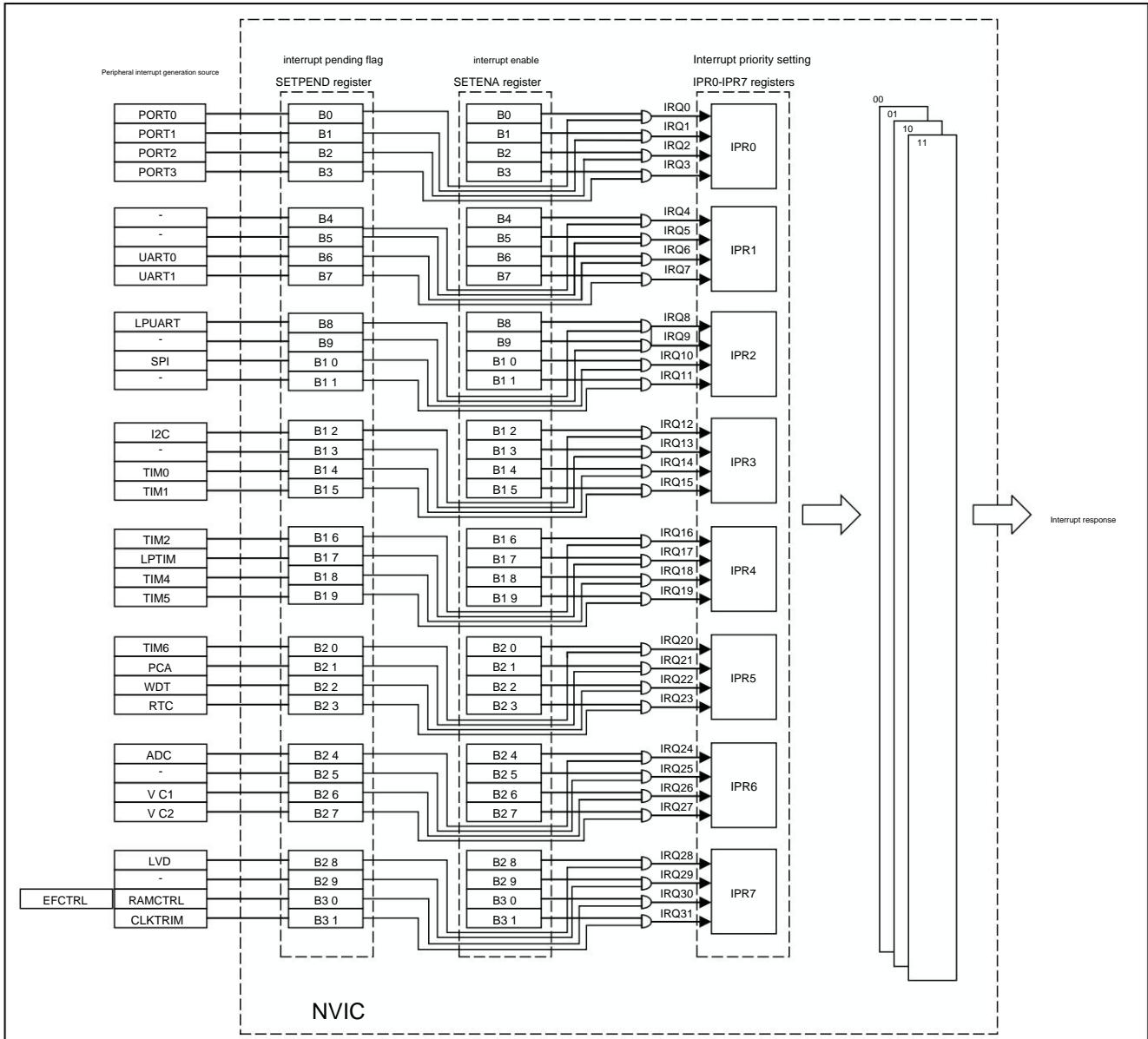


Figure 5-7 Interrupt Structure Diagram

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

ŷ The respective interrupt enable of the peripheral interrupt source is not marked in the figure, only the interrupt signal after the peripheral interrupt is generated is included here.

No. logical block diagram.

ŷ IRQ30 has 2 peripheral interrupt input multiplexing, the interrupt flag bits of these 2 peripherals must be read respectively to determine which

peripheral interrupts.

ŷ If the peripheral interrupt source has a high level, no matter 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 be corresponding to the processor and execute corresponding interrupt routine.

In the interrupt program, the high-level interrupt signal of the peripheral interrupt source must be cleared, and the interrupt pending register SCS\_SETPEND is automatically cleared by hardware.

The interrupt priority registers SCS\_IPR0- SCS\_IPR7 set the priority of 32 interrupt sources, the priority of 00 is the highest.

High, 11 has the lowest priority. When the priorities are the same, the priority is determined by the interrupt number, and the smaller the number, the higher the priority.

## 5.8 Registers

Base address: 0xE000 E000

register	offset address	describe
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 Register
SCS_IPR1	0x404	Interrupt #4 - Interrupt #7 Priority Register
SCS_IPR2	0x408	Interrupt #8 - Interrupt #11 Priority Register
SCS_IPR3	0x40C	Interrupt #12 - Interrupt #15 Priority Register
SCS_IPR4	0x410	Interrupt #16 - Interrupt #19 Priority Register
SCS_IPR5	0x414	Interrupt #20 - Interrupt #23 Priority Register
SCS_IPR6	0x418	Interrupt #24 - Interrupt #27 Priority Register
SCS_IPR7	0x41C	Interrupt #28 - Interrupt #31 Priority Register
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				20	19	18	17	16
SETENA [31:16]														
RW														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
SETENA[15:0]														
RW														

bit	mark	Function description
31:0	SETENA [31:0]	Set enable interrupt #0 to interrupt #31; write "1" to set, write "0" to 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									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	mark	describe
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 Setting Register (SCS\_SETPEND)

Offset address: 0x200

Reset value: 0x0000 0000

31	30	29	28	27	26	25									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	mark	Function description
31:0	SETPEND	Set the pending state of interrupt #0 to interrupt #31; write "1" to set, write "0" to be invalid

[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										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	mark	describe
		31:0 CLRPEND Clear the pending state 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

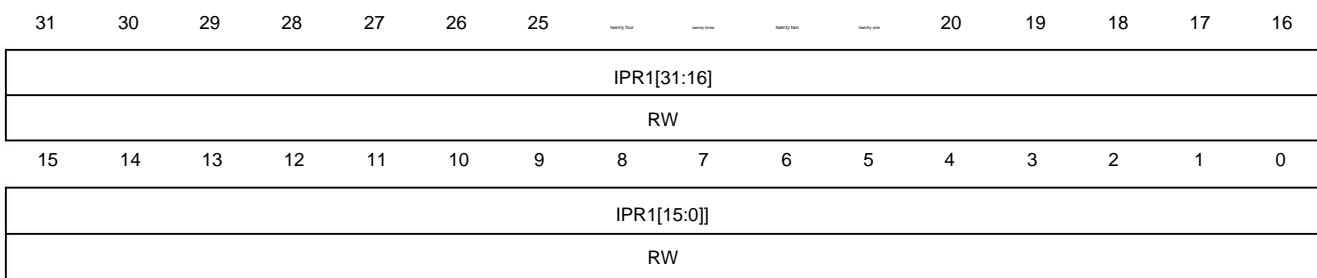
31	30	29	28	27	26	25									20	19	18	17	16
IPR0[31:16]																			
RW																			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
IPR0[15:0]]																			
RW																			

bit mark	Function 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



bit	mark	Function 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									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 mark	Function 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									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 mark	Function 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									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 mark	Function 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									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 mark	Function 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									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 mark	Function 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									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 mark	Function description
31:0	IPR7[31:0] Priority of interrupt #28 to interrupt #31; [31:30]: Priority of interrupt #31 [23:22]: Priority for 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					20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PRIM ASK
R															RW

SCS_PRIMASK															
Reset value: 0x0000_0000															
31	30	29	28	27	26	25					20	19	18	17	16
RESERVED															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															
RO															
BIT	symbol	describe													
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  This special register needs to be accessed through the MSR and MRS special register operation instructions, or can be Accessed with the change processor state instruction CPS. Actions are required when working with time-sensitive applications 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, you must first

Turn on the peripheral's own interrupt enable. The operation of this enable bit is not discussed in this chapter, please refer to the respective peripheral modules

Chapter description.

### 5.9.2 NVIC Interrupt Enable and Clear Enable

The Cortex-M0+ processor supports up to 32 interrupt sources, each with an interrupt enable bit and clear

enable bit. This has a 32-bit interrupt enable register SCS\_SETENA and a 32-bit clear enable register

SCS\_CLRENA. If you want to enable an interrupt, set the corresponding bit in the SCS\_SETENA register. Such as

To clear an interrupt, set the corresponding bit in the SCS\_CLRENA register.

Note that the interrupt enable mentioned here is only for the processor NVIC, the interrupt generation of each peripheral is the same as the

No, it is determined by the peripheral's interrupt control register, independent of 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 suspended. The pending state is stored in a

register, if the processor's current priority has not been lowered to handle the pending request, and there is no hand

automatically clears the pending state, which will remain valid.

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

Can be suspended by operating interrupt set SCS\_SETPEND and interrupt clear suspend SCS\_CLRPEND these two

register to access or modify the interrupt pending state. The interrupt pending status register allows the use of 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. interrupt priority

The programming of the level registers should be done before the interrupt is enabled, which is usually done at the beginning of the program. interrupts should be avoided

Change the interrupt priority after enabling, the results of this situation are unpredictable and not supported by the Cortex-M0+ processor

hold.



## 5.9.5 NVIC interrupt mask

Some time-sensitive applications need to disable all interrupts for a short period of time, you can use the interrupt mask to register

Register SCS\_PRIMASK is implemented. The special register SCS\_PRIMASK has only 1 bit valid and is valid at reset

Defaults to 0 after that. 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 exception are enabled. In fact, when SCS\_PRIMASK is set to

After 1, the current priority of the processor is reduced to 0 (the highest priority that can be summed).

The SCS\_PRIMASK register can be programmed in several ways, using assembly language, using the MS R instruction

After setting and clearing the SCS\_PRIMASK register. If using C language and CMSIS device driver library, user

The following functions can be used 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 Controller

This product has 16 digital general input and output ports P01-P03, P14-P15, P23-P27, P31-P36 and 1 digital

Word general-purpose input port P00. Analog signal ADC/VC/LVD input and output signals, various functional modules (such as SPI,

The input and output signals of UART, I2C, Timer, etc.) and the input and output signals of testing and debugging functions can be combined with digital

Word port multiplexing.

Each port can be configured as an internal pull-up/pull-down input, a high-impedance input (floating

input), push-pull output (CMOS output), open drain output (open drain output), two-speed drive capability output. for

When the chip is reset abnormally, the external device will produce abnormal action. After the chip is reset, the port is configured as a high-impedance input. but

In order to avoid leakage caused by high-impedance input, the user should configure the port accordingly after the chip is started (configuration  
into an internal pull-up/pull-down input or output).

After the digital port is configured as an analog port, the digital function is isolated, and digital "1" and "0" cannot be output. The CPU read end

The result of the mouth is "0".

All ports can provide external interrupts, and each interrupt can be configured as high-level trigger, low-level trigger,

There are 4 types of rising edge trigger and falling edge trigger. You can find out the corresponding interrupt flag by querying the interrupt flag bit of Px\_STAT[n].

off trigger port. In addition, the interrupt of each port can wake up the chip from sleep mode/deep sleep mode to work

model.



## 6.2 Main Features of Port Controller

The port controller supports the following features:

- ÿ Port multiplexing function
  - Analog function pin multiplexing
  - Debug pin multiplexing
  - Digital common pin multiplexing
  - Digital function pin multiplexing
- ÿ Configuration function
  - Support pull-up/pull-down
  - low drive/high drive
  - Push-pull output
  - Open drain output
- ÿ External interrupt source
  - high level/low level
  - rising/falling edge
- ÿ Support interrupt in working mode/sleep mode/deep sleep mode

## 6.3 Port Controller Function 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 the system requirements.

word port. When configured as a digital port, the corresponding registers can also be configured 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 pull-up/pull-down enable of the corresponding bit pin, when it is '0', disable the corresponding bit pin

Pull up/pull down.

#### 2. Two-speed drive output (PxDR)

The drive capability can be changed through the PxDR register, when PxDR is '1', it is low drive capability, and PxDR is '0'

when the drive capacity is high.

#### 3. Open Drain Output (PxOD)

The pin output state is set by the PxOD register. When PxOD is '1', the port open-drain output is enabled, which is

When '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 a low level.

If you need to have the function of outputting a high level at the same time, you need a pull-up resistor.

#### 4. Direction selection (PxDIR)

Used to set the direction of the port pins. When PxDIR is '0', the port is output, when PxDIR is '1'

The candidate port is an input.

#### 5. Output high and low level selection (PxOUT), which can be accessed through the AHB bus

When the port pin is configured as output, if PxOUT is '1', the port pin output is high level, if configured as

Open-drain output requires an external pull-up resistor to pull it high. If PxOUT is '0', output low level.

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

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

Low when '0'

Note: The above features are not available when configured as an analog port.



The relationship between the state of the port and the register configuration is as follows:

IO status	IO Direction Px	ADS	PxDIR	PxOUT	PxIN	PxPU	PxPD	PxOD	PxDR	Px_SEL
simulation	input Output	1	W	W	0	W	W	W	W	W
Floating	enter	0	1	W	X	0	0	W	W	0
drop down	enter	0	1	W	0	0	1	W	W	0
pull up	enter	0	1	W	1	1	0	W	W	0
pull up	enter	0	1	W	1	1	1	W	W	0
1	enter	0	1	W	1	W	W	W	W	0
0	enter	0	1	W	0	W	W	W	W	0
1	output	0	0	1	1	W	W	0	W	0
0	output	0	0	0	0	W	W	0	W	0
1	output	0	0	W	1	W	W	0	W	0
0	output	0	0	W	0	W	W	0	W	0
(SET)1/(CLR)0 output		0	0	W	(SET)1/(CLR)0W		W	0	W	0
0	output	0	0	0	0	W	W	1	W	0
Z	output	0	0	1	X	0	0	1	W	0
0	output	0	0	1	0	0	1	1	W	0
1	output	0	0	1	1	1	0	1	W	0
1	output	0	0	1	1	1	1	1	W	0

Note:

0 - Logic low

1 - Logic high

W - Whatever 0 or 1

X - unknown state

Z - high impedance

Table 6-1 Port Status Truth Table

The port circuit structure is shown in the following figure:

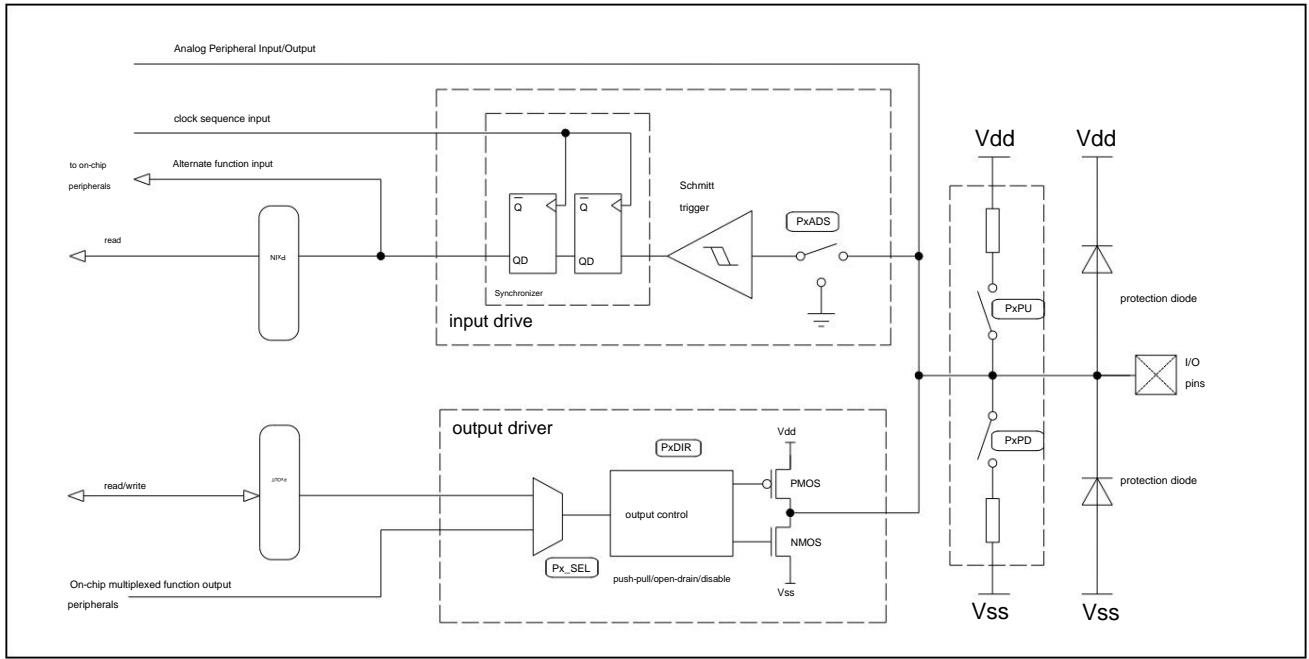


Figure 6-1 Port circuit diagram

### 6.3.2 Port Write

The port input value/output value registers (PxIN/PxOUT) support AHB bus read and write. For AHB bus, the system

The processing cycle of the system clock (HCLK) is not the same as that of other buses. Every two HCLK cycles, the IO toggles by one.

Second-rate. The following figure shows the fastest timing of AHB bus port flipping:

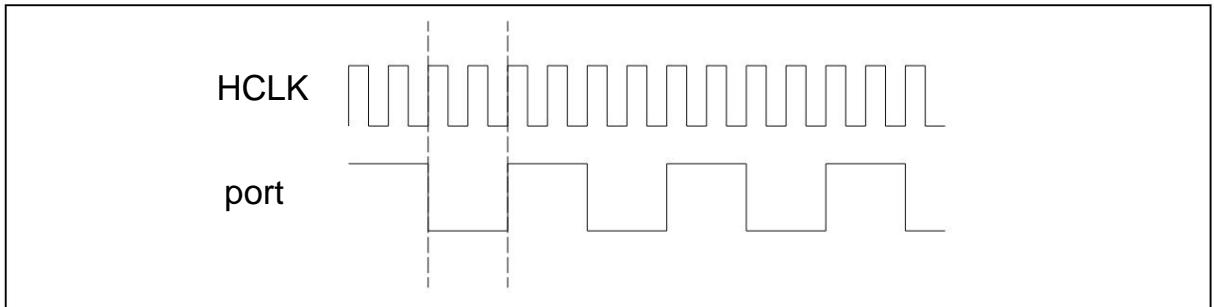


Figure 6-2 Change of AHB bus port with system clock

### 6.3.3 Port reading

Each port can obtain the port pin level by reading the PxIN register. The bits of the PxIN register are combined with

The latches in front of it form a synchronizer, thus avoiding short-term changes in the state of the system clock

Signal instability caused by pin level changes, but also introduces delay. The same as reading port pin data

The steps are as follows:

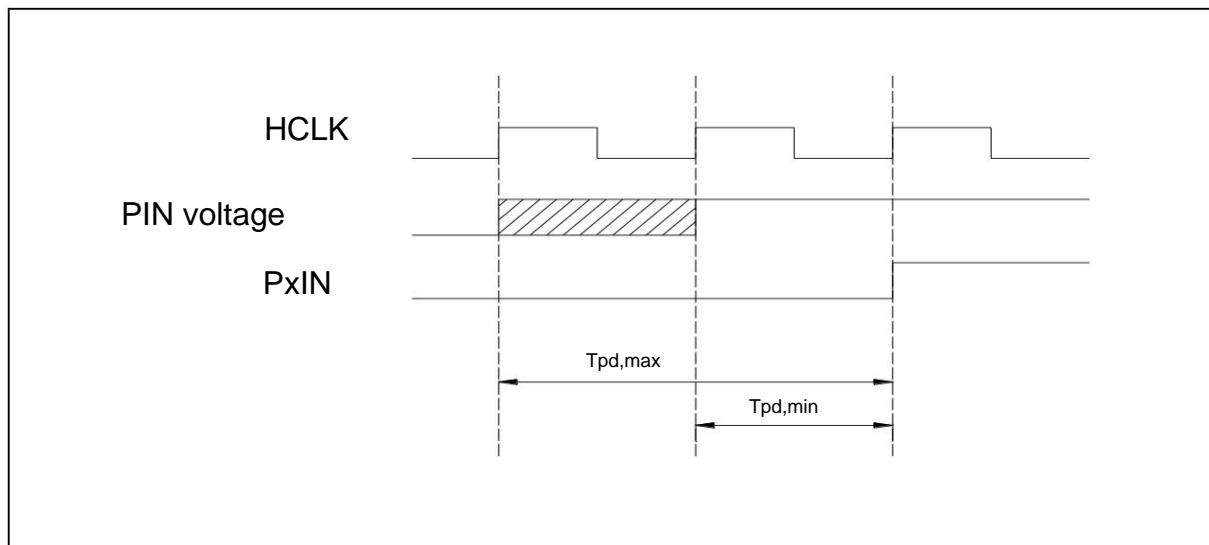


Figure 6-3 Read port pin data synchronization diagram

During the clock period after the rising edge of the system clock, the pin level signal is latched in the internal register, as shown in the shaded area

As shown, after the next rising edge of the system clock, stable pin level signals can be read. system later

Data is latched into the PxIN register on the rising edge of the clock. The signal replacement delay Tpd is 1-2 system clocks.

Notice:

- Handling of unconnected pins:

If there is a missing pin during use, in order to avoid the pin not being determined in other digital input enable modes

The level causes floating current consumption, it is recommended to give the pin a certain level.

### 6.3.4 Port multiplexing function

Port multiplexing is one of the main functions of the port controller. Through the configuration register, the port can be flexibly configured as a module analog port/test debug port/digital general-purpose 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 port, the digital function is isolated at this time, and the digital "1" and "0" cannot be output, and the result of the CPU reading the port is "0". when When PxADS is '0', the port is configured as a digital port. At this time, the digital general port is realized by configuring the Px\_sel register.

Port/digital function port switching, each port can be independently configured as the function port required by the system. test debug side

For the configuration information of the port, please refer to the relevant chapters.

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_RXD	TIM6_CHB	UART0_RXD TIM0_GATE		TIM4_CHB	SPI_MISO	I2C_SDA
AIN6 ADC_VREF	VCIN6	P36	UART1_RXD	TIM6_CHA UART0_RXD		PCA_CH4	TIM5_CHA	SPI莫斯	I2C_SCL
AIN7 XTHI	VCIN7	P01	UART0_RXD	I2C_SDA	UART1_RXD TIM0_TOG		TIM5_CHB	SPI_SCK	TIM2_EXT
AIN8 XTHO		P02	UART0_RXD	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_RXD	NC
LVDIN2	VCIN0	P23	TIM6_CHA	TIM4_CHB	TIM4_CHA	PCA_CH0	SPI_MISO	UART1_RXD VC1_OUT	
AIN0		P24	TIM4_CHB	TIM5_CHB	HCLK_OUT	PCA_CH1	SPI莫斯	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_RXD	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_RXD	RCL_OUT HCLK_OUT	
AIN2	VCIN2	P32	LPTIM_TOGN	PCA_CH2	TIM6_CHB	VC1_OUT	UART1_RXD	PCA_CH4	RTC_1Hz
AIN3	VCIN3	P33	LPUART_RXD	PCA_CH1	TIM5_CHB	PCA_ECI	UART1_RXD	XTL_OUT TIM1_TOGN	
		P00 Reset							

Table 6-2 Port multiplexing table



### 6.3.5 Port Interrupt Function

Each digital general-purpose port can be interrupted by an external signal source, which can be high level/low level/

Rising edge/falling edge 4 types of signals, the corresponding interrupt enable register is high level interrupt enable register/

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

When an interrupt is triggered, you can determine which port triggered the interrupt by querying the interrupt status register.

The corresponding interrupt status flag can be cleared by zeroing the interrupt clear register.



## 6.4 Port configuration operation

### 6.4.1 Port multiplexing operation process

Port multiplexing configured as an analog port

Step1: Set the register Px\_ADS to 1

Port multiplexing configured as digital general purpose port

- a) Set the register Px\_ADS to 0
- b) Set the register Px\_sel to 0
- c) Set the register PxDIR to 1: the port direction is input, the CPU can read the port state PxIN
- d) Set register PxDIR to 0: port direction is output
- e) Set the register PxOUT to 1: the port outputs a high level
- f) Set register PxOUT to 0: port output low level

Port multiplexing configured as digital function port

- a) Set the 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 register PxDIR (according to system requirements)

Port multiplexing 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 bit14 of register GPIO\_CTRL1 to select the output polarity of infrared signal
- e) Set register P2OUT[3] to gate infrared signal output

#### 6.4.2 Port Interrupt Operation Flow

high level interrupt

- a) Set the register Px\_ADS to 0
- b) Set the 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 and clear the interrupt status register Px\_STAT

low level interrupt

- a) Set the register Px\_ADS to 0
- b) Set the 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 and clear the interrupt status register Px\_STAT

rising edge interrupt

- a) Set the register Px\_ADS to 0
- b) Set the 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 and clear the interrupt status register Px\_STAT

falling edge interrupt

- a) Set the register Px\_ADS to 0
- b) Set the 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 and clear the interrupt status register Px\_STAT

#### 6.4.3 Port configuration operation process

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 capability

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 I/O 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 P0	HIE	RW Port	P0 high level interrupt enable configuration register
0x134 P0	LIE	RW Port	P0 low level interrupt enable configuration register
0x138 P0	RIE	RW Port	P0 rising edge interrupt enable configuration register
0x13c P0	FIE	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 P1	DIR	RW Port	P1 I/O Configuration Register
0x144 P1	N	RO Port	P1 Input Value Register
0x148 P1	OUT	RW Port	P1 output value configuration register
0x14c P1	ADS	RW Port	P1 digital-analog configuration register
0x15c P1	DR	RW Port	P1 Drive Capability Configuration Register
0x160 P1	PU	RW Port	P1 pull-up enable configuration register
0x164 P1	PD	RW Port	P1 pull-down enable configuration register
0x16c P1	OD	RW Port	P1 Open-Drain Output Configuration Register
0x170 P1	HIE	RW Port	P1 high level interrupt enable configuration register
0x174 P1	LIE	RW Port	P1 low level interrupt enable configuration register
0x178 P1	RIE	RW Port	P1 rising edge interrupt enable configuration register
0x17c P1	FIE	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 P2	DIR	RW Port	P2 I/O Configuration Register
0x184 P2	N	RO Port	P2 Input Value Register
0x188 P2	OUT	RW Port	P2 output value configuration register
0x18c P2	ADS	RW Port	P2 D/A configuration register
0x19c P2	DR	RW port	P2 drive capability configuration register
0x1a0 P2	PU	RW Port	P2 pull-up enable configuration register
0x1a4 P2	PD	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 I/O 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
0xfc	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
0x304	GPIO_CTRL1	RW Port	Auxiliary Function Configuration Register 1
0x308	GPIO_CTRL2	RW Port	Auxiliary Function Configuration Register 2
0x30c	GPIO_CTRL3	RW Port	Accessibility Configuration Register 3
0x310	GPIO_CTRL4	RW Port	Accessibility 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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reserved
P01_sel															RW	

Bit tag function 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 toggle 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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reserved																P02_sel	
Reserved																RW	

Bit tag function 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 The inverse signal of the TIM0_TOGN Timer0 module toggle signal 101: TIM6_CHA Advanced Timer module channel 2 A signal 110: SPI_CS SP I module host 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 tag	function 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 host 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 I/O 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												P0D	P0D	P0D	Res
												IR3	IR2	IR1	
												RW	RW	RW	

Bit tag function description	
31:4	Reserved
3	P0DIR3 Port P03 Input and Output Configuration Register 1: Configure as input 0: Configure as output
2	P0DIR2 Port P02 Input and Output Configuration Register 1: Configure as input 0: Configure as output
1	P0DIR1 Port P01 Input and Output Configuration Register 1: Configure as input 0: Configure as 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														P0I	P0I
														N3	N2
															N1
														RO	RO
														RO	

Bit tag function 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

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										P0O	P0O	P0O	Res	UT3	UT2	UT1
										RW	RW	RW				

Bit tag	function description	
31:4	Reserved	
3	P0OUT3 port P03 output value configuration register  1: Output high level. If 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 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 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												P0A	P0A	P0A	Res
												DS3	DS2	DS1	
												RW	RW	RW	

Bit tag function description		
31:4	Reserved	
3	P0ADS3 port	P03 digital-analog configuration register  1: Configure as an analog port  0: Configure as a digital port
2	P0ADS2 port	P02 digital-analog configuration register  1: Configure as an analog port  0: Configure as a digital port
1	P0ADS1 port	P01 digital-analog configuration register  1: Configure as an analog port  0: Configure 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												P0D	P0D	P0D	Res
												R3	R2	R1	
												RW	RW	RW	

Bit tag	function 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										P0P U3	P0P U2	P0P U1	Res		
										RW	RW	RW			

Bit tag	function description	
31:4	Reserved	
3	P0PU3 port	P03 pull-up enable configuration register  1: enable  0: Disable
2	P0PU2 port	P02 pull-up enable configuration register  1: enable  0: Disable
1	P0PU1 port	P01 pull-up enable configuration register  1: enable  0: Disable
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										P0P D3	P0P D2	P0P D1	Res		
										RW	RW	RW			

Bit tag	function description	
31:4	Reserved	
3	P0PD3 port P03 pull-down enable configuration register  1: enable  0: Disable	
2	P0PD2 port P02 pull-down enable configuration register  1: enable  0: Disable	
1	P0PD1 port P01 pull-down enable configuration register  1: enable  0: Disable	
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										P0O	P0O	P0O	Res		
										D3	D2	D1			
										RW	RW	RW			

Bit tag	function 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 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														P0H	P0H
														IE3	IE2
														IE1	IE0
														RW	RW

Bit tag	function description	
31:4	Reserved	
3	P0HIE3 port P03 high level interrupt enable configuration register 1: enable 0: Disable	
2	P0HIE2 port P02 high level interrupt enable configuration register 1: enable 0: Disable	
1	P0HIE1 port P01 high level interrupt enable configuration register 1: enable 0: Disable	
0	P0HIE0 port P00 high level interrupt enable configuration register 1: enable 0: Disable	

### 6.5.1.13 Port P0 Low Level Interrupt Enable Configuration Register (P0LIE)

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												POL	POL	POL	POL
												IE3	IE2	IE1	IE0
												RW	RW	RW	RW

Bit tag function description		
31:4	Reserved	
3	P0LIE3 port P03 low level interrupt enable configuration register  1: enable  0: Disable	
2	P0LIE2 port P02 low level interrupt enable configuration register  1: enable  0: Disable	
1	P0LIE1 port P01 low level interrupt enable configuration register  1: enable  0: Disable	
0	P0LIE0 port P00 low level interrupt enable configuration register  1: enable  0: Disable	

### 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												P0R	P0R	P0R	P0R
												IE3	IE2	IE1	IE0
												RW	RW	RW	RW

Bit tag function description																
31:4	Reserved															
3	P0RIE3 Port P03 Rising Edge Interrupt Enable Configuration Register															
		1: enable														
		0: Disable														
2	P0RIE2 Port P02 Rising Edge Interrupt Enable Configuration Register															
		1: enable														
		0: Disable														
1	P0RIE1 Port P01 Rising Edge Interrupt Enable Configuration Register															
		1: enable														
		0: Disable														
0	P0RIE0 Port P00 Rising Edge Interrupt Enable Configuration Register															
		1: enable														
		0: Disable														

### 6.5.1.15 Port P0 Falling Edge Interrupt Enable Configuration Register (P0FIE)

Offset address: 0x13C

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												P0F	P0F	P0F	P0F
												IE3	IE2	IE1	IE0
												RW	RW	RW	RW

Bit tag	function description	
31:4	Reserved	
3	P0FIE3 Port P03 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
2	P0FIE2 Port P02 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
1	P0FIE1 Port P01 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
0	P0FIE0 Port P00 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	

### 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												POS	POS	POS	POS
												TA3	TA2	TA1	TA0
												RO	RO	RO	RO

Bit tag function description		
31:4	Reserved	
3	P0STA3 Port P03 Interrupt Status Register  1: Interrupt trigger  0: No interrupt trigger	
2	P0STA2 Port P02 Interrupt Status Register  1: Interrupt trigger  0: No interrupt trigger	
1	P0STA1 Port P01 Interrupt Status Register  1: Interrupt trigger  0: No interrupt trigger	
0	P0STA0 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												P0C	P0C	P0C	P0C
												LR3	LR2	LR1	LR0
												RW	RW	RW	RW

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reserved																P14_sel	
																RW	

Bit tag function 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 Reverse signal of Timer2 module toggle signal 011: PCA_ECI PCA module external clock input signal 100: ADC_RDY ADC module RDY signal 101: SPI_CS SPI module host mode chip select signal 110: UART0_RXD UART0 module RXD 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 tag function 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 toggle signal 011: TIM4_CHB Advanced Timer Module Channel 0 B Signal 100: LPTIM_GATE Timer3 module gate 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 I/O 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								P1D	P1D	Reserved					
								IR5	IR4						
								RW	RW						

Bit tag	function description	
31:6	Reserved	
5	P1DIR5 Port P15 Input and Output Configuration Register  1: Configure as input  0: Configure as output	
4	P1DIR4 Port P14 Input and Output Configuration Register  1: Configure as input  0: Configure as output	
3:0	Reserved	

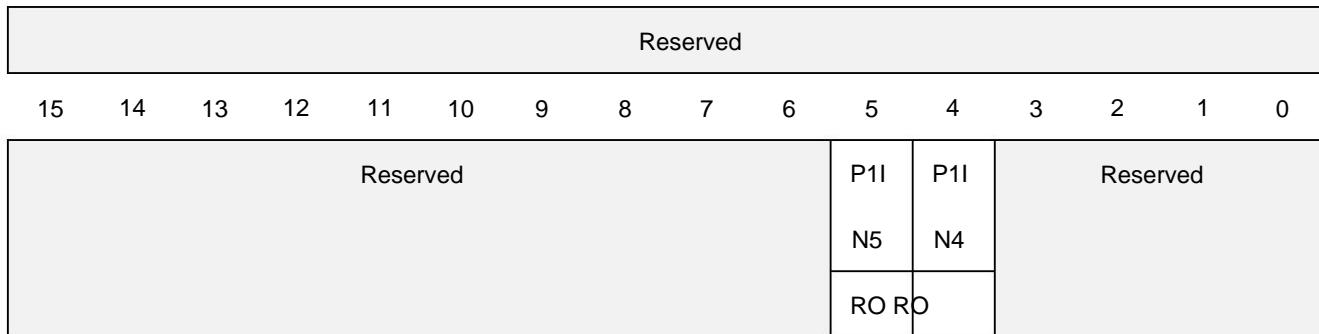
#### 6.5.2.4 Port P1 Input Value Register (P1IN)

Offset address: 0x144

Reset value: NA

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

17 16



Bit tag function 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								P1O	P1O	Reserved					
								UT5	UT4						
								RW	RW						

Bit tag	function description	
31:6	Reserved	
5	P1OUT5 port P15 output value configuration register  1: Output high level. If 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 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-to-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								P1A	P1A	Reserved					
								DS5	DS4						
								RW	RW						

Bit tag	function description	
31:6	Reserved	
5	P1ADS5 port P15 digital-analog configuration register  1: Configure as an analog port  0: Configure as a digital port	
4	P1ADS4 port P14 digital-analog configuration register  1: Configure as an analog port  0: Configure as a digital port	
3:0	Reserved	

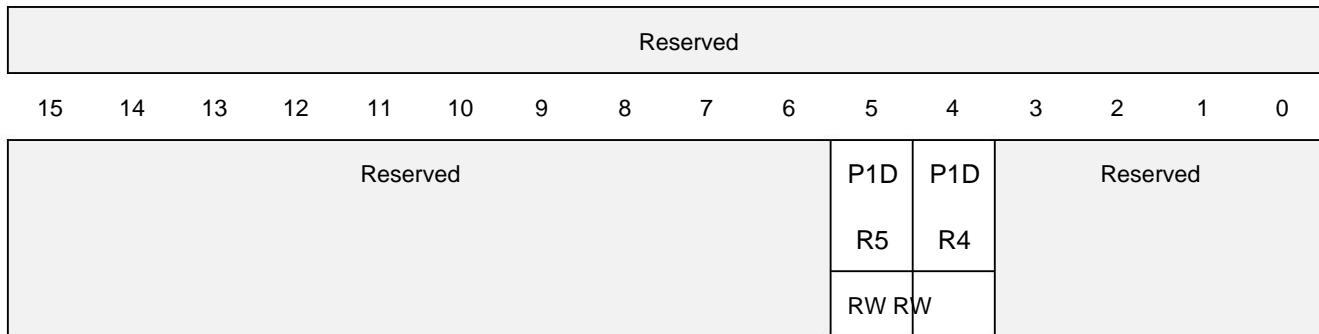
### 6.5.2.7 Port P1 Drive Capability Configuration Register (P1DR)

Offset address: 0x15C

Reset value: 0x0000 0000

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

17 16



Bit tag	function 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								P1P	P1P	Reserved					
								U5	U4						
								RW	RW						

Bit tag	function description	
31:6	Reserved	
5	P1PU5 port P15 pull-up enable configuration register  1: enable  0: Disable	
4	P1PU4 port P14 pull-up enable configuration register  1: enable  0: Disable	
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								P1P	P1P	Reserved					
								D5	D4						
								RW	RW						

Bit tag function description		
31:6	Reserved	
5	P1PD5 port P15 pull-down enable configuration register  1: enable  0: Disable	
4	P1PD4 port P14 pull-down enable configuration register  1: enable  0: Disable	
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								P1O	P1O	Reserved					
								D5	D4						
								RW	RW						

Bit tag	function 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 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								P1H	P1H	Reserved					
								IE5	IE4						
								RW	RW						

Bit tag	function description	
31:6	Reserved	
5	P1HIE5 port P15 high level interrupt enable configuration register  1: enable  0: Disable	
4	P1HIE4 port P14 high level interrupt enable configuration register  1: enable  0: Disable	
3:0	Reserved	

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

Offset address: 0x174

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								P1L	P1L	Reserved					
								IE5	IE4						
								RW	RW						

Bit tag function description		
31:6	Reserved	
5	P1LIE5 port P15 low level interrupt enable configuration register  1: enable  0: Disable	
4	P1LIE4 port P14 low level interrupt enable configuration register  1: enable  0: Disable	
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								P1R	P1R	Reserved					
								IE5	IE4						
								RW	RW						

Bit tag function description		
31:6	Reserved	
5	P1RIE5 Port P15 Rising Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
4	P1RIE4 Port P14 Rising Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
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								P1F	P1F	Reserved					
								IE5	IE4						
								RW	RW						

Bit tag	function description	
31:6	Reserved	
5	P1FIE5 Port P15 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
4	P1FIE4 Port P14 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
3:0	Reserved	

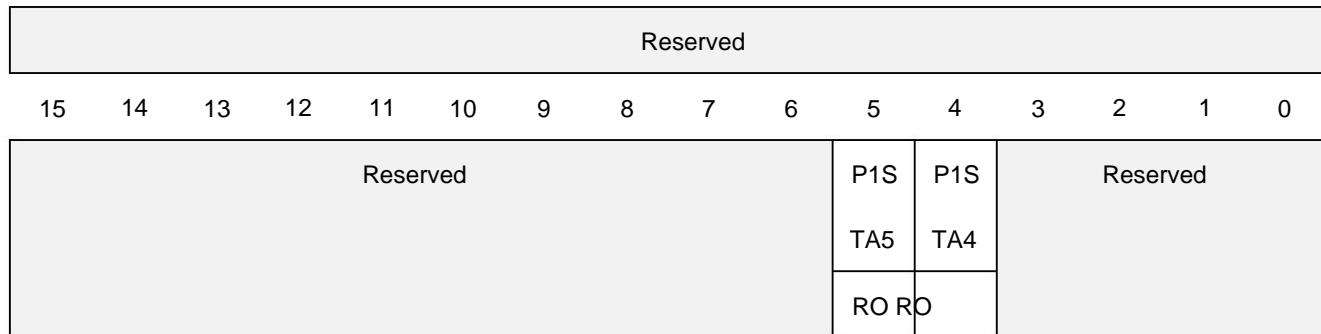
### 6.5.2.15 Port P1 Interrupt Status Register (P1\_STAT)

Offset address: 0x240

Reset value: 0x0000 0000

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

17 16



Bit tag function 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	

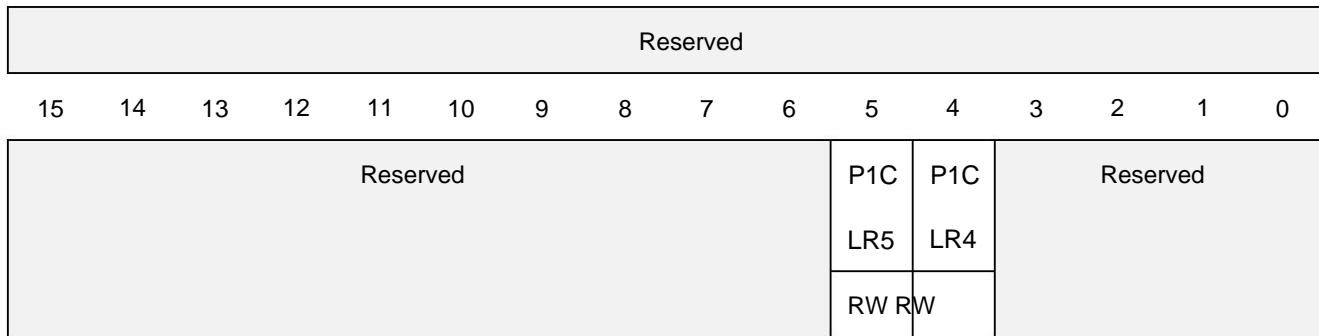
### 6.5.2.16 Port P1 Interrupt Clear Register (P1\_ICLR)

Offset address: 0x250

Reset value: 0xffff ffff

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

17 16



Bit tag function description		
31:6	Reserved	
5	P1CLR5 Port P15 Interrupt Clear Register  1: reserved interrupt flag bit  0: Clear the interrupt flag bit	
4	P1CLR4 Port P14 Interrupt Clear Register  1: reserved 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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reserved
P23_sel															RW	

Bit tag function 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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reserved
Reserved															P24_sel	
															RW	

Bit tag	function 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/Compare 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 tag function 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 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

Bit tag	function 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/Compare Signal  101: LPUART_RXD LPUART module RXD 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 tag function	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 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								P2D	P2D	P2D	P2D	P2D	P2D	Reserved	
								IR7	IR6	IR5	IR4	IR3			
								RW	RW	RW	RW	RW			

Bit tag	function description	
31:8	Reserved	
7	P2DIR7 Port P27 Input and Output Configuration Register  1: Configure as input  0: Configure as output	
6	P2DIR6 port P26 input and output configuration register  1: Configure as input  0: Configure as output	
5	P2DIR5 port P25 input and output configuration register  1: Configure as input  0: Configure as output	
4	P2DIR4 Port P24 I/O Configuration Register  1: Configure as input  0: Configure as output	
3	P2DIR3 Port P23 Input and Output Configuration Register  1: Configure as input  0: Configure as 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								P2I N7	P2I N6	P2I N5	P2I N4	P2I N3	Reserved		
RO RO RO RO RO RO															

Bit tag function 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								P2O UT7	P2O UT6	P2O UT5	P2O UT4	P2O UT3	Reserved		
RW	RW	RW	RW	RW	RW										

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



#### 6.5.3.9 Port P2 Digital-to-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								P2A	P2A	P2A	P2A	P2A	P2A	Reserved	
								DS7	DS6	DS5	DS4	DS3			
								RW	RW	RW	RW	RW			

Bit tag	function description	
31:8	Reserved	
7	P2ADS7 port P27 digital-analog configuration register  1: Configure as an analog port  0: Configure as a digital port	
6	P2ADS6 port P26 digital-analog configuration register  1: Configure as an analog port  0: Configure as a digital port	
5	P2ADS5 port P25 digital-analog configuration register  1: Configure as an analog port  0: Configure as a digital port	
4	P2ADS4 port P24 digital-analog configuration register  1: Configure as an analog port  0: Configure as a digital port	
3	P2ADS3 port P23 digital-analog configuration register  1: Configure as an analog port  0: Configure 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								P2D	P2D	P2D	P2D	P2D	Reserved		
R7 RW								R6	R5	R4	R3				

Bit tag	function 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								P2P	P2P	P2P	P2P	P2P	Reserved		
								U7	U6	U5	U4	U3			
								RW	RW	RW	RW	RW			

Bit tag function description																
31:8	Reserved															
7	P2PU7 port	P27 pull-up enable configuration register														
		1: enable														
		0: Disable														
6	P2PU6 port	P26 pull-up enable configuration register														
		1: enable														
		0: Disable														
5	P2PU5 port	P25 pull-up enable configuration register														
		1: enable														
		0: Disable														
4	P2PU4 port	P24 pull-up enable configuration register														
		1: enable														
		0: Disable														
3	P2PU3 port	P23 pull-up enable configuration register														
		1: enable														
		0: Disable														
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								P2P	P2P	P2P	P2P	P2P	Reserved		
D7	D6	D5	D4	D3				RW	RW	RW	RW	RW			

Bit tag	function description	
31:8	Reserved	
7	P2PD7 port P27 pull-down enable configuration register  1: enable  0: Disable	
6	P2PD6 port P26 pull-down enable configuration register  1: enable  0: Disable	
5	P2PD5 port P25 pull-down enable configuration register  1: enable  0: Disable	
4	P2PD4 port P24 pull-down enable configuration register  1: enable  0: Disable	
3	P2PD3 port P23 pull-down enable configuration register  1: enable  0: Disable	
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								P2O	P2O	P2O	P2O	P2O	P2O	Reserved	
								D7	D6	D5	D4	D3			
								RW	RW	RW	RW	RW			

Bit tag	function 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 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								P2H	P2H	P2H	P2H	P2H	P2H	Reserved	
								IE7	IE6	IE5	IE4	IE3			
								RW	RW	RW	RW	RW			

Bit tag function description																
31:8	Reserved															
7	P2HIE7 port P27 high level interrupt enable configuration register															
		1: enable														
		0: Disable														
6	P2HIE6 port P26 high level interrupt enable configuration register															
		1: enable														
		0: Disable														
5	P2HIE5 port P25 high level interrupt enable configuration register															
		1: enable														
		0: Disable														
4	P2HIE4 port P24 high level interrupt enable configuration register															
		1: enable														
		0: Disable														
3	P2HIE3 port P23 high level interrupt enable configuration register															
		1: enable														
		0: Disable														
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								P2L	P2L	P2L	P2L	P2L	Reserved		
								IE7	IE6	IE5	IE4	IE3			
								RW	RW	RW	RW	RW			

Bit tag	function description	
31:8	Reserved	
7	P2LIE7 port P27 low level interrupt enable configuration register  1: enable  0: Disable	
6	P2LIE6 port P26 low level interrupt enable configuration register  1: enable  0: Disable	
5	P2LIE5 port P25 low level interrupt enable configuration register  1: enable  0: Disable	
4	P2LIE4 port P24 low level interrupt enable configuration register  1: enable  0: Disable	
3	P2LIE3 port P23 low level interrupt enable configuration register  1: enable  0: Disable	
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								P2R	P2R	P2R	P2R	P2R	Reserved		
IE7 RW								IE6 RW	IE5 RW	IE4 RW	IE3 RW				

Bit tag	function description	
31:8	Reserved	
7	P2RIE7 Port P27 Rising Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
6	P2RIE6 Port P26 Rising Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
5	P2RIE5 Port P25 Rising Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
4	P2RIE4 Port P24 Rising Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
3	P2RIE3 Port P23 Rising Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
2:0	Reserved	

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

Offset address: 0x1BC

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								P2F	P2F	P2F	P2F	P2F	Reserved		
								IE7	IE6	IE5	IE4	IE3			
								RW	RW	RW	RW	RW			

Bit tag	function description	
31:8	Reserved	
7	P2FIE7 Port P27 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
6	P2FIE6 Port P26 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
5	P2FIE5 Port P25 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
4	P2FIE4 Port P24 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
3	P2FIE3 Port P23 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
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								P2S	P2S	P2S	P2S	P2S	P2S	Reserved	
								TA7	TA6	TA5	TA4	TA3			
								RO	RO	RO	RO	RO			

Bit tag function 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								P2C	P2C	P2C	P2C	P2C	P2C	Reserved			
								LR7	LR6	LR5	LR4	LR3					
								RW	RW	RW	RW	RW					

Bit tag function description																
31:8	Reserved															
7	P2CLR7 Port P27 Interrupt Clear Register															
		1: reserved interrupt flag bit														
		0: Clear the interrupt flag bit														
6	P2CLR6 Port P26 Interrupt Clear Register															
		1: reserved interrupt flag bit														
		0: Clear the interrupt flag bit														
5	P2CLR5 Port P25 Interrupt Clear Register															
		1: reserved interrupt flag bit														
		0: Clear the interrupt flag bit														
4	P2CLR4 Port P24 Interrupt Clear Register															
		1: reserved interrupt flag bit														
		0: Clear the interrupt flag bit														
3	P2CLR3 Port P23 Interrupt Clear Register															
		1: reserved 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	P31_sel	RW



#### 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 tag	function description		
31:3	Reserved		
2:0	P32_sel Port	P32 function selection.	
		000: GPIO P32	
		001: Inverse signal of LPTIM_TOGN Timer3 module toggle signal	
		010: PCA_CH2	PCA Module Channel 2 Capture/Compare 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/Compare 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

Bit tag	function 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/Compare 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                            Inverse signal of Timer1 module toggle 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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reserved																P34_sel	
Reserved																RW	

Bit tag function 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_RXD LPUART module RXD 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 tag	function 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 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 tag	function 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 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								P3D	Res						
IR6 IR5 IR4 IR3 IR2 IR1								RW	RW	RW	RW	RW			

Bit tag	function description	
31:7	Reserved	
6	P3DIR6 port	P36 I/O configuration register 1: Configure as input 0: Configure as output
5	P3DIR5 port	P35 input and output configuration register 1: Configure as input 0: Configure as output
4	P3DIR4 Port	P34 I/O Configuration Register 1: Configure as input 0: Configure as output
3	P3DIR3 Port	P33 Input and Output Configuration Register 1: Configure as input 0: Configure as output
2	P3DIR2 port	P32 I/O configuration register 1: Configure as input 0: Configure as output
1	P3DIR1 port	P31 input and output configuration register



		1: Configure as input 0: Configure as 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								P3I	Res						
N6 N5 N4 N3 N2 N1								RO							
RO RO RO RO RO RO															

Bit tag	function 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								P3O	Res						
UT6 UT5 UT4 UT3 UT2 UT1								RW	RW	RW	RW	RW	RW		

Bit tag function description															
31:7	Reserved														
6	P3OUT6 port P36 output value configuration register														
		1: Output high level. If 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 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 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 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 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 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-to-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								P3A	Res						
DS6 DS5 DS4 DS3 DS2 DS1								RW	RW	RW	RW	RW	RW		

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

		1: Configure as an analog port  0: Configure as a digital port
0	Reserved	



#### 6.5.4.11 Port P3 Drive Capability Configuration Register (P3DR)

Offset address: 0x1DC

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								P3D	Res						
								R6	R5	R4	R3	R2	R1		
								RW	RW	RW	RW	RW	RW		

Bit tag	function 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								P3P	Res						
U6 U5 U4 U3 U2 U1								RW	RW	RW	RW	RW			

Bit tag	function description	
31:7	Reserved	
6	P3PU6 port P36 pull-up enable configuration register  1: enable  0: Disable	
5	P3PU5 port P35 pull-up enable configuration register  1: enable  0: Disable	
4	P3PU4 port P34 pull-up enable configuration register  1: enable  0: Disable	
3	P3PU3 port P33 pull-up enable configuration register  1: enable  0: Disable	
2	P3PU2 port P32 pull-up enable configuration register  1: enable  0: Disable	
1	P3PU1 port P31 pull-up enable configuration register	



		1: enable 0: Disable
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								P3P	Res						
								D6	D5	D4	D3	D2	D1		
								RW	RW	RW	RW	RW			

Bit tag function description
31:7 Reserved
6 P3PD6 port P36 pull-down enable configuration register 1: enable 0: Disable
5 P3PD5 port P35 pull-down enable configuration register 1: enable 0: Disable
4 P3PD4 port P34 pull-down enable configuration register 1: enable 0: Disable
3 P3PD3 port P33 pull-down enable configuration register 1: enable 0: Disable
2 P3PD2 port P32 pull-down enable configuration register 1: enable 0: Disable
1 P3PD1 port P31 pull-down enable configuration register



		1: enable 0: Disable
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								P3O	Res						
								D6	D5	D4	D3	D2	D1		
								RW	RW	RW	RW	RW	RW		

Bit tag	function 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 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								P3H	Res						
								IE6	IE5	IE4	IE3	IE2	IE1		
								RW	RW	RW	RW	RW	RW		

Bit tag function description															
31:7	Reserved														
6	P3HIE6 port	P36 high level interrupt enable configuration register													
		1: enable													
		0: Disable													
5	P3HIE5 port	P35 high level interrupt enable configuration register													
		1: enable													
		0: Disable													
4	P3HIE4 port	P34 high level interrupt enable configuration register													
		1: enable													
		0: Disable													
3	P3HIE3 port	P33 high level interrupt enable configuration register													
		1: enable													
		0: Disable													
2	P3HIE2 port	P32 high level interrupt enable configuration register													
		1: enable													
		0: Disable													
1	P3HIE1 port	P31 high level interrupt enable configuration register													



		1: enable 0: Disable
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								P3L	Res						
IE6 IE5 IE4 IE3 IE2 IE1								RW	RW	RW	RW	RW			

Bit tag function description																
31:7	Reserved															
6	P3LIE6 port P36 low level interrupt enable configuration register															
	1: enable															
	0: Disable															
5	P3LIE5 port P35 low level interrupt enable configuration register															
	1: enable															
	0: Disable															
4	P3LIE4 port P34 low level interrupt enable configuration register															
	1: enable															
	0: Disable															
3	P3LIE3 port P33 low level interrupt enable configuration register															
	1: enable															
	0: Disable															
2	P3LIE2 port P32 low level interrupt enable configuration register															
	1: enable															
	0: Disable															
1	P3LIE1 port P31 low level interrupt enable configuration register															



		1: enable 0: Disable
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								P3R	Res						
								IE6	IE5	IE4	IE3	IE2	IE1		
								RW	RW	RW	RW	RW	RW		

Bit tag	function description	
31:7	Reserved	
6	P3RIE6 Port P36 Rising Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
5	P3RIE5 port P35 rising edge interrupt enable configuration register  1: enable  0: Disable	
4	P3RIE4 Port P34 Rising Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
3	P3RIE3 Port P33 Rising Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
2	P3RIE2 Port P32 Rising Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
1	P3RIE1 Port P31 Rising Edge Interrupt Enable Configuration Register	



		1: enable 0: Disable
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								P3F	Res						
								IE6	IE5	IE4	IE3	IE2	IE1		
								RW	RW	RW	RW	RW			

Bit tag	function description	
31:7	Reserved	
6	P3FIE6 Port P36 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
5	P3FIE5 Port P35 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
4	P3FIE4 Port P34 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
3	P3FIE3 Port P33 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
2	P3FIE2 Port P32 Falling Edge Interrupt Enable Configuration Register  1: enable  0: Disable	
1	P3FIE1 Port P31 Falling Edge Interrupt Enable Configuration Register	



		1: enable 0: Disable
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								P3S	Res						
TA6 TA5 TA4 TA3 TA2 TA1								RO	RO	RO	RO	RO	RO		
RO RO RO RO RO RO															

Bit tag	function 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								P3C	Res						
								LR6	LR5	LR4	LR3	LR2	LR1		
								RW	RW	RW	RW	RW	RW		

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

		1: reserved interrupt flag bit 0: Clear the interrupt flag bit
0	Reserved	

### 6.5.5 Port Auxiliary Functions

#### 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_p ol		hclk _en	pclk _en	hclk_sel	pclk_sel				ssn_sel				ext_clk_sel		
	RW	RW	RW	RW			RW		RW				RW		

Bit tag	function description	
31:15	Reserved	
14	ir_pol	IR output polarity selection.  0 – positive output  1 – Inverted output
13	hclk_en	hclk output control.  0 – output 0  1 - output hclk
12	pclk_en	pclk output gating.  0 – output 0  1 – output pclk
11:10	hclk_sel	hclk output divider 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	pca_cap3	pca_cap2	pca_cap1	pca_cap0					
						_sel	_sel	_sel	_sel	_sel					
						RW	RW	RW	RW	RW					

bit mark		Function 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 tag	function 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 mark		Function 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 includes a 32kByte FLASH memory, which is divided into 64 Sectors, each Sector

The capacity is 512Byte. This module supports erase, program and read operations of this memory. In addition, this module supports

It supports the protection of FLASH memory erasing and writing, and the write protection of control registers.

### 7.2 Structure block diagram

Address range	Sector number		Address range	Sector 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 FLASH reading and writing in three bit-width modes: Byte (8bits), Half-word (16bits), Word (32bits). Note that the address of Byte operation must be aligned by Byte, and the target address of Half-word operation must be Aligned by Half-word (the lowest bit of the address is 0), the address of Word operation must be aligned by Word (address The lowest two bits are 0). If the address of the read and write operations is not aligned as above, the system will enter Hard Fault Interrupt on error.

#### 7.3.1 Page 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 are all 0xFF. If the erase operation is performed from FLASH, the CPU will stop fetching and the hardware will automatically Wait for the operation to complete (FLASH\_CR.BUSY becomes 0); if the erase operation is performed from RAM, then The CPU will not stop the fetch, user software should wait for the operation to complete (FLASH\_CR.BUSY becomes 0).

Page (Sector) erase operation steps are as follows:

**Step1: Configure** the FLASH erasing and writing parameters, see the erasing and writing sequence chapter for details.

**Step2:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn to enable register rewriting.

**Step3: Configure** FLASH\_CR.OP to 2, 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 turn to enable register rewriting.

**Step6:** Set the corresponding bit of FLASH\_SLOCK to 1 to remove the erasure protection of the sector.

**Step7: Check** whether the corresponding bit of FLASH\_SLOCK is 1, if not, jump to Step5.

Step8: Write arbitrary data to any address in the sector to be erased, triggering erasing of the sector.

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 Chip Erase

Chip Erase can erase all pages (Sectors) at once. After the erase operation is completed, all pages (Sectors) in



The data are all 0xFF. If the erase operation is performed from FLASH, the CPU will stop fetching and the hardware will automatically

Wait for the operation to complete (FLASH\_CR.BUSY becomes 0); if the erase operation is performed from RAM, then

The CPU will not stop the fetch, 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 the FLASH erasing and writing parameters, see the erasing and writing sequence chapter for details.

**Step2:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn to enable register rewriting.

**Step3: Configure** FLASH\_CR.OP to 3, 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 turn to enable register rewriting.

**Step6:** Set FLASH\_SLOCK to 0xFFFF, and remove the erase and write protection of all sectors.

**Step7: Check** whether FLASH\_SLOCK is 0xFFFF, if not, jump to Step5.

Step8: Write any address in the chip to be erased to trigger chip erasing.

example:  $\ast ((\text{unsigned char} *) 0x00000000) = 0x00 .$

**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 write the bit data in the FLASH from 1 to 0, so before writing data, make sure that the address to be written is within the

The data is 0xFF. Supports 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 performed from RAM, the CPU will not stop fetching and the user software should wait for the

The operation is complete (FLASH\_CR.BUSY becomes 0).

Byte write operation steps are as follows:

**Step1:** Configure the FLASH erasing and writing parameters, see the erasing and writing sequence chapter for details.

**Step2:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn 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 turn to enable register rewriting.

**Step6:** Set the corresponding bit of FLASH\_SLOCK to 1 to remove the erase and write protection.

**Step7: Check** whether the corresponding bit of FLASH\_SLOCK is 1, if not, jump to Step5.

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

example:  $\text{* } ((\text{ unsigned char } *) \text{ 0x00001231}) = 0x5A .$

**Step9: Wait for** FLASH\_CR. BUSY to become 0, and the write operation is completed.

**Step10: To write** Byte to other addresses that have been erased and erased, repeat Step8 – Step9.

The half-word write operation steps are as follows:

**Step1: Configure** the FLASH erasing time, see the erasing sequence chapter for details.

**Step2:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn 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 turn to enable register rewriting.

**Step6:** Set the corresponding bit of FLASH\_SLOCK to 1 to remove the erase and write 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:  $\text{* } ((\text{ unsigned short } *) \text{ 0x00001232}) = 0xABCD .$

**Step9: Wait for** FLASH\_CR. BUSY to become 0, and the write operation is completed.

**Step10: To write** the Half-word to other addresses that have been erased and protected, repeat Step8 – Step9.

The steps for writing in Word are as follows:

**Step1: Configure** the FLASH erasing and writing parameters, see the erasing and writing sequence chapter for details.

**Step2:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn 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 turn to enable register rewriting.

**Step6:** Set the corresponding bit of FLASH\_SLOCK to 1 to remove the erase and write protection.

**Step7: Check** whether the corresponding bit of FLASH\_SLOCK is 1, if not, jump to Step5



Step8: Perform word write operation on the target address to be written, and trigger the write operation.

example:  $\ast \text{(( unsigned long *) } 0x00001234) = 0x55667788.$

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

Step10: To write Word to other addresses that have been erased and erased, repeat Step8 – Step9.

#### 7.3.4 Read Operation

Support to read out 3 data lengths: Byte (8bits), Half-word (16bits), Word (32bits), read data

It is little-endian mode, that is, the low address stores the low byte of the data. The read operation requires no operation steps, and can be read at any time data in FLASH.

Example of Byte read operation: `temp = *((unsigned char *)0x00001231)`

Half-word read operation example `temp = * ((unsigned int *) 0x00001232)`

Word read operation example `temp = * ((unsigned long *) 0x00001234)`

## 7.4 Erase and write timing

FLASH memory has strict time requirements for the control signals of erase and programming operations, and the timing of control signals is not

Passing will cause the Erase and Program operations to fail. The erase and write parameters when HCLK is 4MHz are loaded by default when powered on.

If the HCLK frequency is not 4MHz when the Flash is erased and written, the user program should load the HCLK frequency

The corresponding erasing parameters. When operating on FLASH, the frequency range of HCLK is required to be 1MHz ~ 32MHz.

The registers related to erasing and writing timing parameters are: FLASH\_TNVS, FLASH\_TPGS, FLASH\_TPROG,

FLASH\_TSERASE , FLASH\_TIMERASE , FLASH\_TPRCV , FLASH\_TSRCV ,

FLASH\_TMRCV. If HCLK is raised from the default 4MHz to 8MHz, the above FLASH\_Tx register

The value of the timer should be set to 2 times the default value, that is to keep the current  $T_{sysclk} \times FLASH\_Tx$  result equal to the default value

That's it.

The following table lists the corresponding FLASH erasing and writing timing 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_TIMERASE 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

Table 7-1 FLASH erasing time parameters at different frequencies

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

**Step1:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn to enable register rewriting.

**Step2:** Write 0x40 to the FLASH\_TNVS register, if the read value of the register is not 0x40, jump to

Previous.

**Step3:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn to enable register rewriting.

**Step4:** Write 0x2E to the FLASH\_TPGS register, if the read value of the register is not 0x2E, jump to

Previous.



**Step5:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn to enable register rewriting.

**Step6:** Write 0x36 to the FLASH\_TPROG register, if the read value of the register is not 0x36, jump

to the previous step.

**Step7:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn to enable register rewriting.

**Step8:** Write 0x8CA0 to the FLASH\_TSERASE register, if the read value of the register is not 0x8CA0,

then jump to the previous step.

**Step9:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn to enable register rewriting.

Step10: Write 0x445C0 to the FLASH\_TMERASE register, if the read value of the register is not 0x445C0,

then jump to the previous step.

**Step11:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn to enable the register rewriting.

**Step12:** Write 0x30 to the FLASH\_TPRCV register, if the read value of the register is not 0x30, jump

to the previous step.

**Step13:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn to enable the register rewriting.

**Step14:** Write 0x1E0 to the FLASH\_TSRCV register, if the read value of the register is not 0x1E0, jump to

Go to previous step.

**Step15:** Write 0x5A5A and 0xA5A5 to the FLASH\_BYPASS register in turn to enable register rewriting.

Step16 : Write 0x7D0 to the FLASH\_TMRCV register, if the read value of the register is not 0x7D0, then

Jump to the previous step.



## 7.5 Read Wait Cycles

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, setting FLASH\_CR.WAIT to 1. When inserting wait cycles,

The CPU only reads the instruction code in the FLASH memory once every two cycles.

## 7.6 Erase and write protection

### 7.6.1 Erase and write protection bits

The entire 32kByte FLASH memory is divided into 64 sectors, each 4 sectors share an erase and write protection

bit. When the sector is protected, the erase and write operations on the sector are invalid, and an alarm flag and an interrupt signal are generated.

No. When any Sector in the FLASH memory is protected, the chip erasing and writing of the FLASH is invalid.

And generate alarm flag and interrupt signal.

### 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 and written,

inside, then the erase and write operation is invalid and generates an alarm flag and an interrupt signal. .



## 7.7 Register Write Protection

The important controller of this module shields the ordinary write operation and must be modified by the write sequence.

The registers that need to be changed by a write sequence 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 of modifying 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.

Notice:

– Write 0x5a5a and write 0xa5a5 No write operations can be inserted between these two steps, and cannot be interrupted by interrupts,

Otherwise, the Bypass sequence will be invalid, and the sequence 0x5a5a-0xa5a5 needs to be rewritten.

## 7.8 Registers

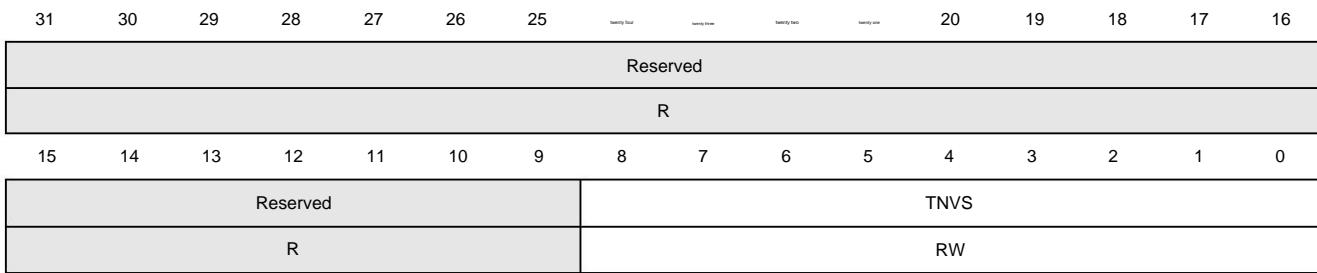
Base address: 0x4002 0000

register	offset address	describe
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 parameter
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



bit mark		Function description
31:9	Reserved	
8:0	TNVS calculation formula: TNVS = 8*HCLK, the unit of HCLK is MHz. For the method of modifying the value of this register, see 7.7. 4MHz example: TNVS = 8*4 = 32.	

## 7.8.2 TPGS parameter register (FLASH\_TPGS)

Offset address: 0x04

Reset value: 0x0000 0017

31	30	29	28	27	26	25				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 mark		Function description
31:8	RESERVED	
7:0	TPGS calculation formula: TPGS = 5.75*HCLK, the unit of HCLK is MHz. For the method of modifying the value of this register, see 7.7. 4MHz example: TPGS = 5.75*4 = 23.	

## 7.8.3 TPROG parameter register (FLASH\_TPROG)

Offset address: 0x08

Reset value: 0x0000 001B

31	30	29	28	27	26	25				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 mark		Function description
31:8	Reserved	
7:0	TPROG calculation formula: TPROG = 6.75*HCLK, the unit of HCLK is MHz. For the method of modifying the value of this register, see 7.7. 4MHz example: TPROG = 6.75*4 = 27.	

### 7.8.4 TSERASE register (FLASH\_TSERASE)

Offset address: 0x0C

Reset value: 0x0000 4650

31	30	29	28	27	26	25					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	mark	Function description
31:18	Reserved	
17:0	TSERASE calculation formula: TSERASE = 4500*HCLK, the unit of HCLK is MHz. For the method of modifying 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					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	mark	Function description
31:21	Reserved	
20:0	TMERASE calculation formula: TMERASE = 35000*HCLK, the unit of HCLK is MHz. For the method of modifying 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					20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPRCV															
RW															

bit	mark	Function description
31:12	Reserved	
11:0	TPRCV	calculation formula: TPRCV = 6*HCLK, the unit of HCLK is MHz. For the method of modifying 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					20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSRCV															
RW															

bit	mark	Function description
31:12	Reserved	
11:0	TSRCV	calculation formula: TSRCV = 60*HCLK, the unit of HCLK is MHz. For the method of modifying the value of this register, see 7.7. 4MHz example: TSRCV = 60*4 = 240.



### 7.8.8 TMRCV parameter register (FLASH\_TMRDV)

Offset address: 0x1C

Reset value: 0x0000 03E8

31	30	29	28	27	26	25					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	mark	Function description
31:13	RESERVED	
12:0	TMRCV calculation formula: TMRCV = 250*HCLK, the unit of HCLK is MHz. For the method of modifying the value of this register, see 7.7. 4MHz example: TMRCV = 250*4 = 1000.	

### 7.8.9 CR Register (FLASH\_CR)

Offset address: 0x20

Reset value: 0x0000 0200

31	30	29	28	27	26	25					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	mark	Function description
31:7	Reserved	
6:5	IE	IE[6]: FLASH erase and write protected address interrupt enable; 0: disable; 1: enable IE[5]: FLASH erasing and writing PC value interrupt enable; 0: disable; 1: enable
4	BUSY	Idle/busy flag bit; 0: idle state; 1: busy state;
3	Reserved	
2	WAIT	Read FLASH wait period; 0: 0 wait period; 1: 1 wait period
1:0	OP	FLASH operation; 00: read; 01: program; 10: sector erase; 11: chip erase

For the method of modifying 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					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	mark	describe
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					20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								ICLR 1	ICLR 0						
R								RW0	RW0						

bit	mark	Function 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 is invalid;	
0	ICLR0 Clear the PC address alarm interrupt flag; write 0 to clear; write 1 is invalid;	



## 7.8.12 BYPASS register (FLASH\_BYPASS)

Offset address: 0x2C

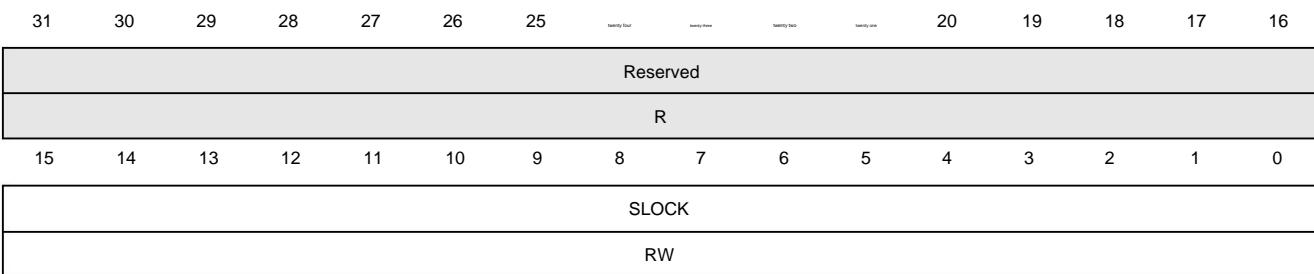
Reset value: 0x0000 0000

bit	mark	describe
31:16	Reserved	
15:0	BYSEQ	<p>Before modifying the register of this module, the sequence of 0x5a5a-0xa5a5 must be written to the BYSEQ[15:0] register. every time you write this sequence, the register can only be modified once.</p> <p>After 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.</p> <p>List. See 7.7 for details.</p>

## 7.8.13 SLOCK register (FLASH\_SLOCK)

Offset address: 0x30

Reset value: 0x0000 0000



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



## 8 RAM Controller (RAM)

### 8.1 Overview

This product contains a static SRAM with a capacity of 2K/4K bytes, which supports 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 cycles. also, The controller also supports parity check, which can perform parity check on each byte of SRAM data and generate parity Checkout error interrupt.

### 8.2 Functional Description

Read and write bit width

The controller supports Byte (8bits), Half-word (16bits), Word (32bits) three bit width read and write operations.

The address of Byte operation must be aligned by Byte, and the target address of Half-word operation must be aligned by Half-word

(The lowest bit of the address is 0), the address of Word operation must be aligned with 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 checking of SRAM data. After reset, this feature is disabled by default. When parity is turned on

After the function, when writing data to the SRAM, do parity check on the data of each Byte, and add the 1bit check value to the

8bits data are stored in SRAM together. When reading data to SRAM, the controller will read 8bits data and

1bit check value, and do parity check, if the check is wrong, set the parity check error flag bit, when the interrupt is enabled

In this case, an error interrupt will be generated.

Notice:

- When the parity check is enabled, the SRAM must be initialized before reading the SRAM data, otherwise it may accidentally touch

Send parity alarm flag or interrupt.

## 8.3 Registers

Base address: 0x4002 0400

register	offset address	describe
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					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 CHK	
R														EN	
RW RW															

bit mark		Function 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: Disable 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									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	mark	Function 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									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	mark	Function 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								20	19	18	17	16
Reserved																		
R																		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reserved																		
R																		

bit mark		Function description
31:1	Reserved	
0	ERRCLR	Error interrupt flag clear bit; write 1: invalid, write 0: clear

## 9 basic timers (TIM0/1/2)

### 9.1 Introduction to Basic Timers

The basic timer includes three timers Timer0/1/2. Timer0/1/2 function exactly the same. Timer0/1/2 are synchronous Timer/Counter, which can be used as a 16-bit timer/counter with auto-reload function, or as a 32-bit timer/counter without reload function timer/counter. Timer0/1/2 can count external pulses or realize system timing.

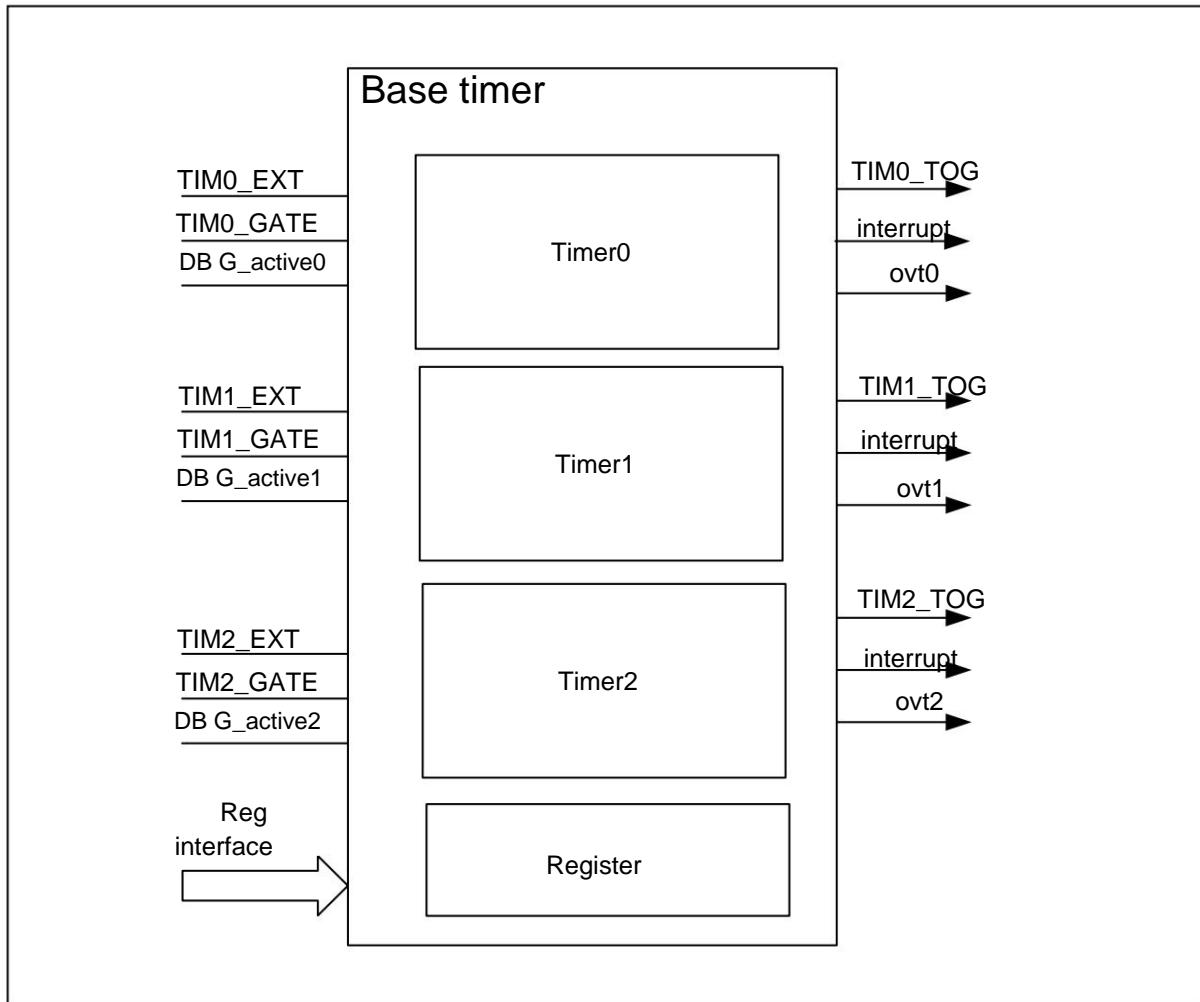


Figure 9-1 Base Timer block diagram

## 9.2 Base Timer function description

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

Timer0/1/2 use EXT, GATE for counting function, EXT is used for the external input clock signal of the counter,

GATE is used for active level count enable signal. When the gate control function is enabled, if and only when the external input GATE power

The counter will count only when it is valid, otherwise the counter is in the hold state. Gate enable is controlled using CR.gate.

By default the gate function is off. The gate level selection is controlled using CR.GATE\_P. The default high level is the gated effective level;

When set to 1, the gated low level is the active level.

TIM0/1/2 use PCLK, GATE for timing function, PCLK is used for timer's internal input clock signal

number, GATE can be used for active level timing enable signal. When the gate control function is enabled, if and only when the external input GATE

When the level is valid, the timer will count, otherwise the timer is in the timer counter stop state. gating enable use

CR.Gate control. By default the gate function is off. Gate level selection is controlled using CR.Gate\_P. The default high level is

Gated active level; when set to 1, the gated active level is low. The timing function can be configured with prescaler. CR.PRS

Controls the divider ratio.

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

The timer/counter of TIM0/1/2 supports two working modes. By setting the MD in the timer control register (CR)

Select work mode. Mode 1 is 32-bit free counting mode. Mode 2 is the 16-bit reload mode.

32-bit free counting mode, after counting to the maximum 0xFFFFFFFF, an interrupt is generated after overflow, and the timer/counter becomes

0X00000000; then continue to count; 16-bit reload mode, overflow after counting to the maximum 0xFFFF, generate an interrupt,

The timer/counter value is loaded with the ARR value and continues to count up.

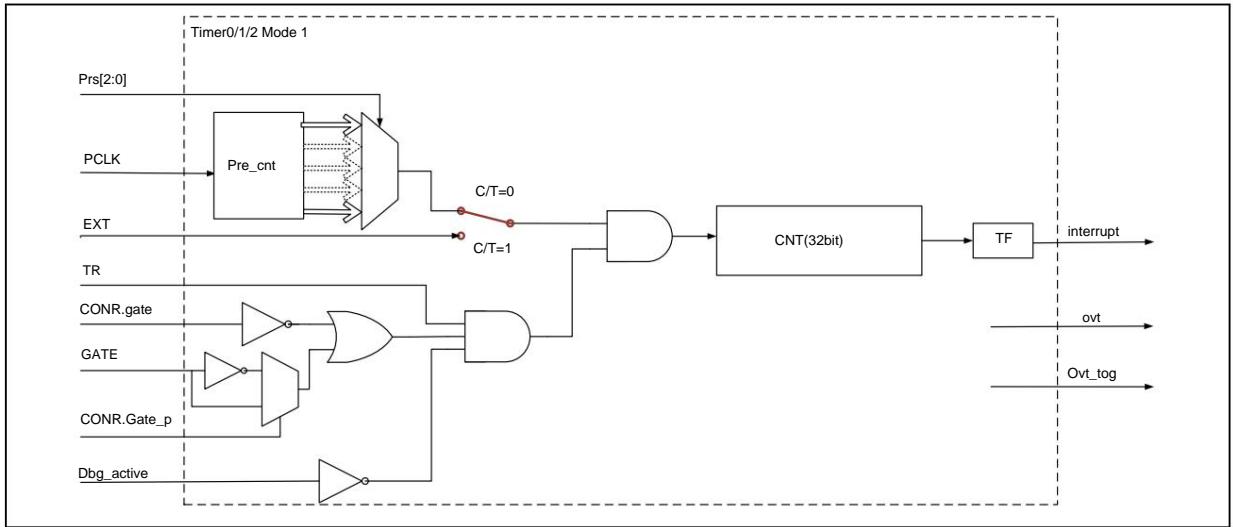


Figure 9-2 Timer Mode 1 Block Diagram

In mode 2 reload mode, the software processing speed needs to be considered when the timing time is set to a small value, otherwise the interrupt will occur.

The interrupt is lost if it is not processed in time.

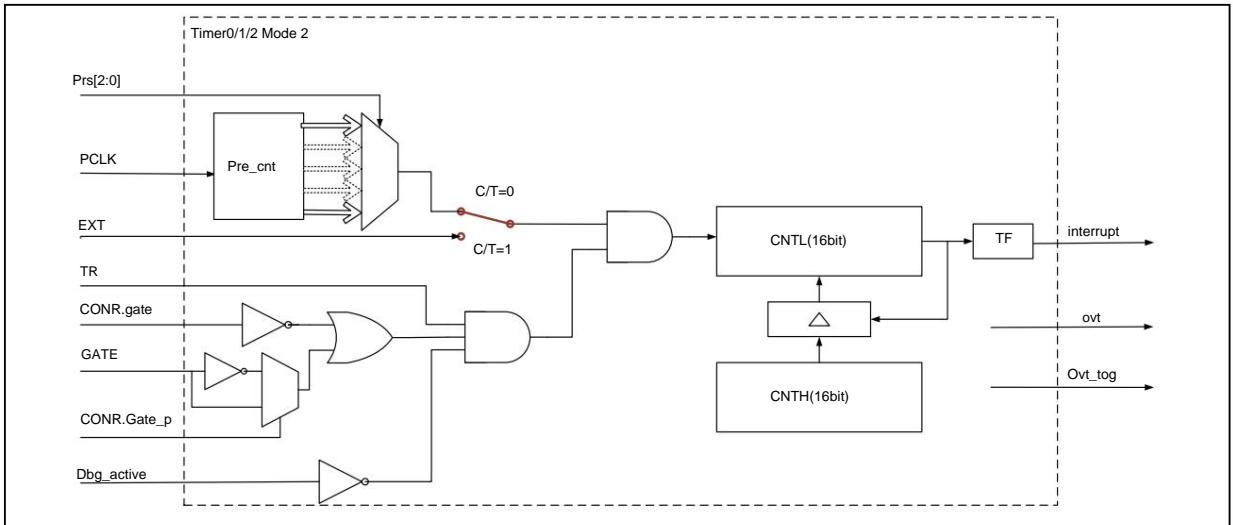


Figure 9-3 Block Diagram of Timer Mode 2

When the corresponding timer TR is set to 1, the timer starts to run. Mode 1 is a 32-bit timer/counter, which starts from

The initial value T set by the register starts to count, and an overflow interrupt is generated after the count reaches the maximum value. Then continue counting from 0. mold

Equation 2 is a 16-bit reload timer/counter. After startup, it starts to count up from the initial value of the register CNT, and counts to the maximum.

After the interrupt is generated after 0xFFFF, the value of the register ARR is reloaded into the counter CNT and continues to count up.

### 9.2.1 Counting function

The count function is used to measure the number of times an event occurs. In the counting function, the counter at each corresponding input

The falling edge of the clock accumulates once. The input signal is sampled by the internal PCLK, so the external input clock frequency cannot exceed

pass the system's Pclk clock. Counting to the maximum value will overflow and generate an interrupt. The interrupt flag needs to be cleared by software.

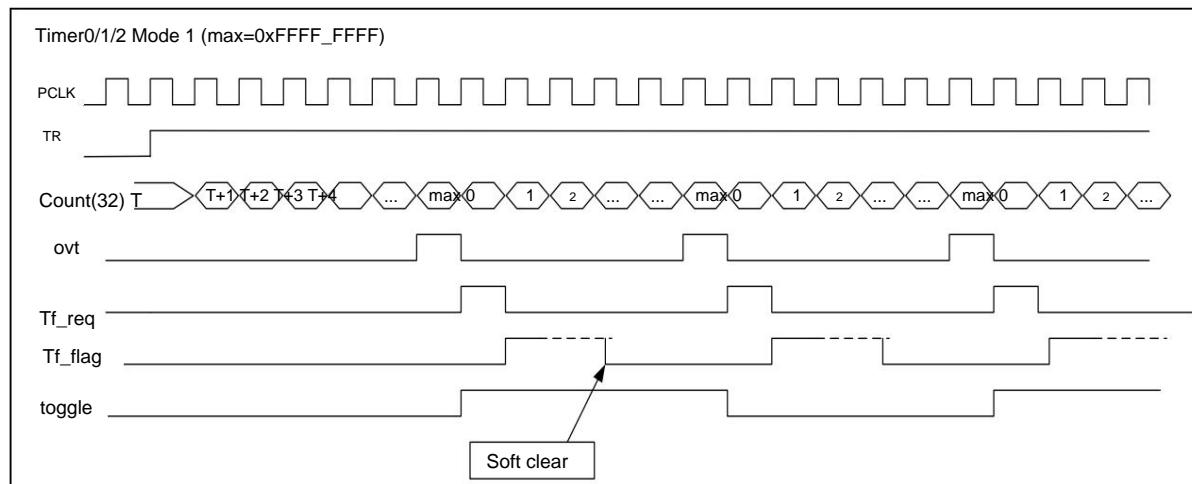


Figure 9-4 Mode 1 Timing Diagram

### 9.2.2 Timing function

The timing function is used to generate interval timing. In the timing function, the timer has a pre-division frequency, and the timer is pre-divided in each pre-division frequency.

One of the clocks is accumulated once, and when the count reaches the maximum value, it will overflow and generate an interrupt. The interrupt flag needs to be cleared by software.

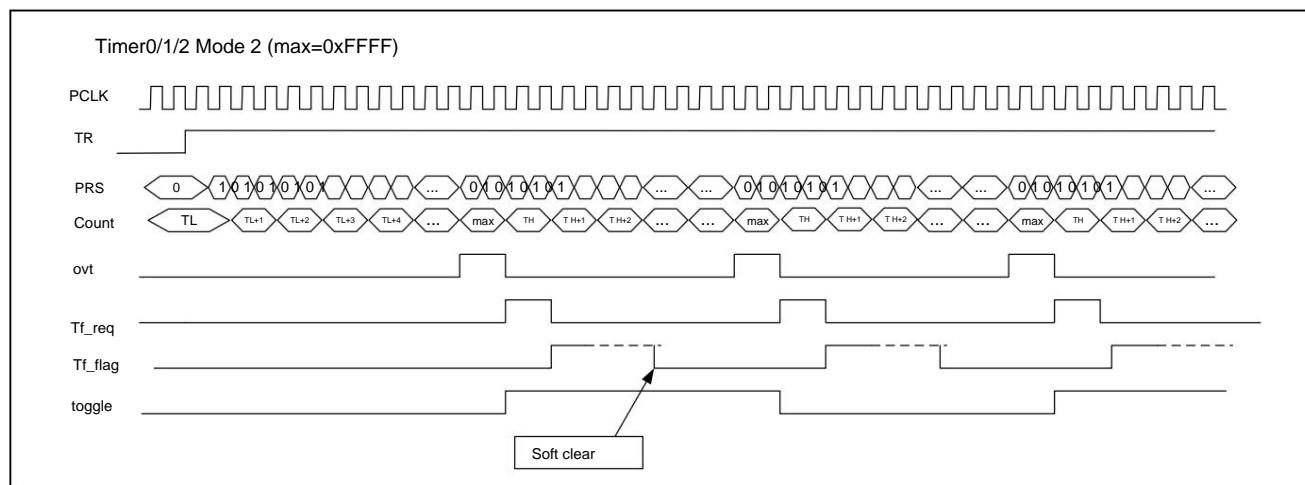


Figure 9-5 Mode 2 timing diagram (prescaler is set to 2)

### 9.2.3 Buzzer function

The function of driving the Buzzer can be realized through the inversion output function of the timer. When CR.TOG\_EN is 1,

TOG,TOGN output reverse. Set CR.TOG\_EN to 0 to set the port TOG and TOGN output at the same time

is 0. When the count clock is 4M, the Buzzer outputs the Timer reload mode with different frequencies 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 input directly from the port, or the RX signal of the UART can be input; it can also be configured as VC input as a 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, and the VC comparison output can be measured.

Pulse width, can realize external control counting.

The configuration selection RX input is controlled by the GPIO\_CTRL register, and the VC control is controlled by the VC control register.

Only one of port selection, UART input selection and VC input selection can be selected as gate control input.

### 9.3.2 Toggle Output Interconnect

The inversion of TIM0 outputs tog0 to the internal module UART0 to control the baud rate of UART0; the inversion of TIM1

Output tog1 to the internal module UART1 to control the baud rate of UART1; the flip output of TIM0/1/2 also outputs

Out to the port, it can drive the Buzzer to realize the control of the buzzer.

## 9.4 Base Timer register description

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 Flags
TIMx_ICLR	0X014	TIM0/1/2 Interrupt Clear Register

Table 9-1 Base Timer Register List

### 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	twenty four	twenty three	twenty two	twenty one	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	CNTs	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	twenty four	twenty three	twenty two	twenty one	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 sign		describe
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			

bit sign		describe
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, after setting to 1, the low-level is valid
8	GATE timer gate	0: No gating, the timer works when TR=1; 1: It only works 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 and TOGN output 0 at the same time 1: TOG, TOGN output signals with opposite phases. Available for buzzer.
2	CT	Counter/timer function selection 0: Timer function, the timer is counted by PCLK. 1: Counter function, the counter is counted by the falling edge of external input. External input is taken by PCLK As such, the external input clock frequency should be lower than 1/2 the 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 running 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 is invalid	

## 10 Low Power Timer (LPTIM)

### 10.1 Introduction to LPTimer

LPTimer is an asynchronous 16-bit timer/counter, which can still pass the internal low-speed RC or

Or external low-speed crystal oscillator timing/counting. Wake up the system in low power mode by interrupt.

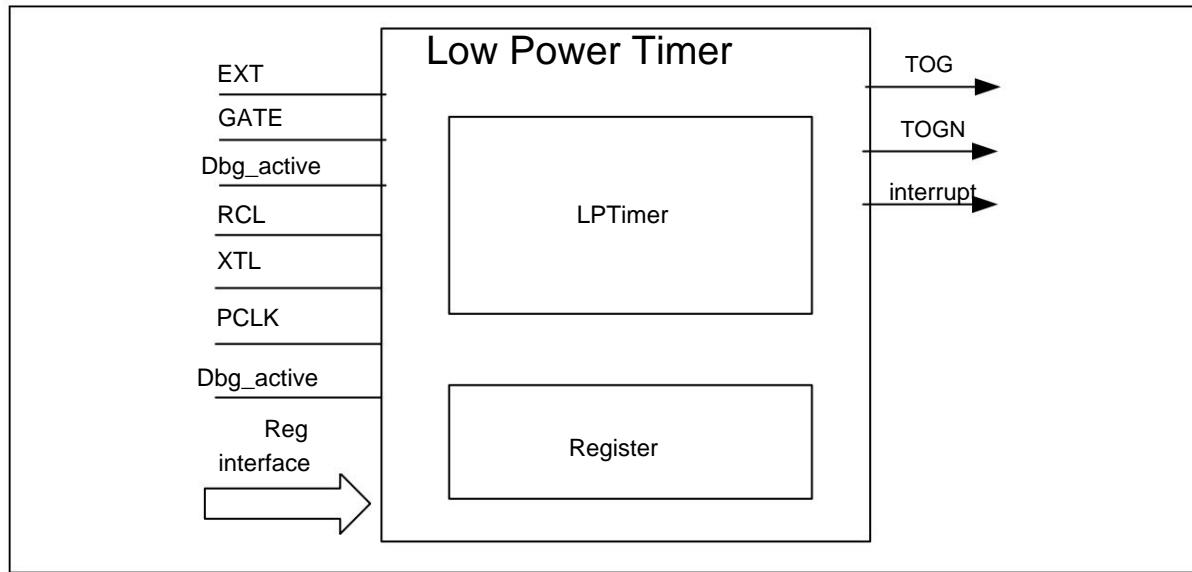


Figure 10-1 LPTimer block diagram

## 10.2 LPTimer function description

LPTimer supports 2 working modes, each timer/counter has independent control start signal, and external

Input clock, gating signal.

LPTimer uses EXT, GATE for counting function, EXT is used for the external input clock signal of the counter,

Gate is used for active level count enable signal.

The timer of LPTimer supports two working modes. By setting the MD option in the timer control register (CR),

Select the working mode. Mode 1 is a 16-bit free counting mode. Mode 2 is the 16-bit reload mode.

When the LPTimer starts, the value of the reload register ARR is automatically loaded into the counter.

LPTimer can choose three kinds of clocks as timer clock, which are selected by control register CR.TCK\_SEL. default

Select PCLK. Clock selection as shown in the table:

TCK_SEL	00	01	10	11
Timer clock PCLK		PCLK	XTL	RCL
Read Timer Count Value	Read Synchronized	Unsynchronized	Read Synchronized	Synchronized

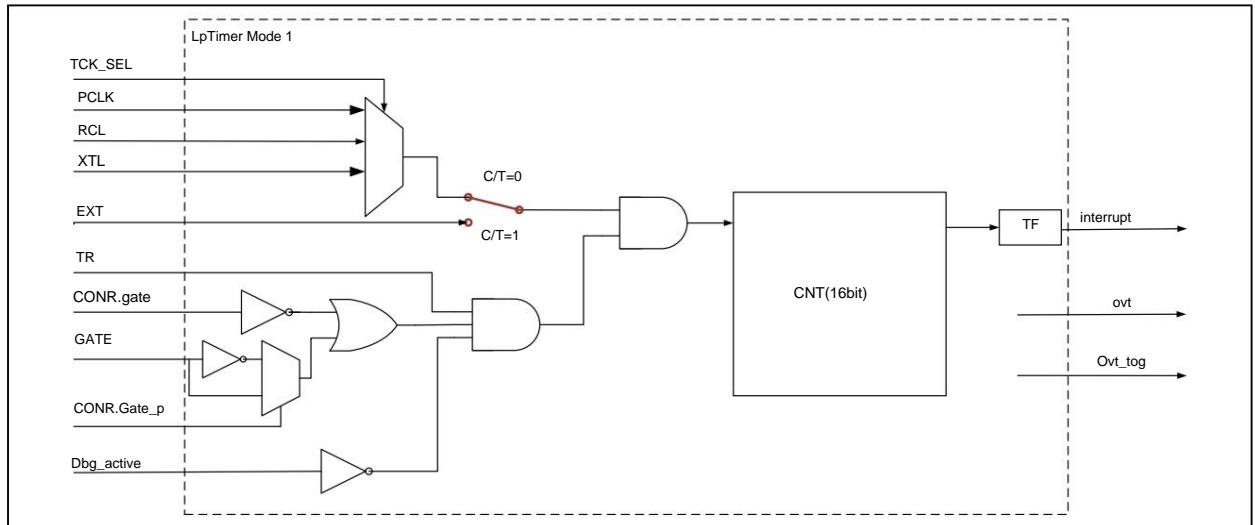


Figure 10-2 LPTimer Mode 1

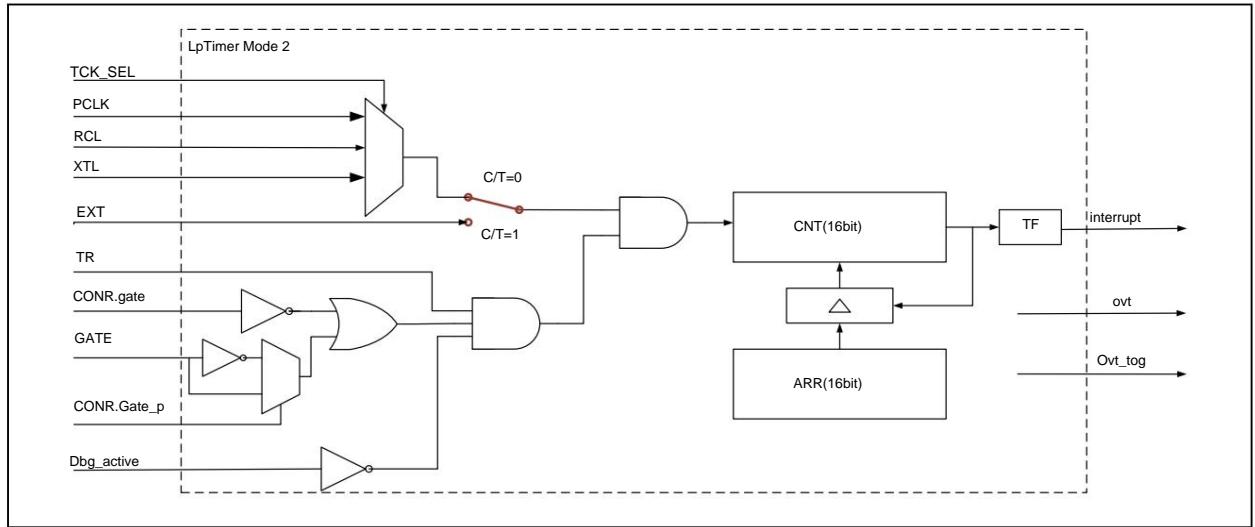


Figure 10-3 LPTimer Mode 2

When the corresponding timer TR is set to 1, the timer starts to run. The counter starts counting from the set value and counts up to

The overflow interrupt will be generated after the maximum 0xFFFF. Mode 1 After the interrupt is generated, the counter is cleared and continues to count up. Mode 2

After the interrupt is generated, the value of the register ARR is reloaded into the counter and counts up. The initial value of the counter CNT is starting

It is automatically loaded by ARR when the timer is activated.

Since the LPTimer is an asynchronous timer, the timer interrupt is synchronized from the Timer clock domain to the pclk pre-, reload mode

When the timer value is set greater than 0Xffff, the interrupt will be lost. It is recommended to use the interrupt function if the value of the reload register is less than

0XFFFC. Not using interrupts does not have this limitation.

### 10.2.1 Counting function

The count function is used to measure the number of times an event occurs. In the counting function, the counter at each corresponding input

The falling edge of the clock accumulates once. The input signal is sampled by the internal count clock, so the external input clock frequency cannot

Exceeds the system's 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 a clock and counts to the maximum

The value overflows and an interrupt is generated.



## 10.3 LPTimer Interconnect

### 10.3.1 GATE Interconnection

The GATE input can be input directly from the port, or the RX signal of the UART can be input; it can also be configured as VC input as a 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, and the VC comparison output can be measured.

Pulse width, can realize external control counting.

The UART selection control register is in the port control register GPIO\_CTRL4, and the VC output control register is in VC Control Module.

### 10.3.2 EXT interconnection

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

The VC pulse count can be measured through the internal interconnect configuration. The VC output control register is in the VC control block.

### 10.3.3 Toggle Output Interconnect

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

## 10.4 LPTimer register description

Base address 0X40000C00

Register offset address	address description	
CNTs	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

Table 10-1 LPTimer Register List



### 10.4.1 Counter count value register (LPTIM\_CNT)

Offset address: 0x060

Reset value: 0x0000 0000

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	Twenty	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	CNTs	Low-power timer count value read-only register. When the timer is started, the initial value of CNT is automatically loaded by ARR load.

### 10.4.2 Reload Register (LPTIM\_ARR)

Offset address: 0x064

Reset value: 0x0000 0000

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	Twenty	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 sign		describe													
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 only when WT_FLAG is 1 can be written. data input. WT2_FLAG will go low after writing to 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	RO	Reserved	TCK_SEL	TOG_EN	CT	MD	TR
	RW	R/W				R/W	R/W	R/W	R/W	R/W

bit sign		describe
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 gate	0: No gating, the timer works when TR=1; 1: Work only when gate is 1 and TR=1;
7	WT_FLAG	WT_FLAG WT, write synchronization flag 0: Synchronizing, writing ARR is invalid at this time 1: The synchronization is completed, and the ARR can be changed at this time
6	Reserved	Reserved bit, read as 0
5:4	TCK_SEL	LPTimer clock select 00 PCLK; 10:XTL; 11 ,RCL
3	TOG_EN	TOG output enable 0: TOG and TOGN output 0 at the same time 1: TOG, TOGN output signals with opposite phases. Available for buzzer.
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 external input to count. Sample clock used The clock selected by TCK_SEL, the external input clock should be lower than 1/2 of the 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 running control bits 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

--	--	--	--	--	--	--	--	--

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

--	--	--	--	--	--	--	--	--

Bit symbol	description	
31:1	Reserved	Reserved bit, read as 0
0	TFC	interrupt flag is cleared, write 0 to clear, write 1 is invalid

## 11 Programmable Counting Array (PCA)

### 11.1 Introduction to PCA

PCA (Programmable Counter Array) supports up to five 16-bit capture/compare

module. The timer/counter can be used as a general-purpose clock count/event counter with capture/compare function. PCA

Each of the modules can be independently programmed to provide input capture, output compare, or pulse width modulation. in addition

Module 4 has an additional watchdog timer mode.

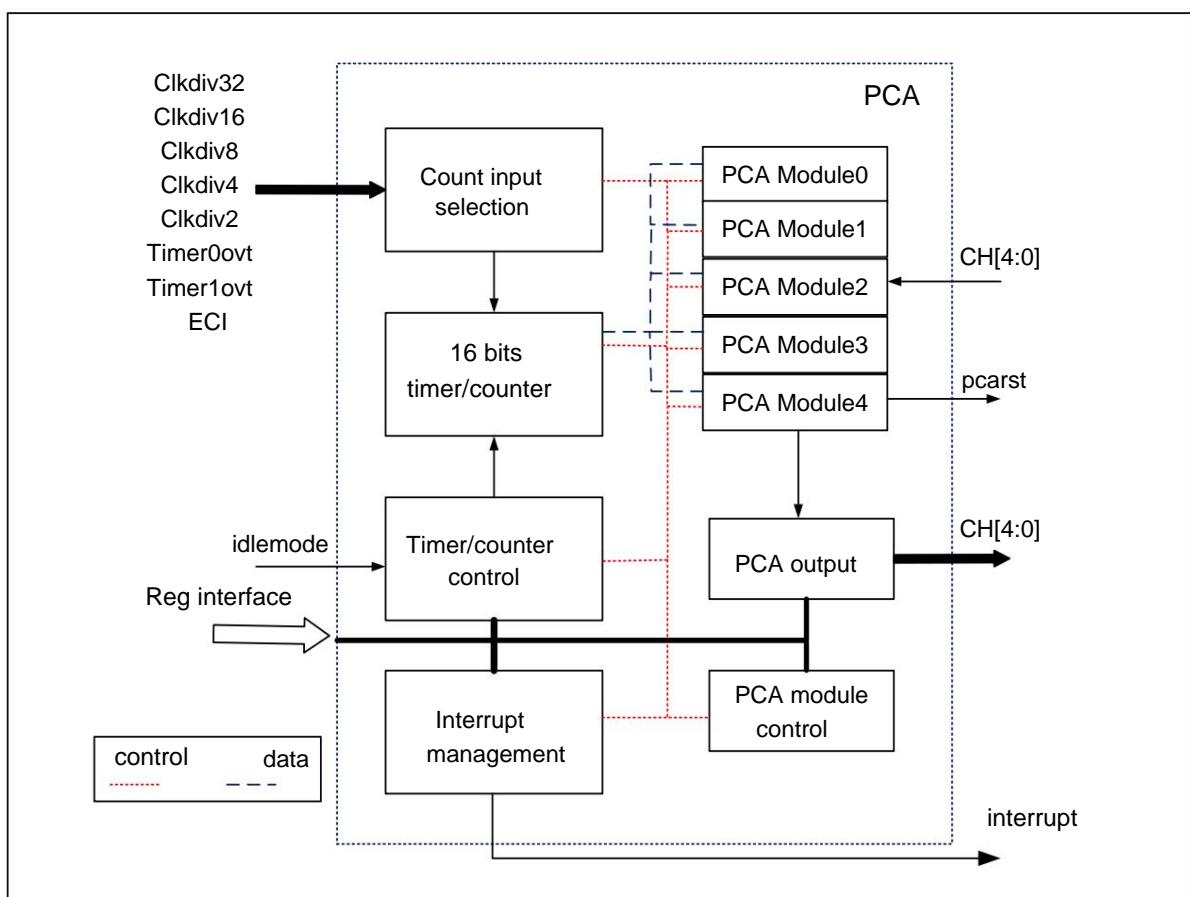


Figure 11-1 Overall block diagram of PCA



## 11.2 PCA function description

Each module can be configured to work independently, with three working modes: edge-triggered capture, output compare, 8-bit pulse

Wide modulation. Each module has its own function registers in the system controller, which are used for configuration

How modules work and exchange data with modules.

Each compare/capture module consists of a compare/capture register bank (CCAPx), and a 16-bit comparator, and

Various logic gate control components. The register bank is used to store the time or number of times, for external trigger capture conditions, or internal

External trigger compare condition. 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:

- ÿ Rising edge, falling edge or any edge trigger in 16-bit capture mode.

- ÿ Compare mode: 16-bit software timer, 16-bit high-speed output, 16-bit watchdog timer (module 4) or 8-bit

- Pulse width modulation.

- ÿ Not started.

The Compare/Capture Module Mode Registers (CCAPMx) determine the corresponding operating mode. For compare/capture modules do

When programmed, they are based on a common time count. The timer/counter is turned on and off via the CCON.CR bit i.e.

Controls the operation of the PCA timer/counter. Capture in a compare/capture block, software timer, high-speed output,

Set the compare/capture flag of the module (CCON.CCFx) and generate a PCA interrupt request, if the corresponding enable

Bits are set in the CCAPMx registers. 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

Counting counter. If the CMOD.CFIE bit is set to "1", the hardware automatically sets the PCA when the CNT overflows

overflow flag (CCON.CF) and generate PCA interrupt request. CMOD.CPS[2:0] Three bits select eight signals

Input to timer/counter.

- ÿ Divide by 32 of the system clock PCLK.

- ÿ Divide by 16 of the system clock PCLK.

- ÿ The system clock PCLK divided by 8.

- ÿ The system clock PCLK divided by 4.

- ÿ Divide by 2 of the system clock PCLK.

ÿ Timer 0 overflow. CNT is incremented each time Timer 0 overflows, thus providing the PCA's availability.

Variable programming frequency input.

ÿ Timer 1 overflow. CNT is incremented each time Timer 1 overflows, thus providing the PCA's availability.

Variable programming frequency input.

ÿ ECI. The CPU samples the PCA ECI every 4 PCLK clock cycles.

When high becomes low, CL automatically increases by 1, so the highest ECI input frequency cannot be higher than the system clock PCLK 1/8 to meet sampling needs.

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 counting starts (CCON.CR=1), in order to prevent counting errors, CNT is prohibited from being written.

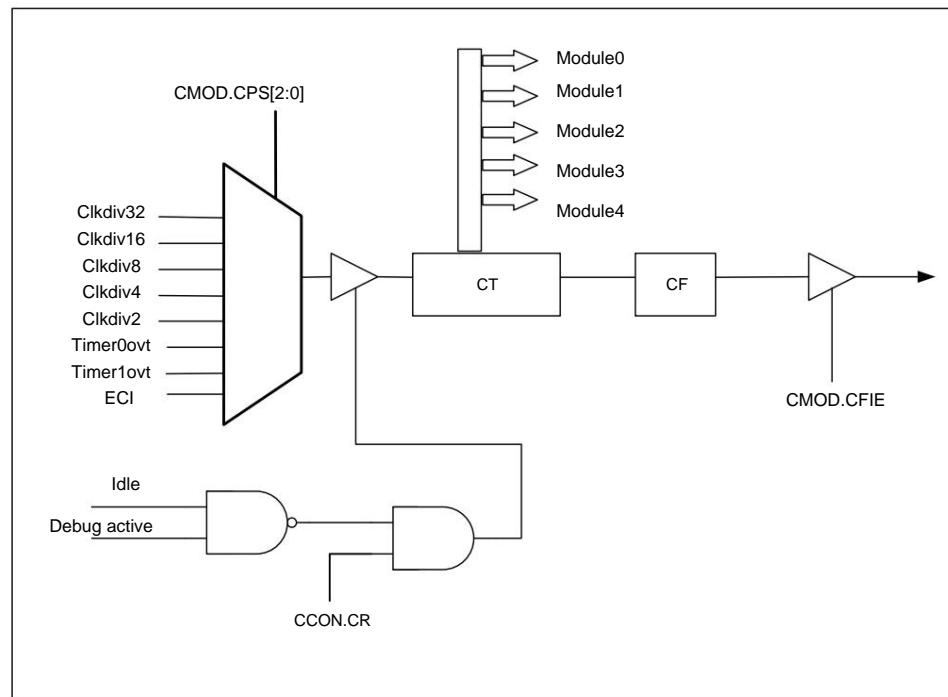


Figure 11-2 PCA Counter Block Diagram



## 11.2.2 PCA capture function

The PCA capture mode provides the functions of 5-channel PCA to measure pulse period, pulse width, duty cycle and phase difference.

A transition on the pin causes the PCA to capture the PCA counter/timer value and load it into the corresponding module.

Block 16-bit Capture/Compare Registers (CCAPx). The CCAPMx.CAPP and CCAPMx.CAPN bits are used for

Select the type of change that triggers the capture: low-to-high (positive edge), high-to-low (negative edge), or

any change (positive or negative). When a capture occurs, the capture/compare flag (CCFn) in CCON is set to

Logic '1' and generate an interrupt request (if CCF interrupt is enabled). When the CPU turns to the interrupt service routine,

The CCFn bit cannot be automatically cleared by hardware, and the user software writes the INTCL register to clear this flag. need to go up and down

Edge capture at the same time, recommended process: enable an edge capture first, and then use it in the interrupt service routine after the capture occurs.

Can another capture, and close the previous capture interrupt. Switch in sequence.

The resolution is equal to the timer/counter clock. The input signal must remain high or low for at least 2

clock cycle to ensure that the input signal can be recognized by the hardware.

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

Capture settings:

- ÿ When it needs to capture at the external rising edge, CCPMx.CAPP = "1" and CCAPMx.CAPN = "0"

- ÿ When capture on external falling edge is required, CCPMx.CAPP = "0" and CCAPMx.CAPN = "1"

- ÿ When external rising and falling edge capture is required, CCPMx.CAPP = "1" and CCAPMx.CAPN = "1"

ÿ When you need to capture at the external rising and falling edges at the same time and you need to know the captured edge, set the

CCPMx.CAPP = "1", after the rising edge capture occurs, switch to the falling edge capture in the interrupt service subroutine,

Set CCAPMx.CAPN = "1", and clear CCPMx.CAPP = "0". Set to after next capture

Rising edge capture, switching the captured edge in turn.

Notice:

- Subsequent captured values from the same module overwrite existing captured values. In order to maintain the captured value, in the interrupt service

The program stores it in RAM, this operation must be completed before the next event occurs, otherwise

The previous capture sample value will be lost.

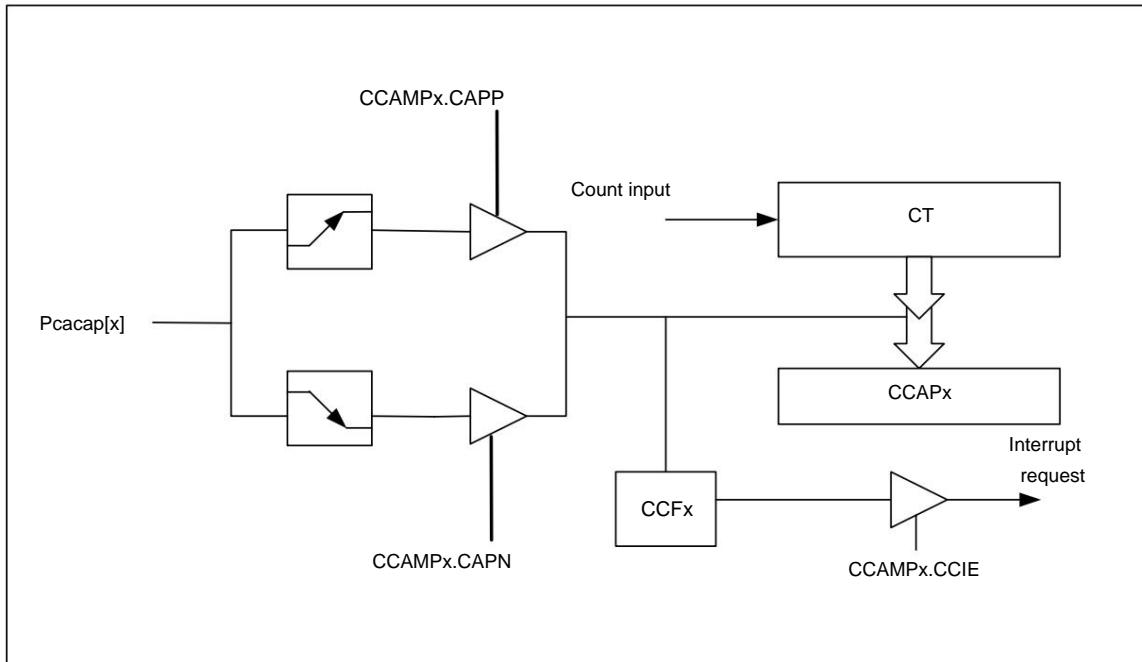


Figure 11-3 PCA capture function block diagram

### 11.2.3 PCA Compare Function

The compare function provides five modules with the following functions, timer, event counter, and pulse width modulation. Four modes of

Use compare function: 16-bit software timer mode, high-speed output mode, WDT mode and PWM mode. in front

Of the three functions, the compare/capture module compares the value of the 16-bit PCA timer/counter with the value preloaded into the module

16-bit value in the CCAPx register. In PWM mode, the PCA module continuously clocks the PCA timer

The /counter low byte register (CNT) is compared with a value in bit 8 of the CCAPxL module register. Every

Comparing once every 4 clock cycles, it matches the clock rate of the fastest PCA timer/counter.

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

To properly use the module in compare mode, follow the general procedure below:

- ÿ Select the operating mode of the PCA module
- ÿ Select the input signal of PCA timer/counter.
- ÿ The compare value is loaded into the module's compare/capture register pair.
- ÿ Set PCA timer/counter running control bits.
- ÿ Interrupt is generated after a match, and the compare/capture flag of the module is cleared.

#### 11.2.3.1 16 -Bit Software Counter Mode

To set up a compare/capture module in 16-bit software timer mode, it is necessary to set CCAPMx.ECOM and

CCAPMx.MAT bits. Once between the PCA timer/counter and compare/capture registers (CCAPx)

A match occurs, which sets the module's compare/capture flags (CCON.CCFx). This will generate an interrupt request,

If the corresponding interrupt enable bit (CCAPMx.CCIE) is set. Interrupt due to compare/capture flag not cleared by hardware

When processing, the user must clear the software flag. In the interrupt service routine, a new 16-bit compare value can be written

into the compare/capture register (CCAPx).

Note: When updating these registers, in order to prevent invalid matches from occurring, user software should first write CCAPxL,

After writing CCAPxH. A write such as CCAPxL clears the disable compare function ECOMx bit, while writing

CCAPxH will also set the ECOMx bit, re-enabling the compare function. i.e. when capture/compare to PCA0

When writing a 16-bit value to the register, the low byte should be written first.

### 11.2.3.2 High-speed output mode

In high-speed output mode, whenever the value in the PCA counter is compared with the module's 16-bit capture/compare register (CCAPx)

When a match occurs, the logic level on the CAP/CMP[x] pins of the module PCA will change. This can provide

It has higher precision than switching IO output, because this high-speed output will not be interrupted to affect the output frequency, relying on

If the CPU switches the IO output, the power consumption and accuracy are lacking.

To set the high-speed output mode of a compare/capture module, set CCAPMx.ECOM, CCAPMx.MAT and

CCAPMx.TOG bit. A match between PCA timer/counter and compare/capture registers (CCAPx) switches

Change the PCA's CAP/CMP[x] signal and set the module's compare/capture flag (CCON.CCFx). by soft

software to set or clear the CAP/CMP[x] signal of 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

When a match occurs, an interrupt request can be generated. Due to the compare/capture flag interrupt that cannot be cleared by hardware, the user must

Clear this flag in software. If the user does not change the compare/capture registers in the interrupt routine, the PCA does not

The comparison value is recounted, and the next rollover occurs if there is a match. In the interrupt service routine, a new 16-bit bit

The comparison 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

Post-CCAPxH. Writing to CCAPxL clears the ECOM bit that disables the compare function, while writing to CCAPxH sets the

ECOM bit to re-enable the compare function.

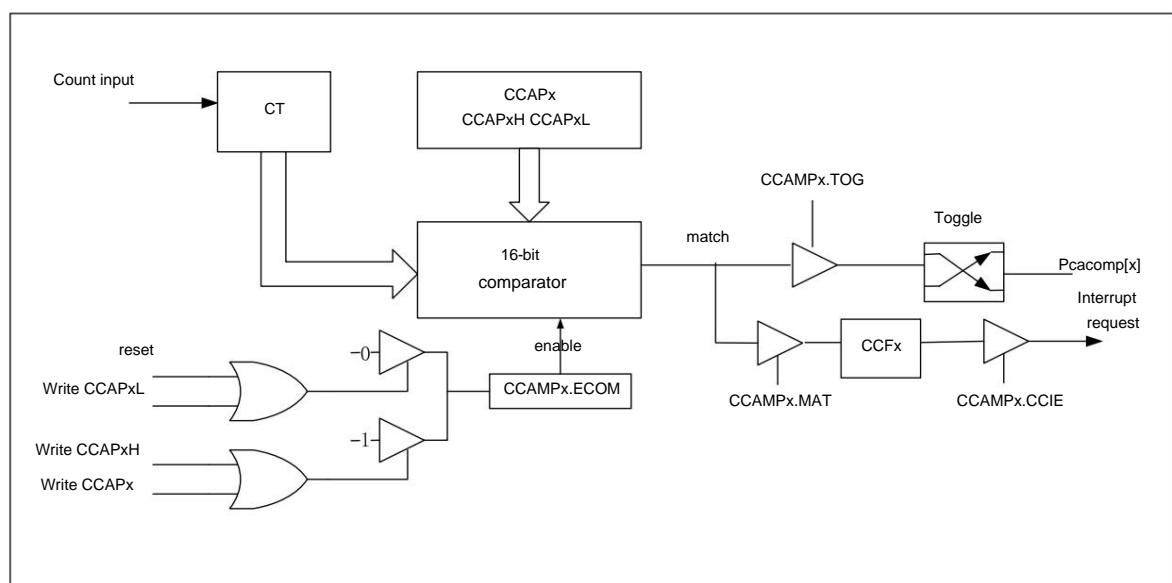


Figure 11-4 PCA Compare Function Block Diagram

### 11.2.3.3 WDT function of PCA module 4

In addition to a WDT hardware module, module 4 of PCA, this product also provides a 16-bit programmable frequency WDT. When the count value of the PCA timer/counter is compared with the value stored in module 4/capture register (CCAP4) When matched, this mode generates a reset signal. The WDT reset signal of the PCA acts as an independent reset signal. With external reset (RST), hardware watchdog reset (WDTRST) and LVD low voltage reset, POR power on and off combined with electrical reset. Users can freely combine or use them individually. Module 4 is the only one with WDT mode Modules for PCA. When not set to WDT, it can be used independently in other modes.

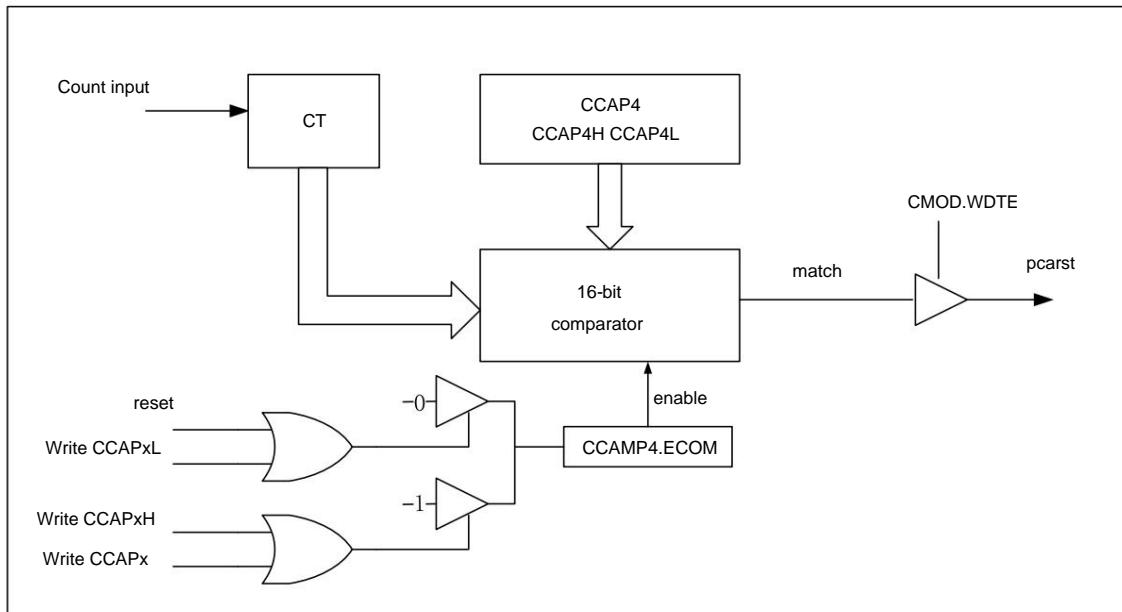


Figure 11-5 PCA WDT functional block diagram

When using PCA module 4 as WDT, you must set CCAPM4.ECOM4, CCAPM4.MAT4 to and CMOD.WDTE. In addition, PCA timer/counter can set CMOD.CPS to select different output count frequency.

A 16-bit compare value is entered in the compare/capture register (CCAP4). In the PCA Timer/Counter (CNT), Enter a 16-bit initial value or use the reset value (0x0000). These values are multiplied by the PCA input pulse rate The difference between the WDT matching runtimes is determined. Set Timer/Counter Run Control Bit (CCON.CR) Start PCA WDT. On each match, the PCA's WDT generates a reset signal. To prevent a PCA WDT To reset, the user has three options.

- ÿ Periodically compare the value of CCAP4 for changes, so a match never occurs.
- ÿ The PCA timer/counter value (CNT) is changed periodically so a match never occurs.



ÿ 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 WDT is not disabled in the third option.

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

PCA WDT configuration process:

1. Configure the WDT compare/capture register PCA\_CCAP4
2. Configure the PCA count register PCA\_CNT
3. Configure PCA\_CCAMP4 to select compare match function
4. Configure PCA\_CMOD to select the input clock and enable the WDT function
5. Start PCA
6. Select clear PCA WDT clear method 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. 5 PCA modules are available

to be independently used to generate a pulse width modulation (PWM) output at the CAP/CMP[x] pin of the corresponding PCA, the pulse

The width is 8-bit resolution. The frequency of the PWM output depends on the time base of the PCA counter/timer. Use modules

The capture/compare register CCAPxL is used to change the duty cycle of the PWM output signal. When PCA Counter/Timing

When the low byte (CNTL) of the device is equal to the value in CCAPxL, the output on the CAP/CMP[x] pin of the PCA

output is set to "1"; when the count value in CNTL overflows, the CAP/CMP[x] output of the PCA is reset to "0". when

The value stored in CCAPxH when the counter/timer low byte CNTL overflows (from 0xFF to 0x00)

is automatically loaded into CCAPxL without software intervention.

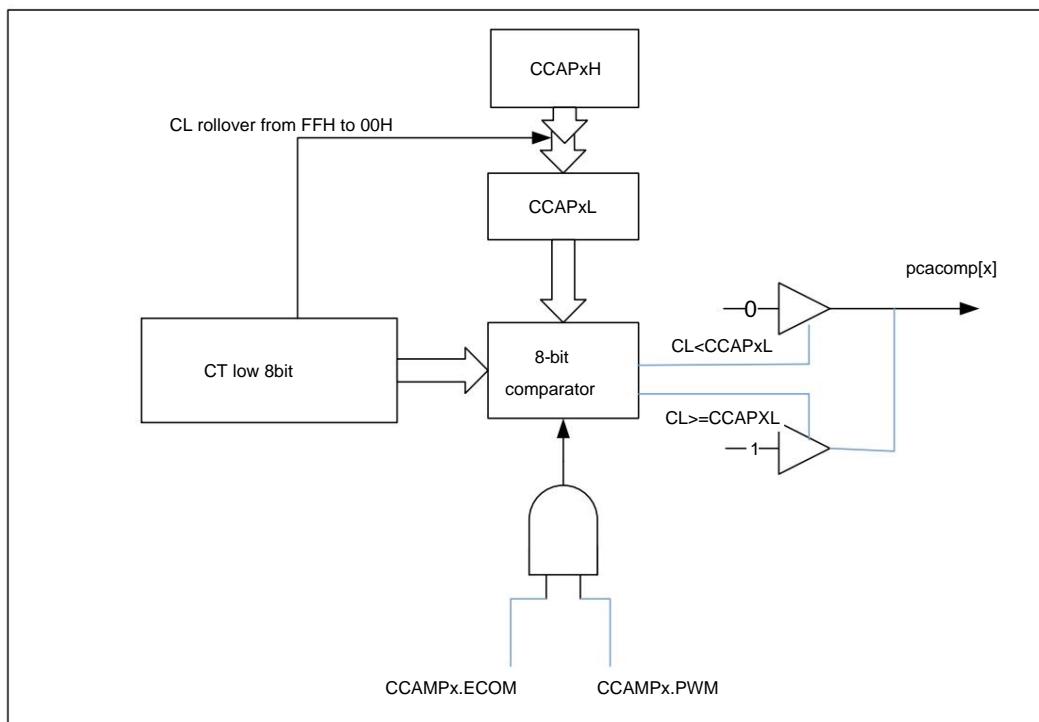


Figure 11-6 PCA PWM functional block diagram

In this mode, the value of the low byte (CNTL) of the PCA timer/counter is continuously compared with the value of the compare/capture register.

The value of the low byte (CCAPxL) is compared. The output waveform is low when CNTL < CCAPxL; when CNTL>=

The output waveform is high at CCAPxL. When CNTL overflows, the system automatically loads the value of CCAPxH into

Within CCAPxL, a new count cycle begins.

The value of CCAPxL determines the duty cycle of the current cycle, and the value of CCAPxH determines the duty cycle of the next cycle. Change

The value in CCAPxL changes the duty cycle of the PWM. As shown in the figure below, adjust the value of CCAPxL to 0~255

A duty cycle of 100% to 0.4% can be achieved. To prevent glitches, it is recommended not to change the value of CCAPxL directly,

CCAPxL is automatically loaded by hardware on the next cycle by changing the value of CCAPxH.

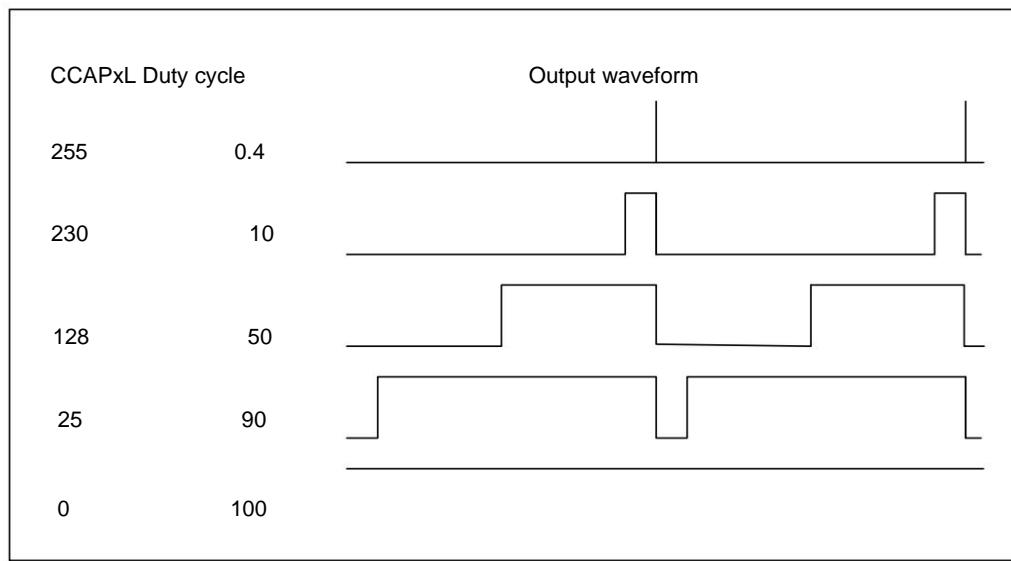


Figure 11-7 PCA PWM output waveform

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

How ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	Works		
X	1	0	0	0	0	0	X capture with positive edge trigger		
X	0	1	0	0	0	0	X capture with negative edge trigger		
X	1	1	0	0	0	0	X Edge-triggered capture		
1	0	0	1	0	0	0	X software timer		
1	0	0	1	1	0	0	X high speed output		
1	0	0	0	0	0	1	X 8-bit PWM		

Table 11-1 PCA compare capture function module settings

### 11.3 Interconnection and control of PCA module and other modules

#### 11.3.1 ECI Interconnect

ECI input can be an external selection of different input ports through IO MUX, or it can be a comparison of internal VC filtered output. The VC output control register is in the VC control block.

#### 11.3.2 PCACAP0

The capture input for channel 0 can be:

- ÿ Input port of external IO MUX
- ÿ MUX input of RX of external UART
- ÿ Compare filtered output of internal VC1

The UART selection control register is in the port control register GPIO\_CTRL2, and the VC output control register is in VC Control Module.

#### 11.3.3 PCACAP1

The capture input for channel 1 can be:

- ÿ Input port of external IO MUX
- ÿ MUX input of RX of external UART
- ÿ Compare filtered output of internal VC2

The UART selection control register is in the port control register GPIO\_CTRL2, and the VC output control register is in VC Control Module.

#### 11.3.4 PCACAP[4:2]

The capture inputs for channels 2,3,4 can be:

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

The UART selection control register is in the port control register GPIO\_CTRL2.

## 11.4 PCA register description

Base address 0X40001000

register	Offset address	description
CCON	0X000	PCA Control Register
CMOD	0X004	PCA Mode Register
CNTs	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 upper 8-bit register
CCAP0L	0X028	PCA compare/capture module 0 lower 8-bit register
CCAP1H	0X02C	PCA compare/capture module 1 upper 8-bit register
CCAP1L	0X030	PCA compare/capture module 1 lower 8-bit register
CCAP2H	0X034	PCA compare/capture module 2 upper 8-bit register
CCAP2L	0X038	PCA compare/capture module 2 lower 8-bit register
CCAP3H	0X03C	PCA compare/capture module 3 upper 8-bit registers
CCAP3L	0X040	PCA compare/capture module 3 lower 8-bit register
CCAP4H	0X044	PCA compare/capture module 4 high 8-bit registers
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 of PCA compare/capture module 0
CCAP1	0X054	16-bit register of PCA compare/capture module 1
CCAP2	0X058	16-bit register of PCA compare/capture module 2
CCAP3	0X05C	16-bit register of PCA compare/capture module 3
CCAP4	0X060	16-bit register of PCA compare/capture module 4

Table 11-2 PCA register list

### 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)  CF is set by hardware when the PCA count overflows, if the CFIE bit in the CMOD register is 1, then The CF flag can generate an interrupt  1: Counter overflow occurs; 0: No overflow;
6	CR	PCA Counter Run Control Bits  1: Start PCA counter counting 0: Turn off PCA counter counting
5	Reserved	Reserved bit
4	CCF4	PCA Counter Module 4 Compare/Capture Flag  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  This bit is set by hardware when a match or capture occurs. (write invalid) This flag will generate a PCA interrupt when CCAPM3.CCIE is set
2	CCF2	PCA Counter Module 2 Compare/Capture Flag  This bit is set by hardware when a match or capture occurs. (write invalid) This flag will generate a PCA interrupt when CCAPM2.CCIE is set
1	CCF1	PCA Counter Module 1 Compare/Capture Flag  This bit is set by hardware when a match or capture occurs. (write invalid) This flag will generate a PCA interrupt when CCAPM1.CCIE is set
0	CCF0	PCA Counter Module 0 Compare/Capture Flag  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\_CMOD)

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	Does PCA stop working in CIDL 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: Enable 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 divided by four for sampling
0	CFIE	PCA counter interrupt enable control signal 1: Enable interrupt 0: Disable interrupts

### 11.4.3 Count register (PCA\_CNT)

Offset address: 0x008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	Twenty	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNTs															
R/W															

Bit symbol	description	
31:16	Reserved	Reserved bit
15:0	CNT timer counter value	CNT can be written only when PCA is stopped, otherwise the writing 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	PCA counter overflow flag is cleared (software write 0 to clear, write 1 to be invalid), 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 to be invalid), the read value is 1
3	CCF3	PCA counter module 3 compare/capture flag clear (software write 0 to clear, write 1 to be invalid), the read value is 1
2	CCF2	PCA counter module 2 compare/capture flag clear (software write 0 to clear, write 1 to be invalid), the read value is 1
1	CCF1	PCA counter module 1 compare/capture flag clear (software write 0 to clear, write 1 to be invalid), the read value is 1
0	CCF0	PCA counter module 0 compare/capture flag clear (software write 0 to clear, write 1 to be invalid), the read value is 1

### 11.4.5 Compare Capture Mode Registers (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: Comparator function is enabled; 0: Stronger function is disabled; When PCA is used for software counter, high speed output, PWM mode, WDT mode, set ECOM Writing to the CCAMPx or CCAMPx registers will automatically set ECOM; writing to the CCAMPLx register will automatically set ECOM 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 enable	match control bits 1: Once the PCA count value matches the value of the compare/capture register of the module, the CCON register will be set. Break flag CCFx (x=0-4) 0: Disable matching function
2	TOG toggle	control bit 1: Working in PCA high-speed output mode, the value of the PCA counter is equal to the value of the compare/capture register of the module. Once matched, the CCPx pin toggles 0: Disable flip function
1	PWM pulse width	modulation control bits 1: Enable CCPx pin as PWM output 0: Disable PWM pulse width modulation function The <b>PWM</b> function is valid only when <b>CCAPMx[6:0]=100_0010</b>
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 sign		describe
31:8	Reserved Reserved bit	
7:0	CCAPx[15:8] compare/capture mode upper 8-bit register  When PCA mode is used in compare/capture mode, it is used to save the upper 8 bits of the 16-bit capture count value; write The CCAPxH register automatically sets the ECOM bit in register CCAPMx.  When PCA mode is used in PWM mode, it is used to control the output duty cycle load register, in the counter  When the lower 8 bits overflow, the load register will be automatically updated to the PWM compare 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 PCA mode is used in compare/capture mode, it is used to save the lower 8 bits of the 16-bit capture count value; write The CCAPxH register automatically clears the ECOM bit in register CCAPMx.  When the PCA mode is used in the PWM mode, it is used to control the output duty cycle 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]. The PWM output is low, otherwise PWM output high level.	

#### 11.4.8 Compare capture 16-bit registers (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	Twenty four	Twenty three	Twenty two	Twenty one	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	<p>Compare/Capture Mode 16-bit Register</p> <p>When PCA mode is used in compare/capture mode, it is used to save the 16-bit capture count value; write CCAPx to register register will set the ECOM bit in register CCAPMx.</p> <p>Writing to the CCAPX register is equivalent to writing to the two 8-bit registers, CCAPxL and CCAPxH. in comparison/ In capture mode, this register can be read and written directly. In PWM mode, use CCAPxL and CCAPxH register</p>

#### 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 CCAP03 CCAP02 CCAP01 CCAP00	R/W	R/W	R/W	R/W

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

## 12 advanced timers (TIM4/5/6)

### 12.1 Introduction to Advanced Timers

The advanced timer is a Timer4/5/6 that contains three timers. Timer4/5/6 high-performance counter with the same function,

It can be used to count to generate different forms of clock waveforms, and a timer can generate a complementary pair of PWM or independent clock waveforms.

Independent 2-way PWM output can capture external input for pulse width or period measurement.

The basic functions and features of the advanced timer are shown in the table.

waveform mode	Sawtooth wave, triangular
basic skills	wave • Increase and decrement
	counting direction • Software
	synchronization • Hardware
	synchronization • Buffer function •
	Quadrature code count • General
	PWM output • Brake protection •
	AOS related action count Compare
	match interrupt count period match
interrupt type	interrupt dead time error interrupt
	table 12-1 Advanced Timer Basic
	Features

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

Table 12-2 Advanced Timer Port List

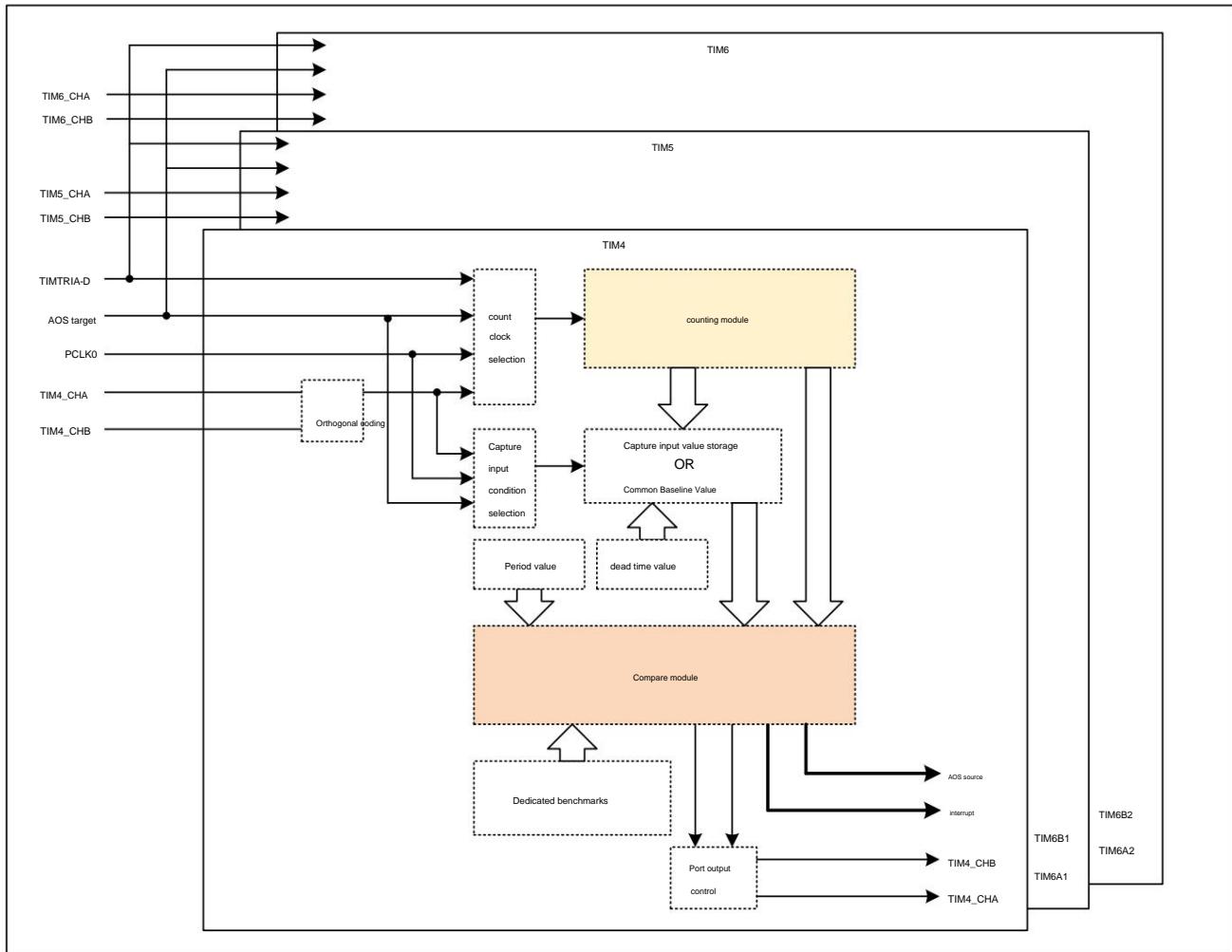


Figure 12-1 Advanced Timer block diagram

## 12.2 Advanced Timer function description

### 12.2.1 Basic Actions

#### 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 again due to not

The same internal counting action is subdivided, and the triangular wave mode is divided into triangular wave A mode and triangular wave B mode. sawtooth wave

The basic waveforms of and triangular waves are shown in Figure 12-2 and Figure 12-3. The difference between the triangle wave A mode and the triangle wave B mode is

There is a difference in buffer transmission. Triangular wave A mode only has one buffer transmission (valley point) per cycle, while triangular wave B

The mode has two buffer transfers (peak and valley) in one cycle.

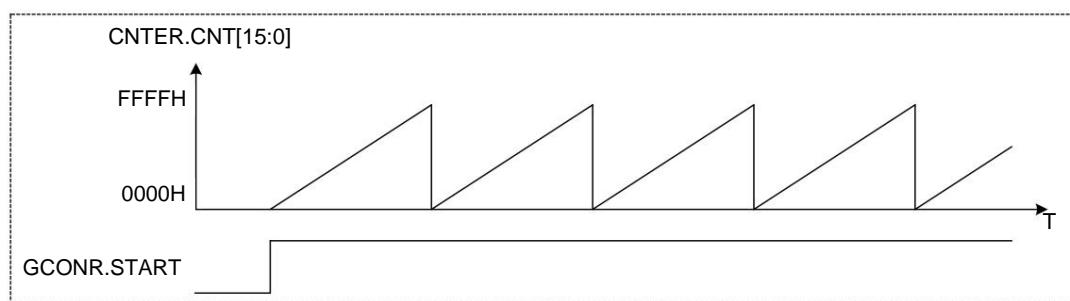


Figure 12-2 Ramp waveform (count up)

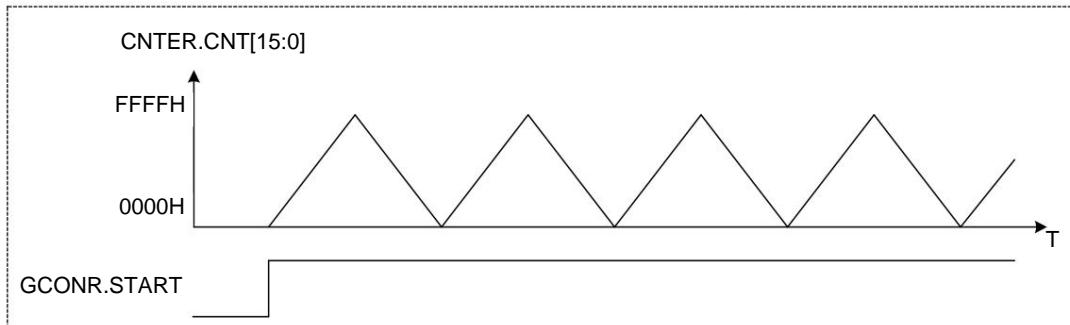


Figure 12-3 Triangle waveform

### 12.2.1.2 Compare Output

Timer4/5/6 A timer has 2 compare output ports (CHxA, CHxB), which can be used between the count value and the count reference.

Outputs the specified level when the value compares match. The GCMAR and GCMBR registers correspond to CHxA and CHxB respectively

Count comparison reference value. When the count value of the counter is equal to GCMAR, the CHxA port outputs the specified power

level; when the count value of the counter is equal to GCMBR, the CHxB port outputs the specified level.

The count start level, stop level, and count comparison match level of CHxA and CHxB ports can be controlled by the port.

Control Register (PCONR) PCONR.STACA, PCONR.STPCA, PCONR.STASTPSA,

PCONR.CMPCA[1:0], PCONR.PERCA[1:0] and PCONR.STACB, PCONR.STPCB,

PCONR.STASTPSB, PCONR.CMPCB[1:0], PCONR.PERCB[1:0] bit setting. Figure 12-4 is

Example of operation of comparison output.

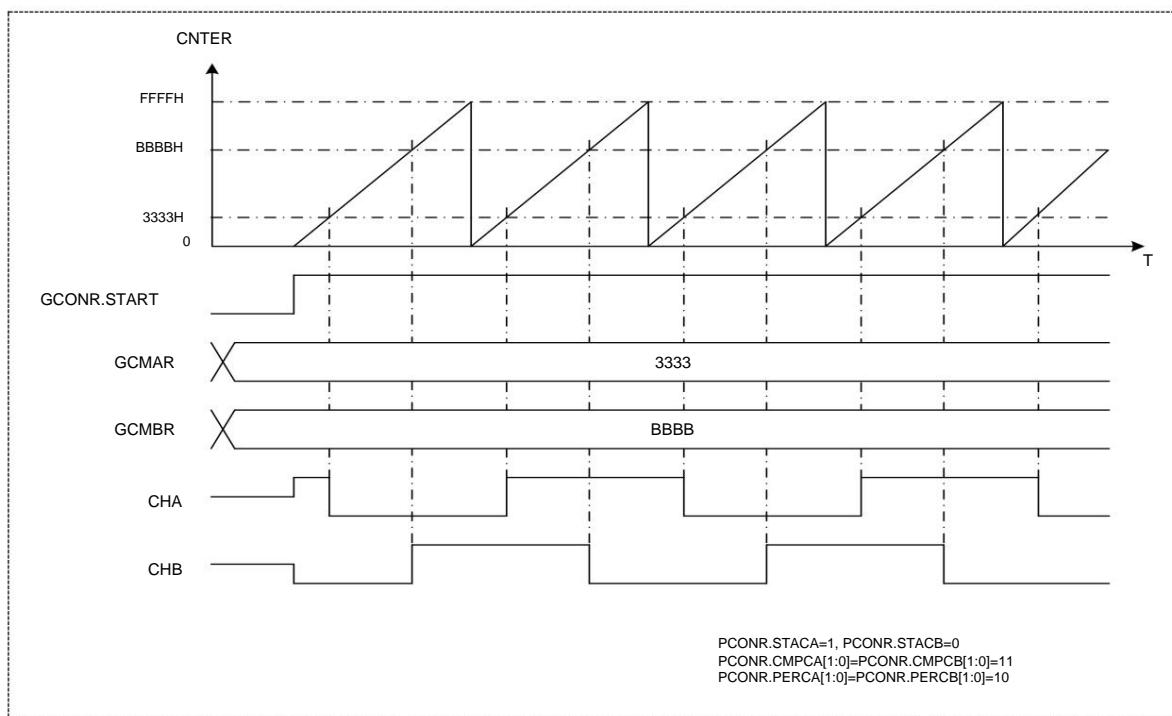


Figure 12-4 Comparison output action

### 12.2.1.3 Capture Input

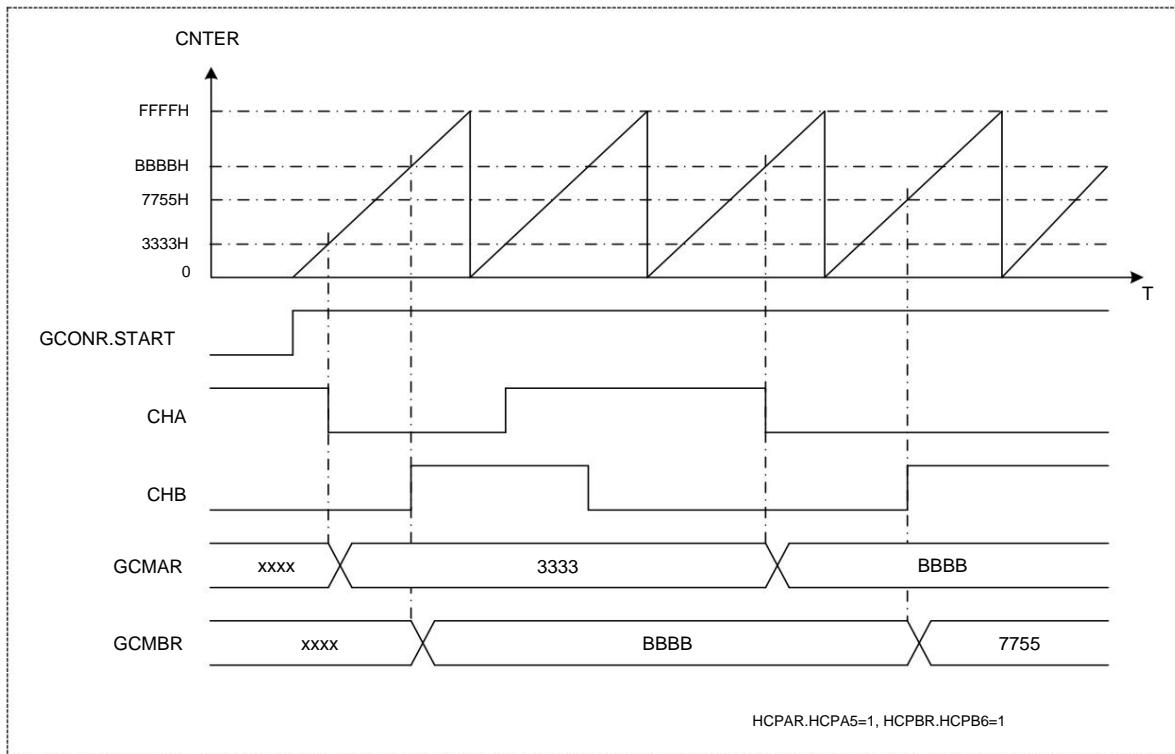


Figure 12-5 Capture input action

Timer4/5/6 all have capture input function, with 2 groups of capture input registers (GCMAR, GCMBR), use

to save the captured count value. Set PCONR.CAPCA of the Port Control Register (PCONR),

When the PCONR.CAPCB bit is 1, the capture input function is valid. When the corresponding capture input condition is set and the

When the condition is valid, the current count value is saved to the corresponding register (GCMAR, GCMBR).

The conditions for each set of capture inputs can be AOS event triggers, TIMTRIA-TIMTRID inputs, CHxA or

The input of CHxB, etc., the specific conditions can be selected through the hardware capture event selection register (HCPAR, HCPBR)

to set. Figure 12-5 shows an example of the action to capture input.



## 12.2.2 Clock Source Selection

Timer4/5/6 count clock can have the following options:

- a. PCLK and the 2, 4, 8, 16, 64, 256, 1024 frequency division of PCLK (GCONR.CKDIV[2:0] set

Certainly)

- b. AOS event trigger input (set by HCUPR.HCUP [19:16] or HCDOR.HCDO [19:16])

- c. Orthogonal encoding input for CHxA and CHxB (HCUPR.HCUP[7:0] or HCDOR.HCDO[7:0]

set up)

- d. Port input for TIMTRIA-TIMTRID (HCUPR.HCUP[15:8] or HCDOR.HCDO[15:8]

set up)

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

When c, d clock, a clock is invalid automatically.



### 12.2.3 Counting direction

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

#### 12.2.3.1 Ramp count direction

In the sawtooth 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 decrement after counting to overflow

Counting mode; when counting down, set GCONR.DIR=1 (counting up), the counter counts to underflow

Then it changes to count up mode.

When counting is stopped, the GCONR.DIR bit is set. Then after counting starts until overflow or underflow, GCONR.DIR settings will be reflected in the count.

#### 12.2.3.2 Triangular wave count direction

In triangular wave mode, the counting direction can only be set when the counter is stopped. Setting the counting direction during counting is invalid.

When counting is stopped, the GCONR.DIR bit is set. Then after counting starts until overflow or underflow, GCONR.DIR settings will be reflected in the count.

## 12.2.4 Digital Filtering

The CHxA, CHxB, TIMTRIA~D port inputs of Timer4/5/6 all have digital filtering function. can be set by

The relevant enable bit of the filter control register (FCONR) enables the filter function of the corresponding port. reference time for filtering

The clock is also set through the Filter Control Register (FCONR).

When the filtered sampling reference clock samples a consistent level on the port 3 times, the level is regarded as a valid level and transmitted to the

Inside the module; the level less than 3 times will be filtered out as external interference and will not be transmitted to the inside of the module. its action

For example, as shown in Figure 12-6.

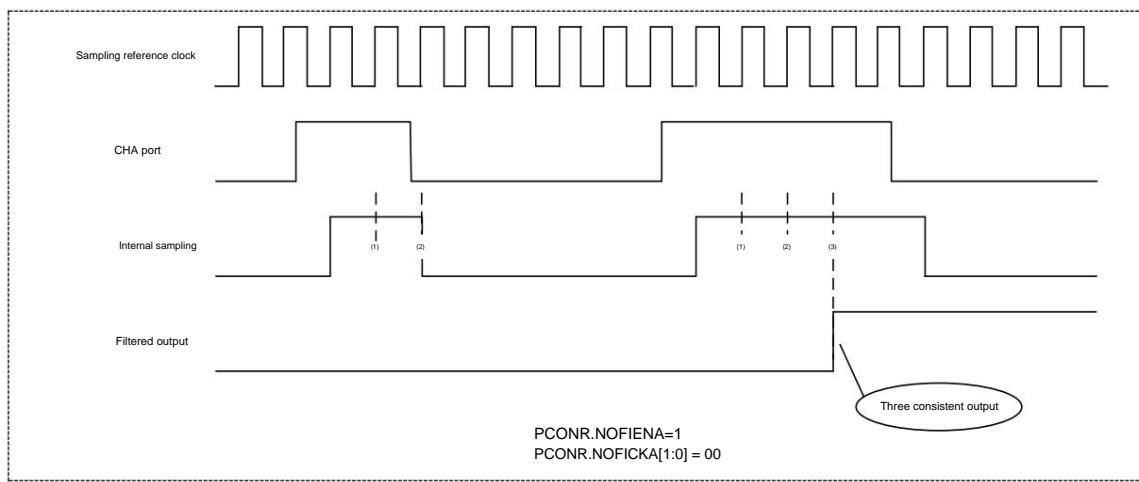


Figure 12-6 Filtering function of capture input port

The TIMTRIA~D port is a port shared by a group of Timer4/5/6. The digital filtering function of this group of ports is only in the

Timer4 is implemented, other timers Timer5/6 are invalid for the digital filtering function setting of this group of ports.

### 12.2.5 Software synchronization

#### 12.2.5.1 Software synchronization startup

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

synchronous start.

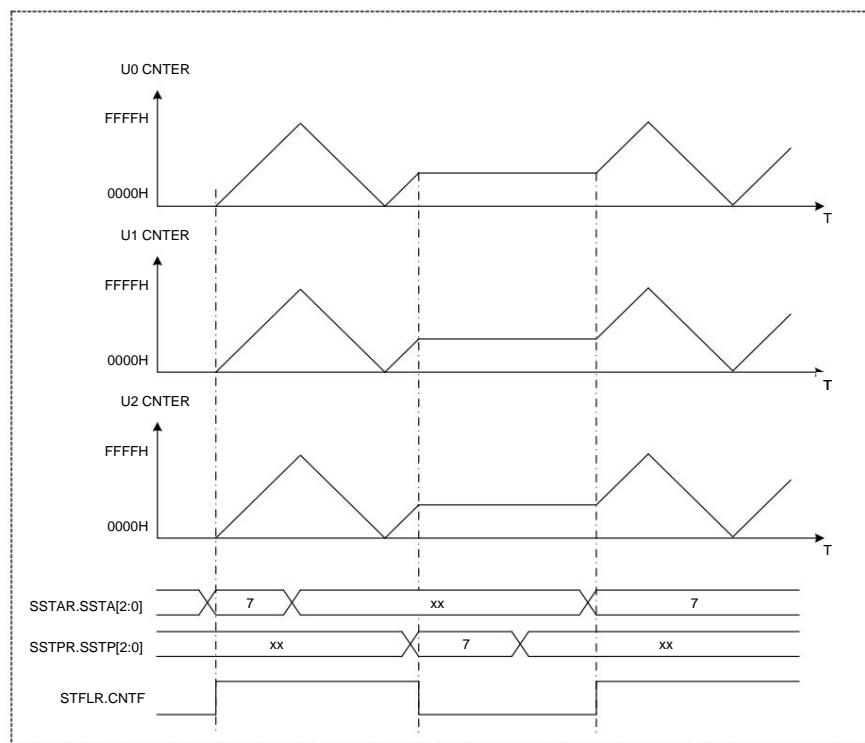


Figure 12-7 Software synchronization action

#### 12.2.5.2 Software Synchronization Stop

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

synchronization stops.

#### 12.2.5.3 Software synchronous clear

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

synchronous clear.

As shown in Figure 12-7, if SSTAR.SSTA0=SSTAR.SSTA1=SSTAR.SSTA2 of Timer4 is set, that is

The software synchronization startup of Timer4/5/6 can be realized.

The software synchronous action related registers (SSTAR, SSTPR, SCLRR) are a group of independent Timer4/5/6,

Register shared among all TIMs, each bit of this group of registers is only valid when writing 1, and writing 0 is invalid. reading



---

When reading the SSTAR register, the counter status of each timer will be read, and when reading SSTPR or SCLRR, it will read the counter status of each timer.

Read 0.

#### **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 for

Implement hardware synchronization between timers.

##### **12.2.6.1 Hardware synchronization startup**

Each Timer4/5/6 can choose to start the counter by hardware, just select the timer with the same hardware start condition.

Synchronous start is achieved when the start condition is valid. The specific hardware start condition is selected by the hardware start event register

(HSTAR) setting.

##### **12.2.6.2 Hardware Synchronization Stop**

Each Timer4/5/6 can choose to stop the counter by hardware, just select the timer with the same hardware stop condition

Synchronous stop is achieved when the stop condition is valid. The specific hardware stop condition is determined by the hardware stop event selection register

(HSTPR) setting.

##### **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.

Synchronous clear is achieved when the clear condition is active. The specific hardware clearing conditions are determined by the hardware clearing event selection register

(HCLRR) setting.

##### **12.2.6.4 Hardware Synchronous Capture Input**

Each Timer4/5/6 can choose to realize the capture input function by hardware, and choose the same capture input function condition.

A timer can synchronize the capture input when the capture input function condition is valid. Specific hardware capture input functions

The conditions are determined by the settings of the hardware capture event selection registers (HCPAR, HCPBR).

##### **12.2.6.5 Hardware synchronization count**

Timer4/5/6 can choose to use hardware input as CLOCK to count, choose the same hardware counting condition

The timer can achieve synchronous counting when the hardware count CLOCK is valid. The specific hardware counting conditions are incremented by the hardware

It is determined by the settings of the 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,

Clearing action. The start, stop, and clearing of the counter also need to be set separately.

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

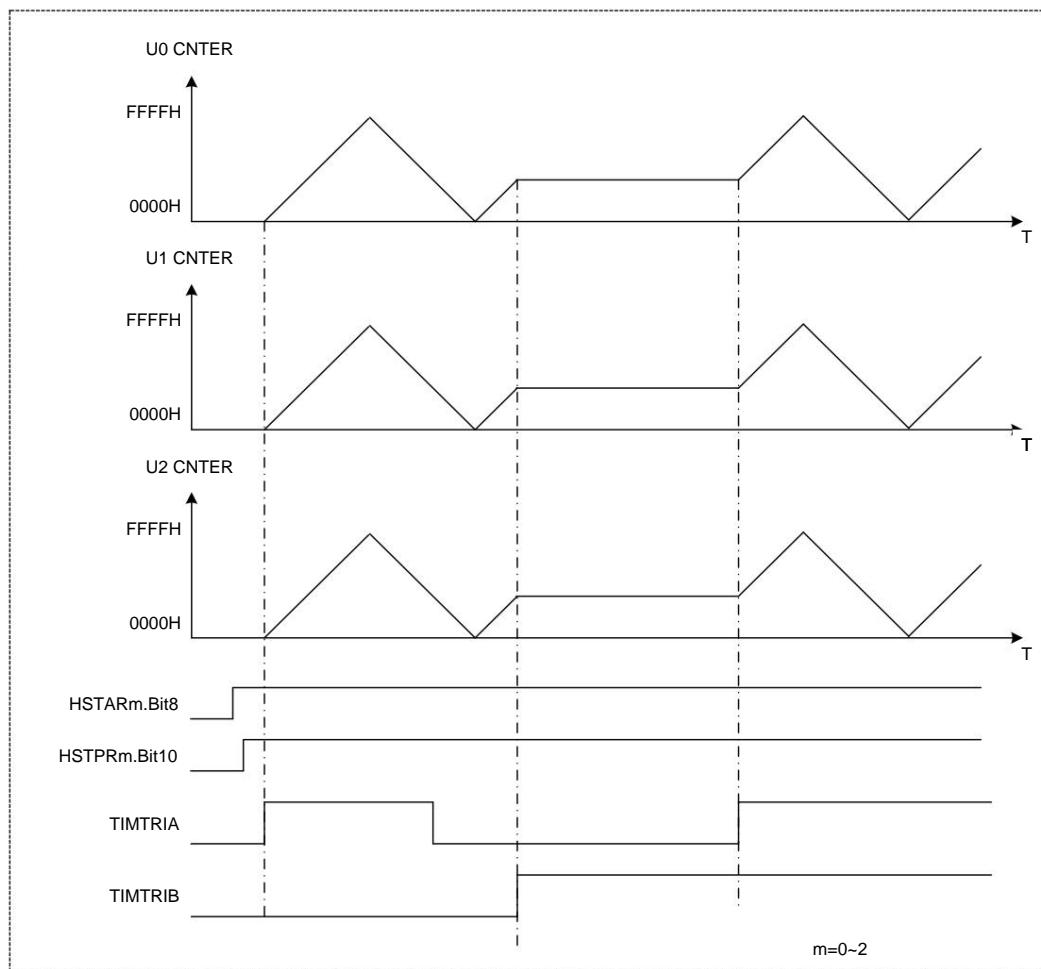


Figure 12-8 Hardware synchronization action

### 12.2.7 Cache function

Buffering action means that by setting the buffer control register (BCONR), at the buffer transfer time point, the selection occurs to next event:

- a. The value of the General Periodic Reference Buffer Register (PERBR) is automatically transferred to the General Periodic Reference Register (PERAR)
- b. The value of the general reference value buffer register (GCMCR, GCMDR) is automatically transferred to the general reference In the value register (GCMAR, GCMBR) (when comparing output)
- c. The value of the general reference value register (GCMAR, GCMBR) is automatically transferred to the general reference value buffer. Stored in registers (GCMCR, GCMDR) (when capturing input)

Figure 12-9 is a timing chart of the single-buffer method of the general-purpose comparison reference value register during the comparison output operation. from

As can be seen in the figure, changing the value of the general comparison reference value register (GCMAR) during the count can adjust the output

Duty cycle, the output period can be adjusted by changing the value of the general period reference value register (PERAR).

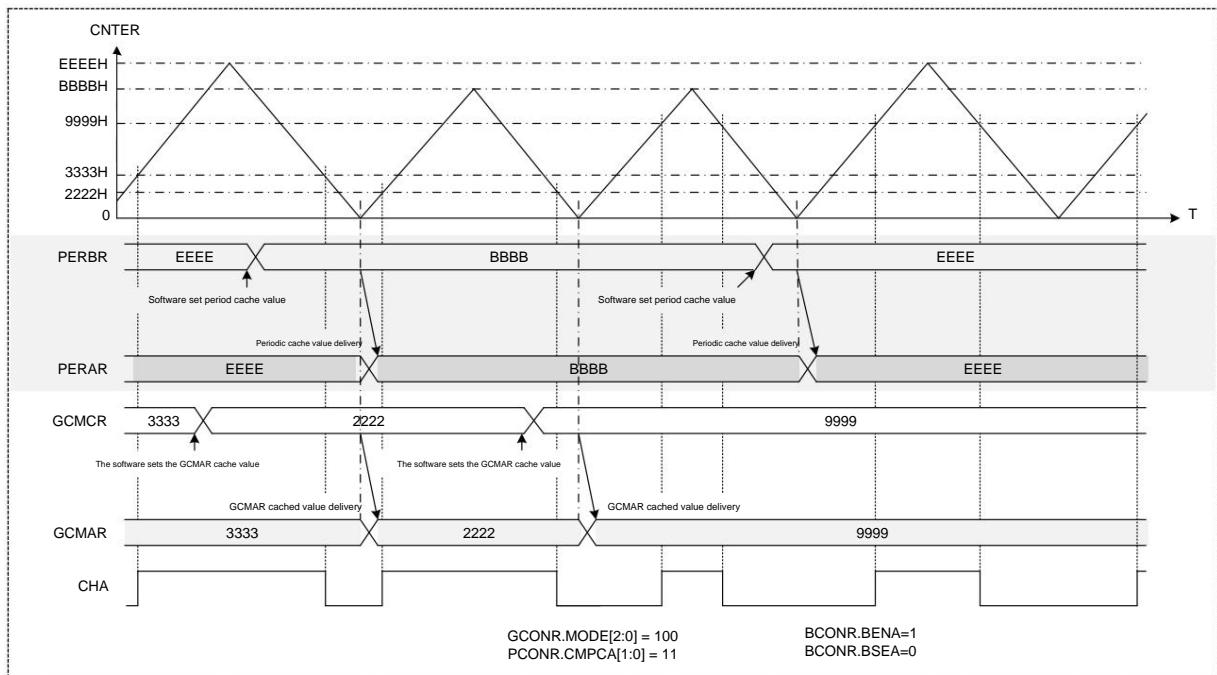


Figure 12-9 Comparison output timing in single buffer mode

**12.2.7.1 Cache transfer time point****12.2.7.2 Generic period reference value buffer transfer time point**

When the period reference value buffer transmission time point is the sawtooth wave, the count-up overflow point or the down-count underflow point, and the triangle wave

Count valley points.

**12.2.7.3 Generic benchmark value buffer transfer time point**

In sawtooth wave A mode, set BCONR.BENA=1 or BCONR.BNEB=1, the buffering action is valid. cache

The transfer occurs at the overflow or underflow point.

In triangle wave A mode, set BCONR.BENA=1 or BCONR.BNEB=1, the buffering action is valid. cache

Teleportation occurs at the count valley.

In triangle wave B mode, set BCONR.BENA=1 or BCONR.BNEB=1, the buffering action is valid. cache

Transfer occurs at count valleys and count peaks.

**12.2.7.4 Capture input value buffer transfer time point**

The transfer time point of the capture input action buffer is when the capture input action is taken.

**12.2.7.5 Buffer transfer during clear action**

In the sawtooth wave count mode or hardware count mode, if there is a clearing action during the normal comparison output action,

The general cycle reference value, general comparison reference value, etc. will be cached once according to the corresponding cache action setting status send.

### 12.2.8 General purpose PWM output

#### 12.2.8.1 PWM Spread Spectrum Output

In order to reduce the interference of PWM output to the outside, there is a spread spectrum configuration in the PWM output stage. Each PWM output

Period fine-tunes the phase of the PWM output.

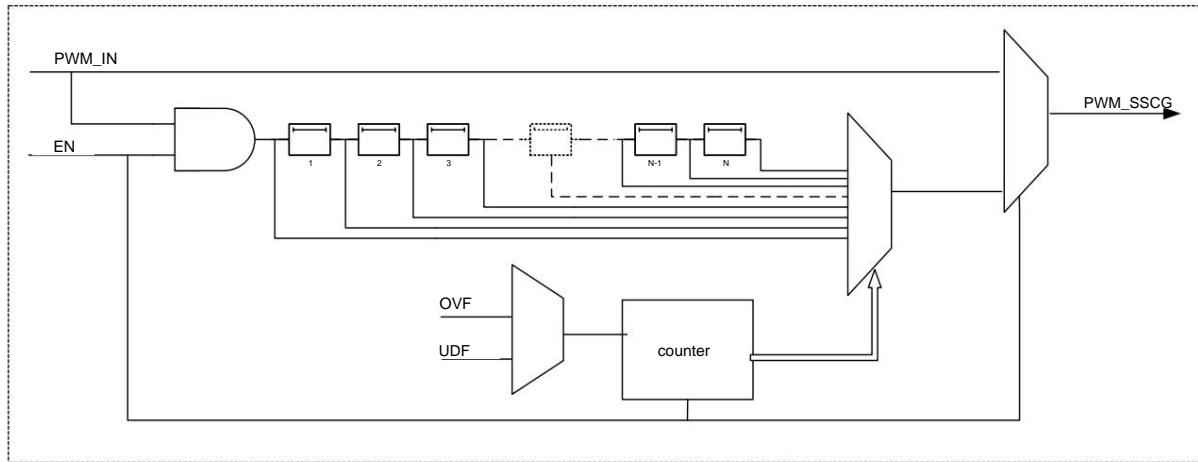


Figure 12-10 Schematic diagram of PWM spread spectrum output

#### 12.2.8.2 Independent PWM outputs

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

The CHA port of Timer6 outputs PWM wave.

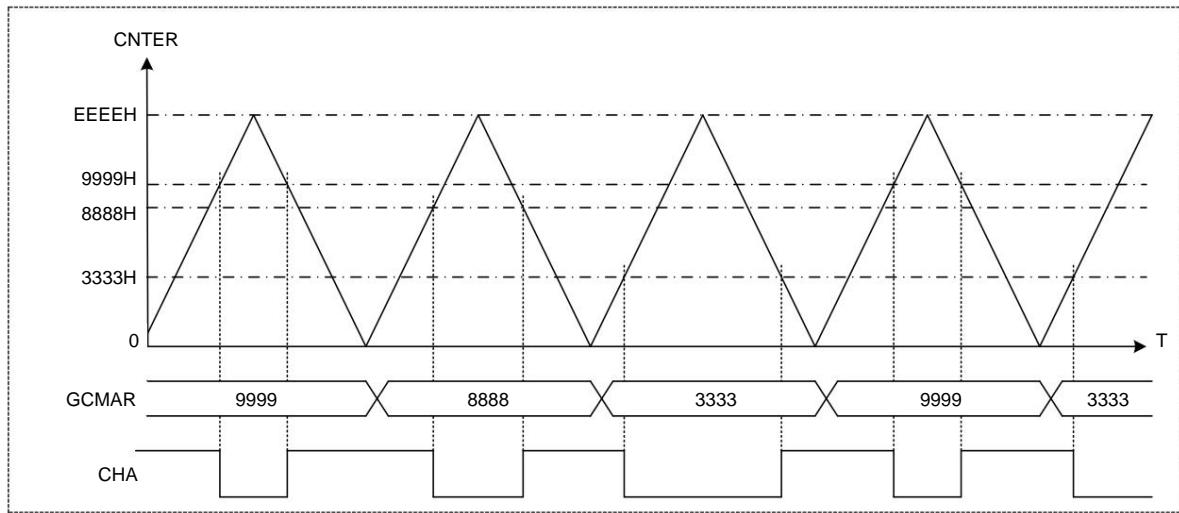


Figure 12-11 CHA output PWM wave

#### 12.2.8.3 Complementary PWM output

CHxA port and CHxB port can be combined to output complementary PWM waveform in different modes.

### 12.2.8.3.1 Software Setting GCMBR Complementary PWM Output

The software set GCMBR complementary PWM output refers to the sawtooth wave mode and the triangle wave A mode, the triangle wave B

In mode, the value of the general compare reference register (GCMBR) used for CHxB port waveform output is determined by the register

The register is set directly and has no direct relationship with the value of the general comparison reference value register (GCMAR).

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

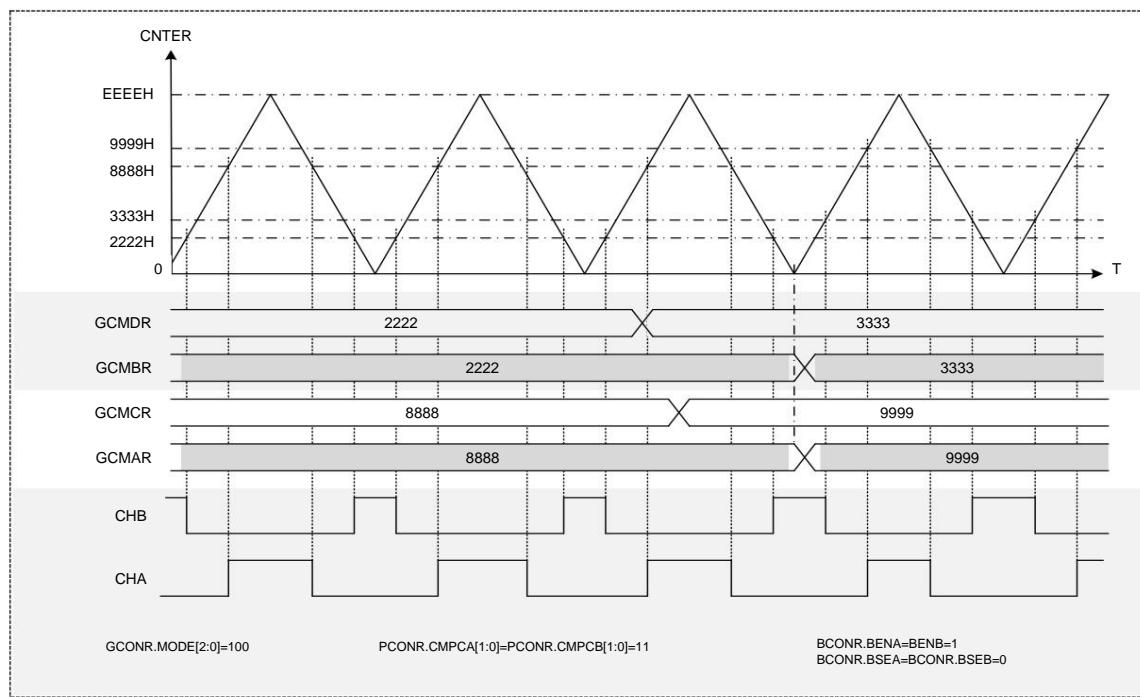


Figure 12-12 Software setting GCMBR complementary PWM wave output in triangle wave A mode

### 12.2.8.3.2 Hardware setting GCMBR complementary PWM output

The hardware setting of GCMBR complementary PWM output means that in triangular wave A mode and triangular wave B mode, it is used for

The value of the general comparison reference register (GCMBR) of the waveform output of the CHxB port is determined by the general comparison reference value.

It is determined by the value operation of the register (GCMAR) and the dead time reference value register (DTUAR, DTDAR).

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

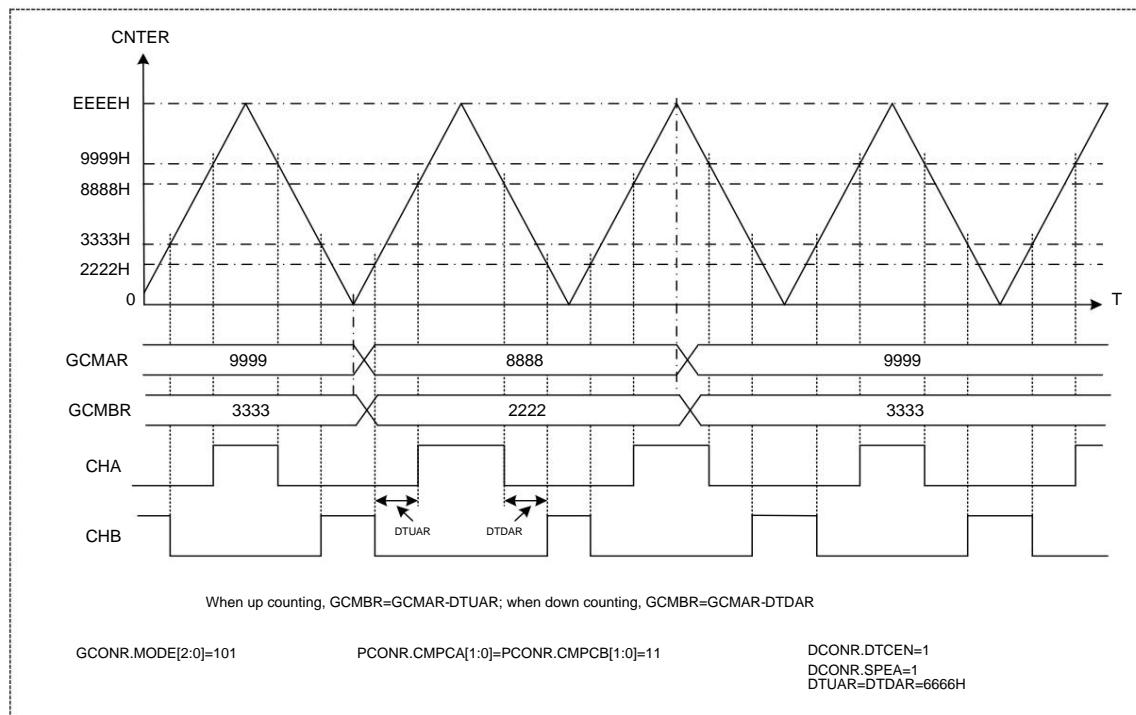


Figure 12-13 Hardware set GCMBR complementary PWM wave output in triangle wave B mode (symmetric dead zone)

#### 12.2.8.4 Polyphase PWM Output

The CHxA and CHxB ports of each timer can output 2-phase independent PWM waves or a set of complementary PWM

Multi-phase PWM wave output can be realized by combining multiple timers and synchronizing actions of software and hardware at the same time.

As shown in Figure 12-14, the combination of Timer4, Timer5 and Timer6 outputs 6-phase PWM wave; as shown in Figure 12-15, Timer4,

The combination of Timer5 and Timer6 outputs 3 complementary PWM waves.

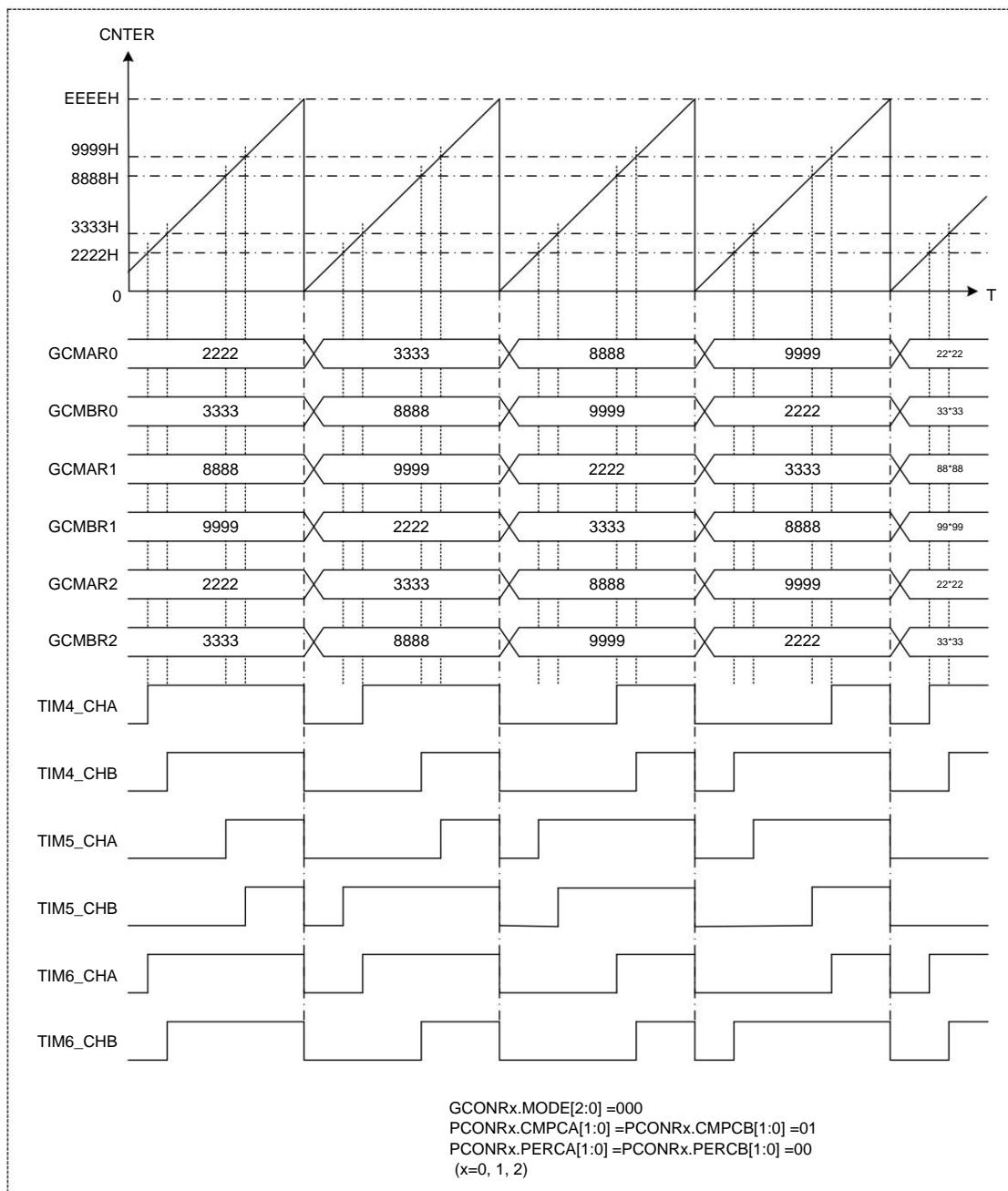


Figure 12-14 6-phase PWM wave

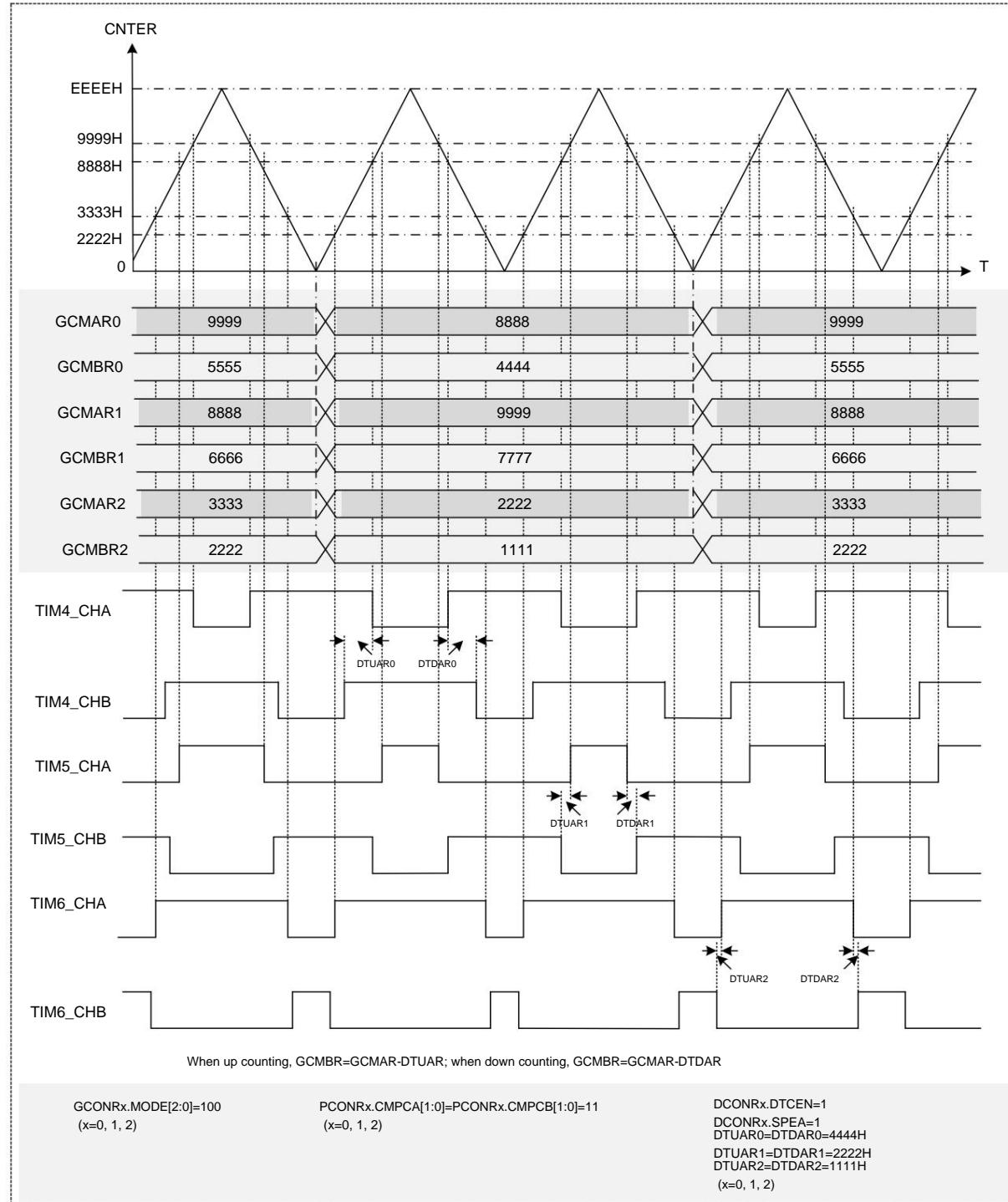


Figure 12-15 Three-phase complementary PWM wave output with dead time in triangular wave A mode

### 12.2.9 Quadrature Code Counting

Treat CHxA input as AIN input, CHxB input as BIN input, any of TIMTRIA-D

One input is regarded as the ZIN input, and the Advanced Timer can realize the quadrature encoding counting of three inputs.

The AIN and BIN of one timer act independently to realize the position counting mode; the AIN, BIN, and BIN of two timers

ZIN combined action can realize revolution counting mode, one timer is used for position counting, and one timer is used for public rotation.

Turn count.

In revolution counting mode, every combination of two timers (combination of timers 4 and 5, timer 4 is used as the position counting unit)

element, timer 5 is used as revolution counting unit) to realize position counting and revolution counting respectively.

The count conditions of AIN and BIN are set by setting the hardware increment event selection register (HCUPR) and the hardware decrement event.

The orthogonal relationship between CHxA and CHxB in the device selection register (HCDOR) is realized; the input action of ZIN is

Set the hardware clear event selection register (HCLRR) of the position unit to realize the position counter of the position counting unit

Clear, realize the revolution counting unit by setting the hardware increment event selection register (HCUPR) of the revolution unit.

The revolution counter counts.

#### 12.2.9.1 Position count mode

The quadrature coding position mode means that the basic counting function and phase difference counting function are realized according to the input of AIN and BIN.

and direction count function.

##### 12.2.9.1.1 Basic count

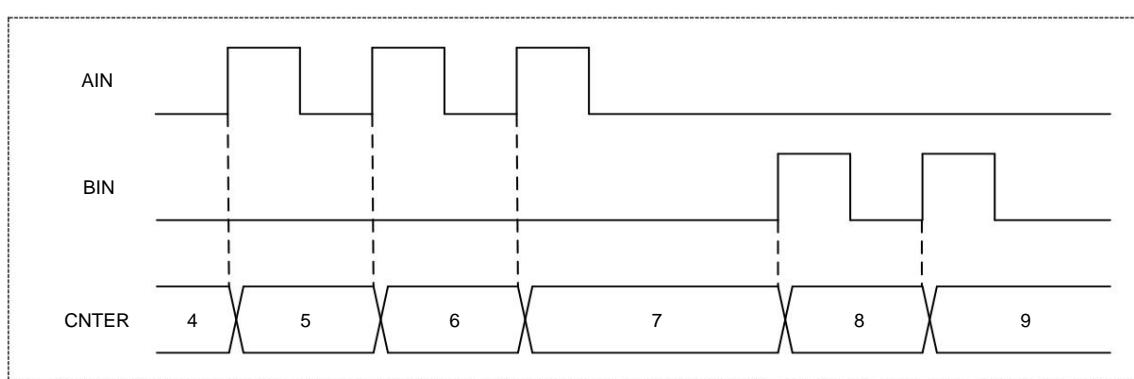


Figure 12-16 Basic counting action in position mode

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

### 12.2.9.1.2 Phase Difference Counting

Phase difference counting refers to counting according to the phase relationship between AIN and BIN. Depending on the settings, you can

Now 1 times count, 2 times count, 4 times count, etc., as shown in Figure 12-17 ~ Figure 12-19 below.

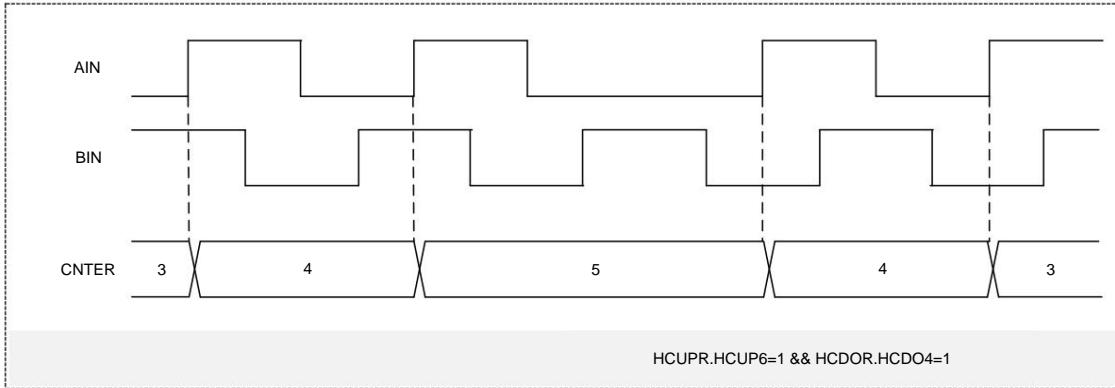


Figure 12-17 Phase difference count action setting in position mode (1 times)

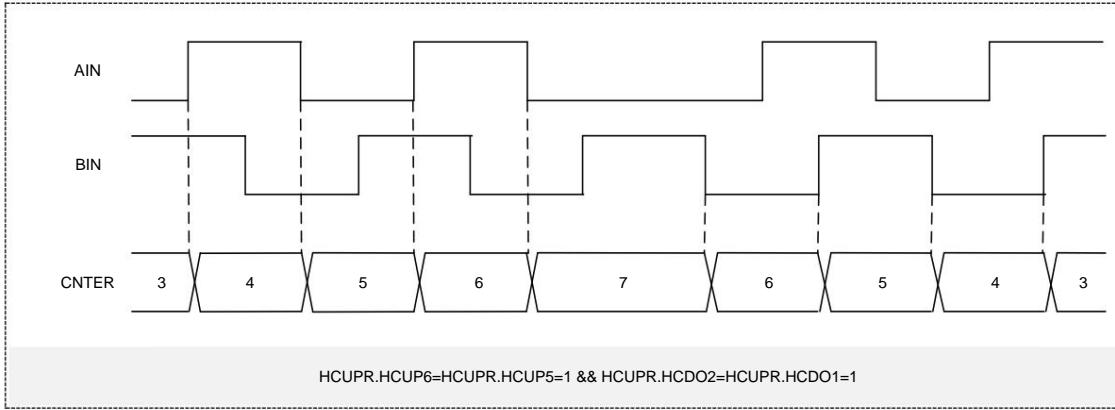


Figure 12-18 Phase difference count action setting in position mode (2 times)

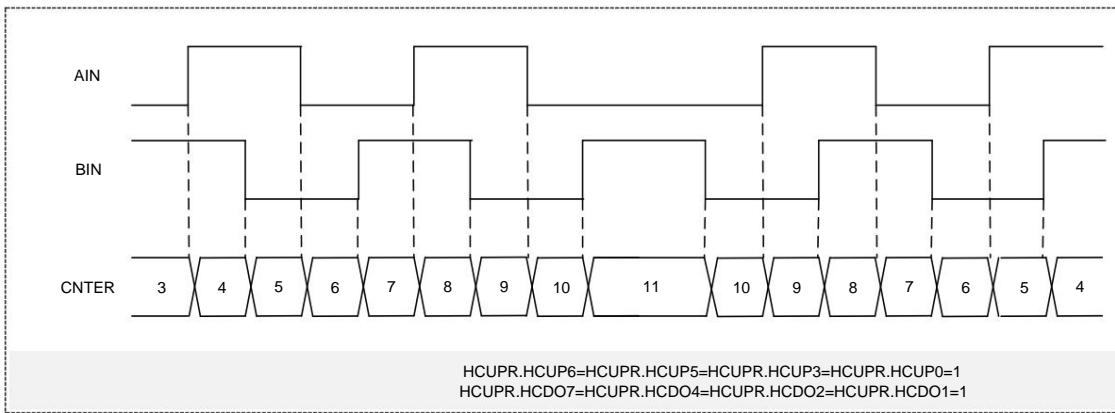


Figure 12-19 Phase difference count action setting in position mode (4 times)

### 12.2.9.1.3 Direction count

Direction counting means setting the input state of AIN as direction control, and using the input of BIN as clock counting,

As shown in Figure 12-20.

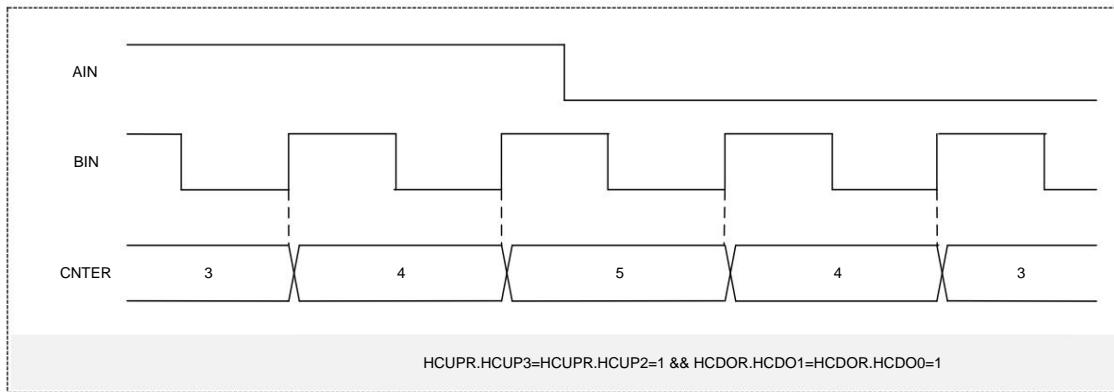


Figure 12-20 Direction counting action in position mode

### 12.2.9.2 Revolution Mode

Orthogonal coding revolution mode means adding the input event of ZIN based on the count of AIN and BIN to realize

Judgment on the number of revolutions, etc. In the revolution mode, according to the counting method of the revolution counter, the Z-phase counting function,

Position counter output count function and Z-phase count and position counter output mixed count function. using two

Advanced Timer implements this functionality.

### 12.2.9.2.12 Phase Count

Z-phase counting means that according to the input of ZIN, the revolution counting unit counts, and at the same time clears the position counting unit

count action.

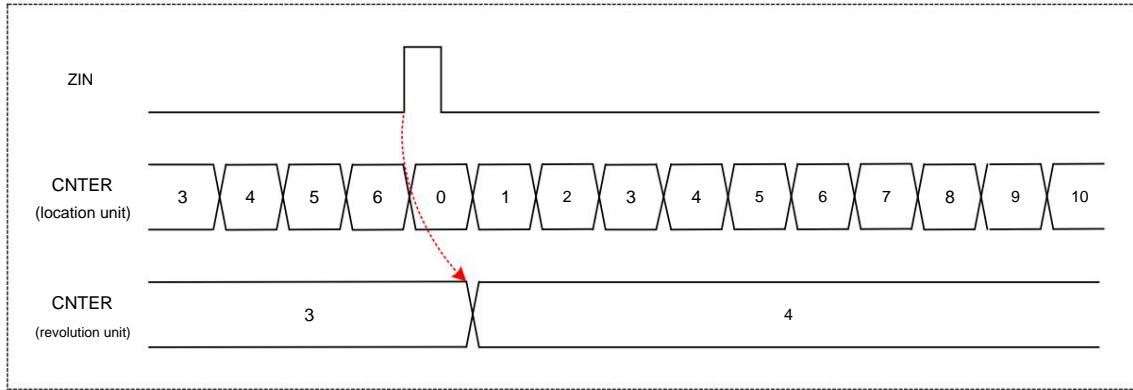


Figure 12-21 Z-phase counting action in revolution mode

#### 12.2.9.2.2 Position overflow count

The position overflow count means that when the position counting unit count overflows or underflows, an overflow event is generated, thereby triggering an overflow.

The counter of the revolution counting unit counts once (in this counting mode, the input of ZIN does not count the revolution).

The counting action of the counting unit and the 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, which can be realized.

The current position overflows the count. The hardware increment (decrement) event selection register (HCUPR or

The increment (decrement) event of HCDOR) selects 1 bit in Bit16:it19, and the AOS module sets the corresponding

The event source of the increment (decrement) event is the count overflow event of the position counting unit. For details, please refer to the AOS chapter.

Festival. As shown in Figure 12-22.

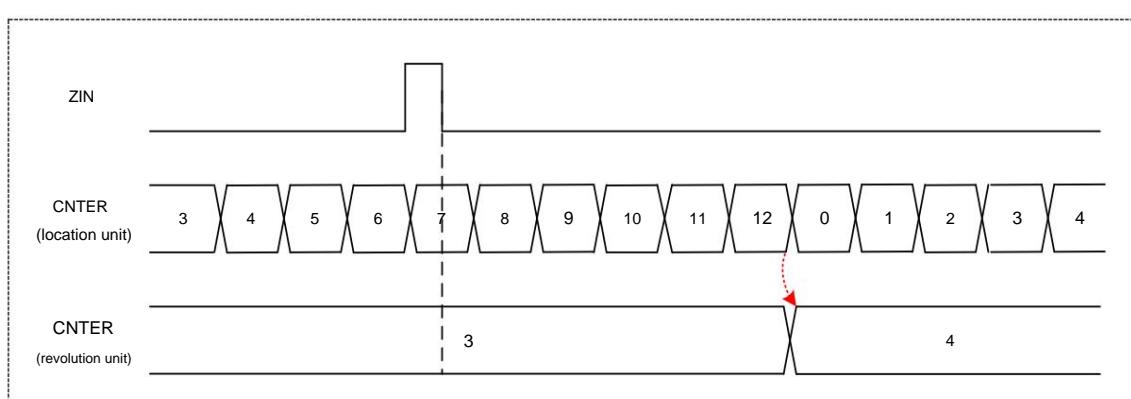


Figure 12-22 Position counter output counting action in revolution mode

#### 12.2.9.2.3 Mixed counts

Mixed counting refers to the counting action that combines the above two counting methods of Z-phase counting and position overflow counting.

The implementation method is also a combination of the above two counting methods. As shown in Figure 12-23.

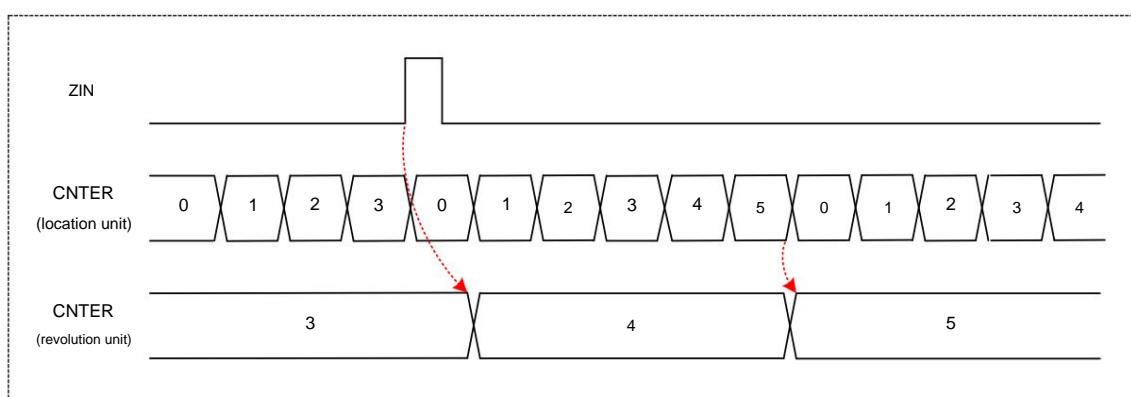


Figure 12-23 Mixed counting action of Z-phase counting and position counter output in revolution mode

#### 12.2.9.2.4 Z-phase action shield

In the Z-phase counting function or mixed counting function of revolution counting mode, the overflow of the position counter can be set.

In a few cycles (GCONR.ZMSK[0:1] setting) after the point or underflow point, the valid input of ZIN is masked,

Counting of the revolution counting unit and clearing of the position counting unit are not performed.

When GCONR.ZMSKPOS of the general control register (GCONR) of the position counting unit is 1, the position counting

The Z-phase masking function of the counting unit is enabled, and the number of periods of Z-phase masking is set by GCONR.ZMSK; revolution counting

When GCONR.ZMSKREV of the unit's general control register (GCONR) is 1, the revolution counting unit's

Z-phase muting function is enabled.

Figure 12-24 shows the 4 count cycles after the position counting unit count overflows when the revolution counting mode is mixed counting

When there is a ZIN phase input, 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 count.

The element is not cleared; the input of the ZIN phase that comes later will operate normally.

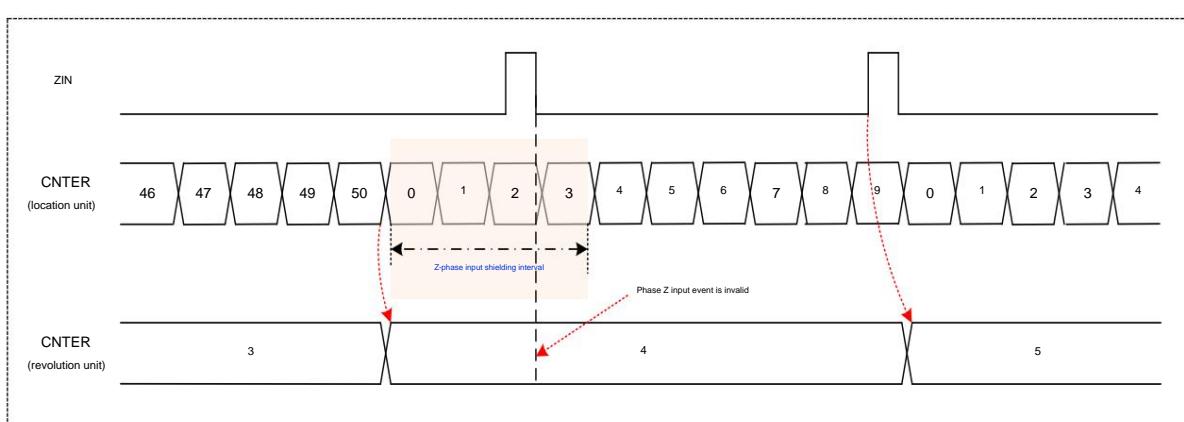


Figure 12-24 Revolution counting mode - mixed counting Z-phase shielding action example 1

Figure 12-25 is the third cycle after the position counting unit count overflows when the revolution counting mode is mixed counting,

The counting direction changes, and the set 4-cycle masking period becomes invalid at this time (the actual ZIN phase masking function

can be maintained for 3 cycles), start counting down. The ZIN phase is masked after a count underflow in the position counting unit

The function is turned on again and becomes invalid after 4 cycles. Input function of ZIN phase during ZIN phase masking

Invalid, that is, the revolution counting unit does not count, and the position counting unit does not clear; the input of the ZIN phase that comes later is normal

action.

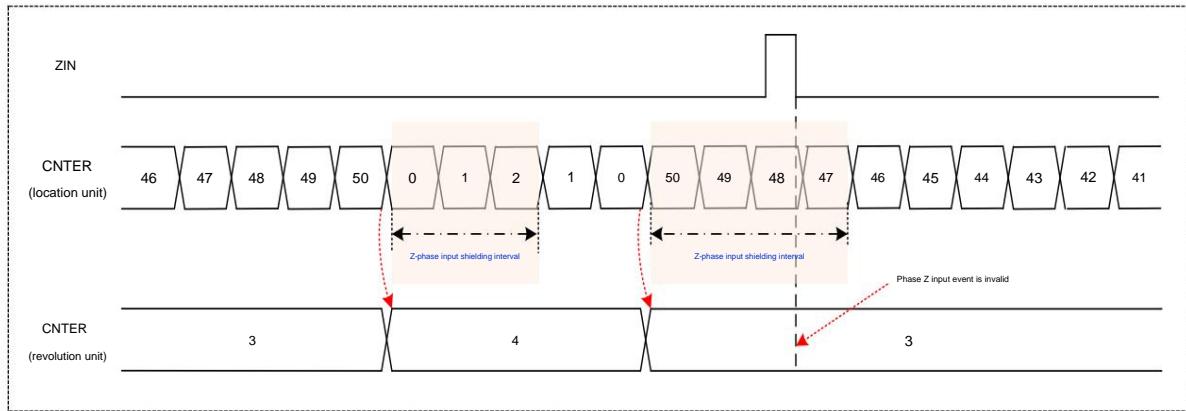


Figure 12-25 Revolution counting mode - mixed counting Z-phase shielding action example 2

### 12.2.10 Periodic interval response

The general-purpose comparison reference register (GCMAR~GCMDR) of Timer4/5/6 can be divided when the count compare matches.

Do not generate a dedicated valid request signal and send it to the AOS module for associated actions with other modules.

The request signal can generate a valid request signal every several cycles. Send by setting the valid period

The VPERR.PCNTS bit of the register (VPERR) is used 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 during its cycle, it will not

A valid request signal is output. Figure 12-26 shows an example of the operation of the periodic interval valid request signal.

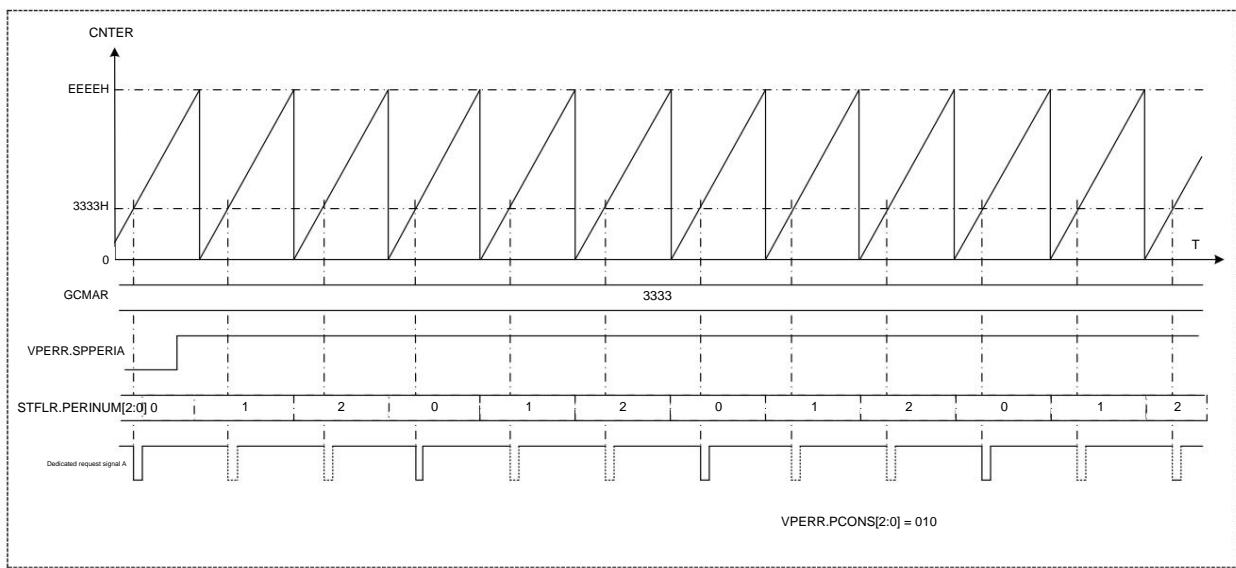


Figure 12-26 Periodic interval valid request signal action

#### 12.2.11 Protection Mechanisms

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

Advanced Timer has 4 common ports to output invalid event interface, these 4 interfaces are connected to brake control mode

4 groups of brake events output by the block. Abnormal condition events gated on each interface can be set from the brake control, when these

When an abnormal condition is detected on the interface, the control of the general-purpose PWM output can be realized.

During the normal output period of the port, if the brake event from the brake control is monitored, the output state of the port can be changed.

to a preset state. When the general PWM output port is in the abnormal event of brake control, the port status

Can become output high impedance, output low, or output high (PCONR.DISVALA

PCONR.DISVALB setting).

For example, if PCONR.DISSELA[1:0]=01&PCONR.DISVALA=01 is set, then at the CHxA terminal

During the normal output of the port, if a brake event occurs on the output invalid condition 1, the output on the CHxA port becomes high.

resistance state.



### 12.2.12 Interrupt Description

Timer4/5/6 each contain 3 types of interrupts totaling 9. They are 4 general-purpose count compare match interrupts (including 2 capture input interrupt), 2 count period match interrupts, 1 dead time error interrupt.

#### 12.2.12.1 Count Compare Match Interrupt

There are a total of 4 general comparison reference value registers (GCMAR-GCMDR), which can be compared with the count value to generate a ratio. match the valid signal. On a count compare match, the Status Flags Register (STFLR) in The STFLR.CMAF~STFLR.CMDF bits will be set to 1 respectively. At this time, if the interrupt control register is set (ICONR) The corresponding bit in ICONR.INTENA~ICONR.INTEND is 1 to enable the interrupt, then the corresponding interrupt requests are also triggered.

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

#### 12.2.12.2 Count Period Match Interrupt

Sawtooth wave counts up to the overflow point, sawtooth wave counts down to the underflow point, triangle wave counts to the valley point or triangle wave count When the peak is counted, the STFLR.OVFF or STFLR.UDFF bits in the Status Flags Register (STFLR) will be reset to Set to 1. At this time, if the ICONR.INTENOVF bit of the Interrupt Control Register (ICONR) is set with the If the ICONR.INTENUDF bit enables the interrupt, the count period match interrupt can be triggered at the corresponding time point.

#### 12.2.12.3 Dead Time Error Interrupt

Load the value of the dead-time reference registers (DTUAR, DTDAR) into the general compare reference registers (GCMBR), if the period limit is exceeded, a dead time error will occur, and the status flag register (STFLR) The STFLR.DTEF bit will be set to 1. At this time, if the interrupt control register (ICONR) is set If the interrupt is enabled by the ICONR.INTENDE bit, the dead time error interrupt will be triggered at that moment.

### 12.2.13 Brake protection

When invalid conditions 0~3 can be set, configure PCONR.DISVALA, PCONR.DISVALB. Invalid conditions are

The hardware will automatically change the port state to the preset state (high level, low level, high impedance state, maintain normal output) when it is in effect.

#### 12.2.13.1 Port Brake and Software Brake

After the port is controlled by polarity selection and is effectively enabled, it is digitally filtered and synchronized to generate the port brake flag;

The port brake flag is used as the invalid condition 3 of Advanced Timer. The port brake flag needs to be cleared by software.

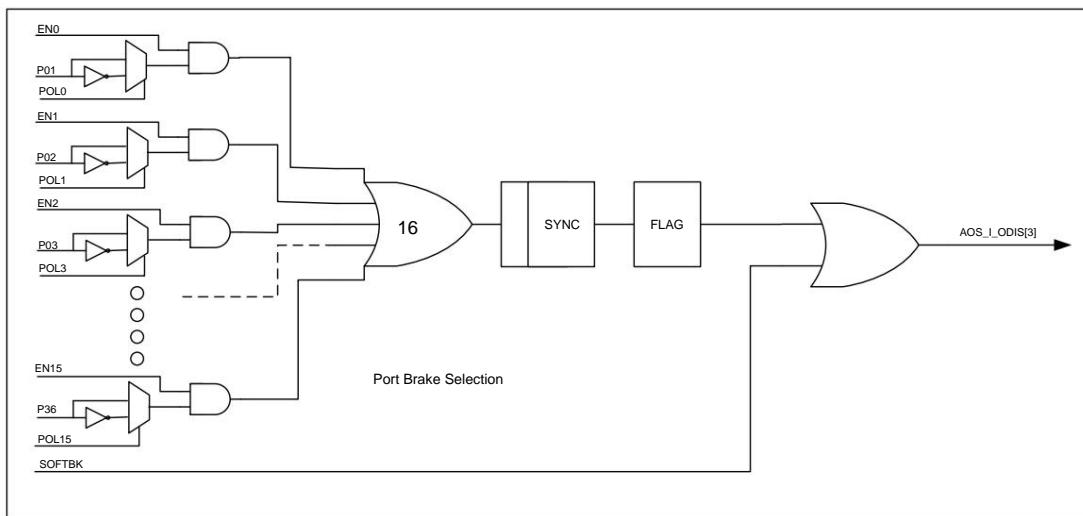


Figure 12-27 Schematic diagram of port brake and software brake

#### 12.2.13.2 Auto Brake 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 as

Inactive condition 2 of Advanced Timer controls PWM braking.

### 12.2.13.3 The output level is the same as the high and 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 is braked

Flag as invalid condition 1 of Advanced Timer. The same high and low brake flag needs to be cleared by software.

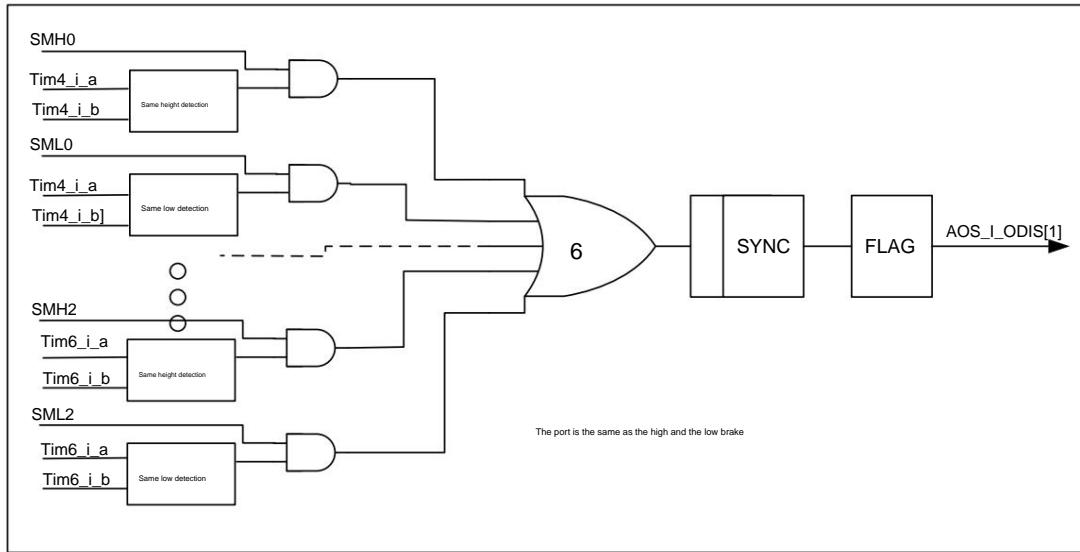


Figure 12-28 Schematic diagram of output same high and same low braking

### 12.2.13.4 VC brake

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

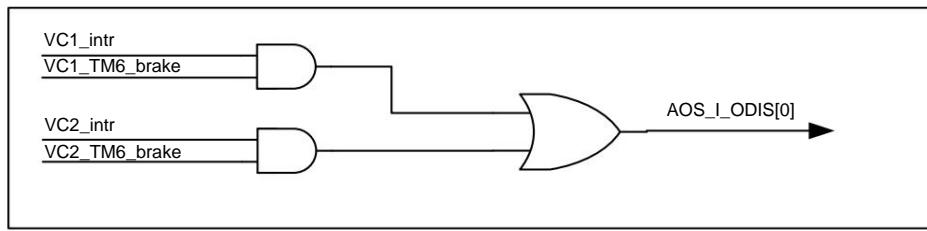


Figure 12-29 Schematic diagram of VC brake control

#### 12.2.14 Internal Interconnection

##### 12.2.14.1 Interrupt trigger output

Because one interrupt of Timer4/5/6 contains multiple interrupt sources. Control the interrupt that triggers the ADC and controls the AOS

The signal has separate control, you can choose different sources, you can choose overflow, underflow, 4 comparison matches, a total of 6

Any interrupt source of the TIMx's interrupt source is used as a trigger condition.

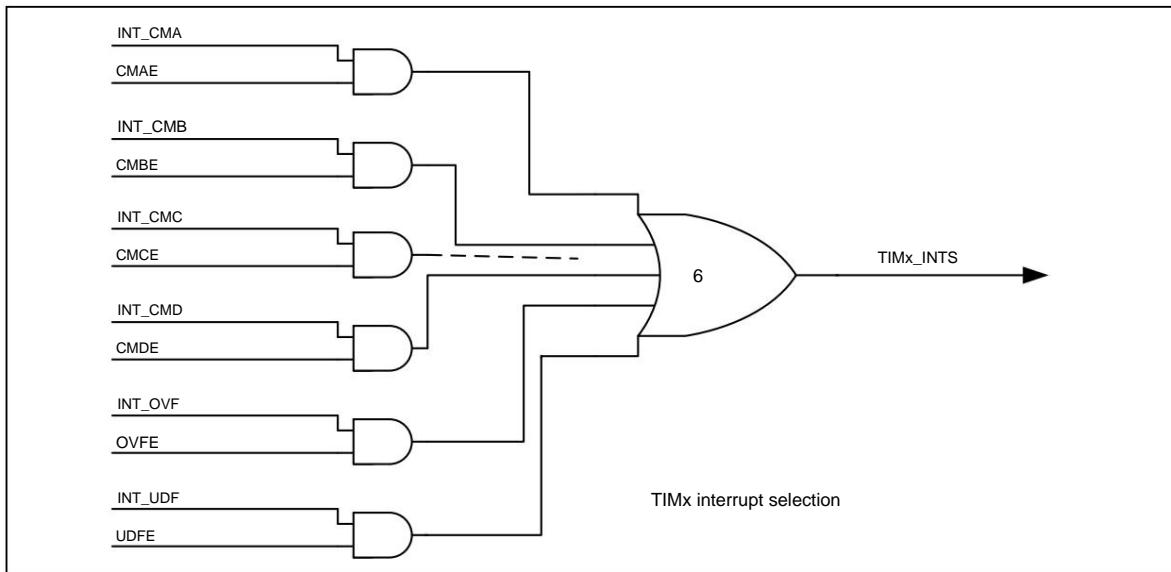


Figure 12-30 Timer4/5/6 Interrupt Selection

##### 12.2.14.2 AOS trigger

AOS is an internal signal of the system, which can trigger the opening of the counter of the Advanced Timer after selecting the control.

Start, stop, clear, add 1, subtract 1 and other functions. Advanced Timer has 4 AOS triggers, each trigger

Interrupt sources for different modules can be selected. The selected signal generates single pulse trigger input to Advanced Timer,

Controls the start, stop, and clearing of the Advanced Timer counter.

Timer4/5/6 uses registers to select different AOS\_i\_TRIG as its own trigger signal. if possible

Using the HSTAR register, an interrupt can be used to trigger the hardware start of the corresponding timer.

	AOS_i_trig0 AOS_i_trig1	AOS_i_trig2	\$selection	AOS_i_trig3
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

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

Table 12-3 AOS source selection

#### 12.2.14.3 Port trigger TRIGA-TRIGD

Port trigger can control the hardware start, stop, clear, capture, counter increment and decrement of Advanced Timer

Digital filtering function is optional, and the port can be configured as any port of the chip.

Selection control signals are independent control	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

Table 12-4 Port trigger selection



#### 12.2.14.4 Interconnection of compare output VC with Advanced Timer

The VC can be interconnected to the capture input of the Advanced Timer, and the edge of the VC output can be captured.

obtain;

#### 12.2.14.5 UART and Advanced Timer interconnection

UARTx\_RX / LPUART\_RX can be interconnected to BaseTimer, LPTimer, PCA and

Advanced Timer. The automatic identification of the baud rate can be realized by software.

The UART selection control register is in the port control register GPIO\_CTRL3, and the VC output control register is in

VC Control Module.

## 12.3 Register Description

CH0 base address 0x40003000

CH1 base address 0x40003400

CH2 base address 0x40003800

register	Offset address	description
TIMx_CNTER	0x000	General Count Reference Register
TIMx_PERAR	0x004	General Periodic Reference Register
TIMx_PERBR	0x008	General Periodic Reference Cache Register
TIMx_GCMAR	0x010	General Purpose Compare A Reference Register
TIMx_GCMBR	0x014	General Purpose Compare B Reference Register
TIMx_GCMCR	0x018	General Purpose Compare C Reference Register
TIMx_GCMDR	0x01C	General Purpose Compare D Reference 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 Time 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 Decrement Event Select Register
TIMx_HCDOR	0x08C	Hardware Decrement Event Select Register
TIMx_IFR	0x100	Interrupt Flag Register
TIMx_ICLR	0x104	interrupt clear register
TIMx_CR	0x108	Spread Spectrum and Interrupt Trigger Select 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
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 Startup Register
TIMx_SSTPR	0x3F8	Software Sync Stop Register
TIMx_SCLR	0x3FC	Software synchronous clear register

Table 12-5 Advanced Timer Register List



### 12.3.1 General Count Reference Register (TIMx\_CNTER)

Address offset: 0x000

Reset value: 0x0000 0000

31	30 29 28	27 26 25 24	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	Function description
31:16	Reserved	-
15:0	CNT[15:0]	Current counter count value

### 12.3.2 General Periodic 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	Function description
31:16	Reserved	-
15:0	PERA[15:0]	count period value, set the count period value of each round of counting

### 12.3.3 General Periodic Buffer Register (TIMx\_PERBR)

Address offset: 0x008

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	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	Function description
31:16	Reserved	-
15:0	PERB[15:0]	Cache count period value, cache value of count period

### 12.3.4 Generic Compare Reference Registers (TIMx\_GCMAR-GCMDR)

Address offsets: 0x0010, 0x0014, 0x0018, 0x001C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	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	Function description
31:16	Reserved	-
15:0	GCMA-D [15:0]	Count comparison reference value, comparison reference value setting, match 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	twenty four	twenty three	twenty two	twenty one	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	Function 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	22	21	20	19	18	17	16	
Reserved									ZMSK	ZMSK	ZMSK		
									R/W	R/W	REV 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 mark		Function description												
31:20	Reserved	-												
19:18	ZMSK Z-phase input mask period number	<p>Quadrature-encoded Z-phase input masked count period value</p> <p>00: Z-phase input shielding function is invalid</p> <p>01: Z-phase input is masked within 4 count cycles after position count overflow or underflow</p> <p>10: Z-phase input is masked within 8 count cycles after position count overflow or underflow</p> <p>11: Phase Z input is masked within 16 count cycles after position count overflow or underflow</p>												
17	ZMSKPOS Z-phase input position counter selection	<p>0: The timer is used as a position counter when the Z phase is input, and the position counter is cleared during the mask period. normal action</p> <p>1: The timer is used as a position counter when the Z phase is input, and the position counter is cleared during the mask period. Hidden</p>												
16	ZMSKREV Z-phase input revolution counter selection	<p>0: When the Z phase is input, the timer acts as a revolution counter, and the revolution counter counts during the mask period. normal action</p> <p>1: When the Z phase is input, the timer is used as a revolution counter, and the revolution counter counts during the mask period. Hidden</p>												
15:9	Reserved	-												
8	DIR	<p>counting direction</p> <p>0: count down; 1: count up</p>												
7	Reserved	-												
6:4	CKDIV count clock selection	<table border="0"> <tr> <td>000: PCLK0</td> <td>001: PCLK0/2</td> <td>010: PCLK0/4</td> <td>011: PCLK0/8</td> </tr> <tr> <td>100:</td> <td>101: PCLK0/64</td> <td>110: PCLK0/256</td> <td>111: PCLK0/1024</td> </tr> <tr> <td>PCLK0/16</td> <td></td> <td></td> <td></td> </tr> </table>	000: PCLK0	001: PCLK0/2	010: PCLK0/4	011: PCLK0/8	100:	101: PCLK0/64	110: PCLK0/256	111: PCLK0/1024	PCLK0/16			
000: PCLK0	001: PCLK0/2	010: PCLK0/4	011: PCLK0/8											
100:	101: PCLK0/64	110: PCLK0/256	111: PCLK0/1024											
PCLK0/16														
3:1	MODE count mode	<p>000: sawtooth wave A mode 100: triangular wave A mode 101: triangular wave B mode</p>												



		Please do not set other values
0	START Counter start	0: Counter is off; 1: Counter is started Note: This bit automatically changes to 0 when a software stop condition or hardware stop condition is active



### 12.3.7 Interrupt Control Register (TIMx\_ICONR)

Address offset: 0x054

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24	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	INTEN	INTEN	Reserved	INTEN	INTEN	INTEN	INTEN
		R/W/R/W/UDF	WR/UDF	OVF		R/W/R/W/UDF	B	R/WA	

bit	mark	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: The underflow occurs during the sawtooth wave or the count reaches the valley point during the triangle wave, the interrupt is invalid 1: The underflow occurs in the sawtooth wave or the count reaches the valley point in the triangular wave, the interrupt is enabled
6	INTENOVF Overflow	Interrupt Enable 0: When the overflow occurs in the sawtooth wave or the count reaches the peak point in the triangular wave, the interrupt is invalid 1: When the overflow occurs in the sawtooth wave or the count reaches the peak point in the 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 disabled. effect 1: When the GCMBR register equals the count value, or when a capture input event occurs, this interrupt can
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 disabled. effect 1: When the GCMAR register equals the count value, or when a capture input event occurs, this interrupt can



### 12.3.8 Port Control Register (TIMx\_PCONR)

Address offset: 0x058

Reset value: 0x0000 0000

31	30	29	28	27	26	25	Twenty four	Twenty five	22	21	20	19	18	17	16
Reserved	DISVALB	DISSELB OUT	ENB		PERCB	CMPCB	STASTPS	STPC	STAC	CAP					
	R/W	R/W	R/W		R/W	R/W	R/W	R/W	R/W	WR	WR	WR	WR	WR	WR

bit	mark	Function
31:29	Reserved	-
28:27	DISVALB	<p>CHxB output state control</p> <p>00: When the selected condition in the forced output invalid condition 0~3 is satisfied, the CHxB port will output normally</p> <p>01: When the selected condition in the forced output invalid condition 0~3 is satisfied, the CHxB port outputs a high-impedance state</p> <p>10: When the selected condition in the forced output invalid condition 0~3 is established, the CHxB port outputs a low level</p> <p>11: When the selected condition in the forced output invalid condition 0~3 is established, the CHxB port outputs a high level</p>
26:25	DISSELB	<p>Forced output disable condition selection B 00:</p> <p>Select forced output disable condition 0; 01: Select forced output disable condition 1</p> <p>10: Select forced output invalid condition 2; 11: Select forced output invalid condition 3</p>
twenty four	OUTENB	<p>output enable B</p> <p>0: The CHxB port output is invalid when the Advanced Timer function is used</p> <p>1: The CHxB port output is valid when the Advanced Timer function is enabled</p>
23:22	PERCB	<p>Port state setting B when period value matches</p> <p>00: When the counter count value is equal to the period value, the CHxB port output remains low level</p> <p>01: When the counter count value is equal to the period value, the CHxB port output is set to high level</p> <p>10: When the counter count value is equal to the period value, the CHxB port output is set to the previous state</p> <p>11: When the counter count value is equal to the period value, the CHxB port output is set to the inverted level</p>
21:20	CMPCB	<p>Port state setting B when the comparison value matches</p> <p>00: When the counter count value is equal to GCMBR, the CHxB port output remains low level</p> <p>01: When the counter count value is equal to GCMBR, the CHxB port output is set to high level</p> <p>10: When the counter count value is equal to GCMBR, the CHxB port output is set to the previous state</p> <p>11: When the counter count value is equal to GCMBR, the CHxB port output is set to the inverted level</p>
19	STASTPSB	<p>count start stop port state select B</p> <p>0: When counting starts or stops, the CHxB port output is determined by STACB, STPCB</p> <p>1: When counting starts or stops, the CHxB port output is set to the previous state</p>



18	STPCB	Count stop port status setting B 0: When counting stops, the CHxB port output is set to low level 1: When counting stops, the CHxB port output is set to high level
17	STACB	Counting start port state setting B 0: When counting starts, the CHxB port output is set to low level 1: When counting starts, the CHxB port output is set to high level
16	CAPCB	Function mode selection B 0: Compare output function; 1: Capture input function
15:13	Reserved	-
12:11	DISVALA	CHxA output state control 00: When the selected condition in the forced output invalid condition 0~3 is satisfied, the CHxA port will output normally 01: When the selected condition in the forced output invalid condition 0~3 is established, the CHxA port outputs a high-impedance state 10: When the selected condition among the forced output invalid conditions 0~3 is established, the CHxA port outputs a low level 11: When the selected condition in the forced output invalid condition 0~3 is established, the CHxA port outputs a high level
10:9	DISSELLA Force Output	Invalid Condition Select 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: The CHxA port output is invalid when the Advanced Timer function is used 1: The CHxA port output is valid when the Advanced Timer function is enabled
7:6	PERCA	Port state setting A when the period value matches 00: When the counter count value is equal to the period value, the CHxA port output remains low level 01: When the counter count value 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 counter count value is equal to the period value, the CHxA port output is set to the inverted level
5:4	Port state setting A when CMPCA compare value matches	00: When the counter count value is equal to GCMAR, the CHxA port output remains low level 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 counter count value is equal to GCMAR, the CHxA port output is set to the inverted level
3	STASTPSA count start	stop port state select A 0: When counting starts or stops, the CHxA port output is determined by STACA, STPCA 1: When counting starts or stops, the CHxA port output is set to the previous state  Note: The counting start here refers to the initial counting or stopping and restarting; the counting stop refers to the initial stop or the counting starts and then stops.
2	STPCA	Count stop port status setting A 0: When counting stops, the CHxA port output is set to low level; 1: When counting stops, the CHxA port output is set to high level
1	STACA	Counting start port state setting A 0: When counting starts, the CHxA port output is set to low level 1: When counting starts, the CHxA port output is set to high level
0	CAPCA function mode	selection A



		0: Compare 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	21	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 mark		Function
31:9	Reserved	-
8	BENP period value buffer transfer	0: Invalid cache delivery 1: Buffer transfer enable (PERBR->PERAR)
7:3	Reserved	-
2	BENB Common Compare Value Buffer Transfer B	0: Invalid cache delivery 1: Buffer transfer enable When comparing output functions: (GCMDR->GCMBR); when capturing input functions: (GCMBR->GCMDR)
1	Reserved	-
0	BENA Generic Comparison Value Buffer Transfer A	0: Invalid cache delivery 1: Buffer transfer enable When comparing output functions: (GCMCR->GCMAR); when capturing input functions: (GCMAR->GCMCR)



### 12.3.10 Dead Time Control Register (TIMx\_DCONR)

Address offset: 0x060

Reset value: 0x0000 0000

31	30	29	28	27	26	25		23	22		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 mark		Function
31:9	Reserved	-
8	SEPA separation settings	0: DTUAR and DTDAR are set separately 1: The value of DTDAR and DTUAR are automatically equal
7:1	Reserved	-
0	DTCEN dead time function	0: Dead zone function is invalid 1: 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	twenty four	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					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								NOFICKGB	NOFI	Res.	NOFICKGA	NOFI			
								R/W	ENGAB		R/W				



3	Reserved	-
2:1	NOFICKGA[1:0] CHxIA port filter sampling reference clock selection 00: PCLK0 PCLK0/64 01: PCLK0/4 10: PCLK0/16 11:	
0	NOFIENGA	CHxIA port capture input filter enable, 0 is invalid; 1 enables

Notice:

- The 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				R/W			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								GE PERI	GE PERIC	GE PERI	GE PERI				
R/W/R/W/R/W/A B															

bit mark		Function
31:21	Reserved	-
20:18	PCNTS active period selection	<p>000: The valid period selection function is invalid</p> <p>001: Valid every 1 cycle</p> <p>010: Valid every 2 cycles</p> <p>011: Valid every 3 cycles</p> <p>100: Valid every 4 cycles</p> <p>101: Valid every 5 cycles</p> <p>110: valid every 6 cycles</p> <p>111: valid every 7 cycles</p>
17:16	PCNTE valid period count condition selection	<p>00: The valid period selection function is invalid</p> <p>01: The upper and lower overflow points of the sawtooth wave count or the triangular wave trough as the counting condition</p> <p>10: The upper and lower overflow points of the sawtooth wave count or the triangular wave peak as the counting condition</p> <p>11: The upper and lower overflow points of the sawtooth wave count or the trough and peak of the triangular wave as the counting conditions</p>
15:4	Reserved	-
3	GEPERID General signal valid period selection D	0: The valid period selection function is invalid; 1: The valid period selection function is enabled
2	GEPERIC Generic signal valid period selection C	0: The valid period selection function is invalid; 1: The valid period selection function is enabled
1	GEPERIB Common signal valid period selection B	0: The valid period selection function is invalid; 1: The valid period selection function is enabled
0	GEPERIA Common signal valid period selection A	0: The valid period selection function is invalid; 1: The 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		20	19	18	17	
DIR R	Reserved							VPERNUM		Reserved					
	R														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DTEF	UDFF	OVFF		Reserved	CMD	CMC	CMB	CMA
							R/WR	WR/W				R/WR	WR/W	F	

bit	mark	Function
31	DIRF	counting direction 0: count down 1: count up
30:24	Reserved	-
23:21	VPERNUM 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: No underflow of sawtooth wave or triangle wave count to the valley point 1: The sawtooth wave underflow occurs or the triangle wave counts to the valley point
6	OVFF	overflow match 0: No sawtooth wave overflow or triangle wave count to the peak point 1: The sawtooth wave overflow occurs or the triangle wave counts to the peak point
5:4	Reserved	-
3	CMDF	count match D 0: The value of the GCMDR register is not equal to the count value; 1: The value of the GCMDR register is the same as the count value value equal
2	CMCF	count match C 0: The value of the GCMCR register is not equal to the count value; 1: The value of the GCMCR register and the count value equal
1	CMBF	count match B 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 the CHxB capture completion action occurs
0	CMAF	count match A 0: The value of the GCMAR register is not equal to the count value, and the CHxA capture completion action has not occurred



		1: The value of the GCMAR register is equal to the count value, or the CHxA capture completion 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
HSTA 15	HSTA 14	HSTA 13	HSTA 12	HSTA 11	HSTA 10	HSTA 9	HSTA 8	HSTA 7	HSTA 6	HSTA 5	HSTA 4
R/WR/W		R/WR/W	R/WR/W	R/WR/W	R/WR/W	R/WR/W	R/WR/W	R/WR/W	R/WR/W	R/WR/W	R/WR/W
3	2	1	0								
											R/W

Bit tag function		
31	STARTS hardware start enable	0: Hardware boot invalid 1: Hardware boot enabled  Note: When hardware startup is valid, SSTAR setting is invalid
30:16	Reserved -	
15	HSTA15 Hardware Start Condition 15: TIMTRID port upsampled to falling edge	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
14	HSTA14 Hardware Start Condition 14: TIMTRID port up sampled to rising edge	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
13	HSTA13 Hardware Start Condition 13: TIMTRIC port upsampled to falling edge	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
12	HSTA12 Hardware Start Condition 12: TIMTRIC port up sampled to rising edge	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
11	HSTA11 Hardware Start Condition 11: TIMTRIB port upsampled to falling edge	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
10	HSTA10 Hardware Start Condition 10: TIMTRIB port is sampled to a rising edge	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
9	HSTA9 Hardware Start Condition 9: TIMTRIA port upsampled to falling edge	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid



8	HSTA8 Hardware Start Condition 8: TIMTRIA port upsampled to rising edge	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
7	HSTA7 Hardware Start Condition 7: Falling Edge on CHxB Port	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
6	HSTA6 Hardware Start Condition 6: CHxB port up sampled to rising edge	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
5	HSTA5 Hardware Start Condition 5: Falling Edge on CHxA Port	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
4	HSTA4 Hardware Start Condition 4: CHxA port up sampled to rising edge	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
3	HSTA3 hardware start condition 3: Event trigger 3 from AOS is valid	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
2	HSTA2 hardware start condition 2: Event trigger 2 from AOS is valid	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
1	HSTA1 Hardware Start Condition 1: Event Trigger 1 from AOS is valid	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid
0	HSTA0 Hardware Start Condition 0: Event Trigger 0 from AOS is valid	0: When the condition is matched, the hardware startup is invalid 1: When the condition is matched, the hardware startup is valid



### 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
HSTP 15	HSTA 14	HSTP 13	HSTP 12	HSTP 11	HSTP 10	HSTP 9	HSTP 8	HSTP 7	HSTP 6	HSTP 5	HSTP 4	HSTP 3	HSTP 2	HSTP 1	HSTP 0
R/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	W

Bit tag function		
31	STOPs hardware stop enable	0: Hardware stop is invalid 1: Hardware stop valid  Note: When the hardware stop is valid, the software stop setting is invalid
30:16	Reserved -	
15	HSTP15 Hardware Stop Condition 15: TIMTRID port upsampled to falling edge	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
14	HSTP14 Hardware Stop Condition 14: Rising edge on TIMTRID port sampled	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
13	HSTP13 Hardware Stop Condition 13: TIMTRIC port upsampled to falling edge	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
12	HSTP12 Hardware Stop Condition 12: TIMTRIC port up sampled to rising edge	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
11	HSTP11 Hardware Stop Condition 11: TIMTRIB port upsampled to falling edge	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
10	HSTP10 Hardware Stop Condition 10: TIMTRIB port up sampled to rising edge	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
9	HSTP9 Hardware Stop Condition 9: TIMTRIA port upsampled to falling edge	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid



8	HSTP8 hardware stop condition 8: rising edge on TIMTRIA port	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
7	HSTP7 Hardware Stop Condition 7: Falling Edge on CHxB Port	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
6	HSTP6 Hardware Stop Condition 6: CHxB port up sampled to rising edge	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
5	HSTP5 Hardware Stop Condition 5: Falling Edge on CHxA Port	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
4	HSTP4 Hardware Stop Condition 4: CHxA port upsampling to rising edge	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
3	HSTP3 hardware stop condition 3: Event trigger 3 from AOS is valid	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
2	HSTP2 Hardware Stop Condition 2: Event Trigger 2 from AOS is valid	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
1	HSTP1 Hardware Stop Condition 1: Event Trigger 1 from AOS is valid	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid
0	HSTP0 Hardware Stop Condition 0: Event Trigger 0 from AOS is valid	0: When the condition is matched, the hardware stop is invalid 1: When the condition is matched, the hardware stop is valid

### 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

CLEAR(S)	Reserved													
R/W														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCEL 15	HCEL 14	HCEL 13	HCEL 12	HCEL 11	HCEL 10	HCEL 9	HCEL 8	HCEL 7	HCEL 6	HCEL 5	HCEL 4	HCEL 3	HCEL 2	HCEL 1	HCEL 0
R/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR

Bit tag	function	
31	STARTS hardware clear enable	<p>0: Hardware clearing is invalid 1: Hardware clearing is valid</p> <p>Note: When hardware clearing is valid, the setting of software clearing is invalid.</p>
30:16	Reserved -	
15	HCEL15 hardware clear condition 15: TIMTRID port upsampled to falling edge	<p>0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid</p>
14	HCEL14 Hardware Cleared Condition 14: TIMTRID port upsampled to rising edge	<p>0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid</p>
13	HCEL13 Hardware Cleared Condition 13: TIMTRIC port upsampled to falling edge	<p>0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid</p>
12	HCEL12 hardware cleared Condition 12: TIMTRIC port upsampled to rising edge	<p>0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid</p>
11	HCEL11 hardware clear condition 11: TIMTRIB port upsampled to falling edge	<p>0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid</p>
10	HCEL10 Hardware Clear Condition 10: TIMTRIB port up sampled to rising edge	<p>0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid</p>
9	HCEL9 Hardware Cleared Condition 9: TIMTRIA port upsampled to falling edge	<p>0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid</p>



8	HCEL8 Hardware Clear Condition 8: TIMTRIA port upsampled to rising edge	0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid
7	HCEL7 Hardware Cleared Condition 7: Falling edge on CHxB port sampled	0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid
6	HCEL6 Hardware Clear Condition 6: CHxB port upsampled to rising edge	0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid
5	HCEL5 Hardware Clear Condition 5: CHxA port upsampled to falling edge	0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid
4	HCEL4 Hardware Clear Condition 4: CHxA port upsampled to rising edge	0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid
3	HCEL3 hardware clear condition 3: Event trigger 3 from AOS is valid	0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid
2	HCEL2 Hardware Clear Condition 2: Event Trigger 2 from AOS is valid	0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid
1	HCEL1 Hardware Clear Condition 1: Event Trigger 1 from AOS is valid	0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid
0	HCEL0 Hardware clear condition 0: Event trigger 0 from AOS is valid	0: When the condition is matched, hardware clearing is invalid 1: When the condition is matched, hardware clearing is valid

### 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/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/W				

Bit tag function															
31:16	Reserved -														
15	HCPA15 Hardware Capture A Condition 15: TIMTRID port upsampled to falling edge  0: Hardware capture A is invalid when the condition is matched  1: Hardware capture A is valid when the condition is matched														
14	HCPA14 Hardware Capture A Condition 14: TIMTRID port up sampled to rising edge  0: Hardware capture A is invalid when the condition is matched  1: Hardware capture A is valid when the condition is matched														
13	HCPA13 Hardware Capture A Condition 13: TIMTRIC port upsampled to falling edge  0: Hardware capture A is invalid when the condition is matched  1: Hardware capture A is valid when the condition is matched														
12	HCPA12 Hardware Capture A Condition 12: TIMTRIC port up sampled to rising edge  0: Hardware capture A is invalid when the condition is matched  1: Hardware capture A is valid when the condition is matched														
11	HCPA11 Hardware Capture A Condition 11: TIMTRIB port upsampled to falling edge  0: Hardware capture A is invalid when the condition is matched  1: Hardware capture A is valid when the condition is matched														
10	HCPA10 Hardware Capture A Condition 10: TIMTRIB port upsampled to rising edge  0: Hardware capture A is invalid when the condition is matched  1: Hardware capture A is valid when the condition is matched														
9	HCPA9 Hardware Capture A Condition 9: TIMTRIA port upsampled to falling edge  0: Hardware capture A is invalid when the condition is matched  1: Hardware capture A is valid when the condition is matched														
8	HCPA8 Hardware Capture A Condition 8: TIMTRIA port up sampled to rising edge  0: Hardware capture A is invalid when the condition is matched  1: Hardware capture A is valid when the condition is matched														
7	HCPA7 Hardware Capture A Condition 7: CHxB port upsampled to falling edge  0: Hardware capture A is invalid when the condition is matched														



		1: Hardware capture A is valid when the condition is matched
6	HCPA6 Hardware Capture A Condition 6: CHxB port up sampled to rising edge	0: Hardware capture A is invalid when the condition is matched 1: Hardware capture A is valid when the condition is matched
5	HCPA5 Hardware Capture A Condition 5: CHxA port upscaled to falling edge	0: Hardware capture A is invalid when the condition is matched 1: Hardware capture A is valid when the condition is matched
4	HCPA4 Hardware Capture A Condition 4: CHxA port up sampled to rising edge	0: Hardware capture A is invalid when the condition is matched 1: Hardware capture A is valid when the condition is matched
3	HCPA3 hardware capture A condition 3: Event trigger 3 from AOS is valid	0: Hardware capture A is invalid when the condition is matched 1: Hardware capture A is valid when the condition is matched
2	HCPA2 hardware capture A condition 2: Event trigger 2 from AOS is valid	0: Hardware capture A is invalid when the condition is matched 1: Hardware capture A is valid when the condition is matched
1	HCPA1 Hardware Capture A Condition 1: Event Trigger 1 from AOS is valid	0: Hardware capture A is invalid when the condition is matched 1: Hardware capture A is valid when the condition is matched
0	HCPA0 Hardware Capture A Condition 0: Event Trigger 0 from AOS is valid	0: Hardware capture A is invalid when the condition is matched 1: Hardware capture A is valid when the condition is matched

### 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
----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

HCPB 15	HCPB 14	HCPB 13	HCPB 12	HCPB 11	HCPB 10	HCPB 9	HCPB 8	HCPB 7	HCPB 6	HCPB 5	HCPB 4	HCPB 3	HCPB 2	HCPB 1	HCPB 0
R/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/W				

Bit tag function		
31:16	Reserved -	
15	HCPB15 Hardware Capture B Condition 15: TIMTRID port upsampled to falling edge	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
14	HCPB14 Hardware Capture B Condition 14: TIMTRID port up sampled to rising edge	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
13	HCPB13 Hardware Capture B Condition 13: TIMTRIC port upsampled to falling edge	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
12	HCPB12 Hardware Capture B Condition 12: TIMTRIC port up sampled to rising edge	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
11	HCPB11 Hardware Capture B Condition 11: TIMTRIB port upsampled to falling edge	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
10	HCPB10 Hardware Capture B Condition 10: TIMTRIB port up sampled to rising edge	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
9	HCPB9 Hardware Capture B Condition 9: TIMTRIA port upsampled to falling edge	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
8	HCPB8 Hardware Capture B Condition 8: TIMTRIA port up sampled to rising edge	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
7	HCPB7 Hardware Capture B Condition 7: CHxB port upsampled to falling edge	0: Hardware capture B is invalid when the condition is matched



		1: Hardware capture B is valid when the condition is matched
6	HCPB6 Hardware Capture B Condition 6: CHxB port up sampled to rising edge	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
5	HCPB5 Hardware Capture B Condition 5: CHxA port upscaled to falling edge	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
4	HCPB4 Hardware Capture B Condition 4: CHxA port up sampled to rising edge	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
3	HCPB3 hardware capture B condition 3: event trigger 3 from AOS is valid	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
2	HCPB2 hardware capture B condition 2: Event trigger 2 from AOS is valid	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
1	HCPB1 Hardware Capture B Condition 1: Event Trigger 1 from AOS is valid	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched
0	HCPB0 Hardware Capture B Condition 0: Event Trigger 0 from AOS is valid	0: Hardware capture B is invalid when the condition is matched 1: Hardware capture B is valid when the condition is matched



### 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/WR	WR/WR/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/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/W				

bit	Mark function	
31:20	Reserved -	
19	HCUP19 Hardware increment condition: Event trigger 3 from AOS is valid	<p>0: When the condition is matched, the hardware increment is invalid</p> <p>1: When the condition is matched, the hardware increment is valid</p>
18	HCUP18 Hardware increment condition: Event trigger 2 from AOS is valid	<p>0: When the condition is matched, the hardware increment is invalid</p> <p>1: When the condition is matched, the hardware increment is valid</p>
17	HCUP17 Hardware increment condition: Event trigger 1 from AOS is valid	<p>0: When the condition is matched, the hardware increment is invalid</p> <p>1: When the condition is matched, the hardware increment is valid</p>
16	HCUP16 Hardware increment condition: event trigger 0 from AOS is valid	<p>0: When the condition is matched, the hardware increment is invalid</p> <p>1: When the condition is matched, the hardware increment is valid</p>
15	HCUP15 Hardware increment condition: TIMTRID port upsampled to falling edge	<p>0: When the condition is matched, the hardware increment is invalid</p> <p>1: When the condition is matched, the hardware increment is valid</p>
14	HCUP14 Hardware increment condition: TIMTRID port upsampled to rising edge	<p>0: When the condition is matched, the hardware increment is invalid</p> <p>1: When the condition is matched, the hardware increment is valid</p>
13	HCUP13 Hardware increment condition: TIMTRIC port upsampled to falling edge	<p>0: When the condition is matched, the hardware increment is invalid</p> <p>1: When the condition is matched, the hardware increment is valid</p>
12	HCUP12 Hardware increment condition: TIMTRIC port upsampled to rising edge	<p>0: When the condition is matched, the hardware increment is invalid</p> <p>1: When the condition is matched, the hardware increment is valid</p>



11	HCUP11 hardware increment condition: TIMTRIB port upsampled to falling edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid
10	HCUP10 hardware increment condition: TIMTRIB port upsampled to rising edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid
9	HCUP9 hardware increment condition: TIMTRIA port upsampled to falling edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid
8	HCUP8 hardware increment condition: TIMTRIA port upsampled to rising edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid
7	HCUP7 Hardware Increment Condition: When the CHxB port is high, the CHxA port is upsampled to a falling edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid
6	HCUP6 Hardware Increment Condition: When CHxB port is high, CHxA port is upsampled to rising edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid
5	HCUP5 Hardware Increment Condition: When the CHxB port is low, the CHxA port is up-sampled to a falling edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid
4	HCUP4 hardware increment condition: when CHxB port is low, CHxA port is up-sampled to rising edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid
3	HCUP3 Hardware Increment Condition: When the CHxA port is high, the CHxB port is upsampled to a falling edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid
2	HCUP2 hardware increment condition: when CHxA port is high, CHxB port is upsampled to rising edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid
1	HCUP1 Hardware Increment Condition: When the CHxA port is low, the CHxB port is up-sampled to a falling edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid
0	HCUP0 Hardware Increment Condition: When the CHxA port is low, the CHxB port is up-sampled to a rising edge	0: When the condition is matched, the hardware increment is invalid 1: When the condition is matched, the hardware increment is valid



### 12.3.20 Hardware Decrement Event Select Register (TIMx\_HCDOR)

Address offset: 0x08C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					26						HCD	HCD	HCD	HCD	
					Reserved						O	O	O	O	
											R/WR	WR/WR	W167		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCD	HCD	HCD	HCD	HCD	HCD	HCD	HCD	HCD	HCD	HCD	HCD	HCD	HCD	HCD	HCD
O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O
R/WR	WR/WR														

bit	Mark function	
31:20	Reserved -	
19	HCDO19 hardware decrement condition: event trigger 3 from AOS is valid	<p>0: When the condition is matched, the hardware decrement is invalid</p> <p>1: When the condition is matched, the hardware decrement is valid</p>
18	HCDO18 hardware decrement condition: event trigger 2 from AOS is valid	<p>0: When the condition is matched, the hardware decrement is invalid</p> <p>1: When the condition is matched, the hardware decrement is valid</p>
17	HCDO17 hardware decrement condition: event trigger 1 from AOS is valid	<p>0: When the condition is matched, the hardware decrement is invalid</p> <p>1: When the condition is matched, the hardware decrement is valid</p>
16	HCDO16 hardware decrement condition: event trigger 0 from AOS is valid	<p>0: When the condition is matched, the hardware decrement is invalid</p> <p>1: When the condition is matched, the hardware decrement is valid</p>
15	HCDO15 hardware decrement condition: TIMTRID port upsampled to falling edge	<p>0: When the condition is matched, the hardware decrement is invalid</p> <p>1: When the condition is matched, the hardware decrement is valid</p>
14	HCDO14 hardware decrement condition: TIMTRID port up sampled to rising edge	<p>0: When the condition is matched, the hardware decrement is invalid</p> <p>1: When the condition is matched, the hardware decrement is valid</p>
13	HCDO13 hardware decrement condition: TIMTRIC port upsampled to falling edge	<p>0: When the condition is matched, the hardware decrement is invalid</p> <p>1: When the condition is matched, the hardware decrement is valid</p>
12	HCDO12 hardware decrement condition: TIMTRIC port up sampled to rising edge	<p>0: When the condition is matched, the hardware decrement is invalid</p> <p>1: When the condition is matched, the hardware decrement is valid</p>



11	HCDO11 hardware decrement condition: TIMTRIB port upsampled to falling edge	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid
10	HCDO10 hardware decrement condition: TIMTRIB port upsampled to rising edge	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid
9	HCDO9 hardware decrement condition: TIMTRIA port upsampled to falling edge	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid
8	HCDO8 hardware decrement condition: TIMTRIA port upsampled to rising edge	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid
7	HCDO7 hardware decrement condition: when CHxB port is high, CHxA port is upsampled to falling edge	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid
6	HCDO6 hardware decrement condition: CHxA port upsampled to rising edge when CHxB port is high	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid
5	HCDO5 hardware decrement condition: when CHxB port is low, CHxA port is upsampled to falling edge	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid
4	HCDO4 hardware decrement condition: CHxA port upsampled to rising edge when CHxB port is low	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid
3	HCDO3 hardware decrement condition: when CHxA port is high, CHxB port is upsampled to falling edge	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid
2	HCDO2 hardware decrement condition: CHxB port upsampled to rising edge when CHxA port is high	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid
1	HCDO1 hardware decrement condition: when CHxA port is low, CHxB port is upsampled to falling edge	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid
0	HCDO0 hardware decrement condition: when CHxA port is low, CHxB port is up-sampled to rising edge	0: When the condition is matched, the hardware decrement is invalid 1: When the condition is matched, the hardware decrement is valid

### 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												SSTA2	SSTA1	SSTA0	
												R/WR	WR/W		

bit	mark	Function
31:3	Reserved	
2	SSTA2	Timer6 software start 0: Invalid software startup 1: Software startup enable
1	SSTA1	Timer5 software start 0: Invalid software startup 1: Software startup enable
0	SSTA0	Timer4 software start 0: Invalid software startup 1: Software startup enable



### 12.3.22 Software Synchronization Stop Register (TIMx\_SSTPR)

Address offset: 0x3F8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												SSTP2	SSTP1	SSTP0	
												R/WR	WR/W		

bit	mark	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	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												SCLR2	SCLR1	SCLR0	
												R/WR	WR/W		

bit	mark	Function
31:3	Reserved	
2	SCLR2	Timer6 software reset 0: Software clearing is invalid 1: Software clear enable
1	SCLR1	Timer5 software reset 0: Software clearing is invalid 1: Software clear enable
0	SCLR0	Timer4 software clear 0: Software clearing 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						20	19	18	17	16	
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4 3	2	1	0	
SAMHF	SMLF	Reserved				DTEF	UDFF	OVFF		Reserved	CMDF	CMCF	CMBF	CMAF	
RO	RO					RO	RO	RO		RO	RO	RO	RO	RO	

bit	mark	place name function
31:16	Reserved	-
15	SAMHF	CHxA/B port high status interrupt flag 0: No simultaneous high levels on CHxA and CHxB ports 1: Simultaneous high level on CHxA and CHxB ports
14	SMLF	CHxA/B port low status interrupt flag 0: No simultaneous low levels on CHxA and CHxB ports 1: Simultaneous low level on 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: No underflow of sawtooth wave or triangle wave count to the valley point 1: The sawtooth wave underflow occurs or the triangle wave counts to the valley point
6	OVFF	Overflow match interrupt flag 0: No sawtooth wave overflow or triangle wave count to the peak point 1: The sawtooth wave overflow occurs or the triangle wave counts to the peak point
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 the same as the count value value equal
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 the same as the count value value equal
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 the CHxB capture completion action occurs
0	CMAF	count match A interrupt flag 0: The value of the GCMAR register is not equal to the count value, and the CHxA capture completion action has not occurred



		1: The value of the GCMAR register is equal to the count value, or the CHxA capture completion 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							20	19	18	17	16
Reserved																	
15	14	13	12	11	10	9	8	7	6	5 4 3		2	1	0			
SAMHC W0	SAMLC W0	Reserved -				DTEC W0	UDFC W0	OVFC W0		Reserved	CMDC W0	CMCC W0	CMBC W0	CMAC W0			

bit	mark	Function
31:16	Reserved	-
15	SAMHC	CHxA/B port high state interrupt flag is cleared, write 1 is invalid, write 0 to clear the corresponding interrupt
14	SAMLC	CHxA/B port low state interrupt flag is cleared, write 1 is invalid, write 0 to clear the corresponding interrupt
13:9	Reserved	-
8	DTEC	Dead time error interrupt flag is cleared, write 1 is invalid, write 0 to clear the corresponding interrupt
7	UDFC	Underflow match interrupt flag is cleared, write 1 is invalid, write 0 to clear the corresponding interrupt
6	OVFC	The overflow matching interrupt flag is cleared, writing 1 is invalid, and writing 0 clears the corresponding interrupt
5:4	Reserved	-
3	CMDC	Count match D interrupt flag is cleared, write 1 is invalid, write 0 to clear the corresponding interrupt
2	CMCC	Count match C interrupt flag is cleared, write 1 is invalid, write 0 to clear the corresponding interrupt
1	CMBC	Count match B interrupt flag is cleared, write 1 is invalid, write 0 to clear the corresponding interrupt
0	CMAC	Count match A interrupt flag is cleared, write 1 is invalid, write 0 to clear 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	DITNA	UDFE	OVFE	Reserved		CMDE	CMCE	CMBE	CMAE	
				R/WR	WR/WR	WR/W					R/WR	WR/WR	/W		

bit	mark	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 is invalid, 1: Enable is valid, change the output delay of PWM every cycle	
9	DITENA PWM channel A spread spectrum enable 0: Enable is invalid, 1: Enable is valid, change the output delay of PWM every cycle	
7	UDFE underflow match enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS_i_tirg	
6	OVFE overflow match enable 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 other two timers will

The value changes at the same time.

31	30	29	28	27	26	25	24		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	SMH2 S	SMH1 SMH0	SML2 SML1	SML0 SOFTB					Reserved	BFILTEN	BFILTS	FSAME	F BRAKE		
	R/WR	WR/WR	WR/WR	WR/WR			K			R/W	R/W	R	R		

bit	mark	Function
31:14	Reserved	-
13	SMH2 channel 2	with high selection 0: The selection is invalid, 1: The selection is valid, AOS_i_odis[1] appears when the same high
12	SMH1 channel 1	with high 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 is 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 is 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 is 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	High and low brake flag, read only
0	F BRAKE	port brake flag, read only

### 12.3.28 AOS Select 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 other two timers will

The value changes at the same time.

31	30	29	28	27	26	25	24		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															FSAM	FBRAKE
															R	R

bit	mark	Function
31:2	Reserved	-
1	FSAME	same high and low brake flag is cleared, write 0 to clear, write 1 is invalid, read is always 1
0	FBRAKE	port brake flag is cleared, write 0 to clear, write 1 is invalid, read is always 1

### 12.3.29 Port Brake Control Register (TIMx\_PTBKRS)

Address offset: 0x118

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register, after any one timer is changed, the other two timers will

The value changes at the same time.

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	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/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/W									

bit	Mark function	
31:16	Reserved -	
15	EN15	P36 Brake port enable: 1 for selection, 0 for invalid
14	EN14	P35 Brake port enable: 1 for selection, 0 for invalid
13	EN13	P34 Brake port enable: 1 for selection, 0 for invalid
12	EN12	P33 Brake port enable: 1 is selected, 0 is invalid
11	EN11	P32 Brake port enable: 1 for selection, 0 for invalid
10	EN10	P31 Brake port enable: 1 for selection, 0 for invalid
9	EN9	P27 Brake port enable: 1 for selection, 0 for invalid
8	EN8	P26 Brake port enable: 1 for selection, 0 for invalid
7	EN7	P26 Brake port enable: 1 for selection, 0 for invalid
6	EN6	P24 Brake port enable: 1 for selection, 0 for invalid
5	EN5	P23 Brake port enable: 1 for selection, 0 for invalid
4	EN4	P15 Brake port enable: 1 for selection, 0 for invalid
3	EN3	P14 Brake port enable: 1 for selection, 0 for invalid
2	EN2	P03 Brake port enable: 1 is selected, 0 is invalid
1	EN1	P02 Brake port enable: 1 is selected, 0 is invalid
0	EN0	P01 Brake port enable: 1 is selected, 0 is 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 other two timers will

The value changes at the same time.

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	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	Mark 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

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 other two timers will

The value changes at the same time.

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	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																											
<table border="1"> <thead> <tr> <th>bit</th><th>mark</th><th>Function</th></tr> </thead> <tbody> <tr> <td>31:16</td><td>Reserved</td><td>-</td></tr> <tr> <td>15:12</td><td>IAOS3S</td><td>TIMx AOS3 trigger source selection</td></tr> <tr> <td>11:8</td><td>IAOS2S</td><td>TIMx AOS2 trigger source selection</td></tr> <tr> <td>7:4</td><td>IAOS1S</td><td>TIMx AOS1 trigger source selection</td></tr> <tr> <td>3:0</td><td>IAOS0S</td><td>TIMx AOS0 trigger source selection</td></tr> </tbody> </table>																bit	mark	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
bit	mark	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 the interrupt source are selected 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	MER_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\_PTBKPx)

Address offset: 0x124

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register, after any one timer is changed, the other two timers will

The value changes at the same time.

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POL15 P R/WR	POL14 POL13 WR/WR	POL12 POL11 WR/WR	POL10 POL9 WR/WR				POL8 WR/WR	POL7 WR/WR	POL6 POL5 WR/WR	POL4 WR/WR	POL3 WR/WR	POL2 WR/W	POL1 WR/W	POL0 W	

bit	Mark function	
31:16	Reserved -	
15	POL15	P36 Brake port polarity selection: 1 is active at low level, 0 is active at high level
14	POL14	P35 Brake port polarity selection: 1 is active at low level, 0 is active at high level
13	POL13	P34 Brake port polarity selection: 1 is active at low level, 0 is active at high level
12	POL12	P33 Brake port polarity selection: 1 is active at low level, 0 is active at high level
11	POL11	P32 Brake port polarity selection: 1 is active at low level, 0 is active at high level
10	POL10	P31 Brake port polarity selection: 1 is active at low level, 0 is active at high level
9	POL9	P27 Brake port polarity selection: 1 is active at low level, 0 is active at high level
8	POL8	P26 Brake port polarity selection: 1 is valid at low level, 0 is valid at high level
7	POL7	P26 Brake port polarity selection: 1 is valid at low level, 0 is valid at high level
6	POL6	P24 Brake port polarity selection: 1 is active at low level, 0 is active at high level
5	POL5	P23 Brake port polarity selection: 1 is active at low level, 0 is active at high level
4	POL4	P15 Brake port polarity selection: 1 is active at low level, 0 is active at high level
3	POL3	P14 Brake port polarity selection: 1 is valid at low level, 0 is valid at high level
2	POL2	P03 Brake port polarity selection: 1 is valid at low level, 0 is valid at high level
1	POL1	P02 Brake port polarity selection: 1 is valid at low level, 0 is valid at high level
0	POL0	P01 Brake port polarity selection: 1 is valid at low level, 0 is valid at high level

## 13 Real Time Clock (RTC)

### 13.1 Introduction to Real Time Clock

Real-time clock/calendar provides information for seconds, minutes, hours, days, weeks, months, years, and the number of days in each month and in leap years.

auto-adjust. Clock operation can be determined in 24 or 12 hour format via the AM/PM register bits. Table 14-

1 shows its basic characteristics.

clock source	Off-chip low-speed crystal oscillator XTL (32.768kHz) On-chip low-speed oscillator RCL (32kHz, 1% accuracy) Off-chip high-speed crystal oscillator XTH Can calculate seconds, minutes, hours, days, weeks,
basic skills	months, and years between 00 and 99. Automatic leap year adjustment Configurable for 24 or 12 hour
	format Programmable start or stop with alarm function with high precision 1Hz square wave output with
	periodic interrupt
interrupt	Has an alarm interrupt

Table 13-1 Basic characteristics of RTC

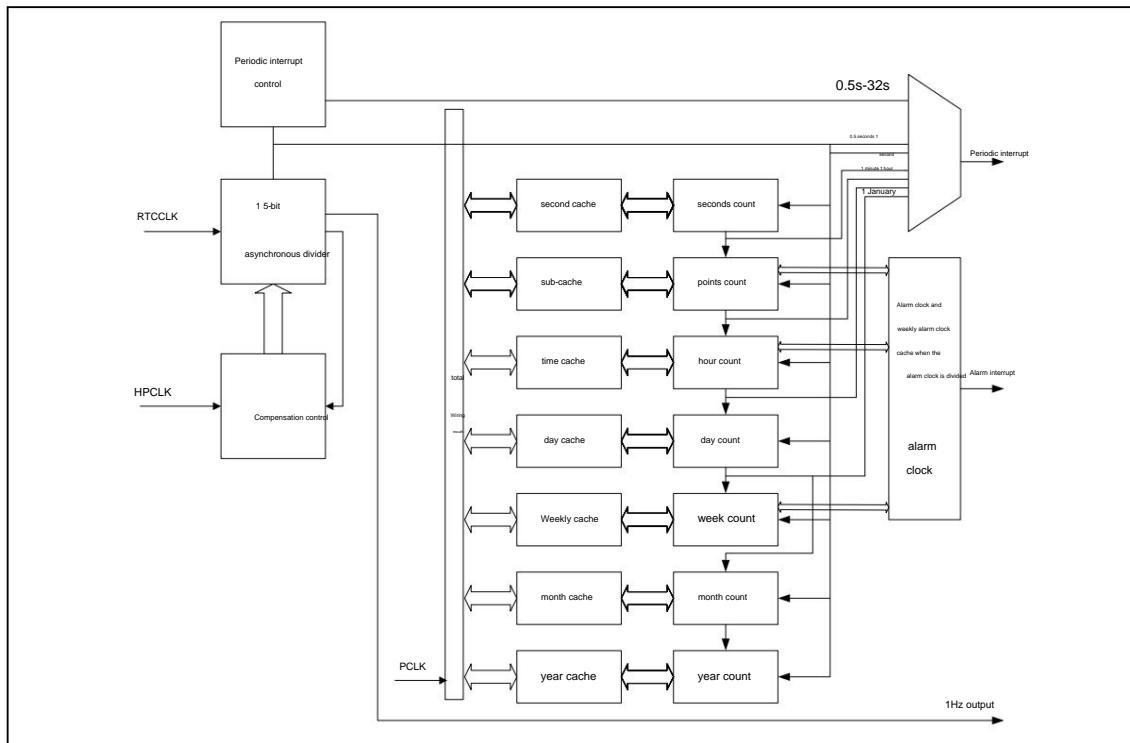


Figure 13-1 RTC block diagram



## 13.2 Real-time clock function 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, and an internal low-speed RC; it is used by default

External low-speed crystal oscillator. Control registers CR0, CR1 and COMPEN are only controlled by power-on reset, other reset sources

These three control registers cannot be reset. The power-on status of other data registers is uncertain and needs to be initialized after power-on.  
affected by any reset.

All date and time values written and read by the software are in BCD code, and do not need to be converted from hexadecimal to decimal.

Any invalid date time will not be able to be written, such as 32nd, 25h, 70s, B month, etc.

### 13.2.1 Power-on settings

The RTC is reset once after power-on. When the system is not powered off, various external reset requests cannot be reset.

RTC, RTC will always be in the counting state. After power-on, set the initial calendar value, alarm clock setting, error compensation

After compensation, interruption, etc., start the RTC.

### 13.2.2 RTC count start setting

1. Set CR0.START=0 to stop counting;
2. Set CR0.AMPM and CR0.PRDS, CR0.PRDX set time system and interrupt period;
3. Set CR1.CKSEL to select the timing clock of RTC;
4. Set the calendar count register for seconds, minutes, hours, weeks, days, months, and years;
5. When clock error compensation is required, set the count 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 count starts, if the system switches to the low power mode immediately, please perform any one of the following confirmations

Then switch the mode.

The control register is in the system control register SYSCTRL1.RTC\_LPW

1. After CR0.START=1 is set, the mode is switched after more than 2 RTC count clocks.
2. After setting CR0.START=1, set CR1.WAIT=1 and query CR1.WAITF=1. reset



CR1.WAIT=0, query CR1.WAITF=0 and then switch the mode.

In RTC low power mode, RTC registers cannot be read or written. In low power mode, the RTC consumes less power flow.

Switching to low power modes while the RTC is running does not require waiting.

#### 13.2.4 Reading the count register

There are three ways to read the count register:

ÿ Mode 1: Read Mode 1 at any time

1. Set CR1.WAIT=1, stop the calendar register counting, and enter the read-write mode;
2. Query until CR1.WAITF=1;
3. Read out 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.

ÿ Mode 2: Read Mode 2 at any time

1. Read out minute, hour, week, day, month, year count register value;
2. Read the second count register value;
3. Read the second count register value again;
4. Determine whether the read values of the two seconds are the same. If the difference is different, start from the first step and end with the same read.

ÿ Mode 3: Interrupt reading mode

Read second, minute, hour, week, day, month, year count register value in RTC periodic interrupt service. because of interruption

The time from the occurrence to the next data change is at least 0.5s.

#### 13.2.5 Write Count Register

1. Set CR1.WAIT=1, stop the calendar register counting, and enter the read-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.

Writing the second, minute, hour, week, day, month, and year count registers does not need to wait for WAIT when the RTC is not activated.

Note:

– Changing the seconds register in count mode resets the seconds count, writes the minute, hour, week, day, month, and year count registers

The value does not affect the RTC count.

### 13.2.6 Alarm setting

1. Set CR1.ALLEN=0, the alarm clock is disabled;
2. Set CR1.ALMI=1, the alarm interrupt is allowed;
3. The minute alarm clock ALMMIN, the hour alarm clock ALMHOUR, and the weekly alarm clock ALMWEEK are set;
4. Set CR1.ALLEN=1, the alarm clock is allowed;
5. Wait for an interrupt to occur;
6. Since the alarm interrupt and the fixed period interrupt share the interrupt request signal, when CR1.ALMF=1, enter the alarm interrupt processing; otherwise, enter the periodic interrupt processing.

### 13.2.7 1Hz output

RTC can choose to output general precision, high precision and high precision 3 kinds of 1Hz clock. When the clock error compensation function

When it is valid, it outputs a high-precision 1Hz clock; when using PCLK of different frequencies, it outputs a high-precision 1Hz clock

Bell. The system control registers need to be configured according to the PCLK frequency, where,

ÿ The 1Hz output setting of general accuracy is as follows: (without clock compensation)

1. Set CR0.START=0 to stop counting;
2. RTC output pin setting;
3. CR0.1HZOE=1, the clock output is allowed;
4. Set CR0.START=1 to start counting;
5. Wait for more than 2 count cycles;
6. 1Hz output starts.

ÿ The high-precision 1Hz output setting is as follows: (low speed compensation)

1. Set CR0.START=0 to stop counting;
2. RTC output pin setting;
3. CR0.1HZOE=1, the clock output is allowed;
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 the RTC with

4M, 6M, 8M, 12M, 16M, 20M, 24M, 32MHz high-speed PCLK clock, the output settings are as follows:

1. Set CR0.START=0 to stop counting;

2. RTC output pin setting;

3. CR0.1HZOE=1, the clock output is allowed;

4. CR0.1HZSEL=1, select to output high-precision 1Hz clock;

5. Configure the high-speed clock compensation clock SYSCTRL1.RTC\_FREQ\_ADJUST

6. Clock error compensation register COMPEN.CR[8:0] Compensation number setting;

7. When the clock error compensation register COMPEN.EN=1, the precision 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

Since there is an error in the external crystal oscillator, when a high-precision counting result needs to be obtained, the error needs to be compensated.

There are two types of compensation methods: the first one is based on the error compensation of its own clock; the second one is based on the error compensation of the high-speed clock.

poor compensation.

The principle and calculation of error compensation based on its own clock:

Since the counter uses a 32.768KHz clock for counting, if you need to compensate for the accuracy per second, you can only follow the

32.768KHz integer period compensation, the minimum unit of compensation per second is  $(1/32768)*106=30.5\text{ppm}$ , no

method to meet the requirements of high precision.

Then to achieve high-precision clock compensation under the 32.768KHz count clock, it is necessary to adjust the algorithm.

Adjust the maximum compensation period by 32 times. If the minimum unit that can only be compensated is 30.5ppm, the

The average compensation unit per second becomes  $30.5\text{ppm}/32=0.96\text{ppm}$ . It meets the clock compensation requirements with higher precision. and



And compensation occurs in a relatively uniform range every 32 seconds. Therefore, a setting of 5 decimal places is introduced into this register.

Certainly.

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 the actual measured value is 0.9999888Hz, then:

$$\text{Actual vibration frequency} = 32768 \times 0.9999888 \quad 32767.63$$

$$\text{Compensation target value} = (\text{actual vibration frequency} - \text{target frequency})/\text{target frequency} \times 10^6$$

$$= (32767.96 - 32768)/32768 \times 10^6$$

$$= -11.29\text{ppm}$$

according to

$$\text{CR}[8:0] = \frac{\text{Compensation target value [ppm]} \times 15}{10^6} + 0001.00000\text{B}$$

Take 2's complement

If the compensation target value is -11.29ppm, the corresponding register value is calculated as follows:

$$\text{CR}[8:0] = (-11.29 \times 215/106) \text{ 2's complement} + 0001.00000\text{B}$$

$$= (-0.37) \text{ 2's complement} + 0001.00000\text{B}$$

$$= 1111.10101\text{B} + 0001.00000\text{B}$$

$$= 0000.10101\text{B}$$

Error compensation principle and calculation based on high-speed **24MHz** clock:

The calculation method of this method is the same as the error compensation based on the own clock. Due to the introduction of 4M-32MHz high speed

clock, the error of 1/32768 of a second that would otherwise need to accumulate over a maximum of 32 seconds can be spread out to every 1 second, for every 1 second

Compensation of minimum 0.96ppm (23 24MHz clock cycles) to achieve an average high precision 1Hz clock per second

output.

## 13.3 RTC Interrupts

The RTC supports two types of interrupts. Alarm clock interrupt, fixed period interrupt. The alarm interrupt and the periodic interrupt share one break the signal.

### 13.3.1 RTC Alarm Interrupt

When CR1.ALMIE=1, if the current calendar time and minute alarm register (ALMMIN), hour alarm register (ALMHOUR) and the weekly alarm register (ALMWEEK) are equal to trigger the alarm interrupt.

### 13.3.2 RTC cycle interrupt

When ALMIE=1 in control register 1 (CR1), after the selected cycle occurs, the regular cycle wake-up interrupt is triggered.

Common interrupts for alarm clock and fixed period are distinguished by flag register bits.

## 13.4 RTC register description

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 count register
RTC_HOUR	0X010	time count 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	Sub-alarm register
RTC_ALMHOUR	0X028	time alarm register
RTC_ALMWEEK	0X02C	Weekly alarm register
RTC_COMPEN	0X030	Clock Error Compensation Register

Table 13-2 RTC register list

### 13.4.1 Control Register 0 (RTC\_CR0)

\*Only power-on reset of this register is valid

Address offset: 0x000

Reset value 0x0000 0000

31	30 29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	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/WR/WR/W				R/W	R/W						

bit	Symbolic function	description
31:15	Reserved -	
14	PRDSEL 0: Use the periodic interrupt interval set by PRDS 1: Use the periodic interrupt time interval set by PRDX	
13:8	PRDX Sets the time interval for generating periodic interrupts. The range that can be set is 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 the 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-hour clock 1: 24-hour clock	
2:0	PRDS sets the interval at which interrupts are generated: 000: No periodic interrupt is generated 001: 0.5 seconds 010: 1 second 011: 1 minute 100: 1 hour 101: 1 day 11x: January	

Note: If you need to write the time interval of the change cycle interrupt when START=1, the operation steps are as follows:



		<p>step1, close the RTC interrupt in NVIC; step2, change the time interval of the periodic interrupt; step3, clear the RTC interrupt flag; step4, enable the RTC interrupt.</p>
--	--	---

### 13.4.2 Control Register 1 (RTC\_CR1)

\*Only power-on reset of this register is valid

Address offset: 0x004

Reset value 0X00000000

31	30	29	28 27	26 25	Twenty four	Twenty three	Twenty two	Twenty one	20	19	18	17	16
Reserved													
15:11	10:8	7	6	5	4	3	2	1	0				
14 Reserved 13	CKSEL	ALMEN	ALMIE	Res.	ALMF	PRDF	Res.	WAITF	WAIT				
	R/W	R/WR/W			RO	RO		R/WR/W					

bit sign		Function description
31:11	Reserved	-
10:8	CKSEL	RTC clock selection 00x: XTL 32.768k 01x: RCL 32k 100: XTH/128 (select this when the crystal oscillator is 4M) 101: XTH/256 (select this when the crystal oscillator is 8M) 110: XTH/512 (select this when the crystal oscillator is 16M) 111: XTH/1024 (select this when the crystal oscillator is 32M)
7	ALMEN 0: disable alarm	1: Enable alarm clock Note: Enabled during START=1 calendar count and ALMIE=1 interrupt enable During ALMEN, turn off the system interrupt to prevent malfunction. After enabling, please clear the ALMF flag remove.
6	ALMIE 0: disable alarm interrupt	1: Enable alarm 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 is matched, 1 is set after a clock of 32.768KHz. Writing 0 clears the flag, writing 1 has no effect.
3	PRDF	0: No periodic interrupt occurred 1: A periodic interrupt has occurred Note: This bit is set after a periodic interrupt occurs. Write 0 to clear this flag, write 1 to have no effect.
2	Reserved	-
1	WAITF	0: non-write/read status 1: write/read status Note: The WAIT bit sets the valid flag. Please make sure this bit is "1" before writing/reading. count



		During the counting process, this bit is cleared to "0" only after the WAIT bit is cleared to "0" and the writing is completed.
0	WAIT	<p>0: Normal counting mode 1: write/read mode</p> <p>Note: Please set this bit to "1" when writing/reading, because the counter is counting continuously, please set it in 1 second</p> <p>Complete the write/read operation within and clear this bit to "0".</p>

### 13.4.3 Second Count Register (RTC\_SEC)

Address offset: 0x008

Reset value: indeterminate

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	Function description
31:7	Reserved	-
6:4	SECH	Seconds count tens value
3:0	SECL	second count unit value

Represents 0-59 seconds in decimal counts. Please write the BCD code of decimal 0-59, when writing the wrong value, write

The entered value will be ignored.

### 13.4.4 Minute Count Register (RTC\_MIN)

Address offset: 0x00C

Reset value: indeterminate

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	Function description
31:7	Reserved	-
6:4	MINH	Minute count tens value
3:0	MINL	Minute count unit value

Represents 0-59 minutes in decimal notation. Please write the BCD code of decimal 0-59, when writing the wrong value, write

The entered value will be ignored.



### 13.4.5 Hour count register (RTC\_HOUR)

Address offset: 0x010

Reset value: indeterminate

bit	symbol	Function description
31:6	Reserved	-
5:4	Counting tens value on HOURH	
3:0	Count units value when HOURL	

In 24-hour time format, it means 0-23 hours. In 12-hour time system, b5=0 means AM, then 01:12 means up

Noon; 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.

Values written out of range are ignored.



Refer to the table below for specific time:

24-hour clock	AMPM=1	12-hour clock	AMPM=0
time	register representation	time	register representation
00 hours	00H	AM 12:00	12H
	01H	AM 01	01H
	02H	AM 02	02H
	03H	AM 03	03H
	04H	AM 04	04H
	05H	AM 05	05H
	06H	AM 06	06H
01:02:03:04:05:06:07	07H	AM 07	07H
08:00	08H	AM 08	08H
09:00	09H	AM 09	09H
10 o'clock	10H	10 am	10H
	11H	11 am	11H
	12H	PM 12:00	32H
	13H	PM 01	21H
	14H	PM 02 hours	22H
	15H	PM 03 hours	23H
	16H	PM 04 hours	24H
	17H	PM 05 hours	25H
11:12:13:14:15:16:17:18	18H	PM 06 hours	26H
19:00	19H	PM 07	27H
20:00	20H	PM 08	28H
	21H	PM 09:00	29H
	22H	10:00 PM	30H
21:22:23	23H	11:00 PM	31H

### 13.4.6 Day count register (RTC\_DAY)

Address offset: 0x018

Reset value: indeterminate

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	Function description
31:6	Reserved	-
5:4	DAYH	day count ten-digit value
3:0	DAYL	day count value

Decimal representation of day 1:31, automatic calculation of leap years and months. The specific representation is as follows:

month	Day count indication
February (ordinary 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: indeterminate

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	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	Function 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, write other values, set the

be ignored. The corresponding relationship between the weekly count values is as follows:

week	Week count representation
Sunday	00H
on 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: indeterminate

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	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	Function description
31:5	Reserved	-
4:0	MON	Month count value

Decimal 1:12 means 1:12 month. Please write the correct decimal 1:12 BCD code, write other values, set the  
be ignored.

### 13.4.9 Year Count Register (RTC\_YEAR )

Address offset: 0x020

Reset value: indeterminate

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	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					

bit	symbol	Function description
31:8	Reserved	-
7:4	YEARH year count	ten-digit value
3:0	YEARL year one-digit count value	

Decimal 0:99 means year 0:99. Counting based on month rounds. Automatically calculate leap years such as: 00, 04, 08, ..., 92,  
96 et al. Please write the correct decimal year count value, writing an incorrect value will be ignored.



### 13.4.10 Minutes Alarm Register (RTC\_ALMMIN)

Address offset: 0x024

Reset value: indeterminate

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	20	19	18	17	16	
Reserved																
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
15	14	13	12	11	10	9	-	8	7	6	5	4	3	2	1	0
Reserved								ALMMNH		ALMMINL						
								R/W		R/W						

bit	symbol	Function description
31:6	Reserved	-
5:4	ALMMNH minute alarm clock match value ten digits	
3:0	ALMMINL minute alarm clock match value one digit	

Please set the BCD code of decimal 0:59. Write other values and no alarm match will occur.

### 13.4.11 Time Alarm Register (RTC\_ALMHOUR)

Address offset: 0x028

Reset value: indeterminate

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	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	Function description
31:6	Reserved	-
5:4	Alarm clock 10-digit matching value at ALMHOURH	
3:0	ALMHOURL when the alarm one digit match value	

Please set the correct alarm matching value according to the time system, otherwise the time alarm matching will not occur.



### 13.4.12 Weekly Alarm Register (RTC\_ALMWEEK)

Address offset: 0x02C

Reset value: indeterminate

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
Reserved															
-															
15      14      13      12      11      10      9      -      7      6      5      4      3      2      1      0															
Reserved															
-															
ALMWEEK															
R/W															

bit	symbol	Function description
31:7	Reserved	-
6:0	ALMWEEK	<p>Weekly alarm matching value.</p> <p>b0:b6 correspond to Sunday:Saturday, respectively, when it is set to "1", it means that the alarm clock is valid on that day of the week.</p> <p>For example, b0=1, b5=1 means the alarm clock setting is valid on Sunday and Friday.</p>

Please set the correct alarm matching value according to the time system, otherwise the time alarm matching will not occur.

### 13.4.13 Clock Error Compensation Register (RTC\_COMPEN)

Address offset: 0x030

\*This register is only valid for power-on reset, reset value: 0x000000020

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	CPS	TSS	7	6	5	4	3	2	1	0
EN	Reserved									CR						
										R/W						

bit	Symbolic function	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 By setting the compensation value, an accuracy compensation of +/-0.96ppm per second can be performed. Compensation value is 9 bits with small The 2's complement of the number point, the last 5 digits are the fractional part. Compensable range 274.6ppm: 212.6ppm. least differential Error +/-0.48ppm. Minimum resolution 0.96ppm. Please refer to the following table for specific compensation accuracy:	<table border="1"> <tr> <td>Compensation value setting compensation number</td> <td></td> </tr> <tr> <td>EN CR[8:0]</td> <td></td> </tr> <tr> <td>1</td> <td>1 0 0 0 0 0 0 0 0 -274.6 ppm</td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td></td> <td>1 0 0 0 0 0 0 0 1 -273.7 ppm</td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>:</td> <td>:</td> <td>:</td> <td>:</td> <td>:</td> <td>:</td> </tr> <tr> <td>0 0 0 0 1</td> <td></td> <td>1</td> <td>1</td> <td>1</td> <td>1 -0.95 ppm</td> </tr> <tr> <td>0 0 0 1 0 0 0 0 0 0 ppm</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>:</td> <td>:</td> <td>:</td> <td>:</td> <td>:</td> <td>:</td> </tr> <tr> <td>0 1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1 0 +211.7 ppm</td> </tr> <tr> <td>0 1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1 1 +212.6 ppm</td> </tr> <tr> <td>0</td> <td>XXXXXXXXXX</td> <td>no compensation</td> <td></td> <td></td> <td></td> </tr> </table>	Compensation value setting compensation number		EN CR[8:0]		1	1 0 0 0 0 0 0 0 0 -274.6 ppm						1 0 0 0 0 0 0 0 1 -273.7 ppm					:	:	:	:	:	:	0 0 0 0 1		1	1	1	1 -0.95 ppm	0 0 0 1 0 0 0 0 0 0 ppm						:	:	:	:	:	:	0 1	1	1	1	1	1 0 +211.7 ppm	0 1	1	1	1	1	1 1 +212.6 ppm	0	XXXXXXXXXX	no compensation			
Compensation value setting compensation number																																																													
EN CR[8:0]																																																													
1	1 0 0 0 0 0 0 0 0 -274.6 ppm																																																												
	1 0 0 0 0 0 0 0 1 -273.7 ppm																																																												
:	:	:	:	:	:																																																								
0 0 0 0 1		1	1	1	1 -0.95 ppm																																																								
0 0 0 1 0 0 0 0 0 0 ppm																																																													
:	:	:	:	:	:																																																								
0 1	1	1	1	1	1 0 +211.7 ppm																																																								
0 1	1	1	1	1	1 1 +212.6 ppm																																																								
0	XXXXXXXXXX	no compensation																																																											

Compensation principle description and calculation:

Since the counter uses a 32.768KHz clock to count, if you need to compensate for the accuracy per second, you can only press

According to the integer period compensation of 32.768KHz, the minimum unit of compensation per second is  $(1/32768)*106=30.5\text{ppm}$ ,  
cannot meet the requirements of high precision.

Then to achieve high-precision clock compensation under the 32.768KHz count clock, it is necessary to adjust the algorithm.



Adjust the maximum compensation period by 32 times. If the minimum unit that can only be compensated is 30.5ppm, the

The compensation unit per second becomes  $30.5\text{ppm}/32=0.96\text{ppm}$ . It meets the clock compensation requirements with higher precision.

And the compensation occurs in a relatively uniform range every 32 seconds. Therefore, 5 decimal places are introduced into this register set up.

The set value is calculated as follows:

$$\text{CR[8 : 0]} = \frac{\text{Compensation target value [ppm]}}{10} + 0001.00000\text{B}$$

ÿ Take 2's complement

If the compensation target value is +20.6ppm, the corresponding register value is calculated as follows:

$$\begin{aligned}\text{CR[8:0]} &= (20.3 \times 215/106) \text{ 2's complement} + 0001.00000\text{B} \\ &= (0.6651904) \text{ 2's complement} + 0001.00000\text{B} \\ &= 0000.10101\text{B} + 0001.00000\text{B} \\ &= 0001.10101\text{B}\end{aligned}$$

If the compensation target value is -20.6ppm, the corresponding register value is calculated as follows:

$$\begin{aligned}\text{CR[8:0]} &= (-20.3 \times 215/106) \text{ 2's complement} + 0001.00000\text{B} \\ &= (-0.6651904) \text{ 2's complement} + 0001.00000\text{B} \\ &= 1111.01011\text{B} + 0001.00000\text{B} \\ &= 0000.01011\text{B}\end{aligned}$$

## 14 Watchdog Timer (WDT)

### 14.1 Introduction to WDT

WDT can be used to detect and resolve faults caused by software errors. When the WDT counter reaches the set overflow time

After that, an interrupt will be triggered or a system reset will be generated. The WDT is driven by a dedicated 10KHz on-chip oscillator.

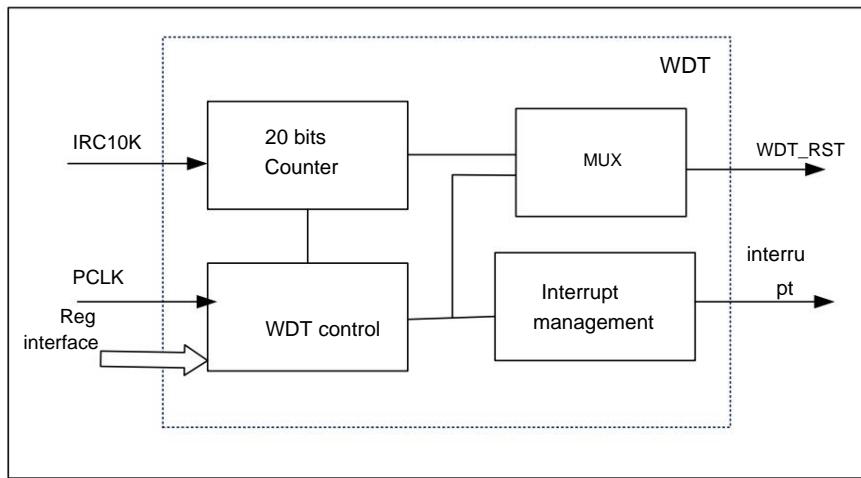


Figure 14-1 Overall block diagram of WDT

## 14.2 WDT function description

ÿ 20Bit free running up counter, overflow time can be configured as 1.6ms – 50s.

ÿ The action after overflow can be configured as interrupt or reset.

ÿ The WDT clock is provided by an independent RC oscillator and can work in Sleep and DeepSleep modes.

ÿ The WDTCON register can only be modified when the WDT is not activated to prevent inadvertent modification after activation.

Configuration of WDT.

### 14.2.1 Interrupt after WDT overflow

In this mode, the WDT will periodically generate interrupts at the set time. Need to clear in interrupt service routine

WDT interrupt flag.

The configuration method is as follows:

Step1: Configure WDT\_CON. WOV and select WDT time-out 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 turn to start the WDT timer.

Step5: In the interrupt service routine, write 0x1E and 0xE1 to the WDT\_RST register 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

The WDT counter needs to be cleared before the WDT overflows to avoid a WDT reset.

The configuration method is as follows:

Step1: Configure WDT\_CON. WOV and select WDT counter overflow time.

Step2: Set WDT\_CON. WINT\_EN to 0, select to reset after WDT overflow.

Step3: Write 0x1E and 0xE1 to the WDT\_RST register in turn to start the WDT timer.

Step4: Write 0x1E and 0xE1 to the WDT\_RST register in turn to clear the WDT count before the WDT overflows

device.

Note: Since the WDT oscillator is a low-precision RC oscillator, it is strongly recommended that the WDT counter overflows when the count value reaches

The WDT is cleared before half the value.

## 14.3 WDT register description

Base address 0X40000C00

register	Offset address	description
WDT_RST	0X080	WDT clear control register
WDT_CON	0X084	WDT control register

Table 14-1 WDT register list

### 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 has been started, write 0x1E and 0xE1 to this register in turn to clear the WDT timer and interrupt flag. Chi.

### 14.3.2 WDT\_CON register

Offset address: 0x084

Reset value: 0x0000 000F

Note:

- This register can only be written when the 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	Res.		R/W	RO		R/W		

Bit symbol	description	
31:16	Reserved	Reserved bit, read as 0
15:8	WCNTL	WDT counter lower 8 bits
7	WDTINT	WDT interrupt flag 1: WDT interrupt has occurred, write 0xE and 0xE1 to the WDT_RST register to clear the interrupt flag. 0: No WDT interrupt occurred.
5	WINT_EN	Action configuration after WDT overflow 1: Interrupt after WDT overflow. 0: Reset after WDT overflow.
4	WDTR	WDT running flag 1: WDT is running 0: WDT stopped
3:0	WOV[3:0]	WDT time-out 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 Universal Synchronous Asynchronous Receiver/Transmitter (UART)

### 15.1 Overview

This product has 2 universal UART modules (UART0/1), the universal synchronous asynchronous transceiver (UART) can

Actively perform full-duplex data exchange with external devices, it supports synchronous one-way communication as well as multi-processor communication. Commonly used

In short-distance, low-speed serial communication. The UART offers 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 support

Multiple working modes.

### 15.2 Structure block diagram

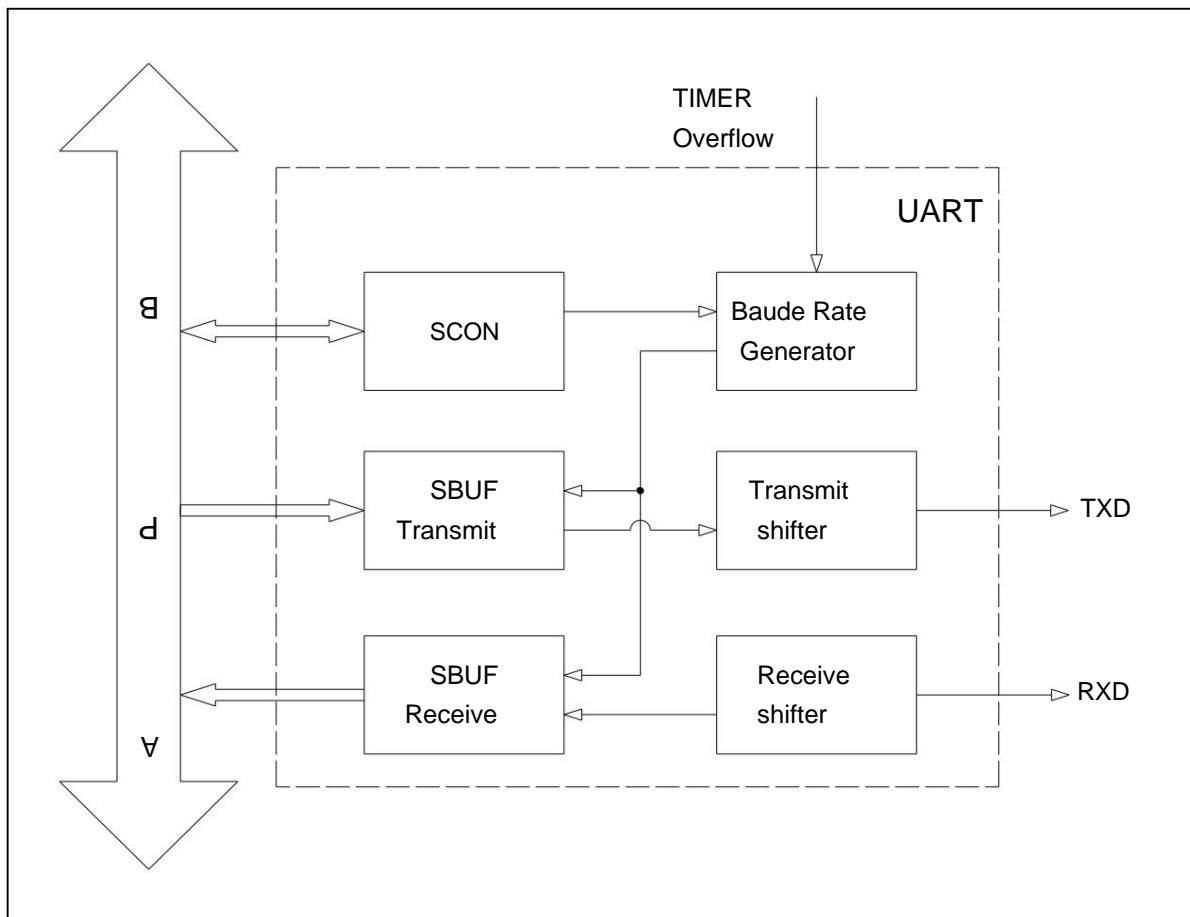


Figure 15-1 UART structure block diagram



## 15.3 Main Features

The generic UART module supports the following basic functions:

ÿ Full-duplex transmission, half-duplex transmission

ÿ Programmable serial communication function

– Two character lengths: 8 bits, 9 bits

– Mode0/1/2/3 four transmission modes

ÿ 16-bit baud rate generator

ÿ Multi-machine communication

ÿ Automatic address recognition

## 15.4 Functional Description

### 15.4.1 Working Mode

The UART supports multiple working modes: synchronous half-duplex mode, asynchronous full-duplex mode. Via `UARTx_SCON.SM0`

Combined with `UARTx_SCON.SM1`, you can configure various working modes required.

#### 15.4.1.1 Mode0~Mode3 function comparison

Configure `UARTx_SCON.SM` to select different transmission modes: Mode0~Mode3. The master of these four working modes

To function pairs are shown in the following table:

Working mode transmission bit width		data composition	baud rate
Mode0	Sync mode half duplex	8bit Data(8bit)	= $\frac{1}{12}$
Mode1	Asynchronous mode full duplex	10bit Start (1bit) + Data (8bit) + Stop(1bit)	= $\frac{( + 1)}{32 \times (65536 \times )}$
Mode2	Asynchronous mode full duplex	11bit Start (1bit) + Data(8bit) + B8(1bit) + Stop(1bit)	= $\frac{( + 1)}{64}$
Mode3	Asynchronous mode full duplex	11bit Start (1bit) + Data (8bit) + B8(1bit) + Stop(1bit)	= $\frac{( + 1)}{32 \times (65536 \times )}$

Table 15-1 Mode0/1/2/3 Data Structure

Note:

- Mode0 can only be used as a master to send UART synchronization shift clock, not as a slave to receive externally input UART synchronization shift clock.
- Represents the current PCLK frequency.
- For the definition of DBAUD, see `UARTx_SCON`.
- TM is the TIMER count value. Note that TIMER must be configured in 16-bit auto-reload mode, counting registers and reloading

The registers must be written with the TM value.

#### 15.4.1.2 Mode0 (Sync Mode, Half Duplex)

When working in Mode0, the UART works in synchronous mode with a baud rate of 1/12 of the fixed PCLK clock.

The data received by the UART is input by RXD, the data sent by the UART is output by TXD, and RXD is input and output at this time. outgoing port. The UART synchronous shift clock is output by TXD, which is the output port at this time. Note that this mode only It can transmit the synchronous shift clock as a master, but cannot receive the clock externally as a slave. In this mode, the transmission The data bit width can only be 8 bits, there is no start bit and end bit.

Clear UARTx\_SCON.SM0 and UARTx\_SCON.SM1 to enter Mode0 operating mode.

When sending data, clear the UARTx\_SCON.REN bit and write the data to the UARTx\_SBUF register. this , the transmit data will be output from RXD (low-order first, high-order last), and the synchronous shift clock will be output from TXD.

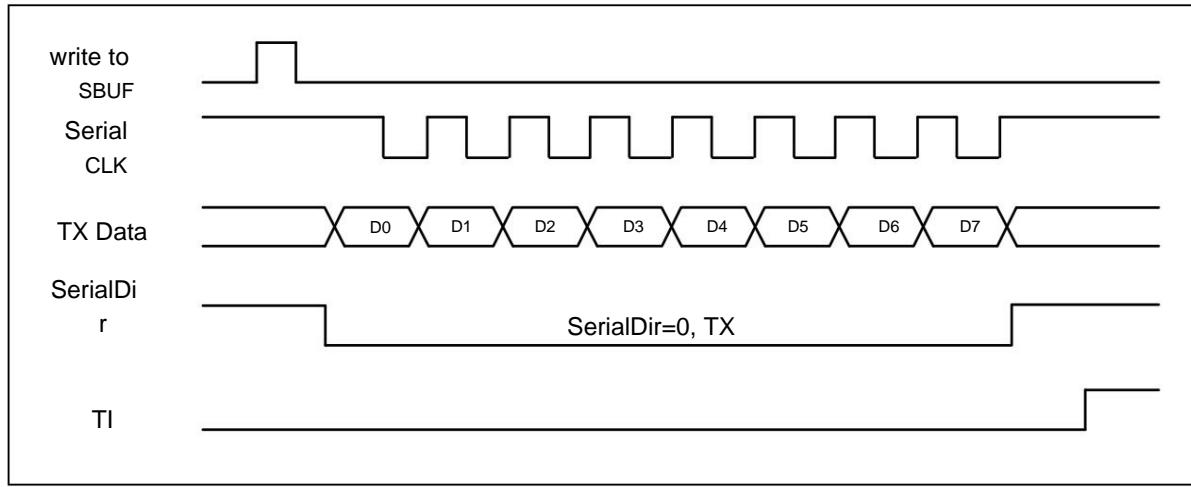


Figure 15-2 Mode0 send data

When data is received, set the UARTx\_SCON.REN bit and clear the UARTx\_ISR.RI bit. when receiving

When finished, the data can be read from the UARTx\_SBUF register. At this time, receive data is input from RXD (low-order first, High bit last), the synchronous shift clock is output from TXD.

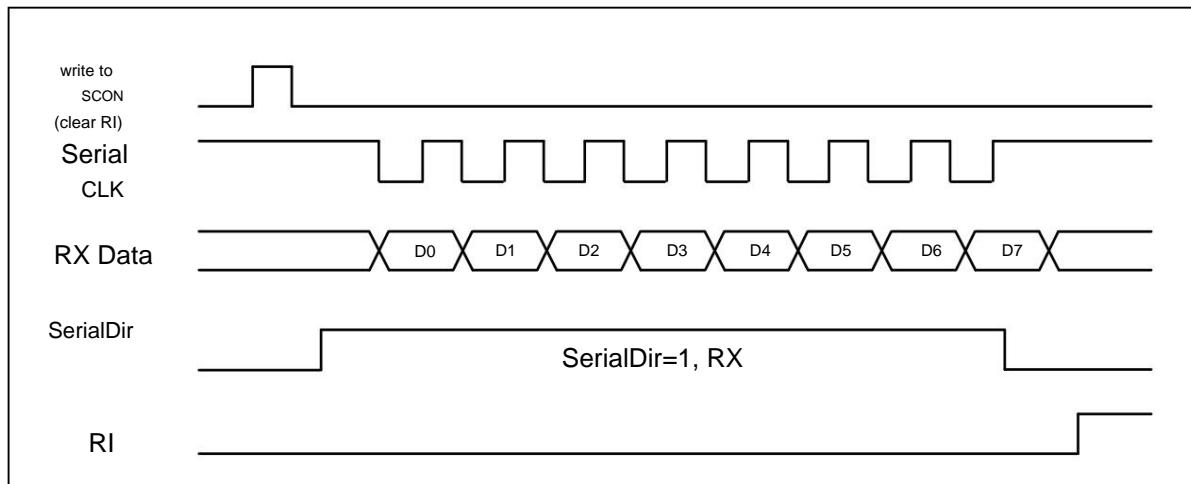


Figure 15-3 Mode0 receive data

#### 15.4.1.3 Mode1 (asynchronous mode, full duplex)

When working in Mode1, sending data is sent through TXD, and receiving data is received through RXD. the data

It consists of 10 bits: start bit "0", followed by 8 data bits (low bit first, high bit last), and finally the end bit.

Beam bit "1".

In this mode, the baud rate is generated by the timer module and is programmable. The baud rate of UART0 is set by TIMER0

Generated, 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, regardless of the value of UARTx\_SCON.REN, write the sent data into UARTx\_SBUF

In the register, the data will be shifted out from TXD (low order first, high order last).

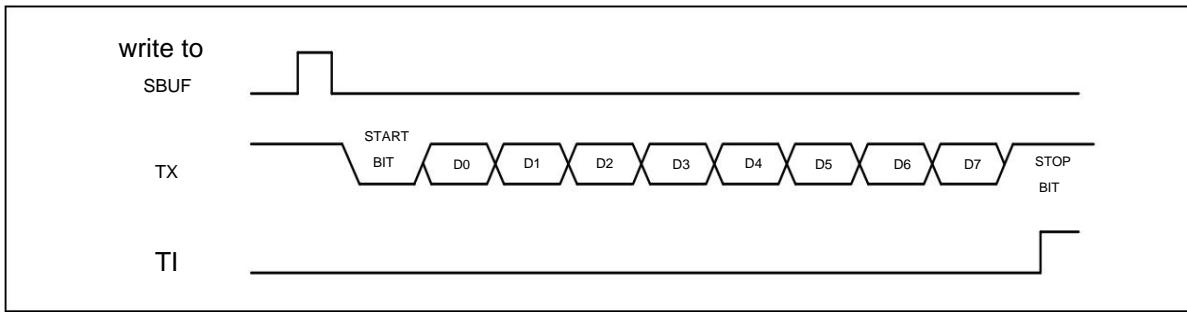


Figure 15-4 Mode1 send data

To receive data, set the UARTx\_SCON.REN bit to 1 and clear the UARTx\_ISR.RI bit to 0. start to pick up

Receive data on RXD (low-order first, high-order last), when the reception is completed, it can be read from the UARTx\_SBUF register read out.

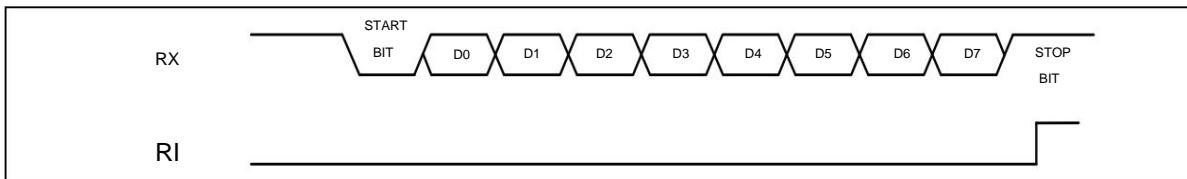


Figure 15-5 Mode1 receive data

#### 15.4.1.4 Mode2 (asynchronous mode, full duplex)

When working in Mode2, sending data is sent through TXD, and receiving data is received through RXD. the data

It consists of 11 bits: start bit "0", followed by 8 data bits, 1 TB8 bit and end bit. additional

The TB8 bit is used in a multi-computer communication environment. When TB8=1, it indicates that the received frame is an address; when TB8=0,

Indicates that a data frame is received. 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 clear `UARTx_SCON.SM1` to 0 to enter Mode2 working mode.

When sending data, regardless of the value of `UARTx_SCON.REN`, write the sent data into `UARTx_SBUF`

In the register, the data will be shifted out from TXD (low order first, high order last).

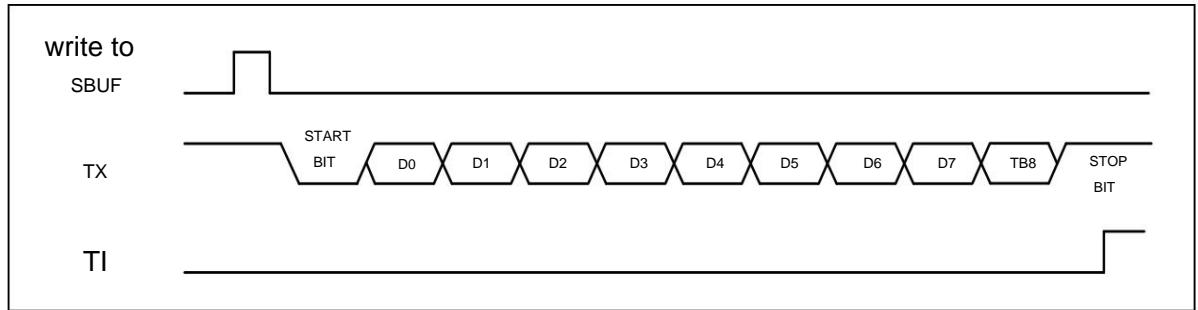


Figure 15-6 Mode2 send data

To receive data, set the `UARTx_SCON.REN` bit to 1 and clear the `UARTx_ISR.RI` bit to 0. start to pick up

Receive data on RXD (low-order first, high-order last), when the reception is completed, it can be read from the `UARTx_SBUF` register read out.

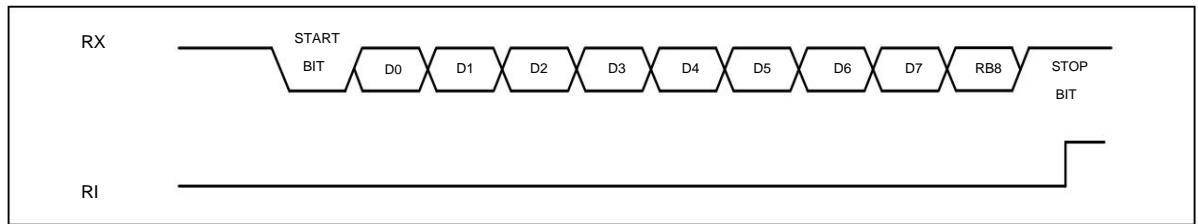


Figure 15-7 Mode2 receive data

#### 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 Mode3

The baud rate is generated by TIMER, not independently generated by the device like Mode2. Baud rate of Mode3

It is programmable and the baud rate is generated in the same way as Mode1. In this product, the baud rate of UART0 is determined by

TIMER0 is generated, 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, regardless of the value of `UARTx_SCON.REN`, write the sent data into `UARTx_SBUF`

In the register, the data will be shifted out from TXD (low order first, high order last).

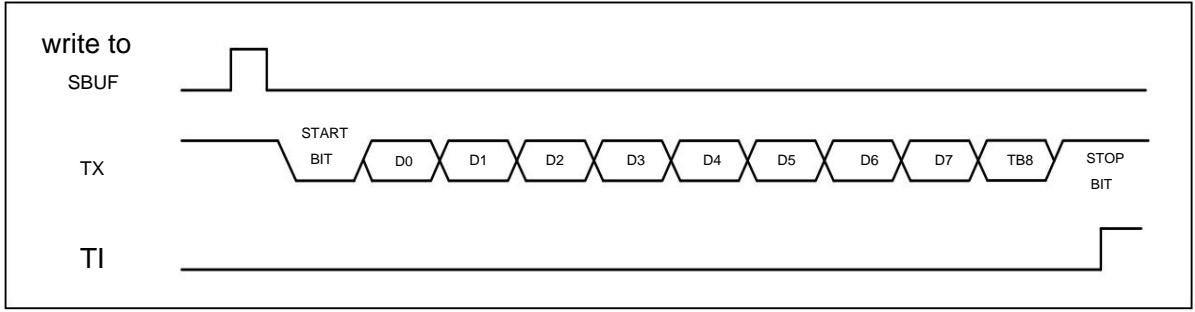


Figure 15-8 Mode3 send data

To receive data, set the `UARTx_SCON.REN` bit to 1 and clear the `UARTx_ISR.RI` bit to 0. start to pick up

Receive data on `RXD` (low-order first, high-order last), when the reception is completed, it can be read from the `UARTx_SBUF` register read out.

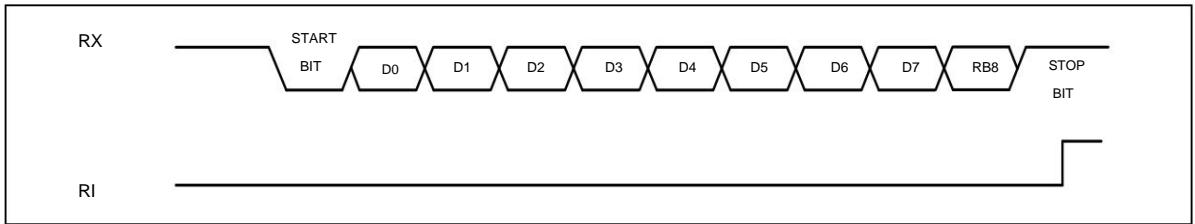


Figure 15-9 Mode3 receive data

### 15.4.2 Baud rate generation

The formulas for the baud rate generated by Mode0~Mode3 are different, as shown below:

$$\text{Mode0 baud rate generation formula: } = \frac{1}{12}$$

$$\text{Mode1 baud rate generation formula: } = \frac{(+1)}{32(65536)}$$

$$\text{Mode2 baud rate generation formula: } = \frac{(+1)}{64}$$

$$\text{Mode3 baud rate generation formula: } = \frac{(+1)}{32(65536)}$$

Note:

- Represents the current PCLK frequency.
- For the definition of DBAUD, see UARTx\_SCON.
- TM is the TIMER count value. Note that TIMER must be configured in 16-bit auto-reload mode, counting registers and reloading

The registers must be written with the TM value.

**15.4.2.1 Mode1/Mode3 baud rate setting example**

baud rate	<b>PCLK = 1MHz</b>					
	Dual baud rate			single baud rate		
	<b>CNTs</b>	Actual baud rate	error %	<b>CNTs</b>	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	<b>PCLK = 4MHz</b>					
	Dual baud rate			single baud rate		
	<b>CNTs</b>	Actual baud rate	error %	<b>CNTs</b>	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	<b>PCLK = 10MHz</b>					
	Dual baud rate			single baud rate		
	<b>CNTs</b>	Actual baud rate	error %	<b>CNTs</b>	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	<b>PCLK = 14MHz</b>					
	Dual baud rate			single baud rate		
	CNTs	Actual baud rate	error %	CNTs	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%	seventy three	19021.74	-0.93%
38400	seventy three	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	<b>PCLK = 20MHz</b>					
	Dual baud rate			single baud rate		
	CNTs	Actual baud rate	error %	CNTs	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	seventy two	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	<b>PCLK = 24MHz</b>					
	Dual baud rate			single baud rate		
	CNTs	Actual baud rate	error %	CNTs	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	<b>PCLK = 2MHz</b>					
	Dual baud rate			single baud rate		
	CNTs	Actual baud rate	error %	CNTs	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	-13.62%
57600	2	62500.00	8.51%	1	62500.00	8.51%
76800	2	62500.00	-13.62%	1	62500.00	-13.62%
115200	1	125000.00	8.51%	1	62500.00	-45.75%

baud rate	<b>PCLK = 8MHz</b>					
	Dual baud rate			single baud rate		
	CNTs	Actual baud rate	error %	CNTs	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	<b>PCLK = 11.0592 MHz</b>					
	Dual baud rate			single baud rate		
	CNTs	Actual baud rate	error %	CNTs	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	-1.00%
115200	6	115200.00	0.00%	3	115200.00	0.00%

baud rate	<b>PCLK = 16MHz</b>					
	Dual baud rate			single baud rate		
	CNTs	Actual baud rate	error %	CNTs	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	<b>PCLK = 32MHz</b>					
	Dual baud rate			single baud rate		
	CNTs	Actual baud rate	error %	CNTs	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	<b>PCLK = 22.12 MHz</b>					
	Dual baud rate			single baud rate		
	CNTs	Actual baud rate	error %	CNTs	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	<small>twenty four</small>	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, the UART has a frame error detection function, and the hardware will automatically detect the number of frames received

Whether it has a valid Stop bit. If the received data is that the hardware did not receive a valid Stop bit as expected, the

Synchronization failure or excessive noise, UARTx\_ISR.FE is set to 1. The UARTx\_ISR.FE bit is set by hardware,

The software clears to 0. If the software does not clear it to 0 in time, the subsequent data received will not be reset even if it has a valid Stop bit.

The UARTx\_ISR.FE flag is cleared to 0.



## 15.6 Multi-machine communication

Mode2/3 has the function of multi-computer communication, for which 1 bit TB8/RB8 is added to its frame format. Will

Set UARTx\_SCON.SM2 to "1" to enable the multi-computer communication bit.

When the multi-machine communication bit is turned on, when sending data, the host can distinguish the current frame through UARTx\_SCON.TB8

Whether it is an address frame (UARTx\_SCON.TB8=1) or a data frame (UARTx\_SCON.TB8=0). When receiving data,

The slave ignores the current received frame with the RB8 bit (bit 9) being "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 receiving

break.

When it is an address frame, since the automatic address recognition function in the multi-machine communication is enabled, the slave machine can detect the received address and

Whether its own address matches.

If the addresses match, the slave sets UARTx\_SCON.RB8 to 1 and UARTx\_ISR.RI to 1. Slave software

After seeing UARTx\_SCON.RB8=1 and UARTx\_ISR.RI=1, clear the UARTx\_SCON.SM2 bit to "0",

Accept a data frame.

If the addresses do not match, the master is not addressing the slave, and the slave hardware keeps UARTx\_SCON.RB8 and

UARTx\_ISR.RI is "0", the software keeps the UARTx\_SCON.SM2 bit as "1" and continues to be in the address monitoring state.

Note: If necessary, you can also open the multi-computer communication bit in Mode1, at this time, 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-computer communication bit is turned on (UARTx\_SCON.SM2 is set to "1"), the automatic address recognition function will also be turned on. the function

Can be implemented in hardware so that the slave can detect that each address frame is received, and if the address matches the slave address,

The receiver will give the UARTx\_ISR.RI receive flag. If the addresses do not match, the receiver will not give any

receive sign.

### 15.7.1 Given address

The UARTx\_SADDR register of the UART device is used to indicate the given address of its own device,

The UARTx\_SADEN registers are address masks that can be used to define don't care bits in the address. when

A bit of UARTx\_SADEN is "0", which means that the address of this bit is an irrelevant bit, that is to say, during the address matching process

, this bit address does not participate in address matching. These don't care bits add addressing flexibility, allowing the host to simultaneously

Addresses one or more slave devices. Note that if a unique matching address needs to be given, UARTx\_SADEN is sent

register must be set to 8'hFF.

**GivenAddr = SADDR & SADEn**

#### 15.7.1.1 Broadcast address

The broadcast address is used to address all slave devices at the same time, and the general broadcast address is 8'hFF.

**BroadCastAddr = SADDR|SADEn**

#### 15.7.1.2 Examples

Suppose the UARTx\_SADDR and UARTx\_SADEn of a slave are configured as follows:

SADDR: 8'b01101001

SADEn: 8'b11111011

Then its given address and broadcast address are as follows:

Given: 8'b0110101x01

Broadcast: 8'b111111x11

It can be seen that the master can use four addresses to address the slave, namely:

8'b01101001 and 8'b01101101 (given address)

8'b11111011 and 8'b11111111 (broadcast address).

## 15.8 Transceiver Cache

### 15.8.1 Receive buffer

The receiving end of the general UART (UART0/1) has a receiving buffer with a frame length (8/9bits), that is to say, when a frame

After the data is received, the data in the receive buffer will be kept until the Stop bit of the next frame of data is received.

After completion, the receive buffer will be updated with a new frame of data.

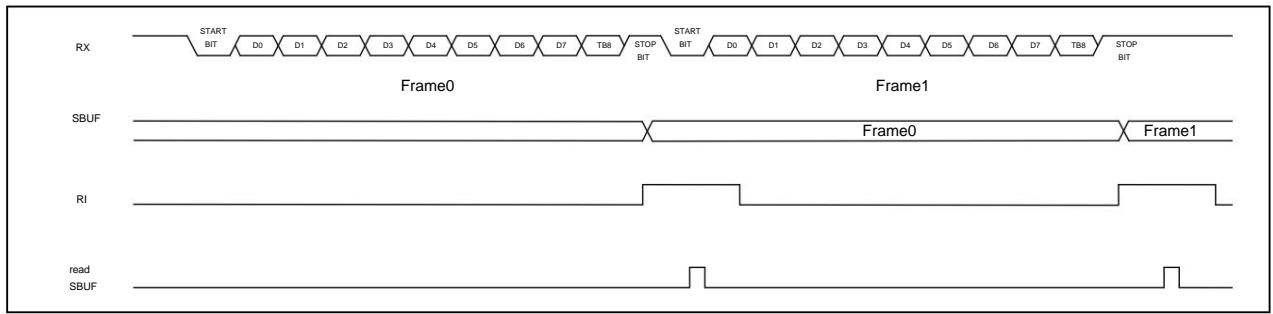


Figure 15-10 Receive buffer

### 15.8.2 Send Buffer

The general purpose UART (UART0/1) transmitter does not support transmit buffering. If in the process of sending data, fill in

The UARTx\_SBUF register will corrupt the data currently being sent. Software should avoid this operation.

## 15.9 Registers

UART0 base address: 0x4000 0000

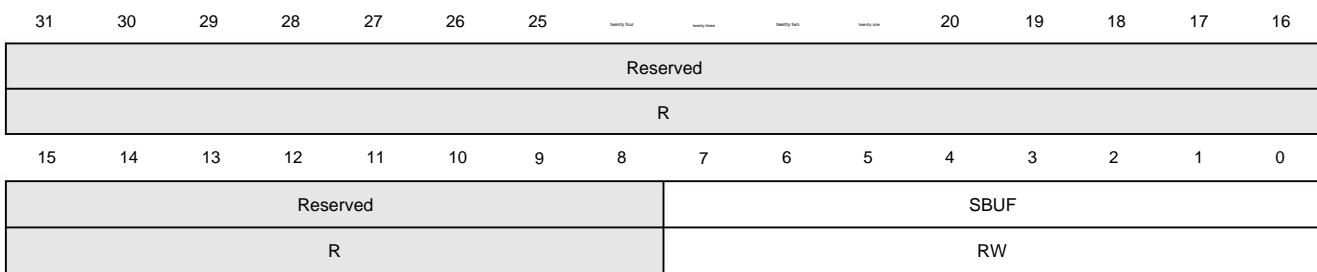
UART1 base address: 0x4000 0100

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 Clear Register

### 15.9.1 Data Register (UARTx\_SBUF)

Offset address: 0x00

Reset value: 0x0000 0000



bit mark		Function description
31:8	Reserved	
7:0	When SBUF sends 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							20	19	18	17	16
Reserved																	
R																	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reserved							DBAs UD	Reser ved	SM01	SM2 REN	TB8		RB8 TIEN	RIEN			
R							RW R		RW	RW RW	RW RW	RW RW					

bit	mark	Function description
31:10	Reserved	
9	DBAUD baud rate magnification 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-computer communication function  1: Enable multi-computer communication function	
4	REN receive enable control  Mode0: 0: Send, 1: Receive  Others: 0: Send, 1: Receive/Send	
3	TB8 TB8 bit to be sent when sending data	
2	RB8 RB8 bit received when data is received	
1	TIEN send complete interrupt enable  0: Disable transmission completion interrupt  1: Enable send completion interrupt	
0	RIEN Receive complete interrupt enable  0: Disable reception completion interrupt  1: Enable receive completion interrupt	

### 15.9.3 Address Register (UARTx\_SADDR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25								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 mark		Function 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								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 mark		Function description
31:8	Reserved	
7:0	SADEN Slave Device Address Mask	

## 15.9.5 Flags Register (UARTx\_ISR)

Offset address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25				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 sign		describe
31:3	Reserved	
2	FE Receive frame error flag; set by hardware, cleared by software 1: FE interrupt flag is valid 0: FE interrupt flag is invalid	
1	TI transmit complete interrupt flag bit; set by hardware, cleared by software 1: TI interrupt flag is valid 0: TI interrupt flag is invalid	
0	RI Receive complete interrupt flag bit; 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

Reset value: 0x0000 0000

31	30	29	28	27	26	25									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			
														CLR	CLR	CLR			
R														WWW					

bit mark		Function description
31:3	RESERVED	
2	FECLR	Receive Frame Error Flag Clear Write 0 to clear, write 1 to be invalid
1	TICLR	transmit complete interrupt flag clear Write 0 to clear, write 1 to be invalid
0	RICLR	receive complete interrupt flag clear Write 0 to clear, write 1 to be invalid



## 16 Low Power Synchronous Asynchronous Transceivers (LPUARTs)

### 16.1 Overview

This product comes with 1 LPUART module, LPUART contains all necessary hardware support to make the minimum power consumption. Asynchronous serial communication can be carried out under the produce.

In order to support low-power 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 mode, turn off the high-frequency PCLK clock, turn on the low-frequency SCLK clock, The LPUART can still send and receive data normally.

SCLK clock source can be selected: PCLK, external low-speed clock (XTL), internal low-speed clock (RCL). exist

When LPMODE=1, the SCLK clock also supports 1/2/4/8/16/32/64/128 times prescaler.

Note that when LPMODE=0, the LPUART receives the Toggle output signal of the TIMER2 clock instead of Overflow signal, so the Toggle output of TIMER2 must be enabled.

## 16.2 Structure block diagram

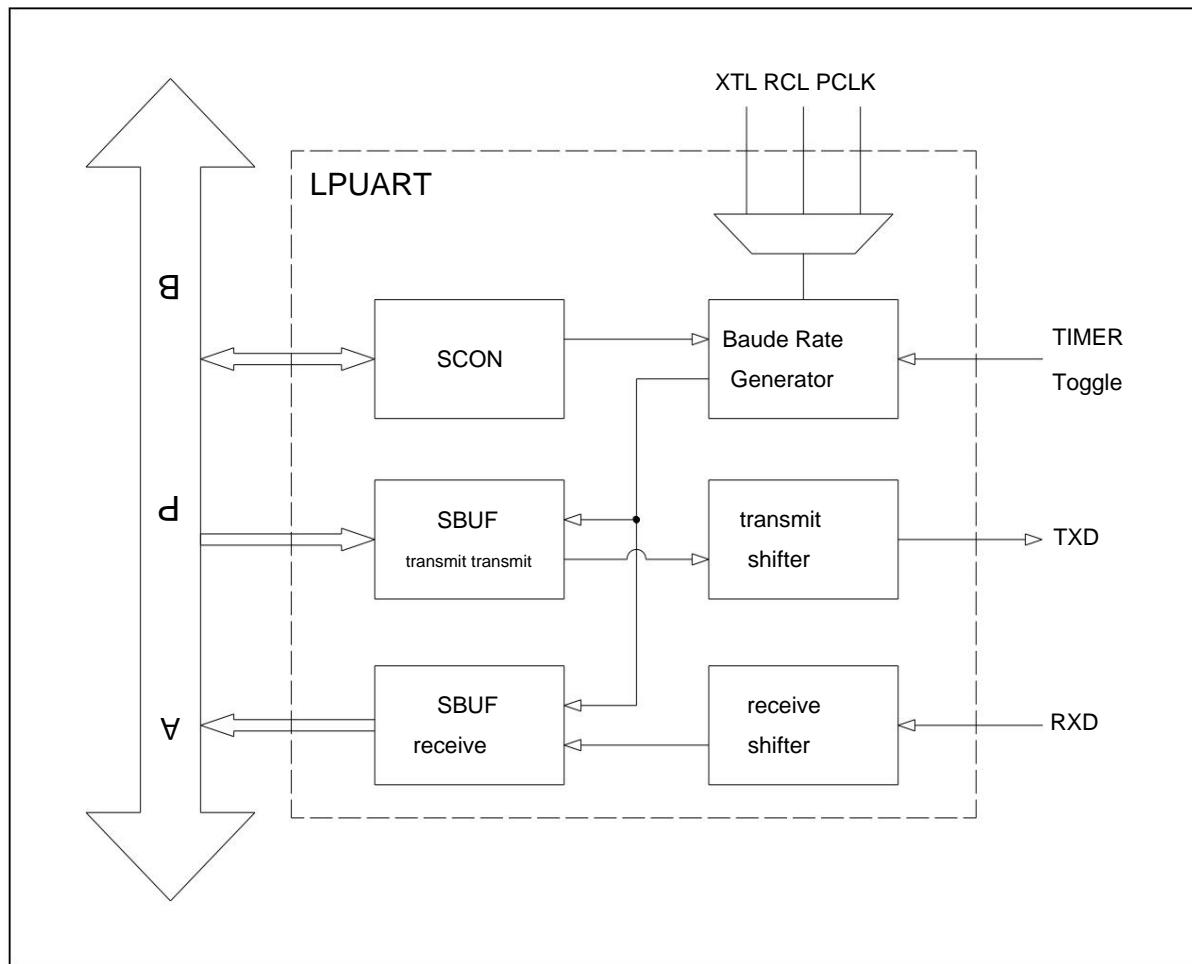


Figure 16-1 LPUART block diagram



## 16.3 Main Features

The LPUART module supports the following basic functions:

- ÿ Configure the clock PCLK
- ÿ Transmission clock SCLK (SCLK can choose XTL, RCL and PCLK)
- ÿ Send and receive data in system low power mode
- ÿ Full-duplex transmission, half-duplex transmission
- ÿ Programmable serial communication function
  - Two character lengths: 8 bits, 9 bits
  - Mode0/1/2/3 four transmission modes
- ÿ 16-bit baud rate counter
- ÿ Multi-machine communication
- ÿ Automatic address recognition



## 16.4 Functional Description

### 16.4.1 Configuration Clock and Transfer Clock

The LPUART module has two clocks: the configuration clock PCLK and the transmission clock SCLK.

ÿ Configure the clock

The configuration clock is used for register configuration of the LPUART module by the system APB bus, and is fixed as PCLK.

ÿ Transfer clock

The transmission clock is used for the LPUART data transmission and reception logic, and XTL, RCL and PCLK can be selected. when SCLK

If it is XTL or RCL, when the system enters DeepSleep, the LPUART can still send and receive data normally.

### 16.4.2 Working Mode

LPUART supports multiple working modes: synchronous half-duplex mode, asynchronous full-duplex mode. by configuration

The value of LPUARTx\_SCON.SM can get the required working mode.

The LPUART adds an LPMODE control bit to the UART. When this bit is "1", only Mode1/3 is supported

working mode, and the baud rate generation method will also change.

#### 16.4.2.1 Mode0~Mode3 function comparison

When **LPMODE=0** :

Four transmission modes can be selected by configuring LPUARTx\_SCON.SM. The function comparison of each mode is as follows:

Working mode transmission bit width		data composition	baud rate
Mode0	Sync mode half duplex	8bit Data(8bit)	= $\frac{1}{12}$
Mode1	Asynchronous mode full duplex	10bit Start (1bit) + Data (8bit) + Stop(1bit)	= $\frac{(\text{SCLK} + 1)}{32 \times (65536 \text{ } \circ)}$
Mode2	Asynchronous mode full duplex	11bit Start (1bit) + Data(8bit) + B8(1bit) + Stop(1bit)	= $\frac{(\text{SCLK} + 1)}{64}$
Mode3	Asynchronous mode full duplex	11bit Start (1bit) + Data (8bit) + B8(1bit) + Stop(1bit)	= $\frac{(\text{SCLK} + 1) \times 32}{(65536 \text{ } \circ)}$

Table 16-1 Mode0/1/2/3 Data Structure

When **LPMODE=1**:

Mode2	Asynchronous mode full duplex	11bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop(1bit)	= $\frac{\text{SCLK}}{4}$
-------	----------------------------------	-------	--	---------------------------

Note:

- Mode0 can only transmit LPUART synchronous shift clock as a master, and cannot accept external input as a slave
- The LPUART synchronizes the shift clock.
- Represents the current SCLK frequency.
- For the definition of DBAUD, see LPUARTx\_SCON.
- TM is the TIMER count value. Note that the TIMER must be configured in 16-bit auto-reload mode, counting registers and reloading

The TM value must be written to the load register.

- PreScale is the prescale factor.

#### 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, wave

The bit rate is fixed at SCLK/12. The transmitted data width is fixed at 8 bits, without start bit and end bit. data

Both reception and transmission are achieved through the RXD pin; the synchronous shift clock is output from the TXD pin. Note that the TXD leads 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 transmit data will be output from RXD (low order first, high order last), and the synchronous shift clock will be output from TXD.

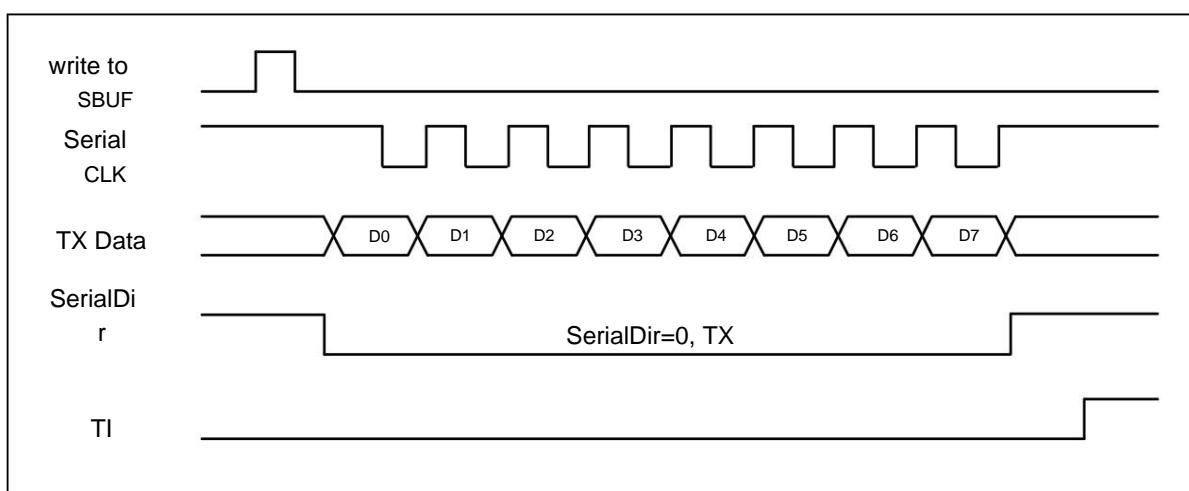


Figure 16-2 Mode0 send data

When receiving data, set LPUART\_SCON.REN to 1 and clear the LPUART\_ISR.RI bit. when connected

After receiving, the data can be read from the LPUART\_SBUF register. At this time, receive data from the RXD input (low

Bit first, high bit last), the synchronous shift clock is output from TXD.

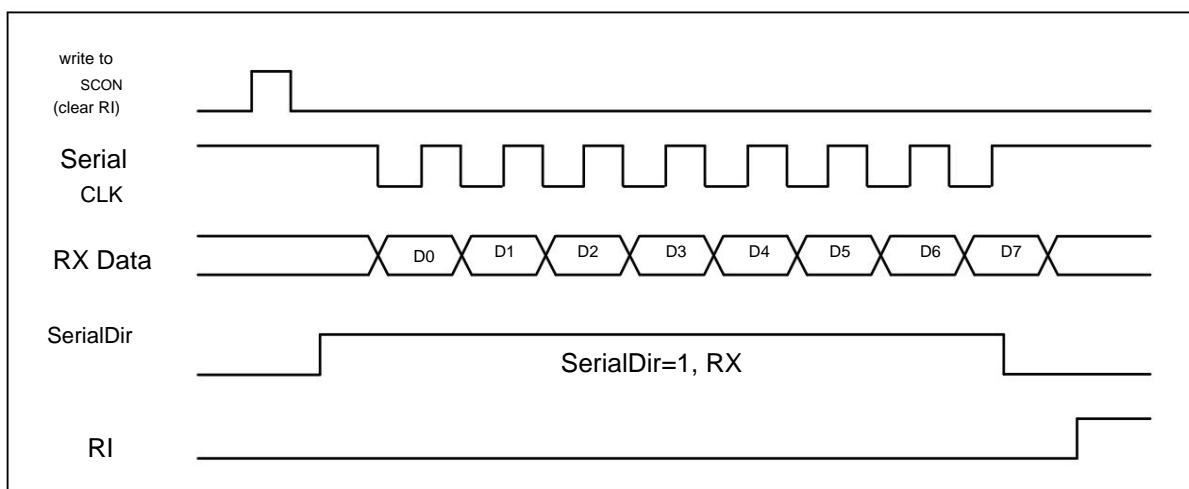


Figure 16-3 Mode0 receive data

#### 16.4.2.3 Mode1 (asynchronous mode, full duplex) data sending and receiving instructions

When working in Mode1, sending data is sent through TXD, and receiving data is received through RXD. the data

It consists of 10 bits: start bit "0", followed by 8 data bits (low bit first, high bit last), and finally the end bit.

Beam bit "1".

In this mode, the baud rate of the 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, Mode1 working mode is supported, but the baud rate calculation method is changed.

Check the baud rate generation chapter.

When sending data, regardless of the value of LPUART\_SCON.REN, write the sent data into LPUART\_SBUF

In the register, the data will be shifted out from TXD (low order first, high order last).

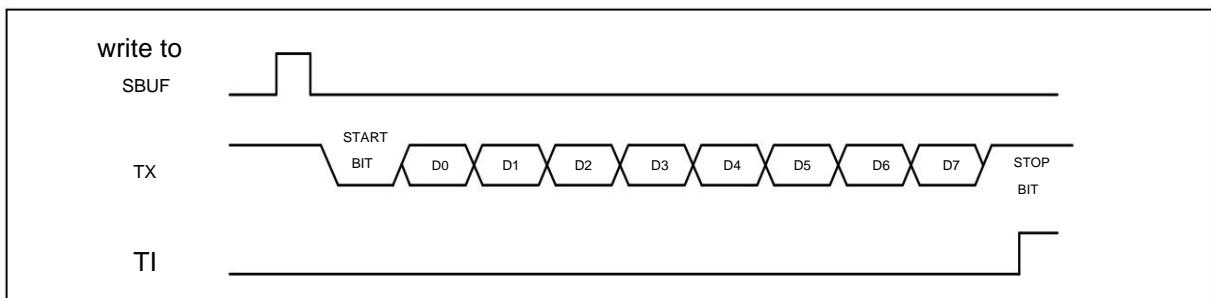


Figure 16-4 Mode1 send data

To receive data, set the LPUART\_SCON.REN bit to 1 and clear the LPUART\_ISR.RI bit to 0. open

Start to receive the data on RXD (low order first, high order last), when the reception is completed, you can read the data from LPUART\_SBUF

Register read.

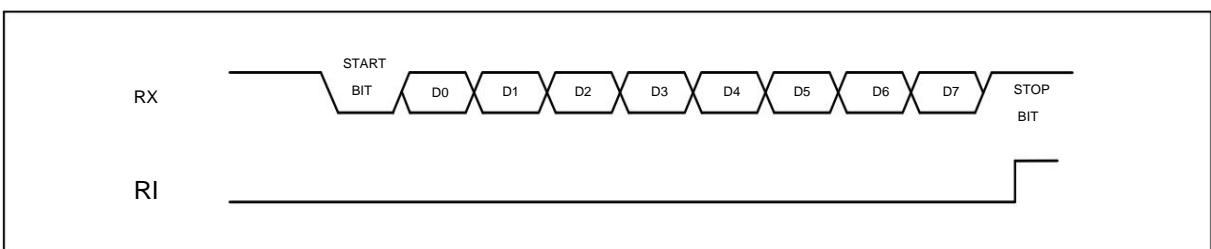


Figure 16-5 Mode1 receive data

#### 16.4.2.4 Mode2 (asynchronous mode, full duplex) data sending and receiving instructions

When working in Mode2, sending data is sent through TXD, and receiving data is received through RXD. the data

It consists of 11 bits: start bit "0", followed by 8 data bits, 1 TB8 bit and stop bit. additional

The TB8 bit is used in a multi-computer communication environment. When TB8=1, it indicates that the received frame is an address; when TB8=0,

Indicates that a data frame is received. 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 0 to enter Mode2 working mode.

When LPMODE=1, Mode2 working mode is not supported.

When sending data, regardless of the value of LPUART\_SCON.REN, write the sent data into

In the LPUART\_SBUF register, the data is shifted out from TXD (low order first, high order last).

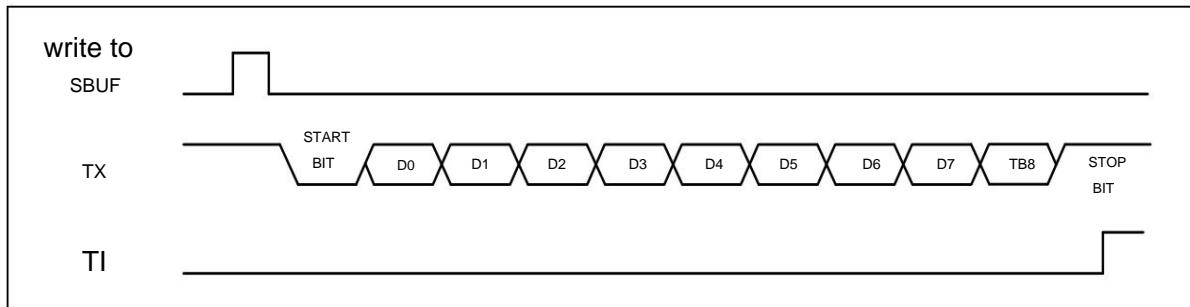


Figure 16-6 Mode2 send data

To receive data, set the LPUART\_SCON.REN bit to 1 and clear the LPUART\_ISR.RI bit to 0. open

Start to receive the data on RXD (low order first, high order last), when the reception is completed, you can read the data from LPUART\_SBUF

Register read.

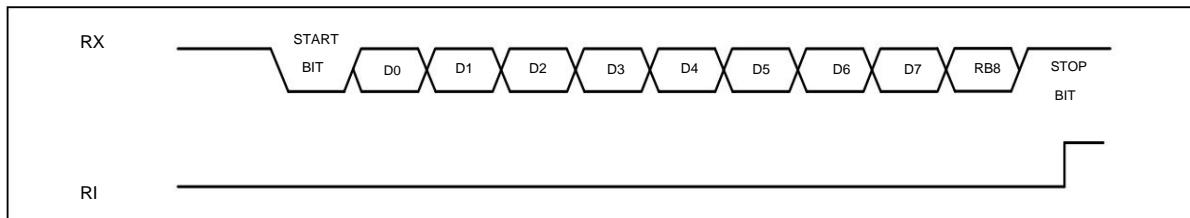


Figure 16-7 Mode2 receive data

#### 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 Mode3

The baud rate is generated by TIMER, not independently generated by the device like Mode2. Baud rate of Mode3

It is programmable and the baud rate is generated in the same way as Mode1.

Set LPUART\_SCON.SM0 to 1 and LPUART\_SCON.SM1 to 1 to enter Mode3 working mode.

When LPMODE=1, Mode3 working mode is supported. However, the baud rate calculation method has changed.

Check the baud rate generation chapter.

When sending data, regardless of the value of LPUART\_SCON.REN, write the sent data into

In the LPUART\_SBUF register, the data is shifted out from TXD (low order first, high order last).

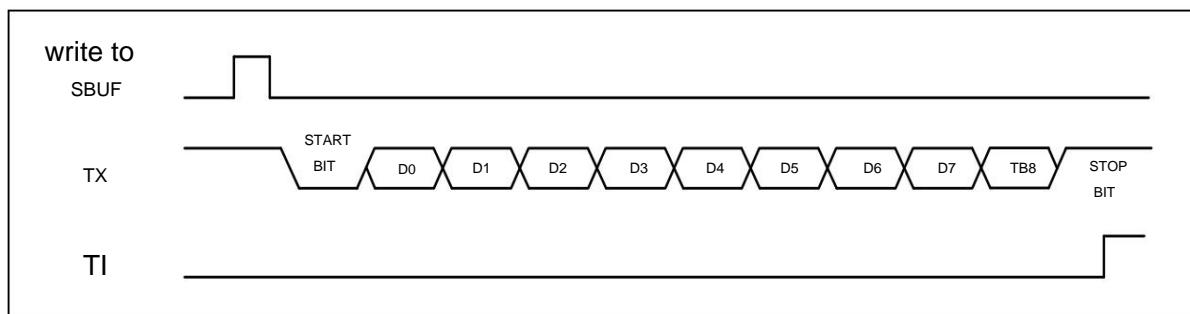


Figure 16-8 Mode3 send data

To receive data, set the LPUART\_SCON.REN bit to 1 and clear the LPUART\_ISR.RI bit to 0. open

Start to receive the data on RXD (low order first, high order last), when the reception is completed, you can read the data from LPUART\_SBUF

Register read.

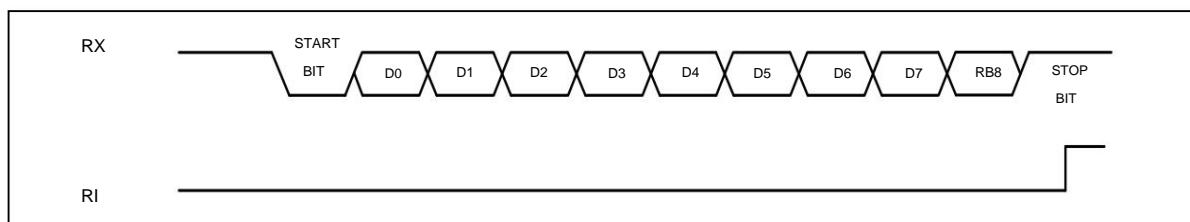


Figure 16-9 Mode3 receive data



### 16.4.3 Baud Rate Generation

#### LPMODE=0

The formula for the baud rate generated by Mode0~Mode3 is as follows:

$$\text{Mode0 baud rate generation formula: } = \frac{\text{---}}{12}$$

$$\text{Mode1 baud rate generation formula: } = \frac{(\text{---} + 1)}{32 \times (65536)}$$

$$\text{Mode2 baud rate generation formula: } = \frac{(\text{---} + 1)}{64}$$

$$\text{Mode3 baud rate generation formula: } = \frac{(\text{---} + 1)}{32 \times (65536)}$$

#### LPMODE=1

When LPMODE=1, Mode0 and Mode2 are not supported.

$$\text{Mode1, Mode3 baud rate generation formula: } = \frac{\text{---}}{\text{---}^4}$$

Note:

- Represents the current SCLK frequency.
- For the definition of DBAUD, see LPUARTx\_SCON.
- TM is the TIMER count value. Note that the TIMER must be configured in 16-bit auto-reload mode, counting registers and reloading

The TM value must be written to the load register.

- PreScale is the prescale factor.

### 16.5 Frame Error Detection

When working in Mode1/2/3, LPUART has frame error detection function, the hardware will automatically detect the received frame

Whether the data has a valid Stop bit. If a valid Stop bit is not received within the expected time when data is accepted, the

The LPUART\_ISR.FE bit is set in the event of synchronization failure or excessive noise. The LPUART\_ISR.FE bit is set by hard

If the software does not clear 0 in time, the subsequent received data will not be valid even if it has a valid Stop bit.

Will clear the LPUART\_ISR.FE flag to 0.



## 16.6 Multi-machine communication

Mode2/3 has the function of multi-computer communication, for which 1 bit TB8/RB8 is added to its frame format. Insert SCON.SM2

Set to "1" to enable the multi-machine communication bit.

When the multi-computer communication bit is turned on, when sending data, the host can use LPUART\_SCON.TB8 to distinguish the current

Whether the 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 receive

interrupt.

When it is an address frame, since the automatic address recognition function in the multi-machine communication is enabled, the slave machine can detect the received address and

Whether its own address matches.

If the addresses match, the slave sets LPUARTx\_ISR.RI to '1' and LPUARTx\_SCON.RB to '1'. Slave

After the software sees LPUARTx\_SCON.RB8=1 and LPUARTx\_ISR.RI=1, it will

The LPUARTx\_SCON.SM2 bit is cleared to "0" to accept the data frame.

If the address does not match, it indicates that the master is not addressing the slave, and the slave hardware keeps LPUARTx\_RB8

and LPUARTx\_ISR.RI is "0", the software keeps the LPUARTx\_SCON.SM2 bit as "1", the slave continues

continue to be in the address listening state.

Note:

If necessary, the multi-computer communication bit can also be enabled in Mode1, at this time, the TB8 bit is replaced by the stop bit. When the slave is connected

When a matching address frame and a valid stop bit are received, LPUARTx\_ISR.RC will be set to '1'.



## 16.7 Automatic address recognition

When the multi-computer communication bit is turned on (LPUART\_SCON.SM2 is set to "1"), the automatic address recognition function will also be turned on. Should

The function is implemented in hardware so that the slave can detect that each address frame is received, and if the address matches the slave address,

The receiver will give the LPUART\_ISR.RI receive flag. If the addresses do not match, the receiver will not give any

Receive flag.

### 16.7.1 Given address

The LPUART\_SADDR register of the LPUART device is used to indicate the given address of its own device,

The LPUART\_SADEN register is an address mask that can be used to define don't care bits in the address. when

A bit of LPUART\_SADEN is "0", indicating that the address of this bit is an irrelevant bit, that is to say, during the address matching process

, this bit address does not participate in address matching. These don't care bits add addressing flexibility, allowing the host to simultaneously

Addresses one or more slave devices. Note that if a unique matching address needs to be given, LPUART\_SADEN is sent

register must be set to 8'hFF.

**GivenAddr = SADDR & SADEN**

### 16.7.2 Broadcast address

The broadcast address is used to address all slave devices at the same time, and the general broadcast address is 8'hFF.

**BroadCastAddr = SADDR|SADEN**

### 16.7.3 Examples

Suppose the LPUART\_SADDR and LPUART\_SADEN of a slave are configured as follows:

SADDR: 8'b01101001

SADEN: 8'b11111011

Then its given address and broadcast address are as follows:

Given: 8'b011010x01

Broadcast: 8'b111111x11

It can be seen that the master can use four addresses to address the slave, namely:

8'b01101001 and 8'b01101101 (given address) 8'b11111011 and

8'b11111111 (broadcast address).

## 16.8 Transceiver Cache

### 16.8.1 Receive buffer

The LPUART receiving end has a receiving buffer with a frame length (8/9bits), that is to say, when a frame of data is received

After receiving the buffer, 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.

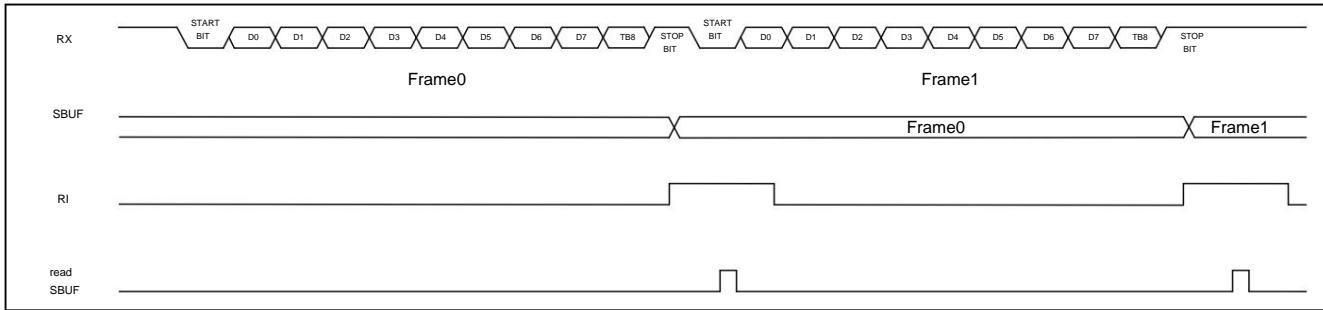


Figure 16-10 Receive buffer

### 16.8.2 Send Buffer

The LPUART transmitter has a transmission buffer with frame length (8/9bits).

When frame data, the CPU can write the data to be sent in the next frame to the send buffer.

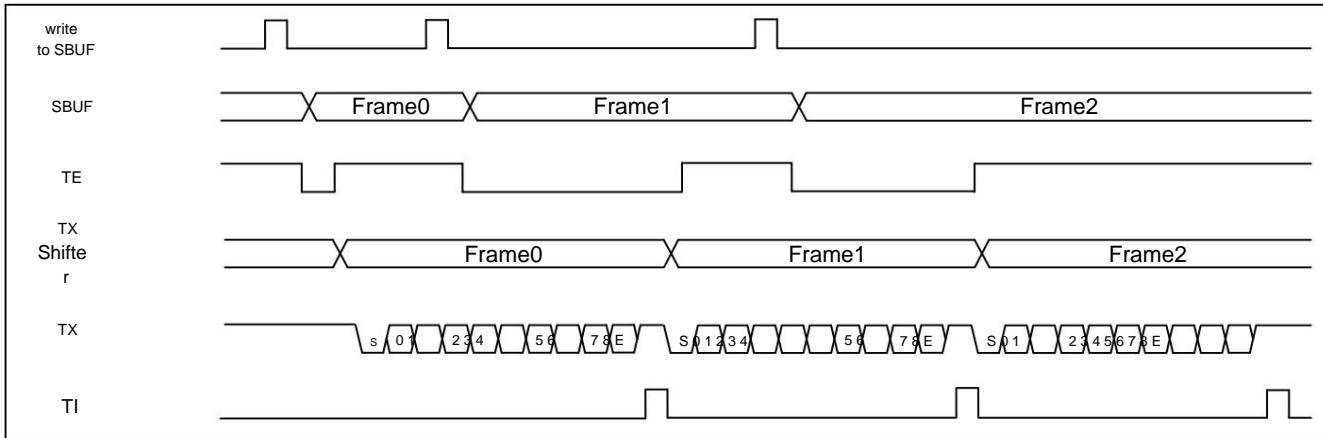


Figure 16-11 Send buffer

Among them, the register bit LPUART\_ISR.TE is the send buffer empty flag bit. The LPUART module contains only one frame

(8/9bits) send buffer, so when the LPUART\_ISR.TE bit is "1", the hardware will automatically mask the software pair

Writes to the LPUART\_SBUF register until the LPUART\_ISR.TE bit changes to '0'. software will be released

Before sending data to fill in the LPUART\_SBUF register, the status of the LPUART\_ISR.TE bits "0" and "1" must be judged.

Otherwise the sent data will be lost.

## 16.9 Registers

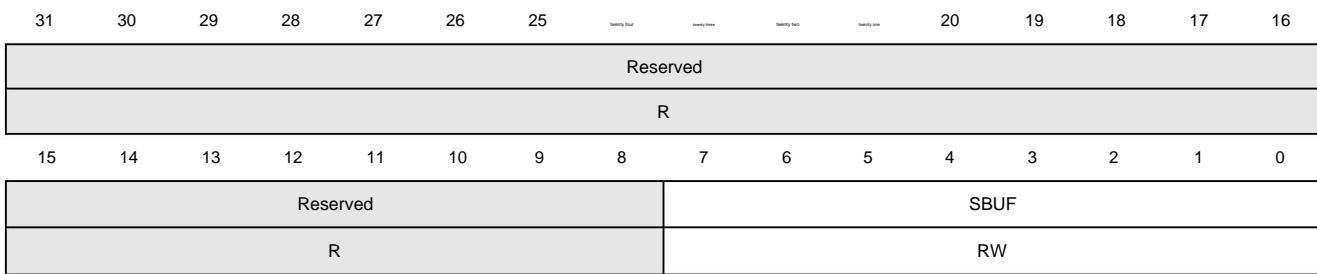
LPUART base address: 0x4000 0200

register name	offset address	describe
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 Clear Register

### 16.9.1 Data Register (LPUART\_SBUF)

Offset address: 0x00

Reset value: 0x0000 0000



bit mark		Function description
31:8	Reserved	
7:0	SBUF	When sending data, it is written into this register; when receiving data, it is read from this register after the data is received.  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									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	LPM ODE	DBAs UD	TEEN	SM01	SM2 REN	TB8 RB8	TIEN RIEN										
	RW	RW				RW	RW RW	RW RW	RW RW										

bit mark		Function description
31:16	Reserved	
15:13	PRS transmission clock SCLK prescaler selection;  000: div128; 001: div64; 010: div32; . . . ; 110: div2; 111: div1  PRS[2:0] is only valid 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 working 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 working 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 Others: 0: Send, 1: Receive/Send
3	TB8 Send TB8 bit
2	RB8 Receive RB8 bit
1	TIEN send completion interrupt enable; 0: disable; 1: enable
0	RIEN receive completion 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								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 mark		Function 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								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 sign		Function 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										20	19	18	17	16
Reserved																				
R																				
Reserved										TE		FE		TI		RI				
R										R		R		R		R				

bit mark		Function description
31:4	Reserved	
3	TE transmit buffer empty interrupt flag, set by hardware and cleared by hardware.  Note: When the value of this bit is "0", the hardware automatically shields the software to write the SBUF operation.  1: TE interrupt is valid 0: TE interrupt is invalid	
2	FE Receive frame error flag, set by hardware and cleared by software  1: FE interrupt is valid 0: FE interrupt is invalid	
1	TI transmission completion interrupt flag bit, set by hardware, cleared by software  1: TI interrupt is valid 0: TI interrupt is invalid	
0	RI Receive complete 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									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														WWW					

bit sign		describe
31:3	Reserved	
2	FECLR clears the received frame error flag;  Write 0 to clear, write 1 to be invalid	
1	TICLR clears the transmit completion interrupt flag bit;  Write 0 to clear, write 1 to be invalid	
0	RICLR Clear receive completion interrupt flag bit; 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 connect the two devices of the bus.

It provides a simple and efficient method for data exchange between devices. each connected to

Devices on the bus have unique addresses, and any device can act as either a master or a slave, but at the same time

Only one host is allowed. The I2C standard is a true multi-master with conflict detection mechanism and arbitration mechanism

It uses an arbitration mechanism to avoid data conflicts and protect data when multiple hosts request control of the bus at the same time.

The I2C bus controller can meet various specifications of the I2C bus and support all transmission modes that communicate with the I2C bus.

The I2C logic handles the transfer of bytes autonomously. It keeps track of serial transfers and has a status register

(I2C\_STAT) can reflect the status of I2C bus controller and I2C bus.

### 17.2 Main Features

The I2C controller supports the following features:

- ÿ Support host send/receive and slave send/receive four working modes
- ÿ Support standard (100Kbps) / fast (400Kbps) / high speed (1Mbps) three working rates
- ÿ Support 7-bit addressing function
- ÿ Support noise filtering function
- ÿ Support broadcast address
- ÿ Support interrupt status query function

### 17.3 Protocol Description

The I2C bus uses the "SCL" (serial clock bus) and "SDA" (serial data bus) of the connected devices to transfer information. host

The computer outputs the serial clock signal on the SCL line, the data is transmitted on the SDA line, and each byte is transmitted (the highest bit MSB to start transmission), followed by an acknowledge bit. One SCL clock pulse transfers one data bit.

### 17.3.1 Data transfer 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

Bit, transmit data, stop signal (P).

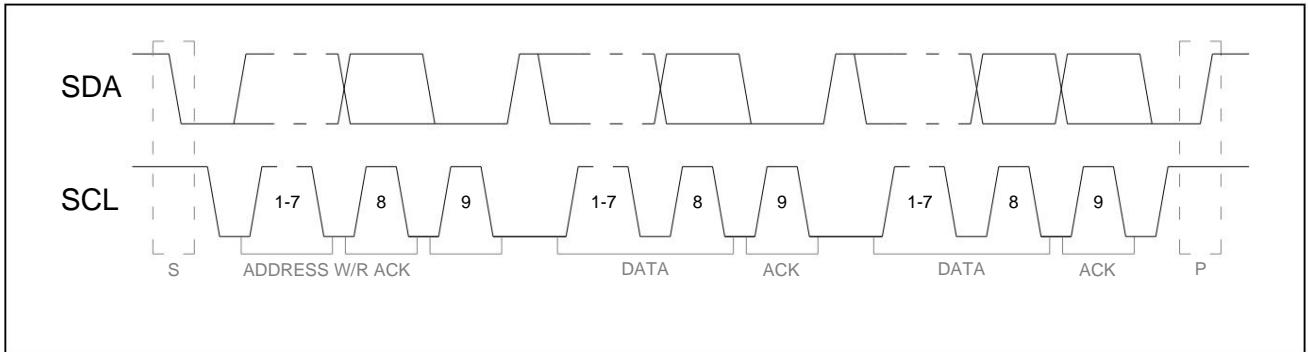


Figure 17-1 I2C transfer protocol

ÿ 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 high-to-low signal appears on the SDA line.

, indicating that a start signal is generated on the bus.

A repeated start signal is generated when there is no stop signal between two start signals. The host adopts this method

communicate with another slave or the same slave in a different transfer direction (eg: from write device to slave device

read) 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. host to total

The line sends a stop signal to end the data transfer.

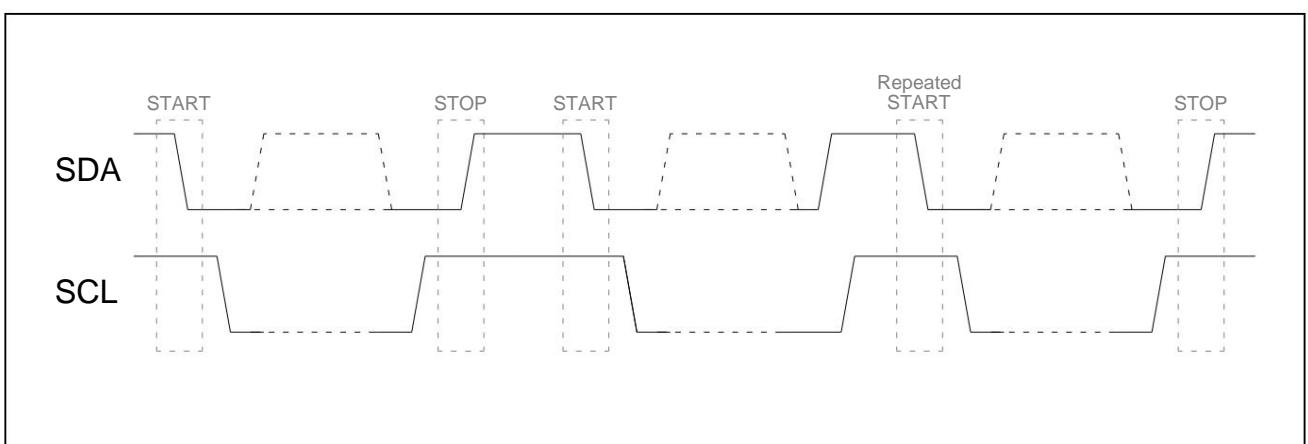


Figure 17-2 START and STOP conditions

ÿ Slave address and read and 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, read and write

The bit controls the data transfer direction of the slave (0: write; 1: read). Slave opportunities addressed by the master pass in the ninth

The SCL clock cycle acknowledges by asserting SDA low.

ÿ Transfer data

During data transfer, one SCL clock pulse transfers one data bit, and the SDA line is only low when SCL is low.

can only be changed.

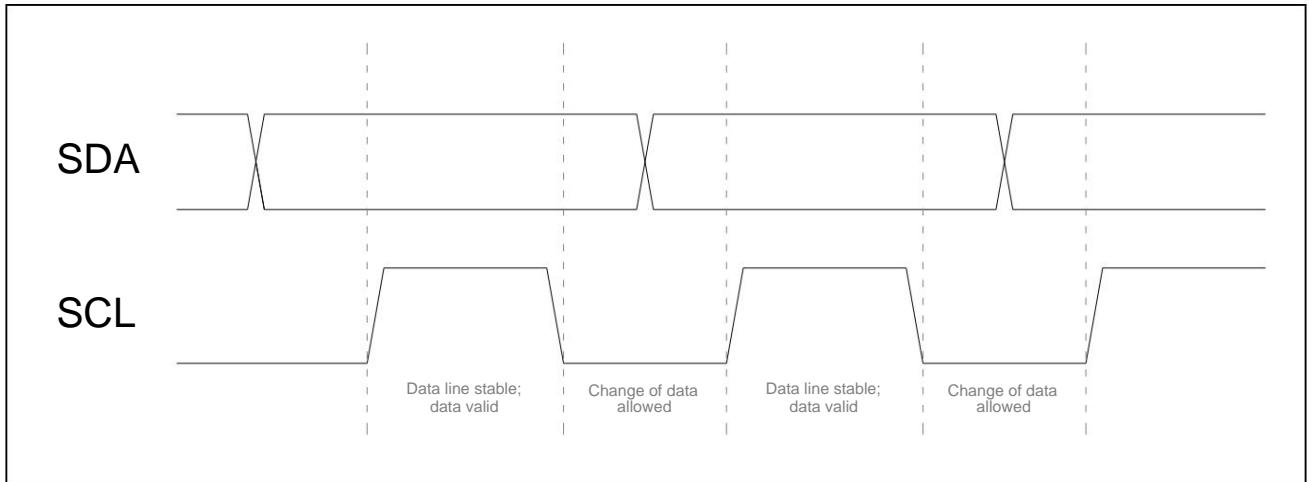


Figure 17-3 I2C bus upper transmission

### 17.3.2 ACKNOWLEDGE ON I2C BUS

Each byte transferred is followed by an acknowledge bit. By pulling the SDA line low, the receiver is allowed to respond to the transmission end. 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 sender.

When the master is used as a sending device, if the slave generates a no response signal (NACK), 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 host acts as a receiver

When a non-response signal (NACK) occurs, the slave releases the SDA line, causing the master to generate a stop signal or repeat start signal.

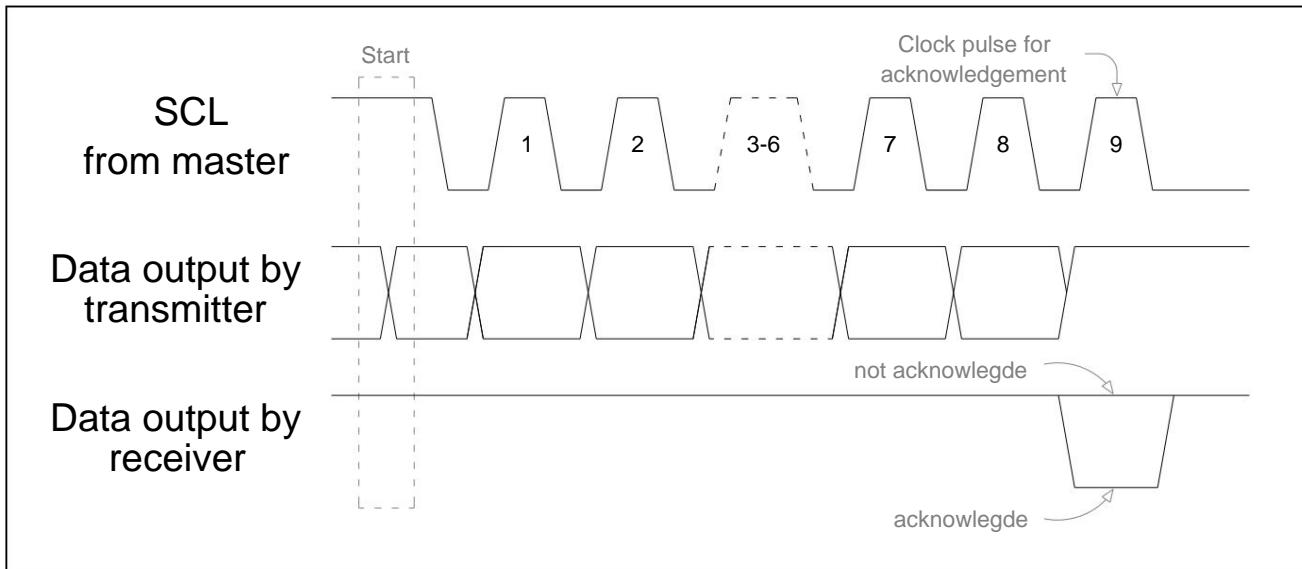


Figure 17-4 Acknowledgment signal on I2C bus

### 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

#### ÿ 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 total

low level on the line. The bus can only appear high when all nodes send high. so,

The clock low time is determined by the device with the longest clock period, while the clock high time is determined by the clock high.

The device with the shortest flat period is determined. Due to this characteristic of I2C, when multiple masters send clock signals at the same time, the total

The line represents the unified clock signal. If the slave wants the master to reduce the transmission speed, it can pull the SCL

Pull low to extend its low level time to notify the host, when the host finds the level of SCL when preparing for the next transmission

When pulled low waits until the slave completes the operation and releases control of the SCL line.

#### ÿ Arbitration on SDA line

Arbitration on the SDA line is also due to the wired AND logic function of the I2C bus. After the host sends data,

Decide whether to exit the competition by comparing the data on the bus. A host that loses quorum immediately switches to unaddressed

slave state to ensure that it can be addressed by the master that wins the arbitration. When the host that lost arbitration continues to output

clock pulses (on SCL) until the current serial byte has been sent. This principle can ensure that the I2C total

The line ensures that data is not lost when multiple masters attempt to control the bus.

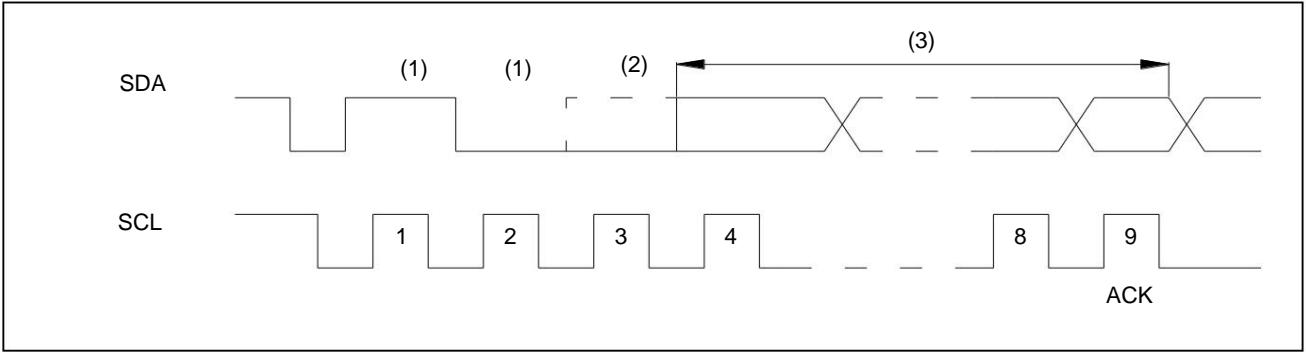


Figure 17-5 Arbitration on the I2C bus

- a) another device transmits serial data;
- b) The other device first de-asserted a logic 1 (dotted line) sent by the I2C master by pulling SDA low. arbitration Lost, I2C enters slave receive mode;
- c) At this time, I2C is in slave receive mode, but still generates clock pulses until the current byte is sent. I2C will No clock pulse is generated for the transfer of the next byte. Once arbitration is won, data transfers on SDA are performed by the new host to start.

#### 17.4 Functional Description

The I2C bus uses two wires between devices connected to the bus "SCL" (serial clock line) and "SDA" (serial data line)

Send information. Filtering logic protects data integrity by filtering glitches on the data bus. Since there is only no direction

port, the I2C component requires the use of an open-drain buffer to the pin. Every device connected to the bus can use the software

files are addressed by a specific address. The I2C standard is a true multi-system with a collision detection mechanism and an arbitration mechanism.

host bus. It prevents data collisions when two or more hosts start transmitting data at the same time. I2C bus

Status can be queried in the status register.

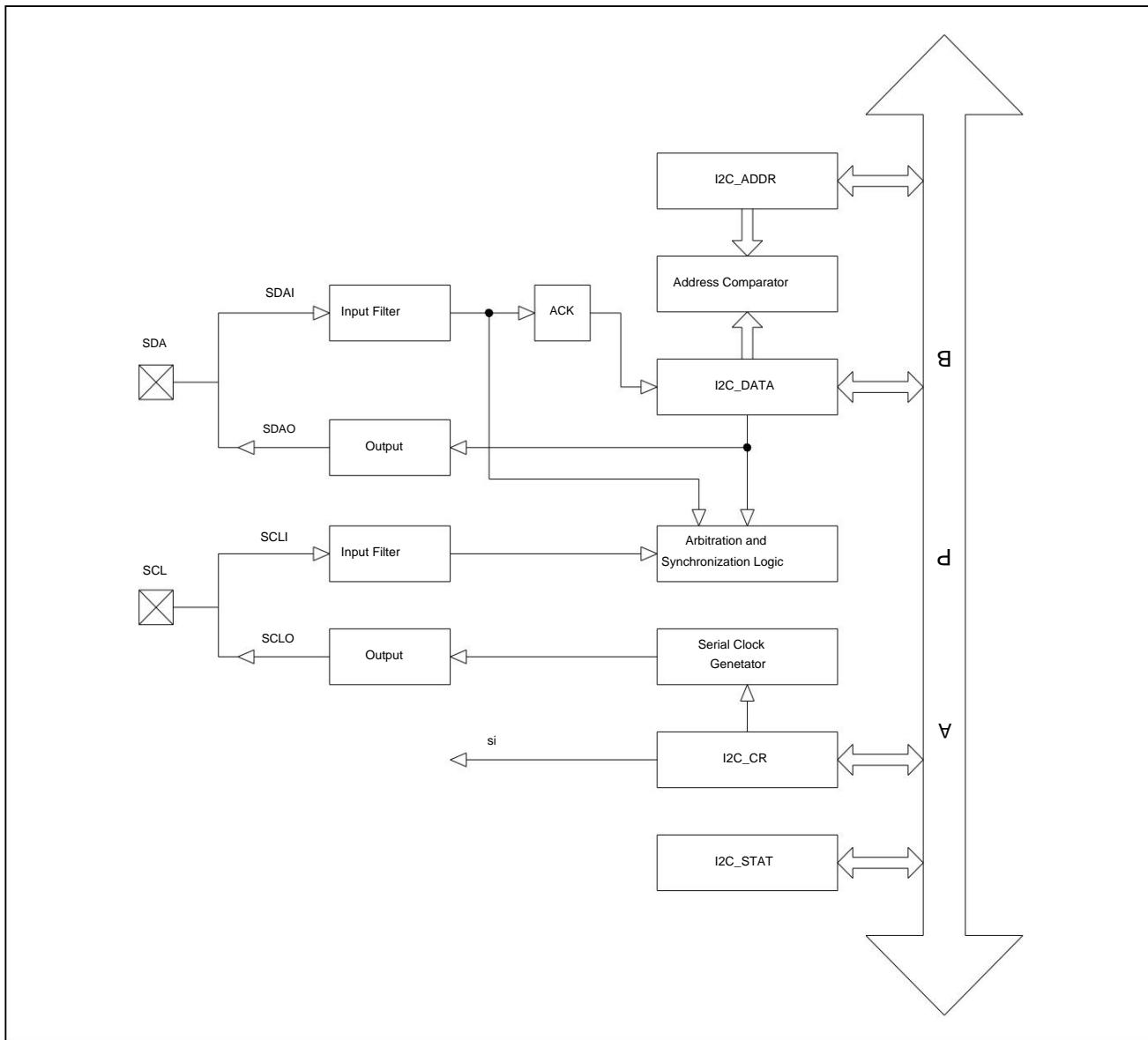


Figure 17-6 I2C function block diagram

### 17.4.1 Serial Clock Generator

The serial clock generator uses an 8-bit counter as a baud rate generator, SCL signal and PCLK signal

The frequency relationship is  $F_{SCL} = F_{PCLK} / 8 / (I2C\_TM.tm + 1)$ , where  $I2C\_TM.tm$  should be greater than 0.

The following table lists the output frequency value of the SCL signal when the PCLK frequency is combined with the  $I2C\_TM.tm$ .

PCLK (KHz)	I2C_TM.tm						
	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

Table 17-1 I2C clock signal baud rate

### 17.4.2 Input Filter

The input signal is synchronized to PCLK, and spikes below the PCLK period are filtered out.

When this module is used as the host, if the value of  $I2C\_TM$  is less than or equal to 9,  $I2C\_CR.H1M$  should be set to 1; for example

If the value of  $I2C\_TM$  is greater than 9, set  $I2C\_CR.H1M$  to 0.

When this module is used as a slave, if the frequency ratio of PCLK and SCL is less than or equal to 30,  $I2C\_CR.H1M$  should be set

It is 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

The I2C comparator compares its own slave address with the received 7-bit slave address. It can use "I2C\_ADDR"

register to program its own slave address. And according to the "i2cadr" bit of "I2C\_ADDR" register and the first

The last received 8-bit byte or the broadcast address (0x00) is compared. If either is the same, "I2C\_CR" is registered



The "si" bit of the device will be set to 1 and an interrupt request will be generated.

#### 17.4.4 ACKNOWLEDGE FLAG

The "aa" flag of the "I2C\_CR" register is the acknowledgement flag. When the "aa" bit is 1, the I2C module will return the data after receiving the data.

Acknowledge bit, when the "aa" bit is 0, the I2C module will return a non-acknowledge bit after receiving the data.

#### 17.4.5 Interrupt Generator

The "si" flag of the "I2C\_CR" register is the interrupt flag. Whenever the value of the status register (I2C\_STAT) is generated

When changing (except when it becomes 0xF8), the "si" flag will be set to 1. When an interrupt is generated, by polling the status register

The state of the I2C bus can be known to the device (I2C\_STAT) to determine the actual source of the interrupt. for the next step

operation, the "si" flag must be cleared by software.

#### 17.4.6 Working Mode

The I2C component can realize 8-bit bidirectional data transmission, and the transmission rate can reach 100Kbps in standard mode and 100Kbps in high

It can reach 400Kbps in high speed mode and 1Mbps in super speed mode, and can work in four modes:

Master sending mode, master receiving mode, slave receiving mode, slave sending mode. There is also a special mode

In broadcast call mode, its operation is similar to slave receive mode.

þ Host send mode

The master sends multiple bytes to the slave, and the master generates the clock, so the set value needs to be filled in I2C\_TM. host

I2C\_CR.sta needs to be set to 1 in machine transmit mode. When the bus is idle, 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,

After clearing the "si" bit, SLA+W is issued on the bus.

"si" is set to 1 after the master sends SLA+W and receives the ACK bit from the slave. Next according to the user-defined format

send data. After all data is sent, set I2C\_CR.sto to 1, clear the "si" bit and issue a STOP signal,

A repeat start signal can also be sent for a new round of data transmission.

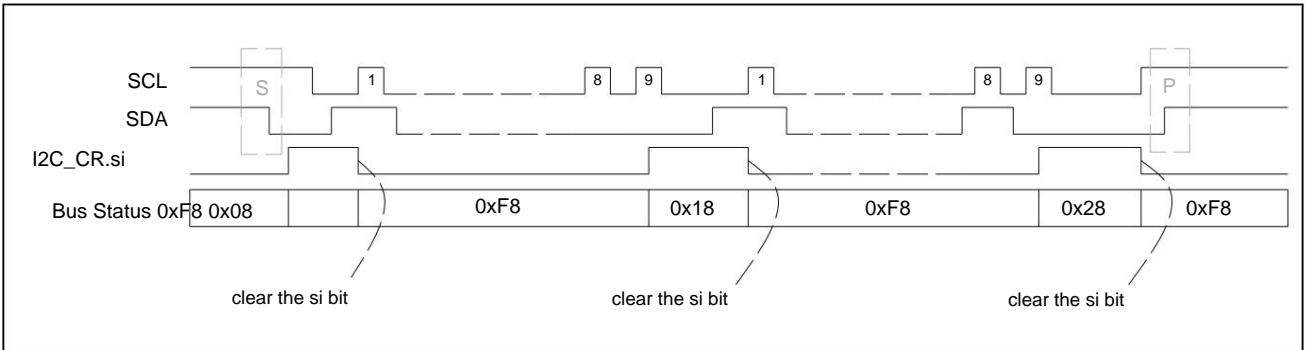


Figure 17-7 Data synchronization diagram in master sending mode

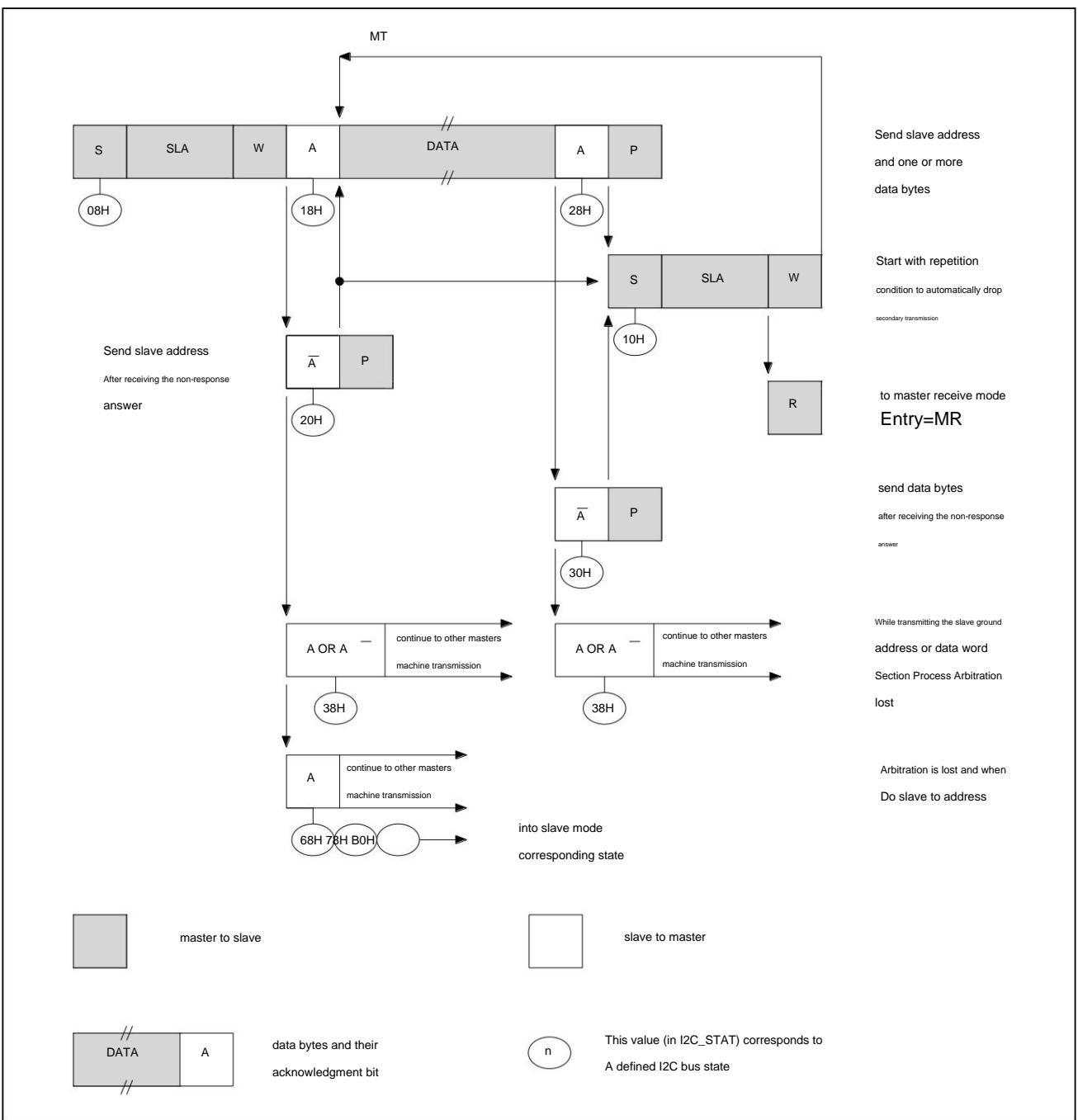


Figure 17-8 I2C master sending state diagram

ÿ Host receive mode

In master receive mode, data is transmitted by the slave. The initial setting is the same as the host sending mode, the host sends the start

bit later, I2C\_DATA should be written with the slave address and the "read bit" (SLA+R). Receive the slave acknowledgment bit ACK

Then I2C\_CR.si is set to 1. After "si" is cleared to 0, it starts to receive slave data. If I2C\_CR.aa is 1, the master receives

The response bit is returned after the data; if it is 0, the host does not respond to NACK after receiving the data. The host can then send a stop letter

number or repeat start signal to start the next round of data transmission.

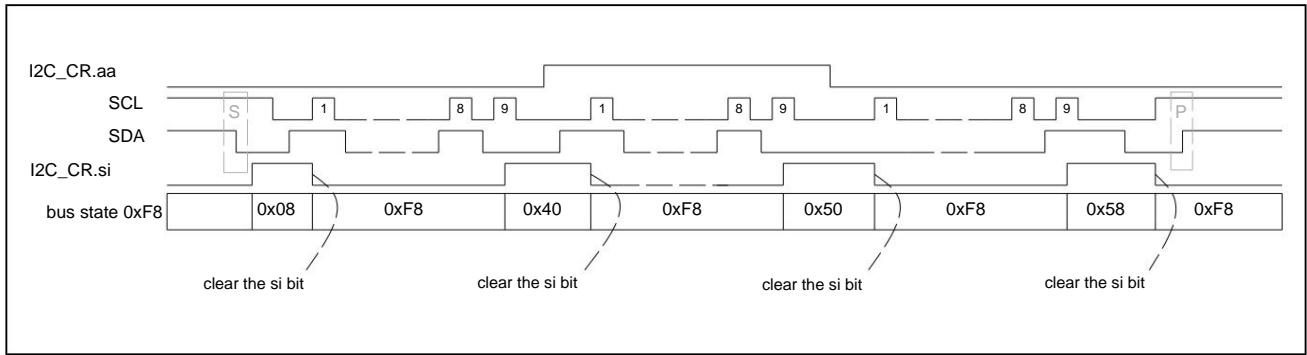


Figure 17-9 Data synchronization diagram in master receive mode

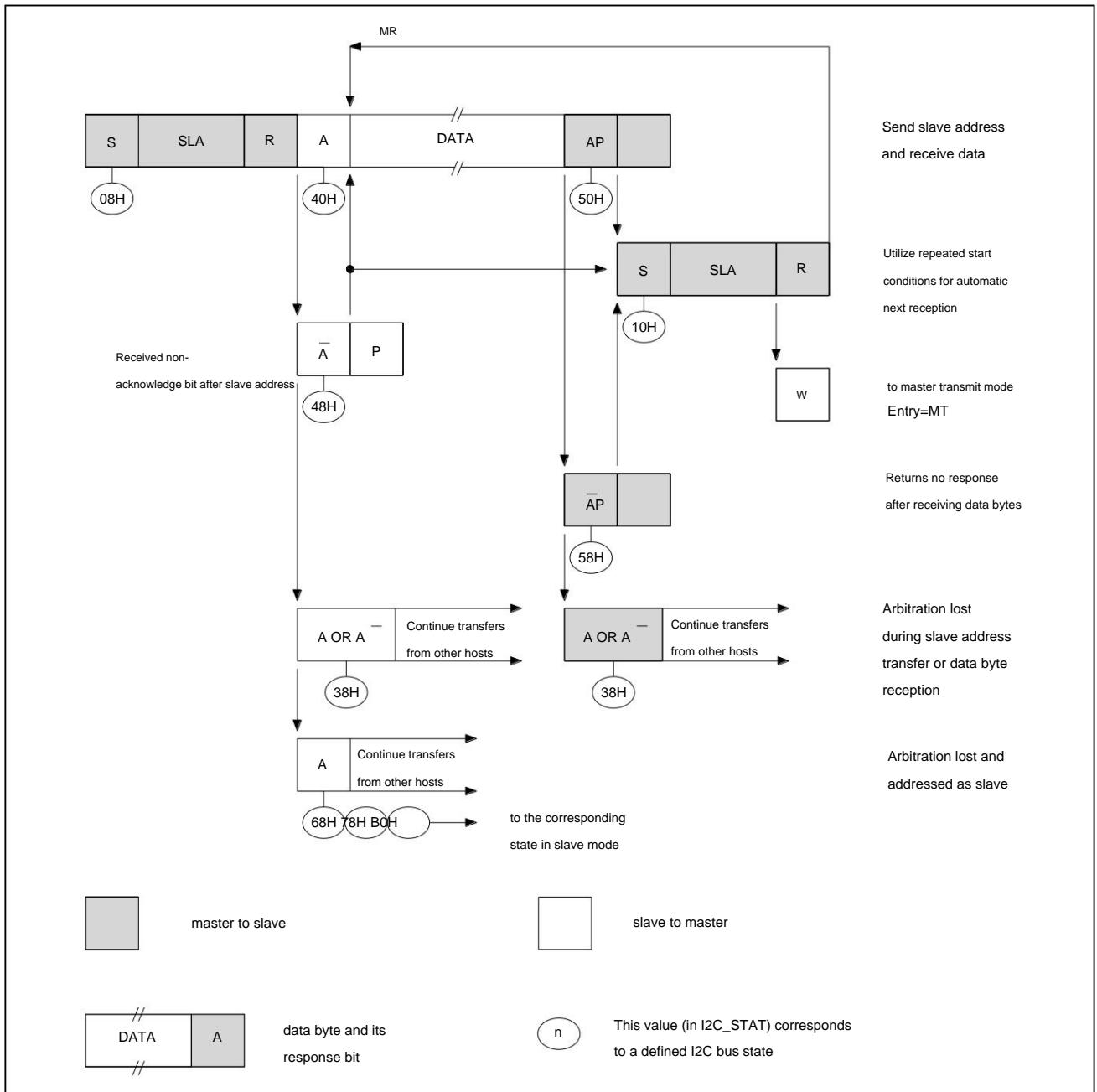


Figure 17-10 I2C host receiving state diagram

### Slave Receive Mode

In slave receiving mode, the slave receives data from the master. Before the transfer starts, I2C\_ADDR should be written to

Slave address, I2C\_CR.aa is set to 1 in response to master addressing. After the above initialization, the slave enters idle mode

mode, waiting for the "write" signal (SLA+W). If the master arbitration fails, it will also directly enter the slave receiver mode.

When the slave is addressed by the "write" signal SLA+W, the "si" bit needs to be cleared in order to receive data from the master. Such as

If I2C\_CR.aa=0 during the transmission, the slave will return the non-acknowledgment bit NACK in the next byte, and the slave will also

Turns to an unaddressed slave, terminates the contact with the master, no longer receives data, and I2C\_DATA keeps the previously received

The data. Slave address recognition can be restored by setting "aa", which means that the "aa" bit temporarily switches the I2C module from

isolated on the I2C bus.

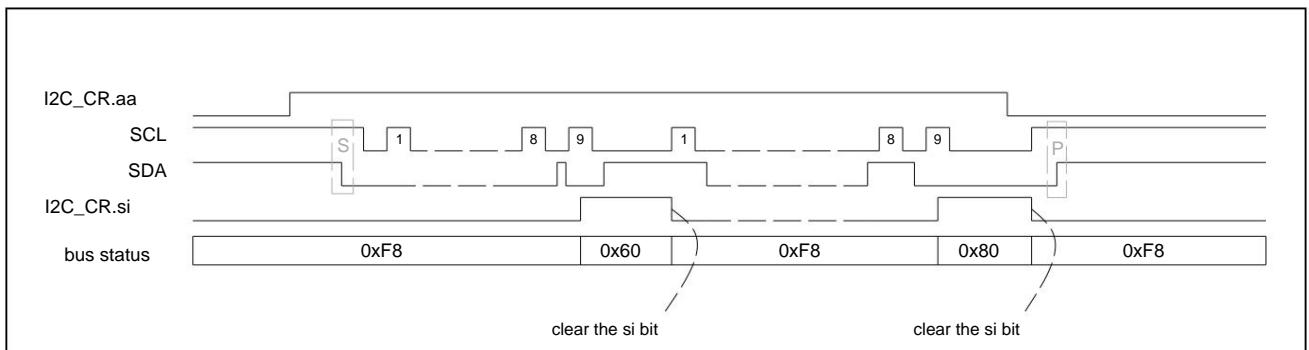


Figure 17-11 Data synchronization diagram in slave receive mode

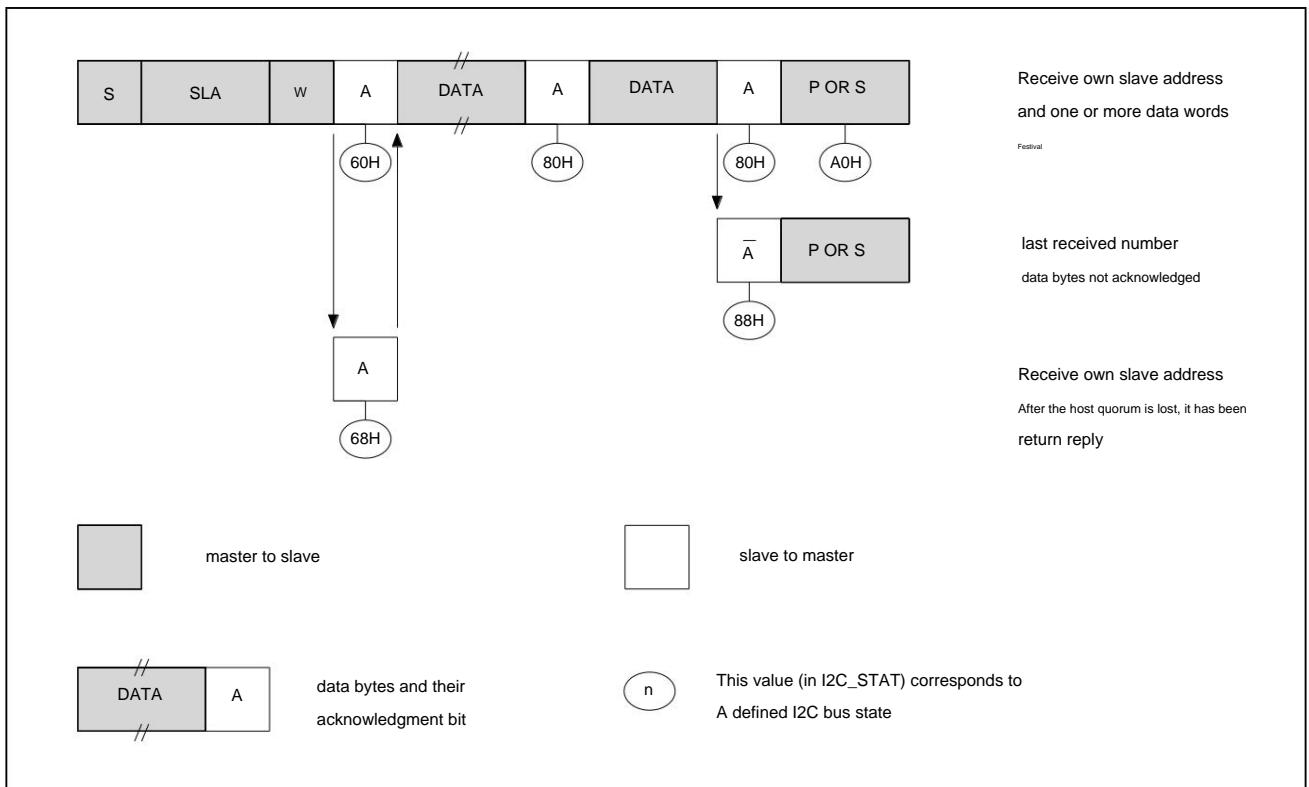


Figure 17-12 Slave receiving state diagram

Slave transmit mode

In slave transmission mode, data is sent from the slave to the master. After initializing the I2C\_ADDR and I2C\_CR.aa values,

The device waits until its own address is addressed by a "read" signal (SLA+R). If the master arbitration fails, it can also enter the slave machine send mode.

When the slave is addressed by the "read" signal SLA+R, "si" needs to be cleared to send data to the master. usually host

Acknowledge bits are returned after each byte of data is received.

If I2C\_CR.aa is cleared during the transfer, the slave will send the last byte of data, and it will be sent in the next transfer.

Sends all 1 data in the input and turns itself into an unaddressed slave.

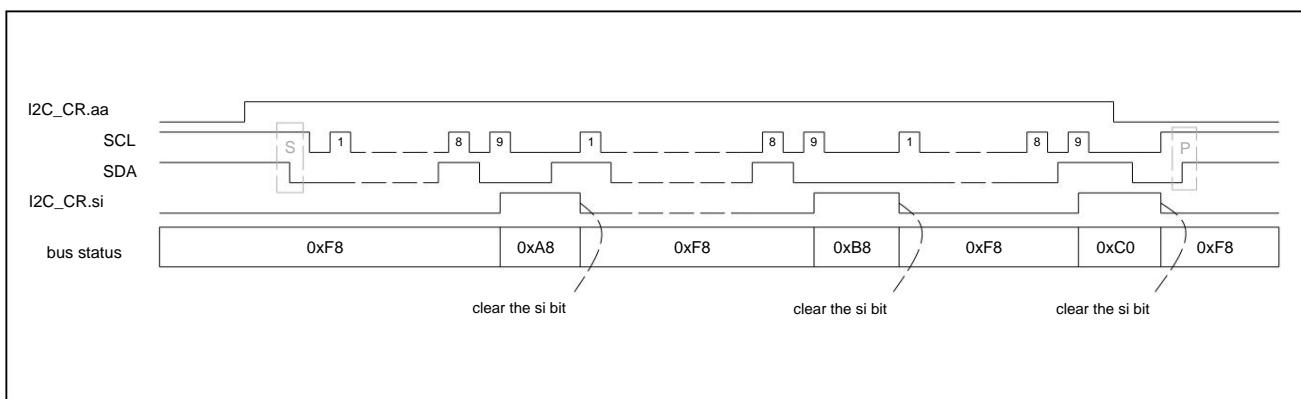


Figure 17-13 Data synchronization diagram in slave send mode

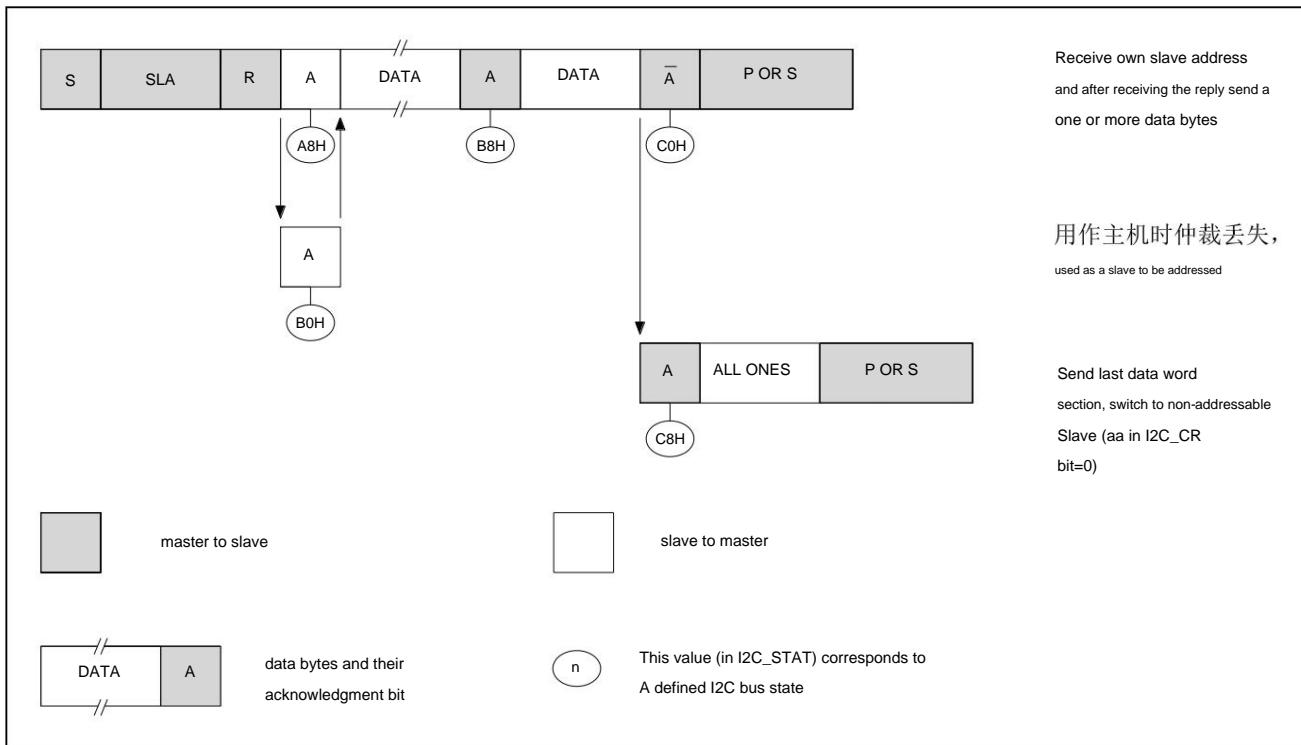


Figure 17-14 I2C slave sending state diagram

ÿ General call mode

The general call mode is a special slave receiving mode, the addressing mode is 0x00, the slave address and read and write are both

is 0. When both I2C\_ADDR.GC and I2C\_CR.aa are set to 1, the receive general call mode is enabled. in this mode

The lower I2C\_STAT value is different from the normal slave receiver mode I2C\_STAT value. Arbitration loss may also go into broadcast call mode.

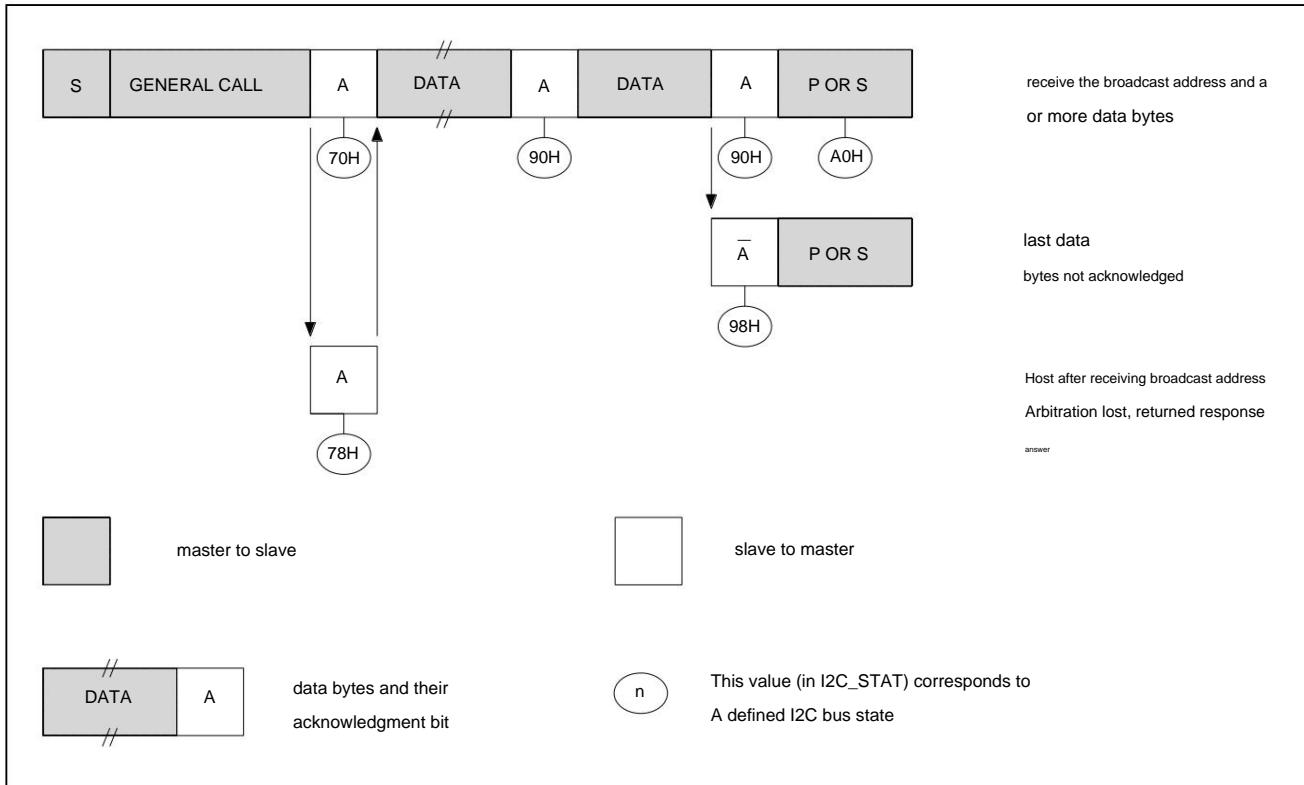


Figure 17-15 I2C general call state diagram



#### 17.4.7 Status code representation

There are two special states in the I2C status register: F8H and 00H.

F8H: This status code indicates that there is no relevant information available because the serial interrupt flag "si" has not been set.

This condition occurs between other states and when the I2C module has not started performing a serial transfer.

00H: This status code indicates that a bus error occurred during I2C serial transfer. When the illegal position of the format frame

A bus error occurs when a START or STOP condition occurs on the These illegal locations refer to the ground during serial transmission

address byte, data byte, or acknowledge bit. Bus errors also occur when external disturbances affect the internal I2C module signals.

error. "si" is set when a bus error occurs.

status code	describe
master transmit mode	
08H	Start condition sent
10H	Repeated START sent
18H	SLA+W sent, ACK received
20H	SLA+W sent, non-ACK received
28H	Data in I2C_DATA has been sent, ACK has been received
30H	Data in I2C_DATA has been sent, non-ACK received
38H	Lost Arbitration on SLA+ Read or Write Data Bytes
Master receive mode	
08H	Start condition sent
10H	Repeated START sent
38H	Arbitration lost in non-ACK
40H	SLA+R sent, ACK received
48H	SLA+R sent, non-ACK received
50H	Data bytes received, ACK returned
58H	Data bytes received, not ACK returned
slave receive mode	
60H	Received its own SLA+W and returned ACK

68H	In the master control, the arbitration is lost in SLA+ read and write, has received its own SLA+W, and has returned ACK
80H	The previous addressing used its own slave address, the data byte has been received, and ACK has been returned
88H	The previous addressing used its own slave address, a 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
slave transmit mode	
A8H	Received its own SLA+R and returned ACK
B0H	Arbitration is lost when the master, has received its own SLA+R, has returned ACK
B8H	Data sent, ACK received
C0H	Data bytes sent, non-ACK received
C8H	Loaded data bytes have been sent, ACK has been received
general call mode	
70H	Broadcast address received (0x00), ACK returned
78H	In the master control, the arbitration is lost in SLA+ read and write, the broadcast address has been received, and ACK has been returned
90H	The previous addressing used the broadcast address, the data byte has been received, and the ACK has been returned
98H	The previous addressing used a broadcast address, a data byte was received, and a non-ACK was returned
A0H	When statically addressed, a STOP condition or a repeated START condition is received
Rest of Miscellaneous Status	
F8H	No relevant status information available, si=0
00H	A bus error occurs during transfer, or external disturbance puts I2C into an undefined state

Table 17-2 I2C Status Code Description



## 17.5 Programming Examples

### 17.5.1 Host Send Example

Step1: Map SCL and SDA to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter;

And configure the SCL and SDA pins as open-drain output mode.

Step2: Set PERI\_CLKEN.I2C to 1 to enable the I2C module clock.

Step3: Write 0 and 1 to PERI\_RESET.I2C in turn to reset the I2C module.

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, continue to the next step, otherwise go to error handling.

Step10: Write SLA+W to I2C\_DATA, set I2C\_CR.sta to 0, set I2C\_CR.si to 0,

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, continue to the next step. Otherwise make an error deal with.

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, continue to the next step. Otherwise make an error deal with.

Step16: If the data to be sent is not completed, 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.

Step18: Wait for I2C\_CR.si to become 1, and the Stop signal has been sent to the bus.

## 17.5.2 Host Reception Example

Step1: Map SCL and SDA to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter;

And configure the SCL and SDA pins as open-drain output mode.

Step2: Set PERI\_CLKEN.I2C to 1 to enable the I2C module clock.

Step3: Write 0 and 1 to PERI\_RESET.I2C in turn to reset the I2C module.

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, continue to the next step, otherwise

Error handling.

Step10: Write SLA+R to I2C\_DATA, set I2C\_CR.sta to 0, set I2C\_CR.si to 0, send

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, continue to the next step, otherwise go to the error

reason.

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, and read the received data from I2C\_DATA.

Step16: Query I2C\_STAT, if the register value is 0x50 or 0x58, continue to the next step, no

Error handling is performed.

Step17: If the data to be received is only missing 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, 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.

Step20: Wait for I2C\_CR.si to become 1, and the Stop signal has been sent to the bus.



### 17.5.3 Slave Receive Example

Step1: Map SCL and SDA to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter;

And configure the SCL and SDA pins as open-drain output mode.

Step2: Set PERI\_CLKEN.I2C to 1 to enable the I2C module clock.

Step3: Write 0 and 1 to PERI\_RESET.I2C in turn to reset the I2C module.

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 register value is 0x60, continue to the next step, otherwise an error occurs

deal with.

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, and read the received data from I2C\_DATA.

Step11: Query I2C\_STAT, if the register value is 0x80, continue to the next step, otherwise an error occurs

deal with

Step12: If the data to be received is not completed, 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: Map SCL and SDA to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter;

And configure the SCL and SDA pins as open-drain output mode.

Step2: Set PERI\_CLKEN.I2C to 1 to enable the I2C module clock.

Step3: Write 0 and 1 to PERI\_RESET.I2C in turn to reset the I2C module.

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, continue to the next step, otherwise, perform the output

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, continue to the next step,

Otherwise, error handling is performed.

Step12: If the data to be sent is not completed, 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

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 status register.

Table 17-3 Register List

### 17.6.1 I2C Baud Rate Counter Enable Register (I2C\_TMRUN)

Address offset: 0x00

Reset value: 0x0000 0000

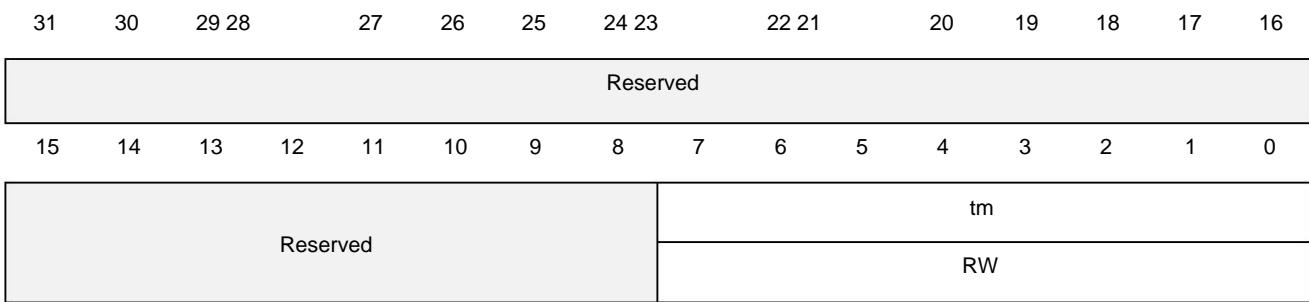
31	30	29	28	27	26	25	24 23			20	19	18	17	16
Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1   0
Reserved														tme RW

Bit tag	function 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



Bit tag function 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

Bit tag	function	description
31:7	Reserved	
6	ens	I2C module enable control  0 - forbidden  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  Read 1, I2C interrupt has occurred  Write 0, I2C performs the next step
2	aa	Acknowledge Control Bit  0 – send NAK  1 – Send ACK
1	Reserved	
0	h1m	I2C filter parameter configuration

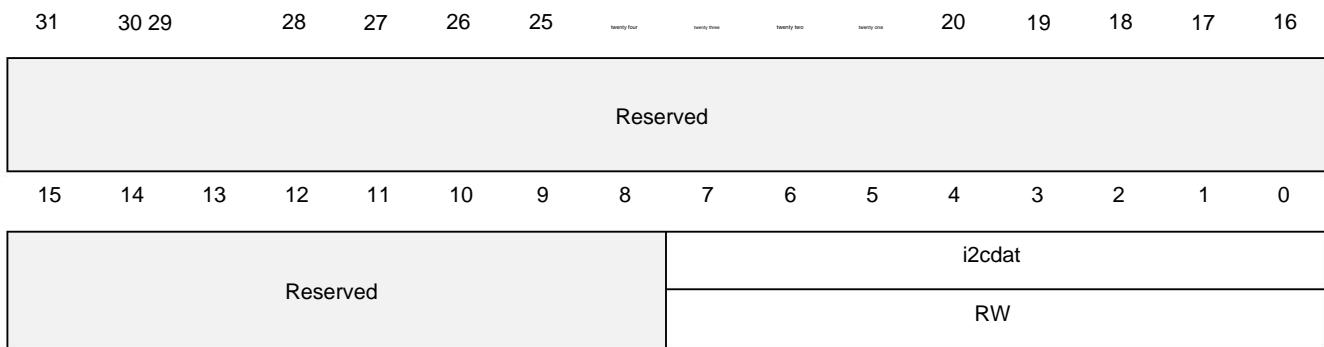


		<p>0 – Advanced filtering, higher noise immunity</p> <p>1 – Simple filtering, faster communication rate</p> <p>Note: See the [Input Filter] chapter for details.</p>
--	--	--

## 17.6.4 I2C Data Register (I2C\_DATA)

Address offset: 0x0c

Reset value: 0x00000000



Bit tag function	description	
31:8	Reserved	
7:0	i2cdat	I2C data register In I2C transmit mode, write the data to be transmitted In I2C receive mode, read the received data

### 17.6.5 I2C Address Register (I2C\_ADDR)

Address offset: 0x10

Reset value: 0x00000000

31	30	29	28	27 26	25	twenty four	seventy three	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		

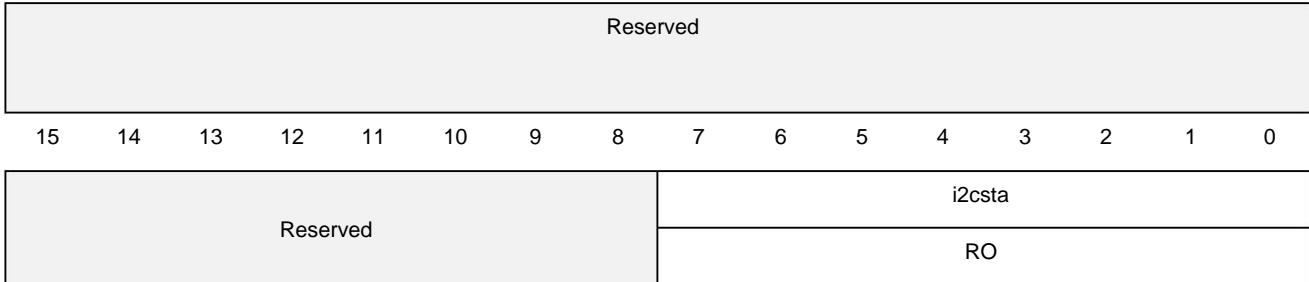
Bit tag function description		
31:8	Reserved	
7:1	i2cadr	I2C slave mode address.
0	GC Broadcast Address Reply Enable	0 - forbidden 1 – enable

## 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
----	----	-------	----	----	----	-------	-------	----	----	----	----	----



Bit tag	function description	
31:8	Reserved	
7:0	i2csta	I2C Status Register  For the specific definition of the status value, please refer to the [Status Code Description] chapter.



## 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 the SPI is used as the master, the CS and SCK signals are output to control the letter process. When the SPI acts as a slave, it communicates under the control of the SSN and SCK signals.

### 18.2 Main Features of SPI

- ÿ Support SPI master mode, SPI slave mode
- ÿ Support standard four-wire full-duplex communication
- ÿ Support to configure serial clock polarity and phase
- ÿ Host mode supports 7 communication rates
  - ÿ The maximum frequency division factor of host mode is PCLK/2, and the maximum communication rate is 16M bps
  - ÿ The maximum frequency division factor of slave mode is PCLK/4, and the maximum communication rate is 12M bps
  - ÿ The frame length is fixed at 8 bits, and the MSB is transmitted first

## 18.3 SPI function description

### 18.3.1 SPI Master Mode

The length of each data frame is fixed to 8 bits, and the first bit of the transmitted data is fixed to the MSB. set SPI\_CR.mstr

If it is 1, the SPI interface works in master mode. The SPI transfer is started by writing data to the SPI\_Data register.

The serial clock is automatically generated by the SCK pin; on the edge of the serial clock, the data in the shift register is sent to the

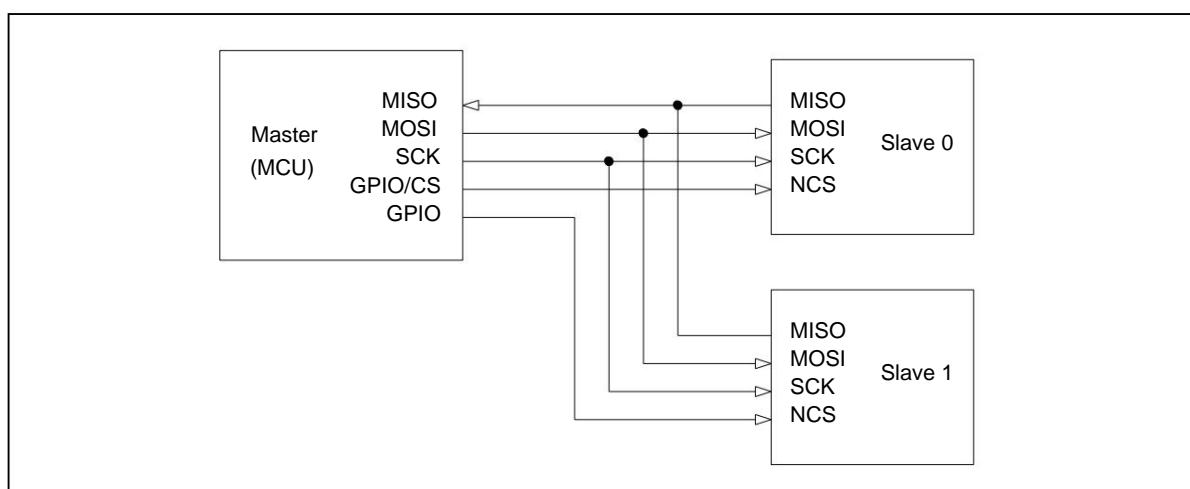
MOSI pin, the data of MISO pin is received into the shift register. Each time the transmission of a data frame is completed,

SPIF will be set by hardware, read SPI\_DATA to clear SPIF flag. The frequency of the SCK pin output clock is determined by

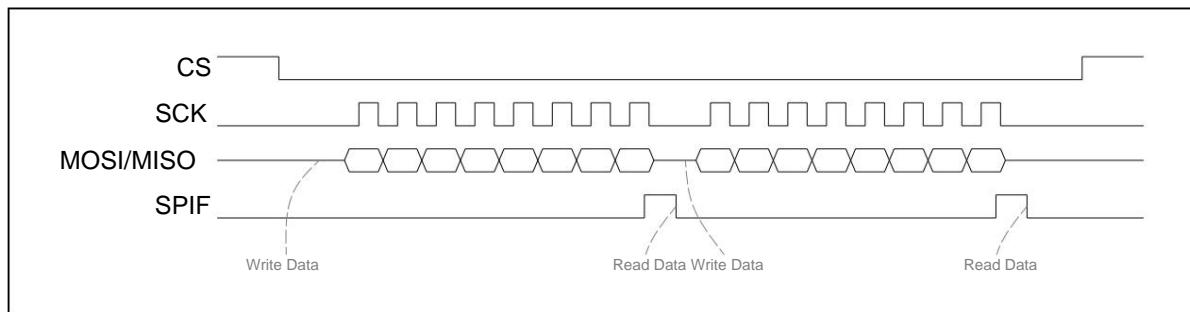
SPI\_CR[spr2:spr0] is controlled, its output frequency range is PCLK/2~PCLK/128;

The output level is controlled by SPI\_SSN.ssn, and the output level of the GPIO pin is controlled by the GPIO related registers.

A block diagram of a typical application of master mode is shown below.



The communication diagram of the host mode is shown in the figure below, where CPOL=0, CPHA=0.



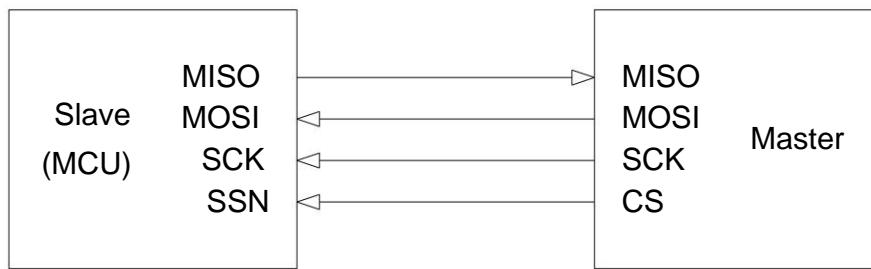
### 18.3.2 SPI Slave Mode

The length of each data frame is fixed to 8 bits, and the first bit of the received data is fixed to the MSB. set SPI\_CR.mstr

If it is 0, the SPI interface works in slave mode. In this mode, the SCK pin is used as an input pin and the serial clock is derived from

For external host; SSN pin is used as input pin and chip select signal comes from external host or fixed to low level. SSN lead

For details, see the auxiliary registers in the GPIO chapter. A typical application block diagram for 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 it as soon as possible

received data. The communication timing of receiving data in slave mode is as follows, where CPOL=0, CPHA=0.

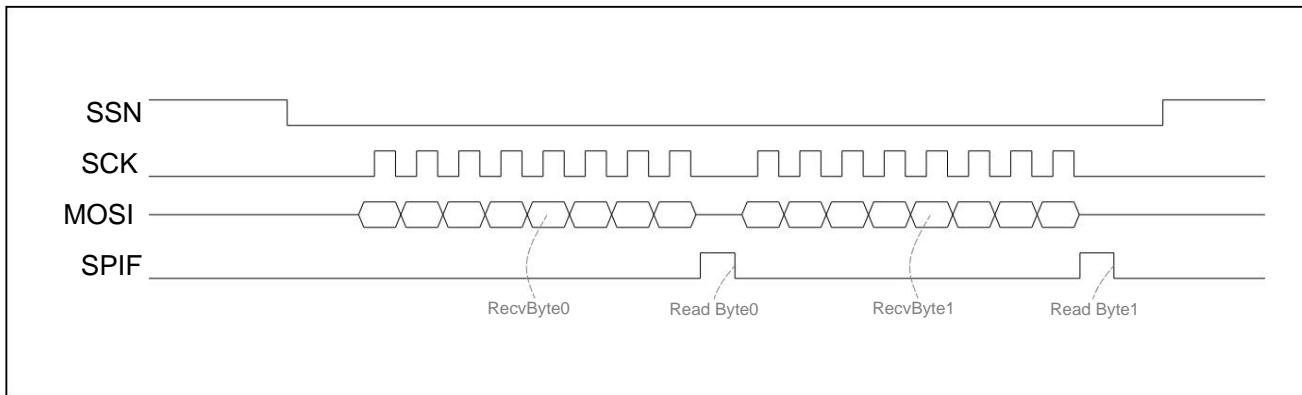


Figure 18-1 Schematic diagram of slave receiving

When the SPI slave needs to send data to the host, it should send the SPI\_DATA register to the SPI\_DATA register as soon as possible after the host pulls down NSS.

Write the first byte of data to be sent in; whenever the SPIF flag is 1, read SPI\_DATA as soon as possible

to clear the SPIF flag and write the subsequent data to be sent to the SPI\_DATA register. Slave mode send count

The communication timing of the data is as follows, where CPOL=0, CPHA=0.

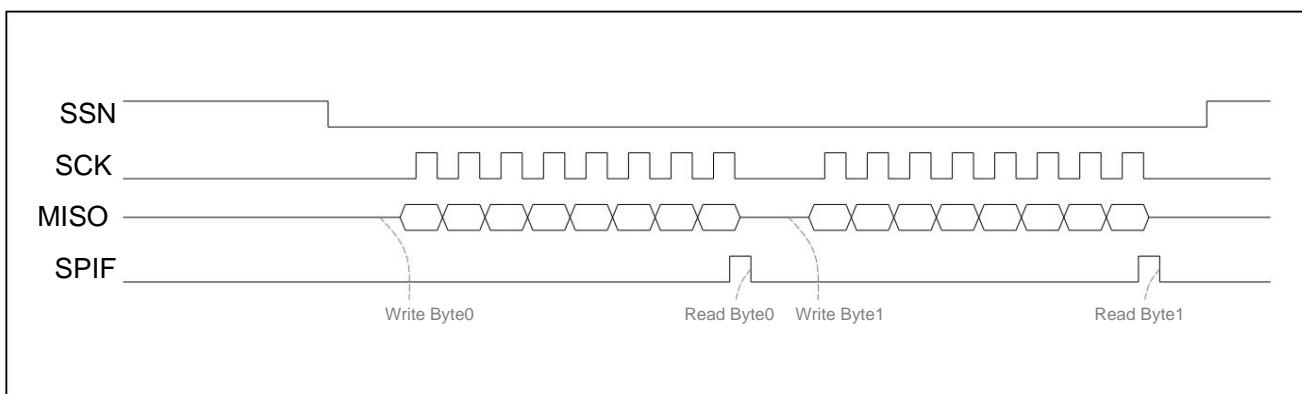


Figure 18-2 Schematic diagram of slave sending

### 18.3.3 SPI Data Frame Format

The SPI interface frame format depends on the configuration of the clock polarity bit CPOL and the clock phase bit CPHA.

When CPOL is 0, the idle state of the SCK line is low. When CPOL is 1, the SCK line idle state is high

level. When CPHA is 0, data is sampled on the first SCK clock transition signal transition. when CPHA

When 1, data is sampled on the second SCK clock transition.

The SPI interface host frame format is shown in the figure below.

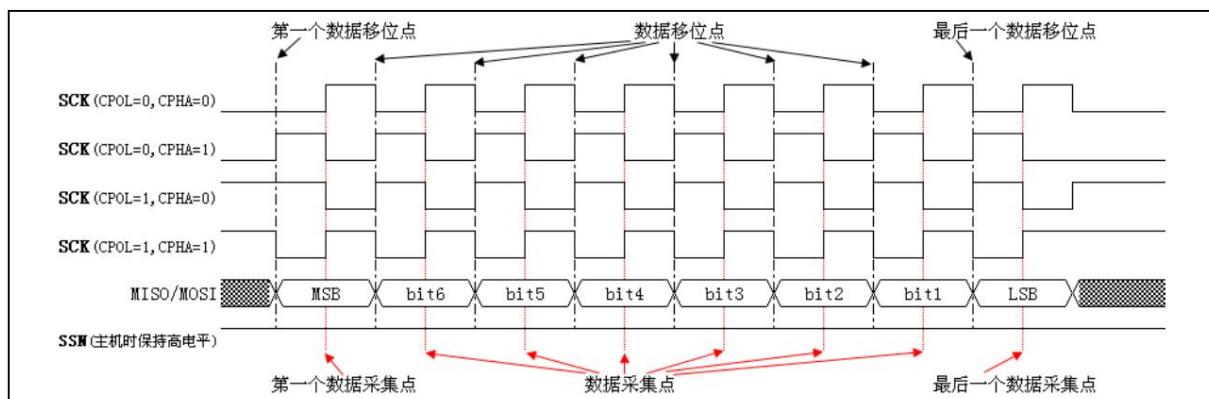


Figure 18-3 Host mode frame format

The SPI interface slave frame format is shown in the figure below.

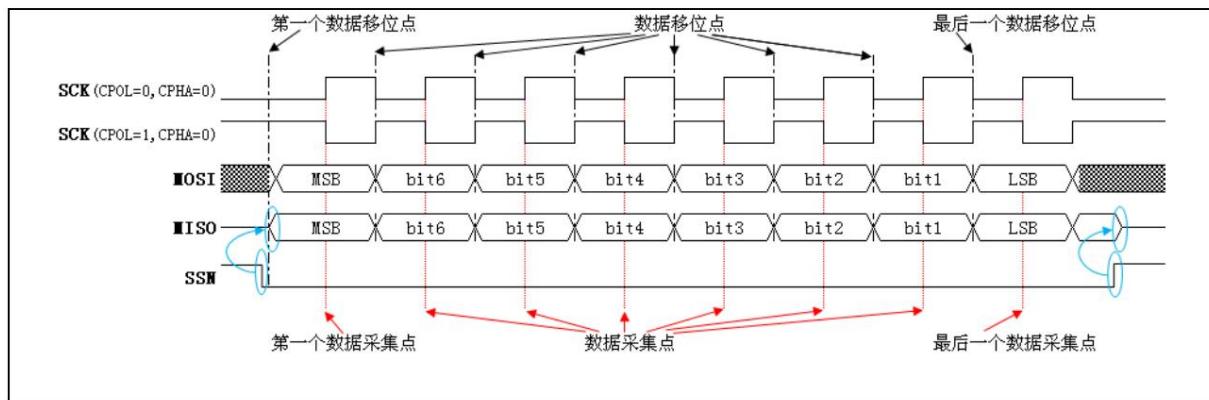


Figure 18-4 Data frame format when slave CPHA is 0

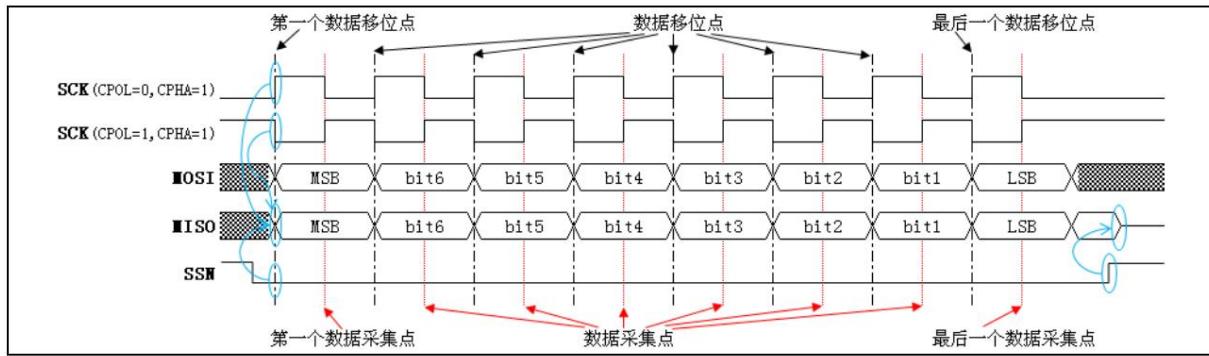


Figure 18-5 Data frame format when slave CHPA is 1

#### 18.3.4 SPI Status Flags and Interrupts

The SPI will generate the following three status flags during operation, and their generation conditions and clearing methods are as follows.

- ÿ When the transmission of a data frame is completed, SPIF will be set by hardware. Read the SPI\_DATA register action to clear this flag.

- ÿ When SPI works in master mode and external SSN input is low level, SPI\_STAT.mdf will be set by hardware, indicates that another SPI master is occupying the bus. When the SSN input is high, the SPI\_STAT.mdf will be hard files are automatically cleared.

- ÿ When the SPI works in slave mode, if the SSN pin is pulled high during data transmission, the 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 situations:

- ÿ SPI transfer is completed, that is, SPI\_STAT.SPIF is 1
- ÿ The host mode of SPI is wrong, that is, SPI\_STAT.mdf is 1

#### 18.3.5 SPI multi-machine system configuration description

- ÿ When the SPI module acts as a host and works in a single host system, it can control the SPI\_CS pin or GPIO pin.

control slave. When the SPI\_CS pin is selected as the slave chip select signal, set SPI\_SSNS.ssn to 0.

Select the slave and set SPI\_SSNS.ssn to 1 to release the slave. When the GPIO pin is selected as the slave chip

When selecting a signal, set the corresponding bit of the GPIOx\_OUT register to 0 to select the slave, and set the GPIOx\_OUT

The corresponding bit of the register is 1 to release the slave.

- ÿ When the SPI module is a slave, configure the source of SPI\_SSNS as needed (see GPIO port auxiliary control for details device). When the SSN is low, the slave can be selected for communication; when the SSN is high, the slave is in an idle state.

selected state.

When the SPI mode works with multiple masters and multiple slaves, all slave chip select signals are connected through GPIO pins.

The host must also connect the SSN signal of other hosts through GPIO pins to monitor whether the bus is occupied.

The operation method that Master0 needs to communicate as shown in the figure below is: wait for Master0.SSN to become high;

GPIO0 output low to inform Master1 to release the SPI bus; output low from GPIO2 to select Slave1; and

Slave1 communicates; output high from GPIO2 to release Slave1; output high from GPIO0 to release Master1.

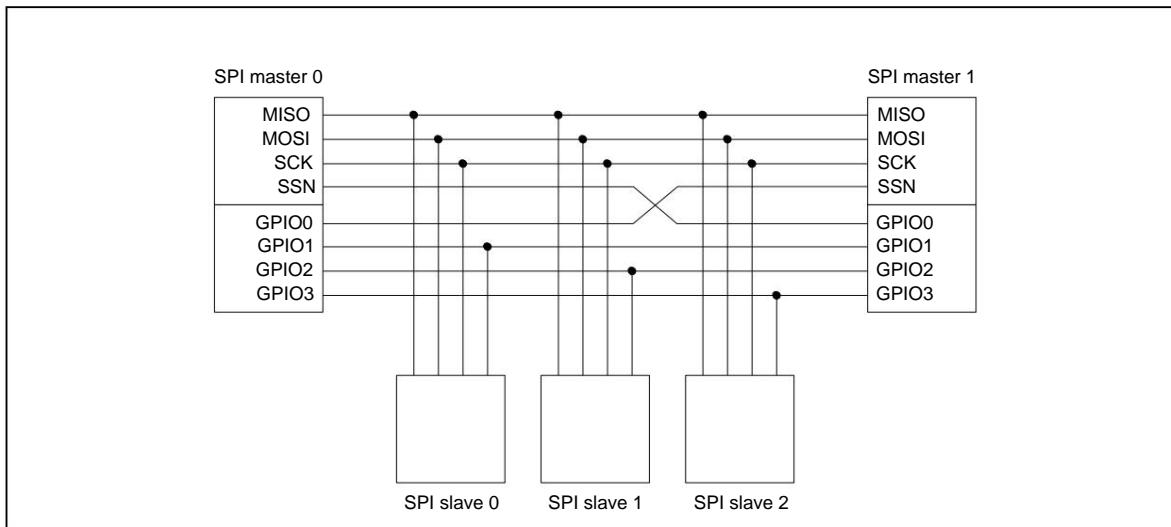


Figure 18-6 Schematic diagram of SPI multi-master/multi-slave system

### 18.3.6 SPI pin configuration description

The SPI can maintain some or all of its functions under some special pin configurations.

The specific situation is as follows ("ÿ" means the pin is configured and used, blank means the pin is not configured):

	SPI_CS(Master)/ SPI_SSN (slave)	SCK	MOSI	MISO	Function Description
host mode	ÿ	ÿ	ÿ	ÿ	General configuration All hosts function normally
		ÿ	ÿ	ÿ	All hosts function normally
	ÿ	ÿ	ÿ		The host sending function is normal
	ÿ	ÿ		ÿ	The host receiving function is normal
		ÿ	ÿ		The host sending function is normal
		ÿ		ÿ	The host receiving function is normal
slave mode	ÿ	ÿ	ÿ	ÿ	General configuration All slaves function normally
	ÿ	ÿ	ÿ		Slave receiving function is normal
	ÿ	ÿ		ÿ	Slave sending function is normal
	ÿ Fixed low level	ÿ	ÿ	ÿ	All slaves function normally
	ÿ Fixed low level	ÿ	ÿ		Slave receiving function is normal
	ÿ fixed low	ÿ		ÿ	Slave sending function is normal

Table 18-1 SPI pin configuration description table

#### Notice:

- The situations not listed in the table are not supported for the time being.

- In the host mode, even if the SPI\_CS chip select output is not used, it is necessary to set SPI.SSN to 1 before sending data.

After sending data, set SPI.SSN to 0.

- In slave mode and the chip select input is fixed low, SPI\_CR.cpha=1 must be satisfied to maintain normal function.

## 18.4 SPI Programming Example

### 18.4.1 SPI Master Transmission Example

Step1: Map CS/SCK/MISO/MOSI to

and configure the CS/SCK/MOSI pins as output mode and configure the MISO pins as input mode.

Step2: Set SPI\_CR.mstr to 1 to make SPI work in host mode.

Step3: Configure SPI\_CR[spr2:spr0] to make the clock rate output by SCK 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 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, jump to Step7 to continue execution.

Step10: Set SPI\_SSN.ssn to 1, make CS pin output high level to release the slave.

Notice:

– GPIO can be used instead of CS to achieve chip select output, which is mostly used in multi-computer communication systems.

– 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 Reception Example

Step1: Map CS/SCK/MISO/MOSI to

and configure the CS/SCK/MOSI pins as output mode and configure the MISO pins as input mode.

Step2: Set SPI\_CR.mstr to 1 to make SPI work in host mode.

Step3: Configure SPI\_CR[spr2:spr0] to make the clock rate output by SCK 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 CS pin output low level to select the slave.

Step7: Write arbitrary data to SPI\_DATA to trigger the host to send SCK.

Step8: The query waits 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, jump to Step7 to continue execution.

Step11: Set SPI\_SSNS.ssn to 1, make CS pin output high level to release the slave.

Notice:

– GPIO can be used instead of CS to achieve chip select output, which is mostly used in multi-computer communication systems.

– SPI\_SSNS.ssn must be set to 0 during the transmission process, and SPI\_SSNS.ssn must be set to 1 after the transmission is completed.

#### **18.4.3 SPI slave transmission example**

Step1: Map SSN/SCK/MISO/MOSI according to the related description of pin digital multiplexing function in GPIO chapter

to the required pins; and configure the SSN/SCK/MOSI pins as input mode, and configure the MISO pins as output mode

Mode. For the source of SSN pin, 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: Query and wait for the SSN pin to be pulled low, 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 receiver example**

Step1: Map SSN/SCK/MISO/MOSI according to the related description of pin digital multiplexing function in GPIO chapter

to the required pins; and configure the SSN/SCK/MOSI pins as input mode, and configure the MISO pins as output mode

Mode. For the source of SSN pin, 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: Query and wait for the SSN pin to be pulled low, the master selects the SPI slave.



Step6: The query waits for SPI\_STAT.SPIF to become 1, and the data sent by the host has been received.

Step7: Read the received data from SPI\_DATA.

Step8: If the data to be received is not completed, 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.

## 18.6 SPI register description

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

Table 18-2 SPI register list

### 18.6.1 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 spr1	Res	mstr cpol cpha spr1 spr0	RW RW RW RW RW RW				

Bit tag function description		
31:8	Reserved	
7	spr2 baud rate selection bit 2  See spr0.	
6	span  0 - forbidden  1 – enable	SPI module enable control
5	Reserved	
4	mstr  0 – Slave mode  1 – Host Mode	SPI working mode configuration
3	cpol  0 – low level  1 – high level	SCK line idle state configuration
2	cpha clock phase configuration  0 - first edge  1 – second edge	
1	spr1 baud rate select bit 1  reference spr0	

0	spr0 baud rate selection bit 0	spr2	spr1	spr0	SCK Rate
0	0	0	0	0	PCLK /2
0	0	0	1	1	PCLK /4
0	1	0	0	0	PCLK /8
0	1	0	1	1	PCLK /16
1	0	0	0	0	PCLK /32
1	0	0	1	1	PCLK/64
1	1	0	0	0	PCLK/128
1	1	1	1	1	Reserved

Table 18-3 Host mode baud rate selection



### 18.6.2 SPI Chip Select Configuration Register (SPI\_SSN)

Address offset: 0x04

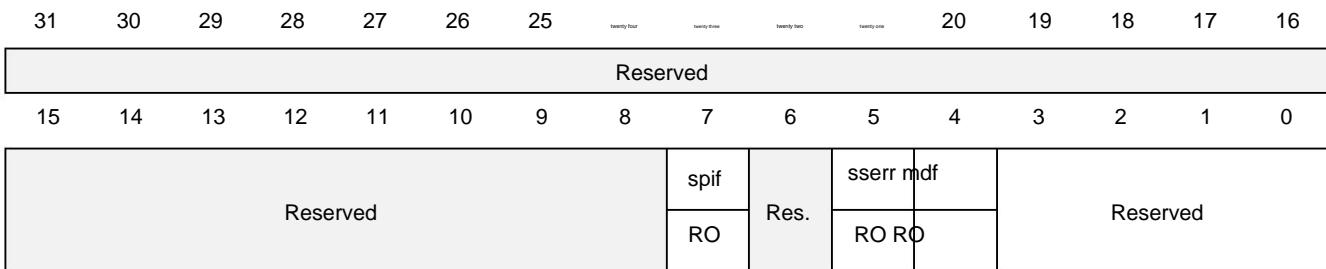
Reset value: 0x000000FF

Bit tag	function description	
31:1	Reserved	
0	SPI_CS output level configuration in ssn host mode	0: SPI_CS port output low level 1: SPI_CS port output high level

### 18.6.3 SPI Status Register (SPI\_STAT)

Address offset: 0x08

Reset value: 0x00000004



Bit tag function description	
31:8	Reserved
7	spf transfer completion flag 1: The SPI bus has completed a byte transfer 0: SPI bus is transferring
6	Reserved
5	sserr slave mode SSN error flag
4	Conflict flag in mdf host mode 1: SSN pin level is low 0: SSN pin level is high
3:0	Reserved

## 18.6.4 SPI Data Register (SPI\_DATA)

Address offset: 0x0c

Reset value: 0x00000000

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	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 tag	function description	
31:8	Reserved	
7:0	spdat data register	<p>In transmit mode, write the byte to be transmitted to this register;</p> <p>In receive mode, the received byte is read from this register;</p>

## 19 Clock Calibration Module (CLKTRIM)

### 19.1 Introduction to CLK\_TRIM

The CLK\_TRIM (Clock Trimming) module is a circuit dedicated to calibrating/monitoring the clock. in calibration mode

Select an accurate clock source to calibrate the inaccurate clock source, repeat the calibration, adjust the parameters of the inaccurate clock source, until the to the frequency of the clock source being calibrated to meet the accuracy requirements. There will be some error in the count value in calibration mode, but within the allowable accuracy error. In monitoring mode, select a stable clock source to monitor the system working clock.

During the monitoring period, monitor whether the working clock of the system fails and generate an interrupt. in calibration mode and

In monitor mode, all 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:

- ÿ Calibration mode
- ÿ Monitoring mode
- ÿ 32-bit reference clock counter can be loaded with initial value
- ÿ 32-bit clock counter to be calibrated with configurable overflow value
- ÿ 6 reference clock sources
- ÿ 4 clock sources to be calibrated
- ÿ 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 the reference clock to calibrate an inaccurate time to be calibrated.

Zhongyuan.

The software is repeatedly calibrated according to the following operation process, and the parameters of the clock source to be calibrated are adjusted until the clock source to be calibrated

Meet the frequency accuracy requirements.

#### 19.3.1.1 Operation process

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 the CLKTRIM\_CR.IE register to enable interrupts.
5. Set the CLKTRIM\_CR .trim\_start register to start the calibration.
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, triggering an interrupt.
8. The interrupt service routine determines that CLKTRIM\_IFR.stop is 1, and reads the register CLKTRIM\_REFCNT  
and the value of CLKTRIM\_CALCNT,
9. Clear the CLKTRIM\_CR .trim\_start register to end the calibration.

Notice:

– During the calibration process, the calibration time may be set too long, and the clock counter to be calibrated may

In case of overflow before CLKTRIM\_IFR.stop is set to 1, CLKTRIM\_IFR.calcnt\_of is set to 1, triggering  
interrupt. When the interrupt service routine finds that CLKTRIM\_IFR. calcnt\_of is set to 1, it is cleared  
CLKTRIM\_CR .trim\_start register ends 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 calibration.

## 19.3.2 CLK\_TRIM monitor mode

The monitoring mode is mainly used to select a stable clock source as the reference clock, and monitor the system under the set time period.

The abnormal state of the system operating clock. Only external XTH clock or external XTL clock can be selected in monitor mode as the monitored clock.

### 19.3.2.1 Operation process

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. ccnval register to the monitored clock counter overflow time.
5. Set the CLKTRIM\_CR .mon\_en register to enable the monitoring function.
6. Set the CLKTRIM\_CR.IE register to enable interrupts.
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 count of the reference clock counter reaches the monitoring interval, judge whether the monitored clock counter overflows.

If it overflows, it means that the monitored clock is working properly. If there is no overflow indicating that the monitored clock has failed,

CLKTRIM\_IFR .xtal32k\_fault/xtal32m\_fault Set to 1 to trigger an interrupt.

10. Handle the interrupt service routine and clear the interrupt flag bit

CLKTRIM\_IFR .xtal32k\_fault/xtal32m\_fault, clear CLKTRIM\_CR .trim\_start register

End monitoring.

## 19.4 CLK\_TRIM Register Description

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 register.
0x14	CLKTRIM_ICLR	RW	interrupt flag clear register
0x18	CLKTRIM_CALCON	RW	Calibration counter overflow value configuration register

Table 19-1 Register List

### 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			

Bit tag	function 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-be-calibrated/monitored clock selection 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	trim/monitor 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 tag	function 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 tag	function 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 tag	function 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	
										xth_f ault	xtl_f ault	calcn t_of	stop	

Bit tag function description														
31:4	Reserved													
3	xth_fault XTH failure flag.													
		Write 0 to CLKTRIM_ICLR.xth_fault_clr to clear this flag												
2	xtl_fault XTL failure flag.													
		Write 0 to CLKTRIM_ICLR.xtl_fault_clr to clear this flag												
1	calcnt_of Calibrate counter overflow flag.													
		Write 0 to CLKTRIM_CR.start to clear this flag												
0	stop Reference counter stop flag.													
		Write 0 to CLKTRIM_CR.start to clear this flag												

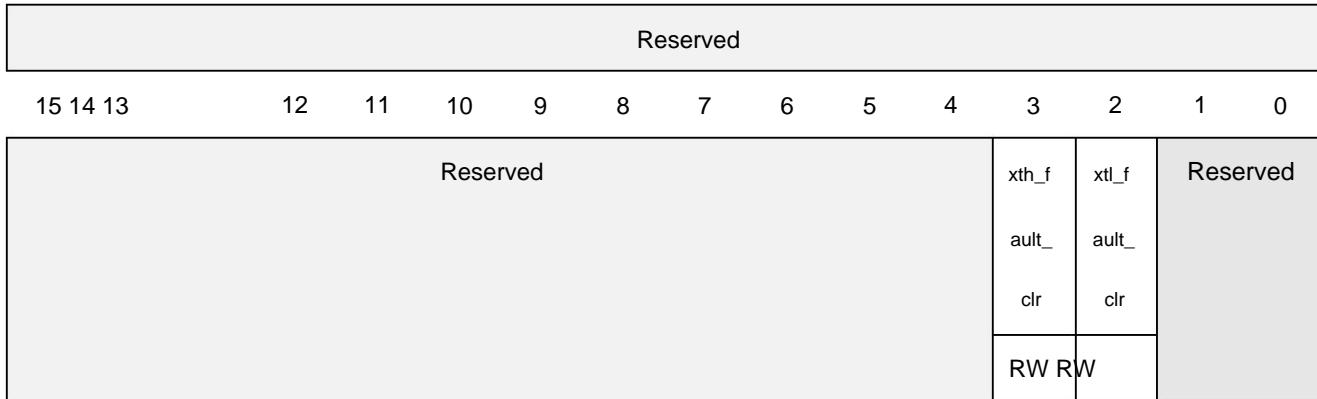
### 19.4.6 Interrupt Flag Clear Register (CLKTRIM\_ICLR)

Offset address: 0x14

Reset value: 0xf

31 30 29 28 27 26 25 24 23 22 21 20 19

18 17 16



bit mark		Function 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 tag	function description
31:0	ccntval Calibration counter overflow value



## 20 Cyclic Redundancy Check (CRC)

### 20.1 Overview

Cyclic Redundancy Check (CRC) calculation unit takes as input a stream or block of data, under the control of a generator polynomial

Generate an output number. This output number is often used to verify the correctness and integrity of data transmission or storage. This module supports

It supports calculating CRC value and checking CRC value.

### 20.2 Main Features

- ÿ One implementation standard: ISO/IEC13239

- ÿ One encoding method: CRC-16,  $x^{16} + x^{12} + x^5 + 1$

- ÿ Three write bit widths: 8bit, 16bit, 32bit

- ÿ Two working modes: CRC encoding mode, CRC check mode

### 20.3 Functional Description

#### 20.3.1 Working Mode

This module supports two working modes: CRC encoding mode and CRC check mode.

The CRC encoding mode is to input a certain amount of raw data to the CRC module and obtain the output generated by the CRC module

value (CRC\_RESULT.RESULT). The CRC check mode is to input a certain number of raw numbers to the CRC module.

According to the +CRC check value, verify whether the original data matches the CRC check value (CRC\_RESULT.FLAG).

#### 20.3.2 Encoding method

This module supports CRC-16 encoding, its calculation result is 16 bits, and the generator 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 "consistent bit width",

The principle of "low first and then high", that is, "every time data is written, it must be written to a register that is equal to the bit width of the valid data.

memory, and the lower-order data is written before the higher-order data".

The following shows how to write the same sequence data with three bit widths, and the output results are the same.



- ÿ 8bit bit width write: 0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77
- ÿ 16bit bit width write: 0x1100, 0x3322, 0x5544, 0x7766
- ÿ 32bit bit width write: 0x33221100, 0x77665544

## 20.4 Programming Examples

### 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 8bit, 16bit, 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 turn, and the write bit width can be 8bit, 16bit, 32bit.

Step 3: The value of 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	describe
CRC_RESULT	0x04	CRC result register
CRC_DATA	0x80	CRC data register



## 20.5.2 RESULT 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 sign		describe
16	FLAG CRC check result	0: 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 Writing 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 the data that needs to be calculated, and supports 3 write bit widths 8bit writing method: * ((uint8_t *)0x40020980) = 0XX 16bit writing method: * ((uint16_t *)0x40020980) = 0XXXX 32bit writing method: * ((uint32_t *)0x40020980) = 0XXXXXXXXX

## 21 Analog-to-Digital Converter (ADC)

### 21.1 Module Introduction

External analog signals need to be converted into digital signals for further processing by the MCU. This series integrates a

A 12-bit high precision, high conversion rate successive approximation analog-to-digital converter (SAR ADC) module. Has the following characteristics:

ÿ 12-bit conversion precision;

ÿ 1Msps conversion speed (VCC>2.7V);

ÿ 12 conversion channels: 9 pin channels, built-in temperature sensor, built-in 1.2v reference voltage, 1/3 power supply

pressure;

ÿ 4 kinds of reference sources: power supply voltage, ExRef pin, built-in 1.5v reference voltage, built-in 2.5v reference voltage;

ÿ ADC voltage input range: 0~Vref;

ÿ 3 conversion modes: single conversion, continuous conversion, cumulative conversion;

ÿ The conversion rate of ADC can be configured by software;

ÿ Built-in signal amplifier, which can convert high-impedance signals;

ÿ Supports on-chip peripherals to automatically trigger ADC conversion, effectively reducing chip power consumption and improving real-time conversion.

### 21.2 ADC Block Diagram

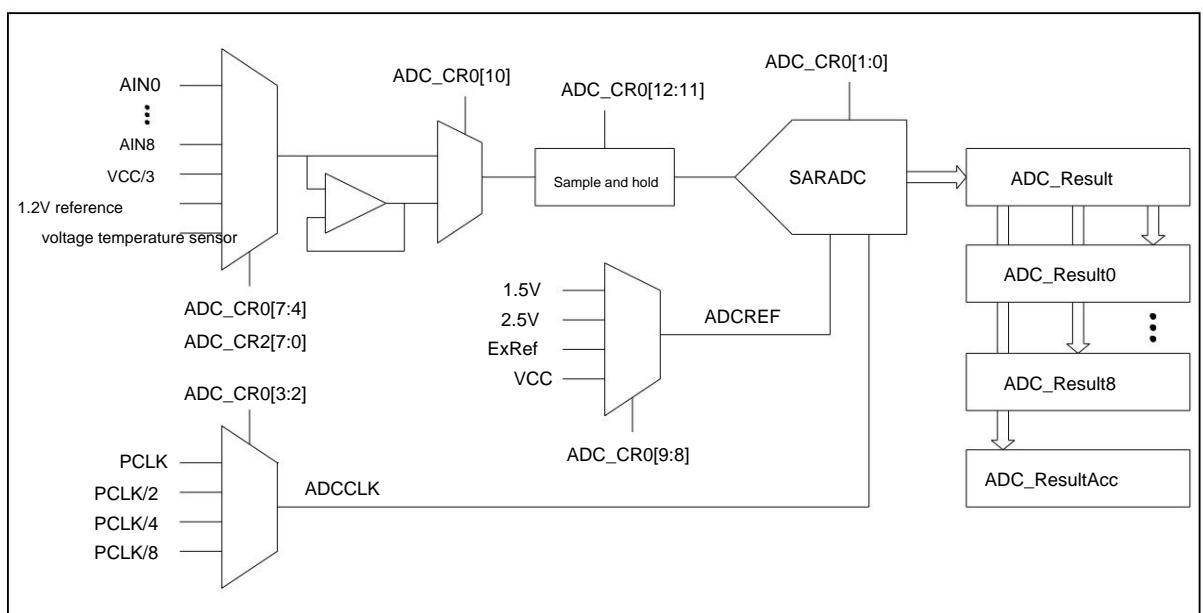


Figure 21-1 ADC schematic block diagram

### 21.3 Conversion timing and conversion speed

The conversion timing of ADC is shown in the following figure: A complete ADC conversion consists of sampling process and successive comparison process.

The sampling process requires 4~12 AdcClks, which are configured by ADC\_CR0.SAM; the successive comparison process requires 16 AdcClk. Therefore, a total of 20~28 AdcClks are required for one ADC conversion.

The unit of ADC conversion speed is sps, which is the number of ADC conversions per second. How to calculate ADC conversion speed

The method is: the frequency of AdcClk / the number of AdcClk required for one ADC conversion.

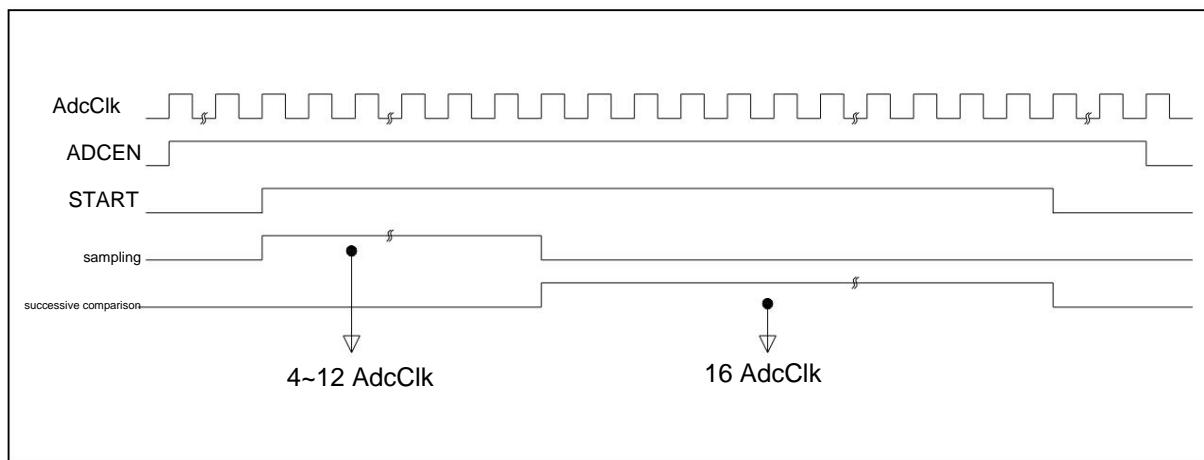


Figure 21-2 ADC conversion timing diagram

The ADC conversion speed is related to the ADC reference voltage and VCC voltage. The maximum conversion speed is shown in the following table:

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 is started, and all 12 ADC channels can be converted.

Change. This mode can be started either by setting the ADC\_CR0.START bit or by setting the ADC\_CR1[9:0] external

External trigger start. Once the ADC conversion of the selected channel is completed, the ADC\_CR0.START bit is automatically cleared and the conversion ends.

The result is stored in the ADC\_result register.

Start the **ADC** single conversion operation flow 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      **ADC**      If the reference voltage does not select the external reference voltage pin, you can skip this step.

Step3: Set BGR\_CR.BGR\_EN to 1 to enable the BGR module.

Step4: Set ADC\_CR0.ADCEN to 1 to enable ADC module.

Step5: Delay 20uS and wait for the ADC and BGR modules to start up.

Step6: Set ADC\_CR1.CT to 0 and select single conversion mode.

Step7: Configure ADC\_CR0. SREF and 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 and 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 ADC\_result register to get ADC conversion result

fruit.

Step12: To convert other channels, repeat Step9~Step11.

Step13: Set ADC\_CR0.ADCEN and BGR\_CR.BGR\_EN to 0, turn off ADC module, BGR module

Piece.

Start the **ADC** single conversion operation flow by 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      **ADC**      If the reference voltage does not select the external reference voltage pin, you can skip this step.



Step3: Set BGR\_CR.BGR\_EN to 1 to enable the BGR module.

Step4: Set ADC\_CR0.ADCEN to 1 to enable ADC module.

Step5: Delay 20uS and wait for the ADC and BGR modules to start up.

Step6: Set ADC\_CR1.CT to 0 and 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 ADC interrupt in NVIC interrupt vector table.

Step11: Configure ADC\_CR0. SREF and 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 and select the channel to be converted.

Step14: Set ADC\_IFR to 0x00, clear ADC interrupt flag.

Step15: Configure ADC\_CR1. TRIGS1 and ADC\_CR1. TRIGS0, and select external trigger conditions.

Step16: When the external trigger condition triggers the ADC to complete the conversion, the ADC module will generate an interrupt. The user can use the ADC

Read the ADC\_result register in the interrupt service routine to get 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 ADC module, BGR module

Piece.

## 21.5 Continuous Conversion Mode

In continuous conversion mode, starting the ADC once can perform multiple conversions on multiple channels in sequence; the convertible ADC

The channels are AIN0~AIN7. The total number of ADC conversions is configured by ADCCR2.ADCCNT;

The channel is configured by ADC\_CR2[7:0]. This mode can either be started by setting the ADC\_CR0.START bit or

Start by setting the external trigger of ADC\_CR2[9:0]. After starting continuous conversions, the ADC module converts sequentially

Channels in AIN0~AIN7 to be converted 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 result will be stored in the corresponding conversion channel.

ADC\_result0~ ADC\_result7 registers. If the total number of conversions is greater than the number of ADC channels to be converted,

Then ADC\_result0~ADC\_result7 registers only save the last conversion result.

The figure below demonstrates the process of 10 consecutive conversions of AIN0, AIN1, and AIN5. START by register

After being set to 1, the state machine inside the ADC converts AIN0, AIN1, and AIN5 in turn until ADCCNT

The count value becomes 0.

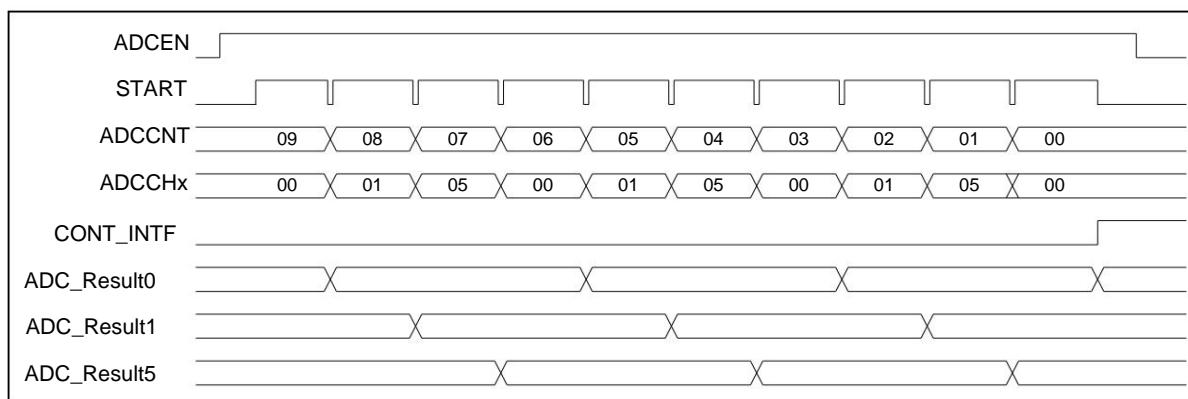


Figure 21-3 Example of ADC Continuous Conversion Process

Start the **ADC** continuous conversion operation flow 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 **ADC** If the reference voltage does not select the external reference voltage pin, you can skip this step.

Step3: Set BGR\_CR.BGR\_EN to 1 to enable the BGR module.

Step4: Set ADC\_CR0.ADCEN to 1 to enable ADC module.

Step5: Delay 20uS and wait for the ADC and BGR modules to start up.

Step6: Set ADC\_CR1.CT to 1 and select continuous conversion mode.



Step7: Set ADC\_CR1[14:12] to 0, and disable the conversion result comparison function.

Step8: Configure ADC\_CR2. ADCCNT, and select the total number of conversions for continuous conversion.

Step9: Configure ADC\_CR0. SREF and 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] and select the channel to be converted.

Step12: Set ADC\_ICLR.CONT\_INTC to 0, clear ADC\_IFR.CONT\_INTF flag.

Step13: Set ADC\_CR0. StateRst to 1 to 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 get 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 ADC module, BGR module

Piece.

## 21.6 Continuous Conversion Accumulation Mode

In continuous conversion accumulation mode, starting the ADC once can perform multiple conversions on multiple channels and

The results are accumulated; the convertible ADC channels are AIN0~AIN7. The total number of ADC conversions is given by

ADCCR2.ADCCNT is configured; the channel to be converted is configured by ADC\_CR2[7:0]. This mode is both

It 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 conversion times are completed.

become. After the ADC module completes the total number of conversions, the ADC\_IFR.CONT\_INTF bit will be automatically set to 1, and the

The accumulated value is stored in the ADC\_result\_acc register.

The following figure demonstrates the process of accumulating 10 consecutive conversions on AIN0, AIN1, and AIN5. by register

After START is set to 1, the state machine inside the ADC converts AIN0, AIN1, and AIN5 in turn, until

The count value of ADCCNT becomes 0. The ADC\_result\_acc register is automatically accumulated every time a conversion is completed.

add. The conversion results of AIN0, AIN1, and AIN5 given in the figure are 0x010, 0x020, and 0x040 in turn.

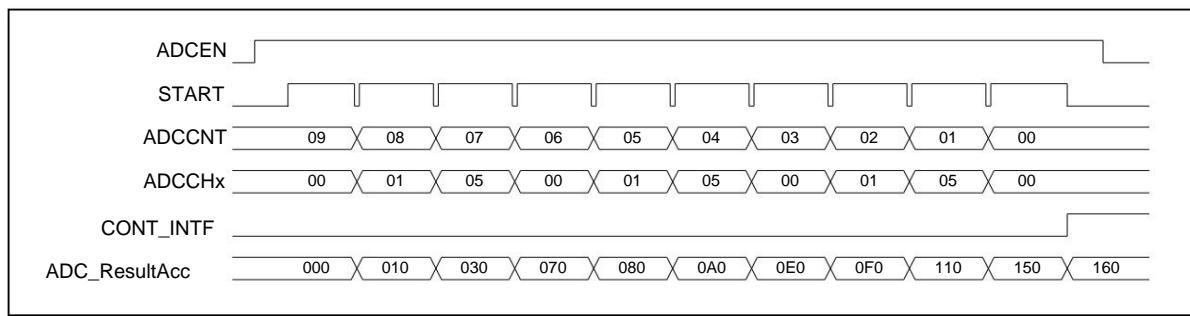


Figure 21-4 Example of ADC continuous conversion accumulation process

Start the **ADC** continuous conversion and accumulation operation flow 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      **ADC**      If the reference voltage does not select the external reference voltage pin, you can skip this step.

Step3: Set BGR\_CR.BGR\_EN to 1 to enable the BGR module.

Step4: Set ADC\_CR0.ADCEN to 1 to enable ADC module.

Step5: Delay 20uS and wait for the ADC and BGR modules to start up.

Step6: Set ADC\_CR1.CT to 1 and select continuous conversion mode.

Step7: Set ADC\_CR1[14:12] to 0, and disable 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, select the total number of conversions for continuous conversion.

Step10: Configure ADC\_CR0.SREF and 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] and select the channel to be converted.

Step13: Set ADC\_ICLR.CONT\_INTC to 0, clear ADC\_IFR.CONT\_INTF flag.

Step14: Set ADC\_CR1.RACC\_CLR to 0, clear ADC\_result\_acc register.

Step15: Set ADC\_CR0.StateRst to 1 to 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 ADC\_result\_acc register to get conversion

Result accumulation value.

Step18: To convert other channels, repeat Step12~Step17.

Step19: Set ADC\_CR0.ADCEN and BGR\_CR.BGR\_EN to 0, turn off ADC module, BGR module

Piece.

## 21.7 Comparison of ADC Conversion Results

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, interval value comparison. This function needs to set the corresponding control bits HtCmp, LtCmp, RegCmp

1. This function can realize the automatic monitoring of the analog quantity, until the ADC conversion result is in line with the user's expectation, the middle

Interrupt application for user program interface.

Upper threshold comparison: when the ADC conversion result is within the range of [ADC\_HT, 4095], then ADC\_IFR.HHT\_INTF

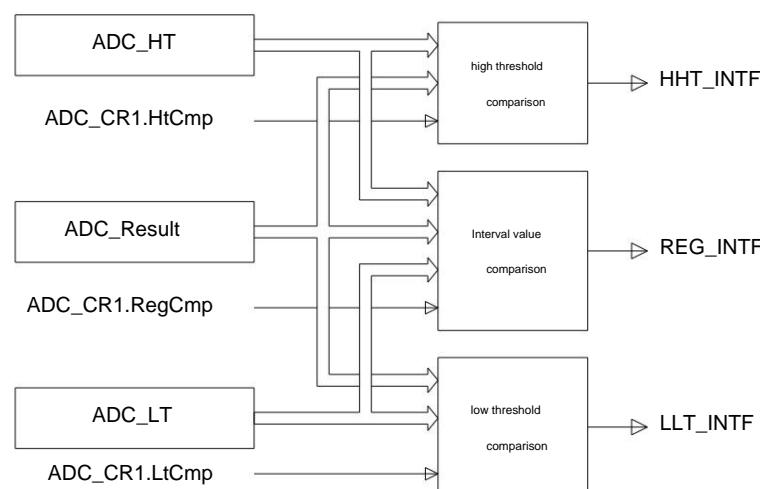
Set to 1; writing 0 to ADC\_ICLR.HHT\_INTC clears ADC\_IFR.HHT\_INTF.

Lower threshold comparison: ADC\_IFR.LLT\_INTF is set to 1 when the ADC conversion result is in the range of [0, ADC\_LT);

Writing 0 to ADC\_ICLR.LLT\_INTC clears ADC\_IFR.LLT\_INTF.

Interval value comparison: When the ADC conversion result is in the interval [ADC\_LT, ADC\_HT), then ADC\_IFR.

REG\_INTF is set; writing 0 to ADC\_ICLR.REG\_INTC clears ADC\_IFR.REG\_INTF.



## 21.8 ADC Interrupts

The ADC interrupt requests are shown in the following table:

interrupt source	interrupt flag	interrupt enable
ADC continuous conversion completed	ADC_IFR.CONT_INTF	ADC_CR0.IE
The ADC conversion result is located in the interval value area	ADC_IFR.REG_INTF	
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 Measuring ambient temperature with a temperature sensor

The output voltage of the temperature sensor will change with the change of the ambient temperature, so according to the output voltage of the temperature sensor, the

The corresponding ambient temperature can be calculated. When the measurement channel of the ADC module selects the output voltage of the temperature sensor, that is

Ambient temperature can be measured.

Calculated as follows:

$$\text{Ambient temperature} = 25 + 0.0839 \times V_{ref} \times (\text{AdcValue} - \text{Trim})$$

Among them:  $V_{ref}$  is the reference voltage of the current ADC module, the value is 1.5 or 2.5.

$\text{AdcValue}$  is the result of the ADC module measuring the output voltage of the temperature sensor, ranging from 0 to 4095.

$\text{Trim}$  is a 16-bit calibration value, which needs to be read from the Flash memory during calculation, and its storage address is detailed below

surface.

ADC reference voltage calibration value storage address	calibration value accuracy
Internal 1.5V	0x00100C34
Internal 2.5V	0x00100C36

A calculation example is as follows:

Condition 1:  $V_{ref}=2.5$ ,  $\text{AdcValue}=0x7E5$ ,  $\text{Trim}=0x76C$ :

$$\text{Temperature 1: } 25 + 0.0839 \times 2.5 \times (0x7E5 - 0x76C) = 50^{\circ}\text{C}.$$

Condition 2:  $V_{ref}=1.5$ ,  $\text{AdcValue}=0x72D$ ,  $\text{Trim}=0x76C$ :

$$\text{Temperature 3: } 25 + 0.0839 \times 1.5 \times (0x72D - 0x76C) = 17^{\circ}\text{C}.$$



The operation process of measuring ambient temperature through ADC :

- Step1: Set BGR\_CR.BGR\_EN to 3 to enable BGR module and temperature sensor module.
- Step2: Set ADC\_CR0.ADCEN to 1 to enable ADC module.
- Step3: Delay 20uS, wait for the ADC and BGR modules to start up.
- Step4: Set ADC\_CR1.CT to 0 and 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, and 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 ADC\_result register to get ADC conversion result  
fruit.
- Step11: Set ADC\_CR0.ADCEN and BGR\_CR.BGR\_EN to 0, turn off ADC module, BGR module  
Piece.
- Step12: Read the calibration value of the temperature sensor and calculate the current ambient temperature according to the formula.

## 21.10 ADC Module Registers

base address 0x40002400

register	offset address	describe
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		ADC conversion result accumulated value
ADC_HT	0x058	ADC Compare Upper Threshold
ADC_LT	0x05C	ADC Compare Lower Threshold
ADC_IFR	0x060	ADC Interrupt Flag Register
ADC_ICLR	0x064	ADC Interrupt Clear Register
ADC_result	0x068	ADC conversion result

Table 21-1 ADC registers

### 21.10.1 ADC Configuration Register 0 (ADC\_CR0)

offset address 0x004

Reset value 0x000013F0

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	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	BUF EN	SREF		SEL		CLKSEL		STA RT	ADC EN			
R/W/R/W			R/W	R/W	R/W		R/W		R/W		R/W	R/WR/W			

bit mark		Function description
31:16	Reserved	Reserve
15	StateRst	ADC Continuous Conversion Status Control 1: Reset ADC continuous conversion state 0: invalid
14	IE	ADC Interrupt Control 1: Enable interrupt 0: Disable interrupts
13	Reserved	Reserve
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, 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 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: Power supply voltage
7:4	SEL	ADC conversion channel selection (single conversion mode) 0000: Select channel 0 to input P2.4 0001: Select channel 1 to input P2.6 0010: Select channel 2 to input P3.2 0011: Select channel 3 to 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 Note: ADC_CR0.BUFEN must be 1 1010: Built-in temperature sensor output voltage Note: ADC_CR0.BUFEN must be 1 1011: Internal reference 1.2V output voltage Note: ADC_CR0.BUFEN must be 1
3:2 CLK\$EL		be 1 ADC clock selection 00: PCLK clock  01: PCLK clock divided by 2 10: PCLK clock divided by 4 11: PCLK clock divided by 8
1	START	ADC conversion control 1: Start ADC conversion 0: Stop ADC conversion ADC enable
0	ADCEN	control 1: Enable ADC 0: Disable ADC

## 21.10.2 ADC Configuration Register 1 (ADC\_CR1)

offset address 0x008

Reset value 0x00007000

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RACC _CLR	RegCm p	HtCmp	LtCmp	HtCmp CT			TRIGS1			TRIGS0					
R/RW/W	R/RW/W	R/RW/W					R/W			R/W					

bit mark		Function description
31:16 Reserved		Reserve
15	RACC_CLR	The ADC conversion result accumulation register is cleared  1: no effect;  0: The ADC conversion result accumulation register (ADC_result_acc) is cleared.  Note: This bit is read as 0, so it is necessary to pay special attention to the value of this bit when operating this register to prevent malfunction.
14	RegCmp	ADC interval comparison control  1: Enable interval comparison  0: prohibit interval comparison
13	HtCmp	ADC High Threshold Compare Control  1: Enable high threshold comparison  0: disable high threshold comparison
12	LtCmp	ADC Low Threshold Compare Control  1: Enable low threshold comparison  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 option 2:  00000: Disable automatic triggering of ADC conversions  00001: Timer0 interrupt, automatically trigger ADC conversion  00010: Timer1 interrupt, automatically trigger ADC conversion  00011: Timer2 interrupt, automatically trigger ADC conversion  00100: LPTimer interrupt, automatically trigger ADC conversion  00101: Timer4 interrupt, automatically trigger ADC conversion



		<p>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, automatically trigger ADC conversion      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</p> <p>Note:</p> <p>1) TIM4/5/6 interrupt triggers automatic conversion of ADC. In addition to enabling the corresponding interrupt of TIM4/5/6, it is also necessary to configure the spread spectrum and interrupt trigger selection register TIMX_CR of Advanced Timer to select the interrupt source that can trigger ADC. 2) The rising edge of each interrupt flag bit is used to trigger the ADC. 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.</p>
4:0	TRIGS0	<p>ADC conversion automatic trigger      selection 2: 00000: Disable automatic trigger ADC      conversion 00001: Select Timer0 interrupt, automatically trigger ADC      conversion 00010: Select Timer1 interrupt, automatically trigger ADC      conversion 00011: Select Timer2 interrupt, automatically trigger ADC      conversion 00100: Select LPTimer interrupt, Automatically trigger ADC      conversion 00101: Select Timer4 interrupt, automatically trigger ADC      conversion 00110: Select Timer5 interrupt, automatically trigger ADC      conversion 00111: Select Timer6 interrupt, automatically trigger ADC      conversion 01000: Select UART0 interrupt, automatically trigger ADC conversion</p>

	<p>01001: Select UART1 interrupt, automatically trigger ADC conversion</p> <p>01010: Select LPUART interrupt, automatically trigger ADC conversion</p> <p>01011: Select VC0 interrupt, automatically trigger ADC conversion</p> <p>01100: Select VC1 interrupt, automatically trigger ADC conversion</p> <p>01101: Select RTC interrupt, automatically trigger ADC conversion</p> <p>01110: Select PCA interrupt, automatically trigger ADC conversion</p> <p>01111: Select SPI interrupt, automatically trigger ADC conversion 10000: Select P01 interrupt, automatically trigger ADC conversion 10001: Select P02 interrupt, automatically trigger ADC conversion 10010: Select P03 interrupt, automatically trigger ADC conversion 10011: Select P14 interrupt, automatically trigger ADC conversion 10100: Select P15 interrupt, automatically trigger ADC conversion 10101: Select P23 interrupt, automatically trigger ADC conversion 10110: Select P24 interrupt, automatically trigger ADC conversion 10111: Select P25 interrupt, automatically trigger ADC conversion 11000: Select P26 interrupt, automatically trigger ADC conversion 11001: Select P27 interrupt, automatically trigger ADC conversion 11010: Select P31 interrupt, automatically trigger ADC conversion 11011: Select P32 interrupt, automatically trigger ADC conversion 11100: Select P33 interrupt, automatically trigger ADC conversion 11101: Select P34 interrupt, automatically trigger ADC conversion 11110: Select P35 interrupt, automatically trigger ADC conversion 11111: Select P36 interrupt, automatically trigger ADC conversion</p> <p><b>Note:</b></p> <p>1) The TIM4/5/6 interrupt triggers the automatic conversion of the ADC. In addition to enabling the corresponding interrupt of the TIM4/5/6, it is also necessary to configure the spread spectrum and interrupt trigger selection register TIMX_CR of the Advanced Timer to select the interrupt source that can trigger the ADC. 2) The rising edge of each interrupt flag bit is used to trigger the ADC. 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.</p>
--	---

### 21.10.3 ADC Configuration Register 2 (ADC\_CR2)

offset address 0x00C

Reset value 0x00000000

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	Twenty	19	18	17	16	
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
ADCCNT								CH7EN	CH6EN	CH5EN	CH4EN	CH3EN	CH2EN	CH1EN		CH0EN
R/W								R/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	/W		

bit mark		Function description
31:16	Reserved	Reserve
15:8	ADCCNT	ADC continuous conversion times configuration 0: Continuous conversion once 1: 2 consecutive conversions ... 255: 256 consecutive conversions
7	CH7EN	ADC Continuous Conversion Channel 7 Enable 1: enable 0: Disable
6	CH6EN	ADC Continuous Conversion Channel 6 Enable 1: enable 0: Disable
5	CH5EN	ADC continuous conversion channel 5 enable 1: enable 0: Disable
4	CH4EN	ADC Continuous Conversion Channel 4 Enable 1: enable 0: Disable
3	CH3EN	ADC continuous conversion channel 3 enable 1: enable 0: Disable
2	CH2EN	ADC Continuous Conversion Channel 2 Enable 1: enable 0: Disable
1	CH1EN	ADC continuous conversion channel 1 enable 1: enable 0: Disable
0	CH0EN	ADC Continuous Conversion Channel 0 Enable



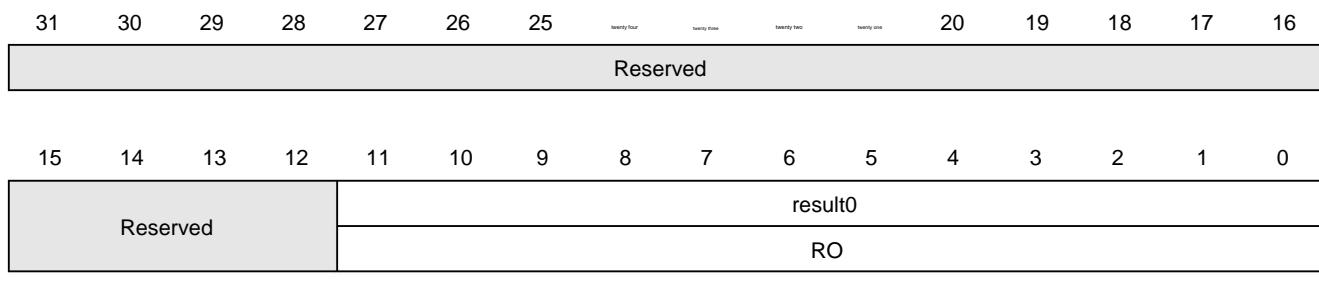
		1: enable 0: Disable
--	--	-------------------------



#### 21.10.4 ADC channel 0 conversion result (ADC\_result0)

offset address 0x030

Reset value 0x00000000

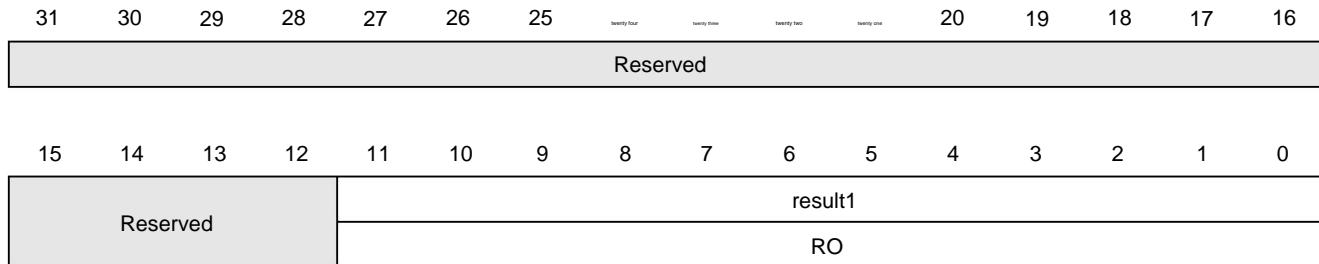


bit mark		Function description
31:12	Reserved	Reserve
11:0	result0	ADC channel 0 conversion result

#### 21.10.5 ADC channel 1 conversion result (ADC\_result1)

offset address 0x034

Reset value 0x00000000



bit mark		Function description
31:12	Reserved	Reserve
11:0	result1	ADC channel 1 conversion result



### 21.10.6 ADC channel 2 conversion result (ADC\_result2)

offset address 0x038

Reset value 0x00000000

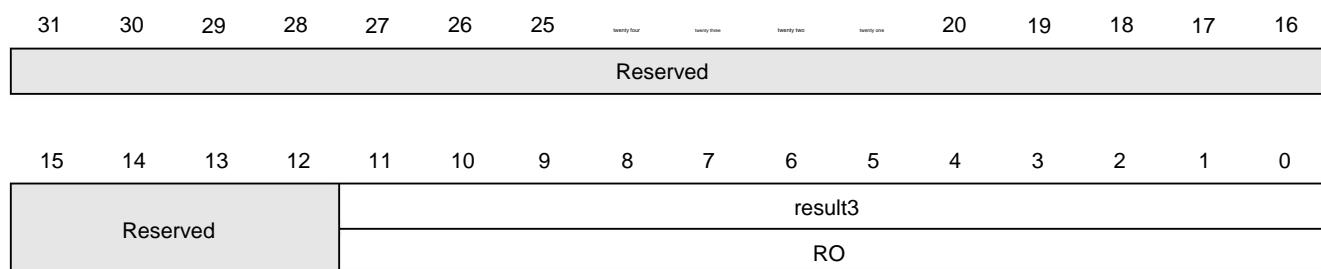


bit mark		Function description
31:12	Reserved	Reserve
11:0	result2	ADC channel 2 conversion result

### 21.10.7 ADC channel 3 conversion result (ADC\_result3)

offset address 0x03C

Reset value 0x00000000



bit mark		Function description
31:12	Reserved	Reserve
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	Twenty four	Twenty three	Twenty two	Twenty one	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	mark	Function description
31:12	Reserved	Reserve
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	Twenty four	Twenty three	Twenty two	Twenty one	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 mark		Function description
31:12	Reserved	Reserve
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	twenty four	twenty three	twenty two	twenty one	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	
Reserved	RO					

bit mark		Function description
31:12	Reserved	Reserve
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	twenty four	twenty three	twenty two	twenty one	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										
Reserved	RO														

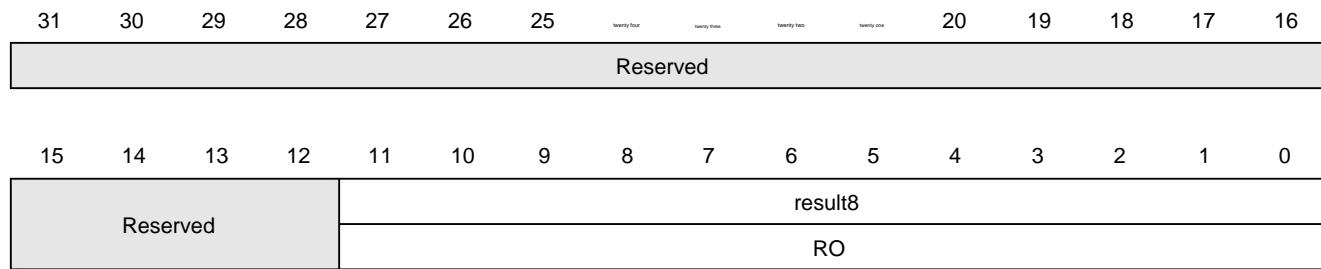
bit mark		Function description
31:12	Reserved	Reserve
11:0	result7	ADC channel 7 conversion result



### 21.10.12 ADC channel 8 conversion result (ADC\_result8)

offset address 0x050

Reset value 0x00000000

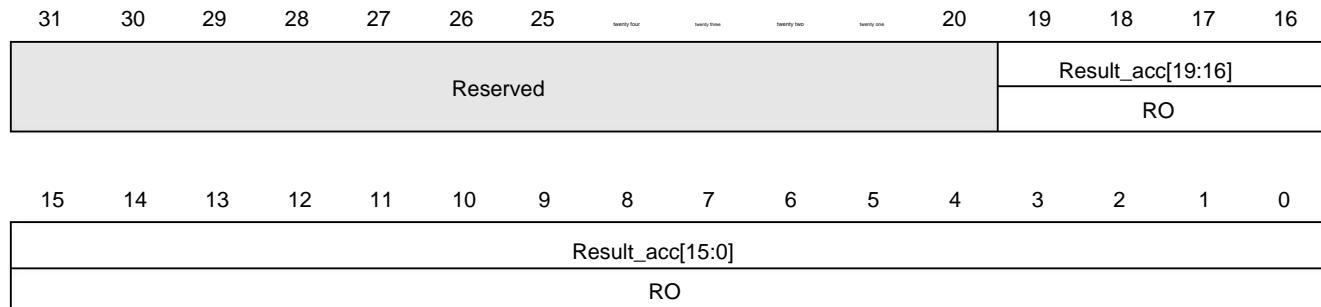


bit mark		Function description
31:12	Reserved	Reserve
11:0	result8	ADC channel 8 conversion result

### 21.10.13 ADC conversion result accumulation value (ADC\_result\_acc)

offset address 0x054

Reset value 0x00000000



bit mark		Function description
31:20	Reserved	Reserve
19:0	Result_acc	ADC conversion accumulated value



### 21.10.14 ADC Compare Upper Threshold (ADC廖)

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 mark		Function description
31:12	Reserved	Reserve
11:0	HT	ADC conversion result comparison upper threshold

### 21.10.15 ADC Compare Lower Threshold (ADC廖)

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 mark		Function description
31:12	Reserved	Reserve
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	Twenty four	Twenty three	Twenty two	Twenty one	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	

bit mark		Function description
31:4	Reserved	Reserve
3	CONT_INTF	Continuous conversion complete flag 1: ADC continuous conversion completed 0: ADC continuous conversion is not completed
2	REG_INTF	ADC conversion result comparison interval flag 1: ADC conversion result is in the range of [ADC_LT, ADC_HT) 0: ADC conversion result is outside the range of [ADC_LT, ADC_HT)
1	HHT_INTF	ADC conversion result comparison upper threshold flag 1: ADC conversion result is in the range of [ADC_HT, 4095] 0: ADC conversion result is outside the range of [ADC_HT, 4095]
0	LLT_INTF	ADC conversion result comparison lower threshold flag 1: ADC conversion result is in the range of [0, ADC_LT) 0: ADC conversion result is outside the range of [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	Twenty four	Twenty three	Twenty two	Twenty one	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

bit mark		Function description
31:4	Reserved	Reserve
3	CONT_INTC	Write 0 to clear the continuous conversion complete flag Writing 1 has no effect
2	REG_INTC	Write 0 to clear the ADC conversion result comparison interval flag Writing 1 has no effect
1	HHT_INTC	Write 0 to clear the ADC conversion result comparison upper threshold Writing 1 has no effect
0	LLT_INTC	Write 0 to clear the ADC conversion result comparison lower threshold flag Writing 1 has no effect

## 21.10.18 ADC result (ADC\_result)

offset address 0x068

Reset value 0x00000000

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

bit mark		Function description
31:12	Reserved	Reserve
11:0	result	ADC conversion result

## 22 analog comparators (VC)

### 22.1 Introduction to Analog Voltage Comparator VC

The analog voltage comparator VC is used to compare the magnitude of the two input analog voltages and output high/low according to the comparison result

level. When the voltage of the "+" input terminal is higher than the voltage of the "-" input terminal, the output of the voltage comparator is a high level; when

When the voltage of the "+" input terminal is lower than the voltage of the "-" input terminal, the output of the voltage comparator is a low level. Inside this product

The integrated analog voltage comparator VC has the following characteristics:

ÿ Support voltage comparison function;

ÿ Supports internal 64-step VCC voltage divider (use the voltage divider source voltage needs to be greater than 1.8V)

ÿ Support 8 external input ports and the reference voltage output by the on-chip BGR as the input of the voltage comparator;

ÿ Support three software-configurable interrupt triggering methods: high level trigger/rising edge trigger/falling edge trigger;

ÿ The output of the voltage comparator can be used as the input of the gate control port of Base Timer and LPTimer;

ÿ The output of the voltage comparator can be used as the brake input or capture input of the Advanced Timer;

ÿ Support working in ultra-low power consumption mode, the interrupt output of the voltage comparator can turn the chip from ultra-low power consumption mode

wake;

ÿ Provide software-configurable filter time to enhance the chip's anti-interference ability.

## 22.2 Voltage Comparator Frame Diagram

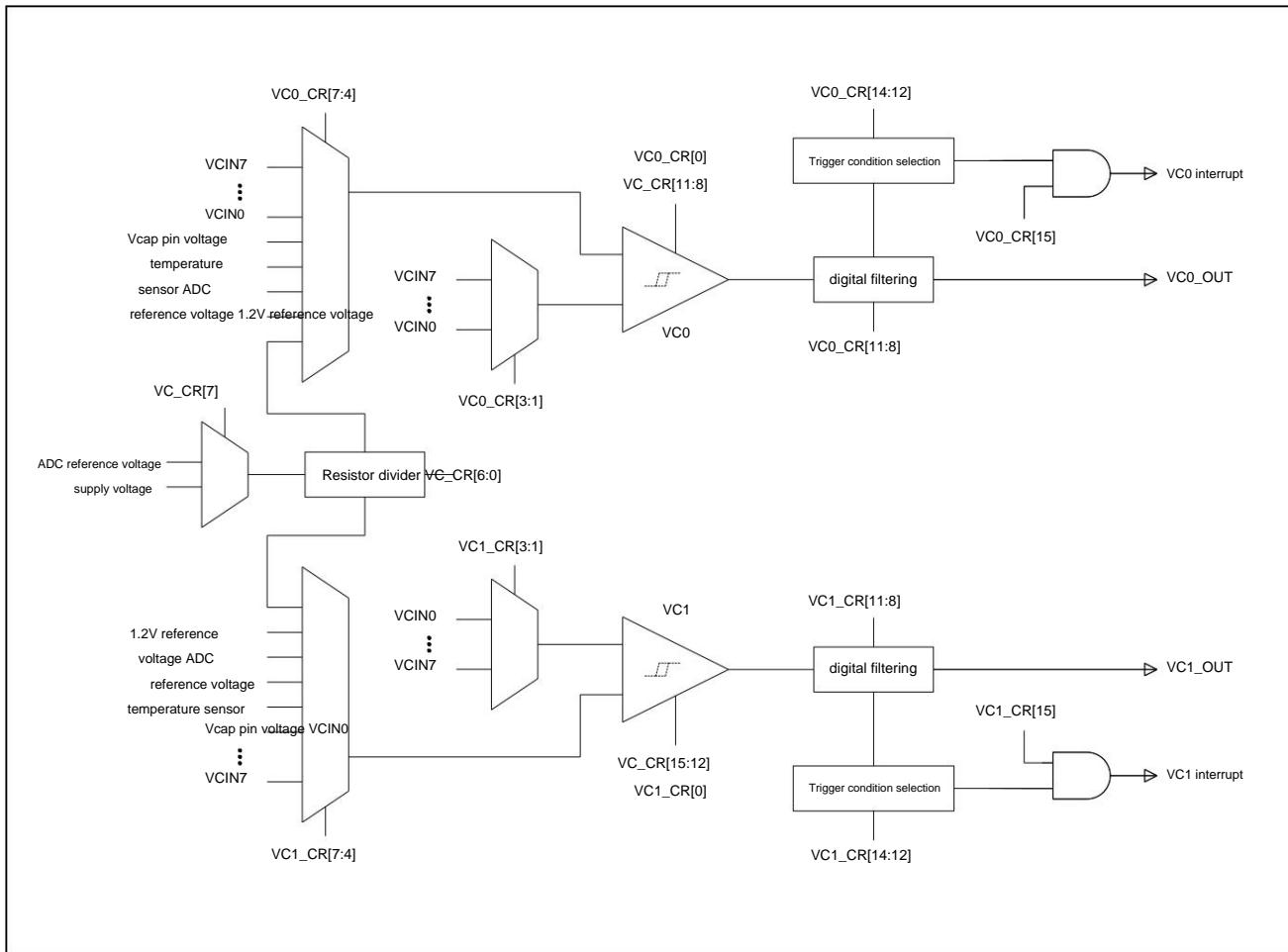


Figure 22-1 VC frame diagram

## 22.3 Setup /Response Time

When using a voltage comparator, the correct result is output from VC enabled or the input voltage across VC changes

The time is determined by the BIAS\_SEL control bit in the VC Control Register (VC\_CR).

If you select the temperature sensor, 1.2V reference voltage, ADC module reference voltage as the input of the comparator, you need to

To open the internal BGR module. The startup time of the internal BGR is about 20us, the voltage comparator needs to wait

The normal output can only be achieved after the internal BGR is stable.

## 22.4 Filter time

In addition to the settling/response time inherent in the voltage comparator, the user can set longer filter times to filter out the system noise, such as high current noise when the motor is stopped.

The digital filtered signal level can be read from the register VC\_IFR.VCx\_Filter; when the GPIO function is configured as

When VCx\_OUT, the digitally filtered signal can be output from GPIO for easy measurement.

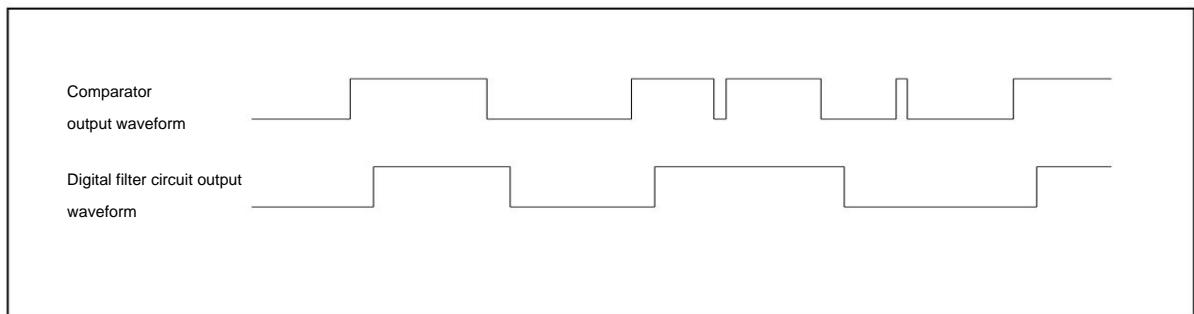


Figure 22-2 VC filter response time

## 22.5 Hysteresis function

The voltage comparator can select the hysteresis function. The figure after the hysteresis function is enabled is as follows

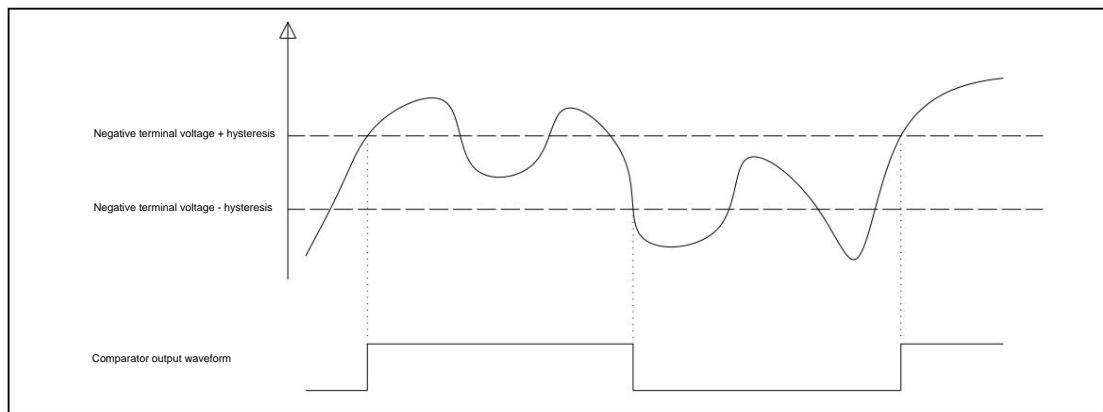


Figure 22-3 VC Hysteresis Function

## 22.6 VC register

base address 0x40002400

register	offset address	describe
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

Table 22-1 VC register

### 22.6.1 VC Configuration Register (VC\_CR)

offset address 0x010

Reset value 0x00000020

31	30	29	28	27	26	25 24	Twenty three	Twenty two	Twenty one	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_BIA_S_SEL	VC_REF2P5_SEL	VC_DI_V_EN	VC_DIV									
R/W	R/W	R/W	R/W	R/W/R/W		R/W									

bit mark		Function description
31:16	Reserved	Reserve
15:14	VC1_HYS_SEL	VC1 Hysteresis Selection: 00: no hysteresis 01: Hysteresis voltage about 10mV 10: Hysteresis voltage about 20mV 11: Hysteresis voltage is about 30mV
13:12	VC1_BIAS_SEL	VC1 power consumption selection (the higher the power consumption, the faster the response speed) 00:300nA 01:1.2uA 10:10uA (requires power supply voltage not less than 2.8V, and needs to manually turn on BGR) 11:20uA (requires power supply voltage not less than 2.8V, and needs to manually turn on 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: Hysteresis voltage about 20mV 11: Hysteresis voltage is about 30mV
9:8	VC0_BIAS_SEL	VC0 power consumption selection (the higher the power consumption, the faster the response speed) 00:300nA 01:1.2uA 10:10uA (requires power supply voltage not less than 2.8V, and needs to manually turn on BGR) 11:20uA (requires power supply voltage not less than 2.8V, and needs to manually turn on BGR) Note: BGR startup time is about 30us
7	VC_REF2P5_SEL	VC_DIV reference voltage Vref selection 0: VCC 1: Reference voltage selected by ADC_CR0.SREF
6	VC_DIV_EN	6-bit DAC enable

		1: enable
5:0	VC_DIV	<p>6-bit DAC configuration</p> <p>000000: 1/64 Vref</p> <p>000001:2/64 Vref</p> <p>000010:3/64 Vref</p> <p>000011:4/64 Vref</p> <p>...</p> <p>111110:63/64 Vref</p> <p>111111: Vref</p>

## 22.6.2 VC0 Configuration Register (VC0\_CR)

offset address 0x0014

Reset value 0x00000000

31	30	29	28	27	26 25	23 22	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 mark		Function description
31:16	Reserved	Reserve
15	IE	VC interrupt enable 1: Enable; 0: Disable
14	level	VC output signal high level triggers INT flag
13	rising	The rising edge of the VC output signal triggers the INT flag
12	Falling	The falling edge of the VC output signal triggers the INT flag
11:9	debounce_time	VC output filter time configuration 111: The filter time is about 28.8ms 110: The filter time is about 7.2ms 101: The filter time is about 1.8ms 100: The filter time is about 450us 011: The filter time is about 112us 010: The filter time is about 28us 001: The filter time is about 14us 000: The filter time is about 7us Note: The filter time configuration is only valid when FLTEN=1.
8	FLTEN	1: Enable VC filter 0: VC no 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: Internal reference 1.2V output voltage 1011: ADC module reference voltage 1100: VCAP pin voltage
3:1	P_SEL	comparator "+" 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	voltage comparator enable 1: enable voltage comparator 0: disable voltage comparator

### 22.6.3 VC1 Configuration Register (VC1\_CR)

offset address 0x0018

Reset value 0x00000000

31	30	29	28	27	26 25		23 22		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			

bit mark		Function description
31:16	Reserved	Reserve
15	IE	VC interrupt enable 1: Enable; 0: Disable
14	level	VC output signal trigger interrupt selection 1: Enable high level trigger INT flag 0: Disable high-level triggering of INT flag
13	rising	VC output signal trigger interrupt selection 1: Enable rising edge to trigger INT flag 0: Prohibit rising edge to trigger INT flag
12	falling	VC output signal trigger interrupt selection 1: Enable falling edge trigger INT flag 0: Disable falling edge trigger INT flag
11:9	debounce_time	VC output filter time configuration 111: The filter time is about 28.8ms 110: The filter time is about 7.2ms 101: The filter time is about 1.8ms 100: The filter time is about 450us 011: The filter time is about 112us 010: The filter time is about 28us 001: The filter time is about 14us 000: The filter time is about 7us Note: The filter time configuration is only valid when FLTEN=1.
8	FLTEN	1: Enable VC filter 0: VC no 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: Internal reference 1.2V output voltage 1011: Reference voltage of ADC module 1100: Voltage and voltage comparator
3:1	P_SEL	"+" input selection of VCAP pin 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 voltage comparator
0	EN	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 T	M6	INV_T imer6	TIM5	INV_T imer5	TIM4	INV_T imer4	PCAE CI	PCAC AP0	INV_ PCA	TM3E CLK	LPTI MG	TIM2 G	TIM1 G	TIMO G	INV_ Timer
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 mark		Function description
31:16	Reserved	Reserve
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	PCAECl	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 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								20	19	18	17	16
Reserved																		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
brake 6	TIM imer6	INV_T imer5	TIM 5	INV_T imer5	TIM 4	INV_T imer4	PCA ECI	PCAC AP1	INV_ PCA	TM3E CLK	LPTI MG	TIM 2G	TIM 1G	TIM 0G	INV_ Timer			
R/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR	WR/WR							

bit mark		Function description
31:16	Reserved	Reserve
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 result 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 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	Twenty four	Twenty three	Twenty two	Twenty one	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	RC	R/W/R/W		

bit mark		Function description
31:4	Reserved	Reserve
3	VC1_Filter	State after VC1 Filter
2	VC0_Filter	State after VC0 Filter
1	VC1_INTF	VC1 interrupt flag, 1 VC1 interrupt occurs; 0 does not interrupt; write 0 to clear the interrupt flag, write 1 is invalid
0	VC0_INTF	VC0 interrupt flag, 1 VC0 interrupt occurs; 0 does not interrupt; write 0 to clear the interrupt flag, write 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 between the monitored voltage and the LVD threshold meets the

When a trigger condition occurs, the LVD will generate an interrupt or reset signal, and the user can perform some urgent tasks based on this signal.

LVDs have the following characteristics:

ÿ 4 monitoring sources, VCC, P03, P23, P25;

ÿ 16-order threshold voltage, flexible and multi-purpose;

ÿ 8 trigger conditions, high level, rising edge, falling edge combination;

ÿ 2 trigger results, reset and interrupt;

ÿ 8th-order filter configuration to prevent false triggering;

ÿ With hysteresis function, strong anti-interference.

### 23.2 LVD Block Diagram

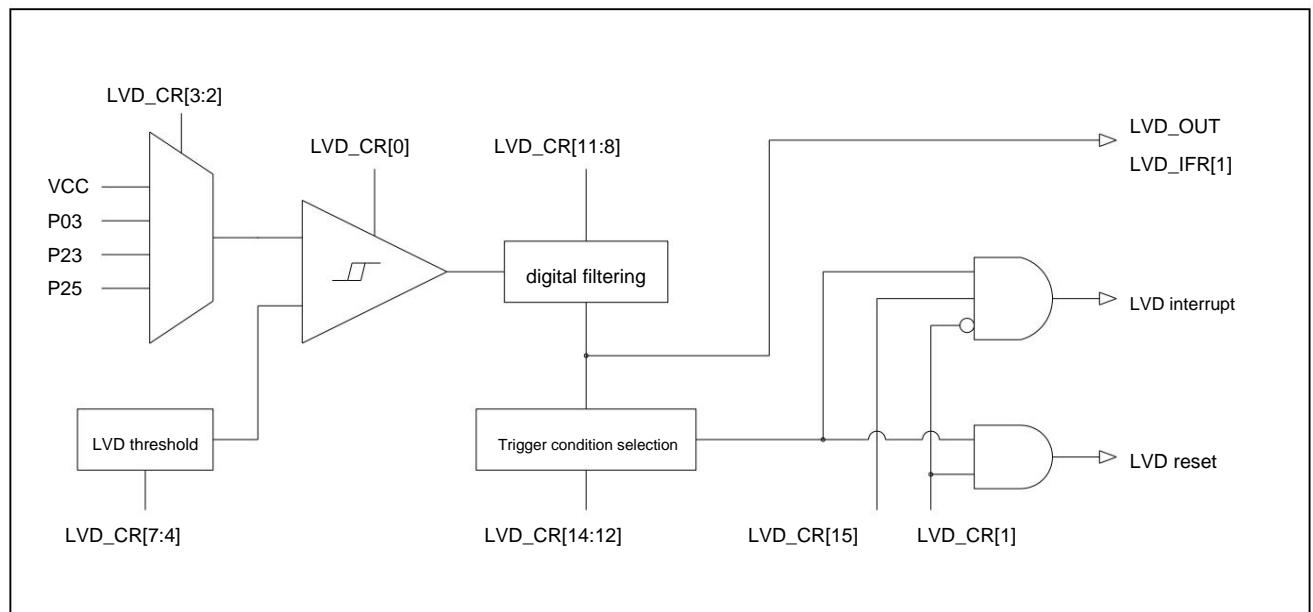


Figure 23-1 LVD block diagram

### 23.3 Digital Filtering

If the working environment of the chip is harsh, the output of the hysteresis comparator will have a noise signal. Enable the digital filter module,

Then the noise signal whose pulse width is less than LVD\_CR.Debounce\_time in the output waveform of the hysteresis comparator can be filtered.

remove. 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 GPIO for easy measurement.

Enable the digital filtering module, the filtering diagram is as follows:

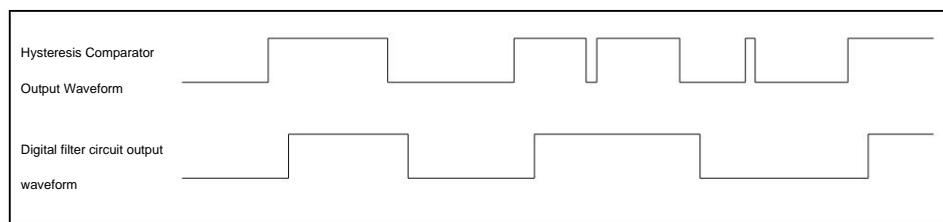


Figure 23-2 LVD filter output

### 23.4 Hysteresis function

The built-in voltage comparator of the LVD has a hysteresis function, and its output signal will wait until the input signal is higher or lower than the threshold voltage.

Inversion occurs after 20mV of voltage. The hysteresis function can enhance the anti-interference ability of the chip, as shown in the figure below.

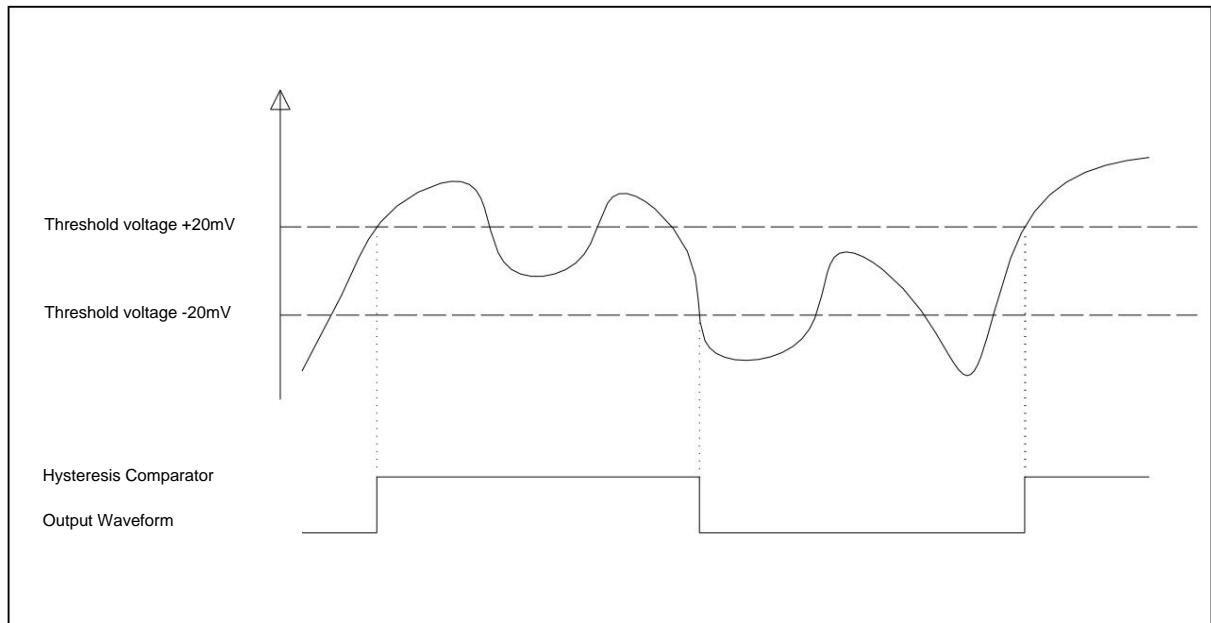


Figure 23-3 LVD Hysteretic Response



## 23.5 Configuration Examples

### 23.5.1 LVD CONFIGURATION FOR 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 and select the voltage source to be monitored.

Step2: Configure LVD\_CR.VTDS and select the threshold voltage of LVD.

Step3: Configure LVD\_CR.Debounce\_time and 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, and select LVD action as reset.

Step7: Set LVD\_CR.LVDEN to 1 to enable LVD.

### 23.5.2 LVD CONFIGURATION FOR VOLTAGE CHANGE INTERRUPT

In this mode, an interrupt is generated when the monitored voltage is above or below the threshold voltage.

The configuration method is as follows:

Step1: Configure LVD\_CR.Source\_sel and select the voltage source to be monitored.

Step2: Configure LVD\_CR.VTDS and select the threshold voltage of LVD.

Step3: Configure LVD\_CR.Debounce\_time and select LVD filter time.

Step4: Configure LVD\_CR.FLTEN to enable LVD filtering.

Step5: Set LVD\_CR.RTEN and LVD\_CR.FTEN to 1, and select level change to trigger LVD action.

Step6: Set LVD\_CR.ACT to 0, and 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

base address 0x40002400

register	offset address	describe
LVD_CR	0x028	LVD Configuration Register
LVD_IFR	0x02C	LVD Interrupt Flag Register

Table 23-1 LVD Register

### 23.6.1 LVD CONFIGURATION REGISTER (LVD\_CR)

offset address 0x028

Reset value 0x00000100

31	30	29	28	27	26	25	Twenty four	Twenty three	Twenty two	Twenty one	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/WR/W	

bit mark		Function description
31:16	Reserved	Reserve
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 above threshold voltage to below threshold voltage) 1: enable; 0: Disabled.
12	FTEN	Falling edge trigger enable (the monitored voltage changes from below threshold voltage to above threshold voltage) 1: enable; 0: Disabled.
11:9	Debounce_time	Digital filter time configuration 111: The filter time is about 28.8ms 110: The filter time is about 7.2ms 101: The filter time is about 1.8ms 100: The filter time is about 450us

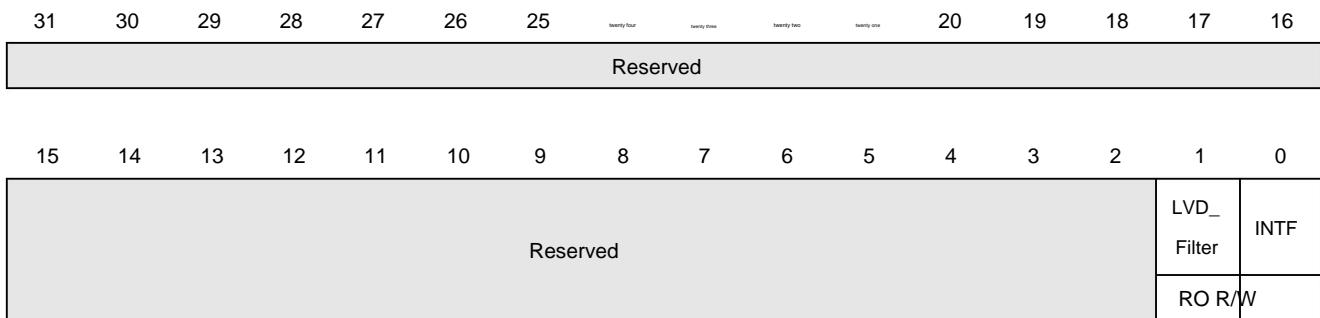


		011: The filter time is about 112us 010: The filter time is about 28us 001: The filter time is about 14us 000: The filter time is about 7us Note: The filter time is only valid when FLTEN is 1. Digital filter
8	FLTEN	function configuration 1: Enable digital filter 0: Disable digital filter LVD monitoring voltage selection
7:4 VTDS		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 power supply
1	ACT	voltage LVD trigger action selection 1: System reset 0: NVIC interrupt LVD control 1:
0	LVDEN	Enable LVD 0: Disable LVD

## 23.6.2 LVD Interrupt Register (LVD\_IFR)

offset address 0x02C

Reset value 0x00000000



bit mark		Function description
31:2	Reserved	Reserve
1	LVD_Filter	Status after LVD Filter
0	INTF	LVD interrupt flag: 1: LVD interrupt occurs; 0: No interrupt occurred;  Writing 0 clears the interrupt flag, writing 1 has no effect.

## 24 analog other registers

base address 0x40002400

register	offset address	describe
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	Twenty four	Twenty three	Twenty two	Twenty one	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
														TS_EN	BGR_EN
														R/WRW	

Bit tag function	function description
31:2	Reserved
1	<p>TS_EN</p> <p>Internal temperature sensor enable 1: Start the internal temperature sensor 0: Disable the internal temperature sensor Note: TS needs about 20us startup time to stabilize.</p>
0	<p>BGR_EN</p> <p>BGR enable control 1: Enable BGR 0: Disable BGR Notice: 1) This register can only be operated when PERI_CLKEN.ADC is 1. 2) After BGR is enabled for 20us, a stable and high-precision reference voltage can be output. After the BGR is stable, it can be used by other modules, so the user operation should add the step of waiting for the BGR to stabilize. 3) BGR must be enabled when ADC is used. 4) When using VC, it is necessary to decide whether to enable BGR according to the configuration of the VC register.</p>

## 25 SWD debugging interface

This series uses ARM Cortex-M0+ core, which has hardware debug module SWD, which supports complex debugging operate. The hardware debug module allows the core to stop when an instruction is fetched (instruction breakpoint) or data is accessed (data breakpoint). kernel halt When it stops, both the internal state of the kernel and the external state of the system can be queried in the IDE. After completing the inquiry, the The core and peripherals can be restored and the program will continue to execute. When the HC32L110 microcontroller is connected to the debugger and starts When debugging, the debugger will use the kernel's hardware debug module for debugging operations.

Notice:

- SWD cannot work in DeepSleep mode, please perform debugging operation in Active and Sleep mode.

### 25.1 SWD debugging additional functions

This product uses an ARM Cortex-M0+ CPU, which contains hardware extensions for advanced debugging The debugging function of this product is the same as that of Cortex-M0+. The debug extension allows the kernel to breakpoint) or stop the kernel when accessing data (data breakpoint). When the kernel is stopped, you can query the internal state of the kernel state and the external state of the system. After the query is complete, the kernel and system are restored and program execution resumes. The debug function is used when the debug host is connected to the MCU and debugged.

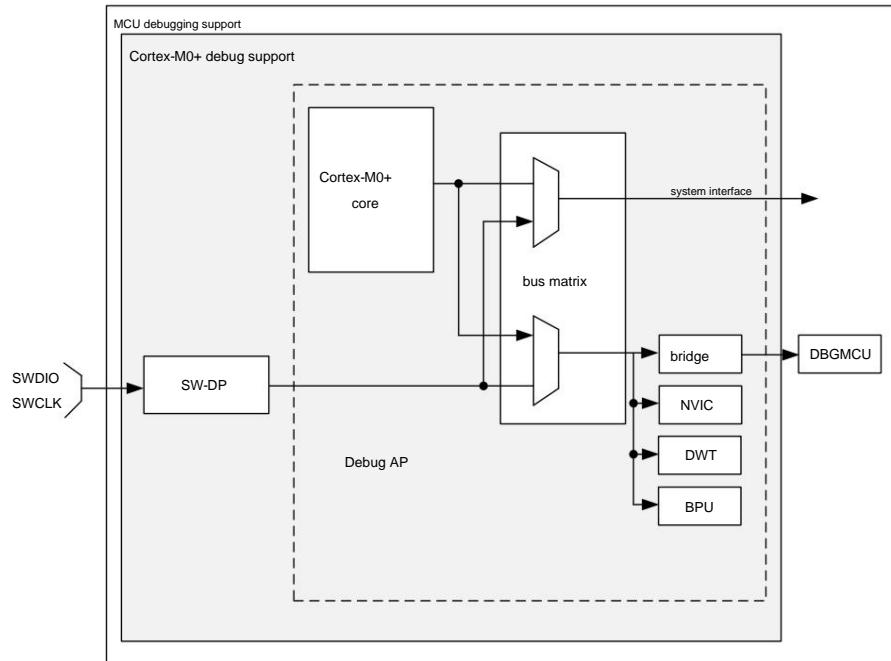


Figure 25-1 Debug Support Block Diagram



The debug capabilities built into the Cortex®-M0+ core are part of the ARM® CoreSight Design Suite.

The ARM® Cortex®-M0+ core provides integrated on-chip debug support. It includes:

- ÿ SW-DP: Serial Line
- ÿ BPU: Breakpoint Unit
- ÿ DWT: Data Watch Point Trigger

Note:

- For details on the debug features supported by the ARM® Cortex®-M0+ core, see Cortex®-M0+ M0+ Technical Reference Manual.

## 25.2 ARM® Reference Documentation

- ÿ Cortex®-M0+ Technical Reference Manual (TRM)

Available from [www.infocenter.arm.com](http://www.infocenter.arm.com).

- ÿ ARM® Debug Interface V5

- ÿ ARM® CoreSight Design Suite Version r1p1 Technical Reference Manual

## 25.3 Debug port pins

### 25.3.1 SWD port pins

The SWD interface of this series needs to use 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 burned

If the [Encryption Chip] option is not enabled in the program, the P27/P31 pins are initialized to be debugged after power-on

dedicated pins used. The user can set the SYSCTRL1.SWD\_USE\_IO register to disable the SWD pin

Debug function, the SWD pin will be released for normal GPIO. The configuration and function of SWD pins are summarized in the following table

shown:

Encryption Chip Option	SWD_USE_IO configuration	P27/P31 function
encryption	0	NA
encryption	1	GPIO
not encrypted	0	SWD
not encrypted	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. Reset of GPIO Control Registers

The state puts the I/O into the equivalent state:

SWDIO: input pull-up

SWCLK: input pull-up

No external resistors are required 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:

- ÿ SWCLK: clock from host to target

- ÿ SWDIO: Bidirectional

Using this protocol, two sets of register banks (DPACC register bank and APACC register bank) can be read and written simultaneously.

device group). When transferring data, the LSB comes first.

For SWDIO bidirectional management, the line must be pulled up on the board (ARM® recommends 100 kÿ).

These pull-up resistors are internally configurable. No external pull-up resistors are required.

Every time the direction of the SWDIO is changed in the protocol, the transition time is inserted and the line is not driven by the host

Nor is it goal driven. By default, this conversion time is one bit time, but can be set by configuring the SWCLK frequency

rate to adjust.

### 25.4.2 SWD protocol sequence

Each sequence consists of three phases:

1. Packet request sent by host (8 bits)
2. The acknowledgment response sent by the target (3 bits)
3. Phase of data transfer sent by host or target (33 bits)

bit name	start	illustrate
0		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 Stop	Unit parity of first few digits
6	Resident	0
7		Not driven by the host. Must be read as 1 by target due to pull-up

See the Cortex®-M0+ TRM for a detailed description of the DPACC and APACC registers.

The packet request is always followed by a transition time (default 1 bit), at which point neither the host nor the target will drive.



bit name		illustrate
0	ACK	001: FAULT 010: WAIT 100: OK

MUST be followed by an ACK response only if a READ transaction occurs or a WAIT or FAULT acknowledgement is received conversion time.

bit name		illustrate
0:31	WDATA or RDATA write or read data	
32 Parity		Single parity of 32 data bits

Conversion time must follow the DATA transfer only when a READ transaction occurs.

#### 25.4.3 SW-DP state machine (reset, idle state, ID code)

The state machine of the SW-DP has an internal ID code for identifying the SW-DP. The code complies with the JEP-106 standard allow. This ID code is the default ARM® code and is set to **0x0BB11477 (equivalent to Cortex®-M0+)**.

Notice:

- The SW-DP state machine is inactive until the target reads this ID code.
- ÿ After power-on reset or after the line is high for more than 50 cycles, the SW-DP state machine is in reset state.  
state.
- ÿ If the line is low for at least two cycles after the reset state, the SW-DP state machine is in the idle state.
- ÿ After the reset state, the state machine must first enter the idle state, and then update the DP-SW ID CODE register  
Perform read access. Otherwise, the target will issue a FAULT confirmation response on another transaction.

For more details on the SW-DP state machine, see *Cortex®-M0+ TRM* and *CoreSight*

design

kit *r1p0TRM*.

#### 25.4.4 DP and AP read/write access

- ÿ Do not delay read access to DP: target response can be sent immediately (if ACK=OK), or delayed  
Send target response (if ACK=WAIT).
- ÿ Delay read access to AP. This means that the access result will be returned on the next transfer. If you want to execute the next  
If this access is not an AP access, the DP-RDBUFF register must be read to get the result.



ÿ The DP-CTRL/STAT register is updated every time an AP read access or RDBUFF read request is made.

READOK flag to know if AP read access was successful.

ÿ SW-DP has write buffer (for DP or AP write), so that even when other operations are not completed,

Write operations are also accepted. If the write buffer is full, the target acknowledges the response with a WAIT. but IDCODE

Except for reads, CTRL/STAT reads, or ABORT writes, which also work when the write buffer is full

been accepted.

ÿ Due to the existence of asynchronous clock domains SWCLK and HCLK, after the write operation (after the parity bit), the

Two additional SWCLK cycles for writes to take effect internally. should be driven low on the line

These cycles are applied when (idle state).

This is especially important when writing to the CTRL/STAT register to initiate a power-up request. else next action

(Operations that are not valid until the kernel is powered on) are executed immediately, which will cause the failure.

## 25.4.5 SW-DP register

These registers can be accessed when APnDP=0:

A[3:2] RW		SELECT register The CTRLSEL bit	register	Notes
00	read		IDCODE	Manufacturer code set to default ARM® for Cortex®-M0+ code. <b>0x0BB11477 (identifies SW-DP)</b>
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-up acknowledgment)
01	read /write	1	WIRE CONTROL	Used to configure the physical serial port protocol (such as the duration of the transition time time)
10	read		READ RESEND	Allows recovery of read data from corrupted debug software transfers, No need to repeat the original AP transmission.
10	Write		SELECT	4-word register window for selecting current access port and activity
11	read /write		READ BUFFER	This read buffer is useful since AP access has been issued (in The result of the read AP request is provided when the next AP transaction is performed). This read buffer captures the data in the AP and appears as

The result of the previous read, without starting 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:

ÿ Shift value A[3:2]

ÿ Current value of DP SELECT register

address	A[3:2]	Value Description
0x0	00	Reserved, the reset value must be maintained.
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-up acknowledgment)
0x8	10	DP SELECT register: 4-word register window for selecting the current access port and activity mouth. – Bit 31:24: APSEL: 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 by the debugger to get the last result after a series of operations fruit

(without requesting a new JTAG-DP operation)

## 25.5 Kernel Debugging

Debug the kernel through the kernel debug registers. Debug access to these registers is done through the debug access port. It is sent by four

Register composition:

Register Description	
DHCSR 32-bit debug stop control and status register	This register provides information about the state of the processor, enables the core to enter a debug stop state and provides processor stepping capabilities.
DCRSR 17-bit Debug Core Register Selector Register:	This register selects the processor registers that need to be read and written.
DCRDR 32-bit debug core register data register:	This register holds the data read and written between the register and the processor selected by the DCRSR (selector) register.
DEMCR 32-bit Debug Exception and Watch Control Register: This	register provides vector capture and debug watch control.

These registers are not reset upon system reset. They can only be reset by a power-on reset. For more details,

See Cortex®-M0+ TRM.

In order to put the kernel into debug halted state immediately after reset, it is necessary to:

- ÿ Enable debug and exception monitoring control register bit 0 (VC\_CORRESET)
- ÿ Enable Debug Stop 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 breakpoints implement PC-based breakpoint functionality.

For more information on the BPU CoreSight identification registers and their addresses and access types, see ARMv6-M

ARM® and ARM® CoreSight Components Technical Reference Manual.



## 25.7 DWT (Data Observation Point)

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 (ie, PC sample registers), and support

Comparator address mask, as described in ARMv6-M ARM®.

### 25.7.2 DWT Program Counter Sample Register

The processor implementing the data watchpoint unit also implements the ARMv6-M optional DWT program counter sample register

controller (DWT\_PCSR). This register allows the debugger to periodically sample the PC without stopping the processor. This provides rough

Brief analysis. See ARMv6-M ARM® for more information.

Cortex®-M0+ DWT\_PCSR records passing condition codes and instructions and instructions that fail condition codes.



## 25.8 MCU Debug Component (DBG)

The MCU debug component helps the debugger provide support for:

- ÿ Low power mode
- ÿ Timer and watchdog clock control during breakpoints

### 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.

The kernel does not allow FCLK or HCLK to be turned off during a debug session. Since they need to be used during debugging connection, so it must remain active. The MCU integrates special methods that allow the user to

Download the debugging software.

### 25.8.2 Debugging support for timers and watchdogs

During a breakpoint, you must choose how the timers and watchdog's counters behave:

- ÿ When a breakpoint occurs, the counter continues to count. For example, this approach is often required when PWM controls a motor Mode.
- ÿ When a breakpoint occurs, the counter stops counting. This method is required for watchdog use.

## 25.9 Debug mode module working state control (DEBUG\_ACTIVE)

Reset value 0x00000FFF (this register setting only works in SWD debug mode)

Offset address: 0x038

31	30 29	28	27	26	25	24 23			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	T	TIM6	TIM5	TIM4	LPTIM	TIM2	TIM1	TIM0	
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		

bit	mark	Function description
31:12	Reserved	Reserve
11	LPTIM	When debugging, Timer3 count function configuration 1: In the SWD debugging interface, suspend the Timer3 counting function 0: In the SWD debugging interface, Timer3 normal counting function
10	Reserved	Reserve
9	RTC	When debugging, the RTC count function is configured 1: In the SWD debugging interface, pause the RTC counting function 0: In the SWD debugging interface, the RTC normal counting function
8	WDT	When debugging, the WDT count function is configured 1: In the SWD debugging interface, suspend the WDT counting function 0: In the SWD debugging interface, the WDT normal counting function
7	PCA	When debugging, the PCA count function is configured 1: In the SWD debugging interface, suspend the PCA counting function 0: In the SWD debugging interface, PCA normal counting function
6	TIM6	When debugging, Timer6 count function configuration 1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debugging interface, the Timer normal counting function
5	TIM5	When debugging, Timer5 count function configuration 1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debugging interface, the Timer normal counting function
4	TIM4	When debugging, Timer4 count function configuration 1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debug interface, Timer normal counting function
3	LPTIM	When debugging, the LpTimer count function is configured 1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debug interface, Timer normal counting function
2	TIM2	When debugging, Timer2 count function configuration 1: In the SWD debugging interface, pause the Timer counting function



		0: In the SWD debugging interface, the Timer normal counting function
1	TIM1	When debugging, Timer1 count function configuration 1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debugging interface, the Timer normal counting function
0	TIM0	When debugging, Timer0 count function configuration 1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debugging interface, the Timer normal counting function



## 26 Devices Electronic Signature

The electronic signature is stored in the system memory area of the flash memory module and can be read by SWD or CPU. it

The included chip identification information is written at the factory, and the user firmware or external device can read the electronic signature to

Automatically match different configurations of HC32Fxxx / HC32Lxxx microcontrollers.

### 26.1 Product Unique Identification (UID) Register (80 -bit)

Typical application scenarios of unique identifiers:

- ÿ Used as serial number
- ÿ UID is used as security when used in conjunction with software cryptographic primitives and protocols before programming the internal Flash
- Keys to improve the security of code in Flash
- ÿ Activate safe bootstrap process, etc.

The 80-bit Unique Device Identifier provides a reference number that is unique to any device and any context. use

The user can never change these bits. The 80-bit unique device identifier can also be in different formats such as single-byte/half-word/word

Read, then concatenate using a custom algorithm.

Base address: 0x0010 0E74

Offset address	description	UID Bit(80bit)							
		7	6		5	4	3	2	1
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,

It is filled with 0x00.

Example: 48433324C3133364B38544100000000 represents the product model HC32L136K8TA.

## 26.3 FLASH Capacity Register

Base address: 0x0010 0C70

31	30	29	28	27	26	25					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 tag	function description	
31:0	FlashSize The capacity of the built-in Flash of the product, in bytes 0x00008000 means the Flash capacity is 32K Byte	



## 26.4 RAM Size Register

Base address: 0x0010 0C74

31	30	29	28	27	26	25					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 tag	function description
31:0	RamSize Product's built-in RAM capacity, 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 tag	function description
15:0	PinCount The number of product pins, in units  0x0020 means the number of product pins is 32

## 27 Appendix A SysTick Timer

### 27.1 Introduction to SysTick Timer

In order for the OS to support multitasking, it needs to perform context switching periodically, which requires hardware resources such as timers.

A source interrupts program execution. When the timer interrupt is generated, the processor will perform OS task scheduling in the exception handling.

At the same time, OS maintenance will also be performed. There is a simple timer called SysTick in the Cortex-M0 processor,

Used to generate periodic interrupt requests.

SysTick is a 24-bit timer and counts down. After the timer counts down to 0, it will be reloaded with a

A programmable value, and a SysTick exception (exception number 15) is generated at the same time, the abnormal event will cause

Execution of SysTick exception handling, which is part of the OS.

For systems that do not require an OS, SysTick timers can also be used for other purposes, such as timing, timing, or

Tasks that require periodic execution provide interrupt sources. The generation of SysTick exceptions is controllable, and if the exception is disabled, it is still

Of course, you can use the SysTick timer in a polling method, such as checking the current count value or polling the overflow flag.

### 27.2 Setting up SysTick

Since the reload value and current value of the SysTick timer are undefined at reset, in order to prevent an abnormal result

If so, the configuration of SysTick needs to follow a certain process:

Step1: Configure SysTick->CTRL. ENABLE to 0, disable SysTick.

Step2: Configure SysTick->CTRL. CLKSOURCE and SYSTICK\_CR[27:25], select SysTick

clock source.

Step3: Configure SysTick->LOAD, and 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 to 1, enable SysTick interrupt.

Step6: Configure SysTick->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:



clock source	SysTick->LOAD	overflow period
RCH 4M	3999	1ms
XTL 32.768K	327	10.01ms

## 27.3 SysTick Register

address	name	CMSIS symbols	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 sign		Function description	Type	reset	value
31:17	Reserved	-	-	-	-
16	COUNTFLAG	Systick timer overflow flag  1: Systick timer underflow occurred. 0: The Systick timer has not overflowed.  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 interrupts	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 sign		Function description	Type	reset	value
31:24	Reserved	-	-	-	-
23:0	RELOAD	SysTick timer reload value	RW	-	

### 27.3.3 SysTick Current Value Register (VAL)

bit sign		Function description	Type	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 Documentation 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. Invalid data is returned when this bit is read.
W1	Only write 1, the hardware will automatically clear 0, write 0 is invalid
R0W1	Software reads this bit as 0 and 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, write 1 is invalid, write 0 to clear
R1W0	Software reads this bit as 1 and 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 a "0" has no effect on the value of this bit.

Res, Reserved Reserved bits, must hold the reset value.

### 28.2 Glossary

This section briefly introduces 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 refers to the ability to reprogram the microcontroller's Flash while the user program is running.

New programming.

ICP (In-Circuit Programming): ICP refers to the ability to use the JTAG protocol,

The SWD protocol or the bootloader program the microcontroller's Flash.

AHB: Advanced High Performance Bus.

APB: Low-speed peripheral bus.

DMA: Direct Memory Access.

TIM: timer



## Version History & Contact

Version revision	revision date	Summary of revisions
Rev1.0	2018/1/23	The first version of the HC32L110 series user manual is released.
Rev1.1	2018/5/4	Version update, revised Flash data, modified part of the description in Chapter 4.
Rev1.2	2018/5/23	Modified the clock switching process and added PCA comparison capture function module settings.
Rev1.3	2018/11/1	Supplemented the description of the function module in Chapter 1, added the description of the pin configuration and function in Chapter 2, supplemented the description of the FLASH operation in Chapter 9, and revised the Chapter 28 Electrical Characteristics Parameters, adding Chapter 29 and Chapter 31.
Rev1.4	2018/11/26	Modify the name: UART2ÿLPUART. Added "Notes" to Sections 2.1 and 2.2.
Rev1.5	2019/2/22	Correct the following data: ÿADC characteristics ÿESD characteristics ÿECFLASH minimum value in memory characteristics ÿIncrease package sizeÿ Add AVCC/AVSS to the pinout diagram.
Rev1.6	2019/7/5	Corrected the following data: ÿ Corrected UID address ÿ Corrected programming mode ÿ Updated QFN pin configuration diagram style.
Rev1.7	2019/12/13	Corrected the following data: ÿ Typical application circuit diagram ÿ LVD block diagram ÿ ADC characteristic unit ÿ Voltage comparator frame diagram ÿ Logo Bit register (UARTx_ISR) ÿXTH and XTl layout and precautions in external clock source characteristics ÿUpdate I2C bus (I2C, Serial Peripheral Interface (SPI), Universal Synchronous Asynchronous Receiver/Transmitter (UART), and Low Power Synchronous Asynchronous Receiver/Transmitter (LPUART) section description.
Rev1.8	2020/1/17	Corrected the following data: ÿAdd CSP16 package ÿDevice electronic signature ÿAppendix A SysTick timer ÿCyclic redundancy check (CRC) ÿSystem control register 1 (SYSCTRL1) adds note items ÿCR register (FLASH_CR) reset value ÿ I2C configuration register (I2C_CR) bit 0.
Rev1.9	2020/3/6	Corrected the following data: ÿAdded attention items in programming mode ÿStep8 of 17.5.1 and 17.5.2.
Rev2.0	2020/4/30	Corrected the following data: ÿincreased VCC/3 accuracy in ADC characteristics ÿCorrected typo in external clock source characteristics ÿInternal clock source characteristics Medium RCL oscillator accuracy.
Rev2.1	2020/5/29	Corrected the following data: ÿAdd Step2 and Step3 in 17.5.1 I2Cx is changed to I2C.
Rev2.2	2020/7/31	Corrected the following data: ÿ Added TIM timer characteristics and communication interface section ÿ EFT level ÿ Internal in general operating conditions AHB/APB clock frequency ÿ Correct the typo The value of VIL.
Rev2.3	2020/9/30	Corrected the following data: ÿ7.8.4 update clerical error ÿClock system description ÿRCH oscillator accuracy in internal clock source characteristicsÿ VIL and VIH of RESETB pin characteristics ÿ increase the SPI characteristics.
Rev2.31	2020/12/31	Deleted product features, pin configuration, package information, etc. (refer to the latest data sheet for related information), and revised the statement.



---

If you have any comments or suggestions in the process of purchasing and using, please feel free to contact us.

Email: [mcu@hdsc.com.cn](mailto:mcu@hdsc.com.cn)

Website: <http://www.hdsc.com.cn/mcu.htm>

Mailing address: 10th Floor, Block A, No. 1867, Zhongke Road, Pudong New Area, Shanghai

Postcode: 201203

---

