XHSC 小华半导体 XIAOHUA SEMICONDUCTOR

HC32L07x Series

32-bit ARM® Cortex®-M0+ Microcontroller

Reference Manual

Rev1.80 August 2026

Statement

★ Xiaohua Semiconductor Co., Ltd. (hereinafter referred to as "XHSC") reserves the right to change, correct, enhance, modify Xiaohua Semiconductor products and/or this document at any time without prior notice. Users can get the latest information before placing orders. XHSC products are sold in accordance with the terms and conditions of sale set forth in the Basic Contract for Purchase and Sales.

★ It is the customer's responsibility to select the appropriate XHSC product for your application and to design, validate and test your application to ensure that your application meets the appropriate standards and any safety, security or other requirements. The customer shall be solely responsible for this.

★ XHSC hereby acknowledges that no intellectual property license has been granted by express or implied means.

★ The resale of XHSC Products shall be invalidated by any warranty commitment of XHSC to such Products if its terms are different from those set forth herein.

★ Any graphics or words marked "®" or "™" are trademarks of XHSC. All other products or services shown on XHSC products are the property of their respective owners.

★ Important Notice: Before using this product, be sure to consult the latest official Errata Sheet to fully understand the product's known corrections and avoid usage risks.

★ Information in this notification replaces and replaces information in previous versions.

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

Contents

Statement **2**
Contents **3**
Table Index **27**
Figure Index **30**
Overview **37**
1 System Structure **38**
1.1 Overview 38
1.2 System address division 39
1.3 Memory and Module Address Assignment 40
2 Operating mode **42**
2.1 Operating mode 44
2.2 Sleep mode 45
2.3 Deep sleep mode 46
3 System Controller (SYSCTRL) **49**
3.1 System Clock Introduction 49
3.1.1 Internal high speed RC clock RCH 50
3.1.2 Internal low speed RC clock RCL 51
3.1.3 External low-speed crystal oscillator clock XTL 51
3.1.4 External high-speed crystal oscillator clock XTH 51
3.1.5 Phase-locked loop clock PLL 51
3.1.6 Internal high speed clock RCH48M 52
3.1.7 Clock start process 52
3.2 System Clock Switching 52
3.2.1 Standard Clock Switching Process 53
3.2.2 RCH switching process between different oscillation frequencies 53
3.2.3 XTL Example from Other Clocks 54
3.2.4 Switching from other clocks to XTH example 54
3.2.5 Switching from other clocks to RCL example 55
3.2.6 Switching from other clocks to RCH example 55
3.2.7 Example of switching between PLL and RCH, the reference clock is RCH 56
3.2.8 Example of switching between PLL and XTH, the reference clock is XTH 57
3.3 Clock Calibration Module 58
3.4 Interrupt Wakeup Control 59
3.4.1 Enable the NVIC corresponding to the module that needs to wake up the processor 60

3.4.2 Method not to execute interrupt service routine after wake-up from deep-sleep mode 60
3.4.3 Using exit hibernation feature 61
3.5 Register 62
3.5.1 System Control Register 0 (SYSCTRL0) 63
3.5.2 System Control Register 1 (SYSCTRL1) 65
3.5.3 System Control Register 2 (SYSCTRL2) 66
3.5.4 RCH Control Register (RCH_CR) 67
3.5.5 XTH Control Register (XTH_CR) 68
3.5.6 RCL Control Register (RCL_CR) 69
3.5.7 XTL Control Register (XTL_CR) 70
3.5.8 PLL Control Register (PLL_CR) 71
3.5.9 Peripheral Module Clock Control Register (PERI_CLKEN0) 72
3.5.10 Peripheral Module Clock Control Register (PERI_CLKEN1) 74
3.5.11 Timer Clock Multiplication Control (OVCK_CR) 75
3.5.12 RC48M Control Register (RC48M_CR) 75
4 Reset Controller (RESET) 76
4.1 Reset Controller Introduction 76
4.1.1 Power-on and power-off reset POR 77
4.1.2 External reset pin reset 77
4.1.3 WDT reset 77
4.1.4 PCA reset 77
4.1.5 LVD low voltage reset 77
4.1.6 Cortex-M0+ SYSRESETREQ reset 77
4.1.7 Cortex-M0+ LOCKUP reset 77
4.2 Register 78
4.2.1 Reset flag register (RESET_FLAG) 78
4.2.2 Peripheral Module Reset Control Register (PERI_RESET0) 79
4.2.3 Peripheral Module Reset Control Register (PERI_RESET1) 81
5 Interrupt Controller (NVIC) 82
5.1 Overview 82
5.2 Interrupt priority 82
5.3 Interrupt digraph 83
5.4 Interrupt Input and Suspend Behavior 84
5.5 Interrupt wait 86
5.6 Interrupt source 86
5.7 Interrupt structure diagram 88
5.8 Register 90

5.8.1 Interrupt Enable Setting Register (NVIC_ISER) 91
5.8.2 Interrupt Enable Clear Register (NVIC_ICER) 91
5.8.3 Interrupt Pending Status Set Register (NVIC_ISPR) 92
5.8.4 Interrupt Pending Status Clear Register (NVIC_ICPR) 92
5.8.5 Interrupt Priority Register (NVIC_IPR0) 93
5.8.6 Interrupt Priority Register (NVIC_IPR1) 93
5.8.7 Interrupt Priority Register (NVIC_IPR2) 94
5.8.8 Interrupt Priority Register (NVIC_IPR3) 94
5.8.9 Interrupt Priority Register (NVIC_IPR4) 95
5.8.10 Interrupt Priority Register (NVIC_IPR5) 95
5.8.11 Interrupt Priority Register (NVIC_IPR6) 96
5.8.12 Interrupt Priority Register (NVIC_IPR7) 96
5.8.13 Interrupt Mask Special Register (PRIMASK) 97
5.9 Basic operation of the software 98
5.9.1 External interrupt enable 98
5.9.2 NVIC interrupt enable and clear enable 98
5.9.3 NVIC interrupt pending and clear pending 98
5.9.4 NVIC interrupt priority 98
5.9.5 NVIC interrupt mask 99
6 Port Controller (GPIO) 100
6.1 Introduction to Port Controllers 100
6.2 Port Controller Main Features 100
6.3 Port Controller Functional Description 101
6.3.1 Port configuration function 101
6.3.2 Port write 103
6.3.3 port read 104
6.3.4 Port digital multiplexing function 105
6.3.5 Port interrupt function 107
6.4 Port configuration operation flow 108
6.4.1 Port multiplexing is configured as an analog port operation flow 108
6.4.2 Port multiplexing configured as a digital universal port operation flow 108
6.4.3 Port multiplexing configured as a digital function port operation flow 108
6.4.4 Port multiplexing is configured as a debug test port operation process 108
6.4.5 Port multiplexing configured as infrared output signal operation process 108
6.4.6 Port high level interrupt operation flow 108
6.4.7 Port low level interrupt operation flow 109
6.4.8 Port rising edge interrupt operation flow 109
6.4.9 Port falling edge interrupt operation flow 109

6.4.10 Port pull-up enable configuration operation flow 109
6.4.11 Port pull-down enable configuration operation flow 109
6.4.12 Port Enhancement Driver Configuration Operation Flow 109
6.4.13 Port open-drain output configuration operation flow 110
6.4.14 Port bit setting operation flow 110
6.4.15 Port bit clearing operation flow 110
6.4.16 Operation flow of port bit setting and clearing 110
6.5 Port Controller Register Description 111
6.5.1 Port general register 117
6.5.2 Port Auxiliary Register 126
6.5.3 Port PA Function Selection Register 133
6.5.4 Port PB Function selection register 141
6.5.5 PORT PC FUNCTION SELECT REGISTER 149
6.5.6 Port PD Function selection register 157
6.5.7 Port PE Function Selection Register 165
6.5.8 Port PF Function Selection Register 173
7 I2C bus (I2C) 179
7.1 Introduction 179
7.2 Main characteristics 179
7.3 protocol description 179
7.3.1 Data transmission on the I2C bus 179
7.3.2 Acknowledgment on the I2C bus 181
7.3.3 Arbitration on the I2C bus 181
7.4 Functional description 183
7.4.1 Serial clock generator 184
7.4.2 Input filter 184
7.4.3 Address comparator 184
7.4.4 response flag 185
7.4.5 interrupt generator 185
7.4.6 Operating mode 185
7.4.7 Status code expression 190
7.5 Programming example 192
7.5.1 Master sending example 192
7.5.2 Master receive example 193
7.5.3 Slave receiving example 194
7.5.4 Slave sending example 194
7.6 Register description 196
7.6.1 I2C Baud Rate Counter Enable Register (I2Cx_TMRUN) 197

7.6.2 I2C Baud Rate Counter Configuration Register (I2Cx_TM) 197
7.6.3 I2C Configuration Register (I2Cx_CR) 198
7.6.4 I2C Data Register (I2Cx_DATA) 199
7.6.5 I2C Address Register (I2Cx_ADDR) 199
7.6.6 I2C Status Register (I2Cx_STAT) 200
8 Serial Peripheral Interface (SPI) 201
8.1 Introduction to SPI 201
8.2 Main Features of SPI 201
8.3 SPI Functional Description 202
8.3.1 SPI master mode 202
8.3.2 SPI slave mode 203
8.3.3 SPI data frame format 204
8.3.4 SPI Status Flags and Interrupts 206
8.3.5 SPI multi-machine system configuration instructions 206
8.3.6 SPI pin configuration description 208
8.4 SPI programming example 209
8.4.1 SPI master sending example 209
8.4.2 SPI master receive example 209
8.4.3 SPI slave sending example 210
8.4.4 SPI Slave Receive Example 211
8.5 SPI register description 212
8.5.1 SPI Configuration Register (SPIx_CR) 213
8.5.2 SPI Chip Select Configuration Register (SPIx_SSN) 214
8.5.3 SPI Status Register (SPIx_STAT) 215
8.5.4 SPI Data Register (SPIx_DATA) 216
8.5.5 SPI Configuration Register 2 (SPIx_CR2) 217
8.5.6 SPI Interrupt Clear Register 2 (SPIx_ICLR) 218
9 Clock Trim Module (CLKTRIM) 219
9.1 Introduction to CLKTRIM 219
9.2 CLKTRIM main features 219
9.3 CLKTRIM function description 219
9.3.1 CLKTRIM calibration mode 219
9.3.2 CLKTRIM monitoring mode 222
9.4 CLKTRIM register description 225
9.4.1 Configuration Register (CLKTRIM_CR) 226
9.4.2 Reference counter initial value configuration register (CLKTRIM_REFCON) 227
9.4.3 Reference Counter Value Register (CLKTRIM_REFCNT) 227
9.4.4 Calibration Counter Value Register (CLKTRIM_CALCNT) 228

9.4.5 Interrupt Flag Register (CLKTRIM_IFR) 228
9.4.6 Interrupt flag clear register (CLKTRIM_ICLR) 229
9.4.7 Calibration Counter Overflow Value Configuration Register (CLKTRIM_CALCON) 229
10 Hardware divider module (HDIV) 230
10.1 Introduction to HDIV 230
10.2 HDIV main characteristics 230
10.3 HDIV function description 231
10.3.1 HDIV operation process 231
10.4 HDIV register description 232
10.4.1 Dividend register (HDIV_DIVIDEND) 233
10.4.2 Divisor register (HDIV_DIVISOR) 233
10.4.3 Quotient register (HDIV_QUOTIENT) 234
10.4.4 Remainder register (HDIV_REMAINDER) 234
10.4.5 Sign register (HDIV_SIGN) 235
10.4.6 Status Register (HDIV_STAT) 235
11 FLASH controller (FLASH) 236
11.1 Overview 236
11.2 FLASH capacity division 236
11.3 Functional description 236
11.3.1 Sector Erase (Sector Erase) 237
11.3.2 Full Chip Erase (Chip Erase) 237
11.3.3 Write operation (Program) 238
11.3.4 Read operation (Read) 240
11.4 Erase time 241
11.5 Read wait period 242
11.6 Erase and write protection 243
11.6.1 Erase and write protection bit 243
11.6.2 PC address erase and write protection 243
11.7 Register write protection 243
11.8 Register 244
11.8.1 TNVS parameter register (FLASH_TNVS) 245
11.8.2 TPGS parameter register (FLASH_TPGS) 245
11.8.3 TPROG parameter register (FLASH_TPROG) 246
11.8.4 TSERASE register (FLASH_TSERASE) 246
11.8.5 TMERASE parameter register (FLASH_TMERASE) 247
11.8.6 TPRCV parameter register (FLASH_TPRCV) 247
11.8.7 TSRCV parameter register (FLASH_TSRCV) 248
11.8.8 TMRCV parameter register (FLASH_TMRCV) 248

11.8.9 CR register (FLASH_CR) 249
11.8.10 IFR register (FLASH_IFR) 249
11.8.11 ICLR register (FLASH_ICLR) 250
11.8.12 BYPASS register (FLASH_BYPASS) 250
11.8.13 SLOCK0 register (FLASH_SLOCK0) 251
11.8.14 SLOCK1 register (FLASH_SLOCK1) 251
12 RAM controller (RAM) 252
12.1 Overview 252
12.2 Functional description 252
12.2.1 RAM address range 252
12.2.2 Read and write bit width 252
12.2.3 Parity 252
12.3 Register 253
12.3.1 Control Register (RAM_CR) 254
12.3.2 Parity Error Address Register (RAM_ERRADDR) 254
12.3.3 Error Interrupt Flag Register (RAM_IFR) 255
12.3.4 Error Interrupt Flag Clear Register (RAM_ICLR) 255
13 DMA controller (DMAC) 256
13.1 Introduction to DMAC 256
13.2 DMAC main characteristics 256
13.3 Functional block diagram 257
13.4 Functional description 257
13.4.1 DMAC transfer modes 257
13.4.2 DMA software block transfer mode 258
13.4.3 DMA software Burst transfer mode 259
13.4.4 DMA hardware block transfer mode 260
13.4.5 DMA hardware Burst transfer mode 261
13.4.6 DMA transfer paused 262
13.4.7 DMA other configuration 263
13.4.8 DMA interrupt 263
13.5 Register 265
13.5.1 DMAC_CONF 266
13.5.2 DMAC_CONFA0、DMAC_CONFA1 267
13.5.3 DMAC_CONFB0、DMAC_CONFB1 269
13.5.4 DMAC_SRCADR0、DMAC_SRCADR1 271
13.5.5 DMAC_DSTADR0、DMAC_DSTADR1 271

14 General purpose timer (TIM0/1/2/3) 272
14.1 Introduction to General-Purpose Timers 272
14.1.1 Basic Features (TIM0/1/2) 272
14.1.2 Basic Features (TIM3) 273
14.2 Timer function description 274
14.2.1 Timer clock 274
14.2.2 Timer counter 274
14.2.3 Timer Prescaler 274
14.2.4 Mode 0 count timer function 275
14.2.5 Mode 1 pulse width measurement PWC 278
14.2.6 Mode 2/3 compare capture mode 282
14.2.7 Mode 2/3 Slave Mode 304
14.2.8 Quadrature code counting function 305
14.2.9 Timer triggers ADC 307
14.2.10 Brake control 308
14.2.11 Timer interconnection 309
14.2.12 GATE input interconnection 309
14.2.13 ETR input interconnection 310
14.2.14 CHx capture input interconnect 310
14.2.15 DMA 311
14.3 Timer register description 313
14.4 Mode 0 Timer Register Description 314
14.4.1 16 -bit Mode Reload Register (TIMx_ARR) 314
14.4.2 16 -bit Mode Count Register (TIMx_CNT) 314
14.4.3 32 -bit mode count register (TIMx_CNT32) 315
14.4.4 Control Register (TIMx_M0CR) 316
14.4.5 Interrupt Flag Register (TIMx_IFR) 317
14.4.6 Interrupt Flag Clear Register (TIMx_ICLR) 317
14.4.7 Dead Time Register (TIMx_DTR) 318
14.5 Pulse Width Measurement PWC Register Description 319
14.5.1 16 -bit Mode Count Register (TIMx_CNT) 319
14.5.2 Control Register (TIMx_M1CR) 320
14.5.3 Interrupt Flag Register (TIMx_IFR) 321
14.5.4 Interrupt Flag Clear Register (TIMx_ICLR) 321
14.5.5 Master-Slave Mode Control Register (TIMx_MSCR) 322
14.5.6 Output Control Filtering (TIMx_FLTR) 323
14.5.7 Control Register (TIMx_CR0) 324
14.5.8 Compare Capture Register (TIMx_CCR0A) 324

14.6 Mode 2,3 register description 325
14.6.1 16 -bit Mode Reload Register (TIMx_ARR) 325
14.6.2 16 -bit Mode Count Register (TIMx_CNT) 325
14.6.3 Control Register(TIMx_M23CR) 326
14.6.4 Interrupt Flag Register (TIMx_IFR) 328
14.6.5 Interrupt Flag Clear Register (TIMx_ICLR) 329
14.6.6 Master-Slave Mode Control Register (TIMx_MSCR) 330
14.6.7 Output Control / Input Filtering (TIMx_FLTR) 331
14.6.8 ADC Trigger Control Register (TIMx_ADTR) 333
14.6.9 Channel 0 Control Register (TIMx_CRCH0) 334
14.6.10 Channel 1/2 Control Registers (TIM3_CRCH1/2) (TIM3 present only) 336
14.6.11 Dead Time Register (TIMx_DTR) 338
14.6.12 Repeat cycle setting value (TIMx_RCR) 339
14.6.13 Channel 0 Compare Capture Register (TIMx_CCR0A/B) 339
14.6.14 Channel 1/2 compare capture register (TIM3_CCR1/2 A/B) (TIM3 exists only) 340
15 Low Power Timer (LPTIM) 341
15.1 Introduction to LPTimer 341
15.2 LPTimer Function Description 341
15.2.1 Counting function 342
15.2.2 Timing function 343
15.3 LPTimer Interconnect 343
15.3.1 GATE interconnection 343
15.3.2 EXT Interconnection 343
15.3.3 Toggle Output Interconnects 344
15.4 LPTimer register description 345
15.4.1 Counter Count Value Register (LPTIMx_CNT) 346
15.4.2 Reload Register (LPTIMx_ARR) 346
15.4.3 Control Register (LPTIMx_CR) 347
15.4.4 Interrupt Flag Register (LPTIMx_IFR) 348
15.4.5 Interrupt Flag Clear Register (LPTIMx_ICLR) 348
16 Programmable Count Array (PCA) 349
16.1 Introduction to PCA 349
16.2 PCA function description 349
16.2.1 PCA Timer / Counter 350
16.2.2 PCA capture function 352
16.2.3 PCA comparison function 353
16.3 Interconnection and control of PCA module and other modules 359
16.4 PCA register description 360

16.4.1 Control Register (PCA_CCON) 361
16.4.2 Mode Register (PCA_CMOD) 362
16.4.3 Count register (PCA_CNT) 363
16.4.4 Interrupt Clear Register (PCA_ICLR) 363
16.4.5 Compare Capture Mode Register (PCA_CCAPM0~4) 364
16.4.6 Compare the upper 8 bits of the capture data register (PCA_CCAP0~4H) 365
16.4.7 Compare the lower 8 bits of the capture data register (PCA_CCAP0~4L) 365
16.4.8 Compare capture 16 -bit register (PCA_CCAP0~4) 366
16.4.9 Compare High Speed Output Flag Register (PCA_CCAPO) 366
16.4.10 Period Register (PCA_CARR) 367
16.4.11 Enhanced PWM Control (PCA_EPWM) 367
17 Advanced Timer (TIM4/5/6) 368
17.1 Introduction to Advanced Timer 368
17.2 Advanced Timer Function Description 370
17.2.1 Basic action 370
17.2.2 Timer selection 372
17.2.3 Counting direction 373
17.2.4 Digital filtering 373
17.2.5 Software synchronization 374
17.2.6 Hardware synchronization 376
17.2.7 Cache function 378
17.2.8 General PWM output 379
17.2.9 Orthogonal coding count 385
17.2.10 Periodic interval response 390
17.2.11 Protection mechanism 391
17.2.12 Interrupt Description 391
17.2.13 DMA 392
17.2.14 brake protection 393
17.2.15 interconnection 394
17.3 Register description 397
17.3.1 Common Count Reference Register (TIMx_CNTER) 399
17.3.2 Universal Period Reference Register (TIMx_PERAR) 399
17.3.3 General purpose period buffer register (TIMx_PERBR) 400
17.3.4 Universal Compare Base Registers (TIMx_GCMAR-GCMDR) 400
17.3.5 Dedicated Compare Reference Registers (TIMx_SCMAR-SCMBR) 401
17.3.6 DEAD TIME REFERENCE REGISTER (TIMx_DTUAR-DTDAR) 401
17.3.7 General Control Register (TIMx_GCONR) 402
17.3.8 Interrupt Control Register (TIMx_ICONR) 403

17.3.9 Port Control Register (TIMx_PCONR) 404
17.3.10 Buffer Control Register (TIMx_BCONR) 406
17.3.11 Dead Time Control Register (TIMx_DCONR) 407
17.3.12 Filter Control Register (TIMx_FCONR) 408
17.3.13 Valid Period Register (TIMx_VPERR) 409
17.3.14 Status Flag Register (TIMx_STFLR) 410
17.3.15 Hardware Start Event Select Register (TIMx_HSTAR) 411
17.3.16 Hardware Stop Event Select Register (TIMx_HSTPR) 413
17.3.17 Hardware Clear Event Select Register (TIMx_HCELR) 415
17.3.18 Hardware Capture A Event Select Register (TIMx_HCPAR) 417
17.3.19 Hardware Capture B Event Select Register (TIMx_HCPBR) 418
17.3.20 Hardware Increment Event Select Register (TIMx_HCUPR) 419
17.3.21 Hardware Decrement Event Select Register (TIMx_HCDOR) 421
17.3.22 Software Synchronization Start Register (TIMx_SSTAR) 423
17.3.23 Software Sync Stop Register (TIMx_SSTPR) 423
17.3.24 Software Synchronous Clear Register (TIMx_SCLRR) 424
17.3.25 Interrupt Flag Register (TIMx_IFR) 425
17.3.26 Interrupt Flag Clear Register (TIMx_ICLR) 426
17.3.27 Spread spectrum and interrupt trigger selection (TIMx_CR) 427
17.3.28 AOS Selection Control Register (TIMx_AOSSR) 428
17.3.29 AOS Selection Control Register Flag Clear (TIMx_AOSCL) 429
17.3.30 Port Brake Control Register (TIMx_PTBKS) 430
17.3.31 Port Trigger Control Register (TIMx_TTRIG) 431
17.3.32 AOS Trigger Control Register (TIMx_ITRIG) 432
17.3.33 Port Brake Polarity Control Register (TIMx_PTBKP) 433
18 Real Time Clock (RTC) 434
18.1 Introduction to Real Time Clocks 434
18.2 Real Time Clock Functional Description 435
18.2.1 Power-on setting 435
18.2.2 RTC count start setting 436
18.2.3 System low power mode switching 436
18.2.4 Read count register 436
18.2.5 Write count register 437
18.2.6 Alarm clock setting 437
18.2.7 1Hz output 437
18.2.8 Clock Error Compensation 438
18.3 RTC Interrupt 440
18.3.1 RTC alarm interrupt 440

18.3.2 RTC cycle interrupt....440
18.4 RTC register description....441
18.4.1 Control Register 0 (RTC_CR0)....442
18.4.2 Control Register 1 (RTC_CR1)....443
18.4.3 Second Count Register (RTC_SEC)....444
18.4.4 Minute Count Register (RTC_MIN)....444
18.4.5 Hour count register (RTC_HOUR)....445
18.4.6 Day Count Register (RTC_DAY)....447
18.4.7 Week Count Register (RTC_WEEK)....448
18.4.8 Month Count Register (RTC_MON)....449
18.4.9 Year Count Register (RTC_YEAR)....449
18.4.10 Seconds Alarm Register (RTC_ALMSEC)....450
18.4.11 Minute Alarm Register (RTC_ALMMIN)....450
18.4.12 Hour Alarm register (RTC_ALMHOUR)....451
18.4.13 Week Alarm Register (RTC_ALMWEEK)....451
18.4.14 Clock Error Compensation Register (RTC_COMPEN)....452
19 Watchdog Timer (WDT)....454
19.1 Introduction to WDT....454
19.2 WDT Functional Description....454
19.2.1 Interrupt generated after WDT overflow....454
19.2.2 Reset after WDT overflow....455
19.3 WDT register description....456
19.3.1 WDT Clear Control Register (WDT_RST)....457
19.3.2 WDT_CON register....457
20 Pulse Counter (PCNT)....458
20.1 Introduction to Pulse Counters....458
20.2 Pulse Counter Main Features....458
20.3 Pulse Counter Functional Description....459
20.3.1 Overall block diagram....459
20.3.2 Signal Description....459
20.3.3 Counting mode....459
20.3.4 Pulse Width Filtering....463
20.3.5 Time-out....463
20.3.6 Pulse input selection....463
20.3.7 Debug observation output....464
20.3.8 Automatic wake-up timer in low-power mode....464
20.4 PCNT register description....466
20.4.1 PCNT start register (PCNT_RUN)....467

20.4.2 PCNT Control Register (PCNT_CTRL) 468
20.4.3 PCNT filter control register (PCNT_FLT) 469
20.4.4 PCNT Timeout Control Register (PCNT_TOCR) 469
20.4.5 PCNT Command Register (PCNT_CMD) 470
20.4.6 PCNT status register 1 (PCNT_SR1) 470
20.4.7 PCNT count register (PCNT_CNT) 471
20.4.8 PCNT count overflow register (PCNT_TOP) 471
20.4.9 PCNT count overflow buffer register (PCNT_BUF) 472
20.4.10 PCNT Interrupt Flag Register (PCNT_IFR) 473
20.4.11 PCNT Interrupt Clear Register (PCNT_ICR) 474
20.4.12 PCNT Interrupt Enable Register (PCNT_IEN) 474
20.4.13 PCNT Synchronization Status Register (PCNT_SR2) 475
20.4.14 PCNT debug observation output selection register (PCNT_DBG) 475
21 General Synchronous Asynchronous Transceiver (UART) 476
21.1 Introduction 476
21.2 Main characteristics 476
21.3 Functional description 477
21.3.1 Operating mode 477
21.3.2 Baud rate generation 481
21.3.3 Frame error detection 485
21.3.4 Multi-machine communication 485
21.3.5 DMAC hardware handshake 486
21.3.6 Hardware flow control 486
21.3.7 Transceiver buffer 487
21.4 Register 489
21.4.1 Data Register (UARTx_SBUF) 490
21.4.2 Control Register (UARTx_SCON) 491
21.4.3 Address Register (UARTx_SADDR) 492
21.4.4 Address Mask Register (UARTx_SADEN) 492
21.4.5 Flag Register (UARTx_ISR) 493
21.4.6 Flag Clear Register (UARTx_ICR) 493
21.4.7 Baud Rate Register (UARTx_SCNT) 494
22 Low Power Synchronous Asynchronous Transceiver (LPUART) 495
22.1 Introduction 495
22.2 Main characteristics 496
22.3 Functional description 496
22.3.1 Configuration Clock and Transmit Clock 496
22.3.2 Operating mode 497

22.3.3 Baud rate generation 501
22.3.4 Frame error detection 505
22.3.5 Multi-machine communication 505
22.3.6 DMAC hardware handshake 506
22.3.7 Hardware flow control 506
22.3.8 Transceiver buffer 507
22.4 Register 509
22.4.1 Data Register (LPUARTx_SBUF) 510
22.4.2 Control register (LPUARTx_SCON) 511
22.4.3 Address Register (LPUARTx_SADDR) 512
22.4.4 Address Mask Register (LPUARTx_SADEN) 512
22.4.5 Flag Bit Register (LPUARTx_ISR) 513
22.4.6 Flag Clear Register (LPUARTx_ICR) 513
22.4.7 Baud Rate Register (LPUARTx_SCNT) 514
23 Cyclic redundancy check (CRC) 515
23.1 Overview 515
23.2 Main characteristics 515
23.3 Functional description 515
23.3.1 Operating mode 515
23.3.2 Encoding 515
23.3.3 Write bit width 515
23.4 Programming example 516
23.4.1 CRC-16 encoding mode 516
23.4.2 CRC-16 check mode 516
23.4.3 CRC-32 encoding mode 516
23.4.4 CRC-32 check mode 516
23.5 Register description 518
23.5.1 Register list 518
23.5.2 Control register (CRC _ CR) 519
23.5.3 Results register (CRC_RESULT) 519
23.5.4 Data register (CRC_DATA) 520
24 True random number generator (TRNG) 521
24.1 Overview 521
24.2 Functional block diagram 521
24.3 Functional description 521
24.4 Register 522
24.4.1 Control Register (TRNG_CR) 523
24.4.2 Mode Register (TRNG_MODE) 523

24.4.3 Data Register 0 (TRNG_DATA0) 524
24.4.4 Data Register 1 (TRNG_DATA1) 524
24.5 Basic operation of the software 525
24.5.1 The operation process of generating 64bits true random number (the first time after power-on) 525
24.5.2 Generating 64bits true random number (not generated for the first time after power-on) 527
25 Advanced Encryption Standard Module (AES) 528
25.1 Function definition 528
25.1.1 AES algorithm 528
25.1.2 AES module function description 530
25.2 Module register description 531
25.2.1 Control Register (AES_CR) 532
25.2.2 Data register (AES_Data) 533
25.2.3 Key register (AES_Key) 534
25.3 Exception mechanism 536
25.4 Operating instructions for this module 536
25.4.1 IP operations 536
25.4.2 Encryption operation process 536
25.4.3 Decryption operation process 536
25.4.4 Data example 537
25.5 Runtime instructions 539
26 Liquid Crystal Controller (LCD) 540
26.1 Introduction to LCD 540
26.2 LCD main characteristics 540
26.3 LCD Block Diagram 541
26.4 LCD drive waveform 541
26.4.1 Static drive waveform 542
26.4.2 1/2Duty 1/2Bias driving waveform 543
26.4.3 1/2Duty 1/3Bias drive waveform 544
26.4.4 1/3Duty 1/2Bias driving waveform 545
26.4.5 1/3Duty 1/3Bias drive waveform 546
26.4.6 1/4Duty 1/2Bias driving waveform 547
26.4.7 1/4Duty 1/3Bias drive waveform 548
26.4.8 1/6Duty 1/3Bias drive waveform 549
26.4.9 1/8Duty 1/3Bias drive waveform 550
26.5 LCD Bias Generation Circuit 551
26.5.1 Internal resistance mode 551

26.5.2 External Capacitor Mode 552
26.5.3 External Resistor Mode 552
26.6 DMA 553
26.7 Interrupt 553
26.8 LCD display mode 553
26.8.1 LCD display mode 1 (MODE = 1) 554
26.8.2 LCD display mode 0 (MODE = 0) 555
26.9 LCD register 556
26.9.1 Configuration Register 0 (LCD_CR0) 557
26.9.2 Configuration Register 1 (LCD_CR1) 559
26.9.3 Interrupt Clear Register (LCD_INTCLR) 560
26.9.4 Output Configuration Register 0 (LCD_POEN0) 560
26.9.5 Output Configuration Register 1 (LCD_POEN1) 561
26.9.6 LCD_RAM0~7 563
26.9.7 LCD_RAM8~F 563
27 Crystal-less USB Clock Calibrator (CTS) 564
27.1 48M clock calibration 564
27.1.1 System characteristics 564
27.1.2 Block diagram 565
27.1.3 Reference source synchronization input 565
27.1.4 Frequency Error Measurement 566
27.1.5 Frequency error trimming 567
27.1.6 CTS initialization and configuration 568
27.2 Other Oscillation Calibration 568
27.2.1 System characteristics 568
27.2.2 Block diagram 569
27.2.3 Calibration Instructions 569
27.3 Timer function description 570
27.3.1 Block diagram 570
27.3.2 Timer Clock Selection 570
27.3.3 Timer 571
27.4 CTS register 572
27.4.1 CTS Control Register (CTS_CR) 573
27.4.2 CTS Configuration Register (CTS_CFGR) 574
27.4.3 CTS Interrupt and Status Register (CTS_ISR) 575
27.4.4 CTS Interrupt Status Clear Register (CTS_ICR) 576
28 Audio interface (I2S) 577
28.1 Functional description 577

28.1.1 Pin multiplexing 578
28.1.2 I2S full duplex 578
28.2 I2S audio protocol description 579
28.2.1 I2S Philips Standard 580
28.2.2 MSB Alignment Standard 582
28.2.3 LSB Alignment Standard 584
28.2.4 PCM standard 586
28.3 I2S clock generator 588
28.4 I2S working mode 592
28.4.1 I2S host mode 592
28.4.2 I2S slave mode 594
28.5 I2S Status Flags 596
28.5.1 Busy flag (BSY) 596
28.5.2 Send buffer is empty (TXE_L TXE_R) 596
28.5.3 Receive buffer is not empty (RXNE_L RXNE_R) 597
28.5.4 Underflow flag (UDR) 597
28.5.5 Overflow flag (OVR) 597
28.5.6 Framing Error Flag (FRE) 597
28.6 I2S Interrupt DMA 598
28.7 I2S register 599
28.7.1 I2Sx Control Register (I2Sx_CR) 600
28.7.2 I2Sx Status Register (I2Sx_SR) 601
28.7.3 I2Sx Configuration Register (I2Sx_CFGR) 602
28.7.4 I2Sx Prescaler Register (I2Sx_PR) 603
28.7.5 I2Sx Interrupt Status Clear Register (I2Sx_ICR) 604
28.7.6 I2Sx Data Register L (I2Sx_DRL) 604
28.7.7 I2Sx Data Register R (I2Sx_DRR) 605
29 USB 2.0 Full Speed Module (USBFS) 606
29.1 Introduction to USBFS 606
29.2 Main Features of USBFS 606
29.2.1 General features 606
29.2.2 Device Mode Properties 606
29.3 USBFS System Block Diagram 607
29.4 USBFS pin description 607
29.5 USBFS function description 608
29.5.1 USBFS clock and working mode 608
29.5.2 USBFS device capabilities 608
29.5.3 USBFS power control 612

29.5.4 USBFS data FIFO 612
29.5.5 USBFS device FIFO architecture 613
29.5.6 USBFS FIFO RAM allocation 614
29.5.7 USBFS system performance 614
29.5.8 USBFS interrupt and Events 615
29.6 USBFS programming model 615
29.6.1 USBFS module initialization 615
29.6.2 USBFS device initialization 616
29.6.3 USBFS device soft disconnect operation 616
29.6.4 Endpoint initialization on USB reset 617
29.6.5 USB enumeration completes 617
29.6.6 Endpoint initialization when a SetAddress command is received 618
29.6.7 Endpoint initialization when a SetConfiguration/SetInterface command is received 618
29.6.8 Endpoint activation (Active) 618
29.6.9 Endpoint inactive (Deactive) 619
29.6.10 Disable (Disable) output endpoint 619
29.6.11 Pause (Stall) asynchronous output endpoint 619
29.6.12 Global output endpoint NAK mode 620
29.6.13 Output endpoint transmission stopped 620
29.6.14 Configure the interrupt input endpoint as a periodic or aperiodic endpoint (shared FIFO mode) 620
29.6.15 Global aperiodic input endpoint NAK mode (shared FIFO mode) 621
29.6.16 NAK occurred 621
29.6.17 Disable non-periodic input endpoints (Shared FIFO mode) 621
29.6.18 Disable cycle input endpoint (Shared FIFO mode) 622
29.6.19 Aperiodic input data transfer timeout (shared FIFO mode) 622
29.6.20 Non-synchronous input endpoint pause (shared FIFO mode) 622
29.6.21 Input endpoint transfer stop (shared FIFO mode) 623
29.6.22 Endpoint mismatch (shared FIFO mode) 623
29.6.23 Control transfer 623
29.6.24 Output data transfer 629
29.6.25 No threshold for acyclic bulk-in data transfer 630
29.6.26 Thresholds for acyclic bulk-in data transfers 630
29.6.27 Asynchronous output data transfer without threshold 630
29.6.28 Asynchronous output data transfers with threshold 630
29.6.29 Synchronous output data transfer with threshold 630
29.6.30 Synchronous output data transfer not completed 631

29.6.31 Periodic input data transfer without threshold 631
29.6.32 Periodic input data transfers have thresholds 632
29.7 Register description 633
29.7.1 USBFS global registers 635
29.7.2 USBFS Device Mode Register 648
29.7.3 USBFS Clock Gating Control Register 676
30 Controller Area Network (CAN) 677
30.1 Introduction 677
30.2 CAN System Block Diagram 678
30.3 Pin description 678
30.4 Functional description 678
30.4.1 Baud rate setting 678
30.4.2 Send buffer 680
30.4.3 Receive buffer 680
30.4.4 Receive Filter Register Set 681
30.4.5 Data transmission 682
30.4.6 Single data transmission 682
30.4.7 Cancel data sending 682
30.4.8 Data reception 683
30.4.9 Error handling 683
30.4.10 Node down 684
30.4.11 Lost Arbitration Position Capture 684
30.4.12 Loop mode 684
30.4.13 Silent mode 685
30.4.14 Software reset function 686
30.4.15 Upward compatible with CAN-FD function 687
30.4.16 Time-triggered TTCAN 687
30.4.17 Interrupt 690
30.5 Register description 691
30.5.1 CAN receive BUF register (CAN_RBUF) 693
30.5.2 CAN transmit BUF register (CAN_TBUF) 695
30.5.3 CAN Configuration and Status Register (CAN_CFG_STAT) 697
30.5.4 CAN Command Register (CAN_TCMD) 698
30.5.5 CAN Transmit Control Register (CAN_TCTRL) 700
30.5.6 CAN Receive Control Register (CAN_RCTRL) 701
30.5.7 CAN Receive and Transmit Interrupt Enable Register (CAN_RTIE) 702
30.5.8 CAN Receive and Transmit Interrupt Status Register (CAN_RTIF) 703
30.5.9 CAN Error Interrupt Enable and Flags Register (CAN_ERRINT) 704

30.5.10 CAN Bit Timing Register (CAN_BT) 705
30.5.11 CAN Error and Arbitration Lost Capture Register (CAN_EALCAP) 706
30.5.12 CAN Warning Limit Register (CAN_LIMIT) 706
30.5.13 CAN Receive Error Counter Register (CAN_RECNT) 707
30.5.14 CAN Transmit Error Counter Register (CAN_TECNT) 707
30.5.15 CAN Filter Group Control Register (CAN_ACFCTRL) 708
30.5.16 CAN Filter Group Enable Register (CAN_ACFEN) 709
30.5.17 CAN filter group code and mask registers (CAN_ACF) 710
30.5.18 TTCAN TB slot pointer register (CAN_TBSLOT) 711
30.5.19 TTCAN Time Triggered Configuration Register (CAN_TTCFG) 712
30.5.20 TTCAN Reference Message Register (CAN_REF_MSG) 713
30.5.21 TTCAN Trigger Configuration Register (CAN_TRG_CFG) 714
30.5.22 TTCAN Trigger Time Register (CAN_TRG_TRIG) 715
30.5.23 TTCAN Trigger Watchdog Time Register (TRG_WTRIG) 715
30.6 Precautions for use 716
30.6.1 CAN bus anti-jamming measures 716
30.6.2 CAN Controller Noise Constraints 716
31 Analog-to-digital converter (ADC) 717
31.1 Module Introduction 717
31.2 ADC block diagram 717
31.3 Conversion Timing and Conversion Speed 718
31.4 Single conversion mode 718
31.5 Scan conversion mode 720
31.5.1 Sequential Scan Conversion Mode 720
31.5.2 In-queue scan conversion mode 723
31.5.3 Scan conversion triggers DMA read 726
31.6 Continuous Conversion Accumulation Mode 726
31.7 ADC conversion external trigger source 729
31.8 ADC conversion result comparison 730
31.9 ADC Interrupt 730
31.10 Measure ambient temperature using a temperature sensor 731
31.11 ADC module registers 733
31.11.1 ADC Basic Configuration Register 0 (ADC_CR0) 735
31.11.2 ADC Basic Configuration Register 1 (ADC_CR1) 737
31.11.3 ADC Sequential Scan Conversion Channel Configuration Register 0 (ADC_SQR0) . 738
31.11.4 ADC Sequential Scan Conversion Channel Configuration Register 1 (ADC_SQR1) . 739
31.11.5 ADC Sequential Scan Conversion Channel Configuration Register 2 (ADC_SQR2) . 740
31.11.6 ADC jumping scan conversion channel configuration register (ADC_JQR) 741

31.11.7 ADC sequential scan conversion channel x conversion result (ADC_SqrResult0 - 15) 741
31.11.8 ADC jump scan conversion channel x conversion result (ADC_JqrResult0 - 3) 742
31.11.9 ADC conversion result (ADC_Result) 742
31.11.10 Accumulated value of ADC conversion result (ADC_ResultAcc) 743
31.11.11 ADC Compare Upper Threshold (ADC_HT) 743
31.11.12 ADC Compare Lower Threshold (ADC_LT) 744
31.11.13 ADC Interrupt Flag Register (ADC_IFR) 745
31.11.14 ADC Interrupt Clear Register (ADC_ICR) 746
31.11.15 ADC single conversion or sequential scan conversion external interrupt trigger source configuration register (ADC_ExtTrigger0) 747
31.11.16 ADC jump scan conversion external interrupt trigger source configuration register (ADC_ExtTrigger1) 749
31.11.17 ADC Single Conversion Start Control Register (ADC_SglStart) 751
31.11.18 ADC Sequential Scan Conversion Start Control Register (ADC_SqrStart) 751
31.11.19 ADC jump scan conversion start control register (ADC_JqrStart) 752
31.11.20 ADC has been converting start stop control register (ADC_allStart) 752
32 Digital to Analog Converter (DAC) 753
32.1 DAC Characteristic 753
32.2 Functional description 754
32.2.1 Block diagram 754
32.2.2 DAC output buffer enable 755
32.2.3 DAC channel enable 755
32.2.4 DAC output voltage 755
32.2.5 DAC trigger selection 755
32.2.6 Single mode function description 756
32.2.7 Dual mode function description 758
32.2.8 Noise generation 763
32.2.9 Triangle wave generation 763
32.2.10 DMA request 764
32.3 DAC register 766
32.3.1 DAC Control Register (DAC_CR0) 767
32.3.2 DAC Software Trigger Register (DAC_SWTRIGR) 769
32.3.3 DAC0 12- bit Right Justified Data Holding Register (DAC_DHR12R0) 769
32.3.4 DAC0 12- bit Left Justified Data Holding Register (DAC_DHR12L0) 770
32.3.5 DAC0 8- bit Right Justified Data Holding Register (DAC_DHR8R0) 770
32.3.6 DAC1 12- bit right-justified data holding register (DAC_DHR12R1) 771
32.3.7 DAC1 12- bit Left Justified Data Holding Register (DAC_DHR12L1) 771

32.3.8 DAC1 8- bit right-justified data holding register (DAC_DHR8R1) 772
32.3.9 Dual DAC 12- bit Right Justified Data Holding Register (DAC_DHR12RD) 772
32.3.10 Dual DAC 12 -bit Left Justified Data Holding Register (DAC_DHR12LD) 773
32.3.11 Dual DAC 8 -bit Right Justified Data Holding Register (DAC_DHR8RD) 773
32.3.12 DAC0 Data Output Register (DAC_DOR0) 774
32.3.13 DAC1 Data Output Register (DAC_DOR1) 774
32.3.14 DAC Status Register (DAC_SR) 775
32.3.15 DAC Trigger Select Register (DAC_ETRS) 775
33 Analog Comparator (VC) 776
33.1 Introduction to Analog Voltage Comparator VC 776
33.2 Voltage Comparator Block Diagram 777
33.3 Setup / Response Time 778
33.4 Filter time 778
33.5 Hysteresis function 778
33.6 VC register 779
33.6.1 VC Configuration Register (VC_CR) 780
33.6.2 VC0 Configuration Register (VC0_CR) 781
33.6.3 VC1 Configuration Register (VC1_CR) 783
33.6.4 VC0 Output Configuration Register (VC0_OUT_CFG) 785
33.6.5 VC1 Output Configuration Register (VC1_OUT_CFG) 786
33.6.6 VC Interrupt Register (VC_IFR) 787
33.6.7 VC2 Configuration Register (VC2_CR) 788
33.6.8 VC2 Output Configuration Register (VC2_OUT_CFG) 790
34 Low Voltage Detector (LVD) 791
34.1 Introduction to LVD 791
34.2 LVD block diagram 791
34.3 Hysteresis function 792
34.4 Digital filtering 792
34.5 Configuration example 793
34.5.1 LVD configured as low voltage reset 793
34.5.2 LVD configured as voltage change interrupt 793
34.6 LVD register 794
34.6.1 LVD configuration register (LVD_CR) 795
34.6.2 LVD Interrupt Register (LVD_IFR) 797
35 Operational Amplifier (OPA) 798
35.1 OPA features 798
35.2 OPA Functional Description 799
35.2.1 OPA0/1/2 (Generic OPA) 799

35.2.2 OPA3/4 --- (can be multiplexed as DAC buffer*2) 800
35.3 OPA register 801
35.3.1 OPA Output Enable Register (OPA_CR0) 802
35.3.2 OPA Control Register (OPA_CR1) 803
35.3.3 OPA configuration register (OPA_CR2) 804
36 Simulate other registers 805
36.1 BGR Configuration Register (BGR_CR) 805
37 SWD debug interface 806
37.1 SWD debugging additional functions 806
37.2 ARM® Reference Documentation 807
37.3 Debug port pins 807
37.3.1 SWD port pin 807
37.3.2 SW-DP pin assignment 807
37.3.3 Internal pull-up on SWD pin 808
37.4 SWD port 808
37.4.1 Introduction to SWD Protocol 808
37.4.2 SWD protocol sequence 808
37.4.3 SW-DP state machine (reset, idle state, ID code) 809
37.4.4 DP and AP read / write access 809
37.4.5 SW-DP register 810
37.4.6 SW-AP Register 810
37.5 Kernel debugging 811
37.6 BPU (Breakpoint Unit) 811
37.6.1 BPU function 811
37.7 DWT (Data Watchpoint) 812
37.7.1 DWT function 812
37.7.2 DWT Program Counter Sample Register 812
37.8 MCU debug group (DBG) 812
37.8.1 Debug support for low power modes 812
37.8.2 Debug support for timers, watchdog 812
37.9 Debug mode module working state control (DEBUG_ACTIVE) 813
38 Device electronic signature 814
38.1 Product Unique Identifier (UID) Register (80 bits) 814
38.2 Product Model Register 814
38.3 FLASH capacity register 815
38.4 RAM capacity register 815
38.5 Pin Count Register 815
39 Appendix A SysTick Timer 816

39.1 Introduction to SysTick Timers 816
39.2 Set SysTick 816
39.3 SysTick register 817
39.3.1 SysTick Control and Status Register (CTRL) 817
39.3.2 SysTick reload register (LOAD) 817
39.3.3 SysTick Current Value Register (VAL) 817
39.3.4 SysTick Calibration Value Register (CALIB) 818
40 Appendix B Document Conventions 819
40.1 List of register-related abbreviations 819
40.2 Glossary 819
Version Revision History 820

Table Index

Table 1-1 Address Division Table 40
Table 2-1 Runnable module diagram in run mode 44
Table 2-2 Runnable Module Diagram in Sleep Mode 46
Table 2-3 Runnable module diagram in deep sleep mode 47
Table 5-1 Cortex-M0+ processor interrupt overview 82
Table 5-2 Correspondence between external interrupt and NVIC interrupt input 86
Table 6-1 Port State Truth Table 103
Table 6-2 Port multiplexing table 107
Table 7-1 I2C clock signal baud rate 184
Table 7-2 I2C status code expression 191
Table 7-3 Register List 196
Table 8-1 SPI pin configuration description table 208
Table 8-2 SPI register list 212
Table 8-3 Master Mode Baud Rate Selection 213
Table 9-1 Register List 225
Table 10-1 Register List 232
Table 11-1 FLASH capacity division 236
Table 11-2 FLASH erasing time parameters at different frequencies 241
Table 12-1 RAM Address Mapping 252
Table 12-2 Register base address 253
Table 14-1 Timer register list 313
Table 15-1 LPTimer register list 345
Table 16-1 PCA comparison capture function module setup 358
Table 16-2 PCA register list 360
Table 17-1 Basic Features of Advanced Timer 368
Table 17-2 Advanced Timer port list 368
Table 17-3 AOS source selection 395
Table 17-4 Port trigger selection 396
Table 17-5 Advanced Timer register list 397
Table 18-1 Basic characteristics of RTC 434
Table 18-2 RTC register list 441
Table 19-1 WDT register list 456
Table 20-1 Register List 466
Table 21-1 Mode0/1/2/3 Data Structure 477
Table 21-2 B8 Data Meaning 477
Table 21-3 PCLK=4MHz baud rate calculation table 482

Table 21-4 PCLK=8MHz baud rate calculation table 482
Table 21-5 PCLK=16MHz baud rate calculation table 483
Table 21-6 PCLK=24MHz baud rate calculation table 483
Table 21-7 PCLK=32MHz baud rate calculation table 484
Table 21-8 PCLK=48MHz baud rate calculation table 484
Table 22-1 Mode0/1/2/3 Data Structure 497
Table 22-2 B8 Data Meaning 497
Table 22-3 SCL is 4MHz baud rate calculation table 502
Table 22-4 SCLK is 8MHz baud rate calculation table 502
Table 22-5 SCLK is 16MHz baud rate calculation table 503
Table 22-6 SCLK is 24MHz baud rate calculation table 503
Table 22-7 SCLK is 32MHz baud rate calculation table 504
Table 22-8 SCLK is 48MHz baud rate calculation table 504
Table 25-1 Register List 531
Table 25-2 128 -bit operational register example 537
Table 25-3 192 -bit operational register example 538
Table 25-4 192 -bit operational register example 539
Table 25-5 AES encryption and decryption running time 539
Table 26-1 LCD register 556
Table 27-1 CTS register 572
Table 28-1 I2Sx Register 599
Table 29-1 USBFS Pin Descriptions 607
Table 29-2 USBFS interrupt/Event Table 615
Table 29-3 List of USBFS registers 633
Table 30-1 CAN Pin Descriptions 678
Table 30-2 CAN bit time setting rules 679
Table 30-3 Software reset range table 686
Table 30-4 CAN Interrupt Table 690
Table 30-5 List of CAN registers 691
Table 30-6 CAN register BYTE/HALFWORD/WORD access arrangement table 692
Table 30-7 Standard Format CAN Receiving Mailbox Format 693
Table 30-8 Extended Format CAN Receive Mailbox Format 694
Table 30-9 Standard format CAN sending mailbox format 695
Table 30-10 Extended Format CAN Send Mailbox Format 696
Table 31-1 ADC register 733
Table 32-1 DAC register 766
Table 33-1 VC register 779
Table 34-1 LVD register 794

Table 35-1 OPA register 801

Figure Index

Figure 1-1 System Architecture Diagram 38
Figure 1-2 Schematic Diagram of Address Area division 39
Figure 2-1 Control Mode Block Diagram 42
Figure 3-1 Clock Control Module Block Diagram 50
Figure 3-2 Schematic diagram of crystal oscillator clock startup 52
Figure 3-3 Schematic diagram of clock switching 58
Figure 3-4 Clock Calibration Schematic 59
Figure 4-1 Reset Source Schematic 76
Figure 5-1 Only the upper two bits of the priority register are used 82
Figure 5-2 Interrupt Vector Table 83
Figure 5-3 Interrupt active and pending states 84
Figure 5-4 Interrupt pending status is cleared and then reasserted 85
Figure 5-5 When the interrupt exits, if the interrupt request remains high, it will cause the interrupt processing to be executed again 85
Figure 5-6 Interrupt pending interrupts generated during interrupt processing can also be acknowledged 85
Figure 5-7 Interrupt Structure Diagram 88
Figure 6-1 Port Circuit Diagram 101
Figure 6-2 AHB/FASTIO bus port changes with system clock 103
Figure 6-3 Read port pin data synchronization diagram 104
Figure 7-1 I2C transfer protocol 180
Figure 7-2 START and STOP conditions 180
Figure 7-3 Bit transfer on the I2C bus 181
Figure 7-4 Acknowledgment signal on I2C bus 181
Figure 7-5 Arbitration on the I2C bus 182
Figure 7-6 I2C function block diagram 183
Figure 7-7 Data synchronization diagram of master sending mode 185
Figure 7-8 I2C master sending state diagram 186
Figure 7-9 Data synchronization diagram of main receiving mode 187
Figure 7-10 I2C master receiving state diagram 187
Figure 7-11 Slave Receive Mode Data Synchronization Diagram 188
Figure 7-12 Slave receiving state diagram 188
Figure 7-13 Slave mode data synchronization diagram 189
Figure 7-14 I2C Slave Transmit State Diagram 189
Figure 7-15 I2C General Call State Diagram 190
Figure 8-1 Slave receiving diagram 203

Figure 8-2 Slave sending diagram 204
Figure 8-3 Host Mode Frame Format 204
Figure 8-4 Data frame format when slave CPHA is 0 205
Figure 8-5 Data frame format when slave CHPA is 1 205
Figure 8-6 Schematic diagram of SPI multi-master/multi-slave system 207
Figure 9-1 Schematic diagram of clock source selection 220
Figure 9-2 Clock Calibration Module Hardware Schematic 221
Figure 9-3 Clock Calibration Waveform Diagram 222
Figure 9-4 CLKTRIM clock selection 223
Figure 9-5 Schematic diagram of clock monitoring waveform 224
Figure 13-1 Functional block diagram 257
Figure 14-1 TIM0/1/2 block diagram 273
Figure 14-2 TIM3 block diagram 274
Figure 14-3 Free Counting Block Diagram 275
Figure 14-4 Heavy Duty Counting Waveform 276
Figure 14-5 16 -bit heavy load counting waveform 276
Figure 14-6 32 -bit free-counting waveform 276
Figure 14-7 Free Count Timing Diagram 277
Figure 14-8 Timing diagram of heavy load counting (prescaler is set to 2) 277
Figure 14-9 PWC Measurement Block Diagram 279
Figure 14-10 High level pulse width measurement 279
Figure 14-11 Falling edge to falling edge period measurement 280
Figure 14-12 Rising edge to rising edge period measurement 280
Figure 14-13 Rising edge-to-rising edge period measurement one-shot mode 281
Figure 14-14 Counter Block Diagram 282
Figure 14-15 Count Up Without Prescaler 283
Figure 14-16 Up Counting with Prescaler 284
Figure 14-17 Down counting without prescaler 284
Figure 14-18 Down counting with prescaler 285
Figure 14-19 Up and down counting with prescaler 285
Figure 14-20 Edge-Aligned Timer Waveform (DIR =1) 285
Figure 14-21 Edge-Aligned Timer Waveform (DIR =0) 286
Figure 14-22 Center Aligned Counter Waveform 286
Figure 14-23 Repeat counter generates update timing 287
Figure 14-24 Buffer enable in triangle wave mode 287
Figure 14-25 Buffer invalidation in triangle wave mode 288
Figure 14-26 Up count buffer enable in sawtooth mode 288
Figure 14-27 The up count buffer is invalid in sawtooth mode 289

Figure 14-28 Count buffer enable in sawtooth mode....289
Figure 14-29 The counter buffer is invalid in sawtooth mode....290
Figure 14-30 Count compare buffer enable in sawtooth mode....290
Figure 14-31 OCREF output block diagram....291
Figure 14-32 Sawtooth counting single point comparison OCREF output waveform(OCMx=111)....292
Figure 14-33 Triangular wave counting single point comparison OCREF output waveform (OCMx=111)....292
Figure 14-34 Sawtooth wave counting double-point comparison OCREF output (OCMx=111)....293
Figure 14-35 Triangular wave counting double-point comparison OCREF output (OCMx=111)....293
Figure 14-36 Independent PWM Output Block Diagram....294
Figure 14-37 PWM output waveform when CCPx=0....294
Figure 14-38 PWM output waveform when CCPx=1....294
Figure 14-39 Complementary PWM Output Block Diagram....295
Figure 14-40 Complementary PWM output waveform....295
Figure 14-41 Complementary PWM output waveform....296
Figure 14-42 Dead time....296
Figure 14-43 Single pulse counting in triangle wave mode....297
Figure 14-44 Counting Single Pulse Mode on Ramp Waveform....298
Figure 14-45 Counting single pulse mode under sawtooth waveform....298
Figure 14-46 Interrupt diagram....299
Figure 14-47 Capturing Functional Block Diagram....300
Figure 14-48 Capture Timing Diagram....300
Figure 14-49 CHA port selection....300
Figure 14-50 CHB port selection....301
Figure 14-51 Slave Mode Schematic....304
Figure 14-52 Gating function....304
Figure 14-53 Trigger and reset functions....305
Figure 14-54 Code count....307
Figure 14-55 ADC trigger....308
Figure 15-1 LPTimer block diagram....341
Figure 15-2 LPTimer Mode 1....342
Figure 15-3 LPTimer Mode 2....342
Figure 16-1 Overall block diagram of PCA....349
Figure 16-2 PCA Counter Block Diagram....351
Figure 16-3 PCA Capture Functional Block Diagram....353
Figure 16-4 PCA Comparison Functional Block Diagram....354
Figure 16-5 PCA 16 -bit PWM functional block diagram....355
Figure 16-6 PCA WDT Functional Block Diagram....356

Figure 16-7 PCA PWM Functional Block Diagram 357
Figure 16-8 PCA PWM Output Waveform 358
Figure 17-1 Advanced Timer Block Diagram 369
Figure 17-2 Sawtooth Waveform (Counting Up) 370
Figure 17-3 Triangle Waveform 370
Figure 17-4 Compare output action 371
Figure 17-5 Capturing Input Actions 372
Figure 17-6 Filter function of the capture input port 374
Figure 17-7 Software Synchronization Action 375
Figure 17-8 Hardware Synchronous Action 377
Figure 17-9 Single-buffer mode comparison output timing 378
Figure 17-10 PWM Spread Spectrum Output Schematic 380
Figure 17-11 CHA output PWM wave 380
Figure 17-12 Software setting GCMBR Complementary PWM wave output in triangular wave A mode 381
Figure 17-13 Triangular wave B mode hardware setting GCMBR Complementary PWM wave output (symmetrical dead zone) 382
Figure 17-14 6-phase PWM wave 383
Figure 17-15 Three-phase complementary PWM wave output with dead time in triangular wave A mode 384
Figure 17-16 Basic counting action in position mode 385
Figure 17-17 Phase difference counting action setting in position mode (1 time) 386
Figure 17-18 Phase difference counting action setting in position mode (2 time) 386
Figure 17-19 Phase difference counting action setting in position mode (4 time) 387
Figure 17-20 Direction counting action in position mode 387
Figure 17-21 Phase Z counting action in revolution mode 388
Figure 17-22 Position counter output counting action in revolution mode 388
Figure 17-23 Z-phase counting and position counter output mixed counting action in revolution mode 389
Figure 17-24 Revolution counting mode - Mixed counting Z-phase shielding action example 1 389
Figure 17-25 Revolution counting mode - Mixed counting Z-phase shielding action example 2 390
Figure 17-26 Cycle Interval Valid Request Signal Action 391
Figure 17-27 Schematic diagram of port braking and software braking 393
Figure 17-28 Output same high and same low brake schematic diagram 394
Figure 17-29 VC brake control diagram 394
Figure 17-30 Timer4/5/6 interrupt selection 395
Figure 18-1 RTC Block Diagram 435
Figure 19-1 Overall block diagram of WDT 454

Figure 20-1 Overall Block Diagram 459
Figure 20-2 Single-channel pulse counting mode plus counting waveform 460
Figure 20-3 Single channel pulse counting mode down counting waveform 460
Figure 20-4 Dual-channel non-crossing pulse counting mode plus counting waveform 461
Figure 20-5 Dual-channel non-crossing pulse counting mode down counting waveform 461
Figure 20-6 Dual-channel quadrature pulse counting mode plus counting waveform 462
Figure 20-7 Dual-channel quadrature pulse counting mode down counting waveform 462
Figure 20-8 Pulse Width Filtered Waveform 463
Figure 21-1 Block Diagram 476
Figure 21-2 Mode0 sending data 478
Figure 21-3 Mode0 receiving data 478
Figure 21-4 Mode1 sending data 479
Figure 21-5 Mode1 receiving data 479
Figure 21-6 Mode2 sending data 479
Figure 21-7 Mode2 receiving data 479
Figure 21-8 Mode3 sending data 480
Figure 21-9 Mode3 receiving data 480
Figure 21-10 UART hardware flow control 486
Figure 21-11 nRTS hardware flow control signal 487
Figure 21-12 nCTS hardware flow control signal 487
Figure 21-13 Receive buffer 488
Figure 21-14 Send cache 488
Figure 22-1 Block Diagram 495
Figure 22-2 Mode0 sending data 498
Figure 22-3 Mode0 receiving data 498
Figure 22-4 Mode1 sending data 499
Figure 22-5 Mode1 receiving data 499
Figure 22-6 Mode2 sending data 499
Figure 22-7 Mode2 receiving data 499
Figure 22-8 Mode3 sending data 500
Figure 22-9 Mode3 receiving data 500
Figure 22-10 LPUART hardware flow control 506
Figure 22-11 nRTS hardware flow control signal 507
Figure 22-12 nCTS hardware flow control signal 507
Figure 22-13 Receive buffer 508
Figure 22-14 Send cache 508
Figure 24-1 TRNG data flow 521
Figure 25-1 Schematic Diagram of AES Encryption and Decryption 528

Figure 25-2 AES Encryption Flowchart 529
Figure 27-1 CTS Counter 566
Figure 28-1 Functional block diagram 577
Figure 28-2 PhilipProtocol WaveForm (16/32 bit full precision) 580
Figure 28-3 Philip Protocol Waveform (24 -bit frame) 580
Figure 28-4 24 -Bit Data Transmission Write Operation 581
Figure 28-5 24 -Bit Data Mode Receive Read Operation 581
Figure 28-6 Philip Protocol 16 -Bit Extended To 32 -Bit Data Frame 581
Figure 28-7 16 -Bit DataRead And Write Operations 582
Figure 28-8 MSB protocol waveform (16/32 bit full precision) 582
Figure 28-9 MSB protocol waveform (24-bit frame) 583
Figure 28-10 MSB Protocol 16 -Bit Extended To 32 -Bit Data Frame 583
Figure 28-11 LSB protocol waveform (16/32 bit full precision) 584
Figure 28-12 LSB protocol waveform (24-bit frame) 584
Figure 28-13 LSB 24 -Bit Data Transmission Write Operation 585
Figure 28-14 LSB 24 -Bit Data Mode Receive Read Operation 585
Figure 28-15 LSB Protocol 16 -Bit Extended To 32 -Bit Data Frame 585
Figure 28-16 16 -Bit DataRead And Write Operations 586
Figure 28-17 PCM protocol waveform (16 bits) 586
Figure 28-18 PCM Protocol Waveform (16 -bit data extended to 32 -bit) 587
Figure 28-19 Audio Sampling Frequency Definition 588
Figure 28-20 Clock Generation 588
Figure 29-1 USBFS System Block Diagram 607
Figure 29-2 USBFS device mode system construction diagram 608
Figure 29-3 Schematic diagram of FIFO architecture in USBFS device mode 613
Figure 30-1 CAN System Block Diagram 678
Figure 30-2 CAN bit time definition diagram 679
Figure 30-3 CAN TBUF register write transmit buffer and schematic diagram 680
Figure 30-4 Schematic diagram of CAN RBUF register read receive buffer 681
Figure 30-5 Schematic diagram of CAN ACF register access filter group 681
Figure 30-6 Schematic diagram of CAN LBMI and LBME 685
Figure 31-1 ADC block diagram 717
Figure 31-2 ADC conversion timing diagram 718
Figure 31-3 ADC sequential scan conversion process example 721
Figure 31-4 ADC skipping scan conversion process example - 723
Figure 31-5 Example of skipping scan during ADC sequential scan 726
Figure 31-6 ADC Continuous Conversion Accumulation Process Example 727

Figure 31-7 Schematic diagram of ADC single conversion or sequential scan conversion external trigger source 729
Figure 31-8 Schematic diagram of external trigger source for ADC jumping scan conversion 729
Figure 31-9 ADC conversion result 730
Figure 32-1 Timing diagram of DAC conversion when trigger is disabled 757
Figure 32-2 DAC conversion of waveform generated by LFSR (using software trigger) 763
Figure 32-3 Generate DAC triangle wave 764
Figure 32-4 DAC conversion to generate triangle waveform (software trigger enabled) 764
Figure 33-1 VC Frame Diagram 777
Figure 33-2 VC Filter Response Time 778
Figure 33-3 VC hysteresis function 778
Figure 34-1 LVD Block Diagram 791
Figure 34-2 LVD hysteresis response 792
Figure 34-3 LVD filter output 792
Figure 37-1 Debug Support Block Diagram 806

Overview

The HC32L072 / HC32L073 series is an MCU with ultra-low power consumption and wide voltage operating range designed to extend the battery life of portable measurement systems. Integrate 12-bit 1Msps high-precision SARADC, 2 12-bit DACs and integrated comparator, operational amplifier, built-in high-performance PWM timer, LCD display, multiple UART, SPI, I2C, I2S, USB, CAN and other rich communications It has built-in information security modules such as AES and TRNG, which have the characteristics of high integration, high anti-interference, high reliability and ultra-low power consumption. The core of this product adopts the Cortex-M0+ core, cooperates with mature Keil & IAR debugging and development software, supports C language, assembly language, and assembly instructions.

Ultra-low power MCU typical applications

- Sensor applications, IoT applications
- Smart meters, wireless modules, thermostats, shelf labels
- Intelligent traffic, alarm system
- Smart home, medical equipment

About this manual

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

1 System Structure

1.1 Overview

This product system consists of the following parts:

- 2 AHB bus masters:
  - Cortex-M0+
  - DMA controller
- 4 AHB bus slaves:
  - FLASH memory
  - SRAM memory
  - AHB0, AHB to APB Bridge, including all APB interface peripherals
  - AHB1, contains all AHB interface peripherals

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

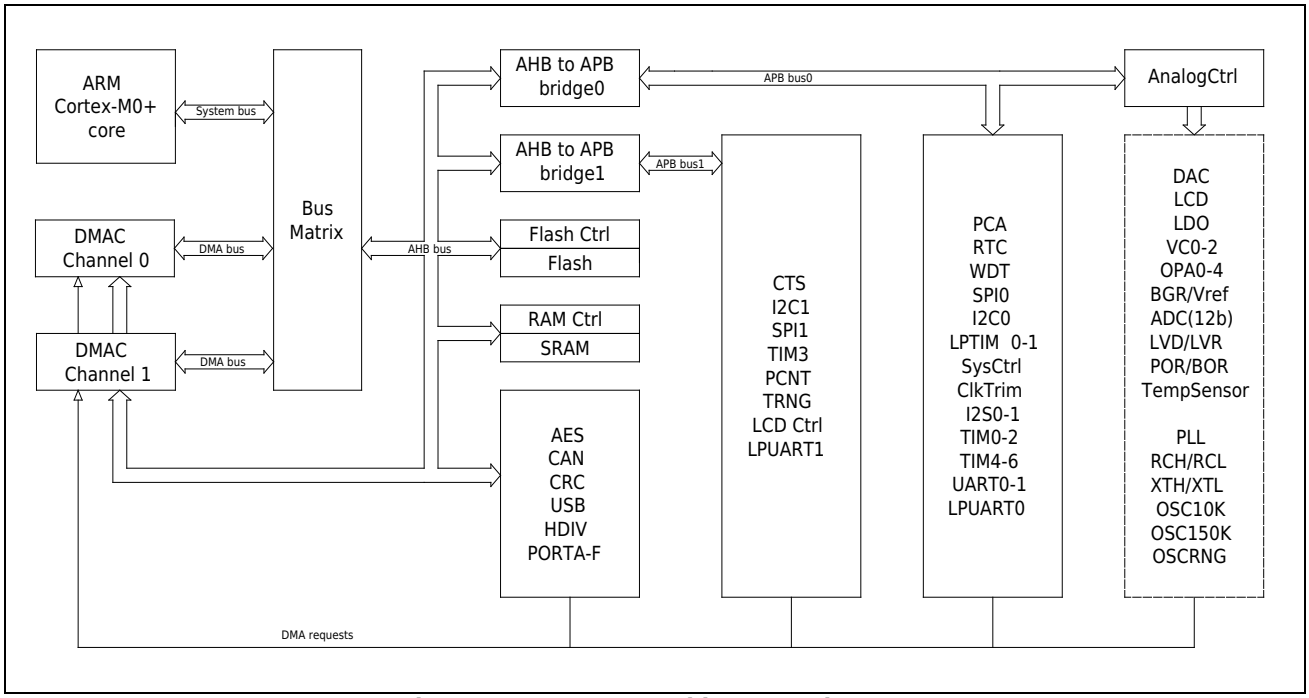
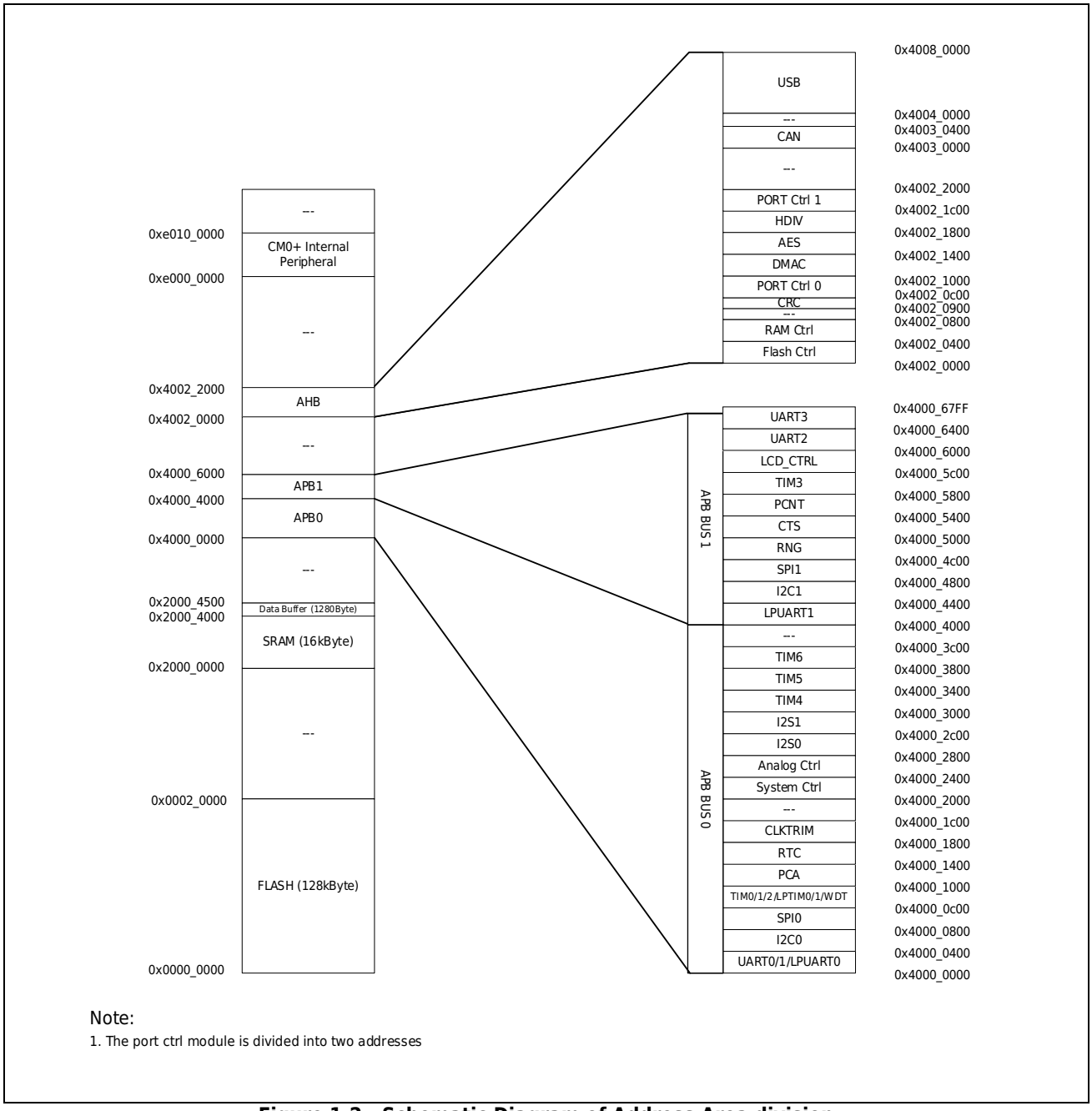


Figure 1-1 System Architecture Diagram

1.2 System address division

The address area division of the entire series of systems is shown in the following figure:



Note:

1. The port ctrl module is divided into two addresses

Figure 1-2 Schematic Diagram of Address Area division

1.3 Memory and Module Address Assignment

Table 1-1 Address Division Table

Boundary Address	Size	Memory Area	Description
0x0000_0000 – 0x0000_FFFF	128KByte	FLASH	
0x0001_0000 – 0x1FFF_FFFF	-	Reserved	
0x2000_0000 – 0x2000_1FFF	16KByte	SRAM	
0x2000_2000 – 0x3FFF_FFFF	-	Reserved	
0x4000_0000 – 0x4000_00FF	256Byte	UART0	
0x4000_0100 – 0x4000_01FF	256Byte	UART1	
0x4000_0200 – 0x4000_02FF	256Byte	LPUART0	
0x4000_0300 – 0x4000_03FF	-	Reserved	
0x4000_0400 – 0x4000_07FF	1KByte	I2C0	
0x4000_0800 – 0x4000_0BFF	1KByte	SPI0	
0x4000_0C00 – 0x4000_0CFF	256Byte	TIM0	
0x4000_0D00 – 0x4000_0DFF	256Byte	TIM1	
0x4000_0E00 – 0x4000_0EFF	256Byte	TIM2	
0x4000_0F00 – 0x4000_0F7F	128Byte	LPTIM0-1	
0x4000_0F80 – 0x4000_0FFF	128Byte	WDT	
0x4000_1000 – 0x4000_13FF	1KByte	PCA	
0x4000_1400 – 0x4000_17FF	1KByte	RTC	
0x4000_1800 – 0x4000_1BFF	1KByte	CLKTRIM	
0x4000_1C00 – 0x4000_1FFF	-	Reserved	
0x4000_2000 – 0x4000_23FF	1KByte	SYSCTRL	
0x4000_2400 – 0x4000_27FF	1KByte	ANALOGCTRL	
0x4000_2800 – 0x4000_2BFF	1KByte	I2S0	
0x4000_2C00 – 0x4000_2FFF	1KByte	I2S1	
0x4000_3000 – 0x4000_33FF	1KByte	TIM4	
0x4000_3400 – 0x4000_37FF	1KByte	TIM5	
0x4000_3800 – 0x4000_3BFF	1KByte	TIM6	
0x4000_3C00– 0x4000_3FFF	-	Reserved	
0x4000_4000– 0x4000_43FF	1KByte	LPUART1	
0x4000_4400– 0x4000_47FF	1KByte	I2C1	
0x4000_4800– 0x4000_4BFF	1KByte	SPI1	
0x4000_4C00– 0x4000_4FFF	1KByte	TRNG	
0x4000_5000– 0x4000_53FF	1KByte	CTS	
0x4000_5400– 0x4000_57FF	1KByte	PCNT	
0x4000_5800– 0x4000_5BFF	1KByte	TIM3	
0x4000_5C00– 0x4000_5FFF	1KByte	LCD	
0x4000_6000 – 0x4000_63FF	1KByte	UART2	

Boundary Address	Size	Memory Area	Description
0x4000_6400 – 0x4000_67FF	1KByte	UART3	
0x4000_6800 - 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	
0x4002_1000 - 0x4002_13FF	1KByte	DMAC	
0x4002_1400 - 0x4002_17FF	1KByte	AES	
0x4002_1800 - 0x4002_1BFF	1KByte	HDIV	
0x4002_1C00 - 0x4002_1FFF	1KByte	PORT CTRL1	
0x4003_0000 - 0x4003_03FF	1KByte	CAN	
0x4004_0000 - 0x4007_FFFF	256KByte	USB	

2 Operating mode

The power management module of this product is responsible for managing the switching between various working modes of this product, and controlling the working status of each functional module in each working mode. The operating voltage (VCC) of this product is 1.8V ~ 5.5V.

This product has the following working modes:

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

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

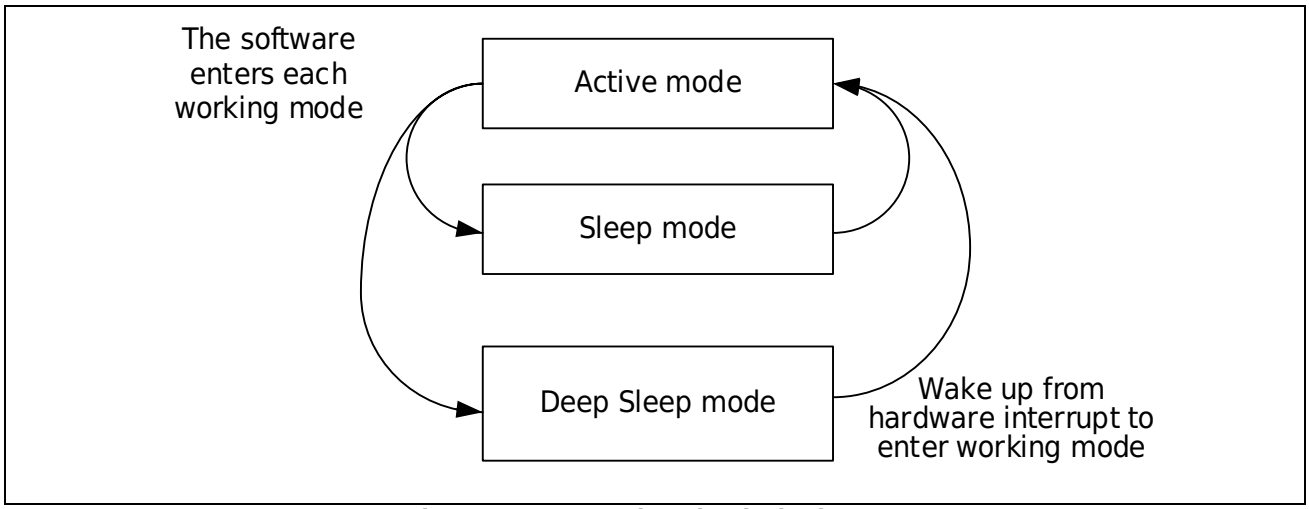


Figure 2-1 Control Mode Block Diagram

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

Interrupt vector number	Interrupt source	Operating mode	Sleep mode	Deep sleep mode
[0]	GPIO_PA	✓	✓	✓
[1]	GPIO_PB	✓	✓	✓
[2]	GPIO_PC/GPIO_PE	✓	✓	✓
[3]	GPIO_PD/GPIO_PF	✓	✓	✓
[4]	DMAC	✓	✓	
[5]	TIM3	✓	✓	
[6]	UART0	✓	✓	
[7]	UART1	✓	✓	
[8]	LPUART0	✓	✓	✓
[9]	LPUART1	✓	✓	✓
[10]	SPI0/I2S0	✓	✓	
[11]	SPI1/I2S1	✓	✓	

Interrupt vector number	Interrupt source	Operating mode	Sleep mode	Deep sleep mode
[12]	I2C0	✓	✓	
[13]	I2C1	✓	✓	
[14]	TIM0	✓	✓	
[15]	TIM1	✓	✓	
[16]	TIM2	✓	✓	
[17]	LPTIM0/LPTIM1	✓	✓	✓
[18]	TIM4	✓	✓	
[19]	TIM5	✓	✓	
[20]	TIM6	✓	✓	
[21]	PCA	✓	✓	
[22]	WDT	✓	✓	✓
[23]	RTC	✓	✓	✓
[24]	ADC/DAC	✓	✓	
[25]	PCNT	✓	✓	✓
[26]	VC0/VC1/VC2/LVD	✓	✓	✓
[27]	USB	✓	✓	
[28]	CAN	✓	✓	
[29]	LCD	✓	✓	✓
[30]	FLASH/RAM	✓	✓	
[31]	CLKTRIM/CTS	✓	✓	✓

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

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

2.1 Operating mode

The active mode of this product:

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

Table 2-1 Runnable module diagram in run mode

Active Mode		
Cortex-M0+	SWD	XTH
FLASH	CAN	RCH
RAM	USB	PLL
DMAC	UART0-3	ADC
TIM0-3	SPI0-1	DAC
TIM4-6	I2C0-1	TRNG
PCA	I2S0-1	OPA0-4
HDIV	CRC	LVD
AES	XTL	RESET
LPUART0-1	RCL	POR/BOR
LPTIM0-1	RTC	VC0-2
PCNT	LCD	WDT
GPIO	CLKTRIM	CTS

Several ways to reduce chip power consumption in run mode:

1) any one of the system clocks (HCLK, PCLK) can be slowed down by programming the prescaler registers (SYSCTRL0.HCLK_PRS, SYSCTRL0. PCLK_PRS). The prescaler can also be used to slow down the clocks to peripherals before entering Sleep mode.
2) In Run mode, turn off the clock (PERIx_CLKx) of unused peripherals to reduce power consumption.
3) In run mode, turn off the clocks of unused peripherals (PERIx_CLKx) to reduce power consumption, and put the system into sleep mode to reduce power consumption even more, and turn off the clocks of unused peripherals (PERIx_CLKEN. x).
4) Use low-power mode instead of sleep mode, because the wake-up time of this product is extremely short (~4us), which can also meet the real-time response requirements of the system.

2.2 Sleep mode

This product sleep mode

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

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

- How to enter sleep mode:
  Enter sleep state by executing WFI instruction. Depending on the value of the SLEEPONEXIT bit in the Cortex-M0+ System Control Register, there are two options for selecting the sleep mode entry mechanism:
  SLEEP-NOW: If the SLEEPONEXIT bit is cleared, the microcontroller enters sleep mode immediately when WFI or WFE is executed.
  SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the microcontroller enters sleep mode immediately when the system exits from the lowest priority interrupt handler.
- How to exit sleep mode:
  If the WFI instruction is executed to enter the sleep mode, any peripheral interrupt responded by the high-priority nested vector interrupt controller can wake up the system from the sleep mode.

Note:

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

Table 2-2 Runnable Module Diagram in Sleep Mode

Sleep Mode		
Cortex-M0+	SWD	XTH
FLASH	CAN	RCH
RAM	USB	PLL
DMAC	UART0-3	ADC
TIM0-3	SPI0-1	DAC
TIM4-6	I2C0-1	TRNG
PCA	I2S0-1	OPA0-4
HDIV	CRC	LVD
AES	XTL	RESET
LPUART0-1	RCL	POR/BOR
LPTIM0-1	RTC	VC0-2
PCNT	LCD	WDT
GPIO	CLKTRIM	CTS

Note: The modules in the gray dotted box are not working in the current state.

2.3 Deep sleep mode

This product deep sleep mode

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

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

How to enter deep sleep mode:

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

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

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

How to exit deep sleep mode:

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

Wake-up settings refer to3.4 Interrupt Wake-Up Control WIC.

Table 2-3 Runnable module diagram in deep sleep mode

Deep Sleep Mode		
Cortex-M0+	SWD	XTH
FLASH	CAN	RCH
RAM	USB	PLL
DMAC	UART0-3	ADC
TIM0-3	SPI0-1	DAC
TIM4-6	I2C0-1	TRNG
PCA	I2S0-1	OPA0-4
HDIV	CRC	LVD
AES	XTL	RESET
LPUART0-1	RCL	POR/BOR
LPTIM0-1	RTC	VC0-2
PCNT	LCD	WDT
GPIO	CLKTRIM	CTS

Note: The modules in the gray dotted box are not working in the current state.

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

Address: 0xE000ED10

Reset value: 0x0000 0000

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

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

3 System Controller (SYSCTRL)

3.1 System Clock Introduction

The clock control module mainly controls the system clock and the peripheral clock. Different clock sources can be configured as the system clock, different frequency divisions of the system clock can be configured, and peripheral clocks can be enabled or disabled. To ensure oscillator accuracy, the internal clocks are calibrated.

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

- Internal high-speed RC clock RCH (output frequency is 4~24MHz)
- Internal low-speed RC clock RCL (38.4K and 32.8K configurable)
- External high-speed crystal oscillator clock XTH
- External low-speed crystal oscillator clock XTL
- Phase-locked loop clock PLL

Note 1: When switching the clock source of the system clock, please strictly follow the operation steps to switch, see Chapter 3.2 for details.

Note 2: XTL can directly input 32.768kHz clock signal from PC14 pin without crystal oscillator. XTH can not be connected to the crystal oscillator, and directly input the 4~32MHz clock signal from the PF00 pin.

This product also contains the following three auxiliary clocks:

- The internal low-speed 10K clock is only used by the watchdog and CLKTRIM modules.
- The internal 150K clock is only used by LVD and VC modules.
- The internal 48M clock is only used by the USB module and can be automatically calibrated by the CTS module.

The figure below shows the clock architecture of this product.

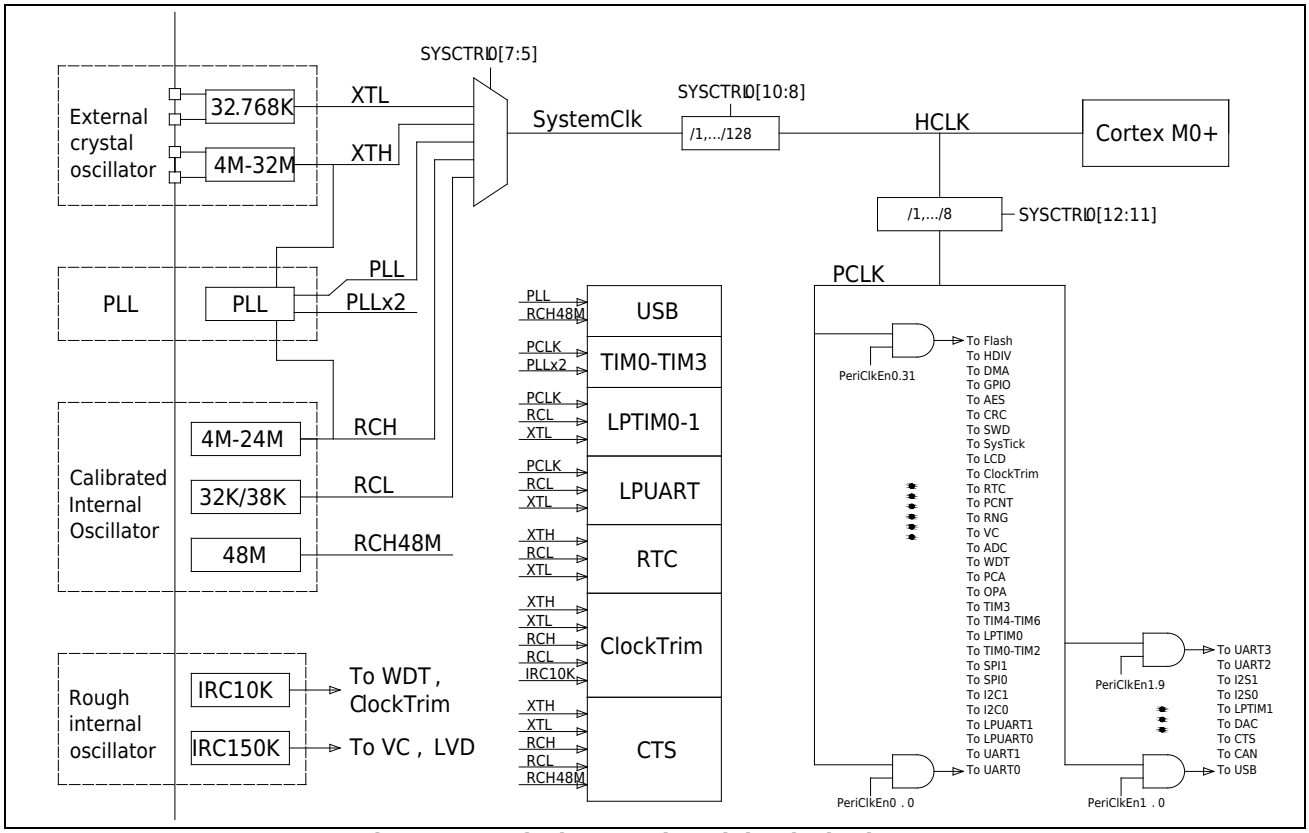


Figure 3-1 Clock Control Module Block Diagram

3.1.1 Internal high speed RC clock RCH

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

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

Five frequencies 4MHz, 8MHz, 16MHz, 22.12MHz, and 24MHz have been preset before leaving the factory; if you need other frequencies, please manually adjust the value of this register.

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

The internal high-speed clock only needs 4us from startup to stabilization. In order to respond to interrupts quickly in deep sleep mode, it is recommended to switch the system clock to RCH before entering deep sleep mode.

3.1.2 Internal low speed RC clock RCL

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

3.1.3 External low-speed crystal oscillator clock XTL

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

XTL can also not be connected to the crystal oscillator, and directly input the 32.768KHz clock signal from the PC14 pin. The method of inputting clock signal from PC14 is: configure PC14 pin as GPIO input; set SYSCTRL1. EXTL_EN to 1.

Note:

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

3.1.4 External high-speed crystal oscillator clock XTH

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

XTH can also not be connected to the crystal oscillator, and directly input the 4~32MHz clock signal from the PF00 pin. The method of inputting clock signal from PF00 is: configure PF00 pin as GPIO input; set SYSCTRL1. EXTH_EN to 1.

Note:

- The crystal and its matching devices must meet the relevant requirements of the **high-speed external clock XTH in the electrical characteristics of the data sheet**.
- In order to ensure the high reliability of XTH work, it is strongly recommended to use an external active crystal oscillator.

3.1.5 Phase-locked loop clock PLL

Built-in PLL supports 8~48MHz clock output. The reference clock sources of the PLL are: RCH, XTH crystal oscillator clock, and PF00 pin input clock.

PLL has a power consumption of about tens of microamps in deep sleep mode. It is recommended to configure the system clock as RCH/XTH and turn off PLL and BGR before entering deep sleep mode. After waking up, reconfigure the system clock as PLL.

3.1.6 Internal high speed clock RCH48M

The internal high-speed clock RCH48M can be used as the clock for crystall-less USB, and it can be automatically calibrated by using the module CTS. Cannot be used as system clock.

3.1.7 Clock start process

The above five clock sources all require startup stabilization time. The following figure uses the external XTH as an example to illustrate the clock startup stabilization process.

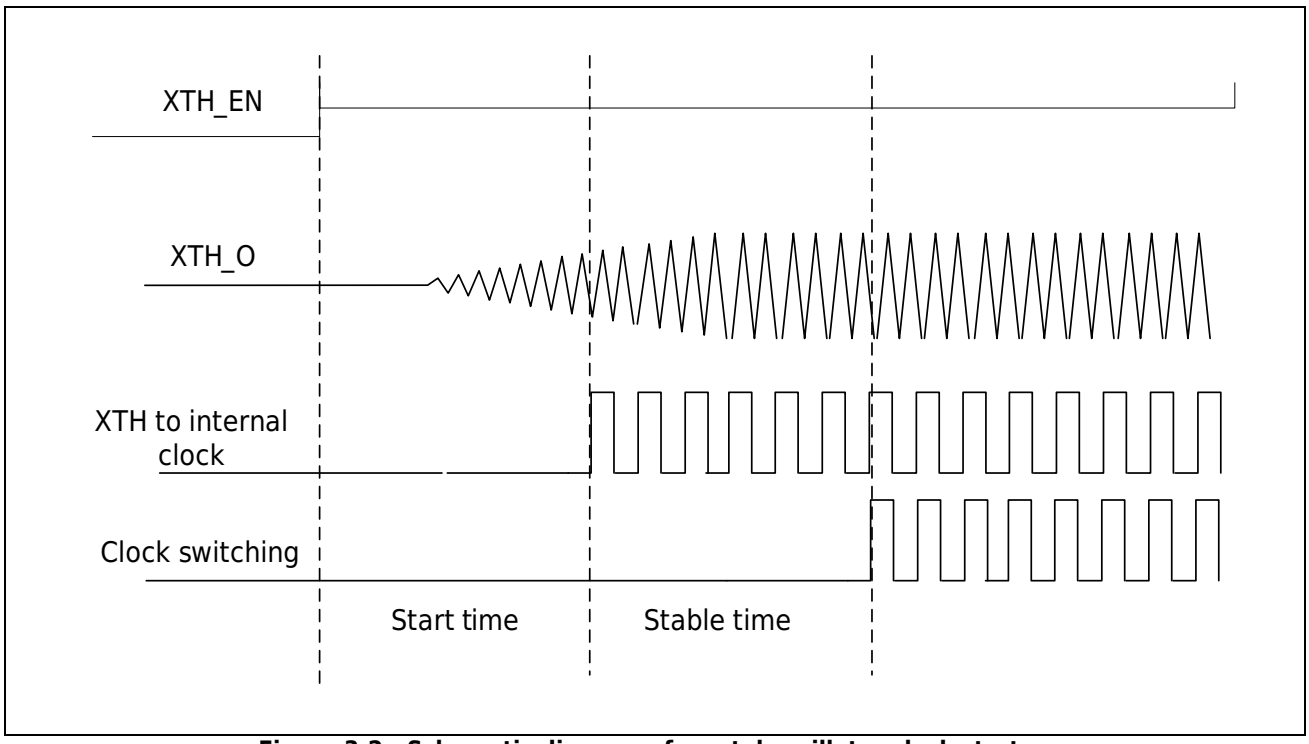


Figure 3-2 Schematic diagram of crystal oscillator clock startup

3.2 System Clock Switching

The switching of the clock source is controlled by the register SYSCTRL0[7:0]. The clock source of the system clock can be switched among RCH, RCL, XTH, XTL, and PLL through SYSCTRL0[7: 5]. Any two of the four clock sources RCH, RCL, XTH, and XTL can be switched with each other during switching; the PLL can only be switched with the two clocks of RCH and XTH. The clock switching operation must be performed according to the seven clock switching processes described below, otherwise abnormalities may occur.

FLASH_CR. WAIT needs to be configured synchronously when the clock is switched. If the clock frequency is not greater than 24MHz, FLASH_CR.WAIT should be set to 0; if the clock frequency is

greater than 24MHz, FLASH_CR.WAIT should be set to 1; if the clock frequency is greater than 48MHz, FLASH_CR.WAIT should be set to 2.

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

3.2.1 Standard Clock Switching Process

The operation process is as follows:

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

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

Step2: Configure the oscillation parameters of the new clock source.

Step3: Enable the oscillator of the new clock source.

Step4: According to the higher frequency of the current clock source and the new clock source, configure FLASH_CR. WAIT according to the procedures in the Flash controller chapter.

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

Step6: Configure SYSCTRL0. CLKSW, select the source of the system clock as the new clock source.

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

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

3.2.2 RCH switching process between different oscillation frequencies

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

Scheme 1:

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

Step2: Set SYSCTRL0.HCLK_PRS to 0x7.

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

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

Step 5: Set SYSCTRL0.HCLK_ PRS is 0x0.

```
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
```

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

Scheme 2:

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

Step2: Switch the system clock to RCH, see 3.2.6 Example of switching from other clocks to RCH.

3.2.3 XTL Example from Other Clocks

The operation process is as follows:

Step1: Set PCADS. 14 and PCADS. 15 to 1, and configure PC14/PC15 pins as analog ports.

Step2: Configure XTL_CR[5:0] according to the characteristics of the crystal oscillator.

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

Step7: Set SYSCTRL0. CLKSW to 3, switch the system clock to XTL.

Step8: Set FLASH_CR. WAIT to 0 according to the procedures in the Flash controller chapter.

Step9: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step10: Set SYSCTRL0.xxx_EN to 0, turn off the original clock.

3.2.4 Switching from other clocks to XTH example

Step1: Set PFADS. 0 and PFADS. 1 to 1, and configure PF00/PF01 pins as analog ports.

Step2: Configure XTH_CR[3:0] according to the characteristics of the crystal oscillator.

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

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

Step5: Set SYSCTRL0. XTH_EN to 1 to enable the crystal oscillator circuit.

Step6: According to the higher frequency of the current clock and XTH, configure according to the process of the Flash controller chapter

FLASH_CR.WAIT.

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

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

Step9: Set SYSCTRL0. CLKSW to 1, switch the system clock to XTH.

Step10: According to the frequency of XTH, configure FLASH_CR. WAIT according to the procedures in the chapter of Flash controller.

Step11: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step12: Set SYSCTRL0.xxx_EN to 0, turn off the original clock.

3.2.5 Switching from other clocks to RCL example

The operation process is as follows:

Step1: Configure RCL_CR.TRIM and RCL_CR.Startup.

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

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

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

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

Step6: Set SYSCTRL0. CLKSW to 2, switch the system clock to RCL.

Step7: Set FLASH_CR. WAIT to 0 according to the procedures in the Flash controller chapter.

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

Step9: Set SYSCTRL0.xxx_EN to 0, turn off the original clock.

3.2.6 Switching from other clocks to RCH example

The operation process is as follows:

Step1: Configure RCH_CR.TRIM.

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

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

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

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

Step6: Set SYSCTRL0. CLKSW to 0, switch the system clock to RCH.

Step7: Set FLASH_CR. WAIT to 0 according to the procedures in the Flash controller chapter.

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

Step9: Set SYSCTRL0.xxx_EN to 0, turn off the original clock.

3.2.7 Example of switching between PLL and RCH, the reference clock is RCH

The process of switching from RCH to PLL is as follows:

Step1: Set BGR_CR to 0x01 and delay 20us, wait for BGR to start and stabilize.

Step2: Set PLL_CR.REFSEL to 3, select PLL clock source as RCH.

Step3: Configure PLL_CR.FRSEL according to the RCH frequency.

Step4: Configure PLL_CR.DIVN and PLL_CR.FOSC according to the PLL output frequency.

Step5: Set PLL_CR. Startup to 7, choose the longest PLL stabilization time.

Step6: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step7: Set SYSCTRL0.PLL_EN to 1 to enable the PLL oscillation circuit.

Step8: According to the PLL output frequency, configure FLASH_CR. WAIT according to the procedures in the Flash controller chapter.

Step9: Query and wait for the PLL_CR. Stable flag to become 1, and the PLL outputs a stable clock.

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

Step11: Set SYSCTRL0. CLKSW to 4, switch the system clock to PLL.

RCH_CR.TRIM must not be changed during switching.

The operation process of switching from PLL to RCH is as follows:

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

Step2: Set SYSCTRL0. CLKSW to 0, switch the system clock to RCH.

Step3: Set FLASH_CR. WAIT to 0 according to the procedures in the Flash controller chapter.

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

Step5: Set SYSCTRL0.PLL_EN to 0, turn off PLL.

Step6: Set BGR_CR to 0x01 to 0, turn off BGR.

RCH_CR.TRIM must not be changed during switching.

3.2.8 Example of switching between PLL and XTH, the reference clock is XTH

The process of switching from RCH to PLL is as follows:

Step1: Set BGR_CR to 0x01 and delay 20us, wait for BGR to start and stabilize.

Step2: Set PLL_CR.REFSEL to 0, select the PLL clock source as XTH.

Step3: According to XTH frequency, configure PLL_CR.FRSEL.

Step4: Configure PLL_CR.DIVN and PLL_CR.FOSC according to the PLL output frequency.

Step5: Set PLL_CR. Startup to 7, choose the longest PLL stabilization time.

Step6: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step7: Set SYSCTRL0.PLL_EN to 1 to enable the PLL oscillation circuit.

Step8: According to the PLL output frequency, configure FLASH_CR. WAIT according to the procedures in the Flash controller chapter.

Step9: Query and wait for the PLL_CR. Stable flag to become 1, and the PLL outputs a stable clock.

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

Step11: Set SYSCTRL0. CLKSW to 4, switch the system clock to PLL.

The operation process of switching from PLL to XTH is as follows:

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

Step2: Set SYSCTRL0. CLKSW to 1, switch the system clock to XTH.

Step3: According to the frequency of XTH, configure FLASH_CR. WAIT according to the procedures in the chapter of Flash controller.

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

Step5: Set SYSCTRL0.PLL_EN to 0, turn off PLL.

Step6: Set BGR_CR to 0x01 to 0, turn off BGR.

The following figure is the clock switching timing diagram:

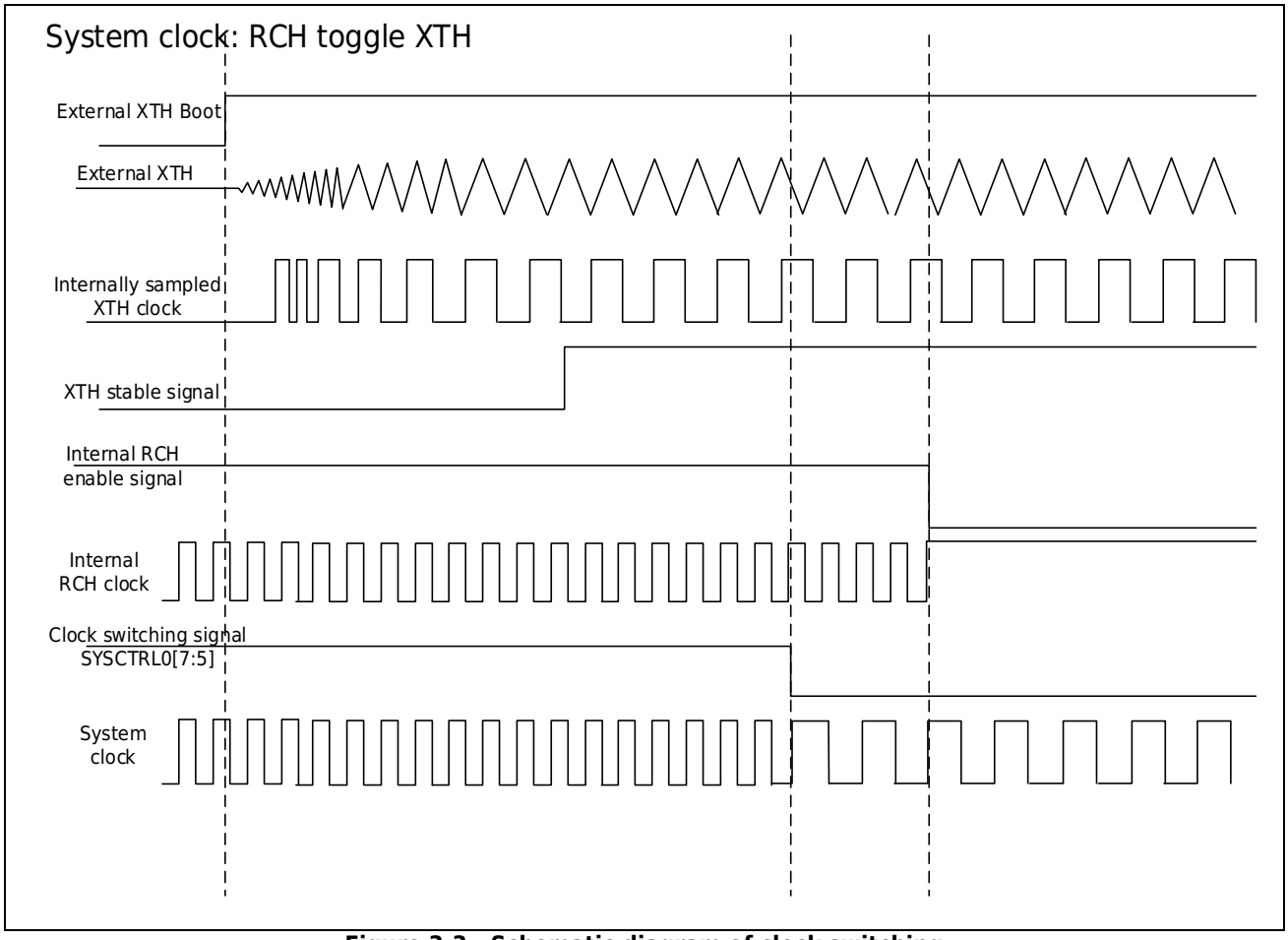


Figure 3-3 Schematic diagram of clock switching

3.3 Clock Calibration Module

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

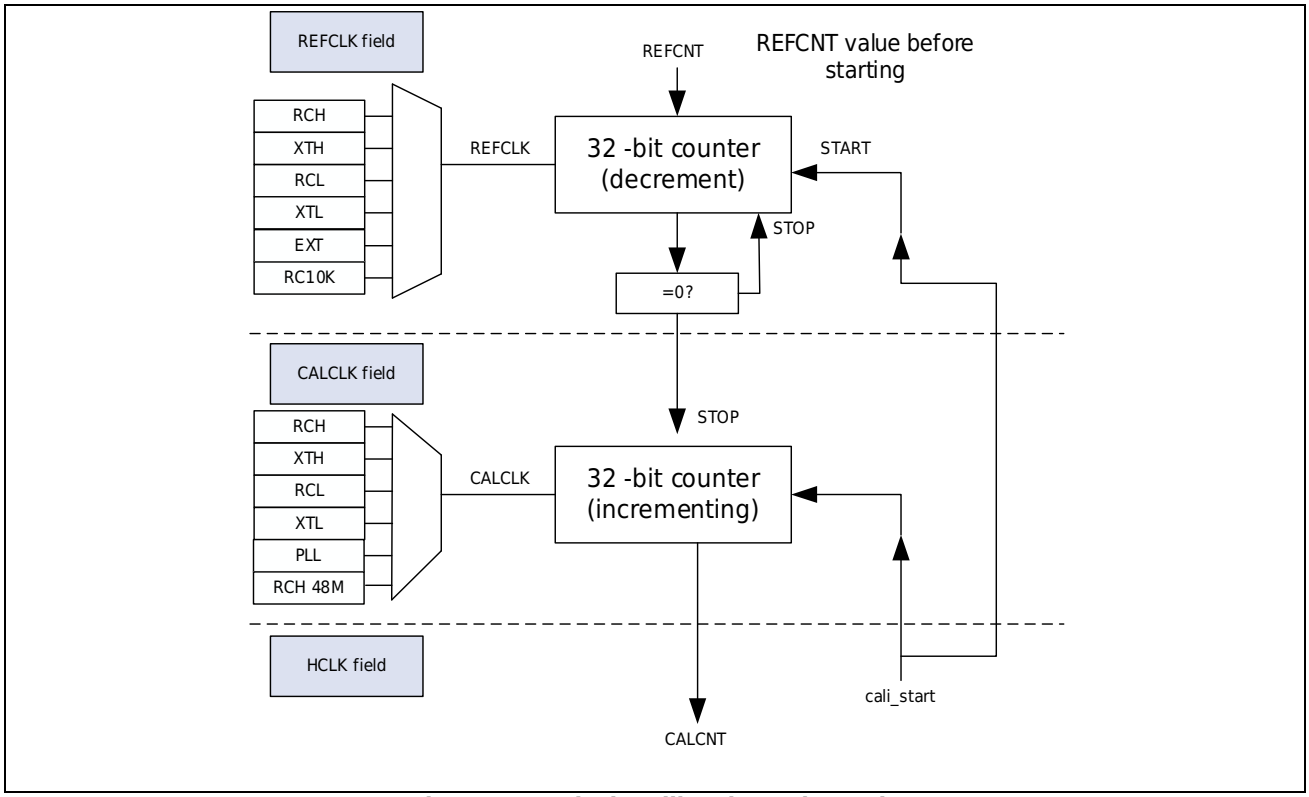


Figure 3-4 Clock Calibration Schematic

3.4 Interrupt Wakeup Control

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

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

PRIMASK status	WFI behavior	Wakeup	ISR execution
0	IRQ Priority > Current Class	Y	Y
0	IRQ Priority ≦ Current Level	N	N
1	IRQ Priority > Current Class	Y	N
1	IRQ Priority ≦ Current Level	N	N

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

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set SCB->SCR.SLEEPDEEP to 1
4. Execute WFI instruction to enter deep sleep mode
5. The system enters the deep sleep mode and waits for the interrupt to wake up, and executes the interrupt service program after waking up

Routine:

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

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

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

Routine:

```
__asm("CPSID I");     //Set PRIMASK
SCB->SCR |= 0x00000004u;
while(1)
{
                        __asm("WFI");
                        M0P_TIMx->ICLR = 0x00;                  //Clear Int Flag
                        NVIC_ClearPendingIRQ(TIMERLP_IRQn ); //Clear Pending Flag
                ...     // perform user-defined operations
}
```

3.4.3 Using exit hibernation feature

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

Cortex-M0 uses the exit dormancy feature to enter dormancy. This situation is similar to the effect of executing WFI immediately after executing an abnormal exit. However, in order to avoid the need to push the stack when entering an exception next time, the processor will not perform the process of popping the stack.

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set SCB->SCR.SLEEPDEEP to 1
4. Set SCB->SCR.SLEEPONEXIT to 1
5. Execute WFI instruction to enter deep sleep mode
6. The system enters the deep sleep mode and waits for the interrupt to wake up, and executes the interrupt service subroutine after waking up
7. Automatically enters sleep mode when exiting interrupt service

Routine:

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

3.5 Register

Base address 0x40002000

Table 3-1 System Control Register

Register	Offset address	Description
SYSCTRL0	0x000	System Control Register 0
SYSCTRL1	0x004	System Control Register 1
SYSCTRL2	0x008	System Control Register 2
RCH_CR	0x00C	RCH Control Register
XTH_CR	0x010	XTH Control Register
RCL_CR	0x014	RCL control register
XTL_CR	0x018	XTL Control Register
PLL_CR	0x03C	PLL Control Register
PERI_CLKEN0	0x020	Peripheral Module Clock Control Register
PERI_CLKEN1	0x024	Peripheral Module Clock Control Register
OVCK_CR	0x054	Timer clock multiplication control
RC48M_CR	0x058	RC48M Control Register

3.5.1 System Control Register 0 (SYSCTRL0)

Offset address: 0x000

Reset value: 0x0000 0001

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WakeupBy RCH	Reserved		PclkPrs		HclkPrs			CLKSW			PLLE N	XTL EN	RCL EN	XTHE N	RCHE N
RW			RW		RW			RW			RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:16	Reserved	Keep
15	Wakeup ByRCH	1: After waking up from Deep Sleep, the system clock source is RCH, and the original clock continues to be enabled. 0: After waking up from Deep Sleep, do not change the system clock source.
14:13	Reserved	Keep
12:11	PclkPrs	PCLK clock source selection 00: HCLK 01: HCLK/2 10: HCLK/4 11: HCLK/8
10:8	HclkPrs	HCLK clock source selection 000: SystemClk 001: SystemClk/2 010: SystemClk/4 011: SystemClk/8 100: SystemClk/16 101: SystemClk/32 110: SystemClk/64 111: SystemClk/128
7:5	CLKSW	SystemClk clock source selection 000: Internal high-speed clock RCH 001: External high-speed crystal oscillator XTH 010: Internal low-speed clock RCL 011: External low-speed crystal oscillator XTL 100: Internal PLL
4	PLLEN	PLL enable control 0: OFF 1: Enable Note: PLL can only be enabled after BGR is enabled and stable.
3	XTLEN	External low-speed crystal oscillator XTL enable control 0: OFF 1: Enable Note: PC14 and PC15 need to be set as analog ports.
2	RCLEN	Internal low-speed clock RCL enable control 0: OFF 1: Enable
1	XTHEN	External high-speed crystal oscillator XTH enable control 0: OFF 1: Enable Note: When the system enters DeepSleep, this high-speed clock will be automatically turned off
0	RCHEN	Internal high-speed clock RCH enable signal. 0: OFF 1: Enable Note: When the system enters DeepSleep, this high-speed clock will be automatically turned off.

Note:

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

3.5.2 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		USB48MSEL	Res	RTC_FREQ_ADJUST			SWD_USE_IO	Res.	LOCKUP_EN	RTC_LPW	Res.	XTL_ALWAYS_ON	EXTL_EN	EXTH_EN	Res.
		RW		RW			RW		RW	RW		RW	RW	RW	

Bit	Marking	Functional description
31:14	Reserved	Reserved bit
13	USB48MSEL	USB clock selection 0: RCH48M; 1: PLL multiplier clock
12	Res.	Reserved bit
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	Res.	Reserved bit
6	LOCKUP_EN	Cortex-M0+ LockUp function configuration 0: off 1: Enable Note: After enabling, the CPU will reset the MCU when it reads an invalid command, which can enhance system reliability.
5	RTC_LPW	RTC module low power control 1: Low power mode enabled 0: Low power mode disabled Note: After enabling, the RTC module enters a low power consumption state, and its registers cannot be read or written.
4	Res.	Reserved bit
3	XTL_ALWAYS_ON	XTL Advanced Enable Control 1: SYSCTRL0.XTL_EN can only be set. 0: SYSCTRL0.XTL_EN can be set and cleared.
2	EXTL_EN	External XTL clock input control 1: XTL output clock is input from PC14. 0: XTL output clock is generated by crystal oscillator. Note: When using PC14 to input clock, you need to set SYSCTRL0.XTL_EN to 1; When the PC14 external input clock is used, the PC15 pin cannot be used as the GPIO. The PC15 port must be set to the analog state.
1	EXTH_EN	External XTH input control 1: XTH output clock is input from PF00. 0: XTH output clock is generated by crystal oscillator. Note: When using PF00 to input clock, you need to set SYSCTRL0.XTH_EN to 1; When the PF00 external input clock is used, the PF01 pin cannot be used as the GPIO. The PF01 port must be set to the analog state.
0	Reserved	Reserved bit

Note:

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

3.5.3 System Control Register 2 (SYSCTRL2)

Offset address: 0x008

Reset value: 0x00000000

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

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

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

3.5.4 RCH Control Register (RCH_CR)

Offset address: 0x00C

Reset value: 0x00000126

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

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

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

3.5.5 XTH Control Register (XTH_CR)

Offset address: 0x010

Reset value: 0x00000022

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

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

Bit	Marking	Functional description
31:7	Reserved	Reserved bit
6	stable	External high-speed clock XTH stable flag bit. 1: It means that XTH is stable and can be used by internal circuits. 0: It means that XTH is not stable and cannot be used by internal circuits. Note: In order to increase system reliability, after querying this flag, the software needs to delay more than 10ms before switching the system clock to XTH.
5:4	Startup	External high-speed clock XTH stabilization time selection 00: 256 cycles; 01: 1024 cycles; 10: 4096 cycles; 11: 16384 cycles; Note: It is highly recommended to set the stabilization time of XTH to 11. If the XTH stabilization time is insufficient, the system will not work stably when performing clock switching or waking up from deep sleep.
3:2	xth_fsel	External crystal oscillator operating frequency selection 11: 24M~32M 10: 16M~24M 01: 8M~16M 00: 4M~8M
1:0	Driver	External Crystal Oscillator Drive Capability Selection 11: The strongest driving ability 10: Default drive capacity (recommended value) 01: Weak driving ability 00: Weakest drive capability Note: It is necessary to select the appropriate driving capability according to the characteristics of the crystal oscillator, the load capacitance and the parasitic parameters of the circuit board. The greater the driving capability, the greater the power consumption; the weaker the driving capability, the smaller the power consumption.

3.5.6 RCL Control Register (RCL_CR)

Offset address: 0x014

Reset value: 0x0000033Fh

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

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

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

3.5.7 XTL Control Register (XTL_CR)

Offset address: 0x018

Reset value: 0x00000021

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

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

Bit	Marking	Functional description
31:7	Reserved	
6	stable	External low-speed crystal oscillator XTL stable flag bit. 1: It means that XTL is stable and can be used by internal circuits. 0: It means that XTL is not stable and cannot be used by internal circuits.
5:4	Startup	External low-speed crystal oscillator XTL stabilization time selection 00: 256 cycles; 01: 1024 cycles; 10: 4096 cycles; 11: 16384 cycles;
3:2	amp_sel	XTL crystal oscillation amplitude. 11: Maximum amplitude 10: Larger amplitude (recommended value) 01: Mormal amplitude 00: Minimum amplitude
1:0	Driver	XTL crystal oscillator drive capability selection 11: The strongest driving ability 10: Strong driving ability 01: general driving ability (Recommended value) 00: Weakest drive capability Note: It is necessary to select the appropriate driving capability according to the characteristics of the crystal oscillator, the load capacitance and the parasitic parameters of the circuit board. The greater the driving capability, the greater the power consumption; the weaker the driving capability, the smaller the power consumption.

3.5.8 PLL Control Register (PLL_CR)

Offset address: 0x03C

Reset value: 0x000010B0F

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Stableopt	FRSEL		LFSEL		IBSEL		DVIN				FOSC			REFSEL	
RW	RW		RW		RW		RW				RW			RW	

Bit	Marking	Functional description
31:19	Reserved	Keep
18	Stable	PLL stable flag 0: PLL is not stable 1: PLL has stabilized
17:15	Startup	PLL Stability Time Selection 000: 128 PLL cycles 001: 256 PLL cycles 010: 512 PLL cycles 011: 1024 PLL cycles 100: 2048 PLL cycles 101: 4096 PLL cycles 110: 8192 PLL cycles 111: 16384 PLL cycles
14:13	FRSEL	PLL input frequency selection, configure as follows according to the frequency of PLL input clock 00: 4M~6M 01: 6M~12M 10: 12M~20M 11: 20M~24M
12:11	LFSEL	PLL filter control bits, please keep the default value
10:9	IBSEL	PLL bias current selection, please keep the default value
8:5	DIVN	The PLL output clock, PLL output frequency and input frequency output frequency = DIVN* input frequency, DIVN allows range 0X2~0XC
4:2	FOSC	PLL output frequency range selection, configure as follows according to PLL output clock frequency 000: 8M~12M 001: 12M~18M 010: 18M~24M 011: 24M~36M 1xx: 36M~48M
1:0	REFSEL	Input Clock Selection x0: Clock generated by XTH crystal oscillator 01: XTH clock input from pin PF00. (See details 3.1.4) 11: RCH clock

3.5.9 Peripheral Module Clock Control Register (PERI_CLKEN0)

Reset value: 0x8080_0000

Offset address: 0x020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLASH	HDIV	DMA	GPIO	AES	CRC	SWD	TICK	Res.	LCD	Trim	RTC	PCNT	TRNG	VC	ADC
RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDT	PCA	OPA	Res.	TIM3	ADV TIM	LP TIM0	BASE TIM	SPI1	SPI0	I2C1	I2C0	LPUART1	LPUART0	UART1	UART0
RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31	FLASH	FLASH controller module clock enable. After closing, the FLASH configuration registers cannot be written, and the programs in the FLASH can still run. 1: Enable; 0: Disable
30	HDIV	HDIV module clock enable. 1: Enable; 0: Disable
29	DMA	DMAC module clock enable. 1: Enable; 0: Disable
28	GPIO	GPIO module clock enable. 1: Enable; 0: Disable
27	AES	AES module clock enable. 1: Enable; 0: Disable
26	CRC	CRC module clock enable. 1: Enable; 0: Disable
25	SWD	SWD module clock enable. 1: Enable; 0: Disable
24	TICK	SysTick timer reference clock enable. 1: Enable; 0: Disable
23	Res.	Reserved bit
22	LCD	LCD module clock enable. 1: Enable; 0: Disable
21	TRIM	CLKTRIM module clock enable. 1: Enable; 0: Disable
20	RTC	RTC module clock enable. 1: Enable; 0: Disable
19	PCNT	PCNT module clock enable. 1: Enable; 0: Disable
18	TRNG	TRNG module clock enable. 1: Enable; 0: Disable
17	VC	VC, LVD, module clock enable. 1: Enable; 0: Disable
16	ADC	ADC, BGR 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	OPA	OPA module clock enable. 1: Enable; 0: Disable
12	Res.	Reserved bit
11	TIM3	TIM3 module clock enable. 1: Enable; 0: Disable
10	ADVTIM	TIM456 module clock enable. 1: Enable; 0: Disable
9	LPTIM0	LPTIM0 module clock enable. 1: Enable; 0: Disable
8	BASETIM	TIM012 module clock enable. 1: Enable; 0: Disable
7	SPI1	SPI1 module clock enable. 1: Enable; 0: Disable

6	SPI0	SPI0 module clock enable. 1: Enable; 0: Disable
5	I2C1	I2C1 module clock enable. 1: Enable; 0: Disable
4	I2C0	I2C0 module clock enable. 1: Enable; 0: Disable
3	LPUART1	LPUART1 module clock enable. 1: Enable; 0: Disable
2	LPUART0	LPUART0 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

3.5.10 Peripheral Module Clock Control Register (PERI_CLKEN1)

Reset value: 0x0000_0000

Offset address: 0x024

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						UART3	UART2	RES	I2S1	I2S0	LPTIM1	DAC	CTS	CAN	USB
						R/W	R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Marking	Functional description
31:10	Reseved	Reserved bit
9	UART3	UART3 module clock enable. 1: Enable; 0: Disable
8	UART2	UART2 module clock enable. 1: Enable; 0: Disable
7	RES.	Reserved bit
6	I2S1	I2S1 module clock enable. 1: Enable; 0: Disable
5	I2S0	I2S0 module clock enable. 1: Enable; 0: Disable
4	LPTIM1	LPTIM1 module clock enable. 1: Enable; 0: Disable
3	DAC	DAC module clock enable. 1: Enable; 0: Disable
2	CTS	CTS module clock enable. 1: Enable; 0: Disable
1	CAN	CAN module clock enable. 1: Enable; 0: Disable
0	USB	USB module clock enable. 1: Enable; 0: Disable

3.5.11 Timer Clock Multiplication Control (OVCK_CR)

Reset value 0x00000
Offset address: 0x054

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															OVCK
															RW

Bit	Marking	Functional description
31:1	Reserved	Reserved bit
0	OVCK	When the timer uses the PLL clock, the timer uses the clock selection control. 0: The timer uses the system clock at the same frequency; 1: The timer uses the system clock with a frequency multiplied by 2 Timer switching clock (when changing OVCK) needs to turn off the Peri_clken of the corresponding timer

3.5.12 RC48M Control Register (RC48M_CR)

Offset address: 0x058
Reset value: 0x0280

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.			EN	Stable	Res.	TRIM									
			RW	R		R/W									

Bit	Marking	Functional description
31:13	Reseved	Reserved bit
12	EN	RCH48M module is enabled. 1: Enable; 0: Disable
11	Stable	RCH48M clock stability flag 0: stable; 1 not stable
10	RES.	Reserved bit
9:0	TRIM	RCH48M calibration value

4 Reset Controller (RESET)

4.1 Reset Controller Introduction

This product has 7 reset signal sources, each reset signal can make the CPU run again, most of the registers will be reset to the reset value, and the program will start to execute from the reset vector.

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

Each reset source is indicated by a corresponding reset flag. Reset flags are set by hardware and need to be cleared by user software. When the chip is reset, if Reset_flag. POR15V or Reset_flag. POR5V is 1, it is a power-on reset. The user program should clear the register Reset_flag to 0 during power-on reset, and then the reset source can be judged by the relevant bits of Reset_flag at the next reset.

The figure below describes the reset sources for each area.

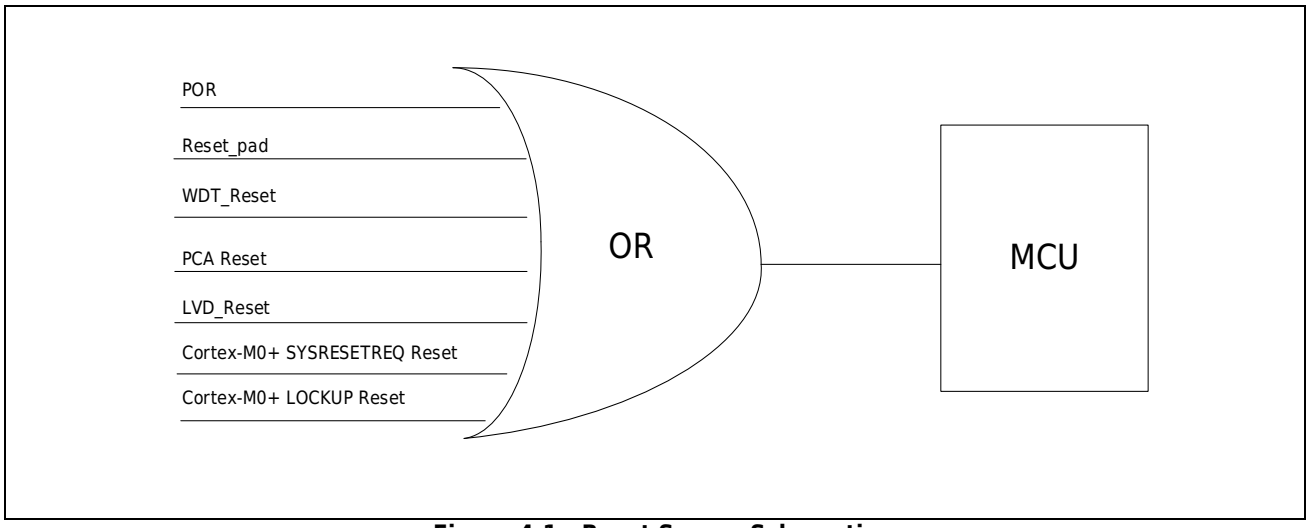


Figure 4-1 Reset Source Schematic

4.1.1 Power-on and power-off reset POR

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

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

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

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

4.1.2 External reset pin reset

A system reset is generated when the external reset pin detects a low level. The reset pin has a built-in pull-up resistor and integrates a glitch filter circuit. The glitch filter circuit will filter the glitch signal less than 20us (typical value), therefore, the low level signal added to the reset pin must be greater than 20us, in order to ensure reliable reset of the chip.

4.1.3 WDT reset

Watchdog reset, please refer to chapter WDT.

4.1.4 PCA reset

PCA reset, please refer to the chapter PCA.

4.1.5 LVD low voltage reset

LVD reset, please refer to chapter LVD.

4.1.6 Cortex-M0+ SYSRESETREQ reset

Cortex-M0+ software reset

4.1.7 Cortex-M0+ LOCKUP reset

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

4.2 Register

4.2.1 Reset flag register (RESET_FLAG)

Reset value: 00000000_00000000_00000000_xxxxxx11b

Address: 0x4000201C

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RSTB	sysreq	lockup	PCA	WDT	LVD	Por15	Por5v
								RW0	RW0	RW0	RW0	RW0	RW0	RW0	RW0

Bit	Marking	Functional description
31:8	Reserved	Reserved bit
7	RSTB	RESETB port reset flag, needs software initialization and clearing, power-on status is uncertain 1: A port reset occurred 0: No port reset occurs Write 0 to clear, write 1 to have no effect
6	Sysreq	Cotrex-M0+ CPU software reset flag, needs software initialization and clearing, power-on status is uncertain 1: Cortrex-M0+ CPU software reset occurred 0: No Cotrex-M0+ CPU software reset occurs Write 0 to clear, write 1 to have no effect
5	Lockup	Cotrex-M0+ CPU Lockup reset flag, needs software initialization and clearing, power-on status is uncertain 1: A Cortrex-M0+ CPU Lockup reset occurred 0: No Cortrex-M0+ CPU Lockup reset occurs Write 0 to clear, write 1 to have no effect
4	PCA	PCA reset flag, needs software initialization and clearing, power-on status is uncertain 1: PCA reset occurs 0: No PCA reset occurs Write 0 to clear, write 1 to have no effect
3	WDT	WDT reset flag, needs software initialization and clearing, power-on status is uncertain 1: WDT reset occurs 0: No WDT reset occurs Write 0 to clear, write 1 to have no effect
2	LVD	LVD reset flag, needs software initialization and clearing, power-on status is uncertain 1: LVD reset occurs 0: No LVD reset occurs Write 0 to clear, write 1 to have no effect
1	POR15V	VCAP domain reset flag 1: VCAP domain reset occurred 0: No reset of VCAP domain occurs Write 0 to clear, write 1 to have no effect
0	POR5V	VCC power domain reset flag 1: VCC power domain reset occurred 0: No reset occurs in VCC power domain Write 0 to clear, write 1 to have no effect

4.2.2 Peripheral Module Reset Control Register (PERI_RESET0)

Reset value: 0x7F7F6FFF

Address: 0x40002028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	HDIV	DMA	GPIO	AES	CRC	SWD	TICK	Res.	LCD	Trim	RTC	PCNT	TRNG	VC	ADC
	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	PCA	OPA	Res.	TIM3	ADV TIM	LP TIM0	BASE TIM	SPI1	SPI0	I2C1	I2C0	LPUA RT1	LPUA RT0	UART 1	UART 0
	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31	Res.	Reserved bit
30	HDIV	HDIV module reset enable. 1: normal operation; 0: module is in reset state
29	DMA	DMAC module reset enable. 1: normal operation; 0: module is in reset state
28	GPIO	GPIO module reset enable. 1: normal operation; 0: module is in reset state
27	AES	AES module reset enable. 1: normal operation; 0: module is in reset state
26	CRC	CRC module reset enable. 1: normal operation; 0: module is in reset state
25	SWD	SWD module reset enable. 1: normal operation; 0: module is in reset state
24	TICK	SYSTICK module reset enable. 1: normal operation; 0: module is in reset state
23	Res.	Reserved bit
22	LCD	LCD module reset enable. 1: normal operation; 0: module is in reset state
21	TRIM	CLKTRIM module reset enable. 1: normal operation; 0: module is in reset state
20	RTC	RTC module reset enable. 1: normal operation; 0: module is in reset state
19	PCNT	PCNT module reset enable. 1: normal operation; 0: module is in reset state
18	TRNG	TRNG module reset enable. 1: normal operation; 0: module is in reset state
17	VC	VC, LVD, module reset enable. 1: normal operation; 0: module is in reset state
16	ADC	ADC module reset enable. 1: normal operation; 0: module is in reset state
15	Res.	Reserved bit
14	PCA	PCA module reset enable. 1: normal operation; 0: module is in reset state
13	OPA	OPA module reset enable. 1: normal operation; 0: module is in reset state
12	Res.	Reserved bit
11	TIM3	TIM3 module reset enable. 1: normal operation; 0: module is in reset state
10	ADVTIM	TIM456 module reset enable. 1: normal operation; 0: module is in reset state
9	LPTIM0	LPTIM0 module reset enable. 1: normal operation; 0: module is in reset state
8	BASETIM	TIM012 module reset enable. 1: normal operation; 0: module is in reset state
7	SPI1	SPI1 module reset enable. 1: normal operation; 0: module is in reset state
6	SPI0	SPI0 module reset enable. 1: normal operation; 0: module is in reset state

5	I2C1	I2C1 module reset enable. 1: normal operation; 0: module is in reset state
4	I2C0	I2C0 module reset enable. 1: normal operation; 0: module is in reset state
3	LPUART1	LPUART1 module reset enable. 1: normal operation; 0: module is in reset state
2	LPUART0	LPUART0 module reset enable. 1: normal operation; 0: module is in reset state
1	UART1	UART1 module reset enable. 1: normal operation; 0: module is in reset state
0	UART0	UART0 module reset enable. 1: normal operation; 0: module is in reset state

4.2.3 Peripheral Module Reset Control Register (PERI_RESET1)

Reset value: 0x0000_03FF

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						UART 3	UART 2	RES.	I2S1	I2S0	LPTIM 1	DAC	CTS	CAN	USB
						R/W	R/W		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Address: 0x4000202C

Bit	Marking	Functional description
31:10	Reseved	Reserved bit
9	UART3	UART3 module reset enable. 1: normal operation; 0: module is in reset state
8	UART2	UART2 module reset enable. 1: normal operation; 0: module is in reset state
7	RES.	Reserved bit
6	I2S1	I2S1 module reset enable. 1: normal operation; 0: module is in reset state
5	I2S0	I2S0 module reset enable. 1: normal operation; 0: module is in reset state
4	LPTIM1	LPTIM1 module reset enable. 1: normal operation; 0: module is in reset state
3	DAC	DAC module reset enable. 1: normal operation; 0: module is in reset state
2	CTS	CTS module reset enable. 1: normal operation; 0: module is in reset state
1	CAN	CAN module reset enable. 1: normal operation; 0: module is in reset state
0	USB	USB module reset enable. 1: normal operation; 0: module is in reset state

5 Interrupt Controller (NVIC)

5.1 Overview

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

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

Table 5-1 Cortex-M0+ processor interrupt overview

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

This chapter only introduces the 32 external interrupt requests of the processor (interrupt #0 to interrupt #31) in detail, and the specific situation of the internal exception of the processor can refer to other related documents. At the same time, this chapter only discusses the interrupt handling mechanism of the NVIC in the processor core, and the interrupt generation mechanism of the peripheral module itself is not discussed here.

5.2 Interrupt priority

Each external interrupt corresponds to a priority register, each priority is 2 bits wide, and uses the highest two bits of the interrupt priority register, and each register occupies 1 byte (8 bits). Under this setting, the available priorities are 0x00 (highest), 0x40, 0x80 and 0xc0 (lowest).

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

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

Preemption occurs when the processor is already running another interrupt handler and the new interrupt has a higher priority than the one currently being executed. The running interrupt processing will be suspended and the new interrupt will be executed instead. This process is usually called interrupt nesting. After the new interrupt is executed, the previous interrupt processing will continue and return to the program thread after it finishes.

If the processor is running another interrupt handler with the same or higher priority, the new interrupt will wait and enter the pending state. Pending interrupts will wait until the current interrupt level changes, for example, after the currently running interrupt handler finishes returning, the current priority is lowered to be lower than the pending interrupt.

If two interrupts occur at the same time and they have the same priority, the interrupt with the lower interrupt number will be executed first. For example, if interrupt #0 and interrupt #1 are enabled and have the same priority, when they are triggered at the same time, interrupt #0 will be executed first.

5.3 Interrupt digraph

When the Cortex-M0+ processor processes an interrupt service request, it needs to first determine the starting address of the exception handling. The required information is called a vector table, as shown in Figure 5-2. The vector table is stored at the beginning of the memory space and contains the exception (interrupt) vectors of the exceptions (interrupts) available in the system, as well as the initial value of the main stack pointer (MSP).

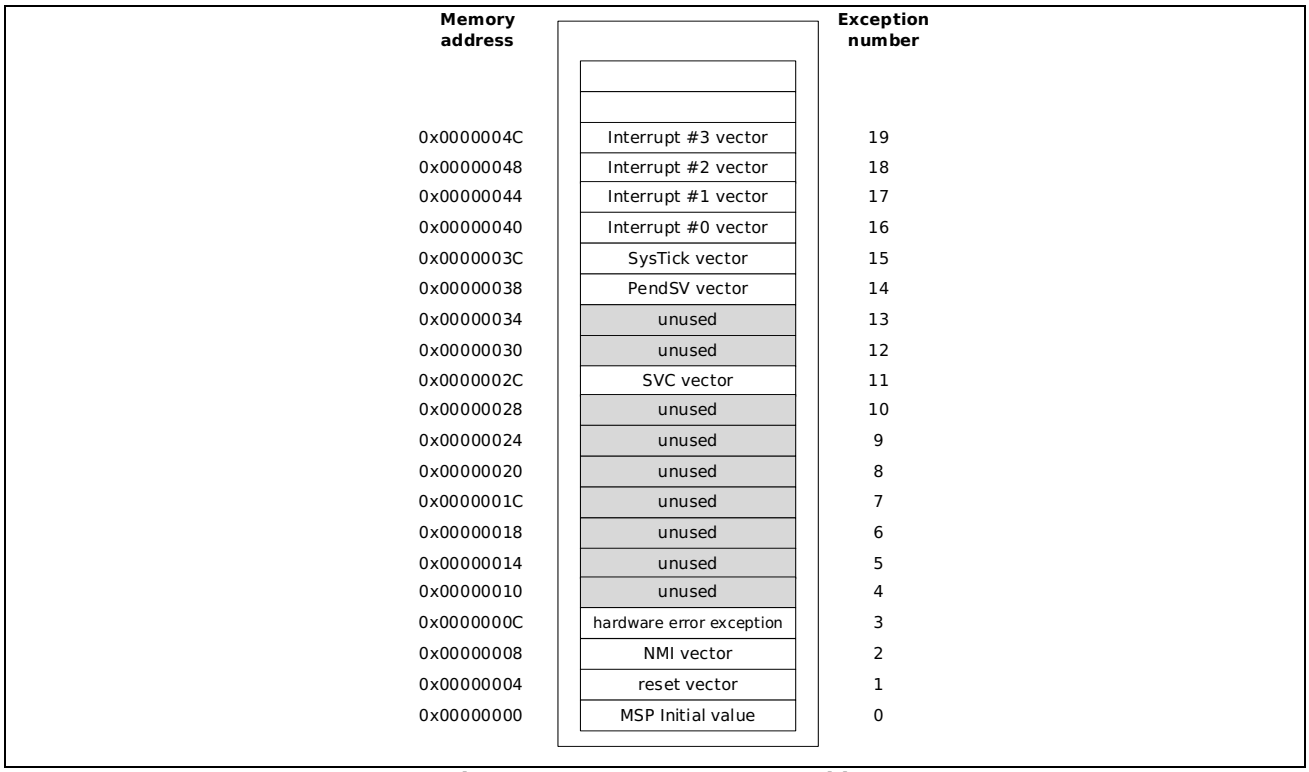


Figure 5-2 Interrupt Vector Table

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

5.4 Interrupt Input and Suspend Behavior

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

The peripherals of this system use level-triggered interrupt output. When an interrupt event occurs, the interrupt signal will be confirmed because the peripheral is connected to the NVIC. This signal remains high until the processor executes the interrupt service and clears the peripheral's interrupt signal. Inside the NVIC, when an interrupt is detected, the pending status of the interrupt will be set, and when the processor receives the interrupt and starts executing the interrupt service routine, the pending status will be cleared. The process is shown in Figure 5-3:

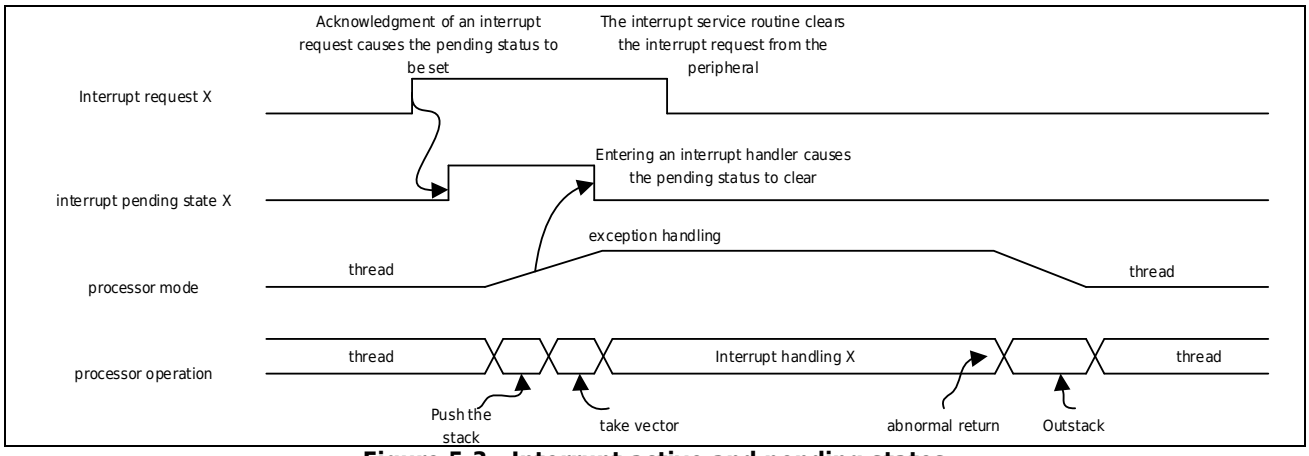


Figure 5-3 Interrupt active and pending states

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

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

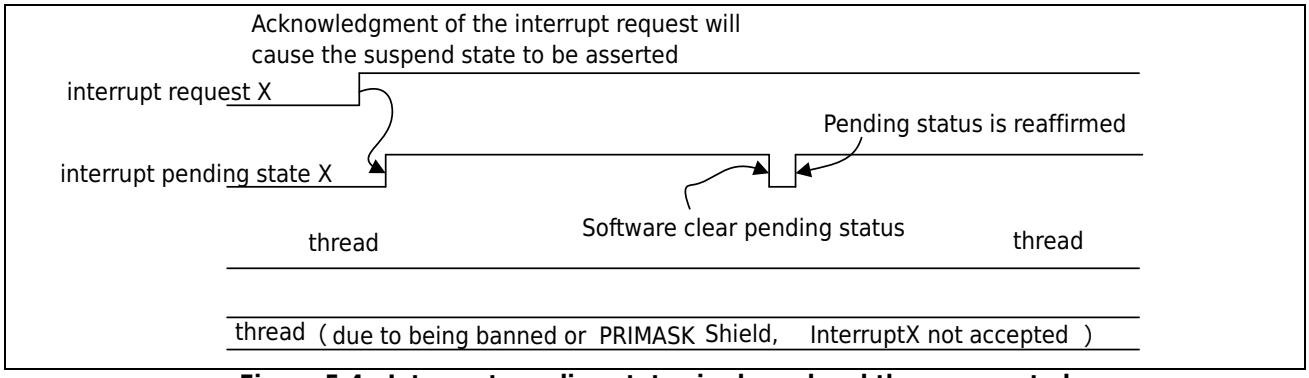


Figure 5-4 Interrupt pending status is cleared and then reasserted

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

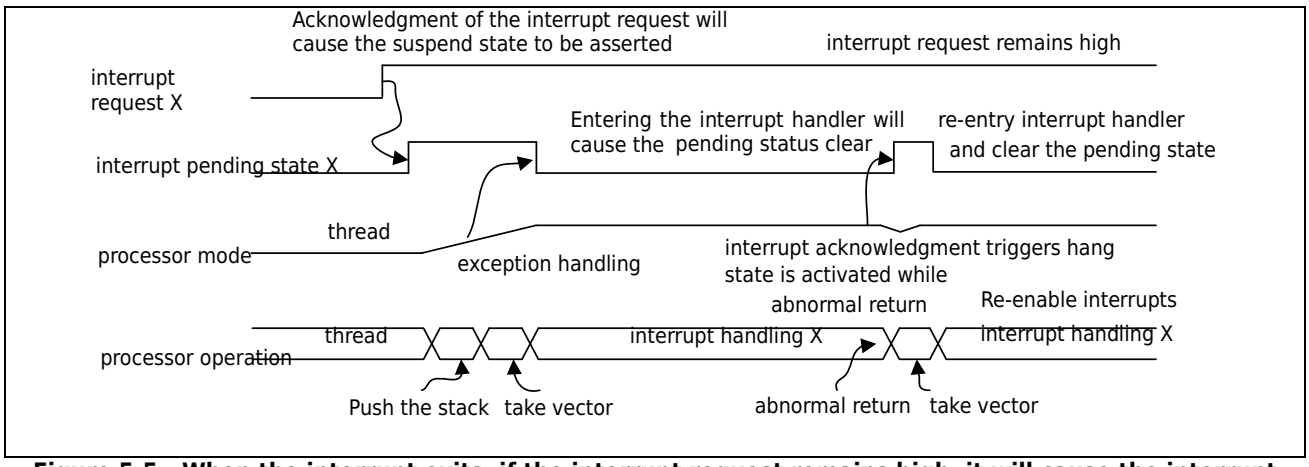


Figure 5-5 When the interrupt exits, if the interrupt request remains high, it will cause the interrupt processing to be executed again

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

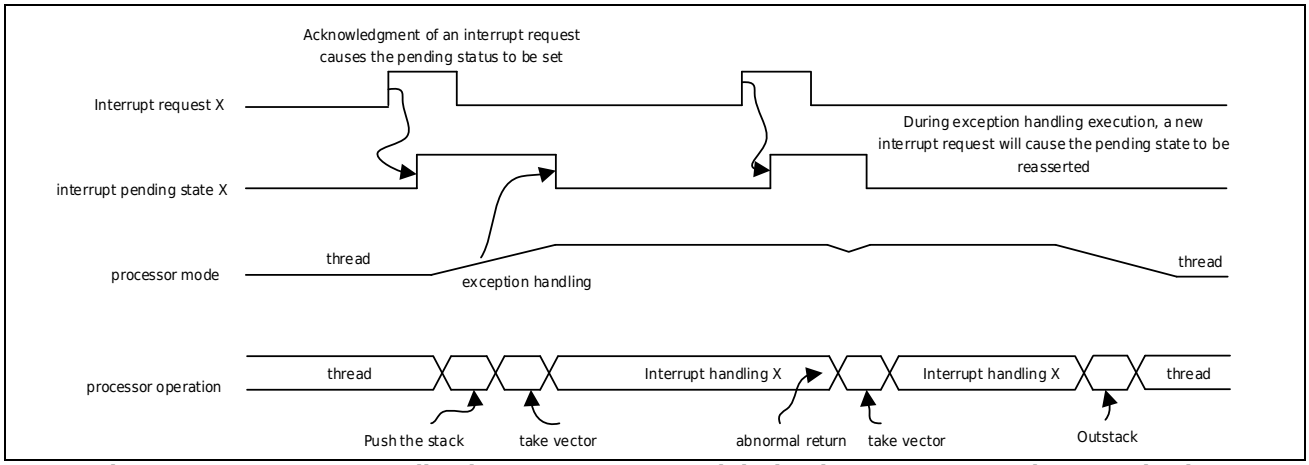


Figure 5-6 Interrupt pending interrupts generated during interrupt processing can also be acknowledged

5.5 Interrupt wait

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

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

The following situations may cause different interrupt waits:

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

5.6 Interrupt source

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

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

NVIC interrupt input	External interrupt source	Active mode	Sleep mode	DeepSleep mode
Interrupt #0	PORTA	✓	✓	✓
Interrupt #1	PORTB	✓	✓	✓
Interrupt #2	PORTC/PORTE	✓	✓	✓
Interrupt #3	PORTD/PORTF	✓	✓	✓
Interrupt #4	DMAC	✓	✓	-
Interrupt #5	TIM3	✓	✓	-
Interrupt #6	UART0/2	✓	✓	-
Interrupt #7	UART1/3	✓	✓	-
Interrupt #8	LPUART0	✓	✓	✓
Interrupt #9	LPUART1	✓	✓	✓
Interrupt #10	SPI0/I2S0	✓	✓	-
Interrupt #11	SPI1/I2S1	✓	✓	-

NVIC interrupt input	External interrupt source	Active mode	Sleep mode	DeepSleep mode
Interrupt #12	I2C0	✓	✓	-
Interrupt #13	I2C1	✓	✓	-
Interrupt #14	TIM0	✓	✓	-
Interrupt #15	TIM1	✓	✓	-
Interrupt #16	TIM2	✓	✓	-
Interrupt #17	LPTIM0/1	✓	✓	✓
Interrupt #18	TIM4	✓	✓	-
Interrupt #19	TIM5	✓	✓	-
Interrupt #20	TIM6	✓	✓	-
Interrupt #21	PCA	✓	✓	-
Interrupt #22	WDT	✓	✓	✓
Interrupt #23	RTC	✓	✓	✓
Interrupt #24	ADC/DAC	✓	✓	-
Interrupt #25	PCNT	✓	✓	✓
Interrupt #26	VC0/VC1/VC2/LVD	✓	✓	✓
Interrupt #27	USB	✓	✓	
Interrupt #28	CAN	✓	✓	
Interrupt #29	LCD	✓	✓	✓
Interrupt #30	FLASH/RAM	✓	✓	-
Interrupt #31	CLKTRIM/CTS	✓	✓	✓

Note:

– Because some module interrupts are reused for the same IRQ interrupt source, when the CPU enters the interrupt operation, it must first determine which module generated the interrupt, and then perform the corresponding interrupt operation.

5.7 Interrupt structure diagram

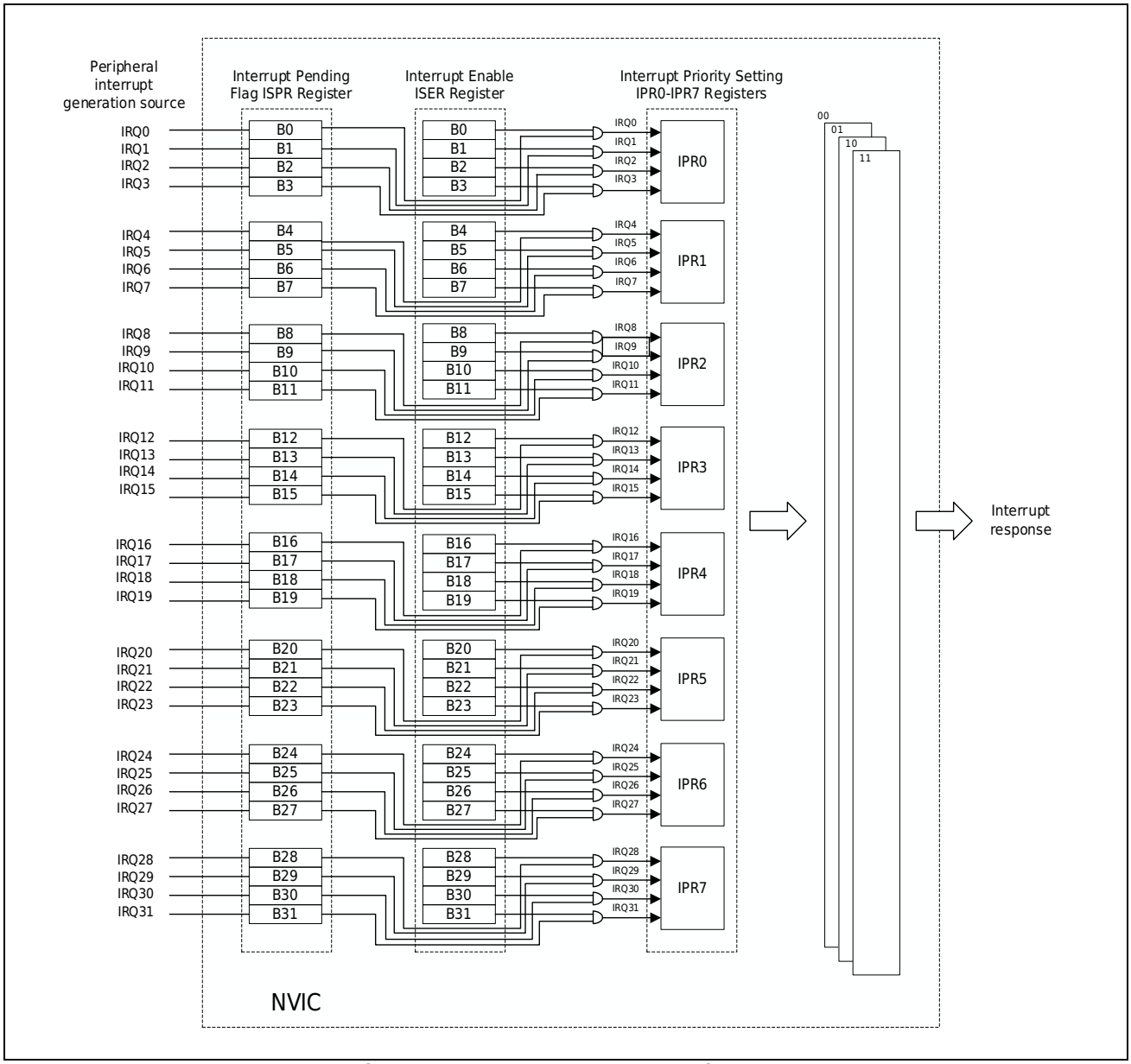


Figure 5-7 Interrupt Structure Diagram

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

- The respective interrupt enables of the peripheral interrupt sources are not marked in the figure, and only the logic block diagram of the interrupt signal after the peripheral interrupt is generated is included here.
- IRQ has more than one peripheral interrupt input multiplexing, and the interrupt flag bits of these peripherals must be read separately to determine which peripheral interrupt it is.
- If the peripheral interrupt source has a high level, regardless of whether the NVIC interrupt enable register SCS_SETENA is set or not, the interrupt pending register SCS_SEPEND will be set, indicating that the corresponding peripheral interrupt source has an interrupt.

- Only when the interrupt enable register NVIC_ISER is set, the corresponding interrupt IRQ will respond to the processor and execute the corresponding interrupt program.
- In the interrupt program, the high-level interrupt signal of the peripheral interrupt source must be cleared, and the interrupt pending register NVIC_ISPR is automatically cleared by the hardware.
- The interrupt priority registers NVIC_IPR0-NVIC_IPR7 set the priority of 32 interrupt sources, 00 has the highest priority and 11 has the lowest priority. When the priority is the same, the priority is determined by the interrupt number, and the smaller the number, the higher the priority.

5.8 Register

Base address: 0xE000 E000

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

5.8.1 Interrupt Enable Setting Register (NVIC_ISER)

Offset address: 0x100

Reset value: 0x0000 0000

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

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

5.8.2 Interrupt Enable Clear Register (NVIC_ICER)

Offset address: 0x180

Reset value: 0x0000 0000

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

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

5.8.3 Interrupt Pending Status Set Register (NVIC_ISPR)

Offset address: 0x200

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SETPEND[31:16]															
RW															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETPEND[15:0]															
RW															

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

5.8.4 Interrupt Pending Status Clear Register (NVIC_ICPR)

Offset address: 0x280

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLRPEND[31:16]															
RW															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRPEND[15:0]															
RW															

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

5.8.5 Interrupt Priority Register (NVIC_IPR0)

Offset address: 0x400

Reset value: 0x0000 0000

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

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

5.8.6 Interrupt Priority Register (NVIC_IPR1)

Offset address: 0x404

Reset value: 0x0000 0000

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

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

5.8.7 Interrupt Priority Register (NVIC_IPR2)

Offset address: 0x408

Reset value: 0x0000 0000

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

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

5.8.8 Interrupt Priority Register (NVIC_IPR3)

Offset address: 0x40C

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

Reset value: 0x0000 0000

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

5.8.9 Interrupt Priority Register (NVIC_IPR4)

Offset address: 0x410

Reset value: 0x0000 0000

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

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

5.8.10 Interrupt Priority Register (NVIC_IPR5)

Offset address: 0x414

Reset value: 0x0000 0000

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

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

5.8.11 Interrupt Priority Register (NVIC_IPR6)

Offset address: 0x418

Reset value: 0x0000 0000

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

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

5.8.12 Interrupt Priority Register (NVIC_IPR7)

Offset address: 0x41C

Reset value: 0x0000 0000

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

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

5.8.13 Interrupt Mask Special Register (PRIMASK)

Offset address: ---

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															PRIMASK
															RW

Bit	Symbol	Description
31:1	Reserved	
0	PRIMASK	When set, all interrupts except NMI and hardware error exceptions will be masked After clearing, all exceptions and interrupts will not be masked The special register needs to be accessed through MSR and MRS special register operation instructions, and can also be accessed by changing the processor state instruction CPS. When dealing with time-sensitive applications, it is necessary to manipulate the PRIMASK register.

5.9 Basic operation of the software

5.9.1 External interrupt enable

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

5.9.2 NVIC interrupt enable and clear enable

The Cortex-M0+ processor supports up to 32 interrupt sources, and each interrupt source corresponds to an interrupt enable bit and a clear enable bit. In this way, there are 32- bit interrupt enable register NVIC_ISER and 32 -bit clear enable register NVIC_ICER. If you want to enable an interrupt, set the corresponding bit of the NVIC_ISER register to 1. If you want to clear an interrupt, set the corresponding bit in the NVIC_ICER register to 1.

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

5.9.3 NVIC interrupt pending and clear pending

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

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

You can access or modify the interrupt pending status by manipulating the interrupt setting pending NVIC_ISPR and interrupt clearing pending NVIC_ICPR registers. The Interrupt Pending Status Register allows software to trigger interrupts.

5.9.4 NVIC interrupt priority

Setting the NVIC_IPR0-NVIC_IPR7 registers determines the priority of SCS_IRQ0-SCS_IRQ32. The programming of the interrupt priority register should be done before the interrupt is enabled, which is usually done at the beginning of the program. Changing the interrupt priority after the interrupt is enabled should be avoided, the result of this situation is unpredictable and is not supported by the Cortex-M0+ processor.

5.9.5 NVIC interrupt mask

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

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

```
void __enable_irq(void);    // Clear PRIMASK
void __disable_irq(void);    // Set PRIMASK
```

6 Port Controller (GPIO)

6.1 Introduction to Port Controllers

This series has 86 digital general-purpose input and output ports PA[15:0], PB[15:0], PC[15:0], PD[15:0], PE[15:0], PF[11:9], PF[7:0]. All unpackaged GPIOs should also be configured as inputs with pull-ups enabled.

The input and output signals of the analog module ADC/VC/LVD/LCD, the input and output signals of each functional module (such as SPI, UART, I2C, Timer, etc.) and digital general-purpose input and output ports are multiplexed.

Each port can be configured as internal pull-up (pull up) / pull-down (pull down) input, high-impedance input (floating input), push-pull output (CMOS output), open drain output (open drain output), two levels drive capability output. "1" and "0" cannot be output, and the result of reading the port input value register by the CPU is "0".

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

After the chip is reset, the port is a high-impedance input (floating input), the purpose is to prevent abnormal actions on external devices when the chip is abnormally reset. However, in order to avoid the leakage caused by high-impedance input, the user should configure the port accordingly after the chip is started (configured as internal pull-up/ pull -down input or output).

6.2 Port Controller Main Features

The port controller supports the following features:

- Port input value / output value register supports FAST IO/AHB bus read and write
- Other registers support AHB bus interface read and write
- Analog function pins / digital common pins / digital function pins are multiplexed
- Support pull-up / pull-down / two-speed drive / open-drain output function selection
- Support interrupt in working mode/sleep mode/deep sleep mode
- Support high level / low level / rising edge / falling edge trigger interrupt
- Support bit set, bit clear, bit clear function

Note: For the introduction of FAST IO, please refer to ARM Cortex-M0+_IntegrationAndImplementationManual.pdf.

6.3 Port Controller Functional Description

6.3.1 Port configuration function

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

The port circuit structure is shown in the figure below:

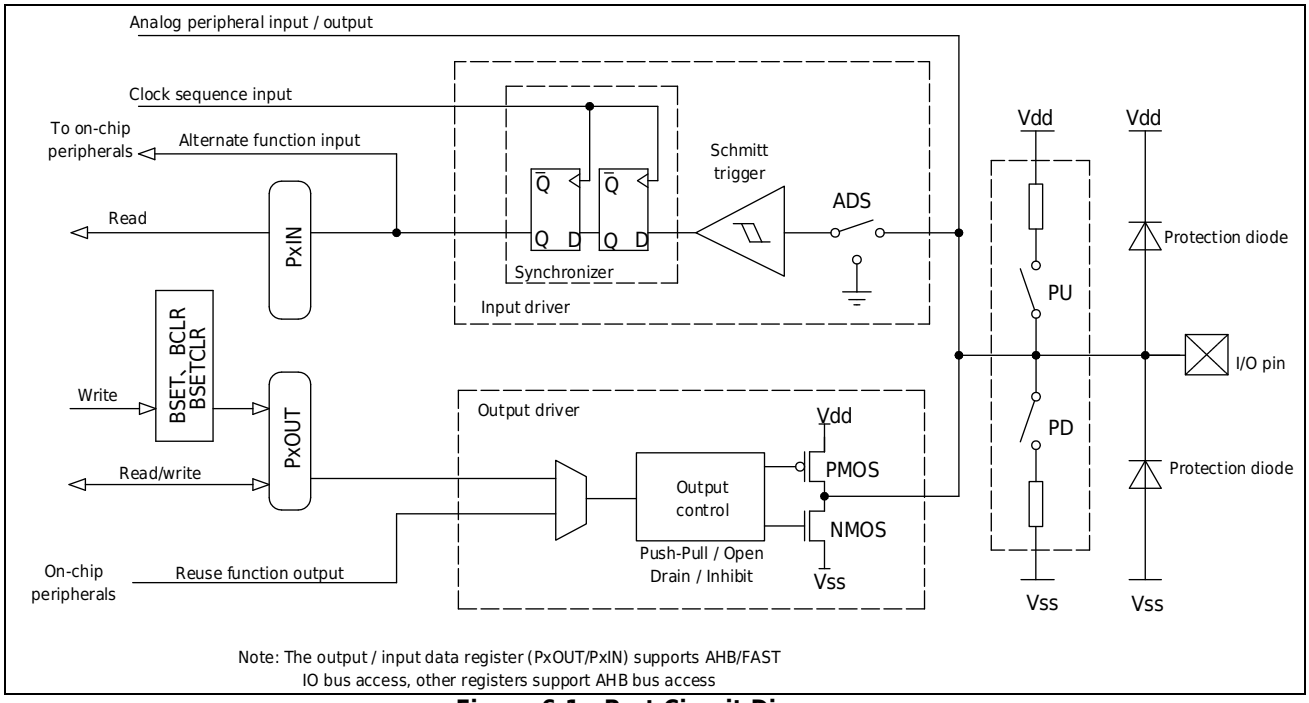


Figure 6-1 Port Circuit Diagram

When the port is configured as a digital port, it can accept digital general-purpose input and output signals (set the Px_SEL register to '0'), or accept the input and output signals of various functional modules (such as SPI, UART, I2C, Timer, etc.) through the configuration register Px_SEL, For details, refer to the port digital multiplexing function module.

You can also configure the corresponding registers to achieve the following features:

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

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

2) Two-speed drive output (PxDR)

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

3) Open Drain Output (PxOD)

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

4) Direction selection (PxDIR)

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

5) Input level status (PxIN)

The synchronized pin level can be obtained by reading the PxIN register. When PxIN is '1', it is high level, and when PxIN is '0', it is low level. Accessible via AHB/FAST IO bus

6) Output high- and low-level selection (PxOUT)

When the port pin is configured as an output, if PxOUT is '1', the port pin output is high level, if PxOUT is '0', the output is low level. Accessible via AHB/FAST IO bus

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

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

Note:

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

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

Table 6-1 Port State Truth Table

IO state	IO direction	PxADS	PxDIR	PxOUT	PxIN	PxBSET	PxBCLR	PxBSETCLR	PxPU	PxPD	PxOD	PxDR	Px_SEL
Simulation	Input/Output	1	W	W	0	W	W	W	W	W	W	W	W
Floating	Input	0	1	W	X	W	W	W	0	0	W	W	0
Drop down	Input	0	1	W	0	W	W	W	0	1	W	W	0
Pull-up	Input	0	1	W	1	W	W	W	1	0	W	W	0
Pull-up	Input	0	1	W	1	W	W	W	1	1	W	W	0
1	Input	0	1	W	1	W	W	W	W	W	W	W	0
0	Input	0	1	W	0	W	W	W	W	W	W	W	0
1	Output	0	0	1	1	0	0	0	W	W	0	W	0
0	Output	0	0	0	0	0	0	0	W	W	0	W	0
1	Output	0	0	W	1	1	0	0	W	W	0	W	0
0	Output	0	0	W	0	0	1	0	W	W	0	W	0
(SET) 1	Output	0	0	W	(SET) 1	0	0	1	W	W	0	W	0
(CLR) 0	Output	0	0	W	(CLR) 0	0	0	1	W	W	0	W	0
0	Output	0	0	0	0	0	0	0	W	W	1	W	0
Z	Output	0	0	1	X	0	0	0	0	0	1	W	0
0	Output	0	0	1	0	0	0	0	0	1	1	W	0
1	Output	0	0	1	1	0	0	0	1	0	1	W	0
1	Output	0	0	1	1	0	0	0	1	1	1	W	0
Note:	0 - Logic low		1 - Logic high		W - Whatever 0 or 1		X - unknow state		Z - high impedance				

6.3.2 Port write

The port input value / output value register (PxIN/PxOUT) supports reading and writing of the AHB bus and the FAST IO bus (controlled by the register GPIO_CTRL2.ahb_sel bit), and only supports reading and writing of the AHB bus for other registers. For these two different buses, the system clock (HCLK) has different processing cycles for these two buses. The following two figures show the fastest timing for two bus port flips:

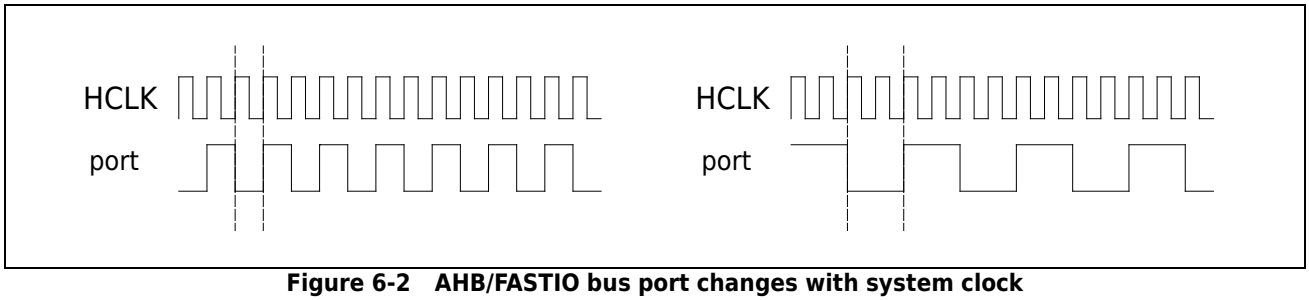


Figure 6-2 AHB/FASTIO bus port changes with system clock

(Left is FAST IO, right is AHB)

For the AHB bus, every two HCLK cycles, the IO flips once, and for the FAST IO bus, every HCLK cycle, the IO flips once.

6.3.3 port read

Each port can get the port pin level by reading the PxIN register. As shown in Figure 6-1, each bit of the PxIN register and its preceding latch form a synchronizer to avoid signal instability caused by pin level changes in a short period of time when the system clock state changes, but also introduces delay. The synchronization diagram for reading port pin data is as follows:

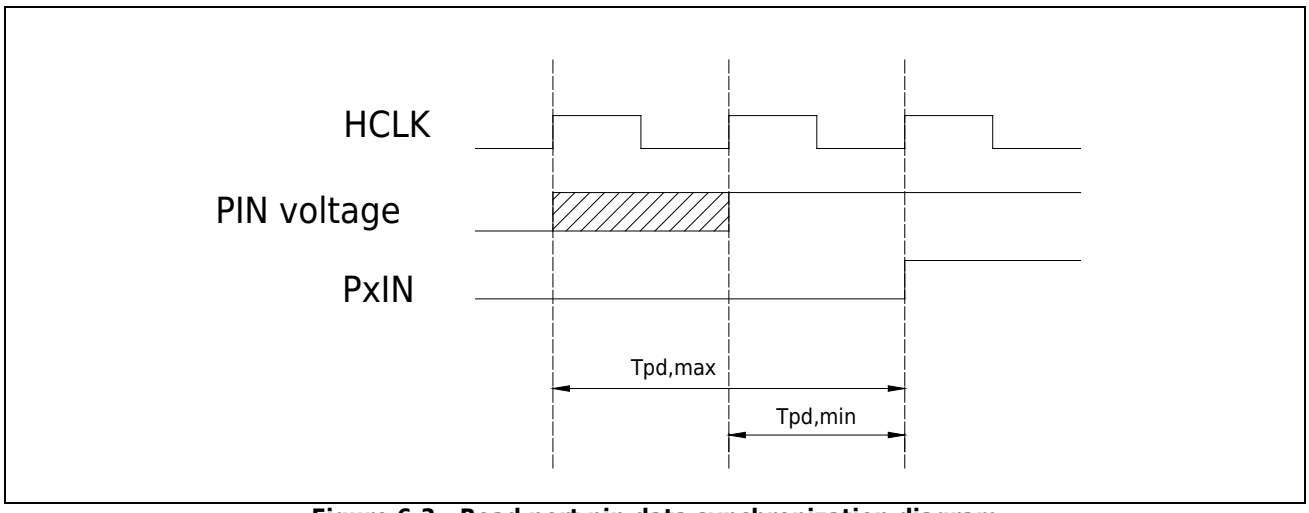


Figure 6-3 Read port pin data synchronization diagram

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

Note:

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

6.3.4 Port digital multiplexing function

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

As shown in the table below: The Px_SEL register is used for digital general-purpose port / digital function port switching, and each port can be independently configured as a functional port required by the system. For the configuration information of the debug port, please refer to the relevant chapters.

PxSEL							
0	1	2	3	4	5	6	7
PA00	UART1_CTS	LPUART1_TXD	TIM0_ETR	VC0_OUT	TIM1_CHA	TIM3_ETR	TIM0_CHA
PA01	UART1_RTS	LPUART1_RXD	TIM0_CHB	TIM1_ETR	TIM1_CHB	HCLK_OUT	SPI1_MOSI
PA02	UART1_TXD	TIM0_CHA	VC1_OUT	TIM1_CHA	TIM2_CHA	PCLK_OUT	SPI1_MISO
PA03	UART1_RXD	TIM0_GATE	TIM1_CHB	TIM2_CHB	SPI1_CS	TIM3_CH1A	TIM5_CHA
PA04	SPI0_CS	UART1_TXD	PCA_CH4	TIM2_ETR	TIM5_CHA	LVD_OUT	TIM3_CH2B
PA05	SPI0_SCK	TIM0_ETR	PCA_ECI	TIM0_CHA	TIM5_CHB	XTL_OUT	XTH_OUT
PA06	SPI0_MISO	PCA_CH0	TIM3_BK	TIM1_CHA	VC0_OUT	TIM3_GATE	LPUART0_CTS
PA07	SPI0_MOSI	PCA_CH1	HCLK_OUT	TIM3_CH0B	TIM2_CHA	VC1_OUT	TIM4_CHB
PA08	UART0_TXD	TIM3_CH0A	CRS_SYNC	CAN_STBY	TIM1_GATE	TIM4_CHA	TIM3_BK
PA09	UART0_TXD	TIM3_CH1A	TIM0_BK	I2C0_SCL		HCLK_OUT	TIM5_CHA
PA10	UART0_RXD	TIM3_CH2A	TIM2_BK	I2C0_SDA	TIM2_GATE	PCLK_OUT	TIM6_CHA
PA11	UART0_CTS	TIM3_GATE	I2C1_SCL	CAN_RX	VC0_OUT	SPI0_MISO	TIM4_CHB
PA12	UART0_RTS	TIM3_ETR	I2C1_SDA	CAN_TX	VC1_OUT	SPI0_MOSI	PCNT_S0
PA13	IR_OUT	UART0_RXD	LVD_OUT	TIM3_ETR	RTC_1HZ	PCNT_S1	VC2_OUT
PA14	UART1_TXD	UART0_TXD	TIM3_CH2A	LVD_OUT	RCH_OUT	RCL_OUT	PLL_OUT
PA15	SPI0_CS	UART1_RXD	LPUART1_RTS	TIM0_ETR	TIM0_CHA	TIM3_CH1A	
PB00	PCA_CH2	TIM3_CH1B	LPUART0_TXD	TIM5_CHB	RCH_OUT	RCL_OUT	PLL_OUT
PB01	PCA_CH3	PCLK_OUT	TIM3_CH2B	TIM6_CHB	LPUART0_RTS	VC2_OUT	TCLK_OUT
PB02	LPTIM0_TOG	PCA_ECI	LPUART1_TXD	TIM4_CHA	TIM1_BK	TIM0_BK	TIM2_BK
PB03	SPI0_SCK	TIM0_CHB	TIM1_GATE	TIM3_CH0A	LPTIM0_GATE	XTL_OUT	XTH_OUT
PB04	SPI0_MISO	PCA_CH0	TIM2_BK	UART0_CTS	TIM2_GATE	TIM3_CH0B	LPTIM0_EXT
PB05	SPI0_MOSI		TIM1_BK	PCA_CH1	LPTIM0_GATE	PCNT_S0	UART0_RTS
PB06	I2C0_SCL	UART0_TXD	TIM1_CHB	TIM0_CHA	LPTIM0_EXT	TIM3_CH0A	LPTIM0_TOG
PB07	I2C0_SDA	UART0_RXD	TIM2_CHB	LPUART1_CTS	TIM0_CHB	LPTIM0_TOGN	PCNT_S1
PB08	I2C0_SCL	TIM1_CHA	CAN_RX	TIM2_CHA	TIM0_GATE	TIM3_CH2A	UART0_TXD
PB09	I2C0_SDA	IR_OUT	SPI1_CS	TIM2_CHA	CAN_TX	TIM2_CHB	UART0_RXD
PB10	I2C1_SCL	SPI1_SCK	TIM1_CHA	LPUART0_TXD	TIM3_CH1A	LPUART1_RTS	UART1_RTS
PB11	I2C1_SDA	TIM1_CHB	LPUART0_RXD	TIM2_GATE	TIM6_CHA	LPUART1_CTS	UART1_CTS

PxSEL							
0	1	2	3	4	5	6	7
PB12	SPI1_CS	TIM3_BK	LPUART0_TXD	TIM0_BK		LPUART0_RTS	TIM6_CHA
PB13	SPI1_SCK	I2C1_SCL	TIM3_CH0B	LPUART0_CTS	TIM1_CHA	TIM1_GATE	TIM6_CHB
PB14	SPI1_MISO	I2C1_SDA	TIM3_CH1B	TIM0_CHA	RTC_1HZ	LPUART0_RTS	TIM1_BK
PB15	SPI1_MOSI	TIM3_CH2B	TIM0_CHB	TIM0_GATE			LPUART1_RXD
PC00	LPTIM0_GATE	PCNT_S0	UART1_CTS	UART2_RTS	I2S0_MCK		
PC01	LPTIM0_TOG	TIM5_CHB	UART1_RTS	PCNT_S0FO	I2S0_SD	UART2_CTS	
PC02	SPI1_MISO	LPTIM0_TOGN	PCNT_S1	UART2_RXD			
PC03	SPI1_MOSI	LPTIM0_EXT	LPTIM0_TOGN	PCNT_S1FO	UART2_TXD		
PC04	LPUART0_TXD	TIM2_ETR	IR_OUT	VC2_OUT	I2S0_WS		
PC05	LPUART0_RXD	TIM6_CHB	PCA_CH4		I2S0_SDIN		
PC06	PCA_CH0	TIM4_CHA	TIM2_CHA	LPTIM1_GATE	I2S1_SCK	UART3_RXD	
PC07	PCA_CH1	TIM5_CHA	TIM2_CHB	LPTIM1_EXT	I2S1_MCK	UART3_TXD	
PC08	PCA_CH2	TIM6_CHA	TIM2_ETR	LPTIM1_TOG	I2S1_SD	UART3_CTS	
PC09	PCA_CH3	TIM4_CHB	TIM1_ETR	LPTIM1_TOGN	I2S1_WS	UART3_RTS	
PC10	LPUART1_TXD	LPUART0_TXD	PCA_CH2				
PC11	LPUART1_RXD	LPUART0_RXD	PCA_CH3	PCNT_S0FO			
PC12	LPUART0_TXD	LPUART1_TXD	PCA_CH4	PCNT_S1FO			
PC13		RTC_1HZ	TIM3_CH1B		I2S0_SCK		
PC14							
PC15							
PD00	CAN_RX	SPI1_CS					
PD01	CAN_TX	SPI1_SCK					
PD02	PCA_ECI	LPUART0_RTS	TIM1_ETR				
PD03	UART1_CTS	SPI1_MISO	LPTIM1_TOG	I2S1_SCK			
PD04	UART1_RTS	SPI1_MOSI	LPTIM1_TOGN	I2S1_MCK			
PD05	UART1_TXD	LPTIM1_GATE	CAN_STBY	I2S1_SD			
PD06	UART1_RXD	LPTIM1_EXT		I2S1_WS			
PD07	UART1_TXD			I2S1_SDIN			
PD08	LPUART0_TXD	I2S0_SCK					
PD09	LPUART0_RXD	I2S0_MCK					
PD10	LPUART0_TXD	I2S0_SD					
PD11	LPUART0_CTS	I2S0_WS					
PD12	LPUART0_RTS	UART2_RTS					
PD13	UART2_RXD	I2S0_SDIN					
PD14	UART2_TXD						
PD15	CRS_SYNC	UART2_CTS					
PE00	TIM1_CHA						

PxSEL							
0	1	2	3	4	5	6	7
PE01	TIM2_CHA						
PE02	PCA_ECI						
PE03	PCA_CH0						
PE04	PCA_CH1						
PE05	PCA_CH2						
PE06	PCA_CH3						
PE07	TIM3_ETR	LPTIM1_GATE					
PE08	TIM3_CH0B	LPTIM1_EXT					
PE09	TIM3_CH0A	LPTIM1_TOG					
PE10	TIM3_CH1B	LPTIM1_TOGN					
PE11	TIM3_CH1A						
PE12	TIM3_CH2B	SPI0_CS	UART3_CTS				
PE13	TIM3_CH2A	SPI0_SCK	UART3_RTS				
PE14	TIM3_CH0B	SPI0_MISO	UART3_RXD				
PE15	TIM3_BK	SPI0_MOSI	UART3_TXD				
PF00	I2C0_SDA	CRS_SYNC	UART1_TXD				
PF01	I2C0_SCL		UART1_RXD				
PF02							
PF03							
PF04							
PF05							
PF06	I2C1_SCL	LPUART1_CTS	UART0_CTS				
PF07	I2C1_SDA	LPUART1_RTS	UART0_RTS				
PF09	TIM0_CHA						
PF10	TIM0_CHB						
PF11							

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

6.4 Port configuration operation flow

6.4.1 Port multiplexing is configured as an analog port operation flow

a) Set register PxADS[n] to 1

6.4.2 Port multiplexing configured as a digital universal port operation flow

a) Set register PxADS[n] to 0
b) Set register Px_SEL to 0
c) Set the register PxDIR[n] to 1: the port direction is input, and the CPU can read the port status PxIN[n].
d) Set the register PxDIR[n] to 0: port direction is output
e) Set the register PxOUT[n] to 1: the port outputs a high level
f) Set the register PxOUT[n] to 0: port output low level

6.4.3 Port multiplexing configured as a digital function port operation flow

a) Set register PxADS[n] to 0
b) Set the register Px_SEL to 1~7 (according to the system requirements, refer to the port multiplexing table)
c) Set register PxDIR[n] (according to system requirements)
d) Set registers PxPU[n]/PxPD[n]/PxOD[n] (according to system requirements)

6.4.4 Port multiplexing is configured as a debug test port operation process

Refer to the chapters related to testing and debugging.

6.4.5 Port multiplexing configured as infrared output signal operation process

Ports PA13, PB09, and PC04 can modulate the internal clock signal with a frequency of 38K into infrared signal output.

a) Set register PAADS[13]/PBADS[9]/PCADS[4] to 0
b) Set register PA13_SEL = 1/PB09_SEL = 2/PC04_SEL = 3
c) Set register PADIR[13] = 0/PBDIR[9] = 0/PCDI R[4] = 0: Port direction is output
d) Set the bit14 of the register GPIO_CTRL1 to select the output polarity of the infrared signal
e) Set the output of the register PAOUT[13]/PBOUT[9]/PCOUT[4] to control the infrared signal

6.4.6 Port high level interrupt operation flow

a) Set register PxADS[n] to 0
b) Set register Px_SEL to 0
c) Set register PxDIR[n] to 1
d) Set register Px_HIE[n] to 1

e) Read the interrupt status register Px_STAT[n] after the interrupt is triggered

f) Set the register Px_ICLR[n] to 0 to clear the interrupt status register Px_STAT[n]

6.4.7 Port low level interrupt operation flow

a) Set register PxADS[n] to 0

b) Set register Px_SEL to 0

c) Set register PxDIR[n] to 1

d) Set register Px_LIE[n] to 1

e) Read the interrupt status register Px_STAT[n] after the interrupt is triggered

f) Set the register Px_ICLR[n] to 0 to clear the interrupt status register Px_STAT[n]

6.4.8 Port rising edge interrupt operation flow

a) Set register PxADS[n] to 0

b) Set register Px_SEL to 0

c) Set register PxDIR[n] to 1

d) Set register Px_RIE[n] to 1

e) Read the interrupt status register Px_STAT[n] after the interrupt is triggered

f) Set the register Px_ICLR[n] to 0 to clear the interrupt status register Px_STAT[n]

6.4.9 Port falling edge interrupt operation flow

a) Set register PxADS[n] to 0

b) Set register Px_SEL to 0

c) Set register PxDIR[n] to 1

d) Set register Px_FIE[n] to 1

e) Read the interrupt status register Px_STAT[n] after the interrupt is triggered

f) Set the register Px_ICLR[n] to 0 to clear the interrupt status register Px_STAT[n]

6.4.10 Port pull-up enable configuration operation flow

a) Set register PxPU[n] to 1

6.4.11 Port pull-down enable configuration operation flow

a) Set register PxPU[n] to 0

b) Set register PxPD[n] to 1

Note: When PxPU[n] and PxPD[n] are set to 1 at the same time, PxPU[n] has higher priority and PxPD[n] is invalid.

6.4.12 Port Enhancement Driver Configuration Operation Flow

a) Set register PxDR[n] to 0

6.4.13 Port open-drain output configuration operation flow

a) Set register PxOD[n] to 1

6.4.14 Port bit setting operation flow

a) Set register PxBSET[n] to 1

6.4.15 Port bit clearing operation flow

a) Set register PxBCLR[n] to 1

6.4.16 Operation flow of port bit setting and clearing

a) Set register PxBSETCLR[n] to 1

6.5 Port Controller Register Description

Register List 1

Base address 1: 0x40020C00

Offset	Register name	Access	Register description
0x00	PA00_SEL	RW	Port PA00 Function configuration register
0x04	PA01_SEL	RW	Port PA01 Function Configuration Register
0x08	PA02_SEL	RW	Port PA02 Function Configuration Register
0x0c	PA03_SEL	RW	Port PA03 Function Configuration Register
0x10	PA04_SEL	RW	Port PA04 Function Configuration Register
0x14	PA05_SEL	RW	Port PA05 Function Configuration Register
0x18	PA06_SEL	RW	Port PA06 Function Configuration Register
0x1c	PA07_SEL	RW	Port PA07 Function Configuration Register
0x20	PA08_SEL	RW	Port PA08 Function Configuration Register
0x24	PA09_SEL	RW	Port PA09 Function Configuration Register
0x28	PA10_SEL	RW	Port PA10 Function Configuration Register
0x2c	PA11_SEL	RW	Port PA11 Function Configuration Register
0x30	PA12_SEL	RW	Port PA12 Function Configuration Register
0x34	PA13_SEL	RW	Port PA13 Function Configuration Register
0x38	PA14_SEL	RW	Port PA14 Function Configuration Register
0x3c	PA15_SEL	RW	Port PA15 Function Configuration Register
0x40	PB00_SEL	RW	Port PB00 Function Configuration Register
0x44	PB01_SEL	RW	Port PB01 Function Configuration Register
0x48	PB02_SEL	RW	Port PB02 Function Configuration Register
0x4c	PB03_SEL	RW	Port PB03 Function Configuration Register
0x50	PB04_SEL	RW	Port PB04 Function Configuration Register
0x54	PB05_SEL	RW	Port PB05 Function Configuration Register
0x58	PB06_SEL	RW	Port PB06 Function Configuration Register
0x5c	PB07_SEL	RW	Port PB07 Function Configuration Register
0x60	PB08_SEL	RW	Port PB08 Function Configuration Register
0x64	PB09_SEL	RW	Port PB09 Function Configuration Register
0x68	PB10_SEL	RW	Port PB10 Function Configuration Register
0x6c	PB11_SEL	RW	Port PB11 Function Configuration Register
0x70	PB12_SEL	RW	Port PB12 Function Configuration Register
0x74	PB13_SEL	RW	Port PB13 Function Configuration Register
0x78	PB14_SEL	RW	Port PB14 Function Configuration Register
0x7c	PB15_SEL	RW	Port PB15 Function Configuration Register
0x80	PC00_SEL	RW	Port PC00 Function Configuration Register
0x84	PC01_SEL	RW	Port PC01 Function Configuration Register

Offset	Register name	Access	Register description
0x88	PC02_SEL	RW	Port PC02 Function Configuration Register
0x8c	PC03_SEL	RW	Port PC03 Function Configuration Register
0x90	PC04_SEL	RW	Port PC04 Function Configuration Register
0x94	PC05_SEL	RW	Port PC05 Function Configuration Register
0x98	PC06_SEL	RW	Port PC06 Function Configuration Register
0x9c	PC07_SEL	RW	Port PC07 Function Configuration Register
0xa0	PC08_SEL	RW	Port PC08 Function Configuration Register
0xa4	PC09_SEL	RW	Port PC09 Function Configuration Register
0xa8	PC10_SEL	RW	Port PC10 Function Configuration Register
0xac	PC11_SEL	RW	Port PC11 Function Configuration Register
0xb0	PC12_SEL	RW	Port PC12 Function Configuration Register
0xb4	PC13_SEL	RW	Port PC13 Function Configuration Register
0xb8	PC14_SEL	RW	Port PC14 Function Configuration Register
0xbc	PC15_SEL	RW	Port PC15 Function Configuration Register
0xc0	PD00_SEL	RW	Port PD00 Function Configuration Register
0xc4	PD01_SEL	RW	Port PD01 Function Configuration Register
0xc8	PD02_SEL	RW	Port PD02 Function Configuration Register
0xcc	PD03_SEL	RW	Port PD03 Function Configuration Register
0xd0	PD04_SEL	RW	Port PD04 Function Configuration Register
0xd4	PD05_SEL	RW	Port PD05 Function Configuration Register
0xd8	PD06_SEL	RW	Port PD06 Function Configuration Register
0xdc	PD07_SEL	RW	Port PD07 Function Configuration Register
0xe0	PD08_SEL	RW	Port PD08 Function Configuration Register
0xe4	PD09_SEL	RW	Port PD09 Function Configuration Register
0xe8	PD10_SEL	RW	Port PD10 Function Configuration Register
0xec	PD11_SEL	RW	Port PD11 Function Configuration Register
0xf0	PD12_SEL	RW	Port PD12 Function Configuration Register
0xf4	PD13_SEL	RW	Port PD13 Function Configuration Register
0xf8	PD14_SEL	RW	Port PD14 Function Configuration Register
0xfc	PD15_SEL	RW	Port PD15 Function Configuration Register
0x100	PADIR	RW	Port PA input and output configuration register
0x104	PAIN	RO	Port PA Input value register
0x108	PAOUT	RW	Port PA output value configuration register
0x10c	PAADS	RW	Port PA digital-analog configuration register
0x110	PABSET	RW	Port PA Bit Set Register
0x114	PABCLR	RW	Port PA Bit Clear Register
0x118	PABSETCLR	RW	Port PA Bit Set Clear Register
0x11c	PADR	RW	Port PA drive capability configuration register

Offset	Register name	Access	Register description
0x120	PAPU	RW	Port PA pull-up enable configuration register
0x124	PAPD	RW	Port PA pull-down enable configuration register
0x12c	PAOD	RW	Port PA open-drain output configuration register
0x130	PAHIE	RW	Port PA high level interrupt enable configuration register
0x134	PALIE	RW	Port PA Low Level Interrupt Enable Configuration Register
0x138	PARIE	RW	Port PA rising edge interrupt enable configuration register
0x13c	PAFIE	RW	Port PA falling edge interrupt enable configuration register
0x200	PA_STAT	RO	Port PA interrupt status register
0x210	PA_ICLR	RW	Port PA interrupt clear register
0x140	PBDIR	RW	Port PB Input and Output Configuration Register
0x144	PBIN	RO	Port PB input value register
0x148	PBOUT	RW	Port PB output value configuration register
0x14c	PBADS	RW	Port PB digital-analog configuration register
0x150	PBBSET	RW	Port PB bit set register
0x154	PBBCLR	RW	Port PB bit clear register
0x158	PBBSETCLR	RW	Port PB bit set clear register
0x15c	PBDR	RW	Port PB Drive Capability Configuration Register
0x160	PBPU	RW	Port PB pull-up enable configuration register
0x164	PBPD	RW	Port PB pull-down enable configuration register
0x16c	PBOD	RW	Port PB Open-Drain Output Configuration Register
0x170	PBHIE	RW	Port PB High Level Interrupt Enable Configuration Register
0x174	PBLIE	RW	Port PB low level interrupt enable configuration register
0x178	PBRIE	RW	Port PB Rising Edge Interrupt Enable Configuration Register
0x17c	PBFIE	RW	Port PB Falling Edge Interrupt Enable Configuration Register
0x240	PB_STAT	RO	Port PB Interrupt Status Register
0x250	PB_ICLR	RW	Port PB Interrupt Clear Register
0x180	PCDIR	RW	Port PC Input and Output Configuration Register
0x184	PCIN	RO	Port PC Input Value Register
0x188	PCOUT	RW	Port PC output value configuration register
0x18c	PCADS	RW	Port PC digital-analog configuration register
0x190	PCBSET	RW	Port PC Bit Set Register
0x194	PCBCLR	RW	Port PC Bit Clear Register
0x198	PCBSETCLR	RW	Port PC bit is set to clear the register
0x19c	PCDR	RW	Port PC Drive Capability Configuration Register
0x1a0	PCPU	RW	Port PC pull-up enable configuration register
0x1a4	PCPD	RW	Port PC pull-down enable configuration register
0x1ac	PCOD	RW	Port PC Open-Drain Output Configuration Register
0x1b0	PCHIE	RW	Port PC High Level Interrupt Enable Configuration Register

Offset	Register name	Access	Register description
0x1b4	PCLIE	RW	Port PC low level interrupt enable configuration register
0x1b8	PCRIE	RW	Port PC Rising Edge Interrupt Enable Configuration Register
0x1bc	PCFIE	RW	Port PC Falling Edge Interrupt Enable Configuration Register
0x280	PC_STAT	RO	Port PC Interrupt Status Register
0x290	PC_ICLR	RW	Port PC Interrupt Clear Register
0x1c0	PDDIR	RW	Port PD Input and Output Configuration Register
0x1c4	PDIN	RO	Port PD Input Value Register
0x1c8	PDOUT	RW	Port PD output value configuration register
0x1cc	PDADS	RW	Port PD digital-analog configuration register
0x1d0	PDBSET	RW	Port PD Bit Set Register
0x1d4	PDBCLR	RW	Port PD Bit Clear Register
0x1d8	PDBSETCLR	RW	Port PD Bit Set Clear Register
0x1dc	PDDR	RW	Port PD Drive Capability Configuration Register
0x1e0	PDPU	RW	Port PD pull-up enable configuration register
0x1e4	PDPD	RW	Port PD pull-down enable configuration register
0x1ec	PDOD	RW	Port PD Open-Drain Output Configuration Register
0x1f0	PDHIE	RW	Port PD High Level Interrupt Enable Configuration Register
0x1f4	PDLIE	RW	Port PD low level interrupt enable configuration register
0x1f8	PDRIE	RW	Port PD Rising Edge Interrupt Enable Configuration Register
0x1fc	PDFIE	RW	Port PD Falling Edge Interrupt Enable Configuration Register
0x2c0	PD_STAT	RO	Port PD Interrupt Status Register
0x2d0	PD_ICLR	RW	Port PD Interrupt Clear Register
0x304	GPIO_CTRL1	RW	Port Miscellaneous Function Configuration Register 1
0x308	GPIO_CTRL2	RW	Port Miscellaneous Function configuration register 2
0x30c	GPIO_TIMGS	RW	Port Auxiliary Function Timer Gate Selection
0x310	GPIO_TIMES	RW	Port auxiliary function timer ETR selection
0x314	GPIO_TIMCPS	RW	Port Auxiliary Function Timer Capture Input Selection
0x318	GPIO_PCAS	RW	Port Auxiliary Function PCA Capture Selection
0x31c	GPIO_PCNT	RW	Port auxiliary function PCNT input selection
0x1000	PE00_SEL	RW	Port PE00 Function Configuration Register
0x1004	PE01_SEL	RW	Port PE01 Function Configuration Register
0x1008	PE02_SEL	RW	Port PE02 Function Configuration Register
0x100c	PE03_SEL	RW	Port PE03 Function Configuration Register
0x1010	PE04_SEL	RW	Port PE04 Function Configuration Register
0x1014	PE05_SEL	RW	Port PE05 Function Configuration Register
0x1018	PE06_SEL	RW	Port PE06 Function Configuration Register
0x101c	PE07_SEL	RW	Port PE07 Function Configuration Register
0x1020	PE08_SEL	RW	Port PE08 Function Configuration Register

Offset	Register name	Access	Register description
0x1024	PE09_SEL	RW	Port PE09 Function Configuration Register
0x1028	PE10_SEL	RW	Port PE10 Function Configuration Register
0x102c	PE11_SEL	RW	Port PE11 Function Configuration Register
0x1030	PE12_SEL	RW	Port PE12 Function Configuration Register
0x1034	PE13_SEL	RW	Port PE13 Function Configuration Register
0x1038	PE14_SEL	RW	Port PE14 Function Configuration Register
0x103c	PE15_SEL	RW	Port PE15 Function Configuration Register
0x1040	PF00_SEL	RW	Port PF00 Function Configuration Register
0x1044	PF01_SEL	RW	Port PF01 Function Configuration Register
0x1048	PF02_SEL	RW	Port PF02 Function Configuration Register
0x104c	PF03_SEL	RW	Port PF03 Function Configuration Register
0x1050	PF04_SEL	RW	Port PF04 Function Configuration Register
0x1054	PF05_SEL	RW	Port PF05 Function Configuration Register
0x1058	PF06_SEL	RW	Port PF06 Function Configuration Register
0x105c	PF07_SEL	RW	Port PF07 Function Configuration Register
0x1064	PF09_SEL	RW	Port PF09 Function Configuration Register
0x1068	PF10_SEL	RW	Port PF10 Function Configuration Register
0x106c	PF11_SEL	RW	Port PF11 Function Configuration Register
0x1100	PEDIR	RW	Port PE input and output configuration register
0x1104	PEIN	RO	Port PE Input Value Register
0x1108	PEOUT	RW	Port PE output value configuration register
0x110c	PEADS	RW	Port PE digital-analog configuration register
0x1110	PEBSET	RW	Port PE Bit Set Register
0x1114	PEBCLR	RW	Port PE Bit Clear Register
0x1118	PEBSETCLR	RW	Port PE bit is set to clear the register
0x111c	PEDR	RW	Port PE Drive Capability Configuration Register
0x1120	PEPU	RW	Port PE pull-up enable configuration register
0x1124	PEPD	RW	Port PE pull-down enable configuration register
0x112c	PEOD	RW	Port PE Open-Drain Output Configuration Register
0x1130	PEHIE	RW	Port PE High Level Interrupt Enable Configuration Register
0x1134	PELIE	RW	Port PE low level interrupt enable configuration register
0x1138	PERIE	RW	Port PE Rising Edge Interrupt Enable Configuration Register
0x113c	PEFIE	RW	Port PE Falling Edge Interrupt Enable Configuration Register
0x1200	PE_STAT	RO	Port PE Interrupt Status Register
0x1210	PE_ICLR	RW	Port PE Interrupt Clear Register
0x1140	PFDIR	RW	Port PF Input and Output Configuration Register
0x1144	PFIN	RO	Port PF Input Value Register
0x1148	PFOUT	RW	Port PF output value configuration register

Offset	Register name	Access	Register description
0x114c	PFADS	RW	Port PF digital-analog configuration register
0x1150	PFBSET	RW	Port PF Bit Set Register
0x1154	PFBCLR	RW	Port PF Bit Clear Register
0x1158	PFBSETCLR	RW	Port PF Bit Set Clear Register
0x115c	PFDR	RW	Port PF Drive Capability Configuration Register
0x1160	PFPU	RW	Port PF pull-up enable configuration register
0x1164	PFPD	RW	Port PF pull-down enable configuration register
0x116c	PFOD	RW	Port PF Open-Drain Output Configuration Register
0x1170	PFHIE	RW	Port PF High Level Interrupt Enable Configuration Register
0x1174	PFLIE	RW	Port PF low level interrupt enable configuration register
0x1178	PFRIE	RW	Port PF Rising Edge Interrupt Enable Configuration Register
0x117c	PFFIE	RW	Port PF Falling Edge Interrupt Enable Configuration Register
0x1240	PF_STAT	RO	Port PF Interrupt Status Register
0x1250	PF_ICLR	RW	Port PF Interrupt Clear Register

6.5.1 Port general register

6.5.1.1 Port Px input and output configuration register (PxDIR) (x = A, B, C, D, E, F)

Address offset:

0x100(PA),0x140(PB),0x180(PC),0x1C0(PD),0x1100(PE),0x1140(PF)

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
PxDIR[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxDIR	Port Px input and output configuration register (corresponding to Px15-Px00) 1: configured as input 0: configured as output Note: Each bit corresponds to a port, for example: PxDIR[15] corresponds to port Px15

6.5.1.2 Port Px Input Value Register (PxIN) (x = A, B, C, D, E, F)

Address offset:

0x104(PA),0x144(PB),0x184(PC),0x1C4(PD),0x1104(PE),0x1144(PF)

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
PxIN[15:0]															
ro															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxIN	Port Px input value register (corresponding to Px15-Px00) 1: Input is high level 0: Input is low level

6.5.1.3 Port Px output value configuration register (PxOUT) (x = A, B, C, D, E, F)

Address offset:

0x108(PA),0x148(PB),0x188(PC),0x1C8(PD),0x1108(PE),0x1148(PF)

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
PxOUT[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxOUT	Port Px output value configuration register (corresponding to Px15-Px00) 1: Output high level If it is configured as an open-drain output, an external pull-up resistor is required to pull it high. 0: Output low level

6.5.1.4 Port Px Digital-to-Analog Configuration Register (PxADS) (x = A, B, C, D, E, F)

Address offset:

0x10C(PA),0x14C(PB),0x18C(PC),0x1CC(PD),0x110C(PE),0x114C(PF)

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
PxADS[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxADS	Port Px digital-analog configuration register (corresponding to Px15-Px00) 1: configured as an analog port 0: configured as a digital port

6.5.1.5 Port Px Bit Set Register (PxBSET) (x = A, B, C, D, E, F)

Address offset:

0x110(PA),0x150(PB),0x190(PC),0x1D0(PD),0x1110(PE),0x1150(PF)

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
PxBSET[15:0]															
W1															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxBSET	Port Px bit bit register (corresponding to Px15-Px00) 1: set 0: keep

6.5.1.6 Port Px Bit Clear Register (PxBCLR) (x = A, B, C, D, E, F)

Address offset:

0x114(PA),0x154(PB),0x194(PC),0x1D4(PD),0x1114(PE),0x1154(PF)

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
PxBCLR[15:0]															
W1															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxBCLR	Port Px bit clear register (corresponding to Px15-Px00) 1: clear 0: keep

6.5.1.7 Port Px Bit Set Clear Register (PxBSETCLR) (x = A, B, C, D, E, F)

Address offset:

0x118(PA),0x158(PB),0x198(PC),0x1D8(PD),0x1118(PE),0x1158(PF)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PxBSET[15:0]															
W1															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxBCLR[15:0]															
W1															

Bit	Marking	Functional description
31:16	PxBSET	Port Px bit set register 1: set 0: keep
15:0	PxBCLR	Port Px bit clear register 1: clear 0: keep

Note:

– When the same bit of PxBSET and PxBCLR are set to 1 at the same time, PxBCLR has high priority. That is, the port is cleared.

6.5.1.8 Port Px pull-up enable configuration register (PxPU) (x = A, B, C, D, E, F)

Address offset:

0x120(PA),0x160(PB),0x1A0(PC),0x1E0(PD),0x1120(PE),0x1160(PF)

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
PxPU[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxPU	Port Px pull-up enable configuration register 1: Enable 0: Prohibited

6.5.1.9 Port Px Drive Capability Configuration Register (PxDR) (x = A, B, C, D, E, F)

Address offset:

0x11C(PA),0x15C(PB),0x19C(PC),0x1DC(PD),0x111C(PE),0x115C(PF)

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
PxDR[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxDR	Port Px drive capability configuration register 1: Low drive capability 0: High drive capability

6.5.1.10 Port Px pull-down enable configuration register (PxPD) (x = A, B, C, D, E, F)

Address offset:

0x124(PA),0x164(PB),0x1A4(PC),0x1E4(PD), 0x1124(PE),0x1164(PF)

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
PxPD[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxPD	Port Px pull-down enable configuration register 1: Enable 0: Prohibited

6.5.1.11 Port Px Open-Drain Output Configuration Register (PxOD) (x = A, B, C, D, E, F)

Address offset:

0x12C(PA),0x16C(PB),0x1AC(PC),0x1EC(PD),0x112C(PE),0x116C(PF)

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
PxOD[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxOD	Port Px 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

6.5.1.12 Port Px High Level Interrupt Enable Configuration Register (PxHIE) (x = A, B, C, D, E, F)

Address offset:

0x130(PA),0x170(PB),0x1B0(PC),0x1F0(PD),0x1130(PE),0x1170(PF)

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
PxHIE[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxHIE	Port Px High Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited

6.5.1.13 Port Px Low Level Interrupt Enable Configuration Register (PxLIE) (x = A, B, C, D, E, F)

Address offset:

0x134(PA),0x174(PB),0x1B4(PC),0x1F4(PD),0x1134(PE),0x1174(PF)

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
PxLIE[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxLIE	Port Px Low Level Interrupt Enable Configuration Register 1: Enable 0: Prohibited

6.5.1.14 Port Px Rising Edge Interrupt Enable Configuration Register (PxRIE) (x = A, B, C, D, E, F)

Address offset:

0x138(PA),0x178(PB),0x1B8(PC),0x1F8(PD),0x1138(PE),0x1178(PF)

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
PxRIE[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxRIE	Port Px rising edge interrupt enable configuration register 1: Enable 0: Prohibited

6.5.1.15 Port Px Falling Edge Interrupt Enable Configuration Register (PxFIE) (x = A, B, C, D, E, F)

Address offset:

0x13C(PA),0x17C(PB),0x1BC(PC),0x1FC(PD),0x113C(PE),0x117C(PF)

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
PxFIE[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	PxFIE	Port Px falling edge interrupt enable configuration register 1: Enable 0: Prohibited

6.5.1.16 Port Px Interrupt Status Register (Px_STAT) (x = A, B, C, D, E, F)

Address offset:

0x200(PA),0x240(PB),0x280(PC),0x2C0(PD),0x1200(PE),0x1240(PF)

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
Px_STAT[15:0]															
ro															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	Px_STAT	Port Px Interrupt Status Register 1: interrupt trigger 0: no interrupt trigger

6.5.1.17 Port Px Interrupt Clear Register (Px_ICLR) (x = A, B, C, D, E, F)

Address offset:

0x210(PA),0x250(PB),0x290(PC),0x2D0(PD),0x1210(PE),0x1250(PF)

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
Px_ICLR[15:0]															
rw															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	Px_ICLR	Port Px Interrupt Clear Register 1: Reserve interrupt flag bit 0: Clear the interrupt flag bit

6.5.2 Port Auxiliary Register

6.5.2.1 Port auxiliary Function configuration register 1 (GPIO_CTRL1)

Address offset: 0x304

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	ir_pol	hclk_en	pclk_en	hclk_sel		pclk_sel		ssn0_sel				ext_clk_sel			
	rw	rw	rw	rw		rw		rw				rw			

Bit	Marking	Functional description
31:15	Reserved	Keep
14	ir_pol	IR output polarity selection. 0 – positive output 1 – Reverse output
13	hclk_en	HCLK output control 0 – outputs 0 1 – output HCLK
12	pclk_en	PCLK output gate. 0 – outputs 0 1 – output PCLK
11:10	hclk_sel	HCLK output frequency division selection. 00：HCLK 01：HCLK/2 10：HCLK/4 11：HCLK/8
9:8	pclk_sel	PCLK output frequency division selection. 00：PCLK 01：PCLK/2 10：PCLK/4 11：PCLK/8
7:4	ssn0_sel	SPI0 SSN signal source selection. 0000: High level 1000: PB01 0001: PA03 1001: PB02 0010: PA04 1010: PB05 0011: PA06 1011: PB06 0100: PA08 1100: PB09 0101: PA09 1101: PB10 0110: PA12 1110: PB12 0111: PA15 1111: PB14
3:0	ext_clk_sel	External clock signal source selection. 0000: High level 1000: PB01 0001: PA03 1001: PB02 0010: PA04 1010: PB05 0011: PA06 1011: PB06 0100: PA08 1100: PB09 0101: PA09 1101: PB10 0110: PA12 1110: PB12 0111: PA15 1111: PB14

6.5.2.2 Port auxiliary function configuration register 2 (GPIO_CTRL2)

Address offset: 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
ahb_sel	Res							tclk_div		tclk_sel		ssn1_sel			
rw								rw		rw		rw			

Bit	Marking	Functional description
31:4	Reserved	Keep
15	ahb_sel	AHB bus access mode 0 – Use the FASTIO bus to access PxIN, PxOUT registers 1 -- Use the AHB bus to access the PxIN, PxOUT registers
14:8	Reserved	Keep
7:6	tclk_div	TCLK output frequency division selection. 00: TCLK 01: TCLK/2 10: TCLK/4 11: TCLK/8
5:4	tclk_sel	tclk output signal source selection. 0 0 RCH48M 0 1 RCH 1 0 XTH 1 1 PLL
3:0	ssn1_sel	SPI1 SSN signal source selection. 0000 High level 0001 PA03 0010 PA04 0011 PA06 0100 PA08 0101 PA09 0110 PA12 0111 PA15 1000 PB01 1001 PB02 1010 PB05 1011 PB06 1100 PB09 1101 PB10 1110 PB12 1111 PB14

6.5.2.3 Port auxiliary function timer gate selection (GPIO_TIMGS)

Address offset: 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
Res.	LPTIM0_G			TIM3_G			TIM2_G			TIM1_G			TIM0_G		
	R/W			R/W			R/W			R/W			R/W		

Bit	Marking	Functional description
31:15	Reserved	Keep
14:12	LPTIM0_G	LPTimer0 timer GATE input selection, see the table below for selection
11:9	TIM3_G	Timer3 timer GATE input selection, see the table below for selection
8:6	TIM2_G	Timer2 timer GATE input selection, see the table below for selection
5:3	TIM1_G	Timer1 timer GATE input selection, see the table below for selection
2:0	TIM0_G	Timer0 timer GATE input selection, see the table below for selection

	TIM0_g	TIM1_g	TIM2_g	TIM3_g	LPTIM0_g
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	UART0_RXD	LPUART0_RXD	UART0_RXD	UART0_RXD	LPUART0_RXD
010	UART1_RXD	LPUART1_RXD	UART1_RXD	UART1_RXD	LPUART1_RXD
011	VC0_OUT	VC0_OUT	VC0_OUT	UART2	VC0_OUT
100	VC1_OUT	VC1_OUT	VC1_OUT	UART3	VC1_OUT
101	PA03	PA08	PA10	VC0_OUT	PB03
110	PB08	PB03	PB04	PA06	PB05
111	PB15	PB13	PB11	PA11	PC00

6.5.2.4 Port auxiliary function timer ETR selection (GPIO_TIMES)

Address offset: 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
Res.	LPTIM0_E			TIM3_E			TIM2_E			TIM1_E			TIM0_E		
	R/W			R/W			R/W			R/W			R/W		

Bit	Marking	Functional description
31:15	Reserved	Keep
14:12	LPTIM0_E	LPTimer0 timer EXT input selection, see the table below for selection
11:9	TIM3_E	Timer3 timer ETR input selection, see the table below for selection
8:6	TIM2_E	Timer2 timer ETR input selection, see the table below for selection
5:3	TIM1_E	Timer1 timer ETR input selection, see the table below for selection
2:0	TIM0_E	Timer0 timer ETR input selection, see the table below for selection

	TIM0_e	TIM1_e	TIM2_e	TIM3_e	LPTIM0_E
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	LPUART0_RXD	UART0_RXD	LPUART0_RXD	UART0_RXD	PCNT_S0
010	LPUART1_RXD	UART1_RXD	LPUART1_RXD	UART1_RXD	LVD_OUT
011	VC0_OUT	VC1_OUT	VC0_OUT	VC1_OUT	VC0_OUT
100	LVD_OUT	LVD_OUT	PCNT_S1	PCNT_S0	VC1_OUT
101	PA00	PA01	PA04	PA00	PB04
110	PA05	PC09	PC04	PA12	PB06
111	PA15	PD02	PC08	PA13	PC03

6.5.2.5 Port auxiliary function timer capture input selection (GPIO_TIMCPS)

Address offset: 0x314

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.	TIM3_CB			TIM3_CA			TIM2_CA			TIM1_CA			TIM0_CA		
	R/W			R/W			R/W			R/W			R/W		

Bit	Marking	Functional description
31:15	Reserved	Keep
14:12	TIM3_CB	Timer3 timer CH0B input selection, see the table below for selection
11:9	TIM3_CA	Timer3 timer CH0A input selection, see the table below for selection
8:6	TIM2_CA	Timer2 timer CHA input selection, see the table below for selection
5:3	TIM1_CA	Timer1 timer CHA input selection, see the table below for selection
2:0	TIM0_CA	Timer0 timer CHA input selection, see the table below for selection

	TIM0_CHA	TIM1_CHA	TIM2_CHA	TIM3_CH0A	TIM3_CH0B
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	UART0_RXD	UART1_RXD	LPUART0_RXD	LPUART1_RXD	UART0_RXD
010	PA00	PA00	VC0_OUT	LPUART0_RXD	UART1_RXD
011	PA02	PA02	PA02	PCNT_S0	PCNT_S1
100	PA05	PA06	PA07	VC0_OUT	VC1_OUT
101	PA15	PB08	PB08	PA08	PA07
110	PB06	PB10	PB09	PB03	PB04
111	PB14	PB13	PC06	PB06	PB13

6.5.2.6 Port auxiliary function PCA capture selection (GPIO_PCAS)

Address offset: 0x318

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				LPTIM1_E			LPTIM1_G			PCA_CH0			PCA_ECI		
				R/W			R/W			R/W			R/W		

Bit	Marking	Functional description
31:12	Reserved	Keep
11:9	LPTIM1_E	LPTimer1 timer EXT input selection, see the table below for selection
8:6	LPTIM1_G	LPTimer1 timer GATE input selection, see the table below for selection
5:3	PCA_ECI	PCA ECI clock input selection, see the table below for selection
2:0	PCA_CH0	PCA CH0 capture port input selection, see the table below for selection

	LPTIM1_E	LPTIM1_G	pca_eci	pca_ch0
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	UART1	UART0	PCNT_S1	PCNT_S0
010	LPUART1_RXD	LPUART0_RXD	LVD_OUT	PCNT_S1
011	VC0_OUT	VC1_OUT	VC0_OUT	LVD_OUT
100	VC2_OUT	VC2_OUT	VC1_OUT	VC1_OUT
101	LPTIM0_TOG	PD5	PA05	PA06
110	PE8	PE7	PB02	PB04
111	PC7	PC6	PD02	PC06

6.5.2.7 Port auxiliary function PCNT pulse input selection (GPIO_PCNT)

Address offset: 0x31C

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												PCNT_S1		PCNT_S0	
												R/W		R/W	

Bit	Marking	Functional description
31:4	Reserved	Keep
3:2	PCNT_S1	PCNT_S1 input selection, see the table below for selection
1:0	PCNT_S0	PCNT_S0 input selection, see the table below for selection

	PCNT_S1	PCNT_S0
00	PX_SEL	PX_SEL
01	VC0_OUT	VC0_OUT
10	VC1_OUT	VC1_OUT
11	VC2_OUT	VC2_OUT

6.5.3 Port PA Function Selection Register

6.5.3.1 Port PA00 Function configuration register (PA00_SEL)

Address offset: 0x00

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PA00_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA00_sel	Port PA00 function selection. 000 ---- GPIO PA00 001 ---- UART1_CTS UART1 module CTS signal 010 ---- LPUART1_TXD LPUART1 module TXD signal 011 ---- TIM0_ETR Timer0 module external clock input signal 100 ---- VC0_OUT VC0 module output / reverse output signal 101 ---- TIM1_CHA Timer1 module channel A signal 110 ---- TIM3_ETR Timer3 module external clock input signal 111 ---- TIM0_CHA Timer0 module channel A signal

6.5.3.2 Port PA01 Function configuration register (PA01_SEL)

Address offset: 0x04

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PA01_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA01_sel	Port PA01 function selection. 000 ---- GPIO PA01 001 ---- UART1_RTS UART1 module RTS signal 010 ---- LPUART1_RXD LPUART1 module RXD signal 011 ---- TIM0_CHB Timer0 module channel B signal 100 ---- TIM1_ETR Timer1 module external clock input signal 101 ---- TIM1_CHB Timer1 module channel B signal 110 ---- HCLK_OUT AHB bus clock output signal 111 ---- SPI1_MOSI SPI1 module master output slave input data signal

6.5.3.3 Port PA02 Function configuration register (PA02_SEL)

Address offset: 0x08

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													PA02_sel		
													rw		

Reset value: 0x0000 0000

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA02_sel	Port PA02 function selection. 000 ---- GPIO PA02 001 ---- UART1_TXD UART1 module TXD signal 010 ---- TIM0_CHA Timer0 module channel A signal 011 ---- VC1_OUT VC1 module output / reverse output signal 100 ---- TIM1_CHA Timer1 module channel A signal 101 ---- TIM2_CHA Timer2 module channel A signal 110 ---- PCLK_OUT APB bus clock output signal 111 ---- SPI1_MISO SPI1 module Master input slave output data signal

6.5.3.4 Port PA03 Function configuration register (PA03_SEL)

Address offset: 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													PA03_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA03_sel	Port PA03 function selection. 000 ---- GPIO PA03 001 ---- UART1_RXD UART1 module RXD signal 010 ---- TIM0_GATE Timer0 module gate control signal 011 ---- TIM1_CHB Timer1 module channel B signal 100 ---- TIM2_CHB Timer2 module channel B signal 101 ---- SPI1_CS SPI1 module master mode chip select signal 110 ---- TIM3_CH1A Timer3 module channel 1A signal 111 ---- TIM5_CHA Timer6 module channel 1A signal

6.5.3.5 Port PA04 Function configuration register (PA04_SEL)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PA04_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA04_sel	Port PA04 function selection. 000 ---- GPIO PA04 001 ---- SPI0_CS SPI0 module master mode chip select signal 010 ---- UART1_TXD UART1 module TXD signal 011 ---- PCA_CH4 PCA module channel 4 capture / comparison signal 100 ---- TIM2_ETR Timer2 module external clock input signal 101 ---- TIM5_CHA Timer6 module channel 1A signal 110 ---- LVD_OUT LVD module output signal 111 ---- TIM3_CH2B Timer3 module channel 2B signal

6.5.3.6 Port PA05 Function configuration register (PA05_SEL)

Address offset: 0x14

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															PA05_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA05_sel	Port PA05 function selection. 000 ---- GPIO PA05 001 ---- SPI0_SCK SPI0 module clock signal 010 ---- TIM0_ETR Timer0 module external clock input signal 011 ---- PCA_ECI PCA module external clock input signal 100 ---- TIM0_CHA Timer0 module channel A signal 101 ---- TIM5_CHB Timer6 module channel 1B signal 110 ---- XTL_OUT External 32K crystal oscillator output signal 111 ---- XTH_OUT External 32M crystal oscillator output signal

6.5.3.7 Port PA06 Function configuration register (PA06_SEL)

Address offset: 0x18

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													PA06_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA06_sel	Port PA06 function selection. 000 ---- GPIO PA06 001 ---- SPI0_MISO SPI0 module Master input slave output data signal 010 ---- PCA_CH0 PCA module channel 0 capture / compare signal 011 ---- TIM3_BK Timer3 module brake signal 100 ---- TIM1_CHA Timer1 module channel A signal 101 ---- VC0_OUT VC0 module output / reverse output signal 110 ---- TIM3_GATE Timer3 module gate control signal 111 ---- LPUART0_CTS LPUART0 module CTS signal

6.5.3.8 Port PA07 Function configuration register (PA07_SEL)

Address offset: 0x1C

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													PA07_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA07_sel	Port PA07 function selection. 000 ---- GPIO PA07 001 ---- SPI0_MOSI SPI0 module master output slave input data signal 010 ---- PCA_CH1 PCA module channel 1 capture / compare signal 011 ---- HCLK_OUT AHB bus clock output signal 100 ---- TIM3_CH0B Timer3 module channel 0B signal 101 ---- TIM2_CHA Timer2 module channel A signal 110 ---- VC1_OUT VC1 module output / reverse output signal 111 ---- TIM4_CHB Timer6 module channel 0B signal

6.5.3.9 Port PA08 Function configuration register (PA08_SEL)

Address offset: 0x20

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															PA08_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA08_sel	PA08_sel: Port PA08 function selection. 000 ---- GPIO PA08 001 ---- UART0_TXD UART0 module TXD signal 010 ---- TIM3_CH0A Timer3 module channel 0A signal 011 ---- CRS_SYNC TBD 100 ---- CAN_STBY TBD 101 ---- TIM1_GATE Timer1 module gate control signal 110 ---- TIM4_CHA Timer6 module channel 0A signal 111 ---- TIM3_BK Timer3 module brake signal

6.5.3.10 Port PA09 Function configuration register (PA09_SEL)

Address offset: 0x24

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															PA09_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA09_sel	Port PA09 function selection. 000 ---- GPIO PA09 001 ---- UART0_TXD UART0 module TXD signal 010 ---- TIM3_CH1A Timer3 module channel 1A signal 011 ---- TIM0_BK Timer0 module brake signal 100 ---- I2C0_SCL I2C0 module clock signal 101 ---- Reserved Reserved 110 ---- HCLK_OUT AHB bus clock output signal 111 ---- TIM5_CHA Timer6 module channel 1A signal

6.5.3.11 Port PA10 Function configuration register (PA10_SEL)

Address offset: 0x28

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															PA10_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA10_sel	PA10_sel: Port PA10 function selection. 000 ---- GPIO PA10 001 ---- UART0_RXD UART0 module RXD signal 010 ---- TIM3_CH2A Timer3 module channel 2A signal 011 ---- TIM2_BK Timer2 module brake signal 100 ---- I2C0_SDA I2C0 module data signal 101 ---- TIM2_GATE Timer2 module gate control signal 110 ---- PCLK_OUT APB bus clock output signal 111 ---- TIM6_CHA Timer6 module channel 2A signal

6.5.3.12 Port PA11 Function configuration register (PA11_SEL)

Address offset: 0x2C

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															PA11_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA11_sel	PA11_sel: Port PA11 function selection. 000 ---- GPIO PA11 001 ---- UART0_CTS UART0 module CTS signal 010 ---- TIM3_GATE Timer3 module gate control signal 011 ---- I2C1_SCL I2C1 module clock signal 100 ---- CAN_RX CAN module receives signal 101 ---- VC0_OUT VC0 module output / reverse output signal 110 ---- SPI0_MISO SPI0 module Master input slave output data signal 111 ---- TIM4_CHB Timer6 module channel 0B signal

6.5.3.13 Port PA12 Function configuration register (PA12_SEL)

Address offset: 0x30

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													PA12_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA12_sel	PA12_sel: Port PA12 function selection. 000 ---- GPIO PA12 001 ---- UART0_RTS UART0 module RTS signal 010 ---- TIM3_ETR Timer3 module external clock input signal 011 ---- I2C1_SDA I2C1 module data signal 100 ---- CAN_TX CAN module sends signal 101 ---- VC1_OUT VC1 module output / reverse output signal 110 ---- SPI0_MOSI SPI0 module master output slave input data signal 111 ---- PCNT_S0 PCNT module input signal 0

6.5.3.14 Port PA13 Function configuration register (PA13_SEL)

Address offset: 0x34

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													PA13_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA13_sel	PA13_sel: Port PA13 function selection. 000 ---- GPIO PA13 001 ---- IR_OUT infrared output signal 010 ---- UART0_RXD UART0 module RXD signal 011 ---- LVD_OUT LVD module output signal 100 ---- TIM3_ETR Timer3 module external clock input signal 101 ---- RTC_1HZ RTC module 1Hz output signal 110 ---- PCNT_S1 PCNT module input signal 1 111 ---- VC2_OUT VC2 module output / reverse output signal

6.5.3.15 Port PA14 Function configuration register (PA14_SEL)

Address offset: 0x38

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															PA14_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA14_sel	PA14_sel: Port PA14 function selection. 000 ---- GPIO PA14 001 ---- UART1_TXD UART1 module TXD signal 010 ---- UART0_TXD UART0 module TXD signal 011 ---- TIM3_CH2A Timer3 module channel 2A signal 100 ---- LVD_OUT LVD module output signal 101 ---- RCH_OUT internal 24M RC clock output signal 110 ---- RCL_OUT internal 38K RC clock output signal 111 ---- PLL_OUT internal PLL clock output signal

6.5.3.16 Port PA15 Function configuration register (PA15_SEL)

Address offset: 0x3C

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															PA15_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PA15_sel	PA15_sel: Port PA15 function selection. 000 ---- GPIO PA15 001 ---- SPI0_CS SPI0 module master mode chip select signal 010 ---- UART1_RXD UART1 module RXD signal 011 ---- LPUART1_RTS LPUART1 module RTS signal 100 ---- TIM0_ETR Timer0 module external clock input signal 101 ---- TIM0_CHA Timer0 module channel A signal 110 ---- TIM3_CH1A Timer3 module channel 1A signal 111 ---- Reserved Reserved

6.5.4 Port PB Function selection register

6.5.4.1 Port PB00 Function configuration register (PB00_SEL)

Address offset: 0x40

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													PB00_sel		
													rw		

Reset value: 0x0000 0000

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB00_sel	Port PB00 function selection. 000 ---- GPIO PB00 001 ---- PCA_CH2 PCA module channel 2 capture / compare signal 010 ---- TIM3_CH1B Timer3 module channel 1B signal 011 ---- LPUART0_TXD LPUART0 module TXD signal 100 ---- TIM5_CHB Timer6 module channel 1B signal 101 ---- RCH_OUT internal 24M RC clock output signal 110 ---- RCL_OUT internal 38K RC clock output signal 111 ---- PLL_OUT internal PLL clock output signal

6.5.4.2 Port PB01 Function configuration register (PB01_SEL)

Address offset: 0x44

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													PB01_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB01_sel	PB01_sel: Port PB01 function selection. 000 ---- GPIO PB01 001 ---- PCA_CH3 PCA module channel 3 capture / compare signal 010 ---- PCLK_OUT APB bus clock output signal 011 ---- TIM3_CH2B Timer3 module channel 2B signal 100 ---- TIM6_CHB Timer6 module channel 2B signal 101 ---- LPUART0_RTS LPUART0 module RTS signal 110 ---- VC2_OUT VC2 module output / reverse output signal 111 ---- TCLK_OUT internal clock output signal

6.5.4.3 Port PB02 Function configuration register (PB02_SEL)

Address offset: 0x48

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															PB02_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB02_sel	PB02_sel: Port PB02 function selection. 000 ---- GPIO PB02 001 ---- LPTIM0_TOG low power timer module flip signal 010 ---- PCA_ECI PCA module external clock input signal 011 ---- LPUART1_TXD LPUART1 module TXD signal 100 ---- TIM4_CHA Timer6 module channel 0A signal 101 ---- TIM1_BK Timer1 module brake signal 110 ---- TIM0_BK Timer0 module brake signal 111 ---- TIM2_BK Timer2 module brake signal

6.5.4.4 Port PB03 Function configuration register (PB03_SEL)

Address offset: 0x4C

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															PB03_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB03_sel	PB03_sel: Port PB03 function selection. 000 ---- GPIO PB03 001 ---- SPI0_SCK SPI0 module clock signal 010 ---- TIM0_CHB Timer0 module channel B signal 011 ---- TIM1_GATE Timer1 module gate control signal 100 ---- TIM3_CH0A Timer3 module channel 0A signal 101 ---- LPTIM0_GATE Low power consumption timer module gate control signal 110 ---- XTL_OUT External 32K crystal oscillator output signal 111 ---- XTH_OUT External 32M crystal oscillator output signal

6.5.4.5 Port PB04 Function configuration register (PB04_SEL)

Address offset: 0x50

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PB04_sel	
														rw	

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB04_sel	PB04_sel: Port PB04 function selection. 000 ---- GPIO PB04 001 ---- SPI0_MISO SPI0 module Master input slave output data signal 010 ---- PCA_CH0 PCA module channel 0 capture / compare signal 011 ---- TIM2_BK Timer2 module brake signal 100 ---- UART0_CTS UART0 module CTS signal 101 ---- TIM2_GATE Timer2 module gate control signal 110 ---- TIM3_CH0B Timer3 module channel 0B signal 111 ---- LPTIM0_EXT low-power timer module external clock input signal

6.5.4.6 Port PB05 Function configuration register (PB05_SEL)

Address offset: 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														PB05_sel	
														rw	

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB05_sel	PB05_sel: Port PB05 function selection. 000 ---- GPIO PB05 001 ---- SPI0_MOSI SPI0 module master output slave input data signal 010 ---- Reserved Reserved 011 ---- TIM1_BK Timer1 module brake signal 100 ---- PCA_CH1 PCA module channel 1 capture / compare signal 101 ---- LPTIM0_GATE Low power consumption timer module gate control signal 110 ---- PCNT_S0 PCNT module input signal 0 111 ---- UART0_RTS UART0 module RTS signal

6.5.4.7 Port PB06 Function configuration register (PB06_SEL)

Address offset: 0x58

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															PB06_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB06_sel	PB06_sel: Port PB06 function selection. 000 ---- GPIO PB06 001 ---- I2C0_SCL I2C0 module clock signal 010 ---- UART0_TXD UART0 module TXD signal 011 ---- TIM1_CHB Timer1 module channel B signal 100 ---- TIM0_CHA Timer0 module channel A signal 101 ---- LPTIM0_EXT Low power consumption timer module external clock input signal 110 ---- TIM3_CH0A Timer3 module channel 0A signal 111 ---- LPTIM0_TOG Low power consumption timer module flip signal

6.5.4.8 Port PB07 Function configuration register (PB07_SEL)

Address offset: 0x5C

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															PB07_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB07_sel	PB07_sel: Port PB07 function selection. 000 ---- GPIO PB07 001 ---- I2C0_SDA I2C0 module data signal 010 ---- UART0_RXD UART0 module RXD signal 011 ---- TIM2_CHB Timer2 module channel B signal 100 ---- LPUART1_CTS LPUART1 module CTS signal 101 ---- TIM0_CHB Timer0 module channel B signal 110 ---- LPTIM0_TOGN Low-power timer module flip reverse signal 111 ---- PCNT_S1 PCNT module input signal 1

6.5.4.9 Port PB08 Function configuration register (PB08_SEL)

Address offset: 0x60

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															PB08_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB08_sel	PB08_sel: Port PB08 function selection. 000 ---- GPIO PB08 001 ---- I2C0_SCL I2C0 module clock signal 010 ---- TIM1_CHA Timer1 module channel A signal 011 ---- CAN_RX CAN module receives signal 100 ---- TIM2_CHA Timer2 module channel A signal 101 ---- TIM0_GATE Timer0 module gate control signal 110 ---- TIM3_CH2A Timer3 module channel 2A signal 111 ---- UART0_TXD UART0 module TXD signal

6.5.4.10 Port PB09 Function configuration register (PB09_SEL)

Address offset: 0x64

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															PB09_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB09_sel	PB09_sel: Port PB09 function selection. 000 ---- GPIO PB09 001 ---- I2C0_SDA I2C0 module data signal 010 ---- IR_OUT infrared output signal 011 ---- SPI1_CS SPI1 module master mode chip select signal 100 ---- TIM2_CHA Timer2 module channel A signal 101 ---- CAN_TX CAN module sends signal 110 ---- TIM2_CHB Timer2 module channel B signal 111 ---- UART0_RXD UART0 module RXD signal

6.5.4.11 Port PB1 0 Function configuration register (PB10_SEL)

Address offset: 0x68

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															PB10_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB10_sel	PB10_sel: Port PB10 function selection. 000 ---- GPIO PB10 001 ---- I2C1_SCL I2C1 module clock signal 010 ---- SPI1_SCK SPI1 module clock signal 011 ---- TIM1_CHA Timer1 module channel A signal 100 ---- LPUART0_TXD LPUART0 module TXD signal 101 ---- TIM3_CH1A Timer3 module channel 1A signal 110 ---- LPUART1_RTS LPUART1 module RTS signal 111 ---- UART1_RTS UART1 module RTS signal

6.5.4.12 Port PB11 Function configuration register (PB11_SEL)

Address offset: 0x6C

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															PB11_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB11_sel	PB11_sel: Port PB11 function selection. 000 ---- GPIO PB11 001 ---- I2C1_SDA I2C1 module data signal 010 ---- TIM1_CHB Timer1 module channel B signal 011 ---- LPUART0_RXD LPUART0 module RXD signal 100 ---- TIM2_GATE Timer2 module gate control signal 101 ---- TIM6_CHA Timer6 module channel 2A signal 110 ---- LPUART1_CTS LPUART1 module CTS signal 111 ---- UART1_CTS UART1 module CTS signal

6.5.4.13 Port PB12 Function configuration register (PB12_SEL)

Address offset: 0x70

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															PB12_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB12_sel	PB12_sel: Port PB12 function selection. 000 ---- GPIO PB12 001 ---- SPI1_CS SPI1 module host mode chip select signal 010 ---- TIM3_BK Timer3 module brake signal 011 ---- LPUART0_TXD LPUART0 module TXD signal 100 ---- TIM0_BK Timer0 module brake signal 101 ---- Reserved Reserved 110 ---- LPUART0_RTS LPUART0 module RTS signal 111 ---- TIM6_CHA Timer6 module channel 2A signal

6.5.4.14 Port PB13 Function configuration register (PB13_SEL)

Address offset: 0x74

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															PB13_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB13_sel	PB13_sel: Port PB13 function selection. 000 ---- GPIO PB13 001 ---- SPI1_SCK SPI1 module clock signal 010 ---- I2C1_SCL I2C1 module clock signal 011 ---- TIM3_CH0B Timer3 module channel 0B signal 100 ---- LPUART0_CTS LPUART0 module CTS signal 101 ---- TIM1_CHA Timer1 module channel A signal 110 ---- TIM1_GATE Timer1 module gate control signal 111 ---- TIM6_CHB Timer6 module channel 2B signal

6.5.4.15 Port PB14 Function configuration register (PB14_SEL)

Address offset: 0x78

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															PB14_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB14_sel	PB14_sel: Port PB14 function selection. 000 ---- GPIO PB14 001 ---- SPI1_MISO SPI1 module master input slave output data signal 010 ---- I2C1_SDA I2C1 module data signal 011 ---- TIM3_CH1B Timer3 module channel 1B signal 100 ---- TIM0_CHA Timer0 module channel A signal 101 ---- RTC_1HZ RTC module 1Hz output signal 110 ---- LPUART0_RTS LPUART0 module RTS signal 111 ---- TIM1_BK Timer1 module brake signal

6.5.4.16 Port PB15 Function configuration register (PB15_SEL)

Address offset: 0x7C

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															PB15_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PB15_sel	PB15_sel: Port PB15 function selection. 000 ---- GPIO PB15 001 ---- SPI1_MOSI SPI1 module master output slave input data signal 010 ---- TIM3_CH2B Timer3 module channel 2B signal 011 ---- TIM0_CHB Timer0 module channel B signal 100 ---- TIM0_GATE Timer0 module gate control signal 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- LPUART1_RXD LPUART1 module RXD signal

6.5.5 PORT PC FUNCTION SELECT REGISTER

6.5.5.1 Port PC00 Function configuration register (PC00_SEL)

Address offset: 0x80

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													PC00_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC00_sel	PC00_sel: Port PC00 function selection. 000 ---- GPIO PC00 001 ---- LPTIM0_GATE Low power consumption timer module gate control signal 010 ---- PCNT_S0 PCNT module input signal 0 011 ---- UART1_CTS UART1 module CTS signal 100 ---- UART2_RTS UART2 module RTS signal 101 ---- I2S0_MCK I2S0 module MCK signal 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.5.2 Port PC01 Function configuration register (PC01_SEL)

Address offset: 0x84

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													PC01_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC01_sel	PC01_sel: Port PC01 function selection. 000 ---- GPIO PC01 001 ---- LPTIM0_TOG Low power consumption timer module flip signal 010 ---- TIM5_CHB Timer6 module channel 1B signal 011 ---- UART1_RTS UART1 module RTS signal 100 ---- PCNT_S0FO PCNT module input signal 0 filter output 101 ---- I2S0_SD I2S0 module SD signal 110 ---- UART2_CTS UART2 module CTS signal 111 ---- Reserved Reserved

6.5.5.3 Port PC02 Function configuration register (PC02_SEL)

Address offset: 0x88

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														PC02_sel	
														rw	

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC02_sel	PC02_sel: Port PC02 function selection. 000 ---- GPIO PC02 001 ---- SPI1_MISO SPI1 module master input slave output data signal 010 ---- LPTIM0_TOGN low-power timer module flip reverse signal 011 ---- PCNT_S1 PCNT module input signal 1 100 ---- UART2_RXD UART2 module RXD signal 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.5.4 Port PC03 Function configuration register (PC03_SEL)

Address offset: 0x8C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														PC03_sel	
														rw	

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC03_sel	PC03_sel: Port PC03 function selection. 000 ---- GPIO PC03 001 ---- SPI1_MOSI SPI1 module master output slave input data signal 010 ---- LPTIM0_EXT Low power consumption timer module external clock input signal 011 ---- LPTIM0_TOGN low-power timer module flip reverse signal 100 ---- PCNT_S1FO PCNT module input signal 1 filter output 101 ---- UART2_TXD UART2 module TXD signal 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.5.5 Port PC04 Function configuration register (PC04_SEL)

Address offset: 0x90

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PC04_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC04_sel	PC04_sel: Port PC04 function selection. 000 ---- GPIO PC04 001 ---- LPUART0_TXD LPUART0 module TXD signal 010 ---- TIM2_ETR Timer2 module external clock input signal 011 ---- IR_OUT infrared output signal 100 ---- VC2_OUT VC2 module output / reverse output signal 101 ---- I2S0_WS I2S0 module WS signal 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.5.6 Port PC05 Function configuration register (PC05_SEL)

Address offset: 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															PC05_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC05_sel	PC05_sel: Port PC05 function selection. 000 ---- GPIO PC05 001 ---- LPUART0_RXD LPUART0 module RXD signal 010 ---- TIM6_CHB Timer6 module channel 2B signal 011 ---- PCA_CH4 PCA module channel 4 capture / comparison signal 100 ---- Reserved Reserved 101 ---- I2S0_SDIN I2S0 module SDIN signal 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.5.7 Port PC06 Function configuration register (PC06_SEL)

Address offset: 0x98

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PC06_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC06_sel	PC06_sel: Port PC06 function selection. 000 ---- GPIO PC06 001 ---- PCA_CH0 PCA module channel 0 capture / compare signal 010 ---- TIM4_CHA Timer6 module channel 0A signal 011 ---- TIM2_CHA Timer2 module channel A signal 100 ---- LPTIM1_GATE Low power consumption timer1 module gate control signal 101 ---- I2S1_SCK I2S1 module SCK signal 110 ---- UART3_RXD UART3 module RXD signal 111 ---- Reserved Reserved

6.5.5.8 Port PC07 Function configuration register (PC07_SEL)

Address offset: 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															PC07_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC07_sel	PC07_sel: Port PC07 function selection. 000 ---- GPIO PC07 001 ---- PCA_CH1 PCA module channel 1 capture / compare signal 010 ---- TIM5_CHA Timer6 module channel 1A signal 011 ---- TIM2_CHB Timer2 module channel B signal 100 ---- LPTIM1_EXT Low power consumption timer1 module external clock input signal 101 ---- I2S1_MCK I2S1 module MCK signal 110 ---- UART3_TXD UART3 module TXD signal 111 ---- Reserved Reserved

6.5.5.9 Port PC08 Function configuration register (PC08_SEL)

Address offset: 0xA0

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															PC08_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC08_sel	PC08_sel: Port PC08 function selection. 000 ---- GPIO PC08 001 ---- PCA_CH2 PCA module channel 2 capture / compare signal 010 ---- TIM6_CHA Timer6 module channel 2A signal 011 ---- TIM2_ETR Timer2 module external clock input signal 100 ---- LPTIM1_TOG Low power consumption timer1 module flip signal 101 ---- I2S1_SD I2S1 module SD signal 110 ---- UART3_CTS UART3 module CTS signal 111 ---- Reserved Reserved

6.5.5.10 Port PC09 Function configuration register (PC09_SEL)

Address offset: 0xA4

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															PC09_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC09_sel	PC09_sel: Port PC09 function selection. 000 ---- GPIO PC09 001 ---- PCA_CH3 PCA module channel 3 capture / compare signal 010 ---- TIM4_CHB Timer6 module channel 0B signal 011 ---- TIM1_ETR Timer1 module external clock input signal 100 ---- LPTIM1_TOGN low power timer1 module flip reverse signal 101 ---- I2S1_WS I2S1 module WS signal 110 ---- UART3_RTS UART3 module RTS signal 111 ---- Reserved Reserved

6.5.5.11 Port PC10 Function configuration register (PC10_SEL)

Address offset: 0xA8

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															PC10_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC10_sel	PC10_sel: Port PC10 function selection. 000 ---- GPIO PC10 001 ---- LPUART1_TXD LPUART1 module TXD signal 010 ---- LPUART0_TXD LPUART0 module TXD signal 011 ---- PCA_CH2 PCA module channel 2 capture/comparison signal 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.5.12 Port PC11 Function configuration register (PC11_SEL)

Address offset: 0xAC

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															PC11_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC11_sel	PC11_sel: Port PC11 function selection. 000 ---- GPIO PC11 001 ---- LPUART1_RXD LPUART1 module RXD signal 010 ---- LPUART0_RXD LPUART0 module RXD signal 011 ---- PCA_CH3 PCA module channel 3 capture / compare signal 100 ---- PCNT_S0FO PCNT module input signal 0 filter output 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.5.13 Port PC12 Function configuration register (PC12_SEL)

Address offset: 0xB0

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															PC12_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC12_sel	PC12_sel: Port PC12 function selection. 000 ---- GPIO PC12 001 ---- LPUART0_TXD LPUART0 module RXD signal 010 ---- LPUART1_TXD LPUART1 module TXD signal 011 ---- PCA_CH4 PCA module channel 4 capture / comparison signal 100 ---- PCNT_S1FO PCNT module input signal 1 filter output 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.5.14 Port PC13 Function configuration register (PC13_SEL)

Address offset: 0xB4

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															PC13_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC13_sel	PC13_sel: Port PC13 function selection. 000 ---- GPIO PC13 001 ---- Reserved Reserved 010 ---- RTC_1HZ RTC module 1Hz output signal 011 ---- TIM3_CH1B Timer3 module channel 1B signal 100 ---- Reserved Reserved 101 ---- I2S0_SCK I2S0 module SCK signal 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.5.15 Port PC14 Function configuration register (PC14_SEL)

Address offset: 0xB8

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															PC14_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC14_sel	PC14_sel: Port PC14 function selection. 000 ---- GPIO PC14 001 ---- Reserved Reserved 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.5.16 Port PC15 Function configuration register (PC15_SEL)

Address offset: 0xBC

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															PC15_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PC15_sel	PC15_sel: Port PC15 function selection. 000 ---- GPIO PC15 001 ---- Reserved Reserved 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6 Port PD Function selection register

6.5.6.1 Port PD00 Function configuration register (PD00_SEL)

Address offset: 0xC0

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													PD00_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD00_sel	Port PD00 function selection. 000 ---- GPIO PD00 001 ---- CAN_RX CAN module receives signal 010 ---- SPI1_CS SPI1 module master mode chip select signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.2 Port PD01 Function configuration register (PD01_SEL)

Address offset: 0xC4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PD01_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD01_sel	Port PD01 function selection. 000 ---- GPIO PD01 001 ---- CAN_TX CAN module sends signal 010 ---- SPI1_SCK SPI1 module clock signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.3 Port PD02 Function configuration register (PD02_SEL)

Address offset: 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															PD02_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD02_sel	PD02_sel: Port PD02 function selection. 000 ---- GPIO PD02 001 ---- PCA_ECI PCA module external clock input signal 010 ---- LPUART0_RTS LPUART0 module RTS signal 011 ---- TIM1_ETR Timer1 module external clock input signal 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.4 Port PD03 Function configuration register (PD03_SEL)

Address offset: 0xCC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PD03_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD03_sel	Port PD03 function selection. 000 ---- GPIO PD03 001 ---- UART1_CTS UART1 module CTS signal 010 ---- SPI1_MISO SPI1 module master input slave output data signal 011 ---- LPTIM1_TOG Low power consumption timer1 module flip signal 100 ---- I2S1_SCK I2S1 module SCK signal 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.5 Port PD04 Function configuration register (PD04_SEL)

Address offset: 0xD0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PD04_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD04_sel	Port PD04 function selection. 000 ---- GPIO PD04 001 ---- UART1_RTS UART1 module RTS signal 010 ---- SPI1_MOSI SPI1 module master output slave input data signal 011 ---- LPTIM1_TOGN low-power timer1 module flip reverse signal 100 ---- I2S1_MCK I2S1 module MCK signal 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.6 Port PD05 Function configuration register (PD05_SEL)

Address offset: 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															PD05_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD05_sel	Port PD05 function selection. 000 ---- GPIO PD05 001 ---- UART1_TXD UART1 module TXD signal 010 ---- LPTIM1_GATE Low power consumption timer1 module gate control signal 011 ---- CAN_STBY CAN module STBY signal 100 ---- I2S1_SD I2S1 module SD signal 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.7 Port PD06 Function configuration register (PD06_SEL)

Address offset: 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														PD06_sel	
														rw	

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD06_sel	PD06_sel: Port PD06 function selection. 000 ---- GPIO PD06 001 ---- UART1_RXD UART1 module RXD signal 010 ---- LPTIM1_EXT Low power consumption timer1 module external clock input signal 011 ---- Reserved Reserved 100 ---- I2S1_WS I2S1 module WS signal 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.8 Port PD07 Function configuration register (PD0 7_SEL)

Address offset: 0xDC

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														PD07_sel	
														rw	

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD07_sel	PD07_sel: Port PD07 function selection. 000 ---- GPIO PD07 001 ---- UART1_TXD UART1 module TXD signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- I2S1_SDIN I2S1 module SDIN signal 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.9 Port PD08 Function configuration register (PD08_SEL)

Address offset: 0xE0

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														PD08_sel	
														rw	

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD08_sel	PD08_sel: Port PD08 function selection. 000 ---- GPIO PD08 001 ---- LPUART0_TXD LPUART0 module TXD signal 010 ---- I2S0_SCK I2S0 module SCK signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.10 Port PD09 Function configuration register (PD09_SEL)

Address offset: 0xE4

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														PD09_sel	
														rw	

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD09_sel	PD09_sel: Port PD09 function selection. 000 ---- GPIO PD09 001 ---- LPUART0_RXD LPUART0 module RXD signal 010 ---- I2S0_MCK I2S0 module MCK signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.11 Port PD10 Function configuration register (PD10_SEL)

Address offset: 0xE8

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															PD10_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD10_sel	PD10_sel: Port PD10 function selection. 000 ---- GPIO PD10 001 ---- LPUART0_TXD LPUART0 module TXD signal 010 ---- I2S0_SD I2S0 module SD signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.12 Port PD11 Function configuration register (PD11_SEL)

Address offset: 0xEC

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															PD11_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD11_sel	PD11_sel: Port PD11 function selection. 000 ---- GPIO PD11 001 ---- LPUART0_CTS LPUART0 module CTS signal 010 ---- I2S0_WS I2S0 module WS signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.13 Port PD12 Function configuration register (PD12_SEL)

Address offset: 0xF0

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															PD12_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD12_sel	PD12_sel: Port PD12 function selection. 000 ---- GPIO PD12 001 ---- LPUART0_RTS LPUART0 module RTS signal 010 ---- UART2_RTS UART2 module RTS signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.14 Port PD13 Function configuration register (PD13_SEL)

Address offset: 0xF4

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															PD13_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD13_sel	PD13_sel: Port PD13 function selection. 000 ---- GPIO PD13 001 ---- UART2_RXD UART2 module RXD signal 010 ---- I2S0_SDIN I2S0 module SDIN signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.15 Port PD14 Function configuration register (PD14_SEL)

Address offset: 0xF8

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														PD14_sel	
														rw	

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD14_sel	PD14_sel: Port PD14 function selection. 000 ---- GPIO PD14 001 ---- UART2_TXD UART2 module TXD signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.6.16 Port PD15 Function configuration register (PD15_SEL)

Address offset: 0xFC

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														PD15_sel	
														rw	

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PD15_sel	PD15_sel: Port PD15 function selection. 000 ---- GPIO PD15 001 ---- CRS_SYNC 010 ---- UART2_CTS UART2 module CTS signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7 Port PE Function Selection Register

6.5.7.1 Port PE00 Function configuration register (PE00_SEL)

Address offset: 0x1000

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													PE00_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE00_sel	Port PE00 function selection. 000 ---- GPIO PE00 001 ---- TIM1_CHA Timer1 module channel A signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.2 Port PE01 Function configuration register (PE01_SEL)

Address offset: 0x1004

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													PE01_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE01_sel	Port PE01 function selection. 000 ---- GPIO PE01 001 ---- TIM2_CHA Timer2 module channel A signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.3 Port PE02 Function configuration register (PE02_SEL)

Address offset: 0x1008

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-															PE02_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE02_sel	Port PE02 function selection. 000 ---- GPIO PE02 001 ---- PCA_ECI PCA module external clock input signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.4 Port PE03 Function configuration register (PE03_SEL)

Address offset: 0x100C

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															PE03_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE03_sel	Port PE03 function selection. 000 ---- GPIO PE03 001 ---- PCA_CH0 PCA module channel 0 capture/compare signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.5 Port PE04 Function configuration register (PE04_SEL)

Address offset: 0x1010

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															PE04_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE04_sel	Port PE04 function selection. 000 ---- GPIO PE04 001 ---- PCA_CH1 PCA module channel 1 capture / compare signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.6 Port PE05 Function configuration register (PE05_SEL)

Address offset: 0x1014

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															PE05_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE05_sel	Port PE05 function selection. 000 ---- GPIO PE05 001 ---- PCA_CH2 PCA module channel 2 capture / comparison signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.7 Port PE06 Function configuration register (PE06_SEL)

Address offset: 0x1018

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															PE06_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE06_sel	Port PE06 function selection. 000 ---- GPIO PE06 001 ---- PCA_CH3 PCA module channel 3 capture / comparison signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.8 Port PE07 Function configuration register (PE07_SEL)

Address offset: 0x101C

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															PE07_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE07_sel	Port PE07 function selection. 000 ---- GPIO PE07 001 ---- TIM3_ETR Timer3 module external clock input signal 010 ---- LPTIM1_GATE Low power consumption timer1 module gate control signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.9 Port PE08 Function configuration register (PE08_SEL)

Address offset: 0x1020

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															PE08_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE08_sel	PE08_sel: Port PE08 function selection. 000 ---- GPIO PE08 001 ---- TIM3_CH0B Timer3 module channel 0B signal 010 ---- LPTIM1_EXT Low power consumption timer1 module external clock input signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.10 Port PE09 Function configuration register (PE09_SEL)

Address offset: 0x1024

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															PE09_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE09_sel	Port PE09 function selection. 000 ---- GPIO PE09 001 ---- TIM3_CH0A Timer3 module channel 0A signal 010 ---- LPTIM1_TOG low power consumption timer1 module flip signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.11 Port PE10 Function configuration register (PE10_SEL)

Address offset: 0x1028

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															PE10_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE10_sel	PE10_sel: Port PE10 function selection. 000 ---- GPIO PE10 001 ---- TIM3_CH1B Timer3 module channel 1B signal 010 ---- LPTIM1_TOGN Low-power timer1 module flip reverse signal 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.12 Port PE11 Function configuration register (PE11_SEL)

Address offset: 0x102C

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															PE11_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE11_sel	PE11_sel: Port PE11 function selection. 000 ---- GPIO PE11 001 ---- TIM3_CH1A Timer3 module channel 1A signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.13 Port PE12 Function configuration register (PE12_SEL)

Address offset: 0x1030

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															PE12_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE12_sel	PE12_sel: Port PE12 function selection. 000 ---- GPIO PE12 001 ---- TIM3_CH2B Timer3 module channel 2B signal 010 ---- SPI0_CS SPI0 module master mode chip select signal 011 ---- UART3_CTS UART3 module CTS signal 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.14 Port PE13 Function configuration register (PE13_SEL)

Address offset: 0x1034

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															PE13_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE13_sel	PE13_sel: Port PE13 function selection. 000 ---- GPIO PE13 001 ---- TIM3_CH2A Timer3 module channel 2A signal 010 ---- SPI0_S CK SPI0 module clock signal 011 ---- UART3_RTS UART3 module RTS signal 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.15 Port PE14 Function configuration register (PE14_SEL)

Address offset: 0x1038

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															PE14_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE14_sel	PE14_sel: Port PE14 function selection. 000 ---- GPIO PE14 001 ---- TIM3_CH0B Timer3 module channel 0B signal 010 ---- SPI0_MISO SPI0 module Master input slave output data signal 011 ---- UART3_RXD UART3 module RXD signal 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.7.16 Port PE15 Function configuration register (PE15_SEL)

Address offset: 0x103C

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															PE15_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PE15_sel	PE15_sel: Port PE15 function selection. 000 ---- GPIO PE15 001 ---- TIM3_BK Timer3 module brake signal 010 ---- SPI0_MOSI SPI0 module host output slave input data signal 011 ---- UART3_TXD UART3 module TXD signal 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.8 Port PF Function Selection Register

6.5.8.1 Port PF00 Function configuration register (PF00_SEL)

Address offset: 0x1040

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													PF00_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PF00_sel	Port PF00 function selection. 000 ---- GPIO PF00 001 ---- I2C0_SDA I2C0 module data signal 010 ---- CRS_SYNC 011 ---- UART1_TXD UART1 module TXD signal 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.8.2 Port PF01 Function configuration register (PF01_SEL)

Address offset: 0x1044

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													PF01_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PF01_sel	PF01_sel: Port PF01 function selection. 000 ---- GPIO PF01 001 ---- I2C0_SCL I2C0 module clock signal 010 ---- TIM4_CHB Timer6 module channel 0B signal 011 ---- UART1_RXD UART1 module RXD signal 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.8.3 Port PF02 Function configuration register (PF02_SEL)

Address offset: 0x1048

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													PF02_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PF02_sel	PF02_sel: Port PF02 function selection. 000 ---- GPIO PF02 001 ---- Reserved Reserved 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.8.4 Port PF03 Function configuration register (PF03_SEL)

Address offset: 0x104C

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													PF03_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PF03_sel	PF03_sel: Port PF03 function selection. 000 ---- GPIO PF03 001 ---- Reserved Reserved 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.8.5 Port PF04 Function configuration register (PF04_SEL)

Address offset: 0x1050

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															PF04_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PF04_sel	PF04_sel: Port PF04 function selection. 000 ---- GPIO PF04 001 ---- Reserved Reserved 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.8.6 Port PF05 Function configuration register (PF05_SEL)

Address offset: 0x1054

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															PF05_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PF05_sel	PF05_sel: Port PF05 function selection. 000 ---- GPIO PF05 001 ---- Reserved Reserved 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.8.7 Port PF06 Function configuration register (PF06_SEL)

Address offset: 0x1058

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															PF06_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PF06_sel	PF06_sel: Port PF06 function selection. 000 ---- GPIO PF06 001 ---- I2C1_SCL I2C1 module clock signal 010 ---- LPUART1_CTS LPUART1 module CTS signal 011 ---- UART0_CTS UART0 module CTS signal 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.8.8 Port PF07 Function configuration register (PF07_SEL)

Address offset: 0x105C

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															PF07_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PF07_sel	PF07_sel: Port PF07 function selection. 000 ---- GPIO PF07 001 ---- I2C1_SDA I2C1 module data signal 010 ---- LPUART1_RTS LPUART1 module RTS signal 011 ---- UART0_RTS UART0 module RTS signal 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.8.9 Port PF09 Function configuration register (PF09_SEL)

Address offset: 0x1064

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															PF09_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PF09_sel	PF09_sel: Port PF09 function selection. 000 ---- GPIO PF09 001 ---- TIM0_CHA Timer0 module channel A signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.8.10 Port PF10 Function configuration register (PF10_SEL)

Address offset: 0x106 8

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															PF10_sel
															rw

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PF10_sel	PF10_sel: Port PF10 function selection. 000 ---- GPIO PF10 001 ---- TIM0_CHB Timer0 module channel B signal 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

6.5.8.11 Port PF11 Function configuration register (PF11_SEL)

Address offset: 0x106C

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													PF11_sel		
													rw		

Bit	Marking	Functional description
31:3	Reserved	Keep
2:0	PF11_sel	PF11_sel: Port PF11 function selection. 000 ---- GPIO PF11 001 ---- Reserved Reserved 010 ---- Reserved Reserved 011 ---- Reserved Reserved 100 ---- Reserved Reserved 101 ---- Reserved Reserved 110 ---- Reserved Reserved 111 ---- Reserved Reserved

7 I2C bus (I2C)

7.1 Introduction

I2C is a two-wire bidirectional synchronous serial bus, which uses a clock line and a data line to transmit information between two devices connected to the bus, providing a simple and efficient method for data exchange between devices. Each device connected to the bus has a unique address, and any device can be used as both a master and a slave, but only one master is allowed at the same time. I2C standard is a real multi-master bus with a conflict detection mechanism and an arbitration mechanism. It can use the arbitration mechanism to avoid data conflicts and protect data when multiple Masters request control of the bus at the same time.

I2C bus controller can meet various specifications of the I2C bus and support all transmission modes communicating with the I2C bus. The I2C logic can handle byte transfers autonomously. It can keep track of the serial transfer, and there is a status register (I2Cx_STAT) that can reflect the status of the I2C bus controller and the I2C bus.

7.2 Main characteristics

I2C controller supports the following features:

- Support four working modes of master sending/receiving and slave sending/receiving
- Support standard (100Kbps) / fast (400Kbps) / high speed (1Mbps) three working rates
- Support 7-bit addressing function
- Support noise filtering function
- Support broadcast address
- Support interrupt status query function

7.3 protocol description

The I2C bus uses "SCL" (serial clock bus) and "SDA" (serial data bus) to connect devices to transfer information. The Master computer outputs a serial clock signal on the SCL line, and the data is transmitted on the SDA line, and each byte is transmitted (the highest bit MSB starts to be transmitted), followed by an acknowledge bit. One SCL clock pulse transfers one data bit.

7.3.1 Data transmission on the I2C bus

Usually the standard I2C transmission protocol consists of four parts: start (S) or repeated start signal (Sr), slave address and read and write bits, transmission data, and stop signal (P).

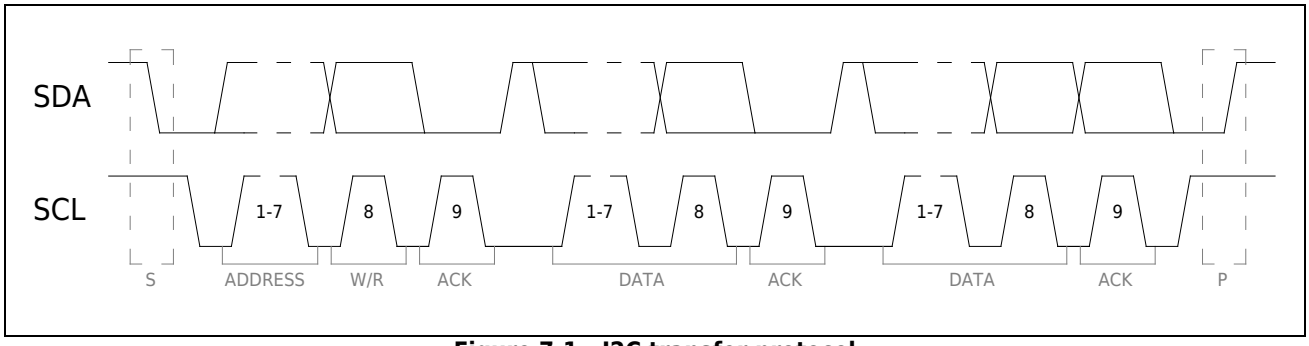


Figure 7-1 I2C transfer protocol

- Start signal, repeat start signal, stop signal
  - When the bus is in an idle state (the SCL and SDA lines are high at the same time), a signal from high to low appears on the SDA line, indicating that a start signal is generated on the bus.
  - A repeated start signal occurs when there is no stop signal between two start signals. This method is used by a master to communicate with another slave or the same slave with a different transfer direction (for example: from writing to the device to reading from the device) without releasing the bus.
  - When the SCL line is high, a low-to-high signal appears on the SDA line, which is defined as a stop signal. The master sends a stop signal to the bus to end the data transfer.

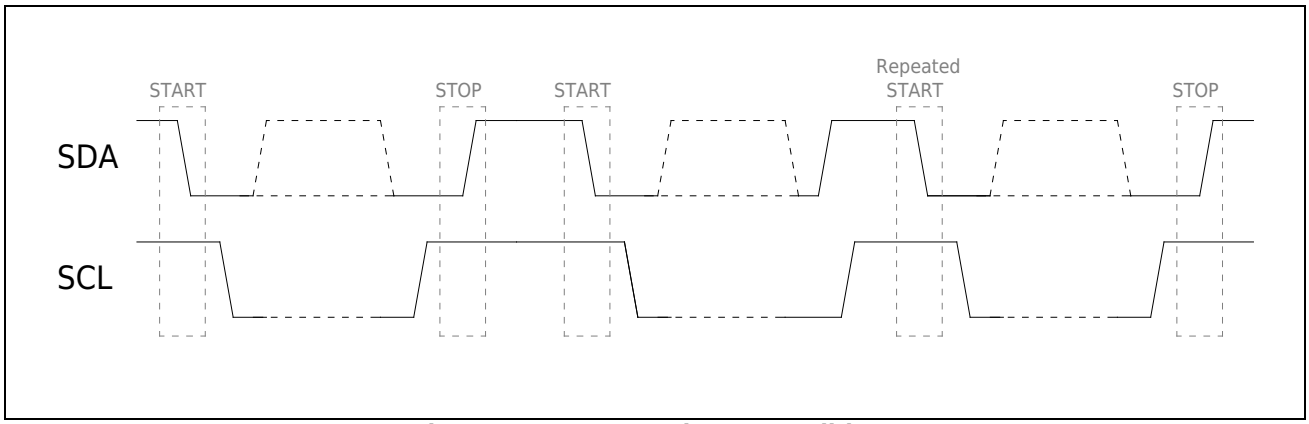


Figure 7-2 START and STOP conditions

- Slave address and read/write bits
  - When the start signal is generated, the Master immediately transmits the first byte of data: 7 -bit slave address + read and write bits, the read and write bits control the data transmission direction of the slave (0: write; 1: read). A slave addressed by the master will acknowledge by pulling SDA low on the ninth SCL clock cycle.
- Transfer data
  - During data transfer, one SCL clock pulse transfers one data bit, and the SDA line can only change when SCL is low.

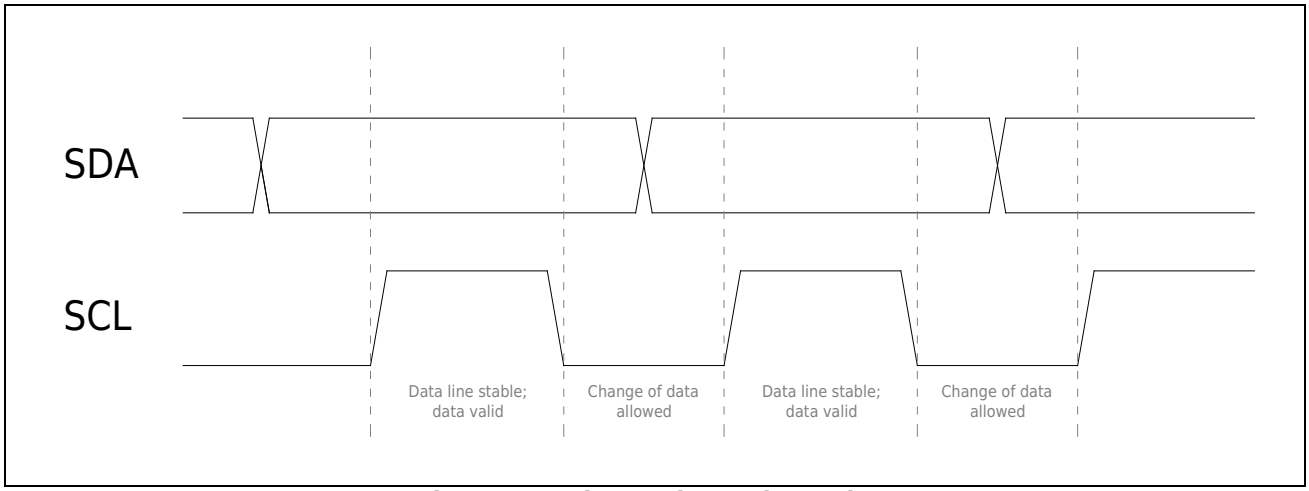


Figure 7-3 Bit transfer on the I2C bus

7.3.2 Acknowledgment on the I2C bus

Each byte transferred is followed by an acknowledge bit. By pulling the SDA line low, the receiver is allowed to echo the transmitter. ACK is a low-level signal. When the clock signal is high, SDA remains low to indicate that the receiving end has successfully received the data from the sending end.

(NACK) is generated on the slave, the Master can generate a stop signal to exit data transmission, or generate a repeated start signal to start a new round of data transmission. When the Master is used as a receiving device, a non-response signal (NACK) occurs, and the slave releases the SDA line, causing the Master to generate a stop signal or a repeated start signal.

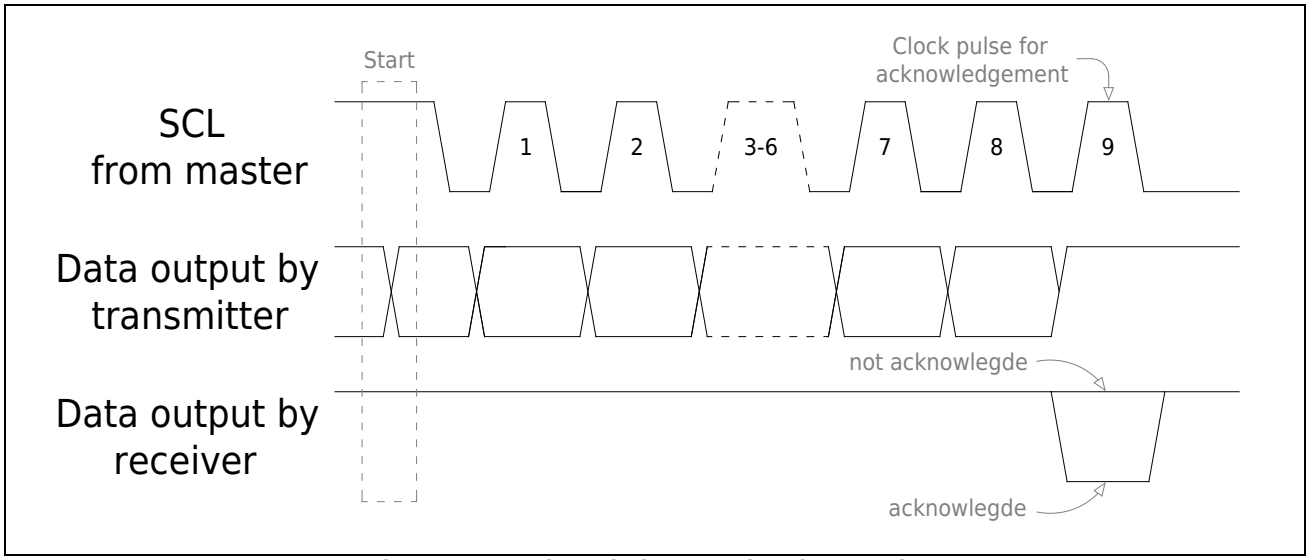


Figure 7-4 Acknowledgment signal on I2C bus

7.3.3 Arbitration on the I2C bus

Arbitration on the I2C bus is divided into two parts: synchronization on the SCL line and arbitration on the SDA line

- Synchronization on the SCL line (clock synchronization)
  - Since the I2C bus has the logic function of "AND", as long as one node on the SCL line sends a low level, the bus will show a low level. The bus can only behave as a high level when all nodes are sending a high level. Therefore, the clock low time is determined by the device with the longest clock level period, and the clock high time is determined by the device with the shortest clock high period. Due to the characteristics of I2C, when multiple Masters send clock signals at the same time, a unified clock signal is represented on the bus. If the slave wants the Master to reduce the transmission speed, it can notify the Master by actively pulling SCL low to extend its low level time. When the Master finds that the SCL level is pulled low when preparing for the next transmission, it waits until the slave completes the operation. And release control of the SCL line.
- Arbitration on SDA Online
  - The arbitration on the SDA line is also due to the logical function of the "AND" of the I2C bus. After the master sends data, it decides whether to exit the competition by comparing the data on the bus. The master that loses the arbitration immediately switches to the unaddressed slave state to ensure that it can be addressed by the master that wins the arbitration. A master that loses arbitration continues to output clock pulses (on SCL) until the current serial byte has been sent. This principle can ensure that the I2C bus will not lose data when multiple Masters attempt to control the bus.

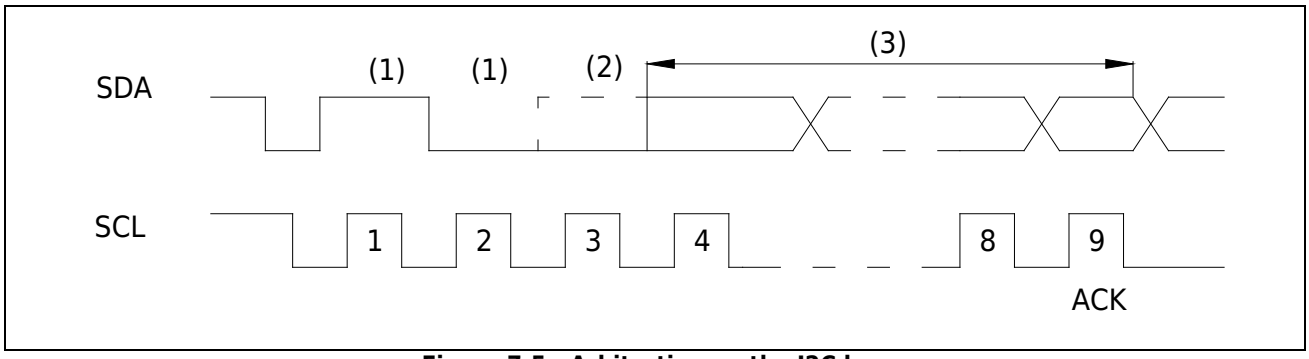


Figure 7-5 Arbitration on the I2C bus

a) Another device sends serial data;

b) Another device first negates a logic 1 sent by the I2C master by pulling SDA low (dotted line). Arbitration is lost, I2C enters slave receive mode;

c) At this time, I2C is in the slave receiving mode, but still generates clock pulses until the current byte is sent. I2C will not generate a clock pulse for the next byte transfer. Once arbitration is won, the data transfer on SDA is initiated by the new master.

7.4 Functional description

The I2C bus uses two wires to transfer information between devices connected to the bus "SCL" (Serial Clock Line) and "SDA" (Serial Data Line). Filtering logic can filter glitches on the data bus to protect data integrity. Since there are only non-directional ports, the I2C component requires the use of open-drain buffers to the pins. Each device connected to the bus can be addressed by a specific address using software. The I2C standard is a true multi-master bus with a collision detection mechanism and an arbitration mechanism. It prevents data collisions when two or more Masters start transmitting data at the same time. I2C bus can be queried in the status register.

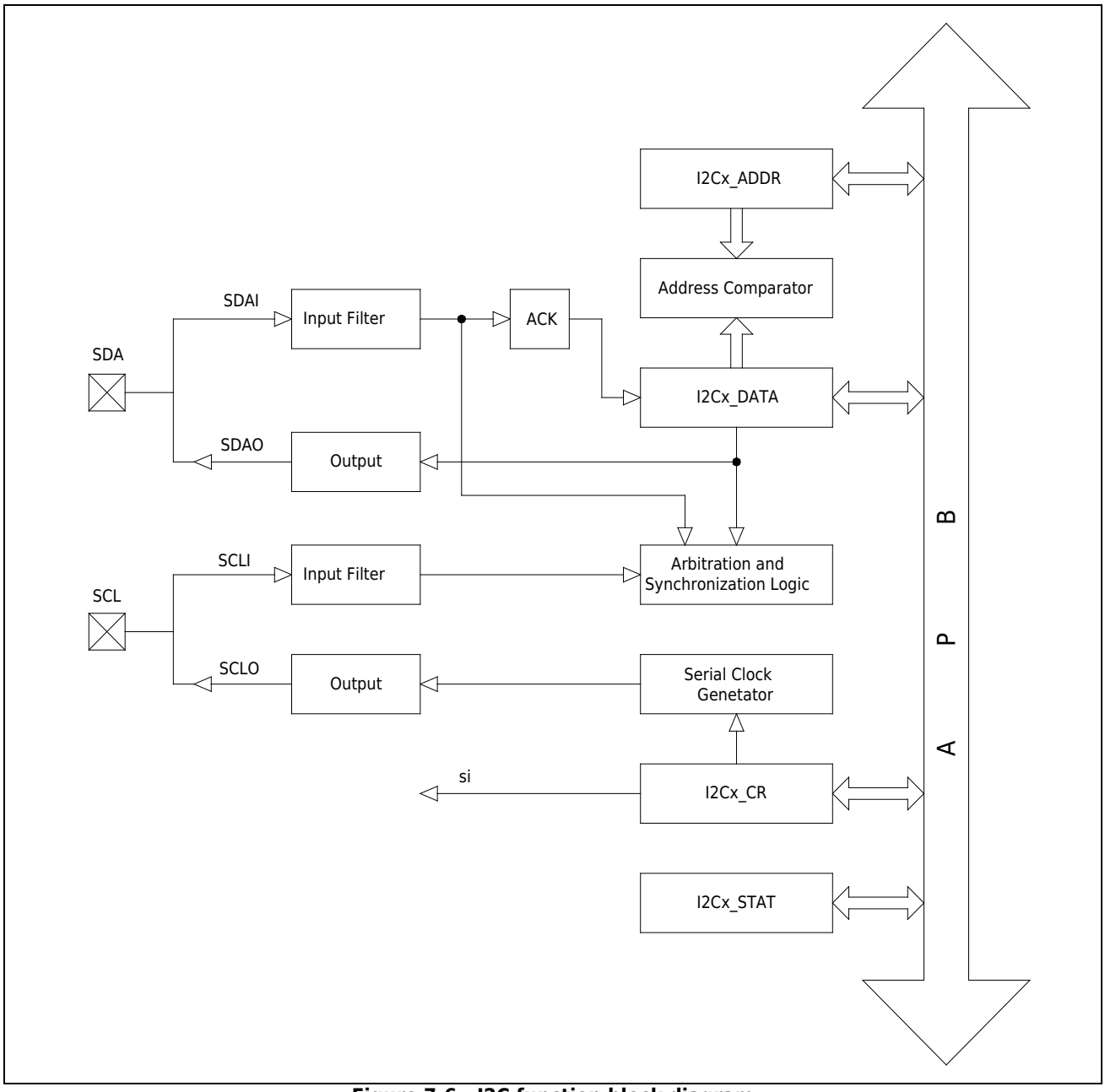


Figure 7-6 I2C function block diagram

7.4.1 Serial clock generator

The serial clock generator uses an 8 -bit counter as the baud rate generator. The frequency relationship between the SCL signal and the PCLK signal is Fscl = Fpclk / 8 / (I2Cx_TM.tm+1), where I2Cx_TM.tm should be greater than 0.

The following table lists the output frequency value of SCL signal when PCLK frequency is combined with I2Cx_TM.tm.

Table 7-1 I2C clock signal baud rate

PCLK (KHz)	I2Cx_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

7.4.2 Input filter

The input signal is synchronous with PCLK, and the spike pulse signal lower than the period of PCLK will be filtered out.

When this module is used as the Master, if the value of I2C_TM is less than or equal to 9, I2C_CR.H1M should be set to 1; if the value of I2C_TM is greater than 9, I2C_CR.H1M should be set to 0.

When this module is used as a slave, if the ratio of PCLK to SCL frequency is less than or equal to 30, I2C_CR.H1M should be set to 1; if the ratio of PCLK to SCL frequency is greater than 30, I2C_CR.H1M should be set to 0.

7.4.3 Address comparator

I2C comparator compares its own slave address with the received 7 -bit slave address. It can program its own slave address using the "I2Cx_ADDR" register. And it will be compared with the first received 8 -bit byte or broadcast address (0x00) according to the "i2cadr" bit of the "I2Cx_ADDR" register. If any one is the same, the "si" bit of the "I2Cx_CR" register will be set to 1 and an interrupt request will be generated.

7.4.4 response flag

"aa" flag in the "I2Cx_CR" register is the answer flag. When the "aa" bit is 1, the I2C module responds with an acknowledgment bit after receiving the data, and when the "aa" bit is 0, the I2C module returns a non-acknowledgement bit after receiving the data.

7.4.5 interrupt generator

"si" flag in the "I2Cx_CR" register is an interrupt flag. "si" flag is set to 1 whenever the value of the status register (I2Cx_STAT) changes (except to 0xF8). When an interrupt is generated, the status of the I2C bus can be obtained by querying the status register (I2Cx_STAT) to determine the actual source of the interrupt. In order to proceed to the next step, the "si" flag must be cleared by software.

7.4.6 Operating mode

The I2C component can realize 8 -bit bidirectional data transmission, the transmission rate can reach 100Kbps in standard mode, 400Kbps in high-speed mode, and 1Mbps in ultra-high-speed mode, and can work in four modes: Master send mode, Master receiving mode, slave receiving mode, slave sending mode. There is also a special mode, general call mode, which operates in a similar way to slave receive mode.

- Host send mode

  The Master sends multiple bytes to the slave, and the Master generates a clock, so it is necessary to fill in the set value in I2Cx_TM. I2Cx_CR.sta needs to be set to 1 in master send mode. When the bus is free, the master initiates a start bit START. I2Cx_CR.si is set to 1 if successful. Next, write the slave address and write bit (SLA+W) into I2Cx_DATA, and after clearing the "si" bit, SLA+W is sent on the bus.

  Next, write the slave address and write bit (SLA+W) into I2Cx_DATA, and after clearing the "si" bit, SLA+W is sent on the bus. The data is then sent according to the user-defined format. After all the data is sent, set I2Cx_CR.sto to 1, clear the "si" bit and send a STOP signal, or send a repeated start signal for a new round of data transmission.

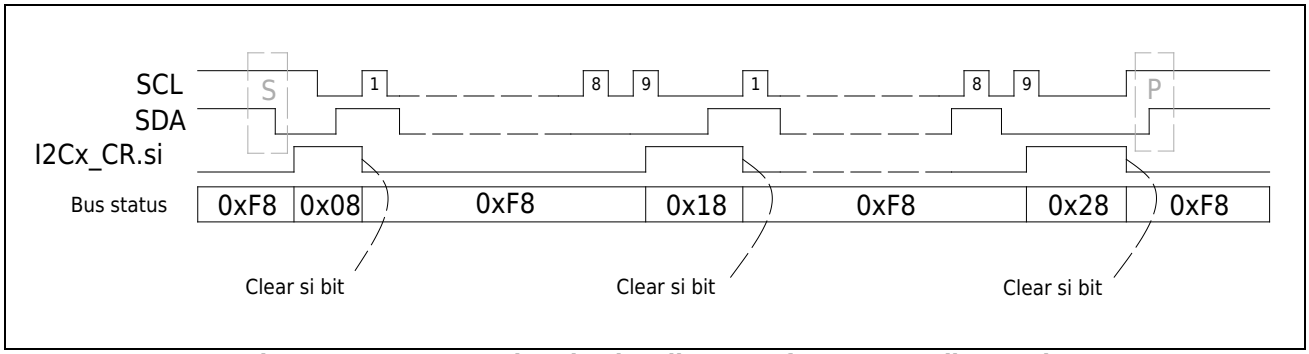


Figure 7-7 Data synchronization diagram of master sending mode

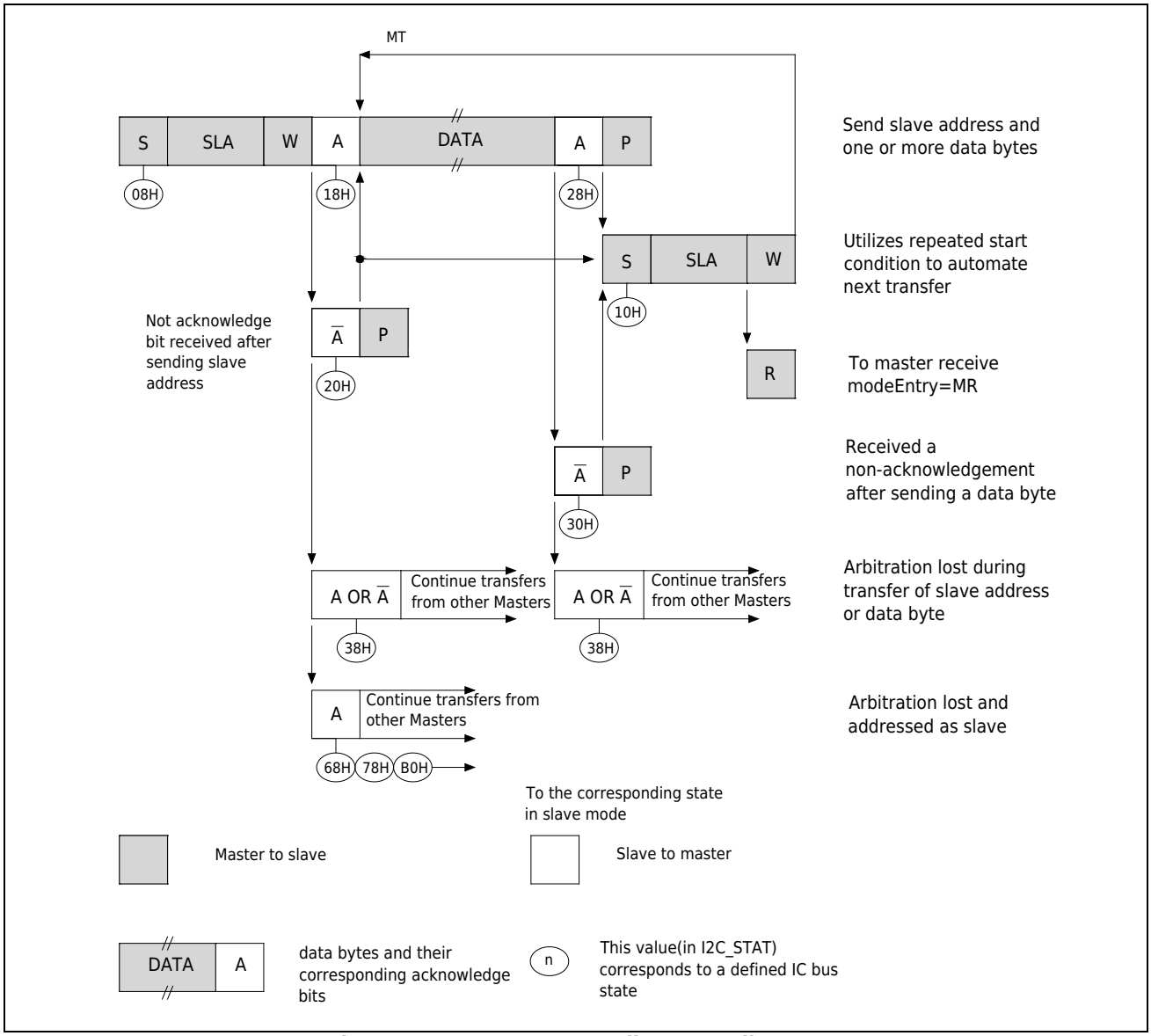


Figure 7-8 I2C master sending state diagram

- Host receiving mode

  The master receives the mode, and the data is transmitted by the slave. The initialization setting is the same as the master sending mode, after the master sends the start bit, I2Cx_DATA should be written to the slave address and "read bit" (SLA+R). I2Cx_CR.si is set to 1 after the slave acknowledge bit ACK is received. " si" is cleared to 0, it starts to receive slave data. If I2Cx_CR.aa is 1, the master responds with an acknowledgment bit after receiving the data; if it is 0, the master does not respond with a NACK after receiving the data. Then the Master can send a stop signal or repeat the start signal to start the next round of data transmission.

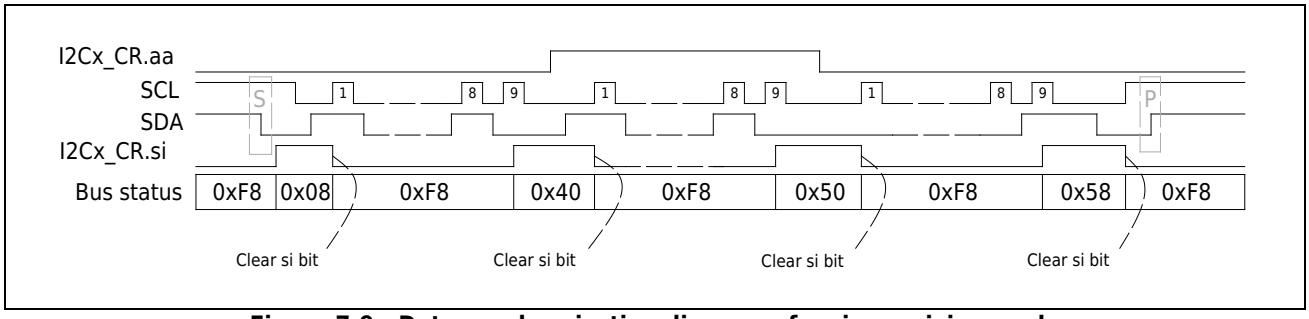


Figure 7-9 Data synchronization diagram of main receiving mode

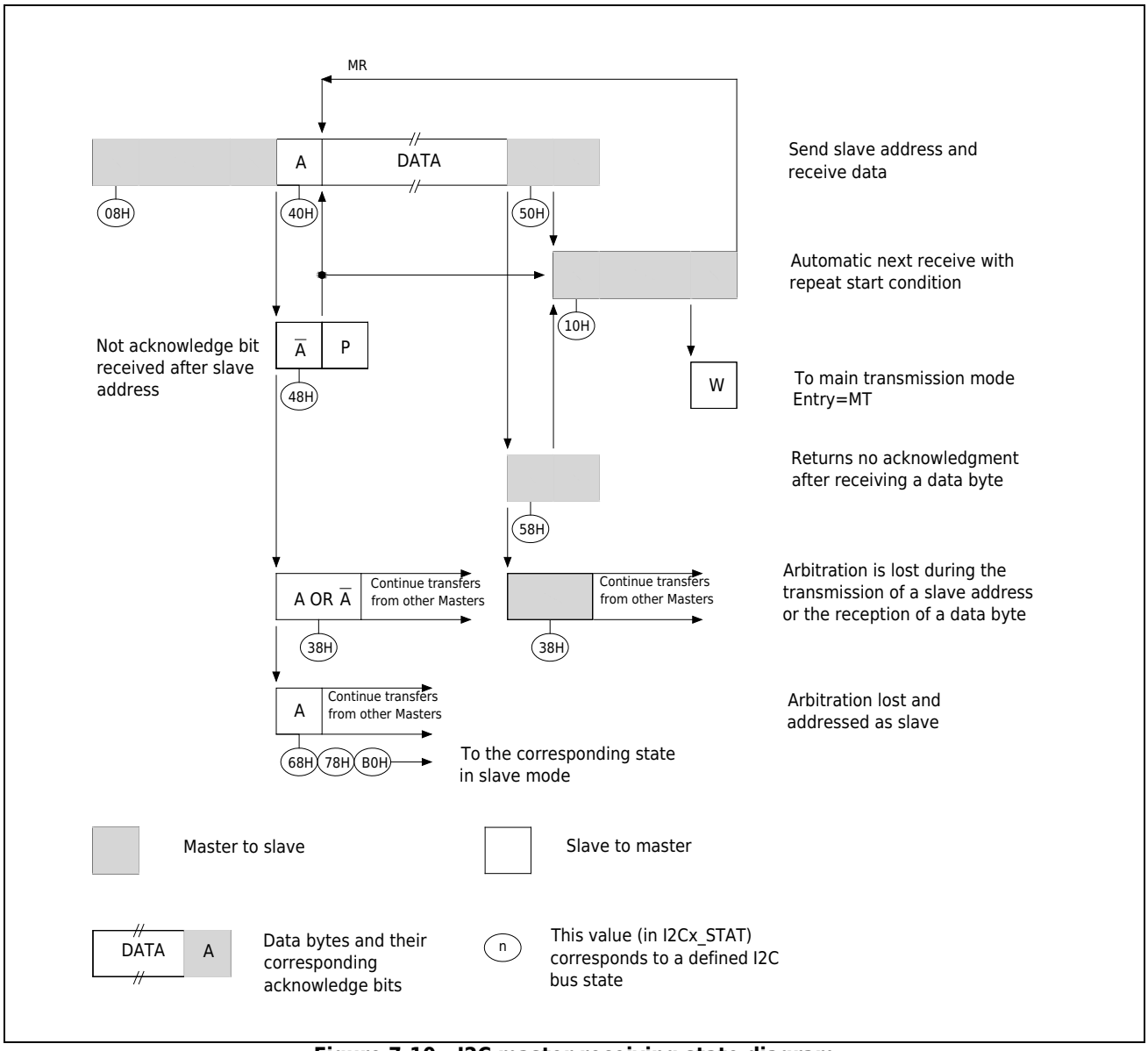


Figure 7-10 I2C master receiving state diagram

- Slave receiving mode

  In slave receiving mode, the slave receives data from the master. Before the transmission starts, I2Cx_ADDR should be written to the slave address, and I2Cx_CR.aa is set to 1 to respond to the addressing of the master. After the above initialization, the slave enters idle mode and waits

for a "write" signal (SLA+W). If the master fails to arbitrate, it will also directly enter the slave receiving mode.

When the slave is addressed by the "write" signal SLA+W, the "si" bit needs to be cleared to receive data from the master. I2Cx_CR.aa=0 during the transmission, the slave will return NACK in the next byte, the slave will also turn into an unaddressed slave, the connection with the master is terminated, no more data is received, and I2C_DATA remains before received data. Slave address recognition can be restored by setting "aa", which means that the "aa" bit temporarily detaches the I2C module from the I2C bus.

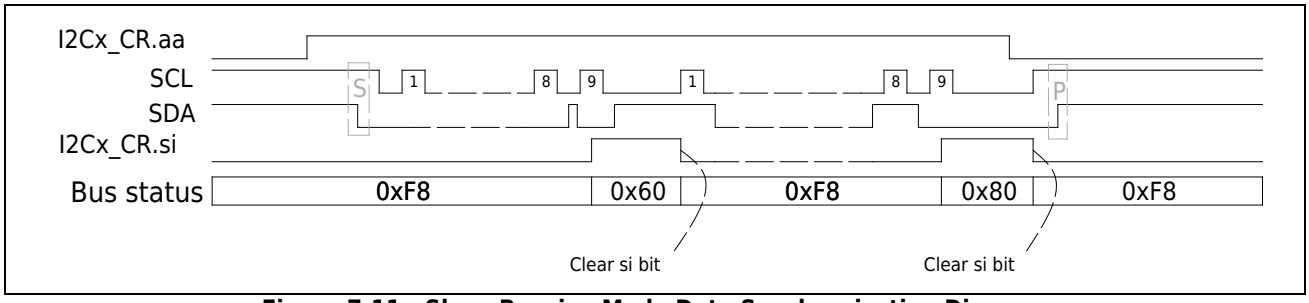


Figure 7-11 Slave Receive Mode Data Synchronization Diagram

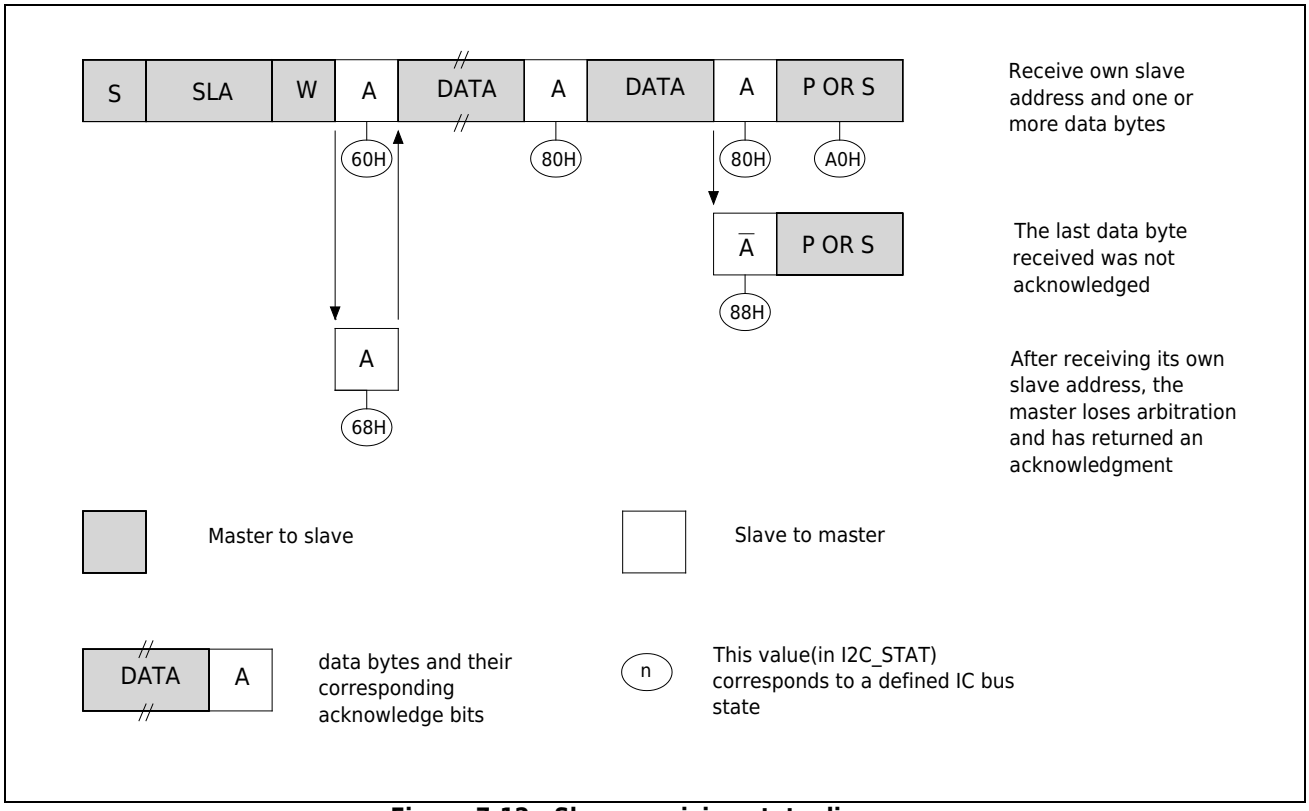


Figure 7-12 Slave receiving state diagram

- Slave mode

  In the slave sending mode, the data is sent from the slave to the master. After initializing the I2Cx_ADDR and I2Cx_CR.aa values, the device waits until its own address is addressed by the "read" signal (SLA+R). If the master fails to arbitrate, it can also enter the slave sending mode. When the slave is addressed by the "read" signal SLA+R, "si" needs to be cleared to send data to the master. Normally the Master returns an acknowledge bit after each byte of data received. I2Cx_CR.aa is cleared during transmission, the slave will send the last byte of data, and send all 1 data in the next transmission, and turn itself into an unaddressed slave.

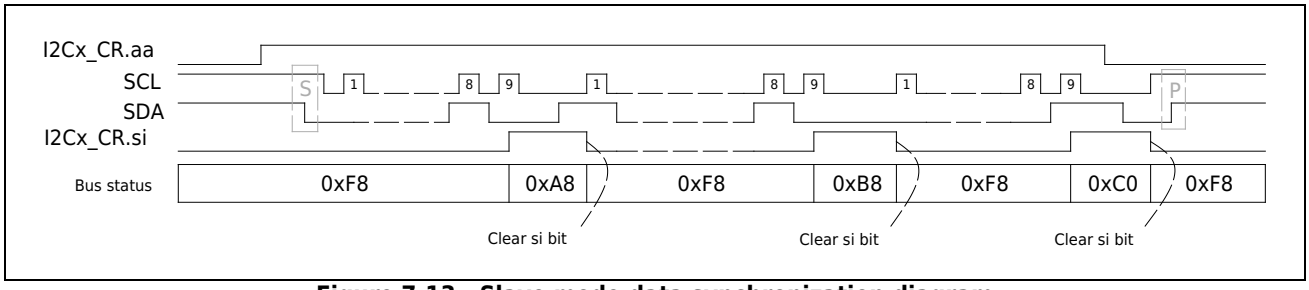


Figure 7-13 Slave mode data synchronization diagram

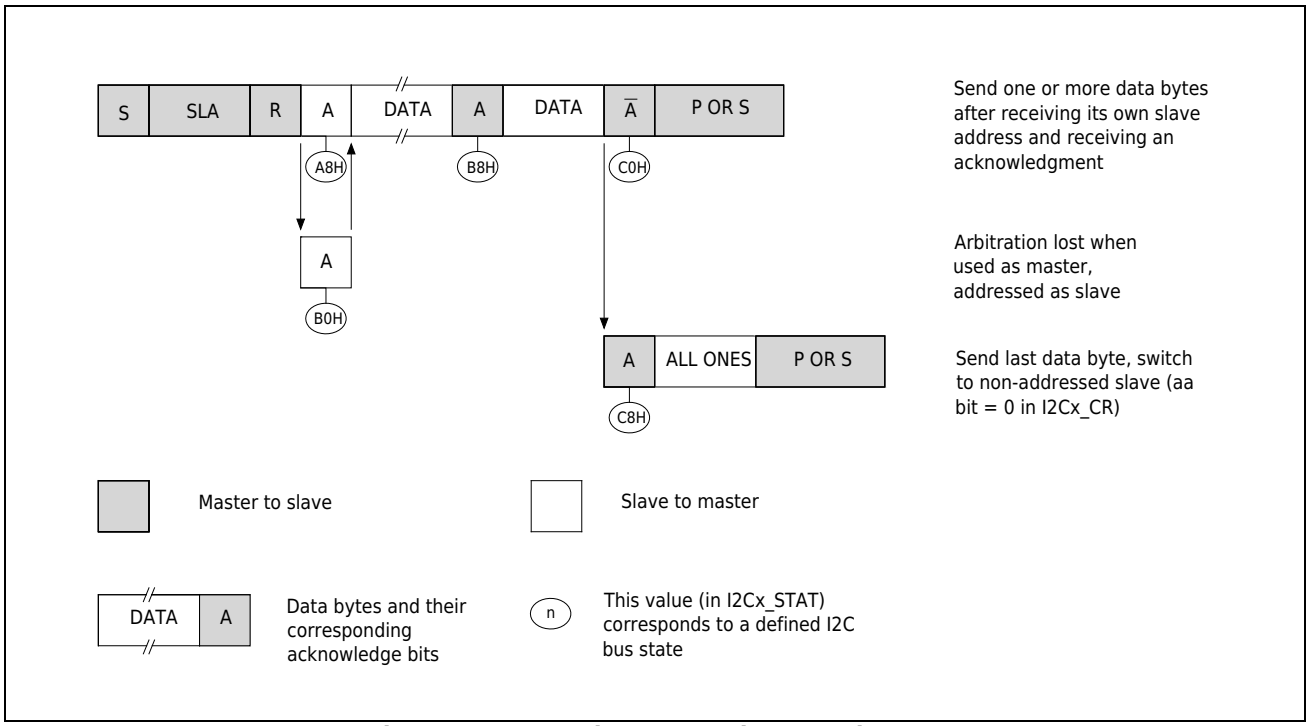


Figure 7-14 I2C Slave Transmit State Diagram

- General call mode

  General call mode is a special slave receiving mode, The addressing mode is 0x00, and the slave address and reading and writing are both 0. When both I2Cx_ADDR.GC and I2Cx_CR.aa are set to 1, the general call mode is enabled. In this mode, the I2Cx_STAT value is different from the normal slave receiver mode I2Cx_STAT value. Arbitration failure may also enter general call mode.

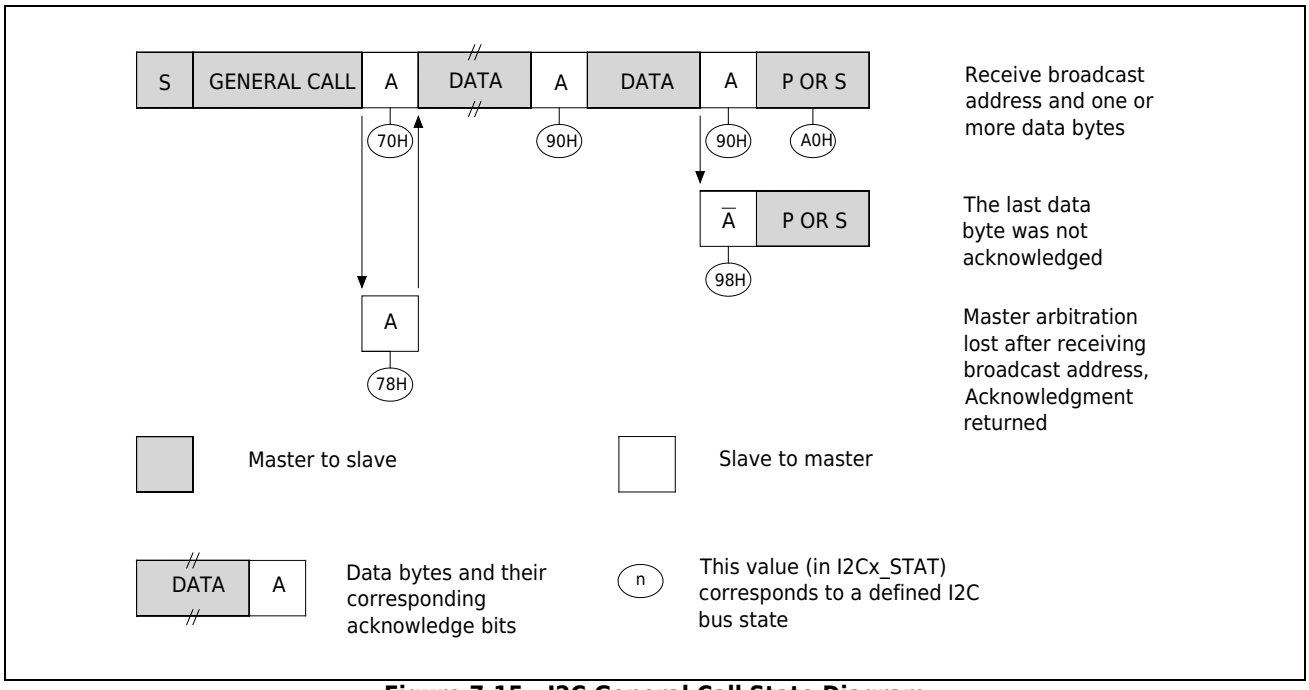


Figure 7-15 I2C General Call State Diagram

7.4.7 Status code expression

There are two special states in the I2C status register: F8H and 00H.

F8H: This status code indicates that no relevant information is available, because the serial interrupt flag "si" has not been set. This condition occurs between other states and before the I2C module has started to perform a serial transfer.

00H: This status code indicates that a bus error occurred during I2C serial transmission. A bus error occurs when a START or STOP condition occurs at an illegal position in a format frame. These illegal locations refer to address bytes, data bytes, or acknowledge bits during serial transfers. when external disturbances affect the internal I2C block signals. "si" is set when a bus error occurs.

Table 7-2 I2C status code expression

Status code	Description
Master sending mode	
08H	Start condition sent
10H	Repeated start condition sent
18H	SLA+W sent, ACK received
20H	SLA+W sent, not ACK received
28H	Data in I2Cx_DATA has been sent, ACK has been received
30H	Data in I2Cx_DATA sent, not ACK received
38H	Loss of Arbitration when SLA+ reads or writes data bytes
Main receiving mode	
08H	Start condition sent
10H	Repeated start condition sent
38H	Arbitration lost in non- ACK
40H	SLA+R sent, ACK received
48H	SLA+R sent, not ACK received
50H	Data byte has been received, ACK has been returned
58H	Data bytes received, non- ACK returned
Slave receive mode	
60H	Has received its own SLA+W and returned ACK
68H	When the master is in SLA+ read and write lost arbitration, has received its own SLA+W, and has returned ACK
80H	The previous addressing used its own slave address, received data bytes, and returned ACK
88H	The previous addressing used its own slave address, received data bytes, and returned non- ACK
A0H	When statically addressed, a STOP condition or a repeated START condition is received
Slave send mode	
A8H	Received its own SLA+R and returned ACK
B0H	When the Master loses arbitration, has received its own SLA+R, and has returned ACK
B8H	ACK received
C0H	Data bytes sent, not ACK received
C8H	Loaded data bytes have been sent, ACK has been received
general call mode	
70H	Received broadcast address (0x00), returned ACK
78H	When the master is in SLA+ read and write lost arbitration, the broadcast address has been received, and ACK has been returned
90H	The previous addressing used the broadcast address, data bytes have been received, and ACK has been returned

Status code	Description
98H	The previous addressing used the broadcast address, the data byte has been received, and a non- ACK has been returned
A0H	When statically addressed, a STOP condition or a repeated START condition is received
Other miscellaneous status	
F8H	No relevant status information available, si=0
00H	A bus error occurs during transmission, or external interference causes I2C to enter an undefined state

7.5 Programming example

7.5.1 Master sending example

Step1: Set PERI_CLKEN0.I2Cx to 1 to enable the I2Cx module clock.

Step2: Write 0 and 1 to PERI_RESET.I2Cx in sequence to reset the I2Cx module.

Step3: Configure the ports used by SCL and SDA of I2C as open-drain mode, and configure the appropriate port direction according to the I2C mode (for example, if the port direction is configured as output, the corresponding output register needs to be initialized before the corresponding bit of DIR is cleared. 1) and finally map SCL and SDA to corresponding pins.

Step4: Configure I2Cx_TM to make the clock rate of SCL meet the application requirements.

Step5: Set I2Cx_TMRUN to 1, enable SCL clock generator.

Step6: Set I2Cx_CR.ens to 1 to enable the I2C module.

Step7: Set I2Cx_CR.sta to 1, the bus tries to send the Start signal.

Step8: Wait for I2Cx_CR.si to become 1, the Start signal has been sent to the bus.

Step9: Query I2Cx_STAT, if the value of this register is 0x08 or 0x10, proceed to the next step, otherwise perform error handling.

Step10: Write SLA+W to I2Cx_DATA, set I2Cx_CR.sta to 0, set I2Cx_CR.si to 0, and send SLA+W.

Step11: Wait for I2Cx_CR.si to become 1, SLA+W has been sent to the bus.

Step12: Query I2Cx_STAT, if the register value is 0x18, proceed to the next step. Otherwise, perform error handling.

Step13: Write the data to be sent to I2Cx_DATA, set I2Cx_CR.si to 0, and send the data.

Step14: Wait for I2Cx_CR.si to become 1, the data has been sent to the bus.

Step15: Query I2Cx_STAT, if the register value is 0x28, proceed to the next step. Otherwise, perform error handling.

Step16: If the data to be sent is not completed, then jump to Step13 to continue execution.

Step17: Set I2Cx_CR.sto to 1, set I2Cx_CR.si to 0, and the bus tries to send a Stop signal.

7.5.2 Master receive example

Step1: Set PERI_CLKEN0.I2Cx to 1 to enable the I2Cx module clock.

Step2: Write 0 and 1 to PERI_RESET.I2Cx in sequence to reset the I2Cx module.

Step3: Configure the ports used by SCL and SDA of I2C as open-drain mode, and configure the appropriate port direction according to the I2C mode (for example, if the port direction is configured as output, the corresponding output register needs to be initialized before the corresponding bit of DIR is cleared. 1) and finally map SCL and SDA to corresponding pins.

Step4: Configure I2Cx_TM to make the clock rate of SCL meet the application requirements.

Step5: Set I2Cx_TMRUN to 1, enable SCL clock generator.

Step6: Set I2Cx_CR.ens to 1 to enable the I2C module.

Step7: Set I2Cx_CR.sta to 1, the bus tries to send the Start signal.

Step8: Wait for I2Cx_CR.si to become 1, the Start signal has been sent to the bus.

Step9: Query I2Cx_STAT, if the register value is 0x08 or 0x10, continue to the next step, otherwise, perform error handling.

Step10: Write SLA+R to I2Cx_DATA, set I2Cx_CR.sta to 0, set I2Cx_CR.si to 0, and send SLA+R.

Step11: Wait for I2Cx_CR.si to become 1, SLA+R has been sent to the bus.

Step12: Query I2Cx_STAT, if the register value is 0x40, proceed to the next step, otherwise, perform error handling.

Step13: Set I2Cx_CR.aa to 1 to enable the response flag.

Step14: Set I2Cx_CR.si to 0, the slave sends data, and the master sends ACK or NACK according to I2Cx_CR.aa.

Step15: Wait for I2Cx_CR.si to become 1, read the received data from I2Cx_DATA.

Step16: Query I2Cx_STAT, if the value of this register is 0x50 or 0x58, continue to the next step, otherwise perform error handling.

Step17: If the data to be received is only the last byte, set I2Cx_CR.aa to 0 and enable the non-response flag.

Step18: If the data to be received is not completed, then jump to Step14 to continue execution.

Step19: Set I2Cx_CR.sto to 1, set I2Cx_CR.si to 0, and the bus tries to send a Stop signal.

7.5.3 Slave receiving example

Step1: Set PERI_CLKEN0.I2Cx to 1 to enable the I2Cx module clock.

Step2: Write 0 and 1 to PERI_RESET.I2Cx in sequence to reset the I2Cx module.

Step3: Configure the ports used by SCL and SDA of I2C as open-drain mode, and configure the appropriate port direction according to the I2C mode (for example, if the port direction is configured as output, the corresponding output register needs to be initialized before the corresponding bit of DIR is cleared. 1) and finally map SCL and SDA to corresponding pins.

Step4: Set I2Cx_CR.ens to 1 to enable the I2C module.

Step5: Configure I2Cx_ADDR as the slave address.

Step6: Set I2Cx_CR.aa to 1 to enable the response flag.

Step7: Wait for I2Cx_CR.si to become 1 and be addressed by SLA+W.

Step8: Query I2C x_STAT, if the value of this register is 0x60, proceed to the next step, otherwise perform error handling.

Step9: Set I2Cx_CR.si to 0, the Master sends data, and the slave returns ACK or NACK according to I2Cx_CR.aa.

Step10: Wait for I2Cx_CR.si to become 1, read the received data from I2Cx_DATA.

Step11: Query I2Cx_STAT, if the value of this register is 0x80, continue to the next step, otherwise perform error handling

Step12: If the data to be received is not completed, then jump to Step9 to continue execution.

Step13: Set I2Cx_CR.aa to 0, and set I2Cx_CR.si to 0.

7.5.4 Slave sending example

Step1: Set PERI_CLKEN0.I2Cx to 1 to enable the I2Cx module clock.

Step2: Write 0 and 1 to PERI_RESET.I2Cx in sequence to reset the I2Cx module.

Step3: Configure the ports used by SCL and SDA of I2C as open-drain mode, and configure the appropriate port direction according to the I2C mode (for example, if the port direction is configured as output, the corresponding output register needs to be initialized before the corresponding bit of DIR is cleared. 1) and finally map SCL and SDA to corresponding pins.

Step4: Set I2Cx_CR.ens to 1 to enable the I2C module.

Step5: Configure I2Cx_ADDR as the slave address.

Step6: Set I2Cx_CR.aa to 1 to enable the response flag.

Step7: Wait for I2Cx_CR.si to become 1 and be addressed by SLA+R.

Step8: Query I2Cx_STAT, if the value of this register is 0xA8, proceed to the next step, otherwise, perform error handling.

Step9: Write the data to be sent to I2Cx_DATA, set I2Cx_CR.si to 0, and send the data.

Step10: Wait for I2Cx_CR.si to become 1, the data has been sent to the bus.

Step11: Query I2Cx_STAT, if the value of this register is 0xB8 or 0xC0, proceed to the next step, otherwise, perform error handling.

Step12: If the data to be sent is not completed, then jump to Step9 to continue execution.

Step13: Set I2Cx_CR.aa to 0, and set I2Cx_CR.si to 0.

7.6 Register description

Register list

I2C0 base address: 0x40000400

I2C1 base address: 0x40004400

Table 7-3 Register List

Offset	Register name	Access	Register description
0x00	I2Cx_TMRUN	RW	I2C baud rate counter enable register.
0x04	I2Cx_TM	RW	I2C baud rate counter configuration register.
0x08	I2Cx_CR	RW	I2C configuration register.
0x0c	I2Cx_DATA	RW	I2C data register.
0x10	I2Cx_ADDR	RW	I2C address register.
0x14	I2Cx_STAT	RO	I2C state register.

7.6.1 I2C Baud Rate Counter Enable Register (I2Cx_TMRUN)

Address offset: 0x00

Reset value: 0x0000 0000

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

Bit	Marking	Functional description
31：1	Reserved	
0	tme	Baud rate counter enable. 0 – disable 1 – enable

7.6.2 I2C Baud Rate Counter Configuration Register (I2Cx_TM)

Address offset: 0x04

Reset value: 0x0000 0000

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

Bit	Marking	Functional description
31:8	Reserved	
7:0	tm	tm: Baud rate counter configuration value. Fscl = Fpclk / 8 / (tm+1) ，其中 tm >0

7.6.3 I2C Configuration Register (I2Cx_CR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ens	sta	sto	si	aa	Res	h1m
									RW	RW	RW	RW	RW		RW

Bit	Marking	Functional description
31:7	Reserved	
6	ens	I2C module enable control 0 – disabled 1 – enable
5	sta	I2C bus control 0 – no function 1 – send START to the bus
4	sto	I2C bus control 0 – no function 1 – Send STOP to the bus
3	si	I2C interrupt flag reads 1, an I2C interrupt has occurred Write 0, I2C performs next operation
2	aa	Acknowledgment control bit 0 – send NAK 1 – send ACK
1	Reserved	
0	h1m	I2C filter parameter configuration 0 – advanced filtering, higher anti-interference performance 1 – Simple filtering, faster communication rate Note: See the [Input Filter] chapter for details.

7.6.4 I2C Data Register (I2Cx_DATA)

Address offset: 0x0c

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								i2cdat							
								RW							

Bit	Marking	Functional description
31:8	Reserved	
7:0	i2cdat	I2C data register In I2C send mode, write the data to be sent In I2C receive mode, read received data

7.6.5 I2C Address Register (I2Cx_ADDR)

Address offset: 0x10

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								i2cadr							GC
								RW							RW

Bit	Marking	Functional description
31:8	Reserved	
7:1	i2cadr	I2C slave mode address.
0	GC	Broadcast Address Acknowledgment Enable 0 – disabled 1 – enable

7.6.6 I2C Status Register (I2Cx_STAT)

Address offset: 0x14

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								i2csta							
								RO							

Bit	Marking	Functional description
31:8	Reserved	
7:0	i2csta	I2C state register. For the specific definition of the status value, see the chapter [State Code Expression]

8 Serial Peripheral Interface (SPI)

8.1 Introduction to SPI

The SPI interface is a synchronous serial data communication interface working in full-duplex mode, using 4 pins for communication: MISO, MOSI, SCK, CS/SSN. When SPI is used as the master, output CS and SCK signals to control the communication process. When SPI acts as a slave, it communicates under the control of SSN and SCK signals.

8.2 Main Features of SPI

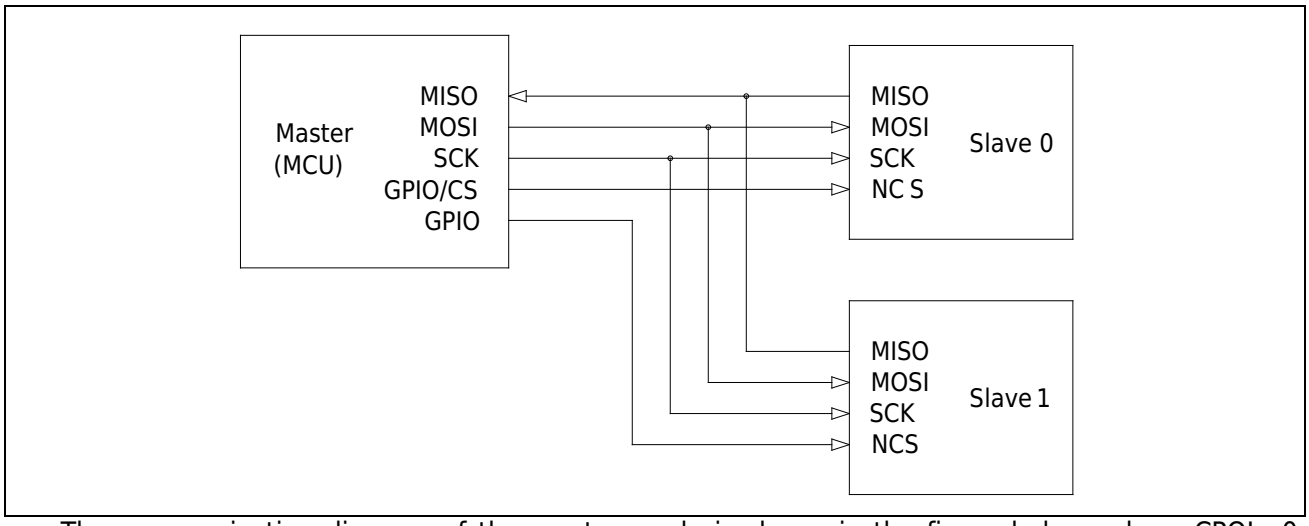
- Support SPI master mode, SPI slave mode
- Support standard four-wire full-duplex communication
- Supports configuration of serial clock polarity and phase
- Master mode supports 7 communication speeds
- The maximum frequency division factor of the host mode is PCLK/2
- The maximum frequency division factor of slave mode is PCLK/4
- The frame length is fixed at 8 bits, and the MSB is transmitted first
- Support DMA software/hardware access

8.3 SPI Functional Description

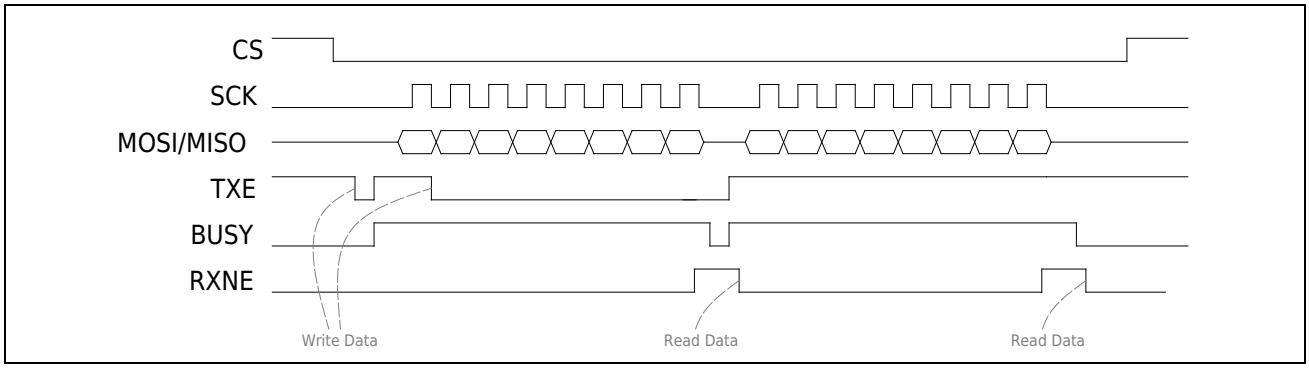
8.3.1 SPI master mode

The length of each data frame is fixed at 8 bits, and the first bit of sent data is fixed at MSB. Set SPIx_CR.mstr to 1, the SPI interface works in master mode. Write data into the SPIx_Data register to start SPI transmission, and the SCK pin will automatically generate a serial clock; on the edge of the serial clock, the data in the shift register is sent to the MOSI pin, and the data on the MISO pin is received into the shift register. SCK pin output clock is controlled by SPIx_CR[spr2:spr0], and the output frequency range is PCLK /2~PCLK/128; the output level of the CS pin is controlled by SPIx_SSN.ssn, and the output of the GPIO pin The level is controlled by GPIO related registers.

A typical application block diagram of master mode is shown below.

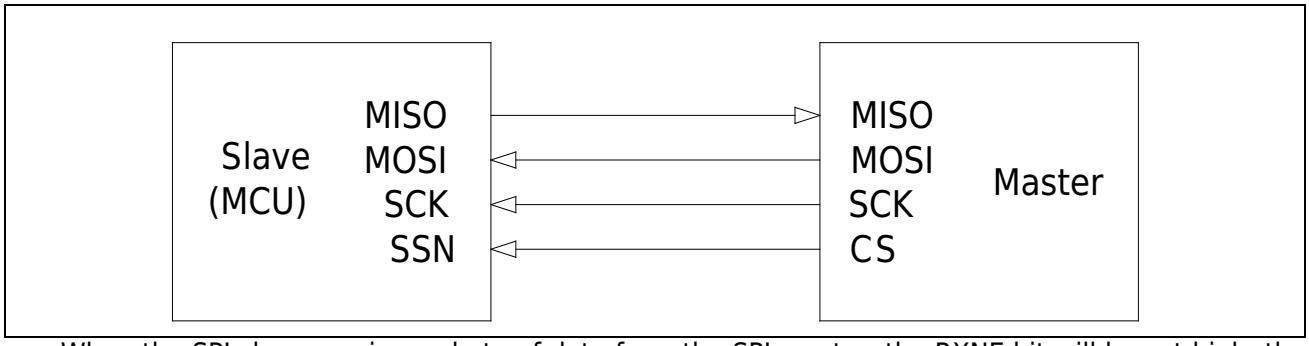


The communication diagram of the master mode is shown in the figure below, where CPOL=0, CPHA=0.



8.3.2 SPI slave mode

The length of each data frame is fixed at 8 bits, and the first bit of received data is fixed at MSB. Set SPIx_CR.mstr to 0, the SPI interface works in slave mode. In this mode, the SCK pin is used as an input pin and the serial clock comes from an external Master; the SSN pin is used as an input pin and the chip select signal comes from an external Master or is fixed at low level. GPIO chapter for details on the SSN pins. A typical application block diagram of the slave mode is shown below.



When the SPI slave receives a byte of data from the SPI master, the RXNE bit will be set high; the user program should read the received data as soon as possible. The communication timing of receiving data in slave mode is as follows, where CPOL=0, CPHA=0.

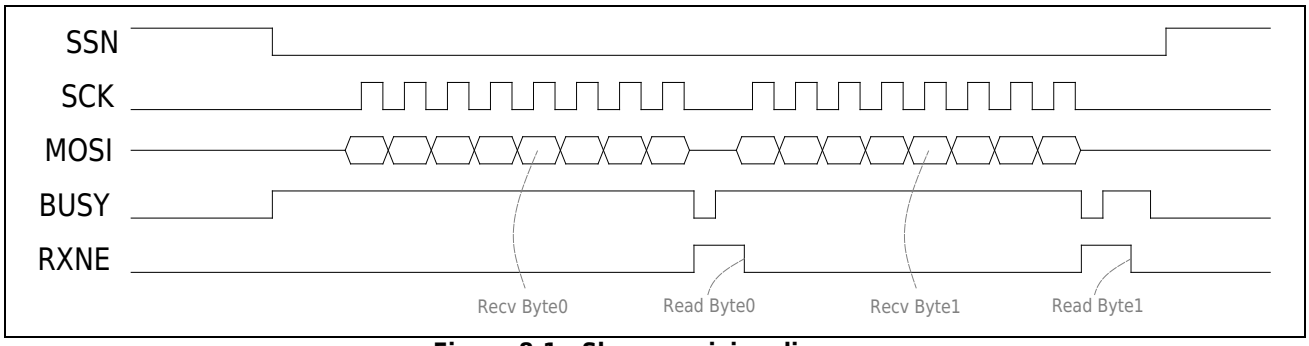


Figure 8-1 Slave receiving diagram

When the SPI slave needs to send data to the Master, the following five steps should be performed in sequence before the Master pulls down NSS: set the PERI_RESET0.SPIx bit to 0, set the PERI_RESET0.SPIx bit to 1, configure the SPI communication parameters, and write to the SPIx_DATA register Write the first byte of data to be sent; after NSS is pulled low, whenever the TXE flag is 1, the subsequent data to be sent should be written to the SPIx_DATA register as soon as possible. The communication timing of sending data in slave mode is as follows, where CPOL=0, CPHA=0.

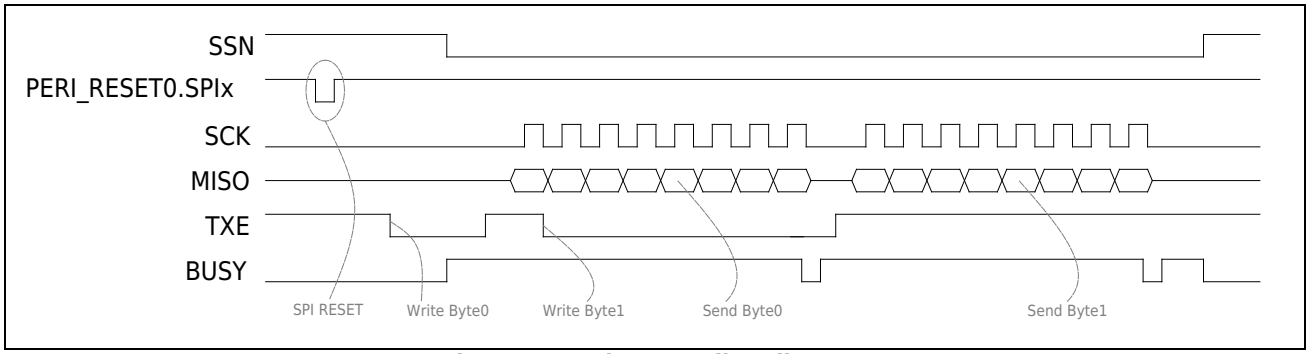


Figure 8-2 Slave sending diagram

8.3.3 SPI data frame format

The SPI interface frame format depends on the configuration of clock polarity bit CPOL and clock phase bit CPHA.

When CPOL is 0, the idle state of SCK line is low. When CPOL is 1, the idle state of SCK line is high level. When CPHA is 0, the data will be sampled when the first SCK clock transition signal jumps. When CPHA is 1, the data will be sampled when the second SCK clock signal jumps.

The frame format of the SPI interface master is shown in the figure below.

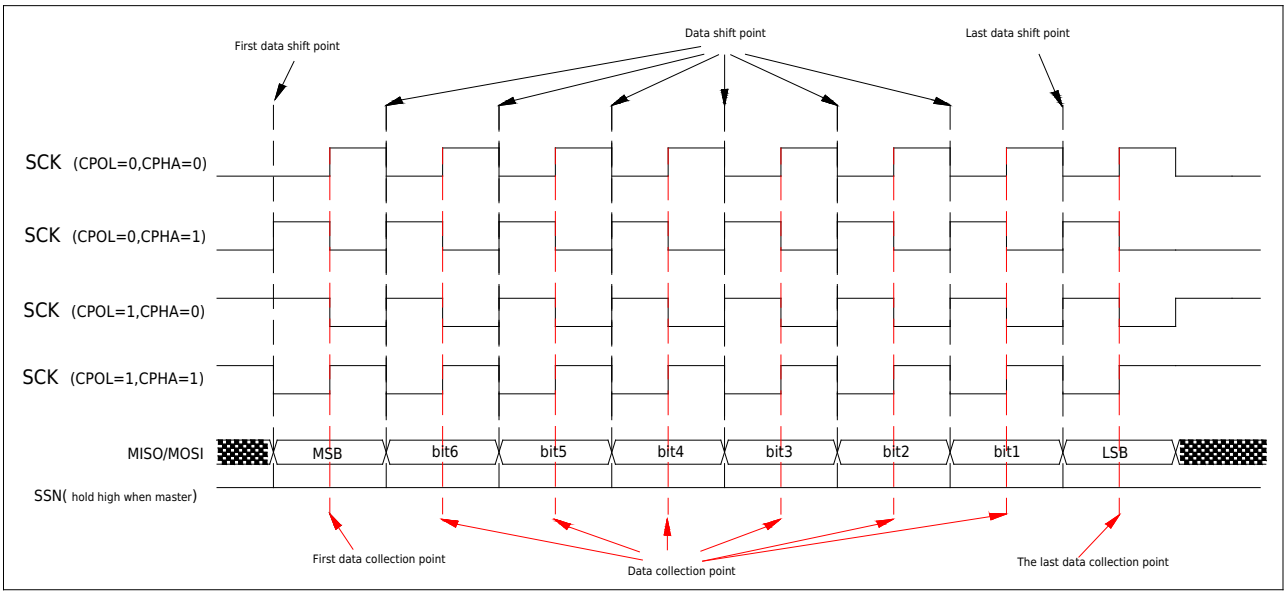


Figure 8-3 Host Mode Frame Format

The SPI interface slave frame format is shown in the figure below.

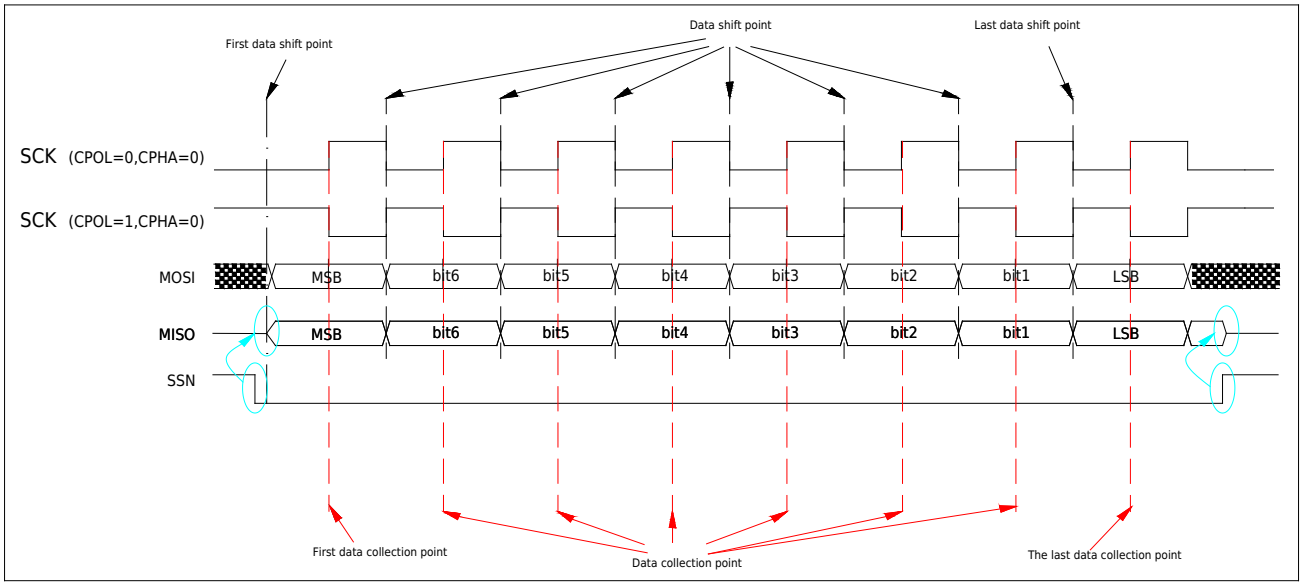


Figure 8-4 Data frame format when slave CPHA is 0

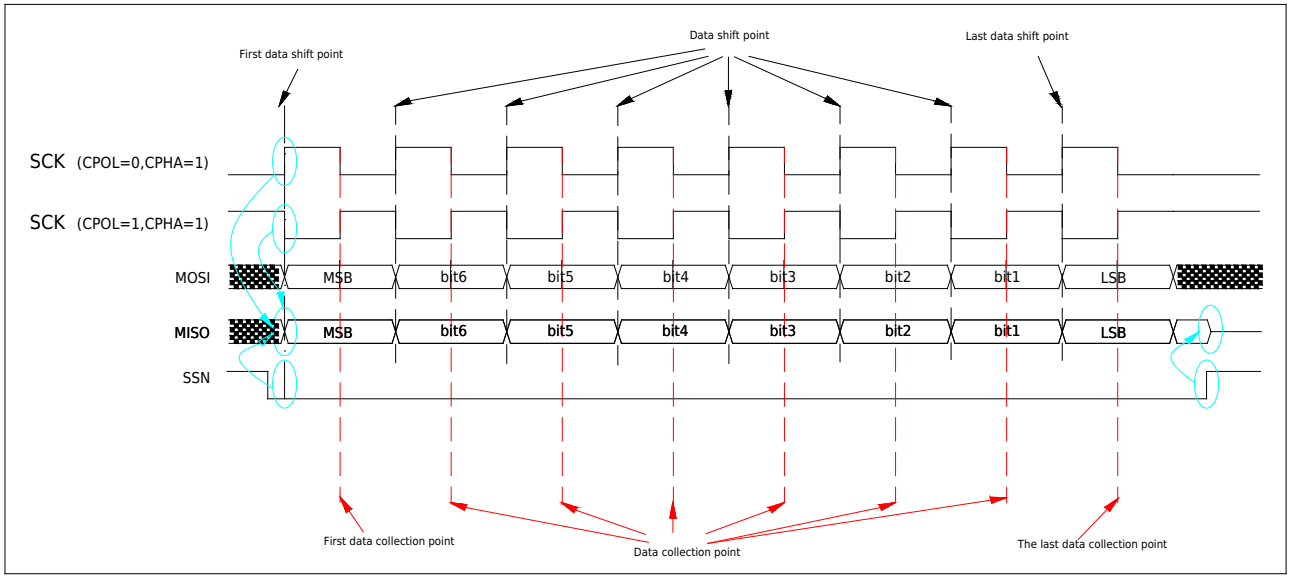


Figure 8-5 Data frame format when slave CHPA is 1

8.3.4 SPI Status Flags and Interrupts

SPI will generate the following four status flags during work, and the generation conditions and clearing methods are as follows.

- When the data in the send buffer (SPIx_Data) is transferred to the send shift register, SPIx_STAT.txe will be set by hardware, indicating that the send buffer is empty and the next data can be written. To SPIx_DATA clears this flag.
- When the data in the receiving shift register is transferred to the receiving buffer (SPIx_Data), SPIx_STAT.rxne will be set by hardware, indicating that the receiving buffer is not empty, and the user needs to read the data as soon as possible. This flag can be cleared by reading data from SPIx_DATA.
- When the SPI works in master mode and the external SSN input is low, SPIx_STAT.mdf will be set by hardware, indicating that other SPI masters are occupying the bus. When the SSN input is high, SPIx_STAT.mdf will be automatically cleared by hardware.
- When the SPI works in slave mode, if the SSN pin is pulled high during data transmission, SPIx_STAT.sserr will be set. Set SPIx_CR.spen to 0 to clear this flag.

If the SPI interrupt is enabled (SPIx_CR2.int_en=1), the following three situations can generate an interrupt:

- The SPI transmit buffer is empty, that is, SPIx_STAT.txe is 1
- SPI receive buffer is not empty, that is, SPIx_STAT.rxne is 1
- SPI is wrong, that is, SPIx_STAT.mdf is 1

In the interrupt service routine, it is necessary to write 0x00 to SPIx_ICLR to clear the internal interrupt flag.

8.3.5 SPI multi-machine system configuration instructions

- When the SPI module works as a master and works in a single-master system, the slave can be controlled through the SPI_CS pin or GPIO pin. SPI_CS pin is selected as the chip select signal of the slave, set SPIx_SSN.ssn to 0 to select the slave, and set SPIx_SSN.ssn to 1 to release the slave. GPIO pin is selected as the chip select signal of the slave, set the corresponding bit of the GPIOx_OUT register to 0 to select the slave, and set the corresponding bit of the GPIOx_OUT register to 1 to release the slave.
- When the SPI module is a slave, configure the source of SPI_SSN as needed (see GPIO port auxiliary controller for details). When the SSN is low, the slave machine can be selected for communication; when the SSN is high, the machine is in an unselected state.
- When the SPI mode works on multi-master and multi-slave, all slave chip select signals are connected through GPIO pins, and the master must also be connected to the SSN signals of other masters through GPIO pins to monitor whether the bus is occupied. Master0 needs to communicate as shown in the figure below is: wait for Master0.SSN to become high; output low

from GPIO0 to notify Master1 to release the SPI bus; output low from GPIO2 to select Slave1; communicate with Slave1; output high from GPIO2 to release Slave1; output high from GPIO0 to release Master1.

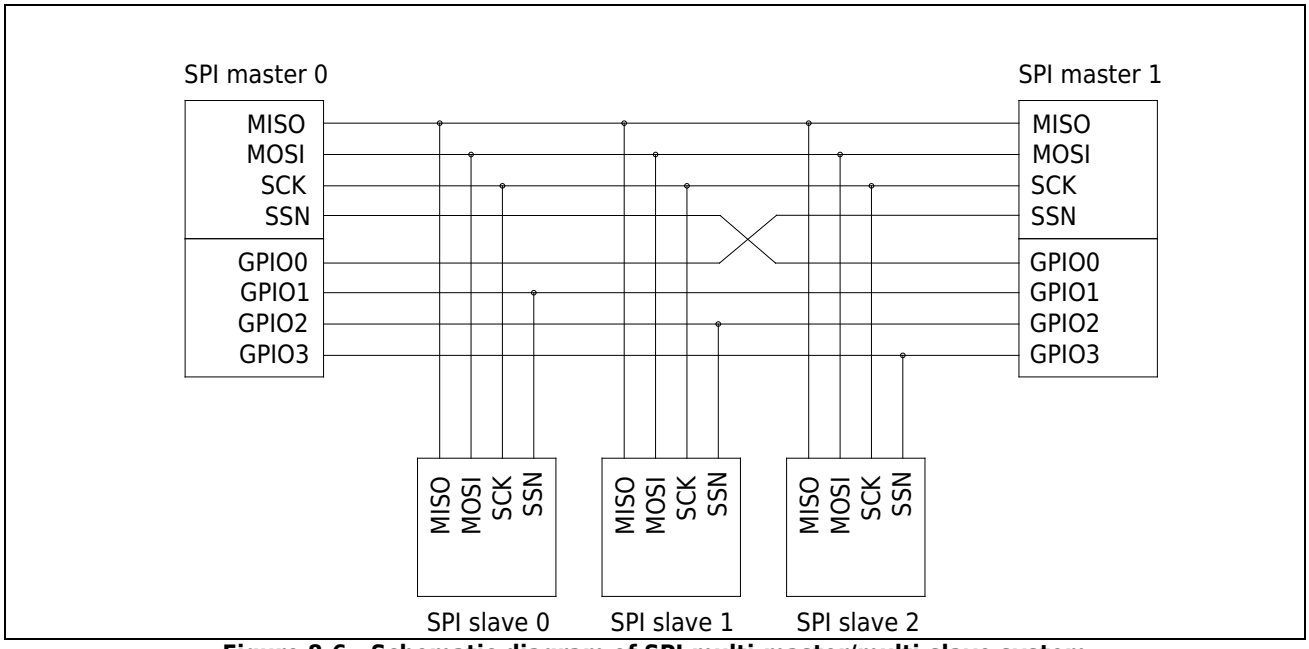


Figure 8-6 Schematic diagram of SPI multi-master/multi-slave system

8.3.6 SPI pin configuration description

SPI can maintain some or all functions under some special pin configurations.

The specific situation is as follows ("✓" means that the pin is configured and used, and blank means that the pin is not configured):

Table 8-1 SPI pin configuration description table

	SPI_CS(Master) / SPI_SSN (slave)	SCK	MOSI	MISO	Functional description
Host mode	✓	✓	✓	✓	general configuration All Masters function normally
		✓	✓	✓	All Masters function normally
	✓	✓	✓		Master sending function is normal
	✓	✓		✓	Master receiving function is normal
		✓	✓		Master sending function is normal
		✓		✓	Master receiving function is normal
Slave mode	✓	✓	✓	✓	general configuration All slaves function normally
	✓	✓	✓		Slave receiving function is normal
	✓	✓		✓	Slave sending function is normal
	✓ fixed low level	✓	✓	✓	All slaves function normally
	✓ fixed low level	✓	✓		Slave receiving function is normal
	✓ fixed low level	✓		✓	Slave sending function is normal

Note:

- Conditions not listed in the table are currently not supported.
- In master mode, even if the SPIx_CS chip select output is not used, SPIx.SSN needs to be set to 1 before sending data, and SPIx.SSN needs to be set to 0 after sending data.
- In slave mode and chip select input is fixed at low level, in order to maintain normal function, SPIx_CR.cpha=1 must be satisfied.

8.4 SPI programming example

8.4.1 SPI master sending example

Step1: Map CS/SCK/MISO/MOSI to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter; configure the CS/SCK/MOSI pins as output mode, and configure the MISO pin as input mode.

Step2: Set SPIx_CR.mstr to 1 to make SPI work in master mode.

Step3: Configure SPIx_CR[spr2:spr0] to make the clock rate of SCK output meet the application requirements.

Step4: Configure SPIx_CR.cpol and SPIx_CR.cpha to make the data frame format meet the application requirements.

Step5: Set SPIx_CR.spen to 1 to enable the SPI interface.

Step6: Set SPIx_SSN.ssn to 0, make the CS pin output low level to select the slave.

Step7: When SPIx_STAT.txe is 1, write the data to be sent to SPIx_DATA as soon as possible.

Step8: If the data to be sent is not completed, then jump to Step7 to continue execution.

Step9: The query waits for SPIx_STAT.txe to become 1, and the last byte data has been sent.

Step10: Query and wait for SPIx_STAT.busy to become 0, the SPI bus is idle.

Step11: Set SPIx_SSN.ssn to 1, make the CS pin output high level to release the slave.

Note:

- GPIO can be used instead of CS to realize chip select output, which is mostly used in multi-machine communication system.
- SPIx_SSN.ssn must be set to 0 during the transmission process, and SPIx_SSN.ssn must be set to 1 after the transmission is completed.

8.4.2 SPI master receive example

Step1: Map CS/SCK/MISO/MOSI to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter; configure the CS/SCK/MOSI pins as output mode, and configure the MISO pin as input mode.

Step2: Set SPIx_CR.mstr to 1 to make SPI work in master mode.

Step3: Configure SPIx_CR[spr2:spr0] to make the clock rate of SCK output meet the application requirements.

Step4: Configure SPIx_CR.cpol and SPIx_CR.cpha to make the data frame format meet the application requirements.

Step5: Set SPIx_CR.spen to 1 to enable the SPI interface.

Step6: Set SPIx_SSN.ssn to 0, make the CS pin output low level to select the slave.

Step7: Write arbitrary data to SPIx_DATA to trigger the Master to send SCK.

Step8: Query and wait for SP Ix_STAT.rxne to become 1, and the data sent by the slave has been received.

Step9: Read the received data from SPIx_DATA.

Step10: If the data to be received is not completed, then jump to Step7 to continue execution.

Step11: Set SPIx_SSN.ssn to 1, make the CS pin output high level to release the slave.

Note:

- GPIO can be used instead of CS to realize chip select output, which is mostly used in multi-machine communication system.
- SPIx_SSN.ssn must be set to 0 during the transmission process, and SPIx_SSN.ssn must be set to 1 after the transmission is completed.

8.4.3 SPI slave sending example

Step1: Map SSN/SCK/MISO/MOSI to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter; and configure the SSN/SCK/MOSI pins as input mode, and configure the MISO pin as output mode. SSN pins, see GPIO Port Auxiliary Controller.

Step2: Set PERI_RESET0.SPIx to 0, so that the SPI module is in reset state.

Step3: Set PERI_RESET0.SPIx to 1 to make the SPI module work.

Step4: Set SPIx_CR.mstr to 0 to make SPI work in slave mode.

Step5: Configure SPIx_CR.cpol and SPIx_CR.cpha to make the data frame format meet the application requirements.

Step6: Set SPIx_CR.spen to 1 to enable the SPI interface.

Step7: Write the first byte data to be sent into SPIx_DATA.

Step8: The query waits for the SSN pin to be pulled low, and the master selects the SPI slave.

Step9: When SPIx_STAT.txe is 1, write the data to be sent to SPIx_DATA as soon as possible.

Step10: When it is found that the SSN pin is low and the data to be sent has not been completed, jump to Step9.

Step11: The query waits for the SSN pin to be pulled high, and the master releases the SPI slave.

Note:

– Whenever the SPI slave needs to send data, it needs to start the initialization operation and sending operation from Step2.

8.4.4 SPI Slave Receive Example

Step1: Map SSN/SCK/MISO/MOSI to the required pins according to the relevant description of the pin digital multiplexing function in the GPIO chapter; and configure the SSN/SCK/MOSI pins as input mode, and configure the MISO pin as output mode. SSN pins, see GPIO Port Auxiliary Controller.

Step2: Set SPIx_CR.mstr to 0 to make SPI work in slave mode.

Step3: Configure SPIx_CR.cpol and SPIx_CR.cpha to make the data frame format meet the application requirements.

Step4: Set SPIx_CR.spen to 1 to enable the SPI interface.

Step5: The query waits for the SSN pin to be pulled low, and the master selects the SPI slave.

Step6: Query and wait for SPIx_STAT.rxne to become 1, and the data sent by the Master has been received.

Step7: Read the received data from SPIx_DATA.

Step8: If the data to be received is not completed, then jump to Step6 to continue execution.

Step9: The query waits for the SSN pin to be pulled high, and the master releases the SPI slave.

8.5 SPI register description

Register list

SPI0 base address: 0x40000800

SPI1 base address: 0x40004800

Table 8-2 SPI register list

Offset	Register name	Access	Register description
0x00	SPIx_CR	RW	SPIx Configuration Register
0x04	SPIx_SSN	RW	SPIx Chip Select Configuration Register
0x08	SPIx_STAT	RO	SPIx Status Register
0x0c	SPIx_DATA	RW	SPIx Data Register
0x10	SPIx_CR2	RW	SPIx Configuration Register 2
0x14	SPIx_ICLR	WO	SPIx Interrupt Clear Register

8.5.1 SPI Configuration Register (SPIx_CR)

Address offset: 0x00

Reset value: 0x0000 0014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								spr2	spen	Res	mstr	cpol	cpha	spr1	spr0
								RW	RW		RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:8	Reserved	
7	spr2	Baud rate selection bit 2 Refer to spr0.
6	spen	SPI module enable control 0 – disabled 1 – enable
5	Reserved	
4	mstr	SPI working mode configuration 0 – Slave mode 1 – Host mode
3	cpol	SCK line idle state configuration 0 – Low level 1 – High level
2	cpha	Clock Phase Configuration 0 – first edge 1 – second edge
1	spr1	Baud rate selection bit 1 Reference spr0
0	spr0	Baud rate selection bit 0

Table 8-3 Master Mode Baud Rate Selection

spr2	spr1	spr0	SCK Rate
0	0	0	PCLK /2
0	0	1	PCLK /4
0	1	0	PCLK /8
0	1	1	PCLK /16
1	0	0	PCLK /32
1	0	1	PCLK /64
1	1	0	PCLK /128
1	1	1	Reserved

8.5.2 SPI Chip Select Configuration Register (SPIx_SSN)

Address offset: 0x04

Reset value: 0x000000FF

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

Bit	Marking	Functional description
31:1	Reserved	
0	ssn	SPI_CS output level configuration in master mode 0: SPI_CS port output low level 1: SPI_CS port outputs high level

8.5.3 SPI Status Register (SPIx_STAT)

Address offset: 0x08

Reset value: 0x00000004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								spif	Res.	sserr	mdf	busy	txe	rxne	Res.
								RO		RO	RO	RO	RO	RO	

Bit	Marking	Functional description
31:8	Reserved	
7	spif	Receive completion flag 1: A byte has been received from the SPI bus 0: receiving
6	Reserved	
5	sserr	SSN error flag in slave mode
4	mdf	In Master mode, the conflict flag 1: SSN pin level is low 0: SSN pin level is high
3	busy	SPI bus transfer status flag 1: SPI bus is transferring data 0: SPI bus is idle
2	txe	Transmit buffer status flag 1: Transmit buffer is empty 0: Transmit buffer is not empty
1	rxne	Receive Buffer Status Flags 1: Receive buffer is not empty 0: receive buffer is empty
0	Reserved	

8.5.4 SPI Data Register (SPIx_DATA)

Address offset: 0x0c

Reset value: 0x00000000

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

Bit	Marking	Functional description
31:8	Reserved	
7:0	spdat	Data register In send mode, write the byte to be sent to this register; In receive mode, the received byte is read from this register;

8.5.5 SPI Configuration Register 2 (SPIx_CR2)

Address offset: 0x10

Reset value: 0x0000 0003

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									rxneie	txeie	hdma_tx	hdma_rx	int_en	Reserved	
									RW	RW	RW	RW	RW		

Bit	Marking	Functional description
31:7	Reserved	
6	rxneie	Receive buffer not empty interrupt enable 0 – disabled 1 – enable
5	txeie	Transmit buffer empty interrupt enable 0 – disabled 1 – enable
4	hdma_tx	DMA hardware access transmit enable. 0 – disabled 1 – enable
3	hdma_rx	DMA hardware access receive enable. 0 – disabled 1 – enable
2	int_en	SPI interrupt enable. 0 – disabled 1 – enable
1:0	Reserved	Please keep the value of these two bits as 1.

8.5.6 SPI Interrupt Clear Register 2 (SPIx_ICLR)

Address offset: 0x14

Reset value: 0x0000 00FF

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															int_clr
															R1W0

Bit	Marking	Functional description
31:1	Reserved	
0	int_clr	SPI interrupt clear 0 – Clear 1 – hold

9 Clock Trim Module (CLKTRIM)

9.1 Introduction to CLKTRIM

The CLKTRIM (Clock Trimming) module is a circuit specially used to calibrate / monitor the clock. In the calibration mode, select an accurate clock source to calibrate the inaccurate clock source, repeat the calibration, and adjust the parameters of the inaccurate clock source until the frequency of the calibrated clock source meets the accuracy requirements. In the calibration mode, the count value will have a certain error, but it is within the allowable precision error range. In the monitoring mode, select a stable clock source to monitor the system's working clock. Under the set monitoring period, monitor whether the system's working clock is invalid and generate an interrupt. In the calibration mode and monitoring mode, the required clock sources must be initialized and enabled. For the specific configuration process, please refer to the [System Controller (SYSCTRL)] chapter.

9.2 CLKTRIM main features

CLKTRIM supports the following features:

- Calibration mode
- Monitoring mode
- 32-bit reference clock counter can be loaded with initial value
- 32-bit clock counter to be calibrated with configurable overflow value
- 6 reference clock sources
- 6 clock sources to be calibrated
- Support interrupt mode

9.3 CLKTRIM function description

9.3.1 CLKTRIM calibration mode

9.3.1.1 Principle of Clock Calibration

When calibrating a clock whose frequency is inaccurate but has parameter settings to adjust the frequency (the clock to be calibrated CAL_CLK), an accurate clock is required as the reference clock (REF_CLK). Set the calibration time Ttrim, the frequency of the clock to be calibrated Fcal, and the frequency of the reference clock Fref, use the clock to be calibrated and the reference clock to count simultaneously, and stop counting at the end of the calibration time. M of the clock counter to be calibrated, the value N of the reference clock counter,

Get the equation Ttrim = M/Fcal = N/Fref

Deduce Fcal = Fref * M / N

Judging from the formula, the smaller the frequency error rate of the reference clock is, the smaller the error rate of the clock to be calibrated is.

After calculating the frequency of the clock to be calibrated, if the error rate exceeds the range required by the system, you can adjust the parameters of the clock to be calibrated, and then repeat the above steps to calibrate until the error rate of the clock frequency to be calibrated meets the system requirements.

9.3.1.2 Clock Calibration Module Hardware Structure

There are 6 clock sources for the clock to be calibrated in the time calibration module, and 6 clock sources for the reference clock, as shown in Figure 9-1:

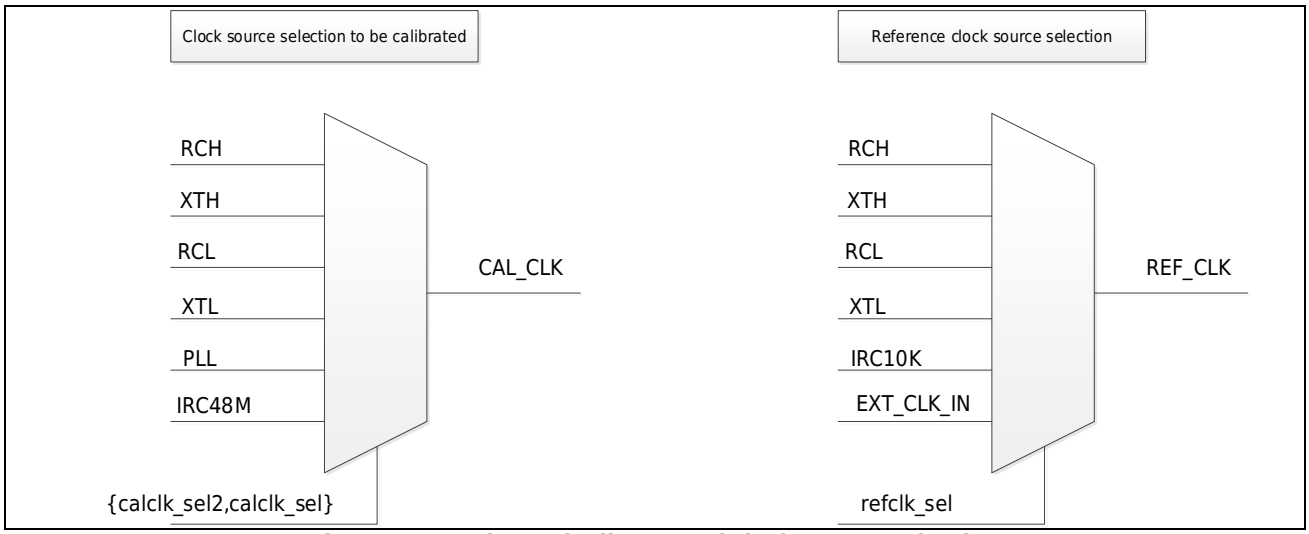


Figure 9-1 Schematic diagram of clock source selection

The selection of the clock to be calibrated is configured by registers CLKTRIM_CR.calclk_sel2 and CLKTRIM_CR.calclk_sel.

The selection of the reference clock is configured by the register CLKTRIM_CR.refclk_sel.

As shown in Figure 9-2, the clock calibration module has two 32-bit counters, one is clocked by the reference clock, and the initial value can be configured as a down counter. The initial value is configured by the register CLKTRIM_REFCON.rcntval, and the counter value can be read from the register CLKTRIM_REFCNT When the counter counts down to 0, it stops counting and generates an interrupt. One is an up-counter that uses the clock to be calibrated as the clock and can configure the overflow value. The overflow value is configured by the register CLKTRIM_CALCON. ccnt val, and the counter value can be read from the register CLKTRIM_CALCNT. Generate an interrupt.

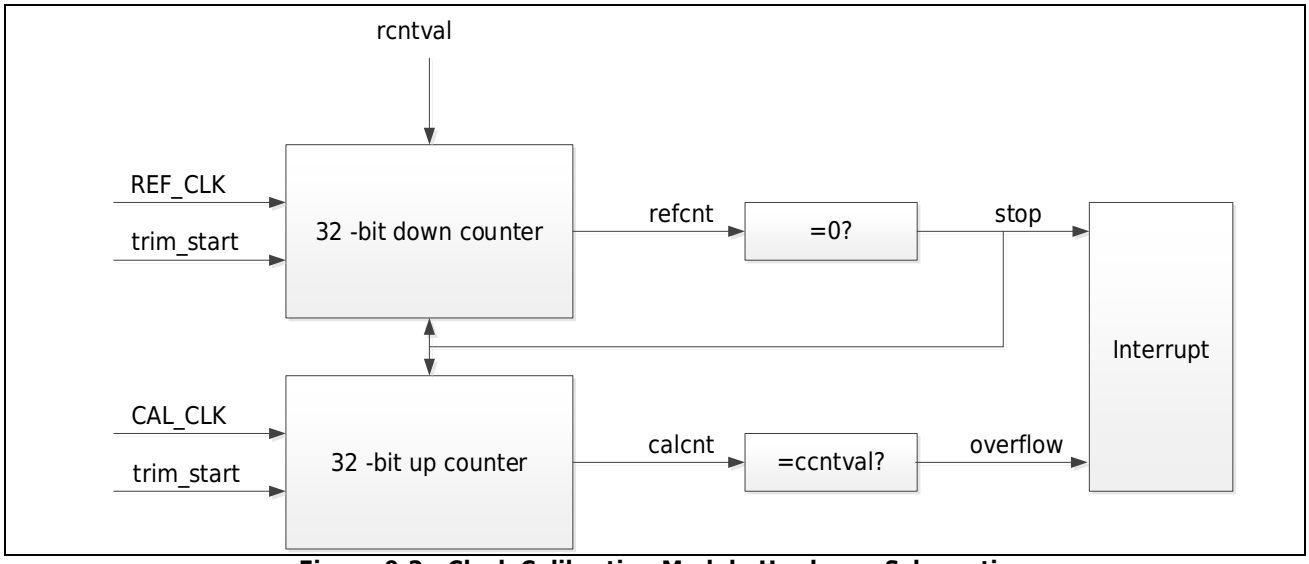


Figure 9-2 Clock Calibration Module Hardware Schematic

9.3.1.3 Clock Calibration Software Flow

1. Set the CLKTRIM_CR.refclk_sel register to select the reference clock.
2. Set the CLKTRIM_CR.calclk_sel register to select the clock to be calibrated.
3. Set the CLKTRIM_REFCON.rcntval register to the calibration time.
4. Set CLKTRIM_CR.IE register to enable interrupt.
5. Set the CLKTRIM_CR.trim_start register to start the trim.
6. The reference clock counter and the clock to be calibrated counter start counting.
7. When the reference clock counter counts down from the initial value to 0, CLKTRIM_IFR.stop is set to 1 and an interrupt is triggered.
8. The interrupt service subroutine judges that CLKTRIM_IFR.stop is 1, reads the values of registers CLKTRIM_REFCNT and CLKTRIM_CALCNT,
9. Clear the CLKTRIM_CR.trim_start register to end the trim.
10. Calculate the frequency of the clock to be calibrated according to the formula mentioned in the clock calibration principle chapter,
    where M = value of register CLKTRIM_CALCNT,
    N = value of register CLKTRIM_REFCON.rcntval
    When the frequency of the clock to be calibrated does not meet the error rate requirements, adjust the parameters of the clock to be calibrated,
    Repeat steps 5 to 10 until the clock to be calibrated meets the error rate requirement.

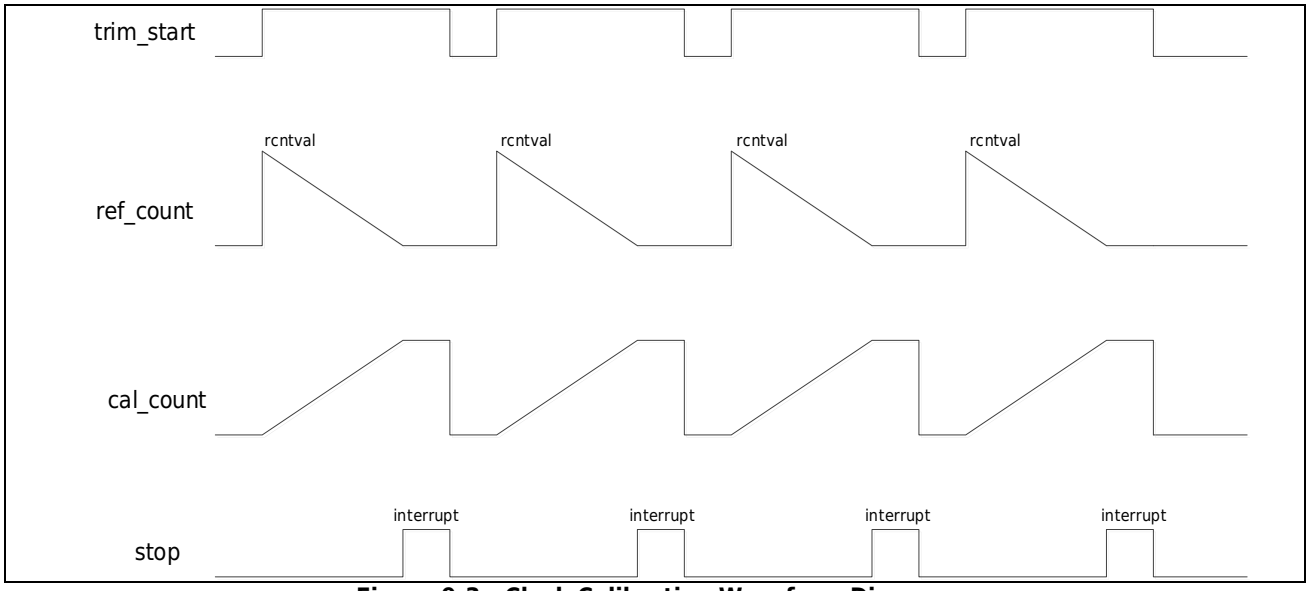


Figure 9-3 Clock Calibration Waveform Diagram

Note:

- In the calibration mode, it is possible that the calibration time is set too long, and the clock counter to be calibrated overflows before CLKTRIM_IFR.stop is set to 1, and CLKTRIM_IFR.calcnt_of is set to 1, triggering an interrupt. When the interrupt service subroutine finds that CLKTRIM_IFR.calcnt_of is set to 1, clear the CLKTRIM_CR.trim_start register to end the calibration.

In this case, the calibration cannot be performed correctly, and the calibration time must be adjusted and re-calibrated.

The specific steps are:

1. Set the CLKTRIM_REFCON.rcntval register to adjust the calibration time.
2. Set the CLKTRIM_CR.trim_start register to restart the trim.

9.3.2 CLKTRIM monitoring mode

9.3.2.1 Clock monitoring principle

In order to monitor whether the system working clock (monitored clock CAL_CLK) is working normally, a stable clock is required as a reference clock (REF_CLK). Set the monitoring time Ttrim, the monitored clock frequency Fcal, the reference clock frequency Fref, the monitored clock counter overflow value CALVAL, use the monitored clock and the reference clock to count at the same time, stop counting when the monitoring time ends, and judge whether the monitored clock counter is in overflow state. If the monitored clock counter has overflowed, it means that the monitored clock works normally within the monitoring time, and the next monitoring is continued. If the counter of the monitored clock does not overflow, it means that the monitored clock is working abnormally within the monitoring time. The monitored clock may have stopped or the frequency has changed

significantly, and the monitored clock is abnormally interrupted, and the system working clock is switched by the hardware.

9.3.2.2 Clock monitoring hardware structure

The monitored clock of the time calibration module has 3 clock sources, and the reference clock has 6 clock sources, as shown in the following figure:

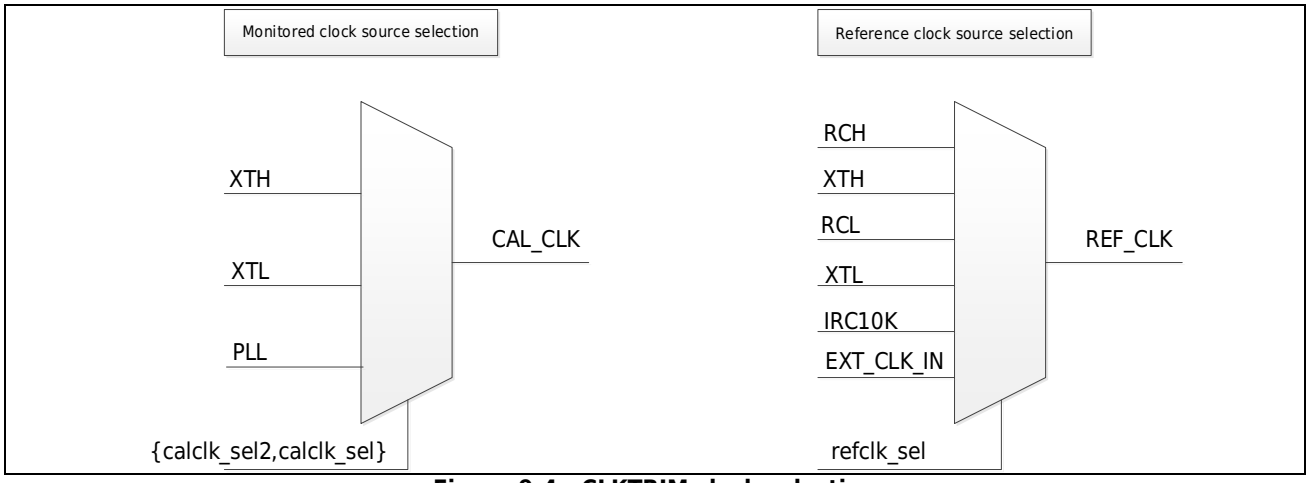


Figure 9-4 CLKTRIM clock selection

The selection of the monitored clock is configured by the register CLKTRIM_CR.calclk_sel2,CLKTRIM_CR.calclk_sel, and the selection of the reference clock is configured by the register CLKTRIM_CR.refclk_sel.

9.3.2.3 Clock Monitoring Software Flow

1. Set the CLKTRIM_CR.refclk_sel register to select the reference clock.
2. Set the CLKTRIM_CR.calclk_sel register to select the monitored clock.
3. Set the CLKTRIM_REFCON.rcntval register to monitor interval time.
4. Set the CLKTRIM_CALCON.ccntval register to the overflow time of the monitored clock counter.
5. Set the CLKTRIM_CR.mon_en register to enable the monitor function.
6. Set CLKTRIM_CR.IE register to enable interrupt.
7. Set the CLKTRIM_CR.trim_start register to start monitoring.
8. The reference clock counter and the monitored clock counter start counting.
9. When the counting of the reference clock counter reaches the monitoring interval time, CLKTRIM_IFR.stop is set to 1, and at the same time, it is judged whether the monitored clock counter overflows, that is, whether CLKTRIM_IFR.calcnt_of is set to 1. If it overflows, that is, CLKTRIM_IFR. calcnt_of is 1, it means that the monitored clock works normally. If there is no overflow, that is, CLKTRIM_IFR.calcnt_of is 0, it means that the monitored clock is invalid, and CLKTRIM_IFR.xtl_fault/xth_fault is set to 1, triggering an interrupt.
10. Process the interrupt service subroutine, clear the interrupt flag bit

CLKTRIM_IFR.xtl_fault/xth_fault, and clear the CLKTRIM_CR.trim_start register to end monitoring.

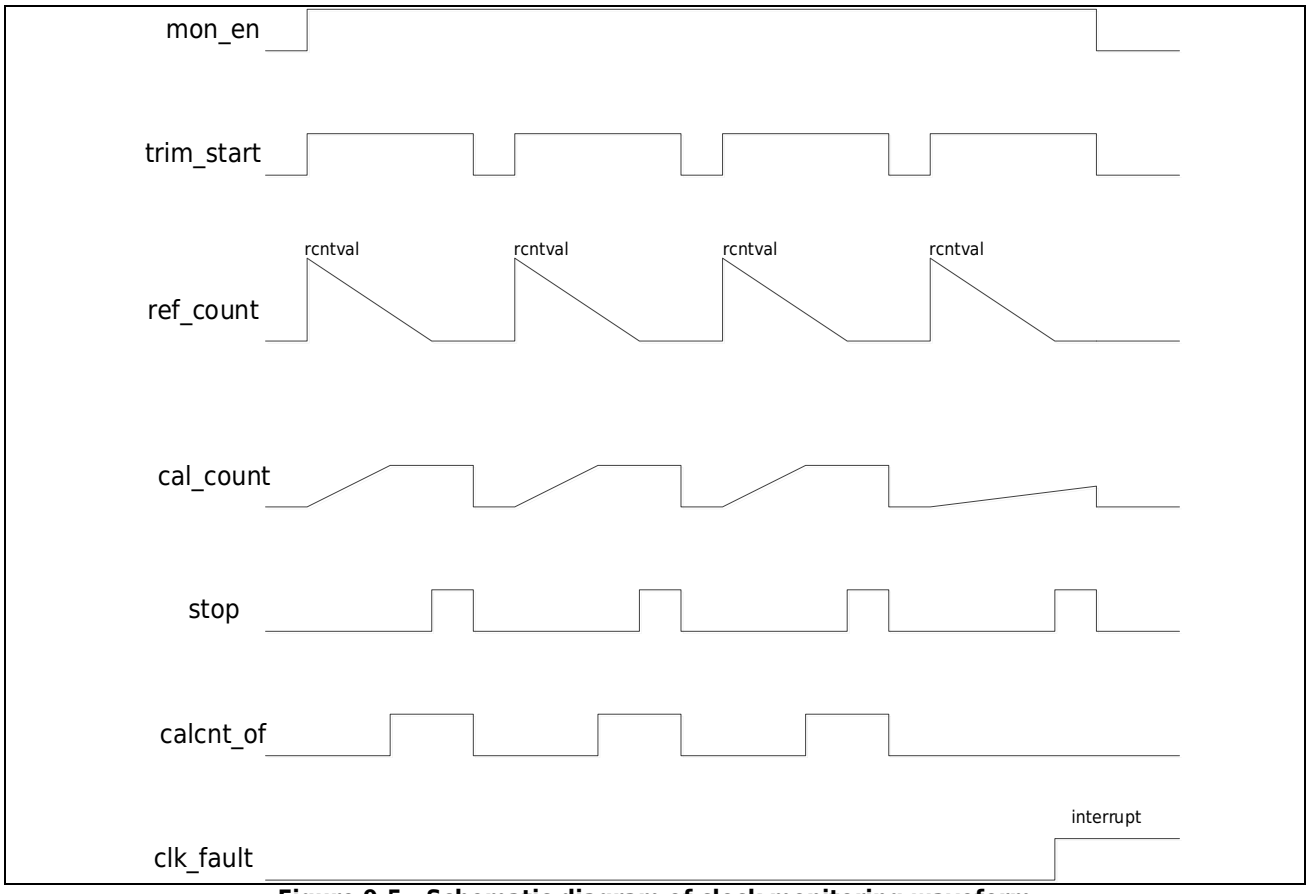


Figure 9-5 Schematic diagram of clock monitoring waveform

9.4 CLKTRIM register description

Register list

Base address: 0x40001800

Table 9-1 Register List

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 bit clear register
0x18	CLKTRIM_CALCON	RW	Calibration Counter Overflow Value Configuration Register

9.4.1 Configuration Register (CLKTRIM_CR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							calclk_sel2	IE	mon_en	calclk_sel		refclk_sel			trim_start
							RW	RW	RW	RW		RW			RW

Bit	Marking	Functional description
31:9	Reserved	
8	calclk_sel2	To-be-calibrated / monitored clock selection high register
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 low register calclk_sel2，calclk_sel: 000 RCH; 001 XTH; 010 RCL; 011 XTL; 100 PLL; 101 IRC48M
3:1	refclk_sel	Reference Clock Select Register 000 ---- RCH 001 ---- XTH 010 ---- RCL 011 ---- XTL 100 ---- IRC10K 101 ---- EXT_CLK_IN
0	trim_start	Calibration / Monitoring Start Register 0 – Stop 1 – Start

9.4.2 Reference counter initial value configuration register (CLKTRIM_REFCON)

Address offset: 0x04

Reset value: 0x00000000

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

Bit	Marking	Functional description
31:0	rcntval	Reference counter initial value

9.4.3 Reference Counter Value Register (CLKTRIM_REFCNT)

Address offset: 0x08

Reset value: 0x00000000

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

Bit	Marking	Functional description
31:0	refcnt	Reference counter value

9.4.4 Calibration Counter Value Register (CLKTRIM_CALCNT)

Address offset: 0x0c

Reset value: 0x00000000

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

Bit	Marking	Functional description
31:0	calcnt	Calibration counter value

9.4.5 Interrupt Flag Register (CLKTRIM_IFR)

Address offset: 0x10

Reset value: 0x0000 0000

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

Bit	Marking	Functional description
31:5	Reserved	
4	pll_fault	PLL failure flag. CLKTRIM_ICLR.pll_fault_clr write zero to clear this flag
3	xth_fault	XTH failure flag. CLKTRIM_ICLR.xth_fault_clr write zero to clear this flag
2	xtl_fault	XTL failure flag. CLKTRIM_ICLR.xtl_fault_clr write zero to clear this flag
1	calcnt_of	Calibration counter overflow flag. CLKTRIM_CR.start write zero to clear this flag
0	stop	Reference counter stop flag. CLKTRIM_CR.start write zero to clear this flag

9.4.6 Interrupt flag clear register (CLKTRIM_ICLR)

Address offset: 0x14

Reset value: 0x1111 1111

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											pll_fault_clr	xth_fault_clr	xtl_fault_clr	Reserved	
											R1W0	R1W0	R1W0		

Bit	Marking	Functional description
31:5	Reserved	
4	pll_fault_clr	Clear the PLL failure flag, write zero to clear.
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	

9.4.7 Calibration Counter Overflow Value Configuration Register (CLKTRIM_CALCON)

Address offset: 0x18

Reset value: 0xffff ffff

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

Bit	Marking	Functional description
31:0	ccntval	Calibration counter overflow value

10 Hardware divider module (HDIV)

10.1 Introduction to HDIV

HDIV (Hardware Divider) is a 32-bit signed/unsigned integer hardware divider.

10.2 HDIV main characteristics

The HDIV hardware divider supports the following features:

- Configurable signed/unsigned integer division calculation
- 32-bit dividend, 16-bit divisor
- Output 32-bit quotient and 32-bit remainder
- Divide by zero warning flag, division end flag
- 10 clock cycles to complete a division operation
- Write the divisor register to trigger the start of the division operation
- Automatically wait for the end of the calculation when reading the quotient register/remainder register

10.3HDIV function description

10.3.1 HDIV operation process

1. Turn on the clock enable register for the hardware divider in the system controller.
2. The configuration register HDIV_SIGN sets signed/unsigned division operation.
3. The configuration register HDIV_DIVIDEND sets the dividend.
4. The configuration register HDIV_DIVISOR sets the divisor.
5. When the division operation starts, query the register HDIV_STAT operation end flag bit div_end, and div_end is 1 to indicate the end of the operation. Read the register HDIV_QUOTIENT to get the quotient, and read the register HDIV_REMAINDER to get the remainder.
6. When the divisor is zero, the division operation ends immediately, and the operation result remains the result of the last operation, and the divisor is zero warning flag bit div_zero is set.
7. Before the end of the division operation, when reading the register HDIV_QUOTIENT/HDIV_REMAINDER, the CPU will be held until the end of the operation.

Example: Calculate an unsigned division, the dividend is 1917887483 (0x7250A3FB), and the divisor is 9597 (0x257D)

Step 1, the configuration register HDIV_SIGN is 0, that is, unsigned division operation

Step 2, configure the register HDIV_DIVIDEND as 0x7250A3FB, that is, set the dividend

Step 3, the configuration register HDIV_DIVISOR is 0x257D, that is, the divisor is set, and the calculation starts

Step 4, query register HDIV_STAT operation end flag div_end, if div_end is 1 flag, the operation ends.

Read the register HDIV_QUOTIENT to get the quotient 199842 (0x30CA2)

Read the register HDIV_REMAINDER to get the remainder 3809 (0xEE1)

10.4HDIV register description

Register list Base address: 0x40021800

Table 10-1 Register List

Offset	Register name	Access	Register description
0x00	HDIV_DIVIDEND	RW	The dividend register.
0x04	HDIV_DIVISOR	RW	Divisor register.
0x08	HDIV_QUOTIENT	RO	Merchant register.
0x0c	HDIV_REMAINDER	RO	Remainder register.
0x10	HDIV_SIGN	RW	Symbolic register
0x14	HDIV_STAT	RO	Status register

10.4.1 Dividend register (HDIV_DIVIDEND)

Address offset: 0x00

Reset value: 0x0000 0000

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

Bit	Marking	Functional description
31:0	DIVIDEND	Dividend value register

10.4.2 Divisor register (HDIV_DIVISOR)

Address offset: 0x04

Reset value: 0x00000001

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
DIVISOR															
RW															

Bit	Marking	Functional description
31:16	Reserved	
15:0	DIVISOR	Divisor value register (writing to this register automatically triggers a division operation)

10.4.3 Quotient register (HDIV_QUOTIENT)

Address offset: 0x08

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
QUOTIENT[31:16]															
RO															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUOTIENT[15:0]															
RO															

Bit	Marking	Functional description
31:0	QUOTIENT	Quotient result register

10.4.4 Remainder register (HDIV_REMAINDER)

Address offset: 0x0c

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REMAINDER[31:16]															
RO															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REMAINDER[15:0]															
RO															

Bit	Marking	Functional description
31:0	REMAINDER	Remainder result register

10.4.5 Sign register (HDIV_SIGN)

Address offset: 0x10

Reset value: 0x00000000

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

Bit	Marking	Functional description
31:1	Reserved	
0	sign	Symbol selection register. 0 --- Unsigned division operation 1 --- Signed division operation

10.4.6 Status Register (HDIV_STAT)

Address offset: 0x14

Reset value: 0x00000001

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														div_zero	div_end
														RO	RO

Bit	Marking	Functional description
31:2	Reserved	
1	div_zero	Divisor by zero warning flag. 0 --- divisor is not zero 1 --- divisor is zero
0	div_end	The end flag of the division operation. 0 --- operation in progress 1 --- operation completed

11 FLASH controller (FLASH)

11.1 Overview

This system includes a FLASH memory with a capacity of 16k/32k/64k/128k bytes (Byte), which is divided into 32/64/128/256 pages (Sector), and the capacity of each page (Sector) is 512 bytes (Byte). FLASH controller supports erasing, programming and reading operations on the FLASH memory. This controller also supports erasing and writing protection of FLASH memory, and writing protection of control registers.

11.2 FLASH capacity division

Table 11-1 FLASH capacity division

Address	Serial number		Address	Serial number
0x0E00 – 0x0FFF	Sector7	.....	0x1FE00 – 0x1FFFF	Sector255
0x0C00 – 0x0DFF	Sector6	.....	0x1FC00 – 0x1FDFF	Sector254
0x0A00 – 0x0BFF	Sector5	.....	0x1FA00 – 0x1FBFF	Sector253
0x0800 – 0x09FF	Sector4	.....	0x1F800 – 0x1F9FF	Sector252
0x0600 – 0x07FF	Sector3	.....	0x1F600 – 0x1F7FF	Sector251
0x0400 – 0x05FF	Sector2	.....	0x1F400 – 0x1F5FF	Sector250
0x0200 – 0x03FF	Sector1	.....	0x1F200 – 0x1F3FF	Sector249
0x0000 – 0x01FF	Sector0	.....	0x1F000 – 0x1F1FF	Sector248

Note:

- 16KByte FLASH consists of Sector0-32;
- 32KByte FLASH consists of Sector0-63;
- 64KByte FLASH consists of Sector0-127;
- 128KByte FLASH consists of Sector0-255.

11.3 Functional description

This controller supports data read and write operations of three-bit widths of Byte (8 bits), half-word (16 bits) and word (32 bits) of FLASH. Note that the target address of the Byte operation must be byte-aligned, the target address of the half-word operation must be half-word-aligned (the lowest bit of the address is 1'b0), and the address of the word operation must be word-aligned (the lowest two bits of the address are 2 'b00). If the target address is not aligned according to the bit width, the operation is invalid and the CPU will enter the Hard Fault interrupt.

This controller adopts high-security hardware design and has the function of FLASH operation source defense: only when the address of the FLASH operation function is located at 0~32K, can the FLASH erase and write operation be performed correctly.

The 0~32K of the FLASH address has higher security, important functions must be placed in this area; such as important program entry, interrupt entry function, high security algorithm module, UID, AES, true random number, RTC algorithm cooperation, composed of high Security authentication system.

11.3.1 Sector Erase (Sector Erase)

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

Sector erase operation steps are as follows:

Step1: Configure FLASH erasing parameters, see chapter 11.4 for details.

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

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

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

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

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

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

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

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

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

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

Note:

- **The address of the code for page erasing the FLASH must be less than 32768.**

11.3.2 Full Chip Erase (Chip Erase)

Full chip erase can erase all pages (Sector) at one time. After the erase operation is completed, the data in all pages (Sector) are 0xFF. If the erase operation is performed from the FLASH, the operation will be prohibited. Because this operation will erase the program segment where the current PC is located. If this happens, the error flag will be set; if the erase operation is performed from RAM, the

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

Chip erase operation steps are as follows:

Step1: Configure FLASH erasing parameters, see chapter 11.4 for details.

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

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

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

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

Step6: Set FLASH_SLOCK0 to 0xFFFFFFFF to remove the erase and write protection of all Sectors.

Step7: Check if FLASH_SLOCK0 is 0xFFFFFFFF, if not 0xFFFFFFFF, jump to Then jump to Step5.

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

Step9: Set FLASH_SLOCK1 to 0xFFFFFFFF to remove the erase and write protection of all Sectors.

Step10: Check whether FLASH_SLOCK1 is 0xFFFFFFFF, if it is not 0xFFFFFFFF, jump to Then jump to Step8.

Step11: Write any address in the Chip to be erased to trigger Chip erasure.

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

Step12: Wait for FLASH_CR. BUSY to become 0, and the Chip erase operation is completed.

11.3.3 Write operation (Program)

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

Byte write operation steps are as follows:

Step1: Configure FLASH erasing parameters, see chapter 11.4 for details.

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

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

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

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

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

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

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

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

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

Step10: To write Byte to other addresses, repeat Step8 – Step9.

Half-word write operation steps are as follows:

Step1: Configure the flash erasing time, see chapter 11.4 for details.

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

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

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

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

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

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

Step8: Perform a Half-word write operation on the target address to be written to trigger the write operation.

Example: * ((unsigned short int *) 0x00001232) = 0xABCD.

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

Step10: To write Half-word to other addresses, repeat Step8 - Step9.

Word write operation steps are as follows:

Step1: Configure FLASH erasing parameters, see chapter 11.4 for details.

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

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

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

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

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

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

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

Example: * ((unsigned int *) 0x00001234) = 0x55667788.

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

Step10: If you need to write Word to other addresses, repeat Step8 - Step9.

Note:

- **The address of the code that writes to FLASH must be less than 32768.**

11.3.4 Read operation (Read)

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

example:

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

Half-word read operation: *temp = * ((unsigned short int *) 0x00001232)*

Word read operation: *temp = * ((unsigned int *) 0x00001234)*

11.4Erase time

FLASH memory has strict time requirements for the control signals of the erasing and programming operations, and failure of the timing of the control signals will cause the erasing and programming operations to fail. When powering on, the erasing parameters when HCLK is 4MHz are loaded by default; if the HCLK frequency is not 4MHz when erasing Flash, the user program should load the erasing parameters corresponding to the HCLK frequency. When operating on FLASH, the highest supported HCLK is 48MHz.

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

The following table shows the corresponding FLASH erasing time parameters at different frequencies:

Table 11-2 FLASH erasing time parameters at different frequencies

Parameter	4M	8M	16M	24M	32M	48M
TNVS	0x20	0x40	0x80	0xC0	0x100	0x180
TPGS	0x17	0x2E	0x5C	0x8A	0xB8	0xFF
TPROG	0x1B	0x36	0x6C	0xA2	0xD8	0x144
TSERASE	0x4650	0x8CA0	0x11940	0x1A5E0	0x23280	0x34BC0
TMERASE	0x222E0	0x445C0	0x88B80	0xCD140	0x111700	0x19A280
TPRCV	0x18	0x30	0x60	0x90	0xC0	0x120
TSRCV	0xF0	0x1E0	0x3C0	0x5A0	0x780	0xB40
TMRCV	0x3E8	0x7D0	0xFA0	0x1770	0x1F40	0x2EE0

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

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

Step2: Write 0x40 to the FLASH_TNVS register, if the read value of this register is not 0x40, then jump to the previous step.

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

Step4: Write 0x2E to the FLASH_TPGS register, if the read value of this register is not 0x2E, then jump to the previous step.

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

Step6: Write 0x36 to the FLASH_TPROG register, if the read value of this register is not 0x36, then jump to the previous step.

Step7: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step8: Write 0x8CA0 to the FLASH_TSERASE register, if the read value of this register is not 0x8CA0, then jump to the previous step.

Step9: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step10: Write 0x445C0 to the FLASH_TMERASE register, if the read value of this register is not 0x445C0, then jump to the previous step.

Step11: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step12: Write 0x30 to the FLASH_TPRCV register, if the read value of this register is not 0x30, then jump to the previous step.

Step13: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step14: Write 0x1E0 to the FLASH_TSRCV register, if the read value of this register is not 0x1E0, then jump to the previous step.

Step15: Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step16: Write 0x7D0 to the FLASH_TMRCV register, if the read value of this register is not 0x7D0, then jump to the previous step.

11.5Read wait period

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

11.6Erase and write protection

11.6.1 Erase and write protection bit

The entire 16k/32k/64k128k byte FLASH memory is divided into 32/64/128/256 pages, and every 4 pages share an erase/write protection bit. When the page is protected, the erasing and writing operations on the page are invalid and an alarm flag and interrupt signal are generated. When any page in the FLASH memory is protected, the whole-chip erasing of the FLASH is invalid, and an alarm flag and an interrupt signal are generated.

11.6.2 PC address erase and write protection

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

11.7Register write protection

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

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

FLASH_TNVS, FLASH_TPGS, FLASH_TPROG, FLASH_TSERASE, FLASH_TMERASE, FLASH_TPRCV, FLASH_TSRCV, FLASH_TMRCV, FLASH_CR, FLASH_SLOCK0, FLASH_SLOCK1.

Registers that can be changed without a write sequence are as follows:

FLASH_ICLR, FLASH_BYPASS.

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

Step1: Write 0x5A5A to the FLASH_BYPASS register.

Step2: Write 0xA5A5 to the FLASH_BYPASS register.

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

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

Step5: Perform other operations.

Note:

- Write 0x5a5a, 0xa5a5, and write the target register. Do not insert any write operations (write ROM, RAM, REG) between these three steps of writing operations, otherwise the value of the target register cannot be rewritten. If rewriting fails, you need to perform these three steps again.

11.8Register

Base address: 0x4002 0000

Register	Offset address	Description
FLASH_TNVS	0x00	Tnvs time parameter
FLASH_TPGS	0x04	Tpgs time parameter
FLASH_TPROG	0x08	Tprog time parameter
FLASH_TSERASE	0x0C	Tserase time parameter
FLASH_TMERASE	0x10	Tmerase time parameter
FLASH_TPRCV	0x14	Tprcv time parameter
FLASH_TSRCV	0x18	Tsrcv time parameters
FLASH_TMRCV	0x1C	Tmrcv time parameter
FLASH_CR	0x20	control register
FLASH_IFR	0x24	Interrupt Flag Register
FLASH_ICLR	0x28	Interrupt Flag Clear Register
FLASH_BYPASS	0x2C	0x5a5a-0xa5a5 Bypass sequence register
FLASH_SLOCK0	0x30	Sector0-127 erase and write protection register
FLASH_SLOCK1	0x34	Sector128-255 erase and write protection register

11.8.1 TNVS parameter register (FLASH_TNVS)

Offset address: 0x00

Reset value: 0x0000 0020

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

Bit	Marking	Functional description
31:9	Reserved	
8:0	TNVS	Calculation formula: TNVS = 8*HCLK, the unit of HCLK is MHz. 4MHz example: TNVS = 8*4 = 32.

11.8.2 TPGS parameter register (FLASH_TPGS)

Offset address: 0x04

Reset value: 0x0000 0017

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

Bit	Marking	Functional description
31:8	Reserved	
7:0	TPGS	Calculation formula: TPGS = 5.75*HCLK, the unit of HCLK is MHz. 4MHz example: TPGS = 5.75*4 = 23. Note: When the calculated value is greater than 0xFF, TPGS should be assigned a value of 0xFF.

11.8.3 TPROG parameter register (FLASH_TPROG)

Offset address: 0x08

Reset value: 0x0000 001B

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

Bit	Marking	Functional description
31:9	Reserved	
8:0	TPROG	Calculation formula: TPROG = 6.75*HCLK, the unit of HCLK is MHz. 4MHz example: TPROG = 6.75*4 = 27.

11.8.4 TSERASE register (FLASH_TSERASE)

Offset address: 0x0C

Reset value: 0x0000 4650

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

Bit	Marking	Functional description
31:18	Reserved	
17:0	TSERASE	Calculation formula: TSERASE = 4500*HCLK, the unit of HCLK is MHz. 4MHz example: TSERASE = 4500*4 = 18000.

11.8.5 TMERASE parameter register (FLASH_TMERASE)

Offset address: 0x10

Reset value: 0x000222E0

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

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

Bit	Marking	Functional description
31:21	Reserved	
20:0	TMERASE	Calculation formula: TMERASE = 35000*HCLK, the unit of HCLK is MHz. 4MHz example: TMERASE = 35000*4 = 140000.

11.8.6 TPRCV parameter register (FLASH_TPRCV)

Offset address: 0x14

Reset value: 0x0000 0018

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

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

Bit	Marking	Functional description
31:12	Reserved	
11:0	TPRCV	Calculation formula: TPRCV = 6*HCLK, the unit of HCLK is MHz. 4MHz example: TPRCV = 6*4 = 24.

11.8.7 TSRCV parameter register (FLASH_TSRCV)

Offset address: 0x18

Reset value: 0x0000 00F0

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

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

Bit	Marking	Functional description
31:12	Reserved	
11:0	TSRCV	Calculation formula: TSRCV = 60*HCLK, the unit of HCLK is MHz. 4MHz example: TSRCV = 60*4 = 240.

11.8.8 TMRCV parameter register (FLASH_TMRCV)

Offset address: 0x1C

Reset value: 0x0000 03E8

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

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

Bit	Marking	Functional description
31:14	Reserved	
13:0	TMRCV	Calculation formula: TMRCV = 250*HCLK, the unit of HCLK is MHz. 4MHz example: TMRCV = 250*4 = 1000.

11.8.9 CR register (FLASH_CR)

Offset address: 0x20

Reset value: 0x0000 0200

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						DPSTB_EN	Reserved		IE		BUSY	WAIT		OP	
						RW			RW		RO	RW		RW	

Bit	Marking	Functional description
31:10	Reserved	
9	DPSTB_EN	FLASH dpstb enable Mask bit; 0: When the system enters deepsleep mode, FLASH does not enter low power consumption mode; 1: When the system enters deepsleep mode, FLASH enters low power consumption mode;
8:7	Reserved	
6:5	IE	IE[6]: FLASH erase and write protected address interrupt enable; 0: Disable; 1: Enable IE[5]: FLASH erase PC value interrupt enable; 0: Disable; 1: Enable
4	BUSY	Idle / busy flag; 0: idle state; 1: busy state;
3:2	WAIT	Read FLASH cycle; 0~24MHz: 00/11, 1 cycle; 24~48MHz: 01: 2 cycles; 48~72MHz: 10:3 cycles; (the highest clock of this series of products is 48MHz)
1:0	OP	FLASH operation; 00: read (read); 01: write (program); 10: page erase (sector erase); 11: full chip erase (chip erase)

11.8.10 IFR register (FLASH_IFR)

Offset address: 0x24

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														IF1	IF0
														RO	RO

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

11.8.11 ICLR register (FLASH_ICLR)

Offset address: 0x28

Reset value: 0x0000 000F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														ICLR1	ICLR0
														R1W 0	R1W 0

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

11.8.12 BYPASS register (FLASH_BYPASS)

Offset address: 0x2C

Reset value: 0x0000 0000

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

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

11.8.13 SLOCK0 register (FLASH_SLOCK0)

Offset address: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLOCK0[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLOCK0[15:0]															
RW															

Bit	Marking	Description
31:0	SLOCK	Sector erase and write protection bit; 0: not allowed to erase and write; 1: allowed to erase and write Bit [0] corresponds to: Sector0-1-2-3 Bit [1] corresponds to: Sector4-5-6-7 Bit [2] corresponds to: Sector8-9-10-11 Bit [3] corresponds to: Sector12-13-14-15 ..... Bit [31] corresponds to: Sector124-125-126-127

11.8.14 SLOCK1 register (FLASH_SLOCK1)

Offset address: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLOCK1[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLOCK1[15:0]															
RW															

Bit	Marking	Description
31:0	SLOCK	Sector erase and write protection bit; 0: not allowed to erase and write; 1: allowed to erase and write Bit [0] corresponds to: Sector128-129-130-131 Bit [1] corresponds to: Sector132-133-134-135 Bit [2] corresponds to: Sector136-137-138-139 Bit [3] corresponds to: Sector140-141-142-143 ..... Bit [31] corresponds to: Sector252-253-254-255

12 RAM controller (RAM)

12.1 Overview

This system includes a SRAM with a capacity of 2k/4k/8k/16k bytes (Byte), which supports three kinds of read and write operations: byte (8 bits), half word (16 bits), and word (32 bits). Read and write operations can be performed at the system clock frequency without waiting for cycles. In addition, this controller also supports parity check, which can perform parity check on each byte (Byte) of SRAM data, and generate a parity check error interrupt.

12.2 Functional description

12.2.1 RAM address range

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

Table 12-1 RAM Address Mapping

Address range	Size	Memory type
0x2000_0000 – 0x2000_3FFF	16KByte	SRAM

12.2.2 Read and write bit width

This controller supports Byte (8 bits), halfword (16 bits), word (32 bits) Read and write operations with three-bit widths. The address of the byte operation must be byte-aligned, the target address of the half-word operation must be half-word-aligned (the lowest bit of the address is 1'b0), the address of the word operation must be word-aligned (the lowest two bits of the address are 2'b00). 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.

12.2.3 Parity

This controller supports parity check of SRAM data. When writing data to SRAM, do a parity check on each byte of data, and store the 1-bit check value and 8-bits data into the SRAM together. When reading data from SRAM, the controller will read 8bits data and 1bit check value, and perform parity check. If the check is wrong, the parity check error flag will be set. When the interrupt is enabled, a Error interrupt.

Note:

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

12.3Register

Base address: 0x4002 0400

Table 12-2 Register base address

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

12.3.1 Control Register (RAM_CR)

Offset address: 0x00

Reset value: 0x0000 0000

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

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

Bit	Marking	Functional description
31:2	Reserved	Keep
1	IE	Error alarm interrupt enable signal; 1: enable alarm interrupt, 0: disable alarm interrupt;
0	Reserved	Keep

12.3.2 Parity Error Address Register (RAM_ERRADDR)

Offset address: 0x04

Reset value: 0x0000 0000

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

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

Bit	Marking	Functional description
31:14	Reserved	
13:0	ERRADDR	Parity check error Byte address; after the interrupt flag is cleared, the address is cleared at the same time;

12.3.3 Error Interrupt Flag Register (RAM_IFR)

Offset address: 0x08

Reset value: 0x0000 0000

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

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

12.3.4 Error Interrupt Flag Clear Register (RAM_ICLR)

Offset address: 0x0C

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ERR CLR
															R1W 0

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

13 DMA controller (DMAC)

13.1 Introduction to DMAC

Direct memory access (DMA) is used to provide high-speed data transmission between peripherals and memory or between memory and memory; the CPU does not need to participate in the transmission process; so the CPU can perform other operations synchronously. The DMAC has two independent DMA channels, each channel is dedicated to managing requests from peripherals or memory access. There is also an arbiter to coordinate the priority of each DMA request.

13.2 DMAC main characteristics

- 2 independent DMA channels, support priority configuration
- 3 data transfer widths: 8-bit, 16-bit, 32-bit
- 4 transmission modes: Software Block, Software Burst, Hardware Block, Hardware Burst
- 2 source address types: peripheral, memory
- 2 target address types: peripheral, memory
- 2 address change modes: fixed, auto-increment
- Transmission address addressing range: 0x00000000 ~ 0xFFFFFFFF
- The number of data blocks to be transmitted is configurable: 1~65536
- The size of the data block to be transmitted is configurable: 1~16
- Support address and transfer number reload function

13.3Functional block diagram

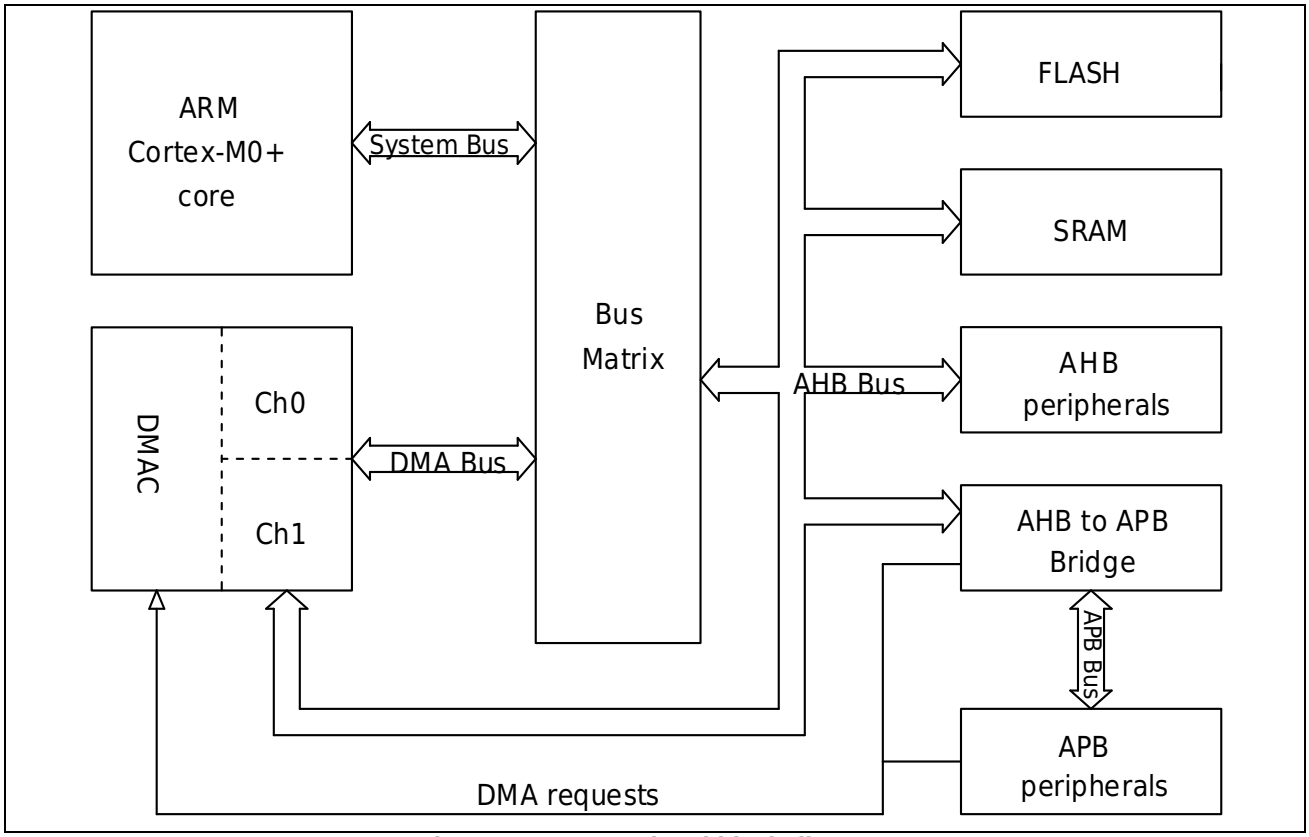


Figure 13-1 Functional block diagram

- DMAC

  The DMAC has two DMA channels, and when conflicts occur between channels, the priority controller conducts arbitration.

- Bus Matrix

  CPU and DMAC are connected to the bus matrix. When the CPU releases the AHB bus, the DMAC can access the AHB bus device and the APB bus device through the Bus Matrix. The priority of CPU accessing the bus is higher than that of DMAC.

- DMA requests

  If the peripheral supports DMA request, the DMA channel can be configured as a hardware trigger, otherwise it can only be configured as a software trigger.

13.4Functional description

13.4.1 DMAC transfer modes

The DMA controller supports 4 transfer modes, software Block transfer mode, software Burst transfer mode, hardware Block transfer mode, and hardware Burst transfer mode.

The 4 transmission modes is shown in the table below:

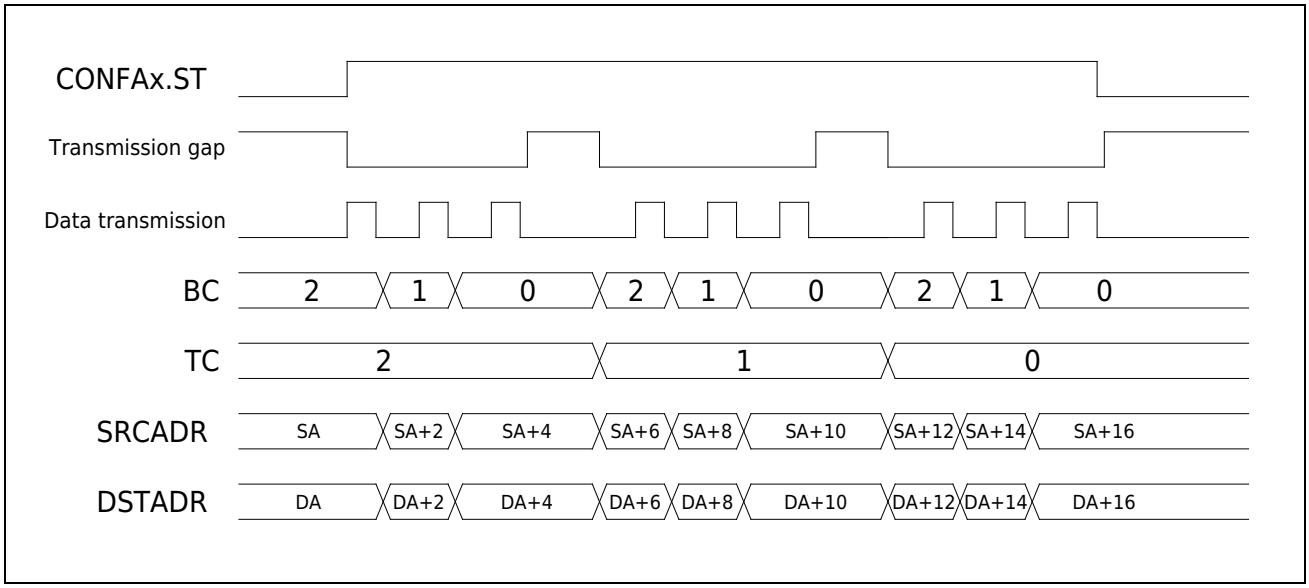
Compare Items	Software Block Transmission	Software Burst Transmission	Hardware Block Transmission	Hardware Burst Transmission
Interrupted by higher priority DMA channel or CPU	Can	Can't	Can	Can't
Conditions that trigger the start of the transfer	Write DMAC Register		Peripheral interrupt flag	
The amount of data that triggers a transfer	(BC+1) * (TC+1)		BC+1	(BC+1) * (TC+1)
The total amount of data transferred after configuration	(BC+1) * (TC+1)			
Main application scenarios	• Memory to memory • Memory to peripherals without DMA requests		• Memory to peripherals with DMA requests • With DMA requests to memory	• Memory to memory • Memory to peripherals without DMA requests

13.4.2 DMA software block transfer mode

When configuring DMAC_CONFAx.TRI_SEL=0 and DMAC_CONFBx.MODE=0, DMAC works in software Block transfer mode.

When the user code writes 1 to DMAC_CONFAx.ST, the DAMC is triggered to start the software Block transfer mode; the DMA transfer stops after completing (BC+1) * (TC+1) data. Each transmission of BC+1 data is completed, the DMAC inserts a transmission gap, and the priority arbiter performs priority arbitration in the gap. The CPU or a higher priority DMA channel requests the bus during this gap, the bus ownership is transferred to the CPU or a higher priority DMA channel; when the CPU or a higher priority DMA channel releases the bus, the unfinished data transfer will continue to be completed.

The software Block transmission diagram is shown below, where SA represents the source address and DA represents the target address; The size of the data block is 3 (BC=2), the number of data blocks is 3 (TC=2), the data width is 16bit, and the address increases automatically.

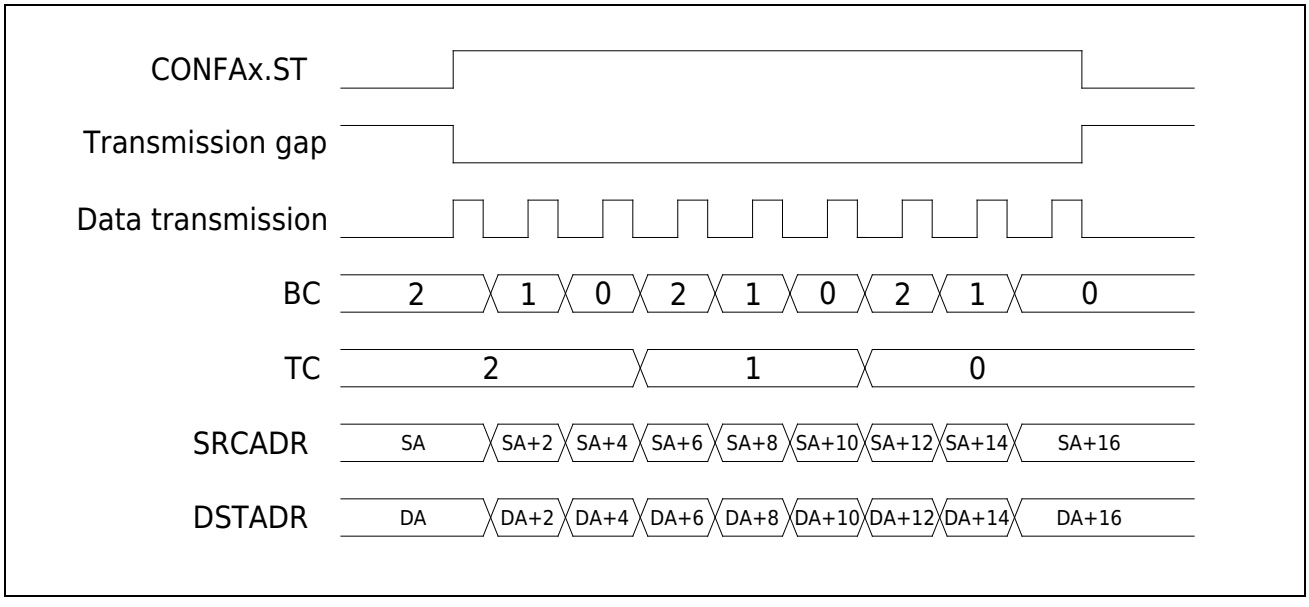


13.4.3 DMA software Burst transfer mode

When configuring DMAC_CONFAx.TRI_SEL=0 and DMAC_CONFBx.MODE=1, DMAC works in software Burst transfer mode.

When the user code writes 1 to DMAC_CONFAx.ST, the DMAC is triggered to start the software Burst transfer mode; the DMA transfer stops after completing (BC+1) * (TC+1) data. The DMA will occupy the bus until all data transfers are completed, and the CPU or a higher-priority DMA channel can only access the bus after the operation is completed. The CPU may not work completely for a period of time.

The software Burst transmission diagram is as follows, where SA represents the source address, DA represents the destination address; the data block size is 3 (BC=2), the number of data blocks is 3 (TC=2), the data width is 16bit, and the address is self-incrementing.



13.4.4 DMA hardware block transfer mode

When configuring DMAC_CONFAx.TRI_SEL= non-zero and DMAC_CONFBx.MODE=0, DMAC works in hardware Block transfer mode.

Whenever the peripheral DMA request signal appears once, the DMAC is triggered to start a hardware block transfer, and transfer (BC+1) data. It needs to appear (TC+1) peripheral DMA request signal to complete the transmission of all DMA data. Every BC+1 data is completed, the DMAC inserts a transmission gap, and the priority arbiter performs priority arbitration in the gap. The CPU or a higher priority DMA channel is also requesting the bus during this gap, the bus ownership is transferred to the CPU or a higher priority DMA channel; when the CPU or a higher priority DMA channel releases the bus, the unfinished data transfer will continue Finish.

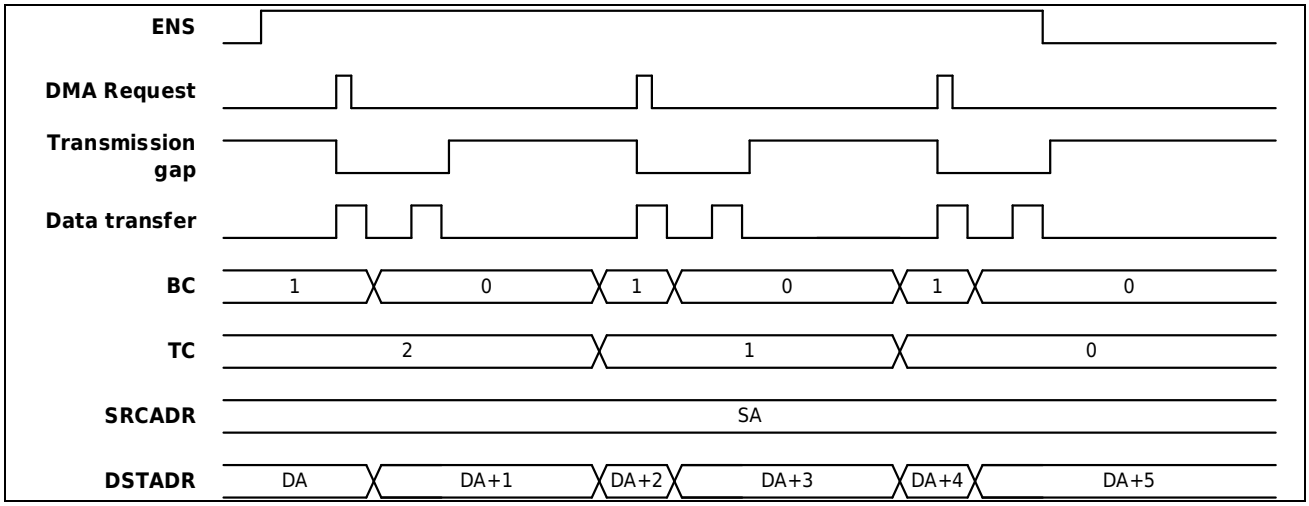
Note:

- **In this mode, generally set BC to 0 to obtain a data from the peripheral or provide a data to the peripheral.**

Request signals supported by DMA are as follows:

- LPUARTx / UARTx receiving Buf is not empty, sending Buf is empty
- SPIx receiving Buf is not empty, sending Buf is empty
- I2Sx receiving Buf is not empty, sending Buf is empty
- TIMx capture complete, comparison match
- ADC queue conversion completed
- ADC sequential conversion complete
- LCD timing interrupt
- DAC conversion complete

The hardware block transmission diagram is as follows, where SA represents the source address, DA represents the target address; the data block size is 2 (BC=1), the number of data blocks is 3 (TC=2), the data width is 8bit, and the target address is self-incrementing, the source address is fixed, and the transmission is automatically disabled after completion.



13.4.5 DMA hardware Burst transfer mode

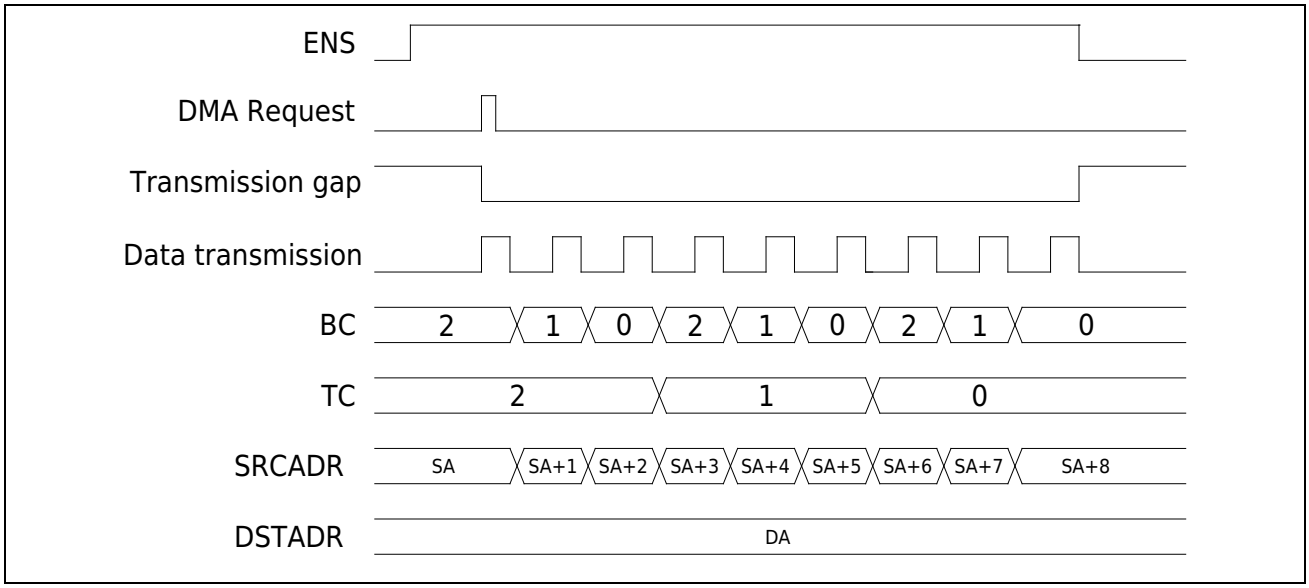
When configuring DMAC_CONFAx.TRI_SEL= non-zero and DMAC_CONFBx.MODE=1, DMAC works in hardware Burst transfer mode.

When the peripheral DMA request signal appears, the DMAC is triggered to start the hardware Burst transmission mode, and stops after the transmission is completed (BC+1) * (TC+1) data. The DMA will occupy the bus until all data transfers are completed, and the CPU or a higher-priority DMA channel can only access the bus after the operation is completed. The CPU may not work completely for a period of time.

Request signals supported by DMAC are as follows:

- LPUARTx / UARTx receiving Buf is not empty, sending Buf is empty
- SPIx receiving Buf is not empty, sending Buf is empty
- I2Sx receiving Buf is not empty, sending Buf is empty
- TIMx capture complete, comparison match
- ADC queue conversion completed
- ADC sequential conversion complete
- LCD timing interrupt
- DAC conversion complete

The hardware Burst transmission diagram is as follows, where SA represents the source address, DA represents the target address; the data block size is 3 (BC=2), the number of data blocks is 3 (TC=2), the data width is 8bit, and the source address is auto-incremented, the target address is fixed, and the transfer is automatically disabled after completion.



13.4.6 DMA transfer paused

When DMAC works in Block transfer mode, if DMAC_CONF.HALT is set to a non-zero value, all channels that are being transferred will enter the pause state during the transfer interval, and no more data transfer will be performed. When DMAC_CONFAx.PAS is configured as 1, the channel corresponding to x that is being transmitted will enter a pause state during the transmission interval, and no data transmission will be performed. DMAC_CONFAx.PAS is configured as 1, the channel corresponding to x that is being transmitted will enter a pause state during the transmission interval and no longer transmit data. configuring DMAC_CONFAx.PAS to 0, DMAC needs to receive the next trigger signal before continuing the unfinished transmission.

When the DMAC works in the Burst transfer mode and no data transfer is performed, configure DMAC_CONF.HALT to a non-zero value, and the DMA channel enters the pause state and no longer performs data transfer. After configuring DMAC_CONF.HALT to 0, DMAC will start a new complete transmission when it receives the next trigger signal. When the DMAC works in the Burst transfer mode and no data transfer is performed, when DMAC_CONFAx.PAS is set to 1, the DMA channel enters the pause state and no data transfer is performed. After configuring DMAC_CONFAx.PAS to 0, DMAC will start a new complete transmission when it receives the next trigger signal.

When the DMAC is working in the Burst transfer mode and data transfer is in progress, writing a non- zero value to DMAC_CONF.HALT or writing 1 to DMAC_CONFAx.PAS cannot pause the ongoing transfer; the DMA channel will not enter the pause after the current transfer is completed state.

13.4.7 DMA other configuration

13.4.7.1 Data width

Configure DMAC_CONFBx.WIDTH, you can configure the width of each data to be transmitted as 8bit, 16bit, 32bit, note that the bit width should be completely consistent with the bit width of the DMA source address and target address.

13.4.7.2 Block size

Configure DMAC_CONFAx.BC, you can configure the number of data to be transmitted in each data block as 1~16. Note that the value of BC should match the number of consecutive operations provided by the source and destination addresses.

If the source address or destination address can only provide or receive one data at a time, then BC can only be configured as 0. For example, UART/LPUART/SPI can only provide one data to DMA each time, and UART/LPUART/SPI/DAC can only receive one data from DMA each time, so BC can only be configured as 0.

If both the source address and the destination address can provide or receive multiple data, BC can be configured to a larger value to speed up the transmission speed. For example, when transferring from FLASH to RAM or from FLASH to CRC, the source address and target address can be operated continuously, so BC can be configured as 15 to achieve fast transfer.

13.4.7.3 Number of data blocks

Configure DMAC_CONFAx.TC, you can configure the number of data blocks to be transferred from 1 to 65536 after each DMA is started.

13.4.7.4 channel priority

When DMAC_CONF.PRIO=0, DMAC works in fixed priority mode. At this time, CH0 has a higher priority than CH1, and only when CH0 does not perform data transmission, CH1 can perform data transmission.

When DMAC_CONF.PRIO=1, DMAC works in cycle priority mode. At this time, CH0 and CH1 have higher priority in turn, and the two channels will occupy the bus in turn for data transmission.

13.4.7.5 Auto reload

If the automatic reload function is enabled, the data block size, number of data blocks, source address, and destination address will be automatically loaded to the last configuration value after the transmission is completed, and there is no need to configure it every time.

13.4.8 DMA interrupt

DMA supports two types of interrupts: transfer error and transfer complete. When an interrupt is triggered, the status register DMAC_CONFBx.STAT can be checked to determine what condition

triggered the interrupt, and the built-in interrupt status flag can be cleared by clearing the register DMAC_CONFBx.STAT.

The five conditions that trigger the DMA interrupt are as follows:

- Transport address out of addressing range
- Peripheral request to stop DMA
- Error accessing the transfer source address
- Access transfer destination address error
- Transmission completion

13.5Register

DMAC base address: 0x4002 1000

Register	Offset address	Control channel	Description
DMAC_CONF	0x00	ALL	All Channel Configuration Registers
DMAC_CONFA0	0x10	CH0	Channel 0 Configuration A Register
DMAC_CONFB0	0x14	CH0	Channel 0 Configuration B Register
DMAC_SRCADR0	0x18	CH0	Channel 0 Transfer Source Address Register
DMAC_DSTADR0	0x1C	CH0	Channel 0 Transfer Destination Address Register
DMAC_CONFA1	0x20	CH1	Channel 1 Configuration A Register
DMAC_CONFB1	0x24	CH1	Channel 1 Configuration B Register
DMAC_SRCADR1	0x28	CH1	Channel 1 Transfer Source Address Register
DMAC_DSTADR1	0x2C	CH1	Channel 1 Transfer Destination Address Register

13.5.1 DMAC_CONF

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN	Res		PRIO	HALT				Reserved							
RW	R		RW	RW											

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

Bit	Marking	Functional description
31	EN	DMA controller enable 1: Enable DMA controller 0: disable DMA controller Note: When the DMA controller is disabled, ongoing transfers will be forcibly stopped, causing unpredictable results.
30:29	Reserved	
28	PRIO	DMA channel priority configuration 1: Circular priority mode, CH0 and CH1 take turns to get bus access right 0: Fixed priority mode, CH0 priority is higher than CH1 priority
27:24	HALT	DMA all channel transfer pause control 0000: Resume all channel data transmission xxxx: suspend all channel data transmission
23:0	Reserved	

13.5.2 DMAC_CONFA0、DMAC_CONFA1

Offset address: 0x10 (DMAC_CONFA0)

0x20（DMAC_CONFA1）

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENS	PAS	ST	TRI_SEL							Reserved		BC			
RW	RW	RW	RW									RW			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TC[15:0]															
RW															

Bit	Marking	Functional description
31	ENS	DMA channel enable control 1: Enable the current DMA channel; if CONFBx.MSK is 0, this bit is automatically cleared when the transfer is completed 0: disable the current DMA channel Note: When a DMA channel is disabled, ongoing transfers will be forcibly stopped, resulting in unpredictable results.
30	PAS	DMA channel data transfer pause control 1: Suspend DMA channel data transfer 0: Resume DMA channel data transfer
29	ST	DMA channel software trigger control 1: Trigger DMA channel to start software Block/Burst transfer, this bit is automatically cleared when the transfer is completed 0: Stop DMA channel software Block/Burst transfer
28:22	TRI_SEL	DMA channel trigger source configuration 0x00: software trigger 0x40: SPI0 receiving Buf is not empty 0x41: SPI0 sends Buf empty 0x42: SPI1 receiving Buf is not empty 0x43: SPI1 sends Buf empty 0x44: ADC queue conversion completed 0x45: ADC sequential conversion completed 0x46: LCD timing interrupt 0x47：Reserved 0x48: UART0 receiving Buf is not empty 0x49: UART0 sends Buf empty 0x4A: UART1 receiving Buf is not empty 0x4B: UART1 sends Buf empty 0x4C: LPUART0 receives Buf is not empty 0x4D: LPUART0 sends Buf empty 0x4E: LPUART1 receiving Buf is not empty 0x4F: LPUART1 sends Buf empty 0x50: DAC0 conversion complete 0x51: DAC1 conversion complete 0x52: TIM0 channel A, capture event or compare event occurs 0x53: TIM0 channel B, capture event or compare event occurs 0x54: TIM1 channel A, capture event or compare event occurs 0x55: TIM1 channel B, capture event or compare event occurs 0x56: TIM2 channel A, capture event or compare event occurs 0x57: TIM2 channel B, capture event or compare event occurs 0x58: TIM3 channel A, capture event or compare event occurs 0x59: TIM3 channel B, capture event or compare event occurs 0x5A: TIM4 channel A, capture event or compare event occurs 0x5B: TIM4 channel B, capture event or compare event occurs 0x5C: TIM5 channel A, capture event or compare event occurs 0x5D: TIM5 channel B, capture event or compare event occurs 0x5E: TIM6 channel A, capture event or compare event occurs 0x5F: TIM6 channel B, capture event or compare event occurs 0x60: UART2 receiving Buf is not empty 0x61: UART2 sends Buf empty

		0x62: UART3 receiving Buf is not empty 0x63: UART3 sends Buf empty 0x64: I2S0 left channel sends Buf empty or receives Buf not empty 0x65: I2S0 right channel sends Buf empty or receives Buf not empty 0x66: I2S1 left channel sends Buf empty or receives Buf not empty 0x67: I2S1 right channel sends Buf empty or receives Buf not empty
21:20	Reserved	Reserved
19:16	BC	Configure the size of the data block to be transmitted by the DMA channel as BC+1 Note: When the source address or destination address can only provide or receive a data every once in a while, the BC value can only be set to 0
15:0	TC	Configure the number of data blocks to be transmitted by the DMA channel as TC+1 During transmission, every time a data block is transmitted, the value of this register will be decremented by 1 Note: The total amount of data transferred by DMA is (TC + 1) * (BC + 1)

13.5.3 DMAC_CONFB0、DMAC_CONFB1

Offset address: 0x14 (DMAC_CONFB0)

0x24 (DMAC_CONFB1)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		MODE		WIDTH		FS	FD	RC	RS	RD	ERR_IE	FIS_IE	STAT		
		RW		RW		RW	RW	RW	RW	RW	RW	RW	RW		

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

Bit	Marking	Functional description
31:30	Reserved	
29:28	MODE	DMA channel transfer mode configuration 00: Block transmission 01: Burst transmission 10：Reserved 11：Reserved
27:26	WIDTH	DMA channel transfer data width configuration 00：8bit 01：16bit 10：32bit 11：Reserved
25	FS	DMA channel source address auto-increment enable 1: The source address remains unchanged during transmission 0: The source address will be incremented after each data transmission is completed; The auto-increment corresponding to the data width of 8bit / 16bit / 32bit is 1 / 2 / 4 respectively
24	FD	DMA channel target address self-increment enable 1: Destination address remains unchanged during transfer 0: The destination address will be incremented after each data transmission is completed; The auto-increment corresponding to the data width of 8bit / 16bit / 32bit is 1 / 2 / 4 respectively
23	RC	DMA channel BC and TC reload enable 1: Enable BC/TC reload, and the value of BC/TC will automatically restore to the initial value written after the transmission is completed 0: Disable BC/TC reloading, and the values of BC/TC are all 0 after the transmission is completed
22	RS	DMA channel source address reload enable 1: Enable the source address to be reloaded, and the value of the source address is the initial value written after the transmission is completed 0: Disable source address overloading, the value of the source address is uncertain after the transmission is completed
21	RD	DMA channel target address reload enable 1: Enable target address reloading, the value of the target address will automatically restore to the initial value written after the transfer is completed 0: Disable target address overloading, the value of the target address is uncertain after the transmission is completed
20	ERR_IE	DMA channel transfer error interrupt enable 1: When a transmission error occurs, an interrupt is generated 0: disable interrupt when transmission error Note: In the interrupt service routine, you need to write 0x00 to STAT to clear the interrupt status
19	FIS_IE	DMA channel transfer complete interrupt enable 1: Generate an interrupt when the transfer is complete 0: When the transfer is complete, disable the interrupt Note: In the interrupt service routine, you need to write 0x00 to STAT to clear the interrupt status
18:16	STAT	DMA channel current transfer status 000: initial value 001: Transmission error, the transmission address exceeds the addressing range 010: Transmission error, peripheral request to stop DMA 011: Transmission error, error in accessing transmission source address

		100: transmission error, access transmission destination address error 101: Transfer complete 110: reserved 111: Transmission paused
15:1	Reserved	
0	MSK	Auto-disable configuration when DMA channel transfer completes 1: CONFAx.ENS remains unchanged when DMA channel transfer is complete 0: CONFAx.ENS is automatically cleared when the DMA channel transfer is complete

13.5.4 DMAC_SRCADR0、DMAC_SRCADR1

Offset address: 0x18 (DMAC_SRCADR0)

0x28（DMAC_SRCADR1）

Reset value: 0x0000 0000

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

Bit	Marking	Functional description
31:0	SRCADR	DMA channel source address configuration Its auto-reload can be configured via CONFBx.RS Note: The address alignment should match the data width, otherwise it will cause address access errors

13.5.5 DMAC_DSTADR0、DMAC_DSTADR1

Offset address: 0x1C (DMAC_DSTADR0)

0x2C（DMAC_DSTADR1）

Reset value: 0x0000 0000

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

Bit	Symbol	Description
31:0	DSTADR	DMA channel target address configuration Its auto-reload can be configured via CONFBx.RD Note: The address alignment should match the data width, otherwise it will cause address access errors

14 General purpose timer (TIM0/1/2/3)

14.1 Introduction to General-Purpose Timers

TIM0/1/2 is a timer composed of 1 counting unit and 2 comparing units respectively. Each timer supports 2 independent PWM outputs or 1 pair of complementary PWM outputs and 1 independent PWM output. Supports 2 capture inputs. TIM0/1/2 can form 3 pairs of complementary PWM output.

TIM3 is a timer composed of 1 counting unit and 6 comparison units, and supports 6 independent PWM outputs or 3 pairs of complementary PWM outputs. Supports 6 capture inputs.

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

14.1.1 Basic Features (TIM0/1/2)

- 2 independent PWM outputs CHA, CHB,
- 1 channel complementary PWM output (CHA, CHB) + 1 channel independent PWM output (gate)
- 1 channel complementary PWM output (CHA, CHB) + 1 channel capture function (gate)
- Up to 2 capture inputs
- Pulse Width Measurement
- Dead zone control
- Brake control
- Edge alignment, symmetric center alignment and asymmetric center alignment PWM output
- Quadrature code counting function
- Single pulse mode
- External counting function
- DMA trigger
- When the system clock selects PLL, the timer clock can be configured as twice the frequency of the system clock

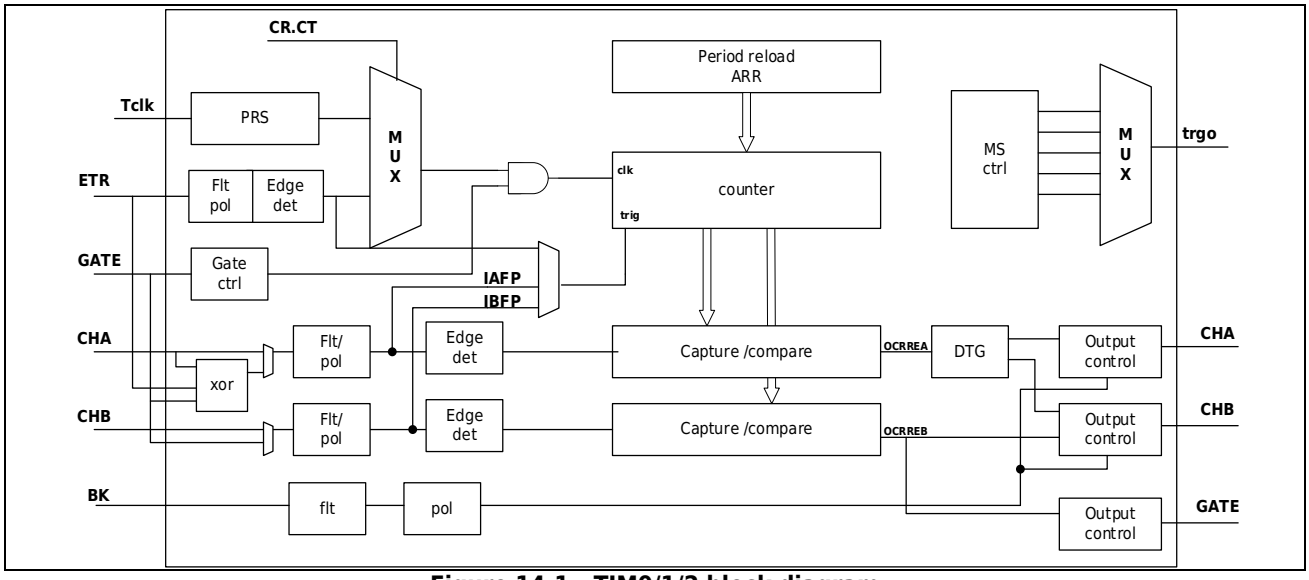


Figure 14-1 TIM0/1/2 block diagram

14.1.2 Basic Features (TIM3)

- 6 independent PWM outputs CH0A, CH0B, CH1A, CH1B, CH2A, CH2B
- 3 complementary PWM outputs (CHxA, CHxB) + 1 independent PWM output (gate)
- 3- way complementary PWM output (CHxA, CHxB) + 1 -way capture function (gate)
- Up to 6 capture inputs
- Pulse Width Measurement
- Dead zone control
- Brake control
- Edge alignment, symmetric center alignment and asymmetric center alignment PWM output
- Quadrature code counting function
- Single pulse mode
- External counting function
- DMA trigger
- When the system clock selects PLL, the timer clock can be configured as twice the frequency of the system clock

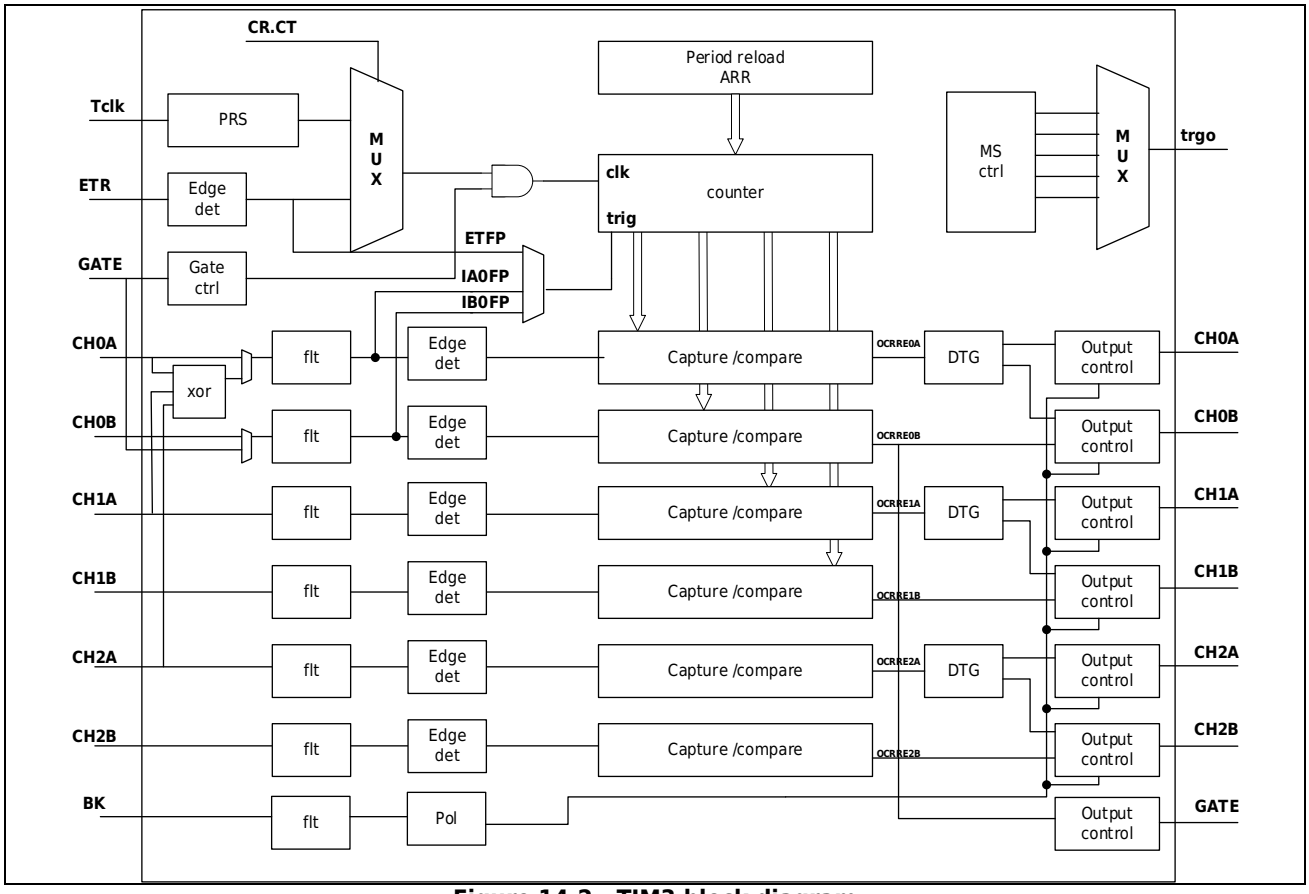


Figure 14-2 TIM3 block diagram

14.2Timer function description

14.2.1 Timer clock

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

14.2.2 Timer counter

The main building block of the programmable general-purpose timer is a 16 -bit counter and its associated auto-reload register. The counter can count up, count down, or alternately count up and down. The counter clock can be divided by a prescaler.

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

14.2.3 Timer Prescaler

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

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

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

14.2.4 Mode 0 count timer function

In this mode, the counter counts up. The counter supports two counting modes, reload mode and free counting mode, and you can choose external clock ETR counting or system clock counting. The gate control gate can control the count mask. Counting to the maximum overflow generates an interrupt. Invert the output port CHA, CHB, in this mode, CHA, CHB are reversed.

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

The counter can be used to control whether the counter counts through the external gate control function. The control relationship is as follows:

M0CR.GATE	M0CR.GATEP	Port GATE input	M0CR.CTEN	Counter
x	x	x	0	Not count
0	x	x	1	Count
1	0	High level	1	Count
1	0	Low level	1	Not count
1	1	High level	1	Not count
1	1	Low level	1	Count

14.2.4.1 Functional block diagram

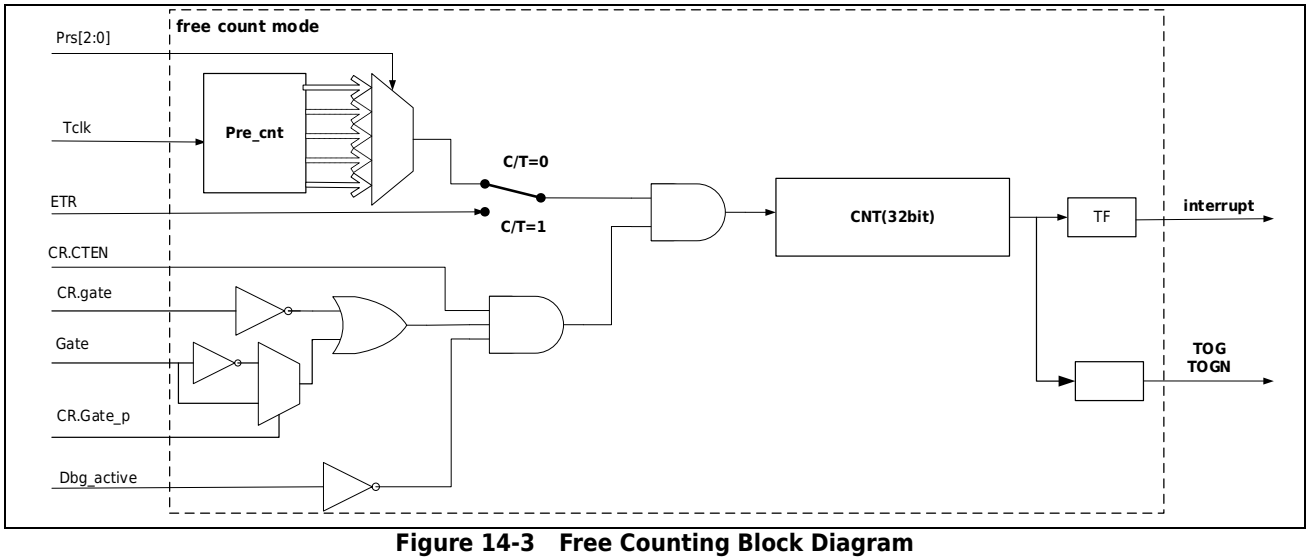


Figure 14-3 Free Counting Block Diagram

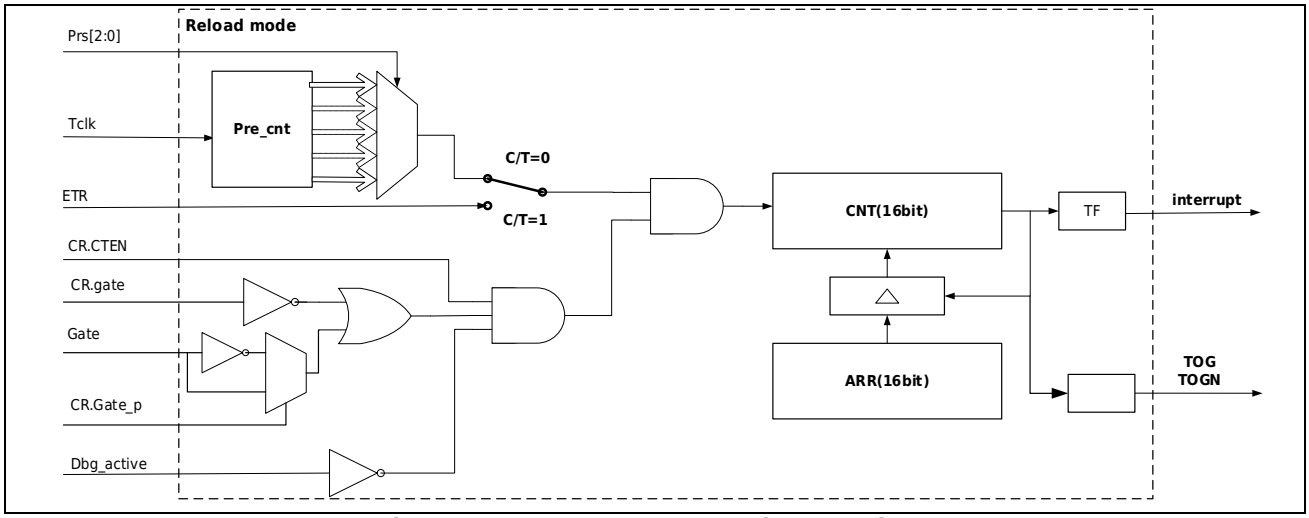


Figure 14-4 Heavy Duty Counting Waveform

14.2.4.2 Count waveform

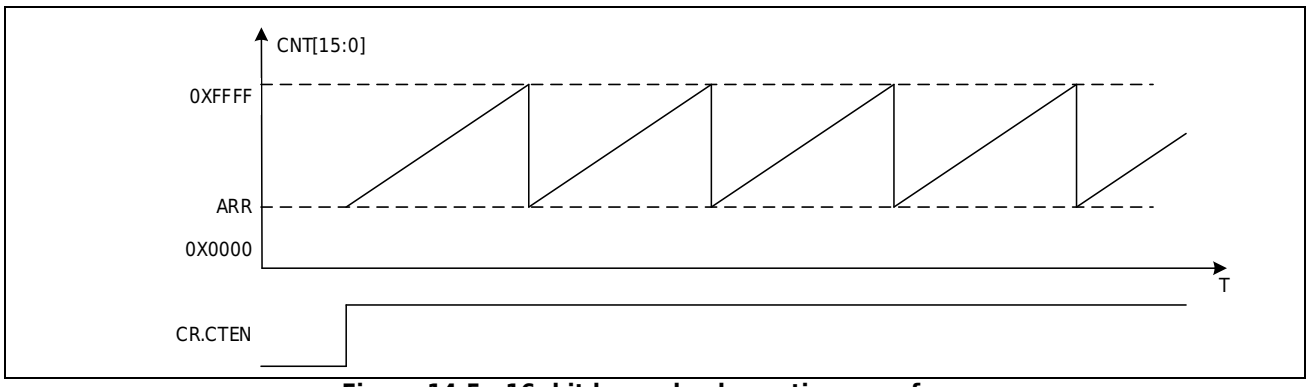


Figure 14-5 16 -bit heavy load counting waveform

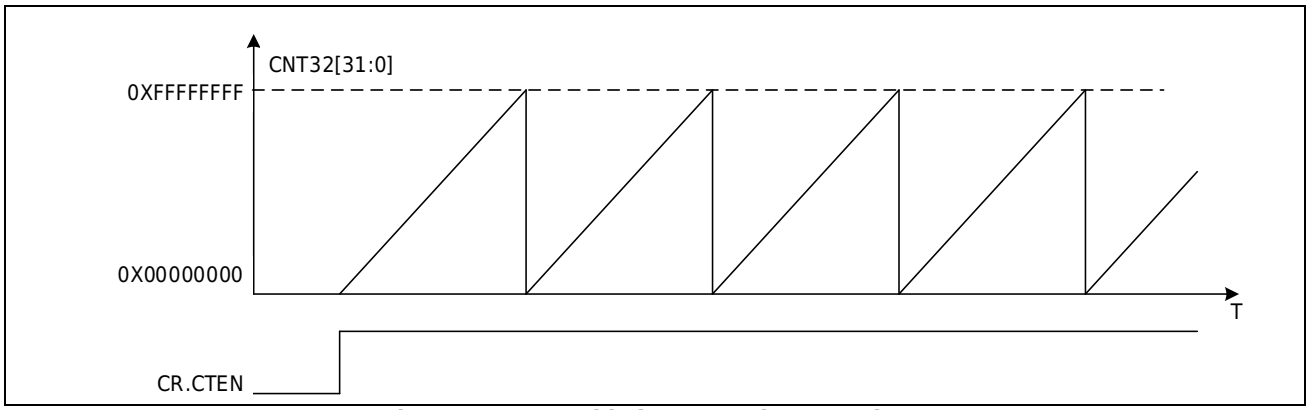


Figure 14-6 32 -bit free-counting waveform

14.2.4.3 Counting function

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

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

14.2.4.4 Timing function

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

14.2.4.5 Timing diagram

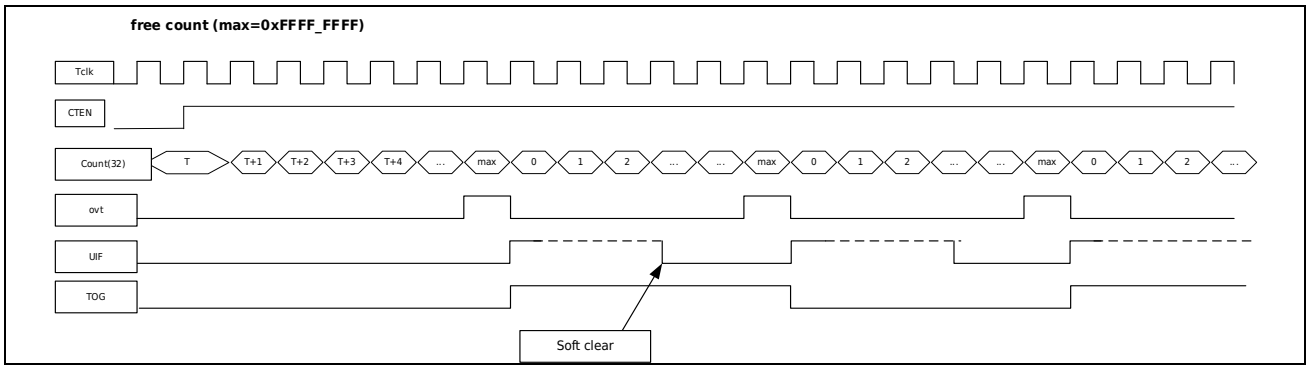


Figure 14-7 Free Count Timing Diagram

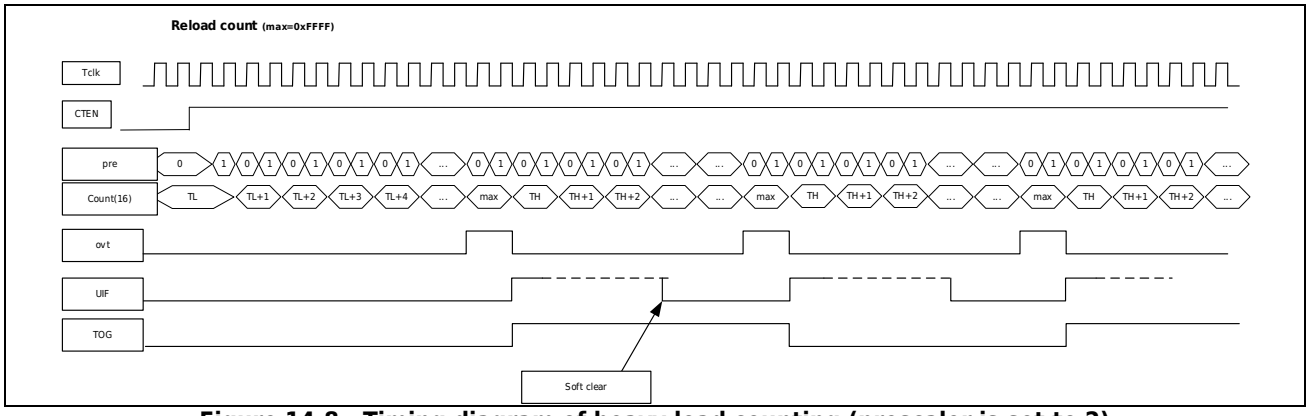


Figure 14-8 Timing diagram of heavy load counting (prescaler is set to 2)

14.2.4.6 Buzzer function

The Buzzer can be realized through the flipping output function of the timer. To use the toggle output requires enabling the DTR.MOE control bit. Setting CR.TOG_EN to 0 can set port CHA, CHB output to 0 at the same time. When the counting clock is 4M, the timer reload mode configuration of Buzzer outputting different frequencies is as follows:

Buzzer frequency	Counter period	Counter count value	Counter reload value	CNTL initial value	CNTH reload value
1000Hz	0.5ms	2000	63536	0XF830	0XF830
2000Hz	0.25ms	1000	64536	0XFC18	0XFC18
4000Hz	0.125ms	500	65036	0XFE0C	0XFE0C

14.2.4.7 Setup example

Reload Timer Settings

1. Set timer mode M0CR.MODE=0
2. Set load value ARR
3. Set the counter initial value CNT
4. Clear interrupt flag
5. Enable interrupt M0CR.UIE
6. Enable reload mode M0CR.MD
7. Start timer M0CR.CTEN

Gated external clock free count setting

1. Set timer mode M0CR.MODE=0
2. Set the counter initial value CNT
3. Enable gate function M0CR.GATE
4. Select gate active level M0CR.GATEP
5. Clear interrupt flag
6. Enable interrupt M0CR.UIE
7. Enable external clock mode M0CR.CT
8. Start timer M0CR.CTEN

BUZZER output control

1. Set the appropriate ARR value according to the output frequency
2. Set the timer to reload mode, refer to reload timer setting
3. Enable Output Enable DTR.MOE
4. Start another timer to control M0CR.TOGEN to realize frequency interval output.

14.2.5 Mode 1 pulse width measurement PWC

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

The first valid edge counter is initialized to 0x0001, the second valid edge will stop counting, and the current count value will be stored in CMAR, and a capture interrupt CAF will be generated. If the counter overflows, an overflow flag will be generated. Setting overflow interrupt enable will generate overflow interrupt.

M1CR.edg1st	0	0	1	1
M1CR.edg2nd	0	1	0	1
Pulse width measurement	Upper edge ~ Upper edge Cycle width	Upper edge ~ lower edge High level width	Lower edge ~ upper edge Low level width	Lower edge ~ lower edge Cycle width

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

14.2.5.1 PWC functional block diagram

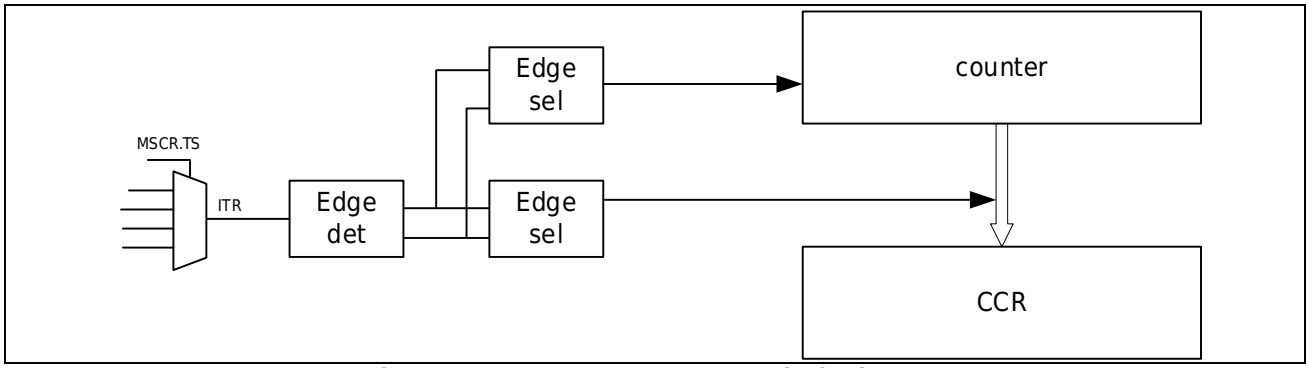


Figure 14-9 PWC Measurement Block Diagram

MSCR.TS	Trigger selection 000: the signal ETFP after the filter phase selection of the port ETR; 001: Internal interconnection signal ITR0 010: internal interconnection signal ITR1; 011: internal interconnection signal ITR2; 100: internal interconnection signal ITR3; 101:Invalid 110: Filtered signal IAFP of port CH0A (polarity selection is invalid in pulse width measurement mode) 111: Filtered signal IBFP of port CH0B (polarity selection is invalid in pulse width measurement mode)

14.2.5.2 PWC waveform measurement timing diagram

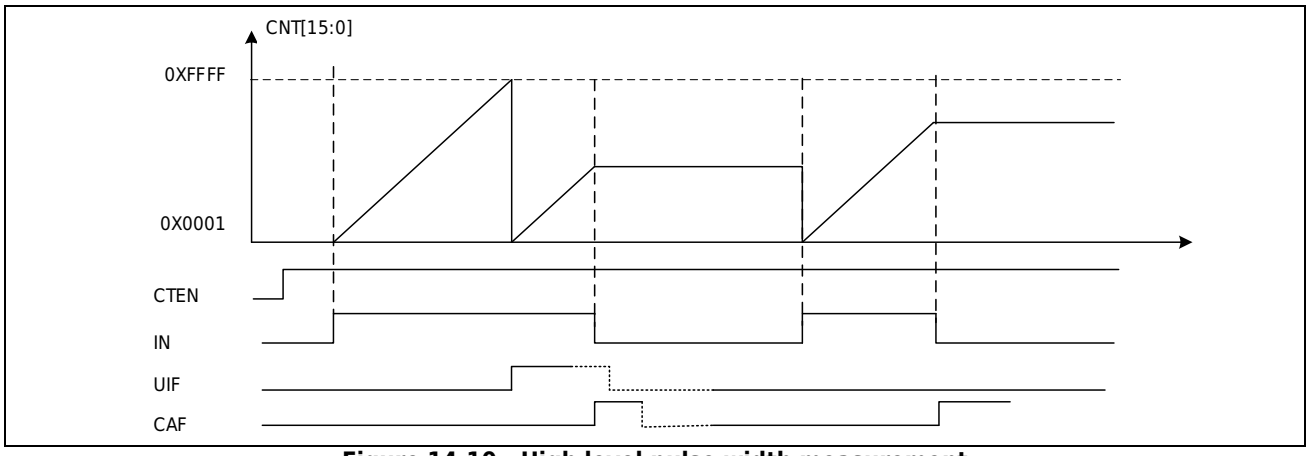


Figure 14-10 High level pulse width measurement

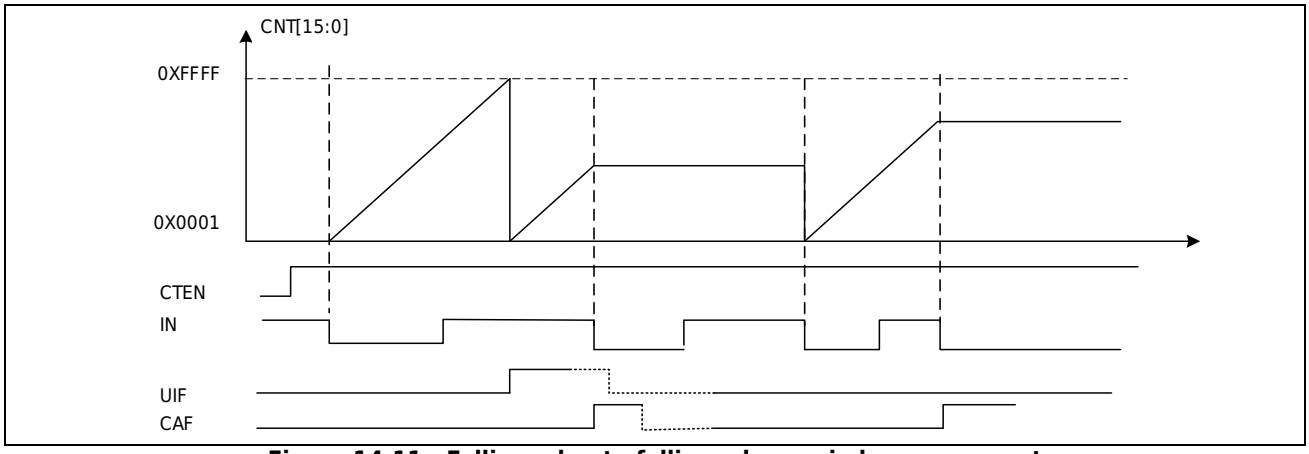


Figure 14-11 Falling edge to falling edge period measurement

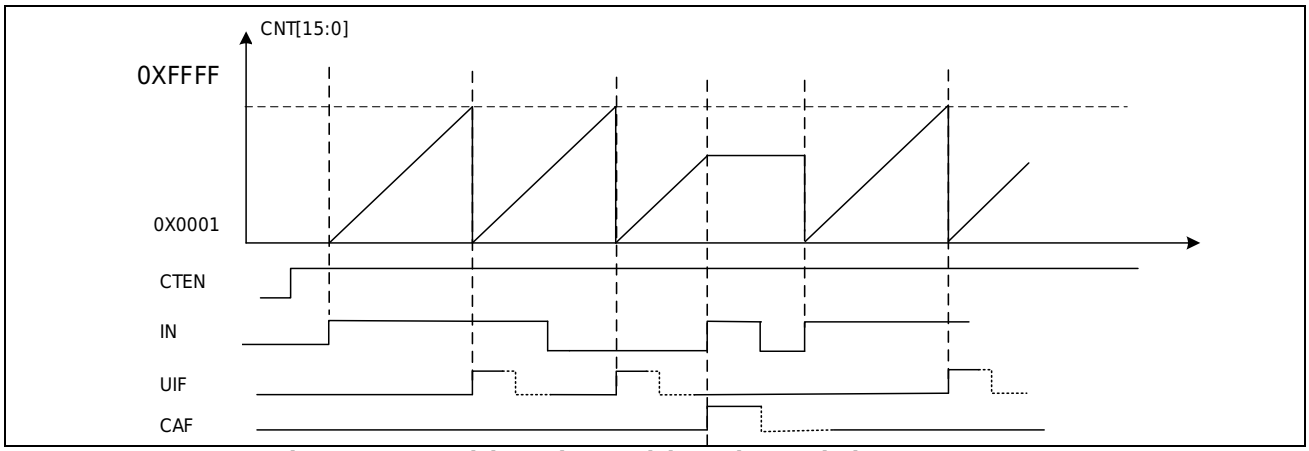


Figure 14-12 Rising edge to rising edge period measurement

Select the measurement signal source by registering MSCR.TS.

000 ETFP: ETR external input and input filtered phase selection signal, external filter and input reverse can be selected

001 ITR0: Timer internal interconnection signal 0, TRGO output of other timers

010 ITR1: Timer internal interconnection signal 1, TRGO output of other timers

011 ITR2: Timer internal interconnection signal 2, TRGO output of other timers

100 ITR3: Timer internal interconnection signal 3, TRGO output of other timers

101 IA0ED: Invalid

110 IAFP: CH0A external input and input filtered phase selection signal, external filtering and input reverse can be selected

111 IBFP: CH0B external input and input filtered phase selection signal, external filtering and input reverse can be selected

	ITR0	ITR1	ITR2	ITR3
Timer0	-	TIM1_TRGO	TIM2_TRGO	TIM3_TRGO
Timer1	TIM0_TRGO	-	TIM2_TRGO	TIM3_TRGO
Timer2	TIM0_TRGO	TIM1_TRGO	-	TIM3_TRGO
Timer3	TIM0_TRGO	TIM1_TRGO	TIM2_TRGO	-

Note: Refer to register description for TRGO output.

14.2.5.3 PWC one-shot mode

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

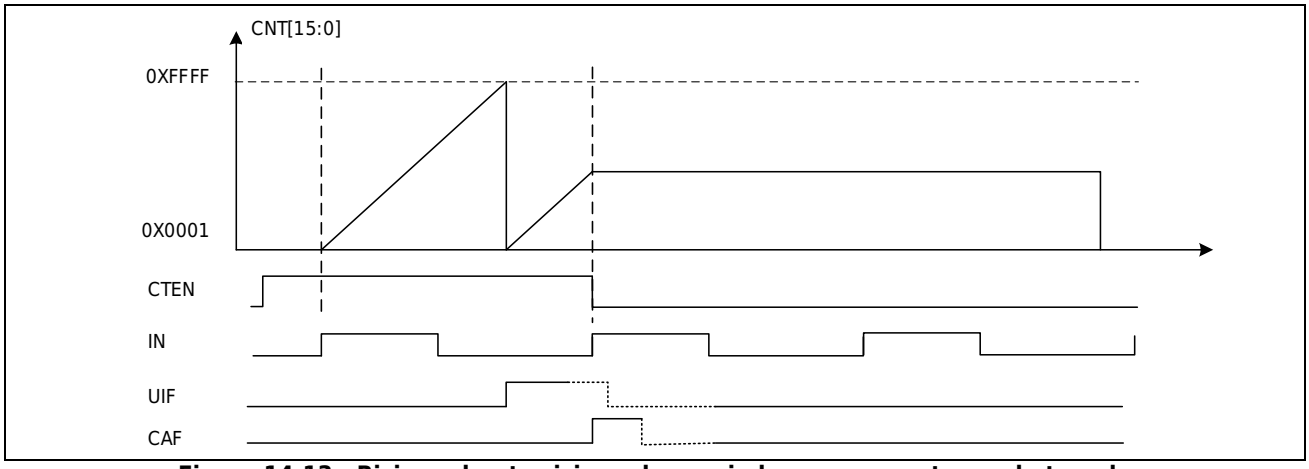


Figure 14-13 Rising edge-to-rising edge period measurement one-shot mode

14.2.5.4 Setup example

Pulse Low Level Measurement Setup

1. Set to pulse measurement mode M1CR.MODE=1
2. Set MSCR.TS to select the signal to measure
3. Set M1CR.edg2dn=0, M1CR.edg1st=1 to choose to measure low level
4. Clear interrupt flag
5. Enable overflow interrupt M1CR.UIE
6. Enable measurement end interrupt CR0.CIEA
7. Enable timer M1CR.CTEN
8. Read CCR0A and the number of overflows in the interrupt service routine and clear the interrupt flag
9. Waiting for next measurement

Pulse high level single measurement setup

1. Set to pulse measurement mode M1CR.MODE=1
2. Set MSCR.TS to select the signal to measure
3. Set pulse single measurement mode M1CR.ONESHOT=1
4. Set M1CR.edg2dn=1, M1CR.edg1st=0 to choose to measure low level

5. Clear interrupt flag
6. Enable overflow interrupt M1CR.UIE
7. Enable measurement end interrupt CR0.CIEA
8. Enable timer M1CR.CTEN
9. Read CCR0A and the number of overflows in the interrupt service routine and clear the interrupt flag
10. End of measurement

14.2.6 Mode 2/3 compare capture mode

14.2.6.1 Counter

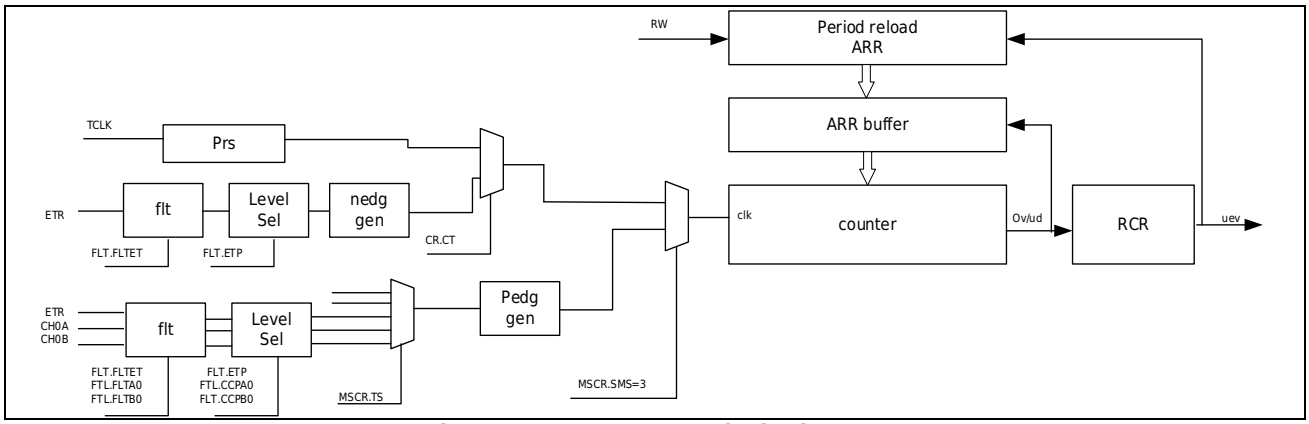


Figure 14-14 Counter Block Diagram

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

- Counter basic units include:
- Counter Register CNT
- Prescaler register CR.PRS
- Autoload Register ARR
- Repeat count register RCR
- Clock selection control registers FLT, CR0, MSCR, CR

The autoload register has a cache function, and the reload value is updated from the cache register to the counter after the counter generates an event update. When the counter is stopped or the cache function is disabled, the autoload register is immediately updated to the cache register. When the timer is running and the parallel cache function is valid, the value written to the autoload register will not be updated to the cache register immediately, and the autoload register will be updated to the cache register only after the event is updated.

Clock selection and gating function, trigger function, reset function refer to the chapter of mode 2/3 slave mode.

14.2.6.2 counter waveform

Mode 2 is a sawtooth counting waveform, and the counting direction can be changed by setting CR.DIR.

CR.DIR is set to 0, the counter is in up-counting mode. In this mode, the counter counts from 0 to the auto-reload value (TIMx_ARR), and then restarts Counts from 0 and generates a counter overflow event. (UEV) is generated when the number of repetitions counted up reaches the number programmed in the repeat counter register plus one (TIMx_RCR+1). Otherwise, an update event will be generated every time the counter overflows.

An update event is also generated when the UG bit of the TIMx_CR register is set (by software or using a slave mode controller).

When an update event occurs, all registers are updated and the update flag (UIF bit in the TIMx_IFR register) is set (depending on the URS bit):

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

The figure below shows the counter waveforms of different counting directions when ARR=0X2C

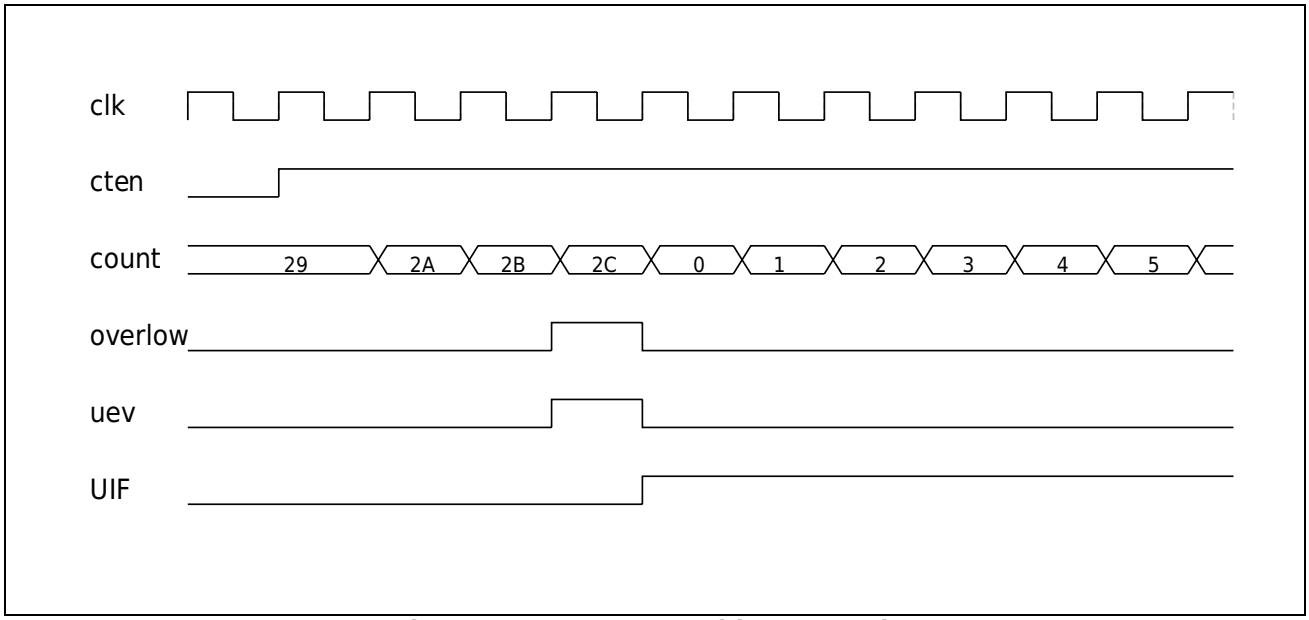


Figure 14-15 Count Up Without Prescaler

The number of counting overflow cycle clocks is ARR+1,

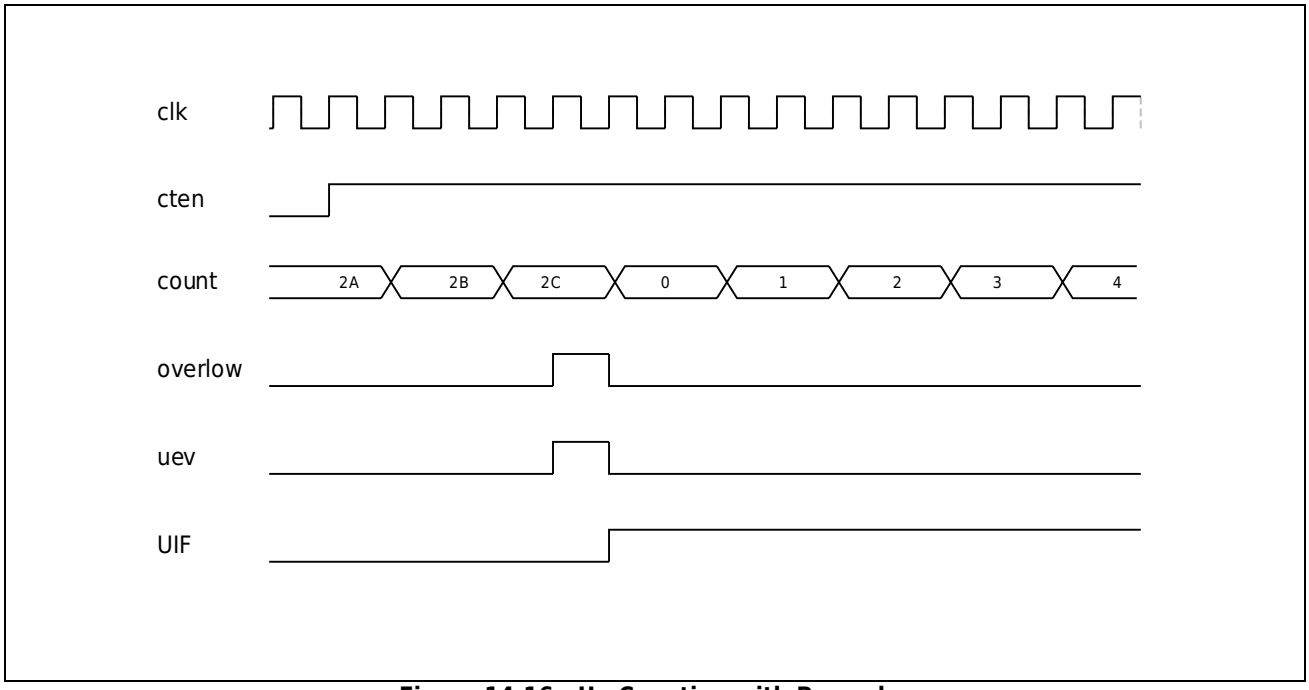


Figure 14-16 Up Counting with Prescaler

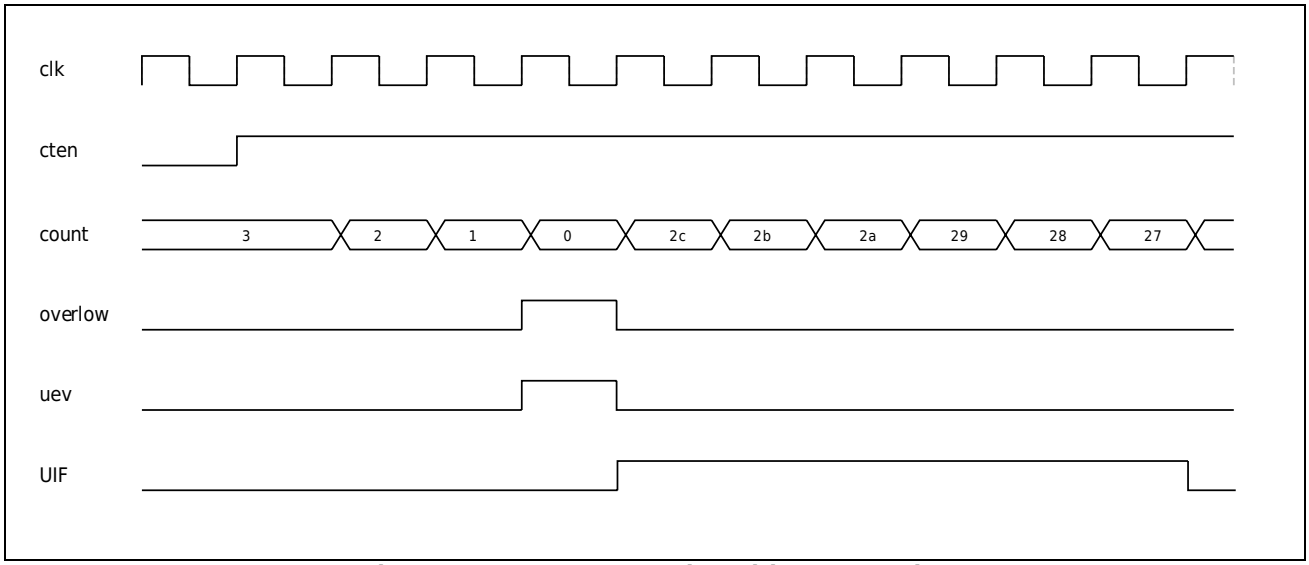


Figure 14-17 Down counting without prescaler

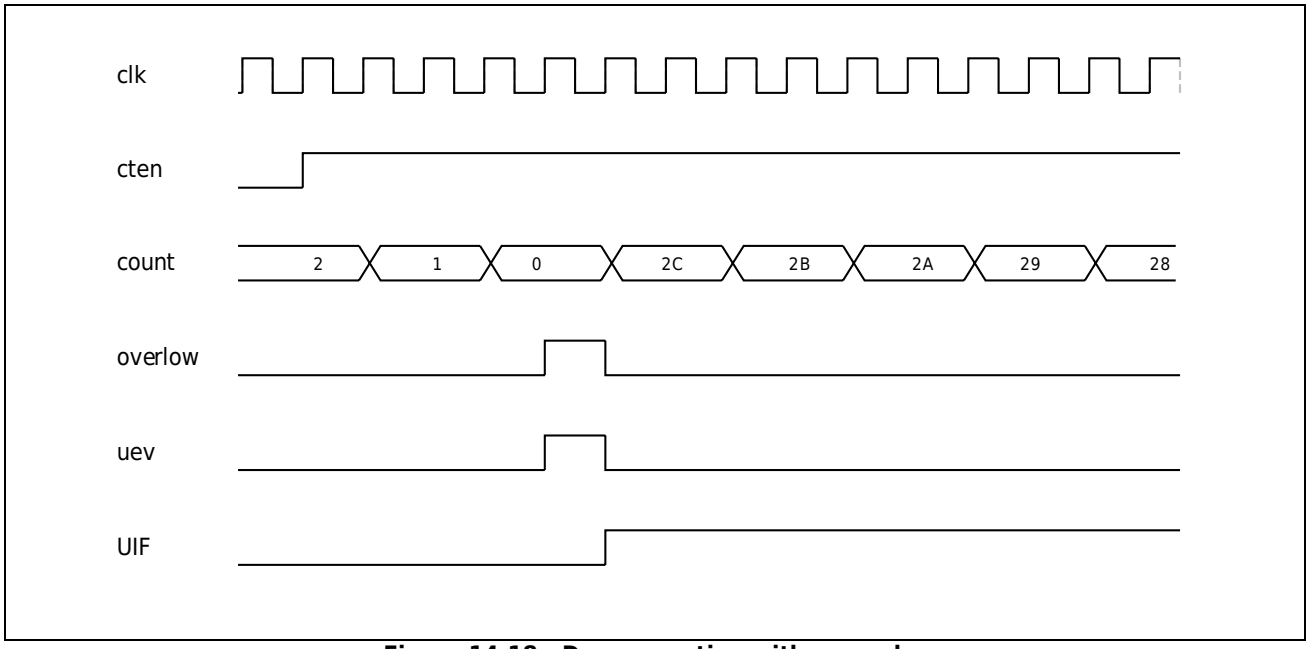


Figure 14-18 Down counting with prescaler

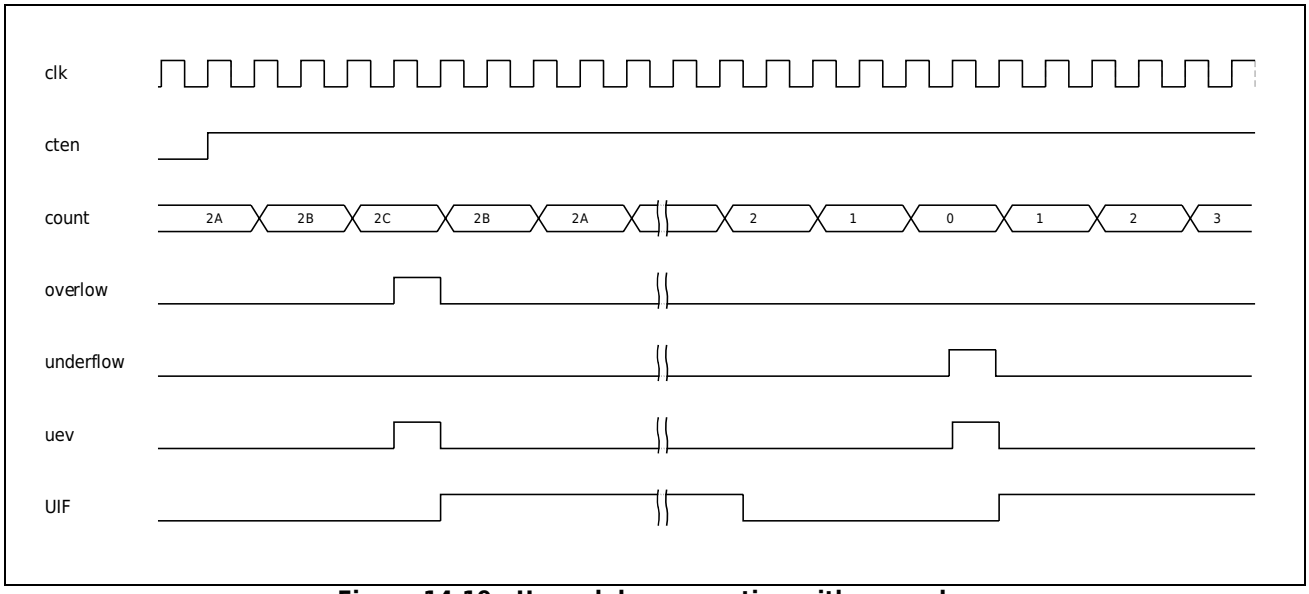


Figure 14-19 Up and down counting with prescaler

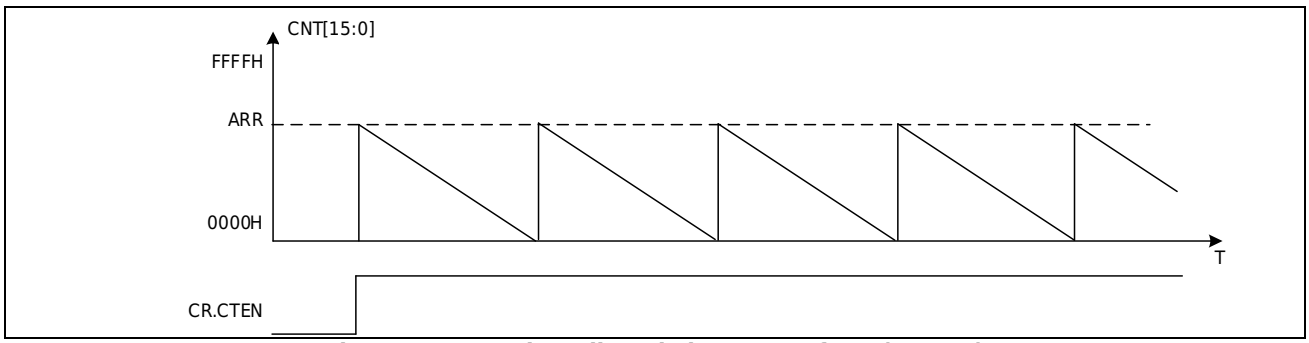


Figure 14-20 Edge-Aligned Timer Waveform (DIR =1)

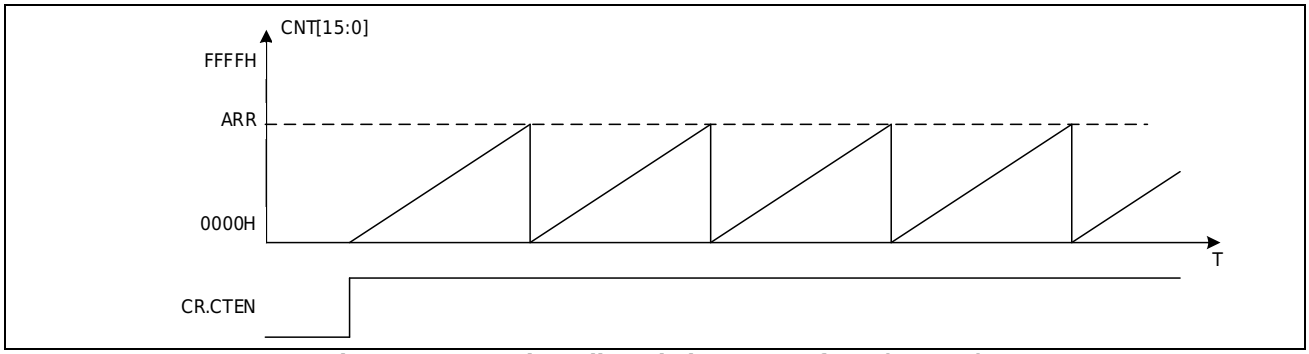


Figure 14-21 Edge-Aligned Timer Waveform (DIR =0)

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

The CR.DIR direction bit is read-only in center-aligned (triangular wave) mode. Write value is invalid. Switching from other modes to center-aligned mode DIR is automatically cleared to 0. DIR is automatically cleared in reset mode by software event update and external trigger in slave mode.

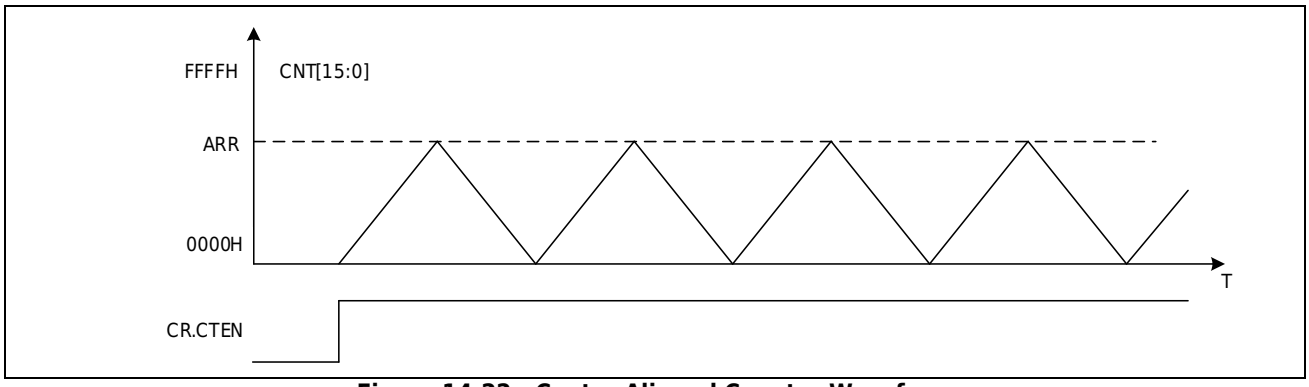


Figure 14-22 Center Aligned Counter Waveform

14.2.6.3 Repeat count

A repeating counter uses the counter's overflow to count down. counting to 0, that is, when the counter overflows once the value set by the repeat register is added. When the cache register is enabled, the cycle reload register is updated to the cycle cache register. In compare mode, the value of the compare register is updated to the compare cache register.

The repeat counter is decremented when the following conditions are true

- Each time the counter overflows in up counting mode
- Each time the counter underflows in down counting mode
- Every overflow and every underflow in triangle wave mode

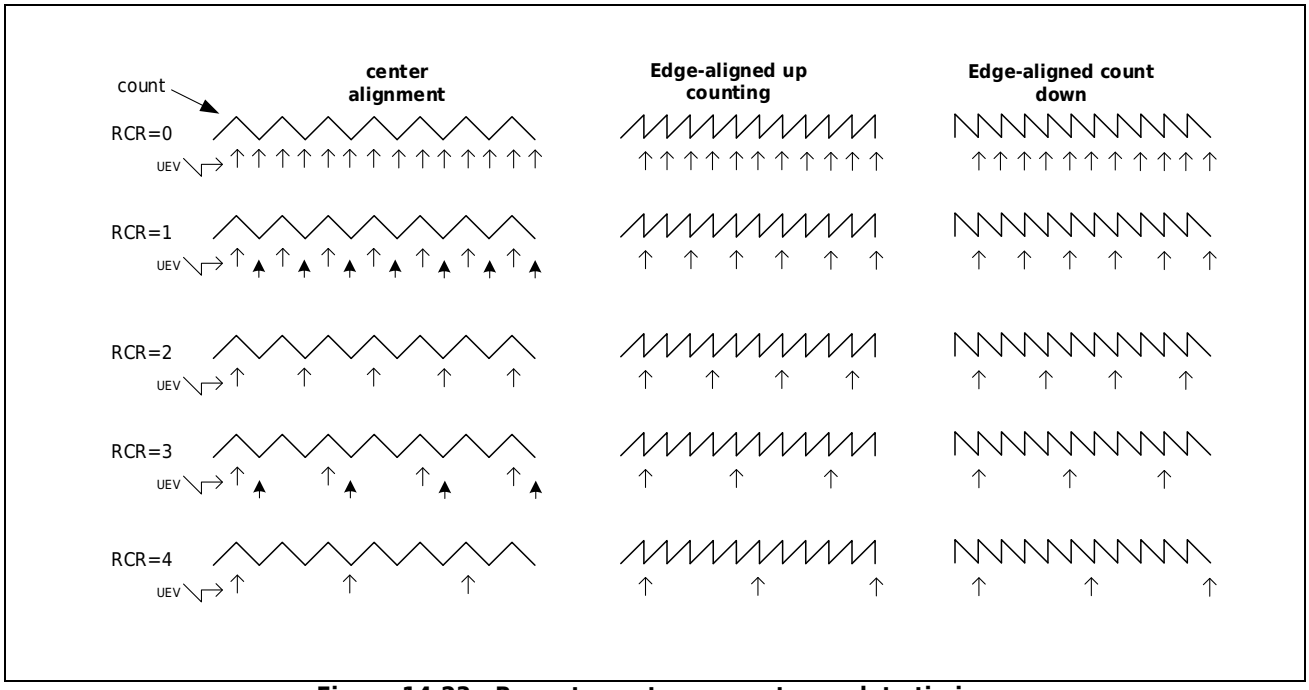


Figure 14-23 Repeat counter generates update timing

In addition to generating events to update UEV through repeated counters on overflow and overflow, you can also generate software and slave mode reset events to update UEV by writing register CR.UG; at this time, you need to configure CR.URS.

14.2.6.4 data cache

The auto-reload data ARR and the comparison register can be configured with a cache function. When the cache function is valid, the written period value ARR and comparison value CCR will only take effect when the UEV event is updated.

The update timing diagram of the auto-reload value in different counting modes is as follows:

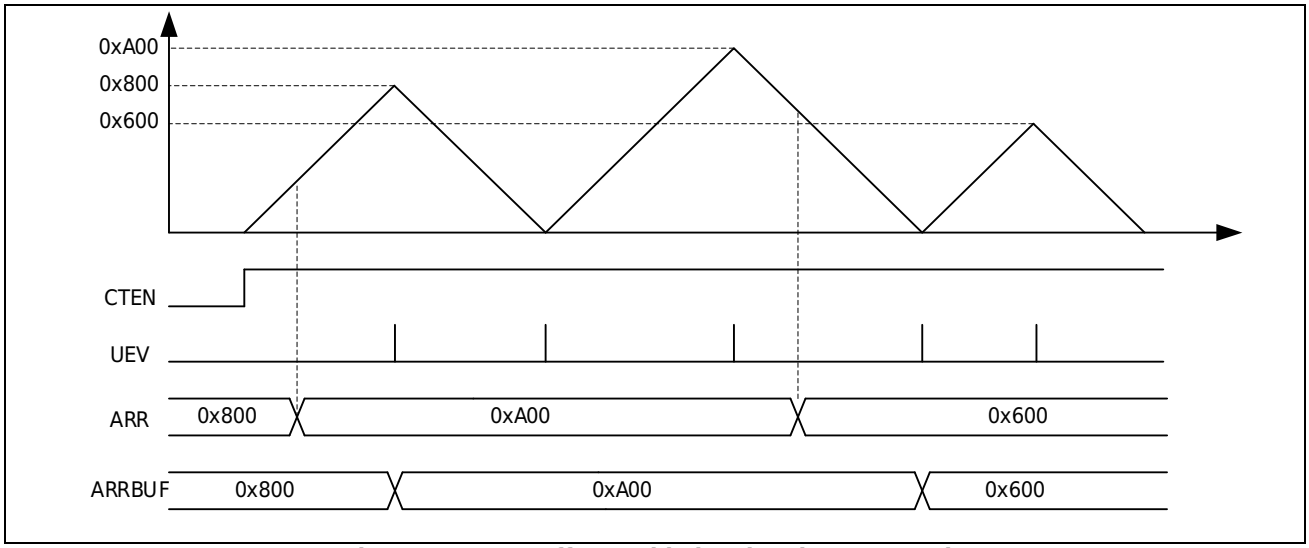


Figure 14-24 Buffer enable in triangle wave mode

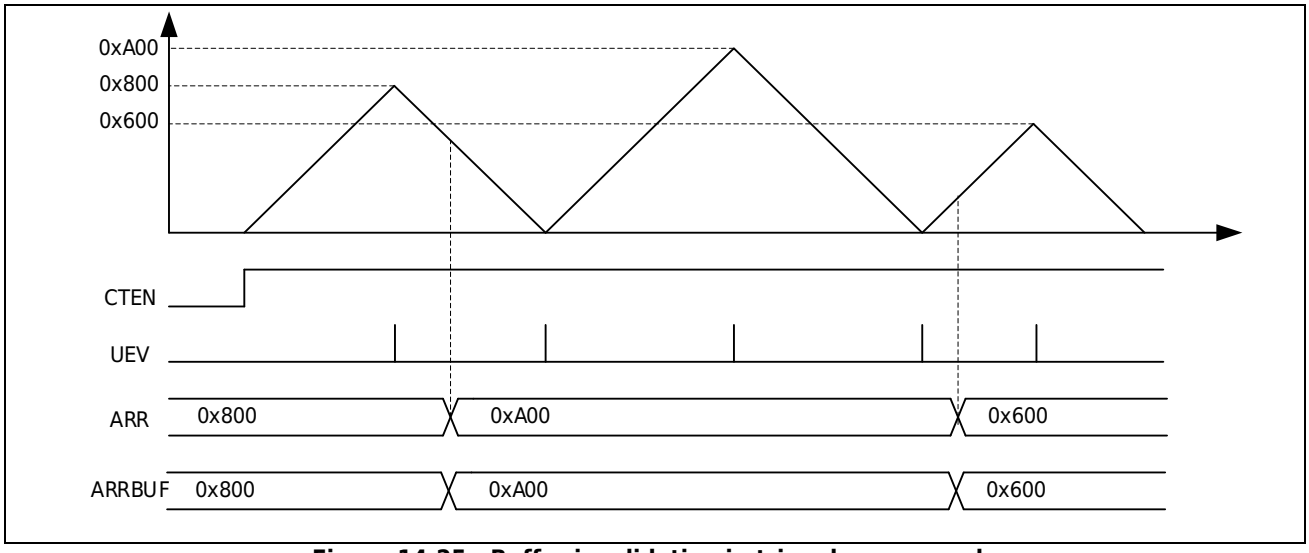


Figure 14-25 Buffer invalidation in triangle wave mode

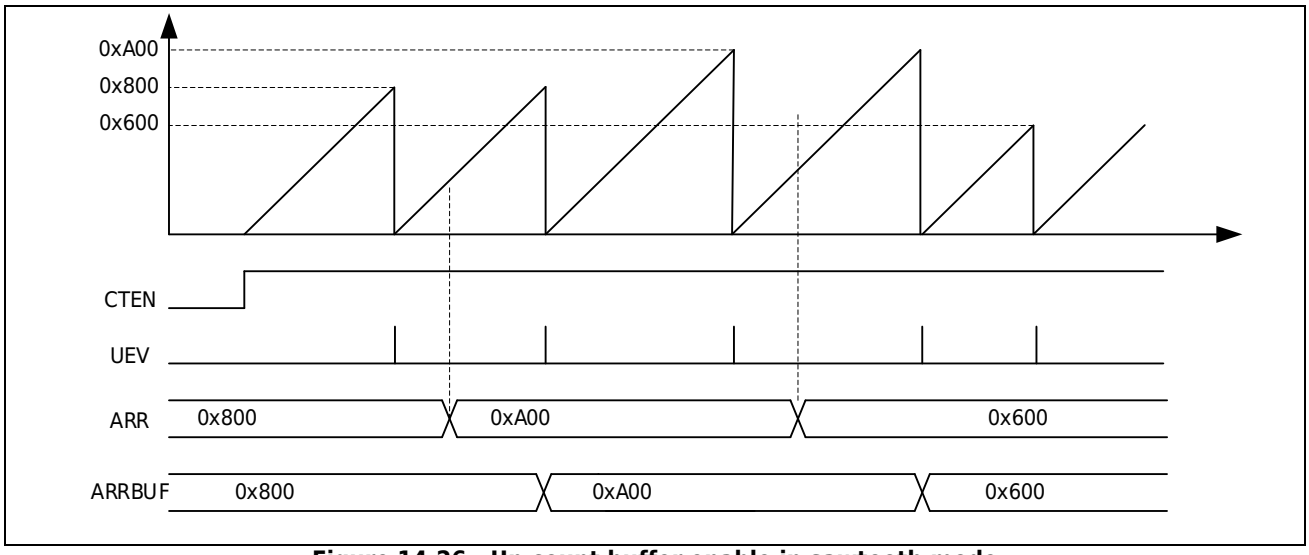


Figure 14-26 Up count buffer enable in sawtooth mode

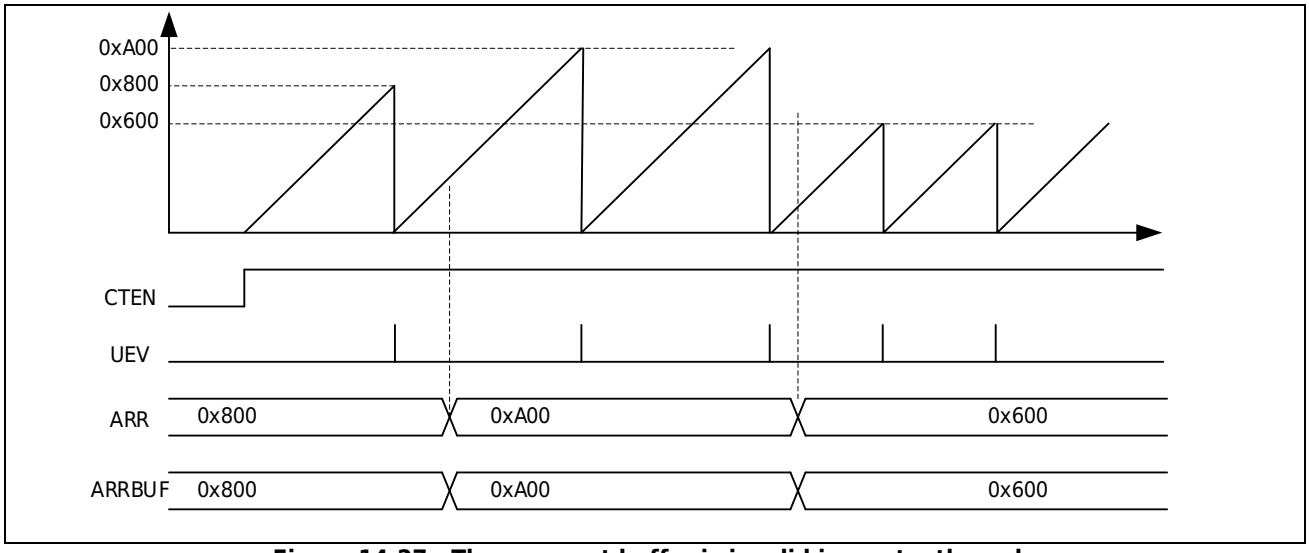


Figure 14-27 The up count buffer is invalid in sawtooth mode

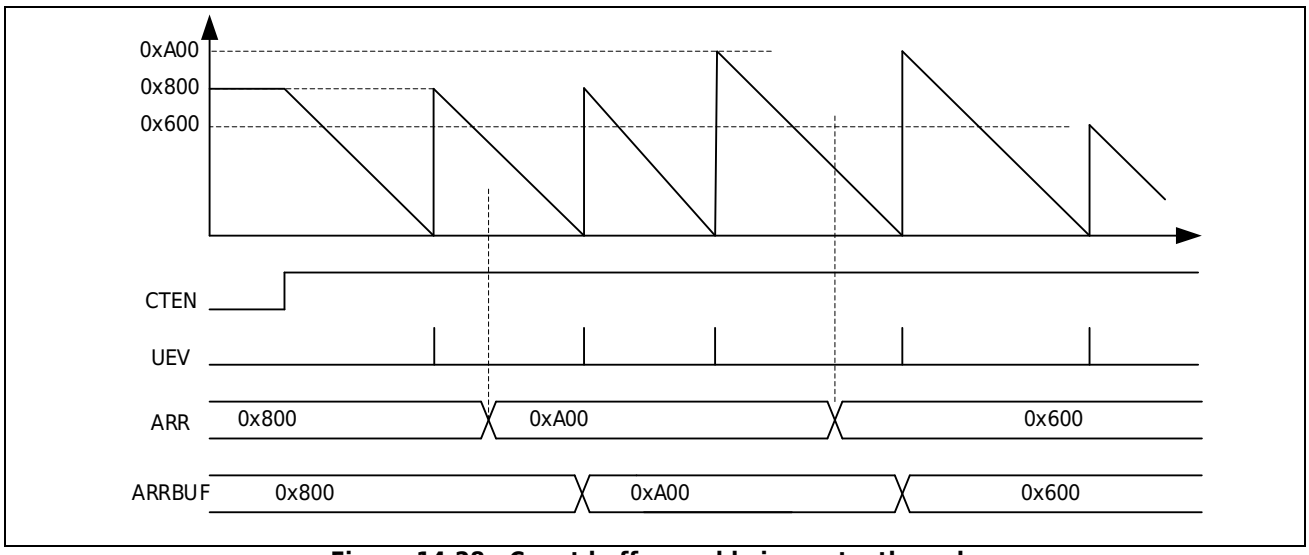


Figure 14-28 Count buffer enable in sawtooth mode

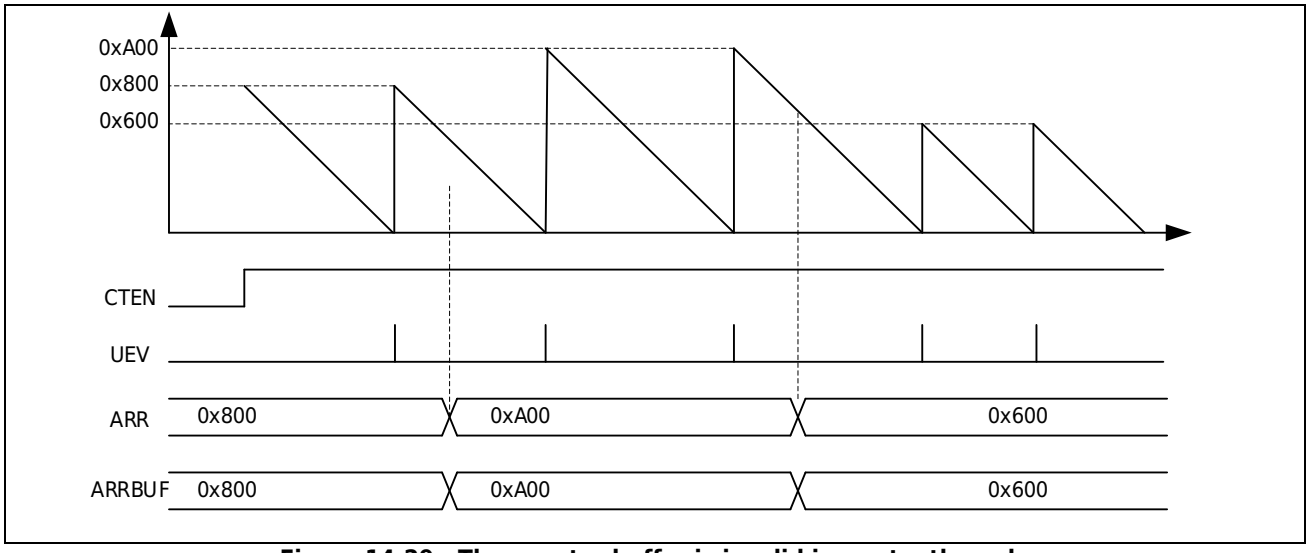


Figure 14-29 The counter buffer is invalid in sawtooth mode

In triangular wave mode and sawtooth counting mode, if the cache is not enabled, when changing ARR, the current counter value should be smaller than the ARR cycle value to be changed, otherwise the current cycle will count to 0XFFFF.

The update status of the comparison cache is consistent with that of the periodic cache, and the sequence diagrams are not listed here.

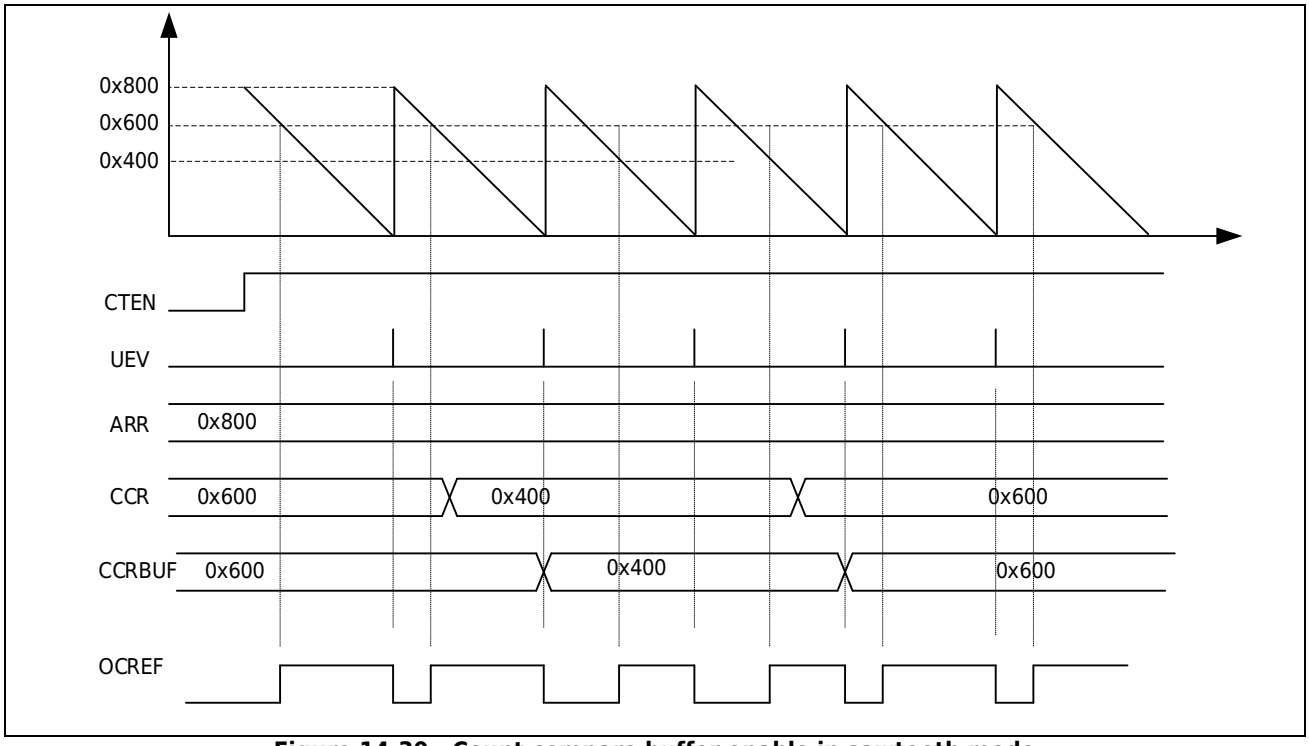


Figure 14-30 Count compare buffer enable in sawtooth mode

14.2.6.5 Compare output OCREF

The comparison output OCREFA can be configured as a single-point comparison, and the comparison register CCRA is used to control the output of OCREFA; the comparison output of OCREFA can also be configured as a two-point comparison, and the comparison registers CCRA and CCRB are used to control the comparison output of OCREFA.

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

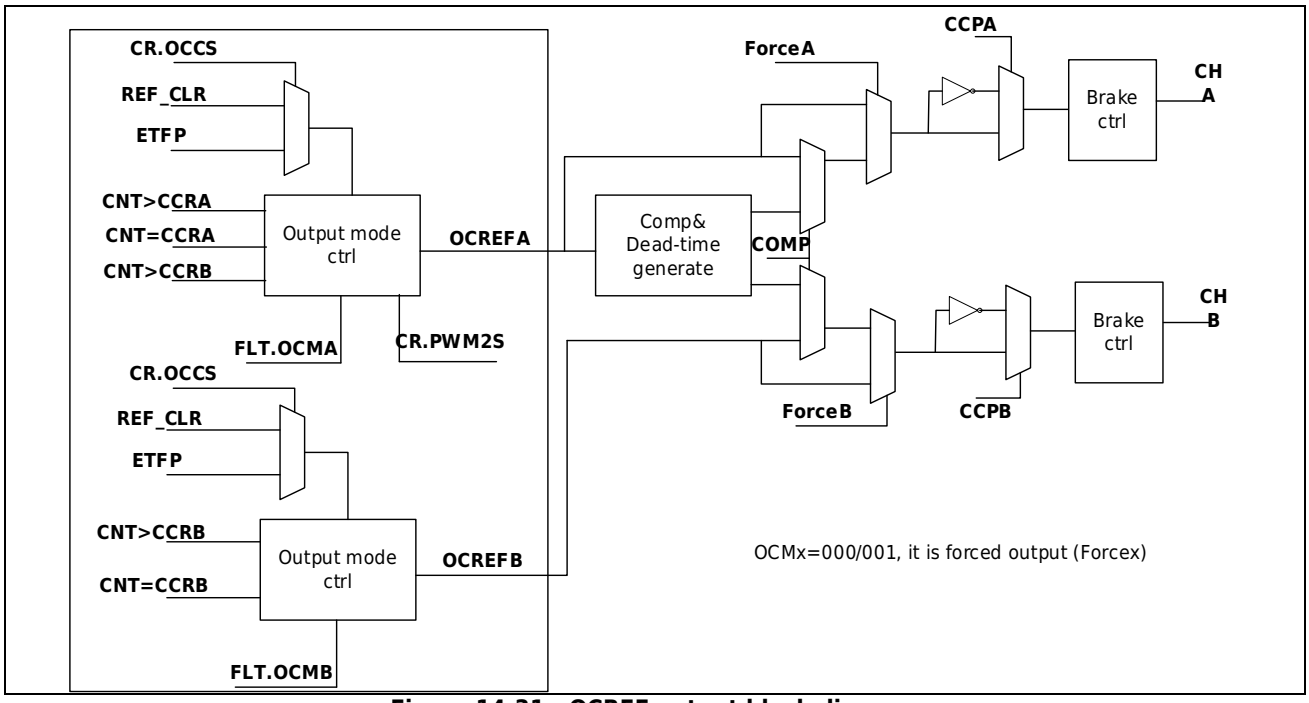


Figure 14-31 OCREF output block diagram

OCREF output is selected using OCMx

000: forced to 0

001: Forced to 1

010: Forced to 0 when comparing matches

011: Forced to 1 when comparing matches

100: flip when comparison matches

101: Output a high level for one count period when comparing matches

110: PWM mode 1

Single point comparison:

When counting up, CNT<CCRxy outputs high, and when counting down, CNT>CCRxy outputs low level

Two-point comparison:

1) Counting CCRxA<CNT≤CCRxB output on the sawtooth wave is low level

2) Counting under sawtooth wave CCRxA<CNT≤CCRxB output is high level

3) Triangular wave up counting CNT<CCRxA output high, down counting CNT>CCRxB is low level

111: PWM mode 2

Single point comparison:

When counting up, CNT<CCRxy outputs low, when counting down, CNT>CCRxy outputs high level

Two-point comparison:

1) Counting CCRxA≤CNT<CCRxB output on the sawtooth wave is high level

2) Counting under sawtooth wave CCRxA≤CNT<CCRxB output is low level

3) Triangular wave up count CNT<CCRxA output low, down count CNT>CCRxB is high level

Note: Forced output has high priority, when forced output is valid, complementary output control is invalid.

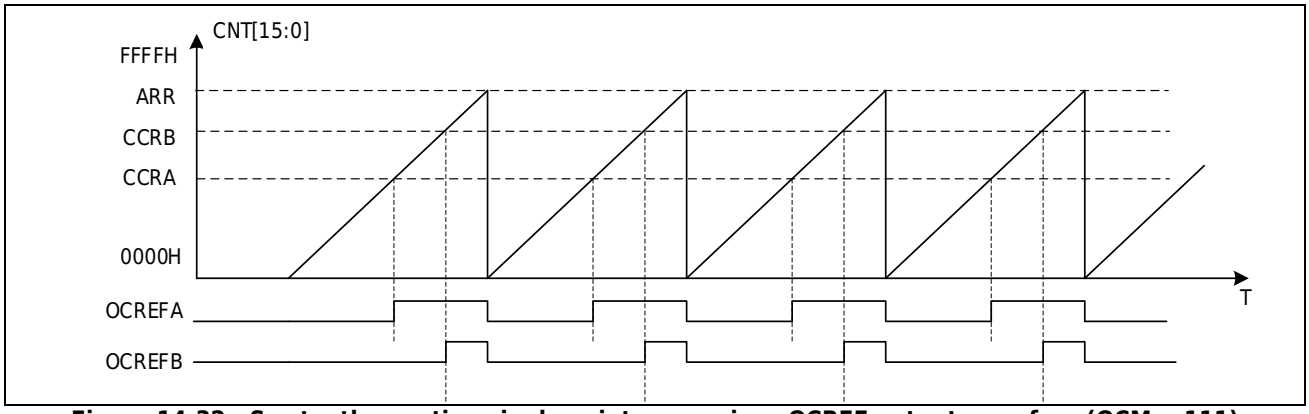


Figure 14-32 Sawtooth counting single point comparison OCREF output waveform(OCMx=111)

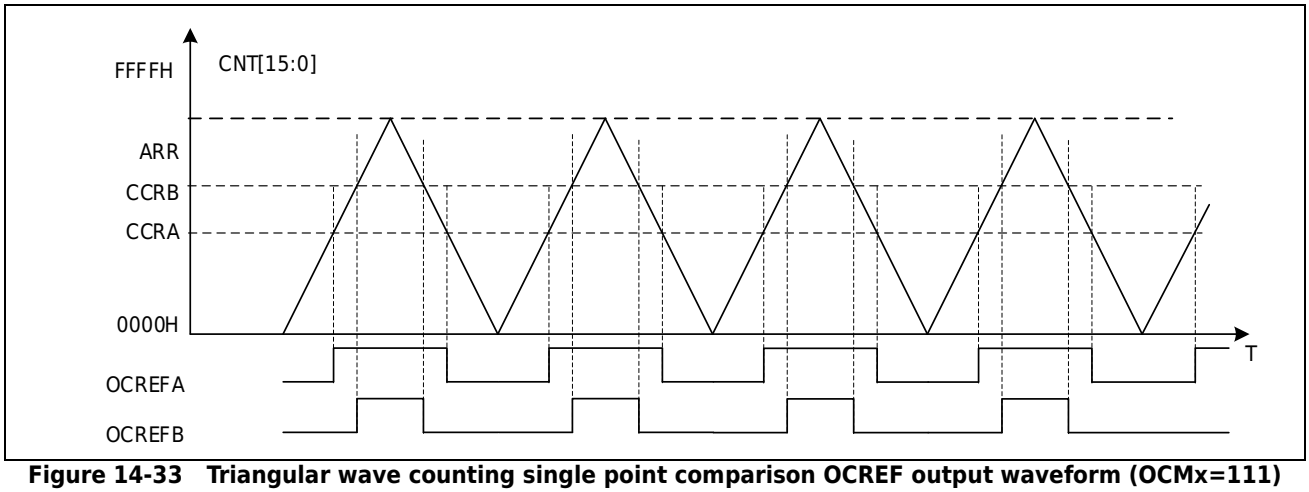


Figure 14-33 Triangular wave counting single point comparison OCREF output waveform (OCMx=111)

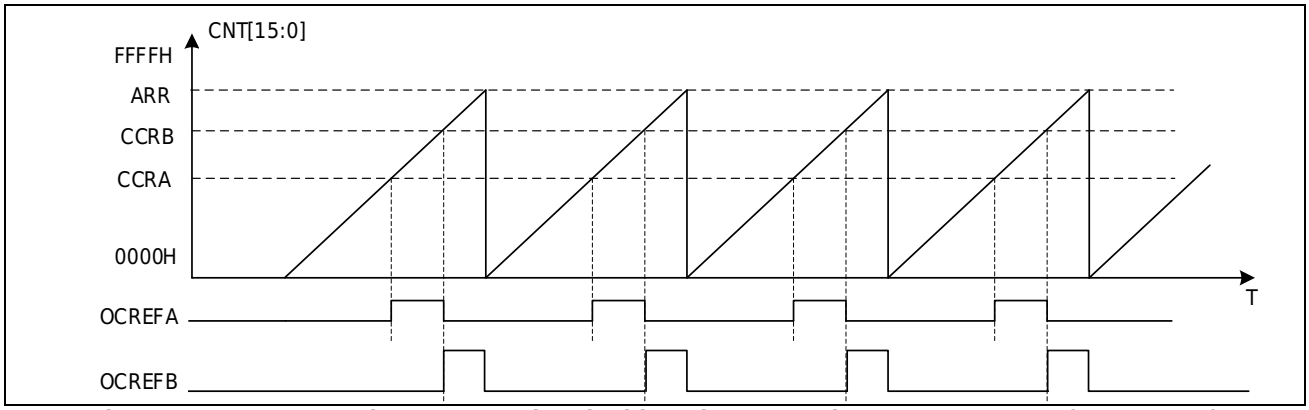


Figure 14-34 Sawtooth wave counting double-point comparison OCREF output (OCMx=111)

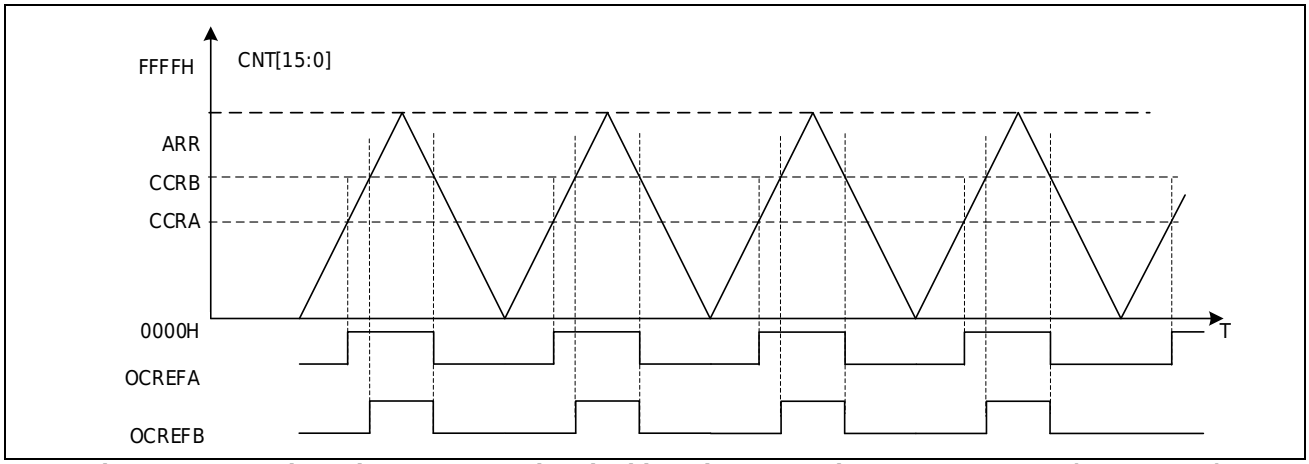


Figure 14-35 Triangular wave counting double-point comparison OCREF output (OCMx=111)

14.2.6.6 Independent PWM output

CHA is controlled by OCREFA, and the output of CHB is controlled by OCREFB. Through CRCHx.CCPA, CRCHx.CCPB can control the reverse of CHA and CHB output.

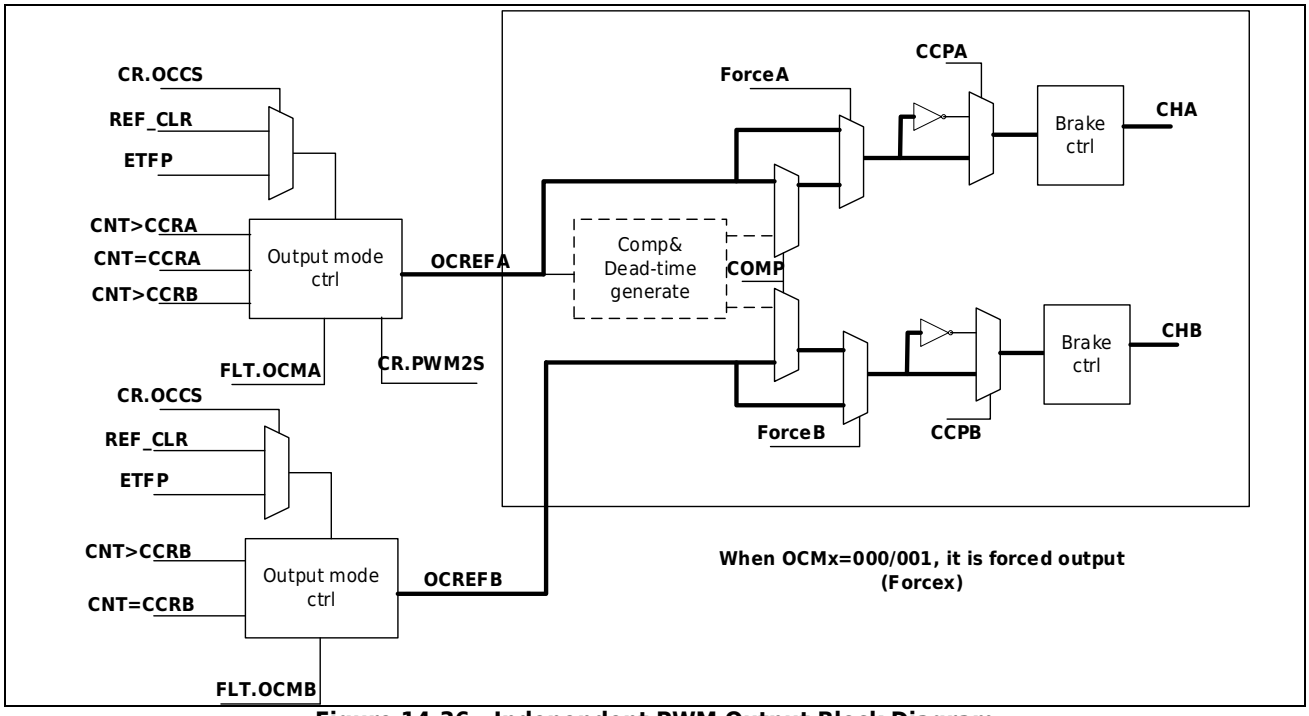


Figure 14-36 Independent PWM Output Block Diagram

Relationship between PWM output and OCREF

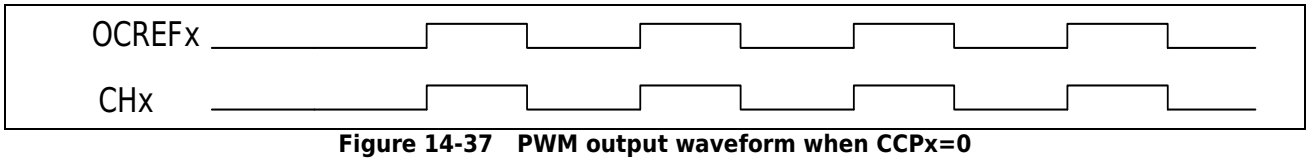


Figure 14-37 PWM output waveform when CCPx=0

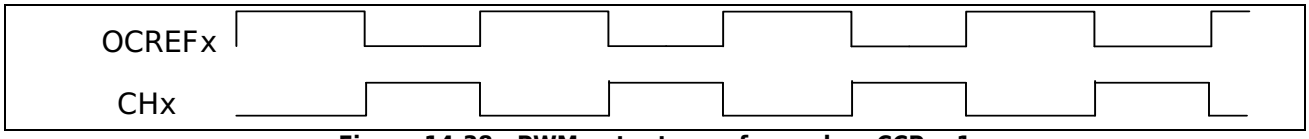


Figure 14-38 PWM output waveform when CCPx=1

14.2.6.7 Complementary PWM output

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

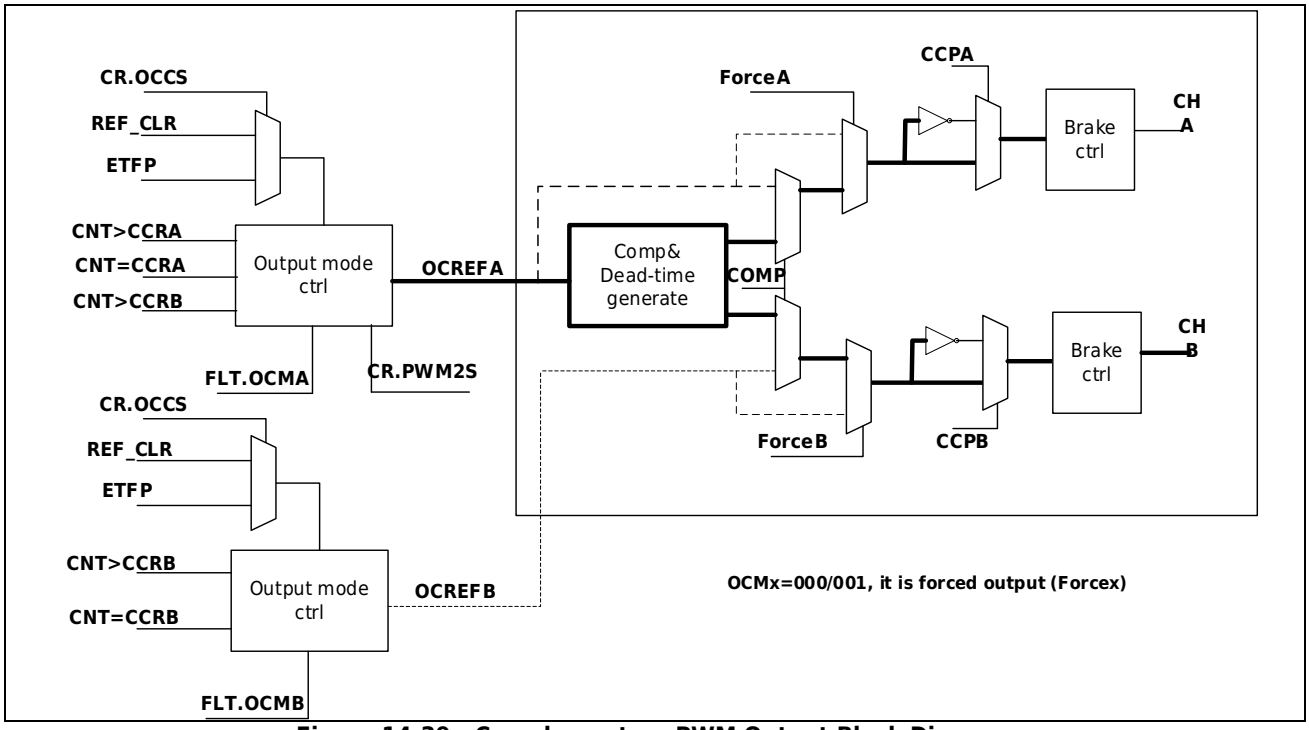


Figure 14-39 Complementary PWM Output Block Diagram

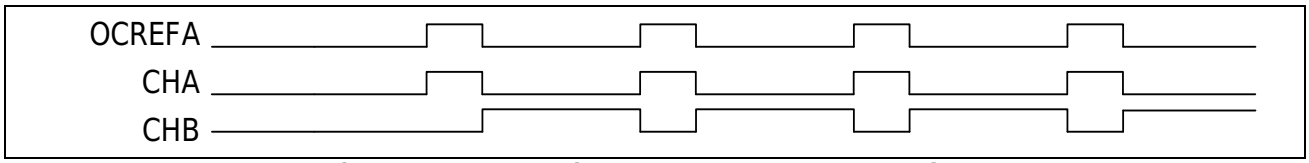


Figure 14-40 Complementary PWM output waveform

14.2.6.8 PWM output with dead zone

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

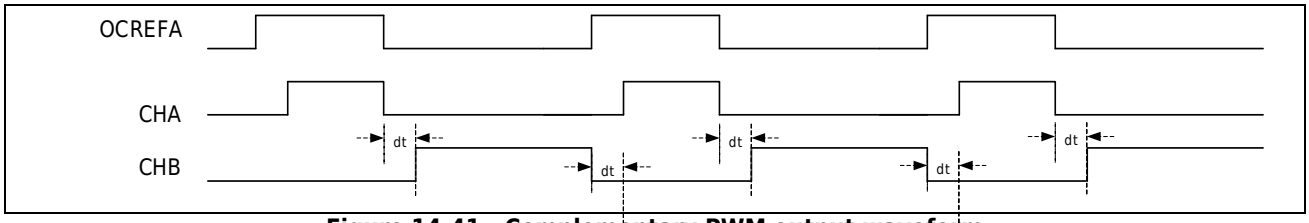


Figure 14-41 Complementary PWM output waveform

The dead time is controlled by 8 -bit DTR, and the relationship between the dead time dt and DTR is as follows

DTR[7] =0	T=DTR[6:0]+2	2-129	step=1
DTR[7:6] =10	T={DTR[5:0]+64}*2 +2	130-256	step=2
DTR[7:5] =110	T={DTR[4:0]+32}*8 +2	258-506	step=8
DTR[7:5] =111	T={DTR[4:0]+32}*16 +2	514-1010	step=16

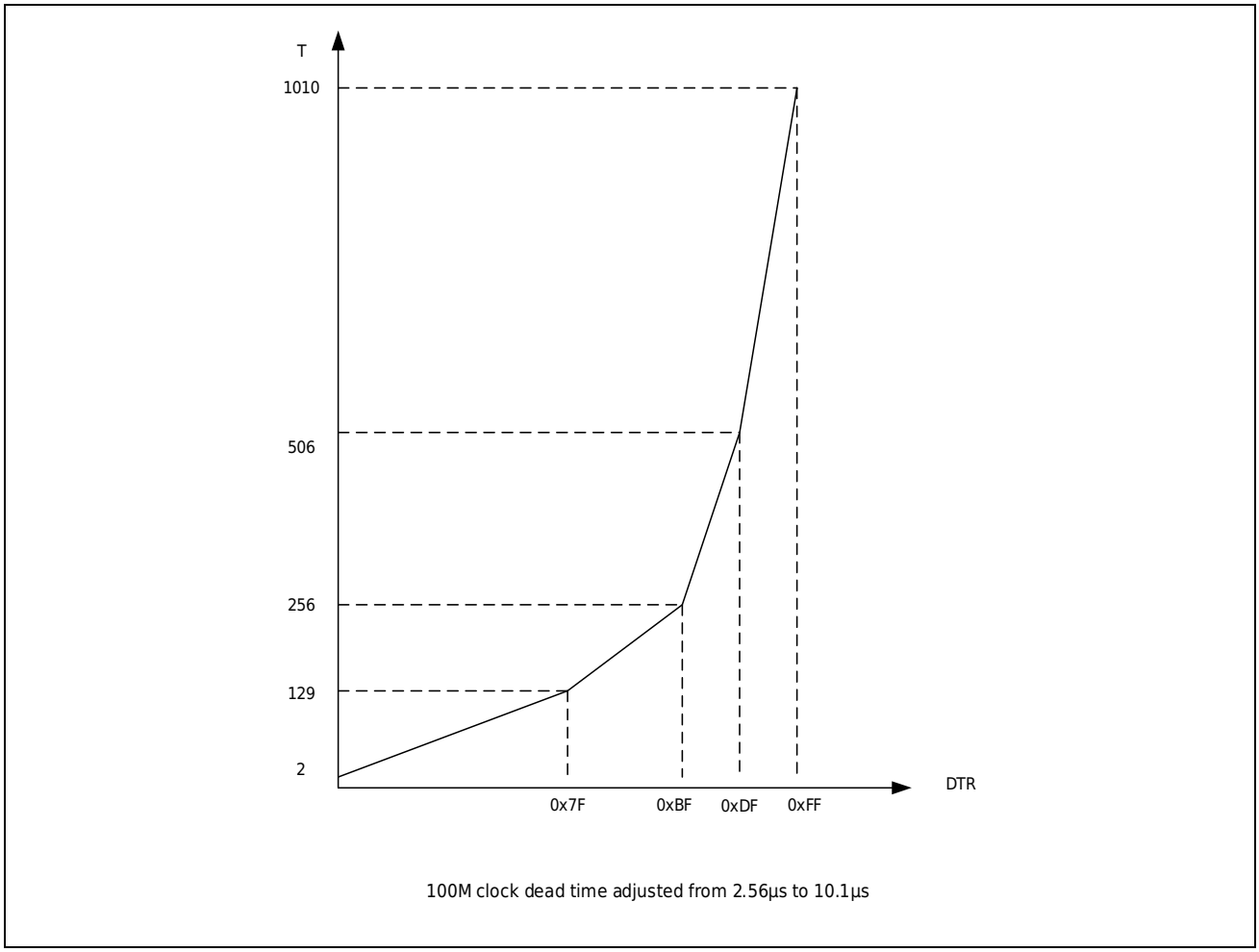


Figure 14-42 Dead time

14.2.6.9 Single pulse output

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

The counter can be started by the slave mode controller. can be generated in output compare mode or PWM mode. Will ONESHOT bit in the TIMx_M23CR register to 1. In this way, the counter will automatically stop when the next update event UEV occurs.

Do not set the initial value of the counter in the sawtooth counting mode to 0 in the single pulse mode, and do not set the counting value in the up counting mode to be greater than or equal to ARR.

The output pulses are only generated correctly when the comparison value differs from the counter initial value. Before starting (when the timer is waiting to be triggered), the following configurations must be made:

- When counting up: CNT<CCRxy ≦ ARR (special attention, 0<CCRxy),
- When counting down: CNT>CCRxy.

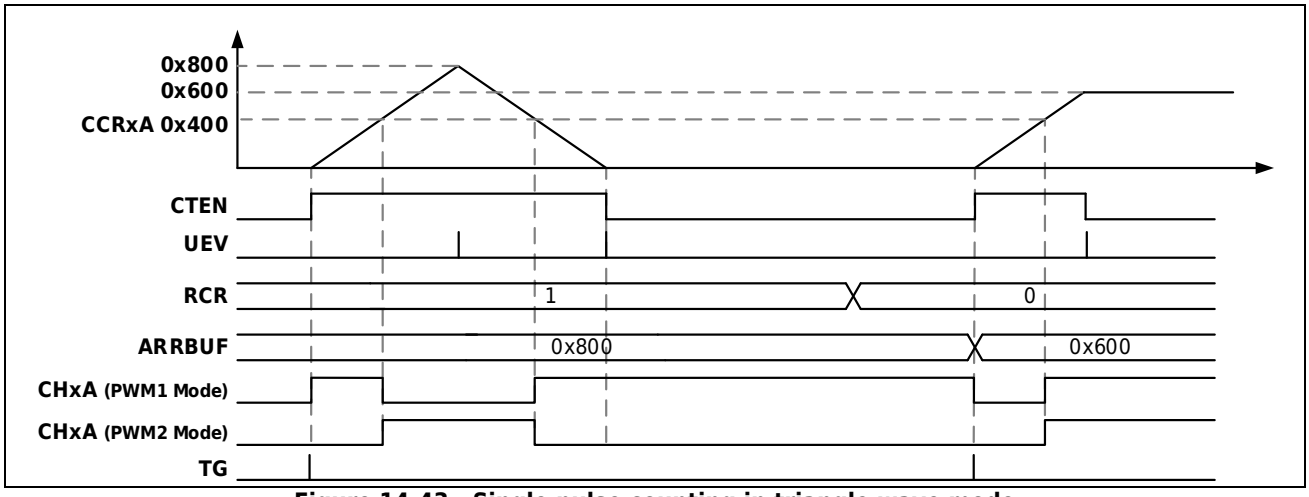


Figure 14-43 Single pulse counting in triangle wave mode

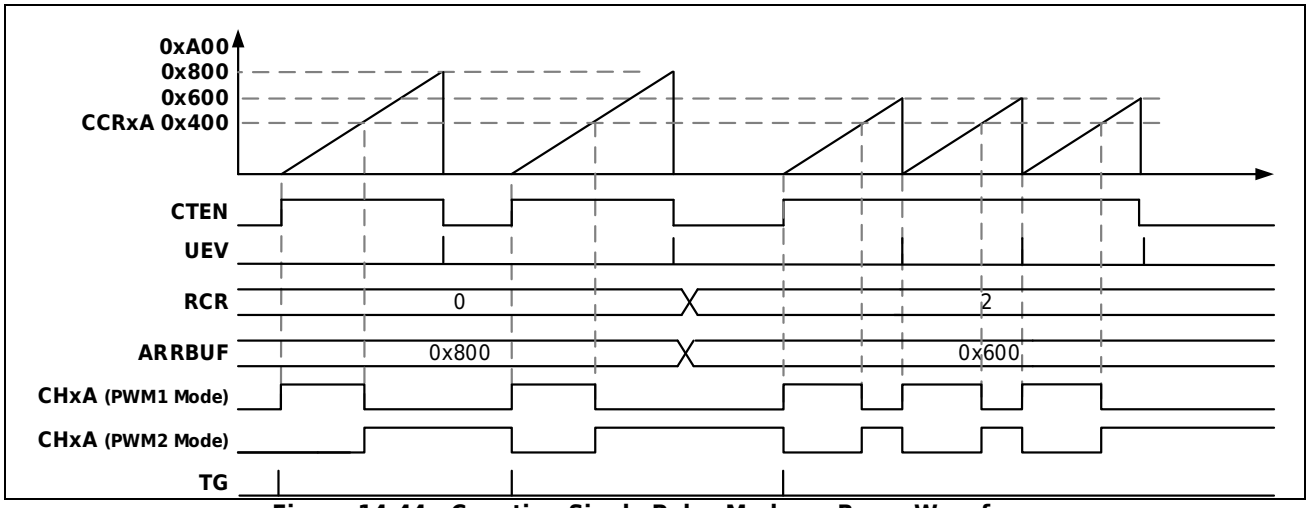


Figure 14-44 Counting Single Pulse Mode on Ramp Waveform

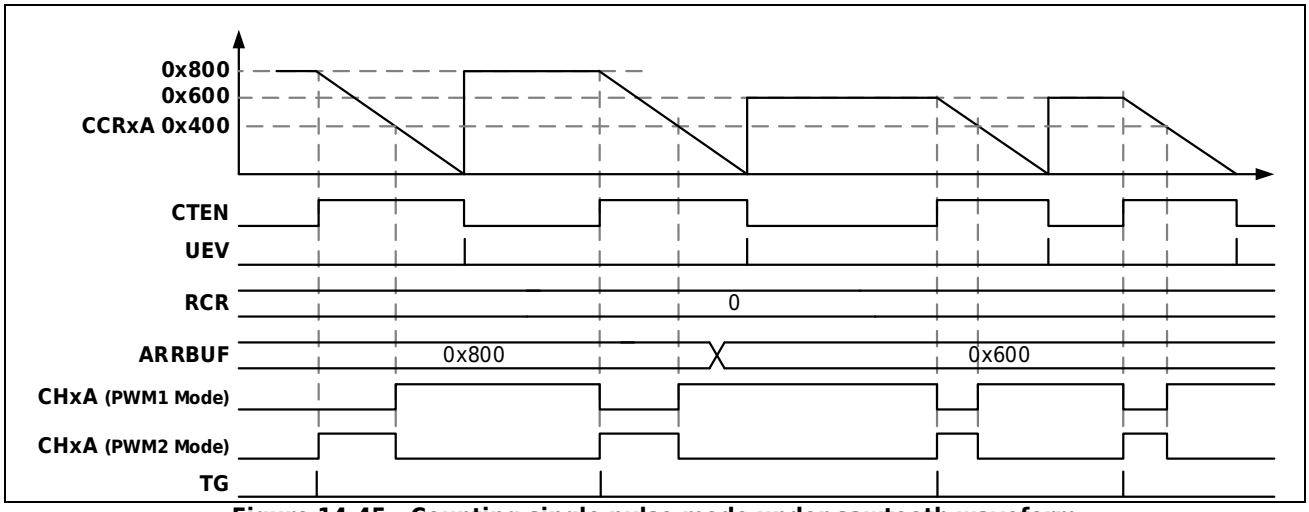


Figure 14-45 Counting single pulse mode under sawtooth waveform

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

14.2.6.10 Compare interrupt

The sawtooth comparison match will set the corresponding flag of the IFR register to 1, and if the interrupt is enabled CRCHx.CIEy (x=0,1,2;y=A,B) will trigger an interrupt.

For triangular wave comparison match, you can select up-count comparison match, down-count comparison match or both comparison matches separately.

Compare A Compare Match When the count value is equal to CCRxA, use CR.CIS control uniformly.

When CR.CIS=2'b00 compare match no output

When CR.CIS=2'b01 compare match when counting up

When CR.CIS=2'b10 compare match when counting down

When CR.CIS=2'b11 Comparing matches when counting up and down

Compare B Compare Match When the count value is equal to CCRxB, different channels can be controlled individually using CRCHx.CISB.

When CRCHx.CISB=2'b00 Channel x compare match no output

When CRCHx.CISB=2'b01 compare match while counting on channel x

When CRCHx.CISB=2'b10 Compare match when counting down channel x

When CRCHx.CISB=2'b11 Comparing matches when channel x counts up and down

The B channel comparison match is controlled separately for more flexible triggering of the ADC. For details, refer to the Timer Trigger ADC chapter.

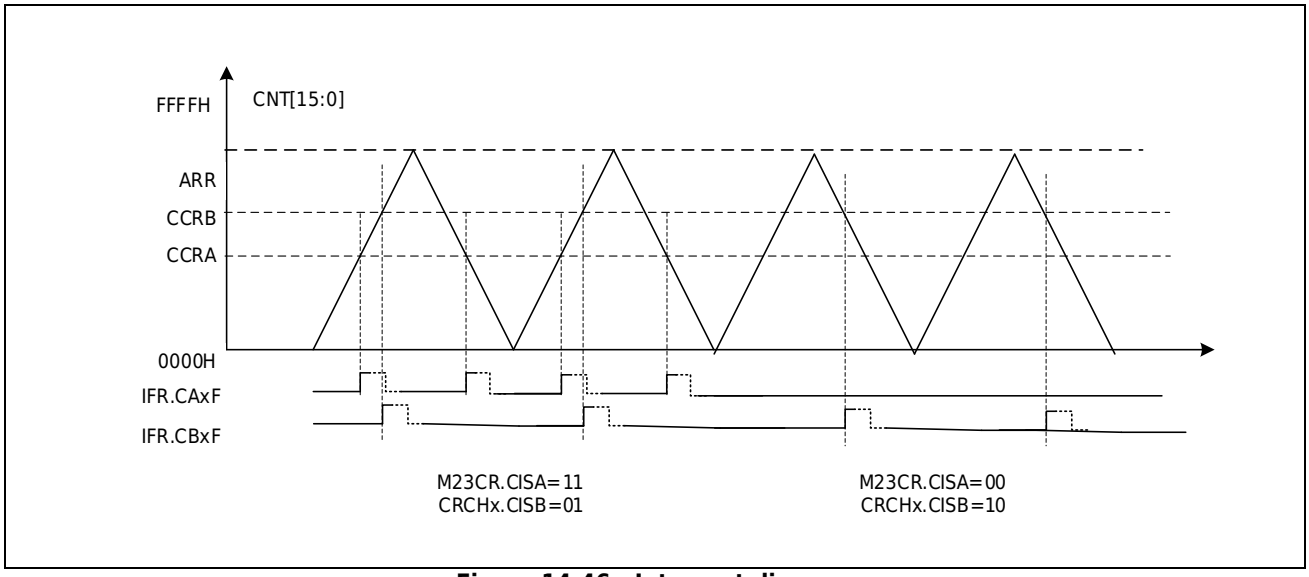


Figure 14-46 Interrupt diagram

14.2.6.11 Capture input

CR.MODE=2/3

The capture function can be set in the triangular wave counting or sawtooth wave counting mode, and the level edge of the capture can be set. When the capture occurs, the captured value is stored in the comparison capture register and a capture interrupt is generated.

The compare capture function of each channel can be set individually, selected by the register CRCHx.CSA/CSB.

the edge of the capture trigger can be selected through the register CRCHx.CFy/CRy (x=0/1/2;y=A/B).

When the capture action occurs again before the capture flag is cleared after the capture occurs, the capture data overwrite flag will be generated.

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

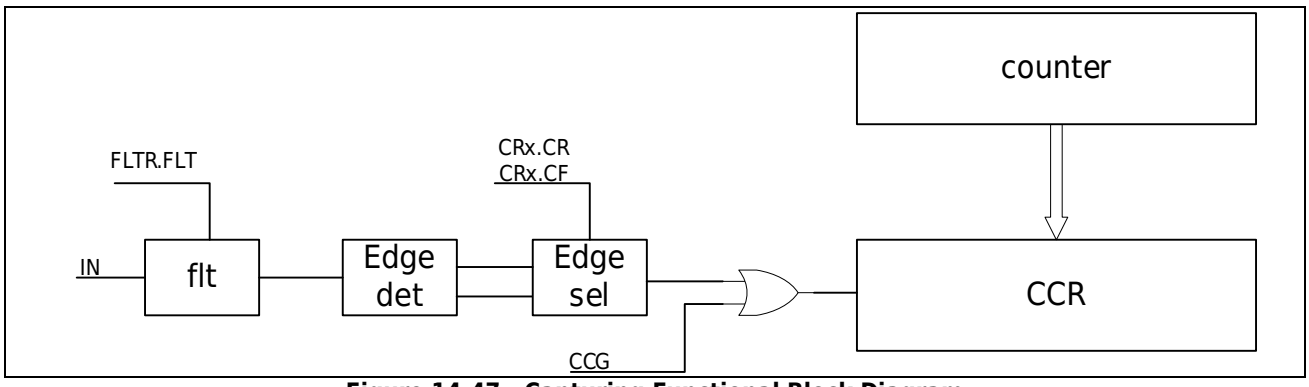


Figure 14-47 Capturing Functional Block Diagram

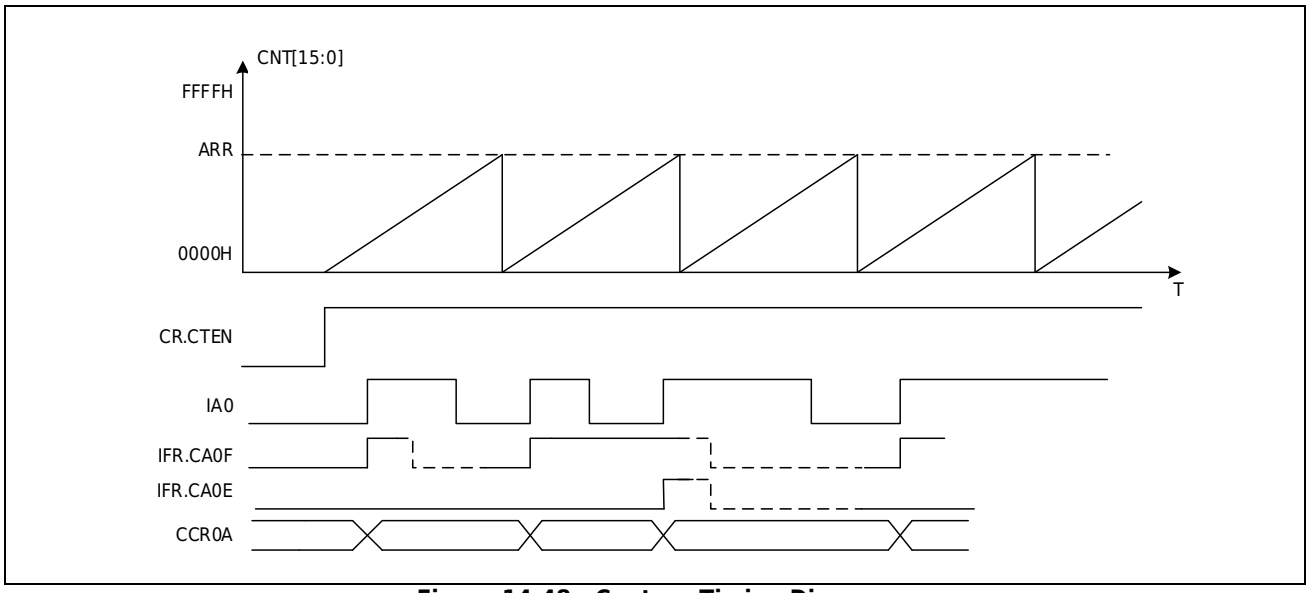


Figure 14-48 Capture Timing Diagram

CH0A selects CH0A input or the XOR input of CH0A, CH1A, and CH2A through MSCR.IA0S.

IA0S	0	1
Timer0/1/2	CHA0	CHA0 ETR GATE XOR input
Timer3	CHA0	CHA0 CHA CHA2 XOR input

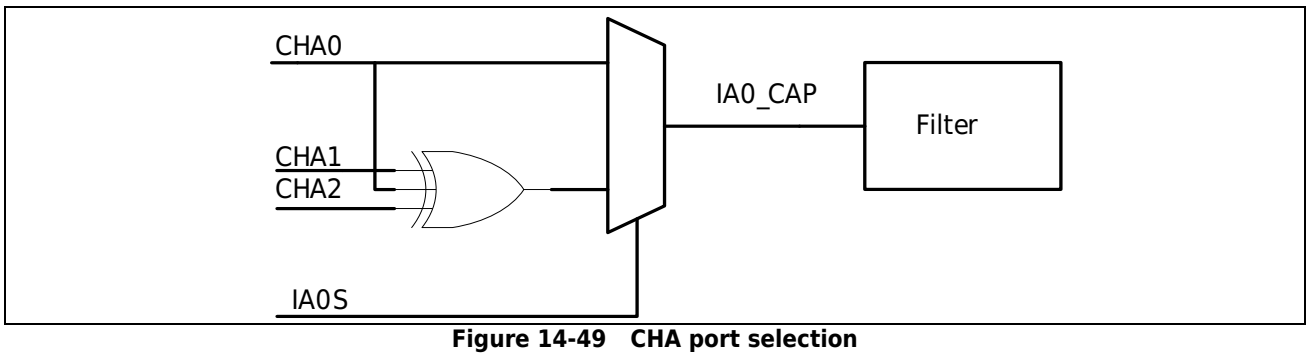


Figure 14-49 CHA port selection

The capture input of CH0B is selected by MSCR.IB0S as input to CH0B or the signal selected by content MSCR.TS.

<table>
<tr><th>IB0S</th><th>0</th><th>1</th></tr>
<tr><td>Timer0/1/2/3</td><td>CHB0</td><td>Internal trigger MSCR.TS select signal
<table>
<tr><th>MCSCR.TS</th><th>Channel B capture input</th></tr>
<tr><td>000</td><td>ETR filter phase selection output signal, filter phase selection can be configured</td></tr>
<tr><td>001</td><td>Interconnect ITR0</td></tr>
<tr><td>010</td><td>Interconnect ITR1</td></tr>
<tr><td>011</td><td>Interconnect ITR2</td></tr>
<tr><td>100</td><td>Interconnect ITR3</td></tr>
<tr><td>101</td><td>CH0A edge</td></tr>
<tr><td>110</td><td>CH0A filter output signal, filter function can be configured</td></tr>
<tr><td>111</td><td>Reserved</td></tr>
</table></td></tr>
</table>

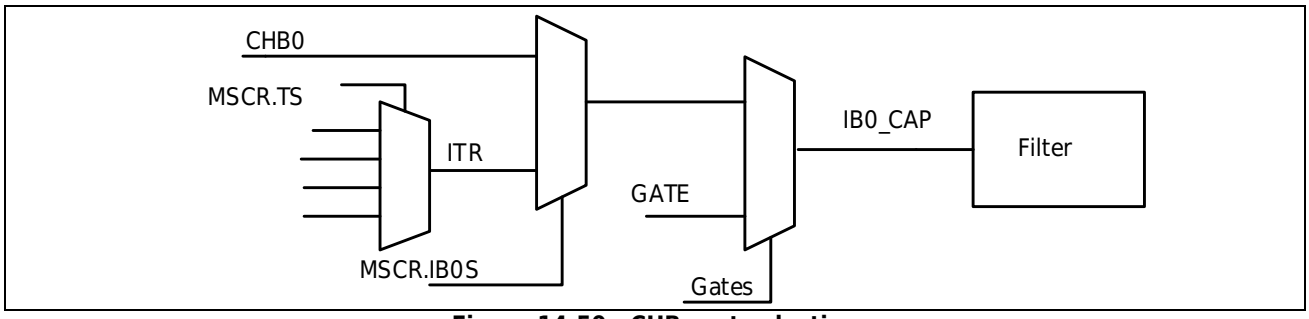


Figure 14-50 CHB port selection

The Gates signal is a signal generated by selecting the PWM complementary output function. When the PWM complementary output is used, the comparison capture register CCR0B is not used, and it is automatically switched to use GATE as the capture / comparison output.

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

14.2.6.12 Setup example

Edge-Aligned Independent PWM Output Settings

1. Set mode CR.MODE=2
2. Set the sawtooth wave counting direction CR.DIR
3. Set PWM period value ARR
4. Set the initial value of the counter (the initial value must be less than the period value)
5. Set PWM comparison value CCRxA, CCRxB

6. Set PWM output comparison mode FLT.OCMA FLT.OCMB to 6 or 7
7. Clear the associated interrupt flag
8. Enable the corresponding interrupt
9. Set output polarity FLT.CCPAx FLT.CCPBx
10. Enable output DTR.MOE
11. Enable timer CR.CTEN

Center-Aligned Complementary PWM Output Settings

1. Set mode CR.MODE=3
2. Set Complementary Output CR.COMP
3. Set PWM period value ARR
4. Set the initial value of the counter (the initial value must be less than the period value)
5. Set the PWM comparison value CCRxA (CCRxB does not need to be set, the PWM output has nothing to do with this register)
6. Set PWM output comparison mode FLT.OCMA FLT.OCMB to 6 or 7
7. Clear the associated interrupt flag
8. Enable the corresponding interrupt
9. Set output polarity FLT.CCPAx FLT.CCPBx
10. Enable output DTR.MOE
11. Enable timer CR.CTEN

Triangular wave non-center alignment with dead zone complementary PWM output setting

1. Set mode CR.MODE=3
2. Set Complementary Output CR.COMP
3. Set two compare enable CR.PWM2S
4. Set PWM period value ARR
5. Set the initial value of the counter (the initial value must be less than the period value)
6. Set the PWM comparison value CCRxA, CCRxB, the upper counting comparison point is CCRxA, and the lower counting comparison point is CCRxB
7. Set PWM output comparison mode FLT.OCMA FLT.OCMB to 6 or 7
8. Clear the associated interrupt flag
9. Enable the corresponding interrupt
10. Set output polarity FLT.CCPAx FLT.CCPBx
11. Set dead zone enable DTR.DTEN
12. Set dead time DTR.DT
13. Enable output DTR.MOE

14. Enable timer CR.CTEN

Capture function settings

Set CH0B as rising edge capture function

1. Select the counter counting method, set mode=2
2. Select CH0B as capture mode CRCHx.CSB=1
3. FLT.FLTB0 as required
4. Select the captured edge CRCHx.CRB=1
5. Set ARR to change the period value
6. Start timer CR.CTEN
7. Clear the associated interrupt flag
8. Clear the capture flag and enable the corresponding interrupt
9. After querying the capture flag, read CCR0B to obtain the capture value

Note: If the capture edge is valid when the timer is not started, the capture flag and capture action will also be generated.

Complementary PWM output uses GATE as PWM output function

1. Set to PWM complementary output mode, refer to " Center Aligned Complementary PWM Output Setting "
2. Set CCR0B to set GATE PWM comparison value
3. Set FLT.OCMB0=6/7 to set PWM output mode
4. Enable PWM output DTR.MOE

Complementary PWM output uses GATE falling edge as capture input function

1. Set to PWM complementary output mode, refer to " Center Aligned Complementary PWM Output Setting "
2. Set GATE as capture input CR.CSG
3. Select GATE to capture the input edge and set CR.CFG (GATE is used as the capture input B channel filter setting is invalid)
4. Start timer CR.CTEN
5. Clear the associated interrupt flag
6. Clear the capture flag and enable the corresponding interrupt
7. After querying the capture flag, read CCR0B to obtain the capture value

14.2.7 Mode 2/3 Slave Mode

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

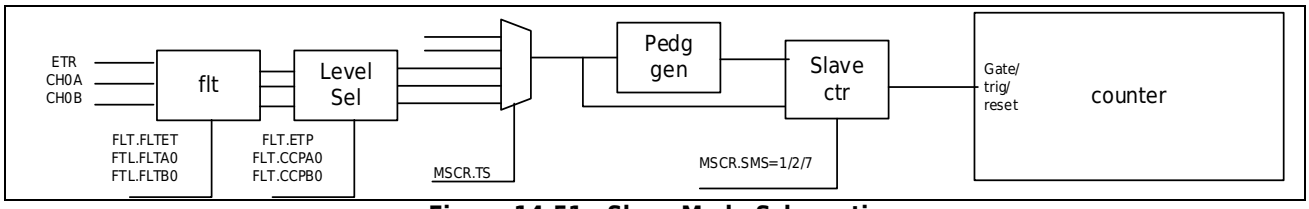


Figure 14-51 Slave Mode Schematic

Select from mode function by MSCR.SMS;

MSCR.SMS	Select from mode function 001: Reset function; 010: Trigger mode; 111: Gating function
MSCR.TS	Trigger selection 000: The signal ETFP after the filter phase selection of port ETR, the filter function can be configured 001: Internal interconnection signal ITR0 010: internal interconnection signal ITR1; 011: internal interconnection signal ITR2; 100: internal interconnection signal ITR3; 101: edge signal of port CH0A; 110: Port CH0A filters the signal IAFP after phase selection, the filter function can be configured 111: The signal IBFP after the filter phase selection of port CH0B, the filter function can be configured

14.2.7.1 Gated count

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

- Configure channel CH0A to detect a low level on CH0A. Configure the input filter bandwidth (in this example, no filtering is required, so keep FLT.FLTA0=000). Set CCPA0=1 in the FLTR register to determine the polarity (only detect low level).
- Set SMS=111 in the SMCR register to configure the timer as gated mode; set TS=110 in the SMCR register to select CH0A as the input source.
- Set CTEN=1 in the CR register to start the counter. In gated mode, if CTEN=0, the counter cannot be started regardless of the trigger input level. As long as CH0A is low, the counter starts counting according to the internal clock, and stops counting once CH0A becomes high.

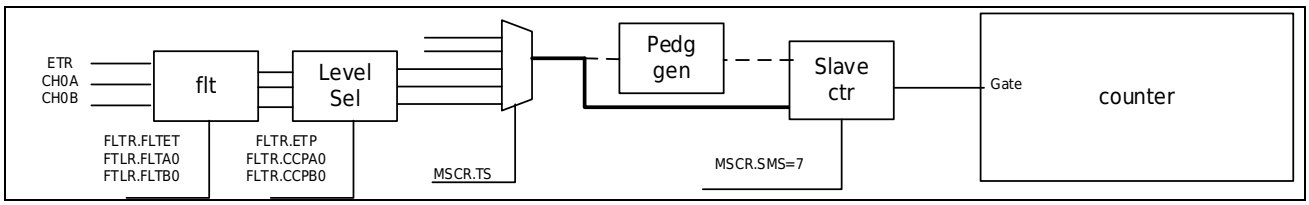


Figure 14-52 Gating function

14.2.7.2 trigger function

Using external triggers (CH0A, CH0B, ETR) can start the timer synchronously, and using the internal interconnection signal of the timer combined with MSCR.MSM can also configure the timer to start synchronously. The trigger signal is the rising edge of the input signal. You can also use software to write CR.TG to start the software trigger function.

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

- Configure channel CH0B to detect the rising edge of CH0B. Configure the input filter bandwidth (in this example, no filter is needed, keep FLT.FLTB0=000). Set CCPB0=0 in the FLTR register to determine the polarity (no inversion).
- Set SMS=010 in the SMCR register to configure the timer as trigger mode; set TS=111 in the SMCR register to select CH0B as the input source. When a rising edge appears on CH0B, the counter starts counting driven by the internal clock and sets the TIF flag at the same time. The delay between the rising edge of CH0B and the start of the counter depends on the resynchronization circuit at the CH0B input.
  Note: If you use the falling edge trigger, first select the trigger polarity, and then select the mode, otherwise false triggers will occur.

14.2.7.3 reset count

When a trigger input event occurs, the counter and its prescaler can be re-initialized; at the same time, if the URS bit of the CR register is low, an update event UEV is also generated; then all preload registers (ARR, CCRx) are updated.

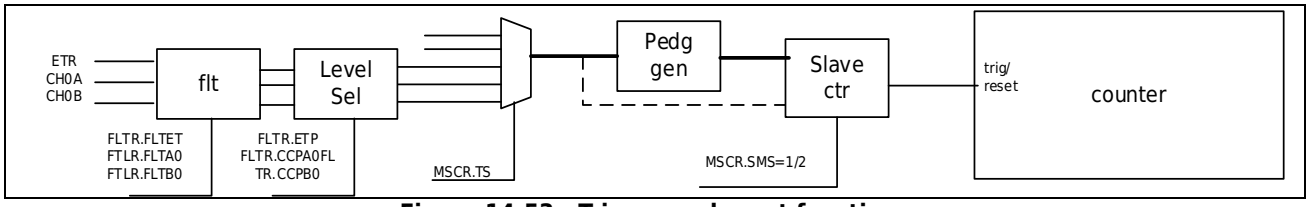


Figure 14-53 Trigger and reset functions

14.2.8 Quadrature code counting function

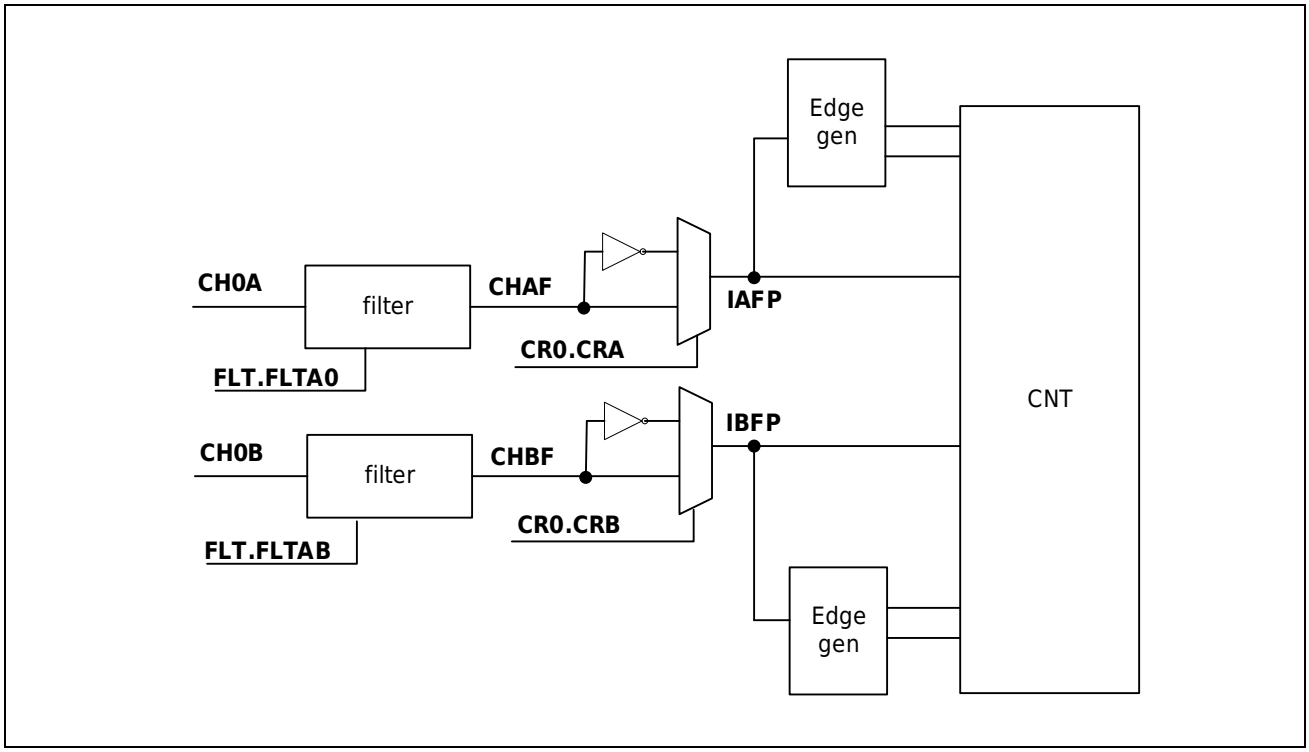
MSCR.SMS=4/5/6, corresponding to the mode 1/2/3 of the orthogonal coding mode. At this time, the counter encodes and counts according to the phase relationship of IAFP and IBFP. IAFP, IBFP is the port input CH0A, CH0B filter phase selection signal.

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

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

			IAFP		IBFP	
	IBFP	**IAFP**	**Rising**	**falling**	**Rising**	**falling**
MOD1	High		Down	Up	-	-
	Low		up	Down	-	-
MOD2		High	-	-	up	Down
		Low	-	-	Down	Up
MOD3	High	High	Down	Up	up	Down
	Low	Low	up	Down	Down	Up

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



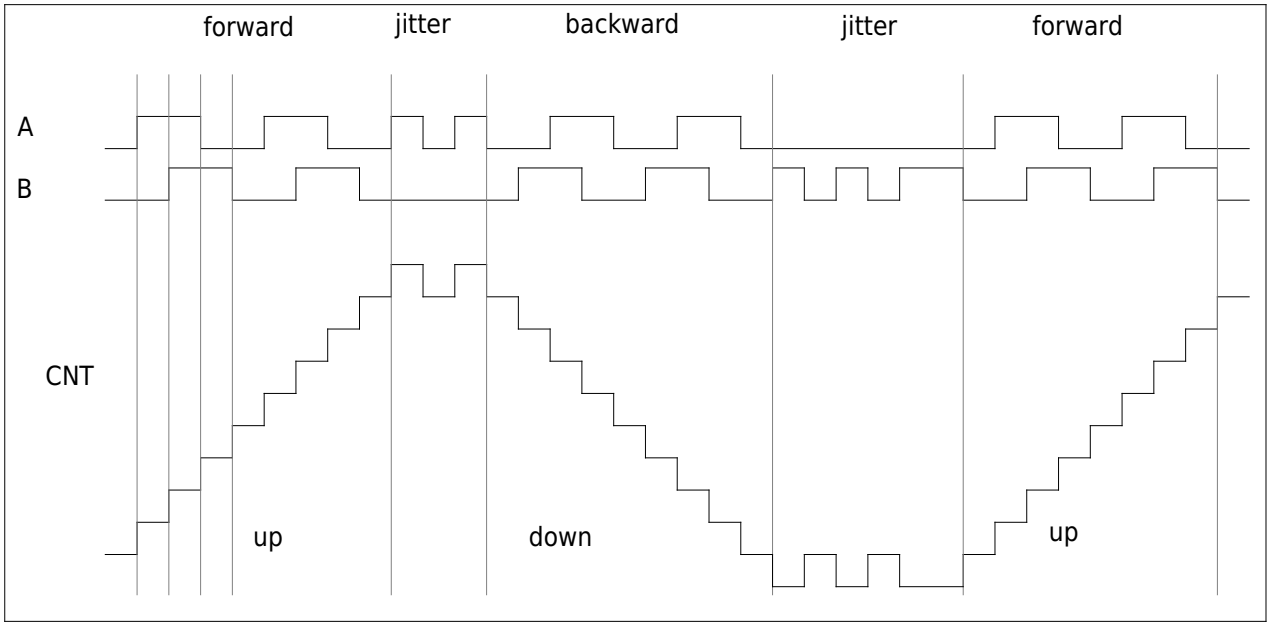


Figure 14-54 Code count

14.2.9 Timer triggers ADC

CCMR0A, CCMR1A, CCMR2A comparison match can be configured to trigger ADC, and center-aligned PWM can only be selected through the control register CR.CIS to trigger on rising match or falling match.

CCMR0B, CCMR1B, CCMR2B comparison match can be configured to trigger ADC. When center-aligned PWM, three matching trigger points (rising, falling, rising and falling) can be controlled respectively through the register CRx.CISB.

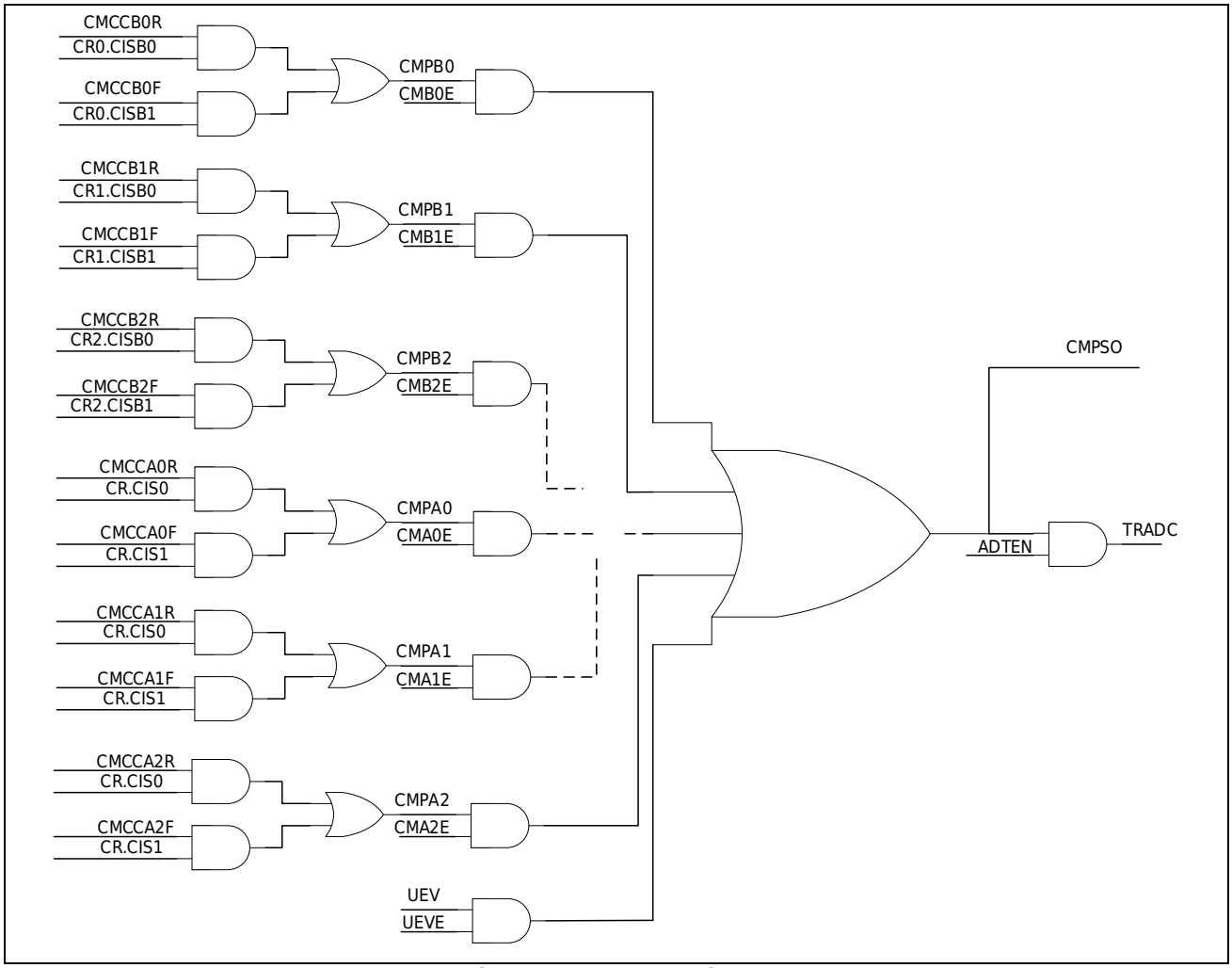


Figure 14-55 ADC trigger

Complementary PWM output B compare as ADC trigger function

1. Set to PWM complementary output mode, refer to " Center Aligned Complementary PWM Output Setting "
2. Set CCRxB to set ADC trigger comparison point
3. Set CRCHx.CISB to select up-counting and down-counting comparison matching (sawtooth wave does not need to be selected)
4. Select the source ADTR for ADC trigger comparison
5. Enable ADC trigger ADTR.ADTE

14.2.10 Brake control

The VC comparison output can control the braking function, the external BK port input can control the braking function, and the system fail can control the braking function. Through CR.BG, the software brake function can be realized, and the output port can be controlled to the set state.

The two-channel output Tim0/1/2 can choose to use the brake port of Tim0 to control the brake function of TIM1/2, or use their own brake input to control their respective brake functions. When DTR.BKSEL is set to 1, TIM1/2 share the brake input of TIM0.

14.2.11 Timer interconnection

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

	ITR0	ITR1	ITR2	ITR3
Timer0	-	TIM1_TRGO	TIM2_TRGO	TIM3_TRGO
Timer1	TIM0_TRGO	-	TIM2_TRGO	TIM3_TRGO
Timer2	TIM0_TRGO	TIM1_TRGO	-	TIM3_TRGO
Timer3	TIM0_TRGO	TIM1_TRGO	TIM2_TRGO	-

Timer TIM0/1/2 start PWM output setting at the same time (use CTEN of TIM0 to trigger TIM1/2) example

1. Refer to the PWM output setting to set the pulse adjustment output of timer 0/1/2
2. Set TIM0 as master mode MSCR.MSM=1
3. Set CTEN of TIM0 to trigger the other two timers MSCR.MMS=1
4. Keep MSCR.MSM=0 of TIM1/2
5. Set TIM1/2 as trigger mode MSCR.SMS=2
6. Select the trigger source of TIM1/2 as TRGO of TIM0, MSCR.TS=1
7. Finally enable the timer TIM0, start the timer CR.CTEN=1

Timer TIM0/1/2 start PWM output setting at the same time (use UG of TIM1 to trigger TIM0/2) example

1. Refer to the PWM output setting to set the pulse adjustment output of timer 0/1/2
2. Set TIM1 to trigger master mode MSCR.MSM=1
3. Set the UG of TIM1 to trigger the other two timers MSCR.MMS=0
4. Keep MSCR.MSM=0 of TIM0/2
5. Set TIM0/2 as trigger mode MSCR.SMS=2
6. Select the trigger source of TIM0/2 as TRGO of TIM0, MSCR.TS=2
7. Finally enable the timer TIM1, start the timer CR.UG=1

14.2.12 GATE input interconnection

The GATE input can be directly input from the port through PX_SEL, or it can be connected to other modules or ports through the port function register GPIO_TIMGS.

When TIMx_G=0x0, the gate control GATE input is the port input selected by PX_SEL. When TIMx_G=0x1~0x7, it is connected to the input or output of other modules.

	TIM0_g	TIM1_g	TIM2_g	TIM3_g
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	UART0_RXD	LPUART0_RXD	UART0_RXD	UART0_RXD
010	UART1_RXD	LPUART1_RXD	UART1_RXD	UART1_RXD
011	VC0_OUT	VC0_OUT	VC0_OUT	LPUART0
100	VC1_OUT	VC1_OUT	VC1_OUT	LPUART1
101	PA03	PA08	PA10	VC0_OUT
110	PB08	PB03	PB04	PA06
111	PB15	PB13	PB11	PA11

14.2.13 ETR input interconnection

ETR input can be directly input from the port, or can be connected to other modules or ports through the selection of the port function register GPIO_TIMES.

When TIMx_E=0x0, the external clock EXT input is the port input selected by PX_SEL. When TIMx_E=0x1~0x7, it is connected to the input or output of other modules.

	TIM0_e	TIM1_e	TIM2_e	TIM3_e
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	LPUART0_RXD	UART0_RXD	LPUART0_RXD	UART0_RXD
010	LPUART1_RXD	UART1_RXD	LPUART1_RXD	UART1_RXD
011	VC0_OUT	VC1_OUT	VC0_OUT	VC1_OUT
100	LVD_OUT	LVD_OUT	PCNT_S1	PCNT_S0
101	PA00	PA01	PA04	PA00
110	PA05	PC09	PC04	PA12
111	PA15	PD02	PC08	PA13

14.2.14 CHx capture input interconnect

The CHA of Timer0/1/2 and the CH0A/CH0B input of Timer3 can be directly input from the port, or can be connected to other modules or ports through the selection of the port function register GPIO_TIMCPS.

	TIM0_CHA	TIM1_CHA	TIM2_CHA	TIM3_CH0A	TIM3_CH0B
000	PX_SEL	PX_SEL	PX_SEL	PX_SEL	PX_SEL
001	UART0_RXD	UART1_RXD	LPUART0_RXD	LPUART1_RXD	UART0_RXD
010	PA00	PA00	VC0_OUT	LPUART0_RXD	UART1_RXD
011	PA02	PA02	PA02	PCNT_S0	PCNT_S1
100	PA05	PA06	PA07	VC0_OUT	VC1_OUT
101	PA15	PB08	PB08	PA08	PA07
110	PB06	PB10	PB09	PB03	PB04
111	PB14	PB13	PC06	PB06	PB13

When TIMx_CHy=0x0, the capture input is the port input selected by PX_SEL. When TIMx_CHy=0x1~0x7, it is connected to the input or output of other modules.

14.2.15 DMA

Timer supports software and hardware to trigger DMA for data transfer. Data is supported to be written to the timer from other locations, or read and written from the timer to other locations. It can be applied to the automatic handling of data after data capture and the automatic adjustment of the pulse width changing the period value or duty cycle. Each timer has two DMA requests, A request trigger source can choose compare capture A, external trigger, B request trigger source can choose compare capture B, event update.

IDREQ	Interrupt Signal of Peripheral
18	TIM0A
19	TIM0B
20	TIM1A
21	TIM1B
22	TIM2A
23	TIM2B
24	TIM3A
25	TIM3B

TIMxA can choose compare capture A, ETRIG

TIMxB can be selected to compare capture B, UEV

14.2.15.1 Setup example

Capture data DMA data transmission

1. enable DMA
2. Enable DMA channel enable
3. Select the channel for timer DMA
4. Set the transmission type, transmission length, transmission method
5. Set source start address, destination start address
6. Set the increment mode of source address and destination address
7. Enable DMA interrupts as needed
8. Set up data capture with reference to the data capture process
9. Enable CRCHx.CDy enables capture data to trigger DAM transfer.

Note: The source address is set to the capture channel register, the source address is fixed, and the destination address is added.

Pulse adjusted DMA data transfer

1. enable DMA
2. Enable DMA channel enable
3. Select the channel for timer DMA
4. Set the transmission type, transmission length, transmission method
5. Set source start address, destination start address
6. Set the increment mode of source address and destination address
7. Enable DMA interrupts as needed
8. Reference Pulse Adjustment Output Setting Pulse Modulation Output
9. Select the condition of hardware triggering DMA according to needs (UEV trigger or comparison match trigger)

Note: The target address is set as the capture channel register, the source address changes, and the target address is fixed.

14.3Timer register description

Table 14-1 Timer register list

Timer	base address	Description
Timer0	0x40000C00	Timer0基地址
Timer1	0x40000D00	Timer1基地址
Timer2	0x40000E00	Timer2基地址
Timer3	0X40005800	Timer3基地址

Register	Offset address	Description
TIMx_ARR	0X000	Timer reload register / cycle
TIMx_CNT	0X004	Timer 16 -bit mode count register
TIMx_CNT32	0X008	Timer 32 -bit mode count register Note: TIMx_CNT32 is the remapping register of TIMx_ARR and TIMx_CNT. In order to facilitate the 32 -bit timer write operation.
TIMx_M0CR	0X00C	Timer mode 0 control register (described by different modes)
TIMx_M1CR	0X00C	Timer mode 1 control register (described in different modes)
TIMx_M23CR	0X00C	Timer mode 23 control register (described in different modes)
TIMx_IFR	0X010	Timer interrupt flag
TIMx_ICLR	0X014	Timer interrupt clear register
TIMx_MSCR	0X018	Master-slave mode control
TIMx_FLTR	0X01c	Filter control
TIMx_ADTR	0X020	ADC trigger control
TIMx_CRCH0	0x024	Compare Unit 0 Control Register
TIMx_CRCH1	0X028	Compare Unit 1 Control Register
TIMx_CRCH2	0x02C	Compare Unit 2 Control Register
TIMx_DTR	0X030	Dead zone register
TIMx_RCR	0X034	Repeat Count Register
TIMx_ARRDM	0X038	Timer reload register / period map address
TIMx_CR0	0x024	control register
TIMx_CCR0A	0X03C	Compare 0A Register
TIMx_CCR0B	0x040	Compare 0B register
TIMx_CCR1A	0X044	Compare 1A Register
TIMx_CCR1B	0X048	Compare 1B Register
TIMx_CCR2A	0X04C	Compare 2A Register
TIMx_CCR2B	0X050	Compare 2B register

14.4Mode 0 Timer Register Description

14.4.1 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															
RW															

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15:0	ARR	16 -bit reload timer reload value register

14.4.2 16 -bit Mode Count Register (TIMx_CNT)

Offset address: 0x004

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

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

Reset value: 0x0000 0000

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15:0	CNT	16 -bit reload timer Count value register

14.4.3 32 -bit mode count register (TIMx_CNT32)

Offset address: 0x008

Reset value: 0x0000 0000

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

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

Bit	Symbol	Description
31:0	CNT32	32- bit timer Count value register Note: Only valid in mode 0 32-bit timer free counting mode, other modes prohibit writing this register

14.4.4 Control Register (TIMx_M0CR)

Offset address: 0x00C

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
Resvered		Mode		Resvered	UIE	GATEP	GATE
		RW			RW	RW	RW

7	6	5	4	3	2	1	0
Resvered	PRS			TOGEN	CT	MD	CTEN
	RW			RW	RW	RW	RW

Reset value: 0x0060 0008

Bit	Symbol	Description
31:14	Reserved	Reserved bit
13:12	MODE	Operating mode 00 Timer mode 0; 01 PWC mode 10 sawtooth wave mode; 11 triangle wave mode
11	Reserved	Reserved bit
10	UIE	Interrupt enable control, enable interrupt after writing 1
9	GATEP	Port GATE polarity control 0: Port GATE active high 1: Port GATE active low
8	GATE	Timer Gate Enable 0: No gate control, timer works when CTEN=1; 1: Gating is enabled, the timer works only when the port GATE is valid and CTEN=1;
7	Reserved	Reserved bit
6:4	PRS	Internal clock frequency division selection 000: 1; 001: 2; 010: 4; 011: 8; 100: 16; 101: 32; 110: 64; 111: 256;
3	TOGEN	Toggle output enable in mode 0 1: Toggle output enable 0: Inversion output turns off CHA, CHB output is low level
2	CT	Counting clock selection 0: Internal count clock TCLK 1: External count clock ETR;
1	MD	Mode selection 32 timing /16 timing mode selection 0: 32 -bit free count 1: 16 -bit reload count
0	CTEN	Timer enable 0: Timer stops; 1: Timer enable

14.4.5 Interrupt Flag Register (TIMx_IFR)

Offset address: 0x010

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															UIF
															RO

Bit	Symbol	Description
31:1	Res.	Reserved bit
0	UIF	overflow interrupt

14.4.6 Interrupt Flag Clear Register (TIMx_ICLR)

Offset address: 0x014

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
Reserved															UIF
															R1W0

Bit	Symbol	Description
31:1	REV	Reserved bit
0	UIF	Overflow interrupt clear, write 0 to clear

14.4.7 Dead Time Register (TIMx_DTR)

Offset address: 0x030

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			MOE	Resvered											
			RW												

Bit	Symbol	Description
31:13	Reserved	Reserved bit
12	MOE	Toggle output enable 0: Toggle output to input state 1: flip port to output state
11:0	Reserved	Reserved bit

14.5Pulse Width Measurement PWC Register Description

14.5.1 16 -bit Mode Count Register (TIMx_CNT)

Offset address: 0x004

Reset value: 0x0000 0000

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

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

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15:0	CNT	Register count value

14.5.2 Control Register (TIMx_M1CR)

Offset address: 0x00C

Reset value: 0x0060 0008

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
Resvered	Oneshot	Mode		Resvered	UIE	Edg2nd	Edg1st
	RW	RW			RW	RW	RW

7	6	5	4	3	2	1	0
Resvered	PRS			Resvered	CT	Resvered	CTEN
	RW				RW		RW

Bit	Symbol	Description
31:15	Reserved	Reserved bit
14	Oneshot	Single trigger mode selection 1: After completing a pulse measurement, it will end automatically, and you need to re-enable CTEN for another measurement. 0: Cyclic measurement
13:12	MODE	Operating mode 00 Timer mode 0; 01 PWC mode 10 sawtooth wave mode; 11 triangle wave mode
11	Reserved	Reserved bit
10	UIE	Interrupt enable control, enable interrupt after writing 1 Counting to 0XFFFF will overflow and generate an overflow flag
9	Edg2nd	Pulse width measurement end edge selection
8	Edg1st	Pulse width measurement start edge selection edg2nd Edg1st / 00 / 01 / 10 / 11 Measurement / Rising - upperiod / Low level width / High level width / Falling edge - falling edge cycle
7	Reserved	Reserved bit
6:4	PRS	Internal clock frequency division selection 000: 1; 001: 2; 010: 4; 011: 8; 100: 16; 101: 32; 110: 64; 111: 256;
3	Reserved	Reserved bit
2	CT	Counting clock selection 0: Internal count clock TCLK 1: External count clock ETR;
1	Reserved	Reserved bit
0	CTEN	Pulse Width Measurement Enable 0: forbidden; 1: Enable

14.5.3 Interrupt Flag Register (TIMx_IFR)

Offset address: 0x010

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													CA0F	Res.	UIF
													RO		RO

Bit	Symbol	Description
31:3	Res	Reserved bit
2	CA0F	Pulse Width Measurement Interrupt Flag
1	Res.	Keep
0	UIF	Overflow interrupt flag

14.5.4 Interrupt Flag Clear Register (TIMx_ICLR)

Offset address: 0x014

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
Reserved													CA0F	Res.	UIF
													R1W0		R1W0

Bit	Symbol	Description
31:3	Res.	Reserved bit
2	CA0F	Pulse width measurement interrupt flag clear, write 0 to clear
1	Res.	Keep
0	UIF	Overflow interrupt clear, write 0 to clear

14.5.5 Master-Slave Mode Control Register (TIMx_MSCR)

Offset address: 0x018

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
Resverved			IB0S	IA0S	Resvered			TS			Resvered				
			RW	RW				RW							

Bit	Symbol	Description
31:13	Res.	Reserved bit
12	IB0S	CH0B input selection 0： CH0B; 1: internal trigger TS selection signal; Note: When the PWM complementary output is automatically selected as the GATE port as the input of CH0B
11	IA0S	IA0 input selection 0： CH0A; 1： CH0A CH1A CH2A XOR(TIM3) 0： CH0A; 1： CH0A ETR GATE XOR(TIM0) Note: After setting to 1, any port change of the port will cause the input to change
10:8	Res.	Reserved bit
7:5	TS	Trigger selection 000: the signal ETFP after the filter phase selection of the port ETR; 001: Internal interconnection signal ITR0 010: internal interconnection signal ITR1; 011: internal interconnection signal ITR2; 100: internal interconnection signal ITR3; 101: edge signal of port CH0A; 110: Filtered phase-selected signal IAFP of port CH0A 111: signal IBFP after filter phase selection of port CH0B;
4:0	Res.	Reserved bit

14.5.6 Output Control Filtering (TIMx_FLTR)

Offset address: 0x01C

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETP	FLTET			Resvered											
RW	RW														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Resvered									FLTB0			Resvered	FLTA0		
									RW				RW		

Bit	Symbol	Description
31	ETP	ETR input phase selection 0: same phase; 1: reverse input;
30:28	FLTET	ETR filter control Filter settings 0xx: Filter invalid 100: pclk 3 continuous effective; 101: pclk/4 3 continuous effective 110: pclk/16 3 continuous effective; 111: pclk/64 3 continuous effective
27:7	Resvered	Reserved bit
6:4	FLTB0	CHB input filter control; Filter settings 0xx: Filter invalid 100: pclk 3 continuous effective; 101: pclk/4 3 continuous effective 110: pclk/16 3 continuous effective; 111: pclk/64 3 continuous effective
3	Resvered	Reserved bit
2:0	FLTA0	CHA input filter control Filter settings 0xx: Filter invalid 100: pclk 3 continuous effective; 101: pclk/4 3 continuous effective 110: pclk/16 3 continuous effective; 111: pclk/64 3 continuous effective

14.5.7 Control Register (TIMx_CR0)

Offset address: 0x024;

Reset value: 0x0000 3000

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							CIEA	Reserved		CSB	CSA	Reserved			
							RW			RW	RW				

Bit	Symbol	Description
31:9	Resvered	Reserved bit
8	CIEA	CIEA pulse width measurement complete interrupt enable 0: Prohibited 1: Enable
7:6	Resvered	Reserved bit
5	CSB	The B channel capture/compare function is selected, and the CSB needs to be set to 1 when the pulse width measurement is filtered using channel B 0: Comparison mode 1: Capture mode
4	CSA	The A channel captures/compares function selection, and the CSA needs to be set to 1 when the pulse width measurement is filtered using channel A 0: Comparison mode 1: Capture mode
3:0	Resvered	Reserved bit

14.5.8 Compare Capture Register (TIMx_CCR0A)

Offset address: 0x03C

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
CCR0A															
R															

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15:0	CCR0A	Pulse Width Measurement Results

14.6Mode 2,3 register description

14.6.1 16 -bit Mode Reload Register (TIMx_ARR)

Offset address: 0x000(0X038 dummy address)

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															
RW															

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15:0	ARR	Reload register / period register with cache function When the counter is not enabled or the cache is not enabled, the cache register can be updated immediately

14.6.2 16 -bit Mode Count Register (TIMx_CNT)

Offset address: 0x004

Reset value: 0x0000 0000

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

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

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15:0	CNT	reload timer Count value register

Note: When using PWM to compare the output, the initial CNT value needs to be less than the value of ARR.

14.6.3 Control Register(TIMx_M23CR)

Offset address: 0x00C

Reset value: 0x0060 0008

31	30	29	28	27	26	25	24
Resvered				DIR	BG	UG	TG
				RW/RO	W1	W1	W1

23	22	21	20	19	18	17	16
OCCE	CIS		BIE	TIE	TDE	URS	OCCS
RW	RW		RW	RW	RW	RW	RW

15	14	13	12	11	10	9	8
CSG	Oneshot	Mode		UDE	UIE	CFG	CRG
RW	RW	RW		RW	RW	RW	RW

7	6	5	4	3	2	1	0
BUFPEN	PRS			Pwm2s	CT	Comp	CTEN
RW	RW			RW	RW	RW	RW

Bit	Symbol	Description
31:28	Reserved	Reserved bit
27	DIR	Counting direction, can only be written in sawtooth wave mode. Read-only in other modes, invalid for writing 0: count up 1: count down Note: DIR is automatically cleared to 0 when switching from other modes to center-aligned mode. DIR is automatically cleared in reset mode by software event update and external trigger in slave mode.
26	BG	Software brake, automatic reset Write 1 to generate software brake; Writing 0 is invalid
25	UG	Software update, automatic reset Write 1 to generate software update; Writing 0 is invalid Initialize the counter and update the cache register to the corresponding register (cache enable), the pre-divider counter will also be cleared.
24	TG	Triggered by software, automatically cleared, it needs to be triggered under the trigger mode SMS=2 and mode=2/3. Write 1 to generate software trigger; Writing 0 is invalid
23	OCCE	OCREF clear enable 1: OCREF_CLR signal can clear OCREF output 0: OCREF output is not affected by OCREF_CLR
22:21	CIS	Center-aligned A compare interrupt mode (B compare interrupt is controlled separately in CRx register CISB) 00: no interrupt, 01: rising edge interrupt, 10: Falling edge interrupt, 11: Both the upper and lower edges are interrupted
20	BIE	Brake Interrupt Enable 1: interrupt enable 0: interrupt disabled
19	TIE	Trigger interrupt enable 1: interrupt enable 0: interrupt disabled
18	TDE	Trigger DMA enable 1: interrupt enable 0: interrupt disabled

17	URS	Update source 0: overflow / underflow / software update UG/ slave mode reset; 1: overflow / underflow
16	OCCS	OCREF Clear Source Selection 0: Voltage comparator VC output, VC selection is set in VCx_OUTCFG register 1: ETR port filters the signal after phase selection When OCCE is valid, OC_clr can clear the comparison output signal of OCREF to zero, (valid when OCMx>1), and continue to compare the output after the next uev event
15	CSG	GATE capture / comparison selection in PWM complementary mode; (only valid in PWM complementary output) Use CCR0B as compare or capture channel for GATE 1: capture; 0: Compare
14	Oneshot	Single trigger mode selection 1: Timer stops after an event update occurs. 0: loop count
13:12	MODE	Operating mode 00 Timer mode 0; 01 PWC mode 10 sawtooth wave mode; 11 triangle wave mode
11	UDE	update DMA enable 1: Enable update triggered DMA 0: disable update trigger DMA
10	UIE	UIE update interrupt enable 1: enable update interrupt 0: disable update interrupt
9	CFG	When GATE is used as a capture input, the falling edge capture is effectively controlled (only valid when PWM complementary output) 1: Falling edge capture is valid 0: Falling edge capture is invalid
8	CRG	GATE is used as a capture input, the rising edge capture is effectively controlled (only valid when PWM complementary output) 1: The upper edge capture is valid 0: The rising edge capture is invalid
7	BUFPEN	reload cache enable 1: The period cache is enabled, and the period value will not be affected until the next event is updated after writing. 0: The period cache is invalid, and the period value will be affected immediately after writing
6:4	PRS	Internal clock frequency division selection 000: 1; 001: 2; 010: 4; 011: 8; 100: 16; 101: 32; 110: 64; 111: 256;
3	PWM2S	OCREFA two-point comparison selection (default is 1) 0: Dual-point comparison is enabled, use CCRA, CCRB comparison to control OCREFA output 1: Single-point compare enable, only use CCRA compare to control OCREFA output Note: OCREFB is not affected, still use CCRB to control OCREFB output
2	CT	Counting clock selection 0: Internal count clock TCLK 1: External count clock ETR;
1	Comp	PWM complementary output mode selection 0: independent PWM output 1: Complementary PWM output
0	CTEN	Timer enable 0: forbidden; 1: Enable It can be enabled by external trigger, and this bit is automatically cleared at the end of oneshot mode

14.6.4 Interrupt Flag Register (TIMx_IFR)

Offset address: 0x010

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
TIF	BIF	CB2E	CB1E	CB0E	CA2E	CA1A	CA0E	CB2F	CB1F	CB0F	CA2F	CA1F	CA0F	Res.	UIF
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO		RO

Reset value: 0x0000 0000

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15	TIF	trigger interrupt flag
14	BIF	Brake Interrupt Flag
13	CB2E	Channel CH2B capture data loss flag 0: no data loss; 1: data loss (only TIM3 exists)
12	CB1E	Channel CH1B Capture Data Loss Flag 0: no data loss; 1: data loss (only TIM3 exists)
11	CB0E	Channel CH0B Capture Data Loss Flag 0: no data loss; 1: data loss
10	CA2E	Channel CH2A Capture Data Loss Flag 0: no data loss; 1: data loss (only TIM3 exists)
9	CA1E	Channel CH1A Capture Data Loss Flag 0: no data loss; 1: data loss (only TIM3 exists)
8	CA0E	Channel CH0A Capture Data Loss Flag 0: no data loss; 1: data loss
7	CB2F	Channel CH2B capture / compare match flag 0: not happening 1: happening (only TIM3 exists)
6	CB1F	Channel CH1B capture / compare match flag 0: not happening 1: happening (only TIM3 exists)
5	CB0F	Channel CH0B capture / compare match flag 0: Does not occur 1: Occurs
4	CA2F	Channel CH2A capture / compare match flag 0: not happening 1: happening (only TIM3 exists)
3	CA1F	Channel CH1A capture / compare match flag 0: not happening 1: happening (only TIM3 exists)
2	CA0F	Channel CH0A capture / compare match flag 0: Does not occur 1: Occurs
1	Res.	Reserved bit
0	UIF	Event Update Interrupt Flag 0: Does not occur 1: Occurs

14.6.5 Interrupt Flag Clear Register (TIMx_ICLR)

Offset address: 0x014

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
TIF	BIF	CB2E	CB1E	CB0E	CA2E	CA1A	CA0E	CB2F	CB1F	CB0F	CA2F	CA1F	CA0F	Res.	UIF
R1W0	R1W0	R1W0	R1W0	R1W0	R1W0	R1W0	R1W0	R1W0	R1W0	R1W0	R1W0	R1W0	R1W0		R1W0

Bit	Symbol	Description
31:16	REV	Reserved bit
15	TIF	Trigger interrupt flag to clear, write 0 to clear
14	BIF	Brake interrupt flag is cleared, write 0 to clear
13	CB2E	Channel CH2B capture data loss flag clear, write 0 to clear (only TIM3 exists)
12	CB1E	Channel CH1B capture data loss flag clear, write 0 to clear (only TIM3 exists)
11	CB0E	Channel CH0B capture data loss flag clear, write 0 to clear
10	CA2E	Channel CH2A captures the data loss flag to clear, write 0 to clear (only TIM3 exists)
9	CA1E	Channel CH1A capture data loss flag clear, write 0 to clear (only TIM3 exists)
8	CA0E	Channel CH0A capture data loss flag clear, write 0 to clear
7	CB2F	Channel CH2B capture / compare match flag clear, write 0 to clear (only TIM3 exists)
6	CB1F	Channel CH1B capture / compare match flag clear, write 0 to clear (only TIM3 exists)
5	CB0F	Channel CH0B capture / compare match flag clear, write 0 to clear
4	CA2F	Channel CH2A capture / compare match flag clear, write 0 to clear (only TIM3 exists)
3	CA1F	Channel CH1A capture / compare match flag clear, write 0 to clear (only TIM3 exists)
2	CA0F	Channel CH0A capture / compare match flag clear, write 0 to clear
1	Res.	Keep
0	UIF	Event update interrupt clear, write 0 to clear

14.6.6 Master-Slave Mode Control Register (TIMx_MSCR)

Offset address: 0x018

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
Resverved			IB0S	IA0S	SMS			TS			MSM	CCDS	MMS		
			RW	RW	RW			RW			RW	RW	RW		

Bit	Symbol	Description
31:13	Res	Reserved bit
12	IB0S	CH0B input selection 0: CH0B; 1: internal trigger TS selection signal; Note: When the PWM complementary output is automatically selected as the GATE port as the input of CH0B
11	IA0S	IA0 input selection 0: CH0A; 1: CH0A CH1A CH2A XOR(TIM3) 0: CH0A; 1: CH0A ETR GATE XOR(TIM0) Note: After setting to 1, any port change of the port will cause the input to change
10:8	SMS	Select from mode function 000: use the internal clock; 001: Reset function; 010: Trigger mode; 011: External Clock Mode 100: Orthogonal encoding counting mode 1; 101: Orthogonal encoding counting mode 2; 110: Orthogonal encoding counting mode 3; 111: Gating function
7:5	TS	Trigger selection 000: the signal ETFP after the filter phase selection of the port ETR; 001: Internal interconnection signal ITR0 010: internal interconnection signal ITR1; 011: internal interconnection signal ITR2; 100: internal interconnection signal ITR3; 101: edge signal of port CH0A; 110: Filtered phase-selected signal IAFP of port CH0A 111: signal IBFP after filter phase selection of port CH0B;
4	MSM	master-slave selection 0: no delay 1: Delay enabled, so that the main sending counter starts at the same time. Note: When using the trigger mode, the slave mode is set to 0, and the master mode is set to 1, so that the master and slave counts can be started at the same time
3	CCDS	DMA comparison trigger selection in comparison mode; 0: compare match triggers DMA; 1: Compare match does not trigger DMA, event update instead of compare match triggers DMA
2:0	MMS	Master mode output select for internal interconnection to ITRx of other timers

Timer 0/1/2	timer 3
000: software update UG, write CR.UG 001: Timer enable CTEN 010: Timer event updates UEV; 011: compare match select output CMPSO; 100: Timer comparison parameter output OCREF0A 101: Timer comparison parameter output OCREF0B 110: Timer comparison parameter output OCREF0B 111: Timer comparison parameter output OCREF0B	000: software update UG, write CR.UG 001: Timer enable CTEN 010: Timer event updates UEV; 011: compare match select output CMPSO; 100: Timer comparison parameter output OCREF0A 101: Timer comparison parameter output OCREF1A 110: Timer comparison parameter output OCREF2A 111: Timer comparison parameter output OCREF0B

14.6.7 Output Control / Input Filtering (TIMx_FLTR)

Offset address: 0x01C

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETP	FLTET			BKP	FLTBK			CCPB2	OCMB2			CCPA2	OCMA2		
-	-			-	-			-	FLTB2			-	FLTA2		
RW	RW			RW	RW			RW	RW			RW	RW		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCPB1	OCMB1			CCPA1	OCMA1			CCPB0	OCMB0			CCPA0	OCMA0		
-	FLTB1			-	FLTA1			CCPB0	FLTB0			CCPA0	FLTA0		
RW	RW			RW	RW			RW	RW			RW	RW		

Bit	Symbol	Description
31	ETP	ETR input phase selection 0: same phase; 1: reverse input;
30:28	FLTET	ETR filter control Filter settings 0xx: Filter invalid 100: pclk 3 continuous effective; 101: pclk/4 3 continuous effective 110: pclk/16 3 continuous effective; 111: pclk/64 3 continuous effective
27	BKP	Brake BK input phase selection 0: same phase; 1: reverse input;
26:24	FLTBK	Brake input filter control Filter settings 0xx: Filter invalid 100: pclk 3 continuous effective; 101: pclk/4 3 continuous effective 110: pclk/16 3 continuous effective; 111: pclk/64 3 continuous effective Note: In order to ensure the PWM output setting of OCMB0, GATE is used as the capture input in PWM complementary mode, and the filter setting is invalid.
23	CCPB2	Comparison function: CH2B channel comparison output phase control 0: normal output; 1: Reverse output
22:20	OCMB2 FLTB2	Comparison function: CH2B channel comparison control; refer to OCMB0 Capture function: CH2B input channel filter setting, refer to FLTBK
19	CCPA2	Comparison function: CH2A channel comparison output phase control 0: normal output; 1: Reverse output
18:16	OCMA2 FLTA2	Comparison function: CH2A channel comparison control; refer to OCMB0 Capture function: CH2A input channel filter setting, refer to FLTBK
15	CCPB1	Comparison function: CH1B channel comparison output phase control 0: normal output; 1: Reverse output
14：12	OCMB1 FLTB1	Comparison function: CH1B channel comparison control; refer to OCMB0 Capture function: CH1B input channel filter setting, refer to FLTBK
11	CCPA1	Comparison function: CH1A channel comparison output phase control 0: normal output; 1: Reverse output
10:8	OCMA1 FLTA1	Comparison function: CH1A channel comparison control; refer to OCMB0 Capture function: CH1A input channel filter setting, refer to FLTBK
7	CCPB0	Compare function: output compare mode CCPBx compare output CHBx port polarity control 0: normal output; 1: Reverse output Encode Count and Slave Mode Gating Function: Input Phase Control CCPB0 slave mode gating, reset, external trigger, external clock using CH0B port input polarity control 0: normal input;

		1: reverse input
6:4	OCMB0 FLTB0	Comparison function: CH0B channel comparison control 000: forced to 0 001: Forced to 1 010: Forced to 0 when comparing matches 011: Forced to 1 when comparing matches 100: flip when comparison matches 101: Output a high level for one count period when comparing matches 110: PWM mode 1 Single point comparison: When counting up, CNT<CCRxy outputs high, and when counting down, CNT>CCRxy outputs low level Two-point comparison: 1) Counting CCRxA<CNT≤CCRxB output on the sawtooth wave is low level 2) Counting under sawtooth wave CCRxA<CNT≤CCRxB output is high level 3) Triangular wave up counting CNT<CCRxA output high, down counting CNT>CCRxB is low level 111: PWM mode 2 Single point comparison: When counting up, CNT<CCRxy output is low, when counting down, CNT>CCRxy output is high level Two-point comparison: 1) Counting CCRxA≤CNT<CCRxB output on the sawtooth wave is high level 2) Counting under sawtooth wave CCRxA≤CNT<CCRxB output is low level 3) Triangular wave up count CNT<CCRxA output low, down count CNT>CCRxB is high level Capture function: CH0B input channel filter setting, refer to FLTBK
3	CCPA0	Compare function: CCPAx compare output CHAx port polarity control 0: normal output; 1: Reverse output Encode Count and Slave Mode Gating Function: Input Phase Control CCPA0 slave mode gating, reset, external trigger, external clock using CH0A port input polarity control 0: normal input; 1: reverse input
2:0	OCMA0 FLTA0	Comparison function: A channel comparison control; refer to OCMB0 Capture function: CH0A input channel filter setting, refer to FLTBK

14.6.8 ADC Trigger Control Register (TIMx_ADTR)

Offset address: 0x020

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								ADTE	CMB2E	CMB1E	CMB0E	CMA2E	CMA1E	CMA0E	UEVE
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Symbol	Description
31:8	Resvered	Reserved bit
7	ADTE	Enable ADC trigger global control 1: Enable 0: Prohibited
6	CMB2E	Channel 2B comparison match triggers ADC enable, only TIM3 exists 1: Enable 0: Prohibited
5	CMB1E	Channel 1B comparison match triggers ADC enable, only TIM3 exists 1: Enable 0: Prohibited
4	CMB0E	Channel 0B compare match trigger ADC enable 1: Enable 0: Prohibited
3	CMA2E	Channel 2A comparison match triggers ADC enable, only TIM3 exists 1: Enable 0: Prohibited
2	CMA1E	Channel 1A comparison match triggers ADC enable, only TIM3 exists 1: Enable 0: Prohibited
1	CMA0E	Channel 0A compare match trigger ADC enable 1: Enable 0: Prohibited
0	UEVE	Event update triggers ADC enable 1: Enable 0: Prohibited

14.6.9 Channel 0 Control Register (TIMx_CRCH0)

Offset address: 0x024

Reset value: 0x0000 3000

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
CCGB	CCGA	CISB		CDEB	CDEA	CIEB	CIEA	-	-	CSB	CSA	CFB	CRB	CFA	CRA
CCGB	CCGA	CISB		CDEB	CDEA	CIEB	CIEA	BUFEB	BUFEA	CSB	CSA	bksb		bksa	
RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	

Bit	Symbol	Description
31:16	Resvered	Reserved bit
15	CCGB	Capture comparison B is triggered by software and automatically cleared by hardware. Only interrupts are generated in compare mode; In capture mode an interrupt is generated and the counter value is captured into the capture register.
14	CCGA	Capture comparison A is triggered by software and automatically cleared by hardware. Only interrupts are generated in compare mode; In capture mode an interrupt is generated and the counter value is captured into the capture register.
13:12	CISB	B channel compare match setting 00 no match; 01 rise match; 10 fall match; 11 double match
11	CDEB	B capture compare trigger DMA enable 0: disable 1: enable
10	CDEA	A capture compare trigger DMA enable 0: Prohibited 1: Enable
9	CIEB	B capture compare trigger interrupt enable 0: Prohibited 1: Enable
8	CIEA	A capture compare trigger interrupt enable 0: Prohibited 1: Enable
7	BUFEB	Compare Function: B Compare Cache Enable Control 0: Prohibited 1: Enable
6	BUFEA	Compare Function: A Compare Cache Enable Control 0: Prohibited 1: Enable
5	CSB	B channel capture / compare function selection 0: Compare Mode 1: capture mode
4	CSA	A channel capture / compare function selection 0: Compare Mode 1: capture mode
3	CFB	B channel falling edge capture enable 0: Prohibited 1: Enable
2	CRB	B channel rising edge capture enable 0: Prohibited 1: Enable
3:2	BKSB	B channel comparison function output brake level control 00: High impedance output 01: no effect on output 10: forced output low level 11: forced output high level;
1	CFA	A channel falling edge capture enable 0: Prohibited 1: Enable
0	CRA	A channel rising edge capture enable 0: Prohibited 1: Enable
1:0	BKSA	A channel comparison function output brake level control 00: High impedance output

		01: no effect on output 10: forced output low level 11: forced output high level;

14.6.10 Channel 1/2 Control Registers (TIM3_CRCH1/2) (TIM3 present only)

Offset address:

TIM3_CRCH1: 0x028;

TIM3_CRCH2: 0x02C

Reset value: 0x0000 3000

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
CCGB	CCGA	CISB		CDEB	CDEA	CIEB	CIEA	-	-	CSB	CSA	CFB	CRB	CFA	CRA
CCGB	CCGA	CISB		CDEB	CDEA	CIEB	CIEA	BUFEB	BUFEA	CSB	CSA	bksb		bksa	
RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	

Bit	Symbol	Description
31:16	Resvered	Reserved bit
15	CCGB	Capture comparison B is triggered by software and automatically cleared by hardware. Only interrupts are generated in compare mode; In capture mode an interrupt is generated and the counter value is captured into the capture register.
14	CCGA	Capture comparison A is triggered by software and automatically cleared by hardware. Only interrupts are generated in compare mode; In capture mode an interrupt is generated and the counter value is captured into the capture register.
13:12	CISB	B channel compare match setting 00 no match; 01 rise match; 10 fall match; 11 double match
11	CDEB	B capture compare trigger DMA enable 0: disable 1: enable
10	CDEA	A capture compare trigger DMA enable 0: Prohibited 1: Enable
9	CIEB	B capture compare trigger interrupt enable 0: Prohibited 1: Enable
8	CIEA	A capture compare trigger interrupt enable 0: Prohibited 1: Enable
7	BUFEB	Compare Function: B Compare Cache Enable Control 0: Prohibited 1: Enable
6	BUFEA	Compare Function: A Compare Cache Enable Control 0: Prohibited 1: Enable
5	CSB	B channel capture / compare function selection 0: Compare Mode 1: capture mode
4	CSA	A channel capture / compare function selection 0: Compare Mode 1: capture mode
3	CFB	B channel falling edge capture enable 0: Prohibited 1: Enable
2	CRB	B channel rising edge capture enable 0: Prohibited 1: Enable
3:2	BKSB	B channel comparison function output brake level control 00: High impedance output 01: keep the previous output l 10: forced output low level 11: forced output high level;
1	CFA	A channel falling edge capture enable 0: Prohibited 1: Enable

0	CRA	A channel rising edge capture enable 0: Prohibited 1: Enable
1:0	BKSA	A channel comparison function output brake level control 00: High impedance output 01: keep the previous output l 10: forced output low level 11: forced output high level;

14.6.11 Dead Time Register (TIMx_DTR)

Offset address: 0x030

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
	VCE	Safeen	MOE	AOE	BKE	DTEN	Bksel	DTR							
	RW	RW	RW	RW	RW	RW	RW	RW							

Bit	Symbol	Description
31:15	Reserved	Reserved bit
14	VCE	VC Brake enable 0: Prohibited 1: Enable
13	Safeen	Safety brake enable (osc fail,brown down,lockup) 0: Prohibited 1: Enable
12	MOE	PWM output enable 0: Prohibited 1: Enable
11	AOE	PWM output is automatically enabled 0: Prohibited 1: Enable
10	BKE	Brake enable 0: Prohibited 1: Enable
9	DTEN	Dead zone control enable 0: Prohibited 1: Enable
8	bksel	Brake selection 0: Use own brake control; 1: TIM1/2 uses the brake control of TIM0 Note: TIM0/TIM3 selection is invalid.
7:0	DTR	Dead Time Register DTR[7] =0 T=DTR[6:0]+2 2-129 step=1 DTR[7:6] =10 T={DTR[5:0]+64}*2 +2 130-256 step=2 DTR[7:5] =110 T={DTR[4:0]+32}*8 +2 258-506 step=8 DTR[7:5] =111 T={DTR[4:0]+32}*16 +2 514-1010 step=16

14.6.12 Repeat cycle setting value (TIMx_RCR)

Offset address: 0x034

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								RCR							
								RW							

Bit	Symbol	Description
31:8	Reserved	Reserved bit
7:0	RCR	Recurrence count value Set RCR+1 cycles to generate an event update after overflow and underflow. When the counter overflows or underflows, the internal RCR_CNT is decremented by 1. When the count reaches zero, RCR_CNT reloads the value of RCR and generates an event update UEV signal

14.6.13 Channel 0 Compare Capture Register (TIMx_CCR0A/B)

Offset address:

TIMx_CCR0A : 0x03C;

TIMx_CCR0B : 0x040

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
CCR0y															
RW															

Reset value: 0x0000 0000

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15:0	CCR0y	Compare the capture register, the comparison has a cache function (y=A,B)

14.6.14 Channel 1/2 compare capture register (TIM3_CCR1/2 A/B) (TIM3 exists only)

Offset address:

TIM3_CCR1A: 0x044

TIM3_CCR1B: 0x048

TIM3_CCR2A: 0x04C

TIM3_CCR2B: 0x050

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
CCRxy															
RW															

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15:0	CCRxy	Compare the capture register, the comparison has a cache function (x=1,2; y=A,B)

15 Low Power Timer (LPTIM)

15.1 Introduction to LPTimer

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

LPTimer supports a maximum prescaler frequency of 256. There are two low-power timers in the system, which can be cascaded internally.

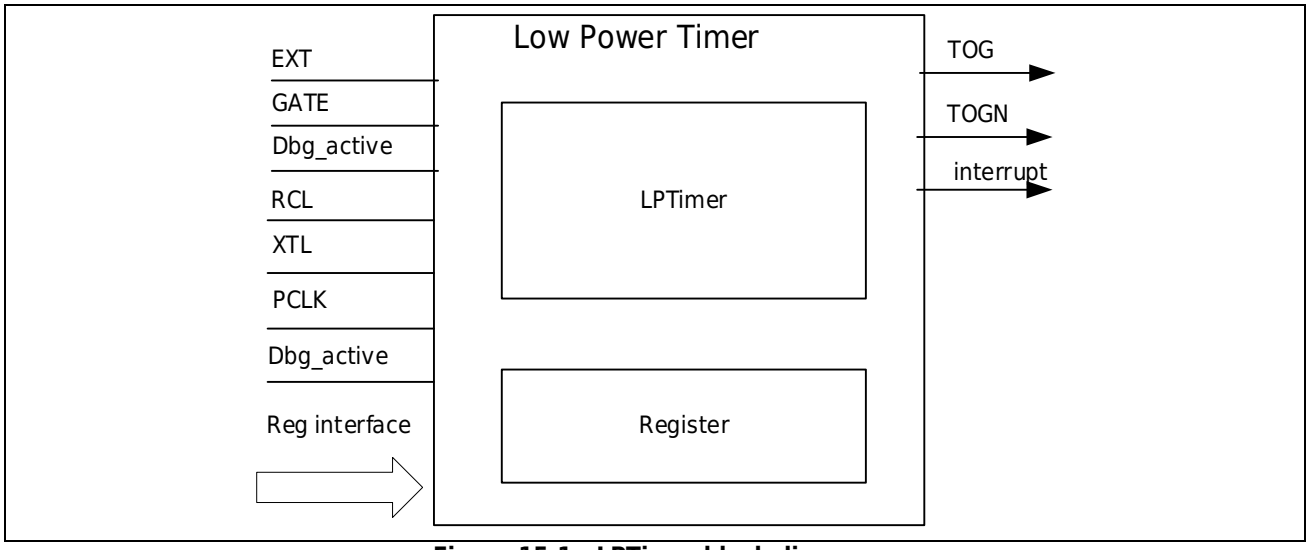


Figure 15-1 LPTimer block diagram

15.2 LPTimer Function Description

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

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

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

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

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

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

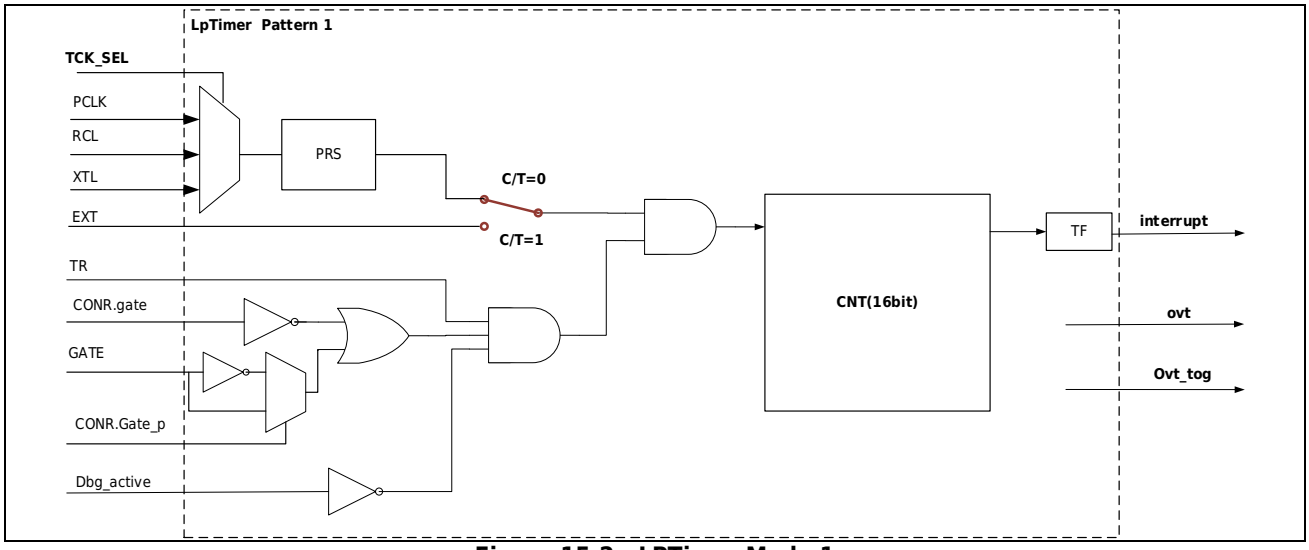


Figure 15-2 LPTimer Mode 1

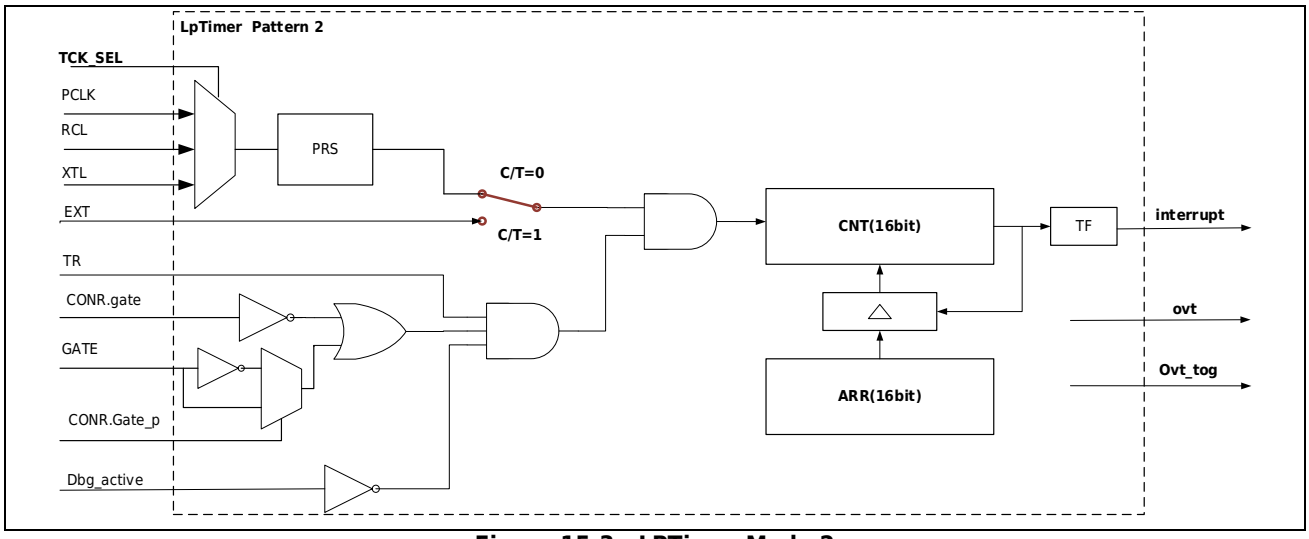


Figure 15-3 LPTimer Mode 2

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

15.2.1 Counting function

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

15.2.2 Timing function

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

The number of timing clock cycles is (0XFFFF-ARR+0X0001)

15.3LPTimer Interconnect

15.3.1 GATE interconnection

The GATE input can be directly input from the port through PX_SEL, or it can be connected to other modules or ports through the port function register GPIO_TIMGS.

When LPTIM_Gx=0x0, the gate control GATE input is the port input selected by PX_SEL. When LPTIM_Gx=0x1~0x7, it is connected to the input or output of other modules.

	LPTIM0_G		**LPTIM1_G**
000	PX_SEL	000	PX_SEL
001	LPUART0_RXD	001	UART0_RXD
010	LPUART1_RXD	010	LPUART0_RXD
011	VC0_OUT	011	VC1_OUT
100	VC1_OUT	100	VC2_OUT
101	PB03	101	PD05
110	PB05	110	PE07
111	PC00	111	PC06

15.3.2 EXT Interconnection

EXT input can be directly input from the port, or can be connected to other modules or ports through the selection of the port function register GPIO_TIMES.

When LPTIM_Ex=0x0, the external clock EXT input is the port input selected by PX_SEL. When LPTIM_Ex=0x1~0x7, it is connected to the input or output of other modules.

	LPTIM0_E		**LPTIM1_E**
000	PX_SEL	000	PX_SEL
001	PCNT_S0	001	UART1_RXD
010	LVD_OUT	010	LPUART1_RXD
011	VC0_OUT	011	VC0_OUT
100	VC1_OUT	100	VC2_OUT
101	PB04	101	LPTIM0_TOG
110	PB06	110	PE08
111	PC03	111	PC07

Set the external input LPTIM1_E of the timer of LPTIM1 to 0x5, select the toggle output LPTIM0_TOG of LPTIM0 as the external clock input, and set LPTIM1 as the external counting mode. In this way, two timers can be used in cascade.

15.3.3 Toggle Output Interconnects

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

15.4LPTimer register description

LPTIMER0 base address 0X40000F00

LPTIMER1 base address 0X40000F40

Table 15-1 LPTimer register list

Register	Offset address	Description
LPTIMx_CNT	0X000	LPTimer count value read-only register
LPTIMx_ARR	0X004	LPTimer reload register
LPTIMx_CR	0X00C	LPTimer Control Register
LPTIMx_IFR	0X010	LPTimer interrupt flag
LPTIMx_ICLR	0X014	LPTimer Interrupt Clear Register

15.4.1 Counter Count Value Register (LPTIMx_CNT)

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
CNT[15:0]															
RO															

Bit	Symbol	Description
31:16	Reserved	Reserved bit, read as 0
15:0	CNT	Low Power Timer Count Value Read Only Register

15.4.2 Reload Register (LPTIMx_ARR)

Offset address: 0x004

Reset value: 0x0000 0000

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

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

Bit	Symbol	Description
31:16	Reserved	Reserved bit, read as 0
15:0	ARR	Low Power Timer Reload Register CR.WT_FLAG needs to be read before writing ARR, and data can only be written when WT_FLAG is 1. WT2_FLAG will go low after writing the ARR register. Note: This value will be loaded into the timer counter when the timer is started or reloaded. The timer counter value cannot be changed by software.

15.4.3 Control Register (LPTIMx_CR)

Offset address: 0x00C

Reset value: 0x0000 0008

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

Bit	Symbol	Description
31:14	Reserved	Reserved bit, read as 0
13:11	PRS	prescaler (cannot be changed while the timer is running) 000 1 001 2 010 4 011 8 100 16 101 32 110 64 111 256
10	IE	Interrupt enable control, enable interrupt after writing 1
9	GATE_P	GATE polarity control, the default high level gate is valid, after setting to 1, the low level is valid
8	GATE	Timer Gating 0: No gate control, the timer works when TR=1; 1: works when the gate is 1 and TR=1;
7	WT_FLAG	WT, write synchronization flag 0: Synchronizing, writing ARR is invalid at this time 1: Synchronization is complete, ARR can be changed at this time
6	Reserved	Reserved bit, read as 0
5:4	TCK_SEL	LPTimer clock selection 00:PCLK; 10:XTL; 11,RCL
3	TOG_EN	TOG output enable 0: TOG, TOGN output 0 at the same time 1: TOG, TOGN output signals with opposite phases. Available for buzzers.
2	CT	Counter / timer function selection 0: Timer function, the timer uses the clock selected by TCK_SEL to count. 1: Counter function, the counter uses the falling edge of the external input to count. The sampling clock uses the clock selected by TCK_SEL, and the external input clock is lower than 1/2 sampling clock.
1	MD	Timer working mode 0: mode 1 no reload mode 16 -bit counter / timer 1: Mode 2 auto-reload 16 -bit counter / timer
0	TR	Timer run control bit 0: timer stopped 1: Timer running

15.4.4 Interrupt Flag Register (LPTIMx_IFR)

Offset address: 0x010

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

Reset value: 0x0000 0000

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

15.4.5 Interrupt Flag Clear Register (LPTIMx_ICLR)

Offset address: 0x014

Reset value: 0x0000 0001

31:8	7	6	5	4	3	2	1	0
Reserved								TFC
								R1W0

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

16 Programmable Count Array (PCA)

16.1 Introduction to PCA

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

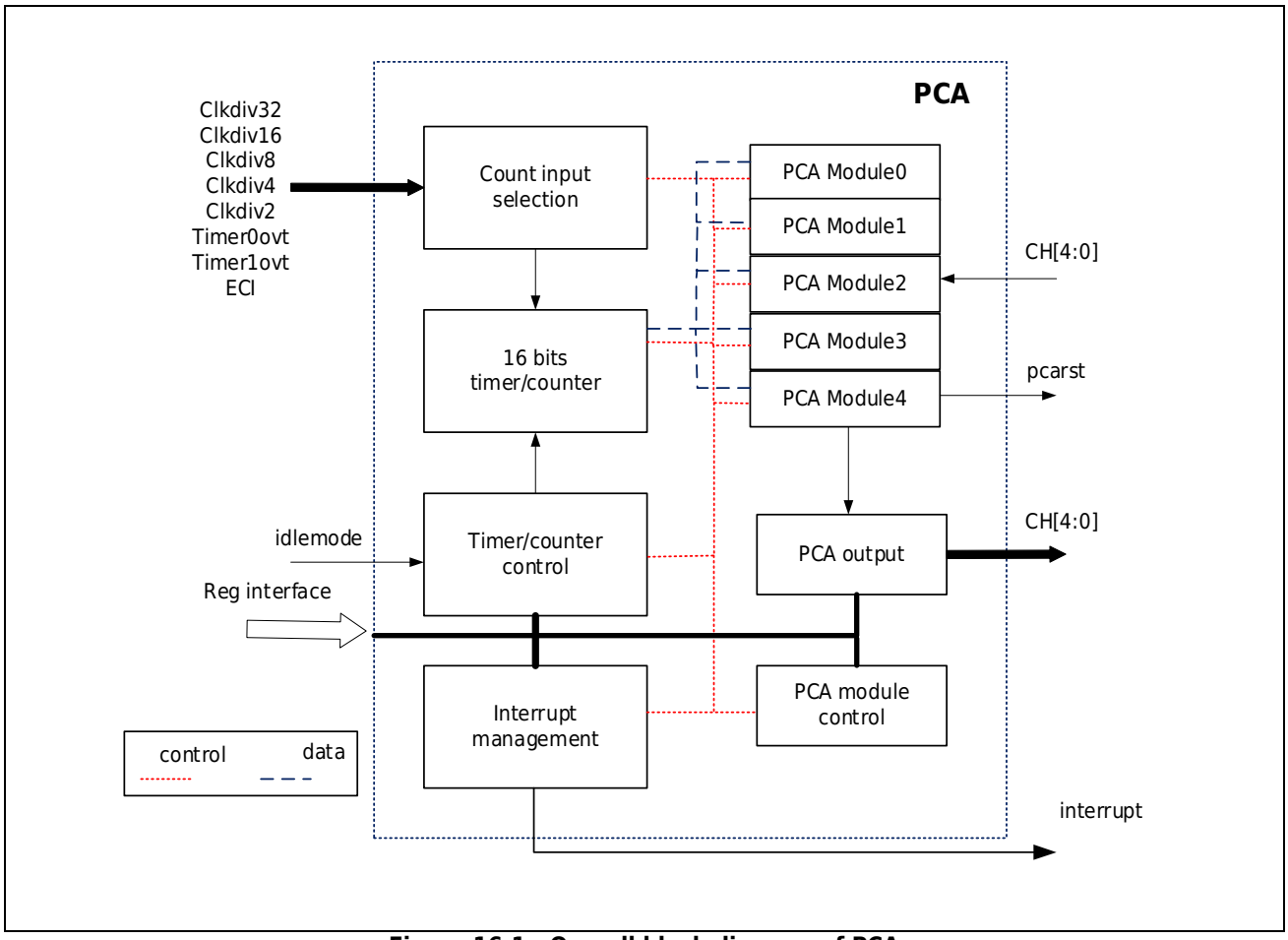


Figure 16-1 Overall block diagram of PCA

16.2 PCA function description

All five modules can be configured to work independently, and there are three working modes: edge-triggered capture, output comparison, and 8/16- bit pulse width modulation. Each module has its own function registers in the system controller, and these registers are used to configure the working mode of the module and exchange data with the module.

Each comparison / capture module is composed of a comparison / capture register group (CCAPx), a 16 -bit comparator, and various logic gate controls. The register group is used to store time or times, for external trigger capture conditions, or internal trigger comparison conditions. In 8 -bit PWM mode, the register (CCAPxL) is used to control the duty cycle of the output waveform, and

CCAPxH is an 8 -bit compare buffer. In 16 -bit PWM mode, CARR is used to control the period of PWM output, and the CCAPx register controls the duty cycle.

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

- 16 -bit capture mode.
- Compare mode: 16 -bit high-speed output, 16 -bit watchdog timer (module 4) or 16/8- bit PWM.
- Disabled.

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

16.2.1 PCA Timer / Counter

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

- The system clock PCLK is divided by 32.
- The system clock PCLK is divided by 16.
- The system clock PCLK is divided by 8.
- The system clock PCLK is divided by 4.
- The system clock PCLK is divided by 2.
- Timer 0 overflow. CNT is incremented each time Timer 0 overflows, thus providing a variable programming frequency input to the PCA.
- Timer 1 overflow. CNT is incremented each time Timer 1 overflows, thus providing a variable programming frequency input to the PCA.
- ECI. The CPU samples the PCA ECI every 4 PCLK clock cycles. When the sampling result changes from high to low each time, CL is automatically incremented by 1. Therefore, the highest ECI input frequency cannot be higher than 1/8 of the system clock PCLK to meet Sampling requirements.

Set the run controller (CCON.CR) to start the PCA timer / counter. When CMOD.CIDL is set to "1", the PCA timer / counter can continue to run in idle mode. The CPU can read the value of CNT at any time, but when the count is started (CCON.CR=1), in order to prevent counting errors, CNT is prohibited from being written.

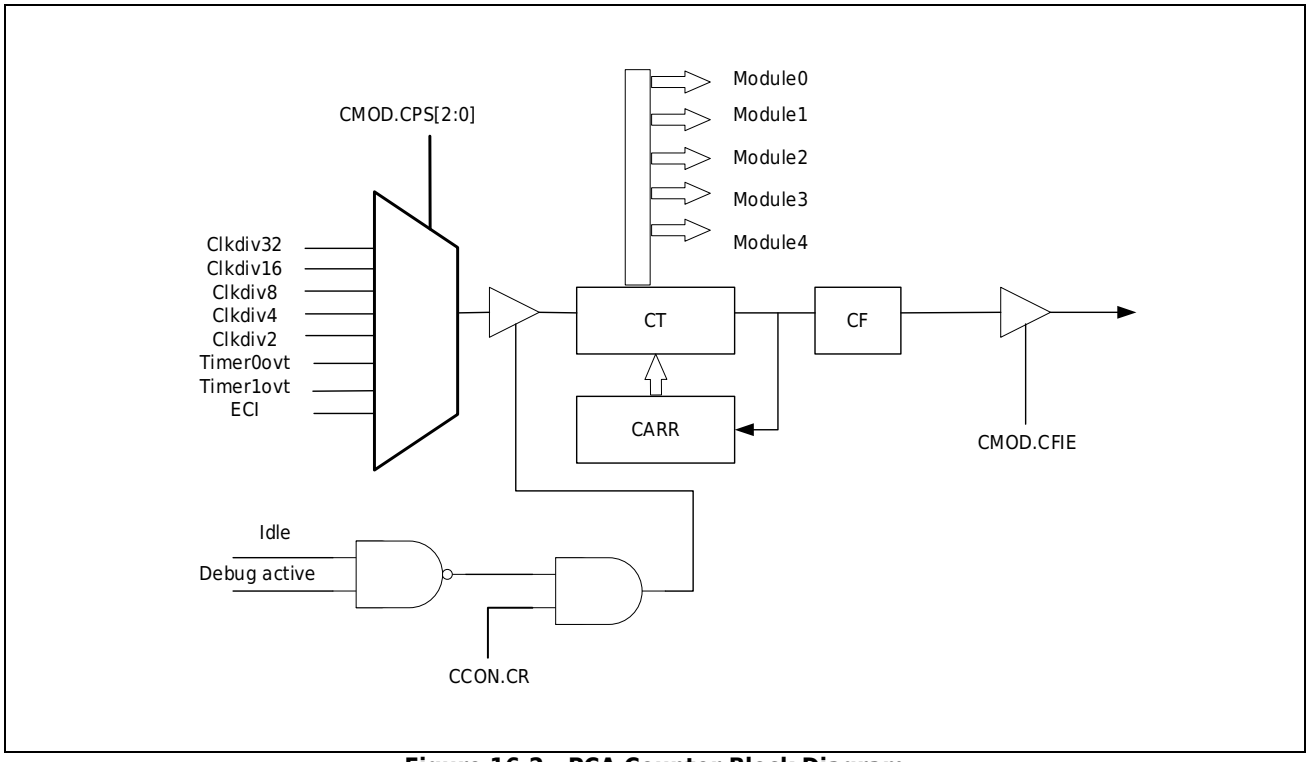


Figure 16-2 PCA Counter Block Diagram

16.2.1.1 16 -bit free count mode

After the counter counts to the maximum value of 0XFFFF and overflows, the value of the counter becomes 0 and starts counting up again. If you need to change the counting period value, you can stop PCA to change the initial value of the counter and continue counting. Can be used in PCA capture mode, 8 -bit PWM mode.

Setup process

1. Keep the EPMW of PCA_EPWM as 0
2. Set PCA_CMOD.CPS to select the count clock
3. PCA_CMOD.CFIE to enable count overflow interrupt as required
4. Clock PCA_CCON.CR starts the PCA counter
5. Overflow to change the initial value of the counter needs to stop the PCA counter

16.2.1.2 16 -bit reload count mode

When the counter counts to the same value as the register CARR, the value of the counter overflows and becomes 0, and continues to count up. It can be used in PCA capture mode and 16 -bit PWM mode.

Setup process

1. Set the EPMW of PCA_EPWM to 1
2. Set PCA_CARR to set the count period value
3. Set PCA_CMOD.CPS to select the count clock

4. PCA_CMOD.CFIE to enable count overflow interrupt as required
5. Clock PCA_CCON.CR starts the PCA counter

16.2.2 PCA capture function

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

A level transition on the pin causes the PCA to capture the value of the PCA counter / timer and load it into the corresponding module's 16 -bit capture / compare register (CCAPx). The CCAPMx.CAPP and CCAPMx.CAPN bits are used to select the type of level change that triggers the capture: low to high (positive edge), high to low (negative edge), or any change (positive or negative edge) along). When a capture occurs, the Capture / Compare Flag (CCFn) in CCON is set to logic '1' and an interrupt request is generated (if CCF interrupts are enabled). When the CPU turns to the interrupt service routine, the CCFn bit cannot be automatically cleared by hardware, and the user software can write the INTCL register to clear this flag bit. If the CCPMx.CAPP and CCAPMx.CAPN bits are both set to logic '1', you can directly read the state of the corresponding port pin to determine whether the capture is triggered by a rising edge or a falling edge.

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

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

Capture settings:

- When capturing on an external rising edge is required, CCPMx.CAPP = “1” and CCAPMx.CAPN = “0”
- When capturing on an external falling edge is required, CCPMx.CAPP = “0” and CCAPMx.CAPN = “1”
- When capture is required on external rising and falling edges, CCPMx.CAPP = "1" and CCAPMx.CAPN = "1"
- Configure capture interrupts and interrupt handlers as required.
- PCA counter to start according to the timer / counter.

Note:

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

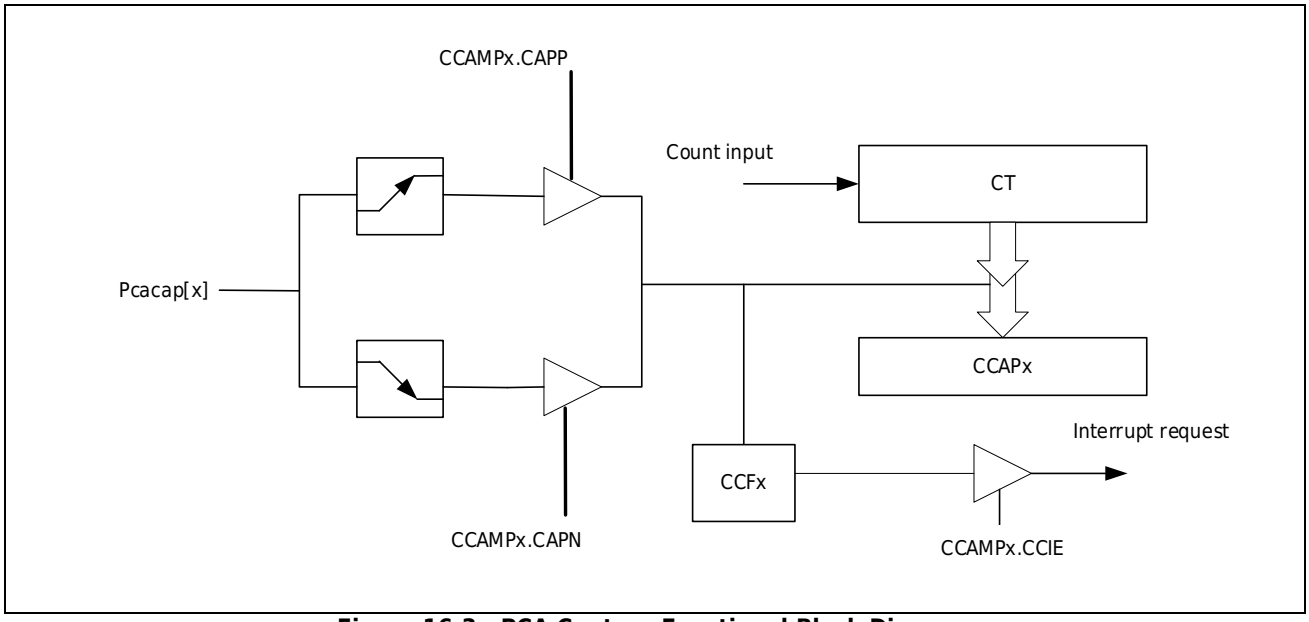


Figure 16-3 PCA Capture Functional Block Diagram

16.2.3 PCA comparison function

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

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

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

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

16.2.3.1 Compare Toggle Output Mode

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

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

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

Note:

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

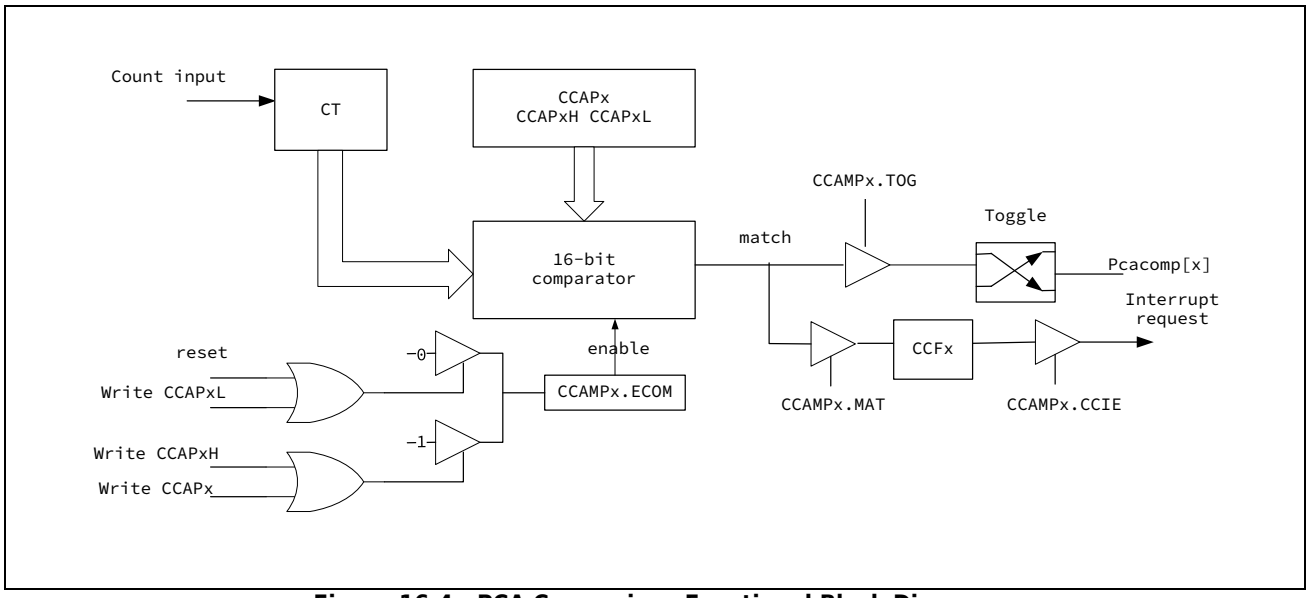


Figure 16-4 PCA Comparison Functional Block Diagram

16.2.3.2 PCA 16 -bit PWM function

In compare output PWM mode, whenever the value in the PCA counter matches the 16 -bit capture / compare register (CCAPx) of the module, the logic level on the CH[x] pin of the module PCA will change. When the counter overflows, the logic level on the CH[x] pin will be cleared. This can provide a set of 16 -bit PWM outputs.

To set the PWM mode of a compare / capture module, set the CCAPMx.ECOM, CCAPMx.MAT and CCAPMx.TOG bits and the EPWM register. PCA timer / counter and the compare / capture register (CCAPx) toggles the PCA 's CH[x] signal and sets the module's compare / capture flag (CCON.CCFx).

The user can also choose to generate an interrupt request, by setting the corresponding interrupt enable bit (CCAPMx.CCIE), when a match occurs, an interrupt request can be generated. Since the compare/capture flag interrupt cannot be cleared by hardware, the user must clear this flag in software.

Note: When using 16-bit PWM, it is recommended not to use the frequency division of PCA, otherwise the duty ratio output will differ by a maximum of one cycle.

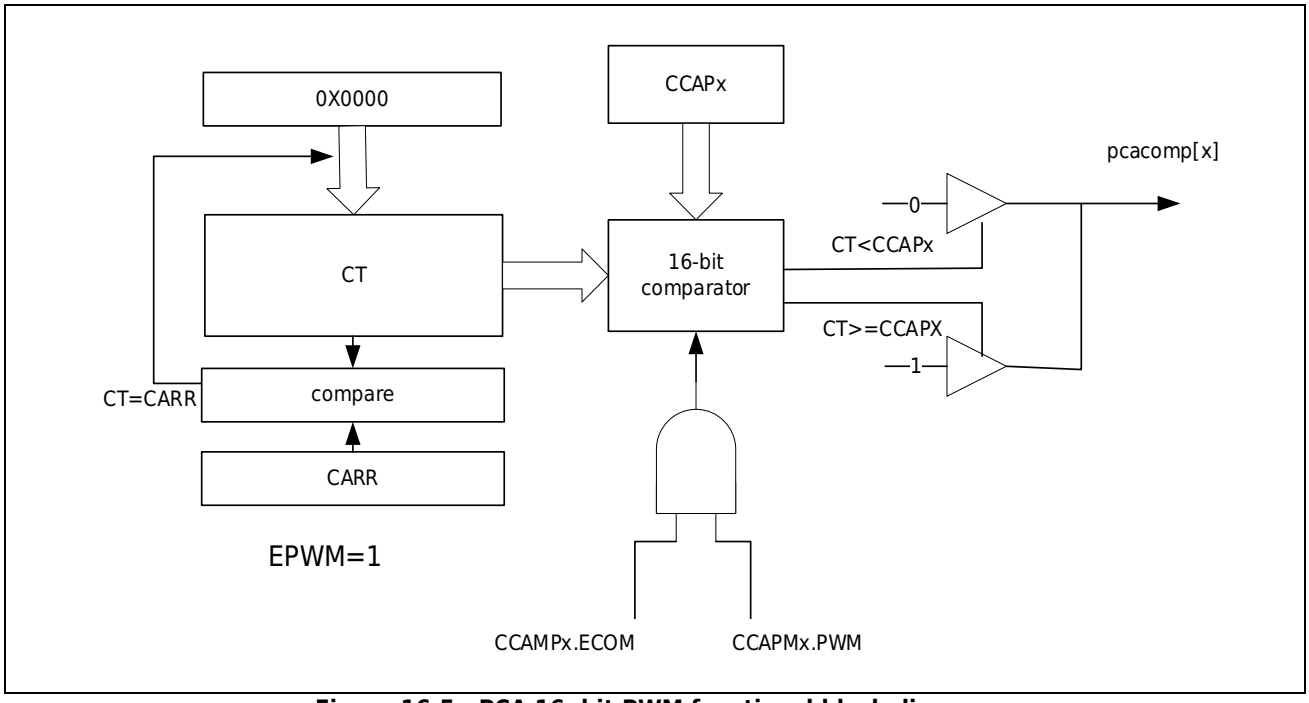


Figure 16-5 PCA 16 -bit PWM functional block diagram

16.2.3.3 WDT function of PCA module 4

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

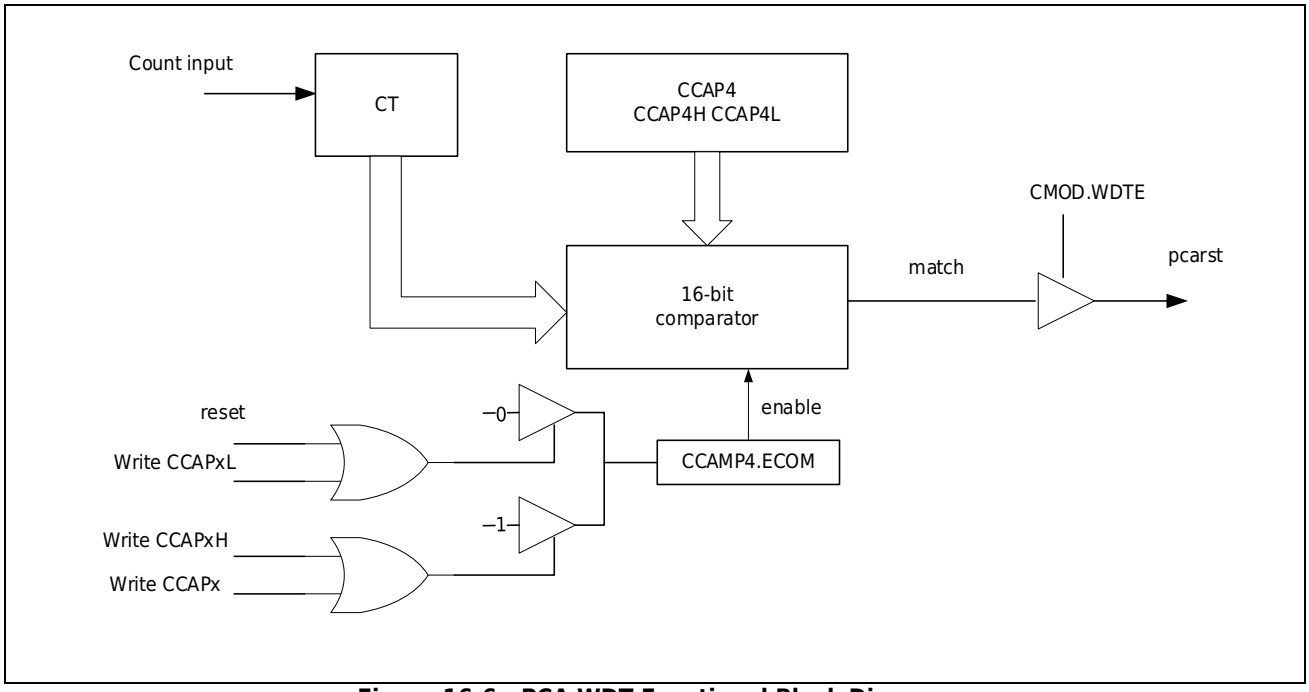


Figure 16-6 PCA WDT Functional Block Diagram

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

Input a 16 -bit compare value in the compare / capture register (CCAP4). In the PCA timer / counter (CNT), enter a 16 -bit initial value or use a reset value (0000h). these values multiplied by the PCA input pulse rate determines the WDT match run time. Setting the Timer / Counter Run Control bit (CCON.CR) starts the PCA WDT. The WDT of the PCA generates a reset signal every time there is a match. To prevent a PCA WDT from resetting, the user has three options.

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

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

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

PCA WDT configuration process

1) Configure WDT compare / capture register PCA_CCAP4
2) Configure the PCA count register PCA_CNT
3) Configure PCA_CCAMP4 to select the compare and match function
4) Configure PCA_CMOD to select the input clock and enable the WDT function
5) Start PCA
6) Select clear PCA WDT clear mode to clear PCA WDT before PCA WDT reset

16.2.3.4 PCA 8 -bit PWM function

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

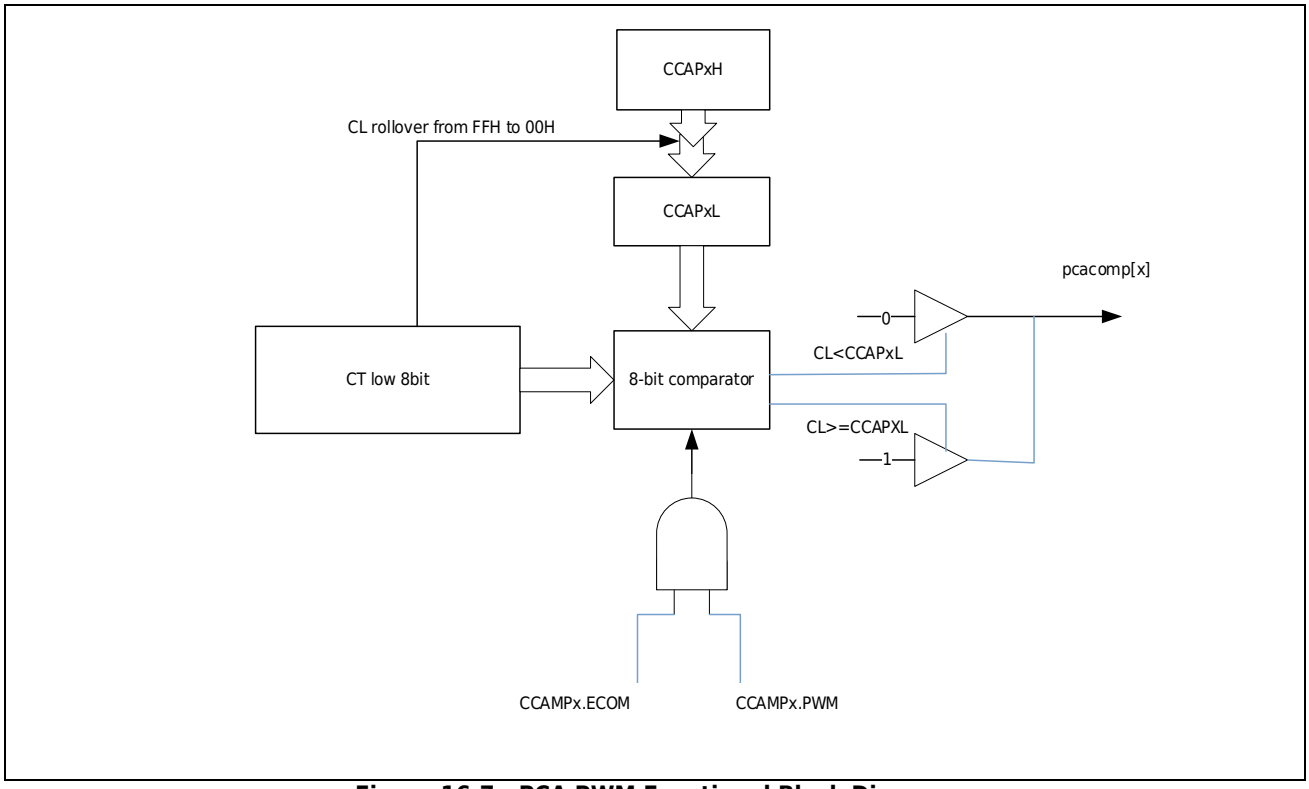


Figure 16-7 PCA PWM Functional Block Diagram

In this mode, the value in the low byte of the PCA timer / counter (CL) is constantly compared with the value in the low byte compare / capture register (CCAPxL). When CL < CCAPxL, the output waveform is low. When both match (CL=CCAPxL), the output waveform goes high until CL overflows from FFH to 00H, and remains high during the end period. On overflow, the value in CCAPxH is automatically loaded into CCAPxL, and a new cycle begins.

The value in CCAPxL determines the duty cycle of the current waveform. The value in CCAPxH determines the duty cycle of the next waveform Pulse Width Modulation that can be changed by changing the value in CCAPxL. As shown, the 8- bit value in CCAPxL can range from 0 (100 % duty cycle), to 255 (0.4 % duty cycle). To change the CCAPxL value without glitching, write a new value

in the high byte register (CCAPxH). When CL rolls over FFH to 00h, this value is automatically loaded into CCAPxL by hardware.

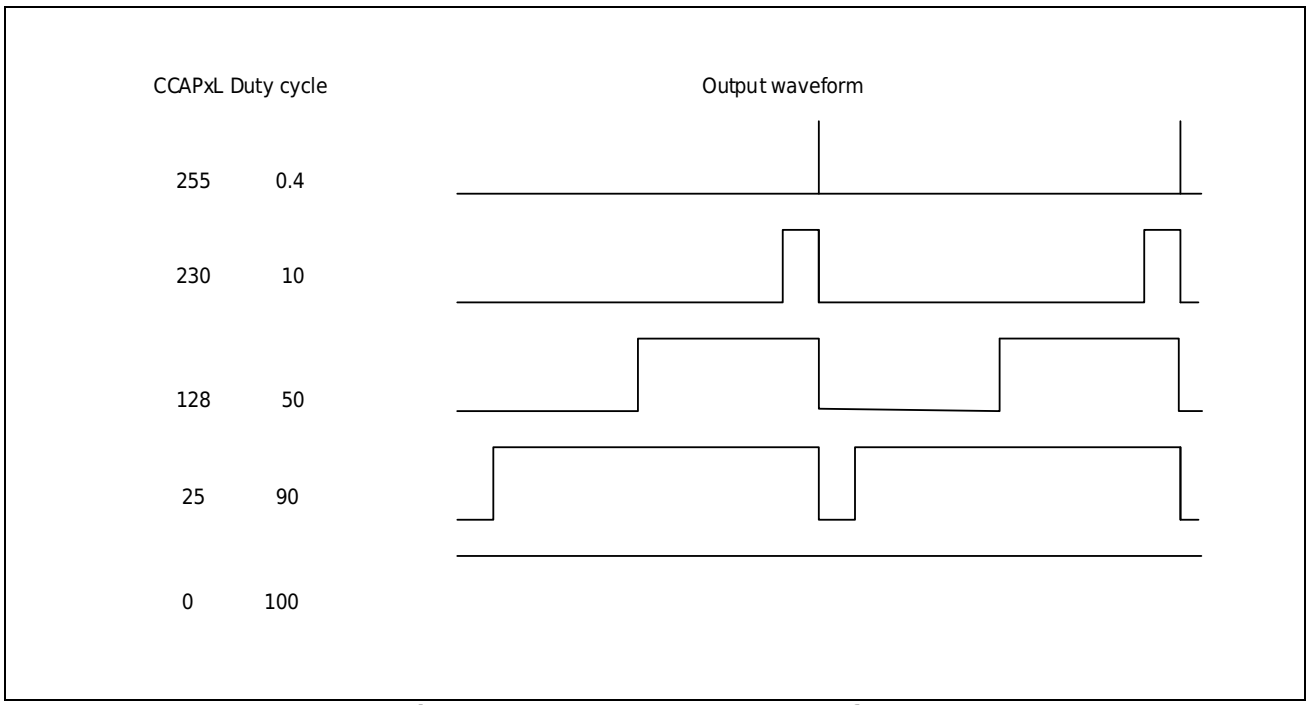


Figure 16-8 PCA PWM Output Waveform

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

Table 16-1 PCA comparison capture function module setup

ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	EPWM	Way of working
X	1	0	0	0	0	X	0	Capture with positive edge trigger
X	0	1	0	0	0	X	0	Trigger capture with negative edge
X	1	1	0	0	0	X	0	Capture with edge trigger
1	0	0	1	0	0	X	0	Software timer
1	0	0	1	1	0	X	0	High speed output
1	0	0	0	0	1	X	0	8 -bit pulse width modulator
1	0	0	1	1	0	X	1	16 -bit pulse width modulator

16.3Interconnection and control of PCA module and other modules

PCA can be connected to other modules or ports through port function selection.

		0	1	2	3	4	5	6	7
GPIO_PCAS[2:0]	PCA_ECI	PX_SEL	PCNT1	LVD	VC0	VC1	PA05	PB02	PD02
GPIO_PCAS[5:3]	PCA_CH0	PX_SEL	PCNT0	PCNT1	LVD	VC1	PA06	PB04	PC06

When GPIO_PCAS[2:0]=0x0, the PCA_ECI input is the port input selected by PX_SEL. When GPIO_PCAS[2:0]=0x1~0X7, it is connected to the input or output of other modules.

When GPIO_PCAS[5:3]=0x0, the PCA_CH0 capture input is the port input selected by PX_SEL. When GPIO_PCAS[5:3]=0x1~0X7, it is connected to the input or output of other modules.

16.4PCA register description

Base address 0X40001000

Table 16-2 PCA register list

Register	Offset address	Description
PCA_CCON	0X000	PCA Control Register
PCA_CMOD	0X004	PCA Mode Register
PCA_CNT	0X008	PCA count register
PCA_ICLR	0X00C	PCA Interrupt Clear Register
PCA_CCAPM0	0x010	PCA Compare/Capture Module 0 Mode Register
PCA_CCAPM1	0x014	PCA Compare/Capture Module 1 Mode Register
PCA_CCAPM2	0x018	PCA Compare/Capture Module 2 Mode Register
PCA_CCAPM3	0x01C	PCA compare / capture module 3 mode register
PCA_CCAPM4	0x020	PCA Compare/Capture Module 4 Mode Register
PCA_CCAP0H	0X024	PCA compare/capture module 0 high 8-bit register
PCA_CCAP0L	0X028	PCA compare/capture module 0 lower 8-bit register
PCA_CCAP1H	0X02C	PCA comparison/capture module 1 high 8-bit register
PCA_CCAP1L	0X030	PCA compare/capture module 1 lower 8-bit register
PCA_CCAP2H	0X034	PCA comparison/capture module 2 high 8-bit register
PCA_CCAP2L	0X038	PCA compare/capture module 2 lower 8-bit register
PCA_CCAP3H	0X03C	PCA comparison/capture module 3 high 8-bit register
PCA_CCAP3L	0X040	PCA comparison / capture module 3 lower 8 -bit registers
PCA_CCAP4H	0X044	PCA comparison/capture module 4 high 8-bit register
PCA_CCAP4L	0X048	PCA compare/capture module 4 lower 8-bit register
PCA_CCAPO	0X04C	PCA PWM and high-speed output flag register
PCA_CCAP0	0X050	16 -bit register for PCA compare / capture module 0
PCA_CCAP1	0X054	16 -bit register for PCA compare / capture module 1
PCA_CCAP2	0X058	16 -bit register for PCA compare / capture module 2
PCA_CCAP3	0X05C	16 -bit register for PCA compare / capture module 3
PCA_CCAP4	0X060	16 -bit register for PCA compare / capture module 4
PCA_CARR	0X064	PCA cycle load register
PCA_EPWM	0X068	PCA PWM Enhancement Register

16.4.1 Control Register (PCA_CCON)

Offset address: 0x000

Reset value: 0x0000/ 0000h

31-8	7	6	5	4	3	2	1	0
Reserved	CF	CR	Reserved	CCF4	CCF3	CCF2	CCF1	CCF0
	RO	RW		RO	RO	RO	RO	RO

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

16.4.2 Mode Register (PCA_CMOD)

Offset address: 0x004

Reset value: 0x0000/ 0000h

31-8	7	6	5	4	3	2	1	0
Reserved	CIDL	WDTE	Reserved		CPS			CFIE
	RW	RW			RW			RW

Bit	Symbol	Description
31:8	Reserved	Reserved bit
7	CIDL	PCA stop working in idle mode IDLE ? 1: In sleep mode (sleep), PCA stops working 0: In sleep mode (sleep), PCA continues to work
6	WDTE	PCA WDT function enable control bit 1: Start PCA module 4 WDT function 0: Disable PCA module 4 WDT function
5:4	Reserved	Reserved bit
3:1	CPS[2:0]	Clock frequency division selection and clock source selection 000: PCLK/32 001: PCLK/16 010: PCLK/8 011: PCLK/4 100: PCLK/2 101: timer0 overflow 110: timer1 overflow 111: ECI external clock, clock PCLK four-frequency sampling
0	CFIE	PCA counter interrupt enable control signal 1: enable interrupt 0: disable interrupt

16.4.3 Count register (PCA_CNT)

Offset address: 0x008

Reset value: 0x0000/ 0000h

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

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

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

16.4.4 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
	R1W0			R1W0	R1W0	R1W0	R1W0	R1W0

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

16.4.5 Compare Capture Mode Register (PCA_CCAPM0~4)

Offset address:

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

CCAPM3: 0x01C; CCAPM4: 0x020;

Reset value: 0x0000/ 0000h

31-8	7	6	5	4	3	2	1	0
Reserved		ECOM	CAPP	CAPN	MAT	TOG	PWM	CCIE
		RW	RW	RW	RW	RW	RW	RW

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

16.4.6 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/ 0000h

31:8	7	6	5	4	3	2	1	0
Reserved	CCAPx[15: 8]							
	RW							

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

16.4.7 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/ 0000h

31:8	7	6	5	4	3	2	1	0
Reserved	CCAPx[7: 0]							
	RW							

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

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

Offset address

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

CCAP3: 0x05C; CCAP4: 0x060;

Reset value: 0x0000/ 0000h

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCAPx[15: 0]															
RW															

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

16.4.9 Compare High Speed Output Flag Register (PCA_CCAPO)

Offset address: 0x04C

Reset value: 0x0000/ 0000h

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

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

16.4.10 Period Register (PCA_CARR)

Offset address: 0x064

Reset value: 0x0000/ 0000h

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
CARR[15: 0]															
RW															

Bit	Symbol	Description
31:16	Reserved	Reserved bit
15:0	CARR	Count cycle reload register

16.4.11 Enhanced PWM Control (PCA_EPWM)

Offset address: 0x068

Reset value: 0x0000/ 0000h

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															EPWM
															RW

Bit	Symbol	Description
31:1	Reserved	Reserved bit
0	EPWM	16 bit PWM enable

17 Advanced Timer (TIM4/5/6)

17.1 Introduction to Advanced Timer

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

The basic functions and features of Advanced Timer are shown in the table.

Table 17-1 Basic Features of Advanced Timer

Waveform mode	Sawtooth wave, triangular wave
Basic functions	• Direction of increments and decrements
	• Software synchronization
	• Hardware synchronization
	• Cache function
	• Orthogonal coding count
	• General purpose PWM output
	• Protection mechanism
	• AOS associated action
Interrupt type	Count comparison match interrupt
	Count cycle match interrupt
	Dead time error interrupt

Table 17-2 Advanced Timer port list

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

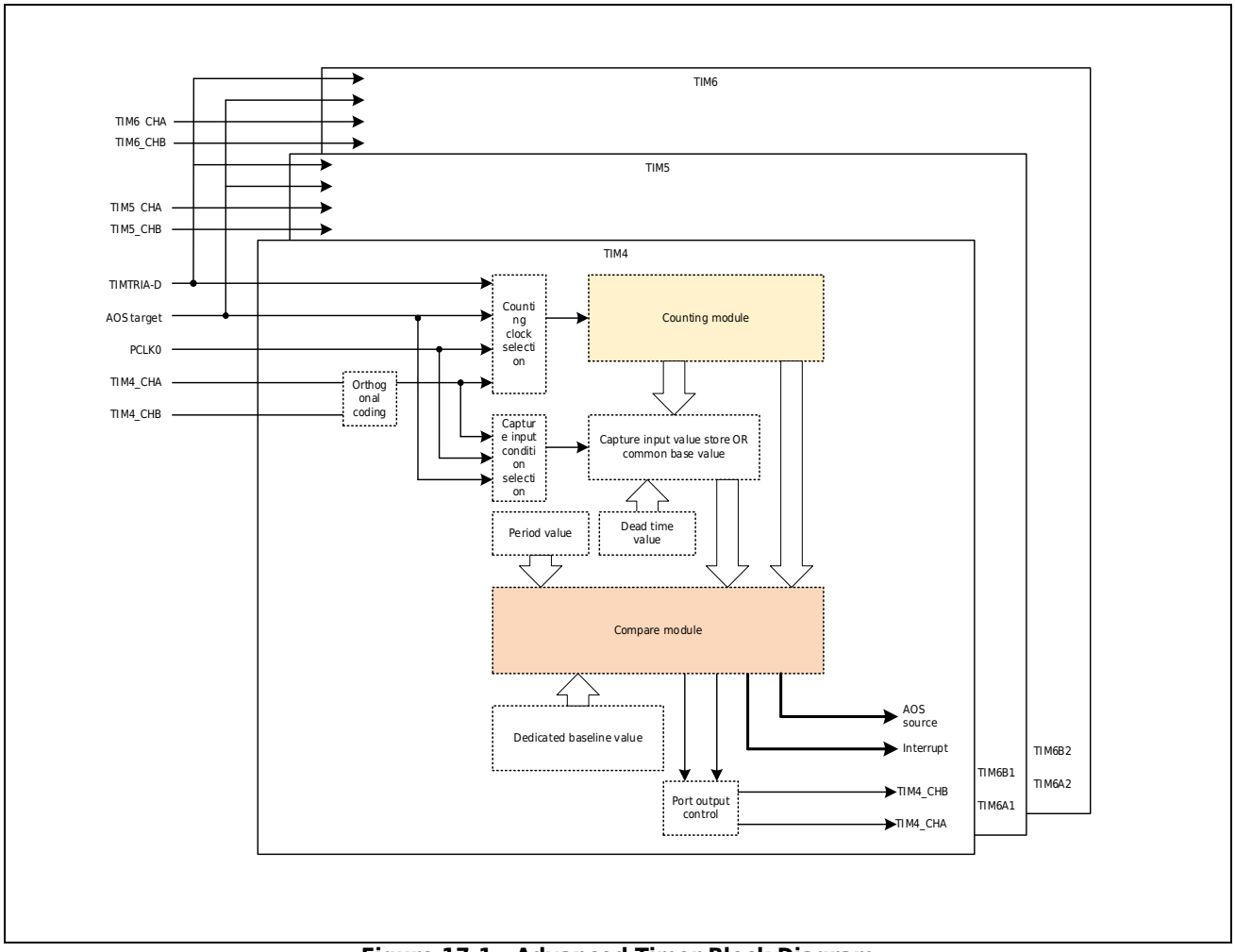


Figure 17-1 Advanced Timer Block Diagram

17.2Advanced Timer Function Description

17.2.1 Basic action

17.2.1.1 Basic Waveform Mode

Timer4/5/6 has 2 basic counting waveform modes, sawtooth wave mode and triangle wave mode. The waveform mode is subdivided due to different internal counting actions. The triangular wave mode is divided into triangular wave A mode and triangular wave B mode. The basic waveforms of sawtooth and triangle waves are shown in Figure 17-2 and Figure 17-3. The difference between the triangular wave A mode and the triangular wave B mode is that there is a difference in the buffer transfer. The triangular wave A mode only has one buffer transfer (valley point) in one cycle, while the triangular wave B mode has two buffer transfers (peak point and valley point) in one cycle.

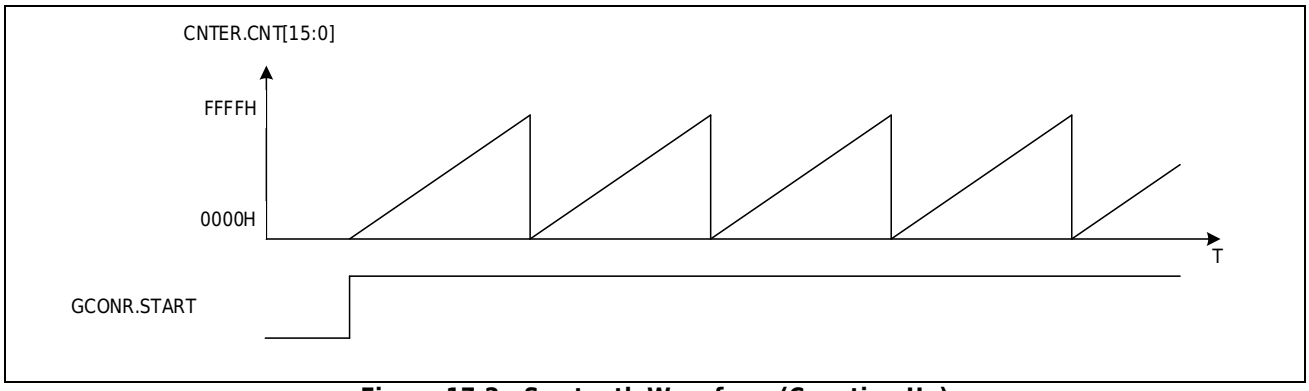


Figure 17-2 Sawtooth Waveform (Counting Up)

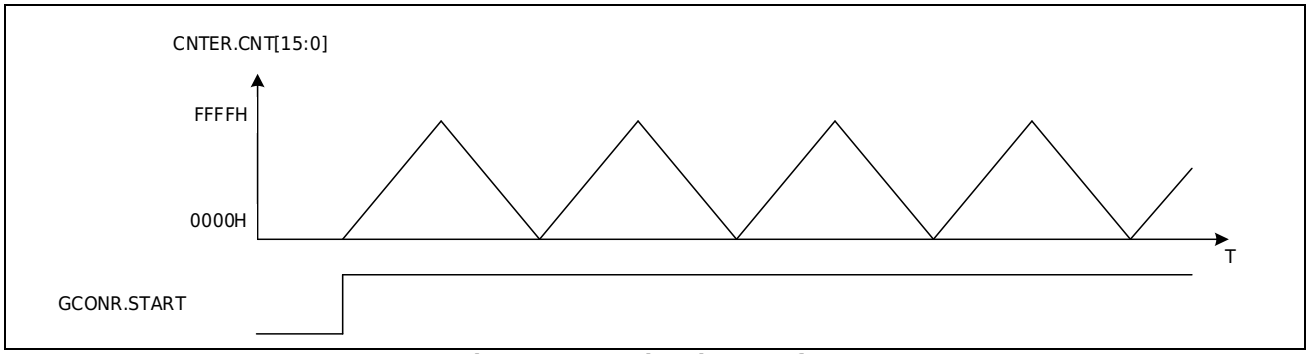


Figure 17-3 Triangle Waveform

17.2.1.2 Comparison output

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

The counting start level, stop level, and counting comparison match level of CHxA and CHxB ports can be set by PCONR.STACA, PCONR.STPCA, PCONR.STASTPSA, PCONR.CMPCA [1:0 of the port control register (PCONR)], PCONR.PERCA[1:0] and PCONR.STACB, PCONR.STPCB, PCONR.STASTPSB, PCONR.CMPCB[1:0], PCONR.PERCB[1:0] bit settings. Figure 17-4 is an example of the comparison output operation.

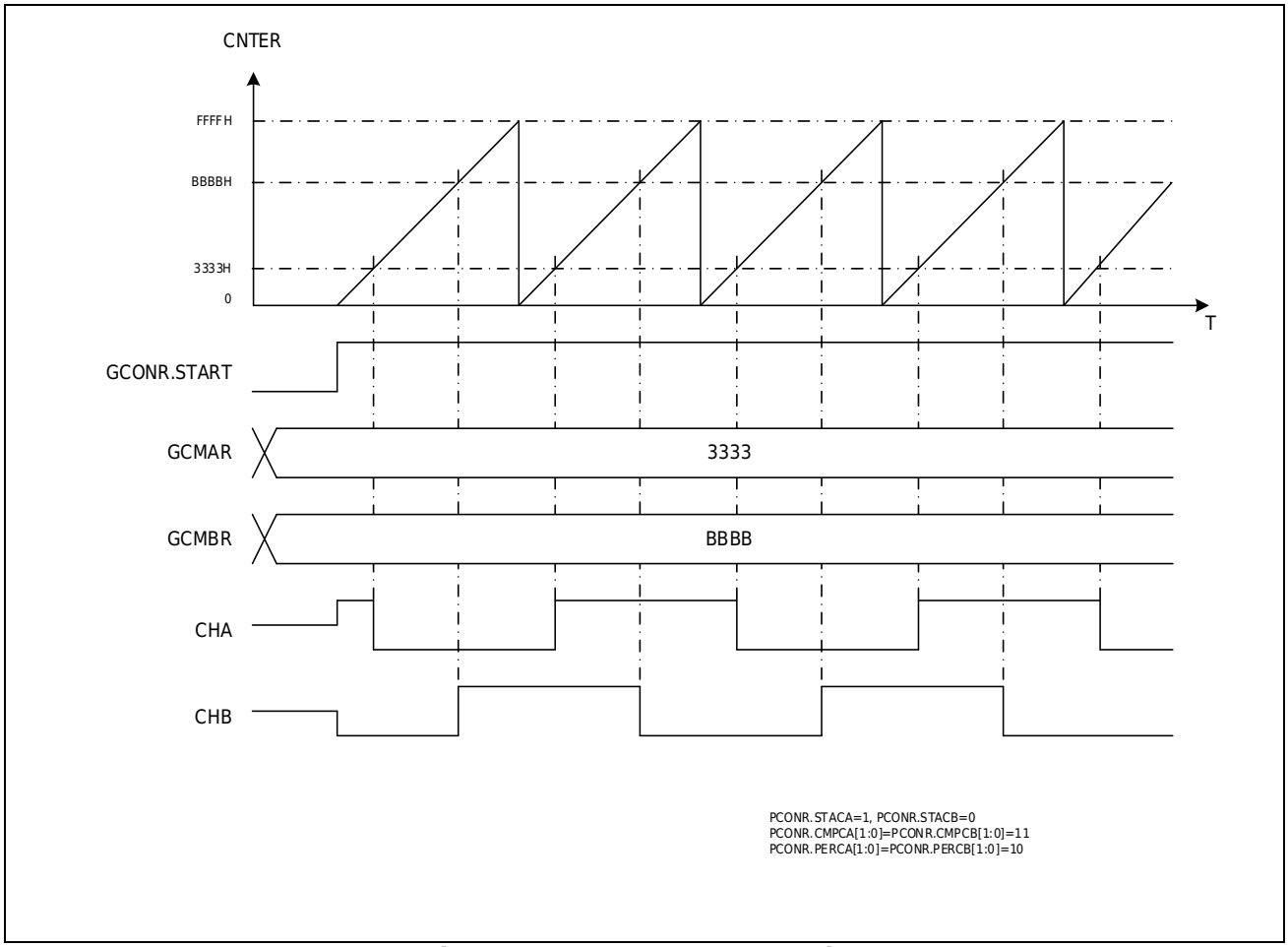


Figure 17-4 Compare output action

17.2.1.3 Capture input

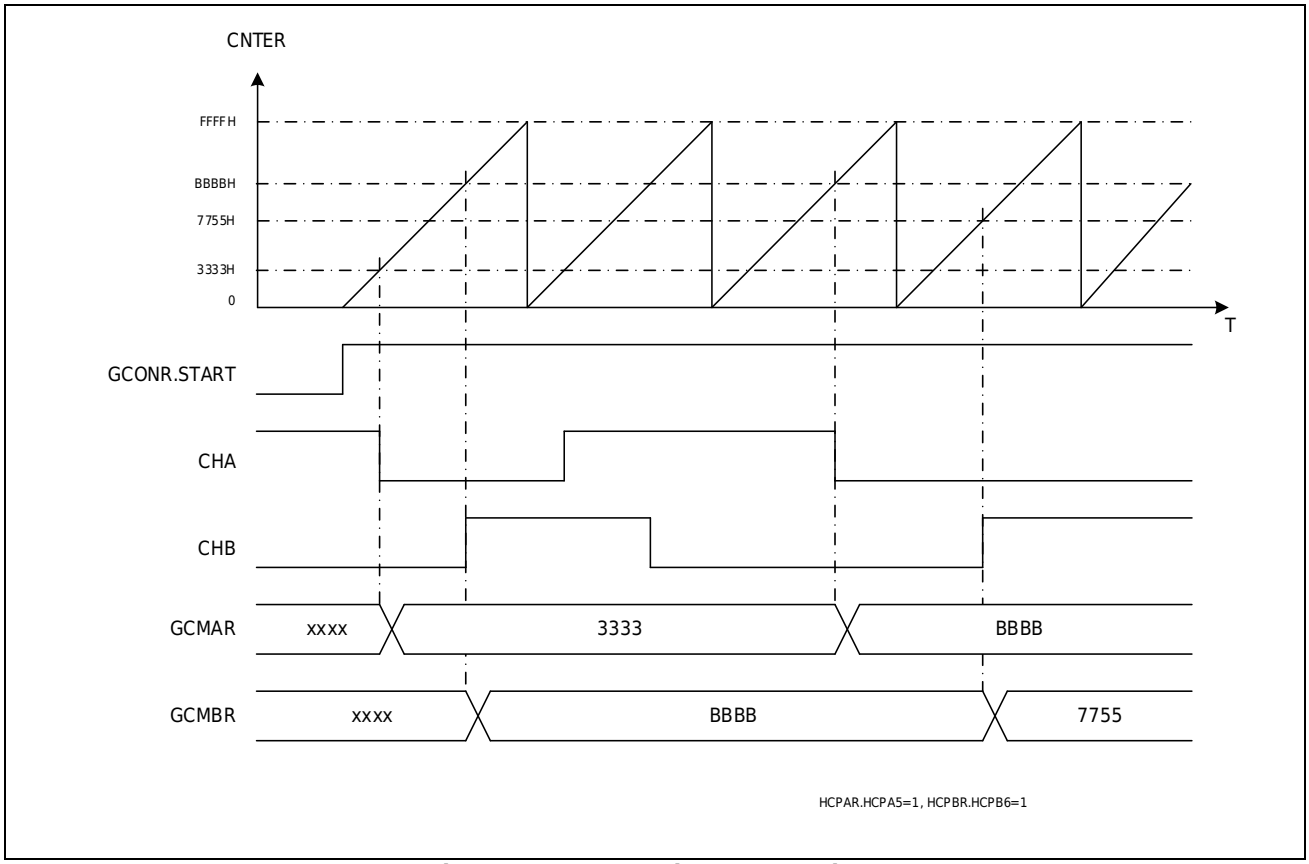


Figure 17-5 Capturing Input Actions

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

The condition of each capture input can be AOS event trigger, TIMTRIA-TIMTRID input, CHxA or CHxB input, etc. The specific condition selection can be set through the hardware capture event selection register (HCPAR, HCPBR). Figure 17-5 is an example of an action to capture input.

17.2.2 Timer selection

Timer4/5/6 can have the following options:

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

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

17.2.3 Counting direction

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

17.2.3.1 Sawtooth counting direction

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

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

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

17.2.3.2 Triangular wave counting direction

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

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

17.2.4 Digital filtering

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

When the filtered sampling reference clock is sampled to the same level three times on the port, the level is transferred to the module as an effective level. A level less than three times consistent will be filtered out as external interference and not transmitted to the module. Its operation is shown in Figure 17-6, for example.

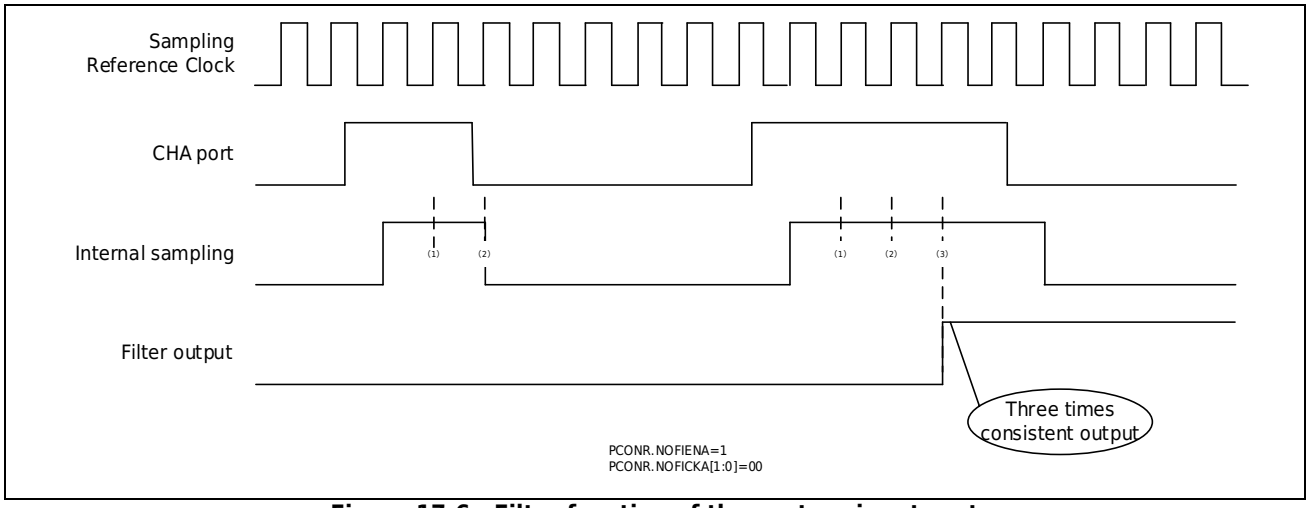


Figure 17-6 Filter function of the capture input port

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

17.2.5 Software synchronization

17.2.5.1 Software synchronization start

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

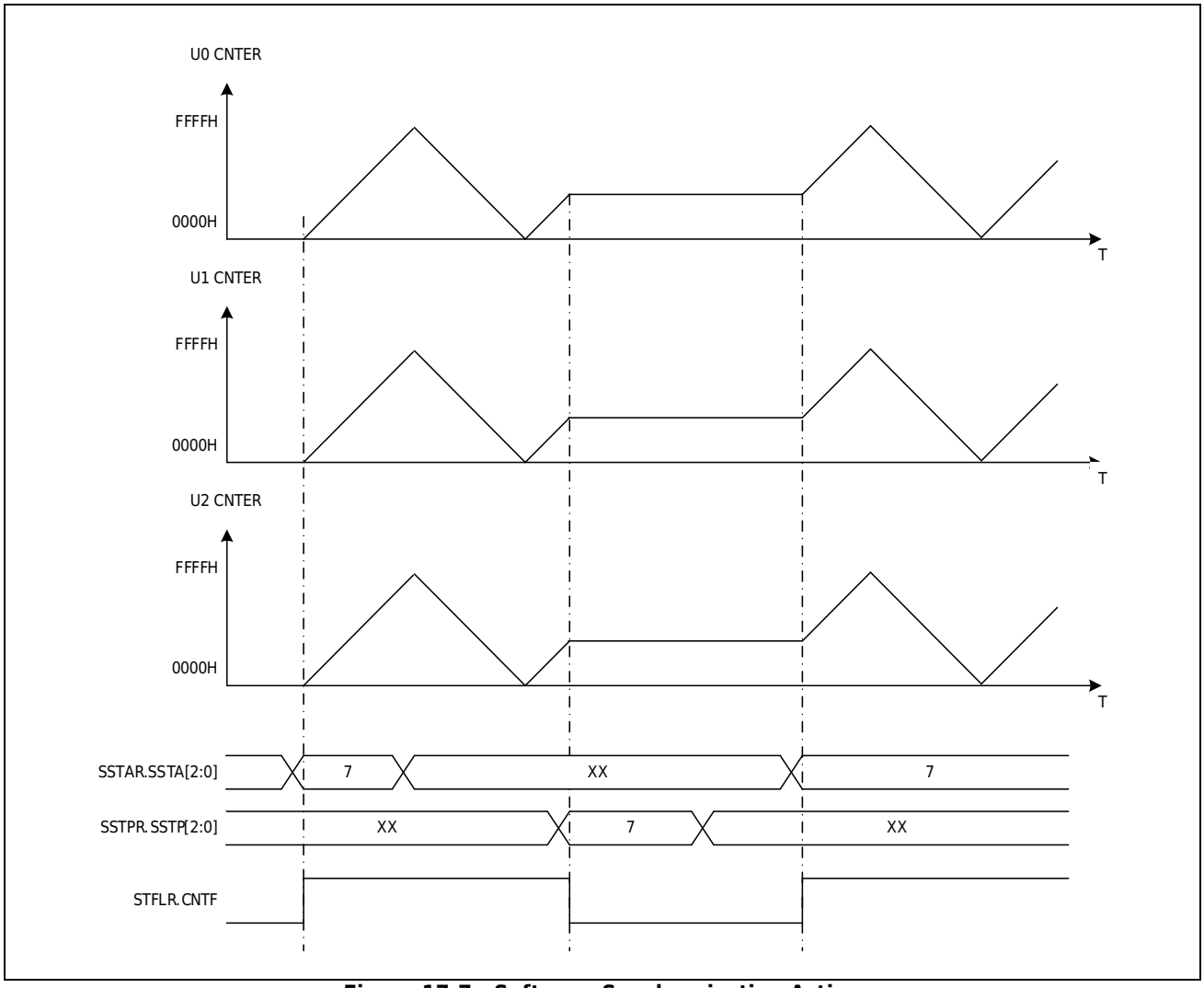


Figure 17-7 Software Synchronization Action

17.2.5.2 Software co-stop

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

17.2.5.3 Software synchronous clear

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

As shown in Figure 17-7, if SSTAR. SSTA0=SSTAR. SSTA1=SSTAR. SSTA2 is set for Timer4, software synchronization startup for Timer4/5/6 can be achieved.

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

17.2.6 Hardware synchronization

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

17.2.6.1 Hardware sync start

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

17.2.6.2 Hardware co-stop

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

17.2.6.3 Hardware synchronous clear

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

17.2.6.4 Hardware Synchronous Capture Input

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

17.2.6.5 Hardware sync count

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

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

Figure 17-8 shows an example of the hardware synchronous operation of Timer4/5/6.

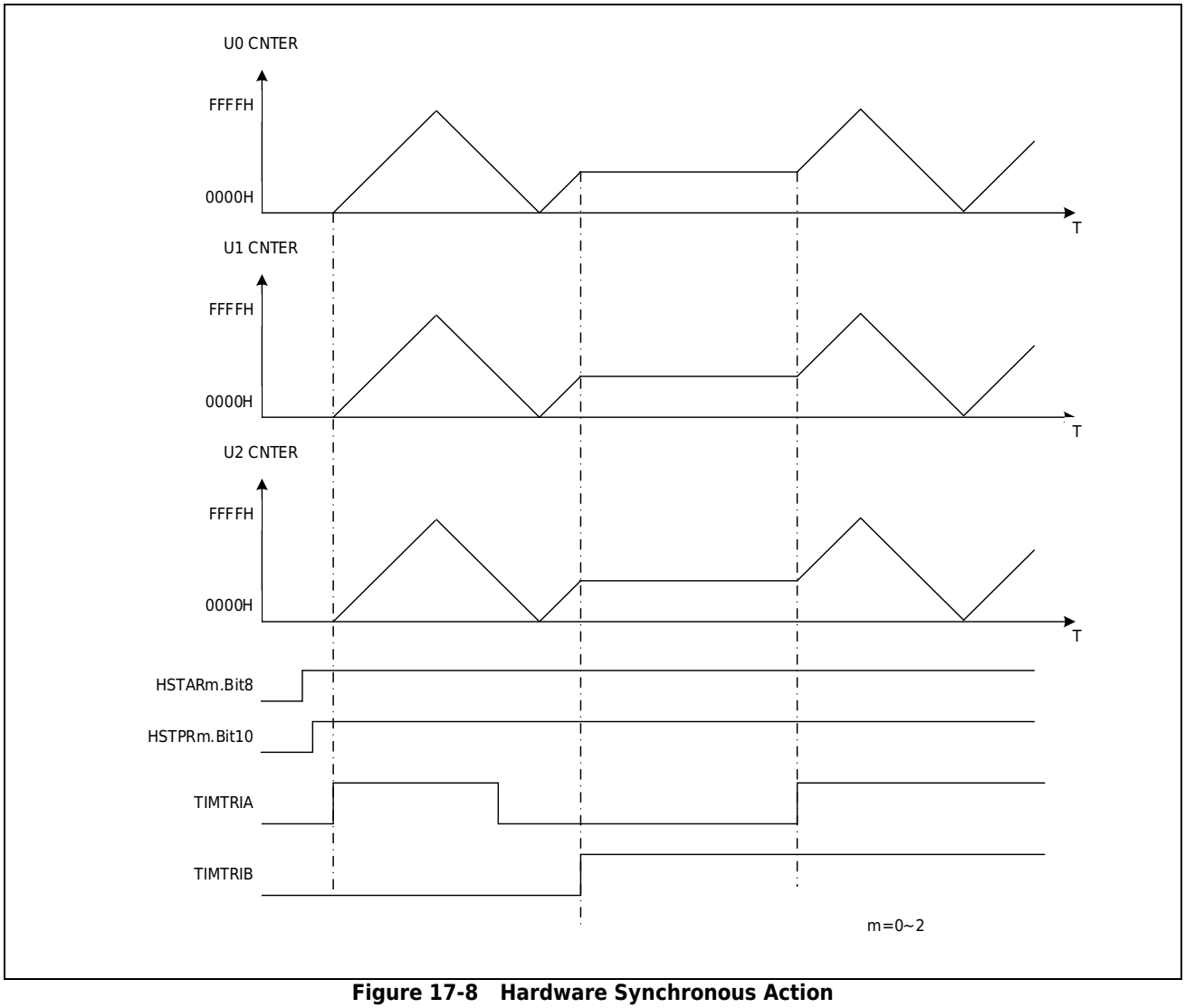


Figure 17-8 Hardware Synchronous Action

17.2.7 Cache function

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

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

Figure 17-9 shows the timing chart of the single cache method of the universal comparison reference value register when comparing output actions. It can be seen from the figure that changing the value of the general comparison reference value register (GCMAR) during counting can adjust the output duty cycle, and changing the value of the general period reference value register (PERAR) can adjust the output cycle.

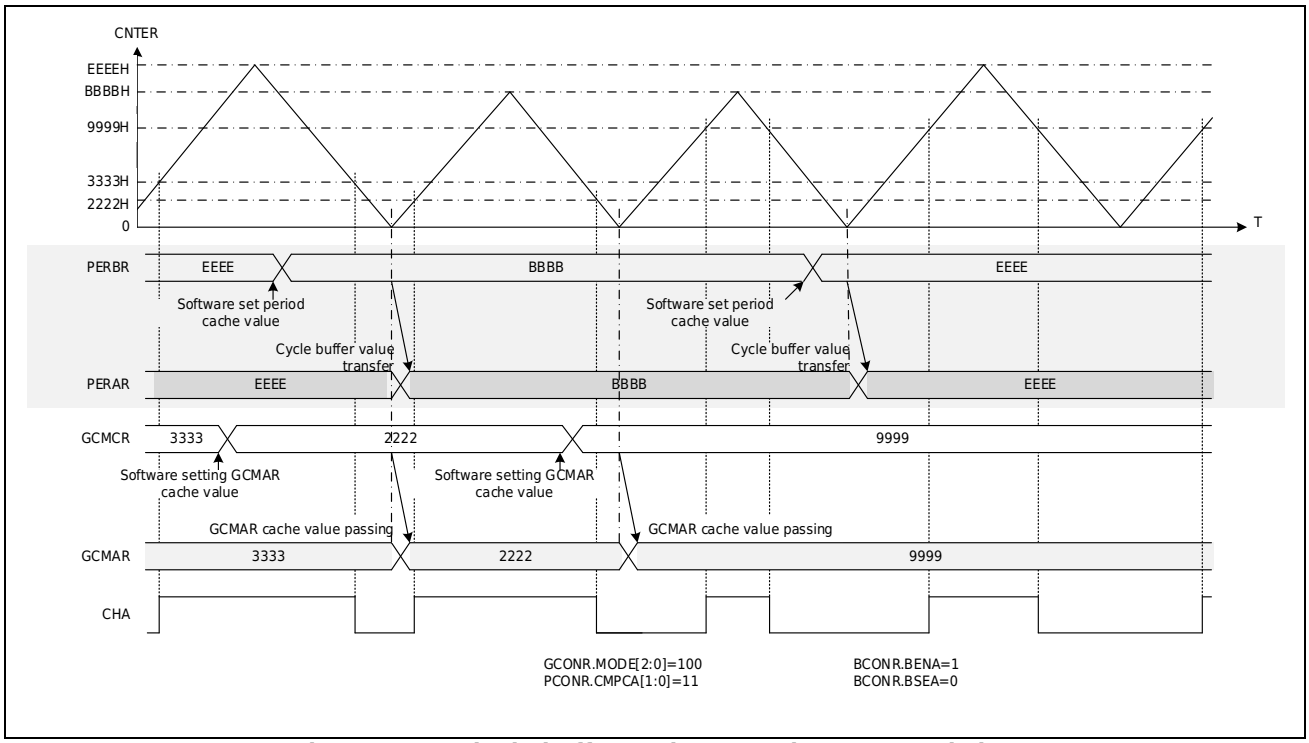


Figure 17-9 Single-buffer mode comparison output timing

17.2.7.1 Cache delivery time point

General cycle reference value cache transmission time point

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

Universal baseline value cache delivery time point

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

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

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

Capture input value buffer transfer time point

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

Buffer transfer during clear action

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

17.2.8 General PWM output

17.2.8.1 PWM spread spectrum output

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

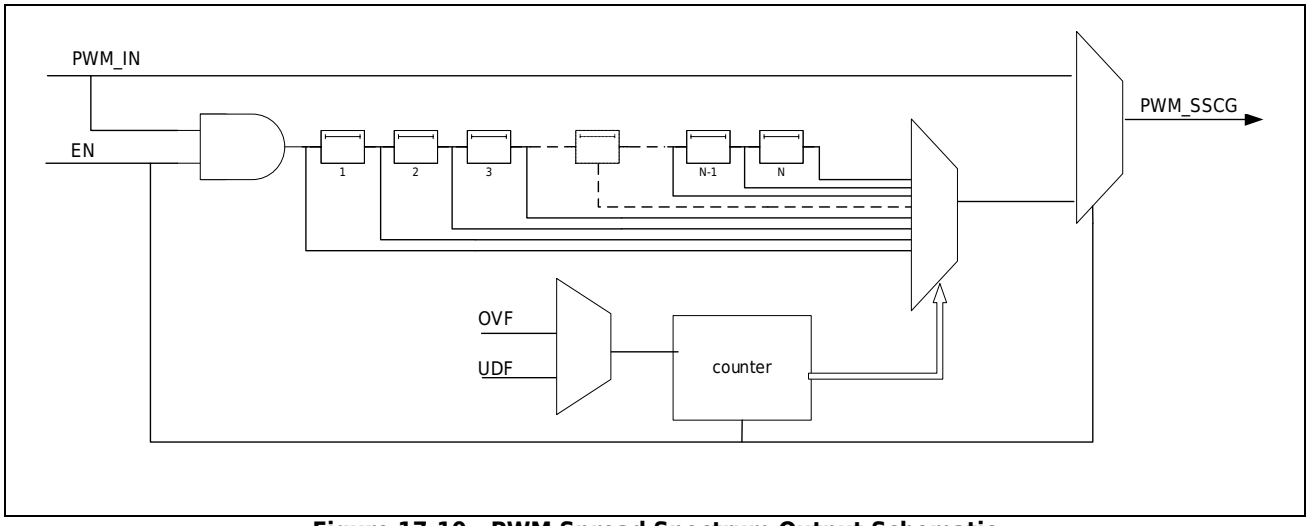


Figure 17-10 PWM Spread Spectrum Output Schematic

17.2.8.2 Independent PWM output

The 2 ports CHxA and CHxB of each timer can independently output PWM waves. As shown in Figure 17-11, the CHA port of Timer6 outputs PWM wave.

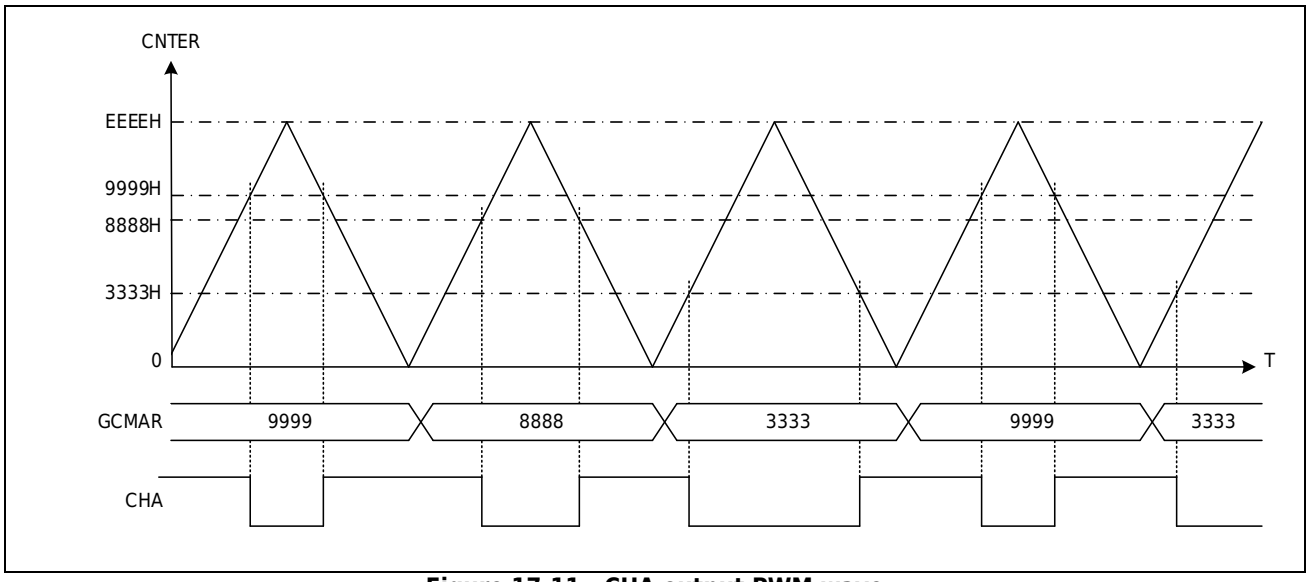


Figure 17-11 CHA output PWM wave

17.2.8.3 Complementary PWM output

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

Software setting GCMBR complementary PWM output

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

Figure 17-12 shows an example of software setting GCMBR complementary PWM wave output.

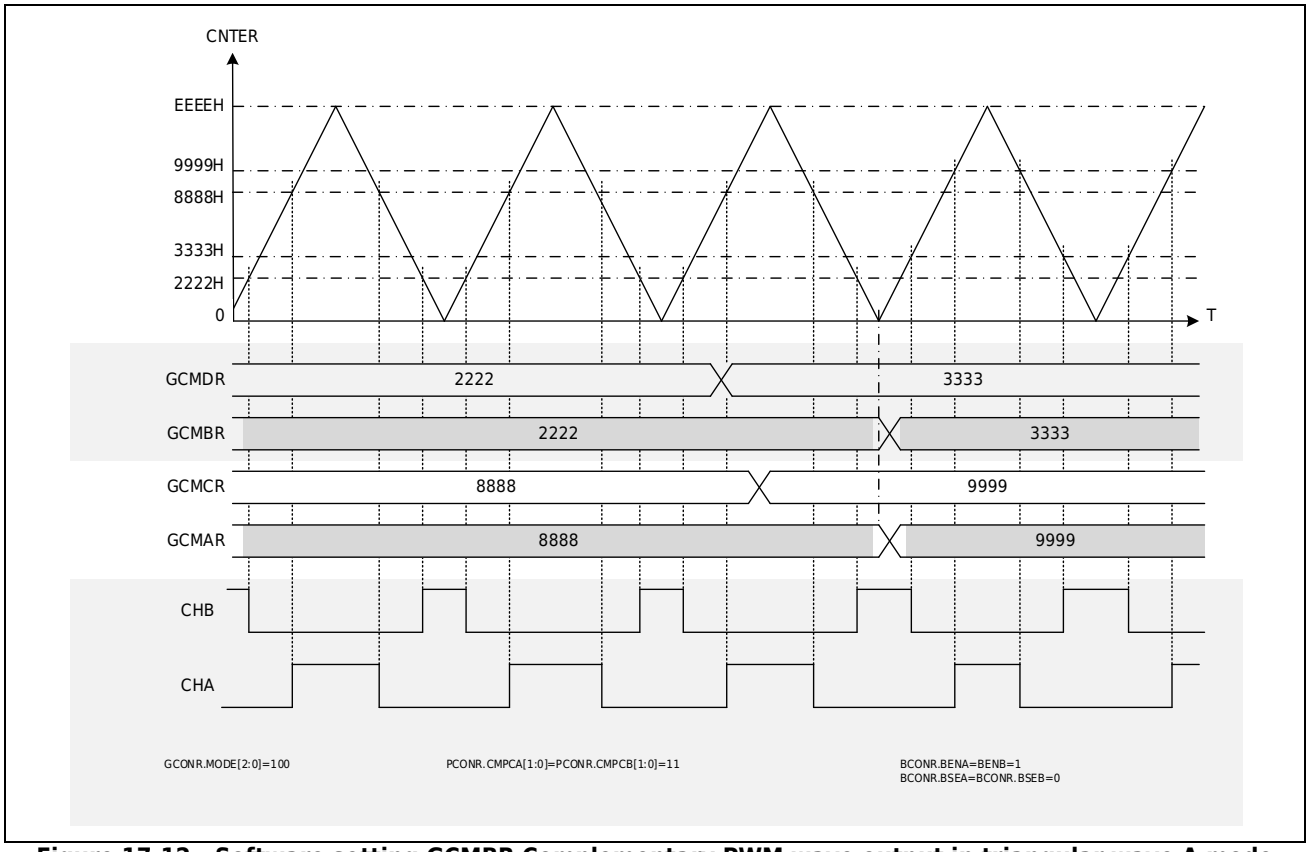


Figure 17-12 Software setting GCMBR Complementary PWM wave output in triangular wave A mode

Hardware setting GCMBR complementary PWM output

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

Figure 17-13 is an example of hardware setting GCMBR complementary PWM wave output.

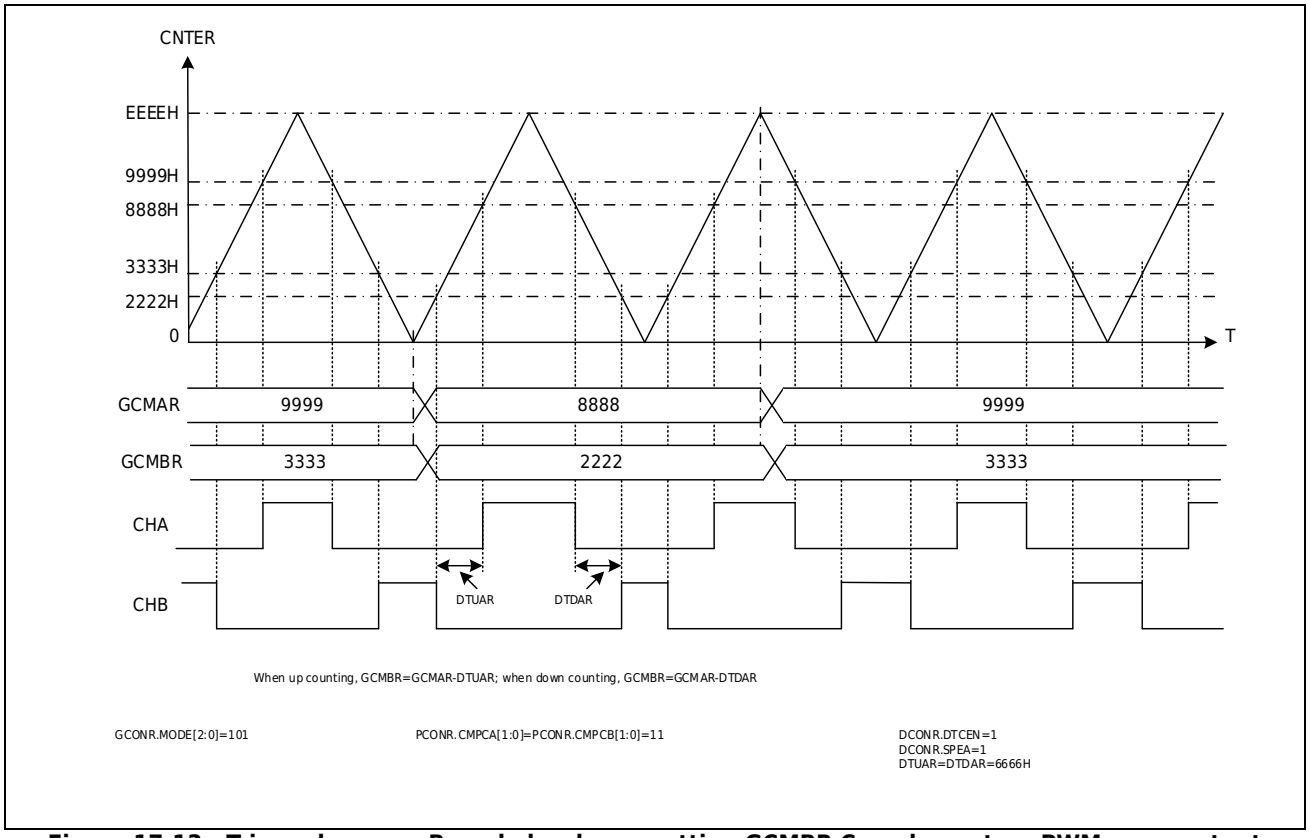


Figure 17-13 Triangular wave B mode hardware setting GCMBR Complementary PWM wave output (symmetrical dead zone)

17.2.8.4 Multi-phase PWM output

The CHxA and CHxB ports of each timer can output 2 -phase independent PWM waves or a group of complementary PWM waves. Multiple timers can be combined and combined with software and hardware synchronous actions to realize multi-phase PWM wave output. As shown in Figure 17-14, the combination of Timer4, Timer5, and Timer6 outputs 6-phase PWM waves; as shown in Figure 17-15, the combination of Timer4, Timer5, and Timer6 outputs 3 complementary PWM waves.

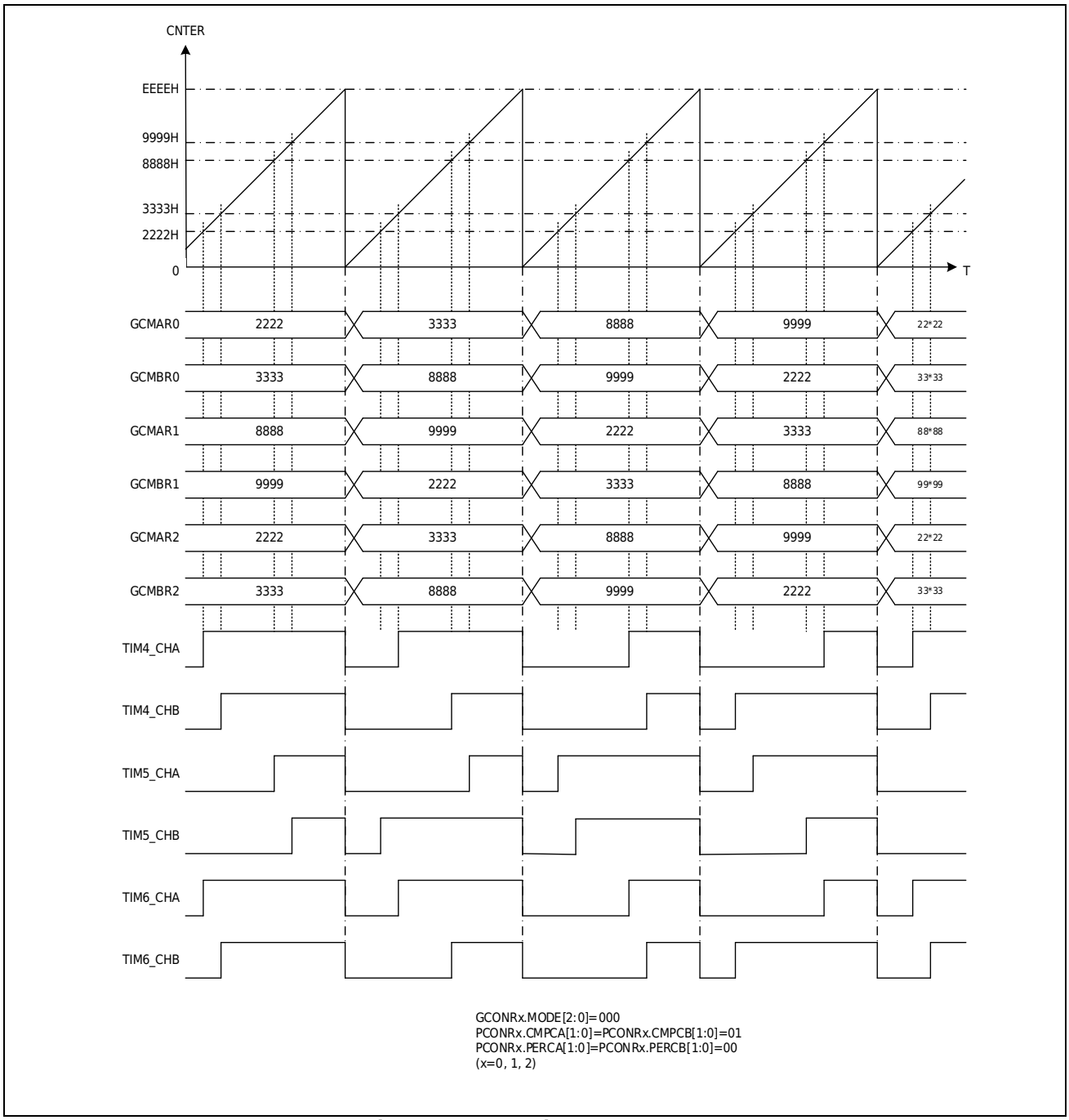


Figure 17-14 6-phase PWM wave

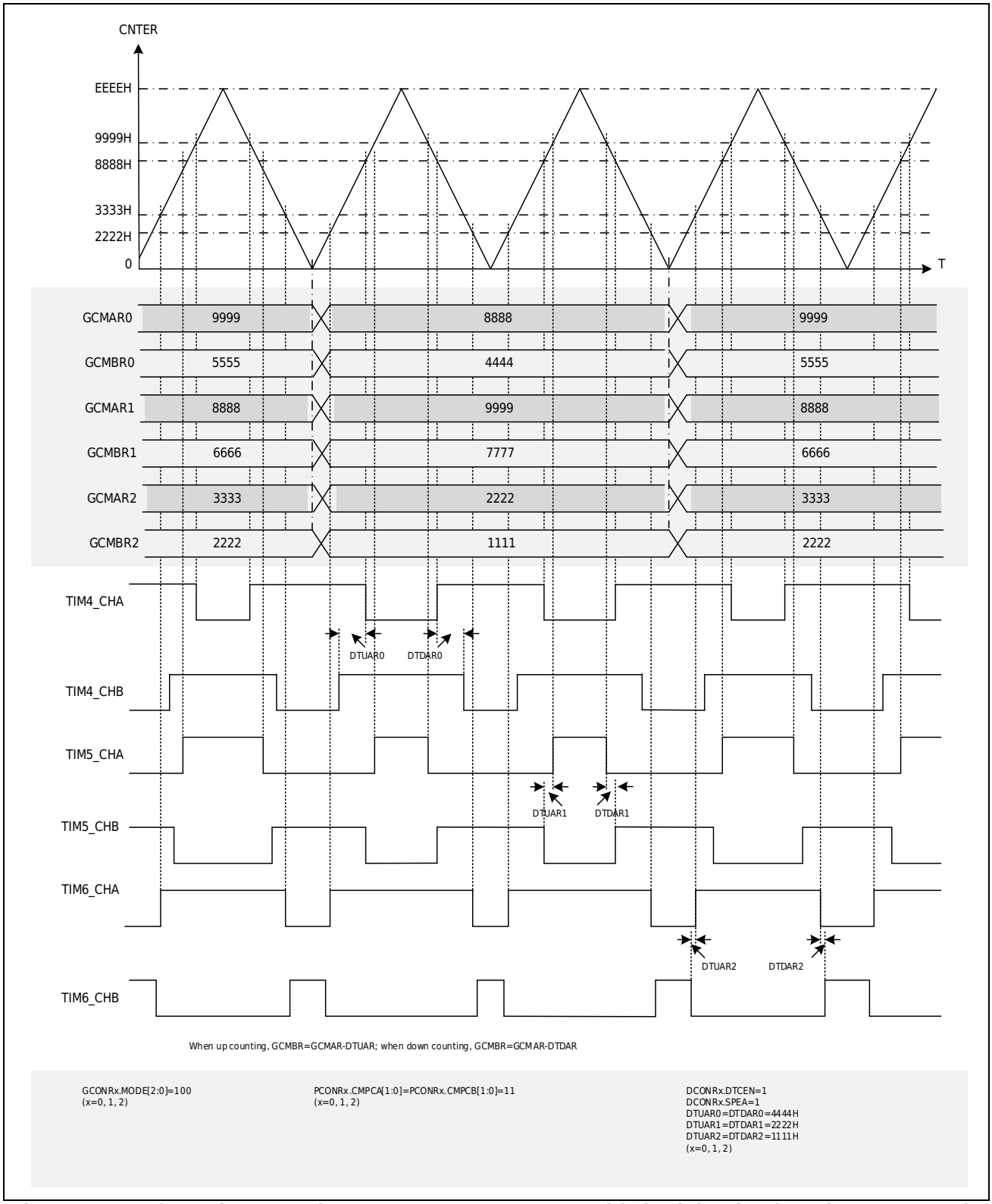


Figure 17-15 Three-phase complementary PWM wave output with dead time in triangular wave A mode

17.2.9 Orthogonal coding count

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

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

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

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

17.2.9.1 Position counting mode

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

Basic count

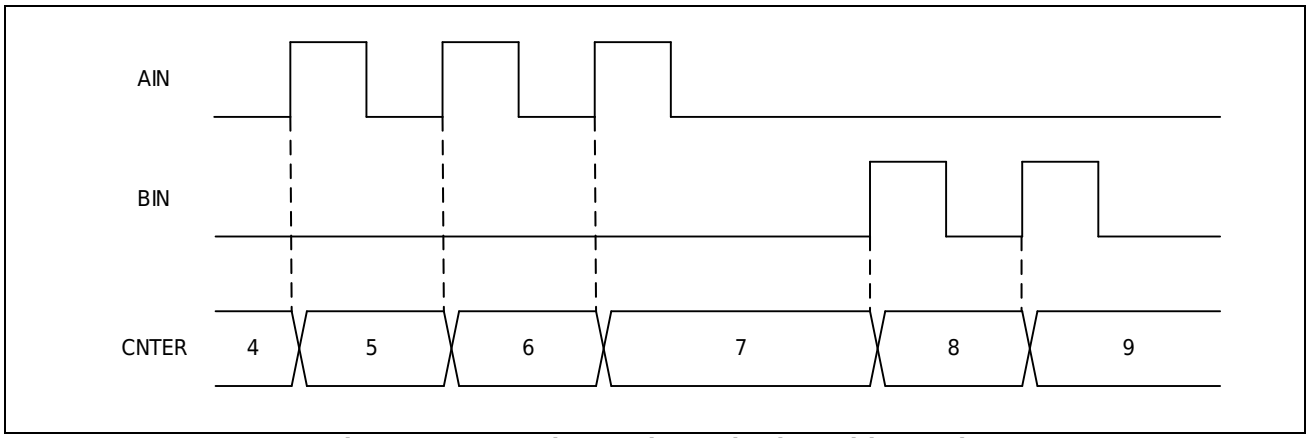


Figure 17-16 Basic counting action in position mode

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

Phase difference count

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

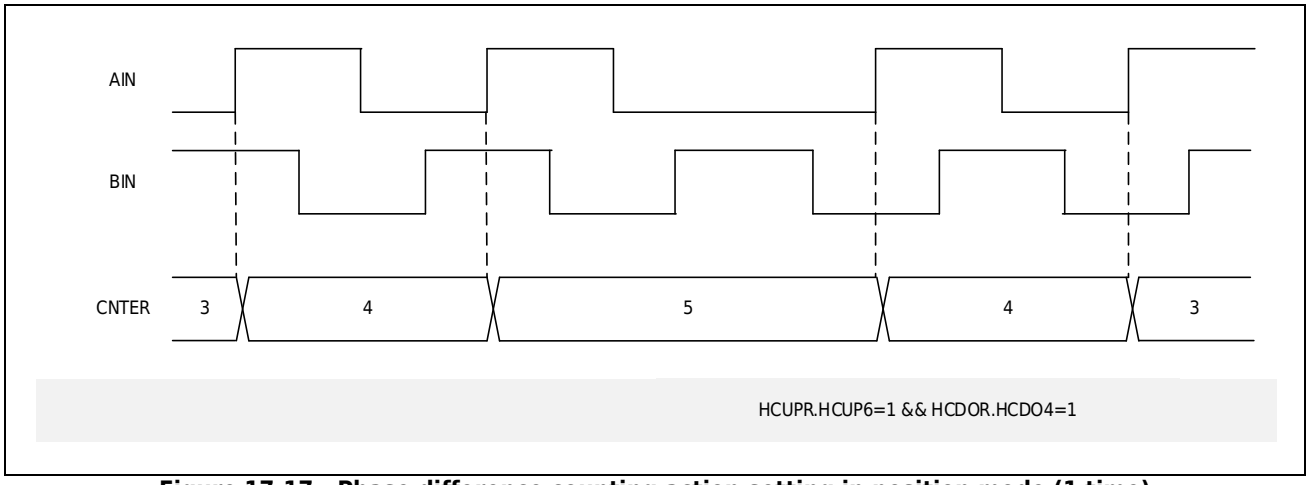


Figure 17-17 Phase difference counting action setting in position mode (1 time)

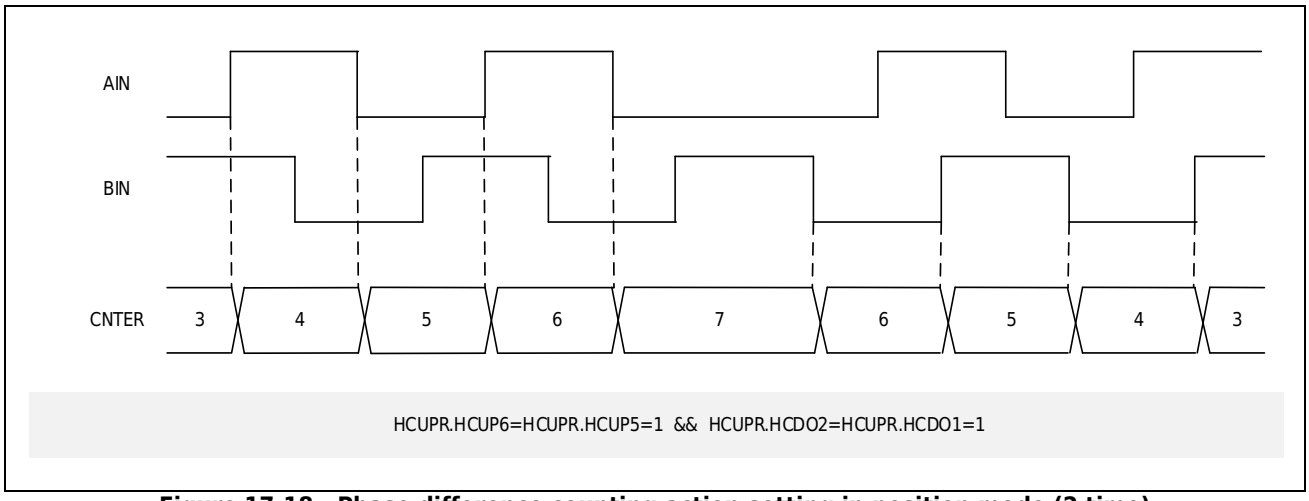


Figure 17-18 Phase difference counting action setting in position mode (2 time)

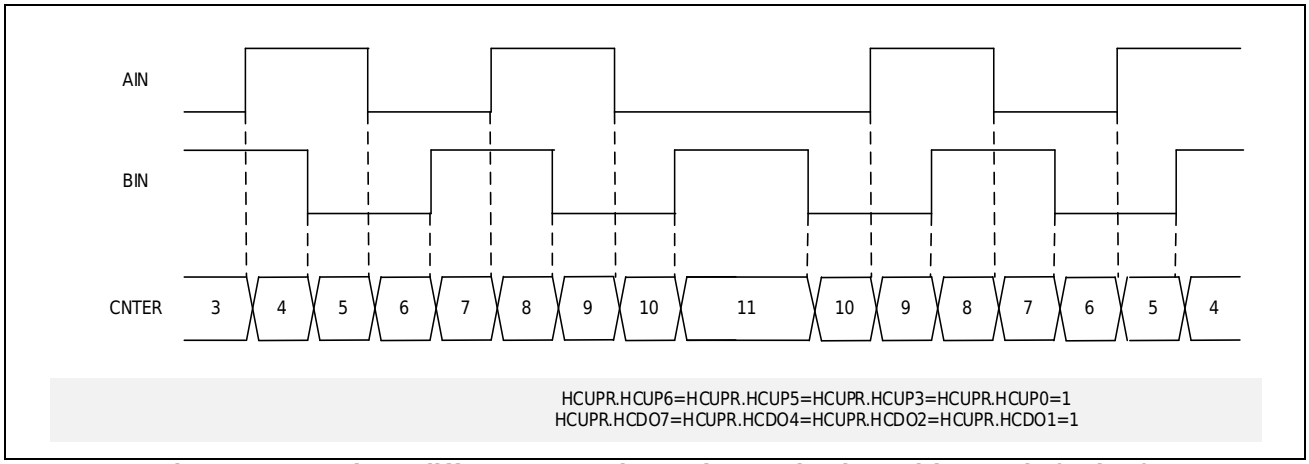


Figure 17-19 Phase difference counting action setting in position mode (4 time)

Direction count

Direction counting refers to setting the input state of AIN as direction control, and using the input of BIN as clock counting, as shown in Figure 17-20.

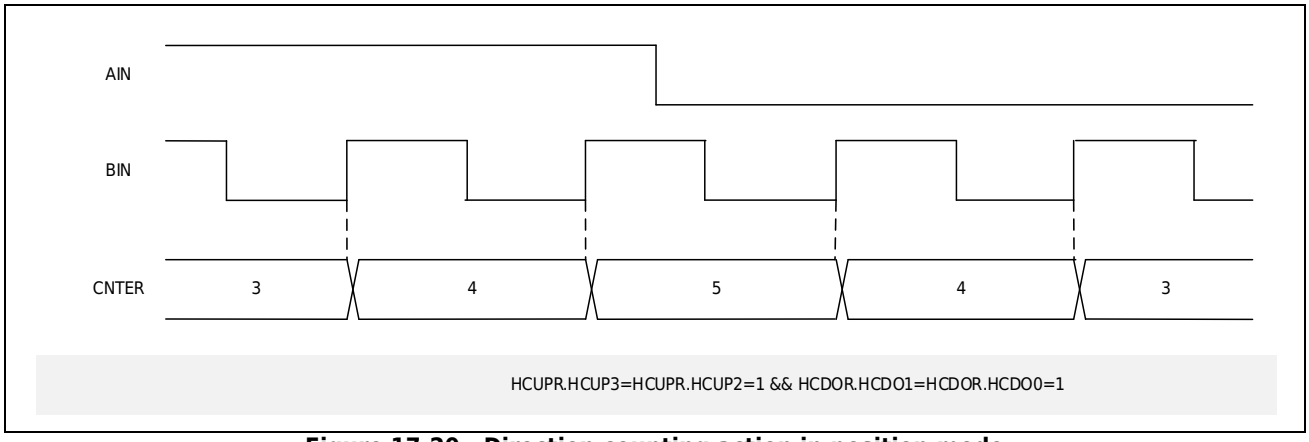


Figure 17-20 Direction counting action in position mode

17.2.9.2 Revolution mode

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

Z phase count

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

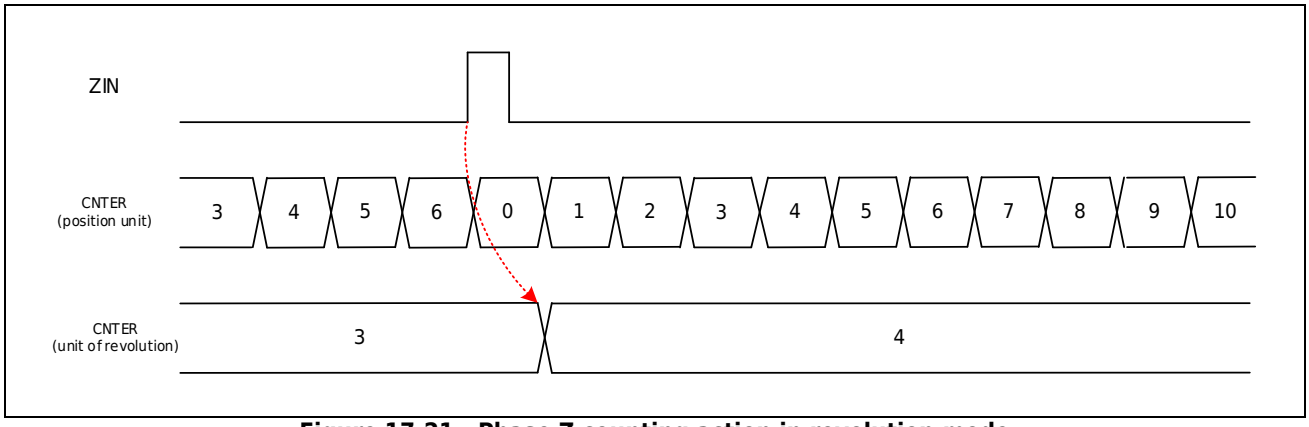


Figure 17-21 Phase Z counting action in revolution mode

Position overflow count

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

The overflow event of the position counting unit realizes the counting of the revolution counting unit through the linkage gating of the AOS module, and the position overflow counting can be realized. The increment (decrement) event selection register (HCUPR or HCDOR) of the hardware increment (decrement) event selection register (HCUPR or HCDOR) of the revolution counting unit selects 1 bit in Bit16:Bit19, and the AOS module sets the event source of the corresponding increment (decrement) event It is the count overflow event of the position counting unit, please refer to the AOS chapter for details. As shown in Figure 17-22.

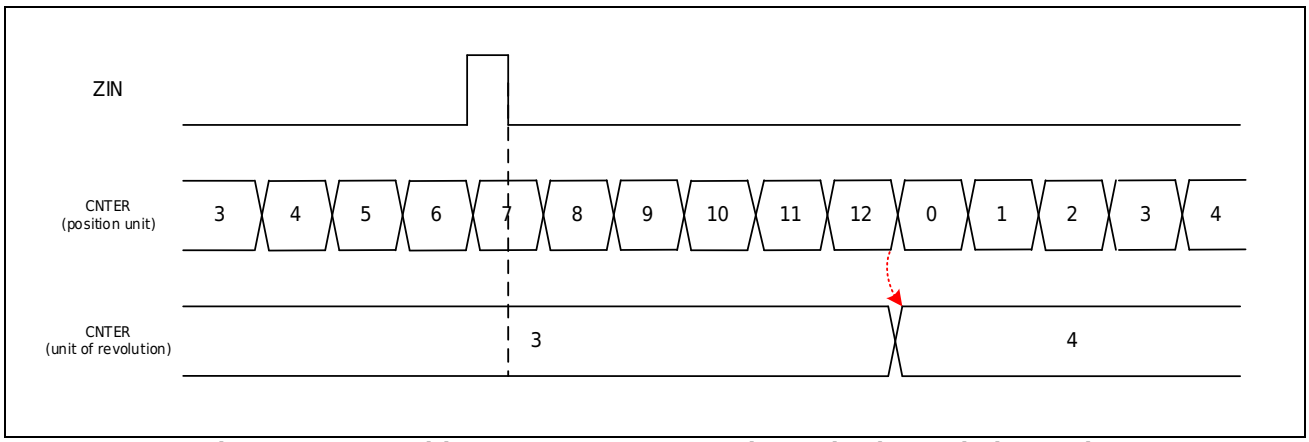


Figure 17-22 Position counter output counting action in revolution mode

mixed count

Mixed counting refers to the counting action that combines the above two counting methods of Z-phase counting and position overflow counting, and its realization method is also a combination of the above two counting methods. As shown in Figure 17-23.

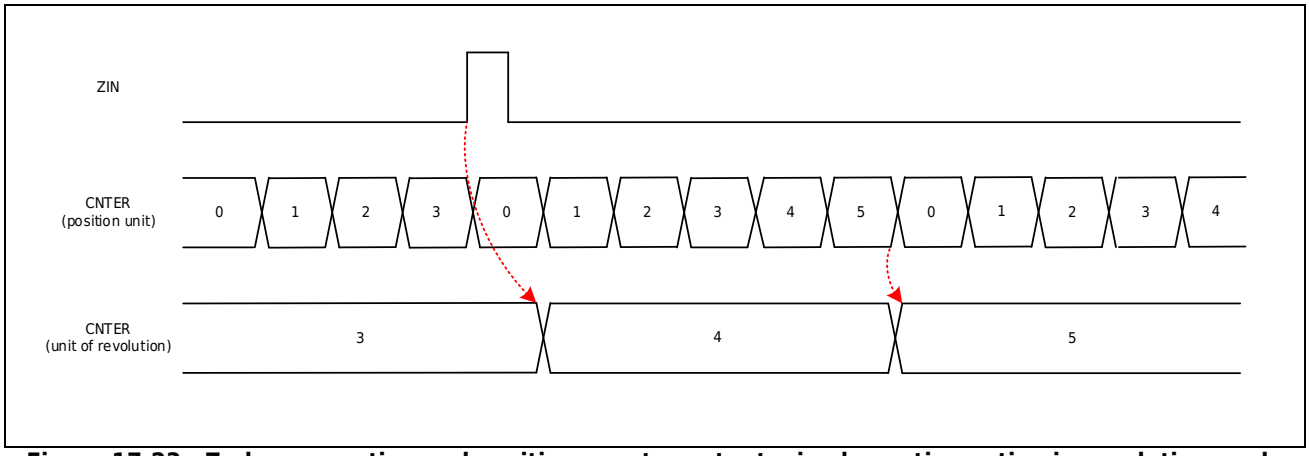


Figure 17-23 Z-phase counting and position counter output mixed counting action in revolution mode

Z phase motion shielding

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

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

Figure 17-24 is when the revolution counting mode is mixed counting, when there is a ZIN phase input within 4 counting cycles after the counting overflow of the position counting unit, the action of the ZIN phase input is invalid, that is, the revolution counting unit does not count, and the position counting unit does not reset ;The subsequent ZIN phase input works normally.

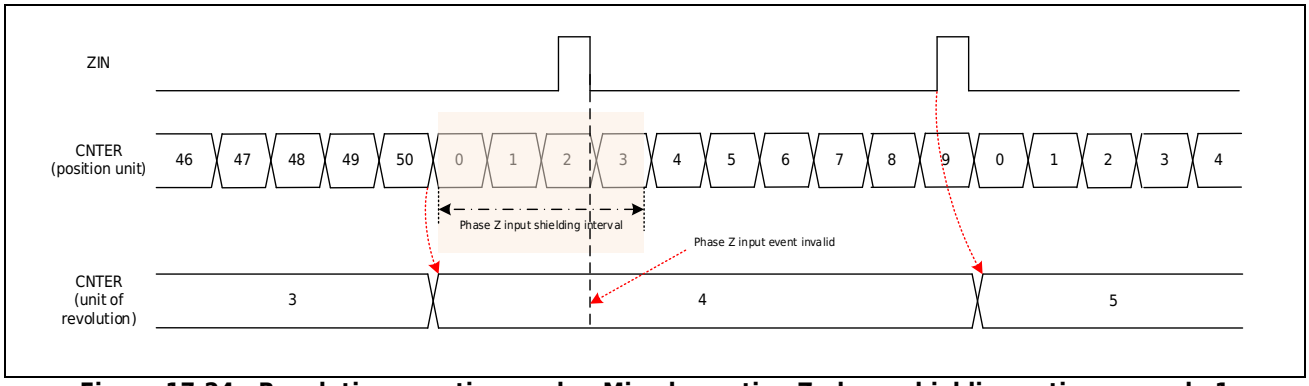


Figure 17-24 Revolution counting mode - Mixed counting Z-phase shielding action example 1

Figure 17-25 shows that when the revolution counting mode is mixed counting, the counting direction changes in the third cycle after the position counting unit overflows, and the masking

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

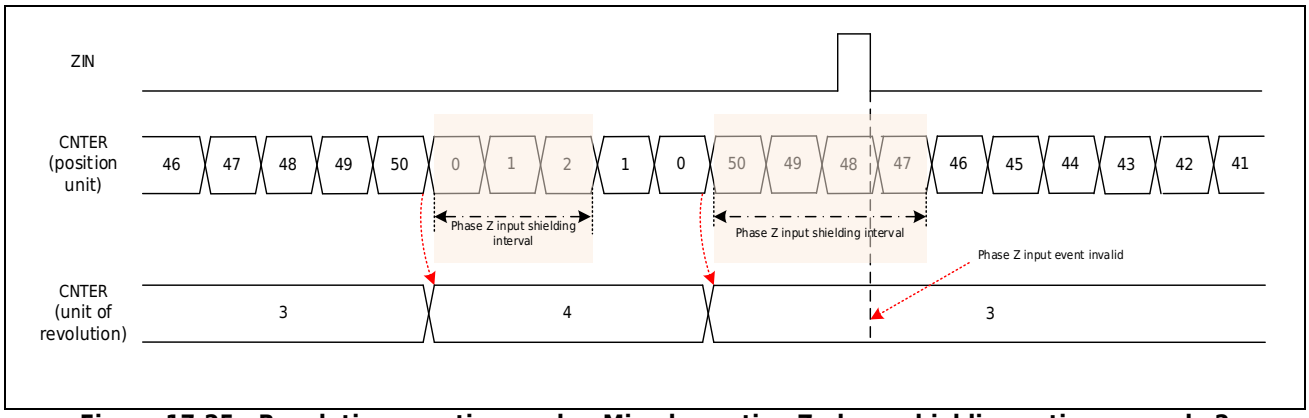


Figure 17-25 Revolution counting mode - Mixed counting Z-phase shielding action example 2

17.2.10 Periodic interval response

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

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

If the timer is stopped and restarted when using the interval response function, please configure VPERR.PCNTE [1:0]=00 before stopping the timer. Otherwise, there may be a deviation in the time when the interval valid request signal is first generated after restarting.

Figure 17-26 shows an example of the operation of the periodic interval valid request signal.

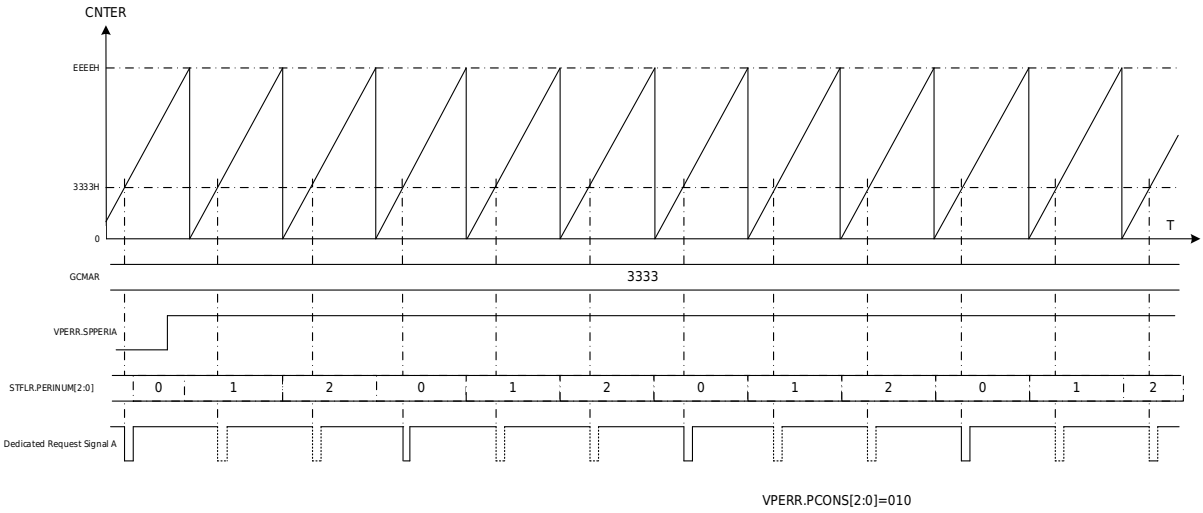


Figure 17-26 Cycle Interval Valid Request Signal Action

17.2.11 Protection mechanism

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

Advanced Timer has 4 shared ports to output invalid event interfaces, these 4 interfaces are connected to 4 groups of brake events output by the brake control module. The abnormal condition events gated on each interface can be set from the brake control, and when abnormal conditions are detected on these interfaces, the control of the general PWM output can be realized.

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

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

17.2.12 Interrupt Description

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

17.2.12.1 Count comparison match interrupt

4 general -purpose comparison reference registers (GCMAR-GCMDR), which can be compared with the count value to generate a comparison match valid signal. When the count compare matches,

the STFLR.CMAF~STFLR.CMDF bits in the status flag register (STFLR) will be set to 1 respectively. At this time, if the corresponding bit in ICONR.INTENA~ICONR.INTEND of the interrupt control register (ICONR) is set to 1 to enable the interrupt, the corresponding interrupt request will also be triggered.

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

17.2.12.2 Count cycle match interrupt

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

17.2.12.3 Dead time error interrupt

the dead time reference register (DTUAR, DTDAR) is loaded into the general comparison reference register (GCMBR), if the cycle limit is exceeded, a dead time error will be generated, and the STFLR.DTEF of the status flag register (STFLR) bit will be set to 1. At this time, if the ICONR.INTENDE bit of the interrupt control register (ICONR) is set to enable the interrupt, the dead time error interrupt will be triggered at this moment.

17.2.13 DMA

Timer supports software and hardware to trigger DMA for data transfer. Data is supported to be written to the timer from other locations, or read and written from the timer to other locations. It can be applied to the automatic handling of data after data capture and the automatic adjustment of the pulse width changing the period value or duty cycle. Each timer has two DMA requests, A request trigger source can choose general comparison capture A, C, special comparison A, count overflow, B request trigger source can choose general comparison capture B, D, special comparison B, count down overflow.

IDREQ	Interrupt Signal of Peripheral
26	TIM4A
27	TIM4B
28	TIM5A
29	TIM5B
30	TIM6A
31	TIM6B

TIMxA _ Trigger source selectable compare capture A, C, count overflow, dedicated compare A

TIMxB _ Trigger source selectable compare capture B, D, count underflow, dedicated compare B

17.2.14 brake protection

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

17.2.14.1 Port brake and software brake

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

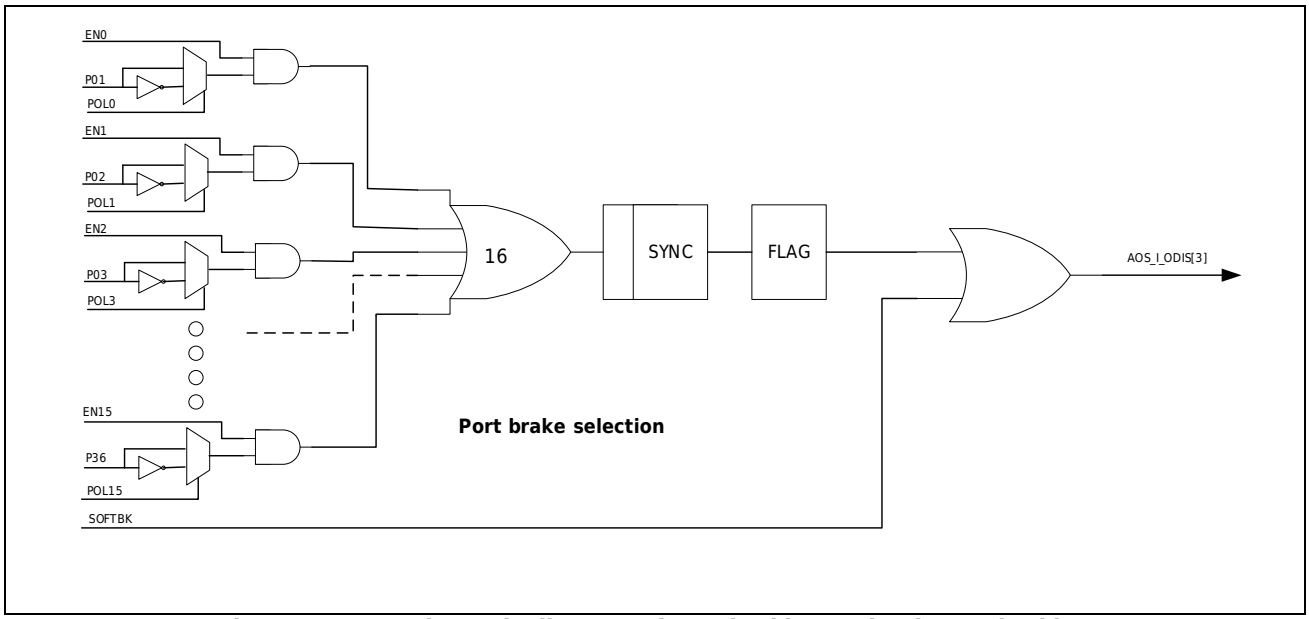


Figure 17-27 Schematic diagram of port braking and software braking

17.2.14.2 Automatic braking in low power mode

the PWM will not work normally after the clock is stopped. Low power consumption mode controls PWM brake as invalid condition 2 of Advanced Timer.

17.2.14.3 Output level same high and same low brake

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

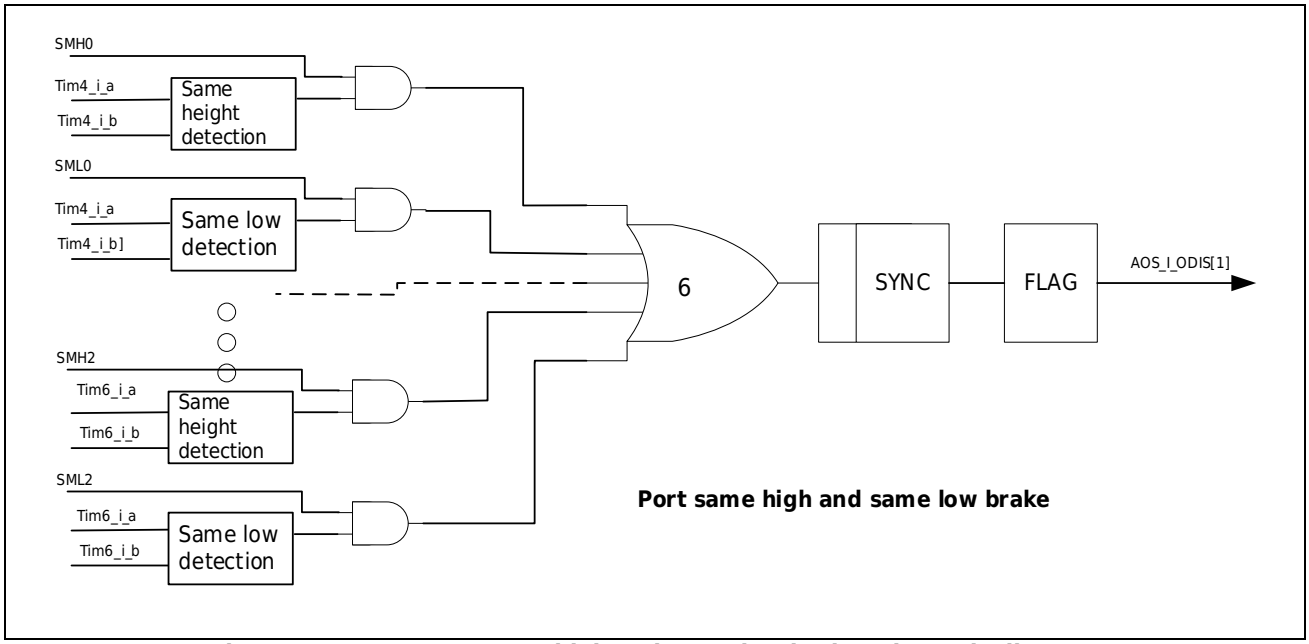


Figure 17-28 Output same high and same low brake schematic diagram

17.2.14.4 VC brake

VC0, VC1, VC2 interrupt flags are enabled as the invalid condition 0 of the Advanced Timer.

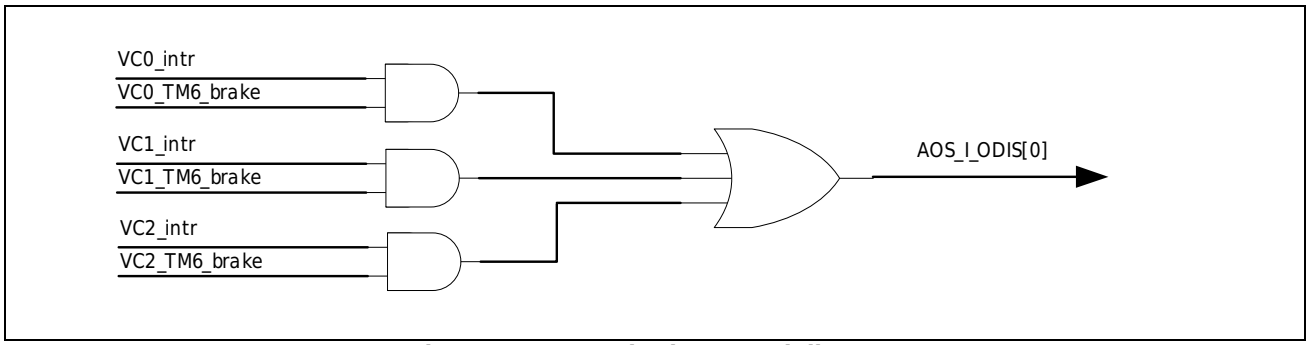


Figure 17-29 VC brake control diagram

17.2.15 interconnection

17.2.15.1 interrupt trigger output

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

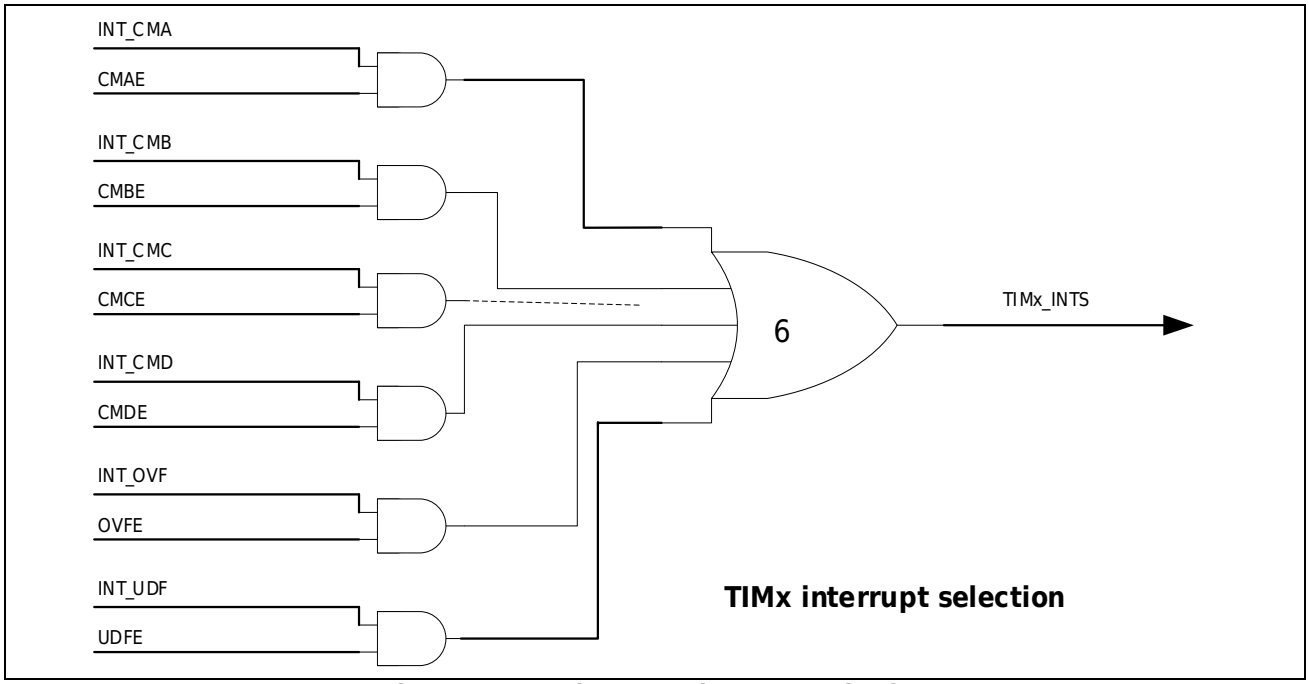


Figure 17-30 Timer4/5/6 interrupt selection

17.2.15.2 AOS trigger

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

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

Table 17-3 AOS source selection

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

	AOS_i_trig0	AOS_i_trig1	AOS_i_trig2	AOS_i_trig3
1001	LPUART0_INT	LPUART0_INT	LPUART0_INT	LPUART0_INT
1010	VC0_INT	VC0_INT	VC0_INT	VC0_INT
1011	VC1_INT	VC1_INT	VC1_INT	VC1_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

17.2.15.3 Port Trigger TRIGA-TRIGD

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

Table 17-4 Port trigger selection

Select control signals to control independently	TRIGA	TRIGB	TRIGC	TRIGD
0000	PA3	PA3	PA3	PA3
0001	PB3	PB3	PB3	PB3
0010	PC3	PC3	PC3	PC3
0011	PD3	PD3	PD3	PD3
0100	PA7	PA7	PA7	PA7
0101	PB7	PB7	PB7	PB7
0110	PC7	PC7	PC7	PC7
0111	PD7	PD7	PD7	PD7
1000	PA11	PA11	PA11	PA11
1001	PB11	PB11	PB11	PB11
1010	PC11	PC11	PC11	PC11
1011	PD1	PD1	PD1	PD1
1100	PA15	PA15	PA15	PA15
1101	PB15	PB15	PB15	PB15
1110	PC5	PC5	PC5	PC5
1111	PD5	PD5	PD5	PD5

17.2.15.4 The comparison output VC is interconnected with Advanced Timer

VC can be interconnected to the capture input of Advanced Timer, and can capture the edge of VC output; VC0 is connected to CHA, VC1 is connected to CHB; the control is in the VC control register.

17.3Register description

CH0 base address 0x40003000

CH1 base address 0x40003400

CH2 base address 0x40003800

Table 17-5 Advanced Timer register list

Register	Offset address	Description
TIMx_CNTER	0x000	General purpose count reference register
TIMx_PERAR	0x004	General purpose period reference register
TIMx_PERBR	0x008	General purpose cycle reference buffer register
TIMx_GCMAR	0x010	General purpose compare A reference value register
TIMx_GCMBR	0x014	General purpose compare B reference value register
TIMx_GCMCR	0x018	General purpose compare C reference value register
TIMx_GCMDR	0x01C	General purpose compare D reference value register
TIMx_SCMAR	0x028	Dedicated compare A reference value register
TIMx_SCMBR	0x02C	Dedicated compare B reference value register
TIMx_DTUAR	0x040	DEAD TIME REFERENCE REGISTER
TIMx_DTDAR	0x044	DEAD TIME REFERENCE REGISTER
TIMx_GCONR	0x050	General control register
TIMx_ICONR	0x054	Interrupt control register
TIMx_PCONR	0x058	Port control register
TIMx_BCONR	0x05C	Cache control register
TIMx_DCONR	0x060	Dead Band Control Register
TIMx_FCONR	0x068	Filter control register
TIMx_VPERR	0x06C	Valid period register
TIMx_STFLR	0x070	Status flag register
TIMx_HSTAR	0x074	Hardware Boot Event Select Register
TIMx_HSTPR	0x078	Hardware Stop Event Select Register
TIMx_HCELR	0x07C	Hardware clear event select register
TIMx_HCPAR	0x080	Hardware Capture Event Select Register
TIMx_HCPBR	0x084	Hardware Capture Event Select Register
TIMx_HCUPR	0x088	Hardware Decrease Event Selection Register
TIMx_HCDOR	0x08C	Hardware Decrease Event Selection Register
TIMx_IFR	0x100	Interrupt Flag Register
TIMx_ICLR	0x104	Interrupt Clear Register
TIMx_CR	0x108	Spread spectrum and interrupt trigger selection register
TIMx_AOSSR	0x110	AOS selection register, shared by three channels
TIMx_AOSCL	0x114	AOS brake flag clear register, shared by three channels

Register	Offset address	Description
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 Enable Register
TIMx_SSTPR	0x3F8	Software Synchronization Stop Register
TIMx_SCLRR	0x3FC	Software synchronous clear register

17.3.1 Common Count Reference Register (TIMx_CNTER)

Address offset: 0x000

Reset value: 0x0000 0000

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

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

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

17.3.2 Universal Period Reference Register (TIMx_PERAR)

Address offset: 0x004

Reset value: 0x0000 FFFF

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PERA[15:0]															
RW															

Bit	Symbol	Functional description
31:16	Reserved	-
15:0	PERA[15:0]	Counting period value, set the counting period value of each round of counting

17.3.3 General purpose period buffer register (TIMx_PERBR)

Address offset: 0x008

Reset value: 0x0000 FFFF

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PERB[15:0]															
RW															

Bit	Symbol	Functional description
31:16	Reserved	-
15:0	PERB[15:0]	Buffer count cycle value, cache value of count cycle

17.3.4 Universal Compare Base Registers (TIMx_GCMAR-GCMDR)

Address offset: 0x0010, 0x0014, 0x0018, 0x001C

Reset value: 0x0000 FFFF

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GCMA-D [15:0]															
RW															

Bit	Symbol	Functional description
31:16	Reserved	-
15:0	GCMA-D [15:0]	Counting comparison reference value, comparison reference value setting, matching signal is valid when it is equal to the count value

17.3.5 Dedicated Compare Reference Registers (TIMx_SCMAR-SCMBR)

Address offset: 0x0028, 0x002C

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
SCMA-B[15:0]															
RW															

Bit	Symbol	Functional description
31:16	Reserved	-
15:0	SCMA-B [15:0]	Counting comparison reference value, comparison reference value setting, matching signal is valid when it is equal to the count value

17.3.6 DEAD TIME REFERENCE REGISTER (TIMx_DTUAR-DTDAR)

Address offset: 0x040, 0x044

Reset value: 0x0000 FFFF

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DTUA/DTDA [15:0]															
RW															

Bit	Symbol	Functional description
31:16	Reserved	-
15:0	DTUA/DA [15:0]	Dead time value, dead time set value

17.3.7 General Control Register (TIMx_GCONR)

Address offset: 0x050

Reset value: 0x00000100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												ZMSK[1:0]		ZMSK POS	ZMSK REV
												RW		RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DIR	Res.	CKDIV[2:0]			MODE[2:0]			START
							RW		RW			RW			RW

Bit	Marking	Functional description
31:20	Reserved	-
19:18	ZMSK[1:0]	Phase Z input muting cycle number Quadrature encoding phase Z input masked count period value 00: Phase Z input shielding function is invalid 01: The Z-phase input is masked within 4 counting cycles after the position count overflows or underflows 10: The Z-phase input is masked within 8 count cycles after the position count overflows or underflows 11: The Z-phase input is masked within 16 count cycles after the position count overflows or underflows
17	ZMSKPOS	Z phase input position counter selection 0: When the Z phase is input, the timer is used as a position counter, and the position counter clearing function works normally during the masking period 1: When the Z phase is input, the timer is used as a position counter, and the function of clearing the position counter is masked during the masking period
16	ZMSKREV	Z phase input revolution counter selection 0: When the Z phase is input, the timer is used as a revolution counter, and the revolution counter counting function works normally during the shielding period 1: When the Z phase is input, the timer is used as a revolution counter, and the counting function of the revolution counter is shielded during the shielding period
15:9	Reserved	-
8	DIR	Counting direction 0: count down; 1: count up
7	Reserved	-
6:4	CKDIV[2:0]	Counting clock selection 000：PCLK0　001：PCLK0/2　010：PCLK0/4　011：PCLK0/8 100：PCLK0/16　101：PCLK0/64　110：PCLK0/256　111：PCLK0/1024
3:1	MODE[2:0]	Counting mode 000: sawtooth wave A mode 100: triangular wave A mode 101: triangular wave B mode Please do not set other values
0	START	Counter start 0: counter off; 1: counter start *Note: This bit will automatically become 0 when the software stop condition or hardware stop condition is valid*

17.3.8 Interrupt Control Register (TIMx_ICONR)

Address offset: 0x054

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												INTENSBD	INTENSBU	INTENSAD	INTENSAU
												RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTENSAMH	INTENSAML	Reserved					INTENDE	INTENUDF	INTENOVF	Reserved		INTEND	INTENC	INTENB	INTENA
RW	RW						RW	RW	RW			RW	RW	RW	RW

Bit	Marking	Function
31:20	Reserved	-
19	INTENSBD	Dedicated down count trigger ADC enable B
18	INTENSBU	Dedicated count up trigger ADC enable B
17	INTENSAD	Dedicated Down Count Trigger ADC Enable A
16	INTENSAU	Dedicated count-up trigger ADC enable A
15	INTENSAMH	Same as high interrupt enable
14	INTENSAML	Same as low interrupt enable
13:9	Reserved	-
8	INTENDE	Dead time error interrupt enable 0: When the dead time is wrong, the interrupt is invalid 1: When the dead time is wrong, the interrupt is enabled
7	INTENUDF	Underflow interrupt enable 0: When an overflow occurs during a sawtooth wave or counts to the valley point during a triangular wave, the interrupt is invalid 1: When an underflow occurs during a sawtooth waveform or counts to a valley point during a triangular waveform, the interrupt is enabled
6	INTENOVF	Overflow interrupt enable 0: When an overflow occurs during a sawtooth wave or counts to the peak point during a triangular wave, the interrupt is invalid 1: When an overflow occurs in a sawtooth wave or counts to the peak point in a triangular wave, the interrupt is enabled
5:4	Reserved	-
3	INTEND	Count match interrupt enable D 0: When the GCMDR register is equal to the count value, the interrupt is invalid 1: When the GCMDR register is equal to the count value, the interrupt is enabled
2	INTENC	Count match interrupt enable C 0: When the GCMCR register is equal to the count value, the interrupt is invalid 1: When the GCMCR register is equal to the count value, the interrupt is enabled
1	INTENB	Count match interrupt enable B 0: When the GCMBR register is equal to the count value, or when a capture input event occurs, the interrupt is invalid 1: The interrupt is enabled when the GCMBR register is equal to the count value, or when a capture input event occurs
0	INTENA	Count Match interrupt enable A 0: When the GCMAR register is equal to the count value, or when a capture input event occurs, the interrupt is invalid 1: The interrupt is enabled when the GCMAR register is equal to the count value, or when a capture input event occurs

17.3.9 Port Control Register (TIMx_PCONR)

Address offset: 0x058

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			DISVALB		DISSELB		OUTENB	PERCB		CMPCB		STASTPSB	STPCB	STACB	CAPCB
			RW		RW		RW	RW		RW		RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DISVALA		DISSELA		OUTENA	PERCA		CMPCA		STASTPSA	STPCA	STACA	CAPCA
			RW		RW		RW	RW		RW		RW	RW	RW	RW

Bit	Marking	Function
31:29	Reserved	-
28:27	DISVALB	CHxB output state control 00: When the condition selected among the forced output invalid conditions 0~3 is met, the CHxB port outputs normally 01: When the selected condition among the forced output invalid conditions 0~3 is met, the CHxB port outputs a high-impedance state 10: When the forced output invalid condition 0~3 is selected, the CHxB port outputs a low level 11: When the selected condition among the forced output invalid conditions 0~3 is met, the CHxB port outputs a high level
26:25	DISSELB	Force output invalid condition selection B 00: select forced output invalid condition 0; 01: select forced output invalid condition 1 10: Select forced output invalid condition 2; 11: Select forced output invalid condition 3
24	OUTENB	Output enable B 0: CHxB port output is invalid when the Advanced Timer function is enabled 1: The CHxB port output is valid when the Advanced Timer function is active
23:22	PERCB	Port status setting when period values matchB 00: When the count value of the counter is equal to the period value, the CHxB port output remains low 01: When the count value of the counter is equal to the period value, the CHxB port output is set to high level 10: When the counter count value is equal to the period value, the CHxB port output is set to the previous state 11: When the count value of the counter is equal to the period value, the CHxB port output is set to the inversion level
21:20	CMPCB	Port status setting when comparison values matchB 00: When the counter count value is equal to GCMBR, the output of CHxB port remains low 01: When the counter count value is equal to GCMBR, the CHxB port output is set to high level 10: When the counter count value is equal to GCMBR, the CHxB port output is set to the previous state 11: When the count value of the counter is equal to GCMBR, the CHxB port output is set to the inverted level
19	STASTPSB	Count start stop port state selection B 0: When counting starts or stops, CHxB port output is determined by STACB, STPCB 1: When counting starts or stops, the CHxB port output is set to the previous state *Note: The counting start here refers to the initial counting start or stop and restart; the counting stop refers to the initial stop or counting start and then stop*
18	STPCB	Count stop port status setting B 0: When counting stops, CHxB port output is set to low level 1: When counting stops, CHxB port output is set to high level
17	STACB	Count start port status setting B 0: When counting starts, CHxB port output is set to low level 1: When counting starts, CHxB port output is set to high level
16	CAPCB	Function mode selection B 0: comparison output function; 1: capture input function
15:13	Reserved	-
12:11	DISVALA	CHxA output state control 00: When the condition selected among the forced output invalid conditions 0~3 is met, the CHxA port outputs normally 01: When the condition selected in the forced output invalid condition 0~3 is met, the CHxA port outputs a high-impedance state 10: When the condition selected in the forced output invalid condition 0~3 is met, the CHxA port outputs a low level 11: When the selected condition among the forced output invalid conditions 0~3 is met, the CHxA port

		outputs a high level
10:9	DISSELA	Force output invalid condition selection A 00: select forced output invalid condition 0; 01: select forced output invalid condition 1 10: Select forced output invalid condition 2; 11: Select forced output invalid condition 3
8	OUTENA	Output enable A 0: CHxA port output is invalid when the Advanced Timer function is active 1: CHxA port output is valid when the Advanced Timer function is active
7:6	PERCA	Port status setting when period values matchA 00: When the count value of the counter is equal to the period value, the CHxA port output remains low 01: When the count value of the counter is equal to the period value, the CHxA port output is set to high level 10: When the counter count value is equal to the period value, the CHxA port output is set to the previous state 11: When the count value of the counter is equal to the period value, the output of the CHxA port is set as an inversion level
5:4	CMPCA	Port State Setting A When Comparing Values Match 00: When the counter count value is equal to GCMAR, the CHxA port output remains low 01: When the counter count value is equal to GCMAR, the CHxA port output is set to high level 10: When the counter count value is equal to GCMAR, the CHxA port output is set to the previous state 11: When the count value of the counter is equal to GCMAR, the CHxA port output is set to the inverted level
3	STASTPSA	Count start stop port state selection A 0: When counting starts or stops, CHxA port output is determined by STACA, STPCA 1: When counting starts or stops, the CHxA port output is set to the previous state *Note: The counting start here refers to the initial counting start or stop and restart; the counting stop refers to the initial stop or counting start and then stop*
2	STPCA	Count stop port state setting A 0: When counting stops, the CHxA port output is set to low level; 1: When counting stops, CHxA port output is set to high level
1	STACA	Count start port status setting A 0: When counting starts, CHxA port output is set to low level 1: When counting starts, CHxA port output is set to high level
0	CAPCA	Functional mode selection A 0: comparison output function; 1: capture input function

17.3.10 Buffer Control Register (TIMx_BCONR)

Address offset: 0x05C

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							BENP	Reserved					BENB	Res.	BENA
							RW						RW		RW

Bit	Marking	Function
31:9	Reserved	-
8	BENP	Period value buffer transfer 0: Buffer transmission is invalid 1: Buffer transfer enable (PERBR->PERAR)
7:3	Reserved	-
2	BENB	General comparison value buffer transfer B 0: Buffer transmission is invalid 1: Buffer transfer enable When comparing output functions: (GCMDR->GCMBR); when capturing input functions: (GCMBR->GCMDR)
1	Reserved	-
0	BENA	General comparison value buffer transfer A 0: Buffer transmission is invalid 1: Buffer transfer enable When comparing output functions: (GCMCR->GCMAR); when capturing input functions: (GCMAR->GCMCR)

17.3.11 Dead Time Control Register (TIMx_DCONR)

Address offset: 0x060

Reset value: 0x0000 0000

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

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

Bit	Marking	Function
31:9	Reserved	-
8	SEPA	Separate settings 0: DTUAR and DTDAR are set separately 1: The value of DTDAR is automatically equal to the value of DTUAR
7:1	Reserved	-
0	DTCEN	Dead zone function 0: Dead zone function is invalid 1: The dead zone function is valid

17.3.12 Filter Control Register (TIMx_FCONR)

Address offset: 0x068

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	NOFICKTD		NOFIENTD	Res.	NOFICKTC		NOFIENTC	Res.	NOFICKTB		NOFIENTB	Res.	NOFICKTA		NOFIENTA
	RW		RW		RW		RW		RW		RW		RW		RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									NOFICKGB		NOFIENGB	Res.	NOFICKGA		NOFIENGA
									RW		RW		RW		RW

Bit	Marking	Function
31	Reserved	-
30:29	NOFICKTD	TRID port filter sampling reference clock selection 00：PCLK0 01：PCLK0/4 10：PCLK0/16 11：PCLK0/64
28	NOFIENTD	TRID port capture input filtering enable, 0 is invalid; 1 is enabled
27	Reserved	-
26:25	NOFICKTC	TRIC port filter sampling reference clock selection 00：PCLK0 01：PCLK0/4 10：PCLK0/16 11：PCLK0/64
24	NOFIENTC	TRIC port capture input filtering enable, 0 is invalid; 1 is enabled
23	Reserved	-
22:21	NOFICKTB	TRIB port filter sampling reference clock selection 00：PCLK0 01：PCLK0/4 10：PCLK0/16 11：PCLK0/64
20	NOFIENTB	TRIB port capture input filtering enable, 0 is invalid; 1 is enabled
19	Reserved	-
18:17	NOFICKTA	TRIA port filter sampling reference clock selection 00：PCLK0 01：PCLK0/4 10：PCLK0/16 11：PCLK0/64
16	NOFIENTA	TRIA port capture input filtering enable, 0 is invalid; 1 is enabled
15:7	Reserved	-
6:5	NOFICKGB	CHxIB port filter sampling reference clock selection 00：PCLK0 01：PCLK0/4 10：PCLK0/16 11：PCLK0/64
4	NOFIENGB	CHxIB port capture input filter enable, 0 is invalid; 1 is enabled
3	Reserved	-
2:1	NOFICKGA	CHxIA port filter sampling reference clock selection 00：PCLK0 01：PCLK0/4 10：PCLK0/16 11：PCLK0/64
0	NOFIENGA	CHxIA port capture input filtering enable, 0 is invalid; 1 is enabled

Note:

– TRIGA-D filter setting is only valid in TIM4, and invalid in Timer5/6.

17.3.13 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	
											RW			RW	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												GEPERID	GEPERIC	GEPERIB	GEPERIA
												RW	RW	RW	RW

Bit	Marking	Function
31:21	Reserved	-
20:18	PCNTS	Valid period selection 000: Valid cycle selection function is invalid 001: Valid every 1 cycle 010: Valid every 2 cycles 011: Valid every 3 cycle 100: Valid every 4 cycles 101: Valid every 5 cycle 110: Valid every 6 cycles 111: Valid every 7 cycle
17:16	PCNTE	Active cycle count condition selection 00: Valid period selection function is invalid 01: The sawtooth wave counts the upper and lower overflow points or the triangular wave trough as the counting condition 10: The sawtooth wave counts the upper and lower overflow points or the triangular wave peak as the counting condition 11: The sawtooth wave counts the upper and lower overflow points or the triangular wave trough and peak as the counting condition
15:4	Reserved	-
3	GEPERID	General signal effective period selection D 0: Valid period selection function is invalid; 1: Valid period selection function is enabled
2	GEPERIC	General signal effective period selection C 0: Valid period selection function is invalid; 1: Valid period selection function is enabled
1	GEPERIB	General signal effective period selection B 0: Valid period selection function is invalid; 1: Valid period selection function is enabled
0	GEPERIA	General signal effective period selection A 0: Valid period selection function is invalid; 1: Valid period selection function is enabled

17.3.14 Status Flag Register (TIMx_STFLR)

Address offset: 0x070

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CMSBDF	CMSBUF	CMSADF	CMSAUF	DTEF	UDFF	OVFF	Reserved		CMDF	CMCF	CMBF	CMAF
			RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW

Bit	Marking	Function
31	DIRF	Counting direction 0: count down 1: count up
30:24	Reserved	-
23:21	VPERNUM	Number of cycles When the effective cycle selection function is enabled, the number of cycles after counting
20:13	Reserved	-
12	CMSBDF	Count down special reference value match B
11	CMSBUF	Count up special reference value match B
10	CMSADF	Count down special base value match A
9	CMSAUF	Up-Count-Specific Baseline Match A
8	DTEF	Dead time error 0: no dead time error occurred; 1: dead time error occurred
7	UDFF	Underflow match 0: Sawtooth underflow does not occur or triangle wave counts to the valley point 1: A sawtooth wave underflow occurs or a triangular wave counts to a valley point
6	OVFF	Overflow match 0: No sawtooth overflow or triangular wave counting to peak 1: Sawtooth overflow occurs or triangular wave counts to peak
5:4	Reserved	-
3	CMDF	Count match D 0: The value of the GCMDR register is not equal to the count value; 1: The value of the GCMDR register is equal to the count value
2	CMCF	Count match C 0: The value of the GCMCR register is not equal to the count value; 1: The value of the GCMCR register is equal to the count value
1	CMBF	Count match B 0: The value of the GCMBR register is not equal to the count value, and the CHxB capture completion action has not occurred 1: The value of the GCMBR register is equal to the count value, or a CHxB capture complete action occurs
0	CMAF	Count Match A 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 a CHxA capture complete action occurs

17.3.15 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														
RW															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSTA15	HSTA14	HSTA13	HSTA12	HSTA11	HSTA10	HSTA9	HSTA8	HSTA7	HSTA6	HSTA5	HSTA4	HSTA3	HSTA2	HSTA1	HSTA0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Function
31	STARTS	Hardware enable 0: hardware startup is invalid 1: Hardware startup is valid *Note: When the hardware boot is valid, the setting of SSTAR is invalid*
30:16	Reserved	-
15	HSTA15	Hardware Start Condition 15: Sampled on falling edge on TIMTRID port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
14	HSTA14	Hardware Start Condition 14: Rising edge sampled on TIMTRID port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
13	HSTA13	Hardware Start Condition 13: Sampled on falling edge on TIMTRIC port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
12	HSTA12	Hardware Start Condition 12: Rising edge sampled on TIMTRIC port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
11	HSTA11	Hardware Start Condition 11: Sampled on falling edge on TIMTRIB port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
10	HSTA10	Hardware Start Condition 10: Rising edge sampled on TIMTRIB port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
9	HSTA9	Hardware start condition 9: TIMTRIA port sampled to falling edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
8	HSTA8	Hardware start condition 8: Sampling to rising edge on TIMTRIA port 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
7	HSTA7	Hardware start condition 7: CHxB port sampled to falling edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
6	HSTA6	Hardware start condition 6: Sampled on CHxB port to rising edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
5	HSTA5	Hardware start condition 5: CHxA port sampled to falling edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
4	HSTA4	Hardware start condition 4: CHxA port sampled to rising edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
3	HSTA3	Hardware start condition 3: Event trigger 3 from AOS is valid 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
2	HSTA2	Hardware start condition 2: Event trigger 2 from AOS is valid 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
1	HSTA1	Hardware start condition 1: Event trigger 1 from AOS is valid 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match
0	HSTA0	Hardware start condition 0: Event trigger 0 from AOS is valid 0: Hardware startup is invalid when conditions match

		1: Hardware startup is valid when conditions match

17.3.16 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														
RW															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSTP15	HSTA14	HSTP13	HSTP12	HSTP11	HSTP10	HSTP9	HSTP8	HSTP7	HSTP6	HSTP5	HSTP4	HSTP3	HSTP2	HSTP1	HSTP0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Function
31	STOPS	Hardware disable 0: hardware stop is invalid 1: Hardware stop is valid *Note: When the hardware stop is valid, the software stop setting is invalid*
30:16	Reserved	-
15	HSTP15	Hardware Stop Condition 15: Falling edge sampled on TIMTRID port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
14	HSTP14	Hardware Stop Condition 14: Rising edge sampled on TIMTRID port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
13	HSTP13	Hardware Stop Condition 13: Falling edge sampled on TIMTRIC port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
12	HSTP12	Hardware Stop Condition 12: Rising edge sampled on TIMTRIC port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
11	HSTP11	Hardware Stop Condition 11: Falling edge sampled on TIMTRIB port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
10	HSTP10	Hardware Stop Condition 10: Rising edge sampled on TIMTRIB port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
9	HSTP9	Hardware Stop Condition 9: Falling edge sampled on TIMTRIA port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
8	HSTP8	Hardware Stop Condition 8: Rising edge sampled on TIMTRIA port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
7	HSTP7	Hardware Stop Condition 7: Sampled on CHxB port to falling edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
6	HSTP6	Hardware Stop Condition 6: Sampled to rising edge on CHxB port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
5	HSTP5	Hardware stop condition 5: CHxA port sampled to falling edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
4	HSTP4	Hardware Stop Condition 4: Sampled to rising edge on CHxA port 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
3	HSTP3	Hardware stop condition 3: Event trigger 3 from AOS is valid 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
2	HSTP2	Hardware stop condition 2: Event trigger 2 from AOS is valid 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
1	HSTP1	Hardware stop condition 1: Event trigger 1 from AOS is valid 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match
0	HSTP0	Hardware stop condition 0: Event trigger 0 from AOS is valid 0: Hardware stop invalid when condition matches

		1: Hardware stops working when conditions match

17.3.17 Hardware Clear Event Select Register (TIMx_HCELR)

Address offset: 0x07C

Reset value: 0x0000 0000

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

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
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Function
31	CLEARS	Hardware clear enable 0: Hardware reset is invalid 1: Hardware clear is valid *Note: When the hardware clear is valid, the setting of software clear is invalid*
30:16	Reserved	-
15	HCEL15	Hardware clear Condition 15: Sampled on falling edge on TIMTRID port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
14	HCEL14	Hardware clear Condition 14: Sampled on rising edge on TIMTRID port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
13	HCEL13	Hardware clear Condition 13: Sampled on falling edge on TIMTRIC port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
12	HCEL12	Hardware clear condition 12: Sampled on rising edge on TIMTRIC port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
11	HCEL11	Hardware clear condition 11: TIMTRIB port sampled on falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
10	HCEL10	Hardware clear condition 10: sampled on rising edge on TIMTRIB port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
9	HCEL9	Hardware clear condition 9: TIMTRIA port sampled to falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
8	HCEL8	Hardware clear condition 8: sampled on rising edge on TIMTRIA port 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
7	HCEL7	Hardware clear condition 7: CHxB port sampled to falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
6	HCEL6	Hardware clear condition 6: CHxB port sampled to rising edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
5	HCEL5	Hardware clear condition 5: CHxA port sampled to falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
4	HCEL4	Hardware clear condition 4: CHxA port sampled to rising edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
3	HCEL3	Hardware clear condition 3: Event trigger 3 from AOS is valid 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
2	HCEL2	Hardware clear condition 2: Event trigger 2 from AOS is valid 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
1	HCEL1	Hardware clear condition 1: Event trigger 1 from AOS is valid 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match
0	HCEL0	Hardware clear condition 0: Event trigger 0 from AOS is valid 0: Hardware zeroing is invalid when conditions match

		1: Hardware zeroing is valid when conditions match

17.3.18 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
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Function
31:16	Reserved	-
15	HCPA15	Hardware Capture A Condition 15: Sampled to falling edge on TIMTRID port 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
14	HCPA14	Hardware Capture A Condition 14: Sample to rising edge on TIMTRID port 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
13	HCPA13	Hardware Capture A Condition 13: Sample to falling edge on TIMTRIC port 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
12	HCPA12	Hardware Capture A Condition 12: Sample to rising edge on TIMTRIC port 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
11	HCPA11	Hardware Capture A Condition 11: Sampled to falling edge on TIMTRIB port 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
10	HCPA10	Hardware Capture A Condition 10: Sampling on TIMTRIB port to rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
9	HCPA9	Hardware capture A condition 9: TIMTRIA port sampled to falling edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
8	HCPA8	Hardware Capture A Condition 8: Sampling on TIMTRIA port to rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
7	HCPA7	Hardware Capture A Condition 7: Sampled on CHxB port to falling edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
6	HCPA6	Hardware Capture A Condition 6: Sampled on CHxB port to rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
5	HCPA5	Hardware Capture A Condition 5: Sampled on CHxA port to falling edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
4	HCPA4	Hardware Capture A Condition 4: Sampled on CHxA port to rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
3	HCPA3	Hardware capture A condition 3: Event trigger 3 from AOS is valid 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
2	HCPA2	Hardware capture A condition 2: Event trigger 2 from AOS is valid 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
1	HCPA1	Hardware capture A condition 1: Event trigger 1 from AOS is valid 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid
0	HCPA0	Hardware capture A condition 0: Event trigger 0 from AOS is valid 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid

17.3.19 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
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Function
31:16	Reserved	-
15	HCPB15	Hardware Capture B Condition 15: Sample to falling edge on TIMTRID port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
14	HCPB14	Hardware Capture B Condition 14: Sampled to rising edge on TIMTRID port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
13	HCPB13	Hardware Capture B Condition 13: Sample to falling edge on TIMTRIC port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
12	HCPB12	Hardware Capture B Condition 12: Sampled to rising edge on TIMTRIC port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
11	HCPB11	Hardware Capture B Condition 11: Sampled to falling edge on TIMTRIB port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
10	HCPB10	Hardware Capture B Condition 10: Sampled to rising edge on TIMTRIB port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
9	HCPB9	Hardware capture B condition 9: TIMTRIA port sampled to falling edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
8	HCPB8	Hardware Capture B Condition 8: Sampling on TIMTRIA port to rising edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
7	HCPB7	Hardware Capture B Condition 7: Sampled on CHxB port to falling edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
6	HCPB6	Hardware Capture B Condition 6: Sampled to rising edge on CHxB port 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
5	HCPB5	Hardware Capture B Condition 5: Sampled on CHxA port to falling edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
4	HCPB4	Hardware Capture B Condition 4: Sampled on CHxA port to rising edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
3	HCPB3	Hardware capture B condition 3: Event trigger 3 from AOS is valid 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
2	HCPB2	Hardware capture B condition 2: Event trigger 2 from AOS is valid 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
1	HCPB1	Hardware capture B condition 1: Event trigger 1 from AOS is valid 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid
0	HCPB0	Hardware capture B condition 0: Event trigger 0 from AOS is valid 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid

17.3.20 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												HCUP19	HCUP18	HCUP17	HCUP16
												RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCUP15	HCUP14	HCUP13	HCUP12	HCUP11	HCUP10	HCUP9	HCUP8	HCUP7	HCUP6	HCUP5	HCUP4	HCUP3	HCUP2	HCUP1	HCUP0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Function
31:20	Reserved	-
19	HCUP19	Hardware increment condition: Event trigger 3 from AOS is valid 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
18	HCUP18	Hardware increment condition: Event trigger 2 from AOS is valid 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
17	HCUP17	Hardware increment condition: Event trigger 1 from AOS is valid 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
16	HCUP16	Hardware increment condition: the event from AOS triggers 0 to be valid 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
15	HCUP15	Hardware increment condition: TIMTRID port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
14	HCUP14	Hardware increment condition: TIMTRID port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
13	HCUP13	Hardware increment condition: TIMTRIC port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
12	HCUP12	Hardware increment condition: TIMTRIC port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
11	HCUP11	Hardware increment condition: TIMTRIB port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
10	HCUP10	Hardware increment condition: TIMTRIB port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
9	HCUP9	Hardware increment condition: TIMTRIA port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
8	HCUP8	Hardware increment condition: TIMTRIA port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
7	HCUP7	Hardware increment condition: When the CHxB port is high level, the CHxA port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
6	HCUP6	Hardware increment condition: When the CHxB port is high level, the CHxA port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
5	HCUP5	Hardware increment condition: When the CHxB port is low level, the CHxA port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
4	HCUP4	Hardware increment condition: When the CHxB port is low level, the CHxA port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
3	HCUP3	Hardware increment condition: When the CHxA port is high level, the CHxB port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches

2	HCUP2	Hardware increment condition: When the CHxA port is high level, the CHxB port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
1	HCUP1	Hardware increment condition: When the CHxA port is low level, the CHxB port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches
0	HCUP0	Hardware increment condition: When the CHxA port is low level, the CHxB port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches

17.3.21 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
Reserved												HCDO 19	HCDO 18	HCDO 17	HCDO 16
												RW	RW	RW	RW

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

Bit	Marking	Function
31:20	Reserved	-
19	HCDO19	Hardware decrement condition: Event trigger 3 from AOS is valid 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
18	HCDO18	Hardware decrement condition: Event trigger 2 from AOS is valid 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
17	HCDO17	Hardware decrement condition: Event trigger 1 from AOS is valid 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
16	HCDO16	Hardware decrement condition: the event from AOS triggers 0 to be valid 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
15	HCDO15	Hardware decrement condition: TIMTRID port sampled to falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
14	HCDO14	Hardware decrement condition: Sampled to rising edge on TIMTRID port 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
13	HCDO13	Hardware decrement condition: Sampled to falling edge on TIMTRIC port 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
12	HCDO12	Hardware decrement condition: Sampled on TIMTRIC port to rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
11	HCDO11	Hardware decrement condition: TIMTRIB port sampled to falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
10	HCDO10	Hardware decrement condition: Sampled on TIMTRIB port to rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
9	HCDO9	Hardware decrement condition: TIMTRIA port sampled to falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
8	HCDO8	Hardware decrement condition: Sampled to rising edge on TIMTRIA port 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
7	HCDO7	Hardware decrement condition: When CHxB port is high level, CHxA port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
6	HCDO6	Hardware decrement condition: When CHxB port is high level, CHxA port is sampled to a rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
5	HCDO5	Hardware decrement condition: When the CHxB port is low, the CHxA port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
4	HCDO4	Hardware decrement condition: When CHxB port is low level, CHxA port is sampled to a rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match

3	HCDO3	Hardware decrement condition: When the CHxA port is high, the CHxB port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
2	HCDO2	Hardware decrement condition: When CHxA port is high level, CHxB port is sampled to a rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
1	HCDO1	Hardware decrement condition: When the CHxA port is low, the CHxB port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match
0	HCDO0	Hardware decrement condition: When CHxA port is low level, CHxB port is sampled to a rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match

17.3.22 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
													RW	RW	RW

Bit	Marking	Function
31:3	Reserved	
2	SSTA2	Timer6 software start 0: Software startup is invalid 1: Software startup enable
1	SSTA1	Timer5 software start 0: Software startup is invalid 1: Software startup enable
0	SSTA0	Timer4 software start 0: Software startup is invalid 1: Software startup enable

17.3.23 Software Sync Stop Register (TIMx_SSTPR)

Address offset: 0x3F8

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													SSTP2	SSTP1	SSTP0
													RW	RW	RW

Bit	Marking	Function
31:3	Reserved	
2	SSTP2	Timer6 software stop 0: Software stop is invalid 1: Software stop enable
1	SSTP1	Timer5 software stop 0: Software stop is invalid 1: Software stop enable
0	SSTP0	Timer4 software stop 0: Software stop is invalid 1: Software stop enable

17.3.24 Software Synchronous Clear Register (TIMx_SCLRR)

Address offset: 0x3FC

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													SCLR2	SCLR1	SCLR0
													RW	RW	RW

Bit	Marking	Function
31:3	Reserved	
2	SCLR2	Timer6 software reset 0: Software reset is invalid 1: Software clear enable
1	SCLR1	Timer5 software reset 0: Software reset is invalid 1: Software clear enable
0	SCLR0	Timer4 software reset 0: Software reset is invalid 1: Software clear enable

17.3.25 Interrupt Flag Register (TIMx_IFR)

Address offset: 0x100

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAMHF	SAMLF	Reserved					DTEF	UDFF	OVFF	Reserved		CMDF	CMCF	CMBF	CMAF
RO	RO						RO	RO	RO			RO	RO	RO	RO

Bit	Marking	Function
31:16	Reserved	-
15	SAMHF	CHxA/B port high state interrupt flag 0: No simultaneous high level on CHxA and CHxB ports 1: High level on both CHxA and CHxB ports
14	SAMLF	CHxA/B port low state interrupt flag 0: No simultaneous low on CHxA and CHxB ports 1: Low level on both CHxA and CHxB ports
13:9	Reserved	-
8	DTEF	Dead Time Error Interrupt Flag 0: no dead time error occurred; 1: dead time error occurred
7	UDFF	Underflow match interrupt flag 0: Sawtooth underflow does not occur or triangle wave counts to the valley point 1: A sawtooth wave underflow occurs or a triangular wave counts to a valley point
6	OVFF	Overflow match interrupt flag 0: No sawtooth overflow or triangular wave counting to peak 1: Sawtooth overflow occurs or triangular wave counts to peak
5:4	Reserved	-
3	CMDF	Count match D interrupt flag 0: The value of the GCMDR register is not equal to the count value; 1: The value of the GCMDR register is equal to the count value
2	CMCF	Count match C interrupt flag 0: The value of the GCMCR register is not equal to the count value; 1: The value of the GCMCR register is equal to the count value
1	CMBF	count match B interrupt flag 0: The value of the GCMBR register is not equal to the count value, and the CHxB capture completion action has not occurred 1: The value of the GCMBR register is equal to the count value, or a CHxB capture complete action occurs
0	CMAF	count match A interrupt flag 0: The value of the 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 a CHxA capture complete action occurs

17.3.26 Interrupt Flag Clear Register (TIMx_ICLR)

Address offset: 0x104

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAMHC	SAMLC	Reserved					DTEC	UDFC	OVFC	Reserved		CMDC	CMCC	CMBC	CMAC
R1W0	R1W0						R1W0	R1W0	R1W0			R1W0	R1W0	R1W0	R1W0

Bit	Marking	Function
31:16	Reserved	-
15	SAMHC	CHxA/B port high status interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
14	SAMLC	CHxA/B port low state interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
13:9	Reserved	-
8	DTEC	The dead time error interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
7	UDFC	The underflow match interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
6	OVFC	The overflow match interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
5:4	Reserved	-
3	CMDC	Count match D interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
2	CMCC	The count matches the C interrupt flag to clear, writing 1 is invalid, writing 0 to clear the corresponding interrupt
1	CMBC	Count match B interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
0	CMAC	Count match A interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt

17.3.27 Spread spectrum and interrupt trigger selection (TIMx_CR)

Address offset: 0x108

Reset value: 0x0000 0300

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									Dma_s_cmb	Dma_s_cma	Dma_g_udf	Dma_g_ovf	Reserved		Dma_g_cmd
									RW	RW	RW	RW			RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Dma_g_cmc	Dma_g_cmb	Dma_g_cma	CMSBE	CMSAE	DITENS	DITENB	DITNA	UDFE	OVFE	Reserved		CMDE	CMCE	CMBE	CMAE
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW

Bit	Marking	Function
31:23	Reserved	-
22	Dma_s_cmb	Dedicated compare DMA enable
21	Dma_s_cma	Dedicated compare DMA enable
20	Dma_g_udf	Underflow DMA enable
19	Dma_g_ovf	Overflow DMA enable
18:17	Reserved	
16	Dma_g_cmd	General purpose compare DMA enable
15	Dma_g_cmc	General purpose compare DMA enable
14	Dma_g_cmb	General purpose compare DMA enable
13	Dma_g_cma	General purpose compare DMA enable
12	CMSBE	Dedicated comparison reference match B enable to trigger ADC
11	CMSAE	Dedicated comparison reference match A enable to trigger ADC
10	DITENS	PWM spread spectrum count selection 0: select underflow, 1: select overflow
9	DITENB	PWM channel B spread spectrum enable 0: Enable invalid, 1: Enable valid, change PWM output delay every cycle
9	DITENA	PWM channel A spread spectrum enable 0: Enable invalid, 1: Enable valid, change PWM output delay every cycle
7	UDFE	Underflow match enables ADC trigger 0: Enable is invalid, 1: Enable is valid, this match can control ADC/AOS_i_tirg
6	OVFE	Overflow match enable to trigger ADC 0: Enable is invalid, 1: Enable is valid, this match 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 match can control ADC/AOS_i_tirg
2	CMCE	Count Match C Enable Trigger ADC 0: Enable is invalid, 1: Enable is valid, this match can control ADC/AOS_i_tirg
1	CMBE	Count match B enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this match can control ADC/AOS_i_tirg
0	CMAE	Count match A enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this match can control ADC/AOS_i_tirg

17.3.28 AOS Selection Control Register (TIMx_AOSSR)

Address offset: 0x110

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SMH2	SMH1	SMH0	SML2	SML1	SML0	SOFTBK	Reserved		BFILTEN	BFILTS		FSAME	FBRAKE
		RW	RW	RW	RW	RW	RW	RW			RW	RW		R	R

Bit	Marking	Function
31:14	Reserved	-
13	SMH2	Same height selection for channel 2 0: The selection is invalid, 1: The selection is valid, AOS_i_odis[1] appears when the same high
12	SMH1	Channel 1 same height selection 0: The selection is invalid, 1: The selection is valid, AOS_i_odis[1] appears when the same high
11	SMH0	Channel 0 same high selection 0: The selection is invalid, 1: The selection is valid, AOS_i_odis[1] appears when the same high
10	SML2	Channel 2 with low selection 0: The selection is invalid, 1: The selection is valid, AOS_i_odis[1] appears when the same low
9	SML1	Channel 1 with low selection 0: The selection is invalid, 1: The selection is valid, AOS_i_odis[1] appears when the same low
8	SML0	Channel 0 with low selection 0: The selection is invalid, 1: The selection is valid, AOS_i_odis[1] appears when the same low
7	SOFTBK	Software brake: Write 1 to realize software brake
13	Reserved	-
4	BFILTEN	Port brake filter enable
3:2	BFILTS	Port brake filter clock selection
1	FSAME	Same high and same low brake flag, read only
0	FBRAKE	Port brake flag, read only

17.3.29 AOS Selection Control Register Flag Clear (TIMx_AOSCL)

Address offset: 0x114

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

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

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

Bit	Marking	Function
31:2	Reserved	-
1	FSAME	Same high and same low brake flag to clear, write 0 to clear, write 1 to be invalid, read constant to 1
0	FBRAKE	The port brake flag is cleared, write 0 to clear, write 1 to be invalid, read 1

17.3.30 Port Brake Control Register (TIMx_PTBKS)

Address offset: 0x118

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Function
31:16	Reserved	-
15	EN15	PD5 brake port enable: 1 selection, 0 invalid
14	EN14	PC15 brake port enable: 1 selection, 0 invalid
13	EN13	PB15 brake port enable: 1 selection, 0 invalid
12	EN12	PA15 brake port enable: 1 selection, 0 invalid
11	EN11	PD1 brake port enable: 1 selection, 0 invalid
10	EN10	PC11 brake port enable: 1 selection, 0 invalid
9	EN9	PB11 brake port enable: 1 selection, 0 invalid
8	EN8	PA11 brake port enable: 1 selection, 0 invalid
7	EN7	PD7 brake port enable: 1 selected, 0 invalid
6	EN6	PC7 brake port enable: 1 selection, 0 invalid
5	EN5	PB7 brake port enable: 1 selection, 0 invalid
4	EN4	PA7 brake port enable: 1 selection, 0 invalid
3	EN3	PD3 brake port enable: 1 selection, 0 invalid
2	EN2	PC3 brake port enable: 1 selection, 0 invalid
1	EN1	PB3 brake port enable: 1 selection, 0 invalid
0	EN0	PA3 brake port enable: 1 selection, 0 invalid

17.3.31 Port Trigger Control Register (TIMx_TTRIG)

Address offset: 0x11C

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIGDS				TRIGCS				TRIGBS				TRIGAS			
RW				RW				RW				RW			

Bit	Marking	Function
31:16	Reserved	-
15:12	TRIGDS	TIMx trigger D port selection
11:8	TRIGCS	TIMx trigger C port selection
7:4	TRIGBS	TIMx trigger B port selection
3:0	TRIGAS	TIMx trigger A port selection

The control signal and port selection are as follows

0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
PA3	PB3	PC3	PD3	PA7	PB7	PC7	PD7	PA11	PB11	PC11	PD1	PA15	PB15	PC15	PD5

17.3.32 AOS Trigger Control Register (TIMx_ITRIG)

Address offset: 0x120

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IAOS3S				IAOS2S				IAOS1S				IAOS0S			
RW				RW				RW				RW			

Bit	Marking	Function
31:16	Reserved	-
15:12	IAOS3S	TIMx AOS3 trigger source selection
11:8	IAOS2S	TIMx AOS2 trigger source selection
7:4	IAOS1S	TIMx AOS1 trigger source selection
3:0	IAOS0S	TIMx AOS0 trigger source selection

The control signal (IAOSxS) and interrupt source selection are as follows (x=0,1,2,3)

0000	0001	0010	0011	0100	0101	0110	0111
TIM0_INT	TIM1_INT	TIM2_INT	LPTIMER_INT	TIM4_INTS	TIM5_INTS	TIM6_INTS	UART0_INT
1000	1001	1010	1011	1100	1101	1110	1111
UART1_INT	LPUART0_INT	VC0_INT	VC1_INT	RTC_INT	PCA_INT	SPI_INT	ADC_INT

17.3.33 Port Brake Polarity Control Register (TIMx_PTBKP)

Address offset: 0x124

Reset value: 0x0000 0000

Timer4/5/6 use the same physical register. After any one timer is changed, the values of the other two timers will be changed at the same time.

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POL15	POL14	POL13	POL12	POL11	POL10	POL9	POL8	POL7	POL6	POL5	POL4	POL3	POL2	POL1	POL0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Function
31:16	Reserved	-
15	POL15	Polarity selection of PD5 brake port: 1 is active at low level, 0 is active at high level
14	POL14	Polarity selection of PC15 brake port: 1 is active at low level, 0 is active at high level
13	POL13	Polarity selection of PB15 brake port: 1 is active at low level, 0 is active at high level
12	POL12	Polarity selection of PA15 brake port: 1 is active at low level, 0 is active at high level
11	POL11	Polarity selection of PD1 brake port: 1 is active at low level, 0 is active at high level
10	POL10	Polarity selection of PC11 brake port: 1 is active at low level, 0 is active at high level
9	POL9	Polarity selection of PB11 brake port: 1 is active at low level, 0 is active at high level
8	POL8	Polarity selection of PA11 brake port: 1 is active at low level, 0 is active at high level
7	POL7	Polarity selection of PD7 brake port: 1 is active at low level, 0 is active at high level
6	POL6	Polarity selection of PC7 brake port: 1 is active at low level, 0 is active at high level
5	POL5	Polarity selection of PB7 brake port: 1 is active at low level, 0 is active at high level
4	POL4	Polarity selection of PA7 brake port: 1 is active at low level, 0 is active at high level
3	POL3	Polarity selection of PD3 brake port: 1 is active at low level, 0 is active at high level
2	POL2	Polarity selection of PC3 brake port: 1 is active at low level, 0 is active at high level
1	POL1	Polarity selection of PB3 brake port: 1 is active at low level, 0 is active at high level
0	POL0	Polarity selection of PA3 brake port: 1 is active at low level, 0 is active at high level

18 Real Time Clock (RTC)

18.1 Introduction to Real Time Clocks

The real-time clock / calendar provides seconds, minutes, hours, day, week, month, and year information, and the number of days in a month and leap year can be automatically adjusted. Clock operation can be done in 24 or 12 hour format via the AM/PM register bits. Table 18-1 shows its basic characteristics. It has a periodic interrupt with a minimum period of 0.5S. If you need a shorter interrupt period, please use LPTIM to generate an interrupt.

Table 18-1 Basic characteristics of RTC

Timer	Off-chip low-speed crystal oscillator XTL (32.768kHz) On-chip low-speed oscillator RCL (32kHz, 1% precision) Off-chip high-speed crystal oscillator XTH
Basic functions	Can calculate seconds, minutes, hours, days, weeks, months, years between 00 and 99
	Automatic leap year adjustment
	Configurable to 24 or 12 hour format
	Programmable start or stop
	With alarm clock function
	With high-precision 1Hz square wave signal output
Interrupt	Periodic interrupt with a minimum period of 0.5S
	Alarm clock interrupt with down-to-the-second precision

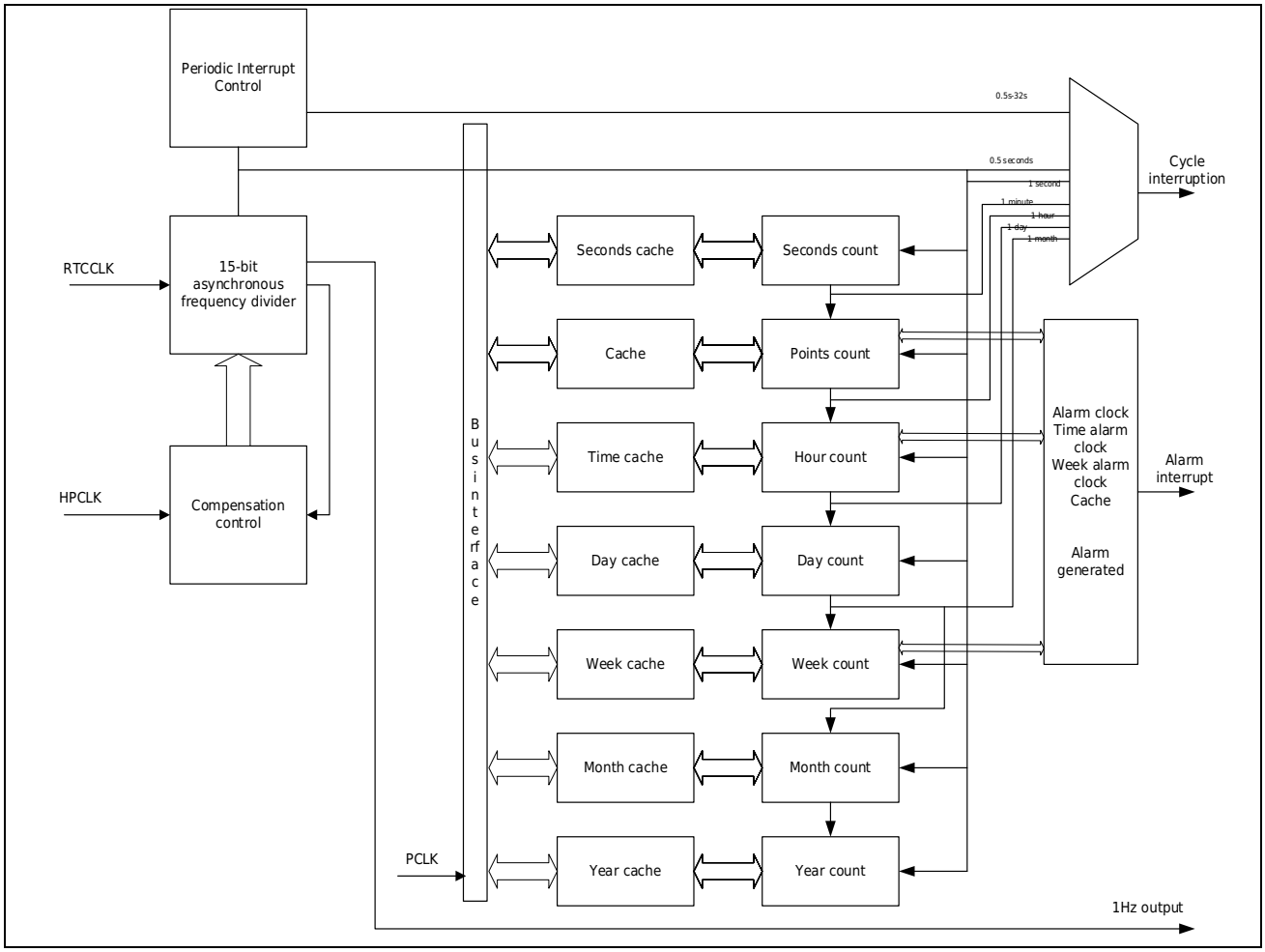


Figure 18-1 RTC Block Diagram

18.2Real Time Clock Functional Description

The clock source of the real-time clock can be configured as an external low-speed crystal oscillator, an external high-speed crystal oscillator, or an internal low-speed RC; the external low-speed crystal oscillator is used by default. The control registers RTC_CR0, RTC_CR1 and RTC_COMPEN are only controlled by power-on reset, and other reset sources cannot reset these three control registers. The power-on status of other data registers is uncertain, and needs to be initialized after power-on, and will not be affected by any reset.

All the date and time values written and read by the software are BCD codes, and there is no need to convert hexadecimal to decimal.

Any invalid date time will not be able to be written, such as 32nd, 25th, 70th second, month B, etc.

18.2.1 Power-on setting

The RTC is reset once after power-on. When the system is not powered off, various external reset requests cannot reset the RTC, and the RTC will always be in the counting state. After power-on, after setting the calendar initial value, alarm clock setting, error compensation, interrupt, etc., start the RTC.

18.2.2 RTC count start setting

1. Set RTC_CR0.START=0, counting stops;
2. Set RTC_CR0.AMPM and RTC_CR0.PRDS, RTC_CR0.PRDX set time and interrupt cycle;
3. Set RTC_CR1.CKSEL to select the timing clock of RTC;
4. Set the calendar counting registers for seconds, minutes, hours, weeks, days, months, and years;
5. When clock error compensation is required, set the counting clock error compensation register RTC_COMPEN;
6. Clear the interrupt flag bit RTC_CR1.ALMF, RTC_CR1.PRDF, and enable the interrupt;
7. Set RTC_CR0.START=1, counting starts.

18.2.3 System low power mode switching

After the RTC counting starts, if the system switches to the low power consumption mode immediately, please perform any of the following confirmations before switching the mode.

The control register is in the system control register SYSCTRL1.RTC_LPW

After setting RTC_CR0.START=1, switch modes after more than 2 RTC count clocks.

After setting RTC_CR0.START=1, set RTC_CR1.WAIT=1 and query RTC_CR1.WAITF=1.

Then set RTC_CR1.WAIT=0, query RTC_CR1.WAITF=0 and then switch modes.

In RTC low power mode, RTC registers cannot be read or written. In low power mode, the RTC consumes less current.

Switching low power mode while RTC is running does not need to wait.

18.2.4 Read count register

There are three ways to read the count register:

Mode 1: Read mode 1 at any time

1. Set RTC_CR1.WAIT=1, stop the calendar register counting, and enter the read and write mode;
2. Query until RTC_CR1.WAITF=1;
3. Read the second, minute, hour, week, day, month, year count register value;
4. Set RTC_CR1.WAIT=0, the counter counts;
5. Query until RTC_CR1.WAITF=0.

Mode 2: Read mode 2 at any time

1. Read minutes, hours, weeks, days, months, and year counting register values;
2. Read out the second count register value;
3. Read the second count register value again;

4. Judging whether the reading values of the two seconds are the same, if they are different, start from the first step again, and end the same reading.

Mode 3: Interrupt read mode

Read the second, minute, hour, week, day, month, year count register value in the RTC cycle interrupt service. Because after the interrupt occurs, it will take at least 0.5s for the next data change.

18.2.5 Write count register

1. Set RTC_CR1.WAIT=1, stop the calendar register counting, and enter the read and write mode;
2. Query until RTC_CR1.WAITF=1;
3. Write the second, minute, hour, week, day, month, year count register value;
4. Set RTC_CR1.WAIT=0, the counter starts counting again. Note: All write operations must be completed within 1 second;
5. Query until RTC_CR1.WAITF=0.

WAIT to write seconds, minutes, hours, weeks, days, months, and year count registers in RTC disabled mode.

Note:

– Changing the seconds register in the counting mode will reset the seconds counting, writing minutes, hours, weeks, days, months, years counting register values will not affect the RTC counting.

18.2.6 Alarm clock setting

1. Set RTC_CR1.ALMEN=0, the alarm clock is disabled;
2. Set RTC_CR1.ALMIE=1, the alarm clock interruption permission;
3. Second alarm clock RTC_ALMSEC, minute alarm clock RTC_ALMMIN, hour alarm clock RTC_ALMHOUR, week alarm clock RTC_ALMWEEK setting;
4. Set RTC_CR1.ALMEN=1, alarm clock permission;
5. Wait for an interrupt to occur;
6. Since the alarm clock interrupt and the fixed-period interrupt share the interrupt request signal, when RTC_CR1.ALMF=1, enter the alarm clock interrupt processing; otherwise, enter the fixed-period interrupt processing.

18.2.7 1Hz output

RTC can choose to output three kinds of 1Hz clocks: general precision, higher precision and high precision. When the clock error compensation function is valid, it outputs a high-precision 1Hz clock; when using PCLK with different frequencies, it outputs a high-precision 1Hz clock. The system control register needs to be configured according to the PCLK frequency, where,

1Hz output of general precision is set as follows: (no clock compensation)

1. Set RTC_CR0.START=0, counting stops;
2. RTC output pin setting;
3. RTC_CR0.1HZOE=1, clock output permission;
4. Set RTC_CR0.START=1 to start counting;
5. Wait for more than 2 count cycles;
6. 1Hz output starts.

The higher precision 1Hz output setting is as follows: (low speed compensation)

1. Set RTC_CR0.START=0, counting stops;
2. RTC output pin setting;
3. RTC_CR0.1HZOE=1, clock output permission;
4. Clock error compensation register RTC_COMPEN.CR compensation number setting;
5. Clock error compensation register RTC_COMPEN.EN=1, error compensation is valid;
6. Set RTC_CR0.START=1 to start counting;
7. Wait for more than 2 count cycles;
8. 1Hz output starts.

When the high-precision 1Hz output is required, it is necessary to provide 4M, 6M, 8M, 12M, 16M, 20M, 24M, 32MHz high-speed PCLK clocks for the RTC on the basis of higher-precision output. The output settings are as follows:

1. Set RTC_CR0.START=0, counting stops;
2. RTC output pin setting;
3. RTC_CR0.1HZOE=1, clock output permission;
4. RTC_CR0.1HZSEL=1, choose to output high-precision 1Hz clock;
5. Configure high-speed clock compensation clock SYSCTRL1.RTC_FREQ_ADJUST
6. Clock error compensation register RTC_COMPEN.CR[8:0] compensation number setting;
7. Clock error compensation register RTC_COMPEN.EN=1, precision compensation is valid;
8. Set RTC_CR0.START=1 to start counting;
9. Wait for more than 2 count cycles;
10. 1Hz output starts.

18.2.8 Clock Error Compensation

Because there is an error in the external crystal oscillator, when it is necessary to obtain a high-precision counting result, the error needs to be compensated. There are two types of compensation methods: the first, error compensation based on its own clock; the second, error compensation based on a high-speed clock.

The principle and calculation of error compensation based on its own clock:

Since the counter uses a 32.768KHz clock to count, if it is necessary to compensate the accuracy per second, it can only be compensated according to the integer cycle of 32.768KHz, and the minimum unit of compensation per second is (1/32768)*10^6=30.5ppm, which cannot be satisfied high precision requirements.

Then, when realizing high-precision clock compensation under the counting clock of 32.768KHz, it is necessary to adjust the algorithm to expand the maximum compensation period by 32 times. Then, when the minimum unit that can only be compensated is 30.5ppm, the average compensation unit per second becomes 30.5ppm/32=0.96ppm. It meets the clock compensation requirement with high precision. And the compensation occurs in a relatively uniform range every 32 seconds. 5 decimal places is introduced in this register.

Example 1:

When the 1Hz clock is directly output in the default state, the compensation target value is calculated by measuring the accuracy of the clock.

Assuming that the actual measured value is 0.9999888Hz, then:

Actual oscillation frequency = 32768 X 0.9999888 ≈ 32767.63

Compensation target value = (actual oscillation frequency – target frequency)/target frequency X 10^6

= （32767.96 – 32768）/32768 X 10^6

= -11.29ppm

Basis

$$CR[8:0]=\left(\frac{\text{Compensation target value[ppm]}\times 2^{15}}{10^6}\right)_{\text{Take 2's complement}}+0001.00000\ B$$

If the compensation target value is -11.29ppm, calculate the corresponding register value as follows:

CR[8:0] = (-11.29 X 215/106) 2's complement + 0001.00000B

= (-0.37) 2's complement + 0001.00000B

= 1111.10101B + 0001.00000B

= 0000.10101B

Principle and calculation of error compensation based on high-speed 24MHz clock:

The calculation method of this method is the same as the error compensation based on its own clock. Due to the introduction of a 4M-32MHz high-speed clock, the 1/32768 second error that

originally needs to be accumulated within a maximum of 32 seconds can be dispersed to every 1 second, and a minimum 0.96ppm (23 24MHz clock cycles) compensation is performed for each 1 second to achieve an average High-precision 1Hz clock output per second.

18.3RTC Interrupt

The RTC supports two types of interrupts. Alarm clock interruption, regular period interruption. The alarm clock interrupt and the periodic interrupt share an interrupt signal.

18.3.1 RTC alarm interrupt

When RTC_CR1.ALMIE=1, if the current calendar time is equal to the second alarm clock register (RTC_ALMSEC), minute alarm clock register (RTC_ALMMIN), hour alarm clock register (RTC_ALMHOUR), and weekly alarm clock register (RTC_ALMWEEK), an alarm clock interrupt is triggered.

18.3.2 RTC cycle interrupt

ALMIE=1 of the control register 1 (RTC_CR1), after the selected period occurs, a fixed-period wake-up interrupt is triggered. Since the alarm clock and the fixed-period shared interrupt, it is distinguished by the flag register bit.

18.4RTC register description

Base address 0X40001400

Table 18-2 RTC register list

Register	Offset address	Description
RTC_CR0	0X000	Control register 0
RTC_CR1	0X004	Control register 1
RTC_SEC	0X008	Second Count Register
RTC_MIN	0X00C	Minute counter register
RTC_HOUR	0X010	Time counter register
RTC_WEEK	0X014	Week Count Register
RTC_DAY	0X018	Day Count Register
RTC_MON	0X01C	Month Count Register
RTC_YEAR	0X020	Year Count Register
RTC_ALMMIN	0X024	Minute alarm register
RTC_ALMHOUR	0X028	Time alarm register
RTC_ALMWEEK	0X02C	Weekly Alarm Register
RTC_COMPEN	0X030	Clock Error Compensation Register
RTC_ALMSEC	0X034	Seconds Alarm Register

18.4.1 Control Register 0 (RTC_CR0)

* Only power-on reset is valid for this register

Address offset: 0x000

Reset value 0x0000 0000

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

15	14	13:8	7	6	5	4	3	2:0
Res.	PRDSEL	PRDX	START	HZ1SEL	HZ1OE	Res.	AMPM	PRDS
	R/W	R/W	R/W	R/W	R/W		R/W	R/W

Bit	Symbol	Functional description
31:15	Reserved	-
14	PRDSEL	0: Use the periodic interrupt time interval set by PRDS 1: Use the periodic interrupt time interval set by PRDX
13:8	PRDX	Set the time interval for generating periodic interrupts, the settable range is from 0.5 seconds to 32 seconds, and the step is 0.5 seconds. 000000: 0.5 seconds 000001: 1 second ..... 111110: 31.5 seconds 111111: 32 seconds
7	START	0: stop RTC counter 1: Enable RTC counter
6	HZ1SEL	0: Normal precision 1Hz output 1: High precision 1Hz output
5	HZ1OE	0: disable 1Hz output 1: Enable 1Hz output
4	Reserved	-
3	AMPM	0: 12 hours 1: 24 hours
2:0	PRDS	Set the interval at which interrupts are generated: 000: no periodic interrupt 001: 0.5 seconds 010: 1 second 011: 1 minute 100: 1 hour 101: 1 day 11x: January _ Note: If you need to write and change the time interval of cycle interruption when START=1, the operation steps are as follows: step1, turn off the RTC interrupt in the NVIC; step2, change the time interval of periodic interruption; step3, clear the RTC interrupt flag; step4, enable RTC interrupt.

18.4.2 Control Register 1 (RTC_CR1)

* Only power-on reset is valid for this register

Address offset: 0x004

Reset value 0X00000000

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

15:11	10:8	7	6	5	4	3	2	1	0
Reserved	CKSEL	ALMEN	ALMIE	Res.	ALMF	PRDF	Res.	WAITF	WAIT
	R/W	R/W	R/W		R/W	R/W		R/W	R/W

Bit	Symbol	Functional description
31:11	Reserved	-
10:8	CKSEL	RTC clock selection 00x: XTL 32.768k 01x: RCL 32k 100: XTH/128 (choose this item when the crystal oscillator is 4M) 101: XTH/256 (choose this item when the crystal oscillator is 8M) 110: XTH/512 (choose this item when the crystal oscillator is 16M) 111: XTH/1024 (select this item when the crystal oscillator is 32M)
7	ALMEN	0: disable alarm clock 1: enable alarm clock Note: When ALMEN is enabled during START=1 calendar counting process and ALMIE=1 interrupt permission, please disable the system interrupt to prevent malfunction. After enabling, please clear the ALMF flag bit.
6	ALMIE	0: disable alarm interrupt 1: enable alarm clock interrupt
5	Reserved	-
4	ALMF	0: No alarm interrupt occurred 1: Alarm interrupt has occurred Note: This bit is only valid when ALMEN=1. When the alarm clock matches, a clock of 32.7689KHz is set to 1. Writing 0 clears the flag, writing 1 has no effect.
3	PRDF	0: Periodic interrupt does not occur 1: A cycle interrupt has occurred to 1 after a periodic interrupt occurs. Writing 0 clears this flag, writing 1 has no effect.
2	Reserved	-
1	WAITF	0: Not writing / reading status 1: Write / read status Note: WAIT bit setting is valid flag. "1" before writing / reading. During the counting process, the bit is cleared to "0" after the WAIT bit is cleared to "0" and waits for the writing to be completed.
0	WAIT	0: normal counting mode 1: write / read mode Note: Please set this bit to "1" when writing / reading. Since the counter is counting continuously, please complete the writing / reading operation within 1 second and clear this bit to "0".

18.4.3 Second Count Register (RTC_SEC)

Address offset: 0x008

Reset value: Indefinite

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

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

Bit	Symbol	Functional description
31:7	Reserved	-
6:4	SECH	Tens of seconds value
3:0	SECL	Second count unit value

Represents 0-59 seconds, counted in decimal. Please write the BCD code of decimal 0-59, when writing wrong value, the written value will be ignored.

18.4.4 Minute Count Register (RTC_MIN)

Address offset: 0x00C

Reset value: Indefinite

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

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

Bit	Symbol	Functional description
31:7	Reserved	-
6:4	MINH	Score tens digit value
3:0	MINL	Sub-count unit value

Represents 0-59 points, using decimal counting. Please write the BCD code of decimal 0-59, when writing wrong value, the written value will be ignored.

18.4.5 Hour count register (RTC_HOUR)

Address offset: 0x010

Reset value: Indefinite

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

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

Bit	Symbol	Functional description
31:6	Reserved	-
5:4	HOURH	Tens of digits
3:0	HOURL	Hour counter value

In 24-hour format, it means 0-23 hours. 12- hour format, b5=0 means AM, then 01:12 means morning; b5=1 means PM, then 21:32 means afternoon.

Please set the correct decimal 0:23 or 01:12, 21:32 BCD code according to the value of AMPM. Writing a value out of range will be ignored.

Please refer to the following table for the specific time:

24 hour clock	**AMPM=1**	**12 hour clock**	**AMPM=0**
Time	Register representation	Time	Register representation
00 HOUR	00H	AM 12 HOUR	12H
01 HOUR	01H	AM 01 HOUR	01H
02 HOUR	02H	AM 02 HOUR	02H
03 HOUR	03H	AM 03 HOUR	03H
04 HOUR	04H	AM 04 HOUR	04H
05 HOUR	05H	AM 05 HOUR	05H
06 HOUR	06H	AM 06 HOUR	06H
07 HOUR	07H	AM 07 HOUR	07H
08 HOUR	08H	AM 08 HOUR	08H
09 HOUR	09H	AM 09 HOUR	09H
10 HOUR	10H	AM 10 HOUR	10H
11 HOUR	11H	AM 11 HOUR	11H
12 HOUR	12H	PM 12 HOUR	32H
13 HOUR	13H	PM 01 HOUR	21H
14 HOUR	14H	PM 02 HOUR	22H
15 HOUR	15H	PM 03 HOUR	23H
16 HOUR	16H	PM 04 HOUR	24H
17 HOUR	17H	PM 05 HOUR	25H
18 HOUR	18H	PM 06 HOUR	26H
19 HOUR	19H	PM 07 HOUR	27H
20 HOUR	20H	PM 08 HOUR	28H
21 HOUR	21H	PM 09 HOUR	29H
22 HOUR	22H	PM 10 HOUR	30H
23 HOUR	23H	PM 11 HOUR	31H

18.4.6 Day Count Register (RTC_DAY)

Address offset: 0x018

Reset value: Indefinite

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

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

Bit	Symbol	Functional description
31:6	Reserved	-
5:4	DAYH	Dozens of days
3:0	DAYL	Day count unit value

Decimal means 1:31 day, automatically calculates leap year and month. The specific expression is as follows:

Month	**Day count display**
February (normal year)	01:28
February (leap year)	01:29
April, June, September, November	01:30
January, March, May, July, August, October, December	01:31

18.4.7 Week Count Register (RTC_WEEK)

Address offset: 0x014

Reset value: Indefinite

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

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

Bit	Symbol	Functional description
31:3	Reserved	-
2:0	WEEK	Week count value

Decimal 0:6 means Sunday:Saturday. Please write the correct decimal 0:6 BCD code, other values will be ignored. The corresponding relationship between the week count value is as follows:

Week	**Week count**
Sunday	00H
Monday	01H
Tuesday	02H
Wednesday	03H
Thursday	04H
Friday	05H
Saturday	06H

18.4.8 Month Count Register (RTC_MON)

Address offset: 0x01C

Reset value: Indefinite

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

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

Bit	Symbol	Functional description
31:5	Reserved	-
4:0	MON	Month count value

Decimal 1:12 represents 1:12 months. Please write in the correct decimal 1:12 BCD code, other values will be ignored.

18.4.9 Year Count Register (RTC_YEAR)

Address offset: 0x020

Reset value: Indefinite

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

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

Bit	Symbol	Functional description
31:8	Reserved	-
7:4	YEARH	Year tens digit value
3:0	YEARL	Year unit count value

Decimal 0:99 means 0:99 years. Count according to the monthly round. Automatically calculate leap years such as: 00, 04, 08, ..., 92, 96, etc. Please write correct decimal year count value, wrong value will be ignored.

18.4.10 Seconds Alarm Register (RTC_ALMSEC)

Address offset: 0x034

Reset value: Indefinite

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

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

Bit	Symbol	Functional description
31:7	Reserved	-
6:4	ALMSECH	Tens digit of second alarm match value
3:0	ALMSECL	Seconds alarm clock matching value unit digit

Please set the BCD code of decimal 0:59. Write other values, no alarm match will occur.

18.4.11 Minute Alarm Register (RTC_ALMMIN)

Address offset: 0x024

Reset value: Indefinite

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

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

Bit	Symbol	Functional description
31:7	Reserved	-
6:4	ALMMINH	Ten digits of minute alarm matching value
3:0	ALMMINL	Minute alarm clock matching value unit digit

Please set the BCD code of decimal 0:59. Write other values, no alarm match will occur.

18.4.12 Hour Alarm register (RTC_ALMHOUR)

Address offset: 0x028

Reset value: Indefinite

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

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

Bit	Symbol	Functional description
31:6	Reserved	-
5:4	ALMHOURH	Time alarm clock ten digit matching value
3:0	ALMHOURL	Hour alarm ones match value

Please set the correct alarm clock matching value according to the time system, otherwise the alarm clock matching will not happen.

18.4.13 Week Alarm Register (RTC_ALMWEEK)

Address offset: 0x02C

Reset value: Indefinite

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

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

Bit	Symbol	Functional description
31:7	Reserved	-
6:0	ALMWEEK	Week alarm match value. b0:b6 correspond to Sunday:Saturday respectively, when corresponding to “1”, it means that the alarm clock is valid on that day of the week. For example, b0=1, b5=1 means that the Sunday and Friday alarm settings are valid.

Please set the correct alarm clock matching value according to the time system, otherwise the alarm clock matching will not happen.

18.4.14 Clock Error Compensation Register (RTC_COMPEN)

Address offset: 0x030

* Only power-on reset is valid for this register, reset value: 0x00000020

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

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

Bit	Symbol	Functional description
31:16	Reserved	-
15	EN	Compensation enable 0: disable clock error compensation 1: Enable clock error compensation
14:9	Reserved	-
8:0	CR	Compensation value Through the compensation value setting, the accuracy compensation of +/-0.96ppm per second can be performed. The compensation value is 9-bit 2's complement code with a decimal point, and the last 5 bits are the decimal part. Compensable range 274.6ppm: 212.6ppm. Minimum differential error +/-0.48ppm. Minimum resolution 0.96ppm. Please refer to the following table for specific compensation accuracy:

Compensation value setting										Compensation
EN	CR[8:0]									
1	1	0	0	0	0	0	0	0	0	-274.6ppm
	1	0	0	0	0	0	0	0	1	-273.7ppm
	:	:	:	:	:	:	:	:	:	:
	0	0	0	0	1	1	1	1	1	-0.95ppm
	0	0	0	1	0	0	0	0	0	0ppm
	:	:	:	:	:	:	:	:	:	:
	0	1	1	1	1	1	1	1	0	+211.7ppm
	0	1	1	1	1	1	1	1	1	+212.6ppm
0	X	X	X	X	X	X	X	X	X	Uncompensated

Explanation and calculation of compensation principle:

Since the counter uses a 32.768KHz clock to count, if it is necessary to compensate the accuracy per second, it can only be compensated according to the integer cycle of 32.768KHz, and the minimum unit of compensation per second is (1/32768)*10^6=30.5ppm, which cannot be satisfied high precision requirements.

Then, when realizing high-precision clock compensation under the counting clock of 32.768KHz, it is necessary to adjust the algorithm to expand the maximum compensation period by 32 times. Then when the minimum unit that can only be compensated is 30.5ppm, the average compensation unit per second becomes 30.5ppm/32=0.96ppm. It meets the clock compensation requirement with high precision. And the compensation occurs in a relatively uniform range every 32 seconds. 5 decimal places is introduced in this register.

The setpoint is calculated as follows:

$$CR[8:0]=\left(\frac{\text{Compensation target value[ppm]}\times 2^{15}}{10^{6}}\right)_{\text{Take 2's complement}}+0001.00000B$$

If the compensation target value is +20.6ppm, calculate the corresponding register value as follows:

CR[8:0] = (20.3 X $2^{15}/10^{6}$) 2's complement + 0001.00000B

= (0.6651904) 2's complement + 0001.00000B

= 0000.10101B + 0001.00000B

= 0001.10101B

If the compensation target value is -20.6ppm, calculate the corresponding register value as follows:

CR[8:0] = (-20.3 X $2^{15}/10^{6}$) 2's complement + 0001.00000B

= (-0.6651904) 2's complement + 0001.00000B

= 1111.01011B + 0001.00000B

= 0000.01011B

19 Watchdog Timer (WDT)

19.1 Introduction to WDT

WDT can be used to detect and resolve failures caused by software errors. When the WDT counter reaches the set overflow time, it will trigger an interrupt or generate a system reset. WDT is driven by a dedicated 10KHz on-chip oscillator.

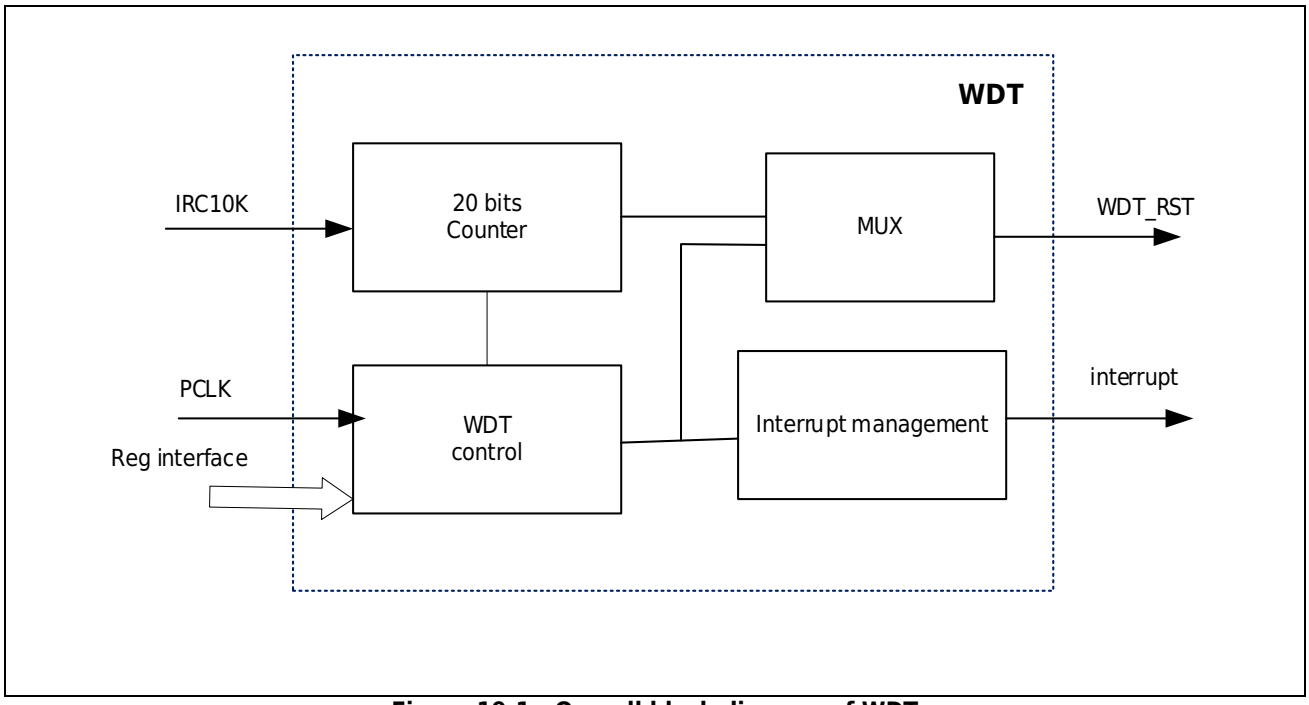


Figure 19-1 Overall block diagram of WDT

19.2 WDT Functional Description

- 20Bit free-running up counter, the overflow time can be configured from 1.6ms to 50s.
- Action after overflow can be configured as interrupt or reset.
- The WDT clock is provided by an independent RC oscillator, which can work in Sleep and DeepSleep modes.
- WDTCON register can only be modified when the WDT is not started to prevent unintentional modification of the WDT configuration after startup.

19.2.1 Interrupt generated after WDT overflow

In this mode, WDT will generate interrupt periodically according to the set time. WDT interrupt flag needs to be cleared in the interrupt service routine.

The configuration method is as follows:

Step1: Configure WDT_CON. WOV, select WDT timer overflow time.

Step2: Set WDT_CON. WINT_EN to 1, select WDT to generate an interrupt after overflow.

Step3: Enable the WDT interrupt in the NVIC interrupt vector table.

Step4: Write 0x1E and 0xE1 to the WDT_RST register in sequence to start the WDT timer.

Step5: Write 0x1E and 0xE1 to the WDT_RST register in the interrupt service routine to clear the interrupt flag.

19.2.2 Reset after WDT overflow

In this mode, the Reset signal will be generated after the WDT counter overflows, which will reset the MCU. User program needs to clear the WDT counter before WDT overflow, so as to avoid WDT reset.

The configuration method is as follows:

Step1: Configure WDT_CON. WOV, select WDT counter overflow time.

Step2: Set WDT_CON. WINT_EN to 0, select WDT to generate reset after overflow.

Step3: Write 0x1E and 0xE1 to the WDT_RST register in sequence to start the WDT timer.

Step4: Before the WDT overflows, write 0x1E and 0xE1 to the WDT_RST register to clear the WDT counter.

Note:

- Since the WDT oscillator is a low-precision RC oscillator, it is strongly recommended to clear the WDT before the WDT counter reaches half of the overflow value.

19.3WDT register description

Base address 0X40000F00

Table 19-1 WDT register list

Register	Offset address	Description
WDT_RST	0X080	WDT Clear Control Register
WDT_CON	0X084	WDT control register

19.3.1 WDT Clear Control Register (WDT_RST)

Offset address: 0x080

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved	WDTRST							
	WO							

Bit	Symbol	Description
31:8	Reserved	Reserved bit, read as 0
7:0	WDTRST	Watchdog start / clear control When the watchdog is not started, write 0x1E and 0xE1 to this register in turn to start the WDT timer. When the watchdog is enabled, write 0x1E and 0xE1 to this register in turn to clear the WDT timer and interrupt flag.

19.3.2 WDT_CON register

Offset address: 0x084

Reset value: 0x0000/ 000F

Note: This register can only be written when WDT is not running.

31:16	15:8	7	6	5	4	3	2	1	0
Reserved	WCNTL	WDTINT	Res.	WINT_EN	WDTR	WOV			
	RO	RO		RW	RO	RW			

Bit	Symbol	Description
31:16	Reserved	Reserved bit, read as 0
15:8	WCNTL	WDT counter low 8 bits
7	WDTINT	WDT interrupt flag 1: WDT interrupt has occurred, write 0x1E, 0xE1 to the WDT_RST register to clear the interrupt flag. 0: No WDT interrupt occurs.
5	WINT_EN	Action configuration after WDT overflow 1: Generate interrupt after WDT overflow. 0: A reset is generated after WDT overflows.
4	WDTR	WDT running flag 1: WDT is running 0: WDT stop
3:0	WOV[3:0]	WDT timeout time configuration

0000: 1.6ms	1000: 500ms
0001: 3.2ms	1001: 820ms
0010: 6.4ms	1010: 1.64s
0011: 13ms	1011: 3.28s
0100: 26ms	1100: 6.55s
0101: 51ms	1101: 13.1s
0110: 102ms	1110: 26.2s
0111: 205ms	1111: 52.4s

20 Pulse Counter (PCNT)

20.1 Introduction to Pulse Counters

The pulse counter (PCNT) module can count the input pulses and supports three pulse modes: single-channel pulse, dual-channel quadrature pulse, and dual-channel non-orthogonal pulse. Counts correctly in low-power modes without software involvement.

20.2 Pulse Counter Main Features

The pulse counter supports the following features:

- 16-bit counter supporting reload function
- Single channel pulse count
- Dual channel non-intersecting pulse counting
- Two-channel quadrature pulse counting without missing codes
- Up/down counting overflow interrupt
- Pulse timeout interrupt
- 4 kinds of decoding error interrupt, non-intermittent pulse mode
- 1 direction change interrupt, quadrature pulse mode
- Multi-stage pulse width filtering
- Configurable input pulse polarity
- Support low-power mode counting
- Support waking up MCU in low power mode
- Support any pulse edge spacing not less than 1 count clock cycle
- Support IO input or VC output as pulse input signal
- Support dual-channel pulse signal shaping and output PCNT_S0FO/PCNT_S1FO through IO observation
- With automatic timing wake-up function in low power consumption mode, the maximum timing is 1024 seconds

20.3Pulse Counter Functional Description

20.3.1 Overall block diagram

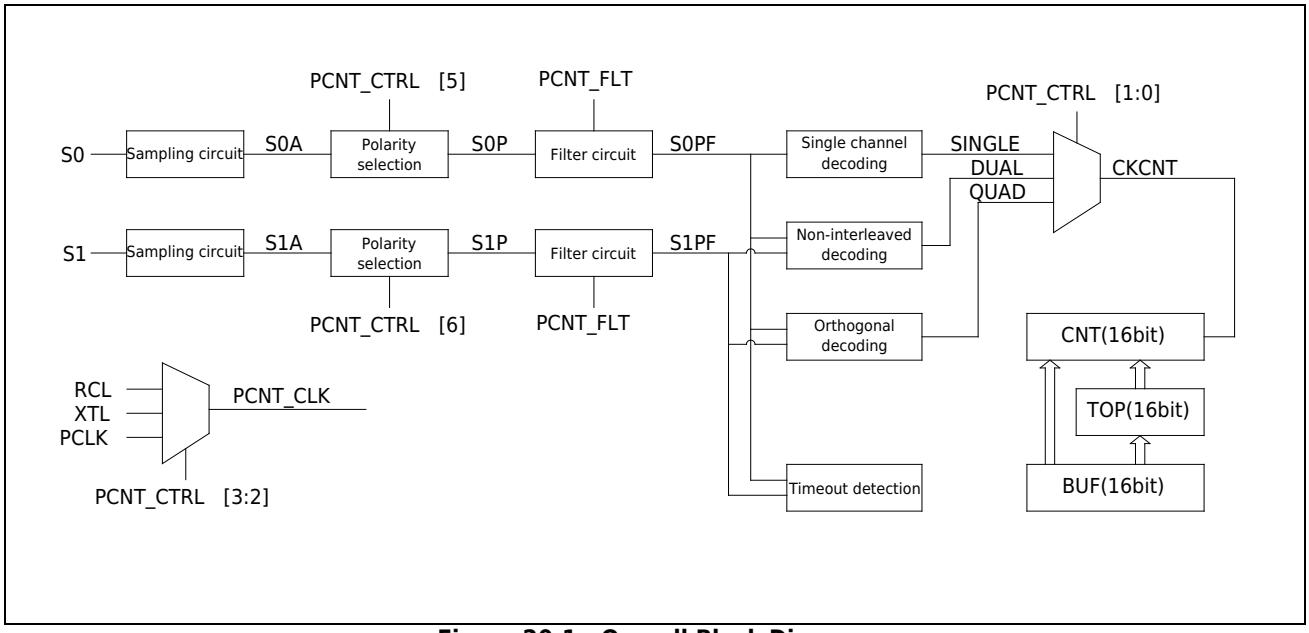


Figure 20-1 Overall Block Diagram

20.3.2 Signal Description

The following takes the input signal S0 as an example to illustrate the processing flow of the signal in the counter module.

- The S0 signal outputs the S0A signal after the sampling module, and the sampling clock is PCNT_CLK.
- The S0A signal outputs the S0P signal after passing through the polarity selection module.
- The S0P signal outputs the S0PF signal after passing through the pulse width filter module, and the clock of the filter module is PCNT_CLK after frequency division.
- The S0PF signal is decoded by the single-channel decoding module, the dual-channel non-intersection decoding module, and the dual-channel orthogonal decoding module to output corresponding SIGNAL, DUAL, and QUAD signals.
- SIGNAL, DUAL, and QUAD output CKCNT signals after passing through the multiplexer to drive the CNT module to count.

20.3.3 Counting mode

This module supports 3 counting modes: single-channel pulse, dual-channel quadrature pulse, and dual-channel non-orthogonal pulse.

20.3.3.1 Single channel pulse counting mode (Single Mode)

When configuring PCNT_CTRL.Mode as 0x00 or 0x01, the pulse counter works in single-channel pulse counting mode. In this mode, the decoding module only counts the S0 pulse signal. When the

PCNT_CLK clock is sampled to the falling edge of the S0PF signal, the counter performs an increment or decrement operation according to the configuration of PCNT_CTRL.DIR.

The count range of the counter is from 0x00 to the upper count threshold (PCNT_TOP).

In the up-counting state, when the count value is equal to PCNT_TOP and the falling edge of the S0PF signal is sampled, the counter overflows, the count value returns to 0 and PCNT_IFR.OF is set, and the interrupt flag needs to be cleared by software. The figure below is the timing diagram of PCNT counting up, in which the value of PCNT_TOP is 99.

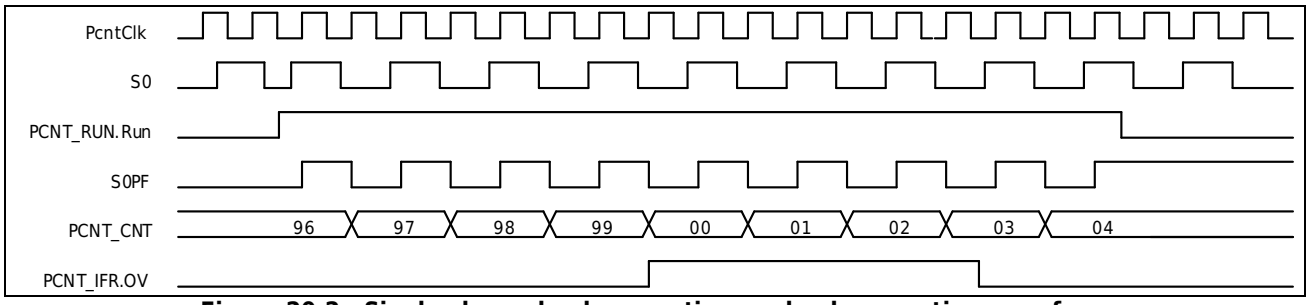


Figure 20-2 Single-channel pulse counting mode plus counting waveform

In the down-counting state, when the count value is equal to 0x00 and the falling edge of the S0PF signal is sampled, the counter underflows, the count value returns to PCNT_TOP and PCNT_IFR.UF is set, and the interrupt flag needs to be cleared by software. The following figure is the timing diagram of PCNT counting down, in which the value of PCNT_TOP is 99.

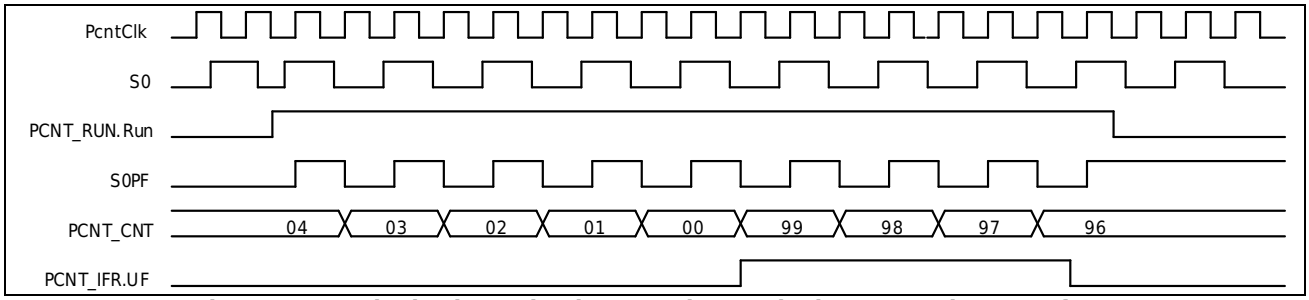
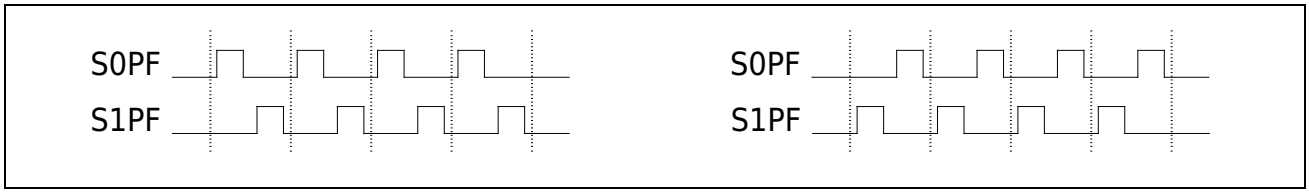


Figure 20-3 Single channel pulse counting mode down counting waveform

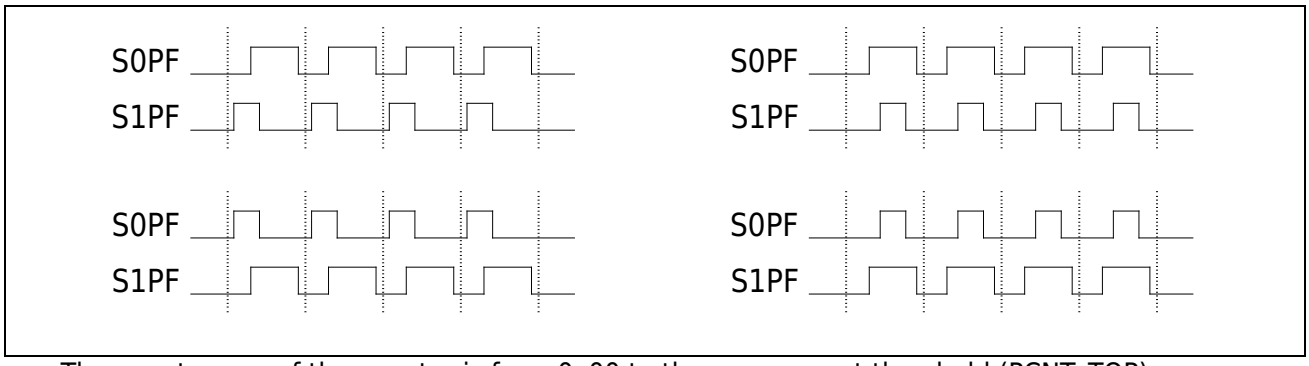
20.3.3.2 Dual-channel non-communicating pulse counting mode (Dual Mode)

PCNT_CTRL.Mode is configured as 0x02, the pulse counter works in the dual-channel non-communicating pulse counting mode. In this mode, the decoding module decodes and counts S0 and S1 pulse signals. When two positive pulses appear in S0PF and S1PF in turn, the counter performs an increment or decrement operation according to the configuration of PCNT_CTRL.DIR.

The two waveforms that the decoding module can count correctly are shown below. S0PF and S1PF are not high at the same time.



Several waveforms that the decoding module cannot count correctly are shown below. S0PF and S1PF are high at the same time.



The count range of the counter is from 0x00 to the upper count threshold (PCNT_TOP).

In the up-counting state, when the count value is equal to PCNT_TOP and two positive pulses appear in S0PF and S1PF in sequence, the counter overflows, the count value returns to 0 and PCNT_IFR.OF is set, and the interrupt flag needs to be cleared by software. The figure below is the timing diagram of PCNT counting up, in which the value of PCNT_TOP is 99.

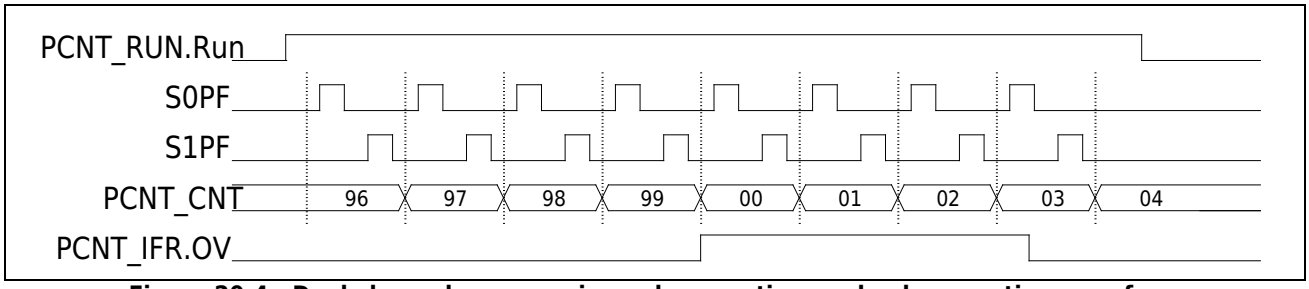


Figure 20-4 Dual-channel non-crossing pulse counting mode plus counting waveform

In the down-counting state, when the count value is equal to 0x00 and two positive pulses appear in S0PF and S1PF in sequence, the counter overflows, the count value returns to PCNT_TOP and PCNT_IFR.UF is set, and the interrupt flag needs to be cleared by software. The following figure is the timing diagram of PCNT counting down, in which the value of PCNT_TOP is 99.

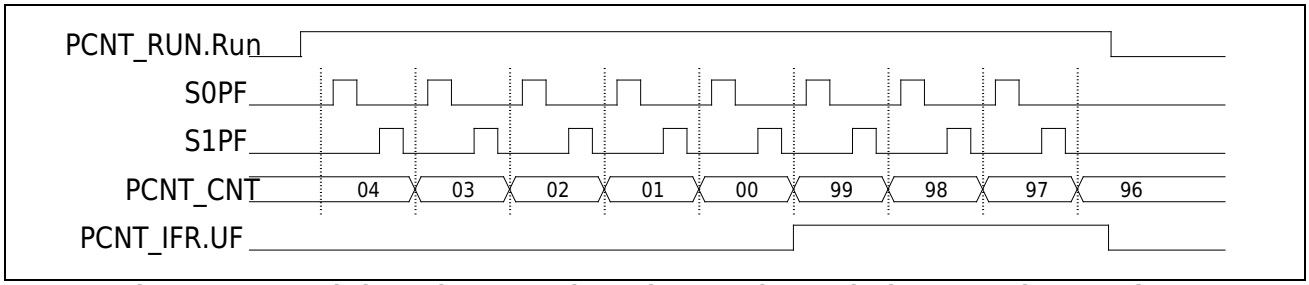
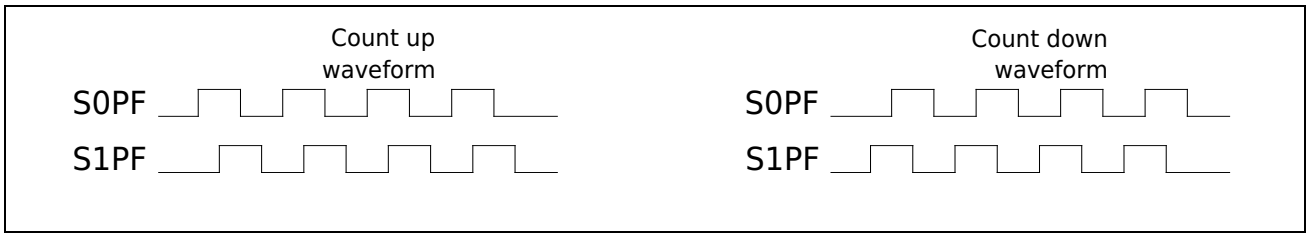


Figure 20-5 Dual-channel non-crossing pulse counting mode down counting waveform

20.3.3.3 Dual channel quadrature pulse counting mode (Quad Mode)

PCNT_CTRL.Mode is configured as 0x03, the pulse counter works in dual-channel quadrature pulse counting mode. In this mode, the decoding module needs two pulse signals of S0 and S1 for decoding and counting. The counter automatically judges the counting direction according to the phase relationship of S0PF and S1PF pulses, and the physical equipment can count correctly in any forward or reverse rotation. When S0PF and S1PF complete a cycle change, the counter performs an increment or decrement operation according to the counting direction.

The waveform of the decoding module to judge the counting direction is as follows:



The count range of the counter is from 0x00 to the upper count threshold (PCNT_TOP).

In the counting state, when the count value is equal to PCNT_TOP and S0PF and S1PF complete a cycle change, the counter overflows, the count value returns to 0 and PCNT_IFR.OF is set, and the interrupt flag needs to be cleared by software. The figure below is the timing diagram of PCNT counting up, in which the value of PCNT_TOP is 99.

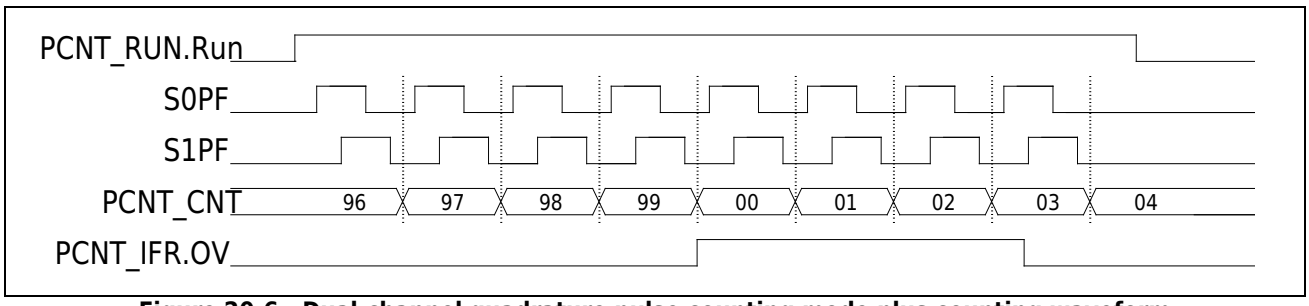


Figure 20-6 Dual-channel quadrature pulse counting mode plus counting waveform

In the down-counting state, when the count value is equal to 0x00 and S0PF and S1PF complete a cycle change, the counter overflows, the count value returns to PCNT_TOP and PCNT_IFR.UF is set, and the interrupt flag needs to be cleared by software. The following figure is the timing diagram of PCNT counting down, in which the value of PCNT_TOP is 99.

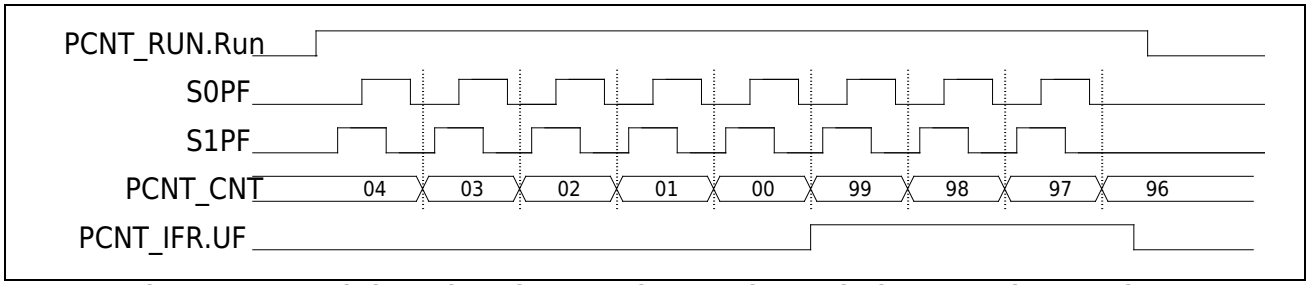


Figure 20-7 Dual-channel quadrature pulse counting mode down counting waveform

20.3.4 Pulse Width Filtering

The pulse width filter uses the frequency division of the counting clock as the filter clock, and counts with this to achieve the purpose of pulse debounce. Among them, the frequency division factor of the filter clock is optional from 1 to 8096 times the configuration register (FLT.ClkDiv), and the filter clock whose level exceeds (FLT.DebTop) (1~7) is regarded as non-jittering and can pass the filter.

This filtering function can be turned on or off by configuration register (FLT.EN).

At the same time, the filter counter starts only after starting PCNT (Run=1).

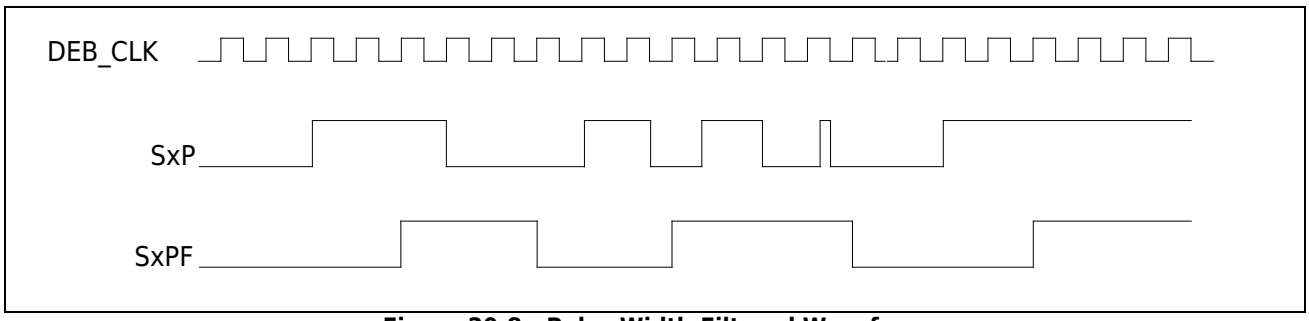


Figure 20-8 Pulse Width Filtered Waveform

The figure shows the filtered waveform when FLT.DebTop = 2.

$$f_{DEB_CLK}=\frac{f_{PCNT_CLK}}{(FLT.ClkDiv+1)}$$

20.3.5 Time-out

Turning on the timeout function (TOCR.EN) will use the filter clock (see 20.3.4 脉冲宽度滤波) to count up the pulse high level time from 0, sampling to a low level will make the counter return to 0, if a high level count value reaches (TOCR .TH) will generate interrupt flag bit IF.TO.

The timeout function uses the filter clock, and the frequency division factor (FLT.ClkDiv) needs to be configured, but it does not depend on whether the filter enable (FLT.EN) is enabled.

the timeout counter starts only after starting PCNT (Run=1).

The overflow time calculation formula is as follows:

$$T_{TO}=\frac{(FLT.ClkDiv+1)\times TOCR.TH}{f_{PCNT_CLK}}$$

20.3.6 Pulse input selection

Compared with HC32L130 / HC32L136 series, only the input of IO can be selected as the pulse input of PCNT. In this series, the output of VC is added as the pulse input of PCNT. The selection is made through the GPIO_PCNT register. For details, refer to the description of the port auxiliary register in the GPIO chapter..

20.3.7 Debug observation output

In order to confirm that the signals entering the pulse decoding module meet expectations, you can select three groups of shaped pulses through the register to output through IO as PCNT_S0FO/PCNT_S1FO observation signals, respectively:

- S0A/S1A
- S0P/S1P
- S0PF/S1PF

By observing the output pulse signal of PCNT_S0FO/PCNT_S1FO, the polarity selection and the configuration of the pulse width filter can be adjusted more intuitively to avoid that the input signal of the decoding circuit does not meet expectations.

20.3.8 Automatic wake-up timer in low-power mode

Through proper configuration, PCNT can realize automatic wake-up in low power consumption mode, and its longest timing period is 1024 seconds.

The specific operation process is as follows:

Step1: Set the GPIO_PCNT register to 0, select IO as the input source of PCNT.
Step2: Make sure that the port function selection register (Pxx_SEL) does not configure any port as PCNT_S0.
Step3: Set CTRL.Mode to 0, select single channel mode.
Step4: Set CTRL. S0P to 1, and select polarity inversion for the S0 input channel.
Step5: Configure CTRL. ClkSel, select XTL or RCL as the counter clock.
Step6: Configure FLT. ClkDiv to select the clock frequency division factor of the filter.
Step7: Configure TOCR.TH to select the timing period of the timeout timer.
Step8: Set TOCR.EN to 1 to enable the timeout timer.
Step9: Enable PCNT interrupt vector table.
Step10: Write 0 to ICR to clear all interrupt flags of PCNT
Step11: Set IEN.TO to 1 to enable timeout interrupt.
Step12: Set RUN to 1 and start PCNT.
Step13: Enter low power consumption mode, wait for PCNT timeout interrupt to wake up MCU.
Step14: In the interrupt service routine, perform the following steps in sequence:
a. Set RUN to 0
b. Write 0 to ICR.TO to clear the timeout interrupt flag
c. Set PCNT_IEN.TO to 0
d. Execute the functions required by the user
e. Query until PCNT_RUN is 0
f. Query until PCNT_IFR.TO is 0
g. Exit the interrupt service routine

Step15: To start the timer again, repeat steps 11~14.

20.4PCNT register description

Register list

Base address: 0x40005400

Table 20-1 Register List

Offset	Register name	Access	Synchronous / asynchronous	Register description
0x00	PCNT_RUN	RW	Asynchronous	PCNT start register
0x04	PCNT_CTRL	RW	Asynchronous long signal	PCNT control register
0x08	PCNT_FLT	RW	Asynchronous long signal	PCNT filter control register
0x0c	PCNT_TOCR	RW	Asynchronous long signal	PCNT Timeout Control Register
0x10	PCNT_CMD	WO1	Asynchronous	PCNT command register
0x14	PCNT_SR1	RO	Synchronize	PCNT Status Register 1
0x18	PCNT_CNT	RO	Synchronize	PCNT count register
0x1c	PCNT_TOP	RO	Synchronize	PCNT count overflow register
0x20	PCNT_BUF	RW	Asynchronous	PCNT count overflow buffer register
0x24	PCNT_IFR	RO	Synchronize	PCNT Interrupt Flag Register
0x28	PCNT_ICR	WO0	Asynchronous	PCNT Interrupt Clear Register
0x2c	PCNT_IEN	RW	Asynchronous long signal	PCNT Interrupt Enable Register
0x30	PCNT_SR2	RO	Asynchronous	PCNT Status Register 2
0x34	PCNT_DBG	RW	Asynchronous long signal	PCNT PCNT_S0FO/PCNT_S1FO output selection register

20.4.1 PCNT start register (PCNT_RUN)

Address offset: 0x00

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															Run
															rw

Reset value: 0x0000 0000

Bit	Marking	Functional description
31:1	Reserved	
0	Run	PCNT start / stop control. 0: Stop 1: Start up

Note:

- The current module regards the configuration register of the pclk clock domain as a long signal, and does not synchronize to the pcnt_clk clock domain. Therefore, when Run=1 starts PCNT, the configuration register is not allowed to be modified to avoid unknown errors. All configuration register modifications must be performed in the PCNT stopped state (run=0).
- After writing a value to the Run bit, a synchronization process is required to start / stop PCNT counting. Reading the Run bit before the synchronization is completed will return the Run status value before writing. Therefore, if you want to start PCNT after stopping PCNT, you need to write 0 to the Run bit, and then continue other operations after checking that the Run bit = 0 (including writing 1 to the Run bit to restart PCNT).

20.4.2 PCNT Control Register (PCNT_CTRL)

Address offset: 0x04

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									S1P	S0P	DIR	ClkSel		Mode	
									rw	rw	rw	rw		rw	

Bit	Marking	Functional description
31:7	Reserved	
6	S1P	S1 channel input polarity selection 0: Not negated 1: Negate
5	S0P	S0 channel input polarity selection 0: Not negated 1: Negate
4	DIR	Counting direction selection 0: Count up 1: Count down Note: Only valid for single-channel mode and dual-channel non-communication mode.
3:2	ClkSel	PCNT_CLK count clock selection 0：PCLK 1：PCLK 2：XTL 3：RCL
1:0	Mode	Pulse counting mode selection 0: Single channel pulse counting mode 1: Single channel pulse counting mode 2: Dual-channel non-communicating pulse counting mode 3: Dual channel quadrature pulse counting mode

20.4.3 PCNT filter control register (PCNT_FLT)

Address offset: 0x08

Reset value: 0x00002000

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

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

Bit	Marking	Functional description
31:17	Reserved	
16	EN	Pulse width filter enable control. 0: Not enabled 1: Enable
15:13	DebTop	Counter threshold Use the filter clock to continuously sample the input pulse. When the continuous level sampling count value reaches the threshold, it is regarded as a normal pulse and can pass through the filter. 0: Illegal value
12:0	ClkDiv	Filter clock frequency division factor Coefficient = ClkDiv + 1

20.4.4 PCNT Timeout Control Register (PCNT_TOCR)

Address offset: 0x0c

Reset value: 0x00000FFF

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

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

Bit	Marking	Functional description
31:17	Reserved	
16	EN	Timeout function enable control. 0: Not enabled 1: Enable
15:12	Reserved	
11:0	TH	Timeout threshold. When the pulse high level is filtered by the clock and the continuous sampling count reaches the threshold, an overtime interrupt flag is generated, and the counter restarts counting.

20.4.5 PCNT Command Register (PCNT_CMD)

Address offset: 0x10

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													B2C	B2T	T2C
													wo1	wo1	wo1

Bit	Marking	Functional description
31:3	Reserved	
2	B2C	Write 1 to immediately synchronize the value in BUF to CNT. The corresponding bit of PCNT_SR2 is 1 during the synchronization process. At this time, the BUF should not be written, and the CNT should not be read.
1	B2T	Writing 1 immediately synchronizes the value in BUF to TOP. The corresponding bit of PCNT_SR2 is 1 during the synchronization process. At this time, BUF should not be written, and TOP should not be read.
0	T2C	Write 1 to immediately synchronize the value in TOP to CNT. During the synchronization process, the corresponding bit of PCNT_SR2 is 1. At this time, 1 should not be written to B2T, and CNT should not be read.

20.4.6 PCNT status register 1 (PCNT_SR1)

Address offset: 0x14

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															DIR
															ro

Bit	Marking	Functional description
31:1	Reserved	
0	DIR	Dual-channel quadrature pulse counting direction indication 0: Count up 1: Count down

20.4.7 PCNT count register (PCNT_CNT)

Address offset: 0x18

Reset value: 0x0000 0000

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

Bit	Marking	Functional description
31:16	Reserved	
15:0	CNT	Counter count value.

20.4.8 PCNT count overflow register (PCNT_TOP)

Address offset: 0x1c

Reset value: 0x0000 00FF

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
TOP															
ro															

Bit	Marking	Functional description
31:16	Reserved	
15:0	TOP	Counter overflow threshold. The counter underflows the reload value.

20.4.9 PCNT count overflow buffer register (PCNT_BUF)

Address offset: 0x20

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
BUF															
rw															

Reset value: 0x0000 00FF

Bit	Marking	Functional description
31:16	Reserved	
15:0	BUF	CNT register, TOP register cache.

20.4.10 PCNT Interrupt Flag Register (PCNT_IFR)

Address offset: 0x24

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								S1E	S0E	BB	FE	DIR	TO	OV	UF
								ro	ro	ro	ro	ro	ro	ro	ro

Bit	Marking	Functional description
31:8	Reserved	
7	S1E	Pulse decode error interrupt flag. In a complete sampling period of non-interlaced coding, the S1 channel does not change. It is only valid in dual-channel non-interlaced encoding mode.
6	S0E	Pulse decode error interrupt flag. In a complete sampling period of non-interlaced coding, the S0 channel does not change. It is only valid in dual-channel non-interlaced encoding mode.
5	BB	Pulse decode error interrupt flag. Dual-channel pulses are sampled back-to-back as high, with no low state in between. It is only valid in dual-channel non-interlaced encoding mode.
4	FE	Pulse decode error interrupt flag. In a complete sampling period of non-intersection coding, what is sampled is not a correct non-intersection coding frame. It is only valid in dual-channel non-interlaced encoding mode.
3	DIR	Quadrature pulse direction change interrupt flag. Quadrature pulses are interpreted by the decoder as a change in count direction. Valid only for dual-channel quadrature pulses.
2	TO	Timeout interrupt flag. In all 3 modes.
1	OV	Overflow interrupt flag. In all 3 modes.
0	UF	Underflow interrupt flag. In all 3 modes.

20.4.11 PCNT Interrupt Clear Register (PCNT_ICR)

Address offset: 0x28

Reset value: 0x000000FF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								S1E	S0E	BB	FE	DIR	TO	OV	UF
								wo0	wo0	wo0	wo0	wo0	wo0	wo0	wo0

Writing 0 to each bit can clear the corresponding interrupt flags of PCNT_IFR in 20.4.10.

See 20.4.10 for detailed explanation of each bit.

20.4.12 PCNT Interrupt Enable Register (PCNT_IEN)

Address offset: 0x2c

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								S1E	S0E	BB	FE	DIR	TO	OV	UF
								rw	rw	rw	rw	rw	rw	rw	rw

Each bit corresponds to the output enable bit of each corresponding interrupt in PCNT_IFR in 20.4.10.

1: Enable. 0: Disabled.

See 20.4.10 for detailed explanation of each bit.

20.4.13 PCNT Synchronization Status Register (PCNT_SR2)

Address offset: 0x30

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													B2C	B2T	T2C
													ro	ro	ro

Bits	Marking	Functional description
31:3	Reserved	
2	B2C	The synchronization status bit when synchronizing the value in BUF to CNT. 1: Synchronization is in progress, CNT register cannot be read 0: Synchronization is complete
1	B2T	The synchronization status bit when synchronizing the value in BUF to TOP. 1: Synchronization is in progress, the TOP register cannot be read 0: Synchronization is complete
0	T2C	Synchronization status bit when synchronizing the value in TOP to CTN. 1: Synchronization is in progress, CNT register cannot be read 0: Synchronization is complete

20.4.14 PCNT debug observation output selection register (PCNT_DBG)

Address offset: 0x34

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														DBG	
														rw	

Bits	Marking	Functional description
31:2	Reserved	
1:0	DBG	Select the source of pulse PCNT_S0FO/PCNT_S1FO observation output. 0: fixed to 0 1: Output after pulse synchronization (S0A/S1A) 2: Output after pulse polarity selection (S0P/S1P) 3: Output after pulse filtering (S0PF/S1PF)

21 General Synchronous Asynchronous Transceiver (UART)

21.1 Introduction

The Universal Synchronous Asynchronous Receiver (UART) can flexibly exchange full-duplex data with external devices, and it supports synchronous one-way communication, single-wire half-duplex communication, and multi-processor communication. It is often used in short-distance, low-speed serial communication. The UART provides a variety of baud rates through a programmable baud rate generator. UART supports multiple working modes.

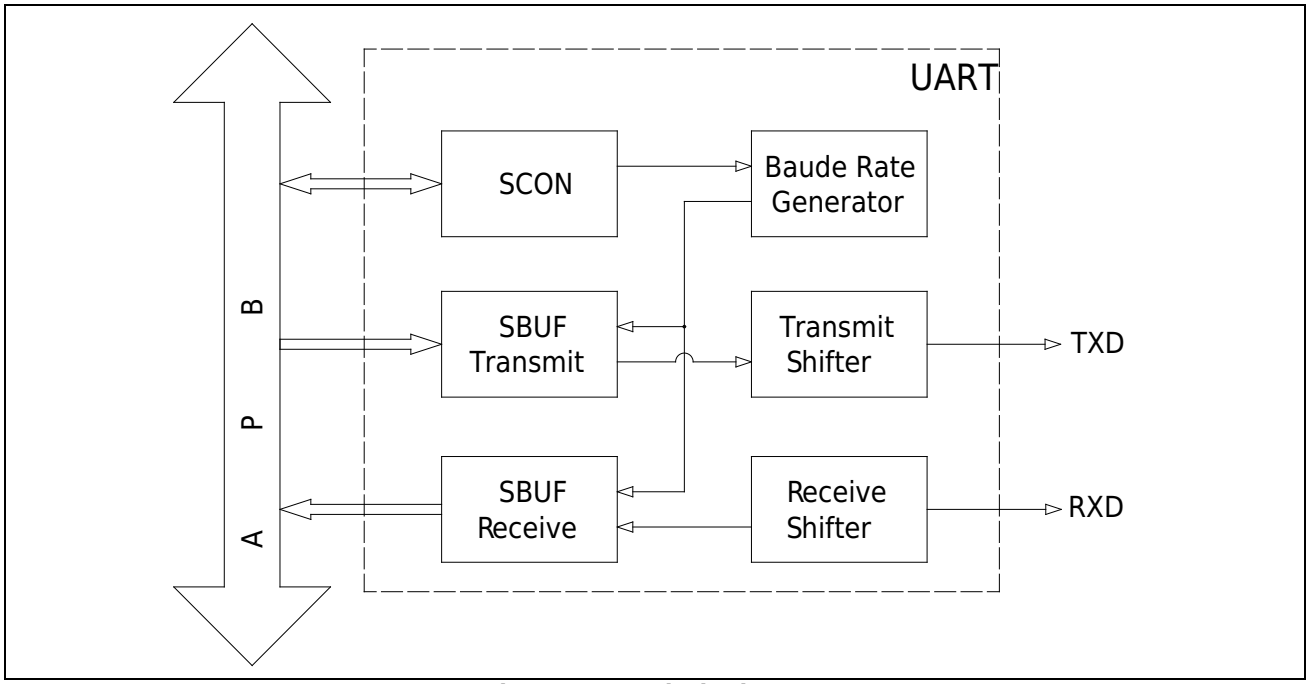


Figure 21-1 Block Diagram

21.2 Main characteristics

General purpose UART module supports the following basic functions:

- Full-duplex transmission, half-duplex transmission, single-wire half-duplex transmission
- Programmable serial communication function
  - Two character lengths: 8 bits, 9 bits
  - Three checkout methods: no checkout, odd checkout, even checkout
  - Three stop lengths: 1 bit, 2 bits, 1.5 bits
- 16- bit baud rate generator
- Multi-machine communication
- Hardware address recognition
- Hardware flow control
- DMAC hardware transmission handshake

21.3Functional description

21.3.1 Operating mode

UART supports multiple working modes: synchronous half-duplex mode, asynchronous full-duplex mode, and single-wire half-duplex mode. Through the combination of UARTx_SCON.SM and UARTx_SCON.HDSEL, various working modes can be configured.

21.3.1.1 Mode0~Mode3 function comparison

Configure UARTx_SCON.SM to select different transmission modes: Mode0~Mode3. The main function comparison of these four working modes is shown in the table below:

Table 21-1 Mode0/1/2/3 Data Structure

Operating mode		Transmission bit width	Data composition	Baud rate
Mode0	Synchronous mode Half duplex	8bit	Data(8bit)	$BaudRate=\frac{f_{PCLK}}{12}$
Mode1	Asynchronous mode Full duplex	10~11bit	Start (1bit) + Data(8bit) + Stop(1~2bit)	$BaudRate=\frac{f_{PCLK}}{OVER*SCNT}$
Mode2	Asynchronous mode Full duplex	11~12bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop(1~2bit)	$BaudRate=\frac{f_{PCLK}}{OVER}$
Mode3	Asynchronous mode Full duplex	11~12bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop(1~2bit)	$BaudRate=\frac{f_{PCLK}}{OVER*SCNT}$

Note:

- Mode0 can only be used as a host to send UART synchronous shift clock, and cannot be used as a slave to receive UART synchronous shift clock input from the outside.
- f_{PCLK}Represents the frequency of the current PCLK.
- The definition of OVER is detailed in UARTx_SCON.
- UARTx_SCNT for the definition of SCNT.
- The B8 data bit is special and has different meanings in different applications, please refer to the following table:

Table 21-2 B8 Data Meaning

Application Scenario	UARTx_SCON. ADRDET	UARTx_SCON. B8CONT[1:0]	B8 Data Meaning
Parity	--	01/10	When receiving, B8 is the parity bit of the received 8-Bit data; When sending, B8 is the parity bit of the 8-Bit data to be sent;
Multi-machine communication	1	--	B8=1, it means that it is currently an address frame; B8=0, it means the current data frame;
Other	0	00/11	8th bit of receive / send number

Note:

– When the multi-machine communication mode is turned on, the parity check of the received data is automatically closed; the parity check of the sent data is still controlled by B8CONT.

21.3.1.2 Mode0 data sending and receiving instructions

When sending data, clear the UARTx_SCON.REN bit and write the data into the UARTx_SBUF register. At this time, the sending data is output from RXD (low bit first, high bit later), and the synchronous shift clock is output from TXD.

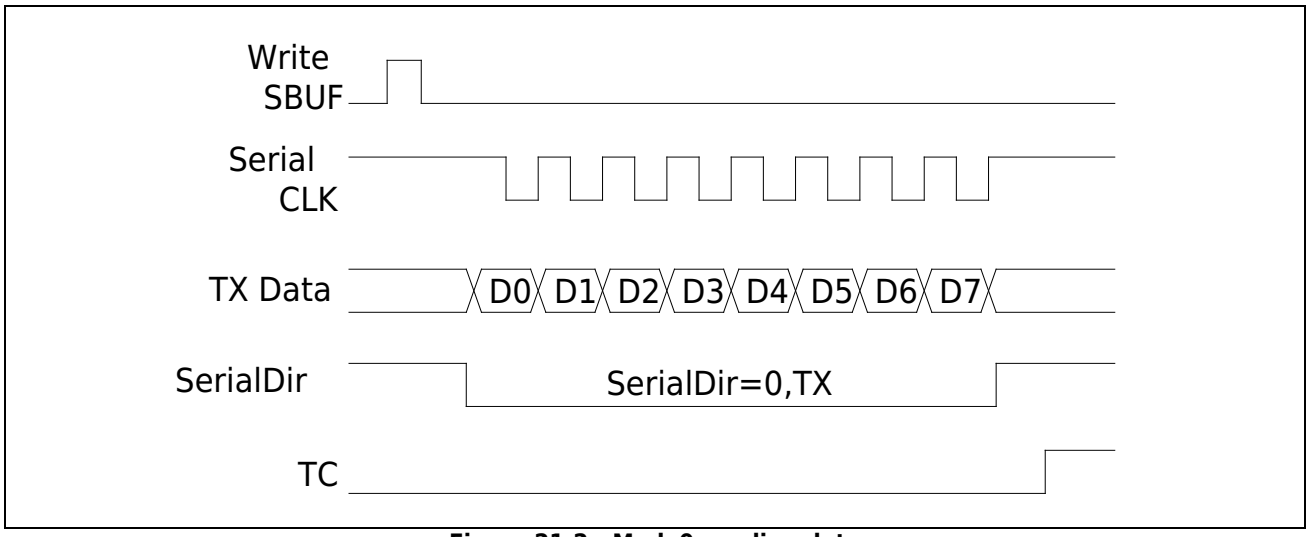


Figure 21-2 Mode0 sending data

When receiving data, set the UARTx_SCON.REN bit and clear the UARTx_ISR.RC bit. When reception is complete, data can be read from the UARTx_SBUF register. At this time, the received data is input from RXD (low bit first, high bit later), and the synchronous shift clock is output from TXD.

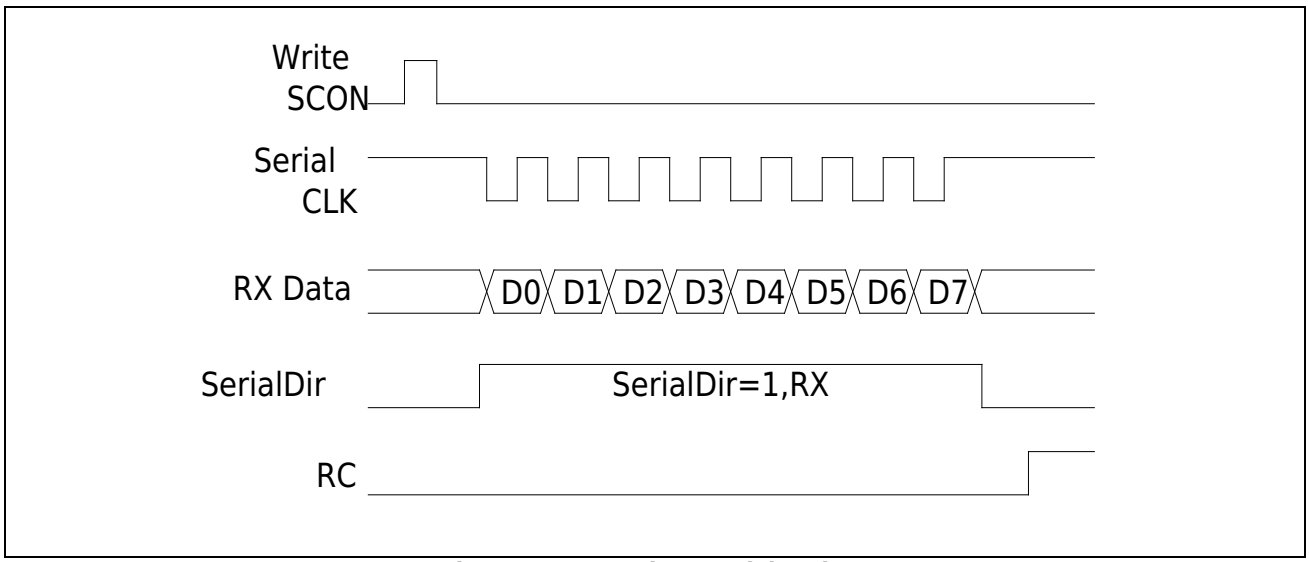


Figure 21-3 Mode0 receiving data

21.3.1.3 Mode1 data sending and receiving instructions

When sending data, it has nothing to do with the value of UARTx_SCON.REN, write the sent data into the UARTx_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

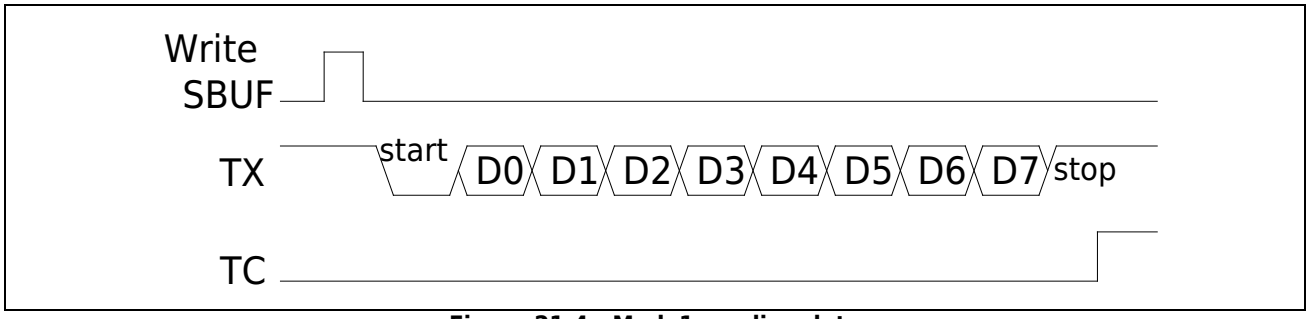


Figure 21-4 Mode1 sending data

When receiving data, set the UARTx_SCON.REN bit to 1 and clear the UARTx_ISR.RC bit to 0. Start to receive the data on RXD (low bit first, high bit later), when the reception is completed, it can be read from the UARTx_SBUF register.

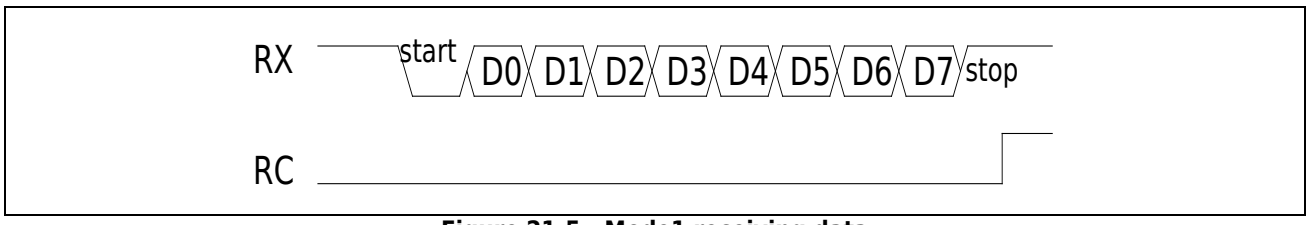


Figure 21-5 Mode1 receiving data

21.3.1.4 Mode2 data sending and receiving instructions

When sending data, it has nothing to do with the value of UARTx_SCON.REN, write the sent data into the UARTx_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

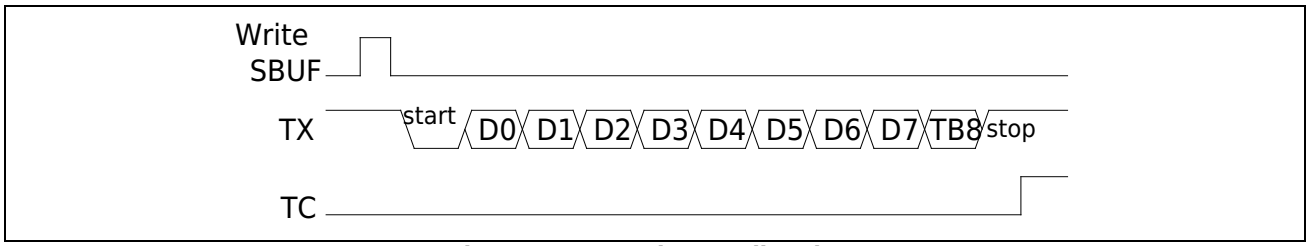


Figure 21-6 Mode2 sending data

When receiving data, you need to set the UARTx_SCON.REN bit to 1, and clear the UARTx_ISR.RC bit to 0. Start to receive the data on RXD (low bit first, high bit later), when the reception is completed, it can be read from the UARTx_SBUF register.

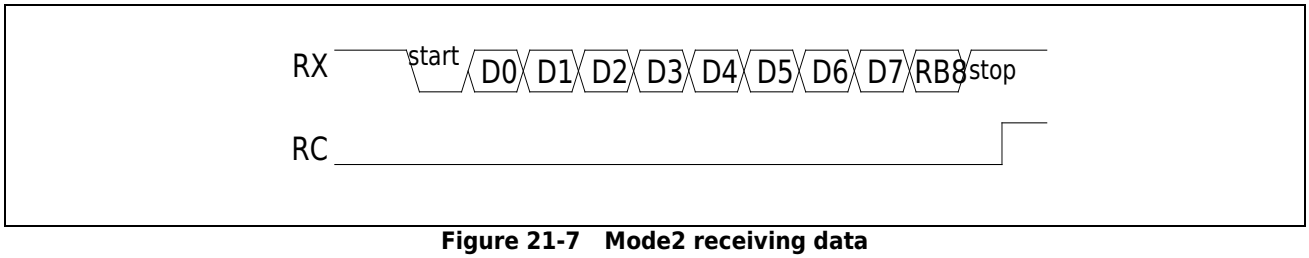


Figure 21-7 Mode2 receiving data

21.3.1.5 Mode3 data sending and receiving instructions

When sending data, it has nothing to do with the value of UARTx_SCON.REN, write the sent data into the UARTx_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

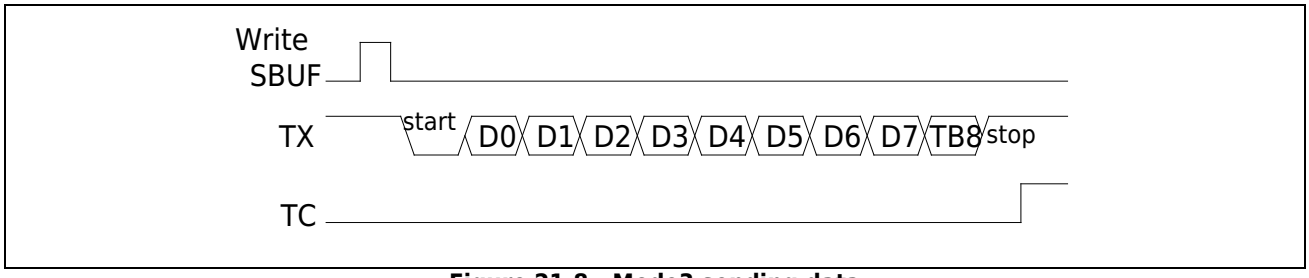


Figure 21-8 Mode3 sending data

When receiving data, set the UARTx_SCON.REN bit to 1 and clear the UARTx_ISR.RC bit to 0. Start to receive the data on RXD (low bit first, high bit later), when the reception is completed, it can be read from the UARTx_SBUF register.

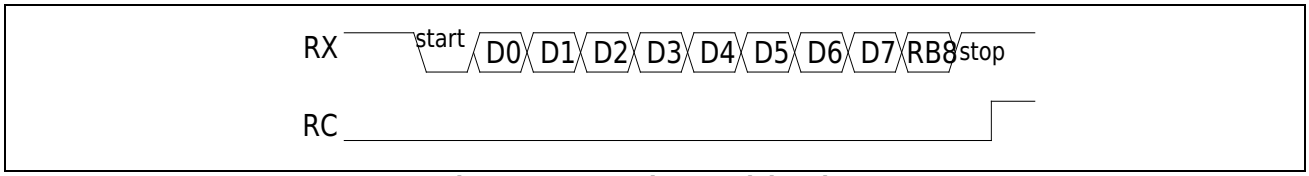


Figure 21-9 Mode3 receiving data

21.3.1.6 Description of single-wire half-duplex data transmission and reception

When UARTx_SCON.HDSEL=1'b1, it can enter the single-wire half-duplex transmission mode:

- The TX and RX signal lines are connected inside the module.
- The RX signal is no longer used, and the received and sent data are all completed through the TX signal. TX signal is done by hardware logic without software control.
- When the sending buffer is empty, the TX signal is always input (receiving state), once a data is filled into the sending buffer, the TX signal becomes output (sending state), when the sending is completed, the sending buffer becomes empty, and the TX signal turns to Back to typing.
- When there is no data transmission, the TXD pin is always in the input state. The user program needs to configure this pin as an open-drain output and connect an external pull-up resistor.

Note:

1. Even if the module is in the receiving process, as long as the sending buffer is filled with data, the TX signal will immediately become output (transmitting state), the receiving process will be interrupted, and an incorrect received data will be obtained.
2. The similarities and differences between single-line half-duplex mode and Mode0 mode:
   Same: Both use one signal to time-division multiplex to complete data transmission and reception.
   Different: Mode0 is synchronous transmission with a synchronous clock, while the single-

wire half-duplex mode is asynchronous transmission, all baud rate calculations still follow the baud rate formula of Mode1-2-3, only the transceiver function is multiplexed to the TX signal.

21.3.2 Baud rate generation

Mode0~Mode3 are different, as shown below for details.

Mode0 baud rate generation formula:BaudRate= $\frac{f_{PCLK}}{12}$

Mode1 baud rate generation formula:BaudRate= $\frac{f_{PCLK}}{OVER*SCNT}$

Mode2 baud rate generation formula:BaudRate= $\frac{f_{PCLK}}{OVER}$

Mode3 baud rate generation formula:BaudRate= $\frac{f_{PCLK}}{OVER*SCNT}$

Note:

- f_{PCLK} represents the frequency of the current PCLK.
- The definition of OVER is detailed in UARTx_SCON.
- The definition of SCNT is detailed in UARTx_SCNT

21.3.2.1 Mode1/Mode3 baud rate setting example

Table 21-3 PCLK=4MHz baud rate calculation table

Baud rate	PCLK = 4 MHz					
	OVER8			OVER16		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	208	2403.85	0.16%	104	2403.85	0.16%
4800	104	4807.69	0.16%	52	4807.69	0.16%
9600	52	9615.38	0.16%	26	9615.38	0.16%
19200	26	19230.77	0.16%	13	19230.77	0.16%
38400	13	38461.54	0.16%	7	35714.29	-6.99%
57600	9	55555.56	-3.55%	4	62500.00	8.51%
76800	7	71428.57	-6.99%	3	83333.33	8.51%
115200	4	125000.00	8.51%	2	125000.00	8.51%
128000	4	125000.00	-2.34%	2	125000.00	-2.34%
250000	2	250000.00	0.00%	1	250000.00	0.00%

Table 21-4 PCLK=8MHz baud rate calculation table

Baud rate	PCLK = 8 MHz					
	OVER8			OVER16		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	417	2398.08	-0.08%	208	2403.85	0.16%
4800	208	4807.69	0.16%	104	4807.69	0.16%
9600	104	9615.38	0.16%	52	9615.38	0.16%
19200	52	19230.77	0.16%	26	19230.77	0.16%
38400	26	38461.54	0.16%	13	38461.54	0.16%
57600	17	58823.53	2.12%	9	55555.56	-3.55%
76800	13	76923.08	0.16%	7	71428.57	-6.99%
115200	9	111111.11	-3.55%	4	125000.00	8.51%
128000	8	125000.00	-2.34%	4	125000.00	-2.34%
256000	4	250000.00	-2.34%	2	250000.00	-2.34%
500000	2	500000.00	0.00%	1	500000.00	0.00%

Table 21-5 PCLK=16MHz baud rate calculation table

Baud rate	PCLK = 16 MHz					
	OVER8			OVER16		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	833	2400.96	0.04%	417	2398.08	-0.08%
4800	417	4796.16	-0.08%	208	4807.69	0.16%
9600	208	9615.38	0.16%	104	9615.38	0.16%
19200	104	19230.77	0.16%	52	19230.77	0.16%
38400	52	38461.54	0.16%	26	38461.54	0.16%
57600	35	57142.86	-0.79%	17	58823.53	2.12%
76800	26	76923.08	0.16%	13	76923.08	0.16%
115200	17	117647.06	2.12%	9	111111.11	-3.55%
128000	16	125000.00	-2.34%	8	125000.00	-2.34%
256000	8	250000.00	-2.34%	4	250000.00	-2.34%
500000	4	500000.00	0.00%	2	500000.00	0.00%
1000000	2	1000000.00	0.00%	1	1000000.00	0.00%

Table 21-6 PCLK=24MHz baud rate calculation table

Baud rate	PCLK = 24 MHz					
	OVER8			OVER16		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	1250	2400.00	0.00%	625	2400.00	0.00%
4800	625	4800.00	0.00%	313	4792.33	-0.16%
9600	313	9584.66	-0.16%	156	9615.38	0.16%
19200	156	19230.77	0.16%	78	19230.77	0.16%
38400	78	38461.54	0.16%	39	38461.54	0.16%
57600	52	57692.31	0.16%	26	57692.31	0.16%
76800	39	76923.08	0.16%	20	75000.00	-2.34%
115200	26	115384.62	0.16%	13	115384.62	0.16%
128000	23	130434.78	1.90%	12	125000.00	-2.34%
256000	12	250000.00	-2.34%	6	250000.00	-2.34%
1000000	3	1000000.00	0.00%	2	750000.00	-25.00%
1500000	2	1500000.00	0.00%	1	1500000.00	0.00%

Table 21-7 PCLK=32MHz baud rate calculation table

Baud rate	PCLK = 32 MHz					
	OVER8			OVER16		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	1667	2399.52	-0.02%	833	2400.96	0.04%
4800	833	4801.92	0.04%	417	4796.16	-0.08%
9600	417	9592.33	-0.08%	208	9615.38	0.16%
19200	208	19230.77	0.16%	104	19230.77	0.16%
38400	104	38461.54	0.16%	52	38461.54	0.16%
57600	69	57971.01	0.64%	35	57142.86	-0.79%
76800	52	76923.08	0.16%	26	76923.08	0.16%
115200	35	114285.71	-0.79%	17	117647.06	2.12%
128000	31	129032.26	0.81%	16	125000.00	-2.34%
256000	16	250000.00	-2.34%	8	250000.00	-2.34%
1000000	4	1000000.00	0.00%	2	1000000.00	0.00%
2000000	2	2000000.00	0.00%	1	2000000.00	0.00%

Table 21-8 PCLK=48MHz baud rate calculation table

Baud rate	PCLK = 48 MHz					
	OVER8			OVER16		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	2500	2400.00	0.00%	1250	2400.00	0.00%
4800	1250	4800.00	0.00%	625	4800.00	0.00%
9600	625	9600.00	0.00%	313	9584.66	-0.16%
19200	313	19169.33	-0.16%	156	19230.77	0.16%
38400	156	38461.54	0.16%	78	38461.54	0.16%
57600	104	57692.31	0.16%	52	57692.31	0.16%
76800	78	76923.08	0.16%	39	76923.08	0.16%
115200	52	115384.62	0.16%	26	115384.62	0.16%
128000	47	127659.57	-0.27%	23	130434.78	1.90%
256000	23	260869.57	1.90%	12	250000.00	-2.34%
1000000	6	1000000.00	0.00%	3	1000000.00	0.00%
2000000	3	2000000.00	0.00%	2	1500000.00	-25.00%
3000000	2	3000000.00	0.00%	1	3000000.00	0.00%

21.3.3 Frame error detection

When working in Mode1/2/3, UART has a frame error detection function. If the hardware does not recognize the stop bit within the expected time when receiving data, resulting in synchronization failure or excessive noise, a frame error will be detected. UARTx_ISR.FE is set when a framing error is detected. UARTx_ISR.FE should be cleared by software in time.

21.3.4 Multi-machine communication

When working in Mode2/3, set the UARTx_SCON.ADRDET bit to "1" to enable the multi-device communication function.

The host can use UARTx_SBUF[8] to distinguish whether the current sending frame is an address frame (UARTx_SBUF[8]=1) or a data frame (UARTx_SBUF[8]=0).

- When it is a data frame, the frame data will not be stored in the UARTx_SBUF register of the slave, and the slave will not generate a receive interrupt.
- When it is an address frame, since the automatic address recognition function in the multi-machine communication has been turned on, the slave can detect whether the received address matches its own address.
  - If the address matches, the slave will set "1" to UARTx_ISR.RC, set "1" to UARTx_SBUF[8], and store the address frame into the UARTx_SBUF register at the same time. After the slave software sees UARTx_SBUF[8]=1 and UARTx_ISR.RC=1, clear the UARTx_SCON.ADEDET bit to "0" and accept the data frame.
  - If the address does not match, it means that the master is not addressing the slave, the slave hardware keeps UARTx_SBUF[8] and UARTx_ISR.RC as "0", the software keeps the UARTx_SCON.ADRDET bit as "1", and continues to be in the address monitoring state.

Note: If necessary, the multi-machine communication bit can also be turned on in Mode1, and the TB8 bit is replaced by the stop bit. When the slave receives a matching address frame and a valid stop bit, UARTx_ISR.RC will be set to "1".

Slave Address Configuration

The slave address consists of the broadcast address and the user configuration address:

- Broadcast Address: UART_SADDR | UART_SADEN
- User Configuration Address: UART_SADDR & UART_SADEN

When an address frame is received, if the address received from the bus, after being ANDed with the value of UART_SADEN, matches the user-configured address, or if the received address, after being ANDed with the broadcast address, matches the broadcast address, it is considered that an address frame matching the slave address has been received.

Example: UART_SADDR=0x55, UART_SADEN=0xF0, then the slave can match addresses 0xF5, 0xF7, 0xFD, 0xFF (broadcast address) and 0x50~0x5F (user-configured address).

21.3.5 DMAC hardware handshake

UART module supports the hardware handshaking logic of the DMAC.

- Setting UARTx_SCON.DMACTXEN to 1 can turn on the DMAC hardware handshaking logic of UART TX. When the transmit buffer is empty, the UART will send a data transfer request TX REQ to the DMAC. DMAC receives this signal, it transfers the sending data of one frame from the DMAC source address to UARTx_SBUF. The above steps are repeated until all the data lengths configured in the DMAC are transferred.
- Setting UARTx_SCON.DMACRXEN to 1 can turn on the DMAC hardware handshaking logic of UART RX. When a frame is received, the UART will send a data transfer request RX REQ to the DMAC. DMAC receives this signal, it transfers the received data from UARTx_SBUF to the target address of DMAC. The above steps are repeated until all the data lengths configured in the DMAC are transferred.

21.3.6 Hardware flow control

UART hardware flow control can be realized by adding nCTS and nRTS signals, that is, the UART hardware module automatically controls the sending and receiving of data according to the high and low levels of nCTS and nRTS, without judging by software. The schematic diagram of hardware flow control between two UART modules is as follows:

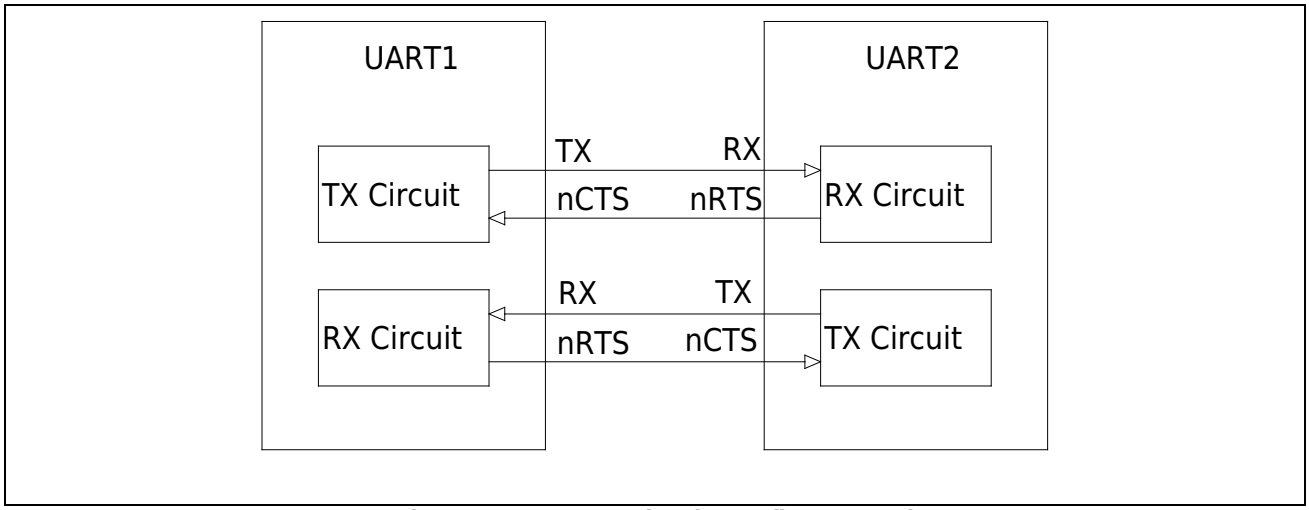


Figure 21-10 UART hardware flow control

- nRTS flow control

When nRTS flow control is enabled (UARTx_SCON.RTSEN is set to 1):

- nRTS will be asserted (connected low) when the UART receive buffer is empty.
- When the receive buffer is full, nRTS will become invalid (connected to a high level), indicating that the sending process will stop after the end of the current frame.

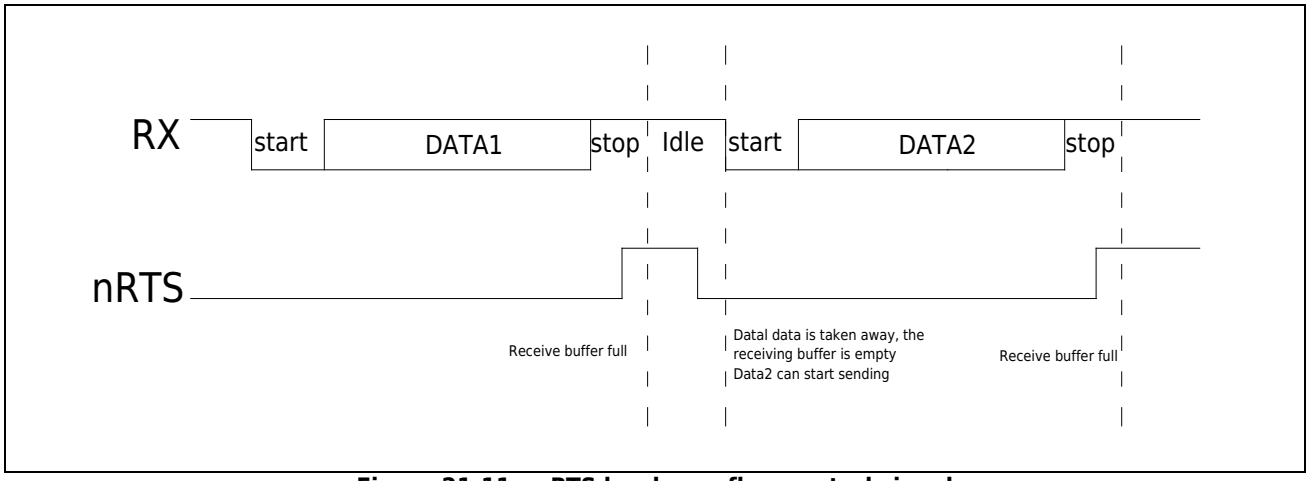


Figure 21-11 nRTS hardware flow control signal

- CTS flow control

When nCTS flow control is enabled (UARTx_SCON.CTSEN is set to 1), before the UART sends the next frame of data, first judge the high and low levels of nCTS:

- If nCTS is active (connected to low level), the UART sends the next frame of data.
- If nCTS is invalid (connected to high level), UART will suspend sending the next frame of data after the current frame is sent.

When nCTS flow control is enabled, once the nCTS signal is inverted, UARTx_SFLAG.CTSIE will be set to 1 by hardware. If UARTx_SCON.CTSIE is set to 1, an interrupt will be generated, and at the same time, the high and low levels of the nCTS signal will be recorded in the UARTx_SFLAG.CTS flag.

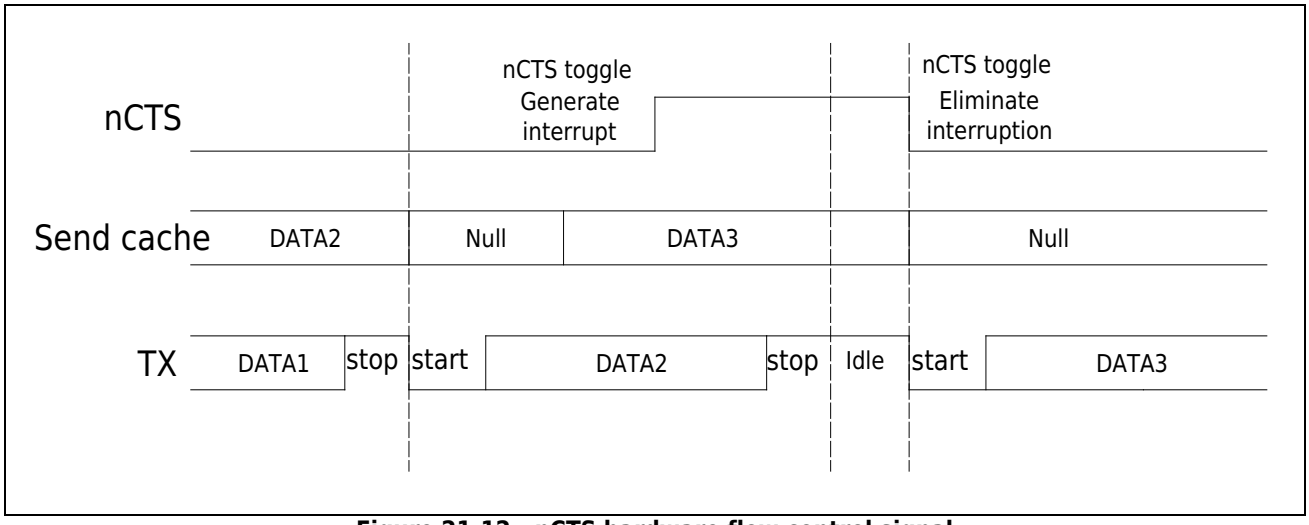


Figure 21-12 nCTS hardware flow control signal

21.3.7 Transceiver buffer

- Receive buffer

The receiving end of the UART module has a frame (8/9-Bit) receiving buffer, which saves the received data frame until the Stop bit of the next frame of data is received and then updates the data frame.

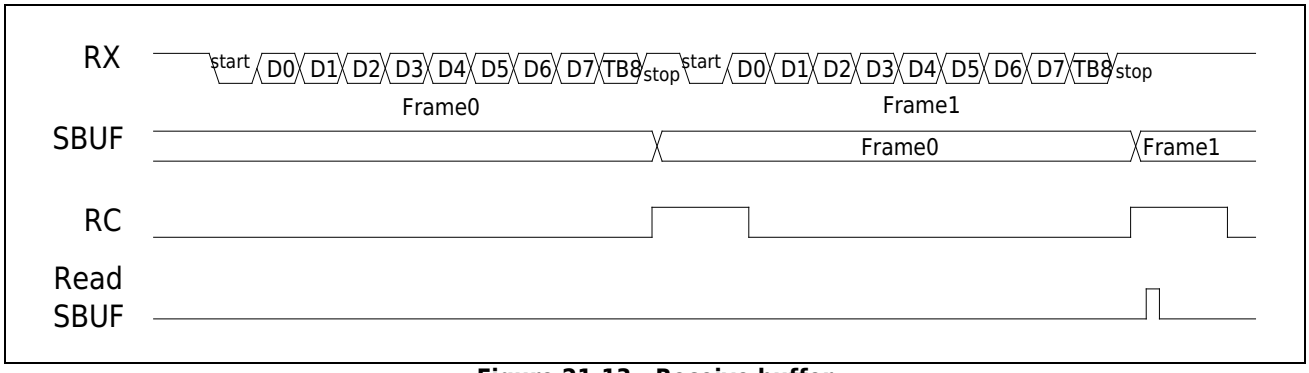


Figure 21-13 Receive buffer

■ Send cache

UART module has a frame (8/9-Bit) sending buffer. When the UART is sending the current frame, the software for the next sending data will be written into UARTx_SBUF.

When UARTx_ISR.TXE=0, it indicates that the current transmit buffer is full, and UARTx_SBUF cannot write the next transmit data. Otherwise the data will be discarded by hardware.

When UARTx_ISR.TXE=1, it indicates that the current sending buffer is empty, and UARTx_SBUF can write the next sending data. After the current data transmission is completed, the hardware automatically loads the data in the sending buffer into the shift register and sends it out.

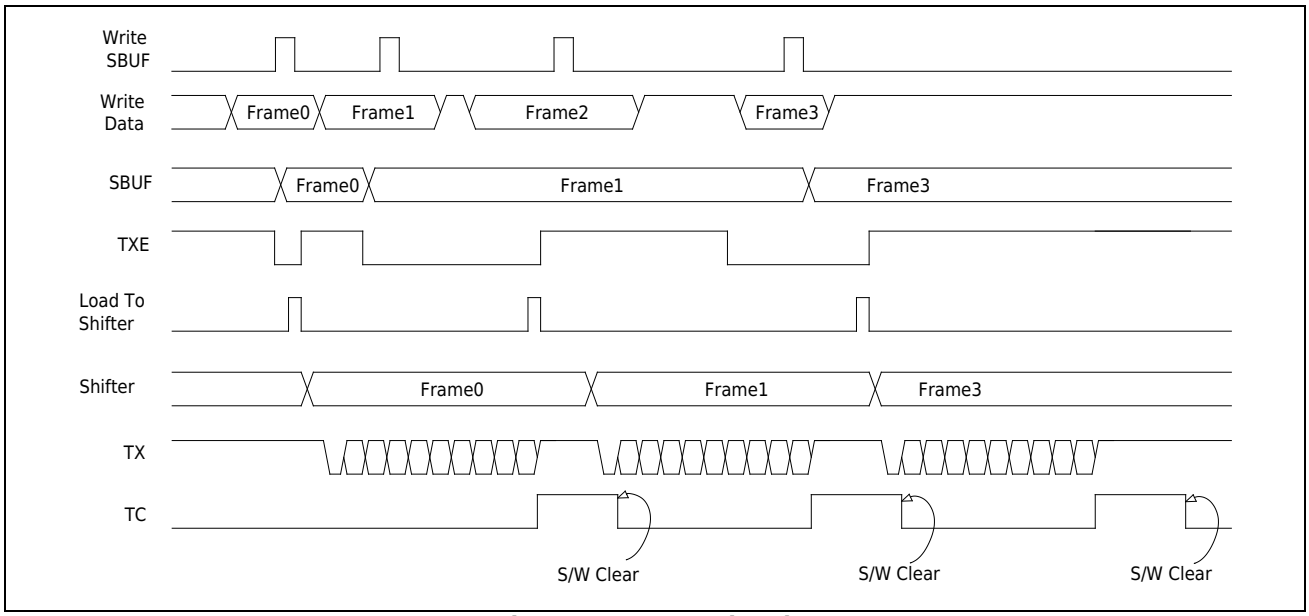


Figure 21-14 Send cache

21.4Register

UART0 base address: 0x4000 0000

UART1 base address: 0x4000 0100

UART2 base address: 0x4000 6000

UART3 base address: 0x4000 6400

Register	Offset address	Description
UARTx_SBUF	0x00	Data register
UARTx_SCON	0x04	control register
UARTx_SADDR	0x08	Address register
UARTx_SADEN	0x0C	Address mask register
UARTx_ISR	0x10	interrupt flag register
UARTx_ICR	0x14	Interrupt flag bit clear register
UARTx_SCNT	0x18	Baud rate register

21.4.1 Data Register (UARTx_SBUF)

Offset address: 0x00

Reset value: 0x0000 0000

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

Bit	Marking	Functional description
31:9	Reserved	
8	DATA[8]	In Mode0/1, read this bit is 0, write this bit is invalid; In Mode2/3, this bit represents the Bit8 data bit, which can be divided into the following two situations: (1) When the hardware parity bit is turned on, this bit is the parity bit of the received data when receiving, and the verification is performed by the hardware. If the verification error occurs, the verification error flag bit PE is set to 1; this bit is invalid when sending, The parity bit of the sent data is calculated and sent by the hardware; (2) When the hardware parity bit is turned off, this bit is received data Bit8 when receiving; this bit is sent data Bit8 when sending; Note: When the multi-machine communication mode is turned on, the parity check of the received data is automatically closed; the parity check of the sent data is still controlled by B8CONT;
7:0	DATA[7:0]	When sending data, write the sending data into this register; when receiving data, read from this register after the data is received.

21.4.2 Control Register (UARTx_SCON)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									HDSEL	FEIE	CTSIE	CTSEN	RTSEN	DMATXEN	DMARXEN
									RW	RW	RW	RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOPBIT		PEIE	Reserved			OVER	TXEIE	SM		ADRDET	REN	B8CONT		TCIE	RCIE
RW		RW				RW	RW	RW		RW	RW	RW		RW	RW

Bit	Marking	Functional description
31:23	Reserved	
22	HDSEL	Single-wire half-duplex mode enable bit; 0: Close single-line half-duplex mode; 1: open single-line half-duplex mode;
21	FEIE	Framing error interrupt enable bit; 0: Disable interrupt; 1: Enable interrupt;
20	CTSIE	CTS signal flip interrupt enable bit; 0: Disable interrupt; 1: Enable interrupt;
19	CTSEN	Hardware flow control signal enable bit; 0: Turn off the flow control signal; 1: Turn on the flow control signal;
18	RTSEN	
17	DMATXEN	The hardware handshake signal enable bit of TX DMAC; 0: Turn off the hardware handshake signal; 1: Turn on the hardware handshake signal;
16	DMARXEN	RX DMAC hardware handshake signal enable bit; 0: Turn off the hardware handshake signal; 1: Turn on the hardware handshake signal;
15:14	STOPBIT	stop bit length selection; 00:1-bit; 01:1.5-bit; 10:2-bit; 11: reserved; Note: Although there is no Stop Bit in Mode0, it is still necessary to keep STOPBIT[1:0] as 2'b00;
13	PEIE	Parity error interrupt enable bit; 0: Disable parity error interrupt; 1: Enable parity error interrupt; The data receiving flag is generated when the data stop bit is received. The hardware parity check and the data receiving interrupt are different due to the stop bit settings. The parity check error interrupt will be ahead of the data receiving interrupt. Be careful when using this interrupt enable. Suggestions below: Method 1: Disable the PEIE interrupt enable, and the software judges whether the ISR.PE parity check is correct after receiving the data interrupt. Method 2: Disable the PEIE interrupt enable, receive data interrupt and judge whether the parity is correct by receiving SBUF.BIT8 software.
12:10	Reserved	
9	OVER	Mode0: Invalid; Mode1/3: 0: 16 sampling frequency division; 1: 8 sampling frequency division; Mode2: 0: 32 sampling frequency division; 1: 16 sampling frequency division;
8	TXEIE	TX empty interrupt enable bit; 0: TX Buffer empty interrupt is off; 1: TX Buffer empty interrupt is on;
7:6	SM	Working mode; 00: mode0; 01: mode1; 10: mode2; 11: mode3;
5	ADRDET	Multi-machine communication address automatic identification enable bit; 0: off; 1: on;
4	REN	Mode0: 0: send; 1: receive; Mode1/2/3: 0: send; 1: receive / send;
3:2	B8CONT	Bit8 data control bit; 00: Determined by software reading and writing SBUF[8]; 01: Hardware even parity; 10: hardware odd parity; 11: reserved;
1	TCIE	Send interrupt enable bit; 0: Send interrupt is off; 1: Send interrupt is on;
0	RCIE	Receive interrupt enable bit; 0: Receive interrupt is off; 1: Receive interrupt is on;

21.4.3 Address Register (UARTx_SADDR)

Offset address: 0x08

Reset value: 0x0000 0000

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

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

Bit	Marking	Functional description
31:8	Reserved	
7:0	SADDR	Slave Device Address Register

21.4.4 Address Mask Register (UARTx_SADEN)

Offset address: 0x0C

Reset value: 0x0000 0000

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

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

Bit	Marking	Functional description
31:8	RESERVED	
7:0	SADEN	Slave Device Address Mask Register

21.4.5 Flag Register (UARTx_ISR)

Offset address: 0x10

Reset value: 0x0000 0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									CTS	CTSIF	PE	TXE	FE	TC	RC
									RO	RO	RO	RO	RO	RO	RO

Bit	Symbol	Description
31:7	Reserved	
6	CTS	CTS signal flag; hardware set 1; hardware clear; 0: CTS signal is low level; 1: CTS signal is high level;
5	CTSIF	CTS interrupt flag; hardware set to 1; software to clear; 0: CTS signal is not inverted; 1: CTS signal is inverted;
4	PE	Parity error flag; set to 1 by hardware; cleared by software; 0: no parity error; 1: parity error;
3	TXE	Tx Buffer empty flag; set by hardware; cleared by hardware; 0: Tx Buffer is not empty; 1: Tx Buffer is empty
2	FE	Framing error flag; 0: set by hardware; cleared by software;
1	TC	Transmit complete interrupt flag; set to 1 by hardware; cleared by software; 0: Transmit not completed; 1: Transmit complete;
0	RC	Receive complete interrupt flag; hardware set 1; software clear; 0: receive not completed; 1: receive complete;

21.4.6 Flag Clear Register (UARTx_ICR)

Offset address: 0x14

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										CTSIFCF	PECF	Res.	FECF	TCCF	RCCF
										R1W 0	R1W 0		R1W 0	R1W 0	R1W 0

Bit	Marking	Functional description
31:6	Reserved	
5	CTSIFCF	CTSIF flag clear bit; write 0 to clear; write 1 to be invalid;
4	PECF	PE flag clear bit; write 0 to clear; write 1 to be invalid;
3	Reserved	
2	FECF	FE flag clear bit; write 0 to clear; write 1 to be invalid;
1	TCCF	TC flag clear bit; write 0 to clear; write 1 to be invalid;
0	RCCF	RC flag clear bit; write 0 to clear; write 1 to be invalid;

21.4.7 Baud Rate Register (UARTx_SCNT)

Offset address: 0x18

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
SCNT															
RW															

Bit	Marking	Functional description
31:16	Reserved	
15:0	SCNT	Baud rate counter

22 Low Power Synchronous Asynchronous Transceiver (LPUART)

22.1 Introduction

The Low Power Synchronous Asynchronous Receiver (LPUART) is a UART that allows full-duplex UART communication with limited power consumption. Even when the microcontroller is in stop mode with very low power consumption, the LPUART waits for the arrival of UART frames. The LPUART contains all necessary hardware support to enable asynchronous serial communication with minimal power consumption.

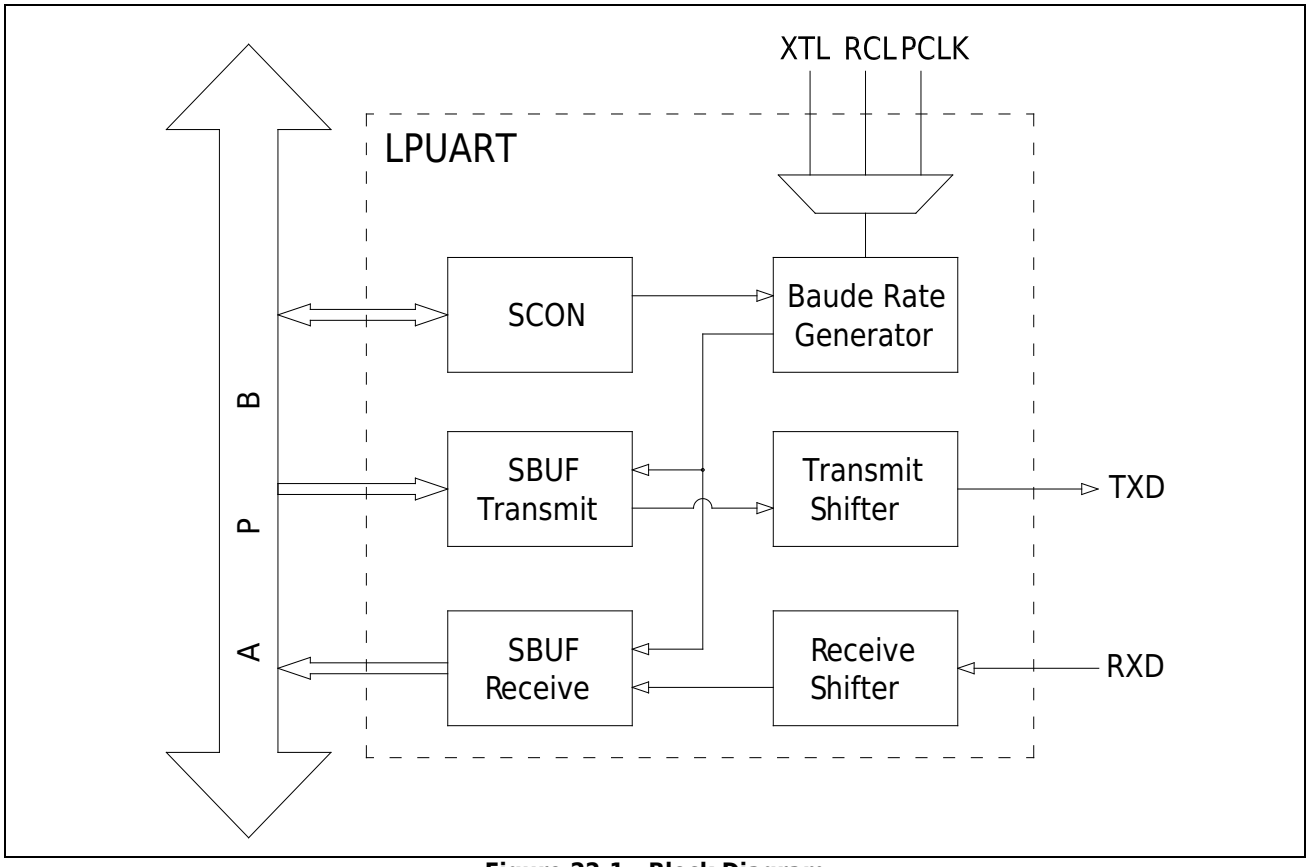


Figure 22-1 Block Diagram

22.2Main characteristics

LPUART module (LPUART0/1) supports the following basic functions:

- Configuration clock PCLK
- Transmission clock SCLK (SCLK can choose XTL, RCL and PCLK)
- Send and receive data in system low power mode
- Full-duplex transmission, half-duplex transmission, single-wire half-duplex transmission
- Programmable serial communication function
  - Two character lengths: 8 bits, 9 bits
  - Three checkout methods: no checkout, odd checkout, even checkout
  - Three stop lengths: 1 bit, 2 bits, 1.5 bits
- 16- bit baud rate counter
- Multi-machine communication
- Hardware address recognition
- Hardware flow control
- DMAC hardware transmission handshake

22.3Functional description

22.3.1 Configuration Clock and Transmit Clock

The LPUART module has two clocks: configuration clock PCLK and transfer clock SCLK.

- Configure clock

The configuration clock (PCLK) is used for register configuration of the LPUART module by the system APB bus, and the configuration clock is fixed as the APB bus clock PCLK. When the system enters Deep Sleep (DeepSleep) mode, the PCLK clock will stop.

- Transmission clock

The transmission clock (SCLK) is used for LPUART data transceiver logic work, and the external low-speed crystal oscillator clock (XTL), the internal low-speed RC clock (RCL) and the PCLK clock can be selected.

When the system enters Deep Sleep (DeepSleep) mode, if SCLK selects external low-speed crystal oscillator clock (XTL) or internal low-speed RC clock (RCL). LPUART can still perform normal data transmission and reception without being affected by the system's deep sleep (DeepSleep) mode.

22.3.2 Operating mode

LPUART supports multiple working modes: synchronous half-duplex mode, asynchronous full-duplex mode, and single-wire half-duplex mode. Through the combination of LPUARTx_SCON.SM and LPUARTx_SCON.HDSEL, various working modes can be configured.

22.3.2.1 Mode0~Mode3 function comparison

Configure LPUARTx_SCON.SM to select different transmission modes: Mode0~Mode3. The main function comparison of these four working modes is shown in the table below:

Table 22-1 Mode0/1/2/3 Data Structure

Operating mode		Transmission bit width	Data composition	Baud rate
Mode0	synchronous mode Half duplex	8-Bit	Data(8bit)	$BaudRate=\frac{f_{sclk}}{12}$
Mode1	Asynchronous mode Full duplex	10-Bit	Start(1bit) + Data(8bit) + Stop (1~2bit)	$BaudRate=\frac{f_{sclk}}{OVER*SCNT}$
Mode2	Asynchronous mode Full duplex	11-Bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop (1~2bit)	$BaudRate=\frac{f_{sclk}}{OVER}$
Mode3	Asynchronous mode Full duplex	11-Bit	Start (1bit) + Data(8bit) + B8(1bit) + Stop (1~2bit)	$BaudRate=\frac{f_{sclk}}{OVER*SCNT}$

Note:

- Mode0 can only be used as a host to send LPUART synchronous shift clock, and cannot be used as a slave to receive externally input LPUART synchronous shift clock;
- f_{sclk} is the clock frequency of SCLK;
- The definition of OVER is detailed in LPUARTx_SCON;
- LPUARTx_SCNT for the definition of SCNT;
- The B8 data bit is special and has different meanings in different applications, please refer to the following table:

Table 22-2 B8 Data Meaning

Application Scenario	LPUARTx_SCON. ADRDET	LPUARTx_SCON. B8CONT[1:0]	B8 Data Meaning
Parity	--	01/10	When receiving, B8 is the parity bit of the received 8-Bit data; When sending, B8 is the parity bit of the sent 8-Bit data;
Multi-machine communication	1	--	B8=1, it means that it is currently an address frame; B8=0, it means the current data frame;
Other	0	00/11	Received / sent DATA[8]

Note:

- When the multi-machine communication mode is turned on, the parity check of the received data is automatically closed; the parity check of the sent data is still controlled by B8CONT.

22.3.2.2 Mode0 data sending and receiving instructions

When sending data, clear the LPUARTx_SCON.REN bit and write the data into the LPUARTx_SBUF register. At this time, the sending data will be output from RXD (low bit first, high bit later), and the synchronous shift clock will be output from TXD.

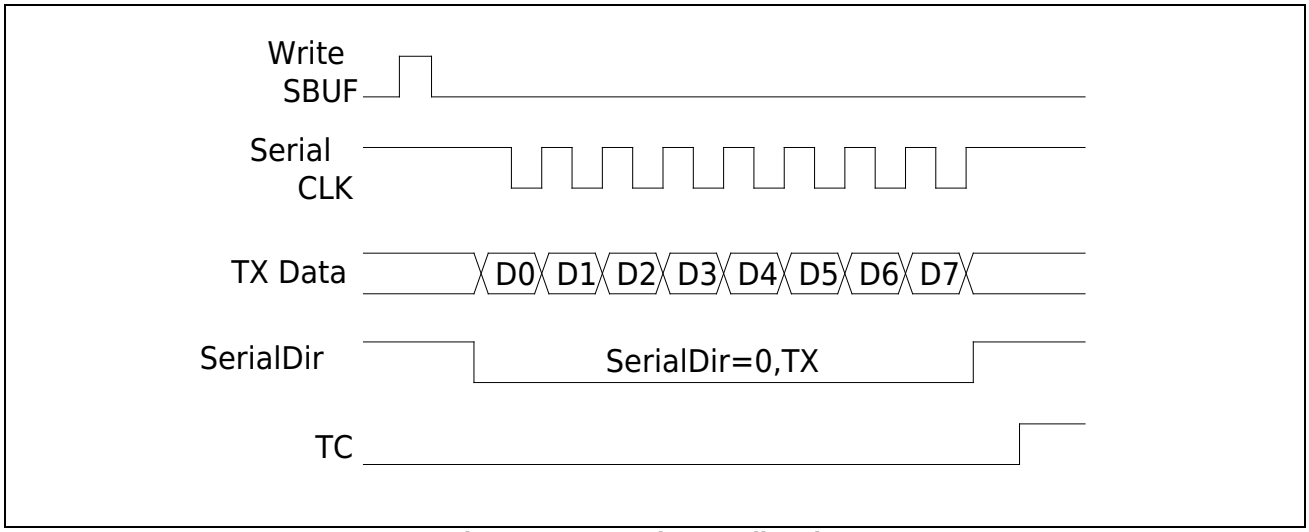


Figure 22-2 Mode0 sending data

When receiving data, set the LPUARTx_SCON.REN bit and clear the LPUARTx_ISR.RC bit. When reception is complete, data can be read from the LPUARTx_SBUF register. At this time, the received data is input from RXD (low bit first, high bit later), and the synchronous shift clock is output from TXD.

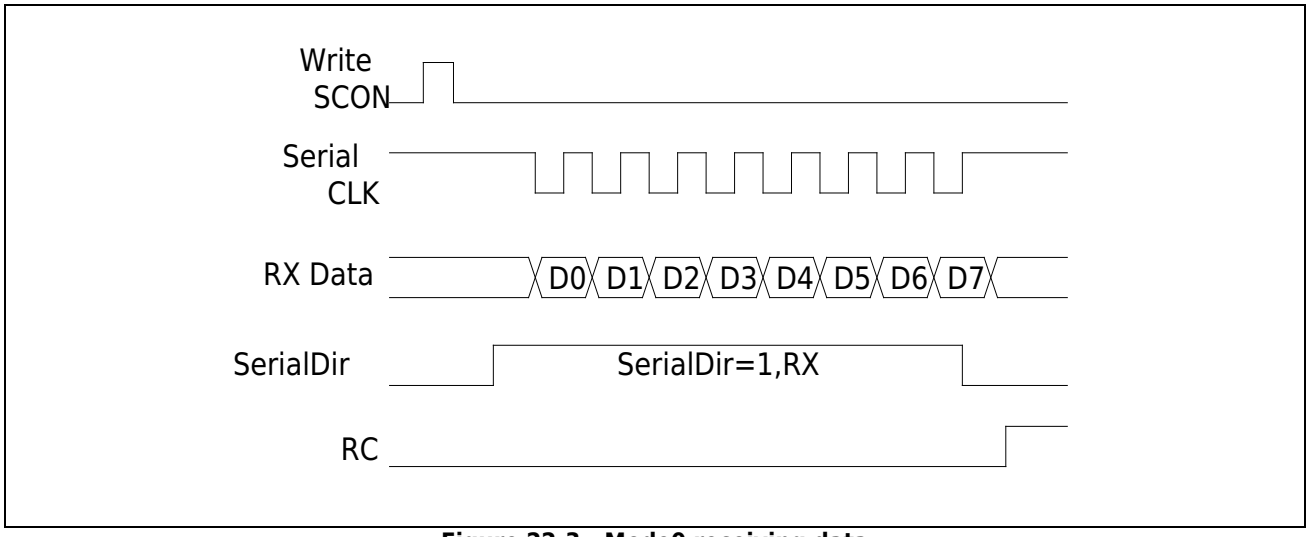


Figure 22-3 Mode0 receiving data

22.3.2.3 Mode1 data sending and receiving instructions

When sending data, it has nothing to do with the value of LPUARTx_SCON.REN, write the sent data into the LPUARTx_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

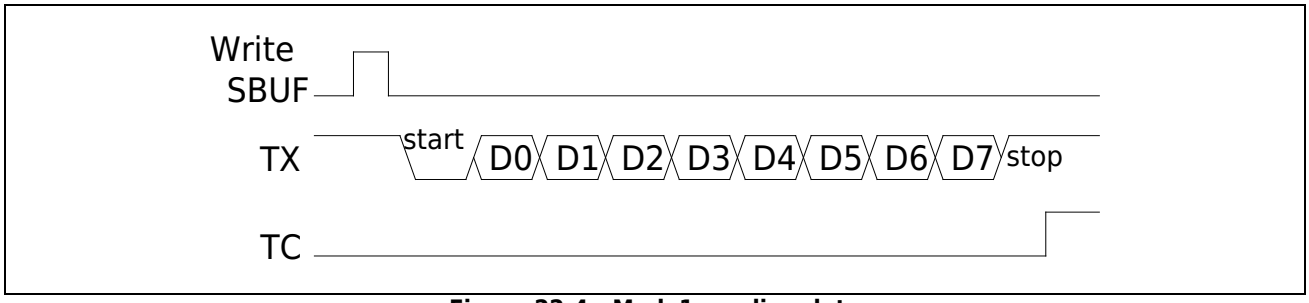


Figure 22-4 Mode1 sending data

When receiving data, set the LPUARTx_SCON.REN bit to 1 and clear the LPUARTx_ISR.RC bit to 0. Start to receive data on RXD (low bit first, high bit later), when the reception is complete, it can be read from the LPUARTx_SBUF register.

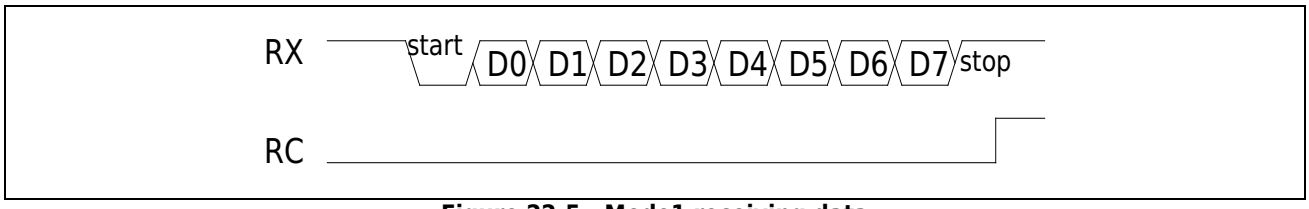


Figure 22-5 Mode1 receiving data

22.3.2.4 Mode2 data sending and receiving instructions

When sending data, it has nothing to do with the value of LPUARTx_SCON.REN, write the sent data into the LPUARTx_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

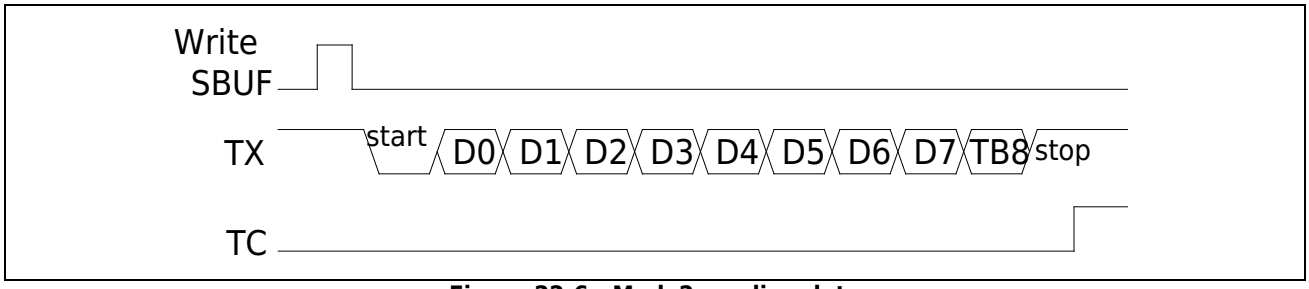


Figure 22-6 Mode2 sending data

When receiving data, it is necessary to set the LPUARTx_SCON.REN bit to 1, and clear the LPUARTx_ISR.RC bit to 0. Start to receive the data on RXD (low bit first, high bit later), when the reception is complete, it can be read from the LPUARTx_SBUF register.

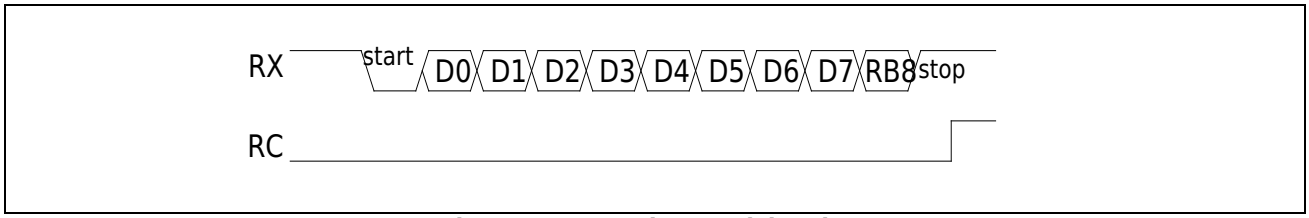


Figure 22-7 Mode2 receiving data

22.3.2.5 Mode3 data sending and receiving instructions

When sending data, it has nothing to do with the value of LPUARTx_SCON.REN, write the sent data into the LPUARTx_SBUF register, and the data will be shifted out from TXD (low bit first, high bit later).

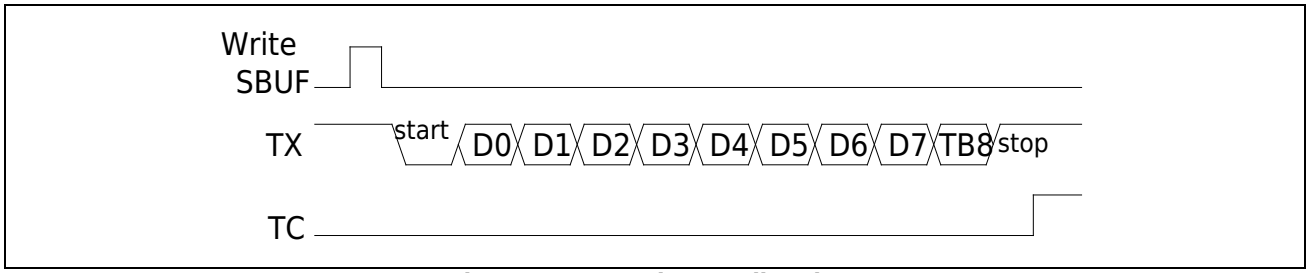


Figure 22-8 Mode3 sending data

When receiving data, set the LPUARTx_SCON.REN bit to 1 and clear the LPUARTx_ISR.RC bit to 0. Start to receive data on RXD (low bit first, high bit later), when the reception is complete, it can be read from the LPUARTx_SBUF register.

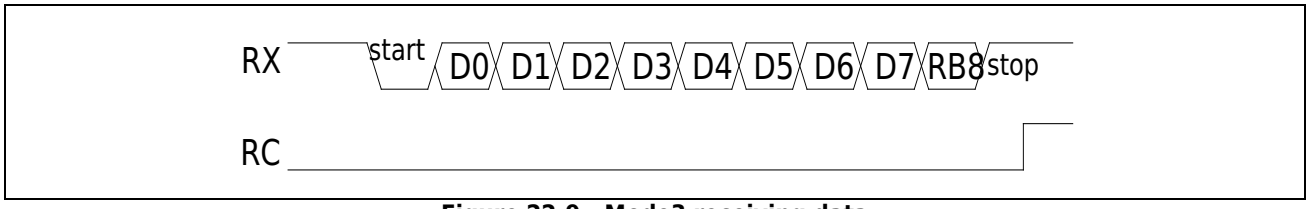


Figure 22-9 Mode3 receiving data

22.3.2.6 Single-wire half-duplex data transmitting and reception

When LPUARTx_SCON.HDSEL=1'b1, it can enter single-wire half-duplex transmission mode:

- The TX and RX signal lines are connected inside the module.
- The RX signal is no longer used, and the received and sent data are all completed through the TX signal. TX signal is done by hardware logic without software control.
- After configuring LPUART_SCON.HDSEL to 0b1, must wait for the transmission duration of one data bit (i.e., after 1/BaudRate seconds) before normal reception can occur.
- When the transmit buffer is empty, the TX signal is always input (receive state). Once a data is filled into the transmit buffer, the TX signal becomes output (transmit status). When the transmission is complete, the transmit buffer becomes empty and the TX signal returns to input (receive state).
- When there is no data transmission, the TXD pin is always in the input state. The user program needs to configure this pin as an open-drain output and connect an external pull-up resistor.

Note:

1. Even if the module is in the process of receiving, as long as the sending buffer is filled with data, the TX signal will immediately become output (transmitting state). The receiving process will be interrupted and a wrong received data will be obtained.

2. single-line half-duplex mode and Mode0 mode:

   The same: both use a signal to time-division multiplex to complete data transmission and reception.

   Different: Mode0 is synchronous transmission, with synchronous clock. The single-wire half-duplex mode is asynchronous transmission, and all baud rate calculations still follow the baud rate formula of Mode1~Mode3, only the transceiver function is multiplexed on the TX signal.

22.3.3 Baud rate generation

Mode0~Mode3 are different, as shown below for details.

Mode0 baud rate generation formula:BaudRate= $\frac{f_{SCLK}}{12}$

Mode1 baud rate generation formula:BaudRate= $\frac{f_{SCLK}}{OVER*SCNT}$

Mode2 baud rate generation formula:BaudRate= $\frac{f_{SCLK}}{OVER}$

Mode3 baud rate generation formula:BaudRate= $\frac{f_{SCLK}}{OVER*SCNT}$

Note:

- f_{SCLK} represents the frequency of the current SCLK;
- The definition of OVER is detailed in LPUARTx_SCON;
- LPUARTx_SCNT for the definition of SCNT.

22.3.3.1 Mode1/Mode3 baud rate setting example

Table 22-3 SCL is 4MHz baud rate calculation table

Baud rate	SCLK = 4 MHz								
	OVER4			OVER8			OVER16		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error %
2400	417	2398.08	-0.08%	208	2403.85	0.16%	104	2403.85	0.16%
4800	208	4807.69	0.16%	104	4807.69	0.16%	52	4807.69	0.16%
9600	104	9615.38	0.16%	52	9615.38	0.16%	26	9615.38	0.16%
19200	52	19230.77	0.16%	26	19230.77	0.16%	13	19230.77	0.16%
38400	26	38461.54	0.16%	13	38461.54	0.16%	7	35714.29	-6.99%
57600	17	58823.53	2.12%	9	55555.56	-3.55%	4	62500.00	8.51%
76800	13	76923.08	0.16%	7	71428.57	-6.99%	3	83333.33	8.51%
115200	9	111111.11	-3.55%	4	125000.00	8.51%	2	125000.00	8.51%
128000	8	125000.00	-2.34%	4	125000.00	-2.34%	2	125000.00	-2.34%
256000	4	250000.00	-2.34%	2	250000.00	-2.34%	1	250000.00	-2.34%
1000000	1	1000000.00	0.00%	1	500000.00	-50.00%	0	/	/

Table 22-4 SCLK is 8MHz baud rate calculation table

Baud rate	SCLK = 8 MHz								
	OVER4			OVER8			OVER16		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	833	2400.96	0.04%	417	2398.08	-0.08%	208	2403.85	0.16%
4800	417	4796.16	-0.08%	208	4807.69	0.16%	104	4807.69	0.16%
9600	208	9615.38	0.16%	104	9615.38	0.16%	52	9615.38	0.16%
19200	104	19230.77	0.16%	52	19230.77	0.16%	26	19230.77	0.16%
38400	52	38461.54	0.16%	26	38461.54	0.16%	13	38461.54	0.16%
57600	35	57142.86	-0.79%	17	58823.53	2.12%	9	55555.56	-3.55%
76800	26	76923.08	0.16%	13	76923.08	0.16%	7	71428.57	-6.99%
115200	17	117647.06	2.12%	9	111111.11	-3.55%	4	125000.00	8.51%
128000	16	125000.00	-2.34%	8	125000.00	-2.34%	4	125000.00	-2.34%
256000	8	250000.00	-2.34%	4	250000.00	-2.34%	2	250000.00	-2.34%
1000000	2	1000000.00	0.00%	1	1000000.00	0.00%	1	500000.00	-50.00%
2000000	1	2000000.00	0.00%	1	1000000.00	-50.00%	0	/	/

Table 22-5 SCLK is 16MHz baud rate calculation table

Baud rate	SCLK = 16 MHz								
	OVER4			OVER8			OVER16		
	CNT	Actual baud rate	Error %	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	1667	2399.52	-0.02%	833	2400.96	0.04%	417	2398.08	-0.08%
4800	833	4801.92	0.04%	417	4796.16	-0.08%	208	4807.69	0.16%
9600	417	9592.33	-0.08%	208	9615.38	0.16%	104	9615.38	0.16%
19200	208	19230.77	0.16%	104	19230.77	0.16%	52	19230.77	0.16%
38400	104	38461.54	0.16%	52	38461.54	0.16%	26	38461.54	0.16%
57600	69	57971.01	0.64%	35	57142.86	-0.79%	17	58823.53	2.12%
76800	52	76923.08	0.16%	26	76923.08	0.16%	13	76923.08	0.16%
115200	35	114285.71	-0.79%	17	117647.06	2.12%	9	111111.11	-3.55%
128000	31	129032.26	0.81%	16	125000.00	-2.34%	8	125000.00	-2.34%
256000	16	250000.00	-2.34%	8	250000.00	-2.34%	4	250000.00	-2.34%
1000000	4	1000000.00	0.00%	2	1000000.00	0.00%	1	1000000.00	0.00%
2000000	2	2000000.00	0.00%	1	2000000.00	0.00%	1	1000000.00	-50.00%
4000000	1	4000000.00	0.00%	1	2000000.00	-50.00%	0	/	/

Table 22-6 SCLK is 24MHz baud rate calculation table

Baud rate	SCLK = 24 MHz								
	OVER4			OVER8			OVER16		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	2500	2400.00	0.00%	1250	2400.00	0.00%	625	2400.00	0.00%
4800	1250	4800.00	0.00%	625	4800.00	0.00%	313	4792.33	-0.16%
9600	625	9600.00	0.00%	313	9584.66	-0.16%	156	9615.38	0.16%
19200	313	19169.33	-0.16%	156	19230.77	0.16%	78	19230.77	0.16%
38400	156	38461.54	0.16%	78	38461.54	0.16%	39	38461.54	0.16%
57600	104	57692.31	0.16%	52	57692.31	0.16%	26	57692.31	0.16%
76800	78	76923.08	0.16%	39	76923.08	0.16%	20	75000.00	-2.34%
115200	52	115384.62	0.16%	26	115384.62	0.16%	13	115384.62	0.16%
128000	47	127659.57	-0.27%	23	130434.78	1.90%	12	125000.00	-2.34%
256000	23	260869.57	1.90%	12	250000.00	-2.34%	6	250000.00	-2.34%
1000000	6	1000000.00	0.00%	3	1000000.00	0.00%	2	750000.00	-25.00%
2000000	3	2000000.00	0.00%	2	1500000.00	-25.00%	1	1500000.00	-25.00%
6000000	1	6000000.00	0.00%	1	3000000.00	-50.00%	0	/	/

Table 22-7 SCLK is 32MHz baud rate calculation table

Baud rate	SCLK = 32 MHz								
	OVER4			OVER8			OVER16		
	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%	CNT	Actual baud rate	Error%
2400	3333	2400.24	0.01%	1667	2399.52	-0.02%	833	2400.96	0.04%
4800	1667	4799.04	-0.02%	833	4801.92	0.04%	417	4796.16	-0.08%
9600	833	9603.84	0.04%	417	9592.33	-0.08%	208	9615.38	0.16%
19200	417	19184.65	-0.08%	208	19230.77	0.16%	104	19230.77	0.16%
38400	208	38461.54	0.16%	104	38461.54	0.16%	52	38461.54	0.16%
57600	139	57553.96	-0.08%	69	57971.01	0.64%	35	57142.86	-0.79%
76800	104	76923.08	0.16%	52	76923.08	0.16%	26	76923.08	0.16%
115200	69	115942.03	0.64%	35	114285.71	-0.79%	17	117647.06	2.12%
128000	63	126984.13	-0.79%	31	129032.26	0.81%	16	125000.00	-2.34%
256000	31	258064.52	0.81%	16	250000.00	-2.34%	8	250000.00	-2.34%
1000000	8	1000000.00	0.00%	4	1000000.00	0.00%	2	1000000.00	0.00%
2000000	4	2000000.00	0.00%	2	2000000.00	0.00%	1	2000000.00	0.00%
4000000	2	4000000.00	0.00%	1	4000000.00	0.00%	1	2000000.00	-50.00%
8000000	1	8000000.00	0.00%	1	4000000.00	-50.00%	0	/	/

Table 22-8 SCLK is 48MHz baud rate calculation table

Baud rate	SCLK = 48 MHz								
	OVER4			OVER8			OVER16		
	CNT	Actual baud rate	Error %	CNT	Actual baud rate	Error %	CNT	Actual baud rate	Error%
2400	5000	2400.00	0.00%	2500	2400.00	0.00%	1250	2400.00	0.00%
4800	2500	4800.00	0.00%	1250	4800.00	0.00%	625	4800.00	0.00%
9600	1250	9600.00	0.00%	625	9600.00	0.00%	313	9584.66	-0.16%
19200	625	19200.00	0.00%	313	19169.33	-0.16%	156	19230.77	0.16%
38400	313	38338.66	-0.16%	156	38461.54	0.16%	78	38461.54	0.16%
57600	208	57692.31	0.16%	104	57692.31	0.16%	52	57692.31	0.16%
76800	156	76923.08	0.16%	78	76923.08	0.16%	39	76923.08	0.16%
115200	104	115384.62	0.16%	52	115384.62	0.16%	26	115384.62	0.16%
128000	94	127659.57	-0.27%	47	127659.57	-0.27%	23	130434.78	1.90%
256000	47	255319.15	-0.27%	23	260869.57	1.90%	12	250000.00	-2.34%
1000000	12	1000000.00	0.00%	6	1000000.00	0.00%	3	1000000.00	0.00%
2000000	6	2000000.00	0.00%	3	2000000.00	0.00%	2	1500000.00	-25.00%
4000000	4	3000000.00	0.00%	2	3000000.00	0.00%	1	3000000.00	0.00%
6000000	3	4000000.00	0.00%	2	3000000.00	/	1	3000000.00	/
12000000	1	12000000.00	0.00%	1	6000000.00	/	0	/	/

22.3.4 Frame error detection

When working in Mode1/2/3, LPUART has a frame error detection function. If the stop bit is not recognized within the expected time when receiving data, resulting in synchronization failure or excessive noise, a frame error will be detected. LPUARTx_ISR.FE is set to 1 when a framing error is detected. LPUARTx_ISR.FE should be cleared by software in time.

22.3.5 Multi-machine communication

When working in Mode2/3, set the LPUARTx_SCON.ADRDET bit to "1" to enable the multi-device communication function.

The host can use LPUARTx_SBUF[8] to distinguish whether the current sending frame is an address frame (LPUARTx_SBUF[8]=1) or a data frame (LPUARTx_SBUF[8]=0).

- When it is a data frame, the frame data will not be stored in the LPUARTx_SBUF register of the slave, and the slave will not generate a receive interrupt.
- When it is an address frame, since the automatic address recognition function in the multi-machine communication has been turned on, the slave can detect whether the received address matches its own address.
  - If the address matches, the slave will set "1" to LPUARTx_ISR.RC, set "1" to LPUARTx_SBUF[8], and store the address frame into the LPUARTx_SBUF register at the same time. LPUARTx_SBUF[8]=1 and LPUARTx_ISR.RC=1, the slave software clears the LPUARTx_SCON.ADEDET bit to "0" and accepts the data frame.
  - If the address does not match, it means that the master is not addressing the slave, the slave hardware keeps LPUARTx_SBUF[8] and LPUARTx_ISR.RC as "0", the software keeps the LPUARTx_SCON.ADRDET bit as "1", and the slave continues to be in the address monitoring state.

Note:

- If necessary, the multi-device communication bit can also be turned on in Mode1, and the TB8 bit is replaced by the stop bit. When the slave receives a matching address frame and a valid stop bit, LPUARTx_ISR.RC will be set to "1".

Slave Address Configuration

The slave address consists of the broadcast address and the user configuration address:

- Broadcast Address: LPUART_SADDR | LPUART_SADEN
- User Configuration Address: LPUART_SADDR & LPUART_SADEN

When an address frame is received, if the address received from the bus, after being ANDed with the value of LPUART_SADEN, matches the user-configured address, or if the received address, after being ANDed with the broadcast address, matches the broadcast address, it is considered that an address frame matching the slave address has been received.

Example: LPUART_SADDR=0x55, LPUART_SADEN=0xF0, then the slave can match addresses 0xF5, 0xF7, 0xFD, 0xFF (broadcast address) and 0x50~0x5F (user-configured address).

22.3.6 DMAC hardware handshake

The LPUART module supports the hardware handshaking logic of the DMAC.

- Setting LPUARTx_SCON.DMACTXEN to 1 can turn on the DMAC hardware handshaking logic of LPUART TX. When the send buffer is empty, the LPUART will send a data transfer request TX REQ to the DMAC. the DMAC receives this signal, it transfers the sending data of one frame from the DMAC source address to LPUARTx_SBUF. The above steps are repeated until all the data lengths configured in the DMAC are transferred.
- Setting LPUARTx_SCON.DMACRXEN to 1 can turn on the DMAC hardware handshaking logic of LPUART RX. When a frame is received, LPUART will send a data transfer request RX REQ to DMAC. When DMAC receives this signal, it transfers the received data from LPUARTx_SBUF to the target address of DMAC. The above steps are repeated until all the data lengths configured in the DMAC are transferred.

22.3.7 Hardware flow control

LPUART hardware flow control can be realized by adding nCTS and nRTS signals, that is, the LPUART hardware module automatically controls the sending and receiving of data according to the high and low levels of nCTS and nRTS, without judging by software. The hardware flow control diagram between two LPUART modules is as follows:

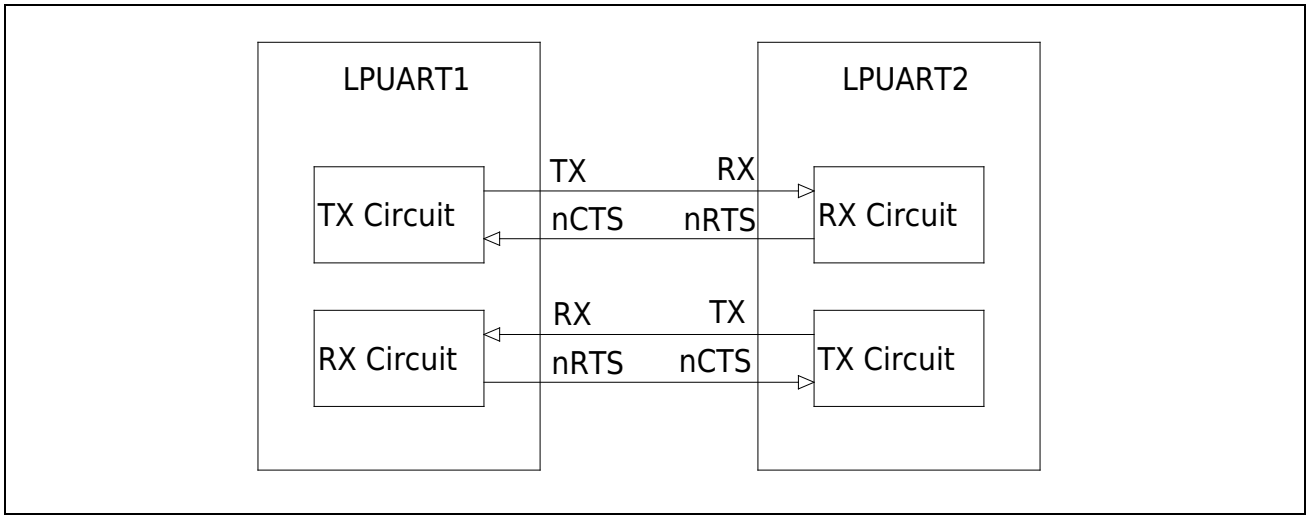


Figure 22-10 LPUART hardware flow control

- nRTS flow control

When nRTS flow control is enabled (LPUARTx_SCON.RTSEN is set to 1):

  – When the LPUART receive buffer is empty, nRTS will be asserted (connected to low level).
  – When the receive buffer is full, nRTS will become invalid (connected to a high level), indicating that the sending process will stop after the end of the current frame.

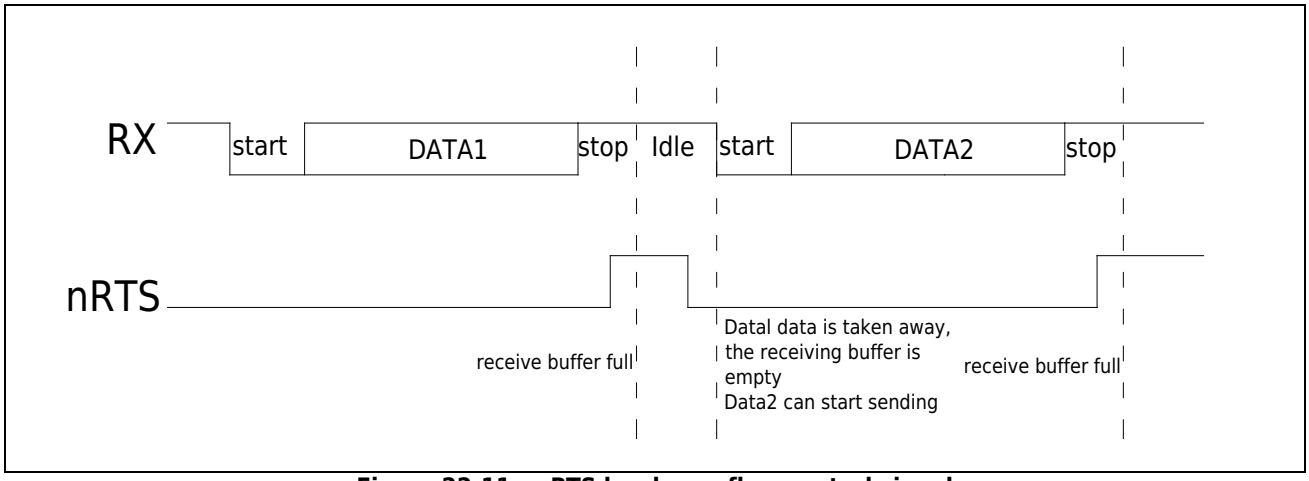


Figure 22-11 nRTS hardware flow control signal

■ CTS flow control

When nCTS flow control is enabled (LPUARTx_SCON.CTSEN is set to 1), before LPUART sends the next frame of data, first judge the high and low level of nCTS:

- If nCTS is active (connected to low level), then LPUART sends the next frame of data.
- If nCTS is invalid (connected to high level), then LPUART will suspend the transmission of the next frame after the current frame transmission is completed

When nCTS flow control is enabled, once the nCTS signal is inverted, LPUARTx_SFLAG.CTSIF will be set to 1 by hardware. If LPUARTx_SCON.CTSIE is set, an interrupt will be generated. At the same time, the high and low levels of the nCTS signal will be recorded in the LPUARTx_SFLAG.CTS flag.

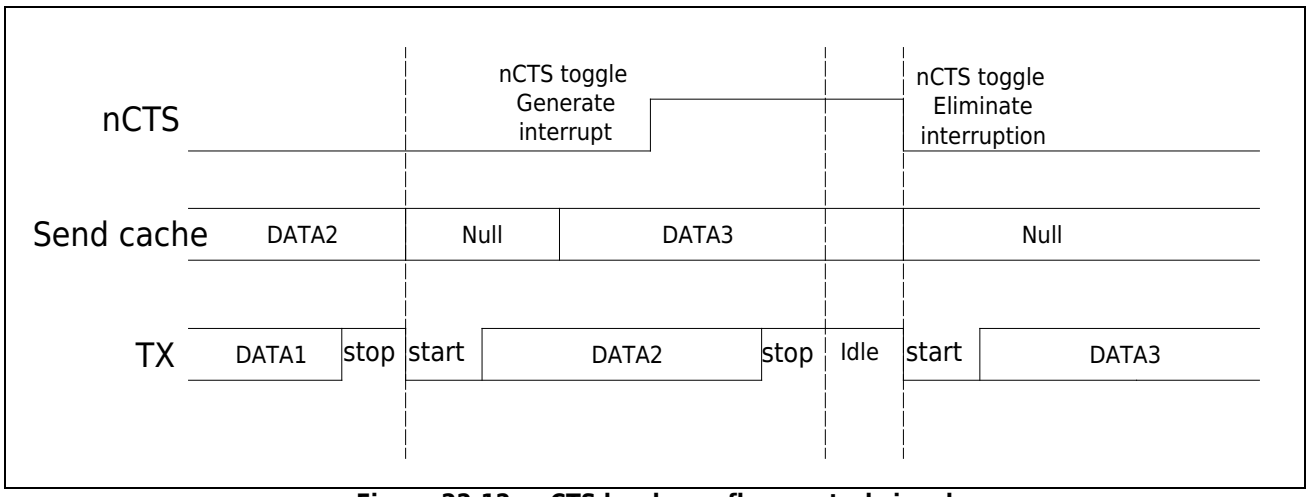


Figure 22-12 nCTS hardware flow control signal

22.3.8 Transceiver buffer

■ Receive buffer

The receiving end of the LPUART module has a frame (8/9-Bit) receiving buffer, that is, the received data frame is kept until the Stop bit of the next frame of data is received to update the data frame.

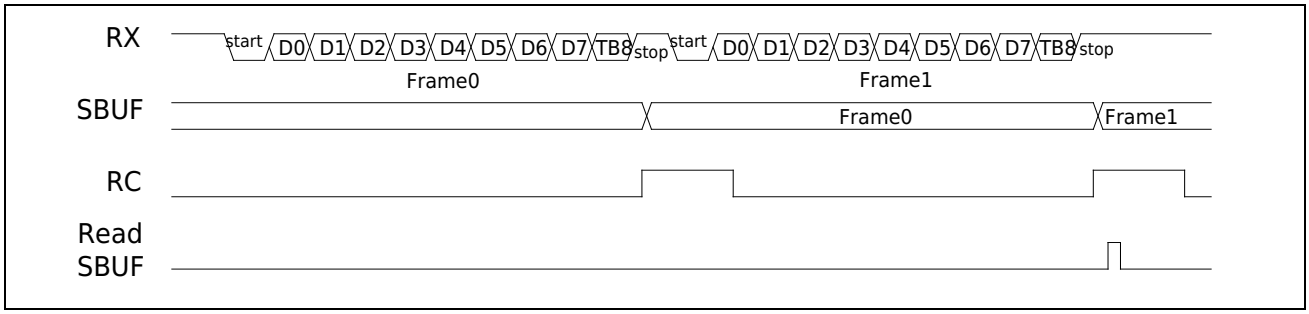


Figure 22-13 Receive buffer

■ Send cache

LPUART module has a frame (8/9-Bit) sending buffer. When the LPUART is sending the current frame, the software writes the next sending data to LPUARTx_SBUF.

When LPUARTx_ISR.TXE=0, it indicates that the current transmit buffer is full, and the next transmit data cannot be written into LPUARTx_SBUF. Otherwise the data will be discarded by hardware.

When LPUARTx_ISR.TXE=1, it indicates that the current sending buffer is empty, and LPUARTx_SBUF can write the next sending data. After the current data transmission is completed, the hardware automatically loads the data in the sending buffer into the shift register and sends it out.

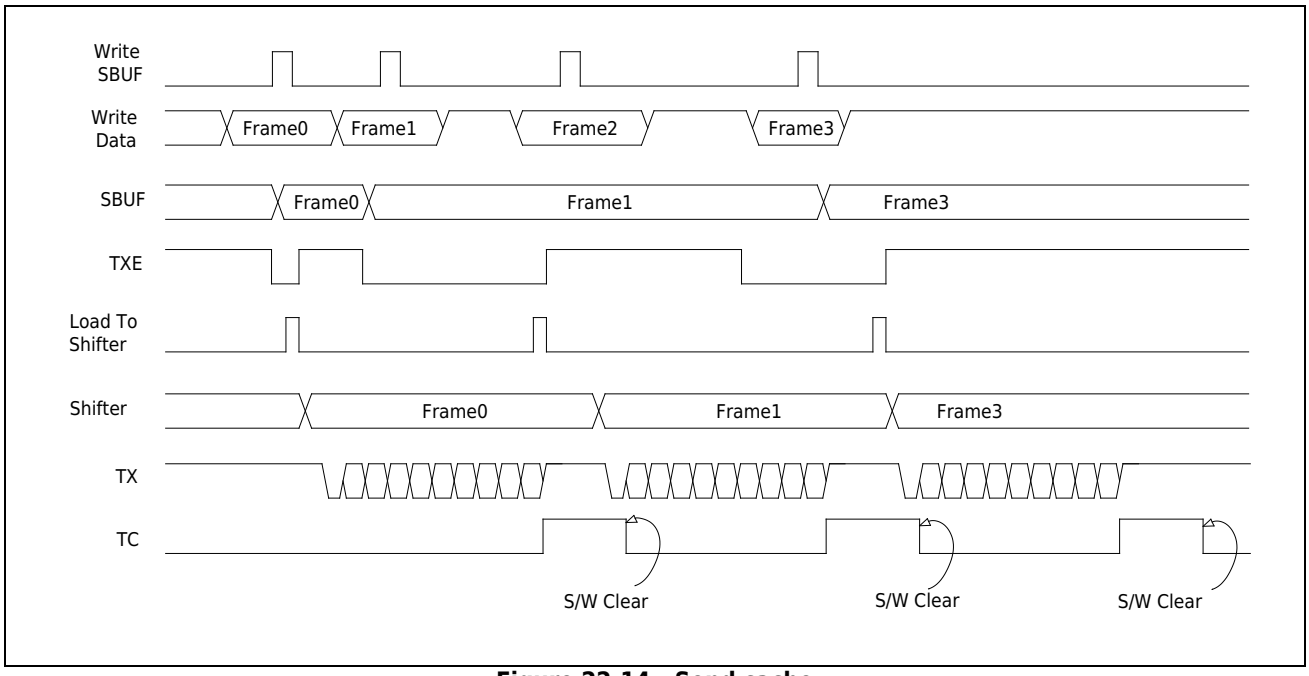


Figure 22-14 Send cache

22.4Register

LPUART0 base address: 0x4000 0200

LPUART1 base address: 0x4000 4000

Register	Offset address	Description
LPUARTx_SBUF	0x00	Data register
LPUARTx_SCON	0x04	control register
LPUARTx_SADDR	0x08	Address register
LPUARTx_SADEN	0x0C	Address mask register
LPUARTx_ISR	0x10	interrupt flag register
LPUARTx_ICR	0x14	Interrupt flag bit clear register
LPUARTx_SCNT	0x18	Baud rate register

22.4.1 Data Register (LPUARTx_SBUF)

Offset address: 0x00

Reset value: 0x0000 0000

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

Bit	Marking	Functional description
31:9	Reserved	
8	DATA[8]	In Mode0/1, read this bit is 0, write this bit is invalid; In Mode2/3, this bit represents the Bit8 data bit, which can be divided into the following two situations: (1) When the hardware parity bit is turned on, this bit is the parity bit of the received data when receiving, and the verification is performed by the hardware. If the verification error occurs, the verification error flag bit PE is set to 1; this bit is invalid when sending, The parity bit of the sent data is calculated and sent by the hardware; (2) When the hardware parity bit is turned off, this bit is received data Bit8 when receiving; this bit is sent data Bit8 when sending; Note: When the multi-machine communication mode is turned on, the parity check of the received data is automatically closed; the parity check of the sent data is still controlled by B8CONT;
7:0	DATA[7:0]	When sending data, write the sending data into this register; when receiving data, read from this register after the data is received.

22.4.2 Control register (LPUARTx_SCON)

Offset address: 0x04

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									HDSEL	FEIE	CTSIE	CTSEN	RTSEN	DMATXEN	DMARXEN
									RW	RW	RW	RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOPBIT		PEIE	SCLKSEL			OVER	TXEIE	SM		ADRDET	REN	B8CONT		TCIE	RCIE
RW		RW	RW			RW	RW	RW		RW	RW	RW		RW	RW

Reset value: 0x0000 0000

Bit	Marking	Functional description
31:23	Reserved	
22	HDSEL	Single-wire half-duplex mode is enabled; 0: Close single-line half-duplex mode; 1: open single-line half-duplex mode;
21	FEIE	Framing error interrupt enable; 0: disable interrupt; 1: enable interrupt;
20	CTSIE	CTS signal flip interrupt enable; 0: Disable interrupt; 1: Enable interrupt;
19	CTSEN	Hardware flow control signal enable bit; 0: Turn off the flow control signal; 1: Turn on the flow control signal;
18	RTSEN	
17	DMATXEN	The hardware handshake signal enable bit of TX DMAC; 0: Turn off the hardware handshake signal; 1: Turn on the hardware handshake signal;
16	DMARXEN	RX DMAC hardware handshake signal enable bit; 0: Turn off the hardware handshake signal; 1: Turn on the hardware handshake signal;
15:14	STOPBIT	stop bit length selection; 00:1-bit; 01:1.5-bit; 10:2-bit; 11: reserved; Note: Although there is no Stop Bit in Mode0, it is still necessary to keep STOPBIT[1:0] as 2'b00;
13	PEIE	Parity error interrupt enable bit; 0: parity error interrupt off; 1: parity error interrupt on; When the system clock uses a high-speed clock and the LPUART uses a low-speed clock, due to the need for synchronization of different clock domains, the hardware parity check and the data reception interrupt are set differently due to the different stop bit settings, and the parity check error interrupt will be ahead of or behind the data reception interrupt. Be careful when using parity interrupt enable. Suggestions below: Turn off the PEIE interrupt enable, and judge whether there is a parity error through the received data bit SBUF.BIT8 software after receiving the data interrupt.
12:11	SCLKSEL	Transmission clock selection bits: 00, 01: PCLK; 10: XTL; 11: RCL;
10:9	OVER	Mode0: Invalid; Mode1/3: 00: 16- sample frequency division; 01: 8 -sample frequency division; 10: 4- sample frequency division; 11: Reserved; Mode2: 00: 32 sampling frequency division; 01: 16 sampling frequency division; 10: 8 sampling frequency division; 11: reserved;
8	TXEIE	TX empty interrupt enable bit; 0: TX Buffer empty interrupt is off; 1: TX Buffer empty interrupt is on;
7:6	SM	Working mode; 00: mode0; 01: mode1; 10: mode2; 11: mode3;
5	ADRDET	Multi-machine communication address automatic identification enable bit; 0: off; 1: on;
4	REN	Mode0: 0: send; 1: receive; Mode1/2/3: 0: send; 1: receive / send;
3:2	B8CONT	Bit8 data control bit; 00: Determined by software reading and writing SBUF[8]; 01: Hardware even parity; 10: hardware odd parity; 11: reserved;
1	TCIE	Send interrupt enable bit; 0: Send interrupt is off; 1: Send interrupt is on;
0	RCIE	Receive interrupt enable bit; 0: Receive interrupt is off; 1: Receive interrupt is on;

22.4.3 Address Register (LPUARTx_SADDR)

Offset address: 0x08

Reset value: 0x0000 0000

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

Bit	Marking	Functional description
31:8	Reserved	
7:0	SADDR	Slave Device Address Register

22.4.4 Address Mask Register (LPUARTx_SADEN)

Offset address: 0x0C

Reset value: 0x0000 0000

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

Bit	Marking	Functional description
31:8	Reserved	
7:0	SADEN	Slave Device Address Mask Register

22.4.5 Flag Bit Register (LPUARTx_ISR)

Offset address: 0x10

Reset value: 0x0000 0008

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									CTS	CTSIF	PE	TXE	FE	TC	RC
									RO	RO	RO	RO	RO	RO	RO

Bit	Symbol	Description
31:7	Reserved	
6	CTS	CTS signal flag; hardware set 1; hardware clear; 0: CTS signal is low level; 1: CTS signal is high level;
5	CTSIF	CTS interrupt flag; hardware set to 1; software to clear; 0: CTS signal is not inverted; 1: CTS signal is inverted;
4	PE	Parity error flag; set to 1 by hardware; cleared by software; 0: no parity error; 1: parity error;
3	TXE	Tx Buffer empty flag; set by hardware; cleared by hardware; 0: Tx Buffer is not empty; 1: Tx Buffer is empty
2	FE	Framing error flag; 0: set by hardware; cleared by software;
1	TC	Transmit complete interrupt flag; set to 1 by hardware; cleared by software; 0: Transmit not completed; 1: Transmit complete;
0	RC	Receive complete interrupt flag; hardware set 1; software clear; 0: receive not completed; 1: receive complete;

22.4.6 Flag Clear Register (LPUARTx_ICR)

Offset address: 0x14

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										CTSIFCF	PECF	Res.	FECF	TCCF	RCCF
										R1W 0	R1W 0		R1W 0	R1W 0	R1W 0

Bit	Marking	Functional description
31:6	Reserved	
5	CTSIFCF	CTSIF flag clear bit; write 0 to clear; write 1 to be invalid;
4	PECF	PE flag clear bit; write 0 to clear; write 1 to be invalid;
3	Reserved	
2	FECF	FE flag clear bit; write 0 to clear; write 1 to be invalid;
1	TCCF	TC flag clear bit; write 0 to clear; write 1 to be invalid;
0	RCCF	RC flag clear bit; write 0 to clear; write 1 to be invalid;

22.4.7 Baud Rate Register (LPUARTx_SCNT)

Offset address: 0x18

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
SCNT															
RW															

Bit	Marking	Functional description
31:16	Reserved	
15:0	SCNT	Baud rate counter

23 Cyclic redundancy check (CRC)

23.1 Overview

A cyclic redundancy check (CRC) calculation unit takes a data stream or data block as input and generates an output number under the control of a generator polynomial. This output number is often used to verify the correctness and integrity of data transmission or storage. This module supports calculating CRC value and checking CRC value.

23.2 Main characteristics

- One implementation standard: ISO/IEC13239
- Two encoding methods: CRC-16, CRC-32
- Three write bit widths: 8bit, 16bit, 32bit
- Two working modes: CRC encoding mode, CRC check mode
- CRC-16 polynomial: $x^{16}+ x^{12}+ x^5+ 1$
- CRC-32 polynomial:$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$

23.3 Functional description

23.3.1 Operating mode

This module supports two working modes: CRC encoding mode and CRC checking mode.

The CRC encoding mode is to input a certain amount of raw data to the CRC module and obtain the output value (CRC_RESULT) generated by the CRC module. The CRC check mode is to point to the CRC module to input a certain amount of original data + CRC check value, and verify whether the original data matches the CRC check value (CRC_CR.FLAG).

23.3.2 Encoding

This module supports two encoding methods CRC-16 and CRC-32, and the calculation results are 16 bits and 32 bits respectively. Configure the encoding method to be used through CRC_CR.CR.

CRC-16 polynomial:$x^{16}+ x^{12}+ x^5+ 1$.

CRC-32 polynomial:$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$.

23.3.3 Write bit width

This module supports three write bit widths: 8bit, 16bit, 32bit. The writing of different bit widths needs to comply with the principle of " consistent bit width, low first and then high ", that is, " every time data is written, it must be written into a register equal to the effective data bit width of this time, and the lower bit Data is written before higher bit data ".

The following shows how the same sequence of data is written using three bit widths, and the output results are the same.

- 8bit bit width write: 0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77
- 16bit bit width write: 0x1100, 0x3322, 0x5544, 0x7766
- 32bit bit width write: 0x33221100, 0x77665544

23.4Programming example

23.4.1 CRC-16 encoding mode

Step 1： Write 0x00 to CRC_CR.CR to select CRC-16.

Step 2： Write 0xFFFF to CRC_RESULT to initialize CRC calculation.

Step 3： Write the original data to be encoded into the CRC_DATA register in sequence, and the write bit width can be selected from 8bit, 16bit, and 32bit.

Step 4： Read CRC_RESULT[15:0] to get the CRC value.

23.4.2 CRC-16 check mode

Step 1： Write 0x00 to CRC_CR.CR to select CRC-16.

Step 2： Write 0xFFFF to CRC_RESULT to initialize CRC calculation.

Step 3： Write the encoded data sequence into the CRC_DATA register in sequence, and the write bit width can be selected from 8bit, 16bit, and 32bit.

Step 4： Determine whether the encoded data sequence has been tampered with according to the value of CRC_CR.FLAG.

23.4.3 CRC-32 encoding mode

Step 5： Write 0x01 to CRC_CR.CR to select CRC-32.

Step 6： Write 0xFFFFFFFF to CRC_RESULT to initialize CRC calculation.

Step 7： Write the original data to be encoded into the CRC_DATA register in sequence, and the write bit width can be selected from 8bit, 16bit, and 32bit.

Step 8： Read CRC_RESULT[31:0] to get the CRC value.

23.4.4 CRC-32 check mode

Step 5： Write 0x01 to CRC_CR.CR to select CRC-32.

Step 6： Write 0xFFFFFFFF to CRC_RESULT to initialize CRC calculation.

Step 7： Write the encoded data sequence into the CRC_DATA register in sequence, and the write bit width can be selected from 8bit, 16bit, and 32bit.

Step 8： Determine whether the encoded data sequence has been tampered with according to the value of CRC_CR.FLAG.

23.5Register description

23.5.1 Register list

Base address: 0x4002 0900

Register	Offset address	Description
CRC_CR	0x00	CRC Control Register
CRC_RESULT	0x04	CRC Result Register
CRC_DATA	0x80	CRC data register

23.5.2 Control register (CRC _ CR)

Offset address: 0x00

Reset value: 0x0000 0001

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														FLAG	CR
R														RO	RW

Bit	Symbol	Functional description
31:2	Reserved	
1	FLAG	CRC check result 0: The current CRC check error 1: The current CRC check is correct
0	CR	CRC encoding method selection 0: CRC-16 encoding 1: CRC-32 encoding

23.5.3 Results register (CRC_RESULT)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESULT[31:16]															
RW															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESULT[15:0]															
RW															

Bit	Symbol	Description
31:0	RESULT	CRC calculation result Read RESULT[15:0] to get the calculation result of CRC-16 Read RESULT[31:0] to get the calculation result of CRC-32 Write 0xFFFF to RESULT[15:0] to initialize CRC-16 calculation Write 0xFFFFFFFF to RESULT[31:0] to initialize CRC-32 calculation

23.5.4 Data register (CRC_DATA)

Offset address: 0x80

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA[31:16]															
WO															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA[15:0]															
WO															

Bit	Symbol	Functional description
31:0	DATA	This register is used to write data that needs to be calculated, and supports 3 write bit widths 8bit writing method: * ((uint8_t *)0x40020980) = 0xXX 16bit writing method: * ((uint16_t *)0x40020980) = 0xXXXX 32bit writing method: * ((uint32_t *)0x40020980) = 0xXXXXXXXX

24 True random number generator (TRNG)

24.1 Overview

The true random number module generates 64 -bit true random numbers.

24.2 Functional block diagram

The following shows the data flow of the TRNG module:

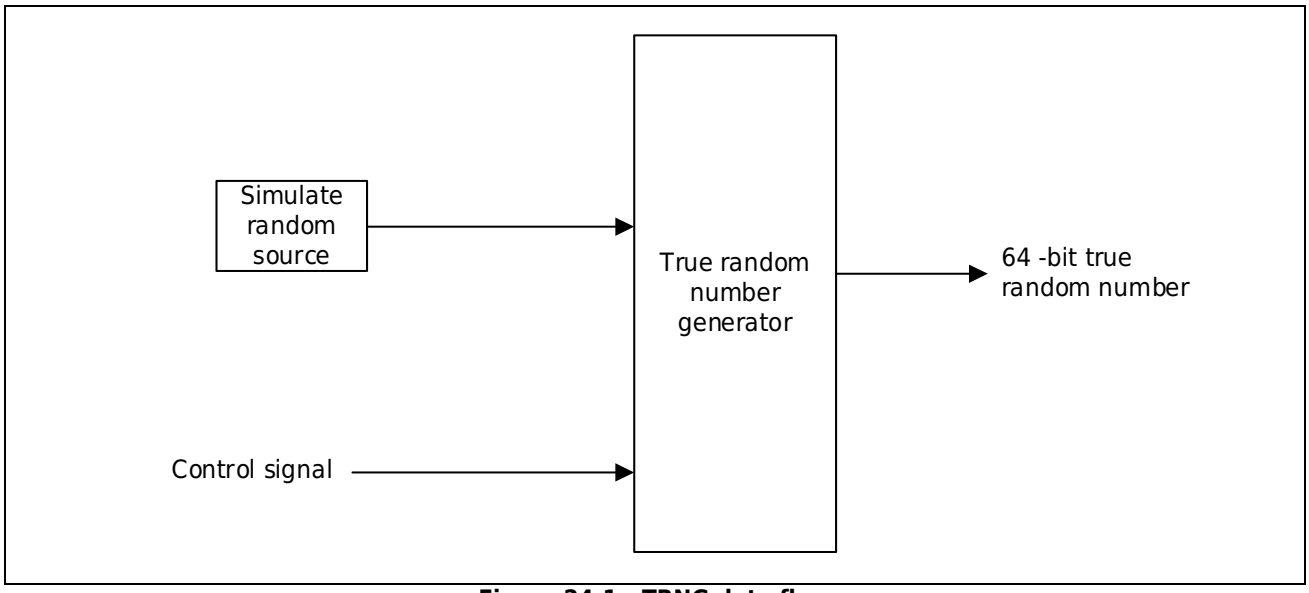


Figure 24-1 TRNG data flow

24.3 Functional description

This module uses an internal analog random source, which can generate 64bits true random numbers every time it is started. In addition, software configuration can be performed on the way of true random number generation. For details, please refer to the register description chapter. The generated 64 -bit true random numbers are stored in DATA0 and DATA1 registers respectively.

24.4Register

Base address: 0x4000 4C00

Register	Offset address	Description
TRNG_CR	0x00	Control register
TRNG_MODE	0x04	Mode register
TRNG_DATA0	0x0C	Data register 0
TRNG_DATA1	0x10	Data register 1

24.4.1 Control Register (TRNG_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														TRNG_RUN	TRNGcir_EN
														RW	RW

Bit	Marking	Functional description
31:2	Reserved	
1	TRNG_RUN	The software writes "1" and starts to generate a new 64bits random number; after the operation is completed, the hardware is cleared; 0: The random number generation is completed; 1: Write 1 to start random number generation, read 1 to indicate that random number is being generated;
0	TRNGcir_EN	Random source circuit enable bit: 0: turn off the random source; 1: Turn on the random source;

24.4.2 Mode Register (TRNG_MODE)

Offset address: 0x04

Reset value: 0x0000 0010

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											TRNG_CNT			TRNG_FDBK	TRNG_LOAD
											RW			RW	RW

Bit	Marking	Description
31:5	Reserved	
4:2	TRNG_CNT	Feedback shift times of 64bits TRNG 3'b000: Shift 0 times (that is, output the sampling value of the random source) 3'b001: shifted 8 times 3'b010: shifted 16 times 3'b011: shifted 32 times 3'b100: shifted 64 times 3'b101: shifted 128 times 3'b110: shifted 256 times 3'b111:Reserved
1	TRNG_FDBK	During the shift operation, whether the feedback signal of the 64bits TRNG is XORed with the random source 0: Do not perform XOR operation; 1: XOR operation;
0	TRNG_LOAD	When generating a new random number, whether the 64bits TRNG obtains a new initial value from the random source 0: Do not load new initial values (generate pseudo-random numbers); 1: load a new initial value (generating a true random number);

24.4.3 Data Register 0 (TRNG_DATA0)

Offset address: 0x0C

Reset value: ---

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA0[31:16]															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA0[15:0]															
RO															

Bit	Marking	Functional description
31:0	DATA0	The software will get the low 32- bit random number when reading this register

24.4.4 Data Register 1 (TRNG_DATA1)

Offset address: 0x10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA1[31:16]															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA1[15:0]															
RO															

Reset value: ---

Bit	Marking	Functional description
31:0	DATA1	32- bit random number when reading this register

24.5Basic operation of the software

24.5.1 The operation process of generating 64bits true random number (the first time after power-on)

To generate a 64bits true random number for the first time after power-on, the following operations are required:

Step1: Turn on the random source circuit: write "1" to Bit0 (TRNG_CR.TRNGcir_En) of the true random number control register, start the random source circuit, and start outputting serial true random numbers.

Step2: Choose to reload the initial value: Set Bit0 (TRNG_MODE.TRNG_LOAD) of the true random number mode register to "1", so that the initial value of the newly generated true random number is obtained from a random source.

Step3: Select the direct feedback method of PRNG64: set Bit1 (TRNG_MODE.TRNG_FDBK) of the true random number mode register to "1", XOR the feedback signal with the random source and input it into the PRNG.

Step4: Select the number of shifts of PRNG64: set Bit4 - Bit2 (TRNG_MODE.TRNG_CNT) of the true random number mode register to "110", and select 256 shifts.

Step5: Generating a true random number: the software writes "1" into Bit1 of the true random number control register (TRNG_CR.TRNG_RUN), and the hardware operates according to the true random number generation configuration. After the operation is completed, the hardware automatically clears Bit1 to "0 ".

Step6: Choose not to reload the initial value: Set Bit0 (TRNGModeReg.TRNG_Load) of the true random number mode register to "0".

Step7: Select the direct feedback method of PRNG64: set Bit1 (TRNG_MODE.TRNG_FDBK) of the true random number mode register to "0", and directly input the feedback signal into the PRNG.

Step8: Select the number of shifts for PRNG64: set Bit4 - Bit2 (TRNG_MODE.TRNG_CNT) of the true random number mode register to "100", and select 64 shifts.

Step9: Generating a true random number: the software writes "1" into Bit1 of the true random number control register (TRNG_CR.TRNG_RUN), and the hardware operates according to the true random number generation configuration. After the operation is completed, the hardware automatically clears Bit1 to "0 ".

Step10: Read the true random number: After the software inquires that the Bit1 (TRNG_CR.TRNG_RUN) of the true random number control register has changed to "0", it

reads the true random number data register 0 (TRNG_DATA0) and the true random number data register 1 (TRNG_DATA1) to get 64Bits true random number.

Step11: After completing the generation of true random numbers, it is recommended to turn off the random source circuit to save power consumption: write "0" to Bit0 (TRNG_CR.TRNGcir_En) of the true random number control register to turn off the random source circuit.

24.5.2 Generating 64bits true random number (not generated for the first time after power-on)

A 64bits true random number for the first time without power-on, the following operations are required:

Step1: Turn on the random source circuit: write "1" to Bit0 (TRNG_CR.TRNGcir_En) of the true random number control register, start the random source circuit, and start outputting serial true random numbers.

Step2: Choose not to reload the initial value: Set Bit0 (TRNG_MODE.TRNG_LOAD) of the true random number mode register to "0".

Step3: Select the direct feedback method of PRNG64: set Bit1 (TRNG_MODE.TRNG_FDBK) of the true random number mode register to "1", and input the feedback signal into the PRNG after XORing with the random source.

Step4: Select the number of shifts for PRNG64: set Bit4 - Bit2 (TRNG_MODE.TRNG_CNT) of the true random number mode register to "110", and select 256 shifts.

Step5: Generating a true random number: the software writes "1" into Bit1 of the true random number control register (TRNG_CR.TRNG_RUN), and the hardware operates according to the true random number generation configuration. After the operation is completed, the hardware automatically clears Bit1 to "0 ".

Step6: Select the direct feedback method of PRNG64: set Bit1 (TRNG_MODE.TRNG_FDBK) of the true random number mode register to "0", and directly input the feedback signal into the PRNG.

Step7: Select the number of shifts for PRNG64: set Bit4 - Bit2 (TRNG_MODE.TRNG_CNT) of the true random number mode register to "100", and select 64 shifts.

Step8: Read the true random number: After the software inquires that the Bit1 (TRNG_CR.TRNG_RUN) of the true random number control register has changed to "0", it reads the true random number data register 0 (TRNG_Data0) and the true random number data register 1 (TRNG_Data1), get 64Bits true random number.

If it is necessary to continue to generate new true random numbers, then go back to Step2 until the requirements are met.

Step9: After completing the generation of true random numbers, it is recommended to turn off the random source circuit to save power consumption: write "0" to Bit0 (TRNG_CR.TRNGcir_En) of the true random number control register to turn off the random source circuit.

25 Advanced Encryption Standard Module (AES)

25.1 Function definition

25.1.1 AES algorithm

AES (The Advanced Encryption Standard) is a new data encryption standard officially announced by the National Institute of Standards and Technology (NIST) on October 2, 2000.

AES is fixed at 128 bits, while the key length supports 128, 192 and 256 bits. For encryption, the input is a plaintext block and a key, and the output is a ciphertext block; for decryption, the input is a ciphertext block and a key, and the output is a plaintext block. This process is shown in Figure 25-1:

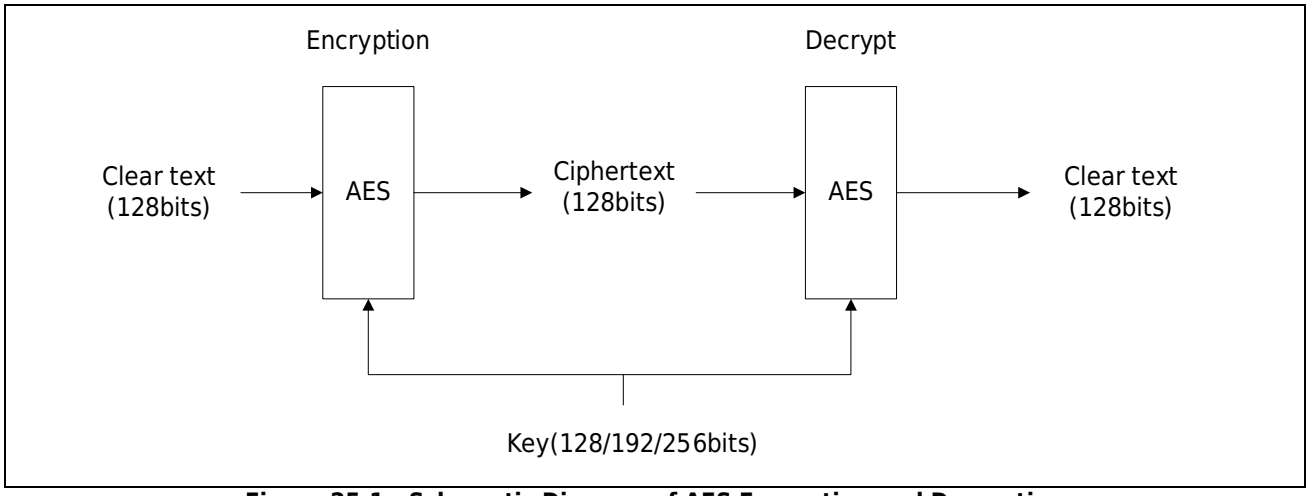


Figure 25-1 Schematic Diagram of AES Encryption and Decryption

The basic unit processed by the AES algorithm is the byte. The 128 -bit information is divided into 16 bytes, which are copied into a 4×4 matrix in order, which is called the state (state). All transformations of AES are transformations based on the state matrix, the intermediate results of the calculation are stored on this matrix.

AES is a key iterative block cipher that involves round-robin repetition of state. AES consists of four operations: SubBytes, ShiftRows, MixColumns, AddRoundKey. Among them, SubBytes includes finding the modular inverse of each byte in GF(28) and an affine transformation; ShiftRows is a byte transposition, which cyclically shifts the rows in the state according to different offsets; MixColumns linearly transforms each column of the state; AddRoundKey performs a bit-by-bit XOR operation between each byte in the state and the round key. The encryption process of AES is shown in Figure 25-2:

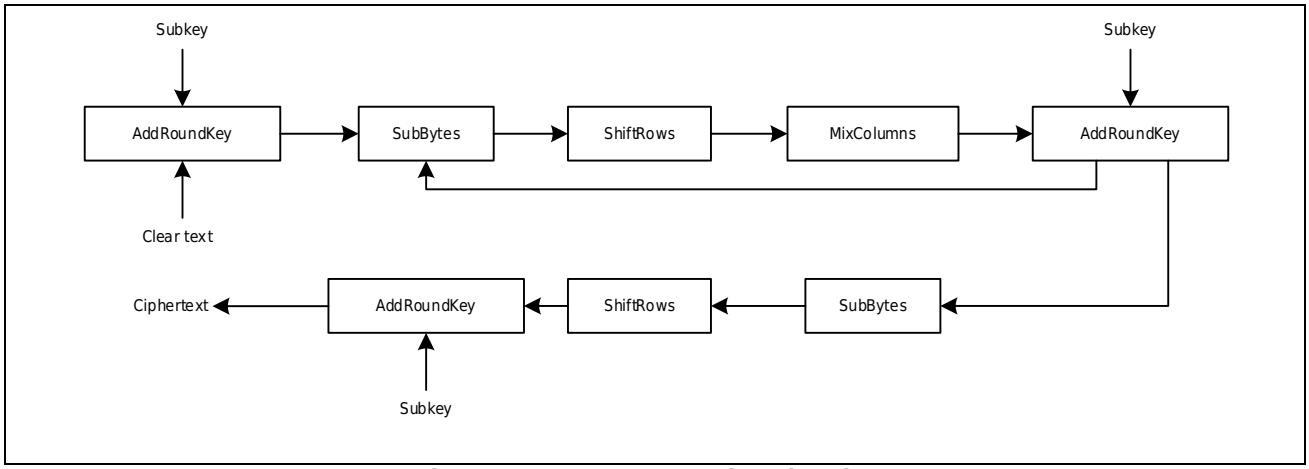


Figure 25-2 AES Encryption Flowchart

The subkey used in the figure needs to be expanded from the initial key, and the key expansion process and encryption process are carried out synchronously.

Since the plaintext is fixed at 128 bits, the number of rounds the encryption process runs depends on the length of the key. For example, when the key is 128 bits, the number of running rounds is 10; when the key is 192 bits, the number of running rounds is 12; when the key is 256 bits, the number of running rounds is 14 rounds. Except for the lack of MixColumns transformation in the last round, the rest of the rounds perform a complete round transformation operation.

The decryption process is different from the encryption process. First, the expansion of all keys must be completed, and the decryption process is used backwards from the last round of expansion; then the four operations of round transformation become the corresponding inverse operations: InvSubBytes, InvShiftRows, InvMixColumns, AddRoundKey. The modular inverse operation in InvSubBytes is still maintained, but the affine transformation is changed to inverse transformation; InvShiftRows and InvMixColumns become corresponding inverse transformations; AddRoundKey remains unchanged.

The calling order of the four operations in the round transformation of the direct decryption process is: InvShiftRows, InvSubBytes, AddRoundKey, InvMixColumns, which is inconsistent with the calling order of the encryption process, but the key used is consistent with the encryption process; the round transformation of the equivalent decryption process is for four The calling order of the operations is: InvSubBytes, InvShiftRows, InvMixColumns, AddRoundKey, which is exactly the same as the calling order of the encryption process, except that the subkeys of each round need to perform InvMixColumns operations.

For the detailed algorithm expression, please refer to the standard " FIPS PUB 197 ".

25.1.2 AES module function description

- Execute the encryption process and decryption process of the AES algorithm standard, and its execution results fully comply with the description of the algorithm principle in " FIPS PUB 197 ";
- 128, 192 and 256 -bit keys are supported.

25.2Module register description

AES base address 0x40021400

Table 25-1 Register List

Register	Offset address	Description
AES_CR	0x00 或 0x40	control register
AES_Data	0x10~0x1C	Data register
AES_Key	0x20~0x3C	key register

25.2.1 Control Register (AES_CR)

Offset address: 0x00 or 0x40

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											Keysize		Reserved	Mode	Start
											RW			RW	RW

Bit	Symbol	Description
31:5	Reserved	Reserved bit, read as 0
4:3	Keysize	2'b00: key length is 128 bits 2'b01: key length is 192 bits 2'b10: key length is 256 bits 2'b11: key length is 128 bits
2	Reserved	Reserved bit, read as 0
1	Mode	0: encryption operation 1: Decryption operation
0	Start	0: The operation of this module is completed or has not been started 1: Start this module for calculation

Note：

1. The AES_CR.Start bit is: after the software writes 1 to this bit, the module will start running. After this operation, the hardware of this module will automatically clear this bit to 0. If the software checks that this bit is 0, it means that this operation is completed.
2. The write operation to this register can only be performed when the module is not in the operation state (that is, when AES_CR.Start = 0), otherwise the hardware will automatically ignore the write operation. Read operations are not subject to this restriction.

25.2.2 Data register (AES_Data)

Offset address: 0x10~0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Data[31:16]															
RW															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data[15:0]															
RW															

Bit	Symbol	Description
31:0	Data	Store 128- bit plaintext / ciphertext of AES algorithm

Note:

1. The data register consists of four 32-bit registers consisting of 128-bit data, which are used to store the plaintext to be encrypted or the ciphertext to be decrypted before the module operation, and store the encrypted ciphertext or decrypted plaintext after the operation is completed.

Encryption operation		Decryption operation	
Before operation	After operation	Before operation	After operation
128-bit plaintext	128-bit ciphertext	128-bit ciphertext	128-bit plaintext

Four 32-bit registers are connected together to form a 128-bit data, and the four registers need to be operated separately during read and write operations. The operation sequence corresponding to the data register is as follows:

Data example: 0xFFEEDDCCBBAA99887766554433221100

Offset address	Register name	Fill in data
0x10	AES_Data0	0x33221100
0x14	AES_Data1	0x77665544
0x18	AES_Data2	0xBBAA9988
0x1C	AES_Data3	0xFFEEDDCC

2. Writing to this register can only be done when the module is not in operation state (that is, when AES_CR.Start = 0), otherwise the hardware will automatically ignore the writing operation to this register.
3. The reading of this register can only be carried out when the module is not in the operation state (that is, when AES_CR.Start = 0), otherwise the reading of this register will get all 0s.

25.2.3 Key register (AES_Key)

Offset address: 0x20~0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Key[31:16]															
RW															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Key[15:0]															
RW															

Bit	Symbol	Description
31:0	Key	Store 128, 192, 256 -bit keys of AES algorithm

1. The key register consists of eight 32- bit registers, which store the input initial key. The write operation needs to operate on 8 32 -bit registers respectively. The corresponding sequence of operations is as follows:

128 -bit key example: 0x0F0E0D0C_0B0A0908_07060504_03020100

Offset address	Register name	Fill in data
0x20	AES_Key0	0x03020100
0x24	AES_Key1	0x07060504
0x28	AES_Key2	0x0B0A0908
0x2C	AES_Key3	0x0F0E0D0C

192 -bit key example: 0x17161514_13121110_0F0E0D0C_0B0A0908_07060504_03020100

Offset address	Register name	Fill in data
0x20	AES_Key0	0x03020100
0x24	AES_Key1	0x07060504
0x28	AES_Key2	0x0B0A0908
0x2C	AES_Key3	0x0F0E0D0C
0x30	AES_Key4	0x13121110
0x34	AES_Key5	0x17161514

256 -bit key example: 0x1F1E1D1C_1B1A1918_17161514_13121110_0F0E0D0C_0B0A0908_07060504_03020100

Offset address	Register name	Fill in data
0x20	AES_Key0	0x03020100
0x24	AES_Key1	0x07060504
0x28	AES_Key2	0x0B0A0908
0x2C	AES_Key3	0x0F0E0D0C
0x30	AES_Key4	0x13121110
0x34	AES_Key5	0x17161514
0x38	AES_Key6	0x1B1A1918
0x3C	AES_Key7	0x1F1E1D1C

2. Writing to this register can only be done when the module is not in operation state (that is, when AES_CR.Start = 0), otherwise the hardware will automatically ignore the writing operation to this register.
3. The reading of this register can only be carried out when the module is not in the operation state (that is, when AES_CR.Start = 0), otherwise the reading of this register will get all 0s.

25.3Exception mechanism

- Only 32 -bit access is supported, and access to other bit widths will cause system exceptions and enter hardware exception interrupts.
- Accessing the offset address of the AES module greater than or equal to the address of 0x80 will cause a system exception and enter a hardware exception interrupt.

25.4Operating instructions for this module

This module has two functions: encryption and decryption. The operations of the two functions have some common features. The following will first introduce the common points, and then introduce the standard operating procedures of each function.

25.4.1 IP operations

1. In the process of AES encryption and decryption, the data register will change. If the operated data of the next operation is the result of this operation, then there is no need to rewrite the data.
2. Support 128, 192 and 256- bit keys, 128- bit key write offset address 0x20~0x2C, 192 -bit key write offset address 0x20~0x34, 256 -bit key write offset address 0x20~0x3C.
3. The method of judging the end of the operation of the module: read AES_CR.Start continuously, if its value becomes 0, it means the end of the operation.

25.4.2 Encryption operation process

Step 1：128 -bit data to be encrypted into the data register (AES_DATA).

Step 2：Write the encryption key into the key register (AES_KEY).

Step 3：Set AES_CR.Keysize according to key length

Step 4：Set AES_CR.Mode to 0 to enable encryption mode.

Step 5：Write 1 to AES_CR.Start in the control register to start the module to operate.
Step 3, Step 4 and Step 5 can be performed simultaneously.

Step 6：Wait for the value of AES_CR.Start to return to 0, and the module operation ends.

Step 7：Read the data register (AES_DATA) to obtain 128 -bit ciphertext.

25.4.3 Decryption operation process

Step 1：128 -bit data to be decrypted into the data register (AES_DATA).

Step 2：Write the decryption key into the key register (AES_KEY).

Step 3：Set AES_CR.Keysize according to key length

Step 4：Set AES_CR.Mode to 1 to enable decryption mode.

Step 5：Write 1 to AES_CR.Start in the control register to start the module to operate.

Step 3, Step 4 and Step 5 can be performed simultaneously.

Step 6：Wait for the value of AES_CR.Start to return to 0, and the module operation ends.

Step 7：Read the data register (AES_DATA) to get 128 -bit plaintext.

25.4.4 Data example

128 -bit plaintext: 0xFFEEDDCCBBAA99887766554433221100

128 -bit key: 0x0F0E0D0C0B0A09080706050403020100

128 -bit ciphertext: 0x5AC5B47080B7CDD830047B6AD8E0C469

Table 25-2 128 -bit operational register example

Before Encryption			
Register	value (key)	Register	value (clear text)
Key0	0x03020100	Data0	0x33221100
Key1	0x07060504	Data1	0x77665544
Key2	0x0B0A0908	Data2	0xBBAA9988
Key3	0x0F0E0D0C	Data3	0xFFEEDDCC
After encryption			
Register	value (key)	Register	value (ciphertext)
Key0	0x03020100	Data0	0xD8E0C469
Key1	0x07060504	Data1	0x30047B6A
Key2	0x0B0A0908	Data2	0x80B7CDD8
Key3	0x0F0E0D0C	Data3	0x5AC5B470

128 -bit plaintext: 0xFFEEDDCCBBAA99887766554433221100

192 -bit key: 0x17161514131211100F0E0D0C0B0A09080706050403020100

128 -bit ciphertext: 0x5AC5B47080B7CDD830047B6AD8E0C469

Table 25-3 192 -bit operational register example

Before Encryption			
Register	value (key)	Register	value (clear text)
Key0	0x03020100	Data0	0x33221100
Key1	0x07060504	Data1	0x77665544
Key2	0x0B0A0908	Data2	0xBBAA9988
Key3	0x0F0E0D0C	Data3	0xFFEEDDCC
Key4	0x13121110		
Key5	0x17161514		
After encryption			
Register	value (key)	Register	value (ciphertext)
Key0	0x03020100	Data0	0xA47CA9DD
Key1	0x07060504	Data1	0xE0DF4C86
Key2	0x0B0A0908	Data2	0xA070AF6E
Key3	0x0F0E0D0C	Data3	0x91710DEC
Key4	0x13121110		
Key5	0x17161514		

128 -bit plaintext: 0xFFEEDDCCBBAA99887766554433221100

256 -bit key:

0x1F1E1D1C1B1A19181716151413121110F0E0D0C0B0A0908070605040302010 0

128 -bit ciphertext: 0x5AC5B47080B7CDD830047B6AD8E0C469

Table 25-4 192 -bit operational register example

Before Encryption			
Register	value (key)	Register	value (clear text)
Key0	0x03020100	Data0	0x33221100
Key1	0x07060504	Data1	0x77665544
Key2	0x0B0A0908	Data2	0xBBAA9988
Key3	0x0F0E0D0C	Data3	0xFFEEDDCC
Key4	0x13121110		
Key5	0x17161514		
Key6	0x1B1A1918		
Key7	0x1F1E1D1C		
After encryption			
Register	value (key)	Register	value (ciphertext)
Key0	0x03020100	Data0	0xCAB7A28E
Key1	0x07060504	Data1	0xBF456751
Key2	0x0B0A0908	Data2	0x9049FCAE
Key3	0x0F0E0D0C	Data3	0x8960494B
Key4	0x13121110		
Key5	0x17161514		
Key6	0x1B1A1918		
Key7	0x1F1E1D1C		

25.5Runtime instructions

The time required by this module from starting an operation (AES_CR.Start writes 1) to the end of the operation (AES_CR.Start returns to 0) is shown in Table 25-5:

Table 25-5 AES encryption and decryption running time

	128 -bit key	192 -bit key	256 -bit key
Encryption	220 cycles	260 cycles	300 cycles
Decrypt	290 cycles	332 cycles	398 cycles

26 Liquid Crystal Controller (LCD)

26.1 Introduction to LCD

The LCD controller is a digital controller/driver for monochrome passive liquid crystal displays (LCD) with up to 8 common terminals (COM) and 48 segment terminals (SEG) to drive 208 (4x52) Or 384 (8x48) LCD picture elements. The exact number of terminals depends on the device pinout as stated in the data sheet.

LCDs are made up of segments (either pixels or complete symbols) that can be either on or off. Each segment contains a layer of liquid crystal molecules aligned between two electrodes. When a voltage higher than the threshold voltage is applied to the liquid crystal, the corresponding segment becomes visible. The segment voltage must be AC to avoid electrophoretic effects in the liquid crystal (which will affect the display). Afterwards, waveforms must be generated across the segment to avoid DC.

Glossary

Liquid Crystal (LCD): A passive display panel with terminals leading directly to the segment.

Common (COM): Electrical connection terminals connected to multiple sections.

Bias (BIAS): The voltage level used when driving the LCD, defined as 1/(the number of voltage levels that drive the LCD display – 1).

Segment (SEG): The smallest visual unit (the smallest constituent element, line or point on an LCD display).

Duty cycle (DUTY): A number defined as 1/(number of common terminals on the LCD display).

Frame: One period of the waveform written to the segment.

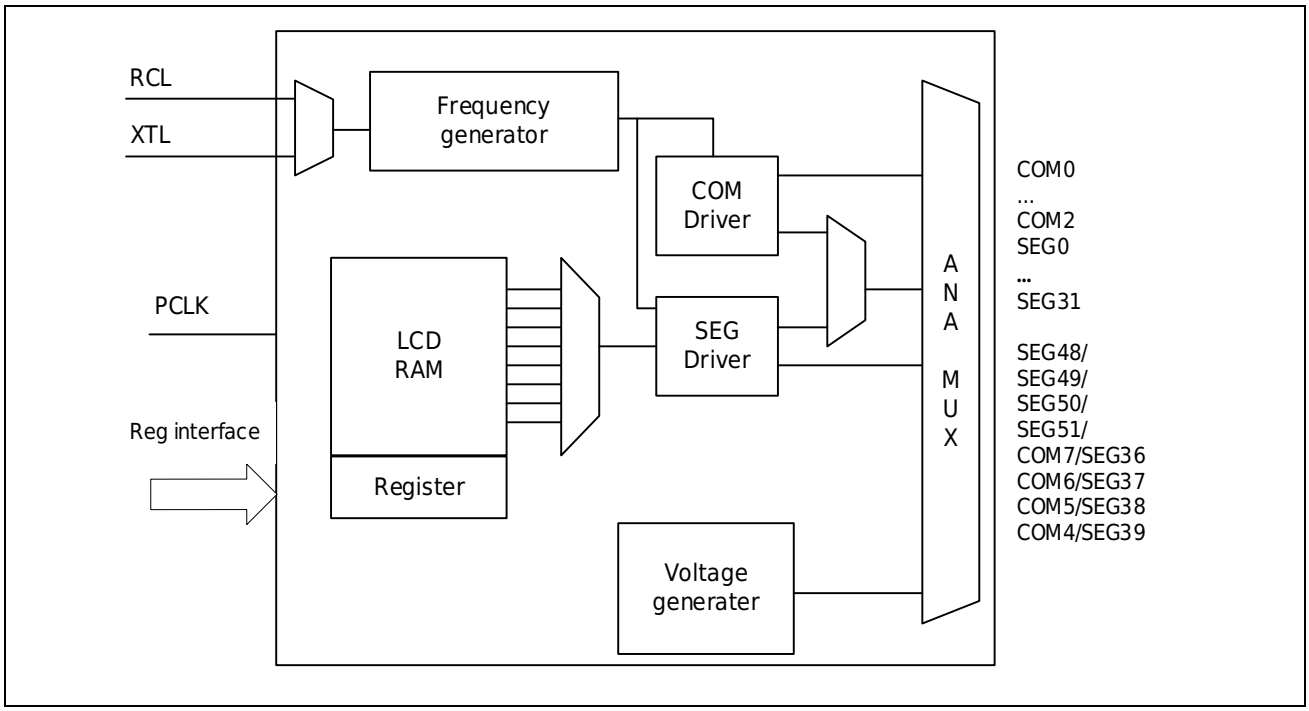
Frame Rate: Frames per second, ie the number of times an LCD segment is excited per second.

26.2 LCD main characteristics

- Highly flexible frame rate control.
- Support static, 1/2, 1/3, 1/4, 1/6 and 1/8 duty cycle.
- Support 1/2, 1/3 offset.
- LCD data RAM with up to 16 registers.
- The contrast of LCD can be configured through software.
- Three drive waveform generation methods.
  - Internal resistor divider, external resistor divider, external capacitor divider method
  - The power consumption of the internal resistor divider method can be configured through software to match the capacitive charge required by the LCD panel
- Support low power mode: LCD controller can display in Active, Sleep, DeepSleep mode.

- Configurable frame interrupt.
- Support LCD blinking function and configurable multiple blinking frequencies.
- The unused LCD segments and common pins can be configured as digital or analog functions.

26.3LCD Block Diagram



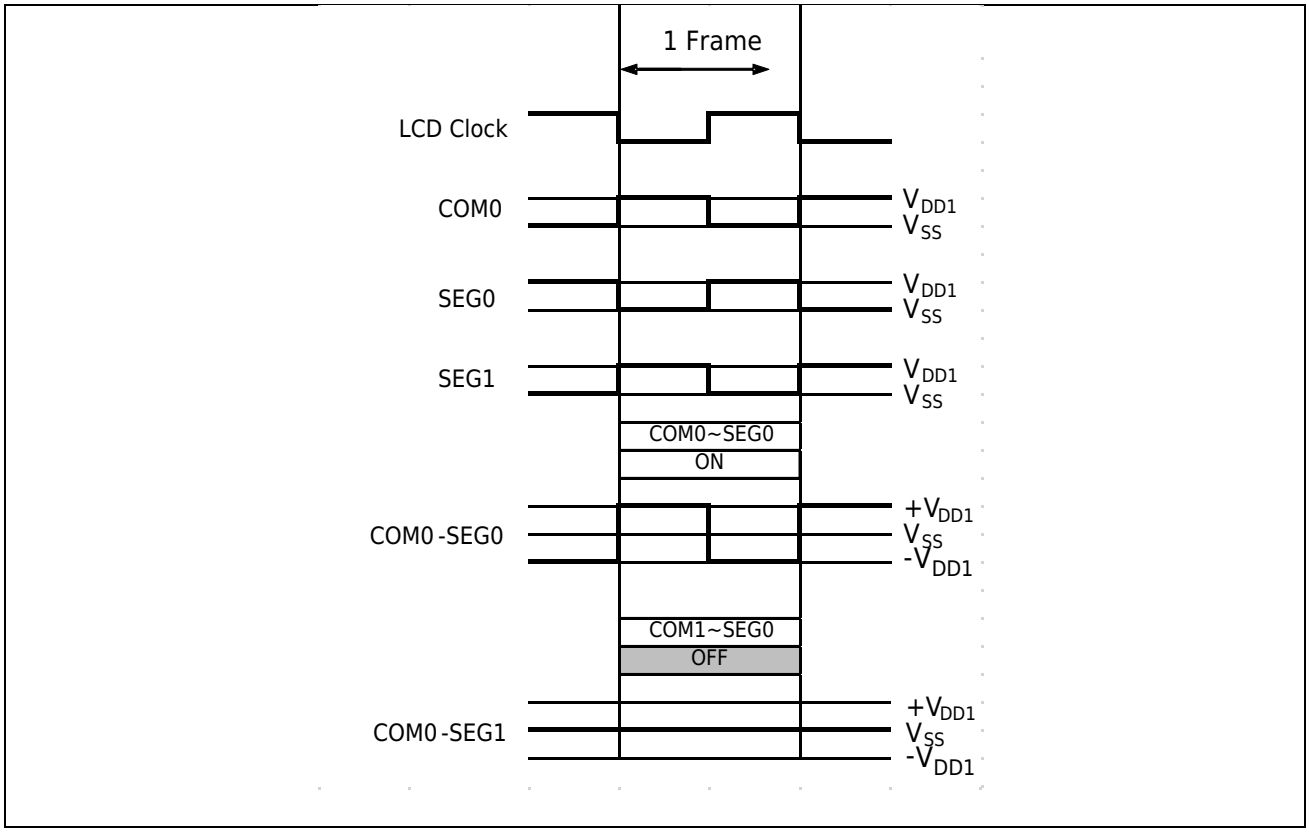
26.4LCD drive waveform

LCD supports 5 driving waveforms with duty cycle (Duty): Static, 1/2, 1/3, 1/4, 1/6 and 1/8, which are set by LCD_CR0.Duty. LCD supports 2 bias (Bias) driving waveforms: 1/2, 1/3, set by LCD_CR0.Bias. The suggested combinations are shown in the table below:

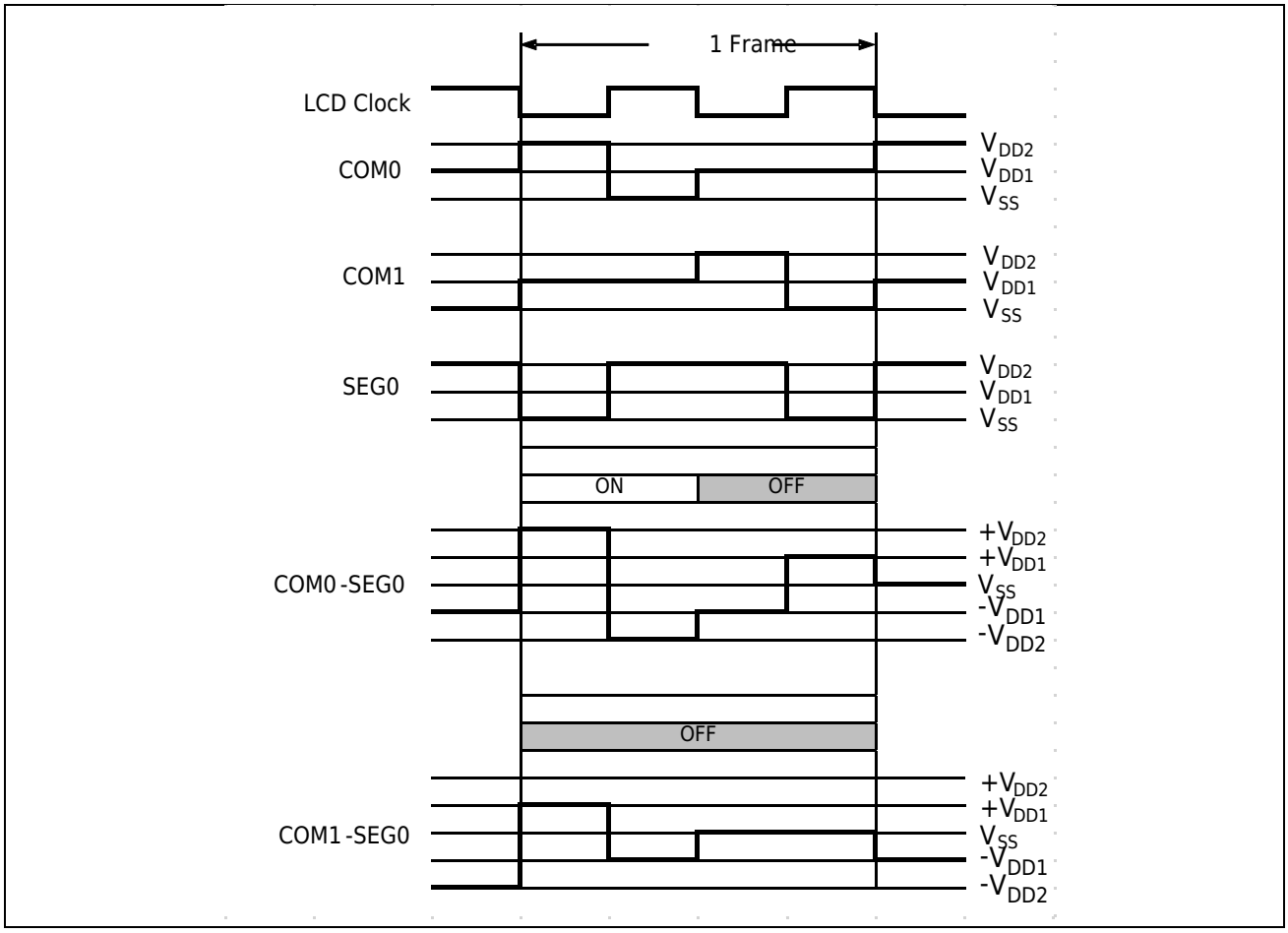
	1/2 Duty	1/3 Duty	1/4 Duty	1/6 Duty	1/8 Duty
1/2 Bias	✓	✓	**Not recommended**	**Not recommended**	**Not recommended**
1/3 Bias	**Not recommended**	**Not recommended**	✓	✓	✓

The driving waveforms in each mode are as follows:

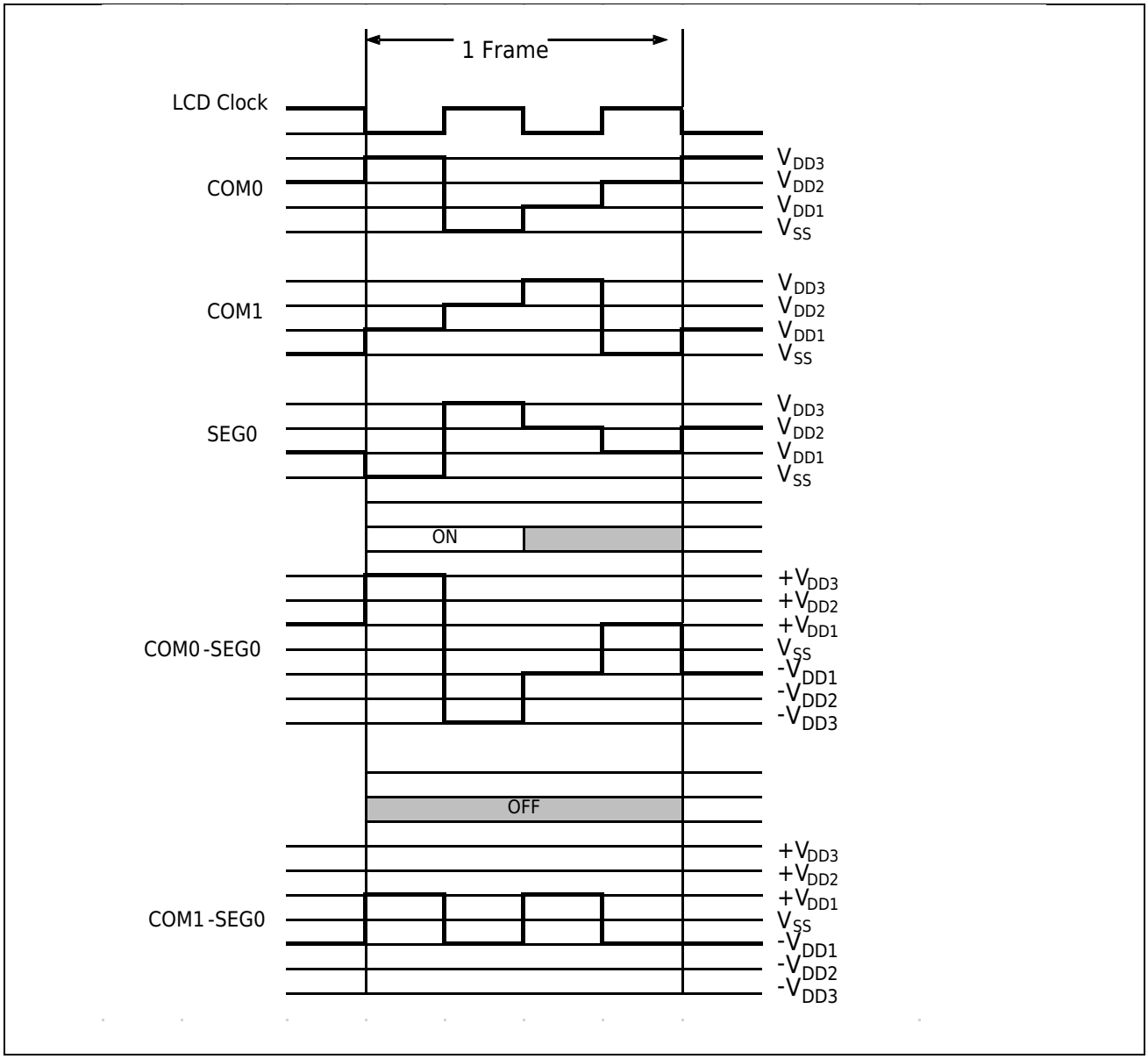
26.4.1 Static drive waveform



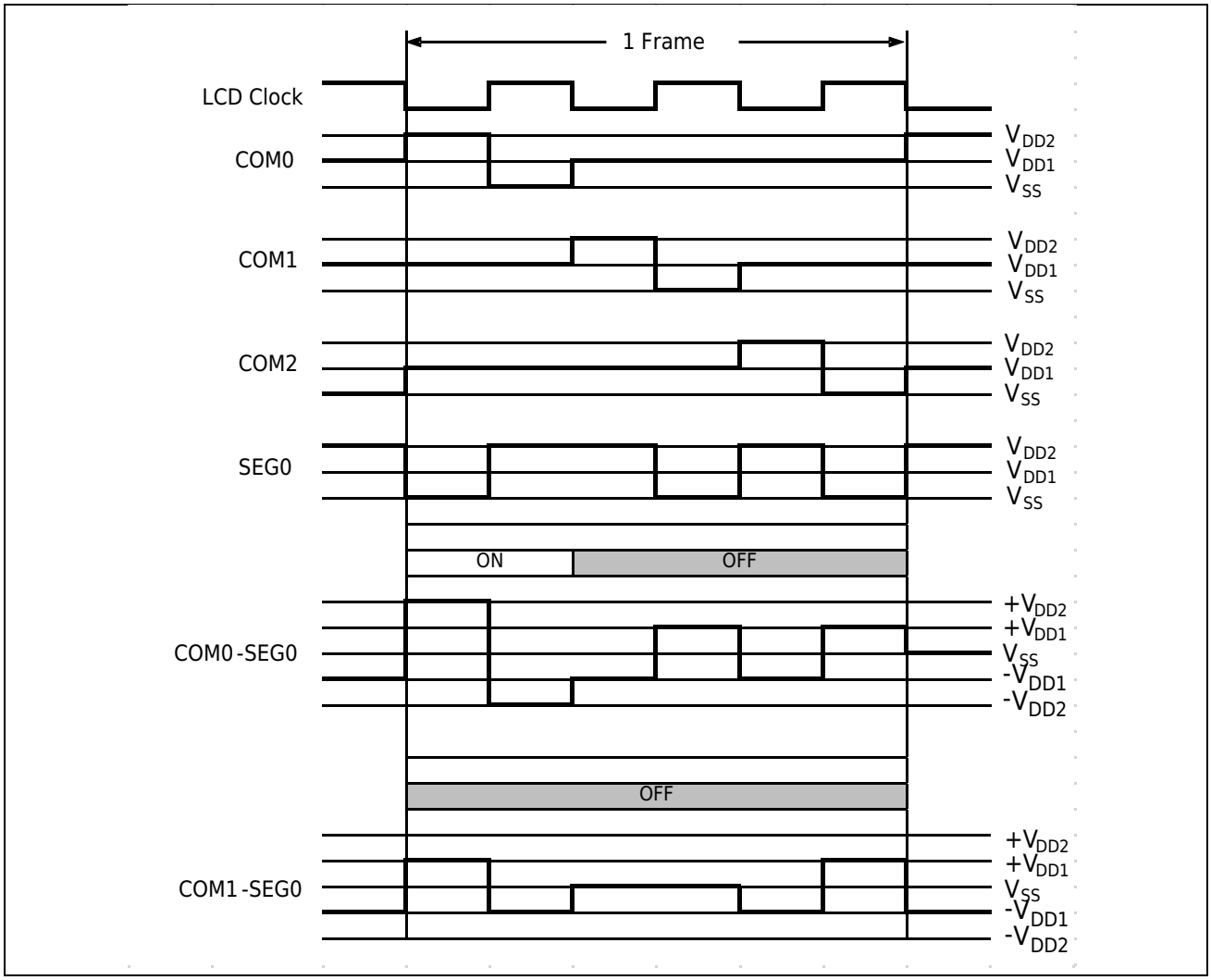
26.4.2 1/2Duty 1/2Bias driving waveform



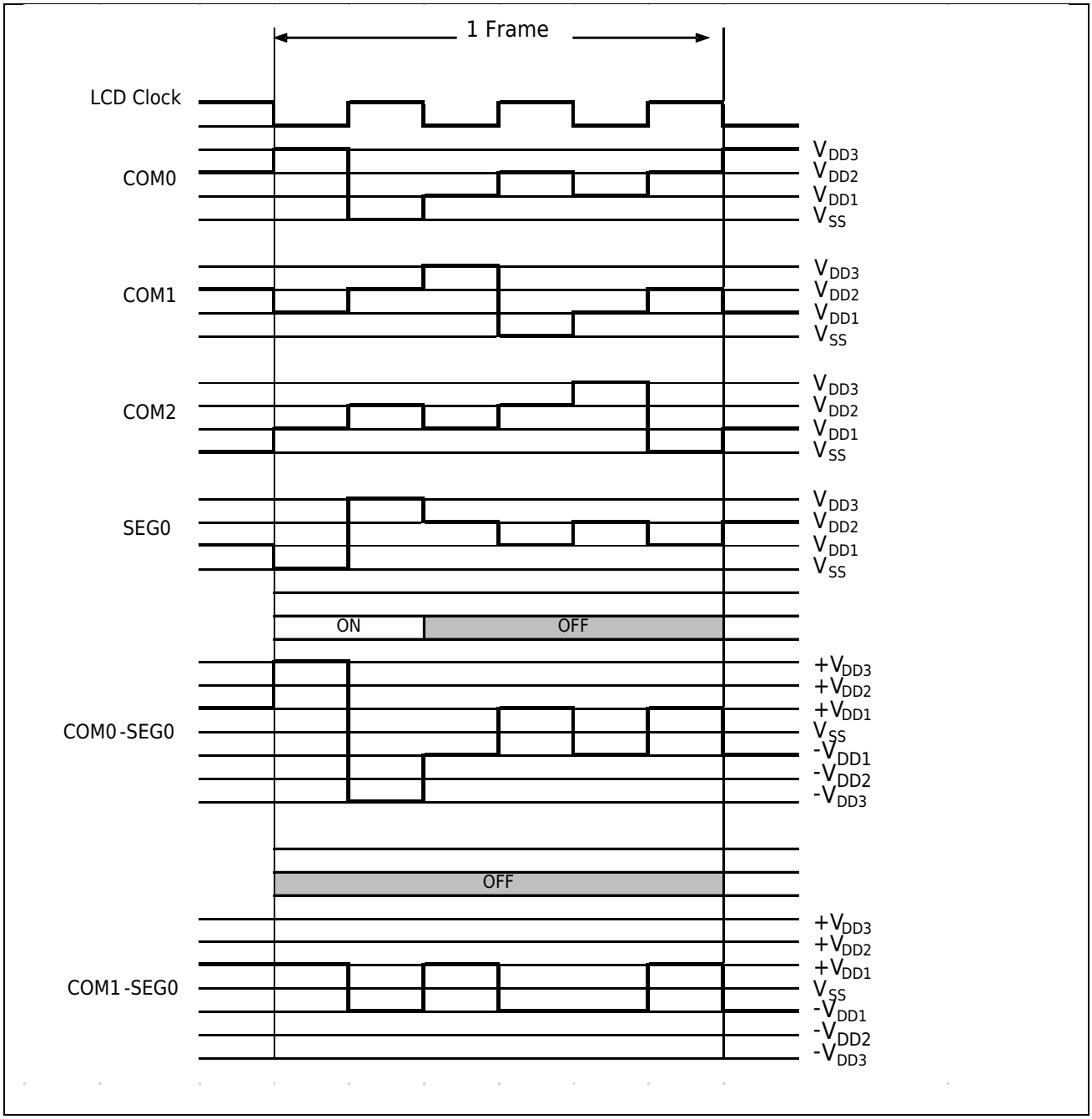
26.4.3 1/2Duty 1/3Bias drive waveform



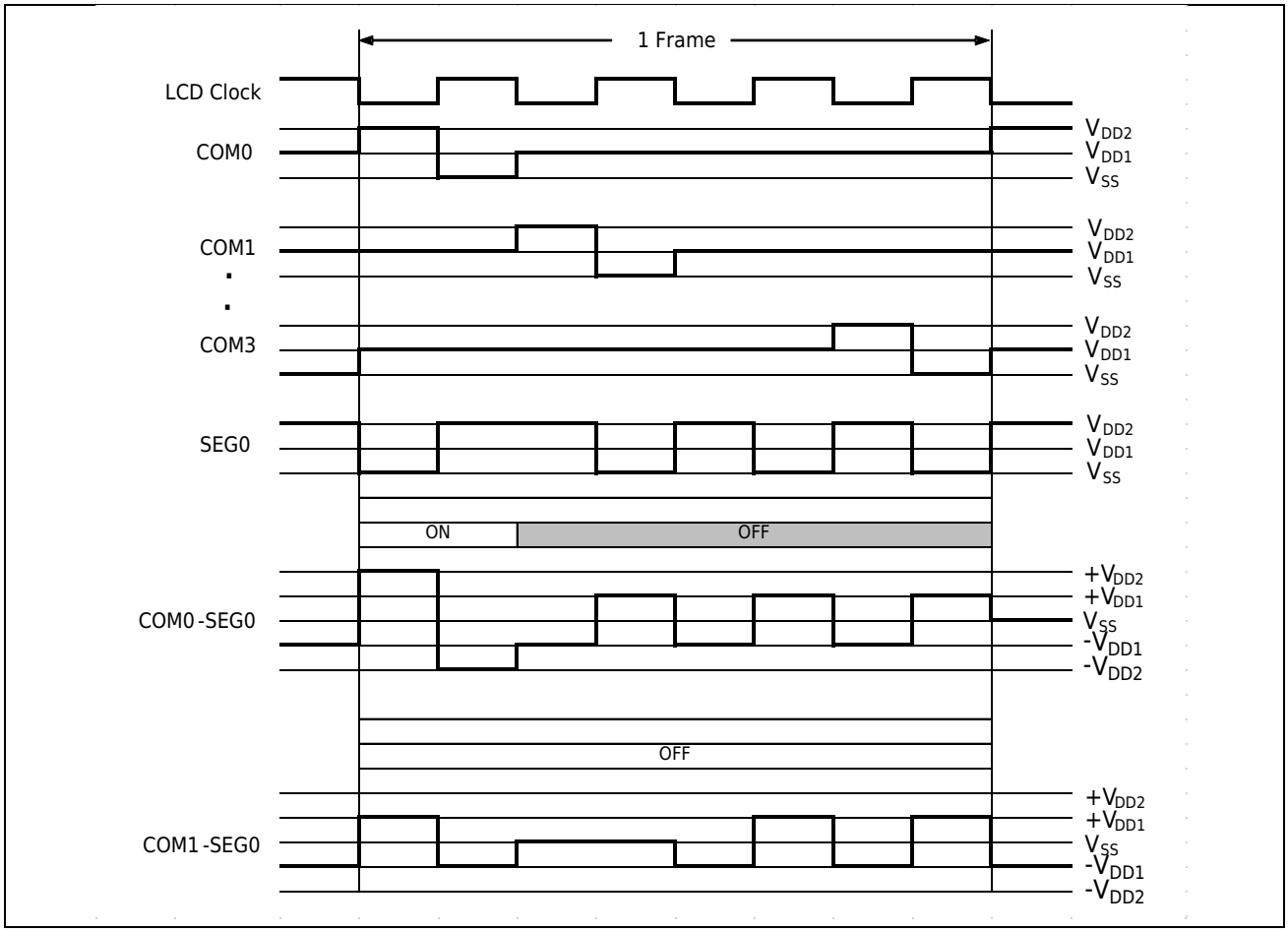
26.4.4 1/3Duty 1/2Bias driving waveform



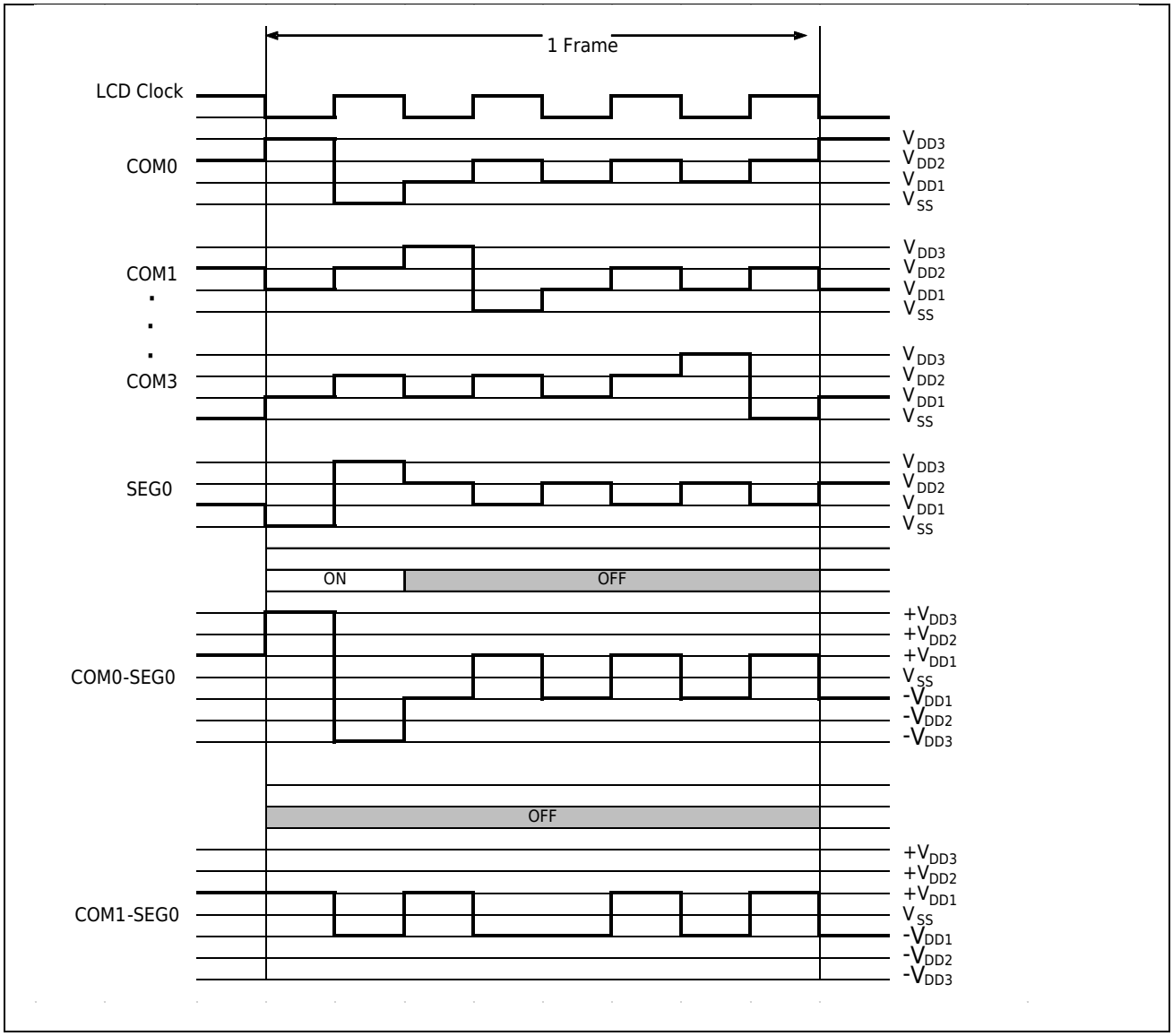
26.4.5 1/3Duty 1/3Bias drive waveform



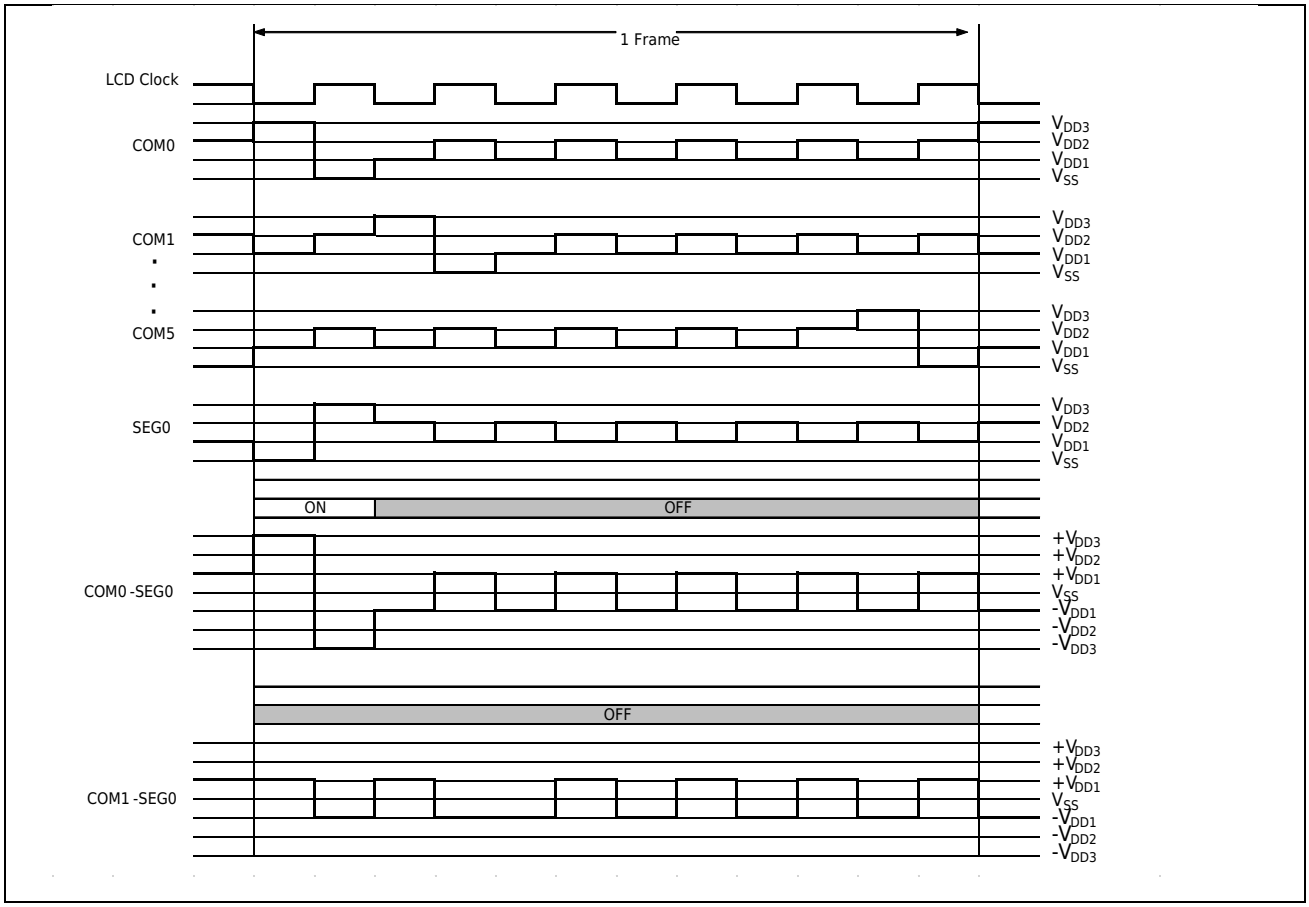
26.4.6 1/4Duty 1/2Bias driving waveform



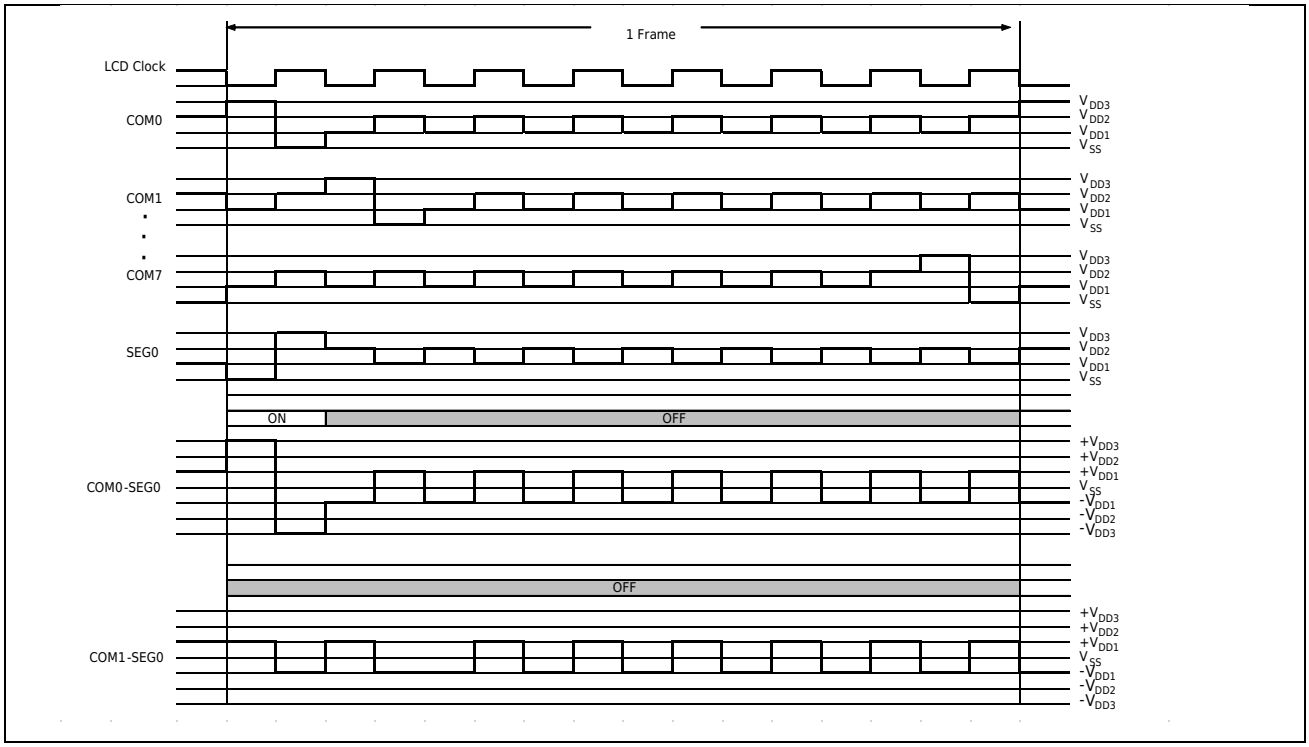
26.4.7 1/4Duty 1/3Bias drive waveform



26.4.8 1/6Duty 1/3Bias drive waveform



26.4.9 1/8Duty 1/3Bias drive waveform



26.5LCD Bias Generation Circuit

LCD Bias voltage has three sources: internal resistor divider, external resistor divider, and external capacitor divider.

1) When the internal resistance voltage division is selected, the chip will automatically switch to the internal circuit, and the driving capability of this mode is weak. When 1/2bias is selected, the resistance of large power consumption / medium power consumption / small power consumption is 240K/360K/720K; when 1/3bias is selected, the resistance of large power consumption / medium power consumption / small power consumption is 360K/ 540K/1080K.
2) When selecting an external resistor to divide the voltage, adjust the resistance value according to the system power consumption and LCD driving capability, and it is recommended to use a 500K resistor. When selecting an external capacitor for voltage division, the drive capability is stronger, and a 100nF capacitor is recommended. However, these two circuits require users to build related circuits on the peripheral pins of the chip and occupy 4 IO ports.

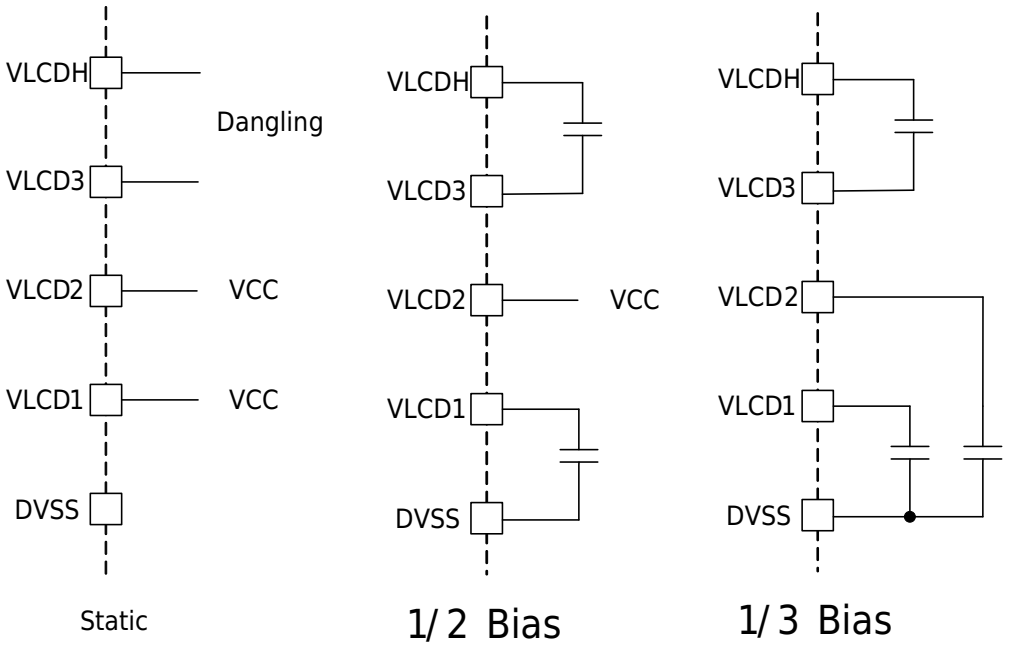
26.5.1 Internal resistance mode

Internal resistance mode VLCDH, VLCD1~VLCD3 can be used as LCD segment output or IO port.

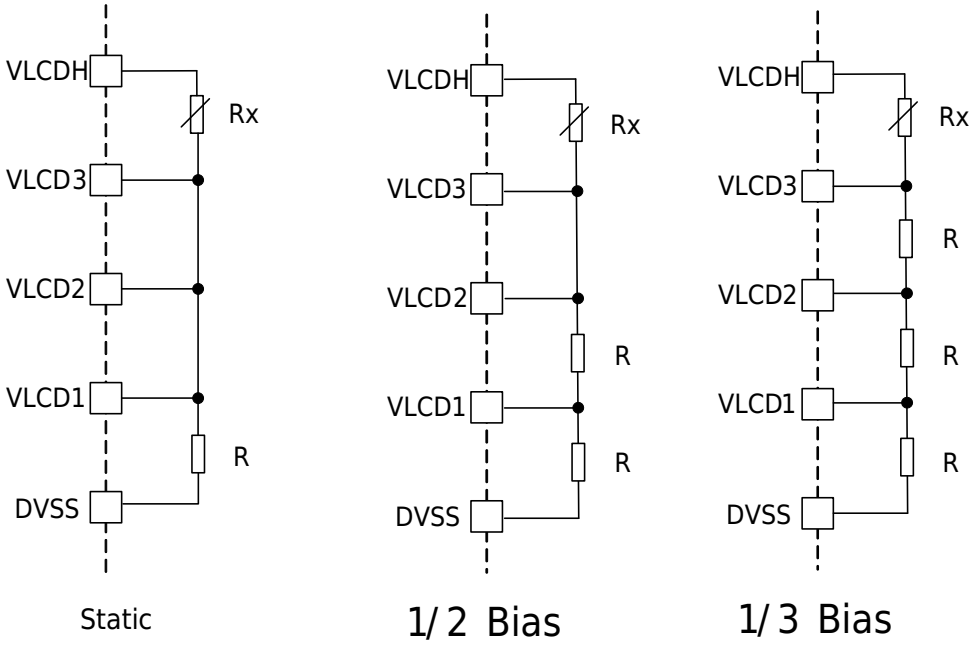
In internal resistance mode, the driving voltage of LCD is controlled by CR0.Contrast, as shown in the table below:

CR0.Contrast	VLCD(1/3 bias)	VLCD(1/2 bias)
0	1.00 * VCC	1.00 * VCC
1	0.94 * VCC	0.92 * VCC
2	0.9 * VCC	0.85 * VCC
3	0.85 * VCC	0.8 * VCC
4	0.81 * VCC	0.75 * VCC
5	0.78 * VCC	0.70 * VCC
6	0.75 * VCC	0.66 * VCC
7	0.72 * VCC	0.63 * VCC
8	0.70 * VCC	0.61 * VCC
9	0.67 * VCC	0.58 * VCC
10	0.65 * VCC	0.55 * VCC
11	0.63 * VCC	0.53 * VCC
12	0.61 * VCC	0.51 * VCC
13	0.59 * VCC	0.48 * VCC
14	0.57 * VCC	0.47 * VCC
15	0.55 * VCC	0.45 * VCC

26.5.2 External Capacitor Mode



26.5.3 External Resistor Mode



Note:

- Rx is an adjustable resistor for adjusting the LCD display contrast
- Select the appropriate resistor R according to the LCD screen used

26.6DMA

LCD supports software and hardware to trigger DMA data transmission, which can automatically move the content to be displayed from RAM or ROM to LCD display RAM. The hardware trigger uses the frame interrupt signal. DMA channel number used by the LCD is [6].

DMA data transfer configuration process:

1. Enable DMA
2. Enable DMA channel enable
3. Select LCD DMA
4. Set the transmission type, transmission length, transmission method
5. Set source start address, destination start address
6. Set the increment mode of source address and destination address
7. Enable DMA interrupts as needed
8. Enable LCD DMA trigger

26.7Interrupt

When the LCD setting is valid, the LCD interrupt can be configured to generate an interrupt for the frame number.

26.8LCD display mode

The LCD supports two display modes. One uses COM as the display unit, and all COM segments of the same SEG are in the same byte (mode 0). The other is that different SEGs of the same COM are in the same byte (mode 1). Selecting the appropriate display mode according to the LCD panel can simplify the program operation.

26.8.1 LCD display mode 1 (MODE = 1)

1/8 Duty

	Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
	SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15	SEG16	SEG17	SEG18	SEG19	SEG20	SEG21	SEG22	SEG23	SEG24	SEG25	SEG26	SEG27	SEG28	SEG29	SEG30	SEG31	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
COM6																																	LCDRAM6
COM7																																	LCDRAM7
	SEG32	SEG33	SEG34	SEG35					SEG40	SEG41	SEG42	SEG43	SEG44	SEG45	SEG46	SEG47	SEG48	SEG49	SEG50	SEG51													
COM0																																	LCDRAM8
COM1																																	LCDRAM9
COM2																																	LCDRAMA
COM3																																	LCDRAMB
COM4																																	LCDRAMC
COM5																																	LCDRAMD
COM6																																	LCDRAME
COM7																																	LCDRAMF

1/6 Duty

	Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
	SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15	SEG16	SEG17	SEG18	SEG19	SEG20	SEG21	SEG22	SEG23	SEG24	SEG25	SEG26	SEG27	SEG28	SEG29	SEG30	SEG31	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
																																	LCDRAM6
																																	LCDRAM7
	SEG32	SEG33	SEG34	SEG35	SEG36	SEG37			SEG40	SEG41	SEG42	SEG43	SEG44	SEG45	SEG46	SEG47	SEG48	SEG49	SEG50	SEG51													
COM0																																	LCDRAM8
COM1																																	LCDRAM9
COM2																																	LCDRAMA
COM3																																	LCDRAMB
COM4																																	LCDRAMC
COM5																																	LCDRAMD
																																	LCDRAME
																																	LCDRAMF

1/4 Duty

	Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
	SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15	SEG16	SEG17	SEG18	SEG19	SEG20	SEG21	SEG22	SEG23	SEG24	SEG25	SEG26	SEG27	SEG28	SEG29	SEG30	SEG31	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
																																	LCDRAM4
																																	LCDRAM5
																																	LCDRAM6
																																	LCDRAM7
	SEG32	SEG33	SEG34	SEG35	SEG36	SEG37	SEG38	SEG39	SEG40	SEG41	SEG42	SEG43	SEG44	SEG45	SEG46	SEG47	SEG48	SEG49	SEG50	SEG51													
COM0																																	LCDRAM8
COM1																																	LCDRAM9
COM2																																	LCDRAMA
COM3																																	LCDRAMB
																																	LCDRAMC
																																	LCDRAMD
																																	LCDRAME
																																	LCDRAMF

1/3 Duty 1/2 Duty

	Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
	SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15	SEG16	SEG17	SEG18	SEG19	SEG20	SEG21	SEG22	SEG23	SEG24	SEG25	SEG26	SEG27	SEG28	SEG29	SEG30	SEG31	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
																																	LCDRAM3
																																	LCDRAM4
																																	LCDRAM5
																																	LCDRAM6
																																	LCDRAM7
	SEG32	SEG33	SEG34	SEG35	SEG36	SEG37	SEG38	SEG39	SEG40	SEG41	SEG42	SEG43	SEG44	SEG45	SEG46	SEG47	SEG48	SEG49	SEG50	SEG51													
COM0																																	LCDRAM8
COM1																																	LCDRAM9
COM2																																	LCDRAMA
																																	LCDRAMB
																																	LCDRAMC
																																	LCDRAMD
																																	LCDRAME
																																	LCDRAMF

26.8.2 LCD display mode 0 (MODE = 0)

1/8 Duty

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	
SEG0								SEG1								SEG2								SEG3								LCDRAM0
SEG4								SEG5								SEG6								SEG7								LCDRAM1
SEG8								SEG9								SEG10								SEG11								LCDRAM2
SEG12								SEG13								SEG14								SEG15								LCDRAM3
SEG16								SEG17								SEG18								SEG19								LCDRAM4
SEG20								SEG21								SEG22								SEG23								LCDRAM5
SEG24								SEG25								SEG26								SEG29								LCDRAM6
SEG28								SEG29								SEG30								SEG31								LCDRAM7
SEG32								SEG40								SEG48																LCDRAM8
SEG33								SEG41								SEG49																LCDRAM9
SEG34								SEG42								SEG50																LCDRAMA
SEG35								SEG43								SEG51																LCDRAMB
								SEG44																								LCDRAMC
								SEG45																								LCDRAMD
								SEG46																								LCDRAME
								SEG47																								LCDRAMF

1/6 Duty

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
COM0	COM1	COM2	COM3	COM4	COM5			COM0	COM1	COM2	COM3	COM4	COM5			COM0	COM1	COM2	COM3	COM4	COM5			COM0	COM1	COM2	COM3	COM4	COM5	COM6	COM7	
SEG0								SEG1								SEG2								SEG3								LCDRAM0
SEG4								SEG5								SEG6								SEG7								LCDRAM1
SEG8								SEG9								SEG10								SEG11								LCDRAM2
SEG12								SEG13								SEG14								SEG15								LCDRAM3
SEG16								SEG17								SEG18								SEG19								LCDRAM4
SEG20								SEG21								SEG22								SEG23								LCDRAM5
SEG24								SEG25								SEG26								SEG29								LCDRAM6
SEG28								SEG29								SEG30								SEG31								LCDRAM7
SEG32								SEG40								SEG48																LCDRAM8
SEG33								SEG41								SEG49																LCDRAM9
SEG34								SEG42								SEG50																LCDRAMA
SEG35								SEG43								SEG51																LCDRAMB
SEG36								SEG44																								LCDRAMC
SEG37								SEG45																								LCDRAMD
								SEG46																								LCDRAME
								SEG47																								LCDRAMF

1/4 Duty

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
COM0	COM1	COM2	COM3					COM0	COM1	COM2	COM3					COM0	COM1	COM2	COM3					COM0	COM1	COM2	COM3					
SEG0								SEG1								SEG2								SEG3								LCDRAM0
SEG4								SEG5								SEG6								SEG7								LCDRAM1
SEG8								SEG9								SEG10								SEG11								LCDRAM2
SEG12								SEG13								SEG14								SEG15								LCDRAM3
SEG16								SEG17								SEG18								SEG19								LCDRAM4
SEG20								SEG21								SEG22								SEG23								LCDRAM5
SEG24								SEG25								SEG26								SEG29								LCDRAM6
SEG28								SEG29								SEG30								SEG31								LCDRAM7
SEG32								SEG40								SEG48																LCDRAM8
SEG33								SEG41								SEG49																LCDRAM9
SEG34								SEG42								SEG50																LCDRAMA
SEG35								SEG43								SEG51																LCDRAMB
SEG36								SEG44																								LCDRAMC
SEG37								SEG45																								LCDRAMD
SEG38								SEG46																								LCDRAME
SEG39								SEG47																								LCDRAMF

1/3 Duty 1/2 Duty

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15	Bit16	Bit17	Bit18	Bit19	Bit20	Bit21	Bit22	Bit23	Bit24	Bit25	Bit26	Bit27	Bit28	Bit29	Bit30	Bit31	
COM0	COM1	COM2						COM0	COM1	COM2						COM0	COM1	COM2						COM0	COM1	COM2						
SEG0								SEG1								SEG2								SEG3								LCDRAM0
SEG4								SEG5								SEG6								SEG7								LCDRAM1
SEG8								SEG9								SEG10								SEG11								LCDRAM2
SEG12								SEG13								SEG14								SEG15								LCDRAM3
SEG16								SEG17								SEG18								SEG19								LCDRAM4
SEG20								SEG21								SEG22								SEG23								LCDRAM5
SEG24								SEG25								SEG26								SEG29								LCDRAM6
SEG28								SEG29								SEG30								SEG31								LCDRAM7
SEG32								SEG40								SEG48																LCDRAM8
SEG33								SEG41								SEG49																LCDRAM9
SEG34								SEG42								SEG50																LCDRAMA
SEG35								SEG43								SEG51																LCDRAMB
SEG36								SEG44																								LCDRAMC
SEG37								SEG45																								LCDRAMD
SEG38								SEG46																								LCDRAME
SEG39								SEG47																								LCDRAMF

26.9LCD register

Base address 0x40005C00

Table 26-1 LCD register

Register	Offset address	Description
LCD_CR0	0x000	LCD Configuration Register 0
LCD_CR1	0x004	LCD Configuration Register 1
LCD_INTCLR	0x008	LCD Interrupt Clear Register
LCD_POEN0	0x00C	LCD Output Configuration Register
LCD_POEN1	0x010	LCD Output Configuration Register
LCD_RAM0	0x040	LCD RAM0
LCD_RAM1	0x044	LCD RAM1
LCD_RAM2	0x048	LCD RAM2
LCD_RAM3	0x04C	LCD RAM3
LCD_RAM4	0x050	LCD RAM4
LCD_RAM5	0x054	LCD RAM5
LCD_RAM6	0x058	LCD RAM6
LCD_RAM7	0x05C	LCD RAM7
LCD_RAM8	0x060	LCD RAM8
LCD_RAM9	0x064	LCD RAM9
LCD_RAMA	0x068	LCD RAM10
LCD_RAMB	0x06C	LCD RAM11
LCD_RAMC	0x070	LCD RAM12
LCD_RAMD	0x074	LCD RAM13
LCD_RAME	0x078	LCD RAM14
LCD_RAMF	0x07C	LCD RAM15

26.9.1 Configuration Register 0 (LCD_CR0)

Offset address 0x000

Reset value 0x000000DA

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Contrast				BSEL			Duty			Bias	CpClk		LcdClk		EN
RW				RW			RW			RW	RW		RW		RW

Bit	Marking	Functional description
31:16	Reserved	Keep
15:12	Contrast	LCD Contrast Adjustment Note: It is only valid when the Bias voltage source selects internal resistor voltage divider. Constrast value, the smaller the amplitude of the LCD waveform. 0X0, the LCD waveform has the largest amplitude and the largest contrast; ..... When 0XF, the LCD waveform amplitude is the smallest, and the contrast is the smallest;
11:9	BSEL	Bias voltage source selection 111：Reserved 110: Internal resistor divider, high power consumption mode 101：Reserved 100: Internal resistor divider, low power consumption mode 011：Reserved 010: internal resistor divider, medium power consumption mode 001: Capacitive voltage divider mode, requires external circuit cooperation 000: External resistance mode, need external circuit to cooperate
8:6	Duty	LCD duty configuration 000: Static 001: 1/2 duty 010: 1/3 duty 011: 1/4 duty 100: Reserved 101: 1/6 duty 110: Reserved 111: 1/8 duty

duty	PAD	FUNCTION
1/4 duty	COM4SEG39PAD	SEG39
	COM5SEG38PAD	SEG38
	COM6SEG37PAD	SEG37
	COM7SEG36PAD	SEG36
1/6 duty	COM4SEG39PAD	COM4
	COM5SEG38PAD	COM5
	COM6SEG37PAD	SEG37
	COM7SEG36PAD	SEG36
1/8 duty	COM4SEG39PAD	COM4
	COM5SEG38PAD	COM5
	COM6SEG37PAD	COM6
	COM7SEG36PAD	COM7

5	Bias	LCD Bias configuration 0: 1/3 bias 1: 1/2 bias
4:3	CpClk	Voltage Pump Clock Frequency Selection 00: 2k Hz 01: 4k Hz 10: 8k Hz 11: 16k Hz
2:1	LcdClk	LCD scan frequency selection 00: 64Hz 01: 128Hz 10: 256Hz 11: 512Hz Note: LCD frame frequency = LCD scanning frequency × Duty
0	EN	LCD enable control 1: Enable 0: Prohibited

26.9.2 Configuration Register 1 (LCD_CR1)

Offset address 0x004

Reset value 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				INTF	Dma En	IE	Mode	ClkSrc	Blink En	BlinkCnt					
				RO	RW	RW	RW	RW	RW	RW					

Bit	Marking	Functional description
31:11	Reserved	Keep
11	INTF	Interrupt logo 1: interrupt 0: no interrupt
10	DmaEn	DMA hardware trigger enable 1: Enable LCD interrupt to trigger DMA, 0: Disable LCD interrupt to trigger DMA
9	IE	Interrupt enable 1: Enable 0: Prohibited
8	Mode	LCD RAM display mode selection 0 mode 0 1 mode 1
7	ClkSrc	LCD clock source selection 1：XTL 0：RCL
8	BlinkEn	LCD splash screen configuration 1: Enable 0: Prohibited
5:0	BlinkCnt	Splash screen frequency and LCD interrupt interval setting Note: LCD blinking frequency = LCD frame frequency / (BlinkCnt+1) LCD interrupt interval = (BlinkCnt+1)*(1/LCD frame frequency)

26.9.3 Interrupt Clear Register (LCD_INTCLR)

Offset address 0x008

Reset value 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved					INTF	Reserved									
					R0W0										

Bit	Marking	Functional description
31:11	Reserved	Keep
10	INTF	Interrupt flag is cleared, write 0 to clear, write 1 to be invalid
9:0	Reserved	Keep

26.9.4 Output Configuration Register 0 (LCD_POEN0)

Offset address 0x00C

Reset value 0xFFFFFFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
S31	S30	S29	S28	S27	S26	S25	S24	S23	S22	S21	S20	S19	S18	S17	S16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
S15	S14	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:0	Sx	Segx output control bit 0: SEG output enable 1: SEG output is off, you can use other functions, such as IO, analog input and output

26.9.5 Output Configuration Register 1 (LCD_POEN1)

Offset address 0x010

Reset value 0x01FF_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved						COM MUX	C3	C2	C1	C0	S51	S50	S49	S48	S47
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
S46	S45	S44	MUX	S43	S42	S41	S40	S39 C4	S38 C5	S37 C6	S36 C7	S35	S34	S33	S32
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
3:0	Sx	Segx output control bit 0: SEG output enable 1: SEG output is off, you can use other functions, such as IO, analog input and output
7:4	SxCy	Segx/COMy output control bits 0: SEG/COM output enable 1: SEG/COM output is off, you can use other functions, such as IO, analog input and output SEG COM pin function selection is determined by CR0.DUTY
11:8	Sx	Segx output control bit 0: SEG output enable 1: SEG output is off, you can use other functions, such as IO, analog input and output
12	MUX	SEG32~SEG35 port function selection, refer to the selection table for details
20:13	Sx	SEG40-SEG51 port function selection
24:21	Cx	COMx output control bit 0: COM output enable 1: COM output is closed, you can use other functions, such as IO, analog input and output
25	COMMUX	COM2/3 pin output in different positions. 0: COM2/3 output on PA11/PA12 pin 1: COM2/3 output on PF6/PA15 pin

VLCDXSEGXPAD Status Description			How registers are configured				
			MUX	S<35: 32>	BSEL		
VLCDXSEGXPAD selects GPIO, when LCD disable(LCD_ON=0)		VLCDHSEG35=IO	1	1111	X	X	X
		VLCD3SEG34=IO					
		VLCD2SEG33=IO					
		VLCD1SEG32=IO					
VLCDXSEGXPAD selects IO and LCD enable (LCD_ON=1), only the internal resistance working mode can be selected at this time	Small resistance (high current) mode	VLCDHSEG35=IO	1	1111	1	1	0
		VLCD3SEG34=IO					
		VLCD2SEG33=IO					
		VLCD1SEG32=IO					
	Medium resistance (medium current) mode	VLCDHSEG35=IO	1	1111	0	1	0
		VLCD3SEG34=IO					
		VLCD2SEG33=IO					
		VLCD1SEG32=IO					
	Large resistance (low current) mode	VLCDHSEG35=IO	1	1111	1	0	0
		VLCD3SEG34=IO					
		VLCD2SEG33=IO					
		VLCD1SEG32=IO					
Select the external resistor working mode, the internal resistor is disconnected		VLCDHSEG35=VOUT	0	0000	0	0	0
		VLCD3SEG34=VLCD3					
		VLCD2SEG33=VLCD2					
		VLCD1SEG32=VLCD1					
Select Internal Resistor Operation Mode	Small resistance (high current) mode	VLCDHSEG35=SEG35/IO	0	XXXX	1	1	0
		VLCD3SEG34=SEG34/IO					
		VLCD2SEG33=SEG33/IO					
		VLCD1SEG32=SEG32/IO					
	Medium resistance (medium current) mode	VLCDHSEG35=SEG35/IO	0	XXXX	0	1	0
		VLCD3SEG34=SEG34/IO					
		VLCD2SEG33=SEG33/IO					
		VLCD1SEG32=SEG32/IO					
	Large resistance (low current) mode	VLCDHSEG35=SEG35/IO	0	XXXX	1	0	0
		VLCD3SEG34=SEG34/IO					
		VLCD2SEG33=SEG33/IO					
		VLCD1SEG32=SEG32/IO					
Select the external capacitor working mode, and the internal resistor is disconnected		VLCDHSEG35=DH1	0	0000	0	0	1
		VLCD3SEG34=DH2					
		VLCD2SEG33=VLCD2					
		VLCD1SEG32=VLCD1					

26.9.6 LCD_RAM0~7

Offset address 0x040~0x5c

Reset value: none

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:0	Dx	LCD point output, display reference LCD display mode 0 corresponds to the SEG COM cross point is not lit; 1 corresponds to the SEG COM cross point is lit;

26.9.7 LCD_RAM8~F

Offset address 0x060~0x7C

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								D23	D22	D21	D20	D19	D18	D17	D16
								RW	RW	RW	RW	RW	RW	RW	RW

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Reset value: none

Bit	Marking	Functional description
31:24	Reserved	Keep
23:0	Dx	LCD point output, display reference LCD display mode 0 corresponds to the SEG COM cross point is not lit; 1 corresponds to the SEG COM cross point is lit;

27 Crystal-less USB Clock Calibrator (CTS)

The clock calibrator can adjust and calibrate the clock frequency of RCH48M, and can also adjust and calibrate the clock frequency of other RC oscillators, and can also be used as a general timer. The main function of this module is RCH48M automatic calibration, and other functions are additional functions when 48M automatic calibration is not used.

Adjustment and Calibration The RCH48M measures the frequency of the oscillator to be calibrated through an optional reference signal. The frequency of the oscillator can be automatically adjusted according to the measured frequency error, or manual adjustment can be used.

The RCH48M provides a precise clock for the USB peripherals, in this case, the base reference signal can be provided by the Start of Frame (SOF) packet signal on the USB bus, which is sent by the USB master at 1ms intervals. The USB module will generate a flip signal with a period of 2ms. The CTS system clock reference signal can also be provided by the XTL oscillator output or external pins, or generated by user software.

RCH24M and RCL oscillation calibration does not support automatic calibration, only manual adjustment is supported. The reference clock source can be flexibly selected.

The timer is a reload mode count timer that can generate a capture interrupt. Support low power wake-up.

27.148M clock calibration

27.1.1 System characteristics

Selectable Base Reference Source with Programmable Prescaler and Polarity

- USB SOF packet reception
- XTL oscillator output
- External pin
- LPTIMER flip

Base reference pulse can be generated by software

Oscillator automatic fine-tuning function without CPU participation

Manual software control option for faster calibration initiation

16 -bit frequency error counter for automatic error capture and reload

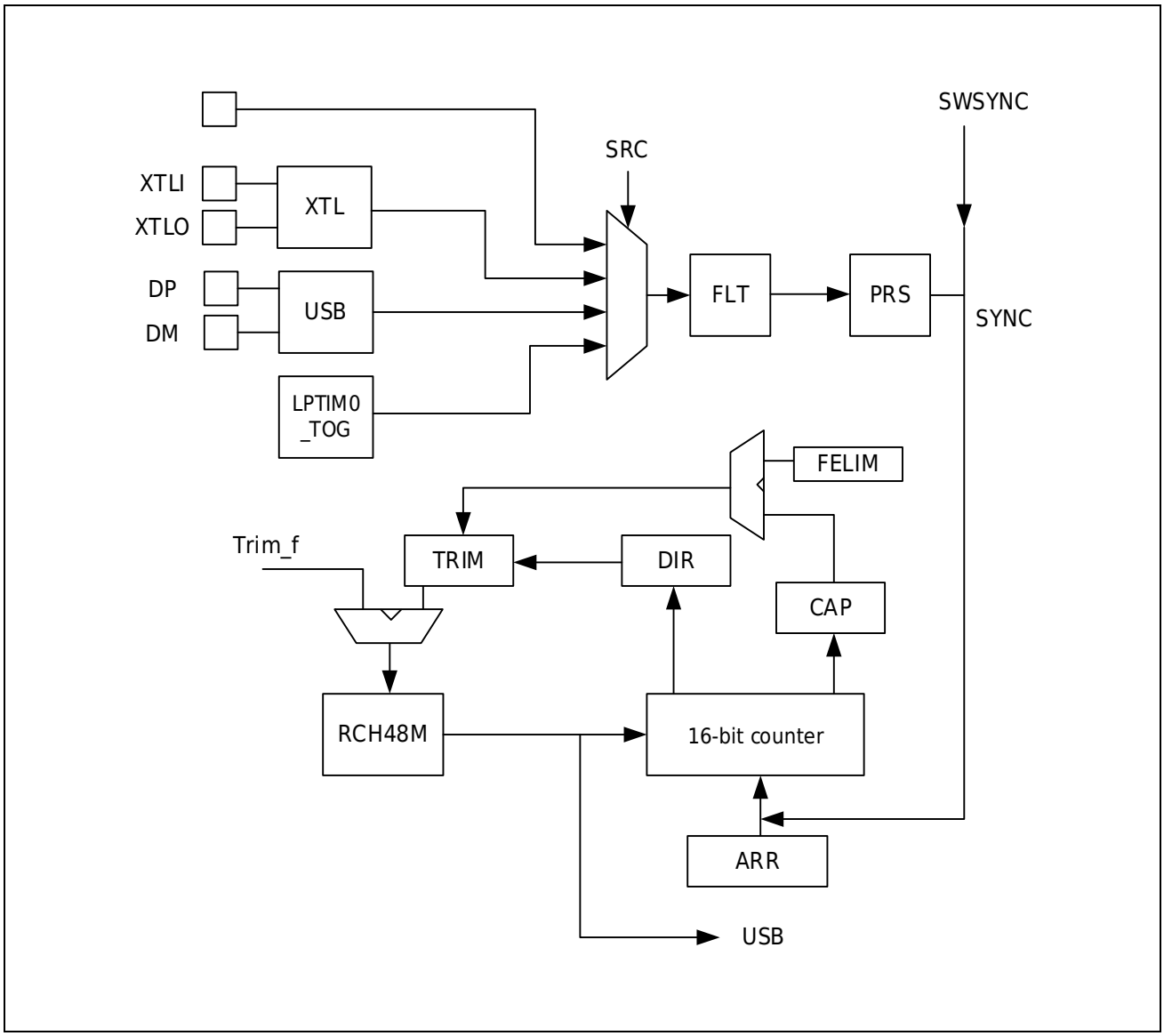
Programmable error limits for automatic frequency error values and status flags

Maskable interrupt

- Counter count underflow (UDF)
- Clock calibration is OK (OK)

- Clock Alignment Warning (WARN)
- Clock Calibration Error (ERR)

27.1.2 Block diagram



27.1.3 Reference source synchronization input

The clock calibration system reference source can be selected through the register CTS_CFGR, which can be an external pin signal, XTL clock, USB_SOF signal or overflow signal of LPTIMER. To stabilize the reference signal input, it is configurable to use a simple digital filter to remove any interference. The signal source can also configure the polarity, and can be divided by a prescaler to obtain a synchronization signal within a reasonable frequency range (usually around 1mS).

The synchronization signal can also be generated by software by setting the SW_SYNC bit in the CTS_CR register to 1. When using SW_SYNC, select the reference clock as the pin signal, and ensure that the pin input level is low.

27.1.4 Frequency Error Measurement

The frequency error counter is a 16 -bit down / up counter that reloads the ARR value on each SYNC event of the reference source. It starts counting down until it reaches a zero value, at which point a UDF (expected underflow) event is generated. It will then increment the count up to the OUTRANGE limit, in which case it will eventually stop counting (if no SYNC event is received) and generate a MISS event. The OUTRANGE limit is defined as the frequency error limit (FELIM field of the CTS_CFGR register) multiplied by 128.

SYNC event is detected, the actual value of the frequency error counter and its direction are stored in the FECAP (frequency error capture) field and FEDIR (frequency error direction) bit of the CTS_ISR register. When a SYNC event is detected during the down counting phase (before reaching the zero value), it means that the actual frequency is less than the target frequency (thus, the TRIM value should be incremented); when a SYNC event is detected during the up counting, it means that the actual frequency is greater than The target frequency (thus, the TRIM value should be decremented).

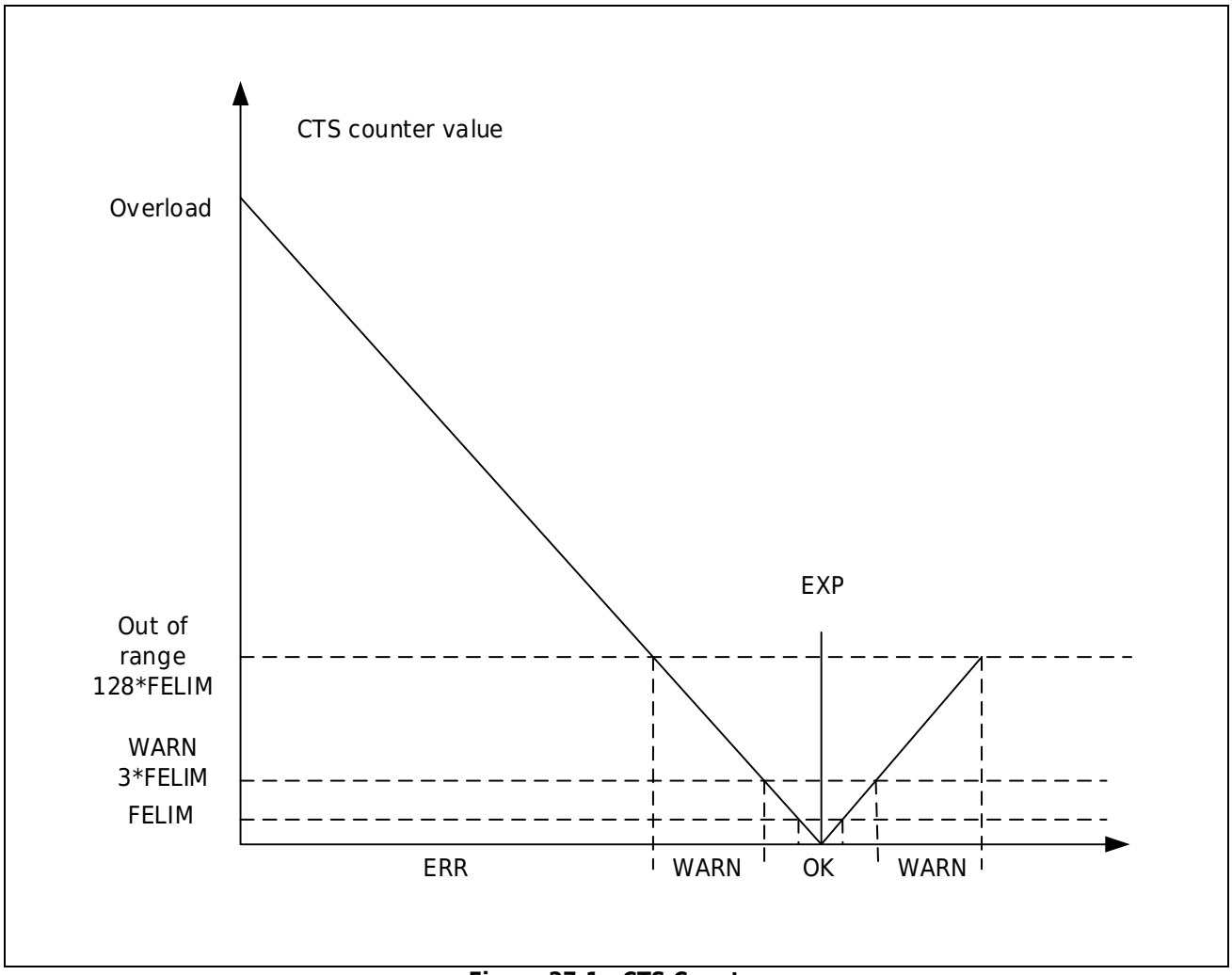


Figure 27-1 CTS Counter

27.1.5 Frequency error trimming

The measured frequency error is estimated by comparing the frequency value to a set of limits:

- TOLERANCE LIMIT (tolerance limit), given directly in the FELIM field of the CTS_CFGR register
- WARNING LIMIT (warning limit), defined as 3 * FELIM value
- OUTRANGE (error limit), defined as 128 * FELIM value

The result of this comparison is used to generate a status indication and to control automatic trimming, which is enabled by setting the AUTO_TRIMEN bit in the CTS_CR register.

- When the frequency error is lower than the tolerance limit, it means that the actual trimming value in the TRIM field is the optimal value, so no trimming operation is required.
  - Indicate OK status
  - TRIM value in AUTOTRIM mode
- When the frequency error is below the warning limit but above or equal to the tolerance limit, it means some kind of trimming is required, but only one trimming step is needed to reach the optimal TRIM value.
  - Indicate OK status
  - Adjust TRIM value by ±1 fine adjustment in AUTOTRIM mode
- When the frequency error is above or equal to the warning limit but below the error limit, it means that a stronger trim operation is required and there is a risk that the optimal TRIM value will not be reached in the next cycle.
  - Indicates WARN status
  - Adjust TRIM value by ±2 fine adjustment in AUTOTRIM mode
- When the frequency error is higher than or equal to the error limit, it means that the frequency is out of the fine-tuning range. When the SYNC input is dirty or one of the SYNC pulses is missing (for example, when a USB SOF is corrupted). This also happens.
  - Indicates ERR or MISS status
  - TRIM value in AUTOTRIM mode

Note:

- If the actual value of the TRIM field is very close to its limit value, if it needs to be adjusted towards its limit value, it will only be adjusted to its limit value, and then the TRIM value remains at the limit value, and the OVF status will be indicated at this time.
- In AUTOTRIM mode (AUTO_TRIMEN bit set in CTS_CR register), the TRIM field of CTS_CR is trimmed by hardware and is read-only.

27.1.6 CTS initialization and configuration

ARR value

The ARR value should be chosen based on the ratio of the target frequency to the prescaled reference sync source frequency. This value is then decremented by 1 to achieve the desired synchronization at a value of zero. The specific formula is as follows:

ARR = (fTARGET / fSYNC) -1

ARR field corresponds to a target frequency of 48 MHz and a sync signal frequency (SOF signal from USB) of 1 kHz.

FELIM value

FELIM value is closely related to the characteristics of the RCH48M oscillator and its typical fine-tuning step size. The optimal value corresponds to half the fine-tuning step size, expressed in ticks of the RCH48M oscillator clock. The following formulas can be used:

FELIM = (fTARGET / fSYNC) * STEP[%] / 100% / 2

Results should always be rounded to the nearest integer value for best nudge response. If the application does not require frequent trimming operations, the trimming hysteresis can be increased by slightly increasing the FELIM value.

FELIM field corresponds to (fTARGET / fSYNC) = 48000 and a typical fine-tuning step size of 0.14%.

Note:

- Misconfiguring the ARR and FELIM fields fails to implement hardware protection, resulting in erratic fine-tuned responses. The intended mode of operation requires a correct setting of the ARR value (according to the sync source frequency), which is also greater than 128 * FELIM value (OUTRANGE limit).

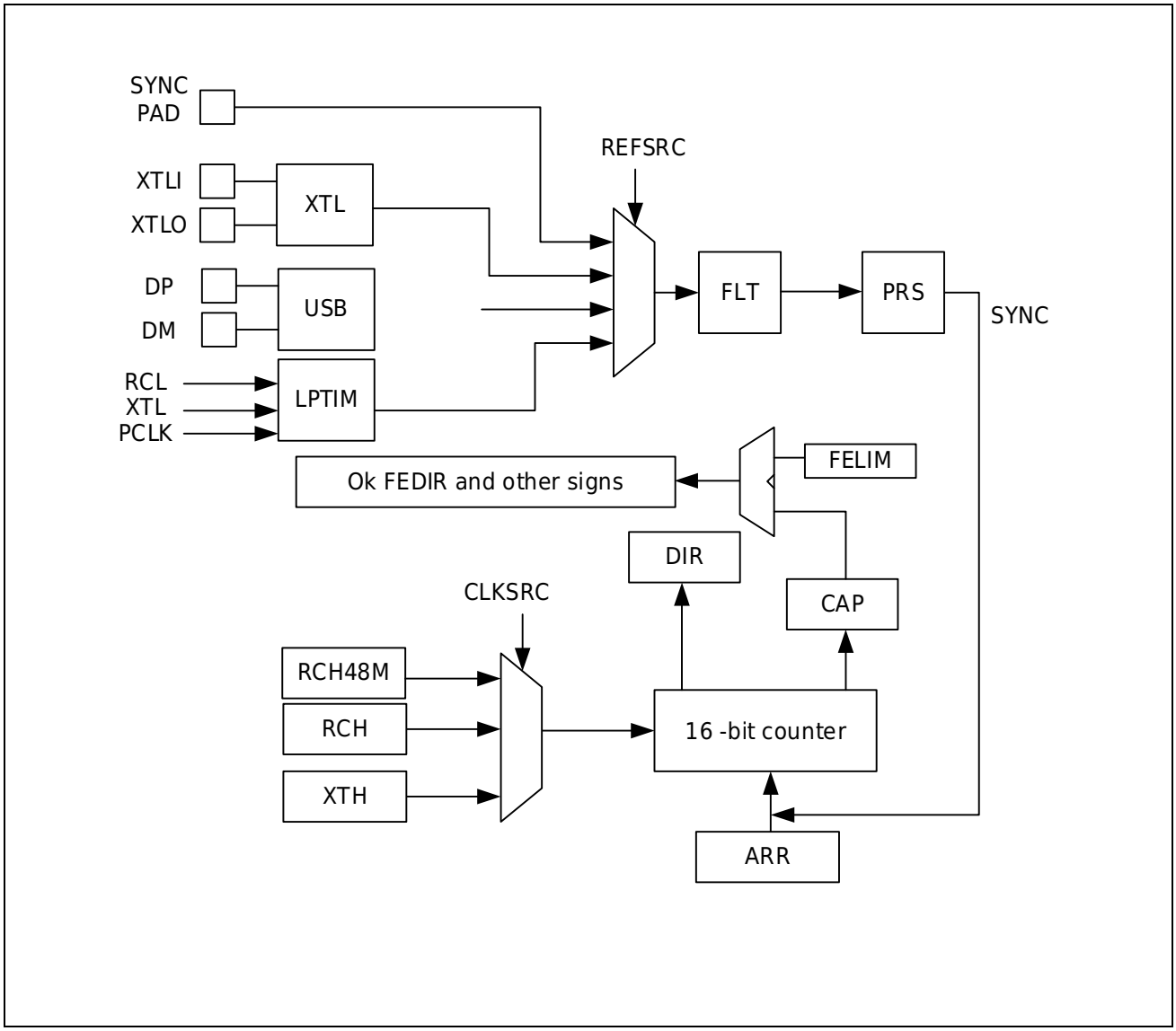
27.2 Other Oscillation Calibration

27.2.1 System characteristics

To calibrate the high-speed clock RCH reference clock, you can choose the port input, or the output of other internal precision clock sources through LPTIMER.

When calibrating the low-speed clock RCL, use RCL as the reference clock, and select the counting clock as an accurate internal clock or an external input accurate clock.

27.2.2 Block diagram



27.2.3 Calibration Instructions

RCH Oscillation Calibration:

1. Set the reference clock as port input or precise XTL, and set the appropriate reference clock frequency division
2. Set the calibration clock to RCH
3. Set CFGR to set the target frequency and error range to be calibrated
4. Judge FEDIR in the interrupt service routine, if it is 1, increase the frequency value, increase the TRIM value; otherwise decrease

The TRIM value is initially set to the median value, and the value to be changed next time is 1/2 of the value to be changed last time, that is, the 2- point method. The 9-bit TRIM value can be calibrated 8 times.

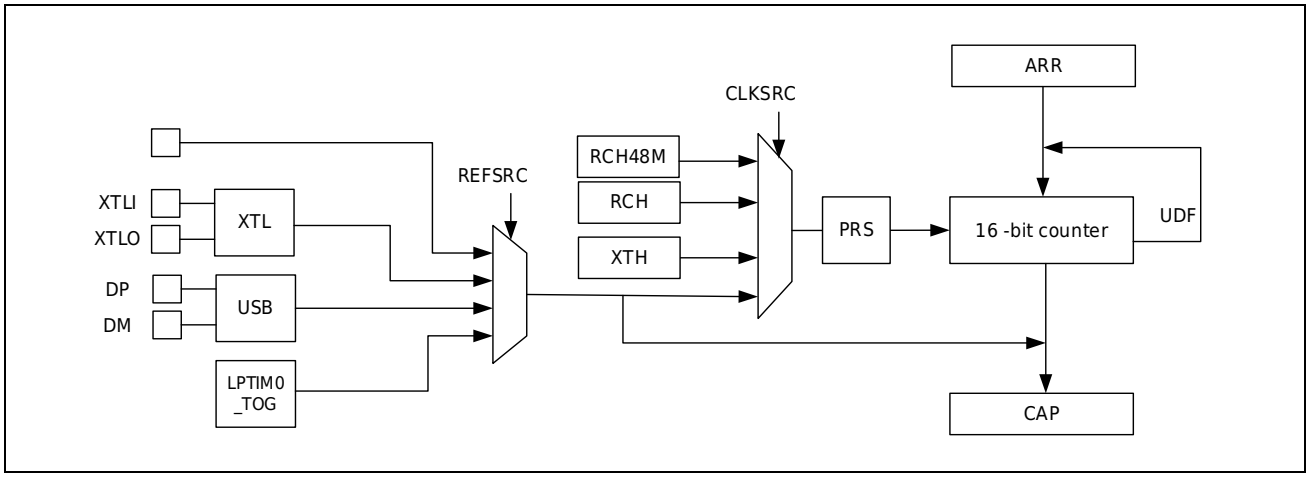
RCL Oscillation Calibration:

1. Set the reference clock RCL as the clock of the LPTIMER timer to generate the LPTIM0 flip signal of the parameter
2. Set the calibration clock to XTH or the calibrated RCH clock
3. Set CFGR to set the target frequency and error range to be calibrated
4. Judge FEDIR in the interrupt service routine, if it is 0, increase the frequency value, increase the TRIM value; otherwise decrease

The TRIM value is initially set to the median value, and the value to be changed next time is 1/2 of the value to be changed last time, that is, the 2- point method. The 9-bit TRIM value can be calibrated 8 times.

27.3Timer function description

27.3.1 Block diagram



Timer Features

- 16- bit down-counting auto-reload low-power timer
- Various clock sources can be configured
- 1/2/4/8/16/32/64/256 prescaler configurable
- Input capture function
- Timer wake-up function
- Support overflow interrupt, capture interrupt

27.3.2 Timer Clock Selection

The timer can choose internal high-speed clocks such as RCH48M, RCH, XTH, etc., or can choose external port input, low-speed XTL and LPTIMER timer reversal signals as counting clocks. When the high-speed clock is selected, the external input can realize the edge capture function of the rising edge or the falling edge.

27.3.3 Timer

The count value of the timer is counted down from ARR. When the count reaches 0, an underflow will occur. If the interrupt is enabled, an underflow interrupt will be generated. The counting cycle is ARR+1.

TIM_EN can only be changed when CEN is 0. When TIM_EN is 1 and CEN is written 1, the timer will reload the reload value to the timer.

27.4CTS register

Base address 0x40005000

Table 27-1 CTS register

Register	Offset address	Description
CTS_CR	0x000	control register
CTS_CFGR	0x004	configuration register
CTS_ISR	0x008	Interrupt and Status Registers
CTS_ICR	0x00C	Interrupt Flag Clear Register

27.4.1 CTS Control Register (CTS_CR)

Offset address 0x00

Reset value 0x00000000

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
TIM_EN	TRIM							SW_SYNC	AUTO_TRIMEN	CEN	CAPIE	UDFIE	ERRIE	WARNIE	OKIE
RW	RW							RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:16	Reserved	Keep
15	TIM_EN	Timer function (can only be rewritten when CEN is 0)
14:8	TRIM	RCH48M oscillation trim value, when AUTO_TRIMEN is 1, is controlled by hardware and read-only; it is a signed number, 0x40 is the minimum, 0x3F is the maximum, 0x00 is the median value; the value has upper and lower limit protection during automatic calibration. During manual TRIM, this value can be read and written.
7	SW_SYNC	Software synchronization, software writes 1, hardware clears. Note: If the target clock frequency of trim is lower than the system clock, software synchronization cannot be used
6	AUTO_TRIM_EN	Auto fine-tuning enable 1: enable; 0: disable Note: When enabled, the written TRIM value will be synchronized to the hardware automatic TRIM value. If you need to keep the automatic TRIM value at the median value, you must clear the TRIM value in manual TRIM mode.
5	CEN	Frequency Error Counter / Timer Enable
4	CAPIE	Timer mode capture interrupt enable
3	UDFIE	Counter underflow interrupt enable
2	ERRIE	Calibration error interrupt enable
1	WARNIE	Calibration warning interrupt enable
0	OKIE	Calibration OK Interrupt Enable

27.4.2 CTS Configuration Register (CTS_CFGR)

Offset address 0x04

Reset value 0x0022BB7F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLKSRC		REFSRC		POL	DIV			FELIM							
RW		RW		RW	RW			RW							

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

Bit	Marking	Functional description
31:30	CLKSRC	Calibration clock selection in calibration mode Count clock selection in timer mode Reserve the default RCH48M 00： RCH48M 01：RCH 10：XTH 11: RFFCLK (used in timer mode only)
29:28	REFSRC	Calibration synchronization signal source selection, timer mode clock selection or capture source selection 00：USB SOF 01：XTL 10：GPIO 11: LPTIM TOG flip output
27	POL	Sync Polarity Selection 0: Rising edge 1: Falling edge Note: It is invalid when using USB_SOF, USB_SOF is an inversion signal with a period of 2ms, which needs to be converted into a signal with a period of 1ms
26:24	PRS	TRIM sync signal divider / timer clock prescaler 000: no frequency division 001: 2 frequency division 010: 4 frequency divisions 011: 8 frequency division 100: 16 frequency division 101: 32 frequency division 110: 64 frequency division 111: 256 frequency division
23:16	FELIM	Frequency Error Limit
15:0	ARR	Counter reload value

Note: This register can only be changed when CEN is 0.

27.4.3 CTS Interrupt and Status Register (CTS_ISR)

Offset address 0x08

Reset value 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIR	Reserved				OVF	MISS	ERR	Reserved				UDFF	ERRF	WARNF	OKF
RO					RO	RO	RO					RO	RO	RO	RO

Bit	Marking	Functional description
31:16	FECAP	Frequency error capture value
15	DIR	Frequency error direction, showing whether the actual frequency is lower than the target frequency or higher than the target frequency 0: Incremental counting direction, the actual frequency is higher than the target frequency 1: Down counting direction, the actual frequency is lower than the target frequency
14:11	Reserved	Keep
10	OVF	When the TRIM value overflows or underflows, this flag is set to 1 by hardware
9	MISS	The calibration reference synchronization is lost (the frequency is too fast and out of range), and the count value increments to FELIM*128 after counting to 0 and no synchronization signal is detected.
8	ERR	Calibration error (frequency too slow and out of range), the reference sync signal is before the counter overflow and the error is greater than FELIM*128,
7:5	Reserved	Keep
4	CAPF	Catch flag
3	UDFF	Counter underflow flag, when the frequency error counter counts to 0, this bit is set to 1 by hardware.
2	ERRF	Calibration error flag, the result of OVF, MISS, ERR logic OR
1	WARNF	Calibration warning sign, the frequency error is greater than or equal to FELIM*3 and less than FELIM*128
0	OKF	Calibration OK sign. When the measurement frequency error is less than FELIM*3, this flag is set to 1 by hardware.

27.4.4 CTS Interrupt Status Clear Register (CTS_ICR)

Offset address 0x0C

Reset value 0x0000001F

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											CAPC	UDFC	ERRC	WARNC	OKC
											R1W0	R1W0	R1W0	R1W0	R1W0

Bit	Marking	Functional description
31:5	Reserved	Keep
4	CAPC	Timer mode capture flag is cleared, write 0 to clear, write 1 to have no effect.
3	UDFC	The synchronization flag is expected to be cleared. write 0 to clear
2	ERRC	The calibration error flag is cleared, and the OVF, MISS, and ERR flags are cleared at the same time. write 0 to clear
1	WARNC	The calibration warning sign clears. write 0 to clear
0	OKC	Calibration OK flag is cleared. write 0 to clear

28 Audio interface (I2S)

I2S features

- Support Philip / MSB / LSB / PCM mode
- Support MCK output
- Support 48K, 44.1K, 32K, 16K, 8K and other different audio sampling rates
- Support data length 16 bits, 24 bits, 32 bits
- Support frame length 16bit / 32bit
- Support DMA data transfer
- Support full-duplex transmission and reception (2 I2S cooperation)
- Support master sending and receiving
- Support slave sending and receiving
- Shared interrupt and port with SPI

28.1 Functional description

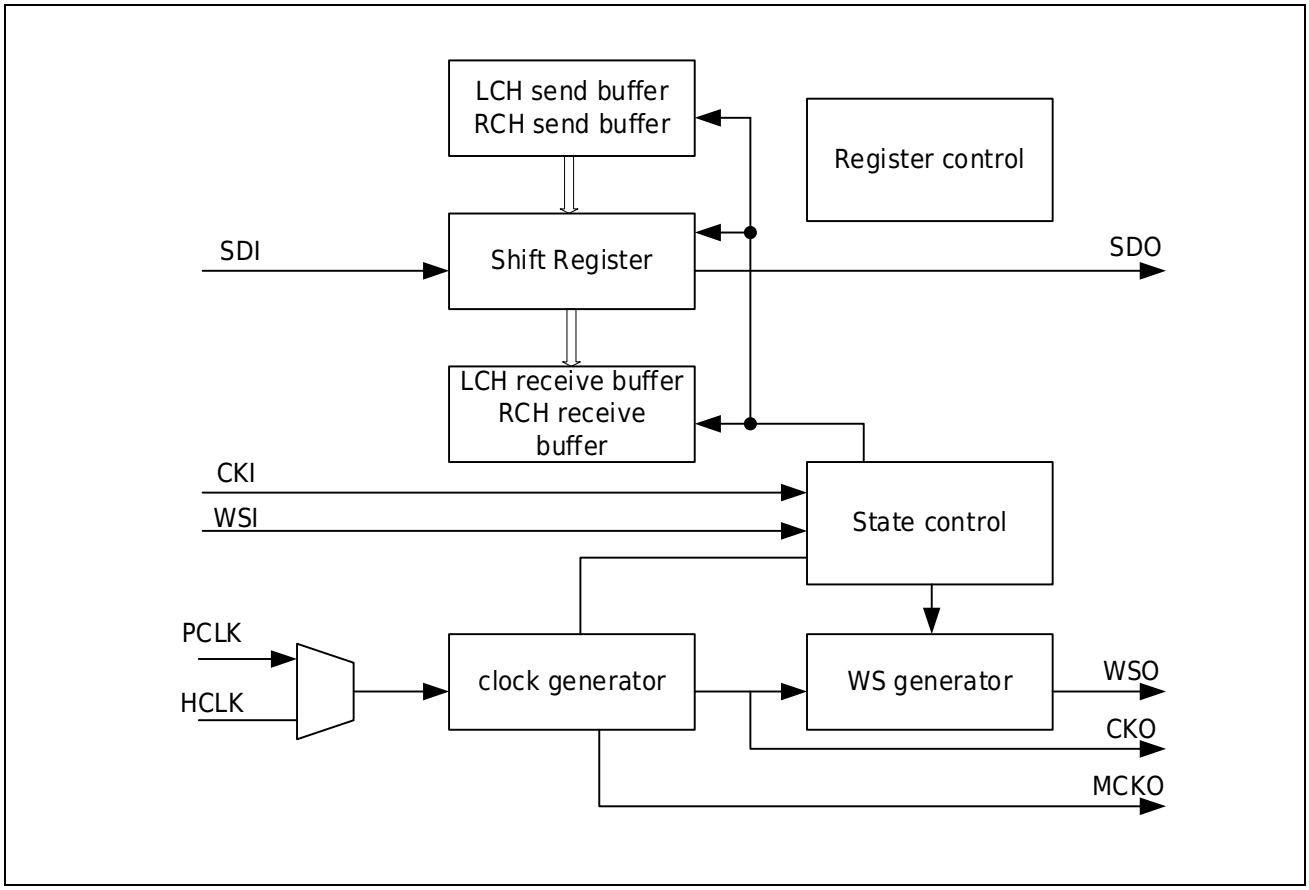


Figure 28-1 Functional block diagram

28.1.1 Pin multiplexing

I2S and SPI IO are multiplexed, and the multiplexing relationship is as follows:

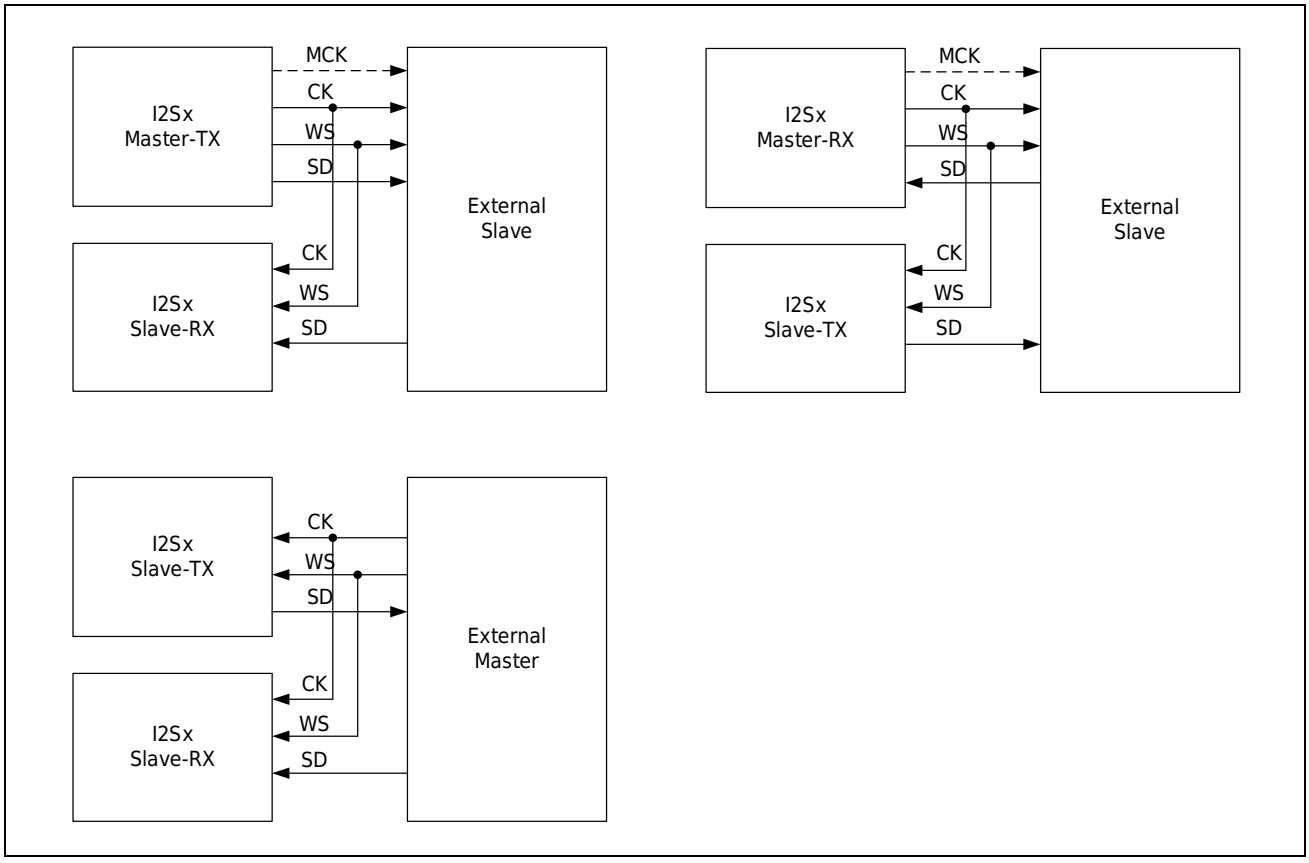
When the I2S function is enabled (the I2S bit in the SYSCTRL_PERI register is set to 1 and the SPI is invalid), the SPI and I2S multiplexed port is used as an audio I2S interface.

I2S	SPI	I2S pin description
SD	MOSI	Serial data (mapped to MOSI pin) for sending or receiving data on two time-division multiplexed data lanes (half-duplex mode only)
WS	NSS	Word Select (mapped to the NSS pin), is the data control signal output in master mode and the data control signal input in slave mode.
CK	SCK	Serial clock (mapped to the SCK pin), is the serial clock output in master mode and the serial clock input in slave mode.
MCK	MISO	Master Clock Output (MCK mapped to the MISO pin), when the I2S is configured in master mode (and the MCKOE bit in the I2Sx_PR register is set), this additional clock is output using the main clock (mapped separately), clocked at a preset rate of 256 × fs Configure frequency generation, where fs is the audio signal sampling frequency.

I2S also has a set of independent IO settings, customers can use I2S and SPI at the same time. In this mode, the interrupt and SPI share the same interrupt entry address, and it is necessary to judge and process the corresponding interrupt service program according to the respective flags.

28.1.2 I2S full duplex

Each I2S supports master mode and slave mode, and supports half-duplex transmission and reception. If full-duplex mode is required, two I2S need to be used for cooperation.



28.2I2S audio protocol description

I2S needs to handle audio data which is usually time multiplexed on two channels left and right. However, there are two 16 -bit registers for transmit or receive. Therefore, the software needs to write the value corresponding to the corresponding channel into the channel data register, and read the corresponding data from the corresponding channel data register. Always send the left channel data first, and then send the right channel data (for the PCM protocol, the right channel data is meaningless).

There are four combinations of data and frame formats, and data can be sent in the following formats:

- Pack 16-bit data in 16-bit frames
- Pack 16-bit data in 32-bit frames
- Pack 24-bit data in 32-bit frames
- Pack 32-bit data in 32-bit frames

When using 16-bit data in a 32-bit packet, only 16 bits of valid data need to be read and written, and 16 bits of invalid data are forced to zero without any software operations or DMA requests (just one read/write operation).

24 -bit and 32 -bit data frames require two CPU read or write operations to the I2Sx_DR register, or two DMA operations. For 24 -bit data frames, the hardware will automatically fill in 8 0s to expand to 32 bits.

For all data formats and communication standards, the most significant bit is always sent first (MSB first).

The I2S interface supports four audio standards, which can be configured using the I2SSTD[1:0] and PCMSYNC bits in the I2Sx_CFGR register.

28.2.1 I2S Philips Standard

Use the WS signal to indicate which channel the data currently being sent belongs to. This signal is valid one clock before the first bit (MSB) of the current channel's data.

The sender changes the data on the falling edge of the clock signal (CK), and the receiver reads the data on the rising edge. WS signal also changes on the falling edge of CK.

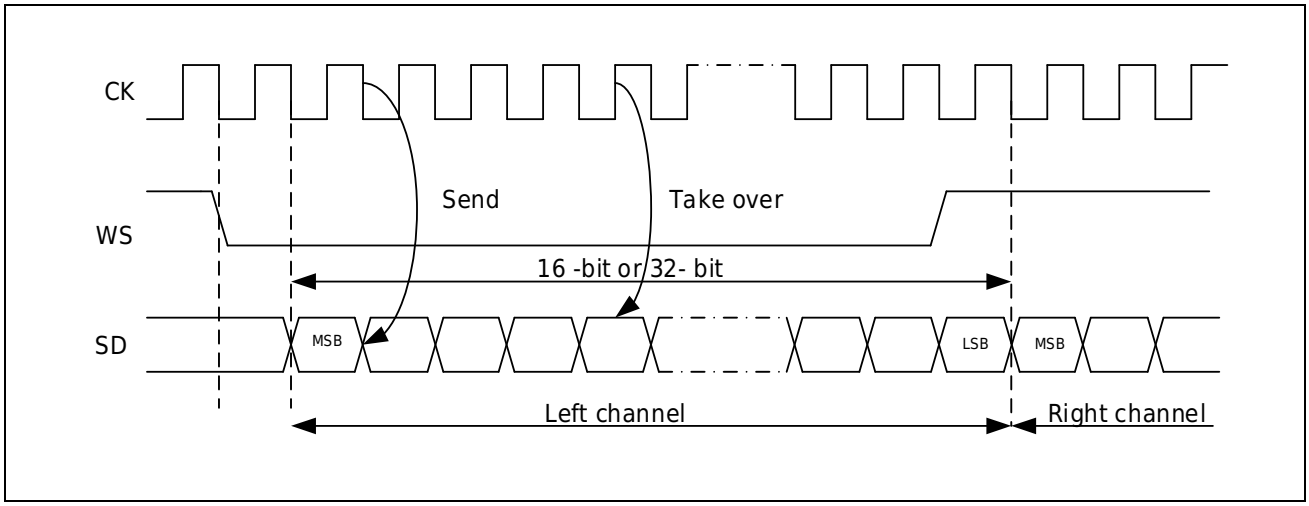


Figure 28-2 PhilipProtocol WaveForm (16/32 bit full precision)

This mode requires two writes or reads to the I2Sx_DRL/I2Sx_DRR registers.

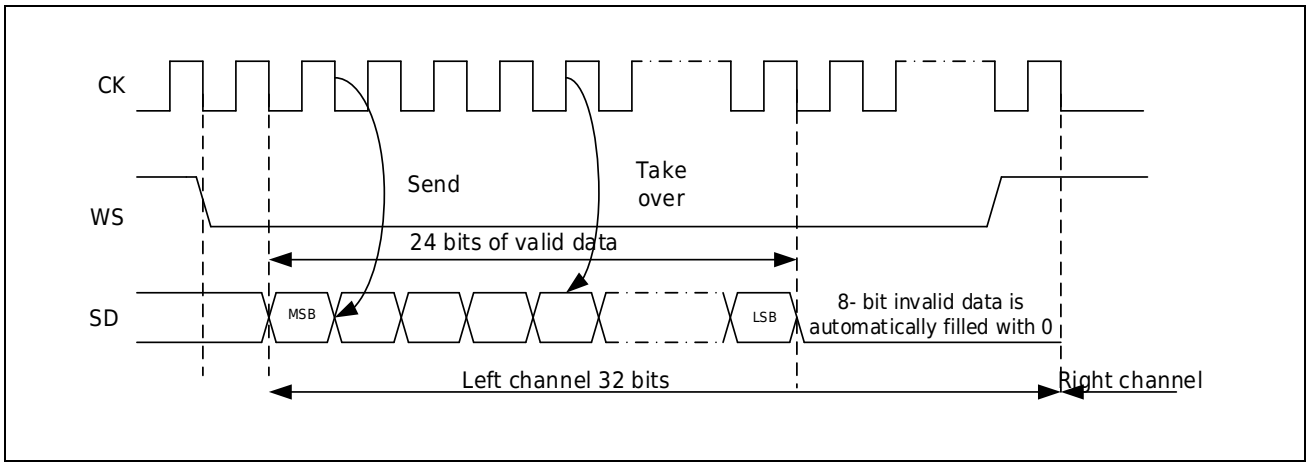


Figure 28-3 Philip Protocol Waveform (24 -bit frame)

This mode requires two writes or reads to the I2Sx_DRL/I2Sx_DRR registers.

In sending mode, if you need to send 0x89ABCD, 0x8FEDCA (24 bits):

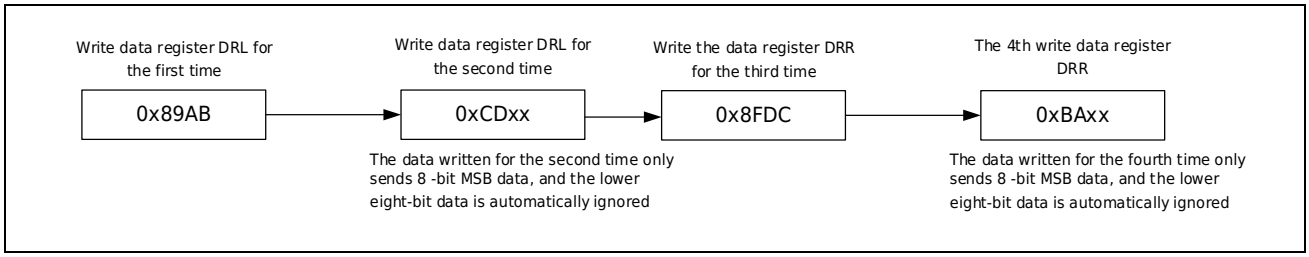


Figure 28-4 24 -Bit Data Transmission Write Operation

In receive mode, if receive data 0x123456,0x89ABCD:

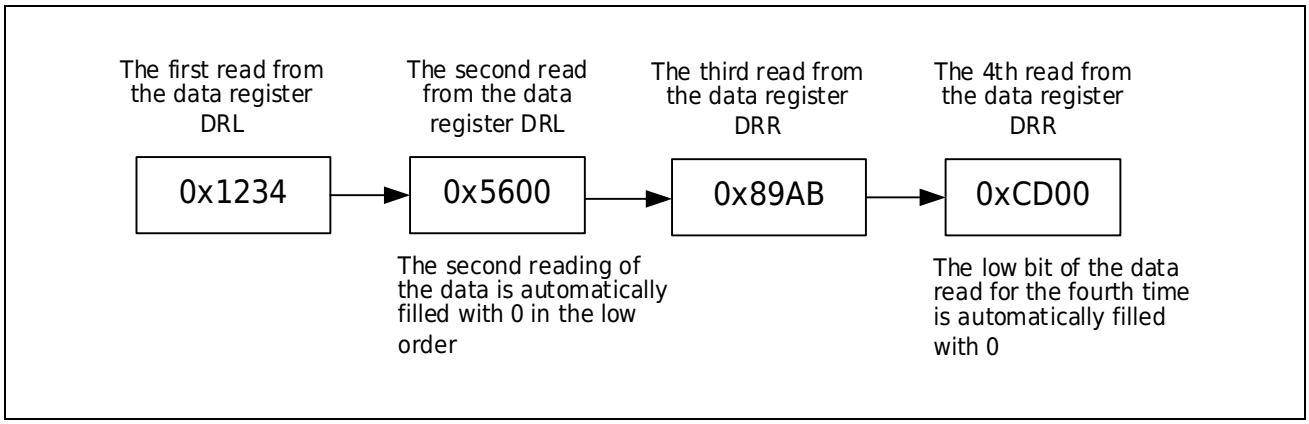


Figure 28-5 24 -Bit Data Mode Receive Read Operation

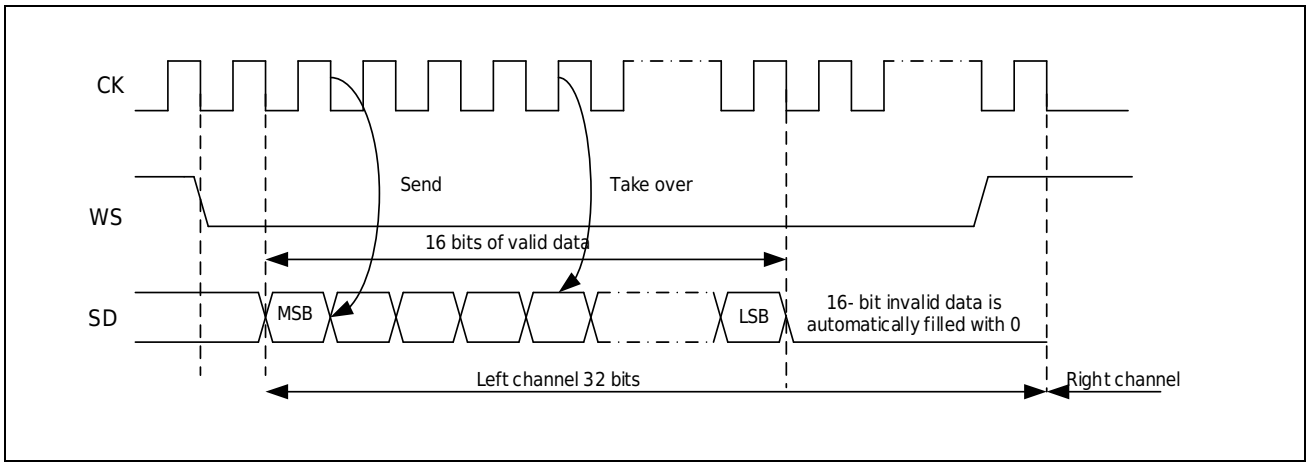


Figure 28-6 Philip Protocol 16 -Bit Extended To 32 -Bit Data Frame

If you choose to extend the 16- bit data frame to a 32 -bit channel frame during the I2S configuration stage, you only need to access the I2Sx_DR register once. The other 16 bits will be automatically filled with 0 by hardware, without software participation.

If the data to be sent or received is 0xCDEF (0xCDEF0000 is extended to 32 bits), you need to perform the operations shown in the figure below.

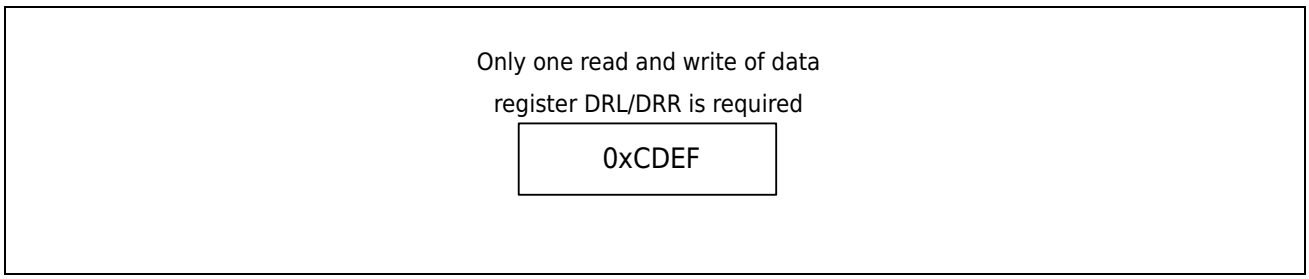


Figure 28-7 16 -Bit DataRead And Write Operations

When sending, each time the MSB is written into the data register I2Sx_DRL/R, after the data is transferred to the shift register, the TXE flag of the corresponding channel will be set to 1, and an interrupt will be triggered when the interrupt is enabled, and the new data to be sent Load into the I2Sx_DR register. This is true even though the lower 16 bits of 0x0000 filled by the hardware have not been sent, because the lower 16 bits are sent by the hardware.

When receiving, the first half word (high 16 bits) is received, the hardware will set the RXNE flag of the corresponding channel to 1, and trigger an interrupt if the interrupt is enabled.

28.2.2 MSB Alignment Standard

This standard generates the WS signal and the first data bit at the same time, the WS left channel corresponds to a high level, and the right channel corresponds to a high level.

The sender changes the data on the falling edge of the clock signal (CK), and the receiver reads the data on the rising edge. WS signal also changes on the falling edge of CK.

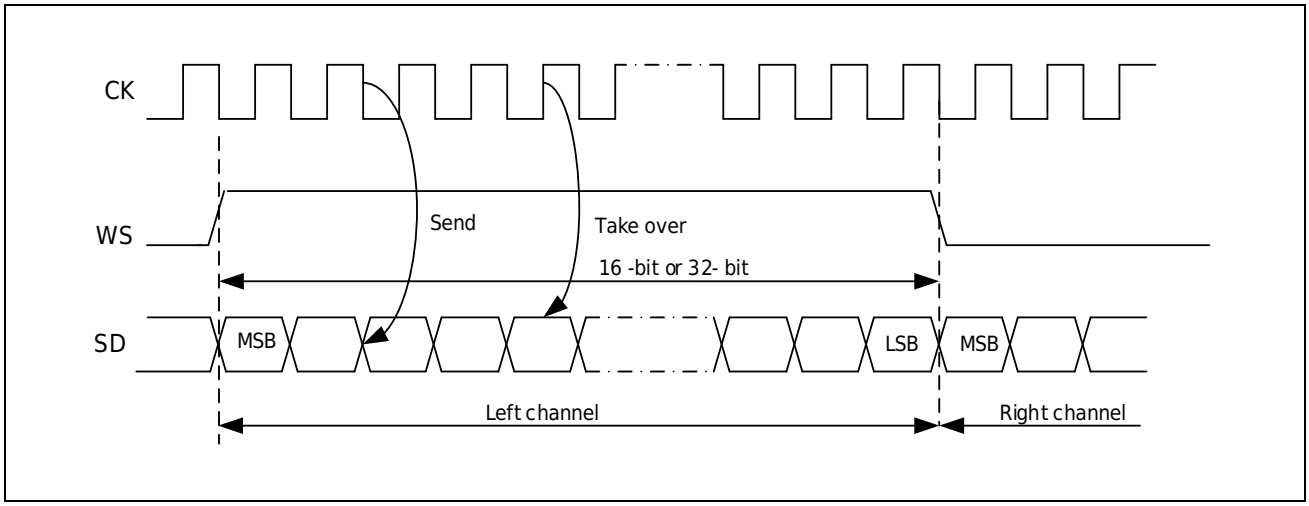


Figure 28-8 MSB protocol waveform (16/32 bit full precision)

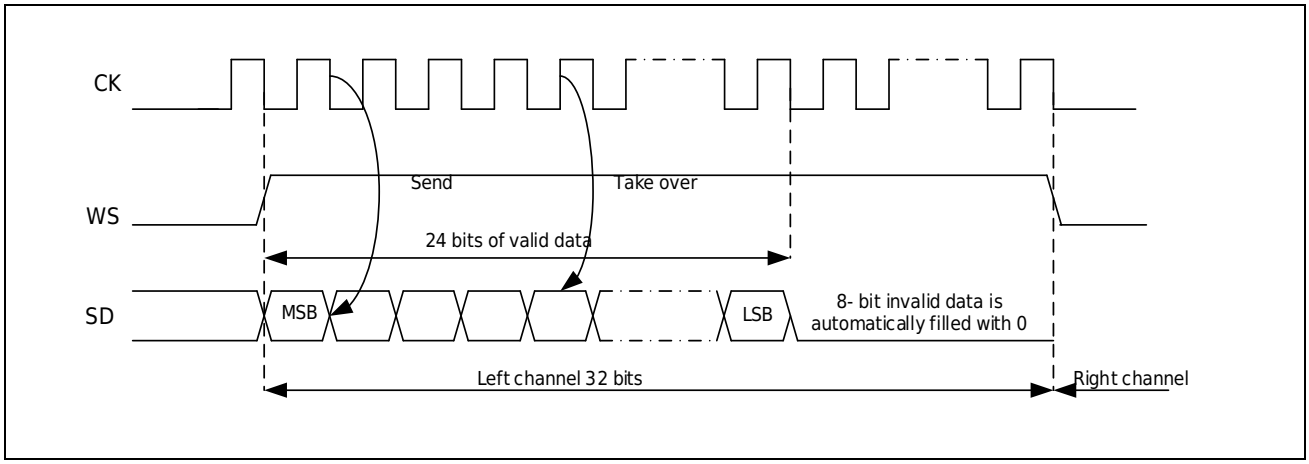


Figure 28-9 MSB protocol waveform (24-bit frame)

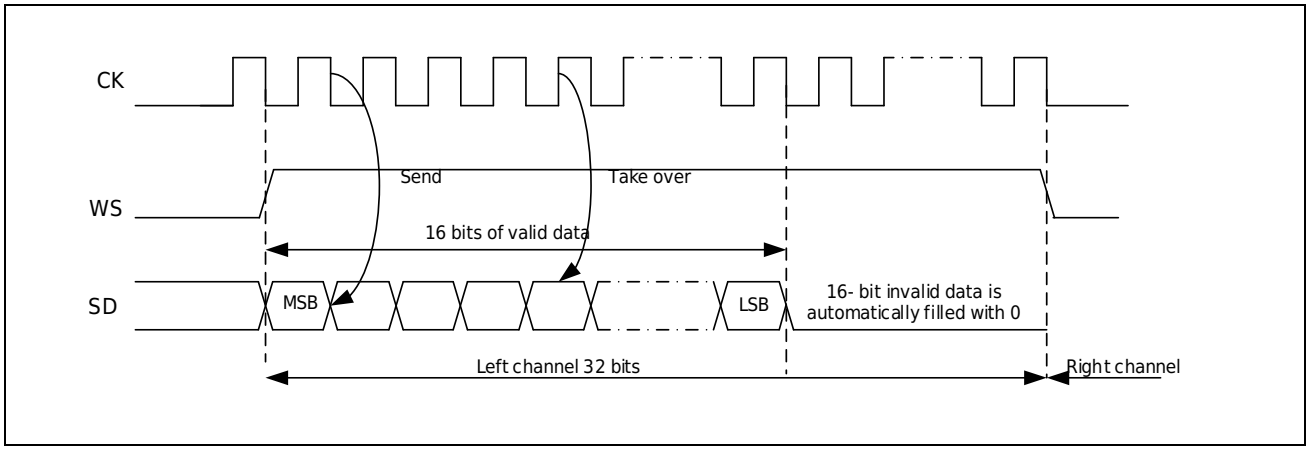


Figure 28-10 MSB Protocol 16 -Bit Extended To 32 -Bit Data Frame

28.2.3 LSB Alignment Standard

This standard is similar to the MSB alignment standard (for 16-bit and 32-bit full-precision frame formats, there is no difference).

The sender changes the data on the falling edge of the clock signal (CK), and the receiver reads the data on the rising edge. WS signal also changes on the falling edge of CK.

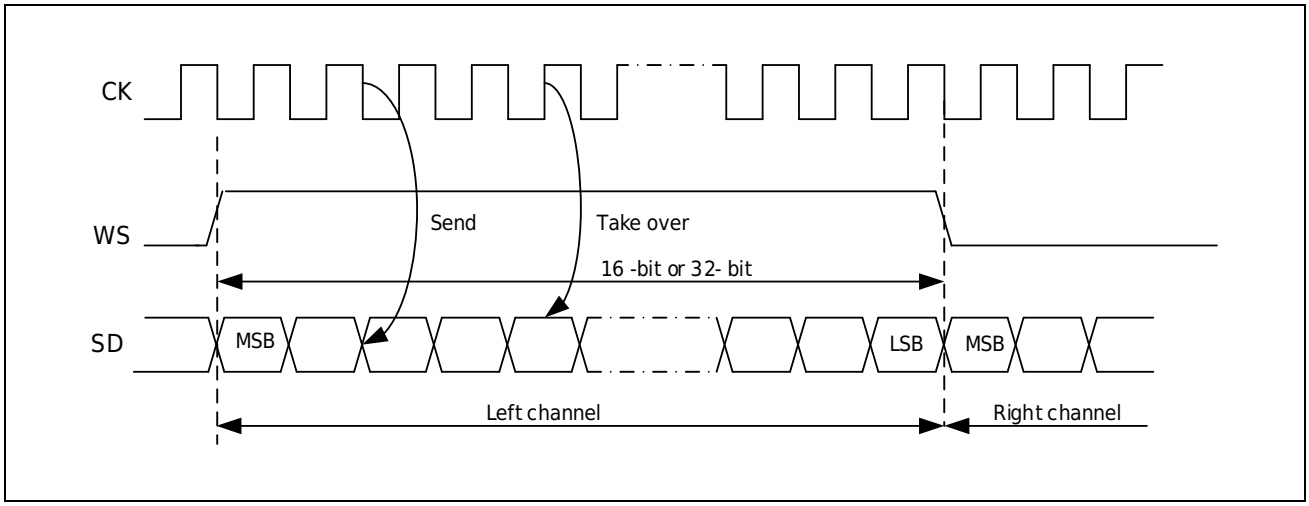


Figure 28-11 LSB protocol waveform (16/32 bit full precision)

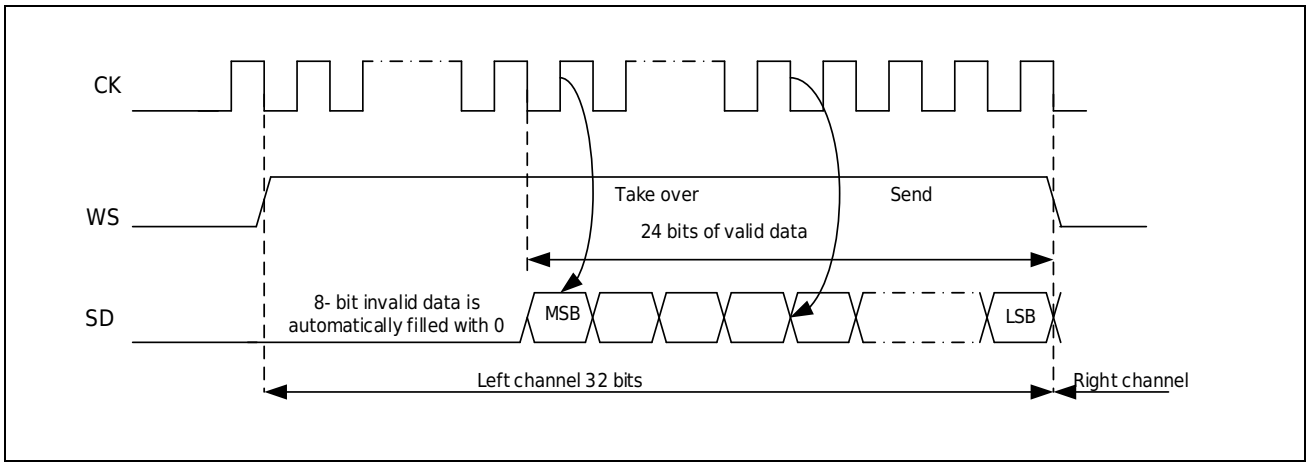


Figure 28-12 LSB protocol waveform (24-bit frame)

In transmit mode, if you need to send data 0x89ABCD, you need to perform two write operations to the I2Sx_DRL/I2Sx_DRR registers by software or DMA. These operations are given below.

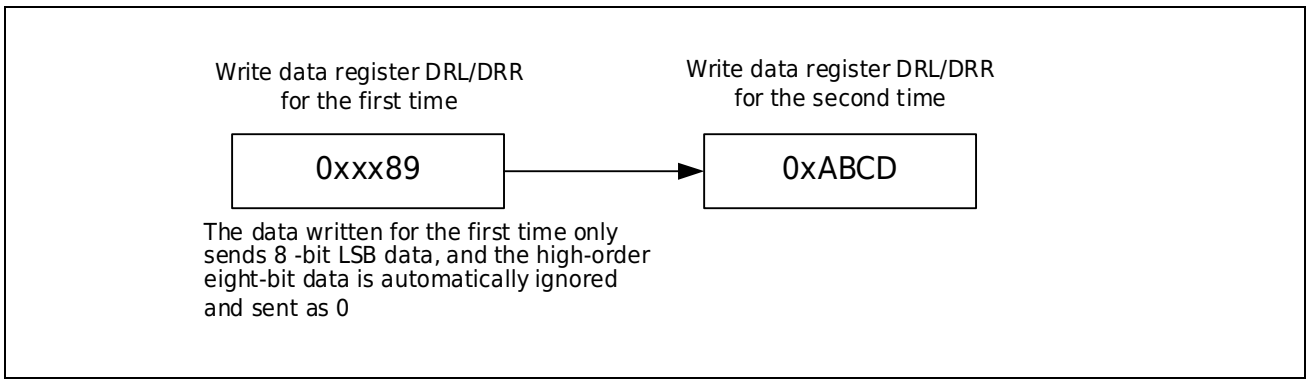


Figure 28-13 LSB 24 -Bit Data Transmission Write Operation

In receive mode, if data 0x123456 is received, two consecutive read operations to the I2Sx_DR register are required on each RXNE event.

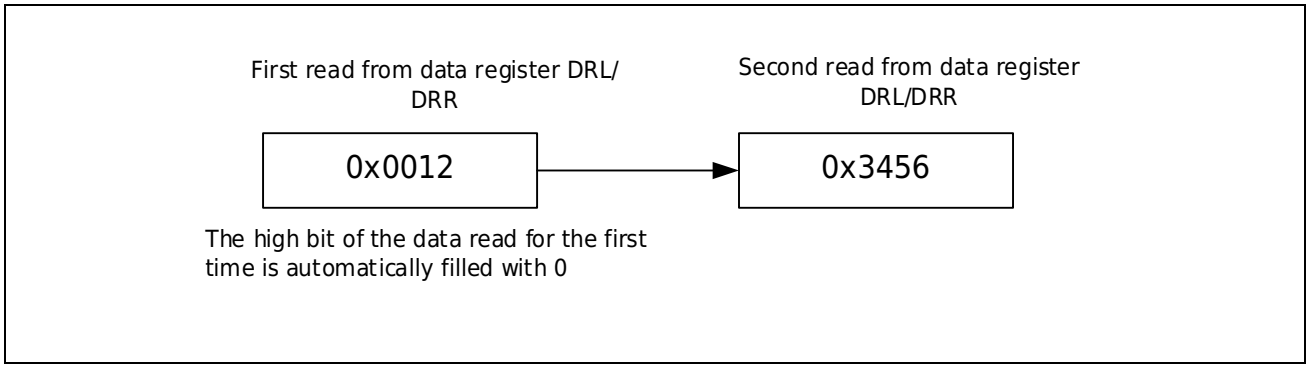


Figure 28-14 LSB 24 -Bit Data Mode Receive Read Operation

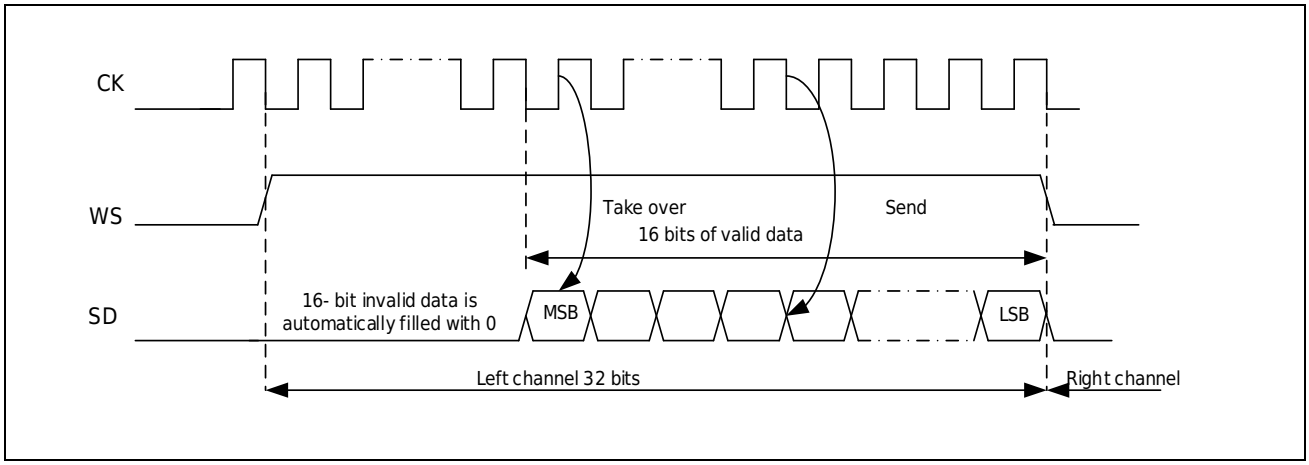


Figure 28-15 LSB Protocol 16 -Bit Extended To 32 -Bit Data Frame

If you choose to extend the 16- bit data frame to a 32 -bit channel frame during the I2S configuration phase, you only need to access the I2Sx_DRL/I2Sx_DRR register once. Extended to 32 bits, the upper half word (16 bits MSB) is set to 0x0000 by hardware.

If the data to be sent or received is 0x7893 (0x0000 7983 is extended 32 -bit data), you need to perform the operations shown below.

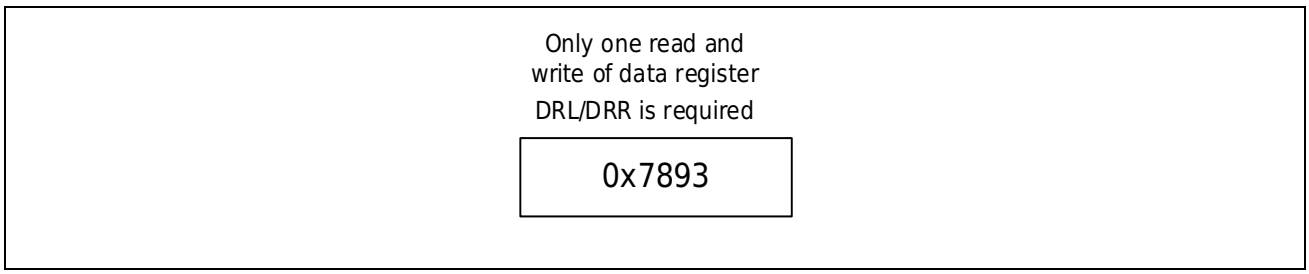


Figure 28-16 16 -Bit DataRead And Write Operations

In the send mode, when a TXE event occurs on the corresponding channel, the application program needs to write the data to be sent on the corresponding channel (in this example, it is 0x7893).

The 0x0000 field (extended to 32 bits) is sent first. After the valid data (0x7893) of the corresponding channel is transferred to the shift register, the TXE flag of the corresponding channel will be set to 1 again.

In receive mode, when a valid halfword is received (instead of the 0x0000 field), the RXNE of the corresponding channel is set.

28.2.4 PCM standard

For the PCM standard, there is no need to use channel information. Two PCM modes are available (Short Frame and Long Frame) and can be configured using the PCMSYNC bit in the I2Sx_CFGR register. PCM uses the rising edge to send data and the falling edge to receive data.

Sending data and receiving data only needs to use the I2Sx_DRL data register, and the I2Sx_DRR data register is invalid in this mode.

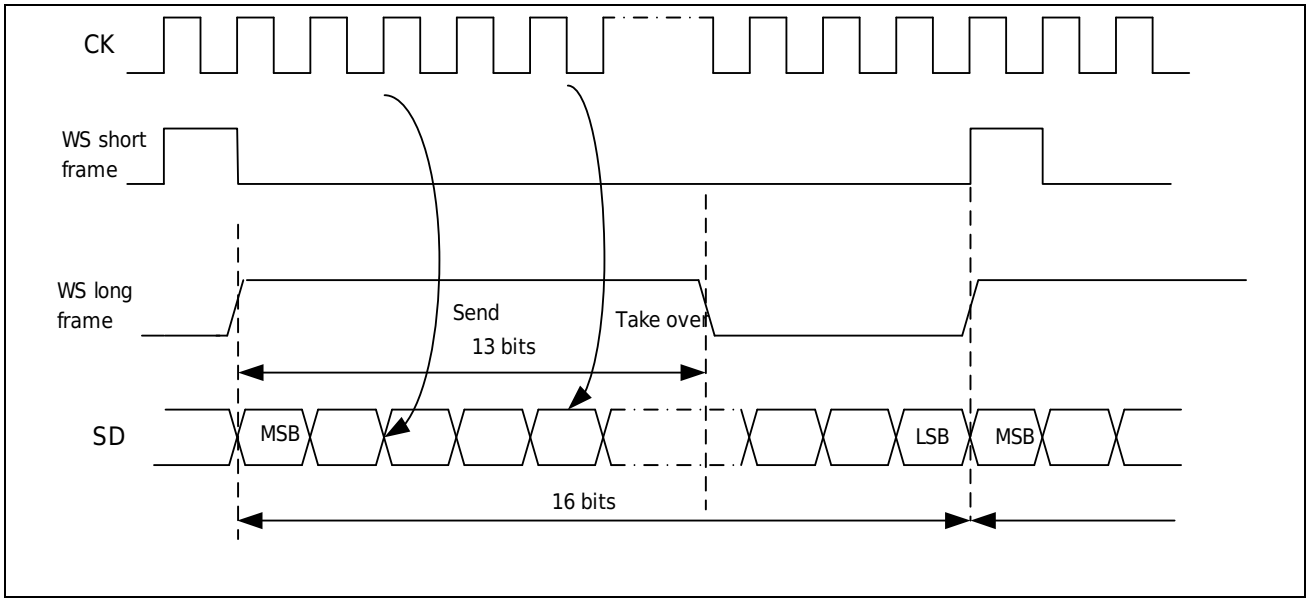


Figure 28-17 PCM protocol waveform (16 bits)

For long frame sync, the WS signal is maintained for 13 cycles in master mode.

For short frame sync, the duration of the WS sync signal is only one cycle.

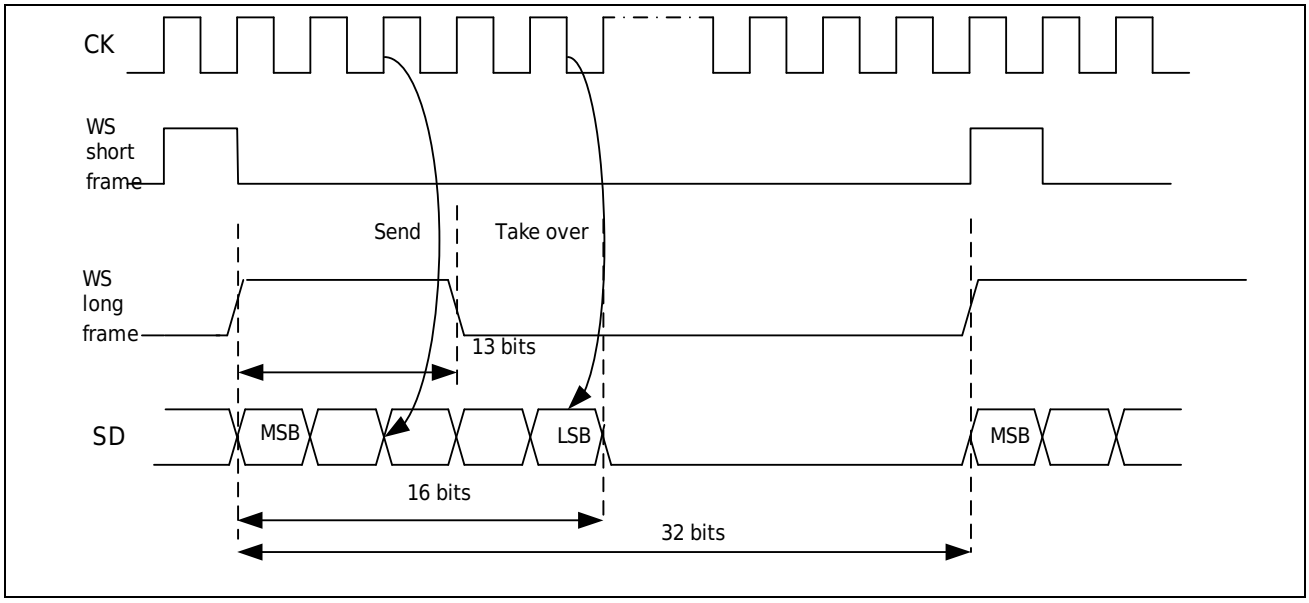


Figure 28-18 PCM Protocol Waveform (16 -bit data extended to 32 -bit)

Note:

– For two modes (host/slave mode) and two synchronizations (short/long synchronization), even in slave mode, you need to specify the number of bits between two sets of continuous data (and two synchronization signals) (I2Sx_CFGR register) DATLEN bit and CHLEN bit in).

28.3I2S clock generator

The I2S bit rate is used to determine the data flow on the I2S data line and the I2S clock signal frequency.

I2S bit rate = number of bits per channel × number of channels × audio sampling frequency

For 16 -bit dual-channel audio, the calculation formula for the I2S bit rate is as follows:

I2S bit rate = 16 x 2 x fS

If the packet is 32 bits wide, then I2S bitrate = 32 x 2 x fS.

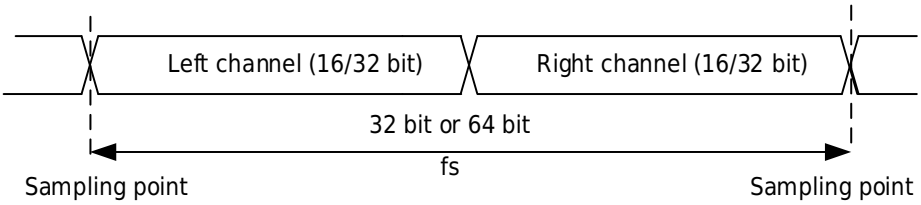


Figure 28-19 Audio Sampling Frequency Definition

When configuring master mode, the linear crossover needs to be set up correctly to communicate at the desired audio frequency.

The figure below shows the communication clock architecture. I2Sx clock can choose PCLK or HCLK.

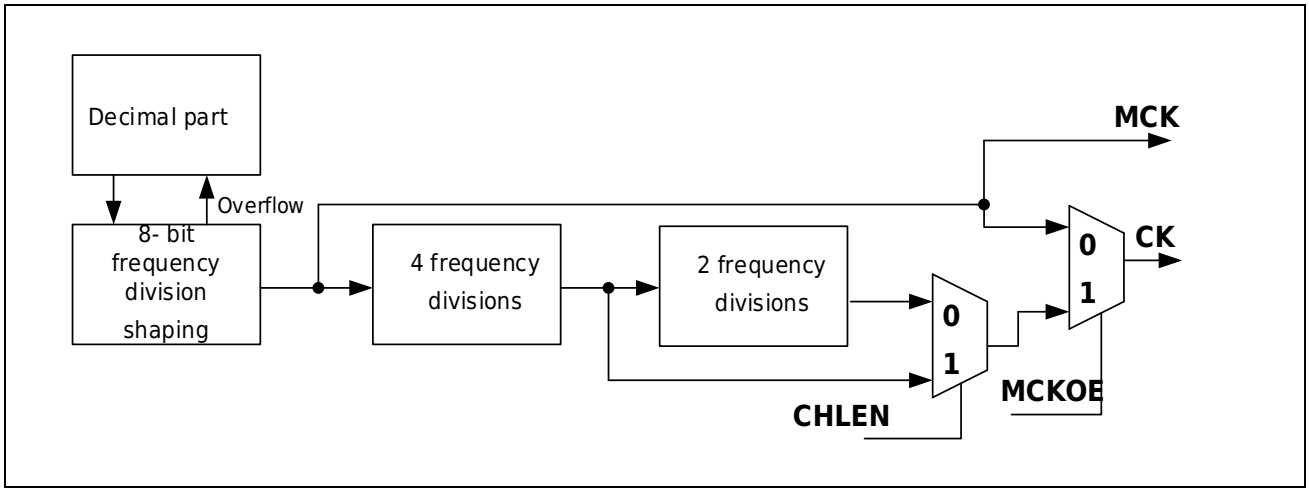


Figure 28-20 Clock Generation

The audio sampling frequency can be 96 kHz, 48 kHz, 44.1 kHz, 32 kHz, 22.05 kHz, 16 kHz, 11.025 kHz, or 8 kHz (or any other value within this range). To achieve the desired frequency, the linear divider needs to be programmed according to the following formula:

When the fractional frequency division FRACT is equal to 0x00:

When outputting the master clock (MCKOE in the I2Sx_PR register is set to 1):

fS = I2SxCLK / [(16*2)*((2*I2SDIV)+ODD)*8)] (when the channel frame width is 16 bits)

fS = I2SxCLK / [(32*2)*((2*I2SDIV)+ODD)*4)] (when the channel frame width is 32 bits)

master clock output is disabled (MCKOE bit is cleared):

fS = I2SxCLK / [(16*2)*((2*I2SDIV)+ODD)] (when the channel frame width is 16 bits)

fS = I2SxCLK / [(32*2)*((2*I2SDIV)+ODD)] (when the channel frame width is 32 bits)

the fractional frequency FRACT is not equal to 0x00:

When outputting the master clock (MCKOE in the I2Sx_PR register is set to 1):

fS = I2SxCLK /{ (16*2)* [2*(I2SDIV+FRACT/64)]*8} (when the channel frame width is 16 bits)

fS = I2SxCLK /{ [(32*2)*[2*(I2SDIV+FRACT/64)]*4} (when the channel frame width is 32 bits)

master clock output is disabled (MCKOE bit is cleared):

fS = I2SxCLK /{ (16*2)* [2*(I2SDIV+FRACT/64)]} (when the channel frame width is 16 bits)

fS = I2SxCLK /{ (32*2)* [2*(I2SDIV+FRACT/64)]} (when the channel frame width is 32 bits)

The following table provides the settings and accuracy values of the clock 48M clock for different sampling rates (without fractional frequency division).

I2S CLK	Channel Length	I2SDIV	I2SODD	MCLK	Sampling rate	Actual Frequency	Error%	freq(mck)
48000000	16	8	0	No output	96000	93750	-2.34	0
48000000	32	4	0	No output	96000	93750	-2.34	0
48000000	16	15	1	No output	48000	48387	0.81	0
48000000	32	8	0	No output	48000	46875	-2.34	0
48000000	16	17	0	No output	44100	44118	0.04	0
48000000	32	8	1	No output	44100	44118	0.04	0
48000000	16	23	1	No output	32000	31915	-0.27	0
48000000	32	11	1	No output	32000	32609	1.9	0
48000000	16	34	0	No output	22050	22059	0.04	0
48000000	32	17	0	No output	22050	22059	0.04	0
48000000	16	47	0	No output	16000	15957	-0.27	0
48000000	32	23	1	No output	16000	15957	-0.27	0
48000000	16	68	0	No output	11025	11029	0.04	0
48000000	32	34	0	No output	11025	11029	0.04	0
48000000	16	94	0	No output	8000	7979	-0.26	0
48000000	32	47	0	No output	8000	7979	-0.26	0
48000000	16	2	0	Output	48000	46875	-2.34	12000000
48000000	32	2	0	Output	48000	46875	-2.34	12000000
48000000	16	2	0	Output	44100	46875	6.29	12000000
48000000	32	2	0	Output	44100	46875	6.29	12000000
48000000	16	3	0	Output	32000	31250	-2.34	8000000
48000000	32	3	0	Output	32000	31250	-2.34	8000000
48000000	16	4	1	Output	22050	20833	-5.52	5333248
48000000	32	4	1	Output	22050	20833	-5.52	5333248
48000000	16	6	0	Output	16000	15625	-2.34	4000000
48000000	32	6	0	Output	16000	15625	-2.34	4000000
48000000	16	8	1	Output	11025	11029	0.04	2823424
48000000	32	8	1	Output	11025	11029	0.04	2823424
48000000	16	11	1	Output	8000	8152	1.9	2086912
48000000	32	11	1	Output	8000	8152	1.9	2086912

The following table provides the settings and accuracy values of the clock 48M clock using **fractional frequency division** for different sampling rates.

I2S CLK	Channel Length	I2SDIV	FRACT	MCLK	Sampling rate	Actual Frequency	Error%	freq(mck)
48000000	16	7	52	No output	96000	96000	0	0
48000000	32	3	58	No output	96000	96000	0	0
48000000	16	15	40	No output	48000	48000	0	0
48000000	32	7	52	No output	48000	48000	0	0
48000000	16	17	0	No output	44100	44118	0.04	0
48000000	32	8	32	No output	44100	44118	0.04	0
48000000	16	23	28	No output	32000	32000	0	0
48000000	32	11	46	No output	32000	32000	0	0
48000000	16	34	1	No output	22050	22049	0	0
48000000	32	17	0	No output	22050	22059	0.04	0
48000000	16	46	56	No output	16000	16000	0	0
48000000	32	23	28	No output	16000	16000	0	0
48000000	16	68	2	No output	11025	11024	-0.01	0
48000000	32	34	1	No output	11025	11024	-0.01	0
48000000	16	93	48	No output	8000	8000	0	0
48000000	32	46	56	No output	8000	8000	0	0
48000000	16	1	61	Output	48000	48000	0	12288000
48000000	32	1	61	Output	48000	48000	0	12288000
48000000	16	2	8	Output	44100	44118	0.04	11294208
48000000	32	2	8	Output	44100	44118	0.04	11294208
48000000	16	2	60	Output	32000	31915	-0.27	8170240
48000000	32	2	60	Output	32000	31915	-0.27	8170240
48000000	16	4	16	Output	22050	22059	0.04	5647104
48000000	32	4	16	Output	22050	22059	0.04	5647104
48000000	16	5	55	Output	16000	16000	0	4096000
48000000	32	5	55	Output	16000	16000	0	4096000
48000000	16	8	32	Output	11025	11029	0.04	2823424
48000000	32	8	32	Output	11025	11029	0.04	2823424
48000000	16	11	46	Output	8000	8000	0	2048000
48000000	32	11	46	Output	8000	8000	0	2048000

28.4I2S working mode

28.4.1 I2S host mode

I2S can be configured as master mode. This means that the serial clock will be output on the CK pin and the word select signal will be generated on the WS pin. The main clock (MCK) can be output or not, which is controlled by the MCKOE bit in the I2Sx_PR register.

Step:

1. Set the I2SDIV[7:0] bits of the I2Sx_PR register to define the serial clock baud rate to achieve the corresponding audio sampling frequency. The ODD and FRACT bits of the I2Sx_PR register are set as required.
2. If the master clock MCK is required for the external DAC/ADC audio components, set the MCKOE bit of the I2Sx_PR register to 1 (I2SDIV and ODD (FRACT)) The value should be calculated based on the state of the MCK output. See Clock Generators for details.
3. Select the I2S standard through the I2SSTD[1:0] and PCMSYNC bits, select the data length through the DATLEN[1:0] bits and select the number of bits per channel by configuring the CHLEN bits. Additionally, the I2S master mode and direction (transmitter or receiver) are selected by the I2SCFG[1:0] bits of the I2Sx_CFGR register.
4. If desired, all possible interrupt sources and DMA functions are selected by writing to the I2Sx_CR2 register.
5. The I2SE bit of the I2Sx_CFGR register must be set to 1.

The IO of WS and CK is configured as output mode. If the MCKOE bit of I2Sx_PR is set, MCK also needs to be configured as an output.

Send sequence

The transmit sequence begins as soon as the halfword is written to the transmit buffer.

Assume that the first data of the left channel written to the sending buffer is written into the data register L; the first data of the right channel written into the sending buffer is written into the data register R. When data is transferred from the transmit buffer to the shift register, TXE of the corresponding channel is set to 1, and the data corresponding to the corresponding channel must be written to the transmit buffer of the corresponding channel.

A complete frame means that the left channel data is sent first and then the right channel data is sent. There are no partial frames where only the left channel is sent.

During first-bit transmission, data is loaded into the 16-bit shift register in parallel as a halfword, then shifted in serial and output on the MOSI/SD pin (MSB first). Every time data is transferred from the transmit buffer to the shift register, the TXE flag of the corresponding channel will be set to 1, and an interrupt will be generated if the TXEIE bit of the I2Sx_CR2 register is set to 1.

See the I2S Audio Protocol Description for more details on write operations in various I2S standard modes.

To ensure continuous audio data transmission, the next data to be transmitted must be written into the I2Sx_DRL/DRR register before the current data transmission ends.

To turn off I2S by clearing I2SE, one must wait for TXE = 1 and BSY = 0.

Receive sequence

This working mode is the same as the sending mode, only the 3rd point is different (please refer to the steps described in the I2S master mode), that is, the master device receiving mode is set by the I2SCFG[1:0] bits.

Audio data is always received in 16-bit packets regardless of data or channel length. This means that whenever the corresponding channel buffer has received data, the RXNE flag of the corresponding channel is set to 1, and if the RXNEIE bit of the I2Sx_CR2 register is set to 1, an interrupt will also be generated. Received audio values of the right or left channel can be entered into the corresponding channel's receive buffer with one or two receive operations, depending on the data and channel length configuration.

Reading the I2Sx_DRL/DRR register clears the RXNE bit of the corresponding channel.

See the I2S Audio Protocol Description for more details on read operations in various I2S standard modes.

If new data is received while previously received data has not been read, an overflow error will be generated and the OVR flag will be set. The value of the data reception register keeps the previous value, and the currently received data is lost.

If the ERRIE bit of the I2Sx_CR2 register is set, an interrupt will be generated to indicate the error.

To turn off I2S, specific actions need to be taken to ensure that I2S completes transfer cycles correctly without initiating new data transfers. The sequence depends on the configuration of data and channel lengths, and the selected audio protocol mode. In the following cases:

Extended 16 -bit data length on 32 -bit channel length (DATLEN = 00 and CHLEN = 1), using LSB-aligned mode (I2SSTD = 10)

a) Wait for the penultimate RXNE = 1 (n – 1)
b) Then wait 17 I2S clock cycles (using software loop)
c) Disable I2S (I2SE = 0)

16 -bit data length extended on 32 -bit channel length (DATLEN = 00 and CHLEN = 1), using MSB aligned, I2S or PCM mode (I2SSTD = 00, I2SSTD = 01 or I2SSTD = 11, respectively)

a) Waiting for the last RXNE
b) Then wait 1 I2S clock cycles (using software loop)

c) Disable I2S (I2SE = 0)

For all other combinations of DATLEN and CHLEN, regardless of the audio mode selected by the I2SSTD bits, the following sequence will be executed to turn off I2S:

a) Wait for the penultimate RXNE = 1 (n – 1)
b) Then wait for one I2S clock cycle (using a software loop)
c) Disable I2S (I2SE = 0)

Note: During transmission, the BSY flag remains low.

28.4.2 I2S slave mode

For slave configuration, I2S can be configured in transmit mode or receive mode.

The rules followed by this working mode are basically the same as the I2S master mode configuration. In slave mode, the I2S interface does not generate a clock.

Clock and WS signals are input from an external master device connected to the I2S interface. This way, the user does not need to configure the clock.

The following configuration steps should be followed:

1. Select the I2S standard by the I2SSTD[1:0] bits, select the data length by the DATLEN[1:0] bits and select the number of bits for each channel in the frame by configuring the CHLEN bits. In addition, the mode of the slave device (transmit or receive) is selected by the I2SCFG[1:0] bits of the I2Sx_CFGR register.
2. If desired, all possible interrupt sources and DMA functions are selected by writing to the I2Sx_CR2 register.
3. The I2SE bit of the I2Sx_CFGR register must be set to 1.

Send sequence

The transmit sequence starts when the external master sends a clock and requests data transfer through the I2Sx_WS signal. The slave must be enabled before the external master can begin communication. The I2S data register must also be loaded before the master device can start communication.

For I2S, MSB alignment and LSB alignment modes, the first data to be sent is the left channel data, and the corresponding channel data is written in the left and right channel data registers respectively. When communication starts, data is transferred from the transmit buffer to the shift register. TXE flag of the corresponding channel is then set to 1 to request that the next data of the corresponding channel be written into the I2S corresponding channel data register.

The slave needs to be ready to send the first data before the master can generate the clock. Setting WS means that the left channel data is sent first.

Note: I2SE must be asserted at least 2 PCLK cycles before the first clock from the master appears on the CK line.

During first bit transmission, data is loaded in parallel from the internal bus as a halfword into the 16 -bit shift register, then shifted in serial and output to the MOSI/SD pin (MSB first). Every time data is transferred from the transmit buffer to the shift register, the TXE flag of the corresponding channel will be set to 1, and an interrupt will be generated if the TXEIE bit of the I2Sx_CR2 register is set to 1.

Note that writing to the transmit buffer can only be attempted when the TXE flag is 1.

I2S Audio Protocol Description for more details on write operations in the various I2S standard modes.

In order to ensure continuous audio data transmission, the next data to be sent must be written into the I2Sx_DRL/R register before the end of the current data transmission. If the first clock edge of the next data communication arrives before data has been written to the I2Sx_DR register, the underflow flag will be set and an interrupt may be generated. In this way, the software is informed that the transmitted data is incorrect. If the ERRIE bit of the I2Sx_CR2 register is set, an interrupt will be generated when the UDR flag in the I2Sx_SR register becomes 1. In this case, I2S must be turned off and the data transfer restarted from the left channel.

To turn off I2S by clearing the I2SE bit, one must wait for TXE = 1 and BSY = 0.

Receive sequence

This working mode is the same as the sending mode, only the first point is different (see the steps described in the I2S slave mode), that is, the slave device receiving mode is set through the I2SCFG[1:0] bits of the I2Sx_CFGR register.

Regardless of data length or channel length, audio data is always received in 16 -bit packets. This means that whenever the receive buffer has data, the RXNE flag of the corresponding channel in the I2Sx_SR register is set, and an interrupt will also be generated if the RXNEIE bit of the I2Sx_CR2 register is set. The received audio value of the right channel or left channel may enter the receive buffer of the corresponding channel through one or two receive operations, depending on the data length and channel length configuration.

Each time the received data is read from the data register of the corresponding channel, reading the I2Sx_DR register will clear the RXNE bit.

See the I2S Audio Protocol Description for more details on read operations for the various I2S standard modes.

If new data is received while previously received data has not been read, an overflow error will be generated and the OVR flag will be set. If the ERRIE bit of the I2Sx_CR2 register is set, an interrupt will be generated to indicate the error.

To turn off I2S in receive mode, I2SE must be cleared immediately after receiving the last RXNE = 1.

Note: The external master should be able to send / receive data in 16 -bit or 32 -bit packets over the audio channel.

28.5I2S Status Flags

The application program can fully monitor the status of the I2S bus through three status flags.

28.5.1 Busy flag (BSY)

BSY flag is set and cleared by hardware (writing to this flag has no effect). This flag indicates the status of the I2S communication layer.

BSY is set to 1, it means that I2S is busy communicating. In master receive mode (I2SCFG = 11), the exception is the BSY flag, which remains low during reception.

If software needs to disable I2S, the BSY flag can be used to detect the end of the transfer. This avoids corrupting the last transmitted data. To do this, the steps described below must be strictly followed.

At the beginning of the transfer (except when I2S is in master receiver mode), set the BSY flag to 1

The BSY flag is cleared by hardware when:

- When the transfer is complete (except for master send mode, where communication is continuous)
- When I2S is disabled

When communication continues:

- In master transmit mode, the BSY flag remains high during all transmissions
- In slave mode, the BSY flag goes low between each transfer for one I2S clock cycle

Note: Do not use the BSY flag to process each data transmission or reception, it is better to use the TXE flag and the RXNE flag instead.

28.5.2 Send buffer is empty (TXE_L TXE_R)

If this flag is set to 1, it means that the send buffer is empty and the data to be sent can be loaded into it. TXE flag is reset when the transmit buffer already contains data to be transmitted. This flag is also reset when I2S is disabled (I2SE bit reset).

28.5.3 Receive buffer is not empty (RXNE_L RXNE_R)

When this flag is set, it indicates that there is valid received data in the receive buffer. This flag is reset when the I2Sx_DRL/R register is read.

28.5.4 Underflow flag (UDR)

In transmit mode, this flag will be set if the first data transmit clock occurs before software has loaded any value into I2Sx_DR. An interrupt can be generated if the ERRIE bit in the I2Sx_CR2 register is set.

The UDR bit is cleared by I2Sx_ICLR.

28.5.5 Overflow flag (OVR)

This flag will be set if new data is received when the previous data has not been read from the I2Sx_DR register. Therefore, incoming data will be lost. An interrupt can be generated if the ERRIE bit in the I2Sx_CR2 register is set.

In this case, the contents of the receive buffer will not be updated with new data received. Execution of the I2Sx_DR register

A read operation will return data previously received correctly. All other halfwords subsequently transmitted by the master will be lost.

To clear the OVR bit, first perform a read operation on the I2Sx_DR register, followed by a read access to the I2Sx_SR register.

28.5.6 Framing Error Flag (FRE)

This flag can be set by hardware only when the I2S is configured in slave mode. This flag will be set if the external master does not change the WS line as the slave expects. If synchronization is lost, recover from this state and connect the external master with the

To resynchronize the I2S slave, perform the following steps:

1. Disable I2S.
2. It is re-enabled when the correct level is detected on the WS line (WS line is high in I2S mode, low in MSB- aligned, LSB- aligned, or PCM modes).

Synchronization failure between master and slave may be due to noise interference on SCK communication clock or WS frame sync signal line. An error interrupt can be generated if the ERRIE bit is set. The sync failure flag (FRE) is cleared by software operation I2Sx_ICLR.

28.6I2S Interrupt DMA

The following table is the I2S interrupt list

Interrupt event	Event flag	enable control
Send buffer empty	TXE_L TXE_R	TXEIE
Receive buffer is not empty	RXNE RXNE_R	RXNEIE
Overflow error	OVR	ERRIE
Underflow error	UDR	
Frame error	FRE	

DMA features

In I2S mode, the data sent and received can be transferred using DMA.

28.7I2S register

Base address

I2S0: 0x4000 2800

I2S1: 0x4000 2C00

Table 28-1 I2Sx Register

Register	Offset address	Description
I2Sx_CR	0x004	Control register
I2Sx_SR	0x008	Interrupt and Status Registers
I2Sx_CFGR	0x01C	configuration register
I2Sx_PR	0x020	Master Mode Divider Register
I2Sx_ICR	0x024	Interrupt Flag Clear Register
I2Sx_DRL	0x028	Left channel send data and receive data register
I2Sx_DRR	0x02C	Right channel send data and receive data register

28.7.1 I2Sx Control Register (I2Sx_CR)

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
Res								TXEIE	RXNEIE	ERRIE	Res.			RDMAEN	LDMAEN
								RW	RW	RW				RW	RW

Bit	Marking	Functional description
31:8	Reserved	Keep
7	TXEIE	Transmit buffer empty interrupt enable (Tx buffer empty interrupt enable) 0: mask TXE interrupt 1: Enable TXE interrupt. An interrupt request is generated when the TXE flag is set.
6	RXNEIE	Receive buffer not empty interrupt enable (RX buffer not empty interrupt enable) 0: Mask RXNE interrupt 1: Enable RXNE Interrupt An interrupt request is generated when the RXNE flag is set to 1.
5	ERRIE	Error interrupt enable This bit is used to control interrupt generation on error conditions (UDR, OVR and FRE). 0: mask error interrupt 1: enable error interrupt
4:2	Reserved	Keep
1	RDMAEN	Right channel buffer DMA data transmission enable (Right buffer DMA enable) When this bit is set to 1, a DMA request will be generated whenever the right channel has data sending and receiving flag set to 1. 0: disable right channel buffer DMA 1: Enable right channel buffer DMA
0	LDMAEN	Left channel buffer DMA channel data transmission enable (Left buffer DMA enable) When this bit is set to 1, a DMA request will be generated whenever the left channel has data to send and the flag is set to 1. 0: disable left channel buffer DMA 1: Enable left channel buffer DMA

28.7.2 I2Sx Status Register (I2Sx_SR)

Offset address 0x08

Reset value 0x0000 8002

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE_R	RXNE_R	OVR_R	Res				FRE	BSY	OVR_L	Res.		UDR_R	UDR_L	TXE_L	RXNE_L
RO	RO	RO					RO	RO	RO			RO	RO	RO	RO

Bit	Marking	Functional description
31:16	Reserved	Keep
15	TXE_R	The right channel send buffer is empty (Transmit buffer empty) 0: send buffer is not empty 1: send buffer is empty
14	RXNE_R	The right channel receive buffer is not empty (Receive buffer not empty) 0: send buffer is empty 1: receive buffer is not empty
13	OVR_R	Right channel overflow flag (Overrun flag) 0: No overflow occurred 1: overflow occurred This flag is set by hardware and cleared by software
12:9	Reserved	Keep
8	FRE	Frame Error 0: no frame error 1: Framing error occurred This bit is set by hardware and cleared by software. Regardless of the audio protocol chosen, it detects changes on the WS line at unexpected times in slave mode, notifying the external master and slave of a failure in synchronization.
7	BSY	BSY：Busy flag 0： I2S is not busy 1： I2S is busy with communication or the send buffer is not empty This flag is set and cleared by hardware.
6	OVR_L	Left channel overflow flag (Overrun flag) 0: No overflow occurred 1: overflow occurred This flag is set by hardware and cleared by software
5:4	Reserved	Keep
3	UDR_R	Right channel underflow flag (Underrun flag) 0: No underflow has occurred 1: Underflow occurred This flag is set by hardware and cleared by software
2	UDR_L	Left channel underflow flag (Underrun flag) 0: No underflow has occurred 1: Underflow occurred This flag is set by hardware and cleared by software
1	TXE_L	Left channel send buffer is empty (Transmit buffer empty) 0: send buffer is not empty 1: send buffer is empty
0	RXNE_L	The left channel receive buffer is not empty (Receive buffer not empty) 0: send buffer is empty 1: receive buffer is not empty

28.7.3 I2Sx Configuration Register (I2Sx_CFGR)

Offset address 0x1C

Reset value 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Resereved					I2SE	I2SCFG		PCMSYNC	CKSEL	I2SSTD		Res	DATLEN		CHLEN
					RW	RW		RW	RW	RW			RW		RW

Bit	Marking	Functional description
31:11	Reserved	Keep
10	I2SE	I2S Enable 0: disable I2S 1: enable I2S
9:8	I2SCFG	I2S configuration mode 00: Slave mode - send 01: Slave mode - receive 10: master mode - send 11: Master mode - receive
7	PCMSYNC	PCM frame synchronization 0: short frame synchronization 1: long frame synchronization Note: This bit is meaningful only when I2SSTD = 11 (using PCM standard)
6	CKSEL	I2S clock selection in master mode 0 PCLK 1 HCLK
5:4	I2SSTD	I2S standard selection 00: I2S Philips standard. 01: MSB alignment standard (left alignment) 10: LSB justified standard (right justified) 11: PCM standard Note: To ensure correct operation, these bits should be configured when I2S is disabled
3	Reserved	Keep the value at 0
2:1	DATLEN	Data length to be transferred 00: 16 -bit data length 01: 24 -bit data length 10: 32 -bit data length 11: Not allowed Note: To ensure correct operation, these bits should be configured with I2S disabled.
0	CHLEN	CHLEN：Channel length (number of bits per audio channel) 0: 16 bits 1: 32 bits Only when DATLEN = 00, the operation of writing 0 of this bit is meaningful, and this bit is forced to write 1 under other data lengths. Note: To ensure correct operation, this bit should be configured when I2S is disabled.

28.7.4 I2Sx Prescaler Register (I2Sx_PR)

Offset address 0x20

Reset value 0x0000 0002

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRACT						MCKOE	ODD	I2SDIV							
RW						RW	RW	RW							

Bit	Marking	Functional description
31:16	Reserved	Keep
15:10	FRACT	Prescaler fractional frequency setting, ODD setting is valid when 0x00, fractional frequency is 0, when non-zero, ODD setting is invalid Actual frequency division value = (I2SDIV +FRACT/64)*2 Note: This bit should be configured when I2S is disabled.
9	MCKOE	Master clock output enable 0: disable main clock output 1: Enable main clock output Note: This bit should be configured when I2S is disabled.
8	ODD	Odd factor for the prescaler 0: The actual frequency division value = I2SDIV *2 1: The actual frequency division value = (I2SDIV * 2)+1 See Prescaler Settings for details Note: This bit should be configured when I2S is disabled.
7:0	I2SDIV	I2SDIV： I2S Linear prescaler I2SDIV[7:0] = 0 is a disabled value. See Prescaler Settings for details Note: These bits should be configured when I2S is disabled.

28.7.5 I2Sx Interrupt Status Clear Register (I2Sx_ICR)

Offset address 0x24

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
Reserved							FRE	Res	OVR	Res		UDF	Res		
							W0		W0			W0			

Bit	Marking	Functional description
31:9	Reserved	Keep
8	FRE	Frame Error flag cleared write 0 to clear Writing 1 is invalid
7	Reserved	Keep
6	OVR	Overrun flag write 0 to clear Writing 1 is invalid
5:4	Reserved	Keep
3	UDF	Underrun flag write 0 to clear Writing 1 is invalid
2:0	Reserved	Keep

28.7.6 I2Sx Data Register L (I2Sx_DRL)

Offset address 0x28

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
DRL															
RW															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	DRL	Data register Left channel data and PCM data received or to be sent. The data register is divided into 2 buffers, one for writing (transmit buffer) and one for reading (receive buffer). When writing to the data register, the data will be written to the transmit buffer, and when reading from the data register, the value in the receive buffer will be returned.

28.7.7 I2Sx Data Register R (I2Sx_DRR)

Offset address 0x2C

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
DRR															
RW															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:0	DRR	Data register Right channel data received or to be sent. Invalid in PCM mode. The data register is divided into 2 buffers, one for writing (transmit buffer) and one for reading (receive buffer). When writing to the data register, the data will be written to the transmit buffer, and when reading from the data register, the value in the receive buffer will be returned.

29 USB 2.0 Full Speed Module (USBFS)

29.1 Introduction to USBFS

The USB Full Speed (USBFS) controller provides a set of USB communication solutions for portable devices. The USBFS controller supports device mode, and the chip integrates a full-speed PHY. Support full speed (FS, 12Mb/s) transceiver in device mode. The USBFS controller supports all four transfer modes defined by the USB 2.0 protocol (control transfer, bulk transfer, interrupt transfer, and isochronous transfer).

29.2 Main Features of USBFS

There are two main categories: general features and device mode features.

29.2.1 General features

- Built-in on-chip USB 2.0 full-speed PHY
- Support device mode
- The module is embedded with DMA, and the AHB burst transfer type can be configured by software
- Power-saving features such as USB suspend, stop RAM clock, stop PHY domain clock
- 1.25KB of dedicated RAM with advanced FIFO control
- The RAM space can be divided into different FIFOs for flexible and effective use of RAM
- Each FIFO can store multiple packets
- Dynamically allocate memory
- The size of the FIFO can be configured as a non-power-of-two value, so that the storage unit can be used continuously
- No application intervention is required within one frame to achieve maximum USB bandwidth

29.2.2 Device Mode Properties

- The device mode supports USB 2.0 full speed (FS, 12Mb/s) transmission.
- 1 bidirectional control endpoint 0
- 4 IN endpoints (1, 3, 5, 7), can be configured as bulk transfer, interrupt transfer or isochronous transfer
- 4 OUT endpoints (2, 4, 6, 8), can be configured as bulk transfer, interrupt transfer or isochronous transfer
- 8 endpoints can be configured as synchronous transmission (periodic transmission) mode
- Contains an aperiodic transmit FIFO (shared by all aperiodic IN endpoints), a periodic transmit FIFO (exclusively shared by periodic IN endpoints) and a receive FIFO (shared by all OUT endpoints)
- Support remote wake-up function.

- Support soft disconnect function

29.3USBFS System Block Diagram

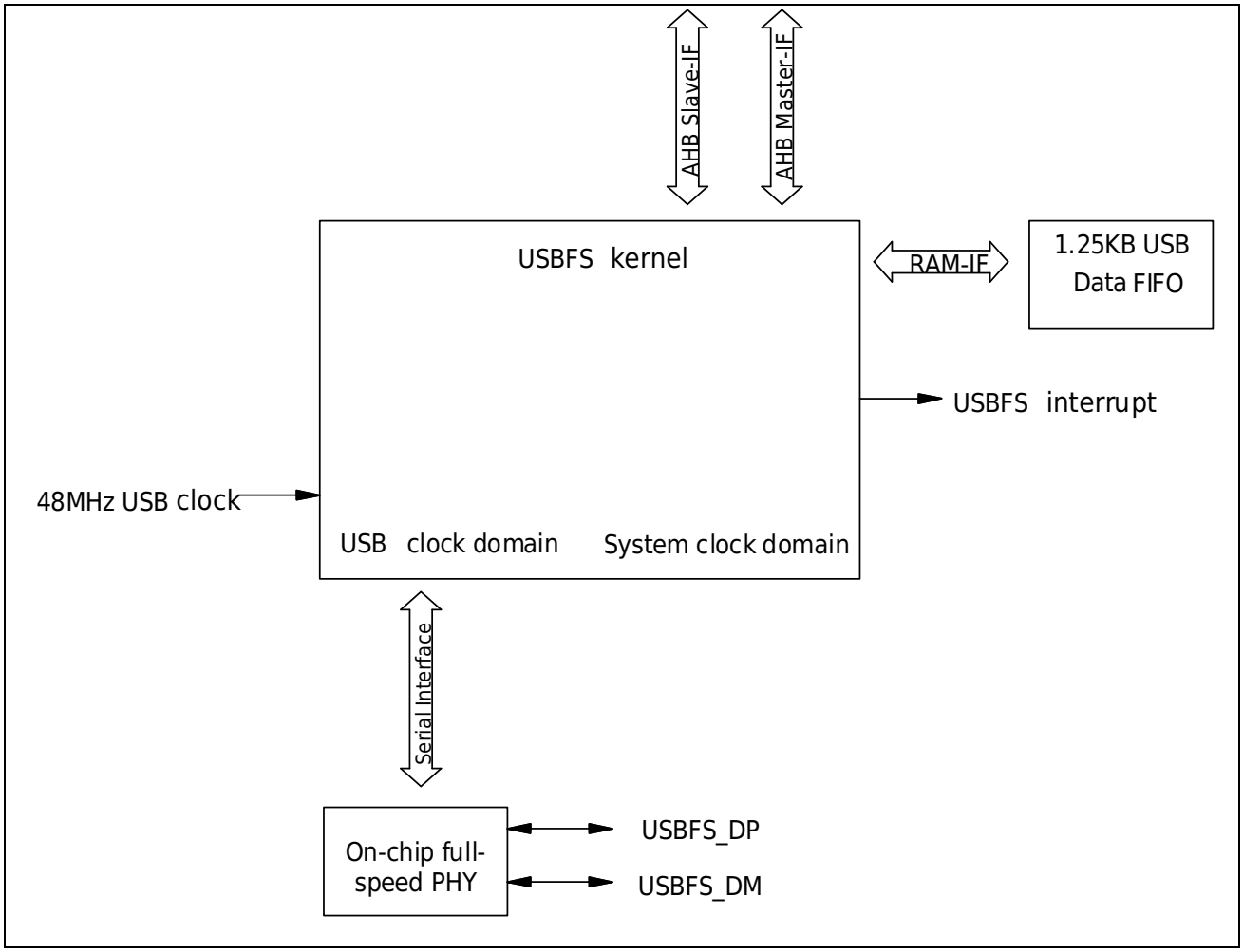


Figure 29-1 USBFS System Block Diagram

29.4USBFS pin description

Table 29-1 USBFS Pin Descriptions

Pin name	Direction	Functional description
USBFS_DP	Input/Output	Differential data D+ signal
USBFS_DM	Input/Output	Differential Data D-Signal

Since the USBFS_DP and USBFS_DM pins are multiplexed with general GPIO, when using USBFS, it is recommended to turn off the digital function of the corresponding pins. For details, please refer to the port controller (GPIO) chapter. In addition, when the USBFS function is not in use, when the digital function pins corresponding to the USBFS_DP and USBFS_DM pins are flipped, additional current consumption will be generated.

29.5USBFS function description

29.5.1 USBFS clock and working mode

The clock used by USBFS includes PHY clock and system clock. The PHY clock frequency needs to be configured as 48MHz. The 48MHz clock can be generated by the internal PLL circuit or by the internal high-speed 48M RC clock. Before using the USBFS module, it needs to be configured in the system controller module. Configure the USBFS clock inside.

USBFS is used as a device and includes an on-chip full-speed PHY.

Pull-up and pull-down resistors have been integrated inside the on-chip full-speed PHY, and USBFS can be automatically selected according to the current mode and connection status.

When USBFS is working, the VCC voltage range is 3.0~3.6V.

29.5.2 USBFS device capabilities

29.5.2.1 Introduction to device functions

A typical USB device mode system construction diagram is as follows:

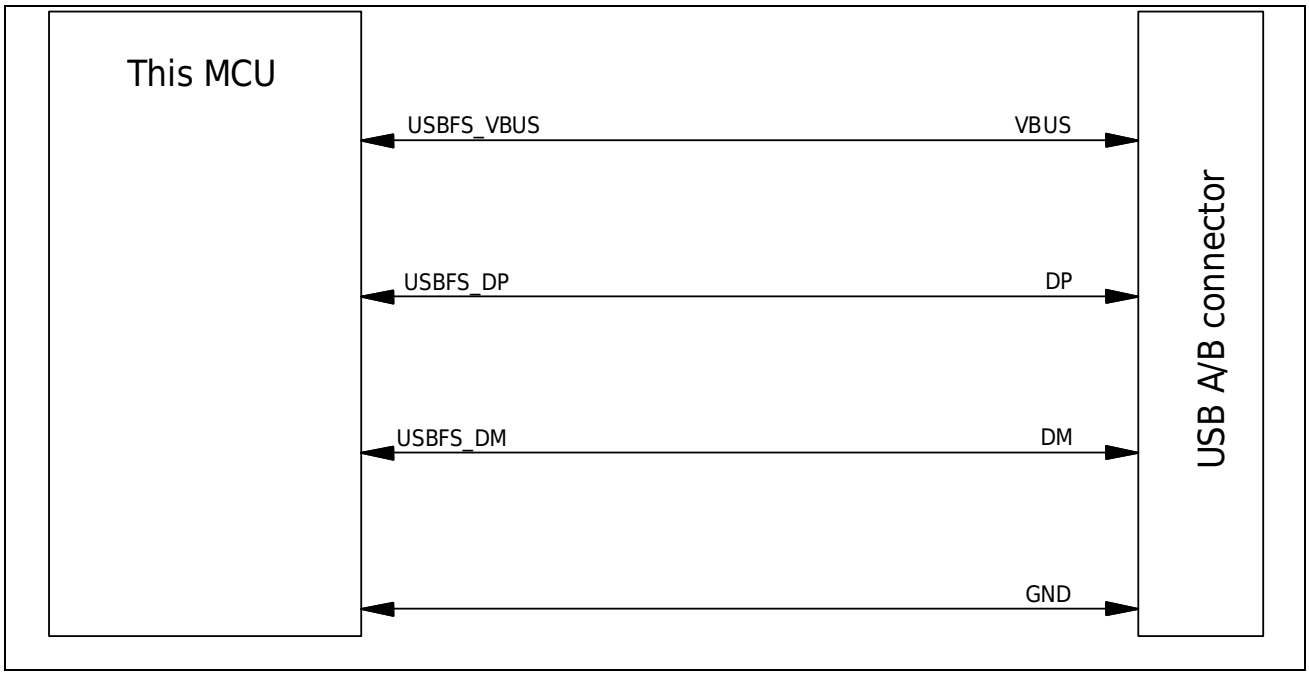


Figure 29-2 USBFS device mode system construction diagram

29.5.2.2 Device power status

When the module detects that USBFS_VBUS is high level, it will make the USB device enter the power supply state (see USB2.0 Section 9.1). Then, USBFS automatically connects the DP pull-up resistor, sends a signal that the full-speed device is connected to the host and generates a session request interrupt (VBUSVINT bit in USBFS_GINTSTS), indicating to enter the power supply state.

In the powered state, USBFS expects to receive a reset signal from the host. Other USB operations cannot be performed. As soon as the reset signal is received, a reset detected interrupt (USBRST in USBFS_GINTST) is generated. After the reset signal ends, an enumeration complete interrupt (ENUMDNE bit in USBFS_GINTSTS) will be generated, and USBFS will then enter the default state.

29.5.2.3 Device default state

By default, USBFS expects to receive a SET_ADDRESS command from the host. Other USB operations cannot be performed. When a valid SET_ADDRESS command is decoded on the USB, the application will write the corresponding address value to the device address field in the device configuration register (DAD bit in USBFS_DCFG). The USBF then enters the address state and is ready to answer host transactions with the configured USB address.

29.5.2.4 Device suspend state

The USBFS device continuously monitors USB activity. After the USB idle time reaches 3ms, an early suspend interrupt (ESUSP bit in USBFS_GINTSTS) will be issued, and the device will enter the suspend state confirmed by the suspend interrupt (USBSUSP bit in USBFS_GINTSTS) after 3ms. Then, the device suspend bit in the device status register (SUSPSTS bit in USBFS_DSTS) is automatically set to 1, and USBFS then enters the suspend state.

Suspend can be exited through the device itself. In this case, the application will set the remote wakeup signal bit in the device control register (RWUSIG bit in USBFS_DCTL) to 1, and clear it to 0 within 1ms to 15ms.

However, if the device detects the resume signal sent by the host, it will generate a resume interrupt (WKUPINT bit in USBFS_GINTSTS), and the device suspend bit will be automatically cleared.

29.5.2.5 Device soft disconnect

The powered state can be exited by software with the soft-disconnect function. The DP pull-up resistor can be removed by setting the soft disconnect bit (SDIS bit in USBFS_DCTL) in the device control register to 1. At this time, although the USB cable is not actually pulled out from the host port, the device disconnection will still occur on the host side Detect interruption.

29.5.2.6 Device endpoint

Endpoint class

The USBFS module implements the following USB endpoints:

- Control endpoint 0:
  - Bi-directional and only handles control messages
  - Use a separate set of registers for IN and OUT transactions
  - Dedicated Control (USBFS_DIEPCTL0/USBFS_DOEPCTL0) Register, Transport Configuration (USBFS_DIEPTSIZ0/USBFS_DOEPTSIZ0) Register, and Status

Interrupt (USBFS_DIEPINT0/USBFS_DOEPINT0) Register. The set of bits available in the control and transfer configuration registers is slightly different from the other endpoints

- 4 IN endpoints (n = 1,3,5,7)
  - Each endpoint can be configured to support isochronous, bulk, or interrupt transfer types
  - Each endpoint has dedicated control (USBFS_DIEPCTLn) registers, transfer configuration (USBFS_DIEPTSIZn) registers, and status interrupt (USBFS_DIEPINTn) registers
  - The device IN endpoint generic interrupt mask register (USBFS_DIEPMSK) can be used to enable/disable the same class of endpoint interrupt sources on all IN endpoints (including EP0)
  - Supports incomplete isochronous IN transfer interrupt (IISOIXFR bit in USBFS_GINTSTS), which will trigger when at least one transfer on an isochronous IN endpoint is incomplete in the current frame. This interrupt is triggered together with the periodic frame interrupt (USBFS_GINTSTS/EOPF)
- 4 OUT endpoints (n = 2,4,6,8)
  - Each endpoint can be configured to support isochronous, bulk, or interrupt transfer types
  - Each endpoint has dedicated control (USBFS_DOEPCTLn) registers, transfer configuration (USBFS_DOEPTSIZn) registers, and status interrupt (USBFS_DOEPINTn) registers
  - The device OUT endpoint general interrupt mask register (USBFS_DOEPMSK) can be used to enable/disable the same type of endpoint interrupt source on all OUT endpoints (including EP0)
  - Support for incomplete isochronous OUT transfer interrupts (INCOMPISOOUT bit in USBFS_GINTSTS), which will trigger when at least one transfer on an isochronous OUT endpoint is incomplete in the current frame. This interrupt is triggered together with the periodic frame interrupt (USBFS_GINTSTS/EOPF)

endpoint control

The application can take the following control over the endpoint through the device endpoint n IN/OUT control register (DIEPCTLn/DOEPCTLn):

- Endpoint enable/disable
- Activate the endpoint under the current configuration
- Set USB transfer type (isochronous, bulk and interrupt)
- Set the supported packet size
- Sets the Tx-FIFO number associated with the IN endpoint
- Set the data0/data1 PID you want to receive or use when sending (bulk/interrupt transfers only)
- Set the odd/even frame for receiving or sending transactions (only for synchronous transfers)
- The NAK bit can be set, so that regardless of the state of the FIFO at this time, a NAK is replied to the host's request

- The STALL bit can be set so that the host's tokens for this endpoint are all replied to STALL by the hardware
- The OUT endpoint can be set to listen mode, that is, no CRC check is performed on the received data

Endpoint transport

The Device Endpoint n Transfer Size registers (DIEPTSIZn/DOEPTSIZn) allow applications to program transfer size parameters and read transfer status. Setting this register must be done before setting the Endpoint Enable bit in the Endpoint Control register. After the endpoint is enabled, these fields become read-only immediately, and the USBFS module updates these fields according to the current transmission status.

The following transfer parameters can be programmed:

- Transfer size in bytes
- The number of packets that make up the entire transfer

Endpoint Status/Status

The Device Endpoint n Interrupt Registers (DIEPINTn/DOPEPINTn) indicate the status of the endpoint on USB and AHB related events. When the OUT endpoint interrupt bit or the IN endpoint interrupt bit (OEPINT bit in USBFS_GINTSTS or IEPINT bit in USBFS_GINTSTS, respectively) is set in the module interrupt register, the application must read these registers for detailed information. Before an application can read these registers, it must first read the Device Overall Endpoint Interrupt (USBFS_DAINT) register to obtain the endpoint number of the device endpoint n interrupt register. The application must clear the corresponding bits in this register to clear the corresponding bits in the DAINT and GINTSTS registers.

- The module provides the following status checking and interrupt generation functions:
- Transfer complete interrupt, indicating to the application that both the AHB and the USB side have completed the data transfer
- Setup phase completed (only for OUT endpoints of control transfer type)
- The associated transmit FIFO is half empty or fully empty (IN endpoint)
- A NAK reply has been sent to the host (IN endpoint for isochronous transfers only)
- IN token received while TxFIFO is empty (only for IN endpoints of bulk and interrupt transfer types)
- OUT token received when the endpoint has not been enabled
- babble error detected
- Application shutdown endpoint takes effect
- The application sets NAK on the endpoint to take effect (only for the IN endpoint of the synchronous transmission type)
- Received more than 3 consecutive setup packets (only for control type OUT endpoints)

- A timeout condition was detected (only for IN endpoints of control transfer type)
- Packets of isochronous transfer type are lost without generating an interrupt

29.5.3 USBFS power control

When the USBFS module is not used, the HCLK and PHY clocks of the USBFS module can be stopped by the system controller module to reduce power consumption.

Power reduction techniques can be used in the USB suspend state when the USB module is used but the device USB session is not started or the device is not connected.

- Stop the PHY clock (STPPCLK bit in USBFS_GCCTL)
  Most of the 48 MHz internal clock domains of the USBFS full-speed module are clock-gated off when the Stop PHY Clock bit in the Clock-Gating Control Register is set to 1. Even if the application still provides the clock input, it will save the dynamic power consumption of the module due to the flipping of the clock signal. It will also turn off most of the units of the transceiver, and only the part responsible for detecting an asynchronous recovery event or a remote wake-up event will remain active. .
- HCLK gating (GATEHCLK bit in USBFS_GCCTL)
  When the GATEHCLK bit in the Clock Gating Control Register is set to 1, most of the system clock domains inside the USBFS module are clock-gated off. Only the register read and write interfaces remain active. Even if the application still provides the clock input, it will save the module's dynamic power consumption due to the clock signal flipping.

To save dynamic power, the USB data FIFO is only clocked when it is accessed by the USBFS module.

29.5.4 USBFS data FIFO

The USBFS system has 1.25KB dedicated RAM and adopts an efficient FIFO control mechanism. The packet FIFO controller module in the USBFS module divides the RAM space into an aperiodic TxFIFO, a periodic TxFIFO (the application program pushes data into it for short-term storage before USB transmission) and an RxFIFO (data received from the USB is stored there briefly before being read by the application).

The aperiodic TxFIFO is shared by all aperiodic IN endpoints, and the RxFIFO is shared by all OUT endpoints. The sizes of the FIFOs are all software configurable to better meet application requirements.

29.5.5 USBFS device FIFO architecture

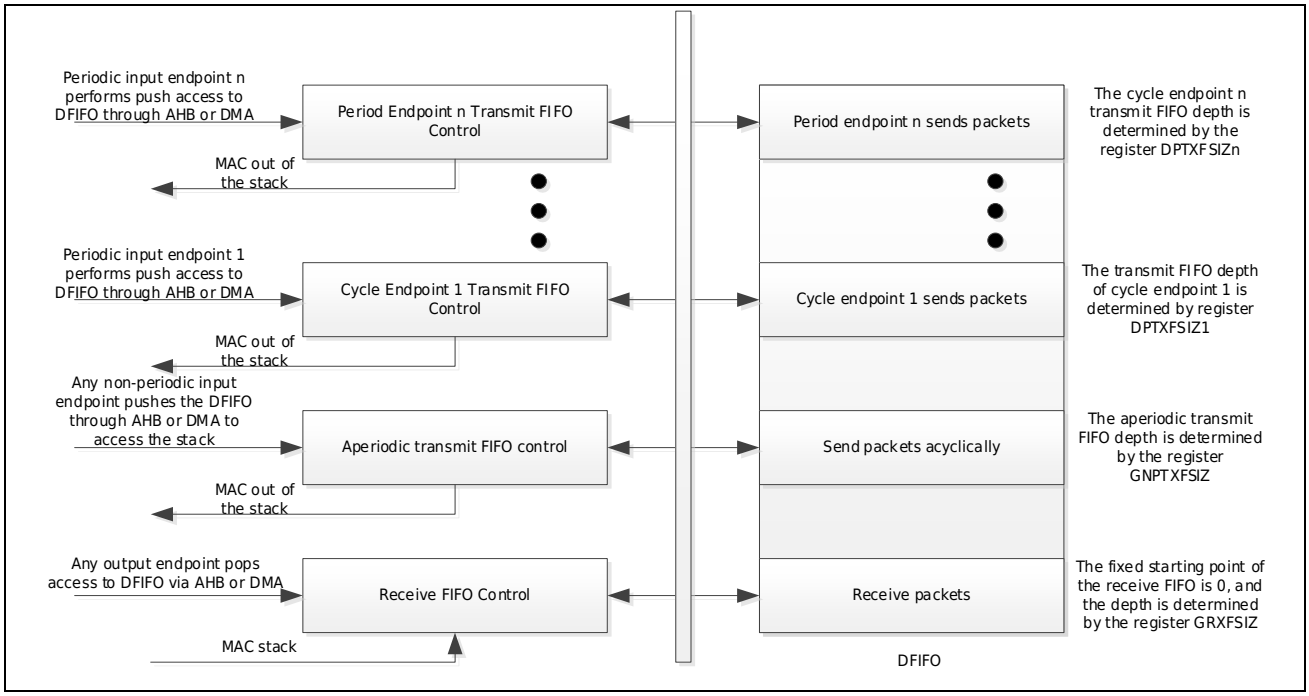


Figure 29-3 Schematic diagram of FIFO architecture in USBFS device mode

29.5.5.1 Device RxFIFO

USBFS devices use a single receive FIFO to receive data sent to all OUT endpoints. As long as there is free space in the RxFIFO, the received data packets are filled into the RxFIFO one by one. In addition to valid data, received packet status (containing OUT endpoint destination number, byte count, data PID, and validation of received data) is also stored by the module. When no space is available, the device replies with a transaction NAK to the host and triggers an interrupt on the addressed endpoint. The size of the receive FIFO is configured in the receive FIFO size register (GRXFSIZ).

A single receive FIFO architecture enables the USB device to more efficiently fill the receive RAM buffer:

- All OUT endpoints share the same RAM buffer (shared FIFO)
- The USBFS module can fill the receive FIFO to the limit for any host serial OUT token

The application will always receive the RxFIFO not empty interrupt (RXFNE bit in USBFS_GINTSTS) as long as at least one packet is available for reading in the RxFIFO. The application program reads the packet information from the receive status read and stack register (GRXSTSP), and finally reads the corresponding data from the receive FIFO by reading the stack address related to the endpoint.

29.5.5.2 DeviceTxFIFO

All non-periodic IN endpoints share one TxFIFO. A TxFIFO is exclusive to the cycle IN endpoint. The application configures the FIFO size for the aperiodic IN endpoint through the aperiodic transmit

FIFO size register (USBFS_GNPTXFSIZ); configures the FIFO size for the periodic IN endpoint n through the periodic IN endpoint transmit FIFOn size register (DPTXFSIZn).

29.5.6 USBFS FIFO RAM allocation

Receive FIFO RAM allocation

The application should allocate RAM for SETUP packets: 11 slots must be reserved in the receive FIFO to receive SETUP packets on the control endpoint. The USBFS module will not write any other data to these locations reserved for SETUP packets. A location will be allocated for the global OUT NAK. Status information is written to the FIFO with each received packet. Therefore, at least (maximum packet size/4)+1 space must be allocated for received packets. If multiple isochronous endpoints are enabled, the space allocated for receiving consecutive packets must be at least twice (maximum packet size/4)+1. Usually, the recommended space is twice the size of (maximum packet/4 + 1), so that when the previous packet is sent to the CPU, the USB can receive subsequent packets at the same time.

The transfer completion status information is pushed into the FIFO along with the last packet received by this endpoint. In general, it is recommended to assign a location for each OUT endpoint.

Transmit FIFO RAM allocation

The minimum RAM space required for each IN endpoint's transmit FIFO is the maximum packet size for that particular IN endpoint.

29.5.7 USBFS system performance

With large RAM buffer, highly configurable FIFO size, fast 32-bit FIFO access via AHB push/pop registers, and especially advanced FIFO control mechanism for optimal USB and system performance. In fact, regardless of the current USB sequence, USBFS can efficiently fill the available RAM space through this mechanism. With these features:

- The application has enough margin to calculate and correct the CPU load to optimize CPU bandwidth utilization:
  - The application program can accumulate a large amount of sending data first, and then send it out via USB
  - Allows sufficient time margin to read data from the receive FIFO
- The USB module can keep working at full speed, that is, to provide the maximum full-speed bandwidth (as much hardware as possible to run automatically, and as little software as possible to participate)
  - The USB module can accumulate a large amount of sending data in advance at its disposal, so that the USB data sending can be autonomously managed
  - There is a lot of empty space in the receive buffer, which can be automatically filled with data from the USB

Because the USBFS module can efficiently fill the 1.25KB RAM buffer and 1.25KB send/receive data is enough to meet the amount of data that can be accommodated in a full-speed frame, the USB system can reach the maximum USB bandwidth within one frame without application program intervention.

29.5.8 USBFS interrupt and Events

The global interrupt is the main interrupt that the software needs to handle, and the flag bit of the global interrupt can be read in the USBFS_GINTSTS register.

Table 29-2 USBFS interrupt/Event Table

Interrupt logo	Description
WKUPINT	Remote Wakeup Interrupt
DATAFSUSP	Data fetch pending
INCOMPISOOUT	Incomplete OUT synchronous transfer
IISOIXFR	IN SYNC TRANSFER NOT COMPLETED
OEPINT	OUT endpoint interrupt
IEPINT	IN endpoint interrupt
EOPF	Periodic end-of-frame interrupt
ISOODRP	Drop sync OUT packet interrupt
ENUMDNE	Enumeration complete
USBRST	USB reset interrupt
USBSUSP	USB suspend interrupt
ESUSP	Early hang interrupt
GONAKEFF	Global OUT NAK active interrupt
GINAKEFF	Global aperiodic IN NAK active interrupt
RXFNE	RxFIFO not empty interrupt
SOF	Start of frame interrupt
MMIS	Pattern mismatch interrupt

29.6USBFS programming model

29.6.1 USBFS module initialization

The work of the USBFS module requires two clocks, the system clock and the PHY clock, and the PHY clock frequency must be configured as 48Mhz. PHY clock can choose PLL clock or internal high-speed RC48M clock.

1. Configure the PLL clock or RC48M clock, and wait for the clock output to be stable (for the operation process, please refer to the chapter of the system controller (SYSCTRL).
2. Configure the system controller register SYSCTRL1.USB48MSEL, set 0 to select the RC48M clock, and set 1 to select the PLL clock.

3. Configure the system controller register PERI_CLKEN1.USB to set 1 to enable the USBFS module clock.

29.6.2 USBFS device initialization

An application must perform the following steps to initialize the module as a device.

1. In Slave mode, USBFS_GINTMSK.NPTxFEmpMsk and USBFS_GINTMSK.RxFLvlMsk are set to 1, in DMA mode, USBFS_GINTMSK.NPTxFEmpMsk and USBFS_GINTMSK.RxFLvlMsk are set to 0
2. Program the following fields in the USBFS_DCFG register:
   - Device Speed (DevSpd)
   - Non-zero length status output handshake signal (NZStsOUTHShk)
   - Period frame interval (PerFrInt)
3. Clear the USBFS_DCTL.SftDiscon bit to let the controller perform the connection to the master.
4. Program the USBFS_GINTMSK register to unmask the following interrupts:
   - USB reset (USBRstMsk)
   - Enumeration Done (EnumDoneMsk)
   - Early Suspend (ErlySuspMsk)
   - USB Suspend (USBSuspMsk)
   - SOF(SofMsk)
5. Wait for USBFS_GINTSTS.USBRst interrupt. This indicates that a reset signal has been detected on the USB, and the reset process lasts about 10ms from the reception of this interrupt. The application must perform the steps listed in the section Endpoint initialization on USB reset.
6. Wait for the USBFS_GINTSTS.EnumDone interrupt. This interrupt signals the end of the reset process on the USB. When this interrupt is received, the application must read the USBFS_DSTS register to determine the enumeration speed and perform the steps listed in Endpoint Initialization on USB Enumeration Completion.
   At this point, the device is ready to accept SOF packets and perform control transfers on control endpoint 0.

29.6.3 USBFS device soft disconnect operation

The application program can set the register USBFS_DCTL.SftDiscon to perform a soft disconnect operation. When the device is in data transfer state, perform the following steps:

1. Set register USBFS_DCTL.SftDiscon
2. Set register USBFS_GRSTCTL.CSftRst
3. Wait for register USBFS_GRSTCTL.CSftRst bit to be cleared by controller
4. Re-execute the USBFS device initialization steps

29.6.4 Endpoint initialization on USB reset

1. Set all output endpoints NAK bits to 1
   USBFS_DOEPCTLn.SNAK = 1 (for all output endpoints)
2. Unmask the following interrupt bits
   USBFS_DAINTMSK.INEP0=1 (control 0 input endpoint)
   USBFS_DAINTMSK.OUTEP0=1 (Control 0 output endpoint)
   USBFS_DOEPMSKn.SetUPMsk=1
   USBFS_DOEPMSKn.XferComplMsk=1
   USBFS_DIEPMSKn.XferComplMsk=1
   USBFS_DIEPMSKn.TimeOUTMsk=1
3. Set data FIFO RAM for each FIFO
   Program the USBFS_GRXFSIZ register to be able to receive output data and SETUP data for control transfers. This register value must be at least equal to 1 maximum packet size for Control Endpoint 0 + 2 words (for the state of the control output packet) + 10 words (for the SETUP packet).
   Program the USBFS_GNPTXFSIZ register to be able to send control input data. This register value must be at least equal to 1 maximum packet size for Control Endpoint 0.
4. Program the following fields in the endpoint-associated registers to control output endpoint 0 to receive SETUP packets.
   – USBFS_DOEPTSIZ0.SUPCnt=3 (receive up to 3 consecutive SETUP packets)
   – DMA mode, the USBFS_DOEPDMA0 register specifies the memory address to store the received SETUP data packet

   At this point, all initialization required to receive the SETUP packet is complete.

29.6.5 USB enumeration completes

1. After the enumeration complete interrupt (USBFS_GINTSTS.EnumDone) occurs, the register USBFS_DSTS.EnumSpd is read to determine the enumeration speed of the device.
2. Program the MPS field in USBFS_DIEPCTL0 to set the maximum packet size. This step configures control endpoint 0. The maximum packet size for the control endpoint depends on the enumeration speed.
3. DMA mode, programming the USBFS_DOEPCTL0 register enables control of output endpoint 0.
   USBFS_DOEPCTL0.EPEna = 1
   At this point, the device is ready to receive SOF packets and is configured to perform control transfers on control endpoint 0.

29.6.6 Endpoint initialization when a SetAddress command is received

This section describes what an application must do when it receives a SetAddress command in a SETUP packet.

1. Program the USBFS_DCFG register with the device address received in the SetAddress command.
2. Program the block to emit input packets for the status phase.

29.6.7 Endpoint initialization when a SetConfiguration/SetInterface command is received

This section describes what an application must do when it receives a SetConfiguration or SetInterface command in a SETUP packet.

1. When the SetConfiguration command is received, the application must program the endpoint registers to configure them with the characteristics of valid endpoints in the new configuration.
2. When a SetInterface command is received, the application must program the endpoint registers specified by the command.
3. An endpoint that was valid in a previous configuration or other setup is not valid in a new configuration or other setup. These invalid endpoints must not be activated.
4. Use the USBFS_DAINTMSK register to enable interrupts for valid endpoints and mask interrupts for invalid endpoints.
5. Set data FIFO RAM for each FIFO.
6. After configuring all required endpoints, the application must program the module to send input packets for the status phase.
   At this point, the device module is ready to receive and send any type of data packet

29.6.8 Endpoint activation (Active)

This section describes the steps required to activate a device endpoint or configure an existing device endpoint to a new type.

1. Program the characteristics of the desired endpoint in the following fields of the USBFS_DIEPCTLn register (for input endpoints) or the USBFS_DOEPCTLn register (for output endpoints).
   - Maximum Packet Size (MPS)
   - USB active endpoint bit (USBActEP) set
   - Endpoint Initial Data Synchronization Bit (DPID) (for interrupt and bulk endpoints)
   - Endpoint type (EPType)
   - TxFIFO number (TxFNum)
2. Once the endpoint is activated, the module starts decoding tokens sent to the endpoint and

replies with a valid handshake if the received token is valid.

29.6.9 Endpoint inactive (Deactive)

This section describes the steps required to deactivate existing endpoints.

1. In the endpoint to be deactivated, clear the USB active endpoint bit (USBActEP) in the USBFS_DIEPCTLn register (for input endpoints) or USBFS_DOEPCTLn register (for output endpoints) to 0.
2. When an endpoint is deactivated, the module ignores tokens sent to it, causing a USB timeout.

29.6.10 Disable (Disable) output endpoint

The application must follow the steps below to disable an enabled output endpoint.

1. USBFS_DCTL.SGOUTNak=1, enter the global output endpoint NAK mode
2. Waiting for interrupt USBFS_GINTSTS.GOUTNakEff
3. USBFS_DOEPCTLn.EPDisable=1，USBFS_DOEPCTLn.SNAK=1
4. Wait for interrupt USBFS_DOEPINTx.EPDisbld,
   When an interrupt occurs, the controller clears the following registers
   USBFS_DOEPCTLx.EPDis = 0
   USBFS_DOEPCTLn.EPEna = 0
5. The program must exit the global output endpoint NAK mode
   USBFS_DCTL.CGOUTNak = 1
   USBFS_DCTL.SGOUTNak=0

29.6.11 Pause (Stall) asynchronous output endpoint

Applications must follow the steps below to suspend an asynchronous output endpoint.

1. USBFS_DCTL.SGOUTNak=1, enter the global output NAK mode
2. Waiting for interrupt USBFS_GINTSTS.GOUTNakEff
3. USBFS_DOEPCTLn.EPDisable=1，USBFS_DOEPCTLn.Stall=1
4. Wait for interrupt USBFS_DOEPINTx.EPDisbld,
   When an interrupt occurs, the controller clears the following registers
   USBFS_DOEPCTLx.EPDis = 0
   USBFS_DOEPCTLn.EPEna = 0
5. The program must exit the global output endpoint NAK mode
   USBFS_DCTL.CGOUTNak = 1
   USBFS_DCTL.SGOUTNak=0

29.6.12 Global output endpoint NAK mode

The program must enter the global output endpoint NAK mode to stop receiving any data in the receive FIFO.

1. USBFS_DCTL.SGOUTNak=1, enter the global output NAK mode.
2. Waiting for the interrupt USBFS_GINTSTS.GOUTNakEff, when an interrupt occurs, it means that the controller has stopped receiving data other than SETUP packets.
3. The program can write the register to mask this interrupt USBFS_GINTMSK.GOUTNakEffMsk=0.
4. If the program wants to exit the global output NAK mode, perform the following operations to clear the interrupt at the same time.
   USBFS_GINTSTS.GOUTNakEff
   USBFS_DCTL.CGOUTNak = 1
   USBFS_DCTL.SGOUTNak=0

29.6.13 Output endpoint transmission stopped

1. Enable all output endpoints USBFS_DOEPCTLn.EPEna = 1.
2. Wait for USBFS_GRSTCTL.AHBIdle to be set, indicating that the AHB bus controller of the controller is idle.
3. USBFS_GRSTCTL.RxFFlsh =1
4. Wait for USBFS_GRSTCTL.RxFFlsh to be cleared, indicating that the receive FIFO is successfully emptied.
5. Execution section Disable (Disable) output endpoint.

29.6.14 Configure the interrupt input endpoint as a periodic or aperiodic endpoint (shared FIFO mode)

In the shared FIFO mode, the interrupt input endpoint can be configured as a periodic input endpoint or an aperiodic input endpoint by setting the value of USBFS_DIEPCTLn.TxFNum.

When configured as an acyclic input endpoint, the program must follow the operation flow of the acyclic input endpoint.

When configured as a periodic input endpoint, the program must follow the operation flow of the periodic input endpoint.

29.6.15 Global aperiodic input endpoint NAK mode (shared FIFO mode)

1. Set USBFS_DCTL.SGNPInNak=1 to notify the controller to stop sending data to the aperiodic input endpoint and enter the global aperiodic input endpoint NAK mode.
2. Waiting for the interrupt flag USBFS_GINTSTS.GINNakEff to be set, the flag is set to indicate that the controller has stopped sending data to the non-periodic input terminal, but the controller can still send data to the non-periodic input terminal before it is set.
3. The interrupt flag USBFS_GINTSTS.GINNakEff can be set by
   USBFS_GINTMSK.GINTNakEffMsk mask,
   Set USBFS_GINTMSK.GINTNakEffMsk=0,
   Shield interrupt flag USBFS_GINTSTS.GINNakEff
   Set USBFS_GINTMSK.GINTNakEffMsk=1,
   Unmask interrupt flag USBFS_GINTSTS.GINNakEff
4. programming USBFS_DCTL.CGNPInNak=1
   USBFS_DCTL.SGNPInNak=0 ,
   To exit the global aperiodic input endpoint NAK mode,
   At the same time clear the interrupt flag USBFS_GINTSTS.GINNakEff.

29.6.16 NAK occurred

1. To stop sending data to a particular input endpoint, the program must set the NAK bit USBFS_DIEPCTLn.SNAK=1, enter the input endpoint NAK mode.
2. Wait for the interrupt flag bit USBFS_DIEPINTn.NAKIntrpt to be set, indicating that the controller has stopped sending data to this endpoint.
3. USBFS_DIEPMSK.NAKMsk = 0, can mask interrupt USBFS_DIEPINTn.NAKIntrpt,
   USBFS_DIEPMSK.NAKMsk = 1, unmask interrupt USBFS_DIEPINTn.NAKIntrpt.
4. If you want to exit the input endpoint NAK mode
   USBFS_DIEPCTLn.CNAK=1

29.6.17 Disable non-periodic input endpoints (Shared FIFO mode)

Disable endpoint operation must be after the endpoint is enabled. To disable an aperiodic input endpoint, the program must disable all aperiodic endpoints.

1. Set USBFS_DCTL.SGNPInNak=1 to enter the global aperiodic input endpoint NAK mode.
2. Wait for the interrupt flag bit USBFS_GINTSTS.GINNakEff to be set.
3. Set USBFS_DIEPCTLn.EPDis=1, USBFS_DIEPCTLn.SNAK=1, one after the other, until all aperiodic endpoints are disabled.
4. Wait for the interrupt flag bit USBFS_DIEPINTn.EPDisbld to be set, which indicates that the controller has disabled the corresponding endpoint, and clear the register bits USBFS_DIEPCTLn.EPEna and USBFS_DIEPCTLn.EPDis at the same time. It is judged in turn that all non-periodic endpoints have been disabled.

5. The program must read the values of all disabled non-periodic input endpoint registers USBFS_DIEPTSIZn, and calculate the amount of data that each endpoint has transferred.
6. The program must clear the aperiodic send FIFO, the operation is as follows:
   USBFS_GRSTCTL.TxFNum=0
   USBFS_GRSTCTL.TxFFlsh=1
   Wait for the controller to clear USBFS_GRSTCTL.TxFFlsh, Indicates that aperiodic transmit FIFO clearing has ended.
7. Program USBFS_DCTL.CGNPInNak=1 USBFS_DCTL.SGNPInNak=0, exit the global aperiodic input endpoint NAK mode.

29.6.18 Disable cycle input endpoint (Shared FIFO mode)

Disable endpoint operation must be after the endpoint is enabled.

1. Set USBFS_DIEPCTLn.SNAK=1,
   Wait for the interrupt flag bit USBFS_DIEPINTn.INEPNakEff to be set,
   This flag bit is set to indicate that the controller has stopped sending data to this endpoint.
2. Set USBFS_DIEPCTLn.EPDis=1,
   Wait for the interrupt flag bit USBFS_DIEPINTn.EPDisbld to be set,
   The flag bit is set to indicate that the controller has disabled the corresponding endpoint, and the register bit is cleared at the same time
   USBFS_DIEPCTLn.EPEna 和 USBFS_DIEPCTLn.EPDis。
3. The program must read the value of the disabled cycle input endpoint register USBFS_DIEPTSIZn, and calculate the amount of data that the endpoint has transferred.
4. The program must clear the aperiodic send FIFO, the operation is as follows:
   USBFS_GRSTCTL.TxFNum= Periodic FIFO Number
   USBFS_GRSTCTL.TxFFlsh=1
   Wait for the controller to clear USBFS_GRSTCTL.TxFFlsh, Indicates that periodic transmit FIFO clearing has ended.

29.6.19 Aperiodic input data transfer timeout (shared FIFO mode)

1. When an aperiodic input endpoint is transmitting data,
   The interrupt flag bit USBFS_DIEPCTLn.TimeOut is set, and the flag bit is set to indicate that the data transmission of the endpoint has timed out.
2. Follow the steps to Disable Aperiodic Input Endpoints (Shared FIFO Mode) .
3. Re-enable the endpoint and transmit timed-out packets.

29.6.20 Non-synchronous input endpoint pause (shared FIFO mode)

1. Set USBFS_DCTL.SGNPInNak=1 to enter the global aperiodic input endpoint NAK mode.
2. Wait for the interrupt flag bit USBFS_GINTSTS.GINNakEff to be set.

3. Set USBFS_DIEPCTLn.EPDis=1, USBFS_DIEPCTLn.Stall=1, one after the other, until all aperiodic endpoints are disabled.
4. Wait for the interrupt flag bit USBFS_DIEPINTn.EPDisbld to be set, which indicates that the controller has disabled the corresponding endpoint, and clear the register bits USBFS_DIEPCTLn.EPEna and USBFS_DIEPCTLn.EPDis at the same time. It is judged in turn that all non-periodic endpoints have been disabled.
5. The program must read the values of all disabled non-periodic input endpoint registers USBFS_DIEPTSIZn, and calculate the amount of data that each endpoint has transferred.
6. The program must clear the aperiodic send FIFO, the operation is as follows:
   USBFS_GRSTCTL.TxFNum=0
   USBFS_GRSTCTL.TxFFlsh=1
   Wait for the controller to clear USBFS_GRSTCTL.TxFFlsh, indicating that the aperiodic transmit FIFO clearing has ended.
7. Set USBFS_DCTL.SGNPInNak=0 to exit the global aperiodic input endpoint NAK mode.
8. Re-enable other endpoints that do not need to be suspended
9. To exit suspend mode, set USBFS_DIEPCTLn.Stall=0.

29.6.21 Input endpoint transfer stop (shared FIFO mode)

1. Set USBFS_DIEPCTLn.EPDis=1.
2. Wait for the interrupt flag bit USBFS_DIEPINTn.EPDisbld to be set, which indicates that the controller has disabled the corresponding endpoint, and clear the register bits DIEPCTLn.EPEna and DIEPCTLn.EPDis at the same time.
3. The program must clear the aperiodic send FIFO, the operation is as follows:
   USBFS_GRSTCTL.TxFNum=0
   USBFS_GRSTCTL.TxFFlsh=1
   Wait for the controller to clear USBFS_GRSTCTL.TxFFlsh, Indicates that aperiodic transmit FIFO clearing has ended.

29.6.22 Endpoint mismatch (shared FIFO mode)

In the shared FIFO mode, all non-periodic input endpoints share one transmit FIFO, and the program must predict the order in which the master accesses the non-periodic input endpoints. When the sequence prediction is wrong, an endpoint mismatch will occur, and the interrupt flag USBFS_DIEPINTn.INTknEPMis will be set. .

29.6.23 Control transfer

There are three types of control transfers:

- Control write transfer
- Control read transfer

- Two-Phase Control Transfer

Each control transfer consists of two or three phases:

- Setup stage
- Data stage
- Status stage

29.6.23.1 Controlling the Write Transfer Programming Flow

Control write transfer Setup phase

1. Create a cache in system memory to hold Setup packets
2. Program register USBFS_DOEPDMA0 to configure the base address of the cache list
3. XferSize, PktCnt and SUPCnt fields of programming register USBFS_DOEPTSIZ0
4. The programming register USBFS_DOEPCTL0.MPS is used to configure the maximum packet size of the control endpoint, and the programming register USBFS_DOEPCTL0.EPEna is set to 1 to enable the control output endpoint
5. Waiting for an output endpoint interrupt
6. When the controller receives the Setup token packet, it will set the USBFS_DOEPINT0 interrupt

a) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.SetUp=0, the program must read the StupPktRcvd bit of the register USBFS_DOEPINT0. If StupPktRcvd=1, it means that the Setup token package has been received. The program must decode the type of control transfer sent by the master. Then the program must check whether USBFS_DOEPINT0.StsPhaseRcvd is set to 1, if it is set to 1, it means entering the status phase, if it is not set to 1, it means entering the data phase.

b) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.SetUp=1, the program must read the StupPktRcvd bit of the USBFS_DOEPINT0 register. If StupPktRcvd=1, it means that the Setup token package has been received. The program must decode the type of control transfer sent by the master. Then the program must enter the data phase. USBFS_DOEPINT0.SetUp=1 indicates that the master sent an output token packet during the data phase of the control write transfer.

Controlling the write transfer data phase

1. Create n buffers to receive `wlength` bytes of data Make sure each buffer contains a packet [n=wlength/MPS+wlengthMOD(MPS)]
2. Program register USBFS_DOEPDMA0 to configure the base address of the descriptor list
3. XferSize, PktCnt and SUPCnt fields of programming register USBFS_DOEPTSIZ0
4. The programming register USBFS_DOEPCTL0.MPS is used to configure the maximum packet size of the control endpoint, and the programming register USBFS_DOEPCTL0.EPEna is set to 1 to enable the control output endpoint

5. Waiting for an output endpoint interrupt
6. Depending on the type of token received, the controller will generate the following interrupts

a) If USBFS_DOEPINT0.XferComp=0 and USBFS_DOEPINT0.SetUp=1, the controller replies NAK to the first output token sent by the master. Then program USBFS_DOEPCTL0.CNAK=1 to clear NAK, and then jump to step 4.

b) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.SetUp=0, read the StupPktRcvd bit of the USBFS_DOEPINT0 register. If StupPktRcvd=0, it means that the output token has been received. If this token is not the last one, jump to step 4 to receive the next one. token. If this is the last token, configure USBFS_DOEPCTL0.Stall=1, so that the controller will suspend receiving the output token packet sent by the master. The data phase is complete and the program will jump to the status phase which controls the write transfer. If StupPktRcvd=1, it indicates that a new control transfer Setup token has been received. The program must decode the type of control transfer sent by the master.

c) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.StsPhseRcvd=1, the interrupt flag USBFS_DOEPINT0.XferComp indicates that the controller received an output token during the data phase and the interrupt flag USBFS_DOEPINT0.StsPhseRcvd indicates the start of the status phase. After the last output token has been processed, the program enters the status phase which controls the write transfer.

Controlling the write transfer status phase

1. Create a buffer list in system memory containing an input buffer
2. Program register USBFS_DIEPDMA0 to configure the base address of the cache list
3. XferSize, PktCnt field of programming register USBFS_DIEPTSIZ0
4. The programming register USBFS_DIEPCTL0.MPS is used to configure the maximum packet size of the control endpoint, and the programming register USBFS_DIEPCTL0.EPEna is set to 1 to enable the control input endpoint
5. Create a buffer list in system memory containing an output buffer
6. Program register USBFS_DOEPDMA0 to configure the base address of the cache list
7. XferSize, PktCnt and SUPCnt fields of programming register USBFS_DOEPTSIZ0
8. The programming register USBFS_DOEPCTL0.MPS is used to configure the maximum packet size of the control endpoint, and the programming register USBFS_DOEPCTL0.EPEna is set to 1 to enable the control output endpoint
9. Wait for control endpoint input and output interrupts to occur at the same time
10. Depending on the type of token received, the controller will generate the following interrupts

    If USBFS_DIEPINT0.XferComp=1, only the input endpoint generates an interrupt, indicating that the DMA has processed the zero-length data packet of the input descriptor and pushed it to the corresponding transmit FIFO. The status phase of the current control transfer is

complete and the program can process the next control transfer.

If USBFS_DOEPINT0.StsPhseRcvd=1, only the output endpoint generates an interrupt, program register USBFS_DIEPCTL0.CNAK=1, accept the input status token, and jump to step 8.

If USBFS_DOEPINT0.XferComp=1, only the output endpoint generates an interrupt, indicating that the master sends a new Setup token in advance.

If USBFS_DOEPINT0.SetUp=0 and USBFS_DOEPINT0.StupPktRcvd=1, it means that the new control transfer Setup token has been received. The program must decode the type of control transfer sent by the master. The program must then proceed to the next stage.

If USBFS_DOEPINT0.SetUp=1 and USBFS_DOEPINT0.StupPktRcvd=1, it means that the new control transfer Setup token has been received.

DOEPINT0.SetUp=1 indicates that the master may send the following token packets:

- Control write transfer data phase output token packet
- Controls the read transfer data phase input token packet
- Two-phase control transfer status phase input token packet

29.6.23.2 Controlling the read transfer programming flow

Control read transfer Setup phase

1. Create a buffer list in system memory containing an output buffer
2. Program register USBFS_DOEPDMA0 to configure the base address of the cache list
3. XferSize, PktCnt and SUPCnt fields of programming register USBFS_DOEPTSIZ0
4. The programming register USBFS_DOEPCTL0.MPS is used to configure the maximum packet size of the control endpoint, and the programming register USBFS_DOEPCTL0.EPEna is set to 1 to enable the control output endpoint
5. Waiting for an output endpoint interrupt
6. When the controller receives the Setup packet, it will set the USBFS_DOEPINT0 interrupt

a) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.SetUp=0, the program must read the StupPktRcvd bit of the USBFS_DOEPINT0 register. If StupPktRcvd=1, it means that the Setup token has been received. The program must decode the type of control transfer sent by the master. The program must then enter the data phase of the control transfer.

b) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.SetUp=1, the program must read the StupPktRcvd bit of the USBFS_DOEPINT0 register. If StupPktRcvd=1, it means that the Setup token has been received. USBFS_DOEPINT0.SetUp=1 indicates that the master sent an input token packet during the data phase of the control read transfer. The program must then proceed to the next stage.

Controls the read transfer data and status phase

1. Create n buffers to receive `wlength` bytes of data Make sure each buffer contains a packet [n=wlength/MPS+wlengthMOD(MPS)]

2. XferSize, PktCnt field of programming register USBFS_DIEPTSIZ0
3. Program register USBFS_DIEPDMA0 to configure the base address of the cache list
4. Programming register USBFS_DIEPCTL0.MPS to configure the maximum packet size of the control endpoint, programming the control input endpoint register USBFS_DIEPCTL0.EPEna to 1 to enable the control input endpoint
5. Program the XferSize, PktCnt and SUPCnt fields of register USBFS_DOEPTSIZ0 to receive an advanced status phase or a new Setup token packet.
6. Program register USBFS_DOEPDMA0 to configure the base address of the cache list
7. Programming register USBFS_DOEPCTL0.MPS to configure the maximum packet size of the control endpoint, programming register USBFS_DOEPCTL0.EPEna is set to 1 to enable the control output endpoint programming register USBFS_DOEPCTL0.CNAK=1 to clear NAK
8. Wait for the input and output interrupt of the control endpoint.
9. Depending on the type of token packet received by the controller, a combination of the following interrupts will be generated:

   If USBFS_DOEPINT0.SetUp=1, and only the output endpoint generates an interrupt, it means that the master sends the input token to start the data phase of this transfer.

   Program USBFS_DIEPCTL0.CNAK=1 to clear NAK and receive the input token packet of this transmission data stage. Program USBFS_DOEPCTL0.CNAK=1 to clear NAK and receive the output token packet in the status phase of this transmission or the SETUP token packet in the Setup phase. Skip to step 9.

   If USBFS_DIEPINT0.XferComp=1, and only the input endpoint generates an interrupt, it means that the DMA has processed the input buffer data packet and pushed it to the corresponding transmit FIFO. When the last buffered XferCompl flag is generated, program USBFS_DIEPCTL0.Stall=1 to suspend responding to additional input token packets sent by the master when all data in the transmit FIFO has been sent to the master and the transmit FIFO is empty. Skip to step 9 and wait for the output endpoint interrupt. If USBFS_DOEPINT0.XferComp=1, and only the output endpoint generates an interrupt, it means that the master sends the status phase of the control transfer or a Setup token packet of a new control transfer.

a) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.SetUp=0, the program must read the StupPktRcvd bit of the USBFS_DOEPINT0 register. If StupPktRcvd=1, it means that the Setup token has been received. The program must decode the type of control transfer sent by the master. If StupPktRcvd=0, it means that the output token of the status phase of this transmission has been received. The program must prepare a buffer for the next control transfer.

b) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.SetUp=1, the program must read the StupPktRcvd bit of the USBFS_DOEPINT0 register. If StupPktRcvd=1, it means that the Setup token has been received. The program must decode the type of control transfer sent

by the master.

29.6.23.3 Two-Phase Control Transfer

Two-stage control transmission Setup stage

1. Create a buffer list in system memory containing an output buffer
2. XferSize, PktCnt and SUPCnt fields of programming register USBFS_DOEPTSIZ0
3. Program register USBFS_DOEPDMA0 to configure the base address of the cache list
4. The programming register USBFS_DOEPCTL0.MPS is used to configure the maximum packet size of the control endpoint, and the programming register USBFS_DOEPCTL0.EPEna is set to 1 to enable the control output endpoint
5. Waiting for an output endpoint interrupt
6. When the controller receives the Setup token packet, it will set the USBFS_DOEPINT0 interrupt

a) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.SetUp=0, the program must read the StupPktRcvd bit of the USBFS_DOEPINT0 register. If StupPktRcvd=1, it means that the Setup token has been received. The program must decode the type of control transfer sent by the master. According to the Wlength field, the program can determine whether the current transfer is a two-phase control transfer. In the case of a two-phase control transfer, the program must enter the status phase of the transfer.

b) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.SetUp=1, the program must read the StupPktRcvd bit of the USBFS_DOEPINT0 register. If StupPktRcvd=1, it means that the Setup token has been received. USBFS_DOEPINT0.SetUp=1 indicates that the master has sent an input token packet during the data phase of this control read transfer.

Two-phase control transfer status phase

1. Creates a buffer list in system memory containing an input buffer.
2. Program the XferSize, PktCnt fields of the USBFS_DIEPTSIZ0 register.
3. Program register USBFS_DIEPDMA0 to configure the base address of the cache list.
4. The programming register USBFS_DIEPCTL0.MPS is used to configure the maximum packet size of the control endpoint, and the programming register USBFS_DIEPCTL0.EPEna is set to 1 to enable the control output endpoint.
5. To program the output endpoint, create a buffer list in system memory containing an output buffer guaranteed to receive a new Setup token packet during the state phase.
6. Program the XferSize, PktCnt and SUPCnt words of register USBFS_DOEPTSIZ0.
7. Program register USBFS_DOEPDMA0 to configure the base address of the cache list.
8. The programming register USBFS_DOEPCTL0.MPS is used to configure the maximum packet size of the control endpoint, and the programming register USBFS_DOEPCTL0.EPEna is set to 1 to enable the control output.
9. Wait for the input and output interrupt of the control endpoint.

10. Depending on the type of token packet received by the controller, a combination of the following interrupts will be generated:

    If USBFS_DOEPINT0.Setup=1, and only the output endpoint generates an interrupt, program USBFS_DIEPCTL0.CNAK=1 to clear NAK and receive the input token packet of this transfer status phase. Skip to step 8.

    If USBFS_DIEPINT0.XferComp=1, only the input endpoint generates an interrupt, indicating that the DMA has processed the zero-length data packet in the input buffer and pushed it to the corresponding transmit FIFO. The two-phase control transfer is complete and the program can process a new control transfer.

    If USBFS_DOEPINT0.XferComp=1, and only output endpoints generate interrupts.

a) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.SetUp=0, the program must read the StupPktRcvd bit of the USBFS_DOEPINT0 register. If StupPktRcvd=1, it means that the Setup token for the new control transfer has been received. The program must decode the type of control transfer sent by the master. The program must handle new control transfers.

b) If USBFS_DOEPINT0.XferComp=1 and USBFS_DOEPINT0.SetUp=1, the program must read the StupPktRcvd bit of the USBFS_DOEPINT0 register. If StupPktRcvd=1, it means that the Setup token for the new control transfer has been received.

    USBFS_DOEPINT0.SetUp=1 indicates that the master may send the following token packets:

    – Control write transfer data phase output token packet
    – Controls the read transfer data phase input token packet
    – Two-phase control transfer status phase input token packet

29.6.24 Output data transfer

29.6.24.1 Control Setup Transfer

1. Programming register USBFS_DOEPTSIZ0

   USBFS_DOEPTSIZ0.SUPCnt = 3
2. Program register USBFS_DOEPDMA0 and register USBFS_DOEPCTL0 according to endpoint characteristics, and enable endpoint USBFS_DOEPCTL0.EPEna = 1
3. Wait for the USBFS_DOEPINT0.SETUP interrupt flag to be set to indicate that the SETUP data transfer is complete.
4. The program must read the register USBFS_DOEPTSIZ0 to determine the number of SETUP packets received and process the last received SETUP packet.
5. In DMA mode, the program must determine whether the register USBFS_DOEPINT0.Back2BackSETup is set. If set, indicates that the controller has received more than 3 back-to-back SETUP packets. In this case, the program must ignore the USBFS_DOEPTSIZ0.SUPCnt value and use the register USBFS_DOEPDMA0 to directly read the last received SETUP packet.

29.6.25 No threshold for acyclic bulk-in data transfer

1. XferSize and PktCnt fields of programming register USBFS_DIEPTSIZn, programming register USBFS_DIEPDMAn.
2. According to the endpoint characteristic programming register USBFS_DIEPCTLn, set the CNAK bit and the EPEna bit.
3. In DMA mode, program the NextEp field to ensure that the controller fetches the input endpoint data in the correct order.
4. The USBFS_DIEPINTn.TimeOut interrupt flag bit is set to indicate that a timeout condition has been detected on this endpoint. For the handling method, please refer to the chapter Aperiodic Input Data Transmission Timeout (Shared FIFO Mode).

29.6.26 Thresholds for acyclic bulk-in data transfers

1. XferSize and PktCnt fields of programming register USBFS_DIEPTSIZn, programming register USBFS_DIEPDMAn.
2. Program the register USBFS_DIEPCTLn according to the endpoint characteristics, set the CNAK bit and the EPEna bit, and program the USBFS_DIEPCTLn.TXFNum field.
3. The USBFS_DIEPINTn.XferCompl interrupt flag bit is set to indicate the completion of an aperiodic input transfer. Read the XferSize and PktCnt fields of the register USBFS_DIEPTSIZn, the values are both 0, indicating that all data has been transferred.

29.6.27 Asynchronous output data transfer without threshold

1. XferSize and PktCnt fields of programming register USBFS_DOEPTSIZn, programming register USBFS_DOEPDMAn.
2. According to the endpoint characteristic programming register USBFS_DOEPCTLn, set the CNAK bit and the EPEna bit.
3. The USBFS_DOEPINTn.XferCompl interrupt flag bit is set to indicate the completion of an asynchronous output transfer. Read the register USBFS_DOEPTSIZn to determine the size of the received data packet.

29.6.28 Asynchronous output data transfers with threshold

For the operation process, please refer to the chapter Asynchronous Output Data Transmission without Threshold.

29.6.29 Synchronous output data transfer with threshold

For the operation process, please refer to the chapter Asynchronous Output Data Transmission without Threshold.

29.6.30 Synchronous output data transfer not completed

1. The interrupt flag bit USBFS_GINTSTS.incompISOIN is set to indicate that at least one synchronous output endpoint has an incomplete transfer in the current frame.
2. If the incomplete transfer of this endpoint is because the synchronous output data has not been completely cleared, the program must clear all synchronous output data in the receive FIFO.
3. When all the data in the receive FIFO is cleared, the program can detect the USBFS_DOEPINTn.XferCompl interrupt. In this case, the program must re-enable the endpoint to receive isochronous output data in the next frame.
4. When the interrupt flag USBFS_GINTSTS.incompISOIN is set, the program must read the control registers (USBFS_DOEPCTLn) of all synchronous output endpoints to determine which endpoint has an incomplete transfer in the current frame.
5. The previous steps must be completed before the USBFS_GINTSTS.SOF interrupt flag is set to ensure that the current frame number has not changed.
6. When an incomplete transfer occurs on a synchronous output endpoint, the program must discard the data in memory and disable the endpoint. USBFS_DOEPCTLn.EPDis = 1。
7. Wait for the interrupt flag bit USBFS_DOEPINTn.EPDisbld to be set, and then re-enable the endpoint to receive new data for the next frame.

29.6.31 Periodic input data transfer without threshold

1. XferSize and PktCnt fields of programming register USBFS_DIEPTSIZn, programming register USBFS_DIEPDMAn.
2. According to the endpoint characteristic programming register USBFS_DIEPCTLn, set the CNAK bit and the EPEna bit.
3. The USBFS_DIEPINTn.INTknTXFEmp interrupt flag is set to indicate that the DMA has not transferred all the data to the transmit FIFO.
4. If the interrupt endpoint is already enabled, ignore the interrupt in step 3. If the synchronous endpoint has already been enabled, the interrupt in step 3 occurs.
5. When the interrupt flag bit USBFS_DIEPINTn.XferCompl is set, but the USBFS_DIEPINTn.INTknTXFEmp interrupt flag bit is not set, it means that the synchronous input transfer is successfully completed. Read the XferSIze and PktCnt fields of the register USBFS_DIEPTSIZn, and the values are both 0, indicating that all data has been transferred.
6. When the interrupt flag USBFS_DIEPINTn.XferCompl is set, and the USBFS_DIEPINTn.INTknTXFEmp interrupt flag is set or not set, it means that the interrupt input transmission is successfully completed. Read the XferSIze and PktCnt fields of the register USBFS_DIEPTSIZn, and the values are both 0, indicating that all data has been transferred.

7. The interrupt flag USBFS_GINTSTS.incompISOIN is set to indicate that the controller has not received at least one cycle input token packet in the current frame.

29.6.32 Periodic input data transfers have thresholds

1. XferSIze and PktCnt fields of programming register USBFS_DIEPTSIZn, programming register USBFS_DIEPDMAn.
2. According to the endpoint characteristic programming register USBFS_DIEPCTLn, set the CNAK bit and the EPEna bit. At the same time, specify the transmit FIFO number in the register USBFS_DIEPCTLn.TXFNum field.
3. The USBFS_DIEPINTn.INTknTXFEmp interrupt flag is set to indicate that the DMA has not transferred all the data to the transmit FIFO.
4. If the interrupt endpoint is already enabled, ignore the interrupt in step 3. If the synchronous endpoint has already been enabled, the interrupt in step 3 occurs.
5. When the interrupt flag bit USBFS_DIEPINTn.XferCompl is set, but the USBFS_DIEPINTn.INTknTXFEmp interrupt flag bit is not set, it means that the synchronous input transfer is successfully completed. Read the XferSize and PktCnt fields of the register USBFS_DIEPTSIZn, and the values are both 0, indicating that all data has been transferred.
6. When the interrupt flag USBFS_DIEPINTn.XferCompl is set, and the USBFS_DIEPINTn.INTknTXFEmp interrupt flag is set or not set, it means that the interrupt input transmission is successfully completed. Read the XferSize and PktCnt fields of the register USBFS_DIEPTSIZn, and the values are both 0, indicating that all data has been transferred.
7. The interrupt flag USBFS_GINTSTS.incompISOIN is set to indicate that the controller has not received at least one cycle input token packet in the current frame.
8. The interrupt flag USBFS_DIEPINTn.TxFifoUndrn is set to indicate that the synchronous transmission of the current frame has a low flow rate. The program can choose to ignore this interrupt, because this interrupt is often caused by USBFS_GINTSTS.incompISOIN at the end of the periodic frame.

29.7Register description

The application program controls the USBFS module by reading and writing the control and status registers through the AHB slave interface. All registers of the USBFS module are 32-bit registers, and their addresses are aligned by 32 bits, so they can only be accessed in 32-bit mode.

The control and status registers are divided into the following categories:

- Module global registers
- Device Mode Register
- Power and Clock Gating Control Registers

For the list of USBFS module registers and the base address, please refer to Table 29 3 List of USBFS registers.

USBFS module register base address: 0x40040000

Table 29-3 List of USBFS registers

Offset	Register name	Access	Register description
0x08	USBFS_GAHBCFG	RW	USBFS AHB control register
0x0c	USBFS_GUSBCFG	RW	USBFS USB configuration register
0x10	USBFS_GRSTCTL	RW	USBFS Reset Register
0x14	USBFS_GINTSTS	RW	USBFS module interrupt register
0x18	USBFS_GINTMSK	RW	USBFS Interrupt Mask Register
0x1c	USBFS_GRXSTSR	RW	USBFS Receive Status Debug Read Register
0x20	USBFS_GRXSTSP	RW	USBFS receive status read and pop register
0x24	USBFS_GRXFSIZ	RW	USBFS receive FIFO size register
0x28	USBFS_GNPTXFSIZ	RW	USBFS aperiodic transmit FIFO size register
0x2c	USBFS_GNPTXSTS	R	USBFS aperiodic transmit FIFO/ queue status register
0x3c	USBFS_CID	RW	USBFS Module ID Register
0x104	USBFS_DPTXFSIZ	RW	USBFS device cycle IN endpoint transmit FIFO size register
0x800	USBFS_DCFG	RW	USBFS Device Configuration Register
0x804	USBFS_DCTL	RW	USBFS Device Control Register
0x808	USBFS_DSTS	R	USBFS Device Status Register
0x810	USBFS_DIEPMSK	RW	USBFS device IN endpoint general purpose interrupt mask register
0x814	USBFS_DOEPMSK	RW	USBFS device OUT endpoint general interrupt mask register
0x818	USBFS_DAINT	R	USBFS device overall endpoint interrupt register
0x81c	USBFS_DAINTMSK	RW	USBFS device overall endpoint interrupt mask register
0x820	USBFS_DTKNQR1	R	USBFS Device Input Token Sequential Queue Read Register 1
0x824	USBFS_DTKNQR2	R	USBFS Device Input Token Sequential Queue Read Register 2

Offset	Register name	Access	Register description
0x830	USBFS_DTKNQR3	R	USBFS Device Input Token Sequential Queue Read Register 3
0x834	USBFS_DTKNQR4	R	USBFS Device Input Token Sequential Queue Read Register 4
0x900	USBFS_DIEPCTL0	RW	USBFS Device IN Endpoint 0 Control Register
0x900+n*0x20(n=1,3,5,7)	USBFS_DIEPCTLn	RW	USBFS device IN endpoint n control register
0x908+n*0x20(n=0,1,3,5,7)	USBFS_DIEPINTn	RW	USBFS device IN endpoint n interrupt register
0x910	USBFS_DIEPSIZ0	RW	USBFS device IN endpoint 0 transfer size register
0x910+n*0x20(n=1,3,5,7)	USBFS_DIEPSIZn	RW	USBFS device IN endpoint n transfer size register
0x914+n*0x20(n=1,3,5,7)	USBFS_DIEPDMAn	RW	USBFS device IN endpoint n DMA address register
0xb00	USBFS_DOEPCTL0	RW	USBFS Device OUT Endpoint 0 Control Register
0xb00+n*0x20(n=2,4,6,8)	USBFS_DOEPCTLn	RW	USBFS device OUT endpoint n control register
0xb08+n*0x20(n=0,2,4,6,8)	USBFS_DOEPINTn	RW	USBFS device OUT endpoint n interrupt register
0xb10	USBFS_DOEPTSIZ0	RW	USBFS device OUT endpoint 0 transfer size register
0xb10+n*0x20(n=2,4,6,8)	USBFS_DOEPTSIZn	RW	USBFS device OUT endpoint n transfer size register
0xb14+n*0x20(n=2,4,6,8)	USBFS_DOEPDMAn	RW	USBFS device OUT endpoint n DMA address register
0xe00	USBFS_GCCTL	RW	USBFS Power and Gating Clock Control Register

29.7.1 USBFS global registers

29.7.1.1 USBFS AHB Control Register (USBFS_GAHBCFG)

AHB Configuration Register

Offset address: 0x08

Reset value: 0x0000 0000

This register can be used to configure the module after power-up or when changing role modes. This register mainly contains configuration parameters related to the AHB system.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved							Reserved	Reserved	Reserved	DMAEN	HBSTLEN[3:0]				GINTMSK

The application must program this register before starting any AHB or USB transaction. Do not change this register after initial programming.

Bit	Marking	Place name	Function	Read and write
b31~b9	Reserved	-	The reset value must be maintained.	R
b8	Reserved	-	The reset value must be maintained.	R
b7	Reserved	-	The reset value must be maintained.	R
b6	Reserved	-	The reset value must be maintained.	R
b5	DMAEN	DMA enable	DMA enable (DMA enable) 0: Module operates in slave mode 1: The module operates in DMA mode	R/W
b4~b1	HBSTLEN	Batch length/type	Batch length / type (Burst length/type) 0000b: single 0001b: INCR 0011:INCR4 0101:INCR8 0111:INCR16 Other values: reserved	R
b0	GINTMSK	Global Interrupt Mask	Global interrupt mask This bit is used to mask or unmask global interrupts. The Interrupt Status Register is updated by the module regardless of the setting of this bit. 0: Mask interrupts triggered by the application 1: Unmask application-triggered interrupts	R/W

29.7.1.2 USBFS USB Configuration Register (USBFS_GUSBCFG)

USBFS USB configuration register

Offset address: 0x00C

Reset value: 0x0000 0A00

This register configures the module. It contains configuration parameters related to USB and USB-PHY.

The application must program this register before starting any AHB or USB transaction. Do not change this register after initial programming.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved		TRDT[3:0]				Reserved			PHYSEL	Reserved			TOCAL[2:0]		

Bit	Marking	Place name	Function	Read and write
b31~b14	Reserved	-	The reset value must be maintained.	R
b13~b10	TRDT	USB turnaround time	USB turnaround time Set the turnaround time in units of PHY clocks. To calculate the value of TRDT, use the following formula: TRDT = 4 × AHB clock + 1 PHY clock For example: 1. If AHB clock frequency = 84 MHz (PHY clock frequency = 48 MHz), then TRDT is set to 9. 2. If AHB clock frequency = 48 MHz (PHY clock frequency = 48 MHz), then TRDT is set to 5.	R/W
b9~b7	Reserved	-	The reset value must be maintained.	R
b6	PHYSEL	Full-Speed Family Transceiver Selection	Full Speed serial transceiver select This bit is write-only and is always 1.	W
b5~b3	Reserved	-	The reset value must be maintained.	R
b2~b0	TOCAL	FS timeout calibration	FS timeout calibration (FS timeout calibration) Additional latency introduced by the PHY includes the number of PHY clocks set by the application in this field, and the module's full-speed inter-packet timeout interval. The delay introduced by different PHYs has different effects on the state of the data line. The USB standard timeout value for full speed operation is 16 to 18 single digit times, inclusive. The application must program this field according to the enumerated speed. The number of bit times added per PHY clock is 0.25 bit times.	

29.7.1.3 USBFS Reset Register (USBFS_GRSTCTL)

USBFS reset register

Offset address: 0x10

Reset value: 0x8000 0000

The application program resets various hardware features in the module through this register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
AHBIDL	DMAREQ	Reserved													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved					TXFNUM[4:0]					TXFFLSH	RXFFLSH	Reserved	Reserved	HSRST	CSRST

Bit	Marking	Place name	Function	Read and write
b31	AHBIDL	AHB Master Idle	AHB master idle (AHB master idle) Indicates that the AHB master state machine is in an idle condition.	R
b30	DMAREQ	AHB Master Idle	DMA request signal (DMA request signal) This bit indicates that a DMA request is in progress. Used for debugging.	R
b29~b11	Reserved	-	Read as "0", write as "0"	R
b10~b6	TXFNUM	TxFIFO number	TxFIFO number (TxFIFO number) FIFO number for FIFO flushing using the TxFIFO flush bit. This field should only be changed after the module has cleared the TxFIFO flush bit. ● 00000: Refresh Tx FIFO 0 ● 00001: Flush TXFIFO 1 ● 00010: refresh TXFIFO 2 ... ● 00101: refresh TXFIFO 15 ● 10000: refresh all transmit FIFOs	R/W
b5	TXFFLSH	TxFIFO refresh	TxFIFO refresh (TxFIFO flush) This bit selectively flushes one or all transmit FIFOs, but cannot do so while the module is processing a communication transaction. The application can only write to this bit after confirming that the module is not currently reading or writing to the TxFIFO. Confirm with the following registers: — Read: NAK active interrupt ensures that the module is not currently performing a read operation on the FIFO — Write: The AHBIDL bit in USBFS_GRSTCTL ensures that the module is not currently doing any writes to the FIFO	R/W
b4	RXFFLSH	RxFIFO refresh	RxFIFO refresh (RxFIFO flush) The application can use this bit to flush the entire RxFIFO, but must first ensure that the module is not currently processing a communication transaction. The application can write to this bit only after confirming that the module is not currently reading or writing to the RxFIFO. The application must wait until this bit is cleared before performing other operations. Usually there is a wait of 8 clock cycles (whichever is the slowest of the PHY or AHB clock).	R/W
b3~b2	Reserved	-	The reset value must be maintained.	R
b1	HSRST	HCLK domain logic soft reset	HCLK domain logic soft reset (HCLK soft reset) Applications use this bit to refresh the control logic in the AHB clock domain. Only the AHB clock domain pipeline is reset. FIFO is not flushed by this bit. After the transaction on the AHB is terminated according to the protocol, all state machines in the AHB clock domain are reset to the idle state. The CSR control bit used by the AHB clock domain state machine is cleared. To clear this interrupt, the status mask bit generated by the AHB clock domain state machine and used to control the status of the interrupt needs to be cleared. Since the interrupt status bit is not cleared, the application can get the status of all module events that occurred after this bit was set. state.	R/W

			This bit is self-clearing and the module will clear it when all necessary logic in it is reset. This process requires several clocks of time, depending on the current state of the module.	
b0	CSRST	Module soft reset	Module soft reset (Core soft reset) Reset the HCLK and PCLK domains as follows: Clear individual interrupt and all CSR register bits except: — RSTPDMODL bit in USBFS_PCGCCTL — GAYEHCLK bit in USBFS_PCGCCTL — PWRCLMP bit in USBFS_PCGCCTL — STPPCLK bit in USBFS_PCGCCTL — FSLSPCS bit in USBFS_HCFG — DSPD bit in USBFS_DCFG Resets all module state machines (except AHB slaves) to idle state and clears all transmit and receive FIFOs. Terminate all transactions on the AHB master as soon as possible after the final data phase of the AHB transfer. Immediately terminates all transactions on the USB. The application can write to this bit anytime it needs to reset the module. This bit is self-clearing and the module will clear it after all necessary logic in it is reset, which takes several clocks depending on the current state of the module. Once this bit is cleared, software must wait at least 3 PHY clocks before accessing the PHY domain (synchronous delay). In addition, the software It must also be determined that bit 31 in this register is set (AHB Master Idle) before starting operation. Software reset is usually used in two situations, one is during software development, and the other is after the user dynamically changes the PHY selection bits in the USB configuration registers listed above. When the user changes the PHY, the corresponding clock will be selected for the PHY and used in the PHY domain. Once a new clock is selected, the PHY domain must be reset for proper operation.	R/W

29.7.1.4 USBFS Global Interrupt Status Register (USBFS_GINTSTS)

USBFS interrupt status register

Offset address: 0x14

Reset value: 0x14000020

This register is used to interrupt the application with system level events.

The FIFO status interrupts are read-only; if software reads or writes to the FIFO while these interrupts are being processed, the FIFO interrupt flag is automatically cleared.

Before enabling the interrupt bit, the application must clear the USBFS_GINTSTS register during initialization to avoid any interrupts before initialization.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WKUINT	VBUSVINT	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	DATAFSUSP	INCOMPISOUT	IISOIXFR	OEPINT	IEPINT	Reserved	Reserved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EOPF	ISOODRP	ENUMDNE	USBRST	USBSUSP	ESUSP	Reserved	Reserved	GOUTNAKEFF	GINAKEFF	Reserved	RXFLNE	SOF	Reserved	Reserved	Reserved

Bit	Marking	Place name	Function	Read and write
b31	WKUPINT	Resume/Remote Wake Interrupt Detected	Resume / remote wakeup detected interrupt When a resume signal is detected on the USB bus, the interrupt will be triggered and cleared by writing 1 to this bit by software.	R/W
b30	VBUSVINT	VBUS active interrupt	VBUSy valid interrupt (VBUS valid interrupt) In device mode, when the USBFS_VBUS pin is detected to change from low to high, the interrupt will be triggered. Write 1 to this bit to clear it to 0 by software.	R/W
b29~b23	Reserved	-	The reset value must be maintained.	R
b22	DATAFSUSP	Data fetch pending	Data fetch suspended This interrupt is only valid in DMA mode. This interrupt indicates that the module stopped fetching data for the IN endpoint due to unavailable TxFIFO space or request queue space. The application uses this interrupt in the endpoint mismatch algorithm. For example, after detecting an endpoint mismatch, the application would do the following: — Set the global aperiodic IN NAK handshake signal to 1 — Disable IN endpoints — empty FIFO — Determine the token sequence based on the IN token sequence learning queue — re-enable the endpoint — If the global aperiodic IN NAK is cleared but the module has not yet acquired data for the IN endpoint and an IN token has been received at the same time, clear the global aperiodic IN NAK handshake signal: the module will generate "FIFO is empty when receiving IN token" interrupt. USBFS then sends a NAK response to the host. To avoid this situation, the application can check the DATAFSUSP interrupt in USBFS_GINTSTS, which can ensure that the global NAK handshake signal is cleared after the FIFO is full. Alternatively, the application can mask the "IN token received while FIFO empty interrupt" while clearing the global IN NAK handshake signal. Write 1 to this bit to clear it to 0 by software.	R/W
b21	INCOMPISOOUT	Incomplete OUT synchronous transfer	INCOMPISOOUT: Incomplete isochronous OUT transfer When the module asserts this interrupt, it indicates that there is an outstanding transfer on at least one isochronous OUT endpoint in the current frame. This interrupt is triggered with the Periodic End of Frame Interrupt (EOPF) bit in this register. Write 1 to this bit to clear it to 0 by software.	R/W
b20	IISOIXFR	IN SYNC TRANSFER NOT COMPLETED	Incomplete isochronous IN transfer When the module sets this interrupt to 1, it indicates that the	R/W

			transmission on at least one synchronous IN endpoint in the current frame is not completed. This interrupt is triggered with the Periodic End of Frame Interrupt (EOPF) bit in this register. Write 1 to this bit to clear it to 0 by software.	
b19	OEPINT	OUT endpoint interrupt	OUT endpoint interrupt (OUT endpoint interrupt) When the module sets this bit, it indicates that there is an interrupt pending on an OUT endpoint in the module (in device mode). The application must read the host USBFS_DAINT register to determine the exact number of the OUT endpoint on which the interrupt occurred, and then read the corresponding USBFS_DOEPINTx register to determine the exact cause of the interrupt. The application must clear the corresponding status bit of the corresponding USBFS_DOEPINTx register before clearing this bit.	R
b18	IEPINT	IN endpoint interrupt	IN endpoint interrupt (IN endpoint interrupt) When the module sets this bit, it indicates that there is an interrupt pending on an IN endpoint in the module (in device mode). The application must read the host USBFS_DAINT register to determine the exact number of the IN endpoint where the interrupt occurred, and then read the corresponding USBFS_DIEPINTx register to determine the exact cause of the interrupt. The application must first convert the corresponding The corresponding status bit of the USBFS_DIEPINTx register must be cleared before the bit can be cleared.	R
b17~b16	Reserved	-	The reset value must be maintained.	R
b15	EOPF	Periodic end-of-frame interrupt	End of periodic frame interrupt Indicates that the current frame has reached the period specified by the Periodic Frame Interval field (PFIVL bit in USBFS_DCFG) in the USBFS_DCFG register. Write 1 to this bit to clear it to 0 by software.	R/W
b14	ISOODRP	Drop sync OUT packet interrupt	Isochronous OUT packet dropped interrupt The module sets this bit to 1 if the module cannot write a sync OUT packet to the RxFIFO due to insufficient space in the RxFIFO to accommodate the largest packet for the sync OUT endpoint. Write 1 to this bit to clear it to 0 by software.	R/W
b13	ENUMDNE	Enum complete interrupt	Enumeration done interrupt When the module sets this bit to 1, it indicates that the velocity enumeration is complete. The application must read the USBFS_DSTS register to obtain the enumerated speed. Write 1 to this bit to clear it to 0 by software.	R/W
b12	USBRST	USB reset interrupt	USB reset interrupt When the module sets this bit, it indicates that a reset signal was detected on the USB. Write 1 to this bit to clear it to 0 by software.	R/W
b11	USBSUSP	USB suspend interrupt	USB suspend interrupt When the module sets this bit, it indicates that a suspend condition has been detected on the USB. When the idle state on the USB bus remains for 3ms, the module will enter the suspend state. Write 1 to this bit to clear it to 0 by software.	R/W
b10	ESUSP	Early hang interrupt	Early suspend interrupt When the module sets this bit to 1, it indicates that the USB has been detected to be idle for 3ms.	R/W
b9~b8	Reserved	-	The reset value must be maintained.	R
b7	GONAKEFF	Global OUT NAK active interrupt	Global OUT NAK effective interrupt (Global OUT NAK effective interrupt) Indicates that the "set global OUT NAK" bit (SGONAK bit in USBFS_DCTL) set by the application in the USBFS_DCTL register has taken effect in the module. This bit is cleared by writing to the "Clear Global OUT NAK" bit in the USBFS_DCTL register (CGONAK bit in USBFS_DCTL).	R
b6	GINAKEFF	Global aperiodic IN NAK active interrupt	Global IN nonperiodic NAK effective interrupt Indicates that the "Set Global Aperiodic IN NAK" bit (SGINAK bit in USBFS_DCTL) set by the application in the USBFS_DCTL register has taken effect in the module. That is, the module has sampled the global IN NAK bit set by the application, and the result has taken effect. This bit is cleared by clearing the Clear Global Aperiodic IN NAK bit in the USBFS_DCTL register (CGINAK bit in UBSFS_DCTL). This interrupt does not necessarily indicate that a NAK handshake has been sent on the USB. The STALL bit has priority over the NAK bit.	R
b5	Reserved	-	The reset value must be maintained.	R
b4	RXFNE	RxFIFO not empty interrupt	RxFIFO non-empty interrupt (RxFIFO non-empty interrupt) Indicates that there is at least one packet in the RxFIFO waiting to be read.	R
b3	SOF	Start of frame interrupt	Start of frame interrupt When the module sets this bit, it indicates that a SOF token has been received on the USB. The application program can get the current frame number by reading the device status register. This interrupt occurs only when the module is running in FS mode.	R/W

			Write 1 to this bit to clear it to 0 by software.	
b2~b0	Reserved	-	The reset value must be maintained.	R

29.7.1.5 USBFS Global Interrupt Mask Register (USBFS_GINTMSK)

USBFS interrupt mask register

Offset address: 0x18

Reset value: 0x00000000

This register is used in conjunction with the module interrupt register to interrupt the application. If an interrupt bit is masked, the interrupt associated with that bit will not be generated.

However, the module interrupt (USBFS_GINTSTS) register bit corresponding to the interrupt will still be set.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WKUIM	VBUSVIM	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	DATAFSUSPM	INCOMPISOOUTM	IISOIXFRM	OEPIM	IEPIM	Reserved	Reserved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EOPFM	ISOODRPM	ENUMDNEM	USBRSTM	USBSUSPM	ESUSPM	Reserved	Reserved	GOUTNAKEFFM	GINAKEFFM	Reserved	RXFNEM	SOFM	Reserved	Reserved	Reserved

Bit	Marking	Place name	Function	Read and write
b31	WKUPIM	Resume/Remote Wakeup Detected Interrupt Mask	Resume/remote wakeup detected interrupt mak 0: mask interrupt 1: enable interrupt	R/W
b30	VBUSVM	VBUS active interrupt mask	VBUS valid interrupt mask (VBUS valid interrupt mask) 0: mask interrupt 1: enable interrupt	R/W
b29~b23	Reserved	-	The reset value must be maintained.	R
b22	DATAFSUSPM	Data Acquisition Pending Interrupt Mask	Data fetch suspended interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b21	INCOMPISOOUTM	Incomplete OUT synchronous transfer interrupt mask	INCOMPISOOUT: Incomplete isochronous OUT transfer interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b20	IISOIXFRM	Incomplete IN synchronous transfer interrupt mask	Incomplete isochronous IN transfer interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b19	OEPIM	OUT endpoint interrupt mask	OUT endpoint interrupt mask (OUT endpoint interrupt mask) 0: mask interrupt 1: enable interrupt	R/W
b18	IEPIM	IN endpoint interrupt mask	IN endpoint interrupt mask (IN endpoint interrupt mask) 0: mask interrupt 1: enable interrupt	R/W
b17~b16	Reserved	-	The reset value must be maintained.	R
b15	EOPFM	Periodic end-of-frame interrupt mask	End of periodic frame interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b14	ISOODRPM	Discard Sync OUT Packet Interrupt Mask	Isochronous OUT packet dropped interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b13	ENUMDNEM	Enumeration complete interrupt mask	Enumeration done interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b12	USBRSTM	USB reset interrupt mask	USB reset interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b11	USBSUSPM	USB suspend interrupt mask	USB suspend interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b10	ESUSPM	Early Suspend Interrupt Mask	Early suspend interrupt mask 0: mask interrupt	R/W

			1: enable interrupt	
b9~b8	Reserved	-	The reset value must be maintained.	R
b7	GONAKEFFM	Global OUT NAK active interrupt mask	Global OUT NAK effective interrupt mask (Global OUT NAK effective interrupt mask) 0: mask interrupt 1: enable interrupt	R/W
b6	GINAKEFFM	Global aperiodic IN NAK active interrupt mask	Global IN nonperiodic NAK effective interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b5	Reserved	-	The reset value must be maintained.	R
b4	RXFNEM	RxFIFO Not Empty Interrupt Mask	RxFIFO non-empty interrupt mask (RxFIFO non-empty interrupt mask) 0: mask interrupt 1: enable interrupt	R/W
b3	SOFM	Start of frame interrupt mask	Start of frame interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b2~b0	Reserved	-	The reset value must be maintained.	R

29.7.1.6 USBFS receive status debug read /USBFS status read and stack register (USBFS_GRXSTSR/USBFS_GRXSTSP)

USBFS Receive status debug read/USBFS status read and pop registers

Read offset address: 0x01C

The offset address of the stack: 0x020

Reset value: 0x0000 0000

Reading the receive status debug read register will return the contents of the top of the receive FIFO. Reading the Receive Status Read and Pop registers will additionally pop the data entry from the top of the RxFIFO.

When the receive FIFO is empty, the module will ignore the read or stack operation of the register and return the value 0x0000 0000. The application must only pop the receive status FIFO when the Receive FIFO Not Empty bit of the Module Interrupt Register (RXFNE bit in USBFS_GINTSTS) is set.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved											PKTSTS[3:0]				DPID[1]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DPID[0]	BCNT[11:0]											EPNUM[3:0]			

Bit	Marking	Place name	Function	Read and write
b31~b21	Reserved	-	The reset value must be maintained.	R
b20~b17	PKTSTS	Packet status	Packet status Indicates the status of received packets 0001: Global OUT NAK (trigger interrupt) 0010: OUT packet received 0011: OUT transfer completed (interrupt triggered) 0100: SETUP transaction completed (interrupt triggered) 0110: SETUP packet received Other values: reserved	R
b16~b15	DPID	Data PID	Data PID (Data PID) Indicates the data PID of the received OUT packet 00：DATA0 10：DATA1 01：DATA2 11：MDATA	R
b14~b4	BCNT	Byte count	Byte count Indicates the number of bytes of the packet received.	R
b3~b0	EPNUM	Endpoint number	Endpoint number Indicates the endpoint number to which the currently received packet belongs.	R

29.7.1.7 USBFS Receive FIFO Size Register (USBFS_GRXFSIZ)

USBFS Receive FIFO size register

Offset address: 0x024

Reset value: 0x0000 0140

This application can program the size of RAM that must be allocated to the RxFIFO.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved					RXFD[10:0]										

Bit	Marking	Place name	Function	Read and write
b31~b11	Reserved	-	The reset value must be maintained.	R
b10~b0	RXFD	RxFIFO depth	RXFD: RxFIFO depth (RxFIFO depth) In units of 32-bit words. Minimum value is 16 The maximum value is 256 Power-on reset value is the maximum Rx data FIFO depth.	R/W

29.7.1.8 USBFS aperiodic transmit FIFO size register (USBFS_GNPTXFSIZ)

USBFS Non-Periodic Transmit FIFO size register

Offset address: 0x028

Reset value: configurable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
NPTxFDep[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															

Bit	Marking	Place name	Function	Read and write
b31~b16	NPTxFDep	TxFIFO depth	Aperiodic transmit FIFO size setting Minimum 16 Maximum value 32768	R/W
b15~b0	Reserved	-	The reset value must be maintained.	R

29.7.1.9 USBFS aperiodic transmit FIFO/ queue status register (USBFS_GNPTXSTS)

USBFS Non-Periodic Transmit FIFO/Queue Status register

Offset address: 0x02c

Reset value: configurable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Res.	NPTxQTop[6:0]							NPTxQSpcAvail[7:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
NPTxFSpcAvail															

Bit	Marking	Place name	Function	Read and write
b31	Reserved		The reset value must be maintained.	R
b30~b24	NPTxQTop	Aperiodic send request queue	Aperiodic send request queue being processed by the MAC	R
b23~b16	NPTxQSpcAvail	Aperiodic send request queue free space	Indicates the free space value of the aperiodic send request queue 0: Aperiodic sending request queue is full 1: 1 position available 2: 2 position available N: n positions available (n <= 8)	R
b15~b0	NPTxFSpcAvail	-Aperiodic transmit FIFO free space	Indicates the aperiodic send FIFO free space value 0: aperiodic transmit FIFO full 1: 1 word available 2: 2 word available N: n words available (n <= 1024)	R

29.7.1.10 USBFS module ID register (USBFS_CID)

USBFS core ID register

Offset address: 0x03C

Reset value: 0x12345678

This register is a programmable user configuration ID register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PRODUCT_ID[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PRODUCT_ID[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	PRODUCT_ID	Product ID field	Product ID field (Product ID field) Application-programmable ID field.	R/W

29.7.1.11 USBFS device cycle IN endpoint transmit FIFO size register (USBFS_DPTXFSIZx) (x = 1..4)

USBFS Device Periodic Transmit FIFO size register

Offset address: 0x104+(x-1)*0x4

This application can program the size that must be allocated to the device TxFIFO.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DPTxFSize[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DPTxFStAddr[11:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	DPTxFSize	Cycle IN endpoint TxFIFO size	Device Period Transmit TxFIFO Depth (Device Period Transmit TxFIFO size) In units of 32-bit words. Minimum value is 4 The maximum value is 256	R
b15~b0	DPTxFStAddr	Period IN endpoint TxFIFOx RAM starting address	Cycle IN endpoint TxFIFOx RAM start address (Period IN endpoint FIFOx transmit RAM start address) This field contains the memory start address of the cycle IN endpoint transmit FIFOx. This address must be aligned to a 32-bit memory location.	R/R_W

29.7.2 USBFS Device Mode Register

The device mode registers affect the operation of the module in device mode.

Bit values in register descriptions are in binary unless otherwise noted.

29.7.2.1 USBFS Device Configuration Register (USBFS_DCFG)

USBFS Device configuration register

Offset address: 0x800

Reset value: 0x0820 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved			PFIVL[1:0]		DAD[6:0]							Reserved	NZLS OHSK	DSPD[1:0]	

This register configures the module into device mode after power-up, certain control commands, or enumeration. Do not change this register after initial programming.

Bit	Marking	Place name	Function	Read and write
b31~b13	Reserved	-	The reset value must be maintained.	R
b12~b11	PFIVL	periodic frame interval	Periodic frame interval Indicates the point within a frame at which the application must be notified using a periodic frame interrupt. This function can be used to determine if all isochronous communications for that frame are complete. 00: 80% frame interval 01: 85% frame interval 10: 90% frame interval 11: 95% frame interval	R/W
b10~b4	DAD	Device address	Device address The application must set this field according to the command parameter after executing each SetAddress control command.	R/W
b3	Reserved	-	The reset value must be maintained.	R
b2	NZLSOHSK	Non-zero length status OUT handshake signal	Non-zero length status OUT handshake signal (Non-zero-length status OUT handshake) During the OUT transaction of the control transfer status phase, the application can use this field to select the handshake signal to be sent after the module receives a non-zero length packet. 1: When receiving a non-zero-length state OUT transaction, reply to the STALL handshake signal, and the received OUT data packet is not sent to the application. 0: Send the received OUT packet (zero length or non-zero length) to the application, and reply the handshake signal based on the NAK and STALL bits of the endpoint in the device endpoint control register.	R/W
b1~b0	DSPD	Device speed	Device speed Indicates the speed at which the application requires the module to enumerate, or the maximum speed supported by the application. However, the actual bus speed can only be determined after the chirp sequence is completed, and this speed is based on the speed of the USB host connected to the module. 00: reserved 01: reserved 10: reserved 11: Full speed (USB 2.0 transceiver clock is 48 MHz)	R/W

29.7.2.2 USBFS Device Control Register (USBFS_DCTL)

USBFS Device control register

Offset address: 0x804

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				POPRGDNE	CGONAK	SGONAK	CGINAK	SGINAK	TCTL[2:0]			GONSTS	GINSTS	SDIS	RWUSIG

Bit	Marking	Place name	Function	Read and write
b31~b12	Reserved	-	The reset value must be maintained.	R
b11	POPRGDNE	Power-on programming complete	Power-on programming done The application uses this bit to indicate that the register has finished programming after waking up from power-down mode.	R/W
b10	CGONAK	Clear global OUT NAK	Clear global OUT NA (Clear global OUT NAK) Writing to this bit will clear the global OUT NAK.	W
b9	SGONAK	Set global OUT NAK	Set global OUT NAK (Set global OUT NAK) Writing to this bit sets the global OUT NAK. Applications use this bit to send NAK handshake signals on all OUT endpoints. The application program can only set this bit to 1 when it is sure that the global OUT NAK valid bit (GONAKEFF bit in USBFS_GINTSTS) in the module interrupt register has been cleared.	W
b8	CGINAK	Clear global IN NAK	Clear global IN NAK (Clear global IN NAK) Writing to this bit will clear the global IN NAK.	W
b7	SGINAK	Set global IN NAK	Set global IN NAK (Set global IN NAK) Writing to this field sets the global aperiodic IN NAK. Applications use this bit to cause all aperiodic IN endpoints to send NAK handshake signals. The application program can only set this bit to 1 when it is sure that the global IN NAK valid bit (GINAKEFF bit in USBFS_GINTSTS) in the module interrupt register has been cleared.	W
b6~b4	Reserved	-	The reset value must be maintained.	R
b3	GONSTS	Global OUT NAK status	Global OUT NAK status (Global OUT NAK status) 0: Handshake will be sent according to FIFO status and NAK and STALL bit settings. 1: No data is received regardless of whether there is free space in the RxFIFO. Reply NAK handshake to all received packets except SETUP transactions. All OUT packets of type isochronous will be dropped.	R
b2	GINSTS	Global IN NAK status	Global IN NAK status (Global IN NAK status) 0: Handshake will be replied based on data availability in transmit FIFO. 1: Make all aperiodic IN endpoints reply with NAK handshake signals, regardless of the availability of data in the send FIFO.	R
b1	SDIS	soft disconnect	Soft disconnect The application uses this bit to signal the USBFS module to perform a soft disconnect. When this bit is set, the host will not see that the device is attached, and the device will not receive signals on the USB. The module remains disconnected until the application program clears this bit. 0: Normal operation. Clearing this bit after a soft disconnect causes the host to receive a device connected event. After reconnecting the device, the USB host restarts device enumeration. 1: Make the host receive a device disconnect event. At full speed, the minimum time for soft disconnection is specified as follows: Suspended state: the minimum time is 1ms+2.5us Idle state: 2.5us Non-idle or suspend state: 2.5us	R/W
b0	RWUSIG	Send a remote wakeup signal	Sending a remote wakeup signal (Remote wakeup signaling) When the application program sets this bit to 1, the module will initiate a remote wake-up signal to wake up the USB host. The application must set this bit to bring the module out of the suspended state. According to the USB 2.0 specification, the application must clear this bit within 1 ms to 15	R/W

ms of setting it.

29.7.2.3 USBFS Device Status Register (USBFS_DSTS)

USBFS Device status register

Offset address: 0x808

Reset value: 0x0000 0002

This register indicates the state of the module when a USB related event occurs. When an interrupt occurs, the interrupted endpoint information must be read from the device overall interrupt (USBFS_DAINT) register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										FNSOF[13:8]					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FNSOF[7:0]								Reserved				EERR	ENUMSPD[1:0]		SUSP STS

Bit	Marking	Place name	Function	Read and write
b31~b22	Reserved	-	The reset value must be maintained.	R
b21~b8	FNSOF	Frame number of received SOF	Frame number of the received SOF (Frame number of the received SOF)	R
b7~b4	Reserved	-	The reset value must be maintained.	R
b3	EERR	Indeterminate error	Erratic error The module sets this bit to report any pending errors. Due to an indeterminate error, the USBFS controller enters the suspend state and generates an interrupt to the early suspend bit of the USBFS_GINTSTS register (ESUSP bit in USBFS_GINTSTS). If the early hang interrupt was triggered by an indeterminate error, the application can only perform a soft disconnect to resume communication.	R
b2~b1	ENUMSPD	Enumeration speed	Enumerated speed Indicates the speed that the USBFS controller is enumerated after detecting the speed through the chirp sequence. 01: reserved 10: reserved 11: Full speed (PHY clock running at 48 MHz) Other values: reserved	R
b0	SUSPSTS	Suspended state	Suspend status In device mode, this bit is set whenever a suspend condition is detected on the USB. When the idle state on the USB bus is kept for 3ms, the module will enter the suspend state. A module exits the suspended state when: — Activity on the USB cable — The application writes to the remote wakeup signal bit of the USBFS_DCTL register (RWUSIG bit in USBFS_DCTL).	R

29.7.2.4 USBFS Device IN Endpoint General Interrupt Mask Register (USBFS_DIEPMSK)

USBFS Device IN endpoint common interrupt mask register

Offset address: 0x810

Reset value: 0x0000 0000

This register works in conjunction with the individual USBFS_DIEPINTx registers for all endpoints to generate interrupts on each IN endpoint. IN endpoint interrupts in the USBFS_DIEPINTx registers can be masked by writing to the corresponding bit in this register. By default, status interrupts are masked.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved									INEPNEM	INEPNMM	ITTXFEMSK	TOM	Reserved	EPDM	XFRCM

Bit	Marking	Place name	Function	Read and write
b31~b7	Reserved	-	The reset value must be maintained.	R
b6	INEPNEM	IN endpoint NAK effective interrupt mask	IN endpoint NAK effective interrupt mask (IN endpoint NAK effective mask) 0: mask interrupt 1: enable interrupt	R/W
b5	INEPNMM	IN token interrupt mask received on EP mismatch	IN token interrupt mask received on EP mismatch (IN token received with EP mismatch mask) 0: mask interrupt 1: enable interrupt	R/W
b4	ITTXFEMSK	IN token interrupt mask received while TxFIFO is empty	IN token interrupt mask received while TxFIFO is empty (IN token received when TxFIFO empty mask) 0: mask interrupt 1: enable interrupt	R/W
b3	TOM	Timeout Interrupt Mask (Asynchronous Endpoints)	Timeout Interrupt Mask (Asynchronous Endpoints) (Timeout condition mask (Non-isochronous endpoints)) 0: mask interrupt 1: enable interrupt	R/W
b2	Reserved	-	The reset value must be maintained.	R
b1	EPDM	Endpoint disable interrupt mask	disabled interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b0	XFRCM	Transport complete interrupt mask	Transfer completed interrupt mask 0: mask interrupt 1: enable interrupt	R/W

29.7.2.5 USBFS device OUT endpoint general interrupt mask register (USBFS_DOEPMSK)

USBFS Device OUT endpoint common interrupt mask register

Offset address: 0x814

Reset value: 0x0000 0000

This register works in conjunction with the individual USBFS_DOEPINTx registers for all endpoints to generate interrupts on each OUT endpoint. OUT endpoint interrupts in the USBFS_DOEPINTx registers can be masked by writing to the corresponding bit in this register. By default, status interrupts are masked.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved											OTEPDM	STUPM	Reserved	EPDM	XFRCM

Bit	Marking	Place name	Function	Read and write
b31~b5	Reserved	-	The reset value must be maintained.	R
b4	OTEPDM	OUT token interrupt mask received when endpoint disabled	OUT token interrupt mask received when endpoint disabled (OUT token received when endpoint disabled mask) Applies to control OUT endpoints only. 0: mask interrupt 1: enable interrupt	R/W
b3	STUPM	The SETUP phase completes the interrupt mask	SETUP phase done interrupt mask (SETUP phase done mask) Applies to control endpoints only. 0: mask interrupt 1: enable interrupt	R/W
b2	Reserved	-	The reset value must be maintained.	R
b1	EPDM	Endpoint disable interrupt mask	disabled interrupt mask 0: mask interrupt 1: enable interrupt	R/W
b0	XFRCM	Transport complete interrupt mask	Transfer completed interrupt mask 0: mask interrupt 1: enable interrupt	R/W

29.7.2.6 USBFS Device Global Endpoint Interrupt Register (USBFS_DAINT)

USBFS Device OUT endpoint common interrupt mask register

Offset address: 0x818

Reset value: 0x0000 0000

When a valid event occurs on the endpoint, the USBFS_DAINT register will interrupt the application via the device OUT endpoint interrupt bit or the device IN endpoint interrupt bit in the USBFS_GINTSTS register (OEPINT or IEPINT bits in USBFS_GINTSTS, respectively). Each endpoint corresponds to an interrupt bit, and both the OUT endpoint and the IN endpoint have up to 16 interrupt bits. Bidirectional endpoints will use the corresponding IN and OUT interrupt bits. When the application sets and clears a bit in the corresponding Device Endpoint x Interrupt Register (USBFS_DIEPINTx/USBFS_DOEPINTx), the corresponding bit in this register is also set and cleared.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
OEPINT[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IEPINT[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	OEPINT	OUT endpoint interrupt bit	OUT endpoint interrupt bits (OUT endpoint interrupt bits) One bit per OUT endpoint: OUT endpoint 0 corresponds to bit 16, and OUT endpoint 4 corresponds to bit 20.	R/W
b15~b0	IEPINT	IN endpoint interrupt bit	IN endpoint interrupt bits One bit per IN endpoint: IN endpoint 0 corresponds to bit 0 and IN endpoint 4 corresponds to bit 4.	R/W

29.7.2.7 USBFS Device Global Endpoint Interrupt Mask Register (USBFS_DAINTMSK)

USBFS Device all endpoints interrupt mask register

Offset address: 0x81C

Reset value: 0x0000 0000

The USBFS_DAINTMSK register is used in conjunction with the device endpoint interrupt register to interrupt the application when an event occurs on the device endpoint. However, the USBFS_DAINT register bit corresponding to the interrupt will still be set.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
OEPINTM[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IEPINTM[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	OEPINTM	OUT endpoint interrupt mask bit	OUT endpoint interrupt mask bits One bit per OUT endpoint: OUT endpoint 0 corresponds to bit 16, and OUT endpoint 4 corresponds to bit 20. 0: mask interrupt 1: enable interrupt	R/W
b15~b0	IEPINTM	IN endpoint interrupt mask bit	IN endpoint interrupt mask bits One bit per IN endpoint: IN endpoint 0 corresponds to bit 0 and IN endpoint 4 corresponds to bit 4. 0: mask interrupt 1: enable interrupt	R/W

29.7.2.8 USBFS device IN command sequence learning queue read register 1 (USBFS_DTKNQR1)

USBFS Device IN Token Sequence Learning Queue Read register 1

Offset address: 0x820

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPTkn[23:8]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EPTkn[7:0]								Wrap Bit	Reserved		INTknWPtr[4:0]				

Bit	Marking	Place name	Function	Read and write
b31~b8	EPTkn	Endpoint command	Each 4 bits represent the endpoint value of the instruction [31:28] Endpoint value for instruction 5 [27:24] Endpoint value for instruction 4 [23:20] Endpoint value for instruction 3 [19:16] Endpoint value for instruction 2 [15:12] Endpoint value for instruction 1 [11:8] Endpoint value for instruction 0	R
b7	WrapBit	Winding bit	This bit is set to 1 to indicate write pointer wrapping. This bit will be cleared when the learning queue is cleared.	R
b6~b5	Reserved		The reset value must be maintained.	R
b4~b0	INTknWPtr	IN instruction queue write pointer	IN instruction queue write pointer	R

29.7.2.9 USBFS device IN command sequential learning queue read register 2 (USBFS_DTKNQR2)

USBFS Device IN Token Sequence Learning Queue Read register 2

Offset address: 0x824

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPTkn[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EPTkn[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	EPTkn	Endpoint command	Each 4 bits represent the endpoint value of the instruction [31:28] Endpoint value for instruction 13 [27:24] Endpoint value for instruction 12 [23:20] Endpoint value for instruction 11 [19:16] Endpoint value for instruction 10 [15:12] Endpoint value for instruction 9 [11:8] Endpoint value for instruction 8 [7:4] Endpoint value for instruction 7 [3:0] Endpoint value for instruction 6	R

29.7.2.10 USBFS device IN command sequential learning queue read register 3 (USBFS_DTKNQR3)

USBFS Device IN Token Sequence Learning Queue Read register 3

Offset address: 0x830

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPTkn[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EPTkn[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	EPTkn	Endpoint command	Each 4 bits represent the endpoint value of the instruction [31:28] Endpoint value for instruction 21 [27:24] Endpoint value for instruction 20 [23:20] Endpoint value for instruction 19 [19:16] Endpoint value for instruction 18 [15:12] Endpoint value for instruction 17 [11:8] Endpoint value for instruction 16 [7:4] Endpoint value for instruction 15 [3:0] Endpoint value for instruction 14	R

29.7.2.11 USBFS device IN instruction sequence learning queue read register 4 (USBFS_DTKNQR4)

USBFS Device IN Token Sequence Learning Queue Read register 4

Offset address: 0x834

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPTkn[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EPTkn[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	EPTkn	Endpoint command	Each 4 bits represent the endpoint value of the instruction [31:28] Endpoint value for instruction 29 [27:24] Endpoint value for instruction 28 [23:20] Endpoint value for instruction 27 [19:16] Endpoint value for instruction 26 [15:12] Endpoint value for instruction 25 [11:8] Endpoint value for instruction 24 [7:4] Endpoint value for instruction 23 [3:0] Endpoint value for instruction 22	R

29.7.2.12 USBFS Device Control IN Endpoint 0 Control Register (USBFS_DIEPCTL0)

USBFS Device control IN endpoint 0 control register

Offset address: 0x900

Reset value: 0x0000 8000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPENA	EPDIS	Reserved		SNAK	CNAK	TXFNUM[3:0]				STALL	Reserved	EPTYP[1:0]		NAKSTS	Reserved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USBAEP	Reserved													MPSIZ[1:0]	

This register is used to control Control Transfer Endpoint 0.

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable The application sets this bit to start a data send on endpoint 0. The module clears this bit before any of the following interrupts are triggered on this endpoint: — Endpoint prohibition —Transmission completion	R/W
b30	EPDIS	Endpoint prohibition	Endpoint disable An application can set this bit to stop sending data on an endpoint even before the transfer on that endpoint is complete. The application must wait for an endpoint disable interrupt to occur before the endpoint is considered disabled. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only if the Endpoint Enable bit for that endpoint is set.	R/W
b29~b28	Reserved	-	The reset value must be maintained.	R
b27	SNAK	Set the NAK bit	Set the NAK bit (Set NAK) Writing to this bit will set the NAK bit of the endpoint. With this bit, the application can control the sending of NAK handshake signals on the endpoint. The module can also set this bit of the endpoint to 1 after the endpoint receives the SETUP packet.	R/W
b26	CNAK	Clear the NAK bit	Clear the NAK bit (Clear NAK) Writing to this bit will clear the NAK bit of the endpoint.	R/W
b25~b22	TXFNUM	TxFIFO number	TxFIFO number (TxFIFO number) This value is set to the FIFO number assigned to IN endpoint 0.	R/W
b21	STALL	STALL handshake	STALL handshake (STALL handshake) The application can only set this bit to 1, the module will clear this bit to 0 when the endpoint receives a SETUP token. If the NAK bit, global IN NAK, or global OUT NAK and this bit are both set, the STALL bit takes precedence.	R/W
b20	Reserved	-	The reset value must be maintained.	R
b19~b18	EPTYP	Endpoint type	Endpoint type The hardware is set to '00', indicating a control type endpoint.	R
b17	NAKSTS	NAK state	NAK status _ Indicates the following results: 0: Module replies to non-NAK handshake according to FIFO status. 1: The module replies with a NAK handshake on this endpoint. When this bit is set (either by the application or by the module), the module stops sending data even if there is still data available in the TxFIFO. Regardless of how this bit is set, the module always responds to SETUP packets with an ACK handshake.	R
b16	Reserved	-	The reset value must be maintained.	R
b15	USBAEP	USB active endpoint	USB active endpoint This bit is always set to 1, indicating that Control Endpoint 0 is always active in all configurations and interfaces.	R
b14~b2	Reserved	-	The reset value must be maintained.	R
b1~b0	MPSIZ	Maximum Packet Size	Maximum packet size Applications must program this field to the maximum packet size for the current logical endpoint. 00: 64 bytes 01: 32 bytes	R/W

10: 16 bytes 11: 8 bytes

29.7.2.13 USBFS device IN endpoint x control register (USBFS_DIEPCTLx) (x=1..4)

USBFS Device IN endpoint x control register

Offset address: 0x900 + (endpoint number × 0x20)

Reset value: 0x0000 0080

Applications use this register to control the behavior of individual logical endpoints (except Endpoint 0).

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPENA	EPDIS	SODDFRM	SD0PID/SEVNFRM	SNAK	CNAK	TXFNUM[3:0]				STALL	Reserved	EPTYP[1:0]		NAKSTS	EONUM/DPID

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USBAEP	Reserved					MPSIZ[10:0]									

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable The application sets this bit to initiate data sending on the endpoint. The module clears this bit before any of the following interrupts are triggered on this endpoint: — SETUP phase completed — Endpoint prohibition —Transmission completion	R/W
b30	EPDIS	Endpoint prohibition	Endpoint disable An application can set this bit to stop sending data on an endpoint even before the transfer on that endpoint is complete. The application must wait for an endpoint disable interrupt to occur before the endpoint is considered disabled. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only if the Endpoint Enable bit for that endpoint is set.	R/W
b29	SODDFRM	set odd-numbered frames	odd frame Applies only to synchronous IN and OUT endpoints. Writing to this field sets the Even/odd-numbered frame (EONUM) field to odd-numbered frames.	R/W
b28	SD0PID/SEVNFRM	set DATA0 PID/SEVNFRM	Set DATA0 PID (Set DATA0 PID) Applies to interrupt/bulk IN endpoints only. Writing to this field sets the endpoint data PID (DPID) field in this register to DATA0. SEVNFRM: Set even frame (Set even frame) Applies only to synchronous IN endpoints. Writing to this field sets the Even/odd-numbered frames (EONUM) field to even frames.	R/W
b27	SNAK	Set the NAK bit	Set the NAK bit (Set NAK) Writing to this bit will set the NAK bit of the endpoint. With this bit, the application can control the sending of NAK handshake signals on the endpoint. When a transfer complete interrupt occurs or on an endpoint After receiving SETUP, the module can also set this bit of the OUT endpoint to 1	R/W
b26	CNAK	Clear the NAK bit	Clear the NAK bit (Clear NAK) Writing to this bit will clear the NAK bit of the endpoint.	R/W
b25~b22	TXFNUM	TxFIFO number	TxFIFO number (TxFIFO number) These bits are used to specify the FIFO number associated with this endpoint. A separate FIFO number must be set for each valid IN endpoint. This field is only valid for IN endpoints.	R/W
b21	STALL	STALL handshake	STALL handshake (STALL handshake) Setting this bit by the application causes the device to reply STALL to all tokens from the USB host. If the NAK bit, global IN NAK, or global OUT NAK is set at the same time as this bit, the STALL bit takes precedence. Only the application can clear this bit, not the module.	R/W
b20	Reserved	-	The reset value must be maintained.	R

b19~b18	EPTYP	Endpoint type	Endpoint type The following are the transport types supported by this logical endpoint. 00: control 01: synchronous 10: Batch 11: interrupt	R
b17	NAKSTS	NAK state	NAK status _ Indicates the following results: 0: Module replies to non-NAK handshake according to FIFO status. 1: The module replies with a NAK handshake on this endpoint. When an application or module sets this bit: For non-synchronous IN endpoints: The module stops sending any data through the IN endpoint even if there is data available in the TxFIFO. For synchronous IN endpoints: The module sends zero-length packets even if there is data available in the TxFIFO. Regardless of how this bit is set, the module always responds to SETUP packets with an ACK handshake.	R
b16	EONUM/ DPID	Even/odd-numbered frames/ Endpoint data PID	Even / odd frame Applies only to synchronous IN endpoints. Indicates the number of the frame in which the module sends/receives synchronized data for this endpoint. The application must program the even/odd-numbered frame number through the SEVNFRM and SODDFRM fields in this register for this endpoint to send/receive isochronous data. 0: even frame 1: odd-numbered frames DPID: Endpoint data PID (Endpoint data PID) Applies to interrupt/bulk IN endpoints only. Contains the PID of packets that will be received or sent on this endpoint. After an endpoint is activated, the application must program the PID of the first packet to be received or sent on this endpoint. The application programs the DATA0 or DATA1 PID using the SD0PID register field. 0：DATA0 1：DATA1	R/W
b15	USBAEP	USB active endpoint	USB active endpoint Indicates whether this endpoint is active in the current configuration and interface. The module clears this bit to 0 for all endpoints except endpoint 0 when a USB reset is detected. After receiving the SetConfiguration and SetInterface commands, the application must program the endpoint register accordingly and set this bit to 1.	R/W
b14~b11	Reserved	-	The reset value must be maintained.	R/W
b10~b0	MPSIZ	Maximum Packet Size	Maximum packet size Applications must program this field to the maximum packet size for the current logical endpoint. This value is in bytes.	

29.7.2.14 USBFS device IN endpoint x interrupt register (USBFS_DIEPINTx) (x=0..4)

USBFS Device IN endpoint x interrupt register

Offset address: 0x908 + (endpoint number × 0x20)

Reset value: 0x0000 0000

This register indicates the status of the endpoint on USB and AHB related events. The application must read this register when the IN endpoint interrupt bit (IEPINT bit in USBFS_GINTSTS) is set in the module interrupt register. Before an application can read this register, it must first read the Device Entire Endpoint Interrupt (USBFS_DAINT) register to obtain the exact endpoint number of the device endpoint x interrupt register. The application must clear the corresponding bits in this register to clear the corresponding bits in the USBFS_DAINT and USBFS_GINTSTS registers.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TXFE	NEPNE	Reserved	TTXFE	TOC	Reserved	EPDISD	XFRC

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	The reset value must be maintained.	R
b7	TXFE	Transmit FIFO is empty	Transmit FIFO is empty (Transmit FIFO empty) This interrupt is asserted when the TxFIFO for this endpoint is half or fully empty. Whether the TxFIFO is half empty or fully empty is determined by the TxFIFO blank bit in the USBFS_GAHBCFG register (TXFELVL bit in USBFS_GAHBCFG).	R
b6	INEPNE	IN endpoint NAK valid	INEPNE: IN endpoint NAK effective (IN endpoint NAK effective) This bit can be cleared when the application NAKs the IN endpoint by writing data to the CNAK bit in USBFS_DIEPCTLx. This interrupt indicates that the module has sampled a NAK that was set (by the application or the module), and the result has taken effect. This interrupt indicates that the IN endpoint NAK bit set by the application program has acted in the module. This interrupt does not guarantee that a NAK handshake was sent on the USB. The STALL bit has priority over the NAK bit. Software can also write 1 to clear this bit.	R/W
b5	Reserved	-	The reset value must be maintained.	R
b4	ITTXFE	IN token received when TxFIFO is empty	IN token received when TxFIFO is empty (IN token received when TxFIFO is empty) Applies to acyclic IN endpoints only. When the TxFIFO (periodic/aperiodic) corresponding to this endpoint is empty, an IN token is received and an interrupt is generated. Write 1 to clear by software.	
b3	TO	Time-out	Timeout condition Applies to control IN endpoints only. Indicates that this endpoint timed out on the most recently received IN token response. Write 1 to clear by software.	R/W
b2	Reserved	-	The reset value must be maintained.	R
b1	EPDISD	Endpoint disable interrupt	Endpoint disable interrupt This bit indicates that the endpoint has been disabled by the application. Write 1 to clear by software.	R/W
b0	XFRC	Transfer complete interrupt	Transfer completed interrupt This field indicates that transfers set on this endpoint have completed transfers on USB and AHB. Write 1 to clear by software.	R/W

29.7.2.15 USBFS device IN endpoint 0 transfer size register (USBFS_DIEPTSIZ0)

USBFS Device IN endpoint 0 transfer size register

Offset address: 0x910

Reset value: 0x0000 0000

The application must modify this register before enabling Endpoint 0. After enabling endpoint 0 through the endpoint enable bit (EPENA in USBFS_DIEPCTL0) in the device control endpoint 0 control register, the module modifies this register. The application can read this register only after the module has cleared the endpoint enable bit.

Non-zero endpoints use the registers of endpoints 1~5.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved											PKTCNT[1:0]		Reserved		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved									XFRSIZ[6:0]						

Bit	Marking	Place name	Function	Read and write
b31~b21	Reserved	-	The reset value must be maintained.	R
b20~b19	PKTCNT	Packet count	Packet count Indicates the number of data packets included in one data transmission of endpoint 0. This field will be decremented each time a packet (max size or short) is read from the TxFIFO.	R/W
b18~b7	Reserved	-	The reset value must be maintained.	R
b6~b0	XFRSIZ	transfer size	Transfer size Indicates the amount of data contained in one data transmission of endpoint 0, in bytes. The module interrupts the application only after the application has transferred this data. The transfer size can be set to the endpoint's maximum packet size to break at the end of each packet. The module decrements this field each time a packet from external memory is written to the TxFIFO.	R/W

29.7.2.16 USBFS device IN endpoint x transfer size register (USBFS_DIEPTSIZx) (x=1..4)

USBFS Device IN endpoint x transfer size register

Offset address: 0x910 + (endpoint number × 0x20)

Reset value: 0x0000 0000

The application must modify this register before enabling the endpoint. After enabling the endpoint through the endpoint enable bit (EPENA bit in USBFS_DIEPCTLx) in the USBFS_DIEPCTLx register, the module modifies this register. The application can read this register only after the module has cleared the endpoint enable bit.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved			PKTCNT[9:0]										XFRSIZ[18:16]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
XFRSIZ[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b29	Reserved	-	The reset value must be maintained.	R
b28~b19	PKTCNT	Packet count	Packet count Indicates the number of data packets contained in a data transmission on this endpoint. This field will be decremented each time a packet (max size or short) is read from the TxFIFO.	R/W
b18~b0	XFRSIZ	transfer size	Transfer size This field contains the amount of data contained in a data transmission of the current endpoint, in bytes. The module interrupts the application only after the application has transferred this data. The transfer size can be set to the endpoint's maximum packet size to break at the end of each packet. The module decrements this field each time a packet from external memory is written to the TxFIFO.	R/W

29.7.2.17 USBFS device IN endpoint x DMA address register (USBFS_DIEPDMAx) (x=0..4)

USBFS Device IN endpoint x transfer size register

Offset address: 0x914 + (endpoint number × 0x20)

Reset value: 0x0000 0000

This register is used to set the DMA address in device endpoint DMA mode.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DMAADDR[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DMAADDR[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	DMAADDR	DMA address	DMA address (DMA address) This bit contains the start address of the external memory area when using DMA for data storage on the endpoint. Note: For control endpoints, the storage area pointed to by this field is also used to store control OUT packets and SETUP transaction packets. When more than three SETUP packets are received consecutively, the SETUP packets in the memory will be overwritten. This register is incremented for every AHB transfer. The application must set a double word aligned address.	R/W

Note:

– When USBFS_GAHBCFG.DMAEN is 1, the value of this register should be less than 0x20002000 (RAM) /0x00010000 (FLASH).

29.7.2.18 USBFS device IN endpoint transmit FIFO status register (USBFS_DTXFSTSx)(x=0..4)

USBFS Device IN endpoint transmit FIFO status register

Offset address: 0x918 + (endpoint number × 0x20)

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
INEPTFSAV[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	The reset value must be maintained.	R
b15~b0	INEPTFSAV	IN endpoint TxFIFO free space	IN endpoint TxFIFO space available (IN endpoint TxFIFO space available) Indicates the amount of free space available in the endpoint TxFIFO. In 32-bit words: 0x0: Endpoint TxFIFO is full 0x1: 1 word available 0x2: 2 words available 0xn: n words are available	R

29.7.2.19 USBFS device control OUT endpoint 0 control register (USBFS_DOEPCTL0)

USBFS Device control OUT endpoint 0 control register

Offset address: 0xB00

Reset value: 0x0000 8000

This register is used to control Control Transfer Endpoint 0.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPENA	EPDIS	Reserved		SNAK	CNAK	Reserved				STALL	SNPM	EPTYP[1:0]		NAKSTS	Reserved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USBAEP	Reserved													MPSIZ[1:0]	

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable The application sets this bit to start a data send on endpoint 0. The module clears this bit before any of the following interrupts are triggered on this endpoint: — SETUP phase completed — Endpoint prohibition —Transmission completion	R/W
b30	EPDIS	Endpoint prohibition	Endpoint disable The application cannot disable control of OUT endpoint 0.	R
b29~b28	Reserved	-	The reset value must be maintained.	R
b27	SNAK	Set the NAK bit	Set the NAK bit (Set NAK) Writing to this bit will set the NAK bit of the endpoint. With this bit, the application can control the sending of NAK handshake signals on the endpoint. The module can also set this bit of the endpoint to 1 after the endpoint receives the SETUP packet.	R/W
b26	CNAK	Clear the NAK bit	Clear the NAK bit (Clear NAK) Writing to this bit will clear the NAK bit of the endpoint.	R/W
b25~b22	Reserved	-	The reset value must be maintained.	R
b21	STALL	STALL handshake	STALL handshake (STALL handshake) When this endpoint receives a SETUP token, the application can only set this bit, and the module will clear it. If the NAK bit, the global OUT NAK, and this bit are set at the same time, the STALL bit takes precedence. Regardless of how this bit is set, the module always responds to SETUP packets with an ACK handshake.	R/W
b20	SNPM	Monitor mode	Snoop mode This bit is used to configure the endpoint into listen mode. In listen mode, the module does not check the OUT packet for correctness before transferring it to the application memory.	R/W
b19~b18	EPTYP	Endpoint type	Endpoint type The hardware is set to '00', indicating a control type endpoint.	R/W
b17	NAKSTS	NAK state	NAK status _ Indicates the following results: 0: Module replies to non-NAK handshake according to FIFO status. 1: The module replies with a NAK handshake on this endpoint. When the application or module sets this bit, the module stops receiving data even if there is room in the RxFIFO to continue receiving packets. Regardless of how this bit is set, the module always responds to SETUP packets with an ACK handshake.	R
b16	Reserved	-	The reset value must be maintained.	R
b15	USBAEP	USB active endpoint	USB active endpoint This bit is always set to 1, indicating that Control Endpoint 0 is always active in all configurations and interfaces.	R
b14~b2	Reserved	-	The reset value must be maintained.	R
b1~b0	MPSIZ	Maximum Packet Size	Maximum packet size The maximum packet size for Control OUT Endpoint 0 is the same value programmed in Control IN Endpoint 0. 00: 64 bytes 01: 32 bytes	R/W

	10: 16 bytes 11: 8 bytes

29.7.2.20 USBFS device OUT endpoint x control register (USBFS_DOEPCTLx) (x=1..4)

USBFS Device OUT endpoint x control register

Offset address: 0xB00 + (endpoint number × 0x20)

Reset value: 0x0000 0000

Applications use this register to control the behavior of individual logical endpoints (except Endpoint 0).

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPENA	EPDIS	SODDFRM / SD1PID	SD0PID/ SEVNFRM	SNAK	CNAK	Reserved				STALL	SNPM	EPTYP[1:0]		NAKSTS	EONUM/ DPID

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USBAEP	Reserved				MPSIZ[10:0]										

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable Set by software, cleared by USBFS 0: endpoint disable 1: Endpoint enable	R/W
b30	EPDIS	Endpoint prohibition	Endpoint disable An application can set this bit to stop data sending/receiving on an endpoint even before the transfer is complete on that endpoint. The application must wait for an endpoint disable interrupt to occur before the endpoint is considered disabled. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only if the Endpoint Enable bit for that endpoint is set.	R/W
b29	SD1PID/ SODDFRM	Set DATA1 PID / set odd frame	Set DATA1 PID (Set DATA1 PID) Applies to interrupt/bulk OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. SODDFRM: Set odd frame (Set odd frame) Applies only to synchronous OUT endpoints. Writing to this field sets the Even/odd-numbered frame (EONUM) field to odd-numbered frames.	R
b28	SD0PID/ SEVNFRM	Set DATA0 PID/ SEVNFRM	Set DATA0 PID (Set DATA0 PID) Applies to interrupt/bulk OUT endpoints only. Writing to this field sets the endpoint data PID (DPID) field in this register to DATA0. SEVNFRM: Set even frame (Set even frame) Applies only to synchronous OUT endpoints. Writing to this field sets the Even/odd-numbered frames (EONUM) field to even frames.	R
b27	SNAK	Set the NAK bit	Set the NAK bit (Set NAK) Writing to this bit will set the NAK bit of the endpoint. With this bit, the application can control the sending of NAK handshake signals on the endpoint. The module can also set this bit of the OUT endpoint to 1 when a transfer complete interrupt occurs or after SETUP is received on the endpoint.	R/W
b26	CNAK	Clear the NAK bit	Clear the NAK bit (Clear NAK) Writing to this bit will clear the NAK bit of the endpoint.	R/W
b25~b22	Reserved	-	The reset value must be maintained.	R
b21	STALL	STALL handshake	STALL handshake (STALL handshake) When this endpoint receives a SETUP token, the application can only set this bit, and the module will clear it. If the NAK bit, the global OUT NAK, and this bit are set at the same time, the STALL bit takes precedence. Only the application can clear this bit, not the module.	R/W
b20	SNPM	Monitor mode	Snoop mode This bit is used to configure the endpoint into listen mode. In the monitor mode, the module will no longer check the correctness of the received data.	R/W
b19~b18	EPTYP	Endpoint type	Endpoint type The following are the transport types supported by this logical endpoint.	R/W

			00: control 01: synchronous 10: Batch 11: interrupt	
b17	NAKSTS	NAK status	NAK status _ Indicates the following results: 0: Module replies to non-NAK handshake according to FIFO status. 1: The module replies with a NAK handshake on this endpoint. When an application or module sets this bit: Even if there is room in the RxFIFO to hold the incoming packet, the module stops receiving any data on the OUT endpoint. Regardless of how this bit is set, the module always responds to SETUP packets with an ACK handshake.	R
b16	EONUM/ DPID	Even/odd-numbered frames/ Endpoint data PID	Even / odd frame Applies to synchronous OUT endpoints only. Indicates the number of the frame in which the module sends/receives synchronized data for this endpoint. The application must program the even/odd-numbered frame number through the SEVNFRM and SODDFRM fields in this register so that this endpoint transmits/receives synchronously data. 0: even frame 1: odd-numbered frames DPID: Endpoint data PID (Endpoint data PID) Applies to interrupt/bulk OUT endpoints only. Contains the PID of packets that will be received or sent on this endpoint. After an endpoint is activated, the application must program the PID of the first packet to be received or sent on this endpoint. The application programs the DATA0 or DATA1 PID using the SD0PID register field. 0：DATA0 1：DATA1	R/W
b15	USBAEP	USB active endpoint	USB active endpoint Indicates whether this endpoint is active in the current configuration and interface. The module clears this bit to 0 for all endpoints except endpoint 0 when a USB reset is detected. After receiving the SetConfiguration and SetInterface commands, the application must program the endpoint register accordingly and set this bit to 1.	R/W
b14~b11	Reserved	-	The reset value must be maintained.	R/W
b10~b0	MPSIZ	Maximum Packet Size	Maximum packet size Applications must program this field to the maximum packet size for the current logical endpoint. This value is in bytes.	

29.7.2.21 USBFS device OUT endpoint x interrupt register (USBFS_DOEPINTx) (x=0..4)

USBFS Device OUT endpoint x interrupt register

Offset address: 0xb08 + (endpoint number × 0x20)

Reset value: 0x0000 0080

This register indicates the status of the endpoint on USB and AHB related events. The application must read this register when the OUT endpoint interrupt bit (OEPINT bit in USBFS_GINTSTS) in the USBFS_GINTSTS register is set. Before the application can read this register, it must first read the USBFS_DAINT register to get the exact endpoint number of the USBFS_DOEPINTx registers. The application must clear the corresponding bits in this register before clearing the corresponding bits in the USBFS_DAINT and USBFS_GINTSTS registers.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved									B2BSTUP	Reserved	OTEPDIS	STUP	Reserved	EPDISD	XFRC

Bit	Marking	Place name	Function	Read and write
b31~b7	Reserved	-	The reset value must be maintained.	R
b6	B2BSTUP	Received consecutive SETUP packets	Received consecutive SETUP packets (Back-to-back SETUP packets received) Applies to control OUT endpoints only. This bit indicates that this endpoint has received more than three consecutive SETUP packets. Software can also write 1 to clear this bit.	R/W
b5	Reserved	-	The reset value must be maintained.	R
b4	OTEPDIS	An OUT token was received when the endpoint prohibited	OUT token received when endpoint disabled Applies to control OUT endpoints only. Indicates that an OUT token was received when the endpoint was not yet enabled, generating an interrupt. Write 1 to clear by software.	R/W
b3	STUP	SETUP stage completed	SETUP phase done (SETUP phase done) Applies to control OUT endpoints only. Indicates that the SETUP phase of the control endpoint is complete, and no consecutive SETUP packets are received in the current control transfer. On this interrupt, the application can decode the received SETUP packet. Write 1 to clear by software.	
b2	Reserved	-	The reset value must be maintained.	R
b1	EPDISD	Endpoint disable interrupt	Endpoint disable interrupt This bit indicates that the endpoint has been disabled by the application. Write 1 to clear by software.	R/W
b0	XFRC	Transfer complete interrupt	Transfer completed interrupt This field indicates that transfers set on this endpoint have completed transfers on USB and AHB. Write 1 to clear by software.	R/W

29.7.2.22 USBFS device OUT endpoint 0 transfer size register (USBFS_DOEPTSIZ0)

USBFS Device OUT endpoint 0 transfer size register

Offset address: 0xB10

Reset value: 0x0000 0000

The application must modify this register before enabling Endpoint 0. After enabling endpoint 0 through the endpoint enable bit (EPENA in USBFS_DIEPCTL0) in the device control endpoint 0 control register, the module modifies this register. The application can read this register only after the module has cleared the endpoint enable bit.

Non-zero endpoints use the registers of endpoints 1~4.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved	STUPCNT[1:0]		Reserved									PKTCNT	Reserved		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved									XFRSIZ[6:0]						

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	The reset value must be maintained.	R
b30~b29	STUPCNT	SETUP packet count	SETUP packet count This field specifies the number of SETUP packets the endpoint can receive consecutively. 01: 1 packet 10: 2 packets 11: 3 packet	
b28~b20	Reserved	-	The reset value must be maintained.	R
b19	PKTCNT	Packet count	Packet count The number of packets that should be received in one transfer. Before the endpoint is enabled, the software sets this bit. After the transmission starts, every time a data packet is received, the value of this field is automatically reduced.	R/W
b18~b7	Reserved	-	The reset value must be maintained.	R
b6~b0	XFRSIZ	transfer size	Transfer size Indicates the amount of data contained in one data transmission of endpoint 0, in bytes. The module interrupts the application only after the application has transferred this data. The transfer size can be set to the endpoint's maximum packet size to break at the end of each packet. The module decrements this field each time a packet is read from the RxFIFO and written to external memory.	R/W

29.7.2.23 USBFS device OUT endpoint x transfer size register (USBFS_DOEPTSIZx) (x=1..4)

USBFS Device OUT endpoint x transfer size register

Offset address: 0xB10 + (endpoint number × 0x20)

Reset value: 0x0000 0000

The application must modify this register before enabling the endpoint. After enabling the endpoint through the endpoint enable bit (EPENA bit in USBFS_DOEPCTLx) in the USBFS_DOEPCTLx register, the module modifies this register. The application can read this register only after the module has cleared the endpoint enable bit.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved			PKTCNT[9:0]										XFRSIZ[18:16]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
XFRSIZ[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b29	Reserved	-	The reset value must be maintained.	R
b28~b19	PKTCNT	Packet count	Packet count Indicates the number of data packets contained in a data transmission on this endpoint. This field will be decremented after every packet (max size or short) written to the RxFIFO.	R/W
b18~b0	XFRSIZ	transfer size	Transfer size This field contains the amount of data contained in a data transmission of the current endpoint, in bytes. The module interrupts the application only after receiving some data. The transfer size can be set to the endpoint's maximum packet size to break at the end of each packet. The module decrements this field each time a packet is read from the RxFIFO and written to external memory.	R/W

29.7.2.24 USBFS device OUT endpoint x DMA address register (USBFS_DOEPDMAx)(x=0..4)

USBFS Device OUT endpoint x transfer size register

Offset address: 0xB14 + (endpoint number × 0x20)

Reset value: 0xXXXX XXXX

This register is used to set the DMA address in device endpoint DMA mode.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DMAADDR[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DMAADDR[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	DMAADDR	DMA address	DMA address (DMA address) This bit contains the start address of the external memory area when using DMA for data transfer on the endpoint. Note: For control endpoints, the storage area pointed to by this field is also used to store control OUT packets and SETUP transaction packets. When more than three SETUP packets are received consecutively, the SETUP packets in the memory will be overwritten. This register is incremented for every AHB transfer. The application must set a double word aligned address.	R/W

Note:

– When USBFS_GAHBCFG.DMAEN is 1, the value of this register should be less than 0x20002000 (RAM) /0x00010000 (FLASH).

29.7.3 USBFS Clock Gating Control Register

Control HCLK and PHY clock by gating the clock control register to reduce power consumption. Bit values in register descriptions are in binary unless otherwise noted.

29.7.3.1 USBFS Clock Gating Control Register (USBFS_GCCTL)

Offset address: 0xE00

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved														GATE HCLK	STPP CLK

Bit	Marking	Place name	Function	Read and write
b31~b2	Reserved	-	The reset value must be maintained.	R
b1	GATEHCLK	Gated HCLK	Gate HCLK (Gate HCLK) When the USB communication is suspended or the session is invalid, the application program will set this bit to stop clocking the modules except the AHB bus slave interface, master interface and wake-up logic. The application clears this bit when USB communication resumes or a new session starts.	R/W
b0	STPPCLK	stop phy clock	Stop PHY clock (Stop PHY clock) The application sets this bit to stop the PH clock when USB communication is suspended, the session is invalidated, or the device is disconnected. The application program clears this bit when USB communication resumes.	R/W

30 Controller Area Network (CAN)

30.1 Introduction

CAN (Controller Area Network) bus is a bus standard that can realize mutual communication between microprocessors or devices without a host. This module follows the CAN bus protocol 2.0A and 2.0B and is upwardly compatible with CAN-FD. The CAN bus controller can handle the data sending and receiving on the bus. In this product, CAN has 8 groups of filters. Filters are used to select messages for an application to receive.

The application sends data to the bus through a high-priority primary transmit buffer (Primary Transmit Buffer, hereinafter referred to as PTB) and 4 secondary transmit buffers (Secondary Transmit Buffer, hereinafter referred to as STB), which is determined by the transmit scheduler Email sending order. The bus data is obtained through 10 receive buffers (Receive Buffer, hereinafter referred to as RB). 4 STBs and 10 RBs can be understood as a 4-level FIFO and a 10-level FIFO, and the FIFO is completely controlled by hardware.

The CAN bus controller can also support time-triggered CAN communication (Time-trigger communication).

Main Features of CAN:

- Fully support CAN2.0A/CAN2.0B protocol.
- Upwardly compatible with CAN-FD protocol.
- Support the highest communication baud rate 1Mbit/s
- Support 1~1/256 baud rate prescaler, flexible configuration of baud rate.
- 10 receive buffers
  - FIFO mode
  - Errors or unreceived data will not overwrite stored messages
- 1 high priority main transmit buffer PTB
- 4 secondary transmit buffers STB
  - FIFO mode
  - priority arbitration
- 8 sets of independent filters
  - Support 11-bit standard ID and 29-bit extended ID
  - Programmable ID CODE bits and MASK bits
- PTB/STB support single sending mode
- Support silent mode
- Support loopback mode
- Supports capturing transmission error types and locating arbitration failure locations
- Programmable error warning value

- Support ISO11898-4 specified time trigger CAN and receive timestamp

30.2CAN System Block Diagram

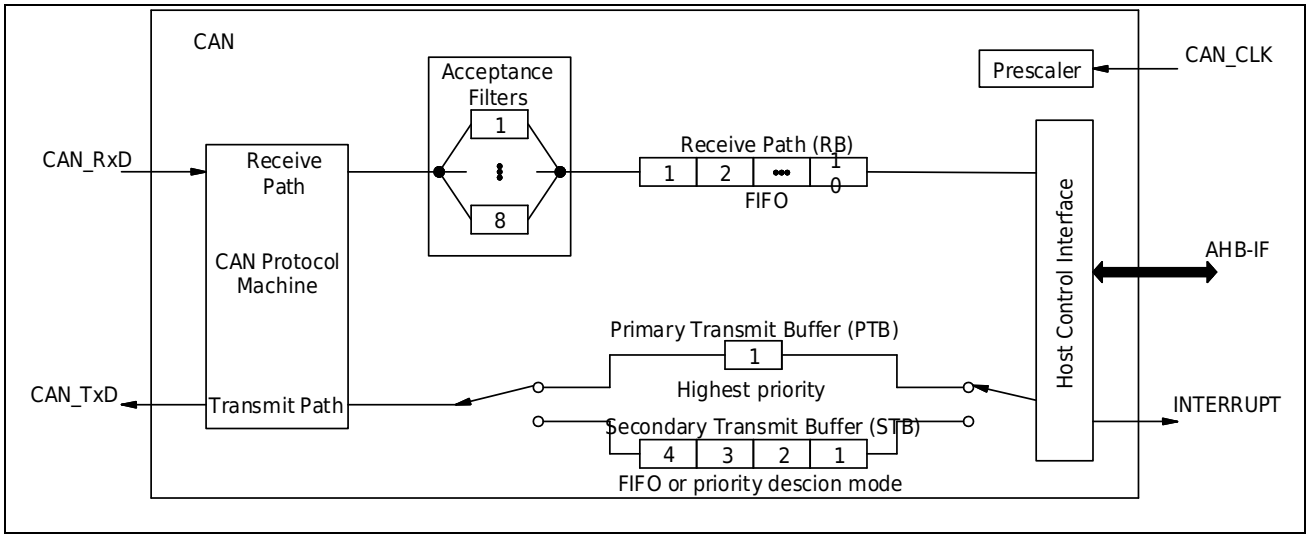


Figure 30-1 CAN System Block Diagram

30.3Pin description

Table 30-1 CAN Pin Descriptions

Pin name	Direction	Functional description
CAN_RxD	Input	CAN receive data signal
CAN_TxD	Output	CAN sends data signal

30.4Functional description

This chapter will describe the CAN function in detail.

30.4.1 Baud rate setting

The figure below shows the CAN bit time definition diagram. The part on the dotted line is the bit time specified by the CAN protocol, and the part below the dotted line is the bit time defined by the CAN controller CAN-CTRL. Among them, segment1 and segment2 can be set through the register BT. The BT register can only be set when CFG_STAT.RESET=1, that is, when the CAN software is reset.

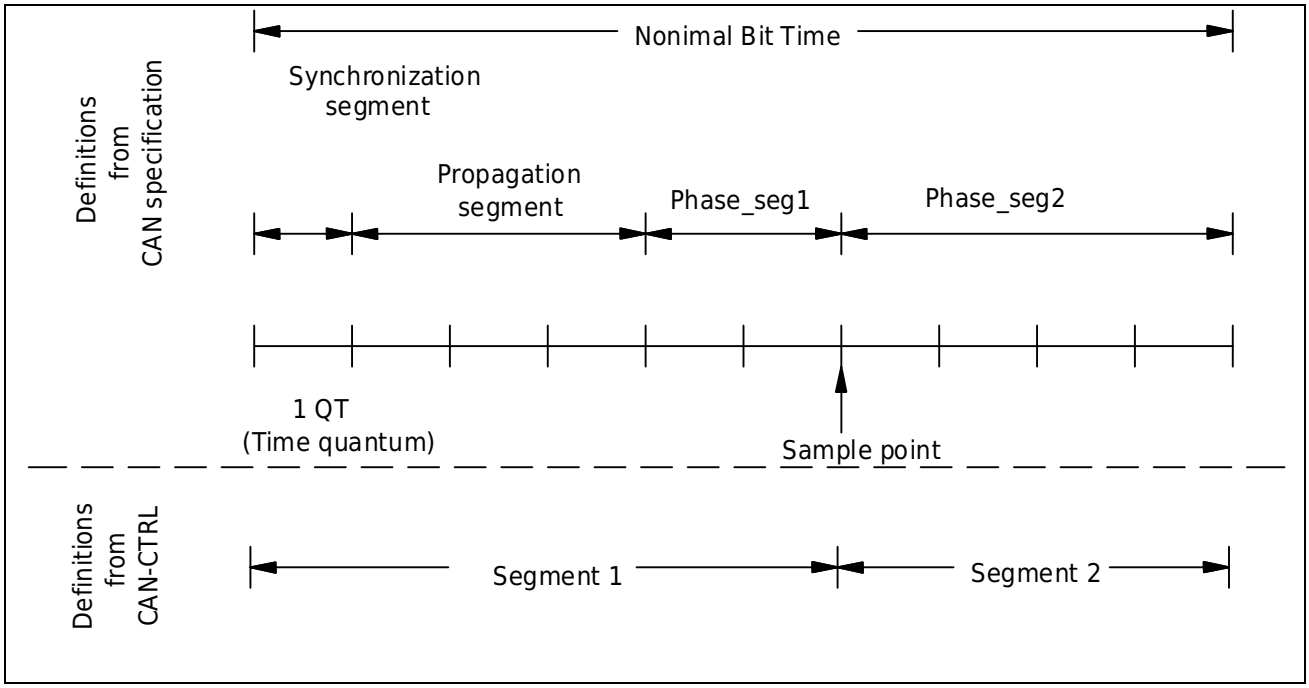


Figure 30-2 CAN bit time definition diagram

Please refer to the following formula for the TQ calculation method, where PRESC is set by the PRESC bit of the BT register . f_{can_clk} It is CAN communication clock frequency.

$$TQ=\frac{PRESC+1}{f_{can_clk}}$$

For the calculation method of sampling points, please refer to the following instructions:

1) If the PRESC bit of the BT register is set to ≥ 1, refer to the sample point shown in Figure 30-2, the sampling point is located at the dividing point between segment1 and segment2;

2) If the PRESC bit of the BT register is set to 0, the sampling point is at the sample point, that is, 2 TQ positions in front of the segment1 and segment2 boundary points.

It is recommended to set the PRESC bit of the BT register to a value ≥1.

Please refer to the following formula for the bit time calculation method, where SEG_1 and SEG_2 are set by the SEG_1 and SEG_2 bits of the BT register.

$$BT=t_{SEG1}+t_{SEG2}=\left((SEG_1+2)\ +(SEG_2+1)\right)\times TQ$$

Table 30-2 CAN bit time setting rules

Bit	Predetermined area	Rule
SEG_1 bit of BT register	0~63	$SEG_1 \geq SEG_2 + 1$ $SEG_2 \geq SJW$
SEG_2 bit of BT register	0~7	
SJW bit of BT register	0~7	

30.4.2 Send buffer

CAN_CTRL provides two sending buffers for sending data, the main sending data buffer PTB and the secondary sending buffer STB. PTB has the highest priority, but only one frame of data can be buffered. STB has a lower priority than PTB, but it can buffer 4 frames of data, and 4 frames of data in STB can work in FIFO mode or priority arbitration mode.

The 4 frames in the STB say that the data can be sent by setting the TSALL bit of the TCMD register to 1. In the FIFO mode, the first written data is sent first. In the priority mode, the data with a small ID is sent first.

Data in a PTB has the highest priority, so a PTB send can postpone an STB send, but an STB that has already won arbitration and started sending cannot be delayed by a PTB send.

A frame of data in PTB and STB takes 4 words and can be accessed through the TBUF register. Select PTB or STB through the TBSEL bit of the TCMD register, TBSEL=0, select PTB, TBSEL=1, select STB. The next SLOT in the STB is selected by the TSNEXT bit of the TCTRL register. The corresponding relationship is shown in the figure below:

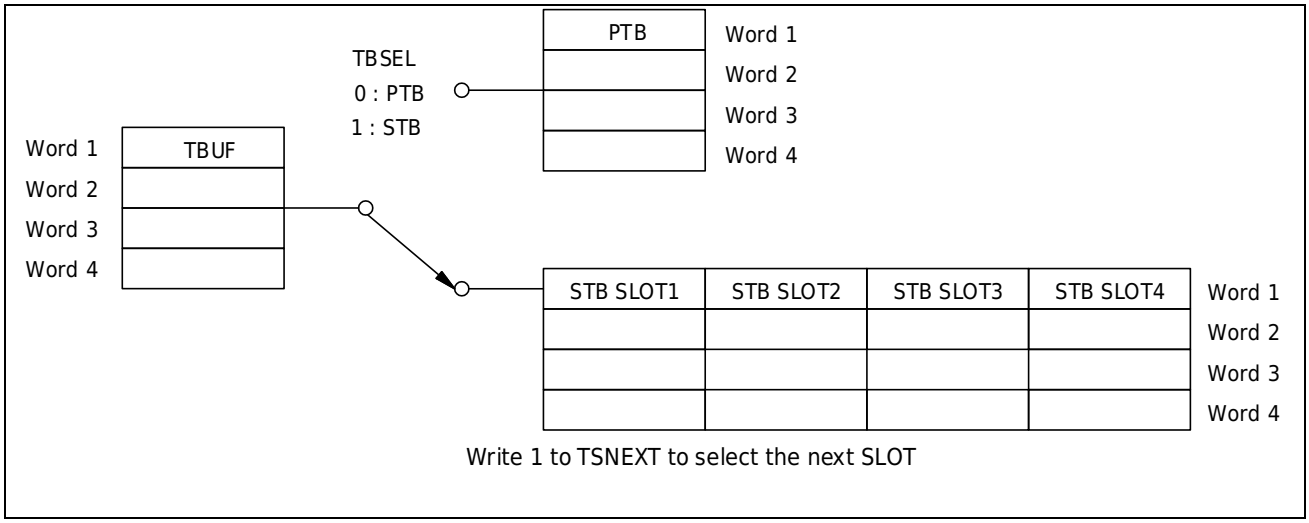


Figure 30-3 CAN TBUF register write transmit buffer and schematic diagram

30.4.3 Receive buffer

CAN_CTRL provides 10 SLOT receiving buffers for storing received data, and the 10 SLOT receiving buffers work in FIFO mode. Each RB SLOT needs to occupy 4 words, read the received data through the RBUF register, always read the earliest received data first, and set the RREL of the RCTRL register to 1 to release the read RB SLOT, and Points to the next RB SLOT.

The schematic diagram of reading RB SLOT through RBUF is as follows.

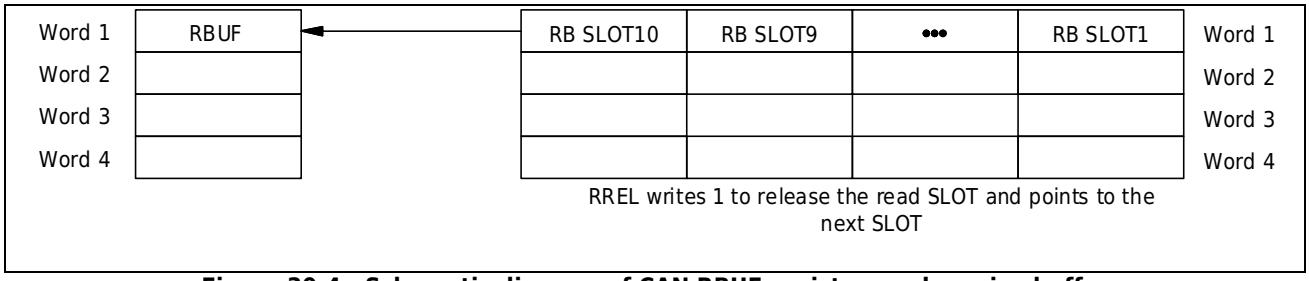


Figure 30-4 Schematic diagram of CAN RBUF register read receive buffer

30.4.4 Receive Filter Register Set

CAN_CTRL provides 8 sets of 32-bit filters to filter the received data to reduce CPU load. The filters can support standard format 11-bit ID or extended format 29-bit ID. Each set of filters has a 32-bit ID CODE register and a 32-bit ID MASK register. The ID CODE register is used to compare the received CAN ID, and the ID MASK register is used to select the CAN ID bit for comparison. When the corresponding ID MASK bit is 1, the ID CODE of this bit is not compared.

The received data will be received as long as it passes any of the 8 sets of filters, and the received data will be stored in RB, otherwise the data will not be received or stored.

Each group of filters is enabled or disabled through the ACFEN register. ID CODE and ID MASK are set by the SELMASK bit of the ACFCTRL register. When SELMASK=0, it points to ID CODE, and when SELMASK=0, it points to ID MASK. The filter is selected by the ACFADR bit of the ACFCTRL register. ID CODE and ID MASK are accessed through the ACF register and can only be set when CFG_STAT.RESET=1, that is, CAN software reset. Please refer to the figure below for the method of ACF registering and accessing the filter register group.

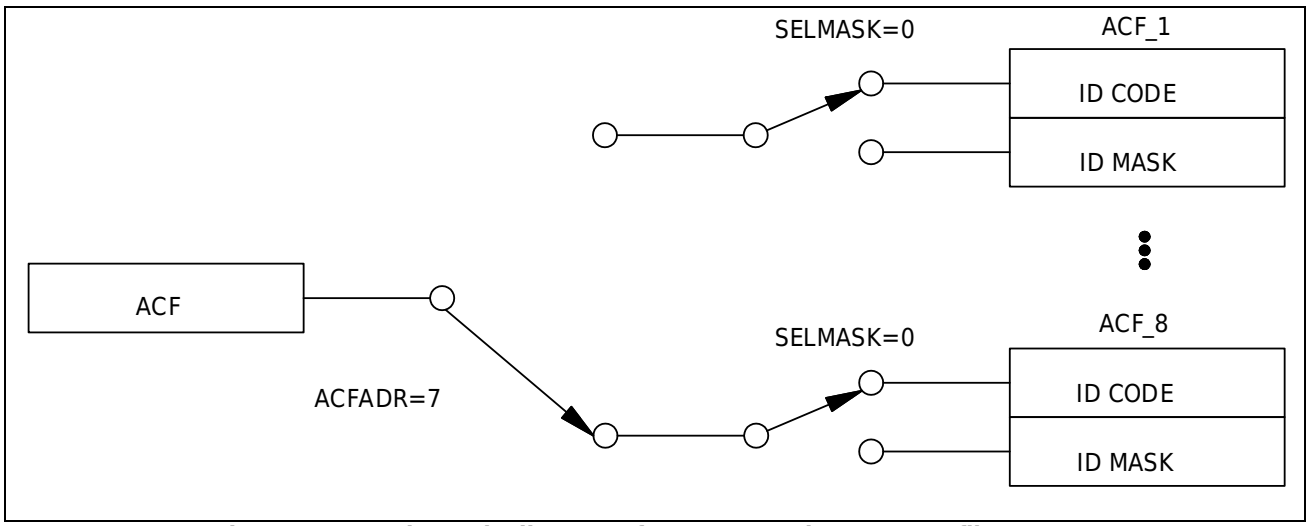


Figure 30-5 Schematic diagram of CAN ACF register access filter group

30.4.5 Data transmission

Before starting to send, it must be ensured that at least one frame of data in PTB or STB has been loaded. During PTB sending, TPE is locked, and the filling status of STB can be confirmed by TSSTAT bit. The sending data setting steps are as follows:

1. Set TBSEL to select sending BUF from PTB and STB
2. Write the data to be sent through the TBUF register.
3. STB is selected, set TSNEXT=1 to complete the loading of STB SLOT.
4. Send enable

– PTB sent using TPE
– STB sends using TSALL or TSONE

5. send complete status acknowledgment

– PTB transmission is completed using TPIF, TPIE is used to enable TPIF
– STB uses TSONE to send TSIF when it is completed, and TSIE is used to enable TSIF
– STB uses TSIF when TSALL transmission is completed. At this time, after all the STB SLOT data that needs to be set are sent, TSIF is set, and TSIE is used to enable TSIF

30.4.6 Single data transmission

When the automatic resend function is not required, it can be set to single-send mode through the register. The TPSS bit of the CFG_STAT register is used for the single-send mode setting of PTB, and the TSSS bit is used for the single-send mode of STB. When the data is successfully sent, the action of single sending and normal sending mode is the same. But when the data is not sent successfully, the following results will appear:

– When TPIF is set (TPIE=1), the corresponding BUF SLOT data will be cleared.
– When an error is sent, KOER is updated and BEIF is set (BEIE=1).
– Arbitration fails, ALIF is set (ALIE=1).

In the single sending mode, TPIF cannot be used alone to judge whether the sending is completed. It needs to judge whether the sending is completed together with BEIF and ALIF.

30.4.7 Cancel data sending

Data transmissions that have been requested but not yet executed can be canceled via TPA or TSA. Cancellation of data transmission will occur in the following situations:

- in arbitration
  – If the node arbitration fails, the data transmission is canceled.
  – If the node arbitration is successful, it will continue to send.
- data sending
  – If data is successfully sent and ACK is received, the corresponding flag and status are normally set. Data transmission is not canceled.

- If the data is sent successfully but no ACK is received, the data sending is canceled and the error counter is incremented.
- The sending data set by TSALL=1, the STB SLOT data that is being sent are sent normally, and the STB SLOT that has not started sending is canceled.

The result of canceling data sending has the following two situations.

- TPA releases PTB and makes TPE=0.
- The TSA releases one STB SLOT or all STB SLOTs depending on whether TSONE or TSALL enabled transmission.

30.4.8 Data reception

The receive filter group can filter out unnecessary received data, reduce the occurrence of interrupts and the reading of RB, thereby reducing the CPU load. The receiving data setting steps are as follows:

1. Set filter groups.
2. Set RFIE, RAIE and AFWL.
3. Wait for RFIF or RAIF.
4. Read the earliest received data from RB FIFO via RBUF.
5. Set RREL=1 to select the next RB SLOT.
6. Repeat 4, 5 until the RB is confirmed to be empty by RSTAT.

30.4.9 Error handling

On the one hand, CAN_CTRL can automatically handle some errors, such as automatically resending data or discarding received frames containing errors, and on the other hand, reports errors to the CPU through interrupts.

CAN nodes have the following three error states:

- Error Active: The node automatically sends an active error flag when it detects an error.
- Error Passive: A passive error flag is automatically sent when a node detects an error.
- Node closed: In the closed state, this node no longer affects the entire CAN network.

CAN_CTRL provides TECNT and RECNT two counters for counting errors. TECNT and RECNT counters increase or decrease according to the rules stipulated in the CAN2.0B agreement. In addition, a programmable CAN error warning LIMIT register is used to generate an error interrupt to notify the CPU.

There are the following five error types during CAN communication, and the error types can be identified by the KOER bit of the EALCAP register.

- Bit error
- Wrong form

- Filling error
- Wrong answer
- CRC error

30.4.10 Node down

When the number of sending errors is greater than 255, the CAN node automatically enters the node shutdown state and does not participate in CAN communication until it returns to the error active state. The CAN node shutdown status can be confirmed by the BUSOFF bit of the CFG_STAT register. BUSOFF is set and EIF interrupt is generated at the same time.

There are two methods for CAN to recover from the node off state to the error active state:

- Power-on reset
- Received 128 consecutive 11-bit stealth bit sequences (recovery sequence)

In the node shutdown state, the TECNT value remains unchanged, and the RECNT is used for the count recovery sequence. After recovering from node shutdown, TECNT and RECNT are reset to 0.

When the node shutdown flag BUSOFF is set, the RESET bit of the CFG_STAT register is also set.

30.4.11 Lost Arbitration Position Capture

CAN_CTRL can accurately capture the position of the arbitration failure bit and reflect it in the ALC register. The ALC register stores the position of the last arbitration failure bit. If the node wins the arbitration, the ALC bit is not updated.

The ALC value is defined as follows:

After the SOF bit, the first ID data bit ALC is 0, the second ID data bit ALC is 1, and so on. Since arbitration only takes place within the arbitration field, the maximum value of ALC is 31. For example, a standard format remote frame arbitrates with an extended frame, if the extended frame fails at the IDE bit, then ALC=12.

30.4.12 Loop mode

CAN_CTRL supports the following two loopback modes:

- Internal loopback
- External loopback

Both loopback modes can receive data frames sent by themselves, which are mainly used for testing purposes.

In the internal loopback mode, the module internally connects the receiving data line to the sending data line, and the sending data is not output. In internal loopback mode, nodes generate self-acknowledgment signals to avoid ACK errors.

The external loopback mode maintains the connection with the transceiver so that the data sent can still appear on the CAN bus. With the help of the transceiver, CAN can receive the data sent by itself. In the external loopback mode, the SACK bit of the RCTRL register can be used to determine whether to generate a self-acknowledgment signal. When SACK=0, no self-acknowledge signal is generated. When SACK=1, a self-acknowledge signal is generated.

In external loopback mode, when SACK=0, the following two situations will occur:

- Other nodes also receive the data frame sent by the node and send a response signal. In this case, the node can successfully send and receive data.
- If no other node returns an acknowledgment signal, an acknowledgment error is generated, the data is resent and the error counter is incremented. At this time, it is recommended to use the single sending mode.

When returning to normal mode from loopback mode, in addition to clearing the mode bit, a software reset of CAN_CTRL is required.

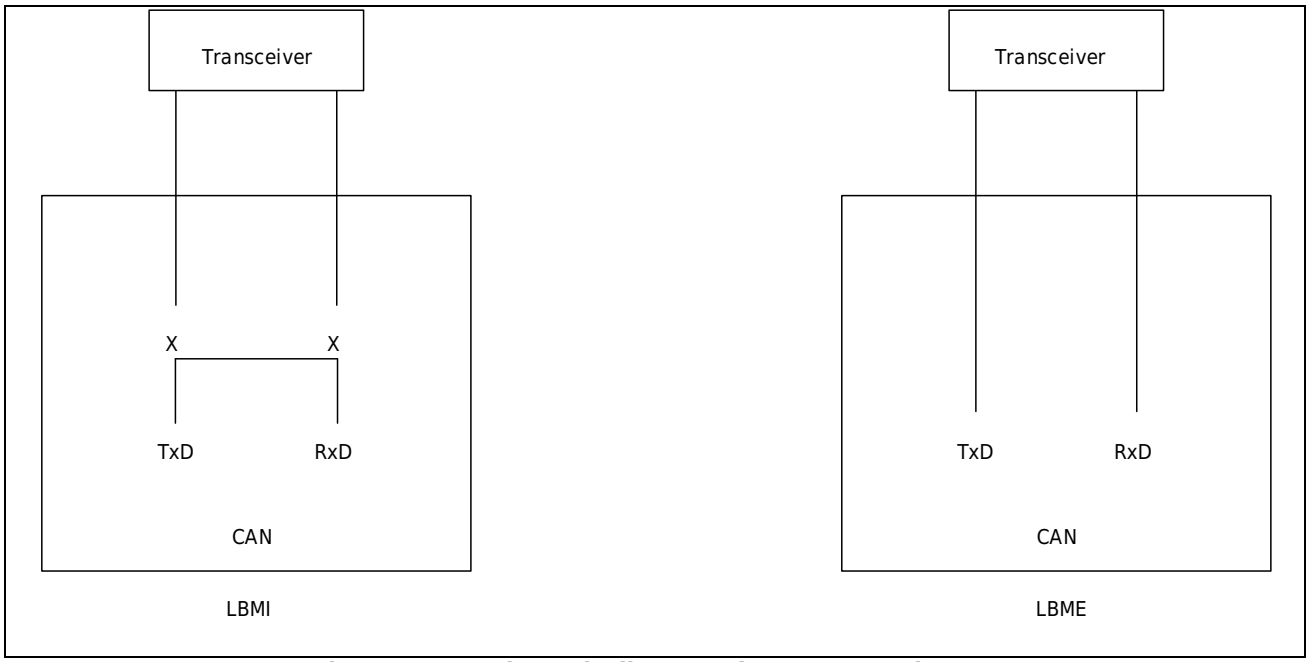


Figure 30-6 Schematic diagram of CAN LBMI and LBME

30.4.13 Silent mode

Silent mode can be used to monitor CAN network data. In silent mode, data can be received from the CAN bus without sending any data to the bus. Set the LOM in the TCMD register to 1 to make the CAN bus controller enter the silent mode, and clear it to 0 to leave the silent mode.

The external loopback mode can be combined with the silent mode to form an external loopback silent mode. At this time, CAN can be regarded as a quiet receiver, but can send data when necessary. In external loopback silent mode, frames containing self-acknowledgment signals are allowed to be sent, but the node does not generate error flags and overload frames.

30.4.14 Software reset function

By setting the RESET bit of the CFG_STAT register to 1, the software reset function is realized. The reset range of the software reset function is shown in the table below.

Table 30-3 Software reset range table

Register bit name	Software reset	Remark	Register bit name	Software reset	Remark
ACFADR	No	-	EWL	Yes	-
ACODE	No	Can only be written by software reset	KOER	Yes	-
AE_x	No	-	LBME	Yes	-
AFWL	No	-	LBMI	Yes	-
AIF	Yes	-	RACTIVE	Yes	Reception stops immediately and no ACK is generated
ALC	Yes	-	RAFIE	No	-
ALIE	No	-	RAFIF	Yes	-
ALIF	Yes	-	RBALL	Yes	-
AMASK	No	Can only be written by software reset	RBUF	Yes	All RBs are marked as empty, the value is variable
BEIE	No	-	REF_ID	No	-
BEIF	Yes	-	REF_IDE	No	-
BUSOFF	No	Cleared by writing 1	RFIE	No	-
EIEF	No	-	RFIF	Yes	-
EIF	No	-	RIE	No	-
EPASS	No	-	RIF	Yes	-
EPIE	No	-	ROIE	No	-
EPIF	Yes	-	ROIF	Yes	-
EWARN	No	No	ROM	No	-

Register bit name	Software reset	Remark	Register bit name	Software reset	Remark
ROV	Yes	-	TSNEXT	Yes	-
RREL	Yes	-	TSONE	Yes	-
PRESC	No	Can only be written by software reset	TPIE	No	-
RSTAT	Yes	-	TPIF	Yes	-
SACK	Yes	-	TPSS	Yes	-
SELMASK	No	-	TSFF	Yes	All STB SLOTs are marked empty
SEG_1	No	Can only be written by software reset	TSIE	No	-
SEG_2	No	Can only be written by software reset	TSIF	Yes	-
SJW	No	Can only be written by software reset	TSSS	Yes	-
TACTIVE	Yes	send immediately stop	TSSTAT	Yes	All STB SLOTs are marked empty
TBE	Yes	-	TTEN	Yes Yes	-
TBF	No	-	TTIF	Yes	-
TBPTR	No	-	TTIE	No	-
TBSEL	Yes	-	TTPTR	No	-
TBUF	Yes	All STBs are marked empty, pointing to PTB	TTTBM	No	-
TECNT	No	Cleared by BUSOFF=1	TTYPE	No	-
TEIF	Yes	-	TT_TRIG	No	-
TPA	Yes	-	TT_WTRIG	No	-
TPE	Yes	-	T_PRESC	No	-
TSA	Yes	-	WTIE	No	-
TSALL	Yes	-	WTIF	Yes	
TSMODE	No	-			

30.4.15 Upward compatible with CAN-FD function

Even if CAN-CTRL receives CAN-FD frames in a CAN-FD network, the receiver will automatically ignore these frames and not return ACK, and wait until the bus is free before sending or receiving the next CAN2.0B frame.

30.4.16 Time-triggered TTCAN

CAN-CTRL provides partial (level 1) hardware support for the time-triggered communication method specified in ISO11898-4. This chapter introduces TTCAN functions from the following 5 parts.

30.4.16.1 TBUF behavior in TTCAN mode

TTTBM=1

When TTTBM=1, PTB and STB SLOT form TB SLOT as well, and send BUF through the TBPTR register. When TBPTR=0, it points to PTB, TBPTR=1 points to STB SLOT1, and so on. The host can mark the

sending BUF SLOT through the TPE and TPF registers. At this time, the TBSEL and TSNEXT registers have no meaning and can be ignored.

When TTTBM=1, PTB does not have any special attributes. Like STB SLOT, the transmission completion flag also uses TSIF.

In TTCAN mode, there is no FIFO mode and priority arbitration mode for sending BUF, and only one selected SLOT can send data.

In TTCAN mode, time-triggered mode is required for transmission start, TPE, TSONE, TSALL, TPSS and TPA are fixed to 0 and ignored.

TTTBM=0

When TTTBM=0, use event-driven communication and receive timestamp function in combination. In this mode, the functions of PTB and STB are the same as when TTEN=0, so PTB always has the highest priority, and STB can work in FIFO mode or arbitration mode.

30.4.16.2 TTCAN function description

After power on, Time Master needs to be initialized according to the ISO 11898-4 protocol. In a CAN network, there can be up to 8 potential Time Masters. Each Time Master has its own reference message ID (the last 3 digits of the ID). These potential Time Masters send respective reference messages according to their own priorities.

After TTEN=1, the 16-bit counter starts to work. When the reference message is successfully received or the Time Master successfully sends the reference message, the CAN controller copies Sync_Mark to Ref_Mark, and Ref_Mark sets the cycle time to 0. The successful reception of the reference message sets the RIF flag and the successful transmission of the reference message sets the TPIF flag or the TSIF flag. At this point the host needs to prepare the trigger conditions for the next action.

The trigger condition may be receiving a trigger. This trigger-only interrupt can be used to detect that an expected message has not been received.

The trigger condition can also be a send trigger. This trigger starts sending the data in the TBUF SLOT specified by the TTPTR register. If the selected TBUF SLOT is marked empty, the transmission is not started, but the interrupt flag is set.

30.4.16.3 TTCAN Timing

CAN_CTRL supports ISO11898-4 level 1. A 16-bit counter is included operating at the bit times defined for PRESC, SEG_1, SET_2. If TTEN=1, there is an additional prescaler T_PRESC.

When SOF of one frame of data, the value of the counter is Sync_Mark. If the frame data is a reference message, copy Sync_Mark to Ref_Mark. The cycle time is equal to the value of the counter

minus Ref_Mark. This time is used as a time stamp for receiving messages or as a trigger time reference for sending messages.

30.4.16.4 TTCAN trigger mode

The trigger mode of TTCAN is defined through the TTYPE register, the TTPTR register specifies sending SLOT, and TT_TRIG specifies the cycle time of the trigger.

Contains the following five trigger methods:

- Trigger immediately
- Time trigger
- Single send trigger
- Send start trigger
- Send stop trigger

All triggers use the TTIF flag except the immediate trigger mode. When TTTBM=1, only time trigger mode is supported.

Trigger immediately

By writing the high bit of TT_TRIG (don't care about the value written), the trigger is started. In this mode, the data in the TBUF SLOT selected by TTPTR will be sent immediately. TTIF is not set.

Time trigger

The time trigger mode only generates interrupts by setting the TTIF flag, and has no other functions. Time triggering can be used if a node expects to receive expected data within a specific time window. If the TT_TRIG value is less than the actual cycle time, TEIF is set and there is no other action.

Single send trigger

The one-shot trigger method is used to send data within the execution time window. At this time, the TSSS bit is ignored.

Set the Tick of up to 16 cycle times specified by ISO11898-4 through the TEW bit, and the setting range is 1~16. If data transmission does not start within the specified transmit enable time window, the frame is discarded. The corresponding sending BUF SLOT is marked as empty, and AIF is set, and the data in the corresponding sending BUF will not be rewritten, because it can be sent again by setting TPF.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and there is no other action.

Send start trigger

The sending start trigger mode is used in the arbitration time window to participate in the arbitration. TSSS is used to decide whether to automatically repeat or send a single mode. A send stop trigger can be used to stop the send if the specified message was not sent successfully.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and there is no other action.

Send stop trigger

The send stop trigger method is used to stop the send that has been started by the send start trigger method. If the sending is stopped, the sending frame is discarded, the AIF is set and the selected TBUF SLOT is marked as empty, but the data in the TBUF SLOT will not be rewritten, and it can be sent again by setting the TPF.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and execution stops.

30.4.16.5 TTCAN trigger gate time

The TTCAN trigger watchdog function is similar to the watchdog function and is used when TTTBM=1. Used to watch the time since the last successful reference message received by the gate. The reference message can be received during the cycle time or after an event, and the application should set the appropriate gate time according to the specific situation.

If cycle count is equal to TT_WTRIG, set WTIF. Write 0 by WTIE to disable the gate trigger.

If TT_WTRIG is smaller than the actual cycle time, TEIF is set.

30.4.17 Interrupt

Table 30-4 CAN Interrupt Table

Interrupt logo	Description
RIF	Receive interrupt
ROIF	Receive overflow interrupt
ROIF	Receive BUF full interrupt
RAFIF	Receive BUF will be full interrupt
TPIF	PTB send interrupt
TSIF	STB send interrupt
EIF	error interrupt
AIF	Cancel send interrupt
EPIE	Error passive interrupt
ALIF	Arbitration Lost Interrupt
BEIF	Bus error interrupt
WTIF	Trigger gate interrupt
TEIF	Trigger error interrupt
TTIF	Time triggered interrupt

30.5Register description

CAN_BASE_ADDR: 0x4003_0000

Table 30-5 List of CAN registers

Register name	Symbol	Offset address	Bit width	Reset value
CAN receive BUF register	CAN_RBUF	0x00~0x0F	128	0xXXXX XXXX
CAN send BUF register	CAN_TBUF	0x50~0x5F	128	0xXXXX XXXX
CAN Configuration and Status Register	CAN_CFG_STAT	0xA0	8	0x80
CAN command register	CAN_TCMD	0xA1	8	0x00
CAN send control register	CAN_TCTRL	0xA2	8	0x90
CAN receive control register	CAN_RCTRL	0xA3	8	0x00
CAN receive and transmit interrupt enable register	CAN_RTIE	0xA4	8	0xFE
CAN Receive and Transmit Interrupt Flag Register	CAN_RTIF	0xA5	8	0x00
CAN Error Interrupt Enable and Flags Register	CAN_ERRINT	0xA6	8	0x00
CAN Warning Limit Register	CAN_LIMIT	0xA7	8	0x1B
CAN Bit Timing Register	CAN_BT	0xA8	32	0x0102 0203
CAN Error and Arbitration Lost Capture Register	CAN_EALCAP	0xB0	8	0x00
CAN receive error counter register	CAN_RECNT	0xB2	8	0x00
CAN Transmit Error Counter Register	CAN_TECNT	0xB3	8	0x00
CAN Filter Group Control Register	CAN_ACFCTRL	0xB4	8	0x00
CAN Filter Group Enable Register	CAN_ACFEN	0xB6	8	0x01
CAN filter group code and mask registers	CAN_ACF	0xB8	32	0xXXXX XXXX
TTCAN TB slot pointer register	CAN_TBSLOT	0xBE	8	0x00
TTCAN Time Trigger Configuration Register	CAN_TTCFG	0xBF	8	0x90
TTCAN Reference Message Register	CAN_REF_MSG	0xC0	32	0xXXXX XXXX
TTCAN trigger configuration register	CAN_TRG_CFG	0xC4	16	0x0000
TTCAN Trigger Time Register	CAN_TRG_TRIG	0xC6	16	0x0000
TTCAN Trigger Watchdog Time Register	CAN_TRG_WTRIG	0xC8	16	0x0000

Table 30-6 CAN register BYTE/HALFWORD/WORD access arrangement table

Address	BYTE access	HALFWORD access		WORD access			
0x00~0x0F	CAN_RBUF	CAN_RBUF		CAN_RBUF			
0x50~0x5F	CAN_TBUF	CAN_TBUF		CAN_TBUF			
0xA0	CAN_CFG_STAT	CAN_TCMD	CAN_CFG_STAT	CAN_RCTRL	CAN_TCTRL	CAN_TCMD	CAN_CFG_STAT
0xA1	CAN_TCMD	-		-			
0xA2	CAN_TCTRL	CAN_RCTRL	CAN_TCTRL	-			
0xA3	CAN_RCTRL	-		-			
0xA4	CAN_RTIE	CAN_RTIF	CAN_RTIE	CAN_LIMIT	CAN_ERRINT	CAN_RTIF	CAN_RTIE
0xA5	CAN_RTIF	-		-			
0xA6	CAN_ERRINT	CAN_LIMIT	CAN_ERRINT	-			
0xA7	CAN_LIMIT	-		-			
0xA8	CAN_BT[7:0]	CAN_BT[15:0]		CAN_BT			
0xA9	CAN_BT[15:8]			-			
0xAA	CAN_BT[23:16]	-		-			
0xAB	CAN_BT[31:24]	CAN_BT[31:16]		-			
0xB0	-	-		CAN_TECNT	CAN_RECNT	-	
0xB1	-	-		-			
0xB2	CAN_RECNT	CAN_TECNT	CAN_RECNT	-			
0xB3	CAN_TECNT	-		-			
0xB4	CAN_ACFCTRL[7:0]	CAN_ACFCTRL		CAN_ACFEN		CAN_ACFCTRL	
0xB5	CAN_ACFCTRL[15:8]	-		-			
0xB6	CAN_ACFEN[7:0]	CAN_ACFEN		-			
0xB7	CAN_ACFEN[15:8]	-		-			
0xB8	CAN_ACF	CAN_ACF		CAN_ACF			
0xBC	-	-		CAN_TTCFG	CAN_TBSLOT	-	-
0xBD	-	-		-			
0xBE	CAN_TBSLOT	CAN_TTCFG	CAN_TBSLOT	-			
0xBF	CAN_TTCFG			-			
0xC0	CAN_REF_MSG[7:0]	CAN_REF_MSG[15:0]		CAN_REF_MSG			
0xC1	CAN_REF_MSG[15:8]	-		-			
0xC2	CAN_REF_MSG[23:16]	CAN_REF_MSG[31:16]		-			
0xC3	CAN_REF_MSG[31:24]	-		-			
0xC4	CAN_TRG_CFG[7:0]	CAN_TRG_CFG		CAN_TRG_TRIG		CAN_TRG_CFG	
0xC5	CAN_TRG_CFG[15:8]	-		-			
0xC6	CAN_TRG_TRIG[7:0]	CAN_TRG_TRIG		-			
0xC7	CAN_TRG_TRIG[15:8]	-		-			
0xC8	CAN_TRG_WTRIG[7:0]	CAN_TRG_WTRIG		-		CAN_TRG_WTRIG	
0xC9	CAN_TRG_WTRIG[15:8]	-		-			

30.5.1 CAN receive BUF register (CAN_RBUF)

CAN Receive Buffer Registers

Offset address: 0x00~0x0F

Reset value: 0xXXXX XXXX

The RBUF register points to the RB SLOT address of the earliest received CAN mailbox, and the RBUF register can be read in any order.

The KOER bit is the register EALCAP.KOER, which is only meaningful when RBALL=1.

The TX bit indicates that the mailbox sent by itself is received in the loopback mode.

The CYCLE_TIME bit is only valid in TTCAN mode, indicating the cycle time when SOF starts.

The data format of the CAN receiving mailbox is as follows:

Table 30-7 Standard Format CAN Receiving Mailbox Format

Address	b7	b6	b5	b4	b3	b2	b1	b0	Function
RBUF	ID[7:0]								ID
RBUF+1	-					ID[10:8]			ID
RBUF+2	-								ID
RBUF+3	-								ID
RBUF+4	IDE=0	RTR	0	0	DLC[3:0]				Control
RBUF+5	KOER[2:0] TX			TX	-				Status
RBUF+6	CYCLE_TIME[7:0]								TTCAN
RBUF+7	CYCLE_TIME[15:8]								TTCAN
RBUF+8	DATA1								Data
RBUF+9	DATA2								Data
RBUF+10	DATA3								Data
RBUF+11	DATA4								Data
RBUF+12	DATA5								Data
RBUF+13	DATA6								Data
RBUF+14	DATA7								Data
RBUF+15	DATA8								Data

Table 30-8 Extended Format CAN Receive Mailbox Format

Address	b7	b6	b5	b4	b3	b2	b1	b0	Function
RBUF	ID[7:0]								ID
RBUF+1	ID[15:8] ID[10:8]								ID
RBUF+2	ID[23:16]								ID
RBUF+3	-			ID[28:24]					ID
RBUF+4	IDE=1	RTR	0	0	DLC[3:0]				Control
RBUF+5	KOER[2:0] TX			TX	-				Status
RBUF+6	CYCLE_TIME[7:0]								TTCAN
RBUF+7	CYCLE_TIME[15:8]								TTCAN
RBUF+8	DATA1								Data
RBUF+9	DATA2								Data
RBUF+10	DATA3								Data
RBUF+11	DATA4								Data
RBUF+12	DATA5								Data
RBUF+13	DATA6								Data
RBUF+14	DATA7								Data
RBUF+15	DATA8								Data

The meanings of the control bits are as follows:

IDE(IDentifier Extension):

0: standard format

1: Extended format

RTR(Remote Transmission Request)

0: data frame

1: remote frame

DLC(Data Length Code):

Data length code, the setting range is 0~8, and the corresponding data length is 0Byte~8Byte

30.5.2 CAN transmit BUF register (CAN_TBUF)

CAN Transmit Buffer Registers

Offset address: 0x50~0x5F

Reset value: 0xXXXX XXXX

The TBUF register points to the next empty CAN sending BUF SLOT, and the TBUF register can be read in any order. Write 1 to TSNEXT by software to mark that the corresponding TBUF SLOT has written data, thus pointing to the next TBUF SLOT.

TBUF can only be accessed by WORD.

The data format of the CAN sending mailbox is as follows:

Table 30-9 Standard format CAN sending mailbox format

Address	b7	b6	b5	b4	b3	b2	b1	b0	Function
TBUF	ID[7:0]								ID
TBUF+1	-					ID[10:8]			ID
TBUF+2	-								ID
TBUF+3	- -								ID
TBUF+4	IDE=0	RTR	0	0	DLC[3:0]				Control
TBUF+5	- TX TX -								-
TBUF+6	-								-
TBUF+7	-								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
TBUF+11	DATA4								Data
TBUF+12	DATA5								Data
TBUF+13	DATA6								Data
TBUF+14	DATA7								Data
TBUF+15	DATA8								Data

Table 30-10 Extended Format CAN Send Mailbox Format

Address	b7	b6	b5	b4	b3	b2	b1	b0	Function
TBUF	ID[7:0]								ID
TBUF+1	ID[15:8] ID[10:8]								ID
TBUF+2	ID[23:16]								ID
TBUF+3	-			ID[28:24]					ID
TBUF+4	IDE=0	RTR	0	0	DLC[3:0]				Control
TBUF+5	- TX TX -								-
TBUF+6	-								-
TBUF+7	-								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
TBUF+11	DATA4								Data
TBUF+12	DATA5								Data
TBUF+13	DATA6								Data
TBUF+14	DATA7								Data
TBUF+15	DATA8								Data

The meanings of the control bits are as follows:

IDE(IDentifier Extension):

0: standard format

1: Extended format

RTR(Remote Transmission Request)

0: data frame

1: remote frame

DLC(Data Length Code):

Data length code, the setting range is 0~8, and the corresponding data length is 0Byte~8Byte

30.5.3 CAN Configuration and Status Register (CAN_CFG_STAT)

CAN Configuration and Status Register

Offset address: 0xA0

Reset value: 0x80

b7	b6	b5	b4	b3	b2	b1	b0
RESET	LBME	LBMI	TPSS	TSSS	RACTIVE	TACTIVE	BUSOFF

Bit	Marking	Place name	Function	Read and write
b7	RESET	Reset request	reset request bit 0: Do not request a partial reset 1: request partial reset Some registers can only be written when RESET=1. For details, please refer to the software reset function. When the node enters the BUS OFF state, the hardware will automatically set the RESET bit to 1. Please note that it takes 11 CAN bit times for the node to participate in communication when RESET=0.	R/W
b6	LBME	External loopback mode enable bit	External loopback mode enable bit 0: disable external loopback mode 1: Enable external loopback mode Note: It is forbidden to set this bit during communication.	R/W
b5	LBMI	Internal loopback mode enable bit	Internal loopback mode enable bit 0: disable internal loopback mode 1: Enable internal loopback mode Note: It is forbidden to set this bit during communication.	R/W
b4	TPSS	PTB single transfer mode	PTB single transfer mode 0: Disable PTB single transfer mode 1: Enable PTB single transfer mode	R/W
b3	TSSS	STB single transfer mode	STB single transfer mode 0: Disable STB single transfer mode 1: Enable STB single transfer mode	R/W
b2	RACTIVE	Receive status signal	Receive status signal 0: not receiving 1: receiving	R
b1	TACTIVE	Sending status signal	Sending status signal 0: not sending 1: Sending	R
b0	BUSOFF	Bus off state	Bus off state 0: Bus valid state 1: Bus off state NOTE: Writing a 1 clears the TECNT and RECNT registers, but only for debug purposes.	R/W

30.5.4 CAN Command Register (CAN_TCMD)

CAN Command Register

Offset address: 0xA1

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
TBSEL	LOM	STBY	TPE	TPA	TSONE	TSALL	TSA

Bit	Marking	Place name	Function	Read and write
b7	TBSEL	Send BUF selection bit	Send BUF selection bit (Transmit Buffer Select) 0: PTB 1: STB When TTEN=1&TTTBM=1, TBSEL is reset to reset value. Note: When writing TBUF register or TSNEXT bit, this bit needs to keep a constant value.	R/W
b6	LOM	Silent mode enable bit	Silent mode enable bit (Listen Only Mode) 0: disable silent mode 1: Enable silent mode Sending is prohibited when LOM=1&LBME=0. When LOM=1&LBME=1, it is forbidden to respond to the corresponding received frames and error frames, but it can send data. Note: It is forbidden to set this bit during communication.	R/W
b5	STBY	STBY set bit	STBY set bit 0: Clear STBY 1: Set STBY (TPE, TSONE, TSALL must be 0 before setting)	R/W
b4	TPE	PTB transmit enable bit	PTB send enable bit (Transmit Primary Enable) 0: Disable PTB sending 1: Enable PTB transmission After this bit is enabled, the Mailbox in PTB will be sent at the next available sending position. STB transmissions that have already started will continue, but the next pending STB transmission will be delayed until the PTB transmission is complete. After this bit is written to 1, it will remain 1 until the PTB transmission is completed or the transmission is canceled by TPA. Software cannot clear this bit by writing a 0. The TPE is reset to the reset value by hardware in the following cases: – RESET=1 – BUSOFF=1 – LOM=1&LBME=0 – TTEN=1&TTTBM=1	R/W
b3	TPA	PTB send cancel bit	PTB send cancel bit (Transmit Primary Abort) 0: do not cancel 1: Cancel the PTB transmission that has been requested by setting TPE to 1 but has not yet started This bit is written 1 by software but cleared by hardware. The TPE bit can be cleared by writing 1, so write 1 at the same time as TPE. The TPE is reset to the reset value by hardware in the following cases: – RESET=1 – BUSOFF=1 – TTEN=1&TTTBM=1	R/W
b2	TSONE	Send a frame of STB data set bit	Send a frame of STB data set bit (Transmit Secondary ONE frame) 0: do not send 1: Send a frame of STB data In FIFO mode, the earliest written data is sent, and in priority mode, the highest priority data is sent After this bit is written to 1, it will remain 1 until the STB transmission is completed or the transmission is canceled by TSA. Software cannot clear this bit by writing a 0. TSONE is reset to the reset value by hardware in the following cases: – RESET=1 – BUFOFF=1 – LOM=1&LBME=0 – TTEN=1&TTTBM=1	R/W
b1	TSALL	Send all STB data set bits	Send all STB data set bits (Transmit Secondary ALL frame) 0: do not send 1: Send all data in STB After this bit is written to 1, it will remain 1 until the STB transmission is completed or the transmission is canceled by TSA. Software cannot clear this bit by writing a 0.	R/W

			TSALL is reset to the reset value by hardware in the following cases: – RESET=1 – BUSOFF=1 – LOM=1&LBME=0 – TTEN=1&TTTBM=1	
b0	TSA	STB sends cancel bit	STB send abort bit (Transmit Secondary Abort) 0: do not cancel 1: Cancel the STB transmission that has been requested by TSONE or TSALL but has not yet started This bit is written 1 by software but cleared by hardware. Write 1 to clear TSONE or TSALL bit. TSA is reset to the reset value by hardware in the following cases: – RESET=1 – BUSOFF=1	R/W

30.5.5 CAN Transmit Control Register (CAN_TCTRL)

CAN Transmit Control Register

Offset address: 0xA2

Reset value: 0x90

b7	b6	b5	b4	b3	b2	b1	b0
-	TSNEXT	TSMODE	TTTBM	-	-	TSSTAT[1:0]	

Bit	Marking	Place name	Function	Read and write
b7	Reserved	-	The reset value must be maintained.	R
b6	TSNEXT	Next STB SLOT	Next STB (Transmit buffer Secondary NEXT) 0: no action 1: The current STB SLOT is filled, pointing to the next SLOT After the application program finishes writing the data in TBUF, it indicates that the current STB SLOT has been filled by setting the TSNEXT bit, so that the hardware points TBUF to the next STB SLOT. The data in the STB SLOT identified by the TSNEXT bit can be sent through the TSONE or TSALL bit. This bit is written to 1 by the application program and cleared to 0 by hardware. After all STB SLOTs are filled, TSNEXT remains 1 until a STB SLOT is released. Note: This bit is fixed to 0 in TTCAN mode.	R/W
b5	TSMODE	STB send mode	STB send mode (Transmit buffer Secondary operation MODE) 0: FIFO mode 1: priority mode FIFO mode is sent according to the order in which data frames are written. The priority mode is automatically judged according to the ID. The smaller the ID, the higher the priority. Regardless of the mode, PTB has the highest priority. Note: The TSMODE bit can only be set when the STB is empty.	R/W
b4	TTTBM	TTCAN BUF mode	TTCAN BUF mode (TTCAN Transmit Buffer Mode) When TTEN=0, TTTBM is ignored. 0: TSMODE decision, PTB and STB 1: Set by TBPTR and TTPTR In TTCAN mode, this bit can be set to 0 when only receiving the time stamp function is required, and TSMODE is used to determine whether to use PTB or STB. Note: The TSMODE bit can only be set when the STB is empty.	R/W
b3~b2	Reserved	-	The reset value must be maintained.	R
b1~b0	TSSTAT	STB state	STB status (Transmission Secondary STATus bits) TTEN=0 00: STB empty 01: STB is less than or equal to half full 10: STB is more than half full 11: STB full TTEN=0&TTTBM=1 00: PTB and STB empty 01: PTB and STB are not full 10: reserved 11: PTB and STB full	R

30.5.6 CAN Receive Control Register (CAN_RCTRL)

CAN Receive Control Register

Offset address: 0xA3

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
SACK	ROM	ROV	RREL	RBALL	-	RSSTAT[1:0]	

Bit	Marking	Place name	Function	Read and write
b7	SACK	Self-response	Self-acknowledgement (Self-ACKnowledge) 0: no self-response 1: When LBME=1, enable the self-acknowledgment function	R/W
b6	ROM	Receive BUF overflow mode setting bit	Receive BUF overflow mode setting bit (Receive buffer Overflow Mode) 0: The earliest received data is overwritten 1: Newly received data is not stored	R/W
b5	ROV	Receive BUF overflow flag bit	Receive BUF overflow flag (Receive buffer OVerflow) 0: no overflow 1: overflow, at least one data is lost Cleared by writing RREL to 1.	R
b4	RREL	Release receive BUF	Release receive BUF (Receive buffer RELease) 0: do not release 1: Indicates that the receiving BUF has been read, and the RBUF register points to the next RB SLOT.	R/W
b3	RBALL	Receive BUF data to store all data frames	Receive BUF data stores all data frames (Receive Buffer stores ALL data frames) 0: Normal mode 1: Store all data including error data.	R/W
b2	Reserved	-	The reset value must be maintained.	R
b1~b0	RSTAT	Receive BUF status	Receive BUF status (Receive buffer STATus) 00: RBUF empty 01: RBUF is not empty but less than AFWL programming value 10: RBUF is greater than or equal to the programmed value of AFWL but not full 11: full (keep this value on overflow)	R

30.5.7 CAN Receive and Transmit Interrupt Enable Register (CAN_RTIE)

CAN Receive and Transmit Interrupt Enable Register

Offset address: 0xA4

Reset value: 0xFE

b7	b6	b5	b4	b3	b2	b1	b0
RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	TSFF

Bit	Marking	Place name	Function	Read and write
b7	RIE	Receive interrupt enable	Receive Interrupt Enable (Receive Interrupt Enable) 0: Prohibited 1: Enable	R/W
b6	ROIE	Receive overflow interrupt enable	Receive overflow interrupt enable (Receive Overrun Interrupt Enable) 0: Prohibited 1: Enable	R/W
b5	RFIE	Receive BUF full interrupt enable	Receive BUF full interrupt enable (RB Full Interrupt Enable) 0: Prohibited 1: Enable	R/W
b4	RAFIE	Receive BUF will be full interrupt enable	Receive BUF will be full interrupt enable (RB Almost Full Interrupt Enable) 0: Prohibited 1: Enable	R/W
b3	TPIE	PTB transmit interrupt enable	PTB transmit interrupt enable (Transmission Primary Interrupt Enable) 0: Prohibited 1: Enable	R/W
b2	TSIE	STB transmit interrupt enable	STB transmit interrupt enable (Transmission Secondary Interrupt Enable) 0: Prohibited 1: Enable	R/W
b1	EIE	Error interrupt enable	Error Interrupt Enable (Error Interrupt Enable) 0: Prohibited 1: Enable	R/W
b0	TSFF	Send BUF full flag	TTEN=0 or TTTBM=0: STB full flag (Transmit Secondary buffer Full Flag) 0: STB SLOT is not fully filled 1: STB SLOT is fully filled TTEN=1 and TTTBM=1: TB full flag (Transmit buffer Full Flag) 0: The sending BUF selected by TBPTR is not fully filled 1: The sending BUF selected by TBPTR is fully filled	

30.5.8 CAN Receive and Transmit Interrupt Status Register (CAN_RTIF)

CAN Receive and Transmit Interrupt Status Register

Offset address: 0xA5

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF

Bit	Marking	Place name	Function	Read and write
b7	RIF	Receive interrupt flag	Receive Interrupt Flag (Receive Interrupt Flag) 0: No data frame received 1: A valid data frame or remote frame is received Write 1 to clear 0 by application program.	R/W
b6	ROIF	Receive overflow interrupt flag	Receive overflow interrupt flag (Receive Overrun Interrupt Flag) 0: Receive overflow does not occur 1: Receive overflow occurs Both ROIF and RFIF are set to 1 on overflow. Write 1 to clear 0 by application program.	R/W
b5	RFIF	Receive BUF full interrupt flag	Receive BUF full interrupt flag (RB Full Interrupt Flag) 0: RB FIFO is not full 1: RB FIFO is full Write 1 to clear 0 by application program.	R/W
b4	RAFIF	Receive BUF will be full interrupt flag	Receive BUF will be full interrupt flag (RB Almost Full Interrupt Flag) 0: The number of filled RB SLOTs is less than the AFWL setting value 1: The number of filled RB SLOTs is greater than or equal to the AFWL setting value Write 1 to clear 0 by application program.	R/W
b3	TPIF	PTB send interrupt flag	PTB transmit interrupt flag (Transmission Primary Interrupt Flag) 0: No PTB send completed 1: The requested PTB send completed successfully Write 1 to clear 0 by application program. Note: In TTCAN mode, TPIF is invalid, only the TSIF flag is applicable	R/W
b2	TSIF	STB send interrupt flag	STB transmit interrupt flag (Transmission Secondary Interrupt Flag) 0: No STB transmission completed 1: The requested STB send completed successfully Write 1 to clear 0 by application program. Note: In TTCAN mode, TPIF is invalid, only use TSIF flag	R/W
b1	EIF	Error interrupt flag	Error Interrupt Flag (Error Interrupt Flag) 0: The BUSOFF bit has not changed, or the relative relationship between the error counter value and the ERROR warning limit setting value has not changed. 1: The BUSOFF bit changes, or the relative relationship between the value of the error counter and the setting value of the ERROR warning limit changes. For example, the value of the error counter changes from less than the set value to greater than the set value, or from greater than the set value to less than the set value. Write 1 to clear 0 by application program.	R/W
b0	AIF	Cancel send interrupt flag	Cancel sending interrupt flag (Abort Interrupt Flag) 0: Send data is not canceled 1: The send message requested by TPA and TSA was successfully canceled. Write 1 to clear 0 by application program.	R/W

30.5.9 CAN Error Interrupt Enable and Flags Register (CAN_ERRINT)

CAN ERRor INTerrupt Enable and Flag Register

Offset address: 0xA6

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
EWARN	EPASS	WPIE	EPIF	ALIE	ALIF	BEIE	BEIF

Bit	Marking	Place name	Function	Read and write
b7	EWARN	Reached the set ERROR WARNING LIMIT	Reached the set ERROR WARNING LIMIT (Error WARNing limit reached) 0: RECNT or TECNT is less than the set value of EWL 1: RECNT or TECNT is greater than or equal to the EWL setting value Write 1 to clear 0 by application program.	R/W
b6	EPASS	Error passive	Error Passive mode active 0: The node is an active error node 1: Passive error node when node Write 1 to clear 0 by application program.	R
b5	EPIE	Error Passive Interrupt Enable	Error Passive Interrupt Enable (Error Passive Interrupt Enable) 0: Prohibited 1: Enable	R/W
b4	EPIF	Error passive interrupt flag	Error Passive Interrupt Flag (Error Passive Interrupt Flag) 0: No change from error active to error passive or error passive to error active 1: Change from error active to error passive or error passive to error active Write 1 to clear 0 by application program.	R/W
b3	ALIE	Arbitration failure interrupt enable	Arbitration Lost Interrupt Enable (Arbitration Lost Interrupt Enable) 0: Prohibited 1: Enable	R/W
b2	ALIF	Arbitration Lost Interrupt Flag	Arbitration Lost Interrupt Flag (Arbitration Lost Interrupt Flag) 0: Arbitration is successful 1: Arbitration failure Write 1 to clear 0 by application program.	R/W
b1	BEIE	Bus Error Interrupt Enable	Bus Error Interrupt Enable (Bus Error Interrupt Enable) 0: Prohibited 1: Enable	R/W
b0	BEIF	Bus error interrupt flag	Bus Error Interrupt Flag (Bus Error Interrupt Flag) 0: No bus error 1: Bus error Write 1 to clear 0 by application program.	R/W

30.5.10 CAN Bit Timing Register (CAN_BT)

CAN Bit Timing Register

Offset address: 0xA8

Reset value: 0x0102 0203

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PRESC[7:0]								-	SJW[6:0]						
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	SEG_2[6:0]							SEG_1[7:0]							

Bit	Marking	Place name	Function	Read and write
b31~b24	PRESC	Prescaler setting	Prescaler setting (Prescaler) When setting the value of this register, divide the module BCLK setting bit (PRESC+1) as the clock of TQ.	R/W
b23	Reserved	-	The reset value must be maintained.	R
b22~b16	SJW	Resynchronization compensation width time setting	Resynchronization compensation width time setting (Bit Timing Segment 2) Resynchronization compensation width time=(SJW+1)*TQ	R/W
b15	Reserved	-	The reset value must be maintained.	R
b14~b8	SEG_2	Bit segment 2 time setting	Bit segment 2 time unit setting (Bit Timing Segment 2) Segment 2 time=(SEG_2+1)*TQ	R/W
b7~b0	SEG_1	Bit segment 1 time setting	Bit segment 1 time unit setting (Bit Timing Segment 1) Segment 1 time=(SEG_1+2)*TQ	R/W

30.5.11 CAN Error and Arbitration Lost Capture Register (CAN_EALCAP)

CAN Error and Arbitration Lost Capture Register

Offset address: 0xB0

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
KOER[2:0]			ALC[4:0]				

Bit	Marking	Place name	Function	Read and write
b7~b5	KOER	Error category	Error category (Kind Of Error) 000: no error 001: bit error 010: wrong form 011: Padding error 100: Response error 101: CRC error 110: Other errors 111: reserved The KOER bit is updated when there is an error, and the KOER bit remains unchanged when sending and receiving normally.	R
b4~b0	ALC	Lost Arbitration Position Capture	Arbitration Lost Capture When the arbitration fails, the ALC records the position when the arbitration fails in one frame of data.	R/W

30.5.12 CAN Warning Limit Register (CAN_LIMIT)

CAN Warning Limits Register

Offset address: 0xA7

Reset value: 0x1B

b7	b6	b5	b4	b3	b2	b1	b0
AFWL[3:0]				EWL[3:0]			

Bit	Marking	Place name	Function	Read and write
b7~b4	AFWL	Receive BUF will be full Warning Limit	Receive buffer almost full Warning Limit (receive buffer Almost Full Warning Limit) The setting range is 1~10. AFWL=0 is meaningless, treat it as AFWL=1.	R/W
b3~b0	EWL	Error Waring Limit programming value	Error Waring Limit programming value (Programmable Error Warning Limit) Error Waring Limit=（EWL+1）*8。 The value set in this register affects the EIF flag.	R/W

30.5.13 CAN Receive Error Counter Register (CAN_RECNT)

CAN Receive Error CouNT Register

Offset address: 0xB2

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
RECNT[7:0]							

Bit	Marking	Place name	Function	Read and write
b7~b0	RECNT	Receive error counter	Receive Error Counter (Receive Error CouNT) The receive error counter increases or decreases according to the error count specified in the CAN protocol. There is no overflow in this counter, and 255 is the maximum value.	R/W

30.5.14 CAN Transmit Error Counter Register (CAN_TECNT)

CAN Transmit Error CouNT Register

Offset address: 0xB3

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
TECNT[7:0]							

Bit	Marking	Place name	Function	Read and write
b7~b0	TECNT	Send error counter	Transmit Error Counter (Transmit Error CouNT) The sending error counter increases or decreases according to the error count specified in the CAN protocol. There is no overflow in this counter, and 255 is the maximum value.	R/W

30.5.15 CAN Filter Group Control Register (CAN_ACFCTRL)

CAN Acceptance Filter Control Register

Offset address: 0xB4

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-		SELMASK	-	ACFADR			

Bit	Marking	Place name	Function	Read and write
b7~b6	Reserved	-	The reset value must be maintained.	R
b5	SELMASK	Select the mask register for the filter	Select the mask register of the filter (SELect acceptance MASK) 0: ACF points to the filter ID register 1: ACF points to the filter MASK register Select a specific filter register set by ACFADR	R/W
b4	Reserved	-	The reset value must be maintained.	R
b3~b0	ACFADR	Filter address	Acceptance filter address（acceptance filter address） ACFADR points to a specific filter, and uses SELMASK to distinguish between ID and MASK. 0000: point to ACF_1 0001: point to ACF_2 0010: point to ACF_3 0011: point to ACF_4 0100: point to ACF_5 0101: point to ACF_6 0110: point to ACF_7 0111， 1000， 1001， 1010， 1011， 1100， 1101， 1110， 1111: point to ACF_8	

30.5.16 CAN Filter Group Enable Register (CAN_ACFEN)

CAN Acceptance Filter Enable Register

Offset address: 0xB6

Reset value: 0x01

b7	b6	b5	b4	b3	b2	b1	b0
AE_8	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1

Bit	Marking	Place name	Function	Read and write
b7	AE_8	ACF_8 enable	ACF_8 enable (Acceptance Filter 8 Enable) 0: Prohibited 1: Enable	R/W
b6	AE_7	ACF_7 enable	ACF_7 enable (Acceptance Filter 7 Enable) 0: Prohibited 1: Enable	R/W
b5	AE_6	ACF_6 enable	ACF_6 enable (Acceptance Filter 6 Enable) 0: Prohibited 1: Enable	R/W
b4	AE_5	ACF_5 enable	ACF_5 enable (Acceptance Filter 5 Enable) 0: Prohibited 1: Enable	R/W
b3	AE_4	ACF_4 enable	ACF_4 enable (Acceptance Filter 4 Enable) 0: Prohibited 1: Enable	R/W
b2	AE_3	ACF_3 enable	ACF_3 enable (Acceptance Filter 3 Enable) 0: Prohibited 1: Enable	R/W
b1	AE_2	ACF_2 enable	ACF_2 enable (Acceptance Filter 2 Enable) 0: Prohibited 1: Enable	R/W
b0	AE_1	ACF_1 enable	ACF_1 enable (Acceptance Filter 1 Enable) 0: Prohibited 1: Enable	R/W

30.5.17 CAN filter group code and mask registers (CAN_ACF)

CAN Acceptance Filter code and mask Register

Offset address: 0xB8

Reset value: 0xXXXX XXXX

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	AIDEE	AIDE	ACODE[28:16] or AMASK_28:16]												
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ACODE[15:0] or AMASK[15:0]															

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	The readout value is indeterminate.	R
b30	AIDEE	IDE bit compare enable	IDE bit comparison enable (Acceptance mask IDE bit check enable) Valid only when SELMASK=1 0: Filter accepts standard format and extended format frames 1: The filter accepts standard format or extended format frames defined by the AIDE bits	R/W
b29	AIDE	IDE bit MASK	IDE bit MASK 0: filter only accepts standard formats 1: Filter only accepts extended formats	R/W
b28~b0	ACODE/ AMASK	Filter CODE/ Filter MASK	Filter CODE (acceptance filter code) Point to the specific filter via ACFADR. When SELMASK=0, it means the CODE of the filter. Use bit 10~bit 0 for the standard format, and bit 29~bit 0 for the extended format. Filter CODE (acceptance filter mask) Point to the specific filter via ACFADR. When SELMASK=1, it indicates the mask of the filter. Use bit 10~bit 0 for the standard format, and bit 29~bit 0 for the extended format.	R/W

30.5.18 TTCAN TB slot pointer register (CAN_TBSLOT)

TTCAN TB Slot Pointer Register

Offset address: 0xBE

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
TBE	TBF	-	-	-	TBPTR[2:0]		

Bit	Marking	Place name	Function	Read and write
b7	TBE	Set TB to empty	Set TB slot to empty (set TB slot to “empty”) 0: No operation 1: The SLOT selected by TBPTR is marked as empty When SLOT is marked empty and TSFF=0, TBE is automatically reset to 0. If this bit is set to 1, there is data being sent in the selected SLOT, then TBE=1, then TBE is reset to 0 after the sending is completed, the sending is wrong, or the sending is canceled. TBE has higher priority than TBF.	R/W
b6	TBE	Set TB to filled	Set TB slot to “ Filled” (set TB slot to “Filled”) 0: No operation 1: SLOT selected by TBPTR is marked as filled TBE is automatically reset to 0 when SLOT is marked as filled and TSFF=1.	R/W
b5~b3	Reserved	-	The reset value must be maintained.	R
b2~b0	TBPTR	TB SLOT pointer	TB SLOT pointer (Pointer to a TB message slot) 000: point to PTB 001: point to STB SLOT1 010: point to STB SLOT2 011: point to STB SLOT3 100: point to STB SLOT4 Others: Prohibiting The pointed TB SLOT can be read and written through TBUF, and can be marked whether it has been filled through TBE and TBF. In TTCAN mode, the TBSEL and TSNEXT registers are invalid. Note: This bit can only be written when TSFF=0.	R/W

30.5.19 TTCAN Time Triggered Configuration Register (CAN_TTCFG)

TTCAN TB Slot Pointer Register

Offset address: 0xBF

Reset value: 0x90

b7	b6	b5	b4	b3	b2	b1	b0
WTIE	WTIF	TEIF	TTIE	TTIF	T_PRESC[1:0]		TTEN

Bit	Marking	Place name	Function	Read and write
b7	WTIE	Trigger gate interrupt enable	Trigger watchdog interrupt enable (Watch Trigger Interrupt Enable) 0: Prohibited 1: Enable	R/W
b6	WTIF	trigger gatekeeper interrupt flag	Trigger the gate interrupt flag (Watch Trigger Interrupt Flag) When CYCLE COUNT value=TT_WTRIG setting value and WTIE=1, WTIF is set. Write 1 to clear 0 by application program.	R/W
b5	TEIF	Trigger error interrupt flag	Error Interrupt Flag (Trigger Error Interrupt Flag) When the set value of TT_TTIG is less than the actual CYCLE_TIME, TEIF is set.	R
b4	TTIE	Time-triggered interrupt enable	Time Trigger Interrupt Enable (Time Trigger Interrupt Enable) 0: Prohibited 1: Enable	R/W
b3	TTIF	Time triggered interrupt flag	Time Trigger Interrupt Flag (Time Trigger Interrupt Flag) When CYCLE COUNT value = TT_TRIG set value and TTIE = 1, TTIF is set. If TT_TRIG is not updated, TTIF is only set once, and the next basic CYCLE is not set. Write 1 to clear 0 by application program.	R/W
b2~b1	T_PRESC	TTCAN counter prescaler	TTCAN Counter Prescaler (TTCAN Timer PRESCaler) 00: Divide by 1 the bit time set by the BT register 01: Divide by 2 the bit time set by the BT register 10: Divide by 4 the bit time set by the BT register 11: Eighth frequency division of the bit time set by the BT register Note: T_PRESC can be written when TTEN=0 or simultaneously when writing TTEN=1.	R/W
b0	TTEN	TTCAN enable	TTCAN enable (Time Trigger Enable) 0: Prohibited 1: Enable TTCAN, the counter starts counting.	R/W

30.5.20 TTCAN Reference Message Register (CAN_REF_MSG)

TTCAN Reference Message Register

Offset address: 0xC0

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
REF_IDE	-		REF_ID[28:16]												
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
REF_ID[15:0]															

Bit	Marking	Place name	Function	Read and write
b31	REF_IDE	IDE bits for reference messages	IDE bit of the reference message (REFerence message IDE bit) 0: standard format 1: Extended format	R/W
b30~b29	Reserved	-	The readout value is indeterminate.	R
b28~b0	REF_ID	ID bit of the reference message	The ID bit of the reference message (REFerence message IDentifier) REF_ID=0: REF_ID[28:0] is valid REF_ID=1: REF_ID[28:0] is valid REF_ID is used to detect reference messages and is applicable for sending and receiving. After the reference message is detected, the Sync_Mark of the current frame becomes Ref_Mark. REF_ID[2:0] is fixed to 0, and its value is not checked, so that up to 8 potential time masters can be supported. After the highest byte of REF_MSG is written, it is necessary to wait for 6 CAN clock cycles to complete the transfer of REF_MSG to the CAN clock domain.	R/W

30.5.21 TTCAN Trigger Configuration Register (CAN_TRG_CFG)

TTCAN Reference Message Register

Offset address: 0xC4

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TEW[3:0]				-	TTYPE[2:0]			-					TTPTR[2:0]		

Bit	Marking	Place name	Function	Read and write
b15~b12	TEW	Send enable window	Transmit Enable Window (Transmit Enable Window) For the Single Shot Transmit Trigger mode (Single Shot Transmit Trigger) of TTCAN, a window of TEW+1 cycle time can be set, and transmission is only allowed within this window.	R/W
b11	Reserved	-	The reset value must be maintained.	R
b10~b8	TTYPE	Trigger type	Trigger type (Trigger Type) 000: Immediate Trigger (Immediate Trigger for immediate transmission) 001: Time Trigger (Time Trigger for receive triggers) 010: Single Shot Transmit Trigger (Single Shot Transmit Trigger for exclusive time windows) 011: Transmit Start Trigger (Transmit Start Trigger for merged arbitrating time windows) 100: Transmit Stop Trigger (Transmit Stop Trigger for merged arbitrating time windows) Other: reserved The trigger time is set by the TT_TRIG register, and the TB Slot is selected by TTPTR.	R/W
b7~b3	Reserved	-	The reset value must be maintained.	R
b2~b0	TTPTR	Send trigger TB slot pointer	Transmit Trigger TB slot pointer (Transmit Trigger TB slot Pointer) 000: point to PTB 001: point to STB SLOT1 010: point to STB SLOT2 011: point to STB SLOT3 100: point to STB SLOT4 Others: Prohibiting If the pointed TB SLOT is marked empty, TEIF is set when the trigger time is reached.	R/W

30.5.22 TTCAN Trigger Time Register (CAN_TRG_TRIG)

TTCAN Reference Message Register

Offset address: 0xC6

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TT_TRIG[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	TT_TRIG	Trigger time	Trigger time (Trigger Time) It is used to specify the cycle time of the trigger. For the sending trigger, the sending SOF time is about TT_TRIG setting value +1 When the highest byte of TT_TRIG is written, the TT_TRIG value begins to be transferred to the CAN clock domain. Therefore, if BYTE is operated, it is necessary to write the low byte first and then the high byte.	R/W

30.5.23 TTCAN Trigger Watchdog Time Register (TRG_WTRIG)

TTCAN Watch Trigger Time Register

Offset address: 0xC8

Reset value: 0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TT_WTRIG[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	TT_TRIG	Trigger time	Trigger time (Trigger Time) Used to specify the cycle time of the gate trigger. When the highest byte of TT_WTRIG is written, the TT_WTRIG value begins to be transferred to the CAN clock domain. Therefore, if BYTE is operated, it is necessary to write the low byte first and then the high byte.	R/W

30.6Precautions for use

30.6.1 CAN bus anti-jamming measures

The CAN bus is widely used in automobiles, industrial control and other industries. If the electromagnetic environment of the CAN application site is relatively harsh, there are factors such as circuit imbalance, space electromagnetic field, and grid incoming lines, which will cause a large amount of communication noise on the CAN bus due to radiation and conduction interference. , resulting in the increase of bus error frames, frequent retransmissions, and the failure of correct data to arrive in time, which seriously affects the quality of data communication. Therefore, in practical applications, efforts should be made to eliminate noise interference and ensure the stable operation of the CAN bus network.

The following are several commonly used CAN bus anti-interference measures (including but not limited to)

1) Increase electrical isolation of CAN bus interface
2) Common transceiver signal ground
3) Use shielded twisted pair cables and ground them properly
4) Improve twisted pair degree of CAN transmission line
5) Add signal protector
6) Improve network topology
7) Application layer software anti-jamming mechanism

30.6.2 CAN Controller Noise Constraints

In the CAN bus network, it should be ensured that the bit time of the communication meets the requirements of the standard protocol. If the noise interference that does not meet the bit time width is introduced, it may cause the abnormal operation of the CAN controller .

31 Analog-to-digital converter (ADC)

31.1 Module Introduction

External analog signals need to be converted into digital signals to be further processed by the MCU. 12 -bit successive approximation analog-to-digital converter (SAR ADC) module with high precision and high conversion rate is integrated inside. Has the following properties:

- 12-bit conversion accuracy;
- 1M SPS conversion speed;
- 40 input channels, including 36 external pin inputs, 1 -channel internal temperature sensor voltage, 1 -channel 1/3 AVCC voltage, and 2-channel DAC outputs;
- 4 reference sources: AVCC voltage, ExRef pin, built-in 1.5v reference voltage, built-in 2.5v reference voltage;
- ADC voltage input range: 0~Vref;
- 4 conversion modes: single conversion, sequential scan continuous conversion, queue scanning continuous conversion, continuous conversion accumulation;
- Input channel voltage threshold monitoring;
- Software can configure ADC conversion rate;
- Built-in signal follower, which can convert high-impedance signals;
- Support on-chip peripherals to automatically trigger ADC conversion, effectively reducing chip power consumption and improving real-time conversion.

31.2 ADC block diagram

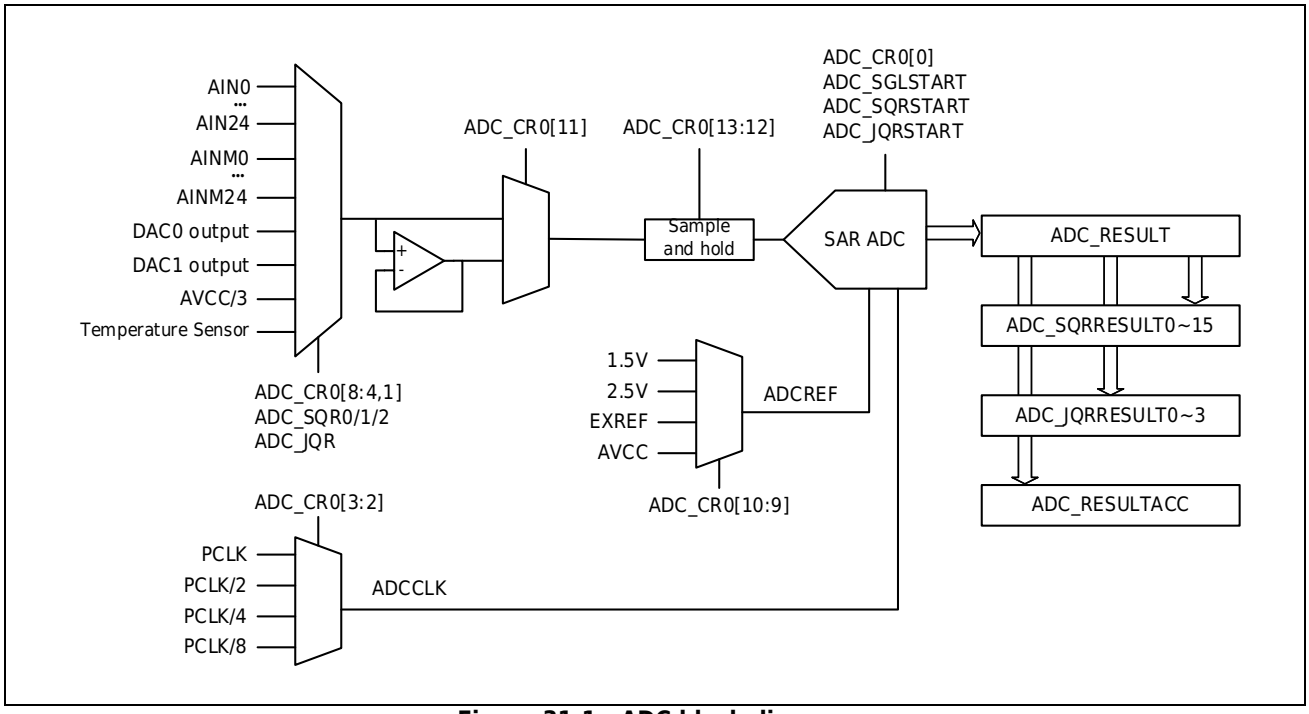


Figure 31-1 ADC block diagram

31.3Conversion Timing and Conversion Speed

The ADC conversion timing is shown in the figure below: a complete ADC conversion consists of a conversion process and a successive comparison process. Among them, the conversion process requires 4~12 ADCCLKs, which are configured by ADC_CR0.SAM; the successive comparison process requires 16 ADCCLKs. Therefore, an ADC conversion requires a total of 20~ 28 ADCCLKs.

ADC conversion speed is SPS, that is, how many ADC conversions are performed per second. The calculation method of the ADC conversion speed is: the frequency of ADCCLK / the number of ADCCLKs required for one ADC conversion.

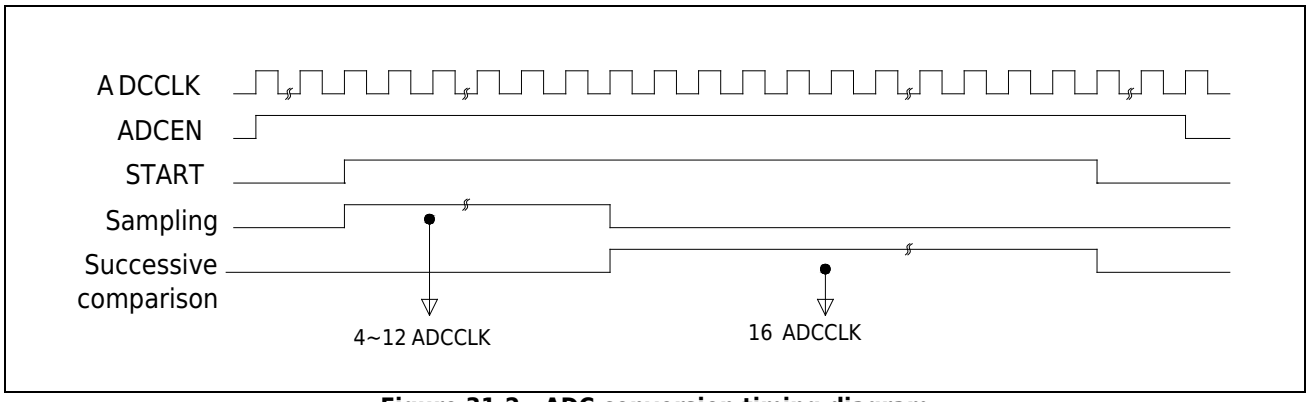


Figure 31-2 ADC conversion timing diagram

The ADC conversion speed is related to the ADC reference voltage and AVCC voltage. The maximum conversion speed is shown in the following table:

ADC reference voltage	AVCC voltage	Maximum conversion speed	Maximum ADCCLK frequency
Internal 1.5V	1.8V~5.5V	200K SPS	4MHz
Internal 2.5V	2.8V~5.5V	200K SPS	4MHz
AVCC / ExRef	1.8V~2.4V	200K SPS	4MHz
AVCC / ExRef	2.4V~2.7V	500K SPS	16MHz
AVCC / ExRef	2.7V~5.5V	1M SPS	24MHz

31.4Single conversion mode

In the single conversion mode, only one conversion is performed after the ADC starts, and all 30 ADC channels can be converted. This mode can be started by setting the ADC_SglStart.Start bit or by setting the external trigger of ADC_ExtTrigger0. Once the ADC conversion of the selected channel is completed, the ADC_IFR.SGLIF bit is automatically set to 1, and the conversion result is saved in the ADC_Result register.

Start the ADC single conversion operation process through the ADC_SglStart.Start bit:

Step1: Configure the corresponding bits of PAADS~PEADS, and configure the ADC channel to be converted as an analog port.

Step2: Set PBADS.bit1 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the ADC reference voltage does not select an external reference voltage pin, this step can be skipped.

Step3: Set BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.En to 1 to enable the ADC module.

Step5: Delay 20us, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.Mode to 0, select single conversion mode.

Step7: Configure ADC_CR0.Ref to select the reference voltage of ADC.

Step8: Set ADC_CR0.InRefEn to 1 to enable the ADC internal reference voltage.

Note: If the ADC reference voltage does not select the internal reference voltage, this step can be skipped.

Step9: Configure ADC_CR0.SAM and ADC_CR0.CkDiv to set the conversion speed of ADC.

Step10: Configure ADC_CR0.SGLMux and select the channel to be converted.

Step11: Set ADC_ICR.SGLIC to 0 and clear the ADC_IFR.SGLIF flag.

Step12: Set ADC_SglStart.Start to 1 to start ADC single conversion.

Step13: Wait for ADC_IFR. SGLIF to become 1 read the ADC_Result register to get the ADC conversion result.

Step14: To convert other channels, repeat Step10~Step13.

Step15: Set ADC_CR0.En and BGR_CR.BGR_EN to 0, turn off the ADC module and BGR module.

ADC single conversion operation process through an external trigger:

Step1: Configure the corresponding bits of PAADS~PEADS, and configure the ADC channel to be converted as an analog port.

Step2: Set PBADS.bit1 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the ADC reference voltage does not select an external reference voltage pin, this step can be skipped.

Step3: Set BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.En to 1 to enable the ADC module.

Step5: Delay 20us, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.Mode to 0, select single conversion mode.

Step7: Set ADC_CR0.IE to 1 to enable ADC interrupt.

Step8: Enable the ADC interrupt in the NVIC interrupt vector table.

Step9: Configure ADC_CR0.Ref to select the reference voltage of ADC.

Step10: Set ADC_CR0.InRefEn to 1 to enable ADC internal reference voltage.

Note: If the ADC reference voltage does not select the internal reference voltage, this step can be skipped.

Step11: Configure ADC_CR0.SAM and ADC_CR0.CkDiv to set the conversion speed of ADC.

Step12: Configure ADC_CR0.SGLMux and select the channel to be converted.

Step13: Set ADC_IFR to 0x0 clear the ADC interrupt flag.

Step14: Configure ADC_ExtTrigger0 and select the external trigger condition.

Step15: When the external trigger condition triggers the ADC to complete the conversion, the ADC module will generate an interrupt. Users can read the ADC_Result register in the ADC interrupt service routine to get the ADC conversion result.

Step16: To convert other channels, repeat Step12~Step15.

Step17: Set ADC_CR0.En and BGR_CR.BGR_EN to 0 turn off the ADC module and BGR module.

31.5Scan conversion mode

The scan conversion mode is divided into two modes: sequential scan conversion and queue scan conversion. When the two modes work independently, multiple conversions can be performed on multiple channels continuously; when the two modes work at the same time, the conversion of the channel configured by jumping in line is given priority.

31.5.1 Sequential Scan Conversion Mode

The sequential scan conversion mode can perform up to 16 consecutive conversions, and the total number of conversions is configured by ADC_SQR2.CNT; all channels can be configured for conversion, and the channel to be converted is configured by ADC_SQRx.CHxMux. This mode can be started by setting the ADC_SqrStart.Start bit or by setting the external trigger of ADC_ExtTrigger0. After starting the conversion, the ADC module converts the channels configured in CHxMux~CH0Mux in turn until the total number of conversions is completed. After the ADC module completes the total number of conversions, the ADC_IFR.SQRIF bit will be automatically set to 1, and the conversion result will be saved in the ADC_SqrResultx~ADC_SqrResult0 register corresponding to the conversion channel. The figure below demonstrates the sequential scan conversion process with 8 conversions for AIN0 AIN1 AIN5. Among them, the sequential scan conversion channels 7, 4, and 1 are configured as AIN1, the conversion channels 6, 3, and 0 are configured as AIN0, and the conversion channels 5 and 2 are configured as AIN5. After ADC_SqrStart.Start is set to 1, the ADC module will convert sequential scan conversion channels 7~0 in turn.

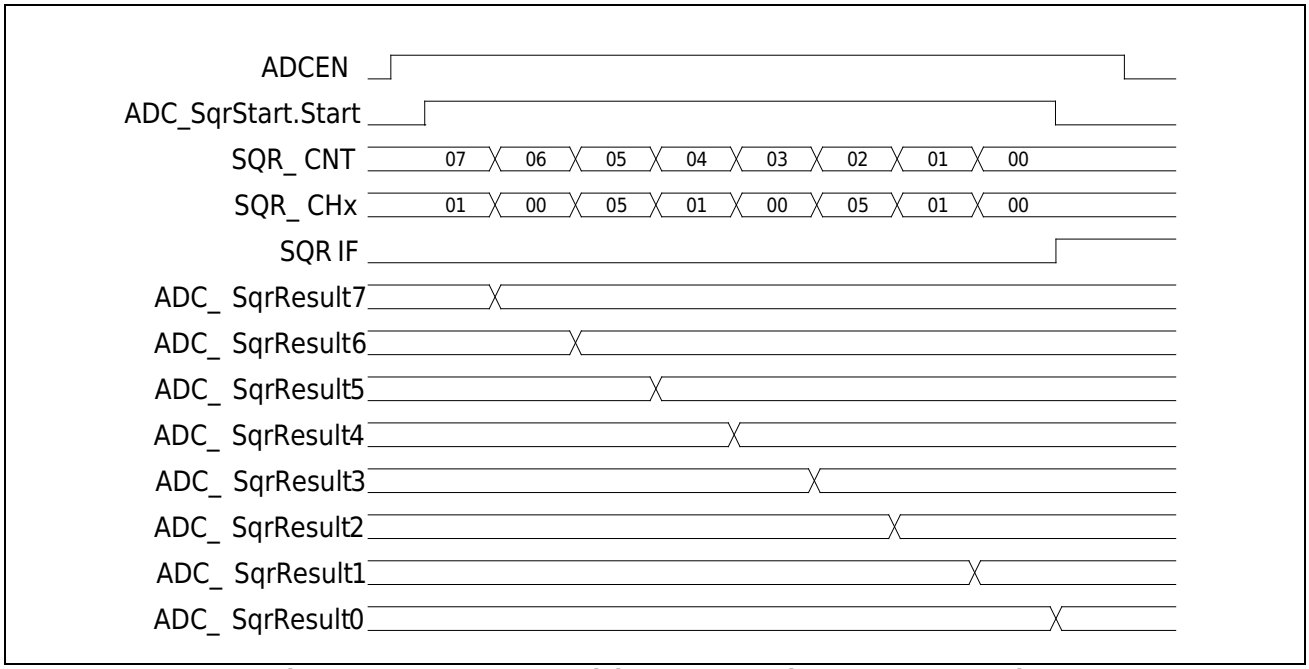


Figure 31-3 ADC sequential scan conversion process example

Start the ADC sequential scan conversion operation process through the ADC_SqrStart.Start bit:

Step1: Configure the corresponding bits of PAADS~PEADS, and configure the ADC channel to be converted as an analog port.

Step2: Set PBADS.bit1 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the ADC reference voltage does not select an external reference voltage pin, this step can be skipped.

Step3: Set BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.En to 1 to enable the ADC module.

Step5: Delay 20us, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.Mode to 1, select scan conversion mode.

Step7: Configure ADC_CR0.Ref to select the reference voltage of ADC.

Step8: Set ADC_CR0.InRefEn to 1 to enable the ADC internal reference voltage.

Note: If the ADC reference voltage does not select the internal reference voltage, this step can be skipped.

Step9: Configure ADC_CR0.SAM and ADC_CR0.CkDiv to set the conversion speed of ADC.

Step10: Configure ADC_SQRx.CHxMux, select the sequential scan conversion channel.

Step11: Configure ADC_SQR2.CNT to select the total number of conversions for sequential scan conversion.

Note: The total number of conversions needs to be consistent with the number of conversion channels configured in Step9.

Step12: Set ADC_ICR.SQRIC to 0, clear the ADC_IFR.SQRIF flag

Step13: Set ADC_SqrStart.Start to 1 to start ADC sequential scan conversion.

Step14: Wait for ADC_IFR.SQRIF to become 1, read the ADC_SqrResultx ~ ADC_SqrResult0 registers to obtain the conversion result of the corresponding channel.

Step15: To convert other channels, repeat Step10~Step14.

Step16: Set ADC_CR0.En and BGR_CR.BGR_EN to 0 turn off the ADC module and BGR module.

Start the ADC sequential scan conversion operation process through an external trigger:

Step1: Configure the corresponding bits of PAADS~PEADS, and configure the ADC channel to be converted as an analog port.

Step2: Set PBADS.bit1 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the ADC reference voltage does not select an external reference voltage pin, this step can be skipped.

Step3: Set BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.En to 1 to enable the ADC module.

Step5: Delay 20us, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.Mode to 1, select scan conversion mode.

Step7: Set ADC_CR0.IE to 1 to enable ADC interrupt.

Step8: Enable the ADC interrupt in the NVIC interrupt vector table.

Step9: Configure ADC_CR0.Ref to select the reference voltage of ADC.

Step10: Set ADC_CR0.InRefEn to 1 to enable ADC internal reference voltage.

Note: If the ADC reference voltage does not select the internal reference voltage, this step can be skipped.

Step11: Configure ADC_CR0.SAM and ADC_CR0.CkDiv to set the conversion speed of ADC.

Step12: Configure ADC_SQRx.CHxMux, select the sequential scan conversion channel.

Step13: Configure ADC_SQR2.CNT to select the total number of conversions for sequential scan conversion.

Note: The total number of conversions needs to be consistent with the number of conversion channels configured in Step11.

Step14: Set ADC_IFR to 0x0, clear the ADC interrupt flag.

Step15: Configure ADC_ExtTrigger0 and select the external trigger condition.

Step16: When the external trigger condition triggers the ADC to complete the conversion, the ADC module will generate an interrupt. Users can read the ADC_SqrResultx ~ ADC_SqrResult0 registers in the ADC interrupt service routine to get the conversion result of the corresponding channel.

Step17: To convert other channels, repeat Step12~Step16.

Step18: Set ADC_CR0.En and BGR_CR.BGR_EN to 0 turn off the ADC module and BGR module.

31.5.2 In-queue scan conversion mode

The jump scan conversion mode can perform up to 4 consecutive conversions, and the total number of conversions is configured by ADC_JQR.CNT; all channels can be configured for conversion, and the channel to be converted is configured by ADC_JQR.CHxMux. This mode can be started by setting the ADC_JqrStart.Start bit or by setting the external trigger of ADC_ExtTrigger1. After starting the conversion, the ADC module converts the channels configured in CHxMux~CH0Mux in turn until the total number of conversions is completed. After the ADC module completes the total number of conversions, the ADC_IFR.JQRIF bit will be automatically set to 1, and the conversion result will be stored in the ADC_JqrResultx~ ADC_JqrResult0 registers corresponding to the conversion channel. The figure below demonstrates the process of performing 4 conversions of AIN0, AIN1, and AIN5 in the queue scan conversion process. Among them, the queue scanning conversion channels 3, 2, 1, and 0 are respectively set to AIN5, AIN0, AIN1, and AIN5. After ADC_JqrStart.Start is set to 1, the ADC module will sequentially convert the queue scanning conversion channels 3~0.

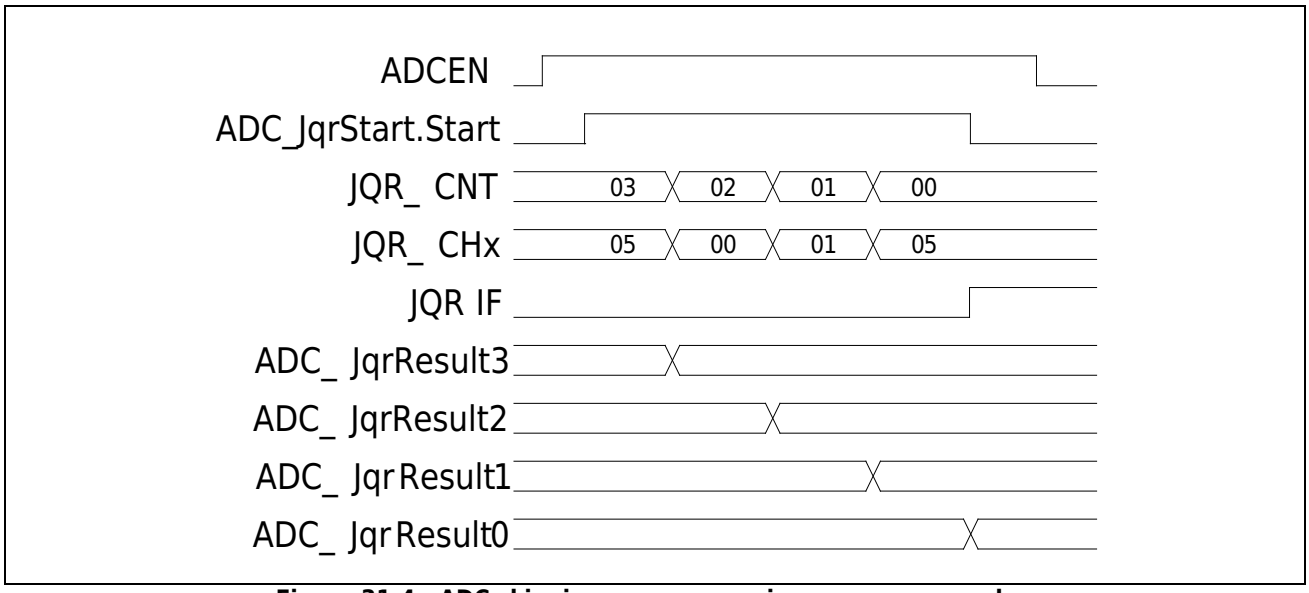


Figure 31-4 ADC skipping scan conversion process example -

Start the ADC jump-in scan conversion operation process through the ADC_JqrStart.Start bit:

Step1: Configure the corresponding bits of PAADS~PCADS, and configure the ADC channel to be converted as an analog port.

Step2: Set PBADS.bit1 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the ADC reference voltage does not select an external reference voltage pin, this step can be skipped.

Step3: Set BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.En to 1 to enable the ADC module.

Step5: Delay 20us, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.Mode to 1, select scan conversion mode.

Step7: Configure ADC_CR0.Ref to select the reference voltage of ADC.

Step8: Set ADC_CR0.InRefEn to 1 to enable the ADC internal reference voltage.

Note: If the ADC reference voltage does not select the internal reference voltage, this step can be skipped.

Step9: Configure ADC_CR0.SAM and ADC_CR0.CkDiv to set the conversion speed of ADC.

Step10: Configure ADC_JQR.CHxMux and select the queue-jumping scan conversion channel.

Step11: Configure ADC_JQR.CNT to select the total number of conversions for queue jump scan conversion.

Note: The total number of conversions needs to be consistent with the number of conversion channels configured in Step9.

Step12: Set ADC_ICR.JQRIC to 0 and clear the ADC_IFR.JQRIF flag.

Step13: Set ADC_JqrStart.Start to 1 to start ADC jump-in scan conversion.

Step14: Wait for ADC_IFR.JQRIF to become 1, and read the ADC_JqrResultx~ADC_JqrResult0 registers to obtain the conversion result of the corresponding channel.

Step15: To convert other channels, repeat Step10~Step14.

Step16: Set ADC_CR0.En and BGR_CR.BGR_EN to 0 turn off the ADC module and BGR module.

Start the ADC queue-queue scan conversion operation process through an external trigger:

Step1: Configure the corresponding bits of PAADS~PCADS, and configure the ADC channel to be converted as an analog port.

Step2: Set PBADS.bit1 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the ADC reference voltage does not select an external reference voltage pin, this step can be skipped.

Step3: Set BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.En to 1 to enable the ADC module.

Step5: Delay 20us, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.Mode to 1, select scan conversion mode.

Step7: Set ADC_CR0.IE to 1 to enable ADC interrupt.

Step8: Enable the ADC interrupt in the NVIC interrupt vector table.

Step9: Configure ADC_CR0.Ref to select the reference voltage of ADC.

Step10: Set ADC_CR0.InRefEn to 1 to enable ADC internal reference voltage.

Note: If the ADC reference voltage does not select the internal reference voltage, this step can be skipped.

Step11: Configure ADC_CR0.SAM and ADC_CR0.CkDiv to set the conversion speed of ADC.

Step12: Configure ADC_JQR.CHxMux, and select the queue-to-scan conversion channel.

Step13: Configure ADC_JQR.CNT to select the total number of conversions for queue jump scan conversion.

Note: The total number of conversions needs to be consistent with the number of conversion channels configured in Step11.

Step14: Set ADC_IFR to 0x0, clear the ADC interrupt flag.

Step15: Configure ADC_ExtTrigger1 and select the external trigger condition.

Step16: When the external trigger condition triggers the ADC to complete the conversion, the ADC module will generate an interrupt. Users can read the ADC_JqrResultx~ADC_JqrResult0 registers in the ADC interrupt service routine to get the conversion result of the corresponding channel.

Step17: To convert other channels, repeat Step12~Step16.

Step18: Set ADC_CR0.En and BGR_CR.BGR_EN to 0 turn off the ADC module and BGR module.

The execution priority of the jump scan conversion is higher than that of the sequential scan conversion. When the sequential scan conversion is in progress, if the queue scan conversion is started, the sequential scan will be suspended after the conversion of the current channel is completed, and then continue to execute the rest after the queue scan conversion is completed. Channel conversion below. The figure below demonstrates the process of starting the queue-hopping scan conversion for AIN2 and AIN6 when performing sequential scan conversion on AIN0, AIN1, AIN5, and AIN8. Among them, sequential scan conversion channels 3, 2, 1, and 0 are set to AIN1, AIN0, AIN5, and AIN8 respectively, and queue-cutting scan conversion channels 1 and 0 are respectively set to AIN6 and AIN2. In the process of sequential scanning for AIN0 conversion, the queue scanning is started. The sequential scanning will complete the conversion of the current AIN0 and then pause. After the queue scanning completes the conversion of AIN6 and AIN2, the sequential scanning will perform the conversion of AIN5 and AIN8.

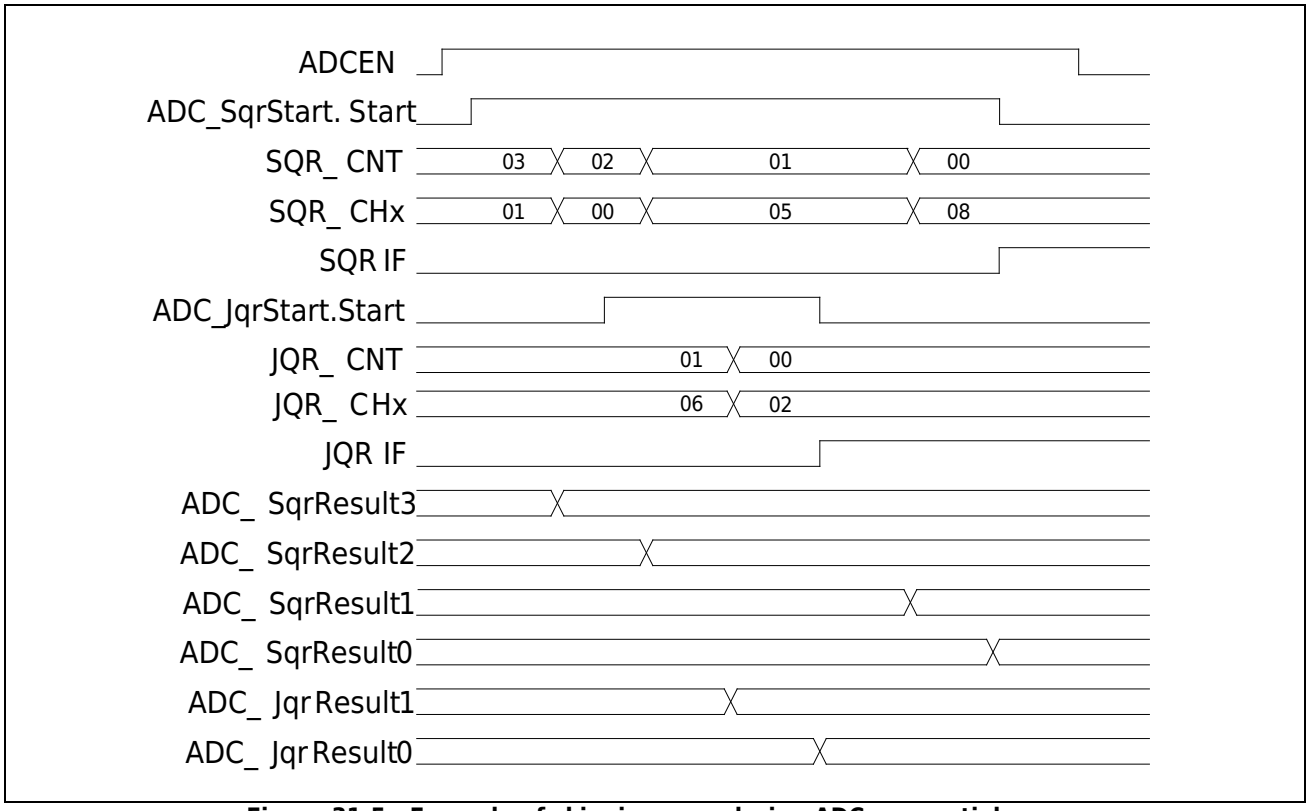


Figure 31-5 Example of skipping scan during ADC sequential scan

31.5.3 Scan conversion triggers DMA read

After the conversion of sequential scan and skip scan is completed, the DMA can be automatically triggered to read the conversion result. In the sequential scan mode, enable this function by configuring ADC_CR1.DmaSqr to 1; in the jump scan mode, enable this function by configuring ADC_CR1.DmaJqr as 1.

31.6Continuous Conversion Accumulation Mode

In the continuous conversion and accumulation mode, starting the ADC once can perform multiple conversions on multiple channels and accumulate the results of each conversion; all channels can be configured to convert. The total number of ADC conversions is configured by ADC_SQR2.CNT; the channel to be converted is configured by ADC_SQRx.CHxMux. This mode can be started by setting the ADC_SqrStart.Start bit or by setting the external trigger of ADC_ExtTrigger0. After starting the continuous conversion, the ADC module converts the channels configured in CHxMux~CH0Mux in turn until the total number of conversions is completed. After the ADC module completes the total number of conversions, the ADC_IFR.SQRIF bit is automatically set to 1, and the accumulated value of the conversion result is stored in the ADC_ResultAcc register.

The figure below demonstrates the process of accumulating 10 consecutive conversions of AIN0, AIN1, and AIN5. Sequential scan conversion channels 9, 6, 3, 0 are configured as AIN0, conversion channels 8, 5, 2 are configured as AIN1, and conversion channels 7, 4, 1 are configured as AIN5. After ADC_SqrStart.Start is set to 1, the ADC module will convert sequentially according to the

sequential scan conversion channel configuration until the count value of SQR_CNT becomes 0. The ADC_ResultAcc register is automatically accumulated each time a conversion is complete. The conversion results of AIN0, AIN1, and AIN5 given in the figure are 0x010, 0x020, and 0x040 in turn.

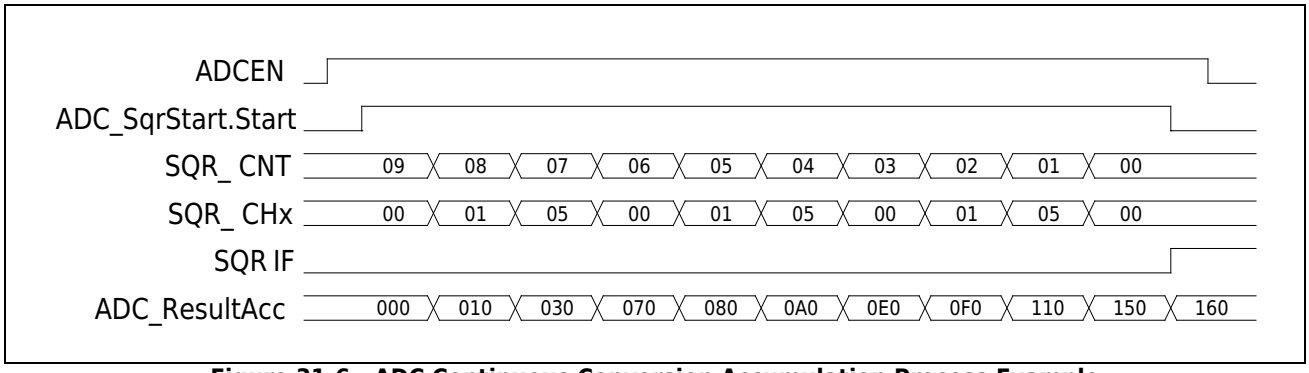


Figure 31-6 ADC Continuous Conversion Accumulation Process Example

Start the ADC continuous conversion and accumulation operation process through the ADC_SqrStart.Start bit:

Step1: Configure the corresponding bits of PAADS~PEADS, and configure the ADC channel to be converted as an analog port.

Step2: Set PBADS.bit1 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the ADC reference voltage does not select an external reference voltage pin, this step can be skipped.

Step3: Set BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.En to 1 to enable the ADC module.

Step5: Delay 20us, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.Mode to 1, select scan conversion mode.

Step7: Set ADC_CR1.RAccEn to 1 to enable the automatic accumulation function of ADC conversion.

Step8: Configure ADC_CR0.Ref to select the reference voltage of ADC.

Step9: Set ADC_CR0.InRefEn to 1 to enable ADC internal reference voltage.

Note: If the ADC reference voltage does not select the internal reference voltage, this step can be skipped.

Step10: Configure ADC_CR0.SAM and ADC_CR0.CkDiv to set the conversion speed of ADC.

Step11: Configure ADC_SQRx.CHxMux, select the sequential scan conversion channel.

Step12: Configure ADC_SQR2.CNT to select the total number of conversions for sequential scan conversion.

Note: The total number of conversions needs to be consistent with the number of conversion channels configured in Step10.

Step13: Set ADC_ICR.SQRIC to 0 and clear the ADC_IFR.SQRIF flag.

Step14: Set ADC_CR1.RAccClr to 0, and clear the ADC_ResultAcc register.

Step15: Set ADC_SqrStart.Start to 1 to start ADC sequential scan conversion.

Step16: Wait for ADC_IFR.SQRIF to become 1, and read the ADC_ResultAcc register to obtain the accumulated value of the conversion result.

Step17: To convert other channels, repeat Step11~Step16.

Step18: Set ADC_CR0.En and BGR_CR.BGR_EN to 0 turn off the ADC module and BGR module.

Start ADC continuous conversion and accumulation operation process through external trigger:

Step1: Configure the corresponding bits of PAADS~PEADS, and configure the ADC channel to be converted as an analog port.

Step2: Set PBADS.bit1 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the ADC reference voltage does not select an external reference voltage pin, this step can be skipped.

Step3: Set BGR_CR.BGR_EN to 1 to enable the BGR module.

Step4: Set ADC_CR0.En to 1 to enable the ADC module.

Step5: Delay 20us, wait for the ADC and BGR module to start up.

Step6: Set ADC_CR1.Mode to 1, select scan conversion mode.

Step7: Set ADC_CR0.IE to 1 to enable ADC interrupt.

Step8: Enable the ADC interrupt in the NVIC interrupt vector table.

Step9: Set ADC_CR1.RAccEn to 1 to enable the automatic accumulation function of ADC conversion.

Step10: Configure ADC_CR0.Ref to select the reference voltage of ADC.

Step11: Set ADC_CR0.InRefEn to 1 to enable ADC internal reference voltage.

Note: If the ADC reference voltage does not select the internal reference voltage, this step can be skipped.

Step12: Configure ADC_CR0.SAM and ADC_CR0.CkDiv to set the conversion speed of ADC.

Step13: Configure ADC_SQRx.CHxMux, select the sequential scan conversion channel.

Step14: Configure ADC_SQR2.CNT to select the total number of conversions for sequential scan conversion.

Note: The total number of conversions needs to be consistent with the number of conversion channels configured in Step12.

Step15: Set ADC_IFR to 0x0, clear the ADC interrupt flag.

Step16: Set ADC_CR1.RAccClr to 0, and clear the ADC_ResultAcc register.

Step17: Configure ADC_ExtTrigger0 and select the external trigger condition.

Step18: When the external trigger condition triggers the ADC to complete the conversion, the ADC module will generate an interrupt. Users can read the ADC_ResultAcc register in the ADC interrupt service routine to get the conversion result accumulation value.

Step19: To convert other channels, repeat Step13~Step18.

Step20: Set ADC_CR0.En and BGR_CR.BGR_EN to 0 turn off the ADC module and BGR module.

31.7ADC conversion external trigger source

ADC conversions can be initiated either by software configuration or by an external trigger.

Configure the ADC_ExtTrigger0 register to set the external trigger source for ADC single conversion or sequential scan conversion.

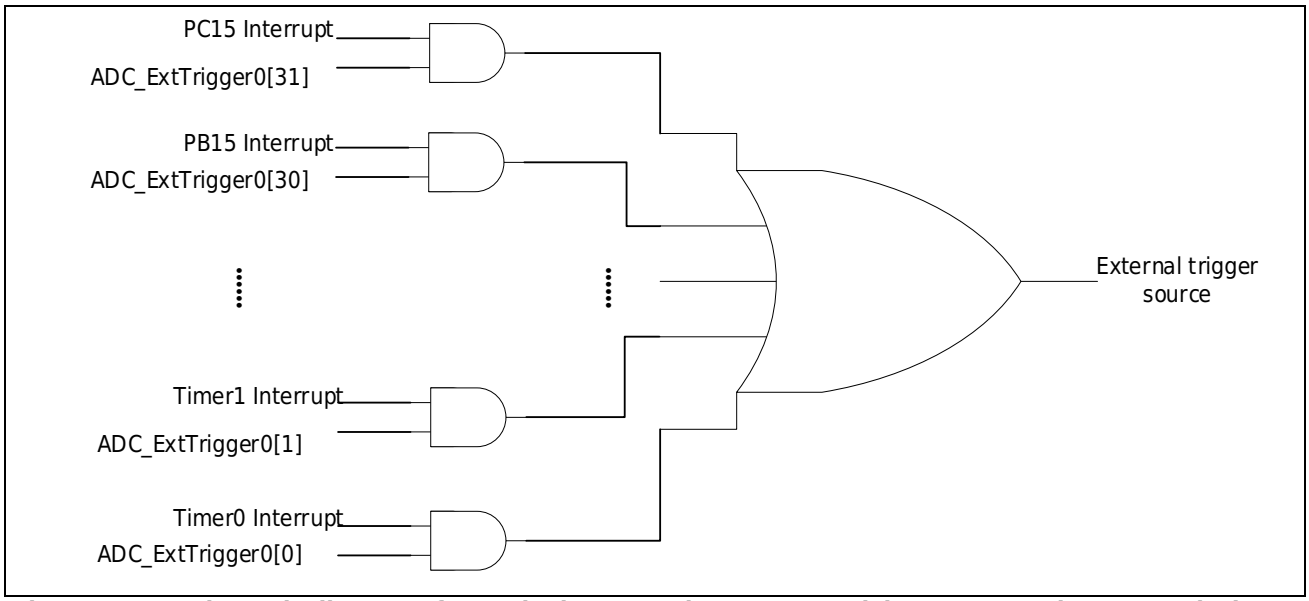


Figure 31-7 Schematic diagram of ADC single conversion or sequential scan conversion external trigger source

Configure the ADC_ExtTrigger1 register to set the external trigger source for ADC jump-in scan conversion.

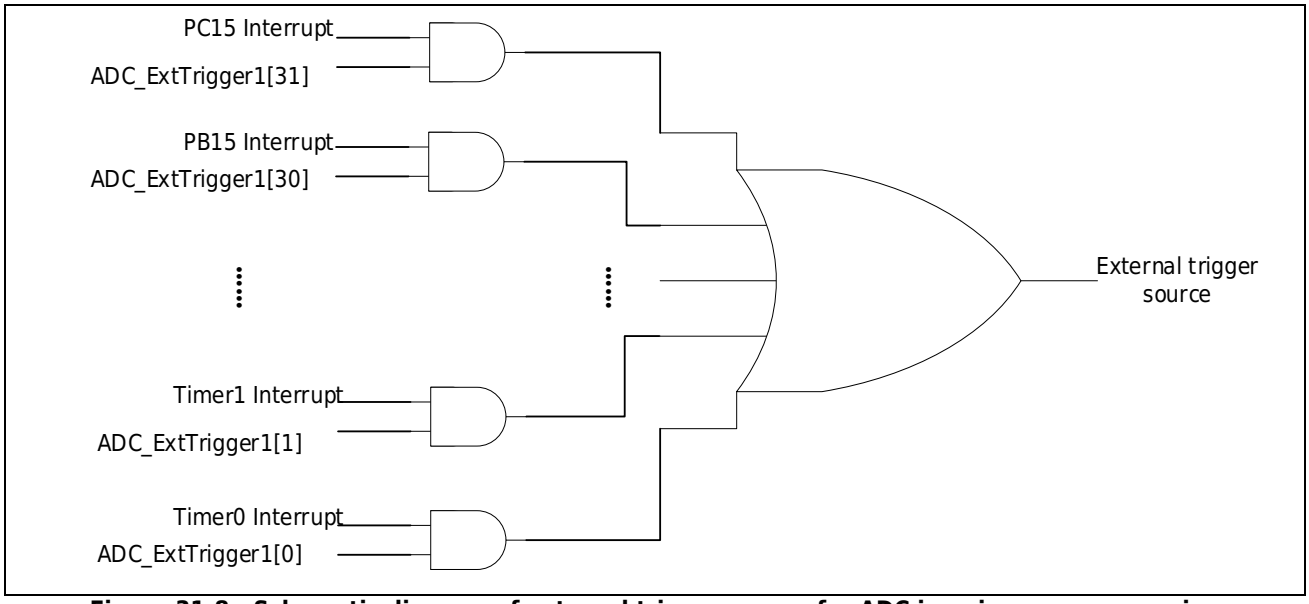


Figure 31-8 Schematic diagram of external trigger source for ADC jumping scan conversion

31.8ADC conversion result comparison

When the ADC conversion is completed, the ADC conversion result can be compared with the threshold set by the user, supporting upper threshold comparison, lower threshold comparison, and interval value comparison. This function needs to set the corresponding control bits HtCmp, LtCmp, RegCmp to 1. This function can realize the automatic monitoring of the analog quantity, until the ADC conversion result meets the user's expectations, an interrupt is generated to apply for user program interface entry. The monitoring channel selection is configured through ADC_CR1.ThCh.

Upper threshold comparison: When the ADC conversion result is within the range [ADC_HT, 4095], ADC_IFR.HTIF is set to 1; writing 0 to ADC_ICR.HTIC clears ADC_IFR.HTIF.

Lower threshold comparison: When the ADC conversion result is within the range [0, ADC_LT), ADC_IFR.LTIF is set to 1; writing 0 to ADC_ICR.LTIC clears ADC_IFR.LTIF.

Interval value comparison: When the ADC conversion result is within the [ADC_LT, ADC_HT) interval, ADC_IFR.REGIF is set to 1; writing 0 to ADC_ICR.REGIC clears ADC_IFR.REGIF.

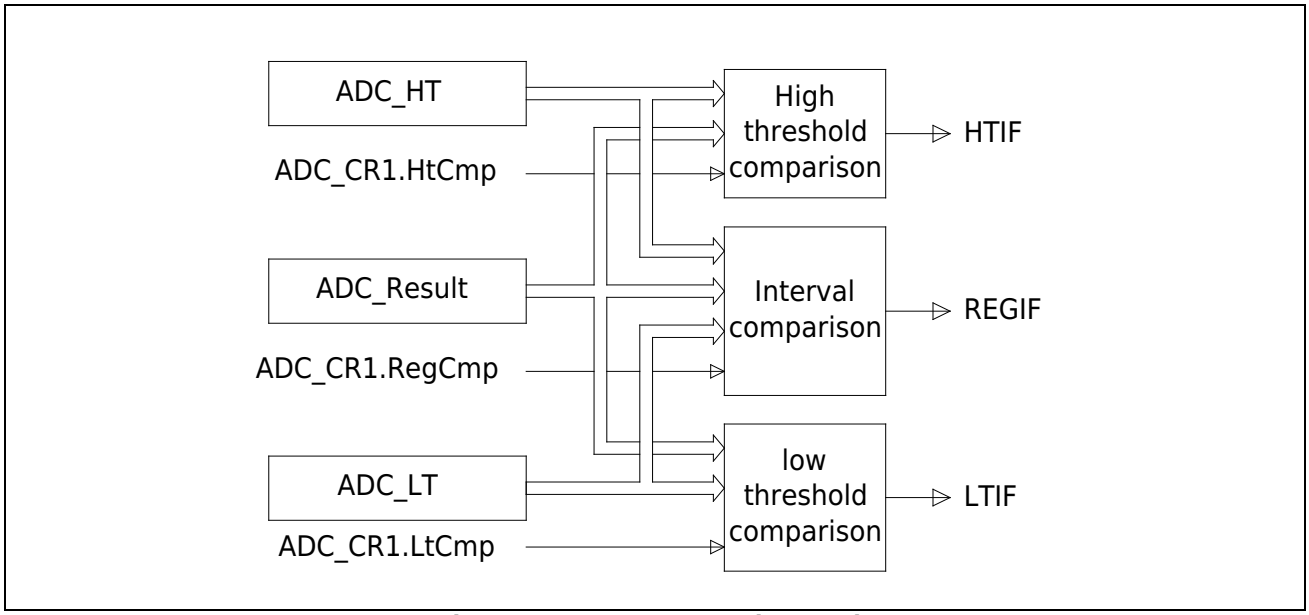


Figure 31-9 ADC conversion result

31.9ADC Interrupt

The ADC interrupt request is shown in the table below:

Interrupt source	Interrupt logo	Interrupt enable
ADC jump queue scan conversion completed	ADC_IFR.JQRIF	ADC_CR0.IE
ADC sequential scan conversion complete	ADC_IFR.SQRIF	
The ADC conversion result is in the interval value area (greater than or equal to the lower threshold and less than the upper threshold)	ADC_IFR.REGIF	
The ADC conversion result is in the upper threshold region	ADC_IFR.HTIF	

(greater than or equal to the upper threshold)		
ADC conversion result comparison lower threshold area (less than lower threshold)	ADC_IFR.LTIF	
ADC single conversion completed	ADC_IFR.SGLIF	

31.10 Measure ambient temperature using a temperature sensor

The output voltage of the temperature sensor will change with the change of the ambient temperature, so the corresponding ambient temperature can be calculated according to the output voltage of the temperature sensor. When the measurement channel of the ADC module selects the output voltage of the temperature sensor, the ambient temperature can be measured.

Calculated as follows:

Ambient Temperature = 25 + 0.0854 × Vref × (AdcValue – Trim)

Among them: Vref is the reference voltage of the current ADC module, and the value is 1.5 or 2.5.

AdcValue is the result of measuring the output voltage of the temperature sensor by the ADC module, and the value is 0~4095.

Trim is a calibration value of 16Bits, which needs to be read from the Flash memory during calculation, and its storage address is detailed in the table below.

ADC reference voltage	Calibration value storage address	Calibration value accuracy
Internal 1.5V	0x00100C34	±3℃
Internal 2.5V	0x00100C36	±3℃

An example calculation is as follows:

Condition 1: Vref=2.5, AdcValue=0x7E5, Trim=0x76C:

Temperature 1: 25 + 0.0854 × 2.5 × (0x7E5 – 0x76C) = 50.8 °C.

Condition 2: Vref=1.5, AdcValue=0x72D, Trim=0x76C:

Temperature 2: 25 + 0.0854 × 1.5 × (0x72D – 0x76C) = 16.9°C.

Operation process of measuring ambient temperature through ADC:

Step1: Set BGR_CR to 3, enable BGR module and temperature sensor module.

Step2: Set ADC_CR0.En to 1 to enable the ADC module.

Step3: Delay 20us, wait for the ADC and BGR module to start up.

Step4: Set ADC_CR1.Mode to 0, select single conversion mode.

Step5: Set ADC_CR0.InRefEn to 1 to enable ADC internal reference voltage.

Step6: Configure ADC_CR0.Ref, select the reference voltage of ADC as internal 1.5V or internal 2.5V.

Step7: Configure ADC_CR0.SAM and ADC_CR0.CkDiv to set the conversion speed of ADC.

Step8: Set ADC_CR0.SGLMux to 0x1C, and select the channel to be converted as the output of the temperature sensor.

Step9: Set ADC_CR0.Buf to 1 to enable the input signal amplifier.

Step10: Set ADC_ICR.SGLIC to 0 and clear the ADC_IFR.SGLIF flag.

Step11: Set ADC_SglStart.Start to 1 to start ADC single conversion.

Step12: Wait for ADC_IFR.SGLIF to become 1, read the ADC_Result register to get the ADC conversion result.

Step13: Set ADC_CR0.En and BGR_CR to 0, turn off the ADC module, BGR module, and temperature sensor module.

Step14: Read the calibration value of the temperature sensor and calculate the current ambient temperature according to the formula.

31.11 ADC module registers

Base address 0x40002400

Table 31-1 ADC register

Register	Offset address	Description
ADC_CR0	0x004	ADC basic configuration register 0
ADC_CR1	0x008	ADC basic configuration register 1
ADC_SQR0	0x040	ADC sequential scan conversion channel configuration register 0
ADC_SQR1	0x044	ADC sequential scan conversion channel configuration register 1
ADC_SQR2	0x048	ADC sequential scan conversion channel configuration register 2
ADC_JQR	0x04C	ADC queue scan conversion channel configuration register
ADC_SqrResult0	0x050	ADC sequential scan conversion channel 0 conversion result
ADC_SqrResult1	0x054	ADC sequential scan conversion channel 1 conversion result
ADC_SqrResult2	0x058	ADC sequential scan conversion channel 2 conversion result
ADC_SqrResult3	0x05C	ADC sequential scan conversion channel 3 conversion result
ADC_SqrResult4	0x060	ADC sequential scan conversion channel 4 conversion result
ADC_SqrResult5	0x064	ADC sequential scan conversion channel 5 conversion result
ADC_SqrResult6	0x068	ADC sequential scan conversion channel 6 conversion result
ADC_SqrResult7	0x06C	ADC sequential scan conversion channel 7 conversion result
ADC_SqrResult8	0x070	ADC sequential scan conversion channel 8 conversion results
ADC_SqrResult9	0x074	ADC sequential scan conversion channel 9 conversion result
ADC_SqrResult10	0x078	ADC sequential scan conversion channel 10 conversion result
ADC_SqrResult11	0x07C	ADC sequential scan conversion channel 11 conversion result
ADC_SqrResult12	0x080	ADC sequential scan conversion channel 12 conversion result
ADC_SqrResult13	0x084	ADC sequential scan conversion channel 13 conversion result
ADC_SqrResult14	0x088	ADC sequential scan conversion channel 14 conversion result
ADC_SqrResult15	0x08C	ADC sequential scan conversion channel 15 conversion result
ADC_JqrResult0	0x090	ADC skipping scan conversion channel 0 conversion result
ADC_JqrResult1	0x094	ADC skipping scan conversion channel 1 conversion result
ADC_JqrResult2	0x098	ADC skipping scan conversion channel 2 conversion result
ADC_JqrResult3	0x09C	ADC skipping scan conversion channel 3 conversion result
ADC_Result	0x0A0	ADC conversion result
ADC_ResultAcc	0x0A4	Accumulated value of ADC conversion result
ADC_HT	0x0A8	ADC comparison upper threshold
ADC_LT	0x0AC	ADC comparison lower threshold
ADC_IFR	0x0B0	ADC Interrupt Flag Register
ADC_ICR	0x0B4	ADC Interrupt Clear Register
ADC_ExtTrigger0	0x0B8	ADC single conversion or sequential scan conversion external interrupt trigger source configuration register
ADC_ExtTrigger1	0x0BC	ADC jumping scan conversion external interrupt trigger source

Register	Offset address	Description
		configuration register
ADC_SglStart	0x0C0	ADC single conversion start control register
ADC_SqrStart	0x0C4	ADC sequential scan conversion start control register
ADC_JqrStart	0x0C8	ADC jump scan conversion start control register
ADC_allstart	0x0CC	ADC has been converting start and stop control register

31.11.1 ADC Basic Configuration Register 0 (ADC_CR0)

Offset address 0x004

Reset value 0x000067F0

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	InRef En	SAM		Buf	Ref		SGLMux					CkDiv		CHM AP	En
RW	RW	RW		RW	RW		RW					RW		RW	RW

Bit	Marking	Functional description	
31:16	Reserved	Keep	
15	IE	ADC interrupt control 1: enable interrupt 0: disable interrupt	
14	InRefEn	ADC internal reference voltage enable 1: enable internal voltage reference 0: disable internal voltage reference	
13:12	SAM	ADC sampling period selection 00: 4 conversion cycles 01: 6 conversion cycles 10: 8 conversion cycles 11: 12 conversion cycles	
11	Buf	Built-in follower enable control 0: Disable the follower, and the external input signal is directly connected to the ADC. 1: Turn on the follower, and the external input signal is connected to the ADC after passing through the follower to measure high-impedance signals. Note 1: When this function is enabled, the maximum rate is 200Ksps Note 2: The following conditions must enable this function The output impedance of the signal to be converted is high The signal to be converted is: AVCC/3, built-in temperature sensor output voltage, DAC output	
10:9	Ref	ADC reference voltage selection 00: Internal 1.5V 01: Internal 2.5V 10: External reference voltage ExRef (PB01) 11: AVCC voltage	
8:4	SGLMux	Single Conversion Mode Conversion Channel Selection (ADC_CR0.CHMAP=0) 00000: AIN0(PA00) 00001: AIN1(PA01) 00010: AIN2(PA02) 00011: AIN3(PA03) 00100: AIN4(PA04) 00101: AIN5(PA05) 00110: AIN6(PA06) 00111: AIN7(PA07) 01000: AIN8(PB00) 01001: AIN9(PB01) 01010: AIN10(PC00) 01011: AIN11(PC01) 01100: AIN12(PC02) 01101: AIN13(PC03) 01110: AIN14(PC04) 01111: AIN15(PC05)	Single Conversion Mode Conversion Channel Selection (ADC_CR0.CHMAP=1) 00000: AIN0(PD08) 00001: AIN1(PD09) 00010: AIN2(PD10) 00011: AIN3(PD11) 00100: AIN4(PA04) 00101: AIN5(PA05) 00110: AIN6(PE08) 00111: AIN7(PE09) 01000: AIN8(PE10) 01001: AIN9(PB01) 01010: AIN10(PE11) 01011: AIN11(PE12) 01100: AIN12(PE13) 01101: AIN13(PE14) 01110: AIN14(PC04) 01111: AIN15(PC05)

		10000: AIN16(PB02) 10001: AIN17(PB10) 10010: AIN18(PB11) 10011: AIN19(PB12) 10100: AIN20(PB13) 10101: AIN21(PB14) 10110: AIN22(PB15) 10111: AIN23(PE15) 11000: AIN24(PC07)	10000: AIN16(PB02) 10001: AIN17(PB10) 10010: AIN18(PB11) 10011: AIN19(PB12) 10100: AIN20(PB13) 10101: AIN21(PB14) 10110: AIN22(PB15) 10111: AIN23(PE15) 11000: AIN24(PC07)
		11001: DAC0 output Note: ADC_CR0.Buf must be 1 11010: DAC1 output Note: ADC_CR0.Buf must be 1 11011: AVCC/3 Note: ADC_CR0.Buf must be 1 11100: Built-in temperature sensor output voltage Note: ADC_CR0.Buf must be 1 11101: Reserved	
3:2	CkDiv	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	CHMAP	OPA to DAC input mapping 0: use the port channel selected by SGLMUX 1: Part of the channel is mapped to the relevant port of the OPA output	
0	En	ADC enable control 1: enable ADC 0: disable ADC	

CHMAP is 1, the following ADC channel ports are mapped to other ports.

AIN6M	PE08
AIN7M	PE09
AIN8M	PE10
AIN10M	PE11
AIN11M	PE12
AIN12M	PE13
AIN13M	PE14
AIN0M	PD08
AIN1M	PD09
AIN2M	PD10
AIN3M	PD11

31.11.2 ADC Basic Configuration Register 1 (ADC_CR1)

Offset address 0x008

Reset value 0x00008000

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
RAccClr	RegCmp	HtCmp	LtCmp	RAccEn	Mode	DmaJqr	DmaSqr	ThCh					Align	Reserved	
WO	RW	RW	RW	RW	RW	RW	RW	RW					RW		

Bit	Marking	Functional description
31:16	Reserved	Keep
15	RAccClr	ADC conversion result accumulation register is cleared 1: no effect; 0: ADC conversion result accumulation register (ADC_ResultAcc) is cleared.
14	RegCmp	ADC interval comparison control 1: enable range comparison 0: interval comparison disabled
13	HtCmp	ADC high threshold compare control 1: enable high threshold compare 0: disable high threshold comparison
12	LtCmp	ADC low threshold compare control 1: enable low threshold compare 0: disable low threshold comparison
11	RAccEn	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	Mode	ADC conversion mode selection 1: scan conversion mode 0: single conversion mode
9	DmaJqr	Jump scan trigger DMA read control 1: Enable skip scan conversion trigger DMA read 0: Disable queue jump scan conversion to trigger DMA read
8	DmaSqr	Sequential scan trigger DMA read control 1: Enable sequential scan conversions to trigger DMA reads 0: Disable sequential scan conversion triggering DMA read
7:3	ThCh	Threshold comparison channel selection 00000: select channel 0 for threshold comparison 00001: Select channel 1 for threshold comparison 00010: Select channel 2 for threshold comparison ..... 11101: Select channel 29 for threshold comparison
2	Align	Conversion result alignment control 1: The conversion result is left-aligned and stored in 16Bits 0: conversion result 16Bits right-aligned storage
1:0	Reserved	Keep

31.11.3 ADC Sequential Scan Conversion Channel Configuration Register 0 (ADC_SQR0)

Offset address 0x040

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		CH5Mux					CH4Mux					CH3Mux			
		RW					RW					RW			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CH2Mux					CH1Mux					CH0Mux				
	RW					RW					RW				

Bit	Marking	Functional description
31:30	Reserved	Keep
29:25	CH5Mux	Sequential scan conversion channel 5 selection, see ADC_CR0.SGLMux for settings
24:20	CH4Mux	Sequential scan conversion channel 4 selection, see ADC_CR0.SGLMux for settings
19:15	CH3Mux	Sequential scan conversion channel 3 selection, see ADC_CR0.SGLMux for settings
14:10	CH2Mux	Sequential scan conversion channel 2 selection, see ADC_CR0.SGLMux for settings
9:5	CH1Mux	Sequential scan conversion channel 1 selection, see ADC_CR0.SGLMux for settings
4:0	CH0Mux	Sequential scan conversion channel 0 selection, see ADC_CR0.SGLMux for settings

31.11.4 ADC Sequential Scan Conversion Channel Configuration Register 1 (ADC_SQR1)

Offset address 0x044

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		CH11Mux					CH10Mux					CH9Mux			
		RW					RW					RW			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CH8Mux					CH7Mux					CH6Mux				
	RW					RW					RW				

Bit	Marking	Functional description
31:30	Reserved	Keep
29:25	CH11Mux	Sequential scan conversion channel 11 selection, see ADC_CR0.SGLMux for settings
24:20	CH10Mux	Sequential scan conversion channel 10 selection, see ADC_CR0.SGLMux for settings
19:15	CH9Mux	Sequential scan conversion channel 9 selection, see ADC_CR0.SGLMux for settings
14:10	CH8Mux	Sequential scan conversion channel 8 selection, see ADC_CR0.SGLMux for settings
9:5	CH7Mux	Sequential scan conversion channel 7 selection, see ADC_CR0.SGLMux for settings
4:0	CH6Mux	Sequential scan conversion channel 6 selection, see ADC_CR0.SGLMux for settings

31.11.5 ADC Sequential Scan Conversion Channel Configuration Register 2 (ADC_SQR2)

Offset address 0x048

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								CNT				CH15Mux			
								RW				RW			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CH14Mux					CH13Mux					CH12Mux				
	RW					RW					RW				

Bit	Marking	Functional description
31:24	Reserved	Keep
23:20	CNT	Sequential Scan Conversions
19:15	CH15Mux	Sequential scan conversion channel 15 selection, see ADC_CR0.SGLMux for settings
14:10	CH14Mux	Sequential scan conversion channel 14 selection, see ADC_CR0.SGLMux for settings
9:5	CH13Mux	Sequential scan conversion channel 13 selection, see ADC_CR0.SGLMux for settings
4:0	CH12Mux	Sequential scan conversion channel 12 selection, see ADC_CR0.SGLMux for settings

31.11.6 ADC jumping scan conversion channel configuration register (ADC_JQR)

Offset address 0x04C

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										CNT		CH3Mux			
										RW		RW			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CH2Mux					CH1Mux					CH0Mux				
	RW					RW					RW				

Bit	Marking	Functional description
31:24	Reserved	Keep
21:20	CNT	Skip Scan Conversions
19:15	CH3Mux	Queue scan conversion channel 3 selection, see ADC_CR0.SGLMux for settings
14:10	CH2Mux	Queue scan conversion channel 2 selection, see ADC_CR0.SGLMux for settings
9:5	CH1Mux	Queue scan conversion channel 1 selection, see ADC_CR0.SGLMux for settings
4:0	CH0Mux	Queue scan conversion channel 0 selection, see ADC_CR0.SGLMux for settings

31.11.7 ADC sequential scan conversion channel x conversion result (ADC_SqrResult0 - 15)

Offset address 0x050 ~ 0x8C

Reset value 0x00000000

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

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

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	Result	ADC sequential scan conversion channel x conversion result

31.11.8 ADC jump scan conversion channel x conversion result (ADC_JqrResult0 - 3)

Offset address 0x090 ~ 0x9C

Reset value 0x00000000

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

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

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	Result	ADC skipping scan conversion channel x conversion result

31.11.9 ADC conversion result (ADC_Result)

Offset address 0x0A0

Reset value 0x00000000

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

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

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	Result	ADC conversion results

31.11.10 Accumulated value of ADC conversion result (ADC_ResultAcc)

Offset address 0x0A4

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												ResultAcc[19:16]			
												RO			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ResultAcc[15:0]															
RO															

Bit	Marking	Functional description
31:20	Reserved	Keep
19:0	ResultAcc	ADC conversion accumulated value

31.11.11 ADC Compare Upper Threshold (ADC_HT)

Offset address 0x0A8

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											
				RW											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	HT	ADC conversion result comparison upper threshold

31.11.12 ADC Compare Lower Threshold (ADC_LT)

Offset address 0x0AC

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											
				RW											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	LT	ADC conversion result comparison lower threshold

31.11.13 ADC Interrupt Flag Register (ADC_IFR)

Offset address 0x0B0

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										JQRIF	SQRIF	REGIF	HHT_INTF	LLT_INTF	S_INTF
										RO	RO	RO	RO	RO	RO

Bit	Marking	Functional description
31:6	Reserved	Keep
5	JQRIF	ADC skipping scan conversion complete flag 1: ADC skipping scan conversion completed 0: ADC jump scan conversion is not completed
4	SQRIF	ADC sequential scan conversion complete flag 1: ADC sequential scan conversion complete 0: ADC sequential scan conversion not completed
3	REGIF	ADC conversion result comparison interval flag 1: The ADC conversion result is within the range of [ADC_LT, ADC_HT) 0: The ADC conversion result is outside the [ADC_LT, ADC_HT) range
2	HTIF	ADC conversion result comparison upper threshold flag 1: The ADC conversion result is within the range [ADC_HT, 4095] 0: ADC conversion result is outside [ADC_HT, 4095] range
1	LTIF	ADC conversion result comparison lower threshold flag 1: The ADC conversion result is in the range [0, ADC_LT) 0: The ADC conversion result is outside the interval [0, ADC_LT)
0	SGLIF	ADC single conversion complete flag 1: ADC single conversion completed 0: ADC single conversion not completed

31.11.14 ADC Interrupt Clear Register (ADC_ICR)

Offset address 0x0B4

Reset value 0x0000003F

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										JQRIC	SQRIC	REGIC	HHT_INTC	LLT_INTC	S_INTC
										R1W0	R1W0	R1W0	R1W0	R1W0	R1W0

Bit	Marking	Functional description
31:6	Reserved	Keep
5	JQRIC	Write 0 to clear the ADC queue scan conversion completion flag Write 1 has no effect
4	SQRIC	Write 0 to clear the ADC sequential scan conversion complete flag Write 1 has no effect
3	REGIC	Write 0 to clear the ADC conversion result comparison interval flag Write 1 has no effect
2	HTIC	Write 0 to clear ADC conversion result comparison upper threshold Write 1 has no effect
1	LTIC	Write 0 to clear ADC conversion result comparison lower threshold flag Write 1 has no effect
0	SGLIC	Write 0 to clear the ADC single conversion complete flag Write 1 has no effect

31.11.15 ADC single conversion or sequential scan conversion external interrupt trigger source configuration register (ADC_ExtTrigger0)

Offset address 0x0B8

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC15	PB15	PA15	PC11	PB11	PA11	PD07	PC07	PB07	PA07	PD03	PC03	PB03	PA03	DMA	SPI1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI0	PCA	RTC	VC1	VC0	LPUART1	LPUART0	UART1	UART0	TIM6	TIM5	TIM4	TIM3	TIM2	TIM1	TIM0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31	PC15	PC15 interrupt triggers ADC conversion
30	PB15	PB15 interrupt triggers ADC conversion
29	PA15	PA15 interrupt triggers ADC conversion
28	PC11	PC11 interrupt triggers ADC conversion
27	PB11	PB11 interrupt triggers ADC conversion
26	PA11	PA11 interrupt triggers ADC conversion
25	PD07	PD07 interrupt triggers ADC conversion
24	PC07	PC07 interrupt triggers ADC conversion
23	PB07	PB07 interrupt triggers ADC conversion
22	PA07	PA07 interrupt triggers ADC conversion
21	PD03	PD03 interrupt triggers ADC conversion
20	PC03	PC03 interrupt triggers ADC conversion
19	PB03	PB03 interrupt triggers ADC conversion
18	PA03	PA03 interrupt triggers ADC conversion
17	DMA	DMA interrupt triggers ADC conversion
16	SPI1	SPI1 interrupt triggers ADC conversion
15	SPI0	SPI0 interrupt triggers ADC conversion
14	PCA	PCA interrupt triggers ADC conversion
13	RTC	RTC interrupt triggers ADC conversion
12	VC1	VC1 interrupt triggers ADC conversion
11	VC0	VC0 interrupt triggers ADC conversion
10	LPUART1	LPUART1 interrupt triggers ADC conversion
9	LPUART0	LPUART0 interrupt triggers ADC conversion
8	UART1	UART1 interrupt triggers ADC conversion
7	UART0	UART0 interrupt triggers ADC conversion

6	TIM6	Timer6 interrupt triggers ADC conversion
5	TIM5	Timer5 interrupt triggers ADC conversion
4	TIM4	Timer4 interrupt triggers ADC conversion
3	TIM3	Timer3 interrupt triggers ADC conversion
2	TIM2	Timer2 interrupt triggers ADC conversion
1	TIM1	Timer1 interrupt triggers ADC conversion
0	TIM0	Timer0 interrupt triggers ADC conversion

Notes:

1) The TIM4/5/6 interrupt triggers the automatic conversion of the ADC. In addition to enabling the corresponding interrupt of TIM4/5/6, it is also necessary to configure the Advanced Timer's spread spectrum and interrupt trigger selection register TIMX_CR to select the interrupt source that can trigger the ADC.
2) The ADC is triggered using the rising edge of each interrupt flag bit. 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.

31.11.16 ADC jump scan conversion external interrupt trigger source configuration register (ADC_ExtTrigger1)

Offset address 0x0BC

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC15	PB15	PA15	PC11	PB11	PA11	PD07	PC07	PB07	PA07	PD03	PC03	PB03	PA03	DMA	SPI1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI0	PCA	RTC	VC1	VC0	LPUART1	LPUART0	UART1	UART0	TIM6	TIM5	TIM4	TIM3	TIM2	TIM1	TIM0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31	PC15	PC15 interrupt triggers ADC conversion
30	PB15	PB15 interrupt triggers ADC conversion
29	PA15	PA15 interrupt triggers ADC conversion
28	PC11	PC11 interrupt triggers ADC conversion
27	PB11	PB11 interrupt triggers ADC conversion
26	PA11	PA11 interrupt triggers ADC conversion
25	PD07	PD07 interrupt triggers ADC conversion
24	PC07	PC07 interrupt triggers ADC conversion
23	PB07	PB07 interrupt triggers ADC conversion
22	PA07	PA07 interrupt triggers ADC conversion
21	PD03	PD03 interrupt triggers ADC conversion
20	PC03	PC03 interrupt triggers ADC conversion
19	PB03	PB03 interrupt triggers ADC conversion
18	PA03	PA03 interrupt triggers ADC conversion
17	DMA	DMA interrupt triggers ADC conversion
16	SPI1	SPI1 interrupt triggers ADC conversion
15	SPI0	SPI0 interrupt triggers ADC conversion
14	PCA	PCA interrupt triggers ADC conversion
13	RTC	RTC interrupt triggers ADC conversion
12	VC1	VC1 interrupt triggers ADC conversion
11	VC0	VC0 interrupt triggers ADC conversion
10	LPUART1	LPUART1 interrupt triggers ADC conversion
9	LPUART0	LPUART0 interrupt triggers ADC conversion
8	UART1	UART1 interrupt triggers ADC conversion
7	UART0	UART0 interrupt triggers ADC conversion

6	TIM6	Timer6 interrupt triggers ADC conversion
5	TIM5	Timer5 interrupt triggers ADC conversion
4	TIM4	Timer4 interrupt triggers ADC conversion
3	TIM3	Timer3 interrupt triggers ADC conversion
2	TIM2	Timer2 interrupt triggers ADC conversion
1	TIM1	Timer1 interrupt triggers ADC conversion
0	TIM0	Timer0 interrupt triggers ADC conversion

Notes:

1) The TIM4/5/6 interrupt triggers the automatic conversion of the ADC. In addition to enabling the corresponding interrupt of TIM4/5/6, it is also necessary to configure the Advanced Timer's spread spectrum and interrupt trigger selection register TIMX_CR to select the interrupt source that can trigger the ADC.
2) The ADC is triggered using the rising edge of each interrupt flag bit. 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.

31.11.17 ADC Single Conversion Start Control Register (ADC_SglStart)

Offset address 0x0C0

Reset value 0x00000000

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

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

Bit	Marking	Functional description
31:1	Reserved	Keep
0	Start	ADC single conversion start control 1: start ADC single conversion 0: stop ADC single conversion

31.11.18 ADC Sequential Scan Conversion Start Control Register (ADC_SqrStart)

Offset address 0x0C4

Reset value 0x00000000

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

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

Bit	Marking	Functional description
31:1	Reserved	Keep
0	Start	ADC sequential scan conversion start control 1: start ADC sequential scan conversion 0: stop ADC sequential scan conversion

31.11.19 ADC jump scan conversion start control register (ADC_JqrStart)

Offset address 0x0C8

Reset value 0x00000000

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

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

Bit	Marking	Functional description
31:1	Reserved	Keep
0	Start	ADC jumping scan conversion start control 1: start ADC jumping scan conversion 0: stop ADC jumping scan conversion

31.11.20 ADC has been converting start stop control register (ADC_allStart)

Offset address 0x0CC

Reset value 0x00000000

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

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

Bit	Marking	Functional description
31:1	Reserved	Keep
0	Start	ADC jumping scan conversion start control 1: enable ADC to convert all the time 0: stop ADC conversion and need to wait until the end of the current conversion Read 0 for no conversion; 1 for conversion

32 Digital to Analog Converter (DAC)

The DAC module is a 12 -bit voltage output digital-to-analog converter. The DAC can be configured in 8- bit or 12- bit mode and can be used with a DMA controller. In 12- bit mode, data can be left-justified or right-justified. Input reference voltage VREF+ (shared with ADC) can be used. The output can be optionally buffered for higher current drive.

32.1 DAC Characteristic

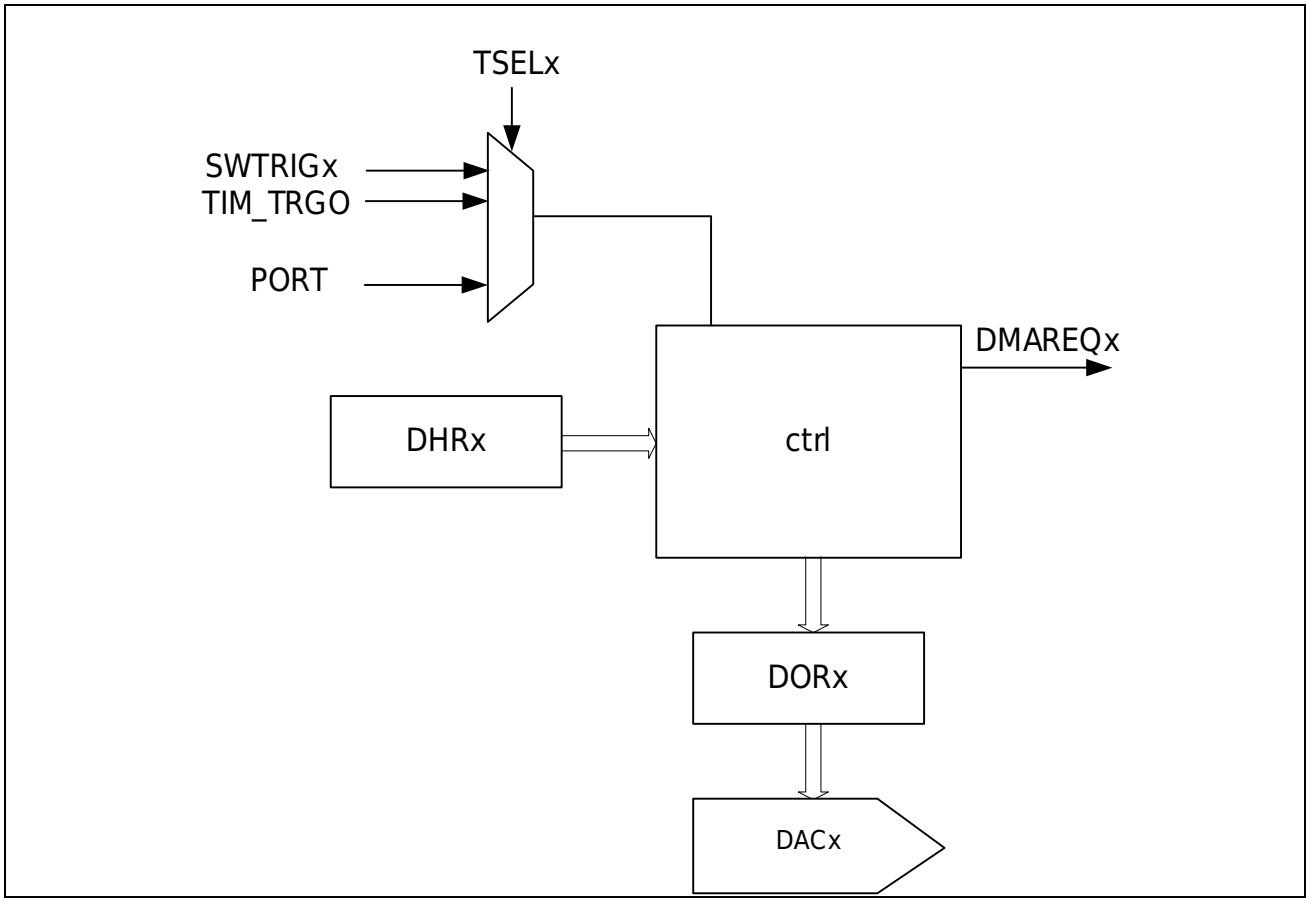
The device integrates two DAC converters, each with one output channel: DAC0_OUT and DAC1_OUT.

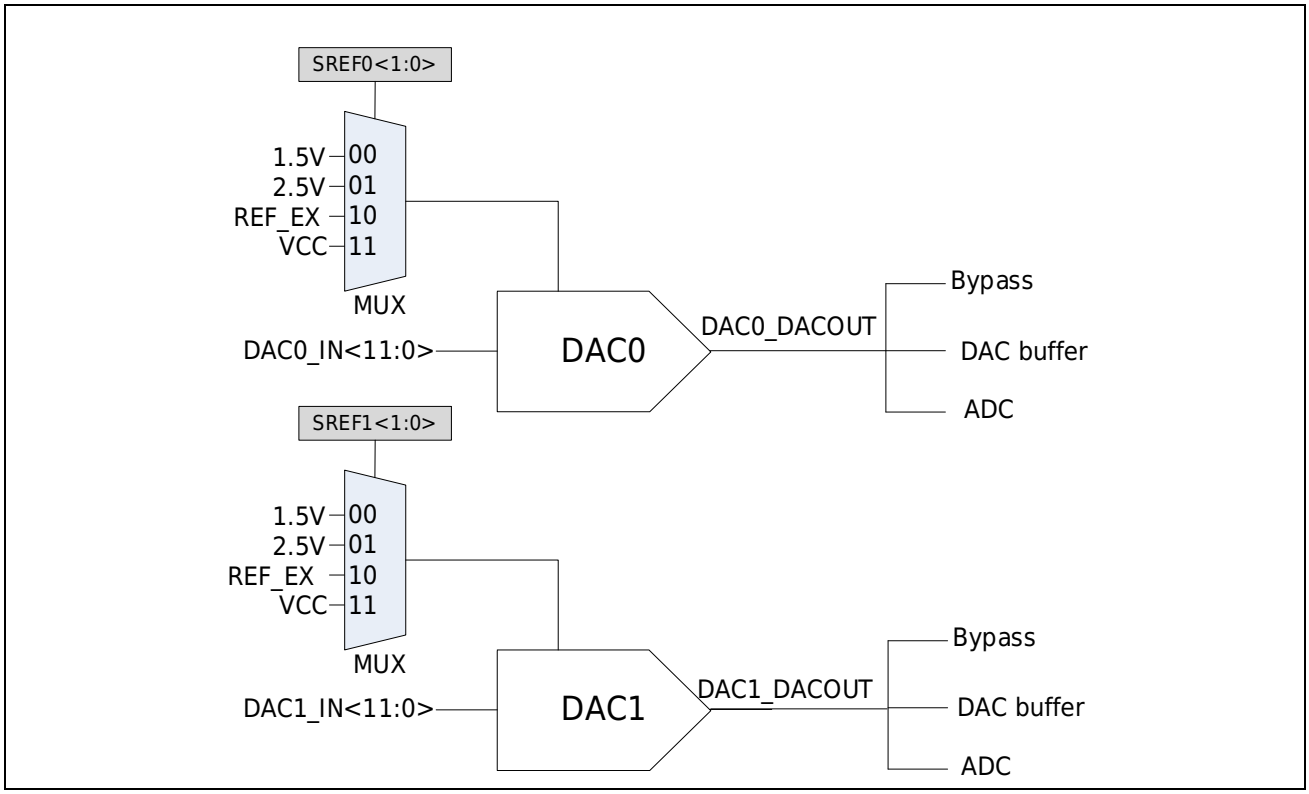
The main features of the DAC are as follows:

- A data holding register
- Data is left-justified or right-justified in 12- bit mode
- Synchronous update function
- Generate noise
- Generate triangle wave
- Dual DAC channels, support separate or simultaneous conversion
- DMA functions (including underflow detection)
- Conversion via external trigger signal
- 4 reference sources: AVCC voltage, ExRef pin, built-in 1.5v reference voltage, built-in 2.5v reference voltage;

32.2Functional description

32.2.1 Block diagram





32.2.2 DAC output buffer enable

The DAC can use an output buffer, which can be used to reduce the output impedance and directly drive an external load without adding an external operational amplifier. For the settings of the output buffer, refer to the chapter < Operational Amplifier OPA>

DAC channel output buffer can be enabled or disabled via the BOFF1 bit in the DAC_CR register.

32.2.3 DAC channel enable

DAC channel can be enabled by setting the corresponding ENx bit in the DAC_CR register to 1. After a period of startup t_{WAKEUP}, each DAC channel is actually enabled.

The ENx bits only enable the analog DAC channel x units. Even if the ENx bit is reset, the DAC channel x digital interface is still enabled.

32.2.4 DAC output voltage

After a linear conversion, the digital input is converted to an output voltage between 0 and VREF+. The analog output voltage at each DAC channel pin is determined by the following formula:

$$DAC_{out} = V_{ref} * DOR/4096$$

32.2.5 DAC trigger selection

If the TENx control bit is set, the conversion can be triggered by an external event (timer counter, external interrupt line) or software. The TSELx[2:0] control bits will determine which possible event will trigger the conversion as shown in the table below.

TSEL	000	001	010	011	100	101	110	111
Trigger source	TIM0_ TRADC	TIM1_ TRADC	TIM2_ TRADC	TIM3_ TRADC	TIM4_ TRADC	TIM5_ TRADC	SW TRIG	EXTI

Whenever a rising edge is detected on the DAC selected timer TRGO output or on the selected external port trigger, the last data stored in the DAC_DHRx register is transferred to the DAC_DORx register. APB1 cycles after the trigger, the DAC_DORx registers will be updated.

If software triggering is selected, the conversion starts as soon as the SWTRIG bit is set. After the DAC_DHRx register contents are loaded into the DAC_DORx register, SWTRIG is reset by hardware.

Note:

– The TSELx[2:0] bits cannot be changed while the ENx bit is set. If software triggering is selected, the contents of the DAC_DHRx registers are transferred to the DAC_DORx registers in only one APB1 clock cycle.

32.2.6 Single mode function description

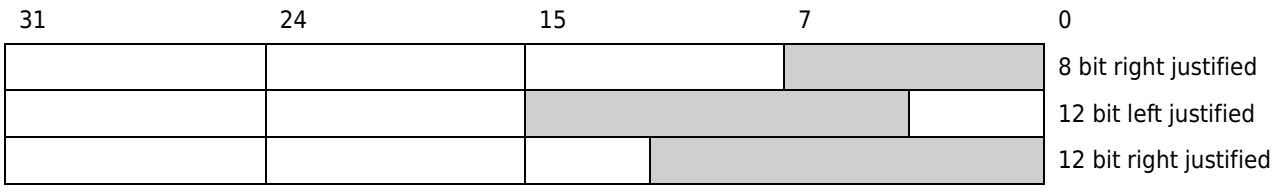
32.2.6.1 DAC data format

Depending on the selected configuration mode, data must be written to the specified registers as follows:

There are three possibilities:

- 8 -bit right justified: software must load data into DAC_DHR8Rx[7:0] bits (store into DHRx[11:4] bits)
- 12- bit left justified: software must load data into DAC_DHR12Lx[15:4] bits (store into DHRx[11:0] bits)
- 12 -bit right justified: software must load data into DAC_DHR12Rx[11:0] bits (store into DHRx[11:0] bits)

According to the loaded DAC_DHRyyyx register, the data written by the user will be shifted and stored into the corresponding DHRx (Data Holding Register x, which is an internal non-memory-mapped register). After that, the DHRx register will be loaded automatically, or loaded to the DORx register triggered by software or external event.



32.2.6.2 DAC conversion

DAC_DORx cannot be written directly, any data must be transferred to DAC channel x by loading the DAC_DHRx register (write to DAC_DHR8Rx, DAC_DHR12Lx, DAC_DHR12Rx).

If triggering is not selected (TENx bit reset in DAC_CR register), then the data stored in the DAC_DHRx register is automatically transferred to the DAC_DORx register after one APB1 clock cycle. However, if the hardware trigger is selected (set the TENx bit in the DAC_CR register) and the trigger condition comes, the transfer will take place after three PCLK1 clock cycles.

When DAC_DORx is loaded with DAC_DHRx content, the analog output voltage will be available after a period of tSETTLING depending on supply voltage and analog output load.

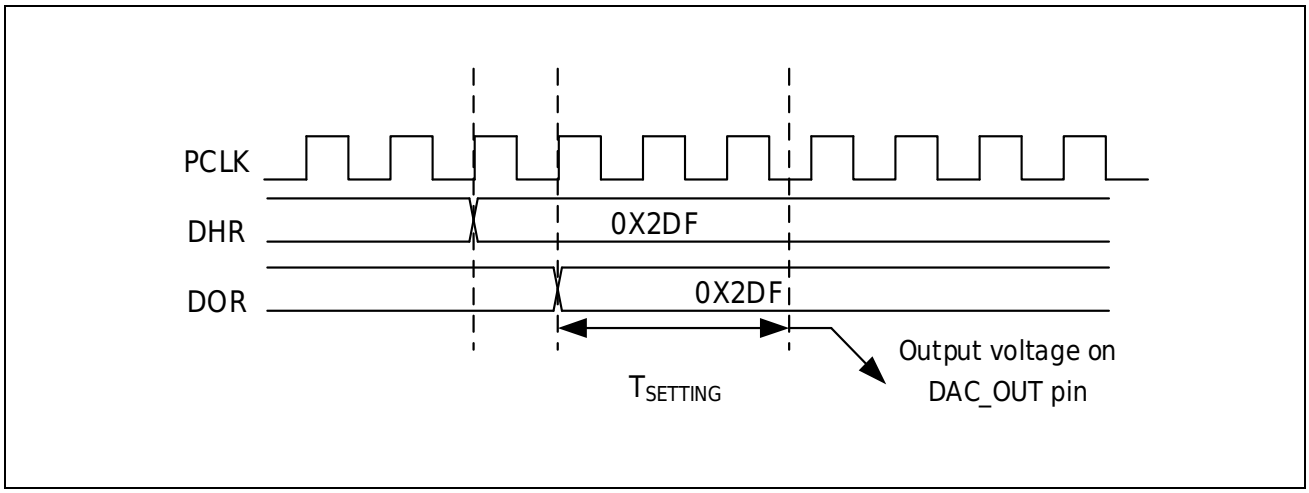


Figure 32-1 Timing diagram of DAC conversion when trigger is disabled

Generate independent triggers for a single LFSR (Linear Feedback Shift Register)

To configure the DAC for this conversion mode (see 32.2.8 Noise Generation), the following sequence needs to be followed:

1. Set the DAC channel trigger enable bit TENx to 1.
2. Configure the trigger source by setting the TSELx[2:0] bits.
3. Configure the WAVEx[1:0] bits of the DAC channel to "01" and configure the same LFSR mask value in the MAMPx[3:0] bits
4. Load the DAC channel data into the desired DAC_DHRx register (DHR12RD, DHR12LD or DHR8RD).

When the DAC channel x trigger arrives, the LFSRx counter content (using the same mask) is added to the DHRx register content and the sum is transferred into DAC_DORx (after three APB clock cycles). LFSRx counters are then updated.

Generate independent triggers for a single triangle wave

To configure the DAC for this conversion mode (see 32.2.8 Noise Generation), the following sequence needs to be followed:

1. Set DAC channel x trigger enable bit TENx to 1.
2. Configure the trigger source by setting the TSELx[2:0] bits.
3. Configure the WAVEx[1:0] bits of DAC channel x to "1x" and configure the same maximum

amplitude value in MAMPx[3:0] bits

4. Load the DAC channel x data into the desired DAC_DHRx register (DHR12RD, DHR12LD or DHR8RD).

When the DAC channel x trigger signal arrives, the DAC channel x triangle wave counter content (using the same triangle wave amplitude) is added to the DHRx register content and the resulting sum is transferred into DAC_DORx (after three APB clock cycles). The DAC channel x triangle wave counter is then updated.

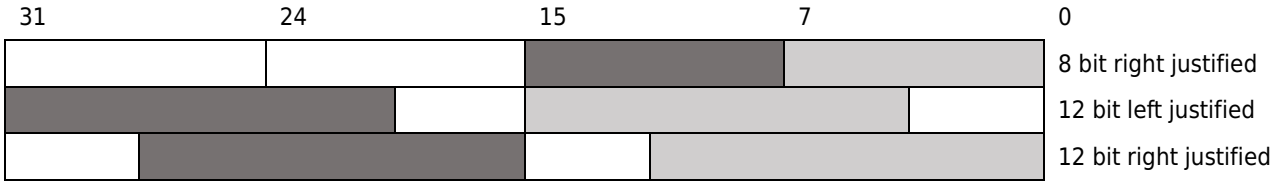
32.2.7 Dual mode function description

32.2.7.1 DAC data format

In DAC dual channel mode, there are three possible ways:

- 8 -bit right alignment: load the data of DAC0 channel to DAC_DHR8RD[7:0] bits (stored in DHR0[11:4] bits), load the data of DAC1 channel into DAC_DHR8RD[15:8] bits (stored in DHR1[11:4] bits)
- 12- bit left alignment: load the data of DAC0 channel to DAC_DHR12RD [15:4] bits (stored to DHR0[11:0] bits), load the data of DAC1 channel to DAC_DHR12RD [31:20] bits (stored to DHR1[11:0] bits)
- 12- bit right alignment: load the data of DAC0 channel to DAC_DHR12RD [11:0] bits (stored to DHR0[11:0] bits), load the data of DAC1 channel to DAC_DHR12RD [27:16] bits (stored to DHR1[11:0] bits)

Data written by the user will be shifted and stored into DHR0 and DHR1 (data holding registers, internal non-memory-mapped registers) according to the loaded DAC_DHRyyyD register. Afterwards, the DHR0 and DHR1 registers will be loaded automatically, or triggered by software or external events into the DOR0 and DOR1 registers respectively.



32.2.7.2 DAC conversion

The DAC channel conversion in dual mode is performed basically the same as in single mode, the data must be loaded by writing to DAC_DHR8Rx, DAC_DHR12Lx, DAC_DHR12Rx, DAC_DHR8RD, DAC_DHR12LD, or DAC_DHR12RD.

32.2.7.3 DAC Double Conversion Mode Description

To efficiently utilize the bus bandwidth in applications requiring two DAC channels simultaneously, the DAC module has three dual registers available for operation: DHR8RD, DHR12RD, and DHR12LD. This allows two DAC channels to be driven simultaneously with only one register access.

Eleven conversion modes are possible through two DAC channels and these three dual registers. But all of these conversion modes can also be implemented through separate DHRx registers if required.

All of these modes are described in the following paragraphs.

Independent triggering (does not generate waveforms)

To configure the DAC for this conversion mode, the following sequence needs to be followed:

1. Set the two DAC channel trigger enable bits TEN0 and TEN1 to 1
2. Set TSEL1[2:0] and TSEL0[2:0] to different values to configure different trigger sources
3. Load DAC dual channel data into desired DHR register (DAC_DHR12RD, DAC_DHR12LD or DAC_DHR8RD)

When the DAC0 channel trigger signal arrives, the content of the DHR0 register is transferred to DAC_DOR9 (after three APB clock cycles).

When the DAC 1 channel trigger signal arrives, the content of the DHR1 register is transferred to DAC_DOR1 (after three APB clock cycles).

Trigger independently (generates a single LFSR)

To configure the DAC for this conversion mode (see 32.2.8 Noise Generation), the following sequence needs to be followed:

1. Set the two DAC channel trigger enable bits TEN1 and TEN0 to 1
2. Set TSEL1[2:0] and TSEL0[2:0] to different values to configure different trigger sources
3. WAVEx[1:0] of both DAC channels to "01" and configure the same LFSR mask value in MAMPx[3:0] bits
4. Load DAC dual channel data into desired DHR register (DHR12RD, DHR12LD or DHR8RD)

When the DAC channel 0 trigger signal arrives, the LFSR0 counter content (using the same mask) is added to the DHR0 register content and the resulting sum is transferred into DAC_DOR0 (after three APB clock cycles). LFSR0 counter is then updated.

When the DAC channel 1 trigger signal arrives, the LFSR1 counter content (using the same mask) is added to the DHR1 register content and the resulting sum is transferred into DAC_DOR1 (after three APB clock cycles). The LFSR1 counter is then updated.

Trigger independently (generate different LFSRs)

To configure the DAC for this conversion mode (see 32.2.8 Noise Generation), the following sequence needs to be followed:

1. Set the two DAC channel trigger enable bits TEN1 and TEN0 to 1
2. Set TSEL1[2:0] and TSEL0[2:0] to different values to configure different trigger sources
3. Set WAVEx[1:0] to "01" for both DAC channels and set different LFSR mask values in

MAMP1[3:0] and MAMP0[3:0] bits

4. Load DAC dual channel data into desired DHR register (DAC_DHR12RD, DAC_DHR12LD or DAC_DHR8RD)

When the DAC channel 0 trigger signal arrives, the LFSR0 counter content (mask configured using MAMP0[3:0]) is added to the DHR0 register content and the resulting sum is transferred into DAC_DOR0 (after three APB clock cycles). LFSR0 counter is then updated.

When the DAC channel 1 trigger signal arrives, the LFSR1 counter content (mask configured using MAMP1[3:0]) is added to the DHR1 register content and the resulting sum is transferred into DAC_DOR1 (after three APB clock cycles). The LFSR1 counter is then updated.

Independent triggering (generates a single triangle wave)

To configure the DAC for this conversion mode (see 32.2.8 Noise Generation), the following sequence needs to be followed:

1. Set DAC channel x trigger enable bit TENx to 1.
2. Configure different trigger sources by setting different values in TSELx [2:0] bits
3. Configure the WAVEx[1:0] bits of DAC channel x to "1x" and configure the same maximum amplitude value in MAMPx[3:0] bits
4. Load DAC channel x data into desired DAC_DHRx register

See Section 15.5.2: DAC Channel Conversions for details on which AHB bus (APB or APB1) to clock the DAC conversions.

When the DAC channel x trigger signal arrives, the DAC channel x triangle wave counter content (using the same triangle wave amplitude) is added to the DHRx register content and the resulting sum is transferred into DAC_DORx (after three APB clock cycles). The DAC channel x triangle wave counter is then updated.

Independent triggering (generation of different triangle waves)

To configure the DAC for this conversion mode (see 32.2.8 Noise Generation), the following sequence needs to be followed:

1. Set DAC channel x trigger enable bit TENx to 1.
2. Configure different trigger sources by setting different values in TSELx [2:0] bits
3. Set the WAVEx[1:0] bits of the DAC channel x to "1x" and set a different maximum amplitude value in the MAMPx[3:0] bits
4. Load DAC channel x data into desired DAC_DHRx register

When the DAC channel x trigger signal arrives, the DAC channel x triangle wave counter content (triangle wave amplitude configured using MAMPx[3:0]) is added to the DHRx register content, and the resulting sum is transferred to DAC_DORx (after three APB clock cycles). The DAC channel x triangle wave counter is then updated.

Synchronization software starts

To configure the DAC for this conversion mode, the following sequence needs to be followed:

1. Load DAC dual channel data into desired DHR register (DAC_DHR12RD, DAC_DHR12LD or DAC_DHR8RD)

This configuration uses one APB clock cycle.

Synchronous triggering (does not generate waveforms)

To configure the DAC for this conversion mode, the following sequence needs to be followed:

1. Set the two DAC channel trigger enable bits TEN1 and TEN0 to 1
2. Set TSEL1[2:0] and TSEL0[2:0] to the same value to configure the same trigger source for both DAC channels
3. Load DAC dual channel data into desired DHR register (DAC_DHR12RD, DAC_DHR12LD or DAC_DHR8RD)

When the trigger signal arrives, the DHR1 and DHR0 register contents will be transferred to DAC_DOR1 and DAC_DOR0 respectively (after three APB clock cycles).

Synchronous triggering (generates a single LFSR)

To configure the DAC for this conversion mode (see 32.2.8 Noise Generation), the following sequence needs to be followed:

1. Set the two DAC channel trigger enable bits TEN1 and TEN0 to 1
2. Set TSEL1[2:0] and TSEL2[2:0] to the same value to configure the same trigger source for both DAC channels
3. WAVEx[1:0] of both DAC channels to "01" and configure the same LFSR mask value in MAMPx[3:0] bits
4. Load the DAC dual channel data into the desired DHR register (DHR12RD, DHR12LD or DHR8RD) When the trigger signal arrives, the LFSR1 counter content (using the same mask) is added to the DHR1 register content and the resulting sum is transferred to DAC_DOR1 (three APB clock cycle later). The LFSR1 counter is then updated. At the same time, the LFSR0 counter content (using the same mask) is added to the DHR0 register content, and the resulting sum is transferred into DAC_DOR0 (after three APB clock cycles). LFSR0 counter is then updated.

Synchronous triggering (generating different LFSRs)

To configure the DAC for this conversion mode (see 32.2.8 Noise Generation), the following sequence needs to be followed:

1. Set the two DAC channel trigger enable bits TEN1 and TEN0 to 1
2. Set TSEL1[2:0] and TSEL0[2:0] to the same value to configure the same trigger source for both DAC channels

3. WAVEx[1:0] of both DAC channels to "01" and set different LFSR mask values in MAMP1[3:0] and MAMP2[3:0] bits
4. Load the DAC dual channel data into the desired DHR register (DAC_DHR12RD, DAC_DHR12LD or DAC_DHR8RD) When the trigger signal arrives, the LFSR1 counter content (mask configured using MAMP1[3:0]) is added to the DHR1 register content and the resulting sum is transferred to DAC_DOR1 (after three APB clock cycles). The LFSR1 counter is then updated.

   At the same time, the LFSR0 counter content (mask configured using MAMP0[3:0]) is added to the DHR0 register content and the resulting sum is transferred into DAC_DOR0 (after three APB clock cycles). LFSR0 counter is then updated.

Synchronous triggering (generates a single triangle wave)

To configure the DAC for this conversion mode (see 32.2.8 Noise Generation), the following sequence needs to be followed:

1. Set DAC channel x trigger enable bit TEN1x to 1.
2. Set the same value in the TSELx [2:0] bits to configure the same trigger source for both DAC channels.
3. Configure the WAVEx[1:0] bits of DAC channel x to "1x" and use the MAMPx[3:0] bits to configure the same maximum amplitude value
4. Load DAC channel x data into desired DAC_DHRx register.

When the trigger signal arrives, the DAC channel x triangle wave counter content (using the same triangle wave amplitude) is added to the DHRx register content and the resulting sum is transferred into DAC_DORx (after three APB clock cycles). The DAC channel x triangle wave counter is then updated.

Synchronous trigger (generate different triangle waves)

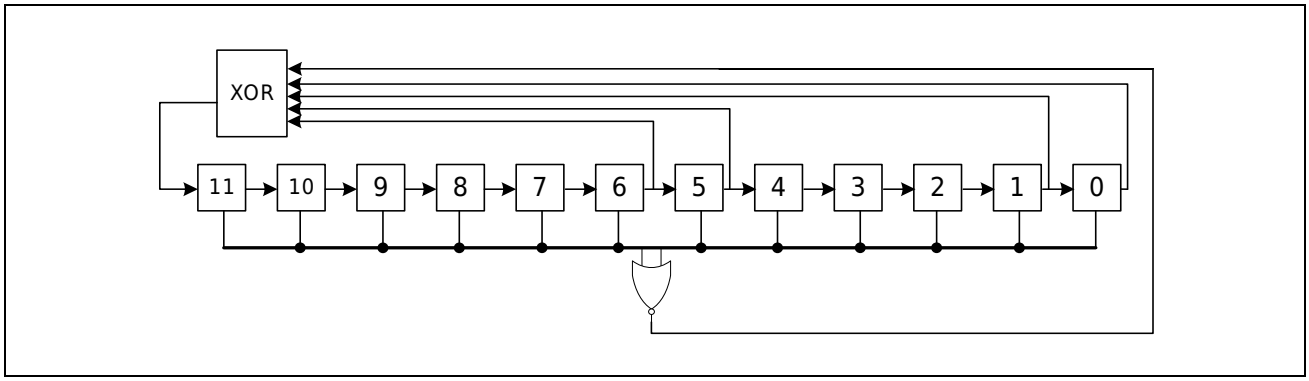
To configure the DAC for this conversion mode (see 32.2.8 Noise Generation), the following sequence needs to be followed:

1. Set DAC channel x trigger enable bit TENx to 1.
2. Set the same value in the TSELx [2:0] bits to configure the same trigger source for DAC channel x.
3. Configure the WAVEx[1:0] bits of the DAC channel x as "1x" and set a different maximum amplitude value in the MAMPx[3:0] bits.
4. Load DAC channel x data into desired DAC_DHRx register.

When the trigger signal arrives, the DAC channel x triangle wave counter content (triangle wave amplitude configured using MAMPx[3:0]) is added to the DHRx register content, and the resulting sum is transferred into DAC_DORx (after three APB clock cycles). The DAC channel x triangle wave counter is then updated.

32.2.8 Noise generation

To generate pseudo noise of variable amplitude, LFSRs (Linear Feedback Shift Registers) are used. Set WAVEx[1:0] to "01" to choose to generate noise. The preload value in LFSR is 0xAAA. After each trigger event, after three APB clock cycles, the register will be updated according to a specific calculation algorithm.



The LFSR value can be partially or completely masked by the MAMPx[3:0] bits in the DAC_CR register. In the case of no overflow, the value will be added to the content of DAC_DHRx and then stored in the DAC_DORx register.

If LFSR is 0x0000, "1" will be injected into it (anti-lock mechanism).

The LFSR waveform generation function can be disabled by resetting the WAVEx[1:0] bits.

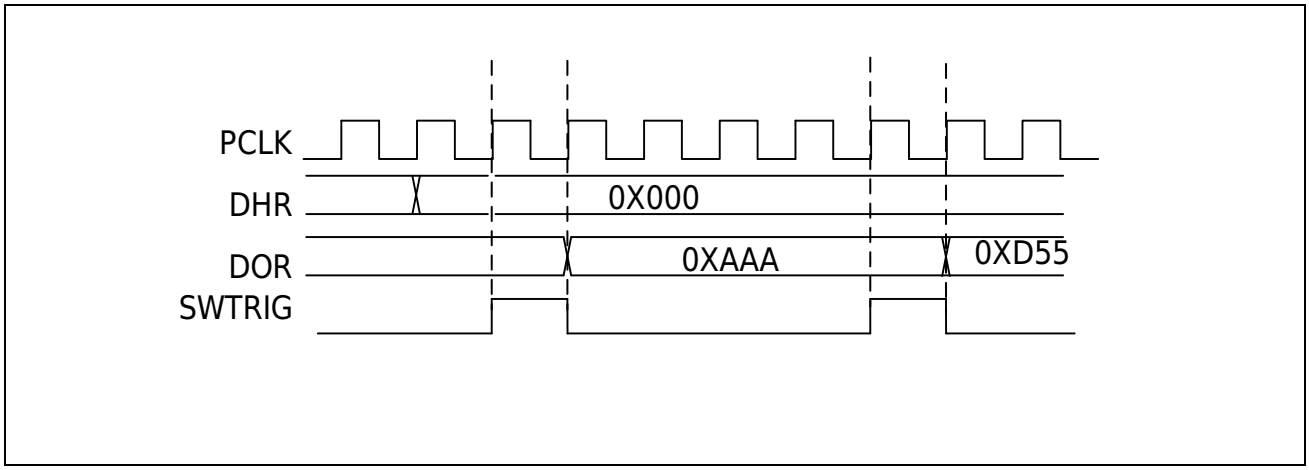


Figure 32-2 DAC conversion of waveform generated by LFSR (using software trigger)

32.2.9 Triangle wave generation

A small-amplitude triangle wave can be superimposed on a DC current or slowly varying signal. Set WAVEx[1:0] to "10" to select DAC to generate triangle wave. The amplitude is configured by the MAMPx[3:0] bits in the DAC_CR register. After each trigger event, after three APB clock cycles, the internal triangle wave counter will be incremented. In case overflow does not occur, the value of this counter is added to the contents of the DAC_DHRx register and the sum is stored in the DAC_DORx register. As long as it is less than the maximum amplitude defined by the MAMPx[3:0]

bits, the triangle wave counter will keep incrementing. Once the configured amplitude is reached, the counter is decremented to zero, then incremented, and so on.

The triangle wave generation function can be turned off by resetting the WAVEx[1:0] bits.

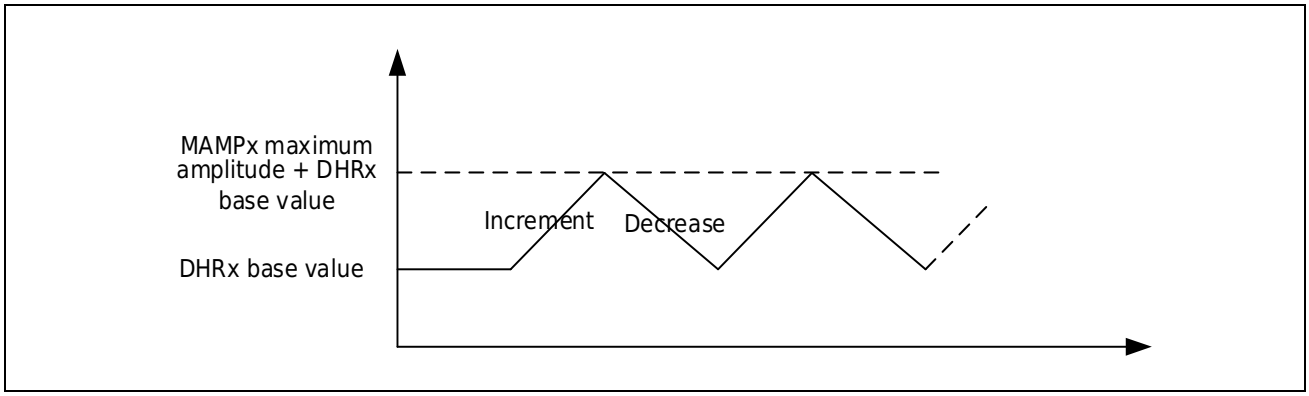


Figure 32-3 Generate DAC triangle wave

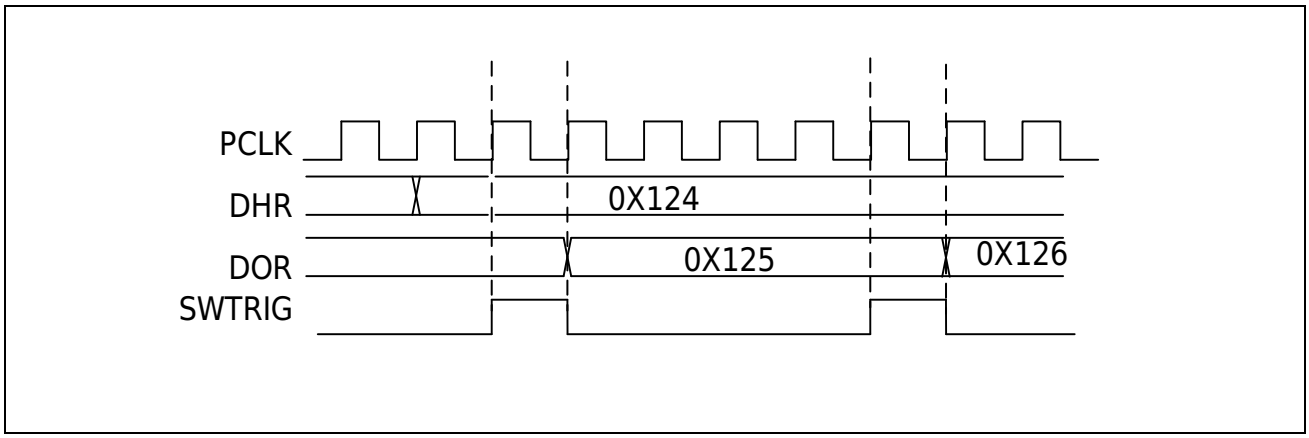


Figure 32-4 DAC conversion to generate triangle waveform (software trigger enabled)

32.2.10 DMA request

Each DAC channel has DMA capability. Two DMA channels are used to handle DMA requests from the DAC channel.

When the DMAENx bit is set, a DAC DMA request will be generated if a trigger occurs. DAC_DHRx register is then transferred to the DAC_DORx register.

In dual channel mode, if both DMAENx bits are set, two DMA requests will be generated. The user should only set the corresponding DMAENx bit if only one DMA request is required. This way the application can manage two DAC channels in dual channel mode with one DMA request and one specific DMA channel.

DMA underflow

There is no buffer queue for DAC DMA requests. In this way, if the acknowledgment of the first external trigger has not been received when the second external trigger arrives, no new request will be issued and the DAM channel underflow flag DMAUDRx in the DAC_SR register will be set to 1 to report this error condition. DMA data transfers are then disabled and no other DMA requests are processed. DAC channel will still continue to convert legacy data.

DMAUDRx flag by writing '0', clear the DMAEN bit for the DMA data stream being used, and reinitialize the DMA and DAC channels to properly restart the DMA transfer. Software should modify the frequency of DAC trigger conversions or reduce the DMA workload to avoid DMA underflow from happening again. DAC conversions can be continued by enabling DMA data transfers and conversion triggers.

For each DAC channel, an interrupt will also be generated if the corresponding DMAUDRIEx bit in the DAC_CR register is enabled.

32.3DAC register

Base address 0x40002500

Table 32-1 DAC register

Register	Offset address	Description
DAC_CR0	0x000	control register
DAC_SWTRIGR	0x004	Software trigger register
DAC_DHR12R0	0x008	DAC0 12-bit right-aligned data register
DAC_DHR12L0	0x00C	DAC0 12 -bit left justified data register
DAC_DHR8R0	0X010	DAC0 8-bit right-aligned data register
DAC_DHR12R1	0x014	DAC1 12-bit right-aligned data register
DAC_DHR12L1	0x018	DAC1 12 -bit left justified data register
DAC_DHR8R1	0x01C	DAC1 8-bit right-aligned data register
DAC_DHR12RD	0X020	DAC0/1 12-bit right-aligned data register
DAC_DHR12LD	0x024	DAC0/1 12- bit left justified data register
DAC_DHR8RD	0x028	DAC0/1 8-bit right-aligned data register
DAC_DOR0	0x02C	DAC0 data output register
DAC_DOR1	0X030	DAC1 data output register
DAC_SR	0X034	Status register
DAC_ETRS	0X038	DAC Test Register / External Trigger Select Register

32.3.1 DAC Control Register (DAC_CR0)

Offset address 0x00

Reset value 0xC000 C000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SREF1		DMAUDRIE1	DMAEN1	MAMP1				WAVE1		TSEL1			TEN1	BOFF1	EN1
RW		RW	RW	RW				RW		RW			RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SREF0		DMAUDRIE0	DMAEN0	MAMP0				WAVE0		TSEL0			TEN0	BOFF0	EN0
RW		RW	RW	RW				RW		RW			RW	RW	RW

Bit	Marking	Functional description
31:30	SREF1	DAC1 reference voltage selection 00: Internal 1.5V 01: Internal 2.5V 10: External reference voltage ExRef (PB01) 11: AVCC voltage
29	DMAUDRIE1	DAC1 DMA underflow interrupt enable 0: Disable DAC1 channel DMA underflow interrupt; 1: Enable DAC1 channel DMA underflow interrupt
28	DMAEN1	DAC1 channel DMA enable 0: disable DAC1 channel DMA mode 1: Enable DAC1 channel DMA mode
27:24	MAMP1	DAC1 channel mask / amplitude selector 0000: Do not mask bit 0 of LFSR / triangle wave amplitude equal to 1 0001: Bits [1:0] of LFSR are not masked / triangle wave amplitude is equal to 3 0010: Bits[2:0] of LFSR not masked / triangle wave amplitude equal to 7 0011: Bits[3:0] of LFSR not masked / triangle wave amplitude equal to 15 0100: Bits[4:0] of LFSR not masked / triangle wave amplitude equal to 31 0101: Bits[5:0] of LFSR not masked / triangle wave amplitude equal to 63 0110: Bits[6:0] of LFSR not masked / triangle wave amplitude equal to 127 0111: Bits[7:0] of LFSR not masked / triangle wave amplitude equal to 255 1000: Bits[8:0] of LFSR not masked / triangle wave amplitude equal to 511 1001: Bits[9:0] of LFSR not masked / triangle wave amplitude equal to 1023 1010: Bits[10:0] of LFSR not masked / triangle wave amplitude equal to 2047 ≥1011: Bits[11:0] of LFSR are not masked / triangular wave amplitude is equal to 4095
23:22	WAVE1	DAC1 channel noise / triangular wave generation enable 00: disable wave generation 01: Enable generating noise waves 1x enable generation of triangle wave
21:19	TSEL1	DAC1 channel trigger selection 000 TIM0 TRADC trigger 001 TIM1 TRADC trigger 010 TIM2 TRADC trigger 011 TIM3 TRADC trigger 100 TIM4 TRADC trigger 101 TIM5 TRADC trigger 110 software trigger 111 external port trigger
18	TEN1	DAC1 channel trigger enable 0: DAC1 channel trigger is disabled, the data written to the DAC_DHR1 register is transferred to the DAC_DOR1 register after one APB1 clock cycle 1: Enable DAC1 channel trigger, the data of DAC_DHR1 register is transferred to DAC_DOR1 register after three APB1 clock cycles
17	BOFF1	DAC1 channel output buffer disable control, enable buffer BUFFEN in OPA control register 0: Enable DAC1 channel output buffer, DAC1 through buffer to port 1: Disable DAC1 channel output buffer, DAC1 output to port To use the buffer as an output port, the BUFEN function of OPA4 needs to be enabled, and OPA4 cannot be enabled
16	EN1	DAC1 channel enable 0: Disable DAC1 channel 1: Enable DAC1 channel

15:14	SREF0	DAC0 reference voltage selection 00: Internal 1.5V 01: Internal 2.5V 10: External reference voltage ExRef (PB01) 11: AVCC voltage
13	DMAUDRIE0	DAC0 DMA underflow interrupt enable 0: Disable DAC0 channel DMA underflow interrupt; 1: Enable DAC0 channel DMA underflow interrupt
12	DMAEN0	DAC0 channel DMA enable 0: disable DAC0 channel DMA mode 1: Enable DAC0 channel DMA mode
11:8	MAMP0	DAC0 channel mask / amplitude selector 0000: Do not mask bit 0 of LFSR / triangle wave amplitude equal to 1 0001: Bits [1:0] of LFSR are not masked / triangle wave amplitude is equal to 3 0010: Bits[2:0] of LFSR not masked / triangle wave amplitude equal to 7 0011: Bits[3:0] of LFSR not masked / triangle wave amplitude equal to 15 0100: Bits[4:0] of LFSR not masked / triangle wave amplitude equal to 31 0101: Bits[5:0] of LFSR not masked / triangle wave amplitude equal to 63 0110: Bits[6:0] of LFSR not masked / triangle wave amplitude equal to 127 0111: Bits[7:0] of LFSR not masked / triangle wave amplitude equal to 255 1000: Bits[8:0] of LFSR not masked / triangle wave amplitude equal to 511 1001: Bits[9:0] of LFSR not masked / triangle wave amplitude equal to 1023 1010: Bits[10:0] of LFSR not masked / triangle wave amplitude equal to 2047 ≥1011: Bits[11:0] of LFSR are not masked / triangular wave amplitude is equal to 4095
7:6	WAVE0	DAC0 channel noise / triangular wave generation enable 00: disable wave generation 01: Enable generating noise waves 1x enable generation of triangle wave
5:3	TSEL0	DAC0 channel trigger selection 000 TIM0 TRADC trigger 001 TIM1 TRADC trigger 010 TIM2 TRADC trigger 011 TIM3 TRADC trigger 100 TIM4 TRADC trigger 101 TIM5 TRADC trigger 110 software trigger 111 external port trigger
2	TEN0	DAC0 channel trigger enable 0: DAC0 channel trigger is disabled, the data written to the DAC_DHR0 register is transferred to the DAC_DOR0 register after one APB1 clock cycle 1: DAC0 channel trigger is enabled, the data in the DAC_DHR0 register is transferred to the DAC_DOR0 register after three APB1 clock cycles
1	BOFF0	DAC0 channel output buffer disable control, enable buffer BUFFEN in OPA control register 0: Enable DAC0 channel output buffer, DAC0 through buffer to port 1: Disable DAC0 channel output buffer, DAC0 output to port To use the buffer as an output port, the BUFEN function of OPA3 needs to be enabled, and OPA3 cannot be enabled
0	EN0	DAC0 channel enable 0: Disable DAC0 channel 1: Enable DAC0 channel

32.3.2 DAC Software Trigger Register (DAC_SWTRIGR)

Offset address 0x04

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														SWTRIG1	SWTRIG0
														W1	W1

Bit	Marking	Functional description
31:2	Reserved	Keep
1	SWTRIG1	DAC1 software trigger Write 1 software trigger, auto clear, write 0 no action
0	SWTRIG0	DAC0 software trigger Write 1 software trigger, auto clear, write 0 no action

32.3.3 DAC0 12- bit Right Justified Data Holding Register (DAC_DHR12R0)

Offset address 0x08

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				DHR0											
				RW											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	DHR0	DAC channel 0 12-bit right-justified data These bits are written by software to specify 12 -bit data for DAC Channel 0.

32.3.4 DAC0 12- bit Left Justified Data Holding Register (DAC_DHR12L0)

Offset address 0x0C

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DHR0												Reserved			
RW															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:4	DHR0	DAC channel 0 12-bit left justified data These bits are written by software to specify 12 -bit data for DAC Channel 0.
3:0	Reserved	Keep

32.3.5 DAC0 8- bit Right Justified Data Holding Register (DAC_DHR8R0)

Offset address 0x10

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DHR0							
								RW							

Bit	Marking	Functional description
31:8	Reserved	Keep
7:0	DHR0	DAC channel 0 8-bit right-justified data These bits are written by software to specify 8 -bit data for DAC Channel 0. DHR are written.

32.3.6 DAC1 12- bit right-justified data holding register (DAC_DHR12R1)

Offset address 0x14

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				DHR1											
				RW											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	DHR1	DAC channel 1 12-bit right-justified data These bits are written by software to specify 12 -bit data for DAC Channel 1.

32.3.7 DAC1 12- bit Left Justified Data Holding Register (DAC_DHR12L1)

Offset address 0x18

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
DHR1												Reserved			
RW															

Bit	Marking	Functional description
31:16	Reserved	Keep
15:4	DHR1	DAC channel 1 12-bit left justified data These bits are written by software to specify 12 -bit data for DAC Channel 1.
3:0	Reserved	Keep

32.3.8 DAC1 8- bit right-justified data holding register (DAC_DHR8R1)

Offset address 0x1C

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DHR1							
								RW							

Bit	Marking	Functional description
31:8	Reserved	Keep
7:0	DHR1	DAC channel 1 8-bit right-justified data These bits are written by software to specify 8 -bit data for DAC Channel 1. DHR are written.

32.3.9 Dual DAC 12- bit Right Justified Data Holding Register (DAC_DHR12RD)

Offset address 0x20

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				DHR1											
				RW											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				DHR0											
				RW											

Bit	Marking	Functional description
31:28	Reserved	Keep
27:16	DHR1	DAC channel 1 12-bit left justified data These bits are written by software to specify 12 -bit data for DAC Channel 1.
15:12	Reserved	Keep
11:0	DHR0	DAC channel 0 12-bit right-justified data These bits are written by software to specify 12 -bit data for DAC Channel 0.

32.3.10 Dual DAC 12 -bit Left Justified Data Holding Register (DAC_DHR12LD)

Offset address 0x24

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DHR1												Reserved			
RW															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DHR0												Reserved			
RW															

Bit	Marking	Functional description
31:20	DHR1	DAC channel 1 12-bit left justified data These bits are written by software to specify 12 -bit data for DAC Channel 1.
19:16	Reserved	Keep
15:4	DHR0	DAC channel 0 12-bit left justified data These bits are written by software to specify 12 -bit data for DAC Channel 0.
3:0	Reserved	Keep

32.3.11 Dual DAC 8 -bit Right Justified Data Holding Register (DAC_DHR8RD)

Offset address 0x28

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
DHR1								DHR0							
RW								RW							

Bit	Marking	Functional description
31:16	Reserved	Keep
15:8	DHR1	DAC channel 1 8-bit right-justified data These bits are written by software to specify 8 -bit data for DAC Channel 0. DHR are written.
7:0	DHR0	DAC channel 0 8-bit right-justified data These bits are written by software to specify 8 -bit data for DAC Channel 0. DHR are written.

32.3.12 DAC0 Data Output Register (DAC_DOR0)

Offset address 0x2C

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				DOR0											
				RO											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	DOR0	DAC 0 channel data output DAC 0 channel

32.3.13 DAC1 Data Output Register (DAC_DOR1)

Offset address 0x30

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				DOR1											
				RO											

Bit	Marking	Functional description
31:12	Reserved	Keep
11:0	DOR1	DAC 1 channel data output DAC 1 channel

32.3.14 DAC Status Register (DAC_SR)

Offset address 0x34

Reset value 0x00004000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DMAUDR1	Reserved												
		RW													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DMAUDR0	Reserved												
		RW													

Bit	Marking	Functional description
31:30	Reserved	Keep
29	DMAUDR1	DAC 1 channel DMA underflow flag 0: No DMA underflow error condition occurred on DAC 1 channel 1: DMA underflow error condition occurred on DAC 1 channel (currently selected trigger source drives DAC 1 channel conversion at a frequency higher than DMA service capability)
28:14	Reserved	Keep
13	DMAUDR0	DAC 0 channel DMA underflow flag 0: No DMA underflow error condition occurred on DAC0 channel 1: DMA underflow error condition occurred on DAC 0 channel (currently selected trigger source drives DAC 0 channel conversion at a frequency higher than DMA service capability)
12:0	Reserved	Keep

32.3.15 DAC Trigger Select Register (DAC_ETRS)

Offset address 0x038

Reset value 0x00000000

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

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

Bit	Marking	Functional description
31:7	Reserved	Reserved bit
6:4	PTRIGSEL	Port trigger selection 000 PA9 001 PB9 010 PC9 011 PD9 100 PE9 101 PF9 11X reserved
3:0	RSV	Reserved bit

33 Analog Comparator (VC)

33.1 Introduction to Analog Voltage Comparator VC

The analog voltage comparator VC is used to compare the size of two input analog voltages, and output high / low level according to the comparison result. When the "+" input voltage is higher than the "-" input voltage, the voltage comparator output is high; when the "+" input voltage is lower than the -" input voltage, the voltage comparator output is low flat. The analog voltage comparator VC integrated in this product has the following characteristics:

- Support voltage comparison function;
- Support internal 64- stage VCC voltage divider (use the voltage divider source voltage to be greater than 1.8V);
- Support 16 external input ports as the input of the voltage comparator;
- Support three software-configurable interrupt trigger modes: high level trigger / rising edge trigger / falling edge trigger;
- The output of the voltage comparator can be used as General purpose timer and Low-power timer control input;
- The output of the voltage comparator can be used as Brake input or capture input for advanced timers and general-purpose timers;
- Support working in ultra-low power consumption mode, the interrupt output of the voltage comparator can wake up the chip from ultra-low power consumption mode;
- Provide software configurable filter time to enhance the anti-interference ability of the chip.

Note: BGR needs to be enabled when using VC, refer to BGR register description.

33.2Voltage Comparator Block Diagram

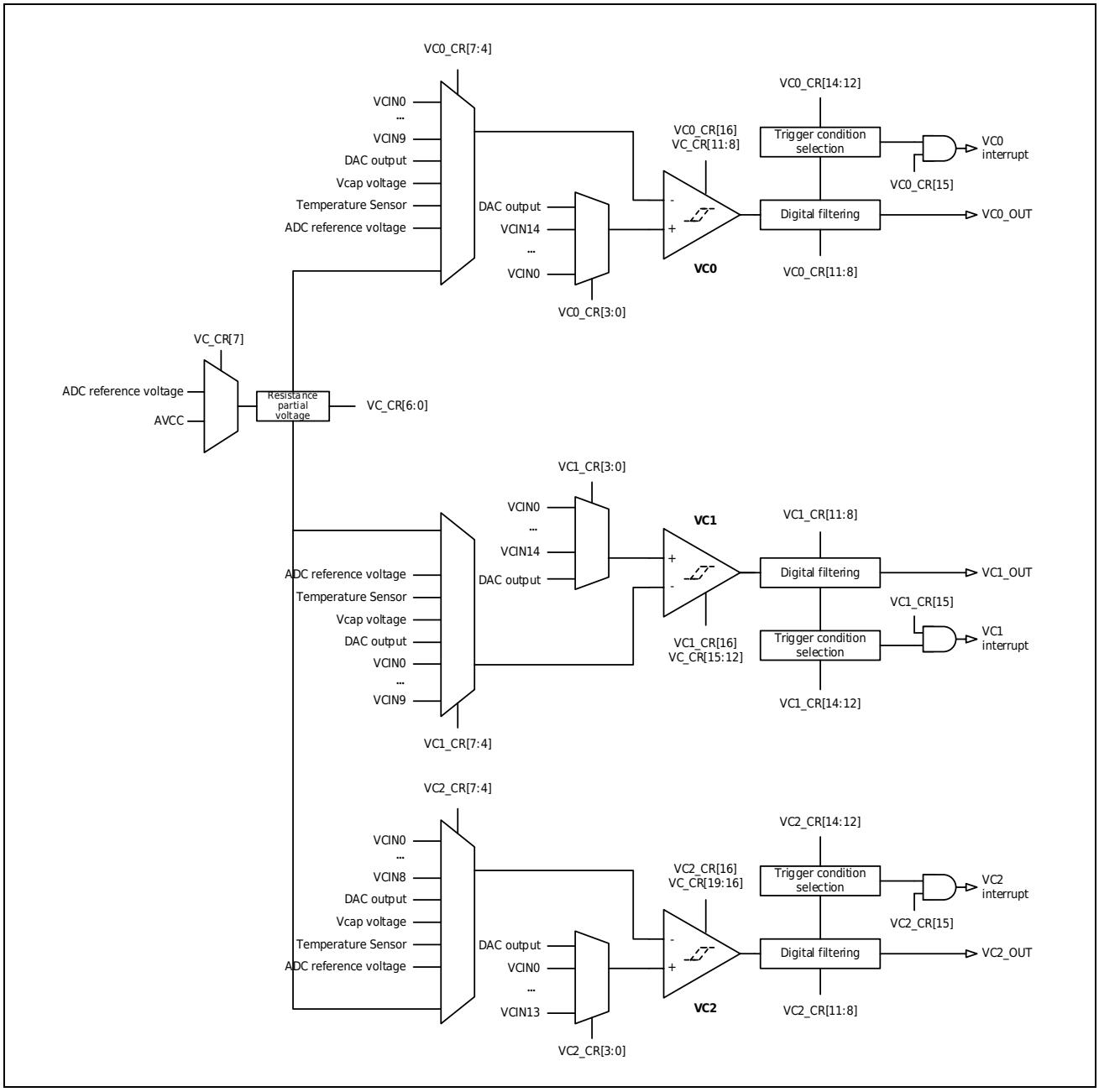


Figure 33-1 VC Frame Diagram

33.3Setup / Response Time

When using a voltage comparator, the time from when VC is enabled or when the input voltage at both ends of VC changes to when the correct result is output is determined by the BIAS_SEL control bit in the VC control register (VC_CR). The larger the current, the faster the VC response, typical value Four levels are adjustable from 200ns to 20us.

If needed, the internal BGR provides a reference voltage (1.5V or 2.5V) as a comparison reference voltage. The start-up time of the internal BGR is about 20us, and the voltage comparator needs to wait for the internal BGR to stabilize before outputting normally.

33.4Filter time

In addition to the inherent settling/response time of the voltage comparator, the user can set a longer filter time to filter out system noise, such as high current noise when the motor stops.

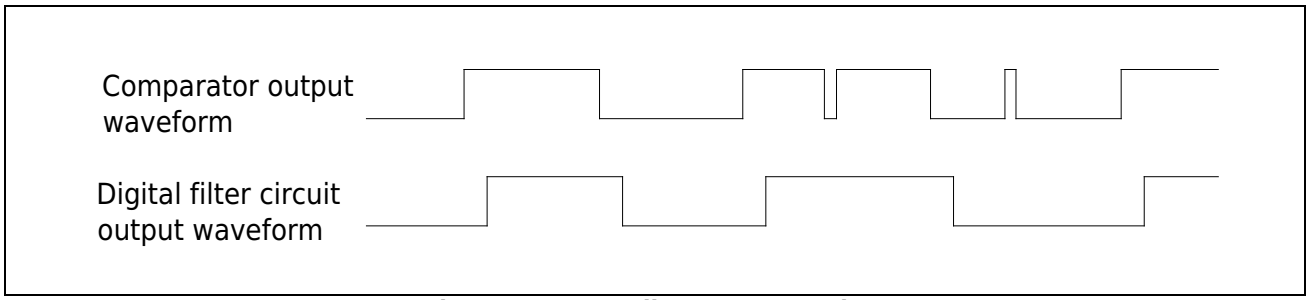


Figure 33-2 VC Filter Response Time

33.5Hysteresis function

The hysteresis function can be selected for the voltage comparator, and the diagram after the hysteresis function is enabled is as follows:

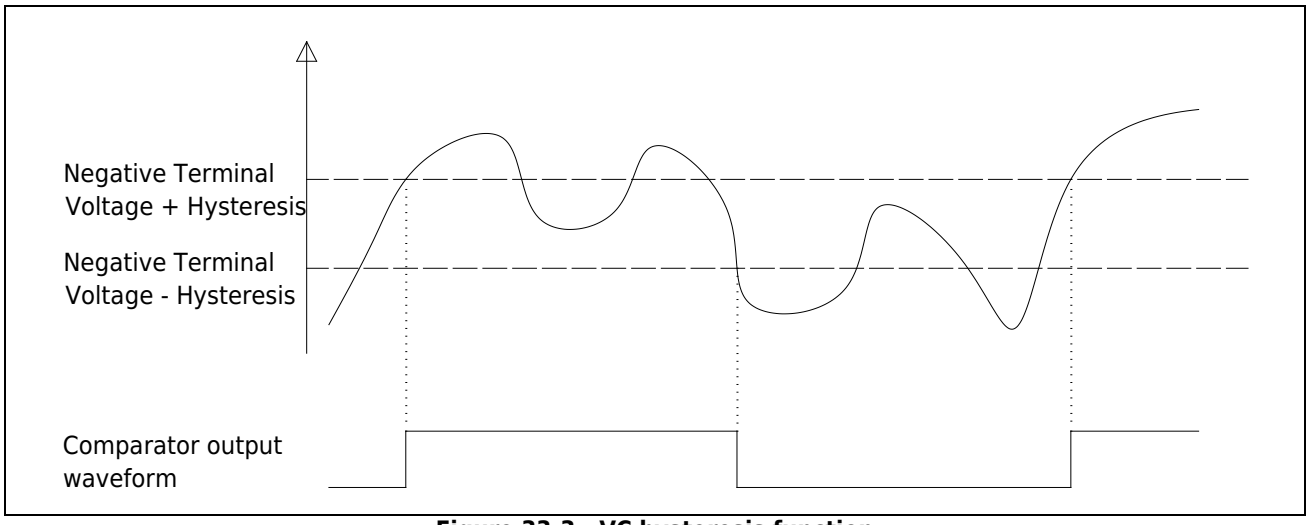


Figure 33-3 VC hysteresis function

33.6VC register

Base address 0x40002400

Table 33-1 VC register

Register	Offset address	Description
VC_CR	0x010	VC0/1 Configuration Register 0
VC0_CR	0x014	VC0 configuration register
VC1_CR	0x018	VC1 configuration register
VC0_ OUT_CFG	0x01C	VC0 output configuration register
VC1_ OUT_CFG	0x020	VC1 Output Configuration Register
VC_IFR	0x024	VC interrupt register
VC2_CR	0x150	VC2 configuration register
VC2_ OUT_CFG	0x154	VC2 Output Configuration Register

33.6.1 VC Configuration Register (VC_CR)

Offset address 0x010

Reset value 0x00000020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												VC2_HYS_SEL		VC2_BIAS_SEL	
												R/W		R/W	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VC1_HYS_SEL		VC1_BIAS_SEL		VC0_HYS_SEL		VC0_BIAS_SEL		VC_REF2P5_SEL	VC_DIV_EN	VC_DIV					
R/W		R/W		R/W		R/W		R/W	R/W	R/W					

Bit	Marking	Functional description
31:16	Reserved	Keep
19:18	VC2_HYS_SEL	VC2 hysteresis selection: 00: no hysteresis 01: Hysteresis voltage about 10mV 10: The hysteresis voltage is about 20mV 11: The hysteresis voltage is about 30mV
17:16	VC2_BIAS_SEL	VC2 power consumption selection (the greater the power consumption, the faster the response speed) 00:300nA 01:1.2uA 10:10uA (BGR needs to be turned on, BGR startup time is about 20us) 11:20uA (BGR needs to be turned on, BGR startup time is about 20us)
15:14	VC1_HYS_SEL	VC1 hysteresis selection: 00: no hysteresis 01: Hysteresis voltage about 10mV 10: The hysteresis voltage is about 20mV 11: The hysteresis voltage is about 30mV
13:12	VC1_BIAS_SEL	VC1 power consumption selection (the greater the power consumption, the faster the response speed) 00:300nA 01:1.2uA 10:10uA (BGR needs to be turned on, BGR startup time is about 20us) 11:20uA (BGR needs to be turned on, BGR startup time is about 20us)
11:10	VC0_HYS_SEL	VC0 hysteresis selection: 00: no hysteresis 01: Hysteresis voltage about 10mV 10: The hysteresis voltage is about 20mV 11: The hysteresis voltage is about 30mV
9:8	VC0_BIAS_SEL	VC0 power consumption selection (the greater the power consumption, the faster the response speed) 00:300nA 01:1.2uA 10:10uA (BGR needs to be turned on, BGR startup time is about 20us) 11:20uA (BGR needs to be turned on, BGR startup time is about 20us)
7	VC_REF2P5_SEL	VC_DIV reference voltage Vref selection 0:VCC 1: The reference voltage selected by ADC_CR0.SREF
6	VC_DIV_EN	6- bit DAC enable 1: Enable
5:0	VC_DIV	6 -bit DAC configuration 000000: 1/64 Vref 000001:2/64 Vref 000010:3/64 Vref 000011:4/64 Vref … 111110:63/64 Vref 111111: Vref

33.6.2 VC0 Configuration Register (VC0_CR)

Offset address 0x014

Reset value 0x00000000

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

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			
RW	RW	RW	RW	RW			RW	RW				RW			

Bit	Marking	Functional description
31:17	Reserved	Keep
16	EN	Comparator Enable 1: enable voltage comparator 0: disable voltage comparator
15	IE	VC interrupt enable 1: enable; 0: disable
14	level	VC output signal triggers interrupt selection 1: enable high level trigger INT flag 0: disable high level triggering INT flag
13	rising	VC output signal triggers interrupt selection 1: enable rising edge to trigger INT flag 0: disable rising edge triggering INT flag
12	falling	VC output signal triggers interrupt selection 1: enable falling edge trigger INT flag 0: disable falling edge triggering INT flag
11:9	debounce_time	VC output filter time configuration 111: The filtering time is about 28.8ms 110: The filtering time is about 7.2ms 101: The filtering time is about 1.8ms 100: The filtering time is about 450us 011: The filtering time is about 112us 010: The filtering time is about 28us 001: The filtering time is about 14us 000: The filtering time is about 7us Note: The filter time configuration is only valid when FLTEN=1.
8	FLTEN	1: start VC filter 0: VC without filter
7:4	N_SEL	Voltage comparator "-" terminal input selection 0000: select channel 0 input PA0 0001: select channel 1 input PA1 0010: select channel 2 input PA2 0011: select channel 3 input PA3 0100: select channel 4 input PA4 0101: select channel 5 input PA5 0110: select channel 6 input PA6 0111: select channel 7 input PA7 1000: select channel 8 input PC4 1001: select channel 9 input PC5 1010:DAC0 1011: Resistor divider output voltage 1100: built-in temperature sensor output voltage 1101: Reserved 1110: REF of ADC (ADC needs to be enabled for use) 1111: VCAP output voltage
3:0	P_SEL	Voltage comparator "+" Terminal input selection 0000: select channel 0 input PC0 0001: select channel 1 input PC1 0010: select channel 2 input PC2 0011: select channel 3 input PC3 0100: select channel 4 input PA0 0101: select channel 5 input PA1 0110: select channel 6 input PA2

		0111: select channel 7 input PA3 1000: select channel 8 input PA4 1001: select channel 9 input PA5 1010: select channel 10 input PA6 1011: select channel 11 input PA7 1100: select channel 12 input PB4 1101: select channel 13 input PB5 1110: select channel 14 input PB6 1111: DAC0

33.6.3 VC1 Configuration Register (VC1_CR)

Offset address 0x018

Reset value 0x00000000

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

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			
RW	RW	RW	RW	RW			RW	RW				RW			

Bit	Marking	Functional description
31:17	Reserved	Keep
16	EN	Comparator Enable 1: enable voltage comparator 0: disable voltage comparator
15	IE	VC interrupt enable 1: enable; 0: disable
14	level	VC output signal triggers interrupt selection 1: enable high level trigger INT flag 0: disable high level triggering INT flag
13	rising	VC output signal triggers interrupt selection 1: enable rising edge to trigger INT flag 0: disable rising edge triggering INT flag
12	falling	VC output signal triggers interrupt selection 1: enable falling edge trigger INT flag 0: disable falling edge triggering INT flag
11:9	debounce_time	VC output filter time configuration 111: The filtering time is about 28.8ms 110: The filtering time is about 7.2ms 101: The filtering time is about 1.8ms 100: The filtering time is about 450us 011: The filtering time is about 112us 010: The filtering time is about 28us 001: The filtering time is about 14us 000: The filtering time is about 7us Note: The filter time configuration is only valid when FLTEN=1.
8	FLTEN	1: start VC filter 0: VC without filter
7:4	N_SEL	Voltage comparator "-" terminal input selection 0000: select channel 0 input PC0 0001: select channel 1 input PC1 0010: select channel 2 input PC2 0011: select channel 3 input PC3 0100: select channel 4 input PA0 0101: select channel 5 input PA1 0110: select channel 6 input PB0 0111: select channel 7 input PB1 1000: select channel 8 input PB2 1001: select channel 9 input PB3 1010: DAC1 1011: Resistor divider output voltage 1100: built-in temperature sensor output voltage 1101: Reserved 1110: REF of ADC (ADC needs to be enabled for use) 1111: VCAP pin voltage
3:0	P_SEL	Voltage comparator "+" Terminal input selection 0000: select channel 0 input PA0 0001: select channel 1 input PA1 0010: select channel 2 input PA2 0011: select channel 3 input PA3 0100: select channel 4 input PA4 0101: select channel 5 input PA5 0110: select channel 6 input PB1

		0111: select channel 7 input PB2 1000: select channel 8 input PB10 1001: select channel 9 input PB12 1010: select channel 10 input PB13 1011: select channel 11 input PB14 1100: select channel 12 input PB4 1101: DAC1 1110: select channel 14 input PB6 1111: select channel 15 input PB7

33.6.4 VC0 Output Configuration Register (VC0_OUT_CFG)

Offset address 0x01C

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
brake	TIM6	INV_TIM6	TIM5	INV_TIM5	TIM4	INV_TIM4	Reserved			TIMBK	TIM3RCLR	TIM2RCLR	TIM1RCLR	TIM0RCLR	INV_Timer
RW	RW	RW	RW	RW	RW	RW				RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:16	Reserved	Keep
15	brake	VC0 as Advanced Timer brake control 1: enable; 0: disable.
14	TIM6	VC0 filter result output to TIM6 capture input CHA 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 CHA 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 CHA 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.
8:6	Res.	Keep
5	TIMBK	VC0 filter result output to Timer0/1/2/3 brake control 1: enable; 0: disable.
4	TIM3RCLR	VC0 filter result output to TIM3 REFCLR enable control 1: enable; 0: disable.
3	TIM2RCLR	VC0 filter result output to TIM2 REFCLR enable control 1: enable; 0: disable.
2	TIM1RCLR	VC0 filter result output to TIM1 REFCLR enable control 1: enable; 0: disable.
1	TIM0RCLR	VC0 filter result output to TIM0 REFCLR enable control 1: enable; 0: disable.
0	INV_Timer	VC0 filter result output is reversed to each TIM0/1/2/3/REFCLR 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.

33.6.5 VC1 Output Configuration Register (VC1_OUT_CFG)

Offset address 0x020

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
brake	TIM6	INV_TIM6	TIM5	INV_TIM5	TIM4	INV_TIM4	Reserved			TIMBK	TIM3RCLR	TIM2RCLR	TIM1RCLR	TIM0RCLR	INV_Timer
RW	RW	RW	RW	RW	RW	RW				RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:16	Reserved	Keep
15	brake	VC1 as Advanced Timer brake control 1: enable; 0: disable.
14	TIM6	VC1 filter result output to TIM6 capture input CHB 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 CHB 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 CHB 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.
8:6	Res.	Keep
5	TIMBK	VC1 filter result output to Timer0/1/2/3 brake control 1: enable; 0: disable.
4	TIM3RCLR	VC1 filter result output to TIM3 REFCLR enable control 1: enable; 0: disable.
3	TIM2RCLR	VC1 filter result output to TIM2 REFCLR enable control 1: enable; 0: disable.
2	TIM1RCLR	VC1 filter result output to TIM1 REFCLR enable control 1: enable; 0: disable.
1	TIM0RCLR	VC1 filter result output to TIM0 REFCLR enable control 1: enable; 0: disable.
0	INV_Timer	VC1 filter result output is reversed to each TIM0/1/2/3/REFCLR 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.

33.6.6 VC Interrupt Register (VC_IFR)

Offset address 0x024

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										VC2_Filter	VC2_INTF	VC1_Filter	VC0_Filter	VC1_INTF	VC0_INTF
										RO	RW0	RO	RO	RW0	RW0

Bit	Marking	Functional description
31:4	Reserved	Keep
5	VC2_Filter	Status after VC2 Filter
4	VC2_INTF	VC2 interrupt flag, 1 VC2 interrupt occurs; 0 no interrupt occurs; writing 0 clears the interrupt flag, writing 1 is invalid
3	VC1_Filter	Status after VC1 Filter
2	VC0_Filter	Status after VC0 Filter
1	VC1_INTF	VC1 interrupt flag, 1 VC1 interrupt occurs; 0 no interrupt occurs; write 0 to clear the interrupt flag, write 1 is invalid
0	VC0_INTF	VC0 interrupt flag, 1 VC0 interrupt occurs; 0 no interrupt occurs; writing 0 clears the interrupt flag, writing 1 is invalid

33.6.7 VC2 Configuration Register (VC2_CR)

Offset address 0x150

Reset value 0x00000000

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

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			
R/W	R/W	R/W	R/W	R/W			R/W	R/W				R/W			

Bit	Marking	Functional description
31:17	Reserved	Keep
16	EN	Comparator Enable 1: enable voltage comparator 0: disable voltage comparator
15	IE	VC interrupt enable 1: enable; 0: disable
14	level	VC output signal triggers interrupt selection 1: enable high level trigger INT flag 0: disable high level triggering INT flag
13	rising	VC output signal triggers interrupt selection 1: enable rising edge to trigger INT flag 0: disable rising edge triggering INT flag
12	falling	VC output signal triggers interrupt selection 1: enable falling edge trigger INT flag 0: disable falling edge triggering INT flag
11:9	debounce_time	VC output filter time configuration 111: The filtering time is about 28.8ms 110: The filtering time is about 7.2ms 101: The filtering time is about 1.8ms 100: The filtering time is about 450us 011: The filtering time is about 112us 010: The filtering time is about 28us 001: The filtering time is about 14us 000: The filtering time is about 7us Note: The filter time configuration is only valid when FLTEN=1.
8	FLTEN	1: start VC filter 0: VC without filter
7:4	N_SEL	Voltage comparator "-" terminal input selection 0000: select channel 0 input PA5 0001: select channel 1 input PB1 0010: select channel 2 input PE11 0011: select channel 3 input PE15 0100: select channel 4 input PB11 0101: select channel 5 input PB14 0110: select channel 6 input PD10 0111: select channel 7 input PD11 1000: select channel 8 input PC7 1001: DAC0 1010: DAC1 1011:NC 1100: Built-in temperature sensor output voltage 1101: Reserved 1110: REF of ADC (ADC needs to be enabled for use) 1111: LDO output voltage
3:0	P_SEL	Voltage comparator "+" Terminal input selection 0000: select channel 0 input PA5 0001: select channel 1 input PB1 0010: select channel 2 input PE9 0011: select channel 3 input PE10 0100: select channel 4 input PE11 0101: select channel 5 input PE13 0110: select channel 6 input PE14

		0111: select channel 7 input PE15 1000: select channel 8 input PB11 1001: select channel 9 input PB14 1010: select channel 10 input PD9 1011: select channel 11 input PD10 1100: select channel 12 input PD11 1101: select channel 13 input PC7 1110: DAC0 1111: DAC1

33.6.8 VC2 Output Configuration Register (VC2_OUT_CFG)

Offset address 0x154

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
brake	TIM6	INV_Timer6	TIM5	INV_Timer5	TIM4	INV_Timer4	Reserved			TIMBK	TIM3RCLR	TIM2RCLR	TIM1RCLR	TIM0RCLR	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

Bit	Marking	Functional description
31:16	Reserved	Keep
15	brake	VC2 as Advanced Timer brake control 1: enable; 0: disable.
14	TIM6	VC2 filter result output to TIM6 capture input CHB enable 1: enable; 0: disable.
13	INV_TIM6	VC2 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	VC2 filter result output to TIM5 capture input CHB enable 1: enable; 0: disable.
11	INV_TIM5	VC2 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	VC2 filter result output to TIM4 capture input CHB enable 1: enable; 0: disable.
9	INV_TIM4	VC2 filter result output to TIM4 reverse enable 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.
8:6	Res.	Keep
5	TIMBK	VC2 filter result output to Timer0/1/2/3 brake control 1: enable; 0: disable.
4	TIM3RCLR	VC2 filter result output to TIM3 REFCLR enable control 1: enable; 0: disable.
3	TIM2RCLR	VC2 filter result output to TIM2 REFCLR enable control 1: enable; 0: disable.
2	TIM1RCLR	VC2 filter result output to TIM1 REFCLR enable control 1: enable; 0: disable.
1	TIM0RCLR	VC2 filter result output to TIM0 REFCLR enable control 1: enable; 0: disable.
0	INV_Timer	VC2 filter result output is reversed to each TIM0/1/2/3/REFCLR 1: enable reverse; 0: disable reverse, the input and VC output are in the same direction.

34 Low Voltage Detector (LVD)

34.1 Introduction to LVD

LVD can be used to monitor the voltage of VCC and chip pins. When the comparison result of the monitored voltage and the LVD threshold meets the trigger condition, the LVD will generate an interrupt or reset signal, and the user can perform some urgent tasks according to the signal.

LVD has the following characteristics:

- 4 -channel monitoring sources, AVCC, PC13, PB08, PB07;
- 16-step threshold voltage, 1.8V~3.3V optional;
- 8 trigger conditions, combinations of high level, rising edge and falling edge;
- 2 trigger results, reset and interrupt;
- 8-stage filter configuration to prevent false triggering;
- With hysteresis function, strong anti-interference.

34.2 LVD block diagram

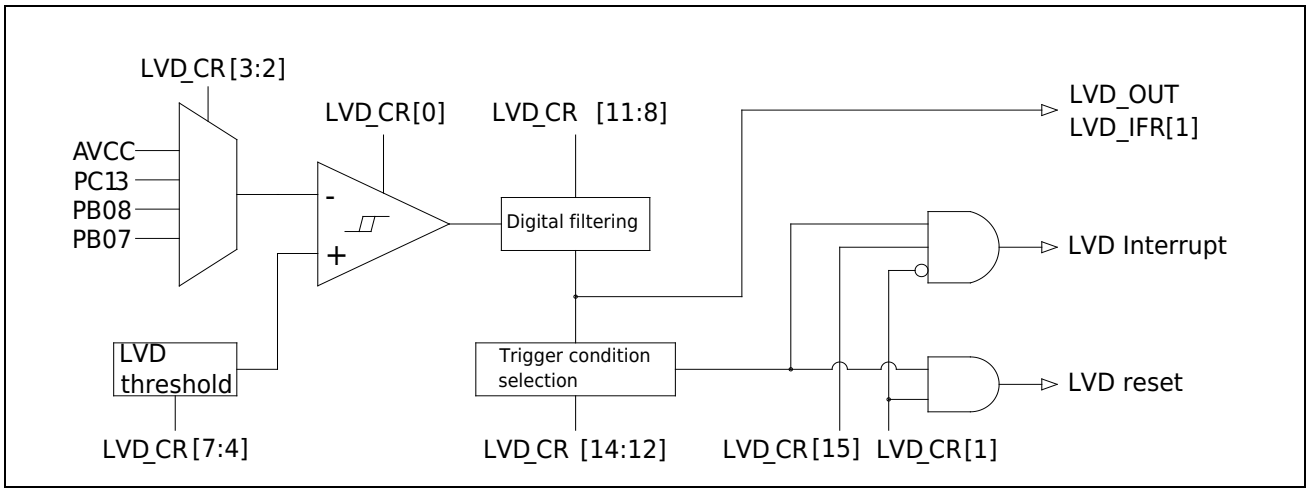


Figure 34-1 LVD Block Diagram

34.3Hysteresis function

The built-in voltage comparator of LVD has a hysteresis function, which can enhance the anti-interference ability of the chip. LVD will not flip until the input signal is higher or lower than the threshold voltage of 20mV. The input and output signals are as follows:

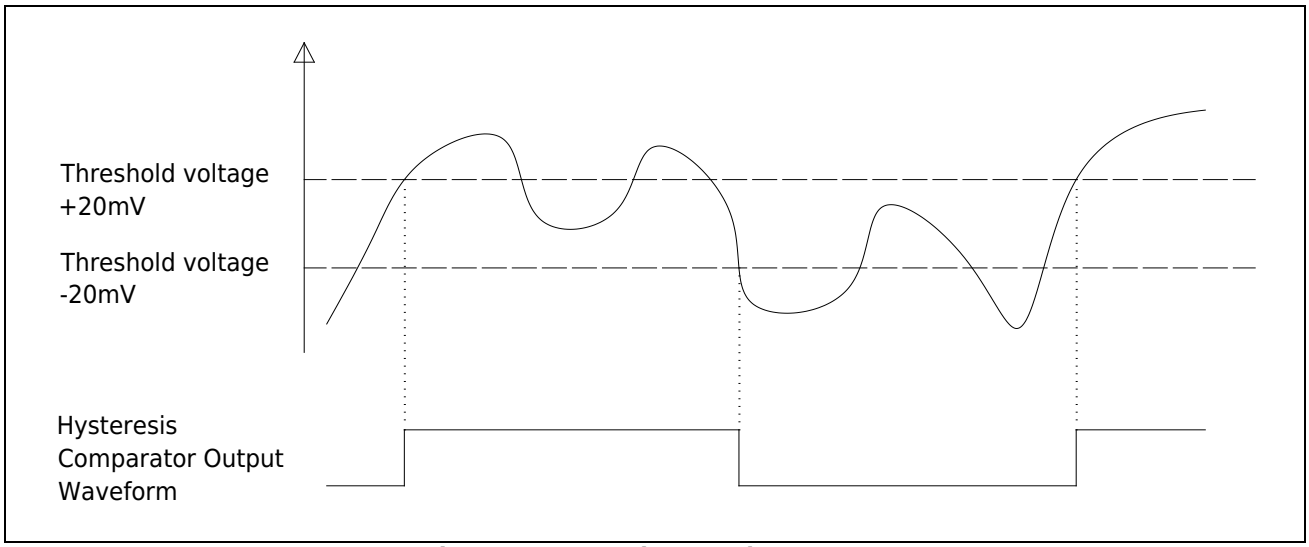


Figure 34-2 LVD hysteresis response

34.4Digital filtering

If the working environment of the chip is bad, the output of the hysteresis comparator will appear noise signal. When the digital filtering module is enabled, the noise signal whose pulse width is less than LVD_CR.Debounce_time in the output waveform of the hysteresis comparator can be filtered out. If the digital filter module is disabled, the input and output signals of the digital filter module are the same.

The digitally filtered signal level can be read from the register LVD_IFR[1]; when the GPIO function is configured as LVD_OUT the digitally filtered signal can be output from the GPIO for easy measurement.

Enable the digital filtering module, and the filtering diagram is as follows:

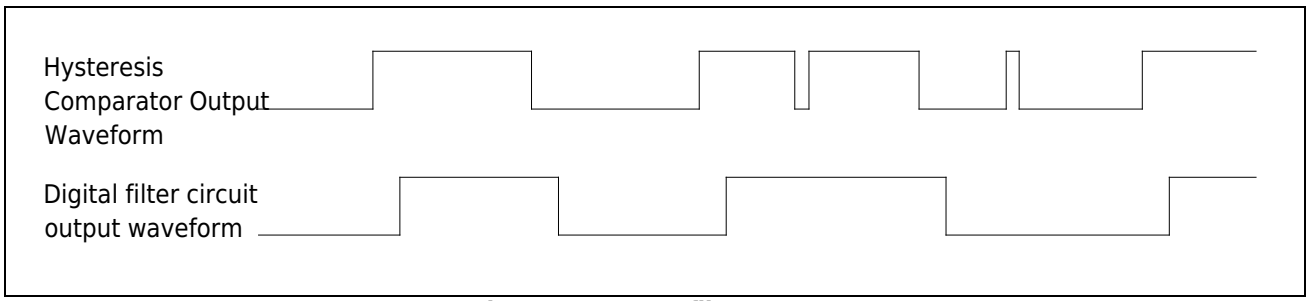


Figure 34-3 LVD filter output

34.5Configuration example

34.5.1 LVD configured as low voltage reset

In this mode, the MCU is reset when the monitored voltage is lower than the threshold voltage.

The configuration method is as follows:

Step1: Configure LVD_CR.Source_sel to select the voltage source to be monitored.

Step2: Configure LVD_CR. VTDS, select the threshold voltage of LVD.

Step3: Configure LVD_CR. Debounce_time, select LVD filter time.

Step4: Configure LVD_CR. FLTEN to enable LVD filtering.

Step5: Set LVD_CR. HTEN to 1, select high level to trigger LVD action.

Step6: Set LVD_CR.ACT to 1, select LVD action as reset.

Step7: Set LVD_CR.LVDEN to 1 to enable LVD.

34.5.2 LVD configured as voltage change interrupt

In this mode, an interrupt is generated when the monitored voltage goes above or below the threshold voltage.

The configuration method is as follows:

Step1: Configure LVD_CR.Source_sel to select the voltage source to be monitored.

Step2: Configure LVD_CR. VTDS, select the threshold voltage of LVD.

Step3: Configure LVD_CR. Debounce_time, select LVD filter time.

Step4: Configure LVD_CR. FLTEN to enable LVD filtering.

Step5: Set LVD_CR. RTEN and LVD_CR. FTEN to 1, select level change to trigger LVD action.

Step6: Set LVD_CR.ACT to 0, select LVD action as interrupt.

Step7: Set LVD_CR.IE to 1 to enable LVD interrupt.

Step8: Enable the LVD interrupt in the NVIC interrupt vector table.

Step9: Set LVD_CR.LVDEN to 1 to enable LVD.

Step10: Execute the operations required by the user in the LVD interrupt service routine; write 0x00 to LVD_IFR to clear the interrupt flag before exiting the interrupt service routine.

34.6LVD register

Base address 0x40002400

Table 34-1 LVD register

Register	Offset address	Description
LVD_CR	0x028	LVD configuration register
LVD_IFR	0x02C	LVD interrupt flag register

34.6.1 LVD configuration register (LVD_CR)

Offset address 0x028

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	HTEN	RTEN	FTEN	Debounce_time			FLTEN	VTDS				Source_sel		ACT	LVDEN
RW	RW	RW	RW	RW			RW	RW				RW		RW	RW

Bit	Marking	Functional description
31:16	Reserved	Keep
15	IE	LVD interrupt enable 1: Enable; 0: Disabled.
14	HTEN	High level trigger enable (the monitored voltage is lower than the threshold voltage) 1: Enable; 0: Disabled.
13	RTEN	Rising edge trigger enable (the monitored voltage changes from higher than the threshold voltage to lower than the threshold voltage) 1: Enable; 0: Disabled.
12	FTEN	Falling edge trigger enable (the monitored voltage changes from lower than the threshold voltage to higher than the threshold voltage) 1: Enable; 0: Disabled.
11:9	Debounce_time	Digital filter time configuration 111: The filtering time is about 28.8ms 110: The filtering time is about 7.2ms 101: The filtering time is about 1.8ms 100: The filtering time is about 450us 011: The filtering time is about 112us 010: The filtering time is about 28us 001: The filtering time is about 14us 000: The filtering time is about 7us Note: The filter time is only valid when FLTEN is 1.
8	FLTEN	Digital filter enable configuration 1: Enable digital filtering 0: disable digital filtering
7:4	VTDS	LVD Threshold Voltage Selection 1111: 3.3v 1110: 3.2v 1101: 3.1v 1100: 3.0v 1011: 2.9v 1010: 2.8v 1001: 2.7v 1000: 2.6v 0111: 2.5v 0110: 2.4v 0101: 2.3v 0100: 2.2v 0011: 2.1v 0010: 2.0v 0001: 1.9v

		0000：1.8v
3:2	Source_sel	LVD monitoring source selection 11: PB07 port input voltage 10: PB08 port input voltage 01: PC13 port input voltage 00: AVCC voltage
1	ACT	LVD trigger action selection 1: system reset 0: NVIC interrupt
0	LVDEN	LVD enable control 1: Enable LVD 0: disable LVD

34.6.2 LVD Interrupt Register (LVD_IFR)

Offset address 0x02C

Reset value 0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														LVD_ Filter	INTF
														RO	RW0

Bit	Marking	Functional description
31:2	Reserved	Keep
1	LVD_Filter	Status after LVD Filter
0	INTF	LVD interrupt flags: 1: LVD interrupt occurred; 0: No interrupt occurred; Writing 0 clears the interrupt flag, writing 1 has no effect.

35 Operational Amplifier (OPA)

The OPA module is suitable for simple amplifier and voltage follower applications. The five internal op amps can be cascaded using external resistors. The input range of OPA is 0V to AVCC, and the output range is 0.1V to AVCC-0.2V.

35.1 OPA features

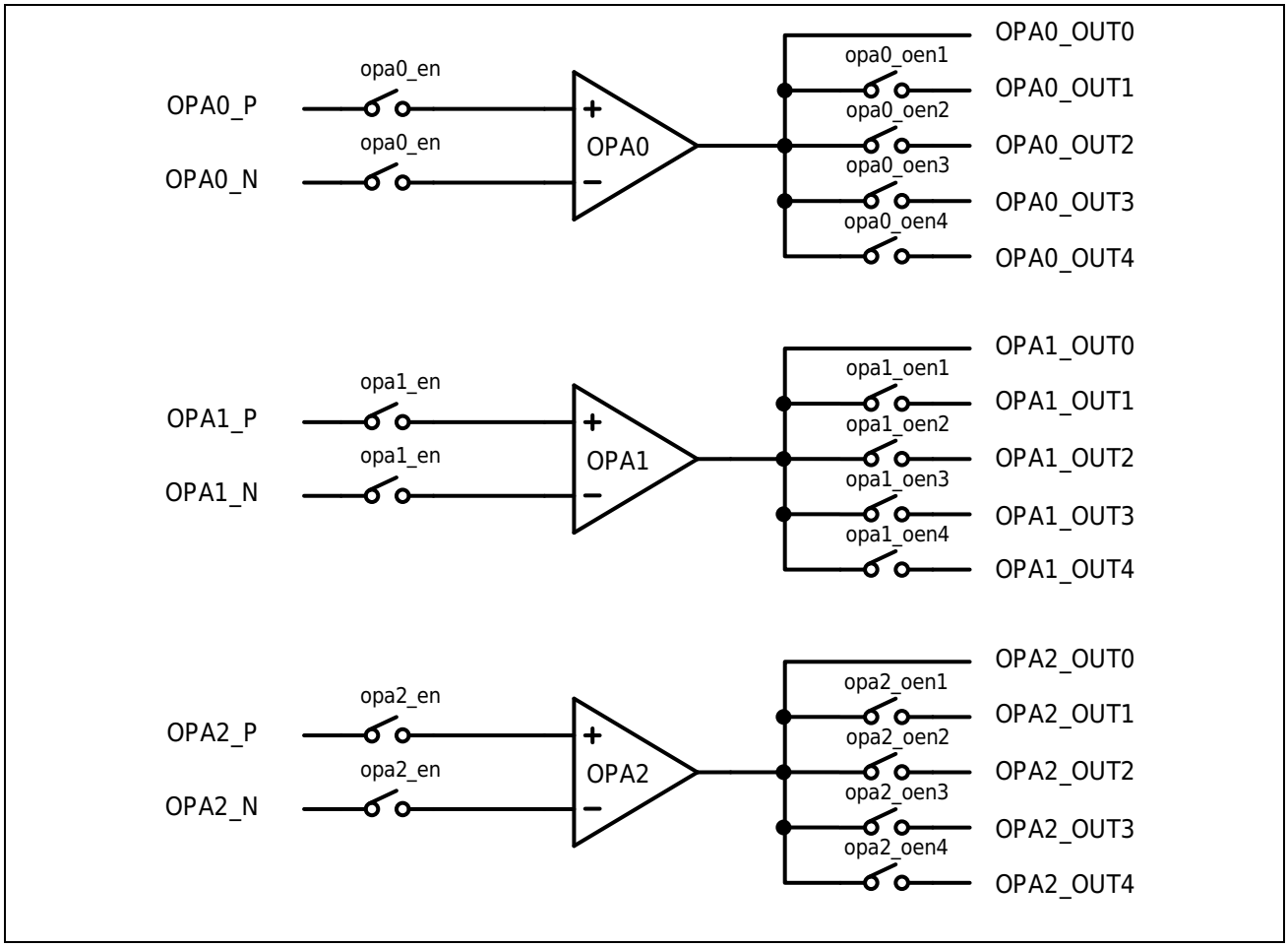
- 5 independently configured op amps
- OPA input range is 0 to AVCC, output range is 0.1V to AVCC-0.2V programmable gain
- Can be configured in the following modes
  - Universal op amp mode (general purpose OPA)
  - DAC voltage follower

35.2OPA Functional Description

5 OPAs can amplify small-signal analog input signals by using external components to form an amplifier, and output the amplified signal.

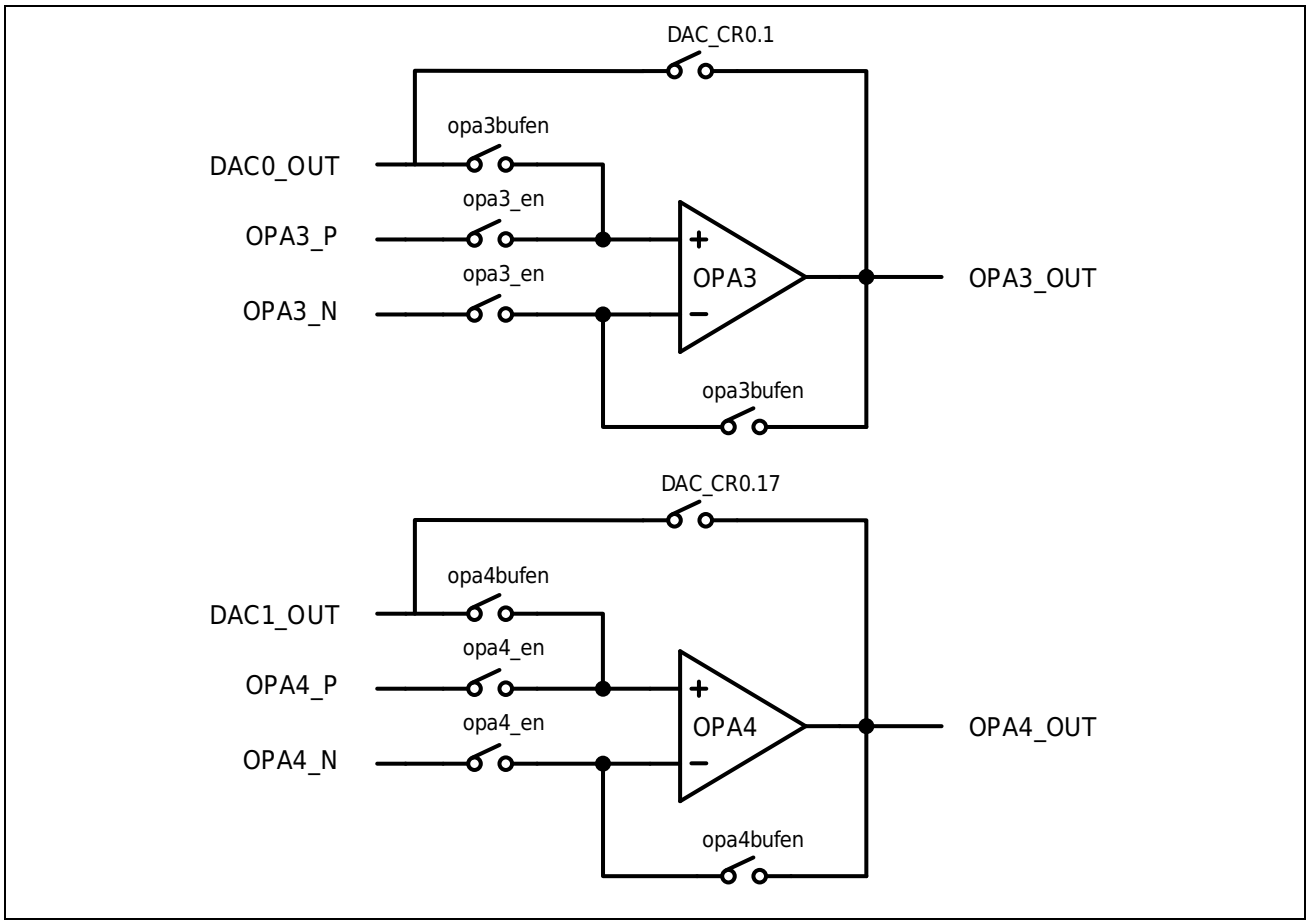
35.2.1 OPA0/1/2 (Generic OPA)

OPA0/1/2 is as follows:



35.2.2 OPA3/4 --- (can be multiplexed as DAC buffer*2)

OPA3/4 is as follows:



35.3OPA register

Base address 0x40002400

Table 35-1 OPA register

Register	Offset address	Description
OPA_CR0	0x030	OP0 Control Register
OPA_CR1	0x034	OP1 Control Register
OPA_CR2	0x038	OP2 Control Register

35.3.1 OPA Output Enable Register (OPA_CR0)

Offset address 0x030

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			OP2 OEN4	OP2 OEN4	OP2 OEN3	OP2 OEN2	OP2 OEN1	OP1 OEN4	OP1 OEN3	OP1 OEN2	OP1 OEN1	OP0 OEN4	OP0 OEN3	OP0 OEN2	OP0 OEN1
			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:12	Reserved	Keep
11	OP2OEN4	OP2 output 4 enable
10	OP2OEN3	OP2 output 3 enable
9	OP2OEN2	OP2 output 2 enable
8	OP2OEN1	OP2 output 1 enable
7	OP1OEN4	OP1 output 4 enable
6	OP1OEN3	OP1 output 3 enable
5	OP1OEN2	OP1 output 2 enable
4	OP1OEN1	OP1 output 1 enable
3	OP0OEN4	OP0 output 4 enable
2	OP0OEN3	OP0 output 3 enable
1	OP0OEN2	OP0 output 2 enable
0	OP0OEN1	OP0 output 1 enable

35.3.2 OPA Control Register (OPA_CR1)

Offset address 0x034

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				OP4BUFEN	OP3BUFEN	EN4	EN3	EN2	EN1	EN0	Reserved				
				RW	RW	RW	RW	RW	RW	RW					

Bit	Marking	Functional description
31:12	Reserved	Keep
11	OP4BUFEN	DAC1 uses OP4 unit to increase cache enable, and OPA4 enable cannot be enabled at the same time
10	OP3BUFEN	DAC0 uses OP3 unit to increase cache enable, and OPA3 enable cannot be enabled at the same time
9	EN4	OPA4 enable
8	EN3	OPA3 enable
7	EN2	OPA2 enable
6	EN1	OPA1 enable
5	EN0	OPA0 enable
4:0	Reserved	Keep

35.3.3 OPA configuration register (OPA_CR2)

Offset address 0x038

Reset value 0x0000 36DB

There is no need to change the configuration, just use the default values

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.	BIASSEL4			BIASSEL3			BIASSEL2			BIASSEL1			BIASSEL0		
	RW			RW			RW			RW			RW		

Bit	Marking	Functional description
31:15	Reserved	Keep
14:12	BIASSEL4	OPA4 Bias Current Selection Customers do not need to change the configuration, just use the default value
11:9	BIASSEL3	OPA3 bias current selection Customers do not need to change the configuration, just use the default value
8:6	BIASSEL2	OPA2 bias current selection Customers do not need to change the configuration, just use the default value
5:3	BIASSEL1	OPA1 bias current selection Customers do not need to change the configuration, just use the default value
2:0	BIASSEL0	OPA0 bias current selection Customers do not need to change the configuration, just use the default value

36 Simulate other registers

Base address 0x40002400

Register	Offset address	Description
BGR_CR	0x000	BGR Control Register

36.1 BGR Configuration Register (BGR_CR)

Offset address 0x000

Reset value 0x00000000

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

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

Bit	Marking	Functional description
31:2	Reserved	Keep
1	TS_EN	Built-in temperature sensor enable control 1: Enable internal temperature sensor 0: disable internal temperature sensor Note: After the temperature sensor is enabled for 20us, it can output a stable signal.
0	BGR_EN	BGR enable control 1: enable BGR 0: disable BGR Note: 1) This register can only be operated when Peri_clken0.ADC is 1. 2) After BGR is enabled for 20us, a stable high-precision reference voltage can be output. After the BGR is stable, it can be used by other modules, so the steps of waiting for the BGR to be stable should be added in the user operation. 3) When using ADC/OPA/PLL, BGR must be enabled. 4) When using VC, it is necessary to decide whether to enable BGR according to the configuration of the VC register.

37 SWD debug interface

This series uses the ARM Cortex-M0+ core, which has a hardware debug module SWD that supports complex debugging operations. The hardware debug module allows the core to stop when fetching instructions (instruction breakpoints) or accessing data (data breakpoints). When the kernel is stopped, both the internal state of the kernel and the external state of the system can be queried in the IDE. After the query is complete, the core and peripherals can be restored and program execution continues. When this series of microcontrollers are connected to a debugger and start debugging, the debugger will use the kernel's hardware debugging module to perform debugging operations.

Note:

- SWD can't work in DeepSleep mode, please debug in Active and Sleep mode.

37.1 SWD debugging additional functions

This product uses the ARM Cortex-M0+ CPU, which includes hardware extensions for advanced debugging functions, so this product has the same debugging functions as Cortex-M0+. Debug extensions allow the kernel to halt the kernel when fetching instructions (instruction breakpoints) or fetching data (data breakpoints). When the kernel stops, you can query the internal state of the kernel and the external state of the system. After the query completes, the kernel and system are restored and program execution is restored.

When the debug host is connected to the MCU and debugged, the debug function will be used.

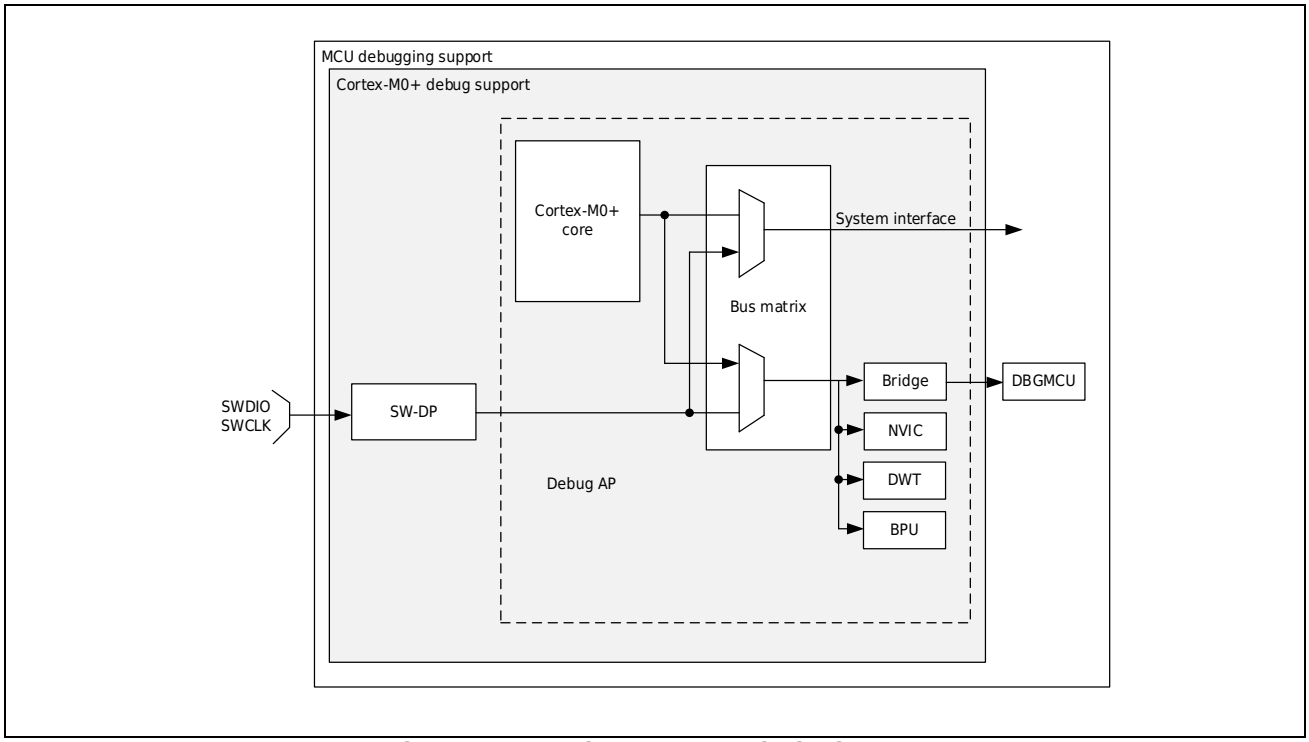


Figure 37-1 Debug Support Block Diagram

Debug features built into the Cortex®-M0+ core are part of the ARM® CoreSight Design Suite.

The ARM® Cortex®-M0+ core provides integrated on-chip debugging support. It includes:

- SW-DP: serial line
- BPU: Breakpoint unit
- DWT: data watchpoint trigger

Note:

– Details on the debug features supported by the ARM® Cortex®-M0+ core can be found in the Cortex® -M0+ Technical Reference Manual.

37.2ARM® Reference Documentation

- Cortex®-M0+ Technical Reference Manual (TRM)

Available from www.infocenter.arm.com.

- ARM® Debug Interface V5
- ARM® CoreSight Design Suite Version r1p1 Technical Reference Manual

37.3Debug port pins

37.3.1 SWD port pin

The SWD interface of this series requires 2 pins, as shown in the table below.

SWD port name	Debug function	Pin assignment
SWCLK	Serial clock	PA14
SWDIO	Serial data input / output	PA13

37.3.2 SW-DP pin assignment

If the [Encryption Chip] option is enabled when programming the program, the SWD debugging function will be disabled after power-on. If the [encrypted chip] option is not enabled when programming the program, the PA13/PA14 pins are initialized as dedicated pins that can be used by the debugger after power-on. Users can set the SYSCTRL1.SWD_USE_IO register to disable the SWD pin debug function, and the SWD pin will be released to be used as a normal GPIO. SWD pins are summarized in the following table:

[Encryption chip] option	SWD_USE_IO configuration	PA13/PA14 function
Encryption	0	NA
Encryption	1	GPIO
No encryption	0	SWD
No encryption	1	GPIO

37.3.3 Internal pull-up on SWD pin

After the user software releases the SW I/O, the GPIO controller takes control of these pins. GPIO control register puts the I/O into the equivalent state:

- SWDIO: input pull-up
- SWCLK: input pull-up

No external resistors need to be added due to built-in pull-up and pull-down resistors.

37.4SWD port

37.4.1 Introduction to SWD Protocol

This synchronous serial protocol uses two pins:

- SWCLK: Clock from host to target
- SWDIO: Bidirectional

Using this protocol, two register sets (DPACC register set and APACC register set) can be read and written simultaneously. When transferring data, the LSB comes first.

For SWDIO bidirectional management, the line must be pulled up on the board (ARM® recommends 100 k). These pull-up resistors are internally configurable. No external pull-up resistors are required.

Every time the SWDIO direction is changed in the protocol, the conversion time is inserted, and the line is not driven by the host or the target. By default, this conversion time is one bit time, but it can be adjusted by configuring the SWCLK frequency.

37.4.2 SWD protocol sequence

Each sequence consists of three phases:

1. The host sends a packet request (8 bits)
2. Acknowledgment response sent by target (3 digits)
3. The host or target sends the data transfer phase (33 bits)

Bit	Name	Note
0	Start up	Must be 1
1	APnDP	0: DP access; 1: AP access
2	RnW	0: write request; 1: read request
4:3	A[3:2]	Address field of DP or AP register
5	Parity	Unit parity for the first few bits
6	Stop	0
7	Reside	Not driven by the host. 1 by target due to pull-up

For a detailed description of the DPACC and APACC registers, refer to the Cortex®-M0+ TRM.

The packet request is always followed by a conversion time (1 bit by default), at which point neither the host nor the target will drive.

Bit	Name	Note
0	ACK	001：FAULT 010：WAIT 100：OK

The ACK response must be followed by transition time only when a READ transaction occurs or a WAIT or FAULT acknowledgment is received.

Bit	Name	Note
0:31	WDATA or RDATA	Write or Read data
32	Parity	Single parity for 32 data bits

The transfer time must be after the DATA transfer only when a READ transaction occurs.

37.4.3 SW-DP state machine (reset, idle state, ID code)

SW-DP has an internal ID code for identifying SW-DP. The code is JEP-106 compliant. This ID code is the default ARM® code, set to 0x0BC11477 (equivalent to Cortex®-M0+).

Note:

- The SW-DP state machine is inactive until the target reads the ID code.
- The SW-DP state machine is in reset state after power-on reset or after the line has been at high level for more than 50 cycles.
- If the line is low for at least two cycles after the reset state, the SW-DP state machine is in the idle state.
- After the reset state, the state machine must first enter the idle state and then perform a read access to the DP-SW ID CODE register. FAULT acknowledgment response on another transaction.

Cortex®-M0+ TRM and CoreSight Design Suite r1p0TRM for more details on the SW-DP state machine.

37.4.4 DP and AP read / write access

- DP is not delayed: the target response can be sent immediately (if ACK=OK) or delayed (if ACK=WAIT).
- Delay read access to AP. This means that access results will be returned on the next transfer. If the next access to be performed is not an AP access, the DP-RDBUFF register must be read to obtain the result.
- Every time an AP read access or RDBUFF read request is made, the READOK flag of the DP-CTRL/STAT register will be updated to know whether the AP read access is successful.
- SW-DP has a write buffer (for DP or AP writes) so that write operations can be accepted even while other operations are still pending. If the write buffer is full, the target acknowledgment

response is WAIT. Except for IDCODE read, CTRL/STAT read or ABORT write, these operations will also be accepted when the write buffer is full.

- Due to the asynchronous clock domains SWCLK and HCLK, two additional SWCLK cycles are required after the write operation (after the parity bit) for the write operation to take effect internally. These cycles should be applied while the line is being driven low (idle state).

This is especially important when writing to the CTRL/STAT register to initiate a power-on request. Otherwise the next operation (one that is only valid after the core is powered on) will be executed immediately, which will cause a failure.

37.4.5 SW-DP register

These registers can be accessed when APnDP=0:

A[3:2]	RW	CTRLSEL bit of the SELECT register	Register	Note
00	Read		IDCODE	Manufacturer code set to default ARM® for Cortex®-M0+ Code. 0x0BC11477 (identifies SW-DP)
00	Write		ABORT	
01	Read / Write	0	DP-CTRL/STAT	Purpose: – Request system or debug power up – Configure transport operations for AP access – Control compare and verify operations – read some status flags (overflow and power-on acknowledgment)
01	Read / Write	1	WIRE CONTROL	Used to configure the physical serial port protocol (such as switching time duration)
10	Read		READ RESEND	Allow recovery of reads from corrupted debug software transfers data without repeating the original AP transmission.
10	Write		SELECT	4 -word register used to select the currently accessed port and active window
11	Read / Write		READ BUFFER	Since the AP access has been issued, this read buffer is very useful Used (provide read AP request when executing the next AP transaction the result of). This read buffer captures data from the AP, shown as

The result of a previous read, without initiating a new operation.

37.4.6 SW-AP Register

These registers can be accessed when APnDP=1:

There are multiple AP registers, which are addressed in the following combinations:

- Shift value A[3:2]
- Current value of DP SELECT register

Address	A[3:2] value	Note
0x0	00	Reserved, must remain at reset value.
0x4	01	DP CTRL/STAT register. Used for: – Request system or debug power up – Configure transport operations for AP access – Control compare and verify operations – read some status flags (overflow and power-on acknowledgment)
0x8	10	DP SELECT Register: Used to select the currently accessed port and active 4 - word register window. – Bits 31:24: APSEL: Select the current AP (select the current AP) – Bits 23:8: Reserved – Bits 7:4: APBANKSEL: Select the active 4- word register window on the current AP – Bits 3:0: Reserved
0xC	11	DP RDBUFF register: used to obtain the final result after performing a series of operations through the debugger (No need to request new JTAG-DP operations)

37.5Kernel debugging

Debug the kernel through the kernel debug registers. Debug access to these registers is through the debug access port. It consists of four registers:

Register	Note
DHCSR	*32*-bit debug stop control and status register This register provides information about the state of the processor, enables the core to enter a debug stop state, and provides processor stepping capabilities.
DCRSR	*17*-bit debug core register selector registers: This register selects the processor register to be read or written.
DCRDR	*32*-bit Debug Core Registers Data Registers: This register holds the data read and written between the register and the processor selected by the DCRSR (selector) register data.
DEMCR	*32*-bit Debug Exception and Watchdog Control Registers: This register provides vector capture and debug monitor control.

These registers are not reset on system reset. They can only be reset by a power-on reset. See Cortex®-M0+ TRM for more details.

In order to put the core into the debug stop state immediately after reset, it is necessary to:

- Enable Debug and Exception Monitoring Control Register Bit 0 (VC_CORRESET)
- Enable Debug Halt Control and Status Register bit 0 (C_DEBUGEN)

37.6BPU (Breakpoint Unit)

The Cortex®-M0+ BPU implementation provides four breakpoint registers.

37.6.1 BPU function

Processor breakpoint implements PC- based breakpoint functionality.

See the ARMv6-M ARM® and ARM® CoreSight Components Technical Reference Manual for more information on the BPU CoreSight Identification Registers, their addresses and access types.

37.7DWT (Data Watchpoint)

The Cortex®-M0+ DWT implementation provides two watchpoint register sets.

37.7.1 DWT function

Processor watchpoints implement data address and PC-based watchpoint functionality (i.e. PC sampling registers) and support for comparator address masks as described in ARMv6-M ARM®.

37.7.2 DWT Program Counter Sample Register

Processors implementing the Data Watchpoint Unit also implement the ARMv6-M optional DWT Program Counter Sampling Register (DWT_PCSR). This register allows the debugger to periodically sample the PC without halting the processor. This provides a rough analysis. See ARMv6-M ARM® for more information.

Cortex®-M0+ DWT_PCSR records passing condition codes and instructions and instructions failing condition codes.

37.8MCU debug group (DBG)

MCU Debug Component helps the debugger provide support for:

- Low power mode
- Timer during breakpoint, clock control of watchdog

37.8.1 Debug support for low power modes

To enter low-power mode, the instruction WFI or WFE must be executed.

The MCU supports several low-power modes that disable the CPU clock or reduce CPU power consumption.

FCLK or HCLK during a debug session. They must remain active as they are required for debug connections during debugging. The MCU incorporates special methods that allow users to debug software in low-power modes.

37.8.2 Debug support for timers, watchdog

During a breakpoint, the behavior of the counters of the timer and watchdog must be selected:

- The counter continues to count while the breakpoint is generated. For example, this is often required when PWM is controlling a motor.
- When a breakpoint is generated, the counter stops counting. This is required when used with a watchdog.

37.9Debug mode module working state control (DEBUG_ACTIVE)

Reset value 0x00001FFF (this register setting has an effect only in SWD debug mode)

Offset address: 0x4000_2038

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			LPTIM1	TIM3	Res.	RTC	WDT	PCA	TIM6	TIM5	TIM4	LPTIM0	TIM2	TIM1	TIM0
			RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:13	Reserved	Keep
12	LPTIM1	When debugging, the LpTimer1 counting function configures 1: In the SWD debugging interface, suspend the counting function of LPTimer1 0: In the SWD debugging interface, LPTimer1 counts normally
11	TIM3	When debugging, Timer3 count function configuration 1: In the SWD debugging interface, the Timer3 counting function is suspended 0: In the SWD debugging interface, Timer3 counts normally
10	Reserved	Keep
9	RTC	When debugging, the RTC count function configuration 1: In the SWD debugging interface, suspend the RTC counting function 0: In the SWD debugging interface, the RTC counts normally
8	WDT	When debugging, the WDT count function configures the 1: In the SWD debugging interface, suspend the WDT counting function 0: In the SWD debugging interface, WDT counts normally
7	PCA	When debugging, the PCA counting function configures 1: In the SWD debugging interface, the PCA counting function is suspended 0: In the SWD debugging interface, PCA counts normally
6	TIM6	When debugging, Timer6 counting function configuration 1: In the SWD debugging interface, the Timer counting function is suspended, and the counting value of the counter needs to be initialized to run again after the pause, otherwise the PWM output phase may be wrong. 0: In the SWD debugging interface, the Timer counts normally
5	TIM5	When debugging, Timer5 count function configuration 1: In the SWD debugging interface, the Timer counting function is suspended, and the counting value of the counter needs to be initialized to run again after the pause, otherwise the PWM output phase may be wrong. 0: In the SWD debugging interface, the Timer counts normally
4	TIM4	When debugging, Timer4 count function configuration 1: In the SWD debugging interface, the Timer counting function is suspended, and the counting value of the counter needs to be initialized to run again after the pause, otherwise the PWM output phase may be wrong. 0: In the SWD debugging interface, Timer counts normally
3	LPTIM0	When debugging, LpTimer0 counting function configuration 1: In the SWD debugging interface, suspend the counting function of LPTimer0 0: In the SWD debugging interface, LPTimer0 counts normally
2	TIM2	When debugging, Timer2 counting function configuration 1: In the SWD debugging interface, the Timer counting function is suspended 0: In the SWD debugging interface, the Timer counts normally
1	TIM1	When debugging, the Timer1 count function configures the 1: In the SWD debugging interface, the Timer counting function is suspended 0: In the SWD debugging interface, the Timer counts normally
0	TIM0	When debugging, Timer0 counting function configuration 1: In the SWD debugging interface, the Timer counting function is suspended 0: In the SWD debugging interface, the Timer counts normally

38 Device electronic signature

The electronic signature is stored in the system storage area of the flash memory module and can be read by SWD or CPU. HC32Fxxx / HC32Lxxx microcontrollers with different configurations.

38.1 Product Unique Identifier (UID) Register (80 bits)

Typical application scenarios for unique identifiers:

- Used as a serial number
- UID is used as a security key when used in conjunction with software cryptographic primitives and protocols to increase the security of code in Flash prior to programming the internal Flash
- Activation of the secure bootstrap process, etc.

The 80-bit unique device identifier provides a reference number that is unique to any device and in any context. The user can never change these bits. The 80-bit UID can also be read in different ways like single byte/halfword/word and then concatenated using a custom algorithm.

Base address: 0x0010 0E74

Offset address	Description	UID Bits（80 bits）							
		7	6	5	4	3	2	1	0
0	Lot Number	UID[7:0]							
1		UID[15:8]							
2		UID[23:16]							
3		UID[31:24]							
4		UID[39:32]							
5		UID[47:40]							
6	X Coordinate on the wafer	UID[55:48]							
7	Y Coordinate on the wafer	UID[63:56]							
8	Wafer Number	UID[71:64]							
9	Rev ID	UID[79:72]							

38.2 Product Model Register

0x0010 0C60 ~ 0x0010 0C6F stores the ASCII code of the product model. If the product model is less than 16 bytes, fill it with 0x00.

Example: The product model represented by 484333324C3133364B38544100000000 is HC32L136K8TA.

38.3FLASH capacity register

Base address: 0x0010 0C70

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

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

Bit	Marking	Functional description
31:0	FlashSize	The capacity of the product's built-in Flash, in bytes 0x00008000 means the Flash capacity is 32K Byte

38.4RAM capacity register

Base address: 0x0010 0C74

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

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

Bit	Marking	Functional description
31:0	RamSize	The capacity of the product's built-in RAM, in bytes 0x00000800 means the RAM capacity is 2K Byte

38.5Pin Count Register

Base address: 0x0010 0C7A

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

Bit	Marking	Functional description
15:0	PinCount	The number of product pins, in unit 0x0020 means that the number of product pins is 32

39 Appendix A SysTick Timer

39.1 Introduction to SysTick Timers

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

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

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

39.2 Set SysTick

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

Step1: Configure SysTick->CTRL. ENABLE is 0, disable SysTick.

Step2: Configure SysTick->CTRL. CLKSOURCE to select the clock source of SysTick.

Step3: Configure SysTick->LOAD, select the overflow period of SysTick.

Step4: Write any value to SysTick->VAL, clear SysTick->VAL and SysTick->CTRL. COUNTFLAG.

Step5: Configure SysTick->CTRL. TICKINT is 1, enable SysTick interrupt.

Step6: Set Sys Tick->CTRL. ENABLE to 1 to enable SysTick.

Step7: Read SysTick->CTRL in the interrupt service routine to clear the overflow flag.

Note: Systick overflow period is SysTick->LOAD+1, the configuration example is as follows:

Timer	SysTick->LOAD	Overflow cycle
HCLK 4MHz	3999	1ms
XTL 32.768KHz	327	10.01ms

39.3SysTick register

Address	Name	CMSIS symbol	Full name
0XE000E010	SYS_CSR	SysTick->CTRL	SysTick Control and Status Register
0XE000E014	SYS_RVR	SysTick->LOAD	SysTick reload register
0XE000E018	SYS_CVR	SysTick->VAL	SysTick current value register
0XE000E01C	SYS_CALIR	SysTick->CALIB	SysTick Calibration Value Register

39.3.1 SysTick Control and Status Register (CTRL)

Bit	Symbol	Functional description	Types of	Reset value
31:17	Reserved	-	-	-
16	COUNTFLAG	SysTick timer overflow flag 1: SysTick timer underflow occurred 0: SysTick timer has not overflowed Reading this register clears the COUNTFLAG flag	RO	0
15:3	Reserved	-	-	-
2	CLKSOURCE	SysTick clock source selection 1: Use the core clock (HCLK) 0: Use reference clock (XTL) Note: When the reference clock is selected as the clock source of SysTick, the XTL clock source and SysCtrl.PERI_CLKEN0.TICK need to be enabled synchronously	RW	0
1	TICKINT	SysTick interrupt enable 1: enable interrupt 0: disable interrupt	RW	0
0	ENABLE	SysTick timer enable 1: enable SysTick 0: disable SysTick	RW	0

39.3.2 SysTick reload register (LOAD)

Bit	Symbol	Functional description	Types of	Reset value
31:24	Reserved	-	-	-
23:0	RELOAD	SysTick timer reload value	RW	Undefined

39.3.3 SysTick Current Value Register (VAL)

Bit	Symbol	Functional description	Types of	Reset value
31:24	Reserved	-	-	-
23:0	CURRENT	Read this register to get the current count value of the SysTick timer Write any value to this register, clear this register and COUNTFLAG	RW	Undefined

39.3.4 SysTick Calibration Value Register (CALIB)

Bit	Symbol	Functional description	Types of	Reset value
31	NOREF	SysTick current count clock flag 1: The current count clock is the core clock 0: The current count clock is external reference clock (XTL)	RO	-
30	SKEW	TENMS Accuracy Indication 1: TENMS value represents roughly 10ms 0: TENMS value represents exactly 10ms	RO	
29:24	Reserved	-	-	
23:0	TENMS	10 millisecond calibration value	RO	-

40 Appendix B Document Conventions

40.1 List of register-related abbreviations

The following abbreviations are used in register descriptions:

RW: Read and write, software can read and write these bits.

RO: Read only, software can only read these bits.

WO: Write only, software can only write to this bit. Reading this bit will return invalid data.

W1: Only write 1, the hardware automatically clears 0, writing 0 is invalid.

R0W1: software reads this bit as 0, writes 1 to clear this bit. Writing a 0 has no effect on the value of this bit.

RW0: Software can read and write this bit, writing 1 is invalid, writing 0 clears.

R1W0: Software reads this bit as 1, 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 "0" has no effect on the value of this bit.

Res, Reserverd: Reserved bit, must keep the reset value.

40.2 Glossary

This section provides brief definitions of acronyms and abbreviations used in this document:

Word: 32 -bit data.

Half Word: 16 -bit data.

Byte: 8 -bit data.

IAP (In-Application Programming): IAP means that the microcontroller's Flash can be reprogrammed while the user program is running.

ICP (In-Circuit Programming): ICP means that the Flash of the microcontroller can be programmed using the JTAG protocol, SWD protocol or bootloader when the device is installed on the user application circuit board.

AHB: Advanced High Performance Bus.

APB: Low-speed peripheral bus.

DMA: Direct Memory Access.

TIM: Timer.

Version Revision History

Version Number	Revision Date	Modify the content
Rev1.00	2019/11/20	Released the first draft.
Rev1.10	2019/12/31	Updated the following information: ①Added QFN32 package; ②Typical application circuit diagram; ③Device electronic signature; ④Appendix A SysTick timer; ⑤Diagrams and precautions for high-speed external clock XTH and low-speed external clock XTL; ⑥ General working conditions description; ⑦System control register 1 (SYSCTRL1) adds notices; ⑧Used a temperature sensor to measure the ambient temperature; ⑨The description of the DMA controller (DMAC) and cyclic redundancy check (CRC) chapters; ⑩The 0 bit of the I2C configuration register (I2Cx_CR).
Rev1.20	2020/04/10	Updated the following information: ① Pin function description; ② 29.7.2.17 and 29.7.2.24 added notices; ③ 7.5.1 and 7.5.2 Step8; ④ package size QFN32; ⑤ 3.2.7 and 3.2.8 add Step; ⑥Added AVCC/3 precision in ADC features; ⑦Changed 44/45/47/48pin of HC32L073KATA; ⑧Changed 32/33/35/36pin of HC32L073JATA.
Rev1.30	2020/05/29	Updated the following information: ① 11.3.2 deleted the note; ② Corrected typos in external low-speed clock input; ③ Added LCD controller; ④ RCL oscillator accuracy in internal RCL oscillator; ⑤ Product features Added low-power timer description in ultra-low-power MCU; ⑥Added Step2 and Step3 in 7.5.
Rev1.40	2020/06/30	Updated the following information: ①Added I2S information in the pin function description; ②Corrected LPTIM to LPTIM0, and corrected LPTIMx_ETR to LPTIMx_EXT; ③Unified pin function names and unified register names; ④Updated 21.3.1.6 and 22.3.2.6 descriptions.
Rev1.50	2020/07/31	Updated the following information: ①Input characteristics - V_{IH} value and value in ports PA, PB, PC, PD, PE, PF; ②Added TIM timer characteristics and communication interface section; ③EFT characteristic level; ④Adjusted 29.7.1 and 29.7.2 Registers.
Rev1.60	2020/09/30	Updated the following information: ①Functional block diagram in function module; ②Added SPI feature and I2S feature; ③11.8.4, 11.8.14 and 35.4.2 updated typos; ④Clock system description; ⑤V_{IL} and V_{IH} of RESETB pin characteristics; ⑥Added input Features - USB_DP, USB_DM; ⑦ bit3 of 28.7.3.
Rev1.61	2020/12/31	Deleted product features, pin configuration, packaging information, etc. (please refer to the latest data sheet for related information), and revised the statement.
Rev1.62	2021/09/17	Replaced with high-resolution vector graphics.
Rev1.63	2022/03/09	Updated the company logo.
Rev1.64	2022/08/13	Updated the following information: ①Chapter "I2C bus ", I2C operation process errata; ②Chapter " External high-speed crystal oscillator clock XTH", added a description that recommends using an active crystal oscillator; ③Chapter " Port digital multiplexing function ", deleted the PF01 function of TIM4_CHB Mapping; ④Chapter " Cycle Interval Response ", added the description of the initialization method of this function and precautions; ⑤Chapter " Baud Rate Setting ", added the description of sampling points; ⑥Chapter " Controller Local Area Network ", added the precautions for using CAN.
Rev1.65	2023/06/21	"Pulse Width Measurement PWC register Description" section, added the function description not opened to the control register (TIMx_CR0).
Rev1.70	2024/05/23	Updated the following information: ①3.5.2 Changed the notes of the EXTL_EN and EXTH_EN function description of the SYSCTRL1 register;

Version Number	Revision Date	Modify the content
		②7.5 Updated the programming example operation steps; ③Added PLL to the monitored clock source in 9.3.2.2 Clock Monitoring Hardware Structure; ④Added notes in 16.2.3.2 PCA 16-bit PWM function; ⑤Changed PC5 to PC15 in 17.3.30 TIMx_PTBKS and 17.3.33 TIMx_PTBKP; ⑥22.4.2 Changed "HSEL" to "HDSEL" in LPUARTx_SCON; ⑦Updated the BEIF function description of 30.5.9 CAN_ERRINT register; ⑧Removed 1.2V related content from the analog-to-Digital Converter (ADC) section; ⑨Removed the reference voltage of 1.2V and the on-chip BGR output from the Analog Voltage Comparator (VC) section.
Rev1.71	2025/01/14	3.5.2 Changed the notes in the function description of EXTL_EN and EXTH_EN of SYSCTRL1 register.
Rev1.72	2025/03/04	Deleted the description of the maximum communication rate between the master and slave machines in the 8.2 Main Features of SPI.
Rev1.73	2026/02/11	Updated the following information: ① In "Mode 2/3 compare capture mode - Single pulse output" section, optimized the details of pulse count timing diagram, and added relevant instructions; ② In the "Mode 2/3 compare capture mode - Capture Input" section, added a new note: "When GATE is configured for compare output, its output mode only supports being set to force the output to 0 or 1 and having the PWM mode active (that is, the GATE output is valid when OCMB0 is configured as 0/1/6/7)"; ③ In the "Single-Wire Half-Duplex Data Transmitting and Reception" section, added the description "After configuring LPUART_SCON.HDSEL to 0b1, must wait for the transmission duration of one data bit (i.e., after 1/BaudRate seconds) before normal reception can occur"; ④Deleted the section "OPA zero calibration", deleted the relevant OPA zero description in the OPA_CR1 register, and deleted the OPA_CR register.
Rev1.80	2026/08/19	Updated the following information: ① In the section "13.4.4 DMA Hardware Block Transmission Mode", modified the hardware block transmission schematic; ② In the section "14.2.6 Mode 2/3 Comparison Capture Mode - Capture Input", modified the constraint for selecting the input signal for CH0B; ③ In the sections "21.3.4 Multi-machine Communication" and "22.3.5 Multi-machine Communication", modified the description of the broadcast address detected by the slave address detection; ④ In the section "37.4 SWD Port", changed SW-DP core ID "0x0BB11477" to "0x0BC11477".