Shanghai Fudan Microelectronics Group Company Limited

复旦微电子

FM33LE0xxA

Low-power MCU Chip

User Manual

2023.09

This information is provided as a reference material for users to choose the appropriate products of Shanghai Fudan Microelectronics Group Co., Ltd. (hereinafter referred to as Fudan Microelectronics) according to their use. It's not allowed to transfer intellectual property and other rights owned by Fudan Microelectronics or a third party. Before using the information recorded in this document to finally make a judgment on whether the information and products are applicable, please be sure to evaluate all the information as a whole system. The purchaser is solely responsible for the selection and use of the products and services of Fudan Microelectronics described in this article. Fudan Microelectronics is not responsible for the purchaser's selection and use of the products and services described in this article. Fudan Micros also does not guarantee that the manual is error-free . Unless expressly approved in writing, Fudan Microelectronics products are not recommended, authorized, or guaranteed for use in military, aviation, aerospace, life-saving and life-support systems. In the product or system that may cause personal injury or death, serious property or environmental damage due to failure or malfunction. Without the permission of Fudan

Microelectronics, it is not allowed to reprint or copy all or part of the content of this material.In the future, daily product updates will be released in due course without notice. When purchasing the products described in this document, please contact the local sales office of Fudan Microelectronics in advance To confirm the latest information, and please follow the information published by Fudan Microelectronics in various ways, including Fudan Microelectronics website (http://www.fmsh.com/).

If you need to know the details of the information or products recorded in this document, please contact the local sales office of Shanghai Fudan Microelectronics Group Co., Ltd.

Contents

CONTENTS **4**

LIST OF TABLES **23**

LIST OF FIGURES **25**

1 FEATURES **32**

1.1 INTRODUCTION 32
1.2 BLOCK DIAGRAM 34
1.3 DEVICE LINEUP 35

2 PINOUT **36**

2.1 PACKAGE AND PIN 36
2.1.1 FM33LE0x5A(LQFP48) *36*
2.1.2 FM33LE0x3A (QFN32) *37*
2.1.3 Pin descriptions (FM33LE0xxA) *38*
2.1.4 Package information *44*
2.2 WELDING INSTALLATION INSTRUCTIONS 48
2.3 MSL RATING 50
2.4 THERMAL RESISTANCE CHARACTERISTICS 50

3 ELECTRICAL CHARACTERISTICS **51**

3.1 PARAMETER CONDITIONS ..... 错误!未定义书签。
3.2 POWER SUPPLY SCHEME 51
3.3 ABSOLUTE MAXIMUM RATINGS 53
3.4 OPERATING CONDITIONS 54
3.4.1 General operating conditions *54*
3.4.2 Current consumption characteristics *54*
3.4.3 Reset and supply detection *56*
3.4.4 High precision reference *57*
3.4.5 Wake up time in low-power mode *58*
3.4.6 External clock source characteristics *59*
3.4.7 Internal clock characteristics *62*
3.4.8 PLL Characteristics *64*
3.4.9 ADC Characteristics *64*
3.4.10 emperature sensor *68*
3.4.11 Analog comparator characteristics *69*
3.4.12 Flash characteristics *69*
3.4.13 GPIO characteristics *69*
3.4.14 LCD characteristics ..... *错误!未定义书签。*

4 POWER MANAGEMENT UNIT (PMU) 72

4.1 POWER SUPPLY 72
4.1.1 Power domains 72
4.1.2 Power supply structure 73
4.1.3 ADC and independent power supply of internal reference 73
4.1.4 ON-chip reference source (AVREF) 73
4.2 CONSUMPTION MODE 74
4.2.1 Introduction 74
4.2.2 Power mode and system frequency 76
4.2.3 Active mode 78
4.2.4 LP Active mode 78
4.2.5 LPRUN mode 79
4.2.6 SLEEP mode 79
4.2.7 DEEPSEELP mode 80
4.3 WAKE-UP EVENTS 81
4.4 THE SYSTEM CLOCK AFTER WAKE-UP 82
4.5 REGISTER 83
4.5.1 Low-power consumption control register (PMU_CR) 83
4.5.2 Wakeup time control register (PMU_WKTR) 84
4.5.3 Wakeup source flag register (PMU_WKFR) 85
4.5.4 PMU interrupt enable register (PMU_IER) 87
4.5.5 PMU interrupt flag register (PMU_ISR) 88

5 INTERNAL REFERENCE SOURCE (VREF) 90

5.1 INTRODUCTION 90
5.2 REGISTER 91
5.2.1 VREF_PATA control register (VREF_PATA_CR) 91
5.2.2 Buffer control register (VREF_BUFCR) 92

6 CPU 93

6.1 INTRODUCTION 93
6.1.1 Processor configuration 93
6.2 REGISTER 94
6.3 EXCEPTIONS AND INTERRUPTS 94
6.3.1 Interrupt vector table 94
6.3.2 Interrupt priority level 96
6.3.3 Error handing 96
6.3.4 Lockup 97
6.4 DEBUG FEATURES 97
6.4.1 Debug function PIN 97
6.4.2 Watchdog control in debug mode 97
6.4.3 DEBUG reset 98
6.5 REGISTER 98
6.5.1 DEBUG Configuration register (DBG_CR) 98

6.5.2 HardFault query register (DBG_HDFR) 99

7 BUS AND MEMORIES 101

7.1 SYSTEM BUS 101
7.2 MEMORY SPACE ALLOCATION 102
7.2.1 Introduction 102
7.2.2 Peripherals address assignment 104
7.3 RAM 106
7.3.1 Introduction 106
7.4 FLASH 106
7.4.1 Introduction 106
7.4.2 Special information sector description 106
7.4.3 Flash Program 110
7.4.4 Data Flash 116
7.4.5 Flash memory protection 116
7.5 REGISTER 119
7.5.1 Flash Read Control Register (FLS_RDCR) 119
7.5.2 Flash Prefetch Control Register (FLS_PFCR) 120
7.5.3 Flash Option Byte Register (FLS_OPTBR) 120
7.5.4 ACLOCK register1 (FLS_ACLOCK1) 121
7.5.5 Flash Erase/Program Control Register (FLS_EPCR) 122
7.5.6 Flash Key Register (FLS_KEY) 123
7.5.7 Flash Interrupt Enable Register (FLS_IER) 123
7.5.8 Flash Interrupt Status Register (FLS_ISR) 124

8 RESET MANAGEMENT UNIT (RMU) 126

8.1 INTRODUCTION 126
8.2 BLOCK DIAGRAM 127
8.3 POWER-ON RESET AND POWER-DOWN RESET (POR&PDR) 127
8.4 SOFTWARE RESET 128
8.5 NRST PIN RESET 129
8.6 REGISTER 130
8.6.1 PDR control register (RMU_PDRCR) 130
8.6.2 BOR control register (RMU_BORCR) 131

9 INDEPENDENT WATCHDOG (IWDT) 132

9.1 IWDT INTRODUCTION 132
9.2 IWDT BLOCK DIAGRAM 132
9.3 IWDT FUNCTION DESCRIPTION 132
9.4 IWDT WINDOW FUNCTION 133
9.5 IWDT FREEZE 134
9.6 REGISTER 135
9.6.1 IWDT serve (IWDT_SERV) 135
9.6.2 IWDT config register (IWDT_CR) 135
9.6.3 IWDT counter register (IWDT_CNT) 136

9.6.4 IWDT interrupt enable register (IWDT_IER) *137*
9.6.5 IWDT interrupt Flag register (IWDT_ISR) *137*

10 WINDOW WATCHDOG (WWDT) **139**

10.1 INTRODUCTION 139
10.2 WWDT BLOCK DIAGRAM 139
10.3 WWDT FUNCTIONAL DESCRIPTION 140
10.4 REGISTER 142
10.4.1 WWDT Control Register (WWDT_CR) *142*
10.4.2 WWDT Config Register (WWDT_CFGR) *142*
10.4.3 WWDT Counter Register (WWDT_CNT) *143*
10.4.4 WWDT Interrupt Enable Register (WWDT_IER) *143*
10.4.5 WWDT Interrupt Status Register (WWDT_ISR) *144*
10.4.6 WWDT Prescaler Register (WWDT_PSC) *144*

11 CLOCK MANAGE UNIT (CMU) **146**

11.1 INTRODUCTION 146
11.2 CLOCK TREE 147
11.2.1 Introduction for SYSCLK switching *148*
11.2.2 Introduction for main clocks *148*
11.2.3 Bus clocks and operating clocks for peripherals *148*
11.2.4 Peripheral clock in sleep mode *150*
11.2.5 LSCLK switching logic *150*
11.3 HIGH FREQUENCY RC OSCILLATORS (RCHF) 151
11.3.1 Introduction *151*
11.3.2 Software control description *151*
11.4 MEDIUM FREQUENCY RC OSCILLATORS (RCMF) 152
11.4.1 Introduction *152*
11.5 HIGH PRECISION LOW POWER RC OSCILLATORS (LPOSC) 153
11.5.1 Introduction *153*
11.6 LOW FREQUENCY CRYSTAL OSCILLATION CIRCUITS (XTLF) 153
11.6.1 Introduction *153*
11.6.2 Software control description *154*
11.6.3 Fail detection *154*
11.7 HIGH FREQUENCY CRYSTAL OSCILLATION CIRCUITS (XTHF) 154
11.7.1 Introduction *154*
11.7.2 Software control description *154*
11.7.3 Fail detection(HFDET) *155*
11.8 PHASE LOCKED LOOP (PLL) 155
11.8.1 Introduction *155*
11.8.2 Software control description *155*
11.9 CLOCK SOURCE IN LOW POWER MODE 156
11.10 CLOCK SELECTION AFTER WAKE UP FROM SLEEP 156
11.11 REGISTER 157
11.11.1 Lockup Reset Control Register (RCC_LKPCR) *157*

11.11.2 Software Reset Register (RCC_SOFTRST) 158
11.11.3 ResetFlag Register (RCC_RSTFR) 158
11.11.4 System Clock Control Register (RCC_SYSCLKCR) 159
11.11.5 RCHF Control Register (RCC_RCHFCR) 161
11.11.6 RCHFTrim Register (RCC_RCHFTR) 161
11.11.7 PLL Control Register (RCC_PLLCR) 162
11.11.8 LPOSC Control Register (RCC_LPOSCCR) 163
11.11.9 LPOSC Trim Register (RCC_LPOSCTR) 163
11.11.10 XTLF Control Register (RCC_XTLFCR) 164
11.11.11 Peripheral bus Clock Control Register1 (RCC_PCLKCR1) 165
11.11.12 Peripheral bus Clock Control Register2 (RCC_PCLKCR2) 166
11.11.13 Peripheral bus Clock Control Register3 (RCC_PCLKCR3) 167
11.11.14 Peripheral bus Clock Control Register4 (RCC_PCLKCR4) 168
11.11.15 LSCLK Select Register (RCC_LSCLKSEL) 170
11.11.16 AHB Master Control Register (RCC_AHBMCR) 170
11.11.17 Peripheral Reset Enable Register (RCC_PRSTEN) 171
11.11.18 AHB Peripherals Reset Control Register (RCC_AHBRSTCR) 171
11.11.19 APB Peripherals Reset Control Register1 (RCC_APBRSTCR1) 172
11.11.20 APB Peripherals Reset Control Register2 (RCC_APBRSTCR2) 174
11.11.21 XTHF Control Register (RCC_XTHFCR) 176
11.11.22 RCMF Control Register (RCC_RCMFCR) 177
11.11.23 RCMF Trim Register (RCC_RCMFTR) 177
11.11.24 Peripheral Operation Clock Control Register1 (RCC_OPCCR1) 178
11.11.25 Peripheral Operation Clock Control Register2 (RCC_OPCCR2) 180

12 OSCILLATION FAIL DETECTION (FDET) 182

12.1 LOW FREQUENCY FAIL DETECTION 182
12.2 HIGH FREQUENCY FAIL DETECTION 182
12.3 REGISTER 183
12.3.1 Fail detection interrupt enable register (FDET_IER) 183
12.3.2 Fail detection interrupt flag register（FDET_ISR） 183

13 SUPPLY VOLTAGE DETECTION (SVD) 185

13.1 INTRODUCTION 185
13.2 BLOCK DIAGRAM 185
13.3 PIN DEFINITION 186
13.4 FUNCTIONAL DESCRIPTION 186
13.5 INTERMITTENT ENABLE MODE 188
13.6 EXTERNAL VOLTAGE DETECTION 188
13.7 DETECTION THRESHOLD 189
13.8 REGISTER 193
13.8.1 SVD Config Register (SVD_CFGR) 193
13.8.2 SVD Control Register (SVD_CR) 194
13.8.3 SVD Interrupt Enable Register (SVD_IER) 195
13.8.4 SVD State and Flag Register (SVD_ISR) 195

13.8.5 SVD Reference Voltage Select Register (SVD_VSR) 196

14 AES ALGORITHM MODULE (AES) 198

14.1 FUNCTION DESCRIPTION 198
14.2 OPERATION MODE 198
14.3 AES DATA STREAM PROCESSING MODES 199
14.3.1 ECB mode 199
14.3.2 CBC mode 200
14.3.3 Suspend mode 202
14.3.4 CTR mode 202
14.3.5 Suspend mode under CTR mode 204
14.3.6 GCM mode 204
14.3.7 MultH mode 207
14.3.8 Recommended GCM Process 208
14.4 DATA TYPE 209
14.5 OPERATION SEQUENCE 211
14.5.1 Mode 1: Encryption 211
14.5.2 Mode 2: Key expansion 212
14.5.3 Mode 3: Decryption 213
14.5.4 Mode 4: Key expansion + Decryption 213
14.5.5 Using the MultH module 214
14.6 DMA INTERFACE 215
14.6.1 MultH Module Interfaces with DMA 216
14.7 ERROR FLAGS 216
14.8 REGISTER 217
14.8.1 AES Control Register (AES_CR) 217
14.8.2 AES Interrupt Enable Register (AES_IER) 219
14.8.3 AES Interrupt Status Register (AES_ISR) 219
14.8.4 AES Data Input Register (AES_DIR) 220
14.8.5 AES Output Register (AES_DOR) 221
14.8.6 AES Key Register x (AES_KEYx) 221
14.8.7 AES Initial Vector Register x (AES_IVRx) 223

15 TRUE RANDOM NUMBER GENERATOR (TRNG) 224

15.1 INTRODUCTION 224
15.2 FUNCTIONAL DESCRIPTION 224
15.2.1 Random number generation 224
15.2.2 Operating clock 225
15.2.3 Random number read 225
15.2.4 CRC calculation 225
15.3 REGISTER 227
15.3.1 Random Number Generator Control Register (RNGCTL_CR) 227
15.3.2 Random Number Generator Data Output Register (RNG_DOR) 228
15.3.3 Random Number Generator Status Register (RNG_SR) 228
15.3.4 CRC Control Register (RNG_CRCCR) 229

15.3.5 CRC Data Input Register (RNG_CRCDIR) *229*
15.3.6 CRC Status Register (RNG_CRCSR) *230*

16 COMPARATOR (COMP) **231**

16.1 INTRODUCTION 231
16.2 BLOCK DIAGRAM 231
16.3 PIN DEFINITION 232
16.4 FUNCTION DESCRIPTION 232
16.4.1 Basic Functions *232*
16.4.2 Clock and reset *233*
16.4.3 Interrupt and trigger signal output *233*
16.4.4 Comparator output connection *234*
16.5 REGISTER 236
16.5.1 COMP Control Register 1（COMP_CR1） *236*
16.5.2 COMP Control Register 2（COMP_CR2） *238*
16.5.3 Comparator Control Register（COMP_ICR） *238*
16.5.4 Comparator Interrupt Status Register（COMP_ISR） *240*

17 HARDWARE DIVIDER (HDIV) **241**

17.1 INTRODUCTION 241
17.2 WORK PROCESS 241
17.3 REGISTER 242
17.3.1 Dividend register (HDIV_END) *242*
17.3.2 Divisor Register (HDIV_SOR) *242*
17.3.3 Quotient Register (HDIV_QUOT) *243*
17.3.4 Remainder Register (HDIV_REMD) *243*
17.3.5 Status Register (HDIV_SR) *244*

18 I²C-SMBUS **245**

18.1 INTRODUCTION 245
18.2 FUNCTION 245
18.3 I²C-SMB MODULE DIAGRAM 246
18.3.1 I²C-SMB module diagram *246*
18.4 PIN DEFINITION 247
18.5 CLOCK AND RESET 247
18.5.1 Communication process ..... *错误!未定义书签。*
18.5.2 Interface timing description ..... *错误!未定义书签。*
18.5.3 Bus timing parameter table ..... *错误!未定义书签。*
18.6 INTERFACE TIMING 248
18.6.1 Communication Process *248*
18.6.2 Interface Timing Description *249*
18.6.3 Bus Timing Parameters Table *250*
18.7 I²C FUNCTION 252
18.7.1 Preset salve address table *252*
18.7.2 I²C Initialization process *253*

18.7.3 I²C master function 255
18.7.4 I²C slave function 276
18.8 SMBUS FUNCTION DESCRIPTION 285
18.8.1 Overview 285
18.8.2 SMBus initialization process 288
18.8.3 SMBus Master Function 289
18.8.4 SMBus Timeout Function 292
18.8.5 CRC-8 293
18.9 REGISTER 294
18.9.1 I2C_SMB control register 1（I2CSMB_CR1） 294
18.9.2 I2C_SMB control register 2（I2CSMB_CR2） 296
18.9.3 I2C_SMB Interrupt Control Register（I2CSMB_IER） 296
18.9.4 I2C_SMB Status And Flag Registers（I2CSMB_ISR） 297
18.9.5 I2C_SMB Baud Rate Setting Register（I2CSMB_BGR） 299
18.9.6 I2C_SMB Host Timing Control Register（I2CSMB_TIMINGR） 299
18.9.7 I2C_SMB Timeout Register（I2CSMB_TOR） 300
18.9.8 I2C_SMB Receive Data Register（I2CSMB_RXDR） 301
18.9.9 I2C_SMB Transmit Data Register（I2CSMB_TXDR） 301
18.9.10 I2C_SMB Slave Address Register（I2CSMB_SADR） 302
18.9.11 CRC Data Register（I2CSMB_CRC_DR） 302
18.9.12 CRC LFSR Register（I2CSMB_CRC_LFSR） 303
18.9.13 CRC Polynomial Register（I2CSMB_CRC_POLY） 304

19 I²C 305

19.1 INTRODUCTION 305
19.2 I²C BLOCK DIAGRAM 305
19.3 CLOCK STRUCTURE 306
19.4 INTERFACE TIMING 307
19.4.1 Interface Timing diagram 307
19.4.2 Interface Timing Description 308
19.5 I²C WORK MODE 309
19.6 I²C SLAVE ADDRESS FORMAT 310
19.7 I²C INITIALIZATION 310
19.7.1 IO configuration 310
19.7.2 Master baud rate configuration 311
19.7.3 Slave input analog filtering and output delay 311
19.8 I²C MASTER FUNCTION 311
19.8.1 7 bit addressing 312
19.8.2 10 bit addressing 317
19.8.3 DMA 320
19.8.4 SCL Clock Stretching 323
19.8.5 Timeout function 323
19.8.6 Programmable Timing 323
19.9 I2C SLAVE FUNCTION 325
19.9.1 Slave adressing 325

19.9.2 Slave sends data 325
19.9.3 Slave receive data 326
19.9.4 Slave low-power receiving wake-up 328
19.9.5 DMA 328
19.9.6 Slave timing 330
19.10 REGISTER 332
19.10.1 I2C Master Config Register (I2C_MSPCFGR) 332
19.10.2 I2C Master Control Register (I2C_MSPCR) 333
19.10.3 I2C Master Interrupt Enable Register (I2C_MSPIER) 334
19.10.4 I2C Master Interrupt Status Register (I2C_MSPISR) 335
19.10.5 I2C Master Status Register (I2C_MSPSR) 336
19.10.6 I2C Master Baud Rate Generation Register (I2C_MSPBGR) 337
19.10.7 I2C Master Transfer Buffer Register (I2C_MSPBUF) 338
19.10.8 I2C Master Timing Control Register (I2C_MSPTCR) 338
19.10.9 I2C Master Time Out Register (I2C_MSPTOR) 339
19.10.10 I2C Slave Control Register (I2C_SSPCR) 340
19.10.11 I2C Slave Interrupt Enable Register (I2C_SSPIER) 341
19.10.12 I2C Slave Interrupt Status Register (I2C_SSPSR) 341
19.10.13 I2C Slave Status Register (I2C_SSPSR) 342
19.10.14 I2C Slave Transfer Buffer Register (I2C_SSPBUF) 343
19.10.15 I2C slave Address Register (I2C_SSPADR) 344

20 UART0/1/4/5 346

20.1 INTRODUCTION 346
20.2 UART BLOCK DIAGRAM 347
20.3 PIN DEFINITION 348
20.4 UART MODE 349
20.5 UART CHARACTER FORMAT 349
20.6 UART FUNCTION DESCRIPTION 350
20.6.1 Clock structure 350
20.6.2 Bit Receiving Sampling 350
20.6.3 Data Transmission 351
20.6.4 Data Reception 353
20.6.5 Low Power Hibernation Wakeup 354
20.6.6 Use DMA for UART communication 354
20.6.7 Transmission completion interrupt in DMA mode 355
20.7 BAUD RATE GENERATION 356
20.7.1 Baud rate generation 356
20.7.2 Adaptive Baud Rate Generation 357
20.8 INFRARED MODULATION 357
20.9 RECEIVE TIMEOUT 358
20.10 TRANSMIT DELAY 358
20.11 REGISTER 358
20.11.1 Infrared Modulation Register（UART_IRCR） 359
20.11.2 UARTx Control status register（UARTx_CSR） 360

20.11.3 UARTx Interrupt enable register（UARTx_IER） 361
20.11.4 UARTx Interrupt flag register (UARTx_ISR) 362
20.11.5 UARTx Timeout and delay registers（UARTx_TODR） 363
20.11.6 UARTx Receive Buffer Register（UARTx_RXBUF） 363
20.11.7 UARTx Transmit Buffer Register（UARTx_TXBUF） 364
20.11.8 UATRx Baud Rate Generate Register（UARTx_BGR） 364

21 ENHANCED UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER (UART2) 366

21.1 INTRODUCTION 366
21.2 UART2 BLOCK DIAGRAM 367
21.3 PIN DEFINITION 368
21.4 UART TYPE DISTINCTION 368
21.5 UART CHARACTER FORMAT 368
21.5.1 UART Mode 368
21.6 UART2 FUNCTION DESCRIPTION 369
21.6.1 Clock Structure 369
21.6.2 Bit Receiving Sampling 370
21.6.3 Data Transmission 371
21.6.4 Data Reception 373
21.6.5 LOW Power Hibernation Wake-up 373
21.6.6 Half duplex single line communication 374
21.7 BAUD RATE GENERATION 375
21.7.1 Baud Rate Generation 375
21.7.2 Adaptive Baud Rate Generation 376
21.8 INFRARED MODULATION 376
21.9 RECEIVE TIMEOUT 377
21.10 TRANSMIT DELAY 377
21.11 LIN BUS COMMUNICATION FUNCTION 377
21.11.1 Introduction 377
21.11.2 LIN Master Operation 380
21.11.3 Slave Operation 381
21.12 REGISTER 383
21.12.1 Infrared Modulation Register (UART_IRCR) 383
21.12.2 UART2 Control status register (UART2_CSR) 384
21.12.3 UART2 Interrupt enable register (UART_IER) 385
21.12.4 UART2 Interrupt flag register (UART2_ISR) 386
21.12.5 UART2 Timeout and delay register (UART2_TODR) 388
21.12.6 UART2 Receive Buffer Register (UART2_RXBUF) 388
21.12.7 UART2 Transmit Buffer Register (UART2_TXBUF) 389
21.12.8 UART2 Baud Rate Generate Register (UART2_BGR) 390
21.12.9 UART2 Model Register (UART2_MCR) 391
21.12.10 UART2 LIN Control Register (UART2_LINCR) 391
21.12.11 UART2 LIN Baud Rate Synchronized Register (UART_LINBSR) 393

22 LOW POWER UART (LPUART) 394

22.1 INTRODUCTION 394
22.2 BLOCK DIAGRAM 395
22.3 PIN DEFINITION 396
22.4 CLOCKS 396
22.5 CHARACTER DESCRIPTION 397
22.6 FUNCTIONAL DESCRIPTION 398
22.6.1 Bit sampling 398
22.6.2 Data reception procedure 398
22.6.3 Data transmit procedure 399
22.6.4 Use DMA for data Receiving and Transmitting 399
22.6.5 Data receiving and wake up in Sleep mode 399
22.6.6 DMA Data Receiving and Transmitting in LPRUN mode 400
22.6.7 Transmission completion interrupt in DMA mode 400
22.7 REGISTER 402
22.7.1 LPUARTx Control Status Register (LPUARTx_CSR) 402
22.7.2 LPUARTx Interrupt Enable Register (LPUARTx_IER) 404
22.7.3 LPUARTx Interrupt Status Register (LPUARTx_ISR) 405
22.7.4 LPUARTx Baud rate Modulation Register (LPUARTx_BMR) 406
22.7.5 LPUARTx Receive Buffer (LPUARTx_RXBUF) 407
22.7.6 LPUARTx Transmit Buffer Register (LPUARTx_TXBUF) 407
22.7.7 LPUARTx Data Matching Register (LPUARTx_DMR) 408

23 SPI 409

23.1 INTRODUCTION 409
23.2 BLOCK DIAGRAM 410
23.3 PIN DEFINITION 410
23.4 INTERFACE TIMING 411
23.4.1 CPHA=0 411
23.4.2 CPHA=1 411
23.4.3 4-wire half-duplex mode (master) 412
23.5 FUNCTIONAL DESCRIPTION 415
23.5.1 I/O Configuration 415
23.5.2 Full-duplex data communication 416
23.5.3 TX-ONLY mode 418
23.5.4 RX-ONLY mode 418
23.5.5 Master SSN Control 418
23.5.6 Data conflicts 420
23.5.7 SPI Transceiver via DMA 420
23.6 REGISTER 422
23.6.1 SPIx Control Register 1 (SPIx_CR1) 422
23.6.2 SPIx Control Register 2 (SPIx_CR2) 423
23.6.3 SPIx Control Register 3 (SPIx_CR3) 426
23.6.4 SPIx Interrupt Enable Register (SPIx_IER) 426
23.6.5 SPIx Interrupt Status Register (SPIx_ISR) 428
23.6.6 SPIx Transmit Buffer (SPIx_TXBUF) 429

23.6.7 SPIx Receive Buffer (SPIx_RXBUF) 429

24 ISO7816 430

24.1 INTRODUCTION 430
24.2 BLOCK DIAGRAM 430
24.3 INTERFACE TIMING 431
24.4 FUNCTIONAL DESCRIPTION 432
24.4.1 Data receive 432
24.4.2 Data transmission 432
24.4.3 DMA control 7816 for sending and receiving data 433
24.5 REGISTER 435
24.5.1 U7816 Control Register (U7816_CR) 435
24.5.2 U7816 Frame Format Register（U7816_FFR） 436
24.5.3 U7816 Extra Guard Time Register（U7816_EGTR） 437
24.5.4 U7816 Prescaler Register（U7816_PSC） 437
24.5.5 U7816 Baud rate Generator Register（U7816_BGR） 438
24.5.6 U7816 Receive Buffer（U7816_RXBUF） 439
24.5.7 U7816 Transmit Buffer（U7816_TXBUF） 439
24.5.8 U7816 Interrupt Enable Register（U7816_IER） 440
24.5.9 U7816 Interrupt Status Register（U7816_ISR） 440

25 DIRECT MEMORY ACCESS CONTROLLER (DMA) 443

25.1 INTRODUCTION 443
25.2 PRINCIPLE OF OPERATION 444
25.3 BLOCK DIAGRAM 445
25.4 WORKFLOW 445
25.5 ACCESS BANDWIDTH 447
25.6 CHANNEL CONTROL 447
25.6.1 DMA Request Mapping 447
25.6.2 Channel Priority 448
25.6.3 Definition of transfer direction 449
25.6.4 Loop Mode 449
25.7 REGISTER 450
25.7.1 DMA Global Control Register (DMA_GCR) 450
25.7.2 Channel X Control Register (DMA_CHxCR) 451
25.7.3 Channel X Memory Address Register (DMA_CHxMAD) 452
25.7.4 Channel & Control Register (DMA_CH7CR) 453
25.7.5 Channel 7 Flash Address Register (DMA_CH7FLSAD) 454
25.7.6 Channel 7 RAM Address Register (DMA_CH7RAMAD) 455
25.7.7 DMA Interrupt Status Register (DMA_ISR) 455

26 CYCLIC REDUNDANCY CHECK CALCULATION UNIT (CRC) 457

26.1 INTRODUCTION 457
26.2 OPERATION FLOW 458
26.3 GOLDEN DATA 459

26.4 DMA INTERFACE 459
26.5 FLASH DATA INTEGRITY CHECK 460
26.6 REGISTER 461
26.6.1 CRC Data Register (CRC_DR) 461
26.6.2 CRC Control Register (CRC_CR) 462
26.6.3 CRC LFSR Register (CRC_LFSR) 463
26.6.4 CRC Output XOR Register (CRC_XOR) 464
26.6.5 CRC Polynomial Register (CRC_POLY) 464

27 ADVANCED TIMER ARRAY (ATIM) 466

27.1 FUNCTION DESCRIPTION 466
27.2 MAIN CHARACTERISTICS 466
27.3 BLOCK DIAGRAM 467
27.4 FUNCTION DESCRIPTION 468
27.4.1 Timing unit 468
27.4.2 Timer operating mode 469
27.4.3 Repeat Counter 478
27.4.4 Preload register 479
27.4.5 Counter operating clock 479
27.4.6 Internal trigger signal（ITRx） 485
27.4.7 Capture/Compare channel 486
27.4.8 Input capture mode 487
27.4.9 Software force output 490
27.4.10 Output compare mode 491
27.4.11 PWM output 493
27.4.12 Complementary output and dead zone insertion 495
27.4.13 Brake function 496
27.4.14 Complementary output channel signal state logic table 498
27.4.15 6-step PWM output 499
27.4.16 Single pulse output 500
27.4.17 External event clears OCxREF 503
27.4.18 Encoder interface mode 504
27.4.19 TIM slave mode 506
27.4.20 DMA access 509
27.4.21 DMA Burst 511
27.4.22 Input XOR function 512
27.4.23 Debug mode 512
27.5 REGISTER 513
27.5.1 ATIM Control register 1（ATIM_CR1） 513
27.5.2 ATIM Control register 2（ATIM_CR2） 514
27.5.3 ATIM Slave mode control register（ATIM_SMCR） 516
27.5.4 ATIM DMA and interrupt enable register（ATIM_DIER） 517
27.5.5 ATIMS tatus register（ATIM_ISR） 519
27.5.6 ATIM event generation register（ATIM_EGR） 520
27.5.7 ATIM Capture/compare mode register1（ATIM_CCMR1） 521

27.5.8 ATIM Capture/compare mode register2（ATIM_CCMR2）.....526
27.5.9 ATIM Capture/compare enable register（ATIM_CCER）.....528
27.5.10 ATIM Counter register（ATIM_CNT）.....529
27.5.11 ATIM Prescaler register（ATIM_PSC）.....530
27.5.12 ATIM Auto-reload register（ATIM_ARR）.....530
27.5.13 ATIM Repeat count register（ATIM_RCR）.....531
27.5.14 ATIM Capture/compare register1（ATIM_CCR1）.....531
27.5.15 ATIM Capture/compare register2（ATIM_CCR2）.....532
27.5.16 ATIM Capture/compare register3（ATIM_CCR3）.....532
27.5.17 ATIM Capture/compare register4（ATIM_CCR4）.....533
27.5.18 ATIM Brake and dead zone control register（ATIM_BDTR）.....533
27.5.19 ATIM DMA Control register（ATIM_DCR）.....535
27.5.20 ATIM DMA access register（ATIM_DMAR）.....537
27.5.21 ATIM Brake input control register（ATIM_BKCR）.....537

28 GENERAL TIMER ARRAY (GPTIM).....539

28.1 INTRODUCTION.....539
28.2 MAIN CHARACTERISTICS.....539
28.3 BLOCK DIAGRAM.....540
28.4 FUNCTION DESCRIPTION.....541
28.4.1 Timing unit.....541
28.4.2 Timer operating mode.....543
28.4.3 Counter operating clock.....551
28.4.4 Capture of internal trigger signal (ITRx).....558
28.4.5 Capture/Compare channel.....559
28.4.6 Input capture mode.....561
28.4.7 Software Force output.....562
28.4.8 Output compare mode.....563
28.4.9 PWM input mode.....564
28.4.10 Single pulse output.....565
28.4.11 External event clear OCxREF.....567
28.4.12 Encoder interface mode.....568
28.4.13 GPTIM slave mode.....569
28.4.14 DMA access.....573
28.4.15 DMA Burst.....574
28.4.16 Input XOR function.....574
28.4.17 Debug mode.....575
28.5 REGISTER.....575
28.5.1 GPTIMx Control register 1（GPTIMx_CR1）.....576
28.5.2 GPTIMx Control register 2（GPTIMx_CR2）.....577
28.5.3 GPTIMx Slave mode control register（GPTIMx_SMCR）.....578
28.5.4 GPTIMx DMA and interrupt enable register（GPTIMx_DIER）.....580
28.5.5 GPTIMx Status register（GPTIMx_ISR）.....582
28.5.6 GPTIMx Event generation register（GPTIMx_EGR）.....583
28.5.7 GPTIMx Capture/compare mode register1（GPTIMx_CCMR1）.....584

28.5.8 GPTIMx Capture/compare mode register2（GPTIMx_CCMR2）.....587
28.5.9 GPTIMx Capture/compare enable register（GPTIMx_CCER）.....590
28.5.10 GPTIMxCounter register（GPTIMx_CNT）.....591
28.5.11 GPTIMx Prescaler register（GPTIMx_PSC）.....592
28.5.12 GPTIMx Auto-reload register（GPTIMx_ARR）.....592
28.5.13 GPTIMx Capture/compare register1（GPTIMx_CCR1）.....593
28.5.14 GPTIMx Capture/compare register2（GPTIMx_CCR2）.....593
28.5.15 GPTIMx Capture/compare register3（GPTIMx_CCR3）.....594
28.5.16 GPTIMx Capture/compare register4（GPTIMx_CCR4）.....595
28.5.17 GPTIMx DMA control register（GPTIMx_DCR）.....595
28.5.18 GPTIMx DMA access register（GPTIMx_DMAR）.....597
28.5.19 GPTIMx ITR Select register（GPTIMx_ITRSEL）.....597

29 BASIC TIMER (BSTIM32).....599

29.1 INTRODUCTION.....599
29.2 MAIN FEATURES.....599
29.3 BLOCK DIAGRAM.....599
29.4 FUNCTIONAL DESCRIPTION.....600
29.4.1 Timing Unit.....600
29.4.2 Timer Operation Mode.....602
29.4.3 Counter Working Clock.....604
29.4.4 Debug mode.....605
29.5 REGISTER.....606
29.5.1 BSTIM Control Register1 (BSTIM_CR1).....606
29.5.2 BSTIM Control Register 2 (BSTIM_CR2).....607
29.5.3 BSTIM Interrupt Enable Register (BSTIM_IER).....608
29.5.4 BSTIM Interrupt Flag Register (BSTIM_ISR).....609
29.5.5 BSTIM Event Generation Register (BSTIM_EGR).....609
29.5.6 BSTIM Counter Register (BSTIM_CNT).....610
29.5.7 BSTIM Prescaler Register (BSTIM_PSC).....610
29.5.8 BSTIM Auto-reload Register (BSTIM_ARR).....611

30 LOW POWER TIMER (LPTIM32).....612

30.1 INTRODUCTION.....612
30.2 BLOCK DIAGRAM.....613
30.3 TIMER FUNCTION.....613
30.3.1 General Timer.....613
30.3.2 External Pulse Trigger Counting.....614
30.3.3 External Asynchronous Pulse Counting.....614
30.3.4 Timeout Mode.....615
30.4 CAPTURE/COMPARE FUNCTION.....616
30.4.1 32bit PWM.....616
30.4.2 Input Capture.....617
30.5 REGISTER.....619
30.5.1 LPTIM Config Register (LPTIM_CFGR).....619

30.5.2 LPTIM Counter Register (LPTIM_CNT) 621
30.5.3 LPTIM Capture/Compare Control and Status Register (LPTIM_CCSR) 621
30.5.4 LPTIM Auto-Reload Register (LPTIM_ARR) 623
30.5.5 LPTIM Interrupt Enable Register (LPTIM_IER) 623
30.5.6 LPTIM Interrupt Status Register (LPTIM_ISR) 624
30.5.7 LPTIM Control Register (LPTIM_CR) 625
30.5.8 LPTIM Capture/Compare Register1 (LPTIM_CCR1) 626
30.5.9 LPTIM Capture/Compare Register2 (LPTIM_CCR2) 626

31 REAL-TIME CLOCK (RTC) 628

31.1 INTRODUCTION 628
31.2 BLOCK DIAGRAM 628
31.3 WORKING PRINCIPLE 629
31.3.1 Low-power time base counter (LTBC) 629
31.3.2 LTBC digital calibration 630
31.3.3 BCD clock 631
31.3.4 RTC enable and disable 633
31.3.5 RTC time setting 633
31.3.6 RTC time reading 633
31.3.7 Leap Year determination 634
31.4 REGISTER 635
31.4.1 RTC Write Enable Register(RTC_WER) 635
31.4.2 RTC Interrupt Enable Register(RTC_IER) 636
31.4.3 RTC InterruptStatus Register(RTC_ISR) 637
31.4.4 BCD Format Time Second Registers(RTC_BCDSEC) 638
31.4.5 BCD Format Time MinuteRegisters(RTC_BCDMIN) 638
31.4.6 BCD Format Time HourRegisters(RTC_BCDHOUR) 639
31.4.7 BCD Format Time DayRegisters(RTC_BCDDAY) 639
31.4.8 BCD Format Time WeekRegisters(RTC_BCDWEEK) 640
31.4.9 BCD Format Time MonthRegisters(RTC_BCDMONTH) 640
31.4.10 BCD Format Time YearRegisters(RTC_BCDYEAR) 641
31.4.11 RTC AlarmRegisters(RTC_ALARM) 641
31.4.12 RTC Time Mark SelectRegister(RTC_TMSEL) 642
31.4.13 RTC Time AdjustRegister(RTC_ADJUST) 642
31.4.14 RTC Time AdjustSign Register(RTC_ADSIGN) 643
31.4.15 RTC Sub-Second Counter Register(RTC_SBSCNT) 643
31.4.16 RTC Backup Registers x (RTC_BKRx) 644

32 LCD DISPLAY 645

32.1 INTRODUCTION 645
32.2 BLOCK DIAGRAM 645
32.3 IO CONFIGURATION 647
32.4 FUNCTION DESCRIPTION 647
32.4.1 Operating Clock and Display Frame Rate 647
32.4.2 LCD Type A Scan Waveform 647

32.4.3 LCD Type B Scan Waveform 649
32.4.4 On-chip Resistor Drive Mode 650
32.4.5 Display Flick Function 650
32.4.6 Bias Voltage Adjustment 651
32.5 REGISTER 652
32.5.1 LCD Control Register (LCD_CR) 652
32.5.2 LCD Test Register (LCD_TEST) 654
32.5.3 Pin Output Data Register in Test Mode 655
32.5.4 LCD Frequency Control Register (LCD_FCR) 656
32.5.5 LCD Flick Time Register (LCD_FLKT) 656
32.5.6 LCD Interrupt Enable Register (LCD_IER) 657
32.5.7 LCD Interrupt Status Register (LCD_ISR) 657
32.5.8 LCD Data Buffer Register x (LCD_DATAx) 658
32.5.9 LCD COM Enable Register (LCD_COMEN) 664
32.5.10 LCD SEG Enable Register (LCD_SEGEN0) 665

33 ADC 666

33.1 INTRODUCTION 666
33.2 BLOCK DIAGRAM 666
33.3 INPUT CHANNEL 667
33.4 WORK SEQUENCE 667
33.5 FUNCTIONAL DESCRIPTION 669
33.5.1 Sampling value and actual voltage conversion 669
33.5.2 Temperature sensor 669
33.5.3 Programmable sampling time 670
33.5.4 Conversion mode 671
33.5.5 Conversion trigger 674
33.5.6 Oversampling and hardware averaging 676
33.5.7 ADC working clock 676
33.5.8 Data conflict and automatic waiting 676
33.5.9 DMA 677
33.5.10 Analog window watchdog（AWD） 684
33.5.11 ADC Calibration 685
33.6 LOW POWER MODE 686
33.7 REGISTER 687
33.7.1 ADC Interrupt and Status Register (ADC_ISR) 687
33.7.2 ADC Interrupt Enable Register (ADC_IER) 688
33.7.3 ADC Control Register (ADC_CR1) 688
33.7.4 ADC Calibration Control Register（ADC_CALR） 689
33.7.5 ADC Config Register (ADC_CFGR) 690
33.7.6 ADC Sampling Time Register (ADC_SMTR) 692
33.7.7 ADC Channel Enable Register (ADC_CHER) 693
33.7.8 ADC Data Register(ADC_DR) 694
33.7.9 ADC Analog Watchdog Threshold Register (ADC_HLTR) 695

34 GENERAL-PURPOSE I/OS 697

34.1 INTRODUCTION 697
34.2 GPIO FUNCTIONAL DESCRIPTION 697
34.2.1 GPIO，Input and Output Enable，Controlled Pull-Up Resister，Controlled Open-Drain Output 698
34.2.2 False Open-Drain Output，No Pull-up Resistor 699
34.3 IO FUNCTION DEFINITION 700
34.3.1 GPIO Input 700
34.3.2 GPIO Output 700
34.3.3 Digital Peripheral Functions 700
34.3.4 Analog Function 703
34.3.5 Analog Function Using External Crystal Pins 704
34.4 NRST PIN 704
34.5 WKUPX PINS 704
34.6 EXTERNAL PIN INTERRUPT (EXTI) 705
34.6.1 Function Description 705
34.6.2 Application guidelines 707
34.7 FAST GPIO OUTPUT 708
34.8 REGISTER 709
34.8.1 GPIOx Input Enable Register (GPIOx_INEN) 710
34.8.2 GPIOx Pull-up Enable Register (GPIOx_PUEN) 711
34.8.3 GPIOx Open-Drain Enable Register (GPIOx_ODEN) 711
34.8.4 GPIOx Function Control Register (GPIO_FCR) 712
34.8.5 GPIOx Data Output Register (GPIOx_DO) 714
34.8.6 GPIOx Data Set Register (GPIOx_DSET) 715
34.8.7 GPIOx Data Reset Register (GPIOx_DRST) 716
34.8.8 GPIOxb Data Input Register (GPIOx_DIN) 716
34.8.9 GPIOx Digital Funnction Select (GPIOx_DFS) 717
34.8.10 GPIOx Analog Channel Enable Register (GPIOxANEN) 718
34.8.11 External Interrupt Input Select Register (GPIO_EXTISEL) 718
34.8.12 External Interrupt Edge Select and Enable Register (GPIO_EXTIEDS) 721
34.8.13 External interrupt Digital Filter Register (GPIO_EXTIDF) 723
34.8.14 External Interrupt and Status Register (GPIO_EXTLLSR) 723
34.8.15 External Interrupt Data Input Register (GPIO_EXTIDI) 724
34.8.16 Frequency Output Select Register (GPIO_FOUTSEL) 725
34.8.17 Wakeup Enable Register (GPIO_PINWKEN) 726

35 SERIAL WIRE DEBUG (SWD) 728

35.1 INTRODUCTION 728
35.2 PROGRAMMER INSTRUCTION 728

36 DEVICE SIGNATURE 729

36.1 MEMORY CAPACITY QUERY REGISTER 729
36.2 DEVICE UID 729

37 DEBUG SUPPORT 731

37.1 INTRODUCTION 731
37.2 DEBUG PIN 731
37.2.1 SWD Pin 731
37.2.2 Pull-up Resistance 732
37.3 SW INTERFACE PROTOCOL 732
37.3.1 SW Protocol Introduction 732
37.3.2 SW Protocol sequence 732
37.3.3 SW-DP ID code 733
37.3.4 Host Read 734
37.3.5 Host Write 734
37.4 SWD-DP REGISTER 735
37.4.1 Register List 735
37.5 CORE DEBUG REGISTER 735
37.6 DEBUG-RELATED CONFIGURATION 736

REVISION HISTORY 737

CONTACT US 738

List of tables

TABLE 1-1 FM33LE0XXA DEVICE LINEUP 35
TABLE 2-1 FM33LE0XXA PIN DESCRIPTIONS 43
TABLE 3-1 FM33LE0XXA ABSOLUTE MAXIMUM RATINGS 53
TABLE 3-2 FM33LE0XXA GENERAL OPERATING CONDITIONS ..... 错误!未定义书签。
TABLE 3-3 ACTIVE CURRENT CHARACTERISTICS 55
TABLE 3-4 LP ACTIVE CURRENT CHARACTERISTICS ..... 错误!未定义书签。
TABLE 3-5 LP RUN CURRENT CHARACTERISTICS 55
TABLE 3-6 SLEEP CURRENT CHARACTERISTICS 55
TABLE 3-7 DEEPSLEEP CURRENT CHARACTERISTICS 56
TABLE 3-8 RESET AND SUPPLY DETECTION 57
TABLE 3-9 HIGH PRECISION REFERENCE CHARACTERISTIC 58
TABLE 3-10 WAKE UP TIME CHARACTERISTICS 59
TABLE 3-11 LOW-FREQUENCY CRYSTAL CHARACTERISTICS 60
TABLE 3-12 HIGH-FREQUENCY CRYSTAL CHARACTERISTICS 61
TABLE 3-13 INTERNAL HIGH-FREQUENCY RCOSCILLATOR CHARACTERISTICS 63
TABLE 3-14 INTERNAL MIDDLE-FREQUENCY RCOSCILLATOR CHARACTERISTICS 63
TABLE 3-15 INTERNAL LOW-FREQUENCY RCOSCILLATOR CHARACTERISTICS 63
TABLE 3-16 PLLCHARACTERISTICS 64
TABLE 3-17 ADC CHARACTERISTICS 66
TABLE 3-18 ADC INPUT IMPEDANCE 68
TABLE 3-19 TEMPERATURE SENSOR PARAMETER 68
TABLE 3-20 ANALOG COMPARATOR CHARACTERISTICS 69
TABLE 3-21 FLASH CHARACTERISTICS 69
TABLE 3-22 GENERAL I/O CHARACTERISTICS 70
TABLE 3-23 NRST PIN CHARACTERISTICS 71
TABLE 3-24 PIN AC CHARACTERISTICS 71
TABLE 3-25 LCD ON-CHIP RESISTOR VOLTAGE DIVIDER ..... 错误!未定义书签。
TABLE 4-1 POWER CONSUMPTION MODE 75
TABLE 4-2 CONSUMPTION MODE AND AVAILABLE SYSTEM CLOCK 77
TABLE 4-3 WAKE-UP EVENTS AND APPLICATIONS 81
TABLE 6-1 FM33LC0XXA PROCESSOR CONFIGURATION 93
TABLE 6-2 CORTEX-M0 CORE REGISTERS 94
TABLE 6-3 FM33LC0XXA INTERRUPT VECTOR TABLE 96
TABLE 6-4 HARDFAULT TYPES 96
TABLE 7-1 PERIPHERALS ADDRESS ASSIGNMENT 105
TABLE 7-2 DATA CONTENT DEFINITION 107
TABLE 7-3 USER OPTION BYTE DEFINITION 107
TABLE 7-4 LOCK INFORMATION DEFINITION 108
TABLE 7-5 LOCK BIT AND FLASH ADDRESS CORRESPONDING TABLE 108
TABLE 7-6 LOCK BIT PERMISSION DEFINITION 117
TABLE 7-7 FLASH ACCESS AUTHORIZATION TABLE 118

TABLE 9-1 IWDT OVERFLOW PERIODIC TABLE 133
TABLE 10-1 WWDT OVERFLOW PERIODIC TABLE 141
TABLE 11-1 SYSTEM CLOCK SWITCHING CONDITIONS 148
TABLE 11-2 MAIN SYSTEM CLOCKS DESCRIPTION 148
TABLE 11-3 PERIPHERAL MODULE OPERATING CLOCKS AND BUS CLOCKS 150
TABLE 11-4 RCHF CALIBRATION DATA 152
TABLE 11-5 RCMF CALIBRATION DATA 152
TABLE 11-6 LPOSC CALIBRATION DATA 153
TABLE 11-7 CLOCK SOURCE IN LOW POWER MODE 156
TABLE 16-1 COMPARATOR PIN LIST 232
TABLE 18-1 I²C PIN LIST 247
TABLE 18-2 I²C INTERFACE TIMING REQUEST ..... 错误!未定义书签。
TABLE 18-3 I²C INTERFACE TIMING REQUIREMENTS 250
TABLE 18-4 I²C-SMBUS SALVE RESERVED ADDRESS DEFINITION 253
TABLE 18-5 SMBUS TIMEOUT EVENTS 286
TABLE 20-1 UART PIN LIST 348
TABLE 20-2 UART MODE 349
TABLE 20-3 UART DATA FRAME FORMAT 350
TABLE 20-4 THE STATUS OF THE INTERRUPT FLAG WHEN UART USES DMA TO SEND DATA 355
TABLE 20-5 COMMON CLOCK FREQUENCY BAUD RATE CALCULATION 356
TABLE 21-1 UART2 PIN LIST 368
TABLE 21-2 UART TYPE LIST 368
TABLE 21-3 UART DATA FRAME FORMAT 369
TABLE 21-4 COMMON CLOCK FREQUENCY BAUD RATE CALCULATION 376
TABLE 22-1 LPUART FRAME FORMAT 397
TABLE 22-2 LPUART BIT MODULATION FACTOR 398
TABLE 25-1 DMA CHANNEL MAPPING 448
TABLE 32-1 FRAME FREQUENCY CALCULATION FORMULA 647
TABLE 32-2 RELATIONSHIP BETWEEN TYPICAL FRAME FREQUENCY AND DF 647
TABLE 34-1 GPIO FUNCTION LOGIC DEFINITION 699
TABLE 34-2 FCR DEFINITION 700
TABLE 34-3 DIGITAL PERIPHERAL FUNCTION SELECTION TABLE 702
TABLE 34-4 EXTERNAL PIN INTERRUPT CONFIGURATION 706

List of figures

FIGURE 2-1 FM33LE0X5A LQFP48 PACKAGE....36
FIGURE 2-2 FM33LE0X3A LQFP32 PACKAGE....错误!未定义书签。
FIGURE 3-1 FM33LE0XX POWER SUPPLY SCHEME....52
FIGURE 3-2 FM33LC0X2M POWER SUPPLY SCHEME....错误!未定义书签。
FIGURE 3-3 THE EQUIVALENT CIRCUIT MODEL OF CRYSTAL OR CERAMIC OSCILLATOR....62
FIGURE 3-4 PARAMETER DESCRIPTION....64
FIGURE 3-5 ADC CHANNEL INPUT IMPEDANCE....67
FIGURE 4-1 CHIP POWER STRUCTURE....73
FIGURE 4-2 POWER CONSUMPTION MODE CONVERT....76
FIGURE 4-3 CONSUMPTION MODE AND SYSTEM CLOCK....77
FIGURE 7-1 SYSTEM BUS DIAGRAM....102
FIGURE 7-2 FM33LE04XA BUS ADDRESS....103
FIGURE 7-3 FM33LE02XA BUS ADDRESS....104
FIGURE 7-4 BOOTSWAP SCHEMATIC....109
FIGURE 7-5 STATE TRANSITION SCHEMATIC DIAGRAM....111
FIGURE 7-6 BOOTSWAP....114
FIGURE 8-1 RESET BLOCK DIAGRAM....127
FIGURE 8-2 POWER-ON/POWER-DOWN RESET DIAGRAM....128
FIGURE 8-3 POWER-ON/POWER-DOWN RESET SEQUENTIAL DIAGRAM....128
FIGURE 9-1 IWDT BLOCK DIAGRAM....132
FIGURE 9-2 IWDT WINDOW DIAGRAM....134
FIGURE 10-1 WWDT BLOCK DIAGRAM....139
FIGURE 10-2 WINDOW WATCHDOG TIMING DIAGRAM....141
FIGURE 11-1 CLOCK TREE DIAGRAM....147
FIGURE 13-1 LOW VOLTAGE DETECTION CIRCUIT DIAGRAM....185
FIGURE 13-2 LOW VOLTAGE DETECTION CIRCUIT DIAGRAM....186
FIGURE 13-3 INTERMITTENT OPERATION MODE OF POWER DETECTION CIRCUIT....187
FIGURE 13-4 SVS DETECTION USING EXTERNAL RESISTOR DIVIDER....188
FIGURE 13-5 SVS DETECTION USING INTERNAL RESISTOR DIVIDER....189
FIGURE 14-1 ECB MODE ENCRYPTION....199
FIGURE 14-2 ECB MODE DECRYPT....200
FIGURE 14-3 CBC MODE ENCRYPTION....201
FIGURE 14-4 CBC MODE DECRYPTION....201
FIGURE 14-5 SUSPEND MODE SEQUENCE....202
FIGURE 14-6 CTR MODE ENCRYPTION....203
FIGURE 14-7 CTR MODE DECRYPTION....204
FIGURE 14-8 STORAGE FORMAT OF 32-BIT COUNTER VALUE AND RANDOM NUMBER....204
FIGURE 14-9 GCM MODE ENCRYPTION....206
FIGURE 14-10 GCM MODE DECRYPTION....207
FIGURE 14-11 MULTH MODULE BLOCK DIAGRAM....208
FIGURE 14-12 DATA STORAGE FORMAT UNDER DIFFERENT DATA TYPE....211

FIGURE 14-13 OPERATION SEQUENCE OF MODE 1: ENCRYPTION 212
FIGURE 14-14 OPERATION SEQUENCE OF MODE 2: KEY EXPANSION 212
FIGURE 14-15 OPERATION SEQUENCE OF MODE 3: DECRYPTION 213
FIGURE 14-16 OPERATION SEQUENCE OF MODE 4: KEY EXPANSION + DECRYPTION 214
FIGURE 14-17 OPERATION SEQUENCE OF USING MULTH MODULE....215
FIGURE 14-18 DMA REQUEST AND DATA TRANSFER SEQUENCE WHILE INPUT....215
FIGURE 14-19 DMA REQUEST AND DATA TRANSFER SEQUENCE WHILE OUTPUT....216
FIGURE 15-1 TRUE RANDOM NUMBER GENERATION MODULE 224
FIGURE 15-2 CLOCKFOR TRNG 225
FIGURE 16-1 COMPARATOR CIRCUIT BLOCK DIAGRAM....231
FIGURE 16-2 SCHEMATIC DIAGRAM OF DIGITAL FILTERING WAVEFORM....233
FIGURE 16-3 COMPARATOR INTERRUPT GENERATION 234
FIGURE 16-4 COMPARATOR TRIGGER OUTPUT SIGNAL GENERATION....234
FIGURE 18-1 I²C-SMB MODULE DIAGRAM 246
FIGURE 18-2 I²C TIMING ....错误!未定义书签。
FIGURE 18-3 EFFECTIVE DATA TIMING....错误!未定义书签。
FIGURE 18-4 START AND STOP DEFINITION....错误!未定义书签。
FIGURE 18-5 OUTPUT ACK ....错误!未定义书签。
FIGURE 18-6 I²C TIMING PARAMETERS....错误!未定义书签。
FIGURE 18-7 I²C BUS PROTOCOL 248
FIGURE 18-8 BIT PROTOCOL 248
FIGURE 18-9 START & STOP CONDITION DEFINITION 248
FIGURE 18-10 ACK....249
FIGURE 18-11 I²C TIMING PARAMETERS LEGEND 251
FIGURE 18-12 FRAME FORMAT WHEN A MASTER WRITES DATA TO A 7-BIT ADDRESS SLAVE 256
FIGURE 18-13 I²C MASTER SOFTWARE SENDING DATA FLOW DIAGRAM 257
FIGURE 18-14 I²C MASTER SENDS DATA FLOW DIAGRAM TO 7-BIT ADDRESS SLAVE 258
FIGURE 18-15 FRAME FORMAT WHEN A MASTER READS DATA FROM A 7-BIT ADDRESSING SLAVE 258
FIGURE 18-16 I²C SOFTWARE RECEIVE DATA FLOW DIAGRAM....260
FIGURE 18-17 I²C READS DATA FLOW DIAGRAM FROM 7-BIT ADDRESS SLAVE 262
FIGURE 18-18 I²C MASTER RECEIVING DATA WAITING BIDIRECTIONAL DATA TRANSFER (COMBINED MODE)....262
FIGURE 18-19 FRAME FORMAT FOR BI-DIRECTIONAL DATA COMMUNICATION....263
FIGURE 18-20 10BIT ADDRESSING, THE MASTER WRITES DATA TO THE SLAVE....263
FIGURE 18-21 I²C SOFTWARE TRANSMIT FLOW DIAGRAM 265
FIGURE 18-22 10BIT ADDRESSING, MASTER READS DATA FROM THE SLAVE 266
FIGURE 18-23 I²C SOFTWARE SEND DATA FLOW DIAGRAM....267
FIGURE 18-24 I²C SOFTWARE SENDING DATA FLOW DIAGRAM....268
FIGURE 18-25 I²C MASTER DMA TRANSMIT FLOWCHART....269
FIGURE 18-26 I²C MASTER DMA RECEIVE FLOWCHART....272
FIGURE 18-27 I²C MULTI-MASTER CLOCK SYNCHRONIZATION 273
FIGURE 18-28 I2C MULTI-MASTER DATA ARBITRATION....274
FIGURE 18-29 MASTER TIMING CONTROL....275
FIGURE 18-30 I²C SLAVE DATA SENDING WAVEFORM 277
FIGURE 18-31 I²C SLAVE DATA RECEIVING WAVEFORM 278

FIGURE 18-32 I²C SLAVE DATA RECEIVING WAVEFORM (SCLSEN=0, RECEIVING OVERFLOW) 279
FIGURE 18-33 SLAVE ADDRESS MATCHING WAKE-UP 280
FIGURE 18-34 I²C SLAVE DMA RECEIVING FLOWCHART 282
FIGURE 18-35 I2C SLAVE DMA SEND FLOW 283
FIGURE 18-36 SDA OUTPUT DELAY WAVEFORM 284
FIGURE 18-37 TLOW:SEXT AND TLOW:MEXT WAVEFORMS 287
FIGURE 18-38 SLAVE ADDRESS MATCHING WAKE-UP 291
FIGURE 18-39 SLAVE START WAKE-UP 291
FIGURE 18-40 SMBUS BUS CLOCK TIMEOUT EVENT 292
FIGURE 18-41 HOST AND SLAVE TIMEOUT EVENTS 293
FIGURE 19-1 I²C BLOCK DIAGRAM 305
FIGURE 19-2 I²C BUS TIMING 307
FIGURE 19-3 DATA VALID TIMING 307
FIGURE 19-4 START & STOP CONDITION DEFINITION 307
FIGURE 19-5 OUTPUT ACK 308
FIGURE 19-6 I2C SLAVE RESERVED ADDRESSES DEFINITION 310
FIGURE 19-7 SLAVE SIGNAL FILTERING 311
FIGURE 19-8 FRAME FORMAT WHEN A MASTER WRITES DATA TO A 7-BIT ADDRESS SLAVE 312
FIGURE 19-9 I²C SOFTWARE SENDING DATA FLOW DIAGRAM 313
FIGURE 19-10 I2C MASTER SENDS DATA FLOW DIAGRAM TO 7-BIT ADDRESS SLAVE 314
FIGURE 19-11 FRAME FORMAT WHEN A MASTER READS DATA FROM A 7-BIT ADDRESSING SLAVE 314
FIGURE 19-12 I²C SOFTWARE RECEIVE DATA FLOW DIAGRAM 315
FIGURE 19-13 I²C READS DATA FLOW DIAGRAM FROM 7-BIT ADDRESS SLAVE 316
FIGURE 19-14 FRAME FORMAT FOR BI-DIRECTIONAL DATA COMMUNICATION 316
FIGURE 19-15 10BIT ADDRESSING, THE MASTER WRITES DATA TO THE SLAVE 317
FIGURE 19-16 I²C SOFTWARE TRANSMIT FLOW DIAGRAM 318
FIGURE 19-17 10BIT ADDRESSING, MASTER READS DATA FROM THE SLAVE 318
FIGURE 19-18 I²C SOFTWARE SEND DATA FLOW DIAGRAM 319
FIGURE 19-19 I²C SOFTWARE SEND DATA FLOW DIAGRAM 320
FIGURE 19-20 I²C MASTER DMA TRANSMIT FLOWCHART 321
FIGURE 19-21 I²C MASTER DMA RECEIVE FLOWCHART 322
FIGURE 19-22 MASTER TIMING CONTROL 324
FIGURE 19-23 SLAVE DATA SENDING WAVEFORM 326
FIGURE 19-24 SLAVE DATA RECEIVING WAVEFORM 327
FIGURE 19-25 SLAVE DATA RECEIVED WAVEFORM (SCLSEN=0, RECEIVED OVERFLOW) 328
FIGURE 19-26 I²C SLAVE DMA RECEIVING FLOWCHART 329
FIGURE 19-27 I²C SLAVE DMA SENDING FLOWCHART 330
FIGURE 19-28 SDA OUTPUT DELAY WAVEFORM 331
FIGURE 20-1 UART BLOCK DIAGRAM 347
FIGURE 20-2 UART CHARACTER FORMAT 349
FIGURE 20-3 BIT SAMPLING 351
FIGURE 20-4 UART ASYNCHRONOUS TRANSMISSION WAVEFORM 1 352
FIGURE 20-5 UART ASYNCHRONOUS TRANSMISSION WAVEFORM 2 353
FIGURE 20-6 UART ASYNCHRONOUS TRANSMISSION WAVEFORM 3 353

FIGURE 20-7 THE INFRARED MODULATION WAVEFORM 358
FIGURE 20-8 UART TRANSMIT DELAY 358
FIGURE 21-1 UART2 BLOCK DIAGRAM 367
FIGURE 21-2 UART CHARACTER FORMAT 369
FIGURE 21-3 BIT RECEIVE 16 TIMES SAMPLING 370
FIGURE 21-4 BIT RECEIVE 8 TIMES SAMPLING 370
FIGURE 21-5 UART ASYNCHRONOUS TRANSMIT WAVEFORM 1 371
FIGURE 21-6 UART ASYNCHRONOUS TRANSMISSION WAVEFORM 2 372
FIGURE 21-7 UART ASYNCHRONOUS TRANSMISSION WAVEFORM 3 372
FIGURE 21-8 THE INFRARED MODULATION WAVEFORM 377
FIGURE 21-9 UART TRANSMIT DELAY 377
FIGURE 21-10 LIN BUS TOPOLOGY FIGURE FIX 378
FIGURE 21-11 LIN BUS FRAME STRUCTURE FIGURE FIX 378
FIGURE 21-12 LIN BREAK FIELD FIGURE FIX 378
FIGURE 21-13 LIN SYNC FIELD FIGURE FIX 379
FIGURE 21-14 LIN FRAME ID FIGURE FIX 380
FIGURE 21-15 LIN BUS WAKE-UP SIGNAL 381
FIGURE 22-1 LPUART BLOCK DIAGRAM 395
FIGURE 22-2 CHARACTER FORMAT 397
FIGURE 23-1 BLOCK DIAGRAM OF SPI 410
FIGURE 23-2 SPI DATA/CLOCK TIMING DIAGRAM (CPHA=0) 411
FIGURE 23-3 SPI DATA/CLOCK TIMING DIAGRAM (CPHA=1) 412
FIGURE 23-4 -WIRE HALF-DUPLEX WRITE OPERATION 412
FIGURE 23-5 -WIRE HALF-DUPLEX READ OPERATION (NO DUMMY CYCLE) 413
FIGURE 23-6 -WIRE HALF-DUPLEX READ OPERATION (WITH DUMMY CYCLE) 414
FIGURE 23-7 SPI MASTER/SPI SLAVE INTERCONNECT 416
FIGURE 23-8 SPI SSN TIMING DIAGRAM (SSNM=1, CPHA=0) 418
FIGURE 23-9 SPI SSN TIMING DIAGRAM (SSNM=0) 419
FIGURE 24-1 ISO7816 BLOCK DIAGRAM 430
FIGURE 24-2 ISO7816 DATA FRAME STRUCTURE 431
FIGURE 24-3 ISO7816 DATA RECEIVE PROCESS 432
FIGURE 24-4 ISO7816 DATA TRANSMISSION PROCESS 433
FIGURE 25-1 DMA BLOCK DIAGRAM 445
FIGURE 25-2 DMA REGISTER CONFIGURATION 445
FIGURE 25-3 DMA WORKFLOW 447
FIGURE 26-1 CRC OPERATION FLOW 458
FIGURE 26-2 CRC THE DATA IN RAM BY DMA 459
FIGURE 27-1 ATIM BLOCK DIAGRAM 467
FIGURE 27-2 PRESCALER WAVEFORM FROM 1 TO 2 469
FIGURE 27-3 WAVEFORM OF PRESCALING FROM 1 TO 4 469
FIGURE 27-4 UPWARD COUNTING WAVEFORM, INTERNAL CLOCK NOT DIVIDED 470
FIGURE 27-5 UPWARD COUNTING WAVEFORM, INTERNAL CLOCK DIVIDED-BY-2 471
FIGURE 27-6 UPDATE EVENT WHEN ARPE=0 (ARR IS NOT PRELOADED) 471
FIGURE 27-7 UPDATE EVENT WHEN ARPE=1 (ARR PRELOAD) 472

FIGURE 27-8 DOWNWARD COUNTING WAVEFORM, INTERNAL CLOCK NOT DIVIDED 473
FIGURE 27-9 DOWNWARD COUNTING WAVEFORM, INTERNAL CLOCK DIVIDED-BY-2 473
FIGURE 27-10 UPDATE EVENT WHEN ARPE=0(ARR IS NOT PRELOADED) 474
FIGURE 27-11 UPDATE EVENT WHEN ARPE=1 (ARR PRELOAD) 474
FIGURE 27-12 TIMING DIAGRAM OF CENTER ALIGNED COUNTER, ATIM_PCS=0, ATIM_ARR=0X6 475
FIGURE 27-13 COUNTER TIMING DIAGRAM, UPDATE EVENT WHEN ARPE=1 (COUNTER UNDERFLOW) 476
FIGURE 27-14 COUNTER TIMING DIAGRAM, UPDATE EVENT WHEN ARPE=1 (COUNTER OVERFLOW) 476
FIGURE 27-15 EXAMPLE OF UPDATE RATE IN 15 DIFFERENT MODES, AND REGISTER SETTINGS OF ATIM_RCR 478
FIGURE 27-16 ATIM CLOCK SOURCE BLOCK DIAGRAM 480
FIGURE 27-17 INTERNAL CLOCK SOURCE MODE WITH CLOCK DIVISION FACTOR OF 1 480
FIGURE 27-18 TI2 EXTERNAL CLOCK CONNECTION EXAMPLE 481
FIGURE 27-19 TIMING IN EXTERNAL CLOCK MODE 1 481
FIGURE 27-20 TIMING IN EXTERNAL CLOCK MODE 1 482
FIGURE 27-21 EXTERNAL TRIGGER INPUT BLOCK DIAGRAM 483
FIGURE 27-22 TIMING 1 IN EXTERNAL CLOCK MODE 2 483
FIGURE 27-23 TIMING 2 IN EXTERNAL CLOCK MODE 2 484
FIGURE 27-24 CAPTURE/COMPARE CHANNEL (CHANNEL 1 INPUT PART) 486
FIGURE 27-25 MAIN CIRCUIT OF CAPTURE/COMPARE CHANNEL 1 487
FIGURE 27-26 OUTPUT SECTION OF CAPTURE/COMPARE CHANNEL (CHANNELS 1 TO 3) 487
FIGURE 27-27 OUTPUT PART OF THE CAPTURE/COMPARE CHANNEL (CHANNEL 4) 487
FIGURE 27-28 PWM INPUT AND CAPTURE MODE TIMING 488
FIGURE 27-29 OUTPUT COMPARE MODE, FLIP OC1 492
FIGURE 27-30 EDGE-ALIGNED PWM WAVEFORM (ARR=7) 493
FIGURE 27-31 CENTER-ALIGNED PWM WAVEFORM (APR=7) 494
FIGURE 27-32 COMPLEMENTARY OUTPUT WITH DEAD ZONE INSERTION 495
FIGURE 27-33 DEAD ZONE WAVEFORM DELAY IS GREATER THAN NEGATIVE PULSE 495
FIGURE 27-34 DEAD ZONE WAVEFORM DELAY IS GREATER THAN POSITIVE PULSE 495
FIGURE 28-1 GPTIM BLOCK DIAGRAM 540
FIGURE 28-2 WAVEFORM OF PRESCALING FROM 1 TO 2 542
FIGURE 28-3 WAVEFORM OF PRESCALING FROM 1 TO 4 542
FIGURE 28-4 UPWARD COUNTING WAVEFORM, INTERNAL CLOCK NOT DIVIDED 544
FIGURE 28-5 UPWARD COUNTING WAVEFORM, INTERNAL CLOCK DIVIDED-BY-2 544
FIGURE 28-6 UPDATE EVENT WHEN ARPE=0 (ARR IS NOT PRELOADED) 545
FIGURE 28-7 UPDATE EVENT WHEN ARPE=1 (ARR PRELOAD) 546
FIGURE 28-8 DOWNWARD COUNTING WAVEFORM, INTERNAL CLOCK NOT DIVIDED 547
FIGURE 28-9 DOWNWARD COUNTING WAVEFORM, INTERNAL CLOCK DIVIDED-BY-2 547
FIGURE 28-10 DOWNWARD COUNTING WAVEFORM, INTERNAL CLOCK DIVIDED-BY-2 548
FIGURE 28-11 DOWNWARD COUNTING， UPDATE EVENT WHEN REPEAT COUNT IS NOT USED 548
FIGURE 28-12 TIMING DIAGRAM OF CENTER ALIGNED COUNTER, ATIM_PCS=0, ATIM_ARR=0X6 549
FIGURE 28-13 COUNTER TIMING DIAGRAM, UPDATE EVENT WHEN ARPE=1 (COUNTER UNDERFLOW) 550
FIGURE 28-14 COUNTER TIMING DIAGRAM, UPDATE EVENT WHEN ARPE=1 (COUNTER OVERFLOW) 551
FIGURE 28-15 ATIM CLOCK SOURCE BLOCK DIAGRAM 551
FIGURE 28-16 INTERNAL CLOCK SOURCE MODE WITH CLOCK DIVISION FACTOR OF 1 551
FIGURE 28-17 TI2 EXTERNAL CLOCK CONNECTION EXAMPLE 552

FIGURE 28-18 TIMING IN EXTERNAL CLOCK MODE 1 552
FIGURE 28-19 TIMING IN EXTERNAL CLOCK MODE 1 554
FIGURE 28-20 EXTERNAL TRIGGER INPUT BLOCK DIAGRAM 554
FIGURE 27-22 TIMING 2 UNDER EXTERNAL CLOCK MODE 556
FIGURE 28-23 CAPTURE/COMPARE CHANNEL (CHANNEL 1 INPUT PART) 559
FIGURE 28-24 MAIN CIRCUIT OF CAPTURE/COMPARE CHANNEL 1 560
FIGURE 28-25 OUTPUT SECTION OF CAPTURE/COMPARE CHANNEL 560
FIGURE 28-26 PWM INPUT CAPTURE MODE TIMING 562
FIGURE 28-27 OUTPUT COMPARE MODE, FLIP OC1 563
FIGURE 28-28 EDGE-ALIGNED PWM WAVEFORM (ARR=7) 564
FIGURE 28-29 CENTER-ALIGNED PWM WAVEFORM (APR=7) 565
FIGURE 28-30 EXAMPLE OF SINGLE PULSE MODE 566
FIGURE 28-31 ETR SIGNAL CLEARS OCXREF OF GPTIM 567
FIGURE 27-32 EXAMPLE OF COUNTER OPERATION IN ENCODER MODE 569
FIGURE 28-33 TIMING IN BIT MODE 570
FIGURE 28-34 TIMING IN GATED MODE 571
FIGURE 28-35 TIMING IN TRIGGER MODE 572
FIGURE 28-36 TIMING IN EXTERNAL CLOCK MODE 2 + TRIGGER MODE 573
FIGURE 29-1 BSTIME32 BLOCK DIAGRAM 599
FIGURE 29-2 WAVEFORM WITH PRESCALED FREQUENCY CHANGING FROM1 TO 2 601
FIGURE 29-3 WAVEFORM WITH PRESCALED FREQUENCY CHANGING FROM 1 TO 4 601
FIGURE 29-4 UP-COUNTING WAVEFORM, NO FREQUENCY DIVISION OF INTERNAL CLOCK 602
FIGURE 29-5 UP-COUNTING WAVEFORM, INTERNAL CLOCK DIVIDED BY 2 603
FIGURE 29-6 UPDATA EVENT WHEN APPE=0 (ARR IS NOT PRELOADED) 603
FIGURE 29-7 UPDATA EVENT WHEN ARPE=1 (ARR PRELOAD) 604
FIGURE 29-8 INTERNAL CLOCK SOURCE MODE, CLOCK DIVISION FACTOR IS 1 604
FIGURE 30-1 LPTIM32 BLOCK DIAGRAM 613
FIGURE 30-2 EXTERNAL ETR PULSE RISING EDGE TRIGGERS COUNTING 614
FIGURE 30-3 EXTERNAL ETR PULSE ASYNCHRONOUS COUNTING (FALLING EDGE) 615
FIGURE 30-4 TIMEOUT MODE 616
FIGURE 30-5 PWM OUTPUT 617
FIGURE 30-6 PWM OUTPUT 618
FIGURE 31-1 RTC BLOCK DIAGRAM 628
FIGURE 31-2 LTBC BLOCK DIAGRAM 630
FIGURE 30-3 RTC TIME READING FLOW CHART 634
FIGURE 32-1 LCD DISPLAY CONTROL MODULE BLOCK DIAGRAM 646
FIGURE 32-2 LCD DRIVE WAVEFORM (1/4 DUTY, 1/3 BIAS, TYPE A) 648
FIGURE 32-3 LCD DRIVE WAVEFORM (1/4 DUTY, 1/3 BIAS, TYPE B) 649
FIGURE 32-4 LCD ON-CHIP RESISTOR BUFFER TYPE DRIVE CIRCUIT 650
FIGURE 33-1 ADC BLOCK DIAGRAM 666
FIGURE 34-1 GPIO BLOCK DIAGRAM 698
FIGURE 34-2 PA11&PA12 BLOCK DIAGRAM 699
FIGURE 34-3 WKUPX FUNCTION BLOCK DIAGRAM 705
FIGURE 34-4 PIN INPUT DIGITAL FILTERING 706

FIGURE 34-5 EXIT SIGNAL INPUT SCHEMATIC 706
FIGURE 37-1 CORTEX-M0 DEBUG SYSTEM DIAGRAM 731

1 Features

1.1 Introduction

The main features of FM33LE0xxA series MCU are as follows：

- 1.8V~5.5V wide supply range
- -40°C ~+125°C temperature range
- Core
  - ARM Cortex-M0
  - Unprivileged/Priviledged support
  - 64Mhz maximum frequency
  - SWD debug interface
  - 24bit Systick timer
- Lower-power platform
  - 120uA/MHz@64MHz,
  - 30uA LPRUN mode@32KHz
  - 5uA Sleep mode
  - 1uA DeepSleep mode (RTC on + all RAM retention + CPU retention)
- Memories
  - 64/128KB Flash memory
  - Flash cycling endurance: 100,000
  - Flash data retention: 10 years@85°C
  - User code protection
  - 16KB Maximum RAM
- Analog peripherals
  - BOR with 4 programmable thresholds
  - Ultra-low-power PDR with 4 programmable thresholds
  - Programmable Supply Voltage Detector
  - 2x low-power analog comparator
  - 12bit 2Msps SAR-ADC

- High precision temperature sensor, +/-5°C over full temperature range

- Communication interfaces
  - UART*5, support LIN2.1 *1 channel
  - LPUART*2
  - 7816 master*1
  - SPI*2, master/slave mode
  - I2C*1, 1Mbps Fm+, master/slave mode
  - I2C_SMB*1，support SMBus
  - 7-channel DMA
  - Programable CRC
- Timers
  - 16bit advanced timer*1
  - 16bit general timer*2
  - 32bit basic timer*1
  - 24-bit Systick*1
  - 32-bit low-power timer*1
  - Windowed watchdog timer*1
  - Independent watchdog timer*1
  - Low-power real-time clock calendar (RTCC), with digital calibration up to +/-0.476ppm
- Security
  - AES 128/192/256bit hardware accelerator, supporting ECB/CBC/CTR/GCM/GMAC
  - True random number generator
- Clocks
  - Programmable high speed RC oscillator, 8/16/24/32MHz, factory-trimmed to +/-0.5%, variation less than +/-2% for 8/16MHz over full temperature range
  - Low power 32768Hz crystal oscillator with fail detector
  - On-chip low speed RC oscillator, 32KHz
  - 4~16MHz high-frequency crystal oscillator
  - PLL, input 1MHz, maximum output 64MHz
- AEC-Q 100 Grade1

1.2 Block Diagram

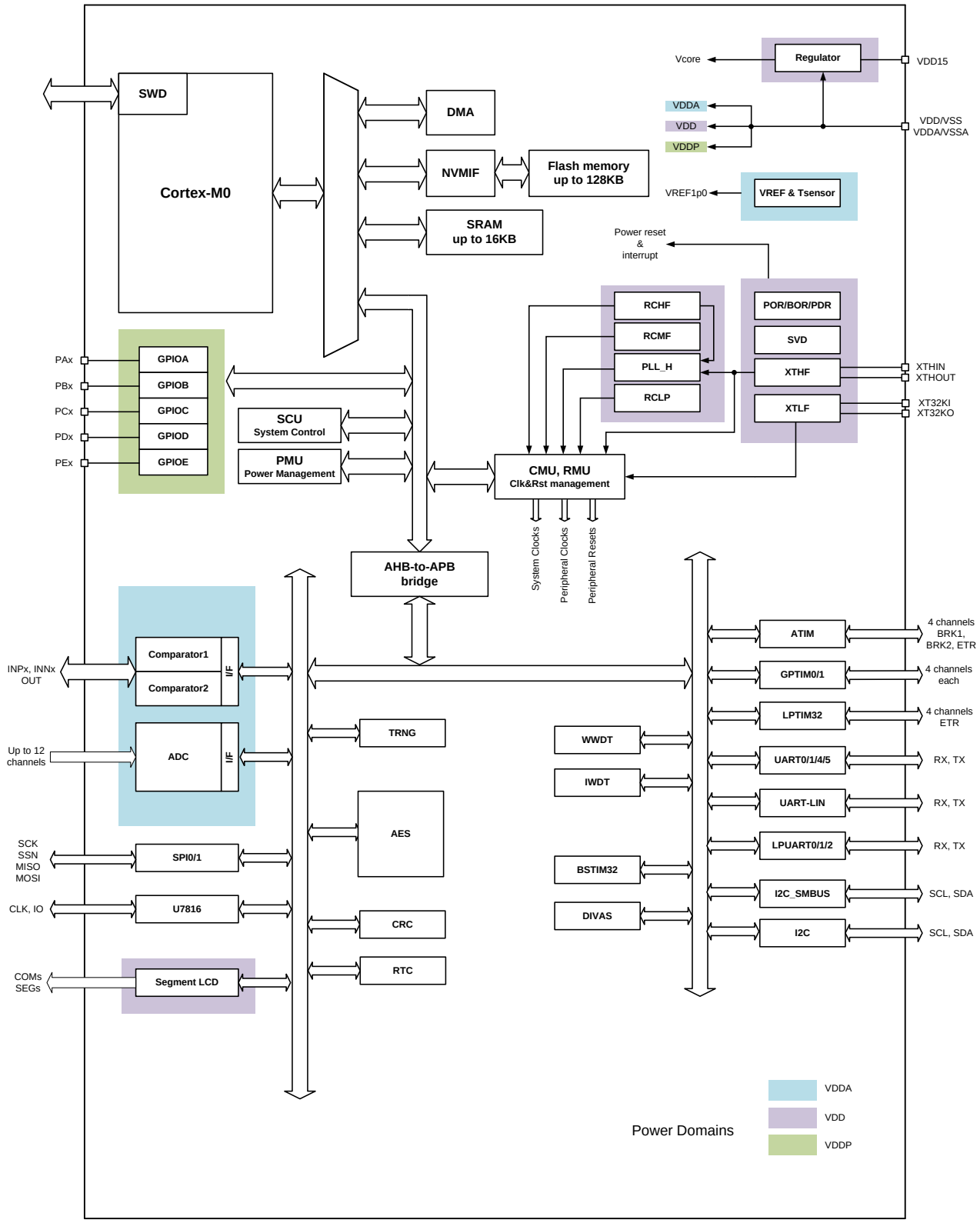


Figure 1-1 FM33LE0xxA block diagram

1.3 Device Lineup

Type	Flash（KBytes）	RAM（KBytes）	Package
FM33LE025A	128	16	LQFP48
FM33LE023A	128	16	QFN32
FM33LE015A	64	16	LQFP48
FM33LE013A	64	16	QFN32

Table 1-1 FM33LE0xxA device lineup

2 Pinout

2.1 Package and pin

2.1.1 FM33LE0x5A(LQFP48)

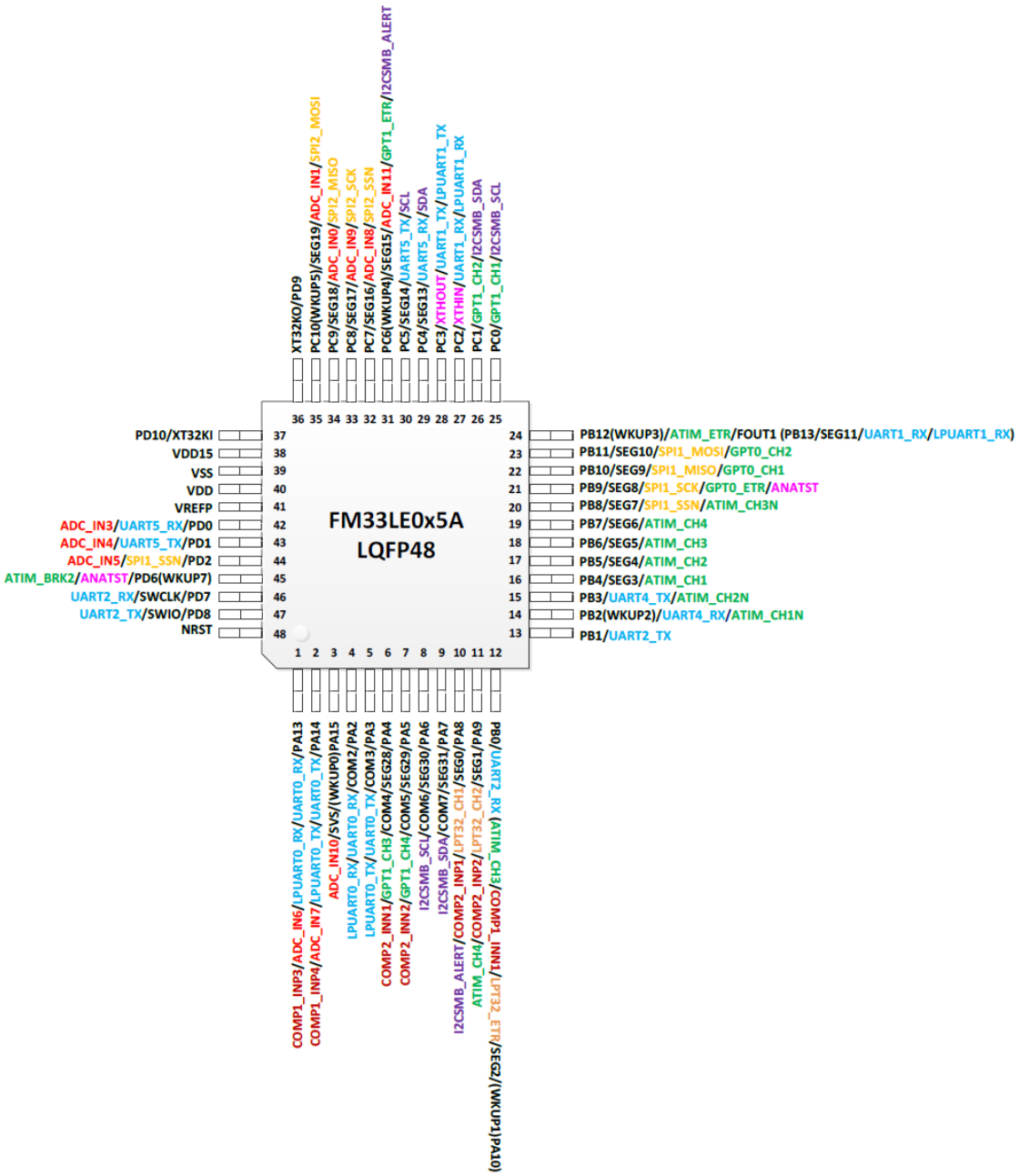


Figure 2-1 FM33LE0x5A LQFP48 package

2.1.2 FM33LE0x3A (QFN32)

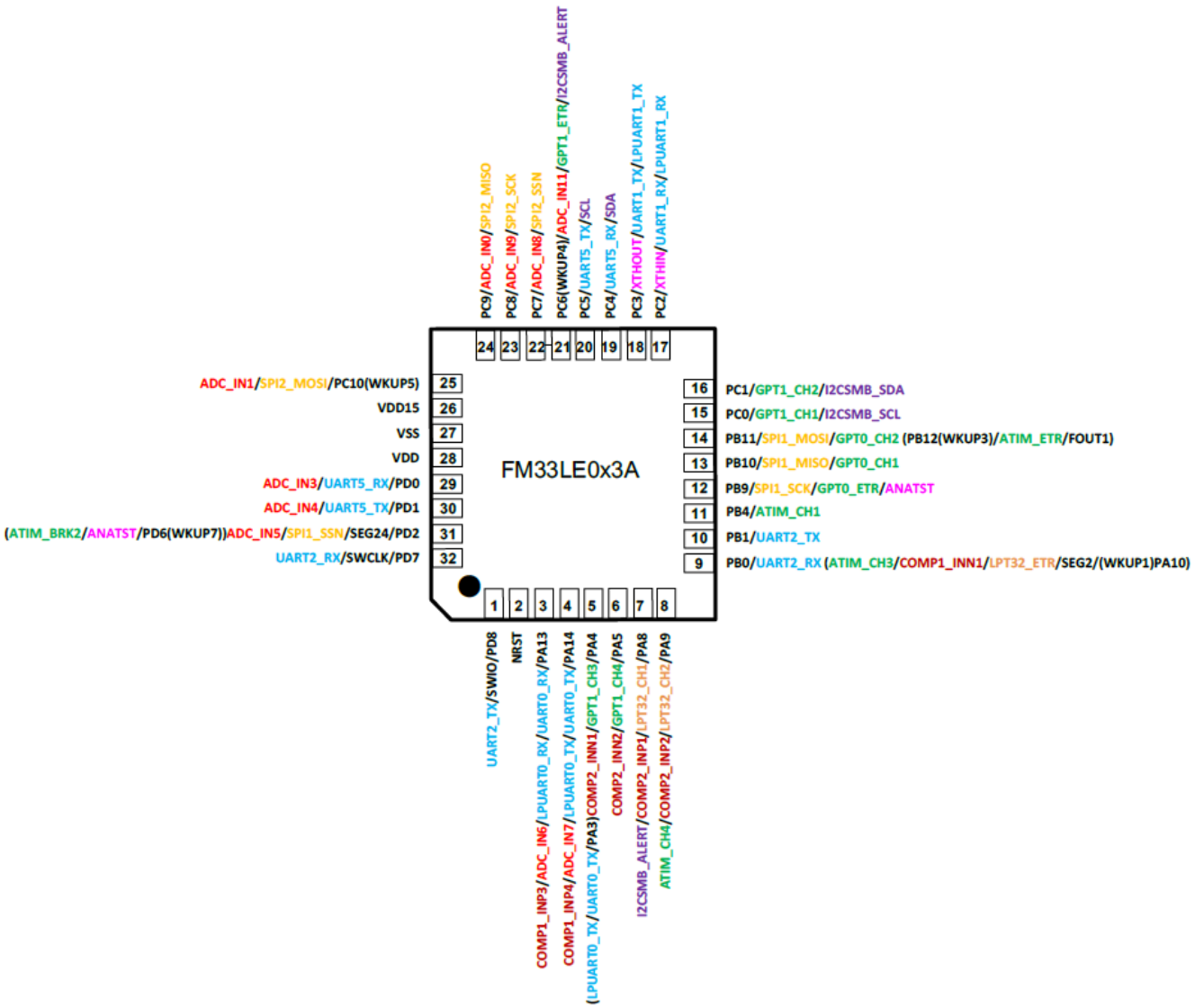


Figure 2-2 FM33LE0x3A QFN32 package

2.1.3 Pin descriptions (FM33LE0xxA)

Pin Number			Pin Function	Descriptions
	LQFP48	QFN32		
	48	2	NRST	Global reset
	1	3	PA13	GPIO
			UART0_RX	UART receive
			LPUART0_RX	Lower-power UART receive
			ADC_IN6	ADC input channel
			COMP1_INP3	Positive input of comparator
	2	4	PA14	GPIO
			UART0_TX	UART transmit
			LPUART0_TX	Low-power UART transmit
			ADC_IN7	ADC input channel
			COMP1_INP4	Positive input of comparator
	3		PA15	GPIO
			WKUP0	External wakeup
			SVS	External power detection
			ADC_IN10	ADC input channel
			PA0	GPIO
			UART4_RX	UART receive
			PA1	GPIO
			UART4_TX	UART transmit
	4		PA2	GPIO
			UART0_RX	UART receive
			LPUART0_RX	Low-power UART receive
	5	5	PA3	GPIO
			UART0_TX	UART transmit
			LPUART0_TX	Low-power UART transmit
	6	5	PA4	GPIO
			GPT1_CH3	General timer external channel
			COMP2_INN1	Comparator input
	7	6	PA5	GPIO
			GPT1_CH4	General timer external channel
			COMP2_INN2	Comparator input
	8		PA6	GPIO

Pin Number			Pin Function	Descriptions
	LQFP48	QFN32		
			I2CSMB_SCL	I2C_SMBUS clock
	9		PA7	GPIO
			I2CSMB_SDA	I2C_SMBUS data
	10	7	PA8	GPIO
			LPT32_CH1	Low-power timer external channel
			COMP2_INP1	Comparator input
			I2CSMB_ALERT	I2C_SMBUS alert signal
	11	8	PA9	GPIO
			LPT32_CH2	Low-power timer external channel
			COMP2_INP2	Comparator input
			ATIM_CH4	Advanced timer external channel
	12	9	PA10	GPIO
			WKUP1	External wakeup
			LPT32_ETR	Low-power timer external trigger input
			COMP1_INN1	Comparator input
			ATIM_CH3	Advanced timer external channel
			PA11	GPIO
			SCL	I2C clock
			PA12	GPIO
			SDA	I2C data
	12	9	PB0	GPIO
			UART2_RX	UART2 (LIN) receive
	13	10	PB1	GPIO
			UART2_TX	UART2 (LIN) transmit
	14		PB2	GPIO
			WKUP2	External wakeup
			UART4_RX	UART receive
			ATIM_CH1N	Advanced timer external channel
	15		PB3	External wakeup
			UART4_TX	UART transmit
			ATIM_CH2N	Advanced timer external channel
	16	11	PB4	GPIO
			SEG3	LCD SEG
			ATIM_CH1	Advanced timer external channel
	17		PB5	GPIO

Pin Number			Pin Function	Descriptions
	LQFP48	QFN32		
			ATIM_CH2	Advanced timer external channel
	18		PB6	GPIO
			ATIM_CH3	Advanced timer external channel
	19		PB7	GPIO
			ATIM_CH4	Advanced timer external channel
	20		PB8	GPIO
			SPI1_SSN	SPI chip select
			ATIM_CH3N	Advanced timer external channel
	21	12	PB9	GPIO
			ANATST	Analog test channel
			SPI1_SCK	SPI clock
			GPT0_ETR	General timer external trigger input
	22	13	PB10	GPIO
			SPI1_MISO	SPI data
			GPT0_CH1	General timer external channel
	23	14	PB11	GPIO
			SPI1_MOSI	SPI data
			GPT0_CH2	General timer external channel
	24	14	PB12	GPIO
			WKUP3	External wakeup
			FOUT1	Clock frequency output
			ATIM_ETR	Advanced timer external trigger input
	23		PB13	GPIO
			UART1_RX	UART receive
			LPUART1_RX	LPUART receive
			PB14	GPIO
			UART1_TX	UART transmit
			LPUART1_TX	LPUART transmit
			VSS	GND
			VDD	Source

Pin Number			Pin Function	Descriptions
	LQFP48	QFN32		
	25	15	PC0	GPIO
			I2CSMB_SCL	I2C_SMBUS clock
			GPT1_CH1	General timer external channel
	26	16	PC1	GPIO
			I2CSMB_SDA	I2C_SMBUS data
			GPT1_CH2	General timer external channel
	27	17	PC2	GPIO
			XTHIN	High-frequency crystal input
			UART1_RX	UART receive
			LPUART1_RX	Low-power UART receive
	28	18	PC3	GPIO
			XTHOUT	High-frequency crystal output
			UART1_TX	UART transmit
			LPUART1_TX	Low-power UART transmit
	29	19	PC4	GPIO
			SCL	I2C clock
			UART5_RX	UART receive
	30	20	PC5	GPIO
			SDA	I2C data
			UART5_TX	UART transmit
	31	21	PC6	GPIO
			WKUP4	External wakeup
			GPT1_ETR	General timer external trigger input
			I2CSMB_ALERT	I2C_SMBUS alert signal
			ADC_IN11	ADC input channel
	32	22	PC7	GPIO
			SPI2_SSN	SPI chip select
			ADC_IN8	ADC input channel
	33	23	PC8	GPIO
			SPI2_SCK	SPI clock
			ADC_IN9	ADC nput channel
	34	24	PC9	GPIO
			SPI2_MISO	SPI data
			ADC_IN0	ADC input channel
	35	25	PC10	GPIO

Pin Number			Pin Function	Descriptions
	LQFP48	QFN32		
			WKUP5	External wakeup
			SPI2_MOSI	SPI data
			ADC_IN1	ADC input channel
			PC11	GPIO
			U7816CLK	7816 interface clock
			GPT0_CH3	General timer external channel
			PC12	GPIO
			U7816IO	7816 interface data
			GPT0_CH4	General timer external channel
	36		PD9	GPIO
			XT32KO	32768Hz Crystal output
	37		PD10	GPIO
			XT32KI	32768Hz Crystal input
	38	26	VDD15	LDO output，external 100nF capacitor to ground
	39	27	VSS	GND
	40	28	VDD	Source
			VREFN	Analog GND
	41		VREFP	Analog Source
			PD11	GPIO
			WKUP6	External wakeup
			FOUT0	Clock frequency output
			ATIM_BRK1	Advanced timer brake input
			ADC_IN2	ADC input channel
	42	29	PD0	GPIO
			UART5_RX	UART receive
			ADC_IN3	ADC input channel
	43	30	PD1	GPIO
			UART5_TX	UART transmit
			ADC_IN4	ADC input channel
	44	31	PD2	GPIO
			SPI1_SSN	SPI chip select
			ADC_IN5	ADC input channel
			PD3	GPIO

Pin Number			Pin Function	Descriptions
	LQFP48	QFN32		
			SPI1_SCK	SPI clock
			PD4	GPIO
			SPI1_MISO	SPI data
			COMP1_INP1	Comparator input
			PD5	GPIO
			SPI1_MOSI	SPI data
			COMP1_INP2	Comparator input
	45		PD6	GPIO
			WKUP7	External wakeup
			ANATST	Analog test channel
			ATIM_BRK2	Advanced timer break input
	46	32	PD7	GPIO
			SWCLK	SWD interface clock
	47	1	PD8	GPIO
			SWIO	SWD interface data

Table 2-1 FM33LE0xxA pin descriptions

2.1.4 Package information

2.1.4.1 LQFP48

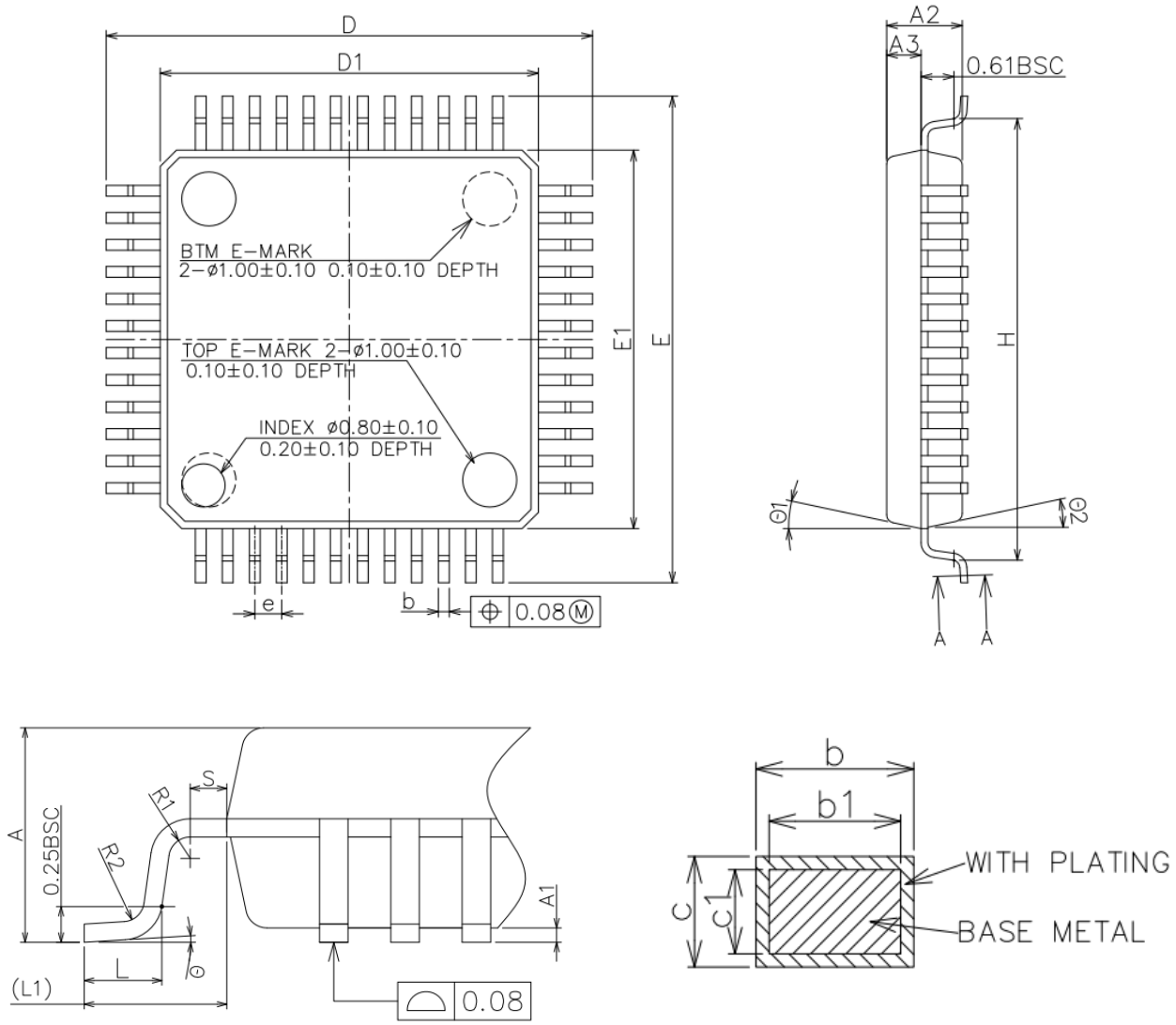


Figure 2-3 LQFP48 package information

Symbol	MIN	NOM	MAX
A	–	–	1.60
A1	0.05	–	0.15
A2	1.35	1.40	1.45
A3	0.59	0.64	0.69
b	0.18	–	0.27

Symbol	MIN	NOM	MAX
b1	0.17	0.20	0.23
c	0.13	–	0.18
c1	0.12	0.127	0.134
D	8.80	9.00	9.20
D1	6.90	7.00	7.10
E	8.80	9.00	9.20
E1	6.90	7.00	7.10
e	0.50BSC		
L	0.45	0.60	0.75
L1	1.00REF		
L2	0.25BSC		
R1	0.08	–	–
R2	0.08	–	0.20
S	0.20	–	–
θ	0°	3.5°	7°
θ1	0°	–	–
θ2	11°	12°	13°
θ3	11°	12°	13°

NOTE:

ALL DIMENSIONS REFER TO JEDEC STANDARD MS-026BDD.

2.1.4.2 QFN32 (non-wettable)

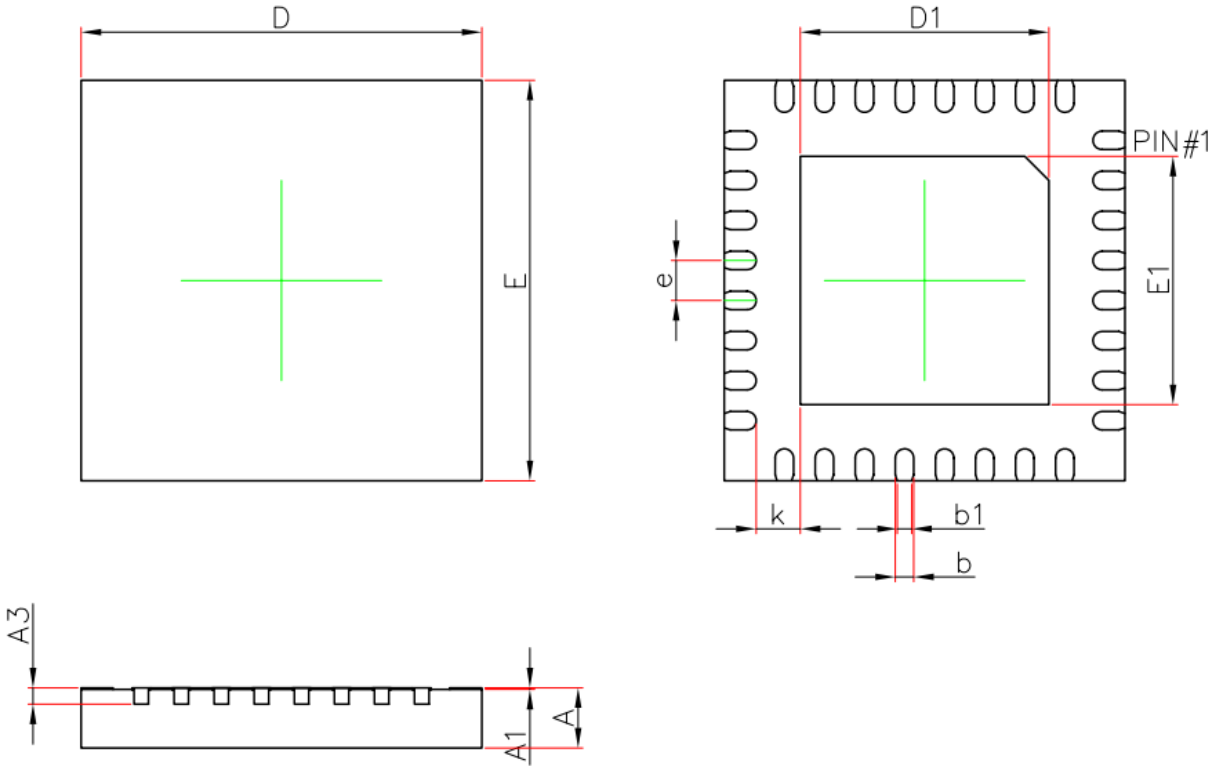


Figure 2-4 QNF32 package information

Symbol	Dimensions In Millimeters		Dimensions In Inches	
	Min.	Max.	Min.	Max.
A	0.700	0.800	0.028	0.031
A1	0.000	0.050	0.000	0.002
A3	0.203 REF.		0.008 REF.	
b	0.180	0.300	0.007	0.012
b1	0.130	0.230	0.005	0.009
D	4.900	5.100	0.193	0.201
D1	3.000	3.200	0.118	0.126
E	4.900	5.100	0.193	0.201
E1	3.000	3.200	0.118	0.126
e	0.500 BSC.		0.020 BSC.	
k	0.550 REF.		0.022 REF.	
L	0.324	0.476	0.013	0.019

2.1.4.3 QFN32 (wettable-flank)

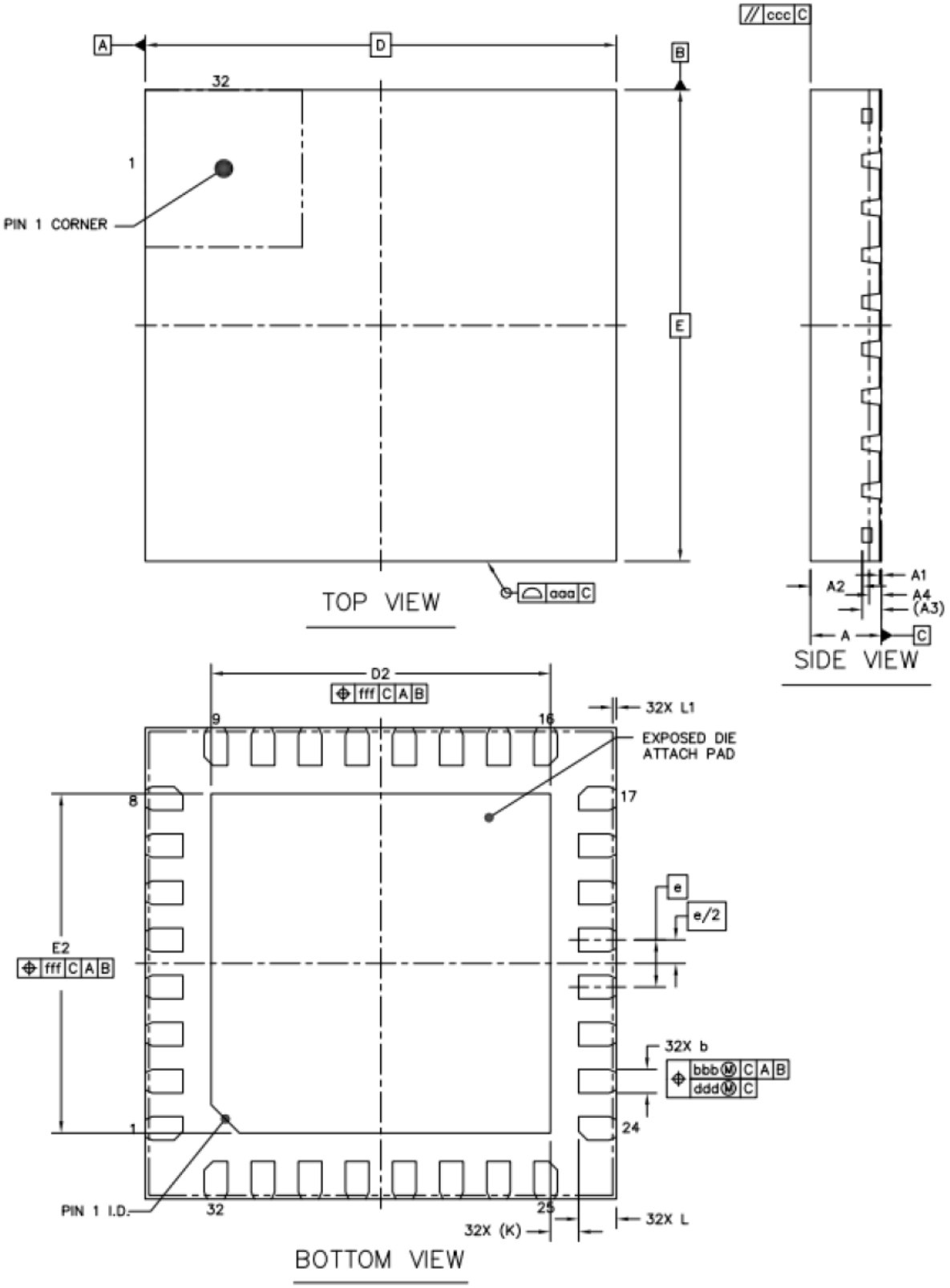


Figure 2-5 QFN32(wettable flank) package

		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.7	0.75	0.8
STAND OFF		A1	0	0.02	0.05
MOLD THICKNESS		A2	---	0.55	---
L/F THICKNESS		A3		0.203 REF	
SIDE WETTABLE DEPTH		A4	0.075	---	0.18
LEAD WIDTH		b	0.2	0.25	0.3
BODY SIZE	X	D		5 BSC	
	Y	E		5 BSC	
LEAD PITCH		e		0.5 BSC	
EP SIZE	X	D2	3.5	3.6	3.7
	Y	E2	3.5	3.6	3.7
LEAD LENGTH		L	0.3	0.4	0.5
SIDE WETTABLE WIDTH		L1	0.01	---	0.09
LEAD TIP TO EXPOSED PAD EDGE		K		0.3 REF	
PACKAGE EDGE TOLERANCE		aaa		0.1	
MOLD FLATNESS		ccc		0.1	
LEAD OFFSET		bbb		0.1	
		ddd		0.05	
EXPOSED PAD OFFSET		fff		0.1	

2.2 Welding installation instructions

Fudan Microelectronics chips are packaged using a lead-free process. The reflow process parameters are recommended to be set following the JEDEC standard.

According to JEDEC standard J-STD-020, the following table shows the recommended peak temperature setting for reflow soldering in Pb-free process. Users can select the appropriate peak reflow temperature in the following table according to the specifications of different thickness and volume of the chip.

Package Thickness	Plastic volume mm^3 <350	Plastic volumemm^3 350 - 2000	Plastic volumemm^3 >2000
<1.6mm	260℃	260℃	260℃
1.6~2.5 mm	260℃	250℃	245℃
>2.5mm	250℃	245℃	245℃

The following table gives the peak reflow temperatures for various package forms:

Package Type	Plastic volume mm	Plastic volume mm^3	Peak Reflow Temperature
LQFP80	1.4	201.6	260℃

For soldering profile setting, please refer to JEDEC standard J-STD-020, Lead-free Process Reflow Temperature Profile Setting for instructions on setting.

Profile Feature	Pb-Free Assembly
Preheat/Soak Temperature Min (T_{smin}) Temperature Max (T_{smax}) Time (t_s) from (T_{smin} to T_{smax})	 150 °C 200 °C 60-120 seconds
Ramp-up rate (T_L to T_p)	3 °C/second max.
Liquidous temperature (T_L) Time (t_L) maintained above T_L	217 °C 60-150 seconds
Peak package body temperature (T_p)	For users T_p must not exceed the Classification temp in Table 4-2. For suppliers T_p must equal or exceed the Classification temp in Table 4-2.
Time (t_p)* within 5 °C of the specified classification temperature (T_c), see Figure 5-1.	30* seconds
Ramp-down rate (T_p to T_L)	6 °C/second max.
Time 25 °C to peak temperature	8 minutes max.

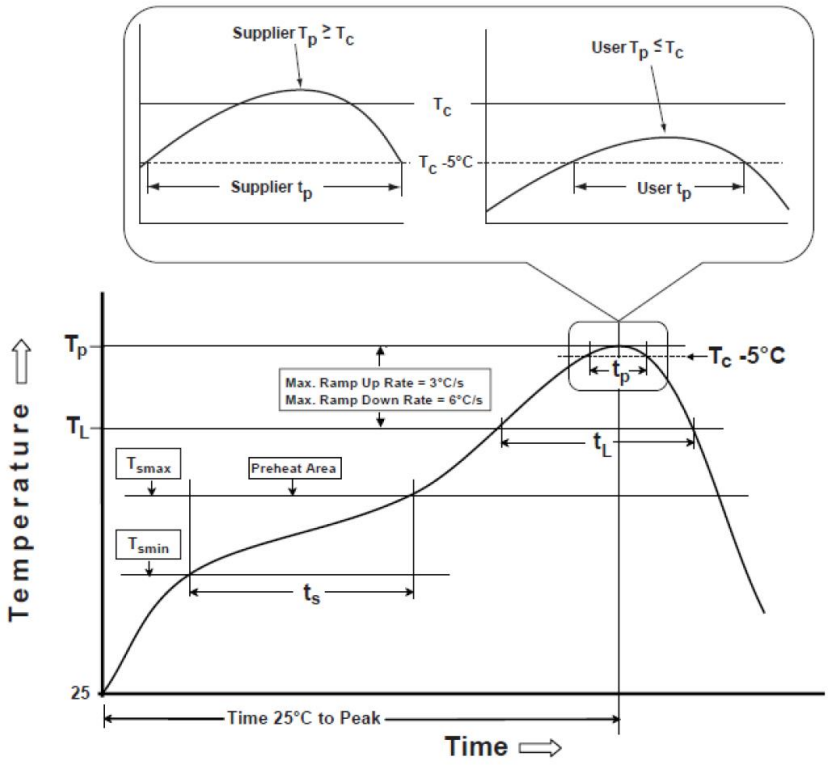


Figure 2-1 Heat-resistant reflow temperature profile for 6JEDEC standards

Special Statement:

- Before soldering the chip, please observe whether the humidity card changes color to confirm the integrity of the moisture-sensitive package.
- If not specified, do not reflow more than 3 times.

2.3 MSL rating

FM33LE0xxA chips have a moisture sensitivity level of MSL3 according to the JEDEC standard: J-STD-020. Soldering should be performed within one week of opening the package and leaving it in a non-dry environment.

2.4 Thermal resistance characteristics

The chip junction temperature can be calculated using the following formula:

$$T_J = T_A + P_D \times \theta_{JA}$$

- T_A: Working environment temperature,℃
- θ_{JA}: Package Thermal Resistance Ratio
- P_D: Chip power consumption, Includes core power and IO power consumption, uint W;

$$P_{IO} = \sum(V_{OL} \times I_{OL}) + \sum((V_{DDIO} - V_{OH}) \times I_{OH})$$

The thermal resistance coefficients of different package forms can be referred to the following table:

Package form	Package Size	Thermal Resistance θ_{JA} (℃/W)
LQFP80	12*12*1.4mm	50
LQFP100	14*14*1.4mm	45
LQFP64	10*10*1.4mm	48
LQFP48	7*7*1.4mm	55
QFN32	5*5*1.4mm	35

Example：

- Assuming that the temperature of the average working environment is TA = 55°C.
- Chip core current IDD=5mA, VDD=3.6V
- 10 IO ports simultaneously output low levels, each IO sink 5mA, VOL=0.3V
- 10 IO ports simultaneously output high levels, each IO sink 5mA, VOL=2.9V

$$P_D = P_{INT} + P_{IO} = 5mA \times 3.6V + 10 \times 5mA \times 0.3V + 10 \times (3.6V - 2.9V) \times 5mA = 68mW$$

For LQFP64 package product, θ_{JA}=48℃/W，chip junction temperature can be calculated.

$$J_T = T_A + P_D \times \theta_{JA} = 55\ ℃ + 0.068W \times 48\ ℃/W\ = 58.264\ ℃$$

3 Electrical Characteristics

3.1 Parameter description

Unless otherwise stated, the electrical parameters listed in this section are tested at the indicated ambient temperatures and supply voltages during chip mass production testing.

Parameters obtained based on feature parameter extraction and design simulation are declared in the table remarks, and these parameters are not covered in the mass production test. Unless otherwise stated, the TYPICAL parameters are obtained by testing a sufficient number of samples from standard mass production process wafer lots at T_A=25℃ and VDD=5V; the TYPICAL parameters are used as a user design reference.

3.2 Test standard

The mass production test of FM33LE0xxA series MCUs is performed with reference to AEC-Q001 and AEC-Q002 standards.

3.3 Parameter test conditions

3.3.1 Power supply scheme

The power supply scheme shown below is used for chip mass production FT testing, VDD= 5V.

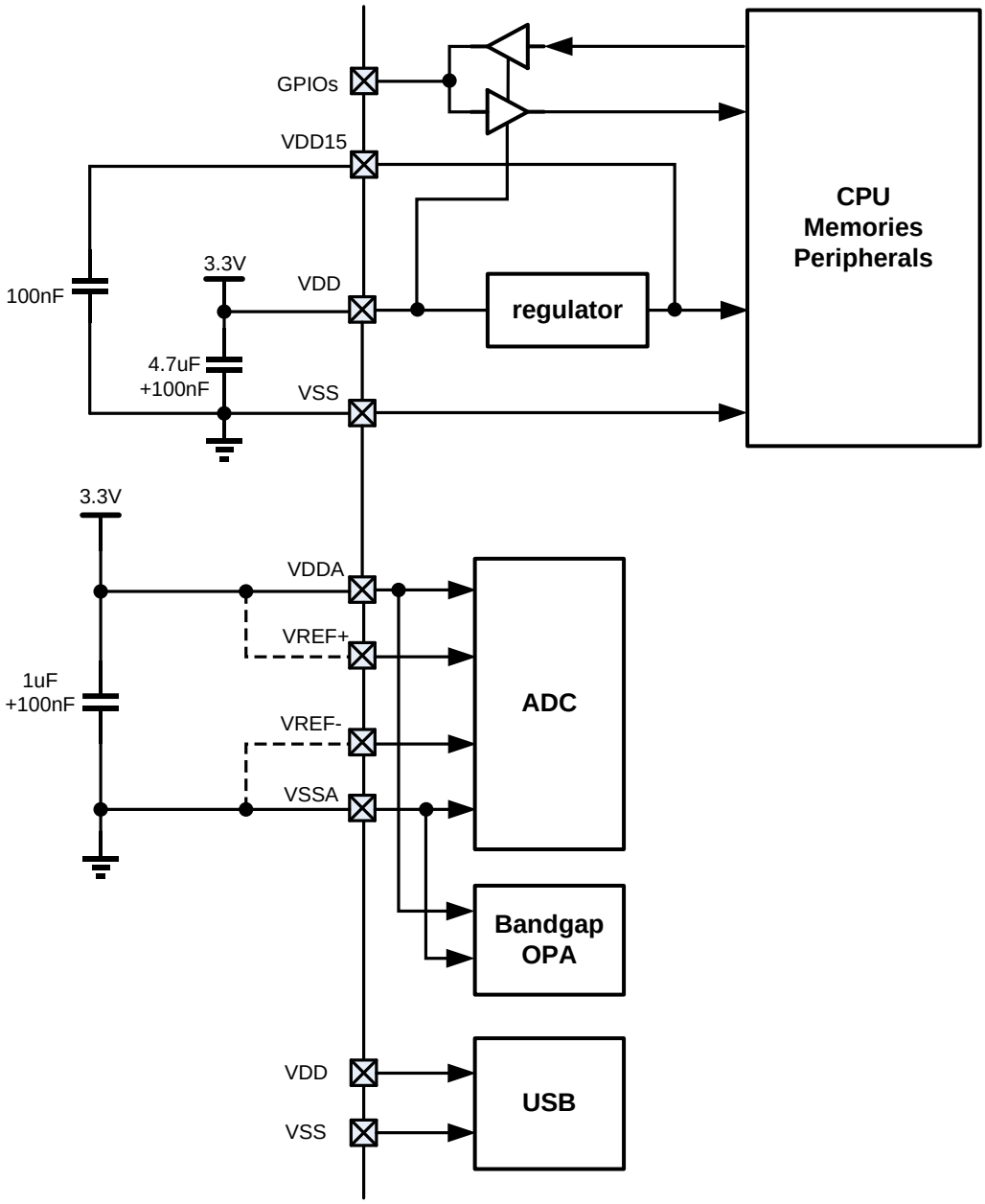


Figure 3-1 FM33LE0xxA power supply scheme

3.4 Absolute maximum ratings

When the voltage and current applied to the chip exceed the absolute maximum rating defined in the limit parameter table, the chip may be permanently damaged. Exceeding the absolute maximum ratings for a short time may affect the reliability and operating life of the device.

Symbol	Parameter	min	max	unit
V_{DD}-V_{SS}	Supply voltage (VDD, VDDA)	-0.3	6.5	V
V_{PIN}	Pin voltage	V_{ss}-0.3	6.5	V
$\|\triangle V_{DD}\|$	Voltage difference(1) between VDDand VDDA	-	50	mV
$\|\triangle V_{SS}\|$	Voltage difference between all ground pins	-	50	mV
T_A	Working temperature	-40	125	℃
T_{STG}	Storage temperature	-55	150	℃
HBM	ESD HBMmode TA=25℃ The test standard conforms to JEDEC JS-001	-	+/-4000	V
CDM	ESD CDM mode TA=25℃ The test standard conforms toJEDEC JS-002	-	+/-1000	V
LU	IO Latchup The test standard conforms toJESD78E	Class-II Level-A		
ΣI_{VDD}	Total current flows into VDD (Source)	-	90	mA
ΣI_{VSS}	Total current flows out of VSS (Sink)	-	70	mA
ΣI_{IO}	Total current of all IO sinks	-	70	mA
	Total current of all IO sources	-	90	mA

Table 3-1 FM33LE0XXA absolute maximum ratings

Note：

1. It is recommended to use the same voltage source to power VDD and VDDA.

3.5 Performance Parameters

3.5.1 General operating conditions

Symbol	Parameter	Conditions	min	max	unit
f_{HCLK}	AHB clock frequency	-	0	64	MHz
f_{PCLK}	APB clock frequency	-	0	64	
VDD	General operating voltage	BOR on	1.8	5.5	V
		BOR off	1.65	5.5	
VDDA	Analog circuit operating voltage	Must satisfy VREFP=VDD VREFN=VSS	1.8	5.5	V
T_J	Junction temperature		-40	130	°C
C_{VDD15}	VDD15 (internal LDO output) Pin Regulator Capacitor	Close to the pin layout	0.1	4.7	uF

3.5.2 Current consumption characteristics

The current consumptions are factory-tested at ambient temperature, and the high and low temperature current parameters are based on characterization.

When power consumption parameters are measured, the MCU is configured as follows:

- All functional pins are configured in GPIO mode, and the input and output enable is turned off to avoid floating leakage
- All peripherals are turned off and the operating clocks are stopped except as otherwise stated
- The maximum current consumption data at room temperature represents the test upper limit standard
- Typical current consumption data at room temperature represent the mean values of a large number of sample distributions
- Unless otherwise specified, all current consumption is tested under VDD=VDDA=5V

Active mode power

Symbol	Parameter	Test conditions		Value			Unit
				Min	Type	Max	
IDD_{RUN}	Current consumption in active mode CPU fetches from Flash Dhrystone	f_{AHB}=16MHz (RCHF) PLL off Flash 0 wait	TA=25℃	-	2.1	2.5	mA
			TA=125℃	-	2.1	2.5	
		f_{AHB}=24MHz (RCHF) PLL off Flash 0 wait	TA=25℃	-	3.0	3.5	mA
			TA=125℃	-	2.9	3.4	
		f_{AHB}=64MHz PLL on Flash 1 wait	TA=25℃	-	8	15	mA
			TA=125℃	-	6	30	

Table 3-2 ACTIVE current characteristics

LP RUN mode power

Symbol	Parameter	Test conditions		Value			Unit
				Min	Type	Max	
IDD_{LPRUN}	Current consumption in LP RUN，CPU fetches from FlashCoremark	f_{AHB}=32768Hz (XTLF) PLL, RCHF, RCMF off Flash 0 wait	TA=25℃		26	32	uA
			TA=125℃		48	100	

Table 3-3 LP RUN current characteristics

SLEEP mode power

Symbol	Parameter	Test conditions		Value			Unit
				Min	Typ	Max	
I_{sleep1}	Sleep mode current	BOR、SVD disabled RTC uses XTLF to run CPU, RAM, peripheral retained RCLP enable	TA=25℃	-	3.5	10	uA
			TA=125℃[1]	-	27	90	

Table 3-4 SLEEP current characteristics

DEEPSLEEP mode

Symbol	Parameter	Test conditions		Value			Unit
				Min	Typ	Max	
$I_{dpsleep}$	DeepSleep mode current	BOR、SVD disabled RTC uses XTLF to run CPU, RAM, peripheral retained RCLP enable	TA=25℃	-	1	2.5	uA
			TA=125℃	-	25	80	

Table 3-5 DEEPSLEEP current characteristics

3.5.3 Reset and supply detection

Symbol	Parameter	Test conditions		Value			Unit
				Min	Typ	Max	
t_{VDD}	VDD rising slope			2		∞	us/V
	VDD falling slope	PDR		200		∞	us/V
		BOR		600		∞	us/V
T_{reset_delay}	Power on reset time delay				0.5		ms
T_{pdr_filter}	Power down reset filtering time				4		us
V_{POR}	Power on reset voltage	T_A=-40~125℃		1.6	1.8	2.1	V
V_{BOR}	Power down reset Voltage T_A=-40~125℃	BORCFG=2'b01		1.4	1.65	1.9	V
V_{PDR}	Power down reset voltage in low-power mode[1] T_A=-40~125℃	PDRCFG==2'b11		1.0	1.4	1.9	V
I_{BOR}	BOR current consumption				1.2		uA
I_{PDR}	PDR current consumption				55		nA
V_{SVD}	Voltage detection threshold level (Comparison	SVD[3:0]=0000	Fall		1.800		V
			Rise		1.900		
		SVD[3:0]=0001	Fall		2.014		V
			Rise		2.114		
		SVD[3:0]=0010	Fall		2.229		V
			Rise		2.329		

Symbol	Parameter	Test conditions		Value			Unit
				Min	Typ	Max	
	benchmark 1.2V)	SVD[3:0]=0011	Fall		2.443		V
			Rise		2.543		
		SVD[3:0]=0100	Fall		2.657		V
			Rise		2.757		
		SVD[3:0]=0101	Fall		2.871		V
			Rise		2.971		
		SVD[3:0]=0110	Fall		3.086		V
			Rise		3.186		
		SVD[3:0]=0111	Fall		3.300		V
			Rise		3.400		
		SVD[3:0]=1000	Fall		3.514		V
			Rise		3.614		
		SVD[3:0]=1001	Fall		3.729		V
			Rise		3.829		
		SVD[3:0]=1010	Fall		3.943		V
			Rise		4.043		
		SVD[3:0]=1011	Fall		4.157		V
			Rise		4.257		
		SVD[3:0]=1100	Fall		4.371		V
			Rise		4.471		
		SVD[3:0]=1101	Fall		4.586		V
			Rise		4.686		
		SVD[3:0]=1110	Fall		4.800		V
			Rise		4.900		
		SVD[3:0]=1111	Fall		-		V
			Rise		-		

Table 3-6 Reset and supply detection

[1] Based on characterization

3.5.4 High precision reference

A high precision voltage reference is built in the chip to provide high precision and high stability reference voltage for ADC and COMP.

When the chip leaves the factory, Fudan Microelectronics will use the on-chip ADC to sample the reference output under the specific power voltage and temperature, and save the conversion result in the flash of the chip. The conversion value can be used as the reference in the user application.

Symbol	Parameter	Bus Addresses
AVREF_RAW	Voltage value of AVREF output Test conditions： TA = ambient temperature VDDA=3V	TBD

The main parameters of high precision reference are as follows:

Symbol	Parameter	Test conditions		Value			Unit
				Min	Typ	Max	
V_{REF}	Reference output voltage	-40℃≤T_A≤125℃		0.96	1.0	1.03	V
T_{setup}	Internal reference start-up time	-		-	5	10	us
T_{coeff}	Internal reference temperature coefficient	-40℃≤T_A≤125℃		-1.5	-	1.5	%
T_{S_VREF}	Sampling time of VREF measured by ADC			10			us
T_{ADC_BUF}	Output fully build delay after VREF buffer enable					100	us
I_{REF}	Reference working current	T_A = 25℃ VDDA=3V	PTAT_EN=0		2		uA
			PTAT_EN=1		3		

Table 3-7 High precision reference characteristic

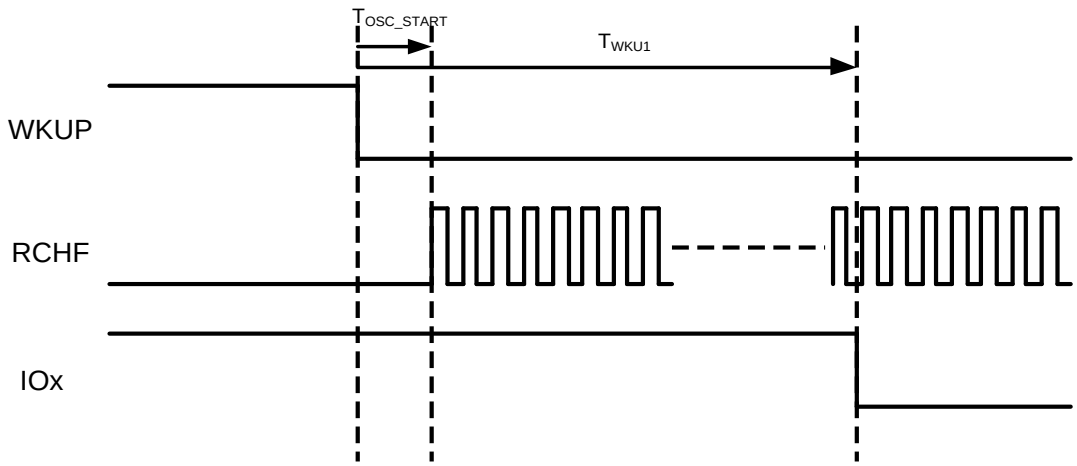
3.5.5 Wake up time in low-power mode

Symbol	Parameter	Test conditions	Value			Unit
			Min	Typ	Max	
T_{WKU1}	Sleep/DeepSleep wake up time[1]	Wake up the chip by WKUP pin, PRIMASK=1 interrupts disabled; After the CPU wakes up, the program toggles some IO output, measuring the time between the edges of the WKUP signal and the IO	-	6.5	-	us
T_{WKU2}	LPRUN wake up time		-	0	-	us

Table 3-8 Wake up time characteristics

[1] Based on characterization

Typical wake up event waveform, for design reference only



In the figure above, T_{OSC_START} represents the RCHF start-up time after the wake event, and the typical value is less than 3us

T_{WKU1} is the time between the arrival of the wake-up event and the toggle of IO after the program runs, with a typical value of 5.5us.

If interrupts are not masked through PRIMASK, the wake event will cause the CPU enters the interrupt service routine. The process of the CPU entering the interrupt service routine introduces an additional delay time.

Note: RCHF 8Mhz is used as the system clock after wake-up for time assessment. If 16Mhz or 24Mhz frequency is selected after wake-up, the wake-up time will be shortened accordingly.

3.5.6 External clock source characteristics

Symbol	Parameter	Test conditions	Value			Unit
			Min	Typ	Max	
f_{XTLF}	XTLF oscillation frequency	External 32768Hz crystal		32768		Hz
T_{start}	XTLF start-up time	External 32768Hz crystal C_{load}=12pF XTLFIPW==3'b000		1	3	s

Table 3-9 Low-frequency crystal characteristics

Symbol	Parameter	Test conditions		Value			Unit
				Min	Typ	Max	
F_{XTHF}	XTHF frequency	VDD=3.3V		4	-	24	MHz
R_{fb}	Feedback Resistor	-		-	200	-	KΩ
gm	Transconductance gain	VDD=5V	HF_CFG =00000	-	1.75	-	mA/V
			HF_CFG =00001	-	3.5	-	
		VDD=3V	HF_CFG =00000	-	1.2	-	
			HF_CFG =00001	-	2.4	-	
			HF_CFG =00010	-	3.48	-	
		VDD=1.6V	HF_CFG =00000	-	0.32	-	
			HF_CFG =00100	-	1.59	-	
			HF_CFG =00111	-	2.55	-	
VDD_{rise}	XTHF Minimum Operating Voltage HF_CFG=00000	4MHz		1.08	-	-	V
		8MHz,		1.6	-	-	
		24MHz		1.8	-	-	
	XTHF Minimum Operating Voltage HF_CFG=11111	4MHz		0.9	-	-	
		8MHz,		1.0	-	-	
		24MHz		1.1	-	-	
IDD	XTHF Operating Current HF_CFG=00000	4MHz		-	150	-	uA
		8MHz		-	180	-	
		16MHz			250		
		24MHz			360		
	XTHF Operating Current HF_CFG=1000	4MHz		-	1000	-	
		8MHz		-	1050	-	
		16MHz			1200		
		24MHz			1450		
T_{start1}	XTHF 8M start-up time	VDD=3.3V HF_CFG=00000		-	2.3	-	ms
		VDD=3.3V, HF_CFG=10000		-	0.5	-	ms
T_{start2}	XTHF 16M start-up time	VDD=3.3V, HF_CFG=00000		-	2.1	-	ms
		VDD=3.3V,		-	0.2	-	ms

Symbol	Parameter	Test conditions	Value			Unit
			Min	Typ	Max	
		HF_CFG=10000				
C_L	Load capacitance	-	5	-	25	pF
C_{IN}	On-chip integrated load capacitance	-	0	-	4	pF

Table 3-10 High-frequency crystal characteristics

Notes.

[1] The above indicators are based on feature parameter extraction

[2] 8~16MHz crystal or ceramic oscillator is recommended with the recommended oscillation strength configuration to reduce the power consumption and noise radiation of the clock oscillator

XTHF typical gm parameters (for design reference only):

HF_CFG	Gm(mA/V)		
	VDD=5V	VDD=3V	VDD=1.6V
00000	1.75	1.2	0.32
00001	3.5	2.4	0.64
00010	4.91	3.48	0.95
00011	6.67	4.68	1.28
00100	8.07	5.77	1.59
00101	9.83	6.96	1.91
00110	11.2	8.05	2.23
00111	13	9.25	2.55
01000	14.4	10.3	2.86
01001	16.2	11.5	3.18
01010	17.6	12.6	3.5

The equivalent circuit model of crystal or ceramic oscillator is shown in the following figure:

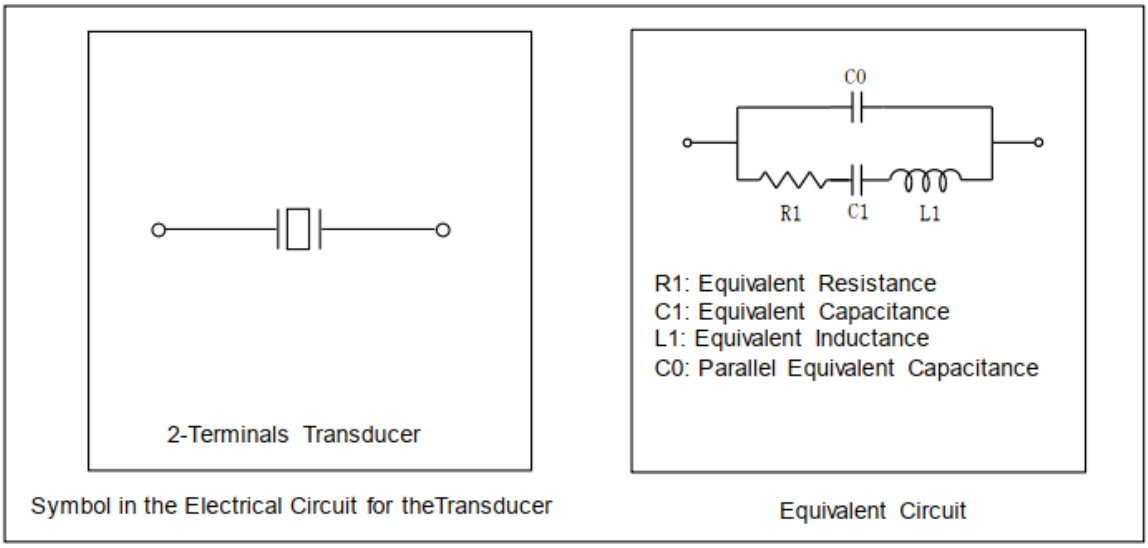


Figure 3-2 The equivalent circuit model of crystal or ceramic oscillator

The minimum loop gain for oscillator starting can be calculated by the following equation.

$$g_{mcrit} = 4 \times ESR \times (2\pi F)^2 \times (C_0 + C_L)^2$$

Where C_L is the load capacitance required by the crystal or the built-in load capacitance of the ceramic oscillator, ESR can be calculated by the following equation.

$$ESR = R_1 \times \left(1 + \frac{C_0}{C_L}\right)^2$$

In order to ensure sufficient safe oscillation margin, it is usually required that g_m is greater than 5 times g_{mcrit}, according to which the appropriate XTHF oscillation strength configuration can be selected according to the crystal or ceramic oscillator manual.

3.5.7 Internal clock characteristics

Internal high-frequency RC oscillator

Symbol	Parameter	Test conditions		Value			Unit
				Min	Typ	Max	
f_{RCHF}	RCHF frequency	VDD=1.8~5.5V	FSEL==2'b00	7.92	8	8.08	MHz
			FSEL==2'b01	15.84	16	16.16	
			FSEL==2'b10	23.76	24	24.24	
			FSEL==2'b11	-	-	-	
ACC_{RCHF}	Frequency accuracy over full temperature range	VDD=1.8~5.5V	FSEL==2'b00 T=-40~+85℃	-2	-	2	%
			FSEL==2'b01 T=-40~+85℃	-1	-	2	%
			FSEL==2'b10	-1.5	-	2.5	%

Symbol	Parameter	Test conditions	Value			Unit
			Min	Typ	Max	
		T=-40~+85℃				
		FSEL==2'b11 T=-40~+85℃				
		FSEL==2'b00 T=-40~+125℃	-2		2.5	%
		FSEL==2'b01 T=-40~+125℃	-4		4	
		FSEL==2'b10 T=-40~+125℃	-5		5	
		FSEL==2'b11 T=-40~+125℃				%

Table 3-11 Internal high-frequency RC oscillator characteristics

Internal middle-frequency RC oscillator

Symbol	Parameter	Test conditions	Value			Unit
			Min	Typ	Max	
f_{RCMF}	RCMF oscillation frequency	VDD=1.8~5.5V T_A=25℃		4		MHz
I_{DD_RCMF}	RCMF current consumption	VDD=1.8~5.5V T_A=25℃		20		uA
ACC_{RCMF}	Frequency accuracy over full temperature range	VDD=1.8~5.5V T_A=-40~+125℃	-6	-	5	%

Table 3-12 Internal middle-frequency RC oscillator characteristics

Internal low-frequency RC oscillator

Symbol	Parameter	Test conditions	Value			Unit
			Min	Typ	Max	
F_{RCLP}	RCLP oscillation frequency	VDD=1.8~5.5V T_A=25℃	28	32	36	KHz
I_{DD_RCLP}	RCLP consumption	VDD=1.8~5.5V T_A=25℃		400		nA
ACC_{RCLP}	Frequency accuracy	VDD=1.8~5.5V T_A=-40~+125℃	-10	-	10	%

Table 3-13 Internal low-frequency RC oscillator characteristics

3.5.8 PLL Characteristics

Symbol	Parameter	Test conditions	Value			Unit
			Min	Typ	Max	
F_{PLL}	PLL output frequency	VDD=1.8~5.5V T_A=25℃	2		64	MHz
I_{DD_PLL}	PLL current consumption	Input frequency 1Mhz, output frequency 32Mhz		330		uA
		Input frequency1Mhz, output frequency 64Mhz		450		
t_{LOCK}	PLL lock time			65		us

Table 3-14 PLL characteristics

3.5.9 ADC Characteristics

3.5.9.1 Parameter description

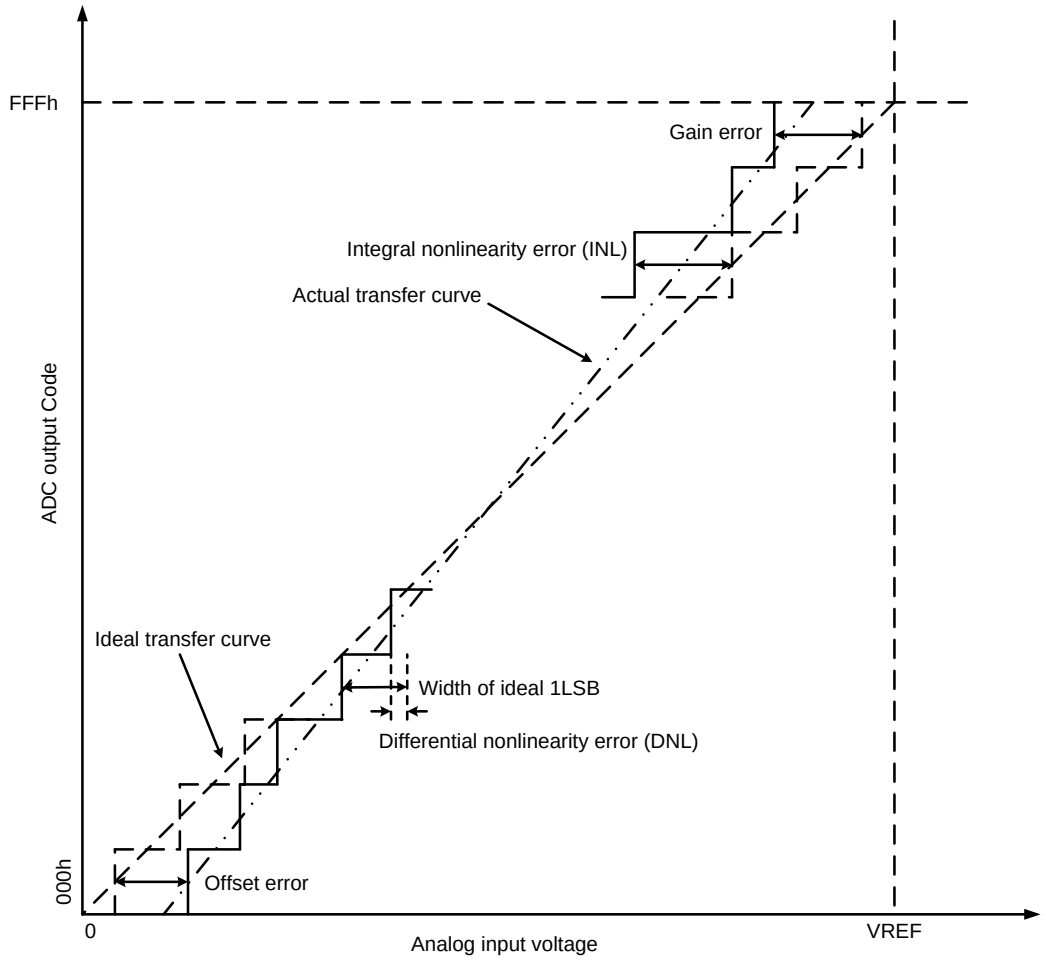


Figure 3-3 parameter description

Differential Nonlinearity (DNL)

DNL indicates the difference between the 1LSB width of the ideal ADC conversion curve and the 1LSB width of the actual ADC conversion curve.

Integral Nonlinearity (INL)

INL indicates the maximum deviation between the actual ADC conversion curve and the ideal ADC conversion curve.

Offset error (Offset error)

Offset error indicates the difference between the position of the first code word jump of the actual ADC and the position of the first code word change of the ideal ADC.

Gain error (Gain error)

Gain error indicates the difference between the position of the last code word change of the actual ADC and the position of the last code word change of the ideal ADC at full amplitude input.

3.5.9.2 Performance characteristics

Symbol	Parameter	Test condition	Value			Unit
			Min	Typ	Max	
VDDA	Operating Voltage Range		1.8		5.5	V
VREF+	Positive Reference Voltage		1.5		VDDA	V
VREF-	Negative reference voltage		0		0.5	V
T_J	Operating junction temperature range		-40		125	°C
V_{AIN}	Input Voltage Range	Single-ended mode	VREF-		VREF+	V
		Differential mode				V
C_s	Sample Hold Capacitance			2.6		pF
R_{AIN}	External Input Impedance				2	KΩ
R_{ADC}	Sampling Switch Impedance			300		Ω
F_{CLK}	ADC operating clock frequency				32	MHz
F_S	ADC sampling frequency	VDDA=2.0~5.5V			2	Msps
		VDDA=1.8~2.0V			1.5	
T_{SAMP}	Sample Hold Time		3.5		10.5	F_{CLK}
T_{CONV}	Conversion time			12.5		F_{CLK}
T_{CAL}	Self-calibration time			4096		F_{CLK}
ADC Dynamic Performace						
ENOB	Effective bits versus input	Single-ended		10.6		bits

Symbol	Parameter	Test condition	Value			Unit
			Min	Typ	Max	
	signal frequency VDDA=3.3V VREF+=VDDA F_S=1Msps -40℃≤T_A≤85℃	mode F_{AIN}=29KHz				
		Single-ended mode F_{AIN}=199KHz		10.5		bits
SNDR	Signal-to-noise distortion ratio VDDA=3.3V VREF+=VDDA F_S=1Msps -40℃≤T_A≤85℃	Single-ended mode F_{AIN}=29KHz		66		dB
		Single-ended mode F_{AIN}=499KHz		65		dB
SFDR	Spurious-free dynamic range VDDA=3.3V VREF+=VDDA F_S=1Msps F_{AIN}=29KHz -40℃≤T_A≤85℃	Single-ended mode		80		dB
ADC Static Performance						
DNL	Differential Nonlinearity	Single-ended mode	-1		2	LSB
INL	Integral nonlinearity	Single-ended mode	-2	±1.1	2	LSB
OffsetError	Misalignment error After calibration	Single-ended mode	-3	-0.1	3	LSB
GainError	Misalignment error After calibration	Single-ended mode	-5	-1.7	5	LSB

Table 3-15 ADC characteristics

3.5.9.3 Input channel impedance

The following figure shows the ADC input channel impedance distribution.

- ADC_Inx represents for fast external channels
- ADC_Iny represents for slow external channels
- R_{IO} represents the pin input switch impedance, R_{ADC1} and R_{ADC2} represent the ADC input fast channel impedance and slow channel impedance
- C_S represents the internal sampling capacitance of ADC, with a typical value of 3pF
- Refer to the following table for impedance parameters

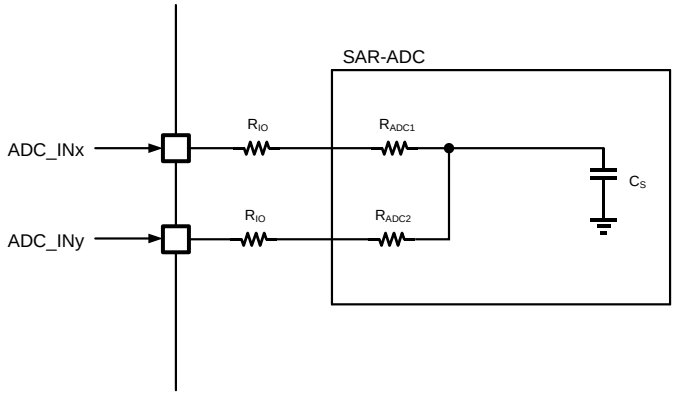


Figure 3-4 ADC channel input impedance

3.5.9.4 Sampling time

The minimum sampling time of ADC input signal is determined by the internal resistor of analog signal source, input channel impedance, pin parasitic capacitance and sampling capacitance.

The minimum sampling time of ADC sampling external input signal can be calculated according to the following formula:

$$T_{samp} = \ln\left(\frac{2^n}{SA}\right) \times \left(R_{AIN} + R_{ADC} + R_{IO}\right) \times C_{ADC}$$

Among them, n=12, SA=0.25LSB (The voltage on the sampling capacitor is established to within 0.25LSB error of the sampled signal level), R_{AIN} represents the internal resistor of the sampled signal source, R_{IO} represents input IO impedance, R_{ADC} represents ADC input channel impedance, C_{AD} represents ADC sampling capacitance. And R_{IO} is 100Ω。

R_{ADC} is affected by power supply voltage, temperature and input signal amplitude. When the input signal is VDDA/2, the switch impedance is the largest. The following table provides the R_{ADC} parameters when the input signal is VDDA/2 under different power and temperature conditions. The user can calculate the required minimum sampling time according to these parameters and the characteristics of the signal source.

Symbol	VDDA	Temperature	Value			Unit
			Min	Type	Max	
Fast channel, single-ended input						
R_{ADC}	5V	-40	-	890	1050	Ω
		55	-	1015	1203	
		125	-	1118	1340	
	3.3V	-40	-	1020	1220	

Symbol	VDDA	Temperature	Value			Unit
			Min	Type	Max	
		55	-	1200	1460	
		125	-	1346	1650	
	1.6V	-40	-	2260	3520	
		55	-	2500	3620	
		125	-	2660	3730	

Table 3-16 ADC input impedance

3.5.10 Temperature sensor

Temperature sensor is factory-trimmed under such condition: VDDA=3.0V and TA=30+/-1°C. Under this condition, ADC is used to sample and convert the temperature sensor output voltage. The conversion result is stored in Flash for user application.

Symbol	Parameter	Test conditions	Data storage address
TS_CAL1	Temperature sensor calibration value 1	VDDA=3.0V, T_A=30+/-1°C	TBD

Symbol	Parameter	Test conditions	Value			Unit
			Min	Typ	Max	
Reso	Resolution	VDDA=3.0V	-	0.3	-	°C/LSB
Slope	Output slope [1]		-	2.51	-	mV/°C
Linerity	Full temperature range linearity [1]	T_A=-40~125C	-	+/-3	+/-5	°C
I_{DDA}	Temperature sensor power consumption (without ADC) [2]	VDDA=3.0V	-	1	2	uA
t_{START}	Temperature sensor start-up time [2]		-	-	10	us
t_{SAMPLE}	Required sampling time for ADC to sample temperature sensor output [2]		10	-	-	us

Table 3-17 Temperature sensor parameter

[1]: Based on characterization

[2]: Based on simulation

3.5.11 Analog comparator characteristics

Symbol	Parameter	Test conditions	Value			Unit
			Min	Typ	Max	
VI_{comp1}	Comparator1 input voltage range		0	-	VDD	V
VI_{comp2}	Comparator2 input voltage range		0	-	VDD	V
I_{comp1}	Comparator1 operating current	VDD=5V	-	2		uA
I_{comp2}	Comparator2 operating current	VDD=5V	-	2		uA
T_{setup1}	Comparator1 start-up time	VDD=5V	-		15	us
T_{setup2}	Comparator2 start-up time	VDD=5V	-		15	us
$T_{propagation1}$	Comparator1 propagation delay	VDD=5V	-		2	us
$T_{propagation2}$	Comparator2 propagation deay	VDD=5V	-		2	us

Table 3-18 Analog comparator characteristics

3.5.12 Flash characteristics

Symbol	Parameter	Test conditions	Value			Unit
			Min	Type	Max	
	Flash size		64K	-	128K	bytes
T_{PROG}	Byte Program Time		6	-	7.5	μs
T_{ERASE}	Sector/Page Erase			2		ms
	Chip Erase			8		ms
N_{ED}	Sector Endurance	T_A=105℃	100,000			Erase/Write cycles
T_{DR}	Data Retention	T_A=105℃	10			yrs

Table 3-19 Flash characteristics

3.5.13 GPIO characteristics

General-purpose IO

Symbol	Parameter	Test Condition		Value			Unit
				Min	Typ	Max	
V_{IL}	Input Low voltage			0		$0.3V_{DD}$	V
V_{IH}	Input High voltage			$0.7V_{DD}$		V_{DD}	V
I_{IL}	Input Low Drain	V_{IL}=0V, T_A=40~125℃				0.5	μA
I_{IH}	Input high leakage	V_{IH}=5V, T_A=40~125℃				0.5	μA
V_{OL}	Output Low voltage[1]	V_{DD}=5V TA=-40~125℃	I_{SINK}=5mA	0.4	0.6	0.8	V
V_{OH}	Output High voltage[1]	V_{DD}=5V TA=-40~125℃	I_{SOURCE}=5mA	4.0	4.3	4.6	V
R_{PU}	Weak pull-up resistor				100		KΩ

Table 3-20 General I/O characteristics

Note [1]: Based on characterization

NRST pin

Symbol	Parameter	Test conditions	Value			Unit
			Min	Typ	Max	
V_{IL}	Input low voltage		0		$0.3V_{DD}$	V
V_{IH}	Input high voltage		$0.7V_{DD}$		V_{DD}	V
I_{IL}	Input low leakage	V_{IL}=0V			0.5	μA
I_{IH}	Input high leakage	V_{IH}=5V			0.5	μA
R_{PU}	Pull-up resistor			10		KΩ
$T_{AFILTER}$	Analog filter length[1]	VDD=5V		100		ns
$T_{DFILTER}$	Digital filter length[1]	VDD=1.8~5.5V -40℃≤T_A≤125℃	50		100	us

Table 3-21 NRST pin characteristics

Note [1]: Based on characterization

GPIO AC characteristics

IO	Symbol	Parameter[1]	Test conditions	min	max	Unit
Non-FM+	Fmax	Maximum frequency	C=30pF,2.7V<Vdd<3.6V	-	45	MHz
			C=30pF,1.6V<Vdd<2.7V	-	22	
			C=10pF,2.7V<Vdd<3.6V	-	80	
			C=10pF,1.6V<Vdd<2.7V	-	40	
	Tr/Tf	Output rise and fall time	C=30pF,2.7V<Vdd<3.6V	-	8.7	ns
			C=30pF,1.6V<Vdd<2.7V	-	16.9	
			C=10pF,2.7V<Vdd<3.6V	-	3.4	
			C=10pF,1.6V<Vdd<2.7V	-	6.7	
FM+	Fmax	Maximum frequency	C=50pF,1.6V<Vdd<3.6V	-	10	MHz
	Tf	Output fall time		-	27	ns

Table 3-22 Pin AC characteristics

Note:

[1] Based on simulation and are not included in the mass production test

4 Power Management Unit (PMU)

4.1 Power supply

4.1.1 Power domains

- VDD

The typical operating voltage rang of the main power supply (VDD) of the chip is 1.8V to 5.5V.

When the chip is powered on, the reset release threshold is mainly determined by the BOR circuit, whose typical reset release voltage is 1.8V. If the chip power supply rise time is very short(less than a few ms), then the reset release voltage is mainly determined by the RC delay, which will be slightly lower than 1.8V in typical cases and slightly higher than 1.8V at low temperature. BOR_PDRCFG can be configured by the software to obtain four power-down reset thresholds, with the default value of 1.6V. If BOR is not enabled and PDR is enabled, the power-down reset threshold is determined by the PDR circuit, and the software can configure three thresholds via PDRCFG, with a default value of about 1.4V.

In summary, the actual VDD voltage range of the chip will be determined jointly by BOR and PDR circuit configuration.

> ***Note:*** *BOR and PDR must not be turned off at the same time under any circumstance, as a normal reset may not occur when the chip powers down and the chip may not work properly when it is powered up again.*

- VDDA

The VDDA is a dedicated analog circuit power supply, which mainly supplies power to analog modules such as ADC, OPA, and reference voltage. The operating voltage range of VDDA is 1.8 to 5.5V, and all analog modules can be guaranteed to work normally within this voltage range.

- VREFP

Only a few package forms have independent VREFP pins, VREFP is the reference voltage input of the ADC. When the ADC is working, it will draw tens to hundreds of uA of current from the VREFP pin. In most package, VREFP is packaged with VDDA.

- VDD15

The VDD15 is a chip core power source that generates 1.5V power output from a linear regulator. All digital circuits, Flash, SRAM and some analog circuits operate under this power domain. It is recommended to connect a 0.1~1uF voltage regulator capacitor to the VDD15 pin. When the main power VDD falls below 1.5V, the output of the voltage regulator will follow the change of VDD.

4.1.2 Power supply structure

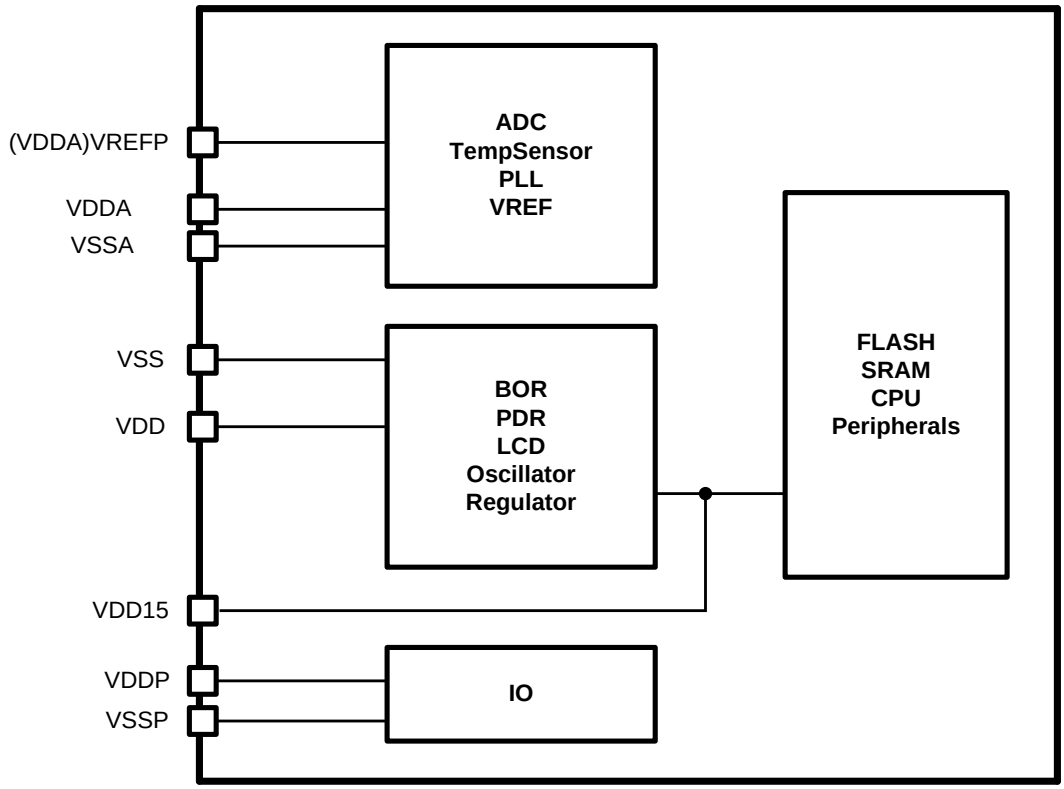


Figure 4-1 Chip power structure

4.1.3 ADC and independent power supply of internal reference

In order to improve the ADC conversion accuracy and reduce the influence of power supply noise, ADC and reference voltage use independent VREFP and VREFN pins to supply power. In the system, the VREFP/VREFN power supply can be filter independently, and the VREFP/VREFN traces can be shielded on the PCB to shield the system noise as much as possible.

However, in some low-pin-count packages, there may not be independent VREFP/VREFN. In this case, VREFP is connected with VDD inside the package, VREFN is connected with VSS inside the package, due to the influence of power ground noise, the ADC's performance will decrease.

When VREFP and VDDA are independent, the VREFP's input reference can be different from that of VDDA, the allowable input range is:

$$1.8\text{V} \leq \text{VREFP} \leq \text{VDD}$$

4.1.4 ON-chip reference source (AVREF)

FM33Lex0 integrates a high-accuracy reference source, which is typical output voltage is about 1.0V, and can normally work in the range of 1.8V to 5.5V.

After this reference voltage is output by Buffer, it can be sampled by ADC and also used for the reference voltage input of the comparator.

This reference source can be turned on or off by software. After turning on the reference source, the AVREF output setup time is less than 5us, and the typical power consumption is about 2uA. When the temperature sensor is turned on, the AVREF power consumption is less than 5uA.

The maximum temperature measurement range supported by the temperature sensor is -55~125°C, and the output voltage of the temperature sensor changes with temperature as a straight line with a positive temperature coefficient, with it's typical slope of 2.56mV/°C. Before the chip leaves the factory, the temperature sensor will be calibrated under the condition of 30°C +/- 1°C. Under this condition, the temperature measurement error in the range of -40~+125°C is within +/-5°C .

4.2 Consumption mode

4.2.1 Introduction

After power-on reset, the chip runs in ACTIVE mode by default. The CPU fetches instructions from Flash, and all peripherals can work normally. The chip supports a variety of low power modes, and the software can choose the appropriate low power mode in the appropriate scenario to balance the power consumption, performance, wake-up time and wake-up conditions.

Supported power modes：

- ACTIVE mode: Normal operation
- LP Active: LDO enters low-power mode, CPU frequency does not exceed 4MHz, all peripherals can work
- LP Run mode: LDO operate in ultra-low power mode, and CPU and peripherals can only operate at low-frequency
- SLEEP mode: CPU halts, Flash halts, LDO works in low-power mode, only some of the peripherals can work
- DEEPSLEEP mode: CPU halts, Flash halts, internal reference disabled, LDO works in low-power mode, only some of the peripherals can work

In addition, the consumptions can be reduced by the several methods below in ACTIVE mode.

- Lowering system clock frequency
- Turn off the bus clock and operating clock for unused peripherals

Power mode	Typical consumptions	Wakeup conditions	Chip status	Typical wakeup time[1]
ACTIVE	100uA/MHz		Full functions Dhrystone	-
LP Active	500uA	Software quit	LDO enters low-power mode Dhrystone	-
Active Sleep		Interrupt wake-up	CPU sleep	-

			LDO enter or not enter low-power mode	
LP Run	25uA@32KHz	Software quit	Low-speed work	-
SLEEP	4uA	SVD interrupt Comparator interrupt RTC interrupt IO interrupt WKUPx interrupt 32K crystal oscillator fail interrupt Watch dog reset NRST pin reset	CPU sleep[2] RCHF, XTHF, PLL, ADC disabled VREF disabled LDO15 disabled RTC operating	3us
DEEPSLEEP	1uA		CPU sleep[2] RCHF, XTHF, PLL, ADC disabled VREF disabled LDO15 disabled RTC operating	6.5us

Table 4-1 Power consumption mode

Note：

[1] Typical wake-up time means the time between the arrival of the wake-up event and the CPU's execution of the wake-up interrupt service routine.

[2] Refer to the ARMv6-M Architecture Reference manual for CPU sleep-entry.

[3] When the CPU tries to enter the low-power mode, if Flash is under erasing/programming, the chip will automatically wait for Flash to finish erasing/programming before entering the low-power mode.

The figure of the power consumption convert is shown as below:

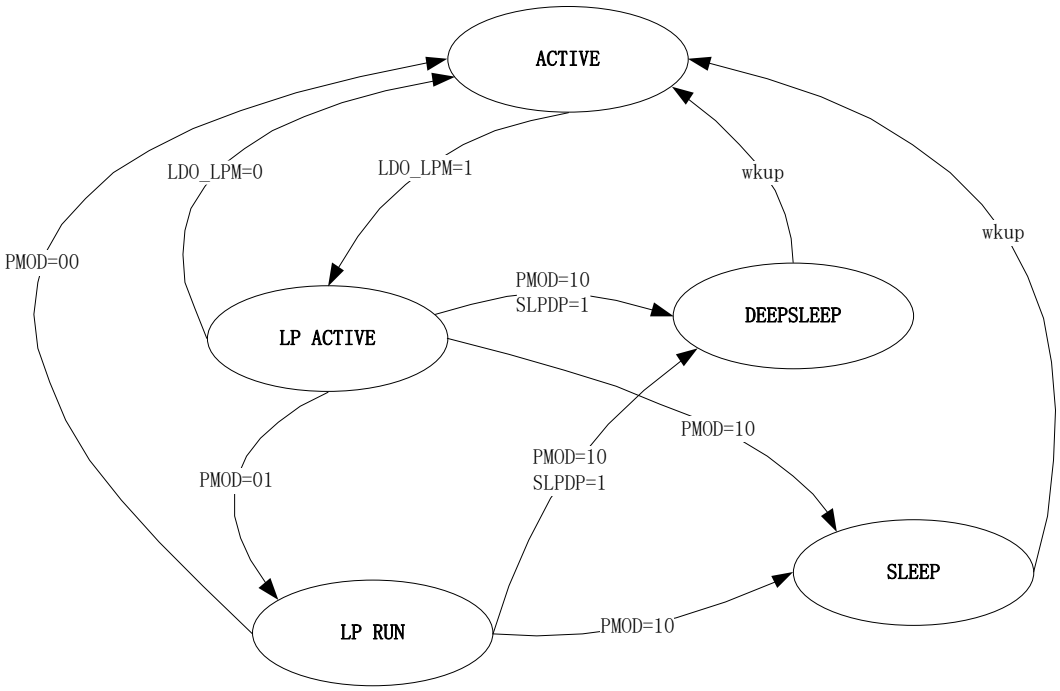


Figure 4-2 Power consumption mode convert

4.2.2 Power mode and system frequency

In different power mode, the range of system frequency is shown as below:

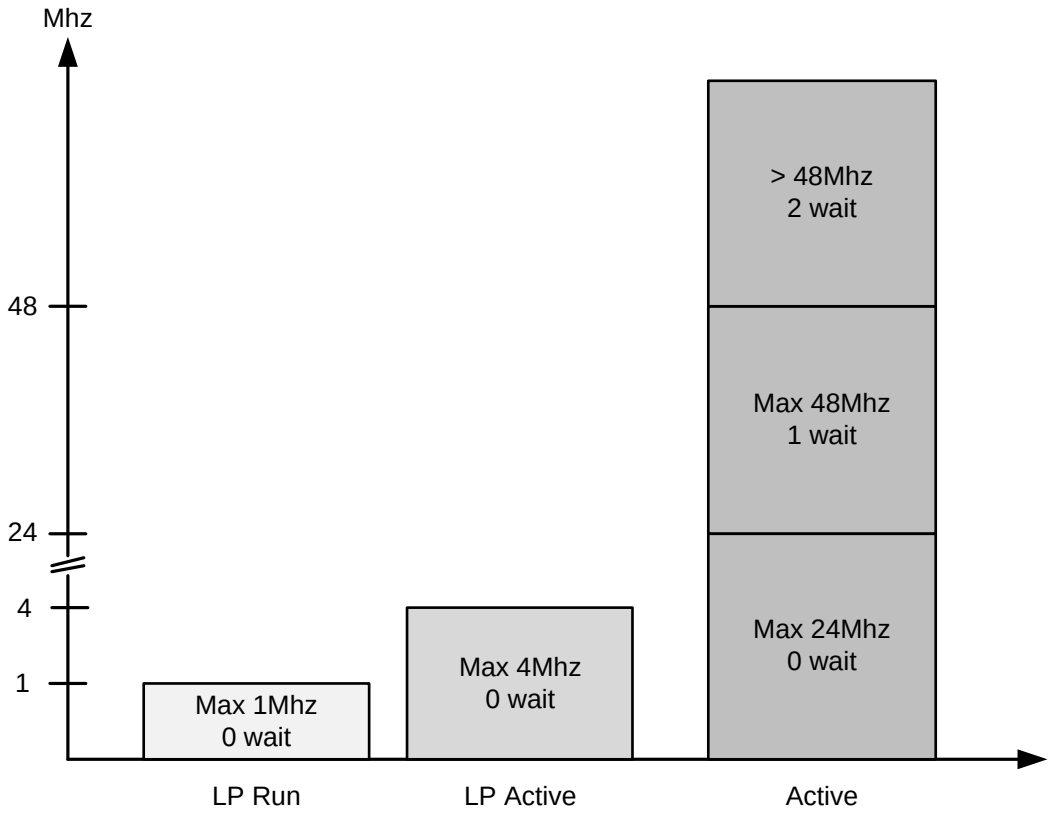


Figure 4-3 Consumption mode and system clock

The acceptable system frequency and available clock sources in different power modes are shown in the table below. Application software should strictly follow the rules of this table. Using high system frequency in low-power mode may cause the system to fail to function properly.

Power mode	CPU frequency	Available clock source	Flash wait	Peripheral
ACTIVE	<=24MHz	All	0	All
	>24MHz, <=48MHz		1	
	>48MHz		2	
LP Active	<=4MHz	RCHF, RCMF, XTLF, LPOSC	0	RCHF,RCMF, XTLF, LPOSC
LP Run	32KHz	XTLF, LPOSC	0	XTLF, LPOSC

Table 4-2 Consumption mode and available system clock

4.2.3 Active mode

The chip is in normal working mode. The chip will enter Active mode after power reset. The default CPU frequency is 8MHz and it can run up to 64MHz. All digital and analog peripheral can run at full speed in Active mode.

When the system frequency is higher than 24MHz, wait cycle must be inserted when accessing Flash, and Flash Prefetch is recommended to be enabled by the software to improve the efficiency of instruction execution.

4.2.4 LP Active mode

Software can enter LP Active mode by setting the LDO_LPM register. At this time, LDO is placed in a low-power mode, while its own power consumption is reduced, the driving capability is also reduced. Therefore, in LP Active mode, it is recommended that the CPU frequency should not exceed 4MHz. At the same time, the peripheral modules can still work with RCHF, RCMF, XTLF, LPOSC, but the PLL and XTHF are forcibly disabled by the hardware and cannot be used.

The typical application scenario of LP Active mode is to keep 1~2 peripherals (such as UART, Timer) running normally for a long time when the CPU is in standby or running at low speed in a scenario that does not require high CPU processing capabilities, so as to provide the best energy efficiency ratio for some special low-power scenario.

Entering LP Active mode：

- Set the system clock 4MHz or lower
  - Ensure that no peripherals are using XTHF or PLL clock
  - Configure the LDO_LPM register

Hardware behavior in LP Active mode：

After entering LP Active, the hardware automatically disables XTHF and PLL, and then LDO enters the low-power mode. All digital and analog peripheral can work.

Exiting LP Active mode：

- Software clears the LDO_LPM register
- Waiting for two NOP instructions
- Set the system clock as needed and resume normal Active mode operation. In LP Active mode, the chip can directly enter the LP RUN / SLEEP / DEEPSLEEP mode by rewriting the PMOD register by CPU.

4.2.5 LPRUN mode

When the chip needs low power and low speed operation, it can enter LPRUN mode. At this time, LDO enter slow-power mode, and the core uses LSCLK or RCMF for frequency division operation with a typical frequency of 32KHz. When high-speed operation is needed, the software can switch from LPRUN mode into ACTIVE mode, and then switch the system clock to a higher frequency.

Entering LPRUN mode

The steps of entering LPRUN mode:

- The software sets the system clock (SYSCLKSEL) to the LSCLK or RCMFPSC
- Configure the PMOD register as 01
- If the system clock configuration does not meet the register conditions above, error interrupt is flagged and access to the LPRUN is forbidden

Hardware behavior in LPRUN mode:

After entering LPRUN mode, the hardware automatically disables RCHF, XTHF, PLL and TRNG, and then LDO enters the low-power mode. SVD, comparator and ADC can still work in LPRUN mode. If the software executes the WFI instruction in LPRUN mode, the CPU and Flash will halt, but the peripherals will continue to work.

Exiting LPRUN mode:

- The software sets the PMOD register to 00
- Software enables RCHF or PLL as needed
- Configure the system clock to be RCHF or PLL after waiting for the clock start-up time

The CPU modifies the PMOD register in LPRUN mode to return to ACTIVE, or to enter SLEEP / DEEPSLEEP mode. If ACTIVE is returned, the hardware automatically puts the LDO in normal mode and unlocks the high-speed clock module.

4.2.6 SLEEP mode

By entering Sleep mode, you can significantly reduce the power consumption of the chip and keep in a state waiting for an event to wake up.

Entering SLEEP mode:

The software enters SLEEP mode according to the following steps:

- Configure the PMOD register to 10
- Execute WFI or WFE instruction

Hardware behavior in SLEEP mode:

After entering SLEEP mode, the chip will disable CPU clock, Flash will enter STOP mode, and the hardware will automatically disable RCHF, PLL, XTHF and TRNG. SVD, OPA and ADC can still work in SLEEP mode.

The digital peripherals can continue to work with low-speed clocks such as RCMF, XTLF and LPOSC.

Exiting SLEEP mode:

According to the steps below to exit the SLEEP mode:

- A specific interrupt event occurs
- The system clock is automatically configured as RCHF
- When the CPU is awakened, it can enter or not enter the interrupt service routine, depending on the software configuration

4.2.7 DEEPSEELP mode

DEEPSLEEP is the power mode with lowest current consumption. In DEEPSLEEP mode, due to the VREF is disabled, sleep power consumption is further reduced by about 2uA compared with SLEEP.

Entering DEEPSLEEP mode:

The software enters DEEPSLEEP mode according to the following steps:

- Set the VREFOFF register
- Configure the PMOD register to 10
- Execute WFI or WFE instruction

Hardware behavior in DEEPSLEEP mode:

In DEEPSLEEP mode, the chip will automatically disable the CPU clock and the internal reference source, Flash will enter the STOP mode, and the hardware will automatically disable RCHF, PLL and TRNG; SVD, OPA, ADC and comparator can still work in DEEPSLEEP mode.

The digital peripherals can continue to work with low-speed clocks such as RCMF, XTLF and LPOSC.

Exiting DEEPSLEEP mode:

Exiting the DEEPSLEEP mode as follows:

- A specific interrupt event occurs
- The system clock is automatically configured as RCHF

- When the CPU is awakened, it can enter or not enter the interrupt service routine, depending on the software configuration

4.3 Wake-up events

Wake-up events	Applications	Effective mode	
		Sleep	DeepSleep
Oscillation stop detection	Maskable, Wake up the chip when 32786Hz crystal fails	√	√
VREF	Maskable, Wake up the chip when an interrupt occurs after VREF1p22 is established	√	√
SVD	Maskable, Wake up the chip when the supply voltage falls below or rises above the threshold	√	√
Comparators	Maskable, Used for external event wake-up	√	√
ADC	Maskable, Various interrupts of ADC can be used to wake up	√	√
RTC	Maskable, Set the wake-up cycle as needed	√	√
IO pin interrupt	Maskable, Used for external event wake-up	√	√
Debug	Nonmaskable, For debug wake-up	√	√
LPUART	Maskable, Wake-up on data receiving	√	√
WKUPx pin	Maskable, Used for external input to wake up	√	√
NRST	Nonmaskable, For global reset	√	√
LPTIM32	Maskable, Used for periodic wake-up	√	√
BSTIM32	Maskable, Used for periodic wake-up	√	√
I2C slave	Maskable, Used for slave receiving wake-up	√	√

Table 4-3 Wake-up events and applications

By enabling PRIMASK feature in Cortex-M0, you can wake up the chip with the above interrupt events, but the CPU does not execute the interrupt handler. At this point, after waking up, the CPU will continue to run after the instruction before sleep.

Note: *After the chip is awakened from sleep mode, the software can quickly identify the current wake-up*

source by polling the PMU.WKFR register. The clear of wake-up sources needs to be performed individually for each peripheral module.

4.4 The system clock after wake-up

When the chip wakes up from Sleep/DeepSleep mode, the chip uses RCHF as the default clock source. The register will retain the frequency configuration and trim value of RCHF, so the system frequency after wake-up will be determined by the configuration register (PMU_CR.WKFSEL). In the fastest case, the chip will start with 24MHz clock after waking up.

When sleeping, the AHBPRES register will reset, and the SYSCLKSEL register will reset to 000 (select RCHF). Therefore, if the system clock is not RCHF before sleep, RCHF will be used by default when wake up.

4.5 Register

Offset address	Name	Symbol
PMU(**Base address:** 0x40000100)		
0x00000000	Power Management Control Register	PMU_CR
0x00000004	Wakeup Time Register	PMU_WKTR
0x00000008	Wakeup Source Flags Register	PMU_WKFR
0x0000000C	PMU Interrupt Enable Register	PMU_IER
0x00000010	PMU Interrupt and Status Register	PMU_ISR

4.5.1 Low-power consumption control register (PMU_CR)

NAME	PMU_CR							
offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-				LDO_LPM		LDO15EN	LDO15EN_B
access	U-0				R/W-01		R-1	R-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				WKFSEL		SLPDP	CVS
access	U-0				R/W-00		R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				RFU		PMOD	
access	U-0				R/W-00		R/W-00	

bit	Name	Description
31:20	**-**	**Reserved，read as 0**
19:18	**LDO_LPM**	LDO Low-power mode configuration 00/01/11：Normal mode 10：LDO enters low-power mode **Note:** When the chip is erasing and programming FLASH, this register cannot be rewritten to 10.
17	**LDO15EN**	LDO15 Enable flag 1：LDO15 is enabled 0：LDO15 os disabled
16	**LDO15EN_B**	The inversed check bit for LDO15 EN (LDO Enable Inversed)

bit	Name	Description
15:12	-	Reserved, read as 0
11:10	**WKFSEL**	Sleep/DeepSleep system frequency after wake up 00：RCHF-8MHz 01：RCHF-16MHz 10：RCHF-24MHz 11：RFU
9	**SLPDP**	DeepSleep control register(Sleep Deep) 1：Enable DeepSleep，Disable reference voltage source 0：Normal Sleep mode In the Sleep mode, if the SLPDP is set, it will be switch into DeepSleep mode. This bit is only valid in Sleep mode
8	**CVS**	Core-Voltage-Scaling configuration 0：Core voltage regulation is disable in low-power mode 1：Reduce core voltage in low-power mode This bit is only valid in Sleep/DeepSleep mode. Configuration of 1 is not recommended
7:4	-	**Reserved, read as 0**
3:2	**RFU**	Dummy register
1:0	**PMOD**	Low-power mode register 00：Active mode / LP Active mode 01：LPRUN mode 10：Sleep mode / DeepSleep mode 11：RFU

4.5.2 Wakeup time control register (PMU_WKTR)

NAME	PMU_WKTR							
offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					STPCLR	T1A	
access	U-0					R/W-0	R/W-01	

Bit	Name	Description
31:3	-	**Reserved，read as 0**
2	**STPCLR**	Flash Stop wake up control (Stop clear) 0：Stop waits for the clock to be synchronously cleared 1：Stop signal is asynchronously cleared
1:0	**T1A**	Programmable extra wake up delay In the Sleep/DeepSleep mode, when the RCHF clock arrives, wait for additional delay time according to this register configuration 00：0us 01：2us 10：4us 11：8us

4.5.3 Wakeup source flag register (PMU_WKFR)

NAME	PMU_WKFR							
offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	ADCWKF	-		RTCWKF	SVDWKF	LFDETWKF	-	IOWKF
access	R-0	U-0		R-0	R-0	R-0	U-0	R-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	I2CWKF	LINWKF	LPU1WKF	LPU0WKF	-		COMP2WKF	COMP1WKF
access	R-0	R-0	R-0	R-0	U-0		R-0	R-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	UART5WKF	UART4WKF	UART2WKF	UART1WKF	UART0WKF	LPTWKF	BSTWKF	DBGWKF
access	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	WKPxF							
Access	R/W-0000 0000							

Bit	Name	Description
31	**ADCWKF**	ADC interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (ADC 82lock82 Flag, auto to clear)
30:29	-	**Reserved, read as 0**
28	**RTCWKF**	RTC interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (RTC wakeup flag, auto to clear)
27	**SVDWKF**	SVD interrupt wake up flag, hardware automatically clear

Bit	Name	Description
		when the interrupt is cancelled (SVD wakeup flag, auto to clear)
26	**LFDETWKF**	32768Hz crystal fail interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (XTLF fail detect wakeup flag, auto to clear)
25	**I2CSMBWKF**	I2CSMB interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (I2CSMB wakeup flag, auto to clear)
24	**IOWKF**	IO interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (GPIO wakeup flag, auto to clear)
23	**I2CWKF**	I2C interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (I2C wakeup flag, auto to clear)
22	**LINWKF**	LIN interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (LIN wakeup flag, auto to clear)
21	**LPU1WKF**	LPUART1 interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (LPUART1 wakeup flag, auto to clear)
20	**LPU0WKF**	LPUART0 interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (LPUART0 wakeup flag, auto to clear)
19:18	**-**	**Reserved, read as 0**
17	**COMP2WKF**	Comparator 2 interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (comparator2 wakeup flag, auto to clear)
16	**COMP1WKF**	Comparator 1 interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (comparator1 wakeup flag, auto to clear)
15	**UART5WKF**	UART5 interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (UART5 wakeup flag, auto to clear)
14	**UART4WKF**	UART4 interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (UART4 wakeup flag, auto to clear)
13	**UART2WKF**	UART2 interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (UART2 wakeup flag, auto to clear)
12	**UART1WKF**	UART1 interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (UART1 wakeup flag, auto to clear)
11	**UART0WKF**	UART0 interrupt wake up flag, hardware automatically clear

Bit	Name	Description
		when the interrupt is cancelled (UART0wakeup flag, auto to clear)
10	**LPTWKF**	LPTIM32 interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (LPTIM wakeup flag, auto to clear)
9	**BSTWKF**	BSTIM32 interrupt wake up flag, hardware automatically clear when the interrupt is cancelled (BSTIM wakeup flag, auto to clear)
8	**DBGWKF**	CPU Debugger wake up flag, software writes 1 to clear
7:0	**WKPxF**	NWKUPx Pin wake up flag, software writes 1 to clear

4.5.4 PMU interrupt enable register (PMU_IER)

NAME	PMU_IER							
offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				LPRUNEIE	LPACTEIE	SLPEIE	-
access	U-0				R/W-0	R/W-0	R/W-0	U-0

Bit	Name	Description
31:4	**-**	**Reserved, read as 0**
3	**LPRUNEIE**	LPRUN mode error interrupt enable 1: Enable LPRUN error interrupt 0: Disable LPRUN error interrupt
2	**LPACTEIE**	LPACTIVE mode error interrupt enable 1: Enable LPACTIVE error interrupt 0: Disable LPACTIVE error interrupt
1	**SLPEIE**	SLEEP error interrupt enable 1: Enable SLEEP error interrupt 0: Disable SLEEP error interrupt
0	**-**	RFU

4.5.5 PMU interrupt flag register (PMU_ISR)

NAME	PMU_ISR							
offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				LPRUNEIF	LPACTEIF	SLPEIF	-
access	U-0				R/W-0	R/W-0	R/W-0	U-0

Bit	Name	Description
31:4	-	**Reserved, read as 0**
3	**LPRUNEIF**	LPRUN error interrupt flag, hardware set, software writes 1 to clear 1：When PMOD=2'h1, set when the system clock does not meet the definition of LPRUN mode 0：When PMOD=2'h1,the system clock meets the definition of LPRUN mode
2	**LPACTEIF**	LPACTIVE error interrupt flag, hardware set, software writes 1 to clear 1：When LDO15LPM=1'h1, set when the system clock does not meet the definition of LPACTIVE mode, that is, when the system is RCHF and greater than 4M,or when the system clock is XTHF/PLL 0：When LDO15LPM=1'h1, the system clock meets the definition of LPACTIVE mode
1	**SLPEIF**	SLEEP error interrupt flag, hardware set, software writes 1 to clear 1：When PMOD=2'h2, set when the SLEEPDEEP register is set before the CPU performs the WFI/WFE instruction 0：When PMOD=2'h2, the CPU enters SLEEP correctly0：

Bit	Name	Description
0	-	RFU

5 Internal Reference Source (VREF)

5.1 Introduction

FM33LE0xxA integrates a high-precision reference source with a typical output voltage of about 1.0V, which can work stably within the entire operating power supply range of the chip. After this reference voltage is output by the Buffer, it can be sampled by the ADC and also used as the reference voltage input of the comparator.

In the entire operating temperature range, the typical temperature coefficient of this reference source is less than 100ppm/℃, and a built-in temperature sensor output is provided for ADC sampling and measuring the current chip's substrate temperature.

AVREF is always enabled when the chip is in ACTIVE mode.

The maximum temperature measurement range supported by the temperature sensor is -55~125℃, the output voltage of the temperature sensor changes with temperature as a straight line with a positive temperature coefficient, with Typical slopes are shown in the electrical parameters section. Before the chip leaves the factory, the temperature sensor will be calibrated under the condition of 30℃±1℃. Under this condition, the temperature measurement error in the range of -40~+125℃is within ±5℃.

5.2 Register

Offset	Name	Symbol
VREF (Base address: 0x4001A800)		
0x0000000C	VREF PTAT Control Register	VREF_PTAT_CR
0x00000010	-	
0x00000014	-	
0x00000018	Buffer Control Register	VREF_BUFCR

5.2.1 VREF_PATA control register (VREF_PATA_CR)

NAME	VREF_CR							
offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						PTAT_EN	VREF_EN
access	U-0						R/W-0	R/W-0

Bit	Name	Description
31:2	-	RFU: Reserved, read as 0
1	PTAT_EN	Band gap temperature sensor enable 0: Disable temperature sensor output 1: Enable temperature sensor output
0	-	RFU: Reserved, read as 0

5.2.2 Buffer control register (VREF_BUFCR)

NAME	VREF_BUFCR							
offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				VPTATBUFFER_OUTEN	VPTATBUFFER_EN	VREFBUFFER_OUTEN	VREFBUFFER_EN
access	U-0				R/W-0	R/W-0	R/W-0	R/W-0

Bit	Name	Description
31:4	-	RFU: Reserved, read as 0
3	VPTATBUFFER_OUTEN	Vptat Buffer module switch channel output enable signal, high level enable is effective (PTAT Buffer Output Enable)
2	VPTATBUFFER_EN	Vptat Buffer module enable signal, high level enable is effective (PTAT Buffer Enable)
1	VREFBUFFER_OUTEN	Vref Buffer module switch channel output enable signal, high level enable is effective (VREF Buffer Output Enable)
0	VREFBUFFER_EN	Vref Buffer module enable signal, high level enable is effective (VREF Buffer Enable)

6 CPU

6.1 Introduction

FM33LE0xxA implements Cortex-M0+ processor, which conforms to ARMv6-M architecture and programming model. For more information, please refer to ARM's website at www.arm.com

Main features:

- Unprivileged/Privileged
- VTOR (interrupt vector table remapping)
- NVIC supports 32 external interrupts
- 1 data watchpoints
- 4 breakpoints
- Single cycle 32-bit hardware multiplier
- SWD debug interface

6.1.1 Processor configuration

Feature	Options	FM33LE0xxA Config
Interrupts	1~32	32
Data endianness	little/big	little
SysTick Timer	Present or absent	Present
watchpoints	0,1,2	1
breakpoints	0,1,2,3,4	4
halting debug support	Present or absent	Present
multiplier	Fast or Small	Fast
Single-Cycle IO	Present or absent	Absent
wake-up interrupt controller(WIC)	Present or absent	Present
Vector Table Offset Register	Present or absent	Present
Unprivileged/Priviledged support	Present or absent	Present
JTAGnSW	JTAG or SWD for DAP	SWD
Memory Protection Unit	Present or absent	Absent

Table 6-1 FM33LC0xx processor configuration

6.2 Register

Core register list:

Name	Descriptions
R0-R12	General register
MSP (R13)	Stack pointer; MSP (Main Stack Pointer) is used in Handler mode, and MSPor PSP (Process Stack Pointer) is selected by CONTROL register in Thread mode
PSP (R13)	
LR (R14)	The Link register holds the return information for subfunctions/function calls/exception handling
PC (R15)	The program pointer
PSR	Application state (APSR), interrupt program state (IPSR), and program execution state (EPSR)
PRIMASK	PRIMASK is used to mask any interrupt which has equal or lower priority than certain pre-configured level
CONTROL	Selects the stack pointer to be used in Threadmode

Table 6-2 Cortex-M0 core registers

Detailed definitions of registers are in the ARMv6-M Architecture Reference manual.

6.3 Exceptions and interrupts

Core exceptions and interrupts management is performed by NVIC. The programmable management register of NVIC is located in the SCS space of PPB bus. NVIC has the following features.

- 32 external interrupts and 5 internal exceptions are supported
- 1 NMI interrupt
- Supports for interrupt nesting
- Vectorized exception entry
- Interrupt mask

When processor core accepts an exception request, it first pushes the core registers R0~R3, R12, R14, PC, and xPSR onto the stack. The link register LR (R14) is updated to the special value used for exception return (EXC_RETURN), and the exception handler pointed by vector table starts to be executed. Note that registers which are not automatically saved to stack must be saved and restored by software.

6.3.1 Interrupt vector table

Position	Priority	Priority type	Acronym	Description	Address
0	-	-	MSP	Main stack pointer	0x0000_0000

Position	Priority	Priority type	Acronym	Description	Address
				initialization address	
1	-3	fixed	Reset	Reset vector	0x0000_0004
2	-2	fixed	NMI	WKUPx interrupt Low power mode error interrupt	0x0000_0008
3	-1	fixed	HardFault	HardFault interrupt vector	0x0000_000C
4-10	-	-	-	Reserved	0x0000_0010~0x0000_002B
11	3	settable	SVC	SVCall system service request	0x0000_002C
12-13	-	-	-	Reserved	0x0000_0030~0x0000_0037
14	5	settable	PendSV	Suspendable system service request	0x0000_0038
15	6	settable	Systick	Internal timer interrupt vector	0x0000_003C
16	7	settable	WWDT	Window watchdog interrupt	0x0000_0040
17	8	settable	SVD	Power monitoring alarm interrupt	0x0000_0044
18	9	settable	RTC	Real-time clock interrupt	0x0000_0048
19	10	settable	FLASH	NVMIF interrupt	0x0000_004C
20	11	settable	FDET	XTLF or XTHF stop detection interrupt	0x0000_0050
21	12	settable	ADC	ADC conversion completion interrupt	0x0000_0054
22	13	settable	-	IWDT interrupt	0x0000_0058
23	14	settable	SPI1	SPI interrupt	0x0000_005C
24	15	settable	SPI2		0x0000_0060
25	16	settable	LCD	LCD display module interrupt	0x0000_0064
26	17	settable	UART0	UART interrupt	0x0000_0068
27	18	settable	UART1		0x0000_006C
28	19	settable	UART4		0x0000_0070
29	20	settable	UART5		0x0000_0074
30	21	settable	HFDET	High frequency crystal stop detection interrupt	0x0000_0078
31	22	settable	U7816	U7816 Interrupt	0x0000_007C
32	23	settable	LPUART1	LPUART1 Interrupt	0x0000_0080
33	24	settable	I2C	I2C Interrupt	0x0000_0084
34	25	settable	I2C_SMBUS	I2C_SMBUS interrupt	0x0000_0088
35	26	settable	AES	AES interrupt	0x0000_008C
36	27	settable	LPTIM32	Low Power Timer Interrupt	0x0000_0090
37	28	settable	DMA	DMA interrupt	0x0000_0094
38	29	settable	WKUPx	WKUP pin interrupt	0x0000_0098
39	30	settable	UART2	UART2 Interrupt	0x0000_009C
40	31	settable	BSTIM32	Basic Timer Interrupts	0x0000_00A0
41	32	settable	COMPx	COMPx interrupt	0x0000_00A4

Position	Priority	Priority type	Acronym	Description	Address
42	33	settable	GPT1	Universal Timer 1 Interrupt	0x0000_00A8
43	34	settable	GPT2	General purpose timer 2 interrupt	0x0000_00AC
44	35	settable	ATIM1	Advanced Timer 1 interrupt	0x0000_00B0
45	36	settable	-	-	0x0000_00B4
46	37	settable	EXTI	External Pin Interrupts	0x0000_00B8
47	38	settable	LPUART0	LPUART0 Interrupt	0x0000_00BC

Table 6-3 FM33LC0xx interrupt vector table

WKUP interrupts can be connected to the NMI or 38# entry. The interrupt entry address is selected through the WKSEL register of the PMU module. When configured as a 38# entry, WKUPx interrupt can be masked by PRIMASK, and the CPU will not enter the interrupt service routine after wake up, but continue to execute from the last instruction before Sleep.

6.3.2 Interrupt priority level

The processor supports three fixed highest priorities and four programmable priorities. When two exceptions of the same priority occur at the same time, the exception with the smaller exception number is executed first.

6.3.3 Error handing

The processor supports only one method of handling hardware errors: Hard Fault exceptions. Hard Fault priority is -1, which means only NMI can preempt. Hard Fault can be triggered by following situations:

Type	Conditions
Memory error	Bus error. A bus error due to the use of an illegal address in the bus transmission.
	An attempt was made to execute the program within the XN region
Program error	Execute an undefined instruction
	Attempt to switch to ARM state
	Unaligned memory access Execute the SVC instruction within higher priority exception handling The EXC_RETURN value is invalid when performing an exception return An attempt was made to execute the BKPT instruction when debugging was not enabled

Table 6-4 HardFault types

The HardFault trigger sources of FM33LE0xxA can be queried through registers to help software developers locate the cause of the error.

6.3.4 Lockup

When the processor has another HardFault during HardFault handling, or HardFault occurs during NMI handling, the processor enters a locked state (stops execution) and outputs a LOCKUP signal, at which point the chip automatically resets the processor core rather than waiting for the watchdog to overrun.

6.4 Debug features

Following debug features are supported:

- Halt, resume and single-step execution of the program
- Access to core registers and special registers
- Hardware breakpoint (4)
- Software breakpoint (Unlimited number of BKPT instructions)
- Data watch point (1)
- Dynamic non-intrusive memory access (no need to stop the processor)
- SWD interface

Debugging features of Cortex-M0 are based on ARM CoreSight debugging Architecture. Please refer to CoreSight Technology System Design Guide and ARM Debug Interface Architecture Specification ADIv5.0 to ADIv5.2 for details.

6.4.1 Debug function PIN

FM33LE0xxA uses SWD debugging interface. In user mode, at least 4 lines (NRST, GND, SWIO, SWCLK) are needed to realize debugging function. The 2-wire debug pins can be reused as GPIO, and their functions are configured by software selection.

NRST pin is used to reset the chip. Through the cooperation of NRST and SWD, the debugger can halt CPU at the very first instruction.

See The I/O Control section for the use of debug pins.

6.4.2 Watchdog control in debug mode

The watchdog can remain enabled or be disabled in debug mode. Software or Debugger can

configure the watchdog to run or stop with the MCUDBGCR register.

6.4.3 DEBUG reset

The DEBUG part of the kernel is only affected by POR and PDR. Other system reset sources, such as watchdog, pin reset and software reset, will not reset the DAP circuit.

This allows the CPU kernel to be reset by pin reset after the chip is powered on, but the debugger can still communicate with the DAP normally and set breakpoints, and immediately put the CPU into debug mode after the reset is released.

It is recommended that the debugger connect to the kernel during system reset (setting breakpoints at the reset vector).

6.5 Register

Offset	Name	Symbol
DBG (Base address: 0x40000000)		
0x00000004	Debug Configuration Register	DBG_CR
0x00000008	HardFault Flag Register	DBG_HDFR

6.5.1 DEBUG Configuration register (DBG_CR)

FM33LC0XXextends the MCUDBGCR register to configure watchdog and timer in Debug mode. The MCUDBGCR register can be overwritten by SWD interface or software.

NAME	DBG_CR							
offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-		AT_STOP	LPT_STOP	GT2_STOP	GT1_STOP	-	BT1_STOP
access	U-0		R/W-1	R/W-1	R/W-1	R/W-1	U-0	R/W-1
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						WWDT_STOP	IWDT_STOP
access	U-0						R/W-1	R/W-1

Bit	Name	Description
31:18	-	RFU：Reserved, read as 0
17	DBG_RCENB	Debug state whether to support read-zero operation, only valid for peripheral registers with read 0 function 1: Disable read 0 0: Read 0 is allowed
16:14	-	RFU：Reserved, read as 0
13	AT_STOP	Stop ATIM under Debug Enable 1: Close ATIM when Debug 0: Keep ATIM in original state when Debug
12	LPT_STOP	Stop LPTIM32 under Debug Enable 1: Close LPTIM32 when Debug 0: Keep LPTIM32 in original state when Debug
11	GT1_STOP	Stop GPTIM1 under Debug Enable) 1：Close GPTIM1 when Debug 0：Keep GPTIM1 in original state when Debug
10	GT0_STOP	Stop GPTIM0 under Debug Enable) 1：Close GPTIM0 when Debug 0：Keep GPTIM0 in original state when Debug
9	-	RFU：Reserved, read as 0
8	BT_STOP	Stop BSTIM under Debug Enable) 1：Close BSTIM32 when Debug 0：Keep BSTIM32 in original state when Debug
7:2	-	RFU：Reserved, read as 0
1	WWDT_STOP	Stop WWDT under Debug Enable) 1：Close WWDT when Debug 0：Keep WWDT in original state when Debug
0	IWDT_STOP	Stop IWDT under Debug Enable 1：Close IWDT when Debug 0：Keep IWDT in original state when Debug

6.5.2 HardFault query register (DBG_HDFR)

NAME	DBG_HDFR							
offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	DABORT_ADDR_FLAG	DABORT_RESP_FLAG	SVCUNDEF_FLAG	BKPT_FLAG	TBIT_FLAG	SPECIAL_OP_FLAG	HDF_REQUEST_FLAG
access	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

Bit	Name	Description
31:7	-	RFU: Reserved, read as 0
6	DABORT_ADDR_FLAG	Address unaligned access error flag, write 1 to clear (Debug Abort Flag for misaligned Address) 1: Address unaligned" access error 0: Address unaligned access was not performed
5	DABORT_RESP_FLAG	Illegal address access error flag, write 1 to clear (Debug Abort Flag for HRESP) 1: An illegal address was accessed in the bus transmission 0: No illegal address was accessed
4	SVCUNDEF_FLAG	Undefined label of SVC instructions, write 1 to clear (SVC undefined instruction Flag) If the SVCall priority is lower than the currently active level, or if HardFault or NMI is active, or PRIMASK is set, the core should treat SVC instructions as though they were UNDEFINED.
3	BKPT_FLAG	Execute the BKPT command flag and write 1 to clear (Break point instruction Flag) 1: Execute the BKPT instruction 0: BKPT instruction is not executed
2	TBIT_FLAG	Thumb-State flag, write 1 to clear (Thumb state Flag) 1: Switch to ARMstate 0: In Thumb-State
1	SPECIAL_OP_FLAG	Special command flag, write 1 to clear (Special OP code Flag) 1: Special instruction executed, such as trying to fetch in the XN region 0: No special instruction is executed
0	HDF_REQUEST_FLAG	Hardfault flag bit, any type of hardfault will set this bit, write 1 to clear (Hardfault Request Flag) 1: Hardfault request 0: No hardfa ult request

7 Bus and Memories

7.1 System bus

The FMLE0 bus architecture consists of the following main components：

- 2 Masters
  - Cortex-M0
  - DMA controller
- 5 Slaves
  - Internal Flash
  - Internal SRAM
  - GPIO controller module
  - System control module
  - AHB-APB bus transfer bridge

The system bus diagram of FM33LE0xxA is as follows, including an AHB-Lite bus and an APB bus.

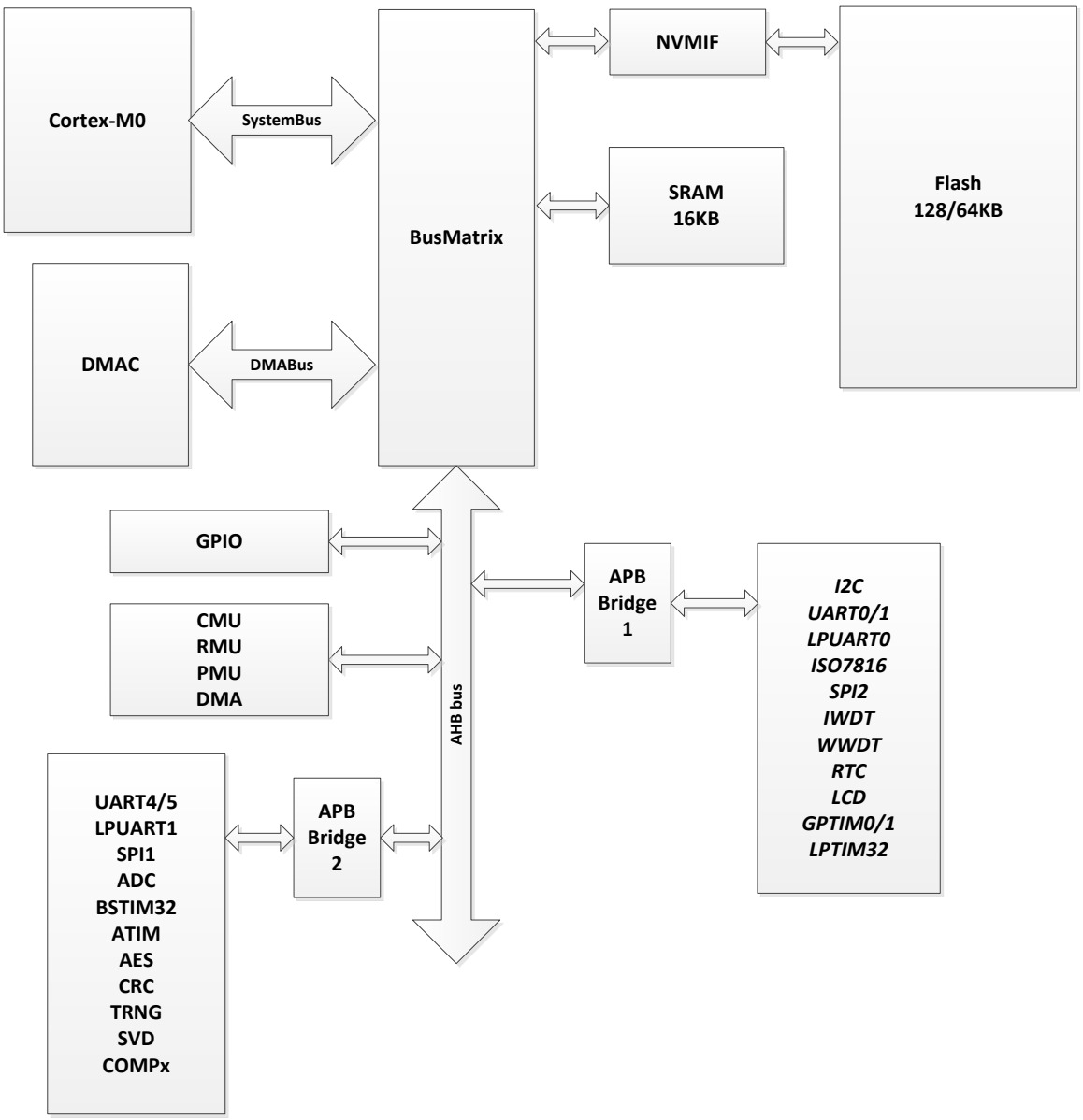


Figure 7-1 System bus diagram

7.2 Memory space allocation

7.2.1 Introduction

The Flash Sector size is 512 bytes, 16 sectors can constitute a Block of 8K bytes.

Flash contains 4 INFO pages, 2 LDT pages, 1 redundant page, and 1 DCT page. Among them, DCT and LDT are reserved and are not available to users. INFO pages are used to hold user configuration Information. All Option pages are logically isolated from the Flash main area.

The address space of FM33LC0XX is allocated as follows (128KB Flash, 16KB RAM)

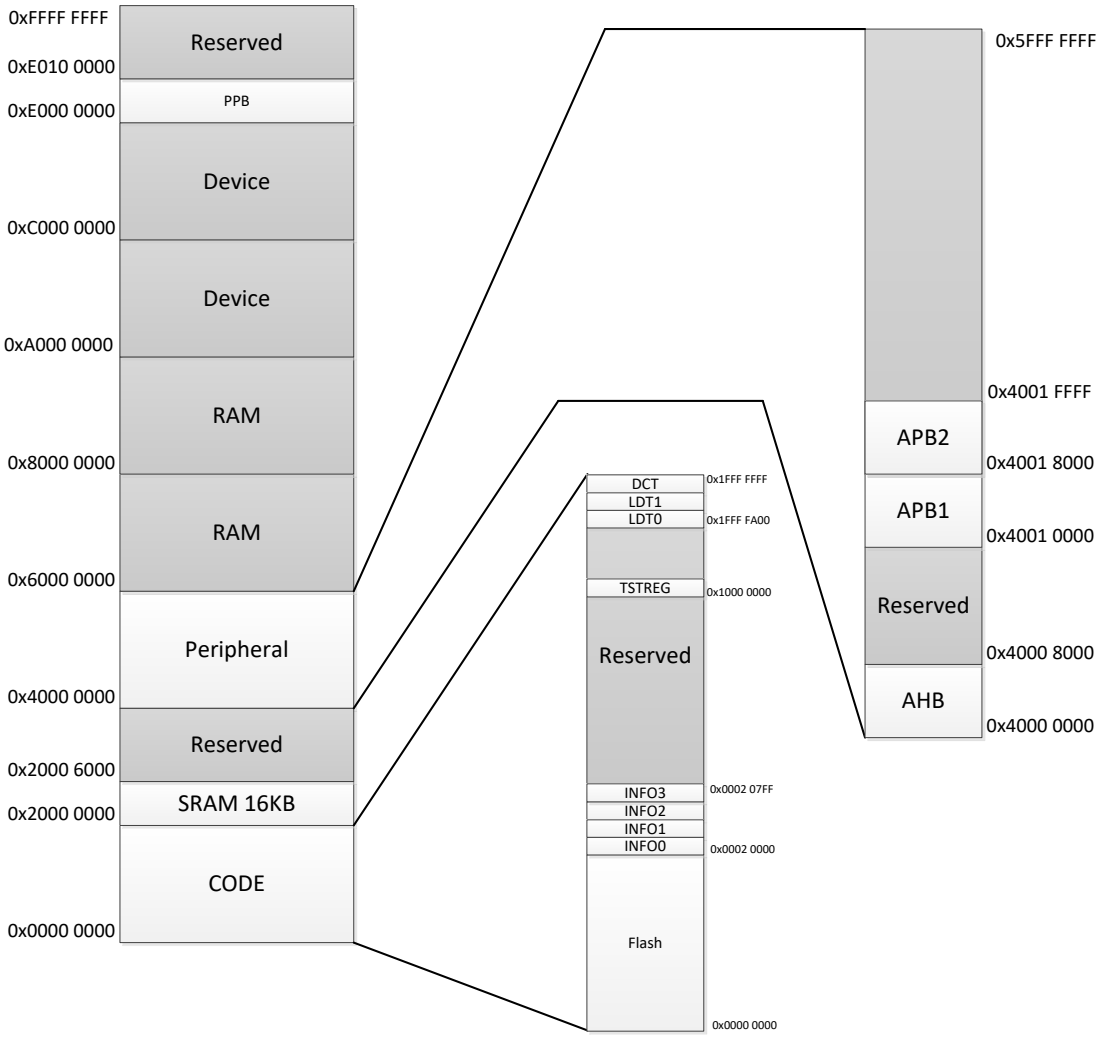


Figure 7-2 FM33LE04xA bus address

64KB Flash + 16KB RAM：

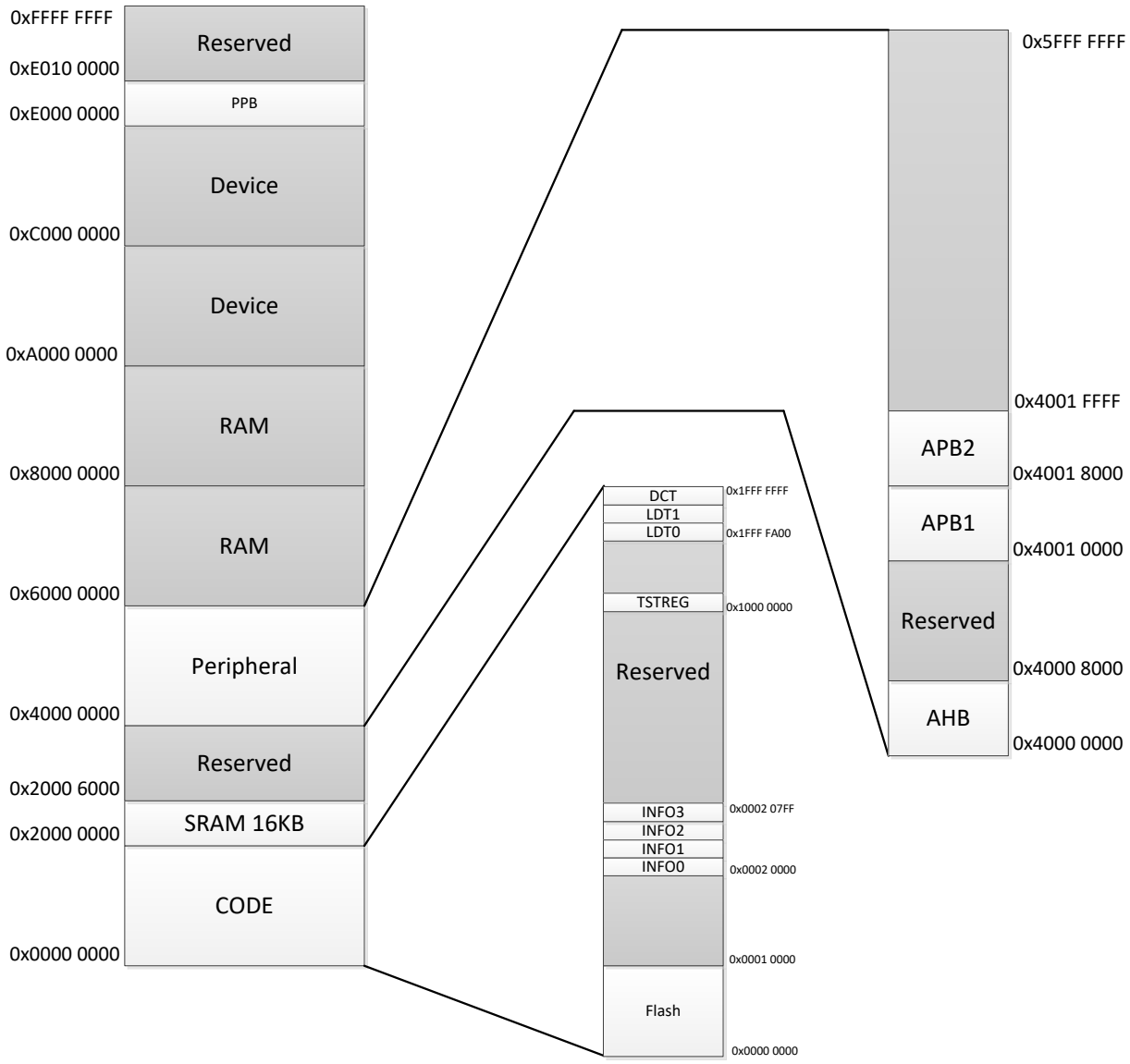


Figure 7-3 FM33LE02xA bus address

7.2.2 Peripherals address assignment

The following table lists the address space allocation for all peripherals, each occupying 1KB of address space.

Bus	Boundary address	Size	Peripheral
AHB	0x4000_0C00~0x4000_0FFF	1KB	GPIO
	0x4000_0000~0x4000_03FF	1KB	SCU, PMU, CMU, RMU
	0x4000_0400~0x4000_07FF	1KB	DMA
	0x4000_1000~0x4000_13FF	1KB	NVMIF
	0x5000_0000~0x5001_FFFF		
	0x5002_0000~0x5003_FFFF		

	0x0000_0000~0x0001_FFFF	128KB	Flash main array
	0x1FFF_F000~0x1FFF_FFFF	4KB	Flash Option cell array
	0x2000_0000~0x2000_3FFF	16KB	SRAM
	0x4001_0000~0x4001_7FFF	32KB	APB1
	0x4001_8000~0x4001_FFFF	32KB	APB2
APB1	0x4001_0000~0x4001_03FF	1KB	ISO7816
	0x4001_0400~0x4001_07FF	1KB	LPUART0
	0x4001_0800~0x4001_0BFF	1KB	SPI2
	0x4001_0C00~0x4001_0FFF	1KB	LCD
	0x4001_1000~0x4001_13FF	1KB	RTC
	0x4001_1400~0x4001_17FF	1KB	IWDT
	0x4001_1800~0x4001_1BFF	1KB	WWDT
	0x4001_1C00~0x4001_1FFF	1KB	UART0
	0x4001_2000~0x4001_23FF	1KB	UART1
	0x4001_2400~0x4001_27FF	1KB	I2C
	0x4001_2800~0x4001_2BFF	1KB	I2C_SMBUS
	0x4001_2C00~0x4001_2FFF	1KB	RAMBIST
	0x4001_3000~0x4001_33FF	1KB	UART2
	0x4001_3400~0x4001_37FF	1KB	LPTIM32
	0x4001_3800~0x4001_3BFF	1KB	GTIMER0
	0x4001_3C00~0x4001_3FFF	1KB	GTIMER1
APB2	0x4001_8000~0x4001_83FF	1KB	CRC
	0x4001_8400~0x4001_87FF	1KB	LPUART1
	0x4001_8800~0x4001_8BFF		Reserved
	0x4001_8C00~0x4001_8FFF	1KB	SPI1
	0x4001_9000~0x4001_93FF		Reserved
	0x4001_9400~0x4001_97FF		Reserved
	0x4001_9800~0x4001_9BFF		Reserved
	0x4001_9C00~0x4001_9FFF		Reserved
	0x4001_A000~0x4001_A3FF	1KB	UART4
	0x4001_A400~0x4001_A7FF	1KB	UART5
	0x4001_A800~0x4001_ABFF	1KB	SVD ,COMP
	0x4001_AC00~0x4001_AFFF	1KB	ADC
	0x4001_B000~0x4001_B3FF	1KB	ATIM
	0x4001_B400~0x4001_B7FF	1KB	BSTIM32
	0x4001_B800~0x4001_BBFF	1KB	AES
	0x4001_BC00~0x4001_BFFF	1KB	TRNG

Table 7-1 Peripherals address assignment

7.3 RAM

7.3.1 Introduction

The FM33LE0xxA contains a 16KB RAM (4K*32), and the main indicators are as follows:

Target	Parameter
Words	4096
Bits	32
Mux	
Width	
Height	
Size	TBD
Leakage	TBD

7.4 Flash

7.4.1 Introduction

The Flash capacity used by FM33LE0xxA is 32k*32, that is, 128KB; The array organization includes page (512B), sector (2KB), mat (128KB)

Main array contains a total of 256 sectors and supports page erasure, sector erasure and matrix erasure.

7.4.2 Special information sector description

In addition of the main array of Flash, there are several special sectors for users to use, the description is as follows:

Area	Description	Use
LDT1	User option data area	User option byte (OPTBYTES)
IF	Information area	4 pages total 2KB, for users to use

7.4.2.1 LDT1 page

LDT1 is the user configuration page that can be modified by user (only by SWD, that is, modified by the user through the programmer).

The bus address of LDT1 is 0x1FFF_FC00~0x1FFF_FDFF;

AHB addr	Bit[31:16]	Bit[15:0]	Description
0x1FFF_FC00	~OPTBYTES[15:0]	OPTBYTES[15:0]	User option byte low half word
0x1FFF_FC04	~OPTBYTES[31:16]	OPTBYTES[31:16]	User option byte high half word
0x1FFF_FC08	LOCK1		ACLOCK configuration word, control low 16 blocks

Table 7-2 Data content definition

OPTBYTES (User option bytes) are defined as follows:

Bitfield	Name	Functional description	Default value
31:24	BTSWPEN	Boot address swapping enable 0x55: Boot swap function allowed Others: Boot swap function forbidden	0xFF
23:20	IWDTSLP	Configure whether IWDT is allowed to stop counting in low-power mode 0xA: Allow to use stopping IWDT counting in Sleep/DeepSleep/RTCBKP mode Others: Forbidden to use stopping IWDT in any mode	0xF
19:16	DFLSEN	Data flashenable 0x5: Enable data flash, the highest 16KB address of main array is defined as data flash Others: Disable data flash	0xF
15:8	ACLKEN	Application code protection enable 0x33: Disable ACLOCK Others: Enable ACLOCK	0x33
7:0	DBRDPEN	Debug access protection enable 0xAA: No debug protection Others: Enable debug protection	0xAA

Table 7-3 User option byte definition

Note: OPTBYTES can be written by user software or SWD interface for a fresh device. But once ACLKEN or DBRDP is enabled, the user must erase all flash (matrix erase) with SWD before rewriting OPTBYTES.

LOCK configuration byte definition:

Bitfield	Name	Functional description	Default value
31:0	LOCK1	Block Lockword1，every 2it corresponds to one 8KB Block 11：Unprotected 01、10：Software read-write prohibited, only fetching allowed 00：Software read-write prohibited, only fetching allowed; SWD read-write prohibited LOCK1[1:0]corresponds to Block0(Flashminimum address 8KBspace) LOCK1[31:30]corresponds to Block15 (Flash address space 120~128KB), and so on.	0xFFFFFFFF

Table 7-4 Lock information definition

The address mapping for LOCK bits is shown in the following table:

Address	LOCK bits
0x0000_0000 ~ 0x0000_1FFF	LOCK1[1:0]
0x0000_2000 ~ 0x0000_3FFF	LOCK1[3:2]
0x0000_4000 ~ 0x0000_5FFF	LOCK1[5:4]
0x0000_6000 ~ 0x0000_7FFF	LOCK1[7:6]
0x0000_8000 ~ 0x0000_9FFF	LOCK1[9:8]
0x0000_A000 ~ 0x0000_BFFF	LOCK1[11:10]
0x0000_C000 ~ 0x0000_DFFF	LOCK1[13:12]
0x0000_E000 ~ 0x0000_FFFF	LOCK1[15:14]
0x0001_0000 ~ 0x0001_1FFF	LOCK1[17:16]
0x0001_2000 ~ 0x0001_3FFF	LOCK1[19:18]
0x0001_4000 ~ 0x0001_5FFF	LOCK1[21:20]
0x0001_6000 ~ 0x0001_7FFF	LOCK1[23:22]
0x0001_8000 ~ 0x0001_9FFF	LOCK1[25:24]
0x0001_A000 ~ 0x0001_BFFF	LOCK1[27:26]
0x0001_C000 ~ 0x0001_DFFF	LOCK1[29:28]
0x0001_E000 ~ 0x0001_FFFF	LOCK1[31:30]

Table 7-5 Lock bit and Flash address corresponding table

Note: In the manufacturer mode, ACLOCKEN and DBRDPEN do not work and will not protect Flash content.

7.4.2.2 Information3 page

Flash also contains four information pages, in which Information3 is used to control the BootSwap function. The Information3 page address is 0x0004_0600~0x0004_07FF.

With BOOTSWAPEN=0x55 in LDT1, BootSwap can be controlled by the INFO3 lowest address content. When the data is 0x5454_ABAB, the chip swaps the logical address of the lowest two 8KB spaces in Flash (note that ACLOCK is only applied to logical address, not the physical address), thus enabling risk-free upgrading of the boot code.

The schematic diagram of BootSwap is as follows:

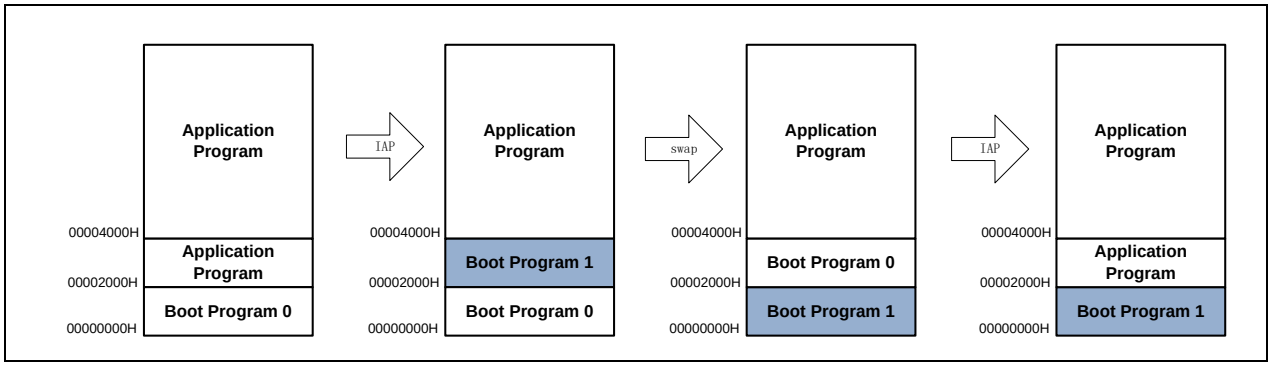


Figure 7-4 BootSwap schematic

After realizing the bootswap function, it is assuming that the boot code occupies a total of 8KB space of 0000~1FFF. When the system is being upgraded, the new boot code should be written into the address of 2000~3FFF first, and then enable BootSwap.

There are several possible conditions:

- Power off occurs when erasing 2000~3FFF address. Because the original boot code is still present, system will boot using original code after reset
- The chip successfully writes boot Program1, then enables BootSwap and perform a soft reset. System will boot using new boot code after reset
- The chip loses power when erasing boot Program0. Since boot Program1 has already been written, system will boot using new boot code after reset

The recommended upgrading procedure is as follows:

- Update the application program
- To upgrade boot code, write the new boot program into the second 8KB space
- Configure INFO3 to enable BootSwap
- Perform a soft reset and boot from the new Boot program
- Rewrite the second 8KB space to the new application

The remapping of the logical address to the Flash physical address is automatically done by chip, and both software and Debugger can only access to the logical address.

Note: *The BootSwap function of IF3 actually only uses the lowest word on this page, and the rest of the*

address space is open for reading and writing (no specific functions), and software or Debugger can write data at will.

7.4.2.3 Information1~2 page (Debugger only, lockable)

These two information pages are open to users. It can only be modified by SWD, and read-only to software.

The bus address is 0x0004_0200~0x0004_05FF, low address is IF1, high address is IF2, with 1KB in total.

The highest address bytes of IF2~1 are sector lock flags.

If SWD rewrites the highest address byte to 0x55, the current sector will be prohibited from programming after chip reset. SWD can erase the page and reprogram it.

Whether there is a lock flag or not, these sectors are readable by SWD and software.

7.4.2.4 Information page (OTP)

IF0 is an OTP page that can only be programmed once and cannot be erased or modified after programming. The bus address of IF0 is 0x0004_0000~0x0004_01FF, 512 bytes in total.

There is on restriction on reading IF0 pages.

7.4.3 Flash Program

7.4.3.1 Introduction

FM33LE0xxA supports the following Flash programming methods:

- In system programming (ISP): Chip programming via FMSH dedicated programmer or online simulation, using SWD interface
- In application programming (IAP): Bootloader code can perform self-programming, the user can define any communication interface to realize online upgrade

Flash must be erased before programming. Flash supports three erasing operations: matrix erase, sector erase and page erase.

7.4.3.2 Flash erase clock

RCHF clock is used to perform Flash erase/program, while the system clock can be any clock. The

supported RCHF frequencies of programming are 8M, 16M and 24M.

The Flash erase clock is independent of the CPU clock, and the two clock are handled as asynchronous clocks

7.4.3.3 Flash erase method

FM33LE0xxA supports Flash erase operations, as well as single and continuous programming.

Flash must perform Key verification before erase/program. The key register must be written with correct values, as well as in correct order, right before flash erase/perform operation. Otherwise, an error interrupt will be issued and flash erase/program will not be performed by hardware. When there is a Flash Key authentication error, Flash erase/program is prohibited until next system reset. Writing any value to the KEY register after a normal erase/program operation causes the state machine to return to its original write-protected state.

The state transition is as follows:

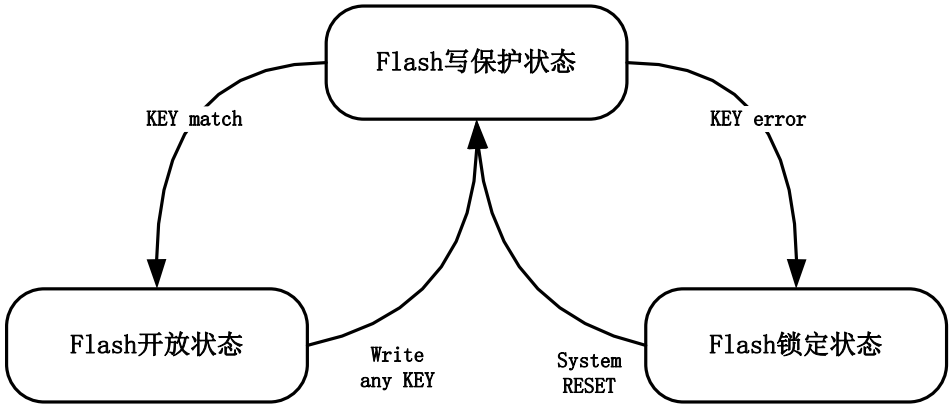


Figure 7-5 State transition schematic diagram

The software can check the current Key input status by querying FLSIF.KEYSTA. Please refer to the register description for details.

7.4.3.4 Matrix Erase

Matrix erase operation can only be initiated by SWD interface (the software prohibits matrix erase operation). The matrix erase operation only erases the main array, and does not erase the special information sector. SWD can initiate matrix erase in manufacturer or user mode, the operation process is as follows:

- Write 10 to ERTYPE register by SWD
- Clear PREQ register by SWD, and then set the EREQ register
- Write the Flash matrix erase Key: 0x9696_9696 and 0x7D7D_7D7D by SWD

- SWD writes erase request 0x1234_ABCD to any Flash address
- Chip starts the matrix erase to Flash and suspends any Master access to Flash
- Interrupt flag and matrix erase flag are set after the matrix erase is finished (The matrix erase flag means that the main array is all erased, and any programming to the main array will clear this flag)
- SWD can modify NVR0 if the matrix erase was performed, otherwise modifying NVR0 is forbidden and will trigger an error interrupt
- After confirming the end of the erasing, the software writes any value to FlashKEY register to restore write-protection

7.4.3.5 Sector Erase

Both SWD and application code can perform sector erase. The procedure is as follows:

- Write 10 to ERTYPE register
- Clear the PREQ register and set the EREQ register
- Write the Flash block erase Key: 0x9696_9696 and 0xEAEA_EAEA
- Write erase request 0x1234_ABCD to any address in the sector that needs to be erased
- Chip checks whether the target sector is locked by ACLOCK, starts erasing the target sector if there is no lock, and triggers an error flag if there is a lock
- After sector erasing is complete, set the interrupt flag
- The software writes any value to FlashKEY register to restore write-protection

7.4.3.6 Page Erase

Both SWD and application code can perform page erase. The procedure is as follows:

- Write 00 or 11 to ERTYPE register
- Clear the PREQ register and set the EREQ register
- Write the Flash block erase Key: 0x9696_9696 and 0xEAEA_EAEA
- Write erase request 0x1234_ABCD to any address in the page that needs to be erased
- Chip checks whether the target sector is locked by ACLOCK, starts erasing the target sector if there is no lock, and triggers an error flag if there is a lock
- After sector erasing is complete, set the interrupt flag
- The software writes any value to FlashKEY register to restore write-protection

7.4.3.7 Single programming

The single programming is initiated by software and write operation is performed directly to Flash through the bus. Byte/half-word/word programming can be executed for each operation. The procedure is as follows:

- Clear the EREQ register and set the PREQ register
- Clear the multiple word programming enable register
- Write the Flash programming Key: 0xA5A5_A5A5 and 0xF1F1_F1F1
- Write data to the Flash target address, an error flag will be set by hardware if the target address is locked by ACLOCK, and programming will be performed if there is no lock
- Byte writing is directly completed. Half-word writing is automatically executed by NVMIF twice, and word programming is executed by NVMIF four times
- The interrupt flag is set after the programming is completed
- The software writes any value to FlashKEY register to restore write protection

7.4.3.8 Multiple words programming

Multiple words programming can program half-sector (256 bytes) to Flash at one time over DMA Memory channels. During multi-word programming the DMA reads data from RAM, and Flash target address must be aligned to half-sector, which means lowest 6bits of Flash address must be 0. In this way, fixed length of data could be programmed into Flash continuously and efficiently.

During the continuous programming, Flash interface is fully occupied by DMA, so any access from CPU will be halted. The procedure of multiple words programming is as follows:

- Clear the EREQ register and set the PREQ register
- Set the multiple word**s** programming enable register (DMA mode enable)
- Write 256 bytes data to RAM
- Configure DMA memory channels, set the transfer direction, read address, and write address
- Enable DMA memory channels
- Write the Flash programming Key: 0xA5A5_A5A5 and 0xF1F1_F1F1
- Software triggers DMA memory channels, which will read RAM 64 times in a row and program Flash. Every time the chip receives a word, and it automatically completes the byte program four times.
- Chip will check whether the programmed sector is locked by ACLOCK, and if locked an error interrupt will be triggered and DMA will stop programming
- An interrupt is triggered when 256-byte programming is completed, then Flash interface is released

- The software writes any value to FlashKEY register to restore write protection.

Note: If Flash erase/program is initiated while CPU is executing from Flash, CPU fetching will be halted until erase/program cycle is finished. If the CPU is executing from RAM, Flash erase/program will not halt CPU execution. During Flash erase/program, if user still wants to respond to interrupts in time, it is recommended to remap vector table into RAM.

7.4.3.9 BootSwap

The Main purpose of BootSwap is to prevent the occurrence of unexpected interruption (power failure, abnormal reset, etc.) when the system is updating the boot code. If the original boot code has been erased at this time, it will lead to the failure of normal operation after the chip is reset.

BootSwap function is realized by programming the lowest address word of information page3.

BootSwap diagram is shown as follows:

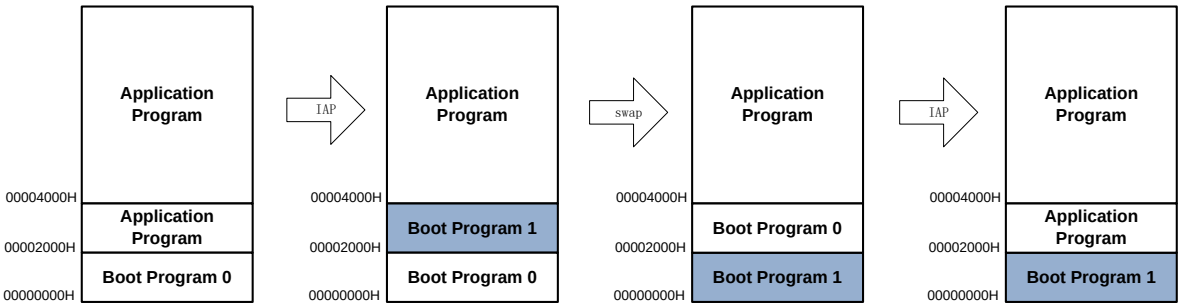


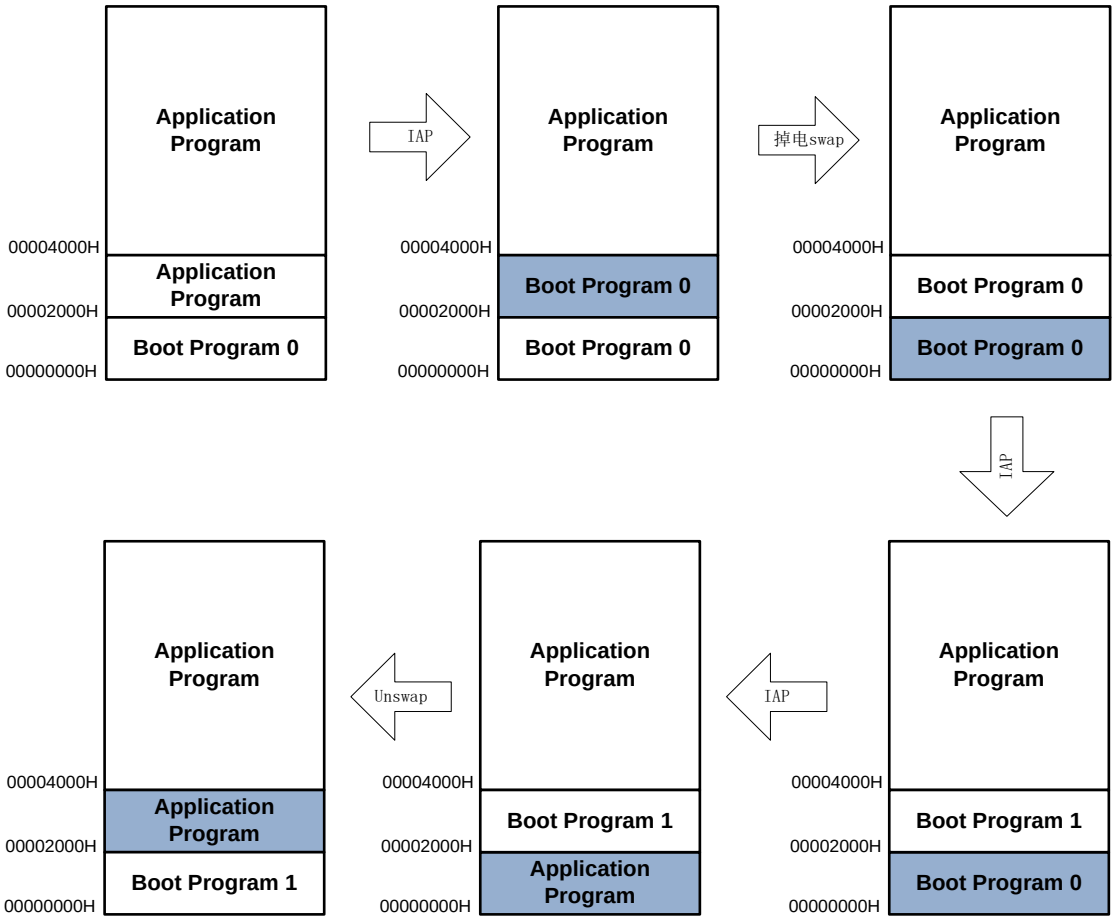
Figure 7-6 BootSwap

When the BootSwap function is enabled, it is assumed that the boot code occupies a total space of 8KB from 0000 to 1FFF. When the system is being upgraded, the new boot code should be written into the address of 0x2000~3FFF first, and then swap boot code area.

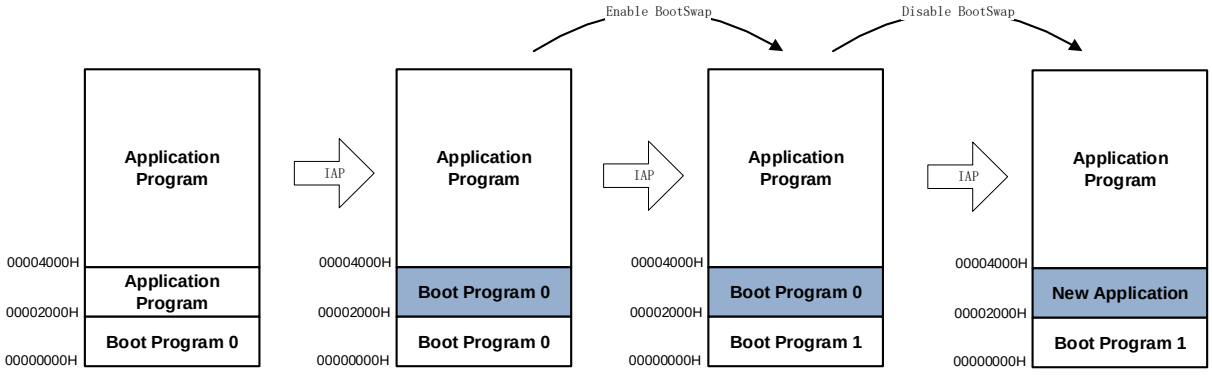
There are several possible conditions:

- Power off occurs when writing 2000~3FFF address. Because the original boot code is still present, system will boot using original code after reset
- The chip successfully writes Boot Program1, then enables BootSwap and perform a soft reset. System will boot using Boot Program 1 code after reset
- The chip loses power when erasing boot Program0. Since boot Program1 has already been written, system will boot using new boot code after reset

Another BootSwap application is shown in the figure below. In order to ensure reliable update of boot code, the 2nd 8KB physical space is used as a backup of the original Boot program. If an abnormal power loss occurs during programming, BootSwap will be triggered:



If there is no abnormal power loss during the Boot program update, soft reset can be skipped, and no swap is needed. Instead, BootSwap can be enabled before the original Boot program is modified, and BootSwap can be disabled after successful update:



The recommended boot code updating procedure is as follows:

- Update application program
- Write the new boot program to a second 8KB space

- Configure the information block, enable BootSwap
- Perform a soft reset, execute the new boot program after reboot
- Rewrite the second 8KB space to the new application

The remapping of the logical address to the Flash physical address is completed by the NVMIF module, and both the program and the DEBUG access it with the logical address.

Register flag (FLSIF.BTSF) is used to indicate whether the current Boot area is physically located in 1st 8KB or 2nd 8KB. Application code can use this flag to determine the current boot status.

7.4.4 Data Flash

After the user configures OPTBYTES to enable DFLSEN, FM33LE0xxA will open 16KB data flash to users for data storage. After the data flash is enabled, the flash capacity of different models of products are divided as shown in the following table:

Model	Data flash size	Data flash address	Program flash size
FM33LE02x	16KB	0x0003_C000~0x0003_FFFF	112KB
FM33LE01x	16KB	0x0001_C000~0x0001_FFFF	48KB

When the data flash is enabled, the corresponding program flash space will be reduced by 16KB. Data flash is always located at the highest 16KB of the flash logical address space.

Data flash and program flash have no difference in access rights, and are also controlled by DBRDP and ACLOCK. But when the chip performs a matrix erase operation, the data flash will not be erased. When the data flash is not enabled, all data in the main array will be erased when the chip executes a matrix erase.

Note: *With data flash enabled, the chip's matrix erasing time is significantly increased (because of the sector erase). For the 128KB capacity model, the time is increased from 8ms to 112ms.*

7.4.5 Flash memory protection

Flash memory protection can be used to protect user code, user data and user configuration information in Flash from being read and tampered with by an unauthorized third party.

Flash Protection includes two types: DBRDP-DeBug ReaD Protection and ACLOCK-Application Code Block Locking. Flash protection is controlled via OPTBYTES in LDT1.

7.4.5.1 Debug interface protection (DBRDP)

The primary purpose of DBRDP is to prevent unauthorized access to the Flash content through the Debug interface.

DBRDP is enabled or disabled by the DBRDPEN configuration word in LDT1 sector (0xAA means DBRDP is disabled, which is default state of a fresh device). When DBRDP is enabled, the Flash main array cannot be read or erased through the SWD interface, and the RAM cannot be accessed through the SWD interface.

Ways to exit DBRDP: After the matrix erase of the flash is completed through SWD, SWD can rewrite OPTBYTES to disable DBRDP at will, and then reset the chip. After the reset is completed, the chip will be in a non-debug protection state.

7.4.5.2 Application code lock (ACLOCK)

The main purpose of ACLOCK is to prevent hacking code reading or tempering to the application code in Flash from hacking code. With the ACLOCK function, you can set some part of Flash as fetch-only, any read-as-data or modifying are prohibited.

ACLOCK works in the granularity of 8KB. The whole Flash contains 32 Blocks with 2bit LOCK information for each Block. The default LOCK word is 0xFFFF_FFFF for a fresh device. The lock bits of power on load are 11, which is unprotected state. When the corresponding LOCK bits are set to 01 or 10, this Block can only be fetched by CPU. When the corresponding LOCK bits are 00, both CPU and SWD are prohibited to read or modify the Block. ACLOCK function is disabled by default. The user needs to enable ACLOCK through the programmer, and the user code should conform to the ACLOCK configuration when compiling (for example, literal pool cannot be compiled to the locked Block).

Function of ACLOCK:

- No protection: All blocks allow CPU to fetch, read, and modify. No restriction on SWD access.
- Read-write protection: Specified blocks allow CPU to fetch only, read & modify by CPU and DMA is prohibited. No restriction on SWD access.
- Software and SWD protection: Specified blocks allow CPU to fetch only, read & modify by CPU, DMA and SWD is prohibited.

The relationship between LOCK bit and Block access permissions is shown in the following table:

LOCK bit	CPU read	CPU fetch	SWDread and erase/program
11	√	√	√
01/10	×	√	√
00	×	√	×

Table 7-6 LOCK bit permission definition

ACLOCK information is loaded into registers after system reset. These registers can also be set by software,

but cannot be cleared by software (it is only possible for software to escalate the protection level).

LOCK register contents are invalid when ACLOCK is not enabled.

Note: *It is forbidden to use ACLOCK to disable the read permission of 1st block. Since the MSP pointer must be read from address 0 after the CPU is reset, ACLOCK will cause the CPU to fail to start normally.*

Exit ACLOCK: Full-space matrix erase must be performed by SWD. After matrix erasing, SWD can modify OPTBYTES to disable ACLOCK, and then reset the chip. After reset, ACLOCK is unactivated.

7.4.5.3 Flash access authorization description

Flash space access authorization allocation

Flash area	DBRDP	LOCK bits (per Block)[3]	Last byte in page	SWD	Application
Main array	ON	00	x	-	Block fetch only
		01/10	x	-	Block fetch only
		11	x	-	R/E/W/F
	OFF	00	x	Block cannot be accessed	Block fetch only
		01/10	x	R/E/W	Block fetch only
		11	x	R/E/W	R/E/W/F
LDT1	ON	x	x	R[2]	R
	OFF	x	x	R/E/W	R
IF3	x	x	x	R/E/W	R/E/W
IF2,1	x	x	55	R/E	R
			others	R/E/W	R
IF0	x	x	x		

Table 7-7 Flash access authorization table

Note:

[1] R: Read, E: Erase, W: Write, F: Fetch

[2] LDT1 erase can be performed after matrix erase

[3] ACLOCKEN is assumed to be valid in the above description. If ACLOCKEN is disabled, LOCK bits have no effect.

7.5 Register

Offset	Name	Symbol
FLS(base address: 0x40001000)		
0x00000000	Flash Read Control Register	FLS_RDCR
0x00000004	Flash Prefetch Control Register	FLS_PFCR
0x00000008	Flash Option Bytes Register	FLS_OPTBR
0x0000000C	Flash Application Code Lock Register1	FLS_ACLOCK1
0x00000010	-	-
0x00000014	Flash Erase/Program Control Register	FLS_EPCR
0x00000018	Flash Key Register	FLS_KEY
0x0000001C	Flash Interrupt Enable Register	FLS_IER
0x00000020	Flash Interrupt Status Register	FLS_ISR

7.5.1 Flash Read Control Register (FLS_RDCR)

NAME	FLS_RDCR							
offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						WAIT	
access	U-0						R/W-00	

Bit	Name	Description
31:2	-	RFU: **Reserved，read as 0**
1:0	WAIT	Flash Wait Cycles Config 00/11: 0 wait cycle 01: 1 wait cycle 10: 2 wait cycles When the system frequency is less than or equal to 24MHz, there is no need to enable wait cycle; If the system frequency is greater than 24MHz and less than 48MHz, enable 1 wait; if the system frequency is greater than 48MHz, enable 2 waits.

7.5.2 Flash Prefetch Control Register (FLS_PFCR)

NAME	FLS_PFCR							
offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							PRFTEN
access	U-0							R/W-0

Bit	Name	Description
31:1	-	RFU：**Reserved，read as 0**
0	PRFTEN	Instruction Prefetch Enable. In the case of WAIT==00, writing 1 is invalid 1: Enable Prefetch 0: Disable Prefetch

7.5.3 Flash Option Byte Register (FLS_OPTBR)

NAME	FLS_OPTBR							
offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	IWDTSLP	-						
access	R-0	U-0						
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-					IF2LOCK	IF1LOCK	-
access	U-0					R-0	R-0	U-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-					DFLSEN	BTSEN	
access	U-0					R-0	R-01	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				ACLOCKEN		DBRDPEN	
access	U-0				R-01		R-01	

Bit	Name	Description
31	IWDTSLP	IWDT Sleep Enable 1: Allows the application to suspend IWDT counting in Sleep mode 0: Prohibit the application from suspendingIWDT counting in Sleep mode
30:19	-	RFU: **reserved, read as 0**
18	IF2LOCK	Information2 Lock Flag (IF2 Lock Enable) 0: Unlock 1: Locked, software cannot modify this page
17	IF1LOCK	Information1 Lock Flag (IF1 Lock Enable) 0: Unlock 1: Locked, software cannot modify this page
16:11	-	RFU: **Reserved, read as 0**
10	DFLSEN	DataFlash Enable 0: Have no data flash 1: Have data flash
9:8	BTSEN	BootSwap Enable 00/01/11: Prohibit BootSwap function 10: Allow BootSwap function
7:4	-	RFU: **Reserved, read as 0**
3:2	ACLOCKEN	AppCode Lock Enable 00/01/11: ACLOCK disable 10: ACLOCK enable
1:0	DBRDPEN	Debug Read Protection Enable 00/01/11: DBRDP disable 10: DBRDP enable

7.5.4 ACLOCK register1 (FLS_ACLOCK1)

NAME	FLS_ACLOCK1							
offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	LOCK1[31:24]							
access	R/W-1111 1111							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	LOCK1[23:16]							
access	R/W-1111 1111							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	LOCK1[15:8]							
access	R/W-1111 1111							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	LOCK1[7:0]							

access	R/W-1111 1111

Bit	Name	Description
31:0	LOCK1	ACLOCK low 32bit, which is apply to control the Block15~Block0 respectively. Each Block is 8KB size, and each Block uses 2 bits for access control (Lock bits). 11: The current Block allows SWD and software to read and write 01/10: The current Block allows SWD read and write, prohibits the software from reading and writing. Software can fetch. 00: The current Block prohibits SWD & software from reading and writing. And the software can only fetch Software can only write 0 to these bits, and write 1 is prohibited

7.5.5 Flash Erase/Program Control Register (FLS_EPCR)

NAME	FLS_EPCR							
offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						ERTYPE	
access	U-0						R/W-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						PREQ	EREQ
access	U-0						R/W-0	R/W-0

Bit	Name	Description
31:10	-	RFU: **Reserved, read as 0**
9:8	ERTYPE	Flash Erase Type 00/11: Page Erase 01: Sector Erase 10: Chip Erase (SWD only)
7:2	-	RFU: **Reserved, read 0**
1	PREQ	Program Request Software reset. Automatically cleared by hardware after programming complete. Software cannot clear.
0	EREQ	Erase Request

Bit	Name	Description
		Software reset. Automatically cleared by hardware after programming complete. Software cannot clear.

7.5.6 Flash Key Register (FLS_KEY)

NAME	FLS_KEY							
offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	KEY[31:24]							
access	W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	KEY[23:16]							
access	W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	KEY[15:8]							
access	W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	KEY[7:0]							
access	W-0000 0000							

Bit	Name	Description
31:0	KEY	Flash Erasure Key input Register Software or SWD must correctly write a valid Key sequence into this address to initiate erase/program.

7.5.7 Flash Interrupt Enable Register (FLS_IER)

NAME	FLS_IER							
offset	0x0000001C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				OTPIE	AUTHIE	KEYIE	CKIE
access	U-0				R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						PRDIE	ERDIE
access	U-0						R/W-0	R/W-0

Bit	Name	Description
31:12	-	RFU: **Reserved, read as 0**
11	OTPIE	OTP program error Interrupt Enable, 1 enable
10	AUTHIE	Flash Authentication Error Interrupt Enable, 1 enable
9	KEYIE	Flash Key Error Interrupt Enable, 1 enable
8	CKIE	Erase/Program Clock Error Interrupt Enable, 1 enable
7:2	-	RFU: **Reserved, read as 0**
1	PRDIE	Program Done Interrupt Enable, 1 enable
0	ERDIE	Erase Done Interrupt Enable, 1 enable

7.5.8 Flash Interrupt Status Register (FLS_ISR)

NAME	FLS_ISR							
offset	0x00000020							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-				KEYSTA			BTSF
access	U-0				R-000			R-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				OTPERR	AUTHERR	KEYERR	CKERR
access	U-0				R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						PRD	ERD
access	U-0						R/W-0	R/W-0

Bit	Name	Function Description
31:20	-	RFU: **Reserve, read as 0**
19:17	KEYSTA	Flash Key Status 000: Flash write-protect status, noKEY entered 001: Matrix erase unlocked state 010: Page erase unlocked state 011: Programming unlocked state 100: Locked state caused by KEY error. Reset is required to unlock. 101/110/111：RFU
16	BTSF	BootSwap Flag Register 0: The boot area is Flash physical address 0000H~1FFFH 1: The boot area is Flash physical address 2000H~3FFFH
15:12	-	RFU: **Reserved, read as 0**

Bit	Name	Function Description
11	OTPERR	OTP Program Error Flag. Hardware set. Write 1 to clear. 1: Try to program the OTP bytes that have been programmed 0: No OTP programming error
10	AUTHERR	Flash Authentication Error Flag. Set when reading or erasing a LOCK block, write 1 to clear. 1: Flash access error 0: Flash have no access error
9	KEYERR	Flash Key Error Flag. Hardware set. Write 1 to clear
8	CKERR	Erase/Program Clock Error Flag, CKERR interrupts are triggered if RCHF is not enabled while writing Flash with NVMIF, write 1 to clear
7:2	-	RFU: Reserved, read as 0
1	PRD	Program Done Flag, hardware set, write 1 to clear
0	ERD	Erase Done Flag, hardware set, write 1 to clear

8 Reset Management Unit (RMU)

8.1 Introduction

Features of Reset circuit：

- Support multiple reset sources, such as POR, PDR, WDG reset, software reset, pin reset, etc
- BOR monitoring main power supply
- BOR power-on reset typical release voltage is 1.8V
- BOR power-down reset with programmable reset voltage: 1.75/1.7/1.65/1.6V, can be disabled
- by software
- Low power PDR
- Filtering and delay function to ensure robust system reset

During system reset phase, all registers are restored to their default values (except RTC internal registers);

After exiting system reset, the MCU uses internal high-speed RC oscillator (RCHF, default frequency of 8MHz) as the system clock by default.

8.2 Block diagram

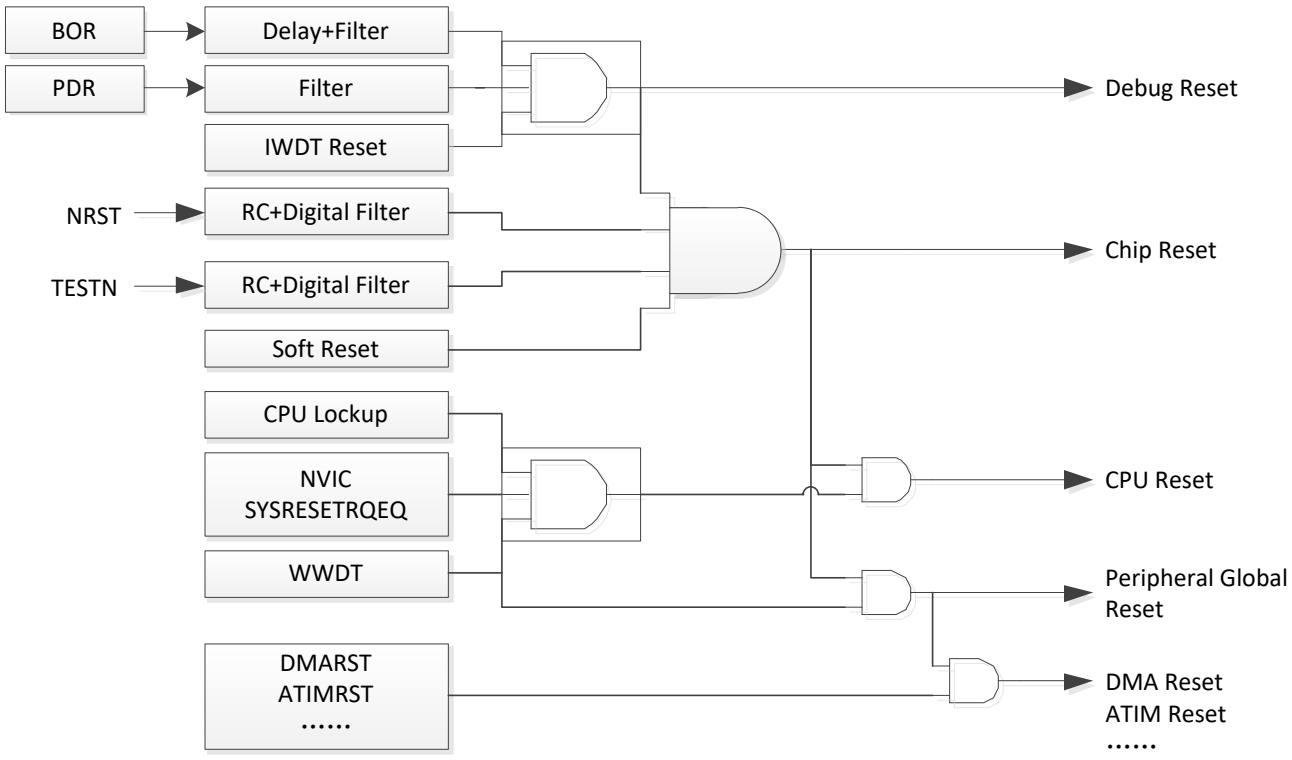


Figure 8-1 Reset block diagram

8.3 Power-on reset and power-down reset (POR&PDR)

The power-on and power-down reset circuit monitors the VDD supply, which consists of BOR and ultra-low-power PDR. The BOR provides accurate threshold voltage with a relative high current consumption. When accurate power-down reset is not mandatory, it is recommended to disable BOR by software and keep PDR enabled for power-down reset.

The power-on reset signal is effective during VDD power on. When VDD voltage exceeds V_{BOR}, the power-on reset is released. When VDD falls to V_{PDR}, power-down reset is activated. Filtering and delay function are implemented to avoid reset bounce when there are glitches on VDD supply.

The power-on reset voltage threshold of V_{BOR} is fixed at 1.8V, while the power-down reset voltage threshold of BOR and PDR can be programmed by software.

The power-on reset voltage threshold of VBOR is fixed at 1.8V, while the power-down reset voltage threshold of BOR and PDR can be programmed by software.

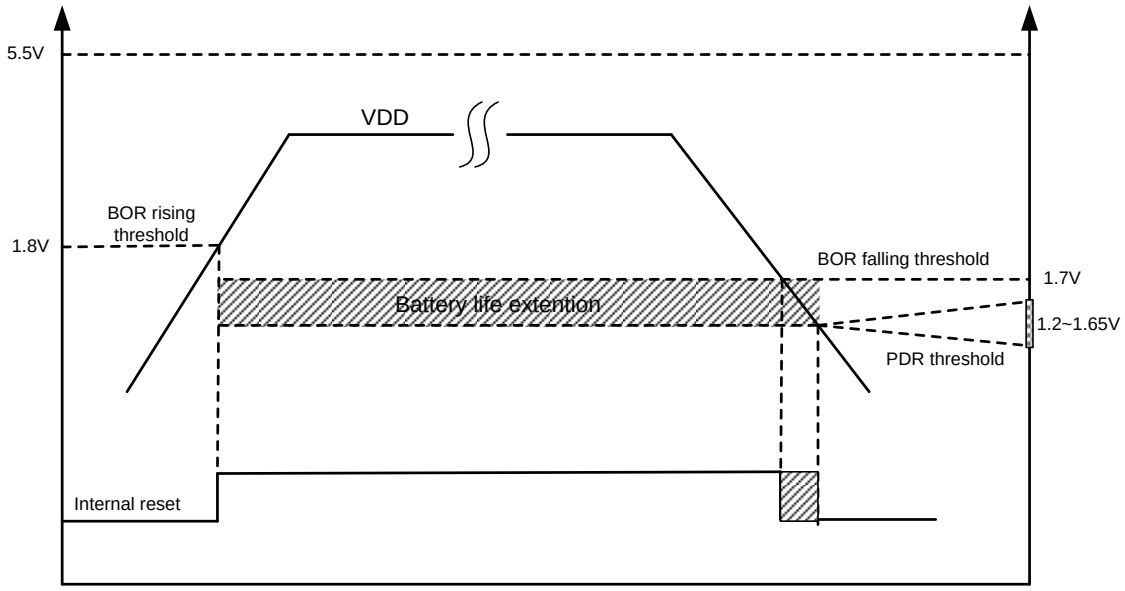


Figure 8-2 Power-on/power-down reset diagram

***Note: The** minimum operational voltage of CPU is 1.65V. If BOR is turned off, VDD has a gray-area between 1.65V and PDR threshold, and he rest of this period cannot ensure the normal operation of the CPU. Then the chip should enter the sleep mode to prevent abnormal program operation. Therefore, when BOR is closed in the application, it is recommended to use SVD to monitor VDD. When 1.8V low-voltage alarm occurs, it is recommended that the program actively enter sleep.*

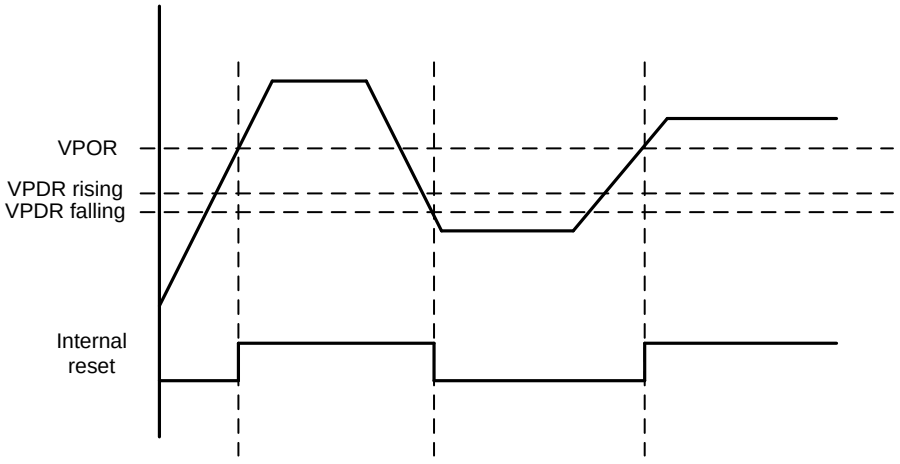


Figure 8-3 Power-on/power-down reset sequential diagram

8.4 Software reset

Software reset is initiated by the CPU writing register, and the operation mode is to write 0x5C5C_ABAB

to the SOFTEST register.

8.5 NRST pin reset

NRST is a special reset pin for the chip. After the NRST remains low for more than 60~90us, the chip will enter the system reset, but the debug logic will not be reset. If the chip is in low-power mode, NRST is effective, which will also reset the chip to exit low-power mode.

8.6 Register

offset	NAME	Symbol
RMU(Base address: 0x4001A800**)**		
0x00000000	PDR Control Register	RMU_PDRCR
0x00000004	BOR Control Register	RMU_BORCR

8.6.1 PDR control register (RMU_PDRCR)

NAME	RMU_PDRCR							
Name	0x00000000							
Offset	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
Bit	-							
Name	U-0							
Access	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
Bit	-							
Name	U-0							
Access	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Bit	-							
Name	U-0							
Access	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Bit	-					CFG		EN
Name	U-0					R/W-11		R/W-1

Bit	Name	Description
31:3	**-**	RFU：Reserved, read as 0
2:1	**CFG**	Ultra-low-power PDR threshold configuration, 00 or 11 gear is recommended 00：1.5V 01：1.3V 10：1.35V 11：1.4V（Default）
0	**EN**	Ultra-low-power PDR enable 0：disable 1：enable

8.6.2 BOR control register (RMU_BORCR)

Name	RMU_BORCR							
Offset	0x00000004							
Bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
Name	-							
Access	U-0							
Bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
Name	-							
Access	U-0							
Bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Name	-							
Access	U-0							
Bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Name	-				CFG		ENB	-
Access	U-0				R/W-01		R/W-0	U-0

Bit	Name	Description
31:4	**-**	RFU：Reserved, read as 0
3:2	**CFG**	BOR power-down reset voltage configuration 00：1.7V 01：1.6V（default） 10：1.65V 11：1.75V
1	**ENB**	BOR power-on reset enable 0：Enable BOR (power-on reset) 1：Disable BOR (power-down reset)
0	**-**	RFU：Reserved, read as 0

9 Independent Watchdog (IWDT)

9.1 IWDT introduction

The independent watchdog is used to monitor system operation. If the CPU runs abnormally and cannot feed watchdog at regular intervals. The watchdog generates a global reset signal after overflow to restart the system for avoiding system locking. The independent watchdog is activated by software after the chip is power on, which is cannot be disable until the chip reset.

For debugging purposes, the IWDT is deactivated under the following conditions.

- When the chip is in debug mode, the software can suspend the IWDT during debug by configuring the DBG_CR register
- When IWDTSLP in OPTBYTES is valid, the software can suspend IWDT counting in sleep mode

The IWDT core is a 12bit up counter that increments from 0 after reset and triggers an IWDT reset. The IWDT reset is a global reset and has the same effect as power-up and power-down reset.

The IWDT operates with LSCLK. Thanks to LFDET circuit, LSCLK will keep working even when XTLF fails. A divided by 128 prescaler is implemented before 12bit up counter.

IWDT supports programmable window function, the software can only clear the dog in the allowed window, and the clear dog outside the window will trigger the IWDT reset.

9.2 IWDT block diagram

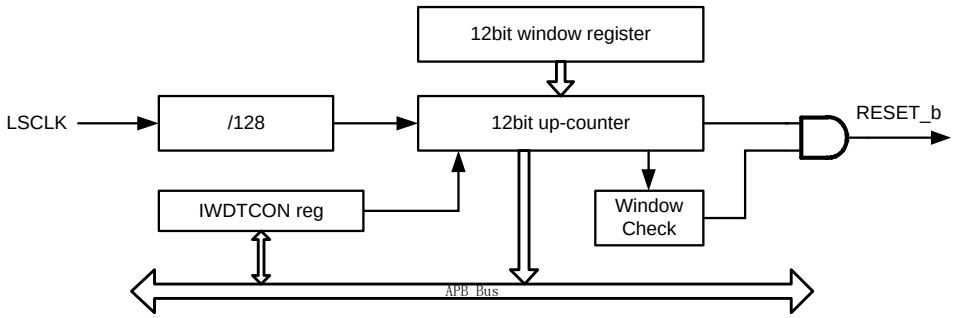


Figure 9-1 IWDT block diagram

9.3 IWDT function description

The watchdog should use a shorter overflow period when the CPU is running normally, while in low power modes such as SLEEP/DEEPSLEEP, the watchdog should use a longer overflow period in order to keep the chip in low power mode for as long as possible.

To fulfil different application requirements, software can modify the IWDT's overflow period configuration in real time. To avoid unpredictable consequences of improper operation, software should follow the following procedure when updating the overflow period configuration.

- Ensure that the watchdog is running
- Feed watchdog
- Rewrite the IWDT_CR register to select the appropriate overflow period
- Read IWDT_CR and make sure it is written correctly
- The overflow period is updated and the CPU is running normally

IWDT uses LSCLK to work. The internal prescaler is 128. The overflow length of the counter after the frequency division can be configured from 1~4096 (8 available gears in total). The overflow period can be calculated as follows:

$tWWDT = T_{LSCLK} * 128 * IWDTOVP$

LSCLK Frequency	Overflow Length Configuration	Overflow period（ms）
32768Hz	32	125
	64	250
	128	500
	256	1000
	512	2000
	1024	4000
	2048	8000
	4096	16000

Table 9-1 IWDT overflow periodic table

9.4 IWDT window function

IWDT supports programmable clear dog window function. The IWDT_WIN register is used to define the allowed dog-clearing window. Only when the counter count value is greater than or equal to the value of IWDT_WIN, the dog-clearing operation is legal. Clearing the dog outside the window will directly initiate an IWDT reset.

After the chip is reset, IWDT_WIN is all 0, which means that the software is allowed to clear the dog at any position by default.

The software can modify the IWDT_WIN register in real time while the IWDT is running. When the software clears the dog, it must read and confirm whether the current count value is within the allowed range of dog clearing.

When the IWDT count value enters the clear dog window, IWDT will trigger an interrupt flag register to notify the software that the current count value has entered the clear dog window.

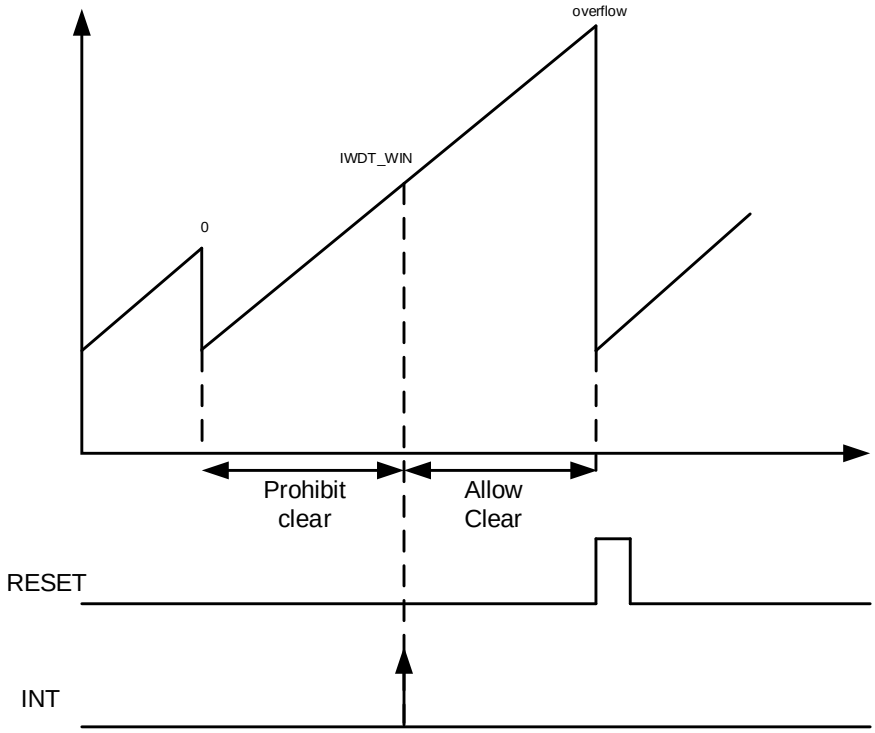


Figure 9-2 IWDT window diagram

9.5 IWDT freeze

The user can configure through OPTBYTES whether to allow IWDT to freeze counting in sleep mode. OPTBYTES needs to use Fudan micro special programmer configuration

When the IWDTSLP in OPTBYTES is valid and the software sets the IWDT_FREEZE register, when the chip enters Sleep/DeepSleep mode, the IWDT count value is automatically frozen (note that IWDT is not turned off, but the count value keeps the current value and no longer increments).

9.6 Register

Offset	Name	Symbol
IWDT(Base address: 0x40011400**)**		
0x00000000	IWDT Service Register	IWDT_SERV
0x00000004	IWDT Config Register	IWDT_CR
0x00000008	IWDT Counter Register	IWDT_CNT
0x0000000C	IWDT Window Register	IWDT_WIN
0x00000010	IWDT Interrupt Enable Register	IWDT_IER
0x00000014	IWDT Interrupt Status Register	IWDT_ISR

9.6.1 IWDT serve (IWDT_SERV)

Name	IWDT_SERV							
Offset	0x00000000							
Bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
Name	SERV[31:24]							
Access	W-0000 0000							
Bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
Name	SERV[23:16]							
Access	W-0000 0000							
Bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Name	SERV[15:8]							
Access	W-0000 0000							
Bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Name	SERV[7:0]							
Access	W-0000 0000							

Bit	Name	Description
31:0	**SERV**	IWDT is turned off by default after power-on reset. Software writes 0x1234_5A5A to this register and then starts IWDT. After that, IWDT cannot be turned off until the next chip reset. After IWDT is started, the software will clear the dog when writing 0x1234_5A5A to this address (IWDT Service Register,write only)

9.6.2 IWDT config register (IWDT_CR)

Name	IWDT_CR							
Offset	0x00000004							
Bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
Name	-							

Access	U-0							
Bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
Name	-							
Access	U-0							
Bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Name	-				FREEZE	-		
Access	U-0				R/W-0	U-0		
Bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Name	-					OVP		
Access	U-0					R/W-001		

Bit	Name	Description
31:12	-	RFU: **Reserved, read as 0**
11	FREEZE	IWDT sleep freezes, only works when the IWDTSLP configuration in OPTBYTES is valid (Freeze in Sleep Enable) 1: Sleep/DeepSleep mode, freeze IWDT count 0: Sleep/DeepSleep mode, keep IWDT running
10:3	-	RFU: **Reserved, read as 0**
2:0	OVP	Configure IWDT overflow period 000: 125ms 001: 250ms 010: 500ms 011: 1s 100: 2s 101: 4s 110: 8s 111: 16s

9.6.3 IWDT counter register (IWDT_CNT)

Name	IWDT_CNT							
Offset	0x00000008							
Bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
Name	-							
Access	U-0							
Bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
Name	-							
Access	U-0							
Bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Name	-				CNT[11:8]			
Access	U-0				R-0000			
Bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

Name	CNT[7:0]
Access	R-0000 0000

Bit	Name	Description
31:12	-	RFU: **Reserved, read as 0**
11:0	WIN	IWDT Window register

9.6.4 IWDT interrupt enable register (IWDT_IER)

Name	IWDT_IER							
Offset	0x00000010							
Bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
Name	-							
Access	U-0							
Bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
Name	-							
Access	U-0							
Bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Name	-							
Access	U-0							
Bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Name	-							IE
Access	U-0							R/W-0

Bit	Name	Description
31:1	-	RFU: **Reserved, read as 0**
0	IE	IWDT interrupt enable 0: Disabled interrupt 1: Enable interrupt

9.6.5 IWDT interrupt Flag register (IWDT_ISR)

Name	IWDT_ISR							
Offset	0x00000014							
Bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
Name	-							
Access	U-0							
Bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
Name	-							
Access	U-0							
Bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Name	-							
Access	U-0							

Bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Name	-							WINF
Access	U-0							R/W-0

Bit	Name	Description
31:1	-	RFU: **Reserved, read as 0**
0	WINF	IWDT enters window interrupt Flag, write 1 to clear 0: No interruption happen 1: The counting value enters the IWDT cleaning window

10 Window Watchdog (WWDT)

10.1 Introduction

The window watchdog is synchronous to CPU, which will reset the whole chip when program fails to feed WWDT in time.

WWDT is disabled by default after the chip is powered on. After the software starts WWDT, it cannot be disabled again until the next reset. WWDT is stopped under sleep mode.

WWDT uses APBCLK with an internal prescaler circuit to guarantee synchronous operation with CPU.

WWDT generates a system reset in the following conditions:

- Counter overflow
- Write a value other than 0xAC to the WWDT reset register (can be used to trigger a soft reset)
- Write 0xAC to the WWDT reset register during the window closed period

A warning interrupt is triggered when the counter reaches 75% of the overflow period.

10.2 WWDT block diagram

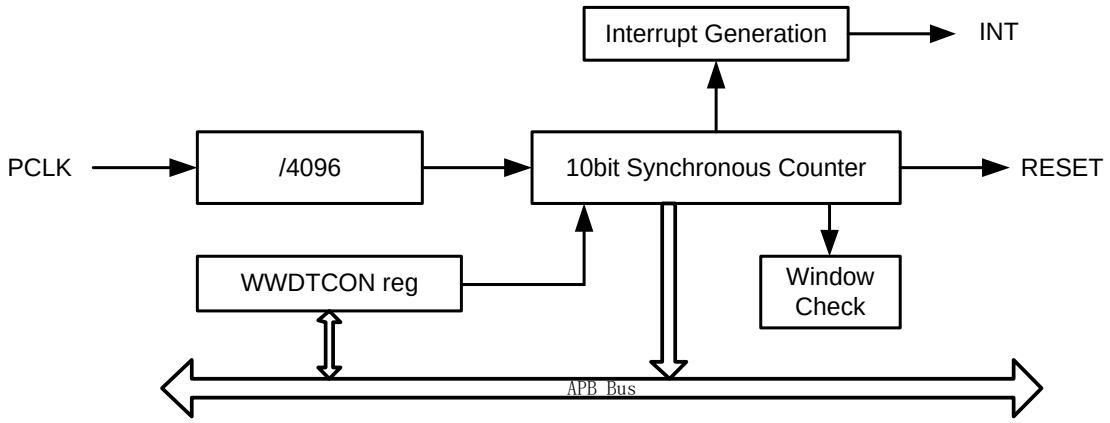


Figure 10-1 WWDT block diagram

10.3 WWDT functional description

After a chip reset, WWDT is disabled by default. WWDT is started by writing 0x5A to the WWDT_CR register. If 0xAC is written to WWDT_CR after WWDT is started, it will clear the counter. WWDT cannot be disabled once enabled until the next reset.

WWDT operates on APBCLK with an internal prescaler of 4096, the overflow period of the counter can be configured from 1 to 1024 (8 available steps in total). The overflow time period is calculated as follows:

$tWWDT = T_{APBCLK} * 4096 * N_{CFG}$

The following table shows calculation examples.

APBCLK freq	Overflow period configuration	Overflow time (ms)
48MHz	1	0.085
	4	0.341
	16	1.365
	64	5.461
	128	10.922
	256	21.845
	512	43.69
	1024	87.38
32MHz	1	0.128
	4	0.512
	16	2.048
	64	8.192
	128	16.384
	256	32.768
	512	64.536
	1024	131.072
16MHz	1	0.256
	4	1.024
	16	4.096
	64	16.384
	128	32.768
	256	65.536
	512	129.072
	1024	262.144
8MHz	1	0.512
	4	2.048
	16	8.192
	64	32.768
	128	65.536
	256	131.072
	512	258.144

APBCLK freq	Overflow period configuration	Overflow time (ms)
	1024	524.288

Table 10-1 WWDT overflow periodic table

The counter must be cleared in a limited window (2nd half of the period). Otherwise, a reset is generated. Software should read counter value before feeding watchdog.

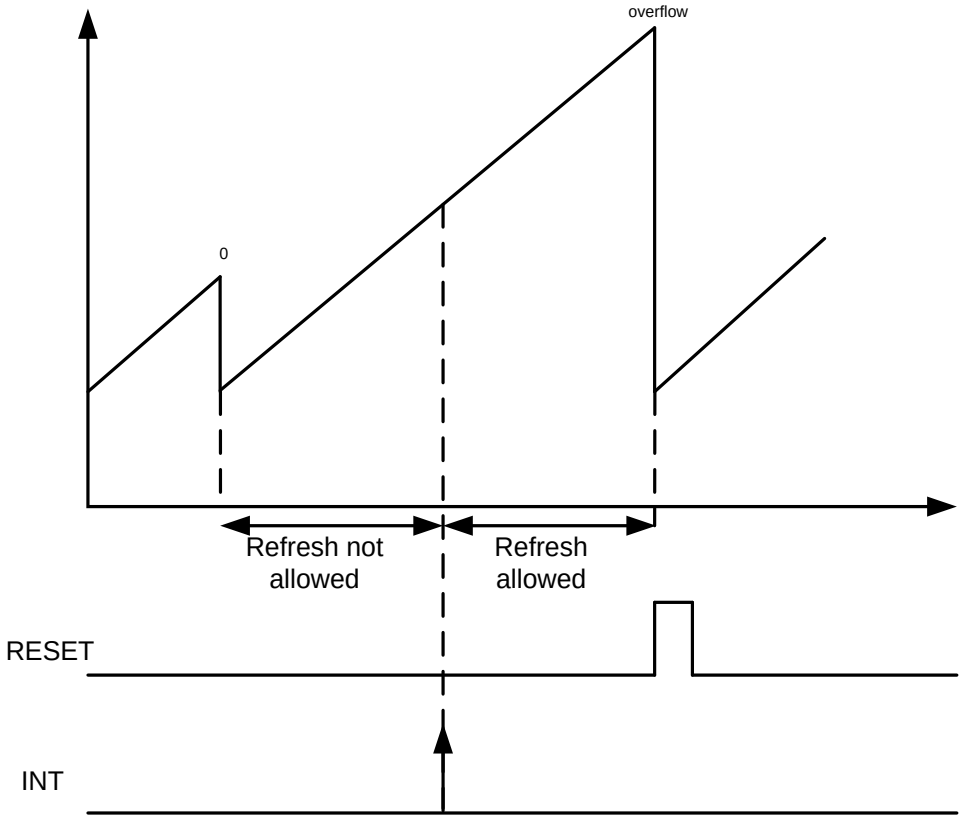


Figure 10-2 Window watchdog timing diagram

10.4 Register

Offset	Name	Symbol
WWDT(Base address: 0x40011800)		
0x00000000	WWDT Control Register	WWDT_CR
0x00000004	WWDT Config Register	WWDT_CFGR
0x00000008	WWDT Counter Register	WWDT_CNT
0x0000000C	WWDT Interrupt Enable Register	WWDT_IER
0x00000010	WWDT Interrupt Status Register	WWDT_ISR
0x00000014	WWDT Prescaler Register	WWDT_PSC

10.4.1 WWDT Control Register (WWDT_CR)

NAME	WWDT_CR							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CON							
access	W-0000 0000							

Bit	Name	Description
31:8	-	Reserved, read as 0
7:0	CON	Write 0x5A to enable WWDT Write 0xAC to clear WWDT

10.4.2 WWDT Config Register (WWDT_CFGR)

NAME	WWDT_CFGR							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							

bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					CFG		
access	U-0					R/W-011		

Bit	Name	Description
31:3	-	Reserved, read as 0
2:0	CFG	WWDT overflow period. The default overflow period is approximately 32ms as the system clock defaults to 8Mhz after power up 000:T_{PCLK} * 4096 * 1 001:T_{PCLK} * 4096 * 4 010:T_{PCLK} * 4096 * 16 011:T_{PCLK} * 4096 * 64 100:T_{PCLK} * 4096 * 128 101:T_{PCLK} * 4096 * 256 110:T_{PCLK} * 4096 * 512 111:T_{PCLK} * 4096 * 1024

10.4.3 WWDT Counter Register (WWDT_CNT)

NAME	WWDT_CNT							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						CNT[9:8]	
access	U-0						R-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CNT[7:0]							
access	R-0000 0000							

Bit	Name	Description
31:10	-	Reserved, read as 0
9:0	CNT	WWDT Counter value,read only

10.4.4 WWDT Interrupt Enable Register (WWDT_IER)

NAME	WWDT_IER							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16

name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							IE
access	U-0							R/W-0

Bit	Name	Description
31:1	-	Reserved, read as 0
0	IE	WWDT Interrupt Enable 0: Disable Interrupt 1: Enable Interrupt

10.4.5 WWDT Interrupt Status Register (WWDT_ISR)

NAME	WWDT_ISR							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							NOVF
access	U-0							R/W-0

Bit	Name	Description
31:1	-	Reserved, read as 0
0	NOVF	WWDT count to 75%interrupt flag, cleared by software writing 1 0: No interruption 1: Interrupt Flag

10.4.6 WWDT Prescaler Register (WWDT_PSC)

NAME	WWDT_PSC							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							

NAME	WWDT_PSC							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				DIV_CNT[11:8]			
access	U-0				R-0000			
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DIV_CNT[7:0]							
access	R-0000 0000							

Bit	Name	Description
31:12	-	Reserved, read as 0
11:0	DIV_CNT	WWDT prescaler Divider Counter, read only

11 Clock Manage Unit (CMU)

11.1 Introduction

The chip contains a 32.768KHz low frequency crystal oscillator (XTLF), a 4~24MHz high frequency crystal oscillator (XTHF), a high frequency RC oscillator (RCHF) up to 24MHz, a 32KHz low power internal RC oscillator (LPOSC), a 4MHz low power RC oscillator, and a phase-lock-loops (PLL). Clock management unit (CMU) integrates these clocks to generate operating clocks for CPU and peripherals.

Features:

- Multiple clock sources can be selected for the system clock
- Clocks can be switched on-the-fly during system operation
- Fail detection for XTLF
- Independent operating clocks for certain peripherals (decoupled from CPU and bus clocks)
- System frequency up to 64MHz

11.2 Clock tree

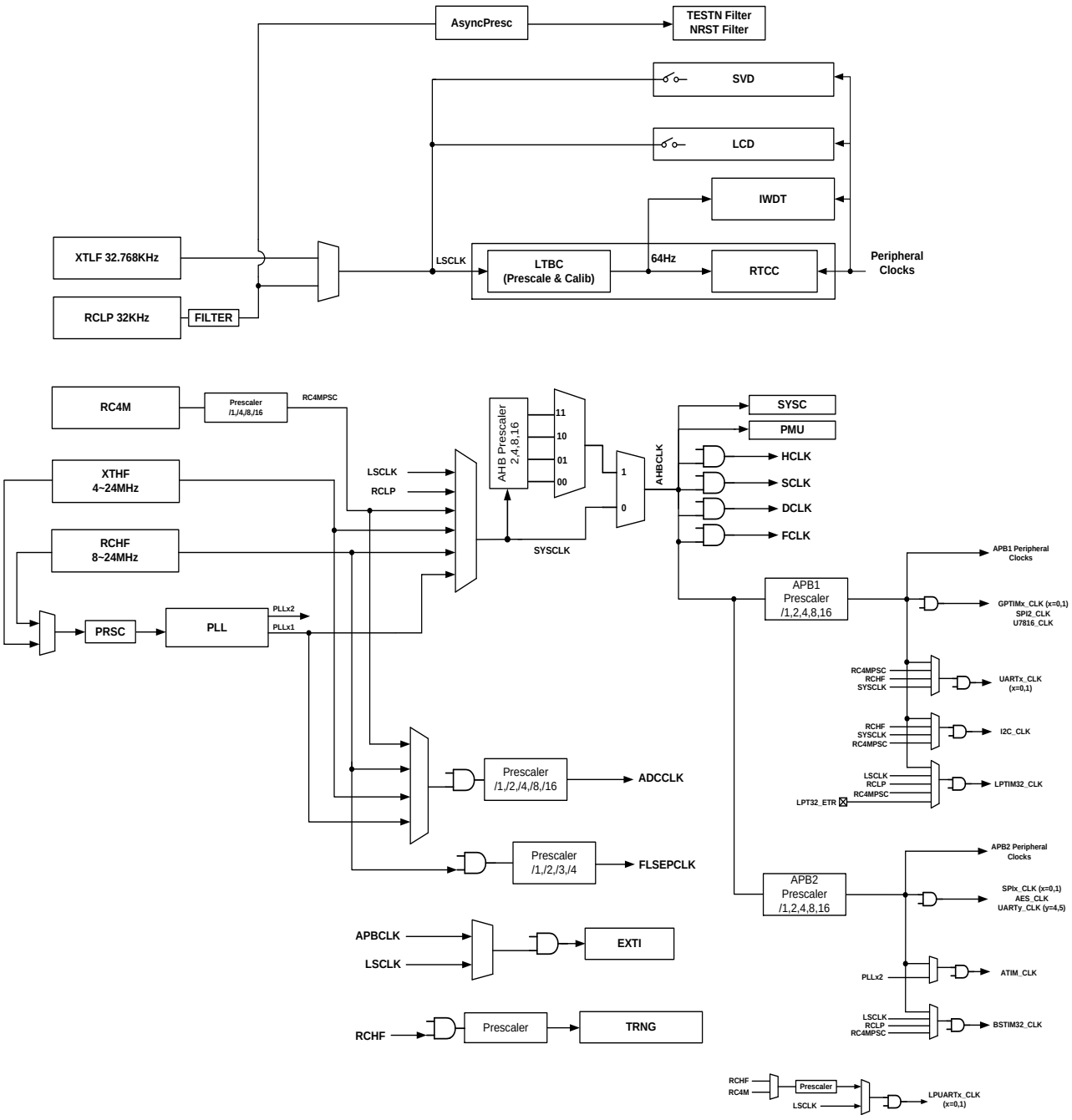


Figure 11-1 Clock tree diagram

The peripheral clocks (SYSCLK) can be generated from XTLF, RCHF, LPOSC, PLL, XTHF, RCMF and their divided clocks. By default, 8MHz RCHF is used as the system clock after power-up. The clock of each peripheral can be controlled separately. When the chip is working, only the module clock that needs to work can be turned on, and the clocks of other modules can be turned off to save power. APBCLK is divided from AHBCLK and is used to drive low speed peripherals.

11.2.1 Introduction for SYSCLK switching

SYSCLK is the main clock of the system. Bus clocks such as AHBCLK and APBCLK and the clock required for CPU operation can be obtained from SYSCLK.

When SYSCLK selects any clock source, the hardware must check whether the corresponding clock source is turned on. If the clock source is not enabled, the software switching operation is invalid, the SYSCLKSEL register will not be rewritten, and the clock switching will not occur.

Target clock	Switching condition
RCHF	RCHF is enabled
RCMF	RCMF is enabled
XTHF	XTHF is enabled and does not fail
PLL	PLL is enabled, and: **1，** If the PLL reference clock is XTHF, XTHF must be enabled and does not fail **2，** If the PLL reference clock is RCHF, RCHF must be enabled
XTLF	XTLF is enabled and does not fail
LPOSC	LPOSC is enabled

Table 11-1 System clock switching conditions

11.2.2 Introduction for main clocks

Name	Source	Description
LSCLK	XTLF, LPOSC	32KHz low frequency system clock, automatically switched to LPOSC when XTLF fails Mainly used for RTC, IWDT, pin filtering, SVD,LCD
SYSCLK	RCHF, PLL, SCLK, XTHF, RCMF	32K~64MHz, AHBCLK is the synchronous or divided version of SYSCLK
HCLK(AHBCLK)	SYSCLK	AHBbus clock driving CPU,RAM,Flashand high-speed peripherals
SCLK	SYSCLK	Core clock
DCLK	SYSCLK	Core Debugclock (This clock must be on when the debug probe is connected)
FCLK	SYSCLK	Core Free-Running clock
APBCLK	AHBCLK	The peripheral bus clock， to drive low peripherals

Table 11-2 Main system clocks description

11.2.3 Bus clocks and operating clocks for peripherals

The bus clocks and operating clocks of some peripherals are independent from each other.

The bus clock is used for AHB or APB bus access. When the software accesses the function registers of the peripheral, the corresponding bus clock must first be enabled through the peripheral

bus clock control register.

The operating clock of the peripheral is the clock actually used for peripheral operation, which may be different from APBCLK or AHBCLK. Before the peripheral module works, it needs to select the required clock source through the peripheral working clock register and open the clock gating.

For peripheral modules where the operating clock and the bus clock are unified, only the bus clock needs to be enabled for normal operation.

Module	Bus clocks	Operating clocks
Independent operating clock peripherals		
UARTx (x=0,1)	APB1CLK	APBCLK
		RCHF
		SYSCLK
		RCMF
LPUART0	APB1CLK	LSCLK
		RCHF
		RCMF
LPUART1	APB2CLK	LSCLK
		RCHF
		RCMF
I2C	APB1CLK	APBCLK
		RCHF
		SYSCLK
		RCMF
ATIM	APB2CLK	APBCLK
		PLLx2_CLK
LPTIM32	APB1CLK	APBCLK
		LSCLK
		LPOSC
		LPTIN
BSTIM32	APB2CLK	APBCLK
		LSCLK
		LPOSC
		RCMF _PSC
ADC	APB2CLK	RCMF
		XTHF
		RCHF
		PLL
NVMIF (Flash erase/program)	AHBCLK	RCHF
EXTI (PADCFG)	AHBCLK	AHBCLK
		LSCLK
TRNG	APB2CLK	RCHF
IWDT	APB1CLK	LSCLK
LCD	APB1CLK	LSCLK
RTC	APB1CLK	LSCLK

Module	Bus clocks	Operating clocks
Non-independent operating clock peripherals		
PMU	AHBCLK	
DMA	AHBCLK	
GPTIMx	APB1CLK	
UARTy (y=4,5)	APB2CLK	
SPI2	APB1CLK	
SPI1	APB2CLK	
7816	APB1CLK	
AES	APB2CLK	
CRC	APB2CLK	
WWDT	APB1CLK	
COMPx	APB2CLK	

Table 11-3 Peripheral module operating clocks and bus clocks

11.2.4 Peripheral clock in sleep mode

In Sleep/DeepSleep mode, SYSCLK is disabled. In Sleep mode AHBCLK and APBCLK are disabled and all peripherals based on AHBCLK or APBCLK stop working. However, peripherals with independent operating clock can still work, e.g., UART0/1, LPUARTx, I2C, ATIM, LPTIM32, BSTIM32.

In order for the above peripherals to continue to work in sleep mode, the software needs to ensure that the above peripherals work with clocks other than SYSCLK and the bus clock before sleep.

11.2.5 LSCLK switching logic

LSCLK is a low-speed clock for RTC, IWDT, SVD and LCD drivers, with a typical frequency of about 32KHz. The source of LSCLK is XTLF or LPOSC, and the chip supports automatic switching or manual switching by software between the two.

The automatic switching function of LSCLK is configured by the LSACTS register and is only valid when XTLF is enabled. At this time, it is assumed that XTLF is the main clock and LPOSC is the backup clock, which is only used to prevent XTLF from abnormally failing. Therefore, LSCATS only works when XTLF is enabled. When XTLF fails unexpectedly, the fail signal output by FDET will automatically switch LSCLK to LPOSC.

When the LSCLK automatic switch is not enabled, the software can manually switch LSCLK through the LSCLKSEL register. After the chip is powered on and reset, XTLFEN=0, LFDET output will not fail, LSCLKSEL selects LPOSC by default, and XTLF has no output. Therefore, LSCLK defaults to LPOSC after power on.

After that, if the software wants to use XTLF, it should enable XTLF and poll the LFDET output until it confirms that XTLF starts to oscillate, then set LSCATS or write 0xAA to LSCLKSEL.

When LSCLK is used as the system clock (SYSCLK), it is recommended that the software enable the automatic switching function of failure of oscillation.

11.3 High frequency RC oscillators (RCHF)

11.3.1 Introduction

High frequency RC oscillators with typical oscillation frequencies of 8/16/24MHz can be used as the system clock at which the MCU operates to achieve a balance between performance and power consumption. To satisfy the need of different applications for MCU execution speed, the output frequency of the high frequency RC oscillator can be adjusted up to 24MHz. The RCHF output frequency is factory-trimmed to within +/-1% of the target frequency at room temperature. RCHF has less than +/-1% frequency variation over the full temperature rang (-40 to +85°C) for the 8MHz output and less than +/-2% frequency variation over the full temperature rang (-40 to +85°C) for the 16MHz output.

11.3.2 Software control description

The chip operates with the RCHF 8MHz clock by default after power-up. The hardware will automatically load the 8MHz calibration value from Flash to ensure that the 8MHz room temperature frequency error is less than +/-0.5%.

Follow the following steps to select other frequencies for SYSCLK.

- Write RCHF_CR.FSEL
- Read the frequency calibration values (corresponding to 8/16/24MHz respectively) from Flash (0x1FFF_FB40,0x1FFF_FB3C,0x1FFF_FB38)
- Write the frequency trim values into the RCHF_TR register to obtain room temperature frequency error less than +/-1%.

The RCHF calibration parameter data format in Flash is as follows:

AHB address	bit[31:16]	bit[15:0]	Description
0x1FFF_FB38	{9'b0000_0000_0, ~RCHF24M_TRIM}	{9'b1111_1111_1, RCHF24M_TRIM}	RCHF 24Mhz trim value (Software

			loading)
0x1FFF_FB3C	{9'b0000_0000_0, ~RCHF16M_TRIM}	{9'b1111_1111_1, RCHF16M_TRIM}	RCHF 16Mhz trim value (Software loading)
0x1FFF_FB40	{9'b0000_0000_0, ~RCHF8M_TRIM}	{9'b1111_1111_1, RCHF8M_TRIM}	RCHF 8Mhz trim value (Load automatically after power-on)

Table 11-4 RCHF calibration data

The RCHF calibration value is 7bit data, and the high 16bit and low 16bit in the above flash address respectively store the calibration value and the ones-complement code check word. If the true code and ones-complement code fail to be checked, the calibration value shall be discarded.

11.4 Medium frequency RC oscillators (RCMF)

11.4.1 Introduction

The RCMF is a low-power medium frequency RC oscillator with a typical frequency of 4MHz and a typical power consumption of only about 20uA. It is used for low-power and low-speed operation of the CPU.

The RCMF is tested and calibrated when it leaves the factory. Before use, the software can read the calibration value from the address 0x1FFF_FB44 and write it into the RCMFTR register to obtain a 4M clock with an error of less than +/-1% at room temperature.

The RCMF calibration parameter data format in Flash is as follows:

AHB address	bit[31:16]	bit[15:0]	Description
0x1FFF_FB44	{9'b0000_0000_0, ~RCMF_TRIM}	{9'b1111_1111_1, RCMF_TRIM}	RCMF trim value

Table 11-5 RCMF calibration data

The RCMF calibration value is 7bit data, and the high 16bit and low 16bit in the above flash address respectively store the calibration value and the ones-complement code check word. If the true code and ones-complement code fail to be checked, the calibration value shall be discarded.

11.5 High precision low power RC oscillators (LPOSC)

11.5.1 Introduction

LPOSC has extremely low power consumption, only a few hundred nA, and can be used as a backup clock for the XTLF or used alone.

The LPOSC is tested and calibrated when it leaves the factory. Before use, the software can read the calibration value from the address 0x1FFF_FB20 and write it into the LPOSCTR register to obtain a calibrated low-frequency clock.

The LPOSC calibration parameter data format in Flash is as follows:

AHB address	bit[31:16]	bit[15:0]	Description
0x1FFF_FB20	{8'b0000_0000, ~LPOSC_TRIM}	{8'b1111_1111, LPOSC_TRIM}	LPOSCtrim value

Table 11-6 LPOSC calibration data

The LPOSC calibration value is 8bit data, and the high 16bit and low 16bit in the above flash address respectively store the calibration value and the ones-complement code check word. If the true code and ones-complement code fail to be checked, the calibration value shall be discarded.

11.6 Low frequency crystal oscillation circuits (XTLF)

11.6.1 Introduction

The low frequency crystal oscillator provides a stable oscillation source through an external 32768Hz crystal with very low power consumption. It is mainly used for the Real Time Clock (RTC) module. The XTLF's feedback resistor is integrated into the chip and the user needs to add load capacitor to external crystal.

A fail detection circuit is integrated into the chip to detect whether the XTLF fails. Interrupt will be generated when XTLF failure is detected, and the CPU will be notified to deal with it in time.

Software can enable or disable XTLF. In order to improve the anti-interference ability, the 4-bit XTLFEN control bit is used, and the 4-bit reset value is 0101, which must be rewritten to 1010 to disable XTLF, and any other data will keep XTLF enabled. Meanwhile, the self-detect circuit working under XTLF will constantly monitor the XTLFEN register data, and if data other than 0101 and 1010 appears, it should be automatically modified to 0101.

11.6.2 Software control description

The XTLF is disabled by default after power-up. When the software starts, a medium driving is used by default to shorten the oscillation time. Once the oscillator has fully started, the software can configure the registers to reduce the oscillation power consumption.

11.6.3 Fail detection

FM33LE0xxA has an on-chip fail detection circuit, which can continuously detect XTLF output when enabled. When an XTLF fail is detected, an alarm interrupt is generated and the software can decide whether to switch LSCLK to LPOSC automatically through the LSCATS register.

When LSCATS=1, the hardware will automatically enable LPOSC and switch LSCLK to LPOSC output when FDET detects XTLF stop; when LSCATS=0, the stop detection will only generate an alarm interrupt and will not automatically switch the clock.

The software can also switch the XTLF by setting LSCATS in the XTLF deactivation state.

The fail detection circuit always opens or closes at the same time with XTLF, and cannot be closed separately. Once XTLF is enabled, the fail detection circuit will open automatically; when XTLF is closed, the fail detection will also close automatically to avoid false triggering of the stop alarm

11.7 High frequency crystal oscillation circuits (XTHF)

11.7.1 Introduction

By connecting external high frequency crystal, the XTHF is able to provide a high precision high frequency clock source for the MCU. Load capacitors should be placed as close as possible to the XTHF pins, and capacitance should be chosen to suit the type of crystal.

The XTHF can accommodate crystals from 4 to 24MHz. The software can enable or disable the XTHF clock via the XTHFEN register.

11.7.2 Software control description

XTHF is disabled by default after power-on. After the power-on reset is complete, the software can enable XTHF as needed. Because the crystal oscillator pins are multiplexed with GPIO, the PC2 and PC3 pins need to be configured as analog functions before XTHF is enabled by the software.

11.7.3 Fail detection (HFDET)

FM33LE0xxA has an on-chip stop detection circuit, which is enabled or disabled together with the XTHF circuit. When the stop detection is enabled, the XTHF output can be continuously detected, and when the XTHF is found to be failed, an alarm interrupt is generated and an advanced timer brake signal is generated. If XTHF is being used as the system operating clock directly or indirectly ('directly' means SYSCLK is selected as XTHF, 'indirectly' means SYSCLK is selected as PLL and PLL uses XTHF as the input reference clock), the fail signal will automatically enable RCHF and switch SYSCLK to RCHF to avoid system crash due to unexpected stop of the high frequency crystal.

The fail detection circuit always opens or closes at the same time with XTHF and cannot be closed separately. Once XTHF is enabled, the stop detection circuit will open automatically; when XTHF is closed, the stop detection will also close automatically to avoid false triggering of the stop alarm.

The HFDET fail detection threshold is about 200KHz, that is, the XTHF clock frequency below 200KHz triggers the stop alarm; the HFDET operating current is about 1.2uA.

11.8 Phase Locked Loop (PLL)

11.8.1 Introduction

The PLL input reference clock can be divided from RCHF or XTHF, and maximum output frequency is 64MHz and its double frequency. The input reference clock and multiplication factor need to be configured before using PLL as system clock.

11.8.2 Software control description

In order to improve reliability, the following points need to be noted in the configuration.

- The software must ensure that RCHF or XTHF is enabled when the PLL input is selected
- PLL cannot be turned off when the PLL output is selected as SYSCLK
- Software should wait for the PLL to lock before configuring SYSCLK as the PLL output
- The PLL clock source RCHF or XTHF must be divided to 1MHz and then multiplied

Configure the PLL output at 64MHz and the system operate at 64MHz frequency:

- Configure PLLCR register to select input clock source and output clock frequency

- Set the Flash wait cycle to 2
- Select the SYSCLK clock as the PLL output

11.9 Clock source in low power mode

In low power mode, some of the clock sources are disabled by hardware, while others remain operational. See the following table for details.

Clock	LPRUN/Sleep/DeepSleep	Intro.
RCHF	X	Hardware forced shutdown
PLL	X	
XTHF	X	
RCMF	O	Software configuration to enable or disable
LPOSC	O	
XTLF	O	

Table 11-7 Clock source in low power mode

11.10 Clock selection after wake up from sleep

When the chip wakes up from Sleep/DeepSleep mode, the hardware automatically turns on RCHF and returns to the frequency output before sleep; the SYSCLKSEL register is reset to 00, the system clock is switched to RCHF, and the AHBPRES register will not be reset and keep the state before sleep; therefore the chip will use RCHF or its divided clocks by default after waking up.

11.11 Register

Offset	Name	Symbol
RCC(Base address:0x40000200)		
0x00000000	Lockup Reset Control Register	RCC_LKPCR
0x00000004	Software Reset Register	RCC_SOFTRST
0x00000008	Reset Flag Register	RCC_RSTFR
0x0000000C	System Clock Control Register	RCC_SYSCLKCR
0x00000010	RCHF Control Register	RCC_RCHFCR
0x00000014	RCHF Trim Register	RCC_RCHFTR
0x00000018	PLL Control Register	RCC_PLLCR
0x0000001C	LPOSC Control Register	RCC_LPOSCCR
0x00000020	LPOSC Trim Register	RCC_LPOSCTR
0x00000024	XTLF Control Register	RCC_XTLFCR
0x00000028	Peripheral bus Clock Control Register1	RCC_PCLKCR1
0x0000002C	Peripheral bus Clock Control Register2	RCC_PCLKCR2
0x00000030	Peripheral bus Clock Control Register3	RCC_PCLKCR3
0x00000034	Peripheral bus Clock Control Register4	RCC_PCLKCR4
0x00000038	LSCLK Select Register	RCC_LSCLKSEL
-	-	-
-	-	-
0x00000044	AHB Master Control Register	RCC_AHBMCR
-	-	-
0x00000050	Peripheral Reset Enable Register	RCC_PRSTEN
0x00000054	AHB Peripherals Reset Control Register	RCC_AHBRSTCR
0x00000058	APB Peripherals Reset Control Register1	RCC_APBRSTCR1
0x0000005C	APB Peripherals Reset Control Register2	RCC_APBRSTCR2
0x00000060	XTHF Control Register	RCC_XTHFCR
0x00000064	RCMF Control Register	RCC_RCMFCR
0x00000068	RCMF Trim Register	RCC_RCMFTR
0x0000006C	Peripheral Operation Clock Control Register1	RCC_OPCCR1
0x00000070	Peripheral Operation Clock Control Register2	RCC_OPCCR2
0x00000074	PHY Control Register	RCC_PHYCR
0x00000078	PHY BCK Control Register	RCC_PHYBCKCR

11.11.1 Lockup Reset Control Register (RCC_LKPCR)

NAME	RCC_LKPCR							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						LKUPRST_EN	-
access	U-0						R/W-0	U-0

bit	name	functional description
31:2	-	RFU: Reserved, read as 0
1	LKUPRST_EN	Lockup Reset Enable 1: Enable SC000 LOCKUP reset 0: Disable SC000 LOCKUP reset
0	-	RFU: Reserved, read as 0

11.11.2 Software Reset Register (RCC_SOFTRST)

NAME	RCC_SOFTRST							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	SOFTRST[31:24]							
access	W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	SOFTRST[23:16]							
access	W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	SOFTRST[15:8]							
access	W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SOFTRST[7:0]							
access	W-0000 0000							

bit	name	functional description
31:0	SOFTRST	Software write 0x5C5C_AABBto trigger a global reset (software reset, write only)

11.11.3 ResetFlag Register (RCC_RSTFR)

NAME	RCC_RSTFR							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

name	-			MDFN_FLAG	NRSTN_FLAG	TESTN_FLAG	PORN_FLAG	PDRN_FLAG
access	U-0			R/W-0	R/W-0	R/W-0	R/W-1	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-		SOFTN_FLAG	IWDTN_FLAG	-	WWDTN_FLAG	LKUPN_FLAG	NVICN_FLAG
access	U-0		R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:12	-	RFU: Reserved, read as 0
12	MDFN_FLAG	MDFN reset Flag, write 1 to clear
11	NRSTN_FLAG	NRST reset Flag, write 1 to clear
10	TESTN_FLAG	TESTN reset Flag, write 1 to clear
9	PORN_FLAG	Power-up-reset Flag, write 1 to clear
8	PDRN_FLAG	Power-down-reset Flag, write 1 to clear
7:6	-	Reserved, read as 0
5	SOFTN_FLAG	Software reset flag, write 1 to clear
4	IWDTN_FLAG	IWDT reset flag, write 1 to clear
3	-	Reserved, read as 0
2	WWDTN_FLAG	WWDT reset flag, write 1 to clear
1	LKUPN_FLAG	Lockup reset flag, write 1 to clear
0	NVICN_FLAG	NVIC reset flag, write 1 to clear

11.11.4 System Clock Control Register (RCC_SYSCLKCR)

NAME	RCC_SYSCLKCR							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-				LSCATS	-	SLP_EN EXTI	-
access	U-0				R/W-1	U-0	R/W-1	U-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-		APBPRES2			APBPRES1		
access	U-0		R/W-000			R/W-000		
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-					AHBPRES		
access	U-0					R/W-011		
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RFU		-		BCKOSEL	SYSCLKSEL		
access	R/W-00		U-0		R/W-0	R/W-000		

bit	name	functional description
31:28	-	Reserved, read as 0
27	LSCATS	LSCLK automatic switch enable 0: When detecting abnormal XTLF failure, LSCLK will not be automatically switched to LPOSC, software can manually switch to LPOSC by writing the LSCLKSEL register 1: When detecting abnormal XTLF failure, automatically enable

bit	name	functional description
		LPOSC and switch LSCLK to LPOSC
26	-	Reserved, read as 0
25	SLP_ENEXTI	EXTI sampling config in Sleep/DeepSleep mode 1: Enable external pin interrupt sampling in Sleep/DeepSleepmode (sampling clock is LSCLK) 0: External pin interrupt sampling is disable in Sleep/DeepSleep mode(EXTI interrupts will not be generated)
24:22	-	Reserved, read as 0
21:19	APBPRES2	APB2clock divide-ratio from AHBCLK (APB2 bus clock Prescaler) 0xx: divided-by-1 100: divided-by-2 101: divided-by-4 110: divided-by-8 111: divided-by-16 Note: This register is located in PD Domain
18:16	APBPRES1	APB1 clock divide-ratio from AHBCLK (APB1bus clock Prescaler) 000/001/010/011: divided-by-1 100: divided-by-2 101: divided-by-4 110: divided-by-8 111: divided-by-16
15:11	-	RFU: Reserved, read as 0
10:8	AHBPRES	AHBclock divide-ratio from SYSCLK (AHB bus clock Prescaler) 000/001/010/011: divided-by-1 100: divided-by-2 101: divided-by-4 110: divided-by-8 111: divided-by-16
7:6	RFU	Dummy register
5:4	-	Reserved, read as 0
3	BCKOSEL	USB PHY BCK output clock selection signal(USB clock select) 0: Select 48M BCKoutput as system clock source 1:Select the two-frequency division of 120M BCK output as the system clock source
2:0	SYSCLKSEL	System clock source selection 000：RCHF 001：XTHF 010：PLL 011：RCHF 100：RCMFPSC 101：LSCLK 110：LPOSC 111：RFU

11.11.5 RCHF Control Register (RCC_RCHFCR)

NAME	RCC_RCHFCR							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-				FSEL			
access	U-0				R/W-0000			
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							EN
access	U-0							R/W-1

bit	name	functional description
31:20	-	RFU: Reserved, read as 0
19:16	FSEL	RCHF frequency selection register 0000:8MHz 0001:16MHz 0010:24MHz Others: RFU
15:1	-	RFU: Reserved, read as 0
0	EN	RCHF enable register 1: Enable RCHF 0: Disable RCHF

11.11.6 RCHFTrim Register (RCC_RCHFTR)

NAME	RCC_RCHFTR							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	TRIM						
access	U-0	R/W-100 0000						

bit	name	functional description
31:7	-	RFU: Reserved, read as 0
6:0	TRIM	RCHF frequency trim register, 7'h00 means the lowest frequency, 7'h7F means the highest frequency, the trimming range is +/-30% of the center frequency, the trimming step is 0.5% After power on, the chip automatically reads the 8MHz trimming value from NVR1 and writes it into this register When the software uses frequencies other than 8MHz, it can read the trimming information from the address specified in NVR1 and write it to this register, thus ensuring the output frequency is accurate at room temperature.

11.11.7 PLL Control Register (RCC_PLLCR)

NAME	RCC_PLLCR							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-	DB						
access	U-0	R/W-010 1111						
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	LOCKED	REFPRSC			OSEL	-	INSEL	EN
access	R-0	R/W-000			R/W-0	U-0	R/W-0	R/W-0

bit	name	functional description
31:23	-	RFU: Reserved, read as 0
22:16	DB	PLL multiplication factor (PLL Divide Boost) 0011111: Output 32times frequency 0101111: Output 48times frequency
15:8	-	RFU: Reserved, read as 0
7	LOCKED	PLL locked flag, the software confirms that the PLL is in a locked state by querying this register 1：PLL is locked 0：PLL is not locked
6:4	REFPRSC	PLL reference clock prescaler(The goal is to generate a 1MHz reference clock to the PLL) 000: divided-by-1 001: divided-by-2 010: divided-by-4 011: divided-by-8 100: divided-by-12 101: divided-by-16 110: divided-by-24 111: divided-by-32

bit	name	functional description
3	OSEL	PLL output selection register 0: Select PLL double output as the PLL clock in the digital circuit 1: Select PLL twice the output as the PLL clock in the digital circuit
2	-	Reserved, read as 0
1	INSEL	PLL input selection register 0: RCHF 1: XTHF
0	EN	PLL enable register 1: Enable PLL 0: Disable PLL

11.11.8 LPOSC Control Register (RCC_LPOSCCR)

NAME	RCC_LPOSCCR							
Offset	0x0000001C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					LPO_CHOP_EN	LPO_ENB	LPM_LPO_OFF
access	U-0					R/W-0	R-0	R/W-0

bit	name	functional description
31:1	-	RFU: Reserved, read as 0
0	LPOENB	LPOSC Enable register 0: Enable LPOSC 1: Disable LPOSC

11.11.9 LPOSC Trim Register (RCC_LPOSCTR)

NAME	RCC_LPOSCTR							
Offset	0x00000020							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							

bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	LPOTRIM							
access	R/W-1000 1101							

bit	name	functional description
31:8	-	RFU: Reserved, read as 0
7:0	LPOTRIM	LPOSC trimming value register 0000 0000: Lowest frequency 1111 1111: Highest frequency

11.11.10 XTLF Control Register (RCC_XTLFCR)

NAME	RCC_XTLFCR							
Offset	0x00000024							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				EN			
access	U-0				R/W-xxxx			
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					IPW		
access	U-0					R/W-000		

bit	name	functional description
31:12	-	Reserved, read as 0
11:8	EN	XTLF enable register, XTLF is disabled by default after power-on 1010: Disable XTLFand FDET 0101: Enable XTLFand FDET Others: RFU When XTLF is working, the software must write 1010 to disable it; when XTLF is not working, the software must write 0101 to enable it
7:3	-	Reserved, read as 0
2:0	IPW	XTLF working current selection, the larger the current, the higher the oscillation intensity. After power-on reset, use the 000 gear to start the oscillation. It is recommended to use the 100or 011 gear during normal operation. Actually, the appropriate current should be selected according to the measured negative resistance characteristics of the adapted crystal. 000: 450 nA 001: 400 nA 010: 350 nA

bit	name	functional description
		011: 300 nA 100: 250 nA 101: 200 nA 110: 150 nA 111: 100 nA

Bit	name	functional description
31:19	--	RFU：Reserved, read as 0
18:16	ST_WAIT	XTLF start-up wait time configuration. The unit is XTLF clock cycles 000：128 001：256 010：512 011：1024 100：2048 101：4096 110：8192 111：16384
15:12	--	RFU：Reserved, read as 0
11:8	XTLFEN	Enable XTLF register. Default is disable XTLF after power-on 1010：Disable XTLF and FDET 0101：Enable XTL and FDET Only 1bit in physically, when XTLF is working, software must write 1010 to turn it off, when XTLF is not working, software must write 0101 to start
7:3	--	RFU：Reserved, read as 0
2:0	XTLFIPW	XTLF working current selection, the larger the current, the higher the oscillation intensity. After power-on reset, use the 000 gear to start the oscillation. It is recommended to use the 100or 011 gear during normal operation. Actually, the appropriate current should be selected according to the measured negative resistance characteristics of the adapted crystal. 000 : 450 nA 001 : 400 nA 010 : 350 nA 011 : 300 nA 100 : 250 nA 101 : 200 nA 110 : 150 nA 111 : 100 nA

11.11.11 Peripheral bus Clock Control Register1 (RCC_PCLKCR1)

NAME	RCC_PCLKCR1							
Offset	0x00000028							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	DCU_PCE	-						
access	R/W-1	U-0						
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16

name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	PAD_PCE	ANAC_PCE	IWDT_PCE	SCU_PCE	PMU_PCE	RTC_PCE	USB_PCE	LPT_PCE
access	R/W-0	R/W-1	R/W-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0

bit	name	functional description
31	DCU_PCE	Debug Control Unit APB bus clock enable,write 1 enable
30:8	-	Reserved, read as 0
7	PAD_PCE	GPIO controller APB bus clock enable, write 1 enable
6	ANAC_PCE	Analog controller APB bus clock enable, write 1 enable
5	IWDT_PCE	IWDTAPB bus clock enable, write 1 enable
4	SCU_PCE	System controller APB bus clock enable, write 1 enable
3	PMU_PCE	PMUAPB bus clock enable, write 1 enable
2	RTC_PCE	RTC APB bus clock enable, write 1 enable
1		RFU；Reserved, read as 0
0	LPT_PCE	LPTIM APB bus clock enable, write 1 enable

11.11.12 Peripheral bus Clock Control Register2 (RCC_PCLKCR2)

NAME	RCC_PCLKCR2							
Offset	0x0000002C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						HDIV_PCE	ADC_PCE
access	U-0						R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	WWDT_PCE	RAMBIST_PCE	FLASH_PCE	DMA_PCE	LCD_PCE	AES_PCE	TRNG_PCE	CRC_PCE
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:10	-	Reserved, read as 0
9	HDIV_PCE	Hardware Divider APB bus clock enable, write 1 enable
8	ADC_PCE	ADC controller APB bus clock enable, write 1 enable
7	WWDT_PCE	WWDT APB bus clock enable, write 1 enable
6	RAMBIST_PCE	RAMBIST APB bus clock enable, write 1 enable
5	FLASH_PCE	Flash interface APB bus clock enable, write 1 enable

bit	name	functional description
4	DMA_PCE	DMA APB bus clock enable, write 1 enable
3	LCD_PCE	LCD APB bus clock enable, write 1 enable
2	AES_PCE	AES APB bus clock enable, write 1 enable
1	RNG_PCE	RNG APB bus clock enable, write 1 enable
0	CRC_PCE	CRC APB bus clock enable, write 1 enable

11.11.13 Peripheral bus Clock Control Register3 (RCC_PCLKCR3)

NAME	RCC_PCLKCR3							
Offset	0x00000030							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							I2C_PCE
access	U-0							R/W-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-					LPUART1_PCE	-	U7816_PCE
access	U-0					R/W-0	U-0	R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	LPUART0_PCE	UCIR_PCE	UART5_PCE	UART4_PCE	-		UART1_PCE	UART0_PCE
access	R/W-0	R/W-0	R/W-0	R/W-0	U-0		R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						SPI2_PCE	SPI1_PCE
access	U-0						R/W-0	R/W-0

bit	name	functional description
31:25	-	Reserved, read as 0
24	I2C_PCE	I2C APB bus clock enable, write 1 enable
23:19	-	Reserved, read as 0
18	LPUART1_PCE	LPUART1 APB bus clock enable, write 1 enable
17	-	Reserved, read as 0
16	U7816_PCE	U7816 APB bus clock enable, write 1 enable
15	LPUART0_PCE	LPUART0 APB bus clock enable, write 1 enable
14	UCIR_PCE	UART infra-red APB bus clock enable, write 1 enable
13	UART5_PCE	UART5 APB bus clock enable, write 1 enable
12	UART4_PCE	UART4 APB bus clock enable, write 1 enable
11:10	-	Reserved, read as 0
9	UART1_PCE	UART1 APB bus clock enable, write 1 enable
8	UART0_PCE	UART0 APB bus clock enable, write 1 enable
7:2	-	Reserved, read as 0
1	SPI2_PCE	SPI2 APB bus clock enable, write 1 enable
0	SPI1_PCE	SPI1 APB bus clock enable, write 1 enable

bit	name	functional description
31:26	-	RFU：Reserved, read as 0
25	I2CSMB_PCE	I2C_SMBUS APB bus clock enable, write 1 enable
24	I2C_PCE	I2C bus clock enable, write 1 enable
23:19	-	RFU：Reserved, read as 0
18	LPUART1_PCE	LPUART1 APB bus clock enable, write 1 enable
17	-	RFU：Reserved, read as 0
16	U7816_PCE	7816 APB bus clock enable, write 1 enable
15	LPUART0_PCE	LPUART APB bus clock enable, write 1 enable
14	UCIR_PCE	UART APB bus clock enable, write 1 enable
13	UART5_PCE	UART5 APB bus clock enable, write 1 enable
12	UART4_PCE	UART4 APB bus clock enable, write 1 enable
11	-	RFU：Reserved, read as 0
10	UART2_PCE	UART2 APB bus clock enable, write 1 enable
9	UART1_PCE	UART1 APB bus clock enable, write 1 enable
8	UART0_PCE	UART0 APB bus clock enable, write 1 enable
7:2	-	RFU：Reserved, read as 0
1	SPI2_PCE	SPI2 APB bus clock enable, write 1 enable
0	SPI1_PCE	SPI1 APB bus clock enable, write 1 enable

11.11.14 Peripheral bus Clock Control Register4 (RCC_PCLKCR4)

NAME	RCC_PCLKCR4							
Offset	0x00000034							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							

access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-			AT_PCE	GT1_PCE	GT0_PCE	-	BT_PCE
access	U-0			R/W-0	R/W-0	R/W-0	U-0	R/W-0

bit	name	functional description
31:5	-	Reserved, read as 0
4	AT_PCE	ATIM APB bus clock enable, write 1 enable
3	GT1_PCE	GPTIM1 APB bus clock enable, write 1 enable
2	GT0_PCE	GPTIM0 APB bus clock enable, write 1 enable
1	-	Reserved, read as 0
0	BT_PCE	BSTIM APB bus clock enable, write 1 enable

11.11.15 LSCLK Select Register (RCC_LSCLKSEL)

NAME	RCC_LSCLKSEL							
Offset	0x00000038							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SEL							
access	R/W-0000 0001							

bit	name	functional description
31:8	-	Reserved, read as 0
7:0	SEL	When LSCLK is XTLF, the software writes 0x55 to this address, which will switch the source of LSCLK to LPOSC When LSCLK is LPOSC, the software writes 0xAA to this address, which will switch the source of LSCLK to XTLF Writing any other value will not change the current LSCLK; this register is only valid when LSCATS is 0

11.11.16 AHB Master Control Register (RCC_AHBMCR)

NAME	RCC_AHBMCR							
Offset	0x00000044							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							MPRIL
access	U-0							R/W-0

bit	name	functional description
31:30	RFUI	Reserved bit
29:1	--	Reserved, read as 0
0	MPRIL	AHB MasterPriority Config Register 0:DMApriority

bit	name	functional description
		1:CPUpriorit1：CPU 优先

11.11.17 Peripheral Reset Enable Register (RCC_PRSTEN)

NAME	RCC_PRSTEN							
Offset	0x00000050							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	PERHRSTEN[31:24]							
access	W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	PERHRSTEN[23:16]							
access	W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	PERHRSTEN[15:8]							
access	W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	PERHRSTEN[7:0]							
access	W-0000 0000							

bit	name	functional description
31:0	PERHRSTEN	Peripheral module reset enable, 32bit virtual register, write only The software writes 0x1357_9BDF to this address to enable the peripheral reset function, and then each module can be reset through the peripheral module reset register The software writes any other data to this address, the peripheral reset function will be disabled

11.11.18 AHB Peripherals Reset Control Register (RCC_AHBRSTCR)

NAME	RCC_AHBRSTCR							
Offset	0x00000054							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						USBRST	DMARST
access	U-0						R/W-0	R/W-0

bit	name	functional description
31:2	-	Reserved, read as 0
1	USBRST	USB module reset, write 1 to reset, write 0 to cancel reset (USB reset Enable) 0:Not reset 1: Reset
0	DMARST	DMA module reset, write 1 to reset, write 0 to cancel reset (DMA reset Enable) 0: Not reset 1: Reset

bit	name	functional description
31:2	-	Reserved, read as 0
1	-	Reserved, read as 0
0	DMARST	DMA module reset, write 1 to reset, write 0 to cancel reset (DMA reset Enable) 0: Not reset 1: Reset

11.11.19 APB Peripherals Reset Control Register1 (RCC_APBRSTCR1)

NAME	RCC_APBRSTCR1							
Offset	0x00000058							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	UART5RST	UART4RST	-				GPT1RST	GPT0RST
access	R/W-0	R/W-0	U-0				R/W-0	R/W-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							LCDRST
access	U-0							R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-	U7816RST	-			SPI2RST	-	
access	U-0	R/W-0	U-0			R/W-0	U-0	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	LPUART0RST	-		I2C1RST	-		LPT32RST
access	U-0	R/W-0	U-0		R/W-0	U-0		R/W-0

bit	name	functional description
31	UART5RST	UART5 module reset, write 1 to reset, write 0 to cancel reset (UART5 reset Enable) 0: Not reset 1: Reset
30	UART4RST	UART4 module reset, write 1 to reset, write 0 to cancel reset (UART4 reset Enable) 0: Not reset 1: Reset
29:26	-	Reserved, read as 0
25	GPT1RST	GPTIM1 module reset, write 1 to reset, write 0 to cancel reset

bit	name	functional description
		(GPTIM1 reset Enable) 0: Not reset 1: Reset
24	GPT0RST	GPTIM0 module reset, write 1 to reset, write 0 to cancel reset (GPTIM0 reset Enable) 0: Not reset 1: Reset
23:17	-	Reserved, read as 0
16	LCDRST	LCD module reset, write 1 to reset, write 0 to cancel reset (LCD reset Enable) 0: Not reset 1: Reset
15	-	Reserved, read as 0
14	U7816RST	U7816 module reset, write 1 to reset, write 0 to cancel reset (U7816 reset Enable) 0: Not reset 1: Reset
13:11	-	Reserved, read as 0
10	SPI2RST	SPI2 module reset, write 1 to reset, write 0 to cancel reset (SPI2 reset) 0: Not reset 1: Reset
9:7	-	Reserved, read as 0
6	LPUART0RST	EUART0 module reset, write 1 to reset, write 0 to cancel reset (LPUART0 reset Enable) 0: Not reset 1: Reset
5:4	-	Reserved, read as 0
3	I2C1RST	I2C1 module reset, write 1 to reset, write 0 to cancel reset (I2C1 reset Enable) 0: Not reset 1: Reset
2:1	-	Reserved, read as 0
0	LPT32RST	LPTIM32 module reset, write 1 to reset, write 0 to cancel reset (LPTIM resetEnable) 0: Not reset 1: Reset

bit	name	functional description
31	UART5RST	UART5 module reset, write 1 to reset, write 0 to cancel reset (UART5 reset Enable) 0: Not reset 1: Reset
30	UART4RST	UART4 module reset, write 1 to reset, write 0 to cancel reset (UART4 reset Enable) 0: Not reset 1: Reset
29	-	RFU: Reserved, read as 0
28	UART2RST	UART2 module reset, write 1 to reset, write 0 to cancel reset (UART4 reset Enable) 0: Not reset

bit	name	functional description
		1: Reset
27:26	-	RFU: Reserved, read as 0
25	GPT1RST	GPTIM1 module reset, write 1 to reset, write 0 to cancel reset (GPTIM1 reset Enable) 0: Not reset 1: Reset
24	GPT0RST	GPTIM0 module reset, write 1 to reset, write 0 to cancel reset (GPTIM1 reset Enable) 0: Not reset 1: Reset
23:17	-	RFU: Reserved, read as 0
16	LCDRST	LCD module reset, write 1 to reset, write 0 to cancel reset (LCD reset Enable) 0: Not reset 1: Reset
15	-	RFU: Reserved, read as 0
14	U7816RST	U7816 module reset, write 1 to reset, write 0 to cancel reset (U7816 reset Enable) 0: Not reset 1: Reset
13:11	-	RFU: Reserved, read as 0
10	SPI2RST	SPI2 module reset, write 1 to reset, write 0 to cancel reset (SPI2 reset) 0: Not reset 1: Reset
9:7	-	RFU: Reserved, read as 0
6	LPUART0RST	EUART0 module reset, write 1 to reset, write 0 to cancel reset (LPUART0 reset Enable) 0: Not reset 1: Reset
5	-	RFU: Reserved, read as 0
4	I2CSMBRST	I2C_SMBUS module reset, write 1 to reset, write 0 to cancel reset (LPUART0 reset Enable) 0: Not reset 1: Reset
3	I2C1RST	I2C1 module reset, write 1 to reset, write 0 to cancel reset (LPUART0 reset Enable) 0: Not reset 1: Reset
2:1	-	RFU: Reserved, read as 0
0	LPT32RST	LPTIM32 module reset, write 1 to reset, write 0 to cancel reset (LPUART0 reset Enable) 0: Not reset 1: Reset

11.11.20 APB Peripherals Reset Control Register2 (RCC_APBRSTCR2)

NAME	RCC_APBRSTCR2							
Offset	0x0000005C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	ATRST	-		BT32RST	-			ADCCRST

access	R/W-0	U-0		R/W-0	U-0			R/W-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	ADCRST	OPARST	-		HDVRST	AESRST	CRCRST	RNGRST
access	R/W-0	R/W-0	U-0		R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			UART1RST	UART0RST	-	SPI1RST	UCIRRST
access	U-0			R/W-0	R/W-0	U-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	LPUART1RST	-						
access	R/W-0	U-0						

bit	name	functional description
31	ATRST	ATIM module reset, write 1 to reset, write 0 to cancel reset (ATIM reset Enable) 0: Not reset 1: Reset
30:29	-	Reserved, read as 0
28	BTRST	BSTIM32 module reset, write 1 to reset, write 0 to cancel reset (BSTIM32 reset Enable) 0: Not reset 1: Reset
27:25	-	Reserved, read as 0
24	ADCCRST	ADC controller reset, write 1 to reset, write 0 to cancel reset (ADC controller reset Enable) 0: Not reset 1: Reset
23	ADCRST	ADC module reset, write 1 to reset, write 0 to cancel reset (ADC reset Enable) 0: Not reset 1: Reset
22	-	Reserved, read as 0
21:20	-	Reserved, read as 0
19	HDVRST	Hardware divider reset, write 1 to reset, write 0 to cancel reset (Hardware Divider Reset Enable) 0: Not reset 1: Reset
18	AESRST	AES module reset, write 1 to reset, write 0 to cancel reset (AES reset Enable) 0: Not reset 1: Reset
17	CRCRST	CRC module reset, write 1 to reset, write 0 to cancel reset (CRC reset Enable) 0: Not reset 1: Reset
16	RNGRST	RNG module reset, write 1 to reset, write 0 to cancel reset (RNG reset Enable) 0: Not reset 1: Reset
15:13	-	Reserved, read as 0
12	UART1RST	UART1 module reset, write 1 to reset, write 0 to cancel reset (UART1 reset Enable)

bit	name	functional description
		0: Not reset 1: Reset
11	UART0RST	UART0 module reset, write 1 to reset, write 0 to cancel reset (UART0 reset Enable) 0: Not reset 1: Reset
10	-	Reserved, read as 0
9	SPI1RST	SPI1 module reset, write 1 to reset, write 0 to cancel reset (SPI1reset Enable) 0: Not reset 1: Reset
8	UCIRRST	UCIR module reset, write 1 to reset, write 0 to cancel reset (UCIR reset Enable) 0: Not reset 1: Reset
7	LPUART1RST	LPUART1 module reset, write 1 to reset, write 0 to cancel reset (LPUART1 reset Enable) 0: Not reset 1: Reset
6:0	-	Reserved, read as 0

11.11.21 XTHF Control Register (RCC_XTHFCR)

NAME	RCC_XTHFCR							
Offset	0x00000060							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-					CFG		
access	U-0					R/W-000		
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							EN
access	U-0							R/W-0

bit	name	functional description
31:19	--	Reserved, read as 0
18:16	ST_WAIT	XTHF start-up wait time configuration. The unit is XTHF clock cycle 000：128 001：256 010：512 011：1024 100：2048 101：4096 110：8192 111：16384

bit	name	functional description
15:11	--	Reserved, read as 0
10:8	HF_CFG	XTHF oscillation strength config 000: Weakest 111:Strongest
7:1	--	Reserved, read as 0
0	XTHFEN	XTHF enable register 0: Disable XTHF 1:Enable XTHF

11.11.22 RCMF Control Register (RCC_RCMFCR)

NAME	RCC_RCMFCR							
Offset	0x00000064							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-						PSC	
access	U-0						R/W-00	
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							EN
access	U-0							R/W-0

bit	name	functional description
31:18	-	Reserved, read as 0
17:16	RCMF_PSC	RCMFoutputprescaler 00: divided-by-1 01: divided-by-4 10: divided-by-8 11: divided-by-16
15:1	-	Reserved, read as 0
0	EN	RCMF enable register 0: Disable RCMF 1: Enable RCMF

11.11.23 RCMF Trim Register (RCC_RCMFTR)

NAME	RCC_RCMFTR							
Offset	0x00000068							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							

access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	TRIM						
access	U-0	R/W-100 0000						

bit	name	functional description
31:7	-	Reserved, read as 0
6:0	TRIM	RCMF frequency trim register, 7'h00 means the lowest frequency, 7'h7F means the highest frequency, the trimming range is +/-30% of the center frequency, the trimming step is 1%

11.11.24 Peripheral Operation Clock Control Register1 (RCC_OPCCR1)

NAME	RCC_OPCCR1							
Offset	0x0000006C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	EXTICKE	EXTICKS	LPUART1CKE	LPUART0CKE	LPUART1CKS		LPUART0CKS	
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-00		R/W-00	
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-			I2CCKE	-		I2CCKS	
access	U-0			R/W-0	U-0		R/W-00	
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	ATCKE	-					UART1CKE	UART0CKE
access	R/W-0	U-0					R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ATCKS		-		UART1CKS		UART0CKS	
access	R/W-00		U-0		R/W-00		R/W-00	

bit	name	functional description
31	EXTICKE	External interrupt operation clock enable, write 1 enable
30	EXTICKS	External interrupt sampling clock select 1: External pin interrupt uses LSCLK sampling 0: External pin interrupt uses HCLK sampling *It is recommended to set up when all EXTI interrupts are turned off, and then enable EXTIinterrupts after the setting is completed
29	LPUART1CKE	LPUART1 operation clock enable, write 1 enable
28	LPUART0CKE	LPUART0 operation clock enable, write 1 enable
27:26	LPUART1CKS	LPUART1 operation clock select 00: LSCLK 01: Divided RCHF 10: Divided RCMF 11: RFU
25:24	LPUART0CKS	LPUART0 operation clock select

bit	name	functional description
		00: LSCLK 01: Divided RCHF 10: Divided RCMF 11: RFU
23:22	-	Reserved, read as 0
21	I2CSMBCKE	I2C_SMBUS operation clock enable
20	I2CCKE	I2C operation clock enable
19:18	I2CSMBCKS	I2C_SMBUS operation clock select 00：APBCLK 01：RCHF 10：SYSCLK 11：RCMF_PSC
17:16	I2CCKS	I2C operation clock select 00: APBCLK 01: RCHF 10: SYSCLK 11: RCMF_PSC
15	ATCKE	ATIM operation clock enable register, write 1 enable
14:12	-	Reserved, read as 0
11	LINCKE	LIN operation clock enable, write 1 enable
10	UART2CKE	UART2 operation clock enable, write 1 enable
9	UART1CKE	UART1 operation clock enable, write 1 enable
8	UART0CKE	UART0 operation clock enable, write 1 enable
7:6	ATCKS	ATIM operation clock source select register 00: APBCLK2 01:USB PHY BCK 120M 10:APBCLK2 11:PLL double frequency
5:4	UART2CKS	UART2 operation clock select 00：APBCLK 01：RCHF 10：SYSCLK 11：RCMF_PSC
3:2	UART1CKS	UART1 operation clock select 00:APBCLK 01:RCHF 10:SYSCLK 11:RCMF_PSC
1:0	UART0CKS	UART0 operation clock select 00:APBCLK 01:RCHF 10:SYSCLK 11:RCMF_PSC

11.11.25 Peripheral Operation Clock Control Register2 (RCC_OPCCR2)

NAME	RCC_OPCCR2							
Offset	0x00000070							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-	RNGPRSC			-	ADCPRSC		
access	U-0	R/W-000			U-0	R/W-000		
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	USBREFCKE	FLASHCKE	RNGCKE	ADCCKE	USBREFCKS		ADCCKS	
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-00		R/W-00	
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			LPTCKE	-		LPTCKS	
access	U-0			R/W-0	U-0		R/W-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-			BTCKE	-		BTCKS	
access	U-0			R/W-0	U-0		R/W-00	

bit	name	functional description
31	-	Reserved, read as 0
30:28	RNGPRSC	RNG operation clock prescaler 000: divided-by-1 001: divided-by-2 010: divided-by-4 011: divided-by-8 100: divided-by-16 101: divided-by-32 110, 111:RFU
27	-	Reserved, read as 0
26:24	ADCPRSC	ADC operation clock prescaler 000: divided-by-1 001: divided-by-2 010: divided-by-4 011: divided-by-8 100: divided-by-16 101: divided-by-32 110/111:RFU
23	USBREFCKE	USB reference clock enable 0: Disable USB reference clock output 1: Enable USBreference clock output
22	FLASHCKE	Flash erase/program clock enable, write 1 enable
21	RNGCKE	RNG operation clock enable, write 1 enable
20	ADCCKE	ADC operation clock enable, write 1 enable
19:18	USBREFCKS	USB reference clock source select 00/11: XTLF (32768Hz) 01: XTHF (12MHz) 10: Divided RCHF
17:16	ADCCKS	ADC operation clock select 00: RCMF_PSC 01: RCHF 10: XTHF 11: PLL
15:13	-	Reserved, read as 0

bit	name	functional description
12	LPTCKE	LPTIM operation clock enable, write 1 enable
11:10	-	Reserved, read as 0
9:8	LPTCKS	LPTIM operation clock select 00: APBCLK1 01: LSCLK 10: LPOSC 11: RCMF_PSC
7:5	-	Reserved, read as 0
4	BTCKE	BSTIM operation clock enable, write 1 enable
3:2	-	Reserved, read as 0
1:0	BTCKS	BSTIM operation clock source select 00: APBCLK2 01: LSCLK 10: LPOSC 11: RCMF_PSC

12 Oscillation Fail Detection (FDET)

12.1 Low frequency fail detection

FM33LE0xxA has an on-chip low-frequency crystal oscillation stop detection circuit. After enabling it, it can continuously detect the XTLF output. When XTLF is found to stop oscillation, an alarm interrupt will be generated. The software can decide whether to automatically switch LSCLK to LPOSC through the LSCATS register.

When LSCATS=1, FDET detects that XTLF stops oscillation, and the hardware will automatically enable LPOSC and switch LSCLK to LPOSC output; when LSCATS=0, stop oscillation detection will only generate an alarm interrupt, and will not automatically switch the clock.

When XTLF is stopped, the software can also switch XTLF by setting LSCTS.

The vibration stop detection circuit is always opened or closed at the same time as XTLF. It cannot be closed separately. Once XTLF is enabled, the vibration stop detection circuit will automatically open; when XTLF is closed, the vibration stop detection will also be automatically closed to avoid false triggering of the vibration stop alarm.

12.2 High frequency fail detection

FM33LE0xxA has an on-chip high-frequency crystal stop vibration detection circuit, which can be enabled or closed together with the XTHF circuit.

After the oscillation stop detection is enabled, the XTHF output can be continuously detected. When XTHF is found to stop oscillating, an alarm interrupt will be generated, and an advanced timer brake signal will be generated; if XTHF is being used directly or indirectly as the system working clock (directly refers to the SYSCLK selection XTHF, indirectly means that SYSCLK is selected as PLL and PLL uses XTHF as input reference clock), then the stop signal will automatically enable RCHF and switch SYSCLK to RCHF to avoid accidental stop of high-frequency crystals and system crashes.

The vibration stop detection circuit is always opened or closed at the same time as XTHF. It cannot be closed separately. Once XTHF is enabled, the vibration stop detection circuit will automatically open; when XTHF is closed, the vibration stop detection will also be automatically turned off to avoid false triggering of the vibration stop alarm.

The HFDET vibration stop detection threshold is about 200KHz, that is, when the XTHF clock frequency is lower than 200KHz, the vibration stop alarm is triggered; the HFDET working current is about 1.2uA.

12.3 Register

offset	Name	Symbol
FDET(Base address:0x4001A838)		
0x00000000	XTLF Oscillation Fail Detection Interrupt Enable Register	FDET_IER
0x00000004	XTLF Oscillation Fail Detection Interrupt Status Register	FDET_ISR

12.3.1 Fail detection interrupt enable register (FDET_IER)

NAME	FDET_IER							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						HFDET_IE	LFDET_IE
access	U-0						R/W-0	R/W-0

bit	name	functional description
31:2	-	RFU: **Reserved, read as 0**
1	HFDET_IE	XTHF fail detect interrupt enable, 1 effective
0	LFDET_IE	XTLF fail detect interrupt enable, 1 effective

12.3.2 Fail detection interrupt flag register（FDET_ISR）

NAME	FDET_ISR							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16

name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						HFDETO	LFDETO
access	U-0						R-0	R-0
NAME	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Offset	-				HFRDY	LFRDY	HFDETIF	LFDETIF
bit	U-0				R-0	R-0	R/W-0	R/W-0

Bit	Name	Description
31:10	-	RFU: **Reserved, read as 0**
9	HFDETO	XTHF fail detect output 1: XTHF oscillating 0: XTHF stopped
8	LFDETO	XTLF fail detect output 1: XTLF oscillating 0: XTLF stopped
7:4	-	RFU: **Reserved, read as 0**
3	HFRDY	XTHF stable Flag, Hardware set, only read Close XTHF or automatically clear when XTHF stops vibration
2	LFRDY	XTHF stable Flag, Hardware set, only read Close XTHF or automatically clear when XTHF stops vibration
1	HFDETIF	XTHF fail detect interrupt flag, write 1 to clear; This register can be cleared only when HFDETO is not 0
0	LFDETIF	XTLF fail detect interrupt flag, write 1 to clear; This register can be cleared only when LFDETO is not 0

Note: *when clearing the flag, clearing bit 0 is to clear LFDETIF, and clearing bit 1 is to clear HFDETIF. Different here from FM33LE0xx*

13 Supply Voltage Detection (SVD)

13.1 Introduction

The supply voltage detection circuit I mainly used to detect the supply of the external mains power supply, to detect the under-voltage or recovery of the external mains power supply in time and to give an interrupt signal. The supply voltage detection circuit can be switched off or periodically enabled to save power.

Features:

- Detect mains power interrupt generated when voltage is below or above set threshold
- Under voltage detection range 1.8v~4.8v, 15 level programmable threshold steps. Step interval 0.214V.
- Voltage detection hysteresis window of 1.0v
- SVD can be turned off or operated intermittently
- Support 1 external channel direct input for comparison with internal reference voltage source
- External channel supports 100mV window by setting reference voltage

13.2 Block Diagram

The below figure is block diagram of supply voltage detection.

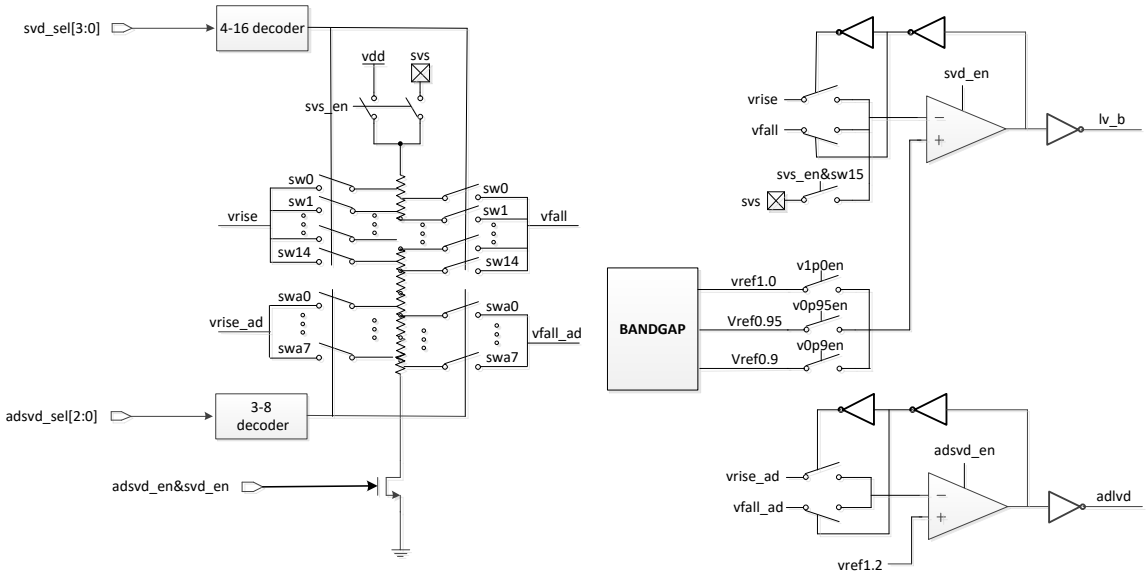


Figure 13-1 Low voltage detection circuit diagram

The SVD has 15 internal channels and one external channel. The internal channel is used for chip power detection and the external channel is used to compare the external input signal with the internal reference voltage.

SVD operate timing sequence diagram:

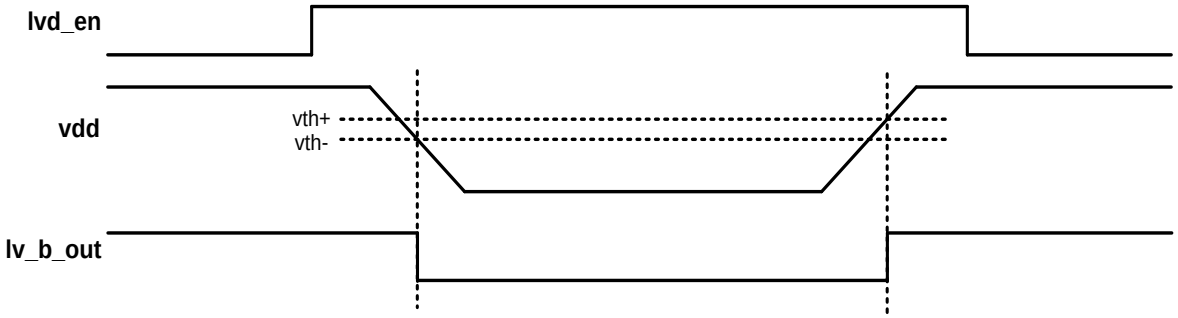


Figure 13-2 Low voltage detection circuit diagram

13.3 Pin definition

The SVD module can directly detect the chip power supply (VDD), or it can detect the external voltage signal through the SVS pin (PA15).

When detecting SVS input, you need to configure the FCR register of PA15 to 11 (analog function)

13.4 Functional description

The supply voltage detection circuit can be used to detect the supply voltage and external voltage.

The power supply voltage generates 15 detection levels through voltage divider resistors, with a detection range of 1.8V to 4.8V and a step of 0.214V. In addition, it supports one external input with 16 detection level. By 1 out of 6 MUX sent to comparator and compared with the internal reference voltage. According to the low voltage warning threshold configuration, if the level to be detected is lower than the reference voltage, the output voltage will jump, and an undervoltage interrupt will be generated to notify the MCU to handle the event in time; When VDD recovers above the threshold (about a 0.1V hysteresis window), undervoltage recovery interruption will occur.

The source detected circuit can be enable or disable by software configure. In order to save power consumption, the enabling time can be divided into two modes: normal enabling mode and intermittent working mode. When intermittent working, time interval is opened by setting the sector DSEF register.

In the long-term enable condition, SVD has a 0.1V hysteresis window from undervoltage to overvoltage, while there is no hysteresis under intermitter enabling; For internal channel, the window problem can be solved by software cooperation, that is, setting a higher overvoltage threshold after undervoltage interruption. However, for SVS channel, it needs special design. The digital circuit locks the decision result of the last intermitter window as the basis of threshold selection in this intermitter window, so as to realize the selection of SVS falling threshold and rising threshold. Accordingly, BG needs to output three reference voltages of 0.9V, 0.95v and 1.0V.

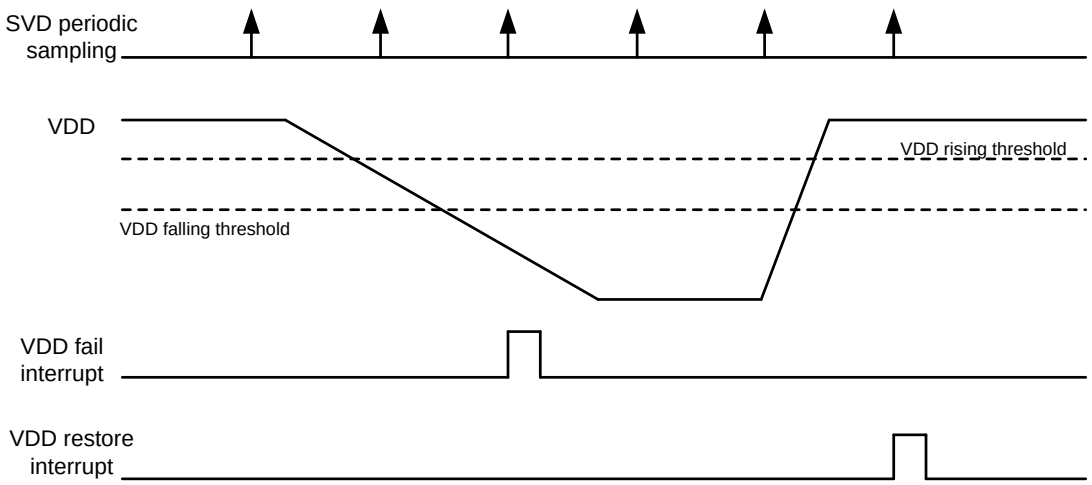


Figure 13-3 Intermittent operation mode of power detection circuit

When intermittent operation, after software enabled the gap of SVD, SVD does not necessarily work immediately, but waits for the next open window. However, in long-term enable condition, after the software starts SVD, SVD will start working after one or two LSCLK clock synchronization cycles. It takes about 100us from SVD on to the stable establishment of the output. The software needs to pay attention when reading the SVD output.

If all clocks are turned off after the chip enters sleep mode and you want to use SVD, you need to set SVD to normally enable and turn off the digital filtering function before sleep.

Operational mode description:

- In the long-term enable/internal channel mode, the detection threshold has a window, the failing threshold window are 0.1V, and the failing threshold is detected when it is not enabled.
- In the intermittent enable / internal channel mode, the falling threshold is detected every time the intermittent enable is started (i.e., when it is not enabled to enable), so there is no threshold window and software cooperation is required. That is, when undervoltage is detected in the previous intermittent enable, the software will adjust the threshold gear by one gear; When the non-undervoltage is detected by the previous intermittent enable, the software restores the threshold gear.
- In the long-term enable/external channel mode, the input reference voltage is three-level input, 0.9V, 0.95V, 1.0V, and there is no window for the detection threshold. Software cooperation is required, that is, when an undervoltage is detected, the software Increase the threshold gear by one gear; when non-undervoltage is detected, the software will restore the gear.
- In the intermittent enable/external channel mode, the input reference voltage is three-level input, 0.9V, 0.95V, 1.0V, and there is no window for the detection threshold. Software cooperation is required, that is, when an undervoltage is detected, the software Increase the threshold gear by one gear; when non-undervoltage is detected, the software will restore the gear.

13.5 Intermittent enable mode

In sleep mode, the average power consumption of SVD can be reduced by intermittent enable. In DeepSleep mode, AVREF is closed by default. At this time, AVREF is also started when SVD is Intermittently started. The digital circuit waits for AVREF to be established before sampling SVD output.

In the SVD startup window, the total power consumption of SVD plus AVREF is less than 3uA. If the on interval is 62.5ms, the average current is about 70nA; if the on interval is 1s,the average current is less than 5uA.

13.6 External voltage detection

In addition to detecting the power supply of the chip, the SVD can also detect the power down or power on of the external voltage signal.

External voltage detection is achieved via the SVS pin (PA15). The SVS input can be divided by an external resistor or divided by an internal resistor and then input to the comparator for detection. When the external voltage is higher than the chip power supply, it is mandatory to use the external resistor divider to get an SVS input lower than the chip power supply; when the external voltage is lower or equal to the chip power supply, it can be directly connected to the SVS pin and divided by internal resistor divider. The PA15 pin needs to be set to the analog function before using SVS.

The following diagram shows the external supply detection with external resistor divider.

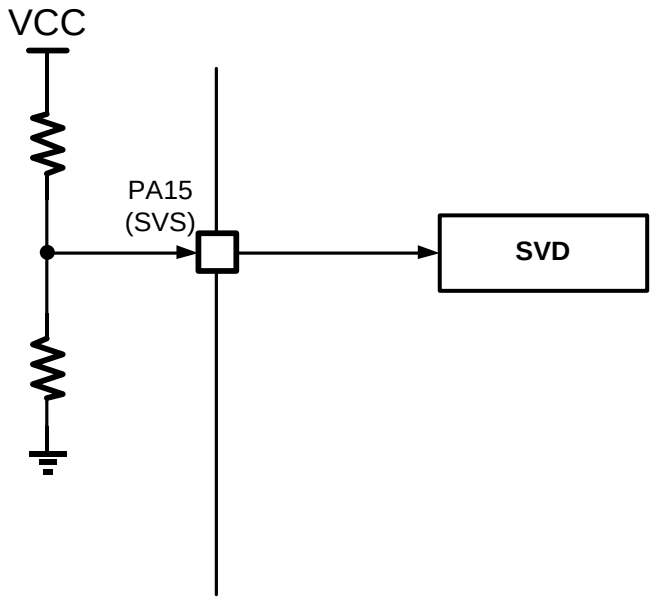


Figure 13-4 SVS detection using external resistor divider

The following diagram shows the external supply detection with internal resistor divider.

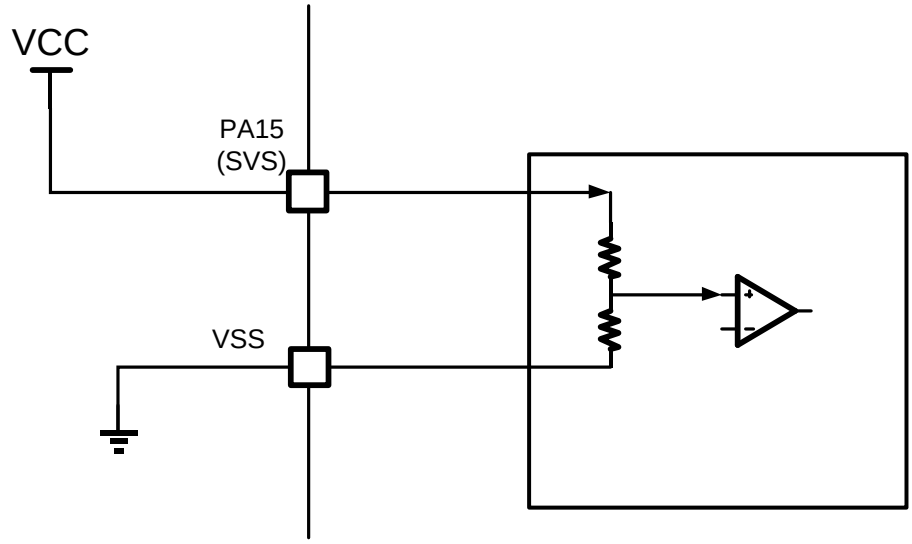


Figure 13-5 SVS detection using internal resistor divider

The registers are configured in the following way.

SVSEN	SVDLVL	Description
0	X	The external power supply detection channel is closed, and only the internal power supply voltage is detected
1	1111	External voltage input without internal divider, direct input to comparator for comparison with internal reference voltage
	0000~1110	The external voltage input is first divided by internal resistor and then fed to the comparator to compare with the internal reference. Refer to the following chapters for the gear position after pressure division

13.7 Detection threshold

The SVSEN and SVDLVL registers allow user to select the voltage detection object and the detection threshold.

Internal power supply detection：SVSEN = 0，{VREF1P0EN, VREF0P95EN, VREF0P9EN} = 100，baseline 1.0V

SVDLVL	Falling Threshold （V）	Rising Threshold （V）
0000	1.799	1.912
0001	2.013	2.128

0010	2.227	2.343
0011	2.442	2.558
0100	2.656	2.774
0101	2.871	2.989
0110	3.085	3.204
0111	3.300	3.420
1000	3.515	3.635
1001	3.73	3.851
1010	3.945	4.064
1011	4.16	4.284
1100	4.376	4.500
1101	4.592	4.716
1110	4.807	4.932
1111	N/A	N/A

Internal power supply detection：SVSEN = 0，{VREF1P0EN, VREF0P95EN, VREF0P9EN} = 010，baseline 0.95V

SVDLVL	Falling Threshold（V）	Rising Threshold（V）
0000	1.704	1.812
0001	1.907	2.016
0010	2.110	2.220
0011	2.313	2.424
0100	2.516	2.628
0101	2.719	2.832
0110	2.923	3.036
0111	3.126	3.240
1000	3.329	3.444
1001	3.533	3.649
1010	3.737	3.853
1011	3.941	4.058
1100	4.146	4.263
1101	4.350	4.468
1110	4.554	4.672
1111	N/A	N/A

Internal power supply detection：SVSEN = 0，{VREF1P0EN, VREF0P95EN, VREF0P9EN} = 001，baseline 0.9V

SVDLVL	Falling Threshold（V）	Rising Threshold（V）

0000	1.609	1.712
0001	1.801	1.904
0010	1.992	2.097
0011	2.184	2.289
0100	2.376	2.482
0101	2.568	2.675
0110	2.760	2.867
0111	2.952	3.060
1000	3.144	3.253
1001	3.336	3.446
1010	3.529	3.639
1011	3.722	3.833
1100	3.915	4.026
1101	4.108	4.220
1110	4.300	4.413
1111	N/A	N/A

External power detection：SVSEN =1，{VREF1P0EN, VREF0P95EN, VREF0P9EN} = 100，baseline 1.0V

SVDLVL	Falling Threshold（V）	Rising Threshold（V）
0000	1.799	1.912
0001	2.013	2.128
0010	2.227	2.343
0011	2.442	2.558
0100	2.656	2.774
0101	2.871	2.989
0110	3.085	3.204
0111	3.300	3.420
1000	3.515	3.635
1001	3.73	3.851
1010	3.945	4.064
1011	4.16	4.284
1100	4.376	4.500
1101	4.592	4.716
1110	4.807	4.932
1111	1.0	1.0

External power detection：SVSEN =1，{VREF1P0EN, VREF0P95EN, VREF0P9EN} = 010，baseline 0.95V

SVDLVL	Falling Threshold （V）	Rising Threshold （V）
0000	1.704	1.812
0001	1.907	2.016
0010	2.110	2.220
0011	2.313	2.424
0100	2.516	2.628
0101	2.719	2.832
0110	2.923	3.036
0111	3.126	3.240
1000	3.329	3.444
1001	3.533	3.649
1010	3.737	3.853
1011	3.941	4.058
1100	4.146	4.263
1101	4.350	4.468
1110	4.554	4.672
1111	0.95	0.95

External power detection：SVSEN =1，{VREF1P0EN, VREF0P95EN, VREF0P9EN} = 010，baseline 0.9V

SVDLVL	Falling Threshold （V）	Rising Threshold （V）
0000	1.609	1.712
0001	1.801	1.904
0010	1.992	2.097
0011	2.184	2.289
0100	2.376	2.482
0101	2.568	2.675
0110	2.760	2.867
0111	2.952	3.060
1000	3.144	3.253
1001	3.336	3.446
1010	3.529	3.639
1011	3.722	3.833
1100	3.915	4.026
1101	4.108	4.220
1110	4.300	4.413
1111	0.9	0.9

13.8 Register

offset	Name	Symbol
SVD(Base address:0x4001A824)		
0x00000000	SVD Config Register	SVD_CFGR
0x00000004	SVD Control Register）	SVD_CR
0x00000008	SVD Interrupt Enable Register	SVD_IER
0x0000000C	SVD Interrupt Status Register	SVD_ISR
0x00000010	SVD reference Voltage Select Register	SVD_VSR

13.8.1 SVD Config Register (SVD_CFGR)

NAME	SVD_CFGR							
offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				-			-
access	U-0				U-0			U-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SVDLVL				DFEN	SVDMOD	SVDITVL	
access	R/W-0000				R/W-1	R/W-0	R/W-00	

Bit	Name	Functional Description
31:12	--	**RFU：Reserved, read as 0**
11:9	--	**RFU：Reserved, read as 0**
8	--	**RFU：Reserved, read as 0**
7:4	SVDLVL	SVD alarm threshold setting, refer to ‘Power detection threshold’ chapter for gear definition
3	DFEN	Digital filter enable (must be set to 1 when SVDMODE=1) (digital filter enable) 1: Start digital filtering of SVD output 0: Turn off the digital filter of SVD output
2	SVDMOD	Select SVD working mode, and set svden after configuration mode to start SVD 1: Intermittent enable mode

Bit	Name	Functional Description
		0: Always enabled mode Note: digital filtering must be enabled in intermittent enable mode
1:0	SVDITVL	SVD Interval 00：62.5ms 01：256ms 10：1s 11：4s

13.8.2 SVD Control Register (SVD_CR)

NAME	SVD_CR							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						SVSCFG	SVDTE
access	U-0						W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						SVSEN	SVDEN
access	U-0						R/W-0	R/W-0

Bit	Name	Functional Description
31:10	--	**RFU：Reserved, read as 0**
9	SVSCFG	SVS compatibility configuration, write only, not readable 0: compatible with fm33le0, external resistance string and internal resistance string are connected in parallel 1: The internal resistance string does not affect the voltage division of the external resistance
8	SVDTE	SVD test enable, avoid writing 1
7:2	--	**RFU：Reserved, read as 0**
1	SVSEN	S SVS external channel control 0: SVS channel disabled 1: SVS channel enabled When SVS_EN = 1, the SVDLVL input can be set to be divided by internal resistors according to the SVDLVL register; if SVDLVL = 1111, then the SVS input is not divided, if SVDLVL ! = 1111, then

Bit	Name	Functional Description
		the SVS input is divided by the internal resistor.
0	SVDEN	SVD enable 1: Enable SVD 0: disable SVD

13.8.3 SVD Interrupt Enable Register (SVD_IER)

NAME	SVD_IER							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						PFIE	PRIE
access	U-0						R/W-0	R/W-0

Bit	Name	Functional Description
31:2	--	**RFU：Reserved, read as 0**
1	PFIE	Power Fall interrupt enable 1: Allow power drop interruption 0: Disable interrupt
0	PRIE	Power Rise interrupt enable 1: Allow power drop interruption 0: Disable interrupt

13.8.4 SVD State and Flag Register (SVD_ISR)

NAME	SVD_ISR							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16

name	-							-
access	U-0							U-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							SVDO
access	U-0							R
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SVDR	-					PFF	PRF
access	R	U-0					R/W-0	R/W-0

Bit	Name	Functional Description
31:17	--	**RFU：Reserved, read as 0**
16	--	**RFU：Reserved, read as 0**
15:9	--	**RFU：Reserved, read as 0**
8	SVDO	SVD power detection output 1: The power supply voltage is higher than the current threshold of SVD 0: The power supply voltage is lower than the current threshold of SVD
7	SVDR	SVD output latch signal, SVD state latched by digital circuit
6:2	--	**RFU：Reserved, read as 0**
1	PFF	Power fall flag, write 1 to clear
0	PRF	Power rise flag, write1to clear

13.8.5 SVD Reference Voltage Select Register (SVD_VSR)

NAME	SVD_VSR							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit0	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					V1P0EN	V0P95EN	V0P9EN
access	U-0					R/W-1	R/W-0	R/W-0

Bit	Name	Functional Description
31:3	--	**RFU：Reserved, read as 0**

Bit	Name	Functional Description
2	V1P0EN	1.0V reference input enable signal (1.0V reference enable) 1: Enable 1.0V reference input 0: turn off 1.0V reference input
1	V0P95EN	0.95V reference input enable signal (0.95V reference enable) 1: Enable 0.95V reference input 0: turn off 0.95V reference input
0	V0P9EN	0.9V reference input enable signal 1：Enable 0.9V reference input 0：Disable 0.9V reference input 0.9V reference input enable signal (0.9V reference enable) 1: Enable 0.9V reference input 0: Disable 0.9V reference input

14 AES Algorithm Module (AES)

14.1 Function description

AES main features

- Support decryption key extension
- Support for 128-bit/192-bit/256-bit key lengths
- Support Electronic Code Book(ECB), Cipher Block Chaining (CBC), Counter mode (CTR) and Galois Counter Mode(GCM)
- DMA support for automatic data transfer
- Support multiplication under Galois Field (2^128); Support Galois Message Authentication Code mode (GMAC)

14.2 Operation Mode

AES module has four operating modes, which are set by MODE[1:0] registers.

- **MODE=1**: Encryption using the key stored in the AES_KEYRx register.
- **MODE=2**: Key extension, which overwrites the encryption key initially stored in the AES_KEYRx register with the key calculation result stored in the internal register after the key extension completed.
- **MODE=3**: Decryption using the decryption key stored in the AES_KEYRx register.
- **MODE=4**: Key extension and decryption with the encryption key stored in the AES_KEYRx register. (not used in CTR mode)

Users should config the operating mode by MODE[1:0] register. Note that the MODE[1:0] can only be written before the AES module is enabled (EN=0). The KEY register should also be configured before AES enabled. After that, users should configure the data stream processing mode register CHMOD[1:0]. In CBC/CTR/GCM mode, the initial vector register (IVR) also needs to be configured.

Then user can enable AES module(EN=1). In Mode 1/Mode 3/Mode 4, the AES module will waits for the software to write input data to the AES_DINR register, and AES calculation will start after writing for 4 times (128bits). In Mode 2, the key extension operation is performed immediately after the AES is enabled.

The CCF Flag will be set after the calculation is finished. If CCFIE=1, an interrupt will be generated. Then the software can acquire the 128-bits result by reading the AES_DOUTR register 4 times.

AES also supports DMA mode. By configuring DMAOUTEN=1 and DMAINEN=1, AES can work with DMA to process data continuously without CPU's intervention.

The error flags RDERR and WRERR are set on an incorrect read or write operation, and if ERRIE is

enabled, a corresponding error interrupt will be generated. AES will continue to work normally after an error occurs.

The AES module can be reset at any time by setting EN register.

14.3 AES data stream processing modes

AES module has 4 data stream processing modes: ECB, CBC, CTR, GCM.

14.3.1 ECB mode

ECB is the default operating mode. IV registers are not used and each block is computed separately for encryption and decryption. The ECB mode encryption and decryption flow can be described by Figure 14-1 and Figure 14-2.

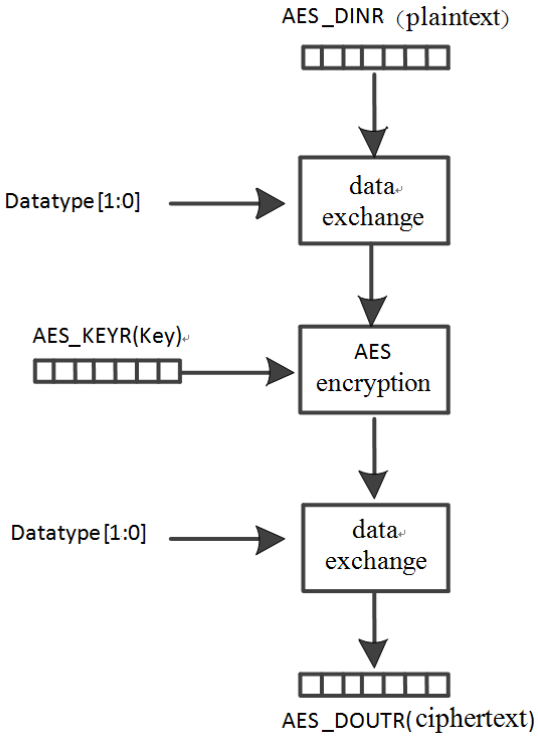


Figure 14-1 ECB mode encryption

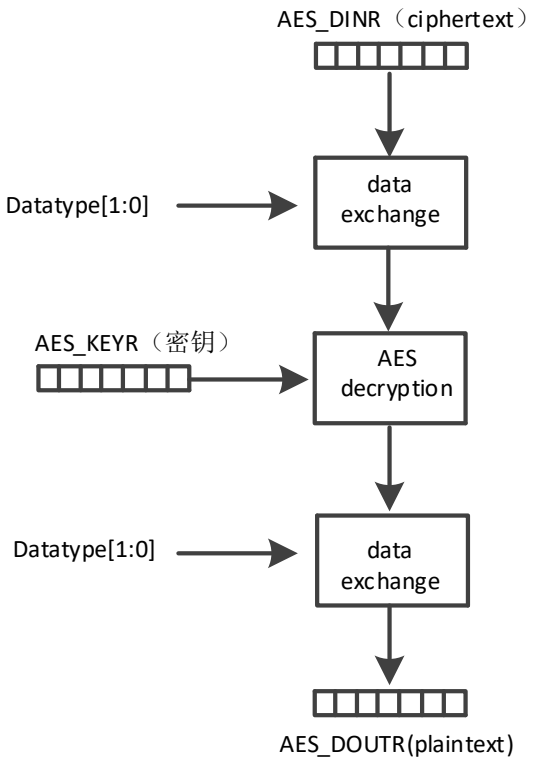


Figure 14-2 ECB mode decrypt

14.3.2 CBC mode

Each block of the plaintext data is used after XORed with the encryption result of the previous block as the input block. The first block requires an initial IVRx register value. The XOR operation is performed before encryption and after encryption when decrypting. The CBC mode encryption and decryption flow can be described by Figure14-3 and Figure14-4.

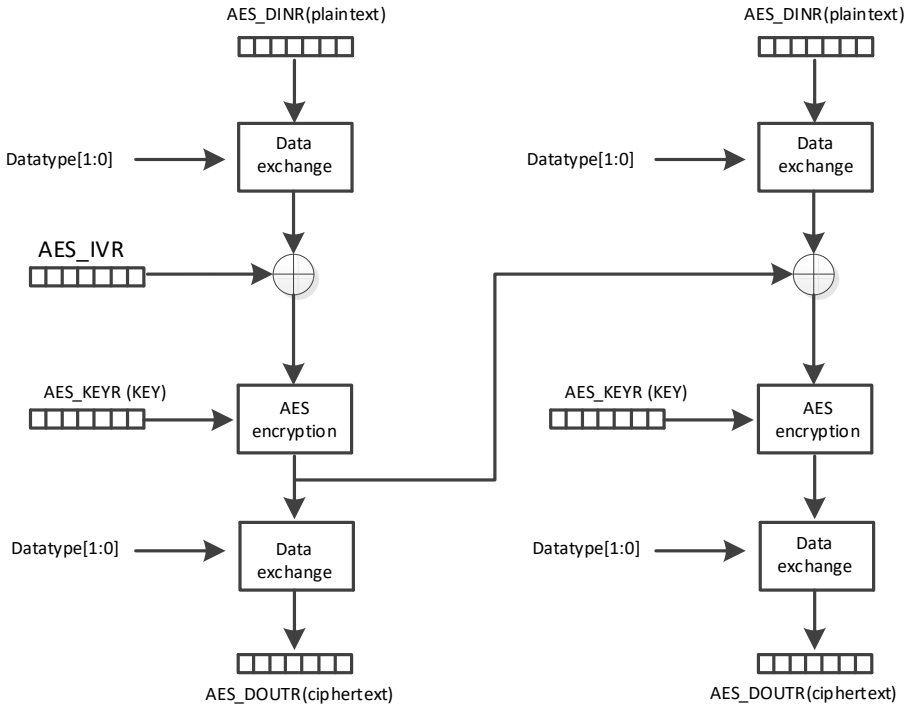


Figure 14-3 CBC mode encryption

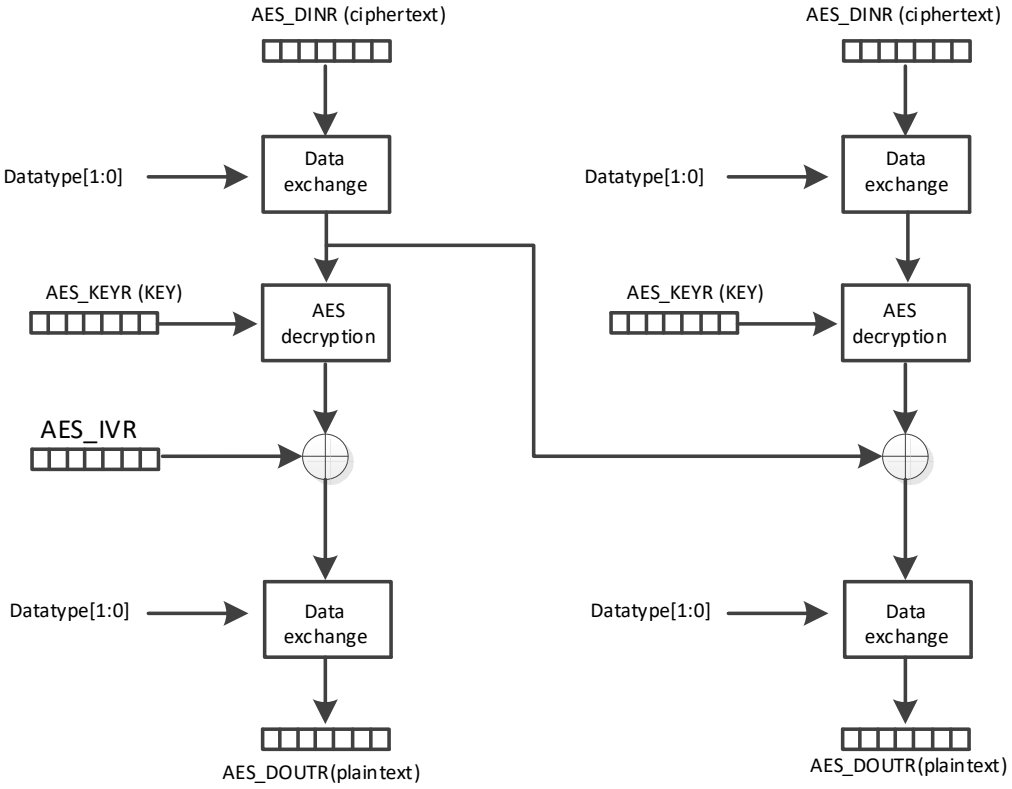


Figure 14-4 CBC mode decryption

Note：While AES is working, the AES_IVR register reads as 0x00000000.

14.3.3 Suspend mode

If a higher priority data needs to be processed, the current data processing can be suspended. Suspended data processing can be resumed in both encryption and decryption operation modes. This mode is available only when data flow is managed by CPU, not available in DMA mode.

Before suspending data processing, the user application must respect the following sequence:

1. read out the entire result of a block.
2. Suspend AES by clearing the EN bit to 0 in AES_CR, then save the value of AES_IVRx register, which needs to be written back to AES_IVRx when the data processing is resumed.

The suspend mode sequence can be described by Figure14-5.

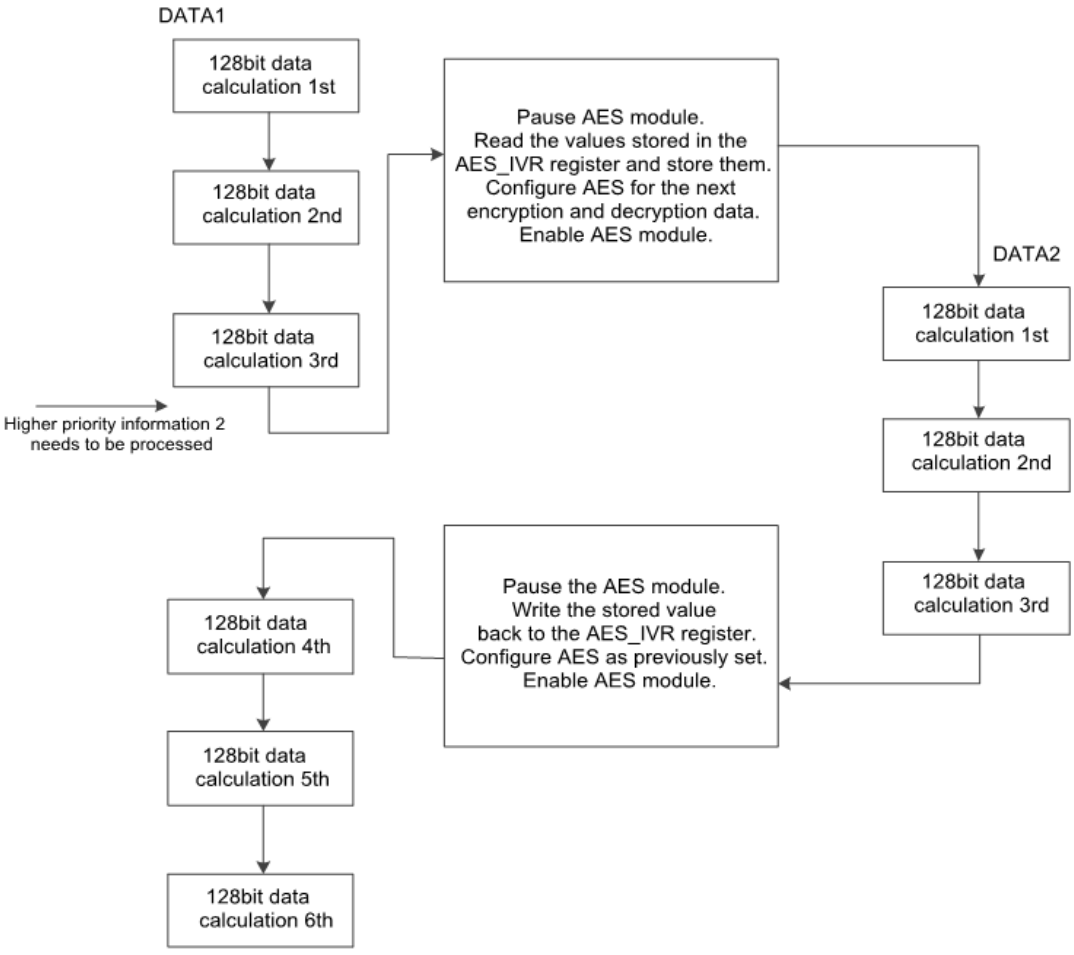


Figure 14-5 Suspend mode sequence

14.3.4 CTR mode

In CTR mode, a 32-bit counter and a random number are used as inputs to the encryption and decryption

module. The result is XORed with the plaintext data.

The CTR mode encryption and decryption flow can be described by Figure14-6 and Figure14-7.

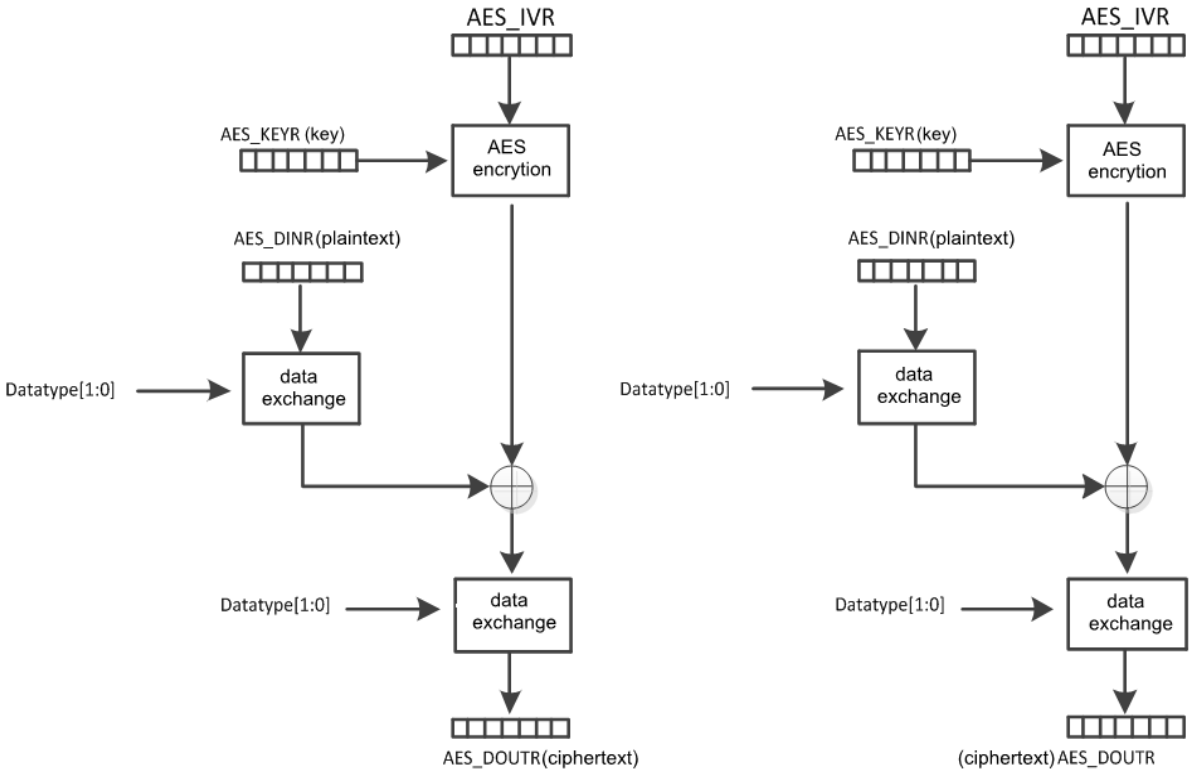


Figure 14-6 CTR mode encryption

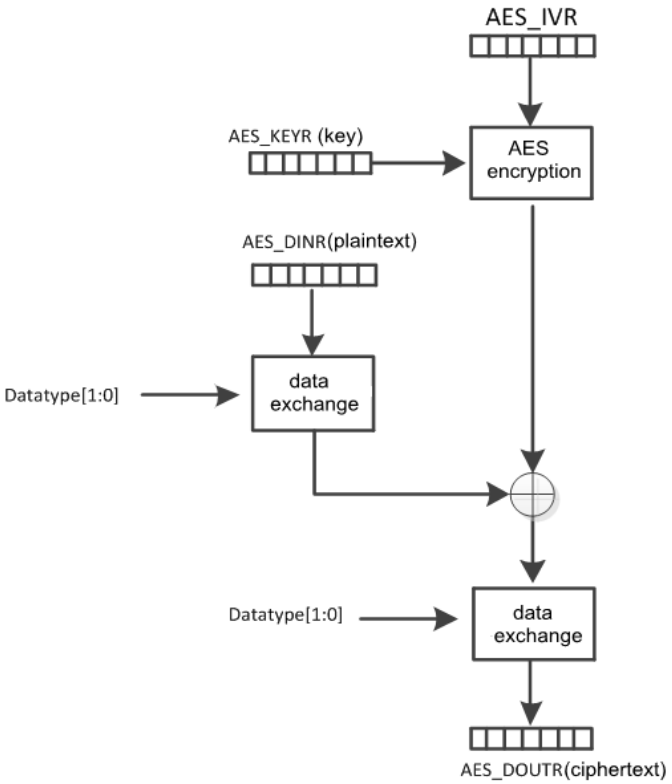


Figure 14-7 CTR mode decryption

Random number (nonce) and 32-bit counter value are stored in AES_IVRx registers as shown in Figure14-8:

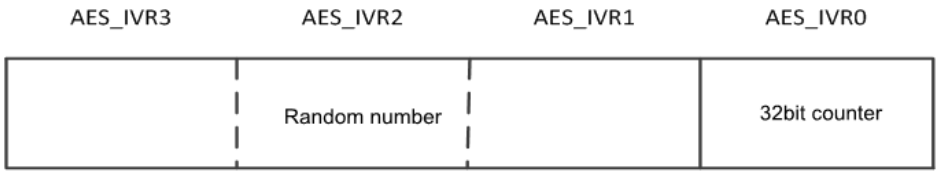


Figure 14-8 Storage format of 32-bit counter value and random number

The key extension and decryption mode under CTR mode is regardless.

14.3.5 Suspend mode under CTR mode

Similar to the suspend mode under CBC mode. Refer to suspend mode under CBC mode.

14.3.6 GCM mode

Refer to the documentation The Galois/Counter Mode of Operation (GCM)for details.

The GCM encryption is defined according to the following formula：

$$
\begin{aligned}
H &= E(K, 0^{128}) \\
Y_0 &= \begin{cases} IV \| 0^{31}1 & \text{if } \mathrm{len}(IV) = 96 \\ \mathrm{GHASH}(H, \{\}, IV) & \text{otherwise.} \end{cases} \\
Y_i &= \mathrm{incr}(Y_{i-1}) \text{ for } i = 1, \ldots, n \\
C_i &= P_i \oplus E(K, Y_i) \text{ for } i = 1, \ldots, n-1 \\
C_n^* &= P_n^* \oplus \mathrm{MSB}_u(E(K, Y_n)) \\
T &= \mathrm{MSB}_t(\mathrm{GHASH}(H, A, C) \oplus E(K, Y_0))
\end{aligned}
$$

where the GHASH function is defined as $\mathrm{GHASH}(H, A, C) = X_{m+n+1}$, and X is defined as

$$
X_i = \begin{cases}
0 & \text{for } i = 0 \\
(X_{i-1} \oplus A_i) \cdot H & \text{for } i = 1, \ldots, m-1 \\
(X_{m-1} \oplus (A_m^* \| 0^{128-v})) \cdot H & \text{for } i = m \\
(X_{i-1} \oplus C_i) \cdot H & \text{for } i = m+1, \ldots, m+n-1 \\
(X_{m+n-1} \oplus (C_m^* \| 0^{128-u})) \cdot H & \text{for } i = m+n \\
(X_{m+n} \oplus (\mathrm{len}(A) \| \mathrm{len}(C))) \cdot H & \text{for } i = m+n+1.
\end{cases}
$$

The encryption and decryption process of GCM mode can be described by Figure14-9 and Figure14-10.

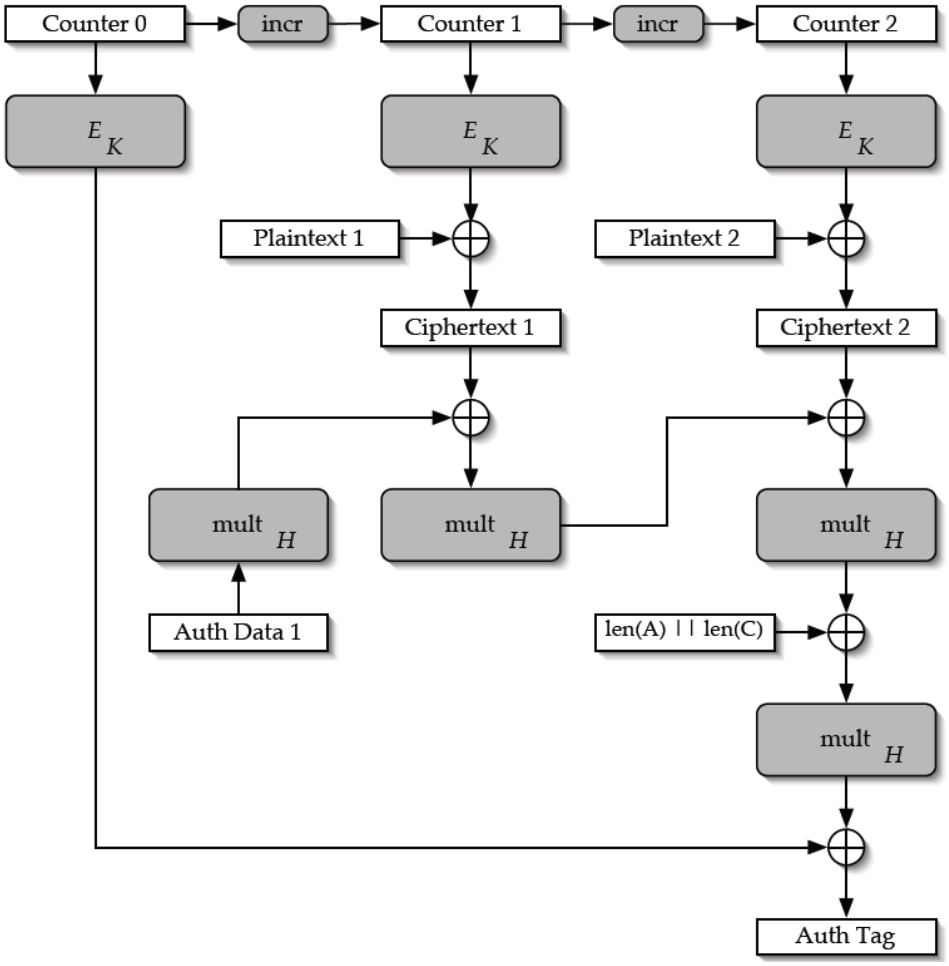


Figure 14-9 GCM mode encryption

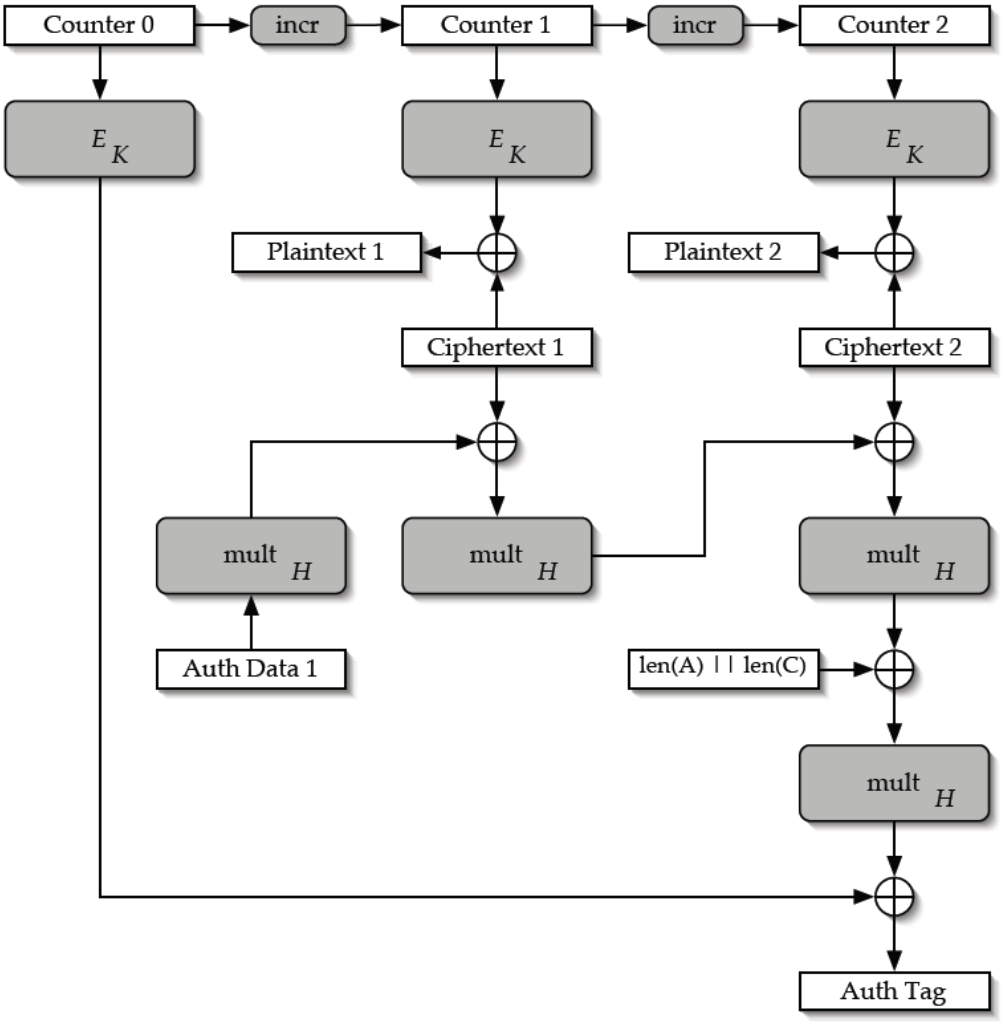


Figure 14-10 GCM mode decryption

The Ek in the diagram indicates the AES encryption module. multH module performs a multiplication under GF(2^128) domain**.** Incr indicates the counter plus one.

In GCM mode the hardware AES module and multH module are dispatched by software. The process of GCM mode encryption and decryption is same as CTR mode. The authentication process is implemented by the software using multH module.

14.3.7 MultH mode

Multiplication under GF(2^128) domains implemented using the following algorithm.

Algorithm 1 Multiplication in $GF(2^{128})$. Computes the value of $Z = X \cdot Y$, where X, Y and $Z \in GF(2^{128})$.

$Z \leftarrow 0, V \leftarrow X$
for $i = 0$ to 127 **do**
  if $Y_i = 1$ **then**
    $Z \leftarrow Z \oplus V$
  end if
  if $V_{127} = 0$ **then**
    $V \leftarrow \text{rightshift}(V)$
  else
    $V \leftarrow \text{rightshift}(V) \oplus R$
  end if
end for
return Z

The input and output registers of the MultH module are multiplexed with those of AES. The block diagram of MultH module can be described by Figure14-11.

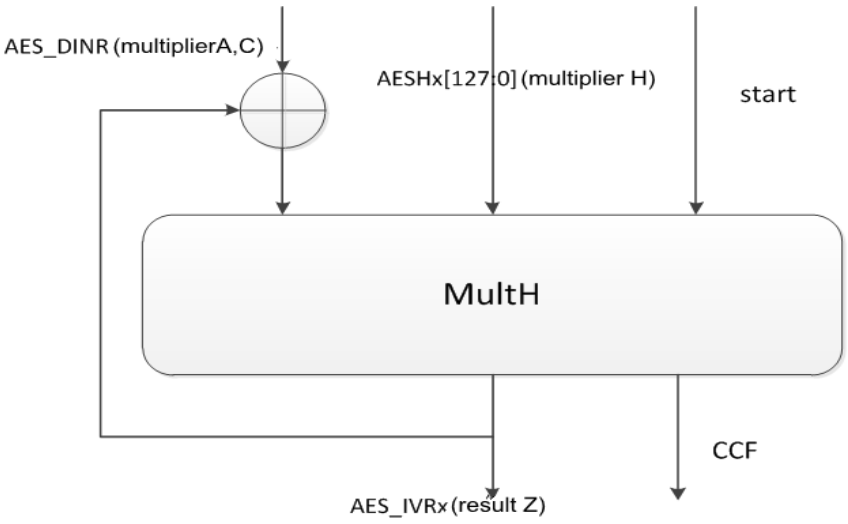


Figure 14-11 MultH module block diagram

The input registers of multH module are multiplexed with the AES input registers AES_DINR and the lower 128 bits of AES_KEYx. Input register multiplexed AES_IVR registers. When used, configure CHMOD[1:0] registers to MultH mode, then configure AES_KEYx and AES_IVR registers to input and output 128 bits each, enable EN, input data to AES_DINR, and wait for CCF to set up that the calculation is completed.

Note: Because the registers are multiplexed, calling multH will erase the AES registers. Therefore, if you want to perform AES calculation after using the MultH module for calculation, you need to rewrite the relevant registers.

14.3.8 Recommended GCM Process

The implementation of GCM mode requires hardware and software cooperation, and this document provides a recommended way to implement it.

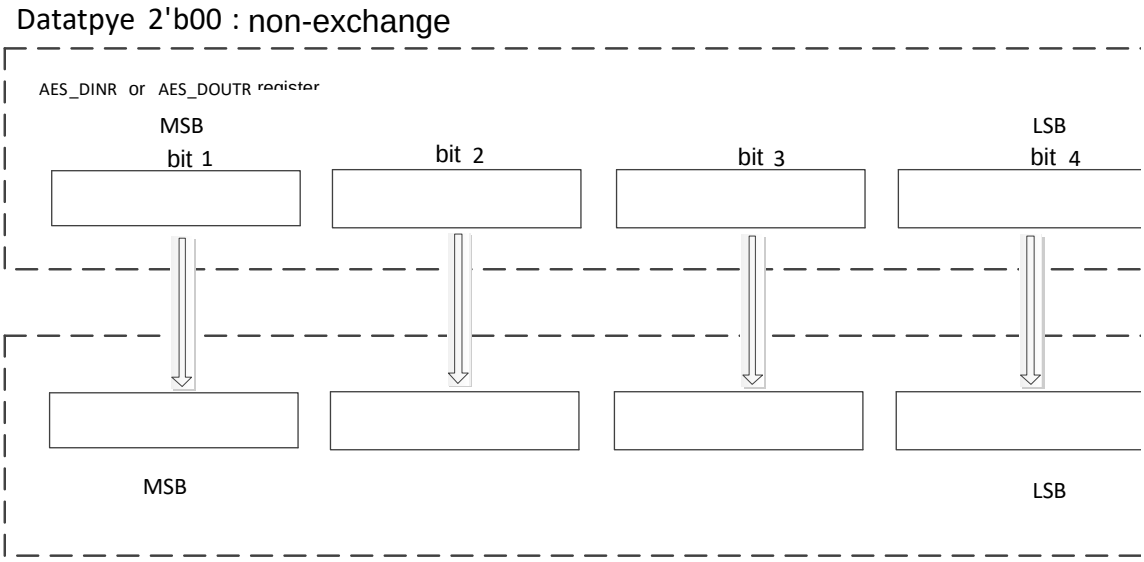
The encryption and decryption process of GCM mode is the same as CTR mode. Only MultH module is used for authentication process.

The recommended GCM Process is described as following:

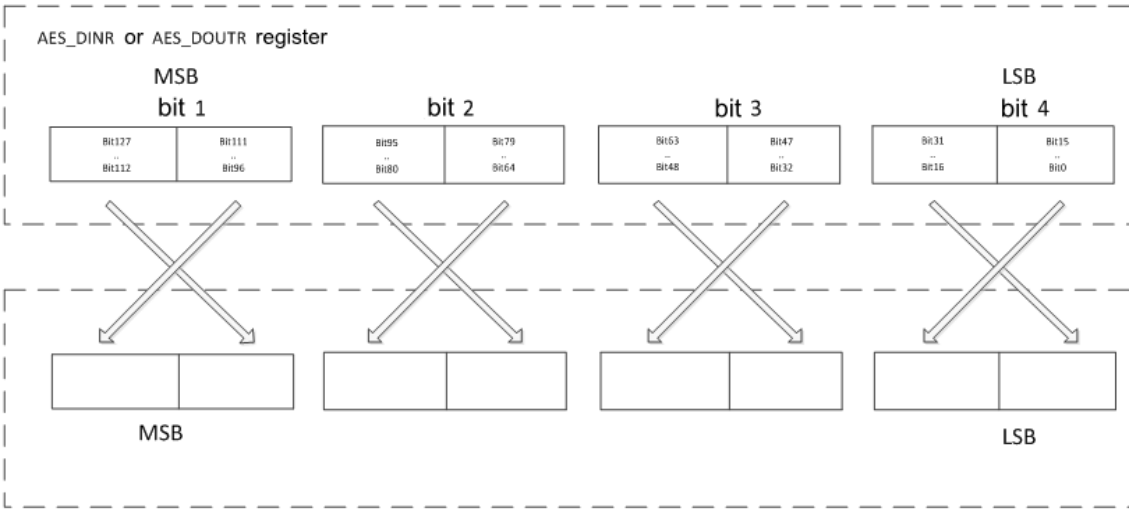
- The AES module is invoked to calculate and store H.
- The AES module is invoked to calculate and store E(K, Y0).
- Use CTR mode to start AES encryption and decryption of continuous data. Initial value of IV register is Y1.
- Perform Continuous calculation of GHASH using multH module
- The value of tag can be calculated by the XOR result of the final GHASH result and E(K, Y0).

14.4 Data type

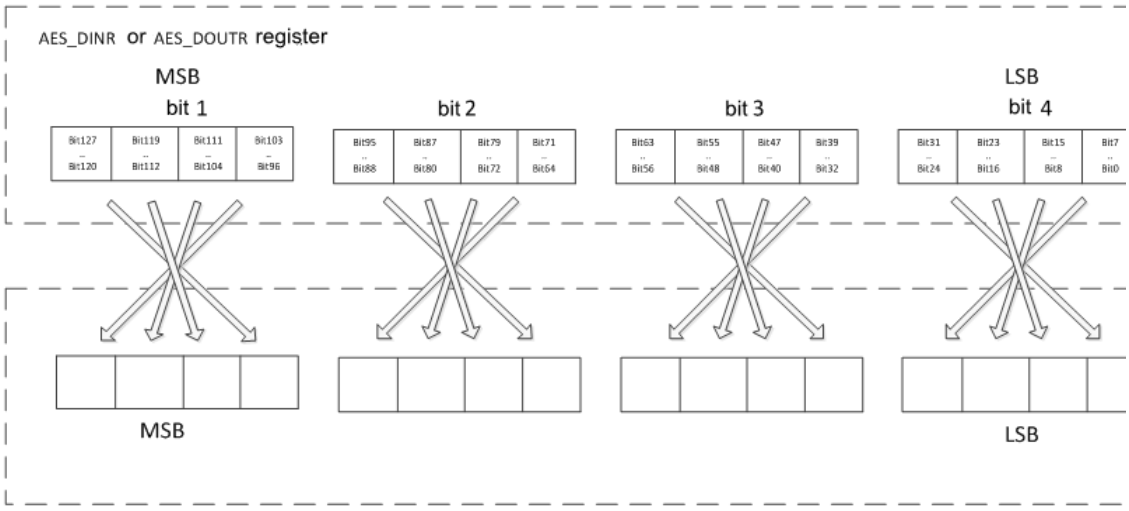
AES reads and writes 32 bits of data at a time, and each 32 bits can be exchanged in a different order according to the setting of the DATATYPE[1:0] register as shown in Figure14-12.



Datatpye 2'b01 : half-word exchange



Datatpye 2'b10 : byte exchange



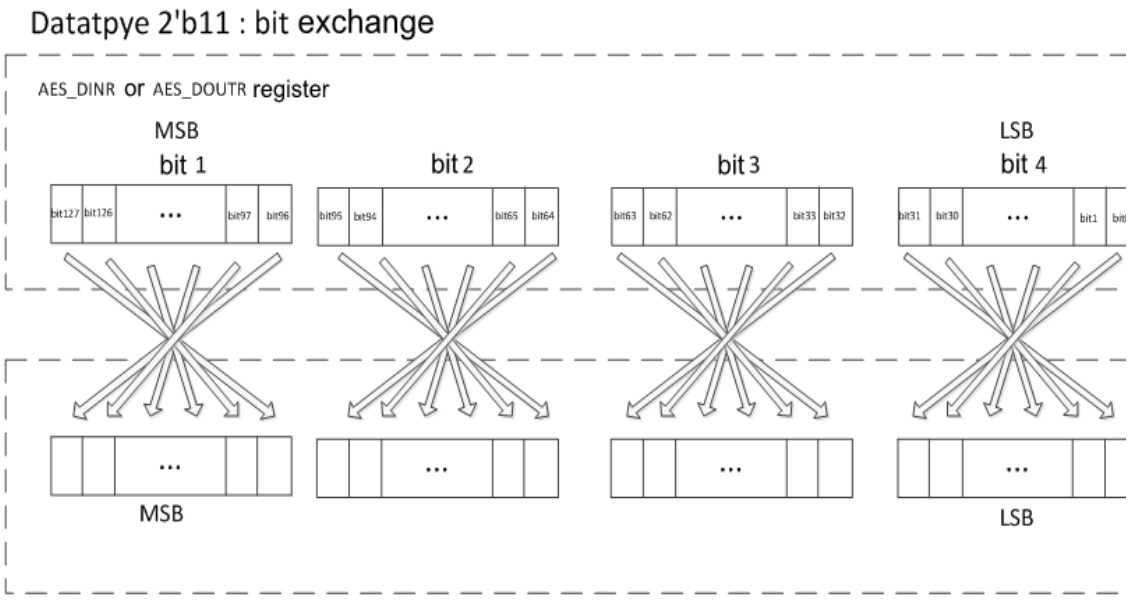


Figure 14-12 Data storage format under different data type

14.5 Operation Sequence

14.5.1 Mode 1: Encryption

- Set EN=0 to reset AES
- Set mode register MODE[1:0]=00, set stream data processing mode register CHMOD[1:0]
- Write AES_KEYRx register, and if under CTR and CBC mode user should also write AES_IVRx register
- Set EN=1 to enable AES
- Input the data by writing 4 times to AES_DINR register
- Wait for the CCF flag to set
- Read 4 times from AES_DOUTR for the entire encryption result
- For the same key, repeat steps 5,6,7 to encrypt the next 128bit block

The operation sequence can be described by Figure14-13.

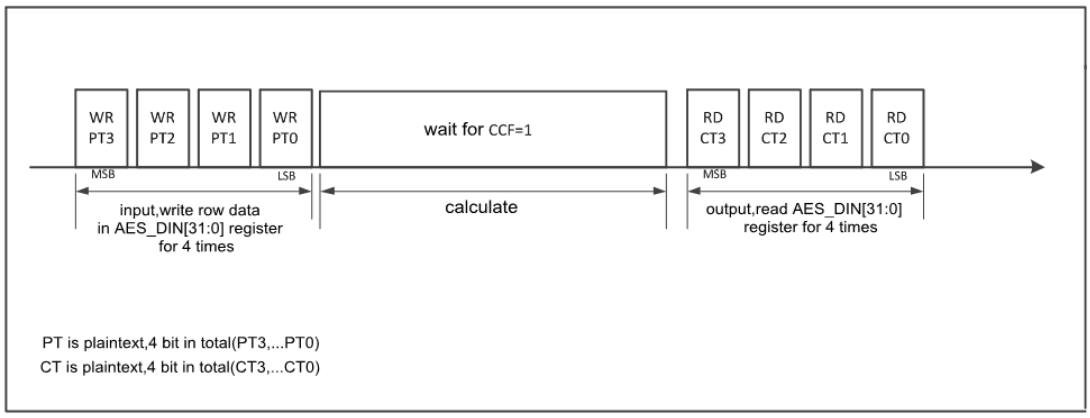


Figure 14-13 Operation sequence of Mode 1: Encryption

14.5.2 Mode 2: Key expansion

- Set EN=0 to reset AES
- Set mode register MODE[1:0]=01 and do not care CHMOD[1:0]
- Write AES_KEYRx register
- Set EN=1 to enable AES
- Wait for the CCF flag to set
- Clear The CCF flag. The expanded key is automatically written back to the AES_KEYRx register. You can read the AES_KEYRx register to get the result if needed. To re-calculate the expanded key, repeat steps 3,4,5,6.

The operation sequence can be described by Figure14-14.

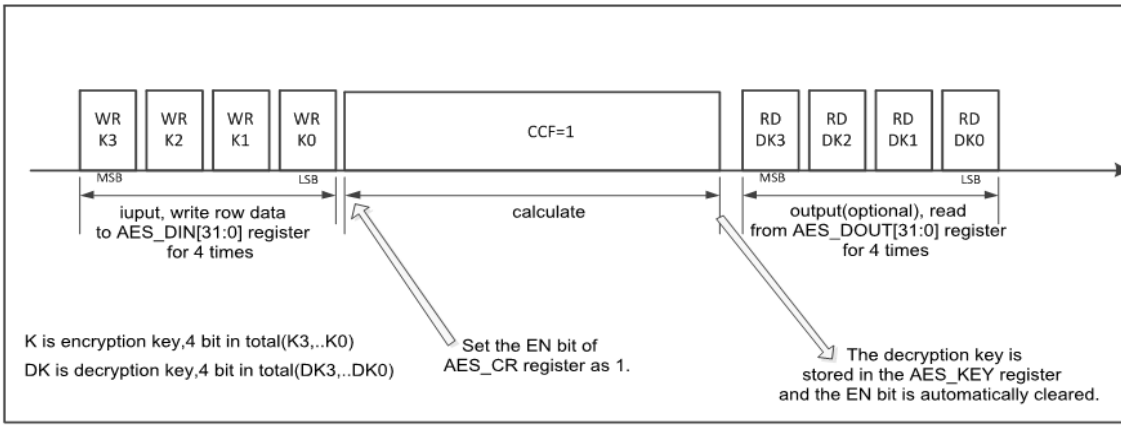


Figure 14-14 Operation sequence of Mode 2: Key expansion

14.5.3 Mode 3: Decryption

- Set EN=0 to reset AES
- Set mode register MODE[1:0]=10, set stream data processing mode register CHMOD[1:0]
- Write AES_KEYRx register (skip this step if the derived key is already calculated by mode 2), and if under CTR and CBC mode the user should also write AES_IVRx register.
- Set EN=1 to enable AES
- Input the data by writing 4 times to AES_DINR register
- Wait for the CCF flag to set
- Read 4 times from AES_DOUTR for the entire decryption result

For the same key, repeat steps 5,6,7 to decrypt the next 128bit block.

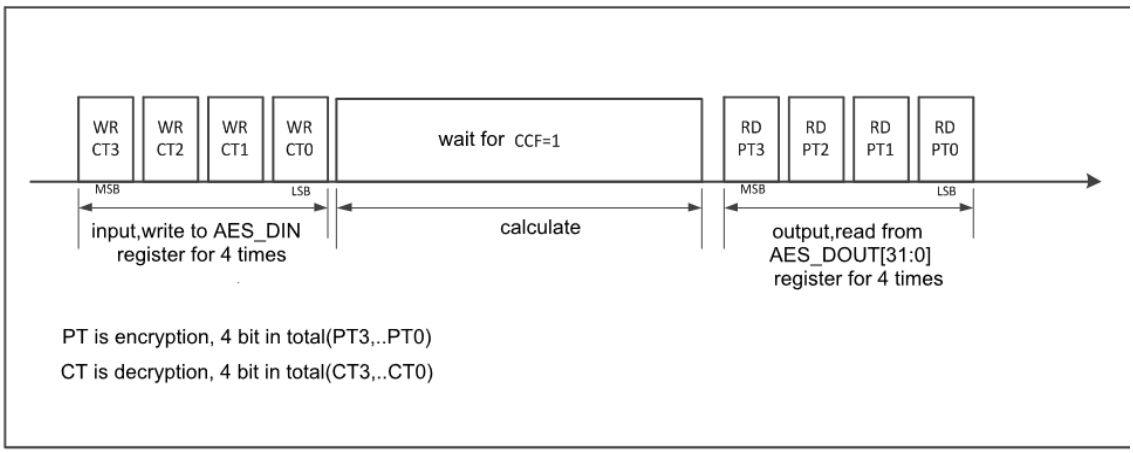


Figure 14-15 Operation sequence of Mode 3: Decryption

14.5.4 Mode 4: Key expansion + Decryption

- Set EN=0 to reset AES
- Set mode register MODE[1:0]=11, set stream data processing mode register CHMOD[1:0]. This mode is disabled under CTR mode. If the user sets MODE[1:0]=11 and CHMOD[1:0]=10, AES module will be forced to enter CTR decryption mode.
- Write AES_KEYRx register, and if under CBC mode the user should also write AES_IVRx register.
- Set EN=1 to enable AES
- Input the data by writing 4 times to AES_DINR register
- Wait for the CCF flag to set
- Read 4 times from AES_DOUTR for the entire decryption result

- For the same key, repeat steps 5,6,7 to decrypt the next 128bit block

  Note：*The AES_KEYRx register in this mode always stores the cipher key, and the derived key is recalculated internally each time without being stored in the AES_KEYRx register.*

The operation sequence can be described by Figure14-16.

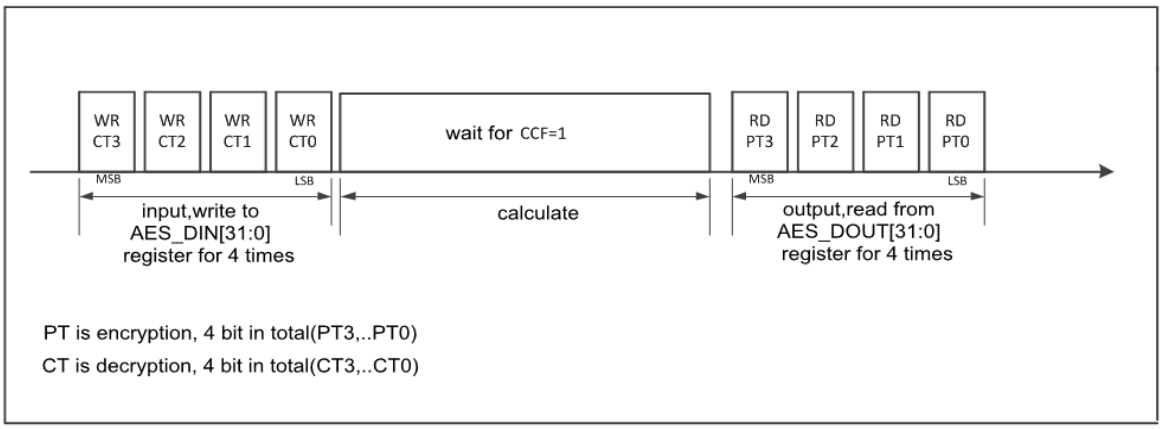


Figure 14-16 Operation sequence of Mode 4: Key expansion + decryption

14.5.5 Using the MultH module

- Set EN=0 to reset AES
- Set stream data processing mode register CHMOD[1:0] = 11. The value of MODE[1:0] register under this mode cannot be 01(mode2: Key expansion). Configuring both MODE[1:0]=01 and CHMOD[1:0]=11 will result in key expansion operation due to higher priority of MODE[1:0] register
- Write the AES_KEYRx register, the high 128bit is the output value of the previous calculation, if it is the first round of calculation, the initial value is 0x00000000. The lower 128bit is the value of H.
- Set EN=1 to enable the multH module
- Input the data by writing 4 times to AES_DINR register. The MultH module will XOR the last calculation result and the value written into AES_DINR register as the multiplier of the multH module. So assigning the result of the last calculation to 0x00000000directly making the input value of AES_DINR register as the multiplier of multH module
- Wait for the CCF flag to set
- Reads the calculation result from the AES_IVR register
- For the same H, repeat steps 5,6 for continuous calculations, which is how a GMAC is implemented.

The operation sequence can be described by Figure14-17.

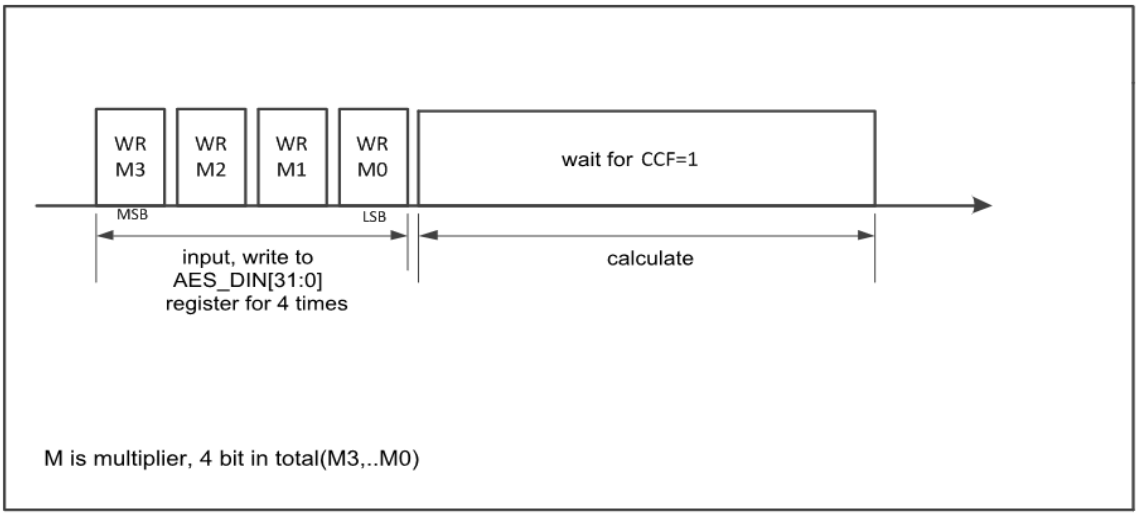


Figure 14-17 Operation sequence of using MultH module

14.6 DMA interface

- An input request channel: When DMAINEN=1, a DMA request is initiated whenever AES needs input data to be written to the AES_DINR register.
- An output request channel: When DMAOUTEN=1, a DMA request is initiated whenever AES needs to output data from the AES_DOUTR register.

Four DMA requests are generated in each phase, and the requests will generate continuously until the AES module is disabled. 128 bits of data are automatically fetched for the next calculation after the AES calculation.

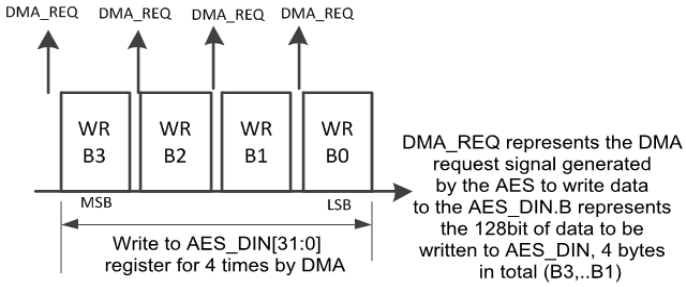


Figure 14-18 DMA request and data transfer sequence while input

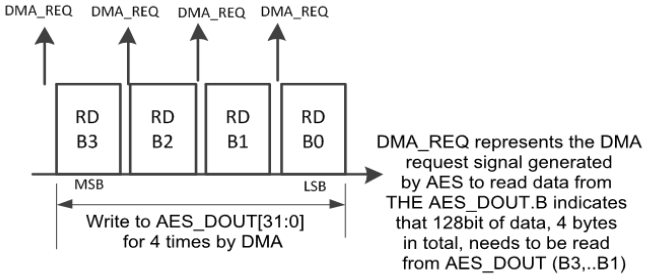


Figure 14-19 DMA request and data transfer sequence while output

14.6.1 MultH Module Interfaces with DMA

MultH calculations can also be done via DMA. When DMAINEN=1 and CHMOD[1:0]=11, a DMA request is initiated whenever AES needs input data to be written to the AES_DINR register. Setting DMAOUTEN=1 is invalidated under this mode and AES module will not generate DMA requests.

14.7 Error flags

A read operation occurs during the compute and input phases will set RDERR.

A write operation occurs during the compute and output phases will setWRERR.

The AES module will not be automatically stopped by the hardware after an error is generated, but it continues to operate as normal.

14.8 Register

Offset	Name	Symbol
AES(Module Base Address:0x4001B800)		
0x00000000	AES Control Register	AES_CR
0x00000004	AES Interrupt Enable Register	AES_IER
0x00000008	AES Interrupt Status Register	AES_ISR
0x0000000C	AES Data Input Register	AES_DIR
0x00000010	AES Data Output Register	AES_DOR
0x00000014	AES Key Register 0	AES_KEY0
0x00000018	AES Key Register 1	AES_KEY1
0x0000001C	AES Key Register 2	AES_KEY2
0x00000020	AES Key Register 3	AES_KEY3
0x00000024	AES Key Register 4	AES_KEY4
0x00000028	AES Key Register 5	AES_KEY5
0x0000002C	AES Key Register 6	AES_KEY6
0x00000030	AES Key Register 7	AES_KEY7
0x00000034	AES Initial Vector Register 0	AES_IVR0
0x00000038	AES Initial Vector Register 1	AES_IVR1
0x0000003C	AES Initial Vector Register 2	AES_IVR2
0x00000040	AES Initial Vector Register 3	AES_IVR3

14.8.1 AES Control Register (AES_CR)

NAME	AES_CR							
offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-	KEYLEN		DMAOEN	DMAIEN	-	-	-
access	U-0	R/W-00		R/W-0	R/W-0	U-0	U-0	U-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	CHMOD		MODE		DATATYP		EN
access	U-0	R/W-00		R/W-00		R/W-00		R/W-0

Bit	Name	Description
31:15	-	**Reserved, read as 0**
14:13	KEYLEN	AES Key Length. Cannot be modified when EN=1. 00：128-bit 01：192-bit 10：256-bit 11：Reserved
12	DMAOEN	DMA Output Enable 0：Disable DMA output 1：Enable DMA output The AES module will automatically generate AES->RAM transfer requests under mode 1, mode 3 and mode 4 after setting this bit to 1. Requests will not be generated in mode 2.
11	DMAIEN	DMA Input Enable 0：Disable DMA input 1：Enable DMA input The AES module will automatically generate RAM->AES transfer requests under mode 1, mode 3, mode 4 and MultH mode after setting this bit to 1. Requests will not be generated in mode 2.
10:7	-	**Reserved, read as 0**
6:5	CHMOD	AES data stream processing Mode. Cannot be modified when EN=1. 00：ECB 01：CBC 10：CTR 11：Use MultH module
4:3	MODE	AES Operation Mode. Cannot be modified when EN=1. Configure MODE=2'b11 when CHMOD=2'b10, AES will be executed according to the case of MODE=2'b10. That is, configure MODE toMode 4 under CTR mode will automatically enter the decryption mode of CTR. 00：Mode 1：Encryption 01：Mode 2：Key expansion 10：Mode 3：Decryption 11：Mode 4：Key expansion + Decryption
2:1	DATATYP	AES Data Type. Specific exchange rules can be found in the AES Data Types section. Cannot be modified when EN=1. 00：32-bit data. No swapping for each word 01：16-bit data with half-word swapping 10：8-bit data with byte swapping 11：1-bit data with bit swapping
0	EN	AES Enable This bit can be cleared at any time to reset the AES module. In

Bit	Name	Description
		mode 2 this bit is automatically cleared by hardware after a calculation is completed. 0：Disable AES 1：Enable AES

14.8.2 AES Interrupt Enable Register (AES_IER)

NAME	AES_IER							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					WRERR_IE	RDERR_IE	CCF_IE
access	U-0					R/W-0	R/W-0	R/W-0

Bit	Name	Description
31:3	-	Reserved, read as 0
2	WRERR_IE	Write Error Interrupt Enable 0：Disable Write Error Interrupt 1：Enable Write Error Interrupt
1	RDERR_IE	Read Error Interrupt Enable 0：Disable Read Error Interrupt 1：Enable Read Error Interrupt
0	CCF_IE	Calculation Complete Interrupt Enable 0：Disable Calculation Complete Interrupt 1：Enable Calculation Complete Interrupt

14.8.3 AES Interrupt Status Register (AES_ISR)

NAME	AES_ISR							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							

bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					WRERR	RDERR	CCF
access	U-0					R/W-0	R/W-0	R/W-0

Bit	Name	Description
31:3	-	Reserved, read as 0
2	WRERR	Write Error Interrupt Flag This flag is set by hardware when write error occurred. It is cleared by software by writing 1 tothis bit.
1	RDERR	Read Error Interrupt Flag This flag is set by hardware when read error occurred. It is cleared by software by writing 1 tothis bit.
0	CCF	AES Calculation CompleteFlag This flag is set by hardware whenAES calculation has completed. It is cleared by software by writing 1 tothis bit.

14.8.4 AES Data Input Register (AES_DIR)

NAME	AES_DIR							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	DIN[31:24]							
access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	DIN[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DIN[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DIN[7:0]							
access	R/W-0000 0000							

Bit	Name	Description
31:0	DIN	AES Data Input register, when inputing data, should write to this register 4 times continuously. Mode 1 (Encryption): Write plaintext from MSB to LSB in 4

		times. Mode 2 (Key expansion): No need to write this register Mode 3 and Mode 4 (Decryption): Write the ciphertext from MSB to LSB in 4 times. MultH Mode: Write the multiplier A or C from MSB to LSB in 4 times.

14.8.5 AES Output Register (AES_DOR)

NAME	AES_DOR							
offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	DOUT[31:24]							
access	R-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	DOUT[23:16]							
access	R-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DOUT[15:8]							
access	R-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DOUT[7:0]							
access	R-0000 0000							

Bit	Name	Description
31:0	DOUT	AES Data Output register. User can read out the result of encryption and decryption in four times when AES calculation is finished. Mode 1 (Encryption): Read out the cipher text from MSB to LSB in 4 times. Mode 2 (Key expansion): no use. Mode 3 and Mode 4 (Decryption): Output the plaintext from MSB to LSB in 4 times. MultH mode: the result of the operation is stored in the IVR register, so there's no need to read the AES_DOUTR register.

14.8.6 AES Key Register x (AES_KEYx)

NAME	AES_KEYx(x=0,1,2,3,4,5,6,7)							
Offset	0x00000014 + x*0x04							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	KEYx[31:24]							

access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	KEYx[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	KEYx[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	KEYx[7:0]							
access	R/W-0000 0000							

Bit	Name	Description
31:0	KEYx	AES Key At most 256-bit, AESKEY0 stores the lowest 32bit, AESLKEY7 stores the highest 32bit. AESKEY0~3 store H[127:0] in MultH mode

14.8.7 AES Initial Vector Register x (AES_IVRx)

NAME	AES_IVRx(x=0,1,2,3)							
Offset	0x00000034 + x*0x04							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	IVRx[31:24]							
access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	IVRx[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	IVRx[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	IVRx[7:0]							
access	R/W-0000 0000							

Bit	Name	Description
31:0	IVRx	AES calculates a 128bit initial vector and saves the calculation result in MultH mode.

15 True Random Number Generator (TRNG)

15.1 Introduction

FM33LE0xxA uses two True random noise sources of Galois as true random number seed, and combines with simple online detection (32-bit all 0 and all 1 detection), LFSR post-processing and pseudo-random LFSR to form a true random number generator.

TRNG's start-up testing and full online testing require firmware implementation.

Galois noise source sampling and LFSR is recommended to use a 4MHz clock. The interval between two fetching of 32bit random numbers should not be less than 32 clock cycles.

The true random number generator has passed the FIPS PUB140-2 test with a success rate of 99.9%.

15.2 Functional description

15.2.1 Random number generation

The following figure shows the block diagram of the true random number generator.

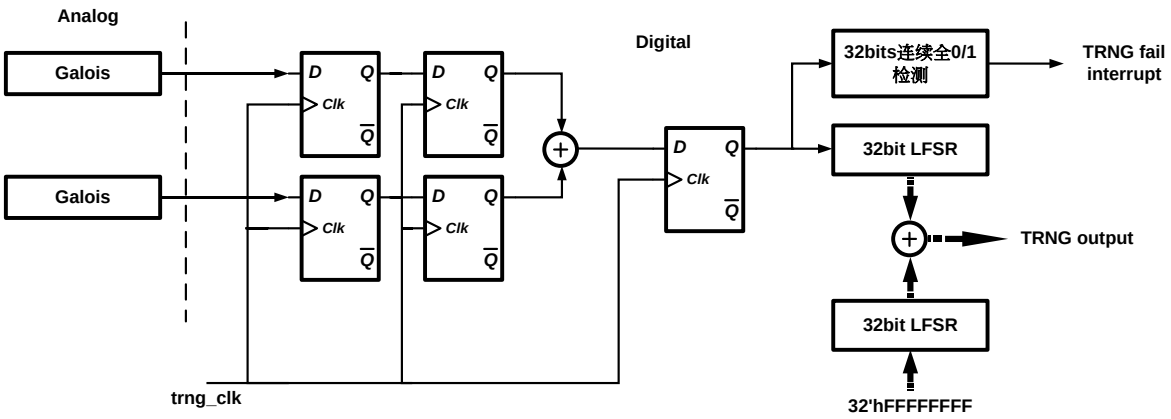


Figure 15-1 True random number generation module

The true random noise sources are 2 Galois ring oscillators. The output of Galois ring oscillators are XORed in the digital circuit and sampled by the system clock, and then processed by LFSR. Before LFSR post-processing, random number online detection is carried out. If 32 bits continuous 0 or 1 are found, TRNG failure alarm interrupt will be generated. At the same time, in order to avoid the poor randomness, another set of LFSR is used to produce a sequence of pseudo random number. And two sets of LFSR are XORed to generate final random data.

15.2.2 Operating clock

The operating clock of random number generator is the divided version of RCHF which is independent of APBCLK. In order to ensure the quality of random numbers, it is generally recommended that 4M clock be used as the random number operating clock, and the frequency divider register (OPCCON2.RNGPRSC) in CMU module should be configured according to the 4MHz target frequency. The clock diagram is as follows:

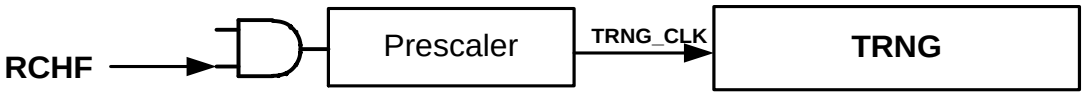


Figure 15-2 Clockfor TRNG

15.2.3 Random number read

When the TRNG module is enabled, the true random noise source and the LFSR post-processing module start to work simultaneously. The software reads out 32 bits of random numbers each time by reading the RNGOUT register. Since the LFSR length is 32 bits, in order to ensure the quality of random numbers, the application should ensure that the interval between two reads of RNGOUT is greater than 32 cycles of TRNG_CLK.

For example, assuming that TRNG_CLK is 4MHz, the interval between two readings to RNGOUT register should not be less than 8us.

15.2.4 CRC calculation

The LFSR used for post-processing can also be used for CRC calculations. During CRC operation, two groups of 32-bit registers are used as input data register and CRC shifting register, and 32-bit data can be calculated at one time. Before CRC operation, the CPU needs to inquire whether the current LFSR is occupied. If LFSR is idle, CRC function can be used. Once the CPU starts the CRC operation, the LFSR is automatically set to the reset value, and then 32bits data shifting is carried out. After the calculation, the CRC start register is cleared without interruption. After starting CRC, the software should keep polling the start register status until the end of the calculation.

CRC polynomial:

CRC32=X32+X26+X23+X22+X16+X12+X11+X10+X8+X7+X5+X4+X2+X1+X0

Software operation flow:

- Polling LFSR_BUSY to confirm that LFSR is not occupied

- Writes the data to CRCDATA0~3
- Set CRC_EN
- Polling and wait for CRC_EN to be cleared
- Reads the result from LFSROUT0~3

15.3 Register

Offset	Name	Symbol
RNGCTL(base address:0x4001A868)		
0x00000000	Random Number Generator Control Register	RNGCTL_CR

Offset	Name	Symbol
RNG (base address:：0x4001BC00)		
0x00000004	Random Number Generator Data Output Register	RNG_DOR
0x00000010	Random Number Generator Status Register	RNG_SR
0x00000014	CRC Control Register	RNG_CRCCR
0x00000018	CRC Data input Register	RNG_CRCDIR
0x0000001C	CRC Status Register	RNG_CRCSR

15.3.1 Random Number Generator Control Register (RNGCTL_CR)

NAME	RNGCTL_CR							
offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							EN
access	U-0							R/W-0

bit	name	functional description
31:1	-	RFU: Reserved, read as 0
0	EN	RNG enable register, software writes 1 to enable (RNG enable) 1: Enable RNG 0: Disable RNG

0x0000001C	CRC Status Register	RNG_CRCSR

15.3.2 Random Number Generator Data Output Register (RNG_DOR)

NAME	RNG_DOR							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	RNGOUT[31:24]							
access	R-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	RNGOUT[23:16]							
access	R-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	RNGOUT[15:8]							
access	R-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RNGOUT[7:0]							
access	R-0000 0000							

bit	name	functional description
31:0	RNGOUT	Random number result or CRC result register (RNG output, read-only)

15.3.3 Random Number Generator Status Register (RNG_SR)

NAME	RNG_SR							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						LFSREN	RNF
access	U-0						R-0	R/W-0

bit	name	functional description
31:2	-	RFU: Reserved, read as 0
1	LFSREN	LFSR status flag, read-only 1: LFSR is running 0: LFSR is not running, CRC can be performed
0	RNF	Random Number generation failureflag 1: The random number failed to pass the quality test 0: Random number passed quality detection

15.3.4 CRC Control Register (RNG_CRCCR)

NAME	RNG_CRCCR							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							CRCEN
access	U-0							R/W-0

bit	name	functional description
31:1	-	RFU: Reserved, read as 0
0	CRCEN	CRC enable control register, software write 1 start CRC, automatically cleared by hardware after CRC calculation is finished (CRC enable) 1: CRC enable 0: CRC disable

15.3.5 CRC Data Input Register (RNG_CRCDIR)

NAME	RNG_CRCDIR							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	CRCIN[31:24]							
access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	CRCIN[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CRCIN[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CRCIN[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:0	CRCIN	CRC Data Input Register

15.3.6 CRC Status Register (RNG_CRCSR)

NAME	RNG_CRCSR							
Offset	0x0000001C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							CRCDONE
access	U-0							R/W-0

bit	name	functional description
31:1	-	RFU: Reserved, read as 0
0	CRCDONE	CRC calculation done flag, software write 0 to clear 1: CRC calculation was completed 0: CRC calculation was not completed

16 Comparator (COMP)

16.1 Introduction

The chip integrates 2 analog comparators and supports the following features:

- 2 comparators, rail-to-rail inputs, support low-power mode and fast mode
- Flexible input selection
  - IO input
  - Internal reference voltage and its voltage divider
- Interrupt events to wake up the MCU
- The output signal can be connected to GPIO, or as a trigger source to timer, ADC

16.2 Block diagram

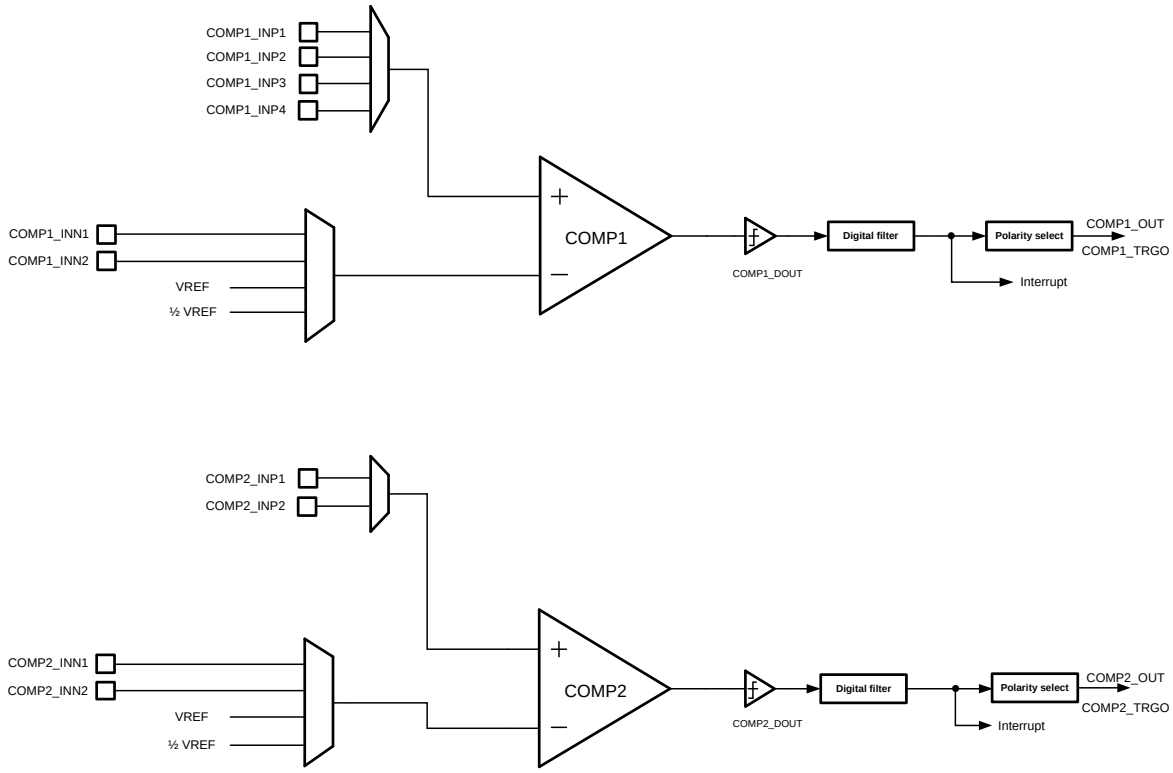


Figure 16-1 Comparator circuit block diagram

The comparator structure is shown above. The reference voltage VREF and its divided version can be used as negative input. The two comparators are identical and the inputs are connected as shown in the structure diagram.

The input voltage range of the comparator is 0~VDD, the setup time is less than 15us, and the transmission delay is less than 2us.

16.3 Pin definition

The comparator input can come from the GPIO analog channel, and its output can be sent to off-chip from FOUT0 and FOUT1.

Pin	COMPx	Symbol	Description
PD4	COMP1	COMP1_INP1	Comparator positive input
PD5		COMP1_INP2	
PA13		COMP1_INP3	
PA14		COMP1_INP4	
PA10		COMP1_INN1	Comparator negative input
PD11		COMP1_OUT(FOUT0)	Comparator output
PA8	COMP2	COMP2_INP1	Comparator positive input
PA9		COMP2_INP2	
PA4		COMP2_INN1	Comparator negative input
PA5		COMP2_INN2	
PB12		COMP2_OUT(FOUT1)	Comparator output

Table 16-1 Comparator pin list

16.4 Function description

16.4.1 Basic Functions

The comparator compares the positive input voltage with the negative input voltage and outputs a logic high when the positive voltage is higher than the negative voltage, and vice versa.

The positive input voltage can be selected from multiple pin inputs, and the negative input voltage

can be selected from pin inputs or internal reference voltage.

The logic signal output by the comparator can generate an interrupt signal.

16.4.2 Clock and reset

The comparator output supports digital filtering. The filtering method is to use APBCLK to continuously sample the output of the comparator, and the level is considered valid when the number of sampling cycles is kept at the same level for 3 times.

The following figure shows a schematic diagram of digital filtering:

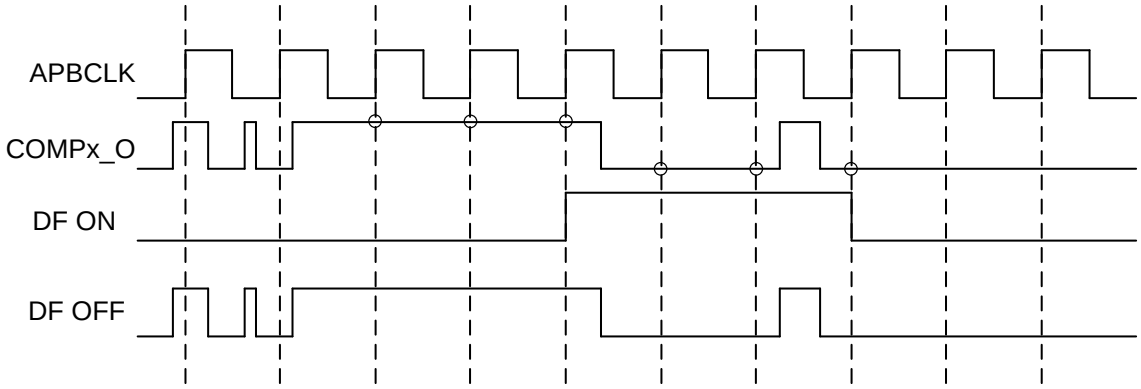


Figure 16-2 Schematic diagram of digital filtering waveform

16.4.3 Interrupt and trigger signal output

The comparator output can generate interrupts and trigger signals for other peripheral modules.

Comparator interrupt

The comparator output can generate independent interrupt events on the rising and falling edges. The CMPxIE register can enable or disable interrupt output. The CMPxIF flag register is set when an interrupt event occurs, and the software writes 1 to clear it. Software can also directly read the output value of the comparator through the CMPxO register.

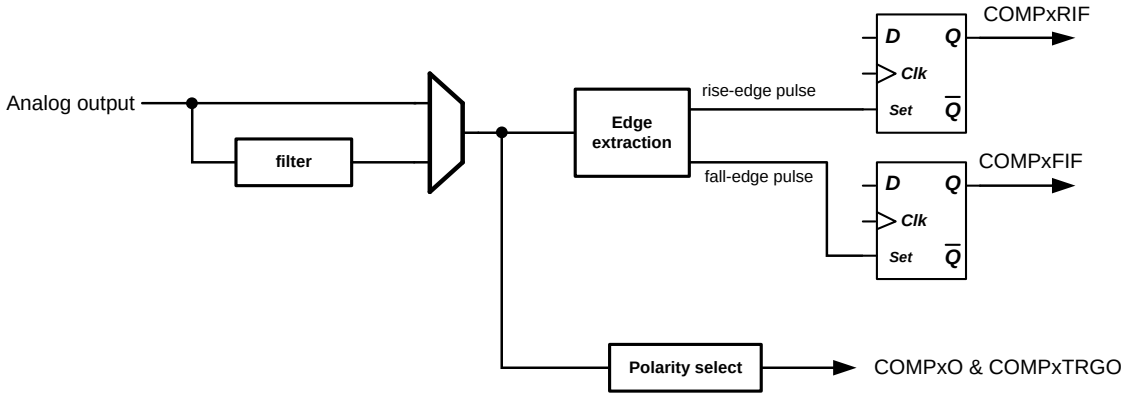


Figure 16-3 Comparator interrupt generation

Comparator trigger signal output

The trigger signal output can be generated separately or at the same time on the rising edge and falling edge of the comparator output. When the trigger signal needs to be output, the COMP bus clock must be enabled. When a trigger event occurs, a high-level trigger signal of the APBCLK cycle is generated on the rising edge of APBCLK. The trigger signal can be connected to the internal trigger input of the timer or the internal trigger input of the ADC.

The CMPxSEL register can be configured to generate the trigger signal output on which edge of the comparator output, and TRGOEN is used to enable or disable the trigger signal output.

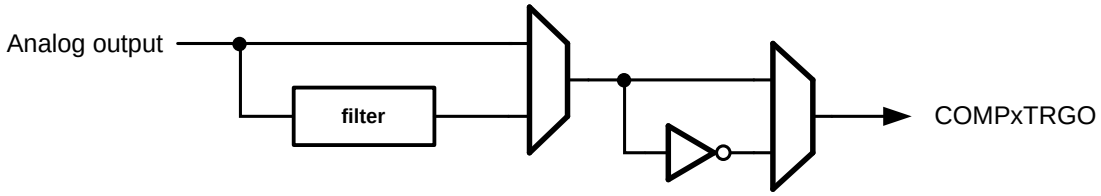


Figure 16-4 Comparator trigger output signal generation

16.4.4 Comparator output connection

The output of the comparator can be connected to the input of GPTIMx so that the timer can automatically record the number of times the comparator output has flipped.

For the specific connection relationship, please refer to 28.4.4 Capture of Internal Trigger Signal (ITRx).

The output of the comparator can also be used as the trigger signal to start ADC conversion, please refer to 28.4.4 Conversion Trigger.

16.5 Register

base address：0x4001A870

Offset	Name	Symbol
COMP(base address：0x4001A870)		
0x00000000	ComparatorControl Register 1	COMP_CR1
0x00000004	Comparator Control Register 2	COMP_CR2
0x00000008	Comparator Interrupt Config Register	COMP_ICR
0x0000000C	Comparator Interrupt Status Register	COMP_ISR

16.5.1 COMP Control Register 1（COMP_CR1）

NAME	**COMP_CR1**							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							CMP1O
access	U-0							R-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-		POL	V1PSEL		V1NSEL		CMP1EN
access	U-0		R/W-0	R/W-00		R/W-00		R/W-0

bit	name	functional description
31:9	-	RFU：Reserved，read as 0
8	CMP1O	Comparator 1 output, software read only
7:6	-	RFU：Reserved，read as 0
5	POL	Comparator1 output Polarity 0: The output is not inverted 1: Inverted output
4:3	V1PSEL	Comparator1 positive input select 00：COMP1_INP1 (PD4) 01：COMP1_INP2 (PD5) 10：COMP1_INP3 (PB12) 11：RFU
2:1	V1NSEL	Comparator1 negative input select

bit	name	functional description
		00：COMP1_INN1 01：RFU 10：VREF = 1.2V 11：VREF/2 = 0.6V
0	CMP1EN	Comparator1 Enable 0：disable comparator 1 1：enable comparator 1

16.5.2 COMP Control Register 2（COMP_CR2）

NAME	COMP_CR2							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							CMP2O
access	U-0							R-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-		POL	-	V2PSEL	V2NSEL		CMP2EN
access	U-0		R/W-0	U-0	R/W-0	R/W-00		R/W-0

bit	name	functional description
31:9	-	RFU：Reserved，read as 0
8	CMP2O	Comparator 2 output, software read only
7:6	-	RFU：Reserved，read as 0
5	POL	Comparator 2 output Polarity 0: The output is not inverted 1: Inverted output
4	-	RFU：Reserved，read as 0
3	V2PSEL	Comparator 2 positive input select 0：COMP2_INP1 (PA8) 1：COMP2_INP2 (PA9)
2:1	V2NSEL	Comparator 2 negative input select 00：COMP2_INN1 (PA4) 01：COMP2_INN2 (PA5) 10：VREF = 1.2V 11：VREF/2 = 0.6V
0	CMP2EN	Comparator 2 Enable 0：disable comparator 2 1：enable comparator 2

16.5.3 Comparator Control Register（COMP_ICR）

NAME	COMP_ICR							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24

name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						CMP2DF	CMP1DF
access	U-0						R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	-	CMP2SEL		CMP1SEL		CMP2IE	CMP1IE
access	U-0	U-0	R/W-00		R/W-00		R/W-0	R/W-0

bit	name	functional description
31:12	-	RFU：Reserved，read as 0
11	BUFBYP	Comparator buffer bypass enable 0：Disable bypass comparator buffer 1：Enable bypass comparator buffer
10	BUFENB	Comparator buffer enable 0：Enable comparator buffer 1：Disable comparator buffer
9	CMP2DF	Comparator2 Digital Filter enable 0: Disable digital filtering 1: Enable digital filtering
8	CMP1DF	Comparator1 Digital Filter enable 0: Disable digital filtering 1: Enable digital filtering
7:6	-	RFU：Reserved，read as 0
5:4	CMP2SEL	Comparator2 interrupt edge select 00/11: Comparator 2 output rising or falling edge generates an interrupt 01: Comparator 2 generates an interrupt on the rising edge of the output 10: Comparator 2 output falling edge generates an interrupt
3:2	CMP1SEL	Comparator1 interrupt edge select 00/11: Comparator 1 output rising or falling edge generates an interrupt 01: Comparator 1 generates an interrupt on the rising edge of the output 10: Comparator 1 output falling edge generates an interrupt
1	CMP2IE	Comparator2 Interrupt Enable 1: Allow interrupt 0: Disable interrupt
0	CMP1IE	Comparator1 Interrupt Enable 1: Allow interrupt

bit	name	functional description
		0: Disable interrupt

*Note: To avoid false triggering of interrupts, the interrupt source selection register should be set when interrupts disabled.

16.5.4 Comparator Interrupt Status Register（COMP_ISR）

NAME	COMP_ISR							
offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						CMP2IF	CMP1IF
access	U-0						R/W-0	R/W-0

bit	name	functional description
31:2	--	RFU：Reserved，read as 0
1	CMP2IF	Comparator 2 interrupt flag, hardware set, software write 1 clear
0	CMP1IF	Comparator 1 interrupt flag, hardware set, software write 1 clear

bit	name	functional description
31:2	-	RFU：Reserved，read as 0
1	CMP2IF	Comparator2 Interrupt Flag, write 1 to clear
0	CMP1IF	Comparator1 Interrupt Flag, write 1 to clear

17 Hardware Divider (HDIV)

17.1 Introduction

The hardware divider module is used to help the software accelerate the division operation. The hardware divider is a signed integer divider, which can output 32bit dividend and 16bit divisor, and output 23bit quotient and 32bit remainder.

HDIV main features:

- Signed integer operations (two's complement format)
- 32bit dividend, 16bit divisor
- Output 32bit quotient and 32bit remainder
- Divide by 0 will warn
- One calculation requires 8 cycles of 24MHz

17.2 Work process

The software calls the hardware divider according to the following steps.

- Write the 32bit dividend (two's complement) to the DIVEND register
- Write the 16bit divisor (two's complement) to the DIVSOR register
- The hardware divider automatically starts operation after the software writes DIVSOR, and sets the BUSY register at the same time
- The software queries the BUSY flag, and BUSY is automatically cleared after the calculation is completed
- Query the DIV_BY_0 flag
- Read the quotient in the QUOT register
- Read the remainder in the REMD register

17.3 Register

Offset	Name	Symbol
HDIV(base address:0x40019000)		
0x00000000	Dividend Register	HDIV_END
0x00000004	Divisor Register	HDIV_SOR
0x00000008	Quotient Register	HDIV_QUOT
0x0000000C	Reminder Register	HDIV_REMD
0x00000010	Status Register	HDIV_SR

17.3.1 Dividend register (HDIV_END)

NAME	HDIV_END							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	DIVEND[31:24]							
access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	DIVEND[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DIVEND[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DIVEND[7:0]							
access	R/W-0000 0000							

17.3.2 Divisor Register (HDIV_SOR)

NAME	HDIV_SOR							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DIVSOR[15:8]							
access	R/W-0000 0000							

bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DIVSOR[7:0]							
access	R/W-0000 0001							

17.3.3 Quotient Register (HDIV_QUOT)

NAME	HDIV_QUOT							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	QUOT[31:24]							
access	R-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	QUOT[23:16]							
access	R-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	QUOT[15:8]							
access	R-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	QUOT[7:0]							
access	R-0000 0000							

bit	name	functional description
31:0	QUOT	32bitsigned quotient,read only (Address only, no actual register, directly return to DW_div module output when read)

17.3.4 Remainder Register (HDIV_REMD)

NAME	HDIV_REMD							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	REMD[15:8]							
access	R-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	REMD[7:0]							
access	R-0000 0000							

bit	name	functional description

31:16	-	**RFU: Reserved, read as 0**
15:0	REMD	16bit signed reminder,read only(Address only, no actual register, directly return to DW_div module output when read)

17.3.5 Status Register (HDIV_SR)

NAME	**HDIV_SR**							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						DIV_BY_0	BUSY
access	U-0						R-0	R-0

bit	**name**	**functional description**
31:2	-	**RFU: Reserved, read as 0**
1	DIV_BY_0	Divided by 0 flag,read only 1: Divisor is 0 0: Divisor is not 0
0	BUSY	Busy flag,read only 1:HDIVis in the process of calculation, the result is not ready 0:The calculation is complete and the result is ready After the software writes the divisor, HDIV starts to calculation, the software should query BUSY to be low before reading the quotient and remainder registers

18 I²C-SMBus

18.1 Introduction

The I2C-SMBus module implements synchronous communication between MCU and external I2C or SMBus devices.

This module supports I2C single-host, multi-host and slave modes, and is compatible with standard-mode 100Kbps, fast-mode 400Kbps, and enhanced fast-mode+ 1Mbps.

This module can also support SMBus rev3.0 (System Management Bus) and PMBus (Power Management Bus) protocol specifications.

The chip supports up to 1 channel independent I2C-SMBus.

18.2 Function

- I2C protocol compatibility
  - host and slave modes
  - multi-host communication
  - Transmission speed support standard mode(100Kbps), fast mode(400Kbps) and Fm+(1Mbps)
  - 7bit and 10bit addressing
  - General Call
  - Support programming setup/hold timing
  - Support clock extension
- SMBus rev 3.0 compatibility
  - Hardware PEC and ACK
  - Support ARP（Address Resolution Protocol）
  - Support Host and Device
  - SMBus alert signal
  - Overtime and IDLE detected
- PMBus rev 1.3 compatibility
- Independent clock operation, not dependent on APB bus clock
- DMA data handling support
- Low power features
  - Low power slave design, no on-chip clock required to complete data transmission and reception

■ Slave wake-up function: support slave address matching wake-up, data frame reception completion wake-up or START detection wake-up

18.3 I²C-SMB module diagram

18.3.1 I²C-SMB module diagram

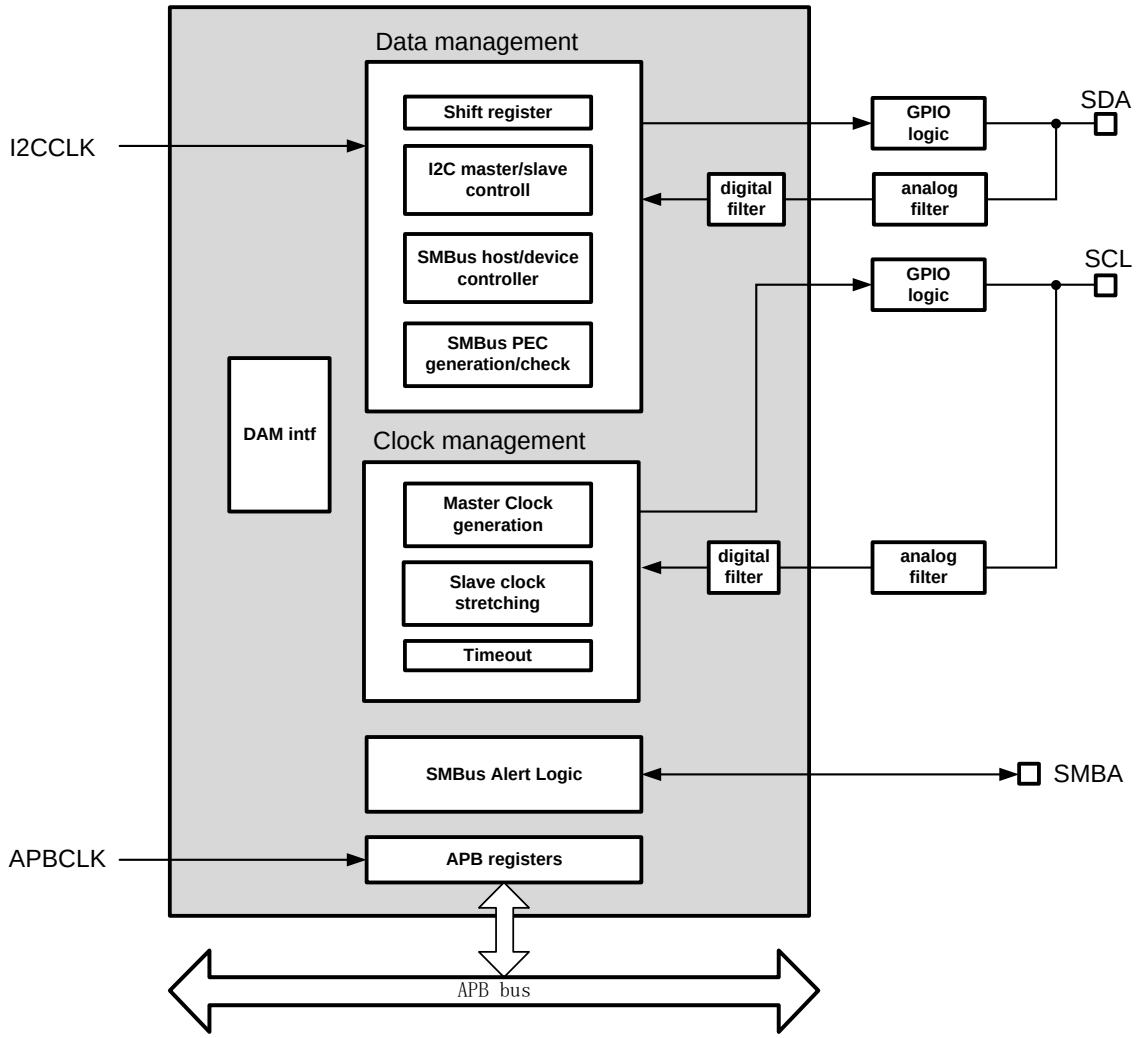


Figure 18-1 I²C-SMB module diagram

18.4 Pin definition

The I2C_SMB module uses up to 3 pins to communicate with external devices.

Pin	I²C-SMBx	Symbol	Function
PB0,PC8	I2C_SMB0	SCLA0	Serial Clock
PB1,PC9		SDAA0	Serial Data
PB11,PC10		SMB0_ALERTB	SMBus Alert#
PD9,PB2	I2C_SMB1	SCLA1	Serial Clock
PD8,PB3		SDAA1	Serial Data
PD13,PB4		SMB1_ALERTB	SMBus Alert#

Table 18-1 I²C pin list

18.5 Clock and reset

Both I2C hosts and slaves use a dual clock structure:

- The bus register clocks of the host and slave are represented by PCLK and originate from APBCLK. PCLK must be enabled when the CPU or DMA needs to access the I2C internal registers.
- The data sending and receiving clock of the host is represented by I2CCLK, which can be derived from APBCLK, but also derived I2CCLK must be enabled to send and receive data.
- The slave performs data transmission and reception by sampling the active edge of the SCL input

The control of PCLK and I2CCLK is done in the CMU module and the corresponding CMU control registers must be properly configured before I2C communication.

By using a dual clock structure, the I2C operation is not limited by the APBCLK configuration. When some peripherals need to work on a very high APBCLK frequency, the I2C can still work on a reduced frequency; or conversely, the CPU works on a lower frequency, which does not affect the I2C data communication at a higher baud rate.

Theoretically there is no relative relationship between PCLK and the baud rate clock, the baud rate clock can be faster or slower than PCLK, but the application needs to pay attention to whether the CPU or DMA has time to carry the data when the difference between the two frequencies is large.

Note: The I2C reset register (I2CSMBRST) in the RMU module needs to be cleared before the module works.

18.6 Interface Timing

18.6.1 Communication Process

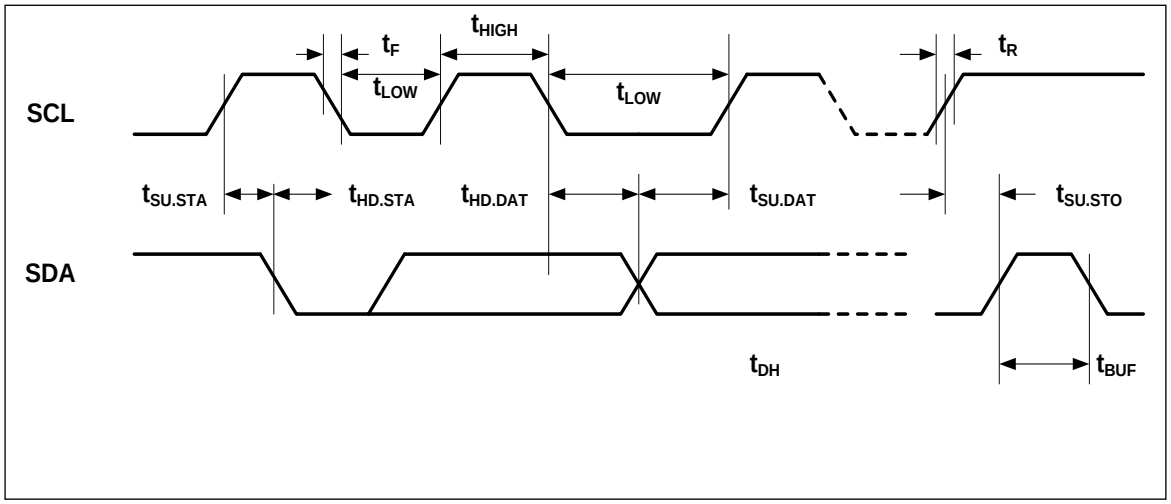


Figure 18-2 I²C Bus Protocol

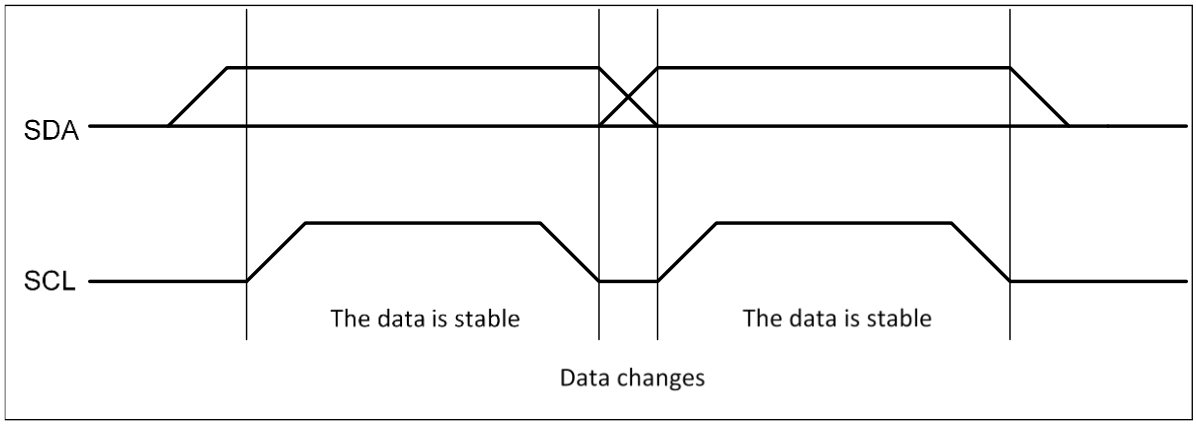


Figure 18-3 Bit Protocol

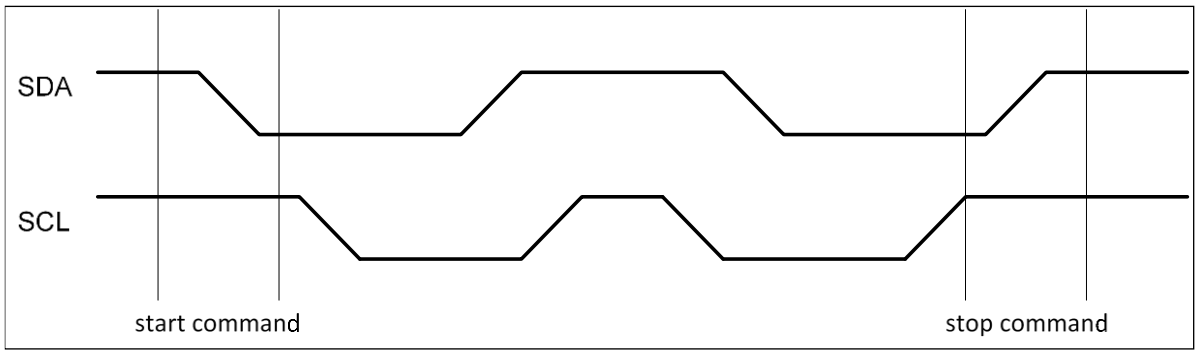


Figure 18-4 Start & Stop condition definition

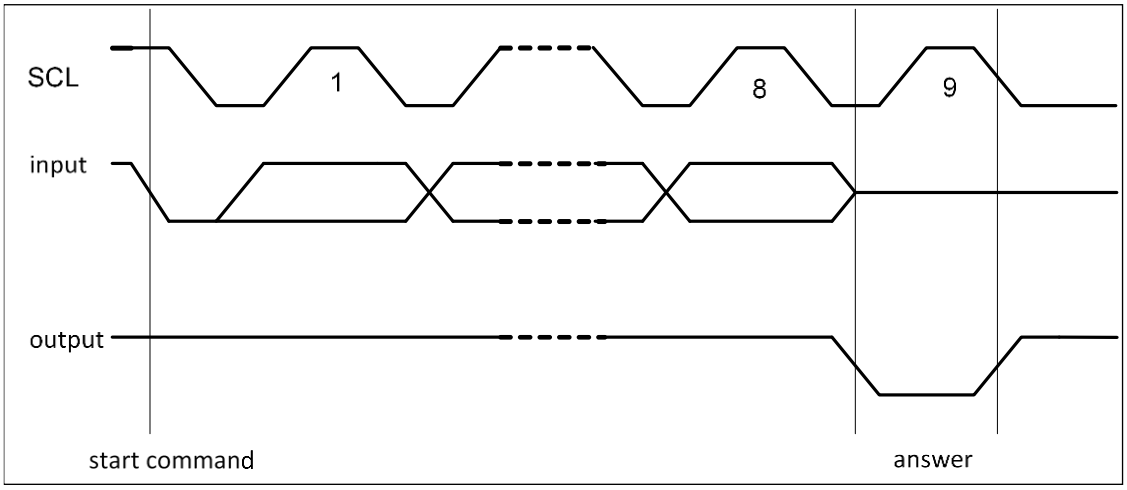


Figure 18-5 ACK

18.6.2 Interface Timing Description

Clock timings: The SDA pin is normally pulled high by the peripheral devices. The data on the SDA pin should change when SCL is low (refer to Figure19-3); when the data changes when SCL is high, it will be considered as a start or stop command as described below.

Start condition: When SCL is high, the change of SDA from high to low is considered as a start condition and must be used as the start of any read/write operation (refer to Figure 19-4).

Stop condition: When SCL is high, the change of SDA from low to high is considered as a stop condition, and after a read operation, the stop condition puts the EEPROM into wait-state or low-power mode (refer to Figure 19-4).

Acknowledge: Data on the SDA is transmitted in groups of 8 bits serially, MSB first, and the receiver should send back an acknowledge bit (hereafter ack) on the 9th cycle after receiving each byte. The clock for ack is provided by the master. The sender leaves SDA undriven during the ack period and the receiver shall pull SDA low to ensure that SDA is low when the clock high, creating a valid ack signal (refer to Figure 19-5).

18.6.3 Bus Timing Parameters Table

The bus timing parameters specified by the I2C protocol are listed below.

Parameter	Sign	Standard mode(100K)		Fast mode (400K)		Fast mode plus (1M)		Units
		min	max	min	max	min	max	
SCLclock frequency	F_{SCL}	0	100	0	400	0	1000	kHz
Start conditon establishment time	$T_{SU:STA}$	4.7	-	0.6	-	0.26	-	us
Start condition stretching time	$T_{HD:STA}$	4.0	-	0.6	-	0.26	-	us
Clock stretching low time	T_{LOW}	4.7	-	1.3	-	0.5	-	us
Clock stretching high time	T_{HIGH}	4.0	-	0.6	-	0.26		us
Data input setup time	$T_{SU:DAT}$	250	-	100	-	50	-	ns
Data input stretching time	$T_{HD:DAT}$	0	-	0	-	0	-	us
SDAand SCLpull-up times	T_R	-	1000	20	300		120	ns
SDAand SCLpull-down time	T_F	-	300	20x(VDD /5.5)	300	20x(VDD /5.5)	120	ns
Stop condition establishment time	$T_{SU:STO}$	4.0	-	0.6	-	0.26	-	us
Bus idle time between STOP and START	T_{BUF}	4.7	-	1.3	-	0.5	-	us
Capacitive load on the bus	Cb	-	400	-	400	-	550	pF
Data validity time	$t_{VD;DAT}$	-	3.45	-	0.9	-	0.45	us
ACK effective time	$t_{VD;ACK}$	-	3.45	-	0.9	-	0.45	us
Min Noise tolerance	V_{nL}	0.1V_{DD}	-	0.1V_{DD}	-	0.1V_{DD}	-	V
Max Noise tolerance	V_{nH}	0.2V_{DD}	-	0.2V_{DD}	-	0.2V_{DD}	-	V

Table 18-2 I^2C Interface Timing Requirements

Each of the above parameters can be found in the corresponding diagram in the following chart.

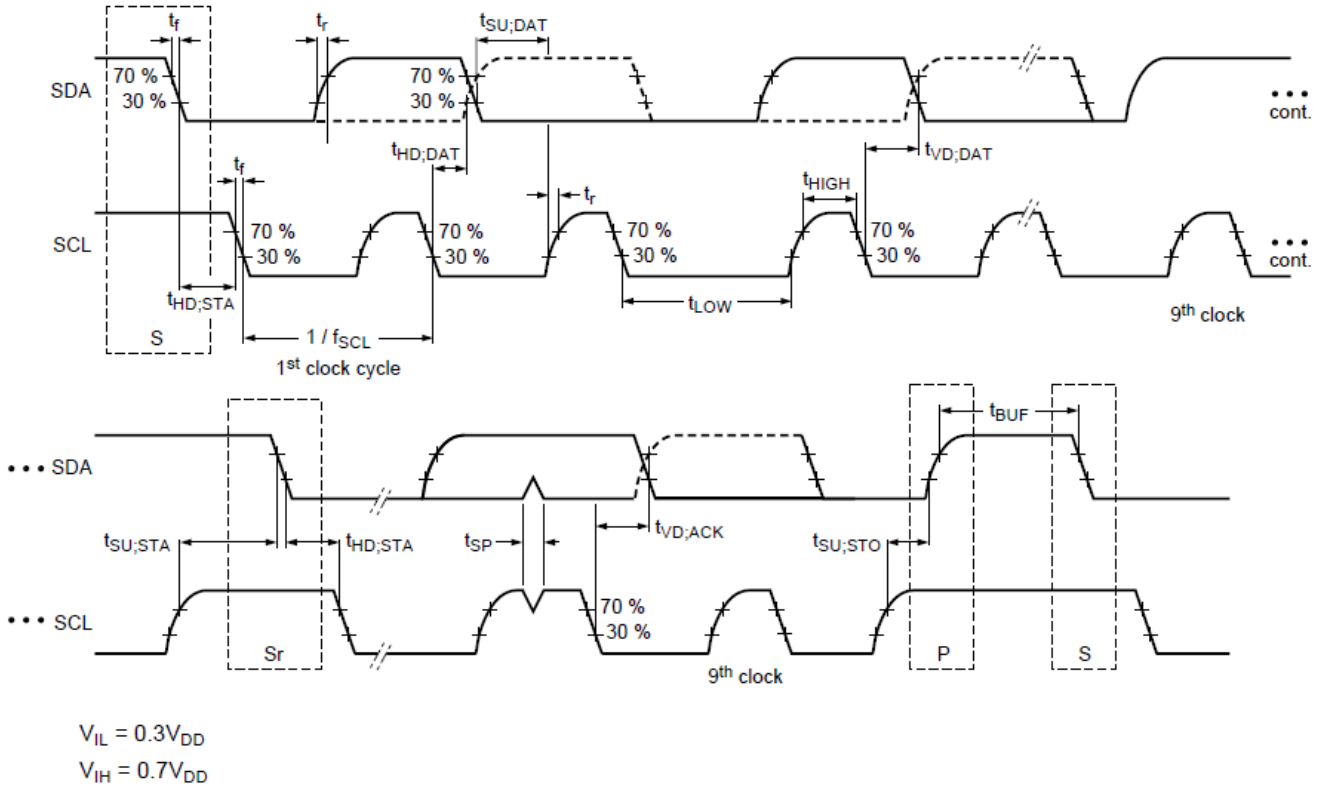


Figure 18-6 I²C Timing Parameters Legend

18.7 I²C function

The I2C module contains host, slave, IO interface control and global configuration registers. The host part is compatible with I2C master and SMBu master, and the slave part is compatible with I2C slave and SMBus slave; meanwhile, this module can also support SMBus Host function.

The module supports the following operating modes.。

- I²C master send
- I²C master receive
- I²C slave receive
- I²C salve send

The I2C module is disable by default after the chip is powered on, and neither the master nor the slave is working. The software enables the I2C module by setting the EN register, and the module is in slave mode by default when enabled and ready to receive addresses and data. When the software writes 1 to the START register, the module initiates the START timing on the bus, at which point the module automatically enters the master mode. When bus arbitration fails, or after sending the STOP timing, the module automatically returns from master mode to slave mode for compatibility with multi-host bus applications.

18.7.1 Preset salve address table

The I2C and SMBus protocols specify a portion of the preset addresses, and the module is able to automatically recognize some of these preset addresses, while the other preset addresses are judged and processed by software at the protocol level.

For 10bit slave address application, when A10EN=1, it is required that the first byte must start with 11110, otherwise the ADDR_ERROR error flag will be triggered. And in the case of A10EN=0, if the slave receives an address byte starting with 11110, the ADDR_ERROR error flag will also be set.

Slave address	R/W_ bit	Description
0000 000	0	General Call address
0000 000	1	START byte
0000 001	X	CBUS address
0000 010	X	Reserved for different bus format
0000 011	X	Reserved for future purpose
0000 1XX	X	HS-mode master code
0001 000	X	SMBus host

0001 001	X	Smart Battery Charger
0001 010	X	Smart Battery Selector Smart Battery System Manager
0001 011	X	Smart Battery
0001 100	X	SMBus Alert Response Address
0101 000	1	PMBus ZONE READ
0101 100	X	Reserved by SMBus for LCD controller
0101 101	X	Reserved by SMBus for CCFL Backlight driver
0110 111	0	PMBus ZONE WRITE
1000 0XX	X	Reserved by SMBus for PCMCIA Socket Controller
1000 100	X	Reserved by SMBus for VGA controller
1001 0XX	X	Prototype Address
1100 001	X	SMBus Device Default Address
1111 0XX	X	10bit slave addressing
1111 1XX	X	Reserved for future purpose

Table 18-3 I²C-SMBus salve reserved address definition

After enabling the SMBus function, the module can automatically recognize SMBus host, SMBus Alert Response Address, SMBus Device Default Address, and give the corresponding flags and interrupt events.

18.7.2 I²C Initialization process

The I2C module must be properly initialized before performing I2C communication. It is recommended that the software follow these steps to perform the initialization operation.

- Clear the I2CSMBRST register of the RMU module to ensure that the I2C module is not in a reset state
- Set the I2CSMB_PCE register of the CMU module to enable the I2C module register bus interface clock
- Configure the I2CSMBCKS and I2CSMBCKE registers of the CMU module to select and enable the I2C operating clock
- Enable or disable analog filtering as required (SCL and SDA input analog filtering, >50ns)
- Enable or disable digital filtering as required

IO configuration

GPIOs used for I2C communication need to be configured for digital peripheral functionality. The IOs used for I2C communication must be configured in open-drain output mode.

Please consult the chip datasheet for more IO related information.

Host Baud Rate Configuration

The I2C master needs to configure the communication baud rate before enable, while the slave communication rate is determined by the SCL sent by the master, so no configuration is required.

BRGH and BRGL baud rate configuration registers are used to generate the communication baud rate. BRGH and BRGL are 9 bit baud rate division coefficients, and the baud rate calculation formula is as follows.

SCL cycle $T_{SCL}=T_{BRGH}+T_{BRGL}$

Where BRGH defines the SCL high level width and BRGL defines the SCL low level width

$T_{BRGH}=T_{I2CCLK}\times(BRGH+1)$

$T_{BRGL}=T_{I2CCLK}\times(BRGL+1)$

T_{I2CCLK} is I^2C operating clock cycle

For example, for 100k baud rate, T_{SCL}=10us; if the I2C operating clock is 8M, T_{I2CCLK}=125ns. Assuming that the duty cycle of SCL is required to be 50%, that is, the SCL high and low levels are 5us wide, according to the above formula, we can calculate BRGH=BRGL=39

Noise filtering

Both SDA and SCL inputs support noise filtering. The noise filtering consists of both analog and digital filtering.

The analog filter has a filter length of not less than 50ns, which meets the I2C protocol requirements for fast-mode and fast-mode plus modes to suppress at least 50ns of noise pulses. The software can enable or disable the analog filtering function through the ANFEN register, regardless of the master or slave mode.

The digital filtering function is separately enabled or disabled by the DFEN register, and both host and slave can enable digital filtering. After digital filtering is enabled, the SDA and SCL bus input to the module's internal signal will not change until the specified number of consecutive samples is consistent. This function can suppress burr signals of a certain width, and the filtering length is determined by DNF×T_{I2CCLK}.

Note: digital filtering enabled, the slave must enable I2CCLK, so it cannot be used in sleep mode. If the slave wishes to operate without I2CCLK, digital filtering must be disabled

18.7.3 I²C master function

The synchronous clock SCL is always provided by the master on the I²C bus, and the SDA data flow direction can be either master send and slave receive, or slave send and master receive. The I²C host function of this module supports the following features：

- Multi-master bus support
- Support for clock synchronization
- Support for clock arbitration
- 7-bit or 10-bit slave addressing
- Support for slave clock extension
- Programmable bus timing
- DMA support

18.7.3.1 7bit Addressing and Data Communication Flow

When 7bit addressing, the first byte sent by the master contains the slave address and the transfer direction bit ($R/\overline{W}$), depending on $R/\overline{W}$ the subsequent transfer is a master writing data to the slave ($R/\overline{W}$ =0) or a master reading data from the slave ($R/\overline{W}$ =1).

Name	Slave Address Byte							
Bit	7	6	5	4	3	2	1	0
Bit name	address							R/W

Bit description：

Bit	Bit name	Description
7-1	address	Slave device address
0	R/W	0: Write，send data (master) 1: Read， request data (slave)

Master writes data to slave

A typical frame structure for 7bit addressing, with the master writing data to the slave, is shown in

the diagram below.

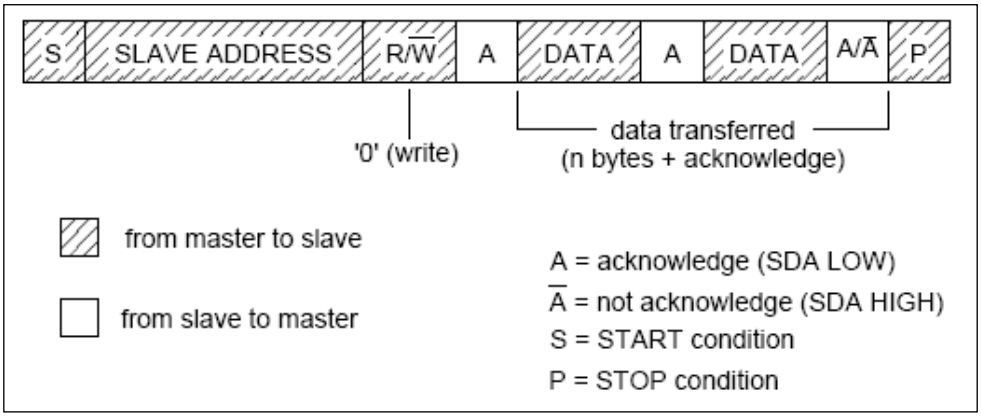


Figure 18-2 Frame format when a master writes data to a 7-bit address slave

1.The master clears the RWN register and initiates the START timing

2. Master sends slave address, slave address contains 7 bits of slave address and 1 bit of R/W flag bit which is 0 when sending data

3. The master sends the first 8-bit data frame

4. The master will determine if a valid ACK is detected at the 9th SCL after each 8-bit data is sent, if the master detects a positive ACK, it will continue to send next byte

5. If the slave cannot reply ACK, the master should send a STOP condition to terminate the transmission after detecting the NACK

6. After the master has finished sending all data, it will send the STOP condition

The software initiates the operation flow of the I^2C master send as follows：

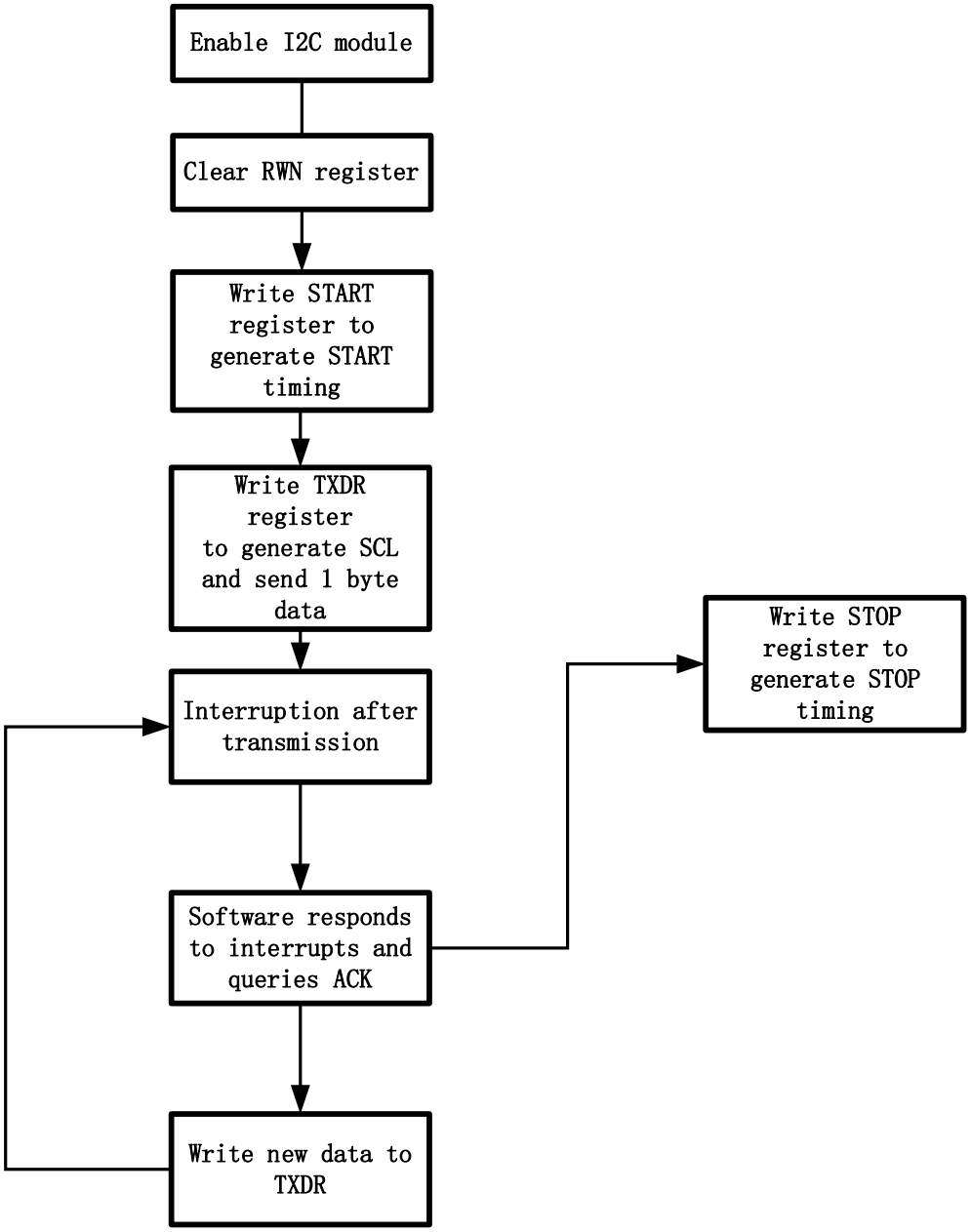


Figure 18-8 I²C master software sending data flow diagram

The waveform of an I²C host writing data to a 7-bit address slave is illustrated below：

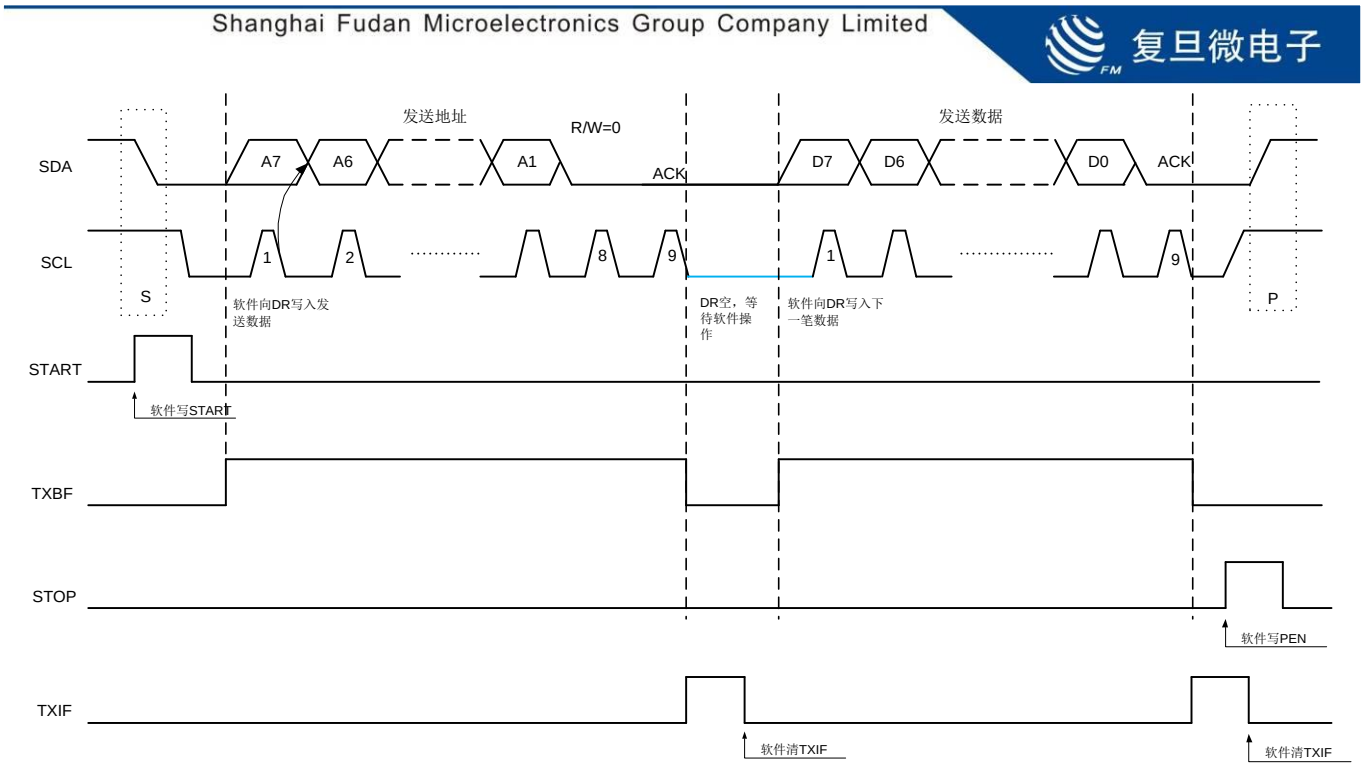


Figure 18-9 I²C master sends data flow diagram to 7-bit address slave

When the data register DR is empty, the hardware will keep SCL low and wait for the software to write the next data to DR. as shown in the blue part of SCL above.

Master reading data from a slave

A typical frame format for 7bit addressing, where the master reads data from the slave, is shown in the diagram below.

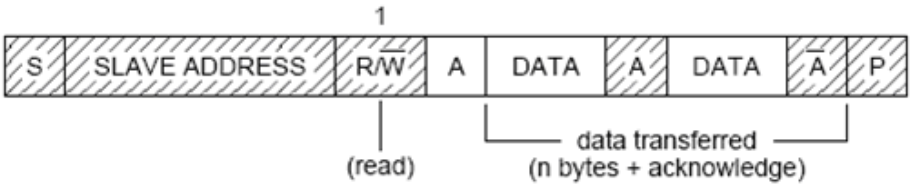


Figure 18-3 Frame format when a master reads data from a 7-bit addressing slave

1、 The master sets the RWN register and initiates the START timing

2、 The master sends the slave address, the slave address contains 7 bits of the slave address and 1 bit of the R/W flag bit which is 1 when the data is read

3、 Set MSPCON.RCEN to 1, the master automatically turn to receive state

4、 The master starts to receive the first byte of 8-bit data, and sends a valid ACK to the slave at the 9th SCL, so as to continue to read the next data byte

5、 After reading the last byte, the master sends a NACK to the slave at the 9th SCL

6、 The master sends STOP bit to terminate the reading

7、 The host sends STOP timing to terminate the read

The operational flow of I2C reception is shown in the following diagram：

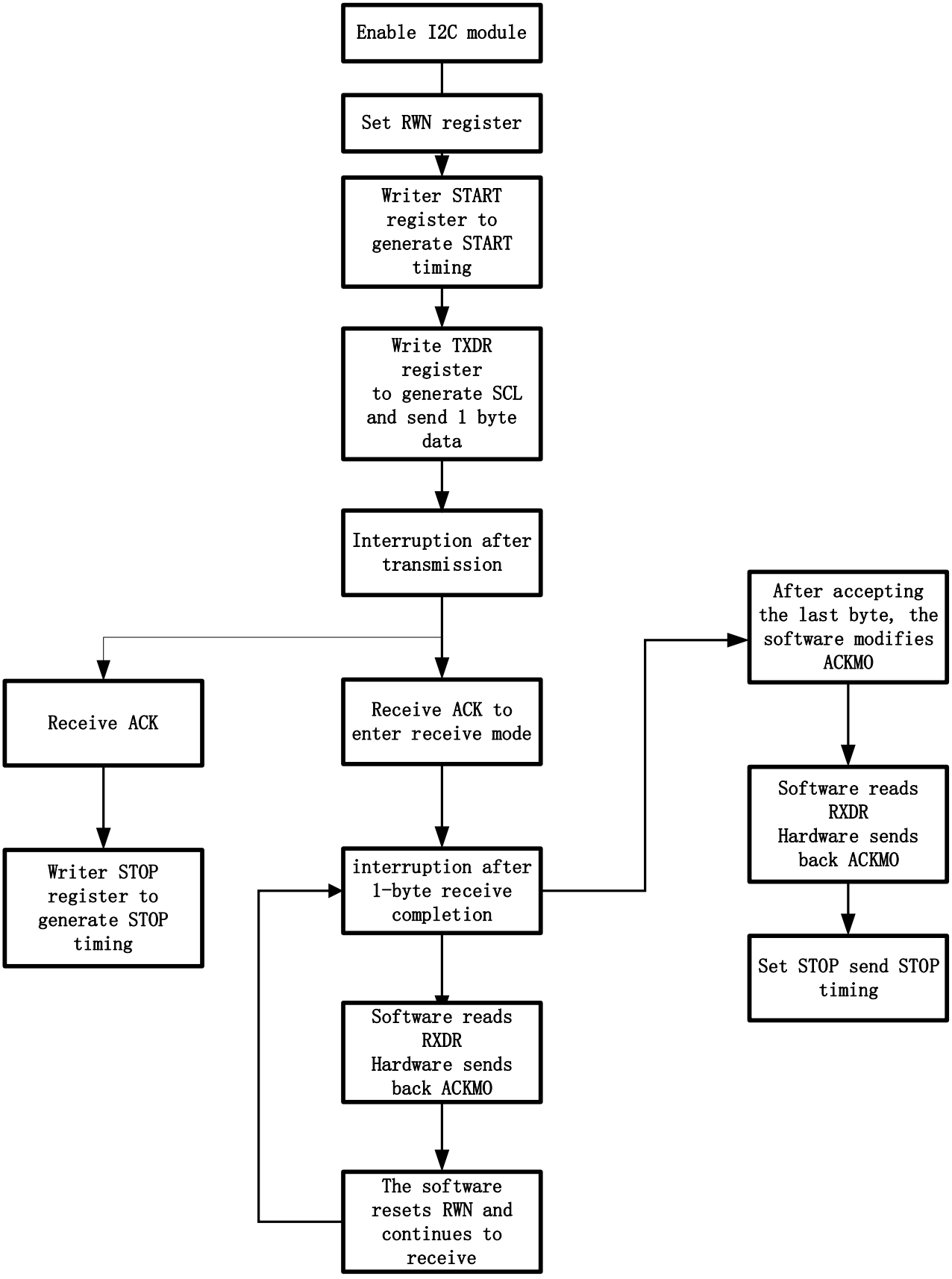


Figure 18-11 I²C software receive data flow diagram

The master sends back the response according to the ACKMO register after receiving the data sent by the slave each time. The ACKMO reset value is 0,by default the master replies ACK. If the software wants the master to reply a NACK, the ACKMO register needs to be rewritten to 1 when previous byte is received. ACKMO will be cleared automatically after the NACK is sent.

The waveform diagram of the I2C master reading data from rhe 7-bit address slave is as follows：

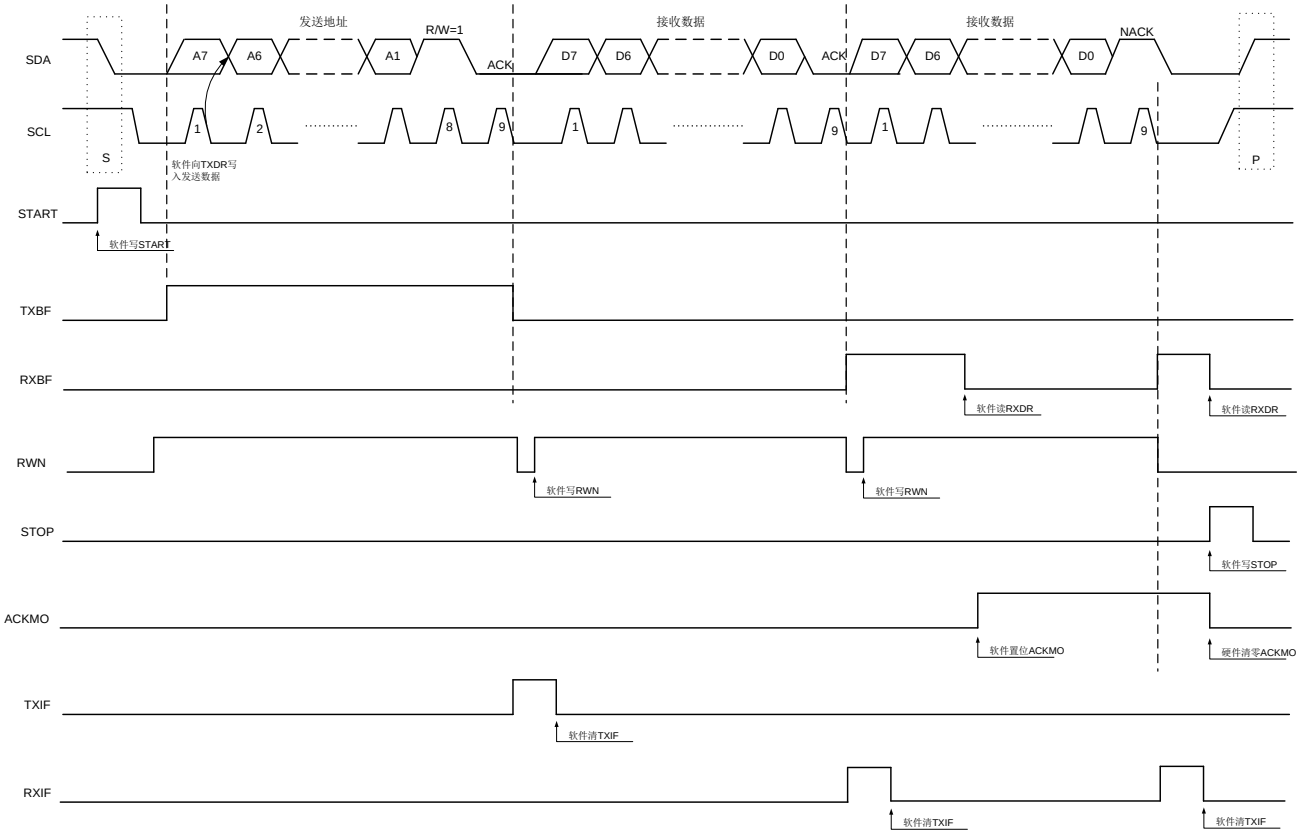


Figure 18-12 I²C reads data flow diagram from 7-bit address slave

During continuous reception by the host, the software reads the slave data by setting the RWN register. When the software does not write RWN, the master will keep waiting to avoid data reception overflow.

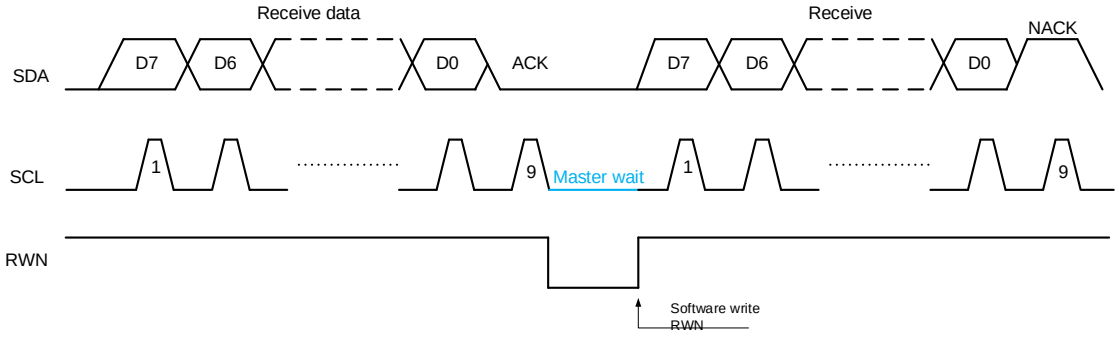


Figure 18-13 I²C master receiving data waiting Bidirectional Data Transfer (Combined Mode)

A typical bi-directional data read/write flow is shown in the diagram below. While the master is writing or reading data, the master can restart a new transaction by sending the Repeated Start condition, so the master can realize bidirectional communication within one transaction.

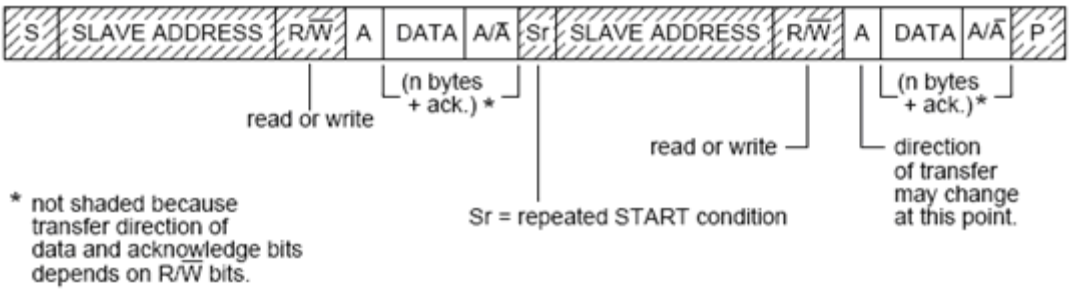


Figure 18-14 Frame format for bi-directional data communication

The software operation procedure for bidirectional communication is similar to that for unidirectional communication, except that the direction of transmission is modified by sending the ReSTART condition and slave address bytes.

18.7.3.2 10bit Addressing and Data Communication Flow

When 10bit addressing, the first byte sent by the master contains part of the slave address (11110_A9_A8) and the transfer direction bit ($R/\overline{W}$), the second byte contains the remaining slave address (A7~A0). After the two byte address have been sent, the data is then transferred.

Master writes data to the slave

A typical data flow for 10bit addressing, where the master writes data to the slave, is shown in the diagram below.

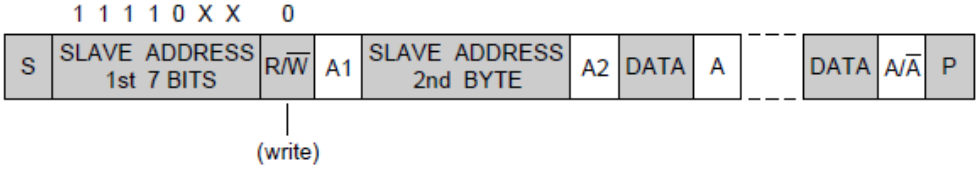


Figure 18-15 10bit addressing, the master writes data to the slave

1. The master clears the RWN register and initiates the START timing

2. The master sends the first slave address byte, starting with 11110, followed by the highest bit of the 2bit slave address, and the R/W flag bit, the R/W bit is 0 when sending data

3. The master checks the ACK sent back by the slave

4. The master sends the second slave address byte, containing the lower 8 bits of fhe slave address

5. The master checks the ACK replied by the slave

6. The master continues to write data to the slave

7. After the master has finished sending all data, it sends the STOP condition

The software procedure of the I²C master transmitting is as follows：

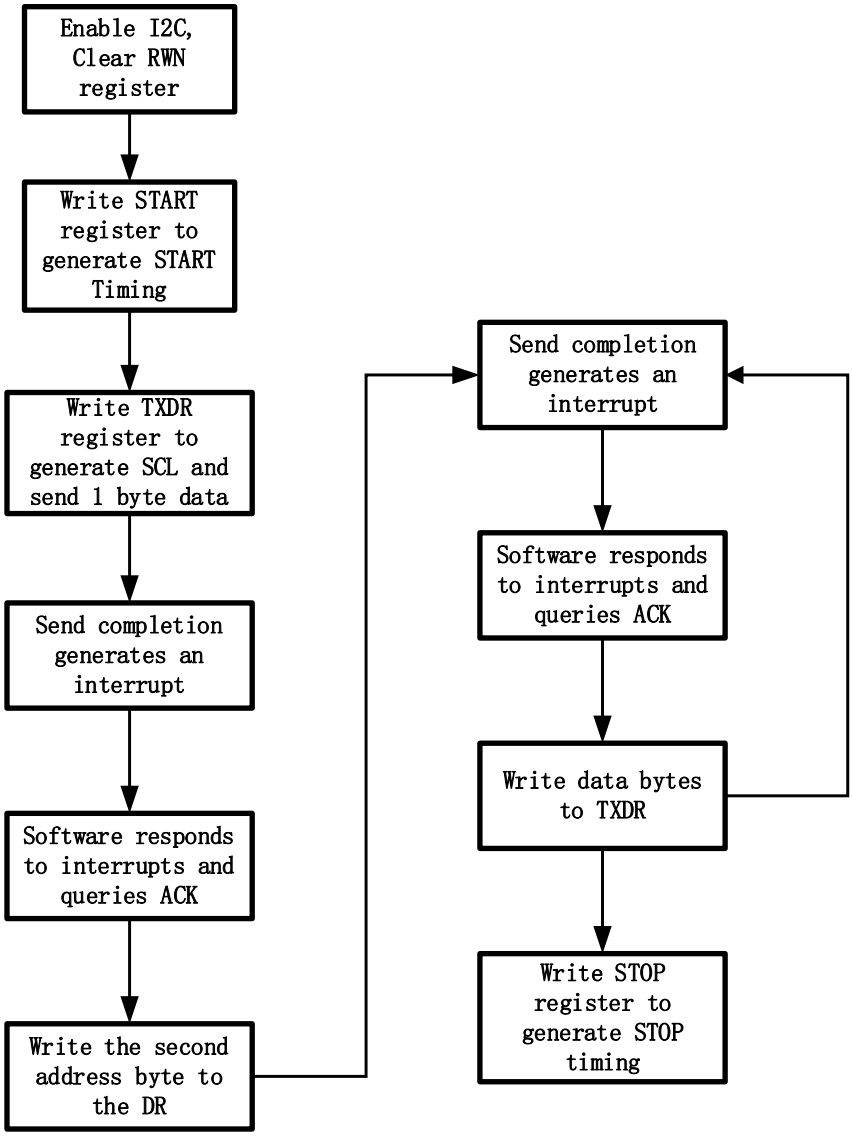


Figure 18-16 I²C software transmit flow diagram

Master reads data from the slave

A typical data flow for 10bit addressing, where the master reads data from the slave, is shown in the diagram belo

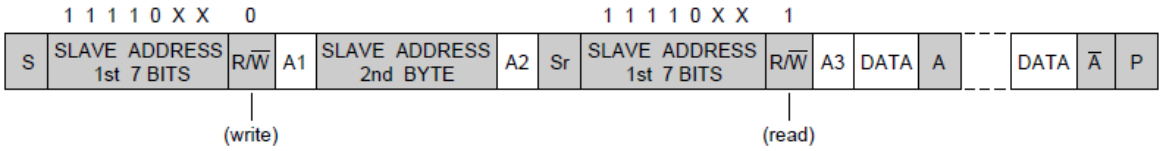


Figure 18-17 10bit addressing, master reads data from the slave

1. The master clears the RWN register and initiates the START timing

2. The master sends the first byte of the slave address, including 5 bits of the leading code 11110, 2 bits of the highest bit of the slave address and 1 bit of the R/W flag

3. The master sends the second byte of the slave address, including the low 8 bits of the address

4. The master sends ReSTART condition

5. Set the RWN register

6. The master sends the first byte of the slave address again, changing R/W to 0

7. The master automatically turns to the receiving state

8. The master starts to receive the first byte of 8-bit data, and sends a valid ACK to the slave at the 9th SCL, thus continuing to read the next data byte

9. After reading the last byte, the master sends a NACK to the slave at the 9th SCL

10. The master sends STOP condition

The software procedure for I^2C receive flow as follows：

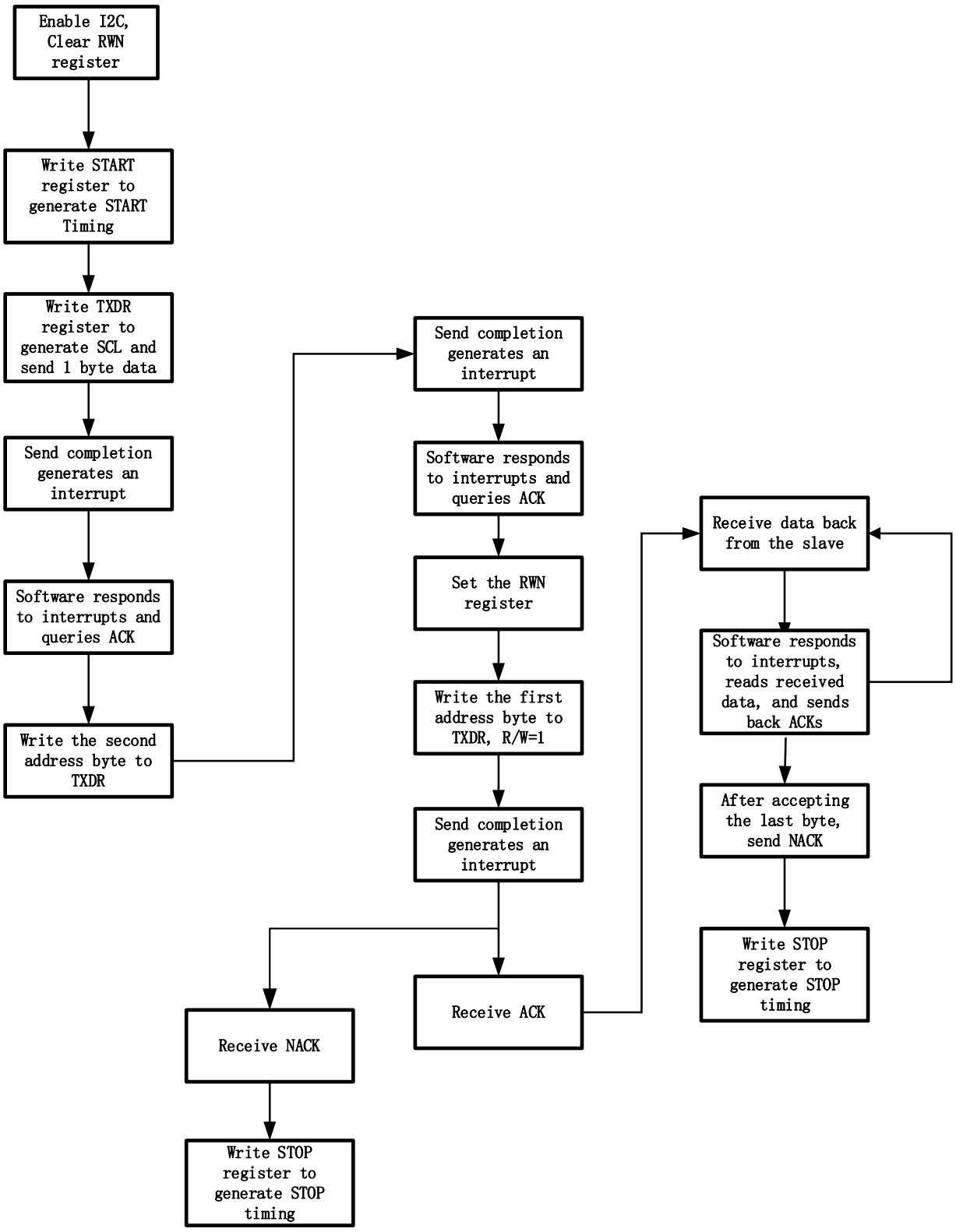


Figure 18-18 I²C software send data flow diagram

Bidirectional data transfer (combined mode)

A typical bi-directional data read/write flow is shown in the figure below. During a master write or

read, the master can restart a new transaction by sending Repeated Start condition, so the master can have bidirectional communication in one transaction.

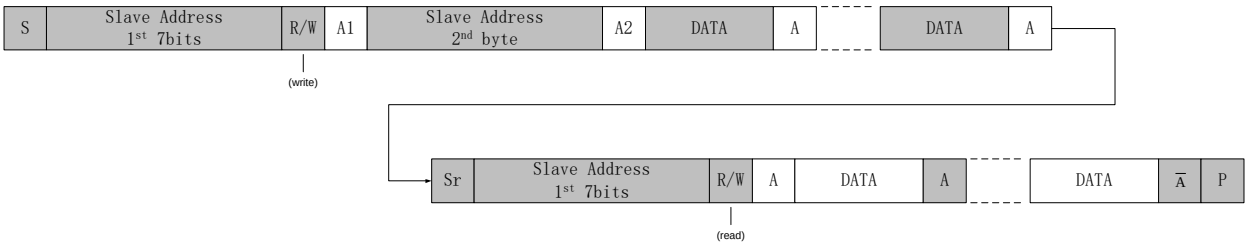


Figure 18-19 I²C software sending data flow diagram

The software procedure for combined transfers is similar to that for unidirectional transfers, except that the transfer direction is modified by sending the ReSTART condition and 1st slave address bytes.

18.7.3.3 DMA

The I²C master supports DMA. It should be noted that the bus clock (APBCLK) of the I²C module must be enabled in order to use the DMA function.

Master using DMA to write data to a slave

When the host sends data using DMA, all data including the slave address byte and send data need to be written to RAM in advance and sent out via DMA request. The software should configure the target DMA channel as I²C_TX beforehand.

In the case of DMAEN=1, START is set and if the data buffer register TXDR is empty, the I²C module will generate a DMA request and the DMA module responds to the request by writing the data to be sent in RAM to TXDR and start the data sending (the first byte is the slave address).In DMA send mode, I²C does not check the legality of the sent data and the software must ensure that The software must ensure that the data in the RAM is correct.

After each byte is sent, the I²C checks the slave ACK and generates a new DMA request if the ACK is correct, or generates a NACK interrupt if a NACK is received and no more DMA requests are generated.

When the DMA completes sending the specified length of data, the DMA transfer completion interrupt is generated. At this time, the STOP timing can be generated by software setting the STOP register, or the I²C hardware can automatically set the STOP register according to the DMA transfer completion signal to generate the STOP timing. The desired strategy can be selected by setting the AUTOEND register.

The flow of the host using DMA for transmitting is as follows.

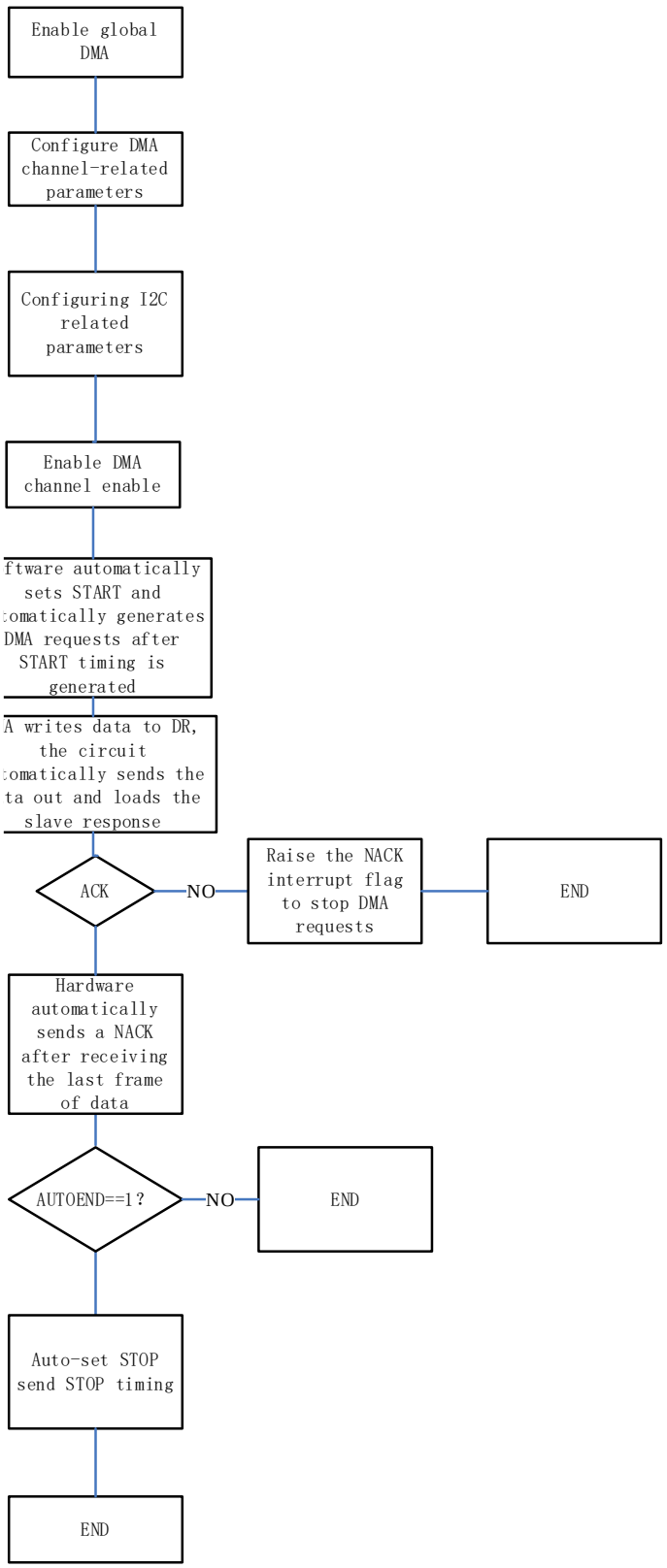


Figure 18-20 I²C master DMA transmit flowchart

The master uses DMA to read data from the slave

In this scenario, the slave addressing byte has to be sent by software. The software should configure the target DMA channel as I²C_RX beforehand.

The software first sets DMAEN=1 after sending the slave address, then enables the corresponding DMA channel as I²C_RX, the I²C automatically enters the receive mode and generates a DMA request after each byte is received, notifying the DMA to read the RXDR content and sending back an ACK to the slave at the same time.

When the DMA transfer reaches the specified length, the DMA's transfer complete flag will notify the I2C to send back NACK. subsequent STOP timing can be generated by software or hardware by setting the STOP register according to the AUTOEND register configuration.

Note: When the I²C host performs data reception via DMA, the number of bytes received by DMA will vary for different AUTOEND configurations and the same DMA transfer length (CHxTSIZE) configuration. When AUTOEND=0, the number of received bytes is CHxTSIZE+1; when AUTOEND=1, the number of received bytes is CHxTSIZE.

The flow of the master using DMA for reception is illustrated below：

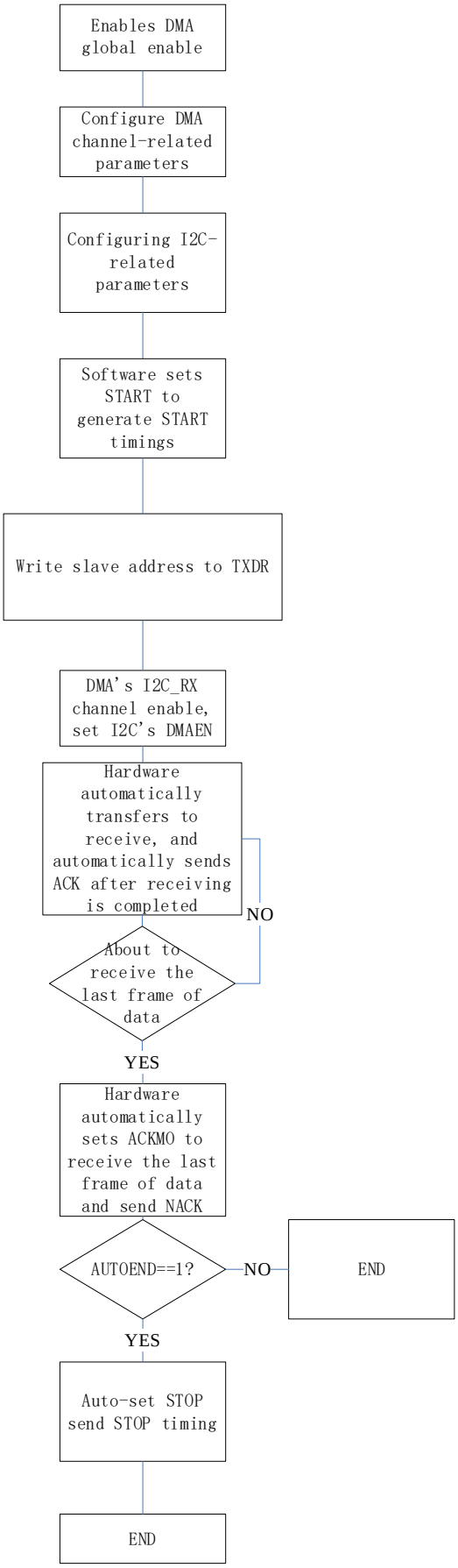

Enables DMA global enable
Configure DMA channel-related parameters
Configuring I2C-related parameters
Software sets START to generate START timings
Write slave address to TXDR
DMA's I2C_RX channel enable, set I2C's DMAEN
Hardware automatically transfers to receive, and automatically sends ACK after receiving is completed
NO
About to receive the last frame of data
YES
Hardware automatically sets ACKMO to receive the last frame of data and send NACK
AUTOEND==1?
NO
END
YES
Auto-set STOP send STOP timing
END

Figure 18-21 I²C master DMA receive flowchart

18.7.3.4 Slave Clock Stretching

I²C bus operation low-speed slaves notify the mastert to suspend data communication by pulling SCL low. the I²C master must support this feature, so at the start of each byte send/receive position, the master needs to automatically check the actual level of SCL on the bus after trying to send SCL high, if it is not high voltage, it means that the slave is doing SCL extension and the master will continuously monitor the SCL voltage until it is high. Until SCL is high, the subsequent operation will not start.

Note: The host will only perform SCL stretch check at the first SCL rising edge of each byte sent and received. If analog filtering or digital filtering is enabled, the SCL checked by the host is the filtered signal of the bus input.

18.7.3.5 Clock Synchronization

Clock synchronization is a synchronization mechanism of the I2C protocol for multi-master communication. When multiple masters start transmitting data on the idle bus at the same time, the clock synchronization mechanism ensures that the bus clocks will not be in a conflicting state.

When multiple hosts pull down SCL, the host with shorter SCL low time will find that other hosts are still driving SCL low when they try to pull up SCL, then the host should keep waiting without driving until SCL bus is pulled up, then start timing SCL high width again. With the line with approach, the low level of the synchronous clock on the bus is always determined by the host with the longest SCL low level length, while the SCL high level is determined by the master with the shortest high-level length.

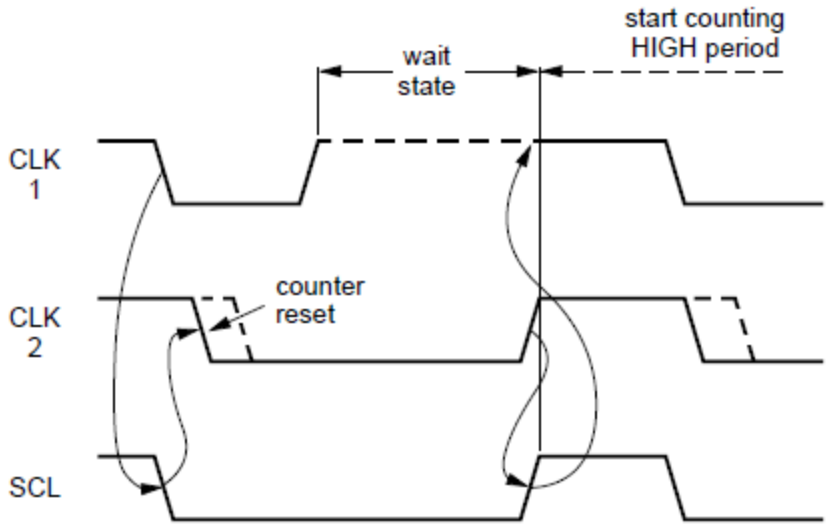


Figure 18-22 I²C Multi-master Clock Synchronization

18.7.3.6 Arbitration

Bus arbitration is also only applicable to multi-host buses. When multiple hosts send START conditions simultaneously within the START condition minimum hold time (tHD,STA), bus conflicts must be avoided and data communication integrity must be ensured by arbitration.

Arbitration is performed on a per data bit basis, and when SCL is high, each host should check in real time whether the SDA signal on the bus matches the level of the data it sends. Notice the wire-to-wire characteristics of the bus, so data conflicts will only be detected if this host sends a 1 and the SDA bus is a 0. At this time this host finds itself losing arbitration, it should take the initiative to turn off the SDA output drive logic to facilitate other hosts to continue normal communication, and if the module enables the slave mode at the same time, the module will immediately enter the slave receiving state.

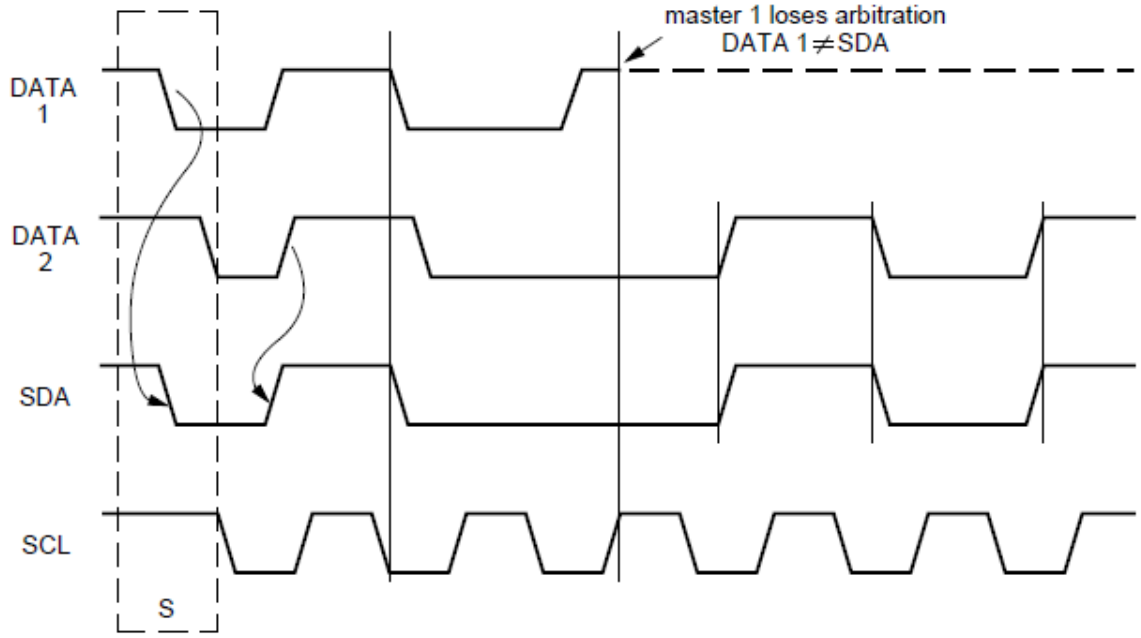


Figure 18-23 I2C Multi-master data arbitration

18.7.3.7 Timeout mechanism

The I²C master also implements a timeout function that generates an alarm interrupt and returns to IDLE status if SCL is stretched by slave.

When the master detects an SCL stretching, its internal timer starts counting. The maximum length of the SCL stretching timeout is 4096 SCL cycles. Assuming a baud rate of 100K, the timeout period is approximately 40ms. And if the baud rate is 400K, the timeout period is approximately 10ms.

The timeout period can be set by the software via the 12bit TIMEOUT register. The software must set the TIMEOUT register with MSPEN is 0. This reset value is 0xFFF, which means the maximum 4096*T_{SCL} timeout period. When the SCL stretching is detected, the TIMEOUT register starts to decrement. And when the counter reaches 0, the counter stops and the TIMEOUT register is reset to 0xFFF, and a timeout interrupt is triggered at the same time. Therefore, the timeout period can be set by modifying the initial value of TIMEOUT.

$T_{SCL\_STRETCHING\_TIMEOUT}$=TIMEOUT[11:0] * T_{SCL}

When a TIMEOUT interrupt occurs, it is recommended that the I²C module is reset by software.

This timeout function can be disabled. The software can also implement its own timeout function of any length by using the timer in combination with the SCL pin state polling.

18.7.3.8 Programmable timing

The I²C module provides flexible timing programming features that allow the user to define the low level width, high level width of the SCL clock, and the setup and hold time of the SDA data.

The BRGH/BRGL registers allow you to set the low and high widths of SCL, and the SDAHD registers allow you to configure the length of the hold and build hold time for SDA data relative to the SCL clock pulse.

SDAHD timing is equally valid for both SDA data bits and ACK/NACK.

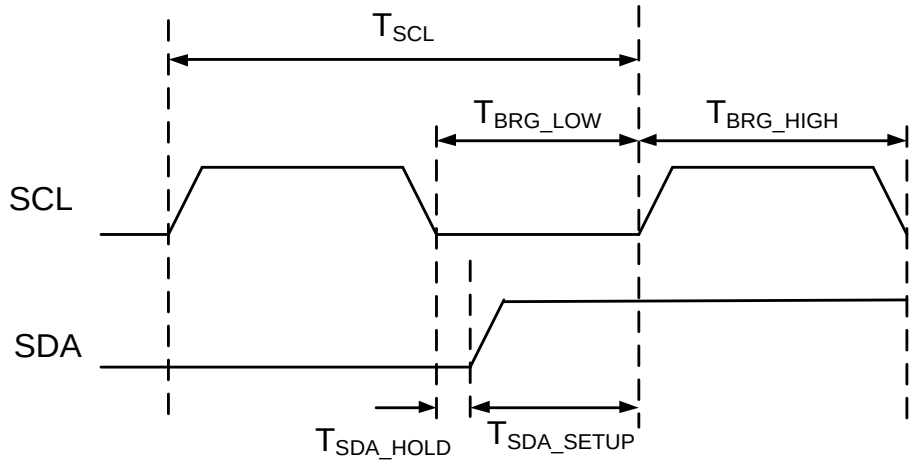


Figure 18-24 Master timing control

In the above figure, T_{SCL} is the communication baud rate and parameters can be defined by the following equation:

$T_{SCL} = T_{BRG\_LOW} + T_{BRG\_HIGH}$

$T_{SDA\_SETUP} = T_{BRG\_LOW} - T_{SDA\_HOLD}$

Note that the configuration of the MSPBGRH, MSPBRGL and SDAHD registers must meet the following requirements, as violation of these requirements will result in abnormal bus timings.

BRGH >= 2

BRGL >= 2

BRGL-1 >= SDAHD >= 1

TIMEOUT >= 1

18.7.4 I²C slave function

After the slave receives 1 byte of data, it generates an interrupt to notify the CPU to process the data. Before the CPU takes the data, the hardware can pull down SCL (software control is enabled) to notify the sender that it is busy, and the sender should suspend sending until SCL is released. If the receiver cannot respond to the ACK, the sender should send P to terminate the communication or send Sr to start a new communication after failing to detect the ACK.

After the slave sends 1 byte of data, an interrupt is generated to notify the CPU, and the hardware pulls down SCL to make the master wait. The CPU responds to the interrupt and prepares the next byte of data before releasing the SCL. The master continues to send SCL to make the slave continue to send data.

Slave Addressing

Depending on the A10EN register status, the slave can support either 7bit or 10bit addressing process. The slave address is defined by the SLAVE_ADDR register.

For 10bit slave address application, i.e., the case of A10EN=1, it is required that the first byte must start with 11110, otherwise the ADDR_ERROR error flag will be triggered. And in the case of A10EN=0, if the slave receives an address byte starting with 11110, the ADDR_ERROR error flag will also be set.

18.7.4.1 Slave input filtering

The slave SCL and SDA inputs support both analog and digital filtering. When using digital filtering, I2CCLK must be enabled; if only analog filtering is used, I2CCLK can work without it.

If you want the I2C slave to continue to work in sleep mode, you cannot enable the digital filtering function, otherwise the I2C module will not be able to detect the bus level change because the digital filtering circuit is not working.

18.7.4.2 Slave send data

Recommended operation process：

- The slave receives the address byte (R/W=1), sends back ACK, generates address match interrupt
- Since R/W=1, the hardware automatically performs SCL stretching, and the slave enters the sending state

- The software responds to the interrupt, queries the R/W flag, and confirms that it is sent by the slave
- The software writes the data to be sent into TXDR
- The hardware automatically releases the SCL
- New SCL arrives, SDA starts output
- The new SCL is coming, SSPBUF is shifted and output to the SDA bus
- Receive ACK and generate transmission completion interrupt

The following figure is a typical schematic diagram of slave data sending waveform：

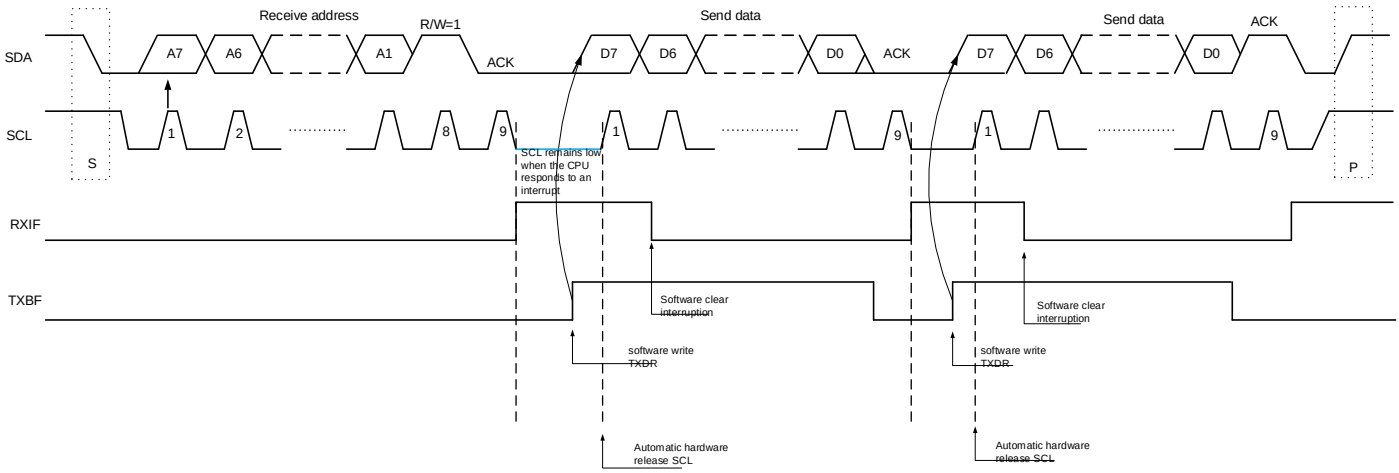


Figure 18-25 I²C Slave data sending waveform

In the slave transmit process, when the slave receives the correct address, the ADM flag is set and the address byte is written to RXDR. hardware automatically pulls down the SCL signal and waits for software to write TXDR. when software writes TXDR, TXBF is set and hardware releases SCL at the same time.

18.7.4.3 Slave receive data

Recommended operation process：

- The slave receives the address byte (R/W=0) and sends back an ACK, generating an address matching interrupt
- Since R/W=0, the hardware automatically performs SCL extension and the slave maintains the receive state

- Software responds to the interrupt, queries the R/W flag, and confirms slave reception
- Software reads RXDR hardware automatically releases SCL and starts receiving data
- Host data byte arrives, hardware sets RXBF flag after byte reception is complete
- Slave sends back ACK and generates receive completion interrupt
- Hardware automatically performs SCL extension (SCLSEN=1)
- Software responds to interrupt, reads RXDR, hardware automatically clears RXBF flag
- Hardware automatically releases SCL
- Repeat data reception process until STOP timing is received, or software sets ACKEN to 0

The following figure is a typical schematic diagram of slave data receiving waveform (SCLSEN=1)：

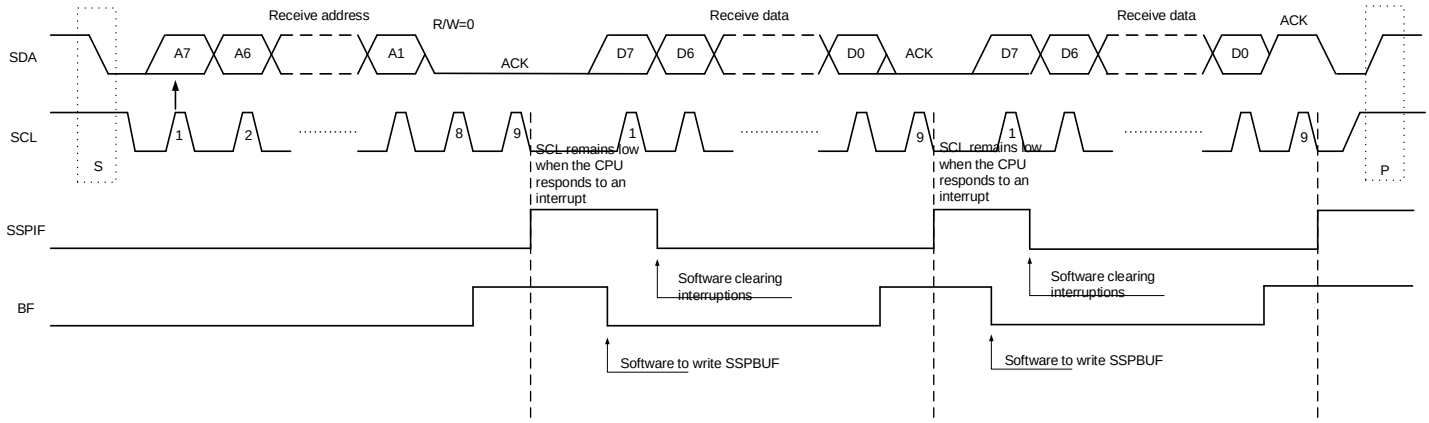


Figure 18-26 I²C Slave data receiving waveform

During the slave receive process, the slave first receives the address byte, and if the address matches, the ADM flag is set, the address byte will be written to RXDR and set the RXBF flag, and then the hardware pulls down the SCL. when the software reads RXDR, the RXBF flag is automatically cleared and the hardware releases the SCL for subsequent data reception.

Note: The address byte is written to RXDR during the slave receive process and causes RXBF to be set. Software needs to read RXDR to clear RXBF and release SCL.

Slave receive data can passively end communication or actively end communication.
If the host actively sends a STOP, the slave passively ends this communication. Or, if the software clears the ACKEN register in the interrupt handler, the slave will send back NACK after receiving the next byte, and the host will send STOP to end this communication after receiving NACK.

Slave SCL stretching

I²C slaves enable SCL stretching (slave clock stretching) by default, but software can disable this

feature (SCLSEN register) to accommodate hosts that do not support slave SCL stretching.

When SCL stretching is enabled, the software can only clear the RXBF flag when the receive buffer is read during SCL stretching after data reception is complete. If data overflow occurs during reception, the RXOV flag is set and the hardware sends back a NACK and the SCL is no longer extended so that the host can issue a STOP; if RXOV is set, it is recommended that the software wait for the STOP flag to be set before reading the receive buffer to clear the BF flag.

Receive data overflow

When the slave receive buffer is full (BF=1), if new data is received again, a receive overflow occurs and the RXOV flag is set. The old data in the receive buffer will be overwritten by the new data. Receive data overflow can only occur if the slave has closed the SCL extension function.

The following diagram shows the occurrence of a data receive overflow if SCLSEN=0:

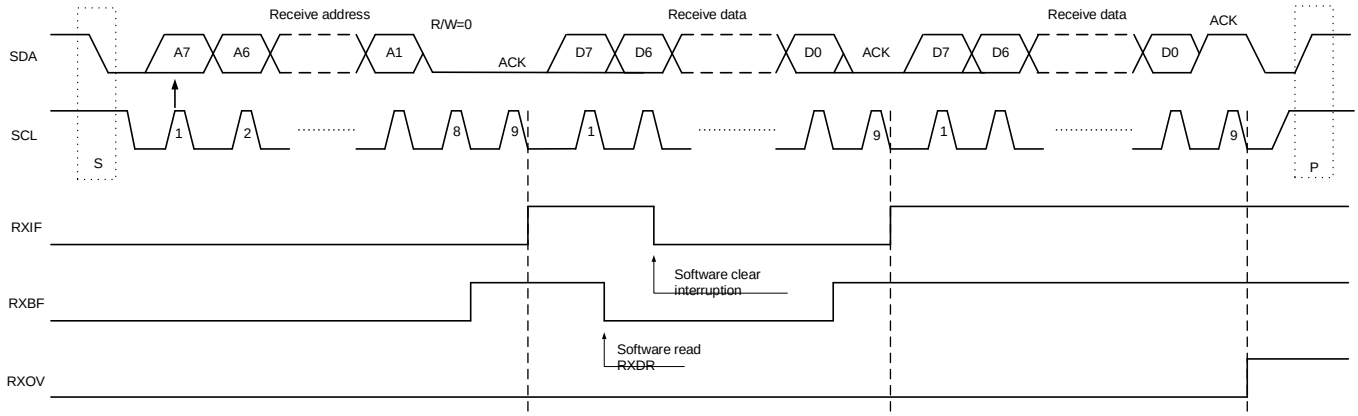


Figure 18-27 I²C Slave data receiving waveform (SCLSEN=0, receiving overflow)

18.7.4.4 Slave low-power receiving wake-up

When the chip is in sleep mode, the I2C slave supports START timing wake-up and address matching wake-up.

- START timing wake-up RCHF clock for slave address reception
- Slave address matching wakes up the MCU for subsequent

The software can also choose to wake up the MCU directly when START timing is detected, and if START is configured to wake up only the RCHF clock, the slave automatically turns on the RCHF when it detects a START event so that the module can complete address byte reception and timeout detection.

If the clock wakes up and finds an address mismatch, the I2C slave does not wake up the MCU,

but automatically turns off the RCHF.

Note: When using the I2C slave wakeup from hibernation function, you must configure I2CCLK to RCHF and turn off the digital filtering function.

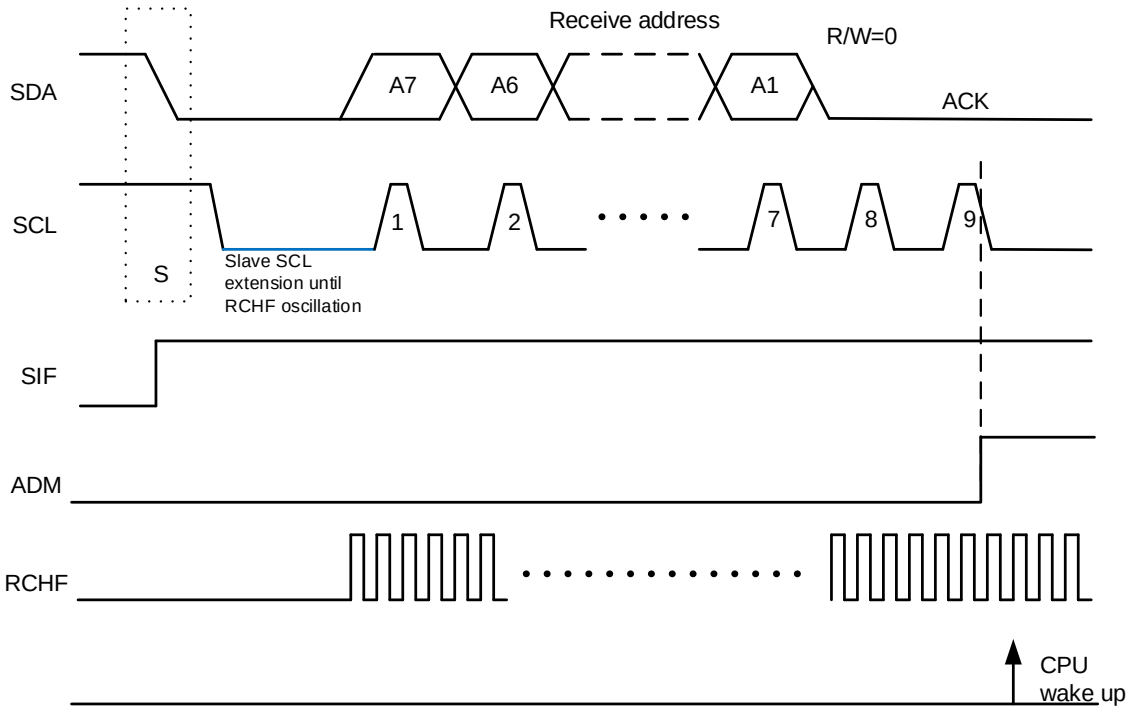


Figure 18-28 Slave address matching wake-up

18.7.4.5 DMA

The I²C slave supports DMA. It should be noted that the bus clock (APBCLK) of the I²C module must be enabled in order to perform the DMA operation. The bus clock is used to generate DMA requests and receive DMA responses.

The slave uses DMA to receive data

When the I²C slave receives the correct address, it generates the ADM interrupt flag. After the software responds to the interrupt, it queries the received R/W bit. If it is 0, the master is ready to write data to the slave. At this time, the software can configure the specific DMA channel as I2C_RX and enable the DMAEN of the I²C slave; then every time the slave completes a byte reception, a DMA request will be generated and the DMA will be notified to read the SSPBUF.

There are two possibilities to end DMA slave reception:

1) The data transfer length has not reached the DMA length configuration, and the master has issued a STOP sequence, the software should respond to the STOP interrupt and actively handle this situation;

2) The data transfer length reaches the DMA length configuration, but because the DMA request is generated after the slave sends back ACK, the software should respond to the DMA transfer completion interrupt and clear ACKEN to zero, so that the slave will send back NACK to end this communication after receiving the next byte.

The flow of receiving by the slave using DMA is as follows：

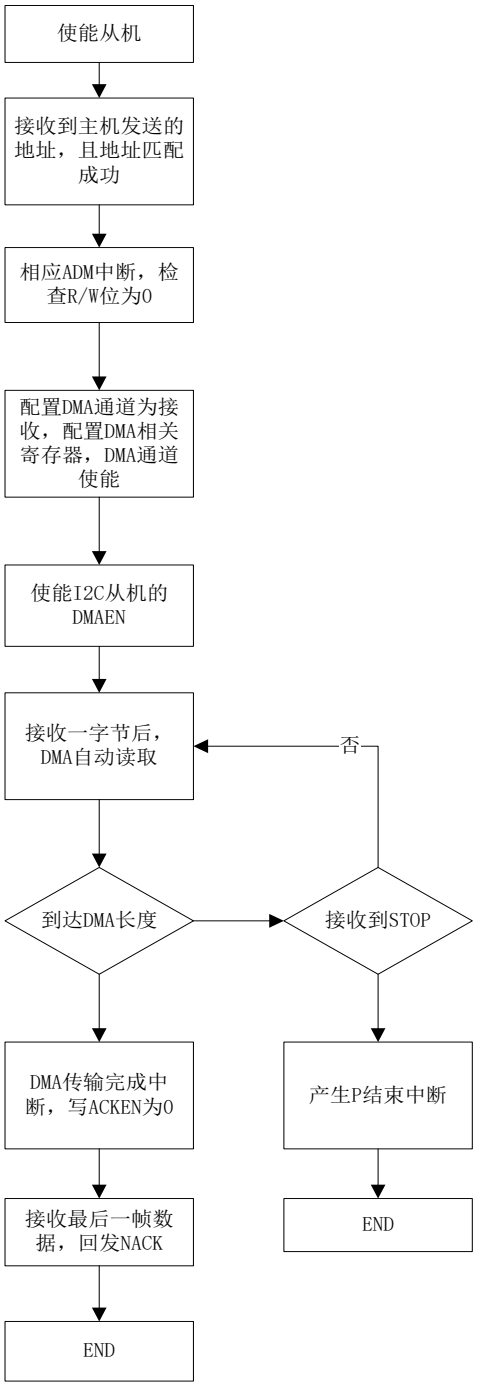


Figure 18-29 I²C slave DMA receiving flowchart

Slave sends data using DMA

When the I2C slave receives the correct address, the ADM interrupt flag is generated and the software responds to the interrupt by querying the received R/W bit, if it is 1 it means the host is ready to read data from the slave. At this point the software needs to configure the specific DMA channel as I2C_TX and enable the DMAEN of the I2C slave; subsequently when the slave data cache TXDR is empty, a DMA request will be generated to notify the DMA to write to TXDR.

Only the host sends back a NACK to end the read operation. When the read data length is greater than the transfer length set by the DMA, the slave will keep pulling the SCL low until the software turns off the I2C slave module since the DMA no longer responds to I2C requests.

The flow of the slave using DMA for transmitting is as follows:

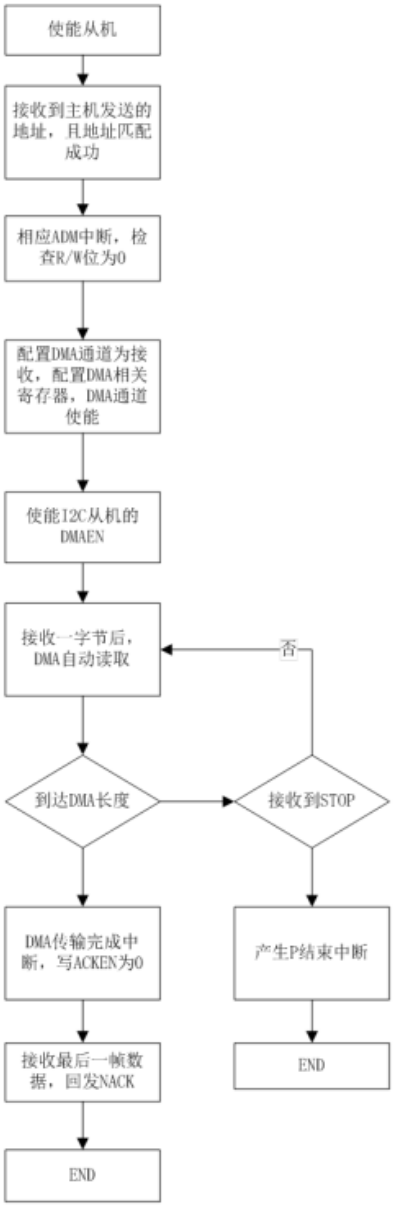


Figure 18-30 I2C slave DMA send flow

18.7.4.6 SDA output delay

Since the data transmission and reception of the slave only uses SCL, some analog delay is needed to realize the data establishment and retention time control of SDA, and the timing of SCL is completely controlled by the master.

The timing control of the slave is shown in the figure below. According to I^2C protocol requirements, the minimum data retention time of SDA relative to the falling edge of SCL is 0ns, that is, the slave can use the falling edge of SCL to send data to meet the requirements. However, considering the actual fall time of the SCL waveform on the bus, in order to better cover the retention time requirements, an extra RC delay greater than 300ns is added to the SDA output. This delay only needs to be applied to the SDA output of the I^2C slave (SSP_SDAO).

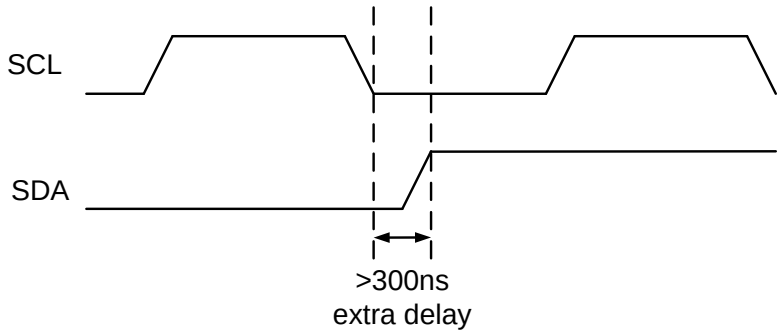


Figure 18-31 SDA output delay waveform

18.8 SMBus Function Description

18.8.1 Overview

SMBus is a two-wire synchronous communication protocol based on the I2C physical layer that can be used for system control and power management tasks and supports dynamic address assignment for hot-plug applications. This module supports SMBus rev3.0. For more details about SMBus protocol, please refer to http://smbus.org

SMBus specifies three types of devices：

- Slave: receives commands sent by the host and gives a response
- Master: sends commands and clocks, and can start or terminate communication
- Host: A special kind of host that can provide an interface to the system CPU. host must be able to act as a host and a slave and support host notify protocol. only one host is allowed to exist in the whole system.

This module can support slave, host, and host modes.

The SMBus protocol layer is a command-based operation. See the SMBus Specification for the set of commands defined by the protocol.

Address resolution protocol (ARP)

The ARP protocol enables SMBus to dynamically assign addresses to slaves, thus enabling "hot-plugging" of the system. The peripheral module supports the SMBus default slave address (device default address: 1100 001) recognition, which is enabled by setting the SMBDEN register.

In order to implement ARP, the slave software needs to maintain a 128-bit SMBus system unique ID (UDID); when the slave receives the device default address, the interrupt flag is set to notify the software that the system host is conducting ARP enumeration protocol, and the software should complete the ARP protocol process to achieve dynamic slave address assignment.

Host Notify protocol

This module supports host notify protocol, when SMBHEN register is set, the peripheral can respond to SMBus host address (0001 000) and can communicate as system host and host slave at the same time.

SMBus Alert

SMBus Alert is an optional signal that allows an SMBus slave to request communication from the host by pulling down SMBus Alert. host receives the SMBALERT# signal and initiates access to all SMBALERT# capable slaves via Alert Response Address. Only slaves with low SMBALERT# will

respond to Alert Response Address addressing.

In SMBus slave mode, software can pull low the SMBA pin by setting the ALERT register when ALERTEN=1. When the slave receives the ARA and recognizes its own address, the slave should clear the ALERT register to release the SMBALERT# signal.

In SMBus Host mode, if the ALERTEN register is set, when a falling edge on the SMBA pin is detected, the ALERT flag register is set and the corresponding interrupt event can be generated.

Packet Error Checking (PEC)

SMBus packet error detection is implemented through CRC-8 error detection codes. PEC is optional for SMBus devices, but is required for devices that need to support ARP.

The error detection code is appended at the end of the data communication, and CRC-8 calculations are applied to all message bytes, including address and RW_ bits, excluding ACK/NACK, START, STOP, and ReSTART.

The SMBus slave sends back a NACK to indicate a PEC error when a PEC error is detected during reception. However, since the ACK/NACK bits themselves may also have bus transmission errors, the master should not consider the slave sending back ACKs as correct PECs. The module can support sending back or not sending back PECs in slave mode.

The CRC-8 polynomial used for PEC: C(x) = x8 + x2 + x1 + 1

Timeouts

This module contains a set of hardware timers used to implement the three timeout types specified by the SMBus protocol. When the module detects a timeout event, it needs to reset the state machine, stop data sending and receiving, restore the IO default no-driver pull-up state, and get ready to receive a new START event.

Timeout detects only the clock signal, and the timeout counter starts working when the bus clock is pulled low.

Symbol	Parameters	limits		unit
		min	max	
$t_{TIMEOUT}$	SCL Low level timeout	25	35	ms
$t_{LOW:SEXT}$	Cumulative SCL low extension time (slave)	-	25	ms
$t_{LOW:MEXT}$	Cumulative SCL low level extension time (master)	-	10	ms

Table 18-4 SMBus Timeout Events

1. tLOW:SEXT is the cumulative clock pull-down time allowed for a slave in one communication (START-to-STOP)

2. tLOW:MEXT is the cumulative clock pull-down time allowed by a host within each byte (START-to-ACK, ACK-to-ACK, ACK-to-STOP)

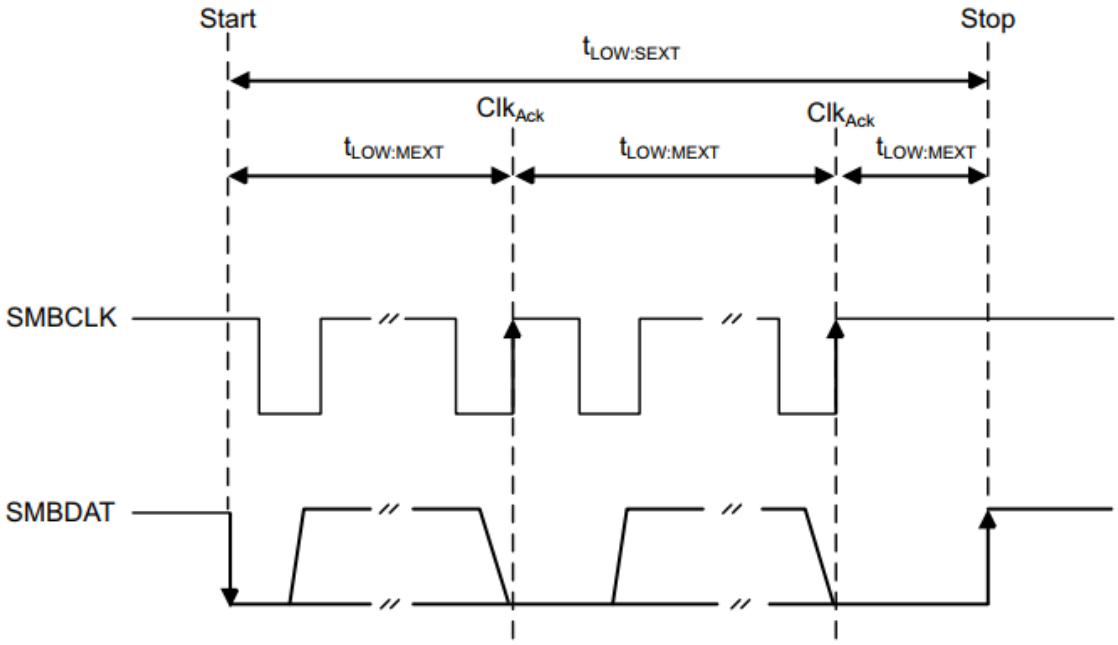


Figure 18-32 tLOW:SEXT and tLOW:MEXT waveforms

Bus IDLE Detection

The SMBus host considers the bus to be idle when it detects that the bus clock and data are held high beyond tHIGH,MAX and can initiate communication.

This module supports hardware bus IDLE detection. tIDLE judgment time is defined by TIMEOUT register.

18.8.2 SMBus initialization process

In addition to the I2C-related initialization, the SMBus special initialization process contains the following.

Special address (slave mode)

- SMBus Device Default address (1100 001) enabled via SMBDEN register
- SMBus Host address (0001 000) enabled by SMBHEN register
- Alert Response address (0001 100) enabled by ALERTEN register

Packet Error Checking

The PEC check byte is calculated by the module's own CRC-8 function.

If CRCEN is set while the SMBus is sending and receiving data, the hardware will synchronize the received data bits into CRC-8 for serial shift calculation.

In the SMBus slave receive mode, the software should control the ACK or NACK back to the PEC byte corresponding to the PEC comparison result.

Timeout detection

The timeout detection is implemented using the TIMEOUTx hardware timer. Where TIMEOUTA is a 12bit timer, TIMEOUTA is used for bus SCL low timeout detection when the TIDLE register is cleared to zero. When the SCL pull-down time exceeds (TIMEOUTA+1)×2048×tI2CCLK, the bus timeout flag register is set in hardware and an interrupt event can be generated.

Another TIMEOUTB 12bit timer is used for tLOW:SEXT and tLOW:MEXT timeout detection. When the accumulated SCL stretch time exceeds (TIMEOUTB+1)×2048×tI2CCLK, the clock stretch timeout flag register is set in hardware and an interrupt event can be generated.

Bus IDLE detection

When the TIDLE register is set, TIMEOUTA is used for bus IDLE detection. If both SCL and SDA on the bus remain high for more than (TIMEOUTA+1)×4×tI2CCLK, the bus idle flag register is set in hardware and an interrupt event can be generated.

18.8.3 SMBus Master Function

18.8.3.1 SMBus Host sending

The SMBus host can send data either by software or by DMA.

In order to support the PEC function, the I2C_SMBUS module comes with a CRC-8 check unit. With CRC enabled (CRCEN=1), the CRC-8 unit automatically completes the CRC calculation of the sent data. The software needs to read the CRC result of the sent data from the CRCDR and send it out after the data is sent.

For DMA sending, there are two operation methods：

- Turn off the AUTOEND function, after the DMA sends all the data bytes, then the software reads the PEC bytes from CRCDR and sends them out
- Pre-calculate the CRC result for the data block to be sent and save it immediately after the data block; AUTOEND can be enabled when the DMA sends, and the length of sending is the data block length + 1 byte CRC

18.8.3.2 SMBus host reception

SMBus host reception can be performed by software or by DMA.

When using software to receive data, the workflow is similar to I2C host reception, the main difference is the PEC function. The CRC calculation during reception has two options：

- CRCEN=0: Software handles the CRC calculation, and for every 1 byte of data received, software is responsible for writing the data to the peripheral CRC module for calculation. After finishing all data reception, the software reads the CRC calculation result and compares it with the last received PEC byte.
- CRCEN=1: The SMBus module automatically performs CRC calculation internally, calling the module's internal CRC-8 unit, and synchronously updates the CRC-8 result for every 1 byte of data received

For the receive mode with CRCEN=1, since the hardware cannot automatically distinguish between data bytes and CRC bytes, the software should close CRCEN and read the CRC result in time after receiving the last byte of data and before starting to receive CRC bytes. If the software cannot handle it in time, the CRC byte reception will change the data checksum result.

When receiving data using DMA, if CRC is enabled, the I2C_SMBUS module automatically completes the CRC calculation of the received data and saves the result in the CRC register. In

this case, to avoid CRC byte reception overwriting the data CRC result, the DMA receive length should be configured as data byte length without CRC byte.

18.8.3.3 SMBus Slave send

SMBus slave sends data flow similar to I2C slave, after setting CRCEN, PEC is calculated by CRC-8 unit and PEC bytes are sent by software.

When DMA sends, there are two operation methods：

- Turn off the AUTOEND function, and after the DMA sends all the data bytes, then the software reads the PEC bytes from the CRCDR and sends them
- Pre-calculate the CRC result for the data block to be sent, and save it immediately after the data block; AUTOEND can be enabled when DMA sends, and the send length is the data block length + 1 byte CRC

18.8.3.4 SMBus Slave Receive

SMBus slave receive data flow is similar to I2C slave, there are two ways to calculate PEC, refer to SMBus host receive chapter.

18.8.3.5 SMBus slave wake-up

In low-power mode, SMBus can wake up the MCU when a START event is received, or it can wake up only the RCHF clock for slave address reception. If configured to wake up the MCU, the slave will wake up the MCU from hibernation mode when it detects a START event. If configured to wake up only the RCHF clock, the slave automatically turns on the RCHF when it detects a START event so that the module can complete address byte reception and timeout detection.

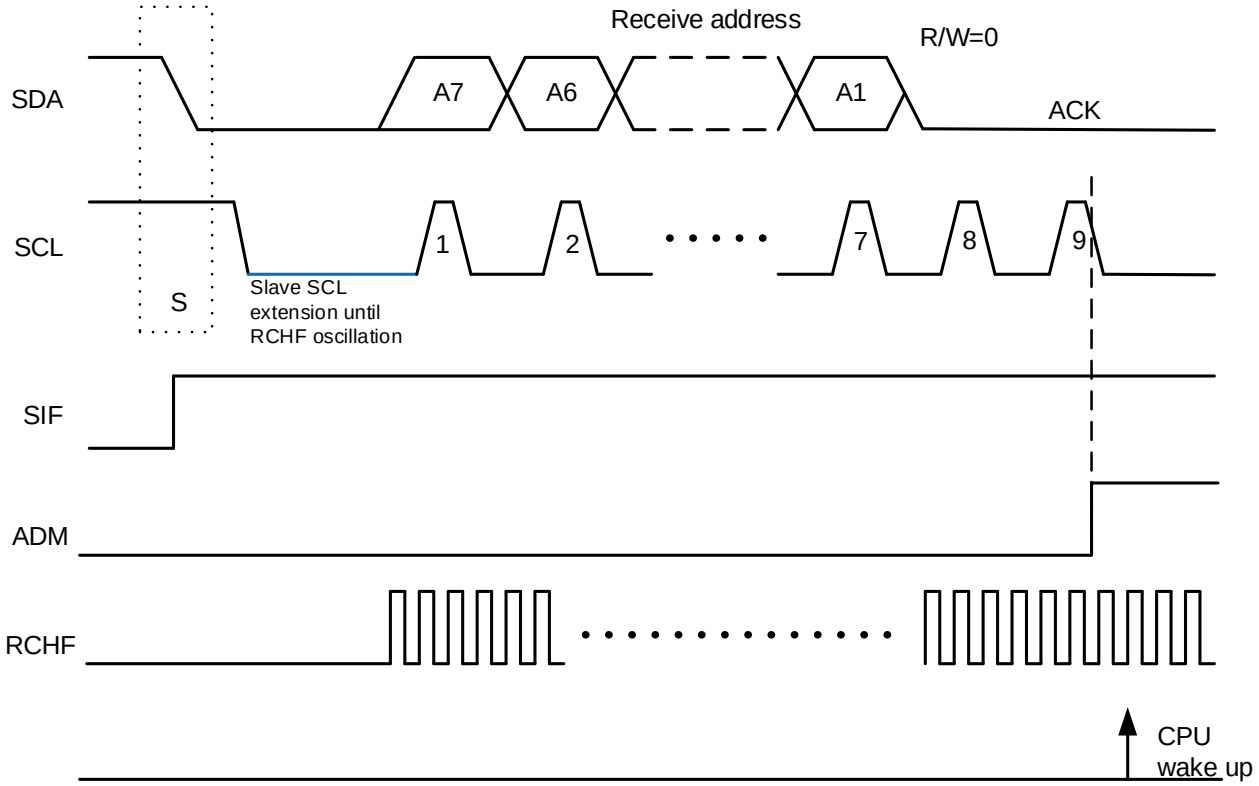


Figure 18-33 Slave address matching wake-up

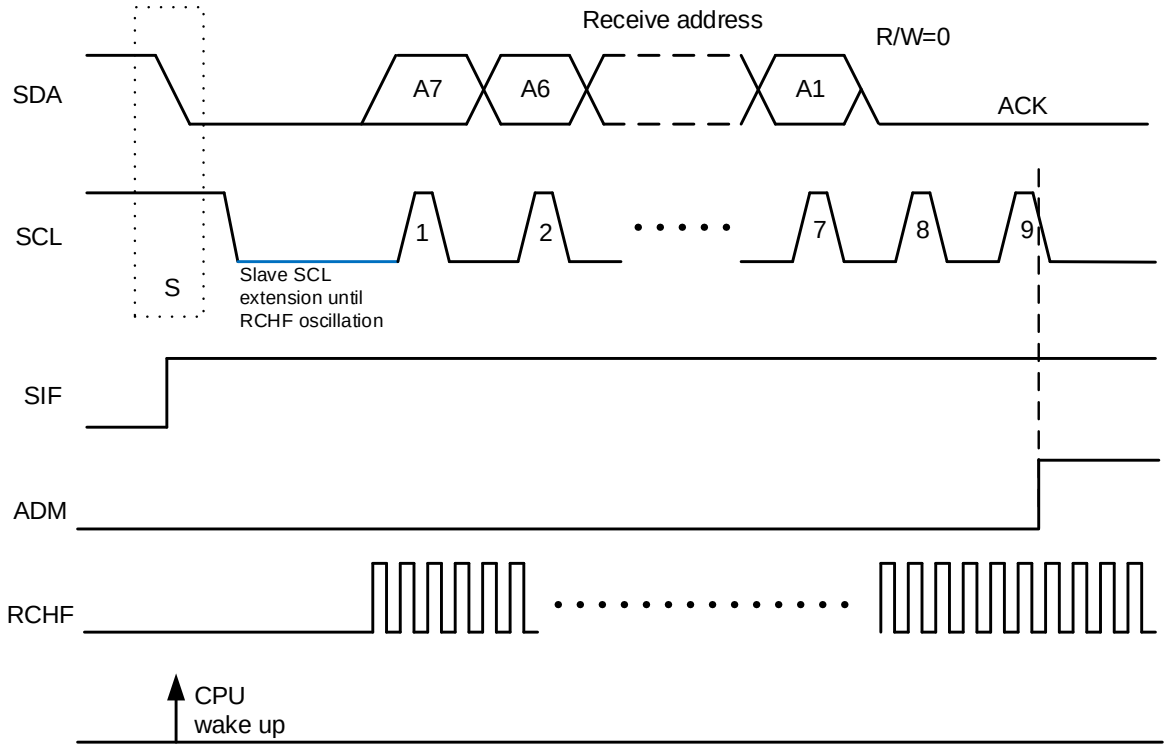


Figure 18-34 Slave START wake-up

Note: When using the SMBus slave wake-up function, the I2CCLK must be configured as RCHF

18.8.4 SMBus Timeout Function

Both the host and slave modes of SMBus support tTIMEOUT detection and generate a timeout interrupt when a timeout event is detected.

The following figure shows the definition of the bus clock timeout：

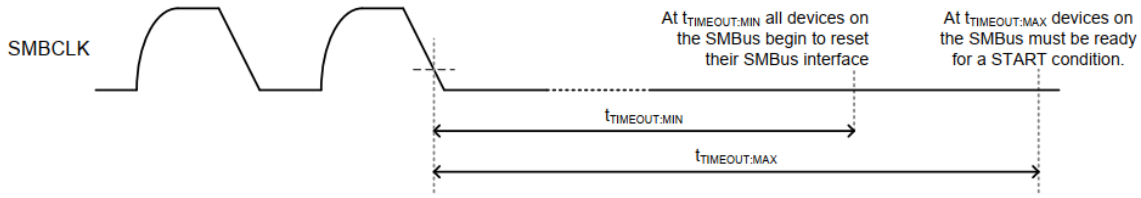


Figure 18-35 SMBus bus clock timeout event

The TIMEOUTA register is used to define the timeout length, and the timeout is defined by the following equation：

(TIMEOUTA+1)×2048×t_{I2CCLK}

The bus timeout range specified by SMBus spec is 25~35ms, and the software can calculate a reasonable TIMEOUTA parameter according to the application.

When the TOEN register is set, the timeout counter starts to operate when the SCL low level is detected for the first time and automatically clears the counter every time the clock high level is detected. If the timeout counter overflows, the timeout interrupt register is set, the timeout counter is automatically turned off, and TOEN is cleared until the next software start-up.

The SMBus host supports tLOW:SEXT detection. When the host detects that the slave has pulled the clock low for a total time longer than the protocol specified limit, the SMBus host generates a timeout interrupt and the software can choose to abort the current transmission and send STOP.

The SMBus slave supports tLOW:MEXT detection. When the slave detects that the total time the master has pulled down the clock exceeds the protocol limit, the SMBus slave generates a timeout interrupt and the subsequent action policy should be defined by the software.

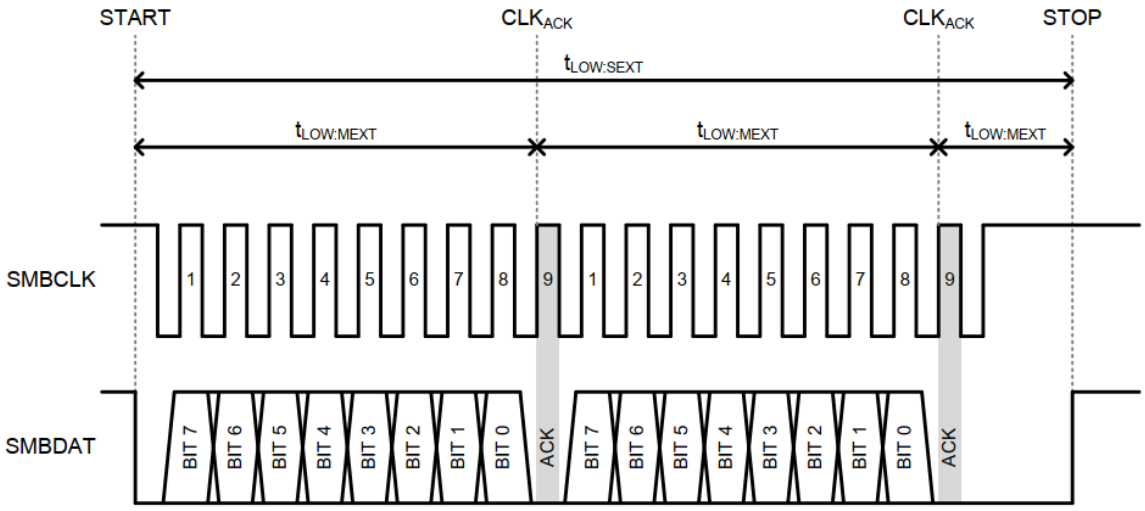


Figure 18-36 Host and slave timeout events

18.8.5 CRC-8

The I2C_SMBUS module comes with its own CRC-8 checksum unit, implemented by serial shift, with configurable initial values and polynomials.

The following Golden data supply is used for testing and verification in the application.

Polynomial	Input	sequence Initial value (hex)		
		All 0	All F	6363
		CRC calculation result (hexadecimal)		
CRC-8	5A5A	0F	D8	C5
	1223344	F9	28	96

The CRC function channel sets the CRCEN register to enable, and then saves the calculation result in the CRCDR register after the communication is finished.

18.9 Register

Offset	Name	Symbol
I2C_SMB （Module Base Address：0x4001 2800）		
0x00000000	I2C SMBus Control Register	I2CSMB_CR1
0x00000004	I2C SMBus Control Register	I2CSMB_CR2
0x00000008	I2C SMBus Intterupt Enable Register	I2CSMB_IER
0x0000000C	I2C Master Interrupt Status Register	I2CSMB_ISR
0x00000010	I2C_SMB Baud Rate Setting Register	I2CSMB_BGR
0x00000014	I2C_SMB Host Timing Control Register	I2CSMB_TIMINGR
0x00000018	I2C_SMB Timeout Register	I2CSMB_TOR
0x0000001C	I2C_SMB Receive Data Register	I2CSMB_RXDR
0x00000020	I2C_SMB transmit data register	I2CSMB_TXDR
0x00000024	I2C_SMB slave address register	I2CSMB_SADR
0x00000028	CRC Data Register	I2CSMB_CRC_DR
0x0000002C	LFSR register	I2CSMB_CRC_LFSR
0x00000030	CRC polynomial register	I2CSMB_CRC_POLY

18.9.1 I2C_SMB control register 1（I2CSMB_CR1）

NAME	I2CSMB_CR1							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	AUTOEND	ALERTEN	SMBDEN	SMBHEN	GCEN	WKUPEN		SCLSEN
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-00		R/W-1
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	RXDMAEN	TXDMAEN	-	ANFEN	DNF			
access	R/W-0	R/W-0	U-0	R/W-0	R/W-000			
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				CRCEN	A10EN	-	EN
access	U-0				R/W-0	R/W-0	U-0	R/W-0

bit	name	functional description
31:24	--	RFU：Reserved, read as 0
23	AUTOEND	Host DMA automatic transfer end (DMA automatic transfer end) 1: After the DMA specified length transfer is completed, the STOP timing is automatically sent 0: Wait for software to take over after DMA transfer of specified length is completed SMBus Alert pin enable

bit	name	functional description
22	ALERTEN	1: Enable Alert function device mode can pull the Alert pin low by writing the ALERT register, and can respond to Alert Response Address 0001100x (ACK) The host mode can respond to Alert signal 0: disable Alert function
21	SMBDEN	SMBus device default address enable 1: Response SMBus device default address 1100001x (ACK) 0: Do not respond to SMBus device default address 1100001x (NACK)
20	SMBHEN	SMBus host address enable 1: Respond to host address 0001 000x (ACK) 0: Do not respond to host address 0001 000x (NACK)
19	GCEN	General Call response enable 1: Respond to general call 0000 0000 (ACK) 0: Do not respond to general call 0000 0000 (NACK)
18:17	WKUPEN	Hibernation Wakeup Enable 00: Disable the slave sleep wakeup function 01: START timing wakes up the clock without waking up the MCU 10: START timing wakes up the clock and MCU at the same time 11: START timing wakes up the clock, address matching wakes up the MCU
16	SCLSEN	Slave clock stretching enable, valid only in slave mode (slave SCL stretching enable) 1: Enable clock stretching 0: disable clock stretching
15	RXDMAEN	Receiver DMA Enable 1: Enable the data receive DMA function 0: Disable the data receive DMA function
14	TXDMAEN	Transmitter DMA Enable 1: Enable the data transmitter DMA function 0: disable the data transmit DMA function
13	--	RFU：Reserved, read as 0
12	ANFEN	Analog Noise Filter Enable 1: Enable bus input analog filtering 0: Disable analog filtering
11:8	DNF	Digital Noise Filter control (Digital Noise Filter) 0000: Disable digital filtering 0001: Digital filter length 1 I2CCLK 0010: Digital filter length 2 I2CCLK ..... 1111: Digital filter length 15 I2CCLK
7:4	--	RFU：Reserved, read as 0
3	CRCEN	CRC calculation enable 1: Enable CRC calculation function 0: disable CRC calculation
2	A10EN	10bit address enable, valid only in slave mode 1: Slave uses 10bit address 0: Slave use 7bit address
1	--	RFU：Reserved, read as 0
0	EN	I2C_SMB module enable control bit (module enable) 1: Module enable 0: Module disable

18.9.2 I2C_SMB control register 2（I2CSMB_CR2）

NAME	I2CSMB_CR2							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-		ALERT	ACKEN	RWN	STOP	RESTART	START
access	U-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:6	--	RFU：Reserved, read as 0
5	ALERT	SMBus Alert pin control register, valid only when ALERTEN=1 1: SMBus module pulls Alert pin low 0: SMBus module withdraws Alert pin
4	ACKEN	Whether to respond to ACK signal in host or slave receive mode (ACK output enable) 1: send back NACK 0: send back ACK
3	RWN	Host read/write direction control register, this bit defines the host transmission direction, should be consistent with bit0 of Address byte, only valid for host mode. 1: Host read 0: Host write Note: When the host receives, the hardware will automatically clear RWN after each byte of data is received, so the software needs to reset RWN after each byte is received continuously
2	STOP	STOP timing generate enable control bit, software write 1 to send STOP timing, hardware will automatically clear zero after sending (Stop Enable)
1	RESTART	Repeated START timing generation enable control bit, software write 1 to send Repeated START timing, hardware auto-zero after sending (Repeated Start Enable)
0	START	START timing generation enable control bit, software write 1 to send START timing, hardware auto-zero after sending (Start Enable)

18.9.3 I2C_SMB Interrupt Control Register（I2CSMB_IER）

NAME	I2CSMB_IER							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24

name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	IDLIE	ERRIE	SIE	PIE	NACKIE	ADIE	TXIE	RXIE
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:8	--	RFU：Reserved, read as 0
7	IDLIE	Bus Idle interrupt enable register (Bus Idle interrupt enable) 1: Allow bus idle interrupt 0：Disable bus idle interrupt
6	ERRIE	Error event interrupt enable 1: Allow error interrupt 0: disable error interrupt *When ERRIE is set, the following events can trigger CPU interrupt* *Arbitration failure* *Bus error* *Data overflow* *Timeout event* *Alert detection*
5	SIE	START event interrupt enable 1: Allow START event interrupt 0: Disable START event interrupt
4	PIE	STOP event interrupt enable 1: Allow STOP event interrupt enable 0: disable STOP event interrupt
3	NACKIE	Non-ACK interrupt enable register in host transmit mode 1: Allow the interrupt generated by NACK 0: disable NACK interrupt generation
2	ADIE	Slave address match interrupt enable 1: Allow slave address match interrupt generation 0: disable slave address match interrupt
1	TXIE	Transmit finished interrupt enable 1: Allow transmit finished interrupt enable 0: disable transmit finished interrupt
0	RXIE	Receive finished interrupt enable 1: Allow receive finished interrupt enable 0: disable receive finished interrupt enable

18.9.4 I2C_SMB Status And Flag Registers（I2CSMB_ISR）

NAME	I2CSMB_ISR							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24

name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							SDIR
access	U-0							R-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-	-	ALIF	TOB	TOA	OVF	BERR	ARLO
access	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	BUSY	IDLIF	SIF	PIF	NACKIF	ADIF	TXIF	RXIF
access	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:17	--	RFU：Reserved, read as 0
16	SDIR	Slave transfer direction, valid only in slave mode, read-only; hardware updates after slave address match 0: host write operation, slave receive 1: host read operation, slave transmit
15:14	--	RFU：Reserved, read as 0
13	ALIF	Alert flag, valid in SMBus host mode, hardware set when Alert pin pulled low is detected, software write 1 clear, write 0 invalid
12	TOB	Timeout B flag, hardware set when tLOW:MEXT or tLOW:SEXT timeout occurs, software write 1 clears, write 0 is invalid
11	TOA	Timeout A flag, hardware set when SCL timeout occurs, software write 1 clears, write 0 is invalid
10	OVF	Data overflow flag, when RXDR or TXDR is not empty, a new data arrival, trigger hardware set, software write 1 clear, write 0 invalid (Overflow Flag)
9	BERR	Bus error flag, indicates that a wrong START or STOP timing is detected during transmission, hardware set, software write 1 clears, write 0 is invalid (Bus Error)
8	ARLO	Arbitration failure flag, set by hardware, cleared by software write 1, invalid by write 0 (Arbitration Lost)
7	BUSY	Bus Busy flag, read-only 1: Bus is in communication 0: Bus is idle
6	IDLIF	SMBus bus IDLE event flag, hardware set, software write 1 cleared, write 0 invalid (SMBus IDLE)
5	SIF	START interrupt flag, valid for slave mode. Hardware set, software write 1 clear, write 0 invalid (START event interrupt flag)
4	PIF	STOP interrupt flag, valid for slave mode. Hardware set, software write 1 cleared (STOP event interrupt flag)
3	NACKIF	NACK interrupt flag register in host/slave transmit mode, hardware set, software write 1 clear, write 0 invalid (Non-ACK interrupt flag)
2	ADIF	Slave address match interrupt flag, hardware set, software write 1 to clear, write 0 to invalid (address match interrupt flag)
1	TXIF	Transmit finished interrupt flag, hardware set, software write 1 clear, write 0 invalid (transmit finished interrupt flag)
0	RXIF	Receive finished interrupt flag, hardware set, software write 1 clear, write 0 invalid (receive finished interrupt flag)

18.9.5 I2C_SMB Baud Rate Setting Register（I2CSMB_BGR）

NAME	I2CSMB_BGR							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							BRGH[8]
access	U-0							R/W-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	BRGH[7:0]							
access	R/W-00010011							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							BRGL[8]
access	U-0							R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	BRGL[7:0]							
access	R/W-00010011							

bit	name	functional description
31:25	--	RFU：Reserved, read as 0
24:16	BRGH	SCL clock high level width sent by the host, counted by the I2C operating clock (Master SCL High level length)
15:9	--	RFU：Reserved, read as 0
8:0	BRGL	The width of SCL clock low level sent by the host, counted by I2C operating clock (Master SCL Low level length)

18.9.6 I2C_SMB Host Timing Control Register（I2CSMB_TIMINGR）

NAME	I2CSMB_TIMINGR							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							SDAHD[8]
access	U-0							R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SDAHD[7:0]							

access	R/W-00001010

bit	name	functional description
31:9	--	RFU：Reserved, read as 0
8:0	SDAHD	Define the SDA hold time parameter with respect to the falling edge of SCL, counted by the I2C operating clock (SDA hold delay) Note: The minimum valid value is 1 and the maximum valid value is BRGL

18.9.7 I2C_SMB Timeout Register（I2CSMB_TOR）

NAME	I2CSMB_TOR							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	TEXTEN	-			TIMEOUTB			
access	R/W-0	U-0			R/W-0000			
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	TIMEOUTB							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	TOEN	-		TIDLE	TIMEOUTA			
access	R/W-0	U-0		R/W-0	R/W-0000			
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TIMEOUTA							
access	R/W-0000 0000							

bit	name	functional description
31	TEXTEN	Cumulative extended clock timeout enable 1: Detect tLOW:MEXT timeout in master mode and tLOW:SEXT timeout in slave mode 0: disable extended clock timeout detection
30:28	--	RFU：Reserved, read as 0
27:16	TIMEOUTB	Used to define the cumulative extended clock timeout threshold tLOW:MEXT defined in host mode, tLOW:SEXT defined in slave mode tLOW:EXT=(TIMEOUTB+1)×2048×tI2CCLK Note: It is prohibited to overwrite this register if TEXTEN=1
15	TOEN	Clock timeout detection enable 1: SCL low timeout detection enable: tTIMEOUT is detected when TIDLE=0, tIDLE is detected when TIDLE=1, the threshold value is defined by TIMEOUTA 0: SCL low timeout detection disable
14:13	--	RFU：Reserved, read as 0
12	TIDLE	Bus idle detection enable 1: TIMEOUTA is used to detect bus idle 0: TIMEOUTA is used to detect SCL low level timeout
11:0	TIMEOUTA	TIDLE=0 for detecting SCL timeout tTIMEOUT=(TIMEOUTA+1)×2048×tI2CCLK

		TIDLE=1 is used to detect bus idle tIDLE=(TIMEOUTA+1)×4×tI2CCLK Note: It is prohibited to overwrite this register if TOEN=1

18.9.8 I2C_SMB Receive Data Register（I2CSMB_RXDR）

NAME	I2CSMB_RXDR							
Offset	0x0000001C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RXDATA							
access	R-0000 0000							

bit	name	functional description
31:8	--	RFU：Reserved, read as 0
7:0	RXDATA	8bit receive data, read only

18.9.9 I2C_SMB Transmit Data Register（I2CSMB_TXDR）

NAME	I2CSMB_TXDR							
Offset	0x00000020							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TXDATA							
access	R/W-0000 0000							

bit	name	functional description
31:8	--	RFU：Reserved, read as 0

7:0	TXDATA	8bit receive data, read only

18.9.10 I2C_SMB Slave Address Register（I2CSMB_SADR）

NAME	I2CSMB_SADR							
Offset	0x00000024							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	SAEN	MASK			-		ADDR	
access	R/W-0	R/W-000			U-0		R/W-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ADDR							
access	R/W-0000 0000							

bit	name	functional description
31:16	--	RFU：Reserved, read as 0
15	SAEN	Slave address enable 1: Enable slave address matching, ACK correct address 0: disable slave address matching, NACK any address
14:12	MASK	Slave address mask 000: no mask 001: mask ADDR[0], i.e. ADDR[0] does not participate in address comparison 010: mask ADDR[1:0], that is, ADDR[1:0] does not participate in address comparison 011: mask ADDR[2:0], i.e. ADDR[2:0] does not participate in address comparison 100: mask ADDR[3:0], i.e. ADDR[3:0] does not participate in address comparison 101: mask ADDR[4:0], i.e. ADDR[4:0] does not participate in address comparison 110: mask ADDR[5:0], that is, ADDR[5:0] does not participate in address comparison 111: mask ADDR[6:0], i.e. ADDR[6:0] does not participate in address comparison
11:10	--	RFU：Reserved, read as 0
9:0	ADDR	10bit slave address, when A10EN=1, all 10bit addresses are valid, when A10EN=0, only the low 7bit address is valid

18.9.11 CRC Data Register（I2CSMB_CRC_DR）

NAME	CRC_DR

Offset	0x00000028							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DR[7:0]							
access	R -1111 1111							

bit	name	functional description
31:8	--	RFU：Reserved, read as 0
7:0	**DR**	CRCDR is used to save the result of CRC calculation after the operation is finished. (CRC Data Register)

18.9.12 CRC LFSR Register（I2CSMB_CRC_LFSR）

NAME	CRC_LFSR							
Offset	0x0000002C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	LFSR[7:0]							
access	R/W-1111 1111							

bit	name	functional description
31:8	--	RFU：Reserved, read as 0
7:0	**LFSR**	CRC Linear Feedback Shift Register Initial CRC value can be written by software before the operation starts

18.9.13 CRC Polynomial Register（I2CSMB_CRC_POLY）

NAME	CRC_POLY							
Offset	0x00000030							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	POLY[7:0]							
access	R/W-0010 0001							

bit	name	functional description
31:8	--	RFU：Reserved, read as 0
7:0	**POLY**	CRC polynomial coefficients (CRC Polynominals)

19 I²C

19.1 Introduction

I2C (inter-integrated circuit) module serves as an interface between the MCU and the serial I2C bus. I2C module works as master or slave, but multi-master mode is not supported.

I2C main features:

- 1x independent I²C interface
- Support master mode and slave mode, does not support multi-master mode
- Support 7bit or 10bit slave address
- Transfer speed supports standard mode(100Kbps), fast mode(400Kbps) and Fm+(1Mbps)
- Support DMA, independent DMA channel of master and slave
- Low-power slave design, can send and receive data without system clock
- Support asynchronous slave address matching wake-up, data frame receiving completion wake-up or START detection wake-up

19.2 I²C block diagram

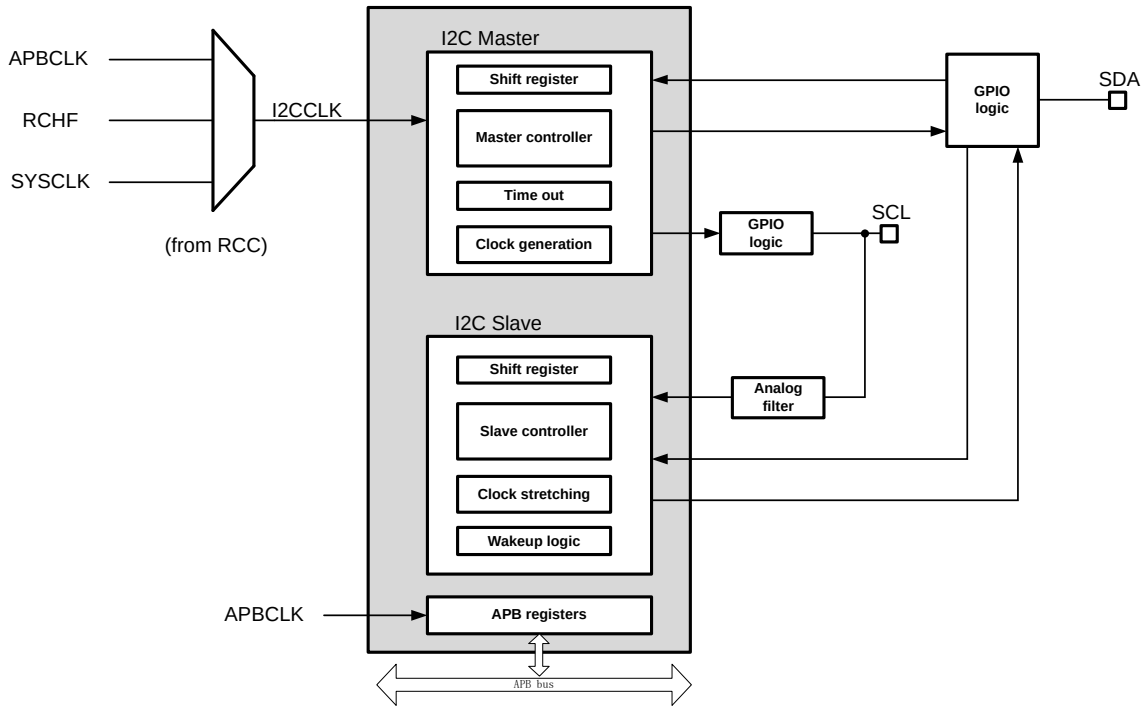


Figure 19-1 I²C block diagram

19.3 Clock Structure

Both I2C master and slave use dual clock structure.

- The bus register clocks of the master and slave are represented by PCLK, which comes from APBCLK. When the CPU or DMA needs to access the I²C internal registers, PCLK must be enabled. Reference 11.11.13 Peripheral bus clock control register 3.
- The data transceiving clock of the master is represented by I2CCLK, which can be derived from APBCLK, but also from RCHF, SYSCLK and RCMF. I²C can work independently of APBCLK. I2CCLK must be enabled to send and receive data. Reference 11.11.24 Peripheral independent clock control register 1.
- The data transceiving clock of the slave uses the SCL bus clock input, so data can be sent and received without the system clock.

The control of both PCLK and I2CCLK is done in the CMU module, and the corresponding CMU control registers must be configured correctly before I²C communication.

The use of a dual clock structure allows I2C to work without being limited by the APBCLK configuration, so that when certain peripherals need to work at a very high APBCLK frequency, I2C can still work at a reduced frequency; or conversely, the CPU works at a lower frequency, which does not affect I2C data communication at a higher baud rate.

Theoretically there is no constraint on the relative relationship between PCLK and baud rate clock, the baud rate clock can be faster or slower than PCLK, but the application needs to pay attention to whether the CPU or DMA has enough time to move the data when the difference between the two frequencies is significant.

19.4 Interface Timing

19.4.1 Interface Timing diagram

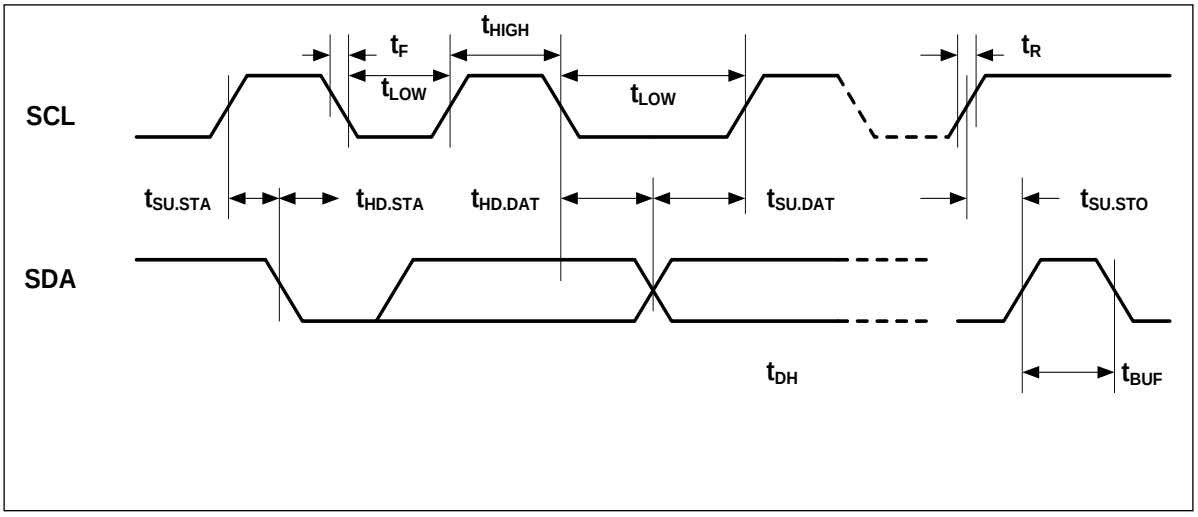


Figure 19-2 I²C Bus Timing

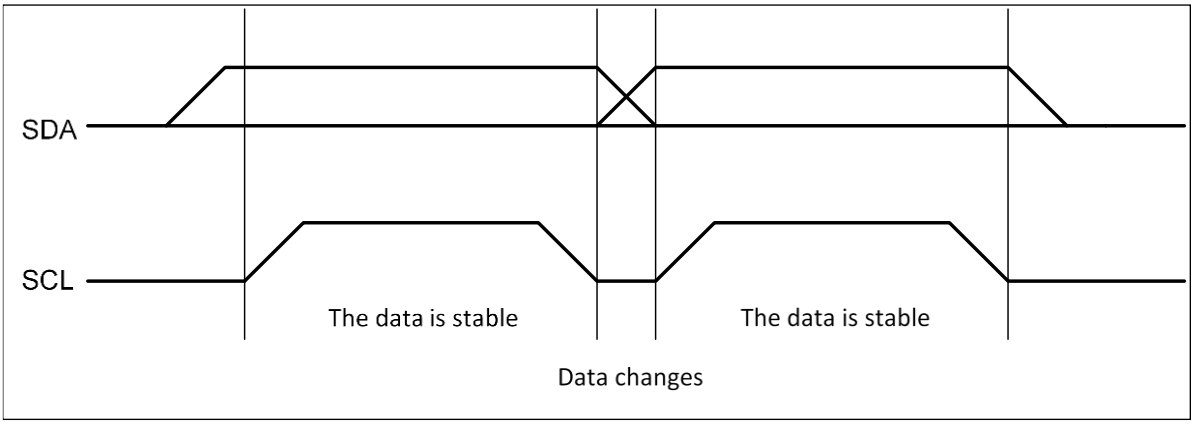


Figure 19-3 Data valid timing

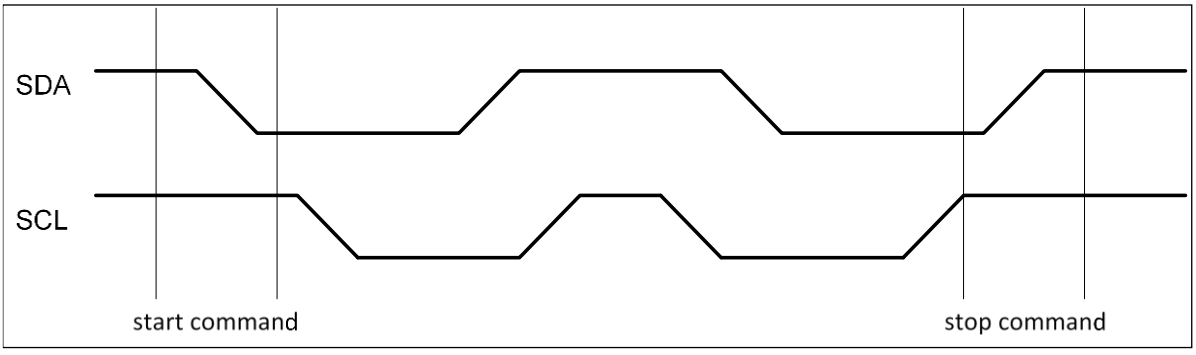


Figure 19-4 Start & Stop condition definition

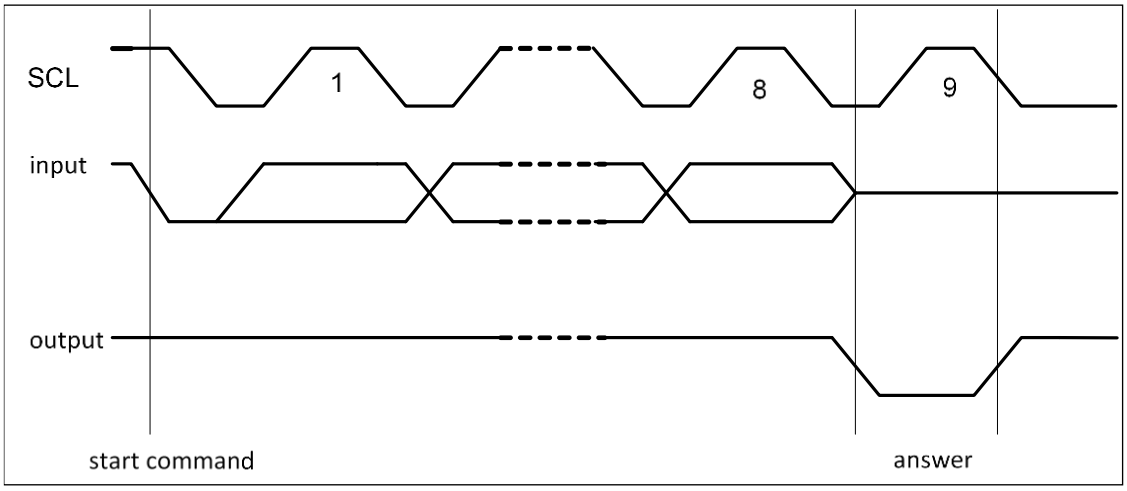


Figure 19-5 Output ACK

19.4.2 Interface Timing Description

Clock timings: The SDA pin is normally pulled high by the peripheral devices. The data on the SDA pin should change when SCL is low (refer to Figure19-3); when the data changes when SCL is high, it will be considered as a start or stop command as described below.

Start condition: When SCL is high, the change of SDA from high to low is considered as a start condition and must be used as the start of any read/write operation (refer to Figure 19-4).

Stop condition: When SCL is high, the change of SDA from low to high is considered as a stop condition, and after a read operation, the stop condition puts the EEPROM into wait-state or low-power mode (refer to Figure 19-4).

Acknowledge: Data on the SDA is transmitted in groups of 8 bits serially, MSB first, and the receiver should send back an acknowledge bit (hereafter ack) on the 9th cycle after receiving each byte. The clock for ack is provided by the master. The sender leaves SDA undriven during the ack period and the receiver shall pull SDA low to ensure that SDA is low when the clock high, creating a valid ack signal (refer to Figure 19-5).

Parameter	Sign	Standard (100K)		Fast (400K)		Units
		MIN	MAX	MIN	MAX	
SCLclock frequency	F_{SCL}	0	100	0	400	kHz
Start conditon establishment time	$T_{SU:STA}$	4.7	—	0.6	—	us
Start condition stretching time	$T_{HD:STA}$	4.0	—	0.6	—	us
Clock stretching low time	T_{LOW}	4.7	—	1.3	—	us

Parameter	Sign	Standard (100K)		Fast (400K)		Units
		MIN	MAX	MIN	MAX	
Clock stretching high time	T_{HIGH}	4.0	—	0.6	—	us
Data input setup time	$T_{SU:DAT}$	250	—	100(4)	—	ns
Data input stretching time	$T_{HD:DAT}$	5.0 0(2)	— 3.45(3)	— 0(2)	— 0.9(3)	us us
SDAand SCLpull-up times	T_R	—	1000	20+0.1Cb(5)	300	ns
SDAand SCLpull-down time	T_F	—	300	20+0.1Cb(5)	300	ns
Stop condition establishment time	$T_{SU:STO}$	4.0	—	0.6	—	us
Bus idle time	T_{BUF}	4.7	—	1.3	—	us
Capacitive load on the bus	Cb	—	400	—	400	Pf
Min Noise tolerance	V_{nL}	$0.1V_{DD}$	—	$0.1V_{DD}$	—	V
Max Noise tolerance	V_{nH}	$0.2V_{DD}$	—	$0.2V_{DD}$	—	V

19.5 I²C work mode

The I²C module supports the following working modes:

- Master receive
- Master send
- Slave receive
- Slave send

After the chip is powered on, the I2C module is disabled by default, and the master and slave do not work. The software needs to select the working mode of the module according to the application, and set MSPEN to enable master communication, or set SSPEN to enable slave communication.

The master and slave cannot work at the same time because they reuse the same IOpins as SCL and SDA. In principle, software is prohibited to set MSPEN and SSPEN to 1 at the same time.

19.6 I²C slave address format

The I²C bus protocol defines the following reserved addresses. For most of the reserved addresses, the I²C slave hardware does not make legal judgments, and the software can customize the processing according to the received address.

But for the 10bit slave address application, that is, when SSPCON.A10EN=1, the 1st byte must start with 11110, otherwise the ADDR_ERROR error flag will be triggered. In the case of SSPCON.A10EN=0, if the slave receives the address byte starting with 11110, it will also set the ADDR_ERROR error flag.

Slave address	R/W_ bit	Description
0000 000	0	General Call address
0000 000	1	START byte
0000 001	X	CBUS address
0000 010	X	Reserved for different bus format
0000 011	X	Reserved for future purpose
0000 1XX	X	HS-mode master code
1111 1XX	X	Reserved for future purpose
1111 0XX	X	10bit slave addressing

Figure 19-2 I2C slave reserved addresses definition

19.7 I²C initialization

The I²C module must be correctly initialized before I2C communication can take place. It is recommended that the software follows the following steps to initialize it.

- Clear the I2CRST register of the RCC module to ensure that the I²C module is not in a reset state
- Set the I2C_APBEN register of the RCC module to enable the I²C module register bus interface clock
- Configure the I2C_CKSEL and I2C_CKEN registers of the RCC module to select and enable the I²C operating clock (If it is in slave mode, this step is not required)
- Configure analog filter enable as required (SCL and SDA input analog filtering, >50ns)

19.7.1 IO configuration

The FM33LE0xxA has up to two sets of pins for data transfer. Before starting I²C communication the FCR register of the corresponding pin needs to be set to AF:

SDA: PA12/PD12

SCL: PA11/PB15

Note that if PA11 and PB15 are configured for SDA function at the same time, PA11 is connected

to the I²C module, and PB15 is invalid; if PA12 and PD12 are configured for SCL function at the same time, both pins in master mode will output SCL signals, and in slave mode, only PA12 is connected to the SCL input of the I²C slave.

PA11 and PA12 are strong driving true OD pins, which must be used with external bus pull-up resistors, have 20mA sink current capability and support Fm+ mode.

19.7.2 Master baud rate configuration

The I2C master needs to configure the communication baud rate before enabling, but the slave does not need to be configured.

MSPBRG[8:0] baud rate configuration register is used to generate communication baud rate. MSPBRG is the 9-bit baud rate division coefficient, and the baud rate calculation formula is as follows:

$T_{SCL}=2T_{BRG}$

$T_{BRG}=2\times T_{I2CCLK}\times(MSPBRG[8:0]+1)$; T_{I2CCLK} is the I²C working clock cycle, namely:

$MSPBRG = F_{I2CCLK}/(4*F_{SCL}) - 1$

For example, for a 100k baudrate, if the I2C working clock is 8M, then MSPBRG=19.

19.7.3 Slave input analog filtering and output delay

The analog filter function is only for the SCL pin, and only the SCLi input signal of the slave can enable the analog filter function.

At the same time, the SDA output delay of the slave can ensure the output hold time of SDA relative to the falling edge of SCL by adding an analog delay greater than 300ns on SDAo

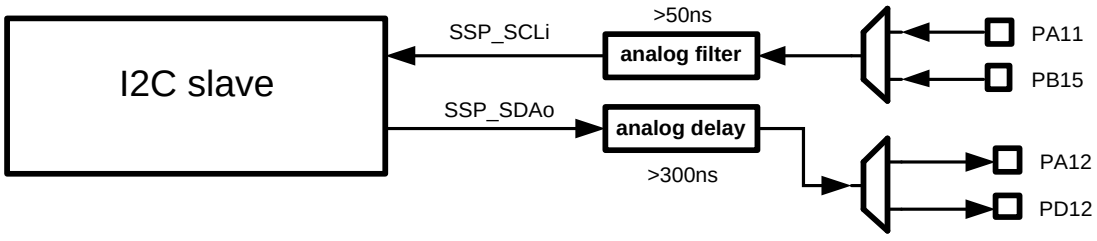


Figure 19-6 Slave Signal Filtering

19.8 I²C master function

The I²C master dose not support a multi-master bus, so all other devicess conneccted to the bus are slaves. The bus is always synchronously clocked by SCL sent from master and the SDA data flow direction can be either master sending, slave receiving or slave sending, master receiving.

I²C bus communication is always initiated by the master., and the master mode supports 7bit or 10bit addressing.

19.8.1 7 bit addressing

In 7bit addressing, the first byte sent by the master contains the slave address and the transfer direction bit ($R/\overline{W}$), depending on $R/\overline{W}$ the subsequent transfer is a master writing data to the slave ($R/\overline{W}$=0) or a master reading data from the slave ($R/\overline{W}$=1).

Name	Slave Address Byte							
Bit	7	6	5	4	3	2	1	0
Bit Name	address							R/W

Bit description:

Bit	Bit Name	功能
7-1	address	Slave device address
0	R/W	0: Write means w=sending data (master sends) 1: Read means request data (slave sends back）

Master writes data to slave

A typical frame structure for 7bit addressing, with the master writing data to the slave, is shown in the diagram below.

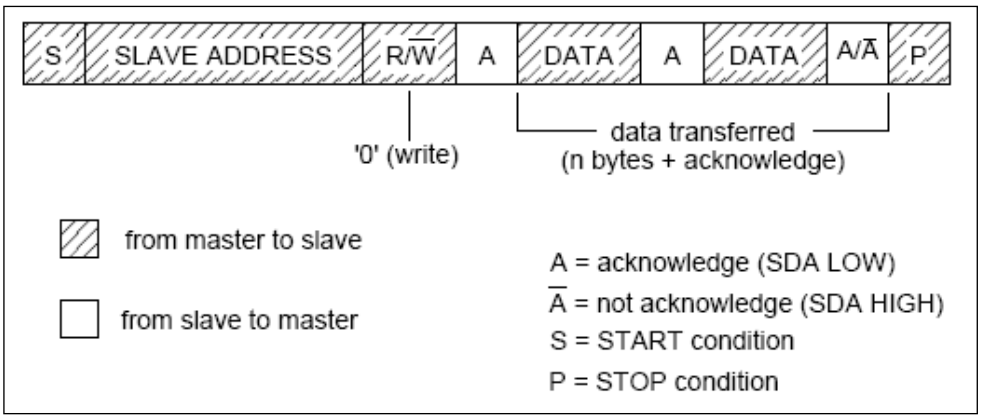


Figure 19-7 Frame format when a master writes data to a 7-bit address slave

1. The master initiates START Timing

2. Master sends slave address, slave address contains 7 bits of slave address and 1 bit of R/W flag bit which is 0 when sending data

3. The master sends the first 8-bit data frame

4. The master will determine if a valid ACK is detected at the 9th SCL after each 8-bit data is sent, if the master detects a positive ACK, it will continue to send next byte

5. If the slave cannot reply ACK, the master should send a STOP condition to terminate the transmission after detecting the NACK

6. After the master has finished sending all data, it will send the STOP Timing

The software initiates the operation flow of the I²C master send as follows:

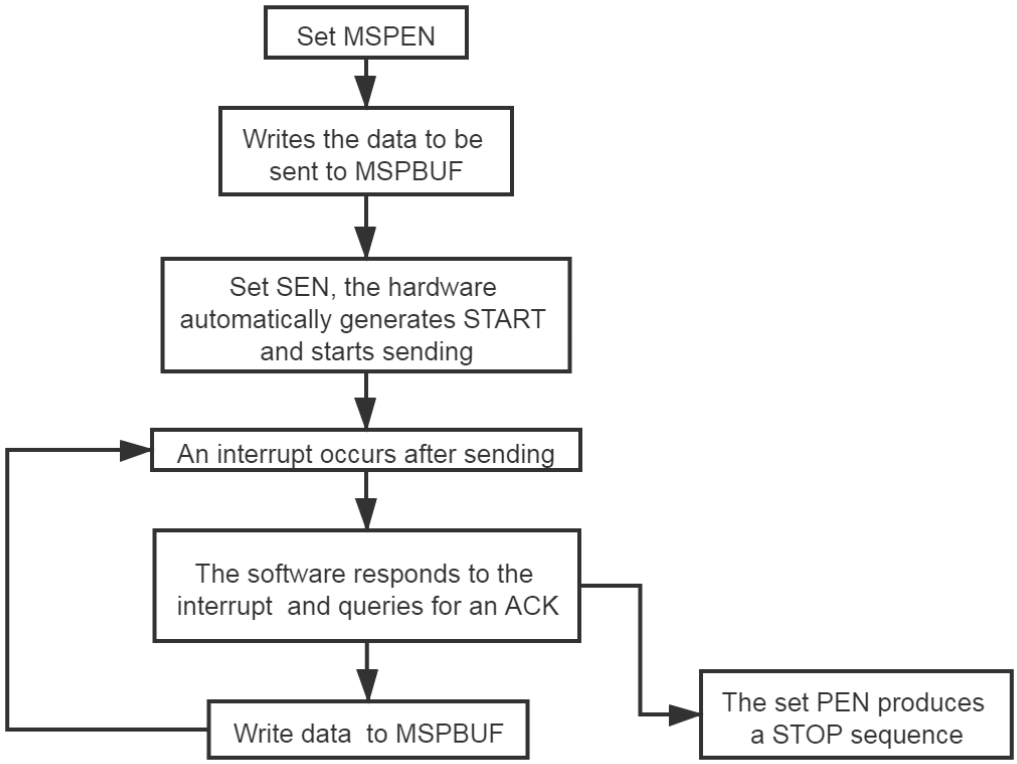


Figure 19-8 I²C software sending data flow diagram

The waveform diagram of the I²C master writing data to the 7-bit address slave is as follows:

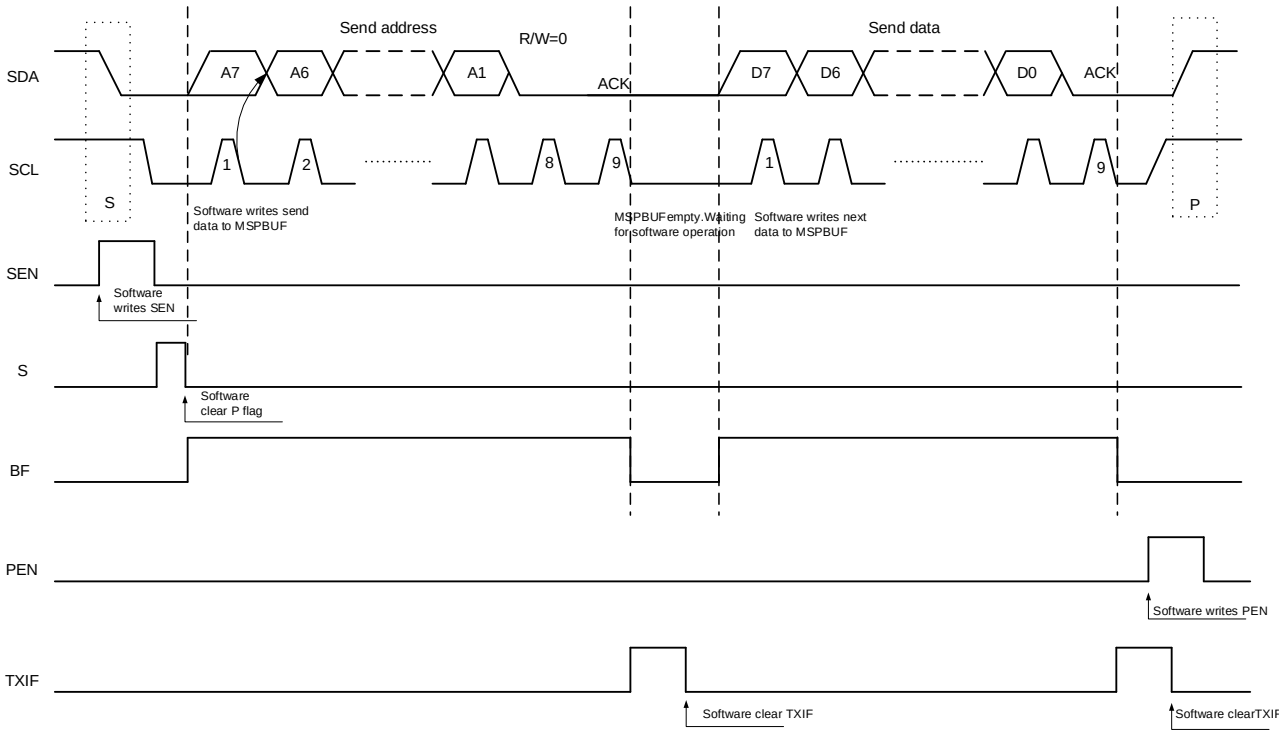


Figure 19-9 I2C master sends data flow diagram to 7-bit address slave

Master reading date from a slave

A typical frame format for 7bit addressing, where the master reads data from the slave, is shown in the diagram below.

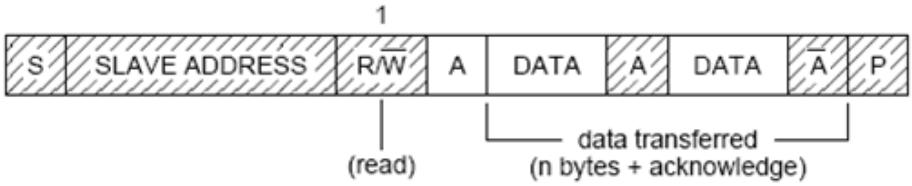


Figure 19-10 Frame format when a master reads data from a 7-bit addressing slave

1. Master initiates START timing

2. The master sends the slave address, the slave address contains 7 bits of the slave address and 1 bit of the R/W flag bit which is 1 when the data is read

3. Set MSPCON.RCEN to 1, the master automatically turn to receive state

4. The master starts to receive the first byte of 8-bit data, and sends a valid ACK to the slave at the 9th SCL, so as to continue to read the next data byte

5.After reading the last byte, the master sends a NACK to the slave at the 9th SCL

6. The master sends STOP bit to terminate the reading

The operational flow of I²C reception is shown in the following diagram:

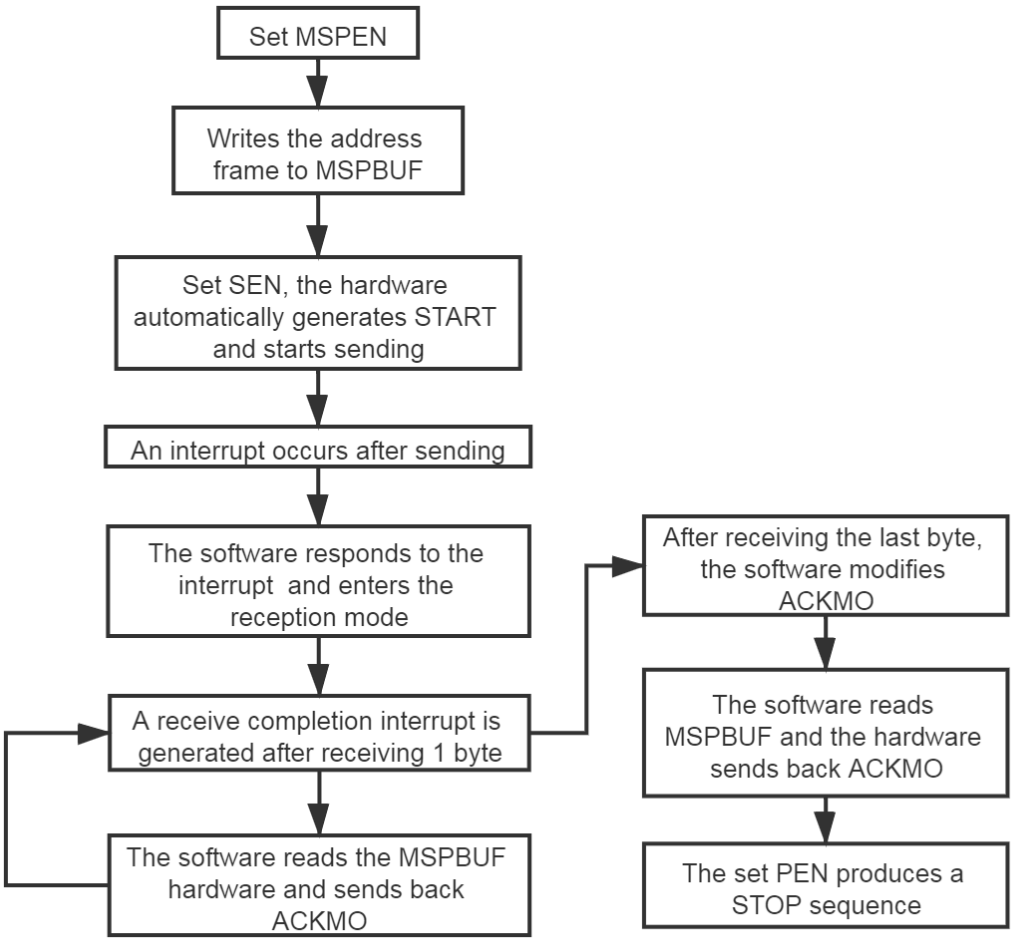


Figure 19-11 I²C software receive data flow diagram

The master sends back a response according to the ACKMO register each time after receiving data sent by the slave. The ACKMO reset value is 0, i.e. by default the master replies ACK. If the software wants the master to reply a NACK, the ACKMO register needs to be rewritten to 1 when previous byte is received. ACKMO will be cleared automatically after the NACK is sent.

The waveform diagram of the I2C master reading data from rhe 7-bit address slave is as follows:

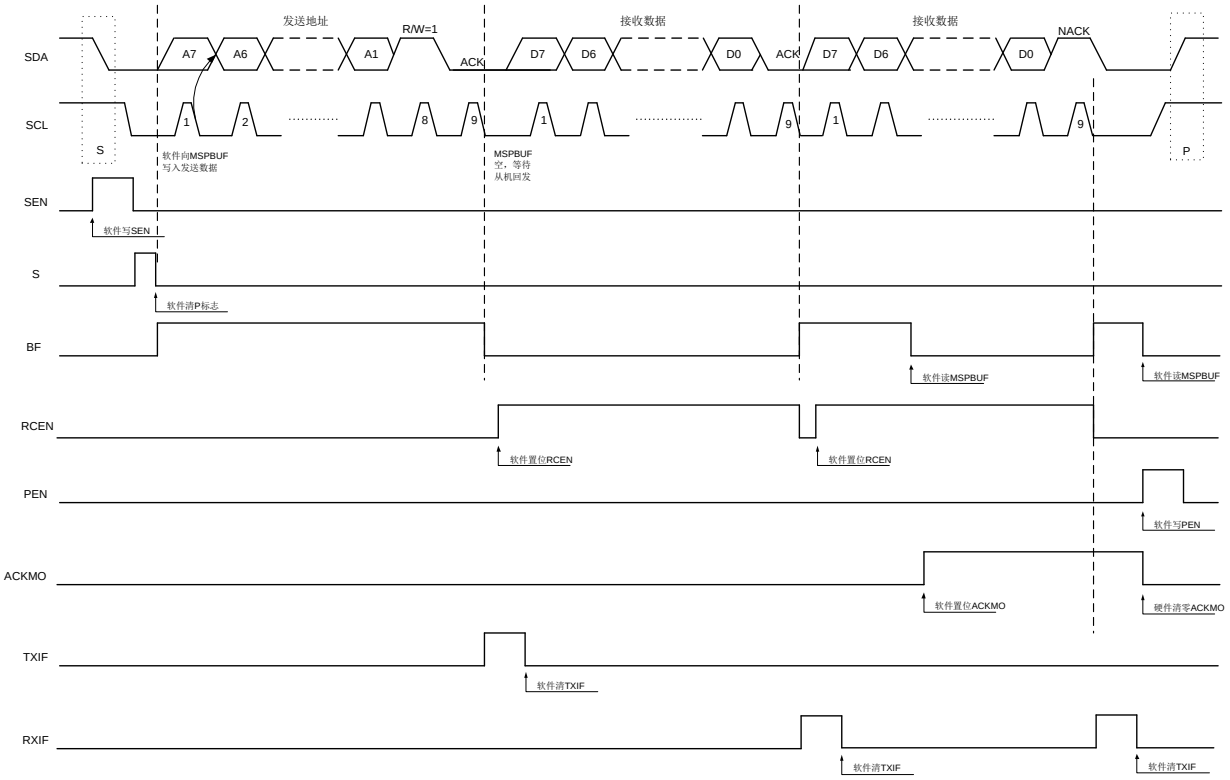


Figure 19-12 I²C reads data flow diagram from 7-bit address slave

Bidirectional Data Transfer (Combined Mode)

A typical bi-directional data read/write flow is shown in the diagram below. While the master is writing or reading data, the master can restart a new transaction by sending the Repeated Start condition, so the master can realize bidirectional communication within one transaction.

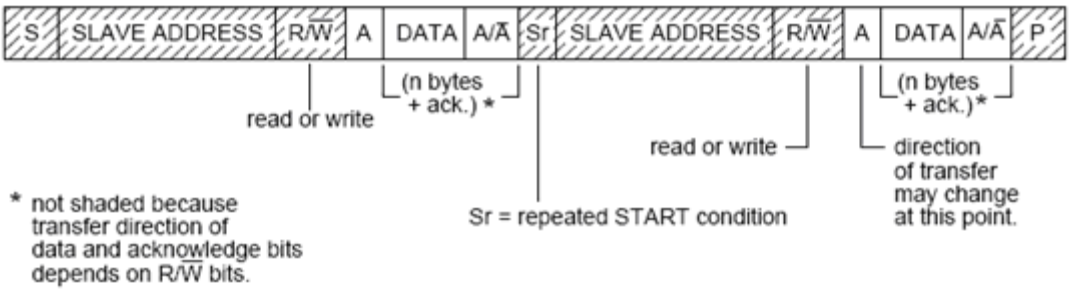


Figure 19-13 Frame format for bi-directional data communication

The software operation procedure for bidirectional communication is similar to that for unidirectional communication, except that the direction of transmission is modified by sending the ReSTART condition and slave address bytes.

19.8.2 10 bit addressing

When 10bit addressing, the first byte sent by the master contains part of the slave address (11110_A9_A8) and the transfer direction bit ($R/\overline{W}$), the second byte contains the remaining slave address (A7~A0). After the two byte address have been sent, the data is then transferred.

Master writes data to the slave

A typical data flow for 10bit addressing, where the master writes data to the slave, is shown in the diagram below.

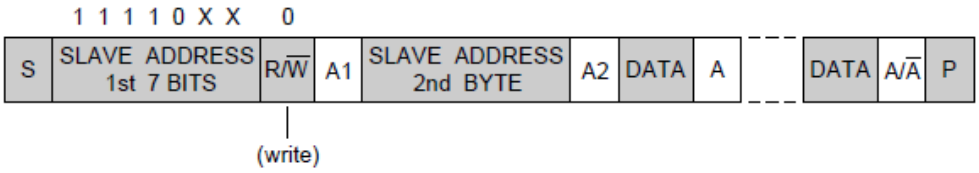


Figure 19-14 10bit addressing, the master writes data to the slave

1. The master intiates START timing

2. The master sends the first slave address byte, starting with 11110, followed by the highest bit of the 2bit slave address, and the R/W flag bit, the R/W bit is 0 when sending data

3. The master checks the ACK sent back by the slave

4. The master sends the second slave address byte, containing the lower 8 bits of fhe slave address

5. The master checks the ACK replied by the slave

6. The master continues to write data to the slave

7. After the master has finished sending all data, it sends the STOP timing

The software procedure of the I²C master transmitting is as follows:

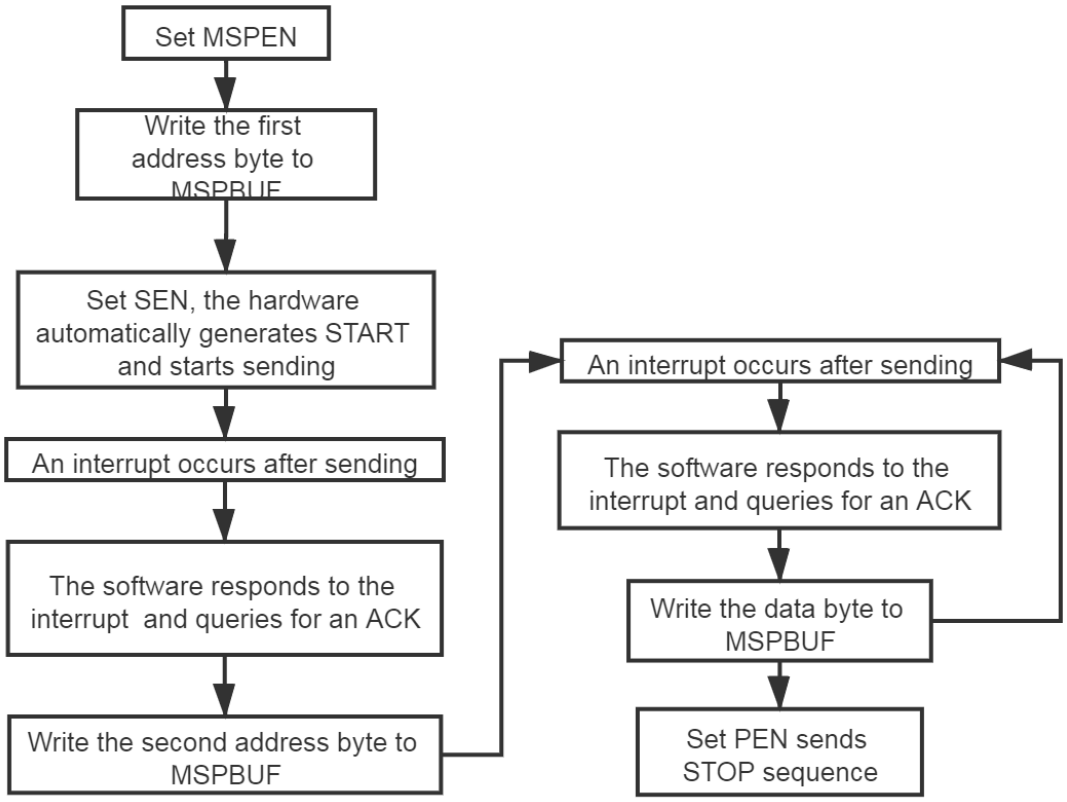


Figure 19-15 I²C software transmit flow diagram

Master reads data from the slave

A typical data flow for 10bit addressing, where the master reads data from the slave, is shown in the diagram below.

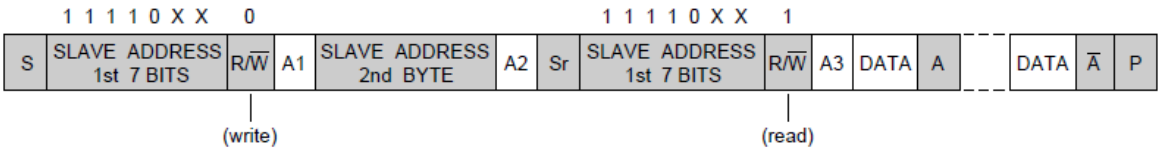


Figure 19-16 10bit addressing, master reads data from the slave

1. Master initiates START timing

2. The master sends the first byte of the slave address, including 5 bits of the leading code 11110, 2 bits of the highest bit of the slave address and 1 bit of the R/W flag bit, the R/W bit is 1 when the data is read

3. The master sends the second byte of the slave address, including the low 8 bits of the address

4. The master sends ReSTART timing

5. The master sends the first byte of the slave address again, changing R/W to 0

6. Set MSPCON.RCEN to 1, the master goes to receive state

7. The master starts to receive the first byte of 8-bit data, and sends a valid ACK to the slave at the 9th SCL, thus continuing to read the next data byte

8. After reading the last byte, the master sends a NACK to the slave at the 9th SCL

9. The master sends STOP timing

The software procedure for I2C receive flow as follows:

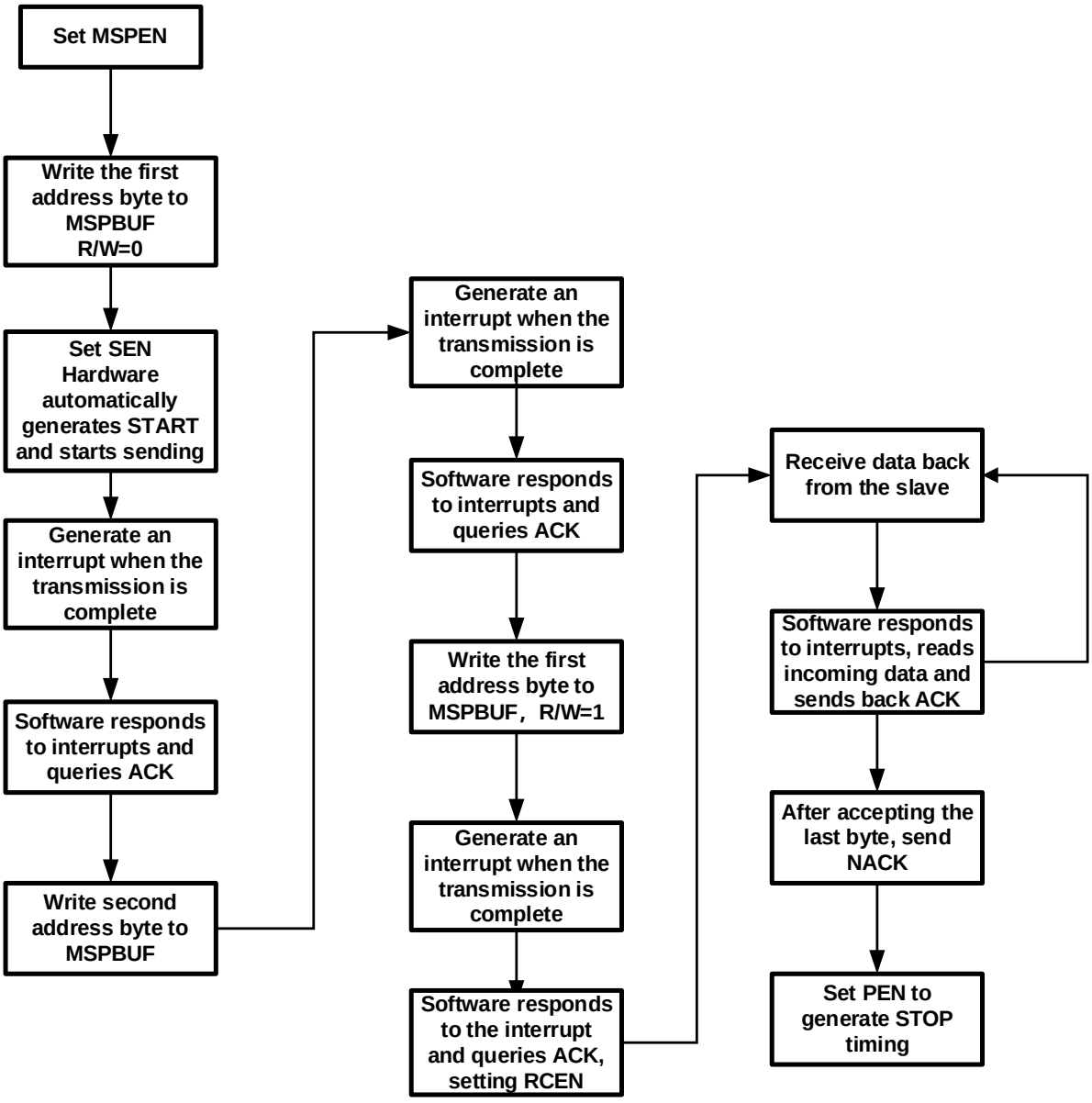


Figure 19-17 I²C software send data flow diagram

Bidirectional data transfer (combined mode)

A typical bi-directional data read/write flow is shown in the figure below. During a master write or read, the master can restart a new transaction by sending Repeated Start condition, so the master can have bidirectional communication in one transaction.

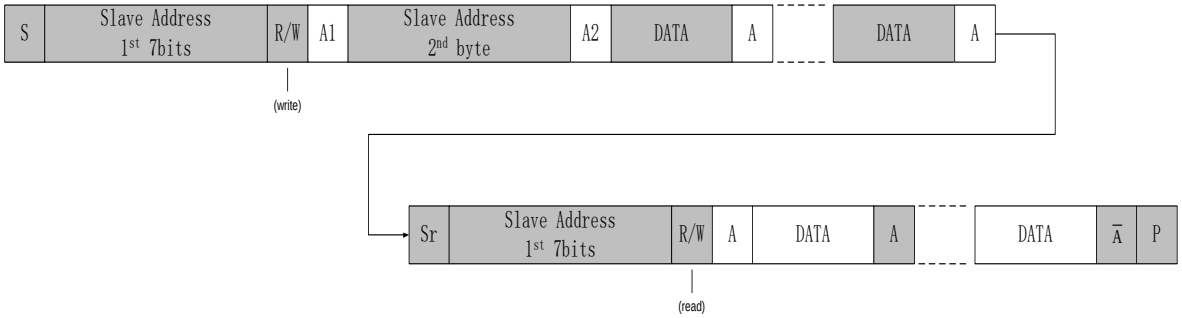


Figure 19-18 I²C software send data flow diagram

The software procedure for combined transfers is similar to that for unidirectional transfers, except that the transfer direction is modified by sending the ReSTART condition and 1st slave address bytes.

19.8.3 DMA

The I²C master supports DMA. It should be noted that the bus clock (APBCLK) of the I²C module must be enabled in order to use the DMA function.

The Master uses DMA to write data to a slave

When the master uses DMA to send data, all data including the slave address byte and the send data needs to be written to RAM in advance and transmitted via a DMA request. The software should configure the target DMA channel as I2C_TX.

In case DMAEN=1, MSPEN is set and if the data buffer MSPBUF is empty, the I²C module generates a DMA request and the DMA module responds by writing the data to MSPBUF and the I²C module automatically sets SEN to generates START condition to start data transmission (first byte is the slave address). The I²C does not check the validity of the data, the software must ensure that the data in RAM is correct.

After each byte is sent, the I²C checks the slave ACK and generates a new DMA request if the ACK is correct, or a NACK interrupt if a NACK is received and no further DMA requests are generated.

When the DMA completes sending the specified length of data, the DMA transfer completion interrupt is generated. Then software can set PEN to generate the STOP condition, or the I²C hardware can automatically set PEN to generate the STOP condition according to the DMA transfer completion signal. The desired strategy can be selected by configuring the AUTOEND register.

The flow of the master using DMA for transmitting is shown below:

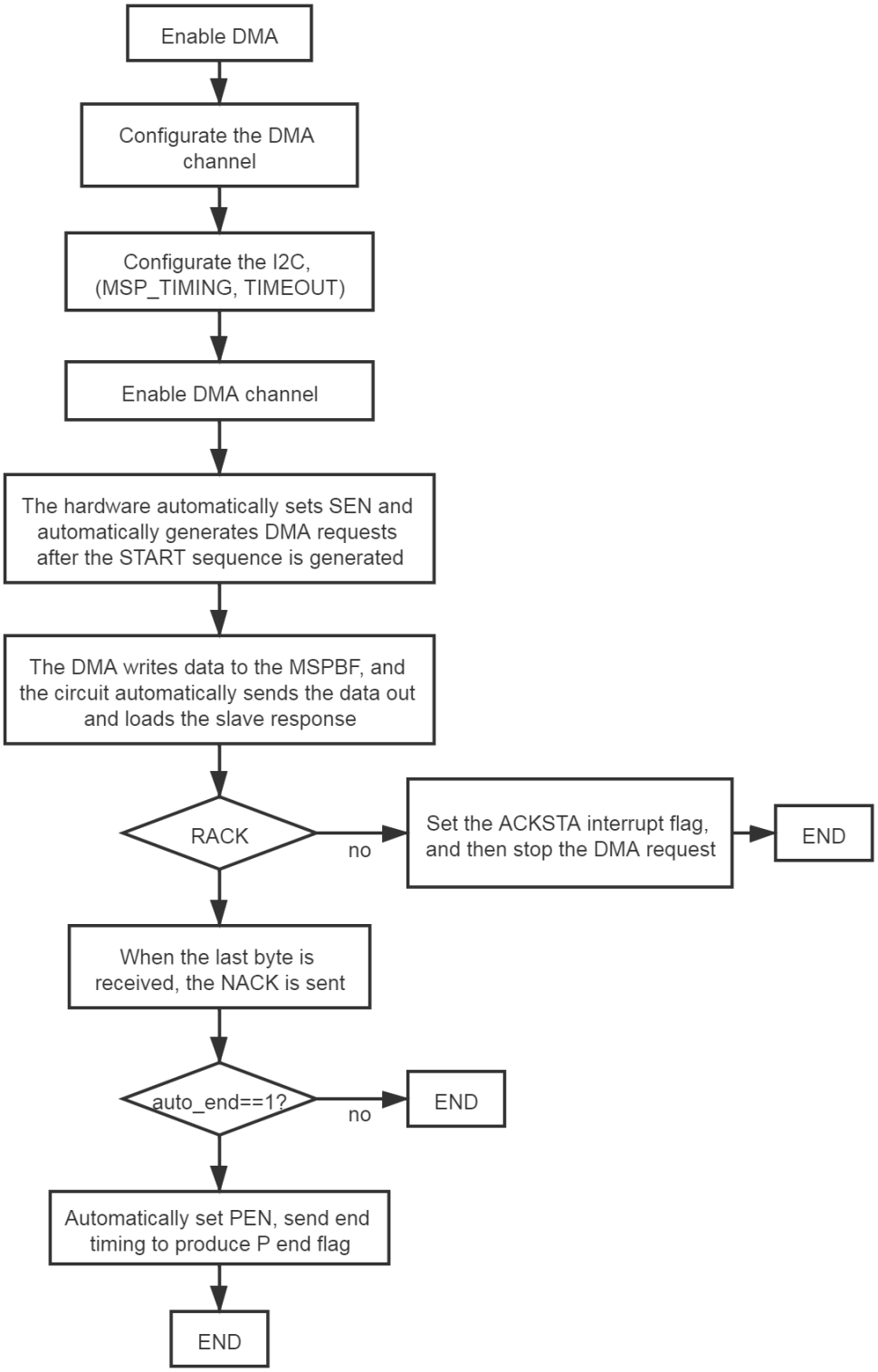


Figure 19-19 I²C master DMA transmit flowchart

The master uses DMA to read data from the slave

In this case, the slave address byte must be sent by software. The software should configure the target DMA channel as I2C_RX.

After the software first sends the slave address, set MSP_AMAEN=1, then enable the corresponding DMA channel, I²C automatically enters the receive mode and generates a DMA request after each byte is received, notifying the DMA to read the MSPBUF contents and reply an ACK to the slave at the same time.

When the DMA transfer reaches the specified length, the DMA's transfer complete flag will notify the I²C to reply an NACK. A STOP condition can be generated by software or hardware, depending on the AUTOEND register configuration.

Note: *When the I²C master receives data through DMA, under different AUTOEND configurations and the same DMA transfer length (CHxTSIZE) configuration, the number of bytes received by DMA will be different. When AUTOEND=0, the received byte count is CHxTSIZE+1; when AUTOEND=1, the received byte count is CHxTSIZE.*

The flow of the master using DMA for reception is illustrated below:

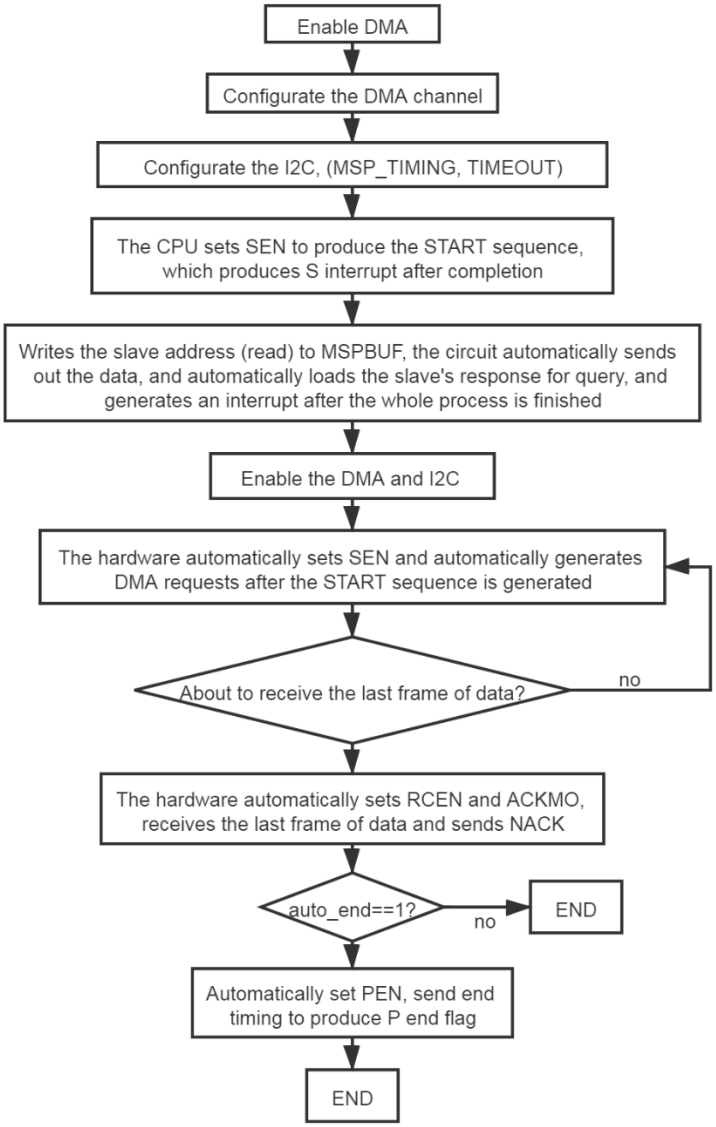


Figure 19-20 I²C master DMA receive flowchart

19.8.4 SCL Clock Stretching

The I²C protocol allows low-speed slaves to suspend data communication by pulling SCL low. The I²C masters must support this feature. Therefore, at the start of each byte transfer, the master checks the actual level of SCL on the bus after attempting to drive SCL high. If it is not high, it means that the slave is performing an SCL stretching and the master will continue to monitor the SCL level until SCL is high before starting subsequent operations.

Note: The master only performs an SCL stretching check at the first SCL rising edge of each byte transfer.

19.8.5 Timeout function

The I²C master also implements a timeout function that generates an alarm interrupt and returns to IDLE status if SCL is stretched by slave.

When the master detects an SCL stretching, its internal timer starts counting. The maximum length of the SCL stretching timeout is 4096 SCL cycles. Assuming a baud rate of 100K, the timeout period is approximately 40ms. And if the baud rate is 400K, the timeout period is approximately 10ms.

The timeout period can be set by the software via the 12bit TIMEOUT register. The software must set the TIMEOUT register with MSPEN is 0. This reset value is 0xFFF, which means the maximum 4096*T_{SCL} timeout period. When the SCL stretching is detected, the TIMEOUT register starts to decrement. And when the counter reaches 0, the counter stops and the TIMEOUT register is reset to 0xFFF, and a timeout interrupt is triggered at the same time. Therefore, the timeout period can be set by modifying the initial value of TIMEOUT.

$T_{SCL\_STRETCHING\_TIMEOUT}$=TIMEOUT[11:0] * T_{SCL}

When a TIMEOUT interrupt occurs, it is recommended that the I²C module is reset by software.

This timeout function can be disabled. The software can also implement its own timeout function of any length by using the timer in combination with the SCL pin state polling.

19.8.6 Programmable Timing

The I²C module provides flexible timing programming features that allow the user to define the low level width, high level width of the SCL clock, and the setup and hold time of the SDA data.

The MSPBRG register allows the low and high level duration of of the SCL to be set, and the SDAHD register configures the hold and setup time of the SDA data relative to the SCL clock pulse.

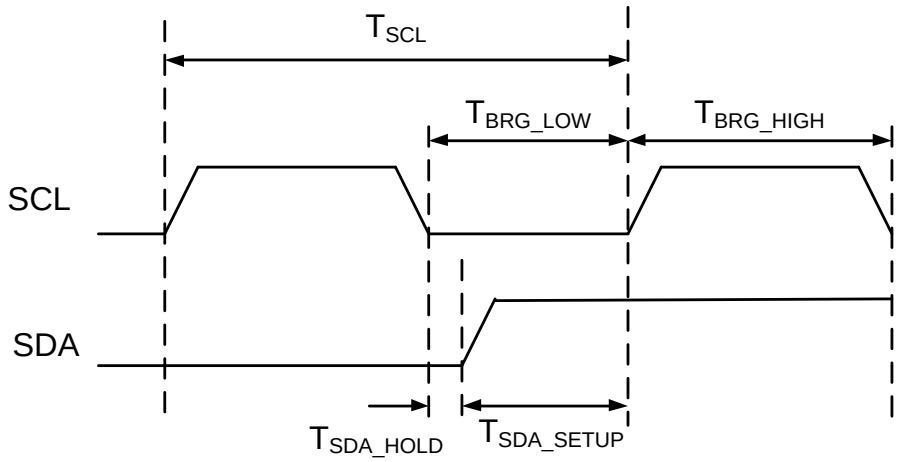


Figure 19-21 Master timing control

In the above figure, T_{SCL} is the communication baud rate and parameters can be defined by the following equation:

$T_{SCL} = T_{BRG\_LOW} + T_{BRG\_HIGH}$

$T_{SDA\_SETUP} = T_{BRG\_LOW} - T_{SDA\_HOLD}$

Note that the configuration of the MSPBGRH, MSPBRGL and SDAHD registers must meet the following requirements, as violation of these requirements will result in abnormal bus timings.

MSPBRGH>= 2

MSPBRGL >=2

MSPBRGL-1>= SDAHD >=1

TIMEOUT >= 1

19.9 I2C slave function

The I2C slave does not need system clock to work, so it can send and receive data and wake up when the chip is sleeping.

After the slave receives 1 byte of data, it generates an interrupt to notify the CPU to process the data. Before the CPU takes the data, the hardware can pull down SCL (software control is enabled) to notify the sender that it is busy, and the sender should suspend sending until SCL is released. If the receiver cannot respond to the ACK, the sender should send P to terminate the communication or send Sr to start a new communication after failing to detect the ACK.

After the slave sends 1 byte of data, an interrupt is generated to notify the CPU, and the hardware pulls down SCL to make the master wait. The CPU responds to the interrupt and prepares the next byte of data before releasing the SCL. The master continues to send SCL to make the slave continue to send data.

19.9.1 Slave adressing

According to the SSPCON.A10EN register status, the slave can support 7bit or 10bit addressing process. The slave address is defined by the SLAVE_ADDR register.

For 10bit slave address applications, that is, when SSPCON.A10EN=1, the 1st byte must start with 11110, otherwise the ADDR_ERROR error flag will be triggered. In the case of SSPCON.A10EN=0, if the slave receives the address byte starting with 11110, it will also set the ADDR_ERROR error flag.

19.9.2 Slave sends data

Recommended operation process:

- The slave receives the address byte (R/W=1), sends back ACK, generates address match interrupt
- Since R/W=1, the hardware automatically performs SCL stretching, and the slave enters the sending state
- The software responds to the interrupt, queries the R/W flag, and confirms that it is sent by the slave
- The software writes the data to be sent into SSPBUF
- The hardware automatically releases the SCL
- The new SCL is coming, SSPBUF is shifted and output to the SDA bus
- Receive ACK and generate transmission completion interrupt
- Repeat the data transmission process until the STOP sequence is received, or the master NACK is received

The following figure is a typical schematic diagram of slave data sending waveform:

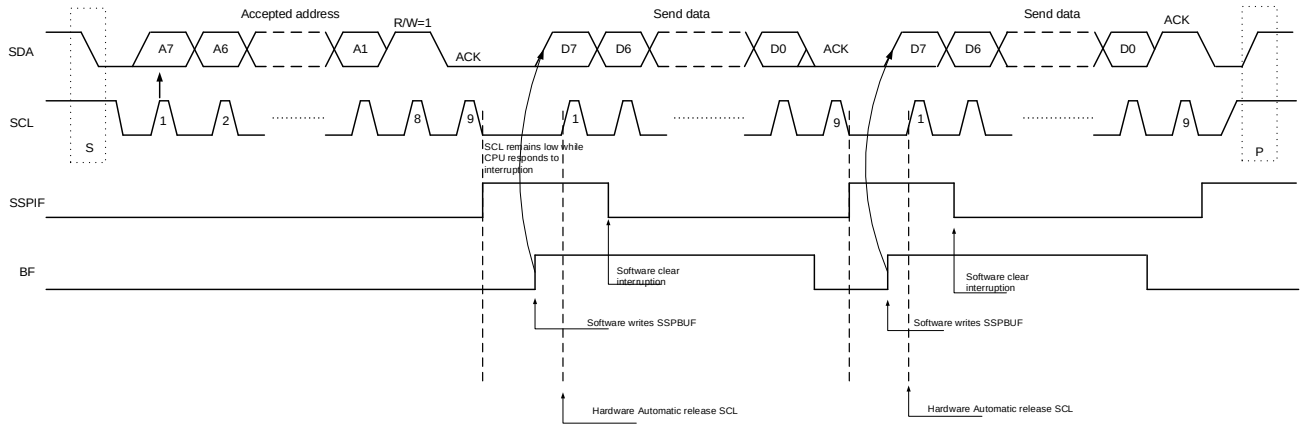


Figure 19-22 Slave data sending waveform

In the slave sending process, when the slave receives the correct address, the ADM flag is set, and the address byte will not be written into SSPBUF, so the BF flag will not be set. The hardware automatically pulls down the SCL signal and waits for the software to write SSPBUF. When the software writes SSPBUF, the BF flag is set, and the hardware releases SCL at the same time.

19.9.3 Slave receive data

Recommended operation process:

- The slave receives the address byte (R/W=0), sends back ACK, generates address match interrupt
- Since R/W=0, the hardware automatically performs SCL stretching, and the slave keeps the sending state
- The software responds to the interrupt, queries the R/W flag, and confirms that it is received by the slave
- Software reads SSPBUF, hardware automatically releases SCL and starts to receive data
- The master data byte arrives, the hardware sets the BF flag after the byte reception is completed
- The slave sends back ACK and generates a reception completion interrupt
- The hardware automatically performs SCL stretching (SCLSEN=1)
- The software responds to the interrupt, reads SSPBUF, and the hardware automatically clears the BF flag
- The hardware release SCL automatically
- Repeat the data reception process until the STOP sequence is received, or the software sets

ACKEN to 0

The following figure is a typical schematic diagram of slave data receiving waveform (SCLSEN=1):

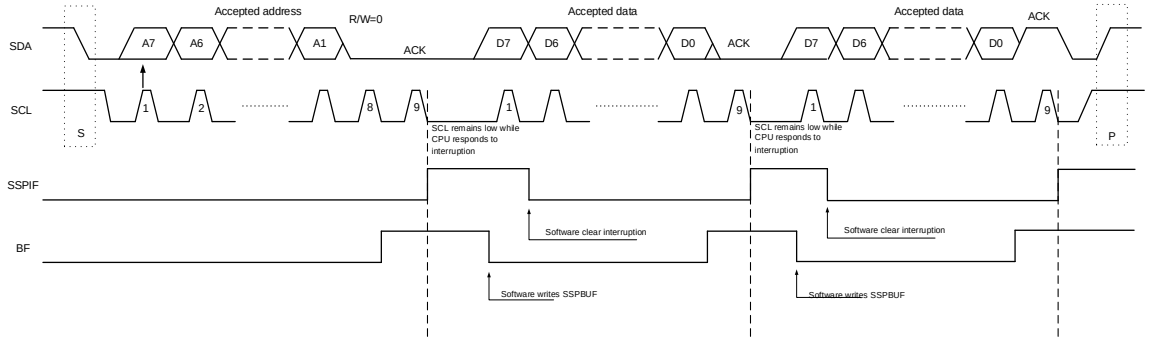


Figure 19-23 Slave data receiving waveform

During slave reception, the slave first receives the address byte, and if the address matches, the ADM flag is set, the address byte will be written to the SSPBUF and the BF flag is set, then the hardware pulls down the SCL. when the software reads the SSPBUF, the BF flag is automatically cleared and the hardware releases the SCL, allowing for subsequent data reception.

Note: In the slave receive process the address byte is written to the SSPBUF and causes the BF to be set, the software needs to read the SSPBUF to clear the BF and release the SCL, whereas in the slave send process the address byte is not written to the SSPBUF and therefore the BF flag is not set.

The slave can passively end communication or actively end communication by receiving data.

If the host actively sends STOP, the slave passively ends this communication. Alternatively, if the software clears the ACKEN register in the interrupt handler, the slave will send a NACK back after receiving the next byte, and the host will send a STOP to end this communication after receiving the NACK.

Salve SCL stretching

SCL stretching is enabled by default on I²C slaves, but can be disabled by software (SCLSEN register) to accommodate hosts that do not support slave SCL stretching.

When SCL stretching is enabled, the software can only clear the BF flag when the receive buffer is read during SCL stretching after the data has been received. If there is a data overflow during reception and the SSPOV flag is set, the hardware sends back a NACK and the SCL is no longer extended so that the host can issue a STOP; if SSPOV is set, it is recommended that the software waits for the STOP flag to be set before reading the receive buffer to clear the BF flag.

Receive data overflow

When the slave receive buffer is full (BF=1), if new data is received again, a receive overflow occurs and the SSPOV flag is set. The old data in the receive buffer will be overwritten by the new data. A receive data overflow is only possible if the slave has closed the SCL extension function.

The following diagram illustrates the occurrence of a data reception overflow with SCLSEN=0.

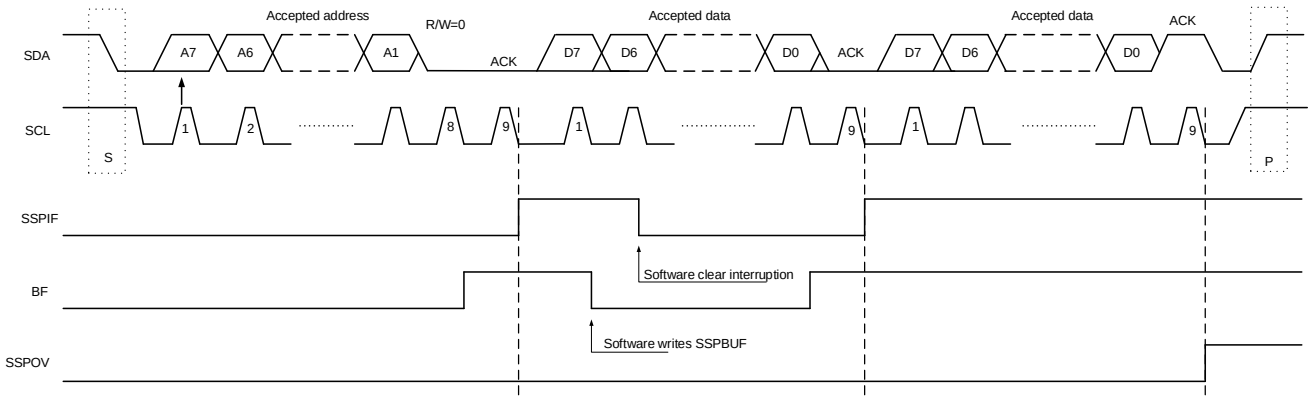


Figure 19-24 Slave data received waveform (SCLSEN=0, Received overflow)

19.9.4 Slave low-power receiving wake-up

As the I²C slave does not require a system clock to operate, it can receive data and wake up the MCU in hibernation mode.

The I²C slave supports START timing wake-up, address matching wake-up and data reception completion wake-up.

Software setup flow:

- Turn off the I²C master
- Set the slave address
- Set SE, ADME or BFE interrupt enable according to the desired wake-up event
- Set the corresponding GPIO to I²C function
- Set SSPEN to start the I²C slave
- Enter hibernation mode and wait for data reception
- When the wake-up event arrives, the software queries the wake-up source and processes the I²C data transfer

19.9.5 DMA

The I²C slave supports DMA. It should be noted that the bus clock (APBCLK) of the I²C module must be enabled in order to perform the DMA operation. The bus clock is used to generate DMA requests and receive DMA responses.

The slave uses DMA to receive data

When the I²C slave receives the correct address, it generates the ADM interrupt flag. After the software responds to the interrupt, it queries the received R/W bit. If it is 0, the master is ready to write data to the slave. At this time, the software can configure the specific DMA channel as I2C_RX

and enable the DMAEN of the I²C slave; then every time the slave completes a byte reception, a DMA request will be generated and the DMA will be notified to read the SSPBUF.

There are two possibilities to end DMA slave reception:

1) The data transfer length has not reached the DMA length configuration, and the master has issued a STOP sequence, the software should respond to the STOP interrupt and actively handle this situation;

2) The data transfer length reaches the DMA length configuration, but because the DMA request is generated after the slave sends back ACK, the software should respond to the DMA transfer completion interrupt and clear ACKEN to zero, so that the slave will send back NACK to end this communication after receiving the next byte.

The flow of receiving by the slave using DMA is as follows:

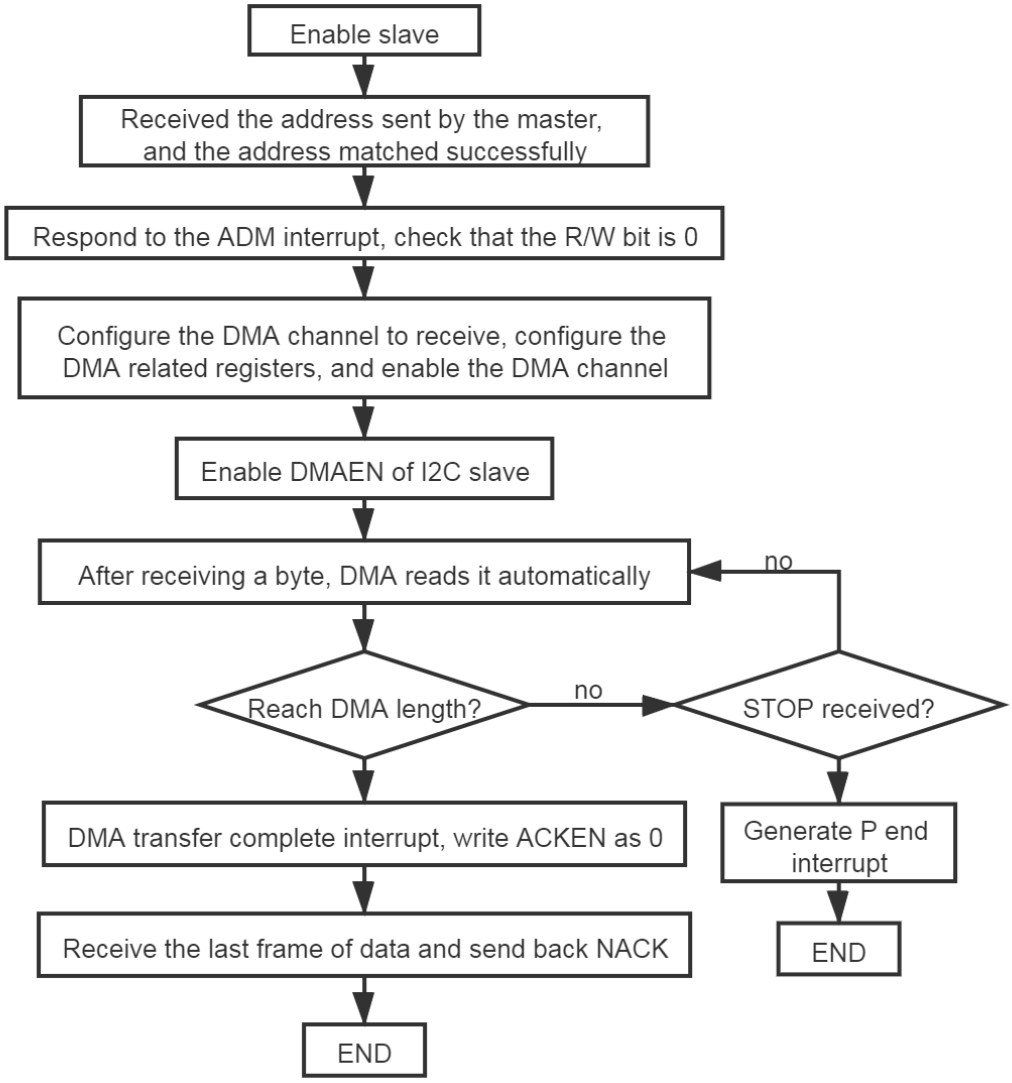


Figure 19-25 I²C slave DMA receiving flowchart

When the I²C slave receives the correct address, it generates the ADM interrupt flag. After the software responds to the interrupt, it queries the received R/W bit. If it is 1, the master is ready to read data from the slave. At this time, the software needs to read the SSPBUF to clear the BF flag, then configure the specific DMA channel as I2C_TX, and enable the DMAEN of the I²C slave; then when the slave data buffer SSPBUF is empty, it will generate a DMA request to notify the DMA to write to the SSPBUF.

Only the master sends back NACK to end the read operation. When the read data length is greater than the transfer length set by DMA, since DMA no longer responds to I²C requests, the slave will pull down SCL until the software disables the I²C slave module.

The flow of the slave using DMA to send is as follows:

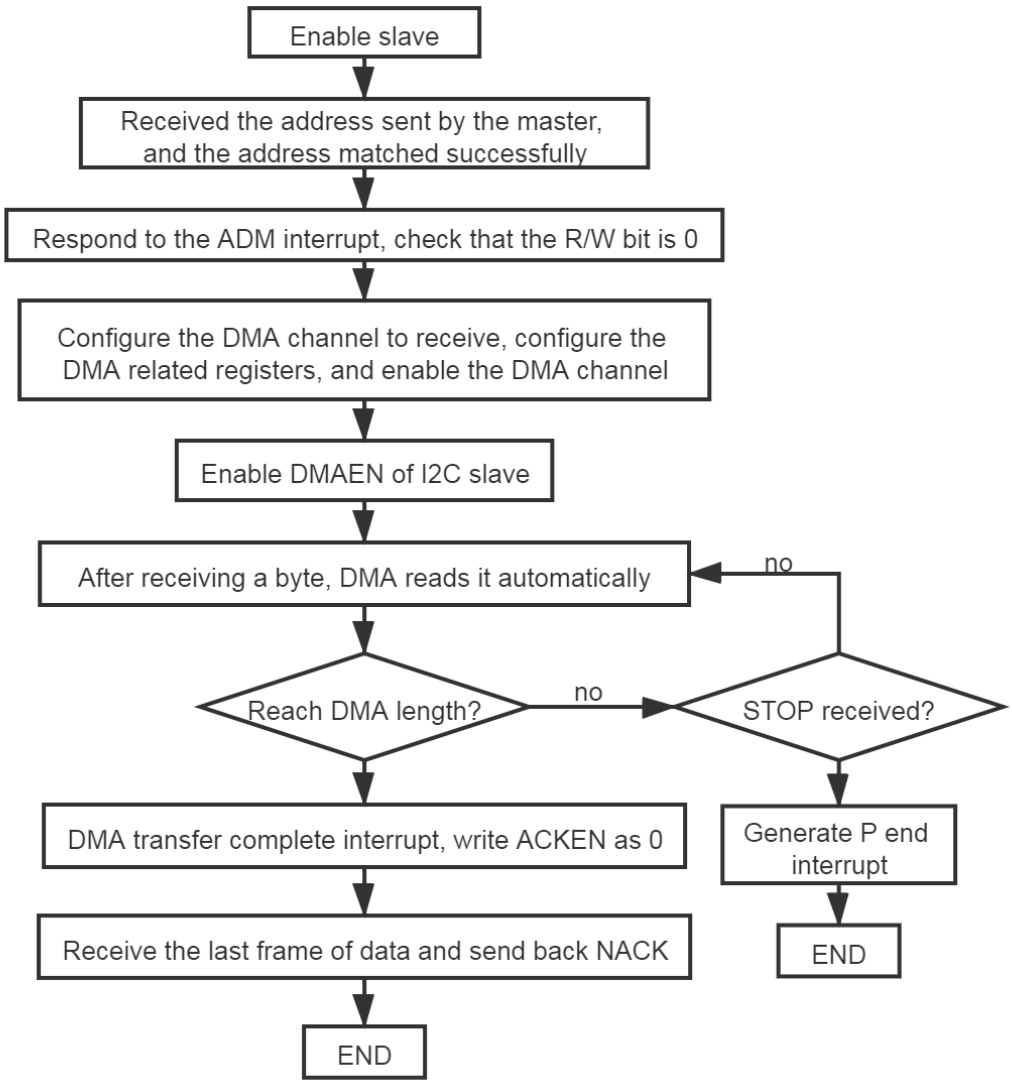


Figure 19-26 I²C slave DMA sending flowchart

19.9.6 Slave timing

Since the data transmission and reception of the slave only uses SCL, some analog delay is

needed to realize the data establishment and retention time control of SDA, and the timing of SCL is completely controlled by the master.

The timing control of the slave is shown in the figure below. According to I²C protocol requirements, the minimum data retention time of SDA relative to the falling edge of SCL is 0ns, that is, the slave can use the falling edge of SCL to send data to meet the requirements. However, considering the actual fall time of the SCL waveform on the bus, in order to better cover the retention time requirements, an extra RC delay greater than 300ns is added to the SDA output. This delay only needs to be applied to the SDA output of the I²C slave (SSP_SDAO).

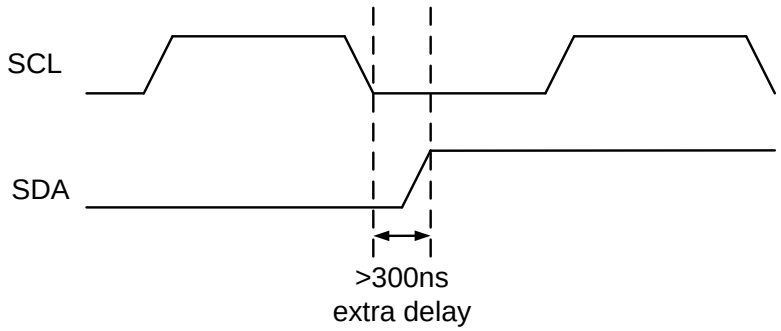


Figure 19-27 SDA output delay waveform

19.10 Register

Offset	Name	Symbol
I2C(Base address:0x40012400)		
0x00000000	I2C Master Config Register	I2C_MSPCFGR
0x00000004	I2C Master Control Register	I2C_MSPCR
0x00000008	I2C Master Interrupt Enable Register	I2C_MSPIER
0x0000000C	I2C Master Interrupt Status Register	I2C_MSPISR
0x00000010	I2C Master Status Register	I2C_MSPSR
0x00000014	I2C Master Baud Rate Generator Register	I2C_MSPBGR
0x00000018	I2C Master Transfer Buffer Register	I2C_MSPBUF
0x0000001C	I2C Master Timing Control Register	I2C_MSPTCR
0x00000020	I2C Master Time-Out Register	I2C_MSPTOR
0x00000024	I2C Slave Control Register	I2C_SSPCR
0x00000028	I2C Slave Interrupt Enable Register	I2C_SSPIER
0x0000002C	I2C Slave Interrupt Status Register	I2C_SSPISR
0x00000030	I2C Slave Status Register	I2C_SSPSR
0x00000034	I2C Slave Transfer Buffer Register	I2C_SSPBUF
0x00000038	I2C Slave Address Register	I2C_SSPADR

19.10.1 I2C Master Config Register (I2C_MSPCFGR)

NAME	I2C_MSPCFGR							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-						AUTOEND	MSP_DMAEN
access	U-0						R/W-0	R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						TOEN	MSPEN
access	U-0						R/W-0	R/W-0

bit	name	functional description
31:18	-	**RFU: Reserved, read as 0**
17	AUTOEND	Master DMA Automatic Ending 1: When the DMA transfer of the specified length data is completed,

bit	name	functional description
		the STOP condition is sent automatically 0: Wait for software to take over after the DMA transfer of the specified length is completed
16	MSP_DMAEN	Master DMA Enable 0: DMA disable 1: DMA enable
15:2	-	RFU: Reserved, read as 0
1	TOEN	SCLPull-down Timeout Enable 1: Timeout enable, the timeout period is defined by the MSPTO register 0: Timeout disable
0	MSPEN	I2CMaster Mode Enable 1: I2C master mode enable 0: I2C master mode disable

19.10.2 I2C Master Control Register (I2C_MSPCR)

NAME	I2C_MSPCR							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				RCEN	PEN	RSEN	SEN
access	U-0				R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:4	-	**RFU: Reserved, read as 0**
3	RCEN	In master receive mode, receive enable 1: Receive enable 0: Receive disable In master communication, after the software sends the address byte, it switches the transfer direction to master reception by setting RCEN, and then it can receive data from the slave. RCNE remains 1 during the receiving process until the software sets PEN to send the STOP sequence.

bit	name	functional description
2	PEN	STOP condition generation enable bit, software writes1 to send STOP condition, cleared by hardware (Stop Enable)
1	RSEN	Repeated START timing generation enable bit, software writes 1 to send Repeated START condition, cleared by hardware (Repeated Start Enable)
0	SEN	START condition generation enable bit, software writes 1 to send START condition, cleared by hardware (Start Enable)

19.10.3 I2C Master Interrupt Enable Register (I2C_MSPIER)

NAME	I2C_MSPIER							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	WCOLE	OVTE	SE	PE	NACKE	TXIE	RXIE
access	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:7	-	**RFU: Reserved, read as 0**
6	WCOLE	Write Collision Interrupt Enable 1: Enable 0: Disable
5	OVTE	SCL Overtime Enable 1: Enable 0: Disable
4	SE	START Interrupt Enable 1: Enable 0: Disable
3	PE	STOP Interrupt Enable 1: Enable 0: Disable
2	NACKE	Master Mode Non-ACK Interrupt Enable 1: Enable

bit	name	functional description
		0: Disable
1	TXIE	Master Mode Trasnmit Done Interrupt Enable 1: Enable 0: Disable
0	RXIE	Master Mode Receive Done Interrupt Enable) 1: Enable 0: Disable

19.10.4 I2C Master Interrupt Status Register (I2C_MSPISR)

NAME	I2C_MSPISR							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	WCOL	OVT	S	P	ACKSTA	TXIF	RXIF
access	U-0	R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:7	-	**RFU: Reserved, read as 0**
6	WCOL	Write collision detection bit, MCU can only write MSPBUF after completing START timing or sending a completed read/write frame, otherwise write collision occurs; hardware set, software writes 1 to clear (Write Collision Interrupt Flag) 1: Write collision occurred 0: No collision
5	OVT	SCLovertime interrupt flag, only works when TOEN is 1 (SCL OverTime Interrupt Flag) 1:SCL overtime occurred 0:No SCL overtime occurred
4	S	START timing sending completion interrupt flag, hardware set, cleared after reading (Start Interrupt flag)
3	P	STOP timing sending completion interrupt flag, hardware set, cleared after reading(Stop Interrupt flag)
2	ACKSTA	Response signal from the slave in master sending mode; this flag

bit	name	functional description
		can generate an interrupt when a NACK is received after a master send; hardware set, software write 1 to clear (Acknowledge Status Flag) 1: Slave responds to NACK 0: Slave responds to ACK
1	TXIF	I2C master mode transmit done interrupt flag, hardware set, software write 1 to clear(Trasnmit Done Interrupt Flag) This flag register is set after the master receives the ACK or NACK sent back from the slave.
0	RXIF	I2C master mode receive done interrupt flag, hardware set, software write 1 to clear (Receive Done Interrupt Flag) This flag register is set after the master sends back an ACK or NACK.

19.10.5 I2C Master Status Register (I2C_MSPSR)

NAME	I2C_MSPSR							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-		BUSY	RW	-	BF	-	ACKMO
access	U-0		R-0	R-0	U-0	R-0	U-0	R/W-0

bit	name	functional description
31:6	-	RFU: **Reserved, read as 0**
5	BUSY	I2C Communication Status Bit (Busy) 1: The interface is in the read/write state and data transfer is in progress 0: Data transfer has been completed
4	RW	I2CTransfer Direction Status Bits (Read/Write) 1: The master reads data from the slave 0: The master writes data to the slave
3	-	RFU: **Reserved, read as 0**
2	BF	Buffer Full Status Bit (Buffer Full)

bit	name	functional description
		Receive. 1: Reception complete, MSPBUF full 0: Reception not complete, MSPBUF empty Send. 1: Transmitting, MSPBUF full 0: Sending complete, MSPBUF empty
1	-	RFU: **Reserved, read as 0**
0	ACKMO	Status of the master response signal in master receive mode (Ack Master output) 1: The master sends back a NACK 0: The master sends back a ACK ***Note: The P flag register must be cleared before the software can set ACKMO***

19.10.6 I2C Master Baud Rate Generation Register (I2C_MSPBGR)

NAME	I2C_MSPBGR							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							MSPBRGH[8]
access	U-0							R/W-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	MSPBRGH[7:0]							
access	R/W-0001 0011							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							MSPBRGL[8]
access	U-0							R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	MSPBRGL[7:0]							
access	R/W-0001 0011							

bit	name	functional description
31:25	-	RFU: **Reserved, read as 0**
24:16	MSPBRGH	The width of the SCL clock low level, counted by the I2C operating clock (Master SCL High level length)
15:9	-	RFU: **Reserved, read as 0**
8:0	MSPBRGL	The width of the SCL clock low level, counted by the I2C operating clock. (Master SCL Low level length)

19.10.7 I2C Master Transfer Buffer Register (I2C_MSPBUF)

NAME	I2C_MSPBUF							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	MSPBUF							
access	R/W-0000 0000							

bit	name	functional description
31:8	-	RFU: **Reserved, read as 0**
7:0	MSPBUF	MSPBUF[7:0]:The reading and writing of data is accomplished through the operation of MSPBUF. When sending, a write operation is performed to the MSPBUF and the data send/receive shift register (MSPSR) is also loaded; when receiving, the MSPBUF and MSPSR form a double buffer structure and the data is read out to the MSPBUF. After receiving a byte of data, the MSPSR loads the data into the MSPBUFand at the same time sets the I2CIF. MSPSR is not a direct register and has no physical address. (Master data Buffer)

19.10.8 I2C Master Timing Control Register (I2C_MSPTCR)

NAME	I2C_MSPTCR							
Offset	0x0000001C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

name	-							SDAHD[8]
access	U-0							R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SDAHD[7:0]							
access	R/W-0000 1010							

bit	name	functional description
31:9	-	RFU: **Reserved, read as 0**
8:0	SDAHD	Defines the SDAhold time parameter with respect to the falling edge of SCL, counted by the I2Coperating clook (SDA hold delay) Note: The minimum effective value is 1, the maximum effective value is MSPBRGL

19.10.9 I2C Master Time Out Register (I2C_MSPTOR)

NAME	I2C_MSPTOR							
Offset	0x00000020							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				TIMEOUT[11:8]			
access	U-0				R/W-1111			
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TIMEOUT[7:0]							
access	R/W-1111 1111							

bit	name	functional description
31:12	-	RFU: **Reserved, read as 0**
11:0	TIMEOUT	Defines the slave SCL low stretching timeout period, which can be rewritten by software with MSPEN=0 (SCL stretching Time Out) $T_{SCL\_STRETCHING\_TIMEOUT}$=TIMEOUT[11:0] * T_{SCL}

19.10.10 I2C Slave Control Register (I2C_SSPCR)

NAME	I2C_SSPCR							
Offset	0x00000024							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-00000000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-00000000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						SCLSEN	SSP_DMAEN
access	U-00000000						R/W-1	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-			ACKEN	SDAO_DLYEN	SCLI_ANFEN	A10EN	SSPEN
access	U-0			R/W-1	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:10	-	RFU: **Reserved, read as 0**
9	SCLSEN	I2C slave clock stretching enable (SCL Stretching Enable) 0: Disable slave clock stretching 1: Enable slave clock stretching *Note: When the slave uses DMA communication, SCLCEN must be set to 1*
8	SSP_DMAEN	I2C slave DMA enable 1: Enable DMA function 0: Disable DMAfunction
7:5	-	RFU: **Reserved, read as 0**
4	ACKEN	ACK enable bit (Slave Ack Enable) 1: Slave will send back ACK after receiving 0: Slave does not send back ACK
3	SDAO_DLYEN	SDAslave output delay enable 0: Bypass slave SDA output delay 1: Enable slave SDA output delay
2	SCLI_ANFEN	SCL slave input analog filter enable 0: Bypassanalog filter 1: Enable analog filter
1	A10EN	10bit Slave address enable 1: Slave uses 10bit address 0: Slave uses7bit address
0	SSPEN	I2C slave enable bit

bit	name	functional description
		1: Enable I2C slave 0: Disable I2C slave

19.10.11 I2C Slave Interrupt Enable Register (I2C_SSPIER)

NAME	I2C_SSPIER							
Offset	0x00000028							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ADEE	SE	PE	WCOLE	SSPOVE	ADME	TXIE	RXIE
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:8	-	RFU: Reserved, read as 0
7	ADEE	Slave Address Error Interrupt Enable, 1 is valid
6	SE	Start Interrupt Enable, 1 is valid
5	PE	Stop Interrupt Enable, 1 is valid
4	WCOLE	Write Collision Interrupt Enable, 1 is valid
3	SSPOVE	Slave Buffer Overflow Interrupt Enable, 1 is valid
2	ADME	Slave address match interrupt enable, 1 is valid
1	TXIE	Transmit Complete Interrupt Enable, 1 is valid
0	RXIE	Receive Complete Interrupt Enable, 1 is valid

19.10.12 I2C Slave Interrupt Status Register (I2C_SSPSR)

NAME	I2C_SSPISR							
Offset	0x0000002C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							

NAME	I2C_SSPISR							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ADE	S	P	WCOL	SSPOV	ADM	TXIF	RXIF
access	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:8	-	RFU: **Reserved, read as 0**
7	ADE	Address error flag, hardware set, write 1 to clear In the case of a 7bit address, the address byte starting with 11110 is received, or when the first byte does not start with 11110 in the case of a 10bit address, ADEE is triggered.
6	S	The start sequence is detected, hardware set, software is automatically cleared after reading (Start flag)
5	P	The stop sequence is detected, hardware set, software is automatically cleared after reading (Stop flag)
4	WCOL	Write Collision flag, hardware set, write 1 to clear 1: In the case of BF=1, the software writes new data to SSPBUF 0:No write collision When WCOL occurs, new data will be discarded
3	SSPOV	Slave buffer overflow flag, hardware set, write 1 to clear 1: In the case of BF=1, the slave receives new data 0:No receive overflow If the slave enables SCL stretching, the received data will not overflow; therefore, SSPOV can only be set when SCLSEN=0.
2	ADM	Slave address matched flag, hardware set, write 1 to clear 1:The received 7bit or 10bit address is consistent with the contents of the SLAVE_ADDR register 0: The received address is inconsistent with SLAVE_ADDR
1	TXIF	I2C slave transmit interrupt flag, hardware set, write 1 to clear
0	RXIF	I2Cslave receive interrupt flag, hardware set, write 1 to clear

19.10.13 I2C Slave Status Register (I2C_SSPSR)

NAME	I2C_SSPSR							
Offset	0x00000030							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							

NAME	I2C_SSPSR							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				BUSY	RW	DA	BF
access	U-0				R-0	R-0	R-0	R-0

bit	name	functional description
31:4	-	RFU: **Reserved, read as 0**
3	BUSY	Slave communication flag (Busy) 1: Slave data receiving and sending 0: Slave idle
2	RW	Read/write direction status register (Read/Write) 1: The slave receives R/W=1, and the slave needs to send data to the master 0:Slave is in the state of receiving data
1	DA	Data/addressframe indication 1:The last byte received is data 0:The last byte received is the address
0	BF	Slave buffer full flag 1:SSPBUFfull 0:SSPBUFempty

19.10.14 I2C Slave Transfer Buffer Register (I2C_SSPBUF)

NAME	I2C_SSPBUF							
Offset	0x00000034							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SSPBUF							

NAME	I2C_SSPBUF
access	R/W-0000 0000

bit	name	functional description
31:8	-	RFU: **Reserved, read as 0**
7:0	SSPBUF	SSPBUF[7:0]: The reading and writing of data is accomplished through the operation of SSPBUF. When sending, a write operation is performed to the SSPBUF and the data send/receive shift register (SSPSR) is also loader; when receiving, the SSPBUF and SSPSR form a double buffer structure and the data is read out to the SSPBUF. After receiving a byte of data, the SSPSR loads the data into the SSPBUF and at the same time sets the I2CIF. SSPSR is not a direct register and has no physical address. (Slave Buffer)

19.10.15 I2C slave Address Register (I2C_SSPADR)

NAME	I2C_SSPADR							
Offset	0x00000038							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						SSPADDR[9:8]	
access	U-0						R/W-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SSPADDR[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:10	-	RFU: **Reserved, read as 0**
9:0	SSPADDR	Slave Address Register A10EN = 1, 10 bits are valid A10EN = 0, only the lower 7 bits are valid

20 UART0/1/4/5

20.1 Introduction

Main features:

- Baud rate software configurable
- 4 independent channels (UART0, UART1, UART4, UART5)
- Full duplex communication port
- Support RX-TX port switching
- UART with data reception completion/reception error interrupt with error type indication
- Configurable data length, supports 6, 7, 8, 9bits
- Configurable stop bits - supports 1 stop bit or 2 stop bits
- Configurable IR modulation output function, and carrier frequency, and carrier duty cycle can be set
- Support DMA
- Support receive timeout mechanism（UART0, UART1, UART4, UART5）
- Support transmit delay function（UART0, UART1）
- Support RXD falling edge hibernation wake-up（UART0, UART1, UART4, UART5）

20.2 UART Block Diagram

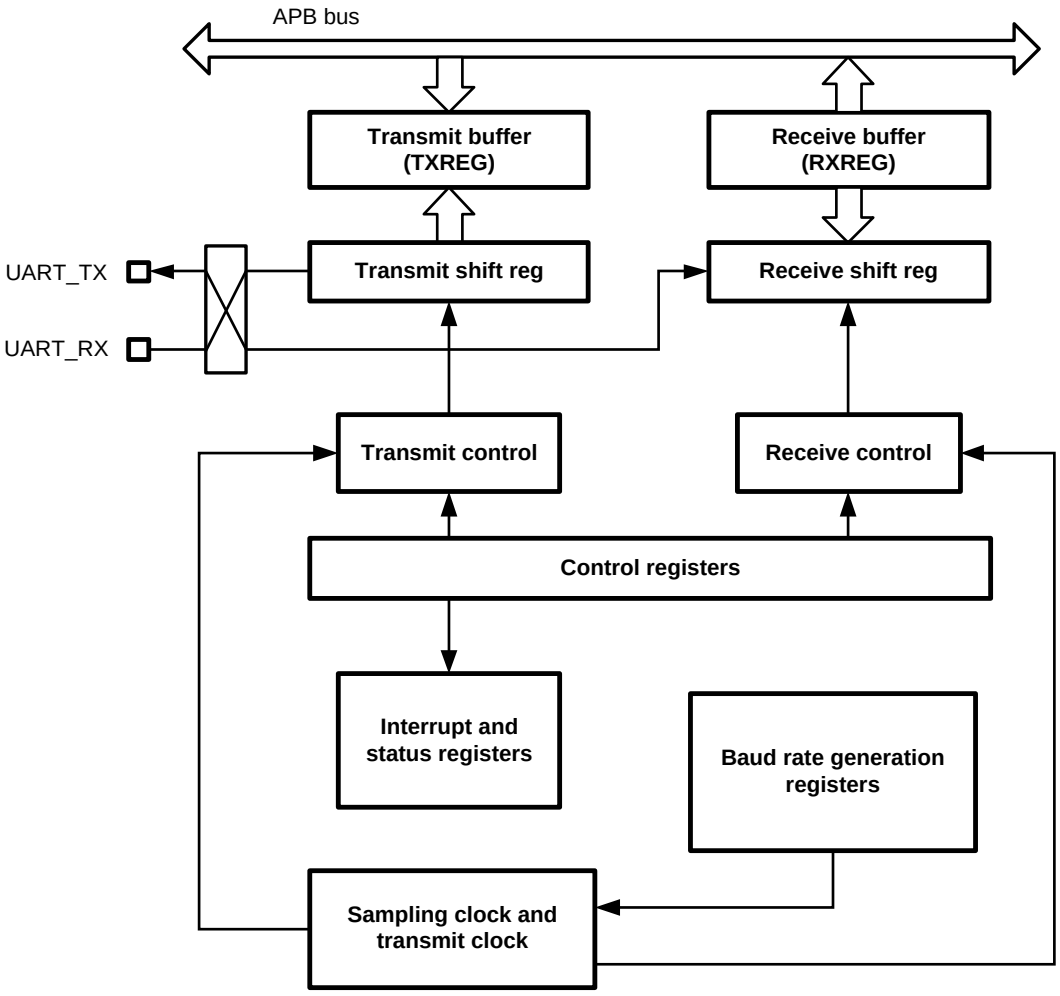


Figure 20-1 UART Block Diagram

20.3 Pin definition

The UART module has two pins to communicate with external devices, and each UART's transceiver signal may be mapped to a different GPIO, which is show as the table below.

Pin		UARTx	Symbol	Function
PA2	PA13	UART0	UART0_RX	Data reception
PA3	PA14		UART0_TX	Data transmission
PC2	PB13	UART1	UART1_RX	Data reception
PC3	PB14		UART1_TX	Data transmission
PA0	PB2	UART4	UART4_RX	Data reception
PA1	PB3		UART4_TX	Data transmission
PC4	PD0	UART5	UART5_RX	Data reception
PC5	PD1		UART5_TX	Data transmission

Table 20-1 UART Pin list

When the UART function is mapped to multiple pins at the same time:

- PA2 and PA13 are configured as digital peripheral functions at the same time
  - Only the RX signal on PA2 will be input into the module
- PC2 and PB13 are configured as digital peripheral functions at the same time
  - Only the RX signal on PA2 will be input into the module
- PC4 and PD0 are configured as digital peripheral functions at the same time
  - Only the RX signal on PC4 will be input into the module
- PA0 and PB2 are configured as digital peripheral functions at the same time
  - Only the RX signal on PA0 will be input into the module
- When the UART sending function is mapped to multiple GPIOs at the same time, these pins will send data at the same time

20.4 UART Mode

FM33L0xxintegrates different types of UART (LPUART), and the differences are shown in the following table：

UART Character	UART0/1	UART2	UART4/5	LPUART0/1
DMA support	Y	Y	Y	Y
Half Duplex/Full Duplex	Y	Y	Y	Y
The infrared emission	Y	Y	Y	-
Dual clock domain (working clock independent of bus)	Y	Y	-	Y
Sleep wake-up	Y	Y	Y	Y
Receive Timeout	Y	Y	-	-
Transmission Delay	Y	Y	-	-
Data Length	6、7、8、9bits			
LIN support	N	Y	N	

Table 20-2 UART Mode

20.5 UART Character Format

The basic timing of UART transmission characters is shown in the figure below. Each character contains at least 1bit START and at least 1bit STOP bits, data lengths can be configured to be 6-9bits, and parity bit can be selected.

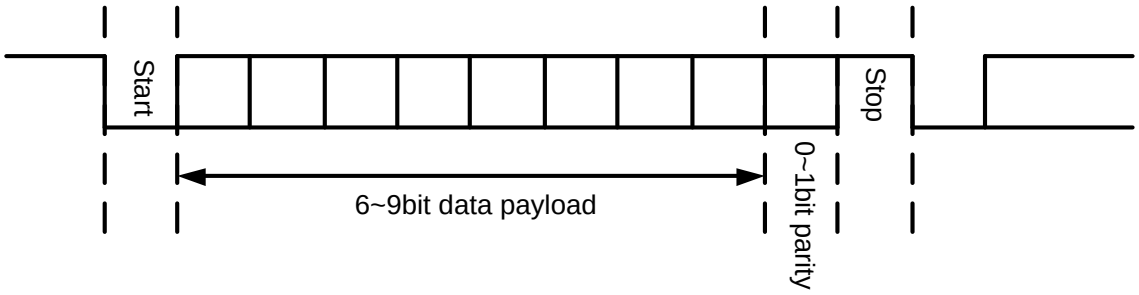


Figure 20-2 UART Character Format

UART supports multiple frame formats controlled by the UARTxCSR.PDSEL register and UARTxCSR.PARITY register, which are listed in the table below:

PDSEL	PARITY	Frame Format[1]
00	00	[Start | 7 bits data | Stop]
	01, 10	[Start | 7 bits data | Parity | Stop]
01	00	[Start | 8 bits data | Stop]
	01, 10	[Start | 8 bits data | Parity | Stop]
10	00	[Start | 9 bits data | Stop]
	01, 10	[Start | 9 bits data | Parity | Stop]
11	00	[Start | 6 bits data | Stop]
	01, 10	[Start | 6 bits data | Parity | Stop]

Table 20-3 UART Data Frame Format

[1]：The Stop bit can be either 1bit or 2bits, depending on the STOPCFG register.

Note: THE PDSEL register is used to configure the data length of the frame. The communication frame length is [start bit + data bit + check bit + stop bit].

20.6 UART Function Description

20.6.1 Clock structure

UART0 and UART1 implement dual clock structure：

- The bus register clock is represented by PCLK, derived from APBCLK. PCLK must be enabled when the CPU or DMA needs to access the UART internal registers.
- Data transfer clock is represented by UCLK, which can not only come from APBCLK, but also from RCHF, SYSCLK, RC4M, which can work independently of APBCLK. UCLK must be enabled before data transmit-receive

The control of PCLK and UCLK are implemented in the CMU module. The corresponding CMU control registers must be correctly configured before UART communication.

The dual clock structure can make UART0 and UART1 unlimited to the configuration of APBCLK. When some peripherals need to work on high APBCLK frequency, UART can still work on the reduced frequency. Or conversely, the CPU operates at a lower frequency and UART data communication remains at a higher baud rate.

Theoretically, there is no relative relation between PCLK and UCLK. UCLK can be faster or slower than PCLK. However, the application needs to pay attention to whether the CPU or DMA has enough time for data transfer when the frequency difference between the two is significant.

Different from UART0 and UART1, UART4 and UART5 adopt a single clock structure. At this time, UCLK=PCLK, and the data sending and receiving clock of UART is also derived from APBCLK.

20.6.2 Bit Receiving Sampling

UART oversampled the received data 16 times within one baud, and two out of three majority decision is made at the middle position of each bit to improve noise immunity.

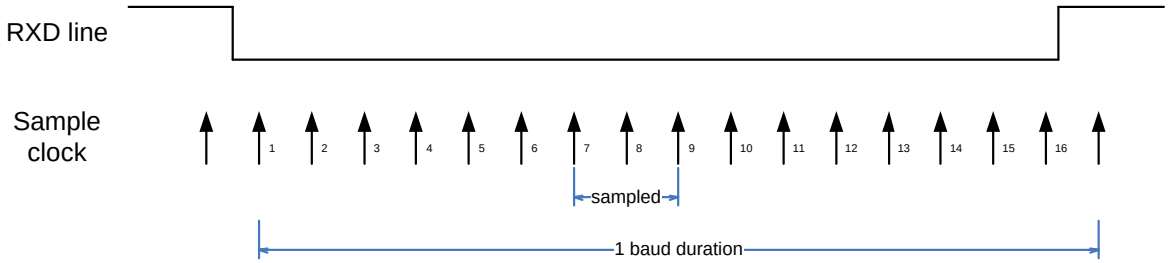


Figure 20-3 Bit Sampling

The bits received into the receive shift register are the result of majority decisions. For example, if the result of the three samples is 001, the result is 0; If it's 011, the result is 1.

Since UART oversamples the input signals 16 times, SPBRG configuration is required to be no less than 16, that is, the UART working clock must be at least 16 times higher than baud rate。

20.6.3 Data Transmission

Transmit shift register (TSR) is used to serialize the data out. The outgoing data must first be written to the TX buffer. When the software sets the TXEN register, if the TX buffer is not empty, UART will load the buffer data into TSR and begin to shift out.

Note: Since the register operation clock and baud rate clock are asynchronous, when transmitting starts, the TSR has to wait for baud rate clock. Therefore, there is a maximum delay of 1 BAUD between TXEN setting and UART transmitting.

TXBE and TXSE are interrupt flag means TX buffer empty and TSR empty, respectively. The software can choose to generate a send completion interrupt at the appropriate point in time.

Software needs to set the Baud rate register SPBRG first, and set TXEN to 1, then write TXBUF register to start transmitting. You can also set the baud rate SPBRG, then write the TXBUF register, and set the TXEN later. If software clears TXEN during transmission, the data transmission will be terminated, and the TX buffer will also be reset.

The following is an example of UART transmitting asynchronously. In this example, the software first writes data to TXBUF, and then set TXEN.

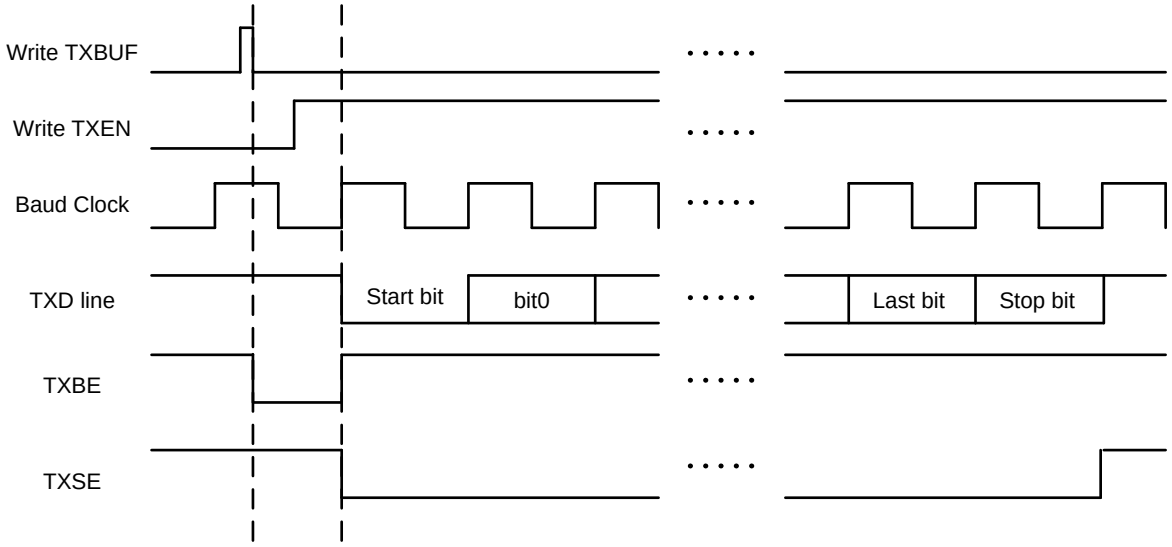


Figure 20-4 UART asynchronous transmission waveform 1

The recommended steps in the figure are as follows:

- Select the appropriate baud rate and initialize SPBRG
- If an interrupt is required, set TXSE_IE or TXBE_IE
- Determine the format of data transmission: Set PDSEL register, determine the length of data transmission; Set the PARITY register to choose whether to send a PARITY bit and the type of PARITY, and set the STOPSEL register to decide whether to send a 1-bit or 2-bit stop
- If the data to send is infrared modulated, write the appropriate value to the IRCON register to obtain the corresponding modulation frequency and duty cycle, and set TXIREN
- Write the data to TXBUF register
- Enable transmission: set TXEN

The software can also set TXEN first and then write TXBUF. UART will immediately start the sending process after the data is written to TXBUF.

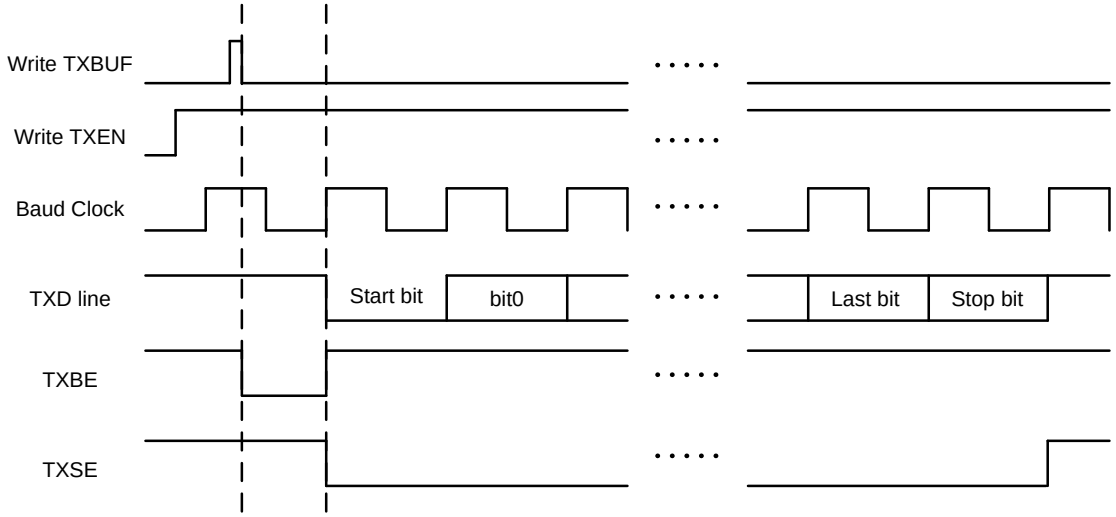


Figure 20-5 UART asynchronous transmission waveform 2

When TXBUF is empty, the software can immediately write the next data to be sent, in order to achieve continuous data transmission without interval.

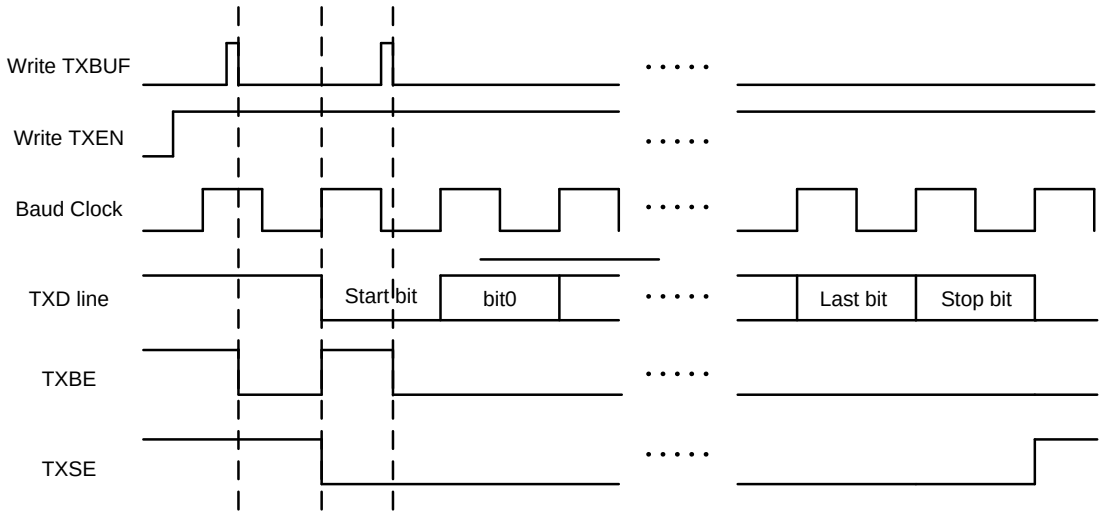


Figure 20-6 UART asynchronous transmission waveform 3

20.6.4 Data Reception

The serial data receiving of UART uses a receive shift register (RSR). When the stop bit is received, RSR feeds the received data into the receive buffer (RXBUFFER). The interrupt flag RXBF is set to 1 after each received byte is copied into the receive buffer. When new data is received when RXBUF is full, the original data in the RX buffer will be overwritten, and RXBF flag is set again. Meanwhile, receive overflow error occurs, and OERR is set to 1. The OERR flag can be cleared by writing 1by software or reading RXBUF.

During the receiving process, if the correct stop bit is not detected, a frame format error occurs and FERR is set to 1. If a parity error occurs, flag bit PERR is set to 1.

The recommended asynchronous receive procedure is as follows:

- Select the appropriate baud rate and initialize SPBRG
- If an interrupt is required, set RXBF_IE
- Set the format of data receiving: set PDSEL register to determine the length of data sent;
- Set the PARITY register to choose whether to send a PARITY bit and the type of PARITY, and set the STOPSEL register to decide whether to send a 1-bit or 2-bit stop
- Enable receiving: Set RXEN
- At the end of a frame, the RXBF bit is set to 1. If the RXBF_IE bit is set to 1, an interrupt will be generated
- Read PERR, FERR, and OERR registers to determine if there are any data error or overflow

Read the received data in the RXBUF register

20.6.5 Low Power Hibernation Wakeup

The UART supports wake-up from chip hibernation triggered by the falling edge of RXD. When the NEWUP register is set, the falling edge event (low level duration >100ns) on RXD input will wake up the chip from hibernation mode, with this function, the UART can receive data in hibernation mode.

The software configuration method is as follows:

- Configure UART register to enable NEWUP
- Configure the UART operating clock as RCHF, and configure the baud rate divider register as needed.
- Configure the corresponding GPIO as UART data receiving function
- Set RXEN to enable reception
- Set the chip to hibernate and wait for the UART reception event.

20.6.6 Use DMA for UART communication

When DMA is enabled, UART will automatically generate the corresponding DMA request when the TX buffer is empty or the RX buffer is full. Application needs to configure DMA channel connections, set the RAM pointer, and enable DMA channels. After that, the DMA will automatically respond to the UART request and complete the data transfer between RAM and UART.

Application example: DMA for UART0 receive

- Configure DMA channel 1 or 3 as RXD0
- Set corresponding channel parameters: RAM pointer, address increment and decrement, channel priority, transmission length, interrupt settings, etc.
- Enable corresponding DMA channels.

- Configure UART parameters.
- The enabled UART waits for the data to be received
- UART automatically generates DMA requests when RX buffer is full
- DMA responds to the request, reads the UART RXBUF, and writes to the specified RAM address.

20.6.7 Transmission completion interrupt in DMA mode

When UART transfers data through DMA, DMA will produce a DMA channel interrupt after a specified length of data has been transferred. But when the channel interrupt occurs, the last frame of data has just been written to the UART TXBF and has not yet been sent. By configuring the DMATXIFCFG register, it is possible to generate a transmit completion interrupt (buffer empty or shift register empty) when the last frame data has been sent. Which will interrupt CPU after all data has been sent.

The software procedure is described as follows:

- Configure DMA channels UART TX
- Disable the DMA channel interrupt.
- Set UART TXBE_IE or TXSE_IE registers to allow interrupts to be generated.
- Set the DMATXIFCFG register, allowing only the last frame of data to produce interrupt output.
- Prepare data to send. Enable the DMA.
- UART transmits continuously until the last frame, no TXBE or TXSE interrupts are generated.
- After the last frame is sent, UART produces a TXBE or TXSE interrupt.

The following table assumes that UART sends N frames via DMA:

TXBE_IE TXSE_IE	DMATXIFCFG	Frame No.	TXBETXSE	UART interrupt
0	x	1~N	After each frame is sent, then set.	Inactive
1	0	1~N	After each frame is sent, then set.	Inactive
	1	1~N-1	After each frame is sent, then set.	Inactive
		N	After each frame is sent, then set.	Active

Table 20-3 The status of the interrupt flag when UART uses DMA to send data

20.7 Baud Rate Generation

20.7.1 Baud rate generation

The Baud rate register is a 16-bit register whose value X is any integer between 16 and 65535.

Baud rate calculation formula:

Baud = F_{CLK} / (SPBRG+1)；

Note: *FCLK can be different clocks in different UART. For UART2~5, FCLK is APBCLK. For UART0 and UART1, FCLK is a working clock independent of APBCLK.*

To support full duplex communication, the receiving and transmitting baud rates are generated separately.

The following table shows the baud rate at common system clock frequencies:

Baud	F_{CLK}=16MHz			F_{CLK}=8MHz		
bps	Actual (bps)	Error%	X+1	Actual (bps)	Error%	X+1
300	300.0019	0.000625	53333	299.9963	-0.00125	26667
1200	1200.03	0.0025	13333	1199.94	-0.005	6667
2400	2399.88	-0.005	6667	2400.24	0.010001	3333
4800	4800.48	0.010001	3333	4799.04	-0.02	1667
9600	9598.08	-0.02	1667	9603.842	0.040016	833
19200	19207.68	0.040016	833	19184.65	-0.07994	417
38400	38369.3	-0.07994	417	38461.54	0.160256	208
57600	57553.96	-0.07994	278	57553.96	-0.07994	139
115200	115107.9	-0.07994	139	115942	0.644122	69
230400	231884.1	0.644122	69	228571.4	-0.79365	35
460800	457142.9	-0.79365	35	470588.2	2.124183	17

Baud	F_{CLK}=24MHz			F_{CLK}=32MHz		
bps	Actual (bps)	Error%	X+1	Actual (bps)	Error%	X+1
300	300	0	80000	299.9991	-0.00031	106667
1200	1200	0	20000	1199.985	-0.00125	26667
2400	2400	0	10000	2400.06	0.0025	13333
4800	4800	0	5000	4799.76	-0.005	6667
9600	9600	0	2500	9600.96	0.010001	3333
19200	19200	0	1250	19196.16	-0.02	1667
38400	38400	0	625	38415.37	0.040016	833
57600	57553.96	-0.07994	417	57553.96	-0.07994	556
115200	115384.6	0.160256	208	115107.9	-0.07994	278
230400	230769.2	0.160256	104	230215.8	-0.07994	139
460800	461538.5	0.160256	52	463768.1	0.644122	69

Table 20-4 Common clock frequency baud rate calculation

20.7.2 Adaptive Baud Rate Generation

Using the Capture function of Timer, the adaptive baud rate can be realized. Generally，the external UART device sends a frame according to the negotiated data content (such as 0xF8) at target baud rate, and the timer counts the high level pulse width of the frame data. The MCU reads the timer capture result to calculate the Baud rate, and writes it into the baud rate register. Then the following data can be received with new baud rate. Refer to the Timer section.

20.8 Infrared modulation

The UART_IRCR register holds an 11-bit frequency divider TZBRG, whose value is any integer between 0 and 2047. All UART share one infrared modulation generator.

Infrared modulation frequency calculation formula:
FIR = F_{APBCLK}/ (TZBRG+ 1)

The infrared modulation method is: when sending data 0, the infrared frequency is modulated; when sending data 1, the infrared frequency is not modulated.

In order to meet the needs of PNP and NPN infrared optical transistor, register IRFLAG bit controls the polarity of infrared modulation output.

When IRFLAG=0, it is positive polarity output, suitable for PNP.

When IRFLAG=1, it is negative polarity output, suitable for NPN.

The TH register is used to configure the ir modulation duty cycle.
Duty ratio：Y = (TZBRG[10:4] * TH) /(TZBRG+1)
When TH=4’b0000，the duty ratio is Y = (TZBRG[10:1]+1)/(X+1)；

When TZBRG[10:4]=7 'h00, the duty ratio is Y = TH/(TZBRG[3:0]+ 1);

If this time TH>TZBRG [3:0], then ir modulation clock IRCLK is fixed at high level.
When the infrared modulation polarity is reversed (IRFLAG=1), the duty cycle is also 1-y

The infrared modulation waveform is shown in the following figure:

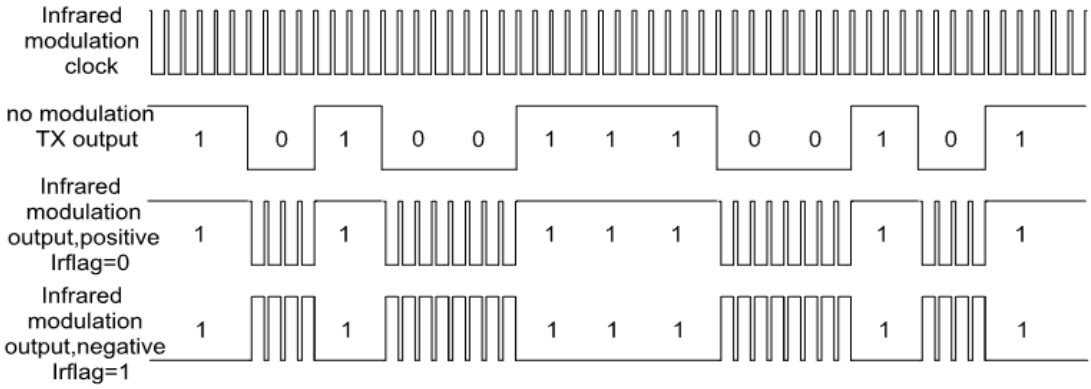


Figure 20-7 The infrared modulation waveform

Duty cycle is defined as the high-level length/period regardless of whether the effective level is 0 or 1.

20.9 Receive timeout

A time-out mechanism is designed for time-sensitive applications such as MODBUS. When the RXTOEN register is enabled, the timeout counter is counted at the baud rate clock. Each time a complete data frame is received, the timeout counter is cleared and the count is restarted. The upper limit of the timeout overflow can be configured by the software with a maximum of 255 baud.

***Note**: Receive timeout function is not supported in UART4 and UART5.*

20.10 Transmit Delay

Through the TXDLY_LEN register, you can control the time interval between two data frames sent, the unit is Baud. The transmit delay is the interval between the end of the last STOP bit of the previous frame and the start bit of the next frame.

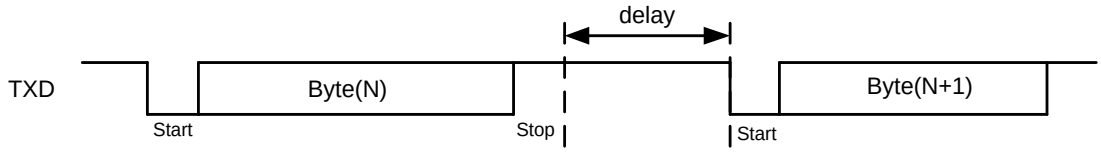


Figure 20-8 UART Transmit Delay

***Note**: UART4 and UART5 does not support sending delay function.*

20.11 Register

Offset	Name	Symbol
UART Public register(module base address：0x40019C00)		
0x00000000	Infrared modulation Control Register	UART_IRCR
UART0Register(module base address：0x40011C00)		
0x00000000	UART0 Control Status Register	UART0_CSR

Offset	Name	Symbol
0x00000004	UART0 Interrupt Enable Register	UART0_IER
0x00000008	UART0 Interrupt Status Register	UART0_ISR
0x0000000C	UART0 Time-Out and Delay Register	UART0_TODR
0x00000010	UART0 Receive Buffer	UART0_RXBUF
0x00000014	UART0 Transmit Buffer	UART0_TXBUF
0x00000018	UART0 Baud rate Generator Register	UART0_BGR
UART1 Register(module base address：0x40012000)		
0x00000000	UART1 Control Status Register	UART1_CSR
0x00000004	UART1 Interrupt Enable Register	UART1_IER
0x00000008	UART1 Interrupt Status Register	UART1_ISR
0x0000000C	UART1 Time-Out and Delay Register	UART1_TODR
0x00000010	UART1 Receive Buffer	UART1_RXBUF
0x00000014	UART1 Transmit Buffer	UART1_TXBUF
0x00000018	UART1 Baud rate Generator Register	UART1_BGR
UART4 Register(module base address：0x4001A000)		
0x00000000	UART4 Control Status Register	UART4_CSR
0x00000004	UART4 Interrupt Enable Register	UART4_IER
0x00000008	UART4 Interrupt Status Register	UART4_ISR
0x00000010	UART4 Receive Buffer	UART4_RXBUF
0x00000014	UART4 Transmit Buffer	UART4_TXBUF
0x00000018	UART4 Baud rate Generator Register	UART4_BGR
UART5 Register(module base address：0x4001A400)		
0x00000000	UART5 Control Status Register	UART5_CSR
0x00000004	UART5 Interrupt Enable Register	UART5_IER
0x00000008	UART5 Interrupt Status Register	UART5_ISR
0x00000010	UART5 Receive Buffer	UART5_RXBUF
0x00000014	UART5 Transmit Buffer	UART5_TXBUF
0x00000018	UART5 Baud rate Generator Register	UART5_BGR

20.11.1 Infrared Modulation Register（UART_IRCR）

NAME	UART_IRCR							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	IRFLAG	TH				TZBRG[10:8]		
access	R/W-0	R/W-0000				R/W-000		
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TZBRG[7:0]							

access	R/W-1101 0010

Bit	Name	Functional description
31:16	-	**Reserved, read as 0**
15	**IRFLAG**	Controls the default output polarity when infrared modulation sends data. 0：Positive polarity 1：Negative polarity
14:11	**TH**	Transmission High Duty
10:0	**TZBRG**	Transmission Baud Rate

20.11.2 UARTx Control status register（UARTx_CSR）

NAME	UARTx_CSR(x=0,1,4,5)							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							BUSY
access	U-0							R-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-						TXIREN	RXTOEN
access	U-0						R/W-0	R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			IOSWAP	-	DMATXIFCFG	BITORD	STOPCFG
access	U-0			R/W-0	U-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	PDSEL		PARITY		RXPOL	TXPOL	RXEN	TXEN
access	R/W-00		R/W-00		R/W-0	R/W-0	R/W-0	R/W-0

Bit	Name	Functional description
31:25	-	**Reserved, read as 0**
24	**BUSY**	UART communication flag, read-only 1：UART is communicating. 0：UARTIDLE
23:18	-	**Reserved, read as 0**
17	**TXIREN**	Send infrared modulation enablement. 1：Enable infrared modulation transmission. 0：Turn off infrared modulation sending.
16	**RXTOEN**	Receive timeout enablement. 1：Enable the receive timeout function. 0：Turn off receive timeout.
15:13	-	**Reserved, read as 0**
12	**IOSWAP**	Exchange of RX and TX pins. 0：The default pin order.(Consistent with package diagram.) 1：swap pin.
11	**NEWUP**	**UART wake-up low-power mode function enable, 1 valid**

Bit	Name	Functional description
10	**DMATXIFCFG**	DMA send completion interrupt, only valid if UART sends through DMA. 1：In the case of IE=1, interrupt signal output is allowed after the last frame is sent in DMA mode;Interrupt signal output is not allowed after the data frame before the last frame has been sent. 0：It is up to IE to decide whether to allow interrupt signal output.
9	**BITORD**	Bit order in which data is sent/received. 0：LSB first 1：MSB first
8	**STOPCFG**	Stop bit width configuration, only valid for sending frame format 0：1stop bit. 1：2stop bit.
7:6	**PDSEL**	Select the data length of each frame;This register is valid for both sending and receiving data. 00：7 data bit 01：8 data bit 10：9 data bit 11：6 data bit
5:4	**PARITY**	parity bit configuration; This register is valid for both sending and receiving data. 00：No parity bit 01：Even 10：Odd 11：RFU
3	**RXPOL**	Receive data polarity configuration. 0：standard 1：inverted
2	**TXPOL**	Transmit data polarity configuration. 0：standard 1：inverted
1	**RXEN**	Receive enable
0	**TXEN**	Transmit enable

20.11.3 UARTx Interrupt enable register（UARTx_IER）

NAME	UARTx_IER(x=0,1,4,5)							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				RXTO_IE	RXERR_IE	-	RXBF_IE
access	U-0				R/W-0	R/W-0	U-0	R/W-0

bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	NEWUP_IE	-					TXBE_IE	TXSE_IE
access	R/W-0	U-0					R/W-0	R/W-0

Bit	Name	Functional description
31:12	-	**Reserved, read as 0**
11	**RXTO_IE**	Receive timeout interrupt enable, 1 enable. (Only UART0 and UART1 are valid)
10	**RXERR_IE**	Receive error interrupt enable, 1 enable.
9	-	**Reserved, read as 0**
8	**RXBF_IE**	Receive buffer full interrupt enablement, 1 enable
7	-	**Reserved, read as 0**
6:2	-	Transmit buffer empty interrupt enable，1 enable
1	**TXBE_IE**	Transmit buffer empty and Transmit Register empty interrupt enable, 1 enable
0	**TXSE_IE**	**Reserved, read as 0**

20.11.4 UARTx Interrupt flag register (UARTx_ISR)

NAME	UARTx_ISR(x=0,1,4,5)							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-					PERR	FERR	OERR
access	U-0					R/W-0	R/W-0	R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				RXTO	-		RXBF
access	U-0				R/W-0	U-0		R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	NEWKF	-					TXBE	TXSE
access	R/W-0	U-0					R-0	R/W-0

Bit	Name	Functional description
31:19	-	**Reserved, read as 0**
18	**PERR**	Parity error interrupt flag, hardware set, software write 1 to clear
17	**FERR**	Frame format error interrupt flag, hardware set, software write 1 to clear
16	**OERR**	The receiving buffer overflow error interrupt flag is set when the ReceiveBuffer is full and the new data is received. When hardware is set and software writes 1 or reads RXBUF, the receiving overflow interrupt will be cleared, and the original data in the receiving buffer will be overwritten by the new data.
15:12	-	**Reserved, read as 0**

Bit	Name	Functional description
11	**RXTO**	Receive timeout interrupt flag, hardware set, software write 1 to clear(UART0 and UART1 only)
10:9	**-**	**Reserved, read as 0**
8	**RXBF**	Receive Buffer full interrupt flag, hardware set, software writes 1 or reads RXBUF to clear
7	**NEWKF**	UART wake-up interrupt flag, set after rxd falling edge, software write 1 to clear zero; (when using UART wake-up function, need to enable NEGWK_EN first)
6:2	**-**	**Reserved, read as 0**
1	**TXBE**	Transmit Buffer empty interrupt flag, hardware set, software write TXBUF to clear.
0	**TXSE**	shift register empty interrupt flag. Hardware set, software write 1 or software write tx buffer to clear.

20.11.5 UARTx Timeout and delay registers（UARTx_TODR）

Name	UARTx_TODR(x=0,1)							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	TXDLY_LEN							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RXTO_LEN							
access	R/W-1111 1111							

Bit	Name	Functional description
31:16	**-**	**Reserved, read as 0**
15:8	**-**	**Reserved, read as 0**
7:0	**RXTO_LEN**	Receive timeout length, maximum 255baud

20.11.6 UARTx Receive Buffer Register（UARTx_RXBUF）

Name	UARTx_RXBUF(x=0,1,4,5)							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							

access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							RXBUF[8]
access	U-0							R-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RXBUF[7:0]							
access	R-0000 0000							

Bit	Name	Functional description
31:9	-	**Reserved, read as 0**
8:0	**RXBUF**	Receive data buffer

7bits of received data is stored in RXBUF[6:0] for 7-bit send/receive

20.11.7 UARTx Transmit Buffer Register（UARTx_TXBUF）

NAME	UARTx_TXBUF(x=0,1,4,5)							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							TXBUF[8]
access	U-0							R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TXBUF[7:0]							
access	R/W-0000 0000							

Bit	Name	Functional description
31:9	-	**Reserved, read as 0**
8:0	**TXBUF**	Transmit data buffer register

When sending and receiving 7-bits, the 7bits data sent will be written into TXBUF[6:0]

20.11.8 UATRx Baud Rate Generate Register（UARTx_BGR）

NAME	UARTx_BGR(x=0,1,4,5)							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16

name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	SPBRG[15:8]							
access	R/W-0000 0011							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SPBRG[7:0]							
access	R/W-0100 0001							

Bit	Name	Functional description
31:16	**-**	**Reserved, read as 0**
15:0	**SPBRG**	Serial Port Baud Rate Generation

Baud rate calculation is detailed in 18.7.1 Baud rate
Note: When SPBRG <= 0x000F, UARTDIV=16'H000F;
when SPBRG > 0x000F, UARTDIV=SPBRG.

21 Enhanced Universal Asynchronous Receiver/transmitter (UART2)

21.1 Introduction

The UART2 serial communication module features the following:

- Baud rate software configurable1 independent channel (UART2)
- 1 independent channel (UART2)
- Full duplex communication port
- Supports half duplex single line communication
- Supports RX-TX port switching
- UART with data receive complete/receive error interrupt with error type indication
- Configurable data length - supports 6, 7, 8, 9bits
- Configurable stop bits - supports 1 stop bit or 2 stop bits
- Configurable as IR modulated output with configurable carrier frequency and configurable carrier duty cycle
- Support for receive timeout mechanism
- Support for transmit delay function

21.2 UART2 Block Diagram

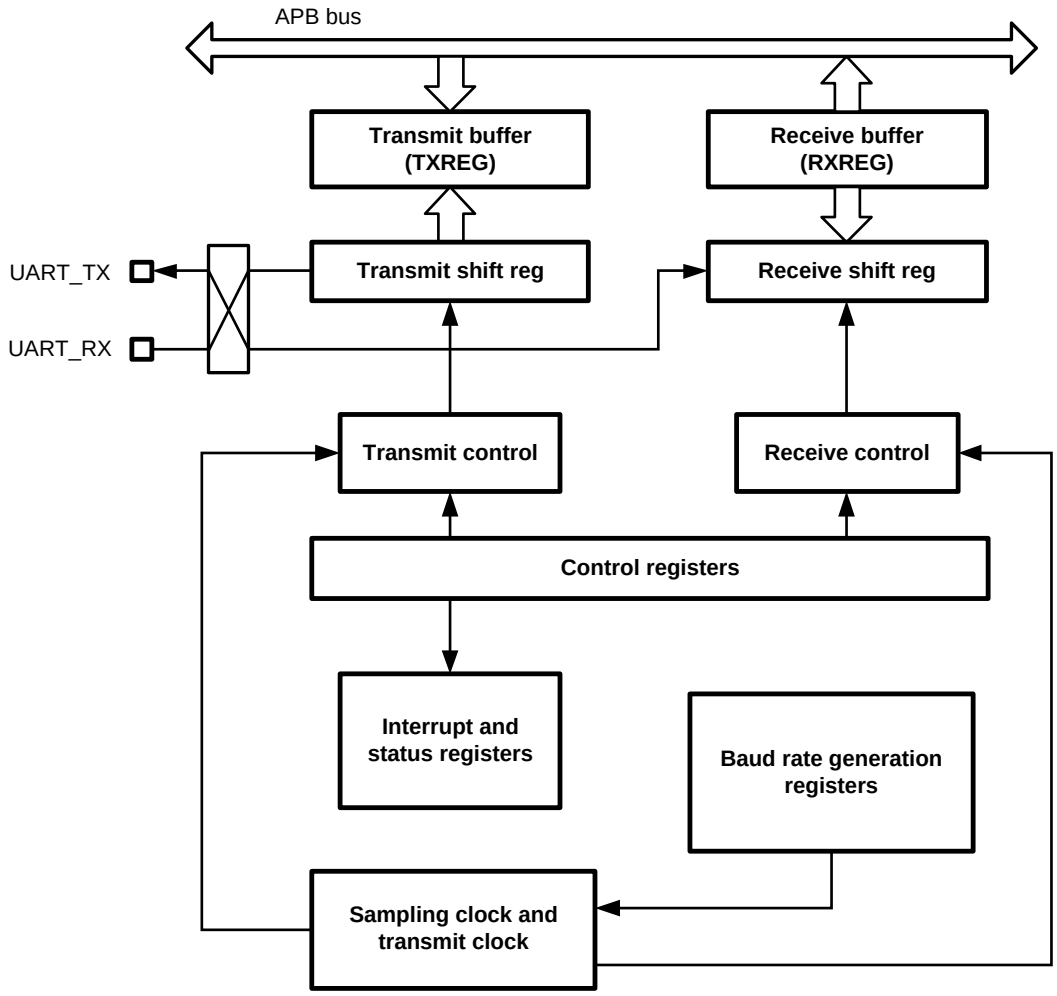


Figure 21-1 UART2 Block Diagram

21.3 Pin Definition

The UART2 module has two pins to communicate with external devices, and each UART2's transceiver signal may be mapped to a different GPIO, which is show as the table below.

Pin			UARTx	Symbol
PD7	PB0		UART2	UART2_RX
PD8	PB1			UART2_TX

Table 21-1 UART2 pin list

21.4 UART Type Distinction

The chip integrates several different types of UARTs (LPUART), the differences of which are shown in the following table.

UART feature	UART0/1	UART2	UART4/5	LPUART0/1
DMA support	Y	Y	Y	Y
Half duplex/Full duplex	Y	Y	Y	Y
Infrared transmitting	Y	Y	Y	-
Dual clock domain (operating clock independent of the bus)	Y	Y	-	Y
Wake-up from sleep	Y	Y	Y	Y
Receive timeout	Y	Y	-	-
Transmit delay	Y	Y	-	-
Data length	6、7、8、9bits			
LIN support	N	Y	N	

Table 21-2 UART type list

21.5 UART Character Format

21.5.1 UART Mode

The basic timing of UART transmission characters is shown in the figure below. Each character contains at least 1bit START and at least 1bit STOP bits, data lengths can be configured to be 6-9bits, and parity bit can be selected.

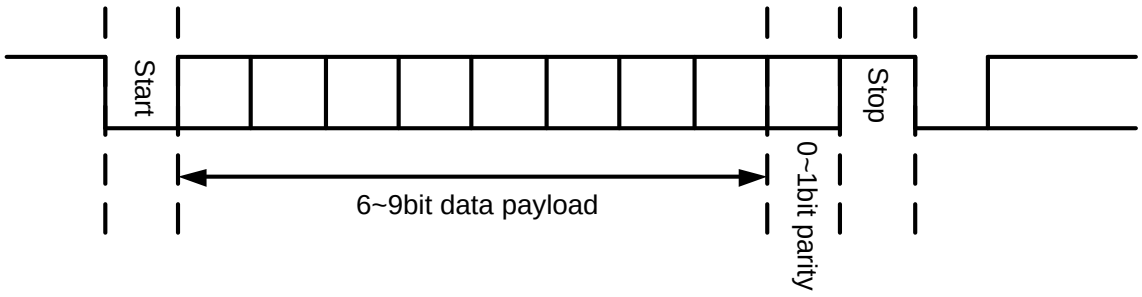


Figure 21-2 UART Character Format

UART supports multiple frame formats controlled by the UARTxCSR.PDsel register and uartxCSR.PARITY register, which are listed in the table below:

PDSEL	PARITY	Frame Format[[1]
00	00	[Start | 7 bits data | Stop]
	01, 10	[Start | 7 bits data | Parity | Stop]
01	00	[Start | 8 bits data | Stop]
	01, 10	[Start | 8 bits data | Parity | Stop]
10	00	[Start | 9 bits data | Stop]
	01, 10	[Start | 9 bits data | Parity | Stop]
11	00	[Start | 6 bits data | Stop]
	01, 10	[Start | 6 bits data | Parity | Stop]

Table 21-3 UART Data Frame Format

[1]：The Stop bit can be either 1bit or 2bits, depending on the STOPCFG register.

Note: *THE PDSEL register is used to configure the data length of the frame. The communication frame length is [start bit + data bit + check bit + stop bit].*

21.6 UART2 Function Description

21.6.1 Clock Structure

UART2 uses a multi-clock structure.

- The bus register clock is represented by PCLK, which is derived from APBCLK and must be enabled when the CPU or DMA needs to access the internal registers of the UART
- The data transceiver clock is represented by UCLK, which can be derived from APBCLK, RCHF, SYSCLK and RCMF, and can work independently of APBCLK. UCLK must be enabled for data transmission and reception.
- LINCLK is used for LIN functions other than data sending and receiving, such as wake-up detection/transmission and interval field detection/transmission, LINCLK works with LSCLK.

The control of PCLK and UCLK is done within the CMU module, and the corresponding CMU control registers must be properly configured before UART2 communication can take place.

The use of a dual clock structure allows the UART2 to operate without being limited by the APBCLK configuration, so that when certain peripherals need to operate at a very high APBCLK frequency, the UART can still operate at a reduced frequency; or conversely, the CPU operating at a lower frequency does not affect the UART's ability to communicate data at a higher baud rate.

Theoretically there is no relative relationship constraint between PCLK and UCLK, UCLK can be faster or slower than PCLK, but the application needs to be careful whether the CPU or DMA has time to carry the data when the difference between the two frequencies is large.

21.6.2 Bit Receiving Sampling

UART oversampled the received data 16 times within one baud, and two out of three majority decision is made at the middle position of each bit to improve noise immunity.

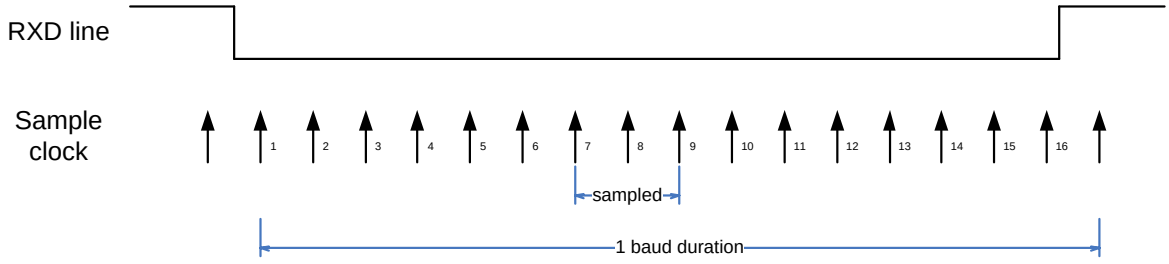


Figure 21-3 3-bit receive 16 times Sampling

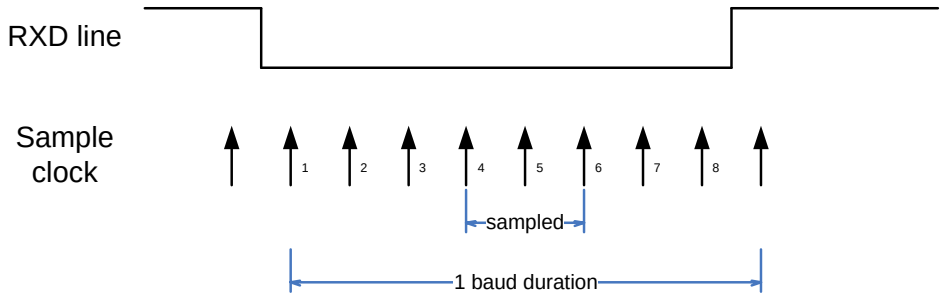


Figure 21-4 4-bit receive 8 times Sampling

The bit received by the receive shift register is the result of a majority verdict. For example, if the result of three samples is 001, the verdict is 0; if it is 011, the verdict is 1.

The OVSM register allows you to configure the oversampling multiplier when the UART receives data. If the UART oversamples the input signal by a factor of 16, it is required that the SPBRG configuration cannot be less than 16, i.e. the UART operating clock must be at least 16 times the baud rate. If the UART oversamples the input signal by a factor of 8, the SPBRG configuration must not be less than 8, i.e. the UART operating clock must be at least 8 times the baud rate.

A higher communication baud rate can be obtained when a smaller oversampling multiple is chosen.

21.6.3 Data Transmission

In transmit mode, the serial data sending circuit of the UART consists mainly of a transmit shift register (TSR), which functions to send the data out bit by bit. The data to be sent must first be written to the transmit buffer. When the software sets the TXEN register, the UART loads the buffer data into the TSR and starts shifting the output if the transmit buffer is not empty.

***Note**: Since the register operation clock and the baud rate clock are asynchronous, there is a maximum delay of 1 baud between the time TXEN is set and the UART starts to send the Start bit, as it needs to wait for the baud rate clock to arrive when the transmission starts.*

In general, the TSR register is empty at the beginning and the data is sent by first setting the baud rate SPBRG, enabling the transmitter module (setting TXEN to 1) and then writing to the TXBUF register to start the transmission. It is also possible to set the baud rate SPBRG, write to the TXBUF register first and then set TXEN to enable the transmitter module to start data transmission. If the transmit module enable bit TXEN is cleared to 0 during data transmission, then data transmission will be interrupted and the transmit module will be reset.

The diagram below shows an example of a UART sending asynchronously. In this example the software first writes data to the TXBUF and then initiates a transmit by setting TXEN.

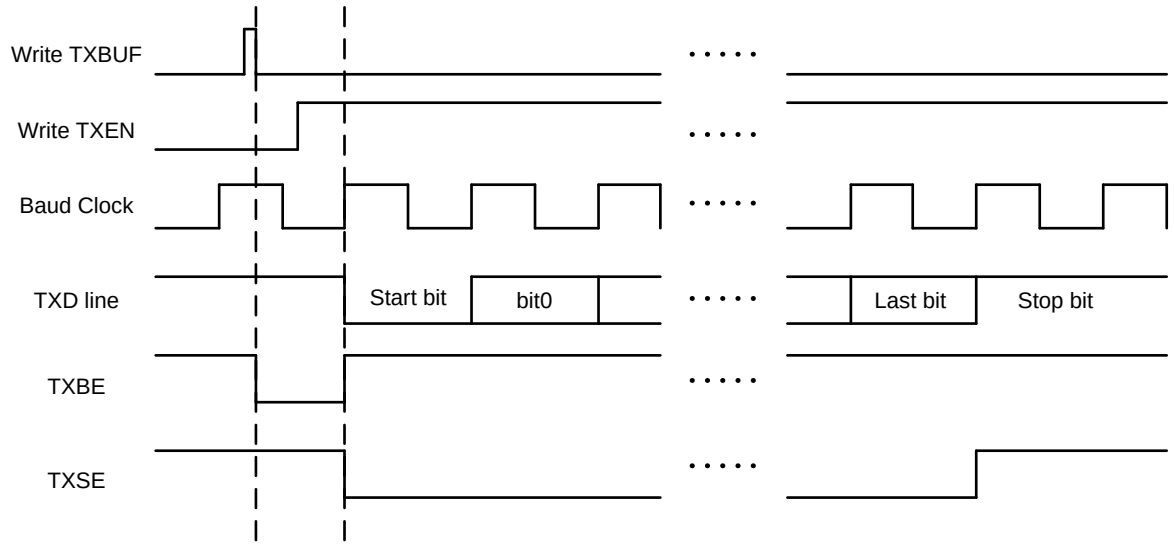


Figure 21-5 UART asynchronous transmit waveform 1

The recommended steps in the figure are as follows:

- Select the appropriate baud rate and initialize the SPBRG
- If interrupt is required, set TXSE_IE or TXBE_IE
- Determine the format of the data to be sent: set the PDSEL register to determine the length of the data to be sent; set the PARITY register to select whether to send a parity bit and the type of parity, and set the STOPSEL register to determine whether to send a 1-bit or 2-bit stop bit
- If the serial data to be sent is infrared modulated, write the appropriate value to the IRCON

register to obtain the appropriate modulation frequency and duty cycle, and set TXIREN

- Write the data to be sent to the TXBUF register (automatic start of transmission)
- Enable the transmitter module: set TXEN

The Software can also set TXEN first and then write TXBUF, UART will immediately start the sending process after the data is written to TXBUF.

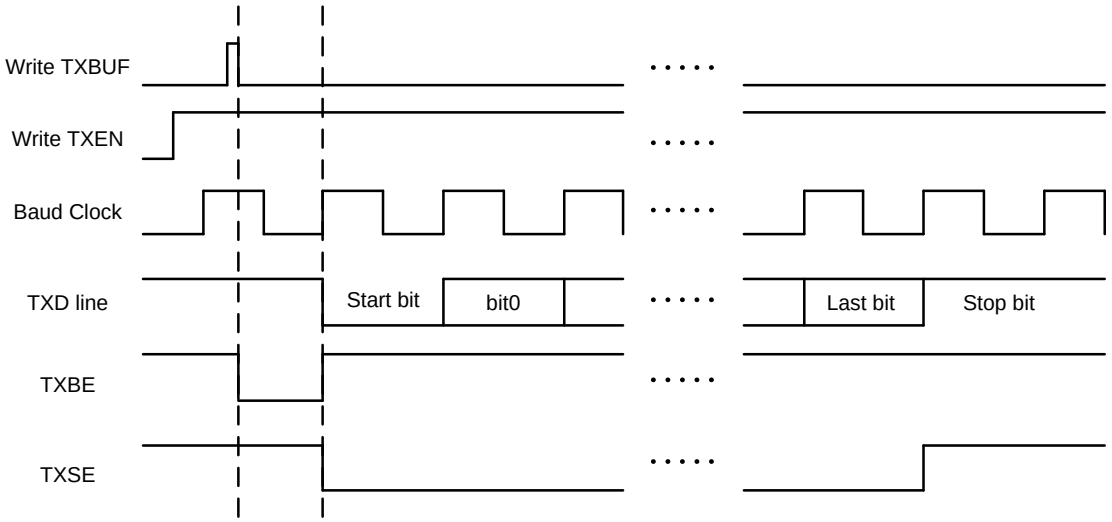


Figure 21-6 UART asynchronous transmission waveform 2

When TXBUF is empty, the software can immediately write the next data to be sent, in order to achieve continuous data transmission without interval.

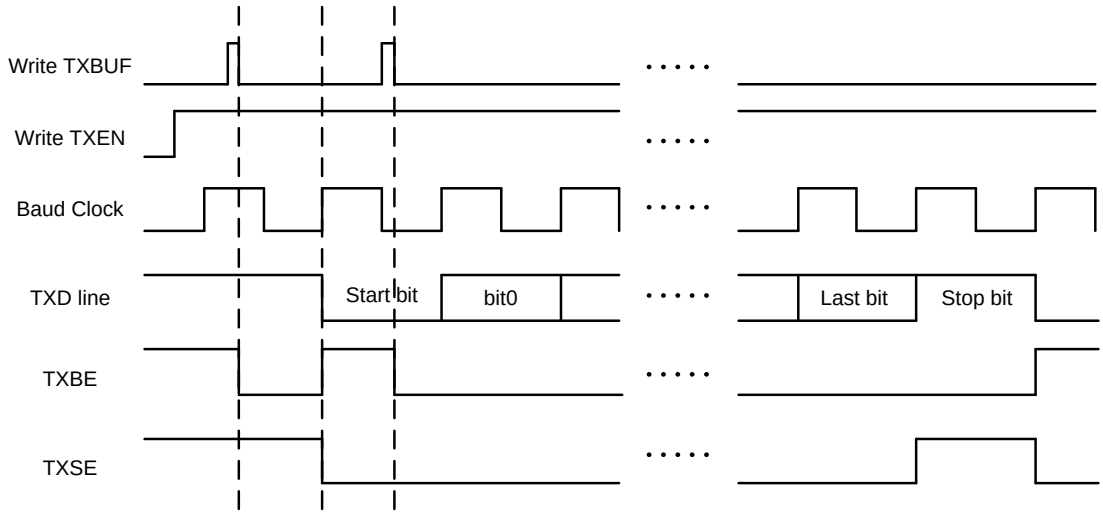


Figure 21-7 UART asynchronous transmission waveform 3

21.6.4 Data Reception

The serial data receiving of UART uses a receive shift register (RSR). When the stop bit is received, RSR feeds the received data into the receive buffer (RXBUF). The interrupt flag RXBF is set to 1 after each received byte is copied into the receive buffer. When new data is received when RXBUF is full, the original data in the RX buffer will be overwritten, and RXBF flag is set again. Meanwhile, receive overflow error occurs, and OERR is set to 1. The OERR flag can be cleared by writing 1by software or reading RXBUF.

During the receiving process, if the correct stop bit is not detected, a frame format error occurs and FERR is set to 1. If a parity error occurs, flag bit PERR is set to 1.

The recommended asynchronous receive procedure is as follows:

- Select the appropriate baud rate and initialize SPBRG
- If an interrupt is required, set RXBF_IE
- Set the format of data receiving: set PDSEL register to determine the length of data sent; Set the PARITY register to choose whether to send a PARITY bit and the type of PARITY, and set the STOPSEL register to decide whether to send a 1-bit or 2-bit stop
- Enable receiving: Set RXEN
- At the end of a frame, the RXBF bit is set to 1. If the RXBF_IE bit is set to 1, an interrupt will be generated
- Read PERR, FERR, and OERR registers to determine if there are any data error or overflow
- Read the received data in the RXBUF register

21.6.5 LOW Power Hibernation Wake-up

The UART2 supports the wake-up from chip hibernation triggered by the falling edge of RXD. When the NEWUP register is set, a falling edge event (low level duration >100ns) on the RXD input will wake up the chip from hibernation mode, with this function, the UART2 can receive data in hibernation mode.

The software configuration method is as follows.

- Configure UART registers to enable NEWUP
- Configure the UART operating clock as RCHF, and configure the baud rate divider register as required

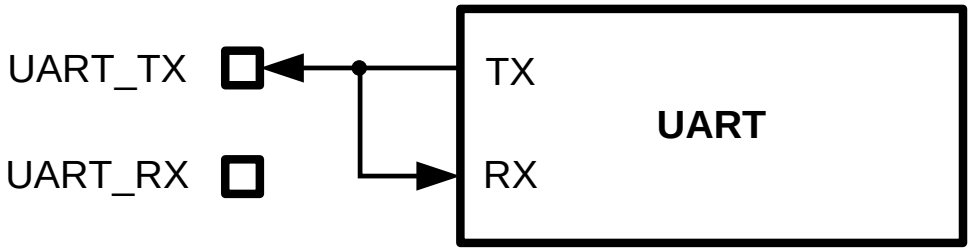
- Configure the corresponding GPIO for UART data reception
- Set RXEN to enable reception
- Set the chip to hibernate and wait for UART receive event

21.6.6 Half duplex single line communication

When the HDSEL register is set, the UART operates in half-duplex mode, when only one signal line is required to complete communication. Half-duplex mode can only be used when the UART is functional, it is forbidden to use it in LIN and smart card mode, i.e. when HDSEL is set, MODE=0 must be ensured.

When HDSEL is set：

- TX and RX signals are connected internally
- The RX pin no longer needs to be used and can be used for GPIO or other functions
- When the UART is not sending data, the TX pin output enable is automatically switched off and the input enable is automatically switched on, the pin remains floating and needs to be given a definite level via a pull-up resistor
- TX pin can be configured as an open-drain output, data conflicts must be handled by software
- UART automatically blocks data reception when data is sent



21.7 Baud Rate Generation

21.7.1 Baud Rate Generation

Baud rate factor is a 16-bit register whose value X is a any integer between 16 and 65535.

Baud rate calculation of UART:

Baud = F_{CLK} / (SPBRG+1);

Note: F_{CLK} can be different clocks in different UART. For UART4 and UART5, FCLK is APBCLK.

For UART0 and UART1, F_{CLK} is a working clock independent of APBCLK.

To support full duplex communication, the receiving and transmitting baud rates are generated separately.

The following table shows the baud rate at common system clock frequencies:

Baud	F_{CLK}=16MHz			F_{CLK}=8MHz		
bps	Actual (bps)	Error%	X+1	Actual (bps)	Error%	X+1
300	300.0019	0.000625	53333	299.9963	-0.00125	26667
1200	1200.03	0.0025	13333	1199.94	-0.005	6667
2400	2399.88	-0.005	6667	2400.24	0.010001	3333
4800	4800.48	0.010001	3333	4799.04	-0.02	1667
9600	9598.08	-0.02	1667	9603.842	0.040016	833
19200	19207.68	0.040016	833	19184.65	-0.07994	417
38400	38369.3	-0.07994	417	38461.54	0.160256	208
57600	57553.96	-0.07994	278	57553.96	-0.07994	139
115200	115107.9	-0.07994	139	115942	0.644122	69
230400	231884.1	0.644122	69	228571.4	-0.79365	35
460800	457142.9	-0.79365	35	470588.2	2.124183	17

Baud	F_{CLK}=24MHz			F_{CLK}=32MHz		
bps	Actual (bps)	Error%	X+1	Actual (bps)	Error%	X+1
300	300	0	80000	299.9991	-0.00031	106667
1200	1200	0	20000	1199.985	-0.00125	26667
2400	2400	0	10000	2400.06	0.0025	13333
4800	4800	0	5000	4799.76	-0.005	6667
9600	9600	0	2500	9600.96	0.010001	3333
19200	19200	0	1250	19196.16	-0.02	1667

Baud	F_{CLK}=24MHz			F_{CLK}=32MHz		
bps	Actual (bps)	Error%	X+1	Actual (bps)	Error%	X+1
38400	38400	0	625	38415.37	0.040016	833
57600	57553.96	-0.07994	417	57553.96	-0.07994	556
115200	115384.6	0.160256	208	115107.9	-0.07994	278
230400	230769.2	0.160256	104	230215.8	-0.07994	139
460800	461538.5	0.160256	52	463768.1	0.644122	69

Table 21-4 Common clock frequency baud rate calculation

21.7.2 Adaptive Baud Rate Generation

It can be realized adaptive baud function by using the Capture function of Timer. There is a one way to achieve. The external UART device sends a frame with the agreed data content (e.g. 0xF8) and the Timer counts the high level pulse width of the frame. The MCU reads the Timer capture result to calculate the baud rate factor and writes it to the baud rate generation register to be used as the clock division count value X for the baud rate generation. At this point the receive state is reset and the start bit is waited for again to receive data at the baud rate generated by the written baud rate factor. Refer to the Timer chapter.

21.8 Infrared modulation

TZBRG register holds an 11-bit frequency division X, whose value is any integer between 0 and 2047. All UART share one infrared modulation generator.

Infrared modulation frequency calculation formula:

FIR = F_{APBCLK}/ (TZBRG+ 1)

The infrared modulation method is: when sending data 0, the infrared frequency is modulated; when sending data 1, the infrared frequency is not modulated. In order to meet the needs of PNP and NPN infrared optical transistor, register IRFLAG bit controls the polarity of infrared modulation output.

The TH register is used to configure the ir modulation duty cycle.

Duty ratio：Y = (TZBRG[10:4] * TH) /(TZBRG+1);

When TH=4’b0000，the duty ratio is Y = (TZBRG[10:1]+1)/(X+1)；

When TZBRG[10:4]=7 'h00, the duty ratio is Y = TH/(TZBRG[3:0]+ 1);

If this time TH>TZBRG [3:0], then ir modulation clock IRCLK is fixed at high level.

When the infrared modulation polarity is reversed (IRFLAG=1), the duty cycle is also 1-Y

The infrared modulation waveform is shown in the following figure:

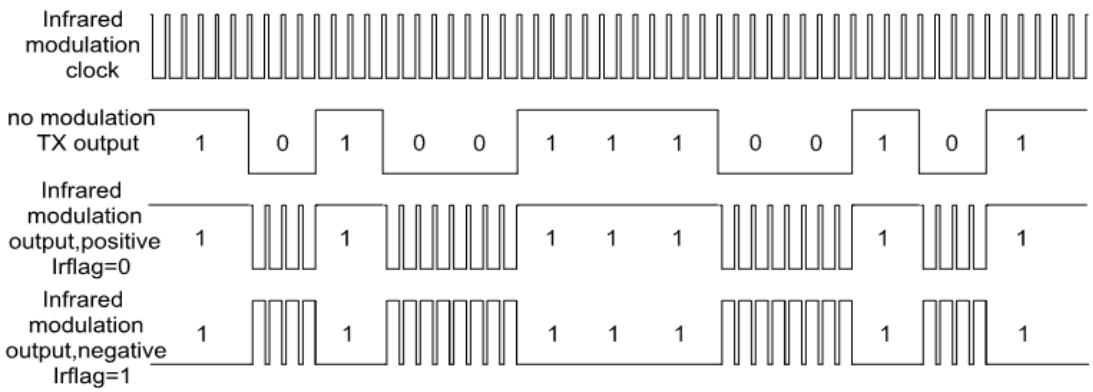


Figure 21-8 The infrared modulation waveform

Duty cycle is defined as the high-level length/period regardless of whether the effective level is 0 or 1.

21.9 Receive Timeout

A time-out mechanism is designed for time-sensitive applications such as MODBUS. When the RXTOEN register is enabled, the timeout counter is counted at the baud rate clock. Each time a complete data frame is received, the timeout counter is cleared and the count is restarted. The upper limit of the timeout overflow can be configured by the software with a maximum of 255 baud.

21.10 Transmit Delay

Through the TXDLY_LEN register, you can control the time interval between two data frames sent, the unit is Baud. The transmit delay is the interval between the end of the last STOP bit of the previous frame and the start bit of the next frame.

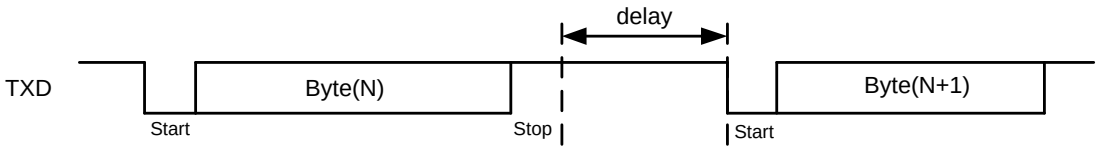


Figure 21-9 UART transmit delay

21.11 LIN Bus Communication Function

21.11.1 Introduction

LIN (Local Interconnect Network) is a low-speed, low-cost serial communication protocol with a communication baud rate of 1 to 20 Kbps, supporting the interconnection of a single master and up to 15 slaves. Each node is connected to the bus via a single wire transceiver, which is connected to the controller via a UART-like interface.

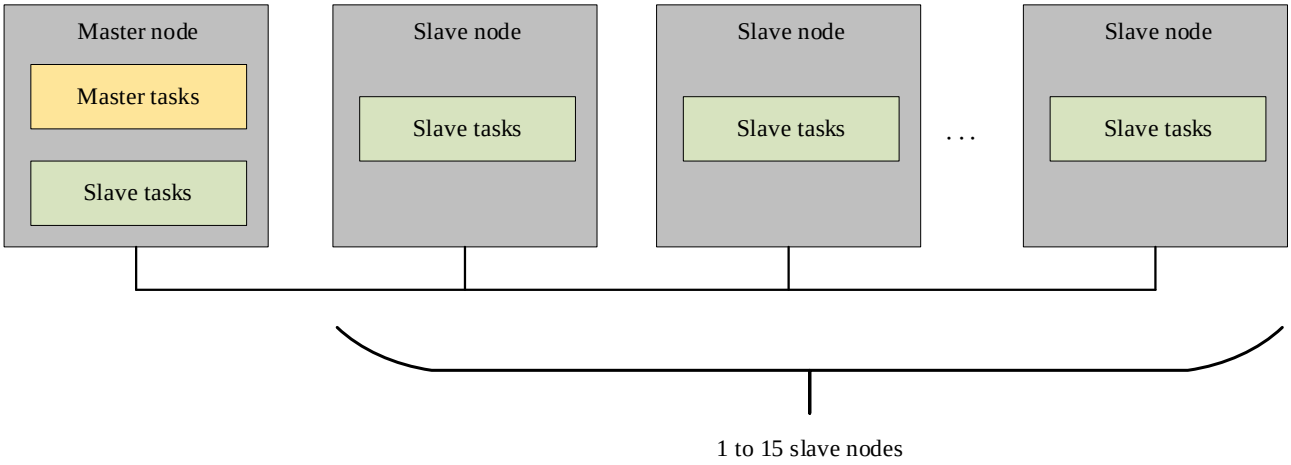


Figure 21-10 LIN bus topology

The frame structure of the LIN bus consists of a header and an answer. The header is always sent by the master, which can continue to send data after sending the header, or receive the answer data from the slave.

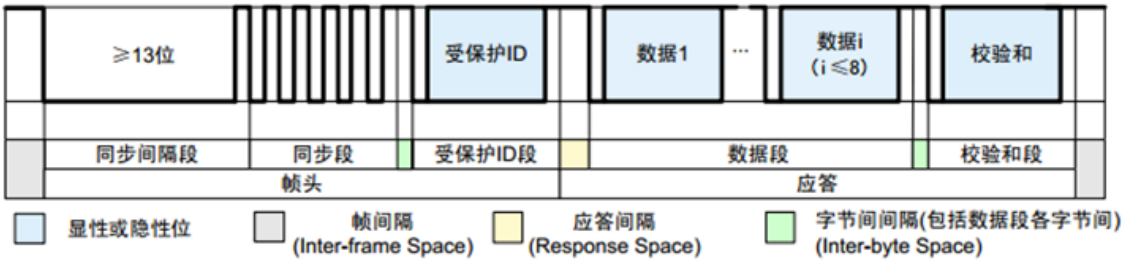


Figure 21-11 LIN bus frame structure figure fix

Except for the synchronous interval segments, all segments are transmitted in the standard UART data transmission format. In LIN frames, the LSB (Least Significant Bit) is sent first and the MSB (Most Significant Bit) is sent last.

21.11.1.1 Break Field

The Break Field consists of a low level of at least 13 bits and interval characters. The bus baud rate is always determined by the host and the start of the frame is considered by the slave when it detects an explicit level on the bus lasting at least 11 bits.

Figure 21-12 LIN break field figure fix

21.11.1.2 Sync Field

The synchronization segment is used to synchronize the slave's baud rate clock, thus reducing the bus requirements for the slave's clock accuracy; the UART-LIN module supports baud rate adaptation based on the synchronization segment when used as a LIN slave.

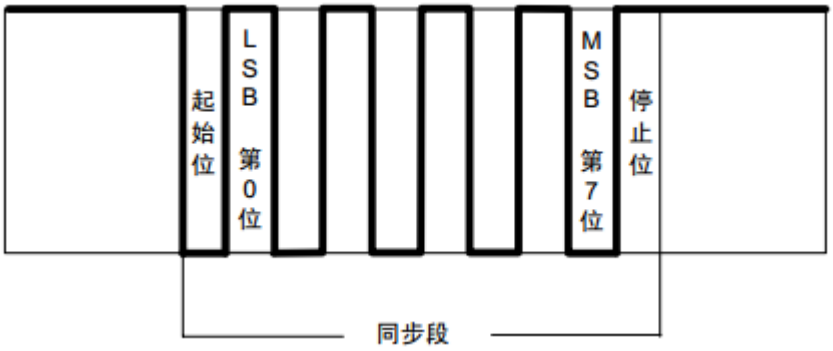


Figure 21-13 LIN sync field figure fix

LIN synchronization is determined by the falling edge and uses byte 0x55 (converted to binary as 01010101b).

$$1\text{位时间} = \frac{\text{第7位的下降沿时刻} - \text{起始位的下降沿时刻}}{8}$$

After the synchronous segment, the slave software completes the automatic adjustment of the communication baud rate according to the count value.

21.11.1.3 PID Field and Data Field

The first 6 bits of the PID segment are called the Frame ID, plus two parity bits called Protected ID. The frame IDs range from 0x00 to 0x3F and there are 64 of them. The Frame ID identifies the class and destination of the frame. The response of the slave task to the frame header (receive/send/ignore the answer part) is based on the frame ID. If the frame ID is transmitted incorrectly, the signal will not reach its destination correctly, hence the introduction of the parity bit.

$$P0 = ID0 \oplus ID1 \oplus ID2 \oplus ID4$$
$$P1 = \neg (ID1 \oplus ID3 \oplus ID4 \oplus ID5)$$

The formula shows that the PID does not have all zeros or all ones, so if the slave node receives "0xFF" or "0x00", it can be judged as a transmission error.

帧的类型		帧 ID
信号携带帧	无条件帧	0x00 ～ 0x3B
	事件触发帧	
	偶发帧	
诊断帧	主机请求帧	0x3C
	从机应答帧(注 1)	0x3D
保留帧		0x3E，0x3F

注：1. 从机应答帧是一个完整的帧，与帧结构中的"应答"(帧的一部分)不同，注意区别。

Figure 21-14 LIN frame ID

The data sent by the node is located in the data segment and consists of 1 to 8 bytes, with the lowest numbered byte being sent first and the numbering increasing.

The protocol does not specify which part of the frame displays the information about the data length code; the content and length of the data are agreed in advance by the system designer according to the frame ID.

In general, for an answer in a frame, only one transmitting node exists on the bus, otherwise an error occurs. The exception is event-triggered frames, where zero, one or more transmit nodes may exist.

PIDs and data segments are sent and received in the same way as on a standard UART and do not require special handling by the LIN module.

21.11.1.4 Checksum Field

The checksum segment checks the content transmitted in the frame and contains the following

- Classic checksum (LIN 1.x): all data segments
- Enhanced checksum (LIN 2.x): all data segments and PID segments

The checksum is a bitwise inverse sum of the bytes of the checksum object, which is then used as the checksum to be sent. The receiver adds the same rounded binary addition to the received data according to the checksum type, without inverting the final sum, and adds the sum to the received checksum.

The checksum function is implemented by software.

21.11.2 LIN Master Operation

In master mode, the LIN module can send wake-up signals, interval segments, synchronization segments and data segments to the bus.

21.11.2.1 Wake-up send

The wake-up signal is sent by software setting the TX_WKUP register, the length of the signal is the number of clock cycles defined by the WKUP_LEN register, the default is 9 LINCLK clocks, which is about 270us. 256 clock cycles are supported at maximum, which is about 7.6ms. register

is automatically cleared to zero and the LINWKTF send completion flag is set.

21.11.2.2 Break and Sync Send

Break field sending is fixed at 13 bits and is sent by software setting the TX_BREAK register.

The synchronous segment is realized by software sending 0x55, and the data segment and check segment are sent the same as the standard UART data sending.

After sending Break and PID, the host can also actively switch to receive, and the data reception is the same as the standard UART reception.

21.11.3 Slave Operation

In slave mode, LIN needs to support wake-up signal and interval segment reception, as well as baud rate adaptation based on synchronous segments. Wake-up signals and Break detection circuits are automatically enabled when LIN slave is enabled.

21.11.3.1 Wake-up Detection

When the LIN bus is in a sleep state, both the master/slave nodes can send a wakeup signal to the bus, which lasts between 250us and 5ms. The rest of the nodes (other than those sending the wakeup signal) determine the wakeup signal with a threshold of greater than 150us. Each slave node must be ready to receive a command (frame header) from the host within 100ms from the end of the explicit pulse of the wakeup signal; the host node must also be woken up and within 100ms the host node sends a frame header to start communication.

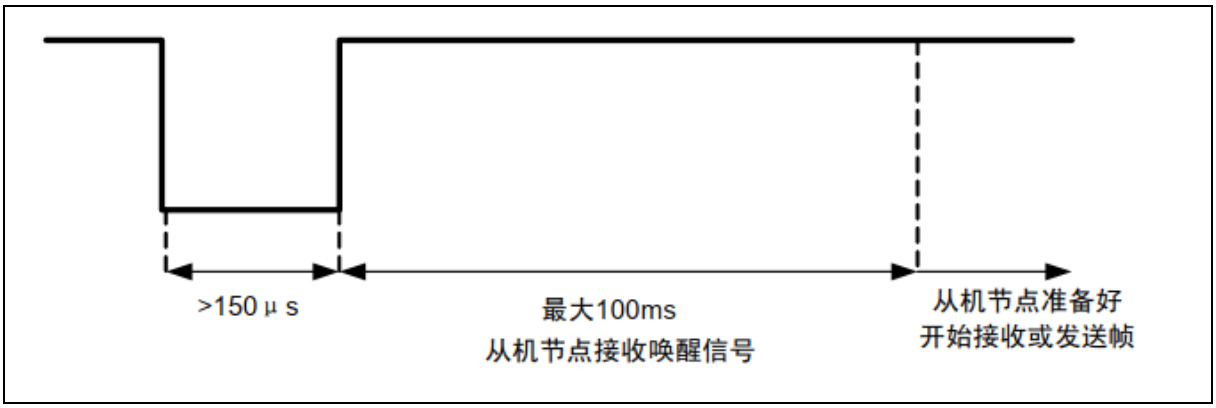


Figure 21-15 LIN bus wake-up signal

Wake-up signal recognition is achieved by a low speed clock driven counter. When the software sets the WKDET_EN register, the counter starts to monitor the RXD input, the counter remains reset when RXD is high and starts counting when RXD goes low. When RXD remains low for the number of clocks defined by WKUP_LEN, the wake-up signal is considered valid and a wake-up interrupt is generated.

The wake-up counter works using LINCLK, i.e. LSCL

K, which is independent of APBCLK, so this part of the function can be used in sleep mode. the LSCLK clock period is about 30us, and the software can set the wake-up counter threshold through the WKUP_LEN register to meet a minimum of 150us. When RXD remains low for longer than the

WKUP_LEN setting, the LINWKF interrupt flag register is set, indicating that the LIN module recognizes a valid wake-up signal on the bus.

21.11.3.2 Break Detection

When the slave is in the wake-up state (LIN_SLEEP=0), the Break segment is first received. The slave samples RXD according to the set baud rate. When reception exceeds 11 bits low in succession, the BF is detected, the Break interrupt flag is set and the LIN slave state machine enters the baud rate synchronization state.

21.11.3.3 Baud Rate Synchronization

In the baud rate synchronous state, the software enables the baud rate adaptive circuit, the baud rate detection module measures the width of the synchronous segment, the software calculates the baud rate, and the software updates the baud rate register at the end of the synchronous segment. During the reception of the synchronous segment the receiver circuit does not work and does not write any data to the RX DATA register.

***Note**: A software write to the baud rate register will cause the baud rate divider counter to clear and restart counting at the new baud rate.*

***Note**: A Break detected while data byte reception is not complete is considered a frame format error.*

21.11.4 LIN baud rate synchronization

Since the LIN baud rate is at a minimum of 1Kbps, and considering that a longer counter is required to calculate the 8bit total length with a higher system master frequency, the baud rate synchronization counter bit width is extended to 20bit, which is sufficient to accommodate the cumulative length of the 8 longest baud bits with a 16bit baud rate counter.

21.12 Register

Base address of UART2：0x4001 3000

Offset	Name	Symbol
UARTIR register(module base address：0x40019C00)		
0x00	Infrared modulation Control Register	UART_IRCR
UART2 register(module base address：0x40013000)		
0x00	UART2 Control Status Register	UART2_CSR
0x04	UART2 Interrupt Enable Register	UART2_IER
0x08	UART2 Interrupt Status Register	UART2_ISR
0x10	UART2 Receive Buffer	UART2_RXBUF
0x14	UART2Transmit Buffer	UART2_TXBUF
0x18	UART2 Baud rate Generator Register	UART2_BGR
0x20	UART2 Mode Control Register	UART2_MCR

21.12.1 Infrared Modulation Register (UART_IRCR)

NAME	UART_IRCR							
offset	0x00							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	IRFLAG	TH				TZBRG[10:8]		
access	R/W-0	R/W-0000				R/W-000		
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TZBRG[7:0]							
access	R/W-11010010							

Bit	Name	Functional description
31:16	**-**	**Reserved, read as 0**
15	**IRFLAG**	Controls the default output polarity when infrared modulation sends data. 0：Positive polarity 1：Negative polarity
14:11	**TH**	Transmission High Duty
10:0	**TZBRG**	Transmission Baud Rate

21.12.2 UART2 Control status register (UART2_CSR)

NAME	UART2_CSR							
offset	0x00							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							BUSY
access	U-0							R-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-						TXIREN	RXTOEN
access	U-0						R/W-0	R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-		OVSM	IOSWAP	NEWUP	-	BITORD	STOPCFG
access	U-0		R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	PDSEL		PARITY		RXPOL	TXPOL	RXEN	TXEN
access	R/W-00		R/W-00		R/W-0	R/W-0	R/W-0	R/W-0

Bit	Names	Functional description
31:25	-	**Reserved, read as 0**
24	**BUSY**	UART communication flag, read-only 1：UART is communicating. 0：UART is idling
23:18	-	**Reserved, read as 0**
17	**TXIREN**	Transmit Infra-red modulation Enable 1：Enable infrared modulation transmission. 0：Turn off infrared modulation sending.
16	**RXTOEN**	Receive Time-Out Enable 1：Enable the receive timeout function. 0：Turn off receive timeout.
15:14	-	**Reserved, read as 0**
13	**OVSM**	Oversampling mode 0：16x oversampling 1：8x oversampling
12	**IOSWAP**	Exchange of RX and TX pins. 0：The default pin order (Consistent with package diagram.) 1：swap pin.
11	**NEWUP**	Neg edge Wakeup enable (Valid for UART0 and UART1 only) 1: Enable RX falling edge wake-up 0: disable wake-up on RX falling edge
10	-	**Reserved, read as 0**

Bit	Names	Functional description
9	**BITORD**	Bit order when data is sent/received 0：LSB first 1：MSB first
8	**STOPCFG**	Stop bit width configuration, valid only for transmit frame format, no stop bits are judged on reception 0: 1-bit stop bit 1: 2-bit stop bit
7:6	**PDSEL**	Data length selection for each frame; this register is valid for both data transmission and reception 00：7 data bit 01：8 data bit 10：9 data bit 11：6 data bit
5:4	**PARITY**	parity bit configuration; This register is valid for both sending and receiving data. 00：No parity bit 01：Even 10：Odd 11：RFU
3	**RXPOL**	Receive data polarity configuration. 0：standard 1：inverted
2	**TXPOL**	Transmit data polarity configuration. 0：standard 1：inverted
1	**RXEN**	1 enable, receive enable
0	**TXEN**	1 enable, transmit enable

21.12.3 UART2 Interrupt enable register (UART_IER)

NAME	UART2_IER							
offset	0x04							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-					LINSYNC_IE	LINB_IE	LINWK_IE
access	U-0					R/W-0	R/W-0	R/W-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

name	-				RXTO_IE	RXERR_IE	-	RXBF_IE
access	U-0				R/W-0	R/W-0	U-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	NEWUP_IE	-					TXBE_IE	TXSE_IE
access	R/W-0	U-0					R/W-0	R/W-0

Bit	Name	Functional description
31:27	-	**Reserved, read as 0**
26	**LINSYNC_IE**	LIN Synchronous segment interrupt enable, 1 enable
25	**LINB_IE**	LIN Break segment interrupt enable, 1 enable
24	**LINWK_IE**	LIN Wake-up interrupt enable, 1 enable
23:17	-	**Reserved, read as 0**
16	-	**Reserved, read as 0**
15:12	-	**Reserved, read as 0**
11	**RXTO_IE**	Receive Time-Out Interrupt Enable, 1 enable
10	**RXERR_IE**	Receive Error Interrupt Enable, 1 enable
9	-	**Reserved, read as 0**
8	**RXBF_IE**	Receive Buffer Full Interrupt Enable, 1 enable
7	**NEWUP_IE**	Neg edge Wakeup Interrupt Enable, 1 enable
6:2	-	**Reserved, read as 0**
1	**TXBE_IE**	Transmit Buffer Empty Interrupt Enable, 1 enable
0	**TXSE_IE**	Transmit Shift register Empty Interrupt Enable, 1 enable

21.12.4 UART2 Interrupt flag register (UART2_ISR)

NAME	UART2_ISR							
offset	0x08							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-					LINSYNCF	LINBF	LINWKF
access	U-0					R/W-0	R/W-0	R/W-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	LINWKTF	-				PERR	FERR	OERR
access	R/W-0	U-0				R/W-0	R/W-0	R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				RXTO	-		RXBF
access	U-0				R/W-0	U-0		R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

name	NEWKF	-	TX_OERR	TXBE	TXSE
access	R/W-0	U-0	R/W-0	R -0	R/W-0

Bit	Name	Functional description
31:27	-	**Reserved, read as 0**
26	**LINSYNCF**	LIN synchronous segment receive completion interrupt flag, hardware set, software write 1 clear.
25	**LINBF**	LIN Slave mode: LIN Break segment receive completion interrupt flag, hardware set, software write 1 to clear LIN master mode: LIN Break segment send completion interrupt flag, hardware set, software write 1 clear
24	**LINWKDF**	LIN wake-up signal detection flag, set in hardware when a valid wake-up signal is recognized, cleared by software write 1
23	**LINWKTF**	LIN Wake-up signal send completion interrupt flag, hardware set, software write 1 to clear
22:19	-	**Reserved, read as 0**
18	**PERR**	Parity error interrupt flag, hardware set, software write 1 to clear
17	**FERR**	Frame format error interrupt flag, set in hardware, cleared by software write 1
16	**OERR**	Receive buffer overflow error interrupt flag, set when the receive buffer is full and new data is received; set in hardware and cleared by software write 1 or when RXBUF is read, On receive overflow, the original data in the receive buffer is overwritten by the new data. (RX buffer Overflow Error flag, write 1 to clear)
15:12	-	**Reserved, read as 0**
11	**RXTO**	Receive timeout interrupt flag, hardware set, software write 1 to clear (Receive Time-Out flag, write 1 to clear)
10:9	-	**Reserved, read as 0**
8	**RXBF**	Receive buffer full interrupt flag, hardware set, software write 1 or clear when RXBUF is read (Receive Buffer Full flag write 1 to clear)
7	**NEWKF**	RX neg edge asynchronous detect interrupt flag, hardware set, software write 1 to clear (Neg edge Wakeup Flag write 1 to clear)
6:3	-	**Reserved, read as 0**
2	**TX_OERR**	Transmit buffer air-break flag, set in hardware and cleared by software when TXBUF is written (Transmit Buffer Empty flag)

Bit	Name	Functional description
1	**TXBE**	Send buffer empty and shift register send complete interrupt flag, set in hardware, cleared by software write 1 or software write to send buffer (Transmit Shift register Empty flag, write 1 to clear)
0	**TXSE**	Parity error interrupt flag, hardware set, software write 1 to clear (Parity Error, write 1 to clear)

21.12.5 UART2 Timeout and delay register (UART2_TODR)

NAME	UART2_TODR							
offset	0x0C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	TXDLY_LEN							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RXTO_LEN							
access	R/W-1111 1111							

Bit	Name	Functional description
31:16	**-**	**Reserved, read as 0**
15:8	**TXDLY_LEN**	Transmit Delay Length. Max. 255baud
7:0	**RXTO_LEN**	(Receive Time-Out Length. Max. 255baud

21.12.6 UART2 Receive Buffer Register (UART2_RXBUF)

Name	UART2_RXBUF							
offset	0x10							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

name	-							RXBUF[8]
access	U-0							R-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RXBUF[7:0]							
access	R-0000 0000							

Bit	Name	Functional description
31:9	-	**Reserved, read as 0**
8:0	**RXBUF**	Receive data buffer register

When 7 bits are sent and received, the 7bits of data received is written to RXBUF[6:0]

21.12.7 UART2 Transmit Buffer Register (UART2_TXBUF)

Name	UART2_TXBUF							
offset	0x14							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							TXBUF[8]
access	U-0							W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TXBUF[7:0]							
access	W-0000 0000							

Bit	Name	Functional description
31:9	-	**Reserved, read as 0**
8:0	**TXBUF**	Transmit data buffer register

When 7 bits are sent and received, the 7bits of data sent is written to TXBUF[6:0]

21.12.8 UART2 Baud Rate Generate Register (UART2_BGR)

Name	UART2_BGR							
offset	0x18							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	SPBRG[15:8]							
access	R/W-00000011							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SPBRG[7:0]							
access	R/W-01000001							

Bit	Name	Functional description
31:16	-	**Reserved, read as 0**
15:0	**SPBRG**	Serial Port Baud Rate Generation Software writes to SPBRG will cause the baud rate counter to clear and restart counting; software should avoid overwriting SPBRG during normal communication

Baud rate calculation is detailed in Baud rate generation

Note: When SPBRG <= 0x000F, UARTDIV=16'H000F;

when SPBRG > 0x000F, UARTDIV=SPBRG.

In smart card mode, Baud ＝ F_{7816}/(SPBRG ＋ 1), F_{7816} is the smart card output operating clock frequency, configured by the CODR register

The minimum available value for SPBRG in smart card mode is 0x1, and configuration of 0x0 is disabled

21.12.9 UART2 Model Register (UART2_MCR)

Name	UART2_MCR							
offset	0x20							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					HDSEL	LINMD	MODE
access	U-0					R/W-0	R/W-0	R/W-0

Bit	Name	Functional description
31:3	-	**Reserved, read as 0**
2	**HDSEL**	UART half duplex single line mode selection 0: Dual-wire UART 1: Single-wire half-duplex UART **Note**: this register is only valid when MODE=0
1	**LINMD**	LIN mode selection register 0: LIN master 1: LIN slave
0	**MODE**	Module control registers 0: Standard UART mode 1: LIN mode

21.12.10 UART2 LIN Control Register (UART2_LINCR)

Name	UART2_LINCR							
offset	0x24							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

name	WKUP_LEN							
access	R/W-0000 1000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					WKDET_EN	TX_BF	TX_WKUP
access	U-0					R/W-0	R/W-0	R/W-0

Bit	Name	Functional description
31:16	-	**Reserved, read as 0**
15:8	**WKUP_LEN**	When sending a wake-up signal, this register defines the LIN bus wake-up signal length When receiving a wake-up signal, this register defines the LIN wake-up signal recognition threshold The unit is LINCLK, i.e. LSCLK; a reset value of 8 indicates 9 clock cycles, approximately 270us; a maximum of 256 clock cycles, approximately 7.6ms
7:3	-	**Reserved, read as 0**
2	**WKDET_EN**	LIN wake-up signal detection enable 0: Wake-up signal detection disabled 1: Enable detection of wake-up signal
1	**TX_BF**	Sends the LIN bus Break signal, which is set by software and automatically cleared by hardware when the transmission is complete. Only valid in LIN host mode
0	**TX_WKUP**	The LIN bus wake-up signal is sent, set by the software, and automatically cleared by the hardware when the transmission is complete. Wake-up can be sent in both LIN master and slave modes *Note: TX_BF and TX_WKUP cannot be set at the same time; if the software sets both TX_BF and TX_WKUP, the hardware will only send a wakeup signal, not a Break*

21.12.11 UART2 LIN Baud Rate Synchronized Register (UART_LINBSR)

Name	UART2_LINBSR							
offset	0x28							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-				BAUD_SYNC			
access	U-0				R-0000			
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	BAUD_SYNC							
access	R-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	BAUD_SYNC							
access	R-0000 0000							

Bit	Name	Functional description
31:20	-	**Reserved, read as 0**
19:0	**BAUD_SYNC**	Synchronous segment counter, which counts the number of clocks in the synchronous segment and is used for baud rate synchronization. The software reads this register after detecting the SYNCF flag and divides by 8 to obtain the current LIN baud rate

22 Low Power UART (LPUART)

22.1 Introduction

LPUART is an enhanced asynchronous serial communication interface. Its working clock can choose bus clock (APBCLK), 32768Hz crystal oscillator clock (XTLF), 32KHz low-power ring oscillator clock (LPOSC), high-frequency ring oscillator clock (RCHF), system clock (SYSCLK), low-power intermediate frequency ring oscillator clock (RCMF). Among them, when the working clock is selected as XTLF or LPOSC, it can support data reception up to 9600 baud rate. At this time, LPUART has extremely low power consumption and can work in Sleep/DeepSleep mode.

Features:

- Asynchronous data transmission and reception
- 2 independent channels (LPUART0, LPUART1)
- Standard UART frame format
  - start bit (1bit)
  - Programmable data word length (6 or 7 or 8 or 9 bits)
  - Programmable parity(Odd or Even or None)
  - Configurable stop bits (1 or 2 stop bits)
- Data polarity control
- Communication under Sleep/DeepSleep mode
- Interrupt flag
  - Receive Buffer full
  - Receive Buffer overflow
  - Receive frame format error
  - Receive parity error
  - START detection
  - Data matching
  - Transmit complete
- Wake up the chip in sleep mode
  - RXD falling edge wake up
  - Start bit detection wakes up
  - 1 byte receive complete wake up
  - 1 byte data match wake up
- Support DMA (not available under Sleep/DeepSleep mode)

22.2 Block Diagram

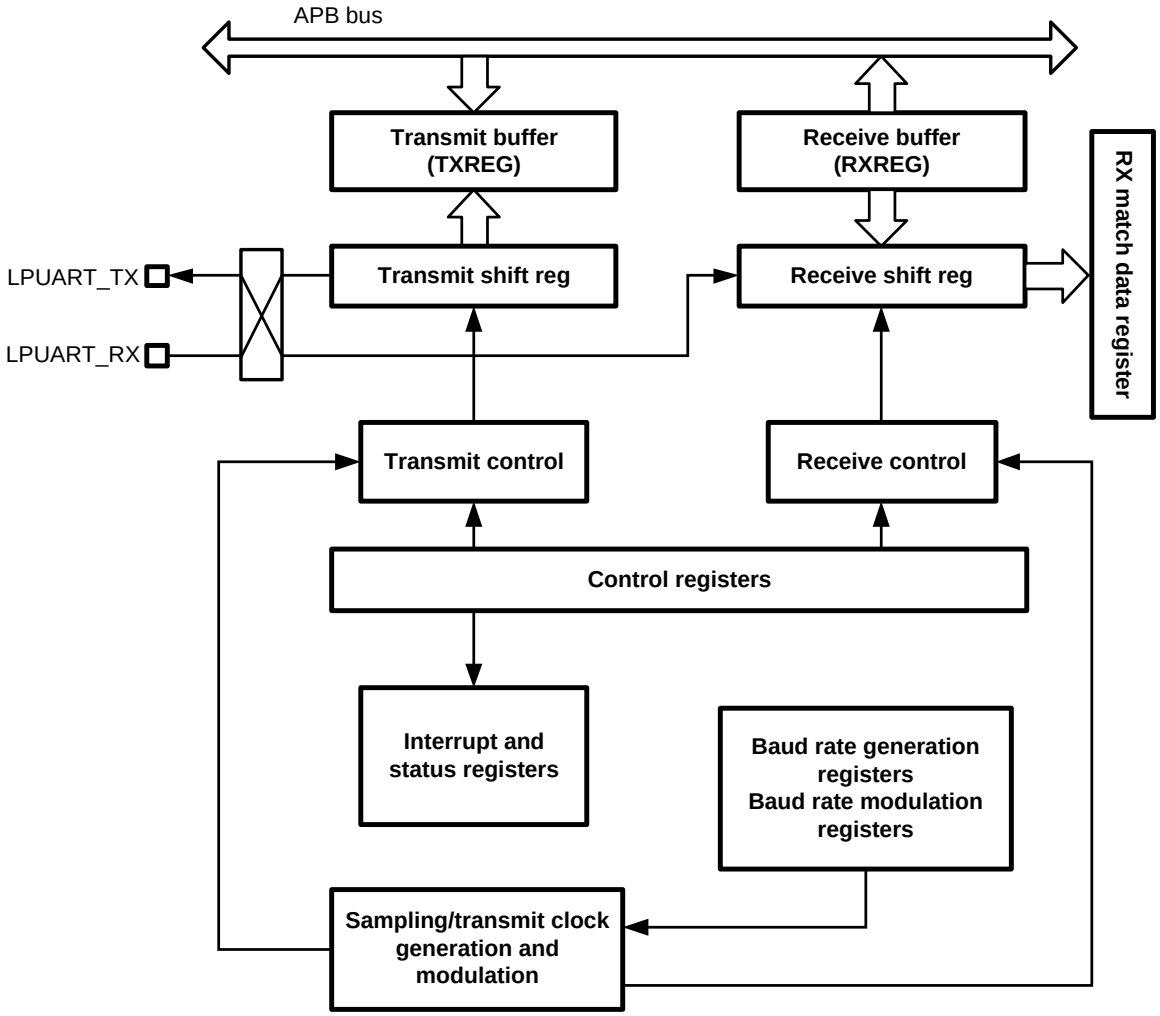


Figure 22-1 LPUART Block Diagram

22.3 Pin Definition

The LPUART module uses 2 pins to communicate with external devices, and the receiving and sending signals of each UART may be mapped to different GPIOs.

Pin		UARTx	Symbol	Function
PA13	PA2	LPUART0	LPUART0_RX	Data Receive
PA14	PA3		LPUART0_TX	Data Transmit
PC2	PB13[1]	LPUART1	LPUART1_RX	Data Receive
PC3	PB14[1]		LPUART1_TX	Data Transmit

[1] Only available for FM33LG0x6 models

When the LPUART function is mapped to multiple pins at the same time:

- PA2 and PA13 are configured as digital peripheral functions at the same time
  - Only the RX signal on PA2 will be input into the module
- PC2 and PB13 are configured as digital peripheral functions at the same time
  - Only the RX signal on PC2 will be input into the module
- When the LPUART transmission function is mapped to multiple GPIOs at the same time, these pins will send data at the same time

22.4 Clocks

LPUART uses a clock independent of APBCLK for data transmission and reception, and the relevant registers need to be configured in the CMU module before work.

LPUART can use XTLF or LPOSC to work. Because LPOSC is not high in accuracy, clock calibration must be performed before using LPOSC for LPUART communication, and LPOSC is calibrated to within +/-1%.

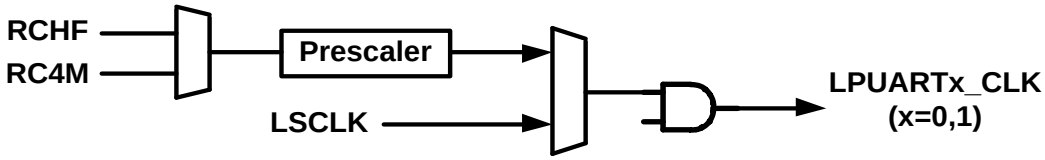
LPOSC calibration should not use XTLF, because XTLF can be used for communication if there is XTLF. Therefore, the LPOSC calibration circuit should use RCHF to work, and the recommended reference input is 8MHz.

In ACTIVE mode, LPUART can also use RCHF to work, at this time the clock accuracy will be higher than LPOSC, in order to obtain better timing fault tolerance performance. When working with RCHF, the prescaler circuit prescales the RCHF to obtain a clock frequency close to 32768Hz. For example, when the RCHF is 8M/16M/24M, the prescaler frequency division coefficient should be 244/488/732 respectively.

In ACTIVE and LP ACTIVE modes, LPUART can use RCMF clock to work. The temperature

coefficient of RCMF is poorer than RCHF, so it is recommended to calibrate RCMF before LPUART work.

See the figure below for the LPUART working clock structure. This part of the functions and registers are implemented in the CMU module.



22.5 Character Description

The basic timing of LPUART characters is similar to a standard UART. Each character frame contains at least 1bit START and at least 1bit STOP bits, data lengths can be configured to be 6-9bits, and parity bits can be selected.

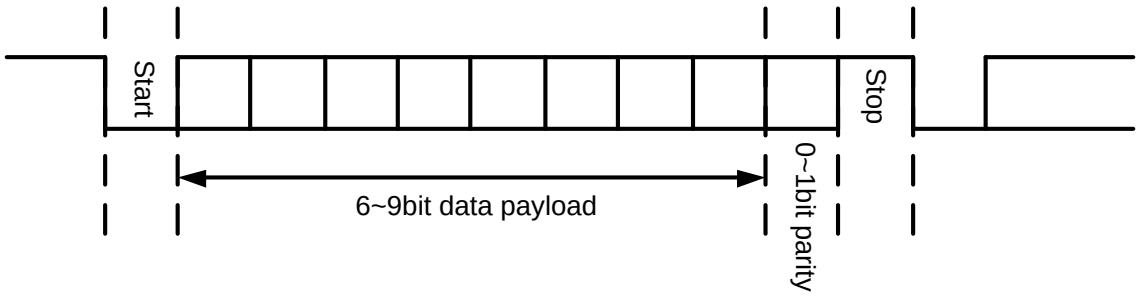


Figure 22-2 Character format

LPUART supports multiple frame formats controlled by the UARTxCSR.PDSEL register and UARTxCSR.PARITY register, as shown in the following table:

PDSEL	PARITY	Frame Format[1]
00	00	[Start | 7 bits data | Stop]
	01, 10	[Start | 7 bits data | Parity | Stop]
01	00	[Start | 8 bits data | Stop]
	01, 10	[Start | 8 bits data | Parity | Stop]
10	00	[Start | 9 bits data | Stop]
	01, 10	[Start | 9 bits data | Parity | Stop]
11	00	[Start | 6 bits data | Stop]
	01, 10	[Start | 6 bits data | Parity | Stop]

Table 22-1 LPUART Frame Format

[1]：The Stop bit can be either 1bit or 2bit, depending on the STOPCFG register.

Note: THE PDSEL register is used to configure the data length of the frame. The communication frame length is [start bit + data bit + parity bit + stop bit].

22.6 Functional Description

22.6.1 Bit sampling

Since the clock of LPUART is only around 32KHz, a standard UART cannot support 9600bps communication. Bit modulation needs to be introduced.

The software needs to reasonably configure the modulation control register MCTL according to different communication baud rates. The recommended configuration parameters are listed as follows:

Baud	MCTL												
	Bit0 (start)	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12
9600	0	1	0	0	1	0	1	0	1	0	1	0	0
4800	1	1	0	1	1	1	1	1	0	1	1	1	1
2400	1	1	0	1	1	0	1	1	0	1	1	0	1
1200	0	1	0	0	1	0	0	1	0	0	1	0	0
600	0	1	1	0	1	0	1	1	0	1	1	0	1
300	0	1	0	0	0	0	1	0	0	0	0	1	0

Table 22-2 LPUART Bit modulation factor

22.6.2 Data reception procedure

- Select the desired baud rate by programming LPUBAUD register
- According to the baud rate, select the appropriate modulation parameters and configure the MCTL register
- Configure the LPUCON register to select frame format, polarity, interrupt parameters, etc.
- Set LPUEN register to enable data receive
- Wait for interrupt

22.6.3 Data transmit procedure

- Select the desired baud rate by programming LPUBAUD register
- According to the baud rate, select the appropriate modulation parameters and configure the MCTL register
- Configure the LPUCON register to select frame format, polarity, interrupt parameters, etc.
- Set LPUEN register to enable data transmit
- Wait for interrupt

22.6.4 Use DMA for data Receiving and Transmitting

When the DMA is enabled, the LPUART will automatically generate the corresponding DMA request when the TX buffer is empty or the RX buffer is full. The application needs to configure DMA channel connections, connect specific channels to LPUART peripherals, set the RAM pointer, and enable DMA channels. After that, DMA will automatically respond to the LPUART request and complete the data transfer between RAM and LPUART.

Application example: LPUART1 receive using DMA

- Configure the DMA channel X as LPUART1_RX
- Set corresponding channel parameters: RAM pointer, address increment or decrement, channel priority, transmission length, interrupt, etc
- Enable corresponding DMA channels
- Configure the LPUART1 parameters
- Enable LPUART1by writing LPUEN.RXEN=1, and wait for incoming data
- LPUART1 automatically generates DMA requests upon arrival of data
- DMA responds to the request, reads the LPUART1 receive buffer, and writes to the specified RAM address

22.6.5 Data receiving and wake up in Sleep mode

LPUART supports data reception and wake up the chip in Sleep and DeepSleep modes. At this time, the power consumption of the chip is extremely low, and it keeps monitoring the RXD pin until a specific event arrives and wakes up the chip to exit the sleep mode.

- Configure the LPUBAUD register to determine the baud rate
- Choose appropriate modulation parameters according to the baud rate and configure the MCTL register
- Configure the LPUCON register, select the frame format and polarity, and select the wake-up event to be the START bit through LPUxCR.RXEV, one frame reception complete, one frame

data match, or RXD falling edge detection

- Configure the LPUEN register to open the receive enable
- The software enters Sleep/DeepSleep

22.6.6 DMA Data Receiving and Transmitting in LPRUN mode

Through LPUART and DMA, the software can automatically send and receive a certain amount of LPUART data in LPRUN mode without CPU intervention, while ensuring that the power consumption of the whole chip is less than 10uA under typical conditions.

- Configure the LPUBAUD register to determine the baud rate.
- According to the baud rate, select the appropriate modulation parameters and configure the MCTL register.
- Configure the LPUCON register to select frame format, polarity, interrupt parameters, etc.
- Configure the DMA channel control register and select LPUART
- If you need to send data, write the outgoing data to the specified location in RAM
- Configure DMA data transmit/receive length and RAM pointer
- Select the system clock as LSCLK
- Enter the LPRUN by software
- Configure the LPUEN register to enable data transfer
- Software can execute WFI/WFE and wait for an interrupt to wake up

22.6.7 Transmission completion interrupt in DMA mode

When LPUART sends data over DMA, DMA produces a DMA channel interrupt after a specified length of data transfer is completed. But when the channel interrupt occurs, the last frame of data has just been written to the LPUART TX buffer and has not yet been sent.

By configuring the DMATXIFCFG register, it is possible to generate a transmission complete interrupt (buffer empty or shift register empty) when the DMA transfer has completed and the last frame data has been sent.

The software procedure is described below：

- Configure DMA channels as LPUART transmit
- Close the DMA channel interrupt enable

- Set LPUART TXBE_IE or TXSE_IE registers to allow interrupts to be generated
- Set the DMATXIFCFG register, allowing only the last frame of data to produce interrupt output
- Prepare data and enable DMA
- LPUART transmits continuously until the last frame, with no TXBE or TXSE interrupts are generated
- After the last frame is sent, LPUART produces a TXBE or TXSE interrupt.

The following table assumes that LPUART sends N frames via DMA：

TXBE_IE TXSE_IE	DMATXIFCFG	Frame No.	TXBE TXSE	LPUART interrupt
0	x	1~N	After each frame is sent, set	Inactive
1	0	1~N	After each frame is sent, set	Inactive
	1	1~N-1	After each frame is sent, set	Inactive
		N	After each frame is sent, set	Active

22.7 Register

Offset	Name	Symbol
LPUART0 Register(module base address：0x40010400)		
0x00000000	LPUART0 Control Status Register	LPUART0_CSR
0x00000004	LPUART0 Interrupt Enable Register	LPUART0_IER
0x00000008	LPUART0 Interrupt Status Register	LPUART0_ISR
0x0000000C	LPUART0 Baud rate Modulation Register	LPUART0_BMR
0x00000010	LPUART0 Receive Buffer Register	LPUART0_RXBUF
0x00000014	LPUART0 Transmit Buffer Register	LPUART0_TXBUF
0x00000018	LPUART0 data Matching Register	LPUART0_DMR
LPUART1 Register(module base address：0x40018400)		
0x00000000	LPUART1 Control Status Register	LPUART1_CSR
0x00000004	LPUART1 Interrupt Enable Register	LPUART1_IER
0x00000008	LPUART1 Interrupt Status Register	LPUART1_ISR
0x0000000C	LPUART1 Baud rate Modulation Register	LPUART1_BMR
0x00000010	LPUART1 Receive Data Register	LPUART1_RXBUF
0x00000014	LPUART1 Transmit Data Register	LPUART1_TXBUF
0x00000018	LPUART1 data Matching Register	LPUART1_DMR

22.7.1 LPUARTx Control Status Register (LPUARTx_CSR)

NAME	LPUARTx_CSR(x=0,1)							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							BUSY
access	U-0							R-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-				WKBYTE_CFG	-	RXEV	
access	U-0				R/W-0	U-0	R/W-00	
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				IOSWAP	DMATXIFCFG	BITORD	STOPCFG
access	U-0				R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	PDSEL		PARITY		RXPOL	TXPOL	RXEN	TXEN
access	R/W-00		R/W-00		R/W-0	R/W-0	R/W-0	R/W-0

Bit	Name	Functional Description
31:25	-	**Reserved, read as 0**

Bit	Name	Functional Description
24	**BUSY**	LPUART communication flag, read-only (Busy) 1：UART is communicating. 0：UART idle
23:20	-	**Reserved, read as 0**
19	**WKBYTE_CFG**	Wakeup Byte Config 1：A wake-up interrupt is triggered when 1 byte is received and the parity and STOP bits are correct 0：After receiving 1 byte, do not check the check bit and STOP bit, trigger the wake-up interrupt directly
18	-	**Reserved, read as 0**
17:16	**RXEV**	The wake interrupt event configuration is used to control under which events the wake interrupt is provided to the CPU (Receive Wakeup Event) 00：START bit detects wake up 01：The 1byte data is received 10：Received data match successfully 11：RXD falling edge detection
15:12	-	**Reserved, read as 0**
11	**IOSWAP**	RX and TX pin swapping (IO swapping) 0: Default pin sequence (consistent with package drawing) 1: Swap the pin sequence
10	**DMATXIFCFG**	DMA transmit completion interrupt enablement is only valid if LPUART is sending through DMA (DMA Transmit Interrupt Config) 1：In the case of IE=1, interrupt is generated after the last frame is sent by DMA; Interrupt is masked before the last frame has been sent 0：It is up to IE to decide whether to mask interrupt or not
9	**BITORD**	Bit order in which data is sent/received (Bit Order) 0：LSB first 1：MSB first
8	**STOPCFG**	Stop bit configuration, only valid for transmitting 0：1 Stop bit 1：2 Stop bit
7:6	**PDSEL**	Select the data length of each frame; This register is valid for both sending and receiving data. (Payload Data length Select) 00：7 bits 01：8 bits 10：9 bits 11：6 bits

Bit	Name	Functional Description
5:4	**PARITY**	Parity bit configuration; This register is valid for both transmitting and receiving data (Parity) 00：None 01：Odd 10：Parity 11：RFU
3	**RXPOL**	Receive Polarity Configuration 0：Standard 1：Inverted
2	**TXPOL**	Transmit Polarity Configuration 0：Standard 1：Inverted
1	**RXEN**	Receive enable 1: enable 0: disable
0	**TXEN**	Transmit enable 1: enable 0: disable

22.7.2 LPUARTx Interrupt Enable Register (LPUARTx_IER)

NAME	LPUARTx_IER(x=0,1)							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			RXEV_IE	-	RXERR_IE	-	RXBF_IE
access	U-0			R/W-0	U-0	R/W-0	U-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						TXBE_IE	TXSE_IE
access	U-0						R/W-0	R/W-0

Bit	Name	Functional description
31:13	**-**	**Reserved, read as 0**
12	**RXEV_IE**	Receive Event Interrupt Enable，1 effective
11	**-**	**Reserved, read as 0**
10	**RXERR_IE**	Receive Error Interrupt Enable，1 effective

Bit	Name	Functional description
9	-	**Reserved, read as 0**
8	**RXBF_IE**	Receive Buffer Full Interrupt Enable，1 effective
7:2	-	**Reserved, read as 0**
1	**TXBE_IE**	Transmit Buffer Empty Interrupt Enable，1 effective
0	**TXSE_IE**	Transmit Shift register Interrupt Enable，1 effective

22.7.3 LPUARTx Interrupt Status Register (LPUARTx_ISR)

NAME	LPUARTx_ISR(x=0,1)							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							RXEVF
access	U-0							R/W-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-				TXOV	PERR	FERR	OERR
access	U-0				R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							RXBF
access	U-0							R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						TXBE	TXSE
access	U-0						R/W-0	R/W-0

Bit	Name	Functional description
31:25	-	**Reserved, read as 0**
24	**RXEVF**	Receive Event Interrupt Flag. Set by hardware, write 1 to clear. The interrupt flag trigger source is configured with the LPUxCR.RXEV register.
23:20	-	**Reserved, read as 0**
19	**TXOV**	Transmit Overflow Error. Set by hardware, write 1 to clear. TXOV is triggered when software writes new data into TX buffer when TX buffer is full.
18	**PERR**	Parity Error Interrupt Flag. Set by hardware, write 1 to clear.
17	**FERR**	Frame Error Interrupt Flag. Set by hardware, write 1 to clear.
16	**OERR**	Receive Buffer Overflow Error Interrupt Flag. Set by hardware, write 1 to clear. OERR is triggered when new data has been received when RX buffer is full.
15:9	-	**Reserved, read as 0**
8	**RXBF**	Receive Buffer Full Interrupt Flag. Set by hardware, write 1 to clear.
7:2	-	**Reserved, read as 0**

Bit	Name	Functional description
1	**TXBE**	Transmit Buffer Empty Interrupt Flag. Set by hardware, cleared by writing data into TX buffer
0	**TXSE**	Transmit Shift register Empty Interrupt Flag. Set by hardware, cleared by writing 1 or byte is moved into transmit shift register

22.7.4 LPUARTx Baud rate Modulation Register (LPUARTx_BMR)

NAME	LPUARTx_BMR(x=0,1)							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-			MCTL[12:8]				
access	U-0			R/W-00000				
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	MCTL[7:0]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					BAUD		
access	U-0					R/W-000		

Bit	Name	Functional description
31:29	**-**	**Reserved, read as 0**
28:16	**MCTL**	LPUART Bit Modulation Control
15:3	**-**	Reserved, read as 0
2:0	**BAUD**	Baud rate control (bps) 000：9600 001：4800 010：2400 011：1200 100：600 101/110/111：300

22.7.5 LPUARTx Receive Buffer (LPUARTx_RXBUF)

NAME	LPUARTx_RXBUF(x=0,1)							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							RXBUF[8]
access	U-0							R-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RXBUF[7:0]							
access	R-0000 0000							

Bit	Name	Functional description
31:9	**-**	**Reserved, read as 0**
8:0	**RXBUF**	Receive Buffer

22.7.6 LPUARTx Transmit Buffer Register (LPUARTx_TXBUF)

NAME	LPUARTx_TXBUF(x=0,1)							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							TXBUF[8]
access	U-0							R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TXBUF[7:0]							
access	R/W-0000 0000							

Bit	Name	Functional description
31:9	-	**Reserved, read as 0**
8:0	**TXBUF**	Transmit Buffer

22.7.7 LPUARTx Data Matching Register (LPUARTx_DMR)

NAME	LPUARTx_DMR(x=0,1)							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							MATD[8]
access	U-0							R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	MATD[7:0]							
access	R/W-0000 0000							

Bit	Name	Functional description
31:9	-	**Reserved, read as 0**
8:0	**MATD**	Matching data register. If RXEV=10, the RXEVF interrupt is triggered when the first byte received is matched with MATD, which can be used to wake up the MCU by certain data receiving in the sleep mode.

23 SPI

23.1 Introduction

Serial Peripheral Interface (SPI) is a serial synchronous communication peripheral to exchange data over 4 wires. The chip provides 2 SPI modules that can be configured as master or slave devices.

Features:

- Full duplex 4-wire serial synchronous transceiver (SCLK, MOSI, MISO, SSN)
- MISO and MOSI pin swap
- Half-duplex 4-wire serial synchronous transceiver (SCLK, SDATA, SSN, DCN)
- 2 independent channels
- Master-slave mode
- Programmable clock polarity and phase
- Programmable bit rate
- Programmable data length (8/16/24/32bits)
- Maximum baud rate up to FAPBCLK/2
- End-of-transmission interrupt flag
- Write conflict error flag
- Master mode error detection, protection and interrupt flags
- DMA support

23.2 Block diagram

The following figure shows the structure diagram of the SPI module.

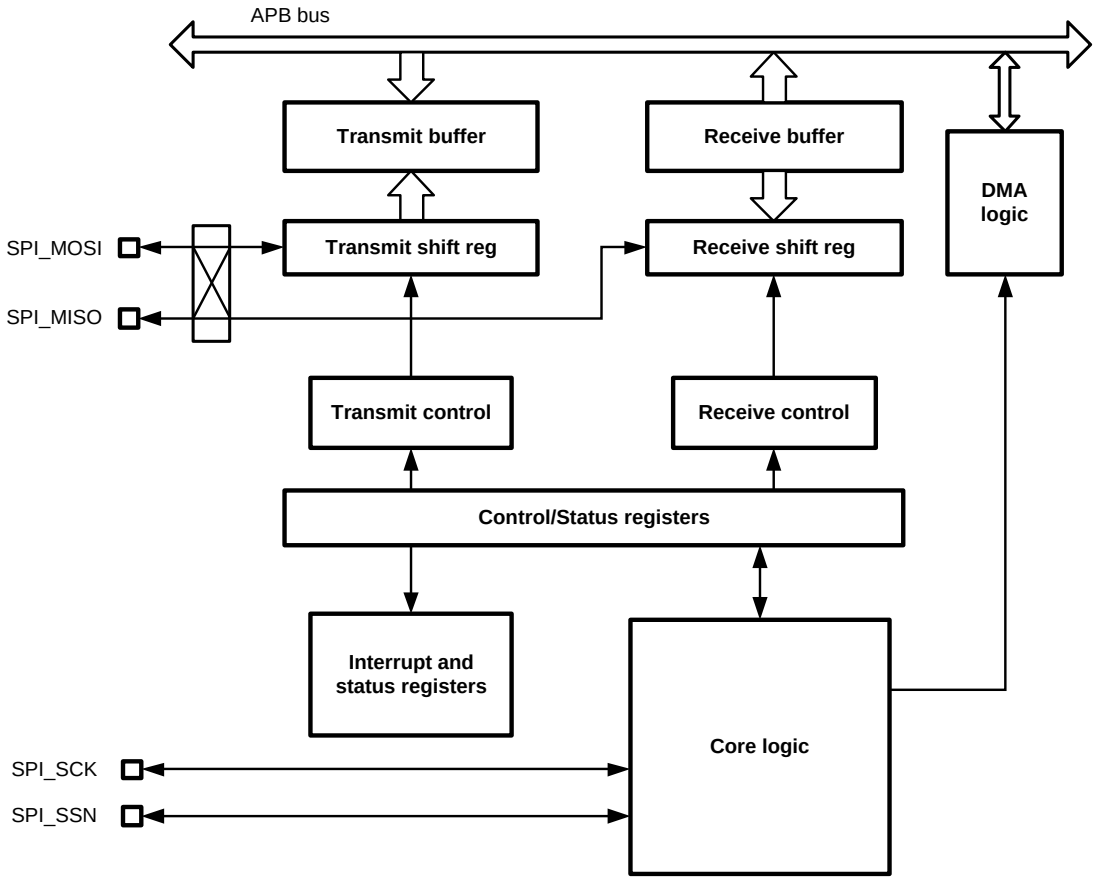


Figure 23-1 Block diagram of SPI

23.3 Pin definition

The SPI module uses four pins to communicate with external devices. The functions of these four pins are defined differently in full-duplex and half-duplex modes, as shown in the table below.

Pin	SPIx	Full Duplex	Function	Half Duplex	Function
PB8/PD2	SPI1	SSN	Chip selection signal	SSN	Chip selection signal
PB9/PD3		SCLK	Clock	SCLK	Clock
PB10/PD4		MISO	Master Input Slave Output	DCN	Command/Data Flag
PB11/PD5		MOSI	Master Output Slave Input	SDATA	Data
PC7	SPI2	SSN	Chip selection signal	SSN	Chip selection signal
PC8		SCLK	Clock	SCLK	Clock
PC9		MISO	Master Input Slave Output	DCN	Command/Data Flag
PC10		MOSI	Master Output	SDATA	Data

			Slave Input		

23.4 Interface timing

In order to be compatible with different SPI protocols, the timing of the SPI serial clock can be set by the clock phase selector bit (CPHA) and the clock polarity selector bit (CPOL) to produce four different combinations. To ensure correct data transfer, the timing configuration of the master and slave devices must be the same.

When in slave mode or when the SPI enable bit (SPE) is 0, there is no serial clock output on the SCK pin.

23.4.1 CPHA=0

If CPHA=0, the SPI module samples data on the first edge of the serial clock, i.e.

if CPOL=1, SCK stays high at bus IDLE, the SPI samples data on the falling edge of the serial clock and sends data on the rising edge of the serial clock.

If CPOL=0, SCK stays low at bus IDLE, the SPI samples data on the rising edge of the serial clock and sends data on the falling edge of the serial clock.

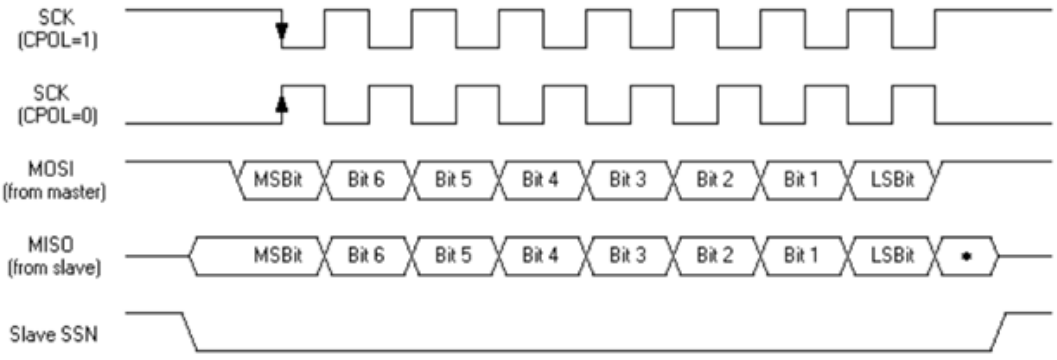


Figure 23-2 SPI Data/Clock Timing Diagram (CPHA=0)

23.4.2 CPHA=1

With CPHA=1, the SPI module samples data on the second edge of the serial clock, i.e.

if CPOL=1, SCK stays high at bus IDLE, samples data on the rising edge of the serial clock and sends data on the falling edge of the serial clock.

If CPOL=0, SCK stays low at bus IDLE, samples data on the falling edge of the serial clock, and sends data on the rising edge of the serial clock.

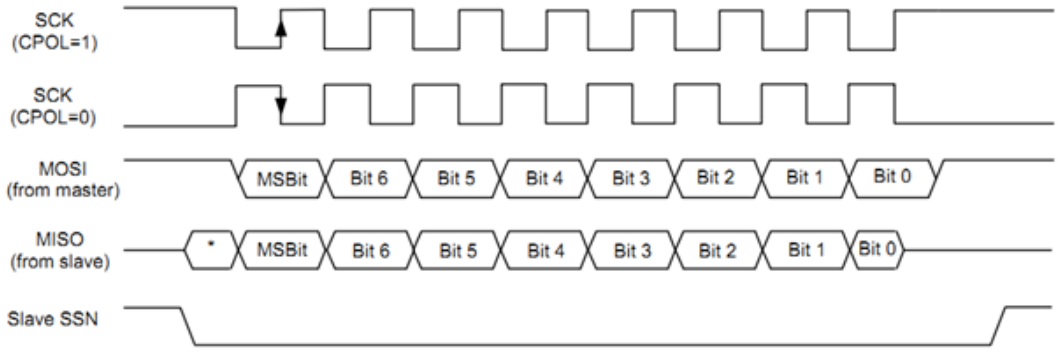


Figure 23-3 SPI Data/Clock Timing Diagram (CPHA=1)

23.4.3 4-wire half-duplex mode (master)

The 4-wire half-duplex mode can support interactive communication with dot matrix LCD or TFT screens. In this mode, the high or low DCN signal is used to distinguish whether a command frame or a data frame is being sent. Both bidirectional data are sent and received via SDATA (MOSI) pin, and the data direction switching is done automatically by hardware. the SPI of FM33LE0xxA only supports 4-wire half-duplex master mode, slave mode is not available.

All communication is initiated by the master, which first sends command frames and then transmits data frames. Command frames and data frames are distinguished by the DCN signal line. The master can write data to and read data from the slave through the 4-wire half-duplex interface.

4-wire half-duplex write operation

The software indicates that the master wants to initiate a write transaction by clearing the HD_RW register.

Before the master initiates a write transaction, it first sends a write command frame. When the write command frame is sent, if the transmit buffer is empty, the hardware will pull up SSN and stop SCLK sending; if new data has been written to the transmit buffer, the hardware will send subsequent data frames continuously.

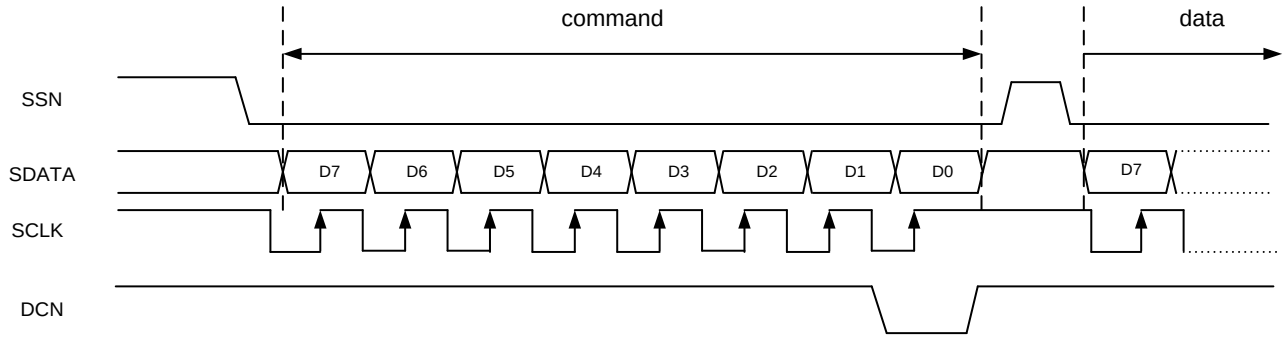


Figure 23-4 4-wire half-duplex write operation

DCN is sampled at the 8th clock rising edge, if it is 0, meaning the current frame is a command frame. Before sending the command frame, the software needs to write 0 to the DCN register, and the hardware automatically sets the DCN register to 1 after the command frame is sent.

4-wire half-duplex read operation

The software indicates that the master wants to initiate a read operation by setting the HD_RW register.

The 4-wire half-duplex read transaction supports 8-bit, 24-bit, and 32-bit data length. When the master initiates a read transaction, it first sends a read command frame. When the read command frame is sent, a dummy cycle can be sent according to the register configuration. During the dummy cycle, the SCLK clock is sent normally, but the master does not drive SDATA and does not accept SDATA input.

After completing the command frame and dummy cycle (optional), the 4-wire half-duplex SPI automatically enters the receive state, the SDATA signal is driven by the slave instead, and the data frames received by the master will be written to the receive buffer. After each data frame is received, the RXBF interrupt flag register will be set. The software should read the data in the receive buffer in time. If both the receive buffer and the receive shift register are full, the hardware will stop SCLK transmission and suspend reading data from the slave until the software or DMA has read the receive buffer.

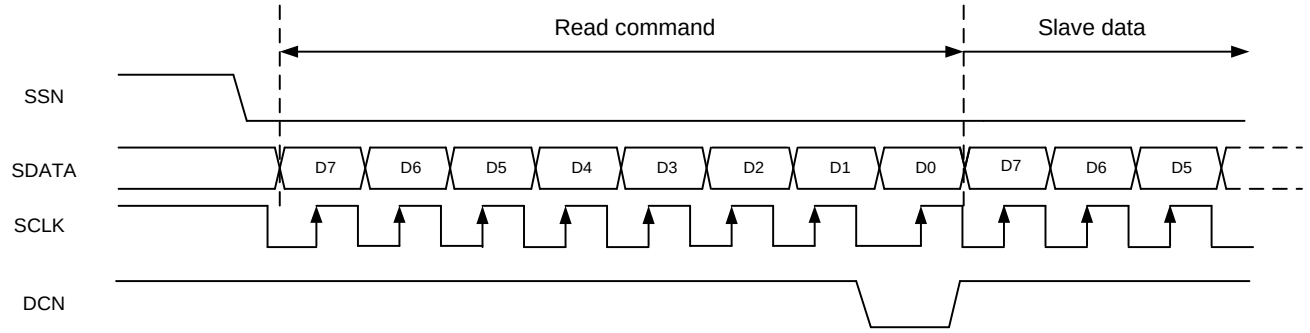


Figure 23-5 4-wire half-duplex read operation (no dummy cycle)

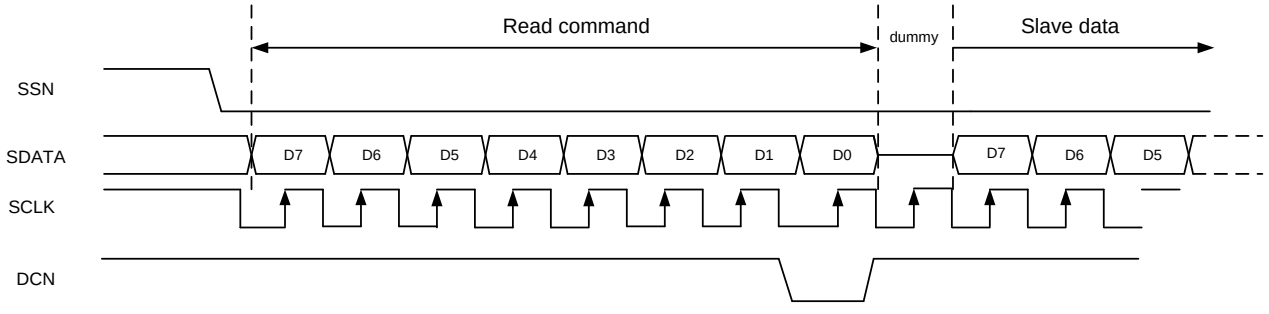


Figure 23-6 4-wire half-duplex read operation (with dummy cycle)

23.5 Functional description

23.5.1 I/O Configuration

Master Output, Slave Input (MOSI)

The Master Output, Slave Input (MOSI) pin is the output of the master device and the input of the slave device and is used for serial data transfer from the master to the slave device. This pin is an output when the SPI is configured as a master device and an input when the SPI is configured as a slave device. The MSB comes first when data is transferred.

Master Input, Slave Output (MISO)

The Master Input, Slave Output (MISO) pin is an output from the device and an input to the master device for serial data transfer from the slave device to the master device. This pin is an input when the SPI is configured as a master device and an output when the SPI is configured as a slave device. The MSB comes first for data transfer.

Serial Clock (SCK)

The serial clock (SCK) pin is an output of the master device and an input of the slave device and is used to synchronize serial data transfers between the master and slave devices on the MOSI and MISO lines. This pin outputs the clock when the SPI is configured as a master device and is an input when the SPI is configured as a slave device.

Slave Select（SSN）

The Slave Select (SSN) pin is used to select slave device as shown in Figure 23-2. The SSN pin must be connected high when the SPI is configured as a master device and low when the SPI is configured as a slave device.

The SPI master and slave devices are connected as shown in the figure below.

The MOSI, MISO and SCK of the master and slave devices are connected together, and the SSN of the master device must be connected high and the SSN of the slave device must be connected low. The master and slave devices are connected into a loop through MOSI and MISO. The master device outputs the clock, and when data is transferred, the master device outputs data through MOSI and the slave device outputs data through MISO. After one byte data transfer, the master and slave devices will exchange 8-bit shift register values.

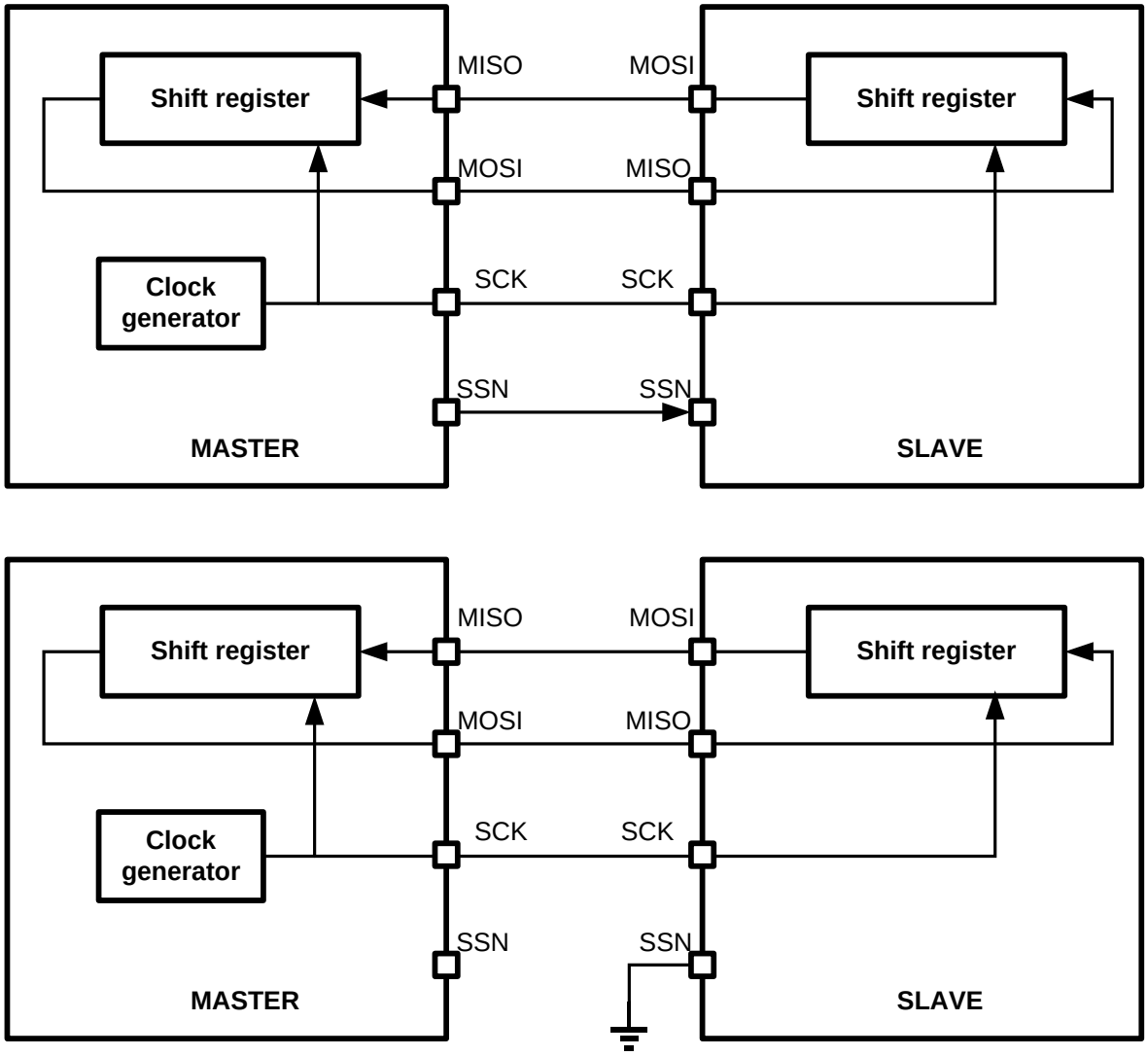


Figure 23-7 SPI Master/SPI Slave Interconnect

23.5.2 Full-duplex data communication

The SPI module enables full-duplex communication by default. If continuous un-interrupted data communication is required, the software needs to fill TX BUFFER in time. Even if the software only uses SPI for data reception, due to the full-duplex property of SPI, the software still needs to write to TX BUFFER, which should be invalid data for example all 0 or all F according to MOSI invalid status.

Transmit Buffer

Software or DMA writes the data to be sent to the TX buffer (SPIx_TXBUF register), and when the transmitting starts, hardware copies the data from the TX buffer to the shift register and starts shifting out. After the data is transferred from the transmit buffer to the shift register, the transmit buffer empty flag (TXBE) is set, indicating that new data can be written to the TXBUF; if the TXIE

register is set, an interrupt is generated. The TXBE register can be cleared by writing data to the TXBUF.

If new data is written to the transmit buffer before the shift register shift is completed, continuous data transmission is guaranteed. Writing TXBUF with TXBE = 0 generates a data conflict, refer 23.5.6 Data conflicts.

Receive Buffer

When the SPI completes a frame of data reception, the received data is copied from the shift register to the receive buffer (SPIx_RXBUF register), and the RXBF flag is set to indicate that there is data in the RXBUF to be processed. If the RXIE register is set, an interrupt is generated. The RXBF flag can be cleared by reading the RXBUF.

Reading RXBUF without RXBF set will return the last received data; if the application does not process RXBF in time and the new data finishes receiving with RXBF set, a data conflict is generated, see 23.5.6 Data conflicts.

BUSY flag

The BUSY register is set when the SPI is sending and receiving data. This register can be used in some scenarios to determine if the last frame of data has been transmitted. For example, TXBE just indicates that the data has been shifted and sent, but the real transmission is ongoing until BUSY flag is cleared.

How to start SPI communication

In master mode, the recommended procedure to initiate SPI communication is as follows.

- Configure SPI module
- Set the SPIEN
- Write data to the TXBUF and the SPI automatically starts data transfer

In slave mode, it is recommended that the application complete the configuration and enable before the master starts sending SCK. Write the first frame of data to be sent to TXBUF, and wait for the master to send SCK to start communication.

How to end SPI communication

In master mode, the recommended procedure to end SPI communication is as follows.

- Wait for the RXBF and TXBE flags to set, when there is still the last frame of data being sent in the shift register

- Query the BUSY flag until BUSY is 0 and the last frame of data is sent and received
- Turn off the SPI module and read the last received data frame if needed

In slave mode, the application can shut down the SPI module after reading any frame of data, and the data that has been shifted into the shift register before shutdown will be ignored.

23.5.3 TX-ONLY mode

In some cases, the SPI communication is half-duplex. When the master only needs to transmit, the TX-ONLY mode can be used by setting the TXO register. Under TX-only the data received by the MISO will not be written to the RX Buffer, and accordingly the RXBF interrupt flag will not be set.

The TXO auto-clear function can be implemented by setting TXO_AC. In TX-ONLY mode, if the TX buffer is empty (TXBE is set) and the transmit shift register is empty, the TXO register is automatically cleared and the TX-ONLY state is exited.

23.5.4 RX-ONLY mode

If the SPI master only needs to perform reception, it enters RX-ONLY mode by setting the RXO register. The SPI module can perform continuous data reception without software writing to the TX Buffer, and the MOSI will hold the IDLE level and TXBE interrupt register will not be set.

23.5.5 Master SSN Control

The SPI master supports hardware or software-controlled SSN signal.

SSN is controlled by hardware when the SSNSEN register is cleared to zero; if the SSNM register is set, the SPI will pull SSN high after each data frame sent, and the SSN high time is configured by the WAIT register (several SCK clock cycles).

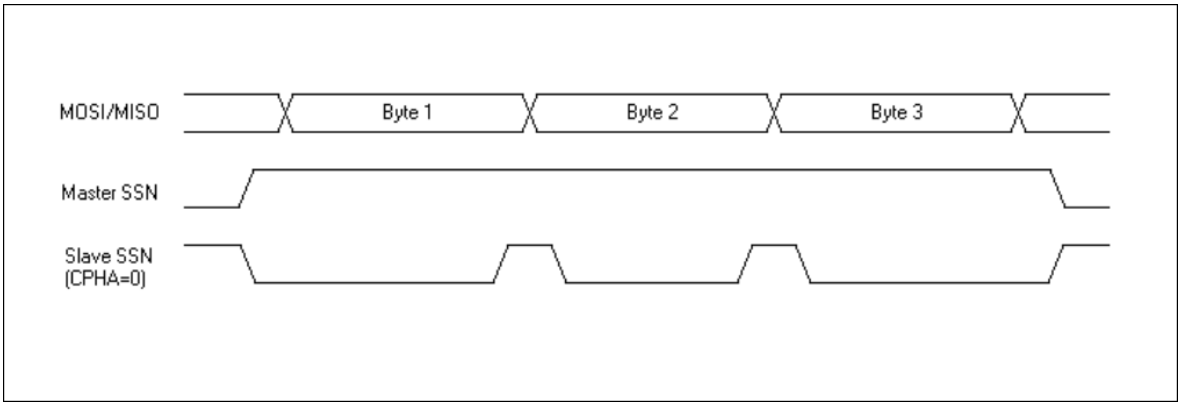


Figure 23-8 SPI SSN Timing Diagram (SSNM=1, CPHA=0)

If the SSNM register is reset, the SPI does not pull up the SSN after each frame of data sent, but goes directly to the next frame transmitting.

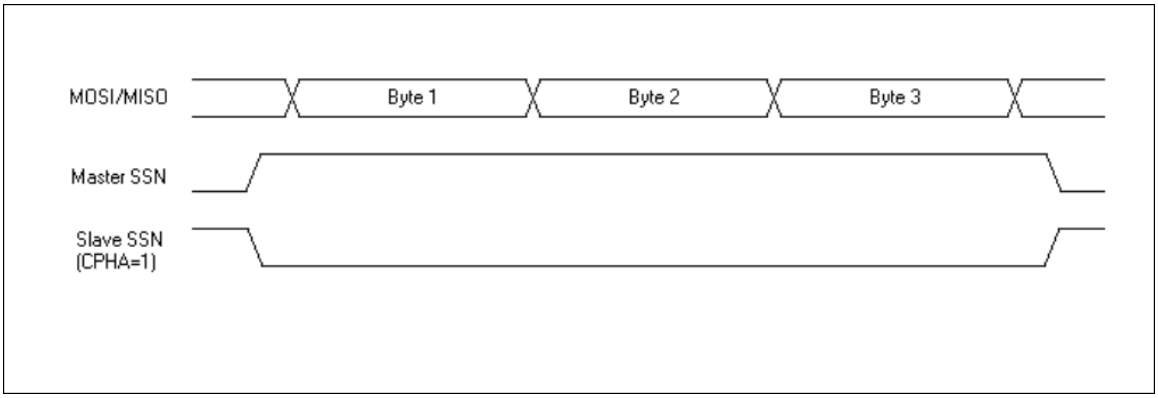


Figure 23-9 SPI SSN Timing Diagram (SSNM=0)

When the SSNSEN register is set, SSN is controlled by software. Software can directly manipulate the SSN level by writing the SPIx_CR2.SSN register bits.

23.5.6 Data conflicts

When the TX Buffer data of the SPI has not been read into the shift register, or when the data in the RX Buffer of the SPI has not been read by software or DMA, a write operation to the TX Buffer or RX Buffer will generate a corresponding conflict error and the TXCOL/RXCOL bits will be set up, generating an interrupt. The write data that causes the conflict will be ignored. Data conflict errors are generated in both master and slave modes.

Write operations to the TX Buffer are initiated by the Master module inside the chip, including the CPU, DMA, etc. The write operation to RX Buffer is initiated by the external SPI device.

When a data conflict occurs, the original data in TX Buffer and RX Buffer is not refreshed and the newly written data is lost.

23.5.7 SPI Transceiver via DMA

When the SPI module is enabled, the SPI module automatically generates the corresponding DMA requests when both the transmit buffer is empty and the receive buffer is full. The application software needs to configure the DMA channel connection in advance, connects the specific channel to the SPI peripheral, set the RAM pointer, and enable the DMA channel. Thereafter the DMA will automatically respond to the SPI request and complete the data handling between RAM and SPI.

Note: If DMA is used for transmitting and receiving full-duplex communication, the software should enable the DMA transmit channel first, and then the DMA receive channel; the opposite may cause the SPI to send an extra byte of dummy data.

Using DMA for SPI reception

- Configure DMA channel 3 or 5 as SPI_RX
- Set RAM pointer address, address increment/decrement, channel priority, transfer length, interrupt settings, etc.
- Enabling the corresponding DMA channel
- Configure SPI module parameters
- Enable the SPI module and wait for data reception
- SPI automatically generates DMA request after receiving data
- DMA responds to the request, reads the SPI receive cache register, and writes to the specified RAM address
- When the specified length of DMA transfer is completed, the DMA will ignore subsequent requests and generate a transfer completion interrupt, and the software should handle the

interrupt and shut down the SPI

- If data is received again before closing SPI, software can clear RXBUF by writing RXBFC

Using DMA for SPI transmission

The DMA transmit process is similar to the receive process described above, with the main difference being that the software cannot shut down the SPI immediately after the specified length of DMA transmission is completed, because the last frame of data is still being shifted and sent at this time, so the software needs to query the BUSY flag until the end of shifting and sending, and then shut down the SPI module.

Data frame length and RAM data organization

The SPI transmission frame length can be configured to 8, 16, 24, or 32 bits.

When the data frame length is 8bit, DMA carries 1byte at a time, 4 carries fill one address of RAM, and small end storage is used within the word:

RAM word: {data3, data2, data1, data0}

When the length of the data frame is 16 bits, the DMA carries 2 bytes at a time, 2 carries fill one RAM address, and the word is stored in the small end: {data3, data2, data1, data0 }

RAM word: {data1, data0}

When the data frame length is 24 bits, the DMA carries 1 word at a time, filling one RAM address with one carry, but the valid data occupies only the lower 24 bits of the RAM word: { data1, data0 }

RAM word: {8'h0, data0}

When the data frame length is 32 bits, the DMA carries 1 word at a time, filling one RAM address in one carry: {8'h0, data0 }

RAM word: {data0}

23.6 Register

offset	name	symbol
SPI1(Base address:0x40018C00)		
0x00000000	SPI1 Control Register1	SPI1_CR1
0x00000004	SPI1 Control Register2	SPI1_CR2
0x00000008	SPI1 Control Register3	SPI1_CR3
0x0000000C	SPI1 Interrupt Enable Register	SPI1_IER
0x00000010	SPI1 Status Register	SPI1_ISR
0x00000014	SPI1 Transmit Buffer	SPI1_TXBUF
0x00000018	SPI1 Receive Buffer	SPI1_RXBUF
SPI2(Base address:0x40010800)		
0x00000000	SPI2 Control Register1	SPI2_CR1
0x00000004	SPI2 Control Register2	SPI2_CR2
0x00000008	SPI2 Control Register3	SPI2_CR3
0x0000000C	SPI2 Interrupt Enable Register	SPI2_IER
0x00000010	SPI2 Status Register	SPI2_ISR
0x00000014	SPI2 Transmit Buffer	SPI2_TXBUF
0x00000018	SPI2 Receive Buffer	SPI2_RXBUF

23.6.1 SPIx Control Register 1 (SPIx_CR1)

NAME	SPIx_CR1(x=1,2)							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				IOSWAP	MSPA	SSPA	MM
access	U-0				R/W-0	R/W-0	R/W-0	R/W-1
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	WAIT		BAUD			LSBF	CPOL	CPHA
access	R/W-00		R/W-000			R/W-0	R/W-0	R/W-0

bit	name	functional description
31:12	-	RFU: **Reserved, read as 0**
11	**IOSWAP**	MOSI and MISO pin swapping (IO swapping) 0: Default pin order 1: Swapping pin order
10	**MSPA**	Master Sampling Position Adjustment, the Master's sampling position adjustment for MISO signals, used to compensate for PCB alignment delays when communicating at high speed 1: Sampling point delayed by half SCK cycle 0: No adjustment

bit	name	functional description
9	**SSPA**	Slave Sending Position Adjustment, Slave MISO sending position adjustment 1: Sending half SCK cycle ahead 0: No adjustment
8	**MM**	Master/Slave mode selection 1: Master mode 0: Slave mode
7:6	**WAIT**	In Master mode, at least (1+WAIT) SCK cycle wait time is added after each frame is sent before transmitting the data of the next frame. If SSN is controlled by hardware and SSNM=1, hardware will pull up SSN automatically.
5:3	**BAUD**	Master mode baud rate configuration bits: 000: $f_{APBCLK}/2$ 001: $f_{APBCLK}/4$ 010: $f_{APBCLK}/8$ 011: $f_{APBCLK}/16$ 100: $f_{APBCLK}/32$ 101: $f_{APBCLK}/64$ 110: $f_{APBCLK}/128$ 111: $f_{APBCLK}/256$ These bits cannot be modified while communication is in progress.
2	**LSBF**	Frame format (LSB First) 0: MSB is sent first 1: LSB is sent first Note: The value of this bit cannot be changed while communication is in progress.
1	**CPOL**	Clock Polarity Selection (Clock Polarity) 1: Serial clock stops at high level 0: Serial clock is stopped at low level Note: The value of this bit cannot be changed while communication is in progress Note: The value of this bit cannot be changed when SSN is low
0	**CPHA**	Clock Phase Selection (Clock Phase) 1: The second clock edge is the first capture edge 0: The first clock edge is the first capture edge Note: The value of this bit cannot be changed while communication is in progress.

23.6.2 SPIx Control Register 2 (SPIx_CR2)

NAME	SPIx_CR2(x=1,2)							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DUMMY_EN	-			RXO	DLEN		HALFDUPLEX
access	R/W-0	U-0			R/W-0	R/W-00		R/W-0

bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	HD_RW	CMD8b	SSNM	TXO_AC	TXO	SSN	SSNSEN	SPIEN
access	R/W-0	R/W-1	R/W-0	R/W-1	R/W-0	R/W-1	R/W-0	R/W-0

bit	name	functional description
31:16	-	RFU: **Reserved, read as 0**
15	**DUMMY_EN**	Whether to insert a dummy cycle in the read operation under 4-wire half-duplex protocol (Dummy cycle Enable) 0: no dummy cycle is inserted 1: insert a dummy cycle after the read command
14:12	-	RFU: **Reserved, read as 0**
11	**RXO**	RXONLY control bit, when this register is set, the SPI can receive continuously without software writing TXBUF (Receive Only mode) 1: Enable the single receive mode of Master 0: turn off the single receive mode (send/receive full duplex)
10:9	**DLEN**	Communication Data Length 00:8bit 01:16bit 10:24bit 11:32bit
8	**HALFDUPLEX**	Communication mode selection (Half-Duplex mode) 0: Standard SPI mode, 4-wire full duplex 1: DCN mode, 4-wire half-duplex
7	**HD_RW**	Read/Write config for Half-Duplex mode 0: Master write to slave in 4-wire half-duplex protocol 1: Master read slave in 4-wire half-duplex protocol
6	**CMD8b**	Define the command frame length in half duplex mode (Command 8 bits) 1: Command frame length is fixed to 8 bits 0: command frame length is defined by DLEN
5	**SSNM**	SSN control mode selection in Master mode (SSN mode) 1: Master pull SSN high after each frame is sent, the time to maintain high level is controlled by WAIT register 0: Master keeps SSN low after each frame is sent
4	**TXO_AC**	TXONLY auto-clear enable 1: TXONLY hardware auto-clear is valid, after the software enables TXO, wait for the hardware to clear after the transmission is finished 0: TXONLY hardware auto-clear enable is disabled
3	**TXO**	TXONLY control bit (Transmit Only mode enable) 1: Enable the Master's Transmit Only mode 0: disable single transmit mode (send and receive full duplex)
2	**SSN**	In Master mode, if SSNSEN is 1, software can control SSN output level by this bit 1: SSN output high level 0: SSN output low level
1	**SSNSEN**	Master mode, software control SSN enable (SSN Software Enable) 1: SSN output in Master mode is controlled by software 0: SSN output in Master mode is automatically controlled by hardware

bit	name	functional description
0	**SPIEN**	SPI enable 1: Enable SPI 0: Turn off SPI and clear the transmit/receive cache

23.6.3 SPIx Control Register 3 (SPIx_CR3)

NAME	SPIx_CR3(x=1,2)							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				TXBFC	RXBFC	MERRC	SERRC
access	U-0				W-0	W-0	W-0	W-0

bit	name	functional description
31:4	**-**	RFU: **Reserved, read as 0**
3	**TXBFC**	Transmit Buffer Clear, software write 1 to clear transmit buffer
2	**RXBFC**	Receive Buffer Clear, software write 1 to clear receive buffer
1	**MERRC**	Master Error Clear, software write 1 to clear MERR
0	**SERRC**	Slave Error Clear, software write 1 to clear SERR

23.6.4 SPIx Interrupt Enable Register (SPIx_IER)

NAME	SPIx_IER(x=1,2)							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					ERRIE	TXIE	RXIE
access	U-0					R/W-0	R/W-0	R/W-0

bit	name	functional description
31:3	**-**	RFU: **Reserved, read as 0**
2	**ERRIE**	SPI Error Interrupt Enable
1	**TXIE**	Transmit Interrupt Enable

bit	name	functional description
0	**RXIE**	Receive Interrupt Enable

23.6.5 SPIx Interrupt Status Register (SPIx_ISR)

NAME	SPIx_ISR(x=1,2)							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			DCN_TX	-	RXCOL	TXCOL	BUSY
access	U-0			R/W-1	U-0	R/W-0	R/W-0	R-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	MERR	SERR	-			TXBE	RXBF
access	U-0	R-0	R-0	U-0			R-1	R-0

bit	name	functional description
31:13	-	RFU: **Reserved, read as 0**
12	**DCN_TX**	In half-duplex mode (HALFDUPLEX=1), the DCN signal level is configured to be sent at the last bit of each data frame (Data/Command transmit config) 0: DCN=0 for command frames 1: DCN=1 for data frames The software should set DCN_TX register before transmitting. If DCN_TX=0, the hardware will automatically set DCN_TX to 1 after a frame is sent, i.e. only one command frame will be sent by default, and all subsequent frames will be data frames.
11	-	RFU: **Reserved, read as 0**
10	**RXCOL**	Receive Collision flag, write 1 to clear
9	**TXCOL**	Transmit Collision flag, write 1 to clear
8	**BUSY**	SPI idle flag, read-only (busy flag) 1: SPI transfer in progress 0: SPI transfer idle
7	-	RFU: **Reserved, read as 0**
6	**MERR**	Master Error flag (Master Error flag) MERR is set when the SSN is pulled high before the 8-bit transfer under the master
5	**SERR**	Slave Error flag SERR is set when the SSN is pulled high before 8 bits are transferred under the Slave
4:3	-	RFU: **Reserved, read as 0**
2	**TXSE**	TX shift register Empty flag bit (TX Shift register Empty flag) 1: Send shift register empty 0: Send shift register full
1	**TXBE**	TX Buffer Empty flag 1: Tx buffer empty, software write TXBUF to clear 0: Tx buffer full
0	**RXBF**	RX Buffer Full flag 1: Receive buffer full, software reads RXBUF to clear

bit	name	functional description
		0: Receive buffer empty

23.6.6 SPIx Transmit Buffer (SPIx_TXBUF)

NAME	SPIx_TXBUF(x=1,2)							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	TXBUF[31:24]							
access	W-0000 0000							
bit	Bit23	Bit23	Bit23	Bit23	Bit23	Bit23	Bit23	Bit23
name	TXBUF[23:16]							
access	W-0000 0000							
bit	Bit15	Bit15	Bit15	Bit15	Bit15	Bit15	Bit15	Bit15
name	TXBUF[15:8]							
access	W-0000 0000							
bit	Bit7	Bit7	Bit7	Bit7	Bit7	Bit7	Bit7	Bit7
name	TXBUF[7:0]							
access	W-0000 0000							

bit	name	functional description
31:0	**TXBUF**	Transmit Buffer

23.6.7 SPIx Receive Buffer (SPIx_RXBUF)

NAME	SPIx_RXBUF(x=1,2)							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	RXBUF[31:24]							
access	R-0000 0000							
bit	Bit23	Bit23	Bit23	Bit23	Bit23	Bit23	Bit23	Bit23
name	RXBUF[23:16]							
access	R-0000 0000							
bit	Bit15	Bit15	Bit15	Bit15	Bit15	Bit15	Bit15	Bit15
name	RXBUF[15:8]							
access	R-0000 0000							
bit	Bit7	Bit7	Bit7	Bit7	Bit7	Bit7	Bit7	Bit7
name	RXBUF[7:0]							
access	R-0000 0000							

bit	name	functional description
31:0	**RXBUF**	SPI Receive Buffer

24 ISO7816

24.1 Introduction

ISO7816 is a serial and synchronous communication interface to exchange 8-bit data over 2-wire with external integrated card. The chip implements two 7816 master interface function.

- 1-channel 7816-3 interface (mapped to two sets of GPIO)
- Card clock output port, configurable output frequency (1MHz~5MHz)
- configurable bit order (MSB First / LSB First)
- The error signal width can be configured as 1/1.5/2 ETU
- Supports error triggered re-transmission function, and the number of re-transmission times can be configured as 0~3 times
- EGT can be set to 0~256 with support for multiple timeout interrupts
- Data reception complete interrupt and error interrupts
- Configurable interrupt generation condition(buffer empty/ shift register empty)
- Support DMA interface

24.2 Block diagram

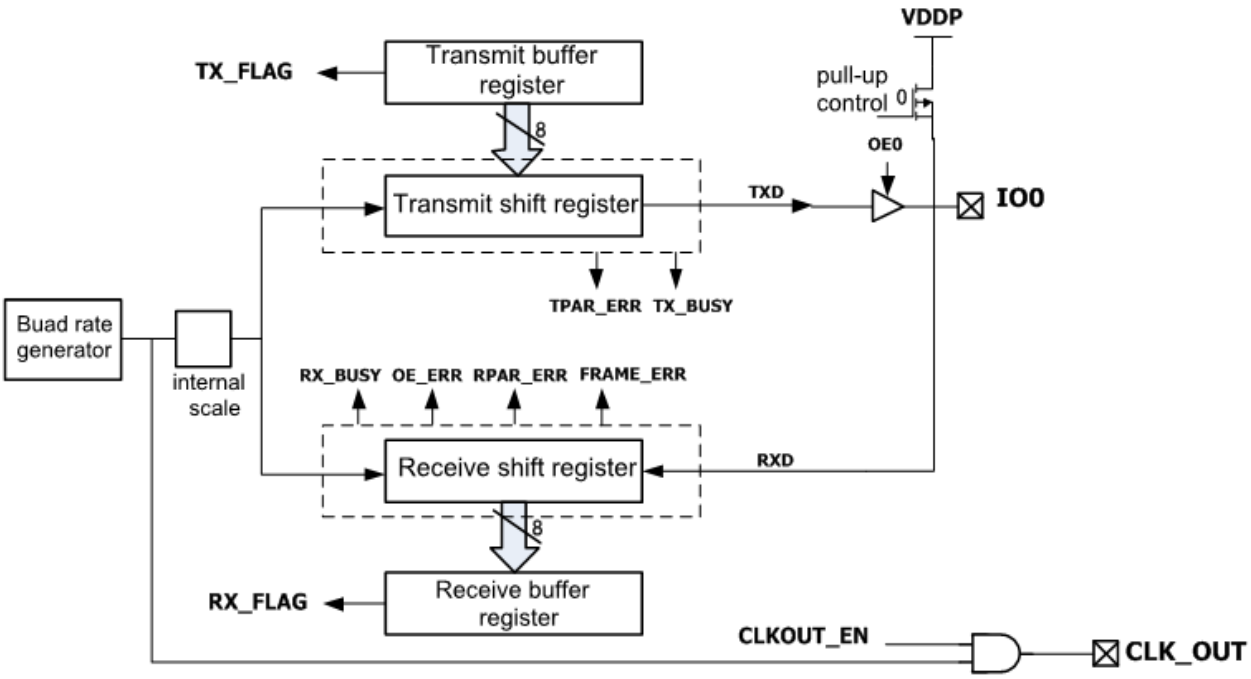


Figure 24-1 ISO7816 block diagram

24.3 Interface timing

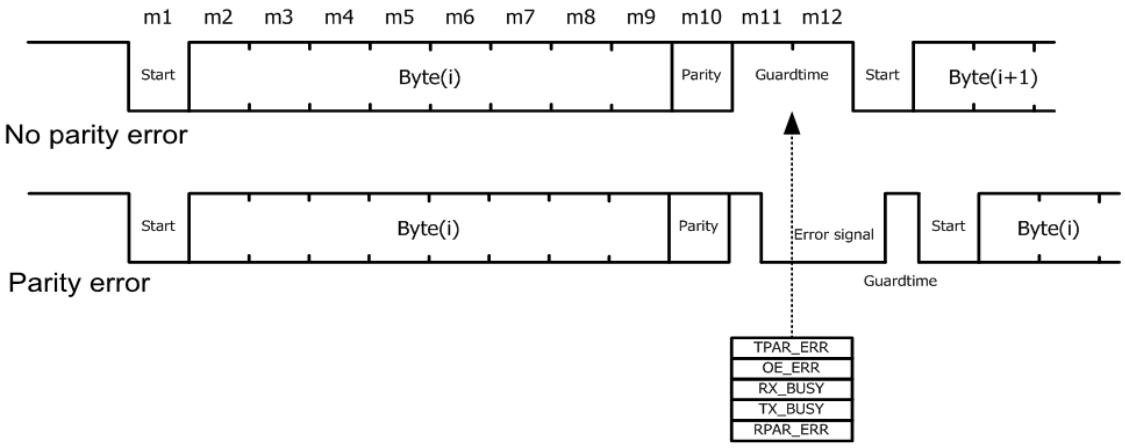


Figure 24-2 ISO7816 data frame structure

According to the protocol of ISO7816, the basic interface sequence of 7816 is as follows:

- A start bit followed by 8 data bits and 1 parity bit ends with GUARDTIME of 1ETU or 2ETU.
- The minimum single-byte data length is 11ETU or 12ETU.
- If the parity is correct, the GUARDTIME of two ETU is inserted to ensure that the data length is 12ETU. In the 11th ETU, RX_BUSY is invalid and possible OE_ERR flags are generated to complete the data transmission. If there is an ERROR in the receiving parity, IO is pulled down at 10.5ETU to produce an ERROR SIGNAL. The ERROR SIGNAL has a minimum of 1 ETU and a maximum of 2 ETU. The RPAR_ERR flag is generated at the 11th ETU as required.
- At the 11th ETU, if the ERROR SIGNAL is not detected, the transmitted data was correct. The data transmission is finished and TX_BUSY is cleared.
- If ERROR SIGNAL is detected at the 11th ETU, it means the data is not correctly received, and the TPAR_ERR flag is generated and data is re-transmitted after waiting for 2 ETU.

24.4 Functional description

24.4.1 Data receive

7816 data receive flow:

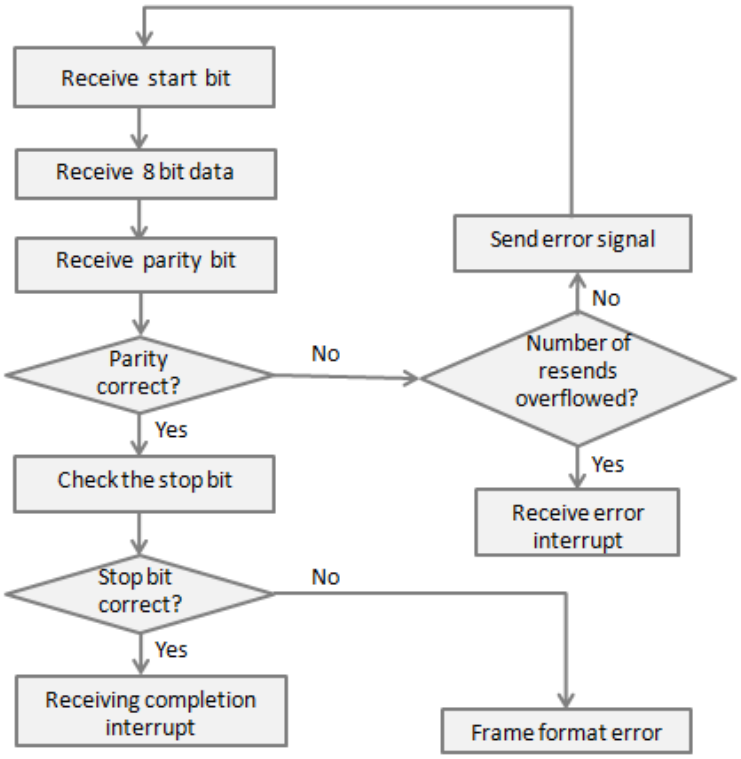


Figure 24-3 ISO7816 data receive process

24.4.2 Data transmission

When TXEN is set, the software can write data to TXBUF to trigger hardware transmission. The software can write data to TXBUF during the transmission process, and the hardware will continue transmission as long as TXBUF is not empty. It should be noted that, since there is only a one buffer in this design, the interval between two TXBUF writes by the software cannot be too short. If TXBUF is written again when previous data has not been loaded to shift register, the previous data will be overwritten. Software can monitor TX_FLAG, TX_FLAG = 1 means the TX buffer register is empty, the data has been moved into the shift register, software can write the next data to TXBUF.

7816 Data transmission process:

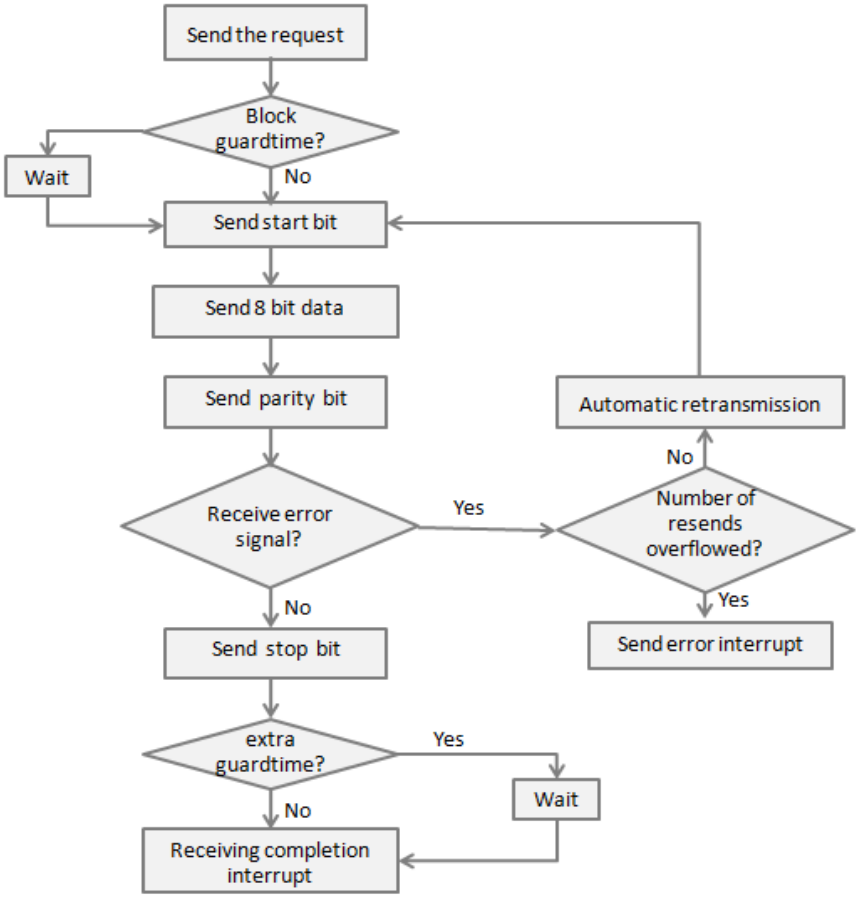


Figure 24-4 ISO7816 data transmission process

24.4.3 DMA control 7816 for sending and receiving data

When the 7816 module is enabled, the 7816 module will automatically generate the corresponding DMA request when the TX buffer is empty or the RX buffer is full. The application needs to configure DMA channel connections, connect specific channels to 7816 peripherals, set the RAM address pointer, and enable DMA channels. The DMA will then automatically respond to the 7816 request and complete the data transfer between RAM and 7816.

Application：Use DMA to control 7816 for sending and receiving data

- Configure DMA channel 5 as U7816_RX
- Set the RAM pointer address, address increment or decrement, channel priority, transmission length, interrupt, and so on
- Enable the corresponding DMA channel
- Configure the 7816 module parameters
- Enable 7816 module, waiting for data reception

- 7816 automatically generates DMA request after receiving data
- Configure DMA channel 0 to U7816_RX
- DMA responds to the request, reads the 7816 receive buffer, and writes to the specified RAM address

24.5 Register

offset	name	symbol
ISO7816(base adress:0x40010000)		
0x00000000	U7816 Control Register	U7816_CR
0x00000004	U7816 Frame Format Register	U7816_FFR
0x00000008	U7816 Extra Guard Time Register	U7816_EGTR
0x0000000C	U7816 Prescaler Register	U7816_PSC
0x00000010	U7816 Baud rate Generator Register	U7816_BGR
0x00000014	U7816 Receive Buffer	U7816_RXBUF
0x00000018	U7816 Transmit Buffer	U7816_TXBUF
0x0000001C	U7816 Interrupt Enable Register	U7816_IER
0x00000020	U7816 Interrupt Status Register	U7816_ISR

24.5.1 U7816 Control Register (U7816_CR)

NAME	U7816_CR							
offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
property	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-		TXEN	RXEN	CKOEN	-		
property	U-0		R/W-0	R/W-0	R/W-0	U-0		

bit	name	functional description
31:6	-	RFU：Reserved, read as 0
5	TXEN	U7816 channel Transmit Enable Bit (Transmit Enable) 1: Channel TX is enabled 0: Channel TX is disabled
4	RXEN	U7816 channel Receive Enable bit 1: Channel receive enabled 0: Channel receive disabled
3	CKOEN	U7816 Clock CLK output Enable bit (Clock output Enable) 1: 7816 clock output enable 0: 7816 clock output disable
2:0	-	reserved

24.5.2 U7816 Frame Format Register（U7816_FFR）

NAME	U7816_FFR							
offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				SFREN	ERSW		ERSGD
property	U-0				R/W-0	R/W-00		R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	BGTEN	REP_T	PAR		RFREN	TREPEN	RREPEN	DICONV
property	R/W-0	R/W-0	R/W-00		R/W-0	R/W-1	R/W-1	R/W-0

bit	name	functional description
31:12	-	RFU: Reserved, read as 0
11	SFREN	Guard Time config bit 1: Guard Time is 3 ETU 0: Guard Time is 2 ETU
10:9	ERSW	ERROR SIGNAL length selection 11: The length of ERROR SIGNAL is 1ETU; 10: The length of ERROR SIGNAL is 1.5ETU; 01: The length of ERROR SIGNAL is 2ETU; 00: The length of ERROR SIGNAL is 2ETU;
8	ERSGD	GUARDTIME length after ERROR SIGNAL (valid only when sending) (Error Signal Guard Time) 1: GUARDTIME after ERROR SIGNAL is 1~1.5ETU. 0: GUARDTIME after ERROR SIGNAL is 2~2.5ETU. When the length of ERROR SIGNAL is an integer ETU, GUARDTIME is 1.5 or 2.5ETU; When the length of ERROR SIGNAL is 1.5ETU, GUARDTIME is 1 or 2ETU
7	BGTEN	BGT control bit. Whether BGT is inserted between receiving and sending. BGT is the minimum time required between receiving and sending (Block Guard Time enable) 1: BGT enable, insert Block Guard Time (12 ETU); 0: BGT disable, Block Guard Time (12 ETU) is not inserted;
6	REP_T	Number of automatic retransmission time when receiving data parity error (Repeated Times) 1: 3 times 0: 1 time
5:4	PAR	Parity type selection 00: Even 01: Odd 10: Always 1 11: no parity

bit	name	functional description
3	RFREN	Receive Guard Time config 1: Guard Time is 1 ETU 0: Guard Time is 2 ETU
2	TREPEN	Transmit Repeat Enable 1: auto re-transmission enabled on error signal. Tx_perity_err flag is set when re-transmission time exceeds REP_T value 0: no re-transmission enabled. Tx_perity_err flag is set when error signal is detected.
1	RREPEN	Receiving Repeat Enable 1: reply error signal on parity error. When single byte receive has been repeated for more than REP_T times, rx_parity_err flag is set 0: no error signal reply on parity error. rx_parity_err flag is set
0	DICONV	Transfer order, bit Direction Conversion 1: Reverse coding, sending and receiving MSB first; Reverse logic level 0: Forward coding, sending and receiving LSB first; Positive logic level

24.5.3 U7816 Extra Guard Time Register（U7816_EGTR）

NAME	U7816_EGTR							
offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
property	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TXEGT							
property	R/W-0000 0000							

bit	name	functional description
31:8	-	RFU: Reserved, read as 0
7:0	TXEGT	Transmit Extra Guard Time

24.5.4 U7816 Prescaler Register（U7816_PSC）

NAME	U7816_PSC							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							

property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
property	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-			CLKDIV				
property	U-0			R/W-0 0011				

bit	name	functional description
31:5	-	RFU：Reserved, read as 0
4:0	CLKDIV	U7816 Clock output Divider $F_{7816}=F_{APBCLK}/(CLKDIV+1)$ Special case: when CLK_DIV is set to 0 or 1,$F_{7816}=F_{APBCLK}/2$ Note: The working clock range specified in the 7816 protocol is 1~5MHz.

24.5.5 U7816 Baud rate Generator Register（U7816_BGR）

NAME	U7816_BGR							
offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				PDIV[11:8]			
property	U-0				R/W-0001			
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	PDIV[7:0]							
property	R/W-0111 0011							

bit	name	functional description
31:12	-	RFU：Reserved, read as 0
11:0	PDIV	U7816 Pre-divider control register controlling the Divider of 7816 communications (baud rate) Baud ＝ F_{7816}/(PDIV ＋ 1) *Note: The minimum available value for PDIV is 0x1, and the configuration 0x0 is disabled*

24.5.6 U7816 Receive Buffer（U7816_RXBUF）

NAME	U7816_RXBUF							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
property	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RXBUF							
property	R-0000 0000							

bit	name	functional description
31:8	-	RFU：Reserved, read as 0
7:0	RXBUF	U7816 Data Receive Buffer

24.5.7 U7816 Transmit Buffer（U7816_TXBUF）

NAME	U7816_TXBUF							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
property	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TXBUF							
property	W-0000 0000							

bit	name	functional description
31:8	-	RFU：Reserved, read as 0
7:0	TXBUF	U7816 Data Transmitbuffer

24.5.8 U7816 Interrupt Enable Register（U7816_IER）

	U7816_IER							
Offset	0x0000001C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
property	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					RXIE	TXIE	LSIE
property	U-0					R/W-0	R/W-0	R/W-0

bit	name	functional description
31:3	-	RFU：Reserved, read as 0
2	RXIE	Data receiving interrupt enabled bit. RXIF Interrupt flag bit (Receive Interrupt Enable) 1: A receive completion interrupt occurs when the RXIF register is set 0: Abort receiving completion
1	TXIE	Data transmit interrupt enable bit. Corresponding TXIF Interrupt flag bit 1: When the TXIF register setting produces the interrupt that sends the completion 0: Interrupt to disable sending completion
0	LSIE	Interruption of line state enabled bit.Line Status Interrupt Enable (ERRIF) 1: A line error interrupt occurred while the ERRIF register was set 0: Disallow line error

24.5.9 U7816 Interrupt Status Register（U7816_ISR）

NAME	U7816_ISR							
Offset	0x00000020							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-					WAIT_RPT	TXBUSY	RXBUSY
property	U-0					R-0	R-0	R-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				TPARER	RPARER	FRERR	OVERR

					R	R		
property	U-0				R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					RXIF	TXIF	ERRIF
property	U-0					R-0	R-1	R-0

bit	name	functional description
31:19	-	RFU：Reserved, read as 0
18	WAIT_RPT	The U7816 interface sends an error signal and is waiting for the data re-transmission.(Waiting for Repeat flag) Set by hardware, software read-only
17	TXBUSY	Transmission busy flag, Automatically reset after transmission complete 1: set when transmission starts, cleared by hardware at middle of STOP bit 0: Data transmission idle
16	RXBUSY	Receiving busy flag, Automatically reset after receiving complete 1: data receiving on-going 0: data receiving idle
15:12	-	RFU：Reserved, read as 0
11	TPARERR	Transmit Parity Error, hardware set, write 1 to clear
10	RPARERR	Receive Parity Error flag, hardware set, write 1 to clear
9	FRERR	Frame Error flag, hardware set, write 1 to clear 1: The frame format is incorrect, the length of the frame byte received is incorrect, or the frame bit or stop bit received is incorrect 0: No parity error while receiving data
8	OVERR	Receive Overflow Error, hardware set, write 1 to clear 1: The receive buffer register has not been read out while new data has been received. Previous data in the receive buffer register is overwritten 0: No overflow error
7:3	-	RFU：Reserved, read as 0
2	RXIF	Receive interrupt flag, hardware set, read data receive buffer to clear 1: 1byte data received, data receiving buffer full 0: No data received. Data receiving buffer is empty
1	TXIF	Transmit interrupt flag, After power-on reset, this flag is automatically set, indicating that the buffer is empty and data can be written. The flag is automatically cleared after the software writes data into TX buffer, and it is set again when data is moved to shift register 1: Tx buffer is empty 0: Tx buffer is not empty
0	ERRIF	Error interrupt flag The bit is ORed from TPARERR, RPARERR, FRERR, OVERR. The software clears the bit by clearing the error flag register above.

25 Direct Memory Access Controller (DMA)

25.1 Introduction

- 7-channel peripheral PDMA, support Peripherals<>RAM transfers
- channel memory MDMA, support Flash<>RAM transfer
- Peripheral DMA transfers are triggered by peripheral requests and do not affect CPU operation during DMA operation
- Peripheral channel maximum transmission length 8192 bytes (8KB), support byte/half-word/word transmission
- Flash->RAM channel maximum transfer length 8192 bytes, word transfer only
- Support Flash continuous programming (RAM->Flash), need to erase in advance, one time programming fixed to 256 bytes
- RAM pointer increment, decrement
- Half interrupt and full interrupt can be generated
- Channel priority configurable (4 levels of priority)

25.2 Principle of Operation

Peripheral DMA is a Peripheral<>RAM channel and uses peripheral request trigger for data transfer. Each peripheral channel can support peripheral->RAM or RAM->peripheral data transfer, and adaptively select byte/half-word/word transfer mode according to the target peripheral type. DMA, as Master, will initiate AHB transactions for data operation after receiving the request. The peripheral target address is automatically located according to the channel selection, and the RAM target address is located according to the register configuration.

Each channel can choose one from multiple peripherals as source or destination, while the software can set the channel priority. When two channels want to access RAM at the same time, the priority will determine who accesses first and the other channel will be suspended until the higher-priority channel has finished transaction.

Peripheral request can be ready to send (RAM/Flash->Peripheral) or receive complete (Peripheral->RAM), data transfer is done through AHB bus. If DMA and CPU access certain peripheral at the same time, which master wins the arbitration depends on the register setting in BusMatrix priority. It should be noted that since most peripherals are connected to the APB bus, the APB bridge is mapped to the AHB as single slave. So when the DMA accesses any peripheral in the APB, bus arbitration will happen even if CPU accesses other peripherals under the APB. The DIR register allows you to configure the transfer direction of each channel, and the software must ensure that the transfer direction configuration is consistent with the actual peripheral requests mounted to this channel. For example, if the current peripheral request mounted to channel 1 is UART0 receive, the DIR register must be configured to 0 (data is read from the peripheral and written to RAM). Every time UART0 finishes receiving a frame of data, it will send RXD0 request to DMA, and DMA will read data from UART0 receive buffer register after responding to the request. If DIR is incorrectly configured to 1, the write operation of UART0 receive buffer register by DMA will be ignored by UART0.

The software can set the memory pointer of DMA, which is used to configure the RAM start address of DMA transfer, and RAM pointer can be chosen to be incremental or decremental. There is another TRFLEN register to configure the number of transfers. According to the starting address and the number of transfers, the end address is calculated. The transfer ends and the channel is disabled when TRFLEN transfers have been finished.

When the channel is enabled, the DMA is ready to accept requests for the peripheral selected by the channel. When half of the configured transfer length is achieved, a HTIF (Half transfer interrupt flag) interrupt is set; when the configured transfer length is fully completed, a TCIF (Transfer complete interrupt flag) interrupt is set. All the above interrupts can be masked by the corresponding interrupt enable registers.

The software can turn off the channel enable at any time before a complete transfer block of the DMA is completed, at which point the DMA will be suspended, and if the software then re-enables the channel, the DMA will continue to perform the previously suspended operation.

25.3 Block Diagram

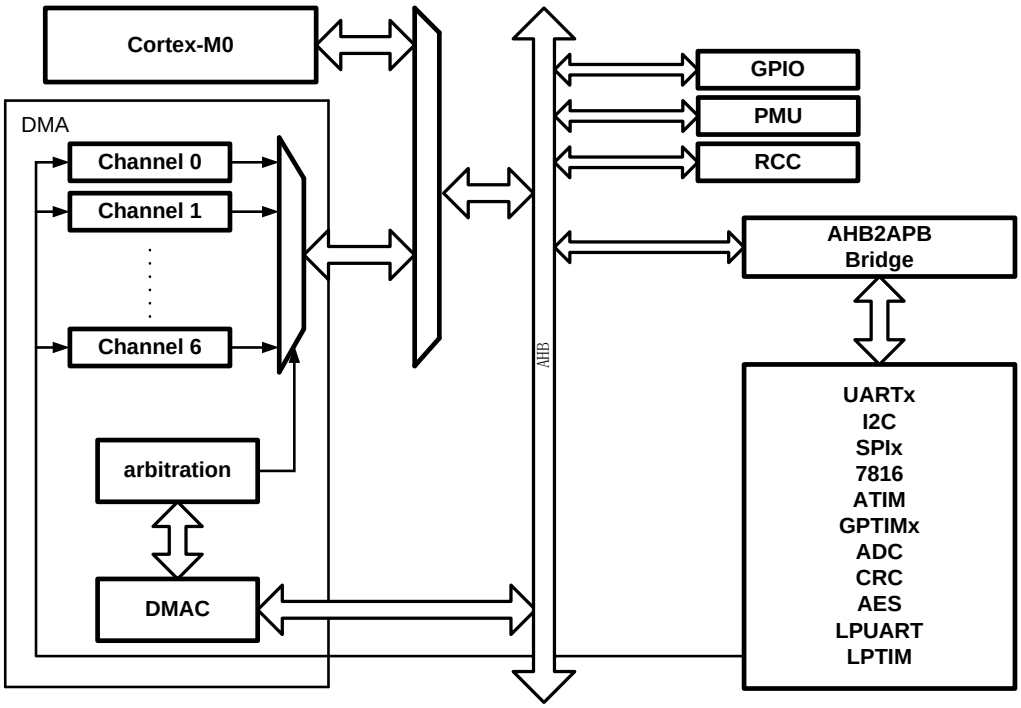


Figure 25-1 DMA block diagram

25.4 Workflow

DMA register configuration:

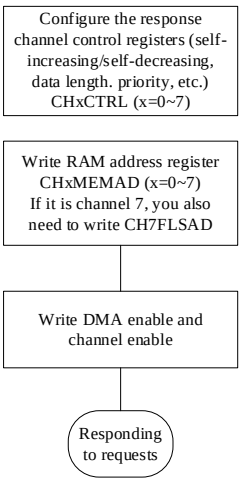


Figure 25-2 DMA register configuration

Channel request processing

The DMA processes the request response in two parts: the channel request processing and the data handling process.

- Channel request processing

a) DMA accepts the request, skip to step b

b) Determining if there are other channels carrying the data, if so, stay at step b until the other channels have completed the current carry; if not, further determine if there are other simultaneous requests, if so, determine if the current channel has a higher priority than the other channels, if so, skip to step c and initiate a request to the data carrying process, if not, stay at step b until the other channels have completed the current carry.

c) And waiting for the data handling completion response signal, if you get a response, skip to step d, otherwise stop at step c.

d) Data carry length +1, determine if the set length is reached, if yes, generate channel enable close pulse; determine if the request is released, if yes, skip to step a, if not, stay at step d to determine if the data transfer reaches the set length, otherwise skip to step a

- Data handling

a) Waiting for the channel request processing to initiate the request

b) Writing the source address to the HADDR

c) Writing destination address to HADDR and read HRDATA at the same time

d) Writing the read HRDATA data to the HWDATA

e) Sending a carry completion response to the channel request handler and skip to step a

The work flow is shown in the following diagram

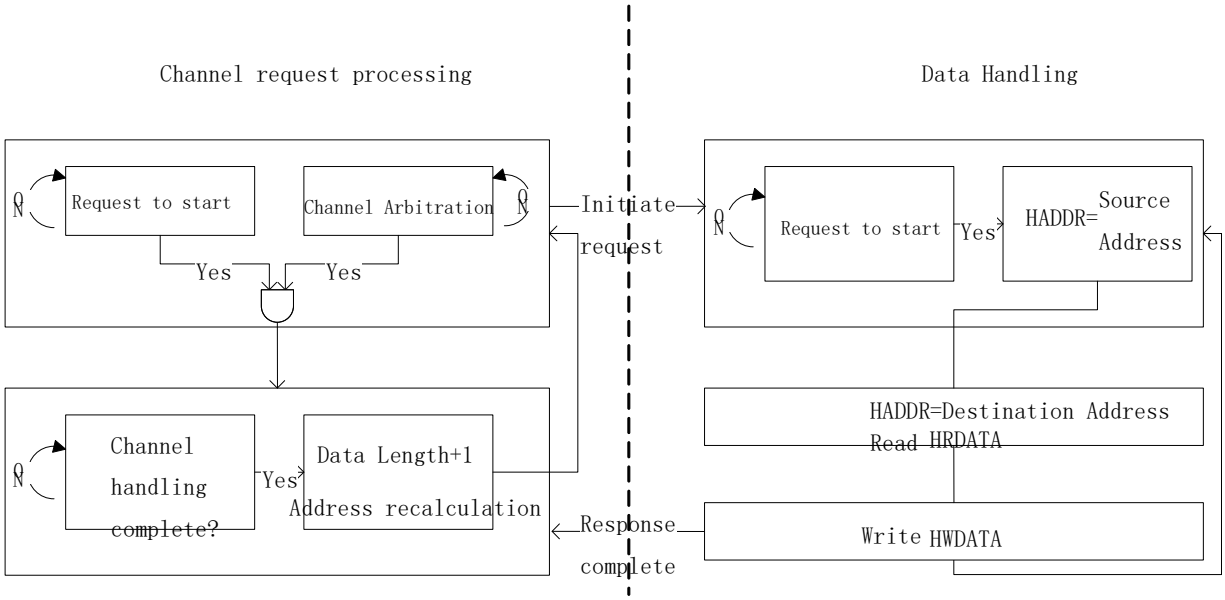


Figure 25-3 DMA workflow

25.5 Access Bandwidth

The DMA peripheral channels support byte/half-word/word access, and transfer bandwidth for each channel is defined by BDW register.

25.6 Channel Control

25.6.1 DMA Request Mapping

DMA has 7 peripheral channels with assignable priority, each channel can accept 8 request responses, and one of the requests is sent to the channel controller according to the configuration register of each channel, and the channel controller selects one of the channel requests according to the busy status and priority of each channel. Peripheral requests are mapped as follows.

Number	Peripherals Request	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6
0	***ADC***	ADC				ADC		
1	***SPI1***				SPI1_RX	SPI1_TX	SPI1_RX	SPI1_TX
2	***SPI2***			SPI2_RX		SPI2_TX	SPI2_RX	SPI2_TX
3	***UART0***		RXD0	TXD0	RXD0	TXD0		
4	***UART1***				RXD1	TXD1	RXD1	TXD1
5	***UART4***			RXD4	TXD4			
6	***UART5***					RXD5		TXD5
7	***LPUART0***	LPUART0_	LPUART0_		LPUART0_		LPUART0_	

Number	Peripherals Request	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6
		RX	TX		RX		TX	
8	***LPUART1***	LPUART1_TX		LPUART1_RX			LPUART1_RX	LPUART1_TX
9	***U7816***						U7816RX	U7816TX
10	***I2C***		I2C_RX	I2C_TX		I2C_RX		I2C_TX
11	***AES***	AES_IN	AES_OUT					
12	***CRC***	CRC						
13	***ATIM***	ATIM_CH1	ATIM_CH2	ATIM_CH3	ATIM_CH4	ATIM_TRIG ATIM_COM ATIM_UEV		
14	***GTIM1***	GTIM1_CH1	GTIM1_CH2	GTIM1_CH3	GTIM1_CH4			GTIM_TRIG GTIM1_UEV
15	***GTIM2***	GTIM2_CH1	GTIM2_CH2	GTIM2_CH3	GTIM2_CH4		GTIM2_TRIG GTIM2_UEV	
16	***LPTIM32***		LPT32_CH1				LPT32_CH2	
		8	8	8	8	8	8	8

Table 25-1 DMA channel mapping

Note, ATIM_ TRIG、ATIM_ COM and ATIM_ UEV requests are only for the DMA Burst mode of the advanced timer, that is, when these requests arrive, DMA will access the DMAR register of ATIM, so these three requests can be combined into one channel; Similarly, GTIMx_TRIG and GTMX_UEV of general Timer_ Trig requests can also be merged into one channel.

The peripheral request mapping is configured through the CHxSSEI register. From top to bottom, the above table represents the valid peripheral request signals when CHxSSEL = 0 ~ 7. For example, for channel 0, when CH0SSEL = 2, the selected peripheral request is EUART1_ TX, that is, the data transmission DMA request of EUART1 is connected to the request input of DMA channel 0.

25.6.2 Channel Priority

DMA has 7 peripheral channels in total, and the priority level of each channel can be configured through registers: very high, high, medium, low. When multiple channels are configured with the same priority level, the higher the channel number, the lower the priority level.

When channel 0 completes its second transfer and is ready for its third transfer, the channel 1 request response is set, and the channel controller switches to channel 1 data transfer according to the channel priority until all channel 1 data is transferred, and the channel register switches back to channel 0 to complete the remaining data transfer. Then the channel register switches back to

channel 0 to complete the rest of the data handling.

25.6.3 Definition of transfer direction

In the DMA channel definition rules_ RX indicates that DMA reads data from peripherals and writes it to ram_ TX indicates that DMA reads data from RAM / flash and writes it to peripherals.

After configuring the peripheral allocation of each channel, the software also needs to configure. The CHX_Dir register sets the channel transmission direction. The wrong direction setting will cause the DMA to fail to work normally.

25.6.4 Loop Mode

Peripheral DMA channel supports circular mode. In the loop mode, when the transfer length defined by the CHxTSIZE register is completed, the DMA will not automatically stop, but will roll-back to the starting address defined by the RAM pointer register and continue the transfer. DMA's half-range interrupt and full-range interrupt will still be set normally, DMA will not terminate the transmission until the software disables the channel.

The loop mode is enabled by setting the CHxCTRL.CIRC register.

The memory DMA channel does not support circular mode.

25.7 Register

Offset	Name	Symbol
DMA(based adress:0x40000400)		
0x00000000	DMA Global Control Register	DMA_GCR
0x00000004	Channel 0 Control Register	DMA_CH0CR
0x00000008	Channel 0 Memory Address Register	DMA_CH0MAD
0x0000000C	Channel 1 Control Register	DMA_CH1CR
0x00000010	Channel 1 Memory Address Register	DMA_CH1MAD
0x00000014	Channel 2 Control Register	DMA_CH2CR
0x00000018	Channel 2 Memory Address Register	DMA_CH2MAD
0x0000001C	Channel 3 Control Register	DMA_CH3CR
0x00000020	Channel 3 Memory Address Register	DMA_CH3MAD
0x00000024	Channel 4 Control Register	DMA_CH4CR
0x00000028	Channel 4 Memory Address Register	DMA_CH4MAD
0x0000002C	Channel 5 Control Register	DMA_CH5CR
0x00000030	Channel 5 Memory Address Register	DMA_CH5MAD
0x00000034	Channel 6 Control Register	DMA_CH6CR
0x00000038	Channel 6 Memory Address Register	DMA_CH6MAD
0x0000003C	Channel 7 Control Register	DMA_CH7CR
0x00000040	Channel 7 Flash Address Register	DMA_CH7FLSAD
0x00000044	Channel 7 RAM Address Register	DMA_CH7RAMAD
0x00000048	DMA Interrupt Status Register	DMA_ISR

25.7.1 DMA Global Control Register (DMA_GCR)

NAME	DMA_GCR							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
property	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						ADDRERR_EN	EN
property	U-0						R/W-0	R/W-0

bit	name	functional description
31:2	-	Reserved, read as 0
1	ADDRERR_EN	DMA error address interrupt enable 1：Error address interrupt enable 0：Error address interrupt disable
0	EN	DMA enable 1：DMA enable 0：DMA disable

25.7.2 Channel X Control Register (DMA_CHxCR)

NAME	DMA_CHxCR(x=0,1,2,3,4,5,6)							
offset	0x00000004 + x*0x08							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	TSIZE[15:8]							
property	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	TSIZE[7:0]							
property	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-		PRI		INC	SSEL		
property	U-0		R/W-00		R/W-0	R/W-000		
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	DIR	BDW		CIRC	FTIE	HTIE	EN
property	U-0	R/W-0	R/W-00		R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:16	TSIZE	Channelx transfer length，1-65336 transmissions
15:14	-	Reserved, read as 0
13:12	PRI	Channelx priority 00：Low 01：Medium 10：High 11：Very High
11	INC	RAM address increase and decrease settings 1：RAM address Increment 0：RAM address decrement
10:8	SSEL	Channelx peripheral request mapping Each channel can accept 8 peripheral requests. For the mapping of peripheral requests, see 24.6.1 DMA request mapping
7	-	Reserved, read as 0

bit	name	functional description
6	**DIR**	Channel transmission direction 0：Read data from peripheral to RAM 1：Read data from RAM and write to peripheral
5:4	**BDW**	Peripheral size 00：8bit 01：16bit 10：32bit 11：RFU
3	**CIRC**	Circular mode 0：Disable Circular mode 1：Enable Circular mode
2	**FTIE**	Channelx Transfer complete interrupt enable 1：Transfer complete interrupt enable 0：Transfer complete interrupt disable
1	**HTIE**	Channelx Half transfer complete interrupt enable 1：Half transfer complete interrupt enable 0：Half transfer complete interrupt disable
0	**EN**	Channelx enable 1：Enable channel 0：Disable channel

25.7.3 Channel X Memory Address Register (DMA_CHxMAD)

NAME	DMA_CHxMAD(x=0,1,2,3,4,5,6)							
Offset	0x00000008 + x*0x08							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	MEMAD[31:24]							
property	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	MEMAD[23:16]							
property	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	MEMAD[15:8]							
property	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	MEMAD[7:0]							
property	R/W-0000 0000							

bit	name	functional description
31:0	**MEMAD**	Channelx memory pointer address, the software writes the memory target address to this register before the DMA transfer is started.

bit	name	functional description
		DMA access will trigger a hardfault when the pointer points to a null address When the pointer points to Flash, writing data to Flash is prohibited. The software can query the destination memory address of the current DMA transfer.

25.7.4 Channel & Control Register (DMA_CH7CR)

NAME	DMA_CH7CR							
Offset	0x0000003C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-				TSIZE[11:8]			
property	U-0				R/W-0000			
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	TSIZE[7:0]							
property	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-		PRI		-	DIR	RI	FI
property	U-0		R/W-00		U-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					FTIE	HTIE	EN
property	U-0					R/W-0	R/W-0	R/W-0

bit	name	functional description
31:28	-	Reserved, read as 0
27:16	**TSIZE**	Channel7 transfer length, 1-8192 transfers, only valid for Flash->RAM transfers, RAM->Flash transfers are fixed length 64 transfers
15:14	-	Reserved, read as 0
13:12	**PRI**	Channel7 priority 00：Low 01：Medium 10：High 11：Very High
11	-	Reserved, read as 0
10	**DIR**	Channel7 transmission direction 1：Flash->RAM transfer 0：RAM->Flash transfer
9	**RI**	Channel7 RAM address increment/decrement setting, valid only in Flash->RAM transfer 1：RAM address Increment

bit	name	functional description
		0：RAM address decrement
8	**FI**	Channel7 Flash address increment/decrement setting, valid only in Flash->RAM transfer 1：Flash address Increment 0：Flash address decrement
7:3	**-**	Reserved, read as 0
2	**FTIE**	Channel7 Transfer complete interrupt enable Transfer complete interrupt enable 1：Transfer complete interrupt enable 0：Transfer complete interrupt disable
1	**HTIE**	Channel7 Half transfer complete interrupt enable 1：Half transfer complete interrupt enable 0：Half transfer complete interrupt disable
0	**EN**	Channel7 enable 1：Enable channel 0：Disable channel

25.7.5 Channel 7 Flash Address Register (DMA_CH7FLSAD)

NAME	DMA_CH7FLSAD							
offset	0x00000040							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-	FLSAD[14:8]						
property	U-0	R/W-000 0000						
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	FLSAD[7:0]							
property	R/W-0000 0000							

bit	name	functional description
31:15	**-**	Reserved, read as 0
14:0	**FLSAD**	Channel7 Flash pointer address, the software writes Flash target address to this register before DMA transfer starts, after DMA starts, this register is incremental or decremental with DMA transfer Software can query the target Flash address of the current DMA transfer

bit	name	functional description
		The low bit of this register (bit5-0) is valid only in Flash->RAM transfer, and the half-sector starting address of Flash is aligned by default in RAM->Flash transfer.

25.7.6 Channel 7 RAM Address Register (DMA_CH7RAMAD)

NAME	DMA_CH7RAMAD							
offset	0x00000044							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				RAMAD[11:8]			
property	U-0				R/W-0000			
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RAMAD[7:0]							
property	R/W-0000 0000							

bit	name	functional description
31:12	-	Reserved, read as 0
11:0	**RAMAD**	Channel7 RAM word pointer address, the software writes the RAM target address (word address) to this register before the DMA transfer starts, after the DMA starts this register is incremental or decremental with the DMA transfer Software can query the current DMA transfer target RAM address

25.7.7 DMA Interrupt Status Register (DMA_ISR)

NAME	DMA_ISR							
Offset	0x00000048							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							ADDRERR
property	U-0							R/W-0

NAME	DMA_ISR							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CHFT[7:0]							
property	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CHHT[7:0]							
property	R/W-0000 0000							

bit	name	functional description
31:17	-	Reserved, read as 0
16	**ADDRERR**	DMA transfer address error flag, set when the memory pointer exceeds the legal address range of RAM and Flash
15:8	**CHFT**	DMA channel x transfer completion flag，This flag is set by hardware. It is cleared by software by writing 1 tothis bit. 1：Corresponding channel transmission completed 0：Corresponding channel transmission is not completed
7:0	**CHHT**	DMA channel x transfer halfway flag， This flag is set by hardware. It is cleared by software by writing 1 tothis bit.

26 Cyclic Redundancy Check Calculation Unit (CRC)

26.1 Introduction

Cyclic Redundancy Check is the most commonly used checksum method for computer and instrument data communication. The CRC calculation unit in FM33LE0xxA is a completely independent module. CRC calculation and verification can be carried out for data communication such as 7816, I2C, UART and SPI through software control.

In addition, CRC can also perform Flash content integrity verification. With DMA, you can compute the CRC result of program content in Flash in real time and generate an integrity signature that is stored with the program in Flash. By verifying this CRC signature, you can verify whether the Flash content is correct and complete.

- Uses fully programmable polynomial with programmable size(7,8,16,32bits)
- Programmable CRC initial value
- CRC fast algorithm: 8bit CRC operation was completed in 1 clock cycle, and 32bit CRC operation was completed in 4 clock cycles
- Supports automatic input/output data order adjustment (byte, half-word, or full-word)
- Supports XOR for output results

26.2 Operation Flow

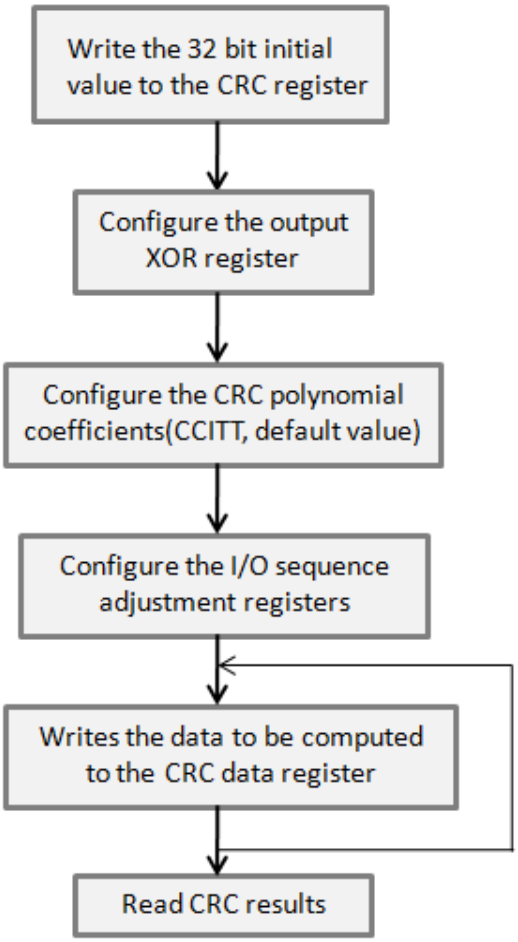


Figure 26-1 CRC operation flow

The CRC configuration and calculation procedure are as follows:

- Configure the initial value of the shift register with a range of 0x0000_0000~0xFFFF_FFFF.
- Configure the output XOR register CRC_XOR
- The software needs to configure input REFLECTIN enable, output REFLECTOUT enable and XOROUT enable.
- The software writes data into the data register (CRC_DR) and then automatically computes successive shifts.
- After the computation, the result is written back to the data register, and the software determines whether the result can be retrieved according to the BUSY bit
- If the polynomial is 7bit, the result is CRC_DR[6:0]; if the polynomial is 8bit, the result is CRC[7:0]; if the polynomial is 16bit, the result is CRC_DR[15:0]; if the polynomial is 32bit, the result is CRC_DR[31:0]; if the polynomial is 3bit, the result is CRC_DR[31:0].

- After the calculation of the previous CRC, the result of the previous CRC will be retained in the data register as the initial value of the shift register of the subsequent data. After the CRC calculation is triggered several times in a row, the software finally reads the CRC value of the cumulative calculated result.

26.3 Golden Data

Golden data are available for testing and validation

Polynomial	Input serial number	Initial value(hexadecimal)		
		ALL 0	ALL F	6363
		CRC result（hexadecimal）		
CRC-8	5A5A	0F	D8	C5
	1223344	F9	28	96
CRC-16	5A5A	5DD9	DDD4	9696
	11223344	7D35	7D11	4698
CRC-CCITT	5A5A	1ACB	07C4	1877
	11223344	DD33	59F3	DD06

26.4 DMA Interface

The channel between CRC and DMA is unidirectional (RAM->CRC).The CRC module can read and verify RAM data through the DMA module, and its workflow is shown in the figure.The CRC made a request to DMA, which received the request, read RAM and wrote the data to the CRCDR register of the CRC module. After receiving the data, the CRC module cancels the DMA request and starts to calculate the verification value. After the verification is completed, the CRC module resets the DMA request.

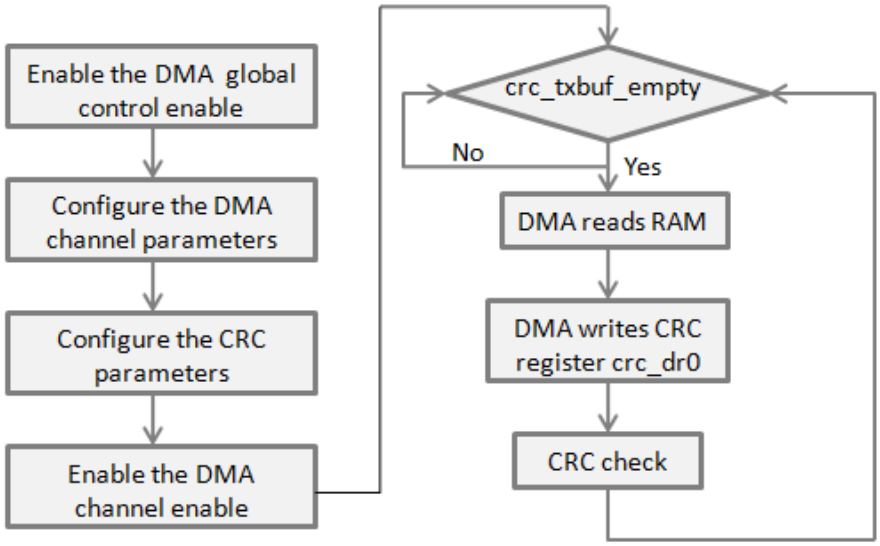


Figure 26-2 CRC the data in RAM by DMA

26.5 Flash Data Integrity Check

The integrity of flash content can be verified through CRC and DMA. The solution is:

- First, the Flash content is moved to RAM through DMA
- Move the Flash data cached in RAM to CRC module for calculation through DMA
- Repeat the above process until the verification of all Flash data is completed

26.6 Register

offset	name	symbol
CRC(base adress:0x40018000)		
0x00000000	CRC Data Register	CRC_DR
0x00000004	CRC Control Register	CRC_CR
0x00000008	CRC Linear Feedback Shift Register	CRC_LFSR
0x0000000C	CRC Output XOR Register	CRC_XOR
0x0000001C	CRC Polynomial Register	CRC_POLY

26.6.1 CRC Data Register (CRC_DR)

NAME	CRC_DR							
offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	DR[31:24]							
property	R/W-1111 1111							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	DR[23:16]							
property	R/W-1111 1111							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DR[15:8]							
property	R/W-1111 1111							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DR[7:0]							
property	R/W-1111 1111							

bit	name	functional description
31:0	**DR**	CRCDR is used as a data entry register and saves the CRC results after the operation. (CRC Data Register) When used as input: if word operation enabled, CRCDR[31:0] would be calculated, with a total of 4 byte operations (from low to high); Otherwise, the CRCDR[7:0] is computed, a total of 1 byte operation. When saving the results: if it is a 7-bit polynomial, it is stored in CRCDR[6:0]; if it is an 8-bit polynomial, it is stored in CRCDR[7:0]; if it is a 16-bit polynomial, it is stored in CRCDR[15:0]; if it is a 32-bit polynomial, it is stored in CRCDR[31:0].

26.6.2 CRC Control Register (CRC_CR)

NAME	CRC_CR							
offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
property	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
property	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						OPWD	PARA
property	U-0						R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RFLTIN		RFLTO	RES	BUSY	XOR	SEL	
property	R/W-00		R/W-0	R-0	R-0	R/W-0	R/W-10	

bit	name	functional description
31:10	-	RFU：**Reserved, read as 0**
9	**OPWD**	Operation by Word 0: Byte operation, CRC calculation only for CRCDR the lowest byte 1: Word operation, CRC calculation is performed for all 4 bytes of CRCDR
8	**PARA**	CRC Parallel Calculation 0: Serial operation. It takes 8 clock cycles to compute 1 byte 1: Parallel computing. It takes 1 clock cycle to compute 1 byte
7:6	**RFLTIN**	CRC Reflected Input 00：input unreflected 01：input reflected by byte 10：input reflected by half-word 11：input reflected by word For example, the calculated data is 0x11223344, If RFLTIN==01, the data is changed to 0x8844CC22 and then calculated If RFLTIN==10, the data is changed to 0x448822CC before calculation If RFLTIN==11, the data is changed to 0x22CC4488 and then calculated
5	**RFLTO**	CRC Reflected Output 0：output unreflected 1：output reflected by byte Such as:

bit	name	functional description
		If RFLTO==1, the output result is 0x2C48 if the currently computed CRC result is 0x1234 If RFLTO==0, output 0x1234 directly Note: This result is not the final output. It needs to see if XOR is 1. See bit2 for details
4	**RES**	CRC Result Flag, read only 0: The CRC result is 0 1: CRC results were not all 0
3	**BUSY**	CRC Busy Flag, read only 0: CRC operation ends 1: CRC operation is in progress
2	**XOR**	Output XORed with CRC_XOR register enable 0: Output no xor CRC_XOR register 1: Output xor CRC_XOR register
1:0	**SEL**	CRC Polynomial width Selection 00：32 bit 01：16 bit 10：8 bit 11：7 bit

26.6.3 CRC LFSR Register (CRC_LFSR)

NAME	CRC_LFSR							
offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	LFSR[31:24]							
property	R/W-1111 1111							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	LFSR[23:16]							
property	R/W-1111 1111							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	LFSR[15:8]							
property	R/W-1111 1111							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	LFSR[7:0]							
property	R/W-1111 1111							

bit	name	functional description
31:0	**LFSR**	CRC Linear Feedback Shift Register The CRC initial value can be written by the software before the operation begins

26.6.4 CRC Output XOR Register (CRC_XOR)

NAME	CRC_XOR							
offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	XOR[31:24]							
property	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	XOR[23:16]							
property	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	XOR[15:8]							
property	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	XOR[7:0]							
property	R/W-0000 0000							

bit	name	functional description
31:0	**XOR**	CRC exclusive XOR register When CRC_CR.xor is 1, the CRC result will XOR the register data before output.

26.6.5 CRC Polynomial Register (CRC_POLY)

NAME	CRC_POLY							
offset	0x0000001C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	POLY[31:24]							
property	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	POLY[23:16]							
property	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	POLY[15:8]							
property	R/W-0001 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	POLY[7:0]							
property	R/W-0010 0001							

bit	name	functional description
31:0	POLY	CRC Polynomials

27 Advanced Timer Array (ATIM)

27.1 Function description

The FM33LE0xxA contains an advanced timer.

The advanced timer includes a 16bit automatic overload counter and a programmable prescaler.

Advanced timers can support a variety of applications, including capture, output comparison, PWM, complementary PWM with dead band insertion.

27.2 Main characteristics

- 16bit up, down, two-way automatic reload counter
- 16bit programmable prescaler, support real-time adjustment of counting clock frequency division
- 4 independent channels can be used for input capture, output comparison, PWM, single pulse output
- Complementary output with programmable dead time insertion
- Independent working clock, the highest frequency is 120MHz
- Repeat counter, support timer to update status after multiple cycles
- Two brake pin input, comparator brake, SVD brake, brake signal filtering and polarity selection, brake signal combination configuration
- Support to generate interrupt or DMA event when the following events occur
  - Counter up/down overflow, counter initialization (software or hardware trigger)
  - Trigger event (counter start, stop, initialization, internal and external trigger)
  - Input capture
  - Output comparison
  - Brake input
- Support incremental quadrature encoder and Hall sensor

Terminology:

- When the OCxREF signal is high, it is called the effective level, and when it is low, it is called the invalid level.
- IDLE mode: Relative to running mode, MOE=0 when a braking event occurs
- Output disabled: GPIO output is enabled and closed, not driven by TIMER
- Off-state: GPIO output enable is turned on, but the TIMER output is in an invalid state
- Invalid state: the state when one or two of the complementary channels output an invalid level

27.3 Block Diagram

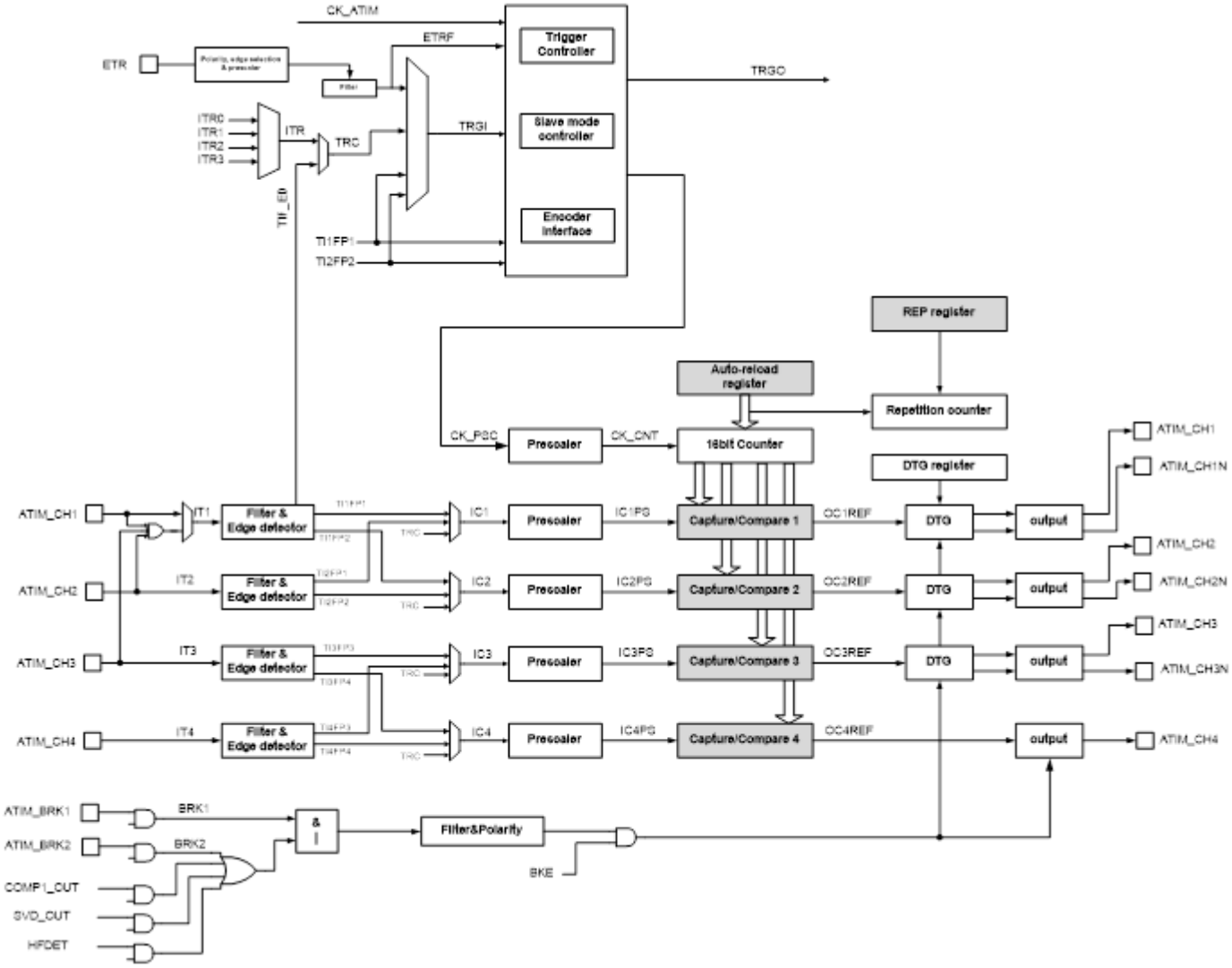


Figure 27-1 ATIM Block Diagram

27.4 Function description

27.4.1 Timing unit

The timing unit of the advanced timer consists of a 16-bit counter and auto-reload register. The counter can count up, down or bidirectionally. The count clock can be obtained after dividing the APBCLK by a 16-bit prescaler.

The counter, auto-reload register and prescaler register can all be rewritten or read by software, even when the counter is running.

The timing unit contains the following registers:

- Counter (ATIM_CNT)
- Prescaler register (ATIM_PSC)
- Automatic reload register (ATIM_ARR)
- Repeat count register (ATIM_RCR)

ARR includes a preload function, which is controlled by the ARPE (Auto Reload Preload Enable) register. When ARPE=0, the ARR register is written, and the written data will be directly transferred to the shadow register; when ARPE=1, the data written to the ARR register occurs in the update event (ATIM_CNT overflow or underflow) When, transfer to the shadow register. Software can also actively trigger ARR update (UEV) through register operations.

The ATIM_CNT working clock is driven by the frequency division clock generated by ATIM_PSC. The CNT only starts counting when the counter enable register (CEN) is set. When CNT=ARR, this round of counting ends, and an update event is sent.

ATIM_PSC is a synchronous prescaler that can divide APBCLK from 1 to 65536. The PSC register is also cached. Rewriting the PSC does not actually rewrite the shadow register. Only when a new update event comes will it be updated from the PSC to the shadow register. Therefore, during the CNT counting process, the software can rewrite the PSC in real time, and the new prescaler ratio will be adopted when the next update event occurs.

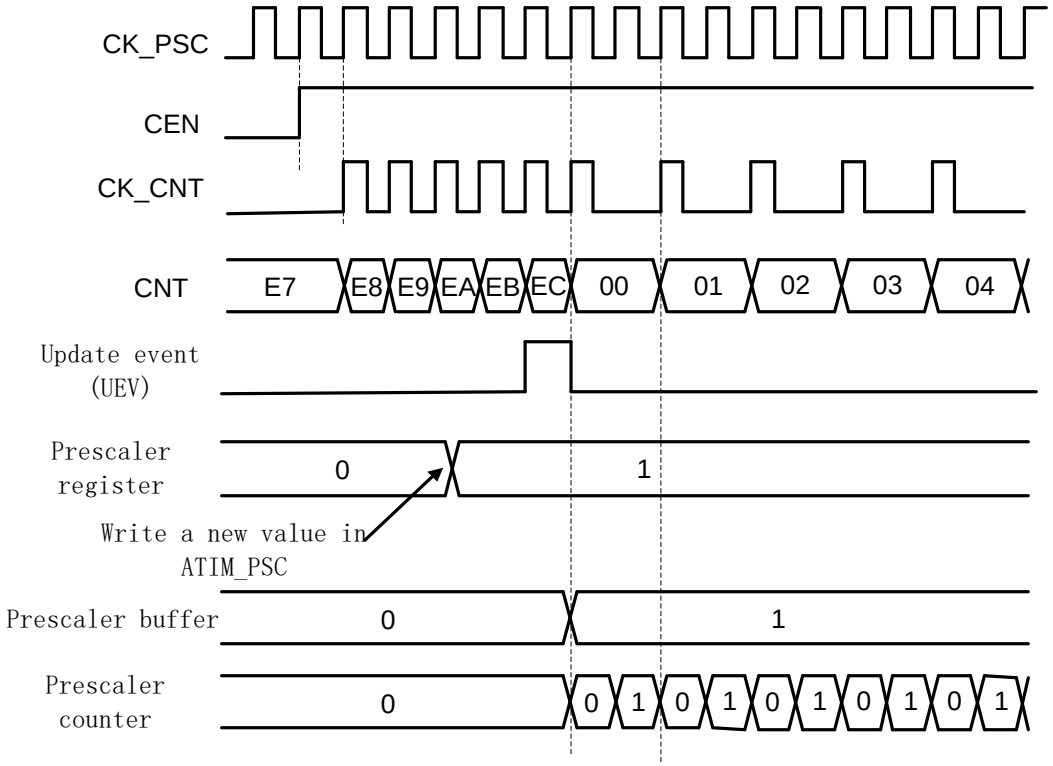


Figure 27-2 Prescaler waveform from 1 to 2

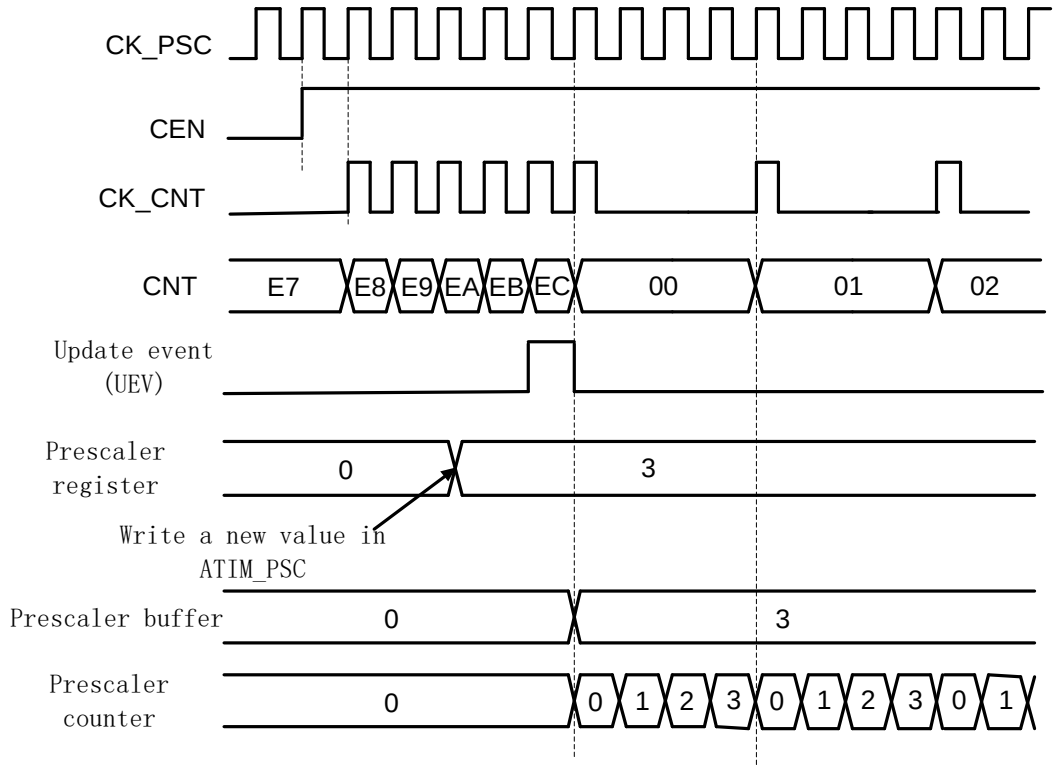


Figure 27-3 Waveform of prescaling from 1 to 4

27.4.2 Timer operating mode

The timer supports up-counting, down-counting and center-counting modes.

Count up

In this mode, the counter starts counting from 0 after being enabled, until CNT=ARR, an overflow event is generated, and then starts counting from 0 again.

If the repeat counting function is enabled, the counter repeats the above process several times (RCR+1) according to the definition of RCR before an overflow event will be generated.

The software can directly trigger the update event by setting the UG register, and the CNT and prescaler counter are automatically cleared at this time. Whether the setting of the UG register triggers the UIF (Update Interrupt Flag) interrupt flag setting is determined by the setting of the URS register.

The update event can be disabled by setting the UDIS register, which can avoid updating the value in the preload register to the working register.

When an update event occurs, the following registers are updated and UIF is set:

- The RCR shadow register is updated to the contents of ATIM_RCR
- The ARR shadow register is updated to ATIM_ARR content
- PSC shadow register is updated to ATIM_PSC content

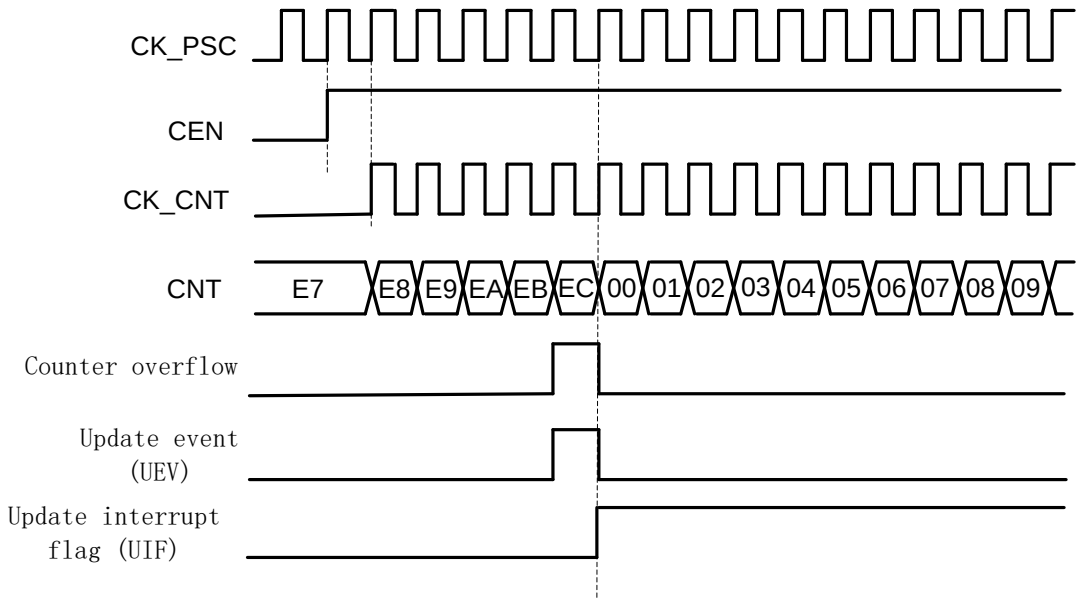


Figure 27-4 Upward counting waveform, internal clock not divided

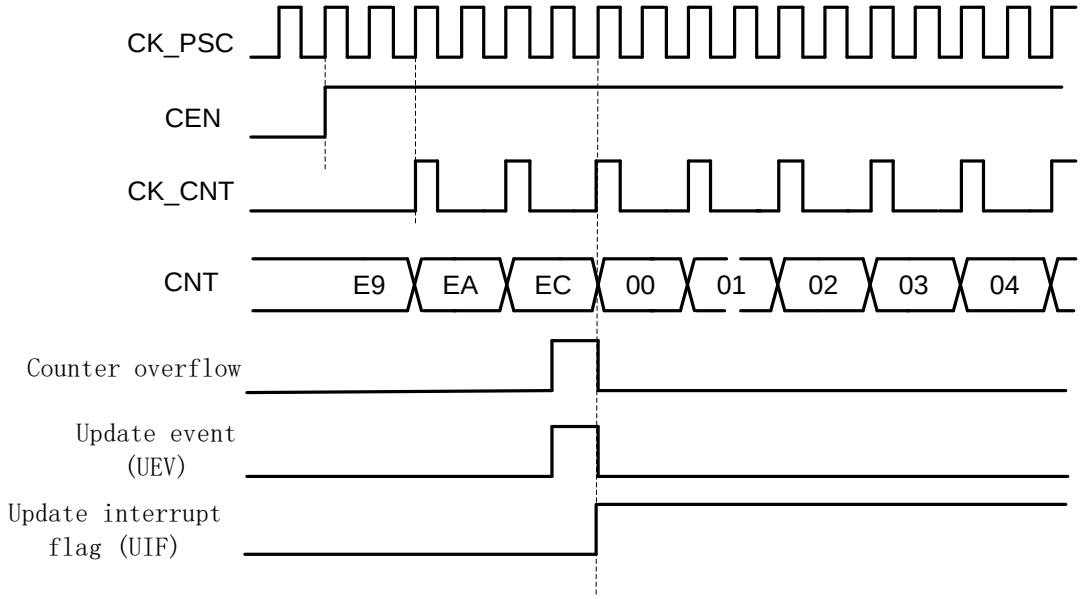


Figure 27-5 Upward counting waveform, internal clock divided-by-2

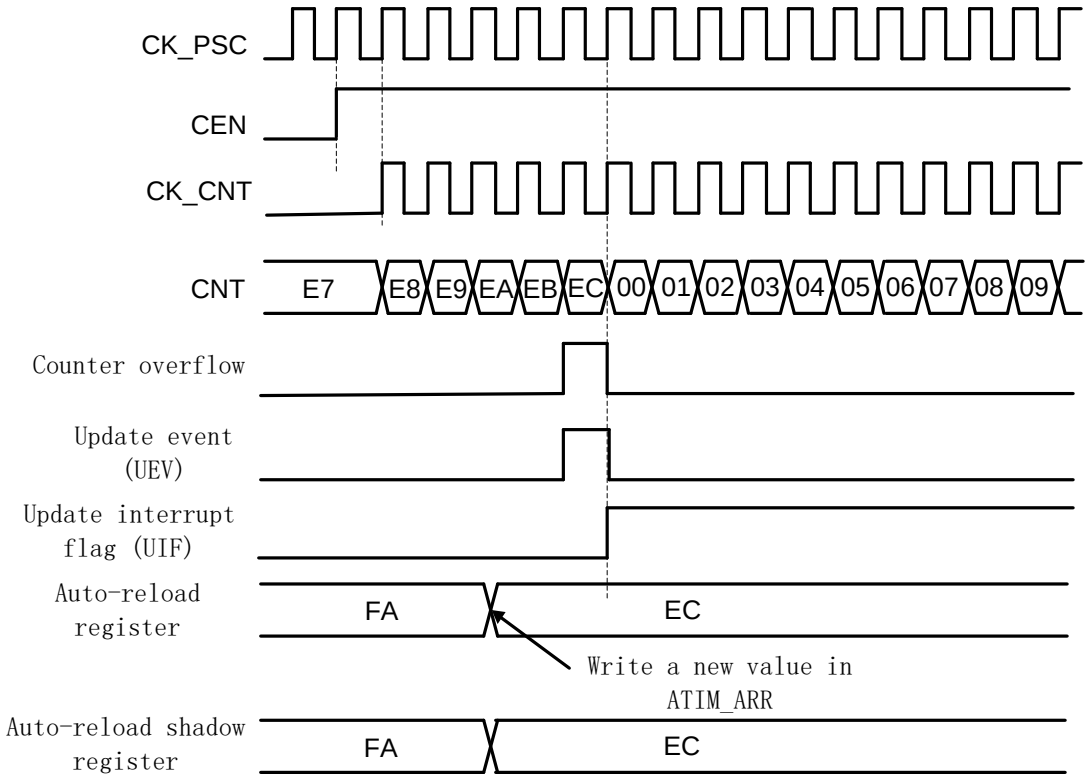


Figure 27-6 Update event when ARPE=0 (ARR is not preloaded)

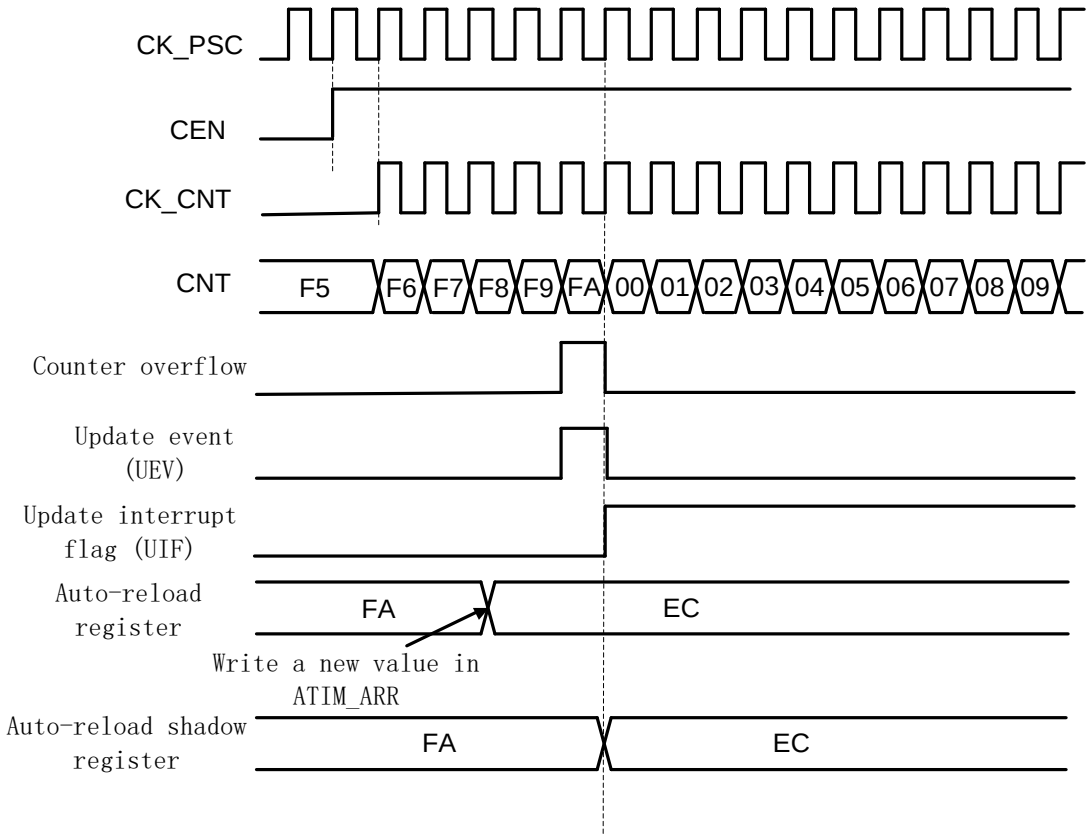


Figure 27-7 Update event when ARPE=1 (ARR preload)

Count down

In the down-counting mode, the counter starts to decrement from the ARR value, and when it reaches 0, an underflow event is generated, and the counter starts counting from ARR again.

If the repeat counting function is enabled, the counter repeats the above process several times (RCR+1) according to the definition of RCR before an overflow event will be generated.

The software can directly trigger the update event by setting the UG register, and the CNT and prescaler counter are automatically cleared at this time. Whether the setting of the UG register triggers the UIF (Update Interrupt Flag) interrupt flag setting is determined by the setting of the URS register.

The update event can be disabled by setting the UDIS register, which can avoid updating the value in the preload register to the working register.

When an update event occurs, the following registers are updated and UIF is set:

- The RCR shadow register is updated to the contents of ATIM_RCR

- The ARR shadow register is updated to ATIM_ARR content
- PSC shadow register is updated to ATIM_PSC content

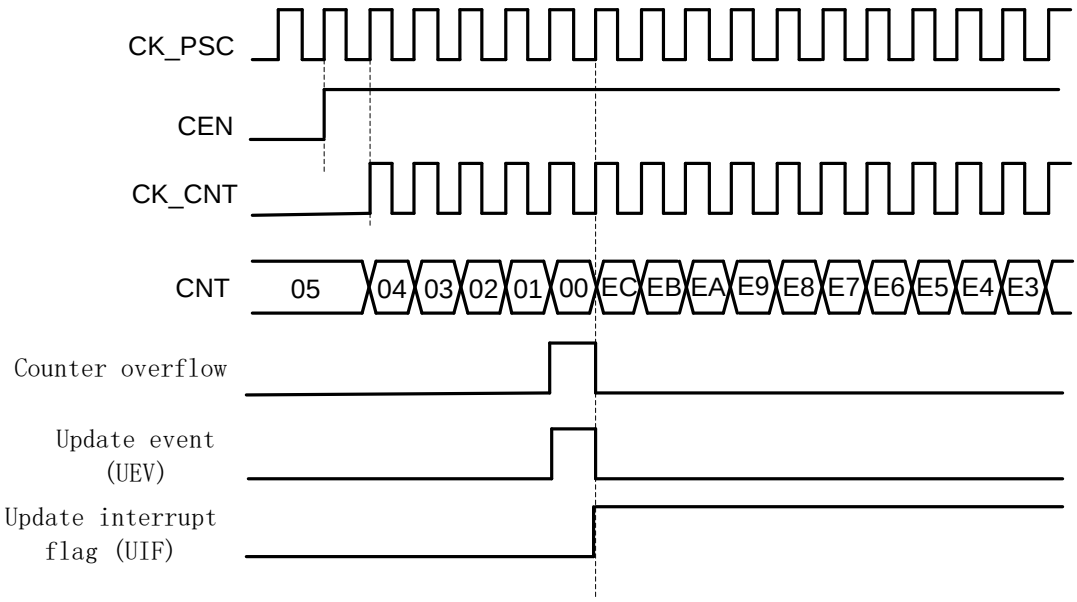


Figure 27-8 Downward counting waveform, internal clock not divided

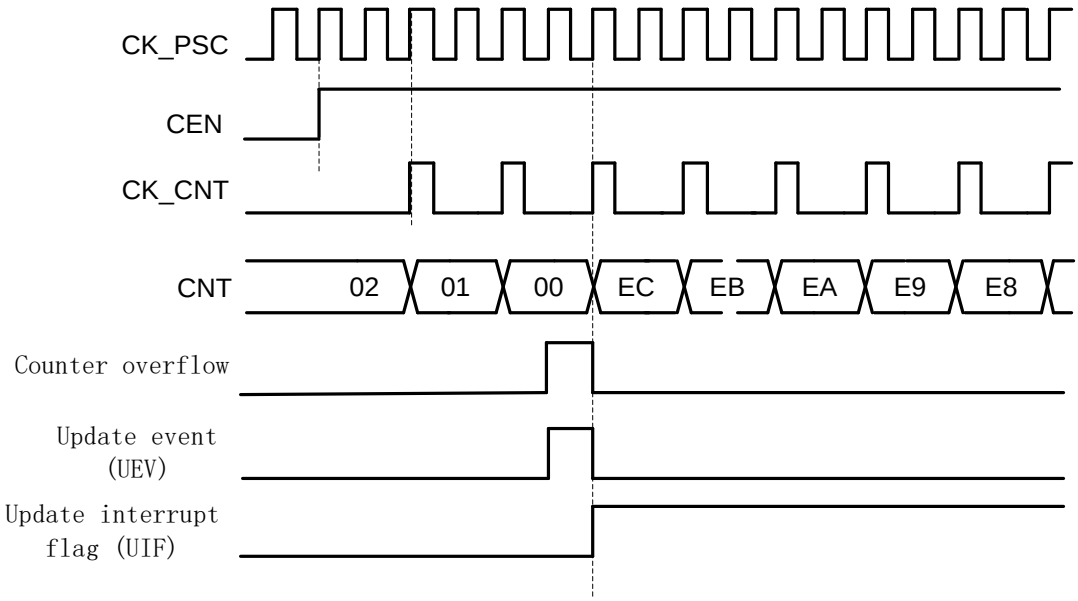


Figure 27-9 Downward counting waveform, internal clock divided-by-2

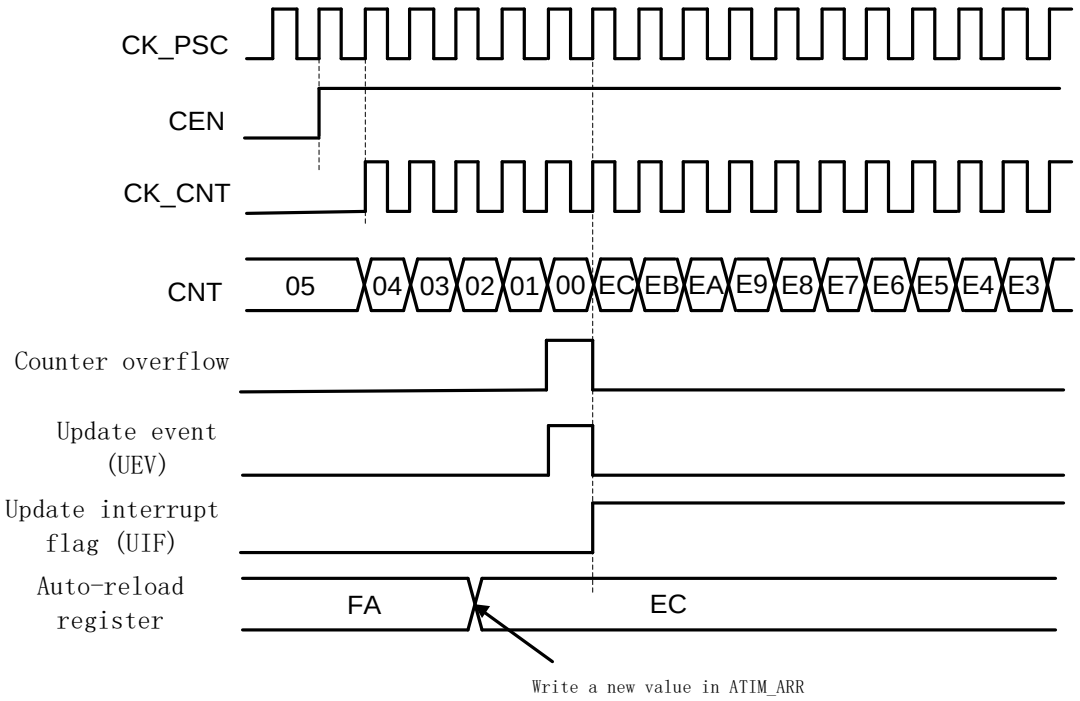


Figure 27-10 Update event when ARPE=0(ARR is not preloaded)

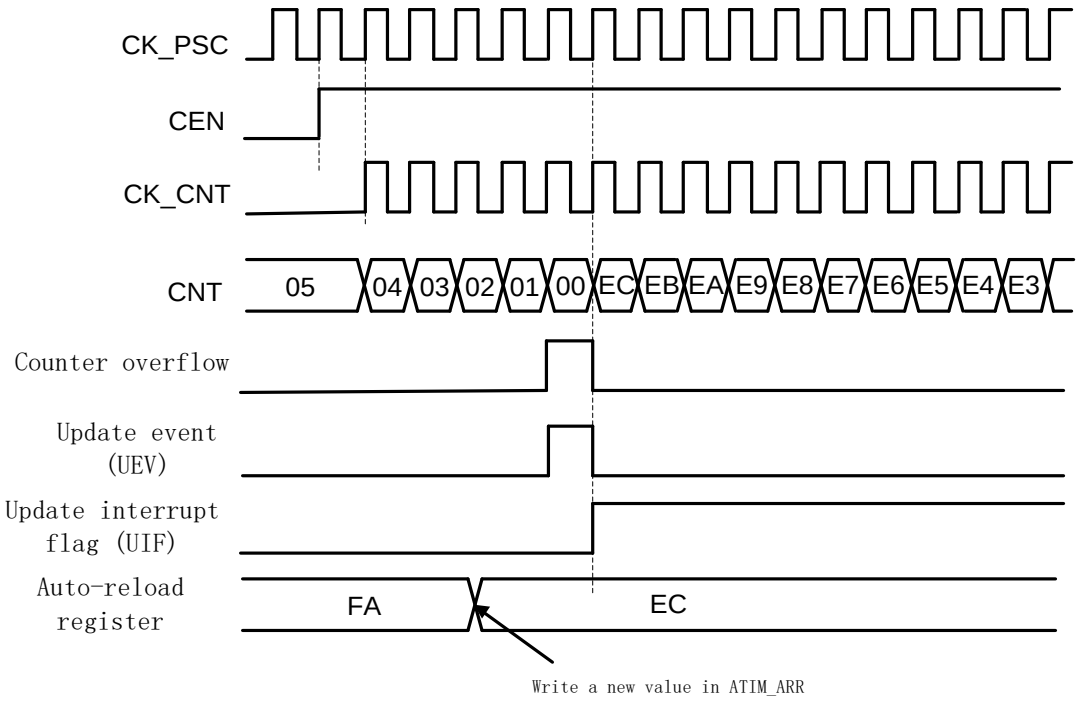


Figure 27-11 Update event when ARPE=1 (ARR preload)

Center alignment count

In the center-aligned mode, the counter starts counting up from 0 until ARR-1 generates an overflow event, and then counts down from ARR to 1, generating an underflow event, and then restarts counting up from 0.

The CMS [1:0] register is used to enable the center-aligned mode and select the output comparison mode in the center-aligned mode. When CMS=00, it is center-aligned counting. When CMS=01, the output comparison function is only valid when counting down. When CMS=10, the output comparison function is only valid when counting up. When CMS=11, The output comparison function is valid when counting up and down.

In center-aligned mode, the DIR register cannot be rewritten by software, but is automatically updated by the hardware as the counting direction changes, indicating the current counting direction.

The counter will update the shadow registers of ARR, PSC and RCR on overflow and underflow events.

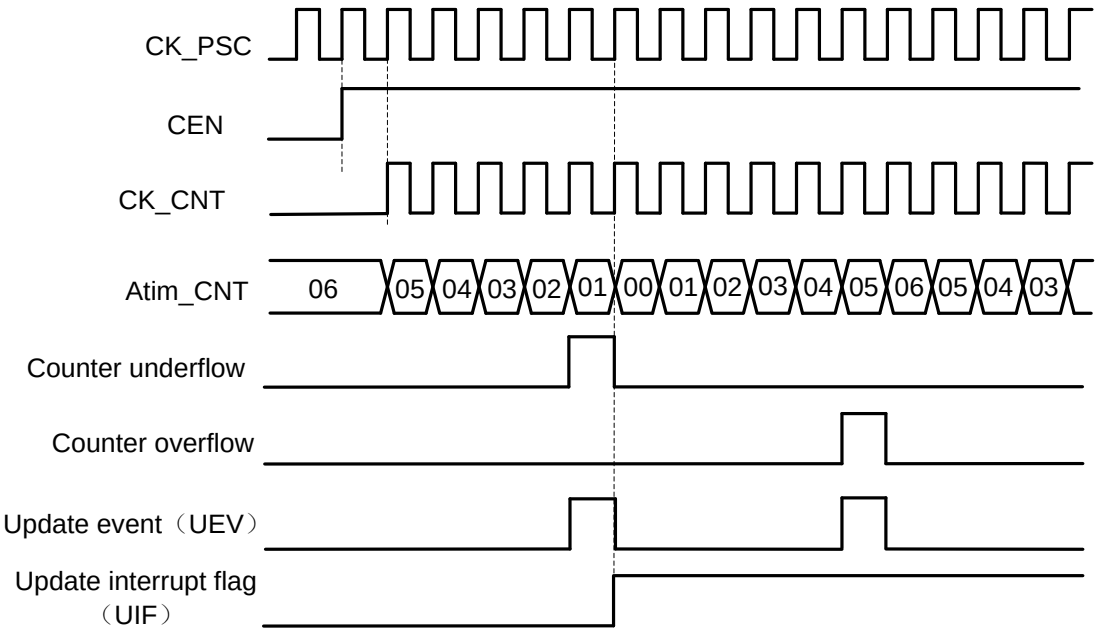


Figure 27-12 Timing diagram of center aligned counter, ATIM_PCS=0, ATIM_ARR=0x6

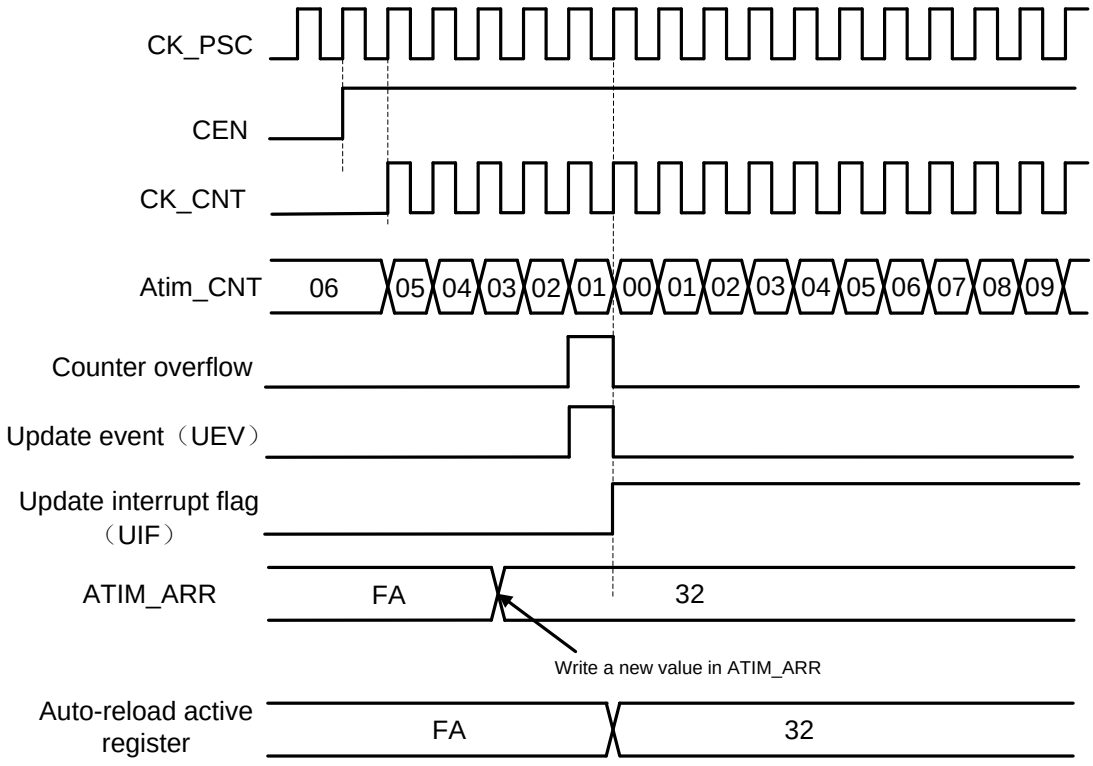


Figure 27-13 Counter timing diagram, update event when ARPE=1 (counter underflow)

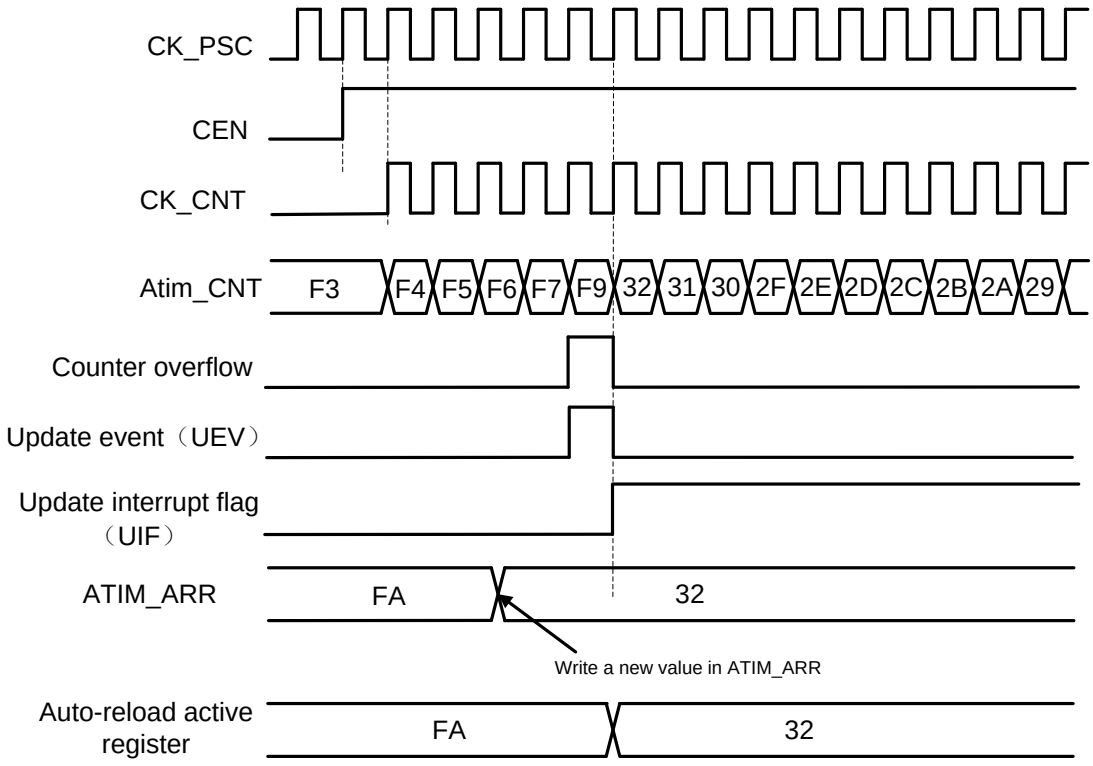


Figure 27-14 Counter timing diagram, update event when ARPE=1 (counter overflow)

27.4.3 Repeat Counter

Update event is generated when the counter overflow or underflow, and the repeat counter is 0. This means that the preload register of ARR, PSC, CCR (compare/capture register, output compare mode) will transfer data to the shadow register after N+1 overflows or underflows, where N is the value of the RCR register.

The repeat counter is decremented under the following conditions:

- Overflow occurs in up-counting mode
- Underflow occurs in down-counting mode
- Each overflow or underflow in the center counting mode

Note that when the update event is triggered by software or slave mode controller, the update event will occur immediately, regardless of the current RCR value, and the repeat counter will also be immediately updated to the value of RCR.

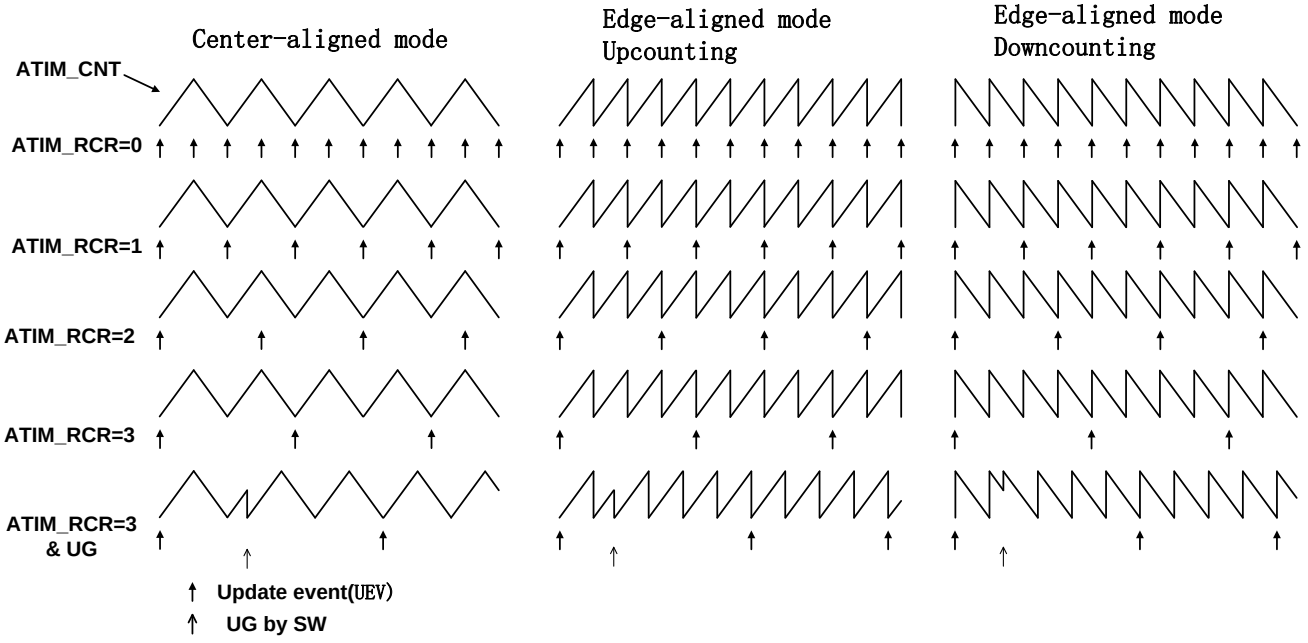


Figure 27-15 Example of update rate in 15 different modes, and register settings of ATIM_RCR

27.4.4 Preload register

The following function registers support the preload function:

- Automatic reload register ARR
- Repeat counting register RCR
- Prescaler register PSC (preload function cannot be turned off)
- Channel Control Register CCR
- CcxE and CcxNE control registers
- OcxM control register

The above registers, except PSC, can be selected to enable or disable the preload function by software.

Registers with preload function include two sets of physical entities:

- Shadow register: the register being used by the actual timer
- Preload register: Registers that can be accessed by software

When preload is disabled, the register features with preload function are as follows:

- Preload register can be accessed and rewritten by software in real time
- The shadow register and the preload register are updated synchronously

If preload is enabled, then:

- All software operations access the preload register
- When an update event occurs, the contents of all preload registers will be transferred to the corresponding shadow register synchronously

27.4.5 Counter operating clock

The counter can use the following clock to work:

- APBCLK-internal clock mode
- External pin input clock (Tix)-external clock mode 1
- External pin trigger input (ETR)-external clock mode 2
- Internal trigger (ITRx)-use a timer's trigger output (TGO) as the counting clock

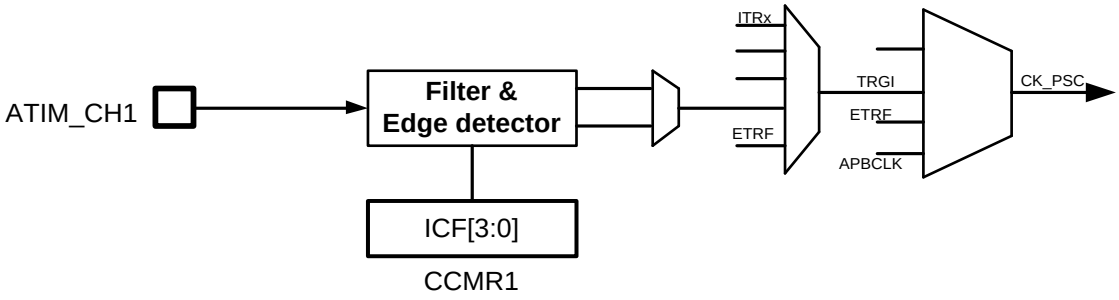


Figure 27-16 ATIM clock source block diagram

27.4.5.1 Internal clock mode

In internal clock mode, slave mode is prohibited (SMS=000), and register bits such as CEN, DIR, UG, etc. are all under software control.

After the software operates the UG register, after the update signal is synchronized by CLK_PSC, the counter value will be reinitialized.

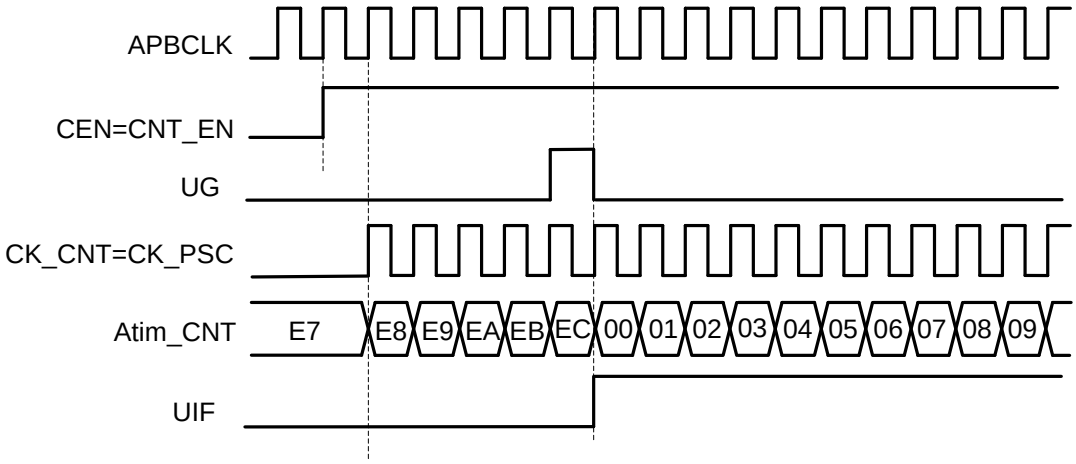


Figure 27-17 Internal clock source mode with clock division factor of 1

27.4.5.2 External clock mode1

In this mode, the external pin input signal is directly used as the counting clock, and SMS=111 is configured, and the counting edge can be configured as a rising or falling edge.

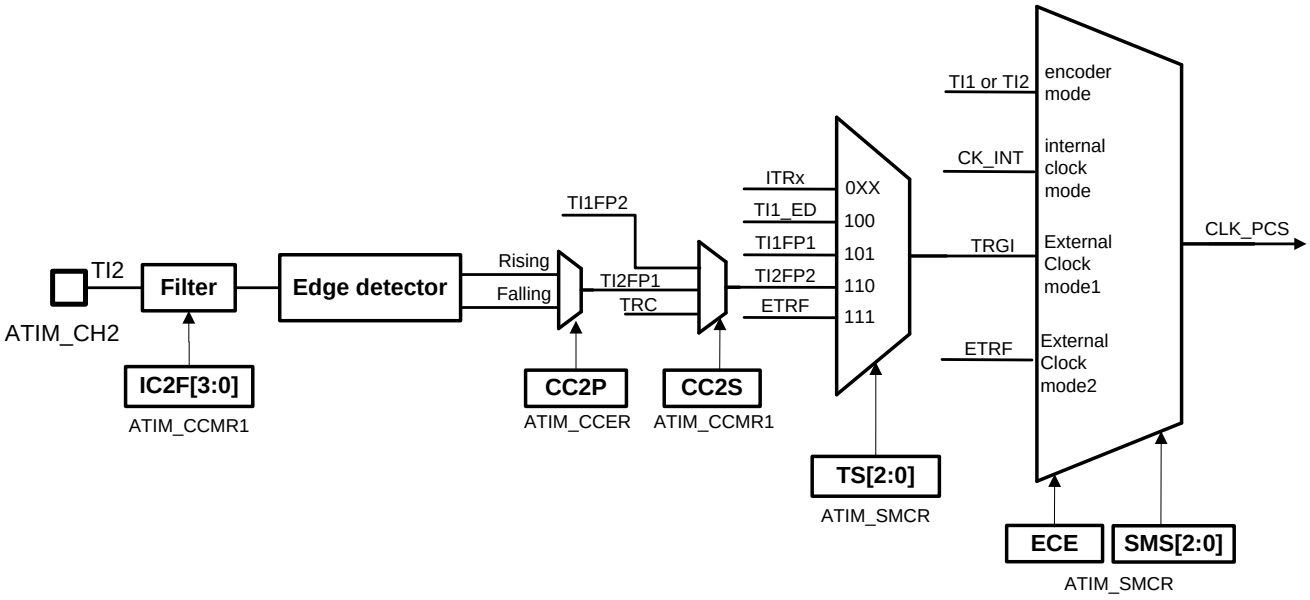


Figure 27-18TI2 external clock connection example

The external input signal will go through the synchronization process of the internal clock before triggering the counter to count. At the same time, the valid edge of the input signal will trigger the TIF mark.

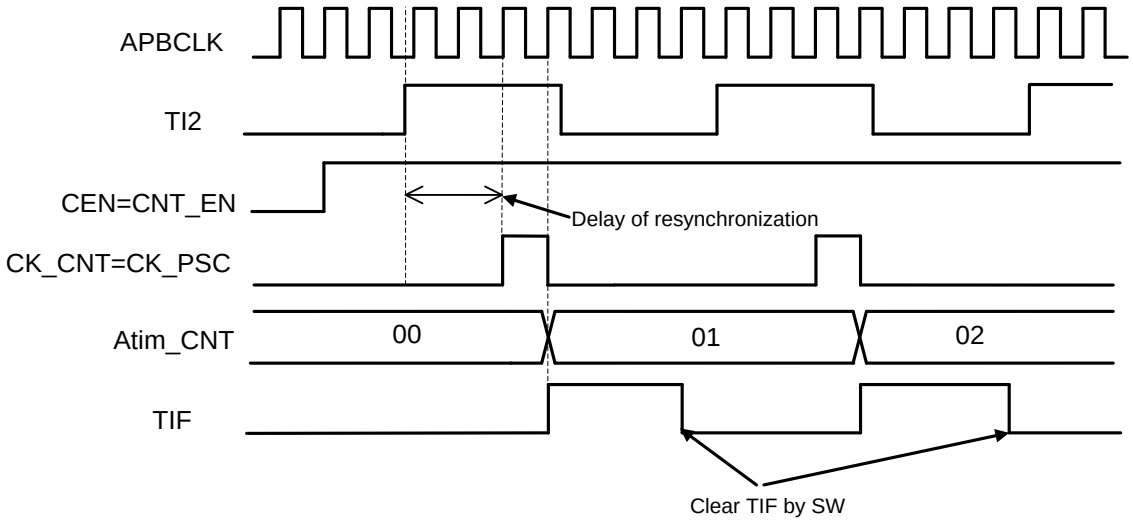


Figure 27-19 Timing in external clock mode 1

When using an external clock to count, still need to enable ATIM's internal clock (APBCLK), because ATIM uses APB_CLK to synchronize and filter the external input clock. In external clock mode 1, the external input clock is filtered and edge selected first to obtain a valid counting edge, which is input to the prescaler module as a valid working clock (CLK_PSC).

The external clock synchronization uses a simple 2-level flip-flop structure. Therefore, in order to avoid metastability, the external input clock width is required to be at least two APB_CLK cycles.

In this mode, only the inputs of channels 1 and 2 can be used as clock inputs, and the required configuration is as follows:

- In the GPIO module, configure the corresponding pin as ATIM_CH2 function
- Turn off the channel enable and configure ATIM_CCER.CC2E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure ATIM_CCMR1.CC2S=01, IC2 is mapped to TI2
- Select the count valid edge, configure ATIM_CCER.CC2P=0, select the up or down edge
- Configure the input filter time, configure ATIM_CCMR1.IC2F[3:0] (IC2F=0000, no input filter)
- Enable external clock mode 1, configure ATIM_SMCR.SMCR=111

- Select the trigger input source, configure ATIM_SMCR.TS=110, select TI2 as the trigger input source
- Turn on the channel enable, configure ATIM_CCER.CC2E=1
- Enable the counter, configure ATIM_CR1.CEN=1

The following figure is an example of a typical external clock counting mode 1:

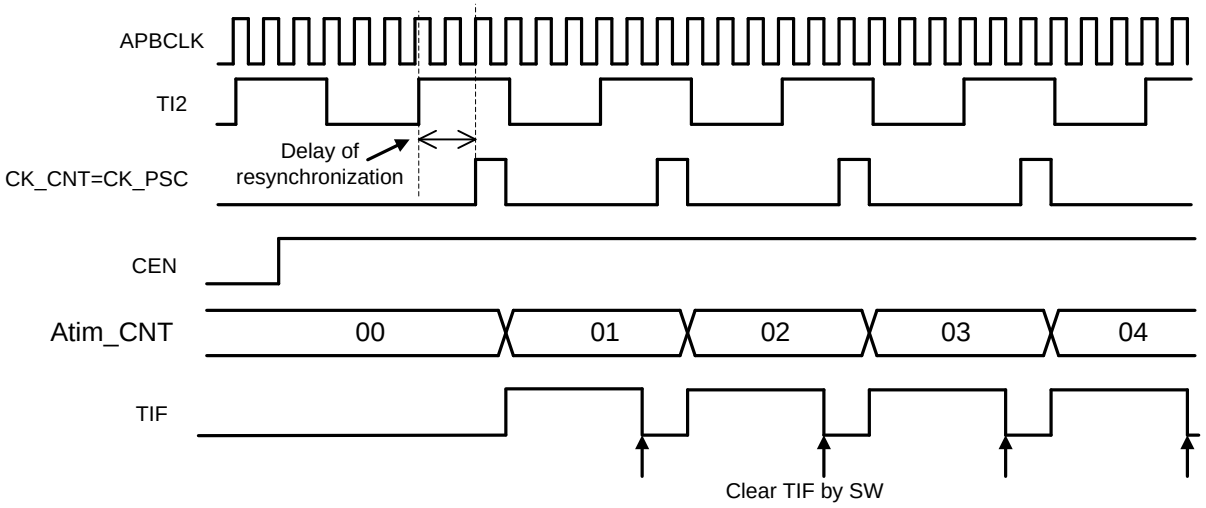


Figure 27-20 Timing in external clock mode 1

27.4.5.3 External clock mode 2

In this mode, the rising or falling edge of the input signal at the ATIM_ETR pin is used for counting (double edges are not supported).

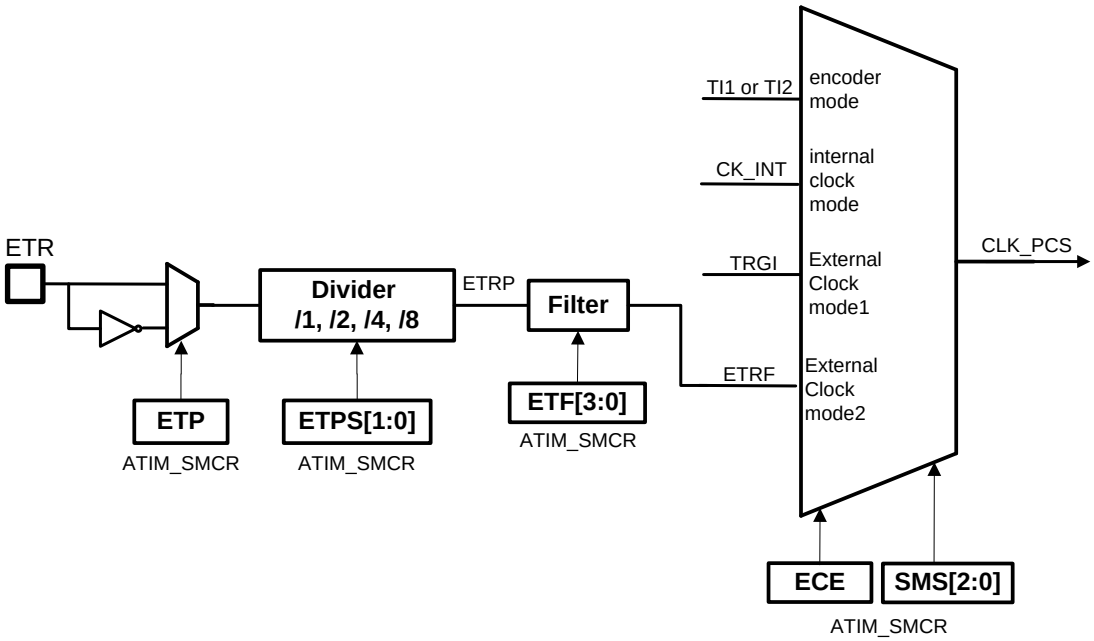


Figure 27-21 External trigger input block diagram

The figure below uses the rising edge of the ETR divided by two to count. The actual counting time is delayed from the rising edge of the ETR input due to the synchronization process of the internal clock.

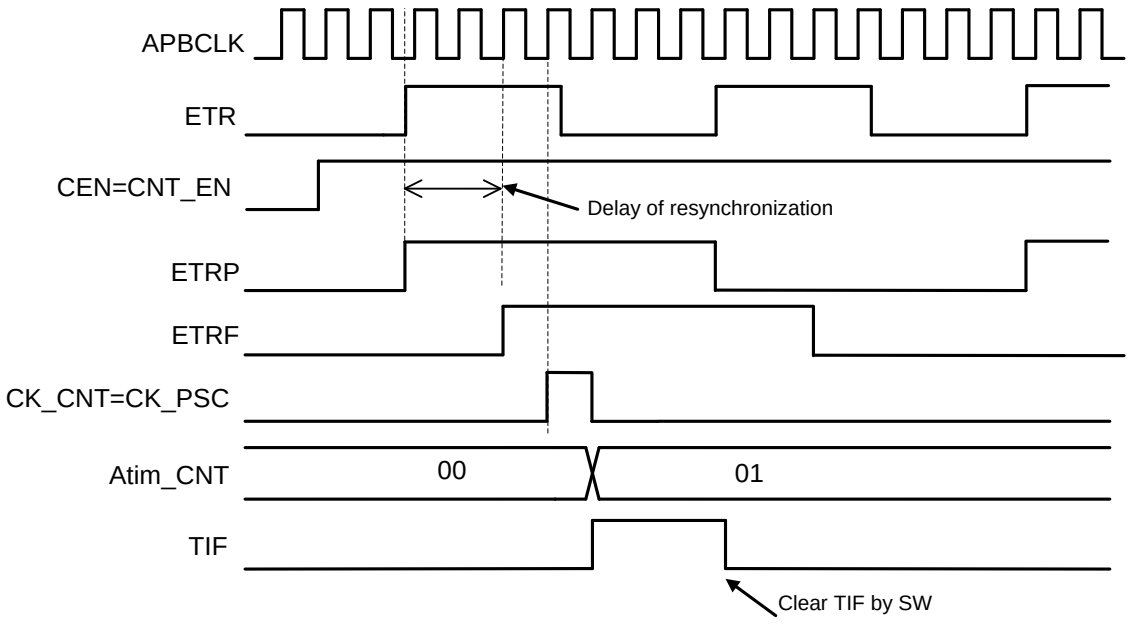


Figure 27-22 Timing 1 in external clock mode 2

The main difference with external clock mode 1 is that the ETR input is directly divided by frequency and then filtered to generate the CK_PSC clock, which means that it can support the application

scenarios where the ETR input frequency is higher than APB_CLK. In this case, you need to input the ETR first Carry out prescaler, and then use it to drive the counter.

The configuration required for this mode is as follows:

- In the GPIO module, configure the corresponding pin for the ATIM_ETR function
- Set ETP for edge selection, ATIM_SMCR.ETP=0
- Set the ETR division ratio, configure ATIM_SMCR.ETPS[1:0]=01
- Configure the input filter time, ATIM_SMCR.ETF[3:0]=0000
- Set the ECE register, enable external clock mode 2, ATIM_SMCR.ECE=1, ATIM_SMCR.SMS=000
- Enable the counter, configure ATIM_CR1.CEN=1

The following figure is an example of a typical external clock mode 2:

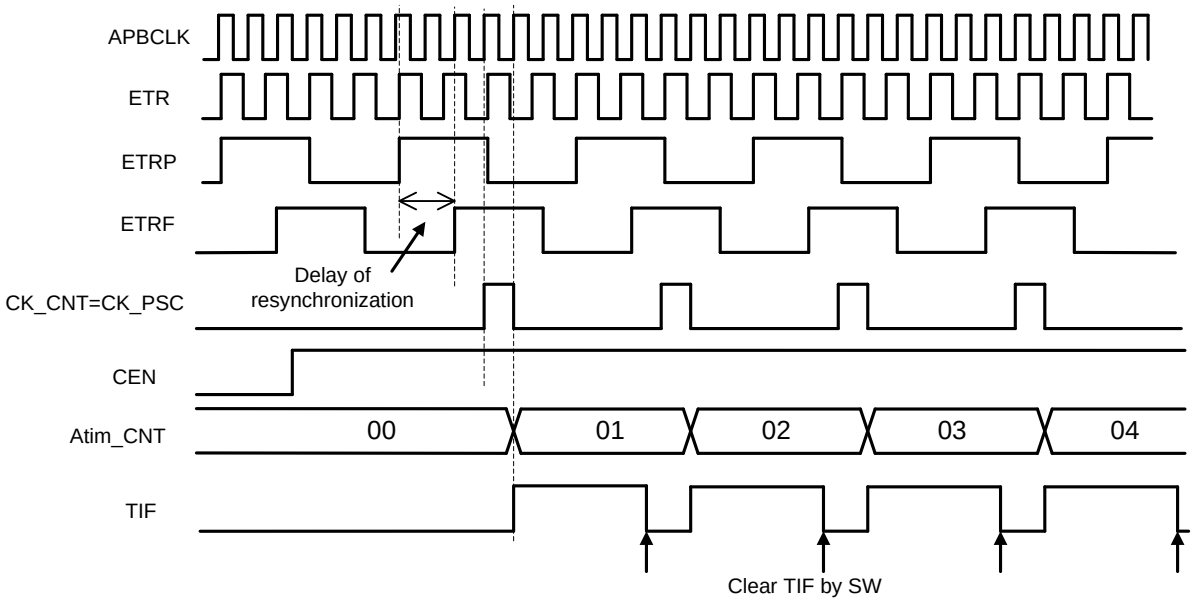


Figure 27-22 Timing 2 in external clock mode 2

When using external clock mode 2, ATIM can still be configured as slave mode: for example, use the ETR input to count, and at the same time use the TRGO of another Timer as the trigger signal, when the trigger event arrives, the reset counter restarts counting.

27.4.6 Internal trigger signal（ITRx）

ATIM supports 4 ITR inputs, which can be used for counting trigger or internal signal capture. When used for internal signal capture, TS needs to be configured as 000~011 to select ITR0~ITR3, and CCxS is configured as 11, that is, TRC is selected as the capture signal.

Each ITR input supports 4 internal signal extensions, which are configured by the ITRxSEL register. Refer to the following table for input signal source:

Slave	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
ATIM	GPTIM0_TRGO	GPTIM1_TRGO	COMP1	COMP2

27.4.7 Capture/Compare channel

ATIM contains 4 capture/compare channels, and each channel consists of a capture compare register (CCR) (including shadow registers), a capture input stage, and a compare output stage.

The input stage circuit will sample the Tix input and generate a filtered signal TixF, and then edge detection and polarity selection will generate the corresponding TixFPx signal. This signal can be used as a counting trigger or a signal to be captured, and is prescaled before being captured.

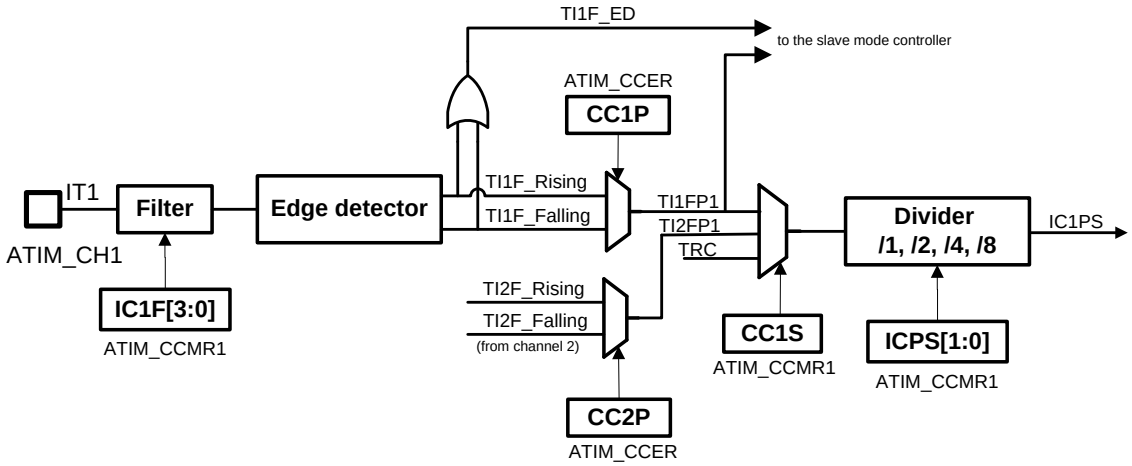


Figure 27-23 Capture/compare channel (channel 1 input part)

The output stage circuit will generate an output reference signal OCxREF, which is fixed to high level and effective as the reference input of the final output circuit. Among them, channels 1~3 support complementary output and dead zone insertion, while channel 4 is relatively simple and does not support complementary output.

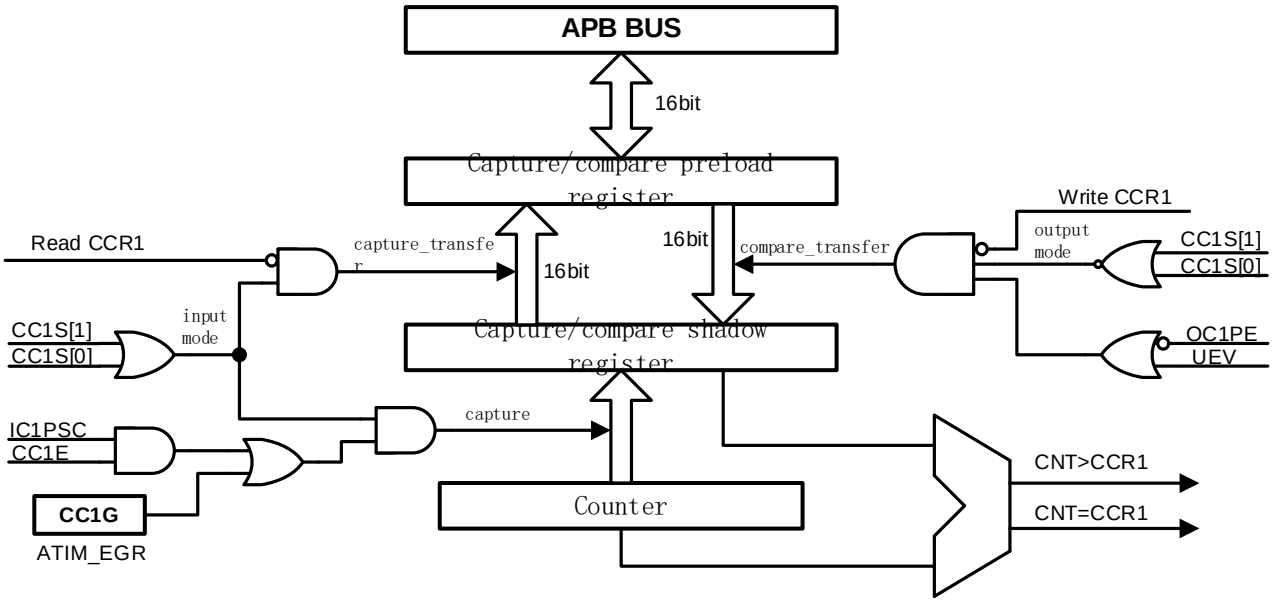


Figure 27-24 Main circuit of capture/compare channel 1

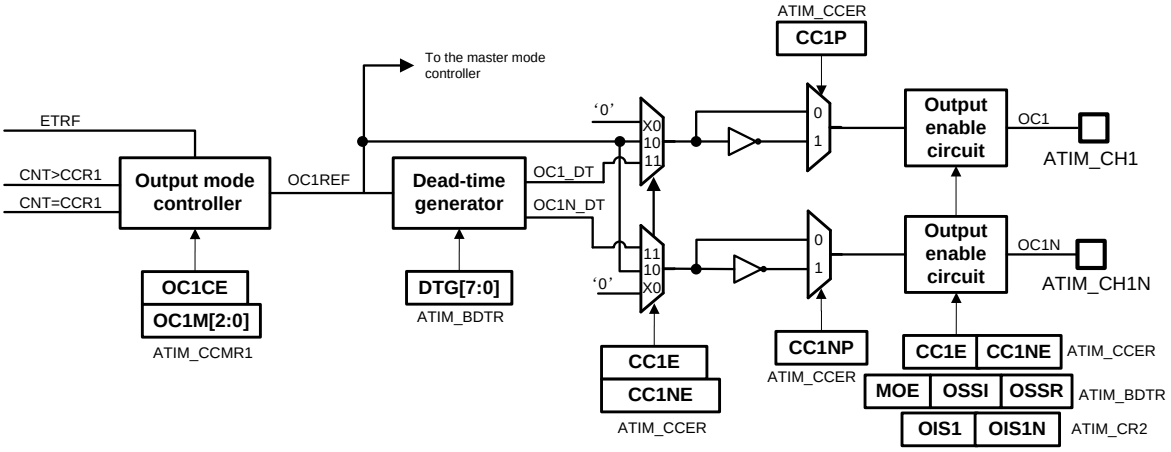


Figure 27-25 Output section of capture/compare channel (channels 1 to 3)

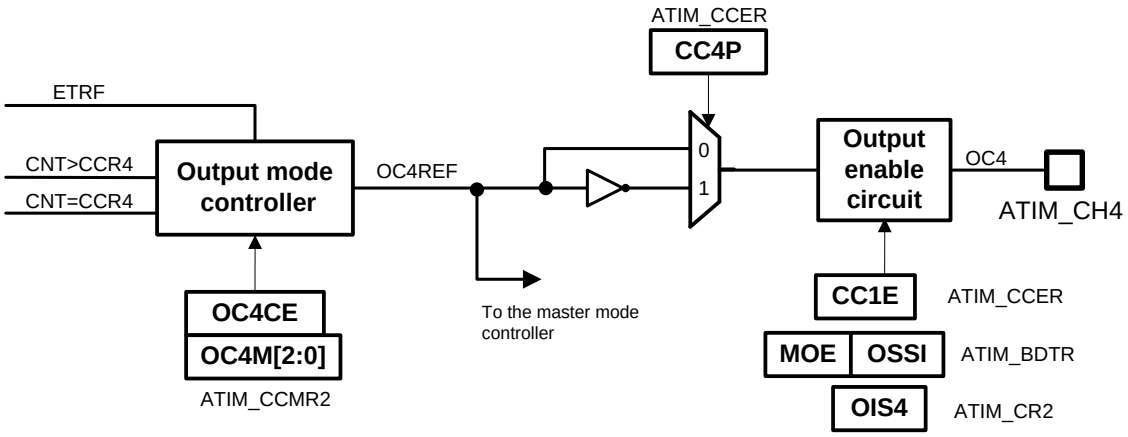


Figure 27-26 Output part of the capture/compare channel (channel 4)

The capture/compare register (CCR) contains the preload register and the shadow register, and software reads and writes always access the preload register. In the capture mode, the captured value is saved in the shadow register and copied to the preload register. In the compare mode, the value of the preload register is copied to the shadow register for comparison with the counter.

27.4.8 Input capture mode

When the expected level change occurs on the Icx signal, a capture will be triggered, and the current counter value is latched into the CCR. At the same time, the CCxIF interrupt flag is set, and the corresponding interrupt or DMA request can be triggered. If a capture event occurs when CCxIF is high, the capture data conflict flag (CCxOF, Over-Capture) is set (the last captured value in CCR is overwritten). CCxIF can be cleared by software or automatically cleared by reading the CCR

register. The CCxOF flag is cleared by writing 1 in software.

Through the cooperation of two or more channels, the input capture of the PWM signal can be realized. For example, if you want to calculate the period and duty ratio of an input signal, you can input this signal from the TI1 pin, and the chip will take the rising edge of the filtered signal to get TI1FP1, take the falling edge of the filtered signal to get TI1FP2, and input TI1FP1 To capture channel 1, input TI1FP2 to capture channel 2, then channel 1 captures the rising edge of the input signal, while channel 2 captures the falling edge of the input signal; after the capture interrupt occurs periodically, the software passes the value of the CCR1 and CCR2 registers, The period and duty cycle of the input signal can be calculated.

To achieve the capture of the counter value to the ATIM_CCR1 register at the rising edge of TI1 input, the configuration steps are as follows:

- In the GPIO module, configure the corresponding pin as ATIM_CH1 function
- Turn off the channel enable and configure ATIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure ATIM_CCMR1.CC1S=01, IC1 is mapped to TI1
- Select the count valid edge, configure ATIM_CCER.CC1P, select the up or down edge
- Configure the input filter time, configure ATIM_CCMR1.IC1F[3:0]
- Configure the input prescaler, configure ATIM_CCMR1.IC1PS[1:0]
- Turn on the channel enable, configure ATIM_CCER.CC1E=1

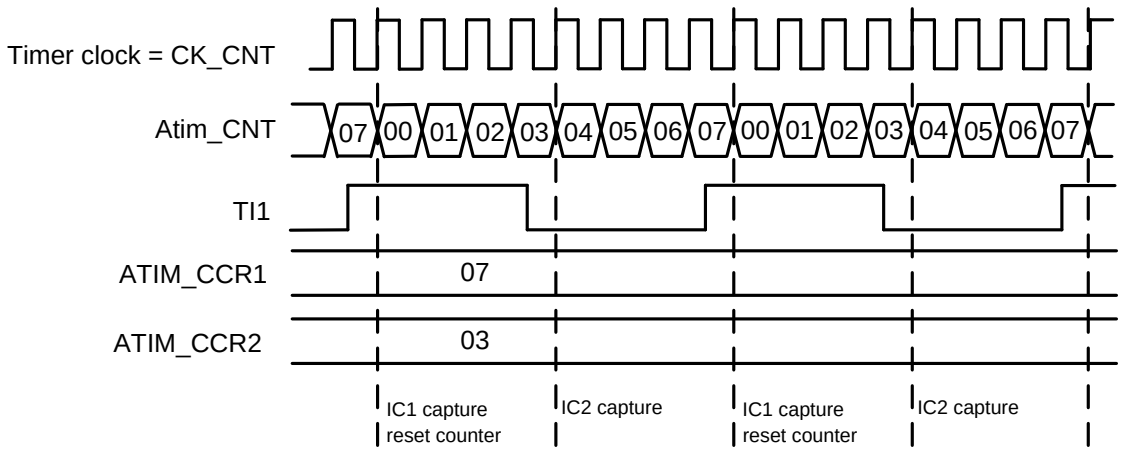


Figure 27-27 PWM input and capture mode timing

If you want to realize the PWM input capture function, you need to make the following settings:

- In the GPIO module, configure the corresponding pin as ATIM_CH1 function
- Turn off the channel enable, configure ATIM_CCER.CC1E=0, ATIM_CCER.CC2E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, the two channels IC1, IC2 are mapped to the same TI1 input port, configure ATIM_CCMR1.CC1S=01, ATIM_CCMR1.CC2S=10
- Select the count valid edge, the two channels IC1, IC2 valid edge polarity is opposite, configure ATIM_CCER.CC1P=0, ATIM_CCER.CC2P=1
- Configure the input filter time, configure ATIM_CCMR1.IC1F[3:0], ATIM_CCMR1.IC2F[3:0]
- Configure the input prescaler, configure ATIM_CCMR1.IC1PS [1:0], ATIM_CCMR1.IC2PS [1:0]
- Select the trigger input signal, configure ATIM_SMCR.TS[2:0] =101
- Set the slave mode controller to reset mode, configure ATIM_SMCR.SMS [2:0] =100
- Turn on the channel enable, configure ATIM_CCER.CC1E=1, ATIM_CCER.CC2E=1

27.4.9 Software force output

In the comparison output mode, the software can directly set the OCxREF force to a specific level, independent of the comparison result of the CCR and the counter.

The software can directly force OCxREF to be valid by writing OcxM=101 register (OCxREF is fixed as high and effective), and by writing OcxM=100, OCxREF can be directly forced to be invalid (low level). However, the software force operation will not cancel the comparison process, and the comparison between CCR and counter will continue.

27.4.10 Output compare mode

In the output compare mode, when CCR is equal to the counter value, OCxREF can be set to valid, invalid, or level inverted. At the same time, the interrupt flag will also be set, and DMA requests can be sent (overwrite the configuration register?).

Output comparison can also be used to output a pulse signal of a specific width (single output).

Steps for usage:

- Select the count clock (internal, external, prescaler, etc.)
- Write the desired data to the ARR and CCR registers
- Set interrupt enable and DMA enable as needed
- Select output mode
- Enable counter

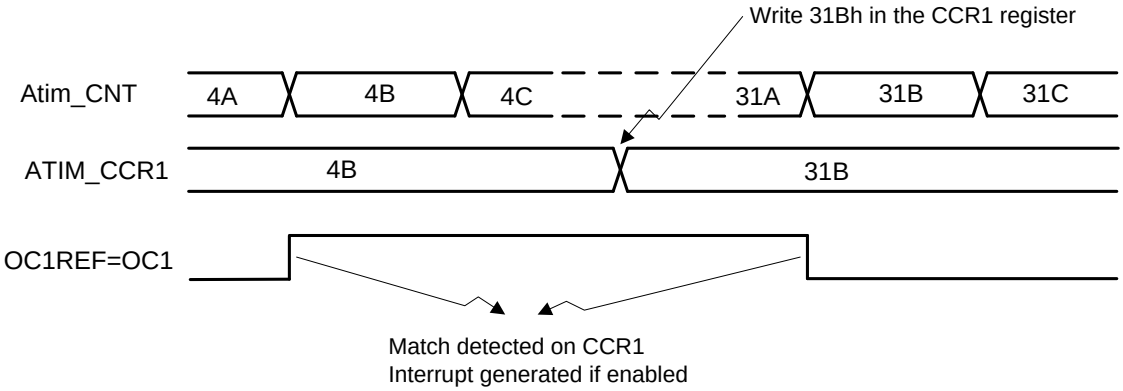


Figure 27-28 Output compare mode, flip OC1

Without enabling preload, the software can rewrite the CCR register at any time to achieve real-time control of the output waveform. If preload is enabled, the CCR shadow register is only updated with the contents of the preload register when the next update event occurs.

27.4.11 PWM output

PWM mode can output pulse width modulation signal, its period is determined by the ARR register, and the duty cycle is determined by the CCR register.

The polarity of the output signal can be configured by the CCxP register. In PWM mode, CNT and CCR are compared in real time. Since the counter supports edge-aligned and center-aligned counting modes, the PWM output also supports edge-aligned and center-aligned modes.

PWM edge alignment mode

In the case of counting up, when configured as PWM mode 1, the OCxREF signal is high when CNT<CCR, otherwise it is low. If the CCR value is greater than the ARR value, OCxREF is fixed to 1; if CCR is 0, OCxREF is fixed to 0.

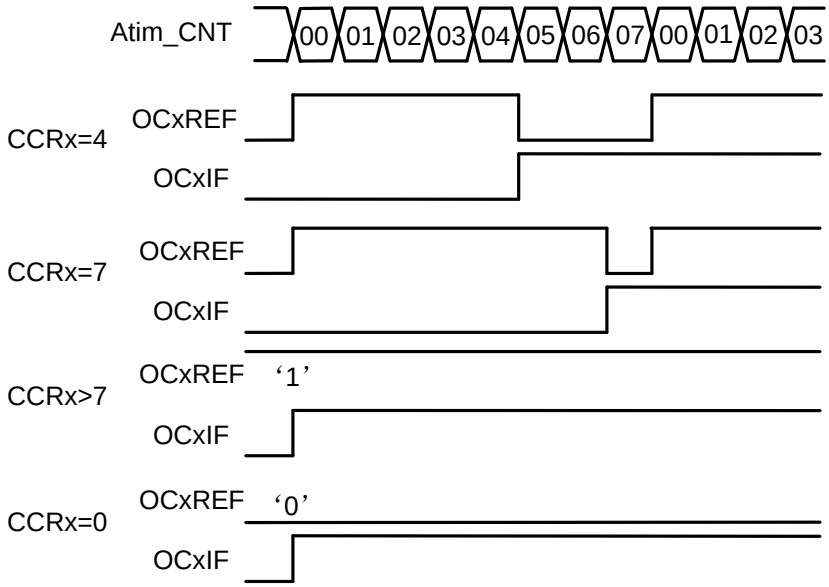


Figure 27-29 Edge-aligned PWM waveform (ARR=7)

When counting down, the definition of OCxREF level is the same as when counting up.

PWM center alignment mode

The OCxREF level definition is the same as the edge alignment mode. The following figure is an example

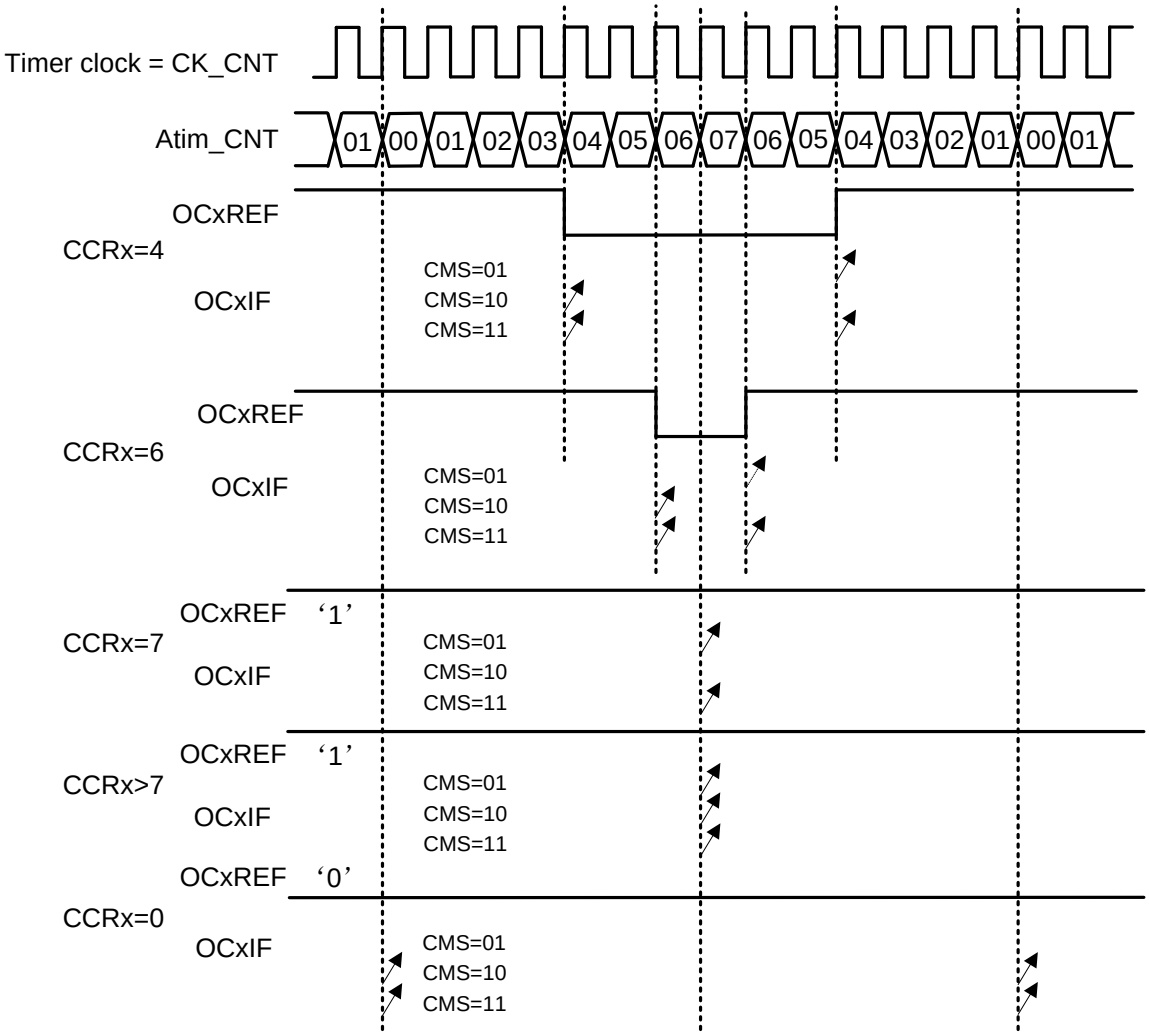


Figure 27-30 Center-aligned PWM waveform (APR=7)

When starting the center-aligned counting, the initial counting direction is determined by the DIR register; then during the counting process, the state of the DIR register is directly controlled by the hardware. For safety, it is recommended that the user program do an update through the UG register before starting the counter, and do not rewrite the counter during the counting process.

27.4.12 Complementary output and dead zone insertion

ATIM channels 1~3 support complementary output and dead zone insertion. The DTG[7:0] register is used to set the dead time (valid for all channels at the same time). The output signal Ocx is in phase with the reference signal OCxREF, and OcxN is inverted from the reference signal; the rising edge of Ocx is the delay of the rising edge of OCxREF, and the rising edge of OcxN is the delay of the falling edge of OCxREF.

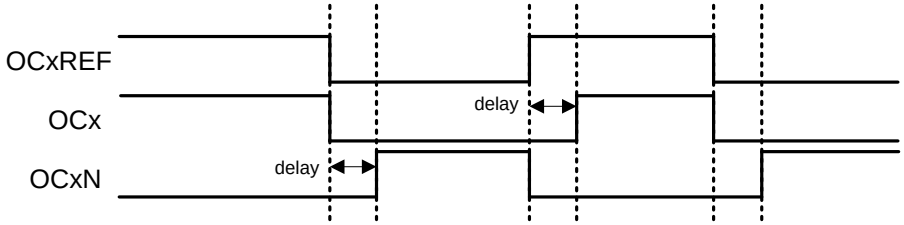


Figure 27-31 Complementary output with dead zone insertion

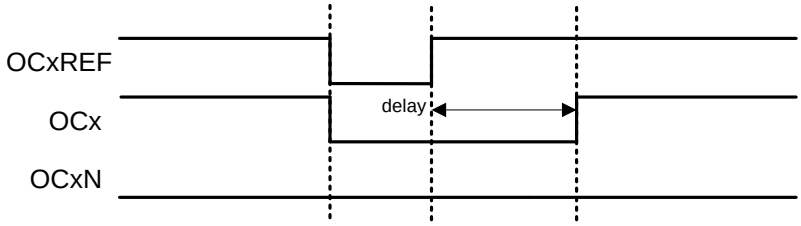


Figure 27-32 Dead zone waveform delay is greater than negative pulse

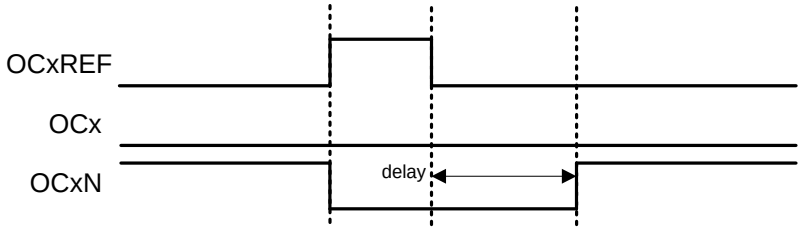


Figure 27-33 Dead zone waveform delay is greater than positive pulse

27.4.13 Brake function

The brake function can use the 2 brake signals input by the external BRK pin, or the valid output generated by the comparator, SVD, XTHF vibration stop detection; the brake circuit is disabled after power-on reset, and the user can enable the brake function by setting the BKE register; The 2 brake inputs can be configured as phase AND or phase OR operation. The combined brake signal can be configured with effective polarity and digital filtering.

The brake input control logic is shown in the figure below:

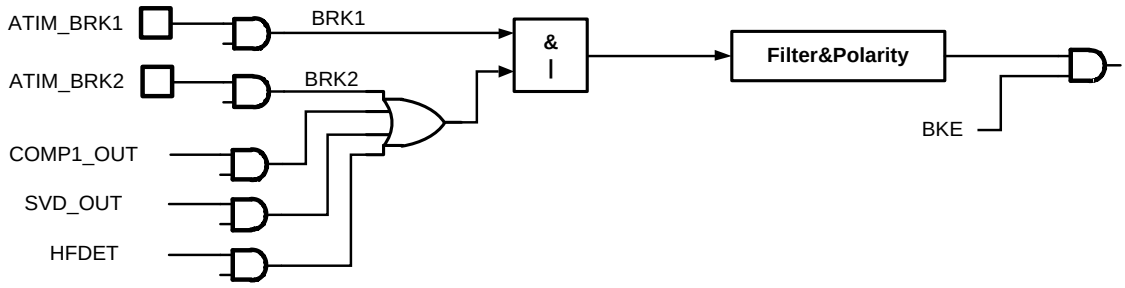


Figure 27-34 Brake control logic

ATIM_BRKx multiplexes the GPIO function. When GPIO is set as a digital peripheral function, its input signal is directly connected to the brake input of ATIM; when GPIO is set to other functions, the brake input port of ATIM is fixed to 1. Through the BRKxGATE register, the actual level of the gated BRKx signal can be controlled, and the software can flexibly set the unused BRKx to 0 or 1 level to meet the needs of the subsequent logic circuit.

When a braking event occurs:

- The output enable register is cleared asynchronously, and the output can be forced to the inactive/idle/reset state by selecting the OSSI register
- Each output channel is driven to the level defined by the OISx register
- When the complementary output is enabled, the output is asynchronously set to the inactive and reset states, and the dead-zone insertion circuit starts to work. After the dead-zone time, the output is driven to the level defined by OISx and OISxN
- The brake flag register is set to bit, and interrupt or DMA can be triggered according to the configuration
- If the automatic output is enabled (AOE=1), the output enable bit (MOE) will be automatically set when the next update event occurs; otherwise, the MOE will remain at 0 until the bit is reset

by the software.

Note that the BRK signal is level valid, so when BRK remains valid, MOE cannot be enabled, and the brake flag BIF cannot be cleared.

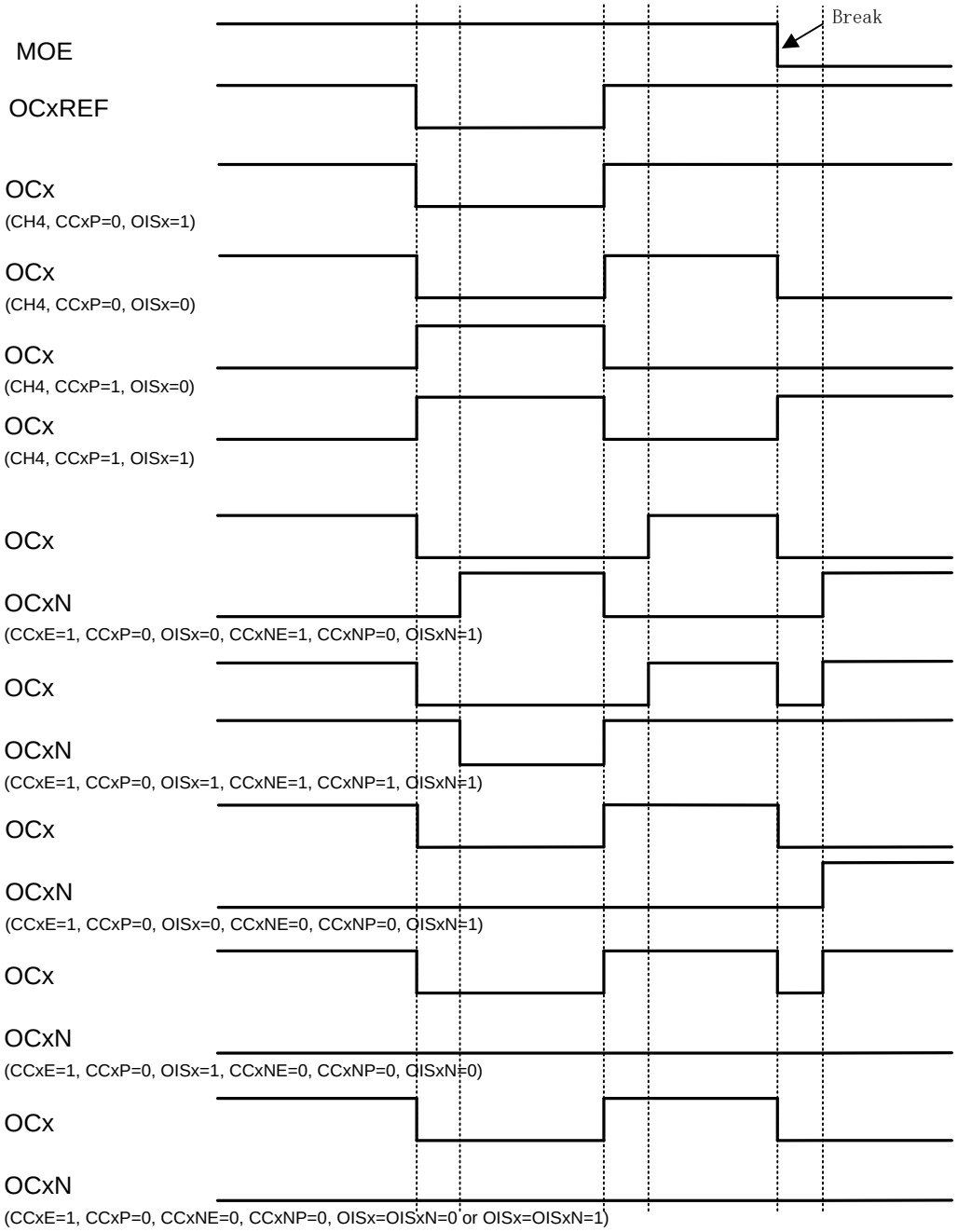


Figure 27-35 Response to brake output

27.4.14 Complementary output channel signal state logic table

The following is the state correspondence table of the control register and the complementary output channel, where MOE is the timer total output enable bit, OSSI defines whether to turn off the IO output or enter the off state in the IDLE state (MOE=0), and OSSR defines the RUN state (MOE= 1) The next is to close the IO output or enter the off state.

Control register					Output status	
MOE	**OSSI**	**OSSR**	**CCxE**	**CCxNE**	**OCx Output status**	**OCxN Output status**
1	X	0	0	0	Output is off (not driven by ATIM),OCx=0,OCx_EN=0	Output is off (not driven by ATIM), OCxN=0,OCxN_EN=0
		0	0	1	Output is off (not driven by ATIM),OCx=0,OCx_EN=0	OCxREF + Polarity OCxN=OCxREF xor CCxNP, OCxN_EN=1
		0	1	0	OCxREF + Polarity OCx=OCxREF xor CCxP, OCx_EN=1	Output Disabled (not driven by the timer) OCxN=0, OCxN_EN=0
		0	1	1	OCREF + Polarity + dead-time OCx_EN=1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN=1
		1	0	0	Output Disabled (not driven by the timer) OCx=CCxP, OCx_EN=0	Output Disabled (not driven by the timer) OCxN=CCxNP, OCxN_EN=0
		1	0	1	Off-State (output enabled with inactive state) OCx=CCxP, OCx_EN=1	OCxREF + Polarity OCxN=OCxREF xor CCxNP, OCxN_EN=1
		1	1	0	OCxREF + Polarity OCx=OCxREF xor CCxP, OCx_EN=1	Off-State (output enabled with inactive state) OCxN=CCxNP, OCxN_EN=1
		1	1	1	OCREF + Polarity + dead-time OCx_EN=1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN=1
0	0	X	0	0	Output is off (not driven by ATIM) OCx=CCxP, OCx_EN=0	Output is off (not driven by ATIM) OCxN=CCxNP, OCxN_EN=0
	0		0	1	Output is off (not driven by ATIM) If no clock: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0 If clocked: OCx=OISx, OCxN=OISxN after dead time	
	0		1	0		
	0		1	1		
	1		0	0	Output is off (not driven by ATIM) OCx=CCxP, OCx_EN=0	Output off (not driven by ATIM) OCxN=CCxNP, OCxN_EN=0
	1		0	1	Off-state (output enable, inactive output) If no clock: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1 If clocked: OCx=OISx, OCxN=OISxN after dead time	
	1		1	0		
	1		1	1		

27.4.15 6-step PWM output

When a channel uses complementary output, the OcxM, CcxE, and CcxNE registers support the preload function, and the value of the preload register is loaded into the shadow register when a commutation (COM) event occurs. Therefore, the user can set the next configuration in advance and update all channels synchronously when a COM event occurs. The COM event can be triggered by the COMbit in ATIM_EGR written by the software, or triggered by the hardware on the rising edge of TRGI.

When a COM event occurs, the commutation flag register is set to bit, and an interrupt or DMA request can be generated.

The following figure is an example of 6-step commutation control. When a COM event occurs, the three examples show the output changes under different configurations.

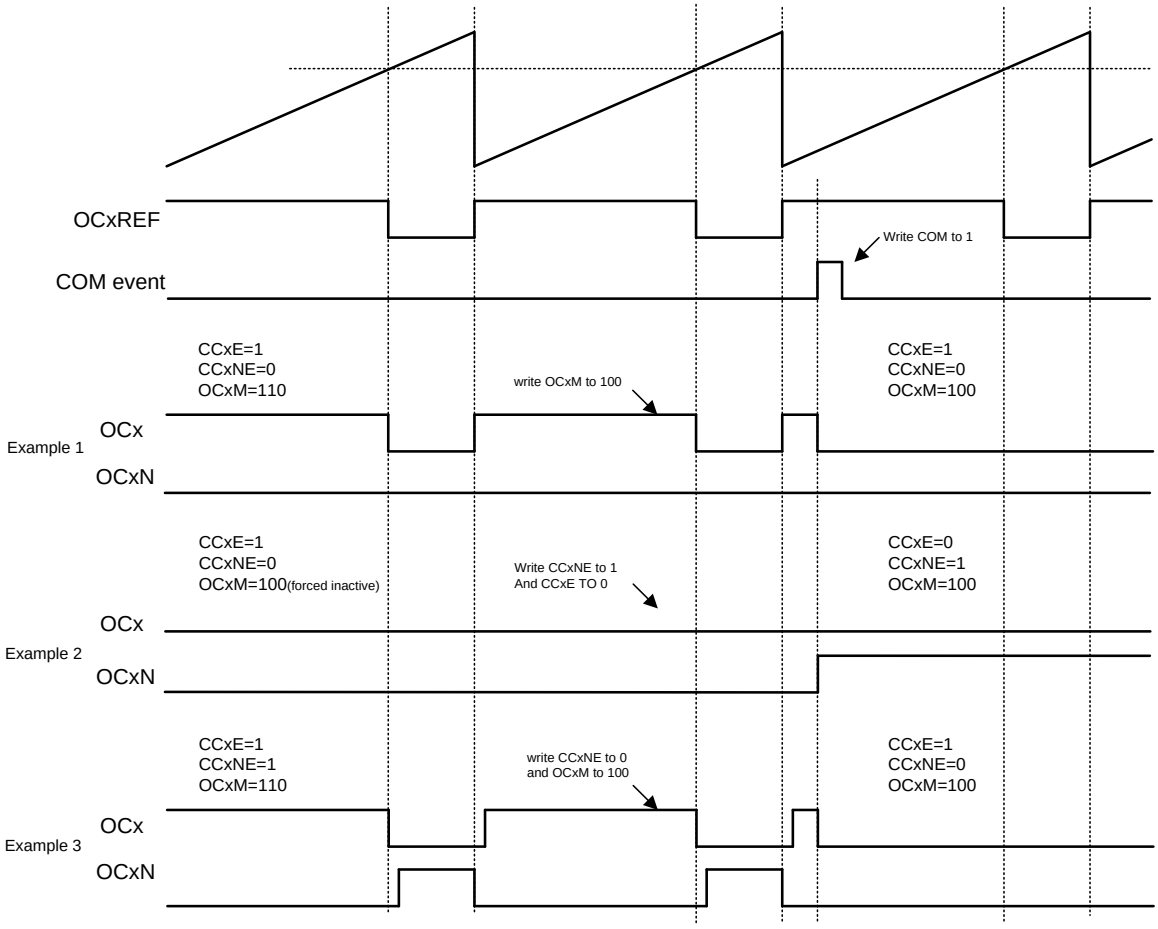


Figure 27-36 Example of generating six-step PWM using COM (OSSR=1)

27.4.16 Single pulse output

Single pulse output is a special case of the comparison output mode, which allows the user to output a pulse signal with a programmable width after a certain event occurs, after a programmable delay.

Unlike other output modes, the counter will automatically stop when the next update event arrives. Only when the initial value of CCR and the counter are different, the pulse can be output correctly. When counting up, CNT<CCR<=ARR is required, when counting down, CNT>CCR is required.

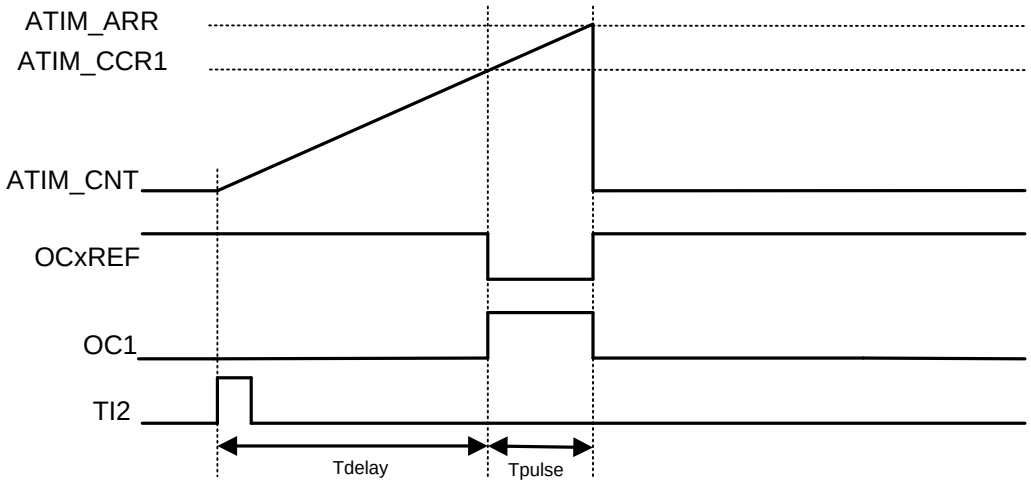


Figure 27-37 Example of single pulse mode

The above figure uses TI2 input as the counter trigger signal. After the count value is equal to CCR, OCxREF outputs low level. After counting to ARR, OCxREF returns to high level, and the counter rolls back to 0 and stops counting.

The configuration that realizes the above function TI2 as an input trigger is as follows:

- In the GPIO module, configure the corresponding pin as ATIM_CH2 function
- Turn off the channel enable and configure ATIM_CCER.CC2E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure ATIM_CCMR1.CC2S=01
- Select valid edge of counting, configure ATIM_CCER.CC2P=0
- Select the trigger input signal, configure ATIM_SMCR.TS[2:0]=110, TI2FP2 as TRGI
- Set the slave mode controller to trigger mode, configure ATIM_SMCR.SMS[2:0]=110, TI2FP2 is used to start the counter
- Turn on the channel enable, configure ATIM_CCER.CC2E=1

The configuration that realizes the above function OC1 as an output is as follows:

- In the GPIO module, configure the corresponding pin as ATIM_CH1 function
- Turn off the channel enable and configure ATIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards
- Output channel, configure ATIM_CCMR1.CC1S=00
- Select the effective edge of counting, configure ATIM_CCMR1.OC1M=111, PWM mode 2
- Turn on the channel enable, configure ATIM_CCER.CC1E=1

Special settings of OPM waveform generation time base:

- The value of ATIM_CCR1 determines Tdelay

- The difference between ATIM_ARR and ATIM_CCR1 determines the Tpulse (ATIM_ARR-ATIM_CCR1)
- Set to single pulse mode,configure ATIM_CR1.OPM=1

27.4.17 External event clears OCxREF

The effective state of OCxREF is not high. By applying a high level to the external ETR pin, OCxREF can be directly pulled down until the next update event. This function is only valid in output compare and PWM modes. To enable this function, you need to set OcxCE to 1.

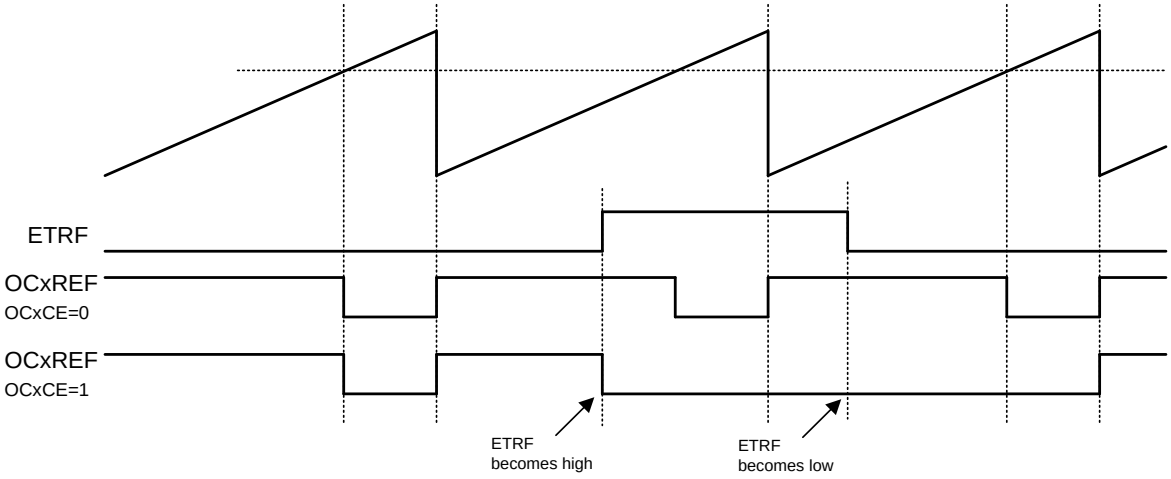


Figure 27-38 ETR signal clears OCxREF of ATIM

27.4.18 Encoder interface mode

The encoder interface mode involves two external input signals, and ATIM determines whether to increment or decrement the count value according to the edge of one signal relative to the level of the other signal. The following table shows the relationship between the counting method and the two input signals:

Valid edge	Corresponding signal level (TI1 corresponds to TI2, TI2 corresponds to TI1)	TI1 signal		TI2 signal	
		Up	Down	Up	Down
Only count at TI1	High	Decrease	Increase	Not counted	Not counted
	Down	Increase	Decrease	Not counted	Not counted
Only count at TI2	High	Not counted	Not counted	Increase	Decrease
	Down	Not counted	Not counted	Decrease	Increase
Count both at TI1 and TI2	High	Decrease	Increase	Increase	Decrease
	Down	Increase	Decrease	Decrease	Increase

Table 27-1 Encoder interface counting method

For example, when the counter counts with the TI1 signal as a clock, if the rising edge of TI1 is sampled until TI2 is high, the counter is decremented; if the falling edge of TI1 is sampled until TI2 is high, the counter is incremented.

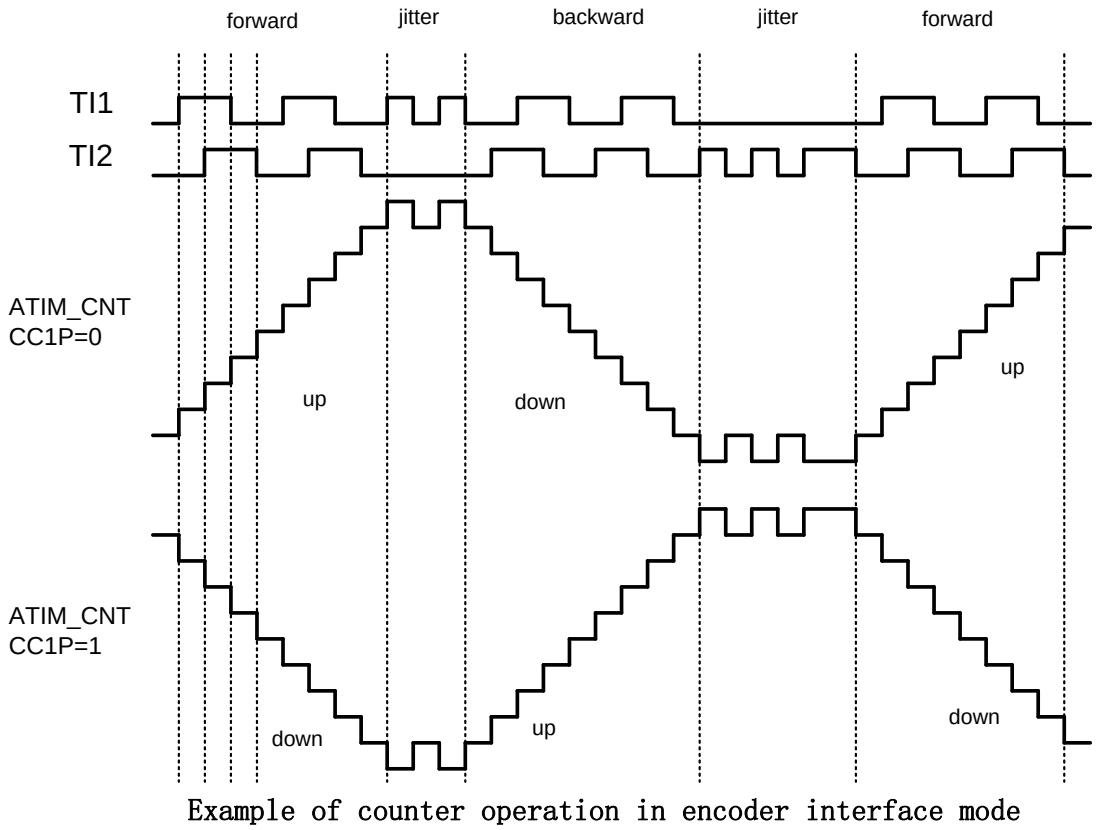


Figure 27-40 Example of counter operation in encoder mode

The encoding mode input channel needs to be set as follows:

- In the GPIO module, configure the corresponding pins as ATIM_CH1, ATIM_CH2 functions
- Turn off the channel enable, configure ATIM_CCER.CC1E=0, ATIM_CCER.CC2E=0, to ensure that the channel configuration is successful afterwards
- Select the input channel, configure ATIM_CCMR1.CC1S=01, ATIM_CCMR1.CC2S=01
- Select the effective edge of counting, configure ATIM_CCER.CC1P=0, ATIM_CCER.CC2P=0
- Set the slave mode controller to encoding mode 3, configure ATIM_SMCR.SMS[2:0]=011
- Turn on the channel enable, configure ATIM_CCER.CC1E=1, ATIM_CCER.CC2E=1

27.4.19 TIM slave mode

When ATIM is used as a slave (triggered by an external event), it can be configured into three working modes: multiple bit mode, gate control mode, and trigger mode.

Complex bit mode

In this mode, an external input event will cause all preload registers in the TIM to reinitialize, and CNT will return to 0 to start counting. Take the following figure as an example, the counter counts normally, and when the external TI1 input rising edge, the counter is cleared and restarted to count.

The configuration in the following figure is as follows:

- In the GPIO module, configure the corresponding pin as ATIM_CH1 function
- Turn off the channel enable, configure ATIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure ATIM_CCMR1.CC1S=01
- Select the effective edge of counting, configure ATIM_CCER.CC1P=0
- Select the trigger input signal, configure ATIM_SMCR.TS[2:0]=101, TI1FP1 as TRGI
- Set the slave mode controller to multiple bit mode, configure ATIM_SMCR.SMS[2:0]=100
- Turn on the channel enable, configure ATIM_CCER.CC1E=1
- Enable the counter, configure ATIM_CR1.CEN=1

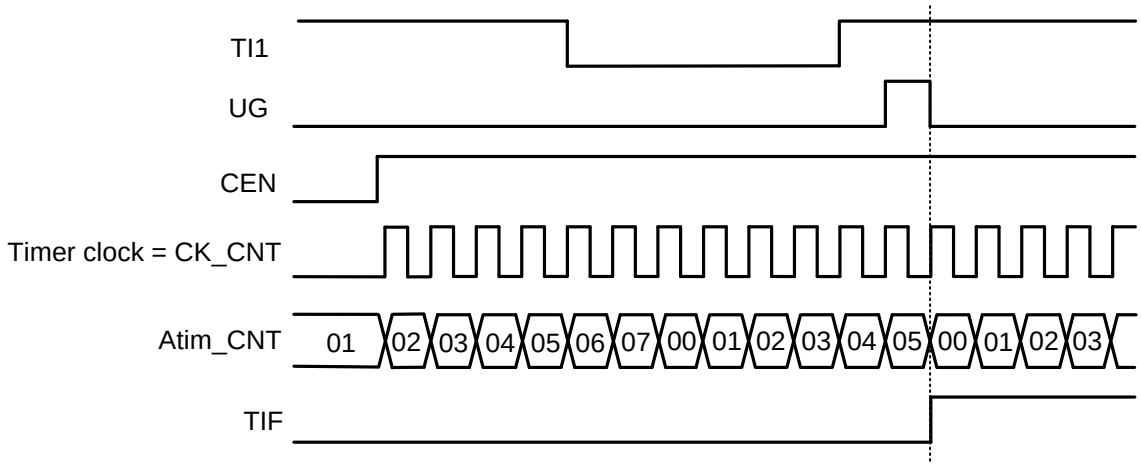


Figure 27-41 Timing in reset mode

Gating mode

In this mode, the counter only works when the input signal is at a specific level. When the level change causes the counter to start or stop counting, the interrupt flag is triggered.

The configuration in the following figure is as follows:

In the GPIO module, configure the corresponding pin as ATIM_CH1 function

Turn off the channel enable, configure ATIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards

Select the input channel, configure ATIM_CCMR1.CC1S=01

Select the effective edge of counting, configure ATIM_CCER.CC1P=0

Select the trigger input signal, configure ATIM_SMCR.TS[2:0]=101, TI1FP1 as TRGI

Set the slave mode controller to gated mode, configure ATIM_SMCR.SMS[2:0]=101

Turn on the channel enable, configure ATIM_CCER.CC1E=1

Enable the counter, configure ATIM_CR1.CEN=1

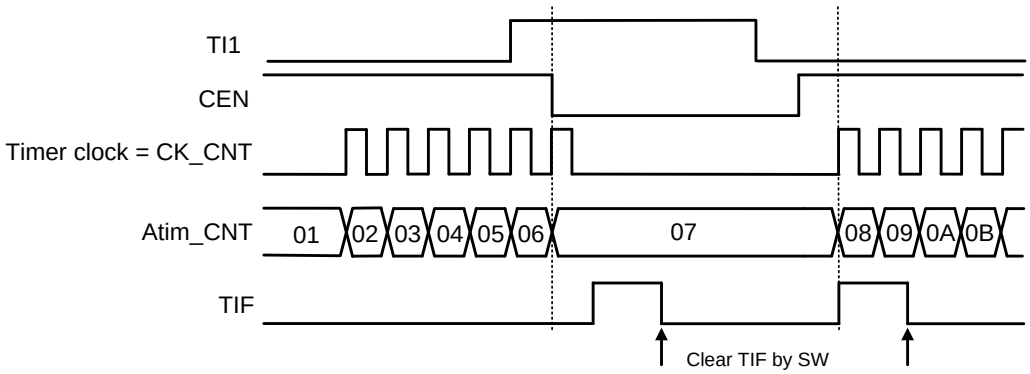


Figure 27-42 Timing in gated mode

Trigger mode

The counter only starts counting after an external input event arrives.

The configuration in the following figure is as follows:

- In the GPIO module, configure the corresponding pin as ATIM_CH1 function
- Turn off the channel enable, configure ATIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure ATIM_CCMR1.CC1S=01
- Select the effective edge of counting, configure ATIM_CCER.CC1P=0
- Select the trigger input signal, configure ATIM_SMCR.TS[2:0]=101, TI1FP1 as TRGI
- Set the slave mode controller to trigger mode, configure ATIM_SMCR.SMS[2:0]=110
- Turn on the channel enable, configure ATIM_CCER.CC1E=1

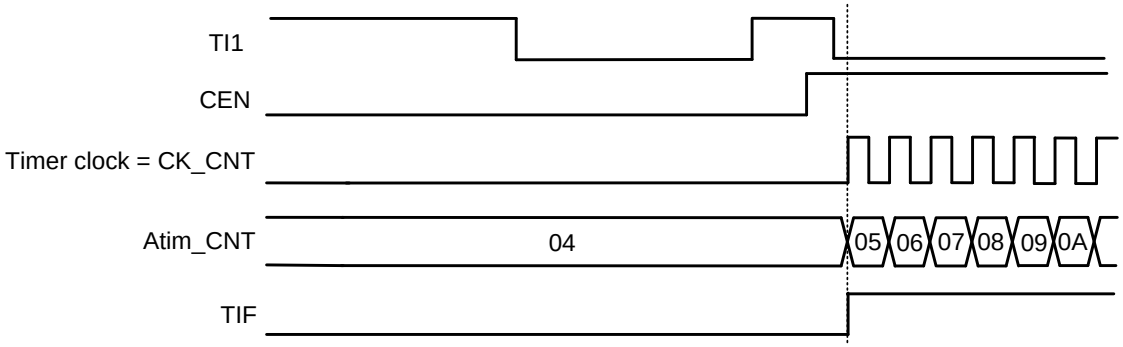


Figure 27-43 Timing in trigger mode

External clock counting mode triggered by external events

ETR can be set as the counting clock, while using another external input as the counter start trigger signal. For example, after detecting the rising edge of TI1, the counter starts counting with the rising edge of the ETR input.

The configuration in the following figure is as follows:

In the GPIO module, configure the corresponding pin as ATIM_CH1, ATIM_ETR function

Set ETP for edge selection, ATIM_SMCR.ETP=0

Set the ETR division ratio, configure ATIM_SMCR.ETPS[1:0]=01

Configure the input filter time, ATIM_SMCR.ETF[3:0]=0000

Set bitECE register, enable external clock mode 2, ATIM_SMCR.ECE=1

Turn off the channel enable, configure ATIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards

Select the input channel, configure ATIM_CCMR1.CC1S=01

Select the effective edge of counting, configure ATIM_CCER.CC1P=0

Select the trigger input signal, configure ATIM_SMCR.TS[2:0]=101, TI1FP1 as TRGI

Set the slave mode controller to trigger mode, configure ATIM_SMCR.SMS[2:0]=110

Turn on the channel enable, configure ATIM_CCER.CC1E=1

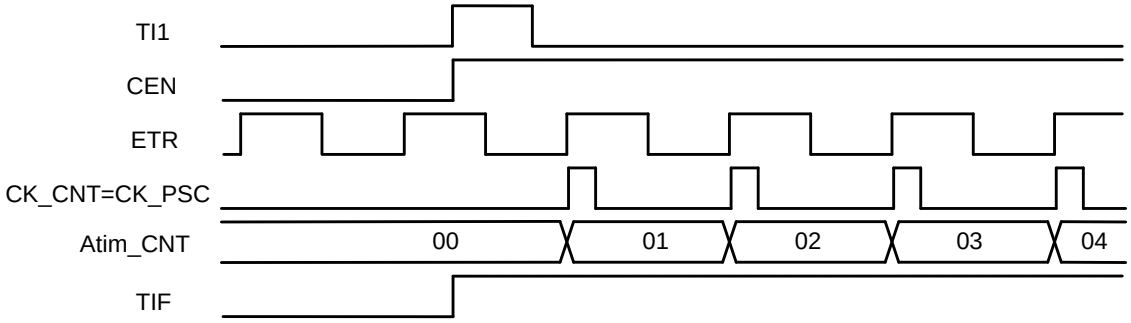


Figure 27-44 Timing in external clock mode 2 + trigger mode

27.4.20 DMA access

ATIM supports 7 types of DMA requests, which are 4 CC channel requests, external trigger requests, user software trigger requests and COM trigger requests.

Each CC channel generates a DMA request, which is used to transfer the content in CCRx to RAM in capture mode, and is used to write data in RAM to CCRx in compare mode; the DMA request of the CC channel can be configured as Single transfer or burst transfer (CCxBURSTEN), single transfer only accesses the CCRx register, and burst transfer accesses a specific set of registers according to the DCR register configuration.

In addition, external trigger events, software trigger events and COM events can also generate DMA requests. When these requests occur, DMA Burst transfer will be started, and data will be written to one or more registers in ATIM, or one will be read from ATIM. Or multiple register values.

DMA request	CCxBURSTEN	DMA.CHxCTRL.DIR	DMA access object	Length of one transfer
ATIM_CH1	0	0	Read CCR1	1
		1	Write CCR1	
	1	0	Read DMAR	DBL
		1	Write DMAR	
ATIM_CH2	0	0	Read CCR2	1
		1	Write CCR2	
	1	0	Read DMAR	DBL
		1	Write DMAR	
ATIM_CH3	0	0	Read CCR3	1
		1	Write CCR3	
	1	0	Read DMAR	DBL
		1	Write DMAR	
ATIM_CH4	0	0	Read CCR4	1
		1	Write CCR4	
	1	0	Read DMAR	DBL

DMA request	CCxBURSTEN	DMA.CHxCTRL.DIR	DMA access object	Length of one transfer
		1	Write DMAR	
ATIM_TRIG	N/A	0	Read DMAR	DBL
		1	Write DMAR	
ATIM_UEV	N/A	0	Read DMAR	DBL
		1	Write DMAR	
ATIM_COM	N/A	0	Read DMAR	DBL
		1	Write DMAR	

27.4.21 DMA Burst

ATIM supports DMA and DMA-Burst access. ATIM can be configured to trigger a DMA request when a specific event occurs, and the capture result in CCR can be written to RAM, or one or more registers can be written from RAM to the preload register of ATIM .

DMA-Burst supports one event to trigger multiple consecutive DMA requests. The main function is to continuously update the contents of multiple registers after the event occurs, so functions such as dynamic real-time adjustment of the output waveform can be realized.

The DMA controller needs to point the peripheral target address to a virtual register ATIM_DMAR. When a specific timer event occurs, ATIM will continuously transmit multiple DMA requests. Each DMA write operation to ATIM_DMAR will be redirected to the actual function register by ATIM.

The DBL register is used to set the DMA burst length, and the DBA register is used to set the base address of the DMA to access the ATIM (relative to the offset of ATIM_CR).

27.4.22 Input XOR function

The input signals of channels 1 to 3 can be XORed, and then connected to the filter and edge circuit input of channel 1 for input capture or triggering of channel 1.

The TI1Sbit of the ATIM_CR2 register is used to select whether the input of channel 1 comes from the exclusive OR of the input of the three channels

27.4.23 Debug mode

When Cortex-M0 enters the debug mode, the timer can stop or continue to work, and its behavior is defined by the DBG_TIMx_STOP register of the DCU module.

When the timer is stopped during Debug, its output will be disabled (MOE is cleared). According to the register configuration, the output signal at this time can be forced to inactive or controlled by the GPIO module. In DMA-Burst mode, all DMA accesses must point to the DMAR virtual register, and ATIM automatically accumulates the internal offset address according to the access. The DBA register is used to specify the target address of the first DMA transfer inside ATIM, and the DBL is used to specify the burst length.

27.5 Register

offset	name	symbol
ATIM(Module Start Address:0x4001B000)		
0x00000000	ATIM Control Register1	ATIM_CR1
0x00000004	ATIM Control Register2	ATIM_CR2
0x00000008	ATIM Slave Mode Control Register	ATIM_SMCR
0x0000000C	ATIM DMA and Interrupt Enable Register	ATIM_DIER
0x00000010	ATIM Interrupt Status Register	ATIM_ISR
0x00000014	ATIM Event Generation Register	ATIM_EGR
0x00000018	ATIM Capture/Compare Mode Register1	ATIM_CCMR1
0x0000001C	ATIM Capture/Compare Mode Register2	ATIM_CCMR2
0x00000020	ATIM Capture/Compare Enable Register	ATIM_CCER
0x00000024	ATIM Counter Register	ATIM_CNT
0x00000028	ATIM Prescaler Register	ATIM_PSC
0x0000002C	ATIM Auto-Reload Register	ATIM_ARR
0x00000030	ATIM Repetition Counter Register	ATIM_RCR
0x00000034	ATIM Capture/Compare Register1	ATIM_CCR1
0x00000038	ATIM Capture/Compare Register2	ATIM_CCR2
0x0000003C	ATIM Capture/Compare Register3	ATIM_CCR3
0x00000040	ATIM Capture/Compare Register4	ATIM_CCR4
0x00000044	ATIM Break and Deadtime Register	ATIM_BDTR
0x00000048	ATIM DMA Control Register	ATIM_DCR
0x0000004C	ATIM DMA Access Register	ATIM_DMAR
0x00000060	ATIM Break Control Register	ATIM_BKCR

27.5.1 ATIM Control register 1（ATIM_CR1）

NAME	ATIM_CR1							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						CKD	
access	U-0						R/W-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ARPE	CMS		DIR	OPM	URS	UDIS	CEN
access	R/W-0	R/W-00		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	description
31:10	-	RFU：**Reserved, read as 0**
9:8	**CKD**	Dead time and digital filter clock frequency divider register (division ratio relative to CK_INT) (Counter cloc Divider)

bit	name	description
		00：$t_{DTS}=t_{CK\_INT}$ 01：$t_{DTS}=2*t_{CK\_INT}$ 10：$t_{DTS}=4*t_{CK\_INT}$ 11：RFU，disable
7	**ARPE**	Auto-Reload Preload Enable 0: ARR register does not enable preload 1: ARR register enables preload
6:5	**CMS**	Counter Mode Selection 00: Edge alignment mode 01: Center-aligned mode 1, the output compare interrupt flag is only set when the counter is counting down 10: Center-aligned mode 2, the output compare interrupt flag is only set when the counter is counting up 11: Center-aligned mode 3, the output compare interrupt flag will be set when the counter is counting up and down
4	**DIR**	counter Direction 0: count up 1: count down Note: When the timer is configured in central counting mode or encoder mode, this register is read-only
3	**OPM**	One Pulse Mode 0: The counter does not stop when the Update Event occurs 1: The counter stops when the Update Event occurs (automatically clears CEN)
2	**URS**	Update Request Selection 0: The following events can generate an update interrupt or DMA request -Counter overflow or underflow -Software set bitUG register -The slave controller generates an update 1: Only counter overflow or underflow will generate update interrupt or DMA request
1	**UDIS**	Update Disable 0: enable the update event; update events are generated when the following events occur -Counter overflow or underflow -Software set bitUG register -The slave controller generates an update 1: The update event is forbidden, and the shadow register is not updated. When the UG sets the bit or the slave controller receives the hardware reset, the counter and prescaler are reinitialized.
0	**CEN**	Counter Enable 0: The counter is off 1: Counter enable Note: The external trigger mode can automatically set bitCEN

27.5.2 ATIM Control register 2（ATIM_CR2）

NAME	ATIM_CR2							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24

name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1
access	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TI1S	MMS			CCDS	CCUS	-	CCPC
access	R/W-0	R/W-000			R/W-0	R/W-0	U-0	R/W-0

bit	name	functional description
31:15	**-**	RFU：**Reserved, read as 0**
14	**OIS4**	Refer toOIS1
13	**OIS3N**	Refer toOIS1N
12	**OIS3**	Refer toOIS1
11	**OIS2N**	Refer toOIS1N
10	**OIS2**	Refer toOIS1
9	**OIS1N**	Output Idle State for OC1N 0: When MOE=0, after dead time, OC1N=0 1: When MOE=0, after dead time, OC1N=1
8	**OIS1**	Output Idle State for OC1 0: When MOE=0 (if complementary output is enabled, after dead time has passed), OC1=0 1: When MOE=0 (if complementary output is enabled, after dead time has passed), OC1=1
7	**TI1S**	Timer Input 1 Selection 0: ATIM_CH1 pin is connected to TI1 input 1: ATIM_CH1, CH2, CH3 pins are connected to TI1 input after XOR
6:4	**MMS**	Master mode selection, used to configure the source of the synchronous trigger signal (TRGO) sent to the slave in the master mode (Master Mode Selection) 000: The UG register of ATIM_EGR is used as TRGO 001: Counter enable signal CNT_EN is used as TRGO, which can be used to start multiple timers at the same time 010: UE (update event) signal is used as TRGO 011: comparison pulse, if the CC1IF flag is about to be set, TRGO outputs a positive pulse 100: OC1REF is used as TRGO 101: OC2REF is used as TRGO 110: OC3REF is used as TRGO 111: OC4REF is used as TRGO Note: The slave timer or ADC must enable the working clock in advance to receive the TRGO sent by the master timer
3	**CCDS**	Capture/Compare DMA Selection 0: Send a DMA request when a capture/compare event occurs 1: Send DMA request when Update Event occurs
2	**CCUS**	Capture/Compare Update Selection

bit	name	functional description
		0: When the capture/compare control register enables preload (CCPC=1), they are only updated when the bitCOMG register is set 1: When the capture/compare control register enables preload (CCPC=1), they are updated when the bitCOMG register is set or the TRGI rising edge
1	-	RFU：Reserved, read as 0
0	**CCPC**	Capture/Compare Preload Control enable 0: CcxE, CcxNE, OcxM registers do not enable preload 1: CcxE, CcxNE, OcxM register enable preload Note: This register is only valid on channels with complementary output functions

27.5.3 ATIM Slave mode control register（ATIM_SMCR）

NAME	ATIM_SMCR							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	ETP	ECE	ETPS		ETF			
access	R/W-0	R/W-0	R/W-00		R/W-0000			
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	MSM	TS			-	SMS		
access	R/W-0	R/W-000			U-0	R/W-000		

bit	name	functional description
31:16	-	RFU：**Reserved, read as 0**
15	**ETP**	External Trigger Polarity 0: High level or rising edge valid 1: Low level or falling edge valid
14	**ECE**	External Clock Enable 0: Turn off external clock mode 2 1: Enable external clock mode 2, the counter clock is the valid edge of ETRF
13:12	**ETPS**	External Trigger Prescaler The frequency of the external trigger signal ETRP can only be at most 1/4 of the ATIM working clock. When the input signal frequency is high, prescaler can be used. 00: no frequency division 01: divide by 2 10: 4 frequency division 11: 8 frequency division
11:8	**ETF**	External Trigger Filter 0000：No filtering 0001：$f_{SAMPLING}=f_{CK\_INT}$, N=2

bit	name	functional description
		0010：$f_{SAMPLING}=f_{CK\_INT}$, N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$, N=8 0100：$f_{SAMPLING}=f_{DTS/2}$, N=6 0101：$f_{SAMPLING}=f_{DTS/2}$, N=8 0110：$f_{SAMPLING}=f_{DTS/4}$, N=6 0111：$f_{SAMPLING}=f_{DTS/4}$, N=8 1000：$f_{SAMPLING}=f_{DTS/8}$, N=6 1001：$f_{SAMPLING}=f_{DTS/8}$, N=8 1010：$f_{SAMPLING}=f_{DTS/16}$, N=5 1011：$f_{SAMPLING}=f_{DTS/16}$, N=6 1100：$f_{SAMPLING}=f_{DTS/16}$, N=8 1101：$f_{SAMPLING}=f_{DTS/32}$, N=5 1110：$f_{SAMPLING}=f_{DTS/32}$, N=6 1111：$f_{SAMPLING}=f_{DTS/32}$, N=8
7	**MSM**	Master Slave Mode 0: No action 1: In trigger mode, the action triggered by TRGI is delayed, so that the current timer and slave timer can be synchronized through TRGO
6:4	**TS**	Trigger selection, used to select the trigger source of the synchronous counter (Trigger Source) 000: Internal trigger signal (ITR0) 001: Internal trigger signal (ITR1) 010: Internal trigger signal (ITR2) 011: Internal trigger signal (ITR3) 100: TI1 edge detection (TI1F_ED) 101: TI1 after filtering (TI1FP1) 110: TI2 after filtering (TI2FP2) 111: External trigger input (ETRF) Note: The TS register can be rewritten only when the SMS=000 means that the slave mode is disabled.
3	-	RFU：Reserved, read as 0
2:0	**SMS**	Slave Mode Selection 000: Slave mode is disabled; after CEN is enabled, the prescaler circuit clock source comes from the internal clock 001: Encoder mode 1; the counter uses TI2FP2 edge and counts according to the level of TI1 010: Encoder mode 2; the counter uses TI1FP1 edge and counts according to the level of TI2 011: Encoder mode 3; the counter uses TI1FP1 and TI2FP2 edges at the same time, and counts according to other input signal levels 100: Multiple bit mode; the rising edge of TRGI initializes the counter and triggers the register update 101: Gate mode; when TRGI is high, the counting clock is enabled, when TRGI is low, the counting clock is stopped 110: Trigger mode; TRGI rising edge triggers the counter to start counting (bit counter will not be reset) 111: External clock mode 1; the rising edge of TRGI directly drives the counter

27.5.4 ATIM DMA and interrupt enable register（ATIM_DIER）

NAME	ATIM_DIER							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-				CC4BURSTEN	CC3BURSTEN	CC2BURSTEN	CC1BURSTEN
access	U-0				R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE
access	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:20	-	RFU：**Reserved, read as 0**
19	**CC4BURSTEN**	CC4 Burst Enable 0: Single mode, only access CCR 1: Burst mode, configure access address and length through DCR
18	**CC3BURSTEN**	CC3 Burst Enable 0: Single mode, only access CCR 1: Burst mode, configure access address and length through DCR
17	**CC2BURSTEN**	CC2 Burst Enable 0: Single mode, only access CCR 1: Burst mode, configure access address and length through DCR
16	**CC1BURSTEN**	CC1 Burst Enable 0: Single mode, only access CCR 1: Burst mode, configure access address and length through DCR
15	-	RFU：Reserved, read as 0
14	**TDE**	Triggered DMA Enable 0: In slave mode, forbid external trigger event to generate DMA request 1: In slave mode, allow external trigger events to generate DMA requests (can be used to automatically update the preload register)
13	**COMDE**	COM event DMA Enable 0: When a COM event occurs, it is forbidden to generate DMA requests 1: When a COM event occurs, DMA requests are allowed
12	**CC4DE**	CC4 DMA Enable 0: Disable CC4 DMA request 1: Allow CC4 DMA request
11	**CC3DE**	CC3 DMA Enable 0: Disable CC3 DMA request 1: Allow CC3 DMA request

bit	name	functional description
10	**CC2DE**	CC2 DMA Enable 0: Disable CC2 DMA request 1: Allow CC2 DMA request
9	**CC1DE**	CC1 DMA Enable 0: Disable CC1 DMA request 1: Allow CC1 DMA request
8	**UDE**	Update Event DMA Enable 0: When Update Event occurs, DMA request is prohibited 1: When Update Event occurs, DMA request is allowed
7	**BIE**	Break event Interrupt Enable 0: Disable interruption of brake events 1: Allow interruption of brake events
6	**TIE**	Trigger event Interrupt Enable 0: Disable trigger event interrupt 1: Allow to trigger event interrupt
5	**COMIE**	COM event Interrupt Enable 0: Disable COM event interrupt 1: Allow COM event interrupt
4	**CC4IE**	CC4 Interrupt Enable 0: Disable capture/compare 4 interrupt 1: Allow capture/compare 4 interrupt
3	**CC3IE**	CC3 Interrupt Enable 0: Disable capture/compare 3 interrupt 1: Allow capture/compare 3 interrupt
2	**CC2IE**	CC2 Interrupt Enable 0: Disable capture/compare 2 interrupt 1: Enable capture/compare 2 interrupt
1	**CC1IE**	CC1 Interrupt Enable 0: Disable capture/compare 1 interrupt 1: Enable capture/compare 1 interrupt
0	**UIE**	Update event Interrupt Enable 0: Disable Update event interrupt 1: Allow Update event interrupt

27.5.5 ATIMS tatus register（ATIM_ISR）

NAME	ATIM_ISR							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			CC4OF	CC3OF	CC2OF	CC1OF	-
access	U-0			R/W-0	R/W-0	R/W-0	R/W-0	U-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:13	-	RFU：**Reserved, read as 0**
12	**CC4OF**	Over-Capture Interrupt Flag for CC4, write 1 to clear Refer toCC1OF
11	**CC3OF**	Over-Capture Interrupt Flag for CC3, write 1 to clear Refer toCC1OF
10	**CC2OF**	Over-Capture Interrupt Flag for CC2, write 1 to clear Refer toCC1OF
9	**CC1OF**	Over-Capture Interrupt Flag for CC1, write 1 to clear) This register is only valid when the corresponding channel is set to input capture mode. The hardware sets the bit, and the software writes 1 to clear it. 0: no overcapture event 1: A new capture occurs when the CC1IF flag is 1
8	-	RFU：Reserved, read as 0
7	**BIF**	Break Interrupt Flag, write 1 to clear
6	**TIF**	Trigger Interrupt Flag, write 1 to clear
5	**COMIF**	COM Interrupt Flag, write 1 to clear
4	**CC4IF**	CC4 Interrupt Flag, write 1 to clear Refer toCC1IF
3	**CC3IF**	CC3 Interrupt Flag, write 1 to clear Refer toCC3IF
2	**CC2IF**	CC2 Interrupt Flag, write 1 to clear Refer toCC2IF
1	**CC1IF**	CC1 Interrupt Flag, write 1 to clear If the CC1 channel is configured as an output: CC1IF is set when the count value is equal to the comparison value, and the software writes 1 to clear it. If the CC1 channel is configured as an input: bit is set when a capture event occurs, the software writes 1 to clear it, or the software reads ATIM_CCR1 to automatically clear it.
0	**UIF**	Update event Interrupt Flag, write 1 to clear When the following events occur, the UIF bit is set and the shadow register is updated -When the repetition ends and UDIS=0, the counter overflows -In the case of URS=0 and UDIS=0, the software sets the bitUG register to initialize the counter -In the case of URS=0 and UDIS=0, the trigger event initializes the counter

27.5.6 ATIM event generation register（ATIM_EGR）

NAME	ATIM_EGR							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							

access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
access	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

bit	name	functional description
31:8	-	RFU：**Reserved, read as 0**
7	**BG**	Software brake, software sets this register to generate a brake event, hardware automatically clears (Break Generate)
6	**TG**	Software trigger, software sets this register to generate a trigger event, hardware automatically clears (Trigger Interrupt Flag)
5	**COMG**	Software COM event, hardware set bit, software write 1 to clear (COMG Generate)
4	**CC4G**	Capture/compare channel 4 software trigger, Refer toCC1G (CC4 Generate)
3	**CC3G**	Capture/compare channel 3 software trigger, Refer toCC1G (CC3 Generate)
2	**CC2G**	Capture/compare channel 2 software trigger, Refer toCC1G (CC2 Generate)
1	**CC1G**	Capture/Compare Channel 1 Software Trigger (CC1 Generate) If the CC1 channel is configured as an output: CC1IF is set to bit, and the corresponding interrupt and DMA request can be generated when it is enabled If the CC1 channel is configured as an input: the current count value is captured to the ATIM_CCR1 register, CC1IF is set to bit, and the corresponding interrupt and DMA request can be generated when it is enabled
0	**UG**	Software Update event, software sets this register to generate Update event, hardware automatically clears (User Generate) When the software sets bitUG, it will reinitialize the counter and update the shadow register, and the prescaler counter will be cleared.

27.5.7 ATIM Capture/compare mode register1（ATIM_CCMR1）

This register is multiplexed into two different functions under output compare and input capture configuration.

NAME	ATIM_CCMR1							
offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	OC2CE	OC2M			OC2PE	OC2FE	CC2S	
	IC2F				IC2PSC		CC2S	
access	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

name	OC1CE	OC1M			OC1PE	OC1FE	CC1S
	IC1F				IC1PSC		CC1S
access	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-00

Output compare mode

bit	name	functional description
31:16	-	RFU：Reserved, read as 0
15	**OC2CE**	OC2 Clear Enable，Refer toOC1CE
14:12	**OC2M**	OC2 Mode，Refer toOC1M
11	**OC2PE**	OC2 Preload Enable，Refer toOC1PE
10	**OC2FE**	OC2 Fast Enable，Refer toOC1FE
9:8	**CC2S**	CC2 Channel Selection 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped to TI2 10: CC2 channel is configured as input, IC2 is mapped to TI1 11: CC2 channel is configured as input, IC2 is mapped to TRC Note: CC2S can only be written when the channel is closed (CC2E=0)
7	**OC1CE**	C1 Clear Enable 0: OC1REF is not affected by ETRF 1: OC1REF is automatically cleared when ETRF high level is detected
6:4	**OC1M**	Output compare 1 mode configuration, this register defines the behavior of the OC1REF signal (OC1 Mode) 000: The comparison result of the output compare register CCR1 and the counter CNT will not affect the output 001: When CCR1=CNT, set OC1REF high 010: When CCR1=CNT, set OC1REF low 011: When CCR1=CNT, flip OC1REF 100: OC1REF is fixed to low (inactive) 101: OC1REF is fixed to high (active) 110: PWM mode 1-When counting up, OC1REF is set high when CNT<CCR1, otherwise it is set low; when counting down, OC1REF is set low when CNT>CCR1, otherwise it is set high 111: PWM mode 2-When counting up, OC1REF is set low when CNT<CCR1, otherwise it is set high; when counting down, OC1REF is set high when CNT>CCR1, otherwise it is set low
3	**OC1PE**	OC1 Preload Enable 0: CCR1 preload register is invalid, CCR1 can be written directly 1: The CCR1 preload register is valid. The read and write operations for CCR1 are all access to the preload register, and the content of the preload register is transferred to the shadow register when an update event occurs
2	**OC1FE**	OC1 Fast Enable 0: Turn off the fast enable, the trigger input will not affect the comparison output 1: Turn on fast enable, the trigger input will immediately change OC1REF to the output when the comparison value matches, regardless of the current actual comparison situation (only used in trigger mode) This function is only valid when the current channel is configured in PWM1 or PWM2 mode
1:0	**CC1S**	CC1 Channel Selection 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped to TI1 10: CC1 channel is configured as input, IC1 is mapped to TI2 11: CC1 channel is configured as input, IC1 is mapped to TRC

bit	name	functional description
		Note: CC1S can only be written when the channel is closed (CC1E=0)

Input capture mode

bit	name	functional description
31:16	-	RFU：Reserved, read as 0
15:12	**IC2F**	IC2 Filter
11:10	**IC2PSC**	IC2 Prescaler
9:8	**CC2S**	CC2 Channel Selection 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC3 is mapped to TI2 10: CC2 channel is configured as input, IC3 is mapped to TI1 11: CC2 channel is configured as input, IC3 is mapped to TRC Note: CC2S can only be written when the channel is closed (CC2E=0)
7:4	**IC1F**	IC1 Filter This register defines the sampling frequency and filter length of TI1 0000: No filtering, sampling usingf_{DTS} 0001：$f_{SAMPLING}=f_{CK\_INT}$, N=2 0010：$f_{SAMPLING}=f_{CK\_INT}$, N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$, N=8 0100：$f_{SAMPLING}=f_{DTS/2}$, N=6 0101：$f_{SAMPLING}=f_{DTS/2}$, N=8 0110：$f_{SAMPLING}=f_{DTS/4}$, N=6 0111：$f_{SAMPLING}=f_{DTS/4}$, N=8 1000：$f_{SAMPLING}=f_{DTS/8}$, N=6 1001：$f_{SAMPLING}=f_{DTS/8}$, N=8 1010：$f_{SAMPLING}=f_{DTS/16}$, N=5 1011：$f_{SAMPLING}=f_{DTS/16}$, N=6 1100：$f_{SAMPLING}=f_{DTS/16}$, N=8 1101：$f_{SAMPLING}=f_{DTS/32}$, N=5 1110：$f_{SAMPLING}=f_{DTS/32}$, N=6 1111：$f_{SAMPLING}=f_{DTS/32}$, N=8
3:2	**IC1PSC**	IC1 Prescaler 00: no frequency division 01: Capture once every 2 event inputs 10: A capture is generated every 4 event inputs 11: A capture is generated every 8 event inputs IC1PSC register is reset when CC1E=0
1:0	**CC1S**	CC1 Channel Selection 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped to TI1 10: CC1 channel is configured as input, IC1 is mapped to TI2 11: CC1 channel is configured as input, IC1 is mapped to TRC Note: CC1S can only be written when the channel is closed (CC1E=0)

27.5.8 ATIM Capture/compare mode register2（ATIM_CCMR2）

This register is multiplexed into two different functions under output compare and input capture configuration.

NAME	ATIM_CCMR2							
Offset	0x0000001C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	OC4CE	OC4M			OC4PE	OC4FE	CC4S	
	IC4F				IC4PSC		CC4S	
access	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	OC3CE	OC3M			OC3PE	OC3FE	CC3S	
	IC3F				IC3PSC		CC3S	
access	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-00	

Output compare mode

bit	name	functional description
31:16	-	RFU：**Reserved, read as 0**
15	**OC4CE**	OC4 Clear Enable，Refer toOC1CE
14:12	**OC4M**	OC4 Mode，Refer toOC1M
11	**OC4PE**	OC4 Preload Enable，Refer toOC1PE
10	**OC4FE**	OC4 Fast Enable，Refer toOC1FE
9:8	**CC4S**	CC4 Channel Selection 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped to TI4 10: CC4 channel is configured as input, IC4 is mapped to TI3 11: CC4 channel is configured as input, IC4 is mapped to TRC Note: CC4S can only be written when the channel is closed (CC4E=0)
7	**OC3CE**	OC3 Clear Enable 0: OC1REF is not affected by ETRF 1: OC1REF is automatically cleared when ETRF high level is detected
6:4	**OC3M**	Output compare 3 mode configuration, this register defines the behavior of the OC3REF signal (OC3 Mode) 000: The comparison result of the output compare register CCR3 and the counter CNT will not affect the output 001: When CCR3=CNT, set OC1REF high 010: When CCR3=CNT, set OC1REF low 011: When CCR3=CNT, flip OC1REF 100: OC3REF is fixed to low (inactive) 101: OC3REF is fixed to high (active)

bit	name	functional description
		110: PWM mode 1-when counting up, OC3REF is set high when CNT<CCR3, otherwise it is set low; when counting down, OC3REF is set low when CNT>CCR3, otherwise it is set high 111: PWM mode 2-When counting up, OC3REF is set low when CNT<CCR3, otherwise it is set high; when counting down, OC3REF is set high when CNT>CCR3, otherwise it is set low
3	**OC3PE**	OC3 Preload Enable 0: CCR3 preload register is invalid, CCR3 can be written directly 1: The CCR3 preload register is valid. The read and write operations for CCR3 all access the preload register, and the content of the preload register is transferred to the shadow register when the update event occurs
2	**OC3FE**	OC3 Fast Enable 0: Turn off the fast enable, the trigger input will not affect the comparison output 1: Turn on fast enable, the trigger input will immediately change OC3REF to the output when the comparison value matches, regardless of the current actual comparison situation This function is only valid when the current channel is configured in PWM1 or PWM2 mode
1:0	**CC3S**	CC4 Channel Selection 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC1 is mapped to TI3 10: CC3 channel is configured as input, IC1 is mapped to TI4 11: CC3 channel is configured as input, IC1 is mapped to TRC Note: CC3S can only be written when the channel is closed (CC3E=0)

Input capture mode

bit	name	functional description
31:16	**-**	RFU：Reserved, read as 0
15:12	**IC4F**	IC4 Filter
11:10	**IC4PSC**	IC4 Prescaler
9:8	**CC4S**	CC4 channel Selection 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped to TI4 10: CC4 channel is configured as input, IC4 is mapped to TI3 11: CC4 channel is configured as input, IC4 is mapped to TRC Note: CC4S can only be written when the channel is closed (CC4E=0)
7:4	**IC3F**	IC3 Filter This register defines the sampling frequency and filter length of TI3 0000：No filtering, sampling usingf_{DTS} 0001：$f_{SAMPLING}=f_{CK\_INT}$, N=2 0010：$f_{SAMPLING}=f_{CK\_INT}$, N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$, N=8 0100：$f_{SAMPLING}=f_{DTS/2}$, N=6 0101：$f_{SAMPLING}=f_{DTS/2}$, N=8 0110：$f_{SAMPLING}=f_{DTS/4}$, N=6 0111：$f_{SAMPLING}=f_{DTS/4}$, N=8

bit	name	functional description
		1000：$f_{SAMPLING}=f_{DTS/8}$, N=6 1001：$f_{SAMPLING}=f_{DTS/8}$, N=8 1010：$f_{SAMPLING}=f_{DTS/16}$, N=5 1011：$f_{SAMPLING}=f_{DTS/16}$, N=6 1100：$f_{SAMPLING}=f_{DTS/16}$, N=8 1101：$f_{SAMPLING}=f_{DTS/32}$, N=5 1110：$f_{SAMPLING}=f_{DTS/32}$, N=6 1111：$f_{SAMPLING}=f_{DTS/32}$, N=8
3:2	**IC3PSC**	IC3 Prescaler 00: no frequency division 01: Capture once every 2 event inputs 10: A capture is generated every 4 event inputs 11: A capture is generated every 8 event inputs IC1PSC register is reset when CC1E=0
1:0	**CC3S**	CC3 channel Selection 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC1 is mapped to TI3 10: CC3 channel is configured as input, IC1 is mapped to TI4 11: CC3 channel is configured as input, IC1 is mapped to TRC Note: CC1S can only be written when the channel is closed (CC1E=0)

27.5.9 ATIM Capture/compare enable register（ATIM_CCER）

NAME	**ATIM_CCER**							
Offset	0x00000020							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-		CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E
access	U-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	**name**	**functional description**
31:14	**-**	RFU：Reserved, read as 0
13	**CC4P**	CC4 Polarity，Refer toCC1P
12	**CC4E**	CC4 Enable，Refer toCC1E
11	**CC3NP**	CC3N Polarity，Refer toCC1NP
10	**CC3NE**	CC3N Enable，Refer toCC1NE
9	**CC3P**	CC3 Polarity，Refer toCC1P
8	**CC3E**	CC3 Enable，Refer toCC1E

bit	name	functional description
7	**CC2NP**	CC2N Polarity，Refer toCC1NP
6	**CC2NE**	CC2N Enable，Refer toCC1NE
5	**CC2P**	CC2 Polarity，Refer toCC1P
4	**CC2E**	CC2 Enable，Refer toCC1E
3	**CC1NP**	CC1N Polarity (CC1N Polarity) 0: OC1N high level is active 1: OC1N low level is active
2	**CC1NE**	CC1N Enable (CC1N Enable) 0: OC1N is invalid, OC1N level is determined by MOE, OSSI, OSSR, OIS1, OIS1N, CC1E registers
1	**CC1P**	CC1 Polarity When CC1 channel is configured as output 0: OC1 high level active 1: OC1 low level active When CC1 channel is configured as input 0: Non-inverted mode-capture is performed on the rising edge of IC1 1: Inversion mode-capture is performed on the falling edge of IC1
0	**CC1E**	CC1 Enable When CC1 channel is configured as output 0: OC1 is not active 1: OC1 active When CC1 channel is configured as input 0: Turn off the capture function 1: Enable the capture function

27.5.10 ATIM Counter register（ATIM_CNT）

NAME	ATIM_CNT							
Offset	0x00000024							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CNT[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CNT[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU：**Reserved, read as 0**
15:0	**CNT**	Counter

27.5.11 ATIM Prescaler register（ATIM_PSC）

NAME	ATIM_PSC							
Offset	0x00000028							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	PSC[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	PSC[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU：Reserved, read as 0
15:0	**PSC**	Counter clock (CK_CNT) prescaler value (Prescaler)$f_{CK\_CNT}=f_{CK\_PSC}/(PSC[15:0]+1)$ This is a preload register, and its content is loaded into the shadow register when the update event occurs

27.5.12 ATIM Auto-reload register（ATIM_ARR）

NAME	ATIM_ARR							
offset	0x0000002C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	ARR[15:8]							
access	R/W-1111 1111							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ARR[7:0]							
access	R/W-1111 1111							

bit	name	functional description
31:16	-	RFU：**Reserved, read as 0**
15:0	**ARR**	Auto-Reload Register This is a preload register, and its content is loaded into the shadow register when the update event occurs

27.5.13 ATIM Repeat count register（ATIM_RCR）

NAME	ATIM_RCR							
Offset	0x00000030							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	REP[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:8	-	RFU：**Reserved, read as 0**
7:0	**REP**	Repetition When REP is not 0, REP is decremented every time the update condition occurs, and the update event is triggered when REP=0

27.5.14 ATIM Capture/compare register1（ATIM_CCR1）

NAME	ATIM_CCR1							
Offset	0x00000034							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CCR1[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CCR1[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU：**Reserved, read as 0**
15:0	**CCR1**	Capture/Compare channel 1 Register **If channel 1 is configured as output:** This is a preload register whose content is loaded into the shadow register and used to compare with the counter to generate OC1 output

bit	name	functional description
		If channel 1 is configured as input: CCR1 saves the counter value when the most recent input capture event occurred, at this time CCR1 is read-only

27.5.15 ATIM Capture/compare register2（ATIM_CCR2）

NAME	ATIM_CCR2							
Offset	0x00000038							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CCR2[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CCR2[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU：**Reserved, read as 0**
15:0	**CCR2**	Capture/Compare channel 2 Register **If channel 2 is configured as output:** This is a preload register whose content is loaded into the shadow register and used to compare with the counter to generate OC2 output **If channel 2 is configured as input:** CCR2 saves the counter value when the most recent input capture event occurred. At this time, CCR2 is read-only

27.5.16 ATIM Capture/compare register3（ATIM_CCR3）

NAME	ATIM_CCR3							
Offset	0x0000003C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CCR3[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CCR3[7:0]							

access	R/W-0000 0000

bit	name	functional description
31:16	-	RFU：**Reserved, read as 0**
15:0	**CCR3**	Capture/Compare channel 3 Register **If channel 3 is configured as output:** This is a preload register whose content is loaded into the shadow register and used to compare with the counter to generate OC3 output **If channel 3 is configured as input:** CCR3 saves the counter value when the most recent input capture event occurred. At this time, CCR3 is read-only

27.5.17 ATIM Capture/compare register4（ATIM_CCR4）

NAME	ATIM_CCR4							
Offset	0x00000040							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CCR4[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CCR4[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU：**Reserved, read as 0**
15:0	**CCR4**	Capture/Compare channel 4 Register **If channel 4 is configured as output:** This is a preload register whose content is loaded into the shadow register and used to compare with the counter to generate OC4 output **If channel 4 is configured as input:** CCR4 saves the counter value when the most recent input capture event occurred. At this time, CCR4 is read-only

27.5.18 ATIM Brake and dead zone control register（ATIM_BDTR）

NAME	ATIM_BDTR							
Offset	0x00000044							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16

name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK	
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DTG							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU：**Reserved, read as 0**
15	**MOE**	Master Output Enable This register controls the output enable of all channels, and the independent output enable of each channel also needs to be controlled by CcxE and CcxNE. The MOE bit is set by software, or the bit is automatically set by hardware trigger when AOE=1. When the brake input is valid, MOE is asynchronously cleared by hardware. 0: Turn off OC and OCN output, the specific IO output status is determined by OSSI 1: Enable OC and OCN output (still need the CcxE and CcxNE status of each channel to determine whether to output)
14	**AOE**	Automatic Output Enable 0: MOE can only be set by software 1: MOE can be set by software or automatically set by the update event
13	**BKP**	Break Polarity 0: The brake input is active low 1: The brake input is active at high level
12	**BKE**	Break Enable 0: prohibit brake input 1: Allow brake input
11	**OSSR**	Off-State Select in Run mode Only when MOE=1, it is valid for the channel with complementary output enabled. 0: When the output channel is not enabled, OC and OCN do not drive GPIO 1: When the output channel is not enabled, OC and OCN drive GPIO to be in an invalid state
10	**OSSI**	Off-State Select in IDLE mode It is valid for the output channel only when MOE=0. 0: When the output channel is not enabled, OC and OCN do not drive GPIO 1: When the output channel is not enabled, OC and OCN drive the idle state first, and start the invalid state after the dead time expires
9:8	**LOCK**	Register write LOCK 00: No write protection 01: Protection level 1-DTG, OISx, OISxN, BKE, BKP, AOE cannot be overwritten 10: Protection level 2-On the basis of level 1, CCxP, CCxNP, OSSR, OSSI cannot be rewritten 11: Protection level 3-On the basis of level 2, OcxM, OcxPE

bit	name	functional description
		cannot be rewritten when the corresponding channel is configured as an output Note: The LOCK register cannot be rewritten after being written to a value other than 00. The write-protected register can only be rewritten after the ATIM module is reset.
7:0	**DTG**	Dead time insertion, used to configure the dead time length of complementary output insertion (Dead Time Generation) 000/001/010/011：DT=DTG[7:0] * t_{DTS} 100/101：DT=(64+DTG[5:0]) * 2 * t_{DTS} 110：DT=(32+DTG[4:0]) * 8 * t_{DTS} 111：DT=(32+DTG[4:0]) * 16 * t_{DTS}

27.5.19 ATIM DMA Control register（ATIM_DCR）

NAME	ATIM_DCR							
Offset	0x00000048							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			DBL				
access	U-0			R/W-0 0000				
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-			DBA				
access	U-0			R/W-0 0000				

bit	name	functional description
31:13	-	RFU：**Reserved, read as 0**
12:8	**DBL**	DMA Burst Length Reading and writing to the ATIM_DMAR register will trigger the burst DMA operation, the burst length is 1~18 00000: length=1 00001: length=2 00010: length=3 00011: length=4 00100: length=5 00101: length=6 00110: length=7 00111: length=8 01000: length=9 01001: length=10 01010: length=11 01011: length=12 01100: length=13 01101: length=14 01110: length=15

bit	name	functional description
		01111: length=16 10000: length=17 10001: length=18 Other: invalid value, write prohibited
7:5	-	RFU：**Reserved, read as 0**
4:0	**DBA**	DMA Burst Address 00000：ATIM_CR1 00001：ATIM_CR2 00010：ATIM_SMCR …… Note: When DBA+DBL exceeds the address range of the ATIM register, the actual burst will automatically stop after being transferred to the highest register address of the ATIM, that is, the burst length will be shortened.

27.5.20 ATIM DMA access register（ATIM_DMAR）

NAME	ATIM_DMAR							
Offset	0x0000004C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	DMAR[31:24]							
access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	DMAR[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DMAR[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DMAR[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:0	**DMAR**	DMA burst access Register When using DMA burst transfer, set the DMA channel peripheral address to ATIM_DMAR, ATIM will generate multiple DMA requests according to the value of DBL

27.5.21 ATIM Brake input control register（ATIM_BKCR）

NAME	ATIM_BKCR							
Offset	0x00000060							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						BRK2GATE	BRK1GATE
access	U-0						R/W-1	R/W-1
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	BRKF				BRKCOMB	HFDET_BRKEN	SVD_BRKEN	COMP_BRKEN
access	R/W-0000				R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:10	-	RFU：**Reserved, read as 0**
9	**BRK2GATE**	ATIM_BRK2 pin input gate control signal (Break 2 Gate) 0: Gate the input of ATIM_BRK2 to 0 1: Not gated

bit	name	functional description
8	**BRK1GATE**	ATIM_BRK1 pin input gate control signal (Break 1 Gate) 0: Gate the input of ATIM_BRK1 to 0 1: Not gated
7:4	**BRKF**	Filter clock and length selection of brake signal (Break Filter) 0000: No filtering 0001：$f_{SAMPLING}=f_{CK\_INT}$, N=2 0010：$f_{SAMPLING}=f_{CK\_INT}$, N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$, N=8 0100：$f_{SAMPLING}=f_{DTS/2}$, N=6 0101：$f_{SAMPLING}=f_{DTS/2}$, N=8 0110：$f_{SAMPLING}=f_{DTS/4}$, N=6 0111：$f_{SAMPLING}=f_{DTS/4}$, N=8 1000：$f_{SAMPLING}=f_{DTS/8}$, N=6 1001：$f_{SAMPLING}=f_{DTS/8}$, N=8 1010：$f_{SAMPLING}=f_{DTS/16}$, N=5 1011：$f_{SAMPLING}=f_{DTS/16}$, N=6 1100：$f_{SAMPLING}=f_{DTS/16}$, N=8 1101：$f_{SAMPLING}=f_{DTS/32}$, N=5 1110：$f_{SAMPLING}=f_{DTS/32}$, N=6 1111：$f_{SAMPLING}=f_{DTS/32}$, N=8
3	**BRKCOMB**	Break Combination 0: Two brake signals phase OR 1: The two brake signals are in phase and
2	**HFDET_BRKEN**	HFDET Break Enable 0: Disable HFDET brake signal 1: Enable HFDET brake signal
1	**SVD_BRKEN**	SVD Break Enable 0: Prohibit SVD brake signal 1: Enable SVD brake signal
0	**COMP_BRKEN**	Comparator Break Enable 0: Prohibit the brake signal of the comparator 1: Enable comparator brake signal

28 General Timer Array (GPTIM)

28.1 Introduction

The FM33LE0xxA contains two advanced timers.

The advanced timer includes a 16bit automatic overload counter and a programmable prescaler.

Advanced timers can support a variety of applications, including capture, output comparison, PWM.

28.2 Main characteristics

- 16bit up, down, two-way counting automatic reload counter
- 16bit programmable prescaler, support real-time adjustment of counting clock frequency division
- Flexible counting clock source selection, use part of the clock to run in sleep mode
- 4 independent channels can be used for input capture, output comparison, PWM (edge or center aligned mode), single pulse output
- Support cascading with other timers
- Support to generate interrupts when the following events occur
  - Counter overflow, counter initialization (software or hardware trigger)
  - Trigger event (counter start, stop, initialization, internal and external trigger)
  - Input capture
  - Output comparison

28.3 Block diagram

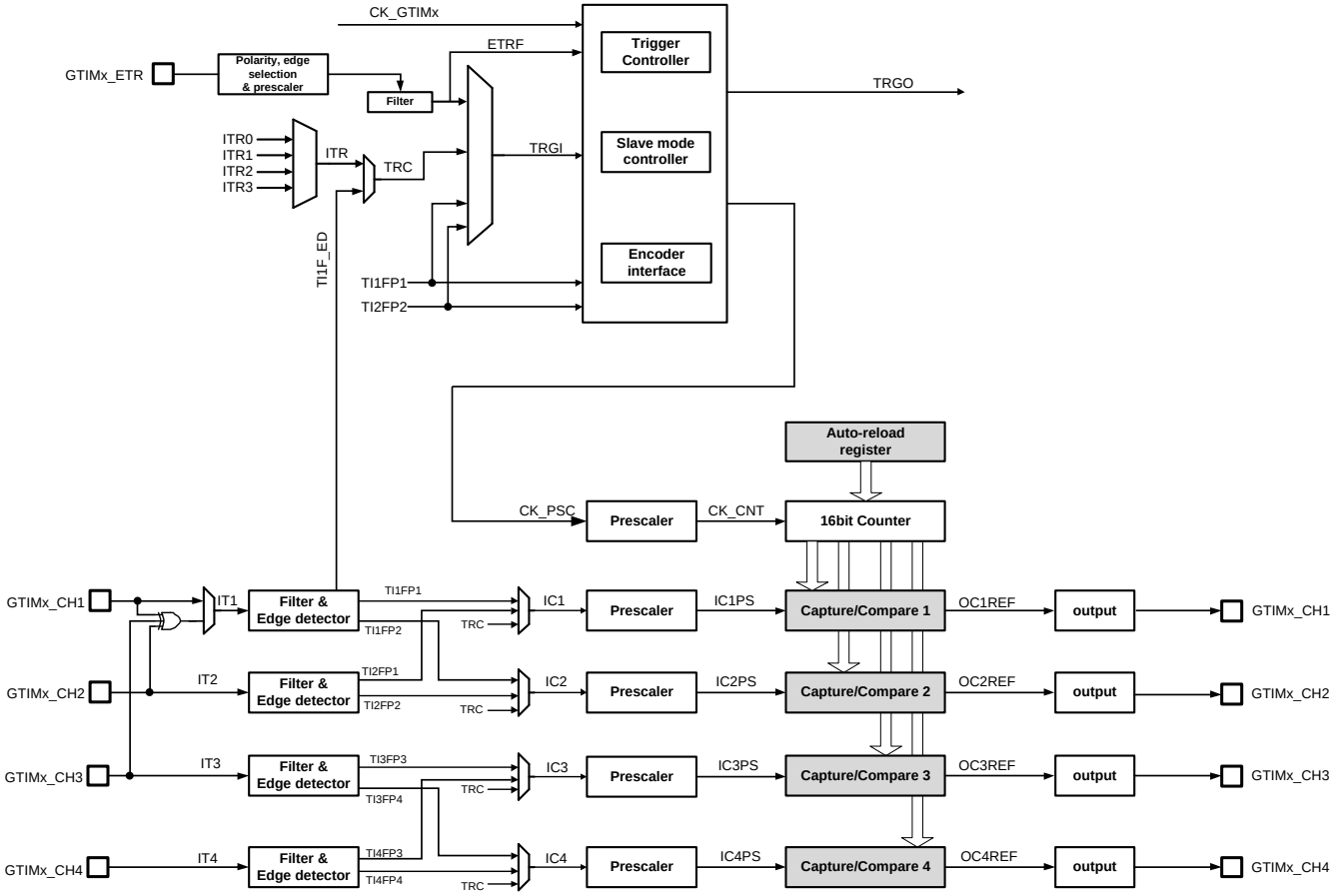


Figure 28-1 GPTIM Block Diagram

28.4 Function description

28.4.1 Timing unit

The timing unit of the advanced timer consists of a 16-bit counter and auto-reload register. The counter can count up, down or bidirectionally. The count clock can be obtained after dividing the APBCLK by a 16-bit prescaler.

The counter, auto-reload register and prescaler register can all be rewritten or read by software, even when the counter is running.

The timing unit contains the following registers:

- Counter (GPTIM_CNT)
- Prescaler register (GPTIM_PSC)
- Automatic reload register (GPTIM_ARR)
- Repeat count register (GPTIM_RCR)

ARR includes a preload function, which is controlled by the ARPE (Auto Reload Preload Enable) register. When ARPE=0, the ARR register is written, and the written data will be directly transferred to the shadow register; when ARPE=1, the data written to the ARR register occurs in the update event (GPTIM_CNT overflow or underflow) When, transfer to the shadow register. Software can also actively trigger ARR update (UEV) through register operations.

The GPTIM_CNT working clock is driven by the frequency division clock generated by GPTIM_PSC. The CNT only starts counting when the counter enable register (CEN) is set. When CNT=ARR, this round of counting ends, and an update event is sent.

GPTIM_PSC is a synchronous prescaler that can divide APBCLK from 1 to 65536. The PSC register is also cached. Rewriting the PSC does not actually rewrite the shadow register. Only when a new update event comes will it be updated from the PSC to the shadow register. Therefore, during the CNT counting process, the software can rewrite the PSC in real time, and the new prescaler ratio will be adopted when the next update event occurs.

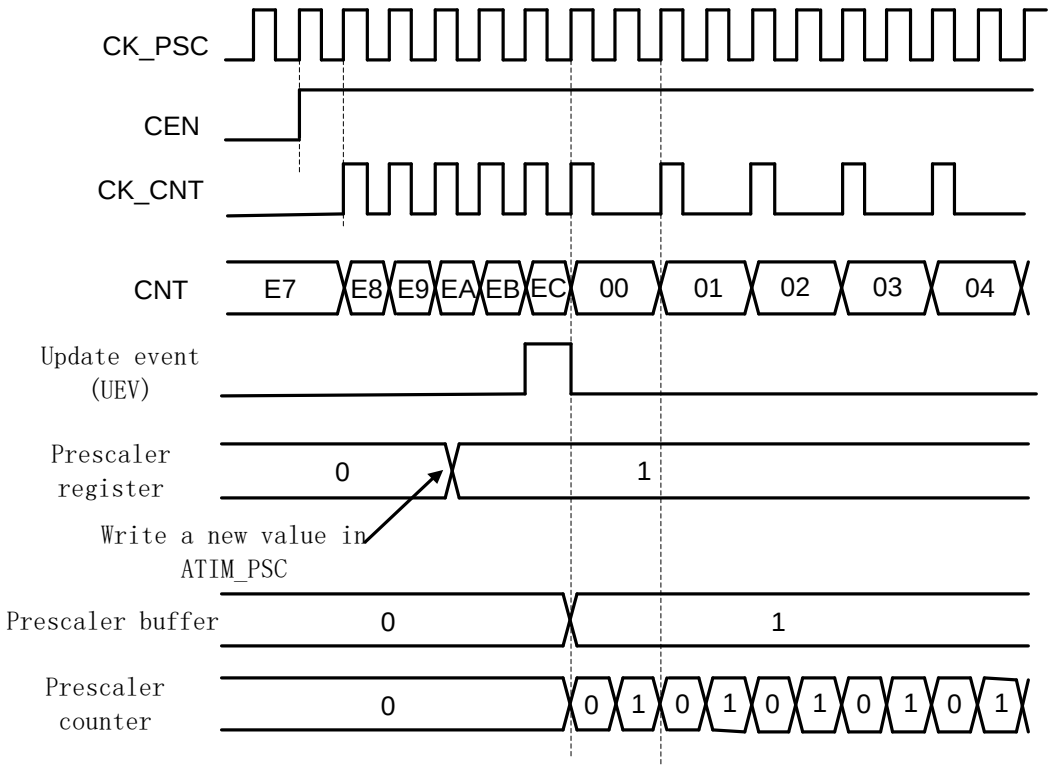


Figure 28-2 Waveform of prescaling from 1 to 2

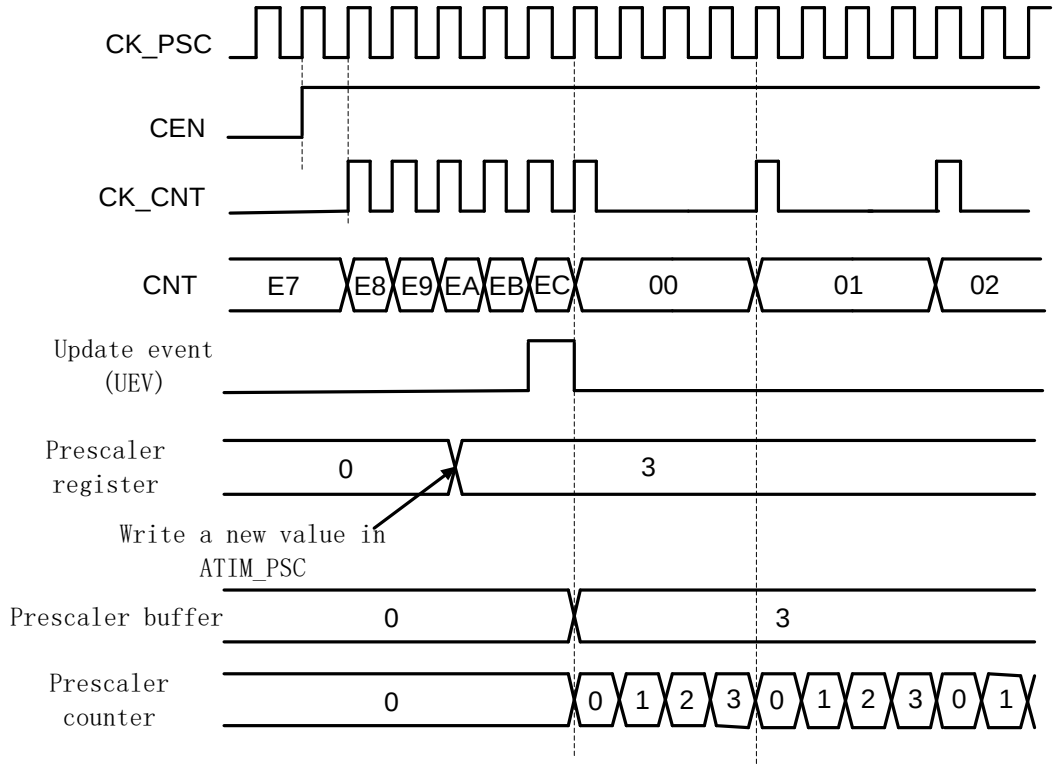


Figure 28-3 Waveform of prescaling from 1 to 4

28.4.2 Timer operating mode

The timer supports up-counting, down-counting and center-counting modes.

Count up

In this mode, the counter starts counting from 0 after being enabled, until CNT=ARR, an overflow event is generated, and then starts counting from 0 again.

If the repeat counting function is enabled, the counter repeats the above process several times (RCR+1) according to the definition of RCR before an overflow event will be generated.

The software can directly trigger the update event by setting the UG register, and the CNT and prescaler counter are automatically cleared at this time. Whether the setting of the UG register triggers the UIF (Update Interrupt Flag) interrupt flag setting is determined by the setting of the URS register.

The update event can be disabled by setting the UDIS register, which can avoid updating the value in the preload register to the working register.

When an update event occurs, the following registers are updated and UIF is set:

- The RCR shadow register is updated to the contents of ATIM_RCR
- The ARR shadow register is updated to ATIM_ARR content
- PSC shadow register is updated to ATIM_PSC content

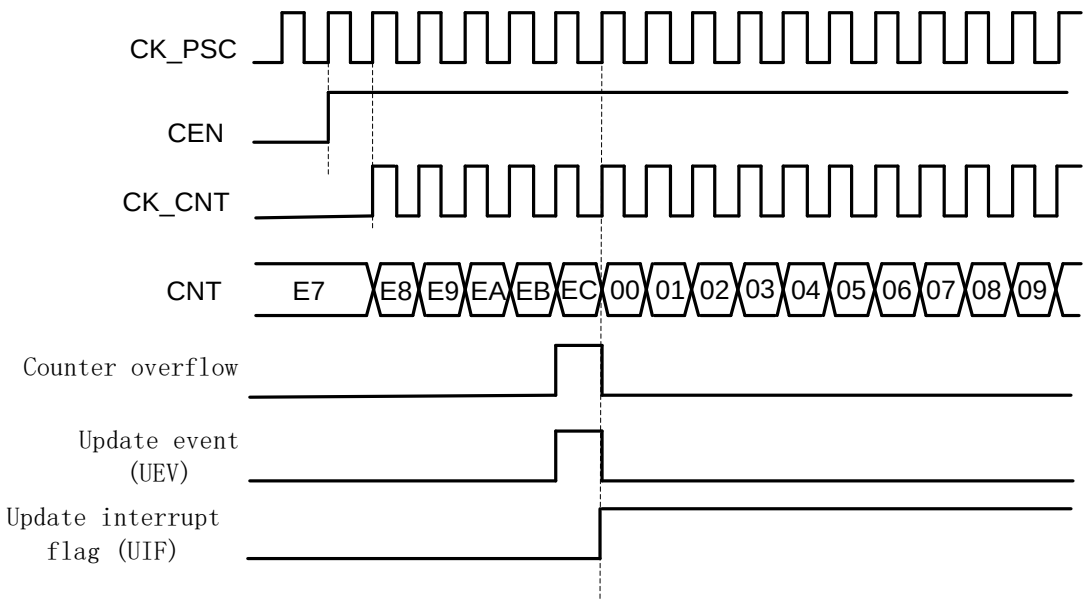


Figure 28-3 Upward counting waveform, internal clock not divided

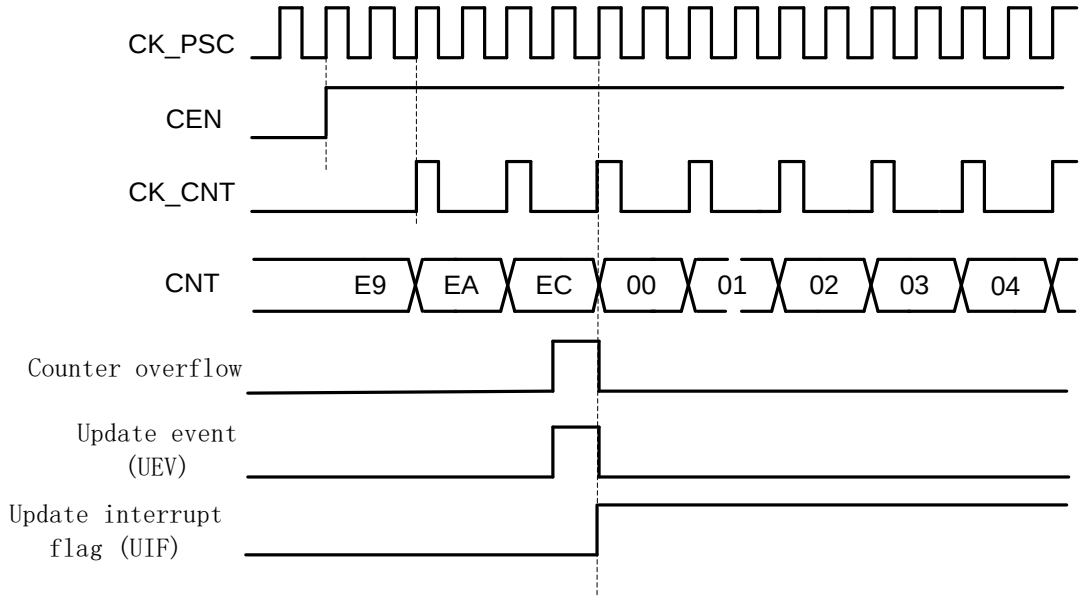


Figure 28-4 Upward counting waveform, internal clock divided-by-2

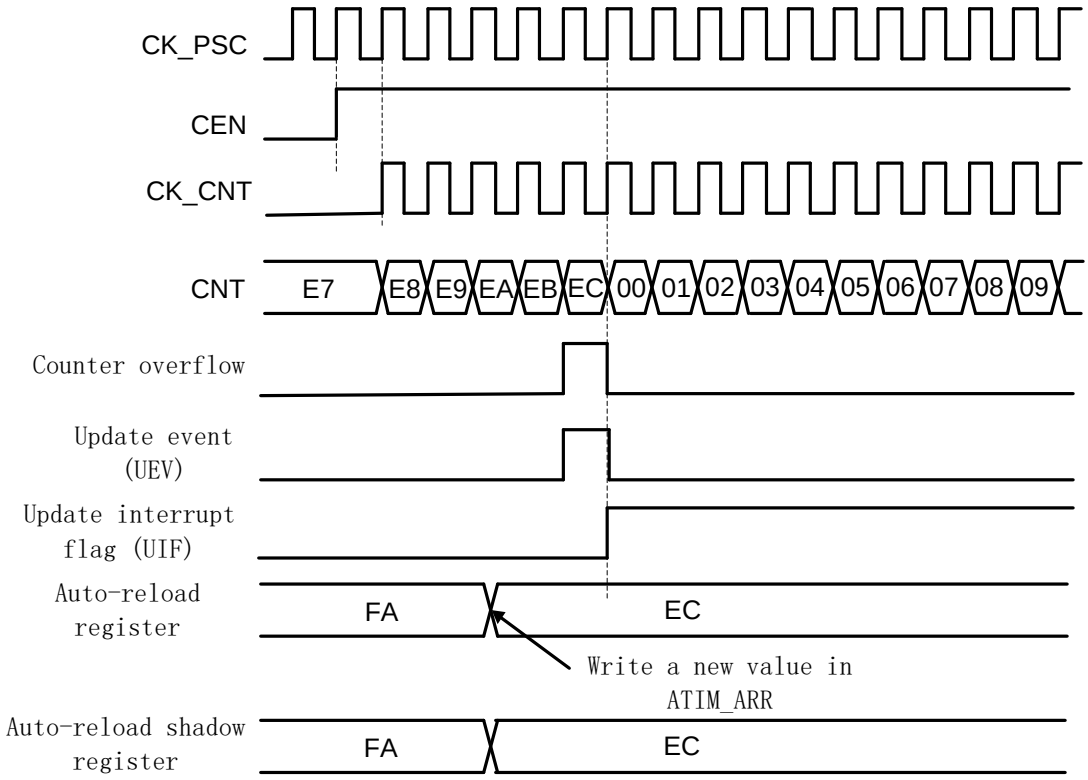


Figure 28-5 Update event when ARPE=0 (ARR is not preloaded)

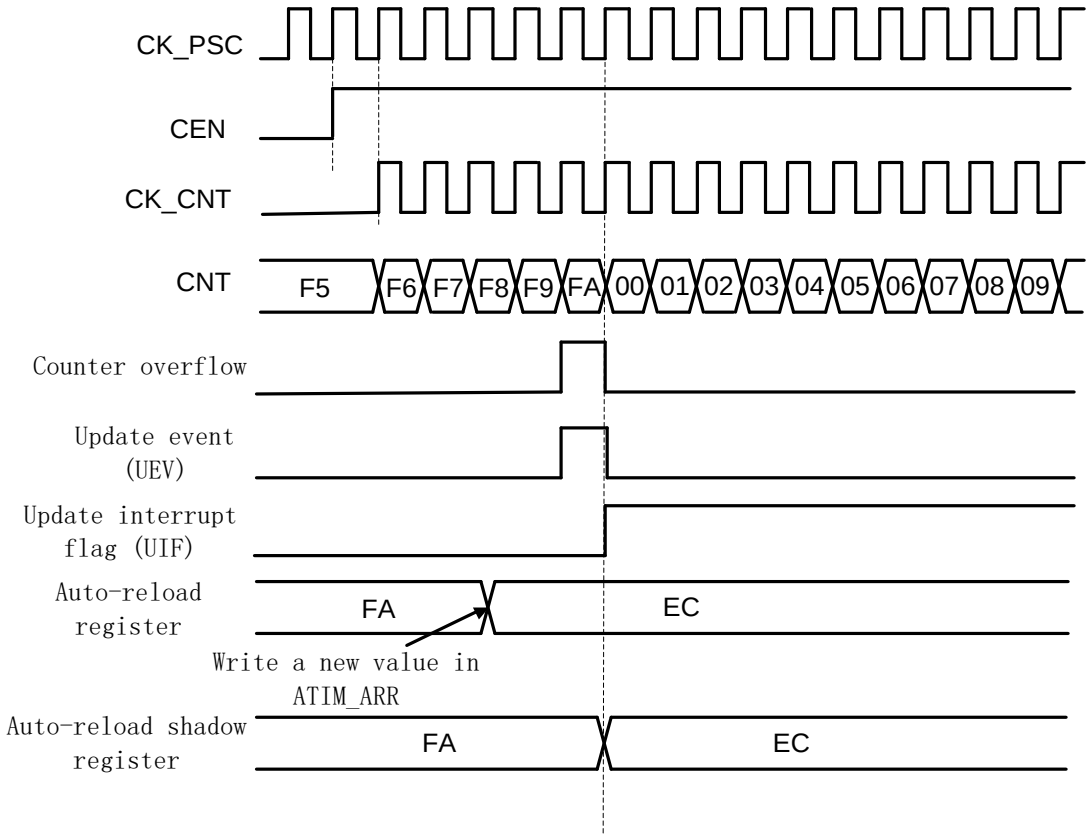


Figure 28-6 Update event when ARPE=1 (ARR preload)

Count down

In the down-counting mode, the counter starts to decrement from the ARR value, and when it reaches 0, an underflow event is generated, and the counter starts counting from ARR again.

If the repeat counting function is enabled, the counter repeats the above process several times (RCR+1) according to the definition of RCR before an overflow event will be generated.

The software can directly trigger the update event by setting the UG register, and the CNT and prescaler counter are automatically cleared at this time. Whether the setting of the UG register triggers the UIF (Update Interrupt Flag) interrupt flag setting is determined by the setting of the URS register.

The update event can be disabled by setting the UDIS register, which can avoid updating the value in the preload register to the working register.

When an update event occurs, the following registers are updated and UIF is set:

- The RCR shadow register is updated to the contents of ATIM_RCR
- The ARR shadow register is updated to ATIM_ARR content
- PSC shadow register is updated to ATIM_PSC content

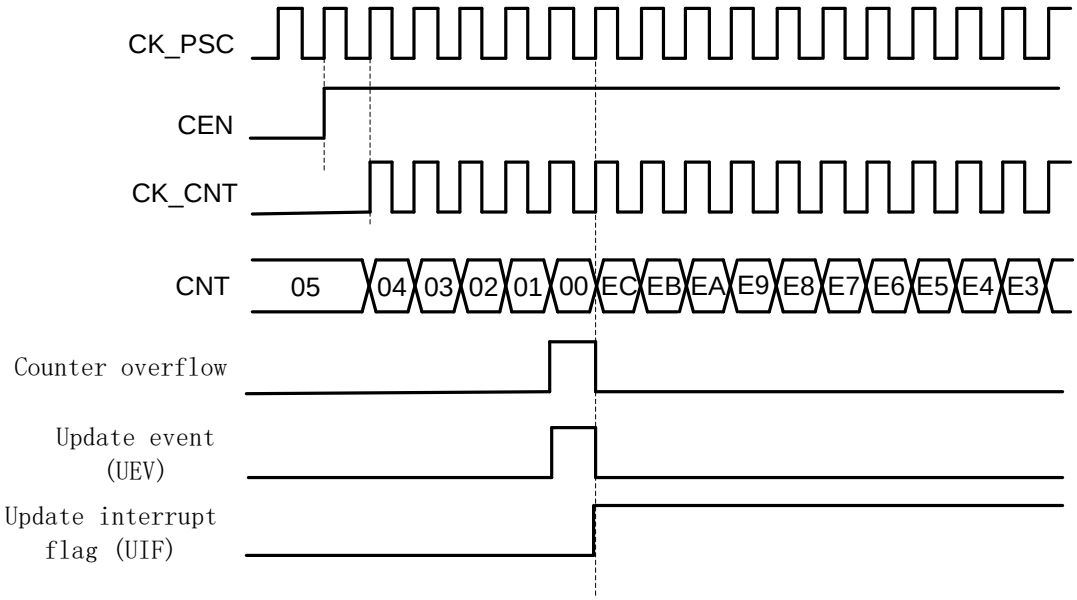


Figure 28-7 Downward counting waveform, internal clock not divided

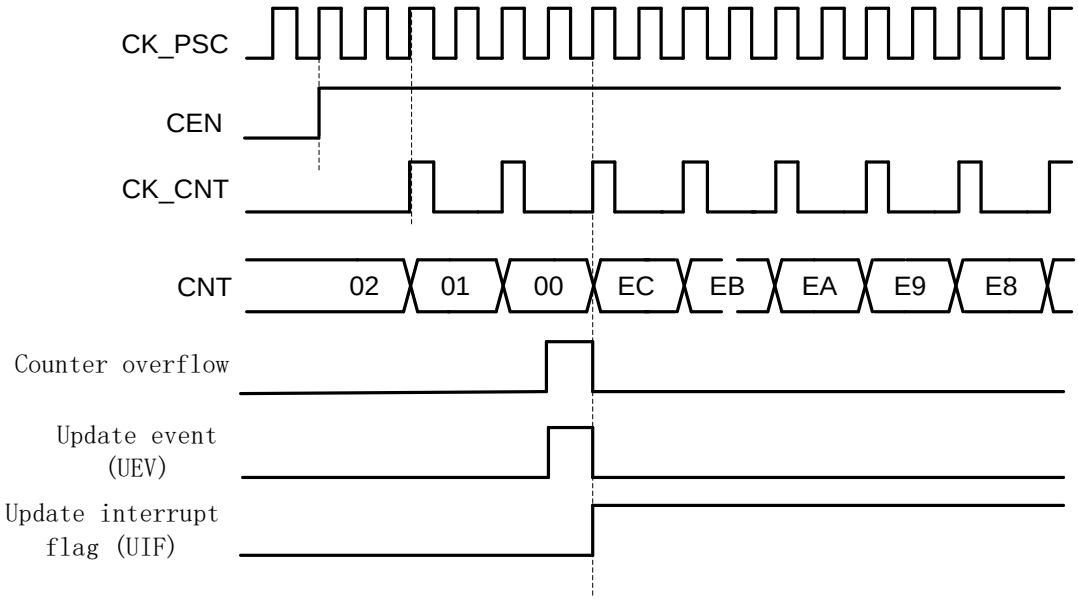


Figure 28-8 Downward counting waveform, internal clock divided-by-2

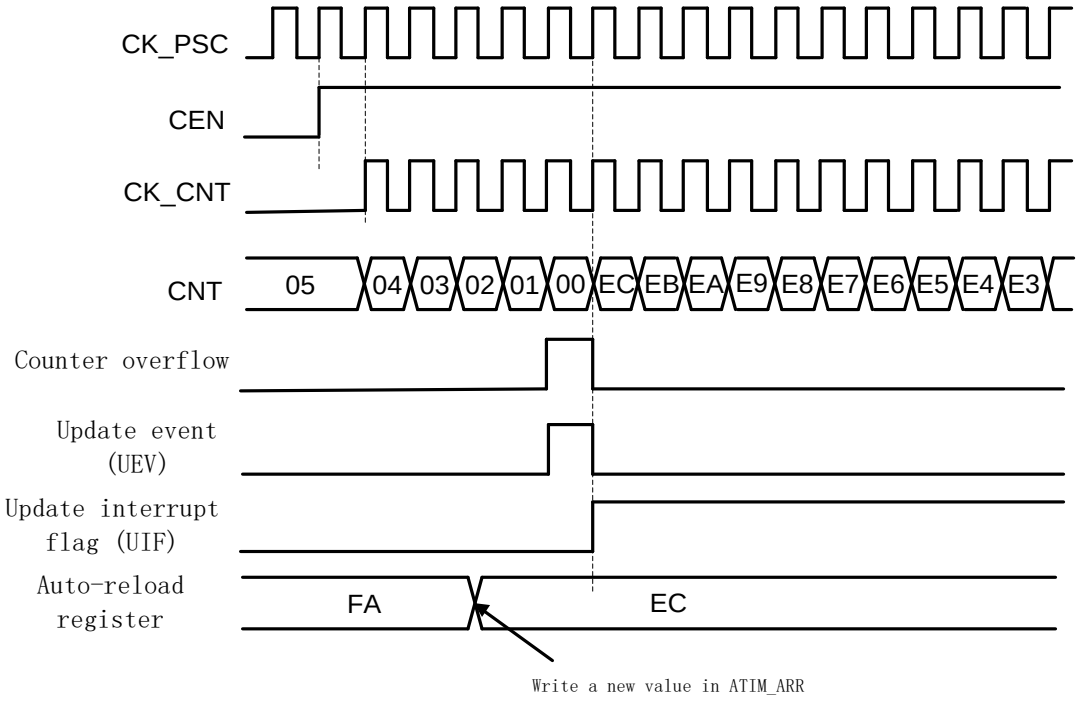


Figure 28-10 Downward counting waveform, internal clock divided-by-2

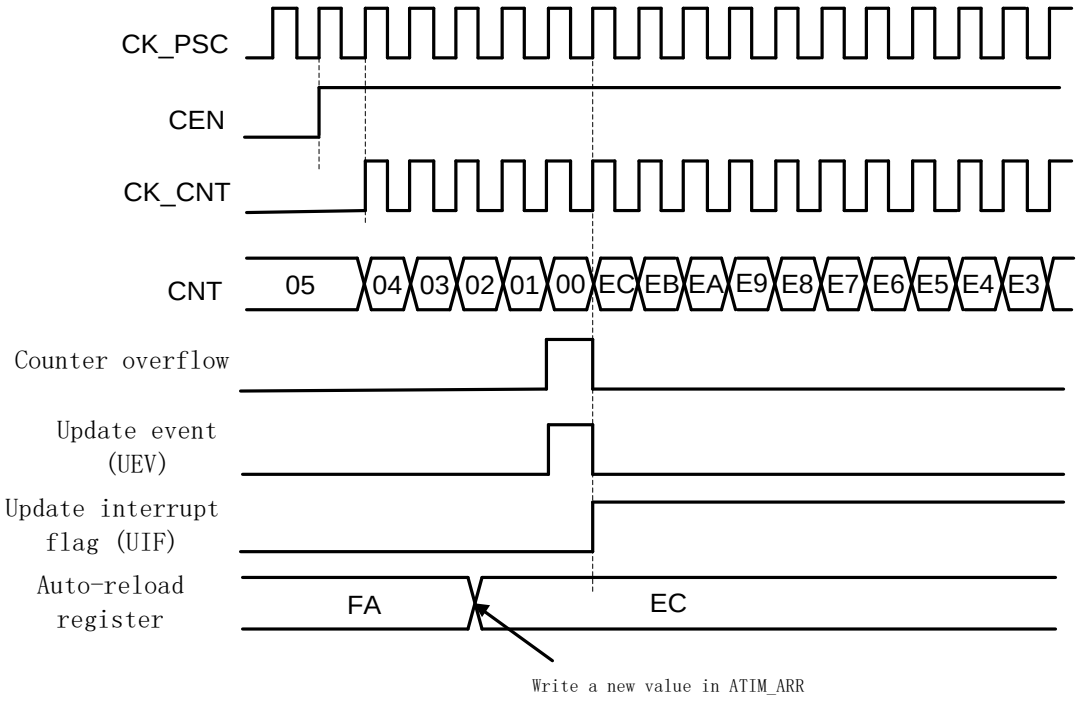


Figure 28-11 Downward counting，update event when repeat count is not used

Center alignment count

In the center-aligned mode, the counter starts counting up from 0 until ARR-1 generates an overflow event, and then counts down from ARR to 1, generating an underflow event, and then restarts counting up from 0.

The CMS[1:0] register is used to enable the center-aligned mode and select the output comparison mode in the center-aligned mode. When CMS!=00, it is center-aligned counting. When CMS=01, the output comparison function is only valid when counting down. When CMS=10, the output comparison function is only valid when counting up. When CMS=11, The output comparison function is valid when counting up and down.

In center-aligned mode, the DIR register cannot be rewritten by software, but is automatically updated by the hardware as the counting direction changes, indicating the current counting direction.

The counter will update the shadow registers of ARR, PSC and RCR on overflow and underflow events.

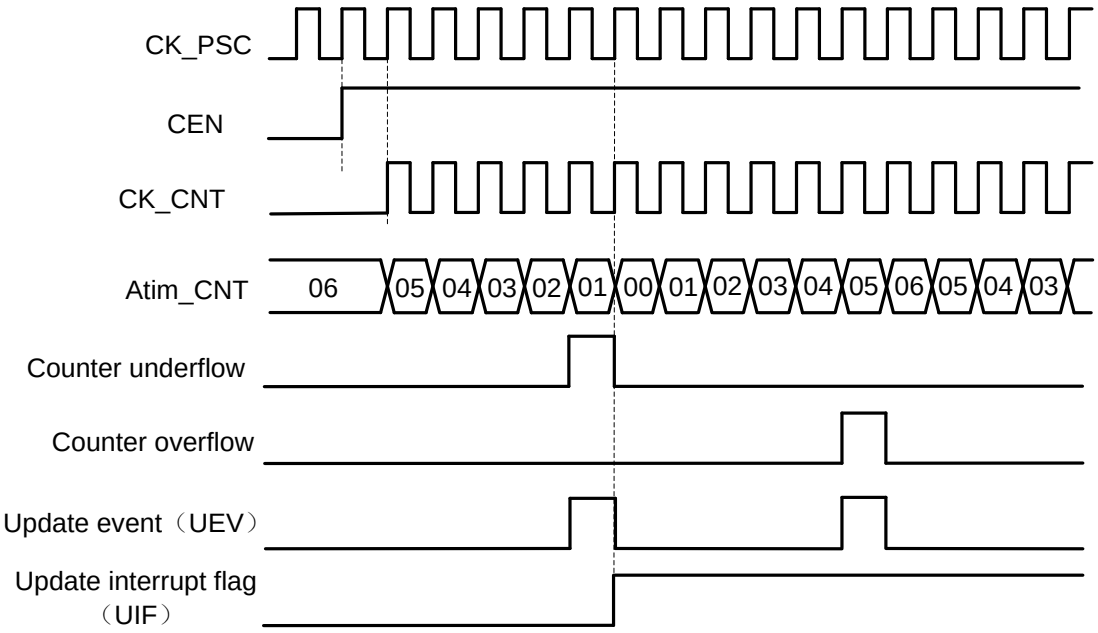


Figure 28-12 Timing diagram of center aligned counter, ATIM_PCS=0, ATIM_ARR=0x6

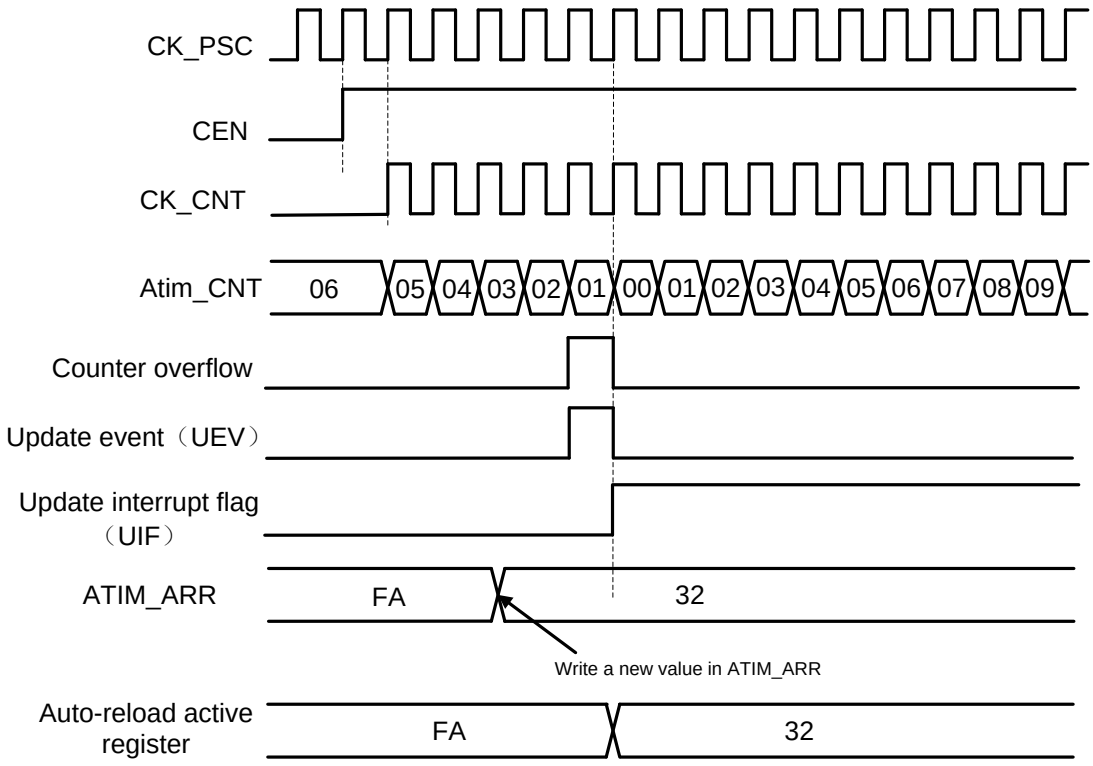


Figure 28-13 Counter timing diagram, update event when ARPE=1 (counter underflow)

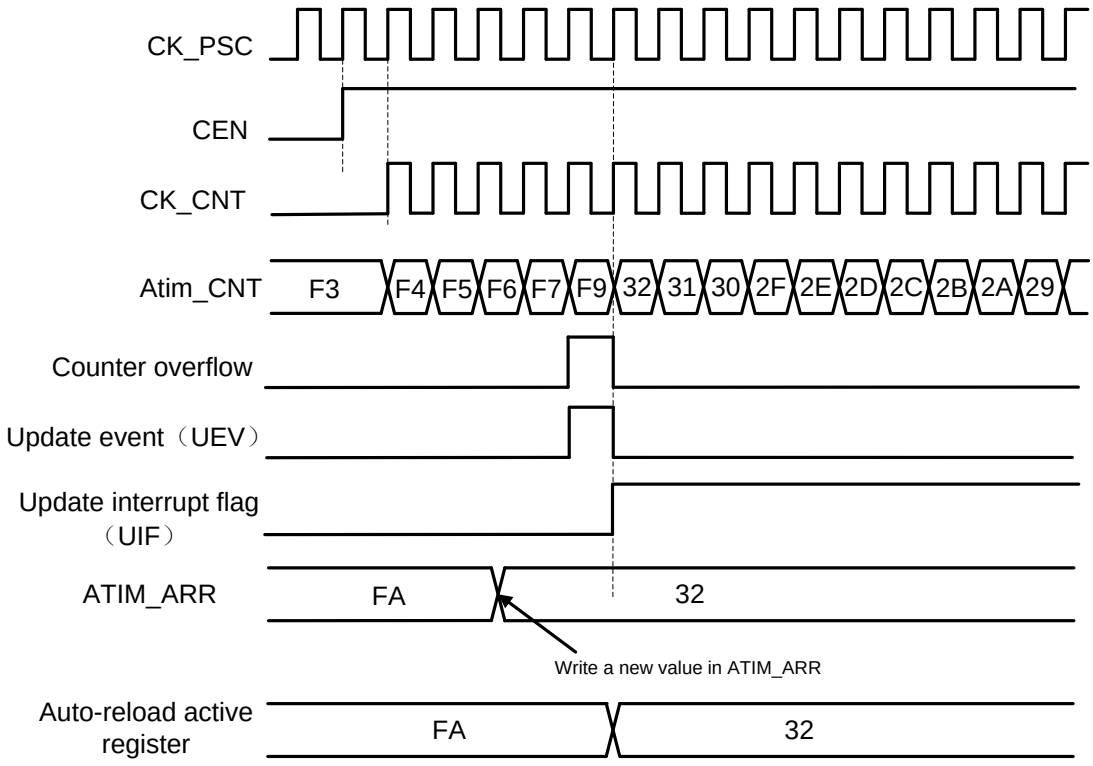


Figure 28-14 Counter timing diagram, update event when ARPE=1 (counter overflow)

28.4.3 Counter operating clock

The counter can use the following clock to work:

- APBCLK-internal clock mode
- External pin input clock (Tix)-external clock mode 1
- External pin trigger input (ETR)-external clock mode 2
- Internal trigger (ITRx)-use a timer's trigger output (TGO) as the counting clock

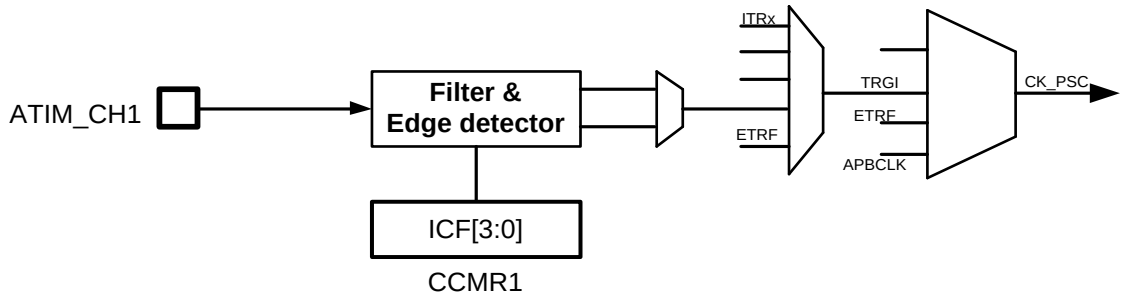


Figure 28-15 ATIM clock source block diagram

28.4.3.1 Internal clock mode

In internal clock mode, slave mode is prohibited (SMS=000), and register bits such as CEN, DIR, UG, etc. are all under software control

After the software operates the UG register, after the update signal is synchronized by CLK_PSC, the counter value will be reinitialized.

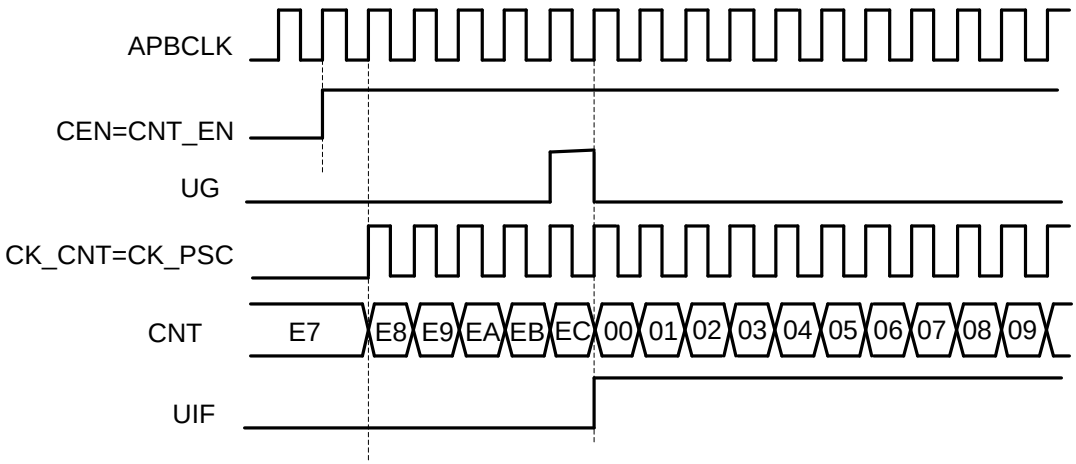


Figure 28-16 Internal clock source mode with clock division factor of 1

28.4.3.2 External clock mode1

In this mode, the external pin input signal is directly used as the counting clock, and SMS=111 is configured, and the counting edge can be configured as a rising or falling edge.

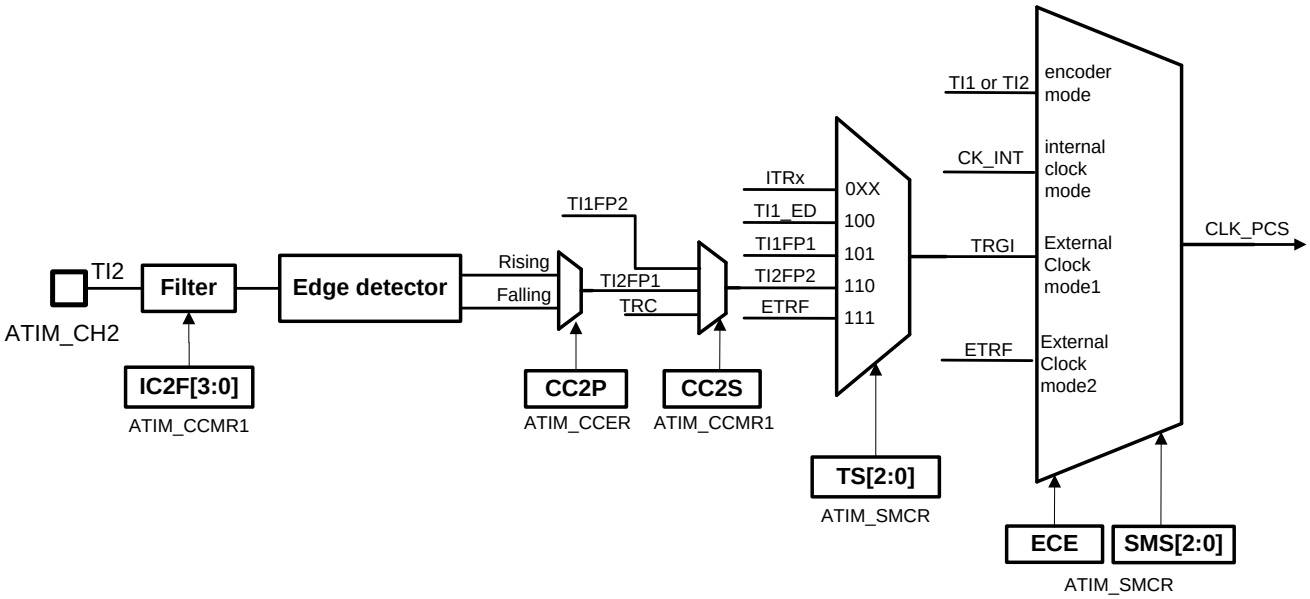


Figure 28-17 TI2 external clock connection example

The external input signal will go through the synchronization process of the internal clock before triggering the counter to count. At the same time, the valid edge of the input signal will trigger the TIF mark.

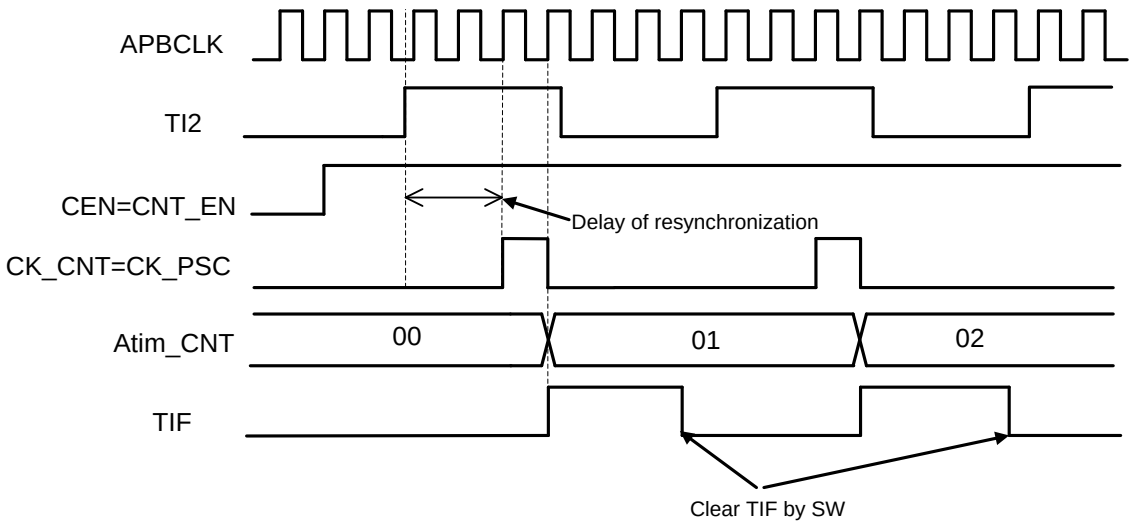


Figure 28-18 Timing in external clock mode 1

When using an external clock to count, still need to enable GPTIM's internal clock (APBCLK), because GPTIM uses APB_CLK to synchronize and filter the external input clock. In external clock mode 1, the external input clock is filtered and edge selected first to obtain a valid counting edge, which is input to the prescaler module as a valid working clock (CLK_PSC).

The external clock synchronization uses a simple 2-level flip-flop structure. Therefore, in order to avoid metastability, the external input clock width is required to be at least two APB_CLK cycles.

In this mode, only the inputs of channels 1 and 2 can be used as clock inputs, and the required configuration is as follows:

- In the GPIO module, configure the corresponding pin as GPTIM_CH2 function
- Turn off the channel enable and configure GPTIM_CCER.CC2E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure GPTIM_CCMR1.CC2S=01, IC2 is mapped to TI2
- Select the count valid edge, configure GPTIM_CCER.CC2P=0, select the up or down edge
- Configure the input filter time, configure GPTIM_CCMR1.IC2F[3:0] (IC2F=0000, no input filter)
- Enable external clock mode 1, configure GPTIM_SMCR.SMCR=111
- Select the trigger input source, configure GPTIM_SMCR.TS=110, select TI2 as the trigger input source
- Turn on the channel enable, configure GPTIM_CCER.CC2E=1
- Enable the counter, configure GPTIM_CR1.CEN=1

The following figure is an example of a typical external clock counting mode 1:

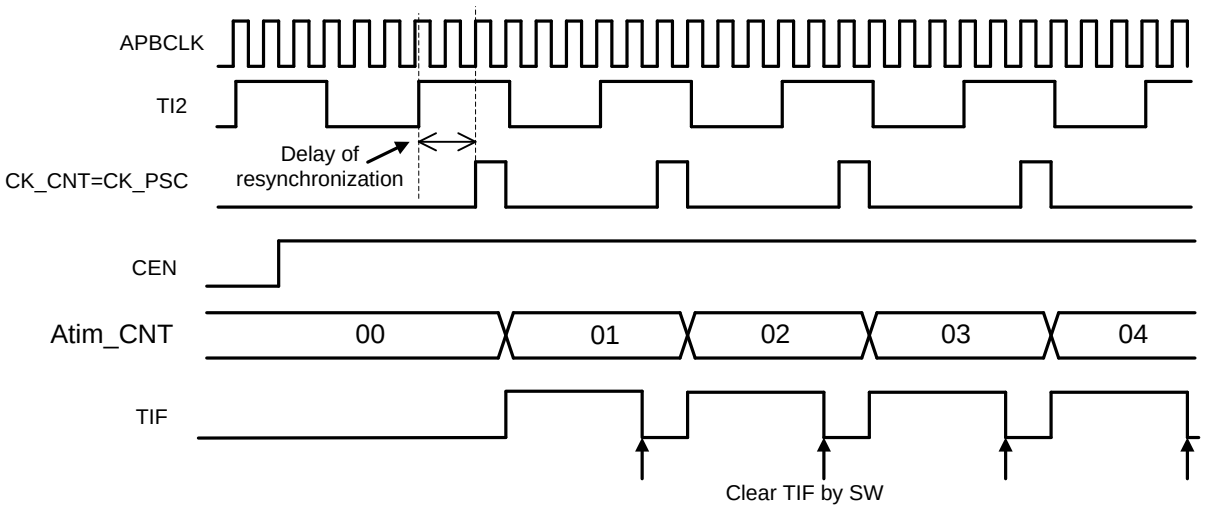


Figure 28-19 Timing in external clock mode 1

28.4.3.3 External clock mode 2

In this mode, the rising or falling edge of the input signal at the GPTIM_ETR pin is used for counting (double edges are not supported).

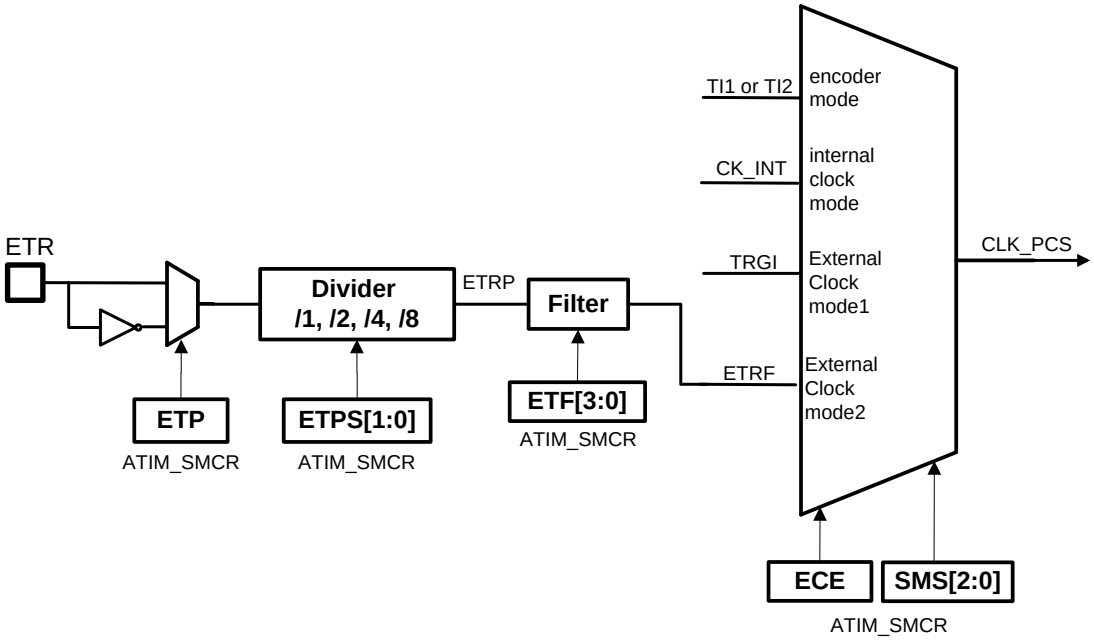


Figure 28-20 External trigger input block diagram

The figure below uses the rising edge of the ETR divided by two to count. The actual counting time is delayed from the rising edge of the ETR input due to the synchronization process of the internal clock.

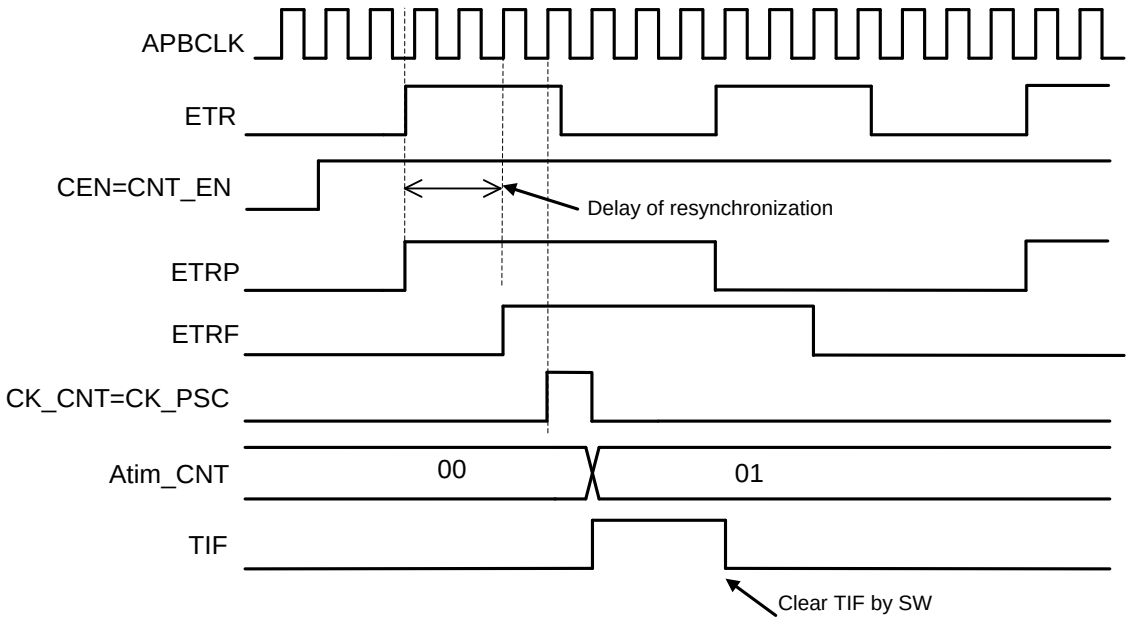


Figure 28-21 1 timing under external clock mode

The main difference with external clock mode 1 is that the ETR input is directly divided by frequency and then filtered to generate the CK_PSC clock, which means that it can support the application scenarios where the ETR input frequency is higher than APB_CLK. In this case, you need to input the ETR first Carry out prescaler, and then use it to drive the counter.

The configuration required for this mode is as follows:

- In the GPIO module, configure the corresponding pin for the GPTIM_ETR function
- Set ETP for edge selection, GPTIM_SMCR.ETP=0
- Set the ETR division ratio, configure GPTIM_SMCR.ETPS[1:0]=01
- Configure the input filter time, GPTIM_SMCR.ETF[3:0]=0000
- Set the ECE register, enable external clock mode 2, GPTIM_SMCR.ECE=1, GPTIM_SMCR.SMS=000
- Enable the counter, configure GPTIM_CR1.CEN=1

The following figure is an example of a typical external clock mode 2:

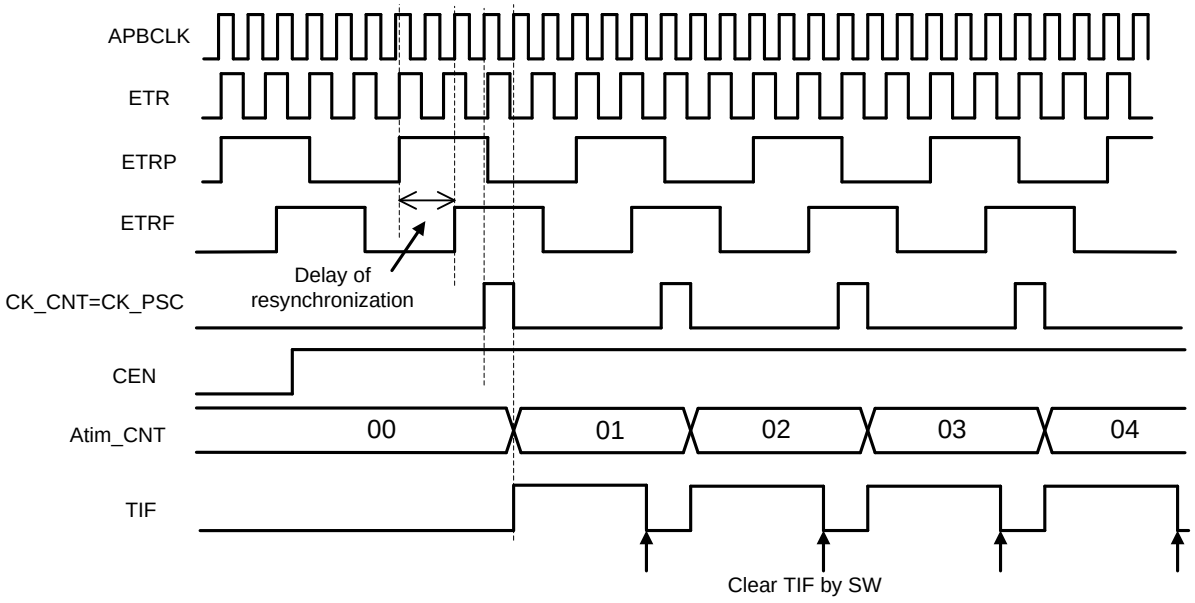


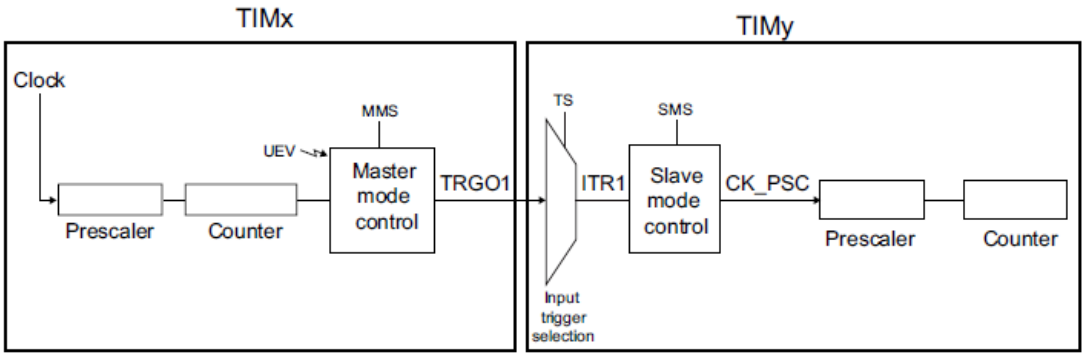
Figure 28-22 Timing 2 under external clock mode

When using external clock mode 2, GPTIM can still be configured as slave mode: for example, use the ETR input to count, and at the same time use the TRGO of another Timer as the trigger signal, when the trigger event arrives, the reset counter restarts counting.

28.4.3.4 Internal trigger mode

Each GPTIM supports 4 ITR inputs, which can be used for counting trigger or internal signal capture. When ITR is selected as the count trigger signal, the GPTIM counter will count during the high level period of each ITR signal, or be triggered by the rising edge of the ITR signal. Timer cascade can be realized through internal trigger mode.

The following figure is an example:



Configure TIMx as master mode and periodically output TRGO pulse signals. Configure TIMy as Slave mode and set the TRGO of TIMx to ITR; when the TIMx.TRGO pulse arrives, TIMy counts once.

The timer cascade based on the internal trigger mode has the following requirements:

- TRGO signal is designed as APBCLK single cycle pulse
- Both TIMx and TIMy work in the APBCLK clock domain
- TRGO is a synchronization pulse for the receiver
- Both Master and Slave working clocks must be enabled

In addition to other timer outputs, the trigger signal that can be used in the internal trigger mode can also be ADC_EOC or comparator mode output. To satisfy the above requirements, it is necessary to process the trigger signals output by ADC and COMP into APBCLK synchronous pulses during the design.

28.4.4 Capture of internal trigger signal (ITRx)

GPTIM supports 4 ITR inputs, which can be used for counting trigger or internal signal capture. When used for internal signal capture, TS needs to be configured as 000~011 to select ITR0~ITR3, and CCxS is configured as 11, that is, TRC is selected as the capture signal.

Each ITR input supports 4 internal signal extensions, which are configured by the ITRxSEL register. Refer to the following table for input signal source:

GPTIM1			Function
ITR0SEL	00	ATIM_TRGO	Count trigger
	01	UART0_RX	Width snap
	10	UART1_RX	Width snap
	11	UART4_RX	Width snap
ITR1SEL	00	GPTIM1_TRGO	Count trigger
	01	XTHF	Cycle capture
	10	RCHF	Cycle capture
	11	LPUART0_RX	Cycle capture
ITR2SEL	00	BSTIM_TRGO	Count trigger
	01	LPUART1_RX	Width snap
	10	LPOSC	Cycle capture
	11	XTLF	Cycle capture
ITR3SEL	00	COMP1_TRGO	Count trigger
	01	RCMF	Cycle capture
	10	COMP2_TRGO	Count trigger
	11	LPT_TRGO	Count trigger
GPTIM2			**Function**
ITR0SEL	00	ATIM_TRGO	Count trigger
	01	UART0_RX	Width snap
	10	UART1_RX	Width snap
	11	UART4_RX	Width snap
ITR1SEL	00	GPTIM0_TRGO	Count trigger
	01	XTHF	Cycle capture
	10	RCHF	Cycle capture
	11	ADC_EOC_TRGO	Count trigger
ITR2SEL	00	BSTIM_TRGO	Count trigger
	01	LSCLK	Cycle capture
	10	LPOSC	Cycle capture
	11	XTLF	Cycle capture
ITR3SEL	00	COMP1_TRGO	Count trigger
	01	RCMF	Cycle capture
	10	COMP2_TRGO	Count trigger

	11	LPT_TRGO	Count trigger

The TRGO signal, which is a count trigger, is processed as an APBCLK width synchronization enable signal before it reaches the ITRx of GPTIM. Signals used for period or width capture are captured by GPTIM without processing.

The red part of the table above represents the signal that needs to be processed by the master into an APBCLK synchronization pulse.

The software should ensure that the correct signal is selected for the correct function, and the wrong configuration will lead to completely wrong results. For example, if ATIM_TRGO is used for width capture, the result is meaningless.

28.4.5 Capture/Compare channel

GPTIM contains 4 capture/compare channels, and each channel consists of a capture compare register (CCR) (including shadow registers), a capture input stage, and a compare output stage.

The input stage circuit will sample the Tix input and generate a filtered signal TixF, and then edge detection and polarity selection will generate the corresponding TixFPx signal. This signal can be used as a counting trigger or a signal to be captured, and is prescaled before being captured.

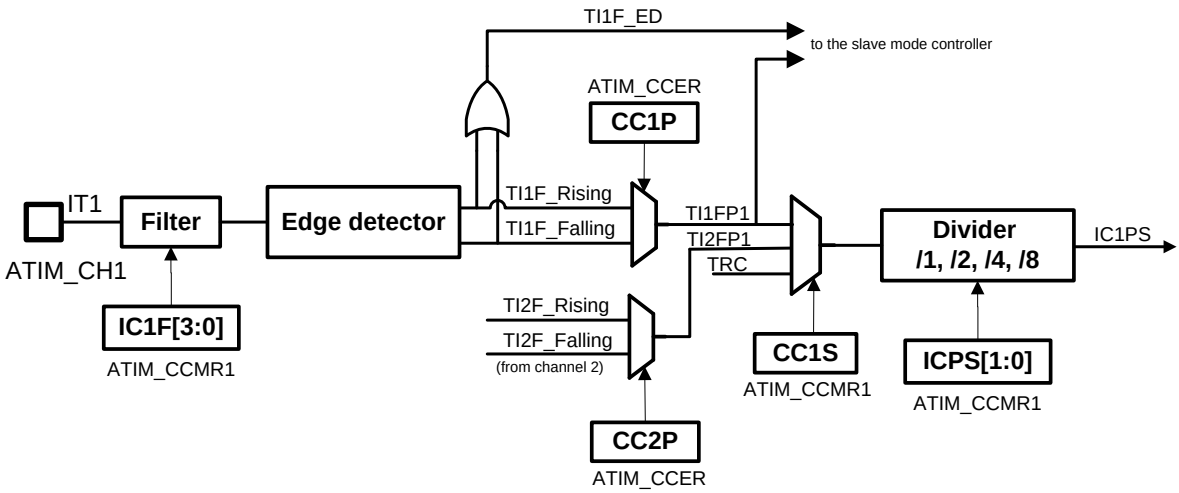


Figure 28-23 Capture/compare channel (channel 1 input part)

The output stage circuit will generate an output reference signal OCxREF, which is fixed to high level and effective as the reference input of the final output circuit. The GPTIM output channel does not support complementary output.

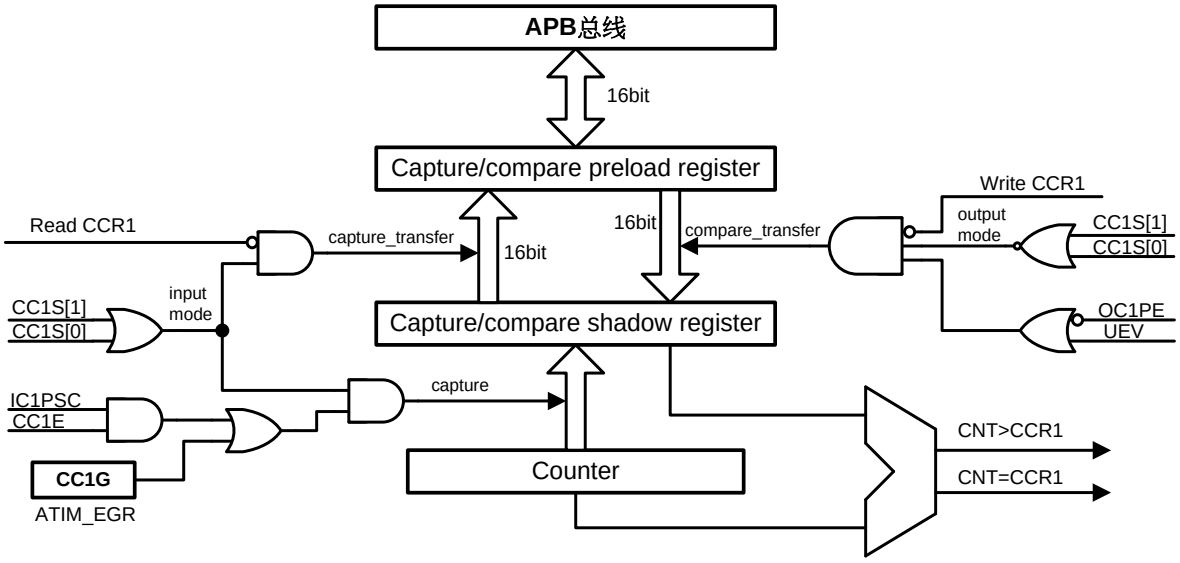


Figure 28-24 Main circuit of capture/compare channel 1

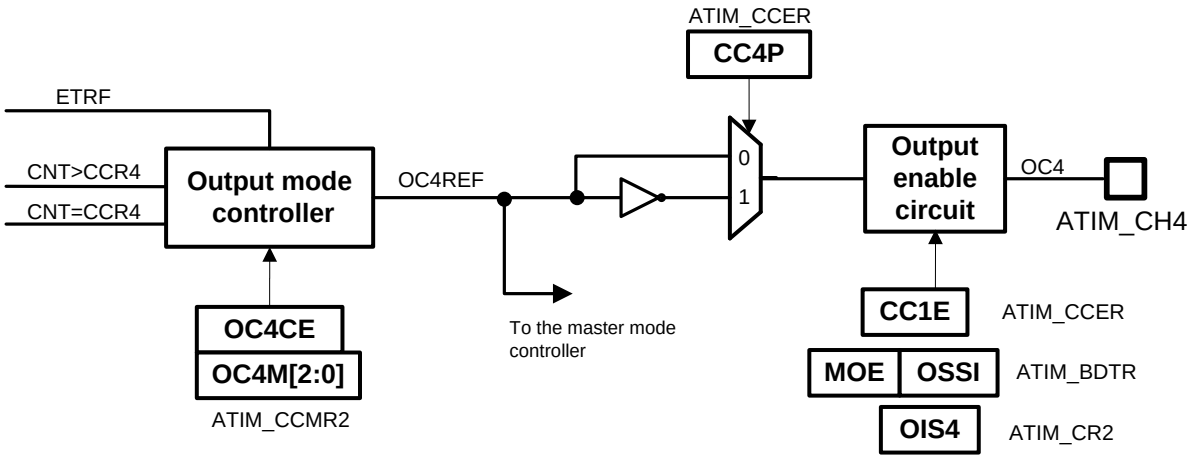


Figure 28-25 Output section of capture/compare channel

The capture/compare register (CCR) contains the preload register and the shadow register, and software reads and writes always access the preload register. In the capture mode, the captured value is saved in the shadow register and copied to the preload register. In the compare mode, the value of the preload register is copied to the shadow register for comparison with the counter.

28.4.6 Input capture mode

When the expected level change occurs on the ICx signal, a capture will be triggered, and the current counter value will be latched into the CCR. At the same time, the CCxIF interrupt flag is set, and the corresponding interrupt or DMA request can be triggered. If a capture event occurs when CCxIF is high, the capture data conflict flag (CCxOF, Over-Capture) is set to bit (the last captured value in CCR is overwritten). CCxIF can be cleared by software or automatically cleared by reading the CCR register. The CCxOF flag is cleared by writing 1 in software.

Through the cooperation of two or more channels, the input capture of the PWM signal can be realized. For example, if you want to calculate the period and duty ratio of an input signal, you can input this signal from the TI1 pin, and the chip will take the rising edge of the filtered signal to get TI1FP1, take the falling edge of the filtered signal to get TI1FP2, and input TI1FP1 To capture channel 1, input TI1FP2 to capture channel 2, then channel 1 captures the rising edge of the input signal, while channel 2 captures the falling edge of the input signal; after the capture interrupt occurs periodically, the software passes the value of the CCR1 and CCR2 registers, The period and duty cycle of the input signal can be calculated.

To achieve the capture of the counter value to the GPTIM_CCR1 register at the rising edge of TI1 input, the configuration steps are as follows:

- In the GPIO module, configure the corresponding pin for the GPTIM_CH1 function
- Turn off the channel enable and configure GPTIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure GPTIM_CCMR1.CC1S=01, IC1 is mapped to TI1
- Select the count valid edge, configure GPTIM_CCER.CC1P, select the up or down edge
- Configure the input filter time, configure GPTIM_CCMR1.IC1F[3:0]
- Configure the input prescaler, configure GPTIM_CCMR1.IC1PS[1:0]
- Turn on the channel enable, configure GPTIM_CCER.CC1E=1

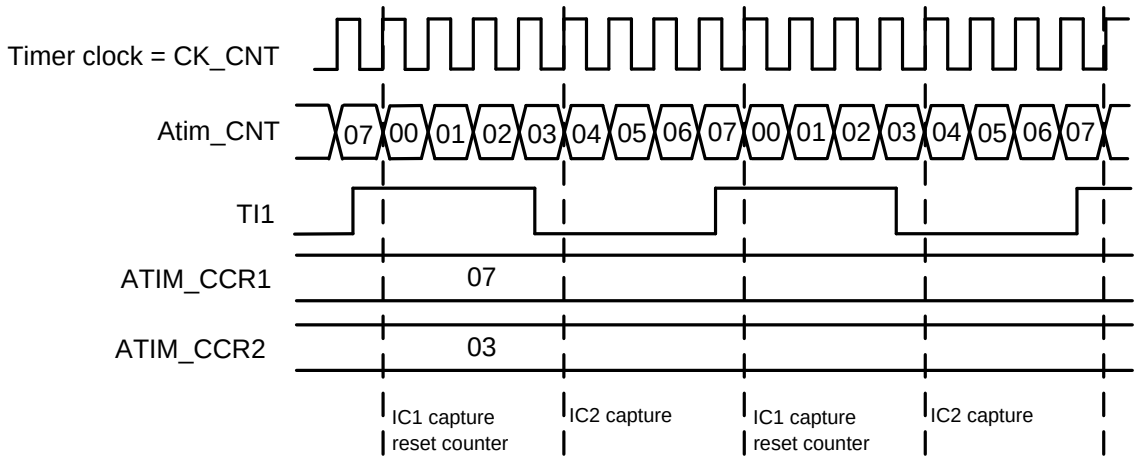


Figure 28-26 PWM input capture mode timing

If you want to realize the PWM input capture function, you need to make the following settings:

- In the GPIO module, configure the corresponding pin for the GPTIM_CH1 function
- Turn off the channel enable, configure GPTIM_CCER.CC1E=0, GPTIM_CCER.CC2E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, the two channels IC1, IC2 are mapped to the same TI1 input port, configure GPTIM_CCMR1.CC1S=01, GPTIM_CCMR1.CC2S=10
- Select the counting valid edge, the two channels IC1, IC2 valid edge polarity is opposite, configure GPTIM_CCER.CC1P=0, GPTIM_CCER.CC2P=1
- Configure the input filter time, configure GPTIM_CCMR1.IC1F[3:0], GPTIM_CCMR1.IC2F[3:0]
- Configure the input prescaler, configure GPTIM_CCMR1.IC1PS [1:0], GPTIM_CCMR1.IC2PS [1:0]
- Select the trigger input signal, configure GPTIM_SMCR.TS[2:0] =101
- Set the slave mode controller to multiple bit mode, configure GPTIM_SMCR.SMS[2:0]=100
- Turn on the channel enable, configure GPTIM_CCER.CC1E=1, GPTIM_CCER.CC2E=1

28.4.7 Software Force output

In the comparison output mode, the software can directly set the OCxREF force to a specific level, independent of the comparison result of the CCR and the counter.

The software can directly force OCxREF to be valid by writing OCxM=101 register (OCxREF is

fixed to high and effective), and by writing OCxM=100, OCxREF can be directly forced to be invalid (low level). However, the software force operation will not cancel the comparison process, and the comparison between CCR and counter will continue.

28.4.8 Output compare mode

In the output comparison mode, when CCR is equal to the counter value, OCxREF can be set to valid, invalid, or level inverted. At the same time, the interrupt flag is also set, and DMA requests can be sent.

Output comparison can also be used to output a pulse signal of a specific width (single output).

Steps for usage:

1. Select the counting clock (internal, external, prescaler, etc.)
2. Write the desired data to the ARR and CCR registers
3. Set interrupt enable and DMA enable as needed
4. Select the output mode
5. Enable the counter

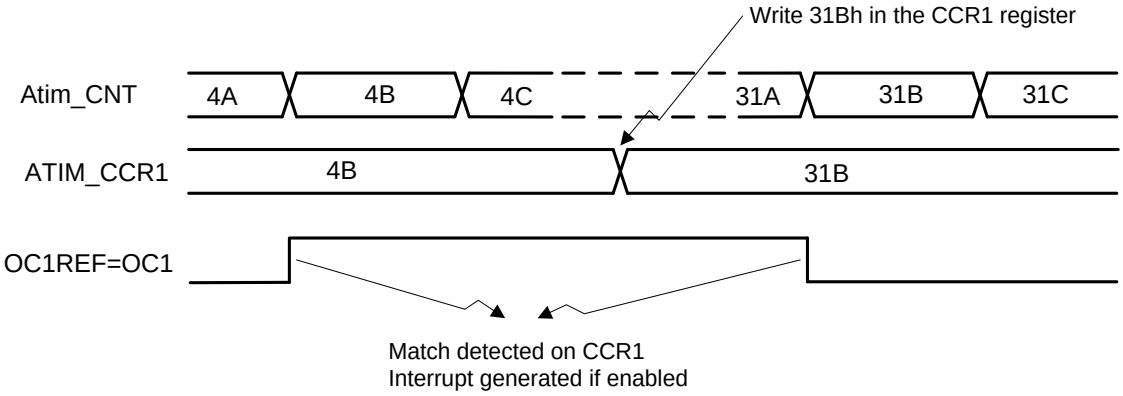


Figure 28-27 Output compare mode, flip OC1

Without enabling preload, the software can rewrite the CCR register at any time to achieve real-time control of the output waveform. If preload is enabled, the CCR shadow register is only updated with the contents of the preload register when the next update event occurs.

28.4.9 PWM input mode

PWM mode can output pulse width modulation signal, its period is determined by the ARR register, and the duty cycle is determined by the CCR register.

The polarity of the output signal can be configured by the CCxP register. In PWM mode, CNT and CCR are compared in real time. Since the counter supports edge-aligned and center-aligned counting modes, the PWM output also supports edge-aligned and center-aligned modes.

PWM edge alignment mode

In the case of counting up, when configured as PWM mode 1, the OCxREF signal is high when CNT<CCR, otherwise it is low. If the CCR value is greater than the ARR value, OCxREF is fixed to 1; if CCR is 0, OCxREF is fixed to 0.

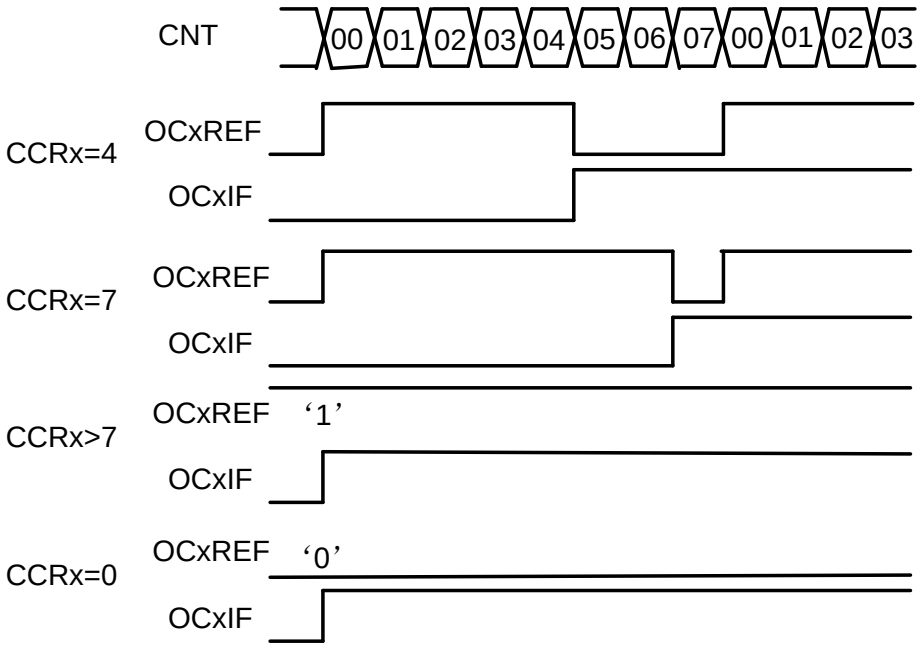


Figure 28-28 Edge-aligned PWM waveform (ARR=7)

PWM center alignment mode

The OCxREF level definition is the same as the edge alignment mode. The following figure is an example:

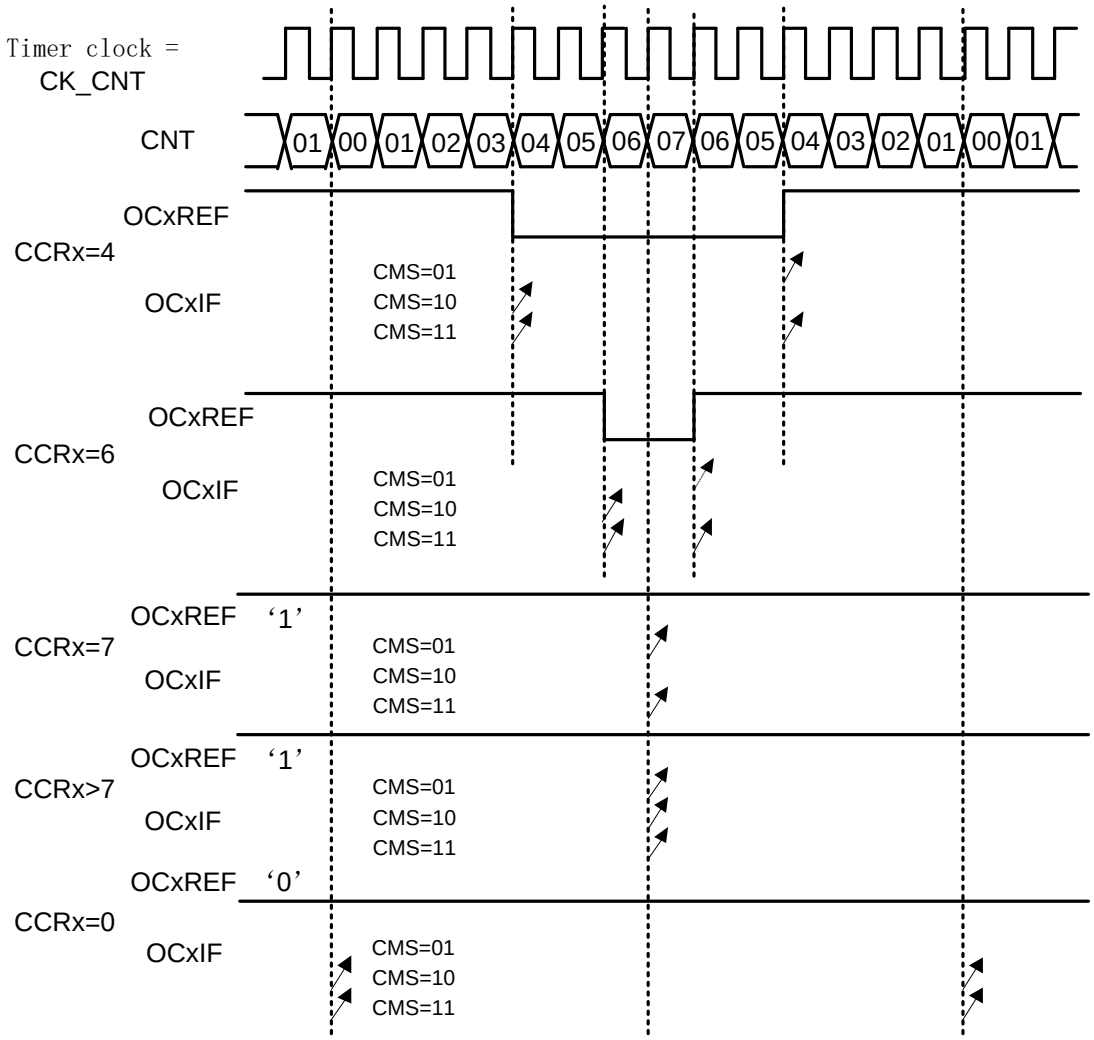


Figure 28-29 Center-aligned PWM waveform (APR=7)

When starting the center-aligned counting, the initial counting direction is determined by the DIR register; then during the counting process, the state of the DIR register is directly controlled by the hardware. For safety, it is recommended that the user program do an update through the UG register before starting the counter, and do not rewrite the counter during the counting process.

28.4.10 Single pulse output

Single pulse output is a special case of the comparison output mode, which allows the user to output a pulse signal with a programmable width after a certain event occurs, after a programmable delay.

Unlike other output modes, the counter will automatically stop when the next update event arrives.

Only when the initial value of CCR and the counter are different, the pulse can be output correctly. When counting up, CNT<CCR<=ARR is required, when counting down, CNT>CCR is required

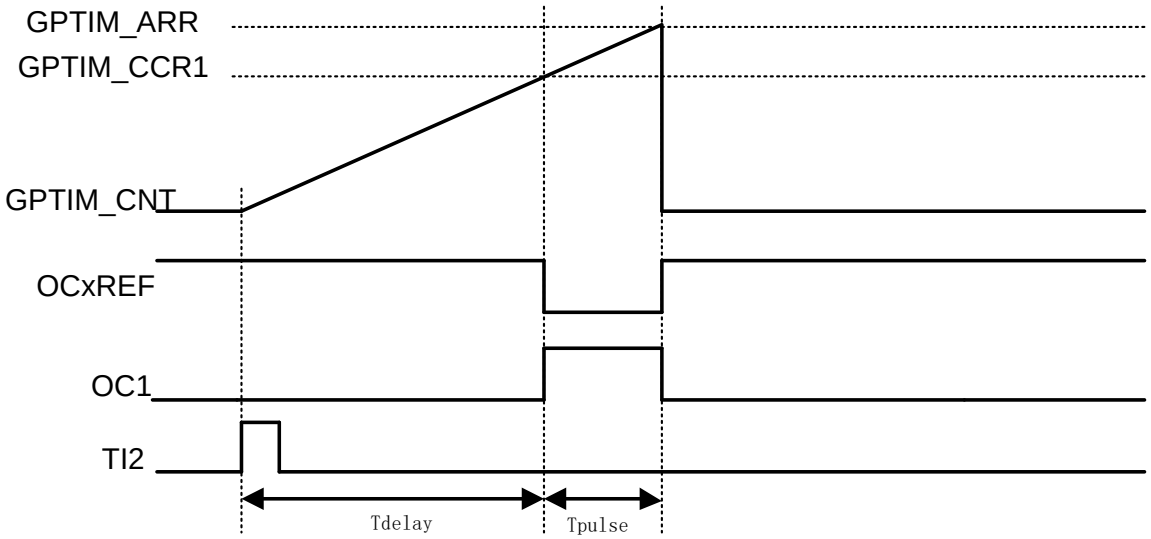


Figure 28-30 Example of single pulse mode

The above figure uses TI2 input as the counter trigger signal. After the count value is equal to CCR, OCxREF outputs low level. After counting to ARR, OCxREF returns to high level, and the counter rolls back to 0 and stops counting.

The configuration that realizes the above function TI2 as an input trigger is as follows:

- In the GPIO module, configure the corresponding pin for the GPTIM_CH2 function
- Turn off the channel enable and configure GPTIM_CCER.CC2E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure GPTIM_CCMR1.CC2S=01
- Select the effective edge of counting, configure GPTIM_CCER.CC2P=0
- Select the trigger input signal, configure GPTIM_SMCR.TS[2:0]=110, TI2FP2 as TRGI
- Set the slave mode controller to trigger mode, configure GPTIM_SMCR.SMS[2:0]=110, TI2FP2 is used to start the counter
- Turn on the channel enable, configure GPTIM_CCER.CC2E=1

The configuration that realizes the above function OC1 as an output is as follows:

- In the GPIO module, configure the corresponding pin for the GPTIM_CH1 function
- Turn off the channel enable and configure GPTIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards
- Output channel, configure GPTIM_CCMR1.CC1S=00
- Select the effective edge of counting, configure GPTIM_CCMR1.OC1M=111, PWM mode 2
- Turn on the channel enable, configure GPTIM_CCER.CC1E=1

Special settings of OPM waveform generation time base:

- The value of GPTIM_CCR1 determines Tdelay
- The difference between GPTIM_ARR and GPTIM_CCR1 determines the Tpulse (GPTIM_ARR-GPTIM_CCR1)
- Set to single pulse mode, configure GPTIM_CR1.OPM=1

28.4.11 External event clear OCxREF

The effective state of OCxREF is not high. By applying a high level to the external ETR pin, OCxREF can be directly pulled down until the next update event. This function is only valid in output comparison and PWM mode, and cannot be used in software force mode. To enable this function, you need to set OCxCE to 1.

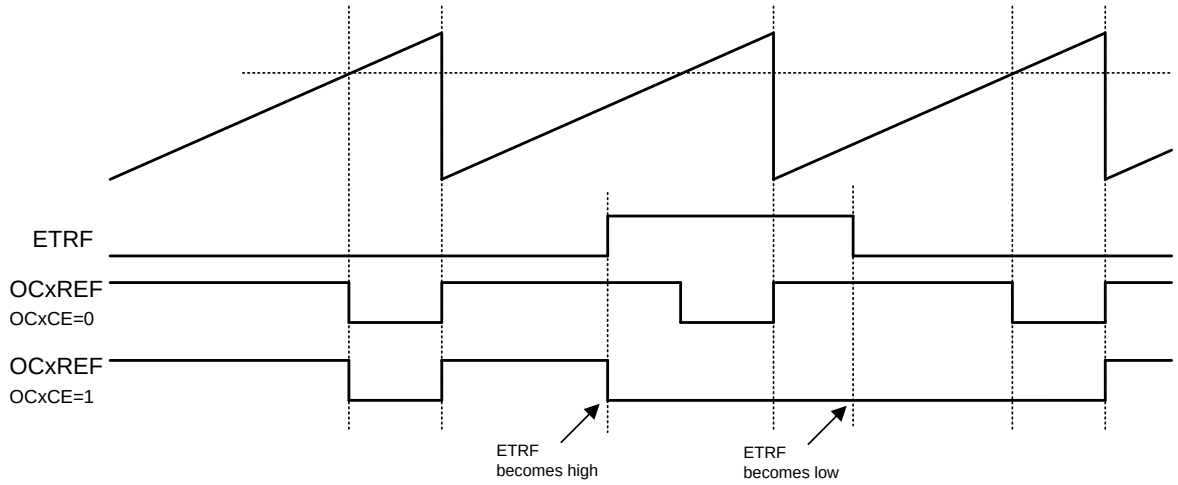


Figure 28-31 ETR signal clears OCxREF of GPTIM

28.4.12 Encoder interface mode

The encoder interface mode involves two external input signals, and GPTIM determines whether to increment or decrement the count value according to the edge of one signal relative to the level of the other signal. The following table shows the relationship between the counting method and the two input signals:

Valid edge	Corresponding signal level (TI1 corresponds to TI2, TI2 corresponds to TI1)	TI1signal		TI2signal	
		Up	Down	Up	Down
Only count at TI1	High	Decrease	Increase	Not counted	Not counted
	Down	Increase	Decrease	Not counted	Not counted
Only count at TI2	High	Not counted	Not counted	Increase	Decrease
	Down	Not counted	Not counted	Decrease	Increase
Count both at TI1 and TI2	High	Decrease	Increase	Increase	Decrease
	Down	Increase	Decrease	Decrease	Increase

Table 28-1 Encoder interface counting method

For example, when the counter counts with the TI1 signal as a clock, if the rising edge of TI1 is sampled until TI2 is high, the counter is decremented; if the falling edge of TI1 is sampled until TI2 is high, the counter is incremented.

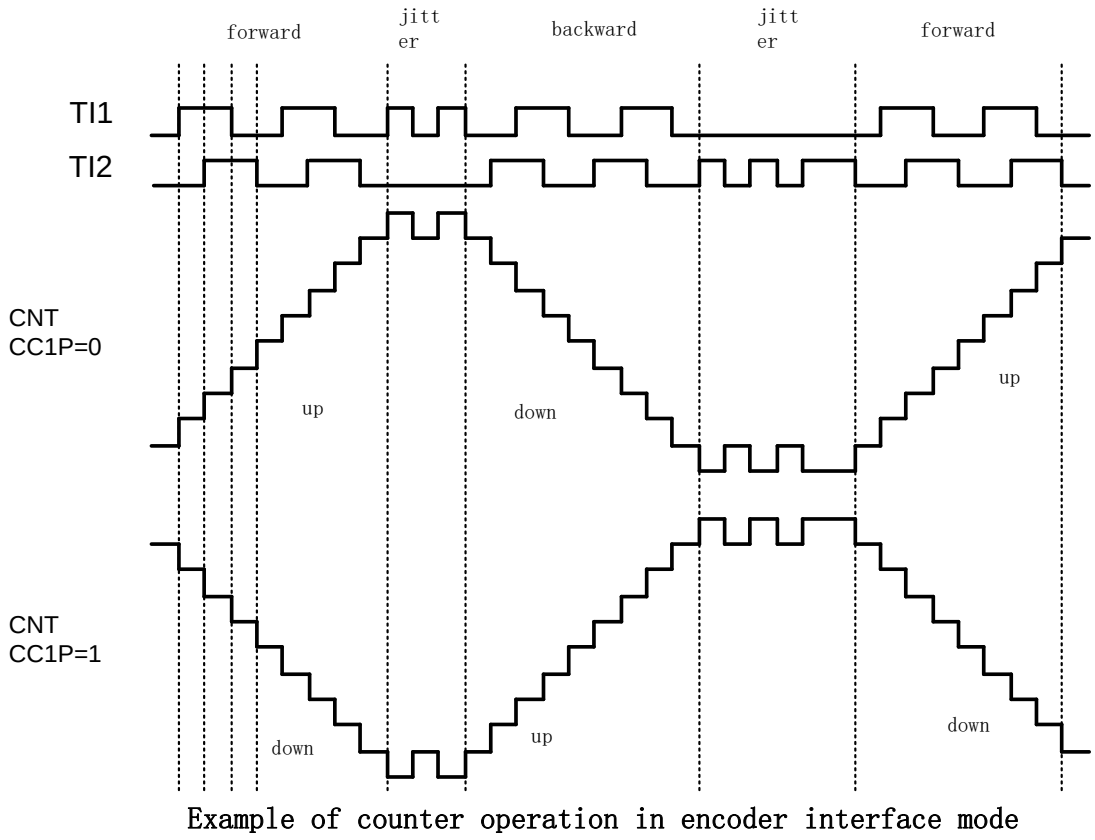


Figure 28-32 Example of counter operation in encoder mode

The encoding mode input channel needs to be set as follows:

- In the GPIO module, configure the corresponding pins as GPTIM_CH1, GPTIM_CH2 functions
- Turn off the channel enable, configure GPTIM_CCER.CC1E=0, GPTIM_CCER.CC2E=0, to ensure that the channel configuration is successful afterwards
- Select the input channel, configure GPTIM_CCMR1.CC1S=01, GPTIM_CCMR1.CC2S=01
- Select the count valid edge, configure GPTIM_CCER.CC1P=0, GPTIM_CCER.CC2P=0
- Set the slave mode controller to encoding mode 3, configure GPTIM_SMCR.SMS[2:0]=011
- Turn on the channel enable, configure GPTIM_CCER.CC1E=1, GPTIM_CCER.CC2E=1

28.4.13 GPTIM slave mode

When GPTIM is used as a slave (triggered by an external event), it can be configured into three working modes: multi-bit mode, gating mode, and trigger mode.

Complex bit mode

When GPTIM is used as a slave (triggered by an external event), it can be configured into three working modes: multi-bit mode, gating mode, and trigger mode.

Complex bit mode

In this mode, an external input event will cause all preload registers in the TIM to reinitialize, and CNT will return to 0 to start counting. Take the following figure as an example, the counter counts normally, and when the external TI1 input rising edge, the counter is cleared and restarted to count.

- The configuration in the following figure is as follows:
- In the GPIO module, configure the corresponding pin for the GPTIM_CH1 function
- Turn off the channel enable, configure GPTIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure GPTIM_CCMR1.CC1S=01
- Select the valid edge of counting, configure GPTIM_CCER.CC1P=0
- Select the trigger input signal, configure GPTIM_SMCR.TS[2:0]=101, and TI1FP1 as TRGI
- Set the slave mode controller to multiple bit mode, configure GPTIM_SMCR.SMS[2:0]=100
- Turn on the channel enable, configure GPTIM_CCER.CC1E=1
- Enable the counter, configure GPTIM_CR1.CEN=1

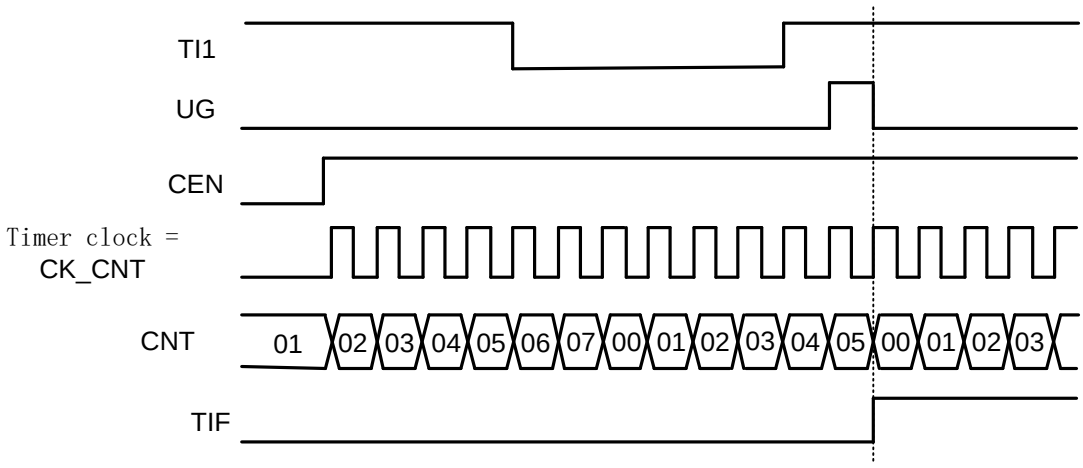


Figure 28-33 Timing in bit mode

Gating mode

In this mode, the counter only works when the input signal is at a specific level. When the level change causes the counter to start or stop counting, the interrupt flag is triggered.

The configuration in the following figure is as follows:

- In the GPIO module, configure the corresponding pin for the GPTIM_CH1 function
- Turn off the channel enable, configure GPTIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure GPTIM_CCMR1.CC1S=01
- Select the valid edge of counting, configure GPTIM_CCER.CC1P=0
- Select the trigger input signal, configure GPTIM_SMCR.TS[2:0]=101, and TI1FP1 as TRGI
- Set the slave mode controller to gated mode, configure GPTIM_SMCR.SMS[2:0]=101
- Turn on the channel enable, configure GPTIM_CCER.CC1E=1
- Enable the counter, configure GPTIM_CR1.CEN=1

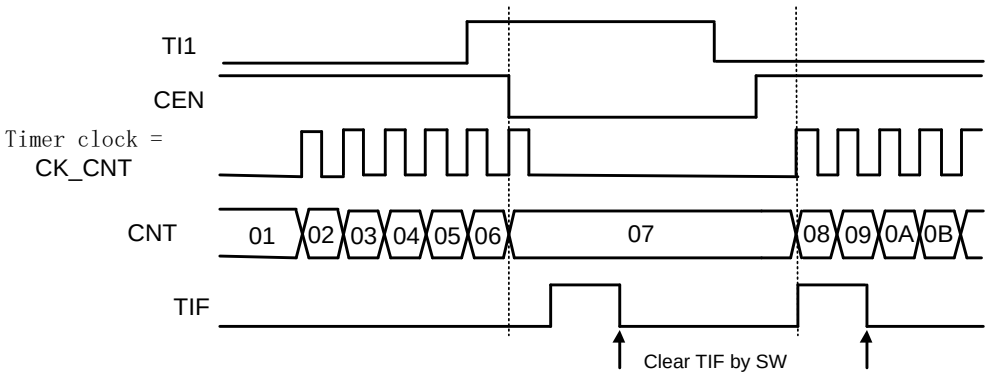


Figure 28-34 Timing in gated mode

Trigger mode

The counter only starts counting after an external input event arrives.

The configuration in the following figure is as follows:

- In the GPIO module, configure the corresponding pin as ATIM_CH1 function
- Turn off the channel enable, configure ATIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure ATIM_CCMR1.CC1S=01

- Select the effective edge of counting, configure ATIM_CCER.CC1P=0
- Select the trigger input signal, configure ATIM_SMCR.TS[2:0]=101, TI1FP1 as TRGI
- Set the slave mode controller to trigger mode, configure ATIM_SMCR.SMS[2:0]=110
- Turn on the channel enable, configure ATIM_CCER.CC1E=1

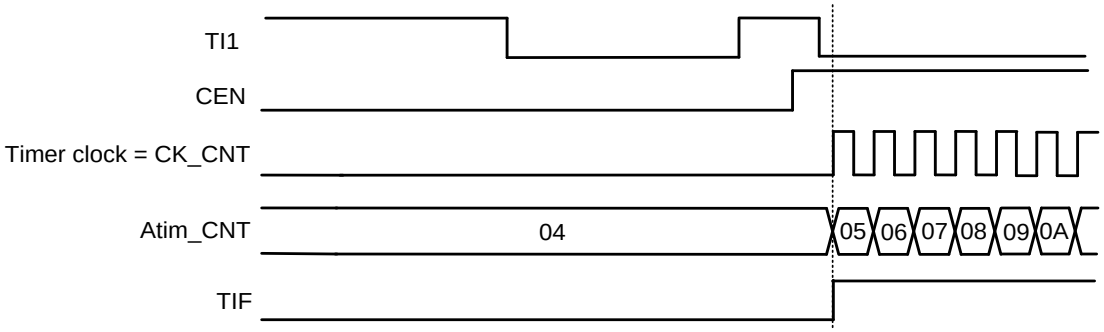


Figure 28-35 Timing in trigger mode

External clock counting mode triggered by external events

ETR can be set as the counting clock, while using another external input as the counter start trigger signal. For example, after detecting the rising edge of TI1, the counter starts counting with the rising edge of the ETR input.

The configuration in the following figure is as follows:

- In the GPIO module, configure the corresponding pin as ATIM_CH1, ATIM_ETR function
- Set ETP for edge selection, ATIM_SMCR.ETP=0
- Set the ETR division ratio, configure ATIM_SMCR.ETPS[1:0]=01
- Configure the input filter time, ATIM_SMCR.ETF[3:0]=0000
- Set bitECE register, enable external clock mode 2, ATIM_SMCR.ECE=1
- Turn off the channel enable, configure ATIM_CCER.CC1E=0 to ensure that the channel configuration is successful afterwards
- Select the input channel, configure ATIM_CCMR1.CC1S=01
- Select the effective edge of counting, configure ATIM_CCER.CC1P=0
- Select the trigger input signal, configure ATIM_SMCR.TS[2:0]=101, TI1FP1 as TRGI

- Set the slave mode controller to trigger mode, configure ATIM_SMCR.SMS[2:0]=110
- Turn on the channel enable, configure ATIM_CCER.CC1E=1

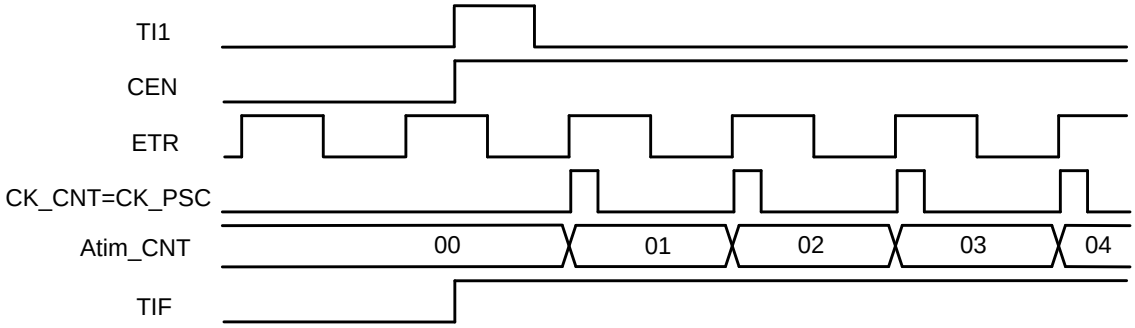


Figure 28-36 Timing in external clock mode 2 + trigger mode

28.4.14 DMA access

GPTIM supports 6 types of DMA requests, which are 4 CC channel requests, external trigger requests and user software trigger requests.

Each CC channel generates a DMA request, which is used to transfer the content in CCRx to RAM in capture mode, and is used to write data in RAM to CCRx in compare mode; the DMA request of the CC channel can be configured as Single transfer or burst transfer (CCxBURSTEN), single transfer only accesses the CCRx register, and burst transfer accesses a specific set of registers according to the DCR register configuration.

In addition, external trigger events and software trigger events can also generate DMA requests. When these two requests occur, DMA Burst transfer will be started, and data will be written to one or more registers in GPTIM, or one or more registers will be read from GPTIM. Multiple register values.

DMA request	CCxBURSTEN	DMA.CHxCTRL.DIR	DMA access object	Length of one transmission
GTIMx_CH1	0	0	Read CCR1	1
		1	Write CCR1	
	1	0	Read DMAR	DBL

DMA request	CCxBURSTEN	DMA.CHxCTRL.DIR	DMA access object	Length of one transmission
		1	Write DMAR	
GTIMx_CH2	0	0	Read CCR2	1
		1	Write CCR2	
	1	0	Read DMAR	DBL
		1	Write DMAR	
GTIMx_CH3	0	0	Read CCR3	1
		1	Write CCR3	
	1	0	Read DMAR	DBL
		1	Write DMAR	
GTIMx_CH4	0	0	Read CCR4	1
		1	Write CCR4	
	1	0	Read DMAR	DBL
		1	Write DMAR	
GTIMx_TRIG	N/A	0	Read DMAR	DBL
		1	Write DMAR	
GTIMx_UEV	N/A	0	Read DMAR	DBL
		1	Write DMAR	

28.4.15 DMA Burst

DMA-Burst supports one event to trigger multiple consecutive DMA requests. The main function is to continuously update the contents of multiple registers after the event occurs, so functions such as dynamic real-time adjustment of the output waveform can be realized.

The DMA controller needs to point the peripheral target address to a virtual register GPTIM_DMAR. When a specific timer event occurs, GPTIM will continuously transmit multiple DMA requests. Each DMA write operation to GPTIM_DMAR will be redirected to the actual function register by GPTIM.

The DBL register is used to set the DMA burst length, and the DBA register is used to set the base address of the DMA to access the GPTIM (relative to the offset of GPTIM_CR).

28.4.16 Input XOR function

The input signals of channels 1 to 3 can be XORed, and then connected to the filter and edge circuit input of channel 1 for input capture or triggering of channel 1.

The TI1Sbit of the GPTIM_CR2 register is used to select whether the input of channel 1 comes from the exclusive OR of the inputs of the three channels.

28.4.17 Debug mode

When Cortex-M0 enters the debug mode, the timer can stop or continue to work, and its behavior is defined by the DBG_TIMx_STOP register of the DCU module.

28.5 Register

Offset	Name	Symbol
GPTIM0(Base Address:0x40013800)		
0x00000000	GPTIM0 Control Register1	GPTIM0_CR1
0x00000004	GPTIM0 Control Register2	GPTIM0_CR2
0x00000008	GPTIM0 Slave Mode Control Register	GPTIM0_SMCR
0x0000000C	GPTIM0 DMA and Interrupt Enable Register	GPTIM0_DIER
0x00000010	GPTIM0 Interrupt Status Register	GPTIM0_ISR
0x00000014	GPTIM0 Event Generation Register	GPTIM0_EGR
0x00000018	GPTIM0 Capture/Compare Mode Register1	GPTIM0_CCMR1
0x0000001C	GPTIM0 Capture/Compare Mode Register2	GPTIM0_CCMR2
0x00000020	GPTIM0 Capture/Compare Enable Register	GPTIM0_CCER
0x00000024	GPTIM0 Counter Register	GPTIM0_CNT
0x00000028	GPTIM0 Prescaler Register	GPTIM0_PSC
0x0000002C	GPTIM0 Auto-Reload Register	GPTIM0_ARR
0x00000034	GPTIM0 Capture/Compare Register1	GPTIM0_CCR1
0x00000038	GPTIM0 Capture/Compare Register2	GPTIM0_CCR2
0x0000003C	GPTIM0 Capture/Compare Register3	GPTIM0_CCR3
0x00000040	GPTIM0 Capture/Compare Register4	GPTIM0_CCR4
0x00000048	GPTIM0 DMA Control Register	GPTIM0_DCR
0x0000004C	GPTIM0 DMA access Register	GPTIM0_DMAR
0x00000060	GPTIM0 Internal Trigger Select Register	GPTIM0_ITRSEL
GPTIM1(Base Address:0x40013C00)		
0x00000000	GPTIM1 Control Register1	GPTIM1_CR1
0x00000004	GPTIM1 Control Register2	GPTIM1_CR2
0x00000008	GPTIM1 Slave Mode Control Register	GPTIM1_SMCR
0x0000000C	GPTIM1 DMA and Interrupt Enable Register	GPTIM1_DIER
0x00000010	GPTIM1 Interrupt Status Register	GPTIM1_ISR
0x00000014	GPTIM1 Event Generation Register	GPTIM1_EGR
0x00000018	GPTIM1 Capture/Compare Mode Register1	GPTIM1_CCMR1
0x0000001C	GPTIM1 Capture/Compare Mode Register2	GPTIM1_CCMR2
0x00000020	GPTIM1 Capture/Compare Enable Register	GPTIM1_CCER
0x00000024	GPTIM1 Counter Register	GPTIM1_CNT
0x00000028	GPTIM1 Prescaler Register	GPTIM1_PSC
0x0000002C	GPTIM1 Auto-Reload Register	GPTIM1_ARR
0x00000034	GPTIM1 Capture/Compare Register1	GPTIM1_CCR1
0x00000038	GPTIM1 Capture/Compare Register2	GPTIM1_CCR2

Offset	Name	Symbol
0x0000003C	GPTIM1 Capture/Compare Register3	GPTIM1_CCR3
0x00000040	GPTIM1 Capture/Compare Register4	GPTIM1_CCR4
0x00000048	GPTIM1 DMA Control Register	GPTIM1_DCR
0x0000004C	GPTIM1 DMA access Register	GPTIM1_DMAR
0x00000060	GPTIM1 Internal Trigger Select Register	GPTIM1_ITRSEL

28.5.1 GPTIMx Control register 1（GPTIMx_CR1）

NAME	GPTIMx_CR1(x=0,1)							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						CKD	
access	U-0						R/W-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ARPE	CMS		DIR	OPM	URS	UDIS	CEN
access	R/W-0	R/W-00		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:10	-	RFU, Reserved, read as 0
9:8	CKD	Dead time and digital filter clock frequency divider register (the divider ratio relative to CK_INT) (Counter lock Divider) 00: tDTS=tCK_INT 01: tDTS=2*tCK_INT 10: tDTS=4*tCK_INT 11: RFU, forbidden to use
7	ARPE	Auto-Reload Preload Enable 0: ARR register does not enable preload 1: ARR register enables preload
6:5	CMS	Counter Mode Selection 00: Edge alignment mode 01: Center alignment mode 1, the output compare interrupt flag is only set when the counter is counting down 10: Center-aligned mode 2, the output compare interrupt flag is only set when the counter is counting up 11: Center-aligned mode 3, the output compare interrupt flag will be set when the counter is counting up and do

bit	name	functional description
4	DIR	counter Direction 0: count up 1: count down Note: When the timer is configured in central counting mode or encoder mode, this register is read-only
3	OPM	One Pulse Mode 0: The counter does not stop when the Update Event occurs 1: The counter stops when the Update Event occurs (automatically clears CEN)
2	URS	Update Request Selection 0: The following events can generate an update interrupt -Counter overflow or underflow -Software set bitUG register -The slave controller generates an update 1: Only counter overflow or underflow will generate update interrupt
1	UDIS	Update Disable 0: enable the update event; update events are generated when the following events occur -Counter overflow or underflow -Software set bitUG register -The slave controller generates an update 1: Disable the update event and do not update the shadow register. When the UG sets the bit or the slave controller receives the hardware reset, the counter and prescaler are reinitialized.
0	CEN	Counter Enable 0: The counter is off 1: Counter enable Note: The external trigger mode can automatically set bitCEN

28.5.2 GPTIMx Control register 2（GPTIMx_CR2）

NAME	GPTIMx_CR2(x=0,1)							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							

access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TI1S	MMS			CCDS	-		
access	R/W-0	R/W-000			R/W-0	U-0		

bit	name	functional description
31:8	-	RFU, Reserved, read as 0
7	TI1S	Timer Input 1 Selection 0: GPTIMx_CH1 input channel 1 1: GPTIMx_CH1, CH2, CH3 enter channel 1 after exclusive OR
6:4	MMS	Master mode selection, used to configure the source of the synchronous trigger signal (TRGO) sent to the slave in the master mode (Master Mode Selection) 000: UG register of GPTIM_EGR is used as TRGO 001: Counter enable signal CNT_EN is used as TRGO, which can be used to start multiple timers at the same time 010: UE (update event) signal is used as TRGO 011: comparison pulse, if the CC1IF flag is about to be set, TRGO outputs a positive pulse 100: OC1REF is used as TRGO 101: OC2REF is used as TRGO 110: OC3REF is used as TRGO 111: OC4REF is used as TRGO Note: The slave timer or ADC must enable the working clock in advance to receive the TRGO sent by the master timer
3	CCDS	Capture/Compare DMA Selection 0: Send a DMA request when a capture/compare event occurs 1: Send DMA request when Update Event occurs
2:0	-	RFU, Reserved, read as 0

28.5.3 GPTIMx Slave mode control register（GPTIMx_SMCR）

NAME	GPTIMx_SMCR(x=0,1)							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	ETP	ECE	ETPS		ETF			
access	R/W-0	R/W-0	R/W-00		R/W-0000			

bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	MSM	TS			-	SMS		
access	R/W-0	R/W-000			U-0	R/W-000		

bit	name	functional description
31:16	-	RFU：Reserved, read as 0
15	**ETP**	External Trigger Polarity 0: High level or rising edge valid 1: Low level or falling edge valid
14	**ECE**	External Clock Enable 0: Turn off external clock mode 2 1: Enable external clock mode 2, the counter clock is the valid edge of ETRF
13:12	**ETPS**	External Trigger Prescaler The frequency of the external trigger signal ETRP can only be at most 1/4 of the GPTIM working clock. When the input signal frequency is high, prescaler can be used. 00: no frequency division 01: divide by 2 10: 4 frequency division 11: 8 frequency division
11:8	**ETF**	External Trigger Filter 0000：No filtering 0001：$f_{SAMPLING}=f_{CK\_INT}$, N=2 0010：$f_{SAMPLING}=f_{CK\_INT}$, N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$, N=8 0100：$f_{SAMPLING}=f_{DTS/2}$, N=6 0101：$f_{SAMPLING}=f_{DTS/2}$, N=8 0110：$f_{SAMPLING}=f_{DTS/4}$, N=6 0111：$f_{SAMPLING}=f_{DTS/4}$, N=8 1000：$f_{SAMPLING}=f_{DTS/8}$, N=6 1001：$f_{SAMPLING}=f_{DTS/8}$, N=8 1010：$f_{SAMPLING}=f_{DTS/16}$, N=5 1011：$f_{SAMPLING}=f_{DTS/16}$, N=6 1100：$f_{SAMPLING}=f_{DTS/16}$, N=8 1101：$f_{SAMPLING}=f_{DTS/32}$, N=5 1110：$f_{SAMPLING}=f_{DTS/32}$, N=6 1111：$f_{SAMPLING}=f_{DTS/32}$, N=8
7	**MSM**	Master Slave Mode 0: No action 1: The action triggered by TRGI is delayed so that the current timer is perfectly synchronized with its slave timer (via TRGO). This setting is suitable for a single external event to synchronize multiple timers.

bit	name	functional description
6:4	**TS**	Trigger Source 000: Internal trigger signal (ITR0) 001: Internal trigger signal (ITR1) 010: Internal trigger signal (ITR2) 011: Internal trigger signal (ITR3) 100: TI1 edge detection (TI1F_ED) 101: TI1 after filtering (TI1FP1) 110: TI2 after filtering (TI2FP2) 111: External trigger input (ETRF) Note: The TS register can be rewritten only when the SMS=000 means that the slave mode is disabled.
3	**-**	RFU：Reserved, read as 0
2:0	**SMS**	Slave Mode Selection 000: Slave mode is disabled; after CEN is enabled, the prescaler circuit clock source comes from the internal clock 001: Encoder mode 1; the counter uses TI2FP1 edge and counts according to the level of TI1FP2 010: Encoder mode 2; the counter uses TI1FP2 edge and counts according to the level of TI2FP1 011: Encoder mode 3; the counter uses TI1FP1 and TI2FP2 edges at the same time, and counts according to other input signal levels 100: Multiple bit mode; the rising edge of TRGI initializes the counter and triggers the register update 101: Gate mode; when TRGI is high, the counting clock is enabled, when TRGI is low, the counting clock is stopped 110: Trigger mode; TRGI rising edge triggers the counter to start counting (bit counter will not be reset) 111: External clock mode 1; the rising edge of TRGI directly drives the counter

28.5.4 GPTIMx DMA and interrupt enable register（GPTIMx_DIER）

NAME	GPTIMx_DIER(x=0,1)							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-				CC4BURSTEN	CC3BURSTEN	CC2BURSTEN	CC1BURSTEN
access	U-0				R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

name	-	TDE	-	CC4DE	CC3DE	CC2DE	CC1DE	UDE
access	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	TIE	-			CC2IE	CC1IE	UIE
access	U-0	R/W-0	U-0			R/W-0	R/W-0	R/W-0

bit	name	functional description
31:20	-	RFU：Reserved, read as 0
19	**CC4BURSTEN**	CC4 Burst Enable 0: Single mode, only access CCR 1: Burst mode, configure access address and length through DCR
18	**CC3BURSTEN**	CC3 Burst Enable 0: Single mode, only access CCR 1: Burst mode, configure access address and length through DCR
17	**CC2BURSTEN**	CC2 Burst Enable 0: Single mode, only access CCR 1: Burst mode, configure access address and length through DCR
16	**CC1BURSTEN**	CC1 Burst Enable 0: Single mode, only access CCR 1: Burst mode, configure access address and length through DCR
15	-	RFU：Reserved, read as 0
14	**TDE**	Triggered DMA Enable 0: In slave mode, forbid external trigger event to generate DMA request 1: In slave mode, allow external trigger events to generate DMA requests (can be used to automatically update the preload register)
13	-	RFU：Reserved, read as 0
12	**CC4DE**	CC4 DMA Enable 0: Disable CC4 DMA request 1: Allow CC4 DMA request
11	**CC3DE**	CC3 DMA Enable 0: Disable CC3 DMA request 1: Allow CC3 DMA request
10	**CC2DE**	CC2 DMA Enable 0: Disable CC2 DMA request 1: Allow CC2 DMA request
9	**CC1DE**	CC1 DMA Enable 0: Disable CC1 DMA request 1: Allow CC1 DMA request

bit	name	functional description
8	**UDE**	Update event DMA Enable 0: When Update Event occurs, DMA request is prohibited 1: When Update Event occurs, DMA request is allowed
7	**-**	RFU：Reserved, read as 0
6	**TIE**	Trigger event Interrupt Enable 0: Disable trigger event interrupt 1: Allow to trigger event interrupt
5	**-**	RFU：Reserved, read as 0
4	**CC3IE**	CC4 Interrupt Enable 0: Disable capture/compare 4 interrupt 1: Allow capture/compare 4 interrupt
3	**CC3IE**	CC3 Interrupt Enable 0: Disable capture/compare 3 interrupt 1: Allow capture/compare 3 interrupt
2	**CC2IE**	CC2 Interrupt Enable 0: Disable capture/compare 2 interrupt 1: Enable capture/compare 2 interrupt
1	**CC1IE**	CC1 Interrupt Enable 0: Disable capture/compare 1 interrupt 1: Enable capture/compare 1 interrupt
0	**UIE**	Update event Interrupt Enable 0: Disable Update event interrupt 1: Allow Update event interrupt

28.5.5 GPTIMx Status register（GPTIMx_ISR）

NAME	GPTIMx_ISR(x=0,1)							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			CC4OF	CC3OF	CC2OF	CC1OF	-
access	U-0			R/W-0	R/W-0	R/W-0	R/W-0	U-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	TIF	-	CC4IF	CC3IF	CC2IF	CC1IF	UIF
access	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:13	**-**	RFU：Reserved, read as 0

bit	name	functional description
12	**CC4OF**	Over-Capture Interrupt Flag for CC4, write 1 to clear Refer toCC1OF
11	**CC3OF**	Over-Capture Interrupt Flag for CC3, write 1 to clear Refer toCC1OF
10	**CC2OF**	Over-Capture Interrupt Flag for CC2, write 1 to clear Refer toCC1OF
9	**CC1OF**	Over-Capture Interrupt Flag for CC1, write 1 to clear This register is only valid when the corresponding channel is set to input capture mode. The hardware sets the bit, and the software writes 1 to clear it. 0: no overcapture event 1: A new capture occurs when the CC1IF flag is 1
8:7	**-**	RFU：Reserved, read as 0
6	**TIF**	Trigger event Interrupt Flag, write 1 to clear
5	**-**	RFU：Reserved, read as 0
4	**CC4IF**	CC4 Interrupt Flag, write 1 to clear Refer toCC1IF
3	**CC3IF**	CC3 Interrupt Flag, write 1 to clear Refer toCC3IF
2	**CC2IF**	CC2 Interrupt Flag, write 1 to clear Refer toCC2IF
1	**CC1IF**	CC1 Interrupt Flag, write 1 to clear If the CC1 channel is configured as an output: CC1IF is set when the count value is equal to the comparison value, and the software writes 1 to clear it. If the CC1 channel is configured as an input: bit is set when a capture event occurs, the software writes 1 to clear it, or the software reads ATIM_CCR1 to automatically clear it.
0	**UIF**	Update event Interrupt Flag,write 1 to clear When the following events occur, the UIF bit is set and the shadow register is updated -When the repeat counter=0 and UDIS=0, the counter overflows -In the case of URS=0 and UDIS=0, the software sets the bitUG register to initialize the counter -In the case of URS=0 and UDIS=0, the trigger event initializes the counter

28.5.6 GPTIMx Event generation register（GPTIMx_EGR）

NAME	GPTIMx_EGR(x=0,1)							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							

access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	TG	-			CC2G	CC1G	UG
access	U-0	W-0	U-0			W-0	W-0	W-0

bit	name	functional description
31:7	-	RFU：Reserved, read as 0
6	TG	Software trigger, software sets this register to generate a trigger event, hardware automatically clears (Trigger Generate)
5:3	-	RFU：Reserved, read as 0
2	CC2G	Capture/compare channel 2 software trigger, Refer toCC1G (CC2 Generate)
1	CC1G	Capture/Compare Channel 1 Software Trigger (CC1 Generate)
0	UG	If the CC1 channel is configured as an output: CC1IF is set to bit, and the corresponding interrupt and DMA request can be generated when it is enabled

28.5.7 GPTIMx Capture/compare mode register1（GPTIMx_CCMR1）

This register is multiplexed into two different functions under output compare and input capture configuration.

NAME	GPTIMx_CCMR1(x=0,1)							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	OC2CE	OC2M			OC2PE	OC2FE	CC2S	
	IC2F				IC2PSC		CC2S	
access	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	OC1CE	OC1M			OC1PE	OC1FE	CC1S	
	IC1F				IC1PSC		CC1S	
access	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-00	

Output compare mode

bit	name	functional description
31:16	-	RFU：Reserved, read as 0
15	**OC2CE**	OC2 Clear Enable，Refer toOC1CE
14:12	**OC2M**	OC2 Mode，Refer toOC1M
11	**OC2PE**	OC2 Preload Enable，Refer toOC1PE
10	**OC2FE**	OC2 Fast Enable，Refer toOC1FE
9:8	**CC2S**	CC2 channel Selection 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped to TI2 10: CC2 channel is configured as input, IC2 is mapped to TI1 11: CC2 channel is configured as input, IC2 is mapped to TRC Note: CC2S can only be written when the channel is closed (CC2E=0) OC2 Clear Enable
7	**OC1CE**	OC2 Clear Enable 0: OC1REF is not affected by ETRF 1: OC1REF is automatically cleared when ETRF high level is detected
6:4	**OC1M**	Output compare 1 mode configuration, this register defines the behavior of the OC1REF signal (OC1 Mode) 000: The comparison result of the output compare register CCR1 and the counter CNT will not affect the output 001: When CCR1=CNT, set OC1REF high 010: When CCR1=CNT, set OC1REF low 011: When CCR1=CNT, flip OC1REF 100: OC1REF is fixed to low (inactive) 101: OC1REF is fixed to high (active) 110: PWM mode 1-When counting up, OC1REF is set high when CNT<CCR1, otherwise it is set low; when counting down, OC1REF is set low when CNT>CCR1, otherwise it is set high 111: PWM mode 2-When counting up, OC1REF is set low when CNT<CCR1, otherwise it is set high; when counting down, OC1REF is set high when CNT>CCR1, otherwise it is set low
3	**OC1PE**	OC1 Preload Enable 0: CCR1 preload register is invalid, CCR1 can be written directly 1: The CCR1 preload register is valid. The read and write operations for CCR1 are all access to the preload register, and the content of the preload register is transferred to the shadow register when an update event occurs
2	**OC1FE**	OC1 Fast Enable 0: Turn off the fast enable, the trigger input will not affect the

bit	name	functional description
		comparison output 1: Turn on the fast enable, the trigger input will immediately change OC1REF to the output when the comparison value matches, regardless of the current actual comparison situation This function is only valid when the current channel is configured in PWM1 or PWM2 mode
1:0	**CC1S**	CC1 channel Selection 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped to TI1 10: CC1 channel is configured as input, IC1 is mapped to TI2 11: CC1 channel is configured as input, IC1 is mapped to TRC Note: CC1S can only be written when the channel is closed (CC1E=0)

Input capture mode

bit	name	functional description
31:16	**-**	RFU：Reserved, read as 0
15:12	**IC2F**	IC2 Filter
11:10	**IC2PSC**	IC2 Prescaler
9:8	**CC2S**	Capture/Compare2 channel Selection 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC3 is mapped to TI2 10: CC2 channel is configured as input, IC3 is mapped to TI1 11: CC2 channel is configured as input, IC3 is mapped to TRC Note: CC2S can only be written when the channel is closed (CC2E=0)
7:4	**IC1F**	IIC1 Filter This register defines the sampling frequency and filter length of TI1 0000: No filtering, sampling using fDTS 0001：$f_{SAMPLING}=f_{CK\_INT}$, N=2 0010：$f_{SAMPLING}=f_{CK\_INT}$, N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$, N=8 0100：$f_{SAMPLING}=f_{DTS/2}$, N=6 0101：$f_{SAMPLING}=f_{DTS/2}$, N=8 0110：$f_{SAMPLING}=f_{DTS/4}$, N=6 0111：$f_{SAMPLING}=f_{DTS/4}$, N=8 1000：$f_{SAMPLING}=f_{DTS/8}$, N=6 1001：$f_{SAMPLING}=f_{DTS/8}$, N=8 1010：$f_{SAMPLING}=f_{DTS/16}$, N=5 1011：$f_{SAMPLING}=f_{DTS/16}$, N=6 1100：$f_{SAMPLING}=f_{DTS/16}$, N=8

bit	name	functional description
		1101：$f_{SAMPLING}=f_{DTS/32}$, N=5 1110：$f_{SAMPLING}=f_{DTS/32}$, N=6 1111：$f_{SAMPLING}=f_{DTS/32}$, N=8
3:2	**IC1PSC**	IC1 Prescaler 00: no frequency division 01: Capture once every 2 event inputs 10: A capture is generated every 4 event inputs 11: A capture is generated every 8 event inputs IC1PSC register is reset when CC1E=0
1:0	**CC1S**	Capture/Compare1 channel Selection 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped to TI1 10: CC1 channel is configured as input, IC1 is mapped to TI2 11: CC1 channel is configured as input, IC1 is mapped to TRC Note: CC1S can only be written when the channel is closed (CC1E=0)

28.5.8 GPTIMx Capture/compare mode register2（GPTIMx_CCMR2）

This register is multiplexed into two different functions under output compare and input capture configuration

NAME	GPTIMx_CCMR2(x=0,1)							
Offset	0x0000001C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	OC4CE	OC4M			OC4PE	OC4FE	CC4S	
	IC2F				IC2PSC		CC4S	
access	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	OC3CE	OC3M			OC3PE	OC3FE	CC3S	
	IC3F				IC3PSC		CC3S	
access	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

Output compare mode

bit	name	functional description
31:16	**-**	RFU：Reserved, read as 0

bit	name	functional description
15	**OC4CE**	OC4 Clear Enable，Refer toOC3CE
14:12	**OC4M**	OC4 Mode，Refer toOC3M
11	**OC4PE**	OC4 Preload Enable，Refer toOC3PE
10	**OC4FE**	OC4 Fast Enable，Refer toOC3FE
9:8	**CC4S**	CC4 channel Selection 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped to TI4 10: CC4 channel is configured as input, IC4 is mapped to TI3 11: CC4 channel is configured as input, IC4 is mapped to TRC Note: CC4S can only be written when the channel is closed (CC4E=0)
7	**OC3CE**	OC3 Clear Enable 0: OC1REF is not affected by ETRF 1: OC1REF is automatically cleared when ETRF high level is detected
6:4	**OC3M**	Output compare 3 mode configuration, this register defines the behavior of the OC3REF signal (OC3 Mode) 000: The comparison result of the output compare register CCR3 and the counter CNT will not affect the output 001: When CCR3=CNT, set OC1REF high 010: When CCR3=CNT, set OC1REF low 011: When CCR3=CNT, flip OC1REF 100: OC3REF is fixed to low (inactive) 101: OC3REF is fixed to high (active) 110: PWM mode 1-when counting up, OC3REF is set high when CNT<CCR3, otherwise it is set low; when counting down, OC3REF is set low when CNT>CCR3, otherwise it is set high 111: PWM mode 2-when counting up, OC3REF is set low when CNT<CCR3, otherwise it is set high; when counting down, OC3REF is set high when CNT>CCR3, otherwise it is set low
3	**OC3PE**	OC3 Preload Enable 0: CCR3 preload register is invalid, CCR3 can be written directly 1: The CCR3 preload register is valid. The read and write operations for CCR3 all access the preload register, and the content of the preload register is transferred to the shadow register when the update event occurs
2	**OC3FE**	OC3 Fast Enable 0: Turn off the fast enable, the trigger input will not affect the comparison output 1: Turn on fast enable, the trigger input will immediately change OC3REF to the output when the comparison value matches, regardless of the current actual comparison situation This function is only valid when the current channel is

bit	name	functional description
		configured in PWM1 or PWM2 mode
1:0	**CC3S**	CC3 channel Selection 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC1 is mapped to TI3 10: CC3 channel is configured as input, IC1 is mapped to TI4 11: CC3 channel is configured as input, IC1 is mapped to TRC Note: CC3S can only be written when the channel is closed (CC3E=0)

Input capture mode

bit	name	functional description
31:16	**-**	RFU：Reserved, read as 0
15:12	**IC4F**	IC4 Filter
11:10	**IC4PSC**	C4 Prescaler
9:8	**CC4S**	CC4 channel Selection 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped to TI4 10: CC4 channel is configured as input, IC4 is mapped to TI3 11: CC4 channel is configured as input, IC4 is mapped to TRC Note: CC4S can only be written when the channel is closed (CC4E=0)
7:4	**IC3F**	IC3 Filter This register defines the sampling frequency and filter length of TI3 0000: No filtering, sampling usingf_{DTS} 0001：$f_{SAMPLING}=f_{CK\_INT}$, N=2 0010：$f_{SAMPLING}=f_{CK\_INT}$, N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$, N=8 0100：$f_{SAMPLING}=f_{DTS/2}$, N=6 0101：$f_{SAMPLING}=f_{DTS/2}$, N=8 0110：$f_{SAMPLING}=f_{DTS/4}$, N=6 0111：$f_{SAMPLING}=f_{DTS/4}$, N=8 1000：$f_{SAMPLING}=f_{DTS/8}$, N=6 1001：$f_{SAMPLING}=f_{DTS/8}$, N=8 1010：$f_{SAMPLING}=f_{DTS/16}$, N=5 1011：$f_{SAMPLING}=f_{DTS/16}$, N=6 1100：$f_{SAMPLING}=f_{DTS/16}$, N=8 1101：$f_{SAMPLING}=f_{DTS/32}$, N=5 1110：$f_{SAMPLING}=f_{DTS/32}$, N=6 1111：$f_{SAMPLING}=f_{DTS/32}$, N=8
3:2	**IC3PSC**	IC3 Prescaler 00: no frequency division

bit	name	functional description
		01: Capture once every 2 event inputs 10: A capture is generated every 4 event inputs 11: A capture is generated every 8 event inputs IC1PSC register is reset when CC1E=0
1:0	**CC3S**	CC3 channel Selection 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC1 is mapped to TI3 10: CC3 channel is configured as input, IC1 is mapped to TI4 11: CC3 channel is configured as input, IC1 is mapped to TRC Note: CC1S can only be written when the channel is closed (CC1E=0)

28.5.9 GPTIMx Capture/compare enable register（GPTIMx_CCER）

NAME	GPTIMx_CCER(x=0,1)							
Offset	0x00000020							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-		CC4P	CC4E	-		CC3P	CC3E
access	U-0		R/W-0	R/W-0	U-0		R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-		CC2P	CC2E	-		CC1P	CC1E
access	U-0		R/W-0	R/W-0	U-0		R/W-0	R/W-0

bit	name	functional description
31:14	**-**	RFU：Reserved, read as 0
13	**CC4P**	CC4 Polarity，Refer toCC1P
12	**CC4E**	CC4 output Enable，Refer toCC1E
11:10	**-**	RFU：Reserved, read as 0
9	**CC3P**	CC3 Polarity，Refer toCC1P
8	**CC3E**	CC3 output Enable，Refer toCC1E
7:6	**-**	RFU：Reserved, read as 0
5	**CC2P**	CC2 Polarity，Refer toCC1P
4	**CC2E**	CC2 output Enable)，Refer toCC1E
3:2	**-**	RFU：Reserved, read as 0
1	**CC1P**	CC1 Polarity When CC1 channel is configured as output: 0: OC1 high effective

bit	name	functional description
		1: OC1 low effective When CC1 channel is configured as input: CC1NP/CC1P is used to select the polarity of TI1FP1 and TI2FP1 00: non-inverted/rising edge 01: Inverted/falling edge 10: Reserve, do not use 11: Non-inverted, both upper and lower edges are valid
0	**CC1E**	CC1 output Enable When CC1 channel is configured as output 0: OC1 output is off, Ocx=0, Ocx_EN=0 1: Ocx=OCxREF+polar selection, Ocx_EN=1 When CC1 channel is configured as input 0: Turn off the capture function 1: Enable the capture function

Output control bit of standard Ocx channel

CcxEbit	Ocx output state
0	Prohibit output (Ocx=0, Ocx_EN=0)
1	Ocx=OCxREF + polarity, Ocx_EN=1

28.5.10 GPTIMxCounter register（GPTIMx_CNT）

NAME	GPTIMx_CNT(x=0,1)							
Offset	0x00000024							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CNT[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CNT[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	**-**	RFU：Reserved, read as 0
15:0	**CNT**	Counter

28.5.11 GPTIMx Prescaler register（GPTIMx_PSC）

NAME	GPTIMx_PSC(x=0,1)							
offset	0x00000028 + x*0x400							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	PSC[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	PSC[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU：Reserved, read as 0
15:0	PSC	CK_CNTCounter Clock Prescaler) $f_{CK\_CNT}=f_{CK\_PSC}/(PSC[15:0]+1)$ This is a preload register, and its content is loaded into the shadow register when the update event occurs

28.5.12 GPTIMx Auto-reload register（GPTIMx_ARR）

NAME	GPTIMx_ARR(x=0,1)							
Offset	0x0000002C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	ARR[15:8]							
access	R/W-1111 1111							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ARR[7:0]							
access	R/W-1111 1111							

bit	name	functional description
31:16	-	RFU：Reserved, read as 0

bit	name	functional description
15:0	**ARR**	Auto-Reload Register This is a preload register, and its content is loaded into the shadow register when the update event occurs

28.5.13 GPTIMx Capture/compare register1（GPTIMx_CCR1）

NAME	GPTIMx_CCR1(x=0,1)							
Offset	0x00000034							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CCR1[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CCR1[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	**-**	RFU：Reserved, read as 0
15:0	**CCR1**	Capture/Compare channel 1 Register **If channel 1 is configured as output:** This is a preload register whose content is loaded into the shadow register and used to compare with the counter to generate OC1 output **If channel 1 is configured as input:** CCR1 saves the counter value when the most recent input capture event occurred, at this time CCR1 is read-only

28.5.14 GPTIMx Capture/compare register2（GPTIMx_CCR2）

NAME	GPTIMx_CCR2(x=0,1)							
Offset	0x00000038							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							

bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CCR2[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CCR2[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU：Reserved, read as 0
15:0	**CCR2**	Capture/Compare channel 2 Register **If channel 2 is configured as output:** This is a preload register whose content is loaded into the shadow register and used to compare with the counter to generate OC2 output **If channel 2 is configured as input:** CCR2 saves the counter value when the most recent input capture event occurred, at this time CCR1 is read-only

28.5.15 GPTIMx Capture/compare register3（GPTIMx_CCR3）

NAME	GPTIMx_CCR3(x=0,1)							
Offset	0x0000003C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CCR3[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CCR3[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU：Reserved, read as 0
15:0	**CCR3**	Capture/Compare channel 3 Register **If channel 3 is configured as output:** This is a preload register whose content is loaded into the shadow register and used to compare with the counter to generate OC3 output

bit	name	functional description
		If channel 3 is configured as input: CCR3 saves the counter value when the most recent input capture event occurred, at this time CCR13is read-only

28.5.16 GPTIMx Capture/compare register4（GPTIMx_CCR4）

NAME	GPTIMx_CCR4(x=0,1)							
Offset	0x00000040							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CCR4[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CCR4[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU：Reserved, read as 0
15:0	**CCR4**	Capture/Compare channel 4 Register **If channel 4 is configured as output:** This is a preload register whose content is loaded into the shadow register and used to compare with the counter to generate OC4 output **If channel 4 is configured as input:** CCR4 saves the counter value when the most recent input capture event occurred, at this time CCR4 is read-only

28.5.17 GPTIMx DMA control register（GPTIMx_DCR）

NAME	GPTIMx_DCR(x=0,1)							
Offset	0x00000048							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							

bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			DBL				
access	U-0			R/W-0 0000				
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-			DBA				
access	U-0			R/W-0 0000				

bit	name	functional description
31:13	-	RFU：Reserved, read as 0
12:8	**DBL**	DMA Burst Length Reading and writing to the GPTIM_DMAR register will trigger the burst DMA operation, the burst length is 1~18 00000: length=1 00001: length=2 00010: length=3 00011: length=4 00100: length=5 00101: length=6 00110: length=7 00111: length=8 01000: length=9 01001: length=10 01010: length=11 01011: length=12 01100: length=13 01101: length=14 01110: length=15 01111: length=16 10000: length=17 10001: length=18 Other: invalid value, write prohibited
7:5	-	RFU：Reserved, read as 0
4:0	**DBA**	DMA Burst offset Address 00000：GPTIM_CR1 00001：GPTIM_CR2 00010：GPTIM_SMCR …… Note: When DBA+DBL exceeds the GPTIM register address range, the actual burst will automatically stop after being transferred to the highest register address of GPTIM, that is, the burst length will be shortened.

28.5.18 GPTIMx DMA access register（GPTIMx_DMAR）

NAME	GPTIMx_DMAR(x=0,1)							
Offset	0x0000004C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DMAR[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DMAR[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU：Reserved, read as 0
15:0	**DMAR**	DMA burst access register When using DMA burst transfer, set the DMA channel peripheral address to GPTIM_DMAR, GPTIM will generate multiple DMA requests according to the value of DBL

28.5.19 GPTIMx ITR Select register（GPTIMx_ITRSEL）

NAME	GPTIMx_ITRSEL(x=0,1)							
Offset	0x00000060							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ITR3SEL		ITR2SEL		ITR1SEL		ITR0SEL	
access	R/W-00		R/W-00		R/W-00		R/W-00	

bit	name	functional description
31:8	-	RFU, Reserved, read as 0
7:6	ITR3SEL	Internal Trigger Source Selection) Capture of internal trigger signal (ITRx) For details, see 28.4.4 Capture of Internal Trigger Signal (ITRx)
5:4	ITR2SEL	
3:2	ITR1SEL	
1:0	ITR0SEL	

29 Basic Timer (BSTIM32)

29.1 Introduction

FM33LE0xxA contains 1 basic timer

The basic timer includes a 32bit auto-reload counter and a programmable prescaler.

The basic timer is mainly used to generate the system time base, and can also generate trigger events to drive ADC sampling.

29.2 Main features

- 32bit up counting automatic reload counter
- 32bit programmable prescaler, support real-time adjustment of counting clock frequency division
- ADC timing trigger function
- Interrupt when the counter overflows

29.3 Block Diagram

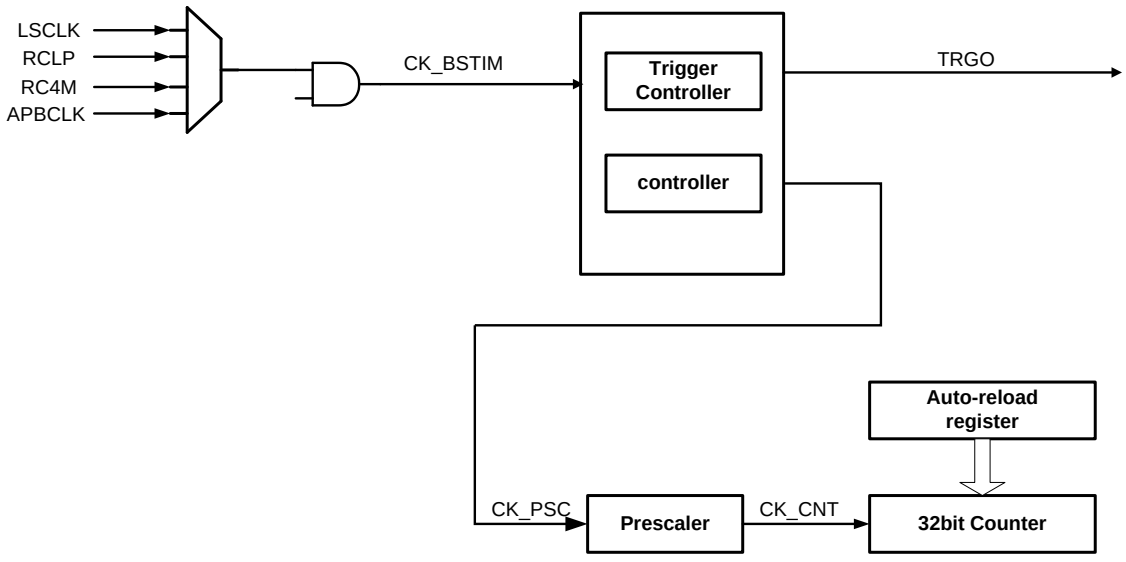


Figure 29-1 BSTIME32 block diagram

29.4 Functional Description

29.4.1 Timing Unit

The timing unit of the basic timer consists of a 32bit counter and an automatic reload register. The counter counts up. The counting clock can be obtained after dividing the APBCLK by a 16bit prescaler.

The counter, auto-reload register and prescaler register can all be rewritten or read by software, even when the counter is running.

The timing unit contains the following registers:

- Counter (BSTIM_CNT)
- Prescaler register (BSTIM_PSC)
- Automatic reload register (BSTIM_ARR)

ARR includes a preload function, software reads and writes ARR can take effect directly, or just access its cache, controlled by ARPE (Auto Reload Preload Enable) register. When ARPE=1, the software reads and writes ARR to access its cache register. When an update event (BSTIM_CNT overflow) occurs, the data in the cache register will be updated to ARR. Software can also actively trigger ARR updates through register operations.

The BSTIM_CNT working clock is driven by the frequency division clock generated by BSTIM_PSC. CNT starts counting only when the counter enable register (CEN) is set. When CNT=ARR, this round of counting ends, and an update event is sent.

BSTIM_PSC is a synchronous prescaler that can divide APBCLK from 1 to 65536. The PSC register is also cached. To rewrite the PSC is actually to rewrite the cache register. Only when a new update event arrives will the PSC be updated from the cache register. Therefore, in the CNT counting process, the software can rewrite the PSC in real time.

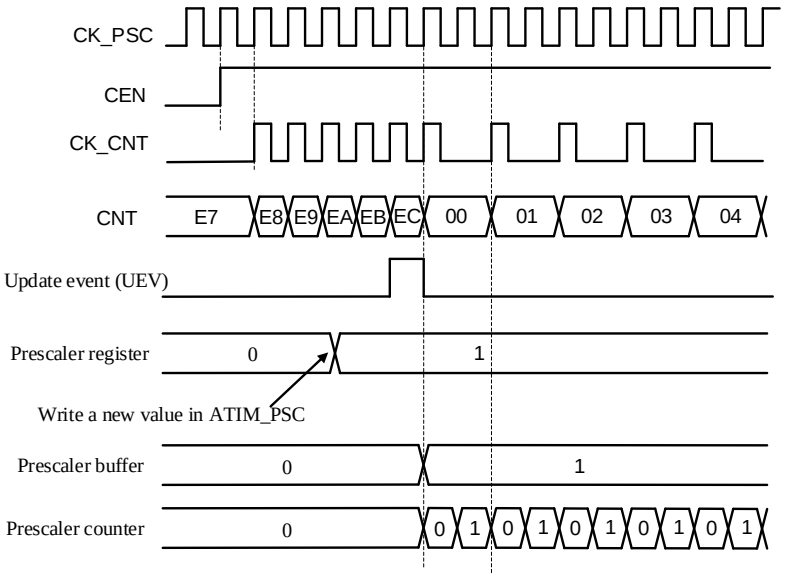


Figure 29-2 Waveform with prescaled frequency changing from1 to 2

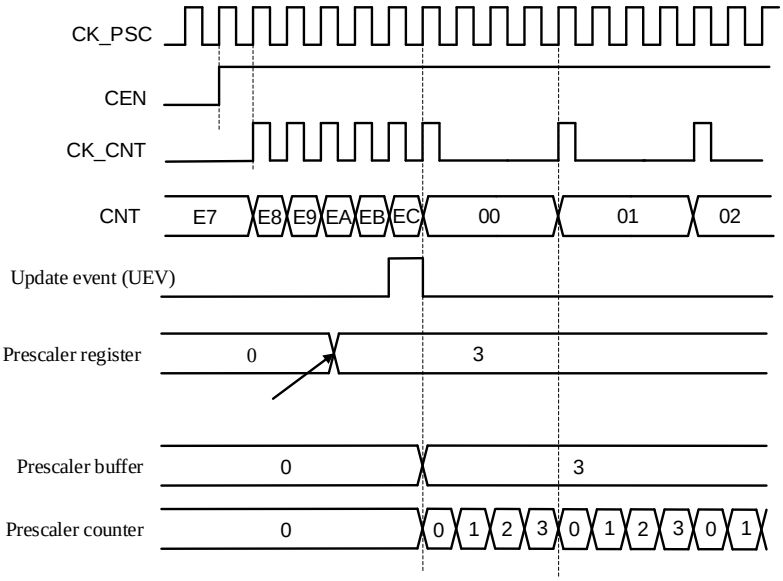


Figure 29-3 Waveform with prescaled frequency changing from 1 to 4

29.4.2 Timer Operation Mode

The general-purpose timer only supports up-counting mode.

Count up

In this mode, the counter starts counting from 0 after being enabled, until CNT=ARR, an overflow event is generated, and then starts counting from 0 again.

The software can directly trigger the update event by setting the UG register, and the CNT and prescaler counter are automatically cleared at this time. Setting the UG register will not trigger the UIF (Update Interrupt Flag) interrupt flag to be set.

The update event can be disabled by setting the UDIS register, which can avoid updating the value in the preload register to the working register.

When an update event occurs, the following registers are updated and the UIF bit is set:

- BSTIM_RCR is updated to the value in the cache
- BSTIM_ARR is updated to the value in the cache
- BSTIM_PSC is updated to the value in the cache

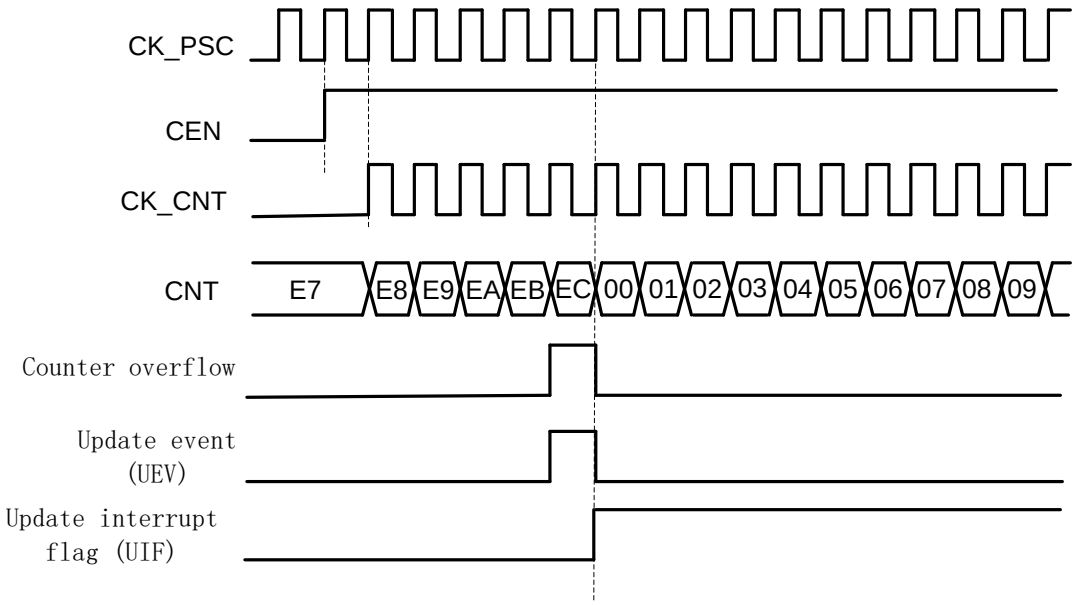


Figure 29-4 Up-counting waveform, no frequency division of internal clock

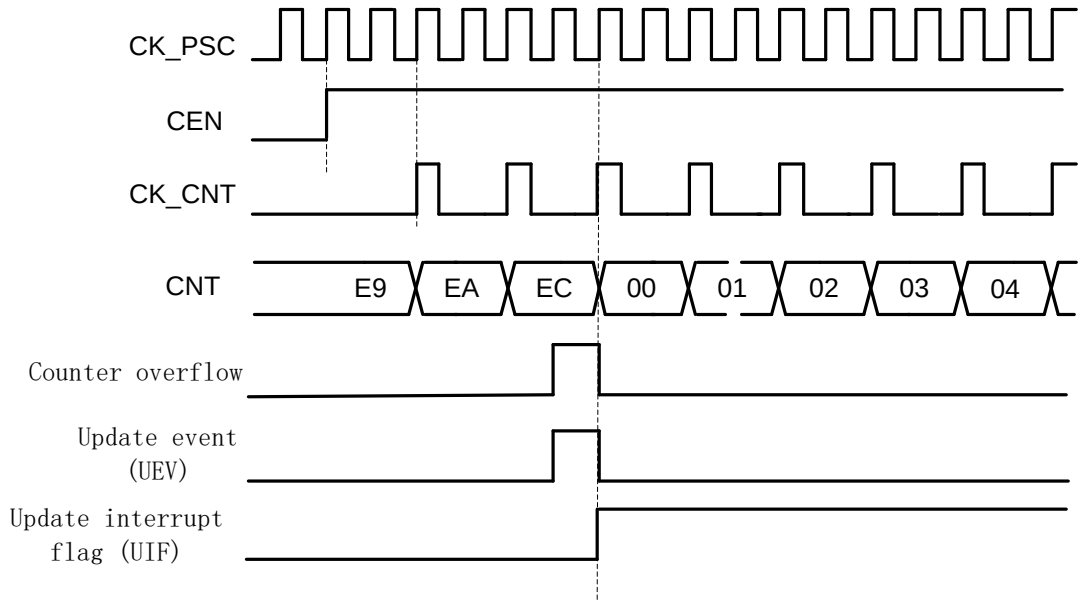


Figure 29-5 Up-counting waveform, internal clock divided by 2

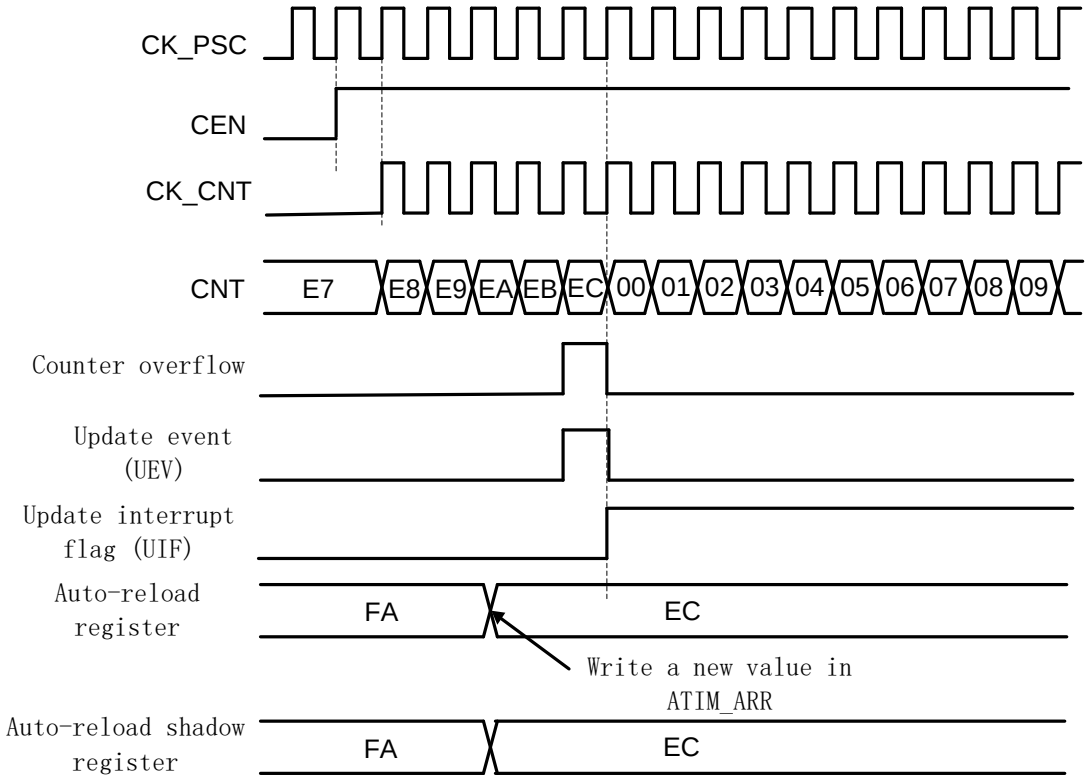


Figure 29-6 Updata event when APPE=0 (ARR is not preloaded)

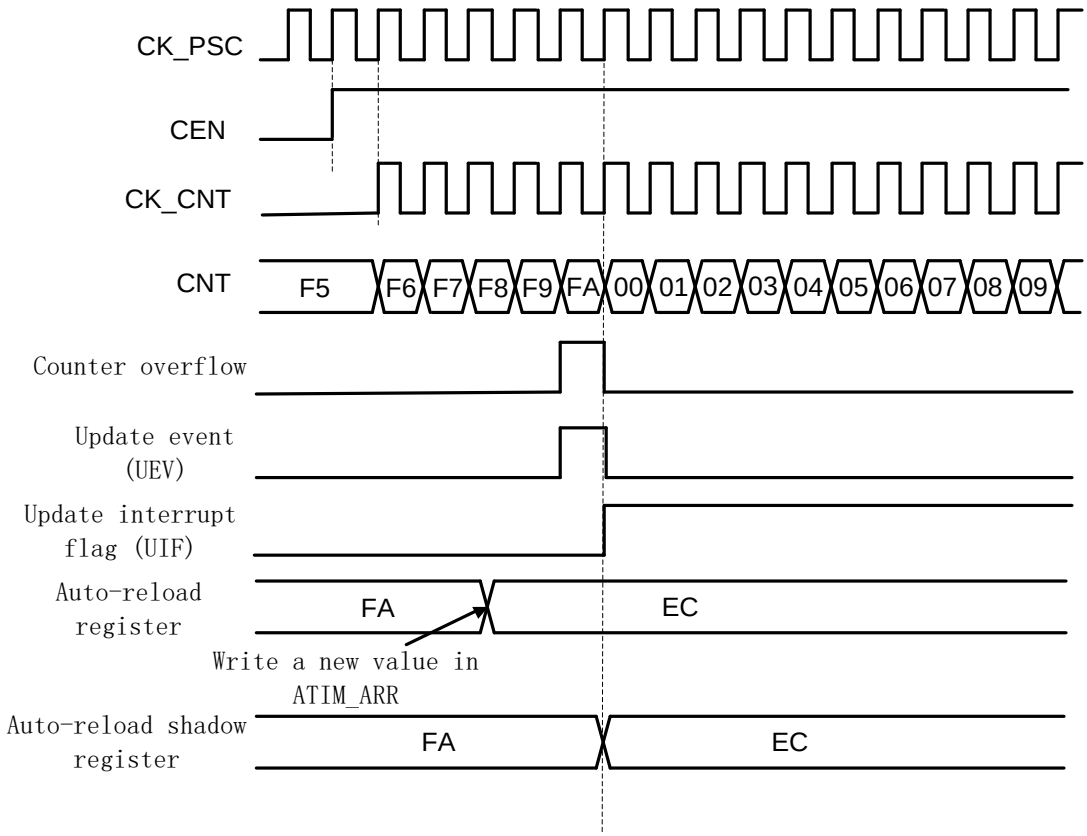


Figure 29-7 Updata event when ARPE=1 (ARR preload)

29.4.3 Counter Working Clock

BSTIM uses internal clock to work, CEN, UG and other register bits are all controlled by software

After the software operates the UG register, after the update signal is synchronized by CLK_PSC, the counter value will be reinitialized.

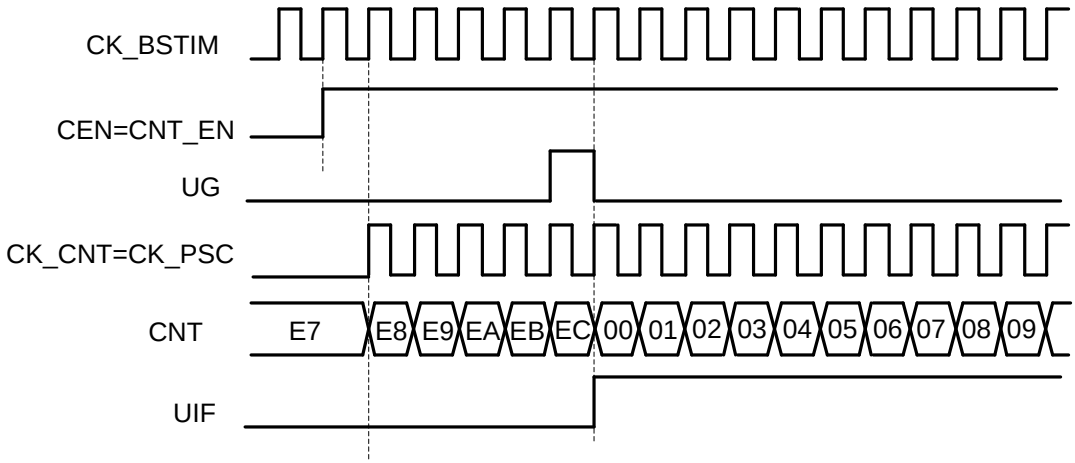


Figure 29-8 Internal clock source mode, clock division factor is 1

29.4.4 Debug mode

When Cortex-M0 enters the debug mode, the timer can stop or continue to work, and its behavior is defined by the DBG_TIMx_STOP register of the DCU module.

29.5 Register

Offset	Name	Symbol
BSTIM32(Base Address:0x4001B400)		
0x00000000	BSTIM Control Register1	BSTIM_CR1
0x00000004	BSTIM Control Register2	BSTIM_CR2
0x0000000C	BSTIM Interrupt Enable Register	BSTIM_IER
0x00000010	BSTIM Interrupt Status Register	BSTIM_ISR
0x00000014	BSTIM Event Generation Register	BSTIM_EGR
0x00000024	BSTIM Counter Register	BSTIM_CNT
0x00000028	BSTIM Prescaler Register	BSTIM_PSC
0x0000002C	BSTIM Auto-Reload Register	BSTIM_ARR

29.5.1 BSTIM Control Register1 (BSTIM_CR1)

NAME	BSTIM_CR1							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ARPE	-			OPM	URS	UDIS	CEN
access	R/W-0	U-0			R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:8	-	**RFU, Reserved, read as 0**
7	ARPE	Auto-Reload Preload Enable 0: ARR register does not enable preload 1: ARR register enables preload
6:4	-	**RFU, Reserved, read as 0**
3	OPM	One Pulse Mode 0: The counter does not stop when the Update Event occurs 1: The counter stops when the Update Event occurs (automatically clears CEN)
2	URS	Update Request Select 0: The following events can generate an update interrupt

bit	name	functional description
		-Counter overflow -Software set bitUG register 1: Only the counter overflow will generate an update interrupt
1	UDIS	Update Disable 0: enable the update event; update events are generated when the following events occur -Counter overflow -Software set bitUG register 1: The update event is forbidden, and the shadow register is not updated. Reinitialize the counter and prescaler when the UG bit is set.
0	CEN	Counter Enable 0: The counter is off 1: Counter enable

29.5.2 BSTIM Control Register 2 (BSTIM_CR2)

NAME	BSTIM_CR2							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	MMS			-			
access	U-0	R/W-000			U-0			

bit	name	functional description
31:7	-	**RFU, Reserved, read as 0**
6:4	MMS	Master mode selection, used to configure the source of the synchronous trigger signal (TRGO) sent to the slave in the master mode (Master Mode Select) 000: UG register of BSTIM_EGR is used as TRGO 001: Counter enable signal CNT_EN is used as TRGO, which can be used to start multiple timers at the same time 010: UE (update event) signal is used as TRGO 011/100/111: RFU

bit	name	functional description
		Note: The slave timer or ADC must enable the working clock in advance to receive the TRGO sent by the master timer
3:0	-	**RFU, Reserved, read as 0**

29.5.3 BSTIM Interrupt Enable Register (BSTIM_IER)

NAME	BSTIM_IER							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							UIE
access	U-0							R/W-0

bit	name	functional description
31:1	**-**	RFU：Reserved, read as 0
0	**UIE**	Update event Interrupt Enable 0: disable update event interrupt 1: Allow update event interrupts

29.5.4 BSTIM Interrupt Flag Register (BSTIM_ISR)

	BSTIM_ISR							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							UIF
access	U-0							R/W-0

bit	name	functional description
31:1	**-**	**RFU：Reserved, read as 0**
0	**UIF**	Update event Interrupt Flag,write 1 to flag When the following events occur, the UIF bit is set and the shadow register is updated -UDIS=0, the counter overflows -In the case of URS=0 and UDIS=0, the software sets the bitUG register to initialize the counter -In the case of URS=0 and UDIS=0, the trigger event initializes the counter

29.5.5 BSTIM Event Generation Register (BSTIM_EGR)

NAME	BSTIM_EGR							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

name	-	UG
access	U-0	W-0

bit	name	functional description
31:1	-	**RFU：Reserved, read as 0**
0	**UG**	Software Update event, software sets this register to generate Update event, hardware automatically clears (User Generate) When the software sets bitUG, it will reinitialize the counter and update the shadow register, and the prescaler counter will be cleared.

29.5.6 BSTIM Counter Register (BSTIM_CNT)

NAME	BSTIM_CNT							
Offset	0x00000024							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	CNT[31:24]							
access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	CNT[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CNT[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CNT[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:0	**CNT**	Counter

29.5.7 BSTIM Prescaler Register (BSTIM_PSC)

NAME	BSTIM_PSC							
Offset	0x00000028							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	PSC[31:24]							
access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	PSC[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

name	PSC[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	PSC[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:0	**PSC**	Counter Clock Prescaler $f_{CK\_CNT}=f_{CK\_PSC}/(PSC[31:0]+1)$ This is a preload register, and its content is loaded into the shadow register when the update event occurs

29.5.8 BSTIM Auto-reload Register (BSTIM_ARR)

NAME	BSTIM_ARR							
Offset	0x0000002C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	ARR[31:24]							
access	R/W-1111 1111							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	ARR[23:16]							
access	R/W-1111 1111							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	ARR[15:8]							
access	R/W-1111 1111							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ARR[7:0]							
access	R/W-1111 1111							

bit	name	functional description
31:0	**ARR**	Auto-Reload Register This is a preload register, and its content is loaded into the shadow register when the update event occurs

30 Low Power Timer (LPTIM32)

30.1 Introduction

LPTIM32 is a 32bits low power timer/counter module. By selecting the appropriate operating clock, LPTIM32 keeps running in various low-power modes and consumes very low power. LPTIM32 can even operate without an internal clock. Therefore, the function of external pulse counting in sleep mode can be realized. In addition, LPTIM32 in combination with an external input trigger signal, a low-power timeout wake-up function can be implemented

The main features of LPTIM32：

- 1 independent 32-bit upcounter
- 3bit asynchronous clock prescaler with 8 dividing factors(1、2、4、8、16、32、64、128)
- Optional operating clock:
  - Internal clock source: LSCLK, LPOSC, APBCLK, RCMF, ADC conversion end signal
  - External clock source: LPT_ETR(With analog filtering)
- Two-channel 32bit capture/compare register
- 32bit Auto Reload Register
- Input polarity selection
- Clockless external pulse counting
- Externally triggered wakeup from sleep timeout
- 32bit PWM output
- 32bit input signal capture, support DMA

30.2 Block Diagram

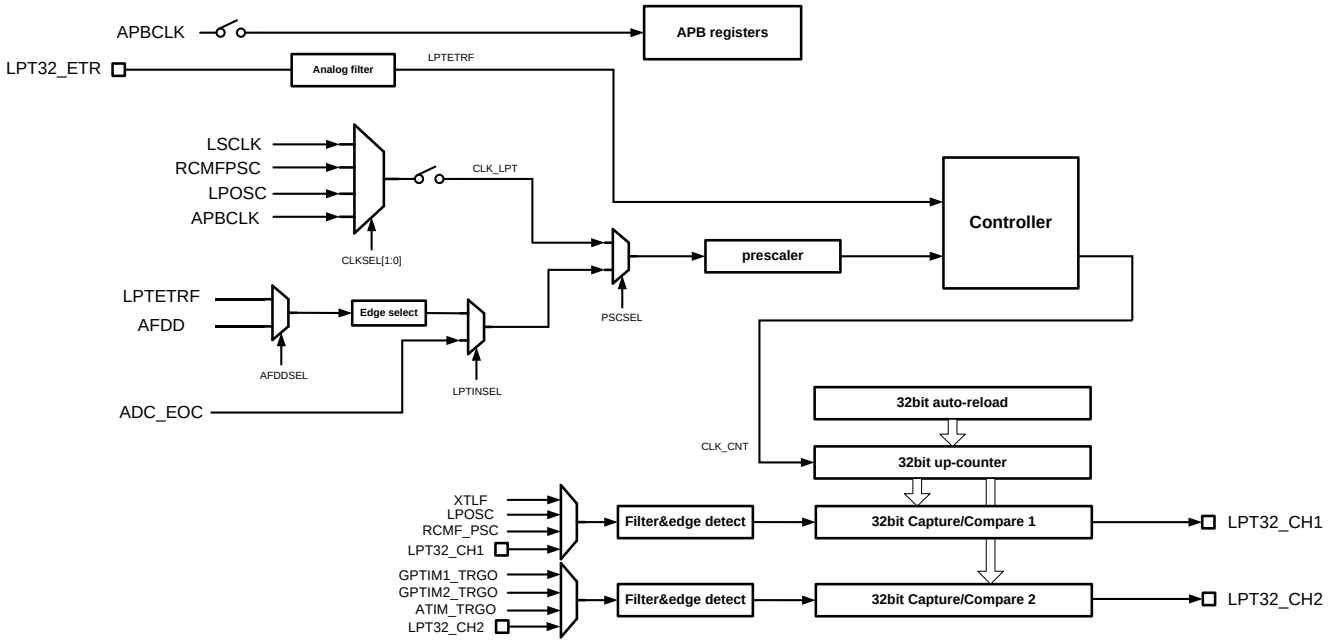


Figure 30-1 LPTIM32 Block Diagram

30.3 Timer Function

LPTIM32 supports 4 timer operating modes: general timer, external pulse triggered counting, external asynchronous pulse counting, and Timeout mode.

30.3.1 General Timer

When LPTIM_CFGR.TMODE=00, LPTIM32 is general timer operation mode.

- CLK_LPT clock counting after using multiplexed selection
- Configure the OPCCR2.LPTCKS register of CMU module to select the appropriate count clock
- There is a synchronization process of two count clocks after the LPTEN enable is set
- When enabled, the timer starts counting up until the count value is equal to LPTIM_ARR.

Single count and continuous count

LPTIM32 has two counting mode—single count and continuous count

Single count mode: When the counter is triggered, it counts to the target value (ARR) and then returns to 0 and stops automatically, generating an overflow interrupt while the hardware automatically clears the LPTEN.

Continuous count mode: The counter starts and stays running until it is turned off. The counter

reaches the target value (LPTARR) and then returns to 0 to restart counting and generates an overflow interrupt.

30.3.2 External Pulse Trigger Counting

In external pulse trigger counting mode (LPTIM_CFGR.TMOD=01), LPTIM32 uses the signal input from LPT32_ETR pin as the trigger signal. LPT32_ETR signal is first sampled and synchronized by LPTIM32 operating clock, and then can trigger the timer increment on its rising edge, falling edge or rising falling edge. Since it is necessary to use CLK_LPT to sample and identify the changing edge of the LPT32_ETR signal, it is required here that the effective level width of the ETR input signal must be greater than two times the CLK_LPT period. The software can set which edge of LPT32_ETR is counted by LPTIM32 through LPTIM_CFGR.TRIGCFG register.

The following figure shows an example of LPT32_ETR triggered counting on rising edge.

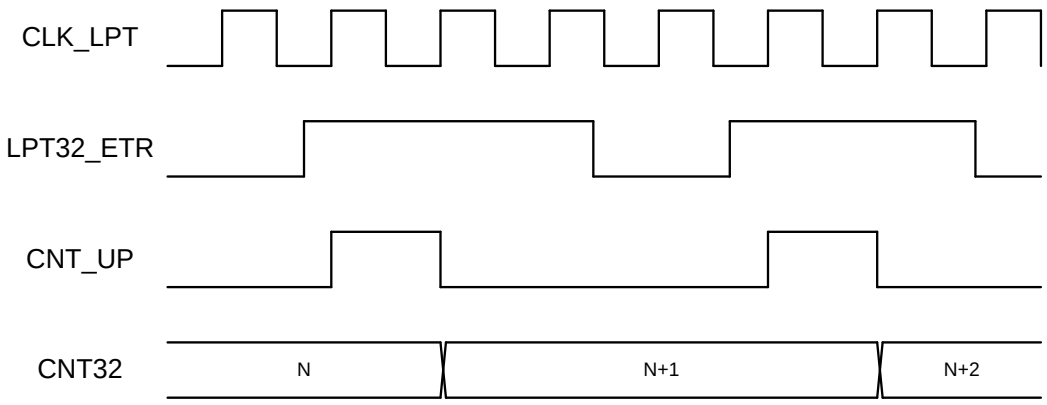


Figure 30-2 External ETR pulse rising edge triggers counting

30.3.3 External Asynchronous Pulse Counting

In external asynchronous pulse counting mode (LPTIM_CFGR.TMOD=10), the LPTIM32 uses the signal input from the LPT32_ETR pin directly as the counting clock (Need to configure PSCSEL=1 and LPTINSEL=0). In this case, the LPTIM32 works fully asynchronously and does not need to enable any internal clock. The software can select whether the timer uses ETR rising or falling edge counting via LPTIM_CFGR.EDGESEL. Since any disturbing signal on the LPT32_ETR pin in this mode may cause the timer to malfunction, it is recommended to enable the ETR input analog filtering function, which is able to filter out glitch signals within about 100ns.

The following figure shows an example of external asynchronous pulse counting on falling edge.

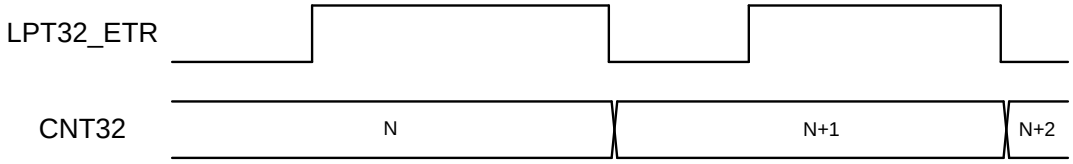


Figure 30-3 External ETR pulse asynchronous counting (falling edge)

30.3.4 Timeout Mode

In Timeout mode (LPTIME_CFGR.TMODE=11), the LPTIM32 uses the signal input from the LPT32_ETR pin as the trigger signal and the timer works with the internal clock CLK_LPT. After the timer starts in Timeout mode, it does not start counting immediately, but waits for the first valid edge of the LPT32_ETR signal to arrive. When the first valid edge arrives, the timer is triggered to start free counting, and thereafter each new valid edge of ETR clears the counter and starts counting again. According to the actual frequency of the external input ETR signal, the counter operating clock and overflow limit (ARR) shall be reasonably configured to keep the timer from overflowing. If the timer overflows, it means no expected ETR event arrives within the specified time interval, then the timer generates an overflow interrupt, the count value returns to 0, and the LPTIM_CR.EN is automatically cleared to end the counting process.

After the LPT32_ETR signal is sampled and synchronized by the LPTIM32 operating clock, it can trigger the counter to clear and restart at its rising edge, falling edge or both edges. Since it is necessary to sample and identify the changing edge of the LPT32_ETR signal using CLK_LPT, it is required that the effective level width of the ETR input signal must be greater than two times the CLK_LPT period. The software can set which edge of LPT32_ETR is counted by LPTIM32 through LPTIM_CFGR.TRIGCFG register.

The following figure is an example of using LPT32_ETR rising edge clearing in timeout mode and eventually overflowing.

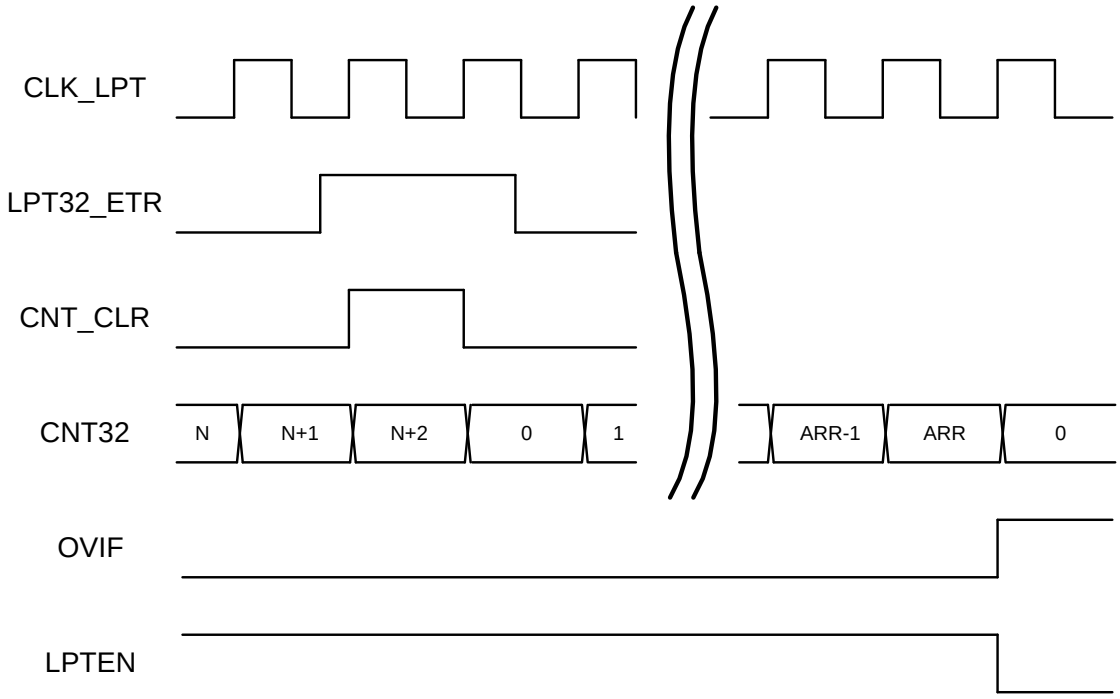


Figure 30-4 Timeout mode

Using Time Out mode and enabling the LPTIM interrupt, the time-out wake-up function triggered by external signals can be realized when the chip is in sleep mode. At this time, as long as there is a periodic signal input on the LPT32_ETR pin, it can keep the chip in sleep. If no trigger signal arrives within the specified time, the LPTIM timeout overflow interrupt will wake up the chip.

30.4 Capture/Compare Function

LPTIM32 comes with two independent 32bit capture/compare channels, with 32bit timer as time base, combined with CCRx register.LPTIM32 supports two channels of 32bit PWM output, or 32bit input capture function.

30.4.1 32bit PWM

LPTIM32 comes with two independent 32bit capture/compare channels, with 32bit timer as time base, combined with CCRx register.LPTIM32 supports two channels of 32bit PWM output, or 32bit input capture function.

After PWM is enabled, the LPTIM32 counts from 0x0000_0000, the output goes high when the count value equals the comparison value (CCRx) and goes low when the count value equals the target value register (LPTARR); the PWM period is determined by the ARR register and the duty cycle is determined by the CCRx register. The polarity of the waveform can be configured.

To implement the PWM output function, LPTCFG.CCxS needs to be configured to 10, at which point LPT_CHx becomes the output channel and the corresponding GPIO automatically enables

the output function (the software needs to configure the GPIO as a digital peripheral function).

The figure below shows an example of PWM output with POLARITY=1.

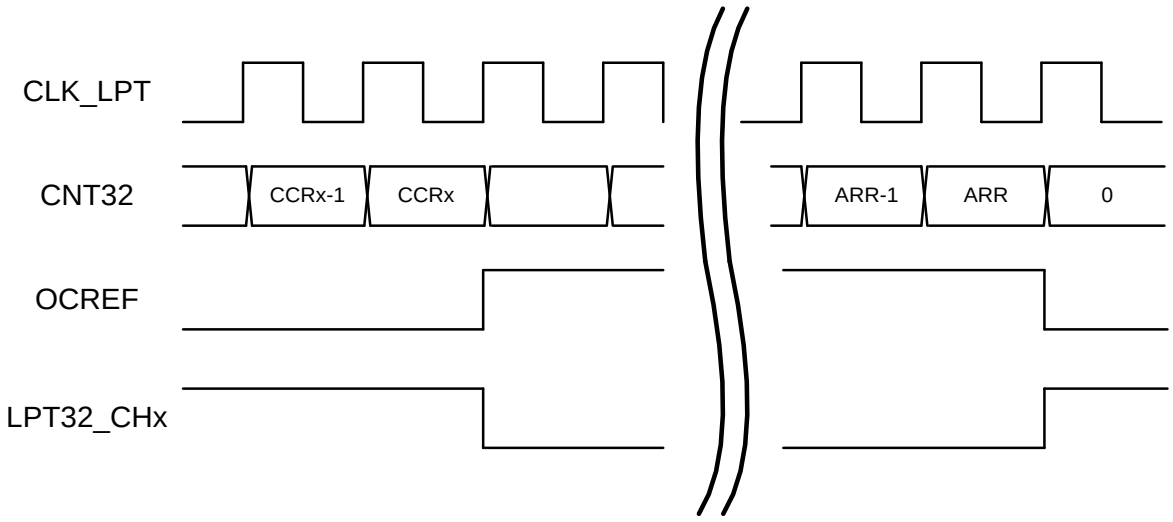


Figure 30-5 PWM output

30.4.2 Input Capture

The LPTIM32's two capture/compare channels enable two independent input signal period or level width capture functions. The input signal capture function can be used in conjunction with DMA to enable automatic handling of multiple consecutive capture results.

The input capture can be configured to capture on the rising edge, falling edge, or both edge of the input signal. Each time a capture occurs, the CAPxEDGE register indicates whether the current capture is a rising or falling edge.

Channel 1 of LPTIM32 can capture the external pin input or chip internal clock signal (XTLF, LPOSC, RCMF_PSC). The period capture of internal clock signal can be used for clock frequency calibration with software; while channels 2 can only capture the external pin input signal.

After enabling the input mode, the 32bit counter is free to count as a time base. When a valid edge of the captured signal arrives, the current count value is latched into the CCRx register and a capture interrupt is generated; if the DMA function is enabled, CCRx will also be read by DMA and written to the specified address in RAM at the same time. The capture interrupt flag is automatically cleared by hardware when the CCRx register is read by softwareor DMA, the capture interrupt flag can also be cleared by software by writing 1 in addition. When the capture interrupt flag is not cleared and a new capture event arrives, the capture conflict interrupt flag (CAPxOVR) will be set.

The following figure shows an example of capturing the rising and falling edges of the input signal.

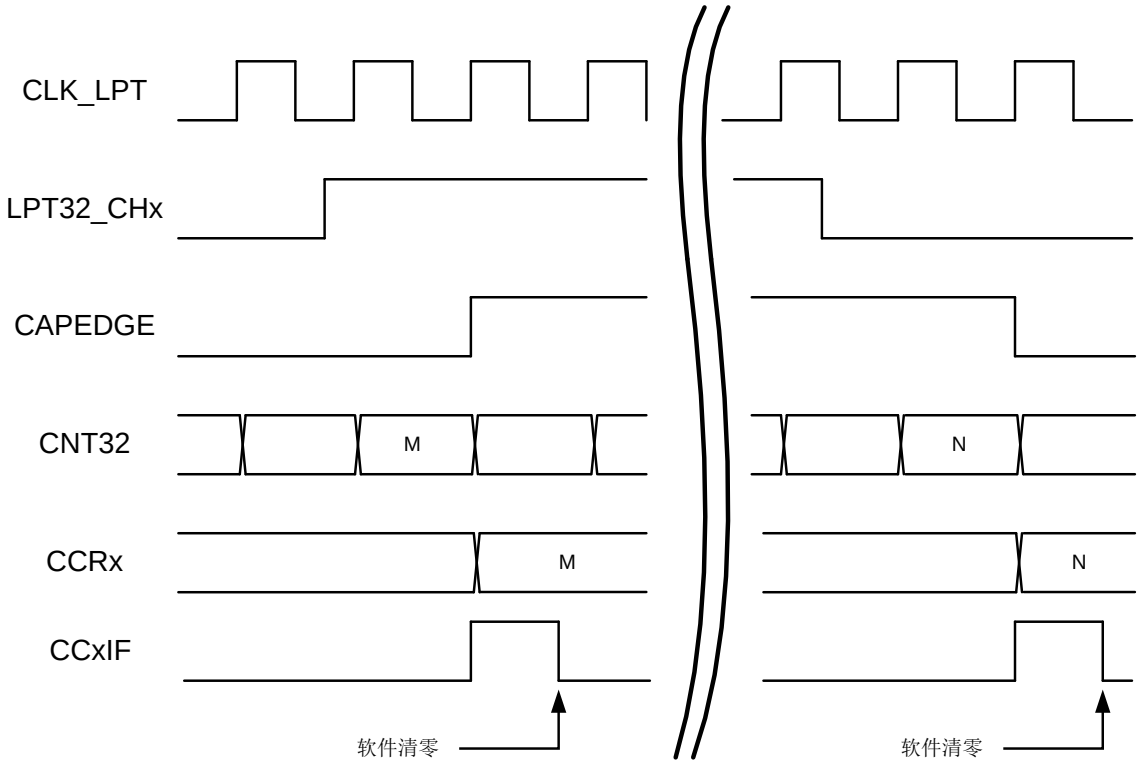


Figure 30-6 PWM output

30.5 Register

Offset	Name	Symbol
LPTIM(Base Address:0x40013400)		
0x00000000	LPTIM Config Register	LPTIM_CFGR
0x00000004	LPTIM Counter Register	LPTIM_CNT
0x00000008	LPTIM Capture/Compare Control and Status Register	LPTIM_CCSR
0x0000000C	LPTIM Auto-Reload Register	LPTIM_ARR
0x00000010	LPTIM Interrupt Enable Register	LPTIM_IER
0x00000014	LPTIM Interrupt Status Register	LPTIM_ISR
0x00000018	LPTIM Control Register	LPTIM_CR
0x00000020	LPTIM Capture/Compare Register1	LPTIM_CCR1
0x00000024	LPTIM Capture/Compare Register2	LPTIM_CCR2

30.5.1 LPTIM Config Register (LPTIM_CFGR)

NAME	LPTIM_CFGR							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							ETR_AFEN
access	U-0							R/W-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-	PSCSEL	LPTINSEL	DIVSEL			-	
access	U-0	R/W-0	R/W-0	R/W-000			U-0	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	EDGESEL	TRIGCFG		-		ONST	TMOD	
access	R/W-0	R/W-00		U-0		R/W-0	R/W-00	

bit	name	functional description
31:25	-	**RFU: Reserved, read as 0**
24	**ETR_AFEN**	External Trigger Input Analog Filter Enable 0:Disable analog filtering 1:Enable analog filtering, the filter width is about 100ns
23:15	-	**RFU: Reserved, read as 0**

bit	name	functional description
14	**PSCSEL**	Prescaler Input Select 0:CLKSELselected clock 1:LPTINSELselected signal
13	**LPTINSEL**	External Trigger Input Source Select 0: Pin input 1:ADC_EOC
12:10	**DIVSEL**	Counter Clock Divider Select 000:divided-by-1 001: divided-by-2 010: divided-by-4 011: divided-by-8 100: divided-by-16 101: divided-by-32 110: divided-by-64 111: divided-by-128
9:8	**-**	**RFU: Reserved, read as 0**
7	**EDGESEL**	ETR Clock Edge Select 0:Rising edge count of LPT_ETR 1:Falling edge count of LPT_ETR
6:5	**TRIGCFG**	External Trigger Edge Selection (Need to use internal clock to synchronize sampling LPT_ETR) 00:LPT_ETR input signal rising edge trigger 01:LPT_ETR input signal falling edge trigger 10/11:External input signal rising and falling edge trigger
4:3	**-**	**RFU: Reserved, read as 0**
2	**ONST**	One State Timer Enable 0: Continuous counting mode: the counter is triggered and remains running until it is turned off. The counter reaches the target value and returns to 0 to restart counting and generates an overflow interrupt. 1:Single count mode: the counter is triggered and counts back to 0 after reaching the target value and stops automatically, generating an overflow interrupt.
1:0	**TMOD**	Timer operation Mode 00:General Timer Mode 01: External pulse trigger counting mode 10:External asynchronous pulse counting mode 11:Timeoutmode

30.5.2 LPTIM Counter Register (LPTIM_CNT)

NAME	LPTIM_CNT							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	CNT32[31:24]							
access	R-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	CNT32[23:16]							
access	R-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CNT32[15:8]							
access	R-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CNT32[7:0]							
access	R-0000 0000							

bit	name	functional description
31:0	**CNT32**	Counter 32bits-wide,read only

30.5.3 LPTIM Capture/Compare Control and Status Register (LPTIM_CCSR)

NAME	LPTIM_CCSR							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-				CAP2SSEL		CAP1SSEL	
access	U-0				R/W-00		R/W-00	
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-		CAP2EDGE	CAP1EDGE	CAPCFG2		CAPCFG1	
access	U-0		R-0	R-0	R/W-00		R/W-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-		POLAR2	POLAR1	CC2S		CC1S	
access	U-0		R/W-0	R/W-0	R/W-00		R/W-00	

bit	name	functional description
31:20	**-**	**RFU: Reserved, read as 0**

bit	name	functional description
19:18	**CAP2SSEL**	Channel 2 Capture Source Select 00:GPTIM0_TRGO 01: GPTIM1_TRGO 10:ATIM_TRGO 11:LPT32_CH2input
17:16	**CAP1SSEL**	Channel 1 Capture Source Select 00:LPT32_CH1input 01:XTLF 10:LPOSC 11:RCMF_PSC
15:14	**-**	RFU: **Reserved, read as 0**
13	**CAP2EDGE**	Channel2 Captured Edge, update when CC2IF is set 0:Falling edge 1: Rising edge
12	**CAP1EDGE**	Channel 1 Captured Edge, update when CC1IF is set 0:Falling edge 1: Rising edge
11:10	**CAP2CFG**	Channel 2 Capture Edge Config 00:Rising edge capture 01:Falling edge capture 10:Rising and falling edge capture 11:RFU
9:8	**CAP1CFG**	Channel 1 Capture Edge Config 00:Rising edge capture 01:Falling edge capture 10:Rising and falling edge capture 11:RFU
7:6	**-**	**RFU: Reserved, read as 0**
5	**POLAR2**	Channel 2 Compare Output Polarity 0:Positive waveform, starts at low, set high when the count value == compare value, restored to low when the count value == ARR 1:Negative polarity waveform, positive polarity waveform is reversed
4	**POLAR1**	Channel 1 Compare Output Polarity 0: Positive waveform, starts at low, set high when the count value == compare value, restored to low when the count value == ARR 1: Negative polarity waveform, positive polarity waveform is reversed
3:2	**CC2S**	Channel 2 Capture/Compare Select 00,11:Disable channel 2 capture/compare function 01:Enable channel 2 capture function(LPT32_CH2is input) 10:Enable channel 2 comparison function (LPT32_CH2is output)

bit	name	functional description
1:0	**CC1S**	Channel 1 Capture/Compare Select 00,11: Disable channel 1 capture/compare function 01:Enable channel 2 capture function(LPT32_CH1is input) 10:Enable channel 2 comparison function (LPT32_CH1is output)

30.5.4 LPTIM Auto-Reload Register (LPTIM_ARR)

NAME	LPTIM_ARR							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	ARR[31:24]							
access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	ARR[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	ARR[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ARR[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:0	**ARR**	Auto-Reload Register When the counter count is equal to ARR, the counter returns to its initial value and starts counting up again

30.5.5 LPTIM Interrupt Enable Register (LPTIM_IER)

NAME	LPTIM_IER							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						OVR2IE	OVR1IE
access	U-0						R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

name	-	TRIGIE	OVIE	CC2IE	CC1IE
access	U-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:10	-	**RFU: Reserved, read as 0**
9	**OVR2IE**	Channel 2 Over-Capture Interrupt Enable 1:Enable Interrupt 0:Disable Interrupt
8	**OVR1IE**	Channel 1 Over-Capture Interrupt Enable 1:Enable Interrupt 0:Disable Interrupt
7:4	-	**RFU: Reserved, read as 0**
3	**TRIGIE**	External Trigger Interrupt Enable 1:External trigger arrival interrupt enable 0:External trigger arrival interrupt disable
2	**OVIE**	Counter Over-Flow Interrupt Enable 1:Counter overflow interrupt enable 0:Counter overflow interrupt disable
1	**CC2IE**	Capture/Compare Channel 2 Interrupt Enable 1:Capture/compare channel 2 interrupt enable 0:Capture/compare channel 2 interrupt disable
0	**CC1IE**	Capture/Compare Channel 1 Interrupt Enable 1:Capture/compare channel 1 interrupt enable 0:Capture/compare channel 1 interrupt disable

30.5.6 LPTIM Interrupt Status Register (LPTIM_ISR)

NAME	LPTIM_ISR							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-						CAP2OVR	CAP1OVR
access	U-0						R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				TRIGIF	OVIF	CC2IF	CC1IF
access	U-0				R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:10	-	**RFU: Reserved, read as 0**
9	**CAP2OVR**	Channel 2 Over-Capture Interrupt Flag, hardware set, write 1 to clear by software 1:In input capture mode, a new capture occurs when CC2IF is 1, and overrun occurs 0: No overrun occurred
8	**CAP1OVR**	Channel 1 Over-Capture Interrupt Flag, hardware set, write 1 to clear by software 1:In input capture mode, a new capture occurs when CC1IF is 1, and overrun occurs 0:No overrun occurred
7:4	-	**RFU: Reserved, read as 0**
3	**TRIGIF**	External Trigger Interrupt Flag,write 1 to clear 1:External trigger arrival interrupt generation 0:No interrupt generation
2	**OVIF**	Counter Over-Flow Interrupt Flag,write 1 to clear 1:Counter overflow interrupt generated 0:No interrupt generation
1	**CC2IF**	Capture/Compare Channel 2 Interrupt Flag, hardware set, write 1 to clear by software 1:Counter value and comparison value 2 match, or a capture event occurs 0:No interrupt generation
0	**CC1IF**	Capture/Compare Channel 1 Interrupt Flag, hardware set, write 1 to clear by software 1:Counter value and comparison value 1 match, or a capture event occurs 0:No interrupt generation

30.5.7 LPTIM Control Register (LPTIM_CR)

NAME	LPTIM_CR							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							

bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							EN
access	U-0							R/W-0

bit	name	functional description
31:1	-	RFU: **Reserved, read as 0**
0	**EN**	LPTIM Enable 1:Enable counter 0:Disable counter

30.5.8 LPTIM Capture/Compare Register1 (LPTIM_CCR1)

NAME	LPTIM_CCR1							
Offset	0x00000020							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	CCR1[31:24]							
access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	CCR1[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CCR1[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CCR1[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:0	**CCR1**	Channel1 Capture/Compare Register

30.5.9 LPTIM Capture/Compare Register2 (LPTIM_CCR2)

NAME	LPTIM_CCR2							
Offset	0x00000024							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	CCR2[31:24]							
access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	CCR2[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	CCR2[15:8]							

access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	CCR2[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:0	**CCR2**	Channel2 Capture/Compare Register

31 Real-time clock (RTC)

31.1 Introduction

The Real-time Clock (RTC) module maintains accurate timing for long periods of time, consumes very low power, and works in all power consumption modes.

The main features are as follows：

- BCD time format, complete perpetual calendar (00~99 years)
- Periodic wake-up interrupt
- Alarm interrupt
- Configurable periodic timing signal output
- Digital adjustment, accuracy +/-0.477ppm
- Feedback resistor integration
- RTC timing section registers are not reset

31.2 Block diagram

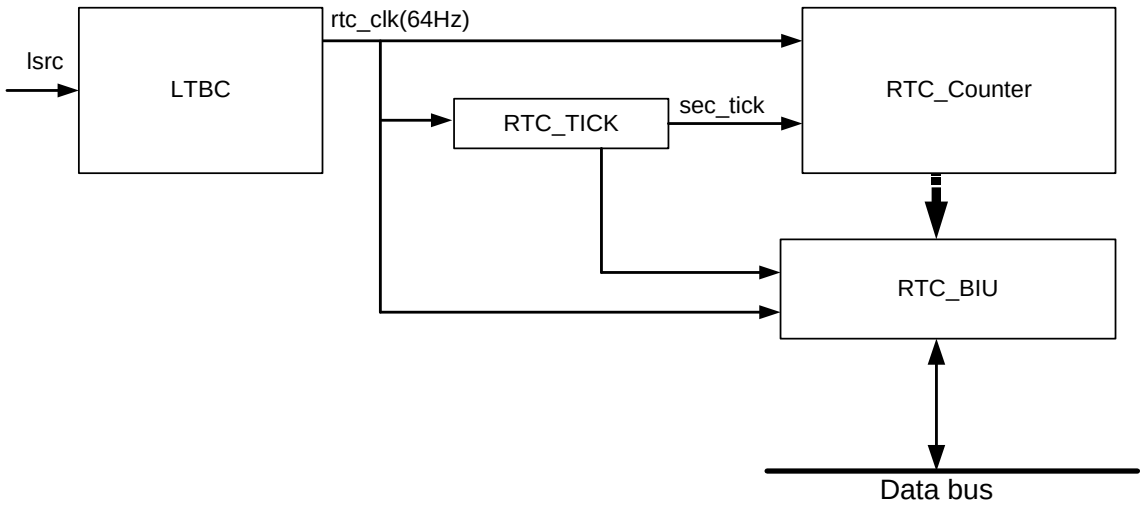


Figure 31-1 RTC block diagram

The LTBC module is a low power time base counter module used to generate the low speed operating clock required by the system, see LTBC function introduction for specific description.

The RTC_TICK module is mainly used to generate the second pulse signal sec_tick that jumps every second and the 2Hz,4Hz,8Hz,16Hz and other signals needed to generate interrupts. The sec_tick signal is output to the RTC_Counter module to realize the counter synchronization of the perpetual calendar.

The RTC_Counter module is the perpetual calendar implementation module of RTC, including a second counter, a minute counter, an hour counter, a day counter, a week counter, a month counter, and a year counter.The module enables automatic recognition of leap years.

31.3 Working principle

The RTC is not reset after power-up, so it is required to set the current time by software before normal operation. The time clock uses 32.768KHz crystal oscillator. Since the crystal oscillator may fail, in order to ensure reliability, the fail detection circuit is enabled to continuously detect the 32.768KHz oscillator output and generate an alarm interrupt if it is found to fail. Meanwhile, the software can configure whether to automatically switch the RTC clock to LPOSC when XTLF stops vibration. If this function is enabled, the RTC time will have a certain error, but it will not stop; if the automatic switching is not enabled, it can also be handled by the software after responding to the vibration stop interruption.

31.3.1 Low-power time base counter (LTBC)

The low power time base counter (LTBC) module is used to generate the low speed operating clock required by the system. Its functions include：

- Get 64Hz RTC and IWDT working clock by prescaling LSCLK
- Digital calibration of the RTC clock can be achieved by adjusting the count period, the minimum step size can be achieved by adjusting once every 32s is 0.952ppm, and the theoretical accuracy after adjustment is +/-0.477ppm
- Precise second time mark can be obtained by using PLL virtual calibration
- Generate 1KHz, 256Hz, 64Hz, 16Hz, 4Hz, 1Hz periodic interrupts, of which 1K and 256Hz are uncalibrated and the others are digitally calibrated (if digital calibration is enabled)
- The 64Hz prescaler circuit is not affected by chip reset
- 1/256s precision timing

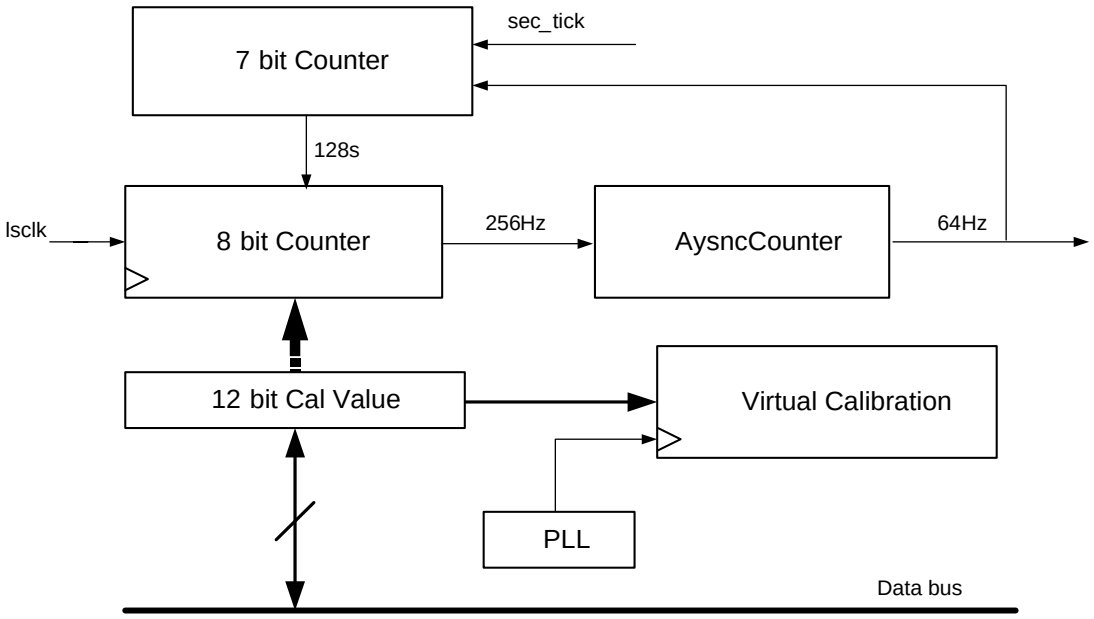


Figure 31-2 LTBC block diagram

31.3.2 LTBC digital calibration

LTBC is mainly composed of a synchronous prescaler counter, an asynchronous divider counter, a clock calibration value register, a virtual calibration circuit and a control register.

The purpose of digital calibration is to enable the RTC to obtain averagely accurate timing over a longer period of time.Since the clock source of the RTC is 32768Hz, the minimum step of digital calibration is 30.5us, and if it is adjusted once in 1 second, the maximum accuracy can only reach 30.517ppm. To get higher accuracy, the adjustment must be made in longer period. FM33LE0xxA takes 32s as a calibration cycle, and each cycle can be adjusted from 0 to +/-511 32768Hz clock cycles, so the highest accuracy is 30.5us/32s=0.952ppm, the maximum adjustment range is +/-(511*30.517us/32s)=+/-487ppm, and the average minimum clock error after adjustment is +/-0.476ppm.

The calibration value consists of 10bit registers, where the highest bit is the sign bit, indicating the increase or decrease of the count value, and the remaining 9bit indicates the absolute value of the increase or decrease. In order to improve the average accuracy per second and avoid large second-to-second jitter, the calibration value is distributed equally within each second, which is implemented as follows.

In addition to the highest sign bit, the remaining 9 bits can be divided into a high 4-bit public value and a 5-bit private value, where the public value indicates the value to be adjusted every second within 32s, and the private value indicates the need to add or subtract 1 in some seconds within 32s.

Bit9	Bit[8:5]	Bit[4:0]
Sign	Common Value (C)	Differential Value (D)

The calibration value formula can be expressed as follows: Correction(ppm) = (C*32 + D)*30.517/32000000

Assuming that the clock increase by 0.953ppm, which is equivalent to only 30.5us in 32s, the calibration value is written as 0_0000_00001, so the public value is 0 and the private value is 1, only need to add 1 to a second period within 32s. And assuming that the clock increase by 487ppm, which is equivalent to 511 30.5us in 32s, the calibration value is written as 0_1111_11111111, the public value is 15, the private value is 31, which means that 15 is added every second in 32s, and there are 31s which0 need to add 1 extra.

Example of calibration value.

ppm	ADJUST[1]	Common	Differential	Expression
0.953	0_0000_00001	0	1	1*30.517/32000000
-125.88	1_0100_00100	4	4	(4*32+4)*30.517/32000000
32.42	0_0001_00010	1	2	(1*32+2)*30.517/32000000
487.32	0_1111_11111	15	31	(15*32+31)*30.517/32000000

Note:

[1] ADJUST: Clock Error Adjustment Register

To avoid timing conflicts, the software should update ADJUST and start clock calibrationafter the second interrupt.

Take ADJUST=0_0001_00000 as an example, add a 32768Hz period at the end of each second.

31.3.3 BCD clock

- **Second count**

The second counter only needs 7 bits, counting from 0 to 59, where bit[3:0] is 1 second unit, counting range 0-9; bit[6:4] is 10 seconds unit, counting range 0-5. When the count is full 60s, the second incoming signal is triggered to add 1 to the minute counter.

Bit6-4	Bit3-0
0-5	0-9

- **Minute count**

The minute counting also requires only 7 bits, and the counting range is the same as seconds, so the implementation method is same.

Bit6-4	Bit3-0
0-5	0-9

- **Hour count**

Hour count range from 0 to 24 with only 6 bits.

Bit5-4	Bit3-0
0-2	0-9

- **Day count**

The day count range is 1-31, only 6 bits, starting from 1, counting up to 28/29/30/31 according to the month and leap year, and triggering the day-in signal to add 1 to the month counter when the count is full.

Bit5-4	Bit3-0
0-3	0-9

- **Week count**

The week count range is 0-6 with only 3 bits and cycle count from 0 to 6.

Bit2-0
0-6

- **Month count**

The month count range is 1-12, only 5bit, counting from 1 to 12, and triggering the month-in signal to add 1 to the year counter when the count is full.

Bit4	Bit3-0
0-1	0-9

- **Year count**

The annual counting range is 0-99, which takes 8bit and cycle from 0 to 99.

Bit7-4	Bit3-0
0-9	0-9

31.3.4 RTC enable and disable

RTC works automatically after power on. The software should set the current time after the 32.768K crystal oscillator is fully activated; the chip uses internal ring oscillator before the crystal oscillator is stable, the deviation is large.

If the software disables XTLF and does not switch LSCLK to LPOSC, the RTC stops timing.

31.3.5 RTC time setting

Software can set the RTC BCD registers directly at any time, usually it is recommended to set the time after XTLF is fully started; since the operation of setting the time register is asynchronous with the RTC timing, it is recommended that the software set the time after the second interrupt event and read out the time value for verification after the time setting.

Note that the hardware does not check the time validity, the software must ensure that the written BCD time is correct.

The FM33LC0xx also supports ms-level timing, i.e. the time can be set to 3.9ms level accuracy (1/256s). In addition, when the software writes the second time, the hardware automatically clears the 64Hz->1Hz intersecond counter to achieve second alignment.

In order to improve the anti-interference capability, FM33LC0xx provides time write protection function. 0xACACACAC must be written to the write protection register before the time register can be rewritten. After the time setting is completed, the software can prohibit the writing of time registers by writing any other value to restore the write protection.

31.3.6 RTC time reading

Time reading mode 1:

- Read the current time register value
- Read the current time register value again
- If the contents of the 2 readings are same, it is the correct current time; if the contents of the two readings are not same, repeat the first two steps.

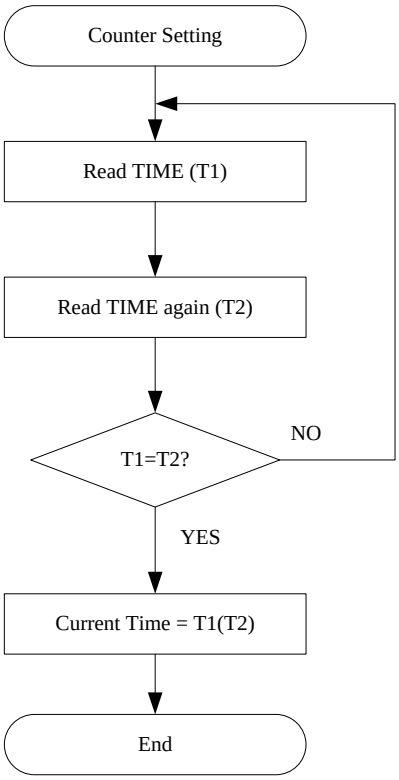


Figure 31-3 RTC time reading flow chart

Time reading mode 2:

Reading the time register immediately by software after 1s interrupt occurs, which ensures that the correct current time value is read.

31.3.7 Leap Year determination

The RTC module of FM33LC0xx automatically determines the leap year.

Leap year condition: (mod 400 == 0) or (mod 4 == 0 and mod 100 <> 0)

31.4 Register

Offset	Name	Symbol
RTC(base address:0x40011000)		
0x00000000	RTC Write Enable Register	RTC_WER
0x00000004	RTC Interrupt Enable Register	RTC_IER
0x00000008	RTC Interrupt Status Register	RTC_ISR
0x0000000C	BCD format time second registers	RTC_BCDSEC
0x00000010	BCD format time minute registers	RTC_BCDMIN
0x00000014	BCD format time hour registers	RTC_BCDHOUR
0x00000018	BCD format time day registers	RTC_BCDDAY
0x0000001C	BCD format time week registers	RTC_BCDWEEK
0x00000020	BCD format time month registers	RTC_BCDMONTH
0x00000024	BCD format time year registers	RTC_BCDYEAR
0x00000028	RTC Alarm Register	RTC_ALARM
0x0000002C	RTC Time Mark Select Register	RTC_TMSEL
0x00000030	RTC time Adjust Register	RTC_ADJUST
0x00000034	RTC time Adjust Sign Register	RTC_ADSIGN
0x0000003C	RTC Sub-Second Counter Register	RTC_SBSCNT
0x00000070	RTC Backup Registers 0	RTC_BKR0
0x00000074	RTC Backup Registers 1	RTC_BKR1
0x00000078	RTC Backup Registers 2	RTC_BKR2
0x0000007C	RTC Backup Registers 3	RTC_BKR3
0x00000080	RTC Backup Registers 4	RTC_BKR4
0x00000084	RTC Backup Registers 5	RTC_BKR5
0x00000088	RTC Backup Registers 6	RTC_BKR6
0x0000008C	RTC Backup Registers 7	RTC_BKR7

31.4.1 RTC Write Enable Register(RTC_WER)

NAME	RTC_WER							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							WE
access	U-0							R/W-0

bit	name	functional description
31:1	-	RFU: **Reserved, read as 0**

bit	name	functional description
0	**WE**	RTC Write Enable Register When the CPU writes 0xACACACAC to RTCWE, the CPU is allowed to write the initial value to the BCD time register of RTC, and then RTCWE is set to 1. When the CPU writes any value not 0xACACACAC to RTCWE, the write protection is restored, and then RTCWE is cleared to 0.

31.4.2 RTC Interrupt Enable Register(RTC_IER)

NAME	RTC_IER							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			ADJ_IE	ALARM_IE	1KHZ_IE	256HZ_IE	64HZ_IE
access	U-0			R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
bit	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
name	16HZ_IE	8HZ_IE	4HZ_IE	2HZ_IE	SEC_IE	MIN_IE	HOUR_IE	DAY_IE
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:13	-	RFU: **Reserved, read as 0**
12	ADJ_IE	Time Adjust Interrupt Enable 1:Interrupt enable 0:Interrupt disable
11	ALARM_IE	Alarm Interrupt Enable 1:Interrupt enable 0:Interrupt disable
10	1KHZ_IE	1KHz Periodic Interrupt Enable 1:Interrupt enable 0:Interrupt disable
9	256HZ_IE	256Hz Periodic Interrupt Enable 1:Interrupt enable 0:Interrupt disable
8	64HZ_IE	64Hz Periodic Interrupt Enable 1:Interrupt enable 0:Interrupt disable
7	16HZ_IE	16Hz Periodic Interrupt Enable 1:Interrupt enable 0:Interrupt disable
6	8HZ_IE	8Hz Periodic Interrupt Enable 1:Interrupt enable 0:Interrupt disable
5	4HZ_IE	4hz Periodic Interrupt Enable 1:Interrupt enable 0:Interrupt disable

bit	name	functional description
4	2HZ_IE	2Hz Periodic Interrupt Enable 1:Interrupt enable 0:Interrupt disable
3	SEC_IE	Second Interrupt Enable 1:Interrupt enable 0:Interrupt disable
2	MIN_IE	Minute Interrupt Enable 1:Interrupt enable 0:Interrupt disable
1	HOUR_IE	Hour Interrupt Enable 1:Interrupt enable 0:Interrupt disable
0	DAY_IE	Day Interrupt Enable 1:Interrupt enable 0:Interrupt disable

31.4.3 RTC InterruptStatus Register(RTC_ISR)

NAME	RTC_ISR							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			ADJ_IF	ALARM_IF	1KHZ_IF	256HZ_IF	64HZ_IF
access	U-0			R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
bit	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
name	16HZ_IF	8HZ_IF	4HZ_IF	2HZ_IF	SEC_IF	MIN_IF	HOUR_IF	DAY_IF
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

bit	name	functional description
31:13	-	RFU: **Reserved, read as 0**
12	ADJ_IF	Time Adjust Interrupt Flag, write 1 to clear 1:Interrupt set 0:No interrupt generation
11	ALARM_IF	Alarm Interrupt Flag, write 1 to clear 1:Interrupt set 0:No interrupt generation
10	1KHZ_IF	1KHz Periodic Interrupt Flag, write 1 to clear 1:Interrupt set 0:No interrupt generation
9	256HZ_IF	256Hz Periodic Interrupt Flag, write 1 to clear 1:Interrupt set 0:No interrupt generation
8	64HZ_IF	64Hz Periodic Interrupt Flag, write 1 to clear 1:Interrupt set 0:No interrupt generation

bit	name	functional description
7	16HZ_IF	16Hz Periodic Interrupt Flag, write 1 to clear 1:Interrupt set 0:No interrupt generation
6	8HZ_IF	8Hz Periodic Interrupt Flag, write 1 to clear 1:Interrupt set 0:No interrupt generation
5	4HZ_IF	4Hz Periodic Interrupt Flag, write 1 to clear 1:Interrupt set 0:No interrupt generation
4	2HZ_IF	2Hz periodic Interrupt Flag, write 1 to clear 1:Interrupt set 0:No interrupt generation
3	SEC_IF	Second Interrupt Flag, write 1 to clear 1:Interrupt set 0:No interrupt generation
2	MIN_IF	Minute Interrupt Flag, write 1 to clear 1:Interrupt set 0:No interrupt generation
1	HOUR_IF	Hour Interrupt Flag,write 1 to clear 1:Interrupt set 0:No interrupt generation
0	DAY_IF	Day Interrupt Flag,write 1 to clear 1:Interrupt set 0:No interrupt generation

31.4.4 BCD Format Time Second Registers(RTC_BCDSEC)

NAME	RTC_BCDSEC							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	SEC						
access	U-0	R/W-xxx xxxx						

bit	name	functional description
31:7	-	RFU: **Reserved, read as 0**
6:0	SEC	Binary-Coded Decimal Format Seconds Register

31.4.5 BCD Format Time MinuteRegisters(RTC_BCDMIN)

Name	RTC_BCDMIN							
Offset	0x00000010							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24

name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	MIN						
access	U-0	R/W-xxx xxxx						

bit	name	functional description
31:7	-	RFU: **Reserved, read as 0**
6:0	MIN	Binary-Coded Decimal Format Minutes Register

31.4.6 BCD Format Time HourRegisters(RTC_BCDHOUR)

NAME	RTC_BCDHOUR							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-		HOUR					
access	U-0		R/W-xx xxxx					

bit	name	functional description
31:6	-	RFU: **Reserved, read as 0**
5:0	HOUR	Binary-Coded Decimal Format Hours Register

31.4.7 BCD Format Time DayRegisters(RTC_BCDDAY)

NAME	RTC_BCDDAY							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							

access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-		DAY					
access	U-0		R/W-xx xxxx					

bit	name	functional description
31:6	-	**RFU: Reserved, read as 0**
5:0	DAY	Binary-Coded Decimal Format Date Register

31.4.8 BCD Format Time WeekRegisters(RTC_BCDWEEK)

NAME	RTC_BCDWEEK							
Offset	0x0000001C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					WEEK		
access	U-0					R/W-xxx		

bit	name	functional description
31:3	-	RFU: **Reserved, read as 0**
2:0	WEEK	Binary-Coded Decimal Format Week Register

31.4.9 BCD Format Time MonthRegisters(RTC_BCDMONTH)

Name	RTC_BCDMONTH							
Offset	0x00000020							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-			MONTH				
access	U-0			R/W-x xxxx				

bit	name	functional description
31:5	-	RFU: **Reserved, read as 0**
4:0	MONTH	Binary-Coded Decimal Format Month Register

31.4.10 BCD Format Time YearRegisters(RTC_BCDYEAR)

NAME	RTC_BCDYEAR							
Offset	0x00000024							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	YEAR							
access	R/W-xxxx xxxx							

bit	name	functional description
31:8	-	RFU: **Reserved, read as 0**
7:0	YEAR	Binary-Coded Decimal Format Year Register

31.4.11 RTC AlarmRegisters(RTC_ALARM)

NAME	RTC_ALARM							
Offset	0x00000028							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-		HOUR					
access	U-0		R/W-00 0000					
bit	Bit15	Bit14	BIT13	BIT12	BIT11	BIT10	Bit9	Bit8
name	-	MIN						
access	U-0	R/W-000 0000						
bit	Bit7	Bit6	BIT5	BIT4	BIT3	BIT2	Bit1	Bit0
name	-	SEC						
access	U-0	R/W-000 0000						

bit	name	functional description
31:22	-	RFU: **Reserved, read as 0**
21:16	HOUR	Alarm Hour Register
15	-	RFU: **Reserved, read as 0**
14:8	MIN	Alarm Minute Register
7	-	RFU: **Reserved, read as 0**

bit	name	functional description
6:0	SEC	Alarm Second Register

31.4.12 RTC Time Mark SelectRegister(RTC_TMSEL)

NAME	RTC_TMSEL							
Offset	0x0000002C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				TMSEL			
access	U-0				R/W-0000			

bit	name	functional description
31:4	-	RFU: **Reserved, read as 0**
3:0	TMSEL	Frequency output selection signal (Time Mark Select) 0000:RFU 0001:RFU 0010: Output second counter feed signal, high level width 1s 0011:Output minute counter feed signal, high level width 1s 0100: Output hour counter feed signal, high level width 1s 0101: Output day counter feed signal, high level width 1s 0110:Output alarm clock matching signal 0111:Output 32 seconds square wave signal 1000:RFU 1001:Reverse output second counter feed signal 1010:Reverse output minute counter feed signal 1011:Reverse output hour counter feed signal 1100: Reverse output day counter feed signal 1101:Reverse output alarm clock match signal 1110:RFU 1111:Output RTC internal second time scale square wave

31.4.13 RTC Time AdjustRegister(RTC_ADJUST)

NAME	RTC_ADJUST							
Offset	0x00000030							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							

access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							ADJUST [8]
access	U-0							R/W-x
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ADJUST[7:0]							
access	R/W-xxxx xxxx							

bit	name	functional description
31:9	-	RFU: **Reserved, read as 0**
8:0	ADJUST	LTBC compensation adjustment value (Time Adjust)

31.4.14 RTC Time AdjustSign Register(RTC_ADSIGN)

NAME	RTC_ADSIGN							
Offset	0x00000034							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-							ADSIGN
access	U-0							R/W-x

bit	name	functional description
31:1	-	RFU: **Reserved, read as 0**
0	ADSIGN	LTBCcompensation direction (Adjust Sign) 0:Increase the initial value of the count 1: Decrease the initial value of the count

31.4.15 RTC Sub-Second Counter Register(RTC_SBSCNT)

NAME	RTC_SBSCNT							
Offset	0x0000003C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							

bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	MSCNT							
access	R/W-xxxx xxxx							

bit	name	functional description
31:8	-	RFU: **Reserved, read as 0**
7:0	MSCNT	Milli-Second counter value, effective digit 8bit, precision 3.9ms.

31.4.16 RTC Backup Registers x (RTC_BKRx)

NAME	RTC_BKRx(x=0,1,2,3,4,5,6,7)							
Offset	0x00000070 + x*0x04							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	BKP[31:24]							
access	R/W-xxxx xxxx							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	BKP[23:16]							
access	R/W-xxxx xxxx							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	BKP[15:8]							
access	R/W-xxxx xxxx							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	BKP[7:0]							
access	R/W-xxxx xxxx							

bit	name	functional description
31:0	BKP	RTC Backup Registers, readable and writable, no reset value

32 LCD Display

32.1 Introduction

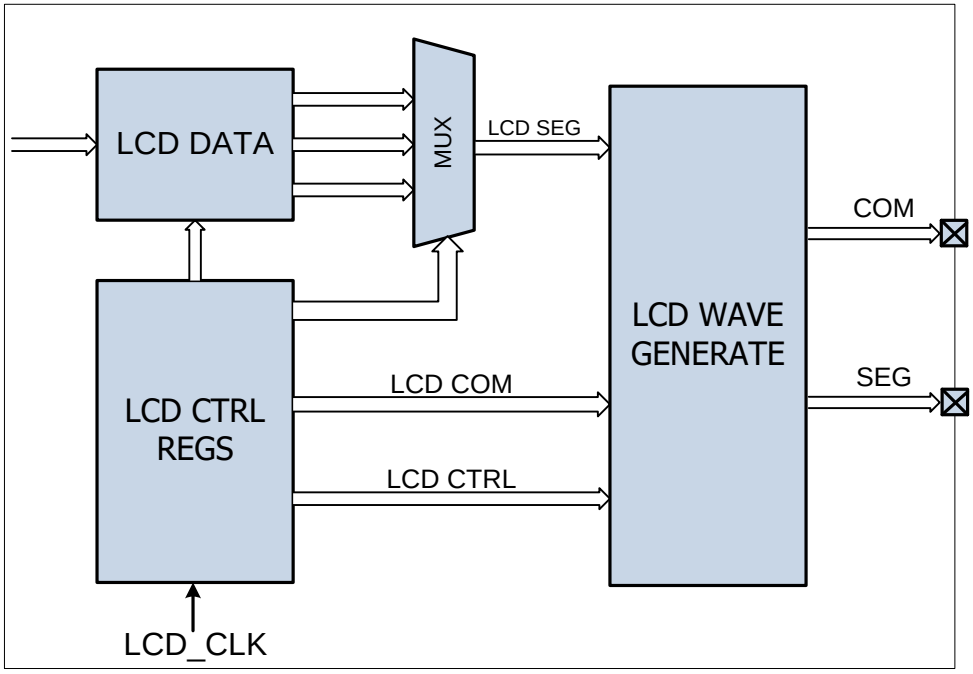
The LCD display driver module is used to drive segmented LCDs, capable of supporting 4, 6 and 8COM, with the maximum number of display segments being 128 segments (4COM), 180 segments (6COM) and 224 segments (8COM) respectively.

Main features:

- Maximum support for 8×28, 6×30, 4×32 display segments
- 1/3bias, 1/4bias
- 16 levels of gray scale adjustable
- LCD driver supports on-chip resistance type
- Support blinking function, and the blinking frequency is adjustable
- Support intermittent light-up function, light-up and off time can be configured
- Support full brightness and full extinction function
- Low power consumption, LCD driver can work in Active mode, Sleep mode and DeepSleep mode
- Supports both Type A and Type B LCD driver waveforms (configurable)
- Typical frame refresh rate 64Hz

32.2 Block Diagram

The block diagram of the LCD display drive module is shown in Figure 32-1:



On-chip BUFFER drive mode

Figure 32-1 LCD display control module block diagram

32.3 IO Configuration

The LCD driver circuit will occupy up to 36 GPIOs when operating. Before using the LCD, you need to set the pins used to the analog function (GPIOx_FCR=11) and turn on the corresponding COMEN or SEGEN. If other analog functions are muxed onto same pin, software must guarantee all analog except for LCD will not use this pin.

32.4 Function Description

32.4.1 Operating Clock and Display Frame Rate

The operating clock of the LCD driver circuit is LSCLK, whose typical frequency is around 32KHz. By configuring the DF register, the frame frequency of the LCD display can be set. The frame frequency is calculated by the following formula (note that DF cannot be 0).

COM number	Frame frequency (Hz)	
	Atype waveform	Btype waveform
4	LCD operating frequency/(4 ×DF[7:0] × 2)	LCD operating frequency/(4 ×DF[7:0] × 4)
6	LCD operating frequency/(6 ×DF[7:0] × 2)	LCD operating frequency/(6 ×DF[7:0] × 4)
8	LCD operating frequency/(8 ×DF[7:0] × 2)	LCD operating frequency/(8 ×DF[7:0] × 4)

Table 32-1 Frame frequency calculation formula

The following table gives an example of the relationship between the DF register values and frame frequency. Frame frequency is set to be around 60Hz typically.

Frame frequency (Hz)	Operating clock (Hz)	4COM		6COM		8COM	
		Atype	Btype	Atype	Btype	Atype	Btype
50	32768	82	41	54	27	41	20
58	32768	70	35	47	24	35	17
64	32768	64	32	42	21	32	16
70	32768	58	29	39	20	29	14
75	32768	54	27	36	18	27	13

Table 32-2 Relationship between typical frame frequency and DF

32.4.2 LCD Type A Scan Waveform

The following figure shows the schematic of the LCD scan waveform for a 1/4 duty, 1/3bias, Type A waveform. Only two common terminals are drawn here as examples.

The 4 common terminals will be activated in sequence. During the time when one common terminal is valid, the SEG outputs the appropriate level, which will be applied to the LCD panel together with the COM level, and the segment code with large voltage difference will be lighten, and the segment code with small voltage difference will not be lighten.

The 1/3 bias means that the LCD driver circuit is able to output 4 drive levels, which are obtained by distributing the bias voltage equally. The LCDBIAS register allows you to configure the VLCD bias voltage level, up to the supply voltage.

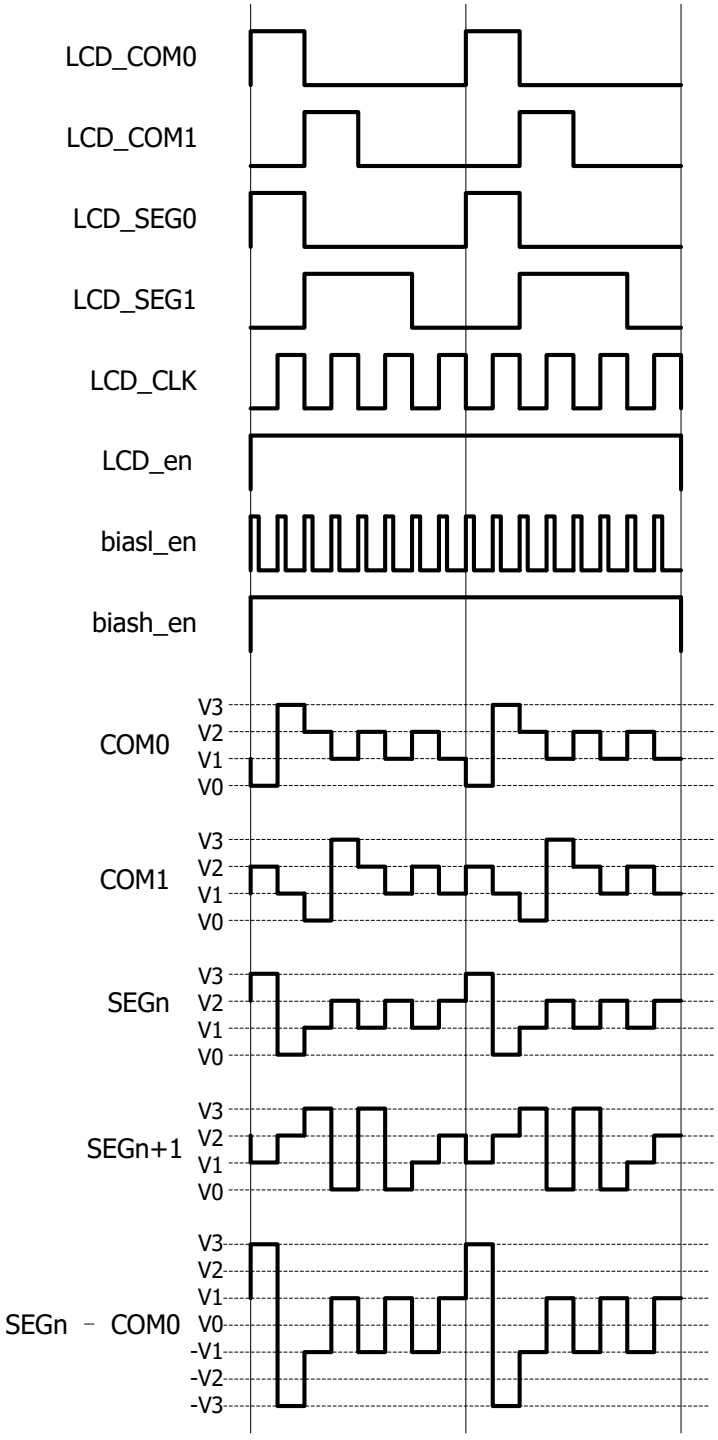


Figure 32-2 LCD drive waveform (1/4 duty, 1/3 bias, type A)

32.4.3 LCD Type B Scan Waveform

The following figure shows the schematic of the LCD scan waveform for a 1/4 duty, 1/3bias, Type B waveform. Only two common terminals are drawn here as examples.

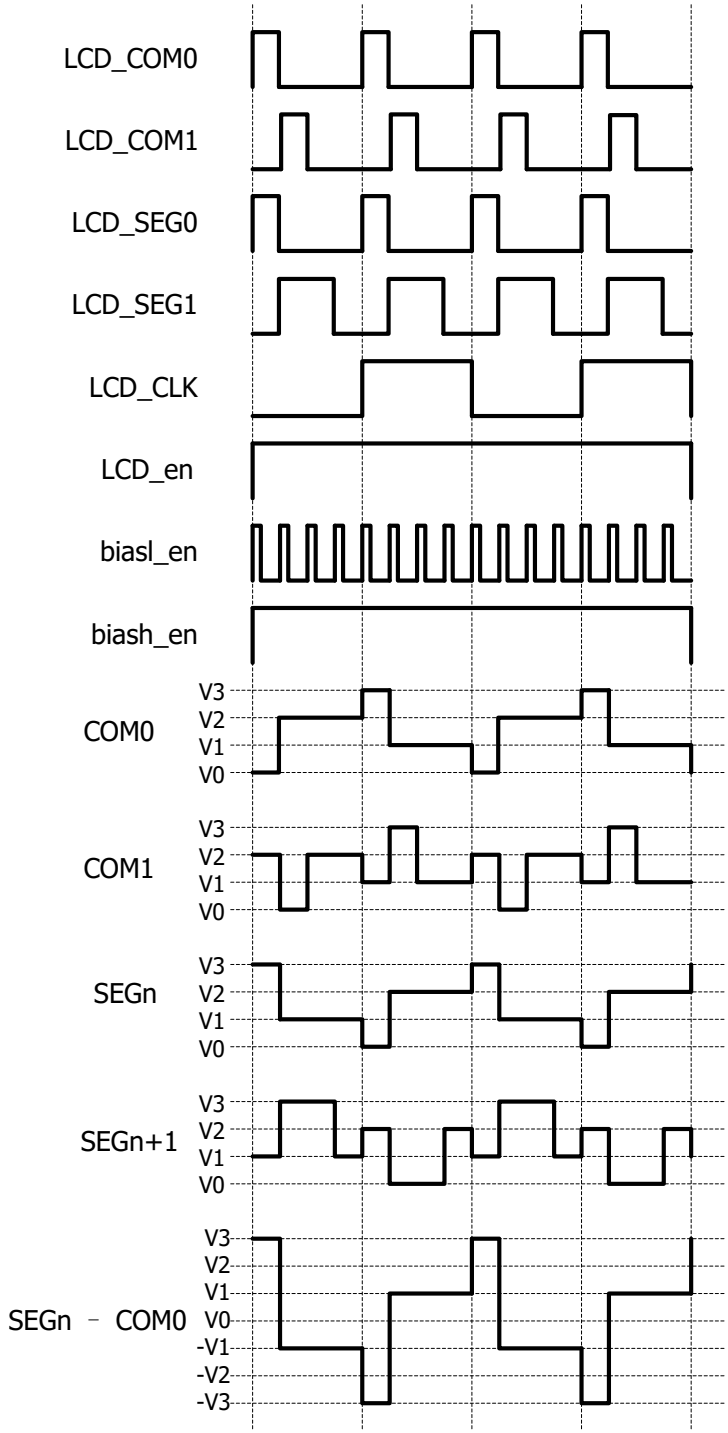


Figure 32-3 LCD drive waveform (1/4 duty, 1/3 bias, type B)

32.4.4 On-chip Resistor Drive Mode

In the on-chip resistor drive mode, the power supply voltage is divided equally through a divider resistor, and the divided voltage is buffered to enhance the drive strength, and the buffer output is connected to the waveform generation module to generate COM and SEG signals. This mode does not require off-chip equipment and has low power consumption.

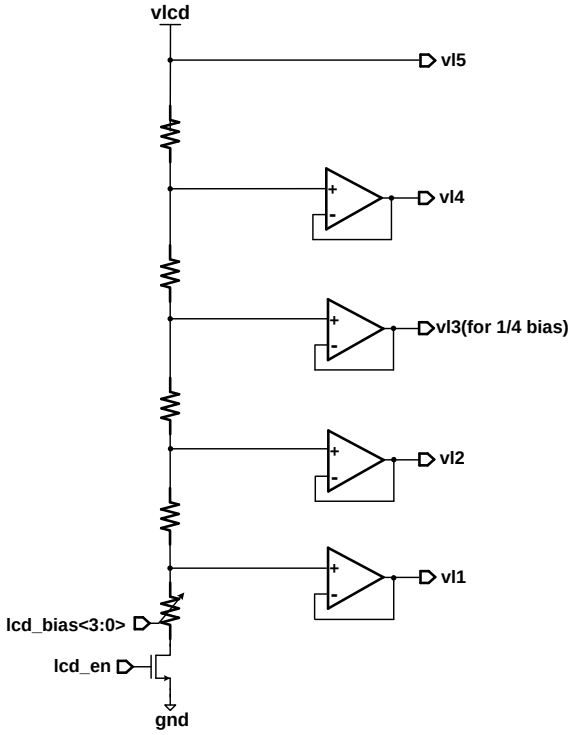


Figure 32-4 LCD on-chip resistor buffer type drive circuit

The output impedance of the driver buffer is approximately 5Kohm.

The LCD drive voltage can be adjusted by configuring the LCDBIAS register, see section 30.4.7 "Bias Voltage Adjustment".

32.4.5 Display Flick Function

The software can set the FLICK bit in the display control register DISPCTRL to 1 to enable the display flicking. After FLICK is enabled, the flicking frequency is determined according to the TON and TOFF register values. Before enabling the FLICK function, TON/TOFF should be set and ENMODE should be set to turn on the display. If TON/TOFF is not set, its reset value is 0 and the display will flick at 64Hz. If the display is not turned on first, the FLICK setting is invalid and there will be no display.

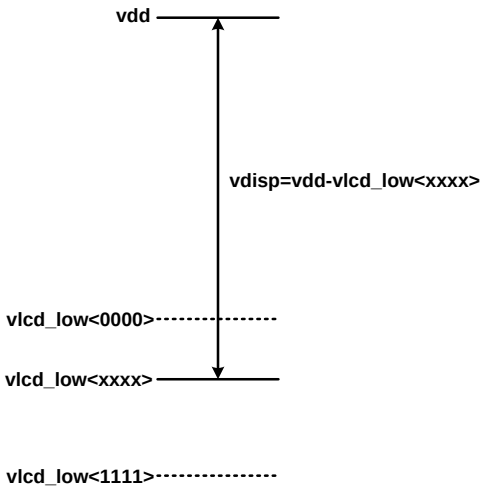
The minimum step of TON/TOFF is T_{step} = COM*DF[7:0]*2*16/32768Hz, the actual ON/OFF time is TON/TOFF * T_{step}. Display and off are synchronized with frame scan, i.e. off after a frame scan, or light up at the beginning of a frame, and the corresponding interrupt is given after off or light up. Since the end-of-frame signal is 64HZ, the count value of TON/TOFF should be the register setting value x16.

32.4.6 Bias Voltage Adjustment

The display voltage range of LCD output can be adjusted to suit different specifications of the LCD panel, and the output voltage range can be expressed as：

VDISP=VDD-VLCD_LOW

VLCD_LOW can be adjusted by LCDBIAS[3:0], LCDBIAS=0000 corresponds to the highest VLCD_LOW voltage, output voltage range VDISP=VDD-VLCD_LOW is minimum; LCDBIAS=1111 corresponds to the lowest VLCD_LOW voltage, output voltage range VDISP=VDD-VLCD_LOW is maximum, as shown in the following figure：



The application should be based on the actual LCD panel characteristics to select the appropriate VDISP voltage.

32.5 Register

Offset	Name	Symbol
LCD(Base address:0x40010C00)		
0x00000000	LCD Control Register	LCD_CR
0x00000004	LCD Test Register	LCD_TEST
0x00000008	LCD Frequency Control Register	LCD_FCR
0x0000000C	LCD Flick Time Register	LCD_FLKT
0x00000014	LCD Interrupt Enable Register	LCD_IER
0x00000018	LCD Interrupt Status Register	LCD_ISR
0x00000024	LCD Data Buffer Registers 0	LCD_DATA0
0x00000028	LCD Data Buffer Registers 1	LCD_DATA1
0x0000002C	LCD Data Buffer Registers 2	LCD_DATA2
0x00000030	LCD Data Buffer Registers 3	LCD_DATA3
0x00000034	LCD Data Buffer Registers 4	LCD_DATA4
0x00000038	LCD Data Buffer Registers 5	LCD_DATA5
0x0000003C	LCD Data Buffer Registers 6	LCD_DATA6
0x00000040	LCD Data Buffer Registers 7	LCD_DATA7
0x00000050	LCD COM Enable Register	LCD_COMEN
0x00000054	LCD SEG Enable Register0	LCD_SEGEN0

32.5.1 LCD Control Register (LCD_CR)

NAME	LCD_CR							
Offset	0x00000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-						IC_CTRL	
access	U-0						R/W-01	
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	ENMODE	FLICK	-		BIAS			
access	R/W-0	R/W-0	U-0		R/W-1110			
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-		BIASMD	ANTIPOLAR	WFT	LMUX		EN
access	U-0		R/W-0	R/W-0	R/W-0	R/W-00		R/W-0

bit	name	functional description
31:18	-	**RFU: Reserved, read as 0**
17:16	**IC_CTRL**	Input Bias Current Control 00:Maximum current 01:Next largest current 10: Second smallest current 11: Minimum current
15	**ENMODE**	LCD Enabling Mode 0:RFU 1: On-chip resistive drive
14	**FLICK**	LCD Flick Enable 1: Display flick, flick frequency is set by TON and TOFF registers 0: Disable flick
13:12	**-**	**RFU: Reserved, read as 0**
11:8	**BIAS**	LCD Bias Voltage Select, for display grayscale control
7:6	**-**	**RFU: Reserved, read as 0**
5	**BIASMD**	Bias Mode Select 1:1/3 Bias 0:1/4 Bias
4	**ANTIPOLAR**	Anti-Polarization Enable 1:COMand SEGgrounded with LCD off 0:COMand SEGfloating with LCD off
3	**WFT**	Waveform Format Select 1:Btype waveform 0:Atype waveform
2:1	**LMUX**	Segment Line Mux 00:4COM 01:6COM 10/11:8COM
0	**EN**	LCD Enable 1:LCDenable 0:LCD disable

LCDBIAS[3:0]	Internal bias voltage at different VDD(V)			
	5	4.5	3.6	3.0
0000	2.74	2.47	1.97	1.64
0001	2.83	2.54	2.03	1.69
0010	2.92	2.62	2.10	1.75
0011	3.01	2.71	2.17	1.81
0100	3.12	2.80	2.24	1.87
0101	3.23	2.90	2.32	1.94
0110	3.35	3.01	2.41	2.01

LCDBIAS[3:0]	Internal bias voltage at different VDD(V)			
	5	4.5	3.6	3.0
0111	3.47	3.13	2.50	2.08
1000	3.61	3.25	2.60	2.17
1001	3.76	3.39	2.71	2.26
1010	3.93	3.53	2.83	2.35
1011	4.10	3.69	2.95	2.46
1100	4.30	3.87	3.09	2.58
1101	4.51	4.06	3.25	2.71
1110	4.75	4.27	3.42	2.85
1111	5.00	4.50	3.60	3.00

32.5.2 LCD Test Register (LCD_TEST)

NAME	LCD_TEST							
Offset	0x00000004							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	LCCTRL	-						TESTEN
access	R/W-0	U-0						R/W-0

bit	name	functional description
31:8	-	**RFU: Reserved, read as 0**
7	**LCCTRL**	LCDtest control bit, valid only in test mode (Line Constant Control) COMand SEG output levels are determined by the pin output data register in test mode. See the table below for the results of SEGor COM output under different settings.
6:1	-	**RFU: Reserved, read as 0**

bit	name	functional description
0	**TESTEN**	Test Mode Enable 1:LCDtest mode enable. In LCD test mode, the LCD pin statically outputs analog DC level, and all register settings related to dynamic scan time and scan waveform are invalid 0:Normal operation mode, test mode is invalid, the relevant test register control is invalid

Pin output level in test mode:

LCCTRL	DISPDATA	COMpin output voltage	SEGpin output voltage
		1/3bias	1/3bias
0	0	V3	V2
0	1	V1	V4
1	0	V2	V3
1	1	V4	V1

32.5.3 Pin Output Data Register in Test Mode

This set of registers is only valid in test mode and shares storage space with related display registers.

NAME	Pin output data register in LCD test mode(TDISPDATA)							
Offset								
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TDISPDATA0	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0
TDISPDATA1	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8
TDISPDATA2	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16
TDISPDATA3	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24
TDISPDATA4	-	-	-	-	-	-	-	-
TDISPDATA5	-	-	-	-	-	-	-	-
TDISPDATA6	-	-	-	-	-	-	-	-
TDISPDATA7	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

32.5.4 LCD Frequency Control Register (LCD_FCR)

NAME	LCD_FCR							
Offset	0x00000008							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DF[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:8	-	RFU: **Reserved, read as 0**
7:0	**DF**	Display Frequency

32.5.5 LCD Flick Time Register (LCD_FLKT)

NAME	LCD_FLKT							
Offset	0x0000000C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	TOFF[7:0]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TON[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	**RFU: Reserved, read as 0**

bit	name	functional description
15:8	**TOFF**	Flick Display-Off Time Register TOFF minimum step is T_{step} = COM*DF[7:0]*2*16/32768Hz, the actual OFF time is TOFF * T_{step}
7:0	**TON**	Flick Display-On Time Register TON minimum step isT_{step} = COM*DF[7:0]*2*16/32768Hz, the actual ON time is TON * T_{step}

32.5.6 LCD Interrupt Enable Register (LCD_IER)

NAME	LCD_IER							
Offset	0x00000014							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						DONIE	DOFFIE
access	U-0						R/W-0	R/W-0

bit	name	functional description
31:2	**-**	RFU: **Reserved, read as 0**
1	**DONIE**	Display-On Interrupt Enable 1:Display-on interrupt enable 0:Display-on interrupt disable
0	**DOFFIE**	Display-OFF Interrupt Enable 1:Display-offinterrupt enable 0:Display-offinterrupt disable

32.5.7 LCD Interrupt Status Register (LCD_ISR)

NAME	LCD_ISR							
Offset	0x00000018							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16

name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						DONIF	DOFFIF
access	U-0						R/W-0	R/W-0

bit	name	functional description
31:2	-	**RFU: Reserved, read as 0**
1	**DONIF**	Display-On Interrupt Flag The hardware generates an interrupt flag when the display changes from off to on, hardware set, write 1 to clear by software
0	**DOFFIF**	Display-OFF Interrupt Flag, write 1 to clear The hardware generates an interrupt flag when the display changes from on to off, hardware set, write 1 to clear by software

32.5.8 LCD Data Buffer Register x (LCD_DATAx)

There are 8 32bit display data registers in the LCD display module. All are readable and writable, and the reset value is 0.

NAME	LCD_DATAx(x=0,1,2,3,4,5,6,7)							
Offset	0x00000024 + x*0x04							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	DSDA[31:24]							
access	R/W-0000 0000							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	DSDA[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DSDA[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DSDA[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:0	DSDA	LCDD is play Data

32.5.8.1 4 COM Data Buffer Register

NAME	4 COM Data Buffer Register							
Offset	0x00000024 ~ 0x00000038							
DISPDATA0	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM0	SEG6 COM0	SEG5 COM0	SEG4 COM0	SEG3 COM0	SEG2 COM0	SEG1 COM0	SEG0 COM0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM0	SEG14 COM0	SEG13 COM0	SEG12 COM0	SEG11 COM0	SEG10 COM0	SEG9 COM0	SEG8 COM0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM0	SEG22 COM0	SEG21 COM0	SEG20 COM0	SEG19 COM0	SEG18 COM0	SEG17 COM0	SEG16 COM0
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	SEG31 COM0	SEG30 COM0	SEG29 COM0	SEG28 COM0	SEG27 COM0	SEG26 COM0	SEG25 COM0	SEG24 COM0
DISPDATA1	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM1	SEG6 COM1	SEG5 COM1	SEG4 COM1	SEG3 COM1	SEG2 COM1	SEG1 COM1	SEG0 COM1
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM1	SEG14 COM1	SEG13 COM1	SEG12 COM1	SEG11 COM1	SEG10 COM1	SEG9 COM1	SEG8 COM1
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM1	SEG22 COM1	SEG21 COM1	SEG20 COM1	SEG19 COM1	SEG18 COM1	SEG17 COM1	SEG16 COM1
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	SEG31 COM1	SEG30 COM1	SEG29 COM1	SEG28 COM1	SEG27 COM1	SEG26 COM1	SEG25 COM1	SEG24 COM1
DISPDATA2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM2	SEG6 COM2	SEG5 COM2	SEG4 COM2	SEG3 COM2	SEG2 COM2	SEG1 COM2	SEG0 COM2
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM2	SEG14 COM2	SEG13 COM2	SEG12 COM2	SEG11 COM2	SEG10 COM2	SEG9 COM2	SEG8 COM2
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM2	SEG22 COM2	SEG21 COM2	SEG20 COM2	SEG19 COM2	SEG18 COM2	SEG17 COM2	SEG16 COM2
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	SEG31 COM2	SEG30 COM2	SEG29 COM2	SEG28 COM2	SEG27 COM2	SEG26 COM2	SEG25 COM2	SEG24 COM2
DISPDATA3	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM3	SEG6 COM3	SEG5 COM3	SEG4 COM3	SEG3 COM3	SEG2 COM3	SEG1 COM3	SEG0 COM3
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

NAME	4 COM Data Buffer Register							
	SEG15 COM3	SEG14 COM3	SEG13 COM3	SEG12 COM3	SEG11 COM3	SEG10 COM3	SEG9 COM3	SEG8 COM3
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM3	SEG22 COM3	SEG21 COM3	SEG20 COM3	SEG19 COM3	SEG18 COM3	SEG17 COM3	SEG16 COM3
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	SEG31 COM3	SEG30 COM3	SEG29 COM3	SEG28 COM3	SEG27 COM3	SEG26 COM3	SEG25 COM3	SEG24 COM3

32.5.8.2 6 COM Data Buffer Register

NAME	6 COM Data Buffer Register							
Offset	0x00000024 ~ 0x00000040							
DISPDATA0	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM0	SEG6 COM0	SEG5 COM0	SEG4 COM0	SEG3 COM0	SEG2 COM0	SEG1 COM0	SEG0 COM0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM0	SEG14 COM0	SEG13 COM0	SEG12 COM0	SEG11 COM0	SEG10 COM0	SEG9 COM0	SEG8 COM0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM0	SEG22 COM0	SEG21 COM0	SEG20 COM0	SEG19 COM0	SEG18 COM0	SEG17 COM0	SEG16 COM0
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
			SEG31 COM0	SEG30 COM0	SEG27 COM0	SEG26 COM0	SEG25 COM0	SEG24 COM0
DISPDATA1	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM1	SEG6 COM1	SEG5 COM1	SEG4 COM1	SEG3 COM1	SEG2 COM1	SEG1 COM1	SEG0 COM1
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM1	SEG14 COM1	SEG13 COM1	SEG12 COM1	SEG11 COM1	SEG10 COM1	SEG9 COM1	SEG8 COM1
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM1	SEG22 COM1	SEG21 COM1	SEG20 COM1	SEG19 COM1	SEG18 COM1	SEG17 COM1	SEG16 COM1
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
			SEG31 COM1	SEG30 COM1	SEG27 COM1	SEG26 COM1	SEG25 COM1	SEG24 COM1
DISPDATA2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM2	SEG6 COM2	SEG5 COM2	SEG4 COM2	SEG3 COM2	SEG2 COM2	SEG1 COM2	SEG0 COM2
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM2	SEG14 COM2	SEG13 COM2	SEG12 COM2	SEG11 COM2	SEG10 COM2	SEG9 COM2	SEG8 COM2

NAME	6 COM Data Buffer Register							
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM2	SEG22 COM2	SEG21 COM2	SEG20 COM2	SEG19 COM2	SEG18 COM2	SEG17 COM2	SEG16 COM2
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
			SEG31 COM2	SEG30 COM2	SEG27 COM2	SEG26 COM2	SEG25 COM2	SEG24 COM2
DISPDATA3	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM3	SEG6 COM3	SEG5 COM3	SEG4 COM3	SEG3 COM3	SEG2 COM3	SEG1 COM3	SEG0 COM3
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM3	SEG14 COM3	SEG13 COM3	SEG12 COM3	SEG11 COM3	SEG10 COM3	SEG9 COM3	SEG8 COM3
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM3	SEG22 COM3	SEG21 COM3	SEG20 COM3	SEG19 COM3	SEG18 COM3	SEG17 COM3	SEG16 COM3
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
			SEG31 COM3	SEG30 COM3	SEG27 COM3	SEG26 COM3	SEG25 COM3	SEG24 COM3
DISPDATA4	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM4	SEG6 COM4	SEG5 COM4	SEG4 COM4	SEG3 COM4	SEG2 COM4	SEG1 COM4	SEG0 COM4
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM4	SEG14 COM4	SEG13 COM4	SEG12 COM4	SEG11 COM4	SEG10 COM4	SEG9 COM4	SEG8 COM4
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM4	SEG22 COM4	SEG21 COM4	SEG20 COM4	SEG19 COM4	SEG18 COM4	SEG17 COM4	SEG16 COM4
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
			SEG31 COM4	SEG30 COM4	SEG27 COM4	SEG26 COM4	SEG25 COM4	SEG24 COM4
DISPDATA5	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM5	SEG6 COM5	SEG5 COM5	SEG4 COM5	SEG3 COM5	SEG2 COM5	SEG1 COM5	SEG0 COM5
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM5	SEG14 COM5	SEG13 COM5	SEG12 COM5	SEG11 COM5	SEG10 COM5	SEG9 COM5	SEG8 COM5
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM5	SEG22 COM5	SEG21 COM5	SEG20 COM5	SEG19 COM5	SEG18 COM5	SEG17 COM5	SEG16 COM5
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
			SEG31 COM5	SEG30 COM5	SEG27 COM5	SEG26 COM5	SEG25 COM5	SEG24 COM5

32.5.8.3 8 COM Data Buffer Register

NAME	8 COM Data Buffer Register							
Offset	0x00000024 ~ 0x00000040							
DISPDATA0	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM0	SEG6 COM0	SEG5 COM0	SEG4 COM0	SEG3 COM0	SEG2 COM0	SEG1 COM0	SEG0 COM0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM0	SEG14 COM0	SEG13 COM0	SEG12 COM0	SEG11 COM0	SEG10 COM0	SEG9 COM0	SEG8 COM0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM0	SEG22 COM0	SEG21 COM0	SEG20 COM0	SEG19 COM0	SEG18 COM0	SEG17 COM0	SEG16 COM0
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
					SEG27 COM0	SEG26 COM0	SEG25 COM0	SEG24 COM0
DISPDATA1	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM1	SEG6 COM1	SEG5 COM1	SEG4 COM1	SEG3 COM1	SEG2 COM1	SEG1 COM1	SEG0 COM1
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM1	SEG14 COM1	SEG13 COM1	SEG12 COM1	SEG11 COM1	SEG10 COM1	SEG9 COM1	SEG8 COM1
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM1	SEG22 COM1	SEG21 COM1	SEG20 COM1	SEG19 COM1	SEG18 COM1	SEG17 COM1	SEG16 COM1
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
					SEG27 COM1	SEG26 COM1	SEG25 COM1	SEG24 COM1
DISPDATA2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM2	SEG6 COM2	SEG5 COM2	SEG4 COM2	SEG3 COM2	SEG2 COM2	SEG1 COM2	SEG0 COM2
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM2	SEG14 COM2	SEG13 COM2	SEG12 COM2	SEG11 COM2	SEG10 COM2	SEG9 COM2	SEG8 COM2
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM2	SEG22 COM2	SEG21 COM2	SEG20 COM2	SEG19 COM2	SEG18 COM2	SEG17 COM2	SEG16 COM2
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
					SEG27 COM2	SEG26 COM2	SEG25 COM2	SEG24 COM2
DISPDATA3	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM3	SEG6 COM3	SEG5 COM3	SEG4 COM3	SEG3 COM3	SEG2 COM3	SEG1 COM3	SEG0 COM3
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

NAME	8 COM Data Buffer Register							
	SEG15 COM3	SEG14 COM3	SEG13 COM3	SEG12 COM3	SEG11 COM3	SEG10 COM3	SEG9 COM3	SEG8 COM3
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM3	SEG22 COM3	SEG21 COM3	SEG20 COM3	SEG19 COM3	SEG18 COM3	SEG17 COM3	SEG16 COM3
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
					SEG27 COM3	SEG26 COM3	SEG25 COM3	SEG24 COM3
DISPDATA4	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM4	SEG6 COM4	SEG5 COM4	SEG4 COM4	SEG3 COM4	SEG2 COM4	SEG1 COM4	SEG0 COM4
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM4	SEG14 COM4	SEG13 COM4	SEG12 COM4	SEG11 COM4	SEG10 COM4	SEG9 COM4	SEG8 COM4
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM4	SEG22 COM4	SEG21 COM4	SEG20 COM4	SEG19 COM4	SEG18 COM4	SEG17 COM4	SEG16 COM4
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
					SEG27 COM4	SEG26 COM4	SEG25 COM4	SEG24 COM4
DISPDATA5	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM5	SEG6 COM5	SEG5 COM5	SEG4 COM5	SEG3 COM5	SEG2 COM5	SEG1 COM5	SEG0 COM5
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM5	SEG14 COM5	SEG13 COM5	SEG12 COM5	SEG11 COM5	SEG10 COM5	SEG9 COM5	SEG8 COM5
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM5	SEG22 COM5	SEG21 COM5	SEG20 COM5	SEG19 COM5	SEG18 COM5	SEG17 COM5	SEG16 COM5
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
					SEG27 COM5	SEG26 COM5	SEG25 COM5	SEG24 COM5
DISPDATA6	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM6	SEG6 COM6	SEG5 COM6	SEG4 COM6	SEG3 COM6	SEG2 COM6	SEG1 COM6	SEG0 COM6
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM6	SEG14 COM6	SEG13 COM6	SEG12 COM6	SEG11 COM6	SEG10 COM6	SEG9 COM6	SEG8 COM6
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM6	SEG22 COM6	SEG21 COM6	SEG20 COM6	SEG19 COM6	SEG18 COM6	SEG17 COM6	SEG16 COM6
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24

NAME	8 COM Data Buffer Register							
					SEG27 COM6	SEG26 COM6	SEG25 COM6	SEG24 COM6
DISPDATA7	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SEG7 COM7	SEG6 COM7	SEG5 COM7	SEG4 COM7	SEG3 COM7	SEG2 COM7	SEG1 COM7	SEG0 COM7
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	SEG15 COM7	SEG14 COM7	SEG13 COM7	SEG12 COM7	SEG11 COM7	SEG10 COM7	SEG9 COM7	SEG8 COM7
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	SEG23 COM7	SEG22 COM7	SEG21 COM7	SEG20 COM7	SEG19 COM7	SEG18 COM7	SEG17 COM7	SEG16 COM7
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
					SEG27 COM7	SEG26 COM7	SEG25 COM7	SEG24 COM7

32.5.9 LCD COM Enable Register (LCD_COMEN)

NAME	LCD_COMEN							
Offset	0x00000050							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-				COMEN[3:0]			
access	U-0				R/W-0000			

bit	name	functional description
31:4	-	RFU: **Reserved, read as 0**
3:0	COMEN	LCD COM Output Enable Control 1: COM output enable 0: COM output disable

32.5.10 LCD SEG Enable Register (LCD_SEGEN0)

NAME	LCD_SEGEN0							
Offset	0x00000054							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	SEG31_COM7_EN	SEG30_COM6_EN	SEG29_COM5_EN	SEG28_COM4_EN	SEGEN[27:24]			
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0000			
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	SEGEN[23:16]							
access	R/W-0000 0000							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	SEGEN[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SEGEN[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31	SEG31_COM7_EN	SEG31 or COM7 enable 1: SEG or COM output enable 0: SEG or COM output disable
30	SEG30_COM6_EN	SEG30 or COM6 enable 1: SEG or COM output enable 0: SEG or COM output disable
29	SEG29_COM5_EN	SEG29 or COM5 enable 1: SEG or COM output enable 0: SEG or COM output disable
28	SEG28_COM4_EN	SEG28 or COM4 enable 1: SEG or COM output enable 0: SEG or COM output disable
27:0	SEGEN	SEG Enable Each bit corresponds to a specific SEG 1: SEG output enable 0: SEG output disable

33 ADC

33.1 Introduction

The FM33LE0xxA has a built-in 2Msps 12bit SAR-ADC, which can measure temperature, battery voltage or other DC signals. The main features are:

- Operating voltage 1.65 to 5.5V
- Input voltage range 0 to VREF+
- Maximum sampling rate 2Msps (F_{ADC}=32Mhz)
- 16 single-ended input channels, including temperature sensor, internal reference voltage, op amp output x2, 12 external channels
- 12 external channels (maximum rate 2Msps), 2 internal channels (temperature sensor channel, AVREF sampling channel)
- Configurable sample-and-hold time
- Support single conversion and continuous conversion
- Support DMA
- Support over-sampling hardware average, up to 16bit output (256 times average)

33.2 Block diagram

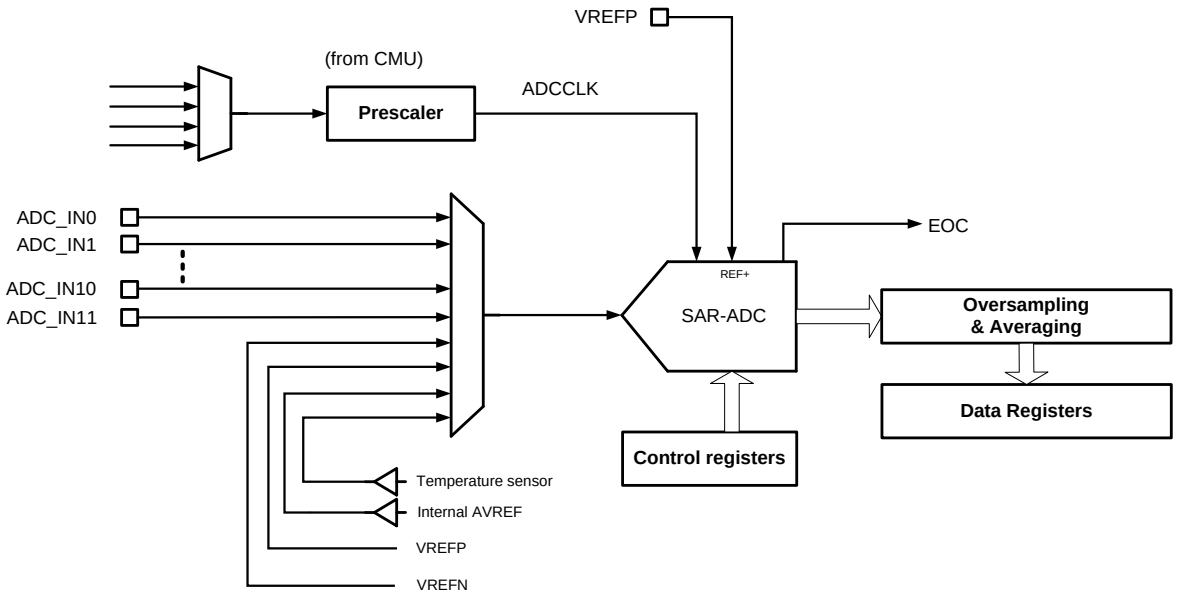


Figure 33-5 ADC block diagram

The ADC has 16 analog input channels, of which 0~11 are directly connected to the PAD input and used as external channels; the remaining 4 channels are used as internal channels.

。

33.3 Input channel

The ADC supports 4 internal channels and 12 external channels.

Channel	IO	Description
ADC_IN0	PC9	External fast channel, only supports single-ended input
ADC_IN1	PC10	
ADC_IN2	PD11	
ADC_IN3	PD0	
ADC_IN4	PD1	
ADC_IN5	PD2	
ADC_IN6	PA13	
ADC_IN7	PA14	
ADC_IN8	PC7	
ADC_IN9	PC8	
ADC_IN10	PA15	
ADC_IN11	PC6	
TS	N/A	Temperature sensor sampling channel
AVREF		Internal reference source sampling channel
VREFP	Reference power supply pin	
VREFN	Reference ground pin	

33.4 Work sequence

The ADC has two timing sequences, offset and normal operation. After power-on, it is recommended to perform a front calibration to obtain better performance. No recalibration is required after calibration, unless the chip is reset globally or the power supply voltage and temperature change greatly. Offset calibration is started by the software CALEN. The calibration process consists of multiple sampling and conversion periods. The number of sampling and conversion periods is set by the OSCAL_CYCLE register, generally, the recommended configuration is 8. 128 ADCCLK cycles are required to complete calibration.Set EOCAL flag register after completing calibration.

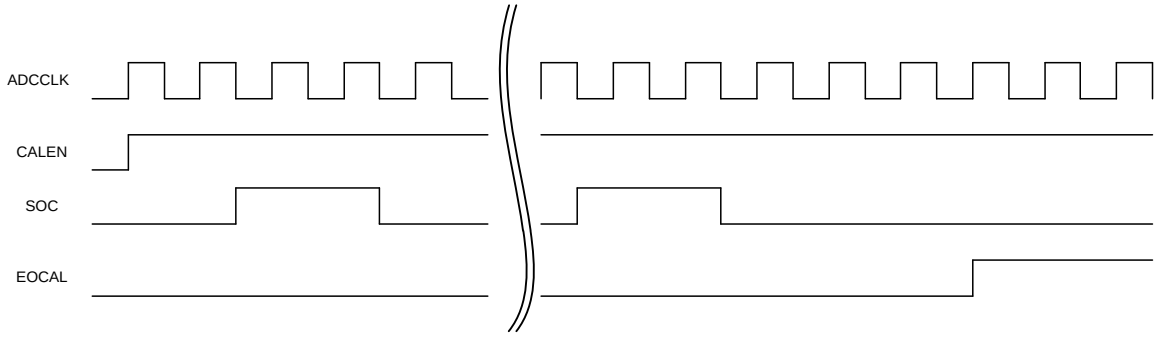


Figure 33-2 ADC Calibration Timing

During sampling and conversion, ADC sampling is started through the SOC signal, and the ADC sampling time is controlled by the high level width of SOC. The conversion is started after the SOC decreases. The conversion period is 14 ADC_CLK cycles, and the sampling time can be configured, with the minimum of 2 ADC_CLK and the maximum of 512 ADC_CLK.

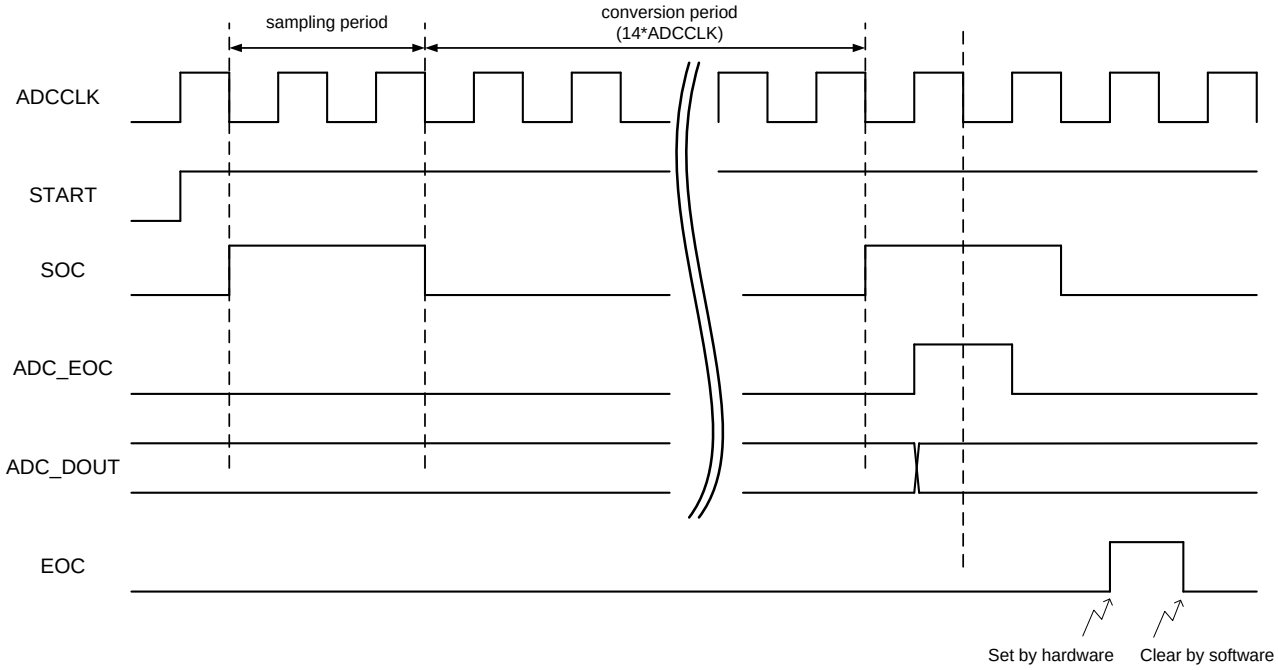


Figure 33-3 ADC Sample Conversion Timing

- The timing diagram of the multi-channel conversion sequence is as follows:

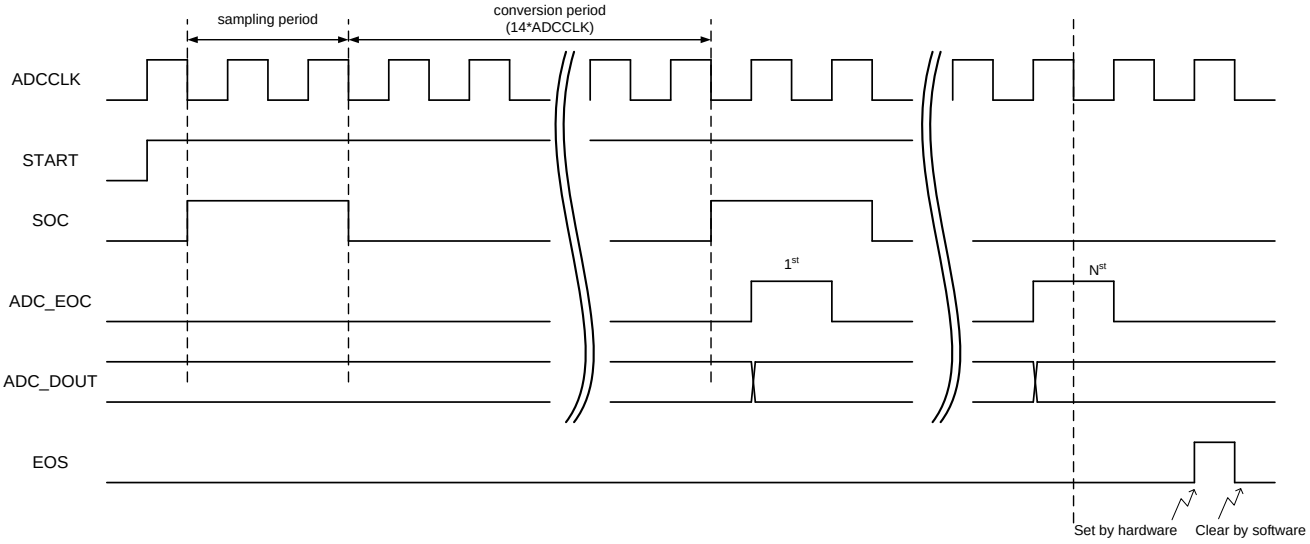


Figure 33-4 ADC sampling sequence timing

When the ADC_CLK is 16Mhz and the sampling time is 2*ADCCLK, the conversion rate is 1Msps. When the ADC_CLK is 32Mhz and the sampling time is 2*ADCCLK, the conversion rate is 2 MSPS.

33.5 Functional description

33.5.1 Sampling value and actual voltage conversion

ADC generally uses the power supply voltage as the reference voltage. When the power supply voltage changes, the conversion value corresponding to a specific input signal level will also change. In order to get accurate absolute voltage, the solution is as follows:

- The ADC samples and converts the fast reference output, and converts the VDDA voltage by the following formula (assuming that the 12-bit conversion result of the ADC sampled fast reference source is AVREF_DATA) :

$$VDDA = \frac{4095}{AVREF\_DATA} \times 1.0V$$

- Assuming that the ADC sampling value for a certain input channel is ADC_DATA, the actual voltage of a certain input channel (12bit output) can be obtained by the following formula:

$$V_{CHANNEL} = \frac{ADC\_DATA}{AVREF\_DATA} \times 1.0V$$

When used, 1.0V is replaced by the actual voltage value of AVREF in FLASH chip

33.5.2 Temperature sensor

The ADC uses the internal channel to measure the temperature sensor output voltage to obtain the conversion data TS_DARA. Then the following formula can be used to calculate the current temperature:

$$\text{Temperature} = \frac{1.0 \times \frac{TS\_DATA}{AVREF\_DATA} - TS\_CAL30}{Slope} + 30$$

Among them, *TS_DATA* is the conversion value of the ADC sampling of the current temperature sensor output; because the accurate level of the current VDDA is not known, this conversion value needs to be scaled according to the conversion result of AVREF; *TS_CAL30* is the conversion result of temperature calibration under the conditions of 30C+/-1C and VDDA=3.0V during chip production. This data is stored in the flash.

Slope represents the output slope of the temperature sensor. Since there is only 30C single point calibration data in the chip, in this case, the temperature can be calculated using the typical slope value of the temperature sensor. Please refer to the electrical parameters in the chip manual.

When using the temperature sensor function, it is necessary to enable the output of the temperature sensor of the internal reference source, that is, to set the PTAT_EN register, and set the VPTAT_BUFFER_EN register to enable the PTAT buffer. After at least 10us establishment time,

enable the ADC to sample the temperature sensor channel.

33.5.3 Programmable sampling time

By adjusting the sampling time, the internal resistance of different input signal sources can be adapted. The sampling time can be selected through the SMTS1 and SMTS2 registers:

SMTSx	Sampling cycles (T_{ADCCLK})
0000	2
0001	4
0010	8
0011	12
0100	16
0101	32
0110	64
0111	80
1000	96
1001	128
1010	160
1011	192
1100	256
1101	320
1110	384
1111	512

The actual ADC sampling conversion time:

T_{CONV} = (Sampling Cycles + 14) * T_{ADCCLK}

The ADC sampling time is mainly determined by the sampling capacitor, the output impedance of the sampled signal, the internal input channel impedance of the chip, and the required sampling accuracy.

The following figure is a schematic diagram of the circuit structure of a single-ended input channel:

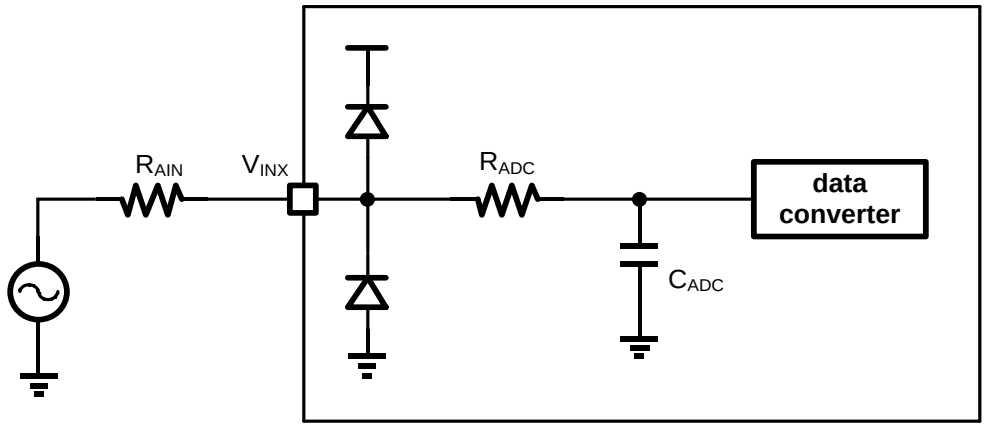


Figure 33-5 ADC single-ended input channel schematic diagram

The required sampling time can be estimated according to the following formula:

$$T_{samp} = \ln\left(\frac{2^n}{SA}\right) \times \left(R_{AIN} + R_{ADC}\right) \times C_{ADC}$$

Among them, $n = 12$, SA represents the allowable sampling error, for example, 0.25 represents 1/4 LSB

In the application, you should calculate and determine the acceptable sampling time according to the relevant parameters and the system parameters in the chip manual, and configure the working clock, sampling period, etc. of the ADC according to this result.

33.5.4 Conversion mode

ADC supports the following conversion modes:

- Single conversion
  - Semi-automatic trigger
  - Automatic trigger
- Continuous conversion

Conversion start can be triggered by software or events, and multiple event trigger sources can be selected through registers.

33.5.4.1 Single Coversion Mode

support semi-automatic trigger and automatic trigger.

Automatic trigger mode: After a software or hardware trigger event starts ADC conversion, the ADC will sample all enabled channels sequentially. After a single channel is sampled, the EOC (End of Conversion) flag is set, and after all channels are sampled, the EOS (End of Sequence) flag is set, and the conversion ends. Assuming that channels 0, 3, and 5 are enabled.

- 1st trigger event: Channels 0, 3, and 5 are sampled sequentially, three EOCs are generated in the process, and EOS is finally generated
- 2nd trigger event: Repeat the above process

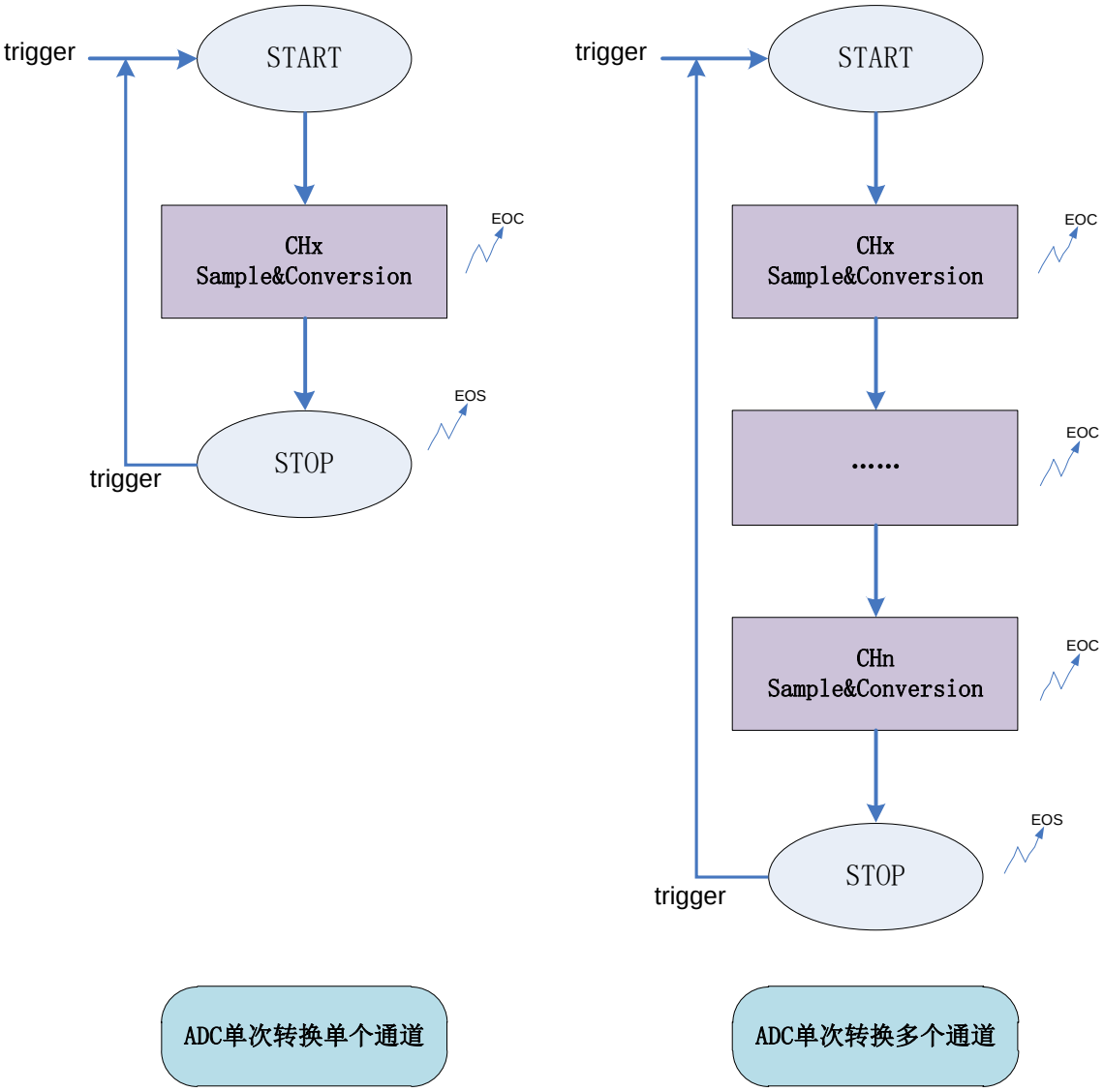


Figure 33-6 ADC single conversion fully automatic trigger mode

Semi-automatic trigger mode: A software or hardware trigger event will only start the ADC once to convert an enabled channel. For example, channel 0, 3 or 5 is enabled.

- 1st trigger event: Channel 0 is sampled to generate EOC
- 2nd trigger event: Channel 3 is sampled to generate EOC
- 3rd trigger event: Channel 5 is sampled to generate EOC and EOS
- 4th trigger event: Channel 0 is sampled to generate EOC
- 5th trigger event: Channel 3 is sampled to generate EOC

……

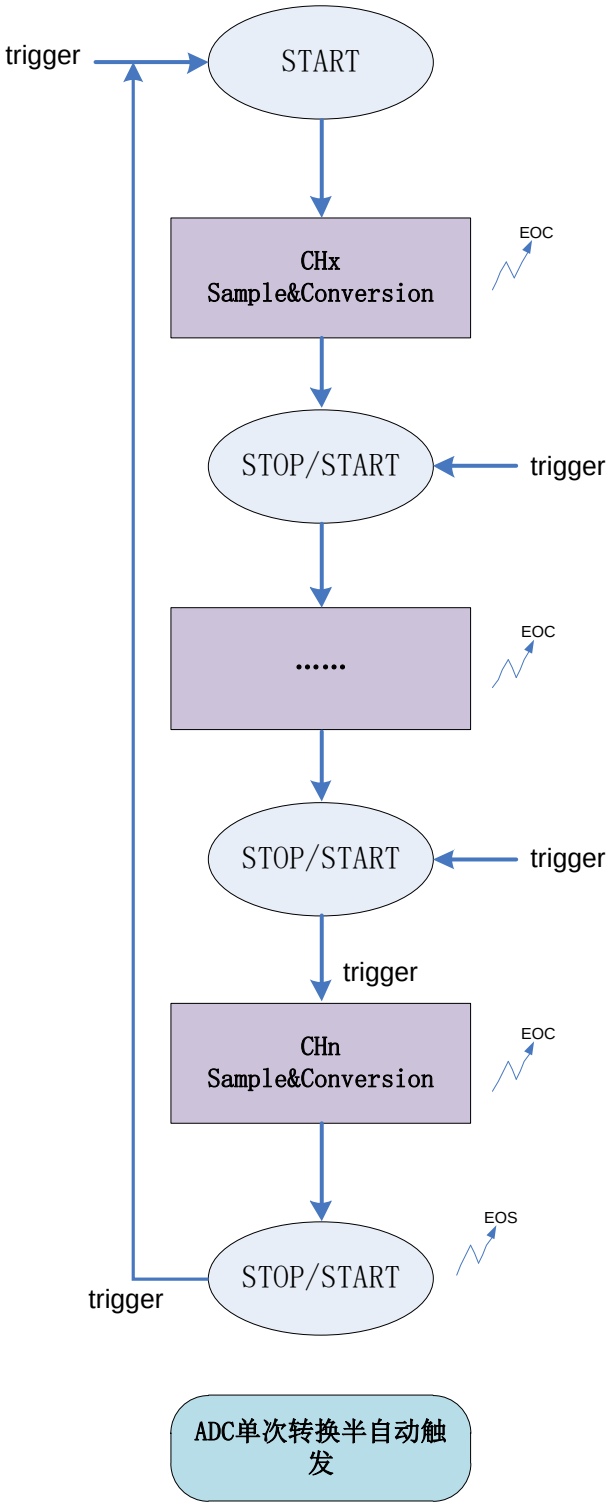


Figure 33-7 ADC single conversion semi-automatic trigger mode

33.5.4.2 Continuous Conversion Mode

After the trigger event arrives, all enabled channels are sampled, and the ADC will not stop automatically, but will sample cyclically until the software stops the ADC.

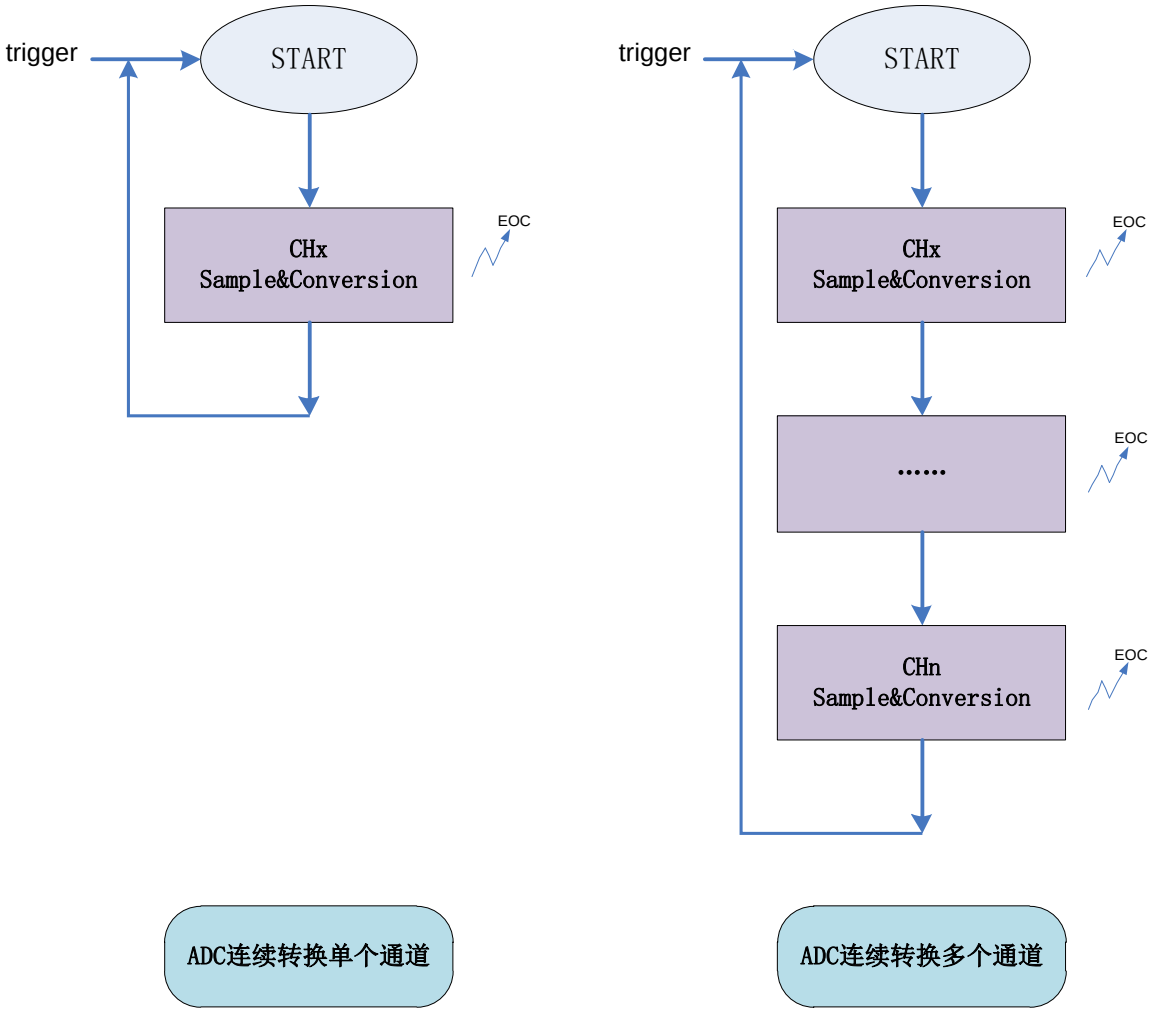


Figure 33-8 ADC Continuous Conversion Mode

After each channel is sampled, the data is stored in the ADC_DATA register. The software should read the data in time before the next conversion, or move the data through DMA.If the data cannot be taken away in time, it will cause an Overrun, the overrun flag is set, and an interrupt can be issued.

33.5.5 Conversion trigger

After the ADC is enabled, the conversion trigger supports software or hardware event triggers.

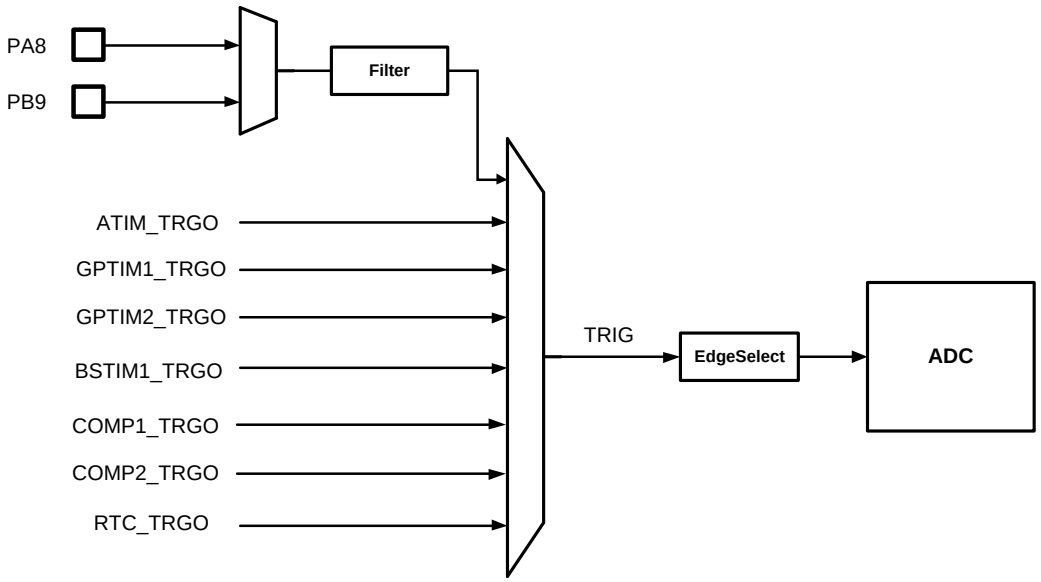
Software trigger

The software starts the conversion by setting the START register.

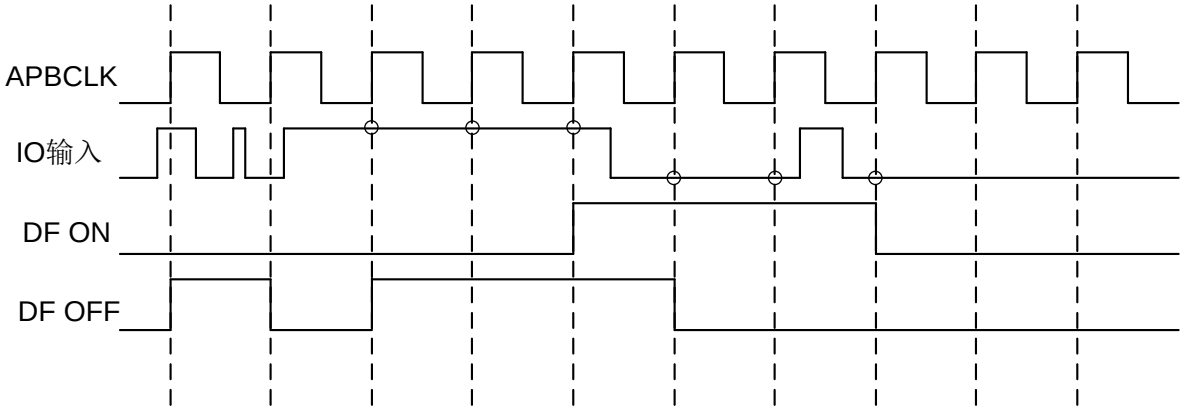
Hardware trigger

ADC has the following hardware trigger sources: RTC_TRGO, ATIM_TRGO, GPTIM0_TRGO, GPTIM1_TRGO, BSTIM_TRGO, comparator output, and 2 GPIO input signals (PA8 and PB9); through the IOTREN register, you can select the rising edge, falling edge or rising and falling edge

of the IO input signal to trigger conversion. If the ADC is in the process of conversion, the trigger signal that comes at this time will be ignored.



The trigger signal input by the pin is selected and then input to the digital filtering and edge detection module. The implementation principle of digital filtering is the same as the digital filtering of IO interrupt, that is, the input signal is sampled three times in succession using APBCLK, and it is considered as a legal signal when the level is the same. The filtered signal then generates a rising edge, falling edge or double-edge trigger signal through the edge selection circuit.



When using the pin input signal to trigger ADC conversion, the following configuration is required:

- Configure PA8 or PB9 as input
- Set IOTRFEN and IOTREN register，configure filtering、trigger edge
- Configure the EXTS register and select the trigger source as external pin input
- Configure ADC working clock、sampling time、sampling channel,etc
- Enable ADC

- A specific level change input on the specified IO will trigger the ADC conversion

33.5.6 Oversampling and hardware averaging

ADC supports hardware oversampling and averaging, which can improve the resolution to a certain extent. The principle is that for low-speed input signals, ENOB can be increased by averaging after multiple consecutive sampling. The oversampling formula is as follows:

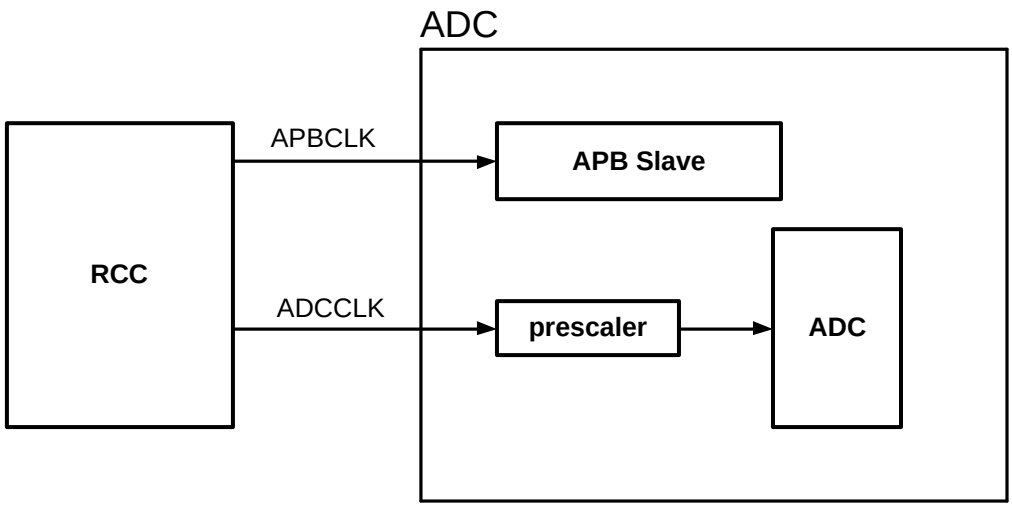
$$result = \frac{\sum_{n=1}^{N} CONVERSION_n}{M}$$

Where N is the oversampling multiple, which can be configured as 2/4/8/16/32/64/128/256, M is the result right shift number, the maximum right shift is 8bit; since each conversion result is 12bit,the result of the maximum accumulation of 256 times is 20bit. After shifting, the final result of 12~16bit can be obtained. The ADC output result is only 16 bits at most. If the result exceeds 16 bits after right shifting, the high bits will also be discarded.

When oversampling is enabled, the EOC signal is set after N consecutive samples. To the application and DMA, it feels as if it has only undergone one sample conversion.

33.5.7 ADC working clock

The ADC adopts a dual clock structure, using APBCLK and an asynchronous working clock ADCCLK at the same time.



33.5.8 Data conflict and automatic waiting

The EOC flag will be set after each conversion. Software or DMA will automatically clear the EOC

after reading the ADC_DATA register. It can also be cleared by software by writing 1 to it. When the EOC flag is not cleared, the arrival of new conversion data will cause data overrun; there are two overrun modes:

OVRMOD=0: keep the old data, discard the new data

OVRMOD=1: New data is written to overwrite old data

When an overrun occurs when using DMA, a new DMA request will not be initiated until the OVR flag is cleared by software.

The ADC controller also supports automatic waiting. If the WAIT register is set by software, the ADC controller will not initiate a new conversion before the ADC_DATA register is read; hardware trigger events that arrive in the waiting state will also be ignored. The following figure is a schematic diagram of enabling automatic waiting when the software triggers the continuous mode:

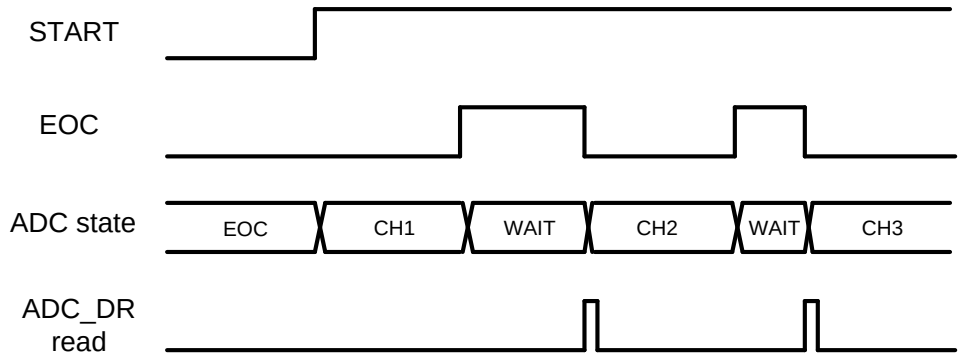


Figure 33-9 Automatic waiting in continuous mode

33.5.9 DMA

In multi-channel conversion or continuous conversion, using DMA to move the conversion result is an efficient solution. When DMAEN is enabled, when each conversion is completed (EOC), the ADC controller module will generate a DMA request to notify the DMA to move the result in the data register to the specified SRAM address. The ADC's DMA interface supports single mode and loop mode:

In automatic mode (ADC_CFGR2.SEMI=0) and semi-automatic mode (ADC_CFGR2.SEMI=1), ADC's DMA interface can support both single mode and loop mode

Single mode

After the conversion is completed, the data transfer is initiated. This process will be repeated until the DMA transfer length configured by the software is completed, and then the ADC controller will

automatically stop the conversion (by receiving the DMA transfer completion interrupt flag signal), turn off the ADC, and no longer initiate requests to the DMA. This mode is mainly used to sample a certain length of a specific analog signal.

Loop mode

Cooperating with the DMA loop mode, the ADC continues to cycle conversions and initiates DMA requests until the software stops the conversion. This mode can be used to process continuous analog signal sampling.

The ADC_CFGR2.SEMI register can be used to configure automatic or semi-automatic conversion in DMA mode, the loop mode or single mode can be configured by the CHxCIRC in DMA module, and the transmission length can be configured by the CHxTSIZE in DMA module. The effect of the register configuration combination is shown in the following table.

ADC_CFGR2.CONT	ADC_CFGR2.SEMI	DMA CHxCIRC	DMA CHxTSIZE	declaration
0	0	0	N	In single-automatic mode, after the trigger event arrives, the enable input channel is converted N times successively and the data is transported to the SRAM; All enable input channels are sampled, and the ADC automatically stops and waits for the next trigger. If the number of data transfers reaches N, DMA shuts down automatically.
0	0	1	N	In single-automatic mode (cyclic storage), after the trigger event arrives, the data is continuously converted and transported for the enable input channels in turn. Until all the enable input channels have been sampled, ADC automatically stops and waits for the next trigger. RAM data space length is N. When the length of transported data exceeds N, it returns to the start address pointed by the DMA channel pointer and overwrites the original data.
0	1	0	N	In the single semi-automatic mode, each trigger event initiates one conversion and samples all enabled input channels in turn. When the number of data transfers reaches N, DMA shuts down automatically.

0	1	1	N	Single semi-automatic mode (cyclic storage), each trigger event initiates one conversion, in turn sampling all enabled input channels; RAM data space length is N. When the length of transported data exceeds N, it returns to the start address pointed by the DMA channel pointer and overwrites the original data.
1	x	0	N	In continuous mode, ADC samples the enabling channel continuously after triggering, and DMA carries data continuously until it completes N times of data carrying, and DMA shuts down automatically.
1	x	1	N	In continuous mode (cyclic storage), ADC continuously samples the enabling channel after triggering, and DMA continuously carries data; RAM data space length is N. When the length of transported data exceeds N, the original data is overwritten by returning to the original address. Until the software shuts down DMA.

Table 33-1 DMA configuration and function

With DMA enabled, if an overrun occurs, the ADC controller will no longer send DMA requests until the OVR flag is cleared.

Note that DMA transfers can be supported in both single and continuous conversion modes; DMA transfer length is defined in terms of EOC times, not EOS, that is, DMA only cares how many times ADC_DATA is transported.

33.5.9.1 DMA Use Case

Uae case 1：

ADC is configured in single-automatic mode, enabling three ADC input channels, and DMA TSIZE is set to six. After the first trigger, the ADC samples the three channels in turn, the DMA carries the data three times, and then the ADC and DMA suspend their work, waiting for the second trigger. After the second trigger, the ADC samples three channels in turn, and DMA carries the data three times. Six DMA transfers are completed and the DMA channel is closed.

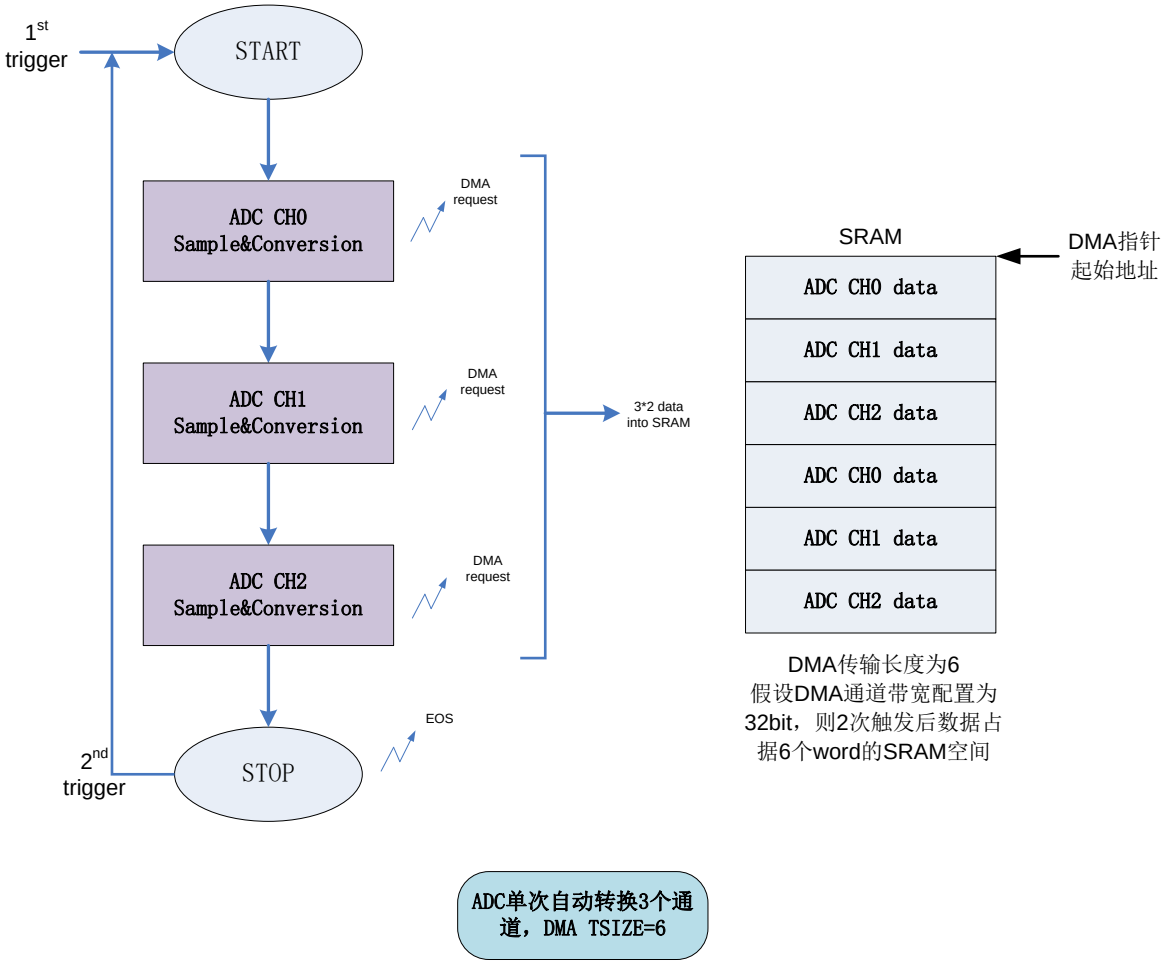


Figure 33-10 ADCsingle automatic trigger+DMA case 1

Use case 2：

ADC is configured in single-automatic mode, enabling three ADC input channels, and DMA TSIZE is set to two. After the first trigger, ADC samples three channels in turn. After DMA carries the data twice, DMA and ADC shut down automatically without sampling the third input channel.

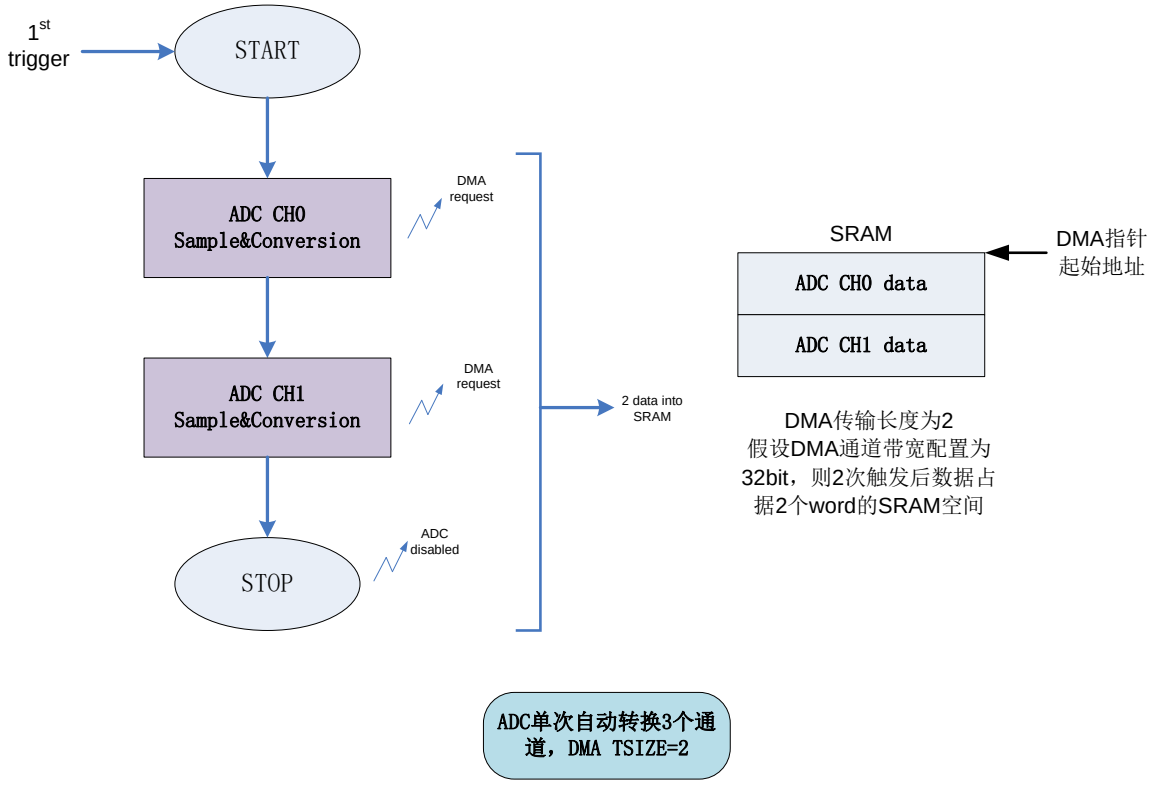


Figure 33-11 ADC single automatic trigger+DMA case 2

Use case 3：

ADC is configured in single semi-automatic mode with three ADC input channels enabled and DMA TSIZE is set to six. After each trigger, DMA carries one transfer, and after a total of six transfers, ADC and DMA are shut down automatically.

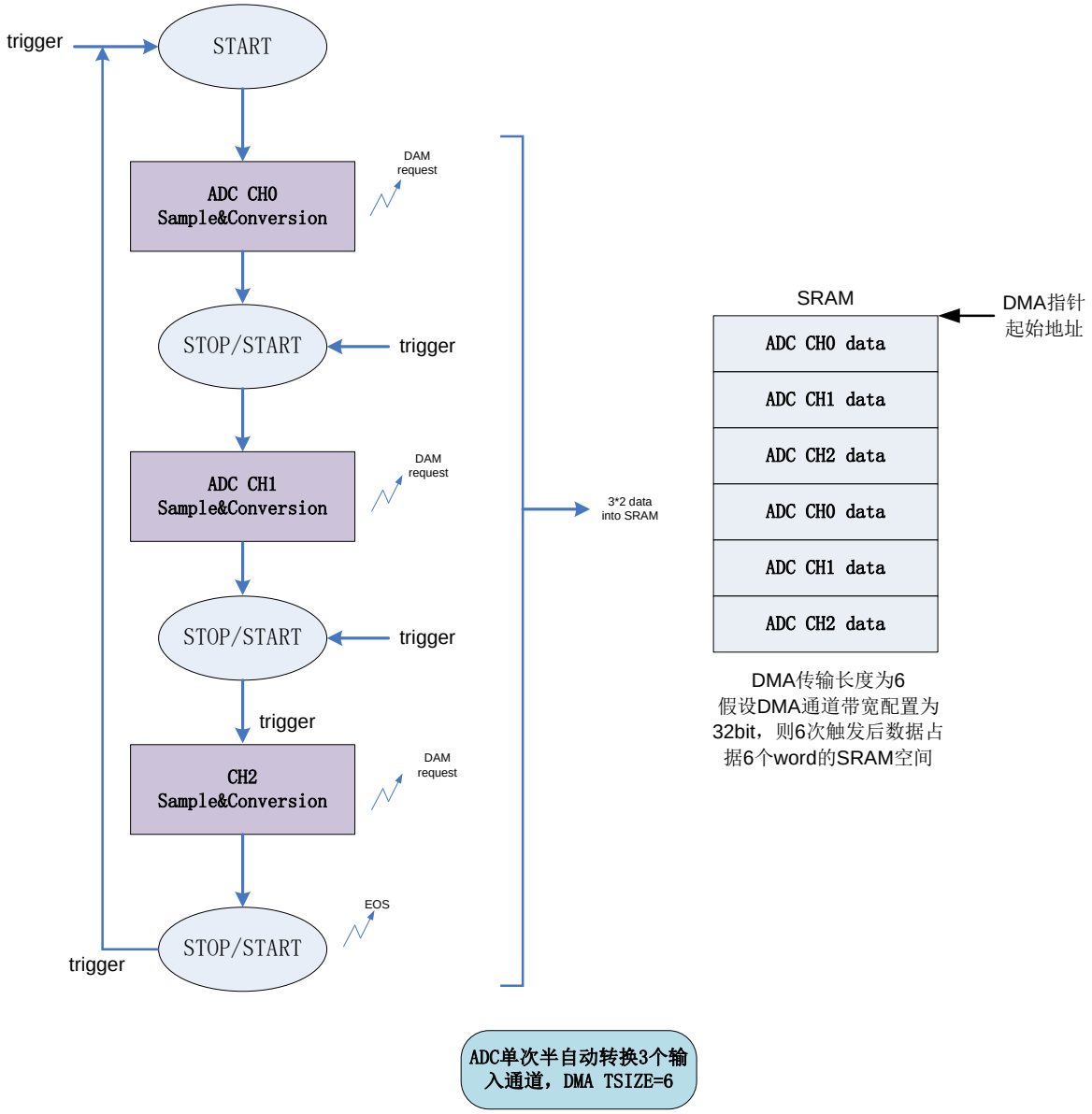


Figure 33-12 ADC single semi-automatic trigger+DMA

Use case 4：

Set ADC to fully automatic mode (SEMI=0), enable three ADC input channels (ADC_IN0/1/2), set DMA TSIZE to 6, DMA pointer to RAM address x, and set DMA to cyclic mode (CIRC=1). After the first trigger, the ADC performs three sampling conversions and DMA carries the data three times to the x/x+1/x+2 address of RAM. After the second trigger, the ADC performs three times of sampling and DMA carries the data three times to the x+3/x+4/x+5 address of RAM. After the third trigger, the ADC samples three times and the DMA carries three pieces of data to the x/x+1/x+2 address of RAM and repeats until the software shuts down the DMA.

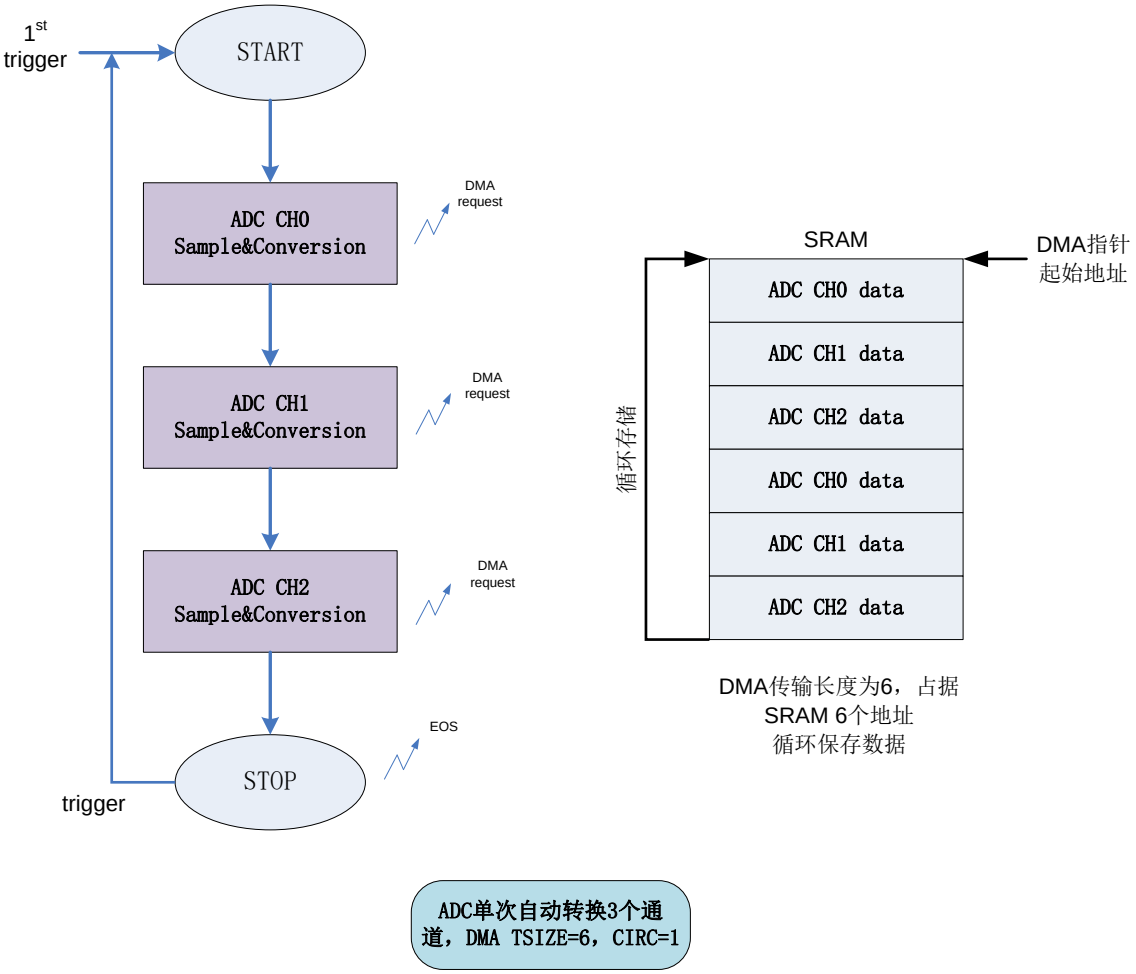


Figure 33-13 ADC automatic trigger+DMA loop mode

Use case 5：

Set ADC to continuous mode (CONT=1), enable three ADC input channels (ADC_IN0/1/2), set DMA TSIZE to six, DMA pointer to RAM address x, and set DMA to cyclic mode (CIRC=1). After the first trigger, ADC continuously samples and converts the enabling channel without interruption, and DMA continuously carries data to the x/x+1/x+2/x+3/x+4/x+5 address of RAM; The cycle continues until the software shuts down DMA.

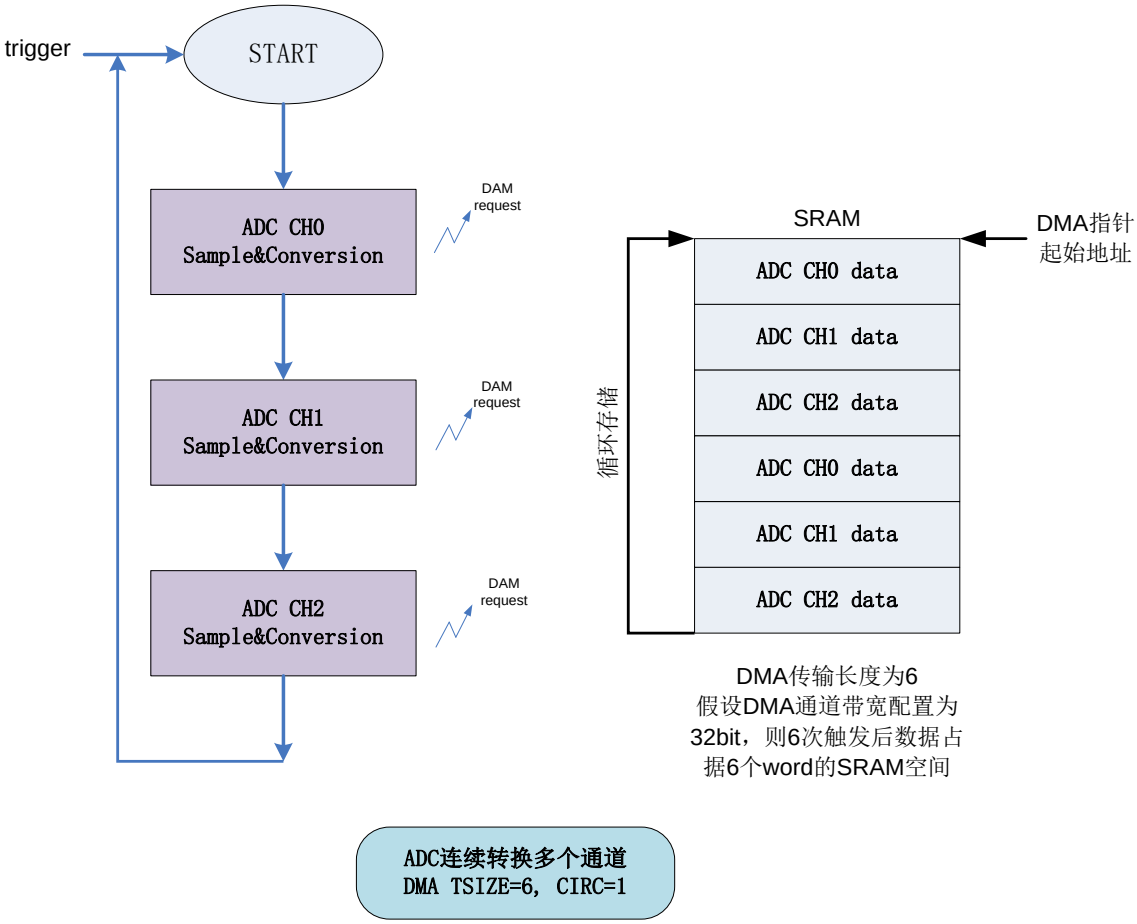


Figure 33-14 ADC continuous mode+DMA loop mode

33.5.10 Analog window watchdog（AWD）

The AWD function is used to monitor whether the input signal level of an analog input channel or all input channels is within the amplitude range set by the register. When the ADC conversion value is higher than AWD_HT or lower than AWD_LT, the interrupt flag register will be set. The flag register is cleared by writing 1 by software.

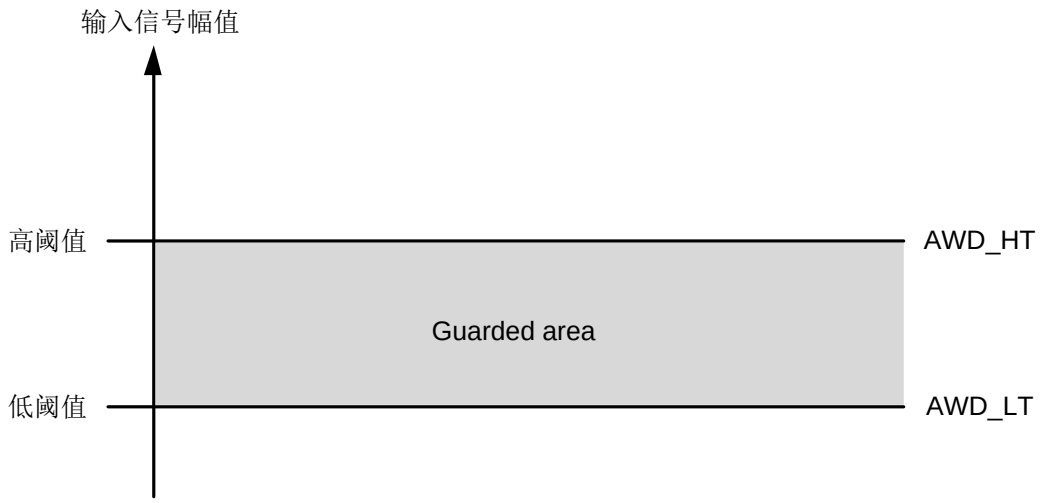


Figure 33-15 ADC simulated watchdog threshold signal

The analog window watchdog function is enabled through the AWDEN register, and single-channel monitoring or all-channel monitoring is configured through the AWDSC register.

33.5.11 ADC Calibration

ADC supports offset self-calibration，It is recommended to process a calibration operation first after the chip is powered on to obtain better accuracy.

Offset calibration involves two parts：Comparator calibration and channel (single-ended) calibration are used to eliminate the comparator offset and sampling channel offset, respectively. Both calibrations are controlled and enabled by registers respectively, and it is recommended that the user perform comparator calibration first and channel (single-ended) calibration later (if necessary).

When CALSEL=0, the software sets CALEN to start comparator calibration, the hardware automatically zeros out the CALEN register after the ADC self-calibration operation is complete, and the EOCAL interrupt flag is set after the calibration operation is complete. If the interrupt enables EOCALIE to be 1, the CPU is notified of the interrupt.

Note: After the ADC is reset, it is recommended to initiate a calibration operation before the first conversion. After the calibration is complete, the ADC can be re-enabled to perform the conversion without the need for another calibration. Therefore, you are advised to perform calibration after the chip is powered on and reset. Calibration is also recommended for higher accuracy when the ADC operating environment (temperature, voltage) changes significantly.

After the calibration operation, the calibration parameters are saved in the internal register of the ADC. The content of the calibration register will not be cleared by the reset of the ADC module or

the reset of the ADC controller. Only the power-on reset will clear the calibration register.

33.6 Low power mode

In low power mode, the chip automatically disables all high-speed clock sources, so ADC can not work.

33.7 Register

Module start address：0x4001_AC00

Offset address	Name	Symbol
0x00	ADC Interrupt and Status Register	ADC_ISR
0x04	ADC Interrupt Enable Register	ADC_IER
0x08	ADC Control Register1	ADC_CR1
0x0C	ADC Calibration Register	ADC_CALR
0x10	ADC Config Register	ADC_CFGR
0x14	ADC Sampling Time Register	ADC_SMTR
0x18	ADC Channel Enable Register	ADC_CHER
0x1C	ADC Data Register	ADC_DR
0x20	ADC Analog Watchdog Threshold Register	ADC_HLTR
0x24	ADC Calibrating Data register	ADC_OSER

33.7.1 ADC Interrupt and Status Register (ADC_ISR)

NAME	ADC_ISR							
Offset	0x00							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	AWD_AH	AWD_UL	EOCAL	BUSY	OVR	EOS	EOC
access	U-0	R/W/Dy-0	R/W/Dy-0	R/W/Dy-0	R-0	R/W/Dy-0	R/W/Dy-0	R/W/Dy-0

Bit	name	functional description
31:7	--	RFU: Reserved, read as 0
6	AWD_AH	Analog Watchdog Above High Threshold Flag
5	AWD_UL	When the sampled value is higher than AWD_HT, hardware set, and write 1 to clear by software
4	EOCAL	End Of Calibration，Hardware set, software write 1 clear zero 1：End of calibration process 0：No calibration process
3	BUSY	ADC Busy Flag, only read 1:ADC is in the process of calibration, sampling or conversion 0:ADC is idle
2	OVR	Over Run Flag, hardware set, and write 1 to clear by software When the last conversion result in the ADC_DATA register has not been read, and the new conversion result comes again, the hardware sets the OVR flag. 0:No data conflict

Bit	name	functional description
		1:Data conflict
1	EOS	End Of Sequence Flag After all the enabled channels are converted, EOS is set, and the software writes 1 to clear it.
0	EOC	End Of Conversion Flag After the conversion of each channel is completed, EOC is set, and the software writes 1 to clear it.

33.7.2 ADC Interrupt Enable Register (ADC_IER)

NAME	ADC_IER							
Offset	0x04							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-	AWD_AHIE	AWD_ULIE	EOCALIE	-	OVRIE	EOSIE	EOCIE
access	U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0

Bit	name	functional description
31:7	--	RFU: Reserved, read as 0
6	AWD_AHIE	Analog Watchdog Above High Threshold Interrupt Enable, 1 is valid
5	AWD_ULIE	Analog Watchdog Under Low Threshold Interrupt Enable, 1 is valid
4	EOCALIE	End Of Calibration Interrupt Enable Register 0:Disable EOCAL interrupt 1:Enable EOCAL interrupt
3	--	RFU: Reserved, read as 0
2	OVRIE	Over Run Interrupt Enable Register 0: Disable data conflict interrupt 1: Enable data conflict interrupt
1	EOSIE	End Of Sequence Interrupt Enable Register 0: Disable EOS interrupt 1: Enable EOS interrupt
0	EOCIE	End Of Conversion Interrupt Enable Register 0: Disable EOC interrupt 1: Enable EOC interrupt

33.7.3 ADC Control Register (ADC_CR1)

NAME	ADC_CR1							
Offset	0x08							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24

name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-						SWTRIG	ADEN
access	U-0						R/W-0	R/W-0

Bit	name	functional description
31:2	--	RFU: Reserved, read as 0
1	SWTRIG	ADC start conversion register(software trigger), software write 1 to start, hardware automatically clears.
0	ADEN	ADC Enable Register Set ADEN before starting the conversion. 0: Disable ADC 1: Enable ADC

33.7.4 ADC Calibration Control Register（ADC_CALR）

NAME	ADC_CALR							
Offset	0x0C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	OSCAL_CYCLE							
access	R/W-0000 0111							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-			OFFL_EN				
access	U-0			R/W-0				
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	VCM_CTRL	VCM_MODE	CMPRDY_DELAY		CMP_MODE	CKDIG_DELAY		CALEN
access	R/W-0	R/W-0	R/W-00		R/W-0	R/W-00		R/W-0

bit	name	functional description
31:24	-	RFU: **Reserved, read as 0**
23:16	OSCAL_CYCLE	Offset cycle number of calibration config
15:13	-	RFU: **Reserved, read as 0**
12	OFFL_EN	offline calibration enable 0：automatically calibration 1：offline calibration
11:8	-	RFU: **Reserved, read as 0**
7	VCM_CTRL	Common mode current configuration

bit	name	functional description
6	VCM_MODE	VCM Control Mode 0：normal 1：long pass
5:4	CMPRDY_DELAY	Comparator Delay Control 00：minimum delay 11：maximum delay
3	CMP_MODE	Comparator Mode Selection 0：normal mode 1：fast mode
2:1	CKDIG_DELAY	Clock Of Digital Delay Selection 00：minimum delay 11：maximum delay
0	CALEN	Offset Calibration Enable Software write 1 Start the calibration cycle, and automatically clear and set the EOCAL register after the calibration. Offset calibration can be divided into the comparator and channel calibration, choose by CALSEL register calibration project. Both calibrations are initiated by the CALEN register.

33.7.5 ADC Config Register (ADC_CFGR)

NAME	ADC_CFGR							
Offset	0x10							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-		AWDCH				AWDSC	AWDEN
access	U-0		R/W-0000				R/W-0	R/W-0
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	OVSS				OVSR			OVSEN
access	R/W-0000				R/W-000			R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-	IOTRFEN	TRGCFG		SEMI	WAIT	CONT	OVRM
access	U-0	R/W-0	R/W-00		R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	EXTS				-	SCANDIR	DMACFG	DMAEN
access	R/W-0000				U-0	R/W-0	R/W-0	R/W-0

Bit	Name	Functional descripition
31:30	--	RFU: **Reserved, read as 0**
29:26	AWDCH	Analog Watchdog Channel Select, only valid when AWDSC=1 0000: AWD monitors ADC_IN0 0001: AWD monitors ADC_IN1 0010: AWD monitors ADC_IN2 0011: AWD monitors ADC_IN3 0100: AWD monitors ADC_IN4 0101: AWD monitors ADC_IN5 0110: AWD monitors ADC_IN6 0111: AWD monitors ADC_IN7 1000: AWD monitors ADC_IN8 1001: AWD monitors ADC_IN9

Bit	Name	Functional description
		1010: AWD monitors ADC_IN10 1011: AWD monitors ADC_IN11 Others: reserved
25	AWDSC	Analog Watchdog Single Channel or Full Channel Select 0: AWD monitors all enabled external input channels 1: AWD monitors a single channel specified by AWDCH
24	AWDEN	Analog Watchdog Enable 0: Disable AWD 1: Enable AWD AWD can only be enabled when START=0
23:20	OVSS	Oversampling Shift Control Register 0000: Does not shift 0001: Shift 1bit to the right 0010: Shift 2bit to the right 0011: Shift 3bit to the right 0100: Shift 4bit to the right 0101: Shift 5bit to the right 0110: Shift 6bit to the right 0111: Shift 7bit to the right 1000: Shift 8bit to the right Others: RFU
19:17	OVSR	Oversampling Ratio Control 000: 2x 001: 4x 010: 8x 011: 16x 100: 32x 101: 64x 110: 128x 111: 256x
16	OVSEN	Oversampling Enable 0: Disable oversampling 1: Enable oversampling
15	--	RFU: Reserved, read as 0
14	IOTRFEN	GPIO Trigger Filter Enable 0: Disable digital filter 1: Enable digital filter
13:12	TRGCFG	Trigger signal enable and polarity selection (Trigger Config) 00: Disable trigger 01: Rising edge trigger 10: Falling edge trigger 11: Trigger on both rising and falling edges
11	SEMI	Single conversion semi-automatic mode, only valid in single conversion (CONT=0), see "Conversion Mode" chapter 0: Automatic mode 1: Semi-automatic mode
10	WAIT	Wait mode control 0: No waiting, if the last converted data is not read in time, Overrun may appear 1: Waiting mode, before the last conversion data is read, the next conversion will not be started
9	CONT	Continuous mode enable 0: Single conversion 1: Continuous conversion
8	OVRM	Overrun mode control

Bit	Name	Functional descripition
		0: When an overrun occurs, keep the last data and discard the converted value 1: When an overrun occurs, overwrite the last data
7:4	EXTS	External trigger select 0000: PA8 0001: PB9 0010: RFU 0011: ATIM_TRGO 0100: GPTIM0_TRGO 0101: GPTIM1_TRGO 0110: RFU 0111: RTC_TRGO 1000: BSTIM_TRGO 1001: RFU 1010: COMP1_TRGO 1011: COMP2_TRGO Others: RFU
3	--	RFU: **Reserved, read as 0**
2	SCANDIR	Channel Scan Direction Control (A total of 16 channels, actually only the enabled channels will be sampled) 0: Forward scan,ADC_IN0->ADC_IN11->REF->TS->OPA1->OPA2 1: Reverse scan,OPA2->OPA1->TS->REF->ADC_IN11->ADC_IN0
1	DMACFG	DMA mode control (DMA Config) 0: Single mode 1: Loop mode
0	DMAEN	DMA Enable 0: Disable DMA 1: Enable DMA

33.7.6 ADC Sampling Time Register (ADC_SMTR)

NAME	ADC_SMTR							
Offset	0x14							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				-			
access	U-0				U-0			
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	SMTS2				SMTS1			
access	R/W-0000				R/W-0000			

Bit	name	functional description
31:12	--	RFU: **Reserved, read as 0**
11:8	--	RFU: **Reserved, read as 0**

<table>
<tr><th>Bit</th><th>name</th><th>functional description</th></tr>
<tr><td>7:4</td><td>SMTS2</td><td>Internal channel sampling time control (*ADC operating clock cycle), used to control the sampling time of VREF, TS, VREFP, VREFN internal channels and external channels ADC_IN8~11

<table><tr><th>SMTSx</th><th>Sampling cycles</th></tr><tr><td>0000</td><td>2</td></tr><tr><td>0001</td><td>4</td></tr><tr><td>0010</td><td>8</td></tr><tr><td>0011</td><td>12</td></tr><tr><td>0100</td><td>16</td></tr><tr><td>0101</td><td>32</td></tr><tr><td>0110</td><td>64</td></tr><tr><td>0111</td><td>80</td></tr><tr><td>1000</td><td>96</td></tr><tr><td>1001</td><td>128</td></tr><tr><td>1010</td><td>160</td></tr><tr><td>1011</td><td>192</td></tr><tr><td>1100</td><td>256</td></tr><tr><td>1101</td><td>320</td></tr><tr><td>1110</td><td>384</td></tr><tr><td>1111</td><td>512</td></tr></table></td></tr>
<tr><td>3:0</td><td>SMTS1</td><td>External channel sampling time control (*ADC operating clock period), used to set the sampling time of ADC_IN0 to 7 external channels
<table><tr><th>SMTSx</th><th>Sampling cycles</th></tr><tr><td>0000</td><td>2</td></tr><tr><td>0001</td><td>4</td></tr><tr><td>0010</td><td>8</td></tr><tr><td>0011</td><td>12</td></tr><tr><td>0100</td><td>16</td></tr><tr><td>0101</td><td>32</td></tr><tr><td>0110</td><td>64</td></tr><tr><td>0111</td><td>80</td></tr><tr><td>1000</td><td>96</td></tr><tr><td>1001</td><td>128</td></tr><tr><td>1010</td><td>160</td></tr><tr><td>1011</td><td>192</td></tr><tr><td>1100</td><td>256</td></tr><tr><td>1101</td><td>320</td></tr><tr><td>1110</td><td>384</td></tr><tr><td>1111</td><td>512</td></tr></table></td></tr>
</table>

33.7.7 ADC Channel Enable Register (ADC_CHER)

NAME	ADC_CHER							
Offset	0x18							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-				VREFP	VREFN	REFCH	TSCH
access	U-0				R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				ECH11	ECH10	ECH9	ECH8
access	U-0				R/W-0	R/W-0	R/W-0	R/W-0
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ECH7	ECH6	ECH5	ECH4	ECH3	ECH2	ECH1	ECH0
access	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

Bit	name	functional description
31:20	--	RFU: **Reserved, read as 0**
19	VREFP	ADC reference voltage positive end measurement channel,write 1 to enable
18	VREFN	ADC reference voltage negative end measurement channel, write 1 enable
17	REFCH	Internal reference voltage measurement channel, write 1 to enable
16	TSCH	Temperature sensor measurement channel, write 1 to enable
15:12	--	RFU: **Reserved, read as 0**
11	ECH11	ADC_IN0~11 measurement channel,write 1 to enable, Only the single-end mode is supported
10	ECH10	
9	ECH9	
8	ECH8	
7	ECH7	
6	ECH6	
5	ECH5	
4	ECH4	
3	ECH3	
2	ECH2	
1	ECH1	
0	ECH0	

33.7.8 ADC Data Register(ADC_DR)

NAME	ADC_DR							
Offset	0x1C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

name	DATA[15:8]							
access	R-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DATA[7:0]							
access	R-0							

Bit	name	functional description
31:16	--	RFU: **Reserved, read as 0**
15:0	DATA	ADC conversion data When oversampling averaging is not enabled, the result is 12bit lower; when oversampling averaging is enabled, the result is 12~16bit

33.7.9 ADC Analog Watchdog Threshold Register (ADC_HLTR)

NAME	ADC_HLTR							
Offset	0x20							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-				AWD_HT[11:8]			
access	U-0				R/W-0			
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	AWD_HT[7:0]							
access	R/W-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-				AWD_LT[11:8]			
access	U-0				R/W-0			
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	AWD_LT[7:0]							
access	R/W-0							

Bit	name	functional description
31:28	--	RFU: **Reserved, read as 0**
27:16	AWD_HT	Analog Watchdog High Threshold
15:12	--	RFU: **Reserved, read as 0**
11:0	AWD_LT	Analog Watchdog Low Threshold

34 General-Purpose I/Os

34.1 Introduction

Main feature of I/Os ports:

- GPIO digital inputs with Schmitt characteristics
- Some GPIO inputs support analog filtering
- Some GPIO inputs support digital filtering
- GPIOs can be configured as pull-up outputs and open-drain outputs
- Keep configuration mode in Sleep/DeepSleep mode

34.2 GPIO Functional Description

FM33LE0xxA mainly has two types of GPIO pins, most of which support input and output, digital peripheral functions, analog peripheral channels, controllable pull-up resistors, and controllable open-drain output functions; the true open-drain pins are only driven by NMOS, and without PMOS, they cannot drive logic high externally.

34.2.1 GPIO，Input and Output Enable，Controlled Pull-Up Resister，Controlled Open-Drain Output

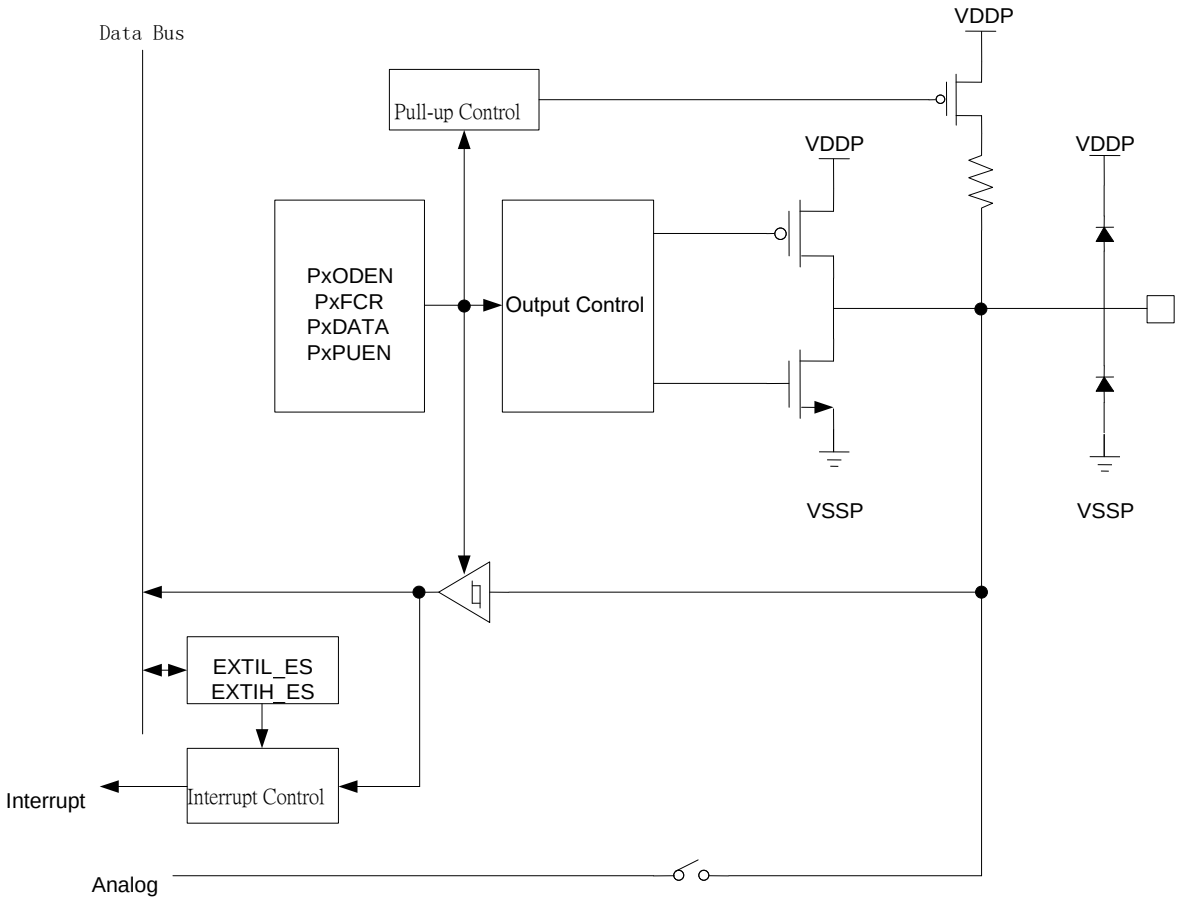


Figure 34-1 GPIO block diagram

Note：*The PA11 and PA12 do not have internal pull-up resistors, so if pin pull-ups are required, please arrange them at board level.*

The control logic is defined as follow：

Registers					PAD Interface		
FCR	**INEN**	**ODEN**	**PUEN**	**DATA**	**INPUT_EN**	**OUTPUT_EN**	**PUEN**
00	0	x	0/1	x	0	0	0/1
	1				1		
01	x	0	0/1	x	0	1	0/1
	x	1		0	0	1	
				1	0	0	
10	x	x	0/1	Peripheral input function	1	0	0/1

	x	0		Peripheral push-pull output function	0	1	
	x	1		Peripheral open-drain output 0	0	1	
				Peripheral open-drain output 1	0	0	
11	x	x	x	x	0	0	0

Table 34-1 GPIO function logic definition

34.2.2 False Open-Drain Output，No Pull-up Resistor

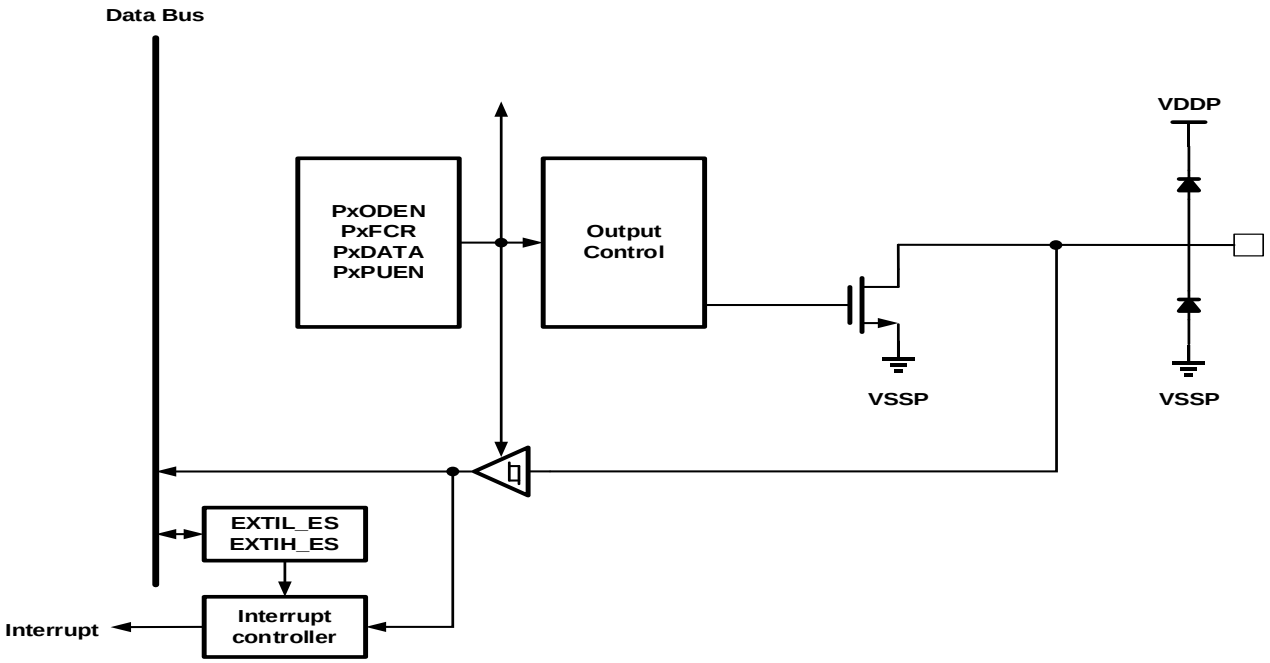


Figure 34-2 PA11&PA12 block diagram

Note：*The PA11 and PA12 do not have internal pull-up resistors, so if pin pull-ups are required, please arrange them at board level.*

False open-drain output can not external connection of signals above the supply voltage.

34.3 IO Function Definition

Most of the chip pins are digital-analog mixed IO, and each general-purpose GPIO has 4bit control registers: FCR[1:0], PUEN, ODEN, where FCR is used to select IO pin function, defined as follows:

FCR: Function Control Register	PAD function
00	GPIO input
01	GPIO output
10	Digital Function (digital peripheral function)
11	Analog

Table 34-2 FCR definition

34.3.1 GPIO Input

When a GPIO is configured for input function and the corresponding input enable register is set：

- The output drive buffer is turned off
- Schmitt trigger is enabled
- Pull-up resistors are enabled or disabled by the GPIOx_PUEN register
- GPIOx_DIN register directly responds to the level status on the IO

34.3.2 GPIO Output

When a GPIO is configured as an output function and the corresponding output enable register is set：

- Output drive buffer enable
  - Open-drain output mode (GPIOx_ODEN=1): IO drive low at output 0, IO drive buffer off at output 1
  - Push-pull output mode (GPIOx_ODEN=0): IO drive low when output 0, IO drive high when output 1
- The pull-up resistor is controlled by the GPIOx_PUEN register to enable or disable
- Software reads the GPIOx_DIN register to get the level state of IO
- Software reads the GPIOx_DO register to get the last written value

34.3.3 Digital Peripheral Functions

When a GPIO is configured as an output function and the corresponding output enable register is

set：

- Output drive buffer enable
  - Open-drain output mode (GPIOx_ODEN=1): IO drive low at output 0, IO drive buffer off at output 1
  - Push-pull output mode (GPIOx_ODEN=0): IO drive low when output 0, IO drive high when output 1
- The pull-up resistor is controlled by the GPIOx_PUEN register to enable or disable
- Software reads the GPIOx_DO register to get the last written value
- Software reads GPx_DIN register to get the level state of IO

Some of the pins support multiple digital peripheral functions, then additional control registers (ADT_AFSELX) are required to distinguish them.

The pins that support multiple digital peripheral functions are：

GPIO	Digital Feature1 PxDFS[x]=0	Digital Feature2 PxDFS[x]=1	Additional AFSEL
PA0	UART4_RX		PADFS[0]
PA1	UART4_TX		PADFS[1]
PA2	UART0_RX	LPUART0_RX	PADFS[2]
PA3	UART0_TX	LPUART0_TX	PADFS[3]
PA4	GPT1_CH3		PADFS[4]
PA5	GPT1_CH4		PADFS[5]
PA6	I2CSMB_SCL		PADFS[6]
PA7	I2CSMB_SDA		PADFS[7]
PA8	LPT32_CH1	I2CSMB_ALERT	PADFS[8]
PA9	LPT32_CH2	ATIM_CH4	PADFS[9]
PA10	LPT32_ETR	ATIM_CH3	PADFS[10]
PA11	SCL		PADFS[11]
PA12	SDA		PADFS[12]
PA13	UART0_RX	LPUART0_RX	PADFS[13]
PA14	UART0_TX	LPUART0_TX	PADFS[14]
PA15			
PB0	UART2_RX		PBDFS[0]
PB1	UART2_TX		PBDFS[1]
PB2	UART4_RX	ATIM_CH1N	PBDFS[2]
PB3	UART4_TX	ATIM_CH2N	PBDFS[3]
PB4	ATIM_CH1		PBDFS[4]
PB5	ATIM_CH2		PBDFS[5]
PB6	ATIM_CH3		PBDFS[6]

GPIO	Digital Feature1 PxDFS[x]=0	Digital Feature2 PxDFS[x]=1	Additional AFSEL
PB7	ATIM_CH4		PBDFS[7]
PB8	SPI1_SSN	ATIM_CH3N	PBDFS[8]
PB9	SPI1_SCK	GPT0_ETR	PBDFS[9]
PB10	SPI1_MISO	GPT0_CH1	PBDFS[10]
PB11	SPI1_MOSI	GPT0_CH2	PBDFS[11]
PB12	FOUT1	ATIM_ETR	PBDFS[12]
PB13	UART1_RX	LPUART1_RX	PBDFS[13]
PB14	UART1_TX	LPUART1_TX	PBDFS[14]
PB15			
PC0	GPT1_CH1	I2CSMB_SCL	PCDFS[0]
PC1	GPT1_CH2	I2CSMB_SDA	PCDFS[1]
PC2	UART1_RX	LPUART1_RX	PCDFS[2]
PC3	UART1_TX	LPUART1_TX	PCDFS[3]
PC4	UART5_RX	SDA	PCDFS[4]
PC5	UART5_TX	SCL	PCDFS[5]
PC6	GPT1_ETR	I2CSMB_ALERT	PCDFS[6]
PC7	SPI2_SSN		PCDFS[7]
PC8	SPI2_SCK		PCDFS[8]
PC9	SPI2_MISO		PCDFS[9]
PC10	SPI2_MOSI		PCDFS[10]
PC11	U7816_CLK	GPT0_CH3	PCDFS[11]
PC12	U7816_IO	GPT0_CH4	PCDFS[12]
PC13			
PC14			
PC15			
PD0	UART5_RX		PDDFS[0]
PD1	UART5_TX		PDDFS[1]
PD2	SPI1_SSN		PDDFS[2]
PD3	SPI1_SCK		PDDFS[3]
PD4	SPI1_MISO		PDDFS[4]
PD5	SPI1_MOSI		PDDFS[5]
PD6	ATIM_BRK2		PDDFS[6]
PD7	SWCLK	UART2_RX	PDDFS[7]
PD8	SWIO	UART2_TX	PDDFS[8]
PD9			PDDFS[9]
PD10			PDDFS[10]
PD11	FOUT0	ATIM_BKR1	PDDFS[11]

Table 34-3 Digital peripheral function selection table

34.3.4 Analog Function

When a GPIO is configured to analog function：

- Output buffer off
- Digital input function off
- Pull-up resistor off
- PxDIN returns 0
- IO analog channel is connected to a specific analog peripheral
- If an IO is connected to multiple analog peripherals at the same time, only one of the multiple analog peripherals can be enabled at the same time

Each GPIO has two analogue channels, a resistive channel and a switching channel. The resistor channel cannot be switched off, any signal on the pin is transmitted to the chip internally and the switch channel can be enabled or disabled by the ANAEN signal.

The vast majority of the chip's analogue signals, such as COM/SEG, are connected to the resistive channels of the GPIO and their output levels are switched off or enabled internally by the DISP module. Other analogue signals go partly on resistive channels and partly on switch channels, Refering the following table.

Analog Function	PAD Channel	Description
COM	Resistor	Connect with analog module directly
SEG	Resistor	
XTHF	Resistor	
XTLF	Resistor	
ADC_INx	Resistor	
VCINx	Switch	When FCR=11, enable switch
VDISPx	Switch	When FCR=11 and ANEN=1, enable switch

34.3.5 Analog Function Using External Crystal Pins

FM33LC0XX supports external 32768Hz crystals and 4~24MHz high frequency crystals.

PC2 and PC3 are GPIO by default, and can be used as XTHFIN and XTHFOUT external high frequency crystals after configured as analog function.

PD9 and PD10 are by default analogue functions with an external 32768Hz crystal; they can also be configured for GPIO use with the XTLF function switched off.

34.4 NRST Pin

The NRST pin is used to generate a chip reset. If the chip is in low power mode, a valid NRST will also cause the chip to exit low power mode.

The NRST pin input has a digital filter, and the filter works with the LPOSC clock. When the chip turns off LPOSC in sleep mode, a low NRST input will force LPOSC to be enabled and reset the chip if the filtering is valid, or automatically turn off LPOSC after the filtering is finished if the filtering is invalid; therefore the burr of the NRST pin in sleep mode will not affect the chip power consumption.

The digital filtering uses a continuous 3 beat sampling of the LPOSC and an asynchronous reset of the input signal high level, i.e. to ensure that the NRST signal remains stable low for 2~3 LPOSC cycles. As the input signal is asynchronous to LPOSC, the actual effective filter length may be between 2~3 LPOSC cycles, i.e. a typical range of 60~90us, which may vary by around 10% with temperature changes.

34.5 WKUPx Pins

The FM33LC0XX has eight WKUP pins that can wake up the chip from Sleep/DeepSleep mode, even if the on-chip oscillators are all stopped.

WKUPx pin input rising edge or falling edge (software configuration) can wake up the chip from Sleep mode. In order to enable this function, the corresponding pin needs to be configured for GPIO input function and the corresponding GPIO_PINWKEN.EN set. Note that the PAD has an internal pull-up resistor, which must be turned off if configured for wake-up on rising edge.

Each WKUP-enabled IO comes with an on-chip analog filter of approximately 100ns, which filters out burrs on the input signal to avoid false triggering.

In Sleep/DeepSleep mode, any pulse greater than 100ns on the enabled WKUPx pin will trigger the chip to wake up.

The eight WKUPx circuits are completely independent in structure, the diagram below shows the block diagram of the structure of one of the WKUP functions

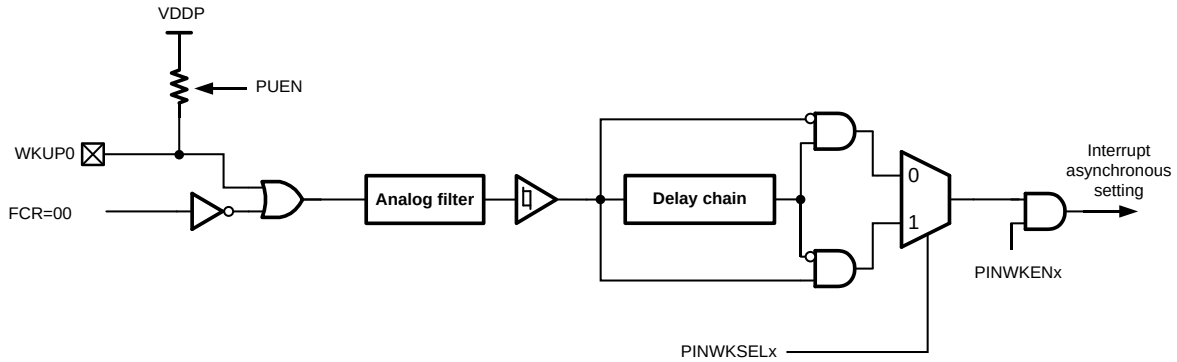


Figure 34-3 WKUPx function block diagram

The WKUPx function requires attention to the initial state of the external pin inputs when used. When enabling WKUP, a false wake-up event may result due to the initial state, which the software should take care to identify and handle.

When using the WKUP function, user must configure the FCR register of the corresponding pin to 00 (GPIO input), set the wake-up edge (GPIO_PINWKEN.SEL) and enable the GPIO_PINWKEN.EN register as required. When a wake-up event is generated on one of the WKUPx pins, the corresponding bit in the wake-up source flag register inside the PMU module will be set automatically.

34.6 External Pin Interrupt (EXTI)

34.6.1 Function Description

The 4 groups of GPIOs (A~D) of FM33LE0xxA can generate up to 16 EXTI interrupts, each group of GPIOs can generate 4 EXTI interrupt flags respectively, and finally all EXTI interrupts are aggregated to the #46 entry of NVIC.

The interrupt flags and pins correspond to the following table.

GPIO	EXTI input selection	EXTI
PA0~PA3	EXTI_ASEL[1:0]	EXTI[0]
PA4~PA7	EXTI_ASEL[3:2]	EXTI[1]
PA8~PA11	EXTI_ASEL[5:4]	EXTI[2]
PA12~PA15	EXTI_ASEL[7:6]	EXTI[3]
PB0~PB3	EXTI_BSEL[1:0]	EXTI[4]
PB4~PB7	EXTI_BSEL[3:2]	EXTI[5]
PB8~PB11	EXTI_BSEL[5:4]	EXTI[6]
PB12~PB15	EXTI_BSEL[7:6]	EXTI[7]
PC0~PC3	EXTI_CSEL[1:0]	EXTI[8]
PC4~PC7	EXTI_CSEL[3:2]	EXTI[9]
PC8~PC11	EXTI_CSEL[5:4]	EXTI[10]
PC12	-	EXTI[11]

PD0~PD3	EXTI_DSEL[1:0]	EXTI[12]
PD4~PD7	EXTI_DSEL[3:2]	EXTI[13]
PD8~PD11	EXTI_DSEL[5:4]	EXTI[14]
PD12	-	EXTI[15]

Table 34-4 External pin interrupt configuration

The GPIO_EXTISEL register is used to select an IO to access the EXTI channel, and the EXTI module can configure whether to digitally filter the input signal.

The digital filtering is implemented by the IO sampling clock to continuously sample input and get same level three times before it is considered a valid level input, as shown in the figure below.

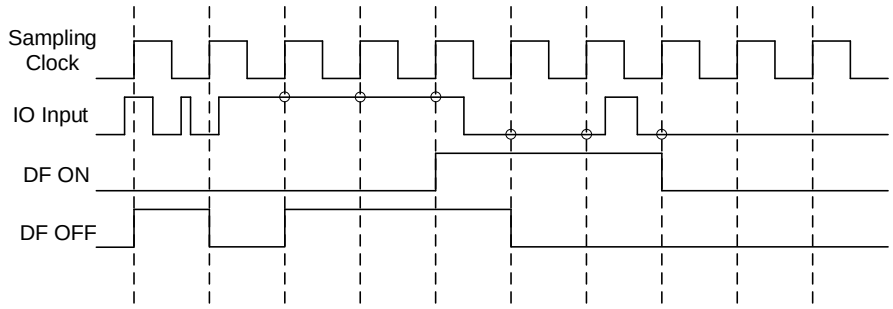


Figure 34-4 Pin input digital filtering

The software can select the sampling clock for digital filtering as APBCLK or LSCLK.

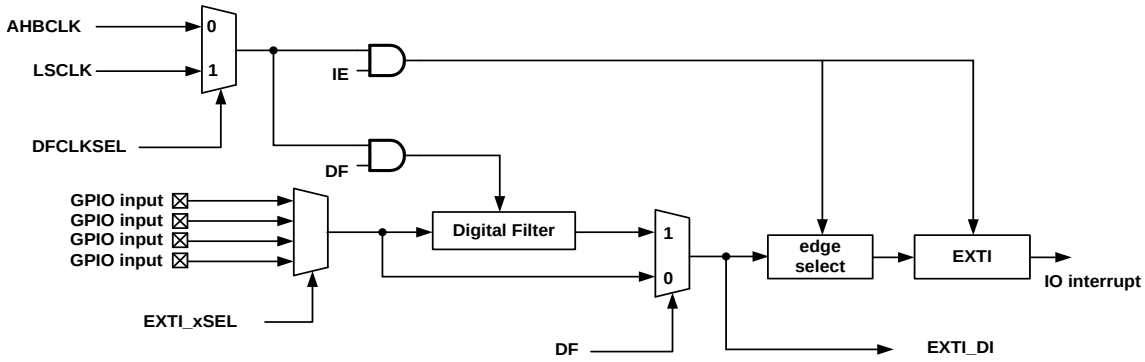


Figure 34-5 EXIT signal input schematic

Users should enable or disable the digital filtering function according to the pin function needs. After enabling the digital filtering, different sampling delays will be introduced to the IO input signal depending on the AHBCLK frequency. The output signal after digital filtering can also be read by software in GPIO_EXTIDI register.

EXTI can also configure the effective edge of the input signal to support rising edge, falling edge, rising falling edge triggered interrupt, or disable EXTI interrupt triggering, as configured by the GPIO_EXTIEDS register.

34.6.2 Application guidelines

To activate the EXTI interrupt wake-up function in Sleep/DeepSleep mode, the following steps are recommended:

- Turn off all EXTI enable
- Configure the SLP_ENEXTI bit in the SYSCLKSEL register to 1 and select LSCLK for EXTI sampling
- Turn on or off the EXTI digital filtering enable as needed
- Configure the corresponding GPIO as input
- Configure the GPIO_EXTISEL register to select the corresponding IO
- Set OPCCR1.EXTICKE to turn on EXTI operating clock enable
- Wait at least 4 LSCLK cycles
- Configure GPIO_EXTIEDS trigger edge selection to enable the required EXTI interrupt
- Enter Sleep mode normally

All EXTIs are turned off by default after the chip is powered on, while the default pin interrupt sampling clock is the system clock AHBCLK. if the user uses the system clock to generate EXTIs, the recommended flow is as follows：

- Turn on digital filter enable (if required)
- Configure GPIO as input
- Set OPCCR1.EXTICKE to turn on EXTI operating clock enable
- Wait at least 4 LSCLK cycles
- Configure GPIO_EXTIEDS trigger edge selection to enable the required EXTI interrupt

If user wish to use a low-speed LSCLK to generate EXTI, the recommended flow is as follows

- Configure the EXTI sample clock as LSCLK
- Turn on digital filtering enable (if required)
- Configure GPIO as input
- Set OPCCR1.EXTICKE to turn on EXTI sample clock enable
- Wait at least 4 LSCLK cycles
- Configure GPIO_EXTIEDS trigger edge to enable the required EXTI interrupt

34.7 Fast GPIO Output

The FM33LE0xxA can quickly change the output data of each GPIO (bitwise operation) through the set-reset function to improve the IO output efficiency, especially the efficiency and reliability of read-modify-write operation (atomic). The method is that each GPIO group output data register has 2 sets of set-reset mapped virtual addresses. Writing 1 to a specific bit of the GPIOx_DSET register can set the bit of the corresponding data register, and writing 1 to a specific address of the GPIOx_DRST register can clear the bit of the corresponding data register.

34.8 Register

Offset	Name	Symbol
GPIOA(base address:0X40000C00)		
0x00000000	GPIOA Input Enable Register	GPIOA_INEN
0x00000004	GPIOA Pull-Up Enable Register	GPIOA_PUEN
0x00000008	GPIOA Open-Drain Enable Register	GPIOA_ODEN
0x0000000C	GPIOA Function Control Register	GPIOA_FCR
0x00000010	GPIOA Data Output Register	GPIOA_DO
0x00000014	GPIOA Data Set Register	GPIOA_DSET
0x00000018	GPIOA Data Reset Register	GPIOA_DRST
0x0000001C	GPIOA Data Input Register	GPIOA_DIN
0x00000020	GPIOA Digital Function Select	GPIOA_DFS
0x00000028	GPIOA Analog channel Enable Register	GPIOA_ANEN
GPIOB(base address:0X40000C40)		
0x00000000	GPIOB Input Enable Register	GPIOB_INEN
0x00000004	GPIOB Pull-Up Enable Register	GPIOB_PUEN
0x00000008	GPIOB Open-Drain Enable Register	GPIOB_ODEN
0x0000000C	GPIOB Function Control Register	GPIOB_FCR
0x00000010	GPIOB Data Output Register	GPIOB_DO
0x00000014	GPIOB Data Set Register	GPIOB_DSET
0x00000018	GPIOB Data Reset Register	GPIOB_DRST
0x0000001C	GPIOB Data Input Register	GPIOB_DIN
0x00000020	GPIOB Digital Function Select	GPIOB_DFS
0x00000028	GPIOB Analog channel Enable Register	GPIOB_ANEN
GPIOC(base address:0X40000C80)		
0x00000000	GPIOC Input Enable Register	GPIOC_INEN
0x00000004	GPIOC Pull-Up Enable Register	GPIOC_PUEN
0x00000008	GPIOC Open-Drain Enable Register	GPIOC_ODEN
0x0000000C	GPIOC Function Control Register	GPIOC_FCR
0x00000010	GPIOC Data Output Register	GPIOC_DO
0x00000014	GPIOC Data Set Register	GPIOC_DSET
0x00000018	GPIOC Data Reset Register	GPIOC_DRST
0x0000001C	GPIOC Data Input Register	GPIOC_DIN
0x00000020	GPIOC Digital Function Select	GPIOC_DFS
0x00000028	GPIOC Analog channel Enable Register	GPIOC_ANEN
GPIOD(base address:0X40000CC0)		
0x00000000	GPIOD Input Enable Register	GPIOD_INEN
0x00000004	GPIOD Pull-Up Enable Register	GPIOD_PUEN
0x00000008	GPIOD Open-Drain Enable Register	GPIOD_ODEN
0x0000000C	GPIOD Function Control Register	GPIOD_FCR
0x00000010	GPIOD Data Output Register	GPIOD_DO
0x00000014	GPIOD Data Set Register	GPIOD_DSET

Offset	Name	Symbol
0x00000018	GPIOD Data Reset Register	GPIOD_DRST
0x0000001C	GPIOD Data Input Register	GPIOD_DIN
0x00000020	GPIOD Digital Function Select	GPIOD_DFS
0x00000028	GPIOD Analog channel Enable Register	GPIOD_ANEN
GPIO(base address:0X40000D00)		
0x00000000	External Interrupt input Select Register	GPIO_EXTISEL
0x00000004	External Interrupt Edge Select and Enable Register	GPIO_EXTIEDS
0x00000008	External Interrupt Digital Filter Register	GPIO_EXTIDF
0x0000000C	External Interrupt and Status Register	GPIO_EXTIISR
0x00000010	External Interrupt Data Input Register	GPIO_EXTIDI
-	-	-
0x00000100	Frequency Output Select Register	GPIO_FOUTSEL
0x00000200	Wakeup Enable Register	GPIO_PINWKEN

34.8.1 GPIOx Input Enable Register (GPIOx_INEN)

NAME	GPIOx_INEN(x=A,B,C,D)							
Offset	PA,y=0 PB,y=1 PC,y=2 PD,y=3 0x00000000 + y*0x40							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	INEN[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	INEN[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	**RFU: Reserved, read as 0**
15:0	INEN	GPIO input enable control (Portx Input Enable) 0: Input disable 1: Input enable

34.8.2 GPIOx Pull-up Enable Register (GPIOx_PUEN)

NAME	GPIOx_PUEN(x=A,B,C,D)							
Offset	PA,y=0 PB,y=1 PC,y=2 PD,y=3 0x00000004 + y*0x40							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	PUEN[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	PUEN[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	**RFU: Reserved, read as 0**
15:0	PUEN	GPIO pull-up control (Portx Pull-Up Enable) 0: Pull-up disable 1: Pull-up enable

34.8.3 GPIOx Open-Drain Enable Register (GPIOx_ODEN)

NAME	GPIOx_ODEN(x=A,B,C,D)							
Offset	PA,y=0 PB,y=1 PC,y=2 PD,y=3 0x00000008 + y*0x40							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	ODEN[15:8]							

access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ODEN[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	**RFU: Reserved, read as 0**
15:0	ODEN	GPIO open-drain output enable (Portx Open-Drain Enable) 0: Open-drain output disable 1: Open-drain output enable

34.8.4 GPIOx Function Control Register (GPIO_FCR)

NAME	GPIOx_FCR(x=A,B,C,D)							
Offset	PA,y=0 PB,y=1 PC,y=2 PD,y=3 0x0000000C + y*0x40							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	Px15FCR		Px14FCR		Px13FCR		Px12FCR	
access	R/W-00		R/W-00		R/W-00		R/W-00	
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	Px11FCR		Px10FCR		Px9FCR		Px8FCR	
access	R/W-00		R/W-00		R/W-00		R/W-00	
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	Px7FCR		Px6FCR		Px5FCR		Px4FCR	
access	R/W-00		R/W-00		R/W-00		R/W-00	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	Px3FCR		Px2FCR		Px1FCR		Px0FCR	
access	R/W-00		R/W-00		R/W-00		R/W-00	

bit	name	functional description
31:30	Px15FCR	Px[15] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
29:28	Px14FCR	Px[14] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function

bit	name	functional description
27:26	Px13FCR	Px[13] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
25:24	Px12FCR	Px[12] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
23:22	Px11FCR	Px[11] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
21:20	Px10FCR	Px[10] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
19:18	Px9FCR	Px[9] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
17:16	Px8FCR	Px[8] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
15:14	Px7FCR	Px[7] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
13:12	Px6FCR	Px[6] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
11:10	Px5FCR	Px[5] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output

bit	name	functional description
		10: Digital function 11: Analog function
9:8	Px4FCR	Px[4] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
7:6	Px3FCR	Px[3] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
5:4	Px2FCR	Px[2] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function
3:2	Px1FCR	Px[1] pin function selection (Portx Function Control Register) 00：GPIO input 01：GPIO output 10：Digital function 11：Analog function
1:0	Px0FCR	Px[0] pin function selection (Portx Function Control Register) 00: GPIO input 01: GPIO output 10: Digital function 11: Analog function

34.8.5 GPIOx Data Output Register (GPIOx_DO)

NAME	GPIOx_DO(x=A,B,C,D)							
Offset	PA,y=0 PB,y=1 PC,y=2 PD,y=3 0x00000010 + y*0x40							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							

bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DO[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DO[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	**RFU: Reserved, read as 0**
15:0	DO	GPIO output data register

34.8.6 GPIOx Data Set Register (GPIOx_DSET)

NAME	GPIOx_DSET(x=A,B,C,D)							
Offset	PA,y=0 PB,y=1 PC,y=2 PD,y=3 0x00000014 + y*0x40							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DSET[15:8]							
access	W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DSET[7:0]							
access	W-0000 0000							

bit	name	functional description
31:16	-	**RFU: Reserved, read as 0**
15:0	DSET	GPIO output data set register Example: Write 0x0000_8000 to GPIOA_DSET, then PADO[15] is set and the rest of the bits remain unchanged. GPIOA _DSET/GPIOB_DSET is 16 bits; GPIOC_DSET/GPIOD_DSET is 13 bits

34.8.7 GPIOx Data Reset Register (GPIOx_DRST)

NAME	GPIOx_DRST(x=A,B,C,D)							
Offset	PA,y=0 PB,y=1 PC,y=2 PD,y=3 0x00000018 + y*0x40							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DRESET[15:8]							
access	W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DRESET[7:0]							
access	W-0000 0000							

bit	name	functional description
31:16	-	**RFU: Reserved, read as 0**
15:0	DRESET	GPIO output data reset register Example: Write 0x0000_8000 to GPIOA_DRST, then PADO[15] is cleared to zero and the rest of the bits remain unchanged GPIOA _DRESET/GPIOB_DRESET is 16 bits; GPIOC_DRESET/GPIOD_DRESET is 13 bits

34.8.8 GPIOxb Data Input Register (GPIOx_DIN)

NAME	GPIOx_DIN(x=A,B,C,D)							
Offset	PA,y=0 PB,y=1 PC,y=2 PD,y=3 0x0000001C + y*0x40							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							

bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DIN[15:8]							
access	R-xxxxxxxx							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DIN[7:0]							
access	R-xxxxxxxx							

bit	name	functional description
31:16	-	**RFU: Reserved, read as 0**
15:0	DIN	Portx input data register This register only occupies address space and has no physical implementation. Software reads this register to return the pin input signal directly, the chip does not latch the pin input.

34.8.9 GPIOx Digital Funnction Select (GPIOx_DFS)

NAME	GPIOx_DFS(x=A,B,C,D)							
Offset	PA,y=0 PB,y=1 PC,y=2 PD,y=3 0x00000020 + y*0x40							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DFS[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DFS[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU: **Reserved, read as 0**
15:0	DFS	Portx Digital Function Select For pins with multiple digital peripheral functions, the GPIOx_DFS register allows you to select which peripheral function to use. Note that the valid register locations are different for different IO groupings, please refer to Table 34-3 for detailed definitions

34.8.10 GPIOx Analog Channel Enable Register (GPIOxANEN)

NAME	GPIOx_ANEN(x=A,B,C,D)							
Offset	PA,y=0 PB,y=1 PC,y=2 PD,y=3 0x00000028 + y*0x40							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	ANEN[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ANEN[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU: **Reserved, read as 0**
15:0	ANEN	Portx Analog channel Enable 1: IO analog channel enable 0: IO analog channel disable *Note:* *PA15 SVS* *PB9 ANATST* *PD6 ANATST* *The GPIO register is valid when using these analog functions corresponding to the above IO; the other registers are invalid.*

34.8.11 External Interrupt Input Select Register (GPIO_EXTISEL)

NAME	GPIO_EXTISEL							
Offset	0x00000100							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-		DSEL					
access	U-0		R/W-00 0000					
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-		CSEL					
access	U-0		R/W-00 0000					

bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	BSEL							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	ASEL							
access	R/W-0000 0000							

bit	name	Functional description
31:30	-	RFU：未实现，读为 0
29:24	DSEL	PortD EXTI 中断输入选择 (External Interrupt PortD Select) EXTI[14]：EXTI_DSEL[5:4] – 00：PD8 01：PD9 10：PD10 11：PD11 EXTI[13]：EXTI_DSEL[3:2] – 00：PD4 01：PD5 10：PD6 11：PD7 EXTI[12]：EXTI_DSEL[1:0] – 00：PD0 01：PD1 10：PD2 11：PD3
23:22	CSEL1	PortC EXTI (External Interrupt PortC Select) EXTI[10]： xx:PC12
21:16	CSEL0	PortC EXTI (External Interrupt PortC Select) EXTI[10]：EXTI_CSEL[5:4] – 00：PC8 01：PC9 10：PC10 11：PC11 EXTI[9]：EXTI_CSEL[3:2] – 00：PC4 01：PC5 10：PC6 11：PC7 EXTI[8]：EXTI_CSEL[1:0] – 00：PC0 01：PC1 10：PC2

bit	name	Functional description
		11：PC3
15:8	BSEL	PortB EXTI (External Interrupt PortB Select) EXTI[7]：EXTI_BSEL[7:6] – 00：PB12 01：PB13 10：PB14 11：PB15 EXTI[6]：EXTI_BSEL[5:4] – 00：PB8 01：PB9 10：PB10 11：PB11 EXTI[5]：EXTI_BSEL[3:2] – 00：PB4 01：PB5 10：PB6 11：PB7 EXTI[4]：EXTI_BSEL[1:0] – 00：PB0 01：PB1 10：PB2 11：PB3
7:0	ASEL	PortA EXTI (External Interrupt PortA Select) EXTI[3]：EXTI_ASEL[7:6] – 00：PA12 01：PA13 10：PA14 11：PA15 EXTI[2]：EXTI_ASEL[5:4] – 00：PA8 01：PA9 10：PA10 11：PA11 EXTI[1]：EXTI_ASEL[3:2] – 00：PA4 01：PA5 10：PA6 11：PA7 EXTI[0]： EXTI_ASEL[1:0] – 00：PA0 01：PA1 10：PA2 11：PA3

34.8.12 External Interrupt Edge Select and Enable Register (GPIO_EXTIEDS)

NAME	GPIO_EXTIEDS							
Offset	0x00000104							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	EXTI15_EDS		EXTI14_EDS		EXTI13_EDS		EXTI12_EDS	
access	R/W-11		R/W-11		R/W-11		R/W-11	
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	EXTI11_EDS		EXTI10_EDS		EXTI9_EDS		EXTI8_EDS	
access	R/W-11		R/W-11		R/W-11		R/W-11	
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	EXTI7_EDS		EXTI6_EDS		EXTI5_EDS		EXTI4_EDS	
access	R/W-11		R/W-11		R/W-11		R/W-11	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	EXTI3_EDS		EXTI2_EDS		EXTI1_EDS		EXTI0_EDS	
access	R/W-11		R/W-11		R/W-11		R/W-11	

bit	name	Functional description
31:30	EXTI15_EDS	EXTI[15] (External Interrupt 15 Edge Select) 00：rising 01：falling 10：both 11：disable
29:28	EXTI14_EDS	EXTI[14] (External Interrupt 14 Edge Select) 00：rising 01：falling 10：both 11：disable
27:26	EXTI13_EDS	EXTI[13] (External Interrupt 13 Edge Select) 00：rising 01：falling 10：both 11：disable
25:24	EXTI12_EDS	EXTI[12] (External Interrupt 12 Edge Select) 00：rising 01：falling 10：both 11：disable
23:22	EXTI11_EDS	EXTI[11] (External Interrupt 11 Edge Select) 00：rising 01：falling

bit	name	Functional description
		10：both 11：disable
21:20	EXTI10_EDS	EXTI[10] (External Interrupt 10 Edge Select) 00：rising 01：falling 10：both 11：disable
19:18	EXTI9_EDS	EXTI[9] (External Interrupt 9 Edge Select) 00：rising 01：falling 10：both 11：disable
17:16	EXTI8_EDS	EXTI[8] (External Interrupt 8 Edge Select) 00：rising 01：falling 10：both 11：disable
15:14	EXTI7_EDS	EXTI[7] (External Interrupt 7 Edge Select) 00：rising 01：falling 10：both 11：disable
13:12	EXTI6_EDS	EXTI[6] (External Interrupt 6 Edge Select) 00：rising 01：falling 10：both 11：disable
11:10	EXTI5_EDS	EXTI[5] (External Interrupt 5 Edge Select) 00：rising 01：falling 10：both 11：disable
9:8	EXTI4_EDS	EXTI[4] (External Interrupt 4 Edge Select) 00：rising 01：falling 10：both 11：disable
7:6	EXTI3_EDS	EXTI[3] (External Interrupt 3 Edge Select) 00：rising 01：falling 10：both 11：disable
5:4	EXTI2_EDS	EXTI[2] (External Interrupt 2 Edge Select)

bit	name	Functional description
		00：rising 01：falling 10：both 11：disable
3:2	EXTI1_EDS	EXTI[1] (External Interrupt 1 Edge Select) 00：rising 01：falling 10：both 11：disable
1:0	EXTI0_EDS	EXTI1[0] (External Interrupt 0 Edge Select) 00：rising 01：falling 10：both 11：disable

34.8.13 External interrupt Digital Filter Register (GPIO_EXTIDF)

NAME	GPIO_EXTIDF							
Offset	0x00000108							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DF[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DF[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	**RFU: Reserved, read as 0**
15:0	DF	EXTI[0~15] input digital filter function enable (External Interrupt Digital Filter Enable) 0: Turn off EXTI digital filtering 1: Enable EXTI digital filtering

34.8.14 External Interrupt and Status Register (GPIO_EXTLLSR)

NAME	GPIO_EXTIISR

Offset	0x0000010C							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	IF[15:8]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	IF[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31:16	-	RFU: **Reserved, read as 0**
15:0	IF	EXTI[0~15] external pin interrupt flag register, a total of 16 pin interrupts can be generated. (External Interrupt Flags) Set by hardware, and can be cleared by software by writing 1.

34.8.15 External Interrupt Data Input Register (GPIO_EXTIDI)

NAME	GPIO_EXTIDI							
Offset	0x00000110							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	DI[15:8]							
access	R-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	DI[7:0]							
access	R-0000 0000							

bit	name	functional description
31:16	-	RFU: **Reserved, read as 0**
15:0	DI	EXTI[0~15] input signal read-only register, the software can read this register to observe the current status of EXTI's 16 input signals. (External Interrupt Data Input)

bit	name	functional description
		Note: When digital filtering is enabled, the software can read the filtered state of an IO input signal from this register.

34.8.16 Frequency Output Select Register (GPIO_FOUTSEL)

NAME	GPIO_FOUTSEL							
Offset	0x00000200							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	-							
access	U-0							
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	FOUT1SEL				FOUT0SEL			
access	R/W-0000				R/W-0000			

位号	助记符	功能描述
31:9	-	**RFU: Reserved, read as 0**
8	-	Forbid to write 1
7:4	FOUT1SEL	PB12 (Frequency Output Select for PB12) 0000：XTLF 0001：LPOSC 0010：RCHF/64 0011：LSCLK 0100：AHBCLK/64 0101：RTCTM 0110：PLLO/64 0111：RTCCLK64Hz 1000：APBCLK1/64 1001：PLLO 1010：RCMFPSC 1011：RCHF 1100：XTHF/64 1101：ADCCLK/64 1110：CLK8K 1111：COMP2O
3:0	FOUT0SEL	PD11 (Frequency Output Select for PD11) 0000：XTLF

位号	助记符	功能描述
		0001：LPOSC 0010：RCHF/64 0011：LSCLK 0100：AHBCLK/64 0101：RTCTM 0110：PLLO/64 0111：RTCCLK64Hz 1000：APBCLK1/64 1001：PLLO 1010：RCMFPSC 1011：RCHF 1100：XTHF/64 1101：COMP1O 1110：CLK8K 1111：ADC_CLK

34.8.17 Wakeup Enable Register (GPIO_PINWKEN)

NAME	GPIO_PINWKEN							
Offset	0x00000300							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	WKISEL	-						
access	R/W-0	U-0						
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	SEL[7:0]							
access	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	EN[7:0]							
access	R/W-0000 0000							

bit	name	functional description
31	WKISEL	WKUP Interrupt Entry Select 0: NMI interrupt 1: #38 entry
30:16	-	RFU: **Reserved, read as 0**
15:8	SEL	Wakeup Edge Select 1: Corresponding WKUP pin for rising edge wake-up 0: Corresponding WKUP pin for falling edge wake-up
7:0	EN	Wakeup Enable

bit	name	functional description
		1: Corresponding WKUP pin function is valid 0: Corresponding WKUP pin function is invalid PINWKEN[x] controls the enable of WKUPx pin

35 Serial wire debug (SWD)

35.1 Introduction

FM33LE0xxA chip can use the dedicated programmer provided by Fudan Microelectronics or download the user program through Bootloader. The programmer communicates with the chip through the serial wire debug (SWD) to complete the program download and perform Checksum verification of the full space contents of the Flash.

35.2 Programmer instruction

For the instruction of the programmer, please refer to the application manual, or contact Fudan Microelectronics.

36 Device signature

Each FM33LE0xxA series MCU has its own device signature, including memory capacity information and a unique device ID number.

36.1 Memory Capacity Query Register

By searching bit2 of SYSCON Register, You can obtain the Flash capacity information of the device.

NAME	SYSCON							
Offset	0x40000000							
bit	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
name	VERIFAIL	RDPROTFAIL	-					
access	R/Dy-0	R/Dy-0	U-0					
bit	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
access	U-0							
bit	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
name	-							
access	U-0							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	-					FLSCFG	-	
access	U-0					R/Dy-00	U-0	

Bit	name	functional description
31	VERIFAIL	Flash configuration information, namely LDT0 area, verification failed flag (Verification code checksum Fail) 1: A certain configuration information verification failed occurs, the error item will remain the default value 0: Verification passed
30	RDPROTFAIL	OPTBYTE/ACLOCK, namely LDT1 area, verification failed flag (Read-out Protection checksum Fail) 1: Verification failed, debug interface and ACLOCK protection related registers are set to 1 0: Verification passed
29:3	--	RFU: **Reserved, read as 0**
2	FLSCFG	Flash size configuration 0: 256KB 1: 128KB (RAM is 24KB)
1:0	--	RFU: **Reserved, read as 0**

36.2 Device UID

Device identifier provides an UID which is globally unique for each device . These bits can never be altered by the user.

The UID is 96bits in total and is stored in a special sector of Flash. This UID can be read during software runtime and is used to implement code protection or secure boot type applications.

UID access address is 0x1FFFFA10.

37 Debug Support

37.1 Introduction

The FM33LE0xxA chip is based on the ARM Cortex-M0 processor and supports the corresponding debug features. Through hardware breakpoints (breakpoints) and data watchpoints (watchpoints), the debugger can stop the CPU core operation during specific instruction fetches and data accesses, inspect the core registers and system peripheral state, and resume the core operation as needed.

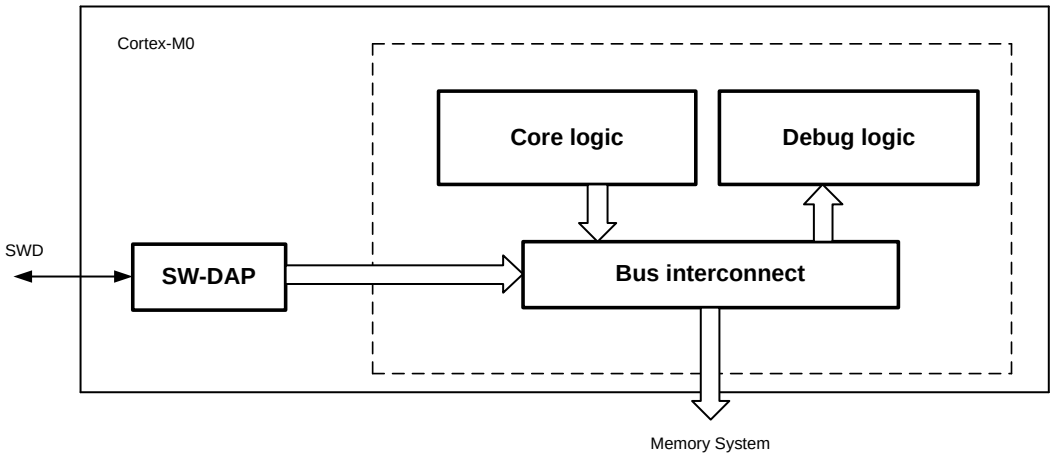


Figure 37-1 Cortex-M0 debug system diagram

For debug features of the Cortex-M0 core, please refer to the Cortex-M0 Technical Reference Manual from ARM.

37.2 Debug Pin

37.2.1 SWD Pin

The SWD pins of FM33LE0xxA series MCU are as follows:

SWD pins	Debug Function	Pin definition
SWDIO	SWD Data Input/Output	PD8
SWCLK	SWD Clock Input	PD7

Note: *Both PD7 and PD8 pins are default to be input state after chip reset, unlike most GPIOs.*

37.2.2 Pull-up Resistance

After chip reset, the SWDIO pin enables the internal pull-up (~ 100K ohm) by default, the SWCLK pin does not enable the internal pull-up resistor by default, so users need to connect external pull-up resistors or enable pull-up resistors by software on the PCB to prevent the floating of the input pin from causing increased leakage.

37.3 SW Interface Protocol

37.3.1 SW Protocol Introduction

SWD protocol uses LSB-first for data sending and receiving. Through the SWD interface, the debug master can read and write DPACC and APACC register sets.

SWIO needs to insert turn-around time on the bus each time the data direction is switched, and neither the host nor the slave will drive SWIO during this time. Between two transmissions, the host must drive the line low to enter the idle state, or continue to send the start bit of a new transmission to continue transmission. after a packet transmission, the host can also be idle to keep the line high or The SWD protocol does not have an explicit reset signal, and the host or target will detect a reset when it does not see the expected signal. By holding the line high for 50 clock cycles followed by a request to read the ID, a successful resynchronization can be ensured after an error or reset is detected.

37.3.2 SW Protocol sequence

Each SWD communication transmission sequence consists of three parts.

1. Packet request (8bits), sent by the host

2. ACK response (3bits), sent back by the target

3. Data transfer phase (33bits), sent by the host or target

where the packet request byte is defined as follows:

Bit	Name	Description
0	Start	Start bit, must be 1
1	ApnDP	AP/DP select 0：DP access 1：AP access
2	RnW	Read/Write select 0：write request 1：read request
4:3	A[3:2]	Address field of DP/AP register
5	Parity	Check bits for Bit0~Bit4 data
6	Stop	0
7	Park	Host not drive， pull-up via bus， target reads as 1

After the packet request is sent, there is always a 1bit turn-around time on the bus.

The ACK response is defined as follows:

Bit	Name	Description
0:2	ACK	001：FAULT 010：WAIT 100：OK

If the host initiates a read operation, or if the ACK is WAIT or FAULT, a turn-around time must be inserted after the ACK.

The data transfer format is as follows:

Bit	Name	Description
0:31	Data	Data of Read or write
32	Parity	Parity bit of 32bit data

37.3.3 SW-DP ID code

The SW-DP of Cortex-M0 has a fixed ID code: 0x0BB11477

The SW-DP is inactive until the host reads the ID code.

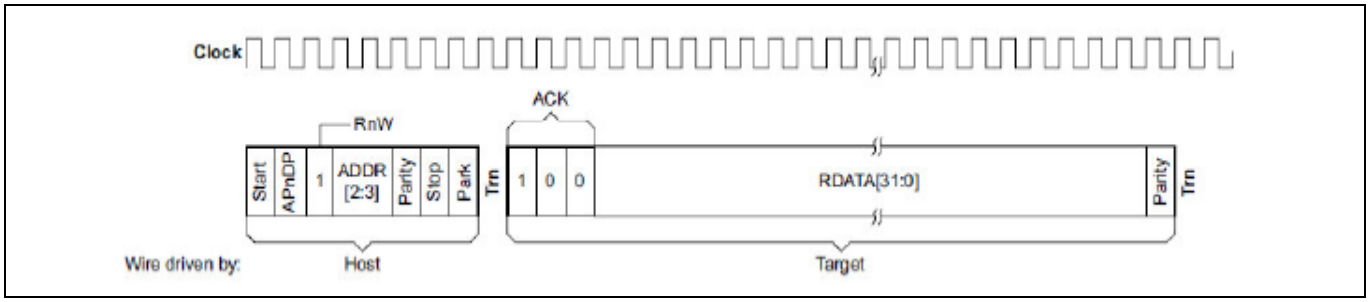
- SW-DP is in RESET state after chip reset, or after SWIO is pulled high for 50 SWCLK cycles
- After pulling SWIO low for at least 2 SWCLK cycles, SW-DP enters IDLE state
- When the SW-DP is in RESET, the host must first bring it into IDLE and then perform a read operation on the ID code register to activate the SW-DP. otherwise the slave will respond with a FAULT response to the host's communication.

37.3.4 Host Read

A successful read operation consists of the following three phases:

- An 8-bit read packet request from the host to the target.
- A 3-bit answer (ack) from the target to the host. A successful OK response is 100, a WAIT response is 010, and a FAULT response is 001.
- A 33-bit data read phase (payload) from the host to the target.

By default, there is a clocked turnaround period between the first and second phases and after the third phase, and a successful read operation is shown below.

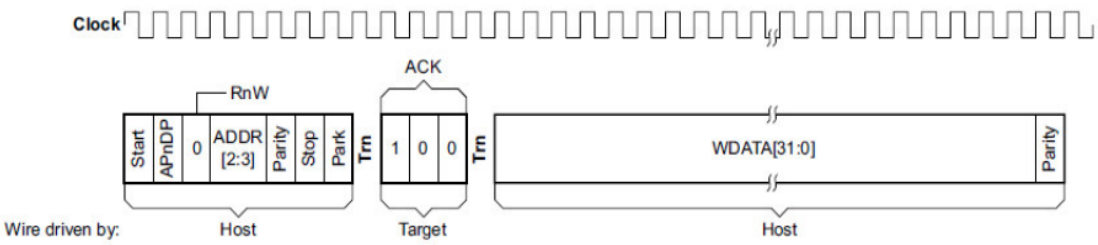


37.3.5 Host Write

A write operation consists of the following three phases

- An 8-bit write packet request (header) from the host to the target.
- A 3-bit answer (ack) from the target to the host. OK ack is 100 and FAULT ack is 001.
- A 33-bit data write phase (payload) from the host to the target.

By default, there is a clocked turnaround period between each two phases, and a successful write operation is shown below.



37.4 SWD-DP Register

37.4.1 Register List

Address (A[3:2])	DPBANKSEL	Name	Access
00	x	DHCSR	RO
		ABORT	WO
01	0x0	CTRL/STAT	RW
	0x1	DLCR	RW
10	x	RESEND	RO
		SELECT	WO
11	x	RDBUFF	RO

For detailed description of the registers, please refer to the Cortex-M0 Technical Reference Manual.

37.5 Core Debug Register

Core debug can be implemented by operating the core debug registers. The host accesses the following core debug registers via SW-DP.

Address	Name	Type	Function
0xE000EDF0	DHCSR	RW	Debug Halting Control and Status Register
0xE000EDF4	DCRSR	WO	Debug Core Register Selector Register
0xE000EDF8	DCRDR	RW	Debug Core Register Data Register
0xE000EDFC	DEMCR	RW	Debug Exception and Monitor Control Register
0xE000EE00 to 0xE000EEFF	-	-	Reserved for Debug Extension

The above debug registers are not affected by system reset and are only affected by power-on reset. Immediate halt after CPU reset can be achieved by：

- Setting bit0 of DEMCR register (VC_CORRESET)
- Displacing bit0 of DHCSR register (C_DEBUGEN)
- Performing a system reset

37.6 Debug-related configuration

By configuring the DBG_CR register, you can set whether the chip's internal timer and watchdog circuit continue to work in the debug state. For details, please refer to 6.5.1 DEBUG Configuration Register (DBG_CR).

Revision History

NUMBER	DATE	PAGE	CHAPTER & DIAGRAM	DETAILS
1.0	2022.07			First publication
1.1	2022.07			Update 125C parameters
1.2	2022.08		3.5.1 7.4.3 32.7.6	1. Revision of the bug that FLASH can be programmed other than by word. 2. Revision of the range of external capacitance of LDO15. 3. Revision of ADC channel sample time register description
1.3	2022.12		32	Delete the DMA cycle mode of the ADC, DMACFG register shall be kept as 0.
1.4	2023.2		2.1.4.3	Add QFN32 wettable flank
1.5	2023.3			Correct some clerical errors
1.6	2023.3		3.5	Modification of the upper and lower limits of some electrical parameters according to the extraction of ortho parameters
1.7	2023.5			Update the upper and lower limits of some parameters according to the extraction results of the three-temperature parameters

Contact Us

Shanghai Fudan Microelectronics Group Co., Ltd.
Address: Building 4, 127 Guotai Road, Shanghai, China
Postcode: 200433
Tel: (86-021) 6565 5050
Fax: (86-021) 6565 9115

Shanghai Fudan Microelectronics(HK)Co.,Ltd.
Address: Unit 506,5/f.,East Ocean Centre, 98 Granville Road,Tsim Sha Tsui East, Kowloon, Hong Kong
Tel: (852) 2116 3288/2116 3338
Fax: (852) 2116 0882

Beijing Fudan Microelectronics Technology Co.,Ltd.
Address:Gehua Buiding.B.423, North Street, Dongcheng District,Beijing City,China
Postcode: 100007
Tel: (86-10) 8418 6608 / 8418 7486
Fax: (86-10) 8418 6211

Shenzhen Fudan Microelectronics Co., Ltd.
Address:Room.1303,Century Bldg,Shengtingyuan Hotel, Huaqiang Rd.(North),Shenzhen,China
Postcode: 518028
Tel: (86-755) 8335 0911 / 8335 1011 / 8335 2011 / 8335 0611
Fax: (86-755) 8335 9011

Shanghai Fudan Microelectronics (HK) Ltd Taiwan Representative Office
Address: Unit 1225,12F,No.252,Sec.1 Neihu Rd.,Neihu Dist.,Taipei City 114,Taiwan
Tel: (886-2) 7721 1889 / 7721 1890
Fax: (886-2) 7722 3888

Shanghai Fudan Microelectronics (HK) LtdSingapore Branch Office
Address: 47 Kallang Pudding Road, #08-06, The Crescent @ Kallang, Singapore 349318
Tel: (65) 6443 0860
Fax: (65) 64431215

Fudan Microelectronics (USA) Inc.
Address: 97 EBrokaw Road, Suite 320, San Jose, CA 95112 USA
Tel: (+1) 408 335 6936

Website: http://www.fmsh.com/