GigaDevice Semiconductor Inc.

GD32C11x
Arm® Cortex®-M4 32-bit MCU

For GD32C113xx

User Manual

Revision 1.4

(Jan. 2026)

Table of Contents

Table of Contents 2
List of Figures 15
List of Tables 21
1. System and memory architecture 23
1.1. ARM Cortex-M4 processor 23
1.2. System architecture 24
1.3. Memory map 26
1.3.1. Bit-banding 30
1.3.2. On-chip SRAM memory 31
1.3.3. On-chip flash memory overview 31
1.4. Boot configuration 31
1.5. Device electronic signature 32
1.5.1. Memory density information 33
1.5.2. Unique device ID (96 bits) 33
1.6. System configuration registers 34
2. Flash memory controller (FMC) 35
2.1. Overview 35
2.2. Characteristics 35
2.3. Function overview 35
2.3.1. Flash memory architecture 35
2.3.2. Read operations 36
2.3.3. Unlock the FMC_CTL register 37
2.3.4. Page erase 38
2.3.5. Mass erase 39
2.3.6. Main flash programming 40
2.3.7. OTP programming 42
2.3.8. Option bytes Erase 43
2.3.9. Option bytes modify 43
2.3.10. Option bytes description 44
2.3.11. Page erase/program protection 45
2.3.12. Security protection 45
2.4. Register definition 45
2.4.1. Wait state register (FMC_WS) 46
2.4.2. Unlock key register (FMC_KEY) 47
2.4.3. Option byte unlock key register (FMC_OBKEY) 47

2.4.4. Status register (FMC_STAT) 48
2.4.5. Control register (FMC_CTL) 48
2.4.6. Address register (FMC_ADDR) 50
2.4.7. Option byte status register (FMC_OBSTAT) 50
2.4.8. Erase/Program protection register (FMC_WP) 51
2.4.9. Product ID register (FMC_PID) 51
3. Power management unit (PMU) 53
3.1. Overview 53
3.2. Characteristics 53
***3.3.* Function overview 53**
3.3.1. Backup domain 54
3.3.2. V_{DD} / V_{DDA} power domain 55
3.3.3. V_{CORE} power domain 57
3.3.4. Power saving modes 57
3.4. Register definition 60
3.4.1. Control register (PMU_CTL) 60
3.4.2. Control and status register (PMU_CS) 61
4. Backup registers (BKP) 63
4.1. Overview 63
4.2. Characteristics 63
4.3. Function overview 63
4.3.1. RTC clock calibration 63
4.3.2. Tamper detection 63
4.4. Register definition 65
4.4.1. Backup data register x (BKP_DATAx) (x= 0..41) 65
4.4.2. RTC signal output control register (BKP_OCTL) 65
4.4.3. Tamper pin control register (BKP_TPCTL) 66
4.4.4. Tamper control and status register (BKP_TPCS) 66
5. Reset and clock unit (RCU) 68
5.1. Reset control unit (RCTL) 68
5.1.1. Overview 68
5.1.2. Function overview 68
5.2. Clock control unit (CCTL) 69
5.2.1. Overview 69
5.2.2. Characteristics 71
5.2.3. Function overview 71
5.3. Register definition 76
5.3.1. Control register (RCU_CTL) 76
5.3.2. Clock configuration register 0 (RCU_CFG0) 78

5.3.3. Clock interrupt register (RCU_INT) 81
5.3.4. APB2 reset register (RCU_APB2RST) 84
5.3.5. APB1 reset register (RCU_APB1RST) 86
5.3.6. AHB enable register (RCU_AHBEN) 89
5.3.7. APB2 enable register (RCU_APB2EN) 91
5.3.8. APB1 enable register (RCU_APB1EN) 93
5.3.9. Backup domain control register (RCU_BDCTL) 96
5.3.10. Reset source/clock register (RCU_RSTSCK) 97
5.3.11. AHB reset register (RCU_AHBRST) 98
5.3.12. Clock configuration register 1 (RCU_CFG1) 99
5.3.13. Deep-sleep mode voltage register (RCU_DSV) 101
5.3.14. Additional clock control register (RCU_ADDCTL) 102
5.3.15. Additional clock interrupt register (RCU_ADDINT) 103
5.3.16. APB1 additional reset register (RCU_ADDAPB1RST) 104
5.3.17. APB1 additional enable register (RCU_ADDAPB1EN) 104

6. Clock trim controller (CTC) 106

6.1. Overview 106

6.2. Characteristics 106

6.3. Function overview 106
6.3.1. Reference sync pulse generator 107
6.3.2. CTC trim counter 107
6.3.3. Frequency evaluation and automatic trim process 108
6.3.4. Software program guide 109

6.4. Register definition 111
6.4.1. Control register 0 (CTC_CTL0) 111
6.4.2. Control register 1 (CTC_CTL1) 112
6.4.3. Status register (CTC_STAT) 113
6.4.4. Interrupt clear register (CTC_INTC) 115

7. Interrupt / event controller (EXTI) 117

7.1. Overview 117

7.2. Characteristics 117

7.3. Function overview 117

7.4. External interrupt and event block diagram 121

7.5. External Interrupt and Event function overview 121

7.6. Register definition 123
7.6.1. Interrupt enable register (EXTI_INTEN) 123
7.6.2. Event enable register (EXTI_EVEN) 123
7.6.3. Rising edge trigger enable register (EXTI_RTEN) 124
7.6.4. Falling edge trigger enable register (EXTI_FTEN) 124
7.6.5. Software interrupt event register (EXTI_SWIEV) 124

7.6.6. Pending register (EXTI_PD) 125
8. General-purpose and alternate-function I/Os (GPIO and AFIO) 126
8.1. Overview 126
8.2. Characteristics 126
8.3. Function overview 126
8.3.1. GPIO pin configuration 127
8.3.2. External interrupt/event lines 128
8.3.3. Alternate functions (AF) 128
8.3.4. Input configuration 128
8.3.5. Output configuration 129
8.3.6. Analog configuration 129
8.3.7. Alternate function (AF) configuration 130
8.3.8. GPIO locking function 130
8.3.9. GPIO I/O compensation cell 131
8.4. Remapping function I/O and debug configuration 131
8.4.1. Overview 131
8.4.2. Characteristics 131
8.4.3. JTAG/SWD alternate function remapping 131
8.4.4. ADC AF remapping 132
8.4.5. TIMER AF remapping 133
8.4.6. USART AF remapping 134
8.4.7. I2C0 AF remapping 135
8.4.8. SPI0/SPI2/I2S AF remapping 135
8.4.9. CAN0/1 AF remapping 136
8.4.10. CTC AF remapping 136
8.4.11. CLK pins AF remapping 136
8.5. Register definition 138
8.5.1. Port control register 0 (GPIOx_CTL0, x=A..E) 138
8.5.2. Port control register 1 (GPIOx_CTL1, x=A..E) 140
8.5.3. Port input status register (GPIOx_ISTAT, x=A..E) 141
8.5.4. Port output control register (GPIOx_OCTL, x=A..E) 142
8.5.5. Port bit operate register (GPIOx_BOP, x=A..E) 142
8.5.6. Port bit clear register (GPIOx_BC, x=A..E) 143
8.5.7. Port configuration lock register (GPIOx_LOCK, x=A..E) 143
8.5.8. Port bit speed register (GPIOx_SPD, x=A..E) 144
8.5.9. Event control register (AFIO_EC) 144
8.5.10. AFIO port configuration register 0 (AFIO_PCF0) 145
8.5.11. EXTI sources selection register 0 (AFIO_EXTISS0) 148
8.5.12. EXTI sources selection register 1 (AFIO_EXTISS1) 149
8.5.13. EXTI sources selection register 2 (AFIO_EXTISS2) 151
8.5.14. EXTI sources selection register 3 (AFIO_EXTISS3) 152
8.5.15. AFIO port configuration register 1 (AFIO_PCF1) 153

8.5.16. IO compensation control register (AFIO_CPSCTL) 154
9. CRC calculation unit (CRC) 155
9.1. Overview 155
9.2. Characteristics 155
9.3. Function overview 156
9.4. Register definition 157
9.4.1. Data register (CRC_DATA) 157
9.4.2. Free data register (CRC_FDATA) 157
9.4.3. Control register (CRC_CTL) 158
10. Direct memory access controller (DMA) 159
10.1. Overview 159
10.2. Characteristics 159
10.3. Block diagram 160
10.4. Function overview 160
10.4.1. DMA operation 160
10.4.2. Peripheral handshake 162
10.4.3. Arbitration 162
10.4.4. Address generation 162
10.4.5. Circular mode 163
10.4.6. Memory to memory mode 163
10.4.7. Channel configuration 163
10.4.8. Interrupt 163
10.4.9. DMA request mapping 164
10.5. Register definition 167
10.5.1. Interrupt flag register (DMA_INTF) 167
10.5.2. Interrupt flag clear register (DMA_INTC) 168
10.5.3. Channel x control register (DMA_CHxCTL) 168
10.5.4. Channel x counter register (DMA_CHxCNT) 170
10.5.5. Channel x peripheral base address register (DMA_CHxPADDR) 171
10.5.6. Channel x memory base address register (DMA_CHxMADDR) 171
11. Debug (DBG) 173
11.1. Overview 173
11.2. JTAG/SW function overview 173
11.2.1. Switch JTAG or SW interface 173
11.2.2. Pin assignment 173
11.2.3. JTAG daisy chained structure 174
11.2.4. Debug reset 174
11.2.5. JEDEC-106 ID code 174
11.3. Debug hold function overview 174

11.3.1. Debug support for power saving mode 174
11.3.2. Debug support for TIMER, I2C, WWDGT, FWDGT and CAN 175
11.4. Register definition 176
11.4.1. ID code register (DBG_ID) 176
11.4.2. Control register (DBG_CTL) 176
12. Analog-to-digital converter (ADC) 180
12.1. Overview 180
12.2. Characteristics 180
12.3. Pins and internal signals 181
12.4. Function overview 182
12.4.1. Foreground calibration function 182
12.4.2. ADC clock 183
12.4.3. ADCON enable 183
12.4.4. Routine sequence 183
12.4.5. Operation modes 183
12.4.6. Conversion result threshold monitor function 186
12.4.7. Data storage mode 186
12.4.8. Sample time configuration 187
12.4.9. External trigger configuration 187
12.4.10. DMA request 188
12.4.11. ADC internal channels 188
12.4.12. Programmable resolution (DRES) 189
12.4.13. On-chip hardware oversampling 189
12.5. ADC sync mode 191
12.5.1. Free mode 192
12.5.2. Routine parallel mode 192
12.5.3. Routine follow-up fast mode 193
12.5.4. Routine follow-up slow mode 193
12.6. ADC interrupts 194
12.7. Register definition 195
12.7.1. Status register (ADC_STAT) 195
12.7.2. Control register 0 (ADC_CTL0) 195
12.7.3. Control register 1 (ADC_CTL1) 197
12.7.4. Sample time register 0 (ADC_SAMPT0) 199
12.7.5. Sample time register 1 (ADC_SAMPT1) 200
12.7.6. Watchdog high threshold register (ADC_WDHT) 201
12.7.7. Watchdog low threshold register (ADC_WDLT) 201
12.7.8. Routine sequence register 0 (ADC_RSQ0) 201
12.7.9. Routine sequence register 1 (ADC_RSQ1) 202
12.7.10. Routine sequence register 2 (ADC_RSQ2) 203
12.7.11. Routine data register (ADC_RDATA) 203

12.7.12. Oversample control register (ADC_OVSAMPCTL) 204
13. Digital-to-analog converter (DAC) 206
13.1. Overview 206
13.2. Characteristics 206
13.3. Function overview 208
13.3.1. DAC enable 208
13.3.2. DAC output buffer 208
13.3.3. DAC data configuration 208
13.3.4. DAC trigger 208
13.3.5. DAC conversion 209
13.3.6. DAC noise wave 209
13.3.7. DAC output voltage 210
13.3.8. DMA request 210
13.3.9. DAC concurrent conversion 210
13.4. Register definition 211
13.4.1. DACx control register 0 (DAC_CTL0) 211
13.4.2. DACx software trigger register (DAC_SWT) 213
13.4.3. DACx_OUT0 12-bit right-aligned data holding register (DAC_OUT0_R12DH) 214
13.4.4. DACx_OUT0 12-bit left-aligned data holding register (DAC_OUT0_L12DH) 214
13.4.5. DACx_OUT0 8-bit right-aligned data holding register (DAC_OUT0_R8DH) 215
13.4.6. DACx_OUT1 12-bit right-aligned data holding register (DAC_OUT1_R12DH) 215
13.4.7. DACx_OUT1 12-bit left-aligned data holding register (DAC_OUT1_L12DH) 216
13.4.8. DACx_OUT1 8-bit right-aligned data holding register (DAC_OUT1_R8DH) 216
13.4.9. DACx concurrent mode 12-bit right-aligned data holding register (DACC_R12DH) 217
13.4.10. DACx concurrent mode 12-bit left-aligned data holding register (DACC_L12DH) 217
13.4.11. DACx concurrent mode 8-bit right-aligned data holding register (DACC_R8DH) 218
13.4.12. DACx_OUT0 data output register (DAC_OUT0_DO) 218
13.4.13. DACx_OUT1 data output register (DAC_OUT1_DO) 219
14. Watchdog timer (WDGT) 220
14.1. Free watchdog timer (FWDGT) 220
14.1.1. Overview 220
14.1.2. Characteristics 220
14.1.3. Function overview 220
14.1.4. Register definition 223
14.2. Window watchdog timer (WWDGT) 226
14.2.1. Overview 226
14.2.2. Characteristics 226
14.2.3. Function overview 226
14.2.4. Register definition 229
15. Real-time Clock (RTC) 231

15.1. Overview 231
15.2. Characteristics 231
15.3. Function overview 231
15.3.1. RTC reset 232
15.3.2. RTC reading 232
15.3.3. RTC configuration 232
15.3.4. RTC flag assertion 233
15.4. Register definition 235
15.4.1. RTC interrupt enable register(RTC_INTEN) 235
15.4.2. RTC control register(RTC_CTL) 235
15.4.3. RTC prescaler high register (RTC_PSCH) 236
15.4.4. RTC prescaler low register(RTC_PSCL) 236
15.4.5. RTC divider high register (RTC_DIVH) 237
15.4.6. RTC divider low register (RTC_DIVL) 237
15.4.7. RTC counter high register(RTC_CNTH) 237
15.4.8. RTC counter low register (RTC_CNTL) 238
15.4.9. RTC alarm high register(RTC_ALRMH) 238
15.4.10. RTC alarm low register (RTC_ALRML) 238
16. TIMER 239
16.1. Advanced timer (TIMERx, x=0, 7) 240
16.1.1. Overview 240
16.1.2. Characteristics 240
16.1.3. Block diagram 241
16.1.4. Function overview 242
16.1.5. TIMERx registers (x=0, 7) 268
16.2. General level0 timer (TIMERx, x=1, 2, 3, 4) 294
16.2.1. Overview 294
16.2.2. Characteristics 294
16.2.3. Block diagram 295
16.2.4. Function overview 296
16.2.5. TIMERx registers(x=1, 2, 3, 4) 310
16.3. General level1 timer (TIMERx, x=8, 11) 332
16.3.1. Overview 332
16.3.2. Characteristics 332
16.3.3. Block diagram 332
16.3.4. Function overview 333
16.3.5. TIMERx registers (x=8, 11) 345
16.4. General level2 timer (TIMERx, x=9, 10, 12, 13) 357
16.4.1. Overview 357
16.4.2. Characteristics 357
16.4.3. Block diagram 358

16.4.4. Function overview 358
16.4.5. TIMERx registers (x=9, 10, 12, 13) 366
16.5. Basic timer (TIMERx, x=5, 6) 375
16.5.1. Overview 375
16.5.2. Characteristics 375
16.5.3. Block diagram 375
16.5.4. Function overview 376
16.5.5. TIMERx registers (x=5, 6) 380
17. Universal synchronous/asynchronous receiver /transmitter (USART) 385
17.1. Overview 385
17.2. Characteristics 385
17.3. Function overview 386
17.3.1. USART frame format 387
17.3.2. Baud rate generation 388
17.3.3. USART transmitter 388
17.3.4. USART receiver 389
17.3.5. Use DMA for data buffer access 391
17.3.6. Hardware flow control 392
17.3.7. Multi-processor communication 393
17.3.8. LIN mode 394
17.3.9. Synchronous mode 395
17.3.10. IrDA SIR ENDEC mode 395
17.3.11. Half-duplex communication mode 397
17.3.12. Smartcard (ISO7816-3) mode 397
17.3.13. USART interrupts 399
17.4. Register definition 401
17.4.1. Status register 0 (USART_STAT0) 401
17.4.2. Data register (USART_DATA) 403
17.4.3. Baud rate register (USART_BAUD) 403
17.4.4. Control register 0 (USART_CTL0) 404
17.4.5. Control register 1 (USART_CTL1) 406
17.4.6. Control register 2 (USART_CTL2) 407
17.4.7. Guard time and prescaler register (USART_GP) 409
17.4.8. Control register 3 (USART_CTL3) 410
17.4.9. Receiver timeout register (USART_RT) 411
17.4.10. Status register 1 (USART_STAT1) 412
17.4.11. Coherence control register (USART_CHC) 413
18. Inter-integrated circuit interface (I2C) 414
18.1. Overview 414
18.2. Characteristics 414

18.3. Function overview 414
18.3.1. SDA and SCL lines 415
18.3.2. Data validation 416
18.3.3. START and STOP signal 416
18.3.4. Clock synchronization 416
18.3.5. Arbitration 417
18.3.6. I2C communication flow 417
18.3.7. Programming model 418
18.3.8. SCL line stretching 427
18.3.9. Use DMA for data transfer 428
18.3.10. Packet error checking 428
18.3.11. SMBus support 428
18.3.12. SAM_V support 430
18.3.13. Status, errors and interrupts 430
18.4. Register definition 432
18.4.1. Control register 0 (I2C_CTL0) 432
18.4.2. Control register 1 (I2C_CTL1) 434
18.4.3. Slave address register 0 (I2C_SADDR0) 435
18.4.4. Slave address register 1 (I2C_SADDR1) 435
18.4.5. Transfer buffer register (I2C_DATA) 436
18.4.6. Transfer status register 0 (I2C_STAT0) 436
18.4.7. Transfer status register 1 (I2C_STAT1) 439
18.4.8. Clock configure register (I2C_CKCFG) 440
18.4.9. Rise time register (I2C_RT) 441
18.4.10. SAM control and status register (I2C_SAMCS) 441
18.4.11. Fast-mode-plus configure register(I2C_FMPCFG) 442
19. Serial peripheral interface/Inter-IC sound (SPI/I2S) 444
19.1. Overview 444
19.2. Characteristics 444
19.2.1. SPI characteristics 444
19.2.2. I2S characteristics 444
19.3. SPI function overview 445
19.3.1. SPI block diagram 445
19.3.2. SPI signal description 445
19.3.3. SPI clock timing and data format 446
19.3.4. NSS function 447
19.3.5. SPI operating modes 448
19.3.6. DMA function 457
19.3.7. CRC function 457
19.3.8. SPI interrupts 458
19.4. I2S function overview 459
19.4.1. I2S block diagram 459

19.4.2. I2S signal description 460
19.4.3. I2S audio standards 460
19.4.4. I2S clock 468
19.4.5. Operation 469
19.4.6. DMA function 473
19.4.7. I2S interrupts 473
19.5. Register definition 475
19.5.1. Control register 0 (SPI_CTL0) 475
19.5.2. Control register 1 (SPI_CTL1) 477
19.5.3. Status register (SPI_STAT) 478
19.5.4. Data register (SPI_DATA) 479
19.5.5. CRC polynomial register (SPI_CRCPOLY) 480
19.5.6. RX CRC register (SPI_RCRC) 480
19.5.7. TX CRC register (SPI_TCRC) 481
19.5.8. I2S control register (SPI_I2SCTL) 482
19.5.9. I2S clock prescaler register (SPI_I2SPSC) 483
19.5.10. Quad-SPI mode control register (SPI_QCTL) of SPI0 484
20. External memory controller (EXMC) 485
20.1. Overview 485
20.2. Characteristics 485
20.3. Function overview 485
20.3.1. Block diagram 485
20.3.2. Basic regulation of EXMC access 486
20.3.3. NOR/PSRAM controller 487
20.4. Register definition 507
20.4.1. SRAM/NOR Flash control registers (EXMC_SNCTL) 507
20.4.2. SRAM/NOR Flash timing configuration registers (EXMC_SNTCFG) 509
20.4.3. SRAM/NOR Flash write timing configuration registers (EXMC_SNWTCFG) 510
21. Controller area network (CAN) 512
21.1. Overview 512
21.2. Characteristics 512
21.3. Function overview 513
21.3.1. Working mode 513
21.3.2. Communication modes 514
21.3.3. Data transmission 515
21.3.4. Data reception 517
21.3.5. Filtering function 519
21.3.6. Time-triggered communication 522
21.3.7. Communication parameters 522
21.3.8. CAN FD operation 524
21.3.9. Transmitter Delay Compensation 525

21.3.10. Error flags 526
21.3.11. CAN interrupts 526
21.4. Register definition 529
21.4.1. Control register (CAN_CTL) 529
21.4.2. Status register (CAN_STAT) 530
21.4.3. Transmit status register (CAN_TSTAT) 532
21.4.4. Receive message FIFO0 register (CAN_RFIFO0) 534
21.4.5. Receive message FIFO1 register (CAN_RFIFO1) 535
21.4.6. Interrupt enable register (CAN_INTEN) 536
21.4.7. Error register (CAN_ERR) 537
21.4.8. Bit timing register (CAN_BT) 538
21.4.9. FD control register (CAN_FDCTL) 539
21.4.10. FD status register (CAN_FDSTAT) 540
21.4.11. FD transmitter delay compensation register (CAN_FDTDC) 541
21.4.12. Date Bit timing register (CAN_DBT) 541
21.4.13. Transmit mailbox identifier register (CAN_TMIx) (x = 0...2) 542
21.4.14. Transmit mailbox property register (CAN_TMPx) (x = 0…2) 543
21.4.15. Transmit mailbox data0 register (CAN_TMDATA0x) (x = 0…2) 544
21.4.16. Transmit mailbox data1 register (CAN_TMDATA1x) (x = 0...2) 544
21.4.17. Rx FIFO mailbox identifier register (CAN_RFIFOMIx) (x = 0,1) 545
21.4.18. Rx FIFO mailbox property register (CAN_RFIFOMPx) (x = 0,1) 545
21.4.19. Rx FIFO mailbox data0 register (CAN_RFIFOMDATA0x) (x = 0,1) 546
21.4.20. Rx FIFO mailbox data1 register (CAN_RFIFOMDATA1x) (x=0,1) 547
21.4.21. Filter control register (CAN_FCTL) 547
21.4.22. Filter mode configuration register (CAN_FMCFG) 548
21.4.23. Filter scale configuration register (CAN_FSCFG) 548
21.4.24. Filter associated FIFO register (CAN_FAFIFO) 549
21.4.25. Filter working register (CAN_FW) 549
21.4.26. Filter x data y register (CAN_FxDATAy) (x=0..27, y=0,1) 550
22. Universal serial bus full-speed interface (USBFS) 551
22.1. Overview 551
22.2. Characteristics 551
22.3. Block diagram 552
22.4. Signal description 552
22.5. Function overview 552
22.5.1. USBFS clocks and working modes 552
22.5.2. USB host function 554
22.5.3. USB device function 556
22.5.4. OTG function overview 557
22.5.5. Data FIFO 558
22.5.6. Operation guide 560

22.6. Interrupts 565
22.7. Register definition 567
22.7.1. Global control and status registers 567
22.7.2. Host control and status registers 588
22.7.3. Device control and status registers 599
22.7.4. Power and clock control register (USBFS_PWRCLKCTL) 623
23. Appendix 625
23.1. List of abbreviations used in register 625
23.2. List of terms 625
23.3. Available peripherals 626
24. Revision history 627

List of Figures

Figure 1-1. The structure of the Cortex®-M4 processor 24
Figure 1-2. GD32C11x series system architecture 26
Figure 2-1. Process of page erase operation 39
Figure 2-2. Process of mass erase operation 40
Figure 2-3. Process of word program operation 42
Figure 3-1. Power supply overview 54
Figure 3-2. Waveform of the POR / PDR 56
Figure 3-3. Waveform of the LVD threshold 56
Figure 5-1. The system reset circuit 69
Figure 5-2. Clock tree 70
Figure 5-3. HXTAL clock source 71
Figure 5-4. HXTAL clock source in bypass mode 72
Figure 6-1. CTC overview 107
Figure 6-2. CTC trim counter 108
Figure 7-1. Block diagram of EXTI 121
Figure 8-1. Basic structure of of a general-pupose I/O 127
Figure 8-2. Basic structure of Input configuration 128
Figure 8-3. Basic structure of Output configuration 129
Figure 8-4. Basic structure of Analog configuration 130
Figure 8-5. Basic structure of Alternate function configuration 130
Figure 9-1. Block diagram of CRC calculation unit 156
Figure 10-1. Block diagram of DMA 160
Figure 10-2. Handshake mechanism 162
Figure 10-3. DMA interrupt logic 164
Figure 10-4. DMA0 request mapping 165
Figure 10-5. DMA1 request mapping 166
Figure 12-1. ADC module block diagram 182
Figure 12-2. Single operation mode 183
Figure 12-3. Continuous operation mode 184
Figure 12-4. Scan operation mode, continuous disable 185
Figure 12-5. Scan operation mode, continuous enable 185
Figure 12-6. Discontinuous operation mode 186
Figure 12-7. 12-bit data storage mode 187
Figure 12-8. 6-bit data storage mode 187
Figure 12-9. 20-bit to 16-bit result truncation 190
Figure 12-10. A numerical example with 5-bit shifting and rounding 190
Figure 12-11. ADC sync block diagram 192
Figure 12-12. Routine parallel mode on 10 channels 193
Figure 12-13. Routine follow-up fast mode (the CTN bit of ADCs are set) 193
Figure 12-14. Routine follow-up slow mode 194
Figure 13-1. DAC block diagram 207

Figure 13-2. DAC LFSR algorithm 209
Figure 13-3. DAC triangle noise wave 210
Figure 14-1. Free watchdog block diagram 221
Figure 14-2. Window watchdog timer block diagram 226
Figure 14-3. Window watchdog timing diagram 227
Figure 15-1. Block diagram of RTC 232
Figure 16-1. Advanced timer block diagram 241
Figure 16-2. Timing chart of internal clock divided by 1 242
Figure 16-3. Timing chart of PSC value change from 0 to 2 243
Figure 16-4. Timing chart of up counting mode, PSC=0/2 244
Figure 16-5. Timing chart of up counting mode, change TIMERx_CAR on the go 245
Figure 16-6. Timing chart of down counting mode, PSC=0/2 246
Figure 16-7. Timing chart of down counting mode, change TIMERx_CAR on the go 246
Figure 16-8. Center-aligned counter timechart 248
Figure 16-9. Repetition timechart for center-aligned counter 249
Figure 16-10. Repetition timechart for up-counter 249
Figure 16-11. Repetition timechart for down-counter 250
Figure 16-12. Channel input capture principle 251
Figure 16-13. Output-compare under three modes 253
Figure 16-14. EAPWM timechart 254
Figure 16-15. CAPWM timechart 254
Figure 16-16. Complementary output with dead-time insertion 257
Figure 16-17. Output behavior in response to a break(The break high active) 258
Figure 16-18. Counter behavior with CI0FE0 polarity non-inverted in mode 2 259
Figure 16-19. Counter behavior with CI0FE0 polarity inverted in mode 2 259
Figure 16-20. Hall sensor is used to BLDC motor 260
Figure 16-21. Hall sensor timing between two timers 261
Figure 16-22. Restart mode 262
Figure 16-23. Pause mode 263
Figure 16-24. Event mode 263
Figure 16-25. Single pulse mode TIMERx_CHxCV = 4 TIMERx_CAR=99 264
Figure 16-26. Timer0 master/slave mode timer example 265
Figure 16-27. Triggering TIMER0 with enable signal of TIMER2 266
Figure 16-28. Triggering TIMER0 and TIMER2 with TIMER2's CI0 input 267
Figure 16-29. General Level 0 timer block diagram 296
Figure 16-30. Timing chart of internal clock divided by 1 297
Figure 16-31. Timing chart of PSC value change from 0 to 2 298
Figure 16-32. Timing chart of up counting mode, PSC=0/2 299
Figure 16-33. Timing chart of up counting mode, change TIMERx_CAR ongoing 299
Figure 16-34. Timing chart of down counting mode, PSC=0/2 300
Figure 16-35. Timing chart of down counting mode, change TIMERx_CAR ongoing 301
Figure 16-36. Timing chart of center-aligned counting mode 302
Figure 16-37. Channel input capture principle 303
Figure 16-38. Output-compare under three modes 305

Figure 16-39. EAPWM timechart 306
Figure 16-40. CAPWM timechart 306
Figure 16-41. Restart mode 308
Figure 16-42. Pause mode 308
Figure 16-43. Event mode 309
Figure 16-44. General level1 timer block diagram 333
Figure 16-45. Timing chart of internal clock divided by 1 334
Figure 16-46. Timing chart of PSC value change from 0 to 2 335
Figure 16-47. Timing chart of up counting mode, PSC=0/2 336
Figure 16-48. Timing chart of up counting mode, change TIMERx_CAR ongoing 336
Figure 16-49. Channel input capture principle 337
Figure 16-50. Output-compare under three modes 339
Figure 16-51. EAPWM timechart 340
Figure 16-52. CAPWM timechart 340
Figure 16-53. Restart mode 342
Figure 16-54. Pause mode 342
Figure 16-55. Event mode 343
Figure 16-56. Single pulse mode TIMERx_CHxCV = 4 TIMERx_CAR=99 344
Figure 16-57. General level2 timer block diagram 358
Figure 16-58. Timing chart of internal clock divided by 1 359
Figure 16-59. Timing chart of PSC value change from 0 to 2 359
Figure 16-60. Timing chart of up counting mode, PSC=0/2 360
Figure 16-61. Timing chart of up counting mode, change TIMERx_CAR on the go 361
Figure 16-62. Channel input capture principle 362
Figure 16-63. Output-compare under three modes 364
Figure 16-64. Basic timer block diagram 376
Figure 16-65. Timing chart of internal clock divided by 1 376
Figure 16-66. Timing chart of PSC value change from 0 to 2 377
Figure 16-67. Timing chart of up counting mode, PSC=0/2 378
Figure 16-68. Timing chart of up counting mode, change TIMERx_CAR ongoing 378
Figure 17-1. USART module block diagram 387
Figure 17-2. USART character frame (8 bits data and 1 stop bit) 387
Figure 17-3. USART transmit procedure 389
Figure 17-4. Receiving a frame bit by oversampling method 390
Figure 17-5. Configuration steps when using DMA for USART transmission 391
Figure 17-6. Configuration steps when using DMA for USART reception 392
Figure 17-7. Hardware flow control between two USARTs 392
Figure 17-8. Hardware flow control 393
Figure 17-9. Break frame occurs during idle state 394
Figure 17-10. Break frame occurs during a frame 394
Figure 17-11. Example of USART in synchronous mode 395
Figure 17-12. 8-bit format USART synchronous waveform (CLEN=1) 395
Figure 17-13. IrDA SIR ENDEC module 396
Figure 17-14. IrDA data modulation 396

Figure 17-15. ISO7816-3 frame format 397
Figure 17-16. USART interrupt mapping diagram 400
Figure 18-1. I2C module block diagram 415
Figure 18-2. Data validation 416
Figure 18-3. START and STOP signal 416
Figure 18-4. Clock synchronization 417
Figure 18-5. SDA line arbitration 417
Figure 18-6. I2C communication flow with 7-bit address 418
Figure 18-7. I2C communication flow with 10-bit address (Master Transmit) 418
Figure 18-8. I2C communication flow with 10-bit address (Master Receive) 418
Figure 18-9. Programming model for slave transmitting (10-bit address mode) 420
Figure 18-10. Programming model for slave receiving (10-bit address mode) 421
Figure 18-11. Programming model for master transmitting (10-bit address mode) 423
Figure 18-12. Programming model for master receiving using Solution A (10-bit address mode) 425
Figure 18-13. Programming model for master receiving mode using solution B (10-bit address mode) 426
Figure 19-1. Block diagram of SPI 445
Figure 19-2. SPI timing diagram in normal mode 446
Figure 19-3. SPI timing diagram in Quad-SPI mode (CKPL=1, CKPH=1, LF=0) 447
Figure 19-4. A typical full-duplex connection 449
Figure 19-5. A typical simplex connection (Master: Receive, Slave: Transmit) 450
Figure 19-6. A typical simplex connection (Master: Transmit only, Slave: Receive) 450
Figure 19-7. A typical bidirectional connection 450
Figure 19-8. Timing diagram of TI master mode with discontinuous transfer 452
Figure 19-9. Timing diagram of TI master mode with continuous transfer 452
Figure 19-10. Timing diagram of TI slave mode 453
Figure 19-11. Timing diagram of NSS pulse with continuous transmission 454
Figure 19-12. Timing diagram of quad write operation in Quad-SPI mode 455
Figure 19-13. Timing diagram of quad read operation in Quad-SPI mode 456
Figure 19-14. Block diagram of I2S 459
Figure 19-15. I2S Philips standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0) 461
Figure 19-16. I2S Philips standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1) 461
Figure 19-17. I2S Philips standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0) 461
Figure 19-18. I2S Philips standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1) 461
Figure 19-19. I2S Philips standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0) 461
Figure 19-20. I2S Philips standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1) 462
Figure 19-21. I2S Philips standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0) 462
Figure 19-22. I2S Philips standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1) 462
Figure 19-23. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0) 462
Figure 19-24. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1) 463
Figure 19-25. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0) 463
Figure 19-26. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1) 463
Figure 19-27. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0) 463
Figure 19-28. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1) 463

Figure 19-29. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0) 463
Figure 19-30. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1) 464
Figure 19-31. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0) 464
Figure 19-32. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1) 464
Figure 19-33. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0) 464
Figure 19-34. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1) 465
Figure 19-35. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=0)..... 465
Figure 19-36. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1)..... 465
Figure 19-37. PCM standard short frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0)..... 465
Figure 19-38. PCM standard short frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=1)..... 465
Figure 19-39. PCM standard short frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0)..... 466
Figure 19-40. PCM standard short frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=1)..... 466
Figure 19-41. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0)..... 466
Figure 19-42. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1)..... 466
Figure 19-43. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=0)..... 466
Figure 19-44. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1)..... 467
Figure 19-45. PCM standard long frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0)..... 467
Figure 19-46. PCM standard long frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=1)..... 467
Figure 19-47. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0)..... 467
Figure 19-48. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=1)..... 467
Figure 19-49. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0)..... 468
Figure 19-50. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1)..... 468
Figure 19-51. Block diagram of I2S clock generator..... 468
Figure 19-52. I2S initialization sequence 470
Figure 19-53. I2S master reception disabling sequence..... 472
Figure 20-1. The EXMC block diagram 486
Figure 20-2. EXMC memory banks..... 487
Figure 20-3. Mode 1 read access..... 491

Figure 20-4. Mode 1 write access 491
Figure 20-5. Mode A read access 492
Figure 20-6. Mode A write access 493
Figure 20-7. Mode 2/B read access 494
Figure 20-8. Mode 2 write access 495
Figure 20-9. Mode B write access 495
Figure 20-10. Mode C read access 496
Figure 20-11. Mode C write access 497
Figure 20-12. Mode D read access 498
Figure 20-13. Mode D write access 499
Figure 20-14. Multiplex mode read access 500
Figure 20-15. Multiplex mode write access 500
Figure 20-16. Read access timing diagram under async-wait signal assertion 502
Figure 20-17. Write access timing diagram under async-wait signal assertion 502
Figure 20-18. Read timing of synchronous multiplexed burst mode 504
Figure 20-19. Write timing of synchronous multiplexed burst mode 505
Figure 21-1. CAN module block diagram 513
Figure 21-2. Transmission register 515
Figure 21-3. State of transmit mailbox 516
Figure 21-4. Reception register 518
Figure 21-5. 32-bit filter 519
Figure 21-6. 16-bit filter 519
Figure 21-7. 32-bit mask mode filter 519
Figure 21-8. 16-bit mask mode filter 520
Figure 21-9. 32-bit list mode filter 520
Figure 21-10. 16-bit list mode filter 520
Figure 21-11. The bit time 523
Figure 21-12. Transmitter Delay Measurement 526
Figure 22-1. USBFS block diagram 552
Figure 22-2. Connection with host or device mode 553
Figure 22-3. Connection with OTG mode 554
Figure 22-4. State transition diagram of host port 554
Figure 22-5. HOST mode FIFO space in SRAM 559
Figure 22-6. Host mode FIFO access register mapping 559
Figure 22-7. Device mode FIFO space in SRAM 560
Figure 22-8. Device mode FIFO access register mapping 560

List of Tables

Table 1-1. The interconnection relationship of the AHB interconnect matrix 24
Table 1-2. Memory map of GD32C11x devices 27
Table 1-3. Boot modes 31
Table 2-1. GD32C11x base address and size for flash memory 35
Table 2-2. Option bytes 44
Table 3-1. Power saving mode summary 59
Table 5-1. Clock output 0 source select 74
Table 5-2. 1.2V domain voltage selected in deep-sleep mode 75
Table 7-1. NVIC exception types in Cortex®-M4 118
Table 7-2. Interrupt vector table 118
Table 7-3. EXTI source 122
Table 8-1. GPIO configuration table 126
Table 8-2. Debug interface signals 131
Table 8-3. Debug port mapping and Pin availability 132
Table 8-4. ADC0/1 external trigger routine conversion AF remapping function (1) 132
Table 8-5. TIMERx alternate function remapping 133
Table 8-6. TIMER4 alternate function remapping (1) 134
Table 8-7. USART0/1/2 alternate function remapping 134
Table 8-8. I2C0 alternate function remapping 135
Table 8-9. SPI0/SPI2/I2S alternate function remapping 135
Table 8-10. CAN0/1 alternate function remapping 136
Table 8-11. CTC alternate function remapping 136
Table 8-12. OSC32 pins configuration 136
Table 8-13. OSC pins configuration 137
Table 10-1. DMA transfer operation 161
Table 10-2. Interrupt events 164
Table 10-3. DMA0 requests for each channel 165
Table 10-4. DMA1 requests for each channel 166
Table 12-1. ADC internal input signals 181
Table 12-2. ADC input pins definition 181
Table 12-3. External trigger source for ADC0 and ADC1 187
Table 12-4. t_{CONV} timings depending on resolution 189
Table 12-5. Maximum output results for N and M combinations (grayed values indicate truncation) 190
Table 12-6. ADC sync mode table 191
Table 13-1. DAC I/O description 207
Table 13-2. DAC triggers and outputs summary 207
Table 13-3. Triggers of DAC 208
Table 14-1. Min/max FWDGT timeout period at 40 kHz (IRC40K) 221
Table 14-2. Min/max timeout value at 60 MHz (f_{PCLK1}) 227
Table 16-1. Timers (TIMERx) are divided into five sorts 239

Table 16-2. Complementary outputs controlled by parameters 256
Table 16-3. Counting direction in different quadrature decoder mode 259
Table 16-4. Examples of slave mode 262
Table 16-5. Examples of slave mode 307
Table 16-6. Examples of slave mode 341
Table 17-1. Description of USART important pins 386
Table 17-2. Configuration of stop bits 387
Table 17-3. USART interrupt requests 399
Table 18-1. Definition of I2C-bus terminology (refer to the I2C specification of Philips semiconductors) 415
Table 18-2. Event status flags 430
Table 18-3. Error flags 431
Table 19-1. SPI signal description 445
Table 19-2. Quad-SPI signal description 446
Table 19-3. NSS function in slave mode 447
Table 19-4. NSS function in master mode 448
Table 19-5. SPI operating modes 448
Table 19-6. SPI interrupt requests 459
Table 19-7. I2S bitrate calculation formulas 468
Table 19-8. Audio sampling frequency calculation formulas 469
Table 19-9. Direction of I2S interface signals for each operation mode 469
Table 19-10. I2S interrupt 474
Table 20-1. NOR flash interface signals description 487
Table 20-2. PSRAM non-muxed signal description 488
Table 20-3. EXMC bank0 supported transactions 488
Table 20-4. NOR/PSRAM controller timing parameters 489
Table 20-5. EXMC timing models 490
Table 20-6. Mode 1 related registers configuration 491
Table 20-7. Mode A related registers configuration 493
Table 20-8. Mode 2/B related registers configuration 495
Table 20-9. Mode C related registers configuration 497
Table 20-10. Mode D related registers configuration 499
Table 20-11. Related registers configuration of multiplex mode 501
Table 20-12. Timing configurations of synchronous multiplexed read mode 504
Table 20-13. Timing configurations of synchronous multiplexed write mode 505
Table 21-1. 32-bit filter number 520
Table 21-2. Filtering index 521
Table 21-3. CAN Event / Interrupt flags 528
Table 22-1. USBFS signal description 552
Table 22-2. USBFS global interrupt 565
Table 23-1. List of abbreviations used in register 625
Table 23-2. List of terms 625
Table 24-1. Revision history 627

1. System and memory architecture

The devices of GD32C11x series are 32-bit general-purpose microcontrollers based on the ARM® Cortex®-M4 processor. The ARM® Cortex®-M4 processor includes three AHB buses which are known as I-Code bus, D-Code bus and System bus. All memory accesses of the ARM® Cortex®-M4 processor are executed on the three buses according to the different purposes and the target memory spaces. The memory organization uses a Harvard architecture, pre-defined memory map and up to 4 GB of memory space, makes the system more flexible and extendable.

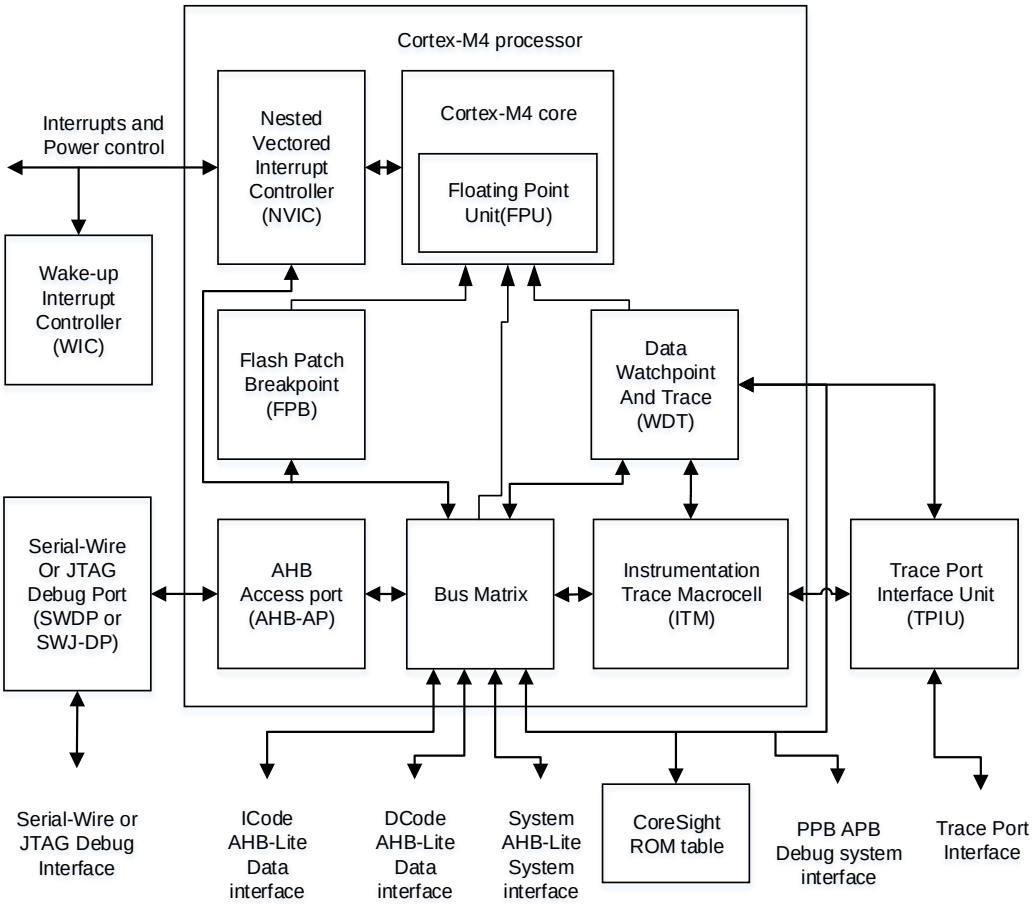
1.1. ARM Cortex-M4 processor

The Cortex®-M4 processor is a 32-bit processor that possesses floating point arithmetic functionality, low interrupt latency and low-cost debug. The characteristics of integrated and advanced make the Cortex®-M4 processor more suitable for products on the market that require the microcontrollers to have high performance and low power consumption. The Cortex®-M4 processor is based on the ARMv7 architecture and supports a powerful and scalable instruction set that includes general data processing I/O control tasks instructions, advanced data processing bit field manipulations instructions, DSP and floating point instructions. Some system peripherals listed below are also provided by Cortex®-M4:

- Internal Bus Matrix connected with I-Code bus, D-Code bus, System bus, Private Peripheral Bus (PPB) and debug accesses.
- Nested Vectored Interrupt Controller (NVIC)
- Flash Patch and Breakpoint (FPB)
- Data Watchpoint and Trace (DWT)
- Instrumentation Trace Macrocell (ITM)
- Serial Wire JTAG Debug Port (SWJ-DP)
- Trace Port Interface Unit (TPIU)
- Floating Point Unit (FPU)

Figure 1-1. The structure of the Cortex®-M4 processor shows the block diagram of the Cortex®-M4 processor. For more information, refer to the ARM® Cortex®-M4 Technical Reference Manual.

Figure 1-1. The structure of the Cortex®-M4 processor



1.2. System architecture

A 32-bit multilayer bus is implemented in the GD32C11x devices, which makes the parallel access paths between multiple masters and slaves in the system possible. The multilayer bus consists of an AHB interconnect matrix, one AHB bus and two APB buses. The interconnection relationship of the AHB interconnect matrix is shown below. In the following table, “1” indicates the corresponding master is able to access the corresponding slave through the AHB interconnect matrix, the blank indicates the corresponding master cannot access the corresponding slave through the AHB interconnect matrix.

Table 1-1. The interconnection relationship of the AHB interconnect matrix

	IBUS	DBUS	SBUS	DMA0	DMA1
FMC-I	1				
FMC-D		1		1	1
SRAM	1	1	1	1	1
EXMC	1	1	1	1	1

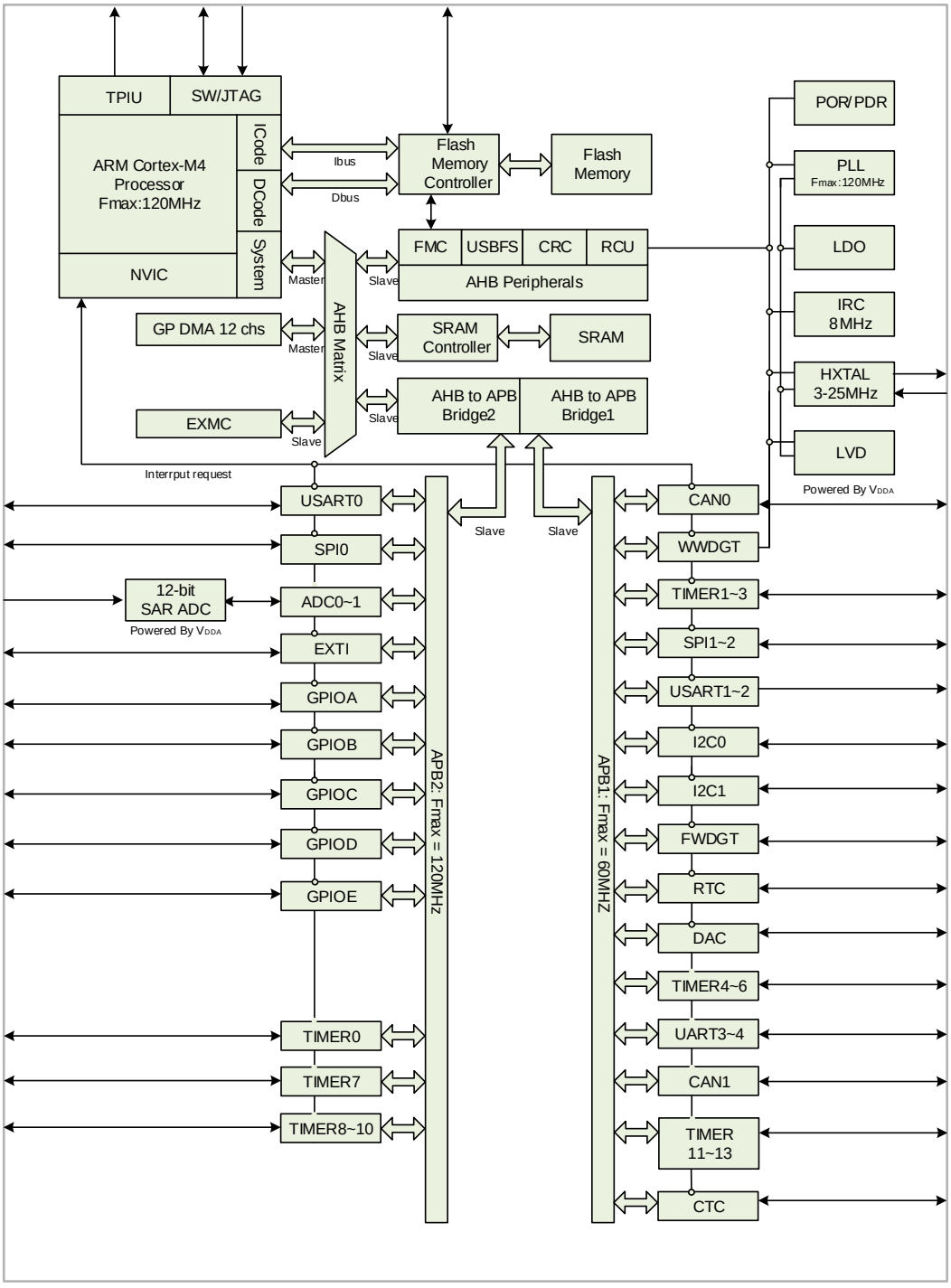
	IBUS	DBUS	SBUS	DMA0	DMA1
AHB			1	1	1
APB1			1	1	1
APB2			1	1	1

As is shown above, there are several masters connected with the AHB interconnect matrix, including IBUS, DBUS, SBUS, DMA0, DMA1. IBUS is the instruction bus of the Cortex®-M4 core, which is used for fetching instruction/vector from the Code region (0x0000 0000 ~ 0x1FFF FFFF). DBUS is the data bus of the Cortex®-M4 core, which is used for loading/storing data and debugging access of the Code region. Similarly, SBUS is the system bus of the Cortex®-M4 core, which is used for fetching instruction/vector, loading/storing data and debugging access of the system regions. The System regions include the internal SRAM region and the Peripheral region. DMA0 and DMA1 are the buses of DMA0 and DMA1 respectively.

There are also several slaves connected with the AHB interconnect matrix, including FMC-I, FMC-D, SRAM, EXMC, AHB, APB1 and APB2. FMC-I is the instruction bus of the flash memory controller, FMC-D is the data bus of the flash memory controller. SRAM is on-chip static random access memories. EXMC is the external memory controller. AHB is the AHB bus connected with all AHB slaves, APB1 and APB2 connected with all APB slaves and all APB peripherals. APB1 is limited to 60 MHz, APB2 can run to full speed (up to 120MHz depending on the device).

As shown in the following figure, these are interconnected using the multilayer AHB bus architecture.

Figure 1-2. GD32C11x series system architecture



1.3. Memory map

The ARM® Cortex®-M4 processor is structured using a Harvard architecture which uses separate buses to fetch instructions and load/store data. The instruction code and data are both located in the same memory address space but in different address ranges. Program memory, data memory, registers and I/O ports are organized within the same linear 4-Gbyte address space. The maximum address range of the Cortex®-M4 is 4-Gbyte due to its 32-bit

bus address width. Additionally, a pre-defined memory map is provided by the Cortex®-M4 processor to reduce the software complexity of repeated implementation for different device vendors. In the map, some regions are used by the ARM® Cortex®-M4 system peripherals which can not be modified. However, the other regions are available to the vendors. ***Table 1-2. Memory map of GD32C11x devices*** shows the memory map of the GD32C11x series devices, including Code, SRAM, peripheral, and other pre-defined regions. Almost each peripheral is allocated 1KB of space. This allows simplifying the address decoding for each peripheral.

Table 1-2. Memory map of GD32C11x devices

Pre-defined regions	Bus	Address	Peripherals
External device	AHB3	0xA000 0000 - 0xA000 0FFF	EXMC - SWREG
External RAM		0x9000 0000 - 0x9FFF FFFF	Reserved
		0x7000 0000 - 0x8FFF FFFF	Reserved
		0x6000 0000 - 0x63FF FFFF	EXMC - NOR/PSRAM/SRAM
Peripheral	AHB1	0x5000 0000 - 0x5003 FFFF	USBFS
		0x4008 0000 - 0x4FFF FFFF	Reserved
		0x4004 0000 - 0x4007 FFFF	Reserved
		0x4002 BC00 - 0x4003 FFFF	Reserved
		0x4002 B000 - 0x4002 BBFF	Reserved
		0x4002 A000 - 0x4002 AFFF	Reserved
		0x4002 8000 - 0x4002 9FFF	Reserved
		0x4002 6800 - 0x4002 7FFF	Reserved
		0x4002 6400 - 0x4002 67FF	Reserved
		0x4002 6000 - 0x4002 63FF	Reserved
		0x4002 5000 - 0x4002 5FFF	Reserved
		0x4002 4000 - 0x4002 4FFF	Reserved
		0x4002 3C00 - 0x4002 3FFF	Reserved
		0x4002 3800 - 0x4002 3BFF	Reserved
		0x4002 3400 - 0x4002 37FF	Reserved
		0x4002 3000 - 0x4002 33FF	CRC
		0x4002 2C00 - 0x4002 2FFF	Reserved
		0x4002 2800 - 0x4002 2BFF	Reserved
		0x4002 2400 - 0x4002 27FF	Reserved
		0x4002 2000 - 0x4002 23FF	FMC
		0x4002 1C00 - 0x4002 1FFF	Reserved
		0x4002 1800 - 0x4002 1BFF	Reserved
		0x4002 1400 - 0x4002 17FF	Reserved
		0x4002 1000 - 0x4002 13FF	RCU
		0x4002 0C00 - 0x4002 0FFF	Reserved

Pre-defined regions	Bus	Address	Peripherals
		0x4002 0800 - 0x4002 0BFF	Reserved
		0x4002 0400 - 0x4002 07FF	DMA1
		0x4002 0000 - 0x4002 03FF	DMA0
		0x4001 8400 - 0x4001 FFFF	Reserved
		0x4001 8000 - 0x4001 83FF	Reserved
	APB2	0x4001 7C00 - 0x4001 7FFF	Reserved
		0x4001 7800 - 0x4001 7BFF	Reserved
		0x4001 7400 - 0x4001 77FF	Reserved
		0x4001 7000 - 0x4001 73FF	Reserved
		0x4001 6C00 - 0x4001 6FFF	Reserved
		0x4001 6800 - 0x4001 6BFF	Reserved
		0x4001 5C00 - 0x4001 67FF	Reserved
		0x4001 5800 - 0x4001 5BFF	Reserved
		0x4001 5400 - 0x4001 57FF	TIMER10
		0x4001 5000 - 0x4001 53FF	TIMER9
		0x4001 4C00 - 0x4001 4FFF	TIMER8
		0x4001 4800 - 0x4001 4BFF	Reserved
		0x4001 4400 - 0x4001 47FF	Reserved
		0x4001 4000 - 0x4001 43FF	Reserved
		0x4001 3C00 - 0x4001 3FFF	Reserved
		0x4001 3800 - 0x4001 3BFF	USART0
		0x4001 3400 - 0x4001 37FF	TIMER7
		0x4001 3000 - 0x4001 33FF	SPI0
		0x4001 2C00 - 0x4001 2FFF	TIMER0
		0x4001 2800 - 0x4001 2BFF	ADC1
		0x4001 2400 - 0x4001 27FF	ADC0
		0x4001 2000 - 0x4001 23FF	Reserved
		0x4001 1C00 - 0x4001 1FFF	Reserved
		0x4001 1800 - 0x4001 1BFF	GPIOE
		0x4001 1400 - 0x4001 17FF	GPIOD
		0x4001 1000 - 0x4001 13FF	GPIOC
		0x4001 0C00 - 0x4001 0FFF	GPIOB
		0x4001 0800 - 0x4001 0BFF	GPIOA
		0x4001 0400 - 0x4001 07FF	EXTI
		0x4001 0000 - 0x4001 03FF	AFIO
	APB1	0x4000 CC00 - 0x4000 FFFF	Reserved
		0x4000 C800 - 0x4000 CBFF	CTC
		0x4000 C400 - 0x4000 C7FF	Reserved
		0x4000 C000 - 0x4000 C3FF	Reserved
		0x4000 8000 - 0x4000 BFFF	Reserved

Pre-defined regions	Bus	Address	Peripherals
		0x4000 7C00 - 0x4000 7FFF	Reserved
		0x4000 7800 - 0x4000 7BFF	Reserved
		0x4000 7400 - 0x4000 77FF	DAC
		0x4000 7000 - 0x4000 73FF	PMU
		0x4000 6C00 - 0x4000 6FFF	BKP
		0x4000 6800 - 0x4000 6BFF	CAN1
		0x4000 6400 - 0x4000 67FF	CAN0
		0x4000 6000 - 0x4000 63FF	CAN SRAM 1K bytes
		0x4000 5C00 - 0x4000 5FFF	Reserved
		0x4000 5800 - 0x4000 5BFF	I2C1
		0x4000 5400 - 0x4000 57FF	I2C0
		0x4000 5000 - 0x4000 53FF	UART4
		0x4000 4C00 - 0x4000 4FFF	UART3
		0x4000 4800 - 0x4000 4BFF	USART2
		0x4000 4400 - 0x4000 47FF	USART1
		0x4000 4000 - 0x4000 43FF	Reserved
		0x4000 3C00 - 0x4000 3FFF	SPI2/I2S2
		0x4000 3800 - 0x4000 3BFF	SPI1/I2S1
		0x4000 3400 - 0x4000 37FF	Reserved
		0x4000 3000 - 0x4000 33FF	FWDGT
		0x4000 2C00 - 0x4000 2FFF	WWDGT
		0x4000 2800 - 0x4000 2BFF	RTC
		0x4000 2400 - 0x4000 27FF	Reserved
		0x4000 2000 - 0x4000 23FF	TIMER13
		0x4000 1C00 - 0x4000 1FFF	TIMER12
		0x4000 1800 - 0x4000 1BFF	TIMER11
		0x4000 1400 - 0x4000 17FF	TIMER6
		0x4000 1000 - 0x4000 13FF	TIMER5
		0x4000 0C00 - 0x4000 0FFF	TIMER4
		0x4000 0800 - 0x4000 0BFF	TIMER3
		0x4000 0400 - 0x4000 07FF	TIMER2
		0x4000 0000 - 0x4000 03FF	TIMER1
SRAM	AHB	0x2007 0000 - 0x3FFF FFFF	Reserved
		0x2006 0000 - 0x2006 FFFF	Reserved
		0x2003 0000 - 0x2005 FFFF	Reserved
		0x2002 0000 - 0x2002 FFFF	Reserved
		0x2001 C000 - 0x2001 FFFF	Reserved
		0x2001 8000 - 0x2001 BFFF	Reserved
		0x2000 8000 - 0x2001 7FFF	Reserved

Pre-defined regions	Bus	Address	Peripherals
		0x2000 0000 - 0x2000 7FFF	SRAM
Code	AHB	0x1FFF F810 - 0x1FFF FFFF	Reserved
		0x1FFF F800 - 0x1FFF F80F	Option Bytes
		0x1FFF F000 - 0x1FFF F7FF	Boot loader
		0x1FFF C010 - 0x1FFF EFFF	
		0x1FFF C000 - 0x1FFF C00F	
		0x1FFF B000 - 0x1FFF BFFF	
		0x1FFF 7A10 - 0x1FFF AFFF	Reserved
		0x1FFF 7800 - 0x1FFF 7A0F	Reserved
		0x1FFF 0000 - 0x1FFF 77FF	Reserved
		0x1FFE C010 - 0x1FFE FFFF	Reserved
		0x1FFE C000 - 0x1FFE C00F	Reserved
		0x1001 0000 - 0x1FFE BFFF	Reserved
		0x1000 0000 - 0x1000 FFFF	Reserved
		0x083C 0000 - 0x0FFF FFFF	Reserved
		0x0830 0000 - 0x083B FFFF	Reserved
		0x0810 0000 - 0x082F FFFF	Reserved
		0x0802 0000 - 0x080F FFFF	Reserved
		0x0800 0000 - 0x0801 FFFF	Main Flash
		0x0030 0000 - 0x07FF FFFF	Reserved
		0x0010 0000 - 0x002F FFFF	Aliased to Main Flash or Boot loader
		0x0002 0000 - 0x000F FFFF	
		0x0000 0000 - 0x0001 FFFF	

1.3.1. Bit-banding

In order to reduce the time required for read-modify-write operations, the Cortex®-M4 processor provides a bit-banding function to perform a single atomic bit operation. The memory map includes two bit-band regions. These occupy the SRAM and Peripherals respectively. These bit-band regions map each word in an alias region of memory to a bit in a bit-band region of memory.

A mapping formula shows how to reference each word in the alias region to a corresponding bit, or target bit, in the bit-band region. The mapping formula is:

$$bit_word_addr = bit_band_base + (byte_offset \times 32) + (bit_number \times 4) \quad (1\text{-}1)$$

where:

- bit_word_addr is the address of the word in the alias memory region that maps to the targeted bit.
- bit_band_base is the starting address of the alias region.
- byte_offset is the number of the byte in the bit-band region that contains the targeted bit.

- bit_number is the bit position (0-7) of the targeted bit.

For example, the alias word at 0x2000 401C maps to bit [7] of the bit-band byte at 0x2000 0200:

bit_word_addr = 0x2200 0000 + (0x200 * 32)+ (7 * 4)= 0x2200 401C (1-2)

Writing to address 0x2200 401C will cause bit 7 of address 0x2000 0200 change .While a read to address 0x2200 401C will return 0x01 or 0x00 according to the value of bit 7 at the SRAM address 0x2000 0200.

1.3.2. On-chip SRAM memory

The GD32C11x devices series contain up to 32 KB of on-chip SRAM which address starts at 0x2000 0000. It supports byte, half-word (16 bits), and word (32 bits) accesses.

1.3.3. On-chip flash memory overview

The devices provide high density on-chip flash memory, which is organized as follows:

- Up to 128KB of main flash memory
- Up to 18KB of information blocks for the boot loader
- Option bytes to configure the device

Refer to ***Flash memory controller (FMC)*** chapter for more details.

1.4. Boot configuration

The GD32C11x devices provide three kinds of boot sources which can be selected by the BOOT0 and BOOT1 pins. The details are shown in the following table. The value on the two pins is latched on the 4th rising edge of CK_SYS after a reset. User can select the required boot source by set the BOOT0 and BOOT1 pins after a power-on reset or a system reset. Once the two pins have been sampled, they are free and can be used for other purposes.

Table 1-3. Boot modes

Selected boot source	Boot mode selection pins	
	Boot1	Boot0
Main Flash Memory	x	0
Boot loader	0	1
On-chip SRAM	1	1

After power-on sequence or a system reset, the ARM® Cortex®-M4 processor fetches the top-of-stack value from address 0x0000 0000 and the base address of boot code from 0x0000 0004 in sequence. Then, starts code execution from the base address of boot code.

Due to the selected boot source, either the main flash memory (original memory space is beginning at 0x0800 0000) or the system memory (original memory space is beginning at

0x1FFF F000) is aliased in the boot memory space which begins at 0x0000 0000. When the on-chip SRAM whose memory space is beginning at 0x2000 0000 is selected as the boot source, in the application initialization code, you have to relocate the vector table in SRAM by using the NVIC exception table and offset register.

The embedded boot loader is located in the system memory, which is used to reprogram the Flash memory. In GD32C11x devices, the boot loader can be activated through the USART0 interface.

1.5. Device electronic signature

The device electronic signature contains memory size information and the 96-bit unique device ID. It is stored in the information block of the Flash memory. The 96-bit unique device ID is unique for each device. It can be used as serial numbers, or part of security keys, etc.

1.5.1. Memory density information

Base address: 0x1FFF F7E0

The value is factory programmed and can never be altered by user.

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

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

Bits	Fields	Descriptions
31:16	SRAM_DENSITY[15:0]	SRAM density The value indicates the on-chip SRAM density of the device in Kbytes. Example: 0x0008 indicates 8 Kbytes.
15:0	FLASH_DENSITY[15:0]	Flash memory density The value indicates the flash memory density of the device in Kbytes. Example: 0x0020 indicates 32 Kbytes.

1.5.2. Unique device ID (96 bits)

Base address: 0x1FFF F7E8

The value is factory programmed and can never be altered by user.

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

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

Bits	Fields	Descriptions
31:0	UNIQUE_ID[31:0]	Unique device ID

Base address: 0x1FFF F7EC

The value is factory programmed and can never be altered by user.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UNIQUE_ID[63:48]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNIQUE_ID[47:32]															
r															

Bits	Fields	Descriptions

31:0	UNIQUE_ID[63:32]	Unique device ID

Base address: 0x1FFF F7F0

The value is factory programmed and can never be altered by user.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UNIQUE_ID[95:80]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNIQUE_ID[79:64]															
r															

Bits	Fields	Descriptions
31:0	UNIQUE_ID[95:64]	Unique device ID

1.6. System configuration registers

Base address: 0x4002 103C

Reset value: 0x0000 0000

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

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

Bits	Fields	Descriptions
7	CEE	Code execution efficiency 0：Default code execution efficiency 1：Code execution efficiency reduce

NOTE:

1. Only bit[7] can be read-modify-write, other bits are not permitted.

2. Flash memory controller (FMC)

2.1. Overview

The flash memory controller, FMC, provides all the necessary functions for the on-chip flash memory. A little waiting time is needed while CPU executes instructions stored from the 128K bytes of the flash. It also provides page erase, mass erase, and program operations for flash memory.

2.2. Characteristics

- Up to 128KB of on-chip flash memory for instruction and data.
- 0~3 waiting time within 128K bytes when CPU executes instructions.
- Flash memory supports 64-bit wide data reading.
- Pre-fetch buffer to speed read operations.
- IBUS cache with 512 bytes which organized as 32 cache line of 2 X 64 bits.
- DBUS cache with 256 bytes which organized as 8 cache line of 4 X 64 bits.
- The flash page size is 1KB
- Word/double-half-word programming, page erase and mass erase operation.
- 512B OTP(One-time program) block used for user data storage.
- 16B option bytes block for user application requirements.
- Option bytes are uploaded to the option byte control registers when the system is reset.
- Flash security protection to prevent illegal code/data access.
- Page erase/program protection to prevent unexpected operation.

2.3. Function overview

2.3.1. Flash memory architecture

The page size is 1 KB. Each page can be erased individually.

The following table shows the details of flash organization.

Table 2-1. GD32C11x base address and size for flash memory

Block	Name	Address range	size(bytes)
Main Flash Block	Page 0	0x0800 0000 - 0x0800 03FF	1KB
	Page 1	0x0800 0400 - 0x0800 07FF	1KB
	Page 2	0x0800 0800 - 0x0800 0BFF	1KB
	. . .	. . .	. . .
	Page 127	0x0801 FC00 - 0x0801 FFFF	1KB

Block	Name	Address range	size(bytes)
Information Block	Boot Loader area	0x1FFF B000- 0x1FFF F7FF	18KB
Option bytes Block	Option bytes	0x1FFF F800 - 0x1FFF F80F	16B
One-time program Block	OTP bytes	0x1FFF_7000~0x1FFF_71FF	512B

NOTE: The Information Block stores the boot loader. This block cannot be programmed or erased by user.

2.3.2. Read operations

The flash can be addressed directly as a common memory space. Any instruction fetch and the data access from the flash are through the IBUS or DBUS from the CPU.

Wait state added:

Must configure the WSCNT bits in the FMC_WS register correctly depend on the AHB clock frequency. The relation between WSCNT and AHB clock frequency is show as the following table.

Table 2-2. The relation between WSCNT and AHB clock frequency

AHB clock frequency	WSCNT configured
<= 30MHz	0 (0 wait state added)
<= 60MHz	1 (1 wait state added)
<= 90MHz	2 (2 wait state added)
<= 120MHz	3 (3 wait state added)

If system reset occurs, the AHB clock frequency is 8MHz and the WSCNT is 0.

Note:

1. If want to increase the AHB clock frequency. First, refer to the Table 2-2, configure the WSCNT bits according to the target AHB clock frequency. Then, increase the AHB clock frequency to the target frequency. It is forbidden to increase the AHB clock frequency before configure the WSCNT.

2. If want to decrease the AHB clock frequency. First, decrease the target AHB clock frequency. Then refer to the Table 2-2, configure the WSCNT bits according the target AHB clock frequency. It is forbidden to configure the WSCNT bits before decrease the AHB clock frequency.

Because the wait state is added, the read efficiency is very low (such as add 3 wait state when 120MHz). In order to speed up the read access, there are some functions performed.

Current buffer:

The current buffer is always enabled. Each time read from flash memory, 64-bit data get and

store in current buffer. The CPU only need 32-bit or 16-bit in each read operation. So in the case of sequential code, the next data can get from current buffer without repeat fetch from flash memory.

Pre-fetch buffer:

The pre-fetch buffer is enabled by set the PFEN bit in the FMC_WS register. The pre-fetch buffer is only performed on IBUS. In the case of sequential code, when CPU execute the current buffer data (64-bit), 32-bit needs at least 2 clocks and 16-bit needs at least 4 clocks. In this case, pre-fetch the data of next double-word address from flash memory and store to Pre-fetch buffer. So when the CPU finish the current buffer and need execute the next data, the pre-fetch buffer hit.

IBUS Cache:

IBUS cache is enabled by set the ICEN bit in the FMC_WS register. The IBUS cache is only used when IBUS fetch data. The IBUS cache have 512 bytes which organized as 32 cache lines, each cache lines is 2 X 64bits.

If the IBUS data is in IBUS cache (IBUS cache hit), the CPU read data from IBUS cache without any wait state. If the IBUS data is not in IBUS cache (IBUS cache miss) and not in current buffer/Pre-fetch buffer, the cache line fetch from flash memory and copied to IBUS cache. If all cache line filled, LRU (least recently used) policy used to replace the cache line.

DBUS Cache:

DBUS cache is enabled by set the DCEN bit in the FMC_WS register. The DBUS cache is only used when DBUS fetch data by CPU (not by DMA). And the option byte is not cacheable. The DBUS cache have 256 bytes which organized as 8 cache lines, each cache lines is 4 X 64bits.

If the DBUS data is in DBUS cache (DBUS cache hit), the CPU read data from DBUS cache without any wait state. If the DBUS data is not in DBUS cache (DBUS cache miss) and not in current buffer, the cache line fetch from flash memory and copied to DBUS cache. If all cache line filled, LRU (least recently used) policy used to replace the cache line.

2.3.3. Unlock the FMC_CTL register

After reset, the FMC_CTL register is not accessible in write mode, and the LK bit in the FMC_CTL register is reset to 1. An unlocking sequence consists of two write operations to the FMC_KEY register to open the access to the FMC_CTL register. The two write operations are writing 0x45670123 and 0xCDEF89AB to the FMC_KEY register. After the two write operations, the LK bit in the FMC_CTL register is reset to 0 by hardware. The software can lock the FMC_CTL again by setting the LK bit in the FMC_CTL register to 1. Any wrong operations to the FMC_KEY, will set the LK bit to 1, and lock the FMC_CTL register, and lead to a bus error.

The OBPG bit and OBER bit in the FMC_CTL are still protected even the FMC_CTL is unlocked. The unlocking sequence consists of two write operations, which are writing 0x45670123 and 0xCDEF89AB to the FMC_OBKEY register. Then the hardware sets the OBWEN bit in the FMC_CTL register to 1. The software can reset OBWEN bit to 0 to protect the OBPG bit and OBER bit in the FMC_CTL register again.

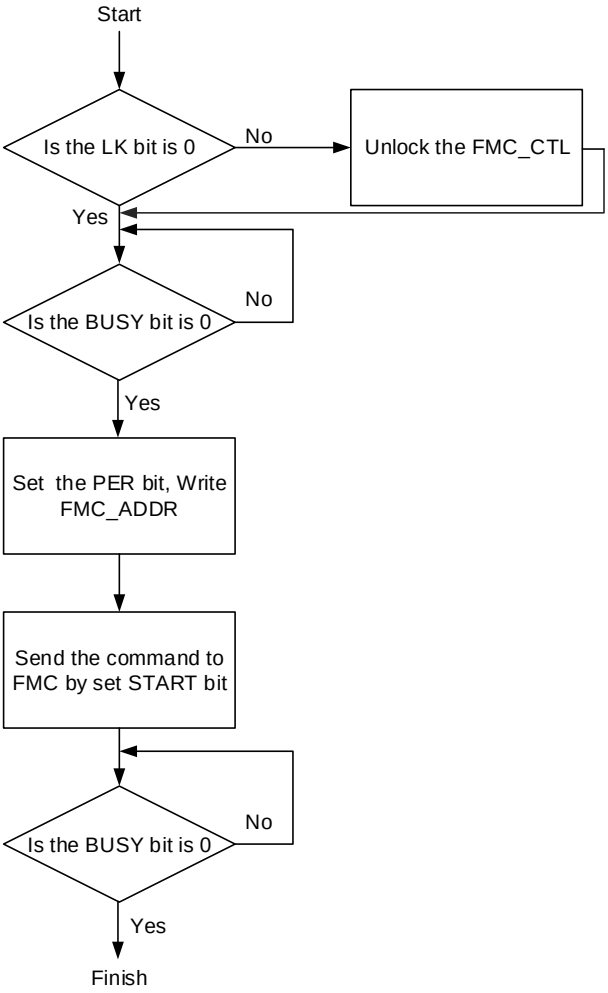
2.3.4. Page erase

The FMC provides a page erase function which is used to initialize the contents of a main flash memory page to a high state. Each page can be erased independently without affecting the contents of other pages. The following steps show the access sequence of the registers for a page erase operation.

- Unlock the FMC_CTL register if necessary.
- Check the BUSY bit in the FMC_STAT register to confirm that no flash memory operation is in progress (BUSY equals to 0). Otherwise, wait until the operation has finished.
- Set the PER bit in the FMC_CTL register.
- Write the page absolute address (0x08XX XXXX) into the FMC_ADDR registers.
- Send the page erase command to the FMC by setting the START bit in the FMC_CTL register.
- Wait until all the operations have finished by checking the value of the BUSY bit in the FMC_STAT register.
- Read and verify the page if required using a DBUS access.

When the operation is executed successfully, the ENDF bit in the FMC_STAT register is set, and an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set. Note that a correct target page address must be confirmed. Otherwise, the software may run out of control if the target erase page is being used to fetch codes or access data. The FMC will not provide any notification when that happens. Additionally, the page erase operation will be ignored on erase/program protected pages. In this condition, a flash operation error interrupt will be triggered by the FMC if the ERRIE bit in the FMC_CTL register is set. The software can check the WPERR bit in the FMC_STAT register to detect this condition in the interrupt handler. The following figure shows the page erase operation flow.

Figure 2-1. Process of page erase operation



2.3.5. Mass erase

The FMC provides a complete erase function which is used to initialize the main flash block contents. This erase can affect entire flash block by setting the MER bit to 1 in the FMC_CTL register. The following steps show the mass erase register access sequence.

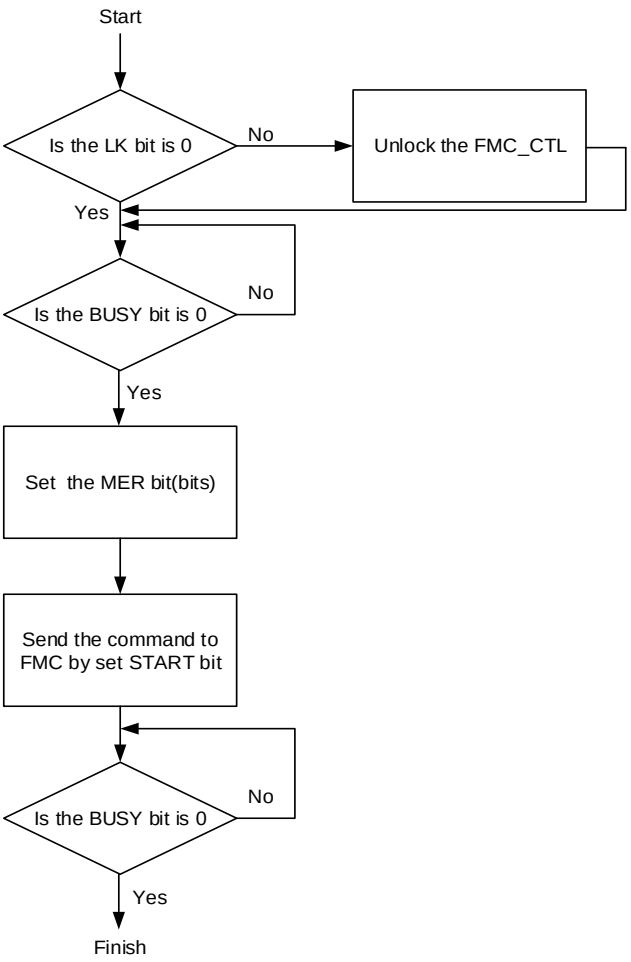
- Unlock the FMC_CTL register if necessary.
- Check the BUSY bit in the FMC_STAT register to confirm that no flash memory operation is in progress (BUSY equals to 0). Otherwise, wait until the operation has finished.
- Set the MER bit in the FMC_CTL register if erase entire flash.
- Send the mass erase command to the FMC by setting the START bit in the FMC_CTL register.
- Wait until all the operations have been finished by checking the value of the BUSY bit in the FMC_STAT register.
- Read and verify the flash memory if required using a DBUS access.

When the operation is executed successfully, the ENDF bit in the FMC_STAT register is set,

and an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set. Since all flash data will be modified to a value of 0xFFFF_FFFF, the mass erase operation can be implemented using a program that runs in SRAM or using the debugging tool that accesses the FMC registers directly. Additionally, the mass erase operation will be ignored if any page is erase/program protected. In this condition, a flash operation error interrupt will be triggered by the FMC if the ERRIE bit in the FMC_CTL register is set. The software can check the WPERR bit in the FMC_STAT register to detect this condition in the interrupt handler.

The following figure indicates the mass erase operation flow.

Figure 2-2. Process of mass erase operation



2.3.6. Main flash programming

The FMC provides a 32-bit word/16-bit half word programming function by DBUS which is used to modify the main flash memory contents. While actually, the data program to flash memory is 32-bits or 64-bits which is defined by the PGW bit in the FMC_WS register.

The following steps show the register access sequence of the programming operation.

- Unlock the FMC_CTL register if necessary.
- Check the BUSY bit in the FMC_STAT register to confirm that no flash memory operation is in progress (BUSY equals to 0). Otherwise, wait until the operation has finished.
- Set the PGW bit if needed.
- Set the PG bit in the FMC_CTL register.
- Write the data to be programed by DBUS with desired absolute address (0x08XX XXXX).
  If DBUS program is 32-bit word and the PGW bit is set to 0(32-bit program to flash memory), the DBUS write once and the data program to flash memory. The data to be programed must word alignment.
  If DBUS program is 32-bit and the PGW bit is set to 1(64-bit program to flash memory), the DBUS write twice to form a 64-bit data and then the 64-bit data program to flash memory. The data to be programed must double-word alignment.
  If DBUS program is 16-bit and the PGW bit is set to 0(32-bit program to flash memory), the DBUS write twice to form a 32-bit data and then the 32-bit data program to flash memory. The data to be programed must word alignment.
  If DBUS program is 16-bit and the PGW bit is set to 1(64-bit program to flash memory), the DBUS write four times to form a 64-bit data and then the 64-bit data program to flash memory. The data to be programed must double-word alignment.
  For less program time, suggest the DBUS program use 32-bit, set the PGW to 1 if the data to be programed is double-word alignment, or set PGW to 0 if the data to be programed is word alignment
- Wait until all the operations have been finished by checking the value of the BUSY bit in the FMC_STAT register.
- Read and verify the Flash memory if required using a DBUS access.

When the operation is executed successfully, the ENDF bit in the FMC_STAT register is set, and an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set. Note that there are some program error need caution:

The programming operation checks the address if it has been erased or not. If the address has not been erased, the PGERR bit in the FMC_STAT register will be set even if programming 0x0. Each word can be programmed only one time after erase and before next erase Note that the PG bit must be set before the word/half word programming operation.

Additionally, the program operation will be ignored on erase/program protected pages and the WPERR bit in the FMC_STAT will be set.

In the following cases, the PGAERR bit in the FMC_STAT register will be set.

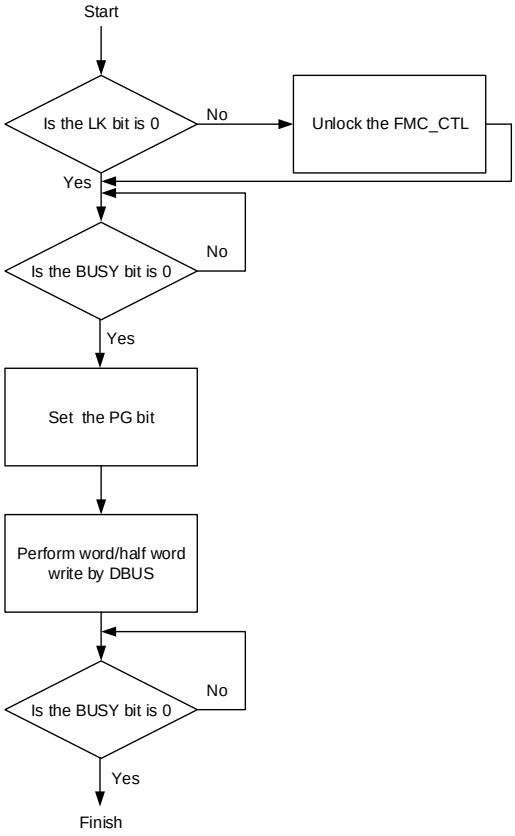
- The DBUS program use byte write (not 32-bit or 16-bit write)
- The DBUS program size is not equal previous size. It not allow mix 32-bit with 16-bit write.
- The DBUS write is not alignment. If DBUS program is 32-bit and the PGW bit is set to 1(64-bit program to flash memory), the second DBUS write must double-word alignment and belong to same double-word address. If DBUS program is 16-bit and the PGW bit is set to 0(32-bit program to flash memory), the second DBUS write must word alignment and belong to same word address. If DBUS program is 16-bit and the PGW bit is set to 1(64-bit program to flash memory), the 2nd/3rd/4th DBUS write must double-word alignment

and belong to same double-word address.

Note: If the program is not write total 64bits/32bits (by set the PGW bit in the FMC_WS register), the data is not program to the flash memory without any notice.

In these conditions, a flash operation error interrupt will be triggered by the FMC if the ERRIE bit in the FMC_CTL register is set. The software can check the PGERR bit, PGAERR bit or WPERR bit in the FMC_STAT register to detect which condition occurred in the interrupt handler. The following figure shows the word programming operation flow.

Figure 2-3. Process of word program operation



Note: Reading the flash should be avoided when a program/erase operation is ongoing in the same bank. And flash memory accesses will fail if the CPU enters the power saving modes.

2.3.7. OTP programming

The OTP programming method is same as the main flash programming. The OTP block can only be programed once and cannot be erased.

Note: It must ensure the OTP programming sequence completely without any unexpected interrupt, such as system reset or power down. If unexpected interrupt occurs, there is very little probability of corrupt the data stored in flash memory.

2.3.8. Option bytes Erase

The FMC provides an erase function which is used to initialize the option bytes block in flash. The following steps show the erase sequence.

- Unlock the FMC_CTL register if necessary.
- Check the BUSY bit in the FMC_STAT register to confirm that no Flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has finished.
- Unlock the option bytes operation bits in the FMC_CTL register if necessary.
- Wait until the OBWEN bit is set in the FMC_CTL register.
- Set the OBER bit in the FMC_CTL register.
- Send the option bytes erase command to the FMC by setting the START bit in the FMC_CTL register.
- Wait until all the operations have been finished by checking the value of the BUSY bit in the FMC_STAT register.
- Read and verify the Flash memory if required using a DBUS access.

When the operation is executed successful, the ENDF bit in the FMC_STAT register is set, and an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set.

2.3.9. Option bytes modify

The FMC provides an erase / program function which is used to modify the option bytes block in flash. There are 8 pairs of option bytes. The MSB is the complement of the LSB in each pair. When the option bytes are modified, the MSB is generated by FMC automatically, not the value of input data. The following steps show the erase sequence.

- Unlock the FMC_CTL register if necessary.
- Check the BUSY bit in the FMC_STAT register to confirm that no Flash memory operation is in progress (BUSY equals to 0). Otherwise, wait until the operation has finished.
- Unlock the option bytes operation bits in the FMC_CTL register if necessary.
- Wait until the OBWEN bit is set in the FMC_CTL register.
- Set the OBPG bit in the FMC_CTL register.
- A 32-bit word/16-bit half word write at desired address by DBUS. There need write once, twice or fourth depend on the DBUS write size and the PGW bit in the FMC_WS register. The write method is similar to main flash programming.
- Wait until all the operations have been finished by checking the value of the BUSY bit in the FMC_STAT register.
- Read and verify the Flash memory if required using a DBUS access.

When the operation is executed successfully, the ENDF bit in the FMC_STAT register is set, and an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set. Note that there programming errors may occur. The PGERR bit and PGAERR bit can be set which is similar to main flash programming.

The modified option bytes only take effect after a system reset.

2.3.10. Option bytes description

The option bytes block is reloaded to the FMC_OBSTAT and FMC_WP registers after each system reset, then the option bytes take effect. The complement option bytes are the opposite of the option bytes. When reload the option bytes, if the complement option byte and option byte do not match, the OBERR bit in the FMC_OBSTAT register will be set, and the option byte will be set to 0xFF. The OBERR bit will not be set if both the option bytes and its complement bytes are 0xFF.The following table shows the detail of option bytes.

Table 2-2. Option bytes

Address	Name	Description
0x1fff f800	SPC	option bytes Security Protection value 0xA5 : no security protection any value except 0xA5 : under security protection
0x1fff f801	SPC_N	SPC complement value
0x1fff f802	USER	[7:3]: reserved [2]: nRST_STDBY 0: generator a reset instead of entering standby mode 1: no reset when entering standby mode [1]: nRST_DPSLP 0: generator a reset instead of entering Deep-sleep mode 1: no reset when entering Deep-sleep mode [0]: nWDG_HW 0: hardware free watchdog 1: software free watchdog
0x1fff f803	USER_N	USER complement value
0x1fff f804	DATA[7:0]	user defined data bit 7 to 0
0x1fff f805	DATA_N[7:0]	DATA complement value bit 7 to 0
0x1fff f806	DATA[15:8]	user defined data bit 15 to 8
0x1fff f807	DATA_N[15:8]	DATA complement value bit 15 to 8
0x1fff f808	WP[7:0]	Page Erase/Program Protection bit 7 to 0 0: protection active 1: unprotected
0x1fff f809	WP_N[7:0]	WP complement value bit 7 to 0
0x1fff f80a	WP[15:8]	Page Erase/Program Protection bit 15 to 8
0x1fff f80b	WP_N[15:8]	WP complement value bit 15 to 8
0x1fff f80c	WP[23:16]	Page Erase/Program Protection bit 23 to 16
0x1fff f80d	WP_N[23:16]	WP complement value bit 23 to 16
0x1fff f80e	WP[31:24]	Page Erase/Program Protection bit 31 to 24 WP[30:24]: Each bit is related to 4KB flash protection. These bits totally controls the first 124KB flash protection. WP[31]: Bit 31 controls the protection of the rest flash

Address	Name	Description
		memory.
0x1fff f80f	WP_N[31:24]	WP complement value bit 31 to 24

2.3.11. Page erase/program protection

The FMC provides page erase/program protection functions to prevent inadvertent operations on the Flash memory. The page erase or program will not be accepted by the FMC on protected pages. If the page erase or program command is sent to the FMC on a protected page, the WPERR bit in the FMC_STAT register will be set by the FMC. If the WPERR bit is set and the ERRIE bit is also set to 1 to enable the corresponding interrupt, then the Flash operation error interrupt will be triggered by the FMC to draw the attention of the CPU. The page protection function can be individually enabled by configuring the WP [31:0] bit field to 0 in the option bytes. If a page erase operation is executed on the option bytes block, all the Flash Memory page protection functions will be disabled. When WP in the option bytes is modified, then a system reset is necessary.

2.3.12. Security protection

The FMC provides a security protection function to prevent illegal code/data access to the Flash memory. This function is useful for protecting the software/firmware from illegal users.

No protection: when setting SPC byte and its complement value to 0x5AA5, no protection performed after power reset. The main flash and option bytes block are accessible by all operations.

Under protection: when setting SPC byte and its complement value to any value except 0x5AA5, the security protection is performed after power reset. Note that a power reset should be followed instead of a system reset if the SPC modification has been performed while the debug module is still connected to JTAG/SWD device. Under the security protection, the main flash can only be accessed by user code and the first 4KB flash is under erase/program protection. In debug mode, boot from SRAM or boot loader mode, all operations to main flash is forbidden. If a read operation to main flash in debug mode, boot from SRAM or boot loader mode, a bus error will be generated. If a program/erase operation to main flash in debug mode, boot from SRAM or boot from boot loader mode, the WPERR bit in the FMC_STAT register will be set. Option bytes block are accessible by all operations, which can be used to disable the security protection. Back to no protection level by setting SPC byte and its complement value to 0x5AA5, then a mass erase for main flash will be performed.

2.4. Register definition

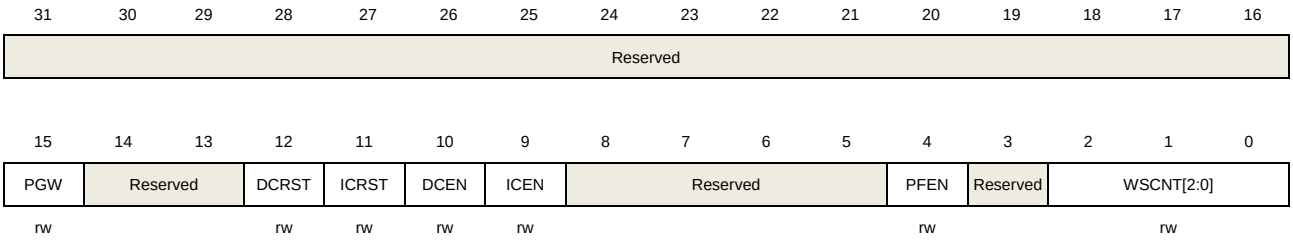
FMC base address: 0x4002 2000

2.4.1. Wait state register (FMC_WS)

Address offset: 0x00

Reset value: 0x0000 0630

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	PGW	Program width to flash memory 0: 32-bit program width to flash memory 1: 64-bit program width to flash memory
14:13	Reserved	Must be kept at reset value.
12	DCRST	DBUS cache reset. This bit can be write only when DCEN is set to 0. 0: No effect 1: DBUS cache reset
11	ICRST	IBUS cache reset. This bit can be write only when ICEN is set to 0. 0: No effect 1: IBUS cache reset
10	DCEN	DBUS cache enable 0: DBUS cache disable 1: DBUS cache enable
9	ICEN	IBUS cache enable 0: IBUS cache disable 1: IBUS cache enable
8:5	Reserved	Must be kept at reset value.
4	PFEN	Pre-fetch enable 0: Pre-fetch disable 1: Pre-fetch enable
3	Reserved	Must be kept at reset value.
2:0	WSCNT[2:0]	Wait state counter register These bits is set and reset by software. 000: 0 wait state added

001: 1 wait state added
010: 2 wait state added
011: 3 wait state added
100 ~111:reserved

2.4.2. Unlock key register (FMC_KEY)

Address offset: 0x04
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

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

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

Bits	Fields	Descriptions
31:0	KEY[31:0]	FMC_CTL unlock register These bits are only be written by software. Write KEY[31:0] with keys to unlock FMC_CTL register.

2.4.3. Option byte unlock key register (FMC_OBKEY)

Address offset: 0x08
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

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

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

Bits	Fields	Descriptions
31:0	OBKEY[31:0]	FMC_CTL option bytes operation unlock register These bits are only be written by software. Write OBKEY[31:0] with keys to unlock option bytes command in the FMC_CTL register.

2.4.4. Status register (FMC_STAT)

Address offset: 0x0C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										ENDF	WPERR	PGAERR	PGERR	Reserved	BUSY
										rc_w1	rc_w1	rc_w1	rc_w1		r

Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5	ENDF	End of operation flag bit When the operation executed successfully, this bit is set by hardware. The software can clear it by writing 1.
4	WPERR	Erase/Program protection error flag bit When erase/program on protected pages, this bit is set by hardware. The software can clear it by writing 1.
3	PGAERR	Program alignment error flag bit This bit is set by hardware when DBUS write data is not alignment. The software can clear it by writing 1.
2	PGERR	Program error flag bit When program to the flash while it is not 0xFFFF, this bit is set by hardware. The software can clear it by writing 1.
1	Reserved	Must be kept at reset value.
0	BUSY	The flash is busy bit When the operation is in progress, this bit is set to 1. When the operation is end or an error is generated, this bit is cleared to 0.

2.4.5. Control register (FMC_CTL)

Address offset: 0x10
Reset value: 0x0000 0080

This register has to be accessed by word (32-bit)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ENDIE	Reserved	ERRIE	OBWEN	Reserved	LK	START	OBER	OBPG	Reserved	MER	PER	PG
			rw		rw	rw		rs	rs	rw	rw		rw	rw	rw

Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12	ENDIE	End of operation interrupt enable bit This bit is set or cleared by software 0: no interrupt generated by hardware. 1: end of operation interrupt enable
11	Reserved	Must be kept at reset value.
10	ERRIE	Error interrupt enable bit This bit is set or cleared by software 0: no interrupt generated by hardware. 1: error interrupt enable
9	OBWEN	Option byte erase/program enable bit This bit is set by hardware when right sequence written to the FMC_OBKEY register. This bit can be cleared by software.
8	Reserved	Must be kept at reset value.
7	LK	FMC_CTL lock bit This bit is cleared by hardware when right sequence written to the FMC_KEY register. This bit can be set by software.
6	START	Send erase command to FMC bit This bit is set by software to send erase command to FMC. This bit is cleared by hardware when the BUSY bit is cleared.
5	OBER	Option bytes erase command bit This bit is set or clear by software 0: no effect 1: option byte erase command
4	OBPG	Option bytes program command bit This bit is set or clear by software 0: no effect 1: option bytes program command
3	Reserved	Must be kept at reset value.
2	MER	Main flash mass erase command bit This bit is set or cleared by software 0: no effect 1: main flash mass erase command

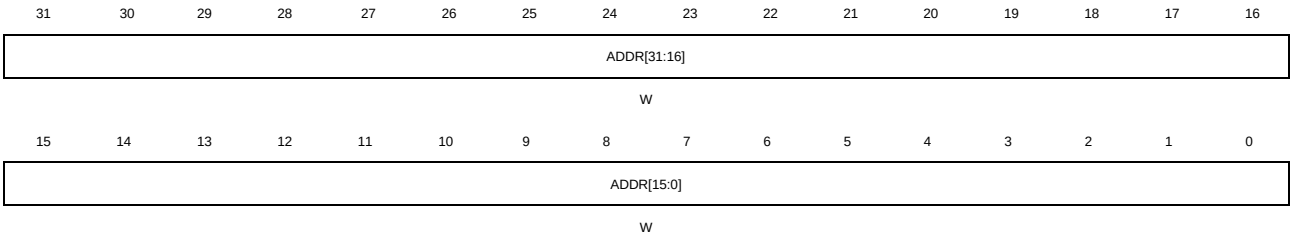
1	PER	Main flash page erase command bit This bit is set or clear by software 0: no effect 1: main flash page erase command
0	PG	Main flash program command bit This bit is set or clear by software 0: no effect 1: main flash program command

Note: This register should be reset after the corresponding flash operation completed.

2.4.6. Address register (FMC_ADDR)

Address offset: 0x14
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:0	ADDR[31:0]	Flash erase/program command address bits These bits are configured by software. ADDR bits are the address of flash to be erased/programmed.

2.4.7. Option byte status register (FMC_OBSTAT)

Address offset: 0x1C
Reset value: 0x0XXX XXXX.

This register has to be accessed by word(32-bit)



Bits	Fields	Descriptions

31:26	Reserved	Must be kept at reset value.
25:10	DATA[15:0]	Store DATA[15:0] of option bytes block after system reset.
9:2	USER[7:0]	Store USER of option bytes block after system reset.
1	SPC	Option bytes security protection code 0: no protection 1: protection
0	OBERR	Option bytes read error bit. This bit is set by hardware when the option bytes and its complement byte do not match, then the option bytes is set to 0xFF.

2.4.8. Erase/Program protection register (FMC_WP)

Address offset: 0x20

Reset value: 0xFFFF FFFF

This register has to be accessed by word(32-bit)

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

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

Bits	Fields	Descriptions
31:0	WP[31:0]	Store WP[31:0] of option bytes block after system reset

2.4.9. Product ID register (FMC_PID)

Address offset: 0x100

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit)

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

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

Bits	Field	Descriptions
31:0	PID[31:0]	Product reserved ID code register These bits are read only by software.

These bits are unchanged constant after power on. These bits are one time program when the chip produced.

3. Power management unit (PMU)

3.1. Overview

The power consumption is regarded as one of the most important issues for the devices of GD32C11x series. Power management unit (PMU) provides three types of power saving modes, including Sleep, Deep-sleep and Standby mode. These modes reduce the power consumption and allow the application to achieve a best tradeoff among the conflicting demands of CPU operating time, speed and power consumption. For GD32C11x devices, there are three power domains, including V_{DD} / V_{DDA} domain, V_{CORE} domain and backup domain, as is shown in the ***Figure 3-1. Power supply overview***. The power of the V_{DD} domain is supplied directly by V_{DD}. An embedded LDO in the V_{DD}/V_{DDA} domain is used to supply the V_{CORE} domain power. A power switch is implemented for the backup domain. It can be powered from the VBAT when the main VDD supply is shut down.

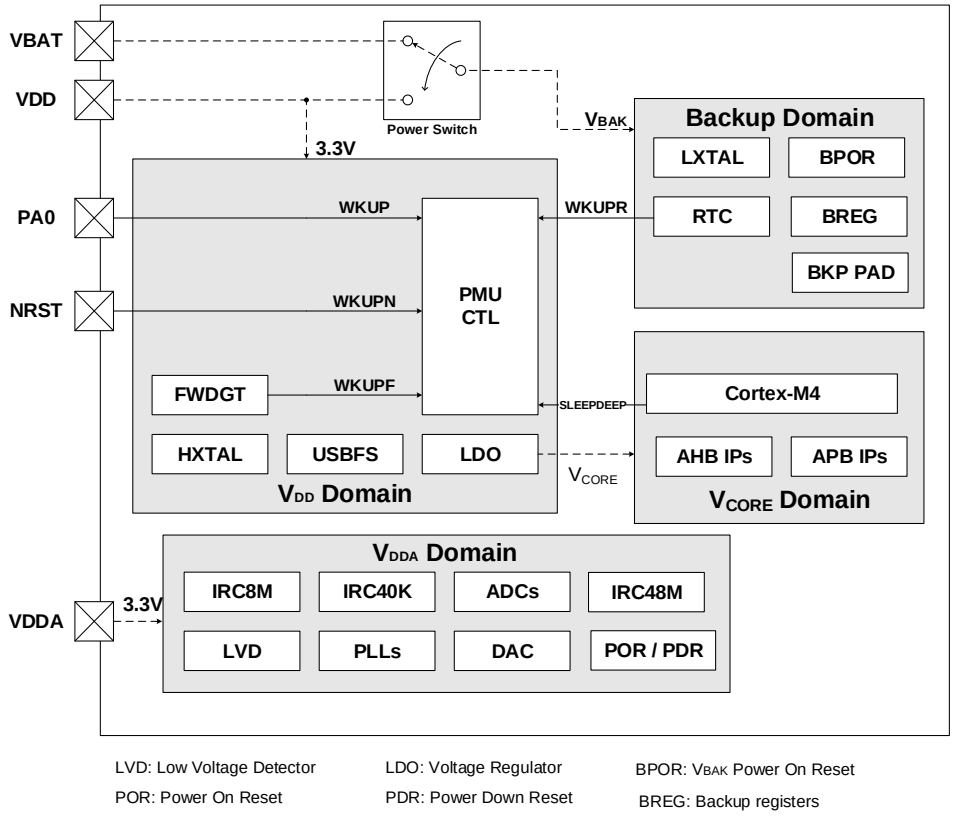
3.2. Characteristics

- Three power domains: backup domain, V_{DD} / V_{DDA} domain and V_{CORE} domain.
- Three power saving modes: Sleep, Deep-sleep and Standby modes.
- Internal Voltage regulator (LDO) supplies around V_{CORE} voltage source for V_{CORE} domain.
- Low Voltage Detector (LVD) can issue an interrupt or event when the power is lower than a programmed threshold.
- Battery power (VBAT) for backup domain when VDD is shut down.
- LDO output voltage is selected for power saving.

3.3. Function overview

Figure 3-1. Power supply overview provides details on the internal configuration of the PMU and the relevant power domains.

Figure 3-1. Power supply overview



3.3.1. Backup domain

The Backup domain is powered by the VDD or the battery power source (VBAT) selected by the internal power switch. The VBAK pin which drives backup domain, supplies power for RTC unit, LXTAL oscillator, BPOR, BREG and three BKP PAD including PC13 to PC15. In order to ensure the content of the backup domain registers and the RTC supply, when VDD supply is shut down, VBAT pin can be connected to an optional standby voltage supplied by a battery or by another source. The power switch is controlled by the Power Down Reset (PDR) circuit in the VDD / VDDA domain. If no external battery is used in the application, it is recommended to connect VBAT pin externally to VDD pin with a 100nF external ceramic decoupling capacitor.

The Backup domain reset sources include the backup domain Power On Reset (BPOR) and the Backup domain software reset. The BPOR signal forces the device to stay in the reset mode until V_{BAK} is completely powered up. Also the application software can trigger the Backup domain software reset by setting the BKPRST bit in the RCU_BDCTL register to reset the Backup domain.

The clock source of the Real Time Clock (RTC) circuit can be derived from the Internal 40KHz RC oscillator (IRC40K) or the Low Speed Crystal oscillator (LXTAL), or HXTAL clock divided by 128. When V_{DD} is shut down, only LXTAL is valid for RTC. Before entering the power saving mode by executing the WFI / WFE instruction, the Cortex®-M4 can setup the RTC register with an expected alarm time and enable the alarm function to achieve the RTC alarm

event. After entering the power saving mode for a certain amount of time, the RTC will wake up the device when the time match event occurs. The details of the RTC configuration and operation will be described in the ***Real-time Clock (RTC)***.

When the backup domain is supplied by VDD (V_{BAK} pin is connected to V_{DD}), the following functions are available:

- PC13 can be used as GPIO or RTC function pin described in the ***Real-time Clock (RTC)***.
- PC14 and PC15 can be used as either GPIO or LXTAL crystal oscillator pins.

When the Backup domain is supplied by VBAT (V_{BAK} pin is connected to V_{BAT}), the following functions are available:

- PC13 can be used as RTC function pin described in the ***Real-time Clock (RTC)***.
- PC14 and PC15 can be used as LXTAL crystal oscillator pins only.

Note: Since PC13, PC14 and PC15 are supplied by the Power Switch which can only be passed by low current, the speed of GPIOs PC13 to PC15 should not exceed 2MHz when they are in output mode (maximum load: 30pF).

3.3.2. V_{DD} / V_{DDA} power domain

V_{DD} / V_{DDA} domain includes two parts: V_{DD} domain and V_{DDA} domain. V_{DD} domain includes HXTAL (High Speed Crystal oscillator), LDO (Voltage Regulator), USBFS (Universal serial bus full-speed interface), FWDGT (Free Watchdog Timer), all pads except PC13 / PC14 / PC15, etc. V_{DDA} domain includes ADC / DAC (AD / DA Converter), IRC8M (Internal 8MHz RC oscillator), IRC48M (Internal 48MHz RC oscillator at 48MHz frequency), IRC40K (Internal 40KHz RC oscillator), POR / PDR (Power On/Down Reset), PLLs (Phase Locking Loop), LVD (Low Voltage Detector), etc.

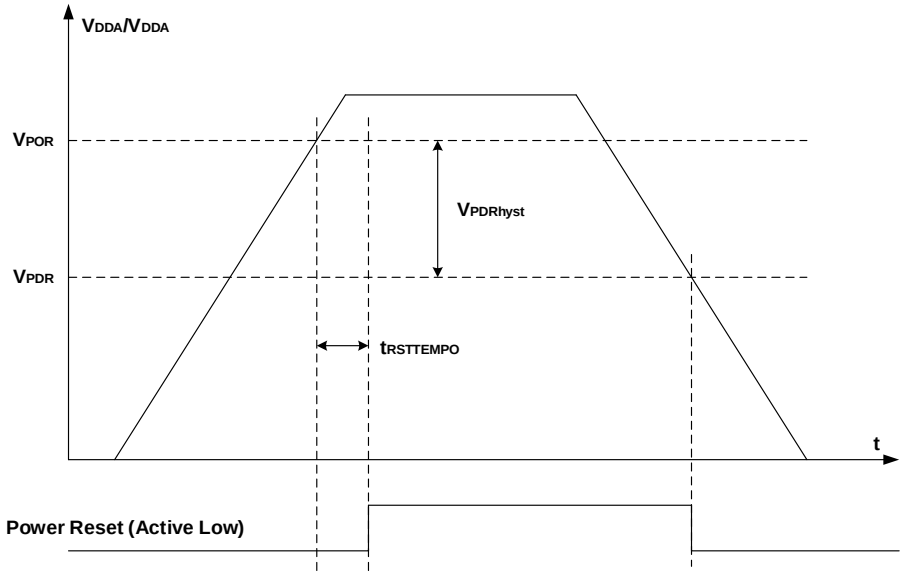
V_{DD} domain

The LDO, which is implemented to supply power for the V_{CORE} domain, is always enabled after the reset. It can be configured to operate in three different status, including the Sleep mode (full power on),in the Deep-sleep mode (on or low power), and in the Standby mode (power off).

V_{DDA} domain

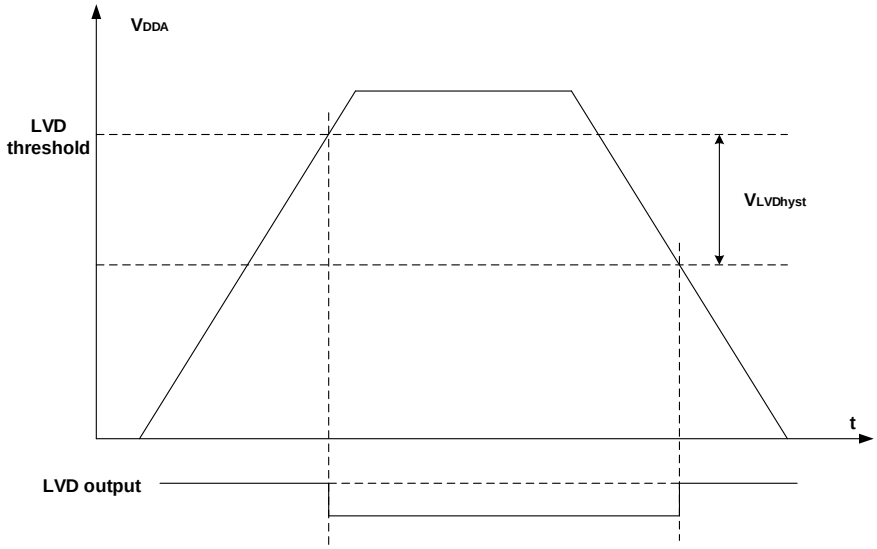
The POR / PDR circuit is implemented to detect VDDA / VDDA and generate the power reset signal which resets the whole chip except the backup domain when the supply voltage is lower than the specified threshold. ***Figure 3-2. Waveform of the POR / PDR*** shows the relationship between the supply voltage and the power reset signal. V_{POR} represents the threshold of power on reset, V_{PDR} represents the threshold of power down reset, $V_{PDRhyst}$ represents the hysteresis value. Please refer to the datasheet for the values of these parameters.

Figure 3-2. Waveform of the POR / PDR



The LVD is used to detect whether the VDDA supply voltage is lower than a programmed threshold selected by the LVDT[2:0] bits in the Power control register (PMU_CTL). The LVD is enabled by setting the LVDEN bit in the PMU_CTL register. And LVDF bit, which is in the Power control and status register (PMU_CS), indicates if V_{DDA} is higher or lower than the LVD threshold. This event is internally connected to the EXTI line 16 and can generate an interrupt if it is enabled through the EXTI registers. ***Figure 3-3. Waveform of the LVD threshold*** shows the relationship between the LVD threshold and the LVD output (LVD interrupt signal depends on EXTI line 16 rising or falling edge configuration). The following figure also shows the relationship between the supply voltage and the LVD signal. Refer to the datasheet for the hysteresis voltage ($V_{LVDhyst}$).

Figure 3-3. Waveform of the LVD threshold



Generally, digital circuits are powered by VDD, while most of analog circuits are powered by VDDA. To improve the ADC and DAC conversion accuracy, the independent power supply

VDDA is implemented to achieve better performance of analog circuits. VDDA can be externally connected to VDD through the external filtering circuit that avoids noise on VDDA, and VSSA should be connected to VSS through the specific circuit independently. Otherwise, when the VDD and VDDA are provided by different power supplies, the difference between VDD and VDDA during power-up and running time should not exceed 0.3V.

To ensure a high accuracy on ADC and DAC, the ADC / DAC independent external reference voltage should be connected to VREFP / VREFN pins. According to the different packages, VREFP pin can be connected to VDDA pin, or external reference voltage which refers to ***Table 12-2. ADC input pins definition*** and ***Table 13-1. DAC I/O description***, VREFN pin must be connected to VSSA pin. The VREFP pin is only available on no less than 100-pin packages, or else the VREFP pin is not available and internally connected to VDDA. The VREFN pin is only available on no less than 100-pin packages, or else the VREFN pin is not available and internally connected to VSSA.

3.3.3. V_{CORE} power domain

The V_{CORE} power domain supplies power for Cortex®-M4 logic, AHB / APB peripherals, the APB interfaces for the Backup domain and the V_{DD} / V_{DDA} domain, etc. Once the V_{CORE} is powered up, the POR will generate a reset sequence on the V_{CORE} power domain. If it is needed to enter the specified power saving mode, the associated control bits must be configured. Then, once a WFI (Wait for Interrupt) or WFE (Wait for Event) instruction is executed, the device will enter the specified power saving mode which will be discussed in the following section. Refer to the datasheet for the V_{CORE} voltage value.

3.3.4. Power saving modes

After a system reset or a power reset, the GD32C11x MCU operates at full function state and all power domains are active. Users can achieve lower power consumption through slowing down the system clocks (HCLK, PCLK1 and PCLK2), closing the clocks of the unused peripherals or configuring the LDO output voltage by LDOVS bits in PMU_CTL register. The LDOVS bits can be configured only when the PLL is off, and the programmed value is selected to drive V_{CORE} domain after the PLL is opened. While the PLL is off, LDO output voltage low mode is selected to drive V_{CORE} domain. Besides, three power saving modes are provided to achieve even lower power consumption. They are Sleep mode, Deep-sleep mode and Standby mode.

Sleep mode

The Sleep mode is corresponding to the SLEEPING mode of the Cortex®-M4. In Sleep mode, only clock of Cortex®-M4 is off. To enter the Sleep mode, it is only necessary to clear the SLEEPDEEP bit in the Cortex®-M4 System Control Register, and execute a WFI or WFE instruction. If the Sleep mode is entered by executing a WFI instruction, any interrupt can wake up the system. If it is entered by executing a WFE instruction, any wakeup event can wake up the system (If SEVONPEND is 1, any interrupt can wake up the system, refer to

Cortex®-M4 Technical Reference Manual). The mode costs the lowest wakeup time as no time is wasted in interrupt entry or exit.

According to the SLEEPONEXIT bit in the Cortex®-M4 System Control Register, there are two options to select the Sleep mode entry mechanism.

- Sleep-now: if the SLEEPONEXIT bit is cleared, the MCU enters Sleep mode as soon as a WFI or WFE instruction is executed.
- Sleep-on-exit: if the SLEEPONEXIT bit is set, the MCU enters Sleep mode as soon as it exits from the ISR with the lowest priority.

Deep-sleep mode

The Deep-sleep mode is based on the SLEEPDEEP mode of the Cortex®-M4. In Deep-sleep mode, all clocks in the V_{CORE} domain are off, and all of IRC8M, IRC48M, HXTAL and PLLs are disabled. The contents of SRAM and registers are preserved. The LDO can operate in normal mode or in low power mode depending on the LDOLP bit in the PMU_CTL register. Before entering the Deep-sleep mode, it is necessary to set the SLEEPDEEP bit in the Cortex®-M4 System Control Register, and clear the STBMOD bit in the PMU_CTL register. Then, the device enters the Deep-sleep mode after a WFI or WFE instruction is executed. If the Deep-sleep mode is entered by executing a WFI instruction, any interrupt from EXTI lines can wake up the system. If it is entered by executing a WFE instruction, any wakeup event from EXTI lines can wake up the system (If SEVONPEND is 1, any interrupt from EXTI lines can wake up the system, refer to Cortex®-M4 Technical Reference Manual). When exiting the Deep-sleep mode, the IRC8M is selected as the system clock. Notice that an additional wakeup delay will be incurred if the LDO operates in low power mode.

Note: In order to enter Deep-sleep mode smoothly, all EXTI line pending status (in the EXTI_PD register) and related peripheral flags must be reset, refer to ***Table 7-3. EXTI source***. If not, the program will skip the entry process of Deep-sleep mode to continue to execute the following procedure.

Standby mode

The Standby mode is also based on the SLEEPDEEP mode of the Cortex®-M4. In Standby mode, the whole V_{CORE} domain is powered off, the LDO is shut down, and all of IRC8M, IRC48M, HXTAL and PLL are disabled. Before entering the Standby mode, it is necessary to set the SLEEPDEEP bit in the Cortex®-M4 System Control Register, set the STBMOD bit in the PMU_CTL register, and clear WUF bit in the PMU_CS register. Then, the device enters the Standby mode after a WFI or WFE instruction is executed, and the STBF status flag in the PMU_CS register indicates whether the MCU has been in Standby mode. There are four wakeup sources for the Standby mode, including the external reset from NRST pin, the RTC alarm, the FWDGT reset and the rising edge on WKUP pin. The Standby mode achieves the lowest power consumption, but spends longest time to wake up. Besides, the contents of SRAM and registers in V_{CORE} power domain are lost in Standby mode. When exiting from the Standby mode, a power-on reset occurs and the Cortex®-M4 will execute instruction code from the 0x0000 0000 address.

Table 3-1. Power saving mode summary

Mode	Sleep	Deep-sleep	Standby
Description	Only CPU clock is off	1. All clocks in the V_{CORE} domain are off 2. Disable IRC8M, IRC48M, HXTAL and PLL	1. The V_{CORE} domain is powered off 2. Disable IRC8M, IRC48M, HXTAL and PLL
LDO Status	On (normal power mode)	On (normal or low power mode)	Off
Configuration	SLEEPDEEP = 0	SLEEPDEEP = 1 STBMOD = 0	SLEEPDEEP = 1 STBMOD = 1, WURST=1
Entry	WFI or WFE	WFI or WFE	WFI or WFE
Wakeup	Any interrupt for WFI. Any event (or interrupt when SEVONPEND is 1) for WFE	Any interrupt from EXTI lines for WFI Any event(or interrupt when SEVONPEND is 1) from EXTI for WFE	1. NRST pin 2. WKUP pin 3. FWDGT reset 4. RTC alarm
Wakeup Latency	None	IRC8M wakeup time, LDO wakeup time added if LDO is in low power mode	Power on sequence

Note: In Standby mode, all I / Os are in high-impedance state except NRST pin, PC13 pin when configured for RTC function, PC14 and PC15 pins when used as LXTAL crystal oscillator pins, and WKUP pin if enabled.

3.4. Register definition

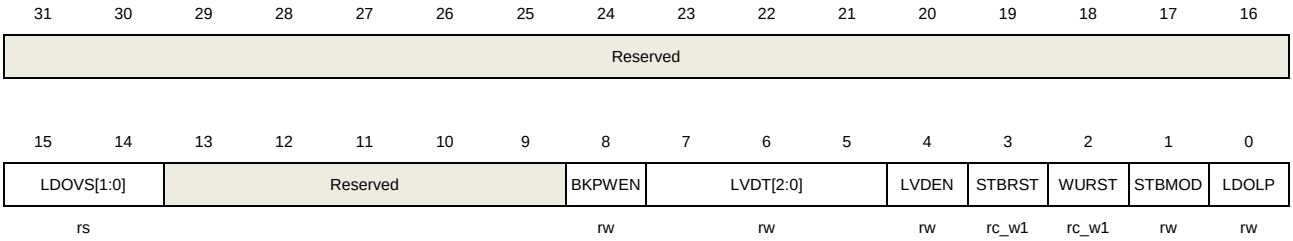
PMU base address: 0x4000 7000

3.4.1. Control register (PMU_CTL)

Address offset: 0x00

Reset value: 0x0000 4000 (reset after wakeup from Standby mode)

This register can be accessed by half-word(16-bit) or word(32-bit)



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:14	LDOVS[1:0]	LDO output voltage select These bits are set by software when the main PLL is closed. The LDO output voltage selected by LDOVS bits takes effect only when the main PLL is enabled. If the main PLL is closed, the LDO output voltage low mode is selected. 00: Reserved (LDO output voltage normal mode) 01: LDO output voltage normal mode 10: Reserved (LDO output voltage low mode) 11: LDO output voltage low mode
13:9	Reserved	Must be kept at reset value.
8	BKPWEN	Backup Domain Write Enable 0: Disable write access to the registers in backup domain 1: Enable write access to the registers in backup domain After reset, any write access to the registers in backup domain is ignored. This bit has to be set to enable write access to these registers.
7:5	LVDT[2:0]	Low Voltage Detector Threshold 000: 2.1V 001: 2.3V 010: 2.4V 011: 2.6V 100: 2.7V 101: 2.9V 110: 3.0V 111: 3.1V

4	LVDEN	Low Voltage Detector Enable 0: Disable Low Voltage Detector 1: Enable Low Voltage Detector
3	STBRST	Standby Flag Reset 0: No effect 1: Reset the standby flag This bit is always read as 0.
2	WURST	Wakeup Flag Reset 0: No effect 1: Reset the wakeup flag This bit is always read as 0.
1	STBMOD	Standby Mode 0: Enter the Deep-sleep mode when the Cortex®-M4 enters SLEEPDEEP mode 1: Enter the Standby mode when the Cortex®-M4 enters SLEEPDEEP mode
0	LDOLP	LDO Low Power Mode 0: The LDO operates normally during the Deep-sleep mode 1: The LDO is in low power mode during the Deep-sleep mode

3.4.2. Control and status register (PMU_CS)

Address offset: 0x04

Reset value: 0x0000 0000 (will not reset after wakeup from Standby mode)

This register can be accessed by half-word(16-bit) or word(32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							WUPEN	Reserved					LVDF	STBF	WUF
							rw						r	r	r

Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	WUPEN	WKUP Pin Enable 0: Disable WKUP pin function 1: Enable WKUP pin function If WUPEN is set before entering the Standby mode, a rising edge on the WKUP pin will wake up the system from the Standby mode. As the WKUP pin is active high, the WKUP pin is internally configured to input pull down mode. And setting this bit will trigger a wakeup event when the input is already high.

7:3	Reserved	Must be kept at reset value.
2	LVDF	Low Voltage Detector Status Flag 0: Low Voltage event has not occurred (V_{DD} is higher than the specified LVD threshold) 1: Low Voltage event occurred (V_{DD} is equal to or lower than the specified LVD threshold) **Note:** The LVD function is disabled in Standby mode.
1	STBF	Standby Flag 0: The device has not entered the Standby mode 1: The device has been in the Standby mode This bit is cleared only by a POR/PDR or by setting the STBRST bit in the PMU_CTL register.
0	WUF	Wakeup Flag 0: No wakeup event has been received 1: Wakeup event occurred from the WKUP pin or the RTC alarm event. This bit is reset by the system or cleared by setting the WURST bit in the PMU_CTL register.

4. Backup registers (BKP)

4.1. Overview

The Backup registers are located in the Backup domain that remains powered-on by V_{BAT} even if V_{DD} power is shut down, they are forty two 16-bit (84 bytes) registers for data protection of user application data, and the wake-up action from Standby mode or system reset will not affect these registers.

In addition, the Backup registers can be used to implement the tamper detection and RTC calibration function.

After reset, any writing access to the registers in Backup domain is disabled, that is, the Backup registers and RTC can not be accessed by writting operation. In order to enable access to the Backup registers and RTC, the Power and Backup interface clocks should be enabled firstly by setting the PMUEN and BKPIEN bits in the RCU_APB1EN register, and writing access to the registers in Backup domain should be enabled by setting the BKPWEN bit in the PMU_CTL register.

4.2. Characteristics

- 84 bytes Backup registers which can keep data under power saving mode. If a tamper event is detected, Backup registers will be reset.
- The active level of Tamper source (PC13) can be configured.
- RTC Clock Calibration register provides RTC alarm and second output selection, and the function of setting the calibration value.
- Tamper control and status register (BKP_TPCS) can control interrupt or event of tamper detection.

4.3. Function overview

4.3.1. RTC clock calibration

In order to improve the RTC clock accuracy, the MCU provides the RTC output for calibration function. The RTC clock, or a clock whose frequency is f_{RTCCLK}/64, can be output on the PC13. It is enabled by setting the COEN bit in the BKP_OCTL register.

The calibration value is set by RCCV[6:0] in the BKP_OCTL register, and the calibration function can slow down the RTC clock by steps of 1000000/2^20 ppm.

4.3.2. Tamper detection

In order to protect the important user data, the MCU provides the tamper detection function,

and it can be independently enabled on TAMPER pin by setting corresponding TPEN bit in the BKP_TPCTL register. To prevent the tamper event from losing, the value of the edge detection signal logically ANDed with the TPEN bit, is used as the input of tamper detection signal. So the tamper detection configuration should be set before TAMPER pin is enabled. When the tamper event is detected, the corresponding TEF bit in the BKP_TPCS register will be set. Tamper event can generate an interrupt if tamper interrupt is enabled. Any tamper event will reset all Backup data registers.

Note: When TPAL=0/1, if the TAMPER pin is already high/low before it is enabled(by setting TPEN bit), an extra tamper event will occur even if there is no rising/falling edge on the TAMPER pin after TPEN bit is set.

4.4. Register definition

BKP base address: 0x4000 6C00

4.4.1. Backup data register x (BKP_DATAx) (x= 0..41)

Address offset: 0x04 to 0x28, 0x40 to 0xBC
Reset value: 0x0000

This register can be accessed by half-word (16-bit) or word (32-bit)

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

Bits	Fields	Descriptions
15:0	DATA[15:0]	Backup data These bits are used for general purpose data storage. The contents of the BKP_DATAx register will be remained even if waking up from Standby mode, system reset or power reset.

4.4.2. RTC signal output control register (BKP_OCTL)

Address offset: 0x2C
Reset value: 0x0000

This register can be accessed by half-word(16-bit) or word(32-bit)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CALDIR	CCOSEL	Reserved				ROSEL	ASOEN	COEN	RCCV[6:0]						
rw	rw					rw	rw	rw	rw						

Bits	Fields	Descriptions
15	CALDIR	RTC clock calibration direction 0: Slow down 1: Speed up This bit is reset only by a Backup domain reset.
14	CCOSEL	RTC clock output selection 0: RTC clock divided by 64 1: RTC clock This bit is reset only by a POR.
13:10	Reserved	Must be kept at reset value.
9	ROSEL	RTC output selection 0: RTC alarm pulse is selected as the RTC output

		1: RTC second pulse is selected as the RTC output This bit is reset only by a Backup domain reset.
8	ASOEN	RTC alarm or second signal output enable 0: Disable RTC alarm or second output 1: Enable RTC alarm or second output When enabled, the TAMPER pin will output the RTC output. This bit is reset only by a Backup domain reset.
7	COEN	RTC clock calibration output enable 0: Disable RTC clock calibration output 1: Enable RTC clock Calibration output When enabled, the TAMPER pin will output the RTC clock or RTC clock divided by 64. ASOEN has the priority over COEN. When ASOEN is set, the TAMPER pin will output the RTC alarm or second signal whether COEN is set or not. This bit is reset only by a POR.
6:0	RCCV[6:0]	RTC clock calibration value The value indicates how many clock pulses are ignored or added every 2^20 RTC clock pulses. These bits are reset only by a Backup domain reset.

4.4.3. Tamper pin control register (BKP_TPCTL)

Address offset: 0x30
Reset value: 0x0000

This register can be accessed by half-word (16-bit) or word (32-bit)

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

Bits	Fields	Descriptions
15:2	Reserved	Must be kept at reset value.
1	TPAL	TAMPER pin active level 0: The TAMPER pin is active high 1: The TAMPER pin is active low
0	TPEN	TAMPER detection enable 0: The TAMPER pin is free for GPIO functions 1: The TAMPER pin is dedicated for the Backup Reset function. The active level on the TAMPER pin resets all data of the BKP_DATAx register.

4.4.4. Tamper control and status register (BKP_TPCS)

Address offset: 0x34

Reset value: 0x0000

This register can be accessed by half-word (16-bit) or word (32-bit)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						TIF	TEF	Reserved					TPIE	TIR	TER
						r	r						rw	w	w

Bits	Fields	Descriptions
15:10	Reserved	Must be kept at reset value.
9	TIF	Tamper interrupt flag 0: No tamper interrupt occurred 1: A tamper interrupt occurred This bit is reset by writing 1 to the TIR bit or writing 0 to the TPIE bit.
8	TEF	Tamper event flag 0: No tamper event occurred 1: A tamper event occurred This bit is reset by writing 1 to the TER bit.
7:3	Reserved	Must be kept at reset value.
2	TPIE	Tamper interrupt enable 0: Disable the tamper interrupt 1: Enable the tamper interrupt This bit is reset only by a system reset or the wake-up from Standby mode.
1	TIR	Tamper interrupt reset 0: No effect 1: Reset the TIF bit This bit is always read as 0.
0	TER	Tamper event reset 0: No effect 1: Reset the TEF bit This bit is always read as 0.

5. Reset and clock unit (RCU)

5.1. Reset control unit (RCTL)

5.1.1. Overview

Reset control uint includes three kinds of reset: power reset, system reset and backup domain reset. The power reset, known as a cold reset, resets the full system except the backup domain. The system reset resets the processor core and peripheral IP components except for the SW-DP controller and the backup domain. The backup domain reset resets the backup domain. The resets can be triggered by an external signal, internal events and the reset generators. More information about these resets will be described in the following sections.

5.1.2. Function overview

Power reset

The power reset is generated by either an external reset as power on/power down reset (POR/PDR reset) or the internal reset generator when exiting standby mode. The power reset sets all registers to their reset values except the backup domain. The power reset whose active signal is low, it will be de-asserted when the internal LDO voltage regulator is ready to provide 1.2V power. The reset service routine vector is fixed at address 0x0000_0004 in the memory map.

System reset

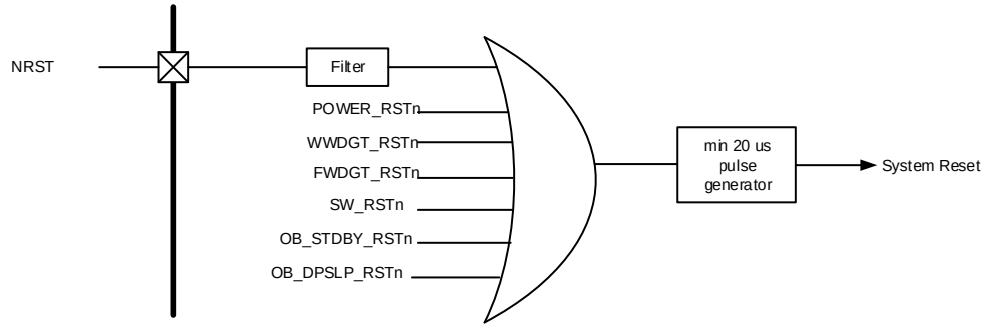
A system reset is generated by the following events:

- A power reset (POWER_RSTn)
- A external pin reset (NRST)
- A window watchdog timer reset (WWDGT_RSTn)
- A free watchdog timer reset (FWDGT_RSTn)
- The SYSRESETREQ bit in Cortex®-M4 application interrupt and reset control register is set (SW_RSTn)
- Reset generated when entering standby mode and resetting nRST_STDBY bit in user option bytes (OB_STDBY_RSTn)
- Reset generated when entering deep-sleep mode and resetting nRST_DPSLP bit in user option bytes (OB_DPSLP_RSTn)

A system reset resets the processor core and peripheral IP components except for the SW-DP controller and the backup domain.

A system reset pulse generator guarantees low level pulse duration of 20 μs for each reset source (external or internal reset).

Figure 5-1. The system reset circuit



Backup domain reset

A backup domain reset is generated by setting the BKPRST bit in the backup domain control register or backup domain power on reset (V_{DD} or V_{BAT} power on in case of both supplies have been powered off previously).

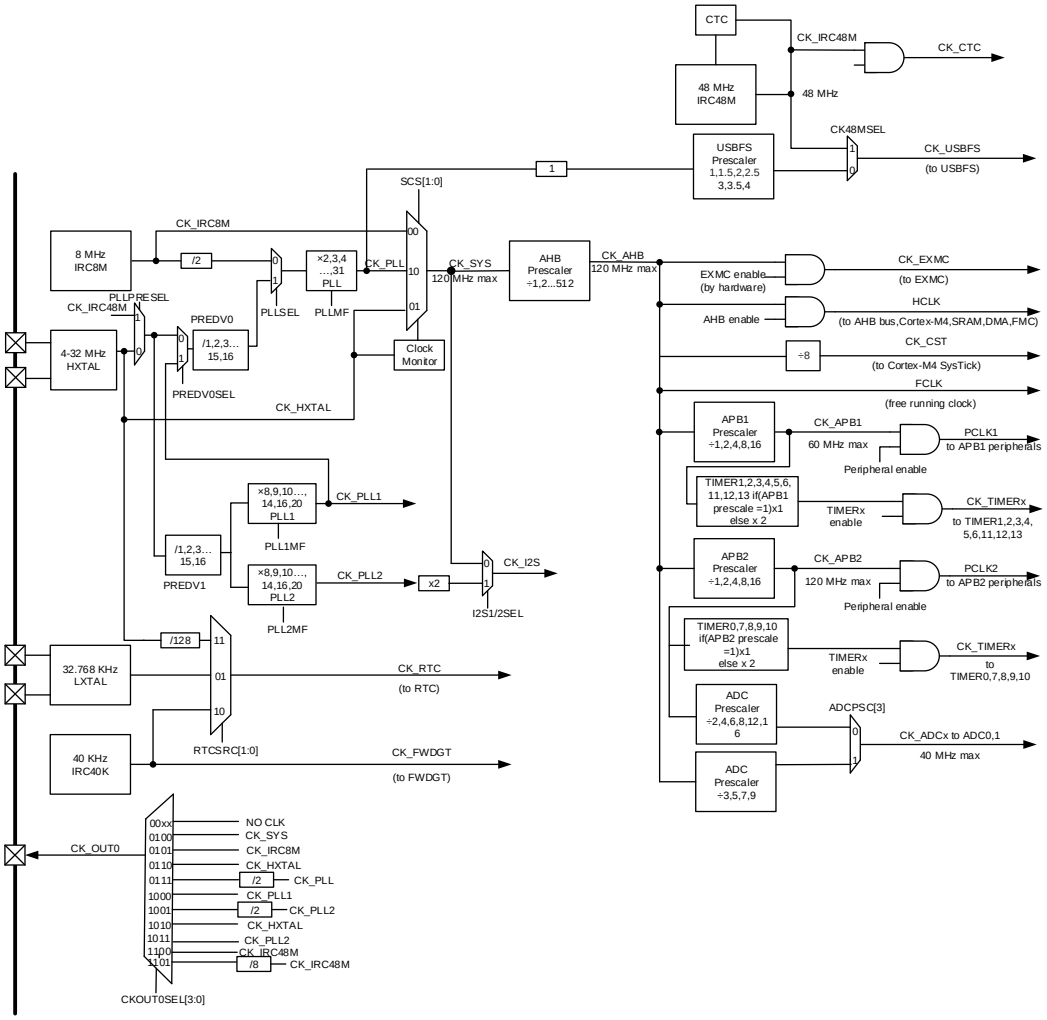
5.2. Clock control unit (CCTL)

5.2.1. Overview

The clock control unit provides a range of frequencies and clock functions. These include a Internal 8M RC oscillator (IRC8M), a Internal 48M RC oscillator (IRC48M), a High speed crystal oscillator (HXTAL), a Low speed Internal 40K RC oscillator (IRC40K), a Low speed crystal oscillator (LXTAL), three Phase Lock Loop (PLL), a HXTAL clock monitor, clock prescalers, clock multiplexers and clock gating circuitry.

The clocks of the AHB, APB and Cortex®-M4 are derived from the system clock (CK_SYS) which can source from the IRC8M, HXTAL or PLL. The maximum operating frequency of the system clock (CK_SYS) can be up to 120 MHz.

Figure 5-2. Clock tree



The frequency of AHB, APB2 and the APB1 domains can be configured by each prescaler. The maximum frequency of the AHB, APB2 and APB1 domains is 120 MHz/120 MHz/60 MHz. The Cortex System Timer (SysTick) external clock is clocked with the AHB clock (HCLK) divided by 8. The systick can work either with this clock or with the AHB clock (HCLK), configurable in the systick control and status register.

The ADCs are clocked by the clock of APB2 divided by 2, 4, 6, 8, 12, 16 or by the clock of AHB divided by 3, 5, 7, 9, which defined by ADCPSC in RCU_CFG0 and RCU_CFG1 register.

The TIMERs are clocked by the clock divided from CK_APB2 and CK_APB1. The frequency of TIMERs clock is equal to CK_APBx(APB prescaler is 1) or twice the CK_APBx(APB prescaler is not 1).

The USBFS is clocked by the clock of CK48M. The CK48M is selected from the clock of CK_PLL or the clock of IRC48M by CK48MSEL bit in RCU_ADDCTL register.

The CTC is clocked by the clock of IRC48M. The IRC48M can be trimmed by CTC unit automatically.

The I2S is clocked by the clock of CK_SYS or PLL2*2 which defined by I2SxSEL bit in RCU_CFG1 register.

The RTC is clocked by LXTAL clock, IRC40K clock or HXTAL clock divided by 128 (defined which clock selected by RTCSRC bit in backup domain control register (RCU_BDCTL)). After the RTC select HXTAL clock divided by 128, the clock disappeared when the 1.2V core domain power off. After the RTC select IRC40K, the clock disappeared when V_{DD} power off. After the RTC select LXTAL, the clock disappeared when V_{DD} and V_{BAT} power off.

The FWDGT is clocked by IRC40K clock, which is forced on when FWDGT started.

5.2.2. Characteristics

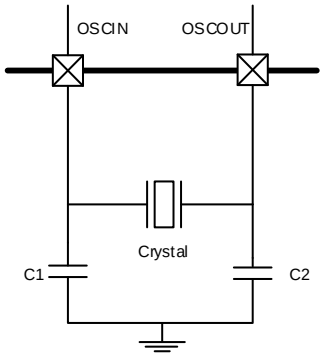
- 4 to 32 MHz High speed crystal oscillator (HXTAL)
- Internal 8 MHz RC oscillator (IRC8M)
- Internal 48 MHz RC oscillator (IRC48M)
- 32,768 Hz Low speed crystal oscillator (LXTAL)
- Internal 40KHz RC oscillator (IRC40K)
- PLL clock source can be HXTAL, IRC8M or IRC48M
- HXTAL clock monitor

5.2.3. Function overview

High speed crystal oscillator (HXTAL)

The high speed external crystal oscillator (HXTAL), which has a frequency from 4 to 32 MHz, produces a highly accurate clock source for use as the system clock. A crystal with a specific frequency must be connected and located close to the two HXTAL pins. The external resistor and capacitor components connected to the crystal are necessary for proper oscillation.

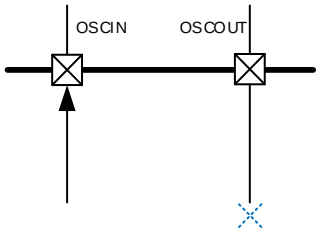
Figure 5-3. HXTAL clock source



The HXTAL crystal oscillator can be switched on or off using the HXTALEN bit in the control register RCU_CTL. The HXTALSTB flag in control register RCU_CTL indicates if the high-speed external crystal oscillator is stable. When the HXTAL is powered up, it will not be released for use until this HXTALSTB bit is set by the hardware. This specific delay period is known as the oscillator “Start-up time”. As the HXTAL becomes stable, an interrupt will be generated if the related interrupt enable bit HXTALSTBIE in the Interrupt register RCU_INT is set. At this point the HXTAL clock can be used directly as the system clock source or the PLL input clock.

Select external clock bypass mode by setting the HXTALBPS and HXTALEN bits in the Control Register RCU_CTL. During bypass mode, the signal is connected to OSCIN, and OSCOUT remains in the suspended state, as shown in ***Figure 5-4. HXTAL clock source in bypass mode***. The CK_HXTAL is equal to the external clock which drives the OSCIN pin.

Figure 5-4. HXTAL clock source in bypass mode



Internal 8M RC oscillators (IRC8M)

The internal 8M RC oscillator, IRC8M, has a fixed frequency of 8 MHz and is the default clock source selection for the CPU when the device is powered up. The IRC8M oscillator provides a lower cost type clock source as no external components are required. The IRC8M RC oscillator can be switched on or off using the IRC8MEN bit in the control register RCU_CTL. The IRC8MSTB flag in the control register RCU_CTL is used to indicate if the internal 8M RC oscillator is stable. The start-up time of the IRC8M oscillator is shorter than the HXTAL crystal oscillator. An interrupt can be generated if the related interrupt enable bit IRC8MSTBIE in the clock Interrupt register RCU_INT is set when the IRC8M becomes stable. The IRC8M clock can also be used as the system clock source or the PLL input clock.

The frequency accuracy of the IRC8M can be calibrated by the manufacturer, but its operating frequency is still less accurate than HXTAL. The application requirements, environment and cost will determine which oscillator type is selected.

If the HXTAL or PLL is the system clock source, to minimize the time required for the system to recover from the Deep-sleep Mode, the hardware forces the IRC8M clock to be the system clock when the system wakes up initially.

Internal 48M RC oscillators (IRC48M)

The internal 48M RC oscillator, IRC48M, has a fixed frequency of 48 MHz. The IRC48M oscillator provides a lower cost type clock source, no need external components when used for USBFS.The IRC48M RC oscillator can be switched on or off using the IRC48MEN bit in the RCU_ADDCTL register. The IRC48MSTB flag in the RCU_ADDCTL register is used to indicate if the internal 48M RC oscillator is stable. An interrupt can be generated if the related interrupt enable bit IRC48MSTBIE in the RCU_ADDINT register is set when the IRC48M becomes stable. The IRC48M clock is used for the clocks of USBFS.

The frequency accuracy of the IRC48M can be calibrated by the manufacturer, but its operating frequency is still not enough accurate, because the USB need the frequency must between 48MHz with 500ppm accuracy. A hardware automatic dynamic trim performed in CTC unit adjust the IRC48M to the needed frequency.

Phase locked loop (PLL)

There are three internal Phase Locked Loop, the PLL, PLL1 and PLL2.

The PLL can be switched on or off by using the PLLEN bit in the RCU_CTL register. The PLLSTB flag in the RCU_CTL register will indicate if the PLL clock is stable. An interrupt can be generated if the related interrupt enable bit, PLLSTBIE, in the RCU_INT register, is set as the PLL becomes stable.

The PLL1 can be switched on or off by using the PLL1EN bit in the RCU_CTL register. The PLL1STB flag in the RCU_CTL register will indicate if the PLL1 clock is stable. An interrupt can be generated if the related interrupt enable bit, PLL1STBIE, in the RCU_INT register, is set as the PLL1 becomes stable.

The PLL2 can be switched on or off by using the PLL2EN bit in the RCU_CTL register. The PLL2STB flag in the RCU_CTL register will indicate if the PLL2 clock is stable. An interrupt can be generated if the related interrupt enable bit, PLL2STBIE, in the RCU_INT register, is set as the PLL2 becomes stable.

The three PLLs are closed by hardware when entering the Deepsleep / Standby mode or HXTAL monitor fail when HXTAL used as the source clock of the PLLs.

Low speed crystal oscillator (LXTAL)

The low speed external crystal or ceramic resonator oscillator, which has a frequency of 32,768 Hz, produces a low power but highly accurate clock source for the Real Time Clock circuit. The LXTAL oscillator can be switched on or off using the LXTALEN bit in the backup domain control register (RCU_BDCTL). The LXTALSTB flag in the backup domain control register (RCU_BDCTL) will indicate if the LXTAL clock is stable. An interrupt can be generated if the related interrupt enable bit, LXTALSTBIE, in the interrupt register RCU_INT is set when the LXTAL becomes stable.

Select external clock bypass mode by setting the LXTALBPS and LXTALEN bits in the backup domain control register (RCU_BDCTL). The CK_LXTAL is equal to the external clock which drives the OSC32IN pin.

Internal 40K RC oscillator (IRC40K)

The internal RC oscillator has a frequency of about 40 kHz and is a low power clock source for the real time clock circuit or the free watchdog timer. The IRC40K offers a low cost clock source as no external components are required. The IRC40K RC oscillator can be switched on or off by using the IRC40KEN bit in the reset source/clock register (RCU_RSTSCK). The IRC40KSTB flag in the reset source/clock register RCU_RSTSCK will indicate if the IRC40K clock is stable. An interrupt can be generated if the related interrupt enable bit IRC40KSTBIE in the clock interrupt register (RCU_INT) is set when the IRC40K becomes stable.

The IRC40K can be trimmed by TIMER4_CH3, user can get the clocks frequency, and adjust the RTC and FWDGT counter. Please refer to TIMER4CH3_IREMAP in AFIO_PCF0 register.

System clock (CK_SYS) selection

After the system reset, the default CK_SYS source will be IRC8M and can be switched to HXTAL or CK_PLL by changing the system clock switch bits, SCS, in the clock configuration register 0, RCU_CFG0. When the SCS value is changed, the CK_SYS will continue to operate using the original clock source until the target clock source is stable. When a clock source is used as the CK_SYS directly or indirectly (by PLL), it is not possible to stop it.

HXTAL clock monitor (CKM)

The HXTAL clock monitor function is enabled by the HXTAL clock monitor enable bit CKMEN in the control register (RCU_CTL). This function should be enabled after the HXTAL start-up delay and disabled when the HXTAL is stopped. Once the HXTAL failure is detected, the HXTAL will be automatically disabled. The HXTAL clock stuck interrupt flag, CKMIF, in the clock interrupt register RCU_INT will be set and the HXTAL failure event will be generated. This failure interrupt is connected to the non-maskable interrupt, NMI, of the Cortex-M4. If the HXTAL is selected as the clock source of CK_SYS, PLL and CK_RTC, the HXTAL failure will force the CK_SYS source to IRC8M, the PLL will be disabled automatically. If the HXTAL is selected as the clock source of PLL, the HXTAL failure will force the PLL closed automatically. If the HXTAL is selected as the clock source of RTC, the HXTAL failure will reset the RTC clock selection.

Clock output capability

The clock output capability is ranging from 0.09375 MHz to 120 MHz. There are several clock signals can be selected via the CK_OUT0 clock source selection bits, CKOUT0SEL, in the clock configuration register 0 (RCU_CFG0). The corresponding GPIO pin should be configured in the properly alternate function I/O (AFIO) mode to output the selected clock signal.

Table 5-1. Clock output 0 source select

Clock Source 0 Selection bits	Clock Source
00xx	NO CLK
0100	CK_SYS
0101	CK_IRC8M
0110	CK_HXTAL
0111	CK_PLL/2
1000	CK_PLL1
1001	CK_PLL2/2
1010	CK_HXTAL
1011	CK_PLL2
1100	CK_IRC48M
1101	CK_IRC48M/8

Voltage control

The 1.2V domain voltage in Deep-sleep mode can be controlled by DSLPVS[1:0] bit in the Deep-sleep mode voltage register (RCU_DSV).

Table 5-2. 1.2V domain voltage selected in deep-sleep mode

DSLPVS[1:0]	Deep-sleep mode voltage(V)
00	1.0
01	0.9
10	0.8
11	1.2

5.3. Register definition

RCU base address: 0x4002 1000

5.3.1. Control register (RCU_CTL)

Address offset: 0x00

Reset value: 0x0000 xx83 where x is undefined.

This register can be accessed by byte (8-bit), half-word (16-bit) and word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		PLL2STB	PLL2EN	PLL1STB	PLL1EN	PLLSTB	PLLEN	Reserved				CKMEN	HXTALBPS	HXTALSTB	HXTALEN
		r	rw	r	rw	r	rw					rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRC8MCALIB[7:0]								IRC8MADJ[4:0]					Reserved	IRC8MSTB	IRC8MEN
r								rw						r	rw

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29	PLL2STB	PLL2 clock stabilization flag Set by hardware to indicate if the PLL2 output clock is stable and ready for use. 0: PLL2 is not stable 1: PLL2 is stable
28	PLL2EN	PLL2 enable Set and reset by software. Reset by hardware when entering Deep-sleep or Standby mode. 0: PLL2 is switched off 1: PLL2 is switched on
27	PLL1STB	PLL1 clock stabilization flag Set by hardware to indicate if the PLL1 output clock is stable and ready for use. 0: PLL1 is not stable 1: PLL1 is stable
26	PLL1EN	PLL1 enable Set and reset by software. Reset by hardware when entering Deep-sleep or Standby mode. 0: PLL1 is switched off 1: PLL1 is switched on
25	PLLSTB	PLL Clock Stabilization flag Set by hardware to indicate if the PLL output clock is stable and ready for use. 0: PLL is not stable

		1: PLL is stable
24	PLLEN	PLL enable Set and reset by software. This bit cannot be reset if the PLL clock is used as the system clock. Reset by hardware when entering Deep-sleep or Standby mode. 0: PLL is switched off 1: PLL is switched on
23:20	Reserved	Must be kept at reset value.
19	CKMEN	HXTAL clock monitor enable 0: Disable the High speed 4 ~ 32 MHz crystal oscillator (HXTAL) clock monitor 1: Enable the High speed 4 ~ 32 MHz crystal oscillator (HXTAL) clock monitor When the hardware detects that the HXTAL clock is stuck at a low or high state, the system clock will switch to the internal high speed clock (IRC8M) by hardware. The way to recover the original system clock by an external reset, power on reset or clearing CKMIF by software. **Note:** When the HXTAL clock monitor is enabled, the hardware will enable the IRC8M internal RC oscillator automatically,no matter what state of the IRC8MEN bit .
18	HXTALBPS	High speed crystal oscillator (HXTAL) clock bypass mode enable The HXTALBPS bit can be written only if the HXTALEN is 0. 0: Disable the HXTAL Bypass mode 1: Enable the HXTAL Bypass mode in which the HXTAL output clock is equal to the input clock.
17	HXTALSTB	High speed crystal oscillator (HXTAL) clock stabilization flag Set by hardware to indicate if the HXTAL oscillator is stable and ready for use. 0: HXTAL oscillator is not stable 1: HXTAL oscillator is stable
16	HXTALEN	High speed crystal oscillator (HXTAL) enable Set and reset by software. This bit cannot be reset if the HXTAL clock is used as the system clock or the PLL input clock when PLL clock is selected to the system clock. Reset by hardware when entering Deep-sleep or Standby mode. 0: High speed 4 ~ 32 MHz crystal oscillator disabled 1: High speed 4 ~ 32 MHz crystal oscillator enabled
15:8	IRC8MCALIB[7:0]	Internal 8MHz RC oscillator calibration value register These bits are load automatically at power on.
7:3	IRC8MADJ[4:0]	Internal 8MHz RC oscillator clock trim adjust value These bits are set by software. The trimming value equal to these bits (IRC8MADJ) added to the IRC8MCALIB [7:0] bits. The trimming value should trim the IRC8M to 8 MHz ± 1%.
2	Reserved	Must be kept at reset value.

1	IRC8MSTB	Internal 8MHz RC oscillator stabilization flag Set by hardware to indicate if the IRC8M oscillator is stable and ready for use. 0: IRC8M oscillator is not stable 1: IRC8M oscillator is stable
0	IRC8MEN	Internal 8MHz RC oscillator enable Set and reset by software. This bit cannot be reset if the IRC8M clock is used as the system clock. Set by hardware when leaving Deep-sleep or Standby mode or the HXTAL clock is stuck at a low or high state when CKMEN is set. 0: Internal 8 MHz RC oscillator disabled 1: Internal 8 MHz RC oscillator enabled

5.3.2. Clock configuration register 0 (RCU_CFG0)

Address offset: 0x04

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
USBFSPSC[2]	Reserved	PLLMF[4]	ADCPSC [2]	CKOUT0SEL[3:0]				USBFSPSC[1:0]		PLLMF[3:0]				PREDV0_LSB	PLLSEL
rw		rw	rw	rw				rw		rw				rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCPSC[1:0]		APB2PSC[2:0]			APB1PSC[2:0]			AHBPSC[3:0]				SCSS[1:0]		SCS[1:0]	
rw		rw			rw			rw				r		rw	

Bits	Fields	Descriptions
31	USBFSPSC[2]	Bit 2 of USBFSPSC see bits 23:22 of RCU_CFG0
30	Reserved	Must be kept at reset value.
29	PLLMF[4]	Bit 4 of PLLMF see bits 21:18 of RCU_CFG0
28	ADCPSC[2]	Bit 2 of ADCPSC see bits 15:14 of RCU_CFG0
27:24	CKOUT0SEL[3:0]	CKOUT0 clock source selection Set and reset by software. 00xx: No clock selected 0100: System clock selected 0101: High speed 8M Internal oscillator clock selected 0110: External high speed oscillator clock selected 0111: (CK_PLL / 2) clock selected 1000: CK_PLL1 clock selected 1001: CK_PLL2 clock divided by 2 selected

1010: CK_HXTAL clock selected
1011: CK_PLL2 clock selected
1100: CK_IRC48M clock selected
1101: CK_IRC48M clock divided by 8 selected

23:22 USBFSPSC[1:0] USBFS clock prescaler selection
Bit 31 of RCU_CFG0 and these bits are written by software to control the USBFS clock prescaler value. The USBFS clock must be 48MHz. These bits can't be reset if the USBFS clock is enabled.
000: CK_USBFS = CK_PLL / 1.5
001: CK_USBFS = CK_PLL
010: CK_USBFS = CK_PLL / 2.5
011: CK_USBFS = CK_PLL / 2
100: CK_USBFS = CK_PLL / 3
101: CK_USBFS = CK_PLL / 3.5
11x :CK_USBFS = CK_PLL / 4

21:18 PLLMF[3:0] The PLL clock multiplication factor
Bit 29 of RCU_CFG0 and these bits are written by software to define the PLL multiplication factor
Note: The PLL output frequency must not exceed 120 MHz
00000: (PLL source clock x 2)
00001: (PLL source clock x 3)
00010: (PLL source clock x 4)
00011: (PLL source clock x 5)
00100: (PLL source clock x 6)
00101: (PLL source clock x 7)
00110: (PLL source clock x 8)
00111: (PLL source clock x 9)
01000: (PLL source clock x 10)
01001: (PLL source clock x 11)
01010: (PLL source clock x 12)
01011: (PLL source clock x 13)
01100: (PLL source clock x 14)
01101: (PLL source clock x 6.5)
01110: (PLL source clock x 16)
01111: (PLL source clock x 16)
10000: (PLL source clock x 17)
10001: (PLL source clock x 18)
10010: (PLL source clock x 19)
10011: (PLL source clock x 20)
10100: (PLL source clock x 21)
10101: (PLL source clock x 22)
10110: (PLL source clock x 23)
10111: (PLL source clock x 24)

		11000: (PLL source clock x 25) 11001: (PLL source clock x 26) 11010: (PLL source clock x 27) 11011: (PLL source clock x 28) 11100: (PLL source clock x 29) 11101: (PLL source clock x 30) 11110: (PLL source clock x 31) 11111: (PLL source clock x 31)
17	PREDV0_LSB	The LSB of PREDV0 division factor This bit is the same as PREDV0[0] of RCU_CFG1. Changing the PREDV0 division factor bit [0] of RCU_CFG1, this bit is also changed. When the PREDV0 division factor bits [3:1] are not set, this bit controls PREDV0 input clock divided by 2 or not.
16	PLLSEL	PLL clock source selection Set and reset by software to control the PLL clock source. 0: (IRC8M / 2) clock selected as source clock of PLL 1: HXTAL or IRC48M(PLLPRESEL of RCU_CFG1 register) selected as source clock of PLL
15:14	ADCPSC[1:0]	ADC clock prescaler selection These bits, bit 28 of RCU_CFG0 and bit 29 of RCU_CFG1 are written by software to define the ADC prescaler factor.Set and cleared by software. 0000: (CK_APB2 / 2) selected 0001: (CK_APB2 / 4) selected 0010: (CK_APB2 / 6) selected 0011: (CK_APB2 / 8) selected 0100: (CK_APB2 / 2) selected 0101: (CK_APB2 / 12) selected 0110: (CK_APB2 / 8) selected 0111: (CK_APB2 / 16) selected 1x00 : (CK_AHB / 3) selected 1x01 : (CK_AHB / 5) selected 1x10 : (CK_AHB / 7) selected 1x11 : (CK_AHB / 9) selected
13:11	APB2PSC[2:0]	APB2 prescaler selection Set and reset by software to control the APB2 clock division ratio. 0xx: CK_AHB selected 100: (CK_AHB / 2) selected 101: (CK_AHB / 4) selected 110: (CK_AHB / 8) selected 111: (CK_AHB / 16) selected
10:8	APB1PSC[2:0]	APB1 prescaler selection Set and reset by software to control the APB1 clock division ratio. Caution: The CK_APB1 output frequency must not exceed 60 MHz.

0xx: CK_AHB selected
100: (CK_AHB / 2) selected
101: (CK_AHB / 4) selected
110: (CK_AHB / 8) selected
111: (CK_AHB / 16) selected

7:4	AHBPSC[3:0]	AHB prescaler selection Set and reset by software to control the AHB clock division ratio 0xxx: CK_SYS selected 1000: (CK_SYS / 2) selected 1001: (CK_SYS / 4) selected 1010: (CK_SYS / 8) selected 1011: (CK_SYS / 16) selected 1100: (CK_SYS / 64) selected 1101: (CK_SYS / 128) selected 1110: (CK_SYS / 256) selected 1111: (CK_SYS / 512) selected
3:2	SCSS[1:0]	System clock switch status Set and reset by hardware to indicate the clock source of system clock. 00: Select CK_IRC8M as the CK_SYS source 01: Select CK_HXTAL as the CK_SYS source 10: Select CK_PLL as the CK_SYS source 11: Reserved
1:0	SCS[1:0]	System clock switch Set by software to select the CK_SYS source. Because the change of CK_SYS has inherent latency, software should read SCSS to confirm whether the switching is complete or not. The switch will be forced to IRC8M when leaving Deep-sleep and Standby mode or HXTAL failure is detected by HXTAL clock monitor when HXTAL is selected directly or indirectly as the clock source of CK_SYS. 00: Select CK_IRC8M as the CK_SYS source 01: Select CK_HXTAL as the CK_SYS source 10: Select CK_PLL as the CK_SYS source 11: Reserved

5.3.3. Clock interrupt register (RCU_INT)

Address offset: 0x08
Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								CKMIC	PLL2 STBIC	PLL1 STBIC	PLL STBIC	HXTAL STBIC	IRC8M STBIC	LXTAL STBIC	IRC40 STBIC K
								w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	PLL2 STBIE	PLL1 STBIE	PLL STBIE	HXTAL STBIE	IRC8M STBIE	LXTAL STBIE	IRC40K STBIE	CKMIF	PLL2 STBIF	PLL1 STBIF	PLL STBIF	HXTAL STBIF	IRC8M STBIF	LXTAL STBIF	IRC40K STBIF
	rw	rw	rw	rw	rw	rw	rw	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23	CKMIC	HXTAL clock stuck interrupt clear Write 1 by software to reset the CKMIF flag. 0: Not reset CKMIF flag 1: Reset CKMIF flag
22	PLL2STBIC	PLL2 stabilization interrupt clear Write 1 by software to reset the PLL2STBIF flag. 0: Not reset PLL2STBIF flag 1: Reset PLL2STBIF flag
21	PLL1STBIC	PLL1 stabilization interrupt clear Write 1 by software to reset the PLL1STBIF flag. 0: Not reset PLL1STBIF flag 1: Reset PLL1STBIF flag
20	PLLSTBIC	PLL stabilization interrupt clear Write 1 by software to reset the PLLSTBIF flag. 0: Not reset PLLSTBIF flag 1: Reset PLLSTBIF flag
19	HXTALSTBIC	HXTAL stabilization interrupt clear Write 1 by software to reset the HXTALSTBIF flag. 0: Not reset HXTALSTBIF flag 1: Reset HXTALSTBIF flag
18	IRC8MSTBIC	IRC8M stabilization interrupt clear Write 1 by software to reset the IRC8MSTBIF flag. 0: Not reset IRC8MSTBIF flag 1: Reset IRC8MSTBIF flag
17	LXTALSTBIC	LXTAL stabilization interrupt clear Write 1 by software to reset the LXTALSTBIF flag. 0: Not reset LXTALSTBIF flag 1: Reset LXTALSTBIF flag
16	IRC40KSTBIC	IRC40K stabilization interrupt clear Write 1 by software to reset the IRC40KSTBIF flag. 0: Not reset IRC40KSTBIF flag 1: Reset IRC40KSTBIF flag

15	Reserved	Must be kept at reset value.
14	PLL2STBIE	PLL2 stabilization interrupt enable Set and reset by software to enable/disable the PLL2 stabilization interrupt. 0: Disable the PLL2 stabilization interrupt 1: Enable the PLL2 stabilization interrupt
13	PLL1STBIE	PLL1 stabilization interrupt enable Set and reset by software to enable/disable the PLL1 stabilization interrupt. 0: Disable the PLL1 stabilization interrupt 1: Enable the PLL1 stabilization interrupt
12	PLLSTBIE	PLL stabilization interrupt enable Set and reset by software to enable/disable the PLL stabilization interrupt. 0: Disable the PLL stabilization interrupt 1: Enable the PLL stabilization interrupt
11	HXTALSTBIE	HXTAL stabilization interrupt enable Set and reset by software to enable/disable the HXTAL stabilization interrupt 0: Disable the HXTAL stabilization interrupt 1: Enable the HXTAL stabilization interrupt
10	IRC8MSTBIE	IRC8M stabilization interrupt enable Set and reset by software to enable/disable the IRC8M stabilization interrupt 0: Disable the IRC8M stabilization interrupt 1: Enable the IRC8M stabilization interrupt
9	LXTALSTBIE	LXTAL stabilization interrupt enable LXTAL stabilization interrupt enable/disable control 0: Disable the LXTAL stabilization interrupt 1: Enable the LXTAL stabilization interrupt
8	IRC40KSTBIE	IRC40K stabilization interrupt enable IRC40K stabilization interrupt enable/disable control 0: Disable the IRC40K stabilization interrupt 1: Enable the IRC40K stabilization interrupt
7	CKMIF	HXTAL clock stuck interrupt flag Set by hardware when the HXTAL clock is stuck. Reset when setting the CKMIC bit by software. 0: Clock operating normally 1: HXTAL clock stuck
6	PLL2STBIF	PLL2 stabilization interrupt flag Set by hardware when the PLL2 is stable and the PLL2STBIE bit is set. Reset when setting the PLL2STBIC bit by software. 0: No PLL2 stabilization interrupt generated 1: PLL2 stabilization interrupt generated

5	PLL1STBIF	PLL1 stabilization interrupt flag Set by hardware when the PLL1 is stable and the PLL1STBIE bit is set. Reset when setting the PLL1STBIC bit by software. 0: No PLL1 stabilization interrupt generated 1: PLL1 stabilization interrupt generated
4	PLLSTBIF	PLL stabilization interrupt flag Set by hardware when the PLL is stable and the PLLSTBIE bit is set. Reset when setting the PLLSTBIC bit by software. 0: No PLL stabilization interrupt generated 1: PLL stabilization interrupt generated
3	HXTALSTBIF	HXTAL stabilization interrupt flag Set by hardware when the High speed 4 ~ 32 MHz crystal oscillator clock is stable and the HXTALSTBIE bit is set. Reset when setting the HXTALSTBIC bit by software. 0: No HXTAL stabilization interrupt generated 1: HXTAL stabilization interrupt generated
2	IRC8MSTBIF	IRC8M stabilization interrupt flag Set by hardware when the Internal 8 MHz RC oscillator clock is stable and the IRC8MSTBIE bit is set. Reset when setting the IRC8MSTBIC bit by software. 0: No IRC8M stabilization interrupt generated 1: IRC8M stabilization interrupt generated
1	LXTALSTBIF	LXTAL stabilization interrupt flag Set by hardware when the Low speed 32,768 Hz crystal oscillator clock is stable and the LXTALSTBIE bit is set. Reset when setting the LXTALSTBIC bit by software. 0: No LXTAL stabilization interrupt generated 1: LXTAL stabilization interrupt generated
0	IRC40KSTBIF	IRC40K stabilization interrupt flag Set by hardware when the Internal 40kHz RC oscillator clock is stable and the IRC40KSTBIE bit is set. Reset when setting the IRC40KSTBIC bit by software. 0: No IRC40K stabilization clock ready interrupt generated 1: IRC40K stabilization interrupt generated

5.3.4. APB2 reset register (RCU_APB2RST)

Address offset: 0x0C

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

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

Reserved										TIMER10 RST	TIMER9 RST	TIMER8 RST	Reserved		
										rw	rw	rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	USART0 RST	TIMER7 RST	SPI0RST	TIMER0 RST	ADC1 RST	ADC0 RST	Reserved		PERST	PDRST	PCRST	PBRST	PARST	Reserved	AFRST
	rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw		rw

Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.
21	TIMER10RST	Timer 10 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER10
20	TIMER9RST	Timer 9 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER9
19	TIMER8RST	Timer 8 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER8
18:15	Reserved	Must be kept at reset value.
14	USART0RST	USART0 Reset This bit is set and reset by software. 0: No reset 1: Reset the USART0
13	TIMER7RST	Timer 7 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER7
12	SPI0RST	SPI0 reset This bit is set and reset by software. 0: No reset 1: Reset the SPI0
11	TIMER0RST	Timer 0 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER0
10	ADC1RST	ADC1 reset

		This bit is set and reset by software. 0: No reset 1: Reset the ADC1
9	ADC0RST	ADC0 reset This bit is set and reset by software. 0: No reset 1: Reset the ADC0
8:7	Reserved	Must be kept at reset value.
6	PERST	GPIO port E reset This bit is set and reset by software. 0: No reset 1: Reset the GPIO port E
5	PDRST	GPIO port D reset This bit is set and reset by software. 0: No reset 1: Reset the GPIO port D
4	PCRST	GPIO port C reset This bit is set and reset by software. 0: No reset 1: Reset the GPIO port C
3	PBRST	GPIO port B reset This bit is set and reset by software. 0: No reset 1: Reset the GPIO port B
2	PARST	GPIO port A reset This bit is set and reset by software. 0: No reset 1: Reset the GPIO port A
1	Reserved	Must be kept at reset value.
0	AFRST	Alternate function I/O reset This bit is set and reset by software. 0: No reset 1: Reset Alternate Function I/O

5.3.5. APB1 reset register (RCU_APB1RST)

Address offset: 0x10
Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DACRST	PMURST	BKPIRST	CAN1 RST	CAN0 RST	Reserved		I2C1RST	I2C0RST	UART4 RST	UART3 RST	USART2 RST	USART1 RST	Reserved
		rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI2RST	SPI1RST	Reserved		WWDGT RST	Reserved		TIMER13 RST	TIMER12 RST	TIMER11 RST	TIMER6 RST	TIMER5 RST	TIMER4 RST	TIMER3 RST	TIMER2 RST	TIMER1 RST
rw	rw			rw			rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29	DACRST	DAC reset This bit is set and reset by software. 0: No reset 1: Reset DAC unit
28	PMURST	Power control reset This bit is set and reset by software. 0: No reset 1: Reset power control unit
27	BKPIRST	Backup interface reset This bit is set and reset by software. 0: No reset 1: Reset backup interface
26	CAN1RST	CAN1 reset This bit is set and reset by software. 0: No reset 1: Reset the CAN1
25	CAN0RST	CAN0 reset This bit is set and reset by software. 0: No reset 1: Reset the CAN0
24:23	Reserved	Must be kept at reset value.
22	I2C1RST	I2C1 reset This bit is set and reset by software. 0: No reset 1: Reset the I2C1
21	I2C0RST	I2C0 reset This bit is set and reset by software. 0: No reset 1: Reset the I2C0

20	UART4RST	UART4 reset This bit is set and reset by software. 0: No reset 1: Reset the UART4
19	UART3RST	UART3 reset This bit is set and reset by software. 0: No reset 1: Reset the UART3
18	USART2RST	USART2 reset This bit is set and reset by software. 0: No reset 1: Reset the USART2
17	USART1RST	USART1 reset This bit is set and reset by software. 0: No reset 1: Reset the USART1
16	Reserved	Must be kept at reset value.
15	SPI2RST	SPI2 reset This bit is set and reset by software. 0: No reset 1: Reset the SPI2
14	SPI1RST	SPI1 reset This bit is set and reset by software. 0: No reset 1: Reset the SPI1
13:12	Reserved	Must be kept at reset value.
11	WWDGTRST	WWDGT reset This bit is set and reset by software. 0: No reset 1: Reset the WWDGT
10:9	Reserved	Must be kept at reset value.
8	TIMER13RST	TIMER13 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER13
7	TIMER12RST	TIMER12 reset This bit is set and reset by software. 0: No reset

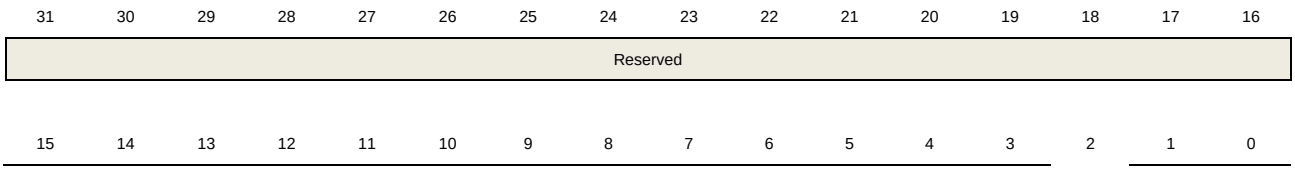
		1: Reset the TIMER12
6	TIMER11RST	TIMER11 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER11
5	TIMER6RST	TIMER6 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER6
4	TIMER5RST	TIMER5 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER5
3	TIMER4RST	TIMER4 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER4
2	TIMER3RST	TIMER3 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER3
1	TIMER2RST	TIMER2 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER2
0	TIMER1RST	TIMER1 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER1

5.3.6. AHB enable register (RCU_AHBEN)

Address offset: 0x14

Reset value: 0x0000 0014

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)



Reserved	USBFSEN	Reserved	EXMCEN	Reserved	CRCEN	Reserved	FMCSPEN	Reserved	SRAMSPEN	DMA1EN	DMA0EN
	rw		rw		rw		rw		rw	rw	rw

Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12	USBFSEN	USBFS clock enable This bit is set and reset by software. 0: Disabled USBFS clock 1: Enabled USBFS clock
11:9	Reserved	Must be kept at reset value.
8	EXMCEN	EXMC clock enable This bit is set and reset by software. 0: Disabled EXMC clock 1: Enabled EXMC clock
7	Reserved	Must be kept at reset value.
6	CRCEN	CRC clock enable This bit is set and reset by software. 0: Disabled CRC clock 1: Enabled CRC clock
5	Reserved	Must be kept at reset value.
4	FMCSPEN	FMC clock enable when sleep mode This bit is set and reset by software to enable/disable FMC clock during Sleep mode. 0: Disabled FMC clock during Sleep mode 1: Enabled FMC clock during Sleep mode
3	Reserved	Must be kept at reset value.
2	SRAMSPEN	SRAM interface clock enable when sleep mode This bit is set and reset by software to enable/disable SRAM interface clock during Sleep mode. 0: Disabled SRAM interface clock during Sleep mode. 1: Enabled SRAM interface clock during Sleep mode
1	DMA1EN	DMA1 clock enable This bit is set and reset by software. 0: Disabled DMA1 clock 1: Enabled DMA1 clock
0	DMA0EN	DMA0 clock enable This bit is set and reset by software. 0: Disabled DMA0 clock

1: Enabled DMA0 clock

5.3.7. APB2 enable register (RCU_APB2EN)

Address offset: 0x18

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										TIMER10EN	TIMER9EN	TIMER8EN	Reserved		
										rw	rw	rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	USART0EN	TIMER7EN	SPI0EN	TIMER0EN	ADC1EN	ADC0EN	Reserved	Reserved	PEEN	PDEN	PCEN	PBEN	PAEN	Reserved	AFEN
	rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw		rw

Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.
21	TIMER10EN	TIMER10 clock enable This bit is set and reset by software. 0: Disabled TIMER10 clock 1: Enabled TIMER10 clock
20	TIMER9EN	TIMER9 clock enable This bit is set and reset by software. 0: Disabled TIMER9 clock 1: Enabled TIMER9 clock
19	TIMER8EN	TIMER8 clock enable This bit is set and reset by software. 0: Disabled TIMER8 clock 1: Enabled TIMER8 clock
18:15	Reserved	Must be kept at reset value.
14	USART0EN	USART0 clock enable This bit is set and reset by software. 0: Disabled USART0 clock 1: Enabled USART0 clock
13	TIMER7EN	TIMER7 clock enable This bit is set and reset by software. 0: Disabled TIMER7 clock 1: Enabled TIMER7 clock
12	SPI0EN	SPI0 clock enable

		This bit is set and reset by software. 0: Disabled SPI0 clock 1: Enabled SPI0 clock
11	TIMER0EN	TIMER0 clock enable This bit is set and reset by software. 0: Disabled TIMER0 clock 1: Enabled TIMER0 clock
10	ADC1EN	ADC1 clock enable This bit is set and reset by software. 0: Disabled ADC1 clock 1: Enabled ADC1 clock
9	ADC0EN	ADC0 clock enable This bit is set and reset by software. 0: Disabled ADC0 clock 1: Enabled ADC0 clock
8:7	Reserved	Must be kept at reset value.
6	PEEN	GPIO port E clock enable This bit is set and reset by software. 0: Disabled GPIO port E clock 1: Enabled GPIO port E clock
5	PDEN	GPIO port D clock enable This bit is set and reset by software. 0: Disabled GPIO port D clock 1: Enabled GPIO port D clock
4	PCEN	GPIO port C clock enable This bit is set and reset by software. 0: Disabled GPIO port C clock 1: Enabled GPIO port C clock
3	PBEN	GPIO port B clock enable This bit is set and reset by software. 0: Disabled GPIO port B clock 1: Enabled GPIO port B clock
2	PAEN	GPIO port A clock enable This bit is set and reset by software. 0: Disabled GPIO port A clock 1: Enabled GPIO port A clock
1	Reserved	Must be kept at reset value.
0	AFEN	Alternate function IO clock enable This bit is set and reset by software.

0: Disabled Alternate Function IO clock

1: Enabled Alternate Function IO clock

5.3.8. APB1 enable register (RCU_APB1EN)

Address offset: 0x1C

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DACEN	PMUEN	BKPIEN	CAN1EN	CAN0EN	Reserved		I2C1EN	I2C0EN	UART4EN	UART3EN	USART2EN	USART1EN	Reserved
		rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI2EN	SPI1EN	Reserved		WWDGTEN	Reserved		TIMER13EN	TIMER12EN	TIMER11EN	TIMER6EN	TIMER5EN	TIMER4EN	TIMER3EN	TIMER2EN	TIMER1EN
rw	rw			rw			rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29	DACEN	DAC clock enable This bit is set and reset by software. 0: Disabled DAC clock 1: Enabled DAC clock
28	PMUEN	PMU clock enable This bit is set and reset by software. 0: Disabled PMU clock 1: Enabled PMU clock
27	BKPIEN	Backup interface clock enable This bit is set and reset by software. 0: Disabled Backup interface clock 1: Enabled Backup interface clock
26	CAN1EN	CAN1 clock enable This bit is set and reset by software. 0: Disabled CAN1 clock 1: Enabled CAN1 clock
25	CAN0EN	CAN0 clock enable This bit is set and reset by software. 0: Disabled CAN0 clock 1: Enabled CAN0 clock
24:23	Reserved	Must be kept at reset value.

22	I2C1EN	I2C1 clock enable This bit is set and reset by software. 0: Disabled I2C1 clock 1: Enabled I2C1 clock
21	I2C0EN	I2C0 clock enable This bit is set and reset by software. 0: Disabled I2C0 clock 1: Enabled I2C0 clock
20	UART4EN	UART4 clock enable This bit is set and reset by software. 0: Disabled UART4 clock 1: Enabled UART4 clock
19	UART3EN	UART3 clock enable This bit is set and reset by software. 0: Disabled UART3 clock 1: Enabled UART3 clock
18	USART2EN	USART2 clock enable This bit is set and reset by software. 0: Disabled USART2 clock 1: Enabled USART2 clock
17	USART1EN	USART1 clock enable This bit is set and reset by software. 0: Disabled USART1 clock 1: Enabled USART1 clock
16	Reserved	Must be kept at reset value.
15	SPI2EN	SPI2 clock enable This bit is set and reset by software. 0: Disabled SPI2 clock 1: Enabled SPI2 clock
14	SPI1EN	SPI1 clock enable This bit is set and reset by software. 0: Disabled SPI1 clock 1: Enabled SPI1 clock
13:12	Reserved	Must be kept at reset value.
11	WWDGTEN	WWDGT clock enable This bit is set and reset by software. 0: Disabled WWDGT clock 1: Enabled WWDGT clock

10:9	Reserved	Must be kept at reset value.
8	TIMER13EN	TIMER13 clock enable This bit is set and reset by software. 0: Disabled TIMER13 clock 1: Enabled TIMER13 clock
7	TIMER12EN	TIMER12 clock enable This bit is set and reset by software. 0: Disabled TIMER12 clock 1: Enabled TIMER12 clock
6	TIMER11EN	TIMER11 clock enable This bit is set and reset by software. 0: Disabled TIMER11 clock 1: Enabled TIMER11 clock
5	TIMER6EN	TIMER6 clock enable This bit is set and reset by software. 0: Disabled TIMER6 clock 1: Enabled TIMER6 clock
4	TIMER5EN	TIMER5 clock enable This bit is set and reset by software. 0: Disabled TIMER5 clock 1: Enabled TIMER5 clock
3	TIMER4EN	TIMER4 clock enable This bit is set and reset by software. 0: Disabled TIMER4 clock 1: Enabled TIMER4 clock
2	TIMER3EN	TIMER3 clock enable This bit is set and reset by software. 0: Disabled TIMER3 clock 1: Enabled TIMER3 clock
1	TIMER2EN	TIMER2 clock enable This bit is set and reset by software. 0: Disabled TIMER2 clock 1: Enabled TIMER2 clock
0	TIMER1EN	TIMER1 clock enable This bit is set and reset by software. 0: Disabled TIMER1 clock 1: Enabled TIMER1 clock

5.3.9. Backup domain control register (RCU_BDCTL)

Address offset: 0x20

Reset value: 0x0000 0018, reset by Backup domain Reset.

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

Note: The LXTALEN, LXTALBPS, RTCSRC and RTCEN bits of the Backup domain control register (RCU_BDCTL) are only reset after a Backup domain Reset. These bits can be modified only when the BKPWEN bit in the Power control register (PMU_CTL) is set.

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCEN	Reserved					RTCSRC[1:0]		Reserved			LXTALDRI[1:0]		LXTAL BPS	LXTAL STB	LXTAL EN
rw						rw					rw		rw	r	rw

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	BKPRST	Backup domain reset This bit is set and reset by software. 0: No reset 1: Resets Backup domain
15	RTCEN	RTC clock enable This bit is set and reset by software. 0: Disabled RTC clock 1: Enabled RTC clock
14:10	Reserved	Must be kept at reset value.
9:8	RTCSRC[1:0]	RTC clock entry selection Set and reset by software to control the RTC clock source. Once the RTC clock source has been selected, it cannot be changed anymore unless the Backup domain is reset. 00: No clock selected 01: CK_LXTAL selected as RTC source clock 10: CK_IRC40K selected as RTC source clock 11: (CK_HXTAL / 128) selected as RTC source clock
7:5	Reserved	Must be kept at reset value.
4:3	LXTALDRI[1:0]	LXTAL drive capability Set and reset by software. This value will be set 0 when Backup domain reset 00: Lower driving capability 01: Medium low driving capability

		10: Medium high driving capability 11: Higher driving capability (reset value) **Note:** The LXTALDRI can not be used in bypass mode.
2	LXTALBPS	LXTAL bypass mode enable Set and reset by software. 0: Disable the LXTAL Bypass mode 1: Enable the LXTAL Bypass mode
1	LXTALSTB	Low speed crystal oscillator stabilization flag Set by hardware to indicate if the LXTAL output clock is stable and ready for use. 0: LXTAL is not stable 1: LXTAL is stable
0	LXTALEN	LXTAL enable Set and reset by software. 0: Disable LXTAL 1: Enable LXTAL

5.3.10. Reset source/clock register (RCU_RSTSCK)

Address offset: 0x24

Reset value: 0x0C00 0000, ALL reset flags reset by power reset only, RSTFC/IRC40KEN reset by system reset.

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LP RSTF	WWDGT RSTF	FWDGT RSTF	SW RSTF	POR RSTF	EP RSTF	Reserved	RSTFC	Reserved							
r	r	r	r	r	r		rw								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														IRC40K STB	IRC40KEN
														r	rw

Bits	Fields	Descriptions
31	LPRSTF	Low-power reset flag Set by hardware when Deep-sleep /standby reset generated. Reset by writing 1 to the RSTFC bit. 0: No Low-power management reset generated 1: Low-power management reset generated
30	WWDGTRSTF	Window watchdog timer reset flag Set by hardware when a window watchdog timer reset generated. Reset by writing 1 to the RSTFC bit. 0: No window watchdog reset generated

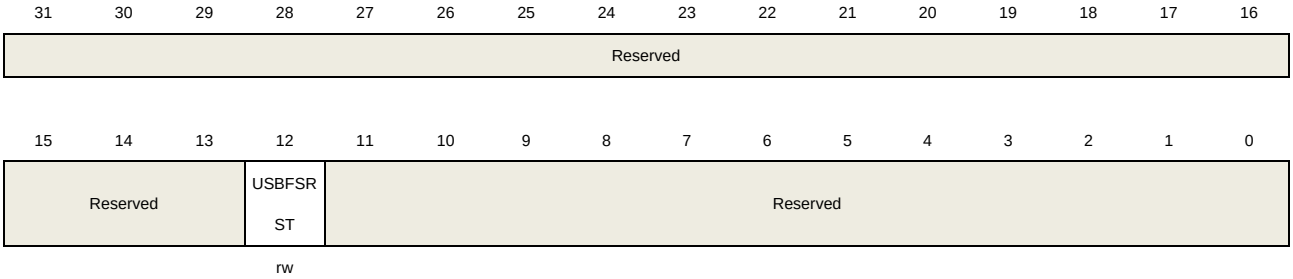
		1: Window watchdog reset generated
29	FWDGTRSTF	Free watchdog timer reset flag Set by hardware when a free watchdog timer reset generated. Reset by writing 1 to the RSTFC bit. 0: No free watchdog timer reset generated 1: Free Watchdog timer reset generated
28	SWRSTF	Software reset flag Set by hardware when a software reset generated. Reset by writing 1 to the RSTFC bit. 0: No software reset generated 1: Software reset generated
27	PORRSTF	Power reset flag Set by hardware when a Power reset generated. Reset by writing 1 to the RSTFC bit. 0: No Power reset generated 1: Power reset generated
26	EPRSTF	External PIN reset flag Set by hardware when an External PIN reset generated. Reset by writing 1 to the RSTFC bit. 0: No External PIN reset generated 1: External PIN reset generated
25	Reserved	Must be kept at reset value.
24	RSTFC	Reset flag clear This bit is set by software to clear all reset flags. 0: Not clear reset flags 1: Clear reset flags
23:2	Reserved	Must be kept at reset value.
1	IRC40KSTB	IRC40K stabilization flag Set by hardware to indicate if the IRC40K output clock is stable and ready for use. 0: IRC40K is not stable 1: IRC40K is stable
0	IRC40KEN	IRC40K enable Set and reset by software. 0: Disable IRC40K 1: Enable IRC40K

5.3.11. AHB reset register (RCU_AHBRST)

Address offset: 0x28

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)



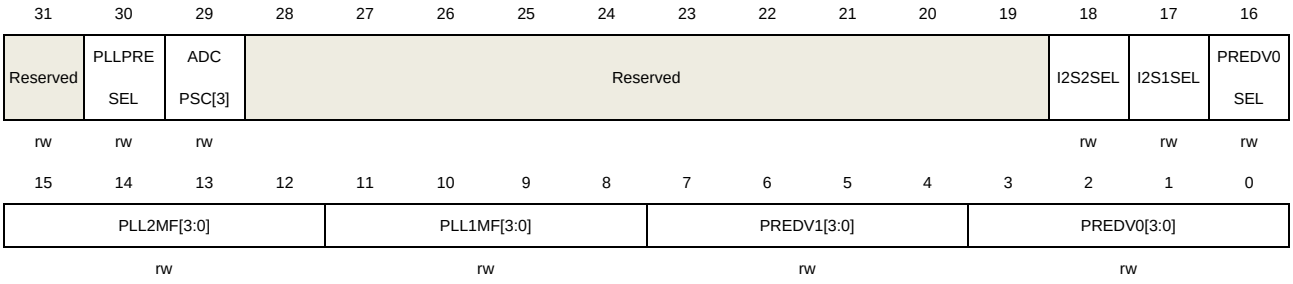
Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12	USBFSRST	USBFS reset This bit is set and reset by software. 0: No reset 1: Reset the USBFS
11:0	Reserved	Must be kept at reset value.

5.3.12. Clock configuration register 1 (RCU_CFG1)

Address offset: 0x2C

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)



Bits	Fields	Descriptions
31	Reserverd	Must be kept at reset value.
30	PLLPRESEL	PLL clock source selection 0: HXTAL selected as PLL source clock 1: CK_IRC48M selected as PLL source clock
29	ADCPSC[3]	Bit 4 of ADCPSC see bits 15:14 of RCU_CFG0 and bit 28 of RCU_CFG0
28:19	Reserved	Must be kept at reset value.
18	I2S2SEL	I2S2 clock source selection Set and reset by software to control the I2S2 clock source.

		0: System clock selected as I2S2 source clock 1: (CK_PLL2 x 2) selected as I2S2 source clock
17	I2S1SEL	I2S1 clock source selection Set and reset by software to control the I2S1 clock source. 0: System clock selected as I2S1 source clock 1: (CK_PLL2 x 2) selected as I2S1 source clock
16	PREDV0SEL	PREDV0 input clock source selection Set and reset by software. 0: HXTAL or IRC48M selected as PREDV0 input source clock 1: CK_PLL1 selected as PREDV0 input source clock
15:12	PLL2MF[3:0]	The PLL2 clock multiplication factor These bits are written by software to define the PLL2 multiplication factor. 00xx: reserve 010x: reserve 0110: (PLL2 source clock x 8) 0111: (PLL2 source clock x 9) 1000 :(PLL2 source clock x 10) 1001: (PLL2 source clock x 11) 1010: (PLL2 source clock x 12) 1011: (PLL2 source clock x 13) 1100: (PLL2 source clock x 14) 1101: reserve 1110: (PLL2 source clock x 16) 1111: (PLL2 source clock x 20)
11:8	PLL1MF[3:0]	The PLL1 clock multiplication factor Set and reset by software. 00xx: reserve 010x: reserve 0110: (PLL1 source clock x 8) 0111: (PLL1 source clock x 9) 1000 :(PLL1 source clock x 10) 1001: (PLL1 source clock x 11) 1010: (PLL1 source clock x 12) 1011: (PLL1 source clock x 13) 1100: (PLL1 source clock x 14) 1101: reserve 1110 :(PLL1 source clock x 16) 1111: (PLL1 source clock x 20)
7:4	PREDV1[3:0]	PREDV1 division factor This bit is set and reset by software. These bits can be written when PLL1 and PLL2 are disable

0000: PREDV1 input source clock not divided
0001: PREDV1 input source clock divided by 2
0010: PREDV1 input source clock divided by 3
0011: PREDV1 input source clock divided by 4
0100: PREDV1 input source clock divided by 5
0101: PREDV1 input source clock divided by 6
0110: PREDV1 input source clock divided by 7
0111: PREDV1 input source clock divided by 8
1000: PREDV1 input source clock divided by 9
1001: PREDV1 input source clock divided by 10
1010: PREDV1 input source clock divided by 11
1011: PREDV1 input source clock divided by 12
1100: PREDV1 input source clock divided by 13
1101: PREDV1 input source clock divided by 14
1110: PREDV1 input source clock divided by 15
1111: PREDV1 input source clock divided by 16

3:0	PREDV0[3:0]	PREDV0 division factor This bit is set and reset by software. These bits can be written when PLL is disable. **Note**: The bit 0 of PREDV0 is same as bit 17 of RCU_CFG0, so modifying Bit 17 of RCU_CFG0 also modifies bit 0 of RCU_CFG1. 0000: PREDV0 input source clock not divided 0001: PREDV0 input source clock divided by 2 0010: PREDV0 input source clock divided by 3 0011: PREDV0 input source clock divided by 4 0100: PREDV0 input source clock divided by 5 0101: PREDV0 input source clock divided by 6 0110: PREDV0 input source clock divided by 7 0111: PREDV0 input source clock divided by 8 1000: PREDV0 input source clock divided by 9 1001: PREDV0 input source clock divided by 10 1010: PREDV0 input source clock divided by 11 1011: PREDV0 input source clock divided by 12 1100: PREDV0 input source clock divided by 13 1101: PREDV0 input source clock divided by 14 1110: PREDV0 input source clock divided by 15 1111: PREDV0 input source clock divided by 16

5.3.13. Deep-sleep mode voltage register (RCU_DSV)

Address offset: 0x34
Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

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

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														DSLPVS[1:0]	
														rw	

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1:0	DSLPVS[1:0]	Deep-sleep mode voltage select These bits are set and reset by software 00 : The core voltage is 1.0V in Deep-sleep mode 01 : The core voltage is 0.9V in Deep-sleep mode 10 : The core voltage is 0.8V in Deep-sleep mode 11 : The core voltage is 1.2V in Deep-sleep mode

5.3.14. Additional clock control register (RCU_ADDCTL)

Address offset: 0xC0

Reset value: 0x8000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IRC48MCALIB[7:0]								Reserved						IRC48MSTB	IRC48MEN
r														r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CK48MSEL
															rw

Bits	Fields	Descriptions
31:24	IRC48MCALIB [7:0]	Internal 48MHz RC oscillator calibration value register These bits are load automatically at power on.
23:18	Reserved	Must be kept at reset value.
17	IRC48MSTB	Internal 48MHz RC oscillator clock stabilization flag Set by hardware to indicate if the IRC48M oscillator is stable and ready for use. 0: IRC48M is not stable 1: IRC48M is stable
16	IRC48MEN	Internal 48MHz RC oscillator enable Set and reset by software. Reset by hardware when entering Deep-sleep or Standby mode. 0: IRC48M disable

		1: IRC48M enable
15:1	Reserved	Must be kept at reset value.
0	CK48MSEL	48MHz clock selection Set and reset by software. This bit used to generate CK48M clock which select IRC48M clock or PLL48M clock. 0: Don't select IRC48M clock(use CK_PLL clock) 1: Select IRC48M clock

5.3.15. Additional clock interrupt register (RCU_ADDINT)

Address offset: 0xCC

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									IRC48MSTBIC	Reserved					
									w						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	IRC48MSTBIE	Reserved							IRC48MSTBIF	Reserved					
	rw								r						

Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22	IRC48MSTBIC	Internal 48 MHz RC oscillator stabilization Interrupt clear Write 1 by software to reset the IRC48MSTBIF flag. 0: Not reset IRC48MSTBIF flag 1: Reset IRC48MSTBIF flag
21:15	Reserved	Must be kept at reset value.
14	IRC48MSTBIE	Internal 48 MHz RC oscillator stabilization Interrupt enable Set and reset by software to enable/disable the IRC48M stabilization interrupt 0: Disable the IRC48M stabilization interrupt 1: Enable the IRC48M stabilization interrupt
13:7	Reserved	Must be kept at reset value.
6	IRC48MSTBIF	IRC48M stabilization interrupt flag Set by hardware when the Internal 48 MHz RC oscillator clock is stable and the IRC48MSTBIE bit is set. Reset by software when setting the IRC48MSTBIC bit. 0: No IRC48M stabilization interrupt generated

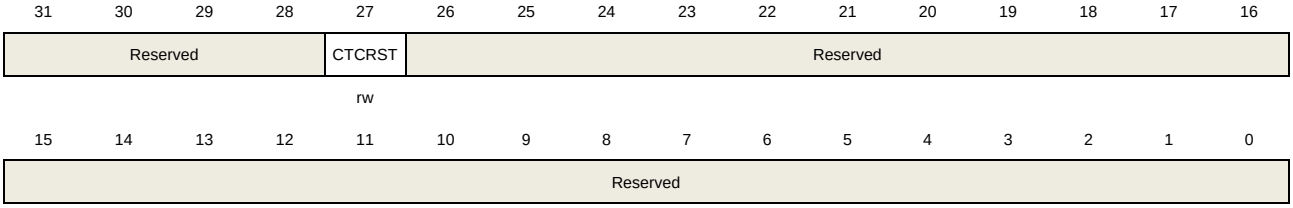
		1: IRC48M stabilization interrupt generated
5:0	Reserved	Must be kept at reset value.

5.3.16. APB1 additional reset register (RCU_ADDAPB1RST)

Address offset: 0xE0

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)



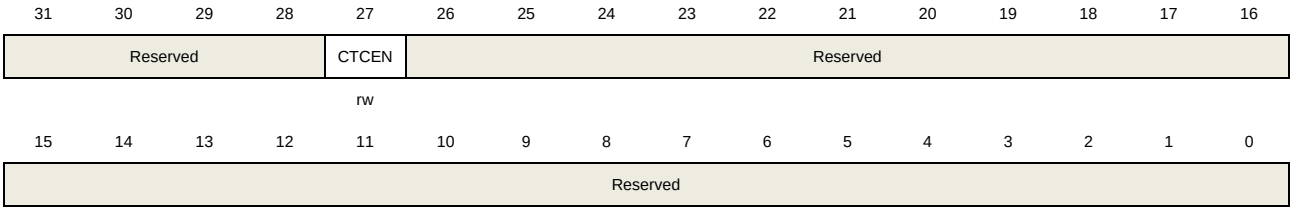
Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27	CTCRST	CTC reset This bit is set and reset by software. 0: No reset 1: Reset CTC
26:0	Reserved	Must be kept at reset value.

5.3.17. APB1 additional enable register (RCU_ADDAPB1EN)

Address offset: 0xE4

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)



Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27	CTCEN	CTC clock enable This bit is set and reset by software. 0: Disabled CTC clock

		1: Enabled CTC clock
26:0	Reserved	Must be kept at reset value.

6. Clock trim controller (CTC)

6.1. Overview

The clock trim controller (CTC) is used to trim internal 48MHz RC oscillator (IRC48M) automatically by hardware. The CTC unit trims the frequency of the IRC48M which is based on an external accurate reference signal source. It can adjust the calibration value to provide a precise IRC48M clock automatically or manually.

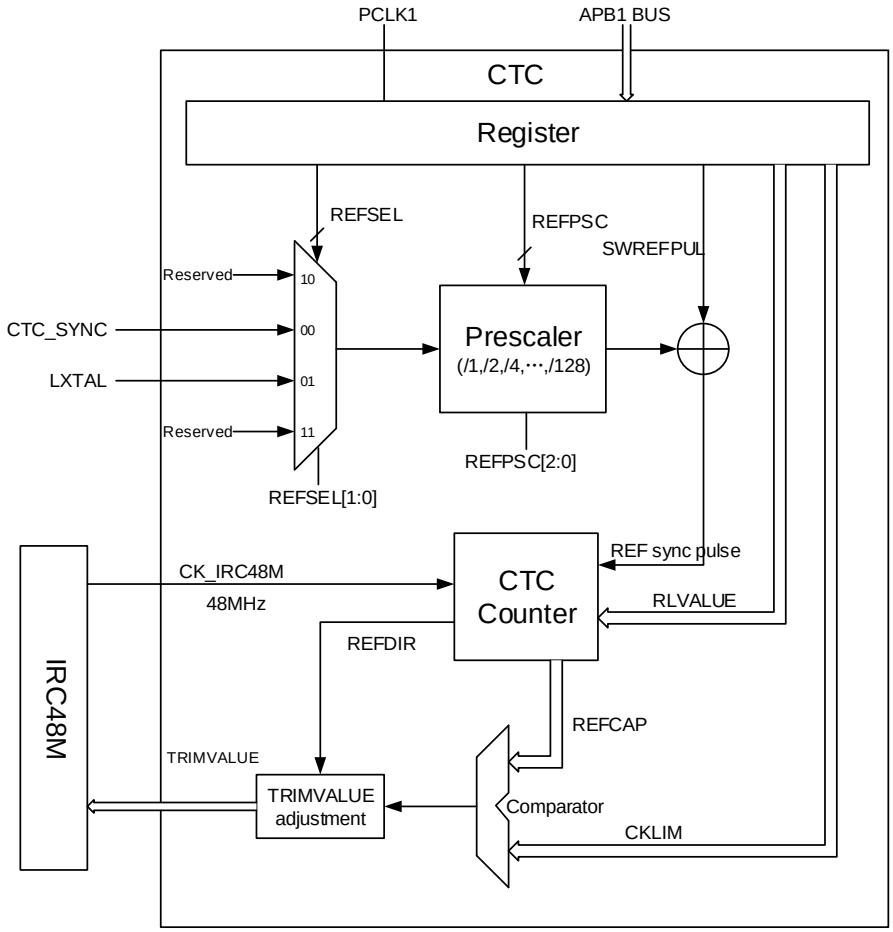
6.2. Characteristics

- Two external reference signal sources: GPIO(CTC_SYNC), LXTAL clock.
- Provide software reference sync pulse.
- Trimmed by hardware without any software action automatically.
- 16 bits trim counter with reference signal source capture and reload function.
- 8 bits clock trim base value used for frequency evaluation and automatic trim.
- Flags or interrupts to indicate whether the clock trim status is OK (CKOKIF), warning (CKWARNIF) or error (ERRIF).

6.3. Function overview

Figure 6-1. CTC overview provides details on the internal configuration of the CTC.

Figure 6-1. CTC overview



6.3.1. Reference sync pulse generator

Firstly, the reference signal source can select GPIO(CTC_SYNC), LXTAL clock by setting REFSEL bits in CTC_CTL1 register.

Secondly, the selected reference signal source uses a configurable polarity by setting REFPOL bit in CTC_CTL1 register, and can be divided to a suitable frequency with a configurable prescaler by setting REFPSC bits in CTC_CTL1 register.

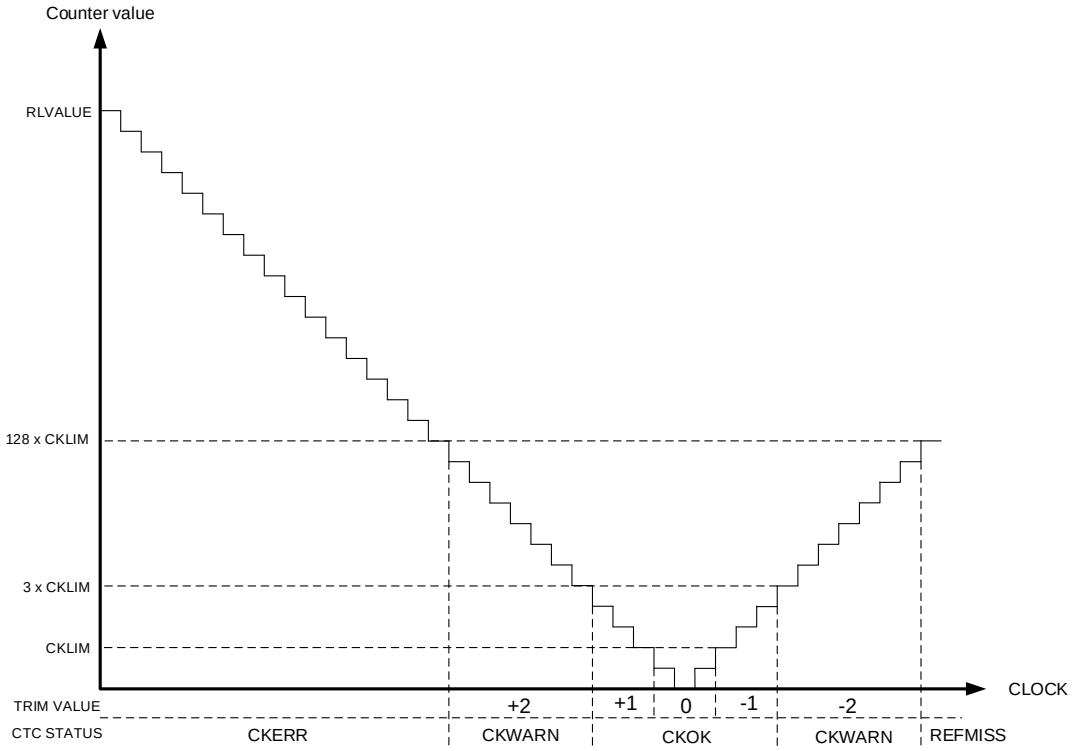
Thirdly, if a software reference pulse is needed, write 1 to SWREFPUL bit in CTC_CTL0 register. The software reference pulse generated in last step is logical OR with the external reference pulse.

6.3.2. CTC trim counter

The CTC trim counter is clocked by CK_IRC48M. After the CNTEN bit in CTC_CTL0 register is set, and a first REF sync pulse is detected, the counter starts down-counting from RLVALUE (defined in CTC_CTL1 register). If any REF sync pulse is detected, the counter reloads the RLVALUE and starts down-counting again. If no REF sync pulse is detected, the

counter down-counts to zero, and then up-counts to 128 x CKLIM (defined in CTC_CTL1 register), and then stops until next REF sync pulse is detected. If any REF sync pulse is detected, the current CTC trim counter value is captured to REFCAP in status register (CTC_STAT), and the counter direction is captured to REFDIR in status register (CTC_STAT). The detail showing in ***Figure 6-2. CTC trim counter***.

Figure 6-2. CTC trim counter



6.3.3. Frequency evaluation and automatic trim process

The clock frequency evaluation is performed when a REF sync pulse occurs. If a REF sync pulse occurs on down-counting, it means the current clock is slower than correct clock (the frequency of 48M). It needs to increase the TRIMVALUE in CTC_CTL0 register. If a REF sync pulse occurs on up-counting, it means the current clock is faster than correct clock (the frequency of 48M). It needs to reduce the TRIMVALUE in CTC_CTL0 register. The CKOKIF, CKWARNIF, CKERR and REFMISS in CTC_STAT register show the frequency evaluation scope.

If the AUTOTRIM bit in CTC_CTL0 register is set, the automatic hardware trim mode is enabled. In this mode, if a REF sync pulse occurs on down-counting, it means the current clock is slower than correct clock, the TRIMVALUE will be increased automatically to raise the clock frequency. Vice versa when it occurs on up-counting, the TRIMVALUE will be decreased to reduce the clock frequency automatically.

- Counter < CKLIM when REF sync pulse is detected.

  When the CKOKIF in CTC_STAT register is set, an interrupt will be generated if CKOKIE

bit in CTC_CTL0 register is 1.

If the AUTOTRIM bit in CTC_CTL0 register is set, the TRIMVALUE in CTC_CTL0 register is not changed.

- CKLIM ≤ Counter < 3 x CKLIM when REF sync pulse is detected.

  When the CKOKIF in CTC_STAT register is set, an interrupt will be generated if CKOKIE bit in CTC_CTL0 register is 1.

  If the AUTOTRIM bit in CTC_CTL0 register is set, the TRIMVALUE in CTC_CTL0 register adds 1 when down-counting or subtracts 1 when up-counting.

- 3 x CKLIM ≤ Counter < 128 x CKLIM when REF sync pulse is detected.

  When the CKWARNIF in CTC_STAT register is set, an interrupt will be generated if CKWARNIE bit in CTC_CTL0 register is 1.

  If the AUTOTRIM bit in CTC_CTL0 register is set, the TRIMVALUE in CTC_CTL0 register adds 2 when down-counting or subtracts 2 when up-counting.

- Counter ≥ 128 x CKLIM when down-counting and a REF sync pulse is detected.

  When the CKERR in CTC_STAT register is set, an interrupt will be generated if ERRIE bit in CTC_CTL0 register is 1.

  The TRIMVALUE in CTC_CTL0 register is not changed

- Counter = 128 x CKLIM when up-counting.

  When the REFMISS in CTC_STAT register is set, an interrupt will be generated if ERRIE bit in CTC_CTL0 register is 1.

  The TRIMVALUE in CTC_CTL0 register is not changed.

If adjusting the TRIMVALUE in CTC_CTL0 register over 63, the overflow will be occurred, while adjusting the TRIMVALUE under 0, the underflow will be occurred. The TRIMVALUE ranges from 0 to 63 (the TRIMVALUE is 63 if overflow, the TRIMVALUE is 0 if underflow). Then, the TRIMERR in CTC_STAT register will be set, and an interrupt will be generated if ERRIE bit in CTC_CTL0 register is 1.

6.3.4. Software program guide

The RLVALUE and CKLIM bits in CTC_CTL1 register are critical to evaluate the clock frequency and automatic hardware trim. The value is calculated by the correct clock frequency (IRC48M:48 MHz) and the frequency of REF sync pulse. The ideal case is REF sync pulse occurs when the CTC counter is zero, so the RLVALUE is:

$$RLVALUE=(F_{clock} \div F_{REF})-1 \quad (6\text{-}1)$$

The CKLIM is set by user according to the clock accuracy. It is recommend to set it to half of the step size, so the CKLIM is:

$$CKLIM=(F_{clock} \div F_{REF}) \times 0.12\% \div 2 \quad (6\text{-}2)$$

The typical step size is 0.12%. Where the F_{clock} is the frequency of correct clock (IRC48M), the F_{REF} is the frequency of reference sync pulse.

6.4. Register definition

CTC base address: 0x4000 C800

6.4.1. Control register 0 (CTC_CTL0)

Address offset: 0x00

Reset value: 0x0000 2000

This register has to be accessed by word (32-bit)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TRIMVALUE[5:0]						SWREFPUL	AUTOTRIM	CNTEN	Reserved	EREFIE	ERRIE	CKWARNIE	CKOKIE
		rw						w	rw	rw		rw	rw	rw	rw

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13:8	TRIMVALUE[5:0]	IRC48M trim value When AUTOTRIM in CTC_CTL0 register is 0, these bits are set and cleared by software. This mode is used for software calibration. When AUTOTRIM in CTC_CTL0 register is 1, these bits are read only. The value is modified by hardware automatically. This mode is used for trim by hardware. The middle value is 32. When increasing 1, the IRC48M clock frequency adds around 57KHz. When decreasing 1, the IRC48M clock frequency subtracts around 57KHz.
7	SWREFPUL	Software reference source sync pulse This bit is set by software, and a reference sync pulse is generated to CTC counter. This bit is cleared by hardware automatically and read as 0. 0: No effect 1: Generates a software reference source sync pulse
6	AUTOTRIM	Hardware automatic trim mode This bit is set and cleared by software. When this bit is set, the hardware automatic trim is enabled, the TRIMVALUE bits in CTC_CTL0 register are modified by hardware automatically, until the frequency of IRC48M clock is closed to 48MHz. 0: Hardware automatic trim disabled 1: Hardware automatic trim enabled
5	CNTEN	CTC counter enable This bit is set and cleared by software. This bit is used to enable or disable the CTC trim counter. When this bit is set, the CTC_CTL1 register cannot be modified.

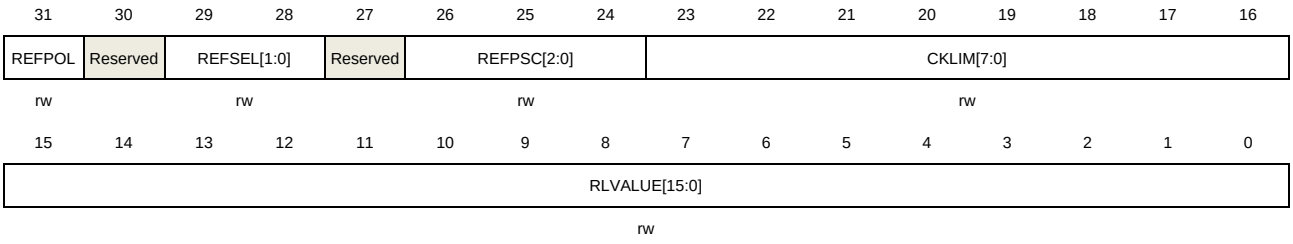
		0: CTC trim counter disabled 1: CTC trim counter enabled.
4	Reserved	Must be kept at reset value.
3	EREFIE	Expected reference (EREFIF) interrupt enable 0: EREFIF interrupt disable 1: EREFIF interrupt enable
2	ERRIE	Error (ERRIF) interrupt enable 0: ERRIF interrupt disable 1: ERRIF interrupt enable
1	CKWARNIE	Clock trim warning (CKWARNIF) interrupt enable 0: CKWARNIF interrupt disable 1: CKWARNIF interrupt enable
0	CKOKIE	Clock trim ok (CKOKIF) interrupt enable 0: CKOKIF interrupt disable 1: CKOKIF interrupt enable

6.4.2. Control register 1 (CTC_CTL1)

Address offset: 0x04
Reset value: 0x2022 BB7F

This register has to be accessed by word (32-bit).

This register cannot be modified when CNTEN is 1.



Bits	Fields	Descriptions
31	REFPOL	Reference signal source polarity This bit is set and cleared by software to select reference signal source polarity 0: Rising edge selected 1: Falling edge selected
30	Reserved	Must be kept at reset value.
29:28	REFSEL[1:0]	Reference signal source selection These bits are set and cleared by software to select reference signal source. 00: GPIO selected 01: LXTAL clock selected

		10: Reserved 11: Reserved.
27	Reserved	Must be kept at reset value.
26:24	REFPSC[2:0]	Reference signal source prescaler These bits are set and cleared by software 000: Reference signal not divided 001: Reference signal divided by 2 010: Reference signal divided by 4 011: Reference signal divided by 8 100: Reference signal divided by 16 101: Reference signal divided by 32 110: Reference signal divided by 64 111: Reference signal divided by 128
23:16	CKLIM[7:0]	Clock trim base limit value These bits are set and cleared by software to define the clock trim base limit value. These bits are used for frequency evaluation and automatic trim process. Please refer to the ***Frequency evaluation and automatic trim process*** for detail.
15:0	RLVALUE[15:0]	CTC counter reload value These bits are set and cleared by software to define the CTC counter reload value. These bits reload to CTC trim counter, when a reference sync pulse is received, so as to start or restart the counter.

6.4.3. Status register (CTC_STAT)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REFDIR	Reserved				TRIM ERR	REF MISS	CKERR	Reserved				EREFIF	ERRIF	CKWARN IF	CKOK IF
r					r	r	r					r	r	r	r

Bits	Fields	Descriptions
31:16	REFCAP[15:0]	CTC counter capture value. When a reference sync pulse occurs, the CTC trim counter value is captured to REFCAP bits.
15	REFDIR	CTC trim counter direction When a reference sync pulse occurs during the counter is working, the CTC trim

		counter direction is captured to REFDIR bit. 0: Up-counting 1: Down-counting
14:11	Reserved	Must be kept at reset value.
10	TRIMERR	Trim value error bit This bit is set by hardware when the TRIMVALUE in CTC_CTL0 register is overflow or underflow. When the ERRIE in CTC_CTL0 register is set, an interrupt occurs. This bit is cleared by writing 1 to ERRIC bit in CTC_INTC register. 0: No trim value error occurs 1: Trim value error occurs
9	REFMISS	Reference sync pulse miss This bit is set by hardware when the reference sync pulse is missing. This occurs when the CTC trim counter reaches 128 x CKLIM during up-counting and no reference sync pulse is detected. This means the clock is too fast to be trimmed to the correct frequency or other error has occurred. When the ERRIE in CTC_CTL0 register is set, an interrupt occurs. This bit is cleared by writing 1 to ERRIC bit in CTC_INTC register. 0: No Reference sync pulse miss occurs 1: Reference sync pulse miss occurs
8	CKERR	Clock trim error bit This bit is set by hardware when the clock trim error occurs. This occurs when the CTC trim counter is greater than or equal to 128 x CKLIM during down-counting when a reference sync pulse is detected. This means the clock is too slow and cannot be trimmed to the correct frequency. When the ERRIE in CTC_CTL0 register is set, an interrupt occurs. This bit is cleared by writing 1 to ERRIC bit in CTC_INTC register. 0: No Clock trim error occurs 1: Clock trim error occurs
7:4	Reserved	Must be kept at reset value.
3	EREFIF	Expected reference interrupt flag This bit is set by hardware when the CTC counter reaches 0. When the EREFIE in CTC_CTL0 register is set, an interrupt occurs. This bit is cleared by writing 1 to EREFIC bit in CTC_INTC register. 0: No Expected reference occurs 1: Expected reference occurs
2	ERRIF	Error interrupt flag This bit is set by hardware when an error occurs. If any error of TRIMERR, REFMISS or CKERR occurs, this bit will be set. When the ERRIE in CTC_CTL0 register is set, an interrupt occurs. This bit is cleared by writing 1 to ERRIC bit in CTC_INTC register. 0: No Error occurs

		1: An error occurs
1	CKWARNIF	Clock trim warning interrupt flag This bit is set by hardware when a clock trim warning occurs. If the CTC trim counter is greater than or equal to 3 x CKLIM and is smaller than 128 x CKLIM when a reference sync pulse is detected, this bit will be set. This means the clock is too slow or too fast, but can be trimmed to the correct frequency. The TRIMVALUE adds 2 or subtracts 2 when a clock trim warning occurs. When the CKWARNIE in CTC_CTL0 register is set, an interrupt occurs. This bit is cleared by writing 1 to CKWARNIC bit in CTC_INTC register. 0: No Clock trim warning occurs 1: Clock trim warning occurs
0	CKOKIF	Clock trim OK interrupt flag This bit is set by hardware when the clock trim is OK. If the CTC trim counter is smaller than 3 x CKLIM when a reference sync pulse is detected, this bit will be set. This means the clock is OK for using. The TRIMVALUE needs not to be adjusted. When the CKOKIE in CTC_CTL0 register is 1, an interrupt occurs. This bit is cleared by writing 1 to CKOKIC bit in CTC_INTC register. 0: No Clock trim OK occurs 1: Clock trim OK occurs

6.4.4. Interrupt clear register (CTC_INTC)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												EREFIC	ERRIC	CKWARNIC	CKOKIC
												w	w	w	w

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	EREFIC	EREFIF interrupt clear bit This bit is written by software and read as 0. Write 1 to clear EREFIF bit in CTC_STAT register. Writing 0 has no effect.
2	ERRIC	ERRIF interrupt clear bit This bit is written by software and read as 0. Write 1 to clear ERRIF, TRIMERR, REFMISS and CKERR bits in CTC_STAT register. Writing 0 has no effect.

1	CKWARNIC	CKWARNIF interrupt clear bit This bit is written by software and read as 0. Write 1 to clear CKWARNIF bit in CTC_STAT register. Writing 0 has no effect.
0	CKOKIC	CKOKIF interrupt clear bit This bit is written by software and read as 0. Write 1 to clear CKOKIF bit in CTC_STAT register. Writing 0 has no effect.

7. Interrupt / event controller (EXTI)

7.1. Overview

Cortex®-M4 integrates the Nested Vectored Interrupt Controller (NVIC) for efficient exception and interrupts processing. NVIC facilitates low-latency exception and interrupt handling and power management controls. It's tightly coupled to the processor core. You can read the Technical Reference Manual of Cortex®-M4.

EXTI (interrupt / event controller) contains up to 19 independent edge detectors and generates interrupt requests or events to the processor. The EXTI has three trigger types: rising edge, falling edge and both edges. Each edge detector in the EXTI can be independently configured and masked.

7.2. Characteristics

- Cortex®-M4 system exception.
- Up to 66 maskable peripheral interrupts.
- 4 bits interrupt priority configuration—16 priority levels.
- Efficient interrupt processing.
- Support exception pre-emption and tail-chaining.
- Wake up system from power saving mode.
- Up to 19 independent edge detectors in EXTI.
- Three trigger types, rising, falling and both edges.
- Software interrupt or event trigger.
- Trigger sources configurable.

7.3. Function overview

The ARM Cortex®-M4 processor and the Nested Vectored Interrupt Controller (NVIC) prioritize and handle all exceptions in Handler Mode. The processor state is automatically stored to the stack on an exception, and automatically restored from the stack at the end of the Interrupt Service Routine(ISR).

The vector is fetched in parallel to the state saving, enabling efficient interrupt entry. The processor supports tail-chaining, which enables back-to-back interrupts to be performed without the overhead of state saving and restoration. The following tables list all exception types.

Table 7-1. NVIC exception types in Cortex®-M4

Exception type	Vector number	Priority (a)	Vector address	Description
-	0	-	0x0000_0000	Reserved
Reset	1	-3	0x0000_0004	Reset
NMI	2	-2	0x0000_0008	Non maskable interrupt.
HardFault	3	-1	0x0000_000C	All class of fault
MemManage	4	Programmable	0x0000_0010	Memory management
BusFault	5	Programmable	0x0000_0014	Prefetch fault, memory access fault
UsageFault	6	Programmable	0x0000_0018	Undefined instruction or illegal state
-	7-10	-	0x0000_001C - 0x0000_002B	Reserved
SVCall	11	Programmable	0x0000_002C	System service call via SWI instruction
Debug Monitor	12	Programmable	0x0000_0030	Debug monitor
-	13	-	0x0000_0034	Reserved
PendSV	14	Programmable	0x0000_0038	Pendable request for system service
SysTick	15	Programmable	0x0000_003C	System tick timer

Table 7-2. Interrupt vector table

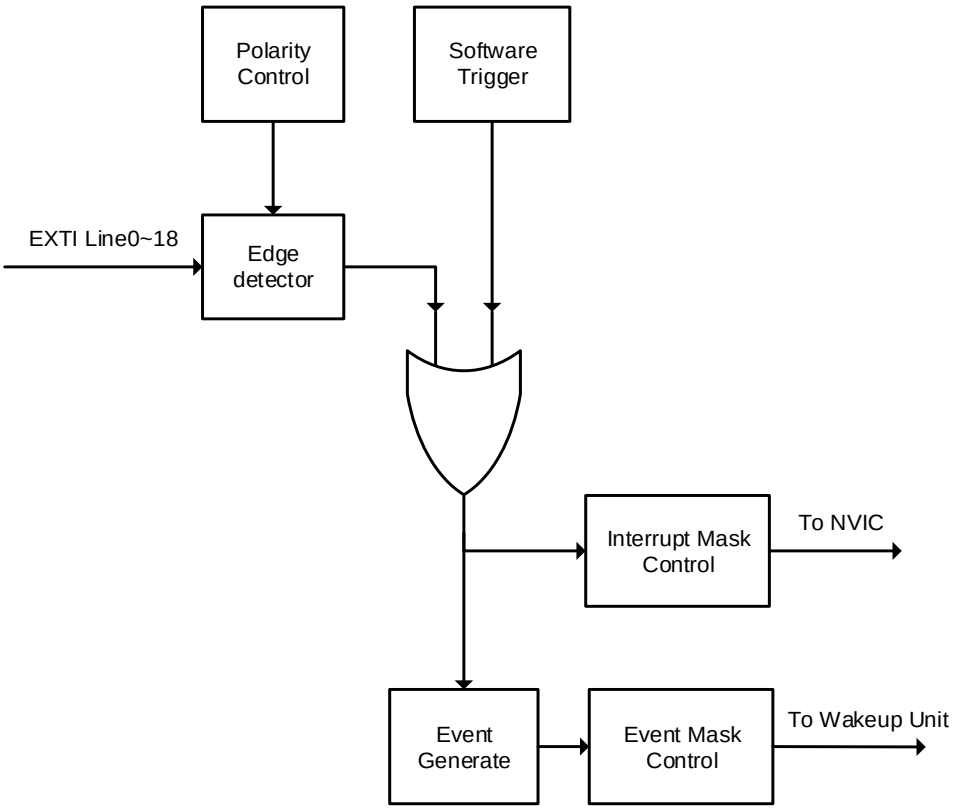
Interrupt number	Vector number	Interrupt description	Vector address
IRQ 0	16	WWDGT interrupt	0x0000_0040
IRQ 1	17	LVD from EXTI interrupt	0x0000_0044
IRQ 2	18	Tamper interrupt	0x0000_0048
IRQ 3	19	RTC global interrupt	0x0000_004C
IRQ 4	20	FMC global interrupt	0x0000_0050
IRQ 5	21	RCU and CTC interrupt	0x0000_0054
IRQ 6	22	EXTI line0 interrupt	0x0000_0058
IRQ 7	23	EXTI line1 interrupt	0x0000_005C
IRQ 8	24	EXTI line2 interrupt	0x0000_0060
IRQ 9	25	EXTI line3 interrupt	0x0000_0064
IRQ 10	26	EXTI line4 interrupt	0x0000_0068
IRQ 11	27	DMA0 channel0 global interrupt	0x0000_006C
IRQ 12	28	DMA0 channel1 global interrupt	0x0000_0070
IRQ 13	29	DMA0 channel2 global interrupt	0x0000_0074
IRQ 14	30	DMA0 channel3 global interrupt	0x0000_0078

Interrupt number	Vector number	Interrupt description	Vector address
IRQ 15	31	DMA0 channel4 global interrupt	0x0000_007C
IRQ 16	32	DMA0 channel5 global interrupt	0x0000_0080
IRQ 17	33	DMA0 channel6 global interrupt	0x0000_0084
IRQ 18	34	ADC0 and ADC1 global interrupt	0x0000_0088
IRQ 19	35	CAN0 TX interrupt	0x0000_008C
IRQ 20	36	CAN0 RX0 interrupt	0x0000_0090
IRQ 21	37	CAN0 RX1 interrupt	0x0000_0094
IRQ 22	38	CAN0 EWMC interrupt	0x0000_0098
IRQ 23	39	EXTI line[9:5] interrupts	0x0000_009C
IRQ 24	40	TIMER0 break and TIMER8 global interrupts	0x0000_00A0
IRQ 25	41	TIMER0 update and TIMER9 global interrupts	0x0000_00A4
IRQ 26	42	TIMER0 trigger and Channel commutation and TIMER10 global interrupts	0x0000_00A8
IRQ 27	43	TIMER0 channel capture compare interrupt	0x0000_00AC
IRQ 28	44	TIMER1 global interrupt	0x0000_00B0
IRQ 29	45	TIMER2 global interrupt	0x0000_00B4
IRQ 30	46	TIMER3 global interrupt	0x0000_00B8
IRQ 31	47	I2C0 event interrupt	0x0000_00BC
IRQ 32	48	I2C0 error interrupt	0x0000_00C0
IRQ 33	49	I2C1 event interrupt	0x0000_00C4
IRQ 34	50	I2C1 error interrupt	0x0000_00C8
IRQ 35	51	SPI0 global interrupt	0x0000_00CC
IRQ 36	52	SPI1 global interrupt	0x0000_00D0
IRQ 37	53	USART0 global interrupt	0x0000_00D4
IRQ 38	54	USART1 global interrupt	0x0000_00D8
IRQ 39	55	USART2 global interrupt	0x0000_00DC
IRQ 40	56	EXTI line[15:10] interrupts	0x0000_00E0
IRQ 41	57	RTC alarm from EXTI interrupt	0x0000_00E4
IRQ 42	58	USBFS wakeup from EXTI interrupt	0x0000_00E8
IRQ 43	59	TIMER7 break and TIMER11 global interrupts	0x0000_00EC

Interrupt number	Vector number	Interrupt description	Vector address
IRQ 44	60	TIMER7 update and TIMER12 global interrupts	0x0000_00F0
IRQ 45	61	TIMER7 trigger and Channel commutation and TIMER13 global interrupts	0x0000_00F4
IRQ 46	62	TIMER7 channel capture compare interrupt	0x0000_00F8
IRQ 47	63	Reserved	0x0000_00FC
IRQ 48	64	EXMC global interrupt	0x0000_0100
IRQ 49	65	Reserved	0x0000_0104
IRQ50	66	TIMER4 global interrupt	0x0000_0108
IRQ51	67	SPI2 global interrupt	0x0000_010C
IRQ52	68	UART3 global interrupt	0x0000_0110
IRQ53	69	UART4 global interrupt	0x0000_0114
IRQ54	70	TIMER5 global interrupt	0x0000_0118
IRQ55	71	TIMER6 global interrupt	0x0000_011C
IRQ56	72	DMA1 channel0 global interrupt	0x0000_0120
IRQ57	73	DMA1 channel1 global interrupt	0x0000_0124
IRQ58	74	DMA1 channel2 global interrupt	0x0000_0128
IRQ59	75	DMA1 channel3 global interrupt	0x0000_012C
IRQ60	76	DMA1 channel4 global interrupt	0x0000_0130
IRQ61	77	Reserved	0x0000_0134
IRQ62	78	Reserved	0x0000_0138
IRQ63	79	CAN1 TX interrupt	0x0000_013C
IRQ64	80	CAN1 RX0 interrupt	0x0000_0140
IRQ65	81	CAN1 RX1 interrupt	0x0000_0144
IRQ66	82	CAN1 EWMC interrupt	0x0000_0148
IRQ67	83	USBFS global interrupt	0x0000_014C

7.4. External interrupt and event block diagram

Figure 7-1. Block diagram of EXTI



7.5. External Interrupt and Event function overview

Th The EXTI contains up to 19 independent edge detectors and generates interrupts request or event to the processor. The EXTI has three trigger types: rising edge, falling edge and both edges. Each edge detector in the EXTI can be configured and masked independently.

The EXTI trigger source includes 16 external lines from GPIO pins and 3 lines from internal modules which refers to ***Table 7-3. EXTI source*** for detail. All GPIO pins can be selected as an EXTI trigger source by configuring AFIO_EXTISSx registers in GPIO module (please refer to ***General-purpose and alternate-function I/Os (GPIO and AFIO)*** section for detail).

EXTI provides not only interrupts but also events signals to the processor. The Cortex®-M4 processor fully supports the Wait For Interrupt (WFI), Wait For Event (WFE) and the Send Event (SEV) instructions. The Wake-up Interrupt Controller (WIC) enables the processor and NVIC to be put into a very low-power sleep mode leaving the WIC to identify and prioritize interrupts and events. EXTI can be used to wake up processor and the whole system when some expected event occurs, such as a special GPIO pin toggling or RTC alarm.

Hardware trigger

Hardware trigger may be used to detect the voltage change of external or internal signals.

The software should follow these steps to use this function:

1. Configure EXTI sources in AFIO module based on application requirement.
2. Configure EXTI_RTEN and EXTI_FTEN to enable the rising or falling detection on related pins. (Software may set both RTENx and FTENx for a pin at the same time to detect both rising and falling changes on this pin).
3. Enable interrupts or events by setting related EXTI_INTEN or EXTI_EVEN bits.
4. EXTI starts to detect changes on the configured pins. The related interrupt or event will be triggered when desired change is detected on these pins. If the interrupt is triggered, the related PDx is set; if the event is triggered, the related PDx is not set. The software should response to the interrupts or events and clear these PDx bits.

Software trigger

Software may also trigger EXTI interrupts or events following these steps:

1. Enable interrupts or events by setting related EXTI_INTEN or EXTI_EVEN bits.
2. Set SWIEVx bits in EXTI_SWIEV register, the related interrupt or event will be triggered immediately. If the interrupt is triggered, the related PDx is set; if the event is triggered, the related PDx is not set.

Table 7-3. EXTI source

EXTI line number	Source
0	PA0 / PB0 / PC0 / PD0 / PE0
1	PA1 / PB1 / PC1 / PD1 / PE1
2	PA2 / PB2 / PC2 / PD2 / PE2
3	PA3 / PB3 / PC3 / PD3 / PE3
4	PA4 / PB4 / PC4 / PD4 / PE4
5	PA5 / PB5 / PC5 / PD5 / PE5
6	PA6 / PB6 / PC6 / PD6 / PE6
7	PA7 / PB7 / PC7 / PD7 / PE7
8	PA8 / PB8 / PC8 / PD8 / PE8
9	PA9 / PB9 / PC9 / PD9 / PE9
10	PA10 / PB10 / PC10 / PD10 / PE10
11	PA11 / PB11 / PC11 / PD11 / PE1
12	PA12 / PB12 / PC12 / PD12 / PE12
13	PA13 / PB13 / PC13 / PD13 / PE13
14	PA14 / PB14 / PC14 / PD14 / PE14
15	PA15 / PB15 / PC15 / PD15 / PE15
16	LVD
17	RTC alarm
18	USBFS wakeup

7.6. Register definition

EXTI base address: 0x4001 0400

7.6.1. Interrupt enable register (EXTI_INTEN)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													INTEN18	INTEN17	INTEN16
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTEN15	INTEN14	INTEN13	INTEN12	INTEN11	INTEN10	INTEN9	INTEN8	INTEN7	INTEN6	INTEN5	INTEN4	INTEN3	INTEN2	INTEN1	INTEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18:0	INTENx	Interrupt enable bit x (x = 0…18) 0: Interrupt from linex is disabled 1: Interrupt from linex is enabled

7.6.2. Event enable register (EXTI_EVEN)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													EVEN18	EVEN17	EVEN16
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVEN15	EVEN14	EVEN13	EVEN12	EVEN11	EVEN10	EVEN9	EVEN8	EVEN7	EVEN6	EVEN5	EVEN4	EVEN3	EVEN2	EVEN1	EVEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18:0	EVENx	Event enable bit x (x = 0…18) 0: Event from linex is disabled 1: Event from linex is enabled

7.6.3. Rising edge trigger enable register (EXTI_RTEN)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													RTEN18	RTEN17	RTEN16
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTEN15	RTEN14	RTEN13	RTEN12	RTEN11	RTEN10	RTEN9	RTEN8	RTEN7	RTEN6	RTEN5	RTEN4	RTEN3	RTEN2	RTEN1	RTEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18:0	RTENx	Rising edge trigger enable bit x (x = 0…18) 0: Rising edge of linex is invalid 1: Rising edge of linex is valid as an interrupt / event request

7.6.4. Falling edge trigger enable register (EXTI_FTEN)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FTEN18	FTEN17	FTEN16
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FTEN15	FTEN14	FTEN13	FTEN12	FTEN11	FTEN10	FTEN9	FTEN8	FTEN7	FTEN6	FTEN5	FTEN4	FTEN3	FTEN2	FTEN1	FTEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18:0	FTENx	Falling edge trigger enable bit x (x = 0…18) 0: Falling edge of linex is invalid 1: Falling edge of linex is valid as an interrupt / event request

7.6.5. Software interrupt event register (EXTI_SWIEV)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													SWIEV18	SWIEV17	SWIEV16
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWIEV15	SWIEV14	SWIEV13	SWIEV12	SWIEV11	SWIEV10	SWIEV9	SWIEV8	SWIEV7	SWIEV6	SWIEV5	SWIEV4	SWIEV3	SWIEV2	SWIEV1	SWIEV0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18:0	SWIEVx	Interrupt / event software trigger x (x = 0…18) 0: Deactivate the EXTIx software interrupt / event request 1: Activate the EXTIx software interrupt / event request

7.6.6. Pending register (EXTI_PD)

Address offset: 0x14

Reset value: undefined

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													PD18	PD17	PD16
													rc_w1	rc_w1	rc_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18:0	PDx	Interrupt pending status x (x = 0…18) 0: EXTI linex is not triggered 1: EXTI linex is triggered This bit is cleared to 0 by writing 1 to it.

8. General-purpose and alternate-function I/Os (GPIO and AFIO)

8.1. Overview

There are up to 80 general purpose I/O pins (GPIO), named PA0 ~ PA15, PB0 ~ PB15, PC0 ~ PC15, PD0 ~ PD15 and PE0 ~ PE15 for the device to implement logic input/output functions. Each GPIO port has related control and configuration registers to satisfy the requirements of specific applications. The external interrupts on the GPIO pins of the device have related control and configuration registers in the Interrupt/Event Controller Unit (EXTI).

The GPIO ports are pin-shared with other alternative functions (AFs) to obtain maximum flexibility on the package pins. The GPIO pins can be used as alternative functional pins by configuring the corresponding registers such as the AF input or output pins.

Each of the GPIO pins can be configured by software as output (push-pull or open-drain), input, peripheral alternate function or analog mode. Each GPIO pin can be configured as pull-up, pull-down or floating. All GPIOs are high-current capable except for analog mode.

8.2. Characteristics

- Input/output direction control.
- Schmitt trigger input function enable control.
- Each pin weak pull-up/pull-down function.
- Output push-pull/open-drain enable control.
- Output set/reset control.
- External interrupt with programmable trigger edge – using EXTI configuration registers.
- Analog input/output configuration.
- Alternate function input/output configuration.
- Port configuration lock.

8.3. Function overview

Each of the general-purpose I/O ports can be configured as 8 modes, including analog inputs, input floating, input pull-down/pull-up, GPIO push-pull/open-drain and AFIO push-pull/open-drain mode by two GPIO configuration registers (GPIOx_CTL0/GPIOx_CTL1), and a 32-bits registers (GPIOx_OCTL). ***Table 8-1. GPIO configuration table*** shows the details.

Table 8-1. GPIO configuration table

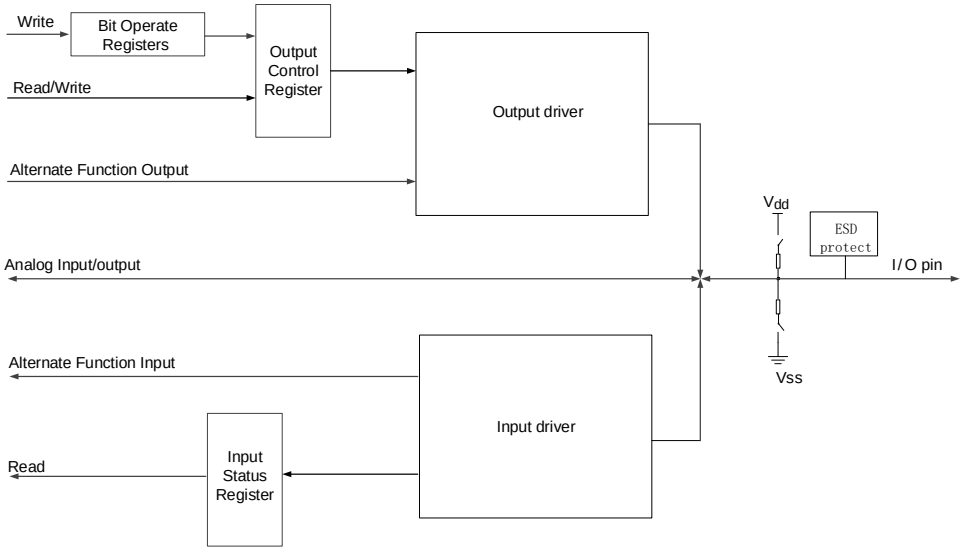
Configuration mode		CTL[1:0]	MD[1:0]	OCTL
Input	Analog	00	00	don't care

	Input floating	01		don't care
	Input pull-down	10		0
	Input pull-up	10		1
General purpose Output (GPIO)	Push-pull	00	00: Reserved 01: Speed up to 10MHz 10: Speed up to 2MHz 11: Speed up to 50MHz 11: Speed up to 120MHz(1) (SPDy required to be set to 0b11)	0 or 1
	Open-drain	01		0 or 1
Alternate Function Output (AFIO)	Push-pull	10		don't care
	Open-drain	11		don't care

When the port output speed is more than 50 MHz, the user should enable the I/O compensation cell. Refer to IO compensation control register (AFIO_CPSCTL).

Figure 8-1. Basic structure of of a general-pupose I/O shows the basic structure of an I/O Port bit.

Figure 8-1. Basic structure of of a general-pupose I/O



8.3.1. GPIO pin configuration

During or just after the reset period, the alternative functions are all inactive and the GPIO ports are configured as the input floating mode without pull-up (PU)/pull-down (PD) resistors. But the JTAG/Serial-Wired Debug pins are configured as input PU/PD mode after the reset.

PA15: JTDI in PU mode

PA14: JTCK / SWCLK in PD mode

PA13: JTMS / SWDIO in PU mode

PB4: NJTRST in PU mode

PB3: JTDO in floating mode

The GPIO pins can be configured as inputs or outputs. When the GPIO pins are configured

as input pins, all GPIO pins have an internal weak pull-up and weak pull-down which can be chosen. And the data on the external pins can be captured at every APB2 clock cycle to the port input status register (GPIOx_ISTAT).

When the GPIO pins are configured as output pins, the user can configure the speed of the ports and choose the output driver mode, push-pull or open-drain mode. The value of the port output control register (GPIOx_OCTL) is output on the I/O pin.

There is no need to read-then-write when programming the GPIOx_OCTL at the bit level, the user can modify only one bit or several bits in a single atomic APB2 write access by programming '1' to the bit operate register (GPIOx_BOP, or for GPIOx_BC). The other bits will not be affected.

8.3.2. External interrupt/event lines

The port can use external interrupt/event lines only if it is configured in input mode.

8.3.3. Alternate functions (AF)

When the port is configured as AFIO (set CTLy bits to "0b10" or "0b11", and set MDy bits to "0b01", "0b10", or "0b11", which is in GPIOx_CTL0/GPIOx_CTL1 registers), the port is used as peripheral alternate functions. The detail alternate function assignments for each port are described in the device datasheet.

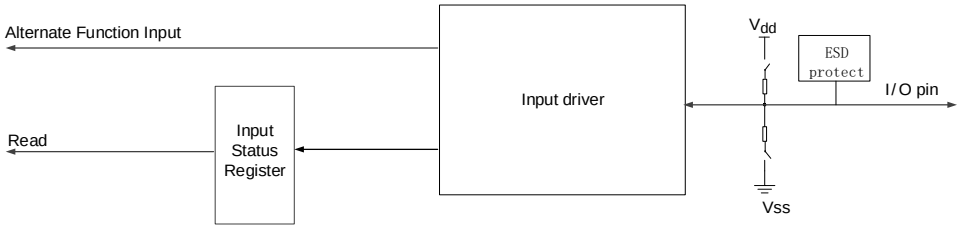
8.3.4. Input configuration

When GPIO pin is configured as input.

- The schmitt trigger input is enabled.
- The weak pull-up and pull-down resistors could be chosen.
- Every APB2 clock cycle the data present on the I/O pin is got to the port input status register.
- Disable the output buffer.

Figure 8-2. Basic structure of Input configuration shows the input configuration of the GPIO pin.

Figure 8-2. Basic structure of Input configuration



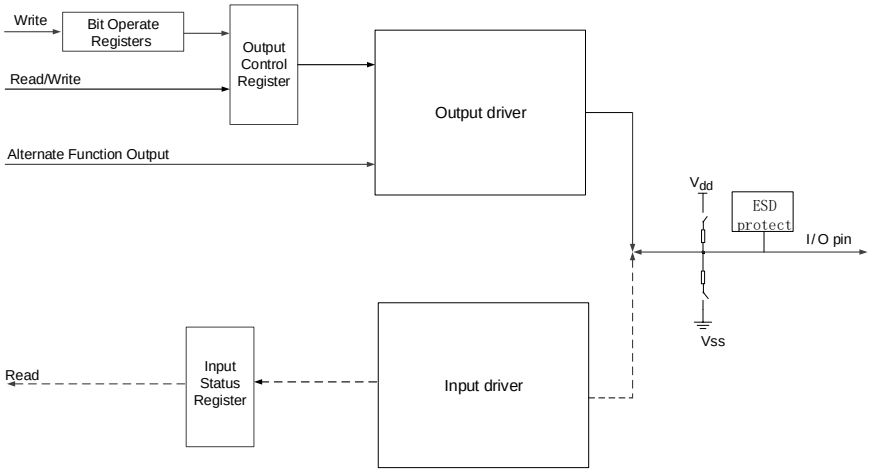
8.3.5. Output configuration

When GPIO pin is configured as output.

- The schmitt trigger input is enabled.
- The weak pull-up and pull-down resistors are disabled.
- The output buffer is enabled.
- Open-drain mode, the pad outputs low level when setting “0” in the output control register;while the pad holds Hi-Z state when set “1” in the output control register.
- Push-pull mode, the pad outputs low level when setting “0” in the output control register;while the pad outputs high level when setting “1” in the output control register.
- A read access to the port output control register gets the last written value.
- A read access to the port input status register gets the I/O state.

Figure 8-3. Basic structure of Output configuration shows the output configuration of the GPIO pin.

Figure 8-3. Basic structure of Output configuration



8.3.6. Analog configuration

When GPIO pin is used as analog configuration.

- The weak pull-up and pull-down resistors are disabled.
- The output buffer is disabled.
- The schmitt trigger input is disabled.
- The port input status register of this I/O port bit is “0”.

Figure 8-4. Basic structure of Analog configuration shows the analog configuration of the GPIO pin.

Figure 8-4. Basic structure of Analog configuration



8.3.7. Alternate function (AF) configuration

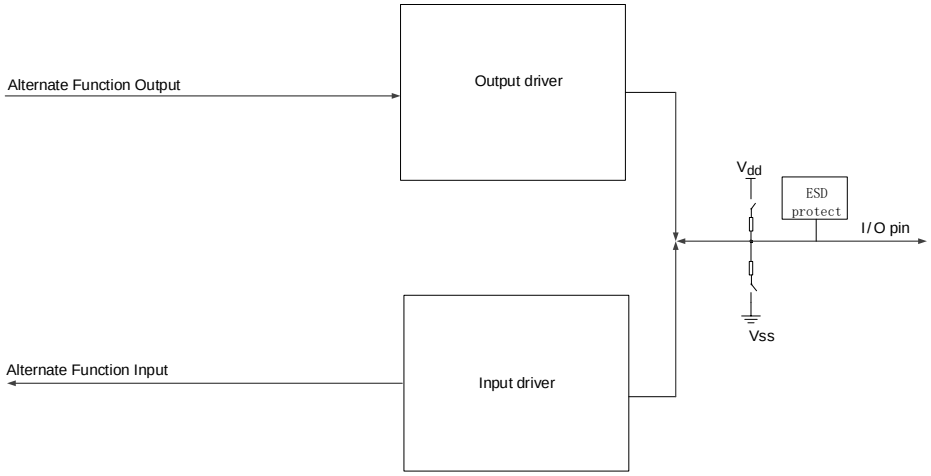
To suit for different device packages, the GPIO supports some alternate functions mapped to some other pins by software.

When be configured as alternate function.

- The output buffer is enabled in open-drain or push-pull configuration.
- The output buffer is driven by the peripheral.
- The schmitt trigger input is enabled.
- The weak pull-up and pull-down resistors could be chosen when input.
- The I/O pin data is stored into the port input status register every APB2 clock.
- A read access to the port input status register gets the I/O state.
- A read access to the port output control register gets the last written value.

Figure 8-5. Basic structure of Alternate function configuration shows the alternate function configuration of the GPIO pin.

Figure 8-5. Basic structure of Alternate function configuration



8.3.8. GPIO locking function

The locking mechanism allows the IO configuration to be protected.

The protected registers are GPIOx_CTL0, GPIOx_CTL1. It allows the I/O configuration to be frozen by the 32-bit locking register (GPIOx_LOCK). When the special LOCK sequence has occurred on LKK bit in GPIOx_LOCK register and the LKy bit is set in GPIOx_LOCK register, the corresponding port is locked and the corresponding port configuration cannot be modified

until the next reset. It is recommended to be used in the configuration of driving a power module.

8.3.9. GPIO I/O compensation cell

If the I/O port output speed need more than 50MHz, it is recommended to use the compensation cell for slew rate control to reduce the I/O noise effects on the power supply.

Compensation cell is disabled after reset, it needs to be enabled by the user. After enabling the compensation cell, the complete flag CPS_RDY is set to indicate that the compensation cell is ready and can be used. If the supply voltage over 2.4 V~3.6V, must disable the compensation cell.

8.4. Remapping function I/O and debug configuration

8.4.1. Overview

In order to expand the flexibility of the GPIO or the usage of peripheral functions, each I/O pin can be configured up to four different functions by setting the AFIO port configuration register (AFIO_PCF0/AFIO_PCF1). Suitable pinout locations can be selected using the peripheral IO remapping function. Additionally, various GPIO pins can be selected as the EXTI interrupt line source by setting the relevant EXTI source selection register (AFIO_EXTISSx) to trigger an interrupt or event.

8.4.2. Characteristics

- EXTI source selection
- Each pin has up to four alternative functions for configuration

8.4.3. JTAG/SWD alternate function remapping

The debug interface signals are mapped on the GPIO ports as shown in the table below.

Table 8-2. Debug interface signals

Pin Name	Function description
PA13	JTMS / SWDIO
PA14	JTCK / SWCLK
PA15	JTDI
PB3	JTDO / TRACESWO
PB4	NJTRST

To reduce the number of GPIOs used for debugging, the user can configure SWJ_CFG[2:0] bits in the AFIO_PCF0 to a different value. Refer to the table below.

Table 8-3. Debug port mapping and Pin availability

SWJ_CFG [2:0]	JTAG-DP and SW-DP	Pin availability				
		PA13	PA14	PA15	PB3	PB4
000	JTAG-DP Enabled and SW-DP Enabled (Reset state)	X	X	X	X	X
001	JTAG-DP Enabled and SW-DP Enabled but without NJTRST	X	X	X	X	√
010	JTAG-DP Disabled and SW-DP Enabled	X	X	√	√	√
100	JTAG-DP Disabled and SW-DP Disabled	√	√	√	√	√
Other	Forbidden					

1. Can't released if using asynchronous trace.
2. " √ " Indicates that the corresponding pin can be used as a general-purpose I/O.
3. "X" Indicates that the corresponding pin can't be used as a general-purpose I/O.
4. The SWJ(Serial Wire JTAG) supports JTAG or SWD access to the Cortex debug port. The default state after reset is SWJ ON without trace.This allows JTAG or SW mode to be enabled by sending a specific sequence on the JTMS/JTCK pin.

8.4.4. ADC AF remapping

Refer to AFIO Port Configuration Register 0 (AFIO_PCF0).

Table 8-4. ADC0/1 external trigger routine conversion AF remapping function (1)

Register	ADC0	ADC1
ADC0_ETRGRT_REMAP = 0	ADC0 externa signal trigger routine conversion is connected to EXTI11	-
ADC0_ETRGRT_REMAP = 1	ADC0 external signal trigger routine conversion is connected to TIMER7_TRGO	-
ADC1_ETRGRT_REMAP = 0	-	ADC1 external signal trigger routine conversion is connected to EXTI11
ADC1_ETRGRT_REMAP = 1	-	ADC1 external signal trigger routine conversion is connected to TIMER7_TRGO

1. Remap available only for High-density and Extra-density devices

8.4.5. TIMER AF remapping

Table 8-5. TIMERx alternate function remapping

Alternate function	TIMERx_REMAP [1:0](x = 0，1，2)			
	TIMERx_REMAP(x = 3，8)		-	
	"0" /"00" (no remap)	"1" /"01" (partial remap)	"10" (partial remap)	"11" (full remap)
TIMER0_ETI	PA12		-	PE7
TIMER0_CH0	PA8		-	PE9
TIMER0_CH1	PA9		-	PE11
TIMER0_CH2	PA10		-	PE13
TIMER0_CH3	PA11		-	PE14
TIMER0_BRKIN	PB12(2)	PA6	-	PE15
TIMER0_CH0_ON	PB13(2)	PA7	-	PE8
TIMER0_CH1_ON	PB14(2)	PB0	-	PE10
TIMER0_CH2_ON	PB15(2)	PB1	-	PE12
TIMER1_CH0/TIMER1_ETI(3)	PA0	PA15	PA0	PA15
TIMER1_CH1	PA1	PB3	PA1	PB3
TIMER1_CH2	PA2		PB10	
TIMER1_CH3	PA3		PB11	
TIMER2_CH0	PA6	-	PB4	PC6
TIMER2_CH1	PA7	-	PB5	PC7
TIMER2_CH2	PB0	-	PB0	PC8
TIMER2_CH3	PB1	-	PB1	PC9
TIMER3_CH0	PB6	PD12	-	-
TIMER3_CH1	PB7	PD13	-	-
TIMER3_CH2	PB8	PD14	-	-
TIMER3_CH3	PB9	PD15	-	-
TIMER8_CH0	PA2	PE5	-	-
TIMER8_CH1	PA3	PE6	-	-

1. TIMER0 remap(full remap) available only for 100-pin packages.
2. TIMER0 remap not available on 36-pin package.
3. TIMER1_CH0 and TIMER1_ETI share the same pin but cannot be used at the same time.

4. TIMER1 remap(full remap) not available on 36-pin package.
5. TIMER2 remap(full remap) available only for 64-pin, 100-pin packages.
6. TIMER3 remap(full remap) available only for 100-pin packages.
7. TIMER8 refer to the AF remap and debug I/O configuration register 1(AFIO_PCF1).

Table 8-6. TIMER4 alternate function remapping (1)

Alternate function	TIMER4CH3_IREMAP = 0	TIMER4CH3_IREMAP = 1
TIMER4_CH3	TIMER4_CH3 is connected to PA3	IRC40K internal clock is connected to TIMER4_CH3 input for calibration purpose

1. Remap available only for High-density and Extra-density and Connectivity lines devices.

8.4.6. USART AF remapping

Refer to AFIO port configuration register 0 (AFIO_PCF0).

Table 8-7. USART0/1/2 alternate function remapping

Register	USART0	USART1	USART2
USART0_REMAP = 0	PA9(USART0_TX) PA10(USART0_RX)		-
USART0_REMAP = 1	PB6(USART0_TX) PB7(USART0_RX)		-
USART1_REMAP = 0	-	PA0(USART1_CTS) PA1(USART1_RTS) PA2(USART1_TX) PA3(USART1_RX) PA4(USART1_CK)	-
USART1_REMAP = 1 (1)	-	PD3(USART1_CTS) PD4(USART1_RTS) PD5(USART1_TX) PD6(USART1_RX) PD7(USART1_CK)	-
USART2_REMAP[1:0] = “00” (no remap)	-	-	PB10(USART2_TX) PB11(USART2_RX) PB12(USART2_CK) PB13(USART2_CTS) PB14(USART2_RTS)
USART2_REMAP [1:0] =“01” (partial remap) (2)	-	-	PC10(USART2_TX) PC11(USART2_RX) PC12(USART2_CK) PB13(USART2_CTS) PB14(USART2_RTS)
USART2_REMAP [1:0] =“11” (full remap)	-	-	PD8(USART2_TX) PD9(USART2_RX)

(3)			PD10(USART2_CK) PD11(USART2_CTS) PD12(USART2_RTS)

1. Remap available only 100-pin packages.
2. Remap available only for 64-pin,100-pin packages.
3. Remap available only 100-pin packages.

8.4.7. I2C0 AF remapping

Refer to AFIO port configuration register 0 (AFIO_PCF0).

Table 8-8. I2C0 alternate function remapping

Register	I2C0_SCL	I2C0_SDA
I2C0_REMAP = 0	PB6	PB7
I2C0_REMAP = 1	PB8	PB9

8.4.8. SPI0/SPI2/I2S AF remapping

Refer to AFIO port configuration register 0 (AFIO_PCF0).

Table 8-9. SPI0/SPI2/I2S alternate function remapping

Register	SPI0	SPI2/I2S
SPI0_REMAP = 0	PA4(SPI0_NSS) PA5(SPI0_SCK) PA6(SPI0_MISO) PA7(SPI0_MOSI) PA2(SPI0_IO2) PA3(SPI0_IO3)	-
SPI0_REMAP = 1	PA15(SPI0_NSS) PB3(SPI0_SCK) PB4(SPI0_MISO) PB5(SPI0_MOSI) PB6(SPI0_IO2) PB7(SPI0_IO3)	-
SPI2_REMAP = 0	-	PA15(SPI2_NSS/ I2S2_WS) PB3(SPI2_SCK/ I2S2_CK) PB4(SPI2_MISO) PB5(SPI2_MOSI/I2S2_SD)
SPI2_REMAP = 1	-	PA4(SPI2_NSS/ I2S2_WS) PC10(SPI2_SCK/ I2S2_CK) PC11(SPI2_MISO) PC12(SPI2_MOSI/I2S2_SD)

8.4.9. CAN0/1 AF remapping

The CAN0 signals can be mapped on Port A, Port B or Port D as shown in table below. For port D, remapping is not possible in devices delivered in 64-pin packages.

Table 8-10. CAN0/1 alternate function remapping

Register (1)	CAN0	CAN1
CAN0_REMAP[1:0] ="00"	PA11(CAN0_RX) PA12(CAN0_TX)	-
CAN0_REMAPI[1:0] ="10"	PB8(CAN0_RX) PB9(CAN0_TX)	-
CAN0_REMAP[1:0] ="11"(2)	PD0(CAN0_RX) PD1(CAN0_TX)	-
CAN1_REMAP = "0"	-	PB12(CAN1_RX) PB13(CAN1_TX)
CAN1_REMAP = "1"	-	PB5(CAN1_RX) PB6(CAN1_TX)

1. CAN0_RX and CAN0_TX in connectivity line devices; CAN_RX and CAN_TX in other devices with a single CAN interface.
2. Remap not available on 36-pin package.

8.4.10. CTC AF remapping

Refer to AFIO port configuration register 1 (AFIO_ PCF1).

Table 8-11. CTC alternate function remapping

Alternate function	CTC_REMAP [1:0] = "00"	CTC_REMAP [1:0] = "01"
CTC_SYNC	PA8	PD15

8.4.11. CLK pins AF remapping

The LXTAL oscillator pins OSC32_IN and OSC32_OUT can be used as general-purpose I/O PC14 and PC15 individually, when the LXTAL oscillator is off. The LXTAL has priority over the GPIOs function.

Note: 1. But when the 1.8V domain is powered off (by entering standby mode) or when the backup domain is supplied by V_{BAT} (V_{DD} no more supplied), the PC14/PC15 GPIO functionality is lost and will be set in analog mode.

2. Refer to the note on IO usage restrictions in Section 3.3.1.

Table 8-12. OSC32 pins configuration

Alternate function	LXTAL= ON	LXTAL= OFF
PC14	OSC32_IN	PC14
PC15	OSC32_OUT	PC15

The HXTAL oscillator pins OSC_IN/OSC_OUT can be used as general-purpose I/O PD0/PD1. PD0/PD1 cannot be used for external interrupt/event generation on 36--pin, 48--pin and 64-

pin packages.

Table 8-13. OSC pins configuration

Alternate function	HXTAL= ON	HXTAL = OFF
PD0	OSC_IN	PD0
PD1	OSC_OUT	PD1

8.5. Register definition

GPIOA base address: 0x4001 0800
GPIOB base address: 0x4001 0C00
GPIOC base address: 0x4001 1000
GPIOD base address: 0x4001 1400
GPIOE base address: 0x4001 1800
AFIO base address: 0x4001 0000

8.5.1. Port control register 0 (GPIOx_CTL0, x=A..E)

Address offset: 0x00
Reset value: 0x4444 4444(x= A, C..E) / 0x4448 4444(x=B)

This register has to be accessed by word (32-bit).

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16
CTL7[1:0]	MD7[1:0]	CTL6[1:0]	MD6[1:0]	CTL5[1:0]	MD5[1:0]	CTL4[1:0]	MD4[1:0]
rw	rw	rw	rw	rw	rw	rw	rw

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
CTL3[1:0]	MD3[1:0]	CTL2[1:0]	MD2[1:0]	CTL1[1:0]	MD1[1:0]	CTL0[1:0]	MD0[1:0]
rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	CTL7[1:0]	Port 7 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
29:28	MD7[1:0]	Port 7 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
27:26	CTL6[1:0]	Port 6 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
25:24	MD6[1:0]	Port 6 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
23:22	CTL5[1:0]	Port 5 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
21:20	MD5[1:0]	Port 5 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.

19:18	CTL4[1:0]	Port 4 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
17:16	MD4[1:0]	Port 4 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
15:14	CTL3[1:0]	Port 3 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
13:12	MD3[1:0]	Port 3 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
11:10	CTL2[1:0]	Port 2 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
9:8	MD2[1:0]	Port 2 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
7:6	CTL1[1:0]	Port 1 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
5:4	MD1[1:0]	Port 1 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
3:2	CTL0[1:0]	Pin 0 configuration bits These bits are set and cleared by software . Input mode (MD[1:0] =00) 00: Analog mode 01: Floating input 10: Input with pull-up / pull-down 11: Reserved Output mode (MD[1:0] >00) 00: GPIO output with push-pull 01: GPIO output with open-drain 10: AFIO output with push-pull 11: AFIO output with open-drain
1:0	MD0[1:0]	Port 0 mode bits These bits are set and cleared by software 00: Input mode (reset state) 01: Output mode(10MHz)

10: Output mode(2MHz)

11: Output mode(50MHz)

8.5.2. Port control register 1 (GPIOx_CTL1, x=A..E)

Address offset: 0x04

Reset value: 0x4444 4444(x= B..E) / 0x8884 4444(x=A)

This register has to be accessed by word (32-bit).

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16
CTL15[1:0]	MD15[1:0]	CTL14[1:0]	MD14[1:0]	CTL13[1:0]	MD13[1:0]	CTL12[1:0]	MD12[1:0]
rw	rw	rw	rw	rw	rw	rw	rw

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
CTL11[1:0]	MD11[1:0]	CTL10[1:0]	MD10[1:0]	CTL9[1:0]	MD9[1:0]	CTL8[1:0]	MD8[1:0]
rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	CTL15[1:0]	Port 15 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
29:28	MD15[1:0]	Port 15 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
27:26	CTL14[1:0]	Port 14 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
25:24	MD14[1:0]	Port 14 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
23:22	CTL13[1:0]	Port 13 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
21:20	MD13[1:0]	Port 13 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
19:18	CTL12[1:0]	Port 12 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
17:16	MD12[1:0]	Port 12 mode bits These bits are set and cleared by software.

		Refer to MD0[1:0] description.
15:14	CTL11[1:0]	Port 11 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
13:12	MD11[1:0]	Port 11 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
11:10	CTL10[1:0]	Port 10 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
9:8	MD10[1:0]	Port 10 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
7:6	CTL9[1:0]	Port 9 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
5:4	MD9[1:0]	Port 9 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.
3:2	CTL8[1:0]	Port 8 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description.
1:0	MD8[1:0]	Port 8 mode bits These bits are set and cleared by software. Refer to MD0[1:0] description.

8.5.3. Port input status register (GPIOx_ISTAT, x=A..E)

Address offset: 0x08

Reset value: 0x0000 XXXX

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISTAT15	ISTAT14	ISTAT13	ISTAT12	ISTAT11	ISTAT10	ISTAT 9	ISTAT 8	ISTAT 7	ISTAT 6	ISTAT 5	ISTAT 4	ISTAT 3	ISTAT 2	ISTAT 1	ISTAT 0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions

31:16	Reserved	Must be kept at reset value.
15:0	ISTATy	Port input status(y=0..15) These bits are set and cleared by hardware. 0: Input signal low 1: Input signal high

8.5.4. Port output control register (GPIOx_OCTL, x=A..E)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OCTL15	OCTL14	OCTL13	OCTL12	OCTL11	OCTL10	OCTL9	OCTL8	OCTL7	OCTL6	OCTL5	OCTL4	OCTL3	OCTL2	OCTL1	OCTL0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	OCTLy	Port output control(y=0..15) These bits are set and cleared by software. 0: Pin outputs low 1: Pin outputs high

8.5.5. Port bit operate register (GPIOx_BOP, x=A..E)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8	CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOP15	BOP14	BOP13	BOP12	BOP11	BOP10	BOP9	BOP8	BOP7	BOP6	BOP5	BOP4	BOP3	BOP2	BOP1	BOP0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:16	CR**y**	Port clear bit y(y=0..15) These bits are set and cleared by software.

		0: No action on the corresponding OCTLy bit 1: Clear the corresponding OCTLy bit to 0
15:0	BOP**y**	Port set bit y(y=0..15) These bits are set and cleared by software. 0: No action on the corresponding OCTLy bit 1: Set the corresponding OCTLy bit to 1

8.5.6. Port bit clear register (GPIOx_BC, x=A..E)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8	CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CRy	Port Clear bit y(y=0..15) These bits are set and cleared by software. 0: No action on the corresponding OCTLy bit 1: Clear the corresponding OCTLy bit to 0

8.5.7. Port configuration lock register (GPIOx_LOCK, x=A..E)

Address offset: 0x18

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LK15	LK14	LK13	LK12	LK11	LK10	LK9	LK8	LK7	LK6	LK5	LK4	LK3	LK2	LK1	LK0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.

16	LKK	Lock sequence key It can only be set by using the lock key writing sequence. And it is always readable. 0: GPIO_LOCK register and the port configuration are not locked. 1: GPIO_LOCK register is locked until the MCU reset. LOCK key configuration sequence Write 1→Write 0→Write 1→ Read 0→ Read 1 **Note:** The value of LK[15:0] must be held during the LOCK Key writing sequence.
15:0	LKy	Port lock bit y(y=0..15) These bits are set and cleared by software. 0: The corresponding bit port configuration is not locked. 1: The corresponding bit port configuration is locked when LKK bit is “1”.

8.5.8. Port bit speed register (GPIOx_SPD, x=A..E)

Address offset: 0x3C
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPD15	SPD 14	SPD 13	SPD 12	SPD 11	SPD 10	SPD 9	SPD 8	SPD 7	SPD 6	SPD 5	SPD 4	SPD 3	SPD 2	SPD 1	SPD 0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	SPDy	Set very high output speed(120MHz) when MDx is 0b11. If the port output speed is more than 50MHz, set this bit to 1 and set MDx to 0b11.These bits are set and cleared by software. 0: No effect 1: Max speed more than 50MHz.(MDx required to be set to 0b11 together) Note: When the port output speed is more than 50 MHz, the user should enable the I/O compensation cell. Refer to CPS_EN bit in AFIO_CPSCTL register.

8.5.9. Event control register (AFIO_EC)

Address offset: 0x00
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								EOE	PORT[2:0]			PIN[3:0]			
								rw	rw			rw			

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	EOE	Event output enable Set and cleared by software.When this bit is set, the Cortex EVENTOUT output is connected to the I/O selected by the PORT[2:0] and PIN[3:0] bits.
6:4	PORT[2:0]	Event output port selection Set and cleared by software.Select the port to output the Cortex EVENTOUT signal. 000: Select PORT A 001: Select PORT B 010: Select PORT C 011: Select PORT D 100: Select PORT E
3:0	PIN[3:0]	Event output pin selection Set and cleared by software. Select the pin to output the Cortex EVENTOUT signal. 0000: Select Pin 0 0001: Select Pin 1 0010: Select Pin 2 ... 1111: Select Pin 15

8.5.10. AFIO port configuration register 0 (AFIO_PCF0)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		TIMER1ITR0_ REMAP	SPI2_ REMAP	Reserved	SWJ_ CFG[2:0]			Reserved	CAN1_ REMAP	Reserved	ADC1_ETRGRT_REMAP	Reserved	ADC0_ETRGRT_REMAP	Reserved	TIMER4CH3_IREMAP
		rw	rw		w				rw		rw		rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD01_ REMAP	CAN0_REMAP [1:0]		TIMER3_ REMAP	TIMER2_REMAP [1:0]		TIMER1_REMAP [1:0]		TIMER0_REMAP [1:0]		USART2_ REMAP[1:0]		USART1_ REMAP	USART0_ REMAP	I2C0_REMAP	SPI0_REMAP
rw	rw		rw	rw		rw		rw		rw		rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29	TIMER1ITI1_REMAP	TIMER1 internal trigger 1 remapping These bits are set and cleared by software. It controls the TMER1_ITI1 internal mapping. 0: Disable the remapping function 1: Connect USBFS SOF (Start of Frame) output to TIMER1_ITI1 for calibration purposes
28	SPI2_REMAP	SPI2/I2S2 remapping This bit is set and reset by software. 0: Disable the remapping function (SPI2_NSS-I2S2_WS/PA15, SPI2_SCK-I2S2_CK/PB3, SPI2_MISO/PB4, SPI2_MOSI-I2S_SD/PB5) 1: Enable the remapping function fully (SPI2_NSS-I2S2_WS/PA4, SPI2_SCK-I2S2_CK/PC10, SPI2_MISO/PC11, SPI2_MOSI-I2S_SD/PC12)
27	Reserved	Must be kept at reset value.
26:24	SWJ_CFG[2:0]	Serial wire JTAG configuration These bits are write-only. 000: JTAG-DP Enabled and SW-DP Enabled(Reset State) 001: JTAG-DP Enabled and SW-DP Enabled , but without NJTRST 010: Disable JTAG-DP and Enable SW-DP 100: Disable JTAG-DP and SW-DP Other: Undefined **Note**: Only one of the three bits can be set at a time.
23	Reserved	Must be kept at reset value.
22	CAN1_REMAP	CAN1 I/O remapping This bit is set and cleared by software.It controls the CAN1_TX and CAN1_RX pins 0: Disable the remapping function (CAN1_RX/PB12,CAN_TX/PB13) 1: Enable the remapping function (CAN1_RX/PB5,CAN_TX/PB6)
21	Reserved	Must be kept at reset value.
20	ADC1_ETRGRT_REMAP	ADC 1 external trigger routine conversion remapping This bit is set and reset by software. 0: Connect the ADC1 external signal trigger routine conversion to EXTI11. 1: Connect the ADC1 external signal trigger routine conversion to TIM7_TRGO.
19	Reserved	Must be kept at reset value.
18	ADC0_ETRGRT_REMAP	ADC 0 external trigger routine conversion remapping This bit is set and reset by software. 0: Connect the ADC0 external signal trigger routine conversion to EXTI11. 1: Connect the ADC0 external signal trigger routine conversion to TIM7_TRGO.
17	Reserved	Must be kept at reset value.
16	TIMER4CH3_IREMA	TIMER4 channel3 internal remapping

	P	This bit is set and reset by software. 0: ConnectTIMER4_CH3 to PA3. 1: Connect the IRC40K internal clock to TIMER4_CH3 input in order to calibration.
15	PD01_REMAP	Port D0/Port D1 mapping to OSC_IN/OSC_OUT This bit is set and cleared by software. 0: Not remap 1: PD0 remapped to OSC_IN, PD1 remapped to OSC_OUT.
14:13	CAN0_REMAP[1:0]	CAN0 alternate interface remapping These bits are set and cleared by software. 00: Disable the remapping function (CAN0_RX/PA11,CAN0_TX/PA12) 01: Not used 10: Enable the remapping function partially(CAN0_RX/PB8,CAN0_TX/PB9) 11: Enable the remapping function fully (CAN0_RX/PD0,CAN0_TX/PD1)
12	TIMER3_REMAP	TIMER3 remapping This bit is set and reset by software 0: Disable the remapping function(TIMER3_CH0/PB6, TIMER3_CH1/PB7, TIMER3_CH2/PB8, TIMER3_CH3/PB9) 1: Enable the remapping function fully(TIMER3_CH0/PD12, TIMER3_CH1/PD13, TIMER3_CH2/PD14, TIMER3_CH3/PD15)
11:10	TIMER2_REMAP [1:0]	TIMER2 remapping These bits are set and reset by software 00: Disable the remapping function(TIMER2_CH0/PA6, TIMER2_CH1/PA7, TIMER2_CH2/PB0, TIMER2_CH3/PB1) 01: Not used 10: Enable the remapping function partially (TIMER2_CH0/PB4, TIMER2_CH1/PB5, TIMER2_CH2/PB0, TIMER2_CH3/PB1) 11: Enable the remapping function fully (TIMER2_CH0/PC6, TIMER2_CH1/PC7, TIMER2_CH2/PC8, TIMER2_CH3/PC9)
9:8	TIMER1_REMAP [1:0]	TIMER1 remapping These bits are set and reset by software 00: Disable the remapping function(TIMER1_CH0-TIMER1_ETI/PA0, TIMER1_CH1 /PA1, TIMER1_CH2/PA2, TIMER1_CH3/PA3) 01: Enable the remapping function partially(TIMER1_CH0-TIMER1_ETI/PA15, TIMER1_CH1/PB3, TIMER1_CH2/PA2, TIMER1_CH3/PA3) 10: not used 11: Enable the remapping function fully(TIMER1_CH0-TIMER1_ETI/PA15, TIMER1_CH1/PB3, TIMER1_CH2/PB10, TIMER1_CH3/PB11)
7:6	TIMER0_REMAP [1:0]	TIMER0 remapping These bits are set and reset by software 00: Disable the remapping function(TIMER0_ETI/PA12, TIMER0_CH0/ PA8, TIMER0_CH1/PA9, TIMER0_CH2/PA10,TIMER0_CH3/PA11, TIMER0_BRKIN/PB12, TIMER0_CH0_ON/PB13, TIMER0_CH1_ON/PB14, TIMER0_CH2_ON/PB15)

		01: Enable the remapping function partially(TIMER0_ETI/PA12, TIMER0_CH0/PA8, TIMER0_CH1/PA9, TIMER0_CH2/PA10,TIMER0_CH3/PA11, TIMER0_BRKIN/PA6, TIMER0_CH0_ON/PA7, TIMER0_CH1_ON/PB0, TIMER0_CH2_ON/PB1) 10: Not used 11: Enable the remapping function fully(TIMER0_ETI/PE7, TIMER0_CH0/ PE9, TIMER0_CH1/PE11, TIMER0_CH2/PE13, TIMER0_CH3/PE14, TIMER0_BRKIN/PE15, TIMER0_CH0_ON/PE8, TIMER0_CH1_ON/PE10, TIMER0_CH2_ON/PE12)
5:4	USART2_REMAP [1:0]	USART2 remapping These bits are set and reset by software 00: Disable the remapping function (USART2_TX/PB10, USART2_RX /PB11, USART2_CK/PB12, USART2_CTS/PB13, USART2_RTS/PB14) 01: Enable the remapping function partially(USART2_TX/PC10, USART2_RX /PC11, USART2_CK/PC12, USART2_CTS/PB13, USART2_RTS/PB14) 10: Not used 11: Enable the remapping function fully(USART2_TX/PD8, USART2_RX /PD9, USART2_CK/PD10, USART2_CTS/PD11, USART2_RTS/PD12)
3	USART1_REMAP	USART1 remapping This bit is set and reset by software 0: Disable the remapping function(USART1_CTS/PA0, USART1_RTS/PA1, USART1_TX/PA2, USART1_RX /PA3, USART1_CK/PA4) 1: Enable the remapping function(USART1_CTS/PD3, USART1_RTS/PD4, USART1_TX/PD5, USART1_RX /PD6, USART1_CK/PD7)
2	USART0_REMAP	USART0 remapping This bit is set and reset by software 0: Disable the remapping function (USART0_TX/PA9, USART0_RX /PA10) 1: Enable the remapping function (USART0_TX/PB6, USART0_RX /PB7)
1	I2C0_REMAP	I2C0 remapping This bit is set and reset by software 0: Disable the remapping function (I2C0_SCL/PB6, I2C0_SDA /PB7) 1: Enable the remapping function (I2C0_SCL/PB8, I2C0_SDA /PB9)
0	SPI0_REMAP	SPI0 remapping This bit is set and cleared by software. 0: Disable the remapping function (SPI0_NSS/PA4, SPI0_SCK /PA5, SPI0_MISO /PA6,SPI0_MOSI /PA7, SPI0_IO2 /PA2, SPI0_IO3 /PA3) 1: Enable the remapping function (SPI0_NSS/PA15, SPI0_SCK /PB3, SPI0_MISO /PB4, SPI0_MOSI /PB5, SPI0_IO2 /PB6, SPI0_IO3 /PB7)

8.5.11. EXTI sources selection register 0 (AFIO_EXTISS0)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI3_SS [3:0]				EXTI2_SS [3:0]				EXTI1_SS [3:0]				EXTI0_SS [3:0]			
rw				rw				rw				rw			

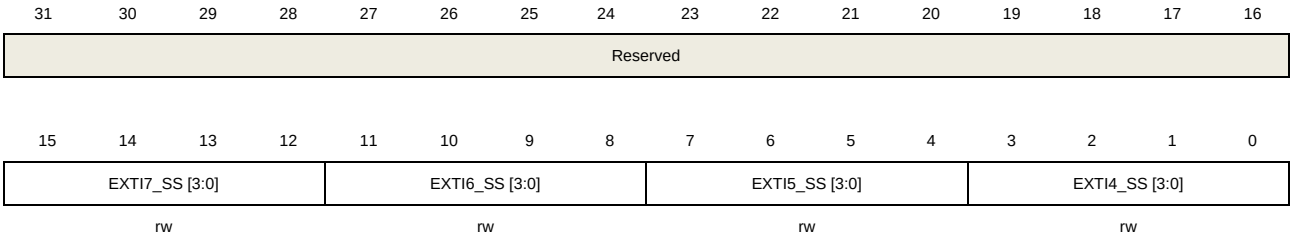
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	EXTI3_SS[3:0]	EXTI3 sources selection 0000: PA3 pin 0001: PB3 pin 0010: PC3 pin 0011: PD3 pin 0100: PE3 pin Other configurations are reserved.
11:8	EXTI2_SS[3:0]	EXTI2 sources selection 0000: PA2 pin 0001: PB2 pin 0010: PC2 pin 0011: PD2 pin 0100: PE2 pin Other configurations are reserved.
7:4	EXTI1_SS[3:0]	EXTI1 sources selection 0000: PA1 pin 0001: PB1 pin 0010: PC1 pin 0011: PD1 pin 0100: PE1 pin Other configurations are reserved.
3:0	EXTI0_SS[3:0]	EXTI0 sources selection 0000: PA0 pin 0001: PB0 pin 0010: PC0 pin 0011: PD0 pin 0100: PE0 pin Other configurations are reserved.

8.5.12. EXTI sources selection register 1 (AFIO_EXTISS1)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	EXTI7_SS[3:0]	EXTI7 sources selection 0000: PA7 pin 0001: PB7 pin 0010: PC7 pin 0011: PD7 pin 0100: PE7 pin Other configurations are reserved.
11:8	EXTI6_SS[3:0]	EXTI6 sources selection 0000: PA6 pin 0001: PB6 pin 0010: PC6 pin 0011: PD6 pin 0100: PE6 pin Other configurations are reserved.
7:4	EXTI5_SS[3:0]	EXTI5 sources selection 0000: PA5 pin 0001: PB5 pin 0010: PC5 pin 0011: PD5 pin 0100: PE5 pin Other configurations are reserved.
3:0	EXTI4_SS[3:0]	EXTI4 sources selection 0000: PA4 pin 0001: PB4 pin 0010: PC4 pin 0011: PD4 pin 0100: PE4 pin Other configurations are reserved.

8.5.13. EXTI sources selection register 2 (AFIO_EXTISS2)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI11_SS [3:0]				EXTI10_SS [3:0]				EXTI9_SS [3:0]				EXTI8_SS [3:0]			
rw				rw				rw				rw			

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	EXTI11_SS[3:0]	EXTI11 sources selection 0000: PA11 pin 0001: PB11 pin 0010: PC11 pin 0011: PD11 pin 0100: PE11 pin Other configurations are reserved.
11:8	EXTI10_SS[3:0]	EXTI10 sources selection 0000: PA10 pin 0001: PB10 pin 0010: PC10 pin 0011: PD10 pin 0100: PE10 pin Other configurations are reserved.
7:4	EXTI9_SS[3:0]	EXTI9 sources selection 0000: PA9 pin 0001: PB9 pin 0010: PC9 pin 0011: PD9 pin 0100: PE9 pin Other configurations are reserved.
3:0	EXTI8_SS[3:0]	EXTI8 sources selection 0000: PA8 pin 0001: PB8 pin 0010: PC8 pin 0011: PD8 pin 0100: PE8 pin

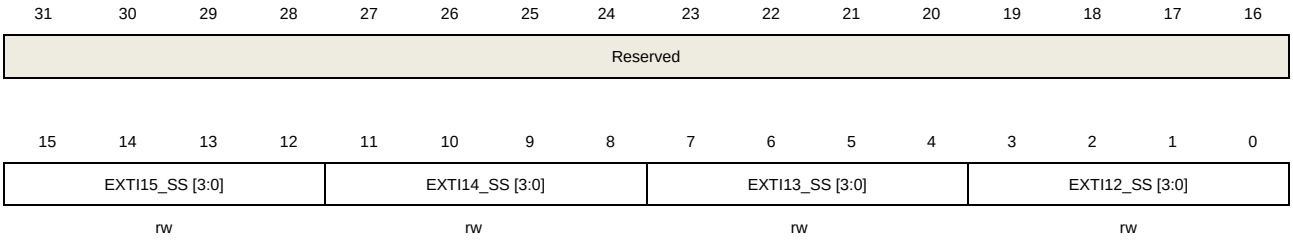
Other configurations are reserved.

8.5.14. EXTI sources selection register 3 (AFIO_EXTISS3)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	EXTI15_SS[3:0]	EXTI15 sources selection 0000: PA15 pin 0001: PB15 pin 0010: PC15 pin 0011: PD15 pin 0100: PE15 pin Other configurations are reserved.
11:8	EXTI14_SS[3:0]	EXTI14 sources selection 0000: PA14 pin 0001: PB14 pin 0010: PC14 pin 0011: PD14 pin 0100: PE14 pin Other configurations are reserved.
7:4	EXTI13_SS[3:0]	EXTI13 sources selection 0000: PA13 pin 0001: PB13 pin 0010: PC13 pin 0011: PD13 pin 0100: PE13 pin Other configurations are reserved.
3:0	EXTI12_SS[3:0]	EXTI12 sources selection 0000: PA12 pin 0001: PB12 pin

0010: PC12 pin

0011: PD12 pin

0100: PE12 pin

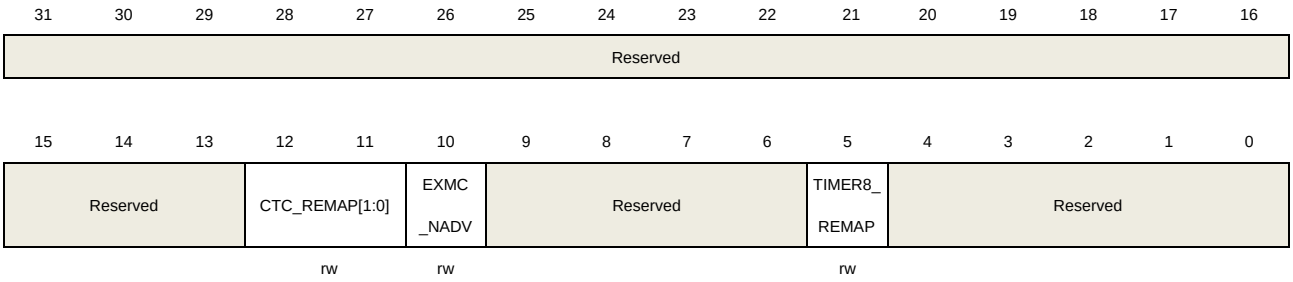
Other configurations are reserved.

8.5.15. AFIO port configuration register 1 (AFIO_PCF1)

Address offset: 0x1C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:11	CTC_REMAP [1:0]	CTC remapping These bits are set and cleared by software, they control the mapping of the CTC_SYNC alternate function onto the GPIO ports 00: Disable the remapping function (PA8) 01: Enable the remapping function (PD15) 10/11: Reserved
10	EXMC_NADV	EXMC_NADV connect/disconnect This bit is set and cleared by software, it controls the use of optional EXMC_NADV signal. 0: The NADV signal is connected to the output(default) 1: The NADV signal is not connected. The I/O pin can be used by another peripheral.
9:6	Reserved	Must be kept at reset value.
5	TIMER8_REMAP	TIMER8 remapping This bit is set and cleared by software, it controls the mapping of the TIMER8_CH0 and TIMER8_CH1 alternate function onto the GPIO ports. 0: Disable the remapping function (TIMER8_CH0 on PA2 and TIMER8_CH1 on PA3) 1: Enable the remapping function (TIMER8_CH0 on PE5 and TIMER8_CH1 on PE6)

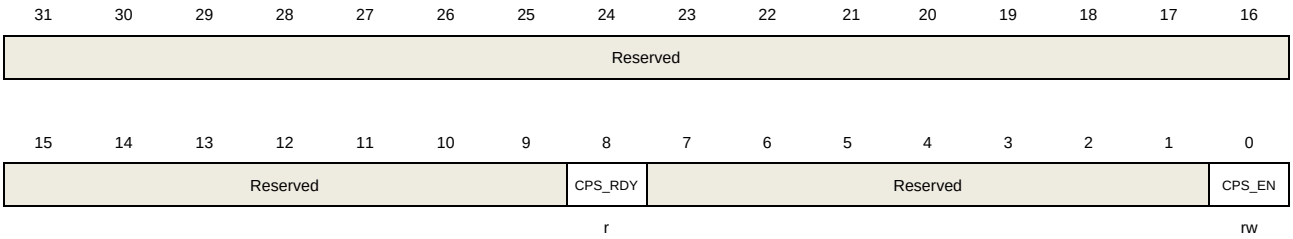
4:0	Reserved	Must be kept at reset value.

8.5.16. IO compensation control register (AFIO_CPSCTL)

Address offset: 0x20

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	CPS_RDY	I/O compensation cell is ready or not. This bit is read-only. 0: I/O compensation cell is not ready 1: I/O compensation cell is ready
7:1	Reserved	Must be kept at reset value.
0	CPS_EN	I/O compensation cell enable When the port output speed is more than 50 MHz, the user should enable the I/O compensation cell. 0: I/O compensation cell is disabled 1: I/O compensation cell is enabled

9. CRC calculation unit (CRC)

9.1. Overview

A cyclic redundancy check (CRC) is an error-detecting code commonly used in digital networks and storage devices to detect accidental changes to raw data.

This CRC calculation unit can be used to calculate 32 bit CRC code with fixed polynomial.

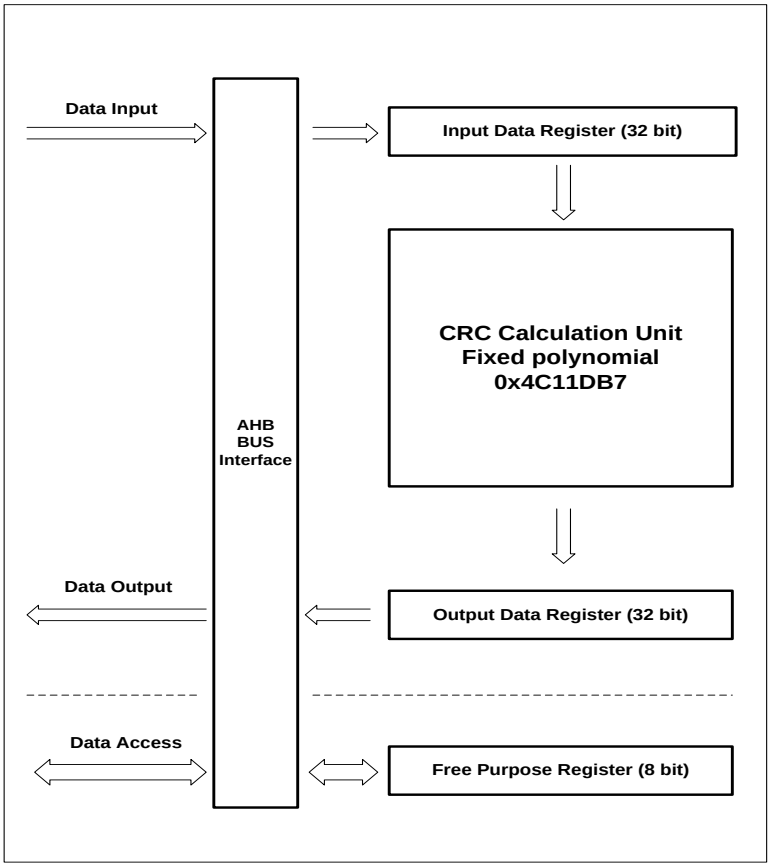
9.2. Characteristics

- 32-bit data input and 32-bit data output. Calculation period is 4 AHB clock cycles for 32-bit input data size from data entered to the calculation result available.
- Free 8-bit register is unrelated to calculation and can be used for any other goals by any other peripheral devices.
- Fixed polynomial: 0x4C11DB7

  $X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^{8}+X^{7}+X^{5}+X^{4}+X^{2}+X+1$

This 32-bit CRC polynomial is a common polynomial used in Ethernet.

Figure 9-1. Block diagram of CRC calculation unit



9.3. Function overview

- CRC calculation unit is used to calculate the 32-bit raw data, and CRC_DATA register will receive the raw data and store the calculation result.

  If the CRC_DATA register has not been cleared by software setting the CRC_CTL register, the new input raw data will be calculated based on the result of previous value of CRC_DATA.

  CRC calculation will spend 4 AHB clock cycles for 32-bit data size, during this period AHB will not be hanged because of the existence of the 32-bit input buffer.

- This module supplies an 8-bit free register CRC_FDATA.

  CRC_FDATA is unrelated to the CRC calculation, any value you write in will be read out at anytime.

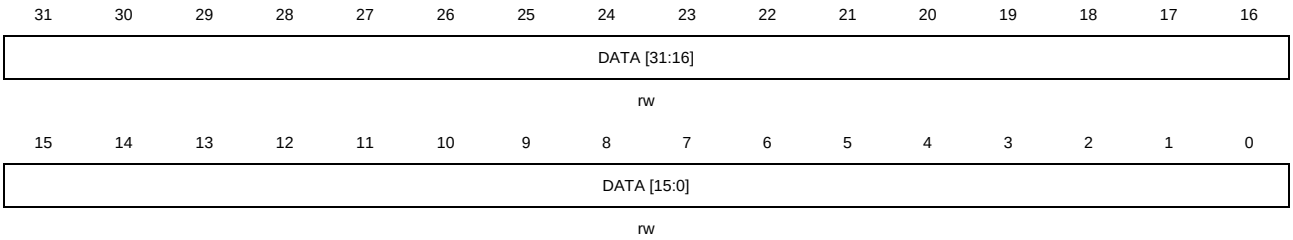
9.4. Register definition

CRC base address: 0x4002 3000

9.4.1. Data register (CRC_DATA)

Address offset: 0x00
Reset value: 0xFFFF FFFF

This register has to be accessed by word (32-bit).

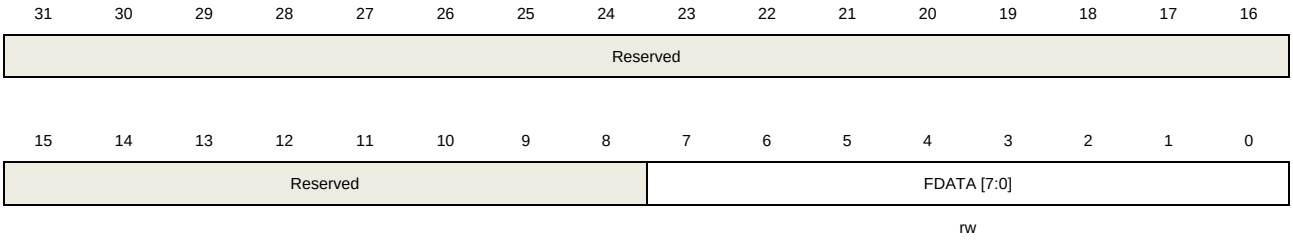


Bits	Fields	Descriptions
31:0	DATA [31:0]	CRC calculation result bits Software writes and reads. This register is used to calculate new data, and the register can be written the new data directly. Written value cannot be read because the read value is the previous CRC calculation result.

9.4.2. Free data register (CRC_FDATA)

Address offset: 0x04
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	FDATA [7:0]	Free Data Register bits Software writes and reads. These bits are unrelated with CRC calculation. This byte can be used for any goal

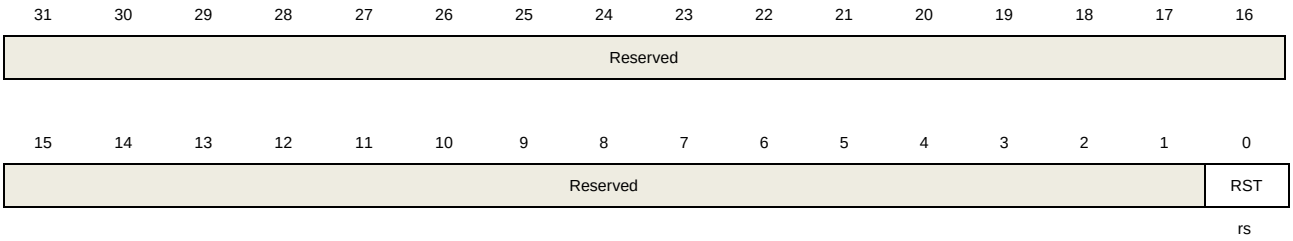
by any other peripheral. The CRC_CTL register will take no effect to the byte.

9.4.3. Control register (CRC_CTL)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	RST	Set this bit can reset the CRC_DATA register to the value of 0xFFFFFFFF then automatically cleared itself to 0 by hardware. This bit will take no effect to CRC_FDATA. Software writes and reads.

10. Direct memory access controller (DMA)

10.1. Overview

The direct memory access (DMA) controller provides a hardware method of transferring data between peripherals and/or memory without intervention from the CPU, thereby freeing up bandwidth for other system functions. Data can be quickly moved by DMA between peripherals and memory as well as memory and memory without any CPU actions. There are 12 channels in the DMA controller (7 for DMA0 and 5 for DMA1). Each channel is dedicated to manage memory access requests from one or more peripherals. An arbiter is implemented inside to handle the priority among DMA requests.

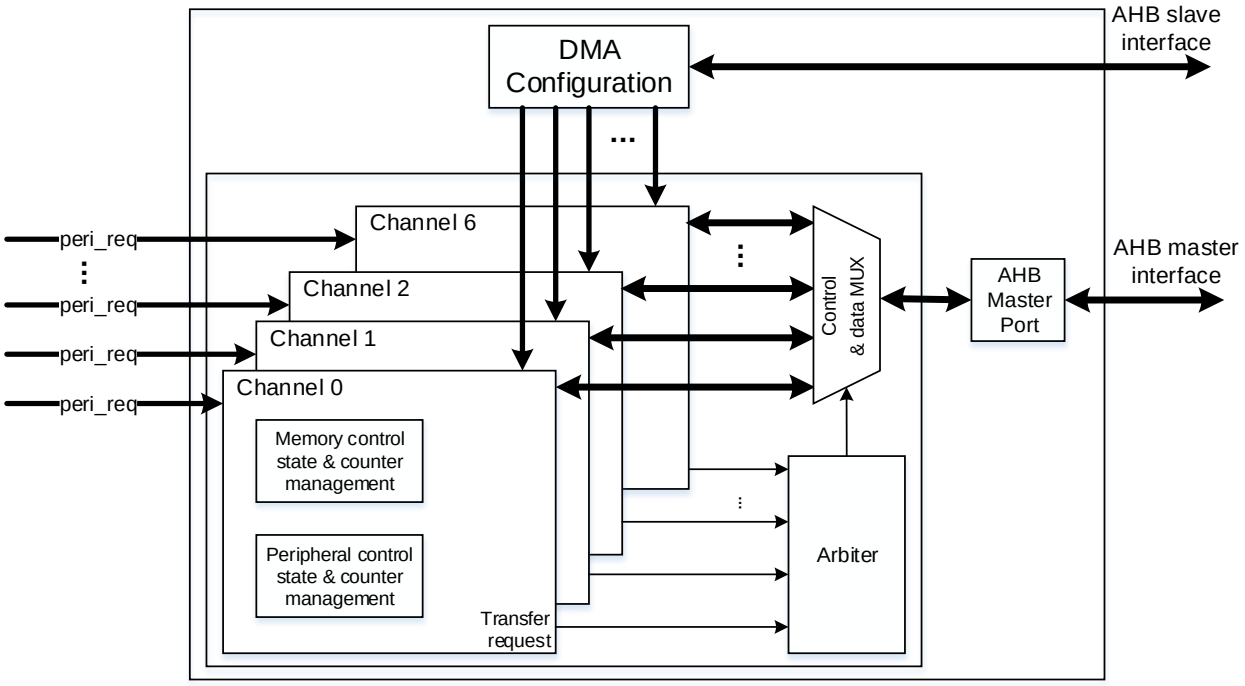
The system bus is shared by the DMA controller and the Cortex®-M4 core. When the DMA and the CPU are targeting the same destination, the DMA access may stop the CPU access to the system bus for some bus cycles. Round-robin scheduling is implemented in the bus matrix to ensure at least half of the system bus bandwidth for the CPU.

10.2. Characteristics

- Programmable length of data to be transferred, max to 65536.
- 12 channels and each channel is configurable (7 for DMA0 and 5 for DMA1).
- AHB and APB peripherals, FLASH, SRAM can be accessed as source and destination.
- Each channel is connected to the fixed hardware DMA request.
- Software DMA channel priority (low, medium, high, ultra high) and hardware DMA channel priority (DMA channel 0 has the highest priority and DMA channel 6 has the lowest priority).
- Support independent 8, 16, 32-bit memory and peripheral transfer.
- Support independent fixed and increasing address generation algorithm of memory and peripheral.
- Support circular transfer mode.
- Support peripheral to memory, memory to peripheral, and memory to memory transfers.
- One separate interrupt per channel with three types of event flags.
- Support interrupt enable and clear.

10.3. Block diagram

Figure 10-1. Block diagram of DMA



As shown in ***Figure 10-1. Block diagram of DMA***, a DMA controller consists of four main parts:

- DMA configuration through AHB slave interface
- Data transmission through two AHB master interfaces for memory access and peripheral access
- An arbiter inside to manage multiple peripheral requests coming at the same time
- Channel management to control address/data selection and data counting

10.4. Function overview

10.4.1. DMA operation

Each DMA transfer consists of two operations, including the loading of data from the source and the storage of the loaded data to the destination. The source and destination addresses are computed by the DMA controller based on the programmed values in the DMA_CHxPADDR, DMA_CHxMADDR, and DMA_CHxCTL registers. The DMA_CHxCNT register controls how many transfers to be transmitted on the channel. The PWIDTH and MWIDTH bits in the DMA_CHxCTL register determine how many bytes to be transmitted in a transfer.

Suppose DMA_CHxCNT is 4, and both PNAGA and MNAGA are set. The DMA transfer operations for each combination of PWIDTH and MWIDTH are shown in ***Table 10-1. DMA***

transfer operation.

Table 10-1. DMA transfer operation

Transfer size		Transfer operations	
Source	Destination	Source	Destination
32 bits	32 bits	1: Read B3B2B1B0[31:0] @0x0 2: Read B7B6B5B4[31:0] @0x4 3: Read BBBAB9B8[31:0] @0x8 4: Read BFBEBDBC[31:0] @0xC	1: Write B3B2B1B0[31:0] @0x0 2: Write B7B6B5B4[31:0] @0x4 3: Write BBBAB9B8[31:0] @0x8 4: Write BFBEBDBC[31:0] @0xC
32 bits	16 bits	1: Read B3B2B1B0[31:0] @0x0 2: Read B7B6B5B4[31:0] @0x4 3: Read BBBAB9B8[31:0] @0x8 4: Read BFBEBDBC[31:0] @0xC	1: Write B1B0[7:0] @0x0 2: Write B5B4[7:0] @0x2 3: Write B9B8[7:0] @0x4 4: Write BDBC[7:0] @0x6
32 bits	8 bits	1: Read B3B2B1B0[31:0] @0x0 2: Read B7B6B5B4[31:0] @0x4 3: Read BBBAB9B8[31:0] @0x8 4: Read BFBEBDBC[31:0] @0xC	1: Write B0[7:0] @0x0 2: Write B4[7:0] @0x1 3: Write B8[7:0] @0x2 4: Write BC[7:0] @0x3
16 bits	32 bits	1: Read B1B0[15:0] @0x0 2: Read B3B2[15:0] @0x2 3: Read B5B4[15:0] @0x4 4: Read B7B6[15:0] @0x6	1: Write 0000B1B0[31:0] @0x0 2: Write 0000B3B2[31:0] @0x4 3: Write 0000B5B4[31:0] @0x8 4: Write 0000B7B6[31:0] @0xC
16 bits	16 bits	1: Read B1B0[15:0] @0x0 2: Read B3B2[15:0] @0x2 3: Read B5B4[15:0] @0x4 4: Read B7B6[15:0] @0x6	1: Write B1B0[15:0] @0x0 2: Write B3B2[15:0] @0x2 3: Write B5B4[15:0] @0x4 4: Write B7B6[15:0] @0x6
16 bits	8 bits	1: Read B1B0[15:0] @0x0 2: Read B3B2[15:0] @0x2 3: Read B5B4[15:0] @0x4 4: Read B7B6[15:0] @0x6	1: Write B0[7:0] @0x0 2: Write B2[7:0] @0x1 3: Write B4[7:0] @0x2 4: Write B6[7:0] @0x3
8 bits	32 bits	1: Read B0[7:0] @0x0 2: Read B1[7:0] @0x1 3: Read B2[7:0] @0x2 4: Read B3[7:0] @0x3	1: Write 000000B0[31:0] @0x0 2: Write 000000B1[31:0] @0x4 3: Write 000000B2[31:0] @0x8 4: Write 000000B3[31:0] @0xC
8 bits	16 bits	1: Read B0[7:0] @0x0 2: Read B1[7:0] @0x1 3: Read B2[7:0] @0x2 4: Read B3[7:0] @0x3	1, Write 00B0[15:0] @0x0 2, Write 00B1[15:0] @0x2 3, Write 00B2[15:0] @0x4 4, Write 00B3[15:0] @0x6
8 bits	8 bits	1: Read B0[7:0] @0x0 2: Read B1[7:0] @0x1 3: Read B2[7:0] @0x2 4: Read B3[7:0] @0x3	1, Write B0[7:0] @0x0 2, Write B1[7:0] @0x1 3, Write B2[7:0] @0x2 4, Write B3[7:0] @0x3

The CNT bits in the DMA_CHxCNT register control how many data to be transmitted on the channel and must be configured before enabling the CHEN bit in the register. During the transmission, the CNT bits indicate the remaining number of data to be transferred.

The DMA transmission is disabled by clearing the CHEN bit in the DMA_CHxCTL register.

- If the DMA transmission has not been completed when the CHEN bit is cleared, two situations may occur when restart this DMA channel:
  - If no register configuration operations of the channel occur before restarting the DMA channel, the DMA will continue to complete the rest of the transmission.
  - If any register (DMA_CHxCNT / DMA_CHxPADDR / DMA_CHxMADDR) configuration operations occur, the DMA will restart a new transmission.
- If the DMA transmission has been finished when clearing the CHEN bit, enabling the DMA channel without any register (DMA_CHxCNT / DMA_CHxPADDR / DMA_CHxMADDR) configuration operation will not launch any DMA transfer.

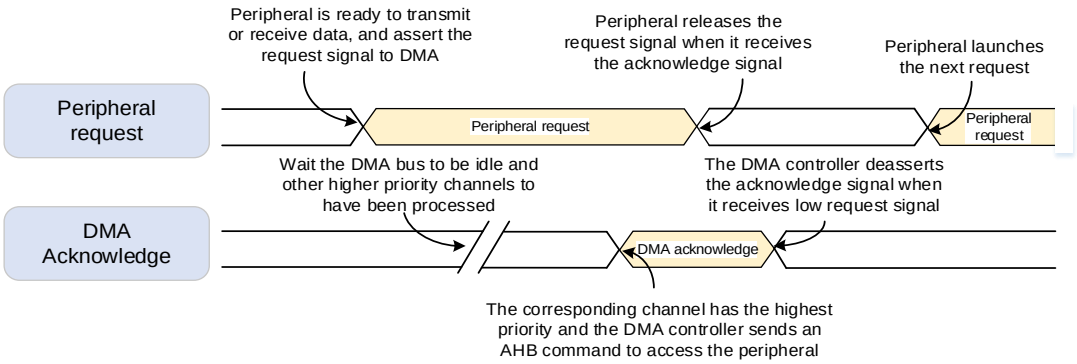
10.4.2. Peripheral handshake

To ensure a well-organized and efficient data transfer, a handshake mechanism is introduced between the DMA and peripherals, including a request signal and an acknowledge signal:

- Request signal asserted by peripheral to DMA controller, indicating that the peripheral is ready to transmit or receive data.
- Acknowledge signal responded by DMA to peripheral, indicating that the DMA controller has initiated an AHB command to access the peripheral.

Figure 10-2. Handshake mechanism shows how the handshake mechanism works between the DMA controller and peripherals.

Figure 10-2. Handshake mechanism



10.4.3. Arbitration

When two or more requests are received at the same time, the arbiter determines which request is served based on the priorities of the channels. There are two-stage priorities, including the software priority and the hardware priority. The arbiter determines which channel is selected to respond according to the following priority rules:

- Software priority: Four levels, including low, medium, high and ultra high by configuring the PRIO bits in the DMA_CHxCTL register.
- For channels with equal software priority level, priority is given to the channel with lower channel number.

10.4.4. Address generation

Two kinds of address generation algorithm are implemented independently for memory and peripheral, including the fixed mode and the increased mode. The PNAGA and MNAGA bit in the DMA_CHxCTL register are used to configure the next address generation algorithm of peripheral and memory.

In the fixed mode, the next address is always equal to the base address configured in the base address registers (DMA_CHxPADDR, DMA_CHxMADDR).

In the increasing mode, the next address is equal to the current address plus 1 or 2 or 4, depending on the transfer data width.

10.4.5. Circular mode

Circular mode is implemented to handle continue peripheral requests (for example, ADC scan mode). The circular mode is enabled by setting the CMEN bit in the DMA_CHxCTL register.

In circular mode, the CNT bits are automatically reloaded with the pre-programmed value and the full transfer finish flag is asserted at the end of every DMA transfer. DMA can always respond the peripheral request until the CHEN bit in the DMA_CHxCTL register is cleared.

10.4.6. Memory to memory mode

The memory to memory mode is enabled by setting the M2M bit in the DMA_CHxCTL register. In this mode, the DMA channel can also work without being triggered by a request from a peripheral. The DMA channel starts transferring as soon as it is enabled by setting the CHEN bit in the DMA_CHxCTL register, and completed when the DMA_CHxCNT register reaches zero.

10.4.7. Channel configuration

When starting a new DMA transfer, it is recommended to respect the following steps:

1. Read the CHEN bit and judge whether the channel is enabled or not. If the channel is enabled, clear the CHEN bit by software. When the CHEN bit is read as '0', configuring and starting a new DMA transfer is allowed.
2. Configure the M2M bit and DIR bit in the DMA_CHxCTL register to set the transfer mode.
3. Configure the CMEN bit in the DMA_CHxCTL register to enable/disable the circular mode.
4. Configure the PRIO bits in the DMA_CHxCTL register to set the channel software priority.
5. Configure the memory and peripheral transfer width, memory and peripheral address generation algorithm in the DMA_CHxCTL register.
6. Configure the enable bit for full transfer finish interrupt, half transfer finish interrupt, transfer error interrupt in the DMA_CHxCTL register.
7. Configure the DMA_CHxPADDR register for setting the peripheral base address.
8. Configure the DMA_CHxMADDR register for setting the memory base address.
9. Configure the DMA_CHxCNT register to set the total transfer data number.
10. Configure the CHEN bit with '1' in the DMA_CHxCTL register to enable the channel.

10.4.8. Interrupt

Each DMA channel has a dedicated interrupt. There are three types of interrupt event, including full transfer finish, half transfer finish, and transfer error.

Each interrupt event has a dedicated flag bit in the DMA_INTF register, a dedicated clear bit in the DMA_INTC register, and a dedicated enable bit in the DMA_CHxCTL register. The

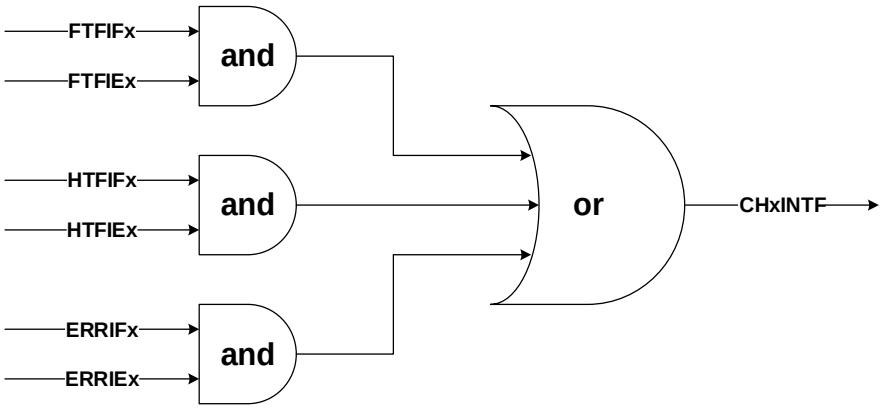
relationship is described in ***Table 10-2. Interrupt events***.

Table 10-2. Interrupt events

Interrupt event	Flag bit	Clear bit	Enable bit
	DMA_INTF	DMA_INTC	DMA_CHxCTL
Full transfer finish	FTFIF	FTFIFC	FTFIE
Half transfer finish	HTFIF	HTFIFC	HTFIE
Transfer error	ERRIF	ERRIFC	ERRIE

The DMA interrupt logic is shown in ***Figure 10-3. DMA interrupt logic***, an interrupt can be produced when any type of interrupt event occurs and is enabled on the channel.

Figure 10-3. DMA interrupt logic



Note: "x" indicates channel number (for DMA0, x=0…6, for DMA1, x=0…4).

10.4.9. DMA request mapping

Several requests from peripherals may be mapped to one DMA channel. They are logically ORed before entering the DMA. For details, see ***Figure 10-4. DMA0 request mapping*** and ***Figure 10-5. DMA1 request mapping***. The request of each peripheral can be independently enabled or disabled by programming the registers of the corresponding peripheral. The user has to ensure that only one request is enabled at a time on one channel. ***Table 10-3. DMA0 requests for each channel*** lists the supported request from peripheral for each channel of DMA0, and ***Table 10-4. DMA1 requests for each channel*** lists the supported request from

peripheral for each channel of DMA1.

Figure 10-4. DMA0 request mapping

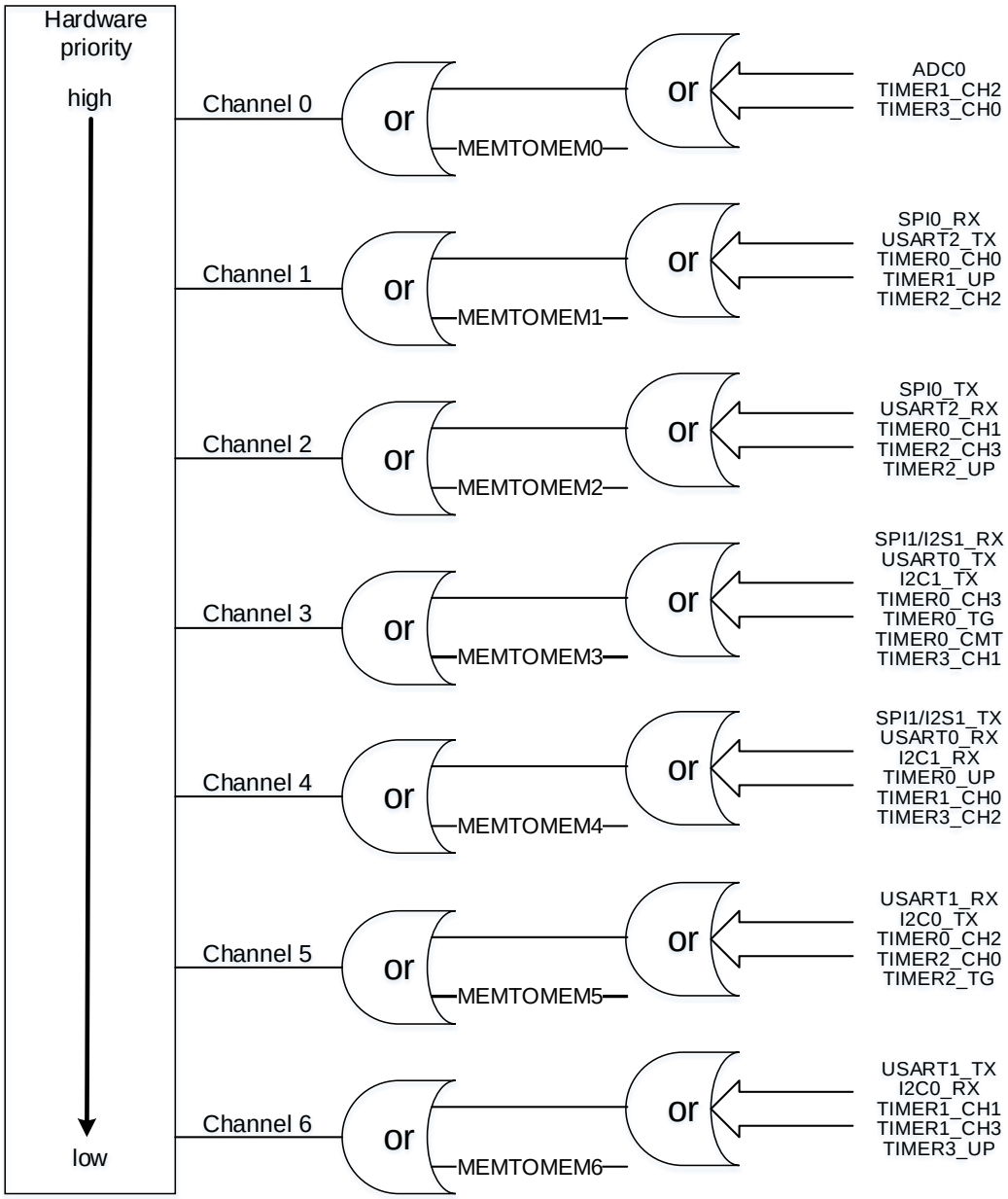


Table 10-3. DMA0 requests for each channel

Peripheral	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6
TIMER0	•	TIMER0_CH0	TIMER0_CH1	TIMER0_CH3 TIMER0_TG TIMER0_CMT	TIMER0_UP	TIMER0_CH2	•
TIMER1	TIMER1_CH2	TIMER1_UP	•	•	TIMER1_CH0	•	TIMER1_CH1 TIMER1_CH3
TIMER2	•	TIMER2_CH2	TIMER2_CH3 TIMER2_UP	•	•	TIMER2_CH0 TIMER2_TG	•
TIMER3	TIMER3_CH0	•	•	TIMER3_CH1	TIMER3_CH2	•	TIMER3_UP

Peripheral	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6
ADC0	ADC0	•	•	•	•	•	•
SPI/I2S	•	SPI0_RX	SPI0_TX	SPI/I2S1_RX	SPI/I2S1_TX	•	•
USART	•	USART2_TX	USART2_RX	USART0_TX	USART0_RX	USART1_RX	USART1_TX
I2C	•	•	•	I2C1_TX	I2C1_RX	I2C0_TX	I2C0_RX

Figure 10-5. DMA1 request mapping

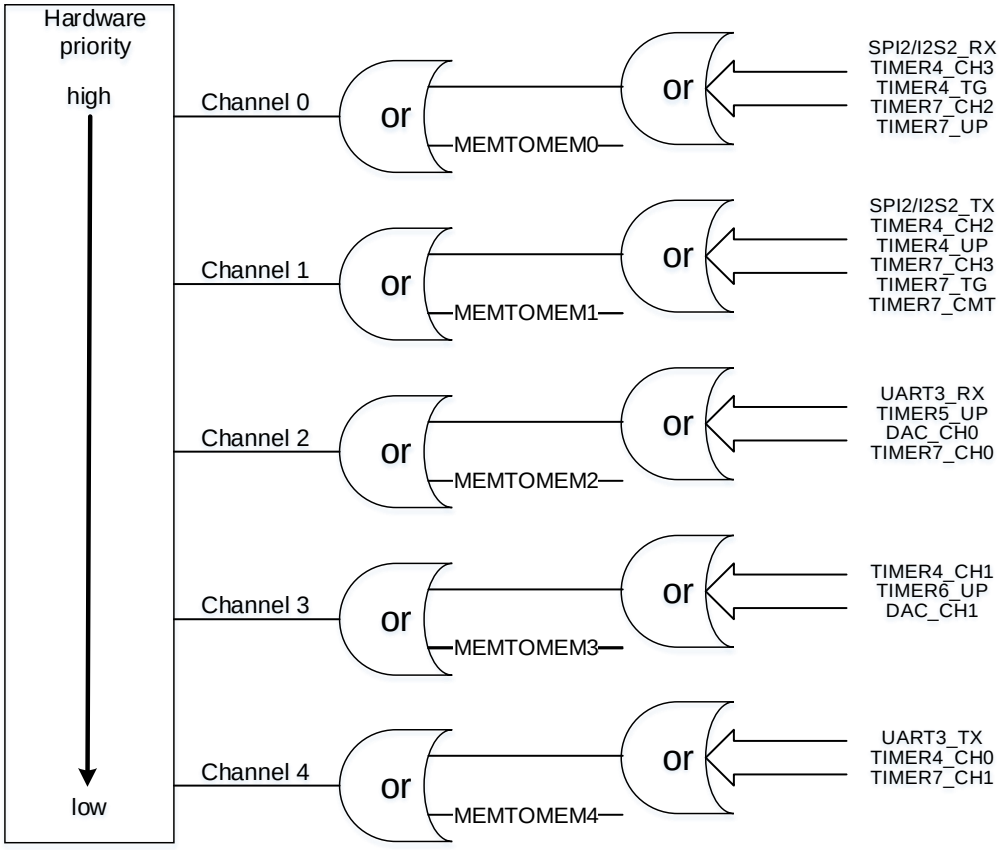


Table 10-4. DMA1 requests for each channel

Peripheral	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4
TIMER4	TIMER4_CH3 TIMER4_TG	TIMER4_CH2 TIMER4_UP	•	TIMER4_CH1	TIMER4_CH0
TIMER5	•	•	TIMER5_UP	•	•
TIMER6	•	•	•	TIMER6_UP	•
TIMER7	TIMER7_CH2 TIMER7_UP	TIMER7_CH3 TIMER7_TG TIMER7_CMT	TIMER7_CH0	•	TIMER7_CH1
DAC	•	•	DAC_CH0	DAC_CH1	•
SPI/I2S	SPI2/I2S2_RX	SPI2/I2S2_TX	•	•	•
USART	•	•	UART3_RX	•	UART3_TX

10.5. Register definition

DMA0 base address: 0x4002 0000
DMA1 base address: 0x4002 0400

Note: For DMA1 having 5 channels, all bits related to channel 5 and channel 6 in the following registers are not suitable for DMA1.

10.5.1. Interrupt flag register (DMA_INTF)

Address offset: 0x00
Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				ERRIF6	HTFIF6	FTFIF6	GIF6	ERRIF5	HTFIF5	FTFIF5	GIF5	ERRIF4	HTFIF4	FTFIF4	GIF4
				r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRIF3	HTFIF3	FTFIF3	GIF3	ERRIF2	HTFIF2	FTFIF2	GIF2	ERRIF1	HTFIF1	FTFIF1	GIF1	ERRIF0	HTFIF0	FTFIF0	GIF0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27/23/19/ 15/11/7/3	ERRIF**x**	Error flag of channel **x** (**x**=0…6) Hardware set and software cleared by configuring DMA_INTC register. 0: Transfer error has not occurred on channel **x** 1: Transfer error has occurred on channel **x**
26/22/18/ 14/10/6/2	HTFIF**x**	Half transfer finish flag of channel **x** (**x**=0…6) Hardware set and software cleared by configuring DMA_INTC register. 0: Half number of transfer has not finished on channel **x** 1: Half number of transfer has finished on channel **x**
25/21/17/ 13/9/5/1	FTFIF**x**	Full transfer finish flag of channel **x** (**x**=0…6) Hardware set and software cleared by configuring DMA_INTC register. 0: Transfer has not finished on channel **x** 1: Transfer has finished on channel **x**
24/20/16/ 12/8/4/0	GIF**x**	Global interrupt flag of channel **x** (**x**=0…6) Hardware set and software cleared by configuring DMA_INTC register. 0: None of ERRIF, HTFIF or FTFIF occurs on channel **x** 1: At least one of ERRIF, HTFIF or FTFIF occurs on channel **x**

10.5.2. Interrupt flag clear register (DMA_INTC)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				ERRIFC6	HTFIFC6	FTFIFC6	GIFC6	ERRIFC5	HTFIFC5	FTFIFC5	GIFC5	ERRIFC4	HTFIFC4	FTFIFC4	GIFC4
				w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRIFC3	HTFIFC3	FTFIFC3	GIFC3	ERRIFC2	HTFIC2	FTFIFC2	GIFC2	ERRIFC1	HTFIFC1	FTFIFC1	GIFC1	ERRIFC0	HTFIFC0	FTFIFC0	GIFC0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27/23/19/ 15/11/7/3	ERRIFC**x**	Clear bit for error flag of channel **x** (**x**=0…6) 0: No effect 1: Clear error flag
26/22/18/ 14/10/6/2	HTFIFC**x**	Clear bit for half transfer finish flag of channel **x** (**x**=0…6) 0: No effect 1: Clear half transfer finish flag
25/21/17/ 13/9/5/1	FTFIFC**x**	Clear bit for full transfer finish flag of channel **x** (**x**=0…6) 0: No effect 1: Clear full transfer finish flag
24/20/16/ 12/8/4/0	GIFC**x**	Clear global interrupt flag of channel **x** (**x**=0…6) 0: No effect 1: Clear GIF**x**, ERRIF**x**, HTFIF**x** and FTFIF**x** bits in the DMA_INTF register

10.5.3. Channel x control register (DMA_CHxCTL)

x = 0...6, where x is a channel number

Address offset: 0x08 + 0x14 × x

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	M2M	PRIO[1:0]		MWIDTH[1:0]		PWIDTH[1:0]		MNAGA	PNAGA	CMEN	DIR	ERRIE	HTFIE	FTFIE	CHEN
	rw	rw		rw		rw		rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions

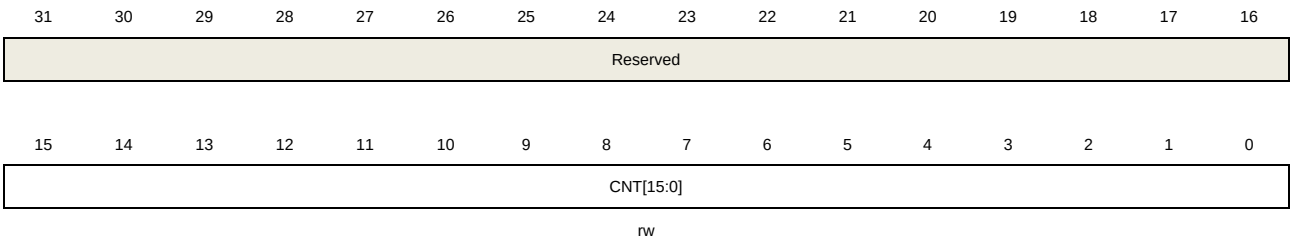
31:15	Reserved	Must be kept at reset value.
14	M2M	Memory to memory mode Software set and cleared 0: Disable memory to memory mode 1: Enable memory to memory mode This bit can not be written when CHEN is ‘1’.
13:12	PRIO[1:0]	Priority level Software set and cleared 00: Low 01: Medium 10: High 11: Ultra high These bits can not be written when CHEN is ‘1’.
11:10	MWIDTH[1:0]	Transfer data size of memory Software set and cleared 00: 8-bit 01: 16-bit 10: 32-bit 11: Reserved These bits can not be written when CHEN is ‘1’.
9:8	PWIDTH[1:0]	Transfer data size of peripheral Software set and cleared 00: 8-bit 01: 16-bit 10: 32-bit 11: Reserved These bits can not be written when CHEN is ‘1’.
7	MNAGA	Next address generation algorithm of memory Software set and cleared 0: Fixed address mode 1: Increasing address mode This bit can not be written when CHEN is ‘1’.
6	PNAGA	Next address generation algorithm of peripheral Software set and cleared 0: Fixed address mode 1: Increasing address mode This bit can not be written when CHEN is ‘1’.
5	CMEN	Circular mode enable Software set and cleared 0: Disable circular mode 1: Enable circular mode

		This bit can not be written when CHEN is ‘1’.
4	DIR	Transfer direction Software set and cleared 0: Read from peripheral and write to memory 1: Read from memory and write to peripheral This bit can not be written when CHEN is ‘1’.
3	ERRIE	Enable bit for channel error interrupt Software set and cleared 0: Disable the channel error interrupt 1: Enable the channel error interrupt
2	HTFIE	Enable bit for channel half transfer finish interrupt Software set and cleared 0:Disable channel half transfer finish interrupt 1:Enable channel half transfer finish interrupt
1	FTFIE	Enable bit for channel full transfer finish interrupt Software set and cleared 0:Disable channel full transfer finish interrupt 1:Enable channel full transfer finish interrupt
0	CHEN	Channel enable Software set and cleared 0:Disable channel 1:Enable channel

10.5.4. Channel x counter register (DMA_CHxCNT)

x = 0...6, where x is a channel number

Address offset: 0x0C + 0x14 × x
Reset value: 0x0000 0000



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CNT[15:0]	Transfer counter These bits can not be written when CHEN in the DMA_CHxCTL register is ‘1’. This register indicates how many transfers remain. Once the channel is enabled, it

is read-only, and it decreases after each DMA transfer. If the register is zero, no transaction can be issued whether the channel is enabled or not. Once the transmission of the channel is complete, the register can be reloaded automatically by the previously programmed value if the channel is configured in circular mode.

10.5.5. Channel x peripheral base address register (DMA_CHxPADDR)

x = 0...6, where x is a channel number

Address offset: 0x10 + 0x14 × x

Reset value: 0x0000 0000

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

Bits	Fields	Descriptions
31:0	PADDR[31:0]	Peripheral base address These bits can not be written when CHEN in the DMA_CHxCTL register is ‘1’. When PWIDTH is 01 (16-bit), the LSB of these bits is ignored. Access is automatically aligned to a half word address. When PWIDTH is 10 (32-bit), the two LSBs of these bits are ignored. Access is automatically aligned to a word address.

10.5.6. Channel x memory base address register (DMA_CHxMADDR)

x = 0...6, where x is a channel number

Address offset: 0x14 + 0x14 × x

Reset value: 0x0000 0000

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

Bits	Fields	Descriptions
31:0	MADDR[31:0]	Memory base address These bits can not be written when CHEN in the DMA_CHxCTL register is ‘1’.

When MWIDTH in the DMA_CHxCTL register is 01 (16-bit), the LSB of these bits is ignored. Access is automatically aligned to a half word address.

When MWIDTH in the DMA_CHxCTL register is 10 (32-bit), the two LSBs of these bits are ignored. Access is automatically aligned to a word address.

11. Debug (DBG)

11.1. Overview

The GD32C11x series provide a large variety of debug, trace and test features. They are implemented with a standard configuration of the ARM CoreSight™ module together with a daisy chained standard TAP controller. Debug and trace functions are integrated into the ARM Cortex-M4. The debug system supports serial wire debug (SWD), trace functions and standard JTAG debug. The debug and trace functions refer to the following documents.

- Cortex-M4 Technical Reference Manual.
- ARM Debug Interface v5 Architecture Specification.

The DBG hold unit helps debugger to debug in power saving mode, TIMER, I2C, WWDGT, FWDGT and CAN. When corresponding bit is set, it provides a clock in power saving mode or holds the state for TIMER, I2C, WWDGT, FWDGT or CAN.

11.2. JTAG/SW function overview

Debug capabilities can be accessed by a debug tool via Serial Wire (SW - Debug Port) or JTAG interface (JTAG - Debug Port).

11.2.1. Switch JTAG or SW interface

By default, the JTAG interface is active. The sequence for switching from JTAG to SWD is as following.

- Send 50 or more TCK cycles with TMS = 1.
- Send the 16-bit sequence on TMS = 1110011110011110 (0xE79E LSB first).
- Send 50 or more TCK cycles with TMS = 1.

The sequence for switching from SWD to JTAG is as following.

- Send 50 or more TCK cycles with TMS = 1.
- Send the 16-bit sequence on TMS = 1110011100111100 (0xE73C LSB first).
- Send 50 or more TCK cycles with TMS = 1.

11.2.2. Pin assignment

The JTAG interface provides a 5-pin standard JTAG, known as JTAG clock (JTCK), JTAG mode selection (JTMS), JTAG data input (JTDI), JTAG data output (JTDO) and JTAG reset (NJTRST, active low). The serial wire debug (SWD) provides a 2-pin SW interface, known as SW data input/output (SWDIO) and SW clock (SWCLK). The two SW pins are multiplexed with two of five JTAG pins, which are SWDIO multiplexed with JTMS, SWCLK multiplexed with JTCK. The JTDO is also used as Trace async data output (TRACESWO) when the async trace is enabled.

The pin assignment is as following.

PA15 : JTDI
PA14 : JTCK/SWCLK
PA13 : JTMS/SWDIO
PB4 : NJTRST
PB3 : JTDO

By default, 5-pin standard JTAG debug mode is chosen after reset. User can also use JTAG function without NJTRST pin, then the PB4 can be used to other GPIO functions. (NJTRST tied to 1 by hardware). If it is switched to SW debug mode, the PA15/PB4/PB3 are released to other GPIO functions. If JTAG and SW are not used, all 5-pin can be released to other GPIO functions. Please refer to ***GPIO pin configuration***.

11.2.3. JTAG daisy chained structure

The Cortex-M4 JTAG TAP is connected to a MCU JTAG TAP. The MCU JTAG IR is of 5-bit width, while the Cortex-M4 JTAG IR is of 4-bit width. So when JTAG is in IR shift step, it first shifts 5-bit BYPASS instruction (5'b11111) for MCU JTAG and then shifts normal 4-bit instruction for Cortex-M4 JTAG. Because of the data shift under MCU JTAG BYPASS mode, adding 1 extra bit to the data chain is needed.

The MCU JTAG IDCODE is 0x790007A3.

11.2.4. Debug reset

The JTAG-DP and SW-DP registers are in the Power On Reset domain. The system reset initializes the majority of the Cortex-M4, excluding NVIC and debug logic, (FPB, DWT, and ITM). The NJTRST reset can reset JTAG TAP controller only. So, debug feature can be performed under system reset. Such as halt-after-reset, it is that the debugger sets halt under system reset, and the core halts immediately after the system reset is released.

11.2.5. JEDEC-106 ID code

The Cortex-M4 integrates JEDEC-106 ID code, which is located in ROM table and mapped to the address of 0xE00FF000_0xE00FFFFF.

11.3. Debug hold function overview

11.3.1. Debug support for power saving mode

When the STB_HOLD bit in DBG control register (DBG_CTL) is set, and entering the standby mode, the clock of AHB bus and system clock are provided by CK_IRC8M, and the debugger can debug in standby mode. When exiting the standby mode, a system reset is generated.

When the DSLP_HOLD bit in DBG control register (DBG_CTL) is set, and entering the Deep-sleep mode, the clock of AHB bus and system clock are provided by CK_IRC8M, and the

debugger can debug in Deep-sleep mode.

When the SLP_HOLD bit in DBG control register (DBG_CTL) is set, and entering the sleep mode, the clock of AHB bus for CPU is not closed, and the debugger can debug in sleep mode.

11.3.2. Debug support for TIMER, I2C, WWDGT, FWDGT and CAN

When the core is halted and the corresponding bit in DBG control register (DBG_CTL) is set, the following events occur.

For TIMER, the timer counters are stopped and held for debugging.

For I2C, SMBUS timeout is held for debugging.

For WWDGT or FWDGT, the counter clock is stopped for debugging.

For CAN, the receive register is stopped counting for debugging.

11.4. Register definition

DBG base address: 0xE004 2000

11.4.1. ID code register (DBG_ID)

Address: 0xE004 2000

Read only

This register has to be accessed by word(32-bit)

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

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

Bits	Fields	Descriptions
31:0	ID_CODE[31:0]	DBG ID code register These bits can only be read by software. These bits are unchanged constant.

11.4.2. Control register (DBG_CTL)

Address offset: 0x04

Reset value: 0x0000 0000; power reset only

This register has to be accessed by word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved.	TIMER10_HOLD	TIMER9_HOLD	TIMER8_HOLD	TIMER13_HOLD	TIMER12_HOLD	TIMER11_HOLD	Reserved			CAN1_HOLD	TIMER7_HOLD	TIMER6_HOLD	TIMER5_HOLD	TIMER4_HOLD	I2C1_HOLD
	rw	rw	rw	rw	rw	rw				rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2C0_HOLD	CAN0_HOLD	TIMER3_HOLD	TIMER2_HOLD	TIMER1_HOLD	TIMER0_HOLD	WWDGT_HOLD	FWDGT_HOLD	Reserved		TRACE_IOEN	Reserved		STB_HOLD	DSLP_HOLD	SLP_HOLD
rw	rw	rw	rw	rw	rw	rw	rw			rw			rw	rw	rw

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	TIMER10_HOLD	TIMER 10 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 10 counter for debugging when the core is halted
29	TIMER9_HOLD	TIMER 9 hold bit This bit is set and reset by software.

		0: no effect 1: hold the TIMER 9 counter for debugging when the core is halted
28	TIMER8_HOLD	TIMER 8 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 8 counter for debugging when the core is halted
27	TIMER13_HOLD	TIMER 13 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 13 counter for debugging when the core is halted
26	TIMER12_HOLD	TIMER 12 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 12 counter for debugging when the core is halted
25	TIMER11_HOLD	TIMER 11 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 11 counter for debugging when the core is halted
24:22	Reserved	Must be kept at reset value.
21	CAN1_HOLD	CAN1 hold bit This bit is set and reset by software. 0: no effect 1: the receive register of CAN1 stops receiving data when the core is halted
20	TIMER7_HOLD	TIMER 7 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 7 counter for debugging when the core is halted
19	TIMER6_HOLD	TIMER 6 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 6 counter for debugging when the core is halted
18	TIMER5_HOLD	TIMER 5 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 5 counter for debugging when the core is halted
17	TIMER4_HOLD	TIMER 4 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 4 counter for debugging when the core is halted

16	I2C1_HOLD	I2C1 hold bit This bit is set and reset by software. 0: no effect 1: hold the I2C1 status to avoid SMBUS timeout for debugging when the core is halted
15	I2C0_HOLD	I2C0 hold bit This bit is set and reset by software. 0: no effect 1: hold the I2C0 status to avoid SMBUS timeout for debugging when the core is halted
14	CAN0_HOLD	CAN0 hold bit This bit is set and reset by software. 0: no effect 1: the receive register of CAN0 stops receiving data when the core is halted
13	TIMER3_HOLD	TIMER 3 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 3 counter for debugging when the core is halted
12	TIMER2_HOLD	TIMER 2 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 2 counter for debugging when the core is halted
11	TIMER1_HOLD	TIMER 1 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 1 counter for debugging when the core is halted
10	TIMER0_HOLD	TIMER 0 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 0 counter for debugging when the core is halted
9	WWDGT_HOLD	WWDGT hold bit This bit is set and reset by software. 0: no effect 1: hold the WWDGT counter clock for debugging when the core is halted
8	FWDGT_HOLD	FWDGT hold bit This bit is set and reset by software. 0: no effect 1: hold the FWDGT counter clock for debugging when the core is halted
7:6	Reserved	Must be kept at reset value.

5	TRACE_IOEN	Trace pin allocation enable This bit is set and reset by software. 0: Trace pin allocation disable 1: Trace pin allocation enable
4:3	Reserved	Must be kept at reset value
2	STB_HOLD	Standby mode hold register This bit is set and reset by software. 0: no effect 1: In the standby mode, the clock of AHB bus and system clock are provided by CK_IRC8M, a system reset generated when exiting standby mode.
1	DSLP_HOLD	Deep-sleep mode hold register This bit is set and reset by software. 0: no effect 1: In the Deep-sleep mode, the clock of AHB bus and system clock are provided by CK_IRC8M.
0	SLP_HOLD	Sleep mode hold register This bit is set and reset by software. 0: no effect 1: In the sleep mode, the clock of AHB is on.

12. Analog-to-digital converter (ADC)

12.1. Overview

A 12-bit successive approximation analog-to-digital converter module(ADC) is integrated on the MCU chip, which can sample analog signals from 16 external channels and 2 internal channels. The 18 ADC sampling channels all support a variety of operation modes. After sampling and conversion, the conversion results can be stored in the corresponding data registers according to the least significant bit(LSB) alignment or the most significant(MSB) bit alignment. An on-chip hardware oversample scheme improves performances and reduces the computational burden of MCU.

12.2. Characteristics

- High performance.
  - ADC sampling rsolution: 12-bit, 10-bit, 8-bit or 6-bit.
  - Foreground calibration function.
  - Programmable sampling time.
  - Data storage mode: the most significant bit and the least significant bit.
  - DMA support.
- Analog input channels.
  - 16 external analog inputs.
  - 1 channel for internal temperature sensor (V_{SENSE}).
  - 1 channel for internal reference voltage (V_{REFINT}).
- Start-of-conversion can be initiated.
  - By software.
  - By hardware triggers.
- Operation modes.
  - Convert a single channel or scan a sequence of channels.
  - Single operation mode converts the selected inputs once for per trigger.
  - Continuous operation mode converts selected inputs continuously.
  - Discontinuous operation mode.
  - SYNC mode(the device with two or more ADCs).
- Conversion result threshold monitor function: analog watchdog.
- Interrupt generation.
  - At the end of routine conversions.
  - Analog watchdog event.
- Oversampling.
  - 16-bit data register.
  - Oversampling ratio adjustable from 2x to 256x.
  - Programmable data shift up to 8-bit.
- Channel input range: $V_{REF-} \leq V_{IN} \leq V_{REF+}$.

12.3. Pins and internal signals

Figure 12-1. ADC module block diagram shows the ADC block diagram. *Table 12-1. ADC internal input signals* gives the ADC internal input signals description and *Table 12-2. ADC input pins definition* gives the ADC pin description.

Table 12-1. ADC internal input signals

Internal signal name	Description
V_{SENSE}	Internal temperature sensor output voltage
V_{REFINT}	Internal voltage reference output voltage

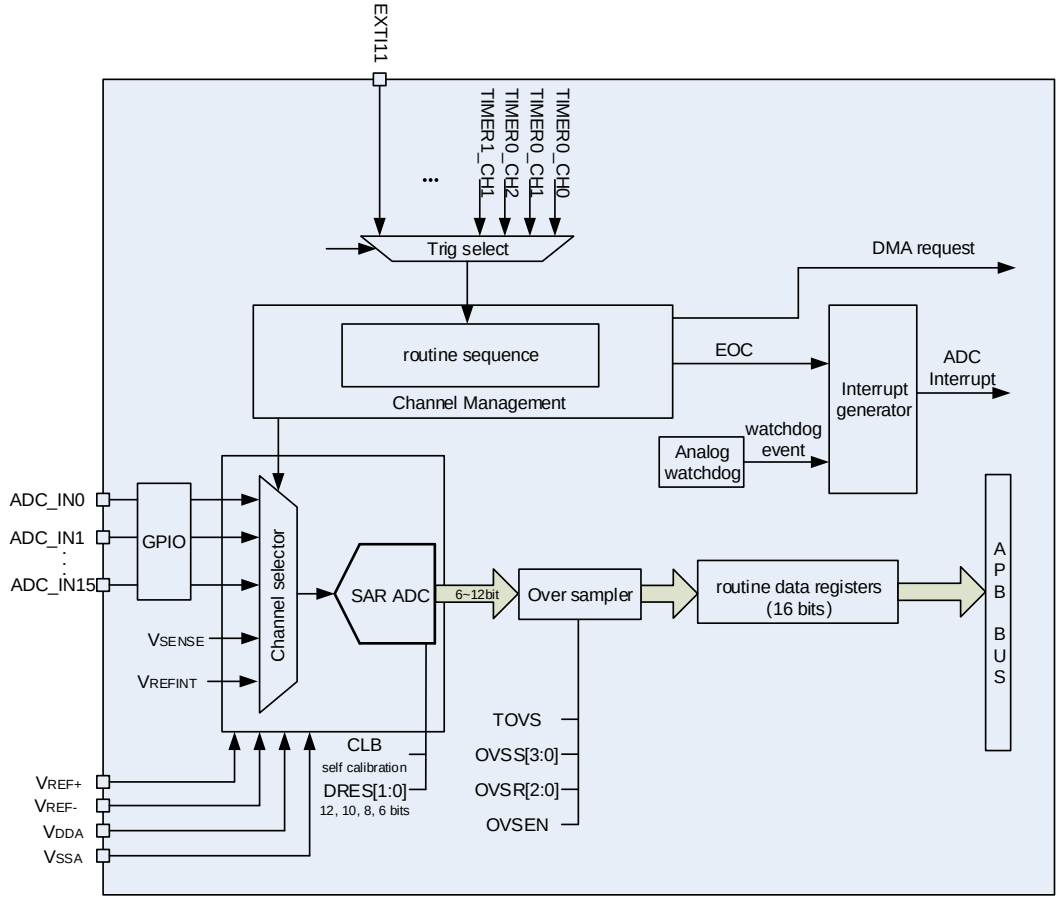
Table 12-2. ADC input pins definition

Name	Description
V_{DDA}	Analog power supply equals to V_{DD}
V_{SSA}	Ground for analog power supply equals to V_{SS}
V_{REF+}	The positive reference voltage for the ADC
V_{REF-}	The negative reference voltage for the ADC, $V_{REF-} = V_{SSA}$
ADCx_IN[15:0]	Up to 16 external channels

Note: V_{DDA} and V_{SSA} have to be connected to V_{DD} and V_{SS} respectively.

12.4. Function overview

Figure 12-1. ADC module block diagram



12.4.1. Foreground calibration function

During the foreground calibration procedure, the ADC calculates a calibration factor which is internally applied to the ADC until the next ADC power-off. The application can not use the ADC until the calibration is completed. The calibration should be performed before starting A/D conversion. The calibration is initiated by setting the CLB bit to 1. The CLB bit stays at 1 during the calibration sequence. Then it is cleared by hardware as soon as the calibration is completed.

When the ADC operating conditions change (such as supply power voltage V_{DDA}, positive reference voltage V_{REF+}, temperature and so on), it is recommended to re-run a calibration cycle.

The internal analog calibration can be reset by setting the RSTCLB bit in ADC_CTL1 register.

Calibration procedure by software:

1. Ensure ADCON=1.
2. Delay 14 CK_ADC to wait for ADC stability.
3. Set RSTCLB (optional).

4. Set CLB=1.
5. Wait for CLB=0.

12.4.2. ADC clock

The CK_ADC clock is synchronous with the AHB and APB2 clock and provided by the clock controller. ADC clock can be divided and configured by RCU controller.

12.4.3. ADCON enable

The ADC module is enabled or disabled by configuring the ADCON bit in the ADC_CTL1 register. The ADC module will keep in reset state if this bit is 0. For power saving, when this bit is 0, the analog sub-module will enter power-down mode. After ADC is enabled, you need delay t_{SU} time for sampling, the value of t_{SU} please refer to the device datasheet.

12.4.4. Routine sequence

The channel management circuit can organize the sampling conversion channels into a sequence: routine sequence. The routine sequence supports up to 16 channels, and each channel is called routine channel.

The RL[3:0] bits in the ADC_RSQ0 register specify the total conversion sequence length. The ADC_RSQ0~ADC_RSQ2 registers specify the selected channels of the routine sequence .

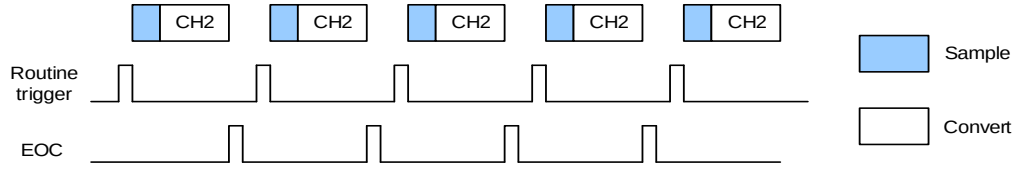
Note: Although the ADC supports 18 multiplexed channels, the maximum length of the sequence is only 16.

12.4.5. Operation modes

Single operation mode

In the single operation mode, the ADC performs conversion on the channel specified in the RSQ0[4:0] bits of ADC_RSQ2 at a routine trigger. When the ADCON has been set high, the ADC samples and converts a single channel, once the corresponding software trigger or external trigger is active.

Figure 12-2. Single operation mode



After conversion of a single routine channel, the conversion data will be stored in the ADC_RDATA register, the EOC will be set. An interrupt will be generated if the EOCIE bit is set.

Software procedure for single operation mode of a routine channel:

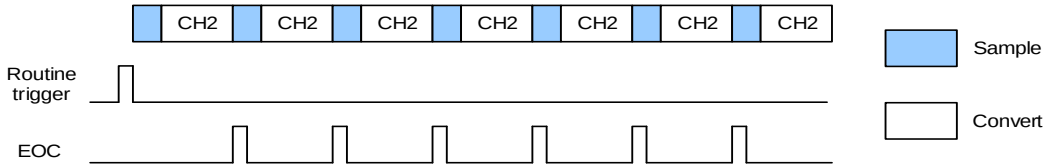
1. Make sure the DISRC, SM in the ADC_CTL0 register and CTN bit in the ADC_CTL1 register are reset.
2. Configure RSQ0 with the analog channel number.
3. Configure ADC_SAMPTx register.
4. Configure ETERC and ETSRC bits in the ADC_CTL1 register if in need.
5. Set the SWRCST bit, or generate an external trigger for the routine sequence.
6. Wait the EOC flag to be set.
7. After a delay of one CK_ADC, read the converted data in the ADC_RDATA register.
8. Clear the EOC flag by writing 0 to it.

Note: After EOC is set, a delay of one CK_ADC is required before reading the ADC conversion result.

Continuous operation mode

The continuous operation mode will be enabled when CTN bit in the ADC_CTL1 register is set. In this mode, the ADC performs conversion on the channel specified in the RSQ0[4:0]. When the ADCON has been set high, the ADC samples and converts specified channel, once the corresponding software trigger or external trigger is active. The conversion data will be stored in the ADC_RDATA register.

Figure 12-3. Continuous operation mode



Software procedure for continuous operation on a routine channel:

1. Set the CTN bit in the ADC_CTL1 register.
2. Configure RSQ0 with the analog channel number.
3. Configure ADC_SAMPTx register.
4. Configure ETERC and ETSRC bits in the ADC_CTL1 register if in need.
5. Set the SWRCST bit, or generate an external trigger for the routine sequence.
6. Wait the EOC flag to be set.
7. After a delay of one CK_ADC, read the converted data in the ADC_RDATA register.
8. Clear the EOC flag by writing 0 to it.
9. Repeat steps 6~8 as soon as the conversion is in need.

Note: After EOC is set, a delay of one CK_ADC is required before reading the ADC conversion result.

To get rid of checking, DMA can be used to transfer the converted data:

1. Set the CTN and DMA bit in the ADC_CTL1 register.
2. Configure RSQ0 with the analog channel number.
3. Configure ADC_SAMPTx register.
4. Configure ETERC and ETSRC bits in the ADC_CTL1 register if in need.
5. Prepare the DMA module to transfer data from the ADC_RDATA.

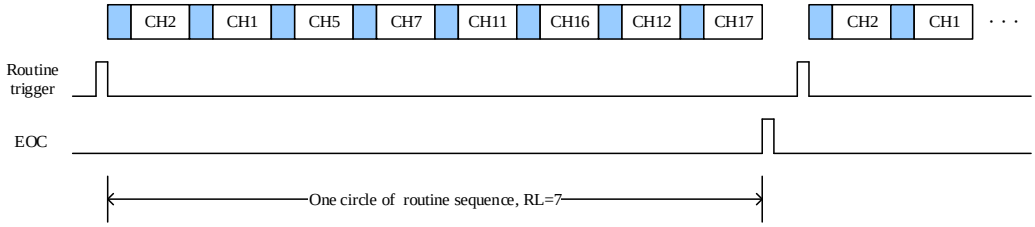
6. Set the SWRCST bit, or generate an external trigger for the routine sequence.

Scan operation mode

The scan operation mode will be enabled when SM bit in the ADC_CTL0 register is set. In this mode, the ADC performs conversion on all channels with a specific routine sequence specified in the ADC_RSQ0~ADC_RSQ2 registers. When the ADCON has been set high, the ADC samples and converts specified channels one by one in the routine sequence till the end of the sequence , once the corresponding software trigger or external trigger is active. The conversion data will be stored in the ADC_RDATA register. After conversion of the routine sequence, the EOC will be set. An interrupt will be generated if the EOCIE bit is set. The DMA bit in ADC_CTL1 register must be set when the routine sequence works in scan mode.

After conversion of a routine sequence, the conversion can be restarted automatically if the CTN bit in the ADC_CTL1 register is set.

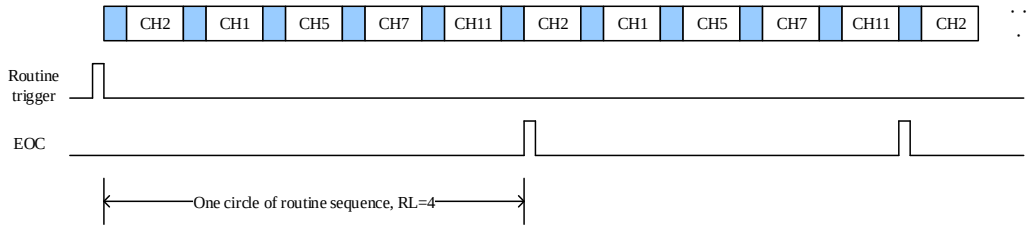
Figure 12-4. Scan operation mode, continuous disable



Software procedure for scan operation mode on a routine sequence:

1. Set the SM bit in the ADC_CTL0 register and the DMA bit in the ADC_CTL1 register.
2. Configure ADC_RSQx and ADC_SAMPTx registers.
3. Configure ETERC and ETSRC bits in the ADC_CTL1 register if in need.
4. Prepare the DMA module to transfer data from the ADC_RDATA.
5. Set the SWRCST bit, or generate an external trigger for the routine sequence.
6. Wait the EOC flag to be set.
7. Clear the EOC flag by writing 0 to it.

Figure 12-5. Scan operation mode, continuous enable

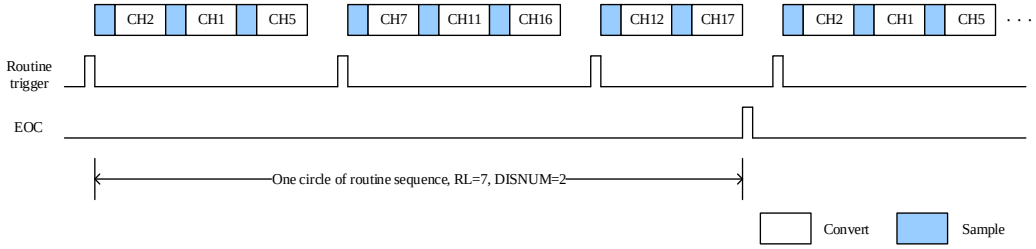


Discontinuous operation mode

The discontinuous operation mode will be enabled when DISRC bit in the ADC_CTL0 register is set. In this mode, the ADC performs a short sequence of n conversions (n does not exceed 8) which is a part of the conversions selected in the ADC_RSQ0~ADC_RSQ2 registers. The value of n is configured by the DISNUM[2:0] bits in the ADC_CTL0 register. When the

corresponding software trigger or external trigger is active, the ADC samples and converts the next n channels configured in the ADC_RSQ0~ADC_RSQ2 registers until all the channels of routine sequence are done. The EOC will be set after every circle of the routine sequence. An interrupt will be generated if the EOCIE bit is set.

Figure 12-6. Discontinuous operation mode



Software procedure for discontinuous operation mode on a routine sequence:

1. Set the DISRC bit in the ADC_CTL0 register and the DMA bit in the ADC_CTL1 register.
2. Configure DISNUM[2:0] bits in the ADC_CTL0 register.
3. Configure ADC_RSQx and ADC_SAMPTx registers.
4. Configure ETERC and ETSRC bits in the ADC_CTL1 register if in need.
5. Prepare the DMA module to transfer data from the ADC_RDATA (refer to the spec of the DMA module).
6. Set the SWRCST bit, or generate an external trigger for the routine sequence.
7. Repeat step6 if in need.
8. Wait the EOC flag to be set.
9. Clear the EOC flag by writing 0 to it.

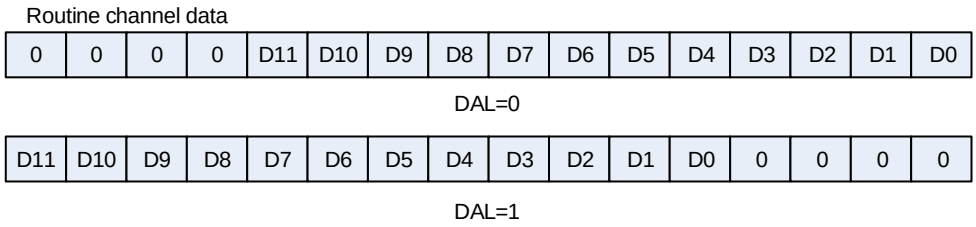
12.4.6. Conversion result threshold monitor function

The analog watchdog is enabled when the RWDEN bit in the ADC_CTL0 register is set for routine sequence. This function is used to monitor whether the conversion result exceeds the set thresholds, and the WDE bit in ADC_STAT register will be set. An interrupt will be generated if the WDEIE bit is set. The ADC_WDHT and ADC_WDLT registers are used to specify the high and low threshold. The comparison is done before the alignment, so the threshold values are independent of the alignment, which is specified by the DAL bit in the ADC_CTL1 register. One or more channels, which are select by the RWDEN, WDSC and WDCHSEL[4:0] bits in ADC_CTL0 register, can be monitored by the analog watchdog.

12.4.7. Data storage mode

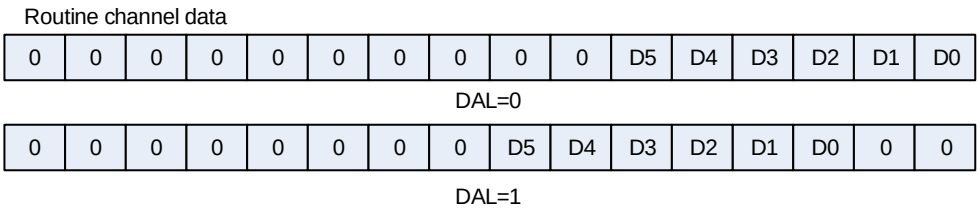
The alignment of data stored after conversion can be specified by DAL bit in the ADC_CTL1 register.

Figure 12-7. 12-bit data storage mode



6-bit resolution data storage mode is different from 12-bit/10-bit/8-bit resolution data storage mode, shown as ***Figure 12-8. 6-bit data storage mode***.

Figure 12-8. 6-bit data storage mode



12.4.8. Sample time configuration

The number of CK_ADC cycles which is used to sample the input voltage can be specified by the SPTn[2:0] bits in the ADC_SAMPT0 and ADC_SAMPT1 registers. Different sampling time can be specified for each channel. For 12-bit resolution, the total sampling and conversion time is “sampling time + 12.5” CK_ADC cycles.

Example:

CK_ADC = 30MHz and sampling time is 1.5 cycles, the total conversion time is “1.5+12.5” CK_ADC cycles, that means 0.467us.

12.4.9. External trigger configuration

The conversion of routine sequence can be triggered by rising edge of external trigger inputs. The external trigger source of routine sequence is controlled by the ETSRC[2:0] bits in the ADC_CTL1 register.

Table 12-3. External trigger source for ADC0 and ADC1

ETSRC[2:0]	Trigger Source	Trigger Type
000	TIMER0_CH0	Hardware trigger
001	TIMER0_CH1	
010	TIMER0_CH2	
011	TIMER1_CH1	
100	TIMER2_TRGO	
101	TIMER3_CH3	
110	EXTI11/ TIMER7_TRGO	

ETSRC[2:0]	Trigger Source	Trigger Type
111	SWRCST	Software trigger

12.4.10. DMA request

The DMA request, which is enabled by the DMA bit of ADC_CTL1 register, is used to transfer data of routine sequence for conversion of more than one channel. The ADC generates a DMA request at the end of conversion of a routine channel. When this request is received, the DMA will transfer the converted data from the ADC_RDATA register to the destination location which is specified by the user.

12.4.11. ADC internal channels

When the TSVREN bit of ADC_CTL1 register is set, the temperature sensor channel (ADC0_IN16) and V_{REFINT} channel (ADC0_IN17) are enabled. The temperature sensor can be used to measure the ambient temperature of the device. The sensor output voltage can be converted into a digital value by ADC. The sampling time for the temperature sensor is recommended to be set to at least t_{s_temp} μs (please refer to the datasheet). When this sensor is not in use, it can be put in power down mode by resetting the TSVREN bit.

The output voltage of the temperature sensor changes linearly with temperature. Because there is an offset, which is up to 45 °C and varies from chip to chip due to the chip production process variation, the internal temperature sensor is more appropriate to detect temperature variations instead of absolute temperature. When it is used to detect accurate temperature, an external temperature sensor part should be used to calibrate the offset error.

The internal voltage reference (V_{REFINT}) provides a stable (bandgap) voltage output for the ADC and Comparators. V_{REFINT} is internally connected to the ADC0_IN17 input channel.

To use the temperature sensor:

1. Configure the conversion sequence (ADC0_IN16) and the sampling time (t_{s_temp} μs) for the channel.
2. Enable the temperature sensor by setting the TSVREN bit in the ADC control register 1 (ADC_CTL1).
3. Start the ADC conversion by setting the ADCON bit or by the triggers.
4. Read the internal temperature sensor output voltage($V_{temperature}$), and get the temperature with the following equation:

$$\text{Temperature (°C)} = \{(V_{25} - V_{temperature}) / \text{Avg_Slope}\} + 25.$$

V_{25}: internal temperature sensor output voltage at 25°C, the typical value please refer to the datasheet.

Avg_Slope: average slope for curve between temperature vs. internal temperature sensor output voltage, the typical value please refer to the datasheet.

12.4.12. Programmable resolution (DRES)

The resolution is configured by programming the DRES[1:0] bits in the ADC_OVSAMPCTL register. For applications that do not require high data accuracy, lower resolution allows faster conversion time. The DRES[1:0] bits must only be changed when the ADCON bit is reset. Lower resolution reduces the conversion time needed for the successive approximation steps as shown in ***Table 12-4. tCONV timings depending on resolution***.

Table 12-4. t_{CONV} timings depending on resolution

DRES[1:0] bits	t_{CONV} (ADC clock cycles)	t_{CONV}(ns) at f_{ADC}=30MHz	t_{SMPL}(min) (ADC clock cycles)	t_{ADC} (ADC clock cycles)	t_{ADC}(ns) at f_{ADC}=30MHz
12	12.5	417 ns	1.5	14	467 ns
10	10.5	350 ns	1.5	12	400 ns
8	8.5	283 ns	1.5	10	333 ns
6	6.5	217 ns	1.5	8	267 ns

12.4.13. On-chip hardware oversampling

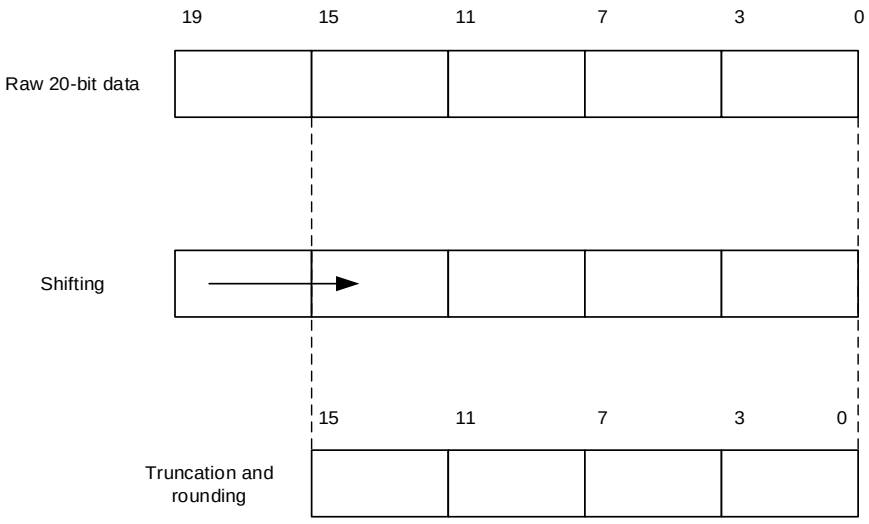
The on-chip hardware oversampling circuit performs data preprocessing to offload the CPU. It can handle multiple conversions and average them into a single data with increased data width up to 16-bit. It provides a result with the following form, where N and M can be adjusted, and $D_{out}(n)$ is the n-th output digital signal of the ADC:

$$Result=\frac{1}{M}*\sum_{n=0}^{N-1} D_{out}(n) \quad (12\text{-}1)$$

The on-chip hardware oversampling circuit performs the following functions: summing and bit right shifting. The oversampling ratio N is defined by the OVSR[2:0] bits in the ADC_OVSAMPCTL register. It can range from 2x to 256x. The division coefficient M means bit right shifting up to 8 bits. It is configured through the OVSS[3:0] bits in the ADC_OVSAMPCTL register.

Summation units can produce up to 20 bits (256 x 12-bit), which is first shifted right. The upper bits of the result are then truncated, keeping only the 16 least significant bits rounded to the nearest value using the least significant bits left apart by the shifting, before being finally transferred into the data register.

Figure 12-9. 20-bit to 16-bit result truncation



Note: If the intermediate result after the shifting exceeds 16 bits, the upper bits of the result are simply truncated.

Figure 12-10. A numerical example with 5-bit shifting and rounding shows a numerical example of the processing, from a raw 20-bit accumulated data to the final 16-bit result.

Figure 12-10. A numerical example with 5-bit shifting and rounding

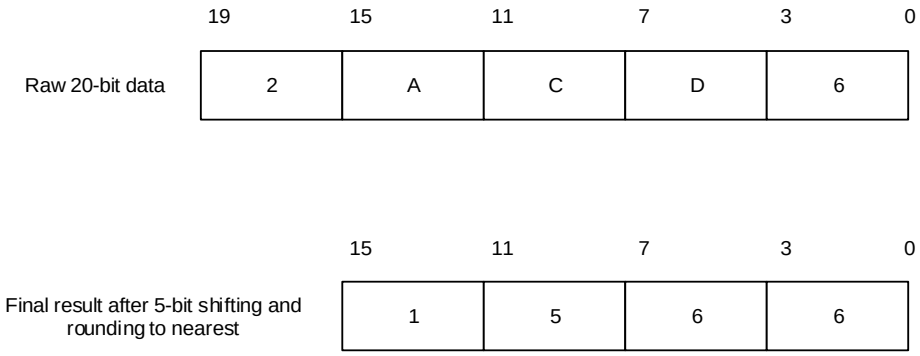


Table 12-5. Maximum output results for N and M combinations (grayed values indicate truncation) below gives the data format for the various N and M combinations, and the raw conversion data equals 0xFFF.

Table 12-5. Maximum output results for N and M combinations (grayed values indicate truncation)

Oversampling ratio	Max Raw data	No-shift OVSS=0000	1-bit shift OVSS=0001	2-bit shift OVSS=0010	3-bit shift OVSS=0011	4-bit shift OVSS=0100	5-bit shift OVSS=0101	6-bit shift OVSS=0110	7-bit shift OVSS=0111	8-bit shift OVSS=1000
2x	0x1FFE	0x1FFE	0x0FFF	0x07FF	0x03FF	0x01FF	0x00FF	0x007F	0x003F	0x001F

4x	0x3FFC	0x3FFC	0x1FFE	0x0FFF	0x07FF	0x03FF	0x01FF	0x00FF	0x007F	0x003F
8x	0x7FF8	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x07FF	0x03FF	0x01FF	0x00FF	0x007F
16x	0xFFF0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x07FF	0x03FF	0x01FF	0x00FF
32x	0x1FFE0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x07FF	0x03FF	0x01FF
64x	0x3FFC0	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x07FF	0x03FF
128x	0x7FF80	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x07FF
256x	0xFFF00	0xFF00	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF

When compared to standard conversion mode, the conversion timings of oversampling mode do not change, and the sampling time is maintained the same as that of standard conversion mode during the whole oversampling sequence. New data is supplied every N conversions, and the equivalent delay is equal to:

$$N \times t_{ADC} = N \times (t_{SMPL} + t_{CONV}) \quad (12\text{-}2)$$

12.5. ADC sync mode

In devices with more than one ADC, the ADC sync mode can be used. In ADC sync mode, the conversion starts alternately or simultaneously triggered by ADC0 to ADC1, according to the sync mode configurated by the SYNCM[3:0] bits in ADC1_CTL0 register.

In sync mode, when configure the conversion which is triggered by an external event, the ADC1 must be configured as triggered by the software. However, the external trigger must be enabled for ADC0 and ADC1.

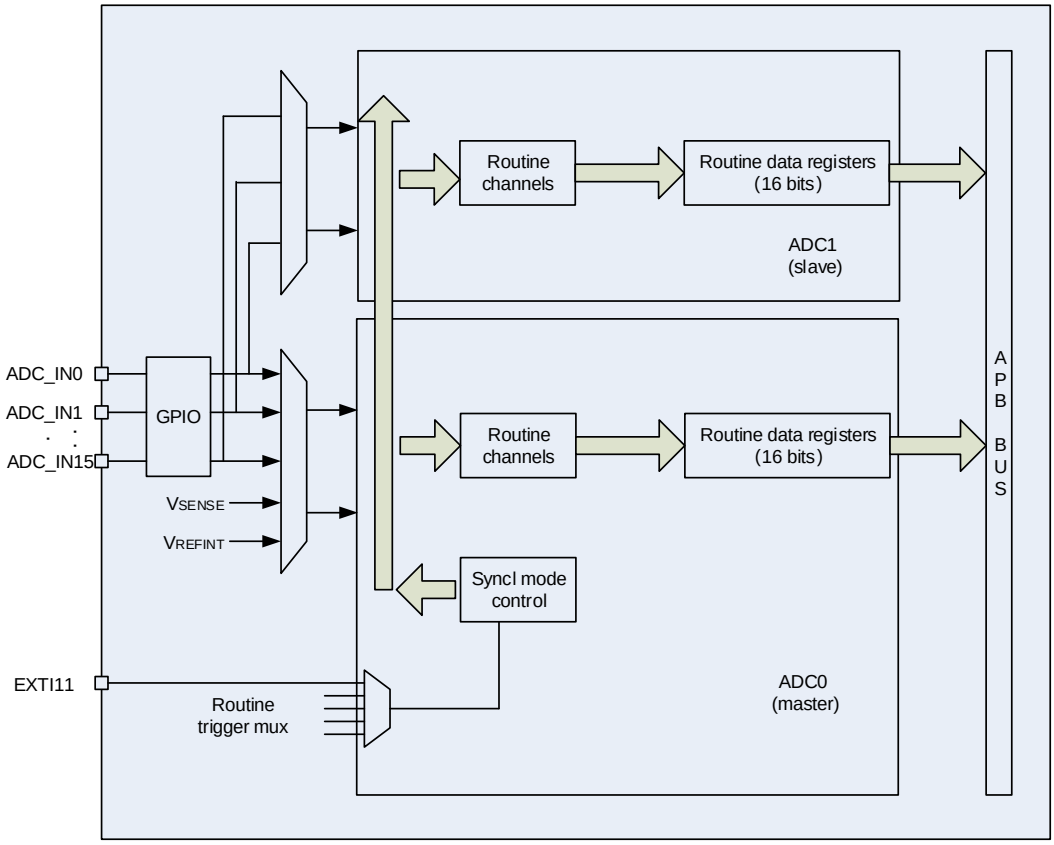
The following modes can be configured in ***Table 12-6. ADC sync mode table***.

Table 12-6. ADC sync mode table

SYNCM[3: 0]	mode
0000	Free mode
0110	Routine parallel mode
0111	Routine follow-up fast mode
1000	Routine follow-up slow mode

In ADC sync mode, the DMA bit must be set even if it is not used; the converted data of ADC1 routine channel can be read from the ADC0 data register.

Figure 12-11. ADC sync block diagram



12.5.1. Free mode

In this mode, each ADC works independently and does not interfere with each other.

12.5.2. Routine parallel mode

This mode converts the routine sequence simultaneously. The source of external trigger comes from the ADC0 routine sequence (configured by the ETSRC[2:0] bits in the ADC_CTL1 register) , and ADC1 routine sequence is configured as software trigger mode.

At the end of conversion event on ADC0 or ADC1, an EOC interrupt is generated (if enabled on one of the two ADC interrupt) when the ADC0/ADC1 routine channels are all converted. The behavior of routine parallel mode shows in the ***Figure 12-12. Routine parallel mode on 10 channels***.

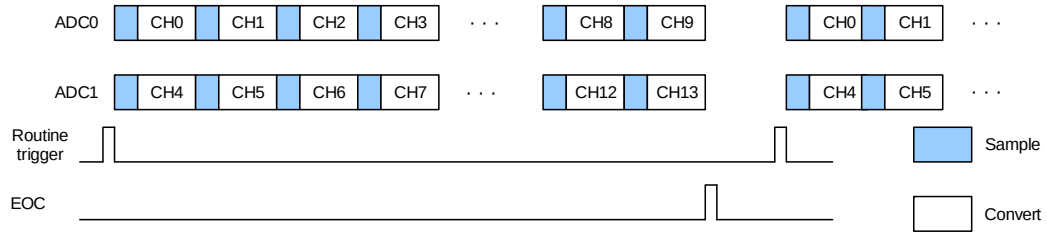
A 32-bit DMA is used, which transfers ADC_RDATA 32-bit register (the ADC_RDATA 32-bit register containing the ADC1 converted data in the [31: 16] bits field and the ADC0 converted data in the [15: 0] bits field) to SRAM.

Note:

1. If two ADCs use the same sampling channel, it should be ensured that the channel is not used at the same time.

2. Two channels sampled by two ADCs at the same time should be configured with the same sampling time.

Figure 12-12. Routine parallel mode on 10 channels



12.5.3. Routine follow-up fast mode

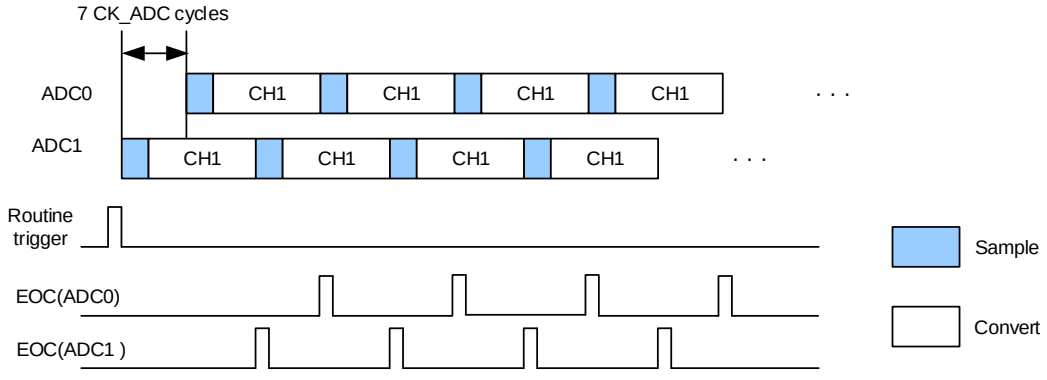
The routine follow-up fast mode is applicable to sample the same channel of two ADCs. The source of external trigger comes from the ADC0 routine channel (selected by the ETSRC[2:0] bits in the ADC_CTL1 register). When the trigger occurs, ADC1 runs immediately and ADC0 runs after 7 ADC clock cycles.

If the continuous mode is enabled for both ADC0 and ADC1, the selected routine channels of two ADCs are continuously converted. The behavior of follow-up fast mode shows in the ***Figure 12-13. Routine follow-up fast mode (the CTN bit of ADCs are set).***

After an EOC interrupt is generated by ADC0 in case of setting the EOCIE bit, we can use a 32-bit DMA, which transfers to SRAM the ADC_RDATA register containing the ADC1 converted data in the [31: 16] bits field and the ADC0 converted data in the [15: 0] bits field.

Note: The sampling time of the routine channel of the two ADCs should be less than 7 ADC clock cycles.

Figure 12-13. Routine follow-up fast mode (the CTN bit of ADCs are set)



12.5.4. Routine follow-up slow mode

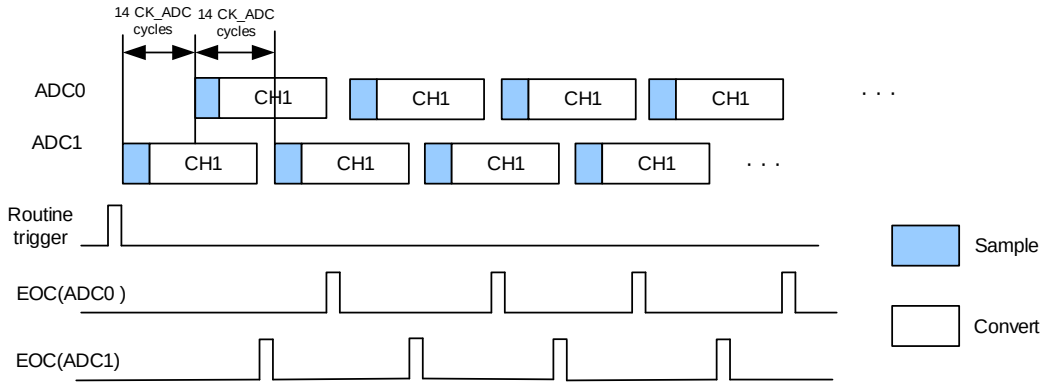
The routine follow-up slow mode is applicable to sample the same channel of two ADCs. The source of external trigger comes from the ADC0 routine channel (selected by the ETSRC[2:0] bits in the ADC_CTL1 register). When the trigger occurs, ADC1 runs immediately, ADC0 runs after 14 ADC clock cycles, after the second 14 ADC clock cycles the ADC1 runs again.

Continuous mode can't be used in this mode, because it continuously converts the routine channel. The behavior of follow-up slow mode shows in the ***Figure 12-14. Routine follow-up slow mode***.

After an EOC interrupt is generated by ADC0 (if EOCIE bit is set), we can use a 32-bit DMA, which transfers to SRAM the ADC_RDATA register containing the ADC1 converted data in the [31: 16] bits field and the ADC0 converted data in the [15: 0] bits field.

Note: The maximum sampling time allowed is < 14 CK_ADC cycles to avoid the overlap between ADC0 and ADC1 sampling phases in the event that they convert the same channel.

Figure 12-14. Routine follow-up slow mode



12.6. ADC interrupts

The interrupt can be generated on one of the events:

- End of conversion for routine sequence.
- The analog watchdog event.

The interrupts of ADC0 and ADC1 are mapped into the same interrupt vector IRQ18.

12.7. Register definition

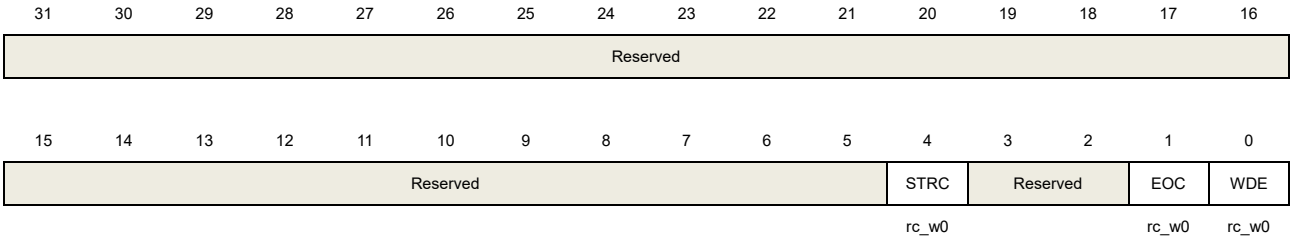
ADC0 base address: 0x4001 2400

ADC1 base address: 0x4001 2800

12.7.1. Status register (ADC_STAT)

Address offset: 0x00
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit)



Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	STRC	Start flag of routine sequence conversion 0: Conversion is not started 1: Conversion is started Set by hardware when routine sequence conversion starts. Cleared by software writing 0 to it.
3:2	Reserved	Must be kept at reset value.
1	EOC	End flag of routine sequence conversion 0: No end of routine sequence conversion 1: End ofroutine sequence conversion Set by hardware at the end of a routine sequence conversion. Cleared by software writing 0 to it or by reading the ADC_RDATA register.
0	WDE	Analog watchdog event flag 0: Analog watchdog event is not happened 1: Analog watchdog event is happening Set by hardware when the converted voltage crosses the values programmed in the ADC_WDLT and ADC_WDHT registers. Cleared by software writing 0 to it.

12.7.2. Control register 0 (ADC_CTL0)

Address offset: 0x04
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								RWDEN	Reserved			SYNCM[3:0]			
								rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DISNUM[2:0]			Reserved	DISRC	Reserved	WDSC	SM	Reserved	WDEIE	EOCIE	WDCHSEL[4:0]				
rw			rw	rw	rw	rw	rw	rw	rw	rw	rw				

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23	RWDEN	Routine channel analog watchdog enable 0: Analog watchdog disable 1: Analog watchdog enable
22:20	Reserved	Must be kept at reset value.
19:16	SYNCM[3:0]	Sync mode selection These bits use to select the operating mode. 0000: Free mode. 0001~0101: Reserved 0110: Routine parallel mode 0111: Routine follow-up fast mode 1000: Routine follow-up slow mode 1001~1111: Reserved **Note**: 1) These bits are only used in ADC0. 2) Users must disable sync mode before any configuration change.
15:13	DISNUM[2:0]	Number of conversions in discontinuous mode The number of channels to be converted after a trigger will be DISNUM+1 in routine sequence.
12	Reserved	Must be kept at reset value.
11	DISRC	Discontinuous mode on routine sequence 0: Discontinuous operation mode disable 1: Discontinuous operation mode enable
10	Reserved	Must be kept at reset value.
9	WDSC	When in scan mode, analog watchdog is effective on a single channel 0: All channels have analog watchdog function 1: A single channel has analog watchdog function
8	SM	Scan mode 0: Scan operation mode disable 1: Scan operation mode enable
7	Reserved	Must be kept at reset value.

6	WDEIE	Interrupt enable for WDE 0: Interrupt disable 1: Interrupt enable
5	EOCIE	Interrupt enable for EOC 0: Interrupt disable 1: Interrupt enable
4:0	WDCHSEL[4:0]	Analog watchdog channel select 00000: ADC channel0 00001: ADC channel1 00010: ADC channel2 00011: ADC channel 3 00100: ADC channel 4 00101: ADC channel 5 00110: ADC channel 6 00111: ADC channel 7 01000: ADC channel 8 01001: ADC channel 9 01010: ADC channel 10 01011: ADC channel 11 01100: ADC channel 12 01101: ADC channel 13 01110: ADC channel 14 01111: ADC channel15 10000: ADC channel16 10001: ADC channel17 Other values are reserved. **Note**: ADC0 analog inputs Channel16 and Channel17 are internally connected to the temperature sensor, and to V_{REFINT} inputs. ADC1 analog inputs Channel16, and Channel17 are internally connected to V_{SSA}.

12.7.3. Control register 1 (ADC_CTL1)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TSVREN	SWRCST	Reserved	ETERC	ETSRC[2:0]			Reserved
								rw	rw		rw		rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				DAL	Reserved.		DMA	Reserved				RSTCLB	CLB	CTN	ADCON
				rw			rw					rw	rw	rw	rw

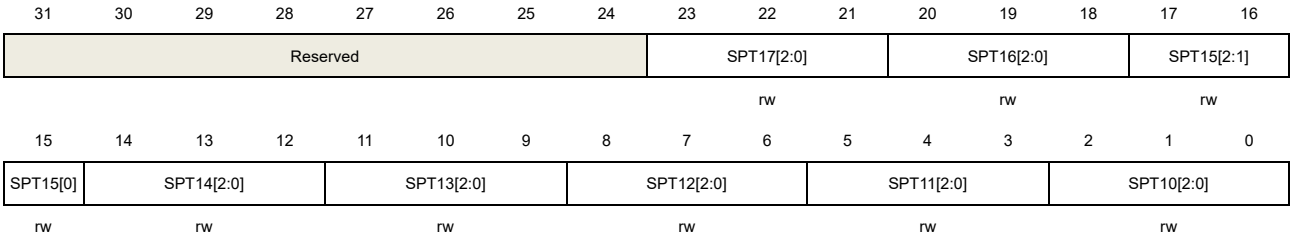
Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23	TSVREN	Channel 16 and 17 enable of ADC0. 0: Channel 16 and 17 of ADC0 disable 1: Channel 16 and 17 of ADC0 enable
22	SWRCST	Software start conversion of routine sequence Set 1 on this bit starts a conversion of a routine sequence if ETSRC is 111. It is set by software and cleared by software or by hardware immediately after the conversion starts.
21	Reserved	Must be kept at reset value.
20	ETERC	External trigger enable for routine sequence 0: External trigger for routine sequence disable 1: External trigger for routine sequence enable
19:17	ETSRC[2:0]	External trigger select for routine sequence For ADC0 and ADC1: 000: Timer 0 CH0 001: Timer 0 CH1 010: Timer 0 CH2 011: Timer 1 CH1 100: Timer 2 TRGO 101: Timer 3 CH3 110: EXTI line 11/ Timer 7 TRGO 111: SWRCST
16:12	Reserved	Must be kept at reset value
11	DAL	Data alignment 0: LSB alignment 1: MSB alignment
10:9	Reserved	Must be kept at reset value.
8	DMA	DMA request enable. 0: DMA request disable 1: DMA request enable
7:4	Reserved	Must be kept at reset value.
3	RSTCLB	Reset calibration This bit is set by software and cleared by hardware after the calibration registers are initialized. 0: Calibration register initialize done. 1: Initialize calibration register start
2	CLB	ADC calibration

		0: Calibration done 1: Calibration start
1	CTN	Continuous mode 0: Continuous operation mode disable 1: Continuous operation mode enable
0	ADCON	ADC ON. The ADC will be wake up when this bit is changed from low to high and take a stabilization time. When this bit is high and “1” is written to it with other bits of this register unchanged, the conversion will start. 0: ADC disable and power down 1: ADC enable

12.7.4. Sample time register 0 (ADC_SAMPT0)

Address offset: 0x0C
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit)



Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:21	SPT17[2:0]	Refer to SPT10[2:0] description
20:18	SPT16[2:0]	Refer to SPT10[2:0] description
17:15	SPT15[2:0]	Refer to SPT10[2:0] description
14:12	SPT14[2:0]	Refer to SPT10[2:0] description
11:9	SPT13[2:0]	Refer to SPT10[2:0] description
8:6	SPT12[2:0]	Refer to SPT10[2:0] description
5:3	SPT11[2:0]	Refer to SPT10[2:0] description
2:0	SPT10[2:0]	Channel sampling time 000: channel sampling time is 1.5 cycles 001: channel sampling time is 7.5 cycles 010: channel sampling time is 13.5 cycles 011: channel sampling time is 28.5 cycles 100: channel sampling time is 41.5 cycles

101: channel sampling time is 55.5 cycles
110: channel sampling time is 71.5 cycles
111: channel sampling time is 239.5 cycles

12.7.5. Sample time register 1 (ADC_SAMPT1)

Address offset: 0x10
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		SPT9[2:0]			SPT8[2:0]			SPT7[2:0]			SPT6[2:0]			SPT5[2:1]	
		rw			rw			rw			rw			rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPT5[0]		SPT4[2:0]			SPT3[2:0]			SPT2[2:0]			SPT1[2:0]			SPT0[2:0]	
rw		rw			rw			rw			rw			rw	

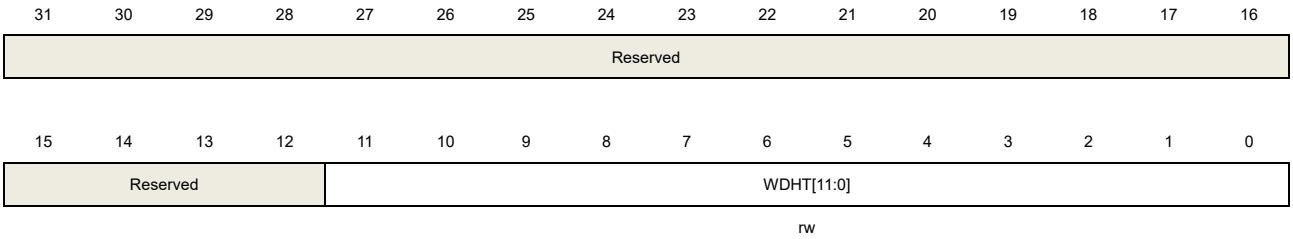
Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29:27	SPT9[2:0]	Refer to SPT0[2:0] description
26:24	SPT8[2:0]	Refer to SPT0[2:0] description
23:21	SPT7[2:0]	Refer to SPT0[2:0] description
20:18	SPT6[2:0]	Refer to SPT0[2:0] description
17:15	SPT5[2:0]	Refer to SPT0[2:0] description
14:12	SPT4[2:0]	Refer to SPT0[2:0] description
11:9	SPT3[2:0]	Refer to SPT0[2:0] description
8:6	SPT2[2:0]	Refer to SPT0[2:0] description
5:3	SPT1[2:0]	Refer to SPT0[2:0] description
2:0	SPT0[2:0]	Channel sampling time 000: channel sampling time is 1.5 cycles 001: channel sampling time is 7.5 cycles 010: channel sampling time is 13.5 cycles 011: channel sampling time is 28.5 cycles 100: channel sampling time is 41.5 cycles 101: channel sampling time is 55.5 cycles 110: channel sampling time is 71.5 cycles 111: channel sampling time is 239.5 cycles

12.7.6. Watchdog high threshold register (ADC_WDHT)

Address offset: 0x24

Reset value: 0x0000 0FFF

This register has to be accessed by word(32-bit)



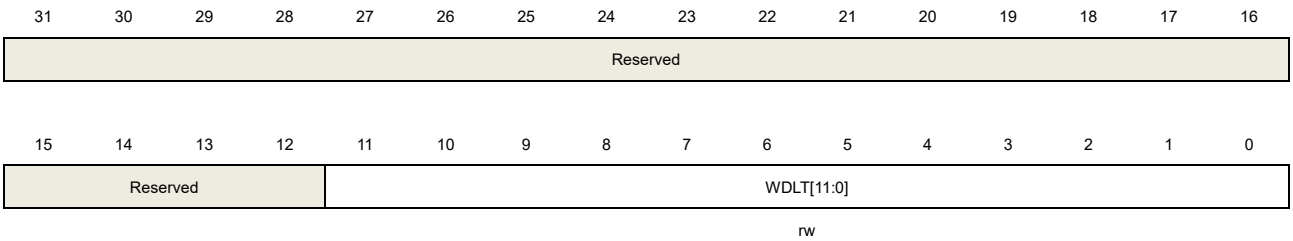
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	WDHT[11:0]	High threshold for analog watchdog These bits define the high threshold for the analog watchdog.

12.7.7. Watchdog low threshold register (ADC_WDLT)

Address offset: 0x28

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit)



Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	WDLT[11:0]	Low threshold for analog watchdog These bits define the low threshold for the analog watchdog.

12.7.8. Routine sequence register 0 (ADC_RSQ0)

Address offset: 0x2C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit)

Reserved	RL[3:0]	RSQ15[4:1]
	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSQ15[0]	RSQ14[4:0]							RSQ13[4:0]				RSQ12[4:0]			
rw	rw							rw				rw			

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:20	RL[3:0]	Routine sequence length. The total number of conversion in routine sequence equals to RL[3:0]+1.
19:15	RSQ15[4:0]	Refer to RSQ0[4:0] description
14:10	RSQ14[4:0]	Refer to RSQ0[4:0] description
9:5	RSQ13[4:0]	Refer to RSQ0[4:0] description
4:0	RSQ12[4:0]	Refer to RSQ0[4:0] description

12.7.9. Routine sequence register 1 (ADC_RSQ1)

Address offset: 0x30

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		RSQ11[4:0]					RSQ10[4:0]					RSQ9[4:1]			
		rw					rw					rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSQ9[0]	RSQ8[4:0]							RSQ7[4:0]				RSQ6[4:0]			
rw	rw							rw				rw			

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29:25	RSQ11[4:0]	Refer to RSQ0[4:0] description
24:20	RSQ10[4:0]	Refer to RSQ0[4:0] description
19:15	RSQ9[4:0]	Refer to RSQ0[4:0] description
14:10	RSQ8[4:0]	Refer to RSQ0[4:0] description
9:5	RSQ7[4:0]	Refer to RSQ0[4:0] description
4:0	RSQ6[4:0]	Refer to RSQ0[4:0] description

12.7.10. Routine sequence register 2 (ADC_RSQ2)

Address offset: 0x34

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		RSQ5[4:0]					RSQ4[4:0]					RSQ3[4:1]			
		rw					rw					rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSQ3[0]		RSQ2[4:0]					RSQ1[4:0]					RSQ0[4:0]			
rw		rw					rw					rw			

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29:25	RSQ5[4:0]	Refer to RSQ0[4:0] description
24:20	RSQ4[4:0]	Refer to RSQ0[4:0] description
19:15	RSQ3[4:0]	Refer to RSQ0[4:0] description
14:10	RSQ2[4:0]	Refer to RSQ0[4:0] description
9:5	RSQ1[4:0]	Refer to RSQ0[4:0] description
4:0	RSQ0[4:0]	The channel number (0..17) is written to these bits to select a channel as the nth conversion in the routine sequence.

12.7.11. Routine data register (ADC_RDATA)

Address offset: 0x4C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADC1RDTR[15:0]															
r															

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

Bits	Fields	Descriptions
31:16	ADC1RDTR[15:0]	ADC1 routine channel data In sync mode, these bits contain the routine data of ADC1. These bits are only used in ADC0.

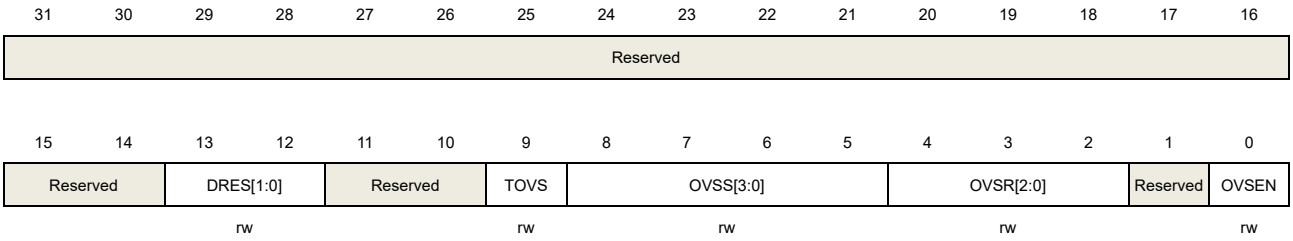
15:0	RDATA[15:0]	Routine channel data These bits contain routine channel conversion value, which is read only.

12.7.12. Oversample control register (ADC_OVSAMPCTL)

Address offset: 0x80

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit)



Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13:12	DRES[1:0]	ADC resolution 00: 12 bits 01: 10 bits 10: 8 bits 11: 6 bits
11:10	Reserved	Must be kept at reset value.
9	TOVS	Triggered Oversampling This bit is set and cleared by software. 0: All oversampled conversions for a channel are done consecutively after a trigger 1: Each conversion needs a trigger for a oversampled channel and the number of triggers is determined by the oversampling ratio(OVSR[2:0]). **Note:** The software allows this bit to be written only when ADCON = 0 (this ensures that no conversion is in progress).
8:5	OVSS[3:0]	Oversampling shift These bits are set and cleared by software. 0000: No shift 0001: Shift 1 bit 0010: Shift 2 bits 0011: Shift 3 bits 0100: Shift 4 bits 0101: Shift 5 bits 0110: Shift 6 bits 0111: Shift 7 bits 1000: Shift 8 bits

		Other: Reserved **Note:** The software allows this bit to be written only when ADCON = 0 (this ensures that no conversion is in progress).
4:2	OVSR[2:0]	Oversampling ratio This bit filed defines the number of oversampling ratio. 000: 2x 001: 4x 010: 8x 011: 16x 100: 32x 101: 64x 110: 128x 111: 256x **Note:** The software allows this bit to be written only when ADCON = 0 (this ensures that no conversion is in progress).
1	Reserved	Must be kept at reset value.
0	OVSEN	Oversampler enable This bit is set and cleared by software. 0: Oversampler disabled 1: Oversampler enabled **Note:** The software allows this bit to be written only when ADCON = 0 (this ensures that no conversion is in progress).

13. Digital-to-analog converter (DAC)

13.1. Overview

The Digital-to-analog converter converts 12-bit digital data to a voltage on the external pins. The digital data can be configured to 8-bit or 12-bit mode, left-aligned or right-aligned mode. DMA can be used to update the digital data on external triggers.

The output voltage can be optionally buffered for higher drive capability.

The DAC channels can work independently or concurrently.

13.2. Characteristics

The main features of DAC are as follows:

- 8-bit or 12-bit resolution.
- Left or right data alignment.
- DMA capability for each channel.
- Conversion update synchronously.
- Conversion triggered by external triggers.
- Configurable internal buffer.
- Extern voltage reference, V_{REFP}.
- Noise wave generation (LFSR noise mode and triangle noise mode).
- Two DAC channels in concurrent mode.

Figure 13-1. DAC block diagram and *Table 13-1. DAC I/O description* show the block diagram of DAC and the pin description of DAC, respectively.

Figure 13-1. DAC block diagram

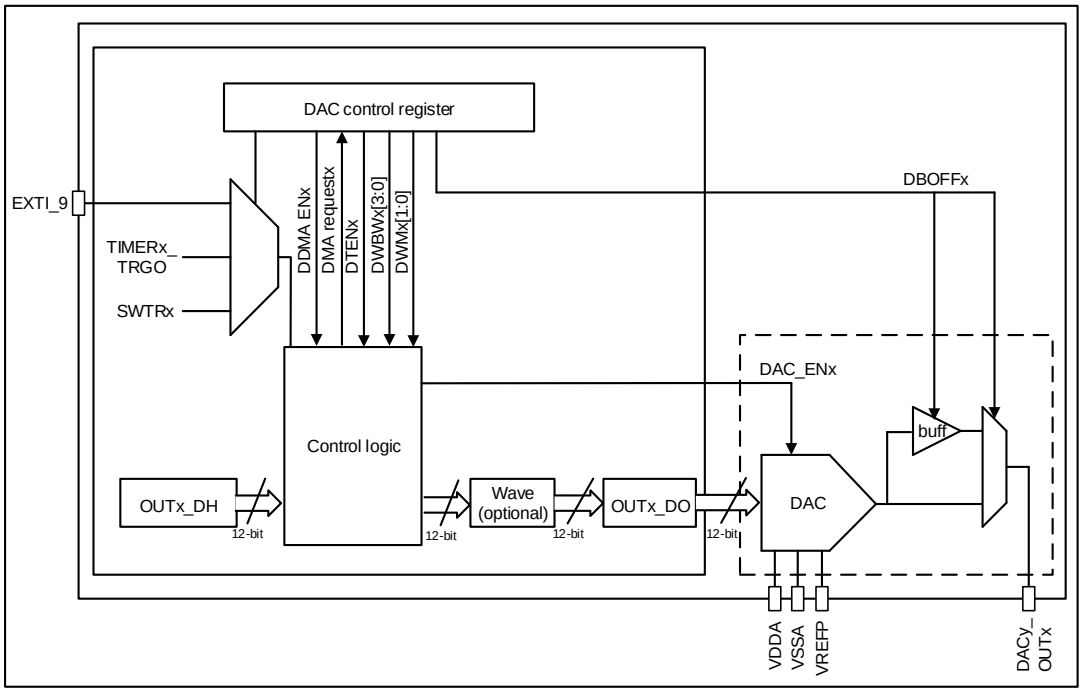


Table 13-1. DAC I/O description

Name	Description	Signal type
V_{DDA}	Analog power supply	Input, analog supply
V_{SSA}	Ground for analog power supply	Input, analog supply ground
V_{REFP}	Positive reference voltage of DAC	Input, analog positive reference
DACy_OUTx	DAC analog output	Analog output signal

The below table details the triggers and outputs of the DAC.

Table 13-2. DAC triggers and outputs summary

	DAC0	
Channel	**Channel0**	**Channel1**
DAC outputs connected to I / Os	PA4	PA5
DAC output buffer	•	•
DAC software trigger	•	
DAC trigger signals from EXTI	EXTI_9	
DAC trigger signals from TIMER	TIMER1_TRGO TIMER2_TRGO TIMER3_TRGO TIMER4_TRGO TIMER5_TRGO TIMER6_TRGO	

Note: The GPIO pins should be configured to analog mode before enable the DAC module.

13.3. Function overview

13.3.1. DAC enable

The DAC can be turned on by setting the DENx bit in the DAC_CTL0 register. A t_{WAKEUP} time is needed to startup the analog DAC submodule.

13.3.2. DAC output buffer

For reducing output impedance and driving external loads without an external operational amplifier, an output buffer is integrated inside each DAC module.

The output buffer, which is turned on by default to reduce the output impedance and improve the driving capability, can be turned off by setting the DBOFFx bit in the DAC_CTL0 register.

13.3.3. DAC data configuration

The 12-bit DAC holding data (OUTx_DH) can be configured by writing any one of the DAC_OUTx_R12DH, DAC_OUTx_L12DH and DAC_OUTx_R8DH registers. When the data is loaded by DAC_OUTx_R8DH register, only the MSB 8 bits are configurable, the LSB 4 bits are forced to 4'b0000.

13.3.4. DAC trigger

The DAC conversion can be triggered by software or rising edge of external trigger source. The DAC external trigger is enabled by setting the DTENx bits in the DAC_CTL0 register. The DAC external triggers are selected by the DTSELx bits in the DAC_CTL0 register, which is shown as ***Table 13-3. Triggers of DAC***.

Table 13-3. Triggers of DAC

DTSELx[2:0]	Trigger Source	Trigger Type
3b'000	TIMER5_TRGO	Hardware trigger
3b'001	TIMER2_TRGO	
3b'010	TIMER6_TRGO	
3b'011	TIMER4_TRGO	
3b'100	TIMER1_TRGO	
3b'101	TIMER3_TRGO	
3b'110	EXTI_9	
3b'111	SWTR	Software trigger

The TIMERx_TRGO signals are generated from the timers, while the software trigger can be generated by setting the SWTRx bits in the DAC_SWT register.

13.3.5. DAC conversion

If the external trigger is enabled by setting the DTENx bit in DAC_CTL0 register, the DAC holding data is transferred to the DAC output data (DAC_OUTx_DO) register when the selected trigger event happened. When the external trigger is disabled, the transfer is performed automatically.

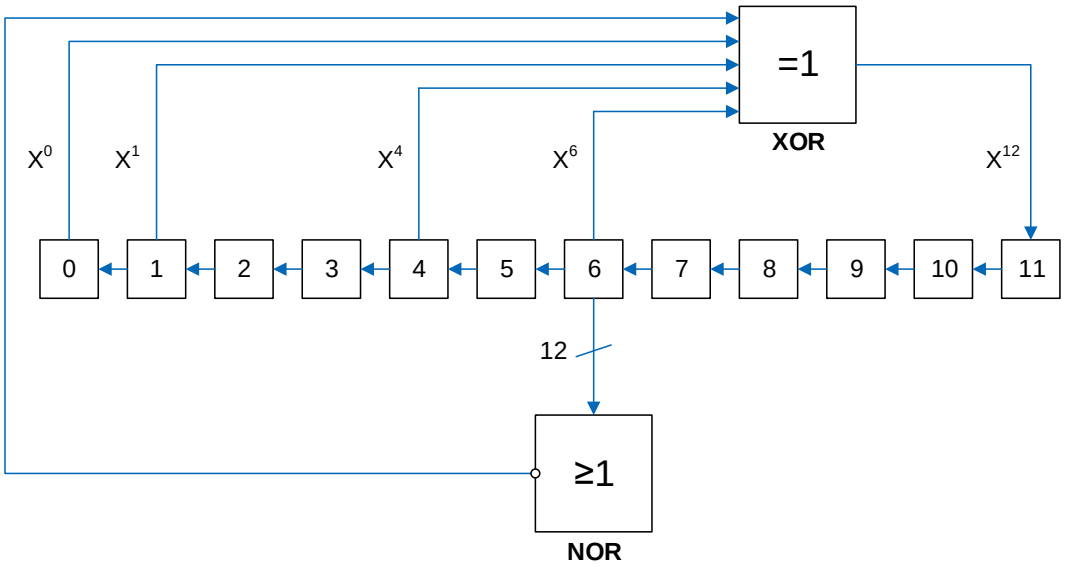
When the DAC holding data (OUTx_DH) is loaded into the DAC_OUTx_DO register, after the time $t_{SETTLING}$ which is determined by the analog output load and the power supply voltage, the analog output is valid.

13.3.6. DAC noise wave

There are two methods of adding noise wave to the DAC output data: LFSR noise wave mode and Triangle wave mode. The noise wave mode can be selected by the DWMx bits in the DAC_CTL0 register. The amplitude of the noise can be configured by the DAC noise wave bit width (DWBWx) bits in the DAC_CTL0 register.

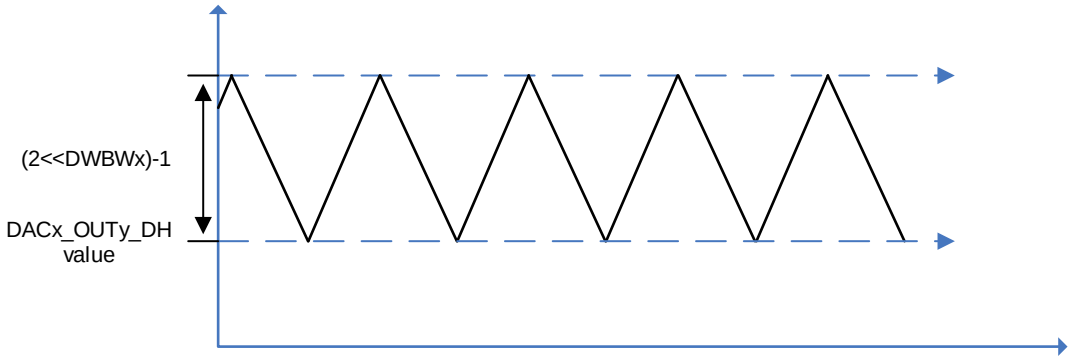
LFSR noise wave mode: there is a Linear Feedback Shift Register (LFSR) in the DAC control logic, it controls the LFSR noise signal which is added to the OUTx_DH value, and then the result is stored into the DAC_OUTx_DO register. When the configured DAC noise wave bit width is less than 12, the noise signal equals to the LSB DWBWx bits of the LFSR register, while the MSB bits are masked.

Figure 13-2. DAC LFSR algorithm



Triangle noise mode: a triangle signal is added to the OUTx_DH value, and then the result is stored into the DAC_OUTx_DO register. The minimum value of the triangle signal is 0, while the maximum value of the triangle signal is (2 << DWBWx) - 1.

Figure 13-3. DAC triangle noise wave



13.3.7. DAC output voltage

The following equation determines the analog output voltage on the DAC pin.

$$V_{DAC_OUT}=V_{REFP}*OUTx_DO/4096 \quad (13\text{-}1)$$

The digital input is linearly converted to an analog output voltage and its range is 0 to V_{REFP}

13.3.8. DMA request

When the external trigger is enabled, the DMA request is enabled by setting the DDMAENx bit of the DAC_CTL0 register. A DMA request will be generated when an external hardware trigger (not a software trigger) occurs.

13.3.9. DAC concurrent conversion

When the two output channels work at the same time, for maximum bus bandwidth utilization in specific applications, two output channels can be configured in concurrent mode. In concurrent mode, the OUTx_DH and OUTx_DO value will be updated at the same time.

There are three concurrent registers that can be used to load the OUTx_DH value: DACC_R8DH, DACC_R12DH and DACC_L12DH. User just need to access a unique register to realize driving two DAC channels at the same time.

When external trigger is enabled, please ensure both DTENx bits be set, DTSEL0/DTSEL1 bits be same to guarantee the simultaneous trigger.

When DMA is enabled, please ensure any DDMAENx bit in one DAC be set.

The noise mode and noise bit width can be configured either the same or different, depending on the application scenario.

13.4. Register definition

DAC0 base address: 0x4000 7400

13.4.1. DACx control register 0 (DAC_CTL0)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			DDMAEN1	DWBW1[3:0]				DWM1[1:0]		DTSEL1[2:0]			DTEN1	DBOFF1	DEN1
			rw	rw				rw		rw			rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DDMAEN0	DWBW0[3:0]				DWM0[1:0]		DTSEL0[2:0]			DTEN0	DBOFF0	DEN0
			rw	rw				rw		rw			rw	rw	rw

Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28	DDMAEN1	DACx_OUT1 DMA enable 0: DACx_OUT1 DMA mode disabled 1: DACx_OUT1 DMA mode enabled
27:24	DWBW1[3:0]	DACx_OUT1 noise wave bit width These bits specify bit width of the noise wave signal of DACx_OUT1. These bits indicate that unmask LFSR bit [n-1, 0] in LFSR noise mode or the amplitude of the triangle is ((2 << (n-1)) - 1) in triangle noise mode, where n is the bit width of wave. 0000: The bit width of the wave signal is 1 0001: The bit width of the wave signal is 2 0010: The bit width of the wave signal is 3 0011: The bit width of the wave signal is 4 0100: The bit width of the wave signal is 5 0101: The bit width of the wave signal is 6 0110: The bit width of the wave signal is 7 0111: The bit width of the wave signal is 8 1000: The bit width of the wave signal is 9 1001: The bit width of the wave signal is 10 1010: The bit width of the wave signal is 11 ≥1011: The bit width of the wave signal is 12
23:22	DWM1[1:0]	DACx_OUT1 noise wave mode These bits specify the mode selection of the noise wave signal of DACx_OUT1

		when external trigger of DACx_OUT1 is enabled (DTEN1=1). 00: wave disabled 01: LFSR noise mode 1x: Triangle noise mode
21:19	DTSEL1[2:0]	DACx_OUT1 trigger selection These bits are only used if bit DTEN1 = 1 and select the external event used to trigger DAC. 000: TIMER5 TRGO 001: TIMER2 TRGO 010: TIMER6 TRGO 011: TIMER4 TRGO 100: TIMER1 TRGO 101: TIMER3 TRGO 110: EXTI line 9 111: Software trigger
18	DTEN1	DACx_OUT1 trigger enable 0: DACx_OUT1 trigger disabled 1: DACx_OUT1 trigger enabled
17	DBOFF1	DACx_OUT1 output buffer turn off 0: DACx_OUT1 output buffer turns on to reduce the output impedance and improve the driving capability 1: DACx_OUT1 output buffer turns off
16	DEN1	DACx_OUT1 enable 0: DACx_OUT1 disabled 1: DACx_OUT1 enabled
15:13	Reserved	Must be kept at reset value.
12	DDMAEN0	DACx_OUT0 DMA enable 0: DACx_OUT0 DMA mode disabled 1: DACx_OUT0 DMA mode enabled
11:8	DWBW0[3:0]	DACx_OUT0 noise wave bit width These bits specify bit width of the noise wave signal of DACx_OUT0. These bits indicate that unmask LFSR bit [n-1, 0] in LFSR noise mode or the amplitude of the triangle is ((2<<(n-1))-1) in triangle noise mode, where n is the bit width of wave. 0000: The bit width of the wave signal is 1 0001: The bit width of the wave signal is 2 0010: The bit width of the wave signal is 3 0011: The bit width of the wave signal is 4 0100: The bit width of the wave signal is 5 0101: The bit width of the wave signal is 6 0110: The bit width of the wave signal is 7 0111: The bit width of the wave signal is 8

		1000: The bit width of the wave signal is 9 1001: The bit width of the wave signal is 10 1010: The bit width of the wave signal is 11 ≥1011: The bit width of the wave signal is 12
7:6	DWM0[1:0]	DACx_OUT0 noise wave mode These bits specify the mode selection of the noise wave signal of DACx_OUT0 when external trigger of DACx_OUT0 is enabled (DTEN0=1). 00: Wave disabled 01: LFSR noise mode 1x: Triangle noise mode
5:3	DTSEL0[2:0]	DACx_OUT0 trigger selection These bits are only used if bit DTEN0 = 1 and select the external event used to trigger DAC. 000: TIMER5 TRGO 001: TIMER2 TRGO 010: TIMER6 TRGO 011: TIMER4 TRGO 100: TIMER1 TRGO 101: TIMER3 TRGO 110: EXTI line 9 111: Software trigger
2	DTEN0	DACx_OUT0 trigger enable 0: DACx_OUT0 trigger disabled 1: DACx_OUT0 trigger enabled
1	DBOFF0	DACx_OUT0 output buffer turn off 0: DACx_OUT0 output buffer turns on to reduce the output impedance and improve the driving capability 1: DACx_OUT0 output buffer turns off
0	DEN0	DACx_OUT0 enable 0: DACx_OUT0 disabled 1: DACx_OUT0 enabled

13.4.2. DACx software trigger register (DAC_SWT)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

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

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

Reserved	SWTR1	SWTR0
	w	w

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	SWTR1	DACx_OUT1 software trigger, cleared by hardware. 0: Software trigger disabled 1: Software trigger enabled
0	SWTR0	DACx_OUT0 software trigger, cleared by hardware. 0: Software trigger disabled 1: Software trigger enabled

13.4.3. DACx_OUT0 12-bit right-aligned data holding register (DAC_OUT0_R12DH)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				OUT0_DH[11:0]											
				rw											

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	OUT0_DH[11:0]	DACx_OUT0 12-bit right-aligned data These bits specify the data that is to be converted by DACx_OUT0.

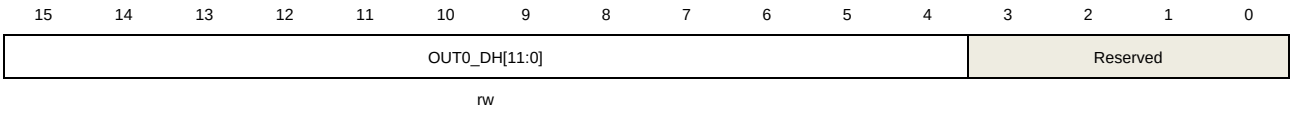
13.4.4. DACx_OUT0 12-bit left-aligned data holding register (DAC_OUT0_L12DH)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

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



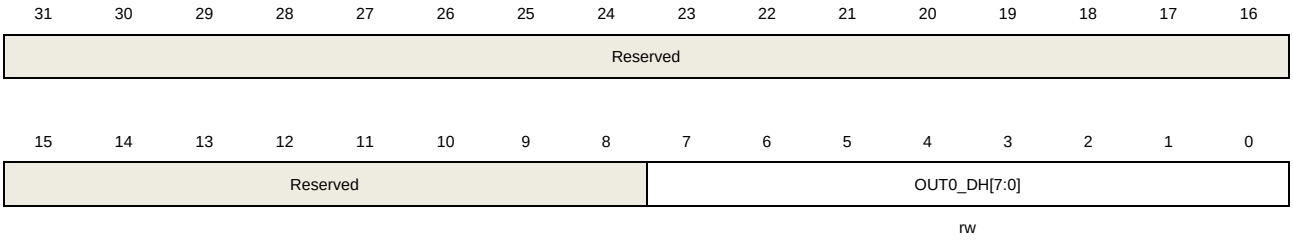
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:4	OUT0_DH[11:0]	DACx_OUT0 12-bit left-aligned data These bits specify the data that is to be converted by DACx_OUT0.
3:0	Reserved	Must be kept at reset value.

13.4.5. DACx_OUT0 8-bit right-aligned data holding register (DAC_OUT0_R8DH)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



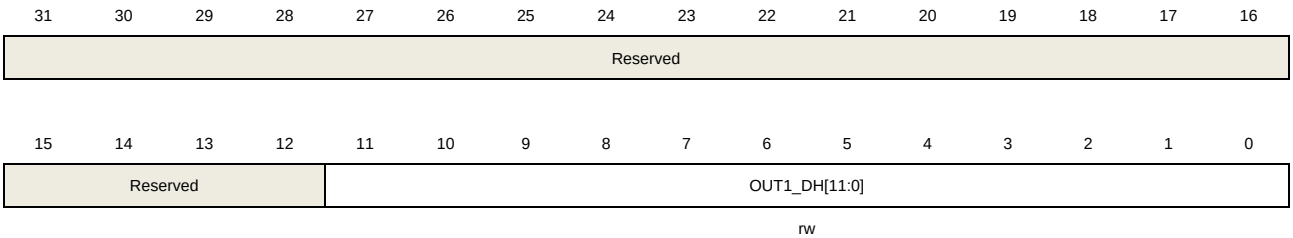
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	OUT0_DH[7:0]	DACx_OUT0 8-bit right-aligned data These bits specify the MSB 8-bit of the data that is to be converted by DACx_OUT0.

13.4.6. DACx_OUT1 12-bit right-aligned data holding register (DAC_OUT1_R12DH)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

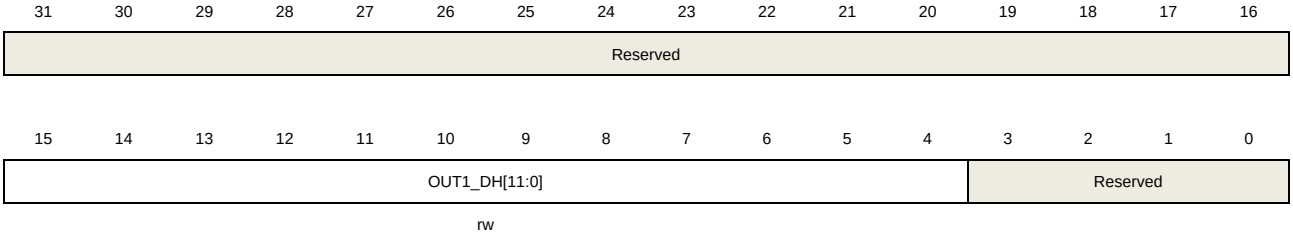


Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	OUT1_DH[11:0]	DACx_OUT1 12-bit right-aligned data These bits specify the data that is to be converted by DACx_OUT1.

13.4.7. DACx_OUT1 12-bit left-aligned data holding register (DAC_OUT1_L12DH)

Address offset: 0x18
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

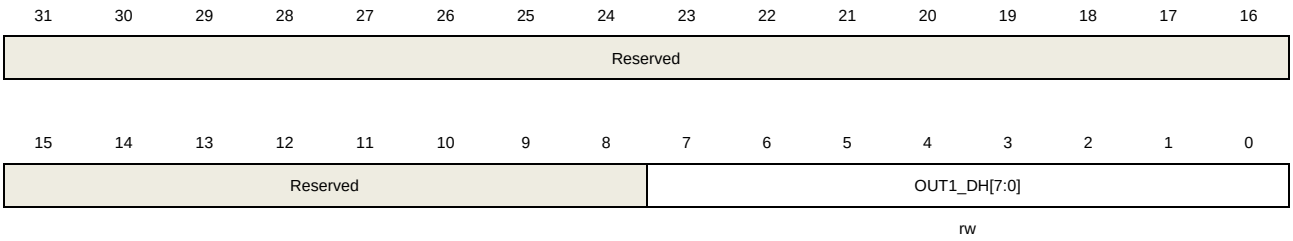


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:4	OUT1_DH[11:0]	DACx_OUT1 12-bit left-aligned data These bits specify the data that is to be converted by DACx_OUT1.
3:0	Reserved	Must be kept at reset value.

13.4.8. DACx_OUT1 8-bit right-aligned data holding register (DAC_OUT1_R8DH)

Address offset: 0x1C
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	OUT1_DH[7:0]	DACx_OUT1 8-bit right-aligned data

These bits specify the MSB 8-bit of the data that is to be converted by DACx_OUT1.

13.4.9. DACx concurrent mode 12-bit right-aligned data holding register (DACC_R12DH)

Address offset: 0x20

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				OUT1_DH[11:0]											
				rw											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				OUT0_DH[11:0]											
				rw											

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:16	OUT1_DH[11:0]	DACx_OUT1 12-bit right-aligned data These bits specify the data that is to be converted by DACx_OUT1.
15:12	Reserved	Must be kept at reset value.
11:0	OUT0_DH[11:0]	DACx_OUT0 12-bit right-aligned data These bits specify the data that is to be converted by DACx_OUT0.

13.4.10. DACx concurrent mode 12-bit left-aligned data holding register (DACC_L12DH)

Address offset: 0x24

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OUT1_DH[11:0]												Reserved			
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT0_DH[11:0]												Reserved			
rw															

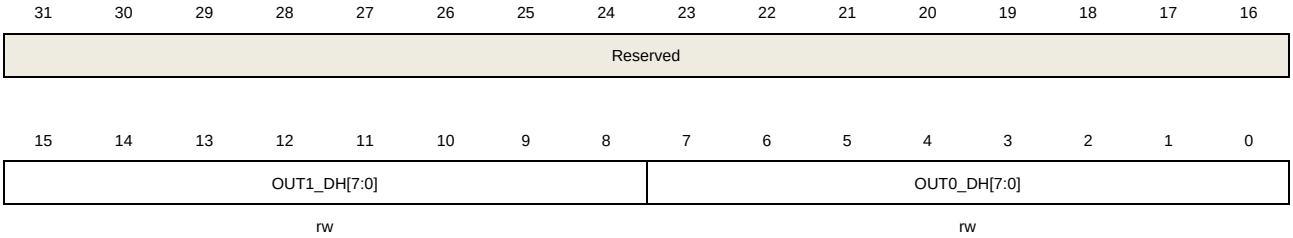
Bits	Fields	Descriptions
31:20	OUT1_DH[11:0]	DACx_OUT1 12-bit left-aligned data These bits specify the data that is to be converted by DACx_OUT1.

19:16	Reserved	Must be kept at reset value.
15:4	OUT0_DH[11:0]	DACx_OUT0 12-bit left-aligned data These bits specify the data that is to be converted by DACx_OUT0.
3:0	Reserved	Must be kept at reset value.

13.4.11. DACx concurrent mode 8-bit right-aligned data holding register (DACC_R8DH)

Address offset: 0x28
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

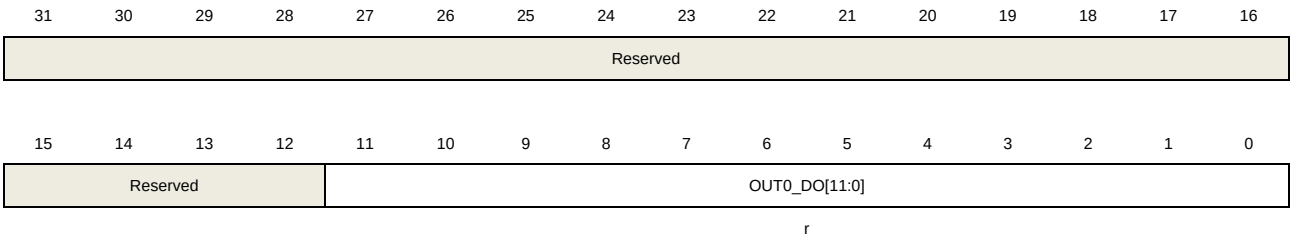


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:8	OUT1_DH[7:0]	DACx_OUT1 8-bit right-aligned data These bits specify the MSB 8-bit of the data that is to be converted by DACx_OUT1.
7:0	OUT0_DH[7:0]	DACx_OUT0 8-bit right-aligned data These bits specify the MSB 8-bit of the data that is to be converted by DACx_OUT0.

13.4.12. DACx_OUT0 data output register (DAC_OUT0_DO)

Address offset: 0x2C
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions

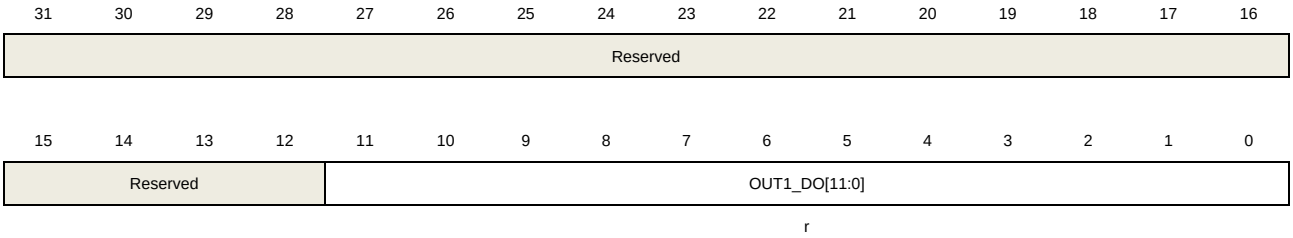
31:12	Reserved	Must be kept at reset value.
11:0	OUT0_DO[11:0]	DACx_OUT0 12-bit output data These bits, which are read only, storage the data that is being converted by DACx_OUT0.

13.4.13. DACx_OUT1 data output register (DAC_OUT1_DO)

Address offset: 0x30

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	OUT1_DO[11:0]	DACx_OUT1 12-bit output data These bits, which are read only, storage the data that is being converted by DACx_OUT1.

14. Watchdog timer (WDGT)

The watchdog timer (WDGT) is a hardware timing circuitry that can be used to detect system failures due to software malfunctions. There are two watchdog timer peripherals in the chip: free watchdog timer (FWDGT) and window watchdog timer (WWDGT). They offer a combination of a high safety level, flexibility of use and high timing accuracy. Both watchdog timers are offered to resolve malfunctions of software.

The watchdog timer will generate a reset when the internal counter reaches a given value. The watchdog timer counter can be stopped while the processor is in the debug mode.

14.1. Free watchdog timer (FWDGT)

14.1.1. Overview

The free watchdog timer (FWDGT) has free clock source (IRC40K). Thereupon the FWDGT can operate even if the main clock fails. It's suitable for the situation that requires an independent environment and lower timing accuracy.

The free watchdog timer causes a reset when the internal down counter reaches 0. The register write protection function in free watchdog can be enabled to prevent it from changing the configuration unexpectedly.

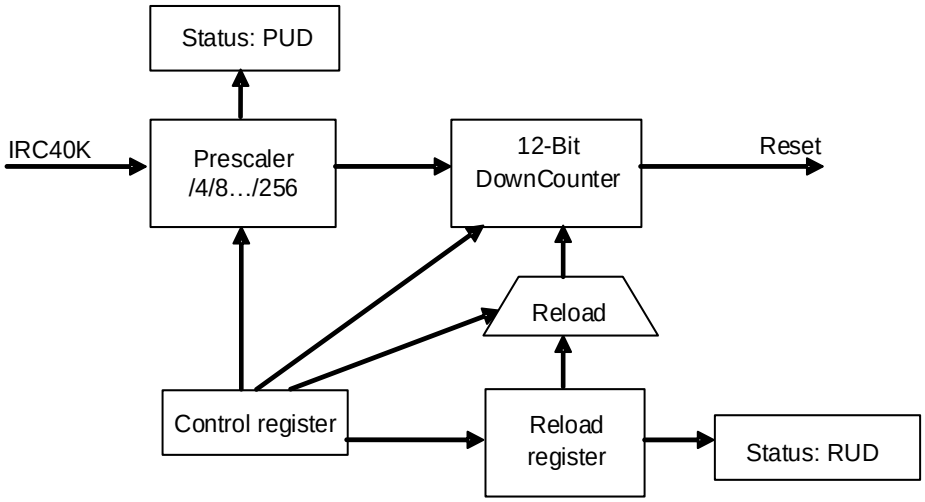
14.1.2. Characteristics

- Free-running 12-bit down counter.
- Reset when the down counter reaches 0, if the watchdog is enabled.
- Free clock source, FWDGT can operate even if the main clock fails such as in standby and Deep-sleep modes.
- Hardware free watchdog bit, automatically start the FWDGT or not when power on.
- FWDGT debug mode, the FWDGT can stop or continue to work in debug mode.

14.1.3. Function overview

The free watchdog consists of an 8-stage prescaler and a 12-bit down counter. ***Figure 14-1. Free watchdog block diagram*** shows the functional block of the free watchdog module.

Figure 14-1. Free watchdog block diagram



The free watchdog is enabled by writing the value (0xCCCC) to the control register (FWDGT_CTL), then the counter starts counting down. When the counter reaches the value (0x000), there will be a reset.

The counter can be reloaded by writing the value (0xAAAA) to the FWDGT_CTL register at any time. The reload value comes from the FWDGT_RLD register. The software can prevent the watchdog reset by reloading the counter before the counter reaches the value (0x000).

The free watchdog can automatically start when power on if the hardware free watchdog bit in the device option bits is set. To avoid a reset, the software should reload the counter before the counter reaches 0x000.

The FWDGT_PSC register and the FWDGT_RLD register are write protected. Before writing these registers, the software should write the value (0x5555) to the FWDGT_CTL register. These registers will be protected again by writing any other value to the FWDGT_CTL register. When an update operation of the prescaler register (FWDGT_PSC) or the reload value register (FWDGT_RLD) is ongoing, the status bits in the FWDGT_STAT register are set.

If the FWDGT_HOLD bit in DBG module is cleared, the FWDGT continues to work even the Cortex®-M4 core halted (Debug mode). The FWDGT stops in Debug mode if the FWDGT_HOLD bit is set.

Table 14-1. Min/max FWDGT timeout period at 40 kHz (IRC40K)

Prescaler divider	PSC[2:0] bits	Min timeout (ms) RLD[11:0]=0x000	Max timeout (ms) RLD[11:0]=0xFFF
1 / 4	000	0.025	409.525
1 / 8	001	0.025	819.025
1 / 16	010	0.025	1638.025
1 / 32	011	0.025	3276.025
1 / 64	100	0.025	6552.025
1 / 128	101	0.025	13104.025
1 / 256	110 or 111	0.025	26208.025

The FWDGT timeout can be more accurate by calibrating the IRC40K.

Note: 1. When after the execution of watchdog reload operation, if the MCU needs enter the deepsleep / standby mode immediately, more than 3 IRC40K clock intervals must be inserted in the middle of reload and deepsleep / standby mode commands by software setting.
2. There must be an interval of 7 or more IRC40K clock cycles between two consecutive watchdog reload operations.

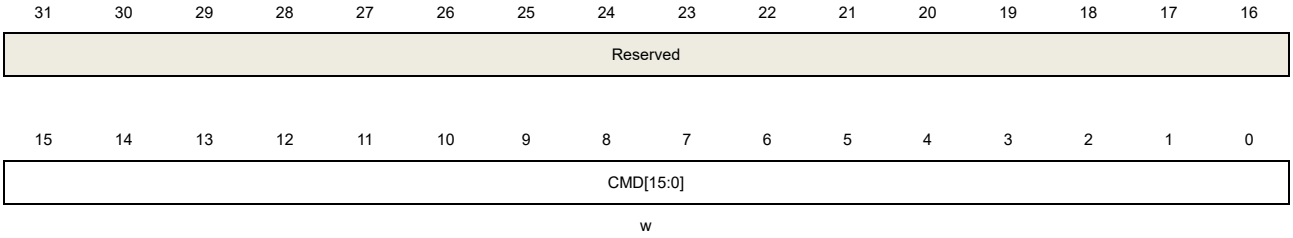
14.1.4. Register definition

FWDGT base address: 0x4000 3000

Control register (FWDGT_CTL)

Address offset: 0x00
Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit) access.

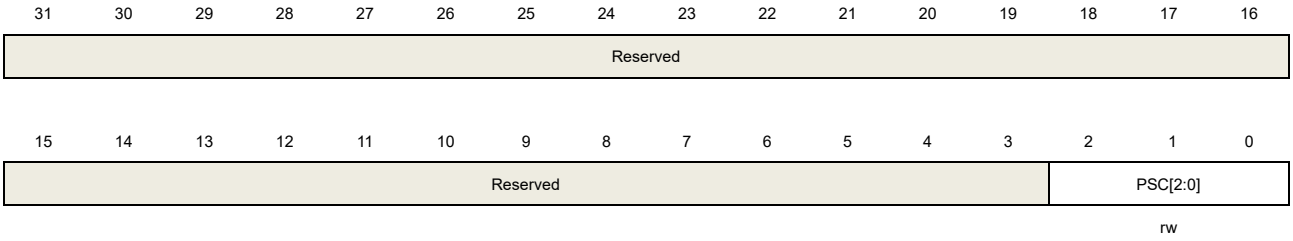


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CMD[15:0]	Write only. Several different functions are realized by writing these bits with different values. 0x5555: Disable the FWDGT_PSC and FWDGT_RLD write protection 0xCCCC: Start the free watchdog counter. When the counter reduces to 0, the free watchdog generates a reset. 0xAAAA: Reload the counter

Prescaler register (FWDGT_PSC)

Address offset: 0x04
Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit) access.



Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2:0	PSC[2:0]	Free watchdog timer prescaler selection. Write 0x5555 to the FWDGT_CTL register before writing these bits. During a write operation to this register, the PUD bit in the FWDGT_STAT register is set and the value read from this register is invalid.

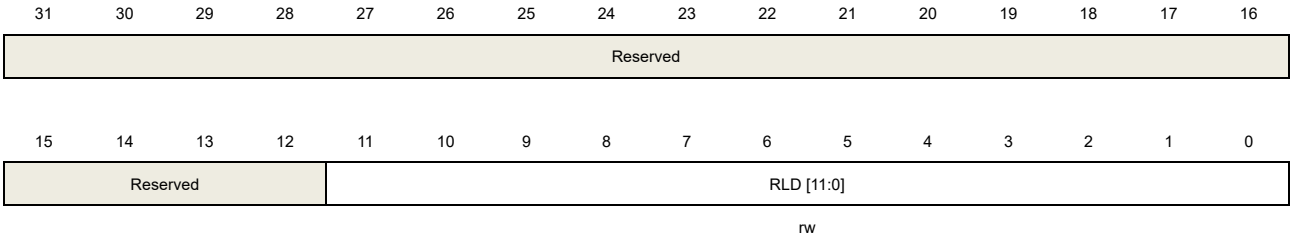
000: 1 / 4
001: 1 / 8
010: 1 / 16
011: 1 / 32
100: 1 / 64
101: 1 / 128
110: 1 / 256
111: 1 / 256

If several prescaler values are used by the application, it is mandatory to wait until PUD bit has been reset before changing the prescaler value. If the prescaler value has been updated, it is not necessary to wait until PUD has been reset before continuing code execution (Before entering low-power mode, it is necessary to wait until PUD is reset).

Reload register (FWDGT_RLD)

Address offset: 0x08
Reset value: 0x0000 0FFF

This register can be accessed by half-word (16-bit) or word (32-bit) access.

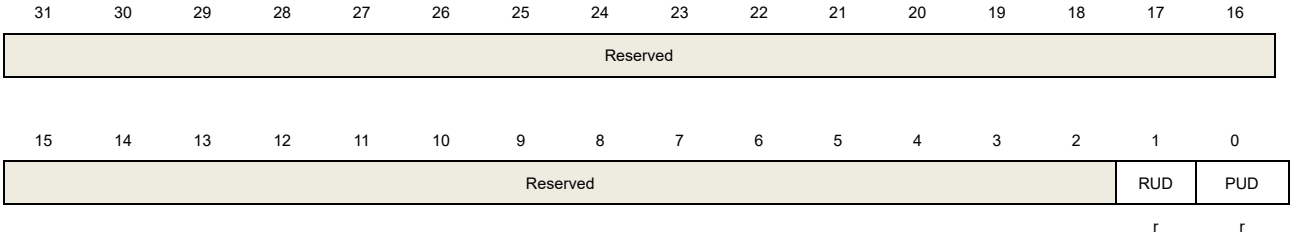


Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	RLD[11:0]	Free watchdog timer counter reload value. Write 0xAAAA to the FWDGT_CTL register will reload the FWDGT counter with the RLD value. These bits are write protected. Write 0x5555 to the FWDGT_CTL register before writing these bits. During a write operation to this register, the RUD bit in the FWDGT_STAT register is set and the value read from this register is invalid. If several reload values are used by the application, it is mandatory to wait until RUD bit has been reset before changing the reload value. If the reload value has been updated, it is not necessary to wait until RUD has been reset before continuing code execution (Before entering low-power mode, it is necessary to wait until PUD is reset).

Status register (FWDGT_STAT)

Address offset: 0x0C
Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit) access.



Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	RUD	Free watchdog timer counter reload value update During a write operation to FWDGT_RLD register, this bit is set and the value read from FWDGT_RLD register is invalid. This bit is reset by hardware after the update operation of the FWDGT_RLD register.
0	PUD	Free watchdog timer prescaler value update During a write operation to FWDGT_PSC register, this bit is set and the value read from FWDGT_PSC register is invalid. This bit is reset by hardware after the update operation of the FWDGT_PSC register.

14.2. Window watchdog timer (WWDGT)

14.2.1. Overview

The window watchdog timer (WWDGT) is used to detect system failures due to software malfunctions. After the window watchdog timer starts, the value of down counter reduces progressively. The watchdog timer causes a reset when the counter reached 0x3F (the CNT [6] bit has been cleared). The watchdog timer also causes a reset when the counter is refreshed before the counter reached the window register value. So the software should refresh the counter in a limited window. The window watchdog timer generates an early wakeup status flag when the counter reaches 0x40. Interrupt occurs if it is enable.

The window watchdog timer clock is prescaled from the APB1 clock. The window watchdog timer is suitable for the situation that requires an accurate timing.

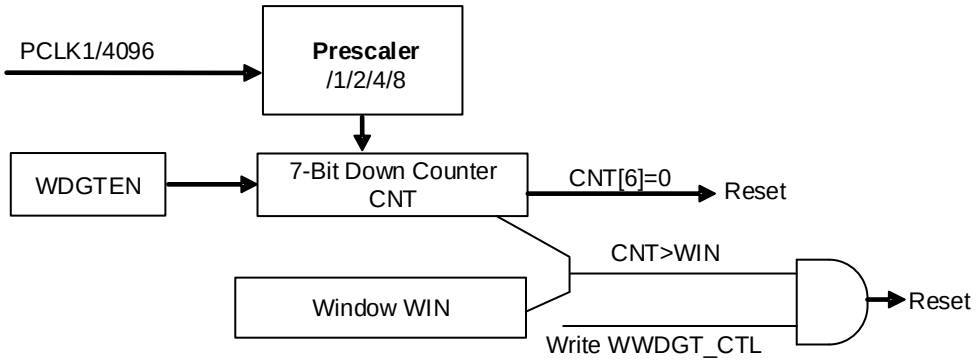
14.2.2. Characteristics

- Programmable free-running 7-bit down counter.
- Generate a reset in two conditions when WWDGT is enabled:
  - Reset when the counter reached 0x3F.
  - The counter is refreshed when the value of the counter is greater than the window register value.
- Early wakeup interrupt (EWI): the watchdog is started and the interrupt is enabled, the interrupt occurs when the counter reaches 0x40.
- WWDGT debug mode, the WWDGT can stop or continue to work in debug mode.

14.2.3. Function overview

If the window watchdog timer is enabled (set the WDGTEN bit in the WWDGT_CTL), the watchdog timer causes a reset when the counter reaches 0x3F (the CNT [6] bit has been cleared), or the counter is refreshed before the counter reaches the window register value.

Figure 14-2. Window watchdog timer block diagram



The window watchdog timer is always disabled after power on reset. The software starts the

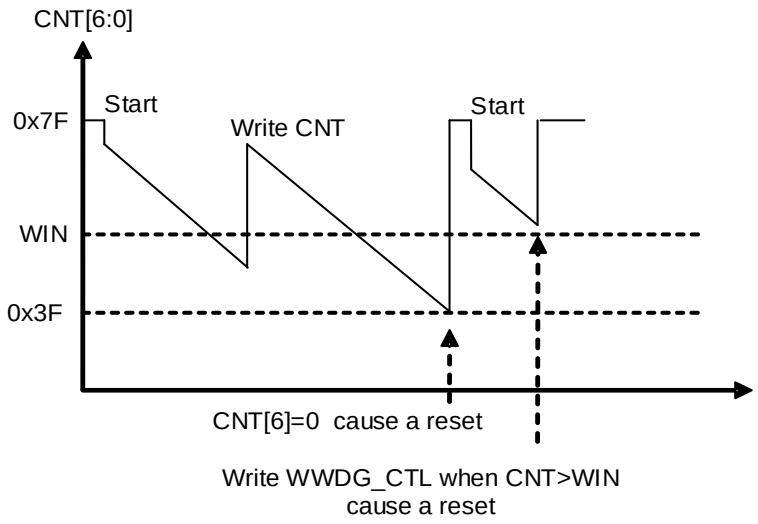
watchdog by setting the WDGTEN bit in the WWDGT_CTL register. When window watchdog timer is enabled, the counter counts down all the time, the configured value of the counter should be greater than 0x3F (it implies that the CNT [6] bit should be set). The CNT [5:0] determine the maximum time interval between two reloading. The count down speed depends on the APB1 clock and the prescaler (PSC [1:0] bits in the WWDGT_CFG register).

The WIN [6:0] bits in the configuration register (WWDGT_CFG) specify the window value. The software can prevent the reset event by reloading the down counter. The counter value is less than the window value and greater than 0x3F, otherwise the watchdog causes a reset.

The early wakeup interrupt (EWI) is enabled by setting the EWIE bit in the WWDGT_CFG register, and the interrupt will be generated when the counter reaches 0x40. The software can do something such as communication or data logging in the interrupt service routine (ISR) in order to analyze the reason of software malfunctions or save the important data before resetting the device. Moreover the software can reload the counter in ISR to manage a software system check and so on. In this case, the WWDGT will never generate a WWDGT reset but can be used for other things.

The EWI interrupt is cleared by writing '0' to the EWIF bit in the WWDGT_STAT register.

Figure 14-3. Window watchdog timing diagram



Calculate the WWDGT timeout by using the formula below.

$$t_{WWDGT}=t_{PCLK1} \times 4096 \times 2^{PSC} \times (CNT[5:0]+1) \quad (ms) \qquad (14\text{-}1)$$

where:

t_{WWDGT}: WWDGT timeout

t_{PCLK1}: APB1 clock period measured in ms

The table below shows the minimum and maximum values of the t_{WWDGT}.

Table 14-2. Min/max timeout value at 60 MHz (f_{PCLK1})

Prescaler divider	PSC[1:0]	Min timeout value CNT[6:0] =0x40	Max timeout value CNT[6:0]=0x7F

1 / 1	00	68.2 μs	4.3ms
1 / 2	01	136.4 μs	8.6 ms
1 / 4	10	272.8μs	17.2 ms
1 / 8	11	545.6 μs	34.4 ms

If the WWDGT_HOLD bit in DBG module is cleared, the WWDGT continues to work even the Cortex®-M4 core halted (Debug mode). While the WWDGT_HOLD bit is set, the WWDGT stops in Debug mode.

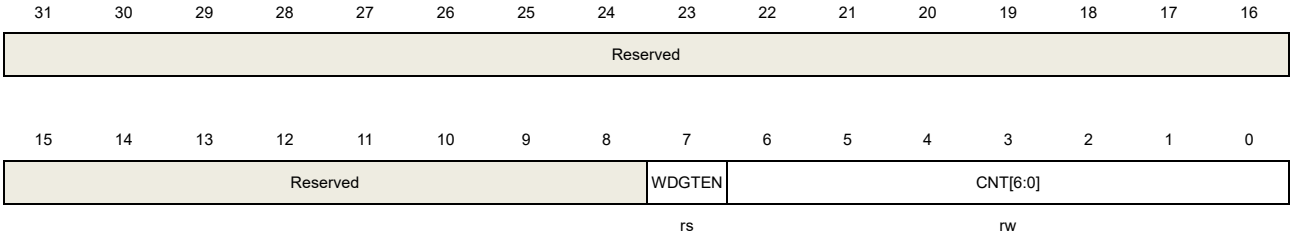
14.2.4. Register definition

WWDGT base address: 0x4000 2C00

Control register (WWDGT_CTL)

Address offset: 0x00
Reset value: 0x0000 007F

This register can be accessed by half-word (16-bit) or word (32-bit).

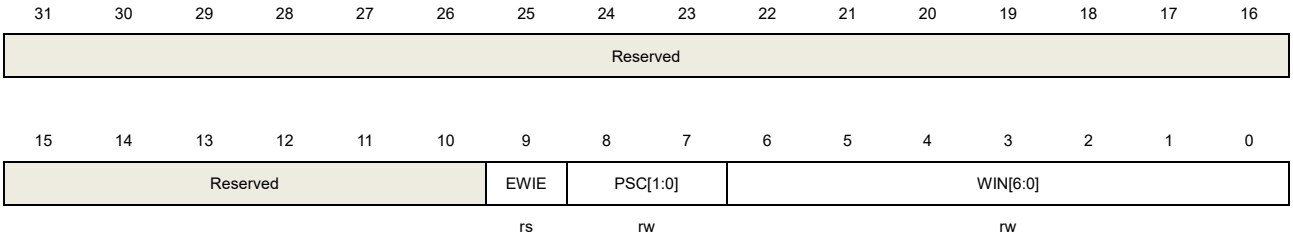


Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	WDGTEN	Start the window watchdog timer. Cleared by a hardware reset. Writing 0 has no effect. 0: Window watchdog timer disabled 1: Window watchdog timer enabled
6:0	CNT[6:0]	The value of the watchdog timer counter. A reset occurs when the value of this counter decreases from 0x40 to 0x3F. When the value of this counter is greater than the window value, writing this counter also causes a reset.

Configuration register (WWDGT_CFG)

Address offset: 0x04
Reset value: 0x0000 007F

This register can be accessed by half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	EWIE	Early wakeup interrupt enable. If the bit is set, an interrupt occurs when the counter reaches 0x40. It can be cleared by a hardware reset or a software reset by setting

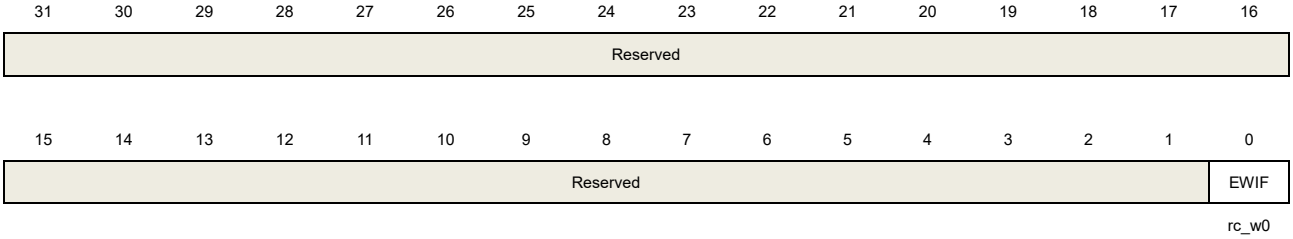
		the WWDGTRST bit of the RCU module.. A write operation of '0' has no effect.
8:7	PSC[1:0]	Prescaler. The time base of the watchdog timer counter. 00: (PCLK1 / 4096) / 1 01: (PCLK1 / 4096) / 2 10: (PCLK1 / 4096) / 4 11: (PCLK1 / 4096) / 8
6:0	WIN[6:0]	The Window value. A reset occurs if the watchdog counter (CNT bits in WWDGT_CTL) is written when the value of the watchdog counter is greater than the Window value.

Status register (WWDGT_STAT)

Address offset: 0x08

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	EWIF	Early wakeup interrupt flag. When the counter reaches 0x40, this bit is set by hardware even the interrupt is not enabled (EWIE in WWDGT_CFG is cleared). This bit is cleared by writing 0. There is no effect when writing 1.

15. Real-time Clock (RTC)

15.1. Overview

The RTC is usually used as a clock-calendar. The RTC circuits are located in two power supply domains. The circuits in the backup domain consist of a 32-bit up-counter, an alarm, a prescaler, a divider and the RTC clock configuration register. That means the RTC settings and time are kept when the device resets or wakes up from Standby mode. While the other circuits in the V_{DD} domain only include the APB interface and a control register. In the following sections, the details of the RTC function will be described.

15.2. Characteristics

- 32-bit programmable counter for counting elapsed time
  - Programmable prescaler: Max division factor is up to 2^{20}
- Separate clock domains:
  - PCLK1 clock domain
  - RTC clock domain (this clock must be at least 4 times slower than the PCLK1 clock)
- RTC clock source:
  - HXTAL clock divided by 128
  - LXTAL oscillator clock
  - IRC40K oscillator clock
- Maskable interrupt source:
  - Alarm interrupt
  - Second interrupt
  - Overflow interrupt

15.3. Function overview

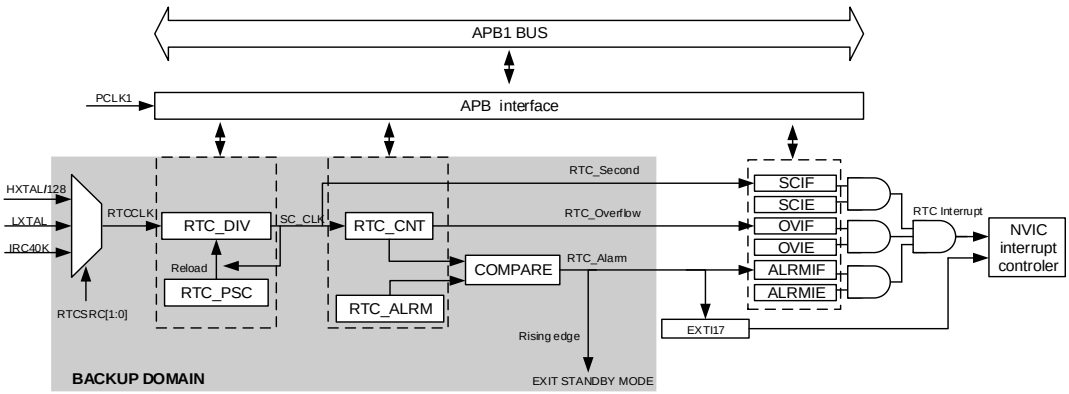
The RTC circuits consist of two major units: APB interface located in PCLK1 clock domain and RTC core located in RTC clock domain.

APB interface is connected with the APB1 bus. It includes a set of registers, which can be accessed by APB1 bus.

RTC core includes two major blocks. One is the RTC prescaler block, which generates the RTC time base clock SC_CLK. RTC prescaler block includes a 20-bit programmable divider (RTC prescaler) which can generate SC_CLK by dividing the RTC source clock. If second interrupt is enabled in the RTC_INTEN register, the RTC will generate an interrupt at every SC_CLK rising edge. Another block is a 32-bit programmable counter, which can be initialized with the value of current system time. If alarm interrupt is enabled in the RTC_INTEN register, the RTC will generate an alarm interrupt when the system time equals to the alarm time

(stored in the RTC_ALRMH/L register).

Figure 15-1. Block diagram of RTC



15.3.1. RTC reset

The APB interface and the RTC_INTEN register are reset by a system reset. The RTC core (prescaler, divider, counter and alarm) is reset only by a backup domain reset.

Steps to enable access to the backup registers and the RTC after reset are as follows:

1. Set the PMUEN and BKPIEN bits in the RCU_APB1EN register to enable the power and backup interface clocks.

2. Enable access to the backup registers and RTC by setting the BKPWEN bit in the PMU_CTL register.

15.3.2. RTC reading

The APB interface and RTC core are located in two different power supply domains.

In the RTC core, only counter and divider registers are readable registers. And the values in the two registers and the RTC flags are internally updated at each rising edge of the RTC clock, which is resynchronized by the APB1 clock.

When the APB interface is enabled from a disabled state, the read operation is not recommended to be done immediately because the first internal update of the registers has not finished. That means, when a system reset, a power reset or a wakeup from standby/Deep-sleep mode occurs, the APB interface is disabled, and the RTC core keeps running. In these cases, the correct read operation is that clear the RSYNF bit in the RTC_CTL register first and then wait for it to be set by hardware. WFI and WFE have no effects on the RTC APB interface.

15.3.3. RTC configuration

The RTC_PSC, RTC_CNT and RTC_ALRM registers in the RTC core are writable. The values of these registers can be configured only when the peripheral has entered

configuration mode. And the CMF bit in the RTC_CTL register is used to indicate the configuration mode status. The write operation takes effect only when the peripheral has exited configuration mode, and it takes at least three RTCCLK cycles. The value of the LWOFF bit in the RTC_CTL register will be set to '1' after the write operation is finished. The new write operation should be performed after the previous one is finished.

The configuration steps are as follows:

A) Wait until the value of LWOFF bit in the RTC_CTL register to be set to '1'.
B) Enter configuration mode by setting the CMF bit in the RTC_CTL register.
C) Write to the RTC registers.
D) Exit configuration mode by clearing the CMF bit in the RTC_CTL register.
E) Wait until the value of LWOFF bit in the RTC_CTL register to be set to '1'.

15.3.4. RTC flag assertion

Before the update of the RTC counter, the RTC second interrupt flag (SCIF) is asserted on the last RTCCLK cycle.

Before the counter equals to the RTC alarm value which is stored in the alarm register plus one, the RTC alarm interrupt flag (ALRMIF) is asserted on the last RTCCLK cycle.

Before the counter equals to 0x0, the RTC overflow interrupt flag (OVIF) is asserted on the last RTCCLK cycle.

The RTC alarm write operation and second interrupt flag must be synchronized by using either of the following sequences:

- Enable the RTC alarm interrupt and update the RTC alarm and/or RTC counter registers in the RTC interrupt service routine.
- Update the RTC alarm and/or the RTC counter registers after the SCIF bit is set in the RTC_CTL register.

Figure 15-2. RTC second and alarm waveform example (RTC_PSC = 3, RTC_ALRM = 2)

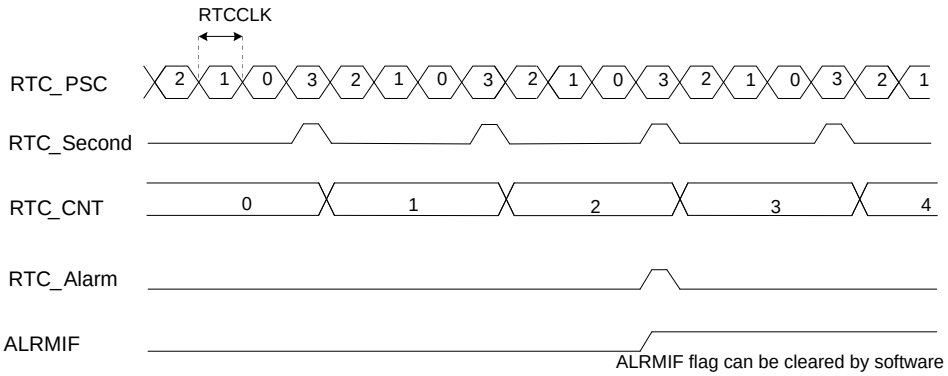
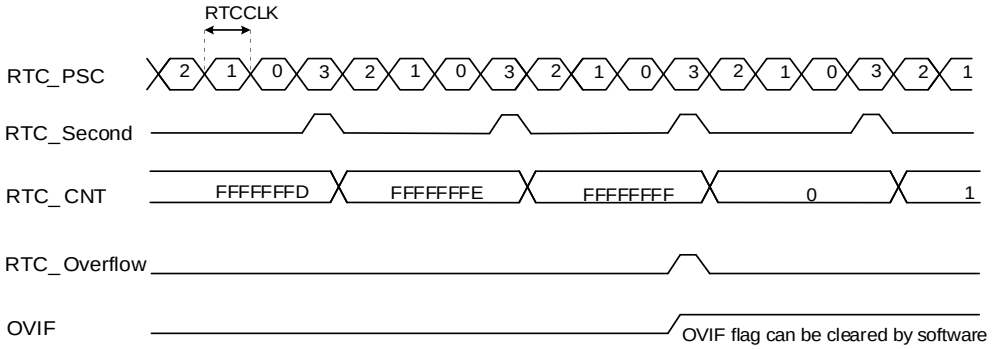


Figure 15-3. RTC second and overflow waveform example (RTC_PSC= 3)



15.4. Register definition

RTC base address: 0x4000 2800

15.4.1. RTC interrupt enable register(RTC_INTEN)

Address offset: 0x00

Reset value: 0x0000

This register can be accessed by half-word (16-bit) or word (32-bit)

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

Bits	Fields	Descriptions
15:3	Reserved	Must be kept at reset value.
2	OVIE	Overflow interrupt enable 0: Disable overflow interrupt 1: Enable overflow interrupt
1	ALRMIE	Alarm interrupt enable 0: Disable alarm interrupt 1: Enable alarm interrupt
0	SCIE	Second interrupt enable 0: Disable second interrupt 1: Enable second interrupt

15.4.2. RTC control register(RTC_CTL)

Address offset: 0x04

Reset value: 0x0020

This register can be accessed by half-word (16-bit) or word (32-bit)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										LWOFF	CMF	RSYNF	OVIF	ALRMIF	SCIF
										r	rw	rc_w0	rc_w0	rc_w0	rc_w0

Bits	Fields	Descriptions
15:6	Reserved	Must be kept at reset value.
5	LWOFF	Last write operation finished flag 0: Last write operation on RTC registers is not finished 1: Last write operation on RTC registers is finished
4	CMF	Configuration mode flag

		0: Exit configuration mode 1: Enter configuration mode
3	RSYNF	Registers synchronized flag 0: Registers not yet synchronized with the APB1 clock 1: Registers synchronized with the APB1 clock
2	OVIF	Overflow interrupt flag 0: Overflow event not detected 1: Overflow event detected. An interrupt will occur if the OVIE bit is set in RTC_INTEN.
1	ALRMIF	Alarm interrupt flag 0: Alarm event not detected 1: Alarm event detected. A RTC global interrupt will occur if the ALRMIE bit is set in RTC_INTEN. And a RTC alarm interrupt will occur if the EXTI17 is enabled in interrupt mode.
0	SCIF	Second interrupt flag 0: Second event not detected. 1: Second event detected. An interrupt will occur if the SCIE bit is set in RTC_INTEN. Set by hardware when the divider reloads the value in RTC_PSCH/L, thus incrementing the RTC counter.

15.4.3. RTC prescaler high register (RTC_PSCH)

Address offset: 0x08
Reset value: 0x0000

This register can be accessed by half-word (16-bit) or word (32-bit)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												PSC[19:16]			
												w			

Bits	Fields	Descriptions
15:4	Reserved	Must be kept at reset value.
3:0	PSC[19:16]	RTC prescaler value high

15.4.4. RTC prescaler low register(RTC_PSCL)

Address offset: 0x0C
Reset value: 0x8000

This register can be accessed by half-word (16-bit) or word (32-bit)

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

PSC[15:0]

w

Bits	Fields	Descriptions
15:0	PSC[15:0]	RTC prescaler value low The frequency of SC_CLK is the RTCCLK frequency divided by (PSC[19:0]+1).

15.4.5. RTC divider high register (RTC_DIVH)

Address offset: 0x10
Reset value: 0x0000

This register can be accessed by half-word (16-bit) or word (32-bit)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												DIV[19:16]			

r

Bits	Fields	Descriptions
15:4	Reserved	Must be kept at reset value.
3:0	DIV[19:16]	RTC divider value high

15.4.6. RTC divider low register (RTC_DIVL)

Address offset: 0x14
Reset value: 0x8000

This register can be accessed by half-word (16-bit) or word (32-bit)

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

r

Bits	Fields	Descriptions
15:0	DIV[15:0]	RTC divider value low The RTC divider register is reloaded by hardware when the RTC prescaler or RTC counter register updated.

15.4.7. RTC counter high register(RTC_CNTH)

Address offset: 0x18
Reset value: 0x0000

This register can be accessed by half-word (16-bit) or word (32-bit)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[31:16]															

rw

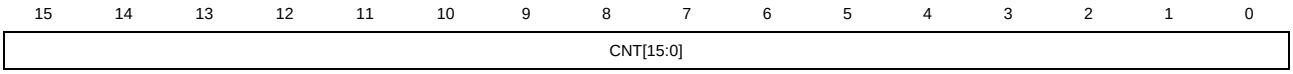
Bits	Fields	Descriptions
15:0	CNT[31:16]	RTC counter value high

15.4.8. RTC counter low register (RTC_CNTL)

Address offset: 0x1C

Reset value: 0x0000

This register can be accessed by half-word (16-bit) or word (32-bit)



rw

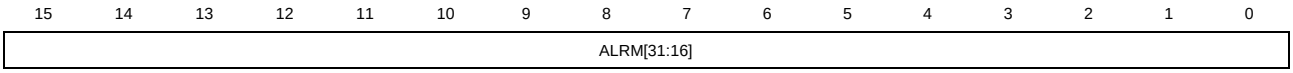
Bits	Fields	Descriptions
15:0	CNT[15:0]	RTC counter value low

15.4.9. RTC alarm high register(RTC_ALRMH)

Address offset: 0x20

Reset value: 0xFFFF

This register can be accessed by half-word (16-bit) or word (32-bit)



w

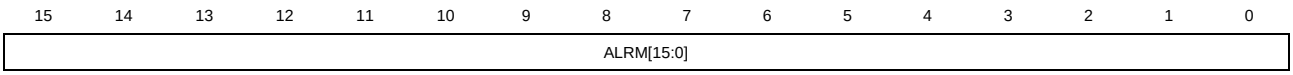
Bits	Fields	Descriptions
15:0	ALRM[31:16]	RTC alarm value high

15.4.10. RTC alarm low register (RTC_ALRML)

Address offset: 0x24

Reset value: 0xFFFF

This register can be accessed by half-word (16-bit) or word (32-bit)



w

Bits	Fields	Descriptions
15:0	ALRM[15:0]	RTC alarm value low

16. TIMER

Table 16-1. Timers (TIMERx) are divided into five sorts

TIMER	TIMER0/7	TIMER1/2/3/4	TIMER8/11	TIMER9/10/12/13	TIMER5/6
TYPE	Advanced	General-L0	General-L1	General-L2	Basic
Prescaler	16-bit	16-bit	16-bit	16-bit	16-bit
Counter	16-bit	16-bit	16-bit	16-bit	16-bit
Count mode	UP, DOWN, Center-aligned	UP, DOWN, Center-aligned	UP ONLY	UP ONLY	UP ONLY
Repetition	●	×	×	×	×
CH Capture/ Compare	4	4	2	1	0
Complementary & Dead-time	●	×	×	×	×
Break	●	×	×	×	×
Single Pulse	●	●	●	×	●
Quadrature Decoder	●	●	×	×	×
Master-slave management	●	●	●	×	×
Inter Connection	● (1)	● (2)	● (3)	×	TRGO TO DAC
DMA	●	●	×	×	● (4)
Debug Mode	●	●	●	●	●

(1)

TIMER0	**ITI0:** TIMER4_TRGO	**ITI1:** TIMER1_TRGO	**ITI2:** TIMER2_TRGO	**ITI3:** TIMER3_TRGO
TIMER7	**ITI0:** TIMER0_TRGO	**ITI1:** TIMER1_TRGO	**ITI2:** TIMER3_TRGO	**ITI3:** TIMER4_TRGO

(2)

TIMER1	**ITI0:** TIMER0_TRGO	**ITI1:** 0	**ITI2:** TIMER2_TRGO	**ITI3:** TIMER3_TRGO
TIMER2	**ITI0:** TIMER0_TRGO	**ITI1:** TIMER1_TRGO	**ITI2:** TIMER4_TRGO	**ITI3:** TIMER3_TRGO
TIMER3	**ITI0:** TIMER0_TRGO	**ITI1:** TIMER1_TRGO	**ITI2:** TIMER2_TRGO	**ITI3:** TIMER7_TRGO
TIMER4	**ITI0:** TIMER1_TRGO	**ITI1:** TIMER2_TRGO	**ITI2:** TIMER3_TRGO	**ITI3:** TIMER7_TRGO

(3)

TIMER8	**ITI0:** TIMER1_TRGO	**ITI1:** TIMER2_TRGO	**ITI2:** TIMER9_TRGO	**ITI3:** TIMER10_ TRGO
TIMER11	**ITI0:** TIMER3_TRGO	**ITI1:** TIMER4_TRGO	**ITI2:** TIMER12_TRGO	**ITI3:** TIMER13_ TRGO

(4) Only update events will generate a DMA request. TIMER5/6 do not have DMAS bit (DMA request source selection).

16.1. Advanced timer (TIMERx, x=0, 7)

16.1.1. Overview

The advanced timer module (TIMER0, TIMER7) is a four-channel timer that supports both input capture and output compare. They can generate PWM signals to control motor or be used for power management applications. The advanced timer has a 16-bit counter that can be used as an unsigned counter.

In addition, the advanced timers can be programmed and be used for counting, their external events can be used to drive other timers.

Timer also includes a dead-time insertion module which is suitable for motor control applications.

Timers are completely independent with each other, but they may be synchronized to provide a larger timer with their counter value increasing in unison.

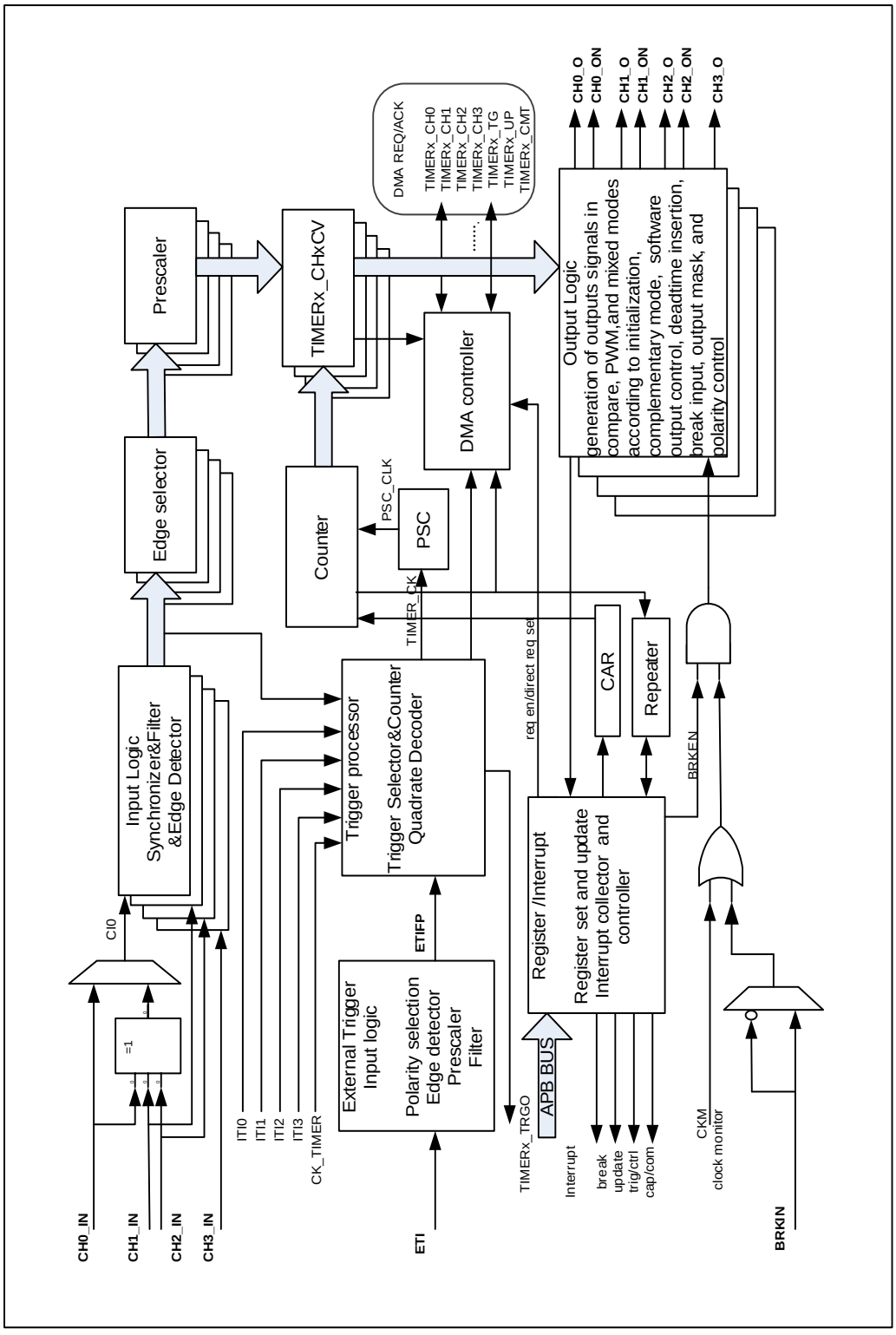
16.1.2. Characteristics

- Total channel num: 4.
- Counter width: 16 bits.
- Clock source of timer is selectable: internal clock, internal trigger, external input, external trigger.
- Multiple counter modes: up counting, down counting and center-aligned counting.
- Quadrature decoder: used for motion tracking and determination of both rotation direction and position.
- Hall sensor function: used for 3-phase motor control.
- Programmable prescaler: 16 bits. The factor can be changed ongoing.
- Each channel is user-configurable: input capture mode, output compare mode, programmable PWM mode and single pulse mode.
- Programmable dead time insertion.
- Auto reload function.
- Programmable counter repetition function.
- Break input.
- Interrupt output or DMA request: update event, trigger event, compare/capture event and break input.
- Daisy chaining of timer module allows a single timer to start multiple timers.
- Timer synchronization allows the selected timers to start counting on the same clock cycle.
- Timer master-slave management.

16.1.3. Block diagram

Figure 16-1. Advanced timer block diagram provides details of the internal configuration of the advanced timer.

Figure 16-1. Advanced timer block diagram



16.1.4. Function overview

Clock source configuration

The advanced timer has the capability of being clocked by either the CK_TIMER or an alternate clock source controlled by SMC (TIMERx_SMCFG bit [2:0]).

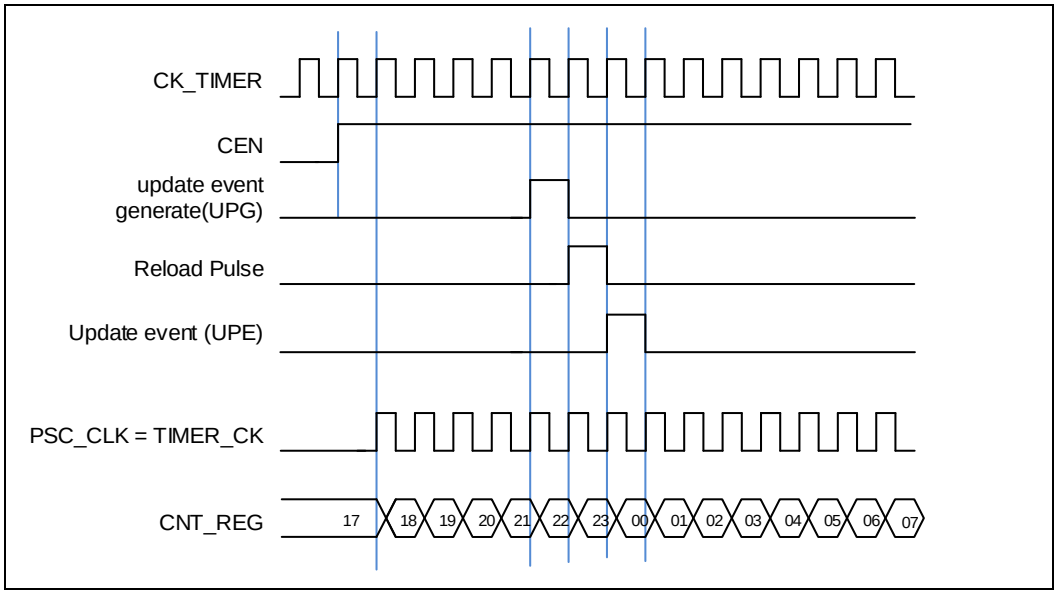
- SMC [2:0] == 3'b000. Internal clock CK_TIMER is selected as timer clock source which is from module RCU.

The default clock source is the CK_TIMER for driving the counter prescaler when the SMC [2:0] == 3'b000. When the CEN is set, the CK_TIMER will be divided by PSC value to generate PSC_CLK.

In this mode, the TIMER_CK, which drives counter's prescaler to count, is equal to CK_TIMER which is from RCU.

If the SMC [2:0] in the TIMERx_SMCFG register are setting to an available value including 0x1, 0x2, 0x3 and 0x7, the prescaler is clocked by other clock sources selected by the TRGS [2:0] in the TIMERx_SMCFG register, details as follows. When the SMC [2:0] bits are set to 0x4, 0x5 or 0x6, the internal clock CK_TIMER is the counter prescaler driving clock source.

Figure 16-2. Timing chart of internal clock divided by 1



- SMC [2:0] == 3'b111 (external clock mode 0). External input pin is selected as timer clock source

The TIMER_CK, which drives counter's prescaler to count, can be triggered by the event of rising or falling edge on the external pin TIMERx_CH0/TIMERx_CH1. This mode can be selected by setting SMC [2:0] to 0x7 and the TRGS [2:0] to 0x4, 0x5 or 0x6.

And, the counter prescaler can also be driven by rising edge on the internal trigger input pin ITI0/1/2/3. This mode can be selected by setting SMC [2:0] to 0x7 and the TRGS [2:0] to 0x0, 0x1, 0x2 or 0x3.

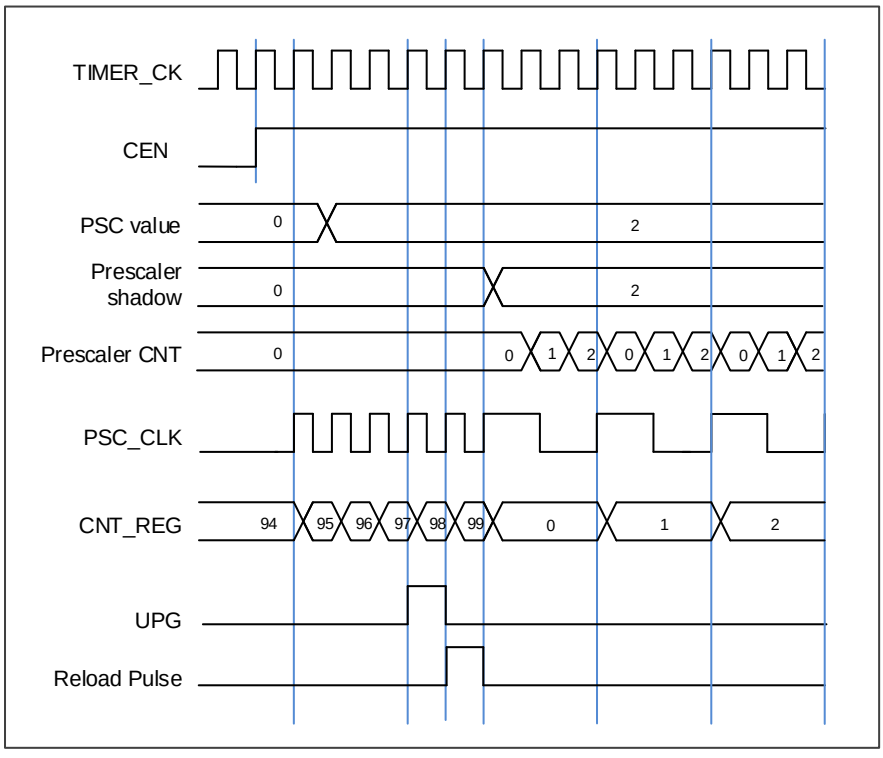
- SMC1== 1'b1 (external clock mode 1). External input ETI is selected as timer clock source (ETI)

The TIMER_CK, which drives counter's prescaler to count, can be triggered by the event of rising or falling edge on the external pin ETI. This mode can be selected by setting the SMC1 bit in the TIMERx_SMCFG register to 1. The other way to select the ETI signal as the clock source is to set the SMC [2:0] to 0x7 and the TRGS [2:0] to 0x7 respectively. Note that the ETI signal is derived from the ETI pin sampled by a digital filter. When the ETI signal is selected as clock source, the trigger controller including the edge detection circuitry will generate a clock pulse on each ETI signal rising edge to clock the counter prescaler.

Clock prescaler

The counter clock (PSC_CK) is obtained by the TIMER_CK through the prescaler, and the prescale factor can be configured from 1 to 65536 through the prescaler register (TIMERx_PSC). The new written prescaler value will not take effect until the next update event.

Figure 16-3. Timing chart of PSC value change from 0 to 2



Counter up counting

In this mode, the counter counts up continuously from 0 to the counter-reload value, which is defined in the TIMERx_CAR register, in a count-up direction. Once the counter reaches the counter reload value, the counter will start counting up from 0 again and an overflow event will be generated. In addition, the update events will be generated after (TIMERx_CREP+1) times of overflow events. The counting direction bit DIR in the TIMERx_CTL0 register should be set to 0 for the up counting mode.

Whenever, if the update event software trigger is enabled by setting the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to 0 and generates an update event.

If set the UPDIS bit in TIMERx_CTL0 register, the update event is disabled.

When an update event occurs, all the shadow registers (repetition counter, auto reload register, prescaler register) are updated.

Figure 16-4. Timing chart of up counting mode, PSC=0/2 show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 16-4. Timing chart of up counting mode, PSC=0/2

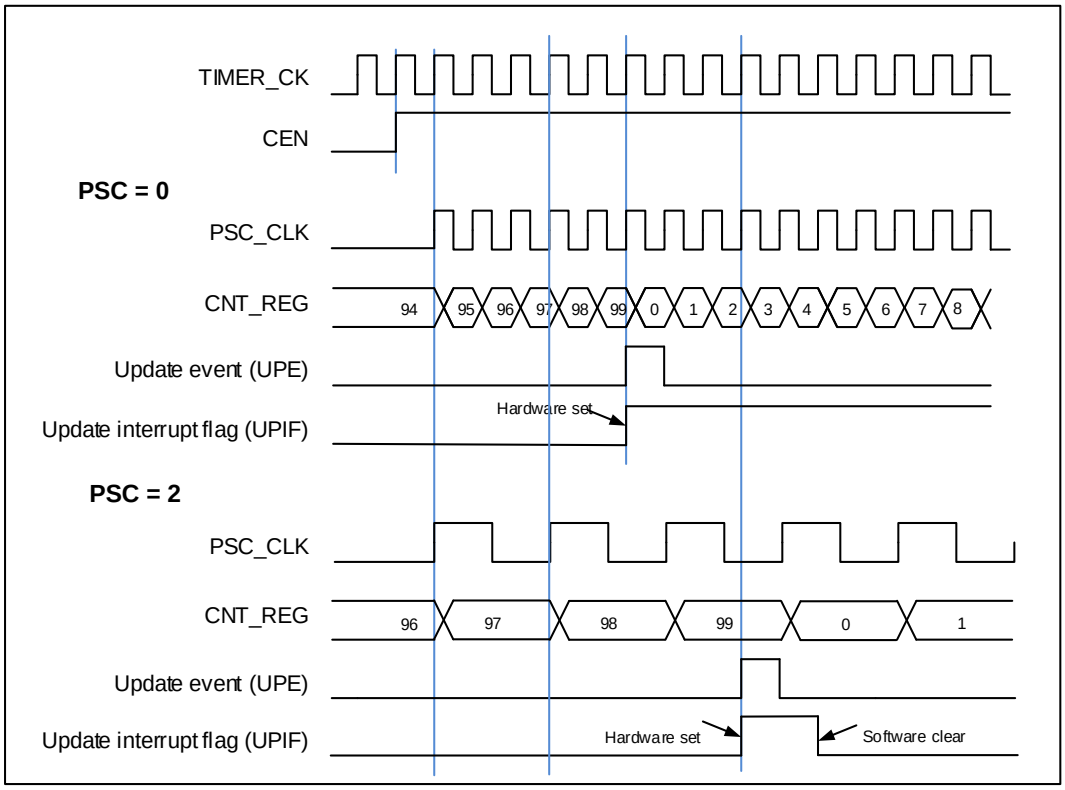
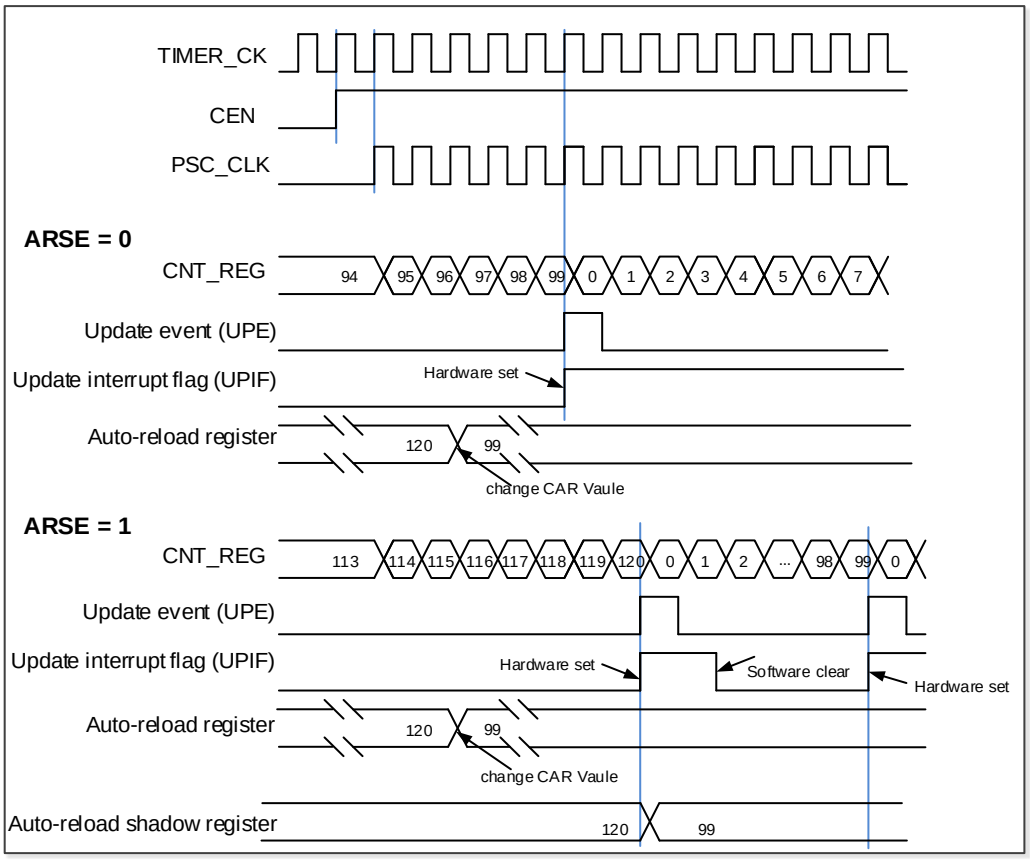


Figure 16-5. Timing chart of up counting mode, change TIMERx_CAR on the go



Counter down counting

In this mode, the counter counts down continuously from the counter-reload value, which is defined in the TIMERx_CAR register, to 0 in a count-down direction. Once the counter reaches to 0, the counter the counter will start counting down from the counter-reload value again and an underflow event will be generated. In addition, the update event will be generated after (TIMERx_CREP+1) times of underflow. The counting direction bit DIR in the TIMERx_CTL0 register should be set to 1 for the down-counting mode.

When the update event is set by the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to the counter-reload value and generates an update event.

If set the UPDIS bit in TIMERx_CTL0 register, the update event is disabled.

When an update event occurs, all the registers (repetition counter, auto reload register, prescaler register) are updated.

Figure 16-6. Timing chart of down counting mode, PSC=0/2 show some examples of the

counter behavior in different clock frequencies when TIMERx_CAR=0x99.

Figure 16-6. Timing chart of down counting mode, PSC=0/2

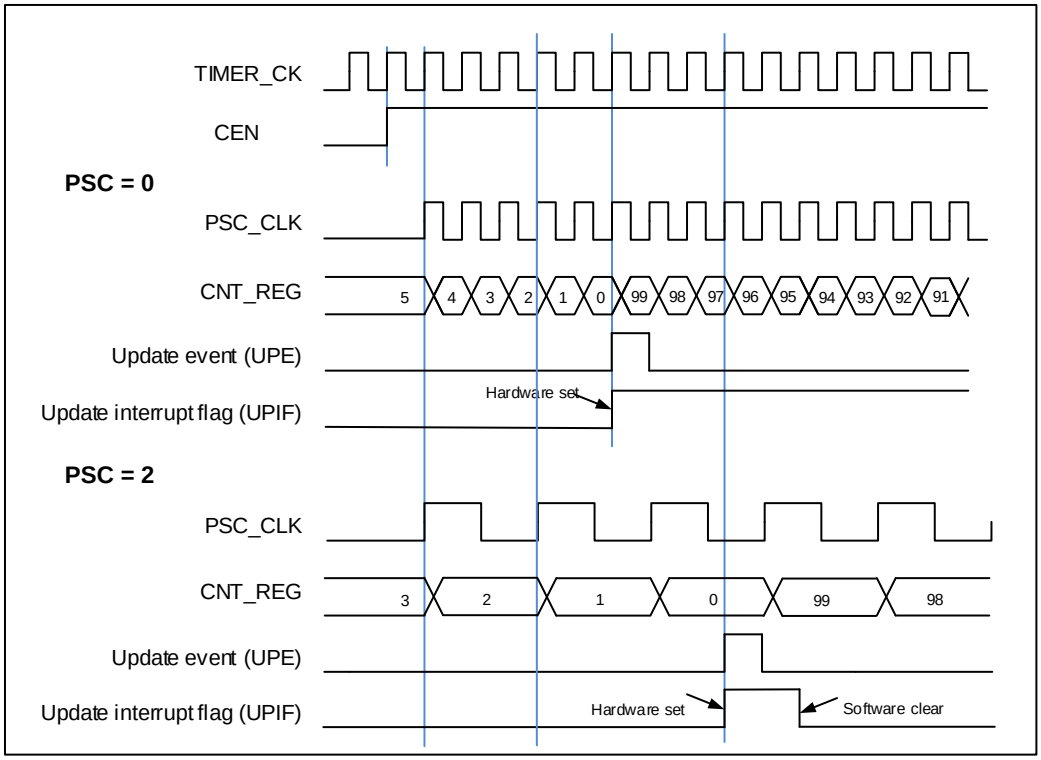
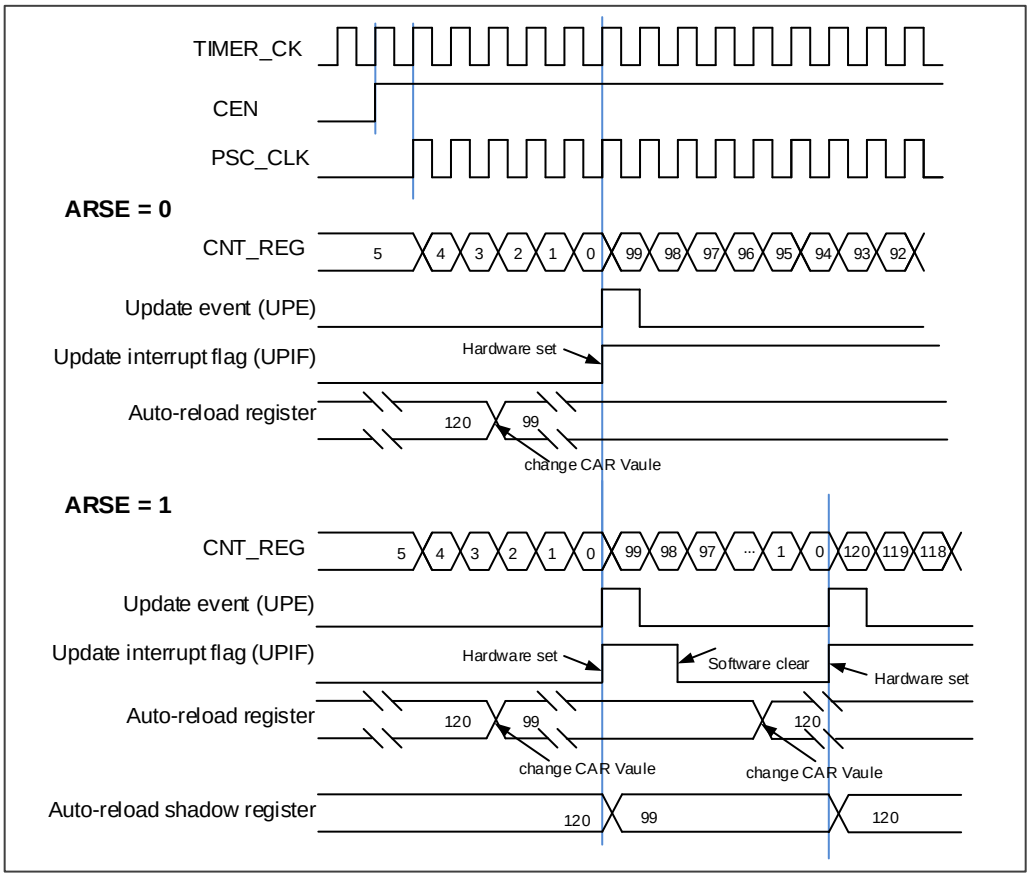


Figure 16-7. Timing chart of down counting mode, change TIMERx_CAR on the go



Counter center-aligned counting

In this mode, the counter counts up from 0 to the counter-reload value and then counts down to 0 alternatively. The Timer module generates an overflow event when the counter counts to the counter-reload value subtract 1 in the up-counting direction and generates an underflow event when the counter counts to 1 in the down-counting direction. The counting direction bit DIR in the TIMERx_CTL0 register is read-only and indicates the counting direction when in the center-aligned mode.

Setting the UPG bit in the TIMERx_SWEVG register will initialize the counter value to 0 and generates an update event irrespective of whether the counter is counting up or down in the center-align counting mode.

The UPIF bit in the TIMERx_INTF register can be set to 1 either when an underflow event or an overflow event occurs. While the CHxIF bit is associated with the value of CAM in TIMERx_CTL0. The details refer to ***Figure 16-8. Center-aligned counter timechart.***

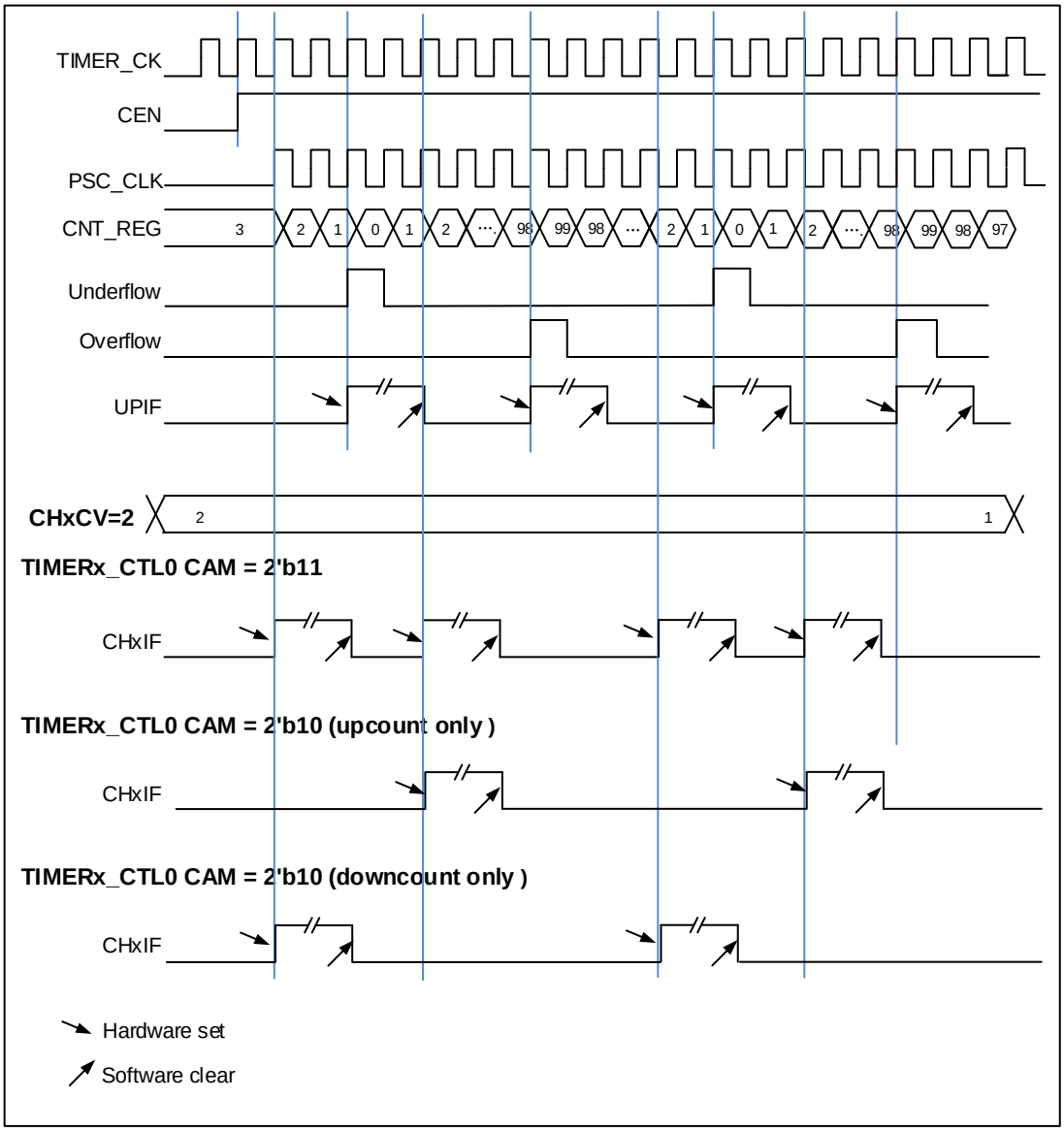
If set the UPDIS bit in the TIMERx_CTL0 register, the update event is disabled.

When an update event occurs, all the shadow registers (repetition counter, auto-reload register, prescaler register) are updated.

Figure 16-8. Center-aligned counter timechart show some examples of the counter

behavior when TIMERx_CAR=0x99. TIMERx_PSC=0x0

Figure 16-8. Center-aligned counter timechart



Update event (from overflow/underflow) rate configuration

The rate of update events generation (from overflow and underflow events) can be configured by the TIMERx_CREP register. Counter repetition is used to generator update event or updates the timer registers only after a given number (N+1) of cycles of the counter, where N is CREP in TIMERx_CREP register. The repetition counter is decremented at each counter overflow (does not exist in down counting mode) and underflow (does not exist in up counting mode).

Setting the UPG bit in the TIMERx_SWEVG register will reload the content of CREP in TIMERx_CREP register and generator an update event.

The new written CREP value will not take effect until the next update event. When the value of CREP is odd, and the counter is counting in center-aligned mode, the update event is

generated (on overflow or underflow) depending on when the written CREP value takes effect. If an update event is generated by software after writing an odd number to CREP, the update events will be generated on the underflow. If the next update event occurs on overflow after writing an odd number to CREP, then the subsequent update events will be generated on the overflow.

Figure 16-9. Repetition timechart for center-aligned counter

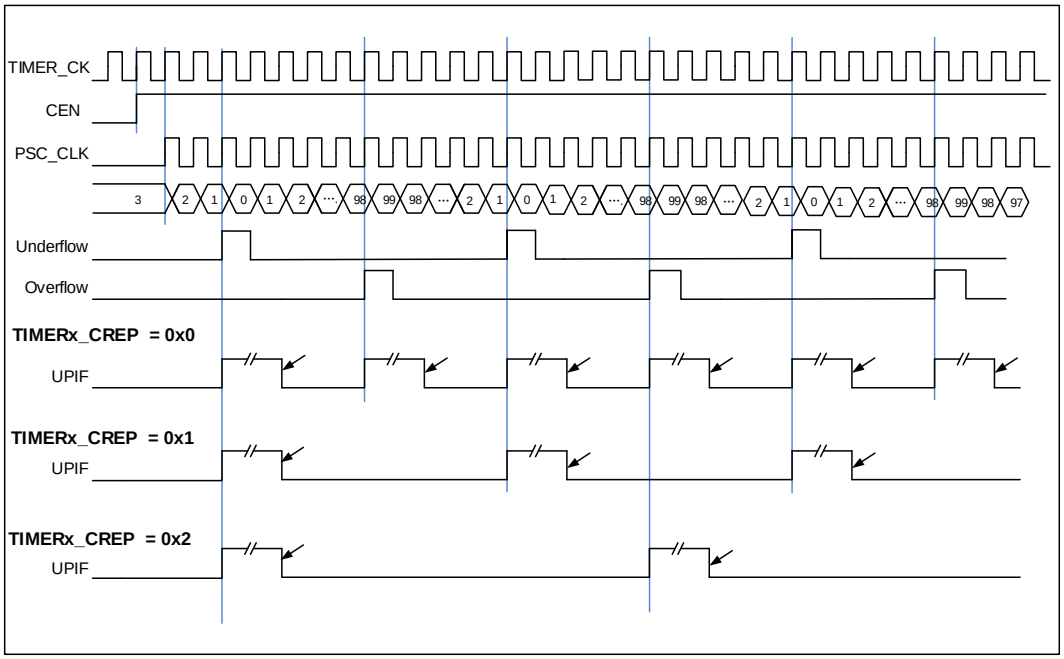


Figure 16-10. Repetition timechart for up-counter

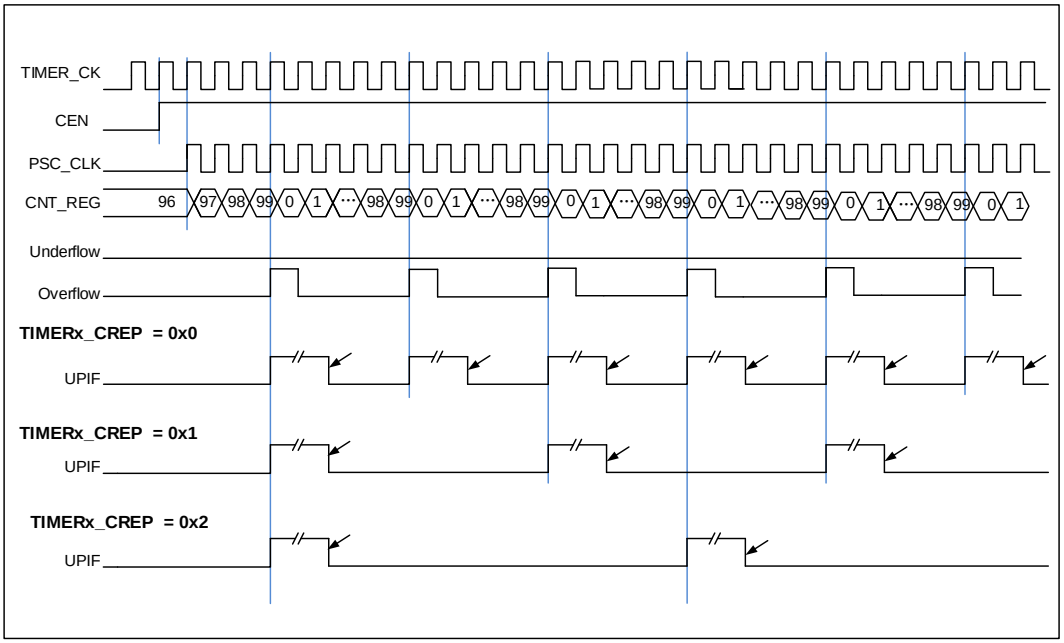
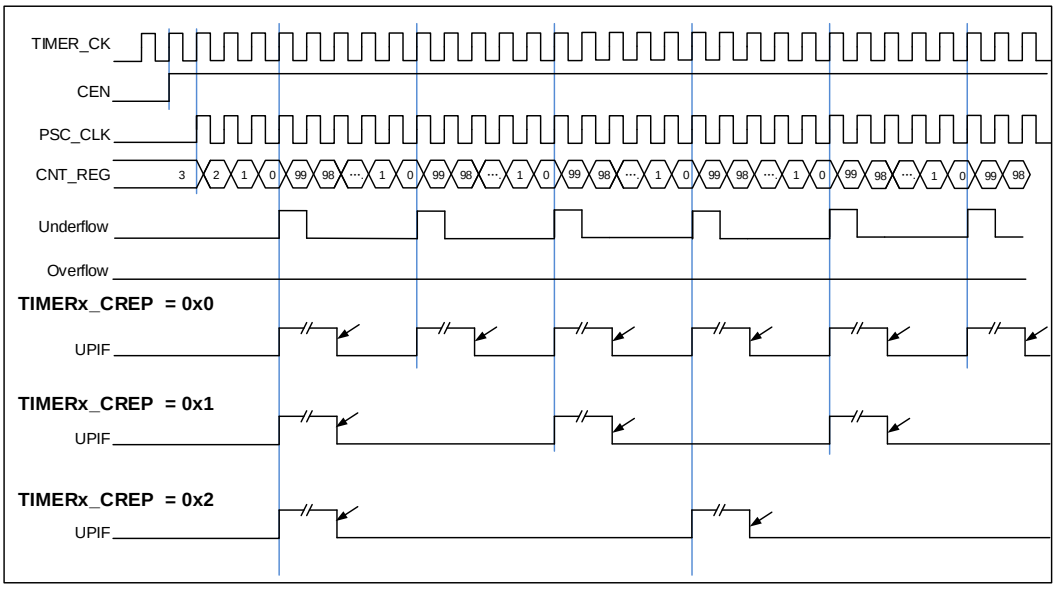


Figure 16-11. Repetition timechart for down-counter



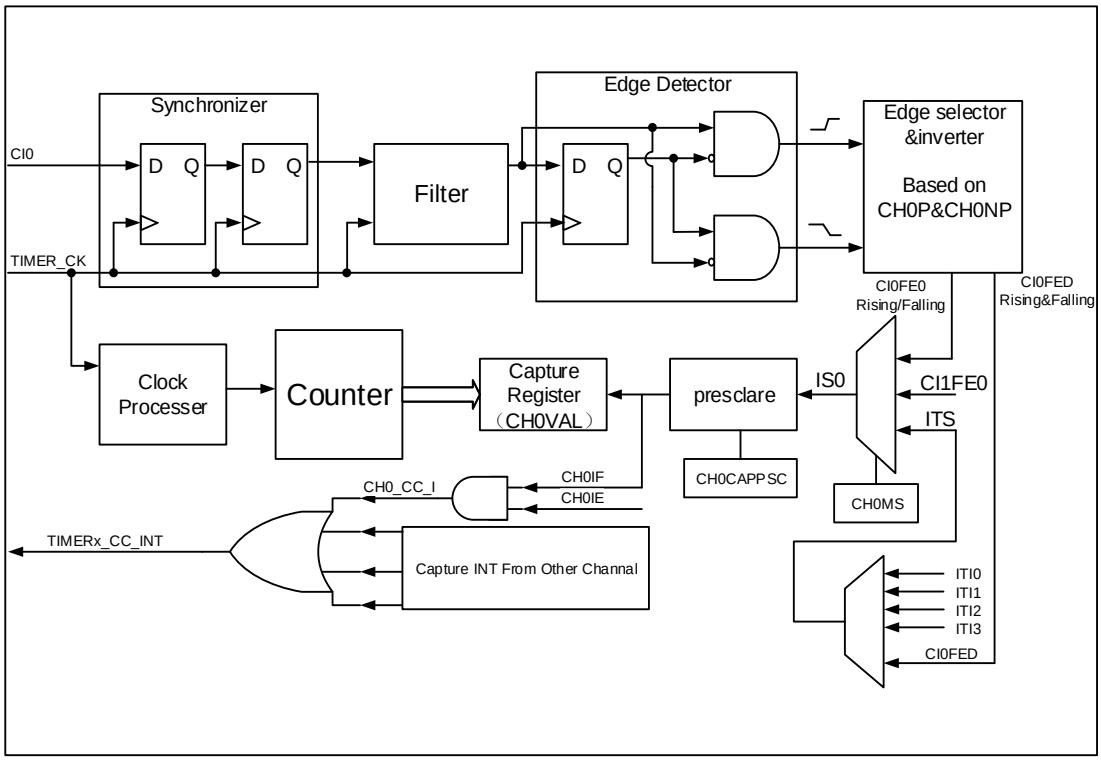
Input capture and output compare channels

The advanced timer has four independent channels which can be used as capture inputs or compare match outputs. Each channel is built around a channel capture compare register including an input stage, channel controller and an output stage.

- **Channel input capture function**

Channel input capture function allows the channel to perform measurements such as pulse timing, frequency, period, duty cycle and so on. The input stage consists of a digital filter, a channel polarity selection, edge detection and a channel prescaler. When a selected edge occurs on the channel input, the current value of the counter is captured into the TIMERx_CHxCV register, at the same time the CHxIF bit is set and the channel interrupt is generated if enabled by CHxIE = 1.

Figure 16-12. Channel input capture principle



One of channels' input signals (CIx) can be chosen from the TIMERx_CHx signal or the Excusive-OR function of the TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 signals. First, the channel input signal (CIx) is synchronized to TIMER_CK domain, and then sampled by a digital filter to generate a filtered input signal. Then through the edge detector, the rising and falling edge are detected. You can select one of them by CHxP. One more selector is for the other channel and trig, controlled by CHxMS. The IC_prescaler make several the input event generate one effective capture event. On the capture event, TIMERx_CHxCV will restore the value of counter.

So the process can be divided to several steps as below:

Step1: Filter configuration. (CHxCAPFLT in TIMERx_CHCTL0)

Based on the input signal and requested signal quality, configure compatible CHxCAPFLT.

Step2: Edge selection. (CHxP/CHxNP in TIMERx_CHCTL2)

Rising or falling edge, choose one by CHxP/CHxNP.

Step3: Capture source selection. (CHxMS in TIMERx_CHCTL0)

As soon as you select one input capture source by CHxMS, you have set the channel to input mode (CHxMS! =0x0) and TIMERx_CHxCV cannot be written any more.

Step4: Interrupt enable. (CHxIE and CHxDEN in TIMERx_DMAINTEN)

Enable the related interrupt enable; you can got the interrupt and DMA request.

Step5: Capture enables. (CHxEN in TIMERx_CHCTL2)

Result: when you wanted input signal is got, TIMERx_CHxCV will be set by counter's value.

And CHxIF is asserted. If the CHxIF is high, the CHxOF will be asserted also. The interrupt and DMA request will be asserted based on the configuration of CHxIE and CHxDEN in TIMERx_DMAINTEN

Direct generation: if you want to generate a DMA request or Interrupt, you can set CHxG by software directly.

The channel input capture function can be also used for pulse width measurement from signals on the TIMERx_CHx pins. For example, PWM signal connect to CI0 input. Select channel 0 capture signals to CI0 by setting CH0MS to 2'b01 in the channel control register (TIMERx_CHCTL0) and set capture on rising edge. Select channel 1 capture signal to CI0 by setting CH1MS to 2'b10 in the channel control register (TIMERx_CHCTL0) and set capture on falling edge. The counter set to restart mode and restart on channel 0 rising edge. Then the TIMERX_CH0CV can measure the PWM period and the TIMERx_CH1CV can measure the PWM duty.

- **Channel output compare function**

In output compare mode, the TIMERx can generate timed pulses with programmable position, polarity, duration and frequency. When the counter matches the value in the TIMERx_CHxCV register of an output compare channel, the channel (n) output can be set, cleared, or toggled based on CHxCOMCTL. When the counter reaches the value in the TIMERx_CHxCV register, the CHxIF bit is set and the channel (n) interrupt is generated if CHxIE = 1. And the DMA request will be assert, if CxCDE=1.

So the process can be divided to several steps as below:
Step1: Clock Configuration. Such as clock source, clock prescaler and so on.

Step2: Compare mode configuration.
* Set the shadow enable mode by CHxCOMSEN
* Set the output mode (Set/Clear/Toggle) by CHxCOMCTL.
* Select the active high polarity by CHxP/CHxNP
* Enable the output by CHxEN

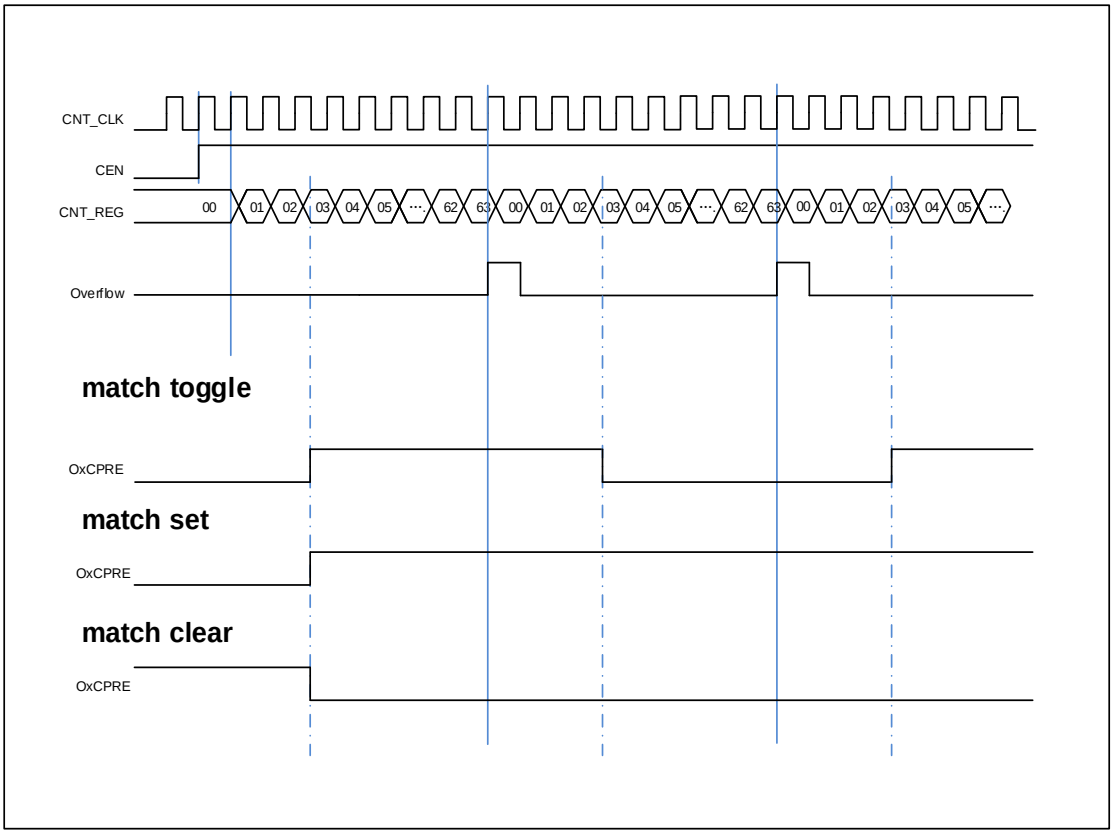
Step3: Interrupt/DMA-request enables configuration by CHxIE/CxCDE

Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV
About the TIMERx_CHxCV; you can change it on the go to meet the waveform you expected.

Step5: Start the counter by CEN.

The timechart below show the three compare modes toggle/set/clear. CAR=0x63, CHxVAL=0x3

Figure 16-13. Output-compare under three modes



Output PWM function

In the output PWM function (by setting the CHxCOMCTL bits to 3'b110 (PWM mode0) or to 3'b 111(PWM mode1), the channel can generate PWM waveform according to the TIMERx_CAR registers and TIMERx_CHxCV registers.

Based on the counter mode, we can also divide PWM into EAPWM (Edge aligned PWM) and CAPWM (Centre aligned PWM).

The EAPWM period is determined by TIMERx_CAR and duty cycle is determined by TIMERx_CHxCV. ***Figure 16-14. EAPWM timechart*** shows the EAPWM output and interrupts waveform.

The CAPWM period is determined by 2*TIMERx_CAR, and duty cycle is by 2*TIMERx_CHxCV. ***Figure 16-15. CAPWM timechart*** shows the CAPWM output and interrupts waveform.

If TIMERx_CHxCV is greater than TIMERx_CAR, the output will be always active under PWM mode0 (CHxCOMCTL==3'b110).
And if TIMERx_CHxCV is equal to zero, the output will be always inactive under PWM mode0 (CHxCOMCTL==3'b110).

Figure 16-14. EAPWM timechart

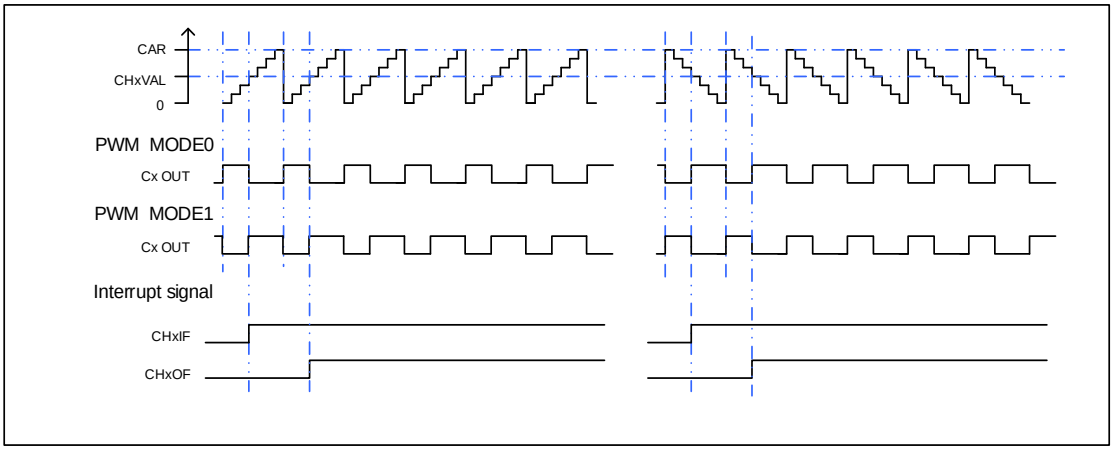
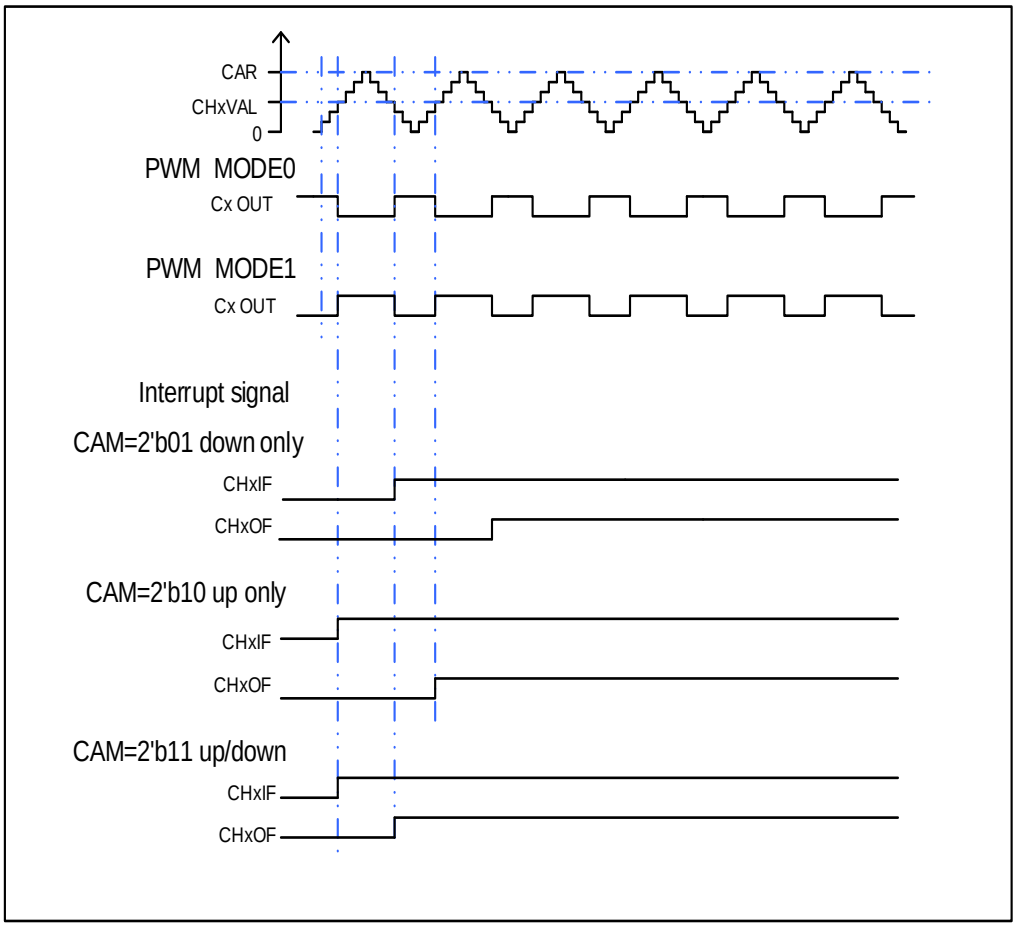


Figure 16-15. CAPWM timechart



Channel output prepare signal

When the TIMERx is used in the compare match output mode, the OxCPRE signal (Channel x Output prepare signal) is defined by setting the CHxCOMCTL filed. The OxCPRE signal has several types of output function. These include, keeping the original level by setting the CHxCOMCTL field to 0x00, set to 1 by setting the CHxCOMCTL field to 0x01, set to 0 by

setting the CHxCOMCTL field to 0x02 or signal toggle by setting the CHxCOMCTL field to 0x03 when the counter value matches the content of the TIMERx_CHxCV register.

The PWM mode 0 and PWM mode 1 outputs are also another kind of OxCPRE output which is setup by setting the CHxCOMCTL field to 0x06/0x07. In these modes, the OxCPRE signal level is changed according to the counting direction and the relationship between the counter value and the TIMERx_CHxCV content. With regard to a more detail description refer to the relative bit definition.

Another special function of the OxCPRE signal is a forced output which can be achieved by setting the CHxCOMCTL field to 0x04/0x05. Here the output can be forced to an inactive/active level irrespective of the comparison condition between the counter and the TIMERx_CHxCV values.

The OxCPRE signal can be forced to 0 when the ETIFE signal is derived from the external ETI pin and when it is set to a high level by setting the CHxCOMCEN bit to 1 in the TIMERx_CHCTL0 register. The OxCPRE signal will not return to its active level until the next update event occurs.

Channel output complementary PWM

Function of complementary is for a pair of CHx_O and CHx_ON. Those two output signals cannot be active at the same time. The TIMERx has 4 channels, but only the first three channels have this function. The complementary signals CHx_O and CHx_ON are controlled by a group of parameters: the CHxEN and CHxNEN bits in the TIMERx_CHCTL2 register and the POEN, ROS, IOS, ISOx and ISOxN bits in the TIMERx_CCHP and TIMERx_CTL1 registers. The outputs polarity is determined by CHxP and CHxNP bits in the TIMERx_CHCTL2 register.

Table 16-2. Complementary outputs controlled by parameters

<table>
<tr><th colspan="5">Complementary Parameters</th><th colspan="2">Output Status</th></tr>
<tr><th>POEN</th><th>ROS</th><th>IOS</th><th>CHxEN</th><th>CHxNEN</th><th>CHx_O</th><th>CHx_ON</th></tr>
<tr><td rowspan="5">0</td><td rowspan="5">0/1</td><td rowspan="3">0</td><td rowspan="2">0</td><td>0</td><td colspan="2">CHx_O / CHx_ON = LOW
CHx_O / CHx_ON output disable(1).</td></tr>
<tr><td>1</td><td colspan="2" rowspan="3">CHx_O/ CHx_ON output “off-state” (2):
the CHx_O/ CHx_ON output inactive level firstly: CHx_O = CHxP, CHx_ON = CHxNP; If the clock for deadtime generator is present, after a deadtime: CHx_O = ISOx, CHx_ON = ISOxN. (3)</td></tr>
<tr><td rowspan="2">1</td><td>0</td></tr>
<tr><td>1</td><td>1</td></tr>
<tr><td>1</td><td>x</td><td>x</td><td colspan="2">CHx_O/ CHx_ON output “off-state”:
the CHx_O/ CHx_ON output inactive level firstly: CHx_O = CHxP, CHx_ON = CHxNP; If the clock for deadtime generator is present, after a deadtime: CHx_O = ISOx, CHx_ON = ISOxN.</td></tr>
<tr><td rowspan="8">1</td><td rowspan="4">0</td><td rowspan="6">0/1</td><td rowspan="2">0</td><td>0</td><td colspan="2">CHx_O/CHx_ON = LOW
CHx_O/CHx_ON output disable.</td></tr>
<tr><td>1</td><td>CHx_O = LOW
CHx_O output disable.</td><td>CHx_ON =OxCPRE⊕(4)CHxNP
CHx_ON output enable.</td></tr>
<tr><td rowspan="2">1</td><td>0</td><td>CHx_O=OxCPRE⊕CHxP
CHx_O output enable.</td><td>CHx_ON = LOW
CHx_ON output disable.</td></tr>
<tr><td>1</td><td>CHx_O=OxCPRE⊕CHxP
CHx_O output enable.</td><td>CHx_ON =(!OxCPRE)(5)⊕ CHxNP.
CHx_ON output enable.</td></tr>
<tr><td rowspan="4">1</td><td rowspan="2">0</td><td>0</td><td>CHx_O = CHxP
CHx_O output “off-state”.</td><td>CHx_ON = CHxNP
CHx_ON output “off-state”.</td></tr>
<tr><td>1</td><td>CHx_O = CHxP
CHx_O output “off-state”</td><td>CHx_ON =OxCPRE⊕CHxNP
CHx_ON output enable</td></tr>
<tr><td rowspan="2">1</td><td>0</td><td>CHx_O=OxCPRE⊕CHxP
CHx_O output enable</td><td>CHx_ON = CHxNP
CHx_ON output “off-state”.</td></tr>
<tr><td>1</td><td>CHx_O=OxCPRE⊕CHxP
CHx_O output enable</td><td>CHx_ON =(!OxCPRE)⊕ CHxNP
CHx_ON output enable.</td></tr>
</table>

Note:

(1) output disable: the CHx_O / CHx_ON are disconnected to corresponding pins, the pin is floating with GPIO pull up/down setting which will be Hi-Z if no pull.

(2) “off-state”: CHx_O / CHx_ON output with inactive state (e.g., CHx_O = 0⊕CHxP = CHxP).

(3) See Break mode section for more details.

(4) ⊕: Xor calculate.

(5) (!OxCPRE)：the complementary output of the OxCPRE signal.

Insertion dead time for complementary PWM

The dead time insertion is enabled when both CHxEN and CHxNEN are 1'b1, and set POEN is also necessary. The field named DTCFG defines the dead time delay that can be used for all channels expect for channel 3. The detail about the delay time, refer to the register TIMERx_CCHP.

The dead time delay insertion ensures that no two complementary signals drive the active state at the same time.

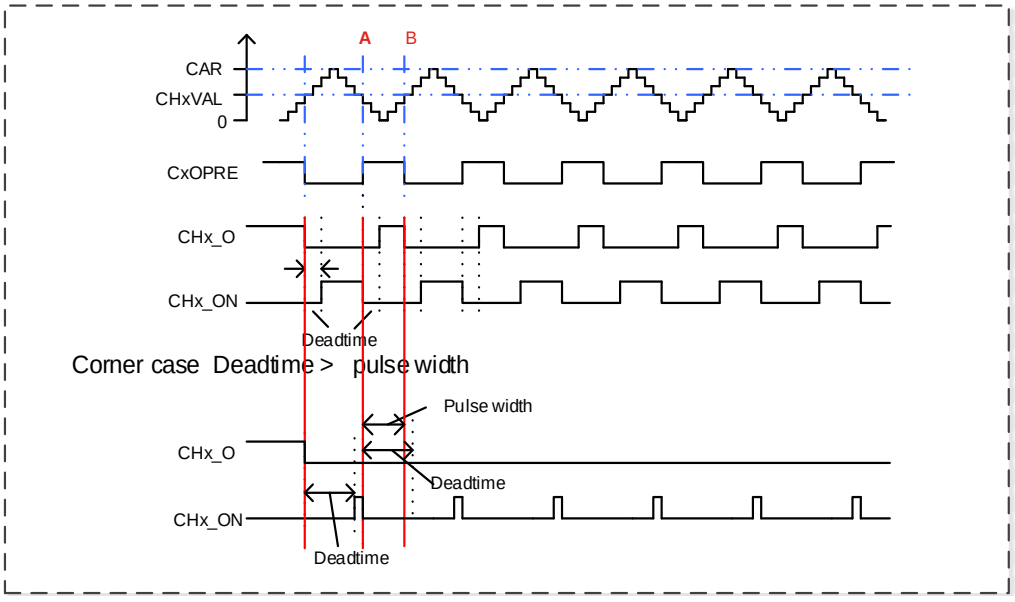
When the channel (x) match (TIMERx counter = CHxVAL) occurs, OxCPRE will be toggled because under PWM0 mode. At point A in the ***Figure 16-16. Complementary output with dead-time insertion*** CHx_O signal remains at the low value until the end of the deadtime delay, while CHx_ON will be cleared at once. Similarly, At point B when counter match (counter = CHxVAL) occurs again, OxCPRE is cleared, CHx_O signal will be cleared at once, while CHx_ON signal remains at the low value until the end of the dead time delay.

Sometimes, we can see corner cases about the dead time insertion. For example:

The dead time delay is greater than or equal to the CHx_O duty cycle, then the CHx_O signal is always the inactive value. (As show in the ***Figure 16-16. Complementary output with dead-time insertion***)

- The dead time delay is greater than or equal to the CHx_ON duty cycle, then the CHx_ON signal is always the inactive value.

Figure 16-16. Complementary output with dead-time insertion



Break mode

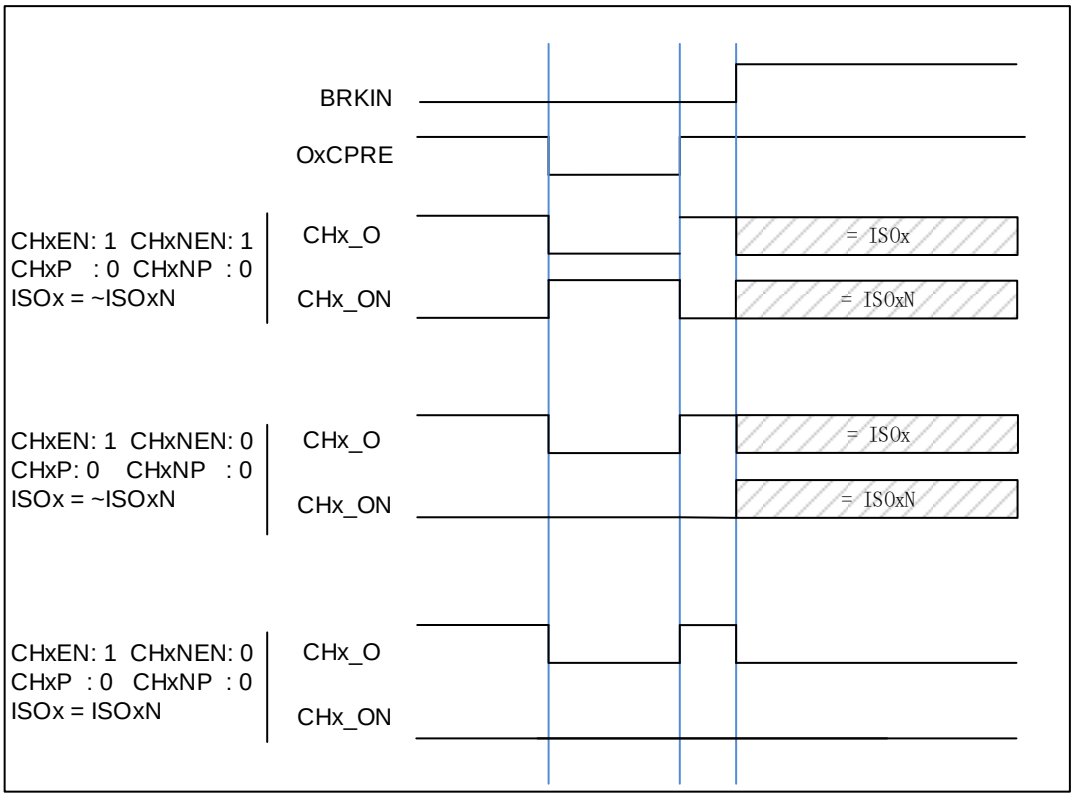
In this mode, the output CHx_O and CHx_ON are controlled by the POEN, IOS and ROS bits in the TIMERx_CCHP register, ISOx and ISOxN bits in the TIMERx_CTL1 register and cannot be set both to active level when break occurs. The break sources are input break pin and

HXTAL stuck event by Clock Monitor (CKM) in RCU. The break function enabled by setting the BRKEN bit in the TIMERx_CCHP register. The break input polarity is setting by the BRKP bit in TIMERx_CCHP.

When a break occurs, the POEN bit is cleared asynchronously, the output CHx_O and CHx_ON are driven with the level programmed in the ISOx bit and ISOxN in the TIMERx_CTL1 register as soon as POEN is 0. If IOS is 0 then the timer releases the enable output else the enable output remains high. The complementary outputs are first put in reset state, and then the dead-time generator is reactivated in order to drive the outputs with the level programmed in the ISOx and ISOxN bits after a dead-time.

When a break occurs, the BRKIF bit in the TIMERx_INTF register is set. If BRKIE is 1, an interrupt generated.

Figure 16-17. Output behavior in response to a break(The break high active)



Quadrature decoder

The quadrature decoder function uses two quadrature inputs CI0FE0 and CI1FE1 derived from the TIMERx_CH0 and TIMERx_CH1 pins respectively to interact to control the counter value. The DIR bit is modified during each input source transition. The counter can be changed by the edges of CI0FE0 only, CI1FE1 only or both CI0FE0 and CI1FE1, the selection mode by setting the SMC[2:0] to 0x01, 0x02 or 0x03. The mechanism for changing the counter direction is shown in ***Table 16-3. Counting direction in different quadrature decoder mode***. The quadrature decoder can be regarded as an external clock with a

directional selection. This means that the counter counts continuously in the interval between 0 and the counter-period value. Therefore, TIMERx_CAR register must be configured before the counter starts to count.

Table 16-3. Counting direction in different quadrature decoder mode

Counting mode	Level	CI0FE0		CI1FE1	
		Rising	Falling	Rising	Falling
Quadrature decoder mode 0 SMC[2:0]=3'b001	CI1FE1=1	Down	Up	-	-
	CI1FE1=0	Up	Down	-	-
Quadrature decoder mode 1 SMC [2:0]=3'b010	CI0FE0=1	-	-	Up	Down
	CI0FE0=0	-	-	Down	Up
Quadrature decoder mode 2 SMC [2:0]=3'b011	CI1FE1=1	Down	Up	X	X
	CI1FE1=0	Up	Down	X	X
	CI0FE0=1	X	X	Up	Down
	CI0FE0=0	X	X	Down	Up

Note:"-" means "no counting", "X" means impossible. "0" means "low level", "1" means "high level".

Figure 16-18. Counter behavior with CI0FE0 polarity non-inverted in mode 2

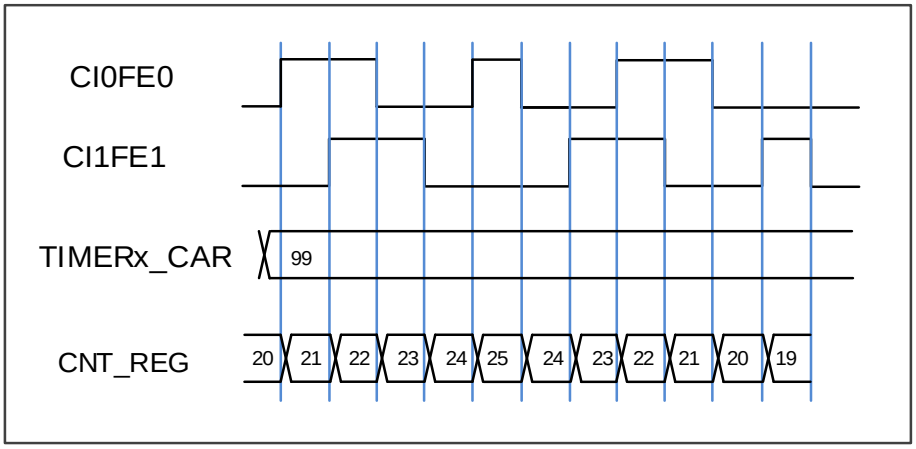
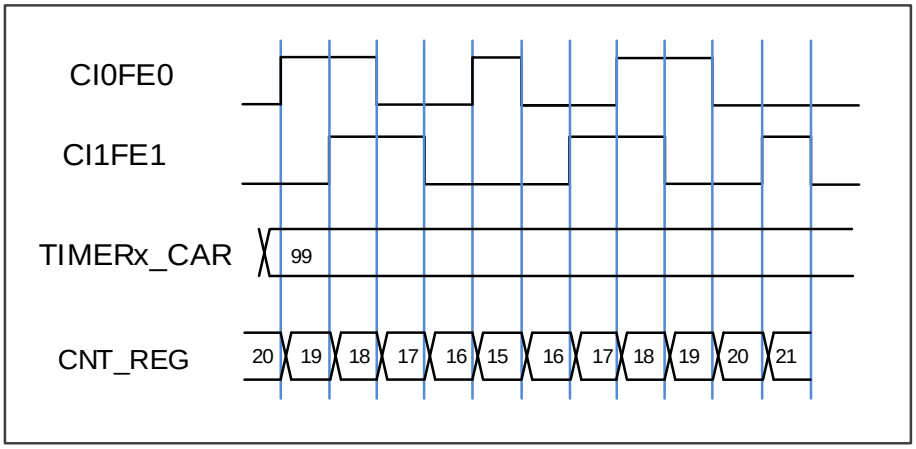


Figure 16-19. Counter behavior with CI0FE0 polarity inverted in mode 2



Hall sensor function

Hall sensor is generally used to control BLDC Motor; the timers can support this function.

Figure 16-20. Hall sensor is used to BLDC motor show how to connect. And we can see we need two timers. First TIMER_in (Advanced/General L0 TIMER) should accept three HALL sensor signals.

Each of the three input of HALL sensors provides a pulse that applied to an input capture pin, can then be analyzed and both speed and position can be deduced.

By the internal connection such as TRGO-ITIx, TIMER_in and TIMER_out can be connected. TIMER_out will generate PWM signal to control BLDC motor's speed based on the ITRx. Then, the feedback circuit is finished, also you change configuration to fit your request.

About the TIMER_in, it need have input XOR function, so you can choose from Advanced/General L0 TIMER.

And TIMER_out need have functions of complementary and Dead-time, so only advanced timer can be chosen. Else, based on the timers' internal connection relationship, pair's timers can be selected. For example:

TIMER_in (TIMER0) -> TIMER_out (TIMER7 ITI0)

TIMER_in (TIMER1) -> TIMER_out (TIMER0 ITI1)

And so on.

After getting appropriate timers combination, and wire connection, we need to configure timers. Some key settings include:

- Enable XOR by setting TI0S, then, each of input signal change will make the CI0 toggle. CH0VAL will record the value of counter at that moment.
- Enable ITIx connected to commutation function directly by setting CCUC and CCSE.
- Configuration PWM parameter based on your request.

Figure 16-20. Hall sensor is used to BLDC motor

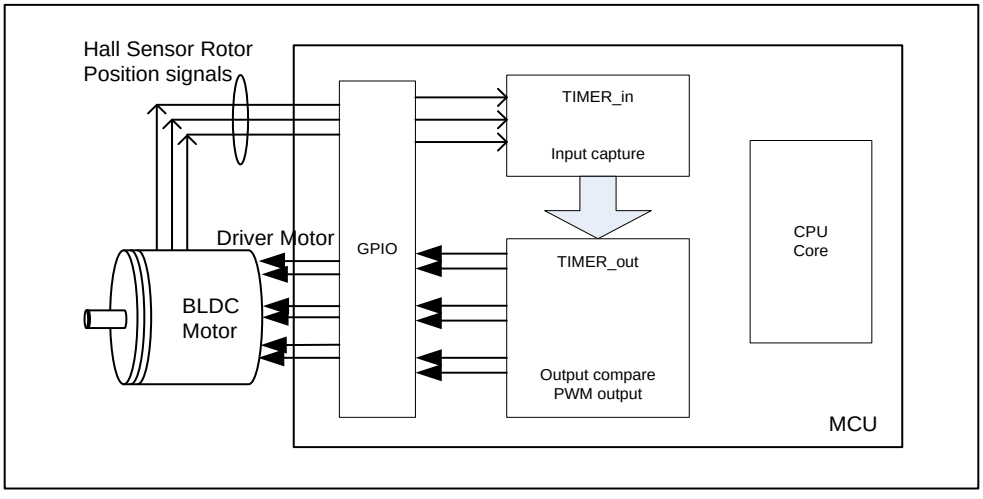
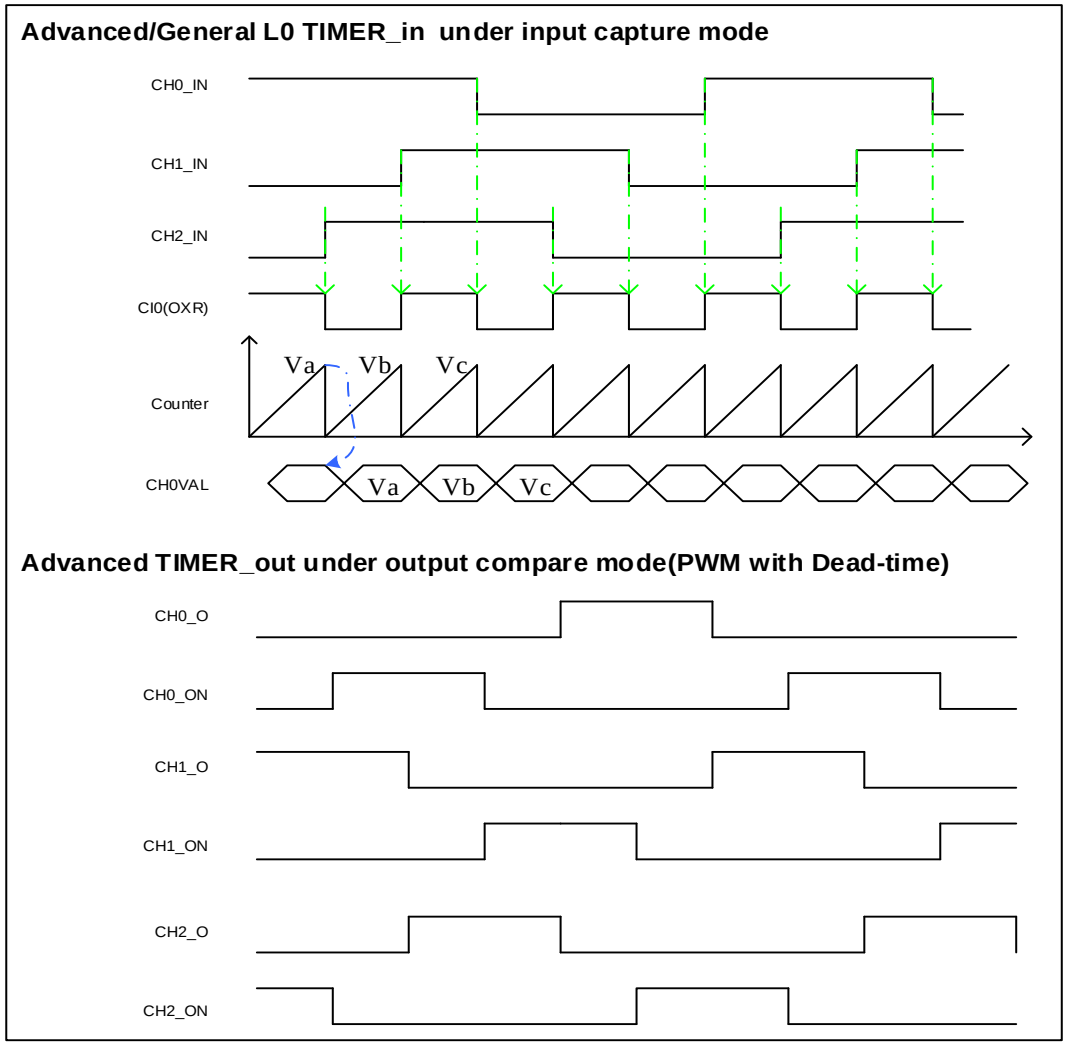


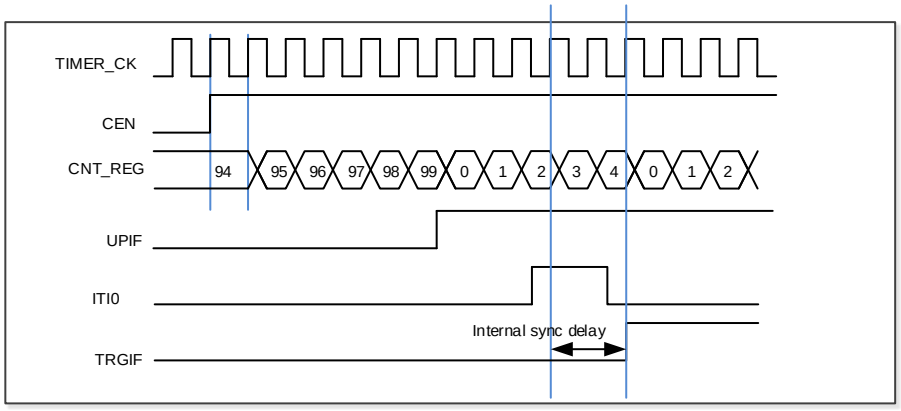
Figure 16-21. Hall sensor timing between two timers

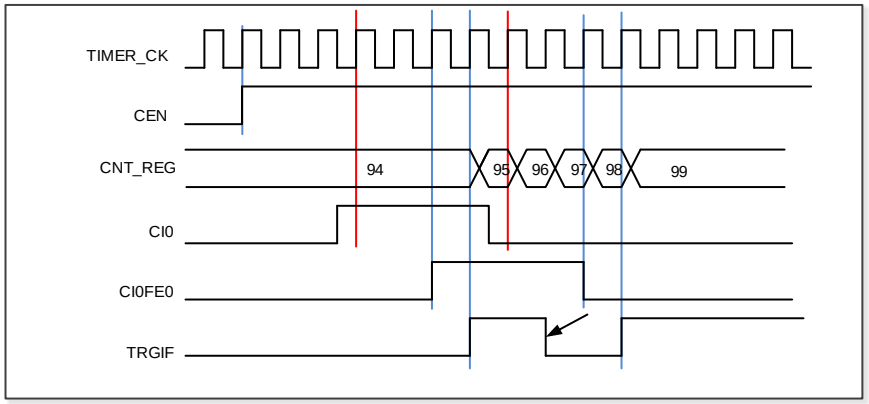
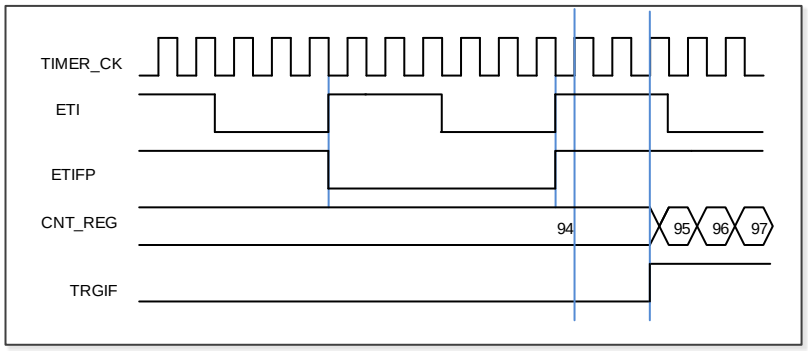


Master-slave management

The TIMERx can be synchronized with a trigger in several modes including the restart mode, the pause mode and the event mode which is selected by the SMC[2:0] in the TIMERx_SMCFG register. The trigger input of these modes can be selected by the TRGS [2:0] in the TIMERx_SMCFG register.

Table 16-4. Examples of slave mode

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
LIST	SMC[2:0] 3'b100 (restart mode) 3'b101 (pause mode) 3'b110 (event mode)	TRGS[2:0] 000: ITI0 001: ITI1 010: ITI2 011: ITI3 100: CI0F_ED 101: CI0FE0 110: CI1FE1 111: ETIFP	If you choose the CI0FE0 or CI1FE1, configure the CHxP and CHxNP for the polarity selection and inversion. If you choose the ETIF, configure the ETP for polarity selection and inversion.	For the ITIx no filter and prescaler can be used. For the CIx, configure Filter by CHxCAPFLT, no prescaler can be used. For the ETIF, configure Filter by ETFC and Prescaler by ETPSC.
Exam1	Restart mode The counter can be clear and restart when a rising trigger input.	TRGS[2:0]=3'b000 ITI0 is the selection.	- For ITI0, no polarity selector can be used.	- For the ITI0, no filter and prescaler can be used.
	Figure 16-22. Restart mode 			
Exam2	Pause mode The counter can be paused when the trigger input is low.	TRGS[2:0]=3'b101 CI0FE0 is the selection.	TI0S=0.（Non-xor） [CH0NP==0, CH0P==0] no inverted. Capture will be sensitive to the rising edge only.	Filter is bypass in this example.

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
	Figure 16-23. Pause mode 			
Exam3	Event mode The counter will start to count when a rising trigger input.	TRGS[2:0]=3'b111 ETIF is the selection.	ETP = 0 no polarity change.	ETPSC = 1, divided by 2. ETFC = 0 , no filter
	Figure 16-24. Event mode 			

Single pulse mode

Single pulse mode is opposite to the repetitive mode, which can be enabled by setting SPM in TIMERx_CTL0. When you set SPM, the counter will be clear and stop when the next update event. In order to get pulse waveform, you can set the TIMERx to PWM mode or compare by CHxCOMCTL.

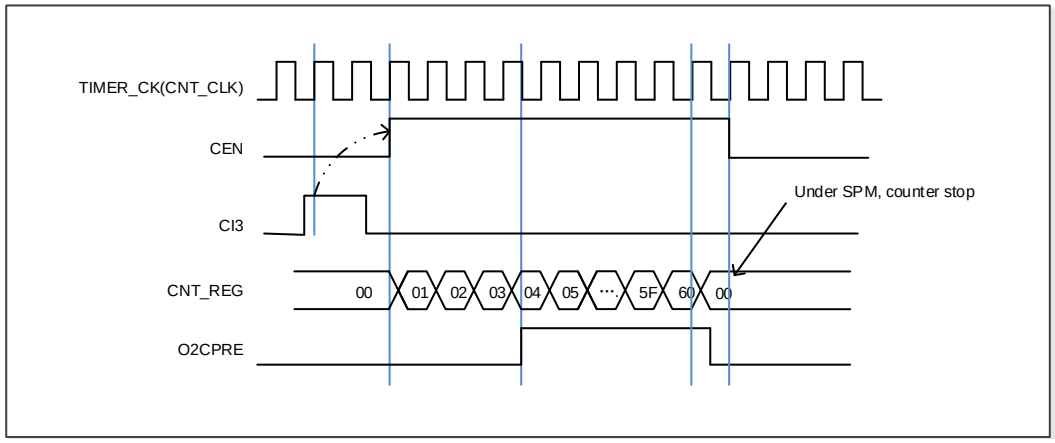
Once the timer is set to operate in the single pulse mode, it is not necessary to set the timer enable bit CEN in the TIMERx_CTL0 register to 1 to enable the counter. The trigger to generate a pulse can be sourced from the trigger signals edge or by setting the CEN bit to 1 using software. Setting the CEN bit to 1 or a trigger from the trigger signals edge can generate a pulse and then keep the CEN bit at a high state until the update event occurs or the CEN bit is written to 0 by software. If the CEN bit is cleared to 0 using software, the counter will be

stopped and its value held.

In the single pulse mode, the trigger active edge which sets the CEN bit to 1 will enable the counter. However, there exist several clock delays to perform the comparison result between the counter value and the TIMERx_CHxCV value. In order to reduce the delay to a minimum value, the user can set the CHxCOMFEN bit in each TIMERx_CHCTL0/1 register. After a trigger rising occurs in the single pulse mode, the OxCPRE signal will immediately be forced to the state which the OxCPRE signal will change to, as the compare match event occurs without taking the comparison result into account. The CHxCOMFEN bit is available only when the output channel is configured to operate in the PWM0 or PWM1 output mode and the trigger source is derived from the trigger signal.

Figure 16-25. Single pulse mode TIMERx_CHxCV = 4 TIMERx_CAR=99 shows an example.

Figure 16-25. Single pulse mode TIMERx_CHxCV = 4 TIMERx_CAR=99

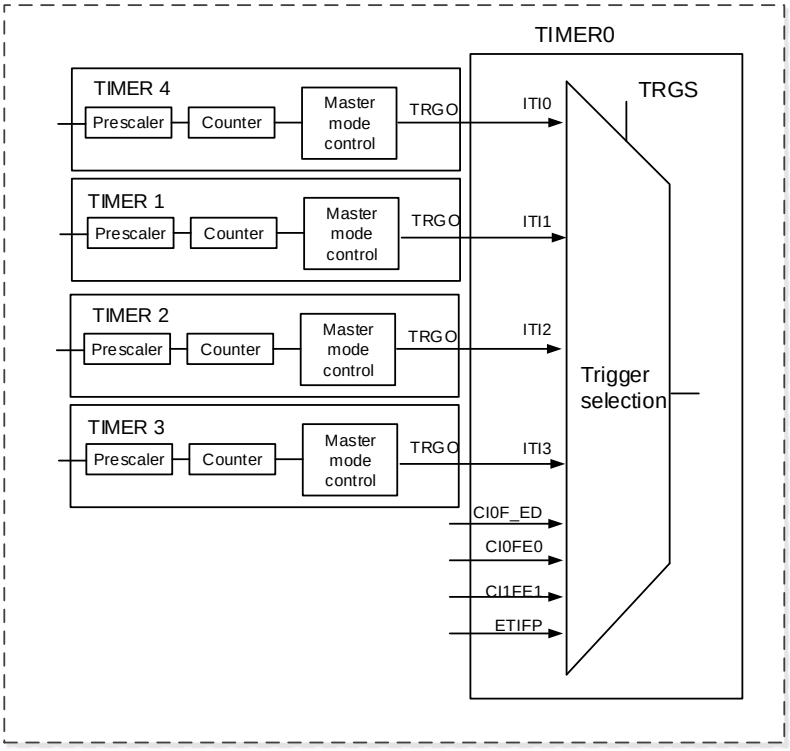


Timers interconnection

Timer can be configured as interconnection, that is, one timer which operate in the master mode outputs TRGO signal to control another timer which operate in the slave mode, TRGO include reset evevt, start evevt, update evevt, capture/compare pulse evevt, compare evevt. slave timer received the ITIx and performs the corresponding mode, include internal clock mode, quadrature decoder mode, restart mode, pause mode, event mode, external clock mode.

Figure 16-26. Timer0 master/slave mode timer example shows the timer0 trigger selection when it is configured in slave mode.

Figure 16-26. Timer0 master/slave mode timer example



Other interconnection examples:

- Timer 2 as prescaler for timer 0

We configure Timer2 as a prescaler for Timer 0. Refer to ***Figure 16-26. Timer0 master/slave mode timer example*** for connections. Do as bellow:

1. Configure Timer2 in master mode and select its update event (UPE) as trigger output (MMC=3'b010 in the TIMER2_CTL1 register). Then timer2 drives a periodic signal on each counter overflow.
2. Configure the Timer2 period (TIMER2_CAR registers).
3. Select the Timer0 input trigger source from Timer2(TRGS=3'b010 in the TIMERx_SMCFG register).
4. Configure Timer0 in external clock mode 0 (SMC=3'b111 in TIMERx_SMCFG register).
5. Start Timer0 by writing '1 in the CEN bit (TIMER0_CTL0 register).
6. Start Timer2 by writing '1 in the CEN bit (TIMER2_CTL0 register).

- Start timer 0 with timer 2's Enable/Update signal

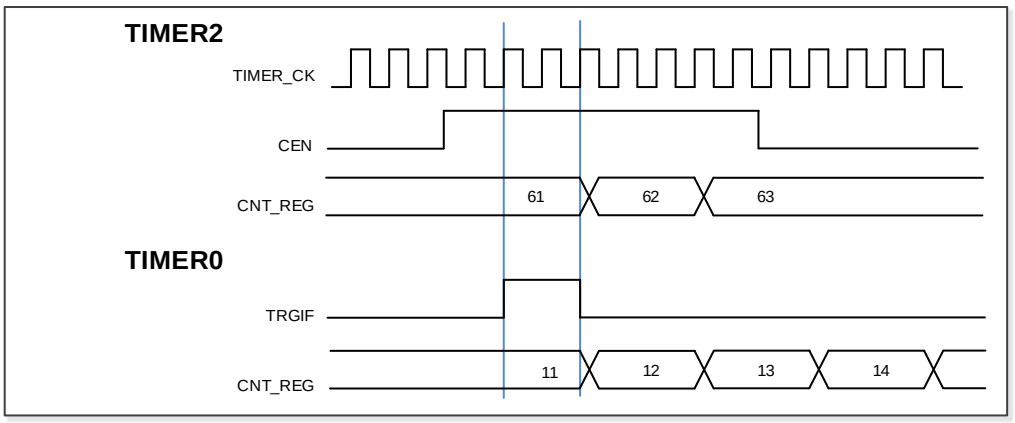
First, we enable Timer0 with the enable out of Timer2. Refer to ***Figure 16-27. Triggering TIMER0 with enable signal of TIMER2***. Timer0 starts counting from its current value on the divided internal clock after trigger by timer2 enable output.

When Timer0 receives the trigger signal its CEN bit is set and the counter counts until we disable timer0. Both counter clock frequencies are divided by 3 by the prescaler compared to

TIMER_CK (fCNT_CLK = fTIMER_CK /3). Do as follow:

1. Configure Timer2 master mode to send its enable signal as trigger output(MMC=3'b001 in the TIMER2_CTL1 register)
2. Configure Timer0 to select the input trigger from Timer2 (TRGS=3'b010 in the TIMERx_SMCFG register).
3. Configure Timer0 in event mode (SMC=3'b 110 in TIMERx_SMCFG register).
4. Start Timer2 by writing 1 in the CEN bit (TIMER2_CTL0 register).

Figure 16-27. Triggering TIMER0 with enable signal of TIMER2



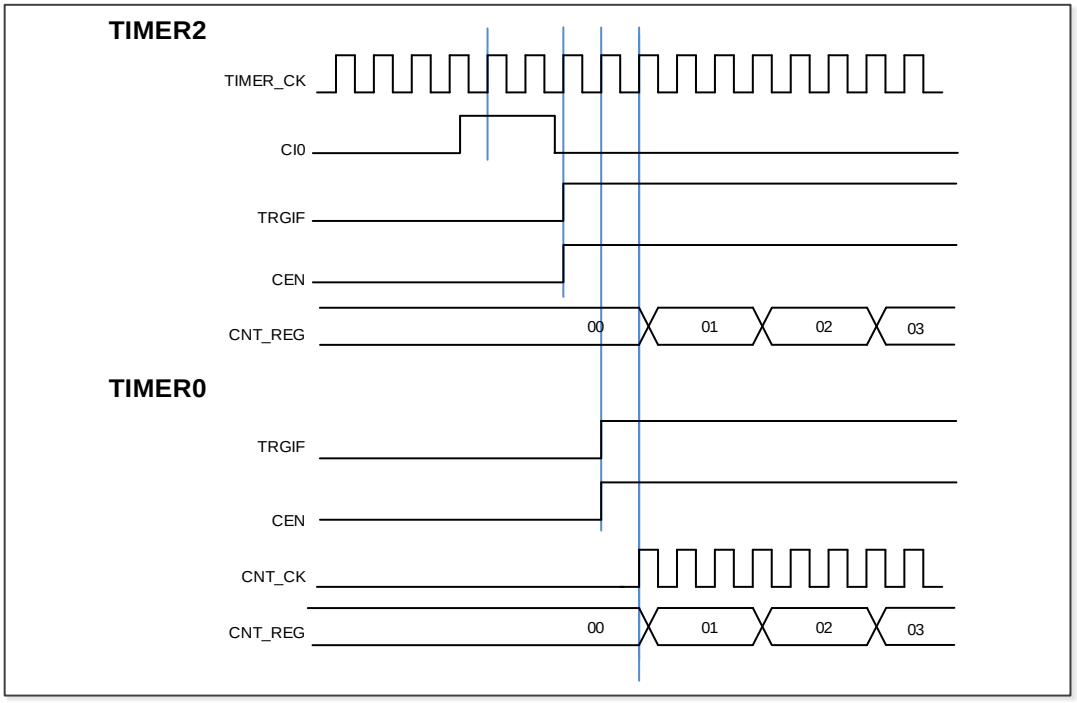
■ Using an external trigger to start 2 timers synchronously

We configure the start of Timer0 is triggered by the enable of Timer2, and Timer2 is triggered by its CI0 input rises edge. To ensure 2 timers start synchronously, Timer2 must be configured in Master/Slave mode. Do as follow:

1. Configure Timer2 slave mode to get the input trigger from CI0 (TRGS=3'b100 in the TIMER2_SMCFG register).
2. Configure Timer2 in event mode (SMC=3'b110 in the TIMER2_SMCFG register).
3. Configure the Timer2 in Master/Slave mode by writing MSM=1 (TIMER2_SMCFG register).
4. Configure Timer0 to get the input trigger from Timer2 (TRGS=3'b010 in the TIMERx_SMCFG register).
5. Configure Timer0 in event mode (SMC=3'b110 in the TIMER0_SMCFG register).

When a rising edge occurs on Timer2's CI0, two timer's counters start counting synchronously on the internal clock and both TRGIF flags are set.

Figure 16-28. Triggering TIMER0 and TIMER2 with TIMER2's CI0 input



Timer DMA mode

Timer's DMA mode is the function that configures timer's register by DMA module. The relative registers are TIMERx_DMACFG and TIMERx_DMATB. Of course, you have to enable a DMA request which will be asserted by some internal event. When the interrupt event was asserted, TIMERx will send a request to DMA, which is configured to M2P mode and PADDR is TIMERx_DMATB, then DMA will access the TIMERx_DMATB. In fact, register TIMERx_DMATB is only a buffer; timer will map the TIMERx_DMATB to an internal register, appointed by the field of DMATA in TIMERx_DMACFG. If the field of DMATC in TIMERx_DMACFG is 0(1 transfer), then the timer's DMA request is finished. While if TIMERx_DMATC is not 0, such as 3(4 transfers), then timer will send 3 more requests to DMA, and DMA will access timer's registers DMATA+0x4, DMATA+0x8, DMATA+0xc at the next 3 accesses to TIMERx_DMATB. In one word, one time DMA internal interrupt event assert, DMATC+1 times request will be send by TIMERx.

If one more time DMA request event coming, TIMERx will repeat the process as above.

Timer debug mode

When the Cortex®-M4 halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register is set to 1, the TIMERx counter stops.

16.1.5. TIMERx registers (x=0, 7)

TIMER0 base address: 0x4001 2C00
TIMER7 base address: 0x4001 3400

Control register 0 (TIMERx_CTL0)

Address offset: 0x00
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKDIV[1:0]		ARSE	CAM[1:0]		DIR	SPM	UPS	UPDIS	CEN
						rw		rw	rw		rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	CKDIV[1:0]	Clock division The CKDIV bits can be configured by software to specify division factor between the CK_TIMER and the dead-time and digital filter sample clock (DTS). 00: $f_{DTS}=f_{CK_TIMER}$ 01: $f_{DTS}= f_{CK_TIMER}$ /2 10: $f_{DTS}= f_{CK_TIMER}$ /4 11: Reserved
7	ARSE	Auto-reload shadow enable 0: The shadow register for TIMERx_CAR register is disabled 1: The shadow register for TIMERx_CAR register is enabled
6:5	CAM[1:0]	Counter aligns mode selection 00: No center-aligned mode (edge-aligned mode). The direction of the counter is specified by the DIR bit. 01: Center-aligned and counting down assert mode. The counter counts under center-aligned and channel is configured in output mode (CHxMS=00 in TIMERx_CHCTL0 register). Only when counting down, CHxF bit can be set. 10: Center-aligned and counting up assert mode. The counter counts under center-aligned and channel is configured in output mode (CHxMS=00 in TIMERx_CHCTL0 register). Only when counting up, CHxF bit can be set. 11: Center-aligned and counting up/down assert mode. The counter counts under center-aligned and channel is configured in output mode (CHxMS=00 in TIMERx_CHCTL0 register). Both when counting up and counting down, CHxF bit

		can be set. After the counter is enabled, cannot be switched from 0x00 to non 0x00.
4	DIR	Direction 0: Count up 1: Count down If the timer work in center-aligned mode or encoder mode, this bit is read only.
3	SPM	Single pulse mode. 0: Single pulse mode disable. The counter continues after update event. 1: Single pulse mode enable. The counter counts until the next update event occurs.
2	UPS	Update source This bit is used to select the update event sources by software. 0: These events generate update interrupts or DMA requests: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: This event generates update interrupts or DMA requests: The counter generates an overflow or underflow event
1	UPDIS	Update disable. This bit is used to enable or disable the update event generation. 0: Update event enable. When an update event occurs, the corresponding shadow registers are loaded with their preloaded values. These events generate update event: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: Update event disable. **Note:** When this bit is set to 1, setting UPG bit or the restart mode does not generate an update event, but the counter and prescaler are initialized.
0	CEN	Counter enable 0: Counter disable 1: Counter enable The CEN bit must be set by software when timer works in external clock mode, pause mode or encoder mode.

Control register 1 (TIMERx_CTL1)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ISO3	ISO2N	ISO2	ISO1N	ISO1	ISO0N	ISO0	TI0S	MMC[2:0]			DMAS	CCUC	Reserved	CCSE
	rw	rw	rw	rw	rw	rw	rw	rw	rw			rw	rw		rw

Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14	ISO3	Idle state of channel 3 output Refer to ISO0 bit
13	ISO2N	Idle state of channel 2 complementary output Refer to ISO0N bit
12	ISO2	Idle state of channel 2 output Refer to ISO0 bit
11	ISO1N	Idle state of channel 1 complementary output Refer to ISO0N bit
10	ISO1	Idle state of channel 1 output Refer to ISO0 bit
9	ISO0N	Idle state of channel 0 complementary output 0: When POEN bit is reset, CH0_ON is set low. 1: When POEN bit is reset, CH0_ON is set high. This bit can be modified only when PROT[1:0] bits in TIMERx_CCHP register is 00.
8	ISO0	Idle state of channel 0 output 0: When POEN bit is reset, CH0_O is set low. 1: When POEN bit is reset, CH0_O is set high. The CH0_O output changes after a dead time if CH0_ON is implemented. This bit can be modified only when PROT[1:0] bits in TIMERx_CCHP register is 00.
7	TI0S	Channel 0 trigger input selection 0: The TIMERx_CH0 pin input is selected as channel 0 trigger input. 1: The result of combinational XOR of TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 pins is selected as channel 0 trigger input.
6:4	MMC[2:0]	Master mode control These bits control the selection of TRGO signal, which is sent by master timer to slave timer for synchronization function. 000: When a counter reset event occurs, a TRGO trigger signal is output. The counter resert source: Master timer generate a reset the UPG bit in the TIMERx_SWEVG register is set 001: Enable. When a conter start event occurs, a TRGO trigger signal is output. The counter start source : CEN control bit is set

		The trigger input in pause mode is high 010: When an update event occurs, a TRGO trigger signal is output. The update source depends on UPDIS bit and UPS bit. 011: When a capture or compare pulse event occurs in channel0, a TRGO trigger signal is output. 100: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O0CPRE. 101: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O1CPRE. 110: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O2CPRE. 111: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O3CPRE.
3	DMAS	DMA request source selection 0: When capture or compare event occurs, the DMA request of channel x is sent 1: When update event occurs, the DMA request of channel x is sent.
2	CCUC	Commutation control shadow register update control When the commutation control shadow registers (for CHxEN, CHxNEN and CHxCOMCTL bits) are enabled (CCSE=1), the update control of the shadow registers are shown as below: 0: The shadow registers update when CMTG bit is set. 1: The shadow registers update when CMTG bit is set or a rising edge of TRGI occurs. When a channel does not have a complementary output, this bit has no effect.
1	Reserved	Must be kept at reset value.
0	CCSE	Commutation control shadow enable 0: The shadow registers (for CHxEN, CHxNEN and CHxCOMCTL bits) are disabled. 1: The shadow registers (for CHxEN, CHxNEN and CHxCOMCTL bits) are enabled. After these bits have been written, they are updated when commutation event comes. When a channel does not have a complementary output, this bit has no effect.

Slave mode configuration register (TIMERx_SMCFG)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

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

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

ETP	SMC1	ETPSC[1:0]	ETFC[3:0]	MSM	TRGS[2:0]	Reserved	SMC[2:0]
rw	rw	rw	rw	rw	rw		rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15	ETP	External trigger polarity This bit specifies the polarity of ETI signal 0: ETI is active at rising edge or high level . 1: ETI is active at falling edge or low level .
14	SMC1	Part of SMC for enable External clock mode1. In external clock mode 1, the counter is clocked by any active edge on the ETIFP signal. 0: External clock mode 1 disabled 1: External clock mode 1 enabled. When the slave mode is configured as restart mode, pause mode or event mode, the timer can still work in the external clock 1 mode by setting this bit. But the TRGS bits must not be 3'b111 in this case. The clock source of the timer will be ETIFP if external clock mode 0 and external clock mode 1 are configured at the same time. **Note:** External clock mode 0 enable is in this register's SMC[2:0] bit-filed.
13:12	ETPSC[1:0]	The prescaler of external trigger The frequency of external trigger signal ETIFP must not be at higher than 1/4 of TIMER_CK frequency. When the external trigger signal is a fast clock, the prescaler can be enabled to reduce ETIFP frequency. 00: Prescaler disable. 01: The prescaler is 2. 10: The prescaler is 4. 11: The prescaler is 8.
11:8	ETFC[3:0]	External trigger filter control The external trigger can be filtered by digital filter and this bit-field configure the filtering capability. Basic principle of digital filter: continuously sample the external trigger signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit-field, it is considered to be an effective level. The filtering capability configuration is as follows:

EXTFC[3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{TIMER_CK}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS_CK}/2$

		4’b0101	8	
		4’b0110	6	$f_{DTS_CK}/4$
		4’b0111	8	
		4’b1000	6	$f_{DTS_CK}/8$
		4’b1001	8	
		4’b1010	5	$f_{DTS_CK}/16$
		4’b1011	6	
		4’b1100	8	
		4’b1101	5	$f_{DTS_CK}/32$
		4’b1110	6	
		4’b1111	8	

7 MSM

Master-slave mode

This bit can be used to synchronize selected timers to begin counting at the same time. The TRGI is used as the start event, and through TRGO, timers are connected together.

0: Master-slave mode disable

1: Master-slave mode enable

6:4 TRGS[2:0]

Trigger selection

This bit-field specifies which signal is selected as the trigger input, which is used to synchronize the counter.

000: ITI0

001: ITI1

010: ITI2

011: ITI3

100: CI0F_ED

101: CI0FE0

110: CI1FE1

111: ETIFP

These bits must not be changed when slave mode is enabled.

3 Reserved

Must be kept at reset value.

2:0 SMC[2:0]

Slave mode control

000: Disable mode. The slave mode is disabled; The prescaler is clocked directly by the internal clock (TIMER_CK) when CEN bit is set high.

001: Quadrature decoder mode 0.The counter counts on CI0FE0 edge, while the direction depends on CI1FE1 level.

010: Quadrature decoder mode 1.The counter counts on CI1FE1 edge, while the direction depends on CI0FE0 level.

011: Quadrature decoder mode 2.The counter counts on both CI0FE0 and CI1FE1 edge, while the direction depends on each other.

100: Restart mode. The counter is reinitialized and an update event is generated on the rising edge of the selected trigger input.

101: Pause mode. The trigger input enables the counter clock when it is high and

disables the counter clock when it is low.

110: Event mode. A rising edge of the trigger input enables the counter.

111: External clock mode 0. The counter counts on the rising edges of the selected trigger.

DMA and interrupt enable register (TIMERx_DMAINTEN)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TRGDEN	CMTDEN	CH3DEN	CH2DEN	CH1DEN	CH0DEN	UPDEN	BRKIE	TRGIE	CMTIE	CH3IE	CH2IE	CH1IE	CH0IE	UPIE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14	TRGDEN	Trigger DMA request enable 0: Disabled 1: Enabled
13	CMTDEN	Commutation DMA request enable 0: Disabled 1: Enabled
12	CH3DEN	Channel 3 capture/compare DMA request enable 0: Disabled 1: Enabled
11	CH2DEN	Channel 2 capture/compare DMA request enable 0: Disabled 1: Enabled
10	CH1DEN	Channel 1 capture/compare DMA request enable 0: Disabled 1: Enabled
9	CH0DEN	Channel 0 capture/compare DMA request enable 0: Disabled 1: Enabled
8	UPDEN	Update DMA request enable 0: Disabled

		1: Enabled
7	BRKIE	Break interrupt enable 0: Disabled 1: Enabled
6	TRGIE	Trigger interrupt enable 0: Disabled 1: Enabled
5	CMTIE	Commutation interrupt enable 0: Disabled 1: Enabled
4	CH3IE	Channel 3 capture/compare interrupt enable 0: Disabled 1: Enabled
3	CH2IE	Channel 2 capture/compare interrupt enable 0: Disabled 1: Enabled
2	CH1IE	Channel 1 capture/compare interrupt enable 0: Disabled 1: Enabled
1	CH0IE	Channel 0 capture/compare interrupt enable 0: Disabled 1: Enabled
0	UPIE	Update interrupt enable 0: Disabled 1: Enabled

Interrupt flag register (TIMERx_INTF)

Address offset: 0x10
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CH3OF	CH2OF	CH1OF	CH0OF	Reserved	BRKIF	TRGIF	CMTIF	CH3IF	CH2IF	CH1IF	CH0IF	UPIF
			rc_w0	rc_w0	rc_w0	rc_w0	.	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bits	Fields	Descriptions

31:13	Reserved	Must be kept at reset value.
12	CH3OF	Channel 3 over capture flag Refer to CH0OF description
11	CH2OF	Channel 2 over capture flag Refer to CH0OF description
10	CH1OF	Channel 1 over capture flag Refer to CH0OF description
9	CH0OF	Channel 0 over capture flag When channel 0 is configured in input mode, this flag is set by hardware when a capture event occurs while CH0IF flag has already been set. This flag is cleared by software. 0: No over capture interrupt occurred 1: Over capture interrupt occurred
8	Reserved	Must be kept at reset value.
7	BRKIF	Break interrupt flag When the break input is inactive, the bit is set by hardware. When the break input is inactive, the bit can be cleared by software. 0: No active level break has been detected. 1: An active level has been detected.
6	TRGIF	Trigger interrupt flag This flag is set on trigger event and cleared by software. When in pause mode, both edges on trigger input generates a trigger event, otherwise, only an active edge on trigger input can generates a trigger event. 0: No trigger event occurred 1: Trigger interrupt occurred
5	CMTIF	Channel commutation interrupt flag This flag is set by hardware when the commutation event of channel occurs, and cleared by software. 0: No channel commutation interrupt occurred 1: Channel commutation interrupt occurred
4	CH3IF	Channel 3 capture/compare interrupt flag Refer to CH0IF description
3	CH2IF	Channel 2 capture/compare interrupt flag Refer to CH0IF description
2	CH1IF	Channel 1 capture/compare interrupt flag Refer to CH0IF description
1	CH0IF	Channel 0 capture/compare interrupt flag This flag is set by hardware and cleared by software.

		If channel 0 is in input mode, this flag is set when a capture event occurs. If channel 0 is in output mode, this flag is set when a compare event occurs. If channel 0 is set to input mode, this bit will be reset by reading TIMERx_CH0CV. 0: No channel 0 interrupt occurred 1: Channel 0 interrupt occurred
0	UPIF	Update interrupt flag This bit is set by hardware when an update event occurs and cleared by software. 0: No update interrupt occurred 1: Update interrupt occurred

Software event generation register (TIMERx_SWEVG)

Address offset: 0x14
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BRKG	TRGG	CMTG	CH3G	CH2G	CH1G	CH0G	UPG
								w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	BRKG	Break event generation This bit is set by software to generate an event and cleared by hardware automatically. When this bit is set, the POEN bit will be cleared and BRKIF flag will be set, related interrupt or DMA transfer can occur if enabled. 0: No generate a break event 1: Generate a break event
6	TRGG	Trigger event generation This bit is set by software and cleared by hardware automatically. When this bit is set, the TRGIF flag in TIMERx_INTF register will be set, related interrupt or DMA transfer can occur if enabled. 0: No generate a trigger event 1: Generate a trigger event
5	CMTG	Channel commutation event generation This bit is set by software and cleared by hardware automatically. When this bit is set, channel's capture/compare control registers (CHxEN, CHxNEN and CHxCOMCTL bits) are updated based on the value of CCSE (in the TIMERx_CTL1).

		0: No affect 1: Generate channel commutation update event
4	CH3G	Channel 3 capture or compare event generation Refer to CH0G description
3	CH2G	Channel 2 capture or compare event generation Refer to CH0G description
2	CH1G	Channel 1 capture or compare event generation Refer to CH0G description
1	CH0G	Channel 0 capture or compare event generation This bit is set by software to generate a capture or compare event in channel 0, it is automatically cleared by hardware. When this bit is set, the CH0IF flag will be set, and the corresponding interrupt or DMA request will be sent if enabled. In addition, if channel 0 is configured in input mode, the current value of the counter is captured to TIMERx_CH0CV register, and the CH0OF flag is set if the CH0IF flag has been set. 0: No generate a channel 0 capture or compare event 1: Generate a channel 0 capture or compare event
0	UPG	Update event generation This bit can be set by software, and automatically cleared by hardware. When this bit is set, the counter is cleared if the center-aligned or up counting mode is selected, while in down counting mode it takes the auto-reload value. The prescaler counter is cleared at the same time. 0: No generate an update event 1: Generate an update event

Channel control register 0 (TIMERx_CHCTL0)

Address offset: 0x18
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1COMCEN	CH1COMCTL[2:0]			CH1COMSEN	CH1COMFEN	CH1MS[1:0]		CH0COMCEN	CH0COMCTL[2:0]			CH0COMSEN	CH0COMFEN	CH0MS[1:0]	
CH1CAPFLT[3:0]				CH1CAPPSC[1:0]				CH0CAPFLT[3:0]				CH0CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions

31:16	Reserved	Must be kept at reset value.
15	CH1COMCEN	Channel 1 output compare clear enable Refer to CH0COMCEN description
14:12	CH1COMCTL[2:0]	Channel 1 compare output control Refer to CH0COMCTL description
11	CH1COMSEN	Channel 1 output compare shadow enable Refer to CH0COMSEN description
10	CH1COMFEN	Channel 1 output compare fast enable Refer to CH0COMFEN description
9:8	CH1MS[1:0]	Channel 1 mode selection This bit-field specifies the direction of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH1EN bit in TIMERx_CHCTL2 register is reset). 00: Channel 1 is programmed as output mode 01: Channel 1 is programmed as input mode, IS1 is connected to CI1FE1 10: Channel 1 is programmed as input mode, IS1 is connected to CI0FE1 11: Channel 1 is programmed as input mode, IS1 is connected to ITS. **Note:** When CH1MS[1:0]=11, it is necessary to select an internal trigger input through TRGS bits in TIMERx_SMCFG register.
7	CH0COMCEN	Channel 0 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the O0CPRE signal will be cleared. 0: Channel 0 output compare clear disable 1: Channel 0 output compare clear enable
6:4	CH0COMCTL[2:0]	Channel 0 compare output control This bit-field specifies the compare output mode of the the output prepare signal O0CPRE. In addition, the high level of O0CPRE is the active level, and CH0_O and CH0_ON channels polarity depends on CH0P and CH0NP bits. 000: Timing mode. The O0CPRE signal keeps stable, independent of the comparison between the register TIMERx_CH0CV and the counter TIMERx_CNT. 001: Set the channel output. O0CPRE signal is forced high when the counter is equals to the output compare register TIMERx_CH0CV. 010: Clear the channel output. O0CPRE signal is forced low when the counter is equals to the output compare register TIMERx_CH0CV. 011: Toggle on match. O0CPRE toggles when the counter is equals to the output compare register TIMERx_CH0CV. 100: Force low. O0CPRE is forced to low level. 101: Force high. O0CPRE is forced to high level. 110: PWM mode0. When counting up, O0CPRE is high when the counter is smaller than TIMERx_CH0CV, and low otherwise. When counting down, O0CPRE is low when the counter is larger than TIMERx_CH0CV, and high otherwise.

		111: PWM mode1. When counting up, O0CPRE is low when the counter is smaller than TIMERx_CH0CV, and high otherwise. When counting down, O0CPRE is high when the counter is larger than TIMERx_CH0CV, and low otherwise. If configured in PWM mode, the O0CPRE level changes only when the output compare mode is adjusted from “Timing” mode to “PWM” mode or the comparison result changes. This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH0MS bit-filed is 00(COMPARE MODE).
3	CH0COMSEN	Channel 0 compare output shadow enable When this bit is set, the shadow register of TIMERx_CH0CV register, which updates at each update event, will be enabled. 0: Channel 0 output compare shadow disable 1: Channel 0 output compare shadow enable The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1) This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH0MS bit-filed is 00.
2	CH0COMFEN	Channel 0 output compare fast enable When this bit is set, the effect of an event on the trigger in input on the capture/compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH0_O is set to the compare level independently from the result of the comparison. 0: Channel 0 output quickly compare disable. 1: Channel 0 output quickly compare enable.
1:0	CH0MS[1:0]	Channel 0 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH0EN bit in TIMERx_CHCTL2 register is reset).). 00: Channel 0 is programmed as output mode 01: Channel 0 is programmed as input mode, IS0 is connected to CI0FE0 10: Channel 0 is programmed as input mode, IS0 is connected to CI1FE0 11: Channel 0 is programmed as input mode, IS0 is connected to ITS **Note:** When CH0MS[1:0]=11, it is necessary to select an internal trigger input through TRGS bits in TIMERx_SMCFG register.

Input capture mode:

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	CH1CAPFLT[3:0]	Channel 1 input capture filter control Refer to CH0CAPFLT description

11:10 CH1CAPPSC[1:0] Channel 1 input capture prescaler
Refer to CH0CAPPSC description

9:8 CH1MS[1:0] Channel 1 mode selection
Same as output compare mode

7:4 CH0CAPFLT[3:0] Channel 0 input capture filter control
The CI0 input signal can be filtered by digital filter and this bit-field configure the filtering capability.
Basic principle of digital filter: continuously sample the CI0 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level.
The filtering capability configuration is as follows:

CH0CAPFLT [3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{CK_TIMER}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	
4'b0110	6	$f_{DTS}/4$
4'b0111	8	
4'b1000	6	$f_{DTS}/8$
4'b1001	8	
4'b1010	5	$f_{DTS}/16$
4'b1011	6	
4'b1100	8	
4'b1101	5	$f_{DTS}/32$
4'b1110	6	
4'b1111	8	

3:2 CH0CAPPSC[1:0] Channel 0 input capture prescaler
This bit-field specifies the factor of the prescaler on channel 0 input. The prescaler is reset when CH0EN bit in TIMERx_CHCTL2 register is clear.
00: Prescaler disable, input capture occurs on every channel input edge
01: The input capture occurs on every 2 channel input edges
10: The input capture occurs on every 4 channel input edges
11: The input capture occurs on every 8 channel input edges

1:0 CH0MS[1:0] Channel 0 mode selection
Same as output compare mode

Channel control register 1 (TIMERx_CHCTL1)

Address offset: 0x1C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3COMCEN	CH3COMCTL[2:0]			CH3COMSEN	CH3COMFEN	CH3MS[1:0]		CH2COMCEN	CH2COMCTL[2:0]			CH2COMSEN	CH2COMFEN	CH2MS[1:0]	
CH3CAPFLT[3:0]				CH3CAPPSC[1:0]				CH2CAPFLT[3:0]				CH2CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	CH3COMCEN	Channel 3 output compare clear enable Refer to CH0COMCEN description
14:12	CH3COMCTL[2:0]	Channel 3 compare output control Refer to CH0COMCTL description
11	CH3COMSEN	Channel 3 output compare shadow enable Refer to CH0COMSEN description
10	CH3COMFEN	Channel 3 output compare fast enable Refer to CH0COMFEN description
9:8	CH3MS[1:0]	Channel 3 mode selection This bit-field specifies the direction of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH3EN bit in TIMERx_CHCTL2 register is reset). 00: Channel 3 is programmed as output mode 01: Channel 3 is programmed as input mode, IS3 is connected to CI3FE3 10: Channel 3 is programmed as input mode, IS3 is connected to CI2FE3 11: Channel 3 is programmed as input mode, IS3 is connected to ITS. **Note:** When CH3MS[1:0]=11, it is necessary to ensure that an internal trigger input is selected through TRGS bits in TIMERx_SMCFG register.
7	CH2COMCEN	Channel 2 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the O2CPRE signal will be cleared. 0: Channel 2 output compare clear disable 1: Channel 2 output compare clear enable
6:4	CH2COMCTL[2:0]	Channel 2 compare output control This bit-field specifies the compare output mode of the the output prepare signal O0CPRE. In addition, the high level of O0CPRE is the active level, and CH0_O and CH0_ON channels polarity depends on CH0P and CH0NP bits.

		000: Timing mode. The O2CPRE signal keeps stable, independent of the comparison between the output compare register TIMERx_CH2CV and the counter TIMERx_CNT. 001: Set the channel output. O2CPRE signal is forced high when the counter is equals to the output compare register TIMERx_CH2CV. 010: Clear the channel output. O2CPRE signal is forced low when the counter is equals to the output compare register TIMERx_CH2CV. 011: Toggle on match. O2CPRE toggles when the counter is equals to the output compare register TIMERx_CH2CV. 100: Force low. O2CPRE is forced to low level. 101: Force high. O2CPRE is forced to high level. 110: PWM mode 0. When counting up, O2CPRE is high when the counter is smaller than TIMERx_CH2CV, and low otherwise. When counting down, O2CPRE is low when the counter is larger than TIMERx_CH2CV, and high otherwise. 111: PWM mode 1. When counting up, O2CPRE is low when the counter is smaller than TIMERx_CH2CV, and high otherwise. When counting down, O2CPRE is high when the counter is larger than TIMERx_CH2CV, and low otherwise. If configured in PWM mode, the O2CPRE level changes only when the output compare mode is adjusted from “Timing” mode to “PWM” mode or the comparison result changes. This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH2MS bit-filed is 00(COMPARE MODE).
3	CH2COMSEN	Channel 2 compare output shadow enable When this bit is set, the shadow register of TIMERx_CH2CV register, which updates at each update event will be enabled. 0: Channel 2 output compare shadow disable 1: Channel 2 output compare shadow enable The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1) This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH0MS bit-filed is 00.
2	CH2COMFEN	Channel 2 output compare fast enable When this bit is set, the effect of an event on the trigger in input on the capture/compare output will be accelerated if the channel is configured in PWM1 or PWM2 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH2_O is set to the compare level independently from the result of the comparison. 0: Channel 2 output quickly compare disable. 1: Channel 2 output quickly compare enable.
1:0	CH2MS[1:0]	Channel 2 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH2EN bit in TIMERx_CHCTL2 register is reset).).

00: Channel 2 is programmed as output mode
01: Channel 2 is programmed as input mode, IS2 is connected to CI2FE2
10: Channel 2 is programmed as input mode, IS2 is connected to CI3FE2
11: Channel 2 is programmed as input mode, IS2 is connected to ITS.
Note: When CH2MS[1:0]=11, it is necessary to ensure that an internal trigger input is selected through TRGS bits in TIMERx_SMCFG register.

Input capture mode:

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	CH3CAPFLT[3:0]	Channel 3 input capture filter control Refer to CH0CAPFLT description
11:10	CH3CAPPSC[1:0]	Channel 3 input capture prescaler Refer to CH0CAPPSC description
9:8	CH3MS[1:0]	Channel 3 mode selection Same as output compare mode
7:4	CH2CAPFLT[3:0]	Channel 2 input capture filter control The CI2 input signal can be filtered by digital filter and this bit-field configure the filtering capability. Basic principle of digital filter: continuously sample the CI2 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level. The filtering capability configuration is as follows:

CH2CAPFLT [3:0]	Times	f_{SAMP}
4’b0000	Filter disabled.	
4’b0001	2	f_{CK_TIMER}
4’b0010	4	
4’b0011	8	
4’b0100	6	$f_{DTS}/2$
4’b0101	8	
4’b0110	6	$f_{DTS}/4$
4’b0111	8	
4’b1000	6	$f_{DTS}/8$
4’b1001	8	
4’b1010	5	$f_{DTS}/16$
4’b1011	6	
4’b1100	8	
4’b1101	5	$f_{DTS}/32$
4’b1110	6	
4’b1111	8	

3:2	CH2CAPPSC[1:0]	Channel 2 input capture prescaler

		This bit-field specifies the factor of the prescaler on channel 2 input. The prescaler is reset when CH2EN bit in TIMERx_CHCTL2 register is clear. 00: Prescaler disable, input capture occurs on every channel input edge 01: The input capture occurs on every 2 channel input edges 10: The input capture occurs on every 4 channel input edges 11: The input capture occurs on every 8 channel input edges
1:0	CH2MS[1:0]	Channel 2 mode selection Same as output compare mode

Channel control register 2 (TIMERx_CHCTL2)

Address offset: 0x20
Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		CH3P	CH3EN	CH2NP	CH2NEN	CH2P	CH2EN	CH1NP	CH1NEN	CH1P	CH1EN	CH0NP	CH0NEN	CH0P	CH0EN
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13	CH3P	Channel 3 capture/compare function polarity Refer to CH0P description
12	CH3EN	Channel 3 capture/compare function enable Refer to CH0EN description
11	CH2NP	Channel 2 complementary output polarity Refer to CH0NP description
10	CH2NEN	Channel 2 complementary output enable Refer to CH0NEN description
9	CH2P	Channel 2 capture/compare function polarity Refer to CH0P description
8	CH2EN	Channel 2 capture/compare function enable Refer to CH0EN description
7	CH1NP	Channel 1 complementary output polarity Refer to CH0NP description
6	CH1NEN	Channel 1 complementary output enable

		Refer to CH0NEN description
5	CH1P	Channel 1 capture/compare function polarity Refer to CH0P description
4	CH1EN	Channel 1 capture/compare function enable Refer to CH0EN description
3	CH0NP	Channel 0 complementary output polarity When channel 0 is configured in output mode, this bit specifies the complementary output signal polarity. 0: Channel 0 complementary output high level is active level 1: Channel 0 complementary output low level is active level When channel 0 is configured in input mode, together with CH0P, this bit is used to define the polarity of CI0. This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 or 10.
2	CH0NEN	Channel 0 complementary output enable When channel 0 is configured in output mode, setting this bit enables the complementary output in channel 0. 0: Channel 0 complementary output disabled 1: Channel 0 complementary output enabled
1	CH0P	Channel 0 capture/compare function polarity When channel 0 is configured in output mode, this bit specifies the output signal polarity. 0: Channel 0 high level is active level 1: Channel 0 low level is active level When channel 0 is configured in input mode, this bit specifies the CI0 signal polarity. [CH0P=0]: The rising edge of CIxFE0 is the active signal for capture or trigger operation in slave mode. And CIxFE0 will not be inverted. [CH0P=1]: The falling edge of CIxFE0 is the active signal for capture or trigger operation in slave mode. And CIxFE0 will be inverted. This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 or 10.
0	CH0EN	Channel 0 capture/compare function enable When channel 0 is configured in output mode, setting this bit enables CH0_O signal in active state. When channel 0 is configured in input mode, setting this bit enables the capture event in channel 0. 0: Channel 0 disabled 1: Channel 0 enabled

Counter register (TIMERx_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CNT[15:0]	This bit-field indicates the current counter value. Writing to this bit-field can change the value of the counter.

Prescaler register (TIMERx_PSC)

Address offset: 0x28
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

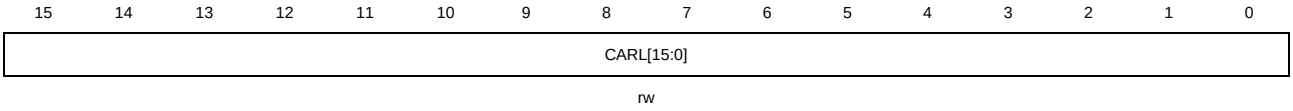
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC[15:0]	Prescaler value of the counter clock The TIMER_CK clock is divided by (PSC+1) to generate the counter clock. The value of this bit-filed will be loaded to the corresponding shadow register at every update event.

Counter auto reload register (TIMERx_CAR)

Address offset: 0x2C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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



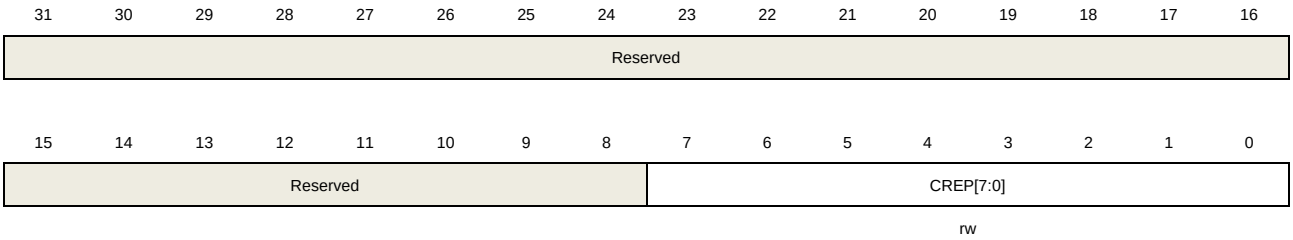
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CARL[15:0]	Counter auto reload value This bit-field specifies the auto reload value of the counter.

Counter repetition register (TIMERx_CREP)

Address offset: 0x30

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



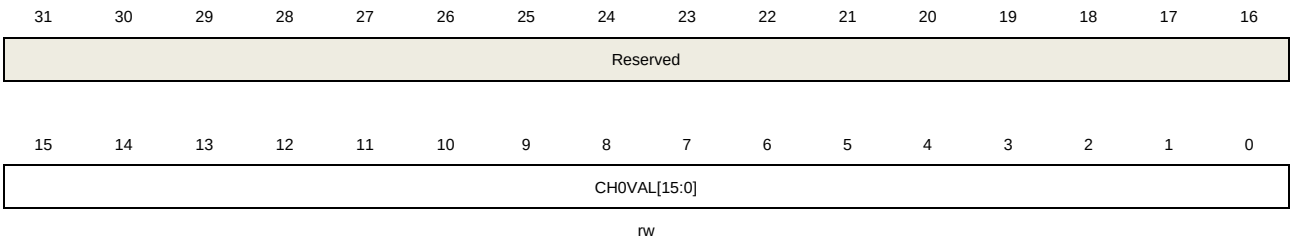
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	CREP[7:0]	Counter repetition value This bit-field specifies the update event generation rate. Each time the repetition counter counts down to zero, an update event will be generated. The update rate of the shadow registers is also affected by this bit-field when these shadow registers are enabled.

Channel 0 capture/compare value register (TIMERx_CH0CV)

Address offset: 0x34

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH0VAL[15:0]	Capture/compare value of channel 0 When channel 0 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 0 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Channel 1 capture/compare value register (TIMERx_CH1CV)

Address offset: 0x38
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH1VAL[15:0]	Capture/compare value of channel 1 When channel 1 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 1 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Channel 2 capture/compare value register (TIMERx_CH2CV)

Address offset: 0x3C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2VAL[15:0]															

rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH2VAL[15:0]	Capture/compare value of channel 2 When channel 2 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 2 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Channel 3 capture/compare value register (TIMERx_CH3CV)

Address offset: 0x40

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3VAL[15:0]															

rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH3VAL[15:0]	Capture/compare value of channel 3 When channel3 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 3 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Complementary channel protection register (TIMERx_CCHP)

Address offset: 0x44

Reset value: 0x0000

This register can be accessed by half-word (16-bit) or word (32-bit)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POEN	OAEN	BRKP	BRKEN	ROS	IOS	PROT[1:0]		DTCFG[7:0]							
rw	rw	rw	rw	rw	rw	rw		rw							

Bits	Fields	Descriptions
15	POEN	Primary output enable The bit can be set to 1 by: - Write 1 to this bit - If OAEN is set to 1, this bit is set to 1 at the next update event. The bit can be cleared to 0 by: - Write 0 to this bit - Valid fault input (asynchronous). When one of channels is configured in output mode, setting this bit enables the channel outputs (CHx_O and CHx_ON) if the corresponding enable bits (CHxEN, CHxNEN in TIMERx_CHCTL2 register) have been set. 0: Disable channel outputs (CHxO or CHxON). 1: Enabled channel outputs (CHxO or CHxON). **Note:** This bit is only valid when CHxMS=2'b00.
14	OAEN	Output automatic enable 0: The POEN bit can only be set by software. 1: POEN can be set at the next update event, if the break input is not active. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.
13	BRKP	Break polarity This bit specifies the polarity of the BRKIN input signal. 0: BRKIN input active low 1; BRKIN input active high
12	BRKEN	Break enable This bit can be set to enable the BRKIN and CCS clock failure event inputs. 0: Break inputs disabled 1; Break inputs enabled This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.
11	ROS	Run mode "off-state" enable When POEN bit is set (Run mode), this bit can be set to enable the "off-state" for the channels which has been configured in output mode. Please refer to ***Table 16-2. Complementary outputs controlled by parameters***. 0: "off-state" disabled. If the CHxEN or CHxNEN bit is reset, the corresponding channel is output disabled. 1: "off-state" enabled. If the CHxEN or CHxNEN bit is reset, the corresponding channel is "off-state". This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.
10	IOS	Idle mode "off-state" enable When POEN bit is reset (Idle mode), this bit can be set to enable the "off-state" for the channels which has been configured in output mode. Please refer to ***Table 16-2.***

Complementary outputs controlled by parameters.

0: “off-state” disabled. If the CHxEN/CHxNEN bits are both reset, the channels are output disabled.

1: “off-state” enabled. No matter the CHxEN/CHxNEN bits, the channels are “off-state”.

This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.

9:8 PROT[1:0] Complementary register protect control

This bit-filed specifies the write protection property of registers.

00: protect disable. No write protection.

01: PROT mode 0.The ISOx/ISOxN bits in TIMERx_CTL1 register and the BRKEN/BRKP/OAEN/DTCFG bits in TIMERx_CCHP register are writing protected.

10: PROT mode 1. In addition of the registers in PROT mode 0, the CHxP/CHxNP bits in TIMERx_CHCTL2 register (if related channel is configured in output mode) and the ROS/IOS bits in TIMERx_CCHP register are writing protected.

11: PROT mode 2. In addition of the registers in PROT mode 1, the CHxCOMCTL/CHxCOMSEN bits in TIMERx_CHCTL0/1 registers (if the related channel is configured in output) are writing protected.

This bit-field can be written only once after the reset. Once the TIMERx_CCHP register has been written, this bit-field will be writing protected.

7:0 DTCFG[7:0] Dead time configure

The relationship between DTVAL value and the duration of dead-time is as follow:

DTCFG[7:5]	The duration of dead-time
3’b0xx	DTCFG[7:0] * t_{DTS_CK}
3’b10x	(64+ DTCFG[5:0]) * t_{DTS_CK} *2
3’b110	(32+ DTCFG[4:0]) * t_{DTS_CK} *8
3’b111	(32+ DTCFG[4:0]) * t_{DTS_CK} *16

Note:

1. t_{DTS_CK} is the period of DTS_CK which is configured by CKDIV[1:0] in TIMERx_CTL0.
2. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.

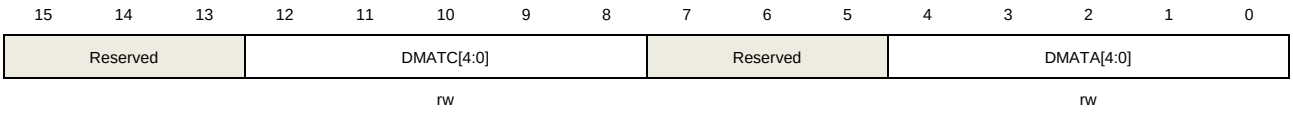
DMA configuration register (TIMERx_DMACFG)

Address offset: 0x48

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

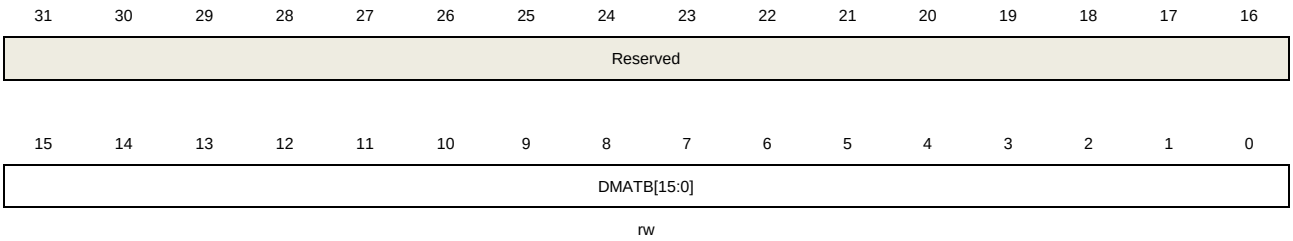


Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:8	DMATC[4:0]	DMA transfer count This filed defines the number(n) of the register that DMA will access(R/W), n = (DMATC [4:0] +1). DMATC [4:0] is from 5'b0_0000 to 5'b1_0001.
7:5	Reserved	Must be kept at reset value.
4:0	DMATA[4:0]	DMA transfer access start address This field defines the start address of accessing the TIMERx_DMATB register by DMA. When the first access to the TIMERx_DMATB register is done, this bit-field specifies the address just accessed. And then the address of the second access to the TIMERx_DMATB register will be (start address + 0x4).

DMA transfer buffer register (TIMERx_DMATB)

Address offset: 0x4C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

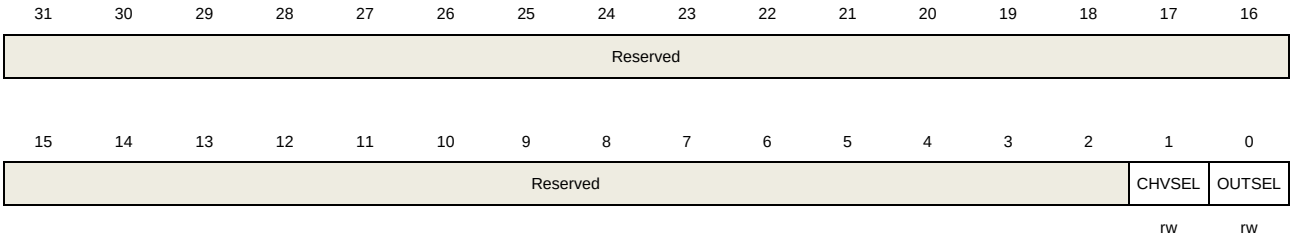


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	DMATB[15:0]	DMA transfer buffer When a read or write operation is assigned to this register, the register located at the address ranges from (start address) to (start address + transfer count * 4) will be accessed. The transfer count is calculated by hardware, and ranges from 0 to DMATC.

Configuration register (TIMERx_CFG)

Address offset: 0xFC
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	CHVSEL	Write CHxVAL register selection bit This bit-field is set and reset by software. 1: If the value to be written to the CHxVAL register is the same as the value of CHxVAL register, the write access is ignored. 0: No effect.
0	OUTSEL	The output value selection bit This bit-field is set and reset by software. 1: If POEN bit and IOS bit are 0, the output is disabled. 0: No effect.

16.2. General level0 timer (TIMERx, x=1, 2, 3, 4)

16.2.1. Overview

The general level0 timer module (TIMER1, 2, 3, 4) is a four-channel timer that supports input capture and output compare. They can generate PWM signals to control motor or be used for power management applications. The general level0 timer has a 16-bit counter that can be used as an unsigned counter.

In addition, the general level0 timers can be programmed and be used for counting, their external events can be used to drive other timers.

Timers are completely independent with each other, but they may be synchronized to provide a larger timer with their counter value increasing in unison.

16.2.2. Characteristics

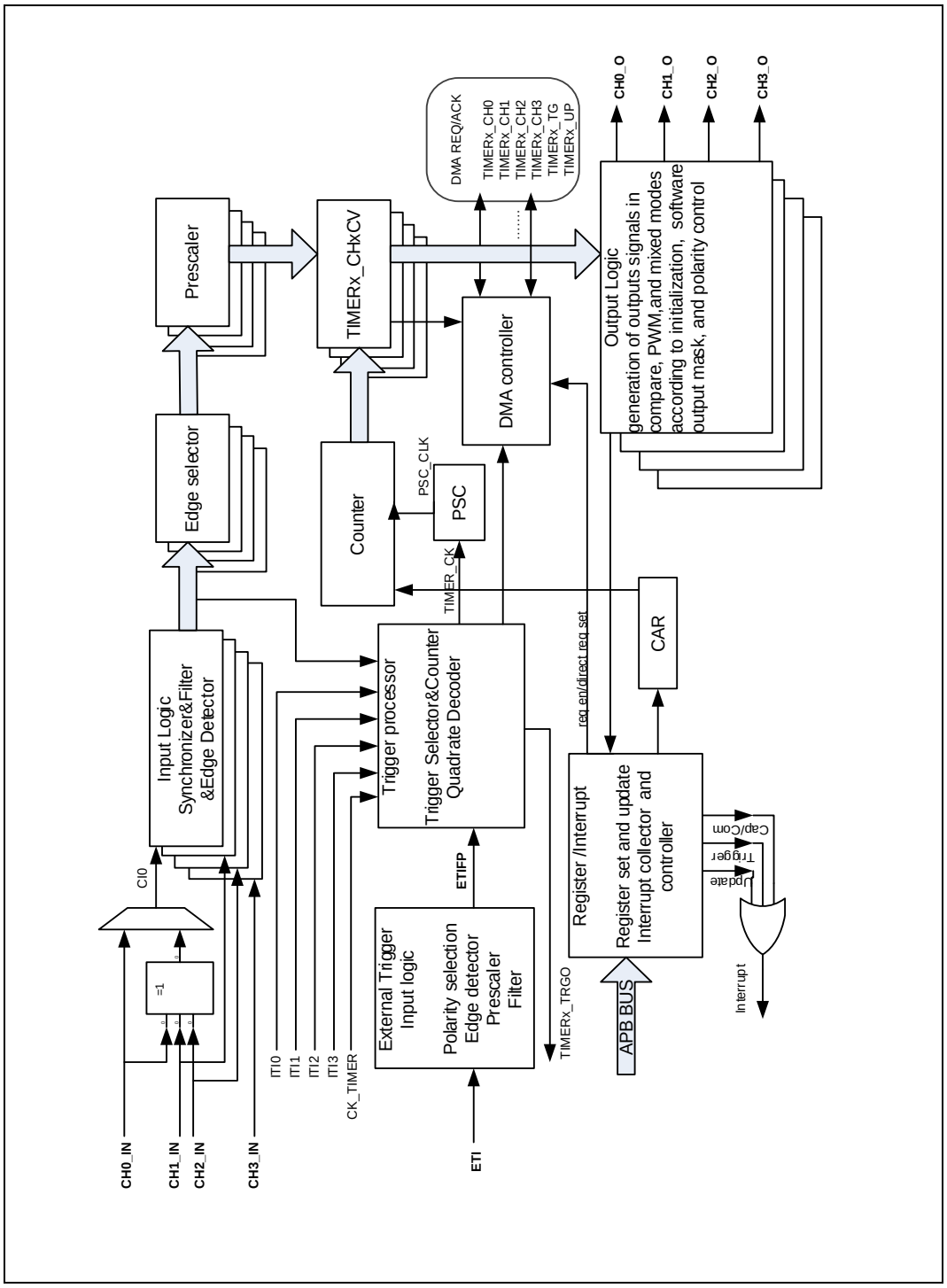
- Total channel num: 4.
- Counter width: 16 bits.
- Clock source of timer is selectable: internal clock, internal trigger, external input, external trigger.
- Multiple counter modes: up counting, down counting and center-aligned counting.

- Quadrature decoder: used for motion tracking and determination of both rotation direction and position.
- Hall sensor function: used for 3-phase motor control.
- Programmable prescaler: 16 bits. The factor can be changed ongoing.
- Each channel is user-configurable: input capture mode, output compare mode, programmable PWM mode and single pulse mode.
- Auto reload function.
- Interrupt output or DMA request: update event, trigger event and compare/capture event.
- Daisy chaining of timer module allows a single timer to start multiple timers.
- Timer synchronization allows the selected timers to start counting on the same clock cycle.
- Timer master-slave management.

16.2.3. Block diagram

Figure 16-29. General Level 0 timer block diagram provides details on the internal configuration of the general level0 timer.

Figure 16-29. General Level 0 timer block diagram



16.2.4. Function overview

Clock source configuration

The general level0 TIMER has the capability of being clocked by either the CK_TIMER or an alternate clock source controlled by SMC (TIMERx_SMCFG bit [2:0]).

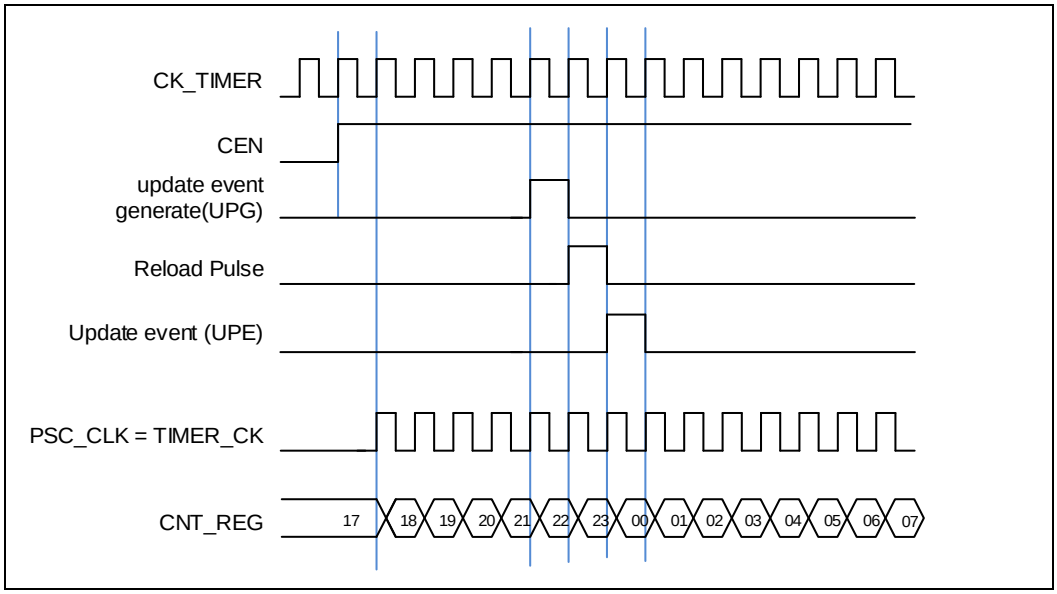
- SMC [2:0] == 3'b000. Internal timer clock CK_TIMER which is from module RCU.

The default internal clock source is the CK_TIMER used to drive the counter prescaler when the SMC [2:0] == 3'b000. When the CEN is set, the CK_TIMER will be divided by PSC value to generate PSC_CLK.

In this mode, the TIMER_CK, driven counter's prescaler to count, is equal to CK_TIMER which is from RCU.

If the SMC [2:0] in the TIMERx_SMCFG register are setting to an available value including 0x1, 0x2, 0x3 and 0x7, the prescaler is clocked by other clock sources selected by the TRGS [2:0] in the TIMERx_SMCFG register and described as follows. When the SMC [2:0] bits are set to 0x4, 0x5 or 0x6, the internal clock CK_TIMER is the counter prescaler driving clock source.

Figure 16-30. Timing chart of internal clock divided by 1



- SMC [2:0] == 3'b111(external clock mode 0). External input pin source

The TIMER_CK, which drives counter's prescaler to count, can be triggered by the event of rising or falling edge on the external pin TIMERx_CI0/TIMERx_CI1. This mode can be selected by setting SMC [2:0] to 0x7 and the TRGS [2:0] to 0x4, 0x5 or 0x6.

And, the counter prescaler can also be driven by rising edge on the internal trigger input pin ITI0/1/2/3. This mode can be selected by setting SMC [2:0] to 0x7 and the TRGS [2:0] to 0x0, 0x1, 0x2 or 0x3.

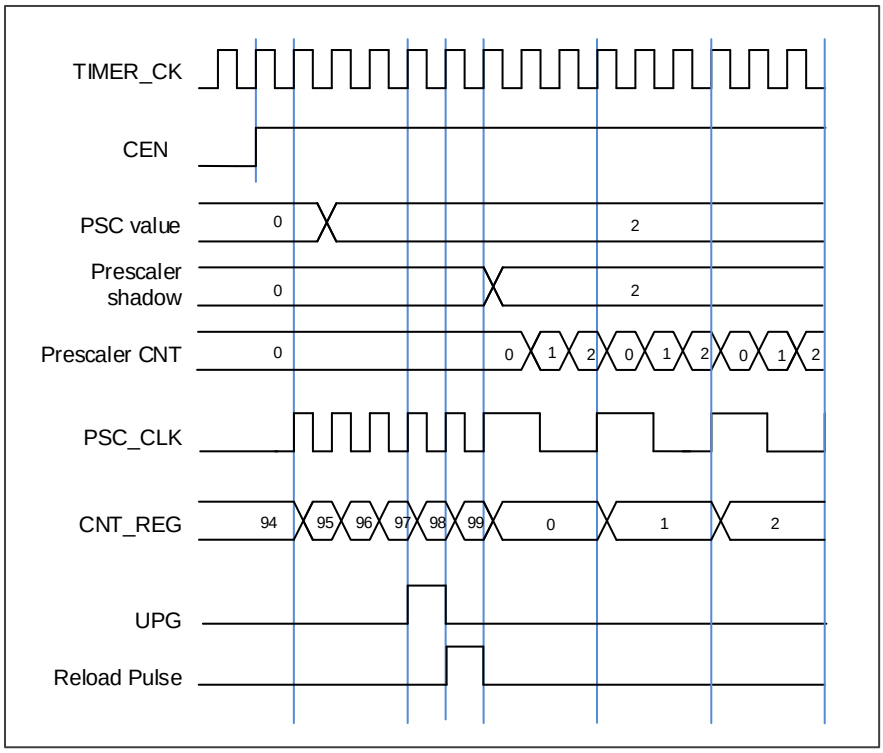
- SMC1== 1'b1(external clock mode 1). External input pin source (ETI)

The TIMER_CK, which drives counter's prescaler to count, can be triggered by the event of rising or falling edge on the external pin ETI. This mode can be selected by setting the SMC1 bit in the TIMERx_SMCFG register to 1. The other way to select the ETI signal as the clock source is to set the SMC [2:0] to 0x7 and the TRGS [2:0] to 0x7 respectively. Note that the ETI signal is derived from the ETI pin sampled by a digital filter. When the ETI signal is selected as clock source, the trigger controller including the edge detection circuitry will generate a clock pulse on each ETI signal rising edge to clock the counter prescaler.

Clock prescaler

The counter clock (PSC_CK) is obtained by the TIMER_CK through the prescaler, and the prescale factor can be configured from 1 to 65536 through the prescaler register (TIMERx_PSC). The new written prescaler value will not take effect until the next update event.

Figure 16-31. Timing chart of PSC value change from 0 to 2



Counter up counting

In this mode, the counter counts up continuously from 0 to the counter-reload value, which is defined in the TIMERx_CAR register, in a count-up direction. Once the counter reaches the counter reload value, the counter will start counting up from 0 again. The update event is generated at each counter overflow. The counting direction bit DIR in the TIMERx_CTL1 register should be set to 0 for the up counting mode.

When the update event is set by the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to 0 and generates an update event.

If the UPDIS bit in TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (counter autoreload register, prescaler register) are updated.

Figure 16-32. Timing chart of up counting mode, PSC=0/2 show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 16-32. Timing chart of up counting mode, PSC=0/2

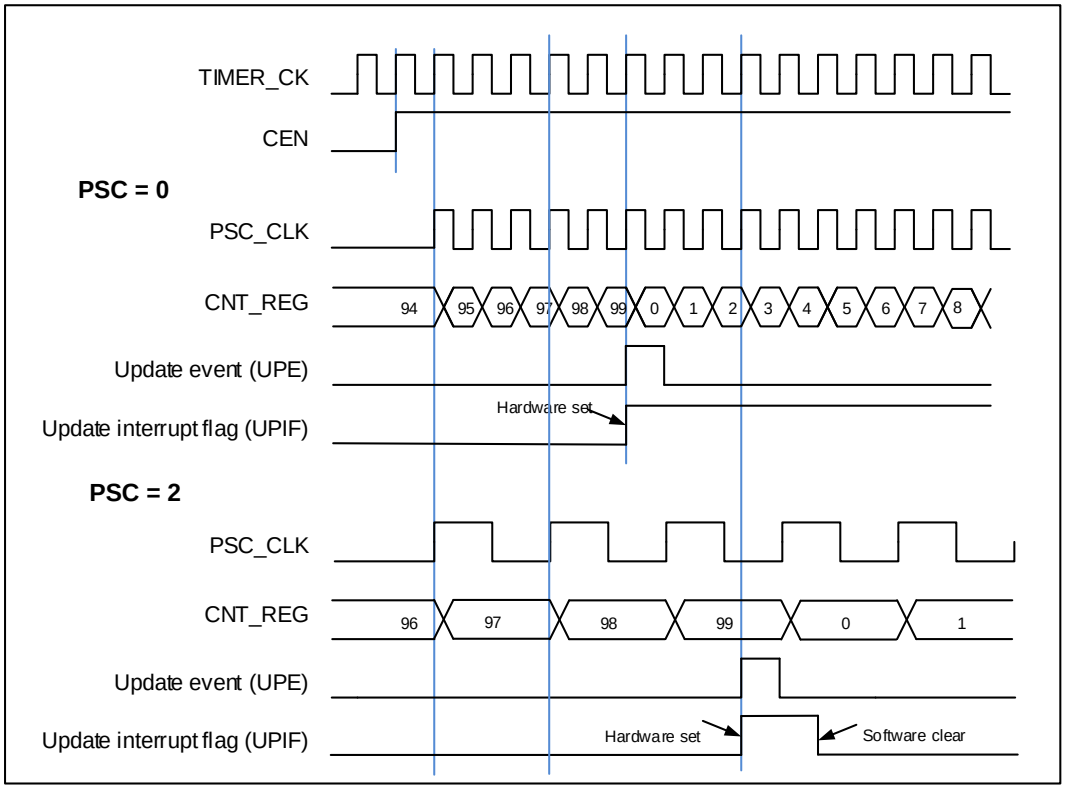
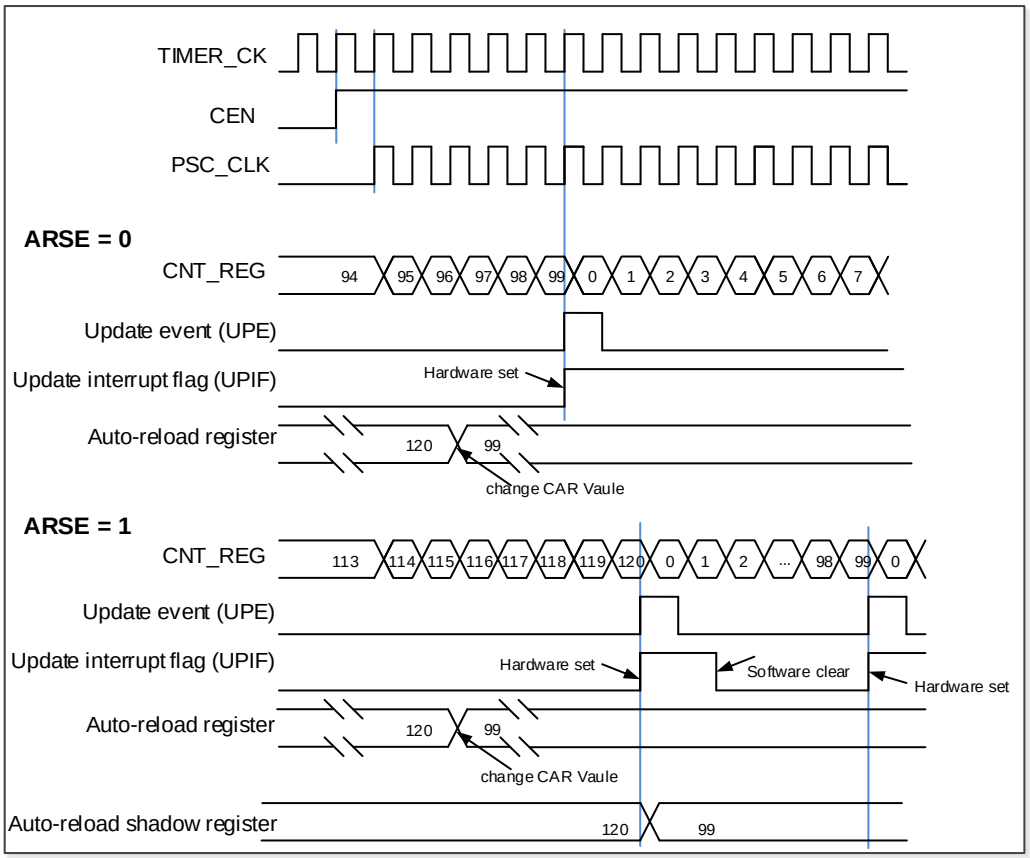


Figure 16-33. Timing chart of up counting mode, change TIMERx_CAR ongoing



Counter down counting

In this mode, the counter counts down continuously from the counter-reload value, which is defined in the TIMERx_CAR register, to 0 in a count-down direction. Once the counter reaches to 0, the counter will start counting down from the counter-reload value. The update event is generated at each counter underflow. The counting direction bit DIR in the TIMERx_CTL0 register should be set to 1 for the down-counting mode.

When the update event is set by the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to the counter-reload value and generates an update event.

If the UPDIS bit in TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (counter autoreload register, prescaler register) are updated.

Figure 16-34. Timing chart of down counting mode, PSC=0/2 show some examples of the counter behavior for different clock frequencies when TIMERx_CAR=0x99.

Figure 16-34. Timing chart of down counting mode, PSC=0/2

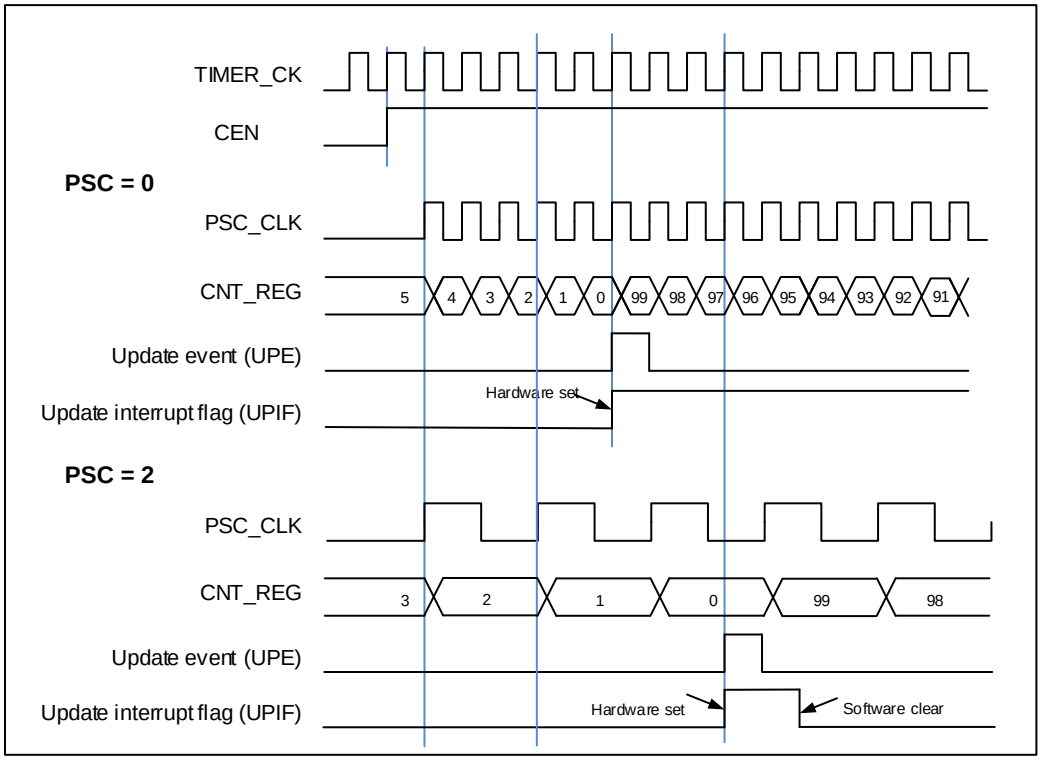
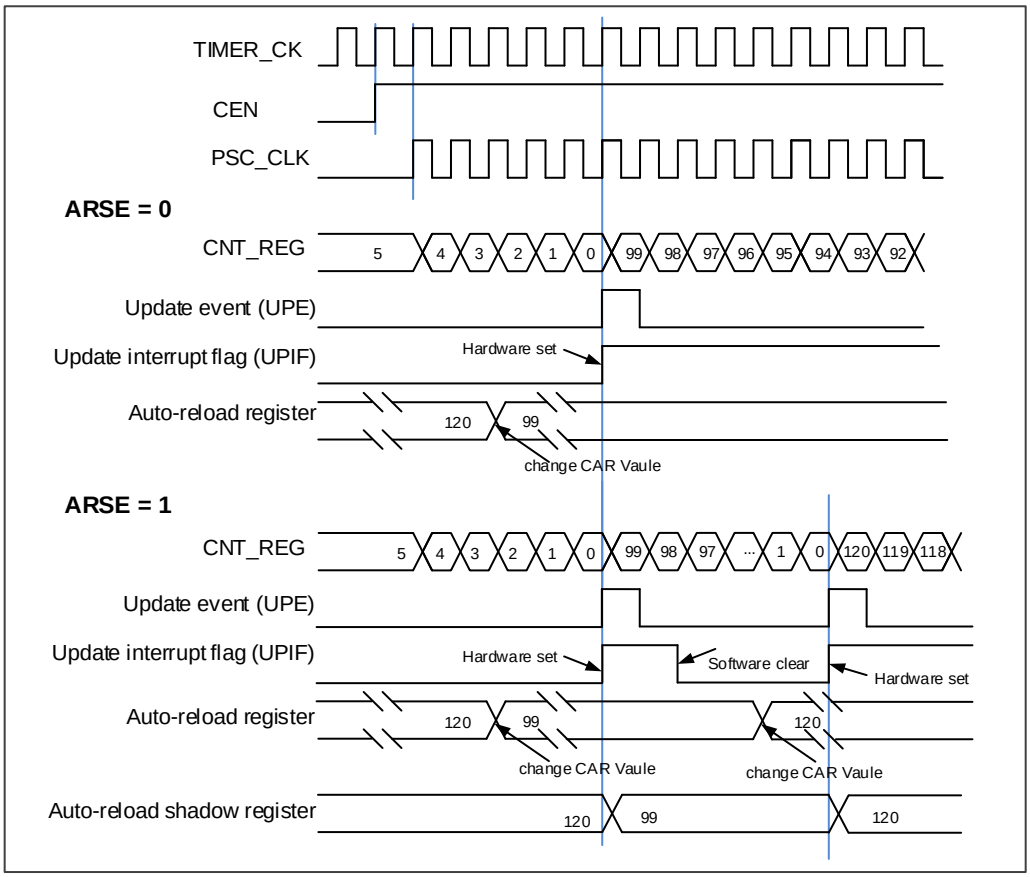


Figure 16-35. Timing chart of down counting mode, change TIMERx_CAR ongoing



Counter center-aligned counting

In this mode, the counter counts up from 0 to the counter-reload value and then counts down to 0 alternatively. The Timer module generates an overflow event when the counter counts to the counter-reload value subtract 1 in the up-counting direction and generates an underflow event when the counter counts to 1 in the down-counting mode. The counting direction bit DIR in the TIMERx_CTL0 register is read-only and indicates the counting direction when in the center-aligned mode.

Setting the UPG bit in the TIMERx_SWEVG register will initialize the counter value to 0 and generates an update event irrespective of whether the counter is counting up or down in the center-align counting mode.

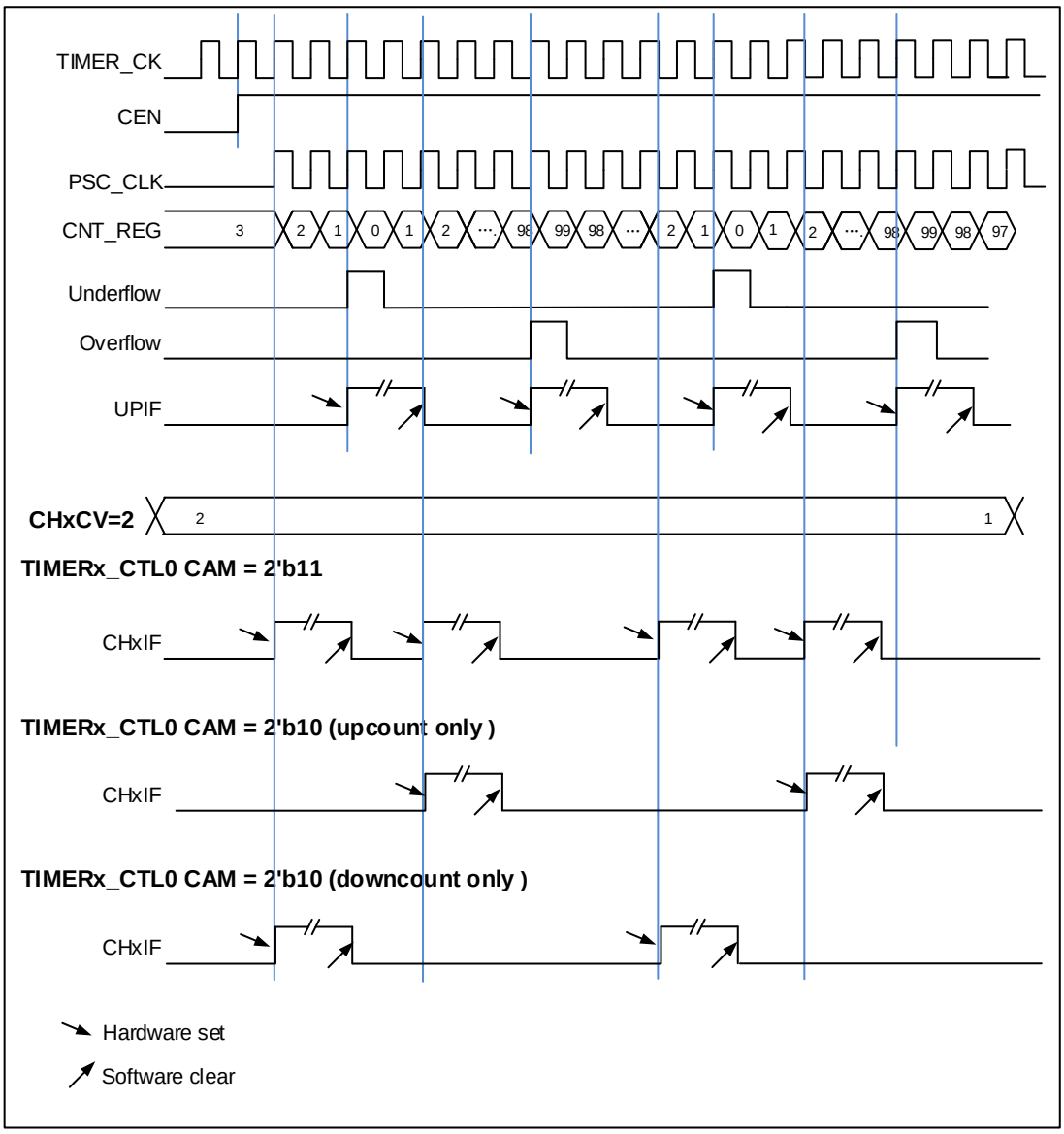
The UPIF bit in the TIMERx_INTF register can be set to 1 either when an underflow event or an overflow event occurs. While the CHxIF bit is associated with the value of CAM in TIMERx_CTL0. The details refer to ***Figure 16-36. Timing chart of center-aligned counting mode***

If the UPDIS bit in the TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (counter autoreload register, prescaler register) are updated.

Figure 16-36. Timing chart of center-aligned counting mode show some examples of the counter behavior when TIMERx_CAR=0x99. TIMERx_PSC=0x0

Figure 16-36. Timing chart of center-aligned counting mode



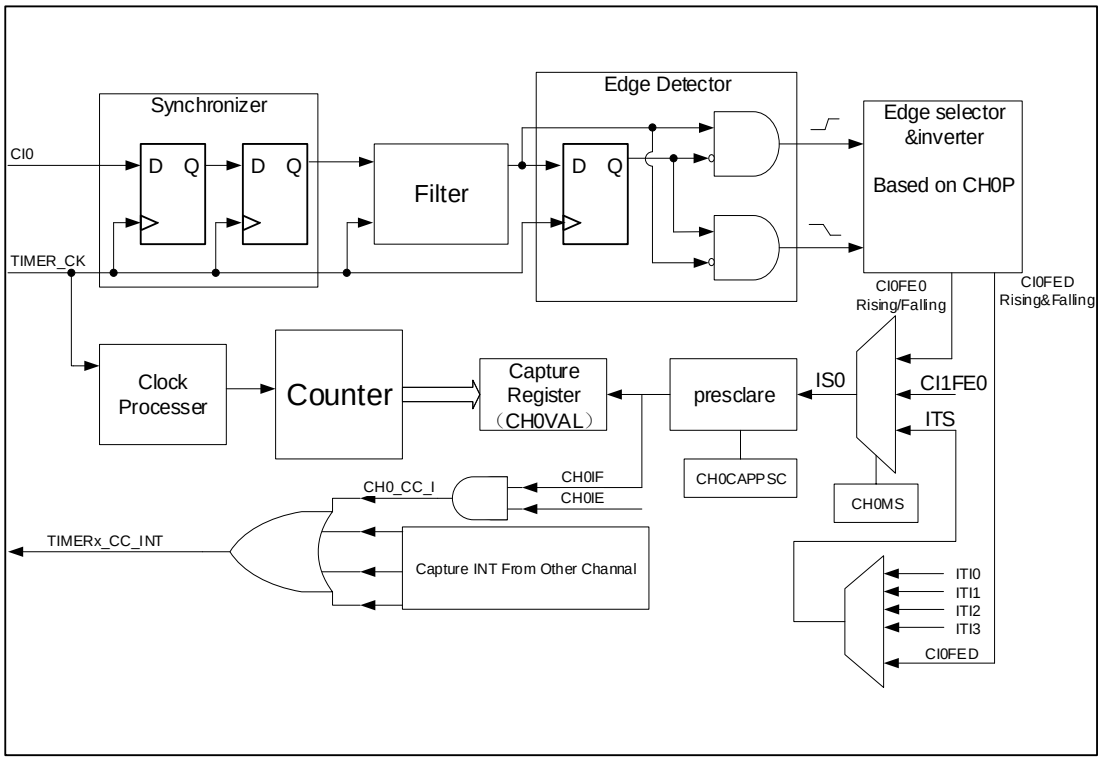
Input capture and output compare channels

The general level0 Timer has four independent channels which can be used as capture inputs or compare match outputs. Each channel is built around a channel capture compare register including an input stage, channel controller and an output stage.

■ Channel input capture function

Capture mode allows the channel to perform measurements such as pulse timing, frequency, period, duty cycle and so on. The input stage consists of a digital filter, a channel polarity selection, edge detection and a channel prescaler. When a selected edge occurs on the channel input, the current value of the counter is captured into the TIMERx_CHxCV register, at the same time the CHxIF bit is set and the channel interrupt is generated if enabled by CHxIE = 1.

Figure 16-37. Channel input capture principle



One of channels' input signals (CIx) can be chosen from the TIMERx_CHx signal or the Excusive-OR function of the TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 signals. First, the channel input signal (CIx) is synchronized to TIMER_CK domain, and then sampled by a digital filter to generate a filtered input signal. Then through the edge detector, the rising and falling edge are detected. You can select one of them by CHxP. One more selector is for the other channel and trig, controlled by CHxMS. The IC_prescaler make several the input event generate one effective capture event. On the capture event, TIMERx_CHxCV will restore the value of Counter.

So the process can be divided to several steps as below:

Step1: Filter Configuration. (CHxCAPFLT in TIMERx_CHCTL0)

Based on the input signal and requested signal quality, configure compatible CHxCAPFLT.

Step2: Edge Selection. (CHxP in TIMERx_CHCTL2)

Rising or falling edge, choose one by CHxP.

Step3: Capture source Selection. (CHxMS in TIMERx_CHCTL0)

As soon as you select one input capture source by CHxMS, you have set the channel to input mode (CHxMS!=0x0) and TIMERx_CHxCV cannot be written any more.

Step4: Interrupt enable. (CHxIE and CHxDEN in TIMERx_DMAINTEN)

Enable the related interrupt enable; you can got the interrupt and DMA request.

Step5: Capture enables. (CHxEN in TIMERx_CHCTL2)

Result: When you wanted input signal is got, TIMERx_CHxCV will be set by counter's value.

And CHxIF is asserted. If the CHxIF is high, the CHxOF will be asserted also. The interrupt and DMA request will be asserted based on the your configuration of CHxIE and CHxDEN in TIMERx_DMAINTEN

Direct generation: If you want to generate a DMA request or interrupt, you can set CHxG by software directly.

The input capture mode can be also used for pulse width measurement from signals on the TIMERx_CHx pins. For example, PWM signal connect to CI0 input. Select channel 0 capture signals to CI0 by setting CH0MS to 2'b01 in the channel control register (TIMERx_CHCTL0) and set capture on rising edge. Select channel 1 capture signal to CI0 by setting CH1MS to 2'b10 in the channel control register (TIMERx_CHCTL0) and set capture on falling edge. The counter set to restart mode and restart on channel 0 rising edge. Then the TIMERX_CH0CV can measure the PWM period and the TIMERx_CH1CV can measure the PWM duty.

- **Channel output compare function**

In channel output compare function, the TIMERx can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CHxVAL register of an output compare channel, the channel (n) output can be set, cleared, or toggled based on CHxCOMCTL. when the counter reaches the value in the CHxVAL register, the CHxIF bit is set and the channel (n) interrupt is generated if CHxIE = 1. And the DMA request will be assert, if CxCDE=1.

So the process can be divided to several steps as below:
Step1: Clock configuration. Such as clock source, clock prescaler and so on.

Step2: Compare mode configuration.
* Set the shadow enable mode by CHxCOMSEN
* Set the output mode (Set/Clear/Toggle) by CHxCOMCTL.
* Select the active high polarity by CHxP
* Enable the output by CHxEN

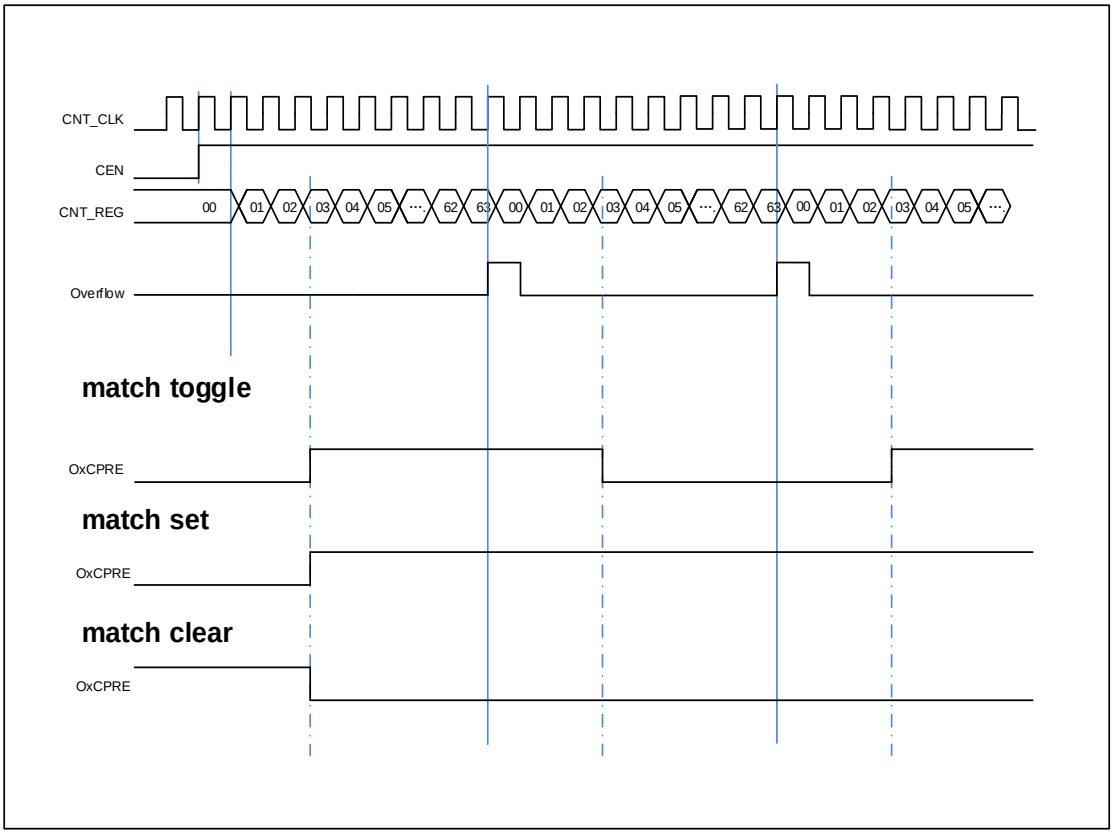
Step3: Interrupt/DMA-request enables configuration by CHxIE/CxCDE

Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV.
About the CHxVAL, you can change it on the go to meet the waveform you expected.

Step5: Start the counter by CEN.

The timechart below show the three compare modes toggle/set/clear. CAR=0x63, CHxVAL=0x3

Figure 16-38. Output-compare under three modes



Output PWM function

In the output PWM function (by setting the CHxCOMCTL bits to 3'b110 (PWM mode0) or to 3'b 111(PWM mode1), the channel can outputs PWM waveform according to the TIMERx_CAR registers and TIMERx_CHxCV registers.

Based on the counter mode, we have can also divide PWM into EAPWM (Edge aligned PWM) and CAPWM (Centre aligned PWM).

The EAPWM period is determined by TIMERx_CAR and duty cycle is by TIMERx_CHxCV. ***Figure 16-39. EAPWM timechart*** shows the EAPWM output and interrupts waveform.

The CAPWM period is determined by 2*TIMERx_CAR, and duty cycle is determined by 2*TIMERx_CHxCV. ***Figure 16-40. CAPWM timechart*** shows the CAPWM output and interrupts waveform.

If TIMERx_CHxCV is greater than TIMERx_CAR, the output will be always active under PWM mode0 (CHxCOMCTL==3'b110).
And if TIMERx_CHxCV is equal to zero, the output will be always inactive under PWM mode0 (CHxCOMCTL==3'b110).

Figure 16-39. EAPWM timechart

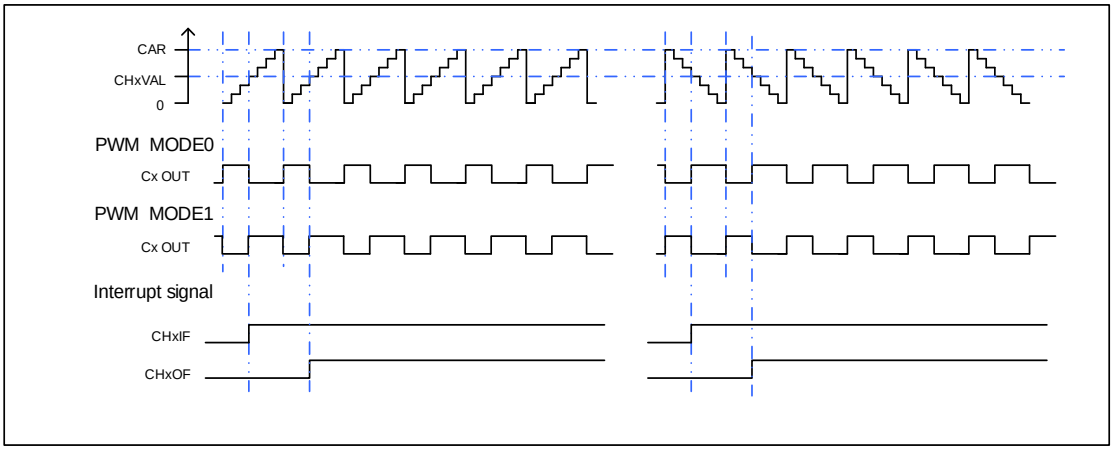
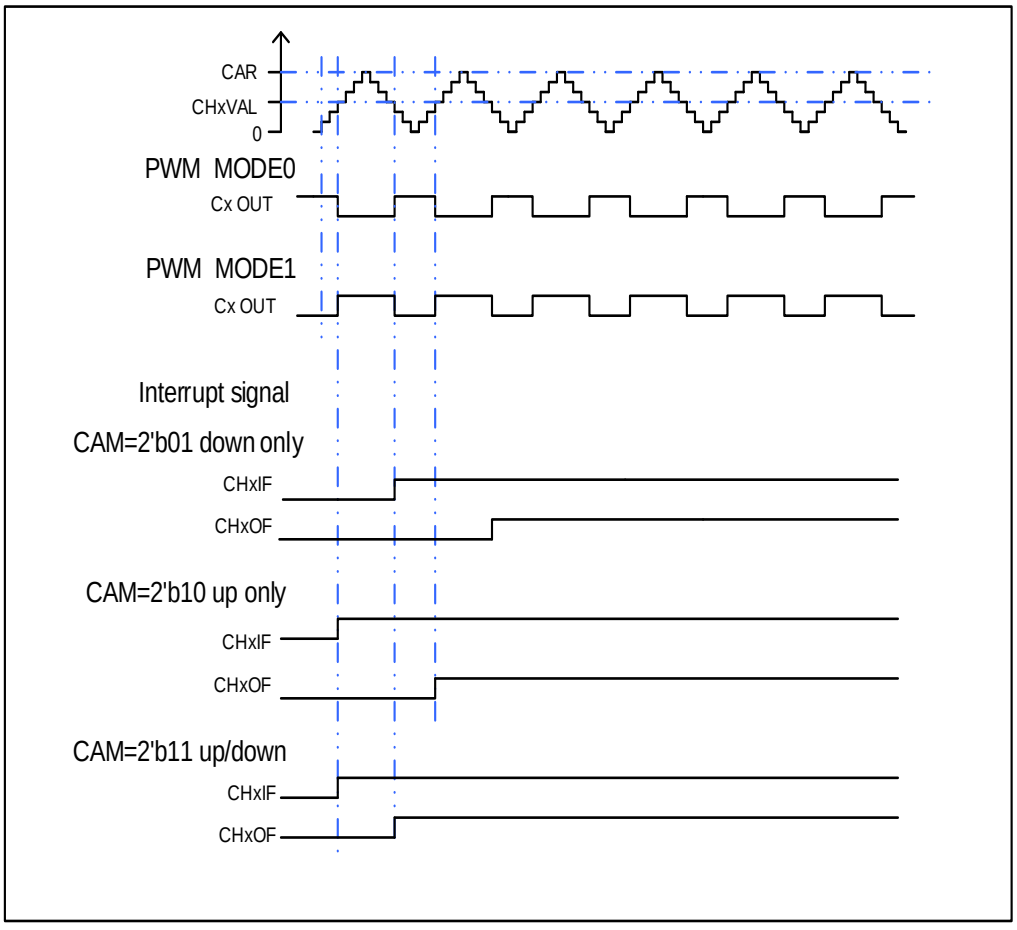


Figure 16-40. CAPWM timechart



Channel output prepare signal

When the TIMERx is used in the compare match output mode, the OxCPRE signal (Channel x Output prepare signal) is defined by setting the CHxCOMCTL filed. The OxCPRE signal has several types of output function. These include, keeping the original level by setting the CHxCOMCTL field to 0x00, set to 1 by setting the CHxCOMCTL field to 0x01, set to 0 by

setting the CHxCOMCTL field to 0x02 or signal toggle by setting the CHxCOMCTL field to 0x03 when the counter value matches the content of the TIMERx_CHxCV register.

The PWM mode 0 and PWM mode 1 outputs are also another kind of OxCPRE output which is setup by setting the CHxCOMCTL field to 0x06/0x07. In these modes, the OxCPRE signal level is changed according to the counting direction and the relationship between the counter value and the TIMERx_CHxCV content. With regard to a more detail description refer to the relative bit definition.

Another special function of the OxCPRE signal is a forced output which can be achieved by setting the CHxCOMCTL field to 0x04/0x05. Here the output can be forced to an inactive/active level irrespective of the comparison condition between the counter and the TIMERx_CHxCV values.

The OxCPRE signal can be forced to 0 when the ETIFE signal is derived from the external ETI pin and when it is set to a high level by setting the CHxCOMCEN bit to 1 in the TIMERx_CHCTL0 register. The OxCPRE signal will not return to its active level until the next update event occurs.

Quadrature decoder

Refer to ***Quadrature decoder***.

Hall sensor function

Refer to ***Hall sensor function***.

Master-slave management

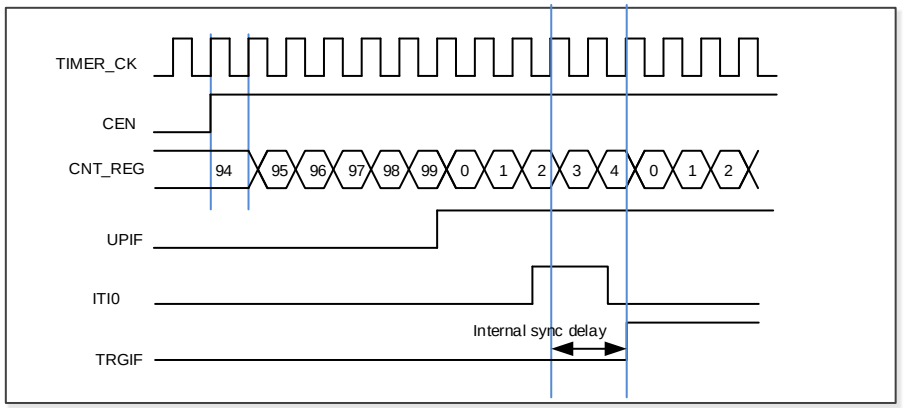
The TIMERx can be synchronized with a trigger in several modes including the restart mode, the pause mode and the event mode which is selected by the SMC[2:0] in the TIMERx_SMCFG register. The trigger input of these modes can be selected by the TRGS[2:0] in the TIMERx_SMCFG register.

Table 16-5. Examples of slave mode

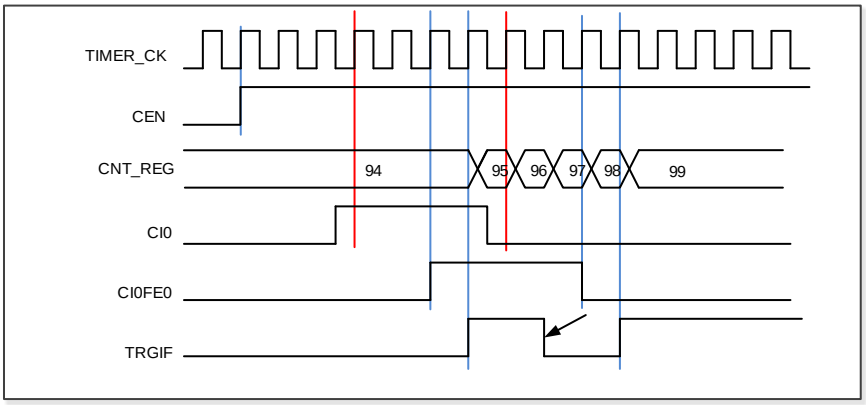
	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
LIST	SMC[2:0] 3'b100 (restart mode) 3'b101 (pause mode) 3'b110 (event mode)	TRGS[2:0] 000: ITI0 001: ITI1 010: ITI2 011: ITI3 100: CI0F_ED 101: CI0FE0 110: CI1FE1 111: ETIFP	If you choose the CI0FE0 or CI1FE1, configure the CHxP for the polarity selection and inversion. If you choose the ETIF, configure the ETP for polarity selection and inversion.	For the ITIx no filter and prescaler can be used. For the CIx, configure Filter by CHxCAPFLT, no prescaler can be used. For the ETIF, configure Filter by ETFC and Prescaler by ETPSC.

<table>
<tr><th></th><th>Mode Selection</th><th>Source Selection</th><th>Polarity Selection</th><th>Filter and Prescaler</th></tr>
<tr><td>Exam1</td><td>Restart mode

The counter can be clear and restart when a rising trigger input.</td><td>TRGS[2:0]=3’b000

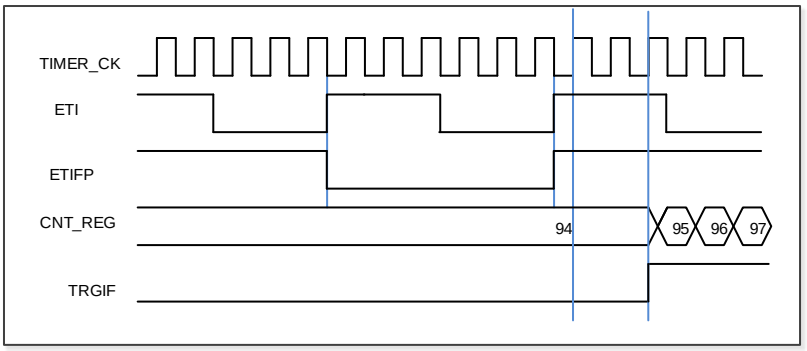
ITI0 is the selection.</td><td>-
For ITI0, no polarity selector can be used.</td><td>-
For the ITI0, no filter and prescaler can be used.</td></tr>
<tr><td colspan="4">Figure 16-41. Restart mode

</td></tr>
<tr><td>Exam2</td><td>Pause mode

The counter can be paused when the trigger input is low.</td><td>TRGS[2:0]=3’b101

CI0FE0 is the selection.</td><td>TI0S=0. （Non-xor），CH0P==0, no inverted. Capture will be sensitive to the rising edge only.</td><td>Filter is bypass in this example.</td></tr>
<tr><td colspan="4">Figure 16-42. Pause mode

</td></tr>
<tr><td>Exam3</td><td>Event mode
The counter will start to count when a rising trigger input.</td><td>TRGS[2:0]=3’b111

ETIF is the selection.</td><td>ETP = 0 no polarity change.</td><td>ETPSC = 1, divided by 2.

ETFC = 0 , no filter</td></tr>
</table>

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
	Figure 16-43. Event mode 			

Single pulse mode

Refer to ***Single pulse mode***.

Timers interconnection

Refer to ***Advanced timer (TIMERx, x=0, 7).***

Timer DMA mode

Timer's DMA mode is the function that configures timer's register by DMA module. The relative registers are TIMERx_DMACFG and TIMERx_DMATB; Of course, you have to enable a DMA request which will be asserted by some internal interrupt event. When the interrupt event was asserted, TIMERx will send a request to DMA, which is configured to M2P mode and PADDR is TIMERx_DMATB, then DMA will access the TIMERx_DMATB. In fact, register TIMERx_DMATB is only a buffer; timer will map the TIMERx_DMATB to an internal register, appointed by the field of DMATA in TIMERx_DMACFG . If the field of DMATC in TIMERx_DMACFG is 0(1 transfer), then the timer's DMA request is finished. While if TIMERx_DMATC is not 0, such as 3(4 transfers), then timer will send 3 more requests to DMA, and DMA will access timer's registers DMASAR+0x4, DMASAR+0x8, DMASAR+0xc at the next 3 accesses to TIMERx_DMATB. In one word, one time DMA internal interrupt event assert, DMATC+1 times request will be send by TIMERx.

If one more time DMA request event coming, TIMERx will repeat the process as above.

Timer debug mode

When the Cortex®-M4 halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register set to 1, the TIMERx counter stops.

16.2.5. TIMERx registers(x=1, 2, 3, 4)

TIMER1 base address: 0x4000 0000
TIMER2 base address: 0x4000 0400
TIMER3 base address: 0x4000 0800
TIMER4 base address: 0x4000 0C00

Control register 0 (TIMERx_CTL0)

Address offset: 0x00
Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKDIV[1:0]		ARSE	CAM[1:0]		DIR	SPM	UPS	UPDIS	CEN
						rw		rw	rw		rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	CKDIV[1:0]	Clock division The CKDIV bits can be configured by software to specify division factor between the CK_TIMER and the dead-time and digital filter sample clock (DTS). 00: $f_{DTS}=f_{CK_TIMER}$ 01: $f_{DTS}= f_{CK_TIMER} /2$ 10: $f_{DTS}= f_{CK_TIMER} /4$ 11: Reserved
7	ARSE	Auto-reload shadow enable 0: The shadow register for TIMERx_CAR register is disabled 1: The shadow register for TIMERx_CAR register is enabled
6:5	CAM[1:0]	Counter aligns mode selection 00: No center-aligned mode (edge-aligned mode). The direction of the counter is specified by the DIR bit. 01: Center-aligned and counting down assert mode. The counter counts under center-aligned and channel is configured in output mode (CHxMS=00 in TIMERx_CHCTL0 register). Only when counting down, CHxF bit can be set. 10: Center-aligned and counting up assert mode. The counter counts under center-aligned and channel is configured in output mode (CHxMS=00 in TIMERx_CHCTL0 register). Only when counting up, CHxF bit can be set. 11: Center-aligned and counting up/down assert mode. The counter counts under

		center-aligned and channel is configured in output mode (CHxMS=00 in TIMERx_CHCTL0 register). Both when counting up and counting down, CHxF bit can be set. After the counter is enabled, cannot be switched from 0x00 to non 0x00.
4	DIR	Direction 0: Count up 1: Count down If the timer work in center-aligned mode or encoder mode, this bit is read only.
3	SPM	Single pulse mode. 0: Single pulse mode disable. The counter continues after update event. 1: Single pulse mode enable. The counter counts until the next update event occurs.
2	UPS	Update source This bit is used to select the update event sources by software. 0: These events generate update interrupts or DMA requests: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: This event generates update interrupts or DMA requests: The counter generates an overflow or underflow event
1	UPDIS	Update disable. This bit is used to enable or disable the update event generation. 0: Update event enable. When an update event occurs, the corresponding shadow registers are loaded with their preloaded values. These events generate update event: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: Update event disable. **Note:** When this bit is set to 1, setting UPG bit or the restart mode does not generate an update event, but the counter and prescaler are initialized.
0	CEN	Counter enable 0: Counter disabled 1: Counter enabled The CEN bit must be set by software when timer works in external clock mode, pause mode or encoder mode.

Control register 1 (TIMERx_CTL1)

Address offset: 0x04
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TI0S	MMC[2:0]			DMAS	Reserved		
								rw	rw			rw			

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	TI0S	Channel 0 trigger input selection 0: The TIMERx_CH0 pin input is selected as channel 0 trigger input. 1: The result of combinational XOR of TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 pins is selected as channel 0 trigger input.
6:4	MMC[2:0]	Master mode control These bits control the selection of TRGO signal, which is sent by master timer to slave timer for synchronization function. 000: When a counter reset event occurs, a TRGO trigger signal is output. The counter resert source: Master timer generate a reset the UPG bit in the TIMERx_SWEVG register is set 001: Enable. When a conter start event occurs, a TRGO trigger signal is output. The counter start source : CEN control bit is set The trigger input in pause mode is high 010: When an update event occurs, a TRGO trigger signal is output. The update source depends on UPDIS bit and UPS bit. 011: When a capture or compare pulse event occurs in channel0, a TRGO trigger signal is output. 100: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O0CPRE. 101: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O1CPRE. 110: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O2CPRE. 111: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O3CPRE.
3	DMAS	DMA request source selection 0: When capture or compare event occurs, the DMA request of channel x is sent 1: When update event occurs, the DMA request of channel x is sent.
2:0	Reserved	Must be kept at reset value.

Slave mode configuration register (TIMERx_SMCFG)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	SMC1	ETPSC[1:0]		ETFC[3:0]				MSM	TRGS[2:0]			Reserved	SMC[2:0]		
rw	rw	rw		rw				rw	rw				rw		

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15	ETP	External trigger polarity This bit specifies the polarity of ETI signal 0: ETI is active at rising edge or high level . 1: ETI is active at falling edge or low level .
14	SMC1	Part of SMC for enable External clock mode1. In external clock mode 1, the counter is clocked by any active edge on the ETIFP signal. 0: External clock mode 1 disabled 1: External clock mode 1 enabled. When the slave mode is configured as restart mode, pause mode or event mode, the timer can still work in the external clock 1 mode by setting this bit. But the TRGS bits must not be 3'b111 in this case. The clock source of the timer will be ETIFP if external clock mode 0 and external clock mode 1 are configured at the same time. **Note:** External clock mode 0 enable is in this register's SMC[2:0] bit-filed.
13:12	ETPSC[1:0]	The prescaler of external trigger The frequency of external trigger signal ETIFP must not be at higher than 1/4 of TIMER_CK frequency. When the external trigger signal is a fast clock, the prescaler can be enabled to reduce ETIFP frequency. 00: Prescaler disable. 01: The prescaler is 2. 10: The prescaler is 4. 11: The prescaler is 8.
11:8	ETFC[3:0]	External trigger filter control The external trigger can be filtered by digital filter and this bit-field configure the filtering capability. Basic principle of digital filter: continuously sample the external trigger signal

according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit-field, it is considered to be an effective level.

The filtering capability configuration is as follows:

EXTFC[3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{TIMER_CK}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS_CK}/2$
4'b0101	8	
4'b0110	6	$f_{DTS_CK}/4$
4'b0111	8	
4'b1000	6	$f_{DTS_CK}/8$
4'b1001	8	
4'b1010	5	$f_{DTS_CK}/16$
4'b1011	6	
4'b1100	8	
4'b1101	5	$f_{DTS_CK}/32$
4'b1110	6	
4'b1111	8	

7 MSM

Master-slave mode

This bit can be used to synchronize selected timers to begin counting at the same time. The TRGI is used as the start event, and through TRGO, timers are connected together.

0: Master-slave mode disable

1: Master-slave mode enable

6:4 TRGS[2:0]

Trigger selection

This bit-field specifies which signal is selected as the trigger input, which is used to synchronize the counter.

000: ITI0

001: ITI1

010: ITI2

011: ITI3

100: CI0F_ED

101: CI0FE0

110: CI1FE1

111: ETIFP

These bits must not be changed when slave mode is enabled.

3 Reserved

Must be kept at reset value.

2:0 SMC[2:0]

Slave mode control

000: Disable mode. The slave mode is disabled; The prescaler is clocked directly

by the internal clock (TIMER_CK) when CEN bit is set high.
001: Quadrature decoder mode 0.The counter counts on CI0FE0 edge, while the direction depends on CI1FE1 level.
010: Quadrature decoder mode 1.The counter counts on CI1FE1 edge, while the direction depends on CI0FE0 level.
011: Quadrature decoder mode 2.The counter counts on both CI0FE0 and CI1FE1 edge, while the direction depends on each other.
100: Restart mode. The counter is reinitialized and an update event is generated on the rising edge of the selected trigger input.
101: Pause mode. The trigger input enables the counter clock when it is high and disables the counter clock when it is low.
110: Event mode. A rising edge of the trigger input enables the counter.
111: External clock mode 0. The counter counts on the rising edges of the selected trigger.

DMA and interrupt enable register (TIMERx_DMAINTEN)

Address offset: 0x0C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TRGDEN	Reserved	CH3DEN	CH2DEN	CH1DEN	CH0DEN	UPDEN	Reserved	TRGIE	Reserved	CH3IE	CH2IE	CH1IE	CH0IE	UPIE
	rw		rw	rw	rw	rw	rw		rw		rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14	TRGDEN	Trigger DMA request enable 0: Disabled 1: Enabled
13	Reserved	Must be kept at reset value.
12	CH3DEN	Channel 3 capture/compare DMA request enable 0: Disabled 1: Enabled
11	CH2DEN	Channel 2 capture/compare DMA request enable 0: Disabled 1: Enabled
10	CH1DEN	Channel 1 capture/compare DMA request enable

		0: Disabled 1: Enabled
9	CH0DEN	Channel 0 capture/compare DMA request enable 0: Disabled 1: Enabled
8	UPDEN	Update DMA request enable 0: Disabled 1: Enabled
7	Reserved	Must be kept at reset value.
6	TRGIE	Trigger interrupt enable 0: Disabled 1: Enabled
5	Reserved	Must be kept at reset value.
4	CH3IE	Channel 3 capture/compare interrupt enable 0: Disabled 1: Enabled
3	CH2IE	Channel 2 capture/compare interrupt enable 0: Disabled 1: Enabled
2	CH1IE	Channel 1 capture/compare interrupt enable 0: Disabled 1: Enabled
1	CH0IE	Channel 0 capture/compare interrupt enable 0: Disabled 1: Enabled
0	UPIE	Update interrupt enable 0: Disabled 1: Enabled

Interrupt flag register (TIMERx_INTF)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Reserved	CH3OF	CH2OF	CH1OF	CH0OF	Reserved	TRGIF	Reserved	CH3IF	CH2IF	CH1IF	CH0IF	UPIF
	rc_w0	rc_w0	rc_w0	rc_w0		rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12	CH3OF	Channel 3 over capture flag Refer to CH0OF description
11	CH2OF	Channel 2 over capture flag Refer to CH0OF description
10	CH1OF	Channel 1 over capture flag Refer to CH0OF description
9	CH0OF	Channel 0 over capture flag When channel 0 is configured in input mode, this flag is set by hardware when a capture event occurs while CH0IF flag has already been set. This flag is cleared by software. 0: No over capture interrupt occurred 1: Over capture interrupt occurred
8:7	Reserved	Must be kept at reset value.
6	TRGIF	Trigger interrupt flag This flag is set on trigger event and cleared by software. When in pause mode, both edges on trigger input generates a trigger event, otherwise, only an active edge on trigger input can generates a trigger event. 0: No trigger event occurred. 1: Trigger interrupt occurred.
5	Reserved	Must be kept at reset value.
4	CH3IF	Channel 3 capture/compare interrupt enable Refer to CH0IF description
3	CH2IF	Channel 2 capture/compare interrupt enable Refer to CH0IF description
2	CH1IF	Channel 1 capture/compare interrupt flag Refer to CH0IF description
1	CH0IF	Channel 0 capture/compare interrupt flag This flag is set by hardware and cleared by software. If channel 0 is in input mode, this flag is set when a capture event occurs. If channel 0 is in output mode, this flag is set when a compare event occurs. If channel 0 is set to input mode, this bit will be reset by reading TIMERx_CH0CV. 0: No channel 0 interrupt occurred 1: Channel 0 interrupt occurred

0	UPIF	Update interrupt flag This bit is set by hardware when an update event occurs and cleared by software. 0: No update interrupt occurred 1: Update interrupt occurred

Software event generation register (TIMERx_SWEVG)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TRGG	Reserved	CH3G	CH2G	CH1G	CH0G	UPG
									w		w	w	w	w	w

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	TRGG	Trigger event generation This bit is set by software and cleared by hardware automatically. When this bit is set, the TRGIF flag in TIMERx_STAT register will be set, related interrupt or DMA transfer can occur if enabled. 0: No generate a trigger event 1: Generate a trigger event
5	Reserved	Must be kept at reset value.
4	CH3G	Channel 3 capture or compare event generation Refer to CH0G description
3	CH2G	Channel 2 capture or compare event generation Refer to CH0G description
2	CH1G	Channel 1 capture or compare event generation Refer to CH0G description
1	CH0G	Channel 0 capture or compare event generation This bit is set by software to generate a capture or compare event in channel 0, it is automatically cleared by hardware. When this bit is set, the CH0IF flag will be set, and the corresponding interrupt or DMA request will be sent if enabled. In addition, if channel 0 is configured in input mode, the current value of the counter is captured to TIMERx_CH0CV register, and the CH0OF flag is set if the CH0IF flag has been set. 0: No generate a channel 0 capture or compare event

		1: Generate a channel 0 capture or compare event
0	UPG	This bit can be set by software, and automatically cleared by hardware. When this bit is set, the counter is cleared if the center-aligned or up counting mode is selected, while in down counting mode it takes the auto-reload value. The prescaler counter is cleared at the same time. 0: No generate an update event 1: Generate an update event

Channel control register 0 (TIMERx_CHCTL0)

Address offset: 0x18
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1COMCEN	CH1COMCTL[2:0]			CH1COMSEN	CH1COMFEN	CH1MS[1:0]		CH0COMCEN	CH0COMCTL[2:0]			CH0COMSEN	CH0COMFEN	CH0MS[1:0]	
CH1CAPFLT[3:0]				CH1CAPPSC[1:0]				CH0CAPFLT[3:0]				CH0CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	CH1COMCEN	Channel 1 output compare clear enable Refer to CH0COMCEN description
14:12	CH1COMCTL[2:0]	Channel 1 compare output control Refer to CH0COMCTL description
11	CH1COMSEN	Channel 1 output compare shadow enable Refer to CH0COMSEN description
10	CH1COMFEN	Channel 1 output compare fast enable Refer to CH0COMFEN description
9:8	CH1MS[1:0]	Channel 1 mode selection This bit-field specifies the direction of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH1EN bit in TIMERx_CHCTL2 register is reset). 00: Channel 1 is programmed as output mode 01: Channel 1 is programmed as input mode, IS1 is connected to CI1FE1 10: Channel 1 is programmed as input mode, IS1 is connected to CI0FE1 11: Channel 1 is programmed as input mode, IS1 is connected to ITS.

		Note: When CH1MS[1:0]=11, it is necessary to select an internal trigger input through TRGS bits in TIMERx_SMCFG register.
7	CH0COMCEN	Channel 0 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the O0CPRE signal will be cleared. 0: Channel 0 output compare clear disable 1: Channel 0 output compare clear enable
6:4	CH0COMCTL[2:0]	Channel 0 compare output control This bit-field specifies the compare output mode of the the output prepare signal O0CPRE. In addition, the high level of O0CPRE is the active level, and CH0_O and CH0_ON channels polarity depends on CH0P and CH0NP bits. 000: Timing mode. The O0CPRE signal keeps stable, independent of the comparison between the register TIMERx_CH0CV and the counter TIMERx_CNT. 001: Set the channel output. O0CPRE signal is forced high when the counter is equals to the output compare register TIMERx_CH0CV. 010: Clear the channel output. O0CPRE signal is forced low when the counter is equals to the output compare register TIMERx_CH0CV. 011: Toggle on match. O0CPRE toggles when the counter is equals to the output compare register TIMERx_CH0CV. 100: Force low. O0CPRE is forced to low level. 101: Force high. O0CPRE is forced to high level. 110: PWM mode0. When counting up, O0CPRE is high when the counter is smaller than TIMERx_CH0CV, and low otherwise. When counting down, O0CPRE is low when the counter is larger than TIMERx_CH0CV, and high otherwise. 111: PWM mode1. When counting up, O0CPRE is low when the counter is smaller than TIMERx_CH0CV, and high otherwise. When counting down, O0CPRE is high when the counter is larger than TIMERx_CH0CV, and low otherwise. If configured in PWM mode, the O0CPRE level changes only when the output compare mode is adjusted from “Timing” mode to “PWM” mode or the comparison result changes. This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH0MS bit-filed is 00(COMPARE MODE).
3	CH0COMSEN	Channel 0 compare output shadow enable When this bit is set, the shadow register of TIMERx_CH0CV register, which updates at each update event, will be enabled. 0: Channel 0 output compare shadow disable 1: Channel 0 output compare shadow enable The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1) This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH0MS bit-filed is 00.
2	CH0COMFEN	Channel 0 output compare fast enable

When this bit is set, the effect of an event on the trigger in input on the capture/compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH0_O is set to the compare level independently from the result of the comparison.
0: Channel 0 output quickly compare disable.
1: Channel 0 output quickly compare enable.

1:0	CH0MS[1:0]	Channel 0 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH0EN bit in TIMERx_CHCTL2 register is reset).). 00: Channel 0 is programmed as output mode 01: Channel 0 is programmed as input mode, IS0 is connected to CI0FE0 10: Channel 0 is programmed as input mode, IS0 is connected to CI1FE0 11: Channel 0 is programmed as input mode, IS0 is connected to ITS **Note:** When CH0MS[1:0]=11, it is necessary to select an internal trigger input through TRGS bits in TIMERx_SMCFG register.

Input capture mode:

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	CH1CAPFLT[3:0]	Channel 1 input capture filter control Refer to CH0CAPFLT description
11:10	CH1CAPPSC[1:0]	Channel 1 input capture prescaler Refer to CH0CAPPSC description
9:8	CH1MS[1:0]	Channel 1 mode selection Same as output compare mode
7:4	CH0CAPFLT[3:0]	Channel 0 input capture filter control The CI0 input signal can be filtered by digital filter and this bit-field configure the filtering capability. Basic principle of digital filter: continuously sample the CI0 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level. The filtering capability configuration is as follows:

CH0CAPFLT [3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{CK_TIMER}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	

		4’b0110	6	f_{DTS}/4
		4’b0111	8	
		4’b1000	6	f_{DTS}/8
		4’b1001	8	
		4’b1010	5	f_{DTS}/16
		4’b1011	6	
		4’b1100	8	
		4’b1101	5	f_{DTS}/32
		4’b1110	6	
		4’b1111	8	
3:2	CH0CAPPSC[1:0]	Channel 0 input capture prescaler This bit-field specifies the factor of the prescaler on channel 0 input. The prescaler is reset when CH0EN bit in TIMERx_CHCTL2 register is clear. 00: Prescaler disable, input capture occurs on every channel input edge 01: The input capture occurs on every 2 channel input edges 10: The input capture occurs on every 4 channel input edges 11: The input capture occurs on every 8 channel input edges		
1:0	CH0MS[1:0]	Channel 0 mode selection Same as output compare mode		

Channel control register 1 (TIMERx_CHCTL1)

Address offset: 0x1C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3COMCEN	CH3COMCTL[2:0]			CH3COMSEN	CH3COMFEN	CH3MS[1:0]		CH2COMCEN	CH2COMCTL[2:0]			CH2COMSEN	CH2COMFEN	CH2MS[1:0]	
CH3CAPFLT[3:0]				CH3CAPPSC[1:0]				CH2CAPFLT[3:0]				CH2CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	CH3COMCEN	Channel 3 output compare clear enable Refer to CH0COMCEN description
14:12	CH3COMCTL[2:0]	Channel 3 compare output control

		Refer to CH0COMCTL description
11	CH3COMSEN	Channel 3 output compare shadow enable Refer to CH0COMSEN description
10	CH3COMFEN	Channel 3 output compare fast enable Refer to CH0COMFEN description
9:8	CH3MS[1:0]	Channel 3 mode selection This bit-field specifies the direction of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH3EN bit in TIMERx_CHCTL2 register is reset). 00: Channel 3 is programmed as output mode 01: Channel 3 is programmed as input mode, IS3 is connected to CI3FE3 10: Channel 3 is programmed as input mode, IS3 is connected to CI2FE3 11: Channel 3 is programmed as input mode, IS3 is connected to ITS. **Note:** When CH3MS[1:0]=11, it is necessary to select an internal trigger input through TRGS bits in TIMERx_SMCFG register.
7	CH2COMCEN	Channel 2 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the O2CPRE signal will be cleared. 0: Channel 2 output compare clear disable 1: Channel 2 output compare clear enable
6:4	CH2COMCTL[2:0]	Channel 2 compare output control This bit-field specifies the compare output mode of the the output prepare signal O0CPRE. In addition, the high level of O0CPRE is the active level, and CH0_O and CH0_ON channels polarity depends on CH0P and CH0NP bits. 000: Timing mode. The O2CPRE signal keeps stable, independent of the comparison between the output compare register TIMERx_CH2CV and the counter TIMERx_CNT. 001: Set the channel output. O2CPRE signal is forced high when the counter is equals to the output compare register TIMERx_CH2CV. 010: Clear the channel output. O2CPRE signal is forced low when the counter is equals to the output compare register TIMERx_CH2CV. 011: Toggle on match. O2CPRE toggles when the counter is equals to the output compare register TIMERx_CH2CV. 100: Force low. O2CPRE is forced to low level. 101: Force high. O2CPRE is forced to high level. 110: PWM mode 0. When counting up, O2CPRE is high when the counter is smaller than TIMERx_CH2CV, and low otherwise. When counting down, O2CPRE is low when the counter is larger than TIMERx_CH2CV, and high otherwise. 111: PWM mode 1. When counting up, O2CPRE is low when the counter is smaller than TIMERx_CH2CV, and high otherwise. When counting down, O2CPRE is high when the counter is larger than TIMERx_CH2CV, and low otherwise. If configured in PWM mode, the O2CPRE level changes only when the output

		compare mode is adjusted from “Timing” mode to “PWM” mode or the comparison result changes. This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH2MS bit-filed is 00(COMPARE MODE).
3	CH2COMSEN	Channel 2 compare output shadow enable When this bit is set, the shadow register of TIMERx_CH2CV register, which updates at each update event will be enabled. 0: Channel 2 output compare shadow disable 1: Channel 2 output compare shadow enable The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1) This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH0MS bit-filed is 00.
2	CH2COMFEN	Channel 2 output compare fast enable When this bit is set, the effect of an event on the trigger in input on the capture/compare output will be accelerated if the channel is configured in PWM1 or PWM2 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH2_O is set to the compare level independently from the result of the comparison. 0: Channel 2 output quickly compare disable. 1: Channel 2 output quickly compare enable.
1:0	CH2MS[1:0]	Channel 2 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH2EN bit in TIMERx_CHCTL2 register is reset).). 00: Channel 2 is programmed as output mode 01: Channel 2 is programmed as input mode, IS2 is connected to CI2FE2 10: Channel 2 is programmed as input mode, IS2 is connected to CI3FE2 11: Channel 2 is programmed as input mode, IS2 is connected to ITS. **Note:** When CH2MS[1:0]=11, it is necessary to select an internal trigger input through TRGS bits in TIMERx_SMCFG register.

Input capture mode:

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	CH3CAPFLT[3:0]	Channel 3 input capture filter control Refer to CH0CAPFLT description
11:10	CH3CAPPSC[1:0]	Channel 3 input capture prescaler Refer to CH0CAPPSC description
9:8	CH3MS[1:0]	Channel 3 mode selection

Same as output compare mode

7:4 CH2CAPFLT[3:0] Channel 2 input capture filter control

The CI2 input signal can be filtered by digital filter and this bit-field configure the filtering capability.

Basic principle of digital filter: continuously sample the CI2 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level.

The filtering capability configuration is as follows:

CH2CAPFLT [3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{CK_TIMER}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	
4'b0110	6	$f_{DTS}/4$
4'b0111	8	
4'b1000	6	$f_{DTS}/8$
4'b1001	8	
4'b1010	5	$f_{DTS}/16$
4'b1011	6	
4'b1100	8	
4'b1101	5	$f_{DTS}/32$
4'b1110	6	
4'b1111	8	

3:2 CH2CAPPSC[1:0] Channel 2 input capture prescaler

This bit-field specifies the factor of the prescaler on channel 2 input. The prescaler is reset when CH2EN bit in TIMERx_CHCTL2 register is clear.

00: Prescaler disable, input capture occurs on every channel input edge

01: The input capture occurs on every 2 channel input edges

10: The input capture occurs on every 4 channel input edges

11: The input capture occurs on every 8 channel input edges

1:0 CH2MS[1:0] Channel 2 mode selection

Same as output compare mode

Channel control register 2 (TIMERx_CHCTL2)

Address offset: 0x20

Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).

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

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		CH3P	CH3EN	Reserved		CH2P	CH2EN	Reserved		CH1P	CH1EN	Reserved		CH0P	CH0EN
		rw	rw			rw	rw			rw	rw			rw	rw

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13	CH3P	Channel 3 capture/compare function polarity Refer to CH0P description
12	CH3EN	Channel 3 capture/compare function enable Refer to CH0EN description
11:10	Reserved	Must be kept at reset value.
9	CH2P	Channel 2 capture/compare function polarity Refer to CH0P description
8	CH2EN	Channel 2 capture/compare function enable Refer to CH0EN description
7:6	Reserved	Must be kept at reset value.
5	CH1P	Channel 1 capture/compare function polarity Refer to CH0P description
4	CH1EN	Channel 1 capture/compare function enable Refer to CH0EN description
3:2	Reserved	Must be kept at reset value.
1	CH0P	Channel 0 capture/compare function polarity When channel 0 is configured in output mode, this bit specifies the output signal polarity. 0: Channel 0 high level is active level 1: Channel 0 low level is active level When channel 0 is configured in input mode, this bit specifies the CI0 signal polarity. [CH0P=0]: The rising edge of CIxFE0 is the active signal for capture or trigger operation in slave mode. And CIxFE0 will not be inverted. [CH0P=1]: The falling edge of CIxFE0 is the active signal for capture or trigger operation in slave mode. And CIxFE0 will be inverted.
0	CH0EN	Channel 0 capture/compare function enable When channel 0 is configured in output mode, setting this bit enables CH0_O signal in active state. When channel 0 is configured in input mode, setting this bit enables the capture event in channel0. 0: Channel 0 disabled

1: Channel 0 enabled

Counter register (TIMERx_CNT)

Address offset: 0x24
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CNT[15:0]	This bit-field indicates the current counter value. Writing to this bit-field can change the value of the counter.

Prescaler register (TIMERx_PSC)

Address offset: 0x28
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC[15:0]	Prescaler value of the counter clock The TIMER_CK clock is divided by (PSC+1) to generate the counter clock. The value of this bit-filed will be loaded to the corresponding shadow register at every update event.

Counter auto reload register (TIMERx_CAR)

Address offset: 0x2C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CARL[15:0]	Counter auto reload value This bit-field specifies the auto reload value of the counter.

Channel 0 capture/compare value register (TIMERx_CH0CV)

Address offset: 0x34
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH0VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH0VAL[15:0]	Capture/compare value of channel0 When channel 0 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 0 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Channel 1 capture/compare value register (TIMERx_CH1CV)

Address offset: 0x38
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH1VAL[15:0]	Capture/compare value of channel1 When channel 1 is configured in input mode, this bit-field indicates the counter value at the last capture event.And this bit-field is read-only. When channel 1 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Configuration register (TIMERx_CFG)

Address offset: 0xFC
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	CHVSEL	Write CHxVAL register selection This bit-field set and reset by software. 1: If write the CHxVAL register, the write value is same as the CHxVAL value, the write access ignored 0: No effect
0	Reserved	Must be kept at reset value.

Channel 2 capture/compare value register (TIMERx_CH2CV)

Address offset: 0x3C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH2VAL[15:0]	Capture/compare value of channel 2 When channel 2 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 2 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Channel 3 capture/compare value register (TIMERx_CH3CV)

Address offset: 0x40
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH3VAL[15:0]	Capture/compare value of channel 3 When channel3 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 3 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the

shadow register updates by every update event.

DMA configuration register (TIMERx_DMACFG)

Address offset: 0x48
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DMATC[4:0]					Reserved			DMATA[4:0]				
			rw								rw				

Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:8	DMATC[4:0]	DMA transfer count This filed defines the number(n) of the register that DMA will access(R/W), n = (DMATC [4:0] +1). DMATC [4:0] is from 5'b0_0000 to 5'b1_0001.
7:5	Reserved	Must be kept at reset value.
4:0	DMATA[4:0]	DMA transfer access start address This field define the first address for the DMA access the TIMERx_DMATB. When access is done through the TIMERx_DMA address first time, this bit-field specifies the address you just access. And then the second access to the TIMERx_DMATB, you will access the address of start address + 0x4.

DMA transfer buffer register (TIMERx_DMATB)

Address offset: 0x4C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.

15:0	DMATB[15:0]	DMA transfer buffer When a read or write operation is assigned to this register, the register located at the address range (Start Addr + Transfer Timer* 4) will be accessed. The transfer Timer is calculated by hardware, and ranges from 0 to DMATC.

16.3. General level1 timer (TIMERx, x=8, 11)

16.3.1. Overview

The general level1 timer module (TIMER8,11) is a two-channel timer that supports input capture and output compare. They can generate PWM signals to control motor or be used for power management applications. The general level1 timer has a 16-bit counter that can be used as an unsigned counter.

In addition, the general level1 timers can be programmed and be used for counting, their external events can be used to drive other timers.

Timers are completely independent with each other, but they may be synchronized to provide a larger timer with their counter value increasing in unison.

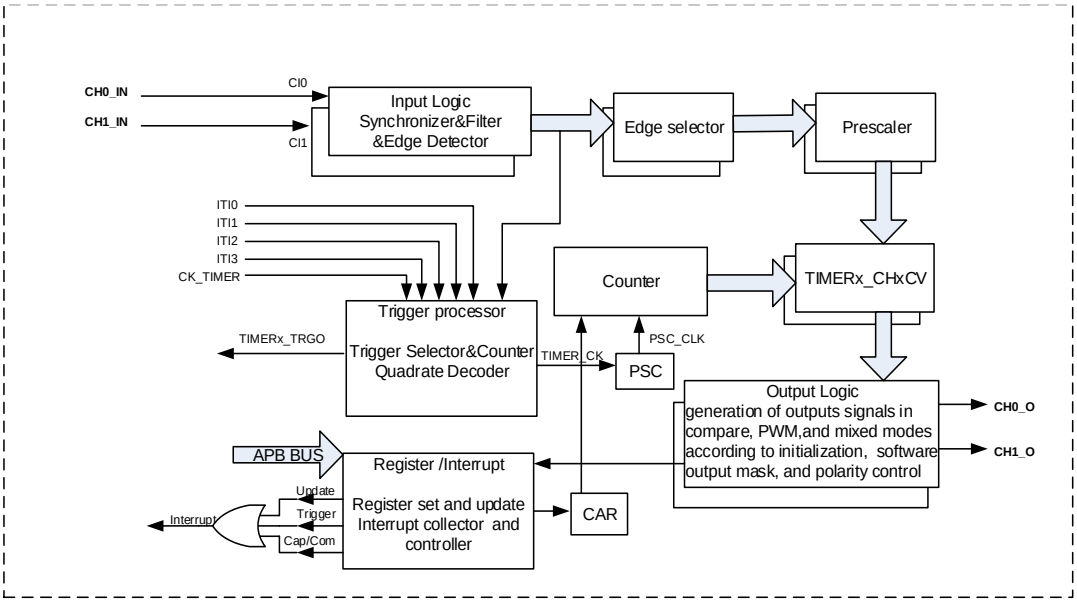
16.3.2. Characteristics

- Total channel num: 2.
- Counter width: 16 bits.
- Clock source of timer is selectable: internal clock, internal trigger, external input.
- Counter mode: up counting only.
- Programmable prescaler: 16 bits. The factor can be changed ongoing.
- Each channel is user-configurable: input capture mode, output compare mode, programmable PWM mode and single pulse mode.
- Auto reload function.
- Interrupt output: event update, trigger event and compare/capture event.
- Daisy chaining of timer modules allows a single timer to start multiple timers.
- Timer synchronization allows the selected timers to start counting on the same clock cycle.
- Timer master-slave management.

16.3.3. Block diagram

Figure 16-44. General level1 timer block diagram provides details on the internal configuration of the general level1 timer.

Figure 16-44. General level1 timer block diagram



16.3.4. Function overview

Clock source configuration

The general level1 TIMER has the capability of being clocked by either the CK_TIMER or an alternate clock source controlled by SMC (TIMERx_SMCFG bit [2:0]).

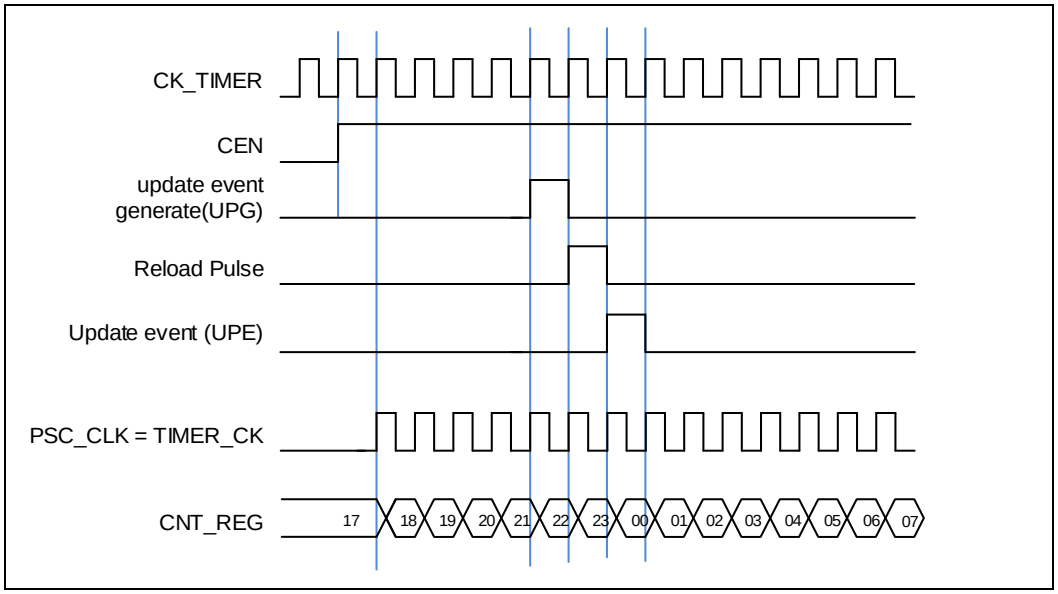
- SMC [2:0] == 3'b000. Internal timer clock CK_TIMER which is from module RCU.

The default internal clock source is the CK_TIMER used to drive the counter prescaler when the SMC [2:0] == 3'b000. When the CEN is set, the CK_TIMER will be divided by PSC value to generate PSC_CLK.

In this mode, the TIMER_CK, driven counter's prescaler to count, is equal to CK_TIMER which is from RCU.

If the SMC [2:0] in the TIMERx_SMCFG register are setting to an available value including 0x1, 0x2, 0x3 and 0x7, the prescaler is clocked by other clock sources selected by the TRGS [2:0] in the TIMERx_SMCFG register and described as follows. When the SMC bits are set to 0x4, 0x5 or 0x6, the internal clock CK_TIMER is the counter prescaler driving clock source.

Figure 16-45. Timing chart of internal clock divided by 1



- SMC [2:0] == 3'b111 (external clock mode 0). External input pin source

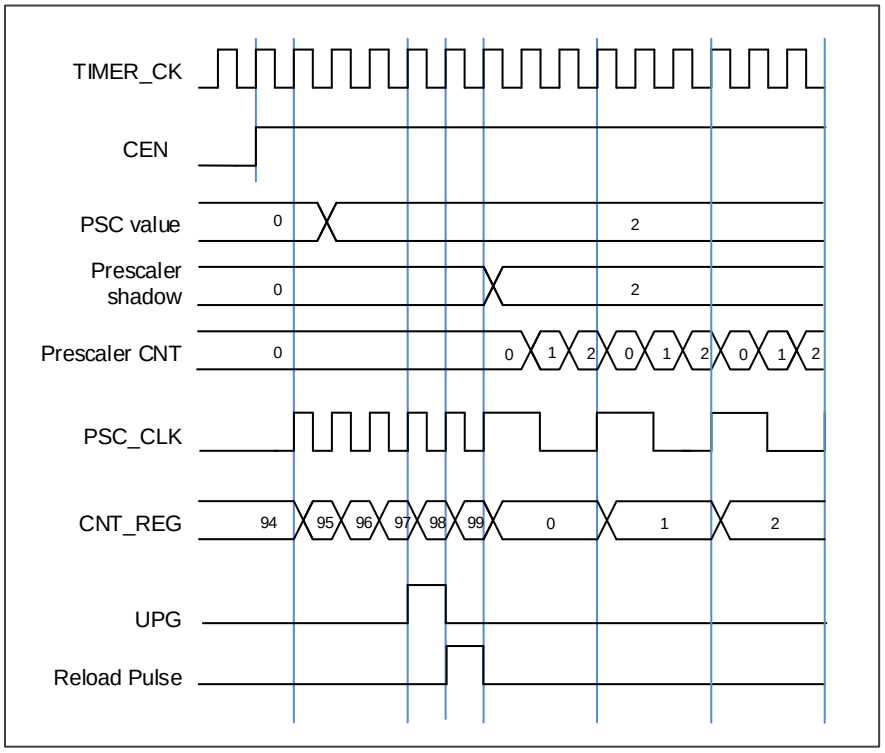
The TIMER_CK, driven counter's prescaler to count, can be triggered by the event of rising or falling edge on the external pin TIMERx_CI0/TIMERx_CI1. This mode can be selected by setting SMC [2:0] to 0x7 and the TRGS [2:0] to 0x4, 0x5 or 0x6.

And, the counter prescaler can also be driven by rising edge on the internal trigger input pin ITI0/1/2/3. This mode can be selected by setting SMC [2:0] to 0x7 and the TRGS [2:0] to 0x0, 0x1, 0x2 or 0x3.

Clock prescaler

The counter clock (PSC_CK) is obtained by the TIMER_CK through the prescaler, and the prescale factor can be configured from 1 to 65536 through the prescaler register (TIMERx_PSC). The new written prescaler value will not take effect until the next update event.

Figure 16-46. Timing chart of PSC value change from 0 to 2



Counter up counting

In this mode, the counter counts up continuously from 0 to the counter-reload value, which is defined in the TIMERx_CAR register, in a count-up direction. Once the counter reaches the counter reload value, the counter will start counting up from 0 again. The update event is generated at each counter overflow. The counting direction bit DIR in the TIMERx_CTL1 register should be set to 0 for the up counting mode.

When the update event is set by the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to 0 and generates an update event.

If the UPDIS bit in TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (counter auto reload register, prescaler register) are updated.

The following figures show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 16-47. Timing chart of up counting mode, PSC=0/2

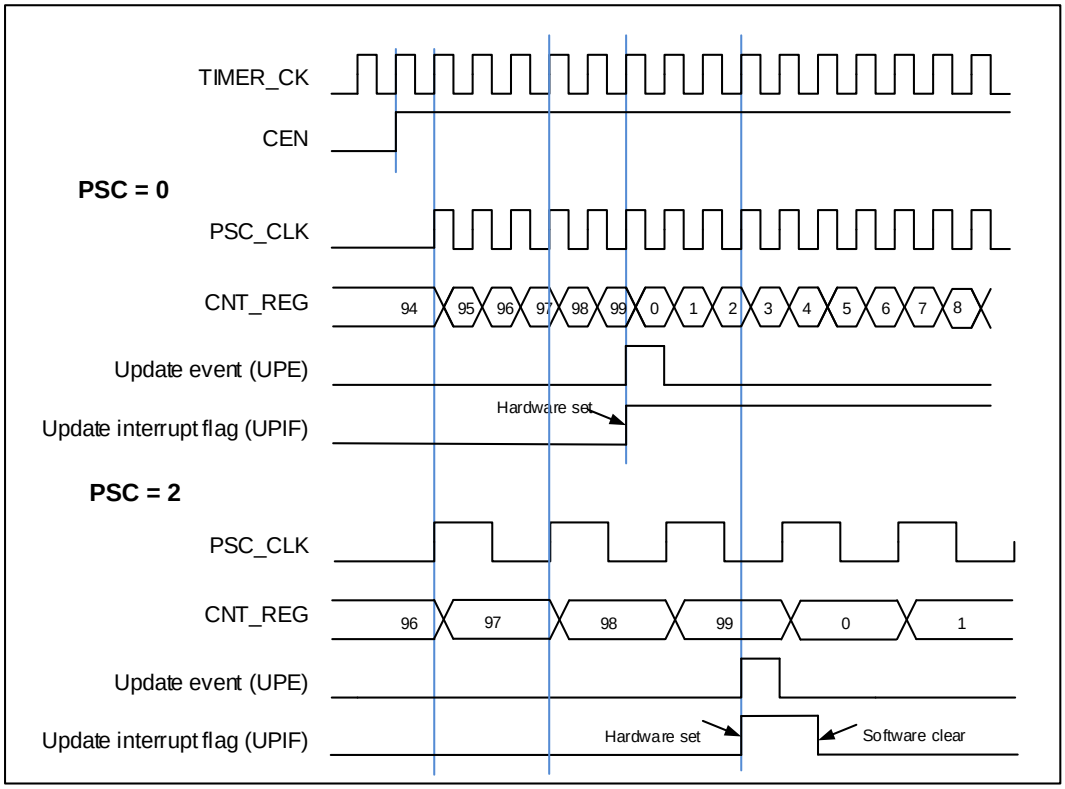
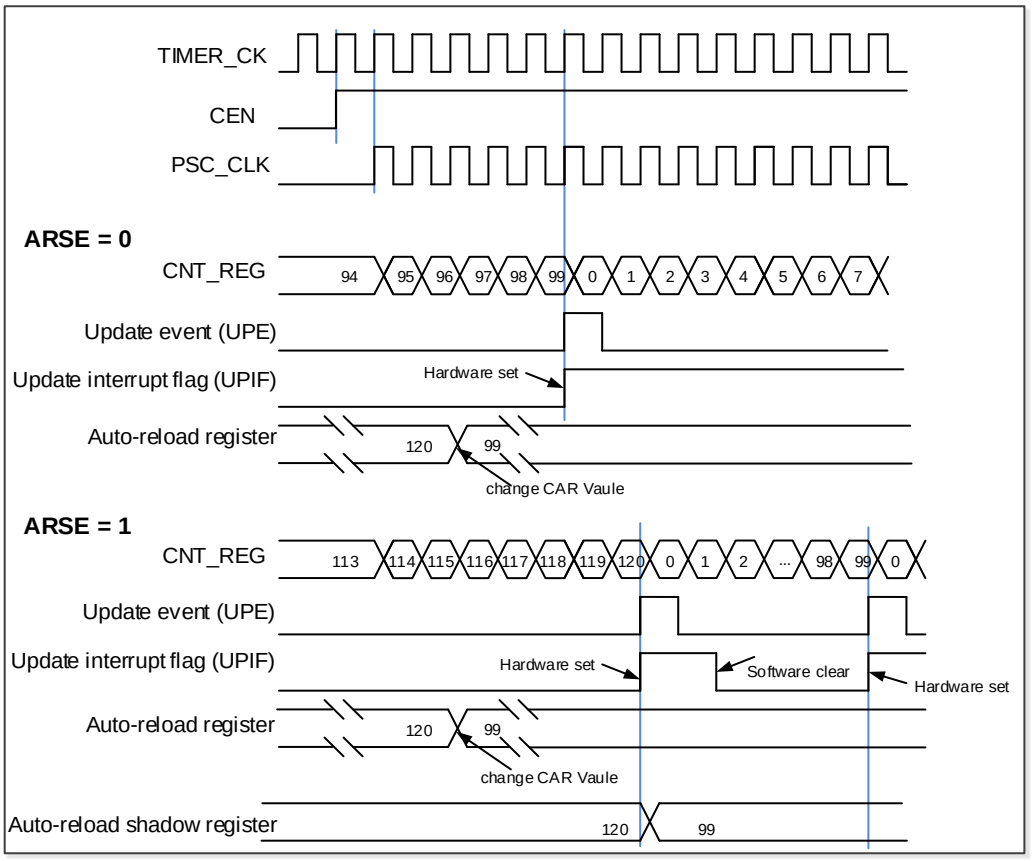


Figure 16-48. Timing chart of up counting mode, change TIMERx_CAR ongoing



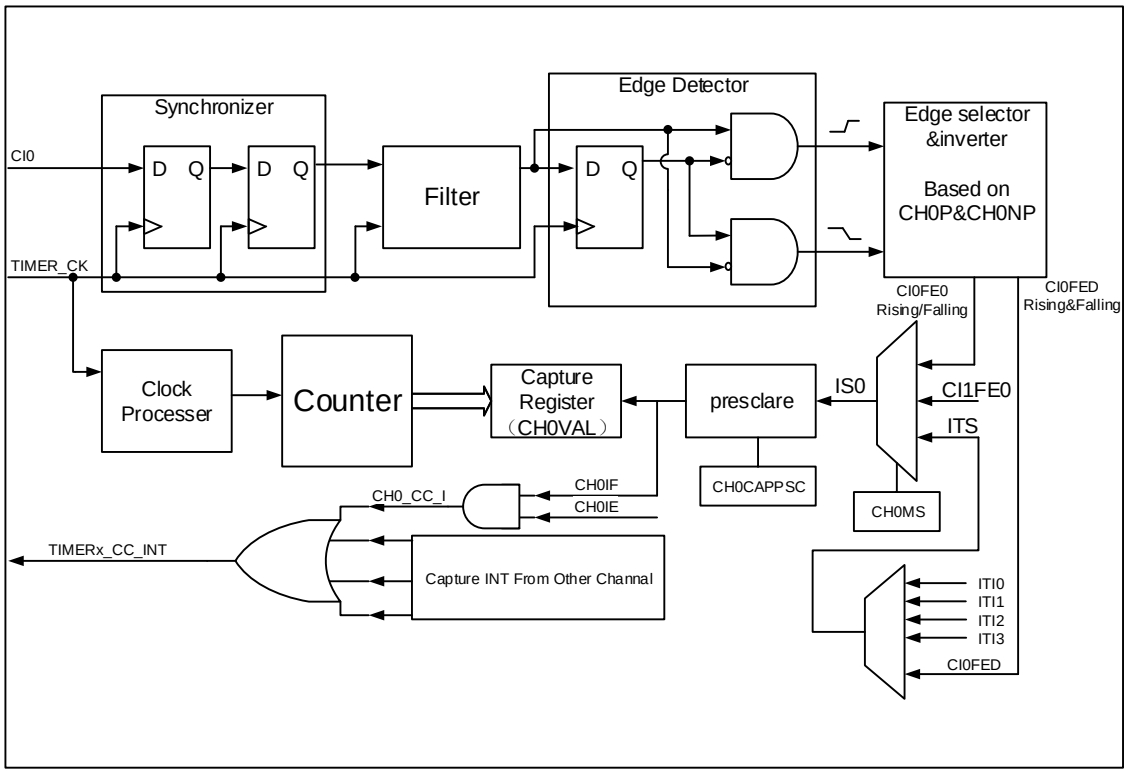
Input capture and output compare channels

The general level1 timer has two independent channels which can be used as capture inputs or compare match outputs. Each channel is built around a channel capture compare register including an input stage, channel controller and an output stage.

- **Channel input capture function**

Channel input capture function allows the channel to perform measurements such as pulse timing, frequency, period, duty cycle and so on. The input stage consists of a digital filter, a channel polarity selection, edge detection and a channel prescaler. When a selected edge occurs on the channel input, the current value of the counter is captured into the TIMERx_CHxCV register, at the same time the CHxIF bit is set and the channel interrupt is generated if enabled by CHxIE = 1.

Figure 16-49. Channel input capture principle



First, the channel input signal (CIx) is synchronized to TIMER_CK domain, and then sampled by a digital filter to generate a filtered input signal. Then through the edge detector, the rising and fall edge are detected. You can select one of them by CHxP. One more selector is for the other channel and trig, controlled by CHxMS. The IC_prescaler make several the input event generate one effective capture event. On the capture event, TIMERx_CHxCV will restore the value of counter.

So the process can be divided to several steps as below:

Step1: Filter configuration. (CHxCAPFLT in TIMERx_CHCTL0)

Based on the input signal and requested signal quality, configure compatible CHxCAPFLT.

Step2: Edge selection. (CHxP/CHxNP in TIMERx_CHCTL2)
Rising or falling edge, choose one by CHxP/CHxNP.

Step3: Capture source selection. (CHxMS in TIMERx_CHCTL0)
As soon as you select one input capture source by CHxMS, you have set the channel to input mode (CHxMS!=0x0) and TIMERx_CHxCV cannot be written any more.

Step4: Interrupt enable. (CHxIE and CHxDEN in TIMERx_DMAINTEN)
Enable the related interrupt enable; you can got the interrupt and DMA request.

Step5: Capture enables. (CHxEN in TIMERx_CHCTL2)

Result: When you wanted input signal is got, TIMERx_CHxCV will be set by Counter's value. And CHxIF is asserted. If the CHxIF is high, the CHxOF will be asserted also. The interrupt and DMA request will be asserted based on the your configuration of CHxIE and CHxDEN in TIMERx_DMAINTEN

Direct generation: If you want to generate a DMA request or Interrupt, you can set CHxG by software directly.

The channel input capture function can be also used for pulse width measurement from signals on the TIMERx_CHx pins. For example, PWM signal connect to CI0 input. Select channel 0 capture signals to CI0 by setting CH0MS to 2'b01 in the channel control register (TIMERx_CHCTL0) and set capture on rising edge. Select channel 1 capture signal to CI0 by setting CH1MS to 2'b10 in the channel control register (TIMERx_CHCTL0) and set capture on falling edge. The counter set to restart mode and restart on channel 0 rising edge. Then the TIMERX_CH0CV can measure the PWM period and the TIMERx_CH1CV can measure the PWM duty.

- **Channel output compare function**

In channel output compare function, the TIMERx can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CHxVAL register of an output compare channel, the channel (n) output can be set, cleared, or toggled based on CHxCOMCTL. when the counter reaches the value in the CHxVAL register, the CHxIF bit is set and the channel (n) interrupt is generated if CHxIE = 1. And the DMA request will be assert, if CxCDE=1.

So the process can be divided to several steps as below:
Step1: Clock configuration. Such as clock source, clock prescaler and so on.

Step2: Compare mode configuration.
* Set the shadow enable mode by CHxCOMSEN
* Set the output mode (Set/Clear/Toggle) by CHxCOMCTL.
* Select the active high polarity by CHxP/CHxNP
* Enable the output by CHxEN

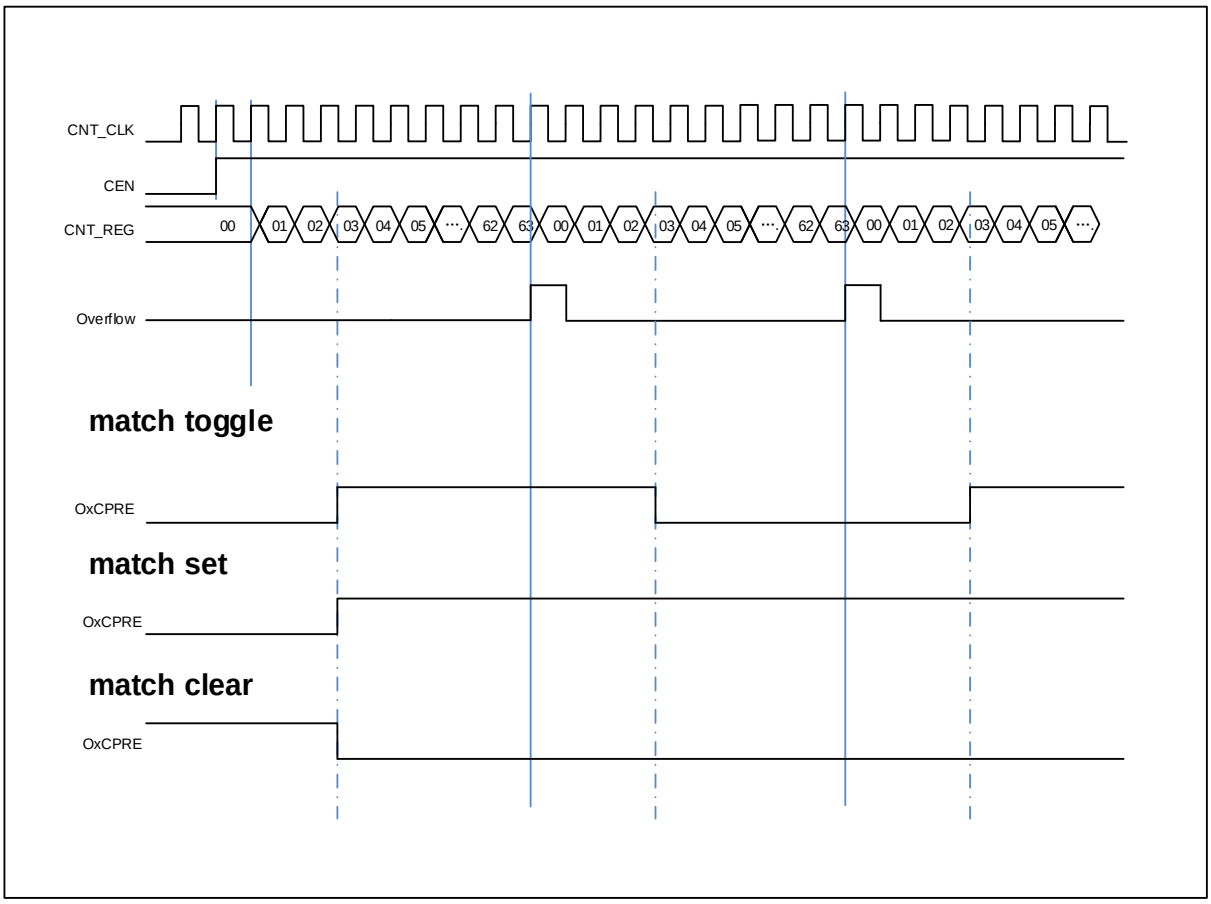
Step3: Interrupt/DMA-request enables configuration by CHxIE/CxCDE

Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV.
About the CHxVAL, you can change it on the go to meet the waveform you expected.

Step5: Start the counter by CEN.

The timechart below show the three compare modes toggle/set/clear. CAR=0x63, CHxVAL=0x3

Figure 16-50. Output-compare under three modes



Output PWM function

In the output PWM function (by setting the CHxCOMCTL bits to 3'b110 (PWM mode0) or to 3'b 111(PWM mode1), the channel can outputs PWM waveform according to the TIMERx_CAR registers and TIMERx_CHxCV registers.

Based on the counter mode, we have can also divide PWM into EAPWM (Edge aligned PWM) and CAPWM (Centre aligned PWM).

The EAPWM period is determined by TIMERx_CAR and duty cycle is by TIMERx_CHxCV. ***Figure 16-51. EAPWM timechart*** shows the EAPWM output and interrupts waveform.

The CAPWM period is determined by 2*TIMERx_CAR, and duty cycle is determined by 2*TIMERx_CHxCV. ***Figure 16-52. CAPWM timechart*** shows the CAPWM output and interrupt waveform.

If TIMERx_CHxCV is greater than TIMERx_CAR, the output will be always active under PWM mode0 (CHxCOMCTL==3'b110).
And if TIMERx_CHxCV is equal to zero, the output will be always inactive under PWM mode0

(CHxCOMCTL==3'b110).

Figure 16-51. EAPWM timechart

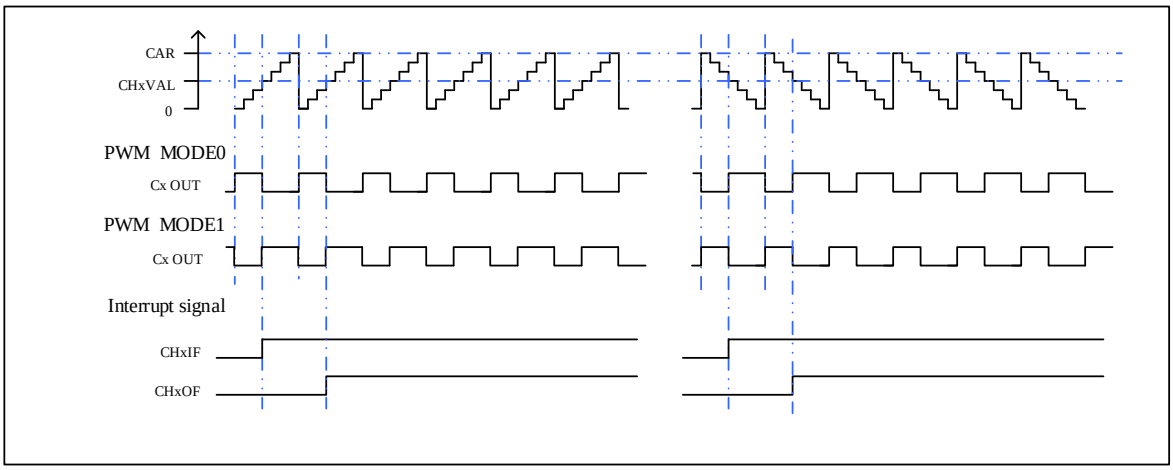
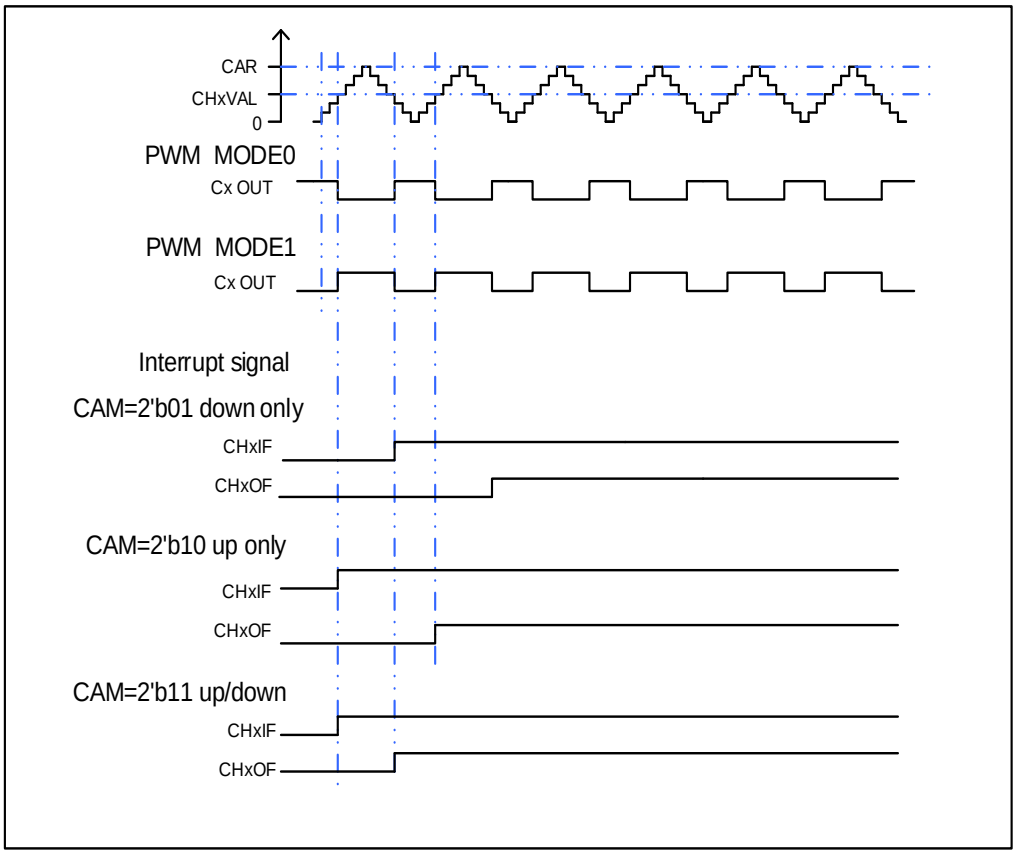


Figure 16-52. CAPWM timechart



Channel output prepare signal

When the TIMERx is used in the compare match output mode, the OxCPRE signal (Channel x Output prepare signal) is defined by setting the CHxCOMCTL filed. The OxCPRE signal has several types of output function. These include, keeping the original level by setting the CHxCOMCTL field to 0x00, set to 1 by setting the CHxCOMCTL field to 0x01, set to 0 by

setting the CHxCOMCTL field to 0x02 or signal toggle by setting the CHxCOMCTL field to 0x03 when the counter value matches the content of the TIMERx_CHxCV register.

The PWM mode 0 and PWM mode 1 outputs are also another kind of OxCPRE output which is setup by setting the CHxCOMCTL field to 0x06/0x07. In these modes, the OxCPRE signal level is changed according to the counting direction and the relationship between the counter value and the TIMERx_CHxCV content. With regard to a more detail description refer to the relative bit definition.

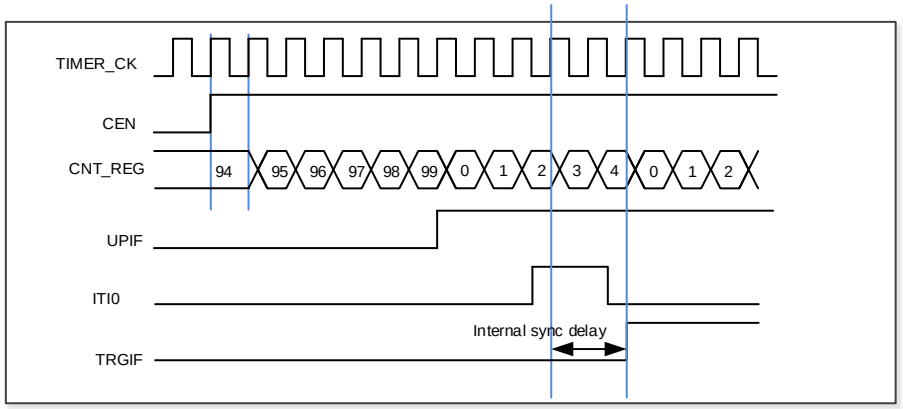
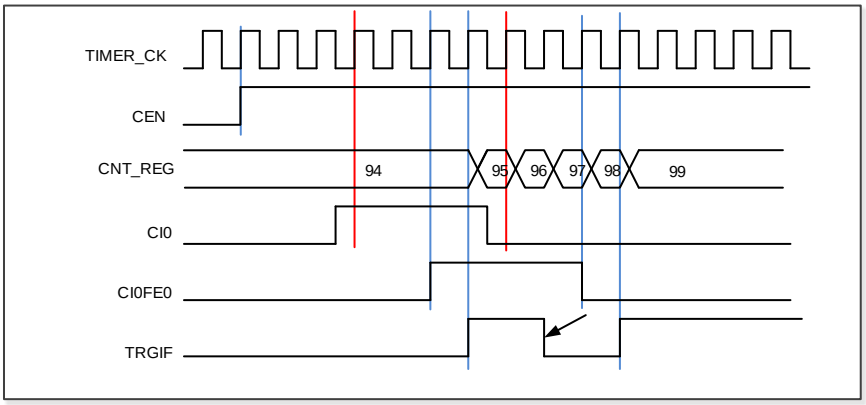
Another special function of the OxCPRE signal is a forced output which can be achieved by setting the CHxCOMCTL field to 0x04/0x05. Here the output can be forced to an inactive/active level irrespective of the comparison condition between the counter and the TIMERx_CHxCV values.

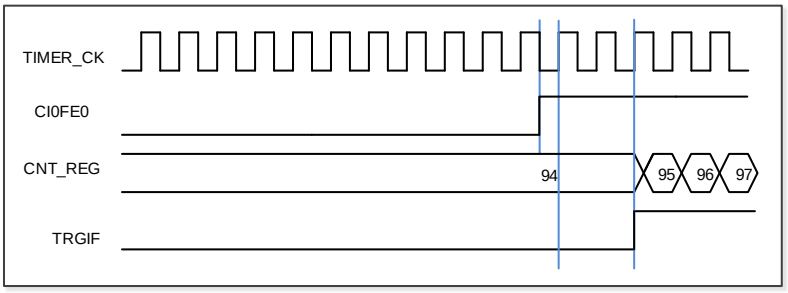
Master-slave management

The TIMERx can be synchronized with a trigger in several modes including the Restart mode, the Pause mode and the Event mode which is selected by the SMC [2:0] in the TIMERx_SMCFG register. The trigger input of these modes can be selected by the TRGS [2:0] in the TIMERx_SMCFG register.

Table 16-6. Examples of slave mode

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
LIST	SMC[2:0] 3'b100 (restart mode) 3'b101 (pause mode) 3'b110 (event mode)	TRGS[2:0] 000: ITI0 001: ITI1 010: ITI2 011: ITI3 100: CI0F_ED 101: CI0FE0 110: CI1FE1 111: Reserved	If you choose the CI0FE0 or CI1FE1, configure the CHxP and CHxNP for the polarity selection and inversion.	For the ITIx no filter and prescaler can be used. For the CIx, configure Filter by CHxCAPFLT, no prescaler can be used.
Exam1	Restart mode The counter can be clear and restart when a rising trigger input.	TRGS[2:0]=3'b000 ITI0 is the selection.	- For ITI0, no polarity selector can be used.	- For the ITI0, no filter and prescaler can be used.

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
	Figure 16-53. Restart mode 			
Exam2	Pause mode The counter can be paused when the trigger input is low.	TRGS[2:0]=3'b101 CI0FE0 is the selection.	CH0P==0, no inverted. Capture will be sensitive to the rising edge only.	Filter is bypass in this example.
	Figure 16-54. Pause mode 			
Exam3	Event mode The counter will start to count when a rising trigger input.	TRGS[2:0]=3'b101 CI0FE0 is the selection.	CH0P==0, no inverted.	Filter is bypass in this example.

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
	Figure 16-55. Event mode 			

Single pulse mode

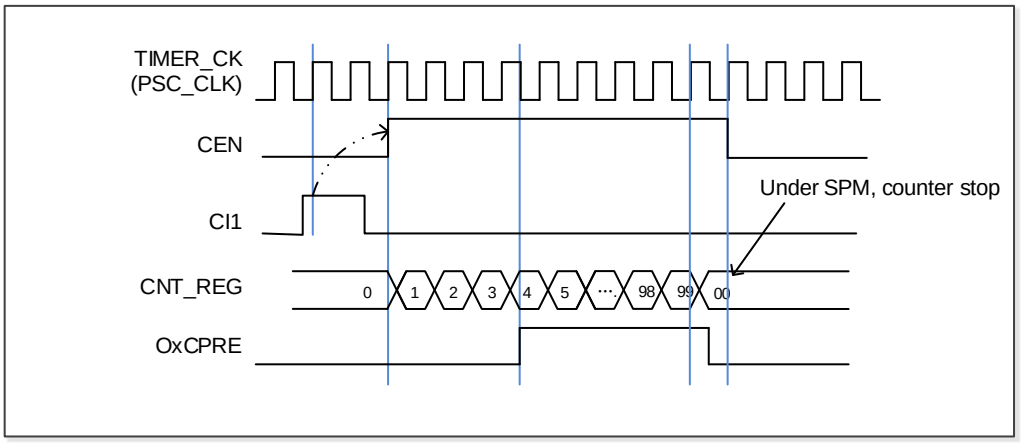
Single pulse mode is opposite to the repetitive mode, which can be enabled by setting SPM in TIMERx_CTL0. When you set SPM, the counter will be clear and stop when the next update event. In order to get pulse waveform, you can set the TIMERx to PWM mode or compare by CHxCOMCTL.

Once the timer is set to operate in the single pulse mode, it is not necessary to set the timer enable bit CEN in the TIMERx_CTL0 register to 1 to enable the counter. The trigger to generate a pulse can be sourced from the trigger signals edge or by setting the CEN bit to 1 using software. Setting the CEN bit to 1 or a trigger from the trigger signals edge can generate a pulse and then keep the CEN bit at a high state until the update event occurs or the CEN bit is written to 0 by software. If the CEN bit is cleared to 0 using software, the counter will be stopped and its value held.

In the single pulse mode, the trigger active edge which sets the CEN bit to 1 will enable the counter. However, there exist several clock delays to perform the comparison result between the counter value and the TIMERx_CHxCV value. In order to reduce the delay to a minimum value, the user can set the CHxCOMFEN bit in each TIMERx_CHCTL0/1 register. After a trigger rising occurs in the single pulse mode, the OxCPRE signal will immediately be forced to the state which the OxCPRE signal will change to, as the compare match event occurs without taking the comparison result into account. The CHxCOMFEN bit is available only when the output channel is configured to operate in the PWM0 or PWM1 output mode and the trigger source is derived from the trigger signal.

Figure 16-56. Single pulse mode TIMERx_CHxCV = 4 TIMERx_CAR=99 shows an example.

Figure 16-56. Single pulse mode TIMERx_CHxCV = 4 TIMERx_CAR=99



Timers interconnection

Refer to ***Advanced timer (TIMERx, x=0, 7)***.

Timer debug mode

When the Cortex®-M4 halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register set to 1, the TIMERx counter stops.

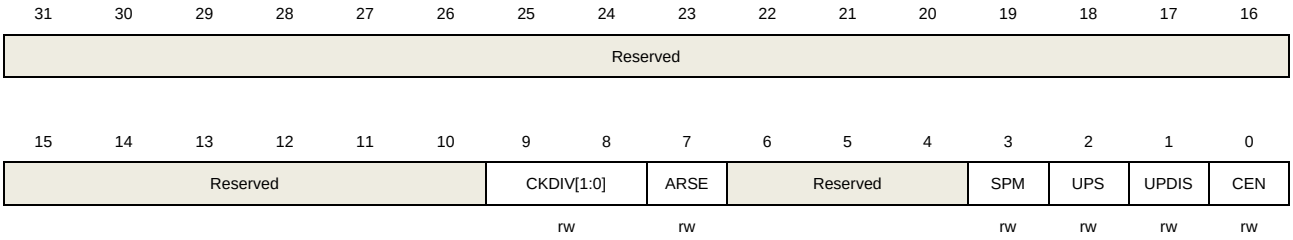
16.3.5. TIMERx registers (x=8, 11)

TIMER8 base address: 0x4001 4C00
TIMER11 base address: 0x4000 1800

Control register 0 (TIMERx_CTL0)

Address offset: 0x00
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	CKDIV[1:0]	Clock division The CKDIV bits can be configured by software to specify division factor between the CK_TIMER and the dead-time and digital filter sample clock (DTS). 00: $f_{DTS}=f_{CK_TIMER}$ 01: $f_{DTS}= f_{CK_TIMER}$ /2 10: $f_{DTS}= f_{CK_TIMER}$ /4 11: Reserved
7	ARSE	Auto-reload shadow enable 0: The shadow register for TIMERx_CAR register is disabled 1: The shadow register for TIMERx_CAR register is enabled
6:4	Reserved	Must be kept at reset value.
3	SPM	Single pulse mode. 0: Single pulse mode disable. The counter continues after update event. 1: Single pulse mode enable. The counter counts until the next update event occurs.
2	UPS	Update source This bit is used to select the update event sources by software. 0: These events generate update interrupts or DMA requests: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: This event generates update interrupts or DMA requests:

		The counter generates an overflow or underflow event
1	UPDIS	Update disable. This bit is used to enable or disable the update event generation. 0: Update event enable. When an update event occurs, the corresponding shadow registers are loaded with their preloaded values. These events generate update event: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: Update event disable. **Note:** When this bit is set to 1, setting UPG bit or the restart mode does not generate an update event, but the counter and prescaler are initialized.
0	CEN	Counter enable 0: Counter disable 1: Counter enable The CEN bit must be set by software when timer works in external clock mode, pause mode or encoder mode.

Slave mode configuration register (TIMERx_SMCFG)

Address offset: 0x08
Reset value: 0x0000

This register can be accessed by half-word (16-bit) or word (32-bit)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								MSM	TRGS[2:0]			Reserved	SMC[2:0]		
								rw	rw			rw	rw		

Bits	Fields	Descriptions
15:8	Reserved	Must be kept at reset value.
7	MSM	Master-slave mode This bit can be used to synchronize selected timers to begin counting at the same time. The TRGI is used as the start event, and through TRGO, timers are connected together. 0: Master-slave mode disable 1: Master-slave mode enable
6:4	TRGS[2:0]	Trigger selection This bit-field specifies which signal is selected as the trigger input, which is used to synchronize the counter. 000: ITI0 001: ITI1 010: ITI2

		011: ITI3 100: CI0F_ED 101: CI0FE0 110: CI1FE1 111: Reserved. These bits must not be changed when slave mode is enabled.
3	Reserved	Must be kept at reset value.
2:0	SMC[2:0]	Slave mode control 000: Disable mode. The slave mode is disabled; The prescaler is clocked directly by the internal clock (TIMER_CK) when CEN bit is set high. 001: Reserved. 010: Reserved. 011: Reserved. 100: Restart mode. The counter is reinitialized and an update event is generated on the rising edge of the selected trigger input. 101: Pause mode. The trigger input enables the counter clock when it is high and disables the counter clock when it is low. 110: Event mode. A rising edge of the trigger input enables the counter. 111: External clock mode0. The counter counts on the rising edges of the selected trigger.

Interrupt enable register (TIMERx_DMAINTEN)

Address offset: 0x0C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TRGIE	Reserved			CH1IE	CH0IE	UPIE
									rw				rw	rw	rw

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	TRGIE	Trigger interrupt enable 0: Disabled 1: Enabled
5:3	Reserved	Must be kept at reset value.

2	CH1IE	Channel 1 capture/compare interrupt enable 0: Disabled 1: Enabled
1	CH0IE	Channel 0 capture/compare interrupt enable 0: Disabled 1: Enabled
0	UPIE	Update interrupt enable 0: Disabled 1: Enabled

Interrupt flag register (TIMERx_INTF)

Address offset: 0x10
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					CH1OF	CH0OF	Reserved		TRGIF	Reserved			CH1IF	CH0IF	UPIF
					rc_w0	rc_w0			rc_w0				rc_w0	rc_w0	rc_w0

Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10	CH1OF	Channel 1 over capture flag Refer to CH0OF description
9	CH0OF	Channel 0 over capture flag When channel 0 is configured in input mode, this flag is set by hardware when a capture event occurs while CH0IF flag has already been set. This flag is cleared by software. 0: No over capture interrupt occurred 1: Over capture interrupt occurred
8:7	Reserved	Must be kept at reset value.
6	TRGIF	Trigger interrupt flag This flag is set on trigger event and cleared by software. When in pause mode, both edges on trigger input generates a trigger event, otherwise, only an active edge on trigger input can generates a trigger event. 0: No trigger event occurred. 1: Trigger interrupt occurred.

5:3	Reserved	Must be kept at reset value.
2	CH1IF	Channel 1 capture/compare interrupt flag Refer to CH0IF description
1	CH0IF	Channel 0 capture/compare interrupt flag This flag is set by hardware and cleared by software. If channel 0 is in input mode, this flag is set when a capture event occurs. If channel 0 is in output mode, this flag is set when a compare event occurs. If channel 0 is set to input mode, this bit will be reset by reading TIMERx_CH0CV. 0: No channel 0interrupt occurred 1: Channel 0 interrupt occurred
0	UPIF	Update interrupt flag This bit is set by hardware when an update event occurs and cleared by software. 0: No update interrupt occurred 1: Update interrupt occurred

Software event generation register (TIMERx_SWEVG)

Address offset: 0x14
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TRGG	Reserved.			CH1G	CH0G	UPG
									w				w	w	w

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	TRGG	Trigger event generation This bit is set by software and cleared by hardware automatically. When this bit is set, the TRGIF flag in TIMERx_STAT register will be set, related interrupt or DMA transfer can occur if enabled. 0: No generate a trigger event 1: Generate a trigger event
5:3	Reserved	Must be kept at reset value.
2	CH1G	Channel 1 capture or compare event generation Refer to CH0G description
1	CH0G	Channel 0 capture or compare event generation

		This bit is set by software to generate a capture or compare event in channel 0, it is automatically cleared by hardware. When this bit is set, the CH0IF flag will be set, and the corresponding interrupt or DMA request will be sent if enabled. In addition, if channel 0 is configured in input mode, the current value of the counter is captured to TIMERx_CH0CV register, and the CH0OF flag is set if the CH0IF flag has been set. 0: No generate a channel 0 capture or compare event 1: Generate a channel 0 capture or compare event
0	UPG	This bit can be set by software, and automatically cleared by hardware. When this bit is set, the counter is cleared. The prescaler counter is cleared at the same time. 0: No generate an update event 1: Generate an update event

Channel control register 0 (TIMERx_CHCTL0)

Address offset: 0x18
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	CH1COMCTL[2:0]			CH1COMSEN	CH1COMFEN	CH1MS[1:0]		Reserved	CH0COMCTL[2:0]			CH0COMSEN	CH0COMFEN	CH0MS[1:0]	
CH1CAPFLT[3:0]				CH1CAPPSC[1:0]				CH0CAPFLT[3:0]				CH0CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14:12	CH1COMCTL[2:0]	Channel 1 compare output control Refer to CH0COMCTL description
11	CH1COMSEN	Channel 1 output compare shadow enable Refer to CH0COMSEN description
10	CH1COMFEN	Channel 1 output compare fast enable Refer to CH0COMFEN description
9:8	CH1MS[1:0]	Channel 1 mode selection This bit-field specifies the direction of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH1EN bit in TIMERx_CHCTL2 register is reset). 00: Channel 1 is programmed as output mode

		01: Channel 1 is programmed as input mode, IS1 is connected to CI1FE1 10: Channel 1 is programmed as input mode, IS1 is connected to CI0FE1 11: Channel 1 is programmed as input mode, IS1 is connected to ITS. **Note:** When CH1MS[1:0]=11, it is necessary to select an internal trigger input through TRGS bits in TIMERx_SMCFG register.
7	Reserved	Must be kept at reset value.
6:4	CH0COMCTL[2:0]	Channel 0 compare output control This bit-field specifies the compare output mode of the the output prepare signal O0CPRE. In addition, the high level of O0CPRE is the active level, and CH0_O and CH0_ON channels polarity depends on CH0P and CH0NP bits. 000: Timing mode. The O0CPRE signal keeps stable, independent of the comparison between the register TIMERx_CH0CV and the counter TIMERx_CNT. 001: Set the channel output. O0CPRE signal is forced high when the counter is equals to the output compare register TIMERx_CH0CV. 010: Clear the channel output. O0CPRE signal is forced low when the counter is equals to the output compare register TIMERx_CH0CV. 011: Toggle on match. O0CPRE toggles when the counter is equals to the output compare register TIMERx_CH0CV. 100: Force low. O0CPRE is forced to low level. 101: Force high. O0CPRE is forced to high level. 110: PWM mode0. When counting up, O0CPRE is high when the counter is smaller than TIMERx_CH0CV, and low otherwise. When counting down, O0CPRE is low when the counter is larger than TIMERx_CH0CV, and high otherwise. 111: PWM mode1. When counting up, O0CPRE is low when the counter is smaller than TIMERx_CH0CV, and high otherwise. When counting down, O0CPRE is high when the counter is larger than TIMERx_CH0CV, and low otherwise. If configured in PWM mode, the O0CPRE level changes only when the output compare mode is adjusted from “Timing” mode to “PWM” mode or the comparison result changes.
3	CH0COMSEN	Channel 0 compare output shadow enable When this bit is set, the shadow register of TIMERx_CH0CV register, which updates at each update event, will be enabled. 0: Channel 0 output compare shadow disable 1: Channel 0 output compare shadow enable The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1)
2	CH0COMFEN	Channel 0 output compare fast enable When this bit is set, the effect of an event on the trigger in input on the capture/compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH0_O is set to the compare level independently from the result of the comparison.

		0: Channel 0 output quickly compare disable. 1: Channel 0 output quickly compare enable.
1:0	CH0MS[1:0]	Channel 0 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH0EN bit in TIMERx_CHCTL2 register is reset).). 00: Channel 0 is programmed as output mode 01: Channel 0 is programmed as input mode, IS0 is connected to CI0FE0 10: Channel 0 is programmed as input mode, IS0 is connected to CI1FE0 11: Channel 0 is programmed as input mode, IS0 is connected to ITS **Note:** When CH0MS[1:0]=11, it is necessary to select an internal trigger input through TRGS bits in TIMERx_SMCFG register.

Input capture mode:

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	CH1CAPFLT[3:0]	Channel 1 input capture filter control Refer to CH0CAPFLT description
11:10	CH1CAPPSC[1:0]	Channel 1 input capture prescaler Refer to CH0CAPPSC description
9:8	CH1MS[1:0]	Channel 1 mode selection Same as output compare mode
7:4	CH0CAPFLT[3:0]	Channel 0 input capture filter control The CI0 input signal can be filtered by digital filter and this bit-field configure the filtering capability. Basic principle of digital filter: continuously sample the CI0 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level. The filtering capability configuration is as follows:

CH0CAPFLT [3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{CK_TIMER}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	
4'b0110	6	$f_{DTS}/4$
4'b0111	8	
4'b1000	6	$f_{DTS}/8$
4'b1001	8	
4'b1010	5	$f_{DTS}/16$

		4’b1011	6	
		4’b1100	8	
		4’b1101	5	f_{DTS}/32
		4’b1110	6	
		4’b1111	8	

3:2	CH0CAPPSC[1:0]	Channel 0 input capture prescaler This bit-field specifies the factor of the prescaler on channel 0 input. The prescaler is reset when CH0EN bit in TIMERx_CHCTL2 register is clear. 00: Prescaler disable, input capture occurs on every channel input edge 01: The input capture occurs on every 2 channel input edges 10: The input capture occurs on every 4 channel input edges 11: The input capture occurs on every 8 channel input edges
1:0	CH0MS[1:0]	Channel 0 mode selection Same as output compare mode

Channel control register 2 (TIMERx_CHCTL2)

Address offset: 0x20

Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CH1NP	Reserved	CH1P	CH1EN	CH0NP	Reserved	CH0P	CH0EN
								rw		rw	rw	rw		rw	rw

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	CH1NP	Channel 1 complementary output polarity Refer to CH0NP description
6	Reserved	Must be kept at reset value.
5	CH1P	Channel 1 capture/compare function polarity Refer to CH0P description
4	CH1EN	Channel 1 capture/compare function enable Refer to CH1EN description
3	CH0NP	Channel 0 complementary output polarity When channel 0 is configured in output mode, this bit should be keep reset value. When channel 0 is configured in input mode, together with CH0P, this bit is used to

		define the polarity of CI0.
2	Reserved	Must be kept at reset value.
1	CH0P	Channel 0 capture/compare function polarity When channel 0 is configured in output mode, this bit specifies the output signal polarity. 0: Channel 0 high level is active level 1: Channel 0 low level is active level When channel 0 is configured in input mode, this bit specifies the CI0 signal polarity. [CH0NP, CH0P] will select the active trigger or capture polarity for CI0FE0 or CI1FE0. [CH0NP=0, CH0P=0]: The rising edge of CIxFE0 is the active signal for capture or trigger operation in slave mode. And CIxFE0 will not be inverted. [CH0NP=0, CH0P=1]: The falling edge of CIxFE0 is the active signal for capture or trigger operation in slave mode. And CIxFE0 will be inverted. [CH0NP=1, CH0P=0]: Reserved. [CH0NP=1, CH0P=1]: The falling and rising edges of CIxFE0 are both the active signal for capture or trigger operation in slave mode. And CIxFE0 will not be inverted.
0	CH0EN	Channel 0 capture/compare function enable When channel 0 is configured in output mode, setting this bit enables CH0_O signal in active state. When channel 0 is configured in input mode, setting this bit enables the capture event in channel0. 0: Channel 0 disabled 1: Channel 0 enabled

Counter register (TIMERx_CNT)

Address offset: 0x24
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CNT[15:0]	This bit-field indicates the current counter value. Writing to this bit-field can change

the value of the counter.

Prescaler register (TIMERx_PSC)

Address offset: 0x28
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC[15:0]	Prescaler value of the counter clock The TIMER_CK clock is divided by (PSC+1) to generate the counter clock. The value of this bit-filed will be loaded to the corresponding shadow register at every update event.

Counter auto reload register (TIMERx_CAR)

Address offset: 0x2C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CARL[15:0]	Counter auto reload value This bit-field specifies the auto reload value of the counter.

Channel 0 capture/compare value register (TIMERx_CH0CV)

Address offset: 0x34

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH0VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH0VAL[15:0]	Capture/compare value of channel0 When channel 0 is configured in input mode, this bit-field indicates the counter value at the last capture event.And this bit-field is read-only. When channel 0 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Channel 1 capture/compare value register (TIMERx_CH1CV)

Address offset: 0x38

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

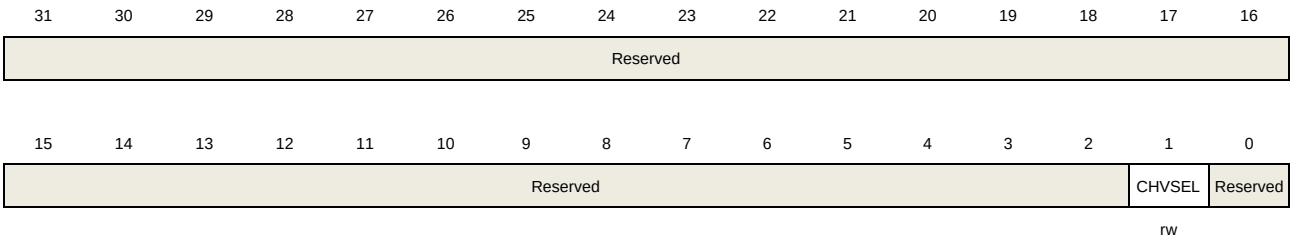
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH1VAL[15:0]	Capture/compare value of channel1 When channel 1 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 1 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Configuration register (TIMERx_CFG)

Address offset: 0xFC
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	CHVSEL	Write CHxVAL register selection This bit-field set and reset by software. 1: If write the CHxVAL register, the write value is same as the CHxVAL value, the write access ignored 0: No effect
0	Reserved	Must be kept at reset value.

16.4. General level2 timer (TIMERx, x=9, 10, 12, 13)

16.4.1. Overview

The general level2 timer module (TIMER9, 10, 12, 13) is a one-channel timer that supports input capture and output compare. They can generate PWM signals to control motor or be used for power management applications. The general level2 timer has a 16-bit counter that can be used as an unsigned counter.

In addition, the general level2 timers can be programmed and be used for counting, their external events can be used to drive other timers.

16.4.2. Characteristics

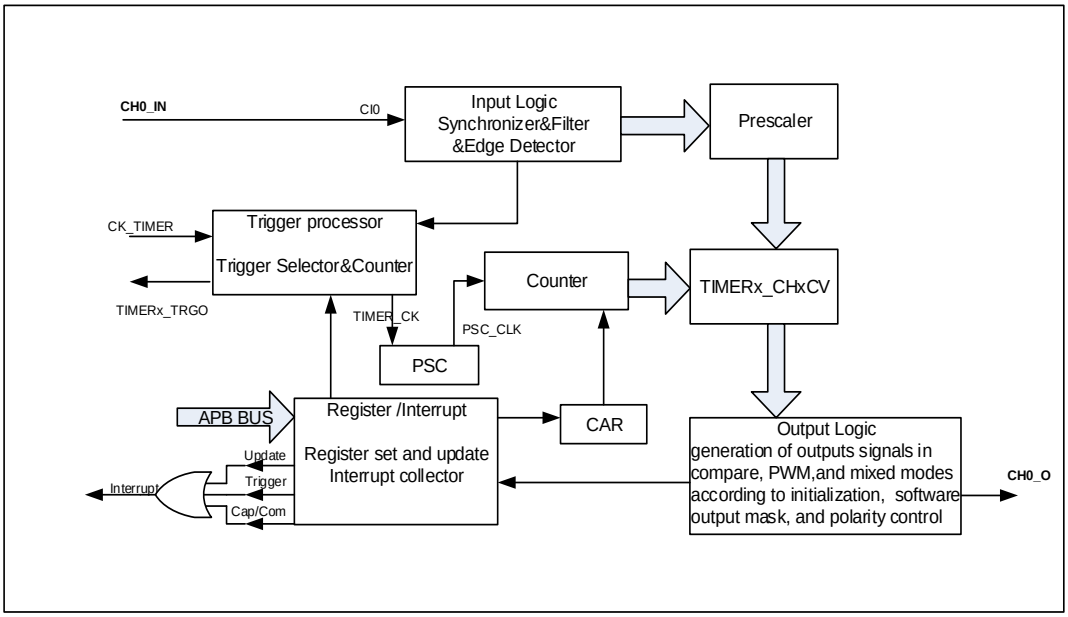
- Total channel num: 1.
- Counter width: 16 bits.
- Clock source of timer: internal clock.
- Counter mode: up counting only.
- Programmable prescaler: 16 bits. The factor can be changed ongoing.

- Each channel is user-configurable: input capture mode, output compare mode, programmable PWM mode.
- Auto reload function.
- Interrupt output: update event and compare/capture event.

16.4.3. Block diagram

Figure 16-57. General level2 timer block diagram provides details on the internal configuration of the general level2 timer.

Figure 16-57. General level2 timer block diagram



16.4.4. Function overview

Clock source configuration

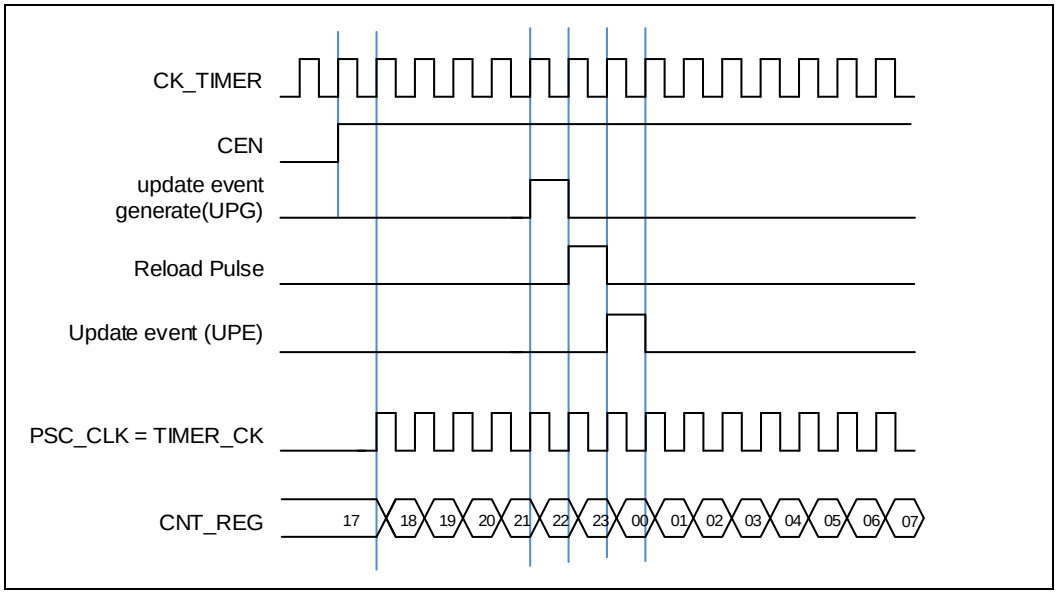
The general level2 TIMER can only being clocked by the CK_TIMER.

- Internal timer clock CK_TIMER which is from module RCU

The general level2 TIMER has only one clock source which is the internal CK_TIMER, used to drive the counter prescaler. When the CEN is set, the CK_TIMER will be divided by PSC value to generate PSC_CLK.

The TIMER_CK, driven counter's prescaler to count, is equal to CK_TIMER which is from RCU

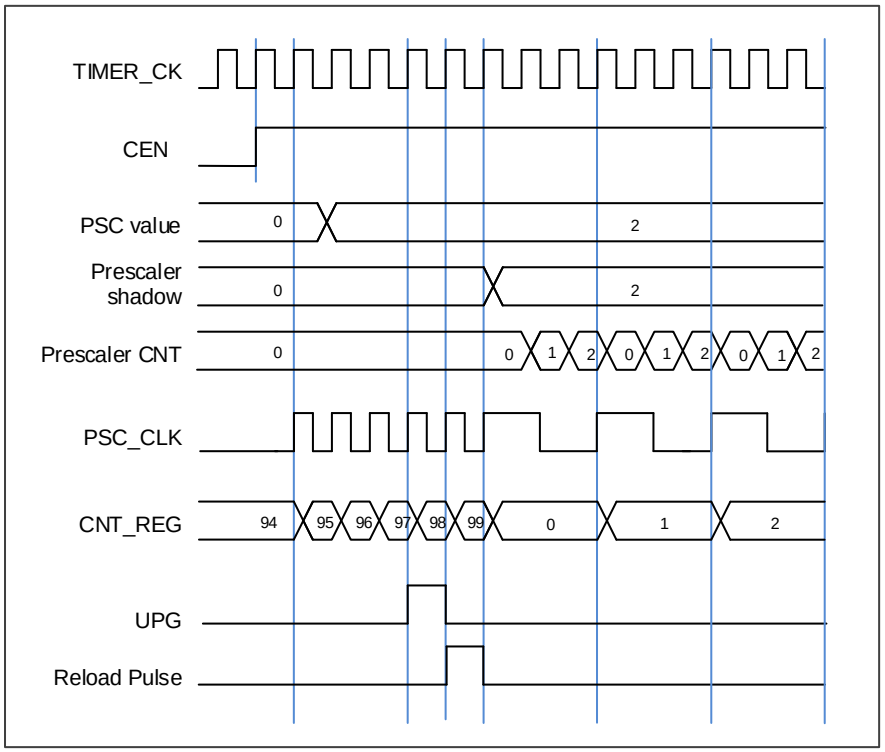
Figure 16-58. Timing chart of internal clock divided by 1



Clock prescaler

The counter clock (PSC_CK) is obtained by the TIMER_CK through the prescaler, and the prescale factor can be configured from 1 to 65536 through the prescaler register (TIMERx_PSC). The new written prescaler value will not take effect until the next update event.

Figure 16-59. Timing chart of PSC value change from 0 to 2



Counter up counting

In this mode, the counter counts up continuously from 0 to the counter-reload value, which is defined in the TIMERx_CAR register, in a count-up direction. Once the counter reaches the counter reload value, the counter will start counting up from 0 again. The update event is generated at each counter overflow. The counting direction bit DIR in the TIMERx_CTL1 register should be set to 0 for the up counting mode.

When the update event is set by the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to 0 and generates an update event.

If the UPDIS bit in TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (counter auto reload register, prescaler register) are updated.

The following figures show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 16-60. Timing chart of up counting mode, PSC=0/2

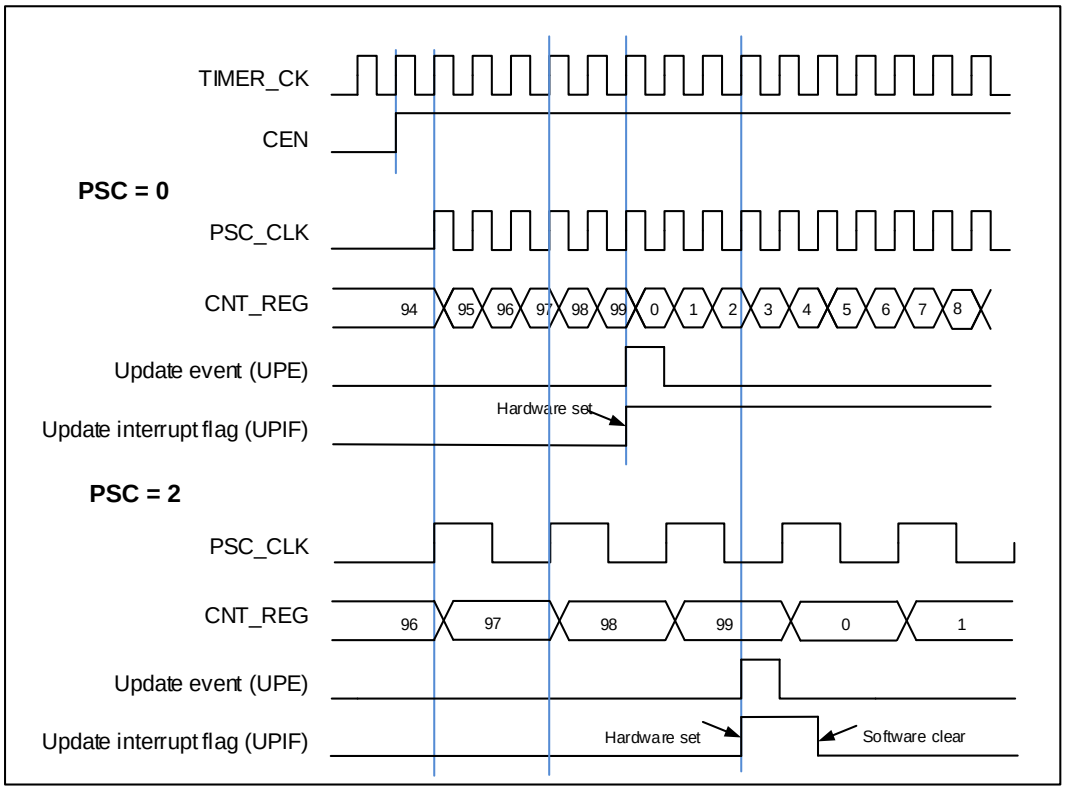
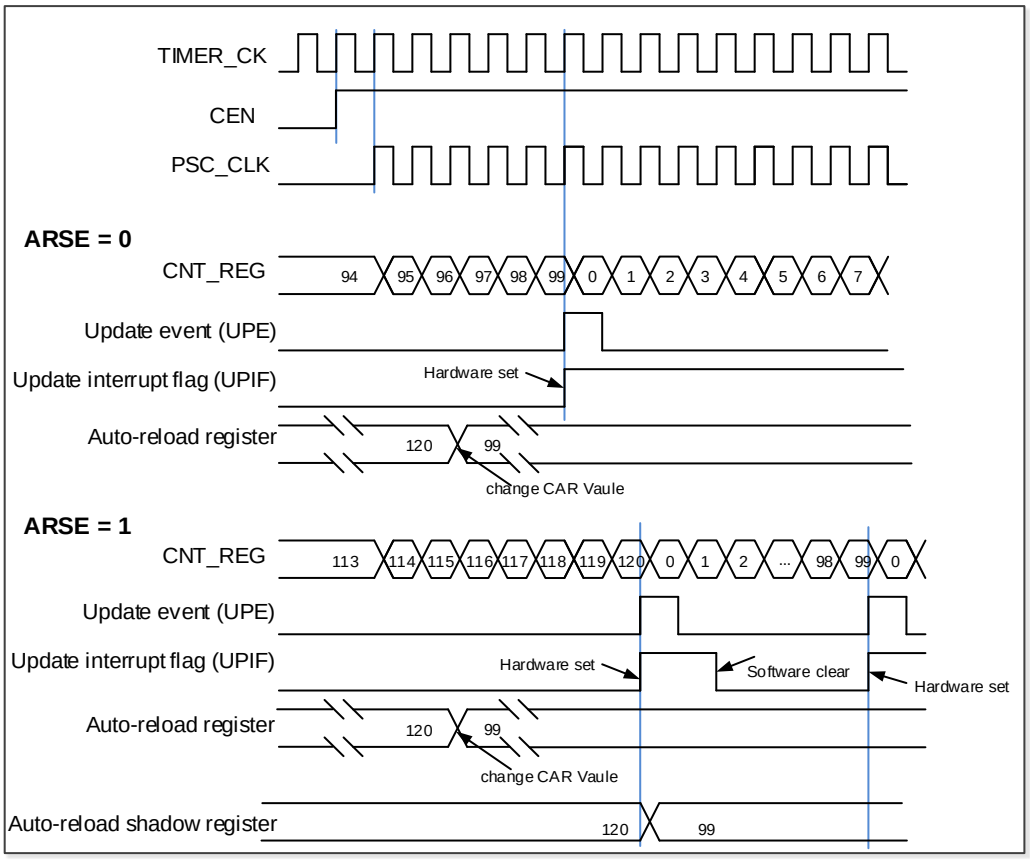


Figure 16-61. Timing chart of up counting mode, change TIMERx_CAR on the go



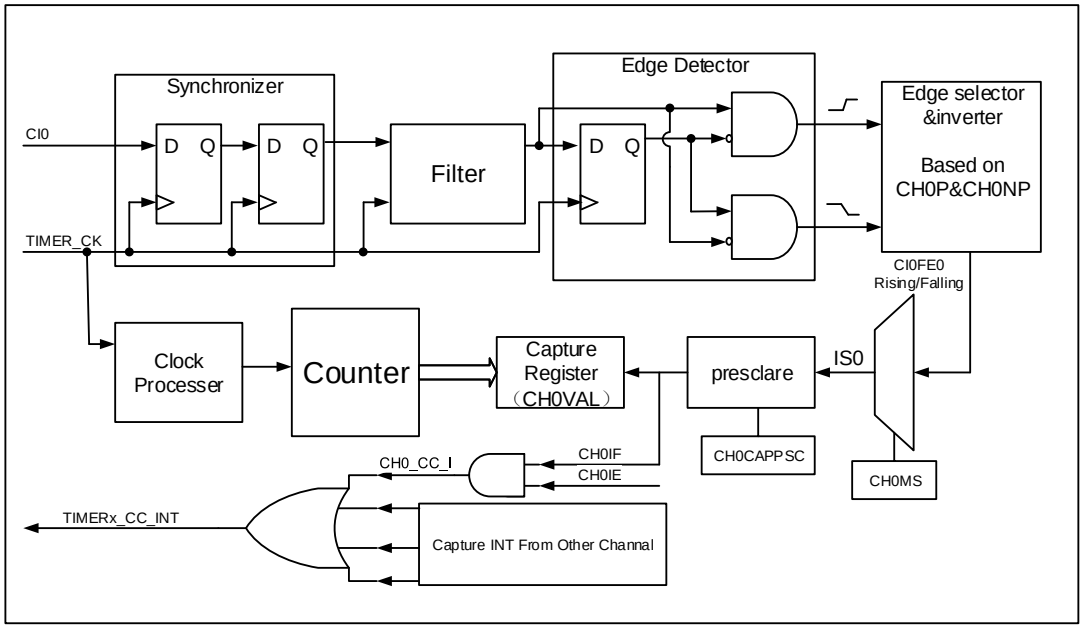
Input capture and output compare channels

The general level2 timer has one independent channel which can be used as capture inputs or compare match outputs. Each channel is built around a channel capture compare register including an input stage, channel controller and an output stage.

- **Channel input capture function**

Channel input capture function allows the channel to perform measurements such as pulse timing, frequency, period, duty cycle and so on. The input stage consists of a digital filter, a channel polarity selection, edge detection and a channel prescaler. When a selected edge occurs on the channel input, the current value of the counter is captured into the TIMERx_CHxCV register, at the same time the CHxIF bit is set and the channel interrupt is generated if enabled by CHxIE = 1.

Figure 16-62. Channel input capture principle



First, the channel input signal (CIx) is synchronized to TIMER_CK domain, and then sampled by a digital filter to generate a filtered input signal. Then through the edge detector, the rising and fall edge are detected. You can select one of them by CHxP. One more selector is for the other channel and trig, controlled by CHxMS. The IC_prescaler make several the input event generate one effective capture event. On the capture event, TIMERx_CHxCV will restore the value of counter.

So the process can be divided to several steps as below:

Step1: Filter configuration. (CHxCAPFLT in TIMERx_CHCTL0)

Based on the input signal and requested signal quality, configure compatible CHxCAPFLT.

Step2: Edge selection. (CHxP/CHxNP in TIMERx_CHCTL2)

Rising or falling edge, choose one by CHxP/CHxNP.

Step3: Capture source selection. (CHxMS in TIMERx_CHCTL0)

As soon as you select one input capture source by CHxMS, you have set the channel to input mode (CHxMS!=0x0) and TIMERx_CHxCV cannot be written any more.

Step4: Interrupt enable. (CHxIE in TIMERx_DMAINTEN)

Enable the related interrupt enable; you can got the interrupt.

Step5: Capture enables. (CHxEN in TIMERx_CHCTL2)

Result: When you wanted input signal is got, TIMERx_CHxCV will be set by Counter's value. And CHxIF is asserted. If the CHxIF is high, the CHxOF will be asserted also. The interrupt will be asserted based on the your configuration of CHxIE in TIMERx_DMAINTEN

Direct generation: If you want to generate a DMA request or Interrupt, you can set CHxG by software directly.

The channel input capture function can be also used for pulse width measurement from signals on the TIMERx_CHx pins. For example, PWM signal connect to CI0 input. Select channel 0 capture signals to CI0 by setting CH0MS to 2'b01 in the channel control register (TIMERx_CHCTL0) and set capture on rising edge. The counter set to restart mode and restart on channel 0 rising edge. Then the TIMERX_CH0CV can measure the PWM period and the TIMERx_CH1CV can measure the PWM duty.

- **Channel output compare function**

In channel Compare function, the TIMERx can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CHxVAL register of an output compare channel, the channel (n) output can be set, cleared, or toggled based on CHxCOMCTL. when the counter reaches the value in the CHxVAL register, the CHxIF bit is set and the channel (n) interrupt is generated if CHxIE = 1.

So the process can be divided to several steps as below:

Step1: Clock configuration. Such as clock source, clock prescaler and so on.

Step2: Compare mode configuration.

* Set the shadow enable mode by CHxCOMSEN
* Set the output mode (Set/Clear/Toggle) by CHxCOMCTL.
* Select the active high polarity by CHxP/CHxNP
* Enable the output by CHxEN

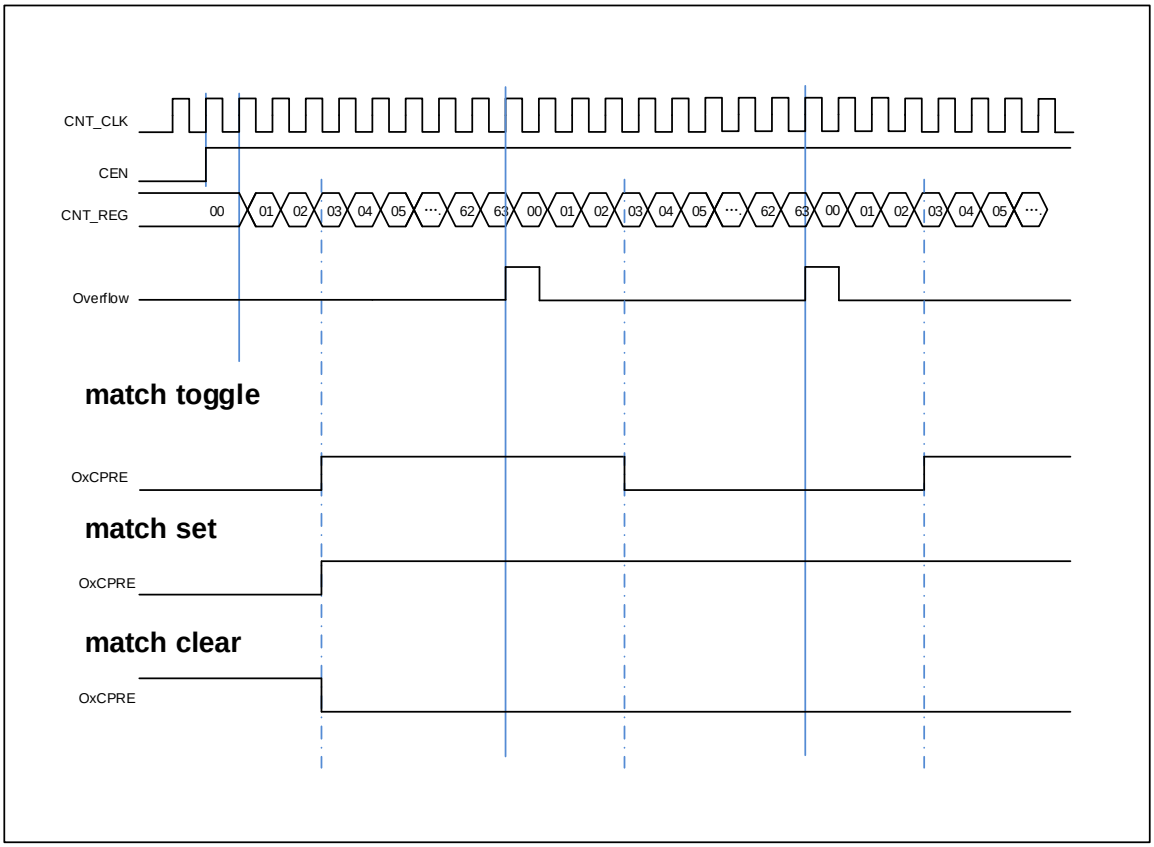
Step3: Interrupt/DMA-request enables configuration by CHxIE

Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV.
About the CHxVAL, you can change it on the go to meet the waveform you expected.

Step5: Start the counter by CEN.

The timechart below show the three compare modes toggle/set/clear. CAR=0x63, CHxVAL=0x3

Figure 16-63. Output-compare under three modes



Channel output prepare signal

When the TIMERx is used in the compare match output mode, the OxCPRE signal (Channel x Output prepare signal) is defined by setting the CHxCOMCTL filed. The OxCPRE signal has several types of output function. These include, keeping the original level by setting the CHxCOMCTL field to 0x00, set to 1 by setting the CHxCOMCTL field to 0x01, set to 0 by setting the CHxCOMCTL field to 0x02 or signal toggle by setting the CHxCOMCTL field to 0x03 when the counter value matches the content of the TIMERx_CHxCV register.

The PWM mode 0 and PWM mode 1 outputs are also another kind of OxCPRE output which is setup by setting the CHxCOMCTL field to 0x06/0x07. In these modes, the OxCPRE signal level is changed according to the counting direction and the relationship between the counter value and the TIMERx_CHxCV content. With regard to a more detail description refer to the relative bit definition.

Another special function of the OxCPRE signal is a forced output which can be achieved by setting the CHxCOMCTL field to 0x04/0x05. Here the output can be forced to an inactive/active level irrespective of the comparison condition between the counter and the TIMERx_CHxCV values.

Timers interconnection

Refer to ***Advanced timer (TIMERx, x=0, 7)***.

Timer debug mode

When the Cortex®-M4 halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register set to 1, the TIMERx counter stops.

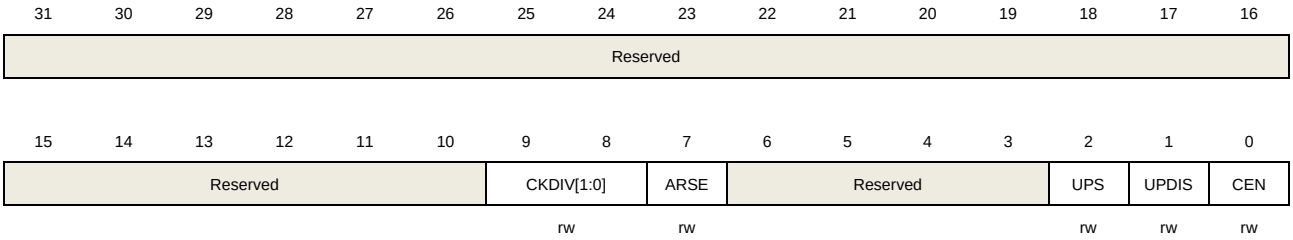
16.4.5. TIMERx registers (x=9, 10, 12, 13)

TIMER9 base address: 0x4001 5000
TIMER10 base address: 0x4001 5400
TIMER12 base address: 0x4000 1C00
TIMER13 base address: 0x4000 2000

Control register 0 (TIMERx_CTL0)

Address offset: 0x00
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	CKDIV[1:0]	Clock division The CKDIV bits can be configured by software to specify division factor between the CK_TIMER and the dead-time and digital filter sample clock (DTS). 00: $f_{DTS}=f_{CK_TIMER}$ 01: $f_{DTS}= f_{CK_TIMER}$ /2 10: $f_{DTS}= f_{CK_TIMER}$ /4 11: Reserved
7	ARSE	Auto-reload shadow enable 0: The shadow register for TIMERx_CAR register is disabled 1: The shadow register for TIMERx_CAR register is enabled
6:3	Reserved	Must be kept at reset value.
2	UPS	Update source This bit is used to select the update event sources by software. 0: These events generate update interrupts or DMA requests: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: This event generates update interrupts or DMA requests: The counter generates an overflow or underflow event

1	UPDIS	Update disable. This bit is used to enable or disable the update event generation. 0: Update event enable. When an update event occurs, the corresponding shadow registers are loaded with their preloaded values. These events generate update event: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: Update event disable. **Note:** When this bit is set to 1, setting UPG bit or the restart mode does not generate an update event, but the counter and prescaler are initialized.
0	CEN	Counter enable 0: Counter disable 1: Counter enable The CEN bit must be set by software when timer works in external clock mode, pause mode or encoder mode.

Interrupt enable register (TIMERx_DMAINTEN)

Address offset: 0x0C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														CH0IE	UPIE
														rw	rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	CH0IE	Channel 0 capture/compare interrupt enable 0: Disabled 1: Enabled
0	UPIE	Update interrupt enable 0: Disabled 1: Enabled

Interrupt flag register (TIMERx_INTF)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CH0OF	Reserved.							CH0IF	UPIF
						rc_w0								rc_w0	rc_w0

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	CH0OF	Channel 0 over capture flag When channel 0 is configured in input mode, this flag is set by hardware when a capture event occurs while CH0IF flag has already been set. This flag is cleared by software. 0: No over capture interrupt occurred 1: Over capture interrupt occurred
8:2	Reserved	Must be kept at reset value.
1	CH0IF	Channel 0 capture/compare interrupt flag This flag is set by hardware and cleared by software. If channel 0 is in input mode, this flag is set when a capture event occurs. If channel 0 is in output mode, this flag is set when a compare event occurs. If channel 0 is set to input mode, this bit will be reset by reading TIMERx_CH0CV. 0: No channel 1 interrupt occurred 1: Channel 1 interrupt occurred
0	UPIF	Update interrupt flag This bit is set by hardware when an update event occurs and cleared by software. 0: No update interrupt occurred 1: Update interrupt occurred

Software event generation register (TIMERx_SWEVG)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														CH0G	UPG

w w

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	CH0G	Channel 0 capture or compare event generation This bit is set by software to generate a capture or compare event in channel 0, it is automatically cleared by hardware. When this bit is set, the CH0IF flag will be set, and the corresponding interrupt or DMA request will be sent if enabled. In addition, if channel 0 is configured in input mode, the current value of the counter is captured to TIMERx_CH0CV register, and the CH0OF flag is set if the CH0IF flag has been set. 0: No generate a channel 0 capture or compare event 1: Generate a channel 0 capture or compare event
0	UPG	This bit can be set by software, and automatically cleared by hardware. When this bit is set, the counter is cleared. The prescaler counter is cleared at the same time. 0: No generate an update event 1: Generate an update event

Channel control register 0 (TIMERx_CHCTL0)

Address offset: 0x18
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved.								Reserved	CH0COMCTL[2:0]			CH0COMSEN	CH0COMFEN	CH0MS[1:0]	
								CH0CAPFLT[3:0]				CH0CAPPSC[1:0]			
									rw			rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6:4	CH0COMCTL[2:0]	Channel 0 compare output control This bit-field specifies the compare output mode of the the output prepare signal O0CPRE. In addition, the high level of O0CPRE is the active level, and CH0_O and CH0_ON channels polarity depends on CH0P and CH0NP bits. 000: Timing mode. The O0CPRE signal keeps stable, independent of the comparison between the register TIMERx_CH0CV and the counter TIMERx_CNT. 001: Set the channel output. O0CPRE signal is forced high when the counter is

		equals to the output compare register TIMERx_CH0CV. 010: Clear the channel output. O0CPRE signal is forced low when the counter is equals to the output compare register TIMERx_CH0CV. 011: Toggle on match. O0CPRE toggles when the counter is equals to the output compare register TIMERx_CH0CV. 100: Force low. O0CPRE is forced to low level. 101: Force high. O0CPRE is forced to high level. 110: PWM mode0. When counting up, O0CPRE is high when the counter is smaller than TIMERx_CH0CV, and low otherwise. When counting down, O0CPRE is low when the counter is larger than TIMERx_CH0CV, and high otherwise. 111: PWM mode1. When counting up, O0CPRE is low when the counter is smaller than TIMERx_CH0CV, and high otherwise. When counting down, O0CPRE is high when the counter is larger than TIMERx_CH0CV, and low otherwise. If configured in PWM mode, the O0CPRE level changes only when the output compare mode is adjusted from “Timing” mode to “PWM” mode or the comparison result changes.
3	CH0COMSEN	Channel 0 compare output shadow enable When this bit is set, the shadow register of TIMERx_CH0CV register, which updates at each update event, will be enabled. 0: Channel 0 output compare shadow disable 1: Channel 0 output compare shadow enable The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1)
2	CH0COMFEN	Channel 0 output compare fast enable When this bit is set, the effect of an event on the trigger in input on the capture/compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH0_O is set to the compare level independently from the result of the comparison. 0: Channel 0 output quickly compare disable. 1: Channel 0 output quickly compare enable.
1:0	CH0MS[1:0]	Channel 0 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH0EN bit in TIMERx_CHCTL2 register is reset).). 00: Channel 0 is programmed as output mode 01: Channel 0 is programmed as input mode, IS0 is connected to CI0FE0 Others: Reserved

Input capture mode:

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:4	CH0CAPFLT[3:0]	Channel 0 input capture filter control

The CI0 input signal can be filtered by digital filter and this bit-field configure the filtering capability.

Basic principle of digital filter: continuously sample the CI0 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level.

The filtering capability configuration is as follows:

CH0CAPFLT [3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{CK_TIMER}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	
4'b0110	6	$f_{DTS}/4$
4'b0111	8	
4'b1000	6	$f_{DTS}/8$
4'b1001	8	
4'b1010	5	$f_{DTS}/16$
4'b1011	6	
4'b1100	8	
4'b1101	5	$f_{DTS}/32$
4'b1110	6	
4'b1111	8	

3:2 CH0CAPPSC[1:0] Channel 0 input capture prescaler

This bit-field specifies the factor of the prescaler on channel 0 input. The prescaler is reset when CH0EN bit in TIMERx_CHCTL2 register is clear.

00: Prescaler disable, input capture occurs on every channel input edge

01: The input capture occurs on every 2 channel input edges

10: The input capture occurs on every 4 channel input edges

11: The input capture occurs on every 8 channel input edges

1:0 CH0MS[1:0] Channel 0 mode selection

Same as output compare mode

Channel control register 2 (TIMERx_CHCTL2)

Address offset: 0x20

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word(32-bit).

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

Reserved	CH0NP	Reserved	CH0P	CH0EN
	rw		rw	rw

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	CH0NP	Channel 0 complementary output polarity When channel 0 is configured in output mode, this bit specifies the complementary output signal polarity. 0: Channel 0 complementary output high level is active level 1: Channel 0 complementary output low level is active level When channel 0 is configured in input mode, in conjunction with CH0P, this bit is used to define the polarity of CI0.
2	Reserved	Must be kept at reset value.
1	CH0P	Channel 0 capture/compare function polarity When channel 0 is configured in output mode, this bit specifies the output signal polarity. 0: Channel 0 high level is active level 1: Channel 0 low level is active level When channel 0 is configured in input mode, this bit specifies the CI0 signal polarity. [CH0NP, CH0P] will select the active trigger or capture polarity for CI0FE0 or CI1FE0. [CH0NP=0, CH0P=0]: The rising edge of CIxFE0 is the active signal for capture or trigger operation in slave mode. And CIxFE0 will not be inverted. [CH0NP=0, CH0P=1]: The falling edge of CIxFE0 is the active signal for capture or trigger operation in slave mode. And CIxFE0 will be inverted. [CH0NP=1, CH0P=0]: Reserved. [CH0NP=1, CH0P=1]: The falling and rising edges of CIxFE0 are both the active signal for capture or trigger operation in slave mode. And CIxFE0 will not be inverted.
0	CH0EN	Channel 0 capture/compare function enable When channel 0 is configured in input mode, setting this bit enables CH0_O signal in active state. When channel 0 is configured in output mode, setting this bit enables the capture event in channel0. 0: Channel 0 disabled 1: Channel 0 enabled

Counter register (TIMERx_CNT)

Address offset: 0x24
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CNT[15:0]	This bit-field indicates the current counter value. Writing to this bit-field can change the value of the counter.

Prescaler register (TIMERx_PSC)

Address offset: 0x28
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC[15:0]	Prescaler value of the counter clock The TIMER_CK clock is divided by (PSC+1) to generate the counter clock. The value of this bit-filed will be loaded to the corresponding shadow register at every update event.

Counter auto reload register (TIMERx_CAR)

Address offset: 0x2C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CARL[15:0]	Counter auto reload value This bit-field specifies the auto reload value of the counter.

Channel 0 capture/compare value register (TIMERx_CH0CV)

Address offset: 0x34
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH0VAL[15:0]															

rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH0VAL[15:0]	Capture/compare value of channel0 When channel 0 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 0 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Configuration register (TIMERx_CFG)

Address offset: 0xFC
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	CHVSEL	Write CHxVAL register selection This bit-field set and reset by software. 1: If write the CHxVAL register, the write value is same as the CHxVAL value, the write access ignored 0: No effect
0	Reserved	Must be kept at reset value.

16.5. Basic timer (TIMERx, x=5, 6)

16.5.1. Overview

The basic timer module (TIMER5, TIMER 6) has a 16-bit counter that can be used as an unsigned counter. The basic timer can be configured to generate a DMA request and a TRGO to connect to DAC.

16.5.2. Characteristics

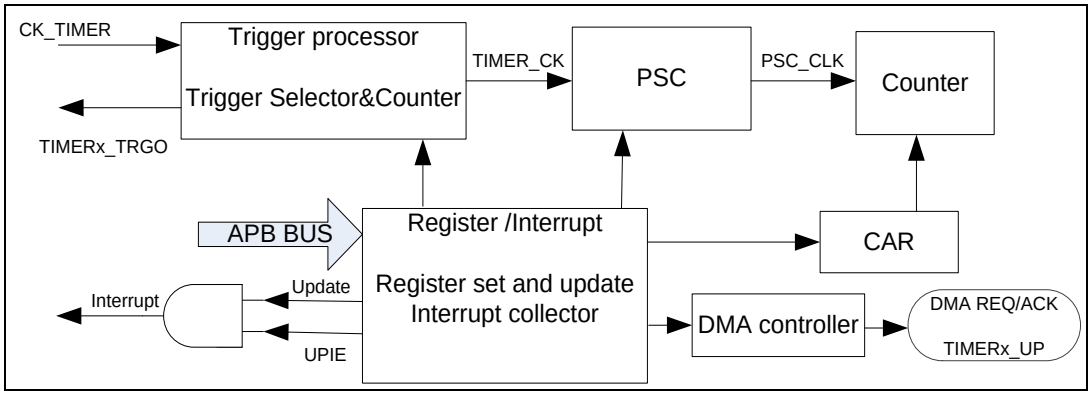
- Counter width: 16 bits.
- Source of count clock is internal clock only.
- Multiple counter modes: count up.
- Programmable prescaler: 16 bits. The factor can be changed ongoing.
- Auto reload function.
- Interrupt output or DMA request: update event.

16.5.3. Block diagram

Figure 16-64. Basic timer block diagram provides details on the internal configuration of

the basic timer.

Figure 16-64. Basic timer block diagram



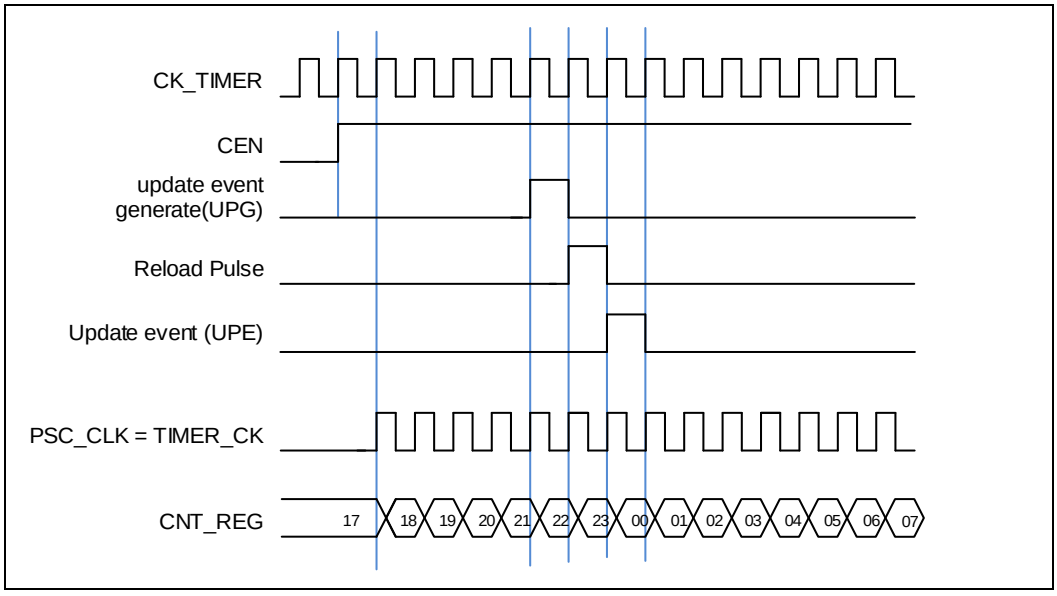
16.5.4. Function overview

Clock source configuration

The basic TIMER can only be clocked by the internal timer clock CK_TIMER, which is from the source named CK_TIMER in RCU

The TIMER_CK, driven counter's prescaler to count, is equal to CK_TIMER used to drive the counter prescaler. When the CEN is set, the CK_TIMER will be divided by PSC value to generate PSC_CLK.

Figure 16-65. Timing chart of internal clock divided by 1

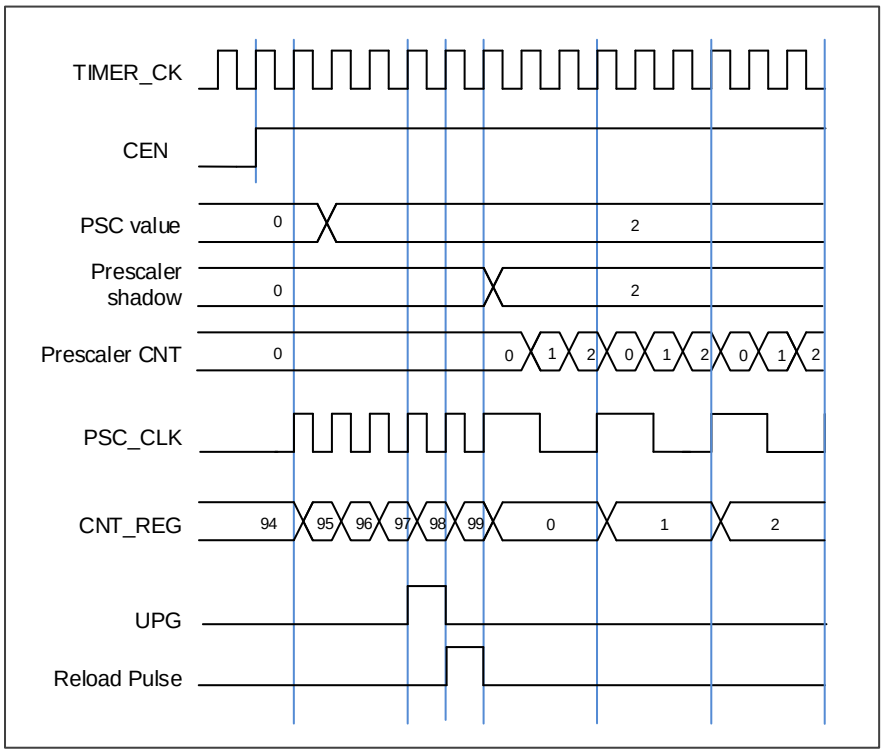


Clock prescaler

The counter clock (PSC_CK) is obtained by the TIMER_CK through the prescaler, and the prescale factor can be configured from 1 to 65536 through the prescaler register

(TIMERx_PSC). The new written prescaler value will not take effect until the next update event.

Figure 16-66. Timing chart of PSC value change from 0 to 2



Counter up counting

In this mode, the counter counts up continuously from 0 to the counter-reload value, which is defined in the TIMERx_CAR register, in a count-up direction. Once the counter reaches the counter reload value, the counter will start counting up from 0 again. The update event is generated at each counter overflow.

When the update event is set by the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to 0 and generates an update event.

If set the UPDIS bit in TIMERx_CTL0 register, the update event is disabled.

When an update event occurs, all the shadow registers (counter auto reload register, prescaler register) are updated.

The following figures show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 16-67. Timing chart of up counting mode, PSC=0/2

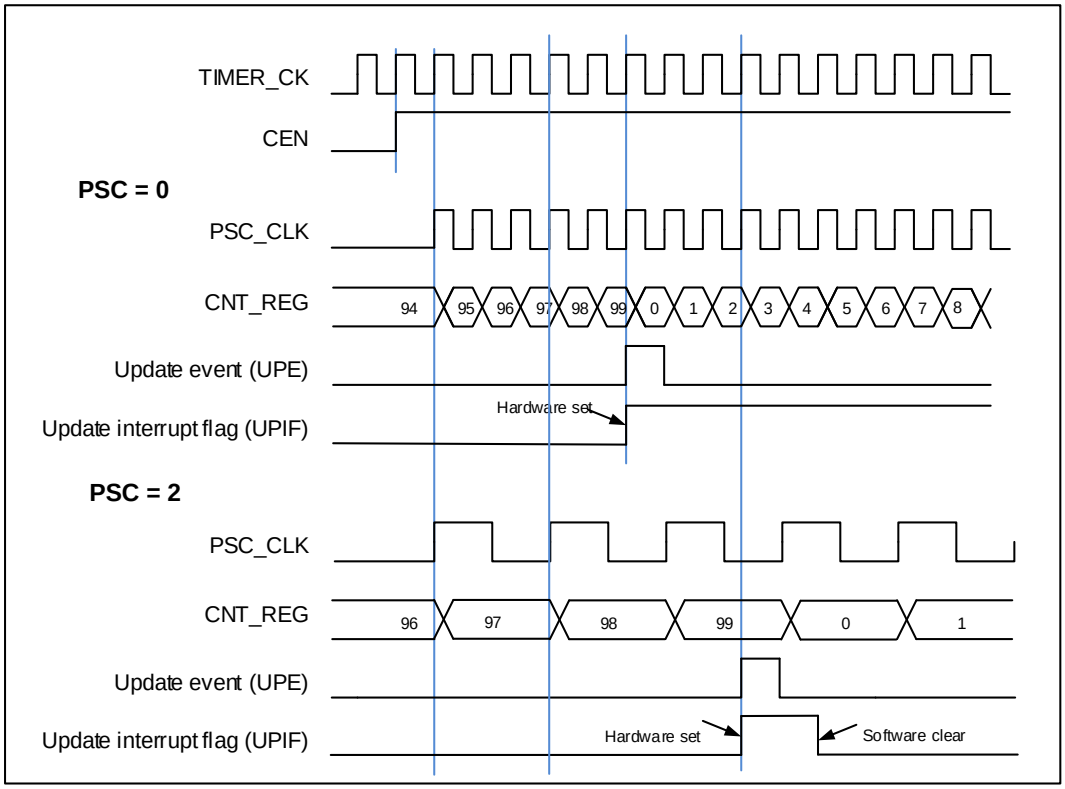
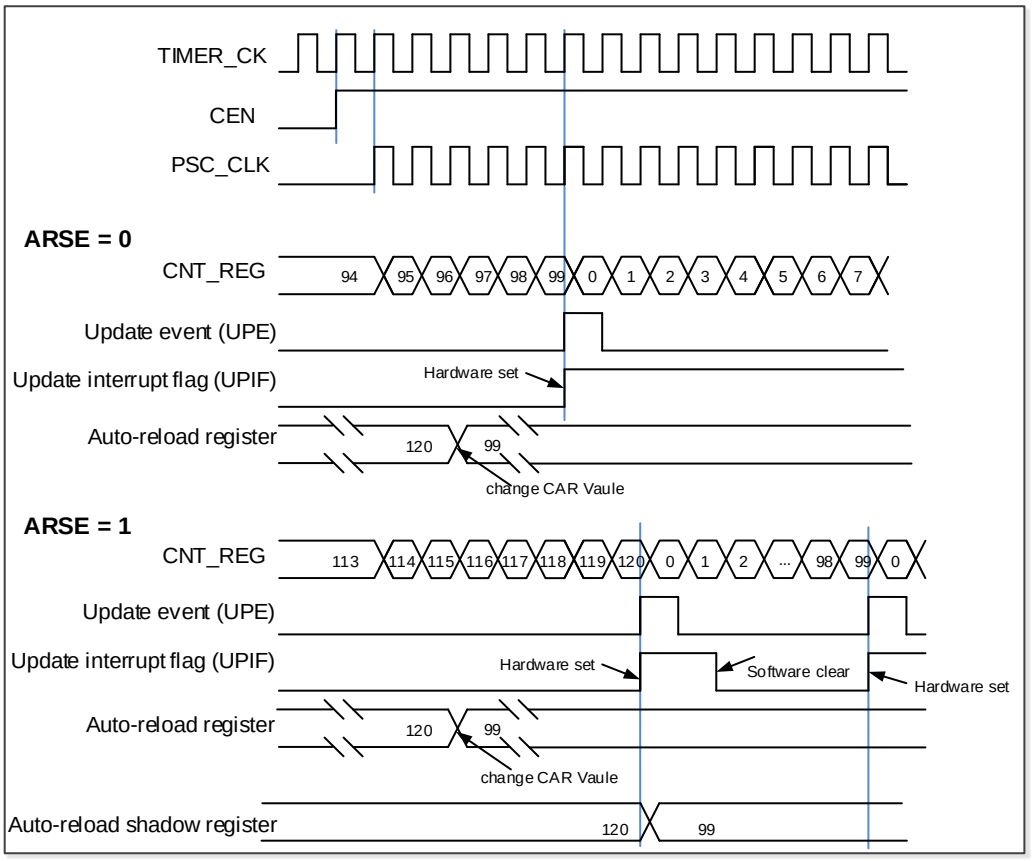


Figure 16-68. Timing chart of up counting mode, change TIMERx_CAR ongoing



Single pulse mode

Single pulse mode is opposite to the repetitive mode, which can be enabled by setting SPM in TIMERx_CTL0. When you set SPM, the counter will be clear and stop when the next update event.

Once the timer is set to operate in the single pulse mode, it is necessary to set the timer enable bit CEN in the TIMERx_CTL0 register to 1 to enable the counter, then the CEN bit keeps at a high state until the update event occurs or the CEN bit is written to 0 by software. If the CEN bit is cleared to 0 using software, the counter will be stopped and its value held.

Timer debug mode

When the Cortex®-M4 halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register set to 1, the TIMERx counter stops.

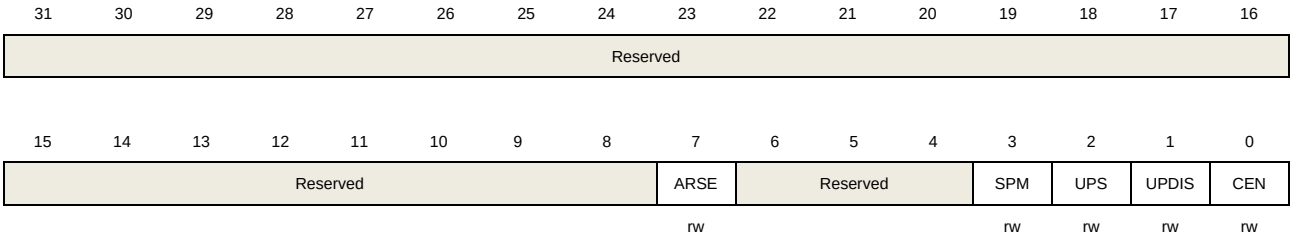
16.5.5. TIMERx registers (x=5, 6)

TIMER5 base address: 0x4000 1000
TIMER6 base address: 0x4000 1400

Control register 0 (TIMERx_CTL0)

Address offset: 0x00
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	ARSE	Auto-reload shadow enable 0: The shadow register for TIMERx_CAR register is disabled 1: The shadow register for TIMERx_CAR register is enabled
6:4	Reserved	Must be kept at reset value.
3	SPM	Single pulse mode. 0: Single pulse mode disable. The counter continues after update event. 1: Single pulse mode enable. The counter counts until the next update event occurs.
2	UPS	Update source This bit is used to select the update event sources by software. 0: These events generate update interrupts or DMA requests: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: This event generates update interrupts or DMA requests: The counter generates an overflow or underflow event
1	UPDIS	Update disable. This bit is used to enable or disable the update event generation. 0: Update event enable. When an update event occurs, the corresponding shadow registers are loaded with their preloaded values. These events generate update event: The UPG bit is set

The counter generates an overflow or underflow event

The restart mode generates an update event.

1: Update event disable.

Note: When this bit is set to 1, setting UPG bit or the restart mode does not generate an update event, but the counter and prescaler are initialized.

0 CEN Counter enable

0: Counter disable

1: Counter enable

The CEN bit must be set by software when timer works in external clock mode, pause mode or encoder mode.

Control register 1 (TIMERx_CTL1)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									MMC[2:0]			Reserved			
									rw						

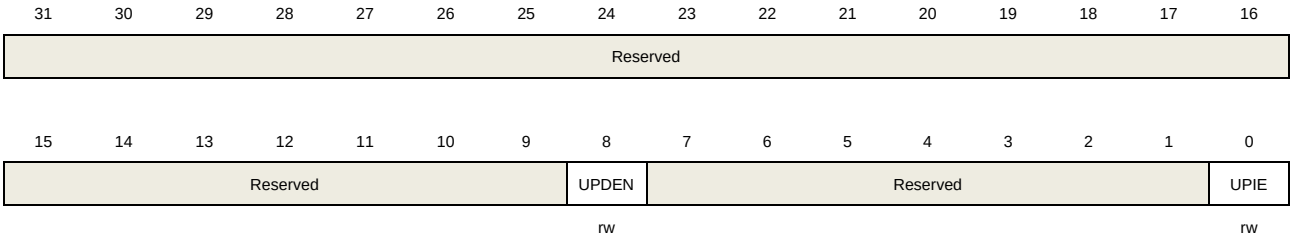
Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6:4	MMC[2:0]	Master mode control These bits control the selection of TRGO signal, which is sent by master timer to slave timer for synchronization function. 000: When a counter reset event occurs, a TGRO trigger signal is output. The counter resert source: Master timer generate a reset the UPG bit in the TIMERx_SWEVG register is set 001: Enable. When a conter start event occurs, a TGRO trigger signal is output. The counter start source : CEN control bit is set The trigger input in pause mode is high 010: When an update event occurs, a TGRO trigger signal is output. The update source depends on UPDIS bit and UPS bit.
3:0	Reserved	Must be kept at reset value.

Interrupt enable register (TIMERx_DMAINTEN)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



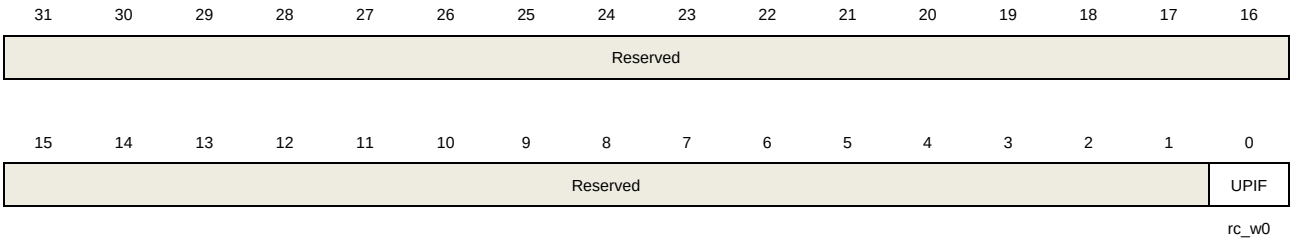
Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	UPDEN	Update DMA request enable 0: Disabled 1: Enabled
7:1	Reserved	Must be kept at reset value.
0	UPIE	Update interrupt enable 0: Disabled 1: Enabled

Interrupt flag register (TIMERx_INTF)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	UPIF	Update interrupt flag This bit is set by hardware when an update event occurs and cleared by software. 0: No update interrupt occurred

1: Update interrupt occurred

Software event generation register (TIMERx_SWEVG)

Address offset: 0x14
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	UPG	This bit can be set by software, and automatically cleared by hardware. When this bit is set, the counter is cleared. The prescaler counter is cleared at the same time. 0: No generate an update event 1: Generate an update event

Counter register (TIMERx_CNT)

Address offset: 0x24
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CNT[15:0]	This bit-field indicates the current counter value. Writing to this bit-field can change the value of the counter.

Prescaler register (TIMERx_PSC)

Address offset: 0x28

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC[15:0]	Prescaler value of the counter clock The TIMER_CK clock is divided by (PSC+1) to generate the counter clock. The value of this bit-filed will be loaded to the corresponding shadow register at every update event.

Counter auto reload register (TIMERx_CAR)

Address offset: 0x2C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CARL[15:0]	Counter auto reload value This bit-field specifies the auto reload value of the counter.

17. Universal synchronous/asynchronous receiver /transmitter (USART)

17.1. Overview

The Universal Synchronous/Asynchronous Receiver/Transmitter (USART) provides a flexible serial data exchange interface. Data frames can be transferred in full duplex or half duplex mode, synchronously or asynchronously through this interface. A programmable baud rate generator divides the UCLK (PCLK1 or PCLK2) to produce a dedicated baud rate clock for the USART transmitter and receiver.

Besides the standard asynchronous receiver and transmitter mode, the USART implements several other types of serial data exchange modes, such as IrDA (infrared data association) SIR mode, smartcard mode, LIN (local interconnection network) mode and half-duplex synchronous mode. It also supports multiprocessor communication mode, and hardware flow control protocol (CTS/RTS). The data frame can be transferred from LSB or MSB bit. The polarity of the TX/RX pins can be configured independently and flexibly.

The USART supports DMA function for high-speed data communication, except UART4.

17.2. Characteristics

- NRZ standard format.
- Asynchronous, full duplex communication.
- Programmable baud-rate generator.
  - Divided from the peripheral clocks, PCLK2 for USART0, PCLK1 for USART1/2 and UART3/4.
  - Oversampling by 16.
  - Maximum speed up to 7.5 MBits/s (PCLK2 120M and oversampling by 16).
- Fully programmable serial interface characteristics:
  - Even, odd or no-parity bit generation/detection.
  - A data word length can be 8 or 9 bits.
  - 0.5, 1, 1.5 or 2 stop bit generation.
- Transmitter and receiver can be enabled separately.
- Hardware flow control protocol (CTS/RTS).
- DMA request for data buffer access.
- LIN break generation and detection.
- IrDA support.
- Synchronous mode and transmitter clock output for synchronous transmission.
- ISO 7816-3 compliant smartcard interface.
  - Character mode (T=0).
  - Block mode (T=1).

  - Direct and inverse convention.
- Multiprocessor communication.
  - Enter into mute mode if address match does not occur.
  - Wake up from mute mode by idle frame or address match detection.
- Various status flags:
  - Flags for transfer detection: receive buffer not empty (RBNE), transmit buffer empty (TBE), transfer complete (TC), and busy (BSY).
  - Flags for error detection: overrun error (ORERR), noise error (NERR), frame error (FERR) and parity error (PERR).
  - Flag for hardware flow control: CTS changes (CTSF).
  - Flag for LIN mode: LIN break detected (LBDF).
  - Flag for multiprocessor communication: IDLE frame detected (IDLEF).
  - Flags for smartcard block mode: end of block (EBF) and receiver timeout (RTF).
  - Interrupt occurs at these events when the corresponding interrupt enable bits are set.

While USART0/1/2 is fully implemented, UART3/4 is only partially implemented with the following features not supported.

- Smartcard mode.
- Synchronous mode.
- Hardware flow control protocol (CTS/RTS).
- Configurable data polarity.

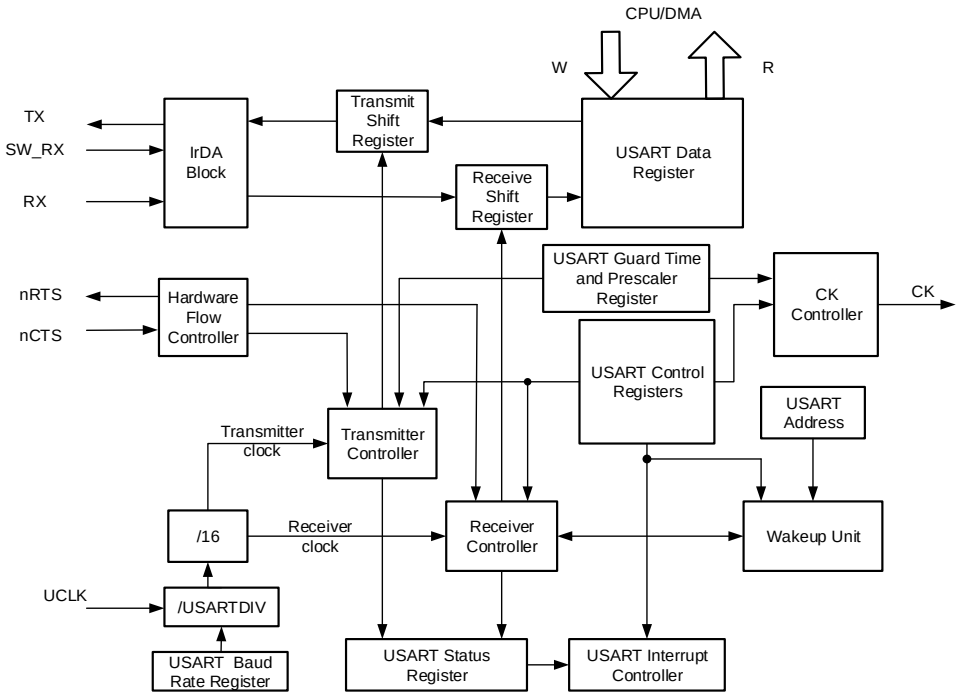
17.3. Function overview

The interface is externally connected to another device by the main pins listed in ***Table 17-1. Description of USART important pins***.

Table 17-1. Description of USART important pins

Pin	Type	Description
RX	Input	Receive data
TX	Output I/O (single-wire/Smartcard mode)	Transmit data. High level when enabled but nothing to be transmitted
CK	Output	Serial clock for synchronous communication
nCTS	Input	Clear to send in hardware flow control mode
nRTS	Output	Request to send in hardware flow control mode

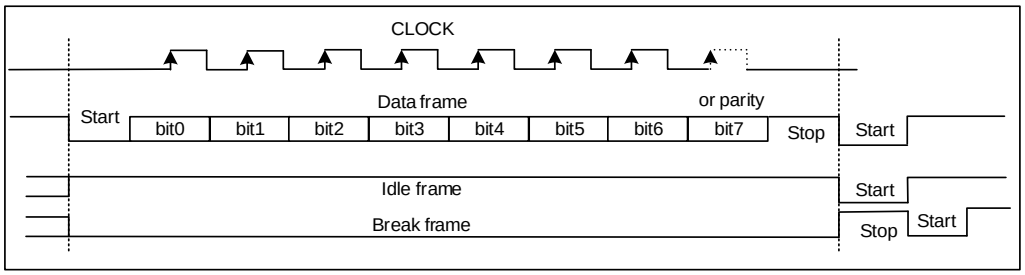
Figure 17-1. USART module block diagram



17.3.1. USART frame format

The USART frame starts with a start bit and ends up with a number of stop bits. The length of the data frame is configured by the WL bit in the USART_CTL0 register. The last data bit can be used as parity check bit by setting the PCEN bit in USART_CTL0 register. When the WL bit is reset, the parity bit is the 7th bit. When the WL bit is set, the parity bit is the 8th bit. The method of calculating the parity bit is selected by the PM bit in USART_CTL0 register.

Figure 17-2. USART character frame (8 bits data and 1 stop bit)



In transmission and reception, the number of stop bits can be configured by the STB[1:0] bits in the USART_CTL1 register.

Table 17-2. Configuration of stop bits

STB[1:0]	stop bit length (bit)	usage description
00	1	default value
01	0.5	Smartcard mode for receiving
10	2	normal USART, single-wire and modem modes
11	1.5	Smartcard mode for transmitting and receiving

In an idle frame, all the frame bits are logic 1. The frame length is equal to the normal USART frame.

A break frame is configured number of low bits followed by the configured number of stop bits. The transfer speed of a USART frame depends on the frequency of the UCLK, the configuration of the baud rate generator and the oversampling mode.

17.3.2. Baud rate generation

The baud-rate divider is a 16-bit number which consists of a 12-bit integer and a 4-bit fractional part. The number formed by these two values is used by the baud rate generator to determine the bit period. Having a fractional baud-rate divider allows the USART to generate all the standard baud rates.

When oversampled by 16, the baud-rate divider (USARTDIV) has the following relationship with the peripheral clock:

$$\text{USARTDIV} = \frac{\text{UCLK}}{16 \times \text{Baud Rate}} \quad (17\text{-}1)$$

1. Get USARTDIV by caculating the value of USART_BUAD:
   If USART_BUAD=0x21D, then INTDIV=33 (0x21), FRADIV=13 (0xD).
   USARTDIV=33+13/16=33.81.
2. Get the value of USART_BUAD by calculating the value of USARTDIV:
   If USARTDIV=30.37, then INTDIV=30 (0x1E).
   16*0.37=5.92, the nearest integer is 6, so FRADIV=6 (0x6).
   USART_BUAD=0x1E6.

Note: If the roundness of FRADIV is 16 (overflow), the carry must be added to the integer part.

17.3.3. USART transmitter

If the transmit enable bit (TEN) in USART_CTL0 register is set, when the transmit data buffer is not empty, the transmitter shifts out the transmit data frame through the TX pin. The polarity of the TX pin can be configured by the TINV bit in the USART_CTL3 register. Clock pulses can output through the CK pin.

After the TEN bit is set, an idle frame will be sent. The TEN bit should not be cleared while the transmission is ongoing.

After power on, the TBE bit is high by default. Data can be written to the USART_DATA when the TBE bit in the USART_STAT0 register is asserted. The TBE bit is cleared by writing USART_DATA register and it is set by hardware after the data is put into the transmit shift register. If a data is written to the USART_DATA register while a transmission is ongoing, it will be firstly stored in the transmit buffer, and transferred to the transmit shift register after the current transmission is done. If a data is written to the USART_DATA register while no transmission is ongoing, the TBE bit will be cleared and set soon, because the data will be

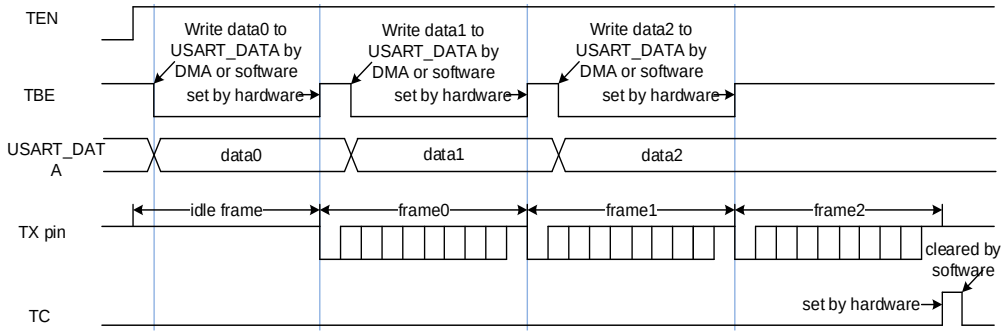
transferred to the transmit shift register immediately.

If a frame is transmitted and the TBE bit is asserted, the TC bit of the USART_STAT0 register will be set. An interrupt will be generated if the corresponding interrupt enable bit (TCIE) is set in the USART_CTL0 register.

The USART transmit procedure is shown in ***Figure 17-3. USART transmit procedure***. The software operating process is as follows:

1. Set the UEN bit in USART_CTL0 to enable the USART.
2. Write the WL bit in USART_CTL0 to set the data bits length.
3. Set the STB[1:0] bits in USART_CTL1 to configure the number of stop bits.
4. Enable DMA (DENT bit) in USART_CTL2 if multibuffer communication is selected.
5. Set the baud rate in USART_BAUD.
6. Set the TEN bit in USART_CTL0.
7. Wait for the TBE to be asserted.
8. Write the data to the USART_DATA register.
9. Repeat step7-8 for each data, if DMA is not enabled.
10. Wait until TC=1 to finish.

Figure 17-3. USART transmit procedure



It is necessary to wait for the TC bit to be asserted before disabling the USART or entering the power saving mode. This bit can be cleared by a software sequence: reading the USART_STAT0 register and then writing the USART_DATA register. If the multibuffer communication is selected (DENT=1), this bit can also be cleared by writing 0 directly.

17.3.4. USART receiver

After power on, the USART receiver can be enabled by the following procedure:

1. Set the UEN bit in USART_CTL0 to enable the USART.
2. Write the WL bit in USART_CTL0 to set the data bits length.
3. Set the STB[1:0] bits in USART_CTL1.
4. Enable DMA (DENR bit) in USART_CTL2 if multibuffer communication is selected.
5. Set the baud rate in USART_BAUD.
6. Set the REN bit in USART_CTL0.

After being enabled, the receiver receives a bit stream after a valid start pulse has been detected. Detection on noisy error, parity error, frame error and overrun error is performed

during the reception of a frame.

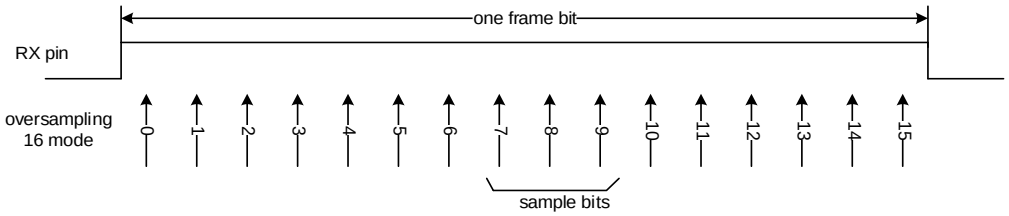
When a frame is received, the RBNE bit in USART_STAT0 is asserted, an interrupt is generated if the corresponding interrupt enable bit (RBNEIE) is set in the USART_CTL0 register. The status of the reception are stored in the USART_STAT0 register.

The software can get the received data by reading the USART_DATA register directly, or through DMA. The RBNE bit is cleared by a read operation on the USART_DATA register, whatever it is performed by software directly, or through DMA.

The REN bit should not be disabled when reception is ongoing, or the current frame will be lost.

By default, the receiver gets three samples to evaluate the value of a frame bit. While in the oversampling 16 mode, the 7th, 8th, and 9th samples are used. If two or more samples of a frame bit is 0, the frame bit is confirmed as a 0, else 1. If the value of the three samples of any bit are not the same, whatever it is data bit, parity bit or stop bit, a noisy error (NERR) will be generated for the frame. An interrupt will be generated if the receive DMA is enabled and the ERRIE bit in USART_CTL2 register is set.

Figure 17-4. Receiving a frame bit by oversampling method



If the parity check function is enabled by setting the PCEN bit in the USART_CTL0 register, the receiver calculates the expected parity value while receiving a frame. The received parity bit will be compared with this expected value. If they are not the same, the parity error (PERR) bit in USART_STAT0 register will be set. An interrupt is generated if the PERRIE bit in USART_CTL0 register is set.

If the RX pin is evaluated as 0 during a stop bit, the frame error (FERR) bit in USART_STAT0 register will be set. An interrupt will be generated if the receive DMA is enabled and the ERRIE bit in USART_CTL2 register is set.

When a frame is received, if the RBNE bit is not cleared yet, the last frame will not be stored in the receive data buffer. The overrun error (ORERR) bit in USART_STAT0 register will be set. An interrupt is generated, if the receive DMA is enabled and the ERRIE bit in USART_CTL2 register is set, or if the RBNEIE is set.

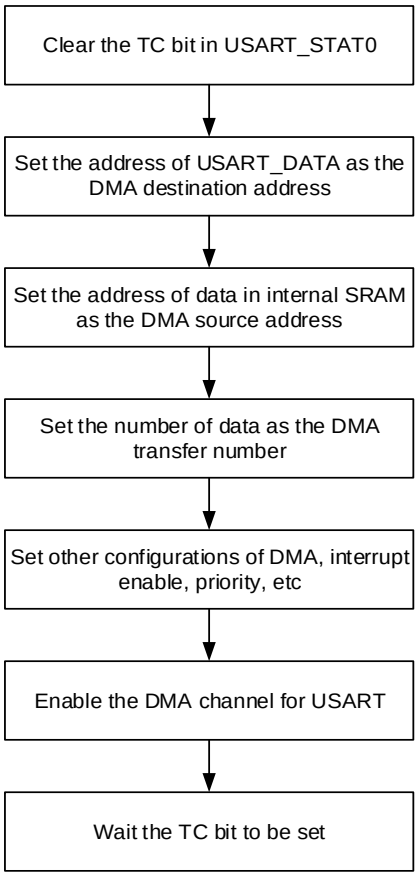
If a noise error (NERR), parity error (PERR), frame error (FERR) or overrun error (ORERR) is generated during a receiving process, then NERR, PERR, FERR or ORERR will be set at same time with RBNE. If DMA is disabled, the software needs to check whether the RBNE interrupt is caused by noise error, parity error, framing error or overflow error when the RBNE interrupt occurs.

17.3.5. Use DMA for data buffer access

To reduce the burden of the processor, DMA can be used to access the transmitting and receiving data buffer. The DENT bit in USART_CTL2 is used to enable the DMA transmission, and the DENR bit in USART_CTL2 is used to enable the DMA reception.

When DMA is used for USART transmission, DMA transfers data from internal SRAM to the transmit data buffer of the USART. The configuration steps are shown in ***Figure 17-5. Configuration steps when using DMA for USART transmission***.

Figure 17-5. Configuration steps when using DMA for USART transmission

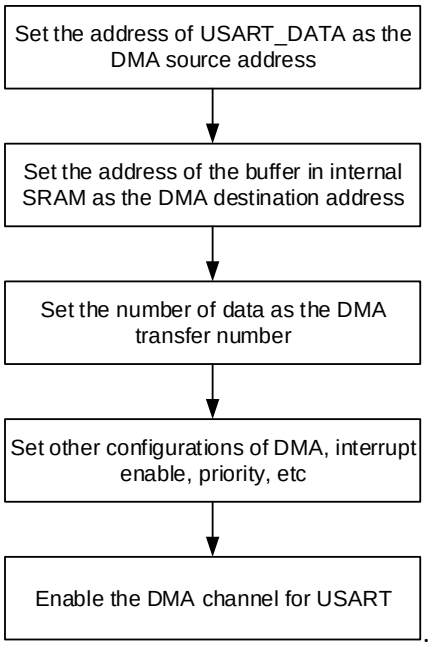


After all of the data frames are transmitted, the TC bit in USART_STAT0 is set. An interrupt occurs if the TCIE bit in USART_CTL0 is set.

When DMA is used for USART reception, DMA transfers data from the receive data buffer of the USART to the internal SRAM. The configuration steps are shown in ***Figure 17-6. Configuration steps when using DMA for USART reception***. If the ERRIE bit in USART_CTL2 is set, interrupts can be generated by the Error status bits (FERR, ORERR

and NERR) in USART_STAT0.

Figure 17-6. Configuration steps when using DMA for USART reception

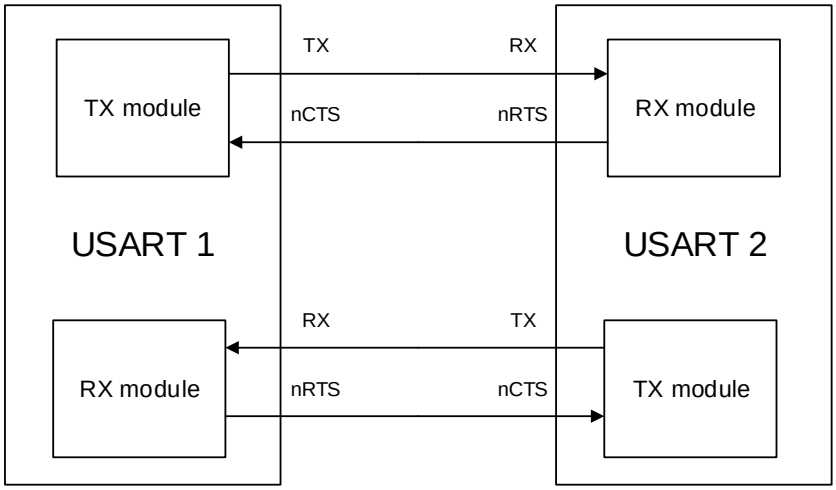


When the number of the data received by USART reaches the DMA transfer number, an end of transfer interrupt will be generated in the DMA module.

17.3.6. Hardware flow control

The hardware flow control function is realized by the nCTS and nRTS pins. The RTS flow control is enabled by writing '1' to the RTSEN bit in USART_CTL2 and the CTS flow control is enabled by writing '1' to the CTSEN bit in USART_CTL2.

Figure 17-7. Hardware flow control between two USARTs



RTS flow control

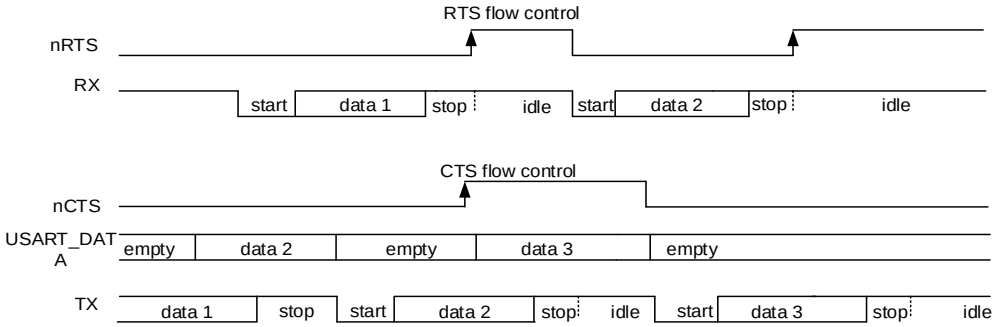
The USART receiver outputs the nRTS, which reflects the status of the receive buffer. When

data frame is received, the nRTS signal goes high to prevent the transmitter from sending next frame. The nRTS signal keeps high when the receive buffer is full, and can be cleared by reading the USART_DATA register.

CTS flow control

The USART transmitter monitors the nCTS input pin to decide whether a data frame can be transmitted. If the TBE bit in USART_STAT0 is '0' and the nCTS signal is low, the transmitter transmits the data frame. When the nCTS signal goes high during a transmission, the transmitter stops after the current transmission is accomplished.

Figure 17-8. Hardware flow control



If the CTS flow control is enabled, the CTSF bit in USART_STAT0 is set when the nCTS pin toggles. An interrupt is generated if the CTSIE bit in USART_CTL2 is set.

17.3.7. Multi-processor communication

In multiprocessor communication, several USARTs are connected as a network. It will be a big burden for a device to monitor all of the messages on the RX pin. To reduce the burden of a device, software can put an USART module into a mute mode by setting the RWU bit in USART_CTL0 register.

If a USART is in mute mode, all of the receive status bits cannot be set. Software can wake up the USART by clearing the RWU bit.

The USART can also be woken up by hardware by one of the two methods: idle frame method and address match method.

The idle frame wake up method is selected by default. When an idle frame is detected on the RX pin, the hardware clears the RWU bit and exits the mute mode. When it is woken up by an idle frame, the IDLEF bit in USART_STAT0 will not be set.

When the WM bit of in USART_CTL0 register is set, the MSB bit of a frame is detected as the address flag. If the address flag is high, the frame is treated as an address frame. If the address flag is low, the frame is treated as a data frame. If the LSB 4 bits of an address frame are the same as the ADDR[3:0] bits in the USART_CTL1 register, the hardware will clear the RWU bit and exits the mute mode. The RBNE bit will be set when the frame that wakes up the USART. The status bits are available in the USART_STAT0 register. If the LSB 4 bits of

an address frame differ from the ADDR[3:0] bits in the USART_CTL1 register, the hardware sets the RWU bit and enters mute mode automatically. In this situation, the RBNE bit is not set.

If the address match method is selected, the receiver does not check the parity value of an address frame by default. If the PCEN bit in USART_CTL0 is set, the MSB bit will be checked as the parity bit, and the bit preceding the MSB bit is detected as the address bit.

17.3.8. LIN mode

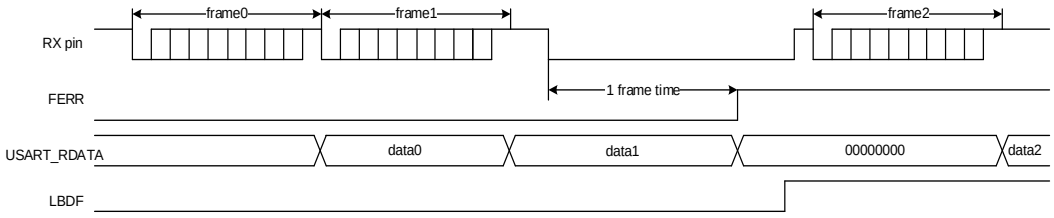
The local interconnection network mode is enabled by setting the LMEN bit in USART_CTL1. The CKEN, WL, STB[1:0] bits in USART_CTL1 and the SCEN, HDEN, IREN bits in USART_CTL2 should be cleared in LIN mode.

When transmitting a normal data frame, the transmission procedure is the same as the normal USART mode. When the SBKCMD bit in USART_CTL0 is set, the USART transmits 13 '0' bits continuously, followed by 1 stop bit.

The break detection function is totally independent of the normal USART receiver. So a break frame can be detected during the idle state or during a frame. The expected length of a break frame can be selected by configuring LBLEN bit in USART_CTL1. When the RX pin is detected at low state for a time that is equal to or longer than the expected break frame length (10 bits when LBLEN=0, or 11 bits when LBLEN=1), the LBDF bit in USART_STAT0 is set. An interrupt occurs if the LBDIE bit in USART_CTL1 is set.

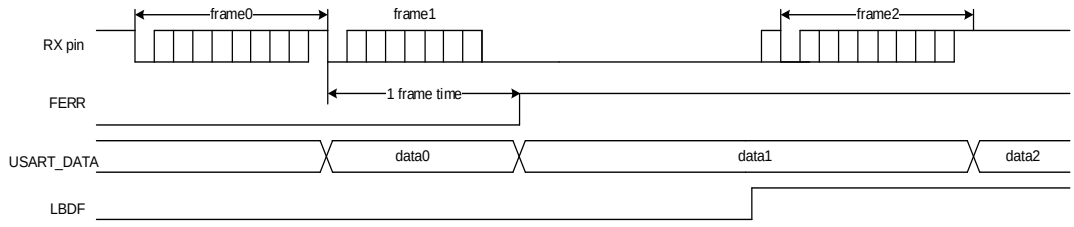
As shown in ***Figure 17-9. Break frame occurs during idle state***, if a break frame occurs during the idle state on the RX pin, the USART receiver will receive an all '0' frame, with an asserted FERR status.

Figure 17-9. Break frame occurs during idle state



As shown in ***Figure 17-10. Break frame occurs during a frame***, if a break frame occurs during a frame on the RX pin, the FERR status will be asserted for the current frame.

Figure 17-10. Break frame occurs during a frame



17.3.9. Synchronous mode

The USART can be used for full-duplex synchronous serial communications only in master mode, by setting the CKEN bit in USART_CTL1. The LMEN bit in USART_CTL1 and SCEN, HDEN, IREN bits in USART_CTL2 should be cleared in synchronous mode. The CK pin is the clock output of the synchronous USART transmitter, and can be only activated when the TEN bit is enabled. No clock pulse will be sent through the CK pin during the transmission of the start bit and stop bit. The CLEN bit in USART_CTL1 can be used to determine whether the clock is output or not during the last (address flag) bit transmission. The CPH bit in USART_CTL1 can be used to determine whether data is captured on the first or the second clock edge. The CPL bit in USART_CTL1 can be used to configure the clock polarity in the USART synchronous idle state.

The CPL, CPH and CLEN bits in USART_CTL1 determine the waveform on the CK pin. Software can only change them when the USART is disabled (UEN=0).

If the REN bit in USART_CTL0 is set, the receiver works differently from the normal USART reception method. The receiver samples the data on the capture edge of the CK pin without any oversampling.

Figure 17-11. Example of USART in synchronous mode

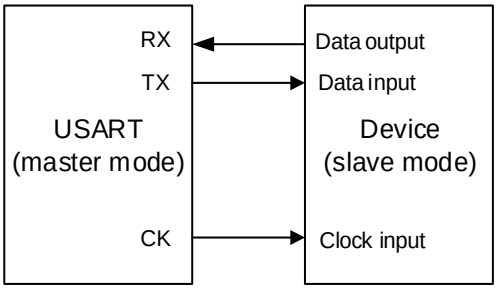
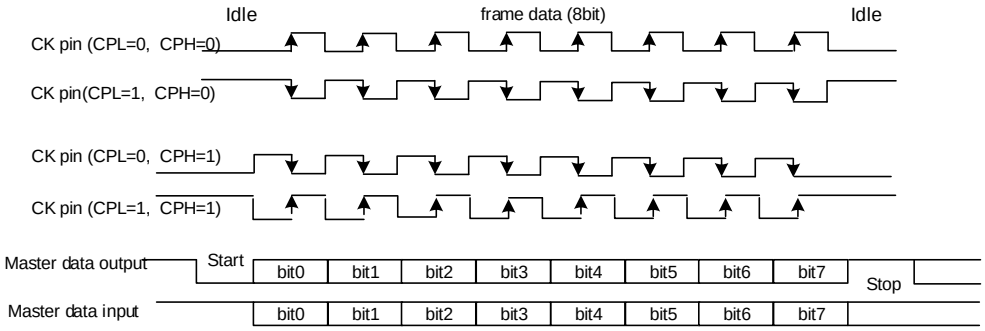


Figure 17-12. 8-bit format USART synchronous waveform (CLEN=1)



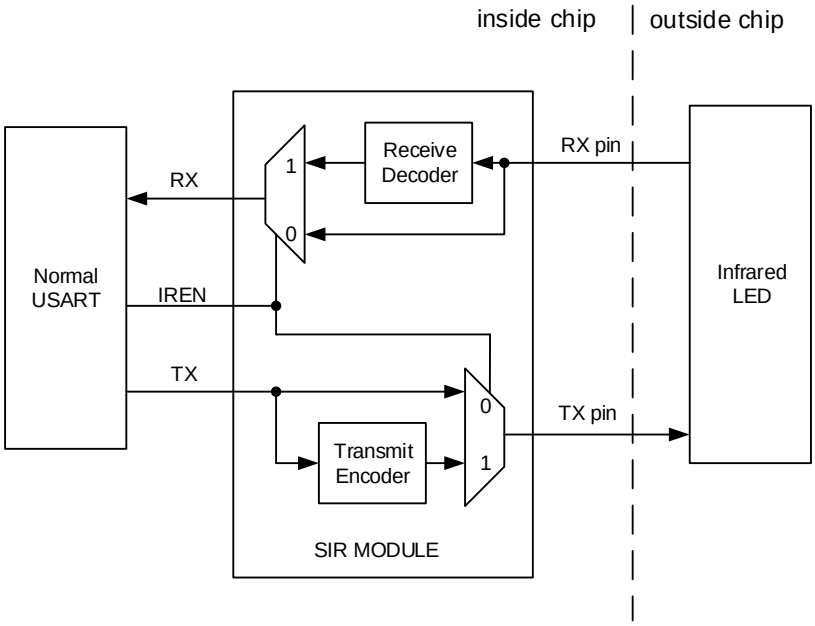
17.3.10. IrDA SIR ENDEC mode

The IrDA mode is enabled by setting the IREN bit in USART_CTL2. The LMEN, STB[1:0], CKEN bits in USART_CTL1 and HDEN, SCEN bits in USART_CTL2 should be cleared in IrDA mode.

In IrDA mode, the USART transmission data frame is modulated in the SIR transmit encoder

and transmitted to the infrared LED through the TX pin. The SIR receive decoder receives the modulated signal from the infrared LED through the RX pin, and puts the demodulated data frame to the USART receiver. The baud rate should not be larger than 115200 for the encoder.

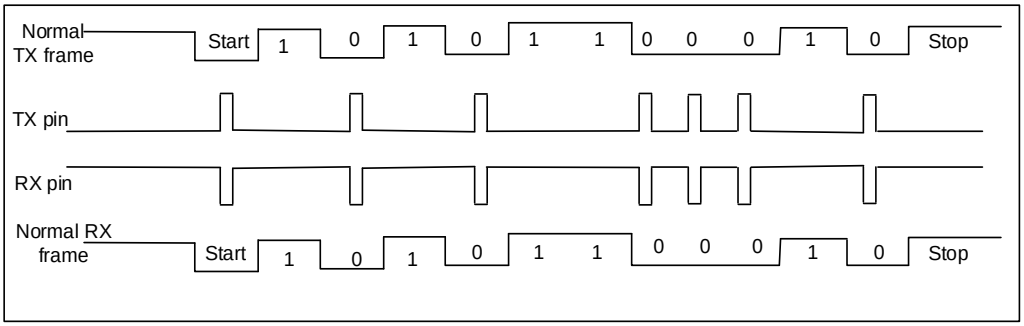
Figure 17-13. IrDA SIR ENDEC module



In IrDA mode, the polarity of the TX pin and RX pin is different. The TX pin is usually at low state, while the RX pin is usually at high state. The IrDA pins keep stable to represent the logic '1', while an infrared light pulse on the IrDA pins (a Return to Zero signal) represents the logic '0'. The pulse width should be 3/16 of a bit period. The IrDA could not detect any pulse if the pulse width is less than 1 PSC clock. While it can detect a pulse by chance if the pulse width is greater than 1 but smaller than 2 times of PSC clock.

Because the IrDA is a half-duplex protocol, the transmission and the reception should not be carried out at the same time in the IrDA SIR ENDEC block.

Figure 17-14. IrDA data modulation



The SIR sub module can work in low power mode by setting the IRLP bit in USART_CTL2. The transmit encoder is driven by a low speed clock, which is divided from the PCLK. The division ratio is configured by the PSC[7:0] bits in USART_GP register, USART BAUD register needs to be configured as 16*PSC[7:0]. The pulse width on the TX pin is 3 cycles of this low speed period. The receiver decoder works in the same manner as the normal IrDA mode.

17.3.11. Half-duplex communication mode

The half-duplex communication mode is enabled by setting the HDEN bit in USART_CTL2. The LMEN, CKEN bits in USART_CTL1 and SCEN, IREN bits in USART_CTL2 should be cleared in half-duplex communication mode.

Only one wire is used in half-duplex mode. The The TX and RX pins are connected together internally, and the RX pin is no longer used. The TX pin should be configured as open drain mode and communication conflicts are handled by software.

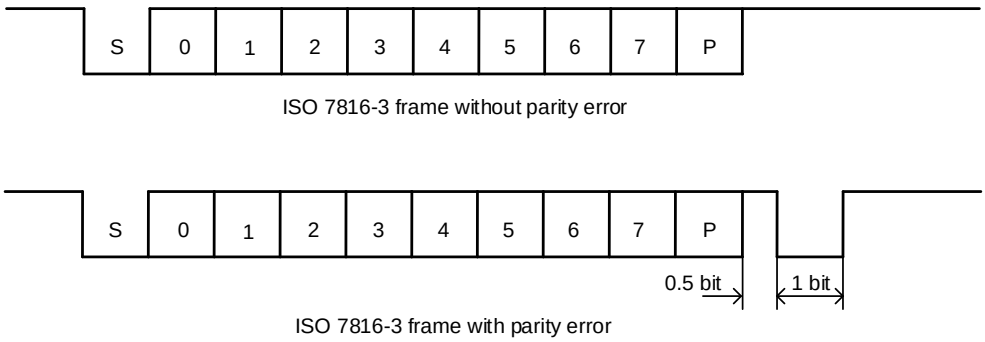
17.3.12. Smartcard (ISO7816-3) mode

The smartcard mode is an asynchronous mode, which is designed to support the ISO7816-3 protocol. Both the character (T=0) mode and the block (T=1) mode are supported. The smartcard mode is enabled by setting the SCEN bit in USART_CTL2. The LMEN bit in USART_CTL1 and HDEN, IREN bits in USART_CTL2 should be cleared in smartcard mode.

A clock is provided to the external smartcard through the CK pin after the CKEN bit is set. The clock is divided from the PCLK. The division ratio is configured by the PSC[4:0] bits in USART_GP register. The CK pin only provides a clock source to the smartcard.

The smartcard mode is a half-duplex communication protocol. When connected to a smartcard, the TX pin must be configured as open drain mode, and an external pull-up resistor will be needed, which drives a bidirectional line that is also driven by the smartcard. The data frame consists of 1 start bit, 9 data bits (1 parity bit included) and 1.5 stop bits. The 0.5 stop bit may be configured for a receiver.

Figure 17-15. ISO7816-3 frame format



Character (T=0) mode

Compared to the timing in normal operation, the transmission time from transmit shift register to the TX pin is delayed by half baud clock, and the TC flag assertion time is delayed by a guard time that is configured by the GUAT[7:0] bits in USART_GP. In smartcard mode, the internal guard time counter starts counting up after the stop bits of the last data frame, and the GUAT[7:0] bits should be configured as the character guard time (CGT) in ISO7816-3 protocol minus 12. The TC status is forced reset while the guard time counter is counting up. When the counter reaches the programmed value TC is asserted high.

During USART transmission, if a parity error event is detected, the smartcard may NACK the current frame by pulling down the TX pin during the last 1 bit time of the stop bits. The USART can automatically resend data according to the protocol for SCRTNUM times. An interframe gap of 2 bits time will be inserted before the start of a resented frame. At the end of the last repeated character the TC bit is set immediately without guard time. The USART will stop transmitting and assert the frame error status if it still receives the NACK signal after the programmed number of retries. The USART will not take the NACK signal as the start bit.

During USART reception, if the parity error is detected in the current frame, the TX pin is pulled low during the last 1 bit time of the stop bits. This signal is the NACK signal to smartcard. Then a frame error occurs in smartcard side. The RBNE/receive DMA request is not activated if the received character is erroneous. According to the protocol, the smartcard can resend the data. The USART stops transmitting the NACK and the error is regarded as a parity error if the received character is still erroneous after the maximum number of retries which is specified in the SCRTNUM bit field. The NACK signal is enabled by setting the NKEN bit in USART_CTL2.

The idle frame and break frame are not supported in the smartcard mode.

Block (T=1) mode

In block (T=1) mode, the NKEN bit in the USART_CTL2 register should be cleared to deactivate the NACK transmission.

When requesting a read from the smartcard, the RT[23:0] bits in USART_RT register should be programmed with the BWT (block wait time) - 11 value and RBNEIE must be set. This timeout period is expressed in baud time units. The RTF bit in USART_STAT1 will be asserted, if no answer is received from the card before the expiration of this period. An interrupt is generated if the RTIE bit in USART_CTL3 is set. The USART generates a RBNE interrupt if the first character is received before the expiration of the RT[23:0] period. If DMA is used to read from the smartcard in block mode, the DMA must be enabled only after the first character is received.

After the first character is received, the RT[23:0] bits should be configured to the CWT (character wait time) - 11 to enable the automatic check of the maximum interframe gap between two consecutive characters. The RTF bit in USART_STAT1 will be asserted, if the smartcard stops sending characters in the RT[23:0] period.

The USART uses a block length counter, which is reset when the USART is transmitting (TBE=0), to count the number of received characters. The length of the block, which must be programmed in the BL[7:0] bits in the USART_RT register, is received from the smartcard in the third byte of the block (prologue field). The block length counter counts up from 0 to the maximum value of BL[7:0] +4. The end of the block status (EBF bit in USART_STAT1) is set after the block length counter reaches the maximum value. An interrupt is generated if the EBIE bit in USART_CTL3 is set. The RTF bit may be set in case that an error in the block length.

If DMA is used for reception, this register field must be programmed to the minimum value

(0x0) before the start of the block. With this value, the end of the block interrupt occurs after the 4th received character. The block length value can be read from the receive buffer at the third byte.

If DMA is not used for reception, the BL[7:0] bits should be firstly configured with the maximum value 0xFF to avoid generating an EBF status. The real block length value can be reconfigured to the BL[7:0] bits after the third byte is received.

Direct and inverse convention

The smartcard protocol defines two conventions: direct and inverse.

When the direct convention is selected, the LSB of the data frame is transferred first, high state on the TX pin represents logic '1', the parity check mode is even. In this case the MSBF and DINV bits in USART_CTL3 should be cleared.

When the inverse convention is selected, the MSB of the data frame is transferred first, high state on the TX pin represents logic '0', the parity check mode is even. In this case the MSBF and DINV bits in USART_CTL3 should be set.

17.3.13. USART interrupts

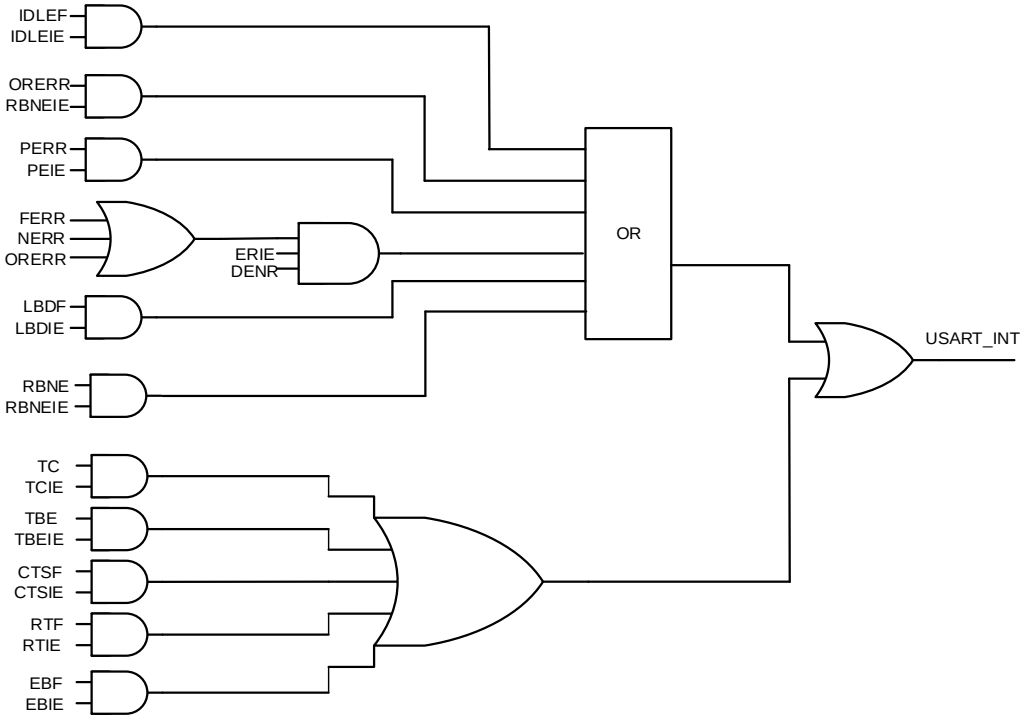
The USART interrupt events and flags are listed in ***Table 17-3. USART interrupt requests***.

Table 17-3. USART interrupt requests

Interrupt event	Event flag	Control register	Enable Control bit
Transmit data buffer empty	TBE	USART_CTL0	TBEIE
CTS toggled flag	CTSF	USART_CTL2	CTSIE
Transmission complete	TC	USART_CTL0	TCIE
Received buff not empty	RBNE	USART_CTL0	RBNEIE
Overrun error	ORERR		
Idle frame	IDLEF	USART_CTL0	IDLEIE
Parity error	PERR	USART_CTL0	PERRIE
Break detected flag in LIN mode	LBDF	USART_CTL1	LBDIE
Receiver timeout	RTF	USART_CTL3	RTIE
End of Block	EBF	USART_CTL3	EBIE
Reception errors (noise flag, overrun error, framing error) in DMA reception	NERR or ORERR or FERR	USART_CTL2	ERRIE

All of the interrupt events are ORed together before being sent to the interrupt controller, so the USART can only generate a single interrupt request to the controller at any given time. Software can service multiple interrupt events in a single interrupt service routine.

Figure 17-16. USART interrupt mapping diagram



17.4. Register definition

USART0 base address: 0x4001 3800

USART1 base address: 0x4000 4400

USART2 base address: 0x4000 4800

UART3 base address: 0x4000 4C00

UART4 base address: 0x4000 5000

17.4.1. Status register 0 (USART_STAT0)

Address offset: 0x00
Reset value: 0x0000 00C0

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CTSF	LBDF	TBE	TC	RBNE	IDLEF	ORERR	NERR	FERR	PERR
						rc_w0	rc_w0	r	rc_w0	rc_w0	r	r	r	r	r

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	CTSF	CTS change flag If CTSEN bit in USART_CTL2 is set, this bit is set by hardware when the nCTS input toggles. An interrupt occurs if the CTSIE bit in USART_CTL2 is set. Software can clear this bit by writing 0 to it. 0: The status of the nCTS line does not change. 1: The status of the nCTS line has changed. This bit is reserved for UART3/4.
8	LBDF	LIN break detected flag This bit is set when LIN break is detected. An interrupt occurs if the LBDIE bit in USART_CTL1 is set. Software can clear this bit by writing 0 to it. 0: The USART does not detect a LIN break. 1: The USART has detected a LIN break.
7	TBE	Transmit data buffer empty This bit is set after power on or when the transmit data has been transferred to the transmit shift register. An interrupt occurs if the TBEIE bit in USART_CTL0 is set. This bit is cleared when the software writes transmit data to the USART_DATA

register.
0: Transmit data buffer is not empty.
1: Transmit data buffer is empty.

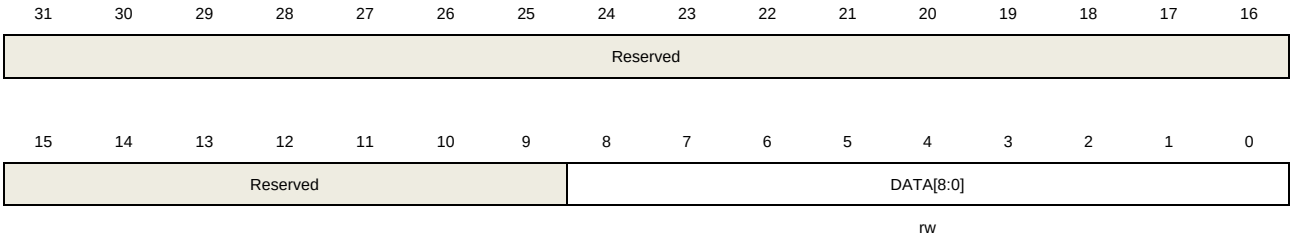
6	TC	Transmission complete This bit is set after power on. If the TBE bit has been set, this bit is set when the transmission of current data is complete. An interrupt occurs if the TCIE bit in USART_CTL0 is set. Software can clear this bit by writing 0 to it. 0: Transmission of current data is not complete. 1: Transmission of current data is complete.
5	RBNE	Read data buffer not empty This bit is set when the read data buffer is filled with a data frame, which has been received through the receive shift register. An interrupt occurs if the RBNEIE bit in USART_CTL0 is set. Software can clear this bit by writing 0 to it or by reading the USART_DATA register. 0: Read data buffer is empty. 1: Read data buffer is not empty.
4	IDLEF	IDLE frame detected flag This bit is set when the RX pin has been detected in idle state for a frame time. An interrupt occurs if the IDLEIE bit in USART_CTL0 is set. Software can clear this bit by reading the USART_STAT0 and USART_DATA registers one by one. 0: The USART module does not detect an IDLE frame. 1: The USART module has detected an IDLE frame.
3	ORERR	Overrun error flag This bit is set if the RBNE is not cleared and a new data frame is received through the receive shift register. An interrupt occurs if RBNEIE bit in USART_CTL0 is set. In multi-processor communication or DMA mode, an interrupt occurs if ERRIE bit in USART_CTL2 is set. Software can clear this bit by reading the USART_STAT0 and USART_DATA registers one by one. 0: The USART does not detect an overrun error. 1: The USART has detected an overrun error.
2	NERR	Noise error flag This bit is set if the USART detects noise on the RX pin when receiving a frame. An interrupt occurs if the ERRIE bit in USART_CTL2 is set. Software can clear this bit by reading the USART_STAT0 and USART_DATA registers one by one. 0: The USART does not detect a noise error. 1: The USART has detected a noise error.

1	FERR	Frame error flag This bit is set when the RX pin is detected low during the stop bits of a receive frame. An interrupt occurs if the ERRIE bit in USART_CTL2 is set. Software can clear this bit by reading the USART_STAT0 and USART_DATA registers one by one. 0: The USART does not detect a framing error. 1: The USART has detected a framing error.
0	PERR	Parity error flag This bit is set when the parity bit of a receive frame does not match the expected parity value. An interrupt occurs if the PERRIE bit in USART_CTL0 is set. Software can clear this bit in the sequence: read the USART_STAT0 register, and then read or write the USART_DATA register. 0: The USART does not detect a parity error 1: The USART has detected a parity error

17.4.2. Data register (USART_DATA)

Address offset: 0x04
Reset value: Undefined

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8:0	DATA[8:0]	Transmitted or received data value Software can write these bits to update the transmitted data or read these bits to get the received data. If the parity check function is enabled, when transmitted data is written to this register, the MSB bit (bit 7 or bit 8 depending on the WL bit in USART_CTL0) will be replaced by the parity bit.

17.4.3. Baud rate register (USART_BAUD)

Address offset: 0x08
Reset value: 0x0000 0000

The software must not write this register when the USART is enabled (UEN=1).

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTDIV[11:0]												FRADIV[3:0]			
rw												rw			

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:4	INTDIV[11:0]	Integer part of baud-rate divider.
3:0	FRADIV[3:0]	Fraction part of baud-rate divider.

17.4.4. Control register 0 (USART_CTL0)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		UEN	WL	WM	PCEN	PM	PERRIE	TBEIE	TCIE	RBNEIE	IDLEIE	TEN	REN	RWU	SBKCMD
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13	UEN	USART enable 0: USART disabled 1: USART enabled
12	WL	Word length 0: 8 data bits 1: 9 data bits
11	WM	Wakeup method in mute mode 0: Wake up by idle frame. 1: Wake up by address match.
10	PCEN	Parity check function enable 0: Parity check function is disabled. 1: Parity check function is enabled.
9	PM	Parity mode

		0: Even parity 1: Odd parity
8	PERRIE	Parity error interrupt enable If this bit is set, an interrupt occurs when the PERR bit in USART_STAT0 is set. 0: Parity error interrupt is disabled. 1: Parity error interrupt is enabled.
7	TBEIE	Transmitter buffer empty interrupt enable If this bit is set, an interrupt occurs when the TBE bit in USART_STAT0 is set. 0: Transmitter buffer empty interrupt is disabled. 1: Transmitter buffer empty interrupt is enabled.
6	TCIE	Transmission complete interrupt enable If this bit is set, an interrupt occurs when the TC bit in USART_STAT0 is set. 0: Transmission complete interrupt is disabled. 1: Transmission complete interrupt is enabled.
5	RBNEIE	Read data buffer not empty interrupt and overrun error interrupt enable If this bit is set, an interrupt occurs when the RBNE bit or the ORERR bit in USART_STAT0 is set. 0: Read data register not empty interrupt and overrun error interrupt disabled. 1: Read data register not empty interrupt and overrun error interrupt enabled.
4	IDLEIE	IDLE line detected interrupt enable If this bit is set, an interrupt occurs when the IDLEF bit in USART_STAT0 is set. 0: IDLE line detected interrupt disabled. 1: IDLE line detected interrupt enabled.
3	TEN	Transmitter enable 0: Transmitter is disabled. 1: Transmitter is enabled.
2	REN	Receiver enable 0: Receiver is disabled. 1: Receiver is enabled.
1	RWU	Receiver wakes up from mute mode. Software can set this bit to make the USART work in mute mode and clear this bit to wake up the USART. If it is configured to wake up by idle frame (WM=0), this bit can be cleared by hardware when an idle frame has been detected. If it is configured to wake up by address matching (WM=1), this bit can be cleared by hardware when receiving an address match frame or set by hardware when receiving an address mismatch frame. 0: Receiver in active mode. 1: Receiver in mute mode.
0	SBKCMD	Send break command

Software can set this bit to send a break frame.

Hardware clears this bit automatically when the break frame has been transmitted.

0: Do not transmit a break frame.

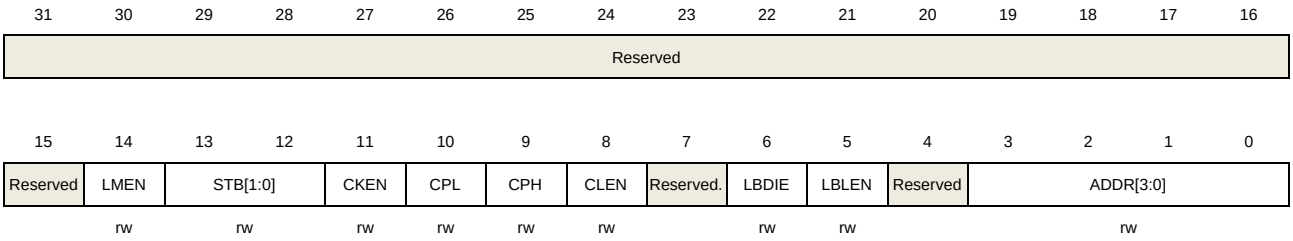
1: Transmit a break frame.

17.4.5. Control register 1 (USART_CTL1)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14	LMEN	LIN mode enable 0: LIN mode disabled 1: LIN mode enabled
13:12	STB[1:0]	Stop bits length 00: 1 stop bit 01: 0.5 stop bits 10: 2 stop bits 11: 1.5 stop bits Only 1 stop bit and 2 stop bits are available for UART3/4.
11	CKEN	CK pin enable 0: CK pin disabled 1: CK pin enabled This bit is reserved for UART3/4.
10	CPL	CK polarity This bit specifies the polarity of the CK pin in synchronous mode. 0: The CK pin is in low state when the USART is in idle state. 1: The CK pin is in high state when the USART is in idle state. This bit is reserved for UART3/4.
9	CPH	CK phase This bit specifies the phase of the CK pin in synchronous mode. 0: The capture edge of the LSB bit is the first edge of CK pin. 1: The capture edge of the LSB bit is the second edge of CK pin.

		This bit is reserved for UART3/4.
8	CLEN	CK length This bit specifies the length of the CK signal in synchronous mode. 0: There are 7 CK pulses for an 8-bit frame and 8 CK pulses for a 9-bit frame. 1: There are 8 CK pulses for an 8-bit frame and 9 CK pulses for a 9-bit frame. This bit is reserved for UART3/4.
7	Reserved	Must be kept at reset value.
6	LBDIE	LIN break detected interrupt enable If this bit is set, an interrupt occurs when the LBDF bit in USART_STAT0 is set. 0: LIN break detected interrupt is disabled. 1: LIN break detected interrupt is enabled.
5	LBLEN	LIN break frame length This bit specifies the length of a LIN break frame. 0: 10 bits 1: 11 bits
4	Reserved	Must be kept at reset value.
3:0	ADDR[3:0]	Address of the USART If it is configured to wake up by address matching (WM=1), the USART enters mute mode when the LSB 4 bits of a received frame do not equal the ADDR[3:0] bits, and wakes up when the LSB 4 bits of a received frame equal the ADDR[3:0] bits.

17.4.6. Control register 2 (USART_CTL2)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					CTSIE	CTSEN	RTSEN	DENT	DENR	SCEN	NKEN	HDEN	IRLP	IREN	ERRIE
					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10	CTSIE	CTS interrupt enable If this bit is set, an interrupt occurs when the CTSF bit in USART_STAT0 is set. 0: CTS interrupt disabled. 1: CTS interrupt enabled.

		This bit is reserved for UART3/4.
9	CTSEN	CTS enable This bit enables the CTS hardware flow control function. 0: CTS hardware flow control disabled. 1: CTS hardware flow control enabled. This bit is reserved for UART3/4.
8	RTSEN	RTS enable This bit enables the RTS hardware flow control function. 0: RTS hardware flow control disabled. 1: RTS hardware flow control enabled. This bit is reserved for UART3/4.
7	DENT	DMA request enable for transmission 0: DMA request is disabled for transmission. 1: DMA request is enabled for transmission.
6	DENR	DMA request enable for reception 0: DMA request is disabled for reception. 1: DMA request is enabled for reception.
5	SCEN	Smartcard mode enable This bit enables the smartcard work mode. 0: Smartcard mode disabled. 1: Smartcard mode enabled. This bit is reserved for UART3/4.
4	NKEN	NACK enable in Smartcard mode This bit enables the NACK transmission when parity error occurs in smartcard mode. 0: Disable NACK transmission. 1: Enable NACK transmission. This bit is reserved for UART3/4.
3	HDEN	Half-duplex enable This bit enables the half-duplex USART mode. 0: Half duplex mode is disabled. 1: Half duplex mode is enabled.
2	IRLP	IrDA low-power This bit selects low-power mode of IrDA mode. 0: Normal mode 1: Low-power mode
1	IREN	IrDA mode enable This bit enables the IrDA mode of USART. 0: IrDA disabled

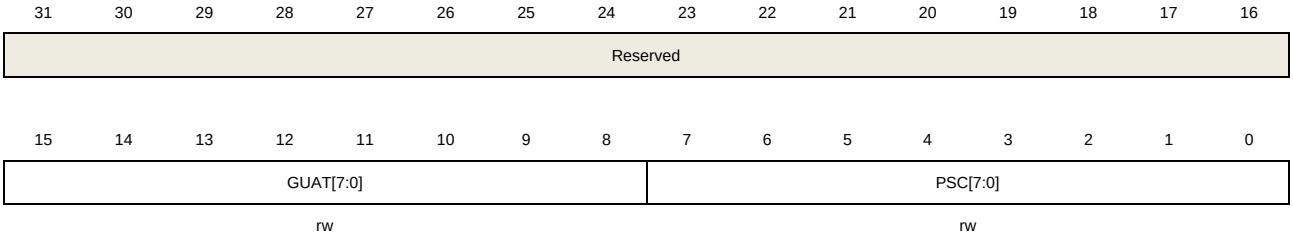
		1: IrDA enabled
0	ERRIE	Error interrupt enable When DMA request for reception is enabled (DENR=1), if this bit is set, an interrupt occurs when any one of the FERR, ORERR and NERR bits in USART_STAT0 is set. 0: Error interrupt disabled. 1: Error interrupt enabled.

17.4.7. Guard time and prescaler register (USART_GP)

Address offset: 0x18

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:8	GUAT[7:0]	Guard time value in smartcard mode TC flag assertion time is delayed by GUAT[7:0] baud clock cycles. These bits are reserved for UART3/4.
7:0	PSC[7:0]	When the USART IrDA low-power mode is enabled, these bits specify the division factor that is used to divide the peripheral clock (PCLK1/PCLK2) to generate the low-power frequency. 00000000: Reserved - never program this value. 00000001: Divided by 1. 00000010: Divided by 2. ... 11111111: Divided by 255. When the USART works in IrDA normal mode, these bits must be set to 00000001. When the USART smartcard mode is enabled, the PSC [4:0] bits specify the division factor that is used to divide the peripheral clock (APB1/APB2) to generate the smartcard clock (CK). The actual division factor is twice as the PSC [4:0] value. 00000: Reserved - never program this value. 00001: Divided by 2. 00010: Divided by 4. ... 11111: Divided by 62.

The PSC [7:5] bits are reserved in smartcard mode.

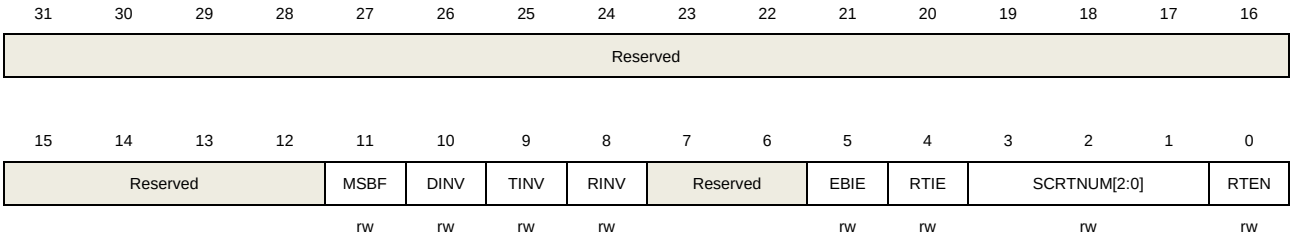
17.4.8. Control register 3 (USART_CTL3)

Address offset: 0x80

Reset value: 0x0000 0000

This register is reserved for UART3/4.

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11	MSBF	Most significant bit first This bit specifies the sequence of the data bits in transmission and reception. 0: Data is transmitted/received with the LSB first. 1: Data is transmitted/received with the MSB first. This bit field cannot be written when the USART is enabled (UEN=1).
10	DINV	Data bit level inversion This bit specifies the polarity of the data bits in transmission and reception. 0: Data bit signal values are not inverted. 1: Data bit signal values are inverted. This bit field cannot be written when the USART is enabled (UEN=1).
9	TINV	TX pin level inversion This bit specifies the polarity of the TX pin. 0: TX pin signal values are not inverted. 1: TX pin signal values are inverted. This bit field cannot be written when the USART is enabled (UEN=1).
8	RINV	RX pin level inversion This bit specifies the polarity of the RX pin. 0: RX pin signal values are not inverted. 1: RX pin signal values are inverted. This bit field cannot be written when the USART is enabled (UEN=1).
7:6	Reserved	Must be kept at reset value.
5	EBIE	Interrupt enable bit of end of block event If this bit is set, an interrupt occurs when the EBF bit in USART_STAT1 is set.

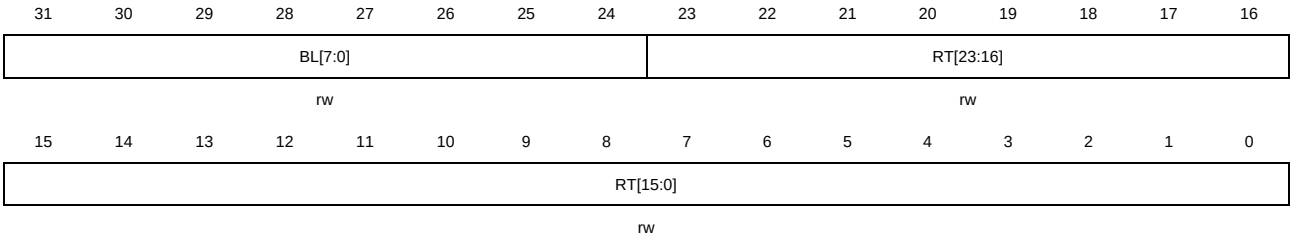
		0: End of block interrupt enabled. 1: End of block interrupt disabled.
4	RTIE	Interrupt enable bit of receive timeout event If this bit is set, an interrupt occurs when the RTF bit in USART_STAT1 is set. 0: Receive timeout interrupt enabled. 1: Receive timeout interrupt disabled.
3:1	SCRTNUM[2:0]	Smartcard auto-retry number In Smartcard mode, these bits specify the number of retries in transmission and reception. In transmission mode, a frame can be retransmitted by SCRTNUM times. If the frame is NACKed by (SCRTNUM+1) times, the FERR is set. In reception mode, a frame reception can be tried by (SCRTNUM+1) times. If the parity bit mismatch event occurs (SCRTNUM+1) times for a frame, the RBNE and PERR bits are set. When these bits are configured as 0x0, there will be no automatic retransmission in transmit mode.
0	RTEN	Receiver timeout enable This bit enables the receive timeout counter of the USART. 0: Receiver timeout function disabled. 1: Receiver timeout function enabled.

17.4.9. Receiver timeout register (USART_RT)

Address offset: 0x84
Reset value: 0x0000 0000

This register is reserved for UART3/4.

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:24	BL[7:0]	Block length These bits specify the block length in Smartcard T=1 reception. The value equals to the number of information characters + the length of the Epilogue Field (1-LEC/2-CRC) - 1. This value, which must be programmed only once per received block, can be programmed after the start of the block reception (using the data from the LEN

		character in the Prologue Field). The block length counter is reset when TBE=0 in Smartcard mode. In other modes, when REN=0 (receiver disabled), or when the EBF bit of USART_STAT1 is written to 0, the block length counter is reset.
23:0	RT[23:0]	Receiver timeout threshold These bits are used to specify receiver timeout value in terms of number of baud clocks. If smartcard mode is not enabled, the RTF bit of USART_STAT1 is set if no new start bit is detected longer than RT bits time after the last received character. If smartcard mode is enabled, the CWT and BWT are implemented by this value. In this case, the timeout measurement is started from the start bit of the last received character. These bits can be written on the fly. The RTF flag will be set if there is no new character is received before the expiration of the RT[23:0] period. These bits must only be programmed once per received character. **Note**：When oversampling by 16, RT[23:0] cannot be configured as 0xFFFFFF.

17.4.10. Status register 1 (USART_STAT1)

Address offset: 0x88

Reset value: 0x0000 0000

This register is reserved for UART3/4.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															BSY
															r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			EBF	RTF	Reserved										
			w0c	w0c											

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	BSY	Busy flag This bit is set when the USART is receiving a data frame. 0: USART reception path is idle. 1: USART reception path is working.
15:13	Reserved	Must be kept at reset value.
12	EBF	End of block flag This bit is set when the number of received bytes (from the start of the block, including the prologue) is equal or greater than BLEN + 4. An interrupt occurs if the EBIE bit in USART_CTL3 is set. Software can clear this bit by writing 0 to it.

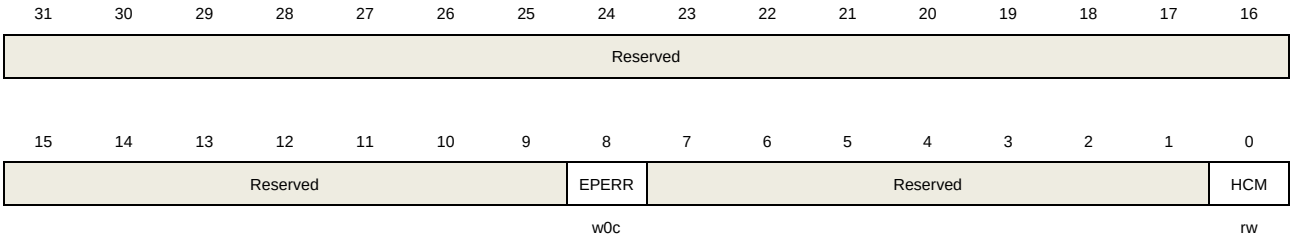
		0: End of block event does not occur. 1: End of block event has occurred.
11	RTF	Receiver timeout flag This bit is set when the RX pin is in idle state for longer than RT bits time. An interrupt occurs if the RTIE bit in USART_CTL3 is set. Software can clear this bit by writing 0 to it. 0: Receiver timeout event does not occur. 1: Receiver timeout event has occurred.
10:0	Reserved	Must be kept at reset value.

17.4.11. Coherence control register (USART_CHC)

Address offset: 0xC0

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	EPERR	Early parity error flag. This flag will be set as soon as the parity bit has been detected, which is before RBNE flag. This flag is cleared by writing 0. 0: No parity error is detected. 1: Parity error is detected.
7:1	Reserved	Must be kept at reset value.
0	HCM	Hardware flow control coherence mode 0: nRTS signal equals to RBNE bit in USART_STAT0 register. 1: nRTS signal is set when the last data bit (parity bit when PCEN is set) has been sampled.

18. Inter-integrated circuit interface (I2C)

18.1. Overview

The I2C (inter-integrated circuit) module provides an I2C interface which is an industry standard two-line serial interface for MCU to communicate with external I2C interface. I2C bus uses two serial lines: a serial data line, SDA, and a serial clock line, SCL.

The I2C interface implements standard I2C protocol with standard-mode, fast-mode and fast-mode-plus as well as CRC calculation and checking, SMBus (system management bus), PMBus (power management bus) and SAM_V (secure access and control module for validation) mode. It also supports multi-master I2C bus. The I2C interface provides DMA mode for users to reduce CPU overload.

18.2. Characteristics

- Parallel-bus to I2C-bus protocol converter and interface.
- Both master and slave functions with the same interface.
- Bi-directional data transfer between master and slave.
- Supports 7-bit and 10-bit addressing and General Call Addressing.
- Multi-master capability.
- Supports standard-mode (up to 100 KHz), fast-mode (up to 400 KHz) and fast-mode-plus (up to 1 MHz).
- Configurable SCL stretching in slave mode.
- Supports DMA mode.
- SMBus 2.0 and PMBus compatible.
- 2 Interrupts: one for successful byte transmission and the other for error event.
- Optional PEC (Packet Error Checking) generation and check.
- Supports SAM_V mode.

18.3. Function overview

Figure 18-1. I2C module block diagram below provides details of the internal configuration of the I2C interface.

Figure 18-1. I2C module block diagram

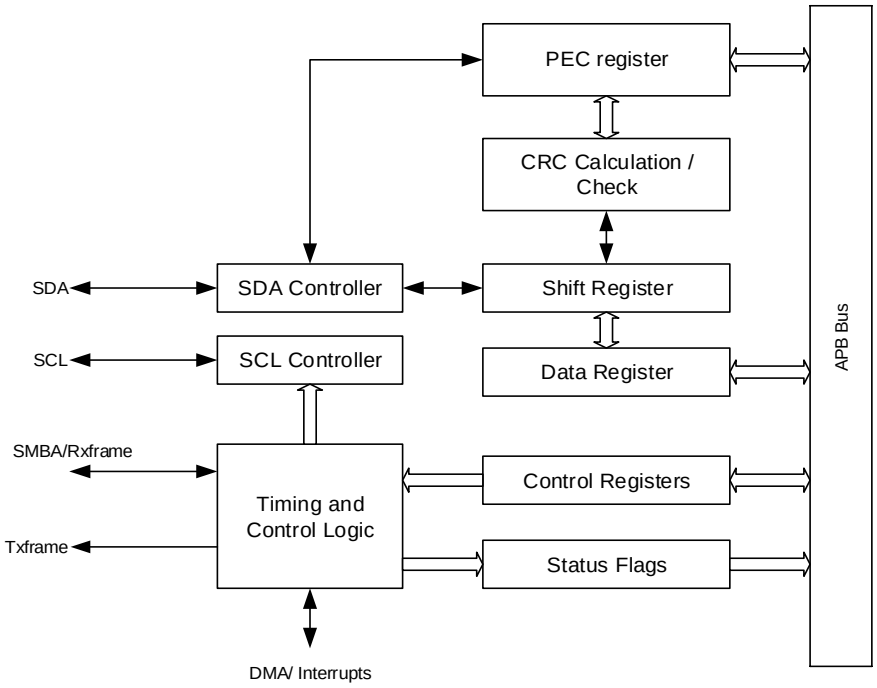


Table 18-1. Definition of I2C-bus terminology (refer to the I2C specification of Philips semiconductors)

Term	Description
Transmitter	The device which sends data to the bus
Receiver	The device which receives data from the bus
Master	The device which initiates a transfer, generates clock signals and terminates a transfer
Slave	The device addressed by a master
Multi-master	More than one master can attempt to control the bus at the same time without corrupting the message
Synchronization	Procedure to synchronize the clock signals of two or more devices
Arbitration	Procedure to ensure that, if more than one master tries to control the bus simultaneously, only one is allowed to do so and the winning master's message is not corrupted

18.3.1. SDA and SCL lines

The I2C module has two external lines, the serial data SDA and serial clock SCL lines. The two wires carry information between the devices connected to the bus.

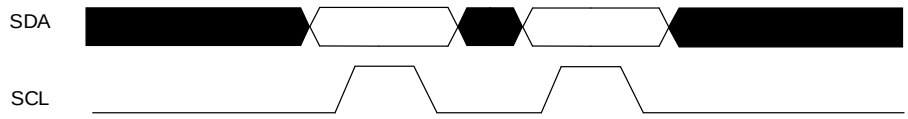
Both SDA and SCL are bidirectional lines, connected to a positive supply voltage via current-source or pull-up resistor. When the bus is free, both lines are HIGH. The output stages of devices connected to the bus must have an open-drain or open-collect to perform the wired-AND function. Data on the I2C-bus can be transferred at rates of up to 100 Kbit/s in the standard mode, up to 400 Kbit/s in the fast mode and up to 1 Mbit/s in the fast mode plus if

the FMPEN bit in I2C_FMPCFG is set. Due to the variety of different technology devices (CMOS, NMOS, bipolar) that can be connected to the I2C-bus, the voltage levels of the logical '0' (LOW) and '1' (HIGH) are not fixed and depend on the associated level of V_{DD}.

18.3.2. Data validation

The data on the SDA line must be stable during the HIGH period of the clock. The HIGH or LOW state of the SDA line can only change when the clock signal on the SCL line is LOW (see ***Figure 18-2. Data validation***). One clock pulse is generated for each data bit to be transferred.

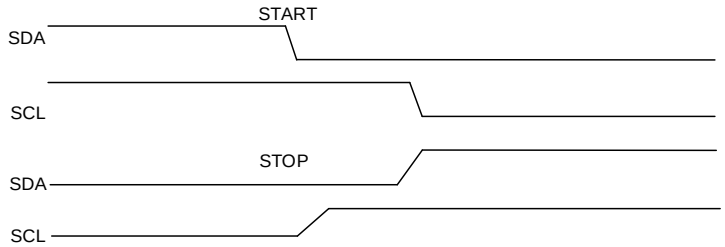
Figure 18-2. Data validation



18.3.3. START and STOP signal

All transmissions begin with a START and are terminated by a STOP (see ***Figure 18-3. START and STOP signal***). A HIGH to LOW transition on the SDA line while SCL is HIGH defines a START signal. A LOW to HIGH transition on the SDA line while SCL is HIGH defines a STOP signal.

Figure 18-3. START and STOP signal



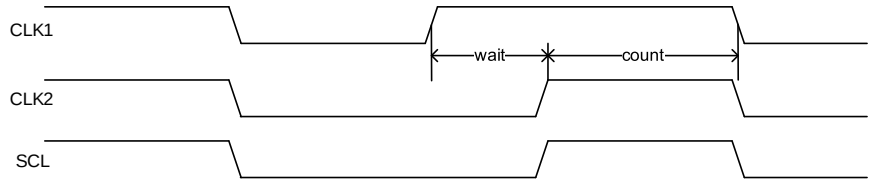
18.3.4. Clock synchronization

Two masters can begin transmitting on a free bus at the same time and there must be a method for deciding which master takes control of the bus and completes its transmission. This is done by clock synchronization and bus arbitration. In a single master system, clock synchronization and bus arbitration are unnecessary.

Clock synchronization is performed using the wired-AND connection of I2C interfaces to the SCL line. This means that a HIGH to LOW transition on the SCL line causes the masters concerned to start counting their LOW period, and once a master clock has gone LOW, it holds the SCL line in that state until the clock HIGH state is reached (see ***Figure 18-4. Clock synchronization***). However, if another clock is still within its LOW period, the LOW to HIGH transition of this clock may not change the state of the SCL line. The SCL line is therefore

held LOW by the master with the longest LOW period. Masters with shorter LOW period enter a HIGH wait-state during this time.

Figure 18-4. Clock synchronization



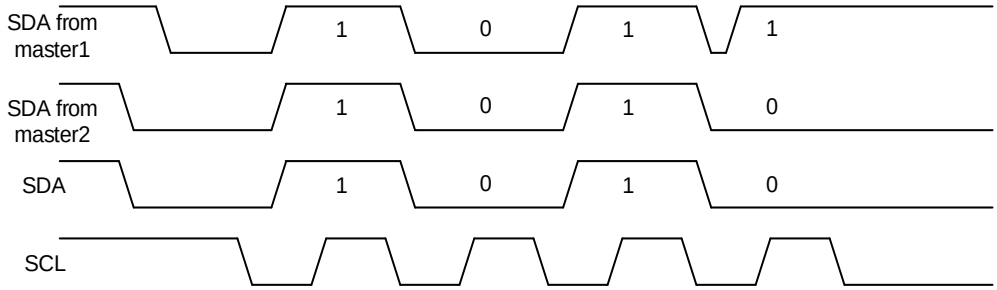
18.3.5. Arbitration

Arbitration, like synchronization, is part of the protocol where more than one master is used in the system. Slaves are not involved in the arbitration procedure.

A master may start a transfer only if the bus is free. Two masters may generate a START signal within the minimum hold time of the START signal which results in a valid START signal on the bus. Arbitration is then required to determine which master will complete its transmission.

Arbitration proceeds bit by bit. During every bit, while SCL is HIGH, each master checks whether the SDA level matches what it has been sent. This process may take many bits. Two masters can even complete an entire transmission without error, as long as the transmissions are identical. The first time a master tries to send a HIGH, but detects that the SDA level is LOW, then the master knows that it has lost the arbitration and turns off its SDA output driver. The other master goes on to complete its transmission.

Figure 18-5. SDA line arbitration



18.3.6. I2C communication flow

Each I2C device is recognized by a unique address (whether it is a microcontroller, LCD driver, memory or keyboard interface) and can be operated as either a transmitter or receiver, depending on the function of the device.

An I2C slave will continue to detect addresses after a START signal on I2C bus and compare the detected address with its slave address which is programmed by software. Once the two addresses match with each other, the I2C slave will send an ACK to the I2C bus and respond to the following command on I2C bus: transmitting or receiving the desired data. Additionally, if General Call is enabled by software, the I2C slave always responds to a General Call

Address (0x00). The I2C block supports both 7-bit and 10-bit address modes.

An I2C master always initiates or ends a transfer using START or STOP signal and it's also responsible for SCL clock generation.

Figure 18-6. I2C communication flow with 7-bit address

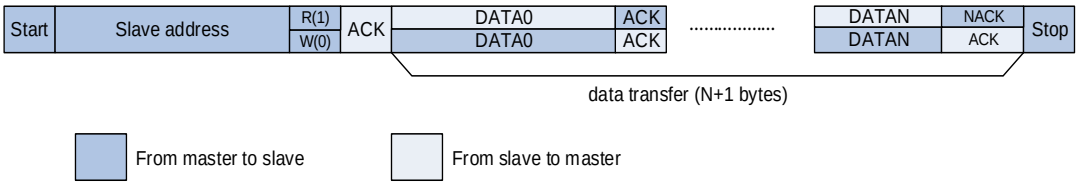


Figure 18-7. I2C communication flow with 10-bit address (Master Transmit)

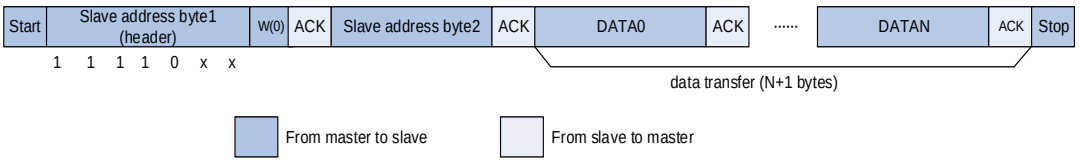
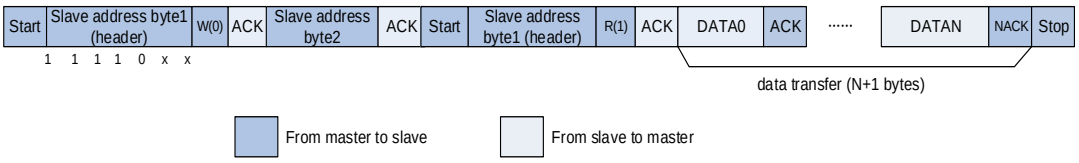


Figure 18-8. I2C communication flow with 10-bit address (Master Receive)



18.3.7. Programming model

An I2C device such as LCD driver may only be a receiver, whereas a memory can both receive and transmit data. In addition to transmitters and receivers, devices can also be considered as masters or slaves when performing data transfers. A master is the device which initiates a data transfer on the bus and generates the clock signals to permit that transfer. At that time, any device addressed is considered as a slave.

An I2C device is able to transmit or receive data whether it's a master or a slave, thus, there're 4 operation modes for an I2C device:

- Master Transmitter
- Master Receiver
- Slave Transmitter
- Slave Receiver

I2C block supports all of the four I2C modes. After system reset, it works in slave mode. After sending a START signal on I2C bus, it changes into master mode. The I2C changes back to slave mode after sending a STOP signal on I2C bus.

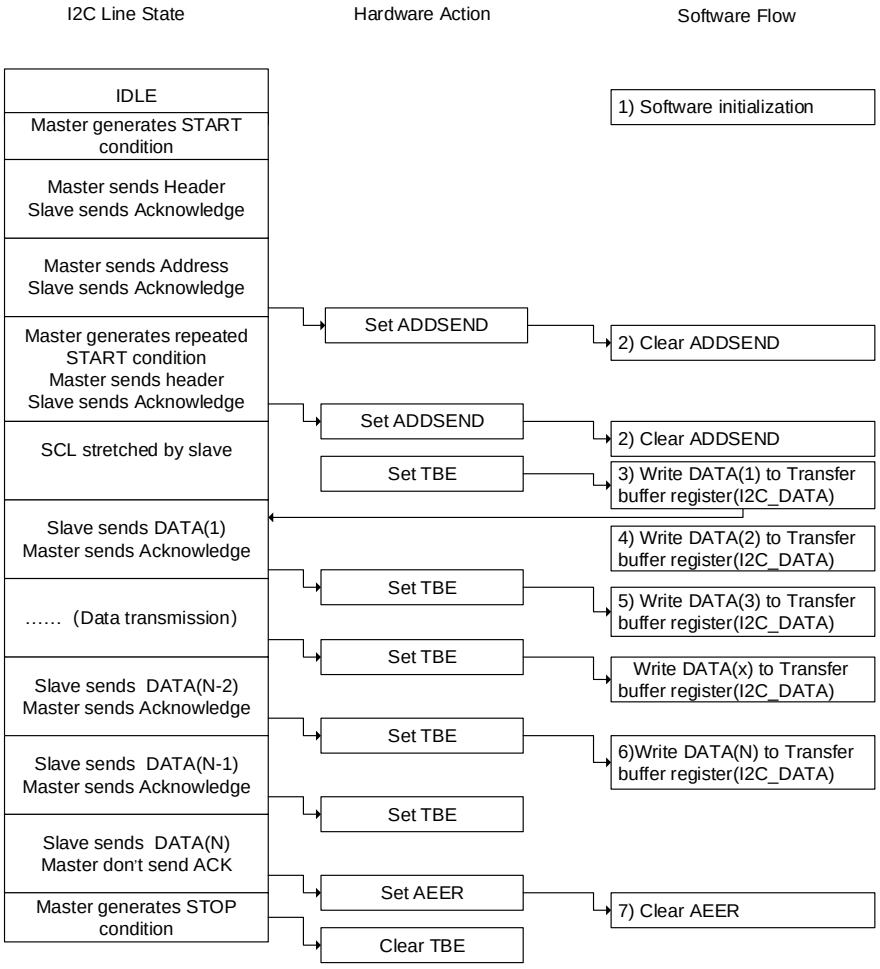
Programming model in slave transmitting mode

As is shown in ***Figure 18-9. Programming model for slave transmitting (10-bit address mode)***, the following software procedure should be followed if users wish to transmit data in

slave transmitter mode:

1. First of all, enable I2C peripheral clock as well as configure clock related registers in I2C_CTL1 to make sure correct I2C timing. After enabled and configured, I2C operates in its default slave state and waits for START signal followed by address on I2C bus.
2. After receiving a START signal followed by a matched address, either in 7-bit format or in 10-bit format, the I2C hardware sets the ADDSEND bit in I2C_STAT0 register, which should be monitored by software either by polling or interrupt. After that, software should read I2C_STAT0 and then I2C_STAT1 to clear ADDSEND bit. If 10-bit addressing format is selected, the I2C master should then send a repeated START signal followed by a header to the I2C bus. The slave sets ADDSEND bit again after it detects the repeated START signal and the following header. The ADDSEND bit must be cleared by software again by reading I2C_STAT0 and then I2C_STAT1.
3. Now I2C enters data transmission stage and hardware sets TBE bit because both the shift register and data register I2C_DATA are empty. Once TBE is set, software should write the first byte of data to I2C_DATA register, TBE is not cleared in this case because the byte written in I2C_DATA is moved to the internal shift register immediately. I2C begins to transmit data to I2C bus as soon as the shift register is not empty.
4. During the transmission of the first byte, software can write the second byte to I2C_DATA, and this time TBE is cleared because neither I2C_DATA nor shift register is empty.
5. After the transmission of the first byte, the TBE bit will be set, the software can write the third byte to the I2C_DATA register and TBE is cleared. After this, any time TBE is set, software can write a byte to I2C_DATA as long as there is still data to be transmitted.
6. During the transmission of the second last byte, software writes the last data to I2C_DATA to clear the TBE flag and doesn't care TBE anymore. So TBE will be set after the byte's transmission and not cleared until a STOP signal.
7. I2C master doesn't acknowledge to the last byte according to the I2C protocol, so after sending the last byte, I2C slave will wait for the STOP signal on I2C bus and sets AERR (Acknowledge Error) bit to notify software that the transmission completes. Software clears AERR bit by writing 0 to it.

Figure 18-9. Programming model for slave transmitting (10-bit address mode)



Programming model in slave receiving mode

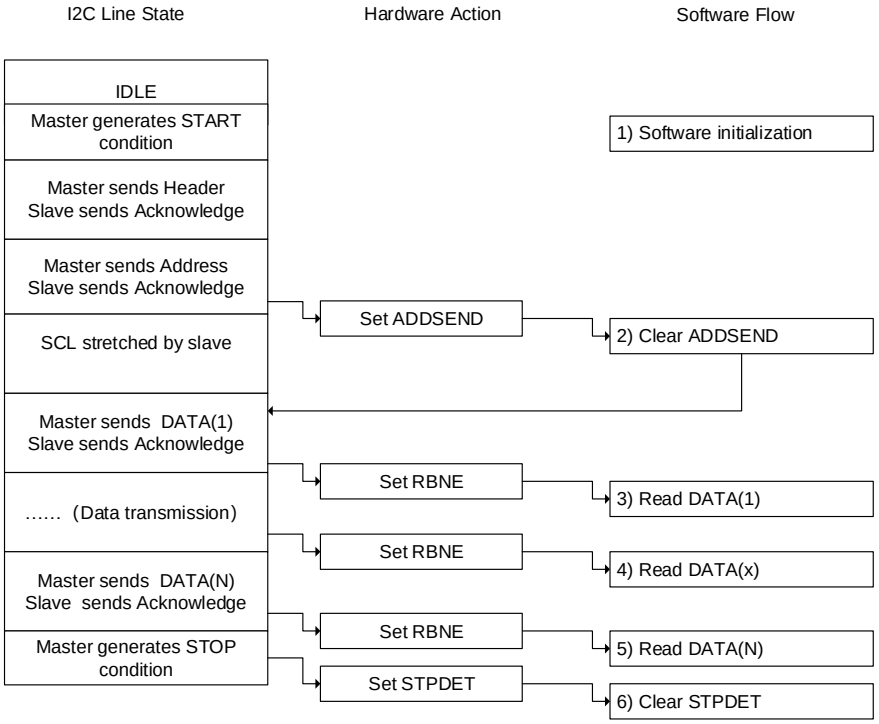
As is shown in ***Figure 18-10. Programming model for slave receiving (10-bit address mode)***, the following software procedure should be followed if users wish to receive data in slave receiver mode:

1. First of all, enable I2C peripheral clock as well as configure clock related registers in I2C_CTL1 to make sure correct I2C timing. After enabled and configured, I2C operates in its default slave state and waits for START signal followed by address on I2C bus.
2. After receiving a START signal followed by a matched 7-bit or 10-bit address, the I2C hardware sets the ADDSEND bit in I2C status register 0, which should be monitored by software either by polling or interrupt. After that software should read I2C_STAT0 and then I2C_STAT1 to clear ADDSEND bit. The I2C begins to receive data on I2C bus as soon as ADDSEND bit is cleared.
3. As soon as the first byte is received, RBNE is set by hardware. Software can now read the first byte from I2C_DATA and RBNE is cleared as well.
4. Any time RBNE is set, software can read a byte from I2C_DATA.

5. After the last byte is received, RBNE is set. Software reads the last byte.

6. STPDET bit is set when I2C detects a STOP signal on I2C bus and software reads I2C_STAT0 and then writes I2C_CTL0 to clear the STPDET bit.

Figure 18-10. Programming model for slave receiving (10-bit address mode)



Programming model in master transmitting mode

As is shown in ***Figure 18-11. Programming model for master transmitting (10-bit address mode)***, the following software procedure should be followed if users wish to make transaction in master transmitter mode:

1. First of all, enable I2C peripheral clock as well as configure clock related registers in I2C_CTL1 to make sure correct I2C timing. After enabled and configured, I2C operates in its default slave state and waits for START signal followed by address on I2C bus.

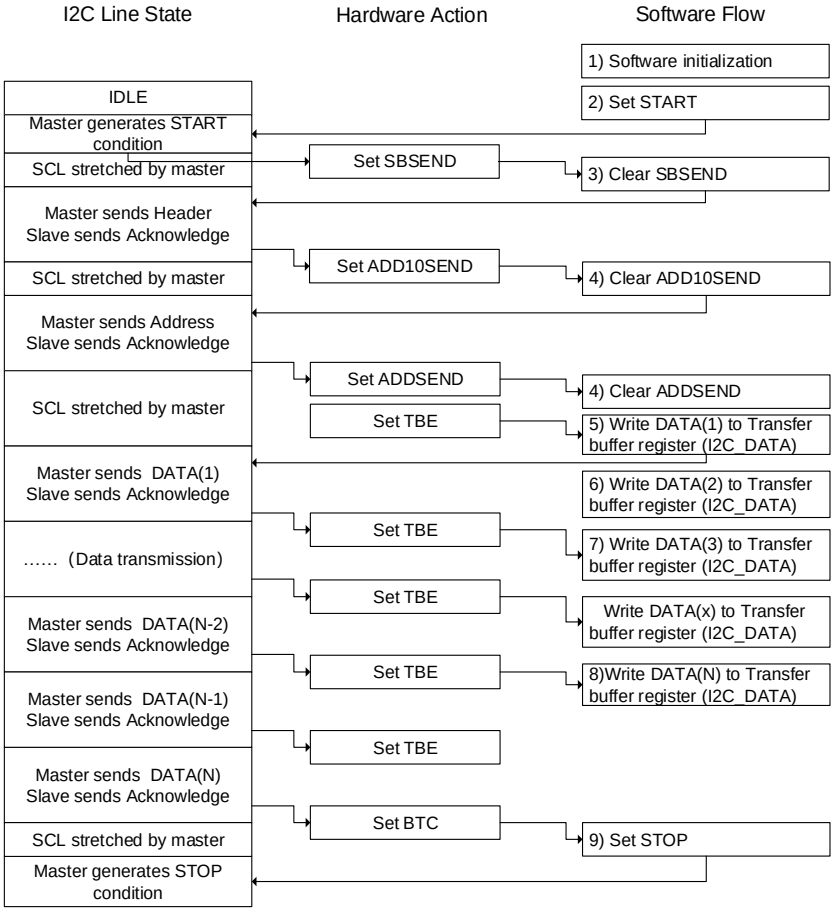
2. Software sets START bit requesting I2C to generate a START signal on I2C bus.

3. After sending a START signal, the I2C hardware sets the SBSEND bit in I2C_STAT0 register and enters master mode. Now software should clear the SBSEND bit by reading I2C_STAT0 and then writing a 7-bit address or header of a 10-bit address to I2C_DATA. I2C begins to send address or header to I2C bus as soon as SBSEND bit is cleared. If the address which has been sent is header of a 10-bit address, the hardware sets ADD10SEND bit after sending the header and software should clear the ADD10SEND bit by reading I2C_STAT0 and writing 10-bit lower address to I2C_DATA.

4. After the 7-bit or 10-bit address has been sent, the I2C hardware sets the ADDSEND bit and software should clear the ADDSEND bit by reading I2C_STAT0 and then I2C_STAT1.

5. Now I2C enters data transmission stage and hardware sets TBE bit because both the shift register and data register I2C_DATA are empty. Software now writes the first byte data to I2C_DATA register, but the TBE will not be cleared because the byte written in I2C_DATA is moved to internal shift register immediately. The I2C begins to transmit data to I2C bus as soon as the shift register is not empty.
6. During the transmission of the first byte, software can write the second byte to I2C_DATA, and this time TBE is cleared because neither I2C_DATA nor shift register is empty.
7. Any time TBE is set, software can write a byte to I2C_DATA as long as there is still data to be transmitted.
8. During the transmission of the second last byte, software writes the last data to I2C_DATA to clear the TBE flag and doesn't care TBE anymore. So TBE will be asserted after the transmission of the byte and not be cleared until a STOP signal.
9. After sending the last byte, I2C master sets BTC bit because both the shift register and I2C_DATA are empty. Software should set the STOP bit to generate a STOP signal, then the I2C clears both TBE and BTC flags.

Figure 18-11. Programming model for master transmitting (10-bit address mode)



Programming model in master receiving mode

In master receiving mode, a master is responsible for generating NACK for the last byte reception and then sending a STOP signal on I2C bus. So, special attention should be paid to ensure the correct ending of data reception. Two solutions for master receiving are provided here for applications: Solution A and B. Solution A requires the software's quick response to I2C events, while Solution B doesn't.

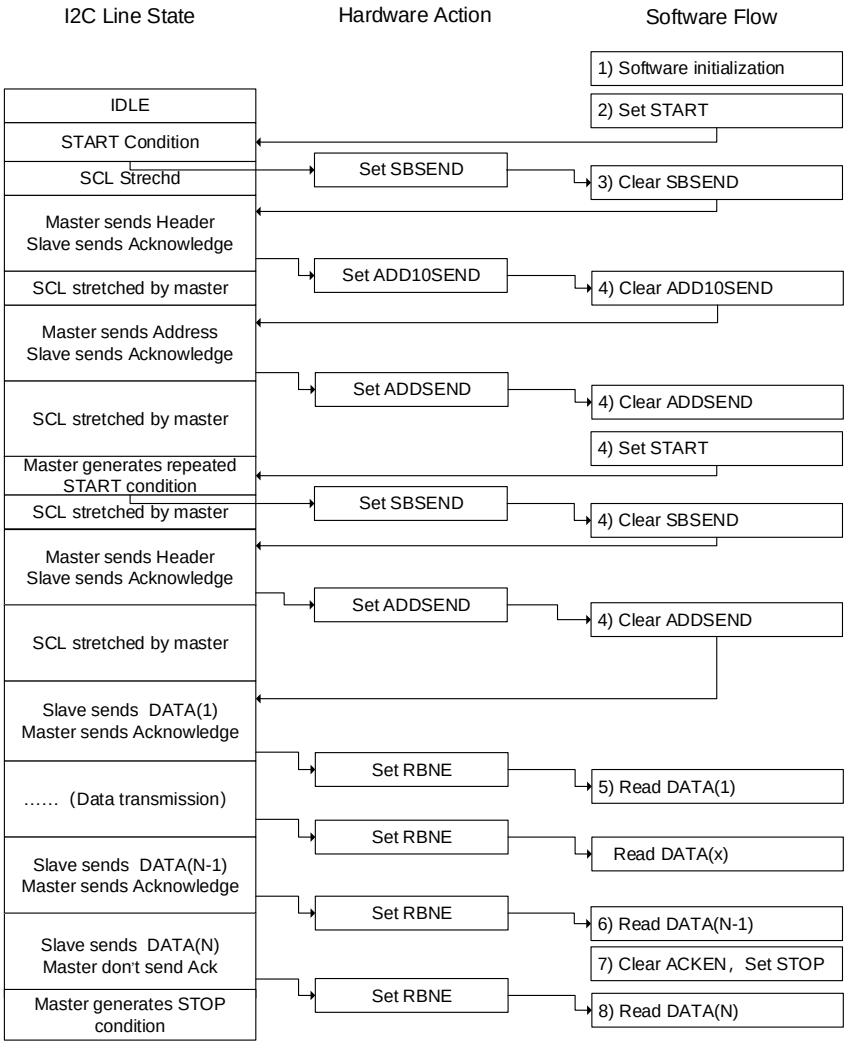
Solution A

1. First of all, enable I2C peripheral clock as well as configure clock related registers in I2C_CTL1 to make sure correct I2C timing. After enabled and configured, I2C operates in its default slave state and waits for START signal followed by address on I2C bus.
2. Software sets START bit requesting I2C to generate a START signal on I2C bus.
3. After sending a START signal, the I2C hardware sets the SBSEND bit in I2C_STAT0 register and enters master mode. Now software should clear the SBSEND bit by reading I2C_STAT0 and then writing a 7-bit address or header of a 10-bit address to I2C_DATA. I2C begins to send address or header to I2C bus as soon as SBSEND bit is cleared. If the address which has been sent is header of a 10-bit address, the hardware sets ADD10SEND bit after sending header and software should clear the ADD10SEND bit by reading I2C_STAT0 and writing 10-bit lower address to I2C_DATA.

4. After the 7-bit or 10-bit address has been sent, the I2C hardware sets the ADDSEND bit and software should clear the ADDSEND bit by reading I2C_STAT0 and then I2C_STAT1. If the address is in 10-bit format, software should then set START bit again to generate a repeated START signal on I2C bus and SBSEND is set after the repeated START is sent out. Software should clear the SBSEND bit by reading I2C_STAT0 and writing header to I2C_DATA. Then the header is sent out to I2C bus, and ADDSEND is set again. Software should again clear ADDSEND by reading I2C_STAT0 and then I2C_STAT1.
5. As soon as the first byte is received, RBNE is set by hardware. Software now can read the first byte from I2C_DATA and RBNE is cleared as well.
6. Any time RBNE is set, software can read a byte from I2C_DATA.
7. After the second last byte (N-1) is received, the software should clear ACKEN bit and set STOP bit. These actions should complete before the end of the last byte's receiving to ensure that NACK will be sent for the last byte.
8. After the last byte is received, RBNE is set. Software reads the last byte. Since ACKEN has been cleared in the previous step, I2C doesn't send ACK for the last byte and it generates a STOP signal after the transmission of the last byte.

The above steps require byte number N>1. If N=1, Step 7 should be performed after Step 4 and completed before the end of the single byte's receiving.

Figure 18-12. Programming model for master receiving using Solution A (10-bit address mode)



Solution B

1. First of all, enable I2C peripheral clock as well as configure clock related registers in I2C_CTL1 to make sure correct I2C timing. After enabled and configured, I2C operates in its default slave state and waits for START signal followed by address on I2C bus.
2. Software sets START bit requesting I2C to generate a START signal on I2C bus.
3. After sending a START signal, the I2C hardware sets the SBSEND bit in I2C_STAT0 register and enters master mode. Now software should clear the SBSEND bit by reading I2C_STAT0 and then writing a 7-bit address or header of a 10-bit address to I2C_DATA. I2C begins to send address or header to I2C bus as soon as SBSEND bit is cleared. If the address which has been sent is a header of 10-bit address, the hardware sets ADD10SEND bit after sending header and software should clear the ADD10SEND bit by reading I2C_STAT0 and writing 10-bit lower address to I2C_DATA.
4. After the 7-bit or 10-bit address has been sent, the I2C hardware sets the ADDSEND bit and software should clear the ADDSEND bit by reading I2C_STAT0 and then I2C_STAT1. If the address is in 10-bit format, software should then set START bit again to generate

a repeated START signal on I2C bus and SBSEND is set after the repeated START is sent out. Software should clear the SBSEND bit by reading I2C_STAT0 and writing header to I2C_DATA. Then the header is sent out to I2C bus, and ADDSEND is set again. Software should again clear ADDSEND by reading I2C_STAT0 and then I2C_STAT1.

5. As soon as the first byte is received, RBNE is set by hardware. Software now can read the first byte from I2C_DATA and RBNE is cleared as well.

6. Any time RBNE is set, software can read a byte from I2C_DATA until the master receives N-3 bytes.

As shown in ***Figure 18-13. Programming model for master receiving mode using solution B (10-bit address mode)***, the N-2 byte is not read out by software, so after the N-1 byte is received, both BTC and RBNE are asserted. The bus is stretched by master to prevent the reception of the last byte. Then software should clear ACKEN bit.

7. Software reads out N-2 byte, clearing BTC. After this, the N-1 byte is moved from shift register to I2C_DATA and bus is released and begins to receive the last byte. Master doesn't send an ACK for the last byte because ACKEN is already cleared.

8. After the last byte is received, both BTC and RBNE are set again, and SCL is stretched low. Software sets STOP bit and master sends out a STOP signal on bus.

9. Software reads the N-1 byte, clearing BTC. After this the last byte is moved from shift register to I2C_DATA.

10. Software reads the last byte, clearing RBNE.

The above steps require that byte number N>2. N=1 and N=2 are similar:

N=1

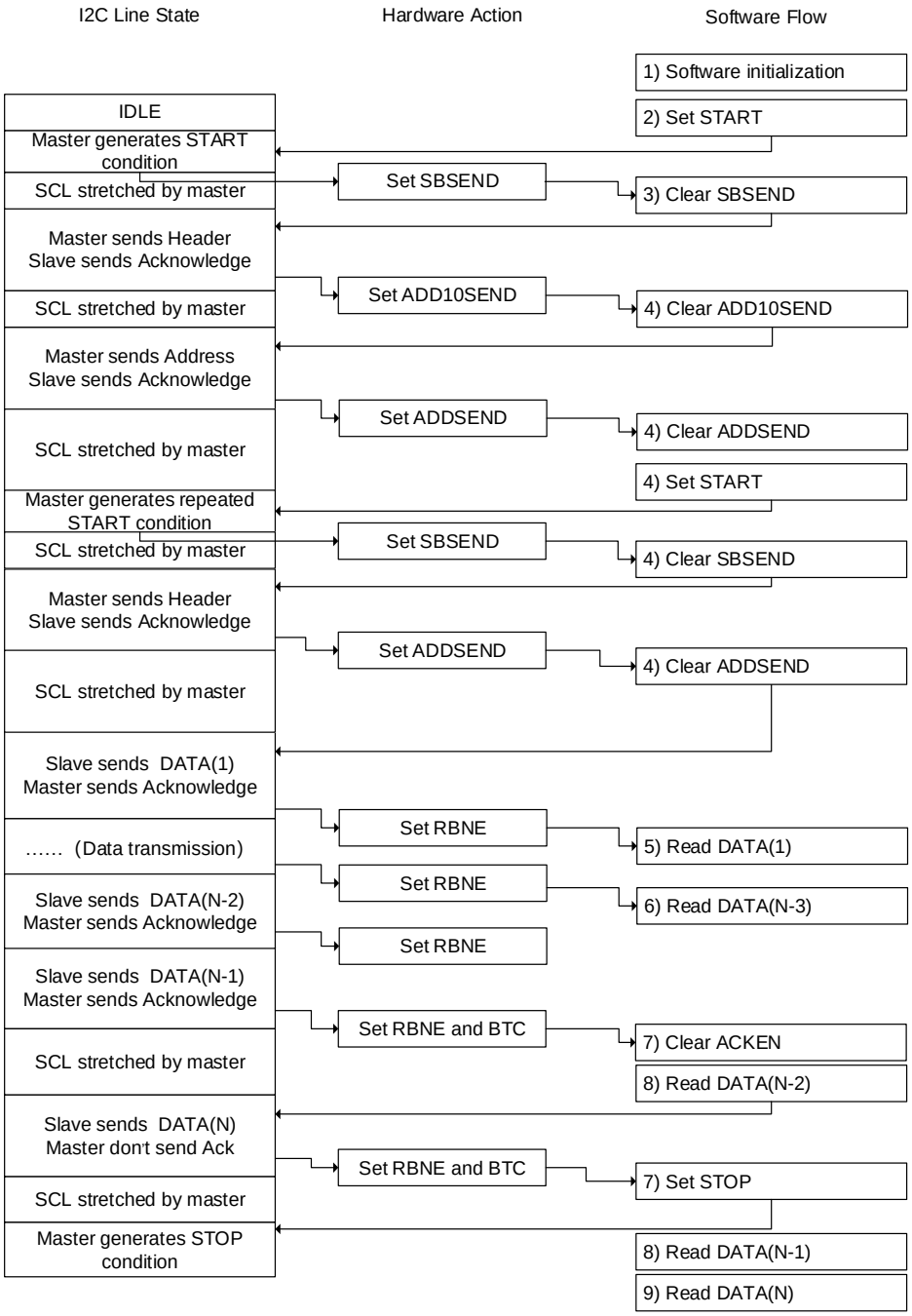
In Step4, software should reset ACKEN bit before clearing ADDSEND bit and set STOP bit after clearing ADDSEND bit. Step 5 is the last step when N=1.

N=2

In Step 2, software should set POAP bit before setting START bit. In Step 4, software should reset ACKEN bit before clearing ADDSEND bit. In Step 5, software should wait until BTC is set and then set STOP bit and read I2C_DATA twice.

Figure 18-13. Programming model for master receiving mode using solution B (10-bit

address mode)



18.3.8. SCL line stretching

The SCL line stretching function is designed to avoid overflow error in reception and underflow error in transmission. As is shown in Programming Model, when the TBE and BTC bits are set in transmitting mode, the transmitter stretches the SCL line low until the transfer buffer register is filled with the next data to be transmitted. When the RBNE and BTC bits are set in receiving mode, the receiver stretches the SCL line low until the data in the transfer buffer is read out.

When works in slave mode, the SCL line stretching function can be disabled by setting the

SS bit in the I2C_CTL0 register. If this bit is set, the software is required to be quick enough to serve the TBE, RBNE and BTC status, otherwise, overflow or underflow situation might occur.

18.3.9. Use DMA for data transfer

As is shown in Programming Model, each time TBE or RBNE is asserted, software should write or read a byte, this may cause CPU to be high overloaded. The DMA controller can be used to process TBE and RBNE flags: each time TBE or RBNE is asserted, DMA controller does a read or write operation automatically. It reduces the load on the CPU. See the DMA section for details on how to configure DMA.

The DMA request is enabled by the DMAON bit in the I2C_CTL1 register. This bit should be set after clearing the ADDSEND status. If the SCL line stretching function is disabled for a slave device, the DMAON bit should be set before the ADDSEND event.

Refer to the specification of the DMA controller for the configuration method of a DMA stream. The DMA controller must be configured and enabled before the I2C transfer. When the configured number of bytes have been transferred, the DMA controller generates End of Transfer (EOT) interrupt. DMA will send an End of Transmission (EOT) signal to the I2C interface and generates a DMA full transfer finish interrupt.

When a master receives two or more bytes, the DMALST bit in the I2C_CTL1 register should be set. The I2C master will send NACK after the last byte. The STOP bit can be set by software to generate a STOP signal in the ISR of the DMA full transfer finish interrupt.

When a master receives only one byte, the ACKEN bit must be cleared before clearing the ADDSEND status. Software can set the STOP bit to generate a STOP signal after clearing the ADDSEND status, or in the ISR of the DMA full transfer finish interrupt.

18.3.10. Packet error checking

There is a CRC-8 calculator in I2C block to perform PEC (Packet Error Checking) for I2C data. The polynomial of the CRC is x8 + x2 + x + 1 which is compatible with the SMBus protocol. If enabled by setting PECEN bit, the PEC will calculate all the data transmitted through I2C including address. I2C is able to send out the PEC value after the last data byte or check the received PEC value with its calculated PEC using the PECTRANS bit. In DMA mode, the I2C will send or check PEC value automatically if PECEN bit and PECTRANS bit are set.

18.3.11. SMBus support

The System Management Bus (abbreviated to SMBus or SMB) is a single-ended simple two-wire bus for the purpose of lightweight communication. Most commonly it is found in computer motherboards for communication with power source for ON / OFF instructions. It is derived from I2C for communication with low-bandwidth devices on a motherboard, especially power related chips such as a laptop's rechargeable battery subsystem (see Smart Battery Data).

SMBus protocol

Each message transmission on SMBus follows the format of one of the defined SMBus protocols. The SMBus protocols are a subset of the data transfer formats defined in the I2C specifications. I2C devices that can be accessed through one of the SMBus protocols are compatible with the SMBus specifications. I2C devices that do not adhere to these protocols cannot be accessed by standard methods as defined in the SMBus and Advanced Configuration and Power Management Interface (abbreviated to ACPI) specifications.

Address resolution protocol

The SMBus is realized based on I2C hardware and it uses I2C hardware addressing, but it adds the second-level software for building special systems. Additionally, its specifications include an Address Resolution Protocol that can make dynamic address allocations. Dynamic reconfiguration of the hardware and software allows bus devices to be 'hot-plugged' and used immediately, without restarting the system. The devices are recognized automatically and assigned unique addresses. This advantage results in a plug-and-play user interface. In this protocol there is a very useful distinction between a system host and all the other devices in the system, that is the host provides address assignment function.

Time-out feature

SMBus has a time-out feature which resets devices if a communication takes too long. This explains the minimum clock frequency is 10 kHz to prevent locking up the bus. I2C can be a 'DC' bus, which means that a slave device stretches the master clock when performing some routines while the master is accessing it. This will notify the master that the slave is busy but does not want to lose the communication. The slave device will continue the communication after its task is completed. There is no limit in the I2C bus protocol of how long this delay can be, whereas for a SMBus system, it would be limited to 25ms. SMBus protocol just assumes that if something takes too long, then it means that there is a problem on the bus and that all devices must reset in order to solve the problem. Slave devices are not allowed to hold the clock low too long.

Packet error checking

SMBus 2.0 and 1.1 allow Packet Error Checking (PEC). In that mode, a PEC byte is appended at the end of each transaction. The byte is a CRC-8 checksum of the entire message including the address and read/write bit. The polynomial used is x8+x2+x+1 (the CRC-8-ATM HEC algorithm, initialized to zero).

SMBus alert

The SMBus has an extra optional shared interrupt signal called SMBALERT# which can be used by slaves to tell the host to ask its slaves about events of interest. SMBus also defines a less common "Host Notify Protocol", providing similar notifications which is based on the I2C multi-master mode but it can pass more data.

SMBus programming flow

The programming flow for SMBus is similar to normal I2C. In order to use SMBus mode, the application should configure several SMBus specific registers, respond to some SMBus specific flags and implement the upper protocols described in SMBus specification.

1. Before communication, SMBEN bit in I2C_CTL0 should be set and SMBSEL and ARPEN bits should be configured to desired values.
2. In order to support address resolution protocol (ARP) (ARPEN=1), the software should respond to HSTSMB flag in SMBus Host Mode (SMBSEL =1) or DEFSMB flag in SMBus Device Mode, and implement the function of ARP protocol.
3. In order to support SMBus Alert Mode, the software should respond to SMBALT flag and implement the related function.

18.3.12. SAM_V support

To support the SAM_V standard, two additional pins are added to the I2C module: txframe and rxframe. Txframe is an output pin, in master mode, it indicates the I2C is busy when it is asserted. Rxframe is an input pin that is supposed to be multiplexed together with the SMBALERT signal.

The SAM_V mode is enabled by setting the SAMEN bit of the I2C_SAMCS register. The status of the txframe and rxframe pin can be reflected by the RFR, RFF, TFR, TFF, RXF, and TXF flags of the I2C_SAMCS register. I2C interrupts will be generated if the corresponding interrupt enable bits are set.

18.3.13. Status, errors and interrupts

There are several status and error flags in I2C, and interrupts may be asserted from these flags by setting some register bits (refer to ***Register definition*** for detail).

Table 18-2. Event status flags

Event Flag Name	Description
SBSEND	START signal sent (master)
ADDSEND	Address sent or received
ADD10SEND	Header of 10-bit address sent
STPDET	STOP signal detected
BTC	Byte transmission completed
TBE	I2C_DATA is empty when transmitting
RBNE	I2C_DATA is not empty when receiving
RFR	SAM_V mode rxframe pin rising edge is detected
RFF	SAM_V mode rxframe pin falling edge is detected
TFR	SAM_V mode txframe pin rising edge is detected
TFF	SAM_V mode txframe pin falling edge is detected

Table 18-3. Error flags

Error Name	Description
BERR	Bus error
LOSTARB	Arbitration lost
OUERR	Over-run or under-run when SCL stretch is disabled.
AERR	No acknowledge received
PECERR	CRC value doesn’t match
SMBTO	Bus timeout in SMBus mode
SMBALT	SMBus Alert

18.4. Register definition

I2C0 base address: 0x4000 5400
I2C1 base address: 0x4000 5800

18.4.1. Control register 0 (I2C_CTL0)

Address offset: 0x00
Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRESET	Reserved	SALT	PECTRANS	POAP	ACKEN	STOP	START	SS	GCEN	PECEN	ARPEN	SMBSEL	Reserved	SMBEN	I2CEN
rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	SRESET	Software resets I2C, software should wait until the I2C lines are released to reset the I2C. 0: I2C is not under reset 1: I2C is under reset
14	Reserved	Must be kept at reset value.
13	SALT	SMBus Alert. Issue alert through SMBA pin. Software can set and clear this bit and hardware can clear this bit. 0: Don’t issue alert through SMBA pin 1: Issue alert through SMBA pin
12	PECTRANS	PEC transfer Software sets and clears this bit while hardware clears this bit when PEC is transferred or START / STOP signal is detected or I2CEN=0. 0: Don’t transfer PEC value 1: Transfer PEC value
11	POAP	Position of ACK and PEC when receiving This bit is set and cleared by software and cleared by hardware when I2CEN=0. 0: ACKEN bit specifies whether to send ACK or NACK for the current byte that is being received. PECTRANS bit indicates that the current receiving byte is a PEC byte. 1: ACKEN bit specifies whether to send ACK or NACK for the next byte that is to be

		received, PECTRANS bit indicates the next byte that is to be received is a PEC byte.
10	ACKEN	Whether or not to send an ACK This bit is set and cleared by software and cleared by hardware when I2CEN=0. 0: ACK will not be sent 1: ACK will be sent
9	STOP	Generate a STOP signal on I2C bus This bit is set and cleared by software and set by hardware when SMBus timeout and cleared by hardware when STOP signal is detected. 0: STOP will not be sent 1: STOP will be sent
8	START	Generate a START signal on I2C bus This bit is set and cleared by software and cleared by hardware when a START signal is detected or I2CEN=0. 0: START will not be sent 1: START will be sent
7	SS	Whether to stretch SCL low when data is not ready in slave mode. This bit is set and cleared by software. 0: SCL stretching is enabled 1: SCL stretching is disabled
6	GCEN	Whether or not to response to a General Call (0x00) 0: Slave won't respond to a General Call 1: Slave will respond to a General Call
5	PECEN	PEC calculation enable 0: PEC calculation disable 1: PEC calculation enable
4	ARPEN	ARP protocol enable in SMBus mode 0: ARP is disabled 1: ARP is enabled
3	SMBSEL	SMBus type selection 0: Device 1: Host
2	Reserved	Must be kept at reset value.
1	SMBEN	SMBus/I2C mode switch 0: I2C mode 1: SMBus mode
0	I2CEN	I2C peripheral enable 0: I2C is disabled

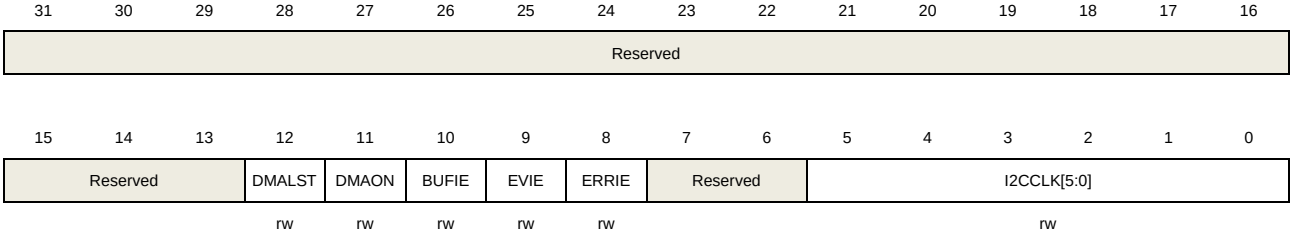
1: I2C is enabled

18.4.2. Control register 1 (I2C_CTL1)

Address offset: 0x04

Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12	DMALST	DMA last transfer configure 0: Next DMA EOT is not the last transfer 1: Next DMA EOT is the last transfer
11	DMAON	DMA mode switch 0: DMA mode switched off 1: DMA mode switched on
10	BUFIE	Buffer interrupt enable 0: Buffer interrupt is disabled, TBE = 1 or RBNE = 1 when EVIE=1 will not generate an interrupt. 1: Buffer interrupt is enabled, which means that interrupt will be generated when TBE = 1 or RBNE = 1 if EVIE=1.
9	EVIE	Event interrupt enable 0: Event interrupt is disabled 1: Event interrupt is enabled, which means that interrupt will be generated when SBSEND, ADDSEND, ADD10SEND, STPDET or BTC flag asserted or TBE=1 or RBNE=1 if BUFIE=1.
8	ERRIE	Error interrupt enable 0: Error interrupt is disabled 1: Error interrupt is enabled, which means that interrupt will be generated when BERR, LOSTARB, AERR, OUERR, PECERR, SMBTO or SMBALT flag is asserted.
7:6	Reserved	Must be kept at reset value.
5:0	I2CCLK[5:0]	I2C peripheral clock frequency I2CCLK[5:0] should be the frequency of input APB1 clock in MHz which is at least 2.

000000 - 000001: Not allowed

000010 - 111100: 2 MHz~60 MHz

111101 - 111111: Not allowed due to the limitation of APB1 clock

Note: In I2C standard mode, the frequencies of APB1 must be equal or greater than 2MHz. In I2C fast mode, the frequencies of APB1 must be equal or greater than 8MHz. In I2C fast mode plus, the frequencies of APB1 must be equal or greater than 24MHz.

18.4.3. Slave address register 0 (I2C_SADDR0)

Address offset: 0x08

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDFORMAT	Reserved					ADDRESS[9:8]		ADDRESS[7:1]							ADDRESS0
rw						rw		rw							rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	ADDFORMAT	Address format for the I2C slave 0: 7-bit address 1: 10-bit address
14:10	Reserved	Must be kept at reset value.
9:8	ADDRESS[9:8]	Highest two bits of a 10-bit address
7:1	ADDRESS[7:1]	7-bit address or bits 7:1 of a 10-bit address
0	ADDRESS0	Bit 0 of a 10-bit address

18.4.4. Slave address register 1 (I2C_SADDR1)

Address offset: 0x0C

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								ADDRESS2[7:1]							DUADEN

											rw					rw

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:1	ADDRESS2[7:1]	The second I2C address for the slave in Dual-Address mode
0	DUADEN	Dual-Address mode enable 0: Dual-Address mode is disabled 1: Dual-Address mode is enabled

18.4.5. Transfer buffer register (I2C_DATA)

Address offset: 0x10

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TRB[7:0]							
								rw							

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	TRB[7:0]	Transmission or reception data buffer

18.4.6. Transfer status register 0 (I2C_STAT0)

Address offset: 0x14

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMBALT	SMBTO	Reserved	PECERR	OUERR	AERR	LOSTARB	BERR	TBE	RBNE	Reserved	STPDET	ADD10SEND	BTC	ADDSEND	SBSEND
rc_w0	rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	r	r		r	r	r	r	r

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	SMBALT	SMBus Alert status

		This bit is set by hardware and cleared by writing 0. 0: SMBA pin not pulled down (device mode) or no Alert detected (host mode) 1: SMBA pin pulled down and Alert address received (device mode) or Alert detected (host mode)
14	SMBTO	Timeout signal in SMBus mode This bit is set by hardware and cleared by writing 0. 0: No timeout error 1: Timeout event occurs (SCL is low for 25 ms)
13	Reserved	Must be kept at reset value.
12	PECERR	PEC error when receiving data This bit is set by hardware and cleared by writing 0. 0: Received PEC matches the calculated PEC 1: Received PEC doesn't match the calculated PEC, I2C will send NACK careless of ACKEN bit.
11	OUERR	Over-run or under-run situation occurs in slave mode, when SCL stretching is disabled. In slave receiving mode, if the last byte in I2C_DATA is not read out while the following byte is already received, over-run occurs. In slave transmitting mode, if the current byte is already sent out, while the I2C_DATA is still empty, under-run occurs. This bit is set by hardware and cleared by writing 0. 0: No over-run or under-run occurs. 1: Over-run or under-run occurs.
10	AERR	Acknowledge error This bit is set by hardware and cleared by writing 0. 0: No acknowledge error 1: Acknowledge error
9	LOSTARB	Arbitration lost in master mode This bit is set by hardware and cleared by writing 0. 0: No arbitration lost 1: Arbitration lost occurs and the I2C block changes back to slave mode.
8	BERR	Bus error A bus error occurs when an unexpected START or STOP signal on I2C bus. This bit is set by hardware and cleared by writing 0. 0: No bus error 1: A bus error detected
7	TBE	I2C_DATA is empty during transmitting This bit is set by hardware after it moves a byte from I2C_DATA to shift register and cleared by writing a byte to I2C_DATA. If both the shift register and I2C_DATA are empty, writing I2C_DATA won't clear TBE (refer to Programming Model for detail). 0: I2C_DATA is not empty

		1: I2C_DATA is empty, software can write
6	RBNE	I2C_DATA is not empty during receiving This bit is set by hardware after it moves a byte from shift register to I2C_DATA and cleared by reading I2C_DATA. If both BTC and RBNE are asserted, reading I2C_DATA won't clear RBNE because the byte in shift register will be moved to I2C_DATA immediately. 0: I2C_DATA is empty 1: I2C_DATA is not empty, software can read
5	Reserved	Must be kept at reset value.
4	STPDET	STOP signal is detected in slave mode This bit is set by hardware and cleared by reading I2C_STAT0 and then writing I2C_CTL0. 0: STOP signal not detected in slave mode 1: STOP signal detected in slave mode
3	ADD10SEND	Header of 10-bit address is sent in master mode This bit is set by hardware and cleared by reading I2C_STAT0 and writing I2C_DATA. 0: No header of 10-bit address is sent in master mode 1: Header of 10-bit address is sent in master mode
2	BTC	Byte transmission is completed. If a byte is already received in shift register but I2C_DATA is still full in receiving mode or a byte is already sent out from shift register but I2C_DATA is still empty in transmitting mode, the BTC flag is asserted if SCL stretching enabled. This bit is set by hardware and cleared by 3 ways as follow: 1. Software clearing: reading I2C_STAT0 followed by reading or writing I2C_DATA 2. Hardware clearing: sending the STOP signal or START signal 3. Bit 0 (I2CEN bit) of the I2C_CTL0 is reset. 0: BTC not asserted 1: BTC asserted
1	ADDSEND	Address is sent and ACK is received in master mode or address is received and matches with its own address in slave mode. This bit is set by hardware and cleared by reading I2C_STAT0 and reading I2C_STAT1. 0: In slave mode, no address is received or the received address does not match witih its own address. In master mode, no address is sent or address has been sent but not received the ACK from slave. 1: In slave mode, address is received and matches witih its own address. In master mode, address has been sent and receives the ACK from slave.
0	SBSEND	START signal is sent out in master mode This bit is set by hardware and cleared by reading I2C_STAT0 and writing I2C_DATA.

0: No START signal sent

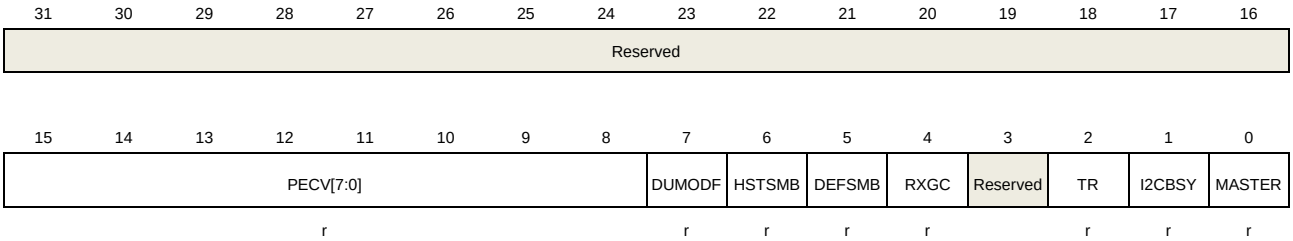
1: START signal sent

18.4.7. Transfer status register 1 (I2C_STAT1)

Address offset: 0x18

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:8	PECV[7:0]	Packet Error Checking value that calculated by hardware when PEC is enabled.
7	DUMODF	Dual flag in slave mode indicates which address matches with the address in Dual-Address mode This bit is cleared by hardware after a STOP or a START signal or I2CEN=0 0: The address matches with SADDR0 address 1: The address matches with SADDR1 address
6	HSTSMB	SMBus host header detected in slave mode This bit is cleared by hardware after a STOP or a START signal or I2CEN=0 0: No SMBus host header is detected 1: SMBus host header is detected
5	DEFSMB	Default address of SMBus device This bit is cleared by hardware after a STOP or a START signal or I2CEN=0. 0: The default address has not been received for SMBus device 1: The default address has been received for SMBus device
4	RXGC	General call address (0x00) received. This bit is cleared by hardware after a STOP or a START signal or I2CEN=0. 0: No general call address (0x00) received 1: General call address (0x00) received
3	Reserved	Must be kept at reset value.
2	TR	Transmitter or receiver This bit indicates whether the I2C is a transmitter or a receiver. It is cleared by hardware after a STOP or a START signal or I2CEN=0 or LOSTARB=1. 0: Receiver

		1: Transmitter
1	I2CBSY	Busy flag This bit is cleared by hardware after a STOP signal 0: No I2C communication. 1: I2C communication active.
0	MASTER	A flag indicating whether I2C block is in master or slave mode. This bit is set by hardware when a START signal generates. This bit is cleared by hardware after a STOP signal or I2CEN=0 or LOSTARB=1. 0: Slave mode 1: Master mode

18.4.8. Clock configure register (I2C_CKCFG)

Address offset: 0x1C
Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FAST	DTCY	Reserved		CLKC[11:0]											
rw	rw			rw											

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	FAST	I2C speed selection in master mode 0: Standard speed 1: Fast speed
14	DTCY	Duty cycle in fast mode or fast mode plus 0: $T_{low}/T_{high}=2$ 1: $T_{low}/T_{high}=16/9$
13:12	Reserved	Must be kept at reset value.
11:0	CLKC[11:0]	I2C clock control in master mode In standard speed mode: $T_{high}=T_{low}=CLKC*T_{PCLK1}$ In fast speed mode or fast mode plus, if DTCY=0: $T_{high}=CLKC*T_{PCLK1}$, $T_{low}=2*CLKC*T_{PCLK1}$ In fast speed mode or fast mode plus, if DTCY=1: $T_{high}=9*CLKC*T_{PCLK1}$, $T_{low}=16*CLKC*T_{PCLK1}$ **Note:** If DTCY is 0, when PCLK1 is an integral multiple of 3, the baud rate will be more accurate. If DTCY is 1, when PCLK1 is an integral multiple of 25, the baud

rate will be more accurate.

18.4.9. Rise time register (I2C_RT)

Address offset: 0x20

Reset value: 0x0000 0002

This register can be accessed by half-word (16-bit) or word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										RISETIME[5:0]					
										rw					

Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5:0	RISETIME[5:0]	Maximum rise time in master mode The RISETIME value should be the maximum SCL rise time incremented by 1.

18.4.10. SAM control and status register (I2C_SAMCS)

Address offset: 0x80

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFR	RFF	TFR	TFF	Reserved		RXF	TXF	RFRIE	RFFIE	TFRIE	TFFIE	Reserved		STOEN	SAMEN
rc_w0	rc_w0	rc_w0	rc_w0			r	r	rw	rw	rw	rw			rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	RFR	Rxframe rise flag, cleared by software by writing 0
14	RFF	Rxframe fall flag, cleared by software by writing 0
13	TFR	Txframe rise flag, cleared by software by writing 0
12	TFF	Txframe fall flag, cleared by software by writing 0
11:10	Reserved	Must be kept at reset value.

9	RXF	Level of rxframe signal
8	TXF	Level of txframe signal
7	RFRIE	Rxframe rise interrupt enable 0: Rxframe rise interrupt disabled 1: Rxframe rise interrupt enabled
6	RFFIE	Rxframe fall interrupt enable 0: Rxframe fall interrupt disabled 1: Rxframe fall interrupt enabled
5	TFRIE	Txframe rise interrupt enable 0: Txframe rise interrupt disabled 1: Txframe rise interrupt enabled
4	TFFIE	Txframe fall interrupt enable 0: Txframe fall interrupt disabled 1: Txframe fall interrupt enabled
3:2	Reserved	Must be kept at reset value.
1	STOEN	SAM_V interface timeout detect enable 0: SAM_V interface timeout detect disabled 1: SAM_V interface timeout detect enabled
0	SAMEN	SAM_V interface enable 0: SAM_V interface disabled 1: SAM_V interface enabled

18.4.11. Fast-mode-plus configure register(I2C_FMPCFG)

Address offset: 0x90

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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

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

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	FMPEN	Fast mode plus enable. The I2C device supports up to 1MHz when this bit is set. 0: Fast mode plus disabled

1: Fast mode plus enabled

19. Serial peripheral interface/Inter-IC sound (SPI/I2S)

19.1. Overview

The SPI/I2S module can communicate with external devices using the SPI protocol or the I2S audio protocol.

The serial peripheral interface (SPI) provides a SPI protocol of data transmission and reception function in master or slave mode. Both full-duplex and simplex communication modes are supported, with hardware CRC calculation and checking. Quad-SPI master mode is only supported in SPI0.

The inter-IC sound (I2S) supports four audio standards: I2S Philips standard, MSB justified standard, LSB justified standard, and PCM standard. I2S works at either master or slave mode for transmission and reception.

19.2. Characteristics

19.2.1. SPI characteristics

- Master or slave operation with full-duplex or half-duplex or simplex mode.
- Separate transmit and receive buffer, 16 bits wide.
- Data frame size can be 8 or 16 bits.
- Bit order can be LSB or MSB.
- Software and hardware NSS management.
- Hardware CRC calculation, transmission and checking.
- Transmission and reception using DMA.
- SPI TI mode supported.
- SPI NSS pulse mode supported.
- Quad-SPI configuration available in master mode (only in SPI0).

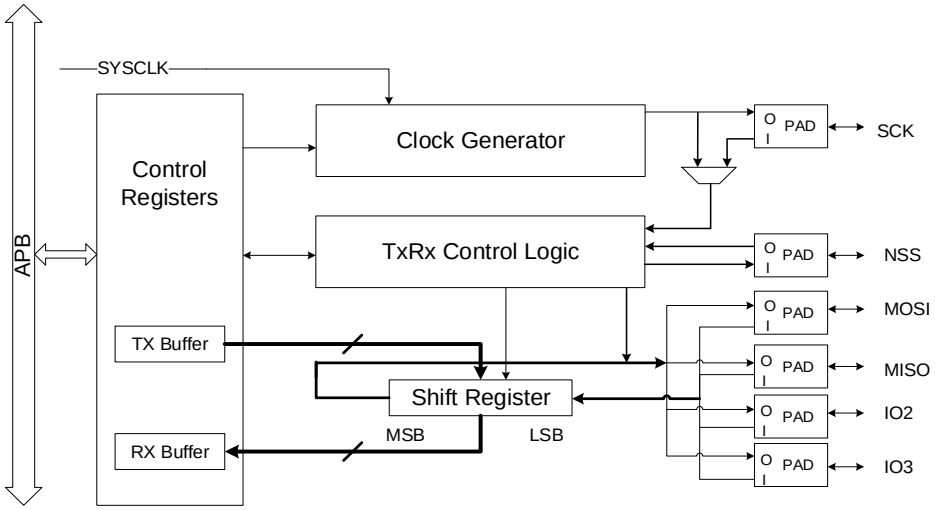
19.2.2. I2S characteristics

- Master or slave operation for transmission/reception.
- Four I2S standards supported: Philips, MSB justified, LSB justified and PCM standard.
- Data length can be 16 bits, 24 bits or 32 bits.
- Channel length can be 16 bits or 32 bits.
- Transmission and reception using a 16 bits wide buffer.
- Audio sample frequency can be 8 kHz to 192 kHz using I2S clock divider.
- Programmable idle state clock polarity.
- Master clock (MCK) can be output.
- Transmission and reception using DMA.

19.3. SPI function overview

19.3.1. SPI block diagram

Figure 19-1. Block diagram of SPI



19.3.2. SPI signal description

Normal configuration (Not Quad-SPI Mode)

Table 19-1. SPI signal description

Pin name	Direction	Description
SCK	I/O	Master: SPI clock output Slave: SPI clock input
MISO	I/O	Master: Data reception line Slave: Data transmission line Master with bidirectional mode: Not used Slave with bidirectional mode: Data transmission and reception line.
MOSI	I/O	Master: Data transmission line Slave: Data reception line Master with bidirectional mode: Data transmission and reception line. Slave with bidirectional mode: Not used
NSS	I/O	Software NSS mode: not used Master in hardware NSS mode: when NSSDRV=1, it is NSS output, suitable for single master application; when NSSDRV=0, it is NSS input, suitable for multi-master

Pin name	Direction	Description
		application. Slave in hardware NSS mode: NSS input, as a chip select signal for slave.

Quad-SPI configuration

SPI is in single wire mode by default and enters into Quad-SPI mode after QMOD bit in SPI_QCTL register is set (only available in SPI0). Quad-SPI mode can only work in master mode.

The SPI is connected to external devices through 6 pins in Quad-SPI mode:

Table 19-2. Quad-SPI signal description

Pin name	Direction	Description
SCK	O	SPI clock output
MOSI	I/O	Transmission/Reception data 0
MISO	I/O	Transmission/Reception data 1
IO2	I/O	Transmission/Reception data 2
IO3	I/O	Transmission/Reception data 3
NSS	O	NSS output

19.3.3. SPI clock timing and data format

CKPL and CKPH bits in SPI_CTL0 register decide the timing of SPI clock and data signal. The CKPL bit decides the SCK level when SPI is in idle state and CKPH bit decides either first or second clock edge is a valid sampling edge. These bits take no effect in TI mode.

Figure 19-2. SPI timing diagram in normal mode

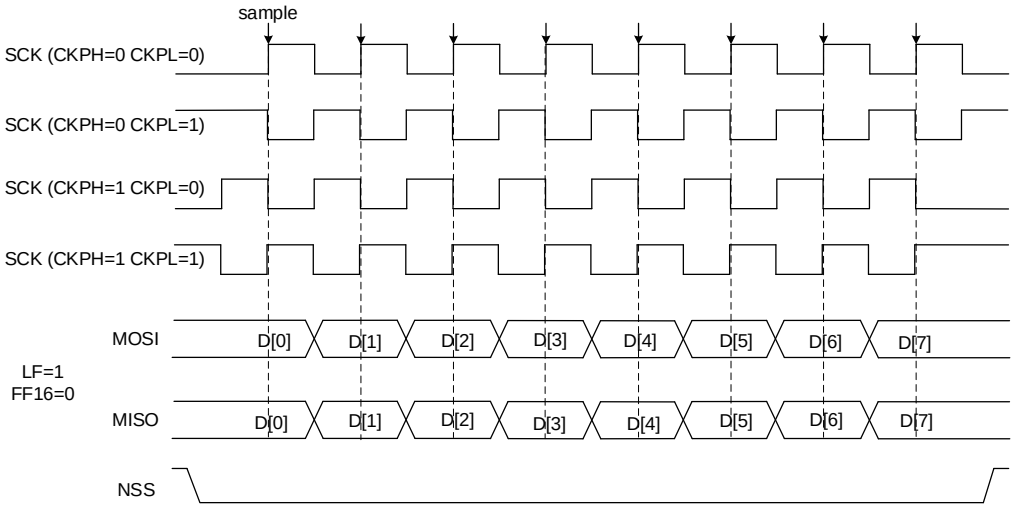
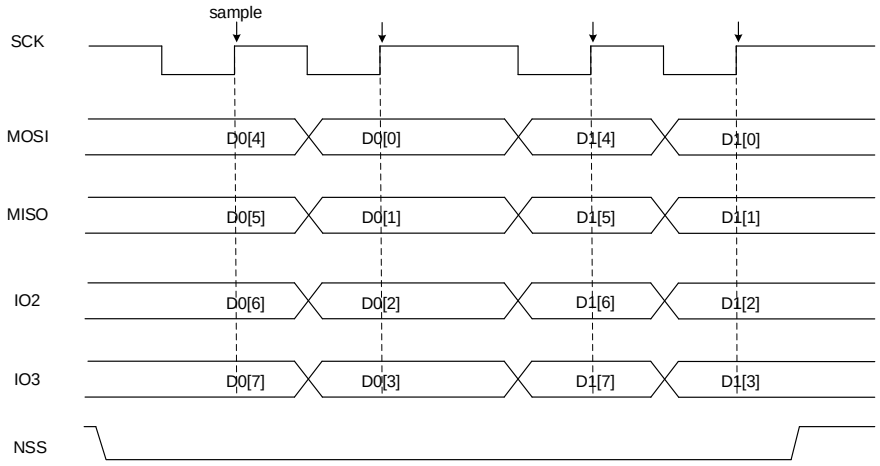


Figure 19-3. SPI timing diagram in Quad-SPI mode (CKPL=1, CKPH=1, LF=0)



In normal mode, the length of data is configured by the FF16 bit in the SPI_CTL0 register. Data length is 16 bits if FF16=1, otherwise is 8 bits. The data frame length is fixed to 8 bits in Quad-SPI mode.

Data order is configured by the LF bit in SPI_CTL0 register, and SPI will first send the LSB first if LF=1, or the MSB first if LF=0. The data order is fixed to MSB first in TI mode.

19.3.4. NSS function

Slave mode

When slave mode is configured (MSTMOD=0), SPI gets NSS level from NSS pin in hardware NSS mode (SWNSSEN = 0) or from SWNSS bit in software NSS mode (SWNSSEN = 1), and SPI transmits/receives data only when NSS level is low. In software NSS mode, NSS pin is not used.

Table 19-3. NSS function in slave mode

Mode	Register configuration	Description
Slave hardware NSS mode	MSTMOD = 0 SWNSSEN = 0	SPI slave gets NSS level from NSS pin.
Slave software NSS mode	MSTMOD = 0 SWNSSEN = 1	SPI slave NSS level is determined by the SWNSS bit. SWNSS = 0: NSS level is low SWNSS = 1: NSS level is high

Master mode

In master mode (MSTMOD=1), if the application uses multi-master connection, NSS can be configured to hardware input mode (SWNSSEN=0, NSSDRV=0) or software mode (SWNSSEN=1). Then, once the NSS pin (in hardware NSS mode) or the SWNSS bit (in software NSS mode) goes low, the SPI automatically enters slave mode and triggers a master fault flag CONFERR.

If the application wants to use NSS line to control the SPI slave, NSS should be configured to hardware output mode (SWNSSEN=0, NSSDRV=1). NSS goes low after SPI is enabled.

The application may also use a general purpose IO as NSS pin to realize more flexible NSS.

Table 19-4. NSS function in master mode

Mode	Register configuration	Description
Master hardware NSS output mode	MSTMOD = 1 SWNSSEN = 0 NSSDRV=1	Applicable to single-master mode. The master uses the NSS pin to control the SPI slave device. At this time, the NSS is configured as the hardware output mode. NSS goes low after enabling SPI.
Master hardware NSS input mode	MSTMOD = 1 SWNSSEN = 0 NSSDRV=0	Applicable to multi-master mode. At this time, NSS is configured as hardware input mode. Once the NSS pin is pulled low, SPI will automatically enter slave mode, and a master configuration error will occur and the CONFERR bit will be set to 1.
Master software NSS mode	MSTMOD = 1 SWNSSEN = 1 SWNSS = 0 NSSDRV: Don't care	Applicable to multi-master mode. Once SWNSS = 0, SPI will automatically enter slave mode, and a master configuration error will occur and the CONFERR bit will be 1.
	MSTMOD = 1 SWNSSEN = 1 SWNSS = 1 NSSDRV: Don't care	The slave can use hardware or software NSS mode.

19.3.5. SPI operating modes

Table 19-5. SPI operating modes

Mode	Description	Register configuration	Data pin usage
MFD	Master full-duplex	MSTMOD = 1 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Transmission MISO: Reception
MTU	Master transmission with unidirectional connection	MSTMOD = 1 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Transmission MISO: Not used
MRU	Master reception with unidirectional connection	MSTMOD = 1 RO = 1	MOSI: Not used MISO: Reception

Mode	Description	Register configuration	Data pin usage
		BDEN = 0 BDOEN: Don't care	
MTB	Master transmission with bidirectional connection	MSTMOD = 1 RO = 0 BDEN = 1 BDOEN = 1	MOSI: Transmission MISO: Not used
MRB	Master reception with bidirectional connection	MSTMOD = 1 RO = 0 BDEN = 1 BDOEN = 0	MOSI: Reception MISO: Not used
SFD	Slave full-duplex	MSTMOD = 0 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Reception MISO: Transmission
STU	Slave transmission with unidirectional connection	MSTMOD = 0 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Not used MISO: Transmission
SRU	Slave reception with unidirectional connection	MSTMOD = 0 RO = 1 BDEN = 0 BDOEN: Don't care	MOSI: Reception MISO: Not used
STB	Slave transmission with bidirectional connection	MSTMOD = 0 RO = 0 BDEN = 1 BDOEN = 1	MOSI: Not used MISO: Transmission
SRB	Slave reception with bidirectional connection	MSTMOD = 0 RO = 0 BDEN = 1 BDOEN = 0	MOSI: Not used MISO: Reception

Figure 19-4. A typical full-duplex connection

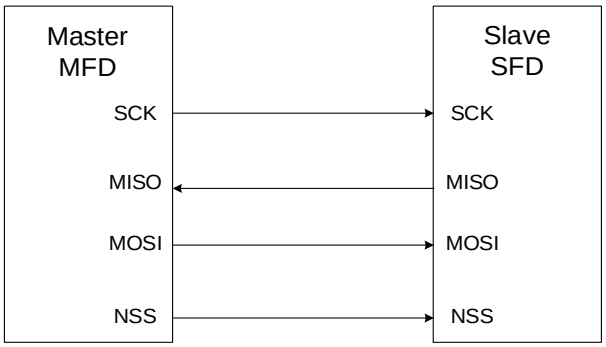


Figure 19-5. A typical simplex connection (Master: Receive, Slave: Transmit)

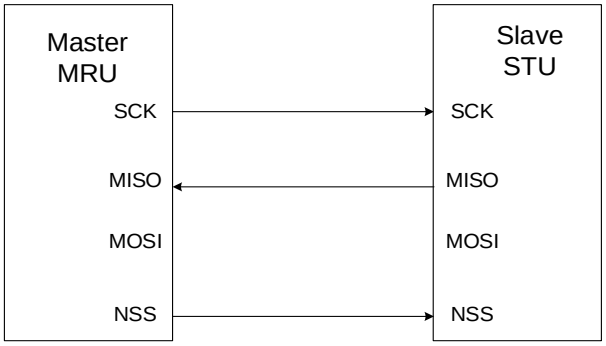


Figure 19-6. A typical simplex connection (Master: Transmit only, Slave: Receive)

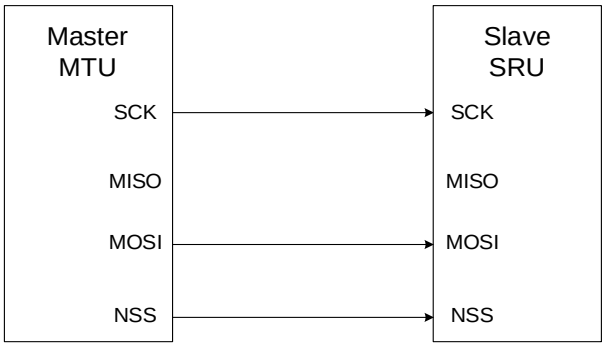
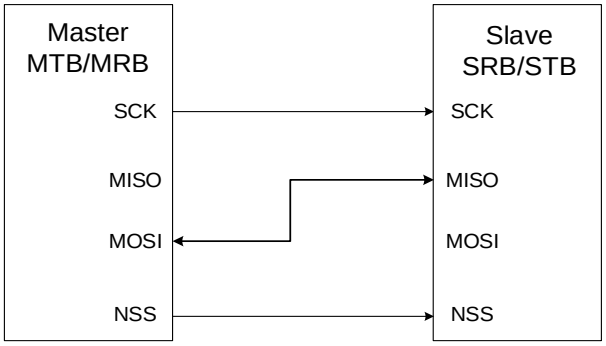


Figure 19-7. A typical bidirectional connection



SPI initialization sequence

Before transmiting or receiving data, application should follow the SPI initialization sequence described below:

1. If master mode or slave TI mode is used, program the PSC [2:0] bits in SPI_CTL0 register to generate SCK with desired baud rate or configure the Td time in TI mode, otherwise, ignore this step.
2. Program data format (FF16 bit in the SPI_CTL0 register).
3. Program the clock timing register (CKPL and CKPH bits in the SPI_CTL0 register).
4. Program the frame format (LF bit in the SPI_CTL0 register).
5. Program the NSS mode (SWNSSEN and NSSDRV bits in the SPI_CTL0 register) according to the application's demand as described above in ***NSS function*** section.

6. If TI mode is used, set TMOD bit in SPI_CTL1 register, otherwise, ignore this step.
7. If NSSP mode is used, set NSSP bit in SPI_CTL1 register, otherwise, ignore this step.
8. Configure MSTMOD, RO, BDEN and BDOEN depending on the operating modes described in ***SPI operating modes*** section.
9. If Quad-SPI mode is used, set the QMOD bit in SPI_QCTL register. Ignore this step if Quad-SPI mode is not used.
10. Enable the SPI (set the SPIEN bit).

Note: During communication, CKPH, CKPL, MSTMOD, PSC[2:0] and LF bits should not be changed.

SPI basic transmission and reception sequence

Transmission sequence

After the initialization sequence, the SPI is enabled and stays at idle state. In master mode, the transmission starts when the application writes a data into the transmit buffer. In slave mode the transmission starts when SCK clock signal at SCK pin begins to toggle and NSS level is low, so application should ensure that data is already written into transmit buffer before the transmission starts in slave mode.

When SPI begins to send a data frame, it first loads this data frame from the data buffer to the shift register and then begins to transmit the loaded data frame, TBE (transmit buffer empty) flag is set after the first bit of this frame is transmitted. After TBE flag is set, which means the transmit buffer is empty, the application should write SPI_DATA register again if it has more data to transmit.

In master mode, software should write the next data into SPI_DATA register before the transmission of current data frame is completed if it desires to generate continuous transmission.

Reception sequence

After the last valid sample clock, the incoming data will be moved from shift register to the receive buffer and RBNE (receive buffer not empty) will be set. The application should read SPI_DATA register to get the received data and this will clear the RBNE flag automatically. In MRU and MRB modes, hardware continuously sends clock signal to receive the next data frame, while in full-duplex master mode (MFD), hardware only receives the next data frame when the transmit buffer is not empty.

SPI operation sequence in different modes (Not Quad-SPI, TI mode or NSSP mode)

In full-duplex mode, either MFD or SFD, the RBNE and TBE flags should be monitored and then follow the sequences described above.

The transmission mode (MTU, MTB, STU or STB) is similar to the transmission sequence of full-duplex mode except that the RBNE bit and RXORERR bit need to be ignored.

The master reception mode (MRU or MRB) is different from the reception sequence of full-duplex mode. In MRU or MRB mode, after SPI is enabled, the SPI continuously generates SCK until the SPI is disabled. So the application should ignore the TBE flag and read out reception buffer in time after the RBNE flag is set, otherwise a data overrun fault will occur.

The slave reception mode (SRU or SRB) is similar to the reception sequence of full-duplex mode except that the TBE bit need to be ignored.

SPI TI mode

SPI TI mode takes NSS as a special frame header flag signal and its operation sequence is similar to normal mode described above. The modes described above (MFD, MTU, MRU, MTB, MRB, SFD, STU, SRU, STB and SRB) are still supported in TI mode. While, in TI mode the CKPL and CKPH bits in SPI_CTL0 registers take no effect and the SCK sample edge is falling edge.

Figure 19-8. Timing diagram of TI master mode with discontinuous transfer

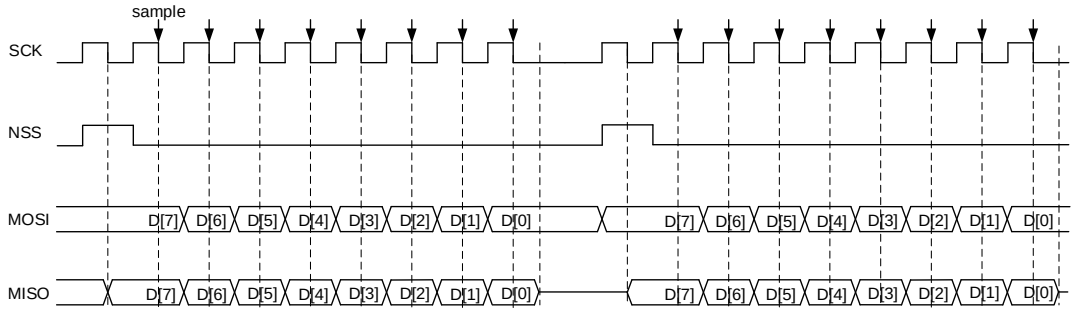
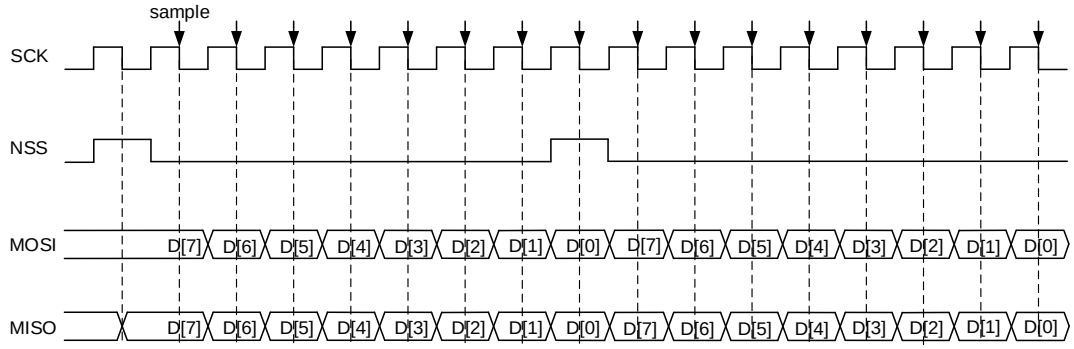
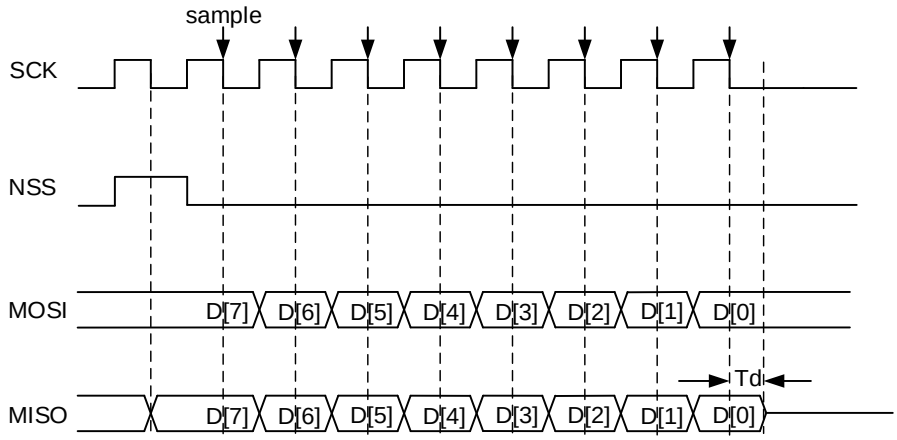


Figure 19-9. Timing diagram of TI master mode with continuous transfer



In master TI mode, SPI can perform continuous or non-continuous transfer. If the master writes SPI_DATA register fast enough, the transfer is continuous, otherwise non-continuous. In non-continuous transfer, there is an extra header clock cycle before each byte. While in continuous transfer, the extra header clock cycle only exists before the first byte and the following bytes' header clock is overlaid at the last bit of pervious bytes.

Figure 19-10. Timing diagram of TI slave mode



In slave TI mode, after the last rising edge of SCK in transfer, the slave begins to transmit the LSB bit of the last data byte, and after a half-bit time, the master begins to sample the line. To make sure that the master samples the right value, the slave should continue to drive this bit after the falling sample edge of SCK for a period of time before releasing the pin. This time is called T_d. T_d is decided by PSC[2:0] bits in SPI_CTL0 register.

$$T_d = \frac{T_{bit}}{2} + 5 * T_{pclk} \quad (19\text{-}1)$$

For example, if PSC[2:0] = 010, T_d is $9*T_{pclk}$.

In slave mode, the slave also monitors the NSS signal and sets an error flag FERR if it detects an incorrect NSS behavior, for example, toggles at the middle bit of a byte.

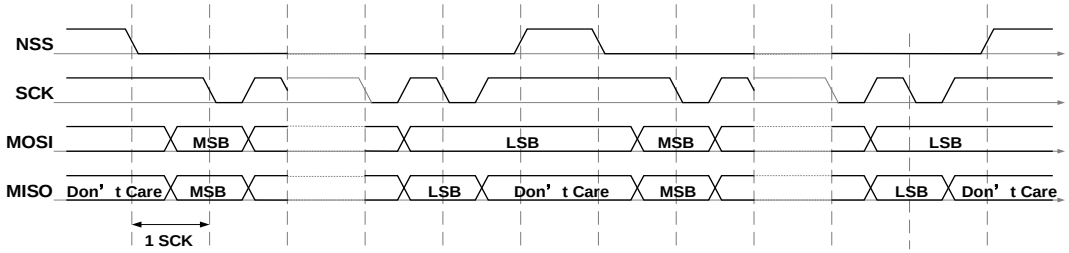
NSS pulse mode operation sequence

This function is controlled by NSSP bit in SPI_CTL1 register. In order to implement this function, several additional conditions must be met: configure the device to master mode, frame format should follow the normal SPI protocol, select the first clock transition as the data capture edge.

In summary, MSTMOD = 1, NSSP = 1, CKPH = 0.

When NSS pulse mode is enabled, a pulse duration of at least 1 SCK clock period is inserted between two successive data frames depending on the status of internal data transmit buffer. Multiple SCK clock cycle intervals are possible if the transfer buffer stays empty. This function is designed for single master-slave configuration for the slave to latch data. The following diagram depicts its timing diagram.

Figure 19-11. Timing diagram of NSS pulse with continuous transmission



Quad-SPI mode operation sequence

The Quad-SPI mode is designed to control Quad-SPI flash.

In order to enter Quad-SPI mode, the software should first verify that the TBE bit is set and TRANS bit is cleared, then set QMOD bit in SPI_QCTL register. In Quad-SPI mode, BDEN, BDOEN, CRCEN, CRCNT, FF16, RO and LF bits in SPI_CTL0 register should be kept cleared and MSTMOD should be set to ensure that SPI is in master mode. SPIEN, PSC, CKPL and CKPH bits should be configured as desired.

There are two operation modes in Quad-SPI mode: quad write and quad read, decided by QRD bit in SPI_QCTL register.

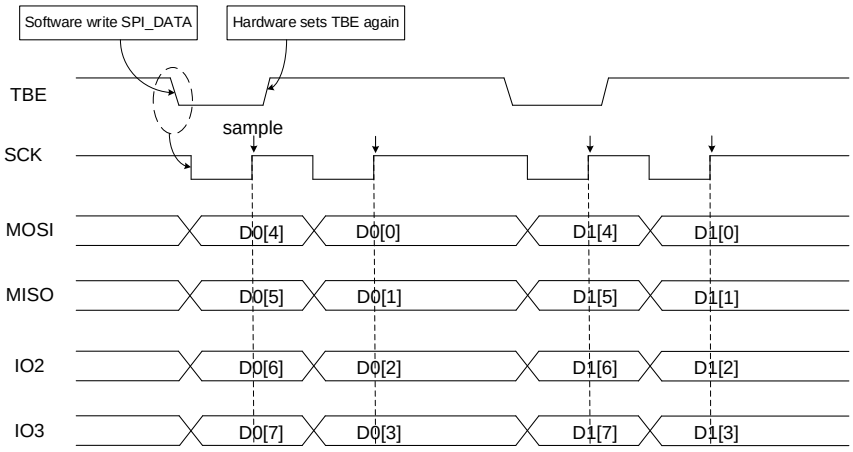
Quad write operation

SPI works in quad write mode when QMOD is set and QRD is cleared in SPI_QCTL register. In this mode, MOSI, MISO, IO2 and IO3 are all used as output pins. SPI begins to generate clock on SCK line and transmit data on MOSI, MISO, IO2 and IO3 as soon as data is written into SPI_DATA (TBE is cleared) and SPIEN is set. Once SPI starts transmission, it always checks TBE status at the end of a frame and stops when condition is not met.

The operation flow for transmitting in quad mode is shown below:

1. Configure clock prescaler, clock polarity, phase, etc. in SPI_CTL0 and SPI_CTL1 based on application requirements.
2. Set QMOD bit in SPI_QCTL register and then enable SPI by setting SPIEN in SPI_CTL0.
3. Write a byte of data to SPI_DATA register and the TBE will be cleared.
4. Wait until TBE is set by hardware again before writing the next byte.

Figure 19-12. Timing diagram of quad write operation in Quad-SPI mode



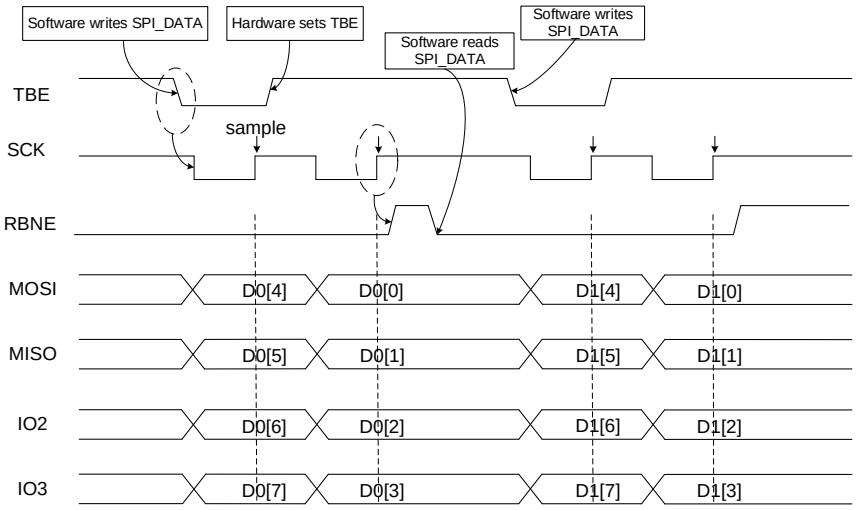
Quad read operation

SPI works in quad read mode when QMOD and QRD bits are both set in SPI_QCTL register. In this mode, MOSI, MISO, IO2 and IO3 are all used as input pins. SPI begins to generate clock on SCK line as soon as a data is written into SPI_DATA (TBE is cleared) and SPIEN is set. Writing data into SPI_DATA is only to generate SCK clocks, so the written data can be any value. Once SPI starts transmission, it always checks SPIEN and TBE status at the end of a frame and stops when condition is not met. So, dummy data should always be written into SPI_DATA to generate SCK.

The operation flow for receiving in quad mode is shown below:

1. Configure clock prescaler, clock polarity, phase, etc. in SPI_CTL0 and SPI_CTL1 register based on application requirements.
2. Set QMOD and QRD bits in SPI_QCTL register and then enable SPI by setting SPIEN in SPI_CTL0 register.
3. Write an arbitrary byte (for example, 0xFF) to SPI_DATA register.
4. Wait until the RBNE flag is set and read SPI_DATA to get the received byte.
5. Write an arbitrary byte (for example, 0xFF) to SPI_DATA to receive the next byte.

Figure 19-13. Timing diagram of quad read operation in Quad-SPI mode



SPI disabling sequence

Different sequences are used to disable the SPI in different operation modes.

MFD SFD

Wait for the last RBNE flag and then receive the last data. Confirm that TBE=1 and TRANS=0. At last, disable the SPI by clearing SPIEN bit.

MTU MTB STU STB

Write the last data into SPI_DATA and wait until the TBE flag is set and then wait until the TRANS flag is cleared. Disable the SPI by clearing SPIEN bit.

MRU MRB

After getting the second last RBNE flag, read out this data and delay for a SCK clock time and then, disable the SPI by clearing SPIEN bit. Wait until the last RBNE flag is set and read out the last data.

SRU SRB

Application can disable the SPI when it doesn't want to receive data, and then wait until the TRANS=0 to ensure the ongoing transfer completes.

TI mode

The disabling sequence of TI mode is the same as the sequences described above.

NSS pulse mode

The disabling sequence of NSSP mode is the same as the sequences described above.

Quad-SPI mode

Before leaving quad wire mode or disabling SPI, software should first check that TBE bit is set and TRANS bit is cleared, then the QMOD bit in SPI_QCTL register and SPIEN bit in SPI_CTL0 register are cleared.

19.3.6. DMA function

The DMA frees the application from data writing and reading process during transfer, to improve the system efficiency.

DMA function in SPI is enabled by setting DMATEN and DMAREN bits in SPI_CTL1 register. To use DMA function, application should first correctly configure DMA modules, then configure SPI module according to the initialization sequence, at last enable SPI.

After being enabled, if DMATEN is set, SPI will generate a DMA request each time when TBE=1, then DMA will acknowledge to this request and write data into the SPI_DATA register automatically. If DMAREN is set, SPI will generate a DMA request each time when RBNE=1, then DMA will acknowledge to this request and read data from the SPI_DATA register automatically.

19.3.7. CRC function

There are two CRC calculators in SPI: one for transmission and the other for reception. The CRC calculation uses the polynomial defined in SPI_CRCPOLY register.

Application can enable the CRC function by setting CRCEN bit in SPI_CTL0 register. The CRC calculators continuously calculate CRC for each bit transmitted and received on lines, and the calculated CRC values can be read from SPI_TCRC and SPI_RCRC registers.

To transmit the calculated CRC value, application should set the CRCNT bit in SPI_CTL0 register after the last data is written to the transmit buffer. In full-duplex mode (MFD or SFD), when the SPI transmits a CRC and prepares to check the received CRC value, the SPI treats the incoming data as a CRC value. In reception mode (MRB, MRU, SRU and SRB), the application should set the CRCNT bit after the second last data frame is received. When CRC checking fails, the CRCERR flag will be set.

If DMA function is enabled, application doesn’t need to configure CRCNT bit and hardware will automatically process the CRC transmitting and checking.

Note: When SPI is in slave mode and CRC function is enable, the CRC calculator is sensitive to input SCK clock whether SPI is enable or not. The software must enable CRC only when the clock is stable to avoid wrong CRC calculation. And when SPI works as a slave, the NSS internal signal needs to be kept low between the data phase and CRC phase.

When SPI is configured as slave mode and the CRC function is used, the CRC calculation will still be performed even when the NSS pin is high (when the NSS signal is high, as long as there is a clock pulse on the SCK pin, the CRC calculation will continue). This situation

occurs when the master device alternately communicates with multiple slave devices. At this time, it is recommended to restart the CRC function when the NSS signal is low. When the slave device is not selected (NSS signal is high) and switches to being selected as a new slave device (NSS signal is low), in order to maintain the synchronization of the next CRC calculation results between the master and slave devices, the CRC values at both ends should be cleared. It is recommended to clear the CRC value according to the following steps:

1. Disable the SPI module (SPIEN=0);
2. Clear the CRCEN bit (CRCEN=0);
3. Set the CRCEN bit (CRCEN=1);
4. Enable the SPI module (SPIEN=1).

19.3.8. SPI interrupts

Status flags

- Transmit buffer empty flag (TBE)

This bit is set when the transmit buffer is empty, the software can write the next data to the transmit buffer by writing the SPI_DATA register.

- Receive buffer not empty flag (RBNE)

This bit is set when receive buffer is not empty, which means that one data is received and stored in the receive buffer, and software can read the data by reading the SPI_DATA register.

- SPI transmitting ongoing flag (TRANS)

TRANS is a status flag to indicate whether the transfer is ongoing or not. It is set and cleared by hardware and not controlled by software. This flag doesn't generate any interrupt.

Error flags

- Configuration fault error (CONFERR)

CONFERR is an error flag in master mode. In NSS hardware mode and if the NSSDRV is not enabled, the CONFERR is set when the NSS pin is pulled low. In NSS software mode, the CONFERR is set when the SWNSS bit is 0. When the CONFERR is set, the SPIEN bit and the MSTMOD bit are cleared by hardware, the SPI is disabled and the device is forced into slave mode.

The SPIEN and MSTMOD bits are write protected until the CONFERR is cleared. The CONFERR bit of the slave cannot be set. In a multi-master configuration, the device can be in slave mode with CONFERR bit set, which means there might have been a multi-master conflict for system control.

- Rx overrun error (RXORERR)

The RXORERR bit is set if a data is received when the RBNE is set. That means, the last data has not been read out and the newly incoming data is received. The receive buffer

contents won't be covered with the newly incoming data, so the newly incoming data is lost.

- Format error (FERR)

In slave TI mode, the slave also monitors the NSS signal and set an error flag if it detects an incorrect NSS behavior, for example: toggles at the middle bit of a byte.

- CRC error (CRCERR)

When the CRCEN bit is set, the CRC calculation result of the received data in the SPI_RCRC register is compared with the received CRC value after the last data, the CRCERR is set when they are different.

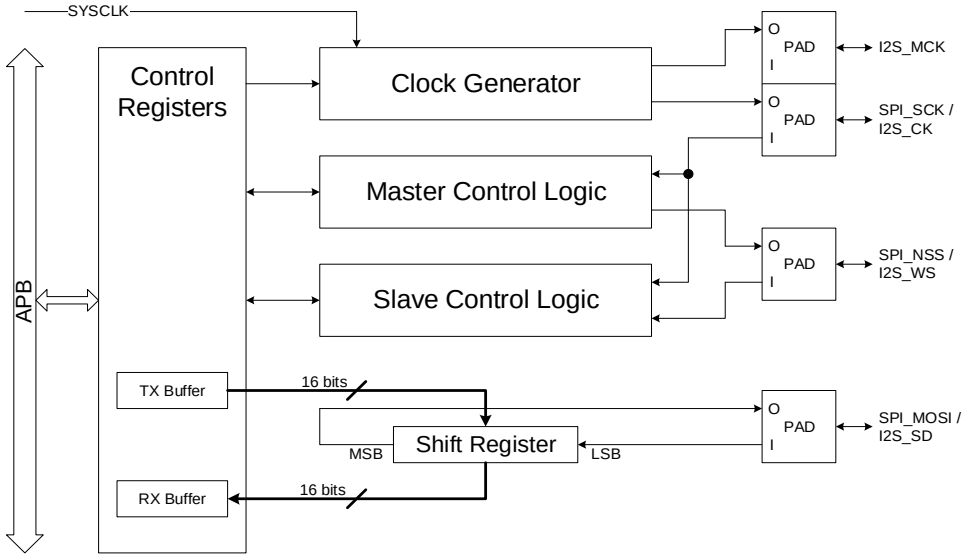
Table 19-6. SPI interrupt requests

Flag	Description	Clear method	Interrupt enable bit
TBE	Transmit buffer empty	Write SPI_DATA register.	TBEIE
RBNE	Receive buffer not empty	Read SPI_DATA register.	RBNEIE
CONFERR	Configuration fault error	Read or write SPI_STAT register, then write SPI_CTL0 register.	ERRIE
RXORERR	Rx overrun error	Read SPI_DATA register, then read SPI_STAT register.	
CRCERR	CRC error	Write 0 to CRCERR bit	
FERR	TI Mode Format Error	Write 0 to FERR bit	

19.4. I2S function overview

19.4.1. I2S block diagram

Figure 19-14. Block diagram of I2S



There are five sub modules to support I2S function, including control registers, clock

generator, master control logic, slave control logic and shift register. All the user configuration registers are implemented in the control registers module, including the TX buffer and RX buffer. The clock generator is used to produce I2S communication clock in master mode. The master control logic is implemented to generate the I2S_WS signal and control the communication in master mode. The slave control logic is implemented to control the communication in slave mode according to the received I2S_CK and I2S_WS. The shift register handles the serial data transmission and reception on I2S_SD.

19.4.2. I2S signal description

There are four pins on the I2S interface, including I2S_CK, I2S_WS, I2S_SD and I2S_MCK. I2S_CK is the serial clock signal, which shares the same pin with SPI_SCK. I2S_WS is the frame control signal, which shares the same pin with SPI_NSS. I2S_SD is the serial data signal, which shares the same pin with SPI_MOSI. I2S_MCK is the master clock signal. It produces a frequency rate equals to 256 x Fs, and Fs is the audio sampling frequency.

19.4.3. I2S audio standards

The I2S audio standard is selected by the I2SSTD bits in the SPI_I2SCTL register. Four audio standards are supported, including I2S Philips standard, MSB justified standard, LSB justified standard, and PCM standard. All standards except PCM handle audio data time-multiplexedly on two channels (the left channel and the right channel). For these standards, the I2S_WS signal indicates the channel side. For PCM standard, the I2S_WS signal indicates frame synchronization information.

The data length and the channel length are configured by the DTLEN bits and CHLEN bit in the SPI_I2SCTL register. Since the channel length must be greater than or equal to the data length, four packet types are available. They are 16-bit data packed in 16-bit frame, 16-bit data packed in 32-bit frame, 24-bit data packed in 32-bit frame, and 32-bit data packed in 32-bit frame. The data buffer for transmission and reception is 16-bit wide. In the case that the data length is 24 bits or 32 bits, two write or read operations to or from the SPI_DATA register are needed to complete the transmission of a frame. In the case that the data length is 16 bits, only one write or read operation to or from the SPI_DATA register is needed to complete the transmission of a frame. When using 16-bit data packed in 32-bit frame, 16-bit 0 is inserted by hardware automatically to extend the data to 32-bit format.

For all standards and packet types, the most significant bit (MSB) is always sent first. For all standards based on two channels time-multiplexed, the channel left is always sent first followed by the channel right.

I2S Philips standard

For I2S Philips standard, I2S_WS and I2S_SD are updated on the falling edge of I2S_CK, and I2S_WS becomes valid one clock before t`he data. The timing diagrams for each configuration are shown below.

Figure 19-15. I2S Philips standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0)

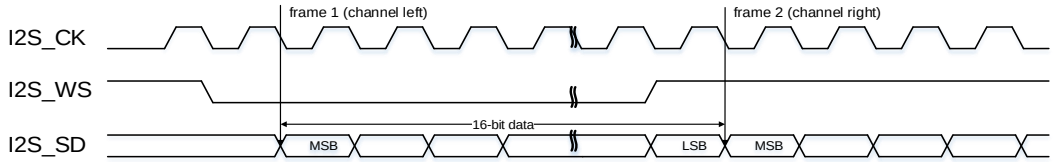
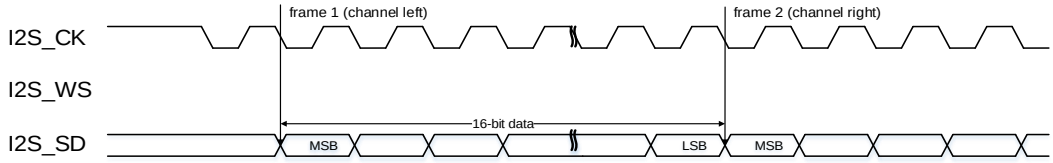


Figure 19-16. I2S Philips standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1)



When the packet type is 16-bit data packed in 16-bit frame, only one write or read operation to or from the SPI_DATA register is needed to complete the transmission of a frame.

Figure 19-17. I2S Philips standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

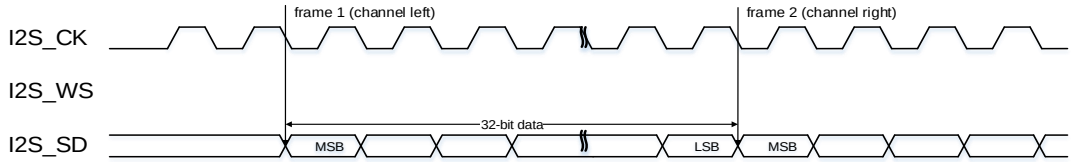
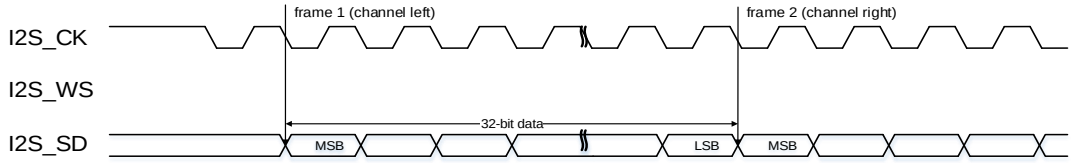


Figure 19-18. I2S Philips standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1)



When the packet type is 32-bit data packed in 32-bit frame, two write or read operations to or from the SPI_DATA register are needed to complete the transmission of a frame. In transmission mode, if a 32-bit data is going to be sent, the first data written to the SPI_DATA register should be the higher 16 bits, and the second one should be the lower 16 bits. In reception mode, if a 32-bit data is received, the first data read from the SPI_DATA register should be the higher 16 bits, and the second one should be the lower 16 bits.

Figure 19-19. I2S Philips standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

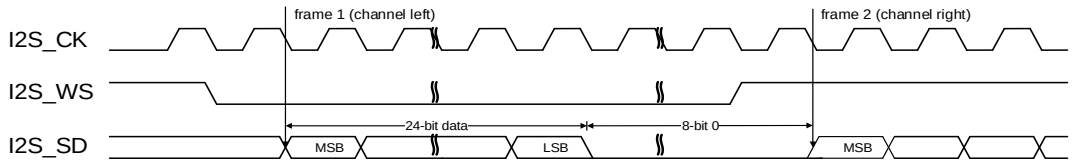
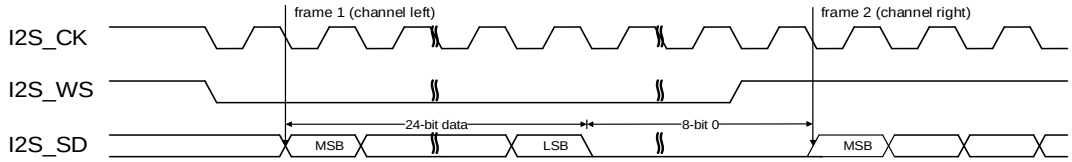


Figure 19-20. I2S Philips standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1)



When the packet type is 24-bit data packed in 32-bit frame, two write or read operations to or from the SPI_DATA register are needed to complete the transmission of a frame. In transmission mode, if a 24-bit data D[23:0] is going to be sent, the first data written to the SPI_DATA register should be the higher 16 bits D[23:8]. And the second one should be a 16-bit data, the higher 8 bits of this 16-bit data should be D[7:0] and the lower 8 bits can be any value. In reception mode, if a 24-bit data D[23:0] is received, the first data read from the SPI_DATA register is D[23:8]. And the second one is a 16-bit data, the higher 8 bits of this 16-bit data are D[7:0] and the lower 8 bits are zeros.

Figure 19-21. I2S Philips standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

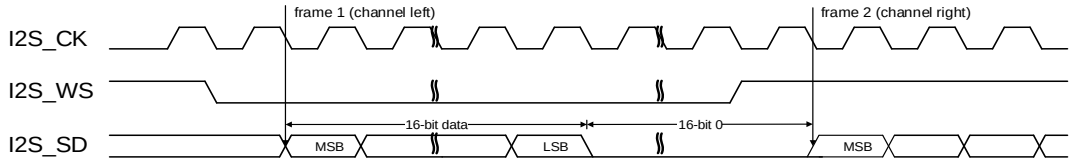
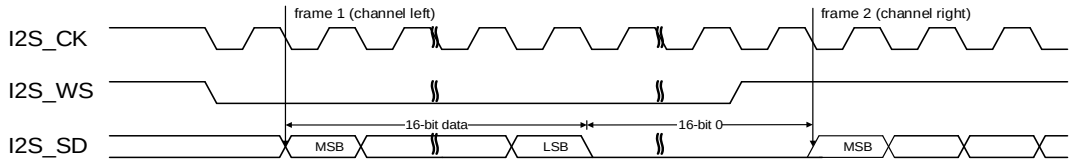


Figure 19-22. I2S Philips standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



When the packet type is 16-bit data packed in 32-bit frame, only one write or read operation to or from the SPI_DATA register is needed to complete the transmission of a frame. The remaining 16 bits are forced by hardware to 0x0000 to extend the data to 32-bit format.

MSB justified standard

For MSB justified standard, I2S_WS and I2S_SD are updated on the falling edge of I2S_CK. The SPI_DATA register is handled in the exactly same way as that for I2S Philips standard. The timing diagrams for each configuration are shown below.

Figure 19-23. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0)

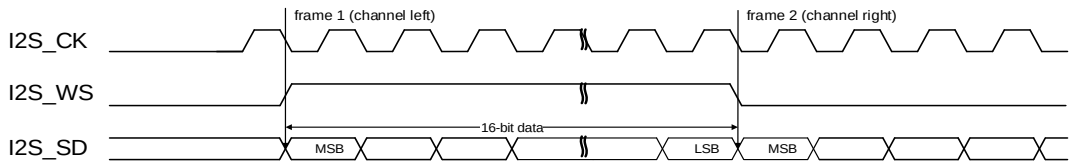


Figure 19-24. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1)

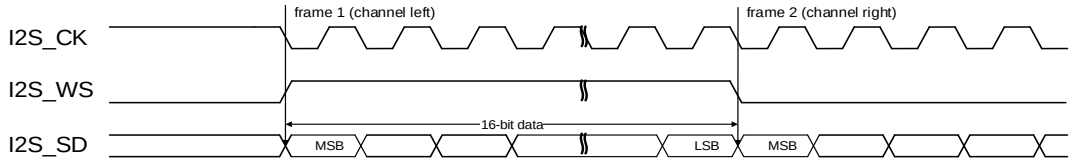


Figure 19-25. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

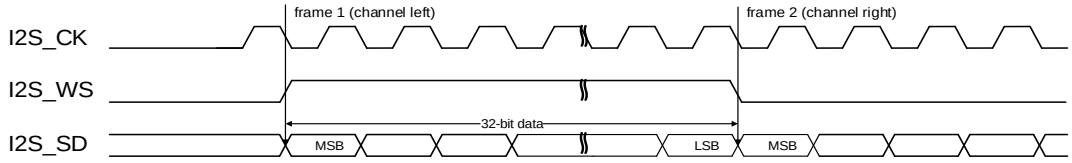


Figure 19-26. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1)

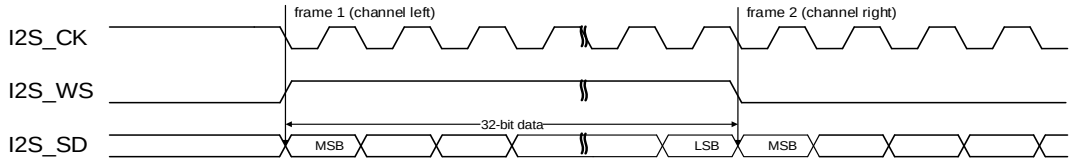


Figure 19-27. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

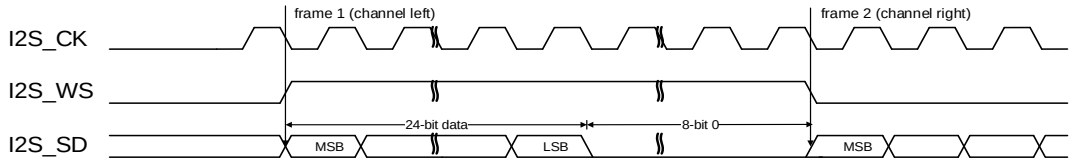


Figure 19-28. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1)

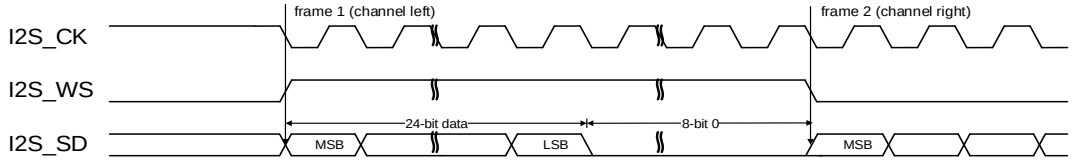


Figure 19-29. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

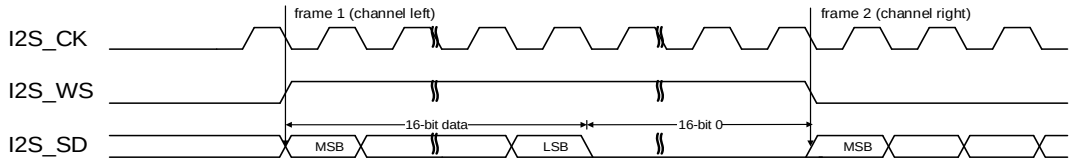
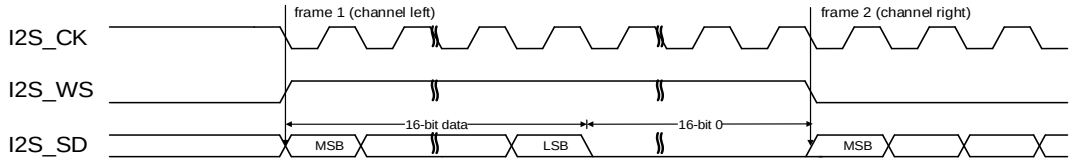


Figure 19-30. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



LSB justified standard

For LSB justified standard, I2S_WS and I2S_SD are updated on the falling edge of I2S_CK. In the case that the channel length is equal to the data length, LSB justified standard and MSB justified standard are exactly the same. In the case that the channel length is greater than the data length, the valid data is aligned to LSB for LSB justified standard while the valid data is aligned to MSB for MSB justified standard. The timing diagrams for the cases that the channel length is greater than the data length are shown below.

Figure 19-31. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

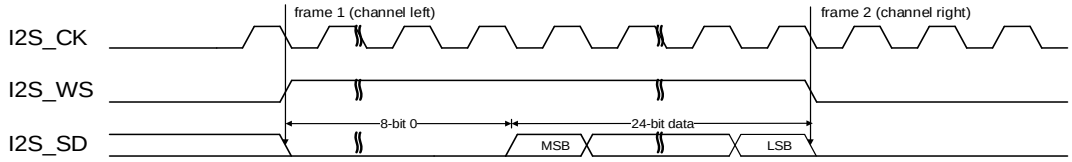
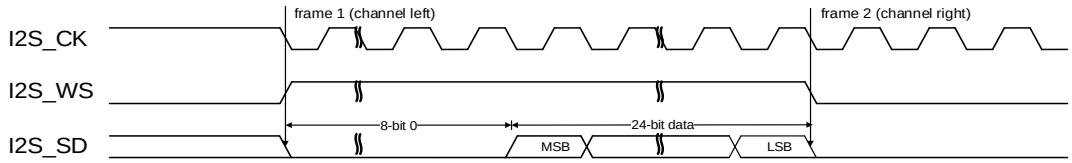


Figure 19-32. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1)



When the packet type is 24-bit data packed in 32-bit frame, two write or read operations to or from the SPI_DATA register are needed to complete the transmission of a frame. In transmission mode, if a 24-bit data D[23:0] is going to be sent, the first data written to the SPI_DATA register should be a 16-bit data. The higher 8 bits of the 16-bit data can be any value and the lower 8 bits should be D[23:16]. The second data written to the SPI_DATA register should be D[15:0]. In reception mode, if a 24-bit data D[23:0] is received, the first data read from the SPI_DATA register is a 16-bit data. The high 8 bits of this 16-bit data are zeros and the lower 8 bits are D[23:16]. The second data read from the SPI_DATA register is D[15:0].

Figure 19-33. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

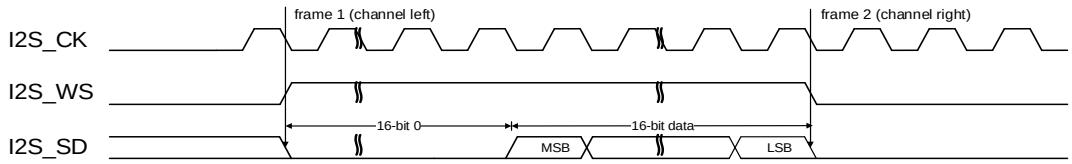
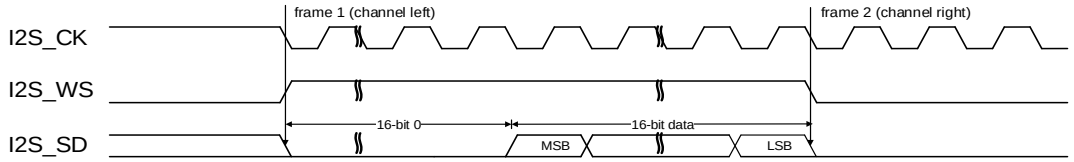


Figure 19-34. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



When the packet type is 16-bit data packed in 32-bit frame, only one write or read operation to or from the SPI_DATA register is needed to complete the transmission of a frame. The remaining 16 bits are forced by hardware to 0x0000 to extend the data to 32-bit format.

PCM standard

For PCM standard, I2S_WS and I2S_SD are updated on the rising edge of I2S_CK, and the I2S_WS signal indicates frame synchronization information. Both the short frame synchronization mode and the long frame synchronization mode are available and configurable using the PCMSMOD bit in the SPI_I2SCTL register. The SPI_DATA register is handled in the exactly same way as that for I2S Philips standard. The timing diagrams for each configuration of the short frame synchronization mode are shown below.

Figure 19-35. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=0)

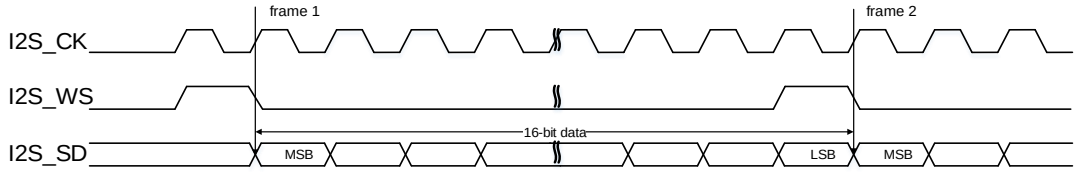


Figure 19-36. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1)

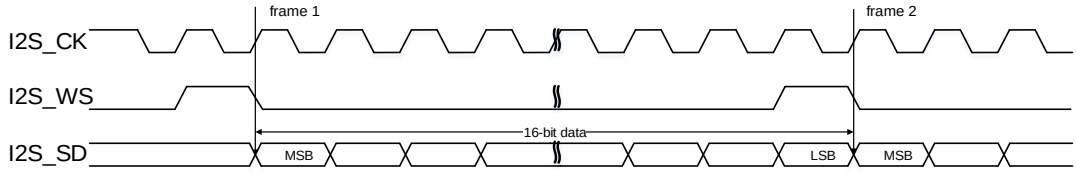


Figure 19-37. PCM standard short frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

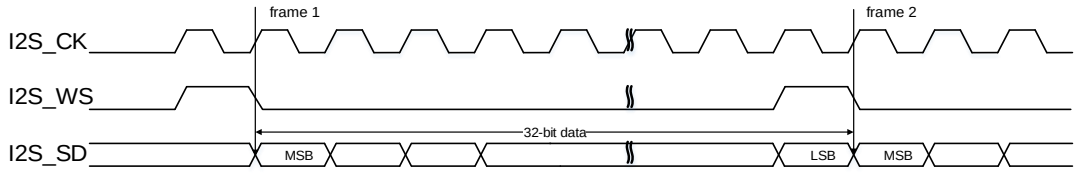


Figure 19-38. PCM standard short frame synchronization mode timing diagram

(DTLEN=10, CHLEN=1, CKPL=1)

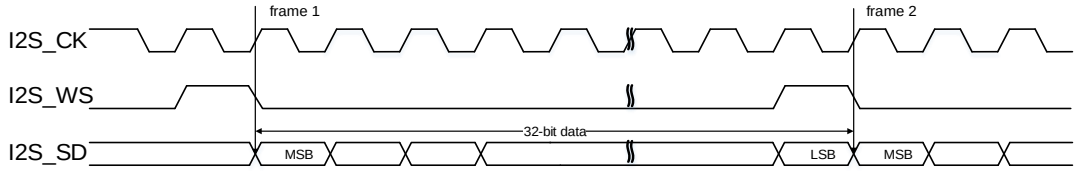


Figure 19-39. PCM standard short frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

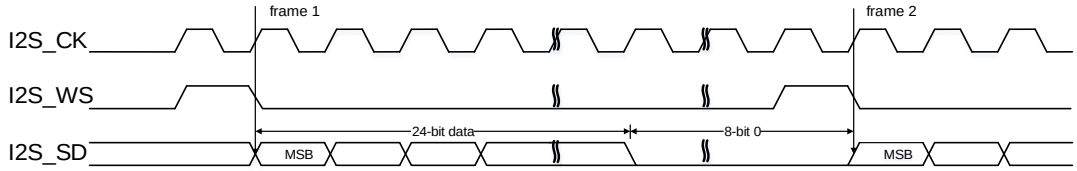


Figure 19-40. PCM standard short frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=1)

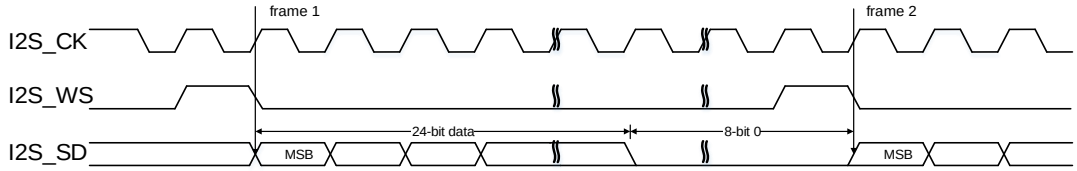


Figure 19-41. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

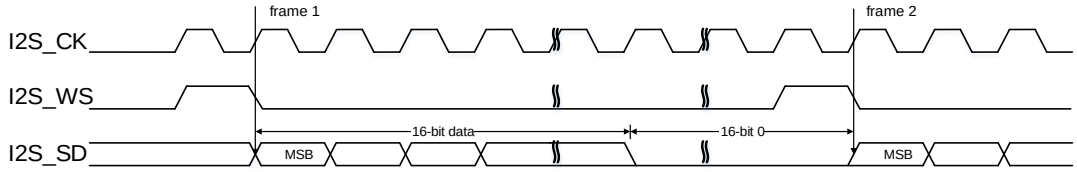
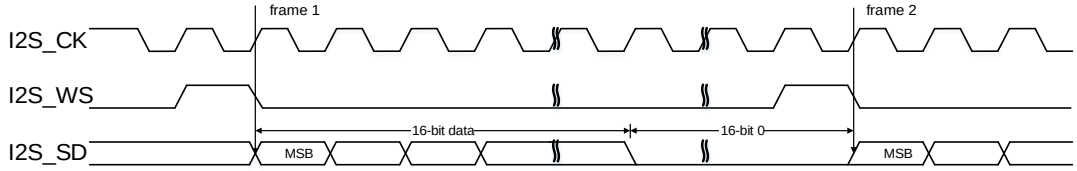


Figure 19-42. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



The timing diagrams for each configuration of the long frame synchronization mode are shown below.

Figure 19-43. PCM standard long frame synchronization mode timing diagram

(DTLEN=00, CHLEN=0, CKPL=0)

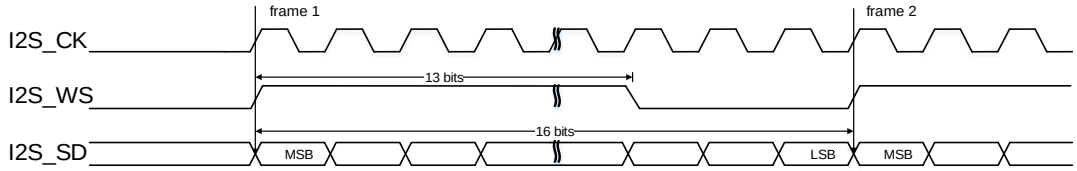


Figure 19-44. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1)

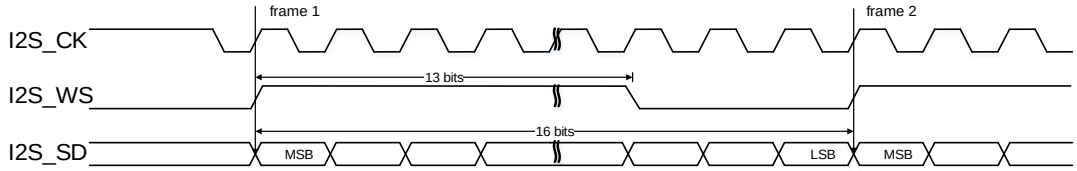


Figure 19-45. PCM standard long frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

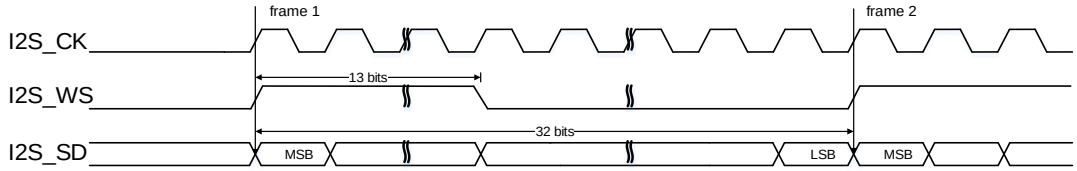


Figure 19-46. PCM standard long frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=1)

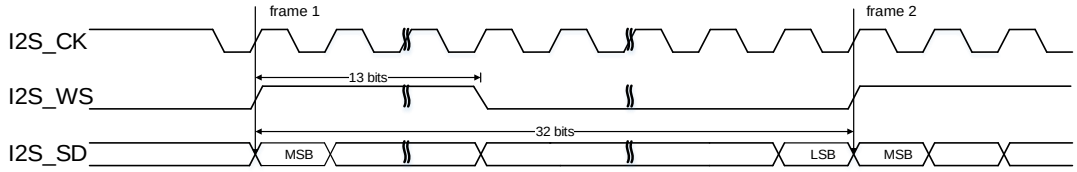


Figure 19-47. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

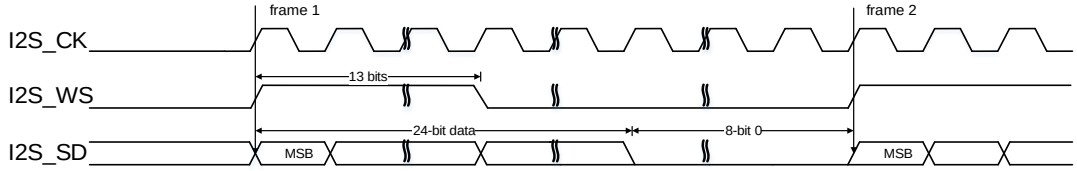


Figure 19-48. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=1)

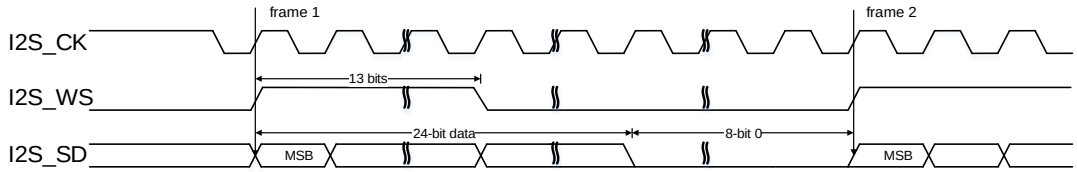


Figure 19-49. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

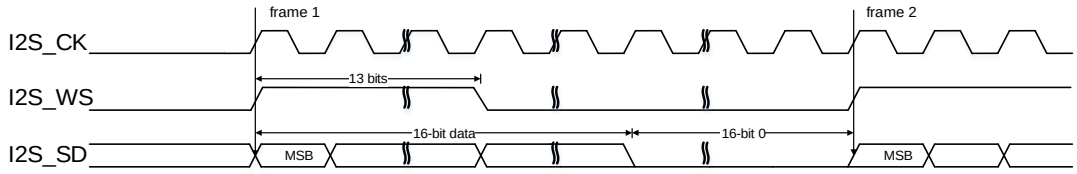
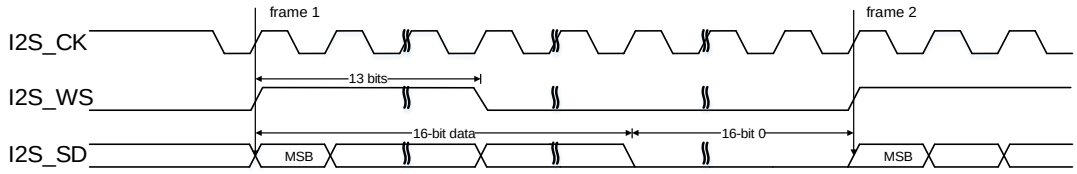
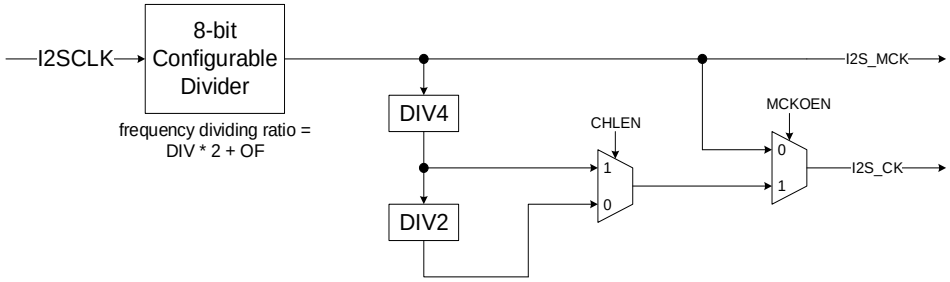


Figure 19-50. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



19.4.4. I2S clock

Figure 19-51. Block diagram of I2S clock generator



The block diagram of I2S clock generator is shown as ***Figure 19-51. Block diagram of I2S clock generator***. The I2S interface clocks are configured by the DIV bits, the OF bit, the MCKOEN bit in the SPI_I2SPSC register and the CHLEN bit in the SPI_I2SCTL register. The source clock is the system clock(CK_SYS). The I2S bitrate can be calculated by the formulas shown in ***Table 19-7. I2S bitrate calculation formulas***.

Table 19-7. I2S bitrate calculation formulas

MCKOEN	CHLEN	Formula
0	0	I2SCLK / (DIV * 2 + OF)
0	1	I2SCLK / (DIV * 2 + OF)
1	0	I2SCLK / (8 * (DIV * 2 + OF))
1	1	I2SCLK / (4 * (DIV * 2 + OF))

The relationship between audio sampling frequency (Fs) and I2S bitrate is defined by the following formula:

Fs = I2S bitrate / (number of bits per channel * number of channels)

So, in order to get the desired audio sampling frequency, the clock generator needs to be

configured according to the formulas listed in ***Table 19-8. Audio sampling frequency calculation formulas.***

Table 19-8. Audio sampling frequency calculation formulas

MCKOEN	CHLEN	Formula
0	0	I2SCLK / (32 * (DIV * 2 + OF))
0	1	I2SCLK / (64 * (DIV * 2 + OF))
1	0	I2SCLK / (256 * (DIV * 2 + OF))
1	1	I2SCLK / (256 * (DIV * 2 + OF))

Note: The configuration value of the I2S serial clock must be set to less than 1/6 of the PCLK clock (excluding 1/6).

19.4.5. Operation

Operation modes

The operation mode is selected by the I2SOPMOD bits in the SPI_I2SCTL register. There are four available operation modes, including master transmission mode, master reception mode, slave transmission mode, and slave reception mode. The direction of I2S interface signals for each operation mode is shown in the ***Table 19-9. Direction of I2S interface signals for each operation mode.***

Table 19-9. Direction of I2S interface signals for each operation mode

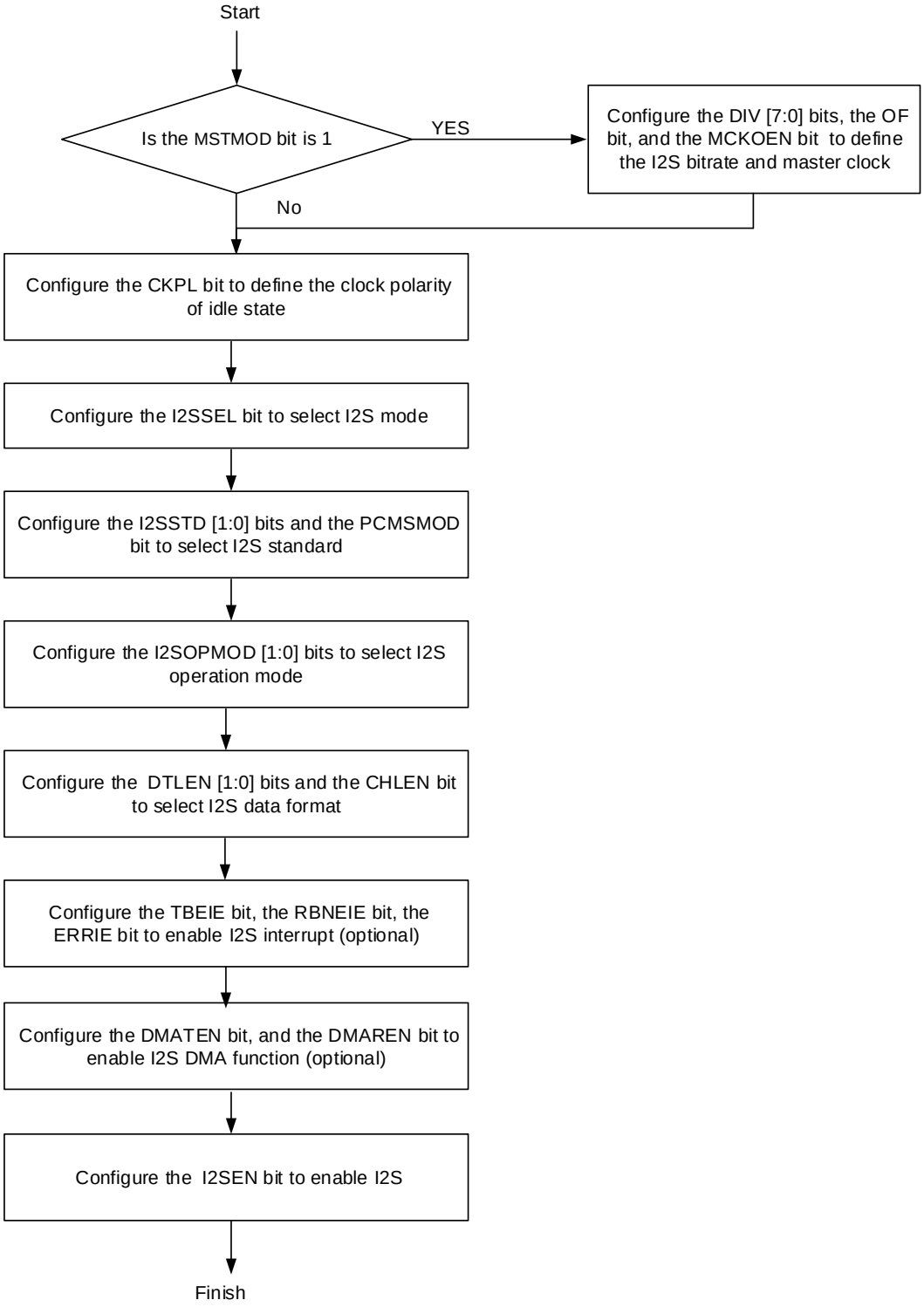
Operation mode	I2S_MCK	I2S_CK	I2S_WS	I2S_SD
Master transmission	Output or NU(1)	Output	Output	Output
Master reception	Output or NU(1)	Output	Output	Input
Slave transmission	Input or NU(1)	Input	Input	Output
Slave reception	Input or NU(1)	Input	Input	Input

1. NU means the pin is not used by I2S and can be used by other functions.

I2S initialization sequence

I2S initialization sequence contains five steps which is shown in ***Figure 19-52. I2S initialization sequence***.

Figure 19-52. I2S initialization sequence



I2S master transmission sequence

The TBE flag is used to control the transmission sequence. As is mentioned before, the TBE flag indicates that the transmit buffer is empty, and an interrupt will be generated if the TBEIE bit in the SPI_CTL1 register is set. At the beginning, the transmit buffer is empty (TBE is high) and no transmission sequence is processing in the shift register. When a half word is written

to the SPI_DATA register (TBE goes low), the data is transferred from the transmit buffer to the shift register (TBE goes high) immediately. At the moment, the transmission sequence begins.

The data is parallel loaded into the 16-bit shift register, and shifted out serially to the I2S_SD pin, MSB first. The next data should be written to the SPI_DATA register, when the TBE flag is high. After a write operation to the SPI_DATA register, the TBE flag goes low. When the current transmission finishes, the data in the transmit buffer is loaded into the shift register, and the TBE flag goes back high. Software should write the next audio data into SPI_DATA register before the current data finishes, otherwise, the audio data transmission is not continuous.

For all standards except PCM, the I2SCH flag is used to distinguish which channel side the data to transfer belongs to. The I2SCH flag is refreshed at the moment when the TBE flag goes high. At the beginning, the I2SCH flag is low, indicating the left channel data should be written to the SPI_DATA register.

In order to disable I2S, it is mandatory to clear the I2SEN bit after the TBE flag is high and the TRANS flag is low.

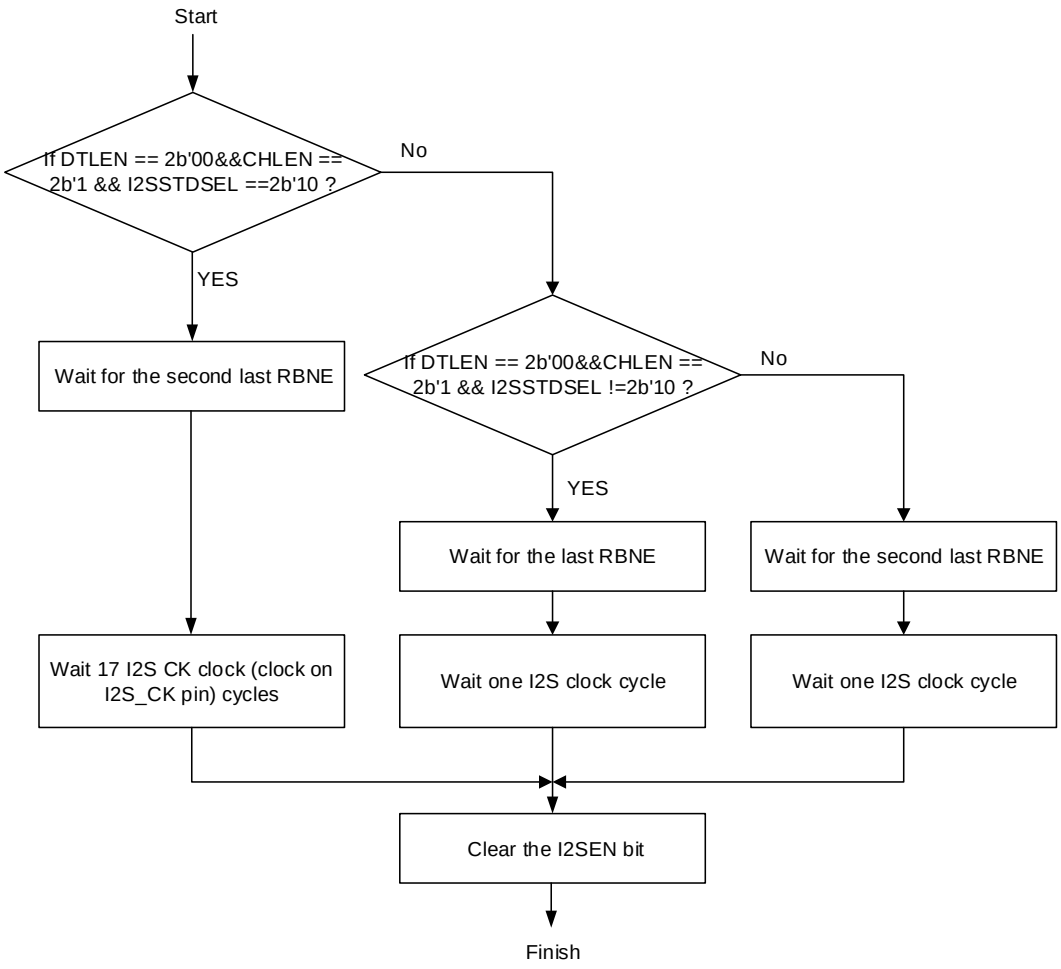
I2S master reception sequence

The RBNE flag is used to control the reception sequence. As is mentioned before, the RBNE flag indicates the receive buffer is not empty, and an interrupt will be generated if the RBNEIE bit in the SPI_CTL1 register is set. The reception sequence begins immediately when the I2SEN bit in the SPI_I2SCTL register is set. At the beginning, the receive buffer is empty (RBNE is low). When a reception sequence finishes, the received data in the shift register is loaded into the receive buffer (RBNE goes high). The data should be read from the SPI_DATA register, when the RBNE flag is high. After a read operation to the SPI_DATA register, the RBNE flag goes low. It is mandatory to read the SPI_DATA register before the end of the next reception. Otherwise, reception overrun error occurs. The RXORERR flag is set and an interrupt may be generated if the ERRIE bit in the SPI_CTL1 register is set. In this case, it is necessary to disable and then enable I2S before resuming the communication.

For all standards except PCM, the I2SCH flag is used to distinguish which channel side the received data belongs to. The I2SCH flag is refreshed at the moment when the RBNE flag goes high.

Different sequences are used to disable the I2S in different standards, data length and channel length. The sequences for each case are shown in ***Figure 19-53. I2S master reception disabling sequence***.

Figure 19-53. I2S master reception disabling sequence



I2S slave transmission sequence

The transmission sequence in slave mode is similar to that in master mode. The differences between them are described below.

In slave mode, the slave has to be enabled before the external master starts the communication. The transmission sequence begins when the external master sends the clock and when the I2S_WS signal requests the transfer of data. The data has to be written to the SPI_DATA register before the master initiates the communication. Software should write the next audio data into SPI_DATA register before the current data finishes. Otherwise, transmission underrun error occurs. The TXURERR flag is set and an interrupt may be generated if the ERRIE bit in the SPI_CTL1 register is set. In this case, it is mandatory to disable and enable I2S to resume the communication. In slave mode, I2SCH is sensitive to the I2S_WS signal coming from the external master.

In order to disable I2S, it is mandatory to clear the I2SEN bit after the TBE flag is high and the TRANS flag is low.

I2S slave reception sequence

The reception sequence in slave mode is similar to that in master mode. The differences between them are described below.

In slave mode, the slave has to be enabled before the external master starts the communication. The reception sequence begins when the external master sends the clock and when the I2S_WS signal indicates a start of the data transfer. In slave mode, I2SCH is sensitive to the I2S_WS signal coming from the external master.

In order to disable I2S, it is mandatory to clear the I2SEN bit immediately after receiving the last RBNE.

19.4.6. DMA function

DMA function is the same as SPI mode. The only difference is that the CRC function is not available in I2S mode.

19.4.7. I2S interrupts

Status flags

There are four status flags implemented in the SPI_STAT register, including TBE, RBNE, TRANS and I2SCH. The user can use them to fully monitor the state of the I2S bus.

- Transmit buffer empty flag (TBE)

This bit is set when the transmit buffer is empty, the software can write the next data to the transmit buffer by writing the SPI_DATA register.

- Receive buffer not empty flag (RBNE)

This bit is set when receive buffer is not empty, which means that one data is received and stored in the receive buffer, and software can read the data by reading the SPI_DATA register.

- I2S transmitting ongoing flag (TRANS)

TRANS is a status flag to indicate whether the transfer is ongoing or not. It is set and cleared by hardware and not controlled by software. This flag will not generate any interrupt.

- I2S channel side flag (I2SCH)

This flag indicates the channel side information of the current transfer and has no meaning in PCM mode. In the transmission mode, the I2SCH flag is updated every time TBE changes from 0 to 1. In the reception mode, the I2SCH flag is updated every time RBNE changes from 0 to 1. This flag will not generate any interrupt.

Error flags

There are three error flags:

- Transmission underrun error flag (TXURERR)

This situation occurs when the transmit buffer is empty if the valid SCK signal starts in slave transmission mode.

- Reception overrun error flag (RXORERR)

This situation occurs when the receive buffer is full and a newly incoming data has been completely received. When overrun occurs, the data in receive buffer is not updated and the newly incoming data is lost.

- Format Error (FERR)

In slave I2S mode, the I2S monitors the I2S_WS signal and an error flag will be set if I2S_WS toggles at an unexpected position.

I2S interrupt events and corresponding enable bits are summed up in the ***Table 19-10. I2S interrupt.***

Table 19-10. I2S interrupt

Interrupt flag	Description	Clear method	Interrupt enable bit
TBE	Transmit buffer empty	Write SPI_DATA register	TBEIE
RBNE	Receive buffer not empty	Read SPI_DATA register	RBNEIE
TXURERR	Transmission underrun error	Read SPI_STAT register	ERRIE
RXORERR	Reception overrun error	Read SPI_DATA register and then read SPI_STAT register.	
FERR	I2S format error	Read SPI_STAT register	

19.5. Register definition

SPI0 base address: 0x4001 3000
SPI1/I2S1 base address: 0x4000 3800
SPI2/I2S2 base address: 0x4000 3C00

19.5.1. Control register 0 (SPI_CTL0)

Address offset: 0x00
Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

This register has no meaning in I2S mode.

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BDEN	BDOEN	CRCEN	CRCNT	FF16	RO	SWNSS EN	SWNSS	LF	SPIEN	PSC[2:0]			MSTMOD	CKPL	CKPH
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw			rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	BDEN	Bidirectional enable 0: 2 line unidirectional transmit mode 1: 1 line bidirectional transmit mode. The data transfers between the MOSI pin of master and the MISO pin of slave.
14	BDOEN	Bidirectional transmit output enable When BDEN is set, this bit determines the direction of transfer. 0: Work in receive-only mode 1: Work in transmit-only mode
13	CRCEN	CRC calculation enable 0: CRC calculation is disabled 1: CRC calculation is enabled
12	CRCNT	CRC next transfer 0: Next transfer is data 1: Next transfer is CRC value When the transfer is managed by DMA, CRC value is transferred by hardware. This bit should be cleared. In full-duplex or transmit-only mode, set this bit after the last data is written to SPI_DATA

register. In receive-only mode, set this bit after the second last data is received.

11	FF16	Data frame format 0: 8-bit data frame format 1: 16-bit data frame format
10	RO	Receive only mode When BDEN is cleared, this bit determines the direction of transfer. 0: Full-duplex mode 1: Receive-only mode
9	SWNSSEN	NSS software mode enable 0: NSS hardware mode. The NSS level depends on NSS pin. 1: NSS software mode. The NSS level depends on SWNSS bit. This bit has no meaning in SPI TI mode.
8	SWNSS	NSS pin selection in NSS software mode 0: NSS pin is pulled low 1: NSS pin is pulled high This bit effects only when the SWNSSEN bit is set. This bit has no meaning in SPI TI mode.
7	LF	LSB first mode 0: Transmit MSB first 1: Transmit LSB first This bit has no meaning in SPI TI mode.
6	SPIEN	SPI enable 0: SPI peripheral is disabled 1: SPI peripheral is enabled
5:3	PSC[2:0]	Master clock prescaler selection 000: PCLK/2 001: PCLK/4 010: PCLK/8 011: PCLK/16 100: PCLK/32 101: PCLK/64 110: PCLK/128 111: PCLK/256 PCLK means PCLK2 when using SPI0 or PCLK1 when using SPI1 and SPI2.
2	MSTMOD	Master mode enable 0: Slave mode 1: Master mode
1	CKPL	Clock polarity selection

0: CLK pin is pulled low when SPI is idle

1: CLK pin is pulled high when SPI is idle

0 CKPH Clock phase selection

0: Capture the first data at the first clock transition

1: Capture the first data at the second clock transition

19.5.2. Control register 1 (SPI_CTL1)

Address offset: 0x04

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TBEIE	RBNEIE	ERRIE	TMOD	NSSP	NSSDRV	DMATEN	DMAREN
								rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	TBEIE	Transmit buffer empty interrupt enable 0: Disable TBE interrupt 1: Enable TBE interrupt. An interrupt is generated when the TBE bit is set.
6	RBNEIE	Receive buffer not empty interrupt enable 0: Disable RBNE interrupt 1: Enable RBNE interrupt. An interrupt is generated when the RBNE bit is set.
5	ERRIE	Errors interrupt enable 0: Disable error interrupt 1: Enable error interrupt. An interrupt is generated when the CRCERR bit, the CONFERR bit, the RXORERR bit or the TXURERR bit is set.
4	TMOD	SPI TI mode enable 0: Disable SPI TI mode 1: Enable SPI TI mode
3	NSSP	SPI NSS pulse mode enable 0: Disable SPI NSS pulse mode 1: Enable SPI NSS pulse mode
2	NSSDRV	Drive NSS output 0: Disable NSS output

1: Enable NSS output

1	DMATEN	Transmit buffer DMA enable 0: Disable transmit buffer DMA 1: Enable transmit buffer DMA, when the TBE bit in SPI_STAT is set, there will be a DMA request on corresponding DMA channel.
0	DMAREN	Receive buffer DMA enable 0: Disable receive buffer DMA 1: Enable receive buffer DMA, when the RBNE bit in SPI_STAT is set, there will be a DMA request on corresponding DMA channel.

19.5.3. Status register (SPI_STAT)

Address offset: 0x08
Reset value: 0x0000 0002

This register can be accessed by half-word (16-bit) or word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							FERR	TRANS	RXORERR	CONFERR	CRCERR	TXURERR	I2SCH	TBE	RBNE
							rc_w0	r	r	r	rc_w0	r	r	r	r

Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	FERR	Format error SPI TI Mode: 0: No TI mode format error 1: TI mode format error occurs I2S Mode: 0: No I2S format error 1: I2S format error occurs This bit is set by hardware and cleared by writing 0.
7	TRANS	Transmitting ongoing bit 0: SPI or I2S is idle. 1: SPI or I2S is currently transmitting and/or receiving a frame This bit is set and cleared by hardware.
6	RXORERR	Reception overrun error bit 0: No reception overrun error occurs. 1: Reception overrun error occurs. This bit is set by hardware and cleared by a read operation on the SPI_DATA register

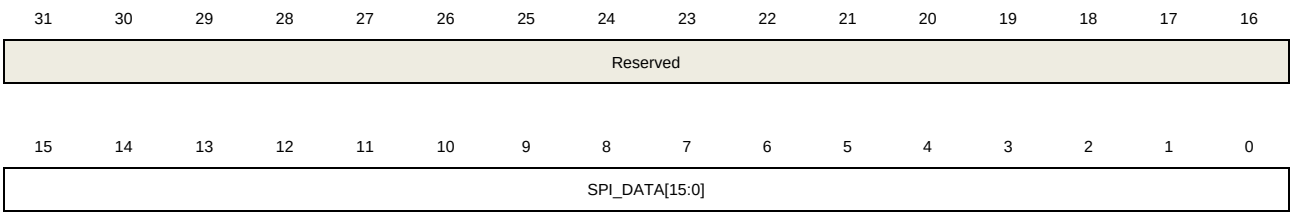
followed by a read access to the SPI_STAT register.

5	CONFERR	SPI Configuration error 0: No configuration fault occurs. 1: Configuration fault occurred. (In master mode, the NSS pin is pulled low in NSS hardware mode or SWNSS bit is low in NSS software mode.) This bit is set by hardware and cleared by a read or write operation on the SPI_STAT register followed by a write access to the SPI_CTL0 register. This bit is not used in I2S mode.
4	CRCERR	SPI CRC error bit 0: The SPI_RCRC value is equal to the received CRC data at last. 1: The SPI_RCRC value is not equal to the received CRC data at last. This bit is set by hardware and cleared by writing 0. This bit is not used in I2S mode.
3	TXURERR	Transmission underrun error bit 0: No transmission underrun error occurs. 1: Transmission underrun error occurs. This bit is set by hardware and cleared by a read operation on the SPI_STAT register. This bit is not used in SPI mode.
2	I2SCH	I2S channel side 0: The next data needs to be transmitted or the data just received is channel left. 1: The next data needs to be transmitted or the data just received is channel right. This bit is set and cleared by hardware. This bit is not used in SPI mode, and has no meaning in the I2S PCM mode.
1	TBE	Transmit buffer empty 0: Transmit buffer is not empty 1: Transmit buffer is empty
0	RBNE	Receive buffer not empty 0: Receive buffer is empty 1: Receive buffer is not empty

19.5.4. Data register (SPI_DATA)

Address offset: 0x0C
Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).



rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	SPI_DATA[15:0]	Data transfer register The hardware has two buffers, including transmit buffer and receive buffer. Write data to SPI_DATA will save the data to transmit buffer and read data from SPI_DATA will get the data from receive buffer. When the data frame format is set to 8-bit data, the SPI_DATA [15:8] is forced to 0 and the SPI_DATA[7:0] is used for transmission and reception, transmit buffer and receive buffer are 8-bit. If the data frame format is set to 16-bit data, the SPI_DATA[15:0] is used for transmission and reception, transmit buffer and receive buffer are 16-bit.

19.5.5. CRC polynomial register (SPI_CRCPOLY)

Address offset: 0x10

Reset value: 0x0000 0007

This register can be accessed by half-word (16-bit) or word (32-bit).

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

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

rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CRCPOLY[15:0]	CRC polynomial value These bits contain the CRC polynomial and they are used for CRC calculation. The default value is 0007h.

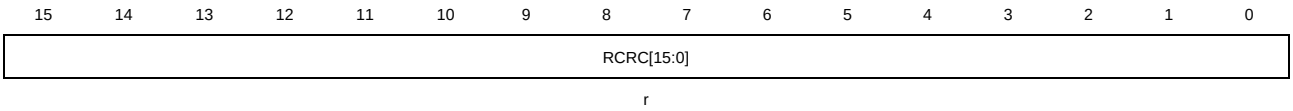
19.5.6. RX CRC register (SPI_RCRC)

Address offset: 0x14

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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



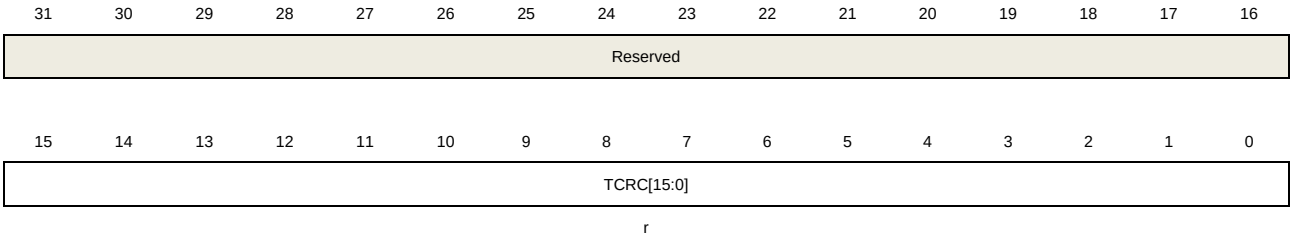
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	RCRC[15:0]	RX CRC value When the CRCEN bit of SPI_CTL0 is set, the hardware computes the CRC value of the received bytes and saves them in RCRC register. If the data frame format is set to 8-bit data, CRC calculation is based on CRC8 standard, and saves the value in RCRC[7:0], when the data frame format is set to 16-bit data, CRC calculation is based on CRC16 standard, and saves the value in RCRC[15:0]. The hardware computes the CRC value after each received bit, when the TRANS is set, a read to this register could return an intermediate value. This register is reset when the CRCEN bit in SPI_CTL0 register or the SPIxRST bit in RCU reset register is set.

19.5.7. TX CRC register (SPI_TCRC)

Address offset: 0x18

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	TCRC[15:0]	TX CRC value When the CRCEN bit of SPI_CTL0 is set, the hardware computes the CRC value of the transmitted bytes and saves them in TCRC register. If the data frame format is set to 8-bit data, CRC calculation is based on CRC8 standard, and saves the value in TCRC[7:0], when the data frame format is set to 16-bit data, CRC calculation is based on CRC16 standard, and saves the value in TCRC[15:0]. The hardware computes the CRC value after each transmitted bit, when the TRANS is set, a read to this register could return an intermediate value. The different frame formats (LF bit of the SPI_CTL0) will get different CRC values. This register is reset when the CRCEN bit in SPI_CTL0 register or the SPIxRST bit in

RCU reset register is set.

19.5.8. I2S control register (SPI_I2SCTL)

Address offset: 0x1C

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				I2SSEL	I2SEN	I2SOPMOD[1:0]		PCMSMOD	Reserved	I2SSTD[1:0]		CKPL	DTLEN[1:0]		CHLEN
				rw	rw	rw		rw		rw		rw	rw		rw

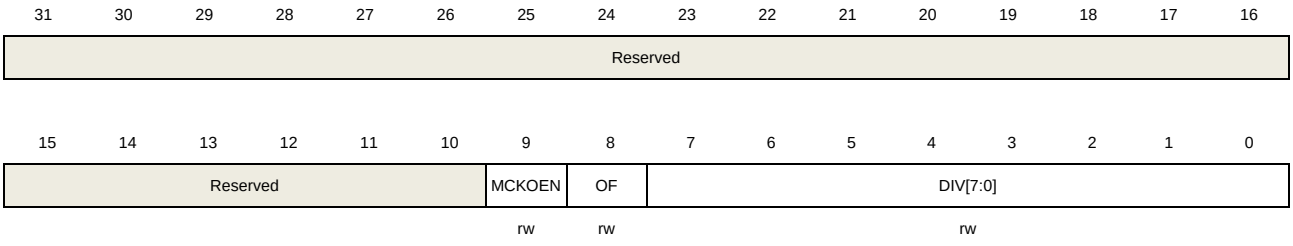
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11	I2SSEL	I2S mode selection 0: SPI mode 1: I2S mode This bit should be configured when SPI/I2S is disabled.
10	I2SEN	I2S enable 0: I2S is disabled 1: I2S is enabled This bit is not used in SPI mode.
9:8	I2SOPMOD[1:0]	I2S operation mode 00: Slave transmission mode 01: Slave reception mode 10: Master transmission mode 11: Master reception mode This bit should be configured when I2S is disabled. This bit is not used in SPI mode.
7	PCMSMOD	PCM frame synchronization mode 0: Short frame synchronization 1: Long frame synchronization This bit has a meaning only when PCM standard is used. This bit should be configured when I2S is disabled. This bit is not used in SPI mode.
6	Reserved	Must be kept at reset value.
5:4	I2SSTD[1:0]	I2S standard selection

		00: I2S Philips standard 01: MSB justified standard 10: LSB justified standard 11: PCM standard These bits should be configured when I2S is disabled. These bits are not used in SPI mode.
3	CKPL	Idle state clock polarity 0: The idle state of I2S_CK is low level 1: The idle state of I2S_CK is high level This bit should be configured when I2S is disabled. This bit is not used in SPI mode.
2:1	DTLEN[1:0]	Data length 00: 16 bits 01: 24 bits 10: 32 bits 11: Reserved These bits should be configured when I2S mode is disabled. These bits are not used in SPI mode.
0	CHLEN	Channel length 0: 16 bits 1: 32 bits The channel length must be equal to or greater than the data length. This bit should be configured when I2S mode is disabled. This bit is not used in SPI mode.

19.5.9. I2S clock prescaler register (SPI_I2SPSC)

Address offset: 0x20
Reset value: 0x0000 0002

This register can be accessed by half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	MCKOEN	I2S_MCK output enable 0: I2S_MCK output is disabled

		1: I2S_MCK output is enabled This bit should be configured when I2S is disabled. This bit is not used in SPI mode.
8	OF	Odd factor for the prescaler 0: Real divider value is DIV * 2 1: Real divider value is DIV * 2 + 1 This bit should be configured when I2S is disabled. This bit is not used in SPI mode.
7:0	DIV[7:0]	Dividing factor for the prescaler Real divider value is DIV * 2 + OF. DIV must not be 0. These bits should be configured when I2S is disabled. These bits are not used in SPI mode.

19.5.10. Quad-SPI mode control register (SPI_QCTL) of SPI0

Address offset: 0x80

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

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

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

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	QRD	Quad-SPI mode read select 0: SPI is in quad wire write mode 1: SPI is in quad wire read mode This bit should be only be configured when SPI is not busy (TRANS bit cleared). This bit is only available in SPI0.
0	QMOD	Quad-SPI mode enable 0: SPI is in single wire mode 1: SPI is in Quad-SPI mode This bit should only be configured when SPI is not busy (TRANS bit cleared). This bit is only available in SPI0.

20. External memory controller (EXMC)

20.1. Overview

The external memory controller EXMC, is used as a translator for CPU to access a variety of external memories. By configuring the related registers, it automatically converts AMBA memory access protocol into a specific memory access protocol, such as SRAM, PSRAM, ROM and NOR Flash. Users could also adjust the timing parameters in the configuration registers to improve memory access efficiency.

20.2. Characteristics

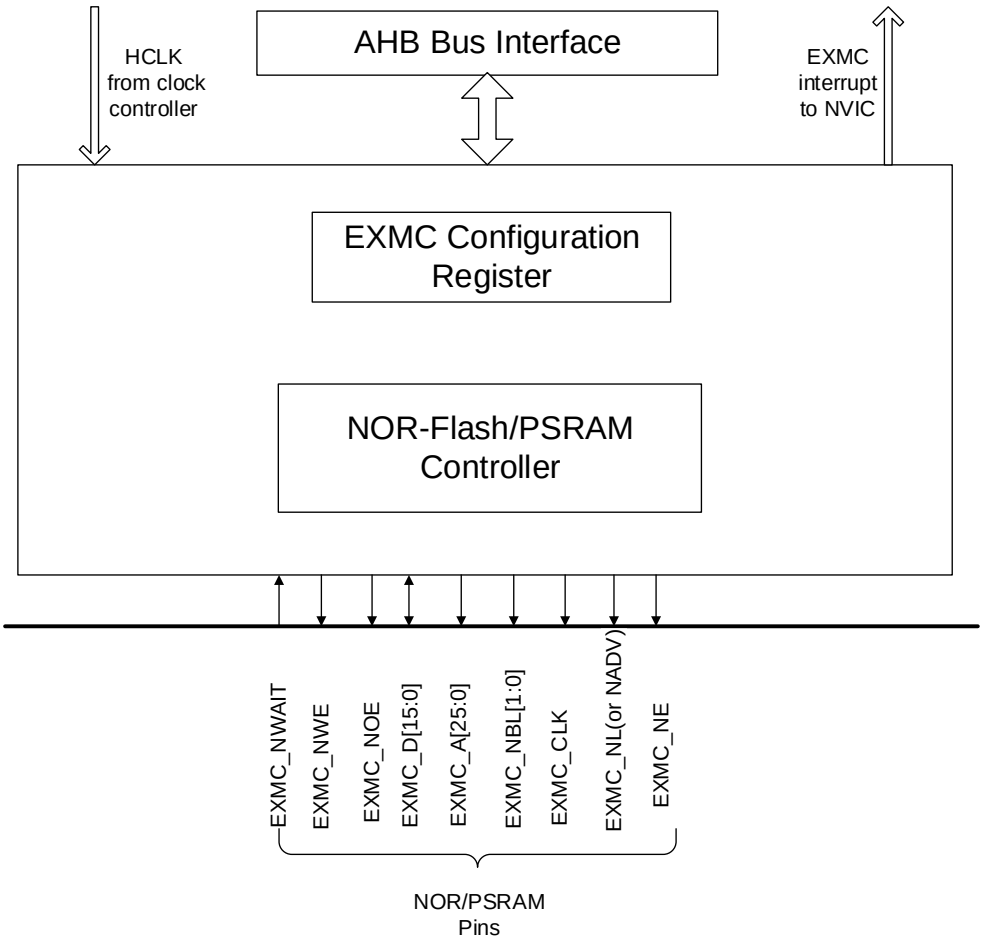
- Supported external memory:
  - SRAM
  - PSRAM
  - ROM
  - NOR Flash
- Protocol translation between the AMBA and the multitude of external memory protocol.
- Offering a variety of programmable timing parameters to meet user's specific needs.
- Independent read/write timing configuration for a sub-set memory types.
- 8 or 16 bits bus width.
- Address and data bus multiplexing mechanism for NOR Flash and PSRAM.
- Write enable and byte selection are provided if needed.
- Automatic AMBA transaction split when internal and external bus width are not compatible.

20.3. Function overview

20.3.1. Block diagram

EXMC is the combination of four modules: The AHB bus interface, EXMC configuration registers, NOR/PSRAM controller and external device interface. AHB clock (HCLK) is the reference clock.

Figure 20-1. The EXMC block diagram

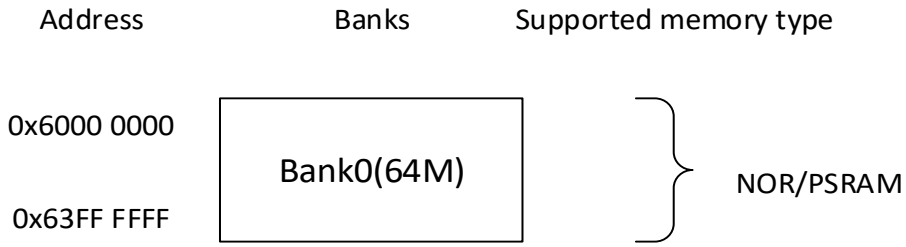


20.3.2. Basic regulation of EXMC access

EXMC is the conversion interface between AHB bus and external device protocol. 32-bit of AHB read/write access can split into several consecutive 8-bit or 16-bit read/write operations respectively. In the process of data transmission, AHB access data width and memory data width may not be the same. In order to ensure consistency of data transmission, read/write access of EXMC follows the following basic regulation.

- When the width of AHB bus equals to the memory bus width, no conversion is applied.
- When the width of AHB bus is greater than memory bus width, the AHB accesses will automatically split into several continuous memory accesses.
- When the width of AHB bus is shorter than memory bus width, if the external memory devices support the byte selection function, such as SRAM, ROM, PSRAM, the application can access the corresponding byte through EXMC_NBL[1:0]. Otherwise, write operation is prohibited, but read operation is allowed unconditionally.

Figure 20-2. EXMC memory banks



EXMC access space is bank0, which is 64 Mbytes, and is used for NOR and PSRAM device access.

HADDR[25:0] is the byte address whereas the external memory may not be byte accessed, this will lead to address inconsistency. EXMC can adjust HADDR to accommodate the data width of the external memory according to the following rules.

- When data bus width of the external memory is 8-bit, in this case the memory address is byte aligned. HADDR[25:0] is connected to EXMC_A[25:0] and then the EXMC_A[25:0] is connected to the external memory address lines.
- When data bus width of the external memory is 16-bit, in this case the memory address is half-word aligned. HADDR byte address must be converted into half-word aligned by connecting HADDR[25:1] with EXMC_A[24:0]. The EXMC_A[24:0] is then connected to the external memory address lines.

20.3.3. NOR/PSRAM controller

NOR/PSRAM memory controller controls bank0, which is designed to support NOR Flash, PSRAM, SRAM, ROM and honeycomb RAM external memory.

Note:

In asynchronous mode, all output signals of controller will change on the rising edge of internal AHB bus clock (HCLK).

In synchronous mode, all output data of controller will change on the falling edge of external memory device clock (EXMC_CLK).

NOR/PSRAM memory device interface description

Table 20-1. NOR flash interface signals description

EXMC pin	Direction	Mode	Functional description
EXMC_CLK	Output	Sync	Clock signal for sync
Non-muxed EXMC_A[25:0]	Output	Async/sync	Address bus signal
Muxed EXMC_A[25:16]			
EXMC_D[15:0]	Input/output	Async/sync (muxed)	Address/data bus

EXMC pin	Direction	Mode	Functional description
	Input/output	Async/sync (non-muxed)	Data bus
EXMC_NE	Output	Async/sync	Chip selection
EXMC_NOE	Output	Async/sync	Output enable(read enable)
EXMC_NWE	Output	Async/sync	Write enable
EXMC_NWAIT	Input	Async/sync	Wait input signal
EXMC_NL(NADV)	Output	Async/sync	Address valid

Table 20-2. PSRAM non-muxed signal description

EXMC pin	Direction	Mode	Functional description
EXMC_CLK	Output	Sync	Clock signal for sync
EXMC_A[25:0]	Output	Async/sync	Address bus
EXMC_D[15:0]	Input/output	Async/sync	Data bus
EXMC_NE	Output	Async/sync	Chip selection
EXMC_NOE	Output	Async/sync	Output enable(read enable)
EXMC_NWE	Output	Async/sync	Write enable
EXMC_NWAIT	Input	Async/sync	Wait input signal
EXMC_NL(NADV)	Output	Async/sync	Latch enable (address valid enable, NADV)
EXMC_NBL[1]	Output	Async/sync	Upper byte enable
EXMC_NBL[0]	Output	Async/sync	Lower byte enable

Supported memory access mode

Table below shows an example of the supported device types, access modes and transactions when the memory data bus is 16-bit for NOR, PSRAM and SRAM.

Table 20-3. EXMC bank0 supported transactions

Memory	Access mode	R/W	AHB transaction width	Memory transaction width	Comments
NOR flash	Async	R	8	16	
	Async	R	16	16	
	Async	W	16	16	
	Async	R	32	16	Split into 2 EXMC accesses
	Async	W	32	16	Split into 2 EXMC accesses
	Sync	R	16	16	
	Sync	R	32	16	
PSRAM	Async	R	8	16	
	Async	W	8	16	Use byte lanes

Memory	Access mode	R/W	AHB transaction width	Memory transaction width	Comments
					EXMC_NBL[1:0]
	Async	R	16	16	
	Async	W	16	16	
	Async	R	32	16	Split into 2 EXMC accesses
	Async	W	32	16	Split into 2 EXMC accesses
	Sync	R	16	16	
	Sync	R	32	16	
	Sync	W	8	16	Use byte lanes EXMC_NBL[1:0]
	Sync	W	16	16	
	Sync	W	32	16	
SRAM and ROM	Async	R	8	8	
	Async	R	8	16	
	Async	R	16	8	Split into 2 EXMC accesses
	Async	R	16	16	
	Async	R	32	8	Split into 4 EXMC accesses
	Async	R	32	16	Split into 2 EXMC accesses
	Async	W	8	8	
	Async	W	8	16	Use byte lanes EXMC_NBL[1:0]
	Async	W	16	8	
	Async	W	16	16	
	Async	W	32	8	
	Async	W	32	16	

NOR Flash/PSRAM controller timing

EXMC provides various programmable timing parameters and timing models for SRAM, ROM, PSRAM, NOR Flash and other external static memories.

Table 20-4. NOR/PSRAM controller timing parameters

Parameter	Function	Access mode	Unit	Min	Max
CKDIV	Sync clock divide ratio	Sync	HCLK	2	16
DLAT	Data latency	Sync	EXMC_CLK	2	17
BUSLAT	Bus latency	Async/sync read	HCLK	1	16
DSET	Data setup time	Async	HCLK	2	256

Parameter	Function	Access mode	Unit	Min	Max
AHLD	Address hold time	Async(muxed)	HCLK	2	16
ASET	Address setup time	Async	HCLK	1	16

Table 20-5. EXMC timing models

Timing model		Extend mode	Mode description	Write timing parameter	Read timing parameter
Async	Mode 1	0	SRAM/PSRAM/CRAM	DSET ASET	DSET ASET
	Mode 2	0	NOR flash	DSET ASET	DSET ASET
	Mode A	1	SRAM/PSRAM/CRAM with EXMC_NOE toggling on data phase	WDSET WASET	DSET ASET
	Mode B	1	NOR flash	WDSET WASET	DSET ASET
	Mode C	1	NOR flash with EXMC_NOE toggling on data phase	WDSET WASET	DSET ASET
	Mode D	1	With address hold capability	WDSET WAHLD WASET	DSET AHLD ASET
	Mode AM	0	NOR flash address/data muxed	DSET AHLD ASET BUSLAT	DSET AHLD ASET BUSLAT
Sync	Mode E	0	NOR/PSRAM/CRAM synchronous read, PSRAM/CRAM synchronous write	DLAT CKDIV	DLAT CKDIV
	Mode SM	0	NOR flash address/data muxed	DLAT CKDIV	DLAT CKDIV

As shown in ***Table 20-5. EXMC timing models***, EXMC NOR Flash/PSRAM controller provides a variety of timing models, users can modify those parameters listed in ***Table 20-4. NOR/PSRAM controller timing parameters*** to adapt to different external memory types and user's requirements. When extended mode is enabled via the EXMODEN bit in EXMC_SNCTL register, different timing patterns for read and write access could be generated independently according to the configuration of EXMC_SNTCFG and EXMC_SNWTCFG registers.

Asynchronous access timing diagram

Mode 1 - SRAM/CRAM

Figure 20-3. Mode 1 read access

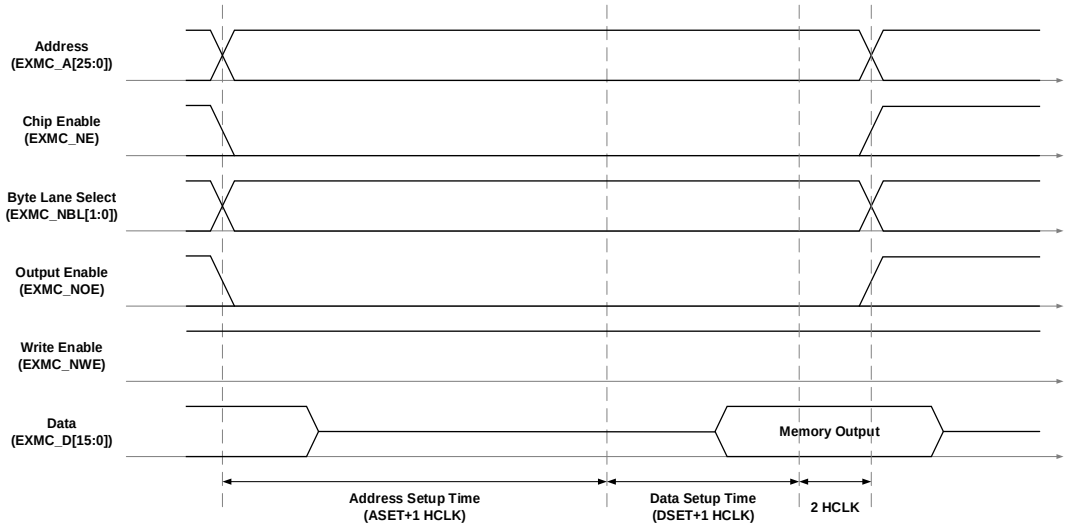


Figure 20-4. Mode 1 write access

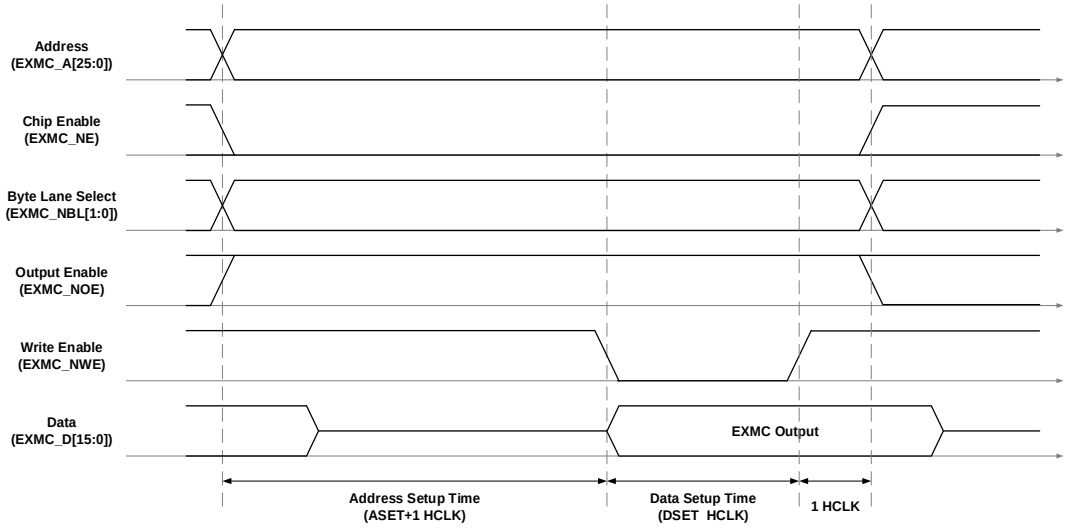


Table 20-6. Mode 1 related registers configuration

Bit position	Bit name	Reference setting value
EXMC_SNCTL		
31-20	Reserved	0x000
19	SYNCWR	0x0
18-16	CPS	0x0
15	ASYNCWAIT	Depends on memory
14	EXMODEN	0x0
13	NRWTEN	0x0
12	WREN	Depends on user
11	NRWTCFG	No effect
10	WRAPEN	0x0
9	NRWTPOL	Meaningful only when the bit 15 is set to 1
8	SBRSTEN	0x0
7	Reserved	0x1

Bit position	Bit name	Reference setting value
6	NREN	No effect
5-4	NRW	Depends on memory
3-2	NRTP	Depends on memory, except 0x2(NOR flash)
1	NRMUX	0x0
0	NRBKEN	0x1
EXMC_SNTCFG		
31-30	Reserved	0x0
29-28	ASYNCMOD	No effect
27-24	DLAT	No effect
23-20	CKDIV	No effect
19-16	BUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	DSET	Depends on memory and user (DSET+1 HCLK for write, DSET+3 HCLK for read)
7-4	AHLD	No effect
3-0	ASET	Depends on memory and user

Mode A - SRAM/PSRAM(CRAM) OE toggling

Figure 20-5. Mode A read access

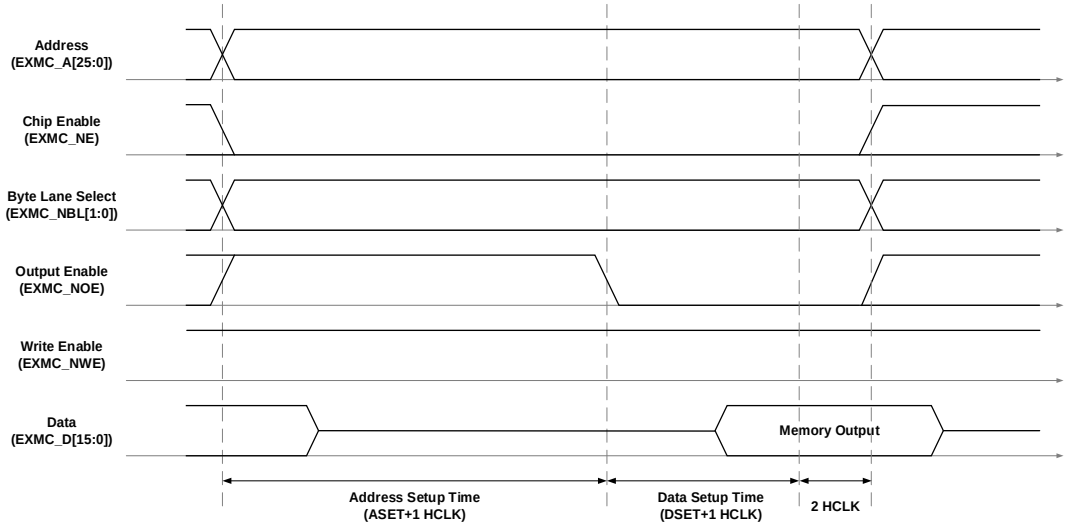
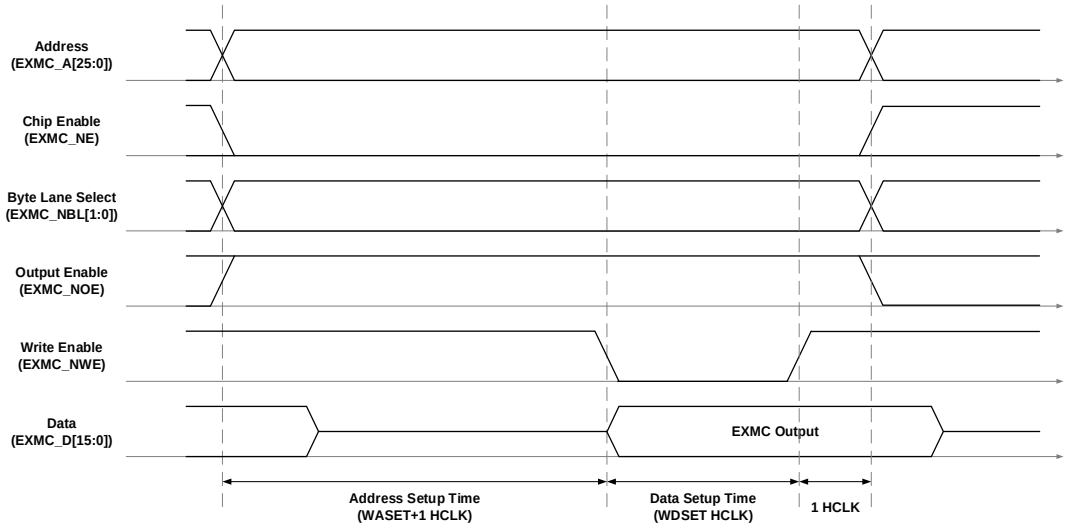


Figure 20-6. Mode A write access



The difference of write timing between mode A and mode 1 is that when read and write timings are specified by the same set of timing configurations, mode A write timing configuration is independent of its read configuration.

Table 20-7. Mode A related registers configuration

Bit position	Bit name	Reference setting value
EXMC_SNCTL		
31-20	Reserved	0x000
19	SYNCWR	0x0
18-16	CPS	0x0
15	ASYNCWTEN	Depends on memory
14	EXMODEN	0x1
13	NRWTEN	0x0
12	WREN	Depends on user
11	NRWTCFG	No effect
10	WRAPEN	0x0
9	NRWTPOL	Meaningful only when the bit 15 is set to 1
8	SBRSTEN	0x0
7	Reserved	0x1
6	NREN	No effect
5-4	NRW	Depends on memory
3-2	NRTP	Depends on memory, except 0x2(NOR flash)
1	NRMUX	0x0
0	NRBKEN	0x1
EXMC_SNTCFG(read)		
31-30	Reserved	0x0
29-28	ASYNCMOD	0x0
27-24	DLAT	No effect
23-20	CKDIV	No effect

Bit position	Bit name	Reference setting value
19-16	BUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	DSET	Depends on memory and user (DSET+3 HCLK for read)
7-4	AHLD	No effect
3-0	ASET	Depends on memory and user
EXMC_SNWTCFG(write)		
31-30	Reserved	0x0
29-28	WASYNCMOD	0x0
27-20	Reserved	0xFF
19-16	WBUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	WDSET	Depends on memory and user (WDSET+1 HCLK for write)
7-4	WAHLD	0x0
3-0	WASET	Depends on memory and user

Mode 2/B - NOR Flash

Figure 20-7. Mode 2/B read access

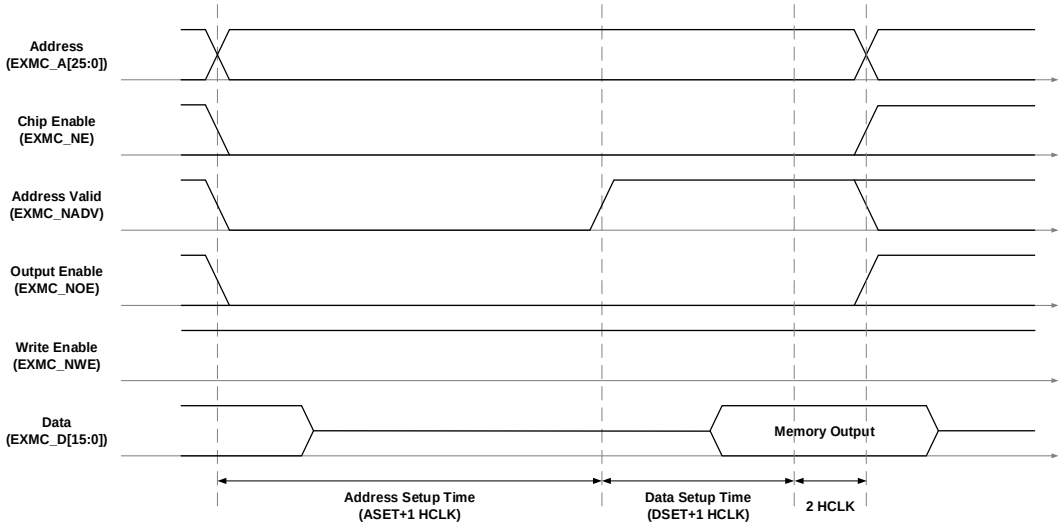


Figure 20-8. Mode 2 write access

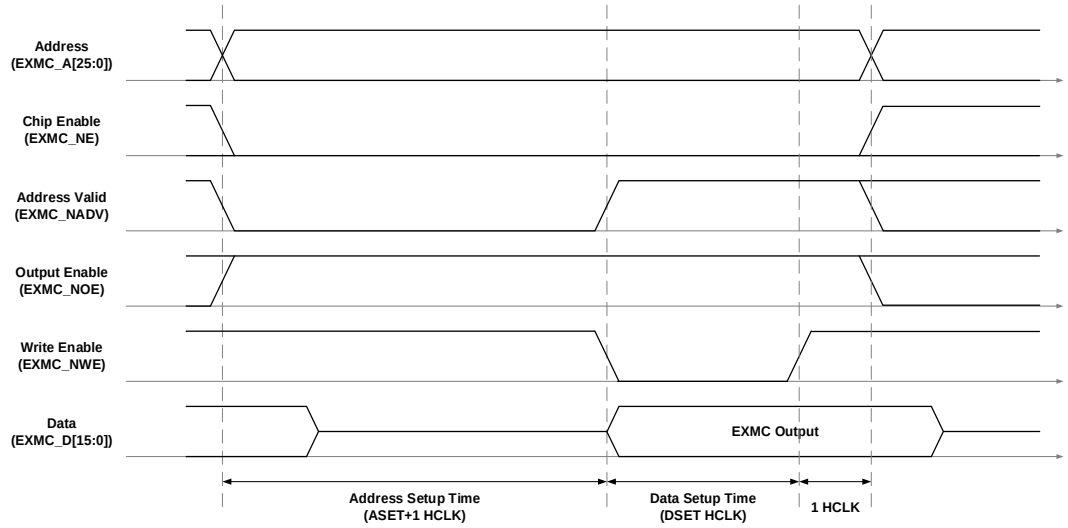


Figure 20-9. Mode B write access

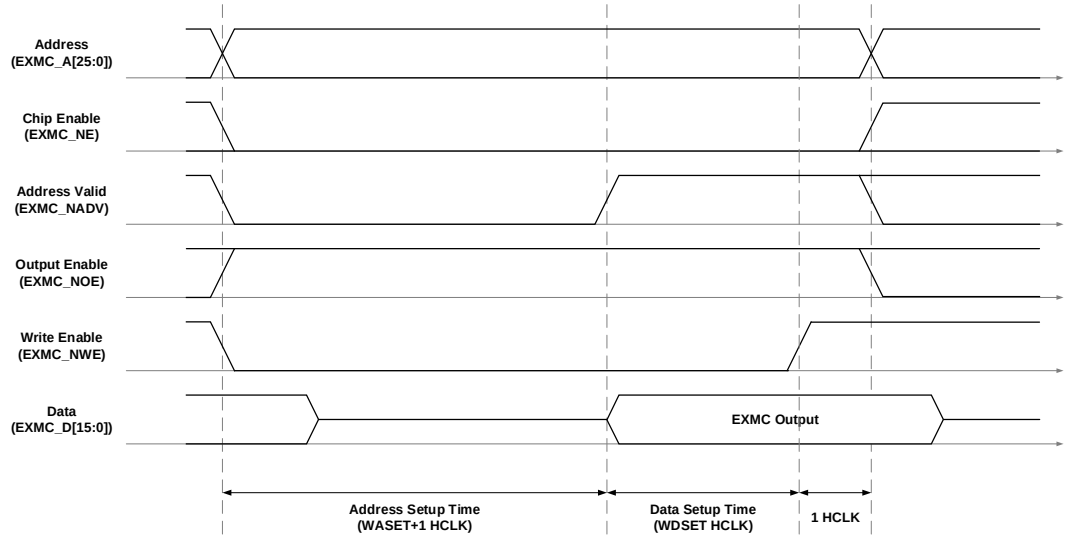


Table 20-8. Mode 2/B related registers configuration

Bit position	Bit name	Reference setting value
EXMC_SNCTL(mode 2, mode B)		
31-20	Reserved	0x000
19	SYNCWR	0x0
18-16	CPS	0x0
15	ASYNCWTEN	Depends on memory
14	EXMODEN	Mode 2: 0x0, mode B:0x1
13	NRWTEN	0x0
12	WREN	Depends on user
11	NRWTCFG	No effect
10	WRAPEN	0x0
9	NRWTPOL	Meaningful only when the bit 15 is set to 1
8	SBRSTEN	0x0
7	Reserved	0x1

Bit position	Bit name	Reference setting value
6	NREN	0x1
5-4	NRW	Depends on memory
3-2	NRTP	0x2, NOR flash
1	NRMUX	0x0
0	NRBKEN	0x1
EXMC_SNTCFG(read and write in mode 2, read in mode B)		
31-30	Reserved	0x0
29-28	ASYNCMOD	Mode B:0x1
27-24	DLAT	No effect
23-20	CKDIV	No effect
19-16	BUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	DSET	Depends on memory and user (DSET+3 HCLK for read)
7-4	AHLD	0x0
3-0	ASET	Depends on memory and user
EXMC_SNWTCFG(write in mode B)		
31-30	Reserved	0x0
29-28	WASYNCMOD	Mode B:0x1
27-20	Reserved	0xFF
19-16	WBUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	WDSET	Depends on memory and user (WDSET+1 HCLK for write)
7-4	WAHLD	0x0
3-0	WASET	Depends on memory and user

Mode C - NOR Flash OE toggling

Figure 20-10. Mode C read access

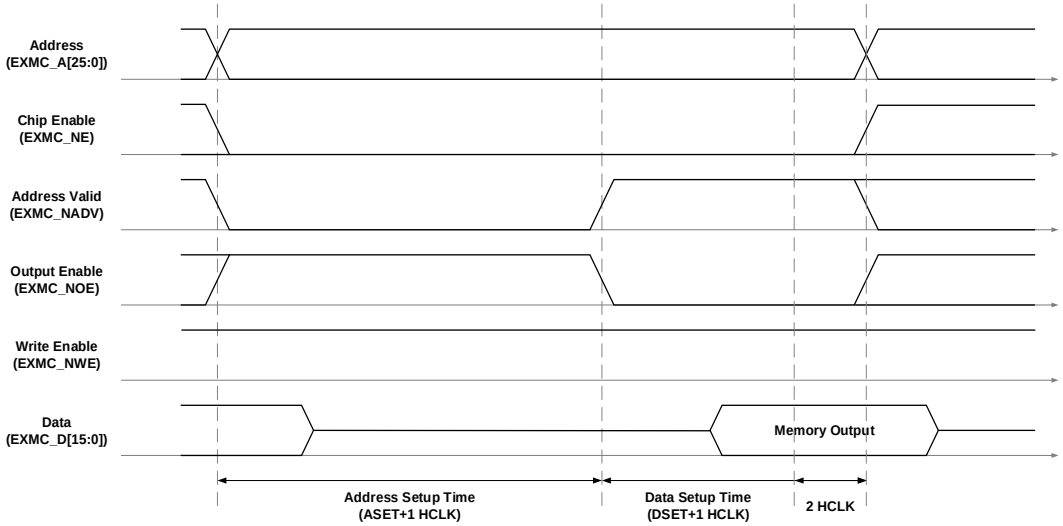
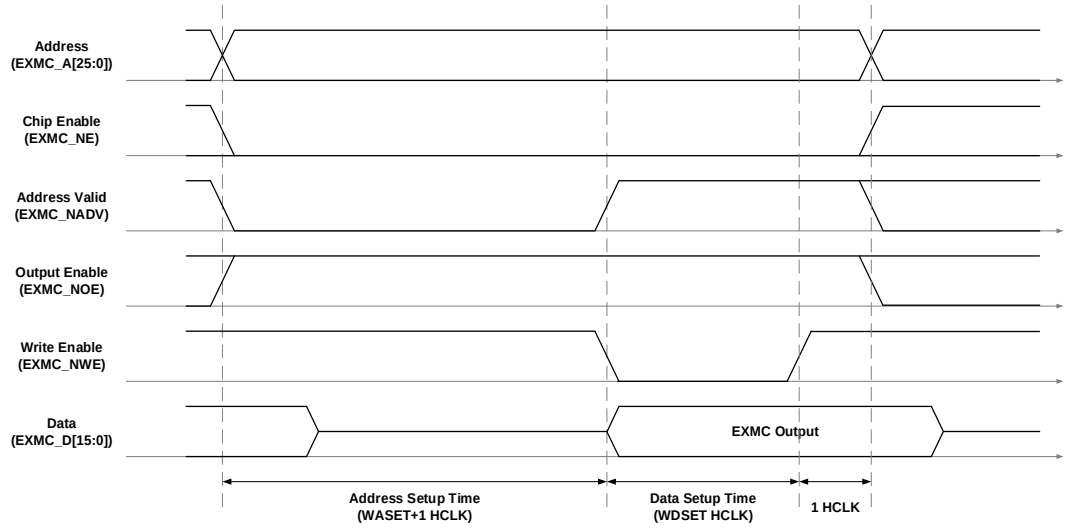


Figure 20-11. Mode C write access



The difference of write timing between mode C and mode 1 is that when read and write timings are specified by the same set of timing configurations, mode C write timing configuration is independent of its read configuration.

Table 20-9. Mode C related registers configuration

Bit position	Bit name	Reference setting value
EXMC_SNCTL		
31-20	Reserved	0x000
19	SYNCWR	0x0
18-16	CPS	0x0
15	ASYNCWTEN	Depends on memory
14	EXMODEN	0x1
13	NRWTEN	0x0
12	WREN	Depends on user
11	NRWTCFG	No effect
10	WRAPEN	0x0
9	NRWTPOL	Meaningful only when the bit 15 is set to 1
8	SBRSTEN	0x0
7	Reserved	0x1
6	NREN	0x1
5-4	NRW	Depends on memory
3-2	NRTP	0x2, NOR flash
1	NRMUX	0x0
0	NRBKEN	0x1
EXMC_SNTCFG		
31-30	Reserved	0x0
29-28	ASYNCMOD	Mode C: 0x2
27-24	DLAT	0x0
23-20	CKDIV	0x0

Bit position	Bit name	Reference setting value
19-16	BUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	DSET	Depends on memory and user (DSET+3 HCLK for read)
7-4	AHLD	0x0
3-0	ASET	Depends on memory and user
EXMC_SNWTCFG		
31-30	Reserved	0x0
29-28	WASYNCMOD	Mode C: 0x2
27-20	Reserved	0xFF
19-16	WBUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	WDSET	Depends on memory and user (WDSET+1 HCLK for write)
7-4	WAHLD	0x0
3-0	WASET	Depends on memory and user

Mode D - Asynchronous access with extended address

Figure 20-12. Mode D read access

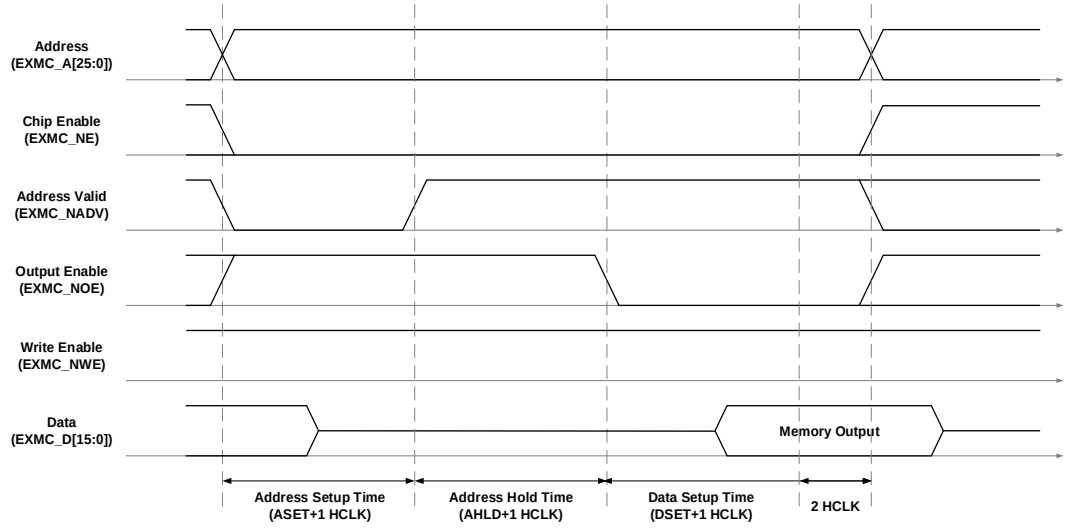


Figure 20-13. Mode D write access

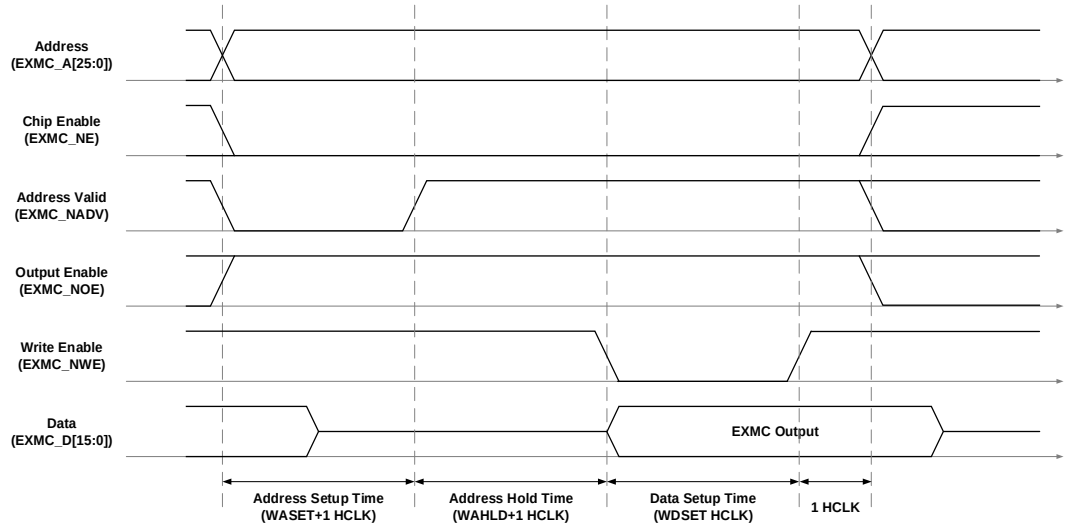


Table 20-10. Mode D related registers configuration

Bit position	Bit name	Reference setting value
EXMC_SNCTL		
31-20	Reserved	0x000
19	SYNCWR	0x0
18-16	CPS	0x0
15	ASYNCWTEN	Depends on memory
14	EXMODEN	0x1
13	NRWTEN	0x0
12	WREN	Depends on user
11	NRWTCFG	No effect
10	WRAPEN	0x0
9	NRWTPOL	Meaningful only when the bit 15 is set to 1
8	SBRSTEN	0x0
7	Reserved	0x1
6	NREN	Depends on memory
5-4	NRW	Depends on memory
3-2	NRTP	Depends on memory
1	NRMUX	0x0
0	NRBKEN	0x1
EXMC_SNTCFG		
31-30	Reserved	0x0
29-28	ASYNCMOD	Mode D: 0x3
27-24	DLAT	Don’t care
23-20	CKDIV	No effect
19-16	BUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	DSET	Depends on memory and user (DSET+3 HCLK for read)

Bit position	Bit name	Reference setting value
7-4	AHLD	Depends on memory and user
3-0	ASET	Depends on memory and user
EXMC_SNWTCFG		
31-30	Reserved	0x0
29-28	WASYNCMOD	Mode D: 0x3
27-20	Reserved	0xFF
19-16	WBUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	WDSET	Depends on memory and user (WDSET+1 HCLK for write)
7-4	WAHLD	Depends on memory and user
3-0	WASET	Depends on memory and user

Mode AM - NOR Flash address / data bus multiplexing

Figure 20-14. Multiplex mode read access

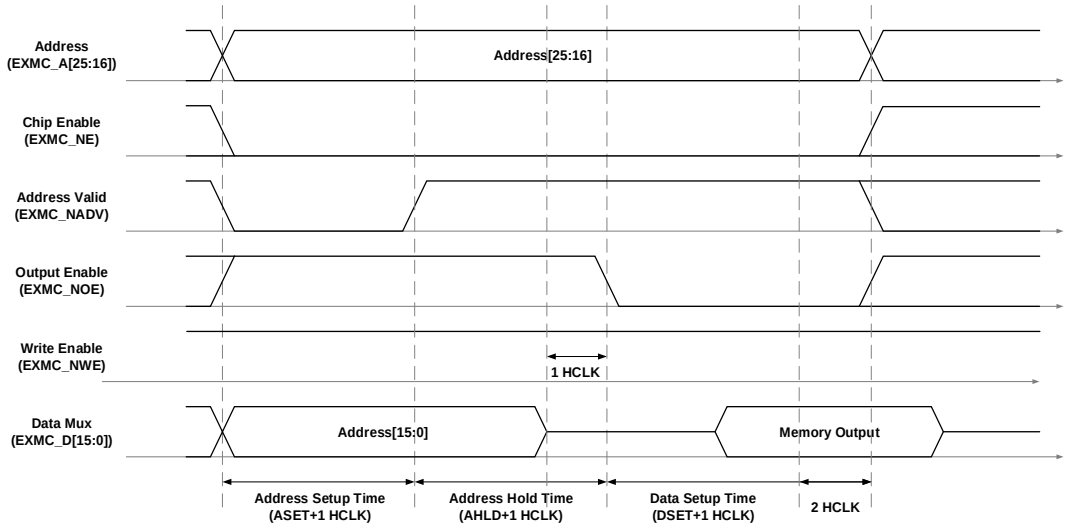


Figure 20-15. Multiplex mode write access

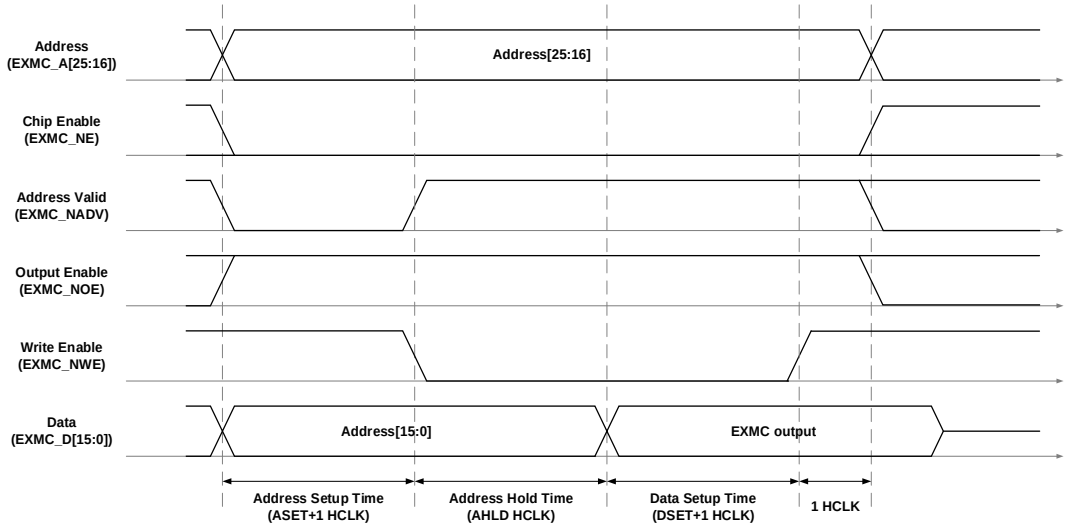


Table 20-11. Related registers configuration of multiplex mode

Bit position	Bit name	Reference setting value
EXMC_SNCTL		
31-20	Reserved	0x000
19	SYNCWR	0x0
18-16	CPS	0x0
15	ASYNCWTEN	Depends on memory
14	EXMODEN	0x0
13	NRWTEN	0x0
12	WREN	Depends on memory
11	NRWTCFG	No effect
10	WRAPEN	0x0
9	NRWTPOL	Meaningful only when the bit 15 is set to 1
8	SBRSTEN	0x0
7	Reserved	0x1
6	NREN	0x1
5-4	NRW	Depends on memory
3-2	NRTP	0x2: NOR flash
1	NRMUX	0x1
0	NRBKEN	0x1
EXMC_SNTCFG		
31-30	Reserved	0x0
29-28	ASYNCMOD	0x0
27-24	DLAT	No effect
23-20	CKDIV	No effect
19-16	BUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	DSET	Depends on memory and user (DSET+2 HCLK for write, DSET+3 HCLK for read)
7-4	AHLD	Depends on memory and user
3-0	ASET	Depends on memory and user

Wait timing of asynchronous communication

Wait function is controlled by the bit ASYNCWAIT in register EXMC_SNCTL. During external memory access, data setup phase will be automatically extended by the active EXMC_NWAIT signal if ASYNCWAIT bit is set. The extended time is calculated as follows:

If memory wait signal is aligned to EXMC_NOE/ EXMC_NWE:

$$T_{DATA_SETUP} \geq \max T_{WAIT_ASSERTION} + 4HCLK \quad (20\text{-}1)$$

If memory wait signal is aligned to EXMC_NE:

If

$$\max T_{WAIT_ASSERTION} \geq T_{ADDRES_PHASE} + T_{HOLD_PHASE} \quad (20\text{-}2)$$

be

$$T_{DATA_SETUP} \geq (\max T_{WAIT_ASSERTION} - T_{ADDRES_PHASE} - T_{HOLD_PHASE}) + 4HCLK \quad (20\text{-}3)$$

Otherwise

$$T_{DATA_SETUP} \geq 4HCLK \quad (20\text{-}4)$$

Figure 20-16. Read access timing diagram under async-wait signal assertion

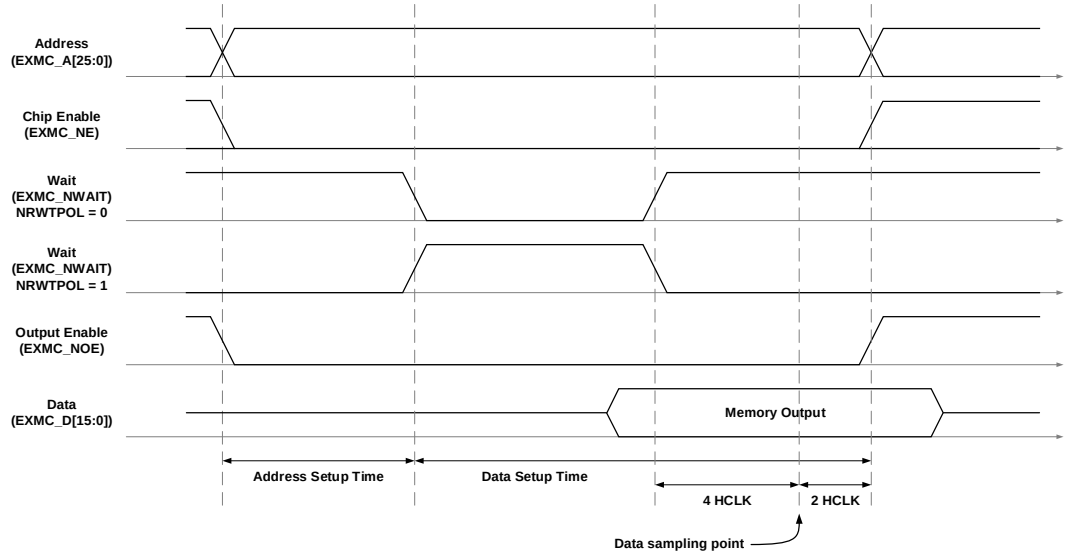
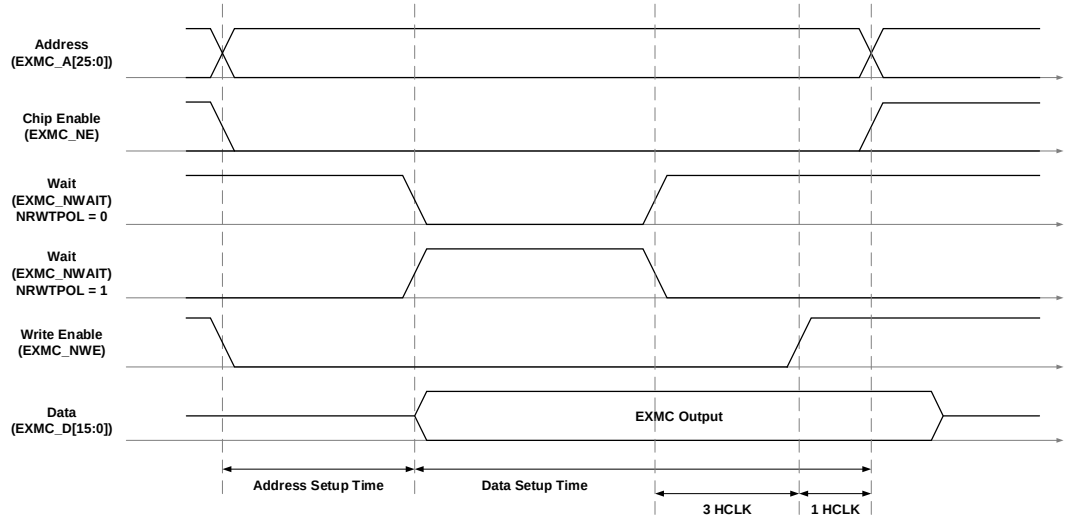


Figure 20-17. Write access timing diagram under async-wait signal assertion



Synchronous access timing

The relationship between memory clock (EXMC_CLK) and system clock (HCLK) is as follows:

$$EXMC_CLK = \frac{HCLK}{CKDIV+1} \quad (20\text{-}5)$$

CKDIV is the synchronous clock divider ratio, it is configured through the CKDIV control field in the EXMC_SNTCFG register.

1. Data latency and NOR Flash latency

Data latency (DLAT) is the number of EXMC_CLK cycles to wait before sampling the data. The relationship between data latency and latency parameter of NOR Flash in specification is as follows.

For specification of NOR Flash excludes the EXMC_NADV cycle, their relationship should be:

NOR Flash latency=DLAT+2 (20-6)

For specification of NOR Flash includes the EXMC_NADV cycle, their relationship should be:

NOR Flash latency=DLAT+3 (20-7)

2. Data wait

Users should guarantee that EXMC_NWAIT signal matches that of the external device. This signal is configured through the EXMC_SNCTL registers, it is enabled by the NRWTEN bit, and the active timing could be one data cycle before the wait state or active during the wait state by the NRWTCFG bit, and the wait signal polarity is set by the NRWTPOL bit.

In NOR Flash synchronous burst access mode, when NRWTEN bit in EXMC_SNCTL register is set, EXMC_NWAIT signal will be detected after a period of data latency. If EXMC_NWAIT signal detected is valid, wait cycles will be inserted until EXMC_NWAIT becomes invalid.

- The valid polarity of EXMC_NWAIT:

NRWTPOL= 1: Valid level of EXMC_NWAIT signal is high.
NRWTPOL= 0: Valid level of EXMC_NWAIT signal is low.

- In synchronous burst mode, EXMC_NWAIT signal has two kinds of configurations:

NRWTCFG = 1: When EXMC_NWAIT signal is active, the data of the current cycle is not valid.
NRWTCFG = 0: When EXMC_NWAIT signal is active, the data of the next cycle is not valid. It is the default state after reset.

During wait state which is inserted via the EXMC_NWAIT signal, the controller continues to send clock pulses to the memory, keep the chip select signal and output signals available, and ignore the invalid data signal.

3. Automatic burst split at CRAM page boundary

Crossing page boundary burst access is prohibited in CRAM 1.5, an automatic burst split functionality is implemented by the EXMC. To guarantee correct burst split operation, users should specify CRAM page size by configuring the CPS bit in EXMC_SNCTL register to inform the EXMC when this functionality should be performed.

4. Mode SM - Single burst transmission

For synchronous burst transmission, if the needed data of AHB is 16-bit, EXMC will perform a burst transmission whose length is 1. If the needed data of AHB is 32-bit, EXMC will make the transmission divided into two 16-bit transmissions, that is, EXMC performs a burst transmission whose length is 2.

For other configurations please refer to ***Table 20-3. EXMC bank0 supported transactions***.

Read timing of synchronous multiplexed burst mode - NOR, PSRAM (CRAM)

Figure 20-18. Read timing of synchronous multiplexed burst mode

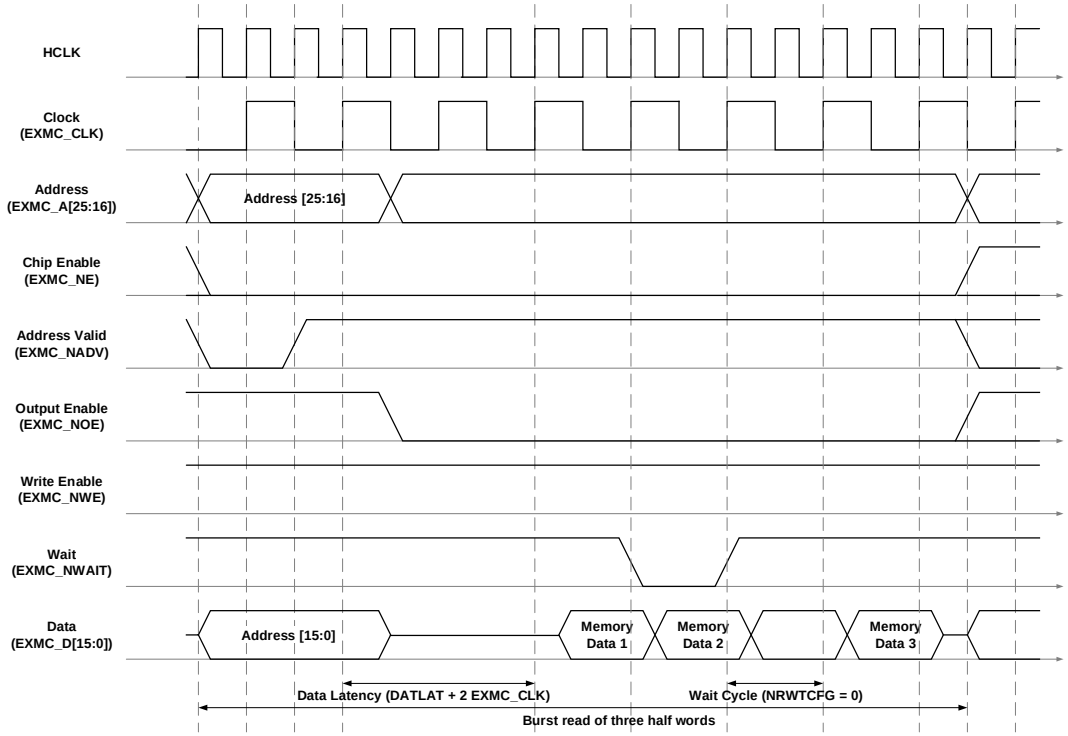


Table 20-12. Timing configurations of synchronous multiplexed read mode

Bit position	Bit name	Reference setting value
EXMC_SNCTL		
31-20	Reserved	0x000
19	SYNCWR	No effect
18-16	CPS	0x0
15	ASYNCWTEN	0x0
14	EXMODEN	0x0
13	NRWTEN	Depends on memory
12	WREN	No effect
11	NRWTCFG	Depends on memory
10	WRAPEN	0x0
9	NRWTPOL	Depends on memory
8	SBRSTEN	0x1, burst read enable
7	Reserved	0x1
6	NREN	Depends on memory
5-4	NRW	0x1
3-2	NRTP	Depends on memory, 0x1/0x2
1	NRMUX	0x1, depends on memory and users
0	NRBKEN	0x1
EXMC_SNTCFG(read)		

Bit position	Bit name	Reference setting value
31-30	Reserved	0x0
29-28	ASYNCMOD	0x0
27-24	DLAT	Data latency
23-20	CKDIV	The figure above: 0x1, EXMC_CLK=2HCLK
19-16	BUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	DSET	No effect
7-4	AHLD	No effect
3-0	ASET	No effect

Mode SM – Write timing of synchronous multiplexed burst mode – PSRAM (CRAM)

Figure 20-19. Write timing of synchronous multiplexed burst mode

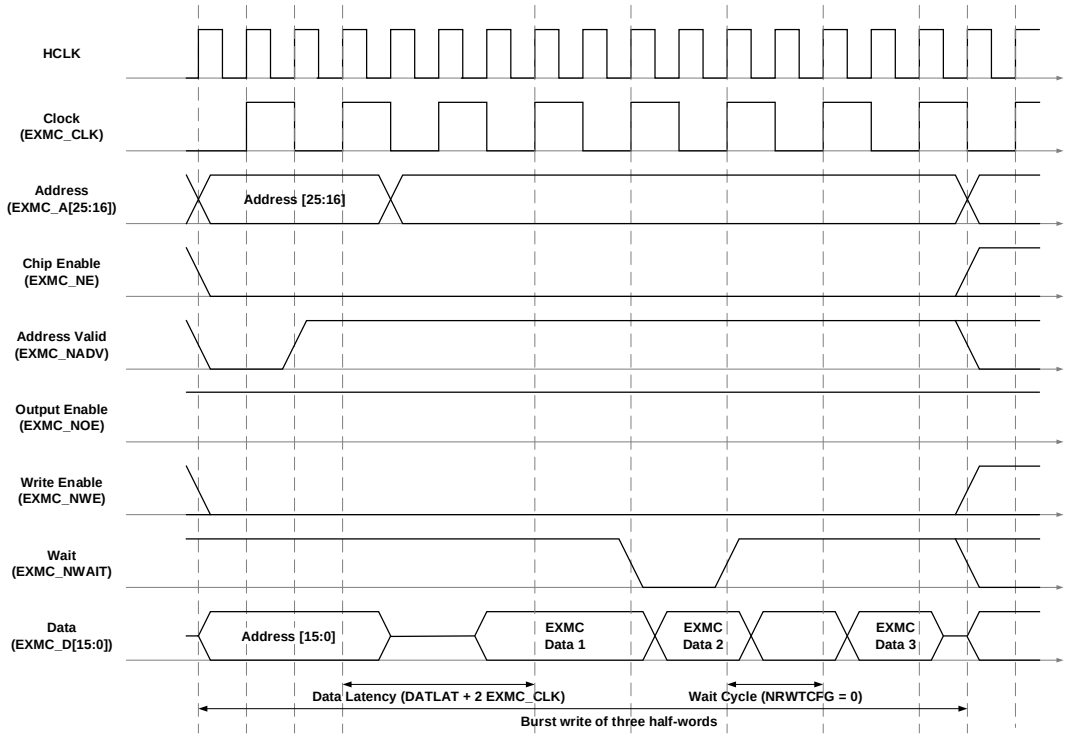


Table 20-13. Timing configurations of synchronous multiplexed write mode

Bit position	Bit name	Reference setting value
EXMC_SNCTL		
31-20	Reserved	0x000
19	SYNCWR	0x1, synchronous write enable
18-16	CPS	0x0
15	AYSNCWAIT	0x0
14	EXMODEN	0x0
13	NRWTEN	Depends on memory
12	WREN	0x1
11	NRWTCFG	0x0(here must be zero)
10	WRAPEN	0x0

Bit position	Bit name	Reference setting value
9	NTWTPOL	Depends on memory
8	SBRSTEN	No effect
7	Reserved	0x1
6	NREN	Depends on memory
5-4	NRW	0x1
3-2	NRTP	0x1
1	NRMUX	0x1, depends on users
0	NRBKEN	0x1
EXMC_SNTCFG(write)		
31-30	Reserved	0x0
29-28	ASYNCMOD	0x0
27-24	DLAT	Data latency
23-20	CKDIV	The figure above: 0x1, EXMC_CLK=2HCLK
19-16	BUSLAT	Time between EXMC_NE rising edge to EXMC_NE falling edge
15-8	DSET	No effect
7-4	AHLD	No effect
3-0	ASET	No effect

20.4. Register definition

EXMC base address: 0xA000 0000

20.4.1. SRAM/NOR Flash control registers (EXMC_SNCTL)

Address offset: 0x00

Reset value: 0x0000 30DB

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												SYNCWR	CPS[2:0]		
												rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ASYNCWAIT	EXMODEN	NRWTEN	WREN	NRWTCFG	WRAPEN	NRWTPOL	SBRSTEN	Reserved	NREN	NRW[1:0]		NRTP[1:0]		NRMUX	NRBKEN
rw	rw	rw	rw	rw	rw	rw	rw		rw	rw		rw		rw	rw

Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19	SYNCWR	Synchronous write 0: Asynchronous write 1: Synchronous write
18:16	CPS[2:0]	CRAM page size 000: Automatic burst split when crossing page boundary 001: 128 bytes 010: 256 bytes 011: 512 bytes 100: 1024 bytes Others: Reserved
15	ASYNCWAIT	Asynchronous wait 0: Disable the asynchronous wait function 1: Enable the asynchronous wait function
14	EXMODEN	Extended mode enable 0: Disable extended mode 1: Enable extended mode
13	NRWTEN	NWAIT signal enable For flash memory access in burst mode, this bit enables/disables wait-state insertion to the NWAIT signal. 0: Disable NWAIT signal

		1: Enable NWAIT signal
12	WREN	Write enable 0: Disable writing in the bank by the EXMC, otherwise an AHB error is reported 1: Enable writing in the bank by the EXMC (default after reset)
11	NRWTCFG	NWAIT signal configuration, only work in synchronous mode 0: NWAIT signal is active one data cycle before wait state 1: NWAIT signal is active during wait state
10	WRAPEN	Wrapped burst mode enable 0: Disable wrap burst mode support 1: Enable wrap burst mode support
9	NRWTPOL	NWAIT signal polarity 0: Low level of NWAIT is active 1: High level of NWAIT is active
8	SBRSTEN	Synchronous burst enable 0: Disable burst access mode 1: Enable burst access mode
7	Reserved	Must be kept at reset value.
6	NREN	NOR Flash access enable 0: Disable NOR Flash access 1: Enable NOR Flash access
5:4	NRW[1:0]	NOR region memory data bus width 00: 8 bits 01: 16 bits(default after reset) 10/11: Reserved
3:2	NRTP[1:0]	NOR region memory type 00: SRAM 01: PSRAM(CRAM) 10: NOR Flash(default after reset) 11: Reserved
1	NRMUX	NOR region memory address/data multiplexing 0: Disable address/data multiplexing function 1: Enable address/data multiplexing function
0	NRBKEN	NOR region enable 0: Disable the corresponding memory bank 1: Enable the corresponding memory bank

20.4.2. SRAM/NOR Flash timing configuration registers (EXMC_SNTCFG)

Address offset: 0x04

Reset value: 0x0FFF FFFF

This register has to be accessed by word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ASYNCMOD[1:0]		DLAT[3:0]				CKDIV[3:0]				BUSLAT[3:0]			
		rw		rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSET[7:0]								AHLD[3:0]				ASET[3:0]			
rw								rw				rw			

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29:28	ASYNCMOD[1:0]	Asynchronous access mode The bits are valid only when the EXMODEN bit in the EXMC_SNCTL register is 1. 00: Mode A access 01: Mode B access 10: Mode C access 11: Mode D access
27:24	DLAT[3:0]	Data latency for NOR Flash. Only valid in synchronous access. 0x0: Data latency of first burst access is 2 EXMC_CLK 0x1: Data latency of first burst access is 3 EXMC_CLK …… 0xF: Data latency of first burst access is 17 EXMC_CLK
23:20	CKDIV[3:0]	Synchronous clock divide ratio. This filed is only effect in synchronous mode. 0x0: Reserved 0x1: EXMC_CLK period = 2 * HCLK period …… 0xF: EXMC_CLK period = 16 * HCLK period
19:16	BUSLAT[3:0]	Bus latency The bits are defined in multiplexed read mode in order to avoid bus contention, and the bits represent the minimum time the data bus used to return to a high impedance state. 0x0: Bus latency = 1 * HCLK period 0x1: Bus latency = 2 * HCLK period …… 0xF: Bus latency = 16 * HCLK period
15:8	DSET[7:0]	Data setup time This field is meaningful only in asynchronous access.

		0x00: Reserved 0x01: Data setup time = 2 * HCLK period …… 0xFF: Data setup time = 256 * HCLK period
7:4	AHLD[3:0]	Address hold time This field is used to set the time of address hold phase, which is only used in mode D and multiplexed mode. 0x0: Reserved 0x1: Address hold time = 2 * HCLK …… 0xF: Address hold time = 16 * HCLK
3:0	ASET[3:0]	Address setup time This field is used to set the time of address setup phase. **Note:** Meaningful only in asynchronous access of SRAM, ROM, NOR Flash. 0x0: Address setup time = 1 * HCLK …… 0xF: Address setup time = 16 * HCLK

20.4.3. SRAM/NOR Flash write timing configuration registers (EXMC_SNWTCFG)

Address offset: 0x104

Reset value: 0x0FFF FFFF

This register is meaningful only when the EXMODEN bit in EXMC_SNCTL is set to 1.

This register has to be accessed by word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		WASYNCMOD[1:0]		Reserved								WBUSLAT[3:0]			
		rw										rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDSET[7:0]								WAHLD[3:0]				WASET[3:0]			
rw								rw				rw			

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29:28	WASYNCMOD[1:0]	Asynchronous access mode The bits are valid only when the EXMODEN bit in the EXMC_SNCTL register is 1. 00: Mode A access 01: Mode B access 10: Mode C access

		11: Mode D access
27:20	Reserved	Must be kept at reset value.
19:16	WBUSLAT[3:0]	Bus latency Bus latency is added at the end of each write transaction to meet the minimum time between consecutive transactions. 0x0: Bus latency = 1 * HCLK period 0x1: Bus latency = 2 * HCLK period …… 0xF: Bus latency = 16 * HCLK period
15:8	WDSET[7:0]	Data setup time This field is meaningful only in asynchronous access. 0x00: Reserved 0x01: Data setup time = 2 * HCLK period …… 0xFF: Data setup time = 256 * HCLK period
7:4	WAHLD[3:0]	Address hold time This field is used to set the time of address hold phase, which is only used in mode D and multiplexed mode. 0x0: Reserved 0x1: Address hold time = 2 * HCLK …… 0xF: Address hold time = 16 * HCLK
3:0	WASET[3:0]	Address setup time This field is used to set the time of address setup phase. **Note:** Meaningful only in asynchronous access of SRAM, ROM, NOR Flash. 0x0: Address setup time = 1 * HCLK 0x1: Address setup time = 2 * HCLK …… 0xF: Address setup time = 16 * HCLK

21. Controller area network (CAN)

21.1. Overview

CAN bus (Controller Area Network) is a bus standard designed to allow microcontrollers and devices to communicate with each other without a host computer.

As CAN network interface, basic extended CAN supports the CAN protocols version 2.0A, 2.0B, ISO11898-1:2015 and BOSCH CAN FD specification. The CAN interface automatically handles the transmission and the reception of CAN frames. The CAN provides 28 scalable/configurable identifier filter banks. The filters are used for selecting the input message as software requirement and otherwise discarding the message. Three transmit mailboxes are provided to the software for transfer messages. The transmission scheduler decides which mailbox will be transmitted firstly. Three complete messages can be stored in every FIFO. The FIFOs are managed completely by hardware. Two receiving FIFOs are used by hardware to store the incoming messages. In addition, the CAN controller provides all hardware functions, which supports the time-triggered communication option, in safety-critical applications.

21.2. Characteristics

- Supports CAN protocols version 2.0A, B.
- Supports CAN FD Frame with up to 64 data bytes (ISO11898-1 and Bosch CAN FD specification V1.0).
- Baud rates up to 1 Mbit/s when classical frames and 6 Mbit/s when FD frames.
- Supports transmitter delay compensation.
- Supports the time-triggered communication.
- Interrupt enable and clear.

Transmission

- Supports 3 transmit mailboxes.
- Supports priority of transmission message.
- Supports time stamp at SOF transmission.

Reception

- Supports 2 Rx FIFOs and each has 3 messages depth.
- 28 scalable/configurable identifier filter banks.
- FIFO lock.

Time-triggered communication

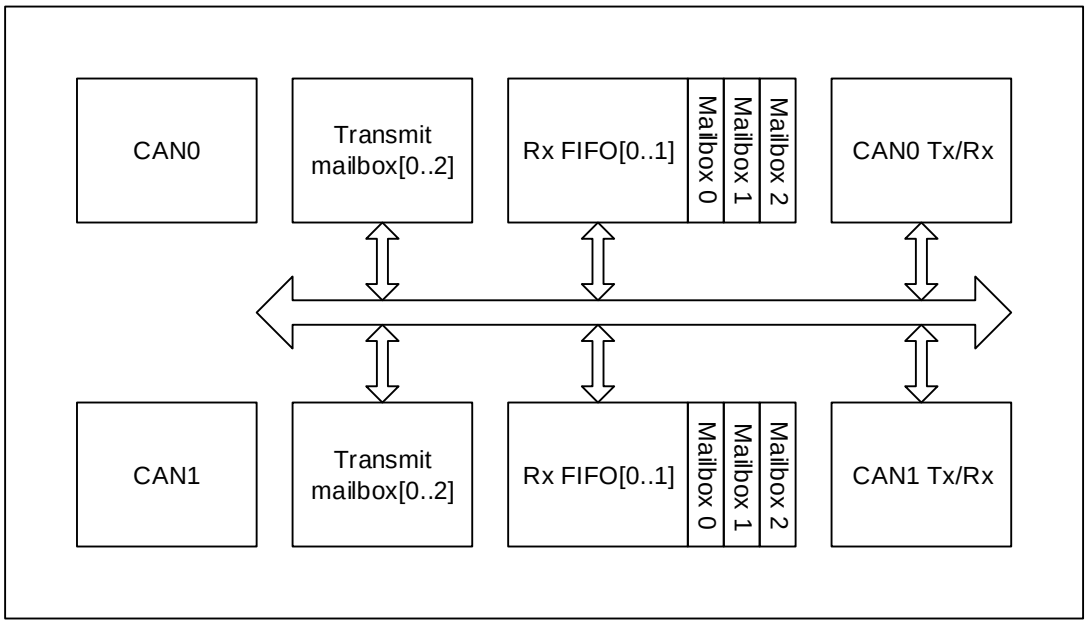
- Disable retransmission automatically in time-triggered communication mode.
- 16-bit free timer.
- Time stamp on SOF reception.

- Time stamp in last two data bytes transmission.

21.3. Function overview

Figure 21-1. CAN module block diagram shows the CAN block diagram.

Figure 21-1. CAN module block diagram



21.3.1. Working mode

The CAN interface has three working modes:

- Sleep working mode.
- Initial working mode.
- Normal working mode.

Sleep working mode

Sleep working mode is the default mode after reset. In sleep working mode, the CAN is in the low-power status and the CAN clock is stopped.

When SLPWMOD bit in CAN_CTL register is set, the CAN enters the sleep working mode. Then the SLPWS bit in CAN_STAT register is set by hardware.

To leave sleep working mode automatically: the AWU bit in CAN_CTL register is set and the CAN bus activity is detected. To leave sleep working mode by software: clear the SLPWMOD bit in CAN_CTL register.

Sleep working mode to initial working mode: set IWMOD bit and clear SLPWMOD bit in CAN_CTL register.

Sleep working mode to normal working mode: clear IWMOD and SLPWMOD bit in CAN_CTL

register.

Initial working mode

When the configuration of CAN bus communication is needed to be changed, the CAN must enter initial working mode.

When IWMOD bit in CAN_CTL register is set, the CAN enters the initial working mode. Then the IWS bit in CAN_STAT register is set.

Initial working mode to sleep working mode: set SLPWMOD bit and clear IWMOD bit in CAN_CTL register.

Initial working mode to normal working mode: clear IWMOD bit and clear SLPWMOD bit in CAN_CTL register.

Normal working mode

The CAN could communicate with other CAN communication nodes in normal working mode

To enter normal working mode: clear IWMOD and SLPWMOD bit in CAN_ CTL register.

Normal working mode to sleep working mode: set SLPWMOD bit in CAN_CTL register and wait the current transmission or reception completed.

Normal working mode to initial working mode: set IWMOD bit in CAN_CTL register, and wait the current transmission or reception completed.

21.3.2. Communication modes

The CAN interface has four communication modes:

- Silent communication mode.
- Loopback communication mode.
- Loopback and silent communication mode.
- Normal communication mode.

Silent communication mode

Silent communication mode means reception available and transmission disable.

The RX pin of the CAN could detect the signal from the network and the TX pin always holds in recessive state.

When the SCMOD bit in CAN_BT register is set, the CAN enters the silent communication mode. When it is cleared, the CAN leaves silent communication mode.

Silent communication mode is useful for monitoring the network messages.

Loopback communication mode

Loopback communication mode means the transmitted messages are transferred into the Rx FIFOs, the RX pin is disconnected from the CAN network and the TX pin can still send messages to the CAN network.

Setting LCMOD bit in CAN_BT register to enter loopback communication mode, while clearing it to leave. Loopback communication mode is useful for self-test.

Loopback and silent communication mode

Loopback and silent communication mode means the RX and TX pins are disconnected from the CAN network while the transmitted messages are transferred into the Rx FIFOs.

Setting LCMOD and SCMOD bit in CAN_BT register to enter loopback and silent communication mode, while clearing them to leave.

Loopback and silent communication mode is used for self-test. The TX pin holds in recessive state. The RX pin holds in high impedance state.

Normal communication mode

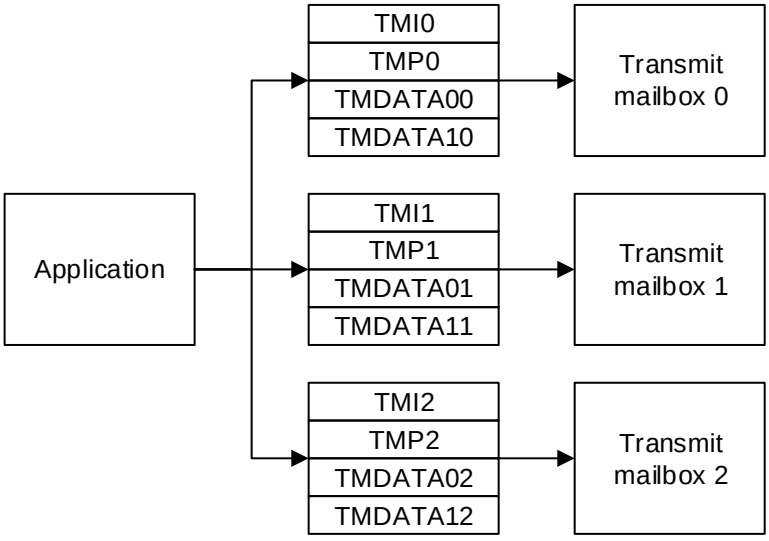
Normal communication mode is the default communication mode when the LCMOD and SCMOD bits in CAN_BT register are cleared.

21.3.3. Data transmission

Transmission register

Three transmit mailboxes are used for the application. Transmit mailboxes are used by configuring four transmission registers: CAN_TMIx, CAN_TMPx, CAN_TMDATA0x and CAN_TMDATA1x. As is shown in ***Figure 21-2. Transmission register***.

Figure 21-2. Transmission register

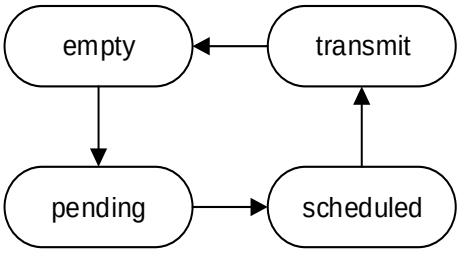


If FD frame would be transmitted, always write TMDATA00 registers when mailbox 0 is used, TMDATA01 register when mailbox 1 is used and TMDATA02 register when mailbox 2 is used until the end. For example, if application wants to transmit 64 bytes data using mailbox0, it needs to write the 64 bytes data through TMDATA00 register for 16 times. The data is stored in internal SRAM.

Transmit mailbox state

A transmit mailbox can be used when it is free (**empty state**). If the mailbox is filled with data, set TEN bit in CAN_TMIx register to prepare for starting the transmission **(pending state**). If more than one mailbox is in the pending state, they need scheduling the transmission (**scheduled state**). A mailbox with highest priority enters into transmit state and starts transmitting the message (**transmit state**). After the message has been sent, the mailbox is free (**empty state**). As is shown in ***Figure 21-3. State of transmit mailbox***.

Figure 21-3. State of transmit mailbox



Transmit status and error

The CAN_TSTAT register includes the transmit status and error bits: MTF, MTFNERR, MAL, MTE.

- MTF: mailbox transmit finished. Typically, MTF is set when the frame in the transmit mailbox has been sent.
- MTFNERR: mailbox transmit finished with no error. MTFNERR is set when the frame in the transmit mailbox has been sent without any error.
- MAL: mailbox arbitration lost. MAL is set when the frame transmission is failed due to the arbitration lost.
- MTE: mailbox transmit error. MTE is set when the frame transmission is failed due to the error detected on the CAN bus.

Steps of sending a frame

To send a frame through the CAN:

Step 1: Select one free transmit mailbox.

Step 2: Configure four transmission registers with the application's acquirement.

Step 3: Set TEN bit in CAN_TMIx register.

Step 4: Check the transmit status. Typically, MTF and MTFNERR are set if transmission is

successful.

Transmission options

Abort

MST bit in CAN_TSTAT register can abort the transmission.

If the transmit mailbox's status is **pending** or **scheduled**, the abort of transmission can be done immediately.

In the **transmit** state, the abort of transmission does not take effect immediately until the transmission is finished. In case that the transmission is successful, the MTFNERR and MTF in CAN_TSTAT are set and state changes to be **empty**. In case that the transmission is failed, the state changes to be **scheduled** and then the abort of transmission can be done immediately.

Priority

When more than one transmit mailbox is pending, the transmission order is given by the TFO bit in CAN_CTL register.

In case that TFO is 1, the three transmit mailboxes work first-in first-out (FIFO).

In case that TFO is 0, the transmit mailbox with lowest identifier has the highest priority of transmission. If the identifiers are equal, the lower mailbox number will be scheduled firstly.

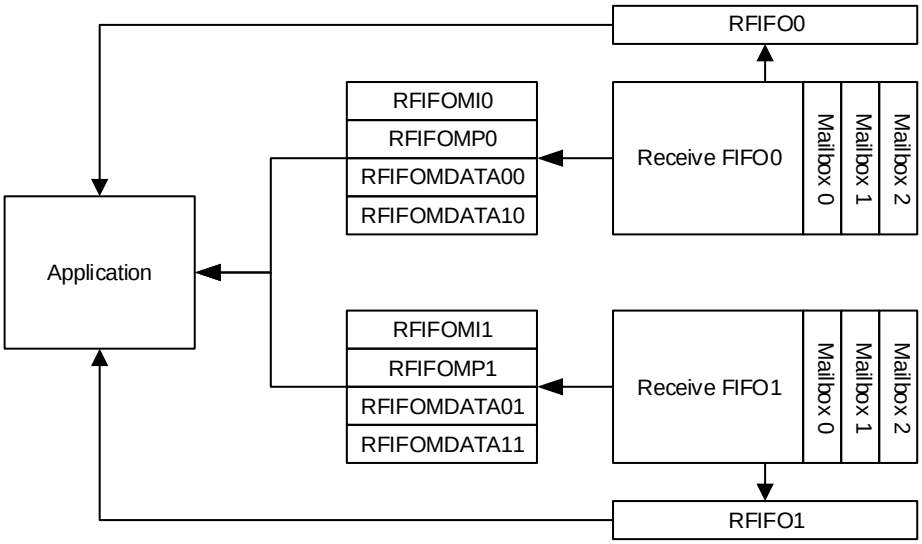
21.3.4. Data reception

Reception register

Two Rx FIFOs are used for the application. Rx FIFOs are managed by five registers: CAN_RFIFOx, CAN_RFIFOMIx, CAN_RFIFOMPx, CAN_RFIFOMDATA0x and CAN_RFIFOMDATA1x. FIFO's status and operation can be handled by CAN_RFIFOx register. Reception frame data can be achieved through the registers: CAN_RFIFOMIx, CAN_RFIFOMPx, CAN_RFIFOMDATA0x and CAN_RFIFOMDATA1x.

Each FIFO consists of three receive mailboxes. As is shown in ***Figure 21-4. Reception register***.

Figure 21-4. Reception register



Rx FIFO

Rx FIFO has three mailboxes. The reception frames are stored in the mailbox according to the arriving sequence. First arrived frame can be accessed by application firstly.

The number of frames in the Rx FIFO and the status can be accessed by the register CAN_RFIFO0 and CAN_RFIFO1.

If at least one frame has been stored in the Rx FIFO0, the frame data is stored in the CAN_RFIFOMI0, CAN_RFIFOMP0, CAN_RFIFOMDATA00 and CAN_RFIFOMDATA10 registers. After reading the current frame, set RFD bit in CAN_RFIFO0 to release a frame in the Rx FIFO and the software can read the next frame.

If FD frame has been received, the data always read from CAN_RFIFOMDATA00 register for FIFO0 and CAN_RFIFOMDATA01 for FIFO1 until the end. For example, if application needs to read 64 bytes data from FIFO0. It needs to read the 64 bytes data through CAN_RFIFOMDATA00 register for 16 times. The received data is stored in internal SRAM.

Rx FIFO status

RFL (Rx FIFO length) bits in CAN_RFIFOx register is 0 when no frame is stored in the Rx FIFO and it is 3 when FIFOx is full.

When RFF bit in CAN_RFIFOx register is set, it indicates FIFOx is full, at this time, RFL is 3.

When a new frame arrives after the FIFO has held three frames, the RFO bit in CAN_RFIFOx register will be set, and it indicates FIFOx is overrun. If the RFOD bit in CAN_CTL register is set, the new frame is discarded. If the RFOD bit in CAN_CTL register is reset, the new frame is stored into the Rx FIFO and the last frame in the Rx FIFO is discarded.

Steps of receiving a message

Step 1: Check the number of frames in the Rx FIFO.

Step 2: Read CAN_RFIFOMIx, CAN_RFIFOMPx, CAN_RFIFOMDATA0x and CAN_RFIFOMDATA1x.

Step 3: Set the RFD bit in CAN_RFIFOx register.

21.3.5. Filtering function

The CAN receives frames from the CAN bus. If the frame passes the filter, it is stored in the Rx FIFOs. Otherwise, the frame will be discarded without intervention by the software.

The identifier of frame is used for the matching of the filter.

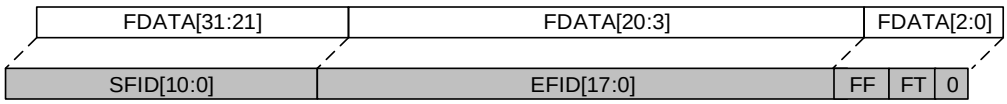
Scale

The filter consists of 28 banks: bank0 to bank27. Each bank has two 32-bit registers: CAN_FxDATA0 and CAN_FxDATA1.

Each filter bank can be configured to 32-bit or 16-bit.

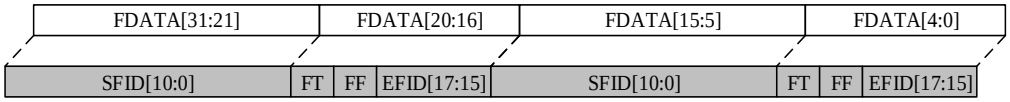
32-bit: SFID[10:0], EFID[17:0], FF and FT bits. As is shown in ***Figure 21-5. 32-bit filter***.

Figure 21-5. 32-bit filter



16-bit: SFID[10:0], FT, FF and EFID[17:15] bits. As is shown in ***Figure 21-6. 16-bit filter***.

Figure 21-6. 16-bit filter



Mask mode

For the Identifier of a data frame to be filtered, the mask mode is used to specify which bits must be the same as the preset Identifier and which bits need not be judged. 32-bit mask mode example is shown in ***Figure 21-7. 32-bit mask mode filter***.

Figure 21-7. 32-bit mask mode filter

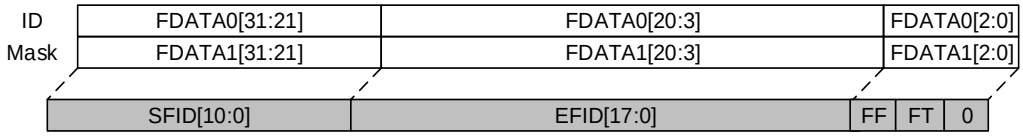
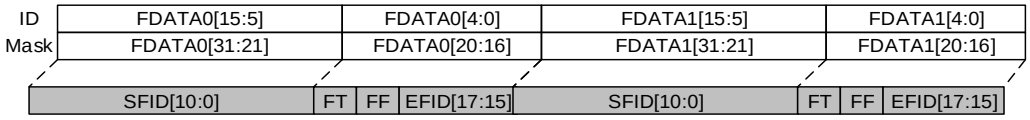


Figure 21-8. 16-bit mask mode filter



List mode

The filter consists of frame identifiers. The filter can determine whether a frame will be discarded or not. When one frame arrived, the filter will check which member can match the identifier of the frame.

32-bit list mode example is shown in ***Figure 21-9. 32-bit list mode filter***.

Figure 21-9. 32-bit list mode filter

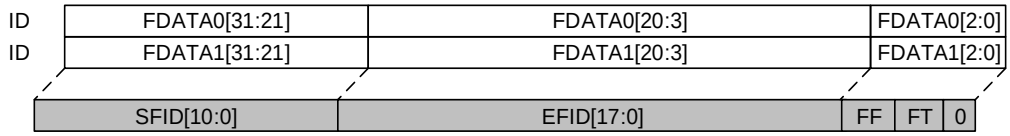
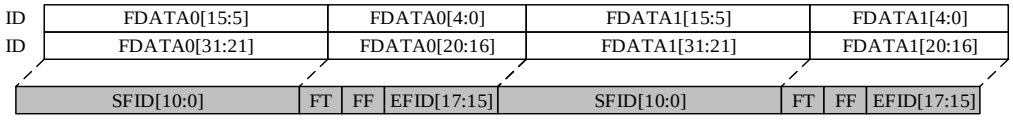


Figure 21-10. 16-bit list mode filter



Filter number

Filter consists of some filter bank. According to the mode and the scale of each of the filter banks, filter has different effects.

For example, there are two filter banks. Bank0 is configured as 32-bit mask mode. Bank1 is configured as 32-bit list mode. The filter number is shown in ***Table 21-1. 32-bit filter number***.

Table 21-1. 32-bit filter number

Filter bank	Filter data register	Filter number
0	F0DATA0-32bit-ID	0
	F0DATA1-32bit-Mask	
1	F1DATA0-32bit-ID	1
	F1DATA1-32bit-ID	2

Associated FIFO

28 banks can be associated with FIFO0 or FIFO1. If the bank is associated with FIFO0, the frames passed the bank will be stored in the FIFO0.

Active

The filter bank needs to be activated if the bank is to be used, otherwise, the filter bank should

be left deactivated.

Filtering index

Each filter number corresponds to a filtering rule. When the frame which is associated with a filter number N passes the filters, the filter index is N. It stores in the FI bits in CAN_RFIFOMPx.

Filter bank has filter index once it is associated with the FIFO no matter whether the bank is active or not.

The example about filtering index is shown in ***Table 21-2. Filtering index***.

Table 21-2. Filtering index

Filter bank	FIFO0	Active	Filter nunber	Filter bank	FIFO1	Active	Filter nunber
0	F0DATA0-32bits-ID	Yes	0	2	F2DATA0[15:0]-16bits-ID	Yes	0
	F0DATA1-32bits-Mask				F2DATA0[31:16]-16bits-Mask		
1	F1DATA0-32bits-ID	Yes	1		F2DATA1[15:0]-16bits-ID		1
	F1DATA1-32bits-ID		2		F2DATA1[31:16]-16bits-Mask		
3	F3DATA0[15:0]-16bits-ID	No	3	4	F4DATA0-32bits-ID	No	2
	F3DATA0[31:16]-16bits-Mask				F4DATA1-32bits-Mask		
	F3DATA1[15:0]-16bits-ID		4	5	F5DATA0-32bits-ID	No	3
	F3DATA1[31:16]-16bits-Mask				F5DATA1-32bits-ID		4
7	F7DATA0[15:0]-16bits-ID	No	5	6	F6DATA0[15:0]-16bits-ID	Yes	5
	F7DATA0[31:16]-16bits- ID		6		F6DATA0[31:16]-16bits- ID		6
	F7DATA1[15:0]-16bits-ID		7		F6DATA1[15:0]-16bits-ID		7
	F7DATA1[31:16]-16bits- ID		8		F6DATA1[31:16]-16bits- ID		8
8	F8DATA0[15:0]-16bits-ID	Yes	9	10	F10DATA0[15:0]-16bits-ID	No	9
	F8DATA0[31:16]-16bits- ID		10		F10DATA0[31:16]-16bits-Mask		
	F8DATA1[15:0]-16bits-ID		11		F10DATA1[15:0]-16bits-ID		10
	F8DATA1[31:16]-		12		F10DATA1[31:16]-		

Filter bank	FIFO0	Active	Filter nunber	Filter bank	FIFO1	Active	Filter nunber
	16bits- ID				16bits-Mask		
9	F9DATA0[15:0]-16bits-ID	Yes	13	11	F11DATA0[15:0]-16bits-ID	No	11
	F9DATA0[31:16]-16bits-Mask				F11DATA0[31:16]-16bits- ID		12
	F9DATA1[15:0]-16bits-ID		14		F11DATA1[15:0]-16bits-ID		13
	F9DATA1[31:16]-16bits-Mask				F11DATA1[31:16]-16bits- ID		14
12	F12DATA0-32bits-ID	Yes	15	13	F13DATA0-32bits-ID	Yes	15
	F12DATA1-32bits-Mask				F13DATA1-32bits-ID		16

Priority

The filters have the priority rules:

1. 32-bits mode is higher than 16-bits mode.
2. List mode is higher than mask mode.
3. Smaller filter number has the higher priority.

21.3.6. Time-triggered communication

The time-triggered CAN protocol is a higher layer protocol on top of the CAN data link layer. Time-triggered communication means that activities are triggered by the elapsing of time segments. In a time-triggered communication system, all time points of message transmission are pre-defined.

In this mode, an internal 16-bit counter starts working, incrementing by 1 at each CAN bit time. This internal counter provides time stamps for sending and receiving data, stored in registers CAN _RFIFOMPx and CAN_TMPx.

The automatic retransmission is disabled in the time-triggered CAN communication.

21.3.7. Communication parameters

Automatic retransmission forbid mode

In time-triggered communication mode, the requirement for automatic retransmission must be disabled and CAN be met by setting ARD position 1 of the CAN_CTL register.

In this mode, the data is sent only once, and if the transmission fails due to arbitration failure or bus error, the CAN bus controller does not automatically resend the data as usual.

At the end of sending, the MTF bit of register CAN_TSTAT is hardware set to 1, and the sending status information can be obtained via MTFNERR, MAL, and MTE.

Bit time

On the bit-level, the CAN protocol uses synchronous bit transmission. This not only enhances the transmitting capacity but also requires a sophisticated method of bit synchronization. While bit synchronization in a character-oriented transmission (asynchronous) is performed upon the reception which the start bit is available with each character, the synchronous transmission protocol just need one start bit available at the beginning of a frame. To ensure that the receiver correctly reads the messages, resynchronization is required. Phase buffer segments' sample point of the front-end and back-end should be inserted a bit interval.

The CAN protocol regulates bus access by bit-wise arbitration. The signal propagated from sender to receiver and back to the sender must be completed within one bit-time. For synchronization, in addition to the phase buffer segments, a propagation delay segment is needed. The propagation delay segment is regarded as signal delays caused by transmitting and receiving nodes in the process of the signal propagation on the bus.

The normal bit time from the CAN protocol has three segments as follows:

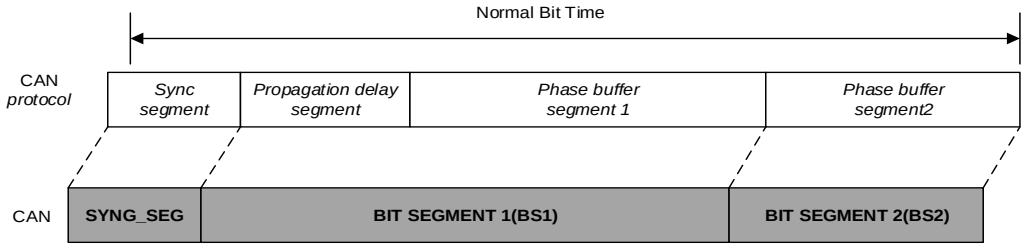
Synchronization segment (SYNC_SEG): a bit change is expected to occur within this time segment. It has a fixed length of one time quantum ($1 \times t_{CAN}$).

Bit segment 1 (BS1): It defines the location of the sample point. It includes the Propagation delay segment and Phase buffer segment 1 in the CAN standard. It may be automatically lengthened to compensate for positive phase drifts due to different frequency of the various nodes of the network.

Bit segment 2 (BS2): It defines the location of the transmit point. It represents the Phase buffer segment 2 in the CAN standard. It may also be automatically shortened to compensate for negative phase drifts.

The bit time is shown as in the ***Figure 21-11. The bit time***.

Figure 21-11. The bit time



The resynchronization Jump Width (SJW): it can be lengthened or shortened to compensate for the Synchronization error of the CAN network node.

The valid edge transition is defined as the first transition from a recessive bit to a dominant

bit within a bit time, provided that the bus state detected at the previous sample point was recessive.

If a valid edge is detected in BS1, not in SYNC_SEG, BS1 is added up to SJW maximumly, so that the sample point is delayed.

Conversely, if a valid edge is detected in BS2, not in SYNC_SEG, BS2 is cut down to SJW at most, so that the transmit point is moved earlier.

Note: For detailed explanations of hard synchronization and resynchronization, please refer to the ISO 11898 standard

Baud rate

The clock of the CAN derives from the APB1 bus. The CAN calculates its baud rate as follow:

$$BaudRate=\frac{1}{Normal\ Bit\ Time} \quad (21\text{-}1)$$

$$Normal\ Bit\ Time=t_{SYNC_SEG}+t_{BS1}+t_{BS2} \quad (21\text{-}2)$$

with:

$$t_{SYNC_SEG}=1\times t_q \quad (21\text{-}3)$$

$$t_{BS1}=(1+BT.BS1)\times t_q \quad (21\text{-}4)$$

$$t_{BS2}=(1+BT.BS2)\times t_q \quad (21\text{-}5)$$

$$t_q=(1+BT.BAUDPSC)\times t_{PCLK1} \quad (21\text{-}6)$$

21.3.8. CAN FD operation

The CAN FD function is enabled by setting FDEN to 1 in CAN_FDCTL register. If FDEN bit is cleared, only calassical frames are supported. If FDEN bit is set, it supports calassical frames and FD frames. Whether the current frame is FD or not could be defined by received FDF bit (the previously reserved bit in CAN frames with 11-bit identifiers or the first previously reserved bit in CAN frames with 29-bit identifiers which now is decoded as FDF bit). If FDF bit is recessive, meaning to be the CAN FD frame, otherwise FDF bit is dominant, meaning to be the classical frame.

The CAN FD supports ISO11898-1 or Bosch CAN FD Specification V1.0 by configuring NISO bit in CAN_FDCTL register.

The two bits following the FDF bit are reserved bit and BRS bit respectively. The received BRS bit determines the bit rate of data. When BRS is dominant, bit rate of data could not switch by configuring the CAN_DBT register. When BRS is recessive, bit rate of data could switch to a higher bit rate inside the frame by configuring the CAN_DBT register. The bit rate of data can be switched in the period from BRS bit to CRC delimiter (refer to ISO11898-1 or Bosch CAN FD Specification V1.0).

When Protocol Exception Handling is enabled (PRED bit in CAN_FDCTL register is 0), it causes the operation state to change to be IDLE and interrupts current frame at the next sample point when recessive reserve bit is received. When Protocol Exception Handling is disabled (PRED bit in CAN_FDCTL register is 1), it will treat a recessive reserve bit as a form error and respond with an error frame. If any recessive reserve bit occurs, set PRE bit in CAN_FDSTAT register to 1.

The transmission of ESI bit (the bit before DLC bits, refer to ISO11898-1 or Bosch CAN FD Specification V1.0) is defined by ESIMOD bit in CAN_FDCTL register and ESI bit in CAN_TMPx register. If ESIMOD bit is 0, it will transmit the dominant bit by error active nodes and transmit the recessive bit by error passive nodes. If ESIMOD bit is set, it will transmit ESI bit in CAN_TMPx register.

The transmission of FDF bit and BRS bit is defined by FDF bit and BRS bit in CAN_TMPx registers.

21.3.9. Transmitter Delay Compensation

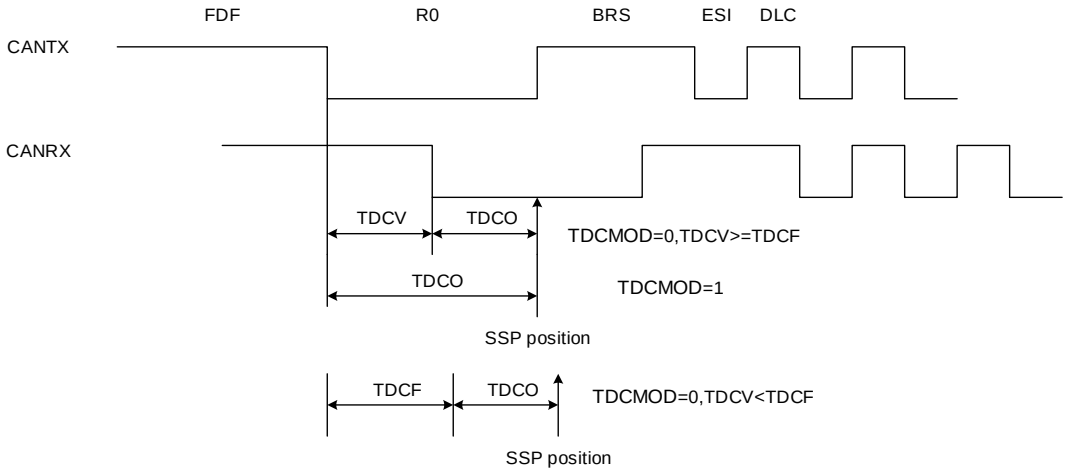
CAN FD supports transmitter delay compensation mechanism, it could be used in applications when the length of the CAN bit time in the DATA-PHASE is shorter than the limit required by the transceiver's internal delay time. The description of transmitter delay compensation, please refer to ISO11898-1 or Bosch CAN FD Specification V1.0.

The transmitter delay compensation is enabled by setting TDCEN bit in CAN_FDCTL register.

The transmitter delay compensation is used to adjust the position of the SSP. The SSP delay is defined as the delay from dominant edges on CANTX to SSP point. If TDCMOD bit in CAN_FDCTL register is set, the SSP delay is defined in TDCO bits of the CAN_FDTDC registers by software. If TDCMOD bit in CAN_FDCTL register is cleared, the hardware automatically uses the falling edges between FDF bit and RES0 bit to calculate the delay from dominant edges on CANTX to dominant edges on CANRX, and store the delay in TDCV bits of CAN_FDSTAT registers. In case that there is dominant glitch, SSP position would be advanced than expected, leading to a calculated error in compensation measurement. To solve this problem, TDCF bits of CAN_FDTDC register could be used to avoid too small TDCV bits. So the value of SSP delay is TDCO bits add TDCV bits if TDCV is larger than TDCF, or else the value of SSP delay is TDCO bits add TDCF bits.

The SSP delay can not exceed 3 data bit time.

Figure 21-12. Transmitter Delay Measurement



21.3.10. Error flags

The state of CAN bus can be reflected by Transmit Error Counter (TECNT) and Receive Error Counter (RECNT) of CAN_ERR register. The value can be increased or decreased by the hardware according to the error, and the software can judge the stability of the CAN network by these values. For details on incorrect counting, refer to the CAN protocol section.

Their values which read by software determine whether the CAN network is stable.By using the CAN_INTEN register (ERRIE bit, etc.), the software can control the interrupt generation when error is detected.

Bus-Off recovery

The CAN controller is in Bus-Off state when TECNT is over than 255. In This state, BOERR bit is set in CAN_ERR register, and no longer able to transmit and receive messages.

According to the ABOR configuration in register CAN_CTL, there are two ways to recover from Bus-Off (to an Error active state). Both of these methods require the CAN bus controller in the offline state to detect the Bus-Off recovery sequence defined by CAN protocol (when CAN_RX detects 128 consecutive 11-bit recessive bits) before automatic recovery.

If ABOR is set, it will be automatically recovered when a Bus-Off recovery sequence is detected.

If ABOR is cleared, CAN controller must be configured to enter initialization mode by setting IWMOD bit in CAN_CTL register, then exit and enter nomal mode. After this operation, it will recover when the recovering sequence is detected.

21.3.11. CAN interrupts

The CAN bus controller occupies 4 interrupt vectors, which are controlled by the register CAN _INTEN.

The interrupt sources can be classified as:

- Transmit interrupt.
- FIFO0 interrupt.
- FIFO1 interrupt.
- Error and status change interrupt.

Transmit interrupt

The transmit interrupt can be generated by any of the following conditions and TMEIE bit in CAN_INTEN register will be set:

- TX mailbox 0 transmit finished: MTF0 bit in the CAN_TSTAT register is set.
- TX mailbox 1 transmit finished: MTF1 bit in the CAN_TSTAT register is set.
- TX mailbox 2 transmit finished: MTF2 bit in the CAN_TSTAT register is set.

Rx FIFO0 interrupt

The Rx FIFO0 interrupt can be generated by the following conditions:

- Rx FIFO0 not empty: RFL0 bits in the CAN_RFIFO0 register are not ‘00’ and RFNEIE0 in CAN_INTEN register is set.
- Rx FIFO0 full: RFF0 bit in the CAN_RFIFO0 register is set and RFFIE0 in CAN_INTEN register is set.
- Rx FIFO0 overrun: RFO0 bit in the CAN_RFIFO0 register is set and RFOIE0 in CAN_INTEN register is set.

Rx FIFO1 interrupt

The Rx FIFO1 interrupt can be generated by the following conditions:

- Rx FIFO1 not empty: RFL1 bits in the CAN_RFIFO1 register are not ‘00’ and RFNEIE1 in CAN_INTEN register is set.
- Rx FIFO1 full: RFF1 bit in the CAN_RFIFO1 register is set and RFFIE1 in CAN_INTEN register is set.
- Rx FIFO1 overrun: RFO1 bit in the CAN_RFIFO1 register is set and RFOIE1 in CAN_INTEN register is set.

Error and working mode change interrupt

The error and working mode change interrupt can be generated by the following conditions:

- Error: ERRIF bit in the CAN_STAT register and ERRIE bit in the CAN_INTEN register are set. Refer to ERRIF description in the CAN_STAT register.
- Wakeup: WUIF bit in the CAN_STAT register is set and WIE bit in the CAN_INTEN register is set.
- Enter sleep working mode: SLPIF bit in the CAN_STAT register is set and SLPWIE bit in the CAN_INTEN register is set.

The CAN bus controller interrupt conditions can refer to ***Table 21-3. CAN Event / Interrupt flags***.

Table 21-3. CAN Event / Interrupt flags

Interrupt event	Interrupt / Event flag		Enable control bit	
Transmit interrupt	Mailbox 0 transmit finished flag (MTF0)		TMEIE	
	Mailbox 1 transmit finished flag (MTF1)			
	Mailbox 2 transmit finished flag (MTF2)			
FIFO0 interrupt	Rx FIFO0 length (RFL0[1:0])		RFNEIE0	
	Rx FIFO0 full (RFF0)		RFFIE0	
	Rx FIFO0 overfull (RFO0)		RFOIE0	
FIFO1 interrupt	Rx FIFO1 length (RFL0[1:0])		RFNEIE1	
	Rx FIFO1 full (RFF0)		RFFIE1	
	Rx FIFO1 overfull (RFO0)		RFOIE1	
EWMC interrupt	Warning error (WERR)	Error interrupt flag (ERRIF)	WERRIE	ERRIE
	Passive error (PERR)		PERRIE	
	Bus-Off error (BOERR)		BOIE	
	Error number (1<= ERRN[2:0] <= 6)		ERRNIE	
	Status change interrupt flag of waking up from sleep working mode (WUIF)		WIE	
	Status change interrupt flag of entering sleep working mode (SLPIF)		SLPWIE	

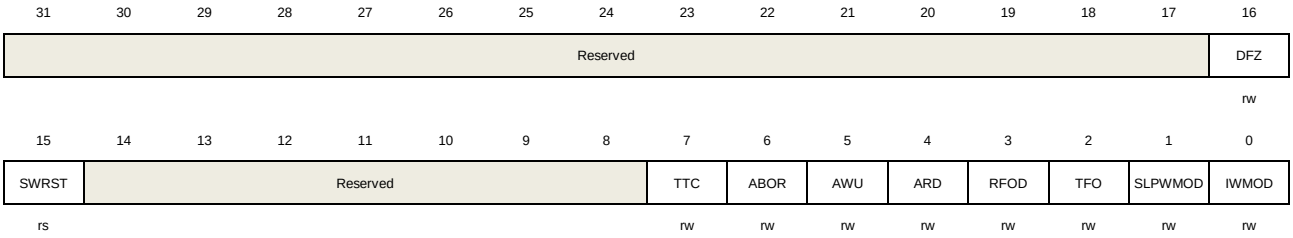
21.4. Register definition

CAN0 base address: 0x4000 6400
CAN1 base address: 0x4000 6800

21.4.1. Control register (CAN_CTL)

Address offset: 0x00
Reset value: 0x0001 0002

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	DFZ	Debug freeze If the CANx_HOLD in DBG_CTL register is set, this bit defines the CAN controller is in debug freezing mode or normal working mode. If the CANx_HOLD in DBG_CTL register is cleared, this bit takes no effect. 0: CAN reception and transmission work normal even during debug 1: CAN reception and transmission stop working during debug
15	SWRST	Software reset 0: No effect 1: Reset CAN to enter sleep working mode. This bit is automatically reset to 0.
14:8	Reserved	Must be kept at reset value.
7	TTC	Time-triggered communication 0: Disable time-triggered communication 1: Enable time-triggered communication
6	ABOR	Automatic Bus-Off recovery 0: The Bus-Off state is left manually by software 1: The Bus-Off state is left automatically by hardware
5	AWU	Automatic wakeup If this bit is set, the CAN leaves sleep working mode when CAN bus activity is detected, and SLPWMOD bit in CAN_CTL register will be cleared automatically. 0: The sleeping working mode is left manually by software 1: The sleeping working mode is left automatically by hardware

4	ARD	Automatic retransmission disable 0: Enable automatic retransmission 1: Disable automatic retransmission
3	RFOD	Rx FIFO overwrite disable 0: Enable Rx FIFO overwrite when Rx FIFO is full and overwrite the FIFO with the incoming frame 1: Disable Rx FIFO overwrite when Rx FIFO is full and discard the incoming frame
2	TFO	Tx FIFO order 0: Order with the identifier of the frame (the smaller identifier has higher priority) 1: Order with first-in and first-out
1	SLPWMOD	Sleep working mode If this bit is set by software, the CAN enters sleep working mode after current transmission or reception is completed. This bit can be cleared by software or hardware. If AWU bit in CAN_CTL register is set, this bit is cleared by hardware when CAN bus activity is detected. 0: Disable sleep working mode 1: Enable sleep working mode
0	IWMOD	Initial working mode 0: Disable initial working mode 1: Enable initial working mode

21.4.2. Status register (CAN_STAT)

Address offset: 0x04

Reset value: 0x0000 0C02

This register has to be accessed by word(32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				RXL	LASTRX	RS	TS	Reserved			SLPIF	WUIF	ERRIF	SLPWS	IWS
				r	r	r	r				rc_w1	rc_w1	rc_w1	r	r

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11	RXL	RX level
10	LASTRX	Last sample value of RX pin
9	RS	Receiving state 0: CAN is not working in the receiving state

		1: CAN is working in the receiving state
8	TS	Transmitting state 0: CAN is not working in the transmitting state 1: CAN is working in the transmitting state
7:5	Reserved	Must be kept at reset value.
4	SLPIF	Status change interrupt flag of entering sleep working mode This bit is set by hardware when entering sleep working mode, and cleared by hardware when the CAN is not in sleep working mode. This bit can also be cleared by software when writting 1 to this bit. 0: CAN is not in the sleep working mode 1: CAN is in the sleep working mode
3	WUIF	Status change interrupt flag of waking up from sleep working mode This bit is set when CAN bus activity event is detected in sleep working mode. This bit can be cleared by software when writting 1 to this bit. 0: Wakeup event is not coming 1: Wakeup event is coming
2	ERRIF	Error interrupt flag This bit is set by the following events. The BOERR bit in CAN_ERR register is set and BOIE bit in CAN_INTEN register is set. Or the PERR bit in CAN_ERR register is set and PERRIE bit in CAN_INTEN register is set. Or the WERR bit in CAN_ERR register is set and WERRIE bit in CAN_INTEN register is set. Or the ERRN bits in CAN_ERR register are set to 1 to 6 (not 0 and not 7) and ERRNIE in CAN_INTEN register is set. This bit is cleared by software when writting 1 to this bit. 0: No error interrupt event 1: Any error interrupt event has happened
1	SLPWS	Sleep working state This bit is set by hardware when the CAN enters sleep working mode after setting SLPWMOD bit in CAN_CTL register. If the CAN leaves normal working mode to sleep working mode, it must wait the current frame transmission or reception to be completed. This bit is cleared by hardware when the CAN leaves sleep working mode. Clear SLPWMOD bit in CAN_CTL register or automatically detect the CAN bus activity when AWU bit is set in CAN_CTL register. If leaving sleep working mode to normal working mode, this bit will be cleared after receiving 11 consecutive recessive bits from the CAN bus. 0: CAN is not in the state of sleep working mode 1: CAN is in the state of sleep working mode
0	IWS	Initial working state This bit is set by hardware when the CAN enters initial working mode after setting IWMOD bit in CAN_CTL register. If the CAN leaves normal working mode to initial working mode, it must wait the current frame transmission or reception to be completed. This bit is cleared by hardware when the CAN leaves initial working

mode after clearing IWMOD bit in CAN_CTL register. If leaving initial working mode to normal working mode, this bit will be cleared after receiving 11 consecutive recessive bits from the CAN bus.
0: CAN is not in the state of initial working mode
1: CAN is in the state of initial working mode

21.4.3. Transmit status register (CAN_TSTAT)

Address offset: 0x08
Reset value: 0x1C00 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TMLS2	TMLS1	TMLS0	TME2	TME1	TME0	NUM[1:0]		MST2	Reserved			MTE2	MAL2	MTFNERR2	MTF2
r	r	r	r	r	r	r		rs				rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MST1	Reserved			MTE1	MAL1	MTFNERR1	MTF1	MST0	Reserved			MTE0	MAL0	MTFNERR0	MTF0
rs				rc_w1	rc_w1	rc_w1	rc_w1	rs				rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31	TMLS2	Transmit mailbox 2 last sending in Tx FIFO This bit is set by hardware when transmit mailbox 2 has the last sending order in the Tx FIFO with at least two frames pending.
30	TMLS1	Transmit mailbox 1 last sending in Tx FIFO This bit is set by hardware when transmit mailbox 1 has the last sending order in the Tx FIFO with at least two frames pending.
29	TMLS0	Transmit mailbox 0 last sending in Tx FIFO This bit is set by hardware when transmit mailbox 0 has the last sending order in the Tx FIFO with at least two frames pending.
28	TME2	Transmit mailbox 2 empty 0: Transmit mailbox 2 not empty 1: Transmit mailbox 2 empty
27	TME1	Transmit mailbox 1 empty 0: Transmit mailbox 1 not empty 1: Transmit mailbox 1 empty
26	TME0	Transmit mailbox 0 empty 0: Transmit mailbox 0 not empty 1: Transmit mailbox 0 empty
25:24	NUM[1:0]	These bits are the number of the Tx FIFO mailbox in which the frame will be transmitted if at least one mailbox is empty. These bits are the number of the Tx FIFO mailbox in which the frame will be

		transmitted at last if all mailboxes are full.
23	MST2	Mailbox 2 stop transmitting This bit is set by the software to stop mailbox 2 transmitting. This bit is reset by the hardware while the mailbox 2 is empty.
22:20	Reserved	Must be kept at reset value.
19	MTE2	Mailbox 2 transmit error This bit is set by hardware when the transmit error occurs. This bit is reset by writing 1 to this bit or MTF2 bit in CAN_TSTAT register. This bit is reset by hardware when next transmit starts.
18	MAL2	Mailbox 2 arbitration lost This bit is set when the arbitration lost occurs. This bit is reset by writting 1 to this bit or MTF2 bit in CAN_TSTAT register. This bit is reset by hardware when next transmit starts.
17	MTFNERR2	Mailbox 2 transmit finished with no error This bit is set when the transmission finishes and no error occurs. This bit is reset by writting 1 to this bit or MTF2 bit in CAN_TSTAT register. This bit is reset by hardware when the transmission finishes with error. 0: Mailbox 2 transmit finished with error 1: Mailbox 2 transmit finished with no error
16	MTF2	Mailbox 2 transmit finished This bit is set by hardware when the transmission finishes or aborts. This bit is reset by writting 1 to this bit or TEN bit in CAN_TMI2 is 1. 0: Mailbox 2 transmit is progressing 1: Mailbox 2 transmit finished
15	MST1	Mailbox 1 stop transmitting This bit is set by software to stop mailbox 1 transmitting. This bit is reset by hardware when the mailbox 1 is empty.
14:12	Reserved	Must be kept at reset value.
11	MTE1	Mailbox 1 transmit error This bit is set by hardware when the transmit error occurs. This bit is reset by writting 1 to this bit or MTF1 bit in CAN_TSTAT register. This bit is reset by hardware when next transmit starts.
10	MAL1	Mailbox 1 arbitration lost This bit is set when the arbitration lost occurs. This bit is reset by writting 1 to this bit or MTF1 bit in CAN_TSTAT register. This bit is reset by hardware when next transmit starts.
9	MTFNERR1	Mailbox 1 transmit finished with no error This bit is set when the transmission finishes and no error occurs. This bit is reset by writting 1 to this bit or MTF1 bit in CAN_TSTAT register. This bit is reset by

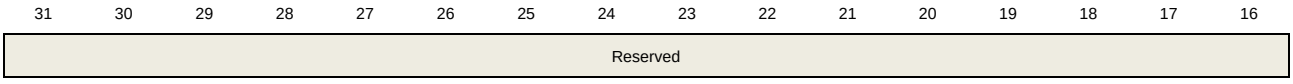
		hardware when the transmission finishes with error. 0: Mailbox 1 transmit finished with error 1: Mailbox 1 transmit finished with no error
8	MTF1	Mailbox 1 transmit finished This bit is set by hardware when the transmission finishes or aborts. This bit is reset by writting 1 to this bit or TEN bit in CAN_TMI1 is 1. 0: Mailbox 1 transmit is progressing 1: Mailbox 1 transmit finished
7	MST0	Mailbox 0 stop transmitting This bit is set by the software to stop mailbox 0 transmitting. This bit is reset by the hardware when the mailbox 0 is empty.
6:4	Reserved	Must be kept at reset value.
3	MTE0	Mailbox 0 transmit error This bit is set by hardware when the transmit error occurs. This bit is reset by writting 1 to this bit or MTF0 bit in CAN_TSTAT register. This bit is reset by hardware when next transmit starts.
2	MAL0	Mailbox 0 arbitration lost This bit is set when the arbitration lost occurs. This bit is reset by writting 1 to this bit or MTF0 bit in CAN_TSTAT register. This bit is reset by hardware when next transmit starts.
1	MTFNERR0	Mailbox 0 transmit finished with no error This bit is set when the transmission finishes and no error occurs. This bit is reset by writting 1 to this bit or MTF0 bit in CAN_TSTAT register. This bit is reset by hardware when the transmission finishes with error. 0: Mailbox 0 transmit finished with error 1: Mailbox 0 transmit finished with no error
0	MTF0	Mailbox 0 transmit finished This bit is set by hardware when the transmission finishes or aborts. This bit is reset by writting 1 to this bit or TEN bit in CAN_TMI0 is 1. 0: Mailbox 0 transmit is progressing 1: Mailbox 0 transmit finished

21.4.4. Receive message FIFO0 register (CAN_RFIFO0)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										RFD0	RFO0	RFF0	Reserved	RFL0[1:0]	
										rs	rc_w1	rc_w1		r	

Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5	RFD0	Rx FIFO0 dequeue This bit is set by software to start dequeuing a frame from Rx FIFO0. This bit is reset by hardware when the dequeuing is done.
4	RFO0	Rx FIFO0 overfull This bit is set by hardware when Rx FIFO0 is overfull and reset by software when writting 1 to this bit. 0: The Rx FIFO0 is not overfull 1: The Rx FIFO0 is overfull
3	RFF0	Rx FIFO0 full This bit is set by hardware when Rx FIFO0 is full and reset by software when writting 1 to this bit. 0: The Rx FIFO0 is not full 1: The Rx FIFO0 is full
2	Reserved	Must be kept at reset value.
1:0	RFL0[1:0]	Rx FIFO0 length These bits are the length of the Rx FIFO0.

21.4.5. Receive message FIFO1 register (CAN_RFIFO1)

Address offset: 0x10
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										RFD1	RFO1	RFF1	Reserved	RFL1[1:0]	
										rs	rc_w1	rc_w1		r	

Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5	RFD1	Rx FIFO1 dequeue This bit is set by software to start dequeuing a frame from Rx FIFO1. This bit is reset by hardware when the dequeuing is done.

4	RFO1	Rx FIFO1 overfull This bit is set by hardware when Rx FIFO1 is overfull and reset by writting 1 to this bit. 0: The Rx FIFO1 is not overfull 1: The Rx FIFO1 is overfull
3	RFF1	Rx FIFO1 full This bit is set by hardware when Rx FIFO1 is full and reset by writting 1 to this bit. 0: The Rx FIFO1 is not full 1: The Rx FIFO1 is full
2	Reserved	Must be kept at reset value.
1:0	RFL1[1:0]	Rx FIFO1 length These bits are the length of the Rx FIFO1.

21.4.6. Interrupt enable register (CAN_INTEN)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRIE	Reserved			ERRNIE	BOIE	PERRIE	WERRIE	Reserved	RFOIE1	RFFIE1	RFNEIE1	RFOIE0	RFFIE0	RFNEIE0	TMEIE
rw				rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:18	Reserved	Must be kept at reset value.
17	SLPWIE	Sleep working interrupt enable 0: Sleep working interrupt disabled 1: Sleep working interrupt enabled
16	WIE	Wakeup interrupt enable 0: Wakeup interrupt disabled 1: Wakeup interrupt enabled
15	ERRIE	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled
14:12	Reserved	Must be kept at reset value.
11	ERRNIE	Error number interrupt enable 0: Error number interrupt disabled

		1: Error number interrupt enabled
10	BOIE	Bus-Off interrupt enable 0: Bus-Off interrupt disabled 1: Bus-Off interrupt enabled
9	PERRIE	Passive error interrupt enable 0: Passive error interrupt disabled 1: Passive error interrupt enabled
8	WERRIE	Warning error interrupt enable 0: Warning error interrupt disabled 1: Warning error interrupt enabled
7	Reserved	Must be kept at reset value.
6	RFOIE1	Rx FIFO1 overfull interrupt enable 0: Rx FIFO1 overfull interrupt disabled 1: Rx FIFO1 overfull interrupt enabled
5	RFFIE1	Rx FIFO1 full interrupt enable 0: Rx FIFO1 full interrupt disabled 1: Rx FIFO1 full interrupt enabled
4	RFNEIE1	Rx FIFO1 not empty interrupt enable 0: Rx FIFO1 not empty interrupt disabled 1: Rx FIFO1 not empty interrupt enabled
3	RFOIE0	Rx FIFO0 overfull interrupt enable 0: Rx FIFO0 overfull interrupt disabled 1: Rx FIFO0 overfull interrupt enabled
2	RFFIE0	Rx FIFO0 full interrupt enable 0: Rx FIFO0 full interrupt disabled 1: Rx FIFO0 full interrupt enabled
1	RFNEIE0	Rx FIFO0 not empty interrupt enable 0: Rx FIFO0 not empty interrupt disabled 1: Rx FIFO0 not empty interrupt enabled
0	TMEIE	Transmit mailbox empty interrupt enable 0: Transmit mailbox empty interrupt disabled 1: Transmit mailbox empty interrupt enabled

21.4.7. Error register (CAN_ERR)

Address offset: 0x18

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RECNT[7:0]								TECNT[7:0]							
r								r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ERRN[2:0]			Reserved	BOERR	PERR	WERR
									rw				r	r	r

Bits	Fields	Descriptions
31:24	RECNT[7:0]	Receive error count defined by the CAN standard
23:16	TECNT[7:0]	Transmit error count defined by the CAN standard
15:7	Reserved	Must be kept at reset value.
6:4	ERRN[2:0]	Error number These bits indicate the error status of bit transformation. They are updated by hardware. When the bit transformation is successful, they are equal to 0. 000: No error 001: Stuff error 010: Form error 011: Acknowledgment error 100: Bit recessive error 101: Bit dominant error 110: CRC error 111: Set by software
3	Reserved	Must be kept at reset value.
2	BOERR	Bus-Off error Whenever the CAN enters Bus-Off state, the bit will be set by hardware.
1	PERR	Passive error Whenever the TECNT or RECNT is greater than 127, the bit will be set by hardware.
0	WERR	Warning error Whenever the TECNT or RECNT is greater than or equal to 96, the bit will be set by hardware.

21.4.8. Bit timing register (CAN_BT)

Address offset: 0x1C
Reset value: 0x0123 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SCMOD	LCMOD	Reserved.	SJW[4:0]					Reserved	BS2[2:0]			BS1[3:0]			
rw	rw		rw						rw			rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	BS2[4:3]	BS1[6:4]	BAUDPSC[9:0]
	rw	rw	rw

Bits	Fields	Descriptions
31	SCMOD	Silent communication mode 0: Silent communication disabled 1: Silent communication enabled
30	LCMOD	Loopback communication mode 0: Loopback communication disabled 1: Loopback communication enabled
29	Reserved	Must be kept at reset value.
28:24	SJW[4:0]	Resynchronization jump width Resynchronization jump width time quantum = SJW[4:0]+1
23	Reserved	Must be kept at reset value.
22:20	BS2[2:0]	Bit segment 2 Bit segment 2 time quantum = BS2[4:0]+1
19:16	BS1[3:0]	Bit segment 1 Bit segment 1 time quantum = BS1[6:0]+1
15	Reserved	Must be kept at reset value.
14:13	BS2[4:3]	Bits 4:3 of BS2 See bits 22:20 of CAN_BT.(Configuration is valid when FDEN=1)
12:10	BS1[6:4]	Bits 6:4 of BS1 See bits 19:16 of CAN_BT. (Configuration is valid when FDEN=1)
9:0	BAUDPSC[9:0]	Baud rate prescaler The CAN baud rate prescaler

21.4.9. FD control register (CAN_FDCTL)

Address offset: 0x20

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ESIMOD	TDCMOD	TDCEN	NISO	PRED	Reserved	FDEN
									rw	rw	rw	rw	rw		rw

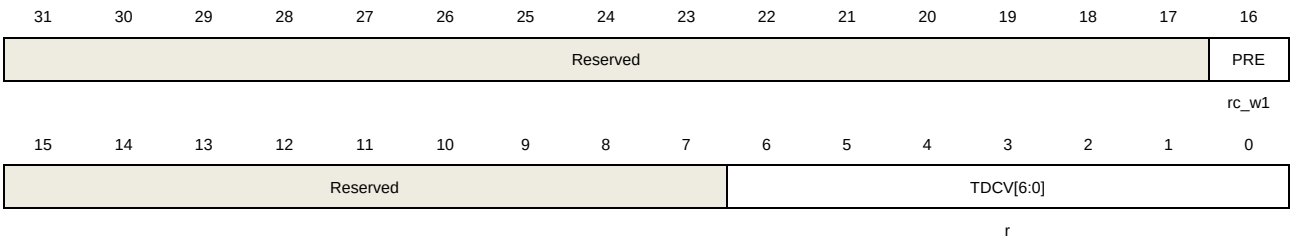
Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	ESIMOD	Error state indicator mode 0: Always displays the node error state. Transmit the dominant bit by error active nodes and transmit the recessive bit by error passive nodes. 1: When the node is not in the error passive state: Displays the message buffer error state by configuring ESI bit in CAN_TMPx registers. When the node is in the error passive state: Displays the node error state.
5	TDCMOD	Transmitter delay compensation mode 0: Measurement and offset 1: Only offset
4	TDCEN	Transmitter delay compensation enable 0: Transmitter delay compensation is disabled 1: Transmitter delay compensation is enabled
3	NISO	ISO/Bosch 0: ISO 1: Bosch
2	PRED	Protocol exception event detection disable 0: Protocol exception event detection enabled (to idle) 1: Protocol exception event detection disabled (regarded as a form error)
1	Reserved	Must be kept at reset value.
0	FDEN	FD operation enable 0: CAN FD function disabled 1: CAN FD function enabled

21.4.10. FD status register (CAN_FDSTAT)

Address offset: 0x24

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



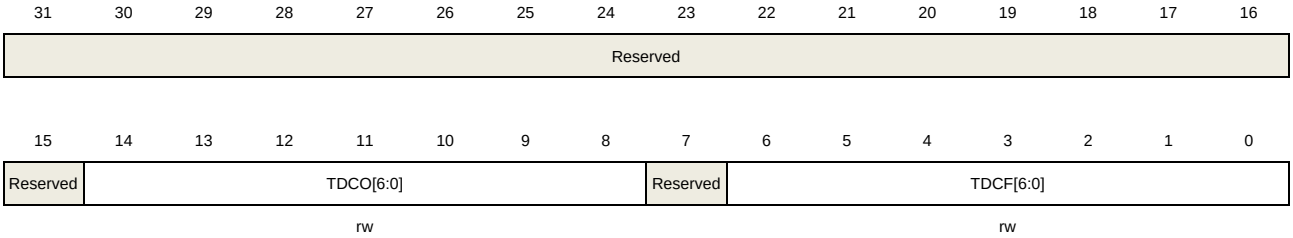
Bits	Fields	Descriptions

31:17	Reserved	Must be kept at reset value.
16	PRE	Protocol exception event This bit is set by hardware when protocol exception event is detected, this bit is cleared by writting 1.
15:7	Reserved	Must be kept at reset value.
6:0	TDCV[6:0]	Transmitter delay compensation value These bits are set by hardware to display transmitter delay compensation value.

21.4.11. FD transmitter delay compensation register (CAN_FDTDC)

Address offset: 0x28
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

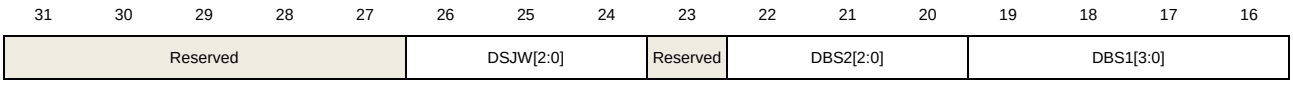


Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14:8	TDCO[6:0]	Transmitter delay compensation offset These bits are set to the transmitter delay compensation offset value which defines the distance between the measured delay from CANTX to CANRX and the second sample point.
7	Reserved	Must be kept at reset value.
6:0	TDCF[6:0]	Transmitter delay compensation filter These bits define the minimum value for the SSP position. Dominant edges on CANRX that would result in an earlier SSP position are ignored for transmitter delay measurement. The feature is enabled when TDCF is configured to a value greater than TDCV.

21.4.12. Date Bit timing register (CAN_DBT)

Address offset: 0x2C
Reset value: 0x0123 0000

This register has to be accessed by word(32-bit).



						rw				rw				rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						DBAUDPSC[9:0]									
						rw									

Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25:24	DSJW[2:0]	Resynchronization jump width Resynchronization jump width time quantum = DSJW[2:0]+1
23	Reserved	Must be kept at reset value.
22:20	DBS2[2:0]	Bit segment 2 Bit segment 2 time quantum = DBS2[2:0]+1
19:16	DBS1[3:0]	Bit segment 1 Bit segment 1 time quantum = DBS1[3:0]+1
15:10	Reserved	Must be kept at reset value.
9:0	DBAUDPSC[9:0]	Baud rate prescaler The CAN baud rate prescaler

21.4.13. Transmit mailbox identifier register (CAN_TMIx) (x = 0...2)

Address offset: 0x180 + 0x10 * x

Reset value: 0xXXXX XXXX (bit0=0)

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SFID[10:0]/EFID[28:18]											EFID[17:13]				
rw											rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EFID[12:0]													FF	FT	TEN
rw													rw	rw	rw

Bits	Fields	Descriptions
31:21	SFID[10:0]/EFID[28:18]	The frame identifier SFID[10:0]: Standard format frame identifier EFID[28:18]: Extended format frame identifier
20:16	EFID[17:13]	The frame identifier EFID[17:13]: Extended format frame identifier
15:3	EFID[12:0]	The frame identifier EFID[12:0]: Extended format frame identifier
2	FF	Frame format

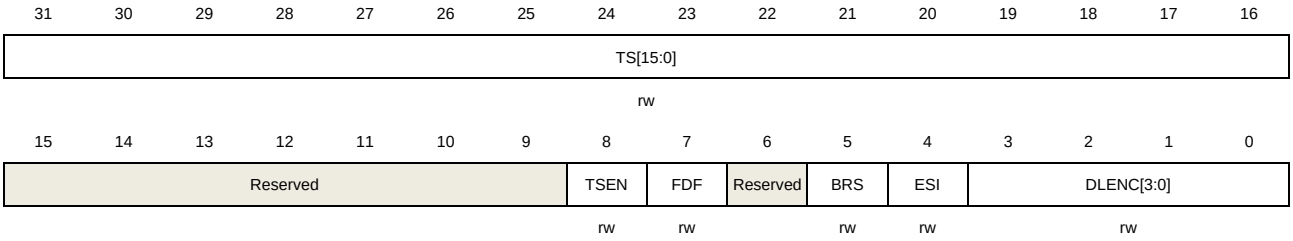
		0: Standard format frame 1: Extended format frame
1	FT	Frame type 0: Data frame 1: Remote frame
0	TEN	Transmit enable This bit is set by software when one frame will be transmitted and reset by hardware when the transmit mailbox is empty. 0: Transmit disabled 1: Transmit enabled

21.4.14. Transmit mailbox property register (CAN_TMPx) (x = 0…2)

Address offset: 0x184 + 0x10 * x

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).



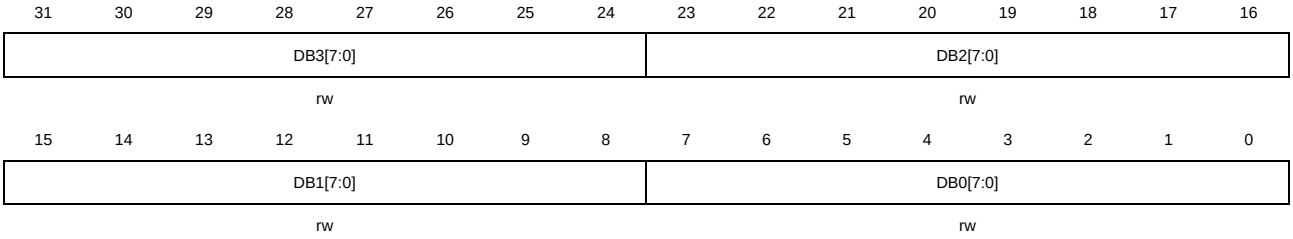
Bits	Fields	Descriptions
31:16	TS[15:0]	Time stamp The time stamp of frame in transmit mailbox.
15:9	Reserved	Must be kept at reset value.
8	TSEN	Time stamp enable 0: Time stamp disabled 1: Time stamp enabled. The TS[15:0] will be transmitted in the DB6 and DB7 in DL. This bit is available when the TTC bit in CAN_CTL is set.
7	FDF	CAN FD frame flag 0: Classical frames 1: FD frames
6	Reserved	Must be kept at reset value.
5	BRS	Bit rate of data switch 0: Bit rate not switch 1: The bit rate shall be switched from the nominal bit rate of the arbitration phase to the preconfigured bit rate of data of the data phase

4	ESI	Error status indicator This bit is valid when ESIMOD bit is 1 in CAN_FDCTL register 0: Transmit the dominant bit in ESI phase 1: Transmit the recessive bit in ESI phase
3:0	DLENC[3:0]	Data length code DLENC[3:0] is the number of bytes in a frame.

21.4.15. Transmit mailbox data0 register (CAN_TMDATA0x) (x = 0…2)

Address offset: 0x188 + 0x10 * x
Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:24	DB3[7:0]	Data byte 3
23:16	DB2[7:0]	Data byte 2
15:8	DB1[7:0]	Data byte 1
7:0	DB0[7:0]	Data byte 0

21.4.16. Transmit mailbox data1 register (CAN_TMDATA1x) (x = 0...2)

Address offset: 0x18C + 0x10 * x
Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:24	DB7[7:0]	Data byte 7

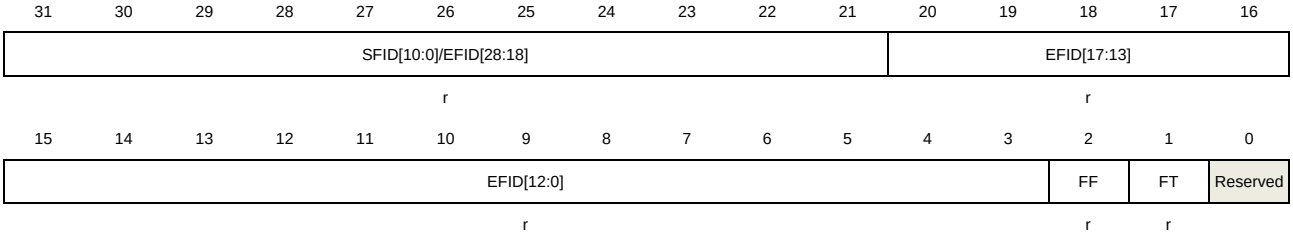
23:16	DB6[7:0]	Data byte 6
15:8	DB5[7:0]	Data byte 5
7:0	DB4[7:0]	Data byte 4

21.4.17. Rx FIFO mailbox identifier register (CAN_RFIFOMIx) (x = 0,1)

Address offset: 0x1B0 + 0x10 * x

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).



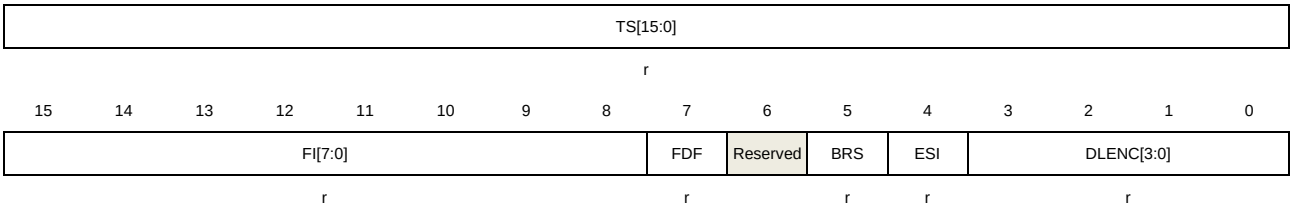
Bits	Fields	Descriptions
31:21	SFID[10:0]/EFID[28:18]	The frame identifier SFID[10:0]: Standard format frame identifier EFID[28:18]: Extended format frame identifier
20:16	EFID[17:13]	The frame identifier EFID[17:13]: Extended format frame identifier
15:3	EFID[12:0]	The frame identifier EFID[12:0]: Extended format frame identifier
2	FF	Frame format 0: Standard format frame 1: Extended format frame
1	FT	Frame type 0: Data frame 1: Remote frame
0	Reserved	Must be kept at reset value.

21.4.18. Rx FIFO mailbox property register (CAN_RFIFOMPx) (x = 0,1)

Address offset: 0x1B4 + 0x10 * x

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).



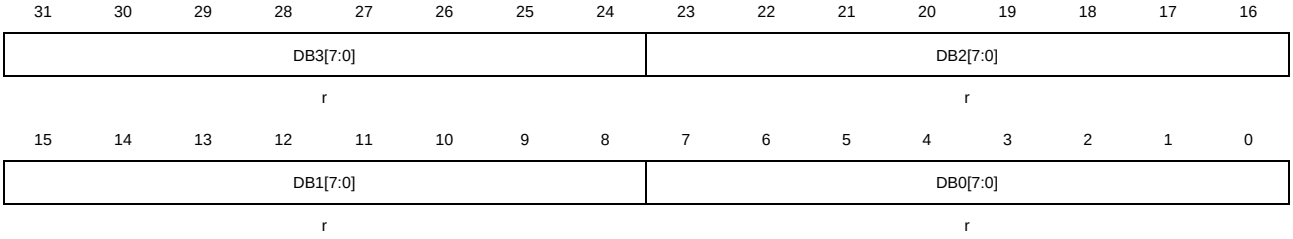
Bits	Fields	Descriptions
31:16	TS[15:0]	Time stamp The time stamp of frame in transmit mailbox.
15:8	FI[7:0]	Filtering index The index of the filter which the frame passes.
7	FDF	CAN FD frame flag 0: Classical frames 1: FD frames
6	Reserved	Must be kept at reset value.
5	BRS	Bit rate of data switch 0: Bit rate not switch 1: The bit rate shall be switched from the nominal bit rate of the arbitration phase to the preconfigured bit rate of data of the data phase
4	ESI	Error status indicator 0: Receive the dominant bit in ESI phase 1: Receive the recessive bit in ESI phase
3:0	DLENC[3:0]	Data length code DLENC[3:0] is the number of bytes in a frame.

21.4.19. Rx FIFO mailbox data0 register (CAN_RFIFOMDATA0x) (x = 0,1)

Address offset: 0x1B8 + 0x10 * x

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).



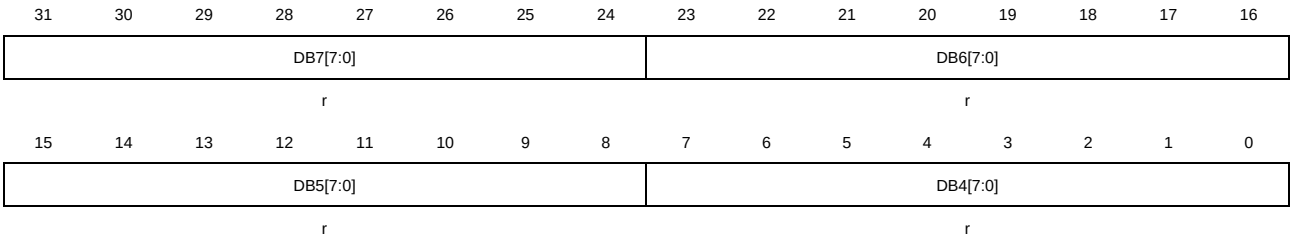
Bits	Fields	Descriptions
31:24	DB3[7:0]	Data byte 3

23:16	DB2[7:0]	Data byte 2
15:8	DB1[7:0]	Data byte 1
7:0	DB0[7:0]	Data byte 0

21.4.20. Rx FIFO mailbox data1 register (CAN_RFIFOMDATA1x) (x=0,1)

Address offset: 0x1BC, 0x1CC
Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).

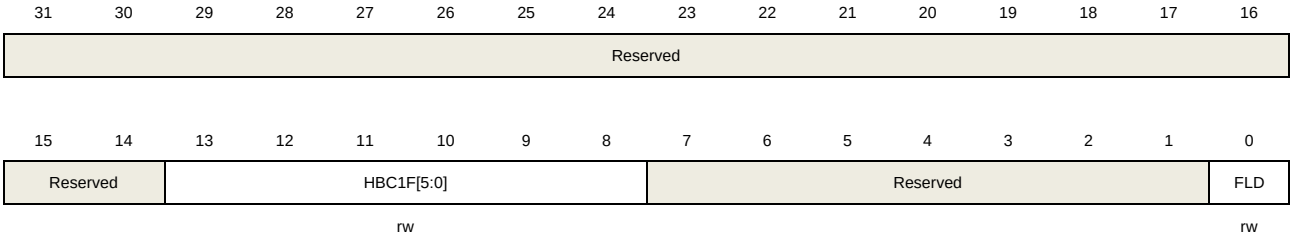


Bits	Fields	Descriptions
31:24	DB7[7:0]	Data byte 7
23:16	DB6[7:0]	Data byte 6
15:8	DB5[7:0]	Data byte 5
7:0	DB4[7:0]	Data byte 4

21.4.21. Filter control register (CAN_FCTL)

Address: 0x40006600
Reset value: 0x2A1C 0E01

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13:8	HBC1F[5:0]	Header bank of CAN1 filter These bits are set and cleared by software to define the first bank for CAN1 filter. Bank0 ~ Bank HBC1F-1 is used for CAN0. Bank HBC1F ~ Bank27 is used for CAN1.

		When set to 0, no bank is used for CAN0. When set to 28, no bank is used for CAN1.
7:1	Reserved	Must be kept at reset value.
0	FLD	Filter lock disable 0: Filter lock enabled 1: Filter lock disabled

21.4.22. Filter mode configuration register (CAN_FMCFG)

Address: 0x40006604

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit). This register can be modified only when FLD bit in CAN_FCTL register is set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				FMOD27	FMOD26	FMOD25	FMOD24	FMOD23	FMOD22	FMOD21	FMOD20	FMOD19	FMOD18	FMOD17	FMOD16
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FMOD15	FMOD14	FMOD13	FMOD12	FMOD11	FMOD10	FMOD9	FMOD8	FMOD7	FMOD6	FMOD5	FMOD4	FMOD3	FMOD2	FMOD1	FMOD0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:0	FMOD**x**	Filter mode 0: Filter **x** with mask mode 1: Filter **x** with list mode

21.4.23. Filter scale configuration register (CAN_FSCFG)

Address: 0x4000660C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit). This register can be modified only when FLD bit in CAN_FCTL register is set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				FS27	FS26	FS25	FS24	FS23	FS22	FS21	FS20	FS19	FS18	FS17	FS16
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FS15	FS14	FS13	FS12	FS11	FS10	FS9	FS8	FS7	FS6	FS5	FS4	FS3	FS2	FS1	FS0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.

27:0	FS**x**	Filter scale 0: Filter **x** with 16-bit scale 1: Filter **x** with 32-bit scale

21.4.24. Filter associated FIFO register (CAN_FAFIFO)

Address: 0x40006614
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit). This register can be modified only when FLD bit in CAN_FCTL register is set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				FAF27	FAF26	FAF25	FAF24	FAF23	FAF22	FAF21	FAF20	FAF19	FAF18	FAF17	FAF16
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FAF15	FAF14	FAF13	FAF12	FAF11	FAF10	FAF9	FAF8	FAF7	FAF6	FAF5	FAF4	FAF3	FAF2	FAF1	FAF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:0	FAF**x**	Filter associated FIFO 0: Filter **x** associated with FIFO0 1: Filter **x** associated with FIFO1

21.4.25. Filter working register (CAN_FW)

Address offset: 0x4000661C
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				FW27	FW26	FW25	FW24	FW23	FW22	FW21	FW20	FW19	FW18	FW17	FW16
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FW15	FW14	FW13	FW12	FW11	FW10	FW9	FW8	FW7	FW6	FW5	FW4	FW3	FW2	FW1	FW0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:0	FW**x**	Filter working 0: Filter **x** working disable

1: Filter **x** working enable

21.4.26. Filter x data y register (CAN_FxDATAy) (x=0..27, y=0,1)

Address: 0x40006640 + 8 * x + 4 * y, (x = 0…27, y=0,1)

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FD31	FD30	FD29	FD28	FD27	FD26	FD25	FD24	FD23	FD22	FD21	FD20	FD19	FD18	FD17	FD16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FD15	FD14	FD13	FD12	FD11	FD10	FD9	FD8	FD7	FD6	FD5	FD4	FD3	FD2	FD1	FD0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	FD**x**	Filter data Mask mode 0: Mask match disabled 1: Mask match enabled List mode 0: List identifier bit is 0 1: List identifier bit is 1

22. Universal serial bus full-speed interface (USBFS)

The USBFS is available on GD32C11x series.

22.1. Overview

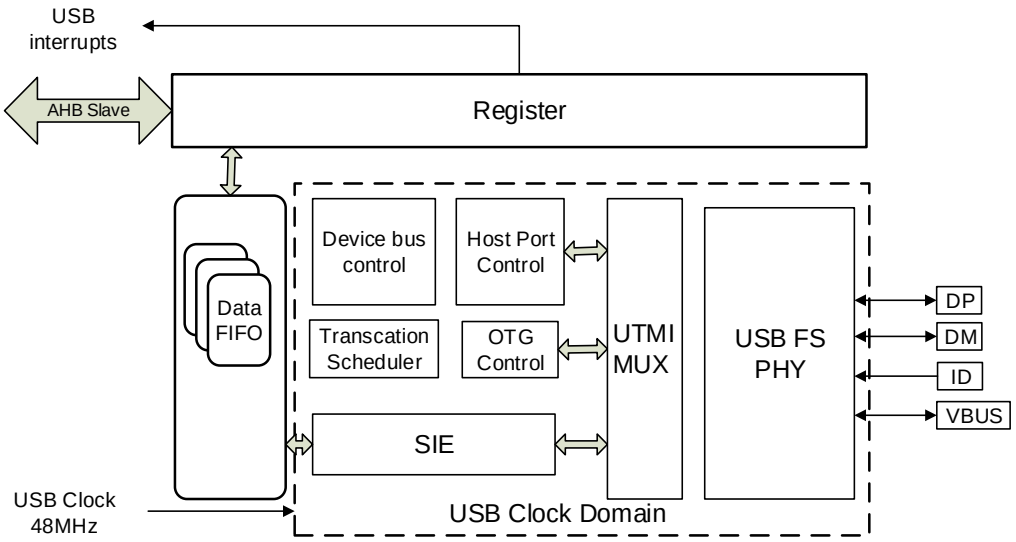
USB Full-Speed (USBFS) controller provides a USB-connection solution for portable devices. USBFS supports host and device modes, as well as OTG mode with HNP (Host Negotiation Protocol) and SRP (Session Request Protocol). USBFS contains a full-speed internal USB PHY and the external PHY chip is not contained. USBFS supports all the four types of transfer (control, bulk, interrupt and isochronous) which are defined in USB 2.0 protocol.

22.2. Characteristics

- Supports USB 2.0 host mode at full-speed (12Mb/s) or low-speed (1.5Mb/s).
- Supports USB 2.0 device mode at full-speed (12Mb/s).
- Supports OTG protocol with HNP (Host Negotiation Protocol) and SRP (Session Request Protocol).
- Supports all the 4 types of transfer: control, bulk, interrupt and isochronous.
- Includes a USB transaction scheduler in host mode to handle USB transaction request efficiently.
- Includes a 1.25KB FIFO RAM.
- Supports 8 channels in host mode.
- Includes 2 Tx FIFOs (periodic and non-periodic) and 1 Rx FIFO (shared by all channels) in host mode.
- Includes 4 Tx FIFOs (one for each IN endpoint) and 1 Rx FIFO (shared by all OUT endpoints) in device mode.
- Supports 4 OUT and 4 IN endpoints in device mode.
- Supports remote wakeup in device mode.
- Includes a full-speed USB PHY with OTG protocol supported.
- Time intervals of SOFs is dynamic adjustable in host mode.
- SOF pulse supports output to PAD.
- Supports detecting ID pin level and VBUS voltage.
- Needs an external component to supply power for connected USB device in host mode or OTG A-Device mode.

22.3. Block diagram

Figure 22-1. USBFS block diagram



22.4. Signal description

Table 22-1. USBFS signal description

I/O port	Type	Description
VBUS	Input/Output	Bus power port
DM	Input/Output	Differential D-
DP	Input/Output	Differential D+
ID	Input	USB identification: Mini connector identification port

22.5. Function overview

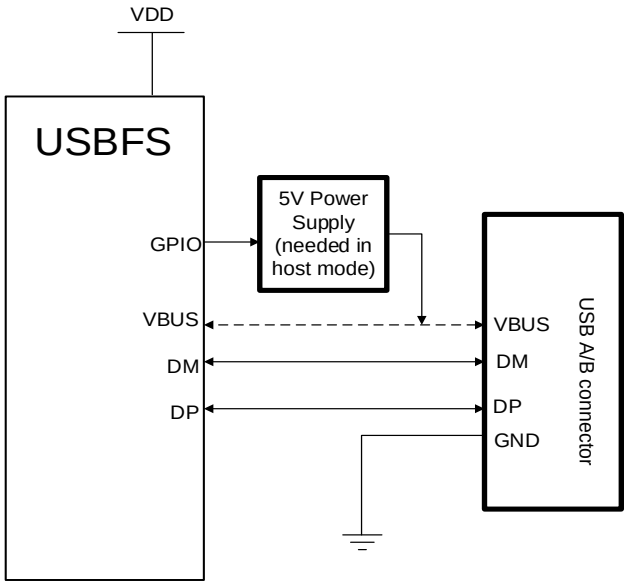
22.5.1. USBFS clocks and working modes

USBFS could be operated as a host, a device or a DRD (Dual-role-Device). It contains an internal full-speed PHY. The maximum speed supported by USBFS is full-speed.

The internal PHY supports full-speed and low-speed in host mode, supports full-speed in device mode, and supports OTG mode with HNP and SRP. The USB clock used by the USBFS should be 48MHz. The 48MHz USB clock is generated from internal clocks in system, and its source and divider factors are configurable in RCU.

The pull-up and pull-down resistors have already been integrated into the internal PHY and they could be controlled by USBFS automatically according to the current mode (host, device or OTG mode) and connection status. A typical connection is shown in ***Figure 22-2. Connection with host or device mode.***

Figure 22-2. Connection with host or device mode

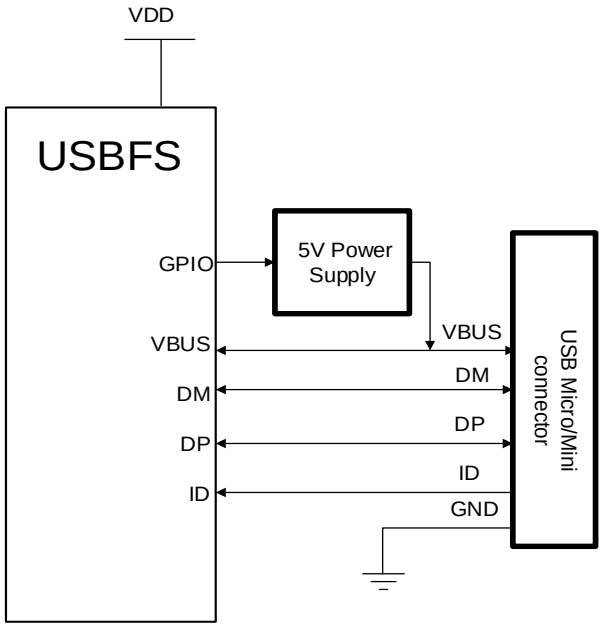


When USBFS works in host mode (FHM bit is set and FDM bit is cleared), the VBUS is 5V power supplied and detecting pin which is used for voltage detection is defined in USB protocol. The internal PHY cannot supply 5V VBUS power and only has some voltage comparers, charge and discharge circuits on VBUS line. Thus, if application needs VBUS power, an external power supply IC is needed. The VBUS connection between USBFS and the USB connector can be omitted in host mode, so USBFS doesn't detect the voltage level on VBUS pin and always assumes that the 5V power is present.

When USBFS works in device mode (FHM bit is cleared and FDM bit is set), the VBUS detection circuit is configured by VBUSIG bit in USBFS_GCCFG register. So if the device does not need to detect the voltage on VBUS pin, it could be configured by setting the VBUSIG bit, then the VBUS pin can be freed for other uses. Otherwise, the VBUS connection cannot be omitted, and USBFS continuously monitors the VBUS voltage. It will immediately switch off the pull-up resistor on DP line once that the VBUS voltage falls below the needed valid value, leading to a disconnection.

The OTG mode connection is described in the ***Figure 22-3. Connection with OTG mode***. When USBFS works in OTG mode, the FHM, FDM bits in USBFS_GUSBCS and VBUSIG bit in USBFS_GCCFG should be cleared. In this mode, the USBFS needs all the four pins: DM, DP, VBUS and ID, and needs to use several voltage comparers to monitor the voltage on these pins. USBFS also contains VBUS charge and discharge circuits to perform SRP request which is described in OTG protocol. The OTG A-Device or B-Device is decided by the level of the ID pin. USBFS controls the pull-up or pull-down resistor during performing the HNP protocol.

Figure 22-3. Connection with OTG mode

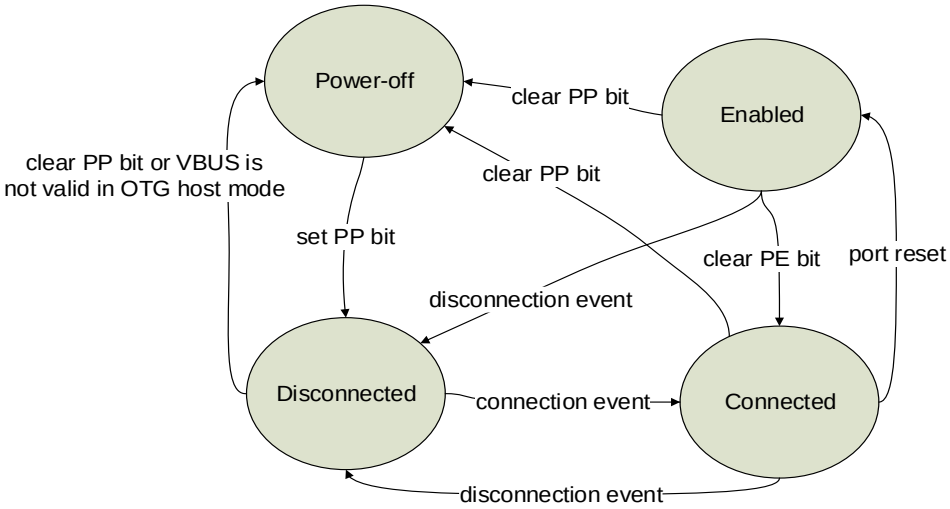


22.5.2. USB host function

USB Host Port State

Host application may control state of the USB port via USBFS_HPCS register. After system initialization, the USB port stays at power-off state. After PP bit is set by software, the internal USB PHY is powered on, and the USB port changes into disconnected state. After a connection is detected, USB port changes into connected state. The USB port changes into enabled state after a port reset is performed on USB bus.

Figure 22-4. State transition diagram of host port



Connection, Reset and Speed identification

As a USB host, USBFS will trigger a connection flag for application after a connection is

detected and will trigger a disconnection flag after a disconnection event.

PRST bit in USBFS_HPCS register is used for USB reset sequence. Application may set this bit to start a USB reset and clear this bit to finish the USB reset. This bit only takes effect when port is at connected or enabled state.

The USBFS performs speed identification during connection, and the speed information will be reported in PS field in USBFS_HPCS register. USBFS identifies the device speed by the voltage level of DM or DP. As described in USB protocol, full-speed device pulls up DP line, while low-speed device pulls up DM line.

Suspend and resume

USBFS supports suspend state and resume operation. When USBFS port is at enabled state, writing 1 to PSP bit in USBFS_HPCS register will cause USBFS to enter into suspend state. In suspend state, USBFS stops sending SOFs on USB bus, and it will lead the connected USB device to enter into suspend state after 3ms. Application can set the PREM bit in USBFS_HPCS register to start a resume sequence, so as to wake up the suspended device, and clear this bit to stop the resume sequence. The WKUPIF bit in USBFS_GINTF will be set and the USBFS wakeup interrupt will be triggered if a host in suspend state detects a remote wakeup signal.

SOF generate

USBFS sends SOF tokens on USB bus in host mode. As described in USB 2.0 protocol, SOF packets are generated (by the host controller or hub transaction translator) each 1ms in full-speed links.

Once that USBFS entered into enabled state, it will send the SOF packet periodically and the period is defined in USB 2.0 protocol. In addition, application may adjust the length of a frame by writing FRI field in USBFS_HFT registers. The FRI bits define the number of USB clock cycles in a frame, so its value should be calculated based on the frequency of USB clock which is used by USBFS. The FRT field bits show that the remaining clock cycles of the current frame and it stops changing during suspend state.

USBFS is able to generate a pulse signal for each SOF packet and output it to a pin. The pulse length is 16 HCLK cycles. If application desires to use this function, it needs to set SOFOEN bit in USBFS_GCCFG register and configure the related pin registers in GPIO.

USB Channels and Transactions

USBFS includes 8 independent channels in host mode. Each channel is able to communicate with an endpoint in USB device. The transfer type, direction, packet length and other information are all configured in channel related registers such as USBFS_HCHxCTL and USBFS_HCHxLEN.

USBFS supports all the four types of transfer: control, bulk, interrupt and isochronous. USB 2.0 protocol divides these transfers into 2 kinds: non-periodic transfer (control and bulk) and periodic transfer (interrupt and isochronous). Based on this, USBFS includes two request queues: periodic request queue and non-periodic request queue, to perform efficient

transaction schedule. A request entry in a request queue described above may represent a USB transaction request or a channel operation request.

Application needs to write packet into data FIFO via AHB bus if it wants to start an OUT transaction on USB bus. USBFS hardware will automatically generate a transaction request entry in request queue after the application writes a whole packet.

The request entries in request queue are processed in order by transaction control module. USBFS always tries to process periodic request queue firstly and secondly process non-periodic request queue.

After a start of frame, USBFS begins to process periodic queue until the queue is empty or bus time required by the current periodic request is not enough, and then process the non-periodic queue. This strategy ensures the bandwidth of periodic transactions in a frame. Each time the USBFS reads and pops a request entry from request queue. If the request is a channel disable request, it immediately disables the channel and prepares to process the next entry.

If the current request is a transaction request and the USB bus time is enough for this transaction, USBFS will employ SIE to generate this transaction on USB bus.

When the required bus time for the current request is not enough in the current frame, and this is a periodic request, USBFS stops processing the periodic queue and starts to process non-periodic request. If this is a non-periodic queue, the USBFS will stop processing any queue and wait until the end of current frame.

22.5.3. USB device function

USB Device Connection

In device mode, USBFS stays at power-off state after initialization. After connecting to a USB host with a 5V power supply through VBUS pin or setting VBUSIG bit in USBFS_GCCFG register, USBFS enters into power-on state. In this state, USBFS begins to switch on the pull-up resistor on DP line and then the host will detect a connection event.

Reset and Speed identification

The USB host always starts a USB reset sequence when it detects a device connection, and USBFS in device mode will trigger a reset interrupt by hardware when it detects the reset event on USB bus.

After the reset sequence, USBFS will trigger an ENUMF interrupt in USBFS_GINTF register and report current enumerated device speed by ES bits in USBFS_DSTAT register, the bit field is always 11(full-speed).

As described in USB 2.0 protocol, USBFS doesn’t support low-speed in device mode.

Suspend and Wakeup

A USB device will enter into suspend state if the USB bus stays at IDLE state and there is no

change on data lines for 3ms. When USB device is in suspend state, most of its clocks are closed to save power. The USB host is able to wake up the suspended device by generating a resume signal on USB bus. When USBFS detects the resume signal, the WKUPIF flag in USBFS_GINTF register will be set and the USBFS wakeup interrupt will be triggered.

In suspend mode, USBFS is also able to remotely wake up the USB bus. Software may set RWKUP bit in USBFS_DCTL register to send a remote wakeup signal, and if remote wakeup is supported in USB host, the host will begin to send the resume signal on USB bus.

Soft Disconnection

USBFS supports soft disconnection. After the device is powered on, USBFS will switch on the pull-up resistor on DP line so that the host can detect the connection. It is able to force a disconnection by setting the SD bit in USBFS_DCTL register. After the SD bit is set, USBFS will directly switch off the pull-up resistor, so that USB host will detect a disconnection on USB bus.

SOF tracking

When USBFS receives a SOF packet on USB bus, it will trigger a SOF interrupt and begin to count the bus time by local USB clock. The frame number of the current frame is reported in FNRSOF field in USBFS_DSTAT register. When the USB bus time reaches EOF1 or EOF2 point (End of Frame, described in USB 2.0 protocol), USBFS will trigger an EOPFIF interrupt in USBFS_GINTF register. These flags and registers can be used to get current bus time and position information.

22.5.4. OTG function overview

USBFS supports OTG function described in OTG protocol 1.3, OTG function includes SRP and HNP protocols.

A-Device and B-Device

A-Device is an OTG capable USB device with a Standard-A or Micro-A plug inserted into its receptacle. The A-Device supplies power to VBUS and it is a host by default at the start of a session. B-Device is an OTG capable USB device with a Standard-B, Micro-B or Mini-B plug inserted into its receptacle, or a captive cable ending being a Standard-A plug. The B-Device is a peripheral by default at the start of a session. USBFS uses the voltage level of ID pin to identify A-Device or B-Device. The ID status is reported in IDPS bit in USBFS_GOTGCS register. For the details of transfer states between A-Device and B-Device, please refer to OTG 1.3 protocol.

HNP

The Host Negotiation Protocol (HNP) allows the host function to be switched between two directly connected On-The-Go devices and eliminates the necessity of switching the cable connections for the change about control of communications between the devices. HNP will be initialized typically by the user or an application on the On-The-Go B-Device. HNP may only be implemented through the Micro-AB receptacle on a device.

Since On-The-Go devices have a Micro-AB receptacle, an On-The-Go device can be a host/device by default, depending on which type of plug (Micro-A plug for host, Micro-B plug for device) is inserted. By utilizing the Host Negotiation Protocol (HNP), an On-The-Go B-Device, which is the default device, may request to be a host. The process for changing the role to be a host is described in next section. This protocol eliminates the necessity of switching the cable connection for the roles change of the connected devices.

When USBFS is in OTG A-Device host mode and it wants to give up its host role, it may firstly set PSP bit in USBFS_HPCS register to make the USB bus enter into suspend status. Then, the B-Device will enter into suspend state 3ms later. If the B-Device wants to change to be a host, HNPREQ bit in USBFS_GOTGCS register should be set and the USBFS will begin to perform HNP protocol on bus, and at last, the result of HNP is reported in HNPS bit in USBFS_GOTGCS register. In additional, it is always available to get the current role (host or device) from COPM bit in USBFS_GINTF register by application.

SRP

The Session Request Protocol (SRP) allows a B-Device to request the A-Device to turn on VBUS and start a session. This protocol allows the A-Device, which may be battery powered, to save power by turning VBUS off when there is no bus activity, while still providing a means for the B-Device to initiate bus activity. As is described in OTG protocol, an OTG device must compare VBUS voltage with several threshold values, and the compared result should be reported in ASV and BSV bits in USBFS_GOTGCS register.

Set SRPREQ bit in USBFS_GOTGCS register to start a SRP request when USBFS is in OTG B-Device mode. USBFS will generate a success flag SRPS in USBFS_GOTGCS register if the SRP requests successfully.

When USBFS is in OTG A-Device mode and it has detected a SRP request from a B-Device, it sets a SESIF flag in USBFS_GINTF register. The 5V power supply for VBUS pin should be prepared to switch on after getting this flag.

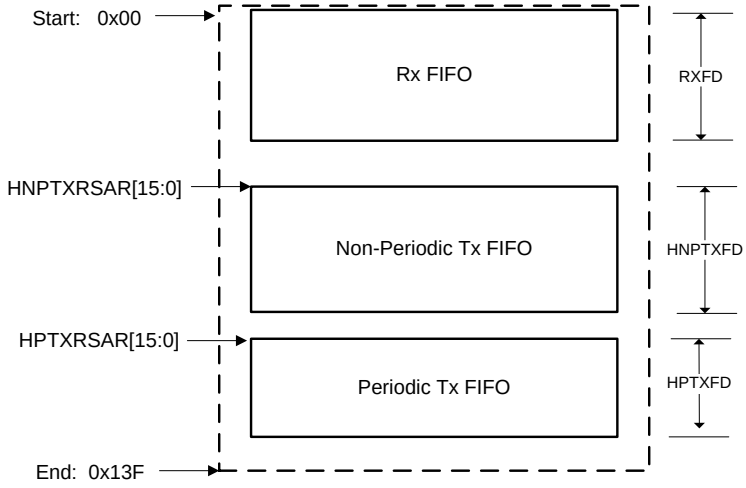
22.5.5. Data FIFO

The USBFS contains a 1.25K bytes data FIFO for packet data storage. The data FIFO is implemented by using an internal SRAM in USBFS.

Host Mode

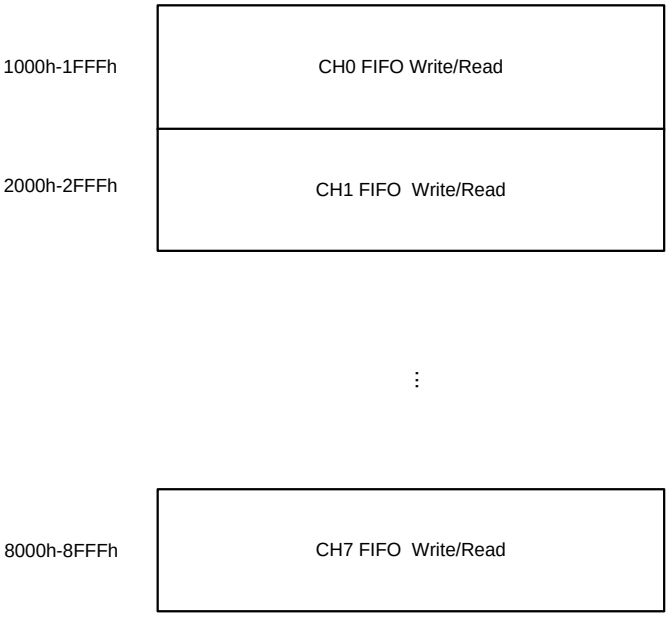
In host mode, the data FIFO space is divided into 3 parts: Rx FIFO for received packet, non-periodic Tx FIFO for non-period transmission packet and periodic Tx FIFO for periodic transmission packet. All IN channels shares the Rx FIFO for packets reception. All the periodic OUT channels share the periodic Tx FIFO for packets transmission. All the non-periodic OUT channels share the non-periodic FIFO for packets transmission. The size and start offset of these data FIFOs should be configured using these registers: USBFS_GRFLEN, USBFS_HNPTFLEN and USBFS_HPTFLEN. ***Figure 22-5. HOST mode FIFO space in SRAM*** describes the structure of these FIFOs in SRAM. The values in the figure are in terms of 32-bit words.

Figure 22-5. HOST mode FIFO space in SRAM



USBFS provides a special register area for the internal data FIFO reading and writing. ***Figure 22-6. Host mode FIFO access register mapping*** describes the register memory area that the data FIFO can access. The addresses in the figure are addressed in bytes. Each channel has its own FIFO access register space, although all non-periodic channels share the same FIFO and all the periodic channels also share the same FIFO. It is important for USBFS to get which channel the current pushed packet belongs to, and the Rx FIFO which the packet belongs to is also able to be accessed by using USBFS_GRSTATR/USBFS_GRSTATP register.

Figure 22-6. Host mode FIFO access register mapping

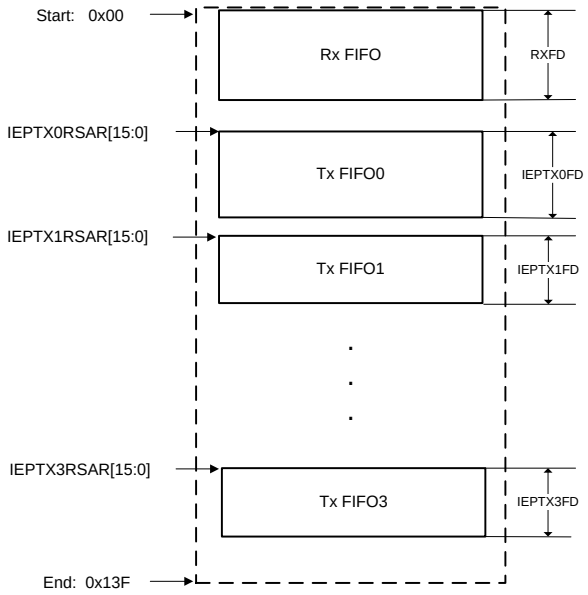


Device mode

In device mode, the data FIFO is divided into several parts: 1 Rx FIFO, and 4 Tx FIFOs (one for each IN endpoint). All the OUT endpoints share the Rx FIFO for receiving packets. The size and start offset of these data FIFOs should be configured by using USBFS_GRFLEN

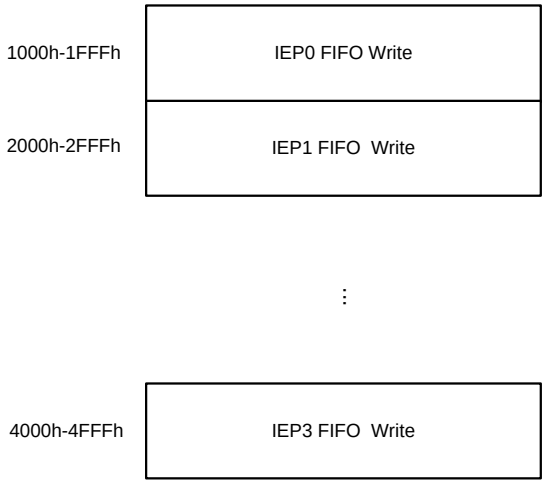
and USBFS_DIEPxTFLEN (x=0…3) registers. ***Figure 22-7. Device mode FIFO space in SRAM*** describes the structure of these FIFOs in SRAM. The values in the figure are in terms of 32-bit words.

Figure 22-7. Device mode FIFO space in SRAM



USBFS provides a special register area for the internal data FIFO reading and writing. ***Figure 22-8. Device mode FIFO access register mapping*** describes the register memory area where the data FIFO can access. The addresses in the figure are addressed in bytes. Each endpoint has its own FIFO access register space. Rx FIFO is also able to be accessed by using USBFS_GRSTATR/USBFS_GRSTATP register.

Figure 22-8. Device mode FIFO access register mapping



22.5.6. Operation guide

This section describes the advised operation guide for USBFS.

Host mode

Global register initialization sequence

1. Program USBFS_GAHBCS register according to application's demand, such as the Tx FIFO's empty threshold, etc. GINTEN bit should be kept cleared at this time.
2. Program USBFS_GUSBCS register according to application's demand, such as the operation mode (host, device or OTG) and some parameters of OTG and USB protocols.
3. Program USBFS_GCCFG register according to application's demand.
4. Program USBFS_GRFLEN, USBFS_HNPTFLEN and USBFS_HPTFLEN register to configure the data FIFOs according to application's demand.
5. Program USBFS_GINTEN register to enable mode fault and host port interrupt and set GINTEN bit in USBFS_GAHBCS register to enable global interrupt.
6. Program USBFS_HPCS register to set PP bit.
7. Wait for a device's connection, and once a device is connected, the connection interrupt PCD in USBFS_HPCS register will be triggered. Then set PRST bit to perform a port reset. Wait for at least 10ms and then clear PRST bit.
8. Wait PEDC interrupt in USBFS_HPCS register and then read PE bit to ensure that the port is successfully enabled. Read PS [1:0] bits to get the connected device's speed and then program USBFS_HFT register to change the SOF interval if needed.

Channel initialization and enable sequence

1. Program USBFS_HCHxCTL registers with desired transfer type, direction, packet size, etc. Ensure that CEN and CDIS bits are kept cleared during configuration.
2. Program USBFS_HCHxINTEN register. Set the desired interrupt enable bits.
3. Program USBFS_HCHxLEN register. PCNT is the number of packets in a transfer and TLEN is the total bytes number of all the transmitted or received packets in a transfer.

   For OUT channel: If PCNT=1, the single packet's size is equal to TLEN. If PCNT>1, the former PCNT-1 packets are considered as max-packet-lengthened packets whose size are defined by MPL field in USBFS_HCHxCTL register, and the last packet's size is calculated based on PCNT, TLEN and MPL. If software wants to send out a zero-lengthened packet, it should program TLEN=0, PCNT=1.

   For IN channel: Because the application doesn't know the actual received data size before the IN transaction finishes, TLEN could be set to a maximum possible value supported by Rx FIFO.
4. Set CEN bit in USBFS_HCHxCTL register to enable the channel.

Channel disable sequence

Software can disable the channel by setting both CEN and CDIS bits at the same time.

USBFS will generate a channel disable request entry in request queue after the register setting operation. When the request entry reaches the top of request queue, it will be processed by USBFS immediately:

For OUT channels, the specified channel will be disabled immediately. Then, a CH flag will be generated and the CEN and CDIS bits will be cleared by USBFS.

For IN channels, USBFS pushes a channel disable status entry into Rx FIFO. Then software should handle the Rx FIFO not empty event: read and pop this status entry, and then, a CH flag will be generated and the CEN and CDIS bits will be cleared.

IN transfers operation sequence

1. Initialize USBFS global registers.
2. Initialize the channel.
3. Enable the channel.
4. After the IN channel is enabled by software, USBFS generates an Rx request entry in the corresponding request queue.
5. When the Rx request entry reaches the top of the request queue, USBFS begins to process this request entry. If bus time for the IN transaction indicated by the request entry is enough, USBFS starts the IN transaction on USB bus.
6. If the IN transaction is finished successfully (ACK handshake received), USBFS pushes the received data packet into the Rx FIFO and triggers ACK flag. Otherwise, the status flag (NAK) reports the transaction result.
7. If the IN transaction described in step 5 is successful and PCNT is larger than 1 in step2, returns to step 3 and continues to receive the remaining packets. If the IN transaction described in step 5 is not successful, returns to step 3 to re-receive the packet again.
8. After all the transactions in a transfer have been successfully received on USB bus, USBFS pushes a TF status entry into the Rx FIFO on top of the last packet data. Thus after reading and popping all the received data packet, the TF status entry is read. USBFS generates TF flag to indicate that the transfer successfully have been finished.
9. Disable the channel. Now the channel is in IDLE state and is ready for other transfers.

OUT transfers operation sequence

1. Initialize USBFS global registers.
2. Initialize and enable the channel.
3. Write a packet into the channel's Tx FIFO (Periodic Tx FIFO or non-periodic Tx FIFO). After the whole packet data is written into the FIFO, USBFS generates a Tx request entry in the corresponding request queue and decreases the TLEN field in USBFS_HCHxLEN register by the written packet's size.
4. When the request entry reaches the top of the request queue, USBFS begins to process

this request entry. If bus time for the transaction indicated by the request entry is enough, USBFS starts the OUT transaction on USB bus.

5. When the OUT transaction indicated by the request entry has been finished on USB bus, PCNT in USBFS_HCHxLEN register is decreased by 1. If the transaction is finished successfully (ACK handshake received), the ACK flag is triggered. Otherwise, the status flag (NAK) reports the transaction result.
6. If the OUT transaction described in step 5 is successful and PCNT is larger than 1 in step 2, returns to step 3 and continues to send the remaining packets. If the OUT transaction described in step 5 is not successful, return to step 3 to re-send the packet again.
7. After all the transactions in a transfer are successfully sent on USB bus, USBFS generates TF flag to indicate that the transfer successfully finishes.
8. Disable the channel. Now the channel is in IDLE state and is ready for other transfers.

Device mode

Global register initialization sequence

1. Program USBFS_GAHBCS register according to application's demand, such as the TxFIFO's empty threshold, etc. GINTEN bit should be kept cleared at this time.
2. Program USBFS_GUSBCS register according to application's demand, such as: the operation mode (host, device or OTG) and some parameters of OTG and USB protocols.
3. Program USBFS_GCCFG register according to application's demand.
4. Program USBFS_GRFLEN, USBFS_DIEP0TFLEN, USBFS_DIEPxTFLEN register to configure the data FIFOs according to application's demand.
5. Program USBFS_GINTEN register to enable Mode Fault, Suspend, SOF, Enumeration Done and USB Reset interrupt, and then, set GINTEN bit in USBFS_GAHBCS register to enable global interrupt.
6. Program USBFS_DCFG register according to application's demand, such as the device address, etc.
7. After the device is connected to a host, the host will perform port reset on USB bus and this will trigger the RST interrupt in USBFS_GINTF register.
8. Wait for ENUMF interrupt in USBFS_GINTF register.

Endpoint initialization and enable sequence

1. Program USBFS_DIEPxCTL or USBFS_DOEPxCTL register with desired transfer type, packet size, etc.
2. Program USBFS_DIEPINTEN or USBFS_DOEPINTEN register. Set the desired interrupt enable bits.
3. Program USBFS_DIEPxLEN or USBFS_DOEPxLEN register. PCNT is the number of

packets in a transfer and TLEN is the total bytes number of all the transmitted or received packets in a transfer.

For IN endpoint：If PCNT=1, the single packet's size is equal to TLEN. If PCNT>1, the former PCNT-1 packets are considered as max-packet-length packets whose size are defined by MPL field in USBFS_DIEPxCTL register, and the last packet's size is calculated based on PCNT, TLEN and MPL. If a zero-length packet is required to be sent, it should program TLEN=0, PCNT=1.

For OUT endpoint：Because the application doesn't know the actual received data size before the OUT transaction finishes, TLEN can be set to a maximum possible value supported by Rx FIFO.

4. Set EPEN bit in USBFS_DIEPxCTL or USBFS_DOEPxCTL register to enable the endpoint.

Endpoint disable sequence

The endpoint could be disabled anytime when the EPEN bit in USBFS_DIEPxCTL or USBFS_DOEPxCTL registers is cleared.

IN transfers operation sequence

1. Initialize USBFS global registers.
2. Initialize and enable the IN endpoint.
3. Write packets into the endpoint's Tx FIFO. At any time, a data packet is written into the FIFO, USBFS decreases the TLEN field in USBFS_DIEPxLEN register by the written packet's size.
4. When an IN token received, USBFS transmits the data packet, and after the transaction finishes on USB bus, PCNT in USBFS_DIEPxLEN register is decreased by 1. If the transaction finishes successfully (ACK handshake received), the ACK flag is triggered. Otherwise, the status flags reports the transaction result.
5. After all the data packets in a transfer have been successfully sent on USB bus, USBFS generates TF flag to indicate that the transfer successfully is finished and the IN endpoint is disabled.

OUT transfers operation sequence

1. Initialize USBFS global registers.
2. Initialize the endpoint and enable the endpoint.
3. When an OUT token is received, USBFS receives the data packet or response with an NAK handshake based on the status of Rx FIFO and register configuration. If the transaction is finished successfully (USBFS receives and saves the data packet into Rx FIFO successfully and sends ACK handshake on USB bus), PCNT in USBFS_DOEPxLEN register is decreased by 1 and the ACK flag is triggered, otherwise, the status flags report the transaction result.

4. After all the data packets in a transfer are successfully received on USB bus, USBFS pushes a TF status entry into the Rx FIFO on top of the last packet data. Thus, after reading and popping all the received data packet, the TF status entry is read. USBFS generates TF flag to indicate that the transfer is successfully finished and the IN endpoint is disabled.

22.6. Interrupts

USBFS has two interrupts: global interrupt and wakeup interrupt.

The source flags of the global interrupt are readable in USBFS_GINTF register and are listed in ***Table 22-2. USBFS global interrupt***.

Table 22-2. USBFS global interrupt

Interrupt flag	Description	Operation mode
SESIF	Session interrupt	Host or device mode
DISCIF	Disconnected interrupt flag	Host Mode
IDPSC	ID pin status change	Host or device mode
PTXFEIF	Periodic Tx FIFO empty interrupt flag	Host Mode
HCIF	Host channels interrupt flag	Host Mode
HPIF	Host port interrupt flag	Host Mode
ISOONCIF/PXNCIF	Isochronous OUT transfer not complete interrupt flag / Periodic transfer not complete interrupt flag	Host or device mode
ISOINCIF	Isochronous IN transfer not complete interrupt flag	Device mode
OEPIF	OUT endpoint interrupt flag	Device mode
IEPIF	IN endpoint interrupt flag	Device mode
EOPFIF	End of periodic frame interrupt flag	Device mode
ISOOPDIF	Isochronous OUT packet dropped interrupt flag	Device mode
ENUMF	Enumeration finished	Device mode
RST	USB reset	Device mode
SP	USB suspend	Device mode
ESP	Early suspend	Device mode
GONAK	Global OUT NAK effective	Device mode
GNPINAK	Global IN Non-Periodic NAK effective	Device mode
NPTXFEIF	Non-Periodic Tx FIFO empty interrupt flag	Host Mode
RXFNEIF	Rx FIFO non-empty interrupt flag	Host or device mode
SOF	Start of frame	Host or device mode
OTGIF	OTG interrupt flag	Host or device mode
MFIF	Mode fault interrupt flag	Host or device mode

Wakeup interrupt can be triggered when USBFS is in suspend state, even if when the

USBFS's clocks are stopped. The source of the wakeup interrupt is WKUPIF bit in USBFS_GINTF register.

22.7. Register definition

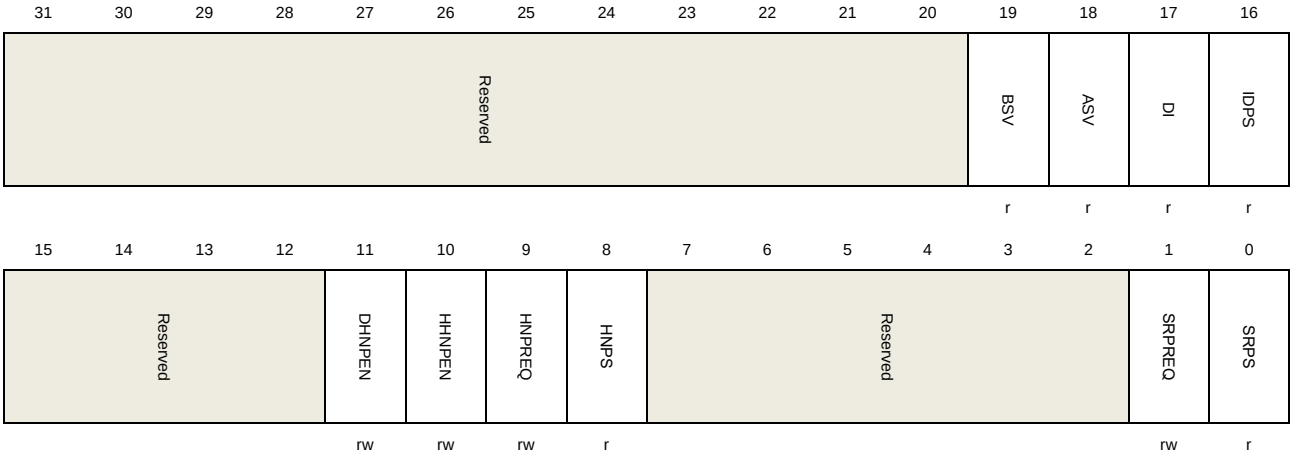
USBFS base address: 0x5000 0000

22.7.1. Global control and status registers

Global OTG control and status register (USBFS_GOTGCS)

Address offset: 0x0000
Reset value: 0x0000 0800

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19	BSV	B-Session valid (described in OTG protocol) 0: VBUS voltage level of a OTG B-Device is below $V_{BSESSVLD}$ 1: VBUS voltage level of a OTG B-Device is not below $V_{BSESSVLD}$ **Note:** Only accessible in OTG B-Device mode.
18	ASV	A-Session valid A-Device mode transceiver status 0: VBUS voltage level of a OTG A-Device is below $V_{ASESSVLD}$ 1: VBUS voltage level of a OTG A-Device is below $V_{ASESSVLD}$ The A-Device is the host by default at the start of a session. **Note:** Only accessible in OTG A-Device mode.
17	DI	Debounce interval Debounce interval of a detected connection 0: Indicates the long debounce interval, when a plug-on and connection occurs on USB bus. 1: Indicates the short debounce interval, when a soft connection is used in HNP protocol.

		Note: Only accessible in host mode.
16	IDPS	ID pin status Voltage level of connector ID pin 0: USBFS is in A-Device mode 1: USBFS is in B-Device mode **Note:** Accessible in both device and host modes.
15:12	Reserved	Must be kept at reset value.
11	DHNPEN	Device HNP enable Enable the HNP function of a B-Device. If this bit is cleared, USBFS will not start HNP protocol when application sets HNPREQ bit in USBFS_GOTGCS register. 0: HNP function is not enabled 1: HNP function is enabled **Note:** Only accessible in device mode.
10	HHNPEN	Host HNP enable Enable the HNP function of an A-Device. If this bit is cleared, USBFS doesn't respond to the HNP request from B-Device. 0: HNP function is not enabled 1: HNP function is enabled **Note:** Only accessible in host mode.
9	HNPREQ	HNP request This bit is set by software to start a HNP on the USB. When HNPEND bit in USBFS_GOTGINTF register is set, this bit can be cleared by writing zero to it, or clearing the HNPEND bit in USBFS_GOTGINTF register. 0: Don't send HNP request 1: Send HNP request **Note:** Only accessible in device mode.
8	HNPS	HNP success flag This bit is set by the core when HNP success, and this bit is cleared when HNPREQ bit is set. 0: HNP failure 1: HNP success **Note:** Only accessible in device mode.
7:2	Reserved	Must be kept at reset value.
1	SRPREQ	SRP request This bit is set by software to start a SRP on the USB. When SRPEND bit in USBFS_GOTGINTF register is set, this bit can be cleared by writing zero to it, or clearing the SRPEND bit in USBFS_GOTGINTF register. 0: No session request 1: Session request **Note:** Only accessible in device mode.

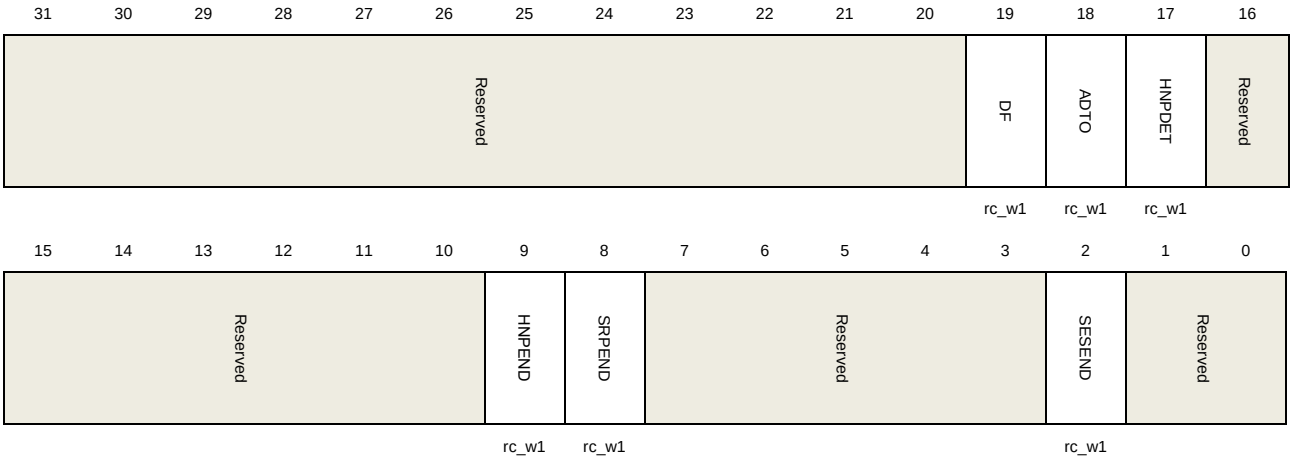
0	SRPS	SRP success flag This bit is set by the core when SRP success, and this bit is cleared when SRPREQ bit is set. 0: SRP failure 1: SRP success **Note:** Only accessible in device mode.

Global OTG interrupt flag register (USBFS_GOTGINTF)

Address offset: 0x0004

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19	DF	Debounce finish Set by USBFS when the debounce is done during device connection. **Note:** Only accessible in host mode.
18	ADTO	A-Device timeout Set by USBFS when it is timed out for the A-Device waiting for a B-Device' connection. **Note:** Accessible in both device and host modes.
17	HNPDET	Host negotiation request detected Set by USBFS when A-Device detects a HNP request. **Note:** Accessible in both device and host modes.
16:10	Reserved	Must be kept at reset value.
9	HNPEND	HNP end Set by the core when a HNP ends. Read the HNPS in USBFS_GOTGCS register to get the result of HNP.

		Note: Accessible in both device and host modes.
8	SRPEND	SRPEND Set by the core when a SRP ends. Read the SRPS in USBFS_GOTGCS register to get the result of SRP. **Note:** Accessible in both device and host modes.
7:3	Reserved	Must be kept at reset value.
2	SESEND	Session end Set by the core when VBUS voltage is below $V_{b_ses_vld}$.
1:0	Reserved	Must be kept at reset value.

Global AHB control and status register (USBFS_GAHBCS)

Address offset: 0x0008

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)



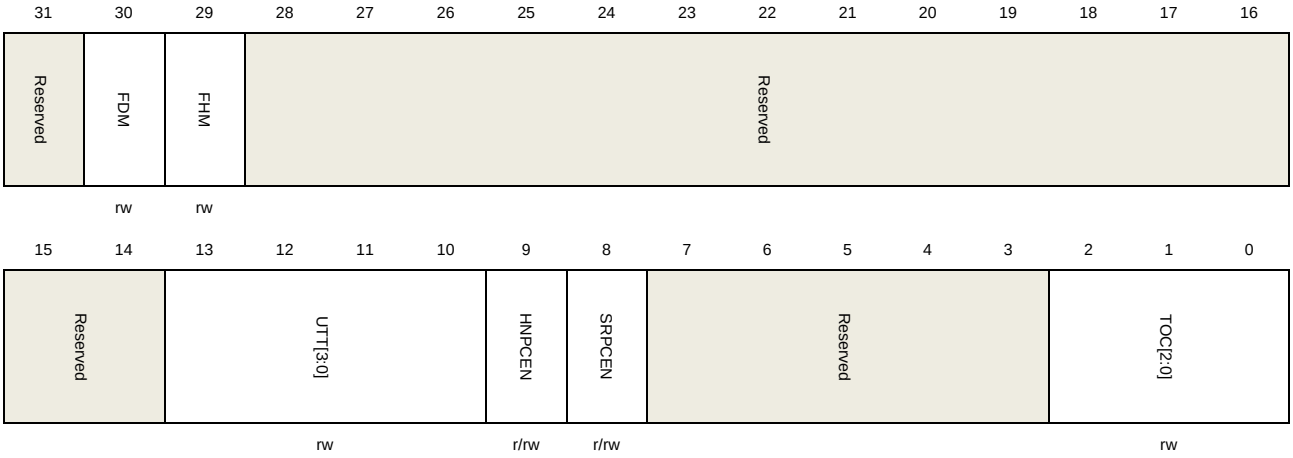
Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	PTXFTH	Periodic Tx FIFO threshold 0: PTXFEIF will be triggered when the periodic Tx FIFO is half empty. 1: PTXFEIF will be triggered when the periodic Tx FIFO is completely empty. **Note:** Only accessible in host mode.
7	TXFTH	Tx FIFO threshold **Device mode**: 0: TXFEIF will be triggered when the IN endpoint Tx FIFO is half empty. 1: TXFEIF will be triggered when the IN endpoint Tx FIFO is completely empty. **Host mode**: 0: NPTXFEIF will be triggered when the non-periodic Tx FIFO is half empty.

		1: NPTXFEIF will be triggered when the non-periodic Tx FIFO is completely empty.
6：1	Reserved	Must be kept at reset value.
0	GINTEN	Global interrupt enable 0: Global interrupt is not enabled. 1: Global interrupt is enabled. **Note:** Accessible in both device and host modes.

Global USB control and status register (USBFS_GUSBCS)

Address offset: 0x000C
Reset value: 0x0000 0A80

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	FDM	Force device mode Setting this bit will force the core to device mode irrespective of the USBFS ID input pin. 0: Normal mode 1: Device mode The application must wait at least 25ms for the change taking effect after setting the force bit. **Note:** Accessible in both device and host modes.
29	FHM	Force host mode Setting this bit will force the core to host mode irrespective of the USBFS ID input pin. 0: Normal mode 1: Host mode The application must wait at least 25ms for the change taking effect after setting the

		force bit. **Note:** Accessible in both device and host modes.
28:14	Reserved	Must be kept at reset value.
13:10	UTT[3:0]	USB turnaround time Turnaround time in PHY clocks. **Note:** Only accessible in device mode.
9	HNPCEN	HNP capability enable Controls whether the HNP capability is enabled. 0: HNP capability is disabled 1: HNP capability is enabled **Note:** Accessible in both device and host modes.
8	SRPCEN	SRP capability enable Controls whether the SRP capability is enabled. 0: SRP capability is disabled 1: SRP capability is enabled **Note:** Accessible in both device and host modes.
7:3	Reserved	Must be kept at reset value.
2:0	TOC[2:0]	Timeout calibration USBFS always uses timeout value required in USB 2.0 when waiting for a packet. The TOC bits are used to add the value in terms of PHY clock. (The frequency of PHY clock is 48MHz.).

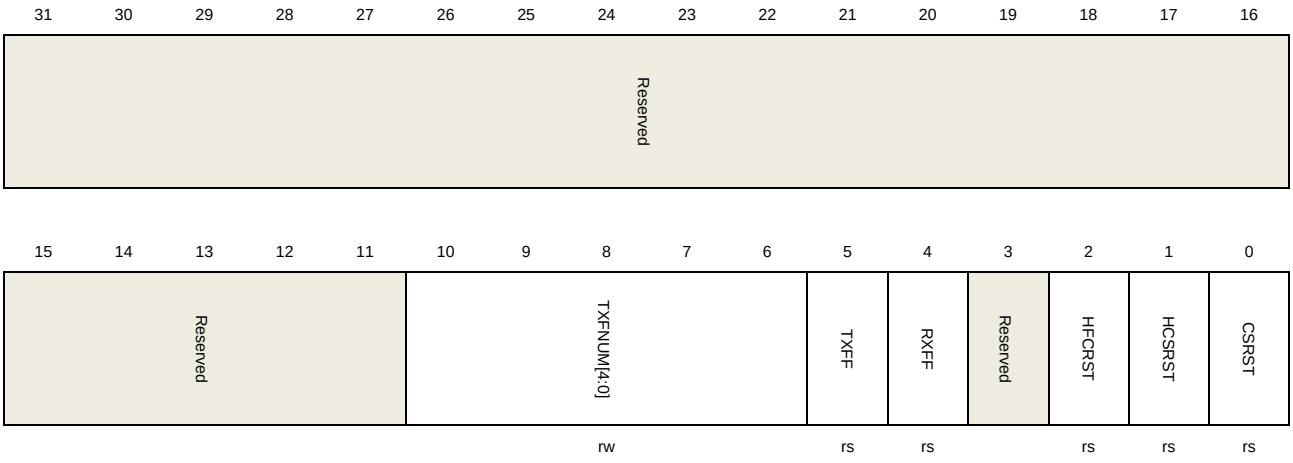
Global reset control register (USBFS_GRSTCTL)

Address offset: 0x0010

Reset value: 0x8000 0000

The application uses this register to reset various hardware features inside the core.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10:6	TXFNUM[4:0]	Tx FIFO number Indicates which Tx FIFO will be flushed when TXFF bit in the same register is set. **Host Mode:** 00000: Only non-periodic Tx FIFO is flushed. 00001: Only periodic Tx FIFO is flushed. 1xxxx: Both periodic and non-periodic Tx FIFOs are flushed. Others: No data FIFO is flushed. **Device Mode:** 00000: Only Tx FIFO0 is flushed 00001: Only Tx FIFO1 is flushed … 00011: Only Tx FIFO3 is flushed 1xxxx: All Tx FIFOs are flushed Others: No data FIFO is flushed
5	TXFF	Tx FIFO flush Set the bit to flush data Tx FIFOs and TXFNUM[4:0] bits determine the FIFO number to be flushed. Hardware automatically clears this bit after the flush process completes. After setting this bit, application should wait until this bit is cleared before any other operation on USBFS. **Note:** Accessible in both device and host modes.
4	RXFF	Rx FIFO flush Set the bit to flush data Rx FIFO. Hardware automatically clears this bit after the flush process completes. After setting this bit, application should wait until this bit is cleared before any other operation on USBFS. **Note:** Accessible in both device and host modes.
3	Reserved	Must be kept at reset value.
2	HFCRST	Host frame counter reset Set by the application to reset the frame number counter in USBFS. After this bit is set, the frame number of the following SOF returns to 0. Hardware automatically clears this bit after the reset process completes. After setting this bit, application should wait until this bit is cleared before any other operation on USBFS. **Note:** Only accessible in host mode.
1	HCSRST	HCLK soft reset Set by the application to reset AHB clock domain circuit. Hardware automatically clears this bit after the reset process completes. After setting this bit, application should wait until this bit is cleared before any other operation on USBFS. **Note:** Accessible in both device and host modes.
0	CSRST	Core soft reset

Resets the AHB and USB clock domains circuits, as well as most of the registers.

Global interrupt flag register (USBFS_GINTF)

Address offset: 0x0014

Reset value: 0x0400 0021

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WKUPIF	SESIF	DISCIF	IDPSC	Reserved.	PTXFEIF	HCIF	HPIF	Reserved		PXNCIF/ ISOONCIF	ISOINCIF	OEPIF	IEPIF	Reserved	
rc_w1	rc_w1	rc_w1	rc_w1		r	r	r			rc_w1	rc_w1	r	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EOPFIF	ISOOPDIF	ENUMF	RST	SP	ESP	Reserved		GONAK	GNPINAK	NPTXFEIF	RXFNEIF	SOF	OTGIF	MFIF	COPM
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1			r	r	r	r	rc_w1	r	rc_w1	r

Bits	Fields	Descriptions
31	WKUPIF	Wakeup interrupt flag This interrupt is triggered when a resume signal (in device mode) or a remote wakeup signal (in host mode) is detected on the USB. **Note:** Accessible in both device and host modes.
30	SESIF	Session interrupt flag This interrupt is triggered when a SRP is detected (in A-Device mode) or VBUS becomes valid for a B-Device (in B-Device mode). **Note:** Accessible in both device and host modes.
29	DISCIF	Disconnected interrupt flag This interrupt is triggered after a device disconnection. **Note:** Only accessible in host mode.
28	IDPSC	ID pin status change Set by the core when ID status changes. **Note:** Accessible in both device and host modes.
27	Reserved	Must be kept at reset value.
26	PTXFEIF	Periodic Tx FIFO empty interrupt flag This interrupt is triggered when the periodic Tx FIFO is either half or completely empty. The threshold is determined by the periodic Tx FIFO empty level bit (PTXFTH) in the USBFS_GAHBCS register. **Note:** Only accessible in host mode.

25	HCIF	Host channels interrupt flag Set by USBFS when one of the channels in host mode has raised an interrupt. First read USBFS_ HACHINT register to get the channel number, and then read the corresponding USBFS_HCHxINTF register to get the flags of the channel that cause the interrupt. This bit will be automatically cleared after the respective channel's flags which cause channel interrupt are cleared. **Note:** Only accessible in host mode.
24	HPIF	Host port interrupt flag Set by the core when USBFS has detected the port status changes in host mode. Software should read USBFS_HPCS register to get the source of this interrupt. This bit will be automatically cleared after the flags that causing a port interrupt are cleared. **Note:** Only accessible in host mode.
23:22	Reserved	Must be kept at reset value.
21	PXNCIF	Periodic transfer not complete interrupt flag USBFS sets this bit when there are periodic transactions not completed at the end of current frame (Host mode).
	ISOONCIF	Isochronous OUT transfer not complete interrupt flag At the end of a periodic frame (defined by EOPFT bit in USBFS_DCFG), USBFS will set this bit if there are still isochronous OUT endpoints for the transactions not completed (Device Mode).
20	ISOINCIF	Isochronous IN transfer not complete interrupt flag At the end of a periodic frame (defined by EOPFT bits in USBFS_DCFG), USBFS will set this bit if there are still isochronous IN endpoints for the transactions not completed (Device Mode). **Note:** Only accessible in device mode.
19	OEPIF	OUT endpoint interrupt flag Set by USBFS when one of the OUT endpoints in device mode has raised an interrupt. Software should first read USBFS_DAEPINT register to get the endpoint number, and then read the corresponding USBFS_DOEPxINTF register to get the flags of the endpoint that cause the interrupt. This bit will be automatically cleared after the respective endpoint's flags which cause this interrupt are cleared. **Note:** Only accessible in device mode.
18	IEPIF	IN endpoint interrupt flag Set by USBFS when one of the IN endpoints in device mode has raised an interrupt. Software should first read USBFS_DAEPINT register to get the endpoint number, and then read the corresponding USBFS_DIEPxINTF register to get the flags of the endpoint that cause the interrupt. This bit will be automatically cleared after the respective endpoint's flags which cause this interrupt are cleared. **Note:** Only accessible in device mode.

17:16	Reserved	Must be kept at reset value.
15	EOPFIF	End of periodic frame interrupt flag When USB bus time in a frame reaches the value defined by EOPFT [1:0] bits in USBFS_DCFG register, USBFS sets this flag. **Note:** Only accessible in device mode.
14	ISOOPDIF	Isochronous OUT packet dropped interrupt flag USBFS sets this bit if it receives an isochronous OUT packet but cannot save it into Rx FIFO as it doesn't have enough space. **Note:** Only accessible in device mode.
13	ENUMF	Enumeration finished USBFS sets this bit after the speed enumeration finishes. Read USBFS_DSTAT register to get the current device speed. **Note:** Only accessible in device mode.
12	RST	USB reset USBFS sets this bit when it detects a USB reset signal on bus. **Note:** Only accessible in device mode.
11	SP	USB suspend USBFS sets this bit when it detects that the USB bus is idle for 3ms and enters suspend state. **Note:** Only accessible in device mode.
10	ESP	Early suspend USBFS sets this bit when it detects that the USB bus is idle for 3ms. **Note:** Only accessible in device mode.
9:8	Reserved	Must be kept at reset value.
7	GONAK	Global OUT NAK effective Write 1 to SGONAK bit in the USBFS_DCTL register and USBFS will set this flag after the SGONAK takes effect. **Note:** Only accessible in device mode.
6	GNPINAK	Global Non-Periodic IN NAK effective Write 1 to SGINAK bit in the USBFS_DCTL register and USBFS will set this flag after the SGINAK takes effect. **Note:** Only accessible in device mode.
5	NPTXFEIF	Non-periodic Tx FIFO empty interrupt flag This interrupt is triggered when the non-periodic Tx FIFO is either half or completely empty. The threshold is determined by the non-periodic Tx FIFO empty level bit (TXFTH) in the USBFS_GAHBCS register. **Note:** Only accessible in host mode.
4	RXFNEIF	Rx FIFO non-empty interrupt flag USBFS sets this bit when there is at least one packet or status entry in the Rx FIFO.

		Note: Accessible in both host and device modes.
3	SOF	Start of frame Host Mode: USBFS sets this bit when it prepares to transmit a SOF or Keep-Alive on USB bus. This bit can be cleared by writing 1. Device Mode: USBFS sets this bit after it receives a SOF token. The application can read the Device Status register to get the current frame number. This bit can be cleared by writing 1. **Note:** Accessible in both host and device modes.
2	OTGIF	OTG interrupt flag USBFS sets this bit when an interrupt flag is set in USBFS_GOTGINTF register. Software should read USBFS_GOTGINTF register to get the source of this interrupt. This bit is cleared after the flags in USBFS_GOTGINTF causing this interrupt are cleared. **Note:** Accessible in both host and device modes.
1	MFIF	Mode fault interrupt flag USBFS sets this bit when software operates host-only register in device mode, or operates device-only register in host mode. These fault operations won't take effect. **Note:** Accessible in both host and device modes.
0	COPM	Current operation mode 0: Device mode 1: Host mode **Note:** Accessible in both host and device modes.

Global interrupt enable register (USBFS_GINTEN)

Address offset: 0x0018
Reset value: 0x0000 0000

This register works with the global interrupt flag register (USBFS_GINTF) to interrupt the application. When an interrupt enable bit is disabled, the interrupt associated with that bit is not generated. However, the global Interrupt flag register bit corresponding to that interrupt is still set.

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WKUPIE	SESIE	DISCIE	IDPSCIE	Reserved.	PTXFEIE	HCIE	HPIE	Reserved		PXNCIE/ ISOONCIE	ISOINCIE	OEPIE	IEPIE	Reserved	
rw	rw	rw	rw		rw	rw	r			rw	rw	rw	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

EOPFIE	ISOOPDIE	ENUMFIE	RSTIE	SPIE	ESPIE	Reserved	GONAKIE	GNPINAKIE	NPTXFEIE	RXFNEIE	SOFIE	OTGIE	MFIE	Reserved
rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	

Bits	Fields	Descriptions
31	WKUPIE	Wakeup interrupt enable 0: Disable wakeup interrupt 1: Enable wakeup interrupt **Note:** Accessible in both host and device modes.
30	SESIE	Session interrupt enable 0: Disable session interrupt 1: Enable session interrupt **Note:** Accessible in both host and device modes.
29	DISCIE	Disconnected interrupt enable 0: Disable disconnected interrupt 1: Enable disconnected interrupt **Note:** Only accessible in device mode.
28	IDPSCIE	ID pin status change interrupt enable 0: Disable connector ID pin status interrupt 1: Enable connector ID pin status interrupt **Note:** Accessible in both host and device modes.
27	Reserved	Must be kept at reset value.
26	PTXFEIE	Periodic Tx FIFO empty interrupt enable 0: Disable periodic Tx FIFO empty interrupt 1: Enable periodic Tx FIFO empty interrupt **Note:** Only accessible in host mode.
25	HCIE	Host channels interrupt enable 0: Disable host channels interrupt 1: Enable host channels interrupt **Note:** Only accessible in host mode.
24	HPIE	Host port interrupt enable 0: Disable host port interrupt 1: Enable host port interrupt **Note:** Only accessible in host mode.
23:22	Reserved	Must be kept at reset value.
21	PXNCIE	Periodic transfer not complete Interrupt enable 0: Disable periodic transfer not complete interrupt 1: Enable periodic transfer not complete interrupt

		Note: Only accessible in host mode.
	ISOONCIE	Isochronous OUT transfer not complete interrupt enable 0: Disable isochronous OUT transfer not complete interrupt 1: Enable isochronous OUT transfer not complete interrupt **Note:** Only accessible in device mode.
20	ISOINCIE	Isochronous IN transfer not complete interrupt enable 0: Disable isochronous IN transfer not complete interrupt 1: Enable isochronous IN transfer not complete interrupt **Note:** Only accessible in device mode.
19	OEPIE	OUT endpoints interrupt enable 0: Disable OUT endpoints interrupt 1: Enable OUT endpoints interrupt **Note:** Only accessible in device mode.
18	IEPIE	IN endpoints interrupt enable 0: Disable IN endpoints interrupt 1: Enable IN endpoints interrupt **Note:** Only accessible in device mode.
17:16	Reserved	Must be kept at reset value.
15	EOPFIE	End of periodic frame interrupt enable 0: Disable end of periodic frame interrupt 1: Enable end of periodic frame interrupt **Note:** Only accessible in device mode.
14	ISOOPDIE	Isochronous OUT packet dropped interrupt enable 0: Disable isochronous OUT packet dropped interrupt 1: Enable isochronous OUT packet dropped interrupt **Note:** Only accessible in device mode.
13	ENUMFIE	Enumeration finish enable 0: Disable enumeration finish interrupt 1: Enable enumeration finish interrupt **Note:** Only accessible in device mode.
12	RSTIE	USB reset interrupt enable 0: Disable USB reset interrupt 1: Enable USB reset interrupt **Note:** Only accessible in device mode.
11	SPIE	USB suspend interrupt enable 0: Disable USB suspend interrupt 1: Enable USB suspend interrupt **Note:** Only accessible in device mode.
10	ESPIE	Early suspend interrupt enable

		0: Disable early suspend interrupt 1: Enable early suspend interrupt **Note:** Only accessible in device mode.
9:8	Reserved	Must be kept at reset value.
7	GONAKIE	Global OUT NAK effective interrupt enable 0: Disable global OUT NAK interrupt 1: Enable global OUT NAK interrupt **Note:** Only accessible in device mode.
6	GNPINAKIE	Global non-periodic IN NAK effective interrupt enable 0: Disable global non-periodic IN NAK effective interrupt 1: Enable global non-periodic IN NAK effective interrupt **Note:** Only accessible in device mode.
5	NPTXFEIE	Non-periodic Tx FIFO empty interrupt enable 0: Disable non-periodic Tx FIFO empty interrupt 1: Enable non-periodic Tx FIFO empty interrupt **Note:** Only accessible in Host mode.
4	RXFNEIE	Receive FIFO non-empty interrupt enable 0: Disable receive FIFO non-empty interrupt 1: Enable receive FIFO non-empty interrupt **Note:** Accessible in both device and host modes.
3	SOFIE	Start of frame interrupt enable 0: Disable start of frame interrupt 1: Enable start of frame interrupt **Note:** Accessible in both device and host modes.
2	OTGIE	OTG interrupt enable 0: Disable OTG interrupt 1: Enable OTG interrupt **Note:** Accessible in both device and host modes.
1	MFIE	Mode fault interrupt enable 0: Disable mode fault interrupt 1: Enable mode fault interrupt **Note:** Accessible in both device and host modes.
0	Reserved	Must be kept at reset value.

Global receive status read/pop registers (USBFS_GRSTATR/USBFS_GRSTATP)

Address offset for Read: 0x001C

Address offset for Pop: 0x0020

Reset value: 0x0000 0000

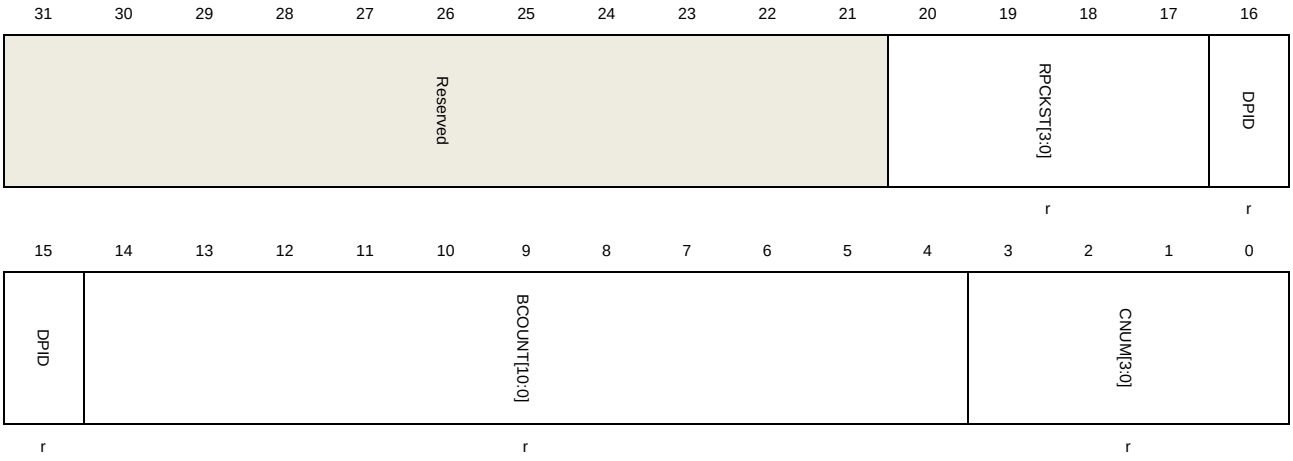
A read to the receive status read register returns the entry of the top of the Rx FIFO. A read

to the receive status pop register pops the top entry out of the Rx FIFO.

The entries in Rx FIFO have different meanings in host and device modes. Software should only read this register after when Rx FIFO non-empty interrupt flag bit of the global interrupt flag register (RXFNEIF bit in USBFS_GINTF) is triggered.

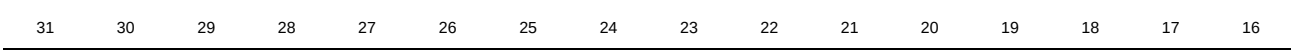
This register has to be accessed by word (32-bit)

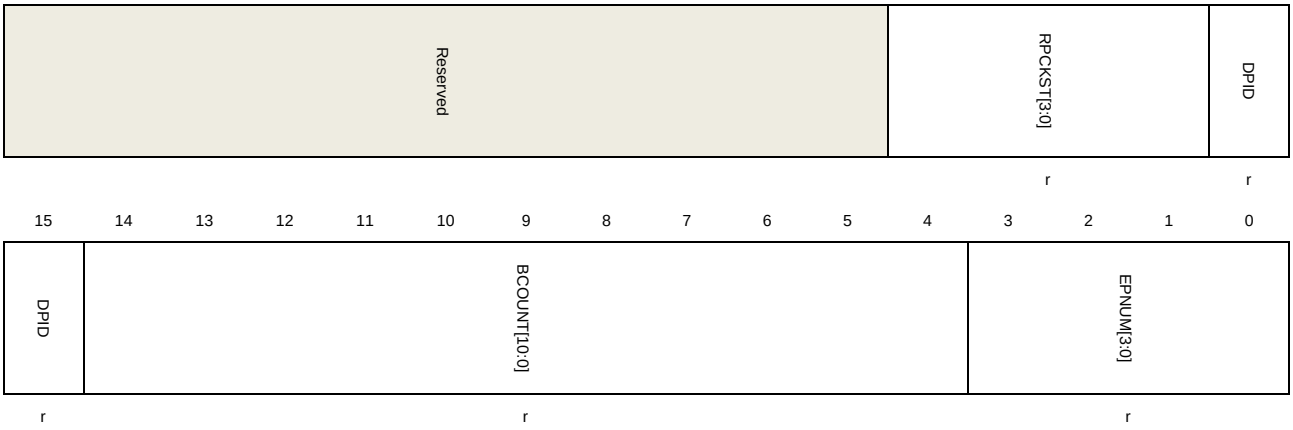
Host mode:



Bits	Fields	Descriptions
31:21	Reserved	Must be kept at reset value.
20:17	RPCKST[3:0]	Received packet status 0010: IN data packet received 0011: IN transfer completed (generates an interrupt if popped) 0101: Data toggle error (generates an interrupt if popped) 0111: Channel halted (generates an interrupt if popped) Others: Reserved
16:15	DPID[1:0]	DATA PID The DATA PID of the received packet 00: DATA0 10: DATA1 Others: Reserved
14:4	BCOUNT[10:0]	Byte count The byte count of the received IN data packet.
3:0	CNUM[3:0]	Channel number The channel number to which the current received packet belongs.

Device mode:





Bits	Fields	Descriptions
31:21	Reserved	Must be kept at reset value.
20:17	RPCKST[3:0]	Received packet status 0001: Global OUT NAK (generates an interrupt) 0010: OUT data packet received 0011: OUT transfer completed (generates an interrupt) 0100: SETUP transaction completed (generates an interrupt) 0110: SETUP data packet received Others: Reserved
16:15	DPID[1:0]	DATA PID The DATA PID of the received OUT data packet 00: DATA0 10: DATA1 Others: Reserved
14:4	BCOUNT[10:0]	Byte count The byte count of the received data packet.
3:0	EPNUM[3:0]	Endpoint number The endpoint number to which the current received packet belongs.

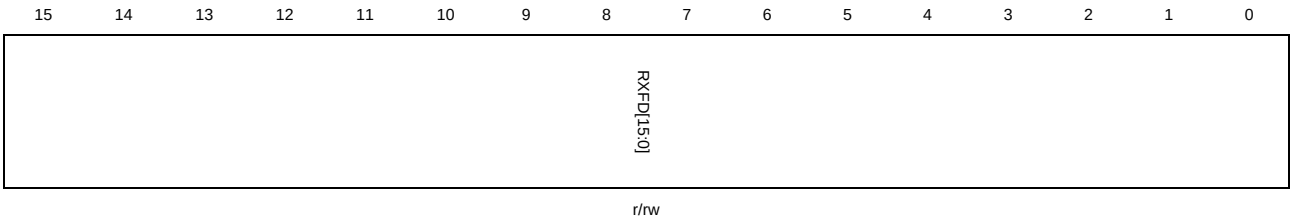
Global receive FIFO length register (USBFS_GRFLEN)

Address offset: 0x024

Reset value: 0x0000 0200

This register has to be accessed by word (32-bit)





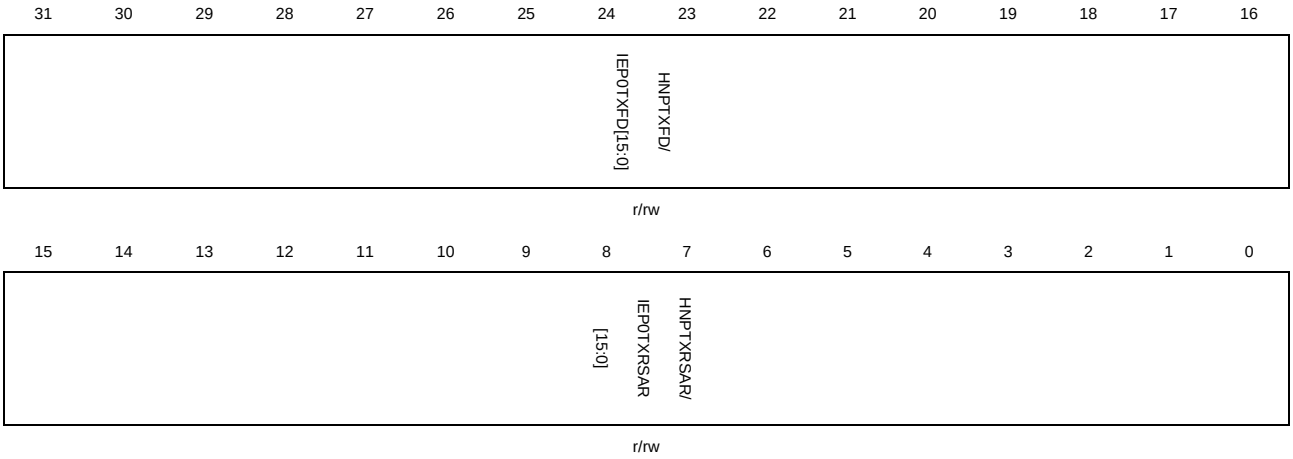
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	RXFD[15:0]	Rx FIFO depth In terms of 32-bit words 1≤RXFD≤1024

Host non-periodic Tx FIFO length register/Device IN endpoint 0 Tx FIFO length (USBFS_HNPTFLEN/USBFS_DIEP0TFLEN)

Address offset: 0x028

Reset value: 0x0200 0200

This register has to be accessed by word (32-bit)



Host Mode:

Bits	Fields	Descriptions
31:16	HNPTXFD[15:0]	Host non-periodic Tx FIFO depth In terms of 32-bit words 1≤HNPTXFD≤1024
15:0	HNPTXRSAR[15:0]	Host non-periodic Tx FIFO RAM start address The start address for non-periodic Tx FIFO RAM is in terms of 32-bit words.

Device Mode:

Bits	Fields	Descriptions
31:16	IEP0TXFD[15:0]	IN endpoint 0 Tx FIFO depth

		In terms of 32-bit words 16≤IEP0TXFD≤140
15:0	IEP0TXRSAR[15:0]	IN endpoint 0 Tx FIFO RAM start address The start address for endpoint 0 Tx FIFO RAM is in terms of 32-bit words.

Host non-periodic Tx FIFO/queue status register (USBFS_HNPTFQSTAT)

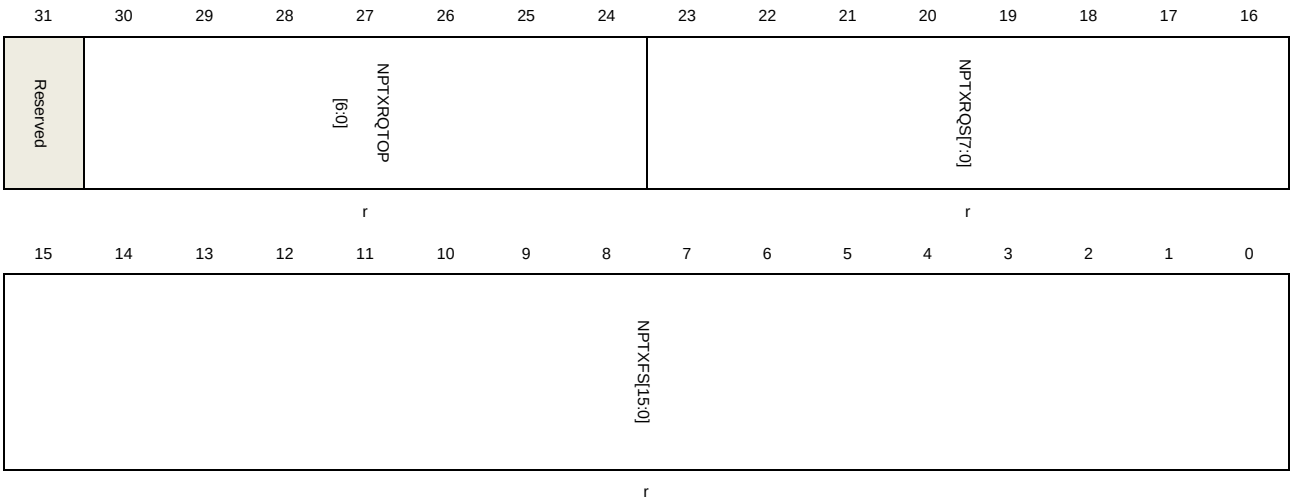
Address offset: 0x002C

Reset value: 0x0008 0200

This register reports the current status of the non-periodic Tx FIFO and request queue. The request queue includes IN, OUT or other request entries in host mode.

Note: In Device mode, this register is not valid.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:24	NPTXRQTOP[6:0]	Top entry of the non-periodic Tx request queue Entry in the non-periodic transmit request queue. Bits 30:27: Channel number Bits 26:25: – 00: IN/OUT token – 01: Zero-length OUT packet – 11: Channel halt request Bit 24: Terminate flag, indicating last entry for selected channel.
23:16	NPTXRQS[7:0]	Non-periodic Tx request queue space The remaining space of the non-periodic transmit request queue. 0: Request queue is full. 1: 1 entry

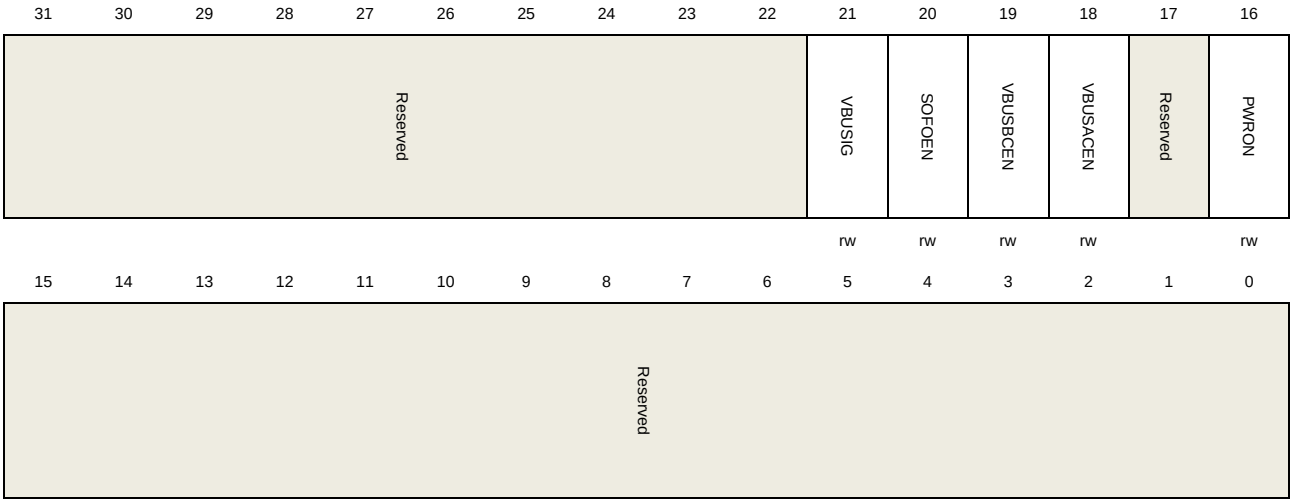
		2: 2 entries … n: n entries (0≤n≤8) Others: Reserved
15:0	NPTXFS[15:0]	Non-periodic Tx FIFO space The remaining space of the non-periodic Tx FIFO. In terms of 32-bit words 0: Non-periodic Tx FIFO is full. 1: 1 word 2: 2 words … n: n words (0≤n≤NPTXFD) Others: Reserved

Global core configuration register (USBFS_GCCFG)

Address offset: 0x0038

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.
21	VBUSIG	VBUS ignored When this bit is set, USBFS doesn't monitor the voltage on VBUS pin and always considers the V_{BUS} voltage as valid both in host mode and in device mode, then frees the V_{BUS} pin for other usage. 0: V_{BUS} is not ignored. 1: V_{BUS} is ignored and always consider VBUS voltage as valid.

20	SOFOEN	SOF output enable 0: SOF pulse output disabled 1: SOF pulse output enabled
19	VBUSBCEN	The V_{BUS} B-Device comparer enable 0: V_{BUS} B-Device comparer disabled 1: V_{BUS} B-Device comparer enabled
18	VBUSACEN	The VBUS A-Device comparer enable 0: V_{BUS} A-Device comparer disabled 1: V_{BUS} A-Device comparer enabled
17	Reserved	Must be kept at reset value.
16	PWRON	Power on This bit is the power switch for the internal embedded full-speed PHY. 0: Embedded full-speed PHY power off 1: Embedded full-speed PHY power on
15:0	Reserved	Must be kept at reset value.

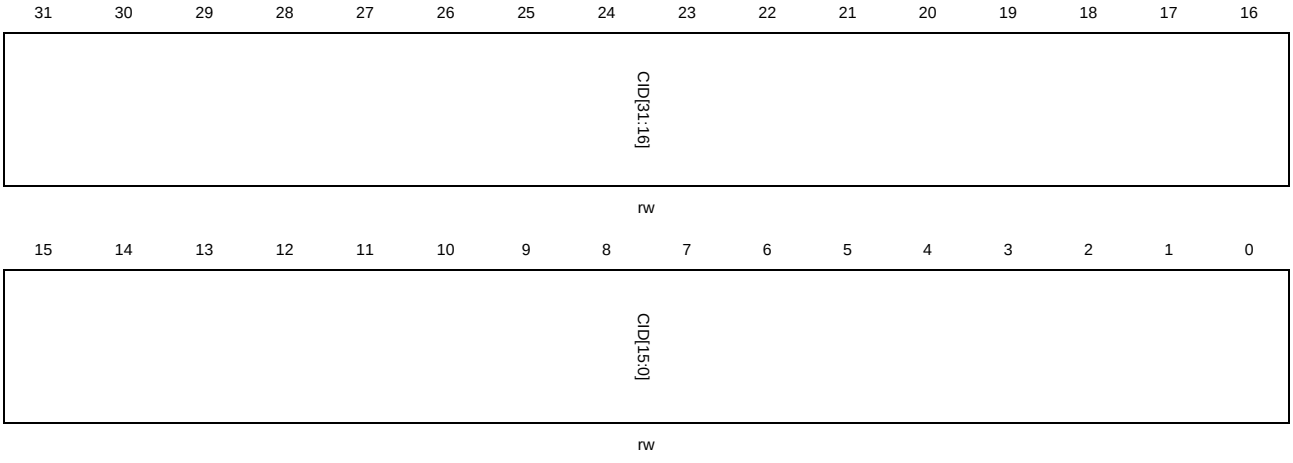
Core ID register (USBFS_CID)

Address offset: 0x003C

Reset value: 0x0000 1000

This register contains the product ID.

This register has to be accessed by word (32-bit)



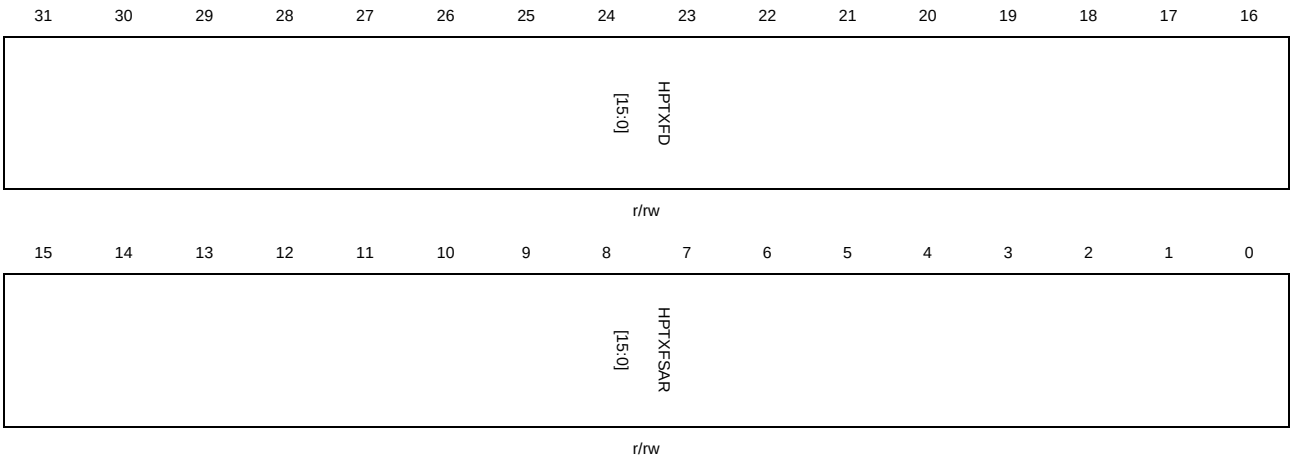
Bits	Fields	Descriptions
31:0	CID[31:0]	Core ID Software can write or read this field and uses this field as a unique ID for its application.

Host periodic Tx FIFO length register (USBFS_HPTFLEN)

Address offset: 0x0100

Reset value: 0x0200 0600

This register has to be accessed by word (32-bit)



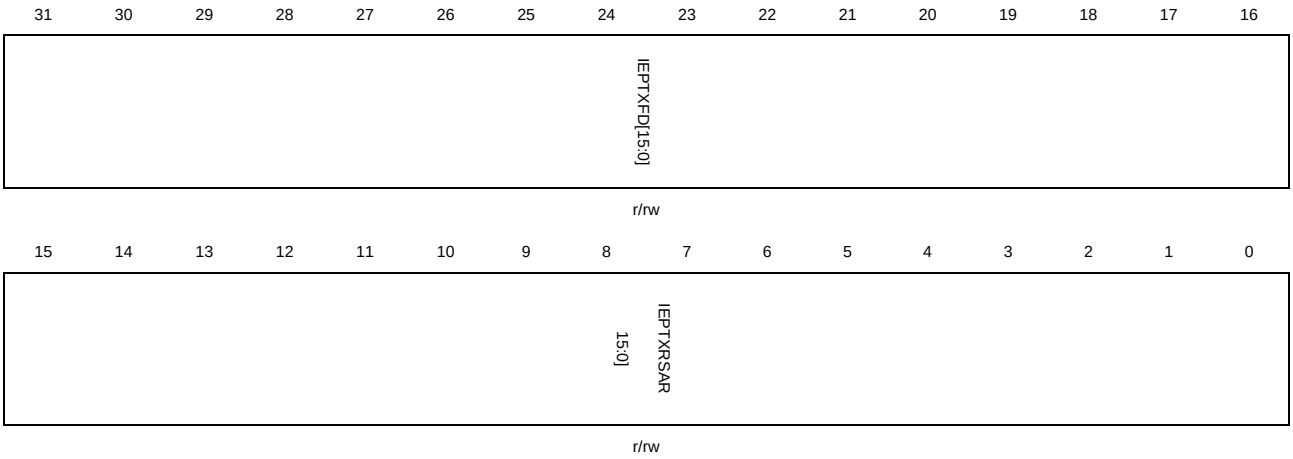
Bits	Fields	Descriptions
31:16	HPTXFD[15:0]	Host periodic Tx FIFO depth In terms of 32-bit words 1≤HPTXFD≤1024
15:0	HPTXFSAR[15:0]	Host periodic Tx FIFO RAM start address The start address for host periodic Tx FIFO RAM is in terms of 32-bit words

Device IN endpoint Tx FIFO length register (USBFS_DIEPxTFLEN) (x = 1…3, where x is the FIFO_number)

Address offset: 0x0104 + (FIFO_number – 1) × 0x04

Reset value: 0x0200 0400

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:16	IEPTXFD[15:0]	IN endpoint Tx FIFO depth In terms of 32-bit words 1≤HPTXFD≤1024
15:0	IEPTXRSAR[15:0]	IN endpoint Tx FIFO RAM start address The start address for IN endpoint Tx FIFO is in terms of 32-bit words.

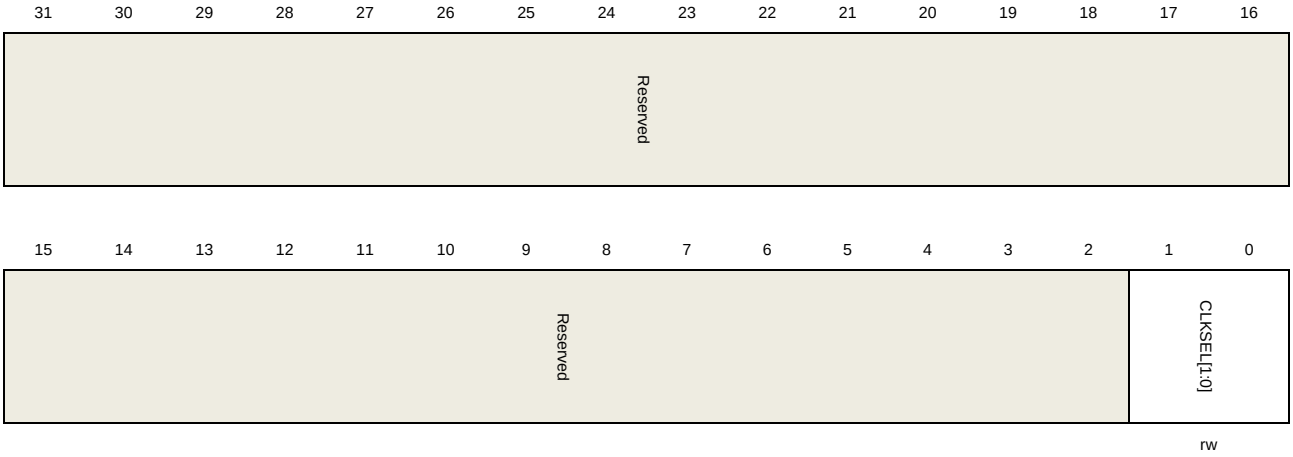
22.7.2. Host control and status registers

Host control register (USBFS_HCTL)

Address offset: 0x0400
Reset value: 0x0000 0000

This register configures the core after power on in host mode. It is not need to modify it after host initialization.

This register has to be accessed by word (32-bit)



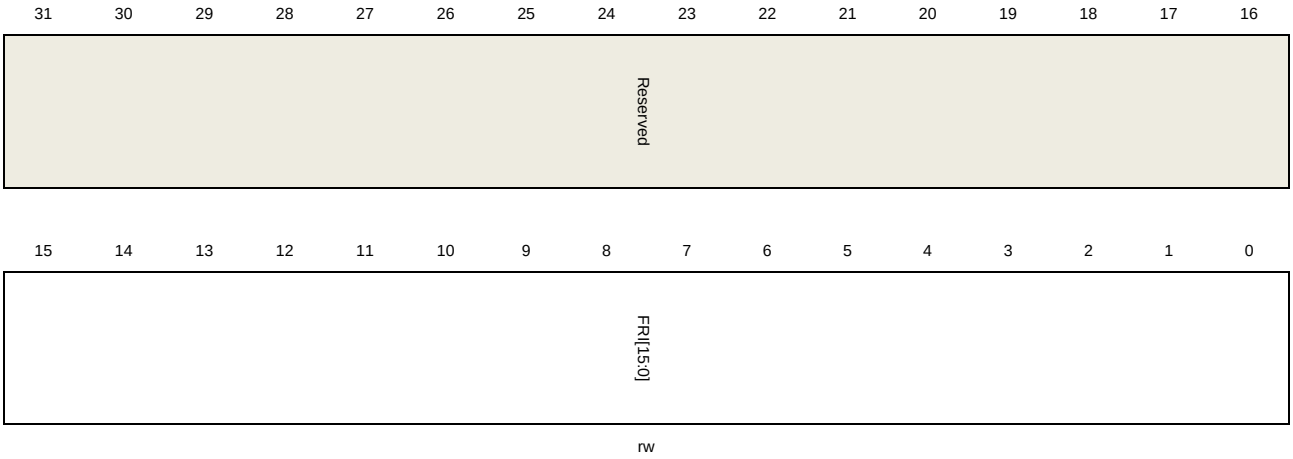
Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1:0	CLKSEL[1:0]	Clock select for USB clock 01: 48MHz clock Others: Reserved

Host frame interval register (USBFS_HFT)

Address offset: 0x0404
Reset value: 0x0000 BB80

This register sets the frame interval when USBFS controller is enumerating USB device.

This register has to be accessed by word (32-bit)



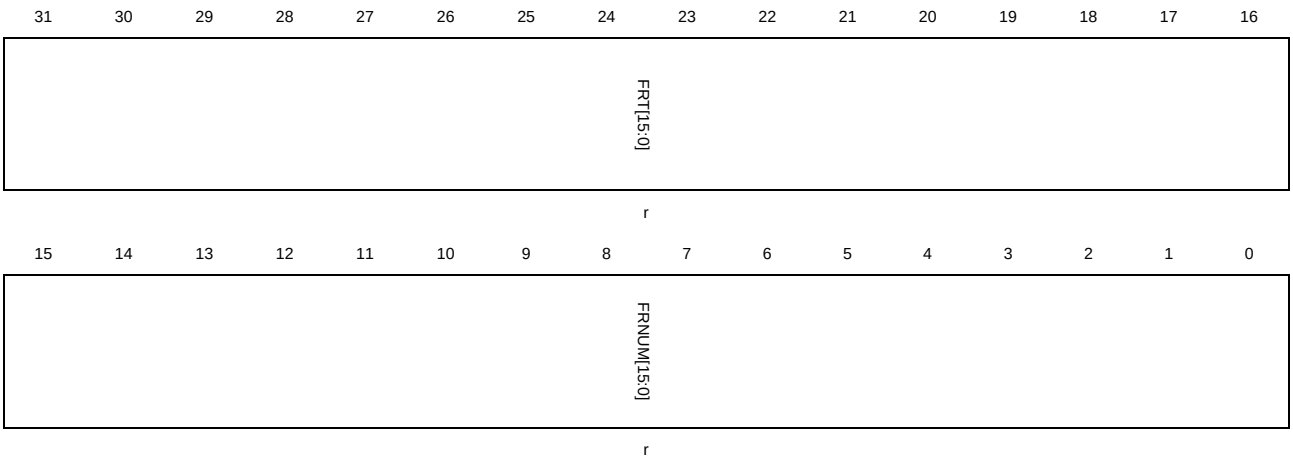
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	FRI[15:0]	Frame interval This value describes the frame time in terms of PHY clocks. Each time when port is enabled after a port reset, USBFS uses a proper value according to the current speed, and software can write to this field to change the value. This value should be calculated using the frequency described below: Full-Speed: 48MHz Low-Speed: 6MHz

Host frame information remaining register (USBFS_HFINFR)

Address offset: 0x408

Reset value: 0xBB80 0000

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions

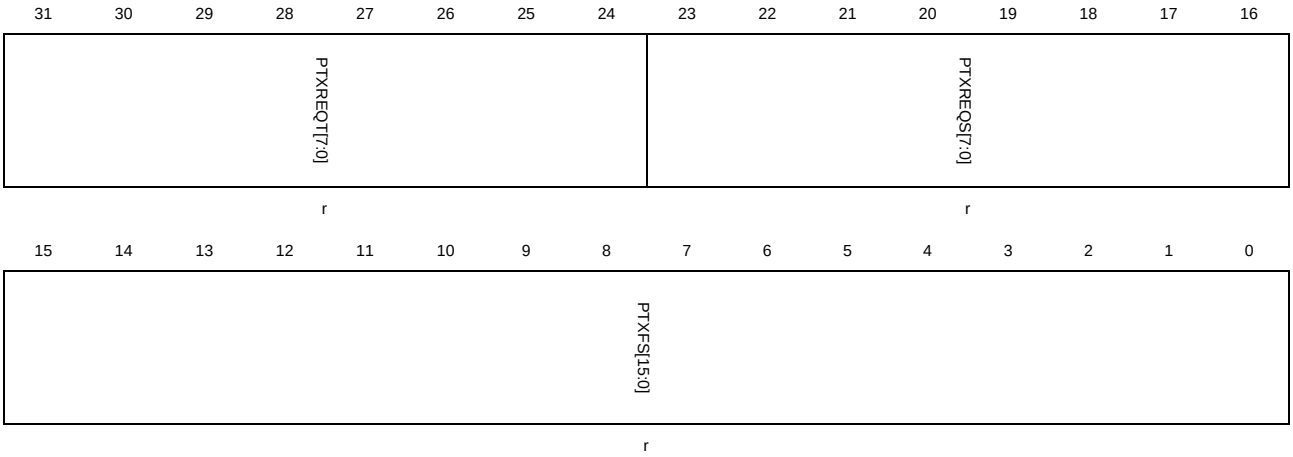
31:16	FRT[15:0]	Frame remaining time This field reports the remaining time of current frame in terms of PHY clock.
15:0	FRNUM[15:0]	Frame number This field reports the frame number of current frame and returns to 0 after it reaches 0x3FFF.

Host periodic Tx FIFO/queue status register (USBFS_HPTFQSTAT)

Address offset: 0x0410
Reset value: 0x0008 0200

This register reports the current status of the host periodic Tx FIFO and request queue. The request queue includes IN, OUT or other request entries in host mode.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:24	PTXREQT[7:0]	Top entry of the periodic Tx request queue Entry in the periodic Tx request queue. Bits 30:27: Channel number Bits 26:25: 00: IN/OUT token 01: Zero-length OUT packet 11: Channel halt request Bit 24: Terminate flag, indicating last entry for selected channel.
23:16	PTXREQS[7:0]	Periodic Tx request queue space The remaining space of the periodic Tx request queue. 0: Request queue is full 1: 1 entry 2: 2 entries … n: n entries (0≤n≤8)

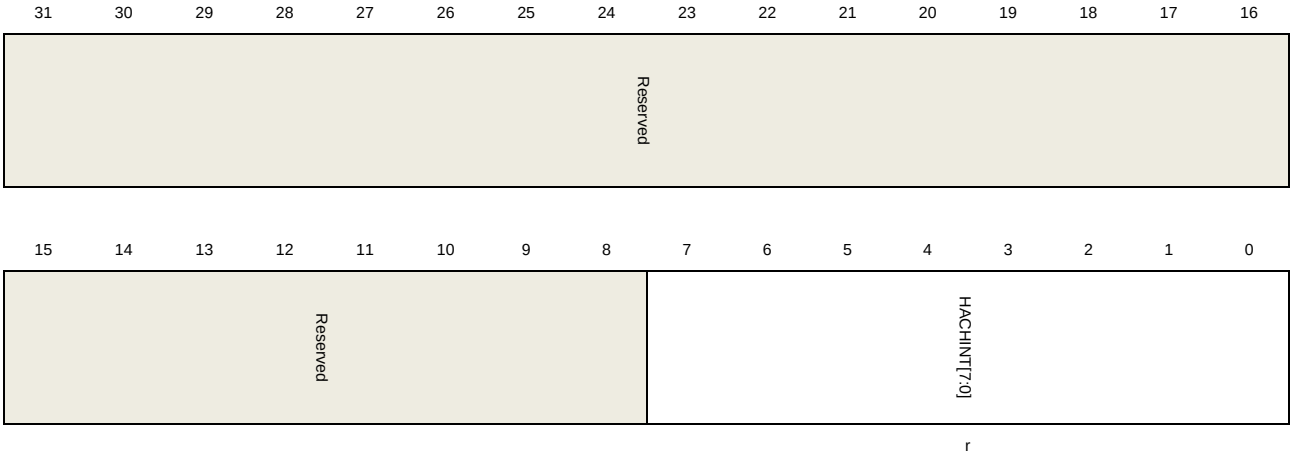
		Others: Reserved
15:0	PTXFS[15:0]	Periodic Tx FIFO space The remaining space of the periodic Tx FIFO. In terms of 32-bit words 0: periodic Tx FIFO is full 1: 1 word 2: 2 words … n: n words (0≤n≤PTXFD) Others: Reserved

Host all channels interrupt register (USBFS_HACHINT)

Address offset: 0x0414
Reset value: 0x0000 0000

When a channel interrupt is triggered, USBFS sets a corresponding bit in this register and software should read this register to know which channel is asserting interrupts.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	HACHINT[7:0]	Host all channel interrupts Each bit represents a channel: Bit 0 for channel 0, bit 7 for channel 7.

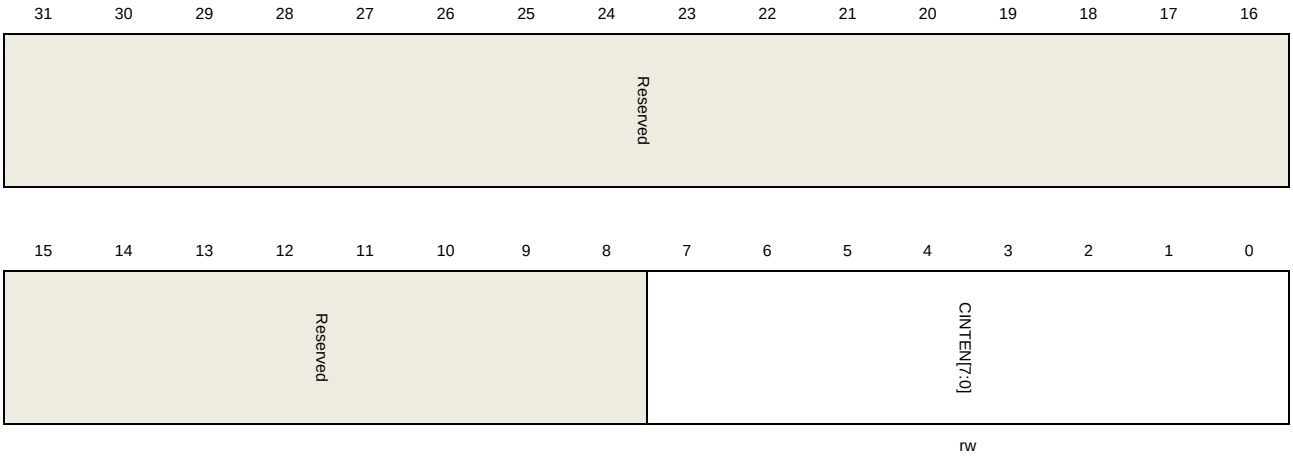
Host all channels interrupt enable register (USBFS_HACHINTEN)

Address offset: 0x0418
Reset value: 0x0000 0000

This register can be used by software to enable or disable a channel's interrupt. Only when the channel whose corresponding bit in this register is set, so as to cause the channel interrupt

flag HCIF set in USBFS_GINTF register.

This register has to be accessed by word (32-bit)



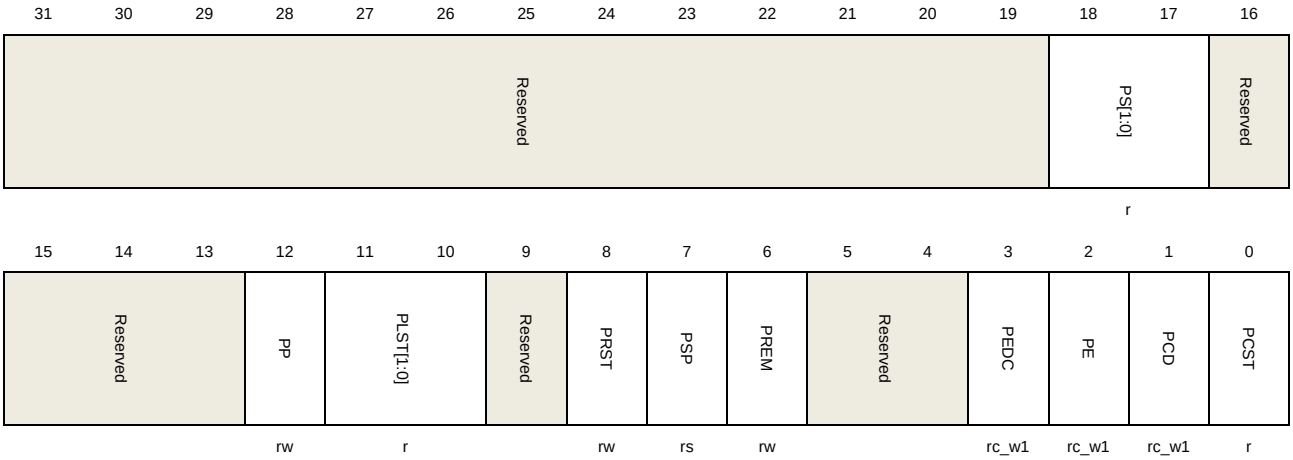
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	CINTEN[7:0]	Channel interrupt enable 0: Disable channel n interrupt 1: Enable channel n interrupt Each bit represents a channel: Bit 0 for channel 0, bit 7 for channel 7.

Host port control and status register (USBFS_HPCS)

Address offset: 0x0440
Reset value: 0x0000 0000

This register controls the port's behavior and also has some flags which report the status of the port. The HPIF flag in USBFS_GINTF register will be triggered if one of these flags (PRST, PEDC and PCD) in this register is set by USBFS.

This register has to be accessed by word (32-bit)



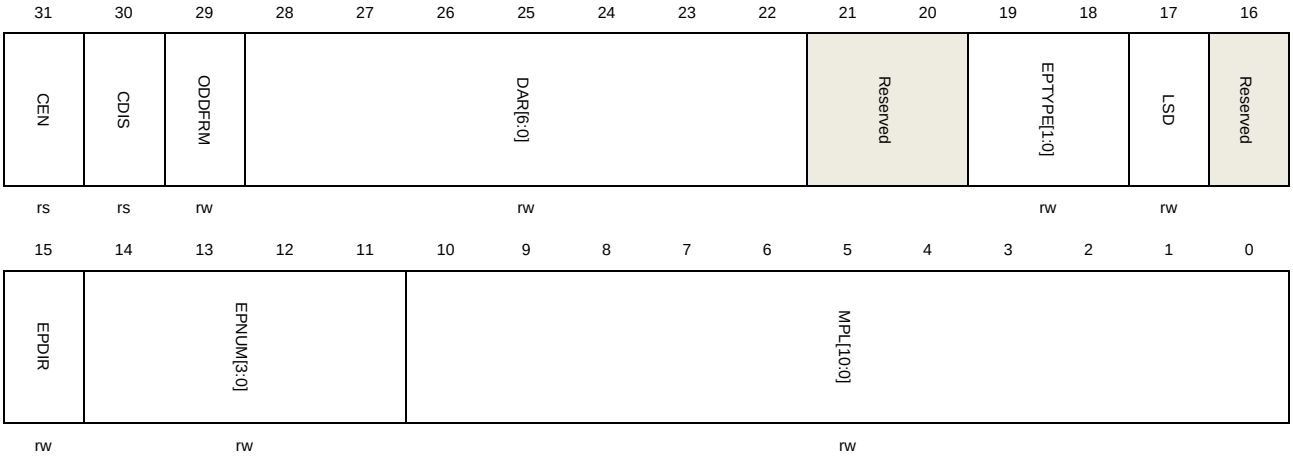
Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18:17	PS[1:0]	Port speed Report the enumerated speed of the device attached to this port. 01: Full speed 10: Low speed Others: Reserved
16:13	Reserved	Must be kept at reset value.
12	PP	Port power This bit should be set before a port is used. Because USBFS doesn't have power supply ability, it only uses this bit to get whether the port is in powered state. Software should ensure the power supply on VBUS before setting this bit. 0: Port is powered off 1: Port is powered on
11:10	PLST[1:0]	Port line status Report the current state of USB data lines. Bit 10: State of DP line Bit 11: State of DM line
9	Reserved	Must be kept at reset value.
8	PRST	Port reset Application sets this bit to start a reset signal on USB port. Application should clear this bit when it wants to stop the reset signal. 0: Port is not in reset state 1: Port is in reset state
7	PSP	Port suspend Application sets this bit to put the port into suspend state. When this bit is set, the port stops sending SOF tokens. This bit can only be cleared by the following operations: – PRST bit in this register is set – PREM bit in this register is set – A remote wakeup signal is detected – A device disconnection is detected 0: Port is not in suspend state 1: Port is in suspend state
6	PREM	Port resume Application sets this bit to start a resume signal on USB port. Application should clear this bit when it wants to stop the resume signal. 0: No resume driven

		1: Resume driven
5:4	Reserved	Must be kept at reset value.
3	PEDC	Port enable/disable change Set by the core when the status of the bit 2 in this register changes.
2	PE	Port enable This bit is automatically set by USBFS after a USB reset signal finishes and cannot be set by software. This bit is cleared by the following events: – A disconnection condition – Software clears this bit 0: Port disabled 1: Port enabled
1	PCD	Port connection detected Set by USBFS when a device connection is detected. This bit can be cleared by writing 1 to this bit.
0	PCST	Port connection status 0: Device is not connected to the port 1: Device is connected to the port

Host channel x control register (USBFS_HCHxCTL) (x = 0…7 where x = channel_number)

Address offset: 0x0500 + (channel_number × 0x20)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31	CEN	Channel enable Set by the application and cleared by USBFS.

		0: Channel disabled 1: Channel enabled Software should follow the operation guide to disable or enable a channel.
30	CDIS	Channel disable Software can set this bit to disable the channel from processing transactions. Software should follow the operation guide to disable or enable a channel.
29	ODDFRM	Odd frame For periodic transfers (interrupt or isochronous transfer), this bit controls that channel's transaction to be processed is in odd frame or even frame. 0: Even frame 1: Odd frame
28:22	DAR[6:0]	Device address The address of the USB device that this channel wants to communicate with.
21:20	Reserved	Must be kept at reset value.
19:18	EPTYPE[1:0]	Endpoint type The transfer type of the endpoint with which this channel communicates. 00: Control 01: Isochronous 10: Bulk 11: Interrupt
17	LSD	Low-speed device The device that this channel communicates with is a low-speed device.
16	Reserved	Must be kept at reset value.
15	EPDIR	Endpoint direction The transfer direction of the endpoint that this channel communicates with. 0: OUT 1: IN
14:11	EPNUM[3:0]	Endpoint number The number of the endpoint that this channel communicates with.
10:0	MPL[10:0]	Maximum packet length The target endpoint's maximum packet length.

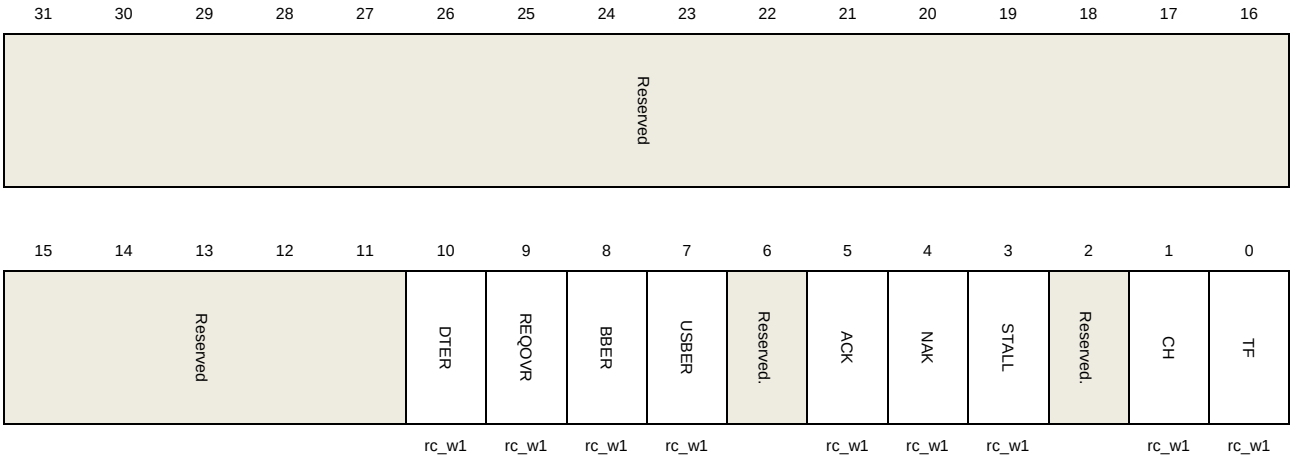
Host channel x interrupt flag register (USBFS_HCHxINTF) (x = 0...7 where x = channel_number)

Address offset: 0x0508 + (channel_number × 0x20)
Reset value: 0x0000 0000

This register contains the status and events of a channel, when a channel interrupt occurs, application should read this register for the respective channel to get the source of the

interrupt. The flag bits in this register are all set by hardware and cleared by writing 1.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10	DTER	Data toggle error The IN transaction gets a data packet but the PID of this packet doesn't match DPID bits in USBFS_HCHxLEN register.
9	REQOVR	Request queue overrun The periodic request queue is full when software starts new transfers.
8	BBER	Babble error A babble condition occurs on USB bus. A typical reason for babble condition is that a device sends a data packet and the packet length exceeds the endpoint's maximum packet length.
7	USBER	USB bus error The USB error flag is set when the following conditions occur during a packet reception: – A received packet has a wrong CRC field – A stuff error detected on USB bus – Timeout when waiting for a response packet
6	Reserved	Must be kept at reset value.
5	ACK	ACK An ACK response is received or transmitted.
4	NAK	NAK A NAK response is received.
3	STALL	STALL A STALL response is received.

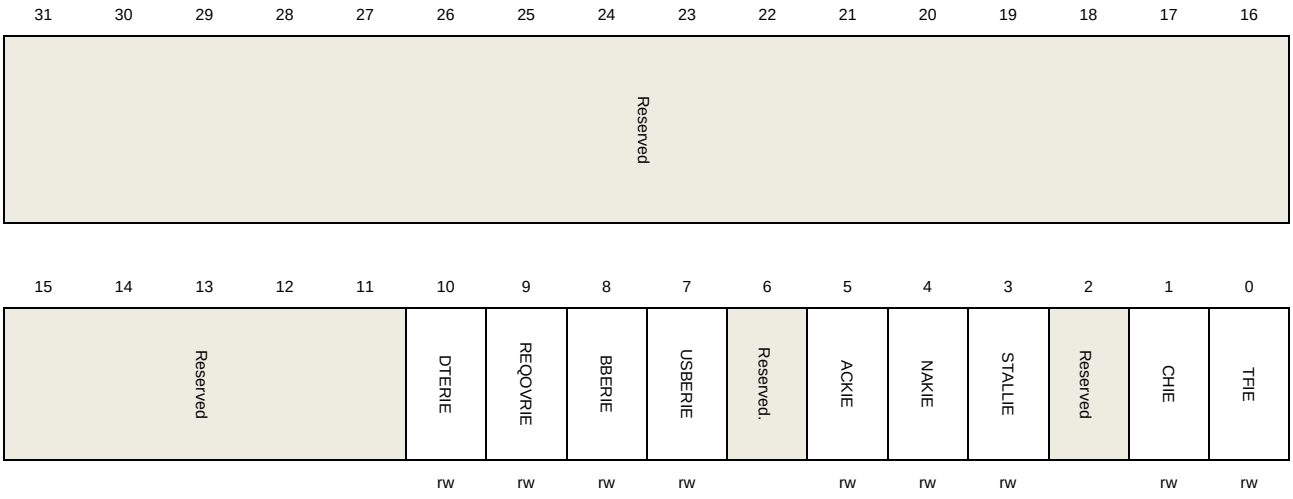
2	Reserved	Must be kept at reset value.
1	CH	Channel halted This channel is disabled by a request, and it will not respond to other requests during the request processing.
0	TF	Transfer finished All the transactions of this channel finish successfully, and no error occurs. For IN channel, this flag will be triggered after PCNT bits in USBFS_HCHxLEN register reach zero. For OUT channel, this flag will be triggered when software reads and pops a TF status entry from the Rx FIFO.

Host channel x interrupt enable register (USBFS_HCHxINTEN) (x = 0…7, where x = channel_number)

Address offset: 0x050C + (channel_number × 0x20)
Reset value: 0x0000 0000

This register contains the interrupt enable bits for the flags in USBFS_HCHxINTF register. If a bit in this register is set by software, the corresponding bit in USBFS_HCHxINTF register is able to trigger a channel interrupt. The bits in this register are set and cleared by software.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10	DTERIE	Data toggle error interrupt enable 0: Disable data toggle error interrupt 1: Enable data toggle error interrupt
9	REQOVRIE	Request queue overrun interrupt enable 0: Disable request queue overrun interrupt

		1: Enable request queue overrun interrupt
8	BBERIE	Babble error interrupt enable 0: Disable babble error interrupt 1: Enable babble error interrupt
7	USBERIE	USB bus error interrupt enable 0: Disable USB bus error interrupt 1: Enable USB bus error interrupt
6	Reserved	Must be kept at reset value.
5	ACKIE	ACK interrupt enable 0: Disable ACK interrupt 1: Enable ACK interrupt
4	NAKIE	NAK interrupt enable 0: Disable NAK interrupt 1: Enable NAK interrupt
3	STALLIE	STALL interrupt enable 0: Disable STALL interrupt 1: Enable STALL interrupt
2	Reserved	Must be kept at reset value.
1	CHIE	Channel halted interrupt enable 0: Disable channel halted interrupt 1: Enable channel halted interrupt
0	TFIE	Transfer finished interrupt enable 0: Disable transfer finished interrupt 1: Enable transfer finished interrupt

Host channel x transfer length register (USBFS_HCHxLEN) (x = 0…7, where x = channel_number)

Address offset: 0x0510 + (channel_number × 0x20)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	DPID[1:0]		PCNT[9:0]										TLEN[18:16]		
	rw		rw										rw		

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

TLEN[15:0]
rw

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:29	DPID[1:0]	DATA PID Software should write this field before the transfer starts. For OUT transfers, this field controls the DATA PID of the first transmitted packet. For IN transfers, this field controls the expected DATA PID of the first received packet, and DTERR will be triggered if the DATA PID doesn't match. After the transfer starts, USBFS changes and toggles this field automatically following the USB protocol. 00: DATA0 10: DATA1 11: SETUP (for control transfer only) 01: Reserved
28:19	PCNT[9:0]	Packet count The number of data packets desired to be transmitted (OUT) or received (IN) in a transfer. Software should program this field before the channel is enabled. After the transfer starts, this field is decreased automatically after each successful data packet transmission.
18:0	TLEN[18:0]	Transfer length The total data bytes number of a transfer. For OUT transfers, this field is the total data bytes of all the data packets desired to be transmitted in an OUT transfer. Software should program this field before the channel is enabled. When software successfully writes a packet into the channel's data Tx FIFO, this field is decreased by the byte size of the packet. For IN transfer each time software reads out a packet from the Rx FIFO, this field is decreased by the byte size of the packet.

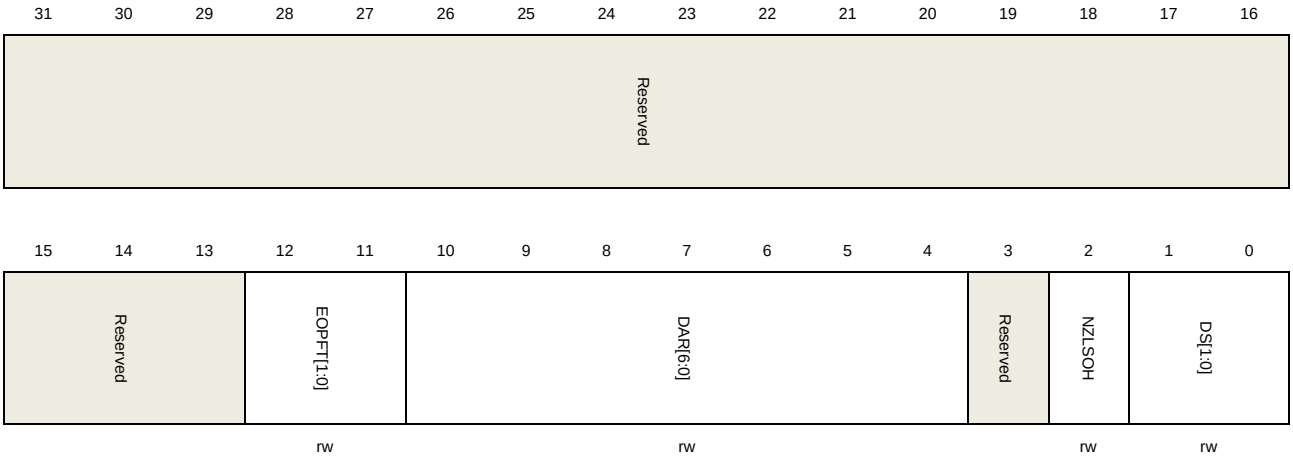
22.7.3. Device control and status registers

Device configuration register (USBFS_DCFG)

Address offset: 0x0800
Reset value: 0x0000 0000

This register configures the core in device mode after power on, certain control commands or enumeration. It is not able to change this register after device initialization.

This register has to be accessed by word (32-bit)



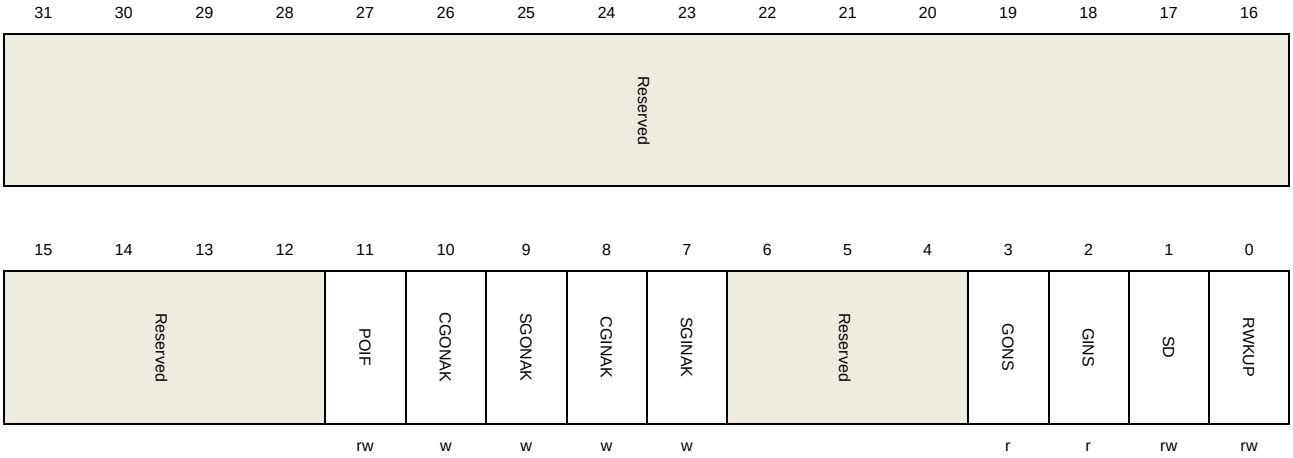
Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:11	EOPFT[1:0]	End of periodic frame time This field defines the percentage time point in a frame that the end of periodic frame (EOPF) flag should be triggered. 00: 80% of the frame time 01: 85% of the frame time 10: 90% of the frame time 11: 95% of the frame time
10:4	DAR[6:0]	Device address This field defines the USB device address. USBFS uses this field to match with the incoming token's device address field. Software should program this field after receiving a set_address command from USB host.
3	Reserved	Must be kept at reset value.
2	NZLSOH	Non-zero-length status OUT handshake When a USB device receives a non-zero-length data packet during status OUT stage, this field controls that USBFS should either receive this packet or reject this packet with a STALL handshake. 0: Treat this packet as a normal packet and response according to the status of NAKS and STALL bits in USBFS_DOEPxCTL register. 1: Send a STALL handshake and don't save the received OUT packet.
1:0	DS[1:0]	Device speed This field controls the device speed when the device connected to a host. 11: Full speed Others: Reserved

Device control register (USBFS_DCTL)

Address offset: 0x0804

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11	POIF	Power-on initialization finished Software should set this bit to notify USBFS that the registers have been initialized after waking up from power down state.
10	CGONAK	Clear global OUT NAK Software sets this bit to clear GONS bit in this register.
9	SGONAK	Set global OUT NAK Software sets this bit to set GONS bit in this register. When GONS bit is zero, setting this bit will also cause GONAK flag in USBFS_GINTF register to be triggered after a while. Software should clear the GONAK flag before writing this bit again.
8	CGINAK	Clear global IN NAK Software sets this bit to clear GINS bit in this register.
7	SGINAK	Set global IN NAK Software sets this bit to set GINS bit in this register. When GINS bit is zero, setting this bit will also cause GINAK flag in USBFS_GINTF register to be triggered after a while. Software should clear the GINAK flag before writing this bit again.
6:4	Reserved	Must be kept at reset value.
3	GONS	Global OUT NAK status 0: The handshake that USBFS responds to OUT transaction packet and whether to

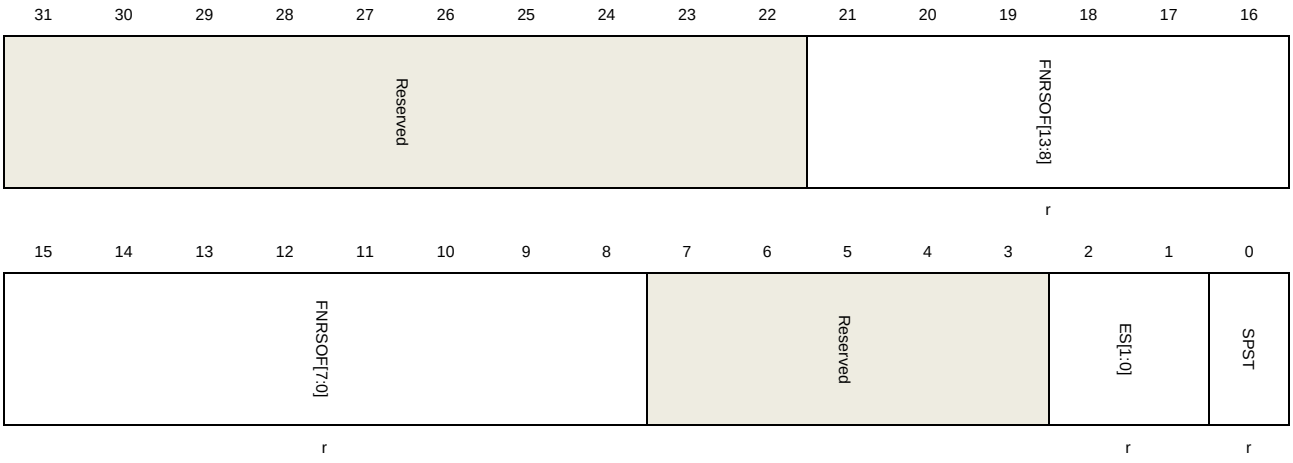
		save the OUT data packet are decided by Rx FIFO status, endpoint's NAK and STALL bits 1: USBFS always responds to OUT transaction with NAK handshake and doesn't save the incoming OUT data packet
2	GINS	Global IN NAK status 0: The response to IN transaction is decided by Tx FIFO status, endpoint's NAK and STALL bits 1: USBFS always responds to IN transaction with a NAK handshake
1	SD	Soft disconnect Software can use this bit to generate a soft disconnection condition on USB bus. After this bit is set, USBFS switches off the pull-up resistor on DP line. This will cause the host to detect a device disconnection. 0: No soft disconnection generated 1: Generate a soft disconnection
0	RWKUP	Remote wakeup In suspend state, software can use this bit to generate a remote wakeup signal to inform host that it should resume the USB bus. 0: No remote wakeup signal generated 1: Generate remote wakeup signal

Device status register (USBFS_DSTAT)

Address offset: 0x0808
Reset value: 0x0000 0000

This register contains status and information of the USBFS in device mode.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.

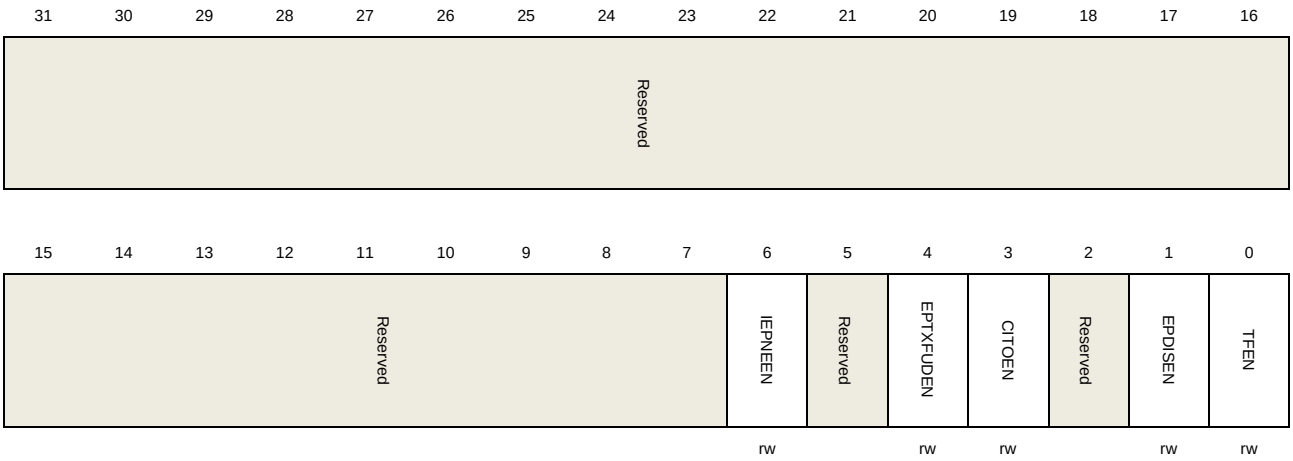
21:8	FNRSOF[13:0]	The frame number of the received SOF USBFS always updates this field after receiving a SOF token.
7:3	Reserved	Must be kept at reset value.
2:1	ES[1:0]	Enumerated speed This field reports the enumerated device speed. Read this field after the ENUMF flag in USBFS_GINTF register is triggered. 11: Full speed Others: Reserved
0	SPST	Suspend status This bit reports whether device is in suspend state. 0: Device is not in suspend state 1: Device is in suspend state

Device IN endpoint common interrupt enable register (USBFS_DIEPINTEN)

Address offset: 0x0810
Reset value: 0x0000 0000

This register contains the interrupt enable bits for the USBFS_DIEPxINTF register. If a bit in this register is set by software, the corresponding bit in USBFS_DIEPxINTF register is able to trigger an endpoint interrupt in USBFS_DAEPINT register. The bits in this register are set and cleared by software.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	IEPNEEN	IN endpoint NAK effective interrupt enable bit 0: Disable IN endpoint NAK effective interrupt 1: Enable IN endpoint NAK effective interrupt

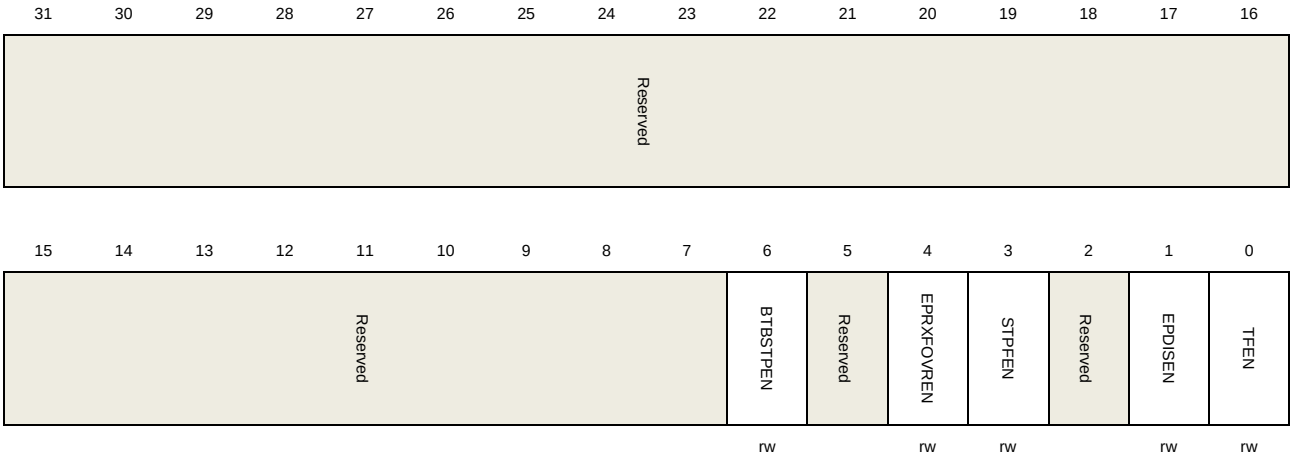
5	Reserved	Must be kept at reset value.
4	EPTXFUDEN	Endpoint Tx FIFO underrun interrupt enable bit 0: Disable endpoint Tx FIFO underrun interrupt 1: Enable endpoint Tx FIFO underrun interrupt
3	CITOEN	Control IN timeout interrupt enable bit 0: Disable control IN timeout interrupt 1: Enable control IN timeout interrupt
2	Reserved	Must be kept at reset value.
1	EPDISEN	Endpoint disabled interrupt enable bit 0: Disable endpoint disabled interrupt 1: Enable endpoint disabled interrupt
0	TFEN	Transfer finished interrupt enable bit 0: Disable transfer finished interrupt 1: Enable transfer finished interrupt

Device OUT endpoint common interrupt enable register (USBFS_DOEPINTEN)

Address offset: 0x0814
Reset value: 0x0000 0000

This register contains the interrupt enable bits for the USBFS_DOEPxINTF register. If a bit in this register is set by software, the corresponding bit in USBFS_DOEPxINTF register is able to trigger an endpoint interrupt in USBFS_DAEPINT register. The bits in this register are set and cleared by software.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	BTBSTPEN	Back-to-back SETUP packets (only for control OUT endpoint) interrupt enable bit

		0: Disable back-to-back SETUP packets interrupt 1: Enable back-to-back SETUP packets interrupt
5	Reserved	Must be kept at reset value.
4	EPRXFOVREN	Endpoint Rx FIFO overrun interrupt enable bit 0: Disable endpoint Rx FIFO overrun interrupt 1: Enable endpoint Rx FIFO overrun interrupt
3	STPFEN	SETUP phase finished (only for control OUT endpoint) interrupt enable bit 0: Disable SETUP phase finished interrupt 1: Enable SETUP phase finished interrupt
2	Reserved	Must be kept at reset value.
1	EPDISEN	Endpoint disabled interrupt enable bit 0: Disable endpoint disabled interrupt 1: Enable endpoint disabled interrupt
0	TFEN	Transfer finished interrupt enable bit 0: Disable transfer finished interrupt 1: Enable transfer finished interrupt

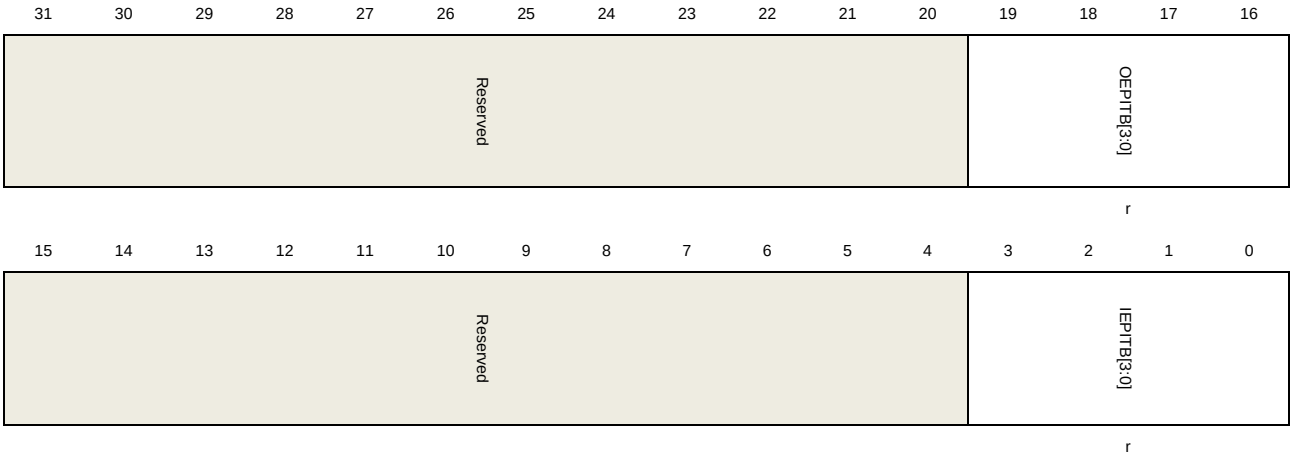
Device all endpoints interrupt register (USBFS_DAEPINT)

Address offset: 0x0818

Reset value: 0x0000 0000

When an endpoint interrupt is triggered, USBFS sets corresponding bit in this register and software should read this register to get which endpoint is asserting an interrupt.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.

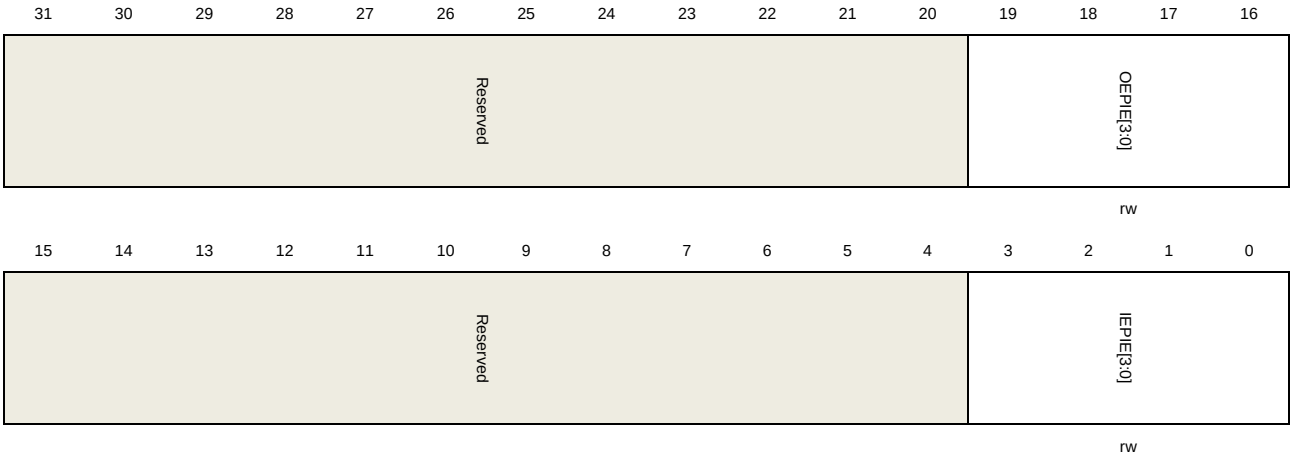
19:16	OEPITB[3:0]	Device all OUT endpoint interrupt bits Each bit represents an OUT endpoint: Bit 16 for OUT endpoint 0, bit 19 for OUT endpoint 3.
15:4	Reserved	Must be kept at reset value.
3:0	IEPITB[3:0]	Device all IN endpoint interrupt bits Each bit represents an IN endpoint: Bit 0 for IN endpoint 0, bit 3 for IN endpoint 3.

Device all endpoints interrupt enable register (USBFS_DAEPINTEN)

Address offset: 0x081C
Reset value: 0x0000 0000

This register can be used by software to enable or disable an endpoint's interrupt. Only when the endpoint whose corresponding bit in this register is set, it is able to trigger the endpoint interrupt flag OEPIF or IEPIF in USBFS_GINTF register.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19:16	OEPIE[3:0]	Out endpoint interrupt enable 0: Disable OUT endpoint n interrupt 1: Enable OUT endpoint n interrupt Each bit represents an OUT endpoint: Bit 16 for OUT endpoint 0, bit 19 for OUT endpoint 3.
15:4	Reserved	Must be kept at reset value.
3:0	IEPIE[3:0]	IN endpoint interrupt enable bits 0: Disable IN endpoint n interrupt 1: Enable IN endpoint n interrupt

Each bit represents an IN endpoint:

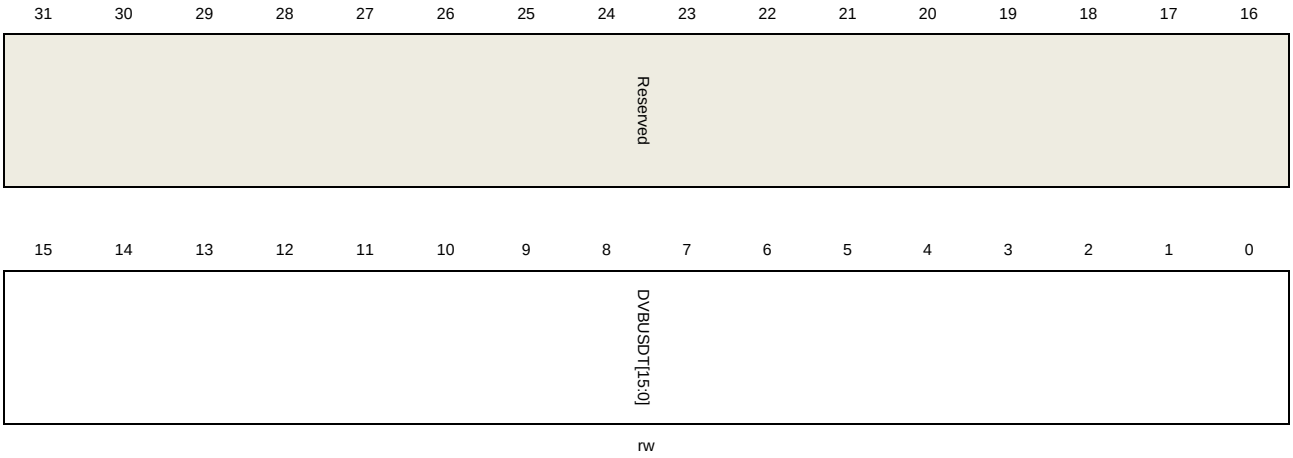
Bit 0 for IN endpoint 0, bit 3 for IN endpoint 3.

Device VBUS discharge time register (USBFS_DVBUSDT)

Address offset: 0x0828

Reset value: 0x0000 17D7

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	DVBUSDT[15:0]	Device V_{BUS} discharge time There is a discharge process after V_{BUS} pulsing in SRP protocol. This field defines the discharge time of V_{BUS}. The actual discharge time is 1024 * DVBUSDT[15:0] * $T_{USBCLOCK}$, where $T_{USBCLOCK}$ is the period time of USB clock.

Device VBUS pulsing time register (USBFS_DVBUSPT)

Address offset: 0x082C

Reset value: 0x0000 05B8

This register has to be accessed by word (32-bit)





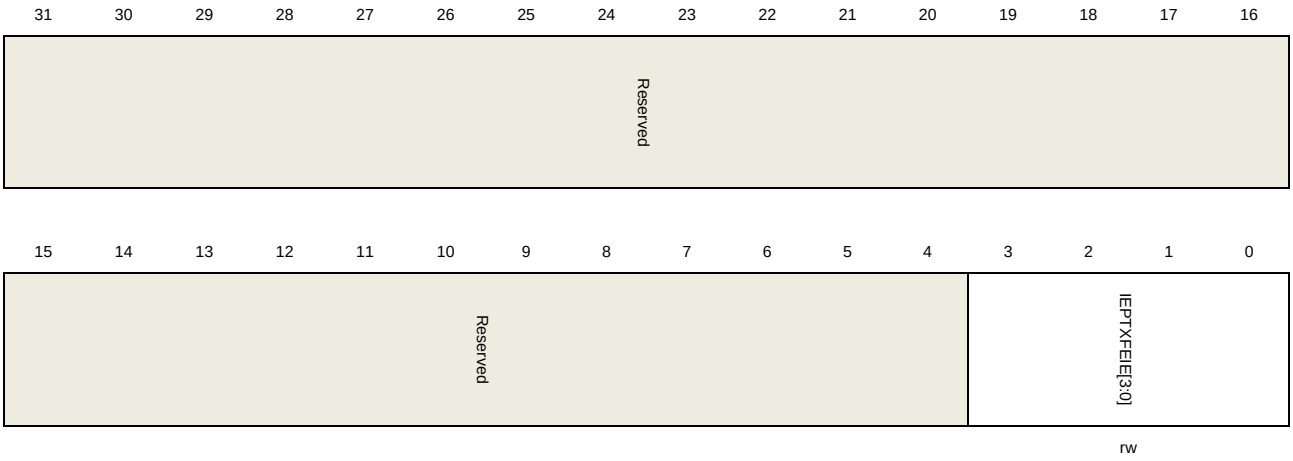
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	DVBUSPT[11:0]	Device V_{BUS} pulsing time This field defines the pulsing time for V_{BUS}. The actual pulsing time is 1024*DVBUSPT[11:0] *$T_{USBCLOCK}$, where $T_{USBCLOCK}$ is the period time of USB clock.

Device IN endpoint FIFO empty interrupt enable register (USBFS_DIEPFEINTEN)

Address offset: 0x0834
Reset value: 0x0000 0000

This register contains the enable bits for the Tx FIFO empty interrupts of IN endpoints.

This register has to be accessed by word (32-bit)



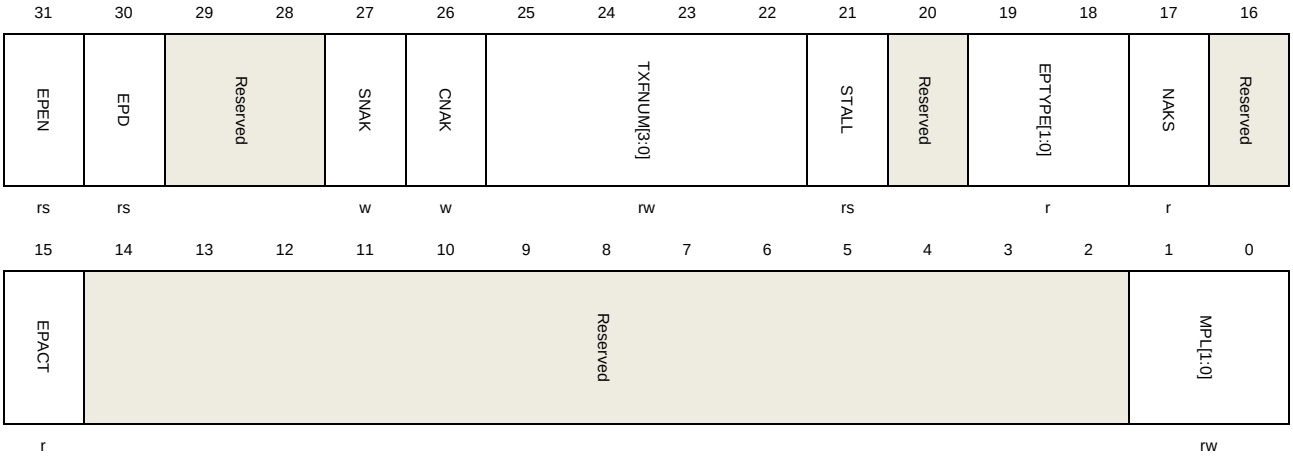
Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	IEPTXFEIE[3:0]	IN endpoint Tx FIFO empty interrupt enable bits This field controls whether the TXFE bits in USBFS_DIEPxINTF registers are able to set an endpoint interrupt bit in USBFS_DAEPINT register. Bit 0 for IN endpoint 0, bit 3 for IN endpoint 3 0: Disable FIFO empty interrupt 1: Enable FIFO empty interrupt

Device IN endpoint 0 control register (USBFS_DIEP0CTL)

Address offset: 0x0900

Reset value: 0x0000 8000

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31	EPEN	Endpoint enable Set by the application and cleared by USBFS. 0: Endpoint disabled 1: Endpoint enabled Software should follow the operation guide to disable or enable an endpoint.
30	EPD	Endpoint disable Software can set this bit to disable the endpoint. Software should follow the operation guide to disable or enable an endpoint.
29:28	Reserved	Must be kept at reset value.
27	SNAK	Set NAK Software sets this bit to set NAKS bit in this register.
26	CNAK	Clear NAK Software sets this bit to clear NAKS bit in this register.
25:22	TXFNUM[3:0]	Tx FIFO number Define the Tx FIFO number of IN endpoint 0.
21	STALL	STALL handshake Software can set this bit to send STALL handshake when receiving IN token. USBFS will clear this bit after a SETUP token is received on the corresponding OUT endpoint 0. This bit has a higher priority than NAKS bit in this register and GINS bit in USBFS_DCTL register. If both STALL and NAKS bits are set, the STALL bit takes effect.

20	Reserved	Must be kept at reset value.
19:18	EPTYPE[1:0]	Endpoint type This field is fixed to '00' for control endpoint.
17	NAKS	NAK status This bit controls the NAK status of USBFS when both STALL bit in this register and GINS bit in USBFS_DCTL register are cleared: 0: USBFS sends data or handshake packets according to the status of the endpoint's Tx FIFO. 1: USBFS always sends NAK handshake to the IN token. This bit is read-only and software should use CNAK and SNAK in this register to control this bit.
16	Reserved	Must be kept at reset value.
15	EPACT	Endpoint active This field is fixed to '1' for endpoint 0.
14:2	Reserved	Must be kept at reset value.
1:0	MPL[1:0]	Maximum packet length This field defines the maximum packet length for a control data packet. As described in USB 2.0 protocol, there are 4 kinds of length for control transfers: 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes

Device IN endpoint x control register (USBFS_DIEPxCTL) (x = 1…3, where x = endpoint_number)

Address offset: 0x0900 + (endpoint_number × 0x20)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPEN	EPD	SODDFRM/SD1PID	SD0PID/SEVNFRM	SNAK	CNAK	TXFNUM[3:0]				STALL	Reserved	EPTYPE[1:0]		NAKS	EOFRM/DPID
rs	rs	w	w	w	w	rw				rw/rs		rw		r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPACT	Reserved				MPL[10:0]										
rw					rw										

Bits	Fields	Descriptions
31	EPEN	Endpoint enable Set by the application and cleared by USBFS. 0: Endpoint disabled 1: Endpoint enabled Software should follow the operation guide to disable or enable an endpoint.
30	EPD	Endpoint disable Software can set this bit to disable the endpoint. Software should follow the operation guide to disable or enable an endpoint.
29	SODDFRM	Set odd frame (for isochronous IN endpoints) This bit has effect only if this is an isochronous IN endpoint. Software sets this bit to set EOFRM bit in this register.
	SD1PID	Set DATA1 PID (for interrupt/bulk IN endpoints) Software sets this bit to set DPID bit in this register.
28	SEVENFRM	Set even frame (for isochronous IN endpoints) Software sets this bit to clear EOFRM bit in this register.
	SD0PID	Set DATA0 PID (for interrupt/bulk IN endpoints) Software sets this bit to clear DPID bit in this register.
27	SNAK	Set NAK Software sets this bit to set NAKS bit in this register.
26	CNAK	Clear NAK Software sets this bit to clear NAKS bit in this register.
25:22	TXFNUM[3:0]	Tx FIFO number Define the Tx FIFO number of this IN endpoint.
21	STALL	STALL handshake Software can set this bit to send STALL handshake when receiving IN token. This bit has a higher priority than NAKS bit in this register and GINS bit in USBFS_DCTL register. If both STALL and NAKS bits are set, the STALL bit takes effect. For control IN endpoint: Only USBFS can clear this bit when a SETUP token is received on the corresponding OUT endpoint. Software is not able to clear it. For interrupt or bulk IN endpoint: Only software can clear this bit
20	Reserved	Must be kept at reset value.
19:18	EPTYPE[1:0]	Endpoint type This field defines the transfer type of this endpoint: 00: Control

		01: Isochronous 10: Bulk 11: Interrupt
17	NAKS	NAK status This bit controls the NAK status of USBFS when both STALL bit in this register and GINS bit in USBFS_DCTL register are cleared: 0: USBFS sends data or handshake packets according to the status of the endpoint's Tx FIFO. 1: USBFS always sends NAK handshake to the IN token. This bit is read-only and software should use CNAK and SNAK in this register to control this bit.
16	EOFRM	Even/odd frame (for isochronous IN endpoints) For isochronous transfers, software can use this bit to control that USBFS only sends data packets for IN tokens in even or odd frames. If the parity of the current frame number doesn't match with this bit, USBFS only responds with a zero-length packet. 0: Only sends data in even frames 1: Only sends data in odd frames
	DPID	Endpoint DATA PID (for interrupt/bulk IN endpoints) There is a DATA PID toggle scheme in interrupt or bulk transfer. Set SD0PID to set this bit before a transfer starts and USBFS maintains this bit during transfers according to the data toggle scheme described in USB protocol. 0: Data packet's PID is DATA0 1: Data packet's PID is DATA1
15	EPACT	Endpoint active This bit controls whether this endpoint is active. If an endpoint is not active, it ignores all tokens and doesn't make any response.
14:11	Reserved	Must be kept at reset value.
10:0	MPL[10:0]	This field defines the maximum packet length in byte

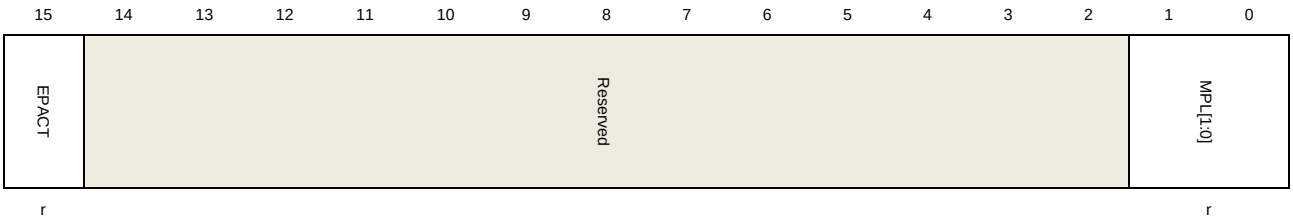
Device OUT endpoint 0 control register (USBFS_DOEP0CTL)

Address offset: 0x0B00

Reset value: 0x0000 8000

This register has to be accessed by word (32-bit)

31	30	29 28	27	26	25 24 23 22	21	20	19 18	17	16
EPEN	EPD	Reserved.	SNAK	CNAK	Reserved	STALL	SNOOP	EPTYPE[1:0]	NAKS	Reserved
rs	r		w	w		rs	rw	r	r	



Bits	Fields	Descriptions
31	EPEN	Endpoint enable Set by the application and cleared by USBFS. 0: Endpoint disabled 1: Endpoint enabled Software should follow the operation guide to disable or enable an endpoint.
30	EPD	Endpoint disable This bit is fixed to 0 for OUT endpoint 0.
29:28	Reserved	Must be kept at reset value.
27	SNAK	Set NAK Software sets this bit to set NAKS bit in this register.
26	CNAK	Clear NAK Software sets this bit to clear NAKS bit in this register
25:22	Reserved	Must be kept at reset value.
21	STALL	STALL handshake Set this bit to send STALL handshake during an OUT transaction. USBFS will clear this bit after a SETUP token is received on OUT endpoint 0. This bit has a higher priority than NAKS bit in this register, i.e. if both STALL and NAKS bits are set, the STALL bit takes effect.
20	SNOOP	Snoop mode This bit controls the snoop mode of an OUT endpoint. In snoop mode, USBFS doesn't check the received data packet's CRC value. 0:Snoop mode disabled 1:Snoop mode enabled
19:18	EPTYPE[1:0]	Endpoint type This field is fixed to '00' for control endpoint.
17	NAKS	NAK status This bit controls the NAK status of USBFS when both STALL bit in this register and GONS bit in USBFS_DCTL register are cleared: 0: USBFS sends data or handshake packets according to the status of the endpoint's Rx FIFO. 1: USBFS always sends NAK handshake for the OUT token.

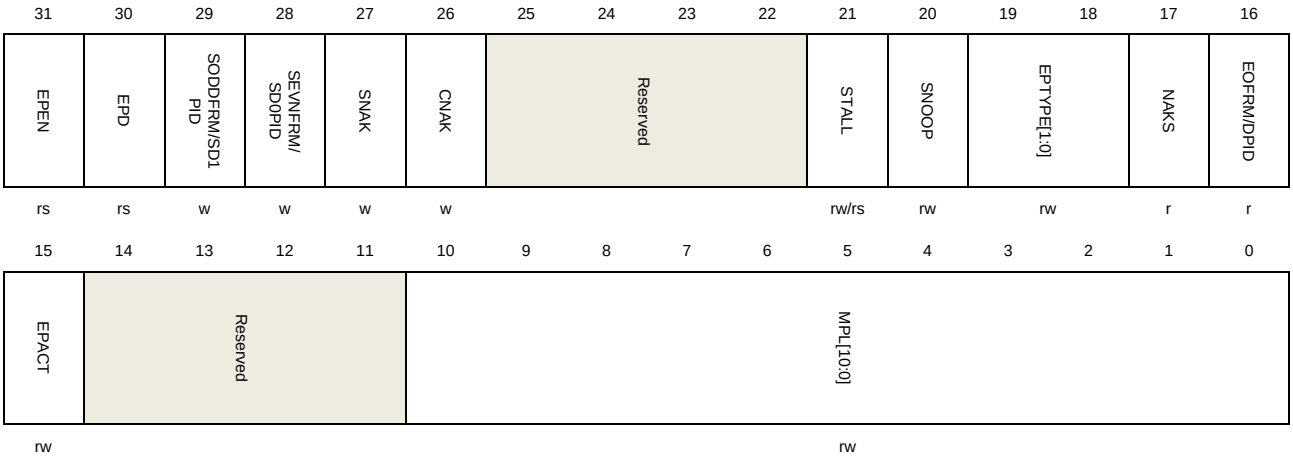
		This bit is read-only and software should use CNAK and SNAK in this register to control this bit.
16	Reserved	Must be kept at reset value.
15	EPACT	Endpoint active This field is fixed to '1' for endpoint 0.
14:2	Reserved	Must be kept at reset value.
1:0	MPL[1:0]	Maximum packet length This is a read-only field, and its value comes from the MPL field of USBFS_DIEP0CTL register: 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes

Device OUT endpoint x control register (USBFS_DOEPxCTL) (x = 1…3, where x = endpoint_number)

Address offset: 0x0B00 + (endpoint_number × 0x20)
Reset value: 0x0000 0000

The application uses this register to control the operations of each logical OUT endpoint except OUT endpoint 0.

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31	EPEN	Endpoint enable Set by the application and cleared by USBFS. 0: Endpoint disabled 1: Endpoint enabled

		Software should follow the operation guide to disable or enable an endpoint.
30	EPD	Endpoint disable Software can set this bit to disable the endpoint. Software should follow the operation guide to disable or enable an endpoint.
29	SODDFRM	Set odd frame (for isochronous OUT endpoints) This bit has effect only if this is an isochronous OUT endpoint. Software sets this bit to set EOFRM bit in this register.
	SD1PID	Set DATA1 PID (for interrupt/bulk OUT endpoints) Software sets this bit to set DPID bit in this register.
28	SEVENFRM	Set even frame (for isochronous OUT endpoints) Software sets this bit to clear EOFRM bit in this register.
	SD0PID	Set DATA0 PID (for interrupt/bulk OUT endpoints) Software sets this bit to clear DPID bit in this register.
27	SNAK	Set NAK Software sets this bit to set NAKS bit in this register.
26	CNAK	Clear NAK Software sets this bit to clear NAKS bit in this register.
25:22	Reserved	Must be kept at reset value.
21	STALL	STALL handshake Software can set this bit to send STALL handshake during an OUT transaction. This bit has a higher priority than NAKS bit in this register and GINAK in USBFS_DCTL register. If both STALL and NAKS bits are set, the STALL bit takes effect. For control OUT endpoint: Only USBFS can clear this bit when a SETUP token is received on the corresponding OUT endpoint. Software is not able to clear it. For interrupt or bulk OUT endpoint: Only software can clear this bit.
20	SNOOP	Snoop mode This bit controls the snoop mode of an OUT endpoint. In snoop mode, USBFS doesn’t check the received data packet’s CRC value. 0:Snoop mode disabled 1:Snoop mode enabled
19:18	EPTYPE[1:0]	Endpoint type This field defines the transfer type of this endpoint: 00: Control 01: Isochronous 10: Bulk 11: Interrupt

17	NAKS	NAK status This bit controls the NAK status of USBFS when both STALL bit in this register and GONS bit in USBFS_DCTL register are cleared: 0: USBFS sends handshake packets according to the status of the endpoint's Rx FIFO. 1: USBFS always sends NAK handshake to the OUT token. This bit is read-only and software should use CNAK and SNAK in this register to control this bit.
16	EOFRM	Even/odd frame (for isochronous OUT endpoints) For isochronous transfers, software can use this bit to control that USBFS only receives data packets in even or odd frames. If the parity of current frame number doesn't match with this bit, USBFS just drops the data packet. 0: Only sends data in even frames 1: Only sends data in odd frames
	DPID	Endpoint data PID (for interrupt/bulk OUT endpoints) These is a data PID toggle scheme in interrupt or bulk transfer. Software should set SD0PID to set this bit before a transfer starts and USBFS maintains this bit during transfers following the data toggle scheme described in USB protocol. 0: Data packet's PID is DATA0 1: Data packet's PID is DATA1
15	EPACT	Endpoint active This bit controls whether this endpoint is active. If an endpoint is not active, it ignores all tokens and doesn't make any response.
14:11	Reserved	Must be kept at reset value.
10:0	MPL[10:0]	This field defines the maximum packet length in bytes.

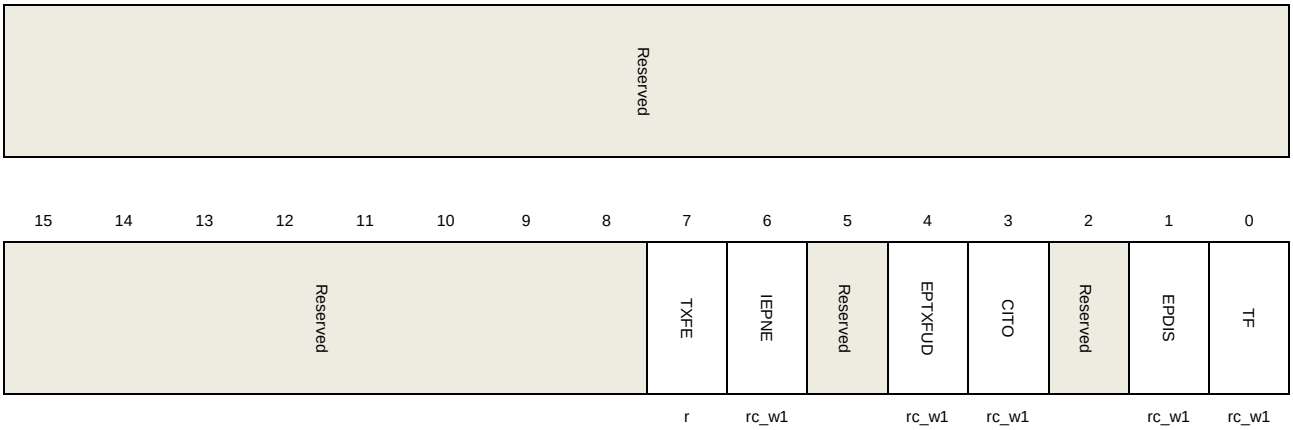
Device IN endpoint x interrupt flag register (USBFS_DIEPxINTF) (x = 0...3, where x = endpoint_number)

Address offset: 0x0908 + (endpoint_number × 0x20)
Reset value: 0x0000 0080

This register contains the status and events of an IN endpoint, when an IN endpoint interrupt occurs, read this register for the respective endpoint to get the source of the interrupt. The flag bits in this register are all set by hardware and cleared by writing 1 except the read-only TXFE bit.

This register has to be accessed by word (32-bit)

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



Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	TXFE	Tx FIFO empty The Tx FIFO of this IN endpoint has reached the empty threshold value defined by TXFTH field in USBFS_GAHBCS register.
6	IEPNE	IN endpoint NAK effective The setting of SNAK bit in USBFS_DIEPxCTL register takes effect. This bit can be cleared either by writing 1 to it or by setting CNAK bit in USBFS_DIEPxCTL register.
5	Reserved	Must be kept at reset value.
4	EPTXFUD	Endpoint Tx FIFO underrun This flag is triggered if the Tx FIFO has no packet data to send when an IN token is received.
3	CITO	Control IN timeout interrupt This flag is triggered if the device waiting for a handshake is timeout in a control IN transaction.
2	Reserved	Must be kept at reset value.
1	EPDIS	Endpoint disabled This flag is triggered when an endpoint is disabled by the software's request.
0	TF	Transfer finished This flag is triggered when all the IN transactions assigned to this endpoint have been finished.

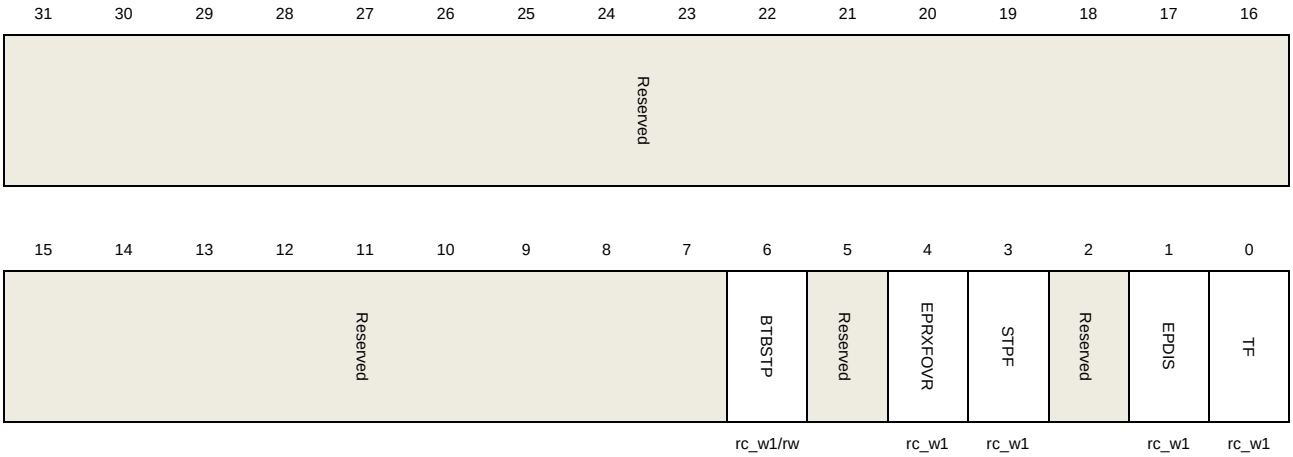
Device OUT endpoint x interrupt flag register (USBFS_DOEPxINTF) (x = 0...3, where x = endpoint_number)

Address offset: 0x0B08 + (endpoint_number × 0x20)
Reset value: 0x0000 0000

This register contains the status and events of an OUT endpoint, when an OUT endpoint

interrupt occurs, read this register for the respective endpoint to get the source of the interrupt. The flag bits in this register are all set by hardware and cleared by writing 1.

This register has to be accessed by word (32-bit)



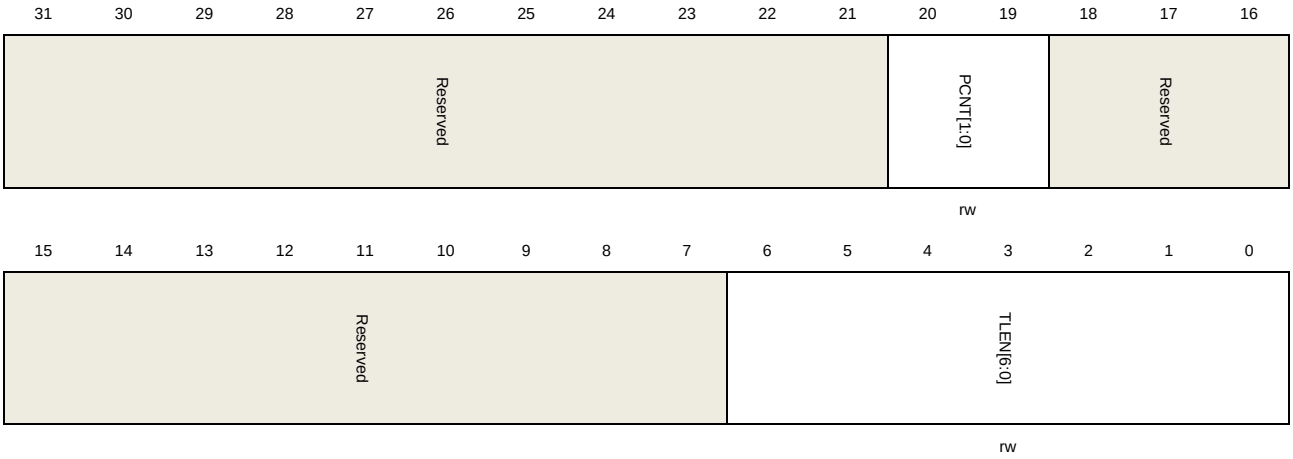
Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	BTBSTP	Back-to-back SETUP packets (Only for control OUT endpoint) This flag is triggered when a control out endpoint has received more than 3 back-to-back setup packets.
5	Reserved	Must be kept at reset value.
4	EPRXFOVR	Endpoint Rx FIFO overrun This flag is triggered if the OUT endpoint's Rx FIFO has no enough space for a packet data when an OUT token is incoming. USBFS will drop the incoming OUT data packet and sends a NAK handshake in this case.
3	STPF	SETUP phase finished (Only for control OUT endpoint) This flag is triggered when a setup phase finished, i.e. USBFS receives an IN or OUT token after a setup token.
2	Reserved	Must be kept at reset value.
1	EPDIS	Endpoint disabled This flag is triggered when an endpoint is disabled by the software's request.
0	TF	Transfer finished This flag is triggered when all the OUT transactions assigned to this endpoint have been finished.

Device IN endpoint 0 transfer length register (USBFS_DIEP0LEN)

Address offset: 0x0910

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)



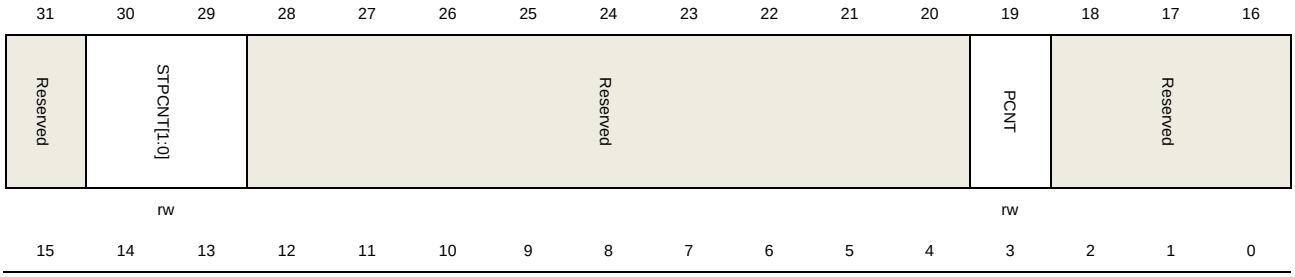
Bits	Fields	Descriptions
31:21	Reserved	Must be kept at reset value.
20:19	PCNT[1:0]	Packet count The number of data packets desired to be transmitted in a transfer. Program this field before the endpoint is enabled. After the transfer starts, this field is decreased automatically after each successful data packet transmission.
18:7	Reserved	Must be kept at reset value.
6:0	TLEN[6:0]	Transfer length The total data bytes number of a transfer. This field is the total data bytes of all the data packets desired to be transmitted in an IN transfer. Program this field before the endpoint is enabled. When software successfully writes a packet into the endpoint's Tx FIFO, this field is decreased by the byte size of the packet.

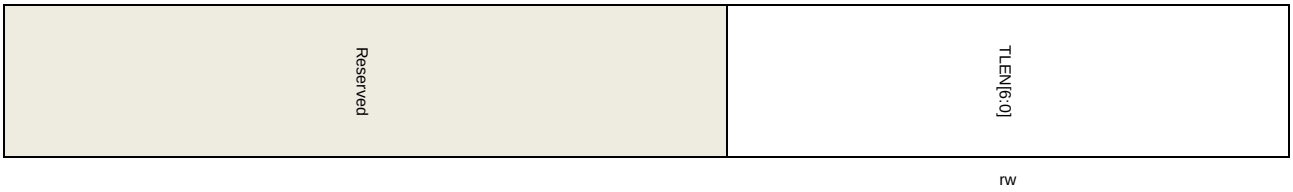
Device OUT endpoint 0 transfer length register (USBFS_DOEP0LEN)

Address offset: 0x0B10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)



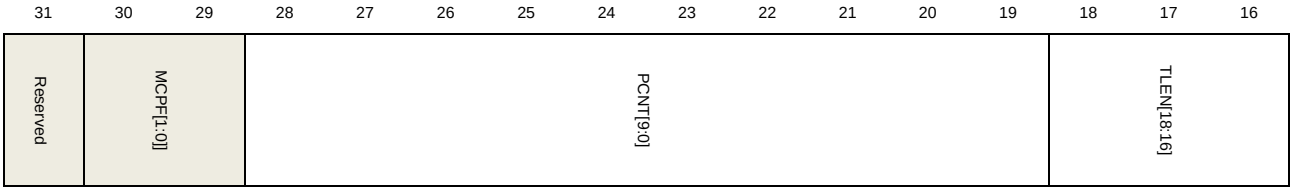


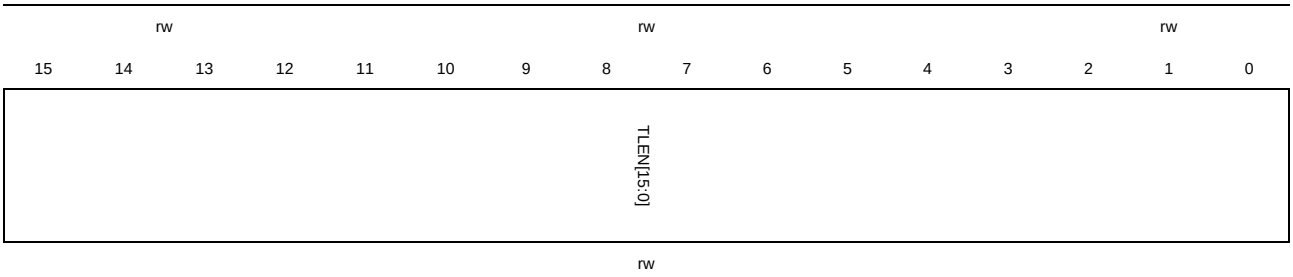
Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:29	STPCNT[1:0]	SETUP packet count This field defines the maximum number of back-to-back SETUP packets this endpoint can accept. Program this field before setup transfers. Each time a back-to-back setup packet is received, USBFS decreases this field by one. When this field reaches zero, the BTBSTP flag in USBFS_DOEP0INTF register will be triggered. 00: 0 packet 01: 1 packet 10: 2 packets 11: 3 packets
28:20	Reserved	Must be kept at reset value.
19	PCNT	Packet count The number of data packets desired to receive in a transfer. Program this field before the endpoint is enabled. After the transfer starts, this field is decreased automatically after each successful data packet reception on bus.
18:7	Reserved	Must be kept at reset value.
6:0	TLEN[6:0]	Transfer length The total data bytes number of a transfer. This field is the total data bytes of all the data packets desired to receive in an OUT transfer. Program this field before the endpoint is enabled. Each time software reads out a packet from the Rx FIFO, this field is decreased by the byte size of the packet.

Device IN endpoint x transfer length register (USBFS_DIEPxLEN) (x = 1…3, where x = endpoint_number)

Address offset: 0x910 + (endpoint_number × 0x20)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)



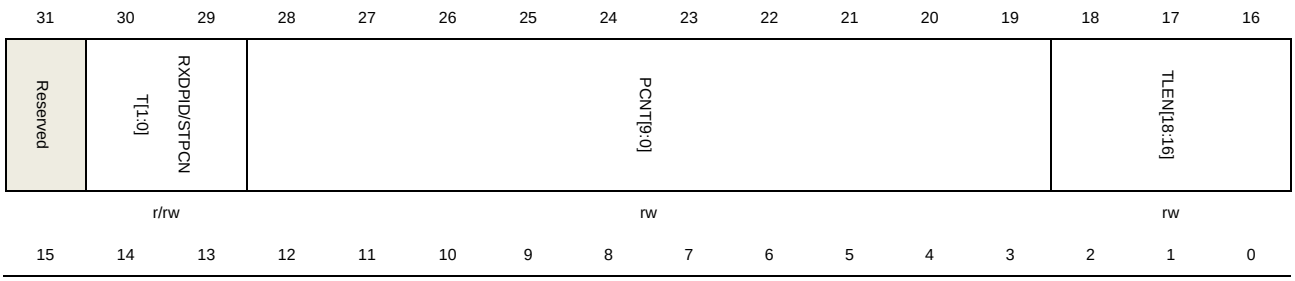


Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:29	MCPF[1:0]	Multi packet count per frame This field indicates the packet count that must be transmitted per frame for periodic IN endpoints on the USB. It is used to calculate the data PID for isochronous IN endpoints by the core. 01: 1 packet 10: 2 packets 11: 3 packets
28:19	PCNT[9:0]	Packet count The number of data packets desired to be transmitted in a transfer. Program this field before the endpoint is enabled. After the transfer starts, this field is decreased automatically after each successful data packet transmission.
18:0	TLEN[18:0]	Transfer length The total data bytes number of a transfer. This field is the total data bytes of all the data packets desired to be transmitted in an IN transfer. Program this field before the endpoint is enabled. When software successfully writes a packet into the endpoint's Tx FIFO, this field is decreased by the byte size of the packet.

Device OUT endpoint x transfer length register (USBFS_DOEPxLEN) (x = 1…3, where x = endpoint_number)

Address offset: 0x0B10 + (endpoint_number × 0x20)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)



TLEN[15:0]
rw

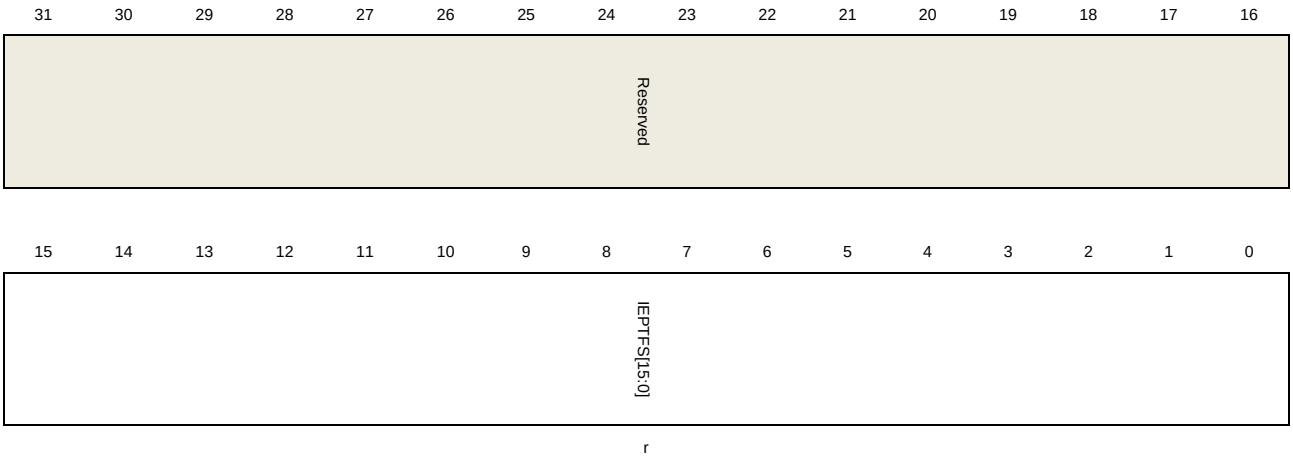
Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:29	RXDPID[1:0]	Received DATA PID (for isochronous OUT endpoints) This field saves the PID of the latest received data packet on this endpoint. 00: DATA0 10: DATA1 Others: Reserved
	STPCNT[1:0]	SETUP packet count (for control OUT Endpoints.) This field defines the maximum number of back-to-back SETUP packets this endpoint can accept. Program this field before SETUP transfers. Each time a back-to-back SETUP packet is received, USBFS decreases this field by one. When this field reaches zero, the BTBSTP flag in USBFS_DOEPxINTF register will be triggered. 00: 0 packet 01: 1 packet 10: 2 packets 11: 3 packets
28:19	PCNT[9:0]	Packet count The number of data packets desired to receive in a transfer. Program this field before the endpoint is enabled. After the transfer starts, this field is decreased automatically after each successful data packet reception on bus.
18:0	TLEN[18:0]	Transfer length The total data bytes number of a transfer. This field is the total data bytes of all the data packets desired to receive in an OUT transfer. Program this field before the endpoint is enabled. Each time after software reads out a packet from the Rx FIFO, this field is decreased by the byte size of the packet.

Device IN endpoint x Tx FIFO status register (USBFS_DIEPxTFSTAT) (x = 0…3, where x = endpoint_number)

Address offset: 0x0918 + (endpoint_number × 0x20)
Reset value: 0x0000 0200

This register contains the information of each endpoint's Tx FIFO.

This register has to be accessed by word (32-bit)

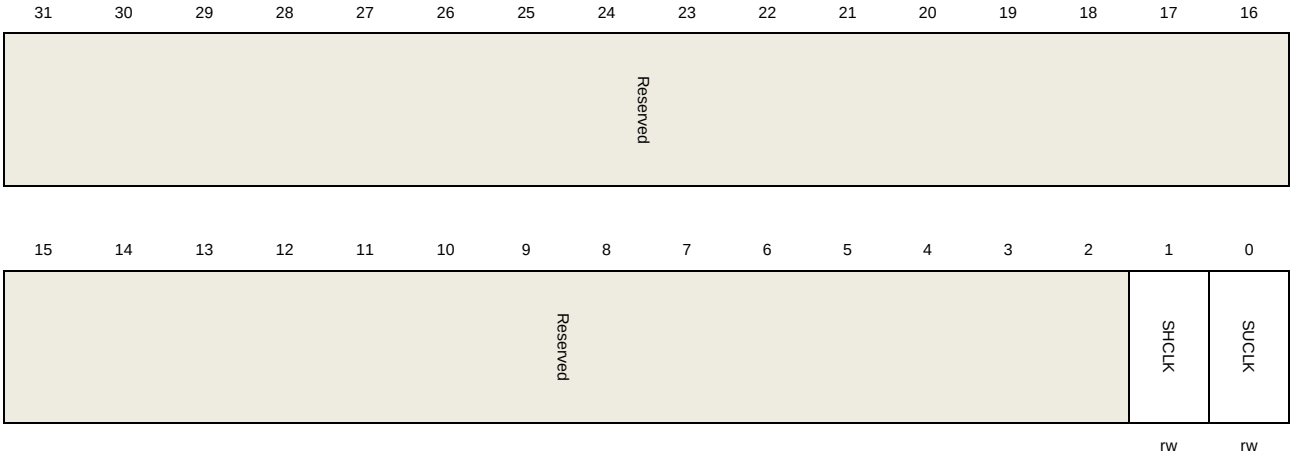


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	IEPTFS[15:0]	IN endpoint's Tx FIFO remaining space IN endpoint's Tx FIFO remaining space is in terms of 32-bit words: 0: Tx FIFO is full. 1: 1 word available … n: n words available

22.7.4. Power and clock control register (USBFS_PWRCLKCTL)

Address offset: 0x0E00
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)



Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.

1	SHCLK	Stop HCLK Stop the HCLK to save power. 0: HCLK is not stopped 1: HCLK is stopped
0	SUCLK	Stop the USB clock Stop the USB clock to save power. 0: USB clock is not stopped 1: USB clock is stopped

23. Appendix

23.1. List of abbreviations used in register

Table 23-1. List of abbreviations used in register

abbreviations for registers	Descriptions
read/write (rw)	Software can read and write to this bit.
read-only (r)	Software can only read this bit.
write-only (w)	Software can only write to this bit. Reading this bit returns the reset value.
read/clear write 1 (rc_w1)	Software can read as well as clear this bit by writing 1. Writing 0 has no effect on the bit value.
read/clear write 0 (rc_w0)	Software can read as well as clear this bit by writing 0. Writing 1 has no effect on the bit value.
toggle (t)	The software can toggle this bit by writing 1. Writing 0 has no effect.
read/set (rs)	Software can read as well as set this bit to 1. Writing '0' has no effect on the bit value.
read/clear by read (rc_r)	Software can read this bit. Reading this bit automatically clears it to '0'. Writing '0' has no effect on the bit value.

23.2. List of terms

Table 23-2. List of terms

Glossary	Descriptions
Word	Data of 32-bit length.
Half-word	Data of 16-bit length.
Byte	Data of 8-bit length.
IAP (in-application programming)	Writing 0 has no effect IAP is the ability to re-program the Flash memory of a microcontroller while the user program is running.
ICP (in-circuit programming)	ICP is the ability to program the Flash memory of a microcontroller using the JTAG protocol, the SWD protocol or the boot loader while the device is mounted on the user application board.
Option bytes	Product configuration bits stored in the Flash memory.
AHB	Advanced high-performance bus.
APB	Advanced peripheral bus.
RAZ	Read-as-zero.
WI	Writes ignored.
RAZ/WI	Read-as-zero, writes ignored.

23.3. Available peripherals

For availability of peripherals and their number across all MCU series types, refer to the corresponding device data datasheet.

24. Revision history

Table 24-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Jul.11, 2022
1.1	1. Modify TIMER2_CH0/TIMER2_CH1 remap pin of ***Table 8-5. TIMERx alternate function remapping*** in GPIO chapter. 2. Delete the description of CI1FE0 and CH1MS in ***General level2 timer (TIMERx, x=9, 10, 12, 13)*** chapter. 3. Update the description of VDD and VDDA in ***VDDA domain***: When the VDD and VDDA are provided by different power supplies, the difference between VDD and VDDA during power-up and running time should not exceed 0.3V. 4. Update ***Digital-to-analog converter (DAC)*** chapter. 5. Update ***Controller area network (CAN)***.chapter.	Jan.13, 2024
1.2	1. Update the software flow in the ***Operation modes*** section. 2. Modify the LVD monitoring voltage to VDDA in the ***VDDA domain*** section. 3. Update ***Table 12-2. ADC input pins definition***. 4. Update ***Figure 3-1. Power supply overview***, and add relevant notes and explanations. 5. Update the descriptions of VDD, VDDA, POR, and PDR in the ***V_{DD} / V_{DDA} power domain*** section.	Feb.20, 2025
1.3	1. Modify the description of Tx pin configuration in half-duplex mode in ***Half-duplex communication mode*** section. 2. Modify the reset values of register ***Port control register 0 (GPIOx_CTL0, x=A..E)*** and register ***Port control register 1 (GPIOx_CTL1, x=A..E)***. 3. Modify the maximum timeout value of SMBUS to 25ms in ***SMBus support*** section. 4. Add I2S serial clock configuration limitations in ***I2S clock*** section.	Aug.5, 2025
1.4	1. Add notes on the use of the SPI module CRC function in Section ***CRC function*** section. 2. Modify the 1.2V power domain to the Vcore domain in ***Power management unit (PMU)***. 3. Removed the IO23_DRV bit field and its related description from the ***Quad-SPI mode control register (SPI_QCTL)*** of SPI0. 4. Section ***USART receiver***: Updated the noise error trigger conditions. The Start bit no longer triggers a noise error. 5. Section ***Smartcard (ISO7816-3) mode***: In Smart Card mode, for Byte mode (T=0), the guard time inserted	Jan.19, 2026

	before retransmitting a data frame has been changed from 2.5 bits to 2 bits. 6. Modify the BSD JTAG TAP declaration in ***JTAG daisy chained structure*** section to MCU JTAG TAP. 7. Add watchdog continuous reload minimum interval time to section ***Function overview***. 8. Add FMC module ***Characteristics*** section description, flash supports 64-bit wide data read. 9. Add notes for USART_RT bit field description in ***Receiver timeout register (USART_RT)***. 10. Add ***IrDA SIR ENDEC mode*** section description: When SIR module works in low power mode, configuring PSC[7:0] requires setting baud rate register value to 16*PSC[7:0]. 11. Modify CAN protocol synchronization edge description in ***Communication parameters*** section and add ISO11898 standard compliance note.	

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the “Product”), is owned by the Company according to the laws of the People’s Republic of China and other applicable laws. The Company reserves all rights under such laws and no Intellectual Property Rights are transferred (either wholly or partially) or licensed by the Company (either expressly or impliedly) herein. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

To the maximum extent permitted by applicable law, the Company makes no representations or warranties of any kind, express or implied, with regard to the merchantability and the fitness for a particular purpose of the Product, nor does the Company assume any liability arising out of the application or use of any Product. Any information provided in this document is provided only for reference purposes. It is the sole responsibility of the user of this document to determine whether the Product is suitable and fit for its applications and products planned, and properly design, program, and test the functionality and safety of its applications and products planned using the Product. The Product is designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only, and the Product is not designed or intended for use in (i) safety critical applications such as weapons systems, nuclear facilities, atomic energy controller, combustion controller, aeronautic or aerospace applications, traffic signal instruments, pollution control or hazardous substance management; (ii) life-support systems, other medical equipment or systems (including life support equipment and surgical implants); (iii) automotive applications or environments, including but not limited to applications for active and passive safety of automobiles (regardless of front market or aftermarket), for example, EPS, braking, ADAS (camera/fusion), EMS, TCU, BMS, BSG, TPMS, Airbag, Suspension, DMS, ICMS, Domain, ESC, DCDC, e-clutch, advanced-lighting, etc.. Automobile herein means a vehicle propelled by a self-contained motor, engine or the like, such as, without limitation, cars, trucks, motorcycles, electric cars, and other transportation devices; and/or (iv) other uses where the failure of the device or the Product can reasonably be expected to result in personal injury, death, or severe property or environmental damage (collectively "Unintended Uses"). Customers shall take any and all actions to ensure the Product meets the applicable laws and regulations. The Company is not liable for, in whole or in part, and customers shall hereby release the Company as well as its suppliers and/or distributors from, any claim, damage, or other liability arising from or related to all Unintended Uses of the Product. Customers shall indemnify and hold the Company, and its officers, employees, subsidiaries, affiliates as well as its suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Product.

Information in this document is provided solely in connection with the Product. The Company reserves the right to make changes, corrections, modifications or improvements to this document and the Product described herein at any time without notice. The Company shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2026 GigaDevice Semiconductor Inc. – All rights reserved