GigaDevice Semiconductor Inc.

GD32E25x
Arm® Cortex®-M23 32-bit MCU

For GD32E251xx, GD32E252xx

User Manual

Revision 1.1

(Aug. 2026)

Table of Contents

Table of Contents 2
List of Figures 14
List of Tables 21
1. System and memory architecture 23
1.1. Arm® Cortex®-M23 processor 23
1.2. System architecture 24
1.3. Memory map 25
1.3.1. On-chip SRAM memory 28
1.3.2. On-chip Flash memory 30
1.4. Boot configuration 30
1.5. System configuration registers (SYSCFG) 32
1.5.1. System configuration register 0 (SYSCFG_CFG0) 32
1.5.2. EXTI sources selection register 0 (SYSCFG_EXTISS0) 34
1.5.3. EXTI sources selection register 1 (SYSCFG_EXTISS1) 35
1.5.4. EXTI sources selection register 2 (SYSCFG_EXTISS2) 36
1.5.5. EXTI sources selection register 3 (SYSCFG_EXTISS3) 36
1.5.6. System configuration register 1 (SYSCFG_CFG1) 37
1.5.7. SYSCFG status register (SYSCFG_STAT) 39
1.5.8. SRAM_ECC register (SYSCFG_SRAM_ECC) 39
1.5.9. IRQ latency register (SYSCFG_CPU_IRQ_LAT) 40
1.6. Device electronic signature 41
1.6.1. Memory density information 41
1.6.2. Unique device ID (96 bits) 41
2. Interrupt / event controller (EXTI) 43
2.1. Overview 43
2.2. Characteristics 43
2.3. Interrupts function overview 43
2.4. External interrupt and event block diagram 46
2.5. External interrupt and event function overview 46
2.6. Register definition 48
2.6.1. Interrupt enable register (EXTI_INTEN) 48
2.6.2. Event enable register (EXTI_EVEN) 48
2.6.3. Rising edge trigger enable register (EXTI_RTEN) 49
2.6.4. Falling edge trigger enable register (EXTI_FTEN) 50

2.6.5. Software interrupt event register (EXTI_SWIEV) 50
2.6.6. Pending register (EXTI_PD) 51
2.6.7. Source default control register (EXTI_CTL) 52
3. Direct memory access controller (DMA) 53
3.1. Overview 53
3.2. Characteristics 53
3.3. Block diagram 54
3.4. Function overview 54
3.4.1. DMA operation 54
3.4.2. Peripheral handshake 56
3.4.3. Arbitration 56
3.4.4. Address generation 57
3.4.5. Circular mode 57
3.4.6. Memory to memory mode 57
3.4.7. Channel configuration 57
3.4.8. Interrupt 58
3.4.9. DMA request mapping 58
3.5. Register definition 61
3.5.1. Interrupt flag register (DMA_INTF) 61
3.5.2. Interrupt flag clear register (DMA_INTC) 61
3.5.3. Channel x control register (DMA_CHxCTL) 62
3.5.4. Channel x counter register (DMA_CHxCNT) 64
3.5.5. Channel x peripheral base address register (DMA_CHxPADDR) 65
3.5.6. Channel x memory base address register (DMA_CHxMADDR) 65
4. Flash memory controller (FMC) 67
4.1. Overview 67
4.2. Characteristics 67
4.3. Function overview 67
4.3.1. Flash memory architecture 67
4.3.2. Error checking and correcting (ECC) 68
4.3.3. Read operations 69
4.3.4. Unlock the FMC_CTL register 70
4.3.5. Page erase 70
4.3.6. Mass erase 71
4.3.7. Main flash programming 73
4.3.8. OTP programming 74
4.3.9. Option byte erase 74
4.3.10. Option byte programming 75
4.3.11. Option byte description 75
4.3.12. Page erase/Program protection 77

4.3.13. Security protection 78
4.4. Register definition 79
4.4.1. Wait state register (FMC_WS) 79
4.4.2. Unlock key register (FMC_KEY) 79
4.4.3. Option byte unlock key register (FMC_OBKEY) 80
4.4.4. Status register (FMC_STAT) 80
4.4.5. Control register (FMC_CTL) 81
4.4.6. Address register (FMC_ADDR) 83
4.4.7. Option byte status register (FMC_OBSTAT) 83
4.4.8. Write protection register (FMC_WP) 84
4.4.9. ECC control and status register (FMC_ECCCS) 85
4.4.10. ECC address register (FMC_ECCADDR) 86
4.4.11. Product ID register (FMC_PID) 86
5. Power management unit (PMU) 87
5.1. Overview 87
5.2. Characteristics 87
5.3. Function overview 87
5.3.1. V_{DD} / V_{DDA} power domain 88
5.3.2. VCORE power domain 90
5.3.3. Power saving modes 91
5.4. PMU registers 94
5.4.1. Control register (PMU_CTL) 94
5.4.2. Control and status register (PMU_CS) 95
6. Backup Domain (BKP) 97
6.1. Overview 97
6.2. Characteristics 97
6.3. Function overview 97
6.3.1. Block diagram 97
6.3.2. Tamper detection 97
6.4. BKP interrupts 98
6.5. Register definition 100
6.5.1. Status register (BKP_STAT) 100
6.5.2. Tamper register (BKP_TAMP) 100
6.5.3. Backup registers (BKP_DATAx) (x=0…4) 102
7. Reset and clock unit (RCU) 103
7.1. Reset control unit (RCTL) 103
7.1.1. Overview 103
7.1.2. Function overview 103

7.2. Clock control unit (CCTL) 104
7.2.1. Overview 104
7.2.2. Characteristics 106
7.2.3. Function overview 106
7.3. Register definition 110
7.3.1. Control register 0 (RCU_CTL0) 110
7.3.2. Configuration register 0 (RCU_CFG0) 112
7.3.3. Interrupt register (RCU_INT) 115
7.3.4. APB2 reset register (RCU_APB2RST) 118
7.3.5. APB1 reset register (RCU_APB1RST) 119
7.3.6. AHB enable register (RCU_AHBEN) 121
7.3.7. APB2 enable register (RCU_APB2EN) 123
7.3.8. APB1 enable register (RCU_APB1EN) 124
7.3.9. Backup domain control register (RCU_BDCTL) 126
7.3.10. Reset source /clock register (RCU_RSTSCK) 127
7.3.11. AHB reset register (RCU_AHBRST) 129
7.3.12. Configuration register 1 (RCU_CFG1) 130
7.3.13. Configuration register 2 (RCU_CFG2) 131
7.3.14. Control register 1 (RCU_CTL1) 132
7.3.15. Voltage key register (RCU_VKEY) 133
7.3.16. Deep-sleep mode voltage register (RCU_DSV) 133
8. General-purpose I / O (GPIO) 135
8.1. Overview 135
8.2. Characteristics 135
8.3. Function overview 135
8.3.1. GPIO pin configuration 137
8.3.2. External interrupt / event lines 138
8.3.3. Alternate functions (AF) 138
8.3.4. Additional functions 138
8.3.5. Input configuration 138
8.3.6. Output configuration 138
8.3.7. Analog configuration 139
8.3.8. Alternate function (AF) configuration 140
8.3.9. GPIO locking function 140
8.3.10. GPIO single cycle toggle function 140
8.3.11. GPIO reset hold 141
8.3.12. Priority Crossbar decoder (only for GD32E251xx) 141
8.3.13. INT0 and INT1 (only for GD32E251xx) 142
8.3.14. Pin match (only for GD32E251xx) 142
8.3.15. Pin not match wakes up from deepsleep mode (only for GD32E251xx) 142
8.4. Register definition 143

8.4.1. Port control register (GPIOx_CTL, x=A, B, F) 143
8.4.2. Port output mode register (GPIOx_OMODE, x=A, B, F) 145
8.4.3. Port output speed register (GPIOx_OSPD, x=A, B, F) 147
8.4.4. Port pull-up/down register (GPIOx_PUD, x=A, B, F) 148
8.4.5. Port input status register (GPIOx_ISTAT, x=A, B, F) 150
8.4.6. Port output control register (GPIOx_OCTL, x=A, B, F) 151
8.4.7. Port bit operate register (GPIOx_BOP, x=A, B, F) 151
8.4.8. Port configuration lock register (GPIOx_LOCK, x=A, B, F) 152
8.4.9. Alternate function selected register 0 (GPIOx_AFSEL0, x=A / B) 152
8.4.10. Alternate function selected register 1 (GPIOx_AFSEL1, x=A / B) 154
8.4.11. Bit clear register (GPIOx_BC, x=A, B, F) 155
8.4.12. Port bit toggle register (GPIOx_TG, x=A, B, F) 155
8.4.13. Crossbar enable register (GPIO_CBREN) 156
8.4.14. Digitial function enable register (GPIO_DFEN) 158
8.4.15. Crossbar skip register (GPIO_SKIP) 159
8.4.16. INT0 and INT1 control register (GPIO_INTCTL) 162
8.4.17. Pin mask register (GPIO_MASK) 163
8.4.18. Pin match register (GPIO_MATCH) 166
8.4.19. Pin match flag register (GPIO_MATCHF) 169

9. Timer (TIMERx) 173

9.1. Advanced timer (TIMERx, x=0) 173
9.1.1. Overview 173
9.1.2. Characteristics 174
9.1.3. Block diagram 175
9.1.4. Function overview 176
9.1.5. TIMERx register definition (x=0) 205

9.2. General level0 timer (TIMERx, x=1, 2) 236
9.2.1. Overview 236
9.2.2. Characteristics 236
9.2.3. Block diagram 236
9.2.4. Function overview 238
9.2.5. TIMERx register definition (x=1, 2) 251

9.3. General level2 timer (TIMERx, x=13) 277
9.3.1. Overview 277
9.3.2. Characteristics 277
9.3.3. Block diagram 277
9.3.4. Function overview 278
9.3.5. TIMERx register definition (x=13) 286

9.4. General level3 timer (TIMERx, x=14) 296
9.4.1. Overview 296
9.4.2. Characteristics 296

9.4.3. Block diagram 296
9.4.4. Function overview 297
9.4.5. TIMERx register definition (x=14) 315
9.5. General level4 timer (TIMERx, x=15, 16) 335
9.5.1. Overview 335
9.5.2. Characteristics 335
9.5.3. Block diagram 335
9.5.4. Function overview 336
9.5.5. TIMERx register definition (x=15, 16) 352
9.6. Basic timer (TIMERx, x=5, 6) 368
9.6.1. Overview 368
9.6.2. Characteristics 368
9.6.3. Block diagram 368
9.6.4. Function overview 368
9.6.5. TIMERx registers(x=5, 6) 373
10. Watchdog timer (WDGT) 378
10.1. Free watchdog timer (FWDGT) 378
10.1.1. Overview 378
10.1.2. Characteristics 378
10.1.3. Function overview 378
10.1.4. Register definition 381
10.2. Window watchdog timer (WWDGT) 385
10.2.1. Overview 385
10.2.2. Characteristics 385
10.2.3. Function overview 385
10.2.4. Register definition 388
11. Debug (DBG) 390
11.1. Overview 390
11.2. SW function overview 390
11.2.1. Pin assignment 390
11.3. Debug hold function overview 390
11.3.1. Debug support for power saving mode 390
11.3.2. Debug support for TIMER, I2C, WWDGT and FWDGT 391
11.4. Register definition 392
11.4.1. ID code register (DBG_ID) 392
11.4.2. Control register 0 (DBG_CTL0) 392
11.4.3. Control register 1 (DBG_CTL1) 394
12. Cyclic redundancy checks management unit (CRC) 396
12.1. Overview 396

12.2. Characteristics 396
12.3. Function overview 397
12.4. Register definition 398
12.4.1. Data register (CRC_DATA) 398
12.4.2. Free data register (CRC_FDATA) 398
12.4.3. Control register (CRC_CTL) 399
12.4.4. Initialization data register (CRC_IDATA) 399
12.4.5. Polynomial register (CRC_POLY) 400
13. Analog to digital converter (ADC) 401
13.1. Overview 401
13.2. Characteristics 401
13.3. Pins and internal signals 402
13.4. Function overview 403
13.4.1. Foreground calibration function 403
13.4.2. Dual clock domain architecture 404
13.4.3. ADC enable 404
13.4.4. Routine and inserted sequence 404
13.4.5. Operation modes 404
13.4.1. Inserted channel operation 409
13.4.2. Convert data latch 410
13.4.3. Conversion result threshold monitor function 411
13.4.4. Data storage mode 412
13.4.5. Sample time configuration 413
13.4.6. External trigger configuration 413
13.4.7. DMA request 415
13.4.8. ADC internal channels 415
13.4.9. Programmable resolution (DRES) 416
13.4.10. On-chip hardware oversampling 416
13.4.11. ADC convertsion signal 418
13.4.12. ADC interrupts 419
13.5. Register definition 420
13.5.1. Status register (ADC_STAT) 420
13.5.2. Control register 0 (ADC_CTL0) 421
13.5.3. Control register 1 (ADC_CTL1) 423
13.5.4. Sample time register 0 (ADC_SAMPT0) 425
13.5.5. Sample time register 1 (ADC_SAMPT1) 426
13.5.6. Inserted channel data offset register x (ADC_IOFFx) (x=0..3) 427
13.5.7. Watchdog high threshold register (ADC_WDHT) 428
13.5.8. Watchdog low threshold register (ADC_WDLT) 428
13.5.9. Routine sequence register 0 (ADC_RSQ0) 428

13.5.10. Routine sequence register 1 (ADC_RSQ1) 429
13.5.11. Routine sequence register 2 (ADC_RSQ2) 430
13.5.12. Inserted sequence register (ADC_ISQ) 430
13.5.13. Latch data register x (ADC_LDATAx) (x= 0..3) 431
13.5.14. Routine data register (ADC_RDATA) 431
13.5.15. Oversampling control register (ADC_OVSAMPCTL) 432
13.5.16. Sample time register 2 (ADC_SAMPT2) 433
13.5.17. Control register 2 (ADC_CTL2) 434
13.5.18. Routine sequence register 3 (ADC_RSQ3) 435
13.5.19. Inserted data register (ADC_IDATA) 435
13.5.20. Latch data control register(ADC_LDCTL) 436

14. Digital-to-analog converter (DAC) 438

14.1. Overview 438

14.2. Characteristics 438

14.3. Function overview 440

14.3.1. DAC enable 440
14.3.2. DAC output buffer 440
14.3.3. DAC data configuration 440
14.3.4. DAC trigger 440
14.3.5. DAC conversion 441
14.3.6. DAC noise wave 441
14.3.7. DAC output voltage 442
14.3.8. DMA request 443
14.3.9. DAC concurrent conversion 443
14.3.10. DAC reset persist mode 443

14.4. Register definition 444

14.4.1. DACx control register 0 (DAC_CTL0) 444
14.4.2. DACx software trigger register (DAC_SWT) 447
14.4.3. DACx_OUT0 12-bit right-aligned data holding register (DAC_OUT0_R12DH) 448
14.4.4. DACx_OUT0 12-bit left-aligned data holding register (DAC_OUT0_L12DH) 448
14.4.5. DACx_OUT0 8-bit right-aligned data holding register (DAC_OUT0_R8DH) 448
14.4.6. DACx_OUT1 12-bit right-aligned data holding register (DAC_OUT1_R12DH) 449
14.4.7. DACx_OUT1 12-bit left-aligned data holding register (DAC_OUT1_L12DH) 449
14.4.8. DACx_OUT1 8-bit right-aligned data holding register (DAC_OUT1_R8DH) 450
14.4.9. DACx concurrent mode 12-bit right-aligned data holding register (DACC_R12DH) 450
14.4.10. DACx concurrent mode 12-bit left-aligned data holding register (DACC_L12DH) 451
14.4.11. DACx concurrent mode 8-bit right-aligned data holding register (DACC_R8DH) 451
14.4.12. DACx_OUT0 data output register (DAC_OUT0_DO) 452
14.4.13. DACx_OUT1 data output register (DAC_OUT1_DO) 452
14.4.14. DACx status register 0 (DAC_STAT0) 453
14.4.15. DACx output control register (DAC_OCTL) 453

15. Comparator (CMP) 455
15.1. Overview 455
15.2. Characteristics 455
15.3. Function overview 455
15.3.1. CMP clock and reset 456
15.3.2. CMP I / O configuration 456
15.3.3. CMP operating mode 458
15.3.4. CMP hysteresis 458
15.3.5. CMP register write protection 459
15.3.6. CMP interrupt 459
15.3.7. CMP pin reset mode 459
15.4. Register definition 460
15.4.1. CMP Control/status register (CMPx_CS) 460
16. Configurable Logic Array (CLA) 463
16.1. Overview 463
16.2. Characteristics 463
16.3. Block diagram 463
16.4. Function overview 464
16.4.1. CLA input Signal Selector 464
16.4.2. LCU control 465
16.4.3. CLA Output 466
16.4.4. Interrupt 466
16.5. Register definition 468
16.5.1. Global control register (CLA_GCTL) 468
16.5.2. Interrupt enable register (CLA_INTE) 468
16.5.3. Interrupt flag register (CLA_INTF) 470
16.5.4. Status register (CLA_STAT) 471
16.5.5. Signal selection register (CLAx_SIGS)(x=0..3) 472
16.5.6. LCU control register (CLAx_LCUCTL) (x=0..3) 472
16.5.7. Control register (CLAx_CTL) (x=0..3) 473
17. VREF 475
17.1. Overview 475
17.2. Characteristics 475
17.3. Function overview 475
18. Infrared ray port (IFRP) 476
18.1. Overview 476
18.2. Characteristics 476

18.3. Function overview 476

19. Universal synchronous/asynchronous receiver / transmitter (USART) 478

19.1. Overview 478

19.2. Characteristics 478

19.3. Function overview 480

19.3.1. USART frame format 480
19.3.2. Baud rate generation 481
19.3.3. USART transmitter 482
19.3.4. USART receiver 483
19.3.5. Use DMA for data buffer access 485
19.3.6. Hardware flow control 486
19.3.7. Multi-processor communication 487
19.3.8. LIN mode 488
19.3.9. Synchronous mode 489
19.3.10. IrDA SIR ENDEC mode 490
19.3.11. Half-duplex communication mode 491
19.3.12. Smartcard (ISO7816-3) mode 491
19.3.13. Auto baudrate detection 493
19.3.14. ModBus communication 494
19.3.15. Receive FIFO 494
19.3.16. Wakeup from deep-sleep mode 495
19.3.17. USART interrupts 495

19.4. Register definition 498

19.4.1. Control register 0 (USART_CTL0) 498
19.4.2. Control register 1 (USART_CTL1) 500
19.4.3. Control register 2 (USART_CTL2) 503
19.4.4. Baud rate generator register (USART_BAUD) 506
19.4.5. Prescaler and guard time configuration register (USART_GP) 506
19.4.6. Receiver timeout register (USART_RT) 507
19.4.7. Command register (USART_CMD) 508
19.4.8. Status register (USART_STAT) 509
19.4.9. Interrupt status clear register (USART_INTC) 513
19.4.10. Receive data register (USART_RDATA) 515
19.4.11. Transmit data register (USART_TDATA) 515
19.4.12. USART coherence control register (USART_CHC) 516
19.4.13. USART receive FIFO control and status register (USART_RFCS) 516

20. Serial peripheral interface/Inter-IC sound (SPI / I2S) 518

20.1. Overview 518

20.2. Characteristics 518

20.2.1. SPI characteristics 518

20.2.2. I2S characteristics 518
20.3. SPI function overview 519
20.3.1. SPI block diagram 519
20.3.2. SPI signal description 519
20.3.3. SPI clock timing and data format 520
20.3.4. Separate transmission and reception FIFO 522
20.3.5. NSS function 523
20.3.6. SPI operation modes 524
20.3.7. DMA function 533
20.3.8. CRC function 534
20.3.9. SPI interrupts 535
20.4. I2S function overview 537
20.4.1. I2S block diagram 537
20.4.2. I2S signal description 537
20.4.3. I2S audio standards 537
20.4.4. I2S clock 546
20.4.5. Operation 547
20.4.6. DMA function 551
20.4.7. I2S interrupts 551
20.5. Register definition 553
20.5.1. Control register 0 (SPI_CTL0) 553
20.5.2. Control register 1 (SPI_CTL1) 555
20.5.3. Status register (SPI_STAT) 557
20.5.4. Data register (SPI_DATA) 559
20.5.5. CRC polynomial register (SPI_CRCPOLY) 559
20.5.6. RX CRC register (SPI_RCRC) 560
20.5.7. TX CRC register (SPI_TCRC) 560
20.5.8. I2S control register (SPI_I2SCTL) 561
20.5.9. I2S clock prescaler register (SPI_I2SPSC) 563
20.5.10. Quad-SPI mode control register (SPI_QCTL) of SPI1 564
21. Inter-integrated circuit interface (I2C) 565
21.1. Inter-integrated circuit interface (I2Cx, x=0, 1) 565
21.1.1. Overview 565
21.1.2. Characteristics 565
21.1.3. Function overview 565
21.1.4. Register definition 582
21.2. Inter-integrated circuit interface (I2C2) 596
21.2.1. Overview 596
21.2.2. Characteristics 596
21.2.3. Function overview 596
21.2.4. Register definition 622

22. Appendix 637
22.1. List of abbreviations used in register 637
22.2. List of terms 638
22.3. Available peripherals 638
23. Revision history 639

List of Figures

Figure 1-1. The structure of the Cortex®-M23 processor 24
Figure 1-2. Series system architecture of GD32E25x series 25
Figure 1-3. ECC decoder 29
Figure 2-1. Block diagram of EXTI 46
Figure 3-1. Block diagram of DMA 54
Figure 3-2. Handshake mechanism 56
Figure 3-3. DMA interrupt logic 58
Figure 3-4. DMA request mapping 59
Figure 4-1. Process of page erase operation 71
Figure 4-2. Process of the mass erase operation 72
Figure 4-3. Process of the double word programming operation 74
Figure 5-1. Power supply overview 88
Figure 5-2. Waveform of the POR/PDR 89
Figure 5-3. Waveform of the LVD threshold 90
Figure 6-1. Block diagram of BKP 97
Figure 7-1. The system reset circuit 104
Figure 7-2. Clock tree 105
Figure 7-3. HXTAL clock source 106
Figure 7-4. HXTAL clock source in bypass mode 107
Figure 8-1. Basic structure of of a general-pupose I / O 137
Figure 8-2. Basic structure of Input configuration 138
Figure 8-3. Basic structure of Output configuration 139
Figure 8-4. Basic structure of Analog configuration 139
Figure 8-5. Basic structure of Alternate function configuration 140
Figure 8-6. Digital functional map 141
Figure 8-7. Crossbar decoder functional map 141
Figure 9-1. Advanced timer block diagram 175
Figure 9-2. Timing chart of internal clock divided by 1 176
Figure 9-3. Timing chart of PSC value change from 0 to 2 177
Figure 9-4. Timing chart of up counting mode, PSC=0/2 178
Figure 9-5. Timing chart of up counting mode, change TIMERx_CAR on the go 179
Figure 9-6. Timing chart of down counting mode, PSC=0/2 180
Figure 9-7. Timing chart of down counting mode, change TIMERx_CAR on the go 180
Figure 9-8. Timing chart of center-aligned counting mode 182
Figure 9-9. Repetition counter timing chart of center-aligned counting mode 183
Figure 9-10. Repetition counter timing chart of up counting mode 183
Figure 9-11. Repetition counter timing chart of down counting mode 184
Figure 9-12. Channel input capture principle 185
Figure 9-13. Channel output compare principle (with complementary output, x=0, 1, 2). 186
Figure 9-14. Channel output compare principle (CHx_O,x=3, 4, 5) 186

Figure 9-15. Output-compare in three modes 188
Figure 9-16. Timing chart of EAPWM 189
Figure 9-17. Timing chart of CAPWM 189
Figure 9-18. Complementary output with dead time insertion 194
Figure 9-19. Output behavior of the channel in response to a break (the break high active) 194
Figure 9-20. Example of counter operation in quadrature decoder interface mode 196
Figure 9-21. Example of quadrature decoder interface mode with CI0FE0 polarity inverted 196
Figure 9-22. Hall sensor is used to BLDC motor 197
Figure 9-23. Hall sensor timing between two timers 198
Figure 9-24. Restart mode 199
Figure 9-25. Pause mode 200
Figure 9-26. Event mode 200
Figure 9-27. Single pulse mode, TIMERx_CHxCV = 4, TIMERx_CAR=99 201
Figure 9-28. TIMER0 Master/Slave mode timer example 202
Figure 9-29. Triggering TIMER0 and TIMER2 with TIMER2's CI0 input 203
Figure 9-30. General Level 0 timer block diagram 237
Figure 9-31. Timing chart of internal clock divided by 1 239
Figure 9-32. Timing chart of PSC value change from 0 to 2 240
Figure 9-33. Timing chart of up counting mode, PSC=0/2 241
Figure 9-34. Timing chart of up counting mode, change TIMERx_CAR on the go. 241
Figure 9-35. Timing chart of down counting mode, PSC=0/2 242
Figure 9-36. Timing chart of down counting mode, change TIMERx_CAR on the go 243
Figure 9-37. Timing chart of center-aligned counting mode 244
Figure 9-38. Channel input capture principle 245
Figure 9-39. Channel output compare principle (x=0, 1, 2, 3) 246
Figure 9-40. Output-compare under three modes 247
Figure 9-41. Timing chart of EAPWM 248
Figure 9-42. Timing chart of CAPWM 249
Figure 9-43. General level2 timer block diagram 278
Figure 9-44. Timing chart of internal clock divided by 1 279
Figure 9-45. Timing chart of PSC value change from 0 to 2 279
Figure 9-46. Timing chart of up counting mode, PSC=0/2 280
Figure 9-47. Timing chart of up counting mode, change TIMERx_CAR on the go 281
Figure 9-48. Channel input capture principle 282
Figure 9-49. Channel output compare principle 283
Figure 9-50. Output-compare in three modes 284
Figure 9-51. PWM mode timechart 285
Figure 9-52. General level3 timer block diagram 297
Figure 9-53. Timing chart of internal clock divided by 1 298
Figure 9-54. Timing chart of PSC value change from 0 to 2 299
Figure 9-55. Timing chart of up counting mode, PSC=0/2 300

Figure 9-56. Timing chart of up counting mode, change TIMERx_CAR on the go 300
Figure 9-57. Repetition counter timing chart of up counting mode 301
Figure 9-58. Channel input capture principle..... 302
Figure 9-59. channel output compare principle (with complementary output, x=0) 303
Figure 9-60. Channel output compare principle (CH1_O)..... 303
Figure 9-61. Output-compare in three modes 305
Figure 9-62. PWM mode timechart 306
Figure 9-63. Complementary output with dead-time insertion. 310
Figure 9-64. Output behavior in response to a break(The break high active) 311
Figure 9-65. Restart mode..... 312
Figure 9-66. Pause mode..... 312
Figure 9-67. Event mode 312
Figure 9-68. Single pulse mode TIMERx_CHxCV = 4 TIMERx_CAR=99..... 313
Figure 9-69. General level4 timer block diagram 336
Figure 9-70. Timing chart of internal clock divided by 1..... 337
Figure 9-71. Timing chart of PSC value change from 0 to 2..... 338
Figure 9-72. Timing chart of up counting mode, PSC=0/2 339
Figure 9-73. Timing chart of up counting mode, change TIMERx_CAR on the go 339
Figure 9-74. Repetition counter timing chart of up counting mode 340
Figure 9-75. Channel input capture principle..... 341
Figure 9-76. Channel output compare principle (with complementary output, x=0) 342
Figure 9-77. Output-compare under three modes..... 344
Figure 9-78. PWM mode timechart 345
Figure 9-79. Complementary output with dead-time insertion. 349
Figure 9-80. Output behavior in response to a break(The break high active) 350
Figure 9-81. Single pulse mode TIMERx_CHxCV = 0x04 TIMERx_CAR=0x60 351
Figure 9-82. Basic timer block diagram..... 368
Figure 9-83. Timing chart of internal clock divided by 1..... 369
Figure 9-84. Timing chart of PSC value change from 0 to 2..... 370
Figure 9-85. Timing chart of up counting mode, PSC=0/2 371
Figure 9-86. Timing chart of up counting mode, change TIMERx_CAR on the go 371
Figure 10-1. Free watchdog block diagram 379
Figure 10-2. Window watchdog timer block diagram 385
Figure 10-3. Window watchdog timing diagram 386
Figure 12-1. Block diagram of CRC calculation unit 396
Figure 13-1. ADC module block diagram 403
Figure 13-2. Single operation mode 405
Figure 13-3. Continuous operation mode 406
Figure 13-4. Scan operation mode, continuous disable..... 407
Figure 13-5. Scan operation mode, continuous enable..... 407
Figure 13-6. Discontinuous operation mode..... 408
Figure 13-7. Repeat operation mode (SM=1)..... 409
Figure 13-8. Auto-insertion, CTN = 1 409

Figure 13-9. Triggered insertion 410
Figure 13-10. Data latch 410
Figure 13-11. Sequence data from ADC_LDATAx 411
Figure 13-12. Data storage mode of 12-bit resolution 412
Figure 13-13. Data storage mode of 10-bit resolution 412
Figure 13-14. Data storage mode of 8-bit resolution 413
Figure 13-15. Data storage mode of 6-bit resolution 413
Figure 13-16. Routine trigger occurs during routine conversion 414
Figure 13-17. 20-bit to 16-bit result truncation 417
Figure 13-18. A numerical example with 5-bit shifting and rounding 417
Figure 13-19. ADC convertsion signal in continuous operation mode 419
Figure 14-1. DAC block diagram 439
Figure 14-2. DAC LFSR algorithm 442
Figure 14-3. DAC triangle noise wave 442
Figure 15-1. CMP block diagram 456
Figure 15-2. CMP hysteresis 459
Figure 16-1. Block diagram of CLA 463
Figure 16-2. CLA interrupt logic 467
Figure 17-1. Precision Reference Connection 475
Figure 18-1. IFRP output timechart 1 476
Figure 18-2. IFRP output timechart 2 477
Figure 18-3. IFRP output timechart 3 477
Figure 19-1. USART module block diagram 480
Figure 19-2. USART character frame (8 bits data and 1 stop bit) 481
Figure 19-3. USART transmit procedure 483
Figure 19-4. Oversampling method of a receive frame bit (OSB=0) 484
Figure 19-5. Configuration step when using DMA for USART transmission 485
Figure 19-6. Configuration step when using DMA for USART reception 486
Figure 19-7. Hardware flow control between two USARTs 486
Figure19-8. Hardware flow control 487
Figure 19-9. Break frame occurs during idle state 489
Figure 19-10. Break frame occurs during a frame 489
Figure 19-11. Example of USART in synchronous mode 489
Figure 19-12. 8-bit format USART synchronous waveform (CLEN=1) 490
Figure 19-13. IrDA SIR ENDEC module 490
Figure 19-14. IrDA data modulation 491
Figure 19-15. ISO7816-3 frame format 492
Figure 19-16. USART receive FIFO structure 495
Figure 19-17. USART interrupt mapping diagram 497
Figure 20-1. Block diagram of SPI 519
Figure 20-2. SPI0 timing diagram in normal mode 520
Figure 20-3. SPI1 timing diagram in normal mode 521
Figure 20-4. SPI timing diagram in Quad-SPI mode (CKPL=1, CKPH=1, LF=0) 521

Figure 20-5. Transmission and reception FIFO 522
Figure 20-6. A typical full-duplex connection 525
Figure 20-7. A typical simplex connection (Master: receive, Slave: transmit) 526
Figure 20-8. A typical simplex connection (Master: transmit only, Slave: receive)..... 526
Figure 20-9. A typical bidirectional connection 526
Figure 20-10. Timing diagram of TI master mode with discontinuous transfer 529
Figure 20-11. Timing diagram of TI master mode with continuous transfer 529
Figure 20-12. Timing diagram of TI slave mode..... 529
Figure 20-13. Timing diagram of NSS pulse with continuous transmit 530
Figure 20-14. Timing diagram of write operation in Quad-SPI mode 531
Figure 20-15. Timing diagram of read operation in Quad-SPI mode 532
Figure 20-16. Block diagram of I2S 537
Figure 20-17. I2S PHILIPS standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0) 538
Figure 20-18. I2S PHILIPS standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1) 538
Figure 20-19. I2S PHILIPS standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0) 538
Figure 20-20. I2S PHILIPS standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1) 539
Figure 20-21. I2S PHILIPS standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0) 539
Figure 20-22. I2S PHILIPS standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1) 539
Figure 20-23. I2S PHILIPS standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0) 539
Figure 20-24. I2S PHILIPS standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1) 540
Figure 20-25. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0).. 540
Figure 20-26. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1).. 540
Figure 20-27. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0).. 540
Figure 20-28. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1).. 540
Figure 20-29. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0).. 541
Figure 20-30. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1).. 541
Figure 20-31. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0).. 541
Figure 20-32. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1).. 541
Figure 20-33. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0)... 541
Figure 20-34. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1)... 542
Figure 20-35. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0)... 542
Figure 20-36. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1)... 542
Figure 20-37. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=0) 542
Figure 20-38. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1) 543
Figure 20-39. PCM standard short frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0) 543
Figure 20-40. PCM standard short frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=1) 543
Figure 20-41. PCM standard short frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0) 543
Figure20-42. PCM standard short frame synchronization mode timing diagram (DTLEN=01,

CHLEN=1, CKPL=1) 543
Figure 20-43. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0) 544
Figure 20-44. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1) 544
Figure 20-45. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=0) 544
Figure20-46. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1) 544
Figure 20-47. PCM standard long frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0) 544
Figure 20-48. PCM standard long frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=1) 544
Figure 20-49. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0) 545
Figure 20-50. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=1) 545
Figure 20-51. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0) 545
Figure 20-52. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1) 545
Figure 20-53. Block diagram of I2S clock generator 546
Figure 20-54. I2S initialization sequence 548
Figure 20-55. I2S master reception disabling sequence 550
Figure 21-1. I2C module block diagram 566
Figure 21-2. Data validation 567
Figure 21-3. START and STOP signal 567
Figure 21-4. Clock synchronization 568
Figure 21-5. SDA line arbitration 568
Figure 21-6. I2C communication flow with 7-bit address 569
Figure 21-7. I2C communication flow with 10-bit address (Master Transmit) 569
Figure 21-8. I2C communication flow with 10-bit address (Master Receive) 569
Figure 21-9. Programming model for slave transmitting (10-bit address mode) 571
Figure 21-10. Programming model for slave receiving (10-bit address mode) 572
Figure 21-11. Programming model for master transmitting (10-bit address mode) 573
Figure 21-12. Programming model for master receiving using Solution A (10-bit address mode) 575
Figure 21-13. Programming model for master receiving mode using solution B (10-bit address mode) 576
Figure 21-14. I2C module block diagram 597
Figure 21-15. Data validation 598
Figure 21-16. START and STOP signal 599
Figure 21-17. I2C communication flow with 10-bit address (Master Transmit) 599

Figure 21-18. I2C communication flow with 7-bit address (Master Transmit) 600
Figure 21-19. I2C communication flow with 7-bit address (Master Receive)..... 600
Figure 21-20. I2C communication flow with 10-bit address (Master Receive when HEAD10R=0) 600
Figure 21-21. I2C communication flow with 10-bit address (Master Receive when HEAD10R=1) 600
Figure 21-22. Data hold time 601
Figure 21-23. Data setup time 602
Figure 21-24. Data transmission 604
Figure 21-25. Data reception 604
Figure 21-26. I2C initialization in slave mode..... 607
Figure 21-27. Programming model for slave transmitting when SS=0 608
Figure 21-28. Programming model for slave transmitting when SS=1 609
Figure 21-29. Programming model for slave receiving 610
Figure 21-30. I2C initialization in master mode..... 611
Figure 21-31. Programming model for master transmitting (N<=255) 612
Figure 21-32. Programming model for master transmitting (N>255) 613
Figure 21-33. Programming model for master receiving (N<=255) 614
Figure 21-34. Programming model for master receiving (N>255) 615
Figure 21-35. SMBus master transmitter and slave receiver communication flow..... 619
Figure 21-36. SMBus master receiver and slave transmitter communication flow..... 619

List of Tables

Table 1-1. Memory map of GD32E25x series 26
Table 1-2. Boot modes 31
Table 2-1. NVIC exception types in Cortex®-M23 44
Table 2-2. Interrupt vector table 44
Table 2-3. EXTI source 47
Table 3-1. DMA transfer operation 55
Table 3-2. interrupt events 58
Table 3-3. DMA requests for each channel 60
Table 4-1. Base address and size for flash memory 67
Table 4-2. The relation between WSCNT and AHB clock frequency 69
Table 4-3. Option byte 75
Table 4-4. OB_WP bit for pages protected 77
Table 5-1. Power saving mode summary 92
Table 6-1. BKP interrupts control 99
Table 7-1. Clock source select 109
Table 7-2. Core domain voltage selected in Deep-sleep mode - 109
Table 8-1. GPIO configuration table 136
Table 9-1. Timers (TIMERx) are devided into six sorts 173
Table 9-2. Complementary outputs controlled by parameters 191
Table 9-3. Counting direction versus quadrature decoder signals 195
Table 9-4. Examples of slave mode 198
Table 9-5. Timerx(x=1,2) enable options 238
Table 9-6. Complementary outputs controlled by parameters 307
Table 9-7. Slave mode example table 311
Table 9-8. TIMERx(x=14) interconnection 314
Table 9-9. Complementary outputs controlled by parameters 346
Table 10-1. Min/max FWDGT timeout period at 40 kHz (IRC40K) 380
Table 10-2. Min-max timeout value at 72 MHz (f_{PCLK1}) 386
Table 13-1. ADC internal input signals 402
Table 13-2. ADC input pins definition 402
Table 13-3. External trigger for regular channels of ADC 414
Table 13-4. External trigger for inserted channels of ADC 414
Table 13-5. t_{CONV} timings depending on resolution 416
Table 13-6. Maximum output results for N and M combimations (grayed values indicates truncation) 418
Table 14-1. DAC I/O description 439
Table 14-2. DAC triggers and outputs summary 439
Table 14-3. External triggers of DAC 440
Table 15-1. CMP inputs and outputs summary 457
Table 16-1. CLAxSIGS0 input selection 464

Table 16-2. CLAxSIGS1 input selection 464
Table 16-3. LCU control 465
Table 19-1. Description of USART important pins 480
Table 19-2. Configuration of stop bits 481
Table 19-3. USART interrupt requests 495
Table 20-1. SPI signal description 519
Table 20-2. Quad-SPI signal description 520
Table 20-3. NSS function in slave mode 523
Table 20-4. NSS function in master mode 524
Table 20-5. SPI operation modes 524
Table 20-6. SPI interrupt requests 536
Table 20-7. I2S bitrate calculation formulas 546
Table 20-8. Audio sampling frequency calculation formulas 546
Table 20-9. Direction of I2S interface signals for each operation mode 547
Table 20-10. I2S interrupt 552
Table 21-1. Definition of I2C-bus terminology (refer to the I2C specification of PHILIPS semiconductors) 566
Table 21-2. Event status flags 580
Table 21-3. Error flags 581
Table 21-4. Definition of I2C-bus terminology (refer to the I2C specification of PHILIPS semiconductors) 597
Table 21-5. Data setup time and data hold time 603
Table 21-6. Communication modes to be shut down 604
Table 21-7. I2C error flags 620
Table 21-8. I2C interrupt events 621
Table 22-1. List of abbreviations used in register 637
Table 22-2. List of terms 638
Table 23-1. Revision history 639

1. System and memory architecture

The GD32E25x series are 32-bit general-purpose microcontrollers based on the Arm® Cortex®-M23 processor. The Cortex®-M23 processor includes AHB buses. All memory accesses of the Cortex®-M23 processor are executed on the AHB buses according to the different purposes and the target memory spaces. The memory organization uses ARMv8M architecture, pre-defined memory map and up to 4 GB of memory space, making the system flexible and extendable.

1.1. Arm® Cortex®-M23 processor

The Cortex®-M23 processor is an energy-efficient processor with a very low gate count. It is intended to be used for microcontroller and deeply embedded applications that require an area-optimized processor. It offers significant benefits to developers, including:

- A simple architecture that is easy to learn and program.
- Ultra-low power, energy-efficient operation.
- Excellent code density.
- Deterministic, high-performance interrupt handling.
- Upward compatibility with Cortex®-M processor family.

The processor delivers high energy efficiency through a small but powerful instruction set and extensively optimized design, providing high-end processing hardware including a single-cycle multiplier and a 17-cycle divider.

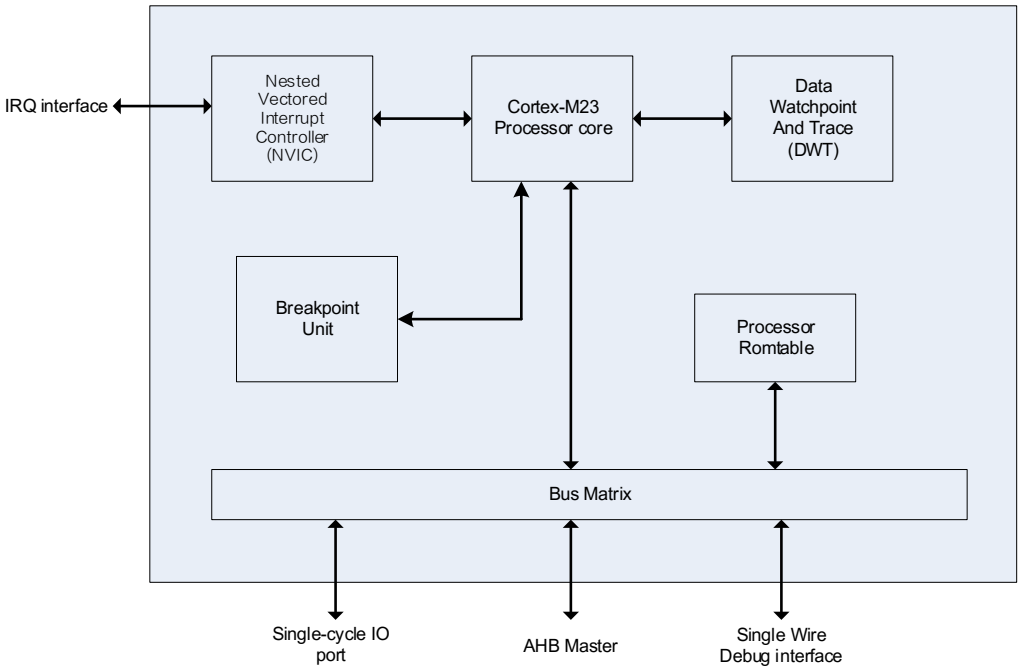
The Cortex®-M23 processor closely integrates a configurable Nested Vectored Interrupt Controller (NVIC), to deliver industry-leading interrupt performance.

Some system peripherals listed below are also provided by Cortex®-M23:

- Low latency, high-speed peripheral I / O port
- A Vector Table Offset Register
- Breakpoint unit
- Data Watchpoint
- Serial Wire Debug Port

The following figure shows the Cortex®-M23 processor block diagram. For more information, refer to the Arm® Cortex®-M23 Technical Reference Manual.

Figure 1-1. The structure of the Cortex®-M23 processor

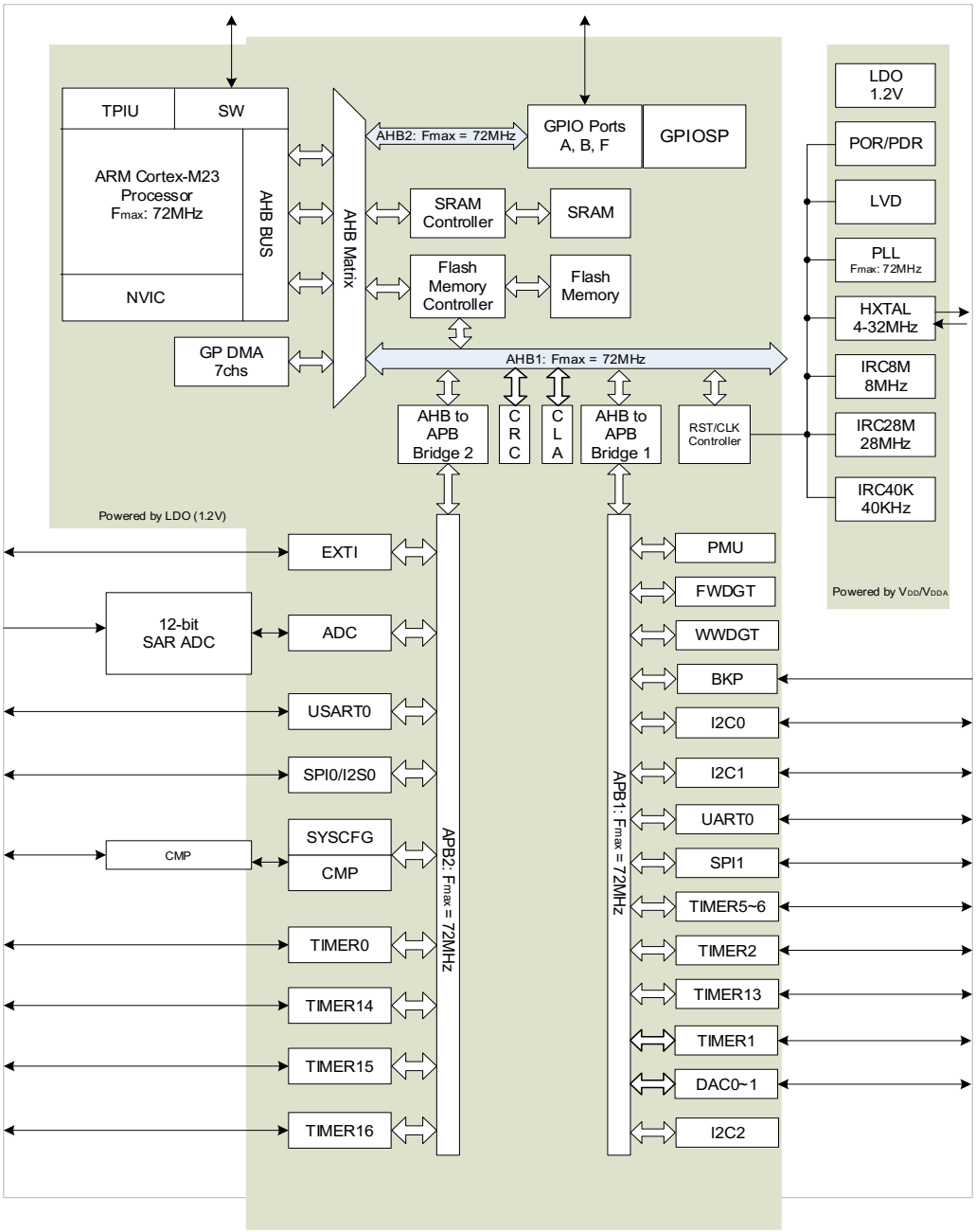


1.2. System architecture

The system architecture of GD32E25x series is shown in the following figure. The AHB matrix based on AMBA 5 AHB-LITE is a multi-layer AHB, which enables parallel access paths between multiple masters and slaves in the system. Two masters are on the AHB matrix, including AHB bus of the Cortex®-M23 core and DMA. The AHB matrix consists of four slaves, including the flash memory controller, internal SRAM, AHB1 and AHB2.

The AHB2 connects with the GPIO ports. The AHB1 connects with the AHB peripherals including two AHB-to-APB bridges which provide full synchronous connections between the AHB1 and the two APB buses. The two APB buses connect with all the APB peripherals.

Figure 1-2. Series system architecture of GD32E25x series



1.3. Memory map

Program memory, data memory, registers and I / O ports are organized within the same linear 4-Gbyte address space which is the maximum address range of the Cortex®-M23 since it has a 32-bit bus address width. Additionally, a pre-defined memory map is provided by the Cortex®-M23 processor to reduce the software complexity of repeated implementation of different device vendors. However, some regions are used by the Arm® Cortex®-M23 system peripherals. The following figure shows the memory map of GD32E25x series, including Code,

SRAM, peripheral, and other pre-defined regions. Each peripheral of either type is allocated 1KB of space. This allows simplifying the address decoding for each peripheral.

Table 1-1. Memory map of GD32E25x series

Pre-defined Regions	Bus	ADDRESS	Peripherals
		0xE000 0000 - 0xE00F FFFF	Cortex®-M23 internal peripherals
External Device		0xA000 0000 - 0xDFFF FFFF	Reserved
External RAM		0x60000000 - 0x9FFFFFFF	Reserved
Peripherals	AHB1	0x5004 0000 - 0x5FFF FFFF	Reserved
		0x5000 0000 - 0x5003 FFFF	Reserved
	AHB2	0x4800 1800 - 0x4FFF FFFF	Reserved
		0x4800 1800 - 0x4800 1BFF	Reserved
		0x4800 1400 - 0x4800 17FF	GPIOF
		0x4800 1000 - 0x4800 13FF	GPIOSP
		0x4800 0C00 - 0x4800 0FFF	Reserved
		0x4800 0800 - 0x4800 0BFF	Reserved
		0x4800 0400 - 0x4800 07FF	GPIOB
		0x4800 0000 - 0x4800 03FF	GPIOA
	AHB1	0x4003 8400 - 0x47FF FFFF	Reserved
		0x4003 8000 - 0x4003 83FF	CLA
		0x4002 4000 - 0x4003 7FFF	Reserved
		0x4002 3400 - 0x4002 3FFF	Reserved
		0x4002 3000 - 0x4002 33FF	CRC
		0x4002 2400 - 0x4002 2FFF	Reserved
		0x4002 2000 - 0x4002 23FF	FMC
		0x4002 1400 - 0x4002 1FFF	Reserved
		0x4002 1000 - 0x4002 13FF	RCU
		0x4002 0400 - 0x4002 0FFF	Reserved
		0x4002 0000 - 0x4002 03FF	DMA
	APB2	0x4001 8000 - 0x4001 FFFF	Reserved
		0x4001 5C00 - 0x4001 7FFF	Reserved
		0x4001 5800 - 0x4001 5BFF	DBG
		0x4001 4C00 - 0x4001 57FF	Reserved
		0x4001 4800 - 0x4001 4BFF	TIMER16
		0x4001 4400 - 0x4001 47FF	TIMER15
		0x4001 4000 - 0x4001 43FF	TIMER14
		0x4001 3C00 - 0x4001 3FFF	Reserved
		0x4001 3800 - 0x4001 3BFF	USART0
		0x4001 3400 - 0x4001 37FF	Reserved
		0x4001 3000 - 0x4001 33FF	SPI0/I2S0
		0x4001 2C00 - 0x4001 2FFF	TIMER0
		0x4001 2800 - 0x4001 2BFF	Reserved

Pre-defined Regions	Bus	ADDRESS	Peripherals
		0x4001 2400 - 0x4001 27FF	ADC
		0x4001 0800 - 0x4001 23FF	Reserved
		0x4001 0400 - 0x4001 07FF	EXTI
		0x4001 0000 - 0x4001 03FF	SYSCFG + CMP
	APB1	0x4000 D000 - 0x4000 FFFF	Reserved
		0x4000 CC00 - 0x4000 CFFF	DAC1
		0x4000 C800 - 0x4000 CBFF	Reserved
		0x4000 C400 - 0x4000 C7FF	Reserved
		0x4000 C000 - 0x4000 C3FF	I2C2
		0x4000 8000 - 0x4000 BFFF	Reserved
		0x4000 7C00 - 0x4000 7FFF	Reserved
		0x4000 7800 - 0x4000 7BFF	Reserved
		0x4000 7400 - 0x4000 77FF	DAC0
		0x4000 7000 - 0x4000 73FF	PMU
		0x4000 6400 - 0x4000 6FFF	Reserved
		0x4000 6000 - 0x4000 63FF	Reserved
		0x4000 5C00 - 0x4000 5FFF	Reserved
		0x4000 5800 - 0x4000 5BFF	I2C1
		0x4000 5400 - 0x4000 57FF	I2C0
		0x4000 4800 - 0x4000 53FF	Reserved
		0x4000 4400 - 0x4000 47FF	UART1
		0x4000 4000 - 0x4000 43FF	Reserved
		0x4000 3C00 - 0x4000 3FFF	Reserved
		0x4000 3800 - 0x4000 3BFF	SPI1
		0x4000 3400 - 0x4000 37FF	Reserved
		0x4000 3000 - 0x4000 33FF	FWDGT
		0x4000 2C00 - 0x4000 2FFF	WWDGT
		0x4000 2800 - 0x4000 2BFF	BKP
		0x4000 2400 - 0x4000 27FF	Reserved
		0x4000 2000 - 0x4000 23FF	TIMER13
		0x4000 1800 - 0x4000 1FFF	Reserved
		0x4000 1400 - 0x4000 17FF	TIMER6
		0x4000 1000 - 0x4000 13FF	TIMER5
		0x4000 0800 - 0x4000 0FFF	Reserved
		0x4000 0400 - 0x4000 07FF	TIMER2
		0x4000 0000 - 0x4000 03FF	TIMER1
SRAM		0x2000 2000 - 0x3FFF FFFF	Reserved
		0x2000 0000 - 0x2000 1FFF	SRAM
Code		0x1FFF F810 - 0x1FFF FFFF	Reserved
		0x1FFF F800 - 0x1FFF F80F	Option bytes

Pre-defined Regions	Bus	ADDRESS	Peripherals
		0x1FFF EC00 - 0x1FFF F7FF	System memory
		0x1FFF 7400 - 0x1FFF EBFF	Reserved
		0x1FFF 7000 - 0x1FFF 73FF	OTP bytes
		0x0801 0000 - 0x1FFF 6FFF	Reserved
		0x0800 0000 - 0x0800 FFFF	Main Flash memory
		0x0001 0000 - 0x07FF FFFF	Reserved
		0x00000000 - 0x0000FFFF	Aliased to Flash or system memory

1.3.1. On-chip SRAM memory

The GD32E25x series contain up to 8KB of on-chip SRAM which starts at the address 0x2000 0000. It supports byte, half-word (16 bits), and word (32 bits) accesses.

Parity check

In order to increase memory robustness, parity check is supported. The user can enable the parity check function using the bit SRAM_PARITY_ECC_CHECK in the user option byte (refer to ***Option byte description***. When enabled, an SRAM_PCEF is generated if the parity checks fail. The SRAM parity check error flag is implemented in the system configuration register 1 (SYSCFG_CFG1). The error flag can be connected to the break input of TIMER0/ TIMER14/ TIMER15/ TIMER16, if the SRAM_PARITY_ERROR_LOCK control bit in the system configuration register 1 (SYSCFG_CFG1) is set to 1. If you need to find the SRAM parity check error address, you can look it up through the SRAM_ERRADDR[10:0] bits in the SYSCFG_CFG0 register.

The real data width of the SRAM is 36 bits, including 32 bits for data and 4 bits for parity (1 bit per byte). When writing, the parity bits are computed and stored into the SRAM. When reading, the parity bits are also computed using the stored data in SRAM. The computed parity bits are compared with the stored parity bits which are computed during the writing access. If they are different, the parity checks fail.

Note: When enabling the SRAM parity check, it is recommended to initialize the whole SRAM memory by software at the beginning of the code, in order to avoid getting parity check errors when reading non-initialized locations. Additionally, the SRAM operations only occur when flash is ready. After system reset and power reset, the SRAM functions cannot be read or written during SYSCFG and option byte updates.

ECC

When reading and writing SRAM, it supports 7-bit ECC function. It can correct 1-bit error and detect multiple bits' error.

It must be written before reading SRAM, otherwise it may cause ECC error. Unaligned read operations will be performed in accordance with 32-bit read operations. Non-aligned write operations will produce a read-modify-write process. For example, when 16-bit data is written into SRAM, firstly the another 16-bit data is read out from the SRAM, and the 16-bit that need to be written are combined to a 32-bit data, and finally the 32-bit data is written into the SRAM together. Therefore, when initializing SRAM, it can only be written in a 32-bit width.

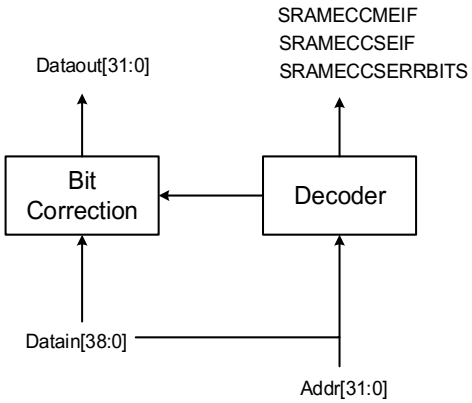
The ECC module is composed of an encoder and a decoder.

Encoder: When performing a SRAM write operation, a 7-bit ECC code will be generated and written into the SRAM together with the data.

Decoder: When performing a SRAM read operation, it uses the same algorithm as the encoder to decode and generate a ECC code. The ECC code includes ECC error status and information which specific bit of the 32-bit data has single bit error. The ECC single-bit correctable error bit can be checked through the ECCSERRBITS0 in the SYSCFG_SRAM_ECC register. If you need to find the last address where an ECC event occurred on the SRAM, you can look it up through the ECCEADDR0 in the SYSCFG_SRAM_ECC register.

The decoder is shown in the *Figure 1-3. ECC decoder* below:

Figure 1-3. ECC decoder



EEIC

The EEIC (ECC Error Interrupt Control) module provides the function of ECC error status management and ECC interrupt configuration.

Single bit error correction event

When a single-bit error correction event is detected in SRAM, EEIC:

(1) The ECCSEIF0 bit in SYSCFG_STAT register will be set. Software can clear it by writing 1.

(2) The ECCEADDR0 bit in SYSCFG_SRAM_ECC records the address where the single-

bit error correction event occurred.

Multi-bits non-correction error event

When a multi-bits non-correction error event is detected in SRAM, EEIC：

(1) The ECCMEIF0 bit in SYSCFG_STAT register will be set. Software can clear it by writing 1.

(2) The ECCEADDR0 bit in SYSCFG_SRAM_ECC records the address where the multi-bits non-correction error event occurred.

Single bit error correction interrupt

Set the ECCSEIE0 bit in SYSCFG_SRAM_ECC register. When a single-bit error correctable event is detected, a corresponding interrupt IRQ 35 will be generated.

Multi-bits non-correction error interrupt

Set the ECCMEIE0 bit in SYSCFG_SRAM_ECC register. When a multi-bits error non-correction event is detected, a NMI interrupt will be generated.

1.3.2. On-chip Flash memory

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

- Up to 64 KB of on-chip flash memory for storing instruction/data.
- 3K bytes information block for bootloader.
- 16 bytes' option bytes block for user application requirements.
- 1 KB OTP (One-time program) block used for user data storage.

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

1.4. Boot configuration

The GD32E252xx series provides three kinds of boot sources which can be selected using the bit BOOT1_n in the user option byte (refer to ***Option byte description***) and the BOOT0 pins. The value on the BOOT0 pin is latched on the 4th rising edge of SYSCLK after a reset. It is up to the user to set the BOOT1_n and BOOT0 after a power-on reset or a system reset to select the required boot source. The default boot source for the GD32E252xx series is the main flash.

The GD32E251xx series provides three kinds of boot sources which can be selected using the bit BOOT1_n and the bit BOOT0_n in the user option byte. It is up to the user to set the BOOT1_n and BOOT0_n after a power-on reset or a system reset to select the required boot source. The default boot source for the GD32E251xx series is the system memory.

The details are shown in the following table.

Table 1-2. Boot modes

Selected boot source	Boot mode selection pins	
	BOOT1	BOOT0
Main Flash Memory	x	0
System Memory	0	1
On-chip SRAM	1	1

1. The BOOT1 value is the opposite of the BOOT1_n value.
2. For GD32E252xx series, the BOOT0 value is mapping to the BOOT0 pin.
3. For GD32E251xx series, the BOOT0 value is the opposite of the BOOT0_n value.

After power-on sequence or a system reset, the Arm® Cortex®-M23 processor fetches the top-of-stack value from address 0x0000 0000 and the base address of boot code from 0x0000 0004 in sequence. Then, it starts executing code from the base address of boot code.

According to the selected boot source, either the main flash memory (original memory space beginning at 0x0800 0000) or the system memory (original memory space beginning at 0x1FFF EC00) is aliased in the boot memory space which begins at the address 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 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. The boot loader can be activated through I2C interface.

Note: To ensure system reliability and stability, during the hardware design phase, refer to the Boot mode section of the datasheet for the corresponding model. Confirm that peripherals which may generate interference signals (such as sensor outputs, watchdog chips, keys, etc.) are not connected to the pins used by the Bootloader, so as to avoid affecting the normal function of the Bootloader and causing abnormal outputs from the connected peripherals.

1.5. System configuration registers (SYSCFG)

SYSCFG base address: 0x4001 0000

1.5.1. System configuration register 0 (SYSCFG_CFG0)

Address offset: 0x00

Reset value: 0x0000 000X (X indicates BOOT_MODE [1:0])

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
SRAM_ERRADDR[10:0]											Reserved	PB9FMPEN	Reserved	PB7FMPEN	PB6FMPEN
r												rw		rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			TIMER16_DMA_RMP	TIMER15_DMA_RMP	USART0_RX_DMA_RMP	USART0_TX_DMA_RMP	ADC_DMA_RMP	Reserved	BOOT0_PB9_RMP	Reserved	PA11_PA12_RMP	Reserved		BOOT_MODE[1:0]	
			rw	rw	rw	rw	rw		rw		rw			r	

Bits	Fields	Descriptions
31:21	SRAM_ERRADDR[10:0]	SRAM parity check error address. **Note:** Physical SRAM word address = SRAM base address/4 + SRAM_ERRADDR These bits are read only.
20	Reserved	Must be kept at reset value.
19	PB9FMPEN	I2C Fm+ or IFRP mode on PB9 pin enable This bit controls I2C Fm+ or IFRP mode, the speed control of the pin is bypassed. 0: Disable mode on the PB9 pin 1: Enable mode on the PB9 pin
18	Reserved	Must be kept at reset value.
17	PB7FMPEN	I2C Fm+ mode on PB7 pin enable This bit controls I2C Fm+ mode, the speed control of the pin is bypassed. 0: Disable Fm+ mode on the PB7 pin 1: Enable Fm+ mode on the PB7 pin
16	PB6FMPEN	I2C Fm+ mode on PB6 pin enable This bit controls I2C Fm+ mode, the speed control of the pin is bypassed. 0: Disable Fm+ mode on the PB6 pin 1: Enable Fm+ mode on the PB6 pin
15:13	Reserved	Must be kept at reset value.
12	TIMER16_DMA_RMP	Timer 16 DMA request remapping enable 0: No remap (TIMER16_CH0 and TIMER16_UP DMA requests are mapped on

		DMA channel 0) 1: Remap (TIMER16_CH0 and TIMER16_UP DMA requests are mapped on DMA channel 1)
11	TIMER15_DMA_RMP	Timer 15 DMA request remapping enable 0: No remap (TIMER15_CH0 and TIMER15_UP DMA requests are mapped on DMA channel 2) 1: Remap (TIMER15_CH0 and TIMER15_UP DMA requests are mapped on DMA channel 3)
10	USART0_RX_DMA_RMP	USART0_RX DMA request remapping enable 0: No remap (USART0_RX DMA requests are mapped on DMA channel 2) 1: Remap (USART0_RX DMA requests are mapped on DMA channel 4)
9	USART0_TX_DMA_RMP	USART0_TX DMA request remapping enable 0: No remap (USART0_TX DMA requests are mapped on DMA channel 1) 1: Remap (USART0_TX DMA requests are mapped on DMA channel 3)
8	ADC_DMA_RMP	ADC DMA request remapping enable 0: No remap (ADC DMA requests are mapped on DMA channel 0) 1: Remap (ADC DMA requests are mapped on DMA channel 1)
7	Reserved	Must be kept at reset value.
6	BOOT0_PB9_RMP	BOOT0 and PB9 remapping bit. This bit is set and cleared by software. When PIN_RSTMD is set, this bit will be retained across all reset events except POR. It controls the mapping of either BOOT0 or PB9 function on the BOOT0 pin. When PIN_RSTMD and BOOT0_PB9_RMP are both set, the BOOT0 function is tied to 0 by hardware after system reset. In this case, the system will boot from main flash without regard to the input value from the BOOT0 pin. 0: No remap (BOOT0 function is mapping on the BOOT0 pin) 1: Remap (PB9 function is mapping on the BOOT0 pin) **Note**: for the GD32E251xx series, this bit is reserved
5	Reserved	Must be kept at reset value.
4	PA11_PA12_RMP	PA11 and PA12 remapping bit for small packages (28 and 20 pins). This bit is set and cleared by software. When PIN_RSTMD is set, this bit will be retained across all reset events except POR. It controls the mapping of either PA9/10 or PA11/12 pin pair on small pin-count packages. 0: No remap (PA9/10 pin pair mapped on the pins) 1: Remap (PA11/12 pin pair mapped on the pins)
3:2	Reserved	Must be kept at reset value.
1:0	BOOT_MODE[1:0]	Boot mode For the GD32E252xx series, bit0 is mapping to the BOOT0 pin; for the GD32E251xx

series, the value of bit0 is the opposite of the BOOT0_n option bit value; the value of bit1 is the opposite of the BOOT1_n option bit value.

x0: Boot from the Main Flash

01: Boot from the System Flash memory

11: Boot from the embedded SRAM

1.5.2. EXTI sources selection register 0 (SYSCFG_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			

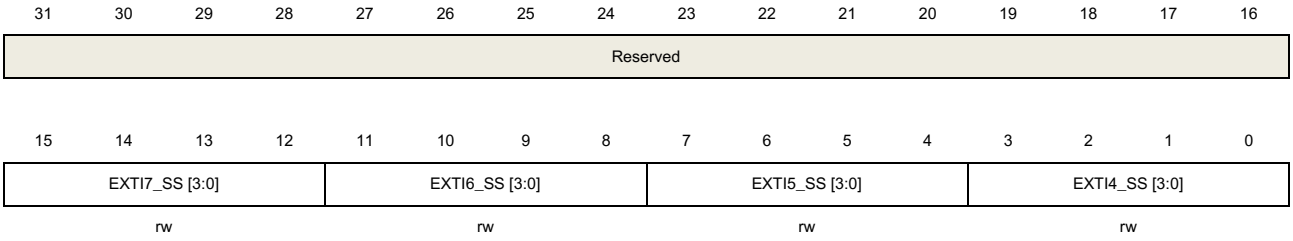
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	EXTI3_SS[3:0]	EXTI 3 sources selection X000: PA3 pin X001: PB3 pin X010-X111: PB3 pin (SDCTL=0), Reserved (SDCTL=1) SDCTL bit in EXTI_SDCTL determines the sources selection of X010-X111.
11:8	EXTI2_SS[3:0]	EXTI 2 sources selection X000: PA2 pin X001: PB2 pin X010-X111: PB2 pin (SDCTL=0), Reserved (SDCTL=1) SDCTL bit in EXTI_SDCTL determines the sources selection of X010-X111.
7:4	EXTI1_SS[3:0]	EXTI 1 sources selection X000: PA1 pin X001: PB1 pin X010-X100: PF1 pin (SDCTL=0), Reserved (SDCTL=1) X110-X111: PF1 pin (SDCTL=0), Reserved (SDCTL=1) SDCTL bit in EXTI_SDCTL determines the sources selection of X011-X111, except X101.
3:0	EXTI0_SS[3:0]	EXTI 0 sources selection X000: PA0 pin X001: PB0 pin

X010-X100: PF0 pin (SDCTL=0), Reserved (SDCTL=1)
X110-X111: PF0 pin (SDCTL=0), Reserved (SDCTL=1)
SDCTL bit in EXTI_SDCTL determines the sources selection of X011-X111, except X101.

1.5.3. EXTI sources selection register 1 (SYSCFG_EXTISS1)

Address offset: 0x0C
Reset value: 0x0000 0000

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



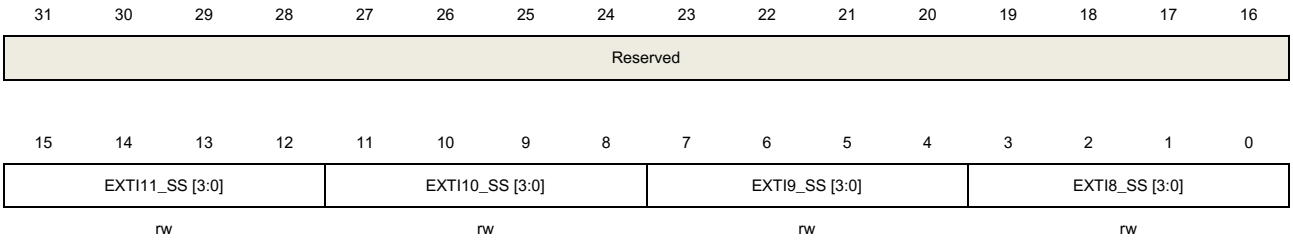
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	EXTI7_SS[3:0]	EXTI 7 sources selection X000: PA7 pin X001: PB7 pin X010-X111: Reserved
11:8	EXTI6_SS[3:0]	EXTI 6 sources selection X000: PA6 pin X001: PB6 pin X010-X111: Reserved
7:4	EXTI5_SS[3:0]	EXTI 5 sources selection X000: PA5 pin X001: PB5 pin X010-X111: PB5 pin (SDCTL=0), Reserved (SDCTL=1) SDCTL bit in EXTI_SDCTL determines the sources selection of X010-X111.
3:0	EXTI4_SS[3:0]	EXTI 4 sources selection X000: PA4 pin X001: PB4 pin X010-X111: PB4 pin (SDCTL=0), Reserved (SDCTL=1) SDCTL bit in EXTI_SDCTL determines the sources selection of X010-X111.

1.5.4. EXTI sources selection register 2 (SYSCFG_EXTISS2)

Address offset: 0x10

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	EXTI11_SS[3:0]	EXTI 11 sources selection X000: PA11 pin X001: PB3 pin X010-X111: PB3 pin (SDCTL=0), Reserved (SDCTL=1) SDCTL bit in EXTI_SDCTL determines the sources selection of X010-X111.
11:8	EXTI10_SS[3:0]	EXTI 10 sources selection X000: PA10 pin X001: PB10 pin X010-X111: PB10 pin (SDCTL=0), Reserved (SDCTL=1) SDCTL bit in EXTI_SDCTL determines the sources selection of X010-X111.
7:4	EXTI9_SS[3:0]	EXTI 9 sources selection X000: PA9 pin X001: PB9 pin X010-X111: PB9 pin (SDCTL=0), Reserved (SDCTL=1) SDCTL bit in EXTI_SDCTL determines the sources selection of X010-X111.
3:0	EXTI8_SS[3:0]	EXTI 8 sources selection X000: PA8 pin X001: PB8 pin X010-X111: PB8 pin (SDCTL=0), Reserved (SDCTL=1) SDCTL bit in EXTI_SDCTL determines the sources selection of X010-X111.

1.5.5. EXTI sources selection register 3 (SYSCFG_EXTISS3)

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
EXTI15_SS [3:0]				EXTI14_SS [3:0]				EXTI13_SS [3:0]				EXTI12_SS [3:0]			
rw				rw				rw				rw			

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	EXTI15_SS[3:0]	EXTI 15 sources selection X000: PA15 pin X001: PB3 pin X010-X111: PB3 pin (SDCTL=0), Reserved (SDCTL=1) SDCTL bit in EXTI_SDCTL determines the sources selection of X010-X111.
11:8	EXTI14_SS[3:0]	EXTI 14 sources selection X000: PA14 pin X001-X111: Reserved
7:4	EXTI13_SS[3:0]	EXTI 13 sources selection X000: PA13 pin X001-X111: Reserved
3:0	EXTI12_SS[3:0]	EXTI 12 sources selection X000: PA12 pin X001-X111: Reserved

1.5.6. System configuration register 1 (SYSCFG_CFG1)

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							PIN_RSTMD	Reserved							VREF_EN
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							SRAM_PCEF	Reserved					LVD_LOCK	SRAM_PARITY_ERROR_LOCK	LOCKUP_LOCK
							rc_w1						rw	rw	rw

Bits	Fields	Descriptions

31:25	Reserved	Must be kept at reset value.
24	PIN_RSTMD	Pin reset mode This bit controls whether the GPIO pin configurations (GPIOx_CTL, GPIOx_OMODE, GPIOx_OSPD, GPIOx_PUD, GPIOx_OCTL, BOOT0_PB9_RMP (only for the GD32E252xx series), and PA11_PA12_RMP) are reset when any reset event occurs. This bit will be retained across all reset events except POR. After a POR event, this bit is reset to 0. 0: GPIO pin configuration is reset when any reset event occurs. 1: GPIO pin configuration will retain state across any reset event except for the POR event.
23:17	Reserved	Must be kept at reset value.
16	VREF_EN	VREF enable This bit is set and cleared by software. When any of DACx DRSTMDx is set, this bit will be retained across all reset events except POR. It controls whether the VREF function is enabled or not. 0: VREF is disabled 1: VREF is enabled
15:9	Reserved	Must be kept at reset value.
8	SRAM_PCEF	SRAM parity check error flag This bit is set by hardware when an SRAM parity check error occurs. It is cleared by software by writing 1. 0: No SRAM parity check error detected 1: SRAM parity check error detected
7:3	Reserved	Must be kept at reset value.
2	LVD_LOCK	LVD lock This bit is set by software and cleared by reset system only. 0: The LVD interrupt is disconnected from the break input of TIMER0/14/15/16. LVDEN and LVDT[2:0] in the PMU_CTL register can be programmed. 1: The LVD interrupt is connected from the break input of TIMER0/14/15/16. LVDEN and LVDT[2:0] in the PMU_CTL register are read only.
1	SRAM_PARITY_ERROR_LOCK	SRAM parity check error lock This bit is set by software and cleared by reset system only. 0: The SRAM parity check error is disconnected from the break input of TIMER0/14/15/16 1: The SRAM parity check error is connected from the break input of TIMER0/14/15/16.
0	LOCKUP_LOCK	Cortex®-M23 LOCKUP output lock This bit is set by software and cleared by reset system only. 0: The Cortex®-M23 LOCKUP output is disconnected from the break input of

TIMER0/14/15/16.

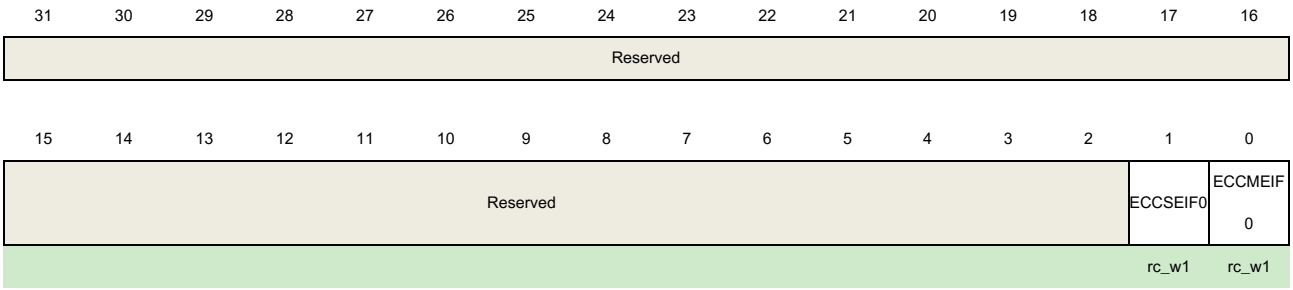
1: The Cortex®-M23 LOCKUP output is connected from the break input of TIMER0/14/15/16.

1.5.7. SYSCFG status register (SYSCFG_STAT)

Address offset: 0x24

Reset value: 0x0000 0000

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



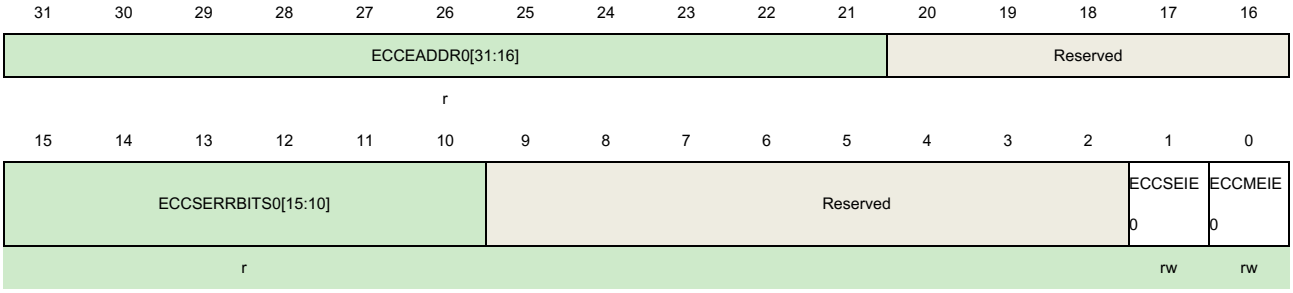
Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	ECCSEIF0	SRAM single bit correction event flag The software can clear it by writing 1. 0: no SRAM single bit correction event is detected. 1: SRAM single bit correction event is detected.
0	ECCMEIF0	SRAM multi bits non-correction event flag The software can clear it by writing 1. 0: no SRAM multi bits non-correction event is detected. 1: SRAM multi bits non-correction event is detected.

1.5.8. SRAM_ECC register (SYSCFG_SRAM_ECC)

Address offset: 0x28

Reset value: 0x0000 0000

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



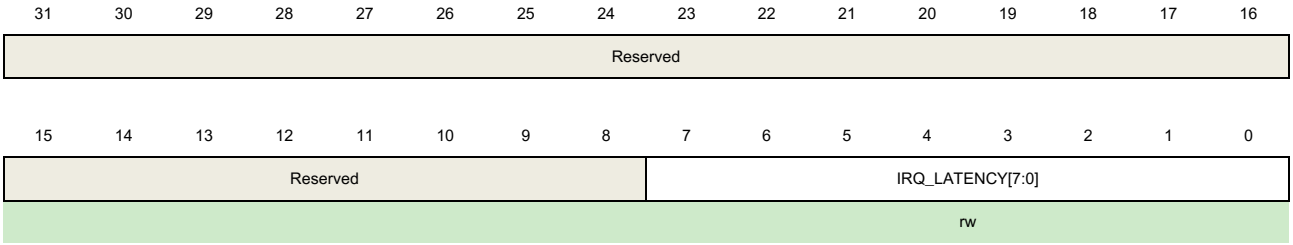
Bits	Fields	Descriptions
31:21	ECCEADDR0[31:21]	Indicates the last address of ECC event on SRAM occurred Physical SRAM word address = SRAM base address / 4 + ECCEADDR1.
20:16	Reserved	Must be kept at reset value.
15:10	ECCSERRBITS0[15:10]	Indicates the error bit Which one bit has an ECC single-bit correctable error. 0: no error 1: bit 0 … 39: bit 39
9:2	Reserved	Must be kept at reset value.
1	ECCSEIE0	SRAM single bit correction interrupt enable 0: SRAM single bit correction interrupt is disabled. 1: SRAM single bit correction interrupt is enabled.
0	ECCMEIE0	SRAM multi bits non-correction interrupt enable 0: SRAM non-correction interrupt is disabled. 1: SRAM non-correction interrupt is enabled.

1.5.9. IRQ latency register (SYSCFG_CPU_IRQ_LAT)

Address offset: 0x100

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	IRQ_LATENCY[7:0]	IRQ_LATENCY specifies the minimum number of cycles between an interrupt that becomes pended in the NVIC, and the vector fetch for that interrupt being issued on the AHB-Lite interface. If IRQ_LATENCY is set to 0, interrupts are taken as quickly as possible. For non-zero values, the Cortex®-M23 processor ensures that a minimum of IRQ_LATENCY+1 hclk cycles exist between an interrupt becoming pended in the

NVIC and the vector fetch for the interrupt being performed.

1.6. Device electronic signature

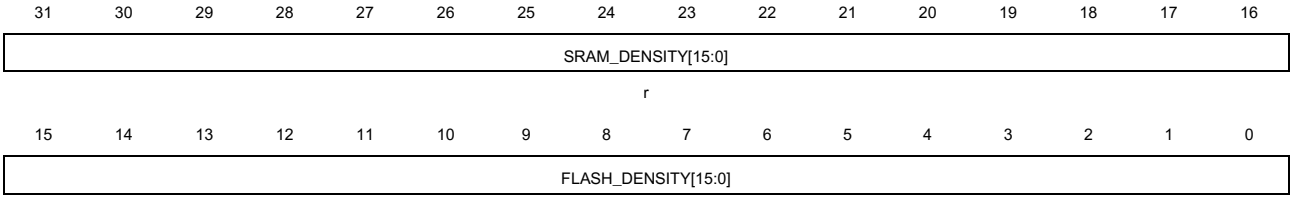
The device electronic signature contains memory density information and the 96-bit unique device ID. It is stored in the information block of the Flash memory. The 96-bit unique device ID is unique for any device. It can be used as serial numbers, or part of security keys, etc.

1.6.1. Memory density information

Base address: 0x1FFF F7E0

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

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



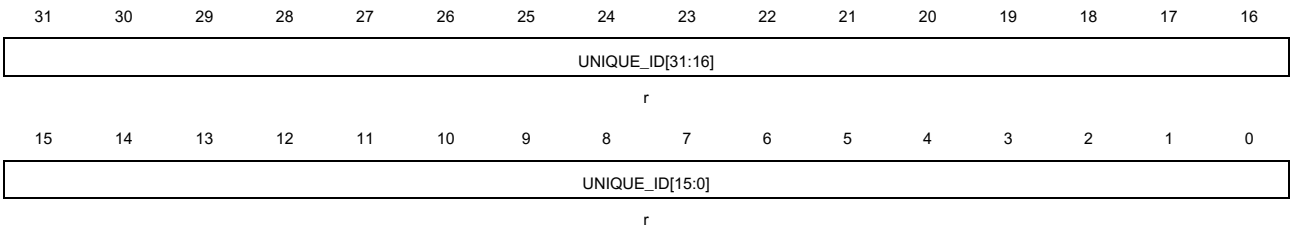
Bits	Fields	Descriptions
31:16	SRAM_DENSITY [15:0]	SRAM density The value indicates the on-chip SRAM density of the device in Kbytes. Example: 0x0008 indicates 8 Kbytes.
15:0	FLASH_DENSITY [15:0]	Flash memory density The value indicates the Flash memory density of the device in Kbytes. Example: 0x0020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

1.6.2. Unique device ID (96 bits)

Base address: 0x1FFF F7AC

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

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



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

Base address: 0x1FFF F7B0

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

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
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 F7B4

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

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
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

2. Interrupt / event controller (EXTI)

2.1. Overview

Cortex®-M23 integrates the Nested Vectored Interrupt Controller (NVIC) for efficient exception and interrupts processing. NVIC facilitates low-latency exception and interrupt power management controls. It's tightly coupled to the processer core. You can read the Technical Reference Manual of Cortex®-M23 for more details about NVIC.

EXTI (interrupt / event controller) contains up to 24 independent edge detectors and generates interrupt requests or events to the processer. 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.

2.2. Characteristics

- Cortex®-M23 system exception
- Up to 36 maskable peripheral interrupts
- 2 bits interrupt priority configuration - 4 priority levels
- Efficient interrupt processing
- Support exception pre-emption and tail-chaining
- Wake up system from power saving mode
- Up to 24 independent edge detectors in EXTI
- Three trigger types: rising, falling and both edges
- Software interrupt or event trigger
- Trigger sources configurable
- External pin EXTI source default can be configured 0(invalid)

2.3. Interrupts function overview

The ARM Cortex®-M23 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 2-1. NVIC exception types in Cortex®-M23

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

The SysTick calibration value is 9000 and SysTick clock frequency is fixed to HCLK*0.125. So this will give a 1ms SysTick interrupt if HCLK is configured to 72MHz.

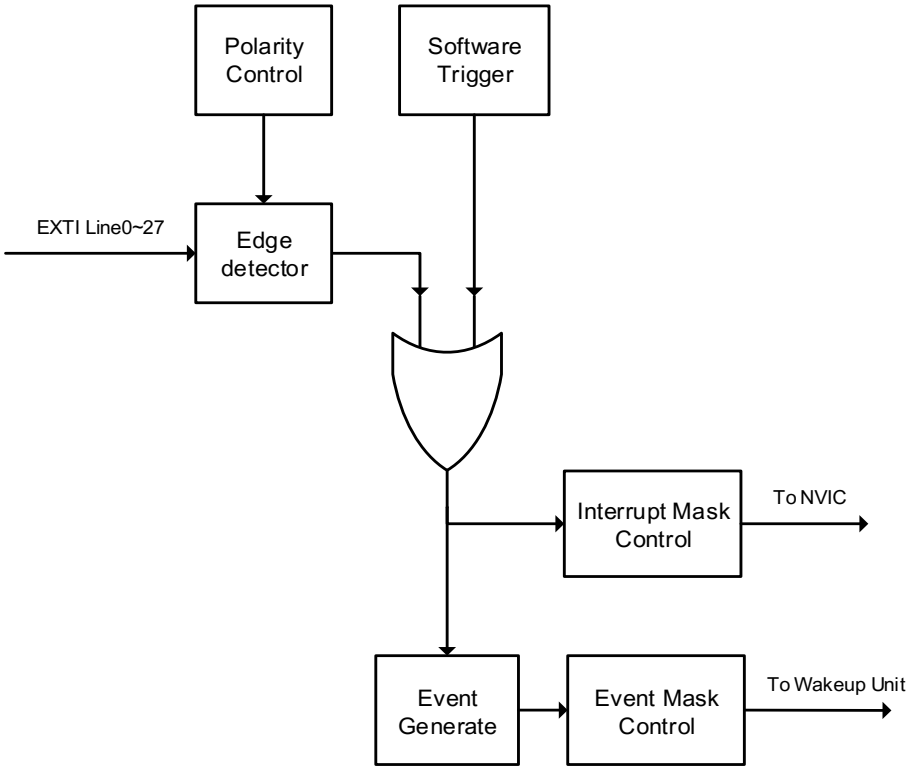
Table 2-2. Interrupt vector table

Interrupt number	Vector number	Peripheral Interrupt description	Vector address
IRQ 0	16	WWDGT interrupt	0x0000_0040
IRQ 1	17	LVD from EXTI interrupt	0x0000_0044
IRQ 2	18	BKP tamper interrupt	0x0000_0048
IRQ 3	19	FMC global interrupt	0x0000_004C
IRQ 4	20	RCU global interrupt	0x0000_0050
IRQ 5	21	EXTI Line0-1 interrupt	0x0000_0054
IRQ 6	22	EXTI Line2-3 interrupt	0x0000_0058
IRQ 7	23	EXTI Line4-15 interrupt	0x0000_005C
IRQ 8	24	EXTI Line 20/22 interrupt	0x0000_0060
IRQ 9	25	DMA channel0 global interrupt	0x0000_0064
IRQ 10	26	DMA channel1-2 global interrupt	0x0000_0068
IRQ 11	27	DMA channel3-4 global interrupt	0x0000_006C
IRQ 12	28	ADC and EXTI Line 21 CMP interrupt	0x0000_0070
IRQ 13	29	TIMER0 break, update, trigger and commutation interrupt	0x0000_0074

Interrupt number	Vector number	Peripheral Interrupt description	Vector address
IRQ 14	30	TIMER0 capture compare interrupt	0x0000_0078
IRQ 15	31	TIMER1 global interrupt	0x0000_007C
IRQ 16	32	TIMER2 global interrupt	0x0000_0080
IRQ 17	33	TIMER5 and DAC global interrupt	0x0000_0084
IRQ 18	34	TIMER6 interrupt	0x0000_0088
IRQ 19	35	TIMER13 global interrupt	0x0000_008C
IRQ 20	36	TIMER14 global interrupt	0x0000_0090
IRQ 21	37	TIMER15 global interrupt	0x0000_0094
IRQ 22	38	TIMER16 global interrupt	0x0000_0098
IRQ 23	39	I2C0 event interrupt	0x0000_009C
IRQ 24	40	I2C1 event interrupt	0x0000_00A0
IRQ 25	41	SPI0 global interrupt	0x0000_00A4
IRQ 26	42	SPI1 global interrupt	0x0000_00A8
IRQ 27	43	USART0 global interrupt	0x0000_00AC
IRQ 28	44	UART1 global interrupt	0x0000_00B0
IRQ 29	45	I2C2 event interrupt	0x0000_00B4
IRQ 30	46	I2C2 error interrupt	0x0000_00B8
IRQ 31	47	GPIO port not match interrupt	0x0000_00BC
IRQ 32	48	I2C0 error interrupt	0x0000_00C0
IRQ 33	49	Through EXTI Line EXTI18:I2C2 wakeup	0x0000_00C4
IRQ 34	50	I2C1 error interrupt	0x0000_00C8
IRQ 35	51	ECC single bit error interrupt	0x0000_00CC
IRQ 36	52	Reserved	0x0000_00D0
IRQ 37	53	Reserved	0x0000_00D4
IRQ 38	54	Reserved	0x0000_00D8
IRQ 39	55	CLA global interrupt	0x0000_00DC
IRQ 40-41	56-57	Reserved	0x0000_00E0-0x0000_00E4
IRQ 42	58	Reserved	0x0000_00E8
IRQ 43-47	59-63	Reserved	0x0000_00EC-0x0000_00FC
IRQ 48	64	DMA channel5-6 global interrupt	0x0000_0100
IRQ 49-50	65-66	Reserved	0x0000_0104-0x0000_0108
IRQ 51	67	Reserved	0x0000_010C
IRQ52-66	68-82	Reserved	0x0000_0110-0x0000_0148
IRQ67	83	Reserved	0x0000_014C

2.4. External interrupt and event block diagram

Figure 2-1. Block diagram of EXTI



2.5. External interrupt and event function overview

The EXTI contains up to 24 independent edge detectors and generates interrupts request or event to the processer. 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 8 lines from internal modules which refers to ***Table 2-3. EXTI source***.All GPIO pins can be selected as an EXTI trigger source by configuring SYSCFG_EXTISSx registers in SYSCFG module (please refer to ***System configuration registers (SYSCFG)*** section for detail).

EXTI can provide not only interrupts but also event signals to the processor. The Cortex®-M23 processor fully implements 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 event. EXTI can be used to wake up processor and the whole system when some expected event occurs, such as a special GPIO pin toggling.

Table 2-3. EXTI source

EXTI line number	Source
0	PA0 / PB0 / PF0
1	PA1 / PB1 / PF1
2	PA2 / PB2
3	PA3 / PB3
4	PA4 / PB4
5	PA5 / PB5
6	PA6 / PB6
7	PA7 / PB7
8	PA8 / PB8
9	PA9 / PB9
10	PA10 / PB10
11	PA11 / PB3
12	PA12
13	PA13
14	PA14
15	PA15 / PB3
16	LVD(power low voltage detected)
17	Reserved
18	I2C2 wakeup
19	Tamper
20	PB9 / PB8 / PF0 / PB5 / PB4 / PB3 / PA15
21	CMP output
22	PB9 / PB8 / PF0 / PB5 / PB4 / PB3 / PA15
23	Reserved
24	GPIO port not match Wakeup
25	USART0 Wakeup

2.6. Register definition

EXTI base address: 0x4001 0400

2.6.1. Interrupt enable register (EXTI_INTEN)

Address offset: 0x00

Reset value: 0x0E80 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						INTEN25	INTEN24	Reserved	INTEN22	INTEN21	INTEN20	INTEN19	INTEN18	Reserved	INTEN16
						rw	rw		rw	rw	rw	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
31:26	Reserved	Must be kept at reset value
25:24	INTENx	Interrupt enable bit x(x=24..25) 0: Interrupt from linex is disabled 1: Interrupt from linex is enabled
23	Reserved	Must be kept at reset value
22:18	INTENx	Interrupt enable bit x(x=18..22) 0: Interrupt from linex is disabled 1: Interrupt from linex is enabled
17	Reserved	Must be kept at reset value
16:0	INTENx	Interrupt enable bit x(x=0..16) 0: Interrupt from linex is disabled 1: Interrupt from linex is enabled

2.6.2. 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						EVEN25	EVEN24	Reserved	EVEN22	EVEN21	EVEN20	EVEN19	EVEN18	Reserved	EVEN16
						rw	rw		rw	rw	rw	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
31:26	Reserved	Must be kept at reset value
25:24	EVENx	Event enable bit x(x=24..25) 0: Event from linex is disabled 1: Event from linex is enabled
23	Reserved	Must be kept at reset value
22:18	EVENx	Event enable bit x(x=22..18) 0: Event from linex is disabled 1: Event from linex is enabled
17	Reserved	Must be kept at reset value
16:0	EVENx	Event enable bit x(x=0..16) 0: Event from linex is disabled 1: Event from linex is enabled

2.6.3. 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						RTEN25	RTEN24	Reserved	RTEN22	RTEN21	RTEN20	RTEN19	RTEN18	Reserved	RTEN16
						rw	rw		rw	rw	rw	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
31:26	Reserved	Must be kept at reset value
25:24	RTENx	Rising edge trigger enable (x=24..25) 0: Rising edge of Linex is invalid 1: Rising edge of Linex is valid as an interrupt/event request
23	Reserved	Must be kept at reset value
22:18	RTENx	Rising edge trigger enable (x=18..22) 0: Rising edge of Linex is invalid 1: Rising edge of Linex is valid as an interrupt/event request

17	Reserved	Must be kept at reset value
16:0	RTENx	Rising edge trigger enable (x=0..16) 0: Rising edge of Linex is invalid 1: Rising edge of Linex is valid as an interrupt/event request

2.6.4. 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						FTEN25	FTEN24	Reserved	FTEN22	FTEN21	FTEN20	FTEN19	FTEN18	Reserved	FTEN16
						rw	rw		rw	rw	rw	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
31:26	Reserved	Must be kept at reset value
25:24	FTENx	Falling edge trigger enable (x=24..25) 0: Falling edge of Linex is invalid 1: Falling edge of Linex is valid as an interrupt/event request
23	Reserved	Must be kept at reset value
22:18	FTENx	Falling edge trigger enable (x=18..22) 0: Falling edge of Linex is invalid 1: Falling edge of Linex is valid as an interrupt/event request
17	Reserved	Must be kept at reset value
16:0	FTENx	Falling edge trigger enable (x=0..16) 0: Falling edge of Linex is invalid 1: Falling edge of Linex is valid as an interrupt/event request

2.6.5. 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						SWIEV25	SWIEV24	Reserved	SWIEV22	SWIEV21	SWIEV20	SWIEV19	SWIEV18	Reserved	SWIEV16
						rw	rw		rw	rw	rw	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
31:26	Reserved	Must be kept at reset value
25:24	SWIEVx	Interrupt/Event software trigger (x=24..25) 0: Deactivate the EXTIx software interrupt/event request 1: Activate the EXTIx software interrupt/event request
23	Reserved	Must be kept at reset value
22:18	SWIEVx	Interrupt/Event software trigger (x=18..22) 0: Deactivate the EXTIx software interrupt/event request 1: Activate the EXTIx software interrupt/event request
17	Reserved	Must be kept at reset value
16:0	SWIEVx	Interrupt/Event software trigger (x=0..16) 0: Deactivate the EXTIx software interrupt/event request 1: Activate the EXTIx software interrupt/event request

2.6.6. 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						PD25	PD24	Reserved	PD22	PD21	PD20	PD19	PD18	Reserved	PD16
						rc_w1	rc_w1		rc_w1	rc_w1	rc_w1	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
31:26	Reserved	Must be kept at reset value
25:24	PDx	Interrupt pending status (x=24..25) 0: EXTI Linex is not triggered 1: EXTI Linex is triggered. This bit is cleared to 0 by writing 1 to it.
23	Reserved	Must be kept at reset value
22:18	PDx	Interrupt pending status (x=18..22) 0: EXTI Linex is not triggered

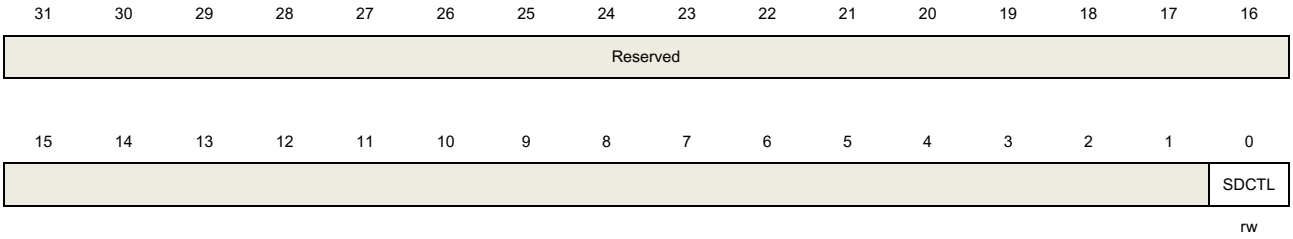
		1: EXTI Linex is triggered. This bit is cleared to 0 by writing 1 to it.
17	Reserved	Must be kept at reset value
16:0	PDx	Interrupt pending status (x=0..16) 0: EXTI Linex is not triggered 1: EXTI Linex is triggered. This bit is cleared to 0 by writing 1 to it.

2.6.7. Source default control register (EXTI_CTL)

Address offset: 0x18

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	SDCTL	0: EXTI source default is the last group pin 1: EXTI source default is invalid

3. Direct memory access controller (DMA)

3.1. 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 7 channels in the DMA controller. 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.

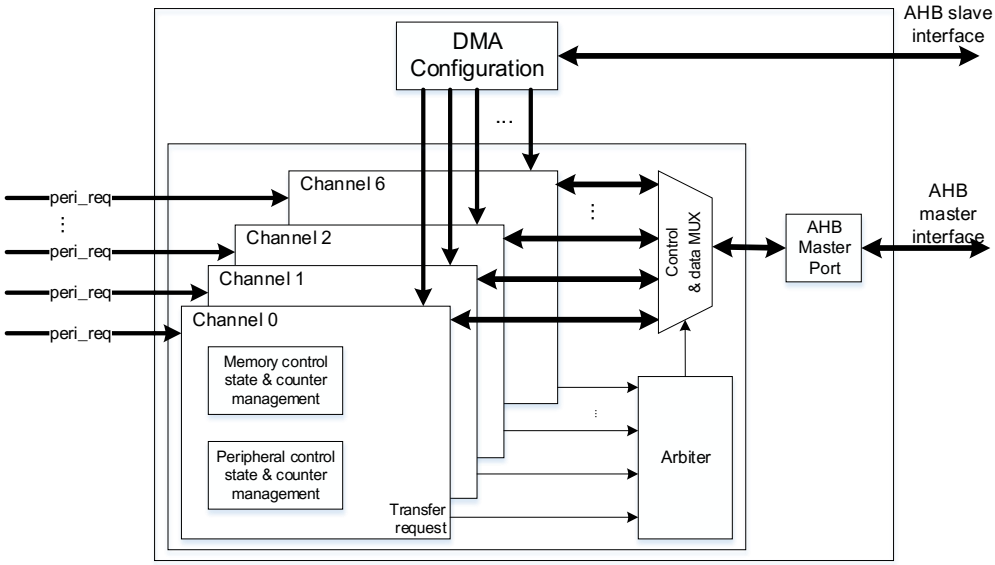
The system bus is shared by the DMA controller and the Cortex®-M23 core. When the DMA and the CPU are targeting the same destination, the DMA access may stop the CPU access to the system bus for some bus cycles. Round-robin scheduling is implemented in the bus matrix to ensure at least half of the system bus bandwidth for the CPU.

3.2. Characteristics

- Programmable length of data to be transferred, max to 65535.
- 7 channels and each channel are configurable.
- AHB and APB peripherals, FLASH, SRAM can be accessed as source and destination
- Each channel is connected to 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.

3.3. Block diagram

Figure 3-1. Block diagram of DMA



As shown in ***Figure 3-1. 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.

3.4. Function overview

3.4.1. 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 the following ***Table 3-1. DMA transfer operation***.

Table 3-1. 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[15:0] @0x0 2: Write B5B4[15:0] @0x2 3: Write B9B8[15:0] @0x4 4: Write BDBC[15:0] @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 enable the CHEN bit in the register. During the transmission, the CNT bits indicate the remaining number of data items to be transferred.

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

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

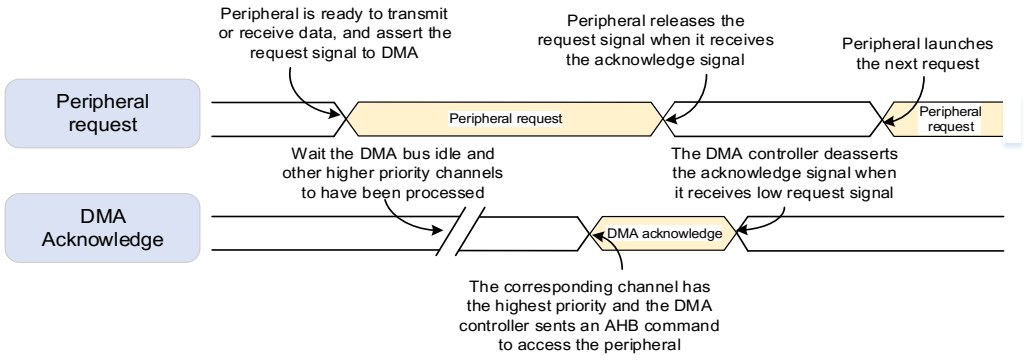
3.4.2. Peripheral handshake

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 3-2. Handshake mechanism shows how the handshake mechanism works between the DMA controller and peripherals.

Figure 3-2. Handshake mechanism



3.4.3. Arbitration

When two or more requests are received at the same time, the arbiter determines which request is served based on the priorities of 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.

3.4.4. 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.

3.4.5. 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.

3.4.6. 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.

3.4.7. 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.

3.4.8. 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.

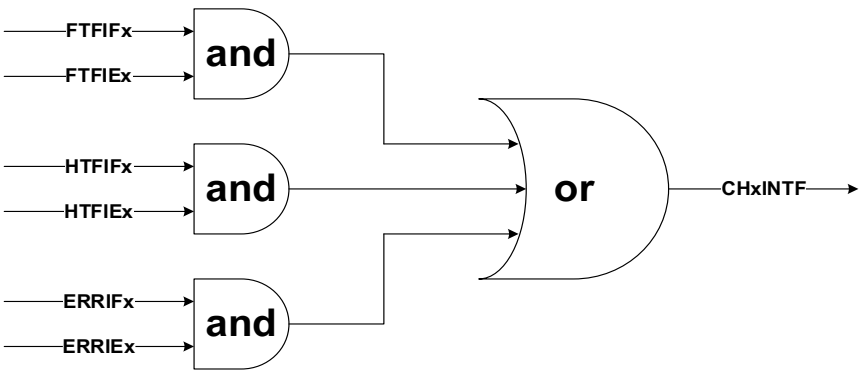
Each interrupt event has a dedicated flag bit in the DMA_INTF register, a dedicated clear bit in the DMA_INTC register, and a dedicated enable bit in the DMA_CHxCTL register. The relationship is described in the following ***Table 3-2. interrupt events***.

Table 3-2. 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 the ***Figure 3-3. DMA interrupt logic***, an interrupt can be produced when any type of interrupt event occurs and enabled on the channel.

Figure 3-3. DMA interrupt logic



Note: "x" indicates channel number (x=0…6).

3.4.9. 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 the following ***Figure 3-4. DMA 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 3-3. DMA requests for each channel*** lists the support request from peripheral for each channel of DMA.

Figure 3-4. DMA request mapping

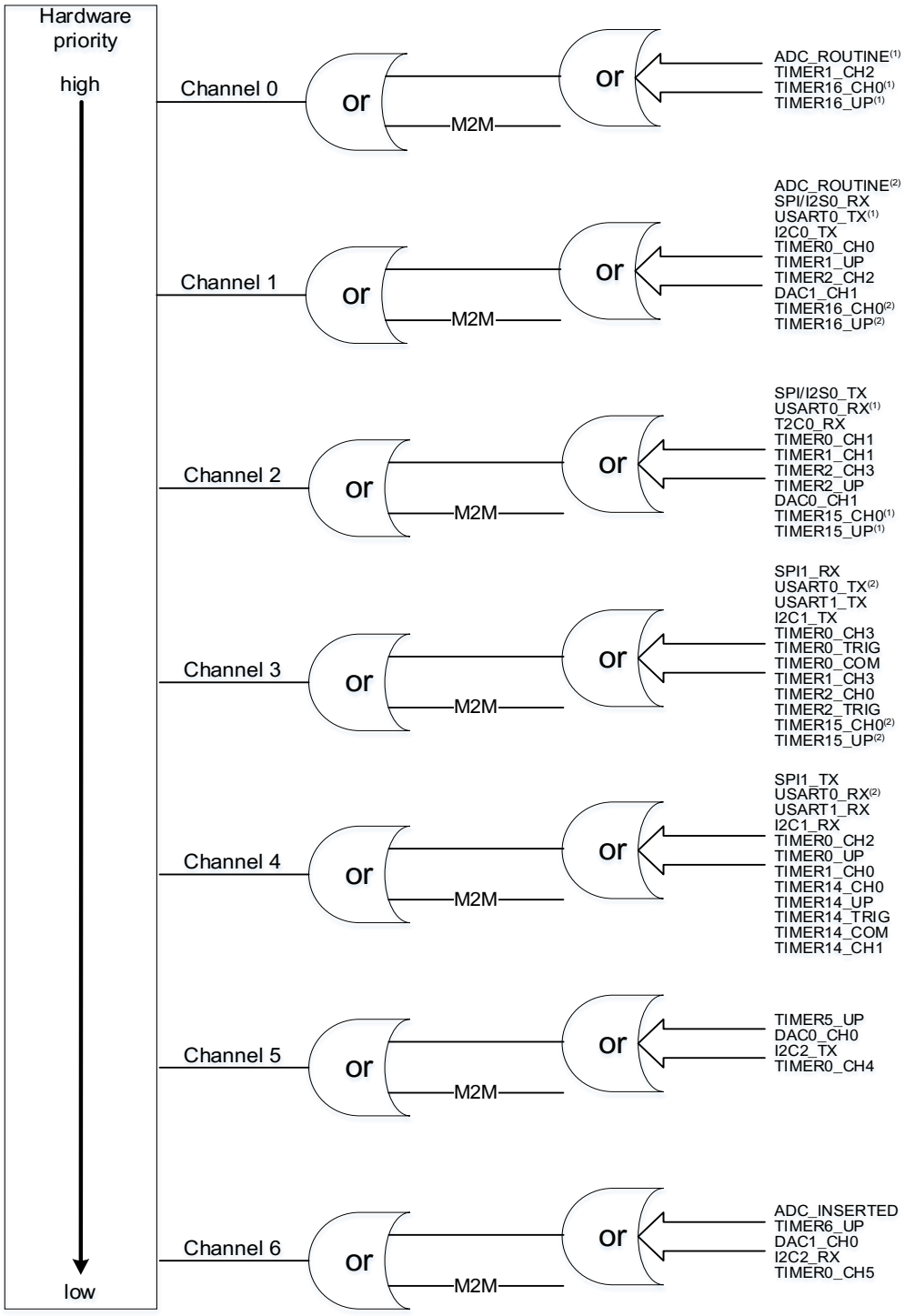


Table 3-3. DMA requests for each channel

Peripheral	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6
ADC	ADC_ROUTINE (1)	ADC_ROUTINE (2)	●	●	●	●	ADC_INSERTED
SPI / I2S	●	SPI / I2S0_RX	SPI / I2S0_TX	SPI1_RX	SPI1_TX	●	●
USART	●	USART0_TX(1)	USART0_RX(1)	USART0_TX(2) UART1_TX	USART0_RX(2) UART1_RX	●	●
I²C	●	I2C0_TX	I2C0_RX	I2C1_TX	I2C1_RX	I2C2_TX	I2C2_RX
TIMER0	●	TIMER0_CH0	TIMER0_CH1	TIMER0_CH3 TIMER0_TRIG TIMER0_COM	TIMER0_CH2 TIMER0_UP	TIMER0_CH4	TIMER0_CH5
TIMER1	TIMER1_CH2	TIMER1_UP	TIMER1_CH1	TIMER1_CH3	TIMER1_CH0	●	●
TIMER2	●	TIMER2_CH2	TIMER2_CH3 TIMER2_UP	TIMER2_CH0 TIMER2_TRIG	●	●	●
TIMER5	●	●	●	●	●	TIMER5_UP	●
TIMER6	●	●	●	●	●	●	TIMER6_UP
DAC0	●	●	DAC0_CH1	●	●	DAC0_CH0	●
DAC1	●	DAC1_CH1	●	●	●	●	DAC1_CH0
TIMER14	●	●	●	●	TIMER14_CH0 TIMER14_UP TIMER14_TRIG TIMER14_COM TIMER14_CH1	●	●
TIMER15	●	●	TIMER15_CH0(1) TIMER15_UP(1)	TIMER15_CH0(2) TIMER15_UP(2)	●	●	●
TIMER16	TIMER16_CH0(1) TIMER16_UP(1)	TIMER16_CH0(2) TIMER16_UP(2)	●	●	●	●	●

(1) When the corresponding remapping bit in the SYSCFG_CFG0 register is cleared, the request is mapped on the channel.

(2) When the corresponding remapping bit in the SYSCFG_CFG0 register is set, the request is mapped on the channel.

3.5. Register definition

DMA base address: 0x4002 0000

3.5.1. Interrupt flag register (DMA_INTF)

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				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
31:20	Reserved	Must be kept at reset value
27/23/19/15 /11/7/3	ERRIFx	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	HTFIFx	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	FTFIFx	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	GIFx	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

3.5.2. Interrupt flag clear register (DMA_INTC)

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				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
31:20	Reserved	Must be kept at reset value.
27/23/19/15 /11/7/3	ERRIFCx	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	HTFIFCx	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	FTFIFCx	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	GIFCx	Clear global interrupt flag of channel x (x=0…6) 0: No effect 1: Clear GIFx, ERRIFx, HTFIFx and FTFIFx bits in the DMA_INTF register

3.5.3. Channel x control register (DMA_CHxCTL)

Address offset: 0x08 + 0x14 × x (x = 0...6, where x is a channel number)

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	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

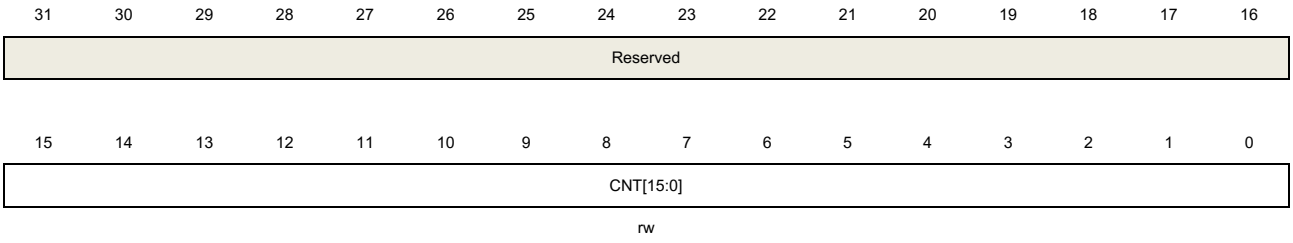
		Software set and cleared 0: Read from peripheral and write to memory 1: Read from memory and write to peripheral This bit can not be written when CHEN is '1'.
3	ERRIE	Enable bit for channel error interrupt Software set and cleared 0: Disable the channel error interrupt 1: Enable the channel error interrupt
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

3.5.4. Channel x counter register (DMA_CHxCNT)

Address offset: 0x0C + 0x14 × x, (x = 0...6, where x is a channel number)

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	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 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.

3.5.5. Channel x peripheral base address register (DMA_CHxPADDR)

Address offset: 0x10 + 0x14 × x, (x = 0...6, where x is a channel number)
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
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.

3.5.6. Channel x memory base address register (DMA_CHxMADDR)

Address offset: 0x14 + 0x14 × x, (x = 0...6, where x is a channel number)
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
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.

4. Flash memory controller (FMC)

4.1. 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 in the 64K bytes of the flash. It also provides page erase, mass erase, and double word program for flash memory.

4.2. Characteristics

- Up to 64 KB of on-chip flash memory for storing instruction/data.
- 0~2 waiting time within 64 KB when CPU executes instruction.
- Flash memory supports 64-bit wide date reading.
- Pre-fetch buffer to speed read operations.
- 3K bytes information block for bootloader.
- 16 bytes option bytes block for user application requirements.
- 1 KB OTP (One-time program) block used for user data storage.
- 1K bytes page size.
- ECC with single bit error corrected and double bit errors detected.
- Double word programming, page erase and mass erase capability.
- Flash read protection to prevent illegal code/data access.
- Page erase/program protection to prevent unexpected operation.

4.3. Function overview

4.3.1. Flash memory architecture

The flash memory consists of up to 64 KB main flash organized into 64 pages with 1 KB capacity per page and a 3 KB Information Block for the Bootloader. The main flash memory contains a total of up to 64 pages which can be erased individually. The *Table 4-1. Base address and size for flash memory* shows the base address and size.

Table 4-1. Base address and size for flash memory

Block	Name	Address	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
	. .	. .	. .

Block	Name	Address	size(bytes)
	.	.	.
	Page 63	0x0800 FC00 - 0x0800 FFFF	1KB
Information block	Bootloader	0x1FFF EC00 - 0x1FFF F7FF	3KB
Option byte block	Option byte	0x1FFF F800 - 0x1FFF F80F	16B
One-time program block	OTP bytes	0x1FFF 7000 - 0x1FFF 73FF	1KB

Note: The Information block stores the bootloader, this block cannot be programmed or erased by user.

4.3.2. Error checking and correcting (ECC)

The ECC mechanism supports:

- One bit error detection and correction
- Two bits errors detection

When one error is detected and corrected:

- When ECC error occurs, the ECCCOR bit in FMC_ECCCS register will be set. If the ECCCORIE bit in FMC_ECCCS register is set, an interrupt is generated. The ECCADDR bits notice the ECC error address in each space. The ECCCOR bit in FMC_ECCCS register can be cleared by writing 1 to ECCCOR bit.

When two errors are detected:

- When ECC error occurs, the ECCDET bit in FMC_ECCCS register will be set. If the ECCDETIE bit in FMC_ECCCS register is set, an NMI interrupt is generated. The ECCADDR bits notice the ECC error address in each space. And the data return all F. The ECCDET bit in FMC_ECCCS register can be cleared by writing 1 to ECCDET bit.

Note:

1. Data in Flash memory are 72-bits words: 8 bits are added per double words (64 bits), which the added 8 bits are the ECC code and calculated by hardware and cannot be accessed by user.
2. For a virgin data 0xFF FFFF FFFF FFFF FFFF, ECC is not supported.
3. When a new ECC error occurs, ECCADDR will be updated to the latest error address.
4. When an ECC error is reported, a new read at the error address may not generate an ECC error if the data is still present in the current buffer / prefetch buffer, even if ECCCOR and ECCDET bits are cleared.
5. Prefetching buffer reading data or instructions can also generate ECC errors, even if the data or instructions are not being used by CPU. To correctly display the value of the ECC error address register, it is necessary to disable the prefetch function; otherwise, the displayed error address may be rolled forward due to prefetching.

6. When the original data programmed is all F, the ECC calculation will bypass this operation. Erasing is not required, and programming other values again can be written normally.

4.3.3. 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 AHB BUS from the CPU.

Wait state added

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 shown in ***Table 4-2. The relation between WSCNT and AHB clock frequency***.

Table 4-2. The relation between WSCNT and AHB clock frequency

AHB clock frequency	WSCNT configured
<= 24MHz	0 (0 wait state added)
<= 48MHz	1 (1 wait state added)
<= 72MHz	2 (2 wait state added)

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

Note:

1. If it is needed to increase the AHB clock frequency. First, refer to ***Table 4-2. The relation between WSCNT and AHB clock frequency***, 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 configuring the WSCNT.

2. If it is needed to decrease the AHB clock frequency. First, decrease the target AHB clock frequency. Then refer to ***Table 4-2. The relation between WSCNT and AHB clock frequency***, 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 2 wait state when 72MHz). 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 will be 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 setting the PFEN bit in the FMC_WS register. In the case

of sequential code, when CPU executes 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 finishes the current buffer and needs execute the next data, the pre-fetch buffer hits.

4.3.4. Unlock the FMC_CTL register

After reset, the FMC_CTL register is not accessible in write mode, except for the OBRLD bit, which is used for reloading the option byte, and the LK bit in FMC_CTL register is 1. An unlocking sequence consists of two write operations to the FMC_KEY register can 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 FMC_CTL register is set to 0 by hardware. The software can lock the FMC_CTL again by setting the LK bit in FMC_CTL register to 1. If there is any wrong operations on the FMC_KEY register, the LK bit in FMC_CTL register will be set, and the FMC_CTL register will be locked, then it will generate a bus error.

The OBPG bit and OBER bit in FMC_CTL are also protected by FMC_OBKEY register. The unlocking sequence includes two write operations, which are writing 0x45670123 and 0xCDEF89AB to FMC_OBKEY register. And then set the OBWEN bit in FMC_CTL register to 1. The software can set OBWEN bit to 0 to protect the OBPG bit and OBER bit in FMC_CTL register again.

4.3.5. Page erase

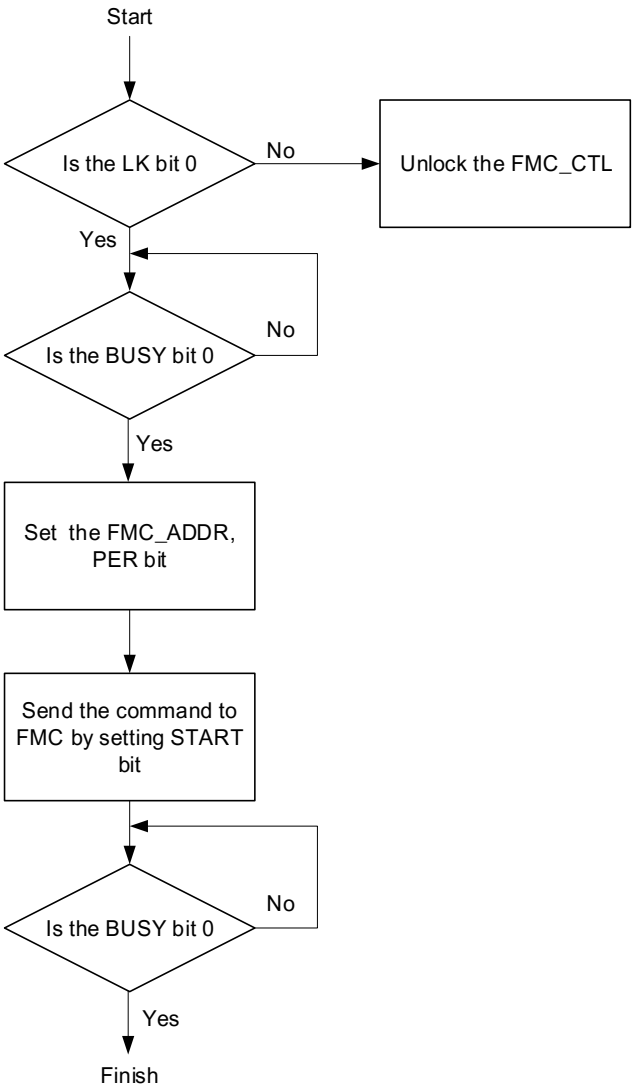
The FMC provides a page erase function which is used for initializing the contents of a main flash memory page to a high state. Each page can be erased independently without affecting the contents of other pages. The following steps show the access sequence of the register for a page erase operation.

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

When the operation is executed successfully, an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set, and the ENDF in FMC_STAT register is set. If the target erasing page is being used for fetching codes or accessing data, the software may run out of control. The FMC will not provide any notification when this occurs. Additionally, the

page erase operation will be ignored on protected pages. 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 in the FMC_STAT register to detect this condition in the interrupt handler. The end of this operation is indicated by the ENDF bit in the FMC_STAT register. The *Figure 4-1. Process of page erase operation* shows the page erase operation flow.

Figure 4-1. Process of page erase operation



4.3.6. Mass erase

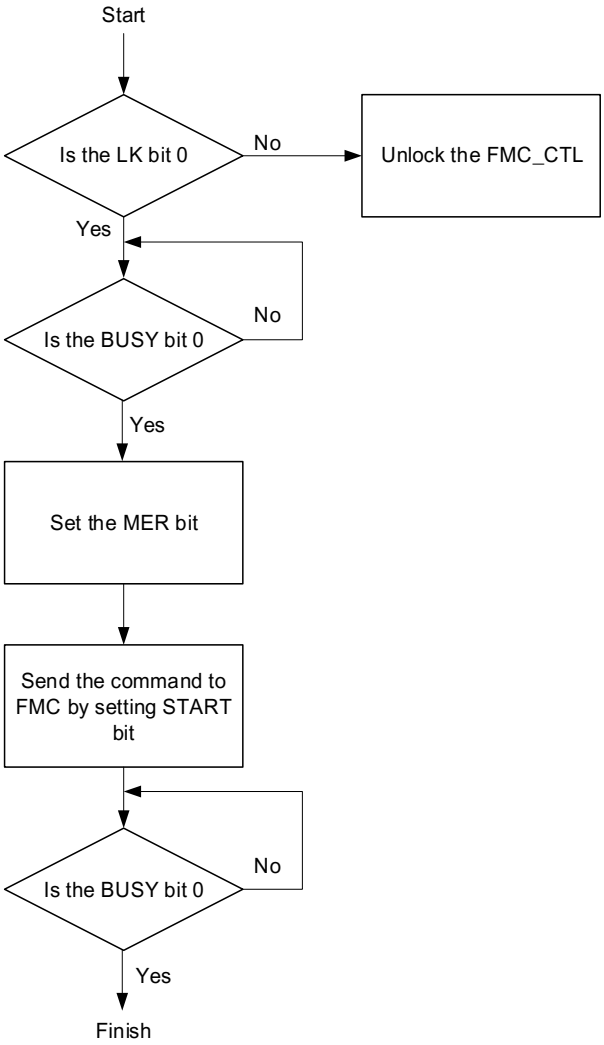
The FMC provides a complete erase function which is used for initializing the main flash block contents. The following steps show the mass erase register access sequence.

- Unlock the LK bit in FMC_CTL register if necessary.
- Check the BUSY bit in FMC_STAT register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.

- Write the mass erase command by setting MER bit in FMC_CTL register.
- Send the mass erase command to the FMC by setting the START bit in FMC_CTL register.
- Wait until all the operations have been completed wihich is judged by checking the value of the BUSY bit in FMC_STAT register.
- Read and verify the flash memory by using a DBUS access if necessary.

When the operation is executed successfully, an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set, and the ENDF in FMC_STAT register is set. Since all flash data will be reset to a value of 0xFFFF FFFF, the mass erase operation can be implemented using a program that runs in SRAM or by using the debugging tool to access the FMC registers directly. The end of this operation is indicated by the ENDF bit in the FMC_STAT register. (The starting address of programming operation should be 0x0800 0000) The ***Figure 4-2. Process of the mass erase operation*** indicates the mass erase operation flow.

Figure 4-2. Process of the mass erase operation



4.3.7. Main flash programming

The FMC provides a 64-bit double words programming function by DBUS which is used to modify the main flash memory contents.

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

- Unlock the LK bit in FMC_CTL register if necessary.
- Check the BUSY bit in FMC_STAT register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.
- Write the program command by setting the PG bit in FMC_CTL register.
- Write the 64-bit data to be programmed by DBUS with desired absolute address (0x08XX XXXX).
- Wait until all the operations have been completed wihich is judged by checking the value of the BUSY bit in FMC_STAT register.
- Read and verify the flash memory by using a DBUS access if necessary.

When the operation is executed successfully, an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set, and the ENDF in FMC_STAT register is set. Note that it is necessary to check the address has been erased before the double word programming operation. If the address has not been erased, PGERR bit will set when programming the address even if programming 0x0. Additionally, the program operation will be ignored on protected pages, and WPERR bit will be set. Then 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 and WPERR bit in the FMC_STAT register to detect this condition in the interrupt handler. The end of this operation is indicated by the ENDF bit in the FMC_STAT register.

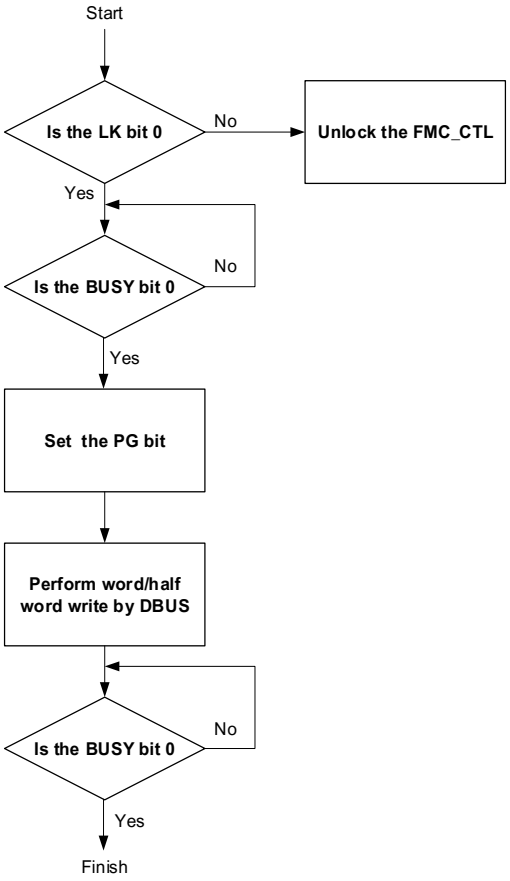
In the following cases, the PGAERR bit in the FMC_STAT register will be set.

- The DBUS program use 8-bit write.
- Mix 32-bit with 16-bit write.
- The DBUS write is not aligned. If DBUS program is 32-bit, the second DBUS write must be double-word aligned and belong to same double-word address. If DBUS program is 16-bit, the 2nd/3rd/4th DBUS write must be double-word aligned and belong to same double-word address.

Note: If the program is not write total 64 bits, the data is not programmed to the flash memory without any abnormal tips.

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 *Figure 4-3. Process of the double word programming operation* displays the word programming operation flow.

Figure 4-3. Process of the double word programming operation



4.3.8. OTP programming

The OTP programming method is same as the main flash programming. The OTP block can only be programmed 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 to corrupt the data stored in flash memory.

4.3.9. Option byte erase

The FMC provides an erase function which is used for initializing the option byte block in flash. The following steps show the erase sequence.

- Unlock the LK bit in FMC_CTL register if necessary.
- Unlock the OBWEN bit in FMC_CTL register if necessary.
- Check the BUSY bit in FMC_STAT register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.
- Write the option byte erase command by setting OBER bit in FMC_CTL register.

- Send the option byte erase command to the FMC by setting the START bit in FMC_CTL register.
- Wait until all the operations have been completed wihich is judged by checking the value of the BUSY bit in FMC_STAT register.
- Read and verify the flash memory by using a DBUS access if necessary.

When the operation is executed successfully, an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set, and the ENDF in FMC_STAT register is set. The end of this operation is indicated by the ENDF bit in the FMC_STAT register.

4.3.10. Option byte programming

The FMC provides a 64-bit double word programming function which is used for modifying the option byte block contents. The following steps show the programming operation sequence.

- Unlock the LK bit in FMC_CTL register if necessary.
- Unlock the OBWEN bit in FMC_CTL register if necessary.
- Check the BUSY bit in FMC_STAT register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.
- Write the program command by setting the OBPG bit in FMC_CTL register.
- A 64-bit double word write at desired address.
- Wait until all the operations have been completed wihich is judged by checking the value of the BUSY bit in FMC_STAT register.
- Read and verify the flash memory by using a DBUS access if necessary.

When the operation is executed successfully, an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set, and the ENDF in FMC_STAT register is set. Note that it is necessary to check the address has been erased before the double word programming operation. If the address has not been erased, PGERR bit will set when programming the address even if programming 0x0. The end of this operation is indicated by the ENDF bit in the FMC_STAT register.

4.3.11. Option byte description

The option bytes block of flash memory reloaded to FMC_OBSTAT and FMC_WP registers after each system reset or OBRLD bit set in FMC_CTL register, and the option bytes have been effected. The option complement bytes are the opposite of option bytes. When option bytes reload, if the option complement bytes and option bytes does not match, the OBERR bit in FMC_OBSTAT register is set, and the option byte is set to 0xFF. The ***Table 4-3. Option byte*** is the detail of option bytes.

Table 4-3. Option byte

Address	Name	Factory value	Description
0x1fff f800	OB_SPC	0xA5	option byte Security Protection Code

Address	Name	Factory value	Description
			0xA5: No protection any value except 0xA5 and 0xCC: low level protection 0xCC: high level protection
0x1fff f801	OB_SPC_N	0x5A	OB_SPC complement value
0x1fff f802	OB_USER	For GD32E252xx series: 0xFF For GD32E251xx series: 0xF7	option byte which user defined [7]: NRST_MDSEL For GD32E252xx series: 0: PB11 pin as GPIO 1: PB11 pin as reset pin For GD32E251xx series: 0: PB11 pin as reset pin 1: PB11 pin as GPIO [6]: SRAM_PARITY_ECC_CHECK 0: Enable SRAM parity check, disable SRAM ECC 1: Disable SRAM parity check, enable SRAM ECC **Note:** To prevent data reading errors after the SRAM ECC function is enabled from disable state, the SRAM ECC function takes effect only after the Power on reset. If the SRAM ECC function is enabled, the SRAM parity check function is disabled. [5]: VDDA_VISOR 0: Disable V_{DDA} monitor 1: Enable V_{DDA} monitor [4]: BOOT1_n 0: BOOT1 bit is 1 1: BOOT1 bit is 0 [3]: BOOT0_n 0: BOOT0 bit is 1 1: BOOT0 bit is 0 Note: This bit only applies to GD32E251xx. [2]: nRST_STDBY 0: Generate a reset instead of entering standby mode 1: No reset when entering standby mode [1]: nRST_DPSLP 0: Generate a reset instead of entering Deep-sleep mode 1: No reset when entering Deep-sleep

Address	Name	Factory value	Description
			mode [0]: nWDG_SW 0: Hardware free watchdog timer 1: Software free watchdog timer
0x1fff f803	OB_USER_N	For GD32E252xx series: 0x00 For GD32E251xx series: 0x08	OB_USER complement value
0x1fff f804	OB_DATA[7:0]	0xFF	user defined data bit 7 to 0
0x1fff f805	OB_DATA_N[7:0]	0x00	OB_DATA complement value bit 7 to 0
0x1fff f806	OB_DATA[15:8]	0xFF	user defined data bit 15 to 8
0x1fff f807	OB_DATA_N[15:8]	0x00	OB_DATA complement value bit 15 to 8
0x1fff f808	OB_WP[7:0]	0xFF	Page Erase/Program Protection bit 7 to 0
0x1fff f809	OB_WP_N[7:0]	0x00	OB_WP complement value bit 7 to 0
0x1fff f80a	OB_WP[15:8]	0xFF	Page Erase/Program Protection bit 15 to 8
0x1fff f80b	OB_WP_N[15:8]	0x00	OB_WP complement value bit 15 to 8

4.3.12. 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 then be set by the FMC. If the WPERR bit is set and the ERRIE bit is set, then the flash operation error interrupt will be triggered by the FMC. The page protection function can be individually enabled by configuring the OB_WP[15:0] bit field to 0 in the option byte. If a page erase operation is executed on the Option Byte region, all the flash memory page protection functions will be disabled. After setting or resetting OB_WP in the option byte, the software need to set OBRLD in FMC_CTL register or a system reset to reload the OB_WP bits in FMC_WP register. The ***Table 4-4. OB_WP bit for pages protected*** shows which pages are protected by set OB_WP[15:0].

Table 4-4. OB_WP bit for pages protected

OB_WP bit	pages protected
OB_WP[0]	page 0 ~ page 3
OB_WP[1]	page 4 ~ page 7
OB_WP[2]	page 8 ~ page 11
.	.
.	.
.	.

OB_WP bit	pages protected
OB_WP[14]	page 56 ~ page 59
OB_WP[15]	page 60 ~ page 63

4.3.13. Security protection

The FMC provides a security protection function to prevent illegal code/data access on the flash memory. This function is useful for protecting the software/firmware from illegal user operations. There are 3 levels for security protection:

No protection: when setting OB_SPC byte and its complement value to 0x5AA5, no protection perform. The main flash and option bytes block are accessible by all operations.

Low level protection: when setting OB_SPC byte and its complement value to any value except 0x5AA5 and 0x33CC, low level protection perform. The main flash can only be accessed by user code. In debug mode, boot from SRAM or boot from bootloader mode, all operations to main flash is forbidden. If a read operation is executed to main flash in debug mode, boot from SRAM or boot from bootloader mode, a bus error will be generated. If a program/erase operation is executed to main flash in debug mode, boot from SRAM or boot from bootloader mode, the PGERR bit in FMC_STAT register will be set. At low level protection, option bytes block are accessible by all operations. If program back to no protection level by setting OB_SPC byte and its complement value to 0x5AA5, a mass erase for main flash will be performed.

High level protection: when set OB_SPC byte and its complement value to 0x33CC, high level protection perform. When this level is programmed in debug mode, boot from SRAM or boot from bootloader mode is disabled. The main flash block is accessible by all operations from user code. The option byte cannot be erased, and the OB_SPC byte and its complement value cannot be reprogrammed. So, if high level protection is programmed, it cannot move back to low level protection or no protection level.

Note: In the case of read protection, the first 4KB of flash cannot be programmed or erased.

4.4. Register definition

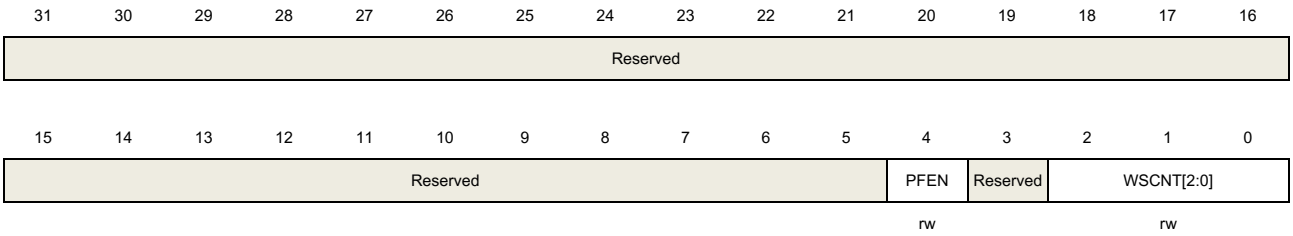
FMC base address: 0x4002 2000

4.4.1. Wait state register (FMC_WS)

Address offset: 0x00

Reset value: 0x0000 0030

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



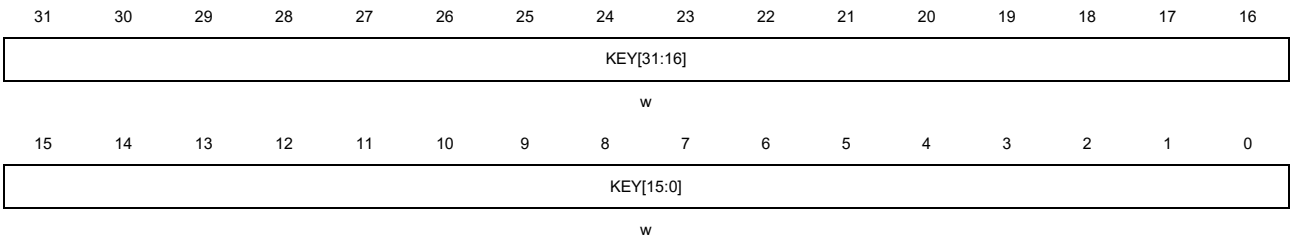
Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	PFEN	Pre-fetch enable 0: Pre-fetch disable 1: Pre-fetch enable
3	Reserved	Must be kept at reset value.
2:0	WSCNT[2:0]	Wait state counter register These bits set and reset by software. 000: 0 wait state added 001: 1 wait state added 010: 2 wait state added 011 ~ 111: Reserved

4.4.2. Unlock key register (FMC_KEY)

Address offset: 0x04

Reset value: 0x0000 0000

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



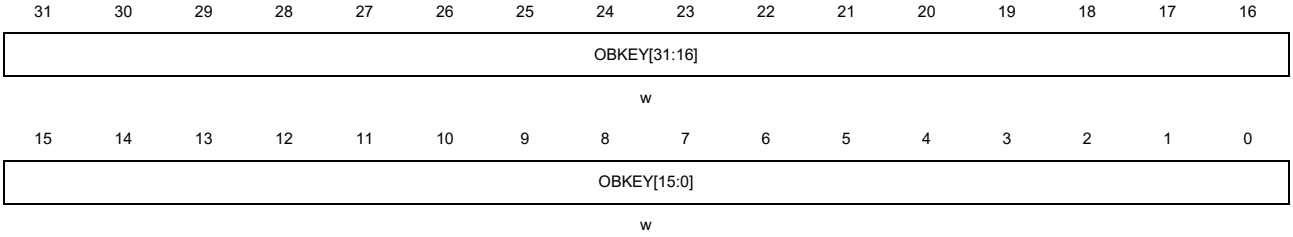
Bits	Fields	Descriptions
31:0	KEY[31:0]	FMC_CTL unlock registers These bits are only be written by software Write KEY[31:0] with key to unlock FMC_CTL register.

4.4.3. Option byte unlock key register (FMC_OBKEY)

Address offset: 0x08

Reset value: 0x0000 0000

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



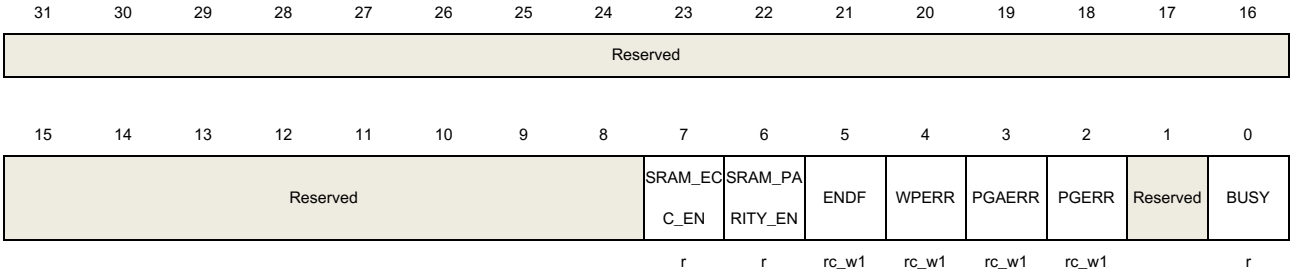
Bits	Fields	Descriptions
31:0	OBKEY[31:0]	FMC_CTL option byte operation unlock registers These bits are only be written by software Write OBKEY[31:0] with key to unlock option byte command in FMC_CTL register.

4.4.4. Status register (FMC_STAT)

Address offset: 0x0C

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	SRAM_ECC_EN	SRAM ECC enable bit Current SRAM ECC enable status When SRAM ECC is enabled, this bit is 1, otherwise it is 0.

6	SRAM_PARITY_EN	SRAM parity check enable bit Current SRAM parity checking enable status When SRAM parity checking is enabled, this bit is 1, otherwise it is 0.
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 erasing/programming 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 programming 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 busy bit When the operation is in progress, this bit is set to 1. When the operation is end or an error generated, this bit is clear to 0.

4.4.5. Control register (FMC_CTL)

Address offset: 0x10

Reset value: 0x0000 0080, NRST_MDSEL_REQ and NRST_MDSEL_VALUE are reset by power reset only, other bits are reset by system reset.

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														NRST_MDSEL_REQ	NRST_MDSEL_VALUE
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RPCD	RPCS	OBRLD	ENDIE	Reserved	ERRIE	OBWEN	Reserved	LK	START	OBER	OBPG	Reserved	MER	PER	PG
rw	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	NRST_MDSEL_REQ	This bit indicates wether NRST_MDSEL is controlled by register 0: Not controlled by the register

		1: Controlled by the register, NRST_MDSEL_VALUE takes effect
16	NRST_MDSEL_VALUE	PB11's value when NRST_MDSEL_REQ is set to 1. 0: PB11 pin as GPIO 1: PB11 pin as NRST reset pin
15	RPCD	RAM parity check disable 0: RAM parity check is enabled when RPCS is 1 1: RAM parity check is disabled when RPCS is 1 **Note:** This bit is valid only when SRAM_PARITY_ECC_CHECK is 0
14	RPCS	RAM parity check control selection 0: RAM parity check is controlled by option byte 1: RAM parity check is controlled by RPCD **Note:** This bit is valid only when SRAM_PARITY_ECC_CHECK is 0
13	OBRLD	Option byte reload bit This bit is set by software. 0: No effect 1: Force option byte reload, and generate a system reset
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 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 sequent written to 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 byte erase command bit This bit is set or cleared by software.

		0: No effect 1: Option byte erase command
4	OBPG	Option byte program command bit This bit is set or cleared by software. 0: No effect 1: Option byte 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 cleared by software. 0: No effect 1: Main flash page erase command
0	PG	Main flash page program command bit This bit is set or cleared by software. 0: No effect 1: Main flash page program command

4.4.6. Address register (FMC_ADDR)

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
ADDR[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR[15:0]															
rw															

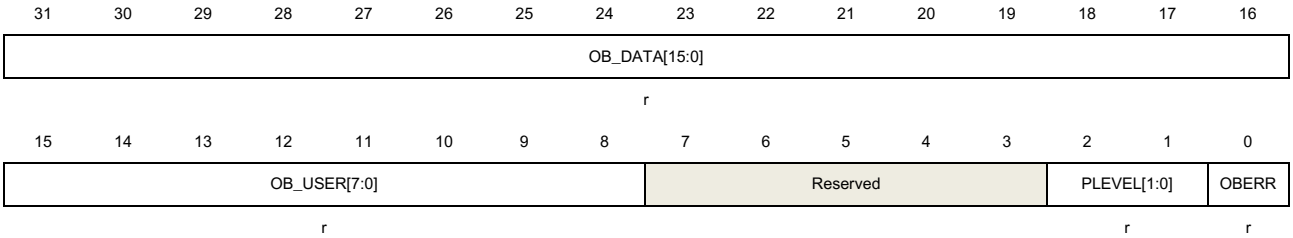
Bits	Fields	Descriptions
31:0	ADDR[31:0]	Flash command address bits These bits are set by software. ADDR bits are the address of flash erase command

4.4.7. Option byte status register (FMC_OBSTAT)

Address offset: 0x1C

Reset value: 0xXXXX XX0X

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



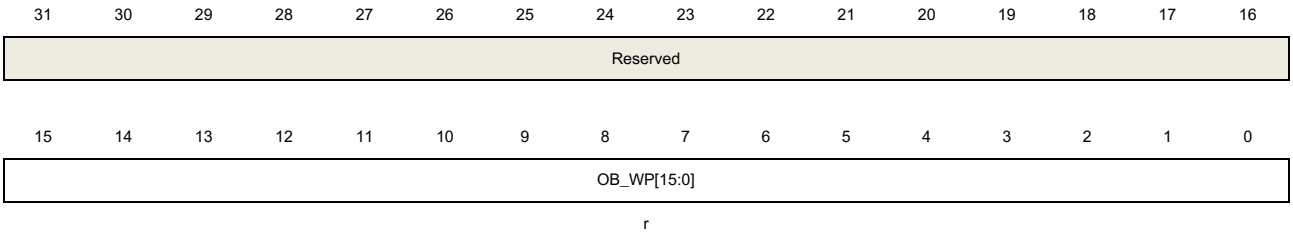
Bits	Fields	Descriptions
31:16	OB_DATA[15:0]	Store OB_DATA[15:0] of option byte block after system reset
15:8	OB_USER[7:0]	Store OB_USER byte of option byte block after system reset
7:3	Reserved	Must be kept at reset value.
2:1	PLEVEL[1:0]	Security Protection level 00: No protection level 01: Low level protection 11: High level protection
0	OBERR	Option byte read error bit. This bit is set by hardware when the option byte and its complement byte do not match, and the option byte is set to 0xFF.

4.4.8. Write protection register (FMC_WP)

Address offset: 0x20

Reset value: 0x0000 XXXX

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



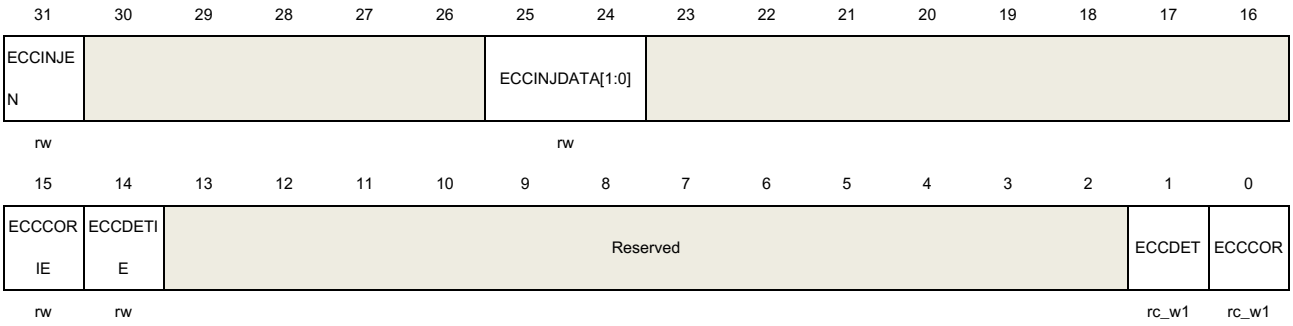
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	OB_WP[15:0]	Store OB_WP[15:0] of option byte block after system reset 0: Protection active 1: Unprotected

4.4.9. ECC control and status register (FMC_ECCCS)

Address offset: 0x30

Reset value: 0x0000 0000

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



Bits	Field	Descriptions
31	ECCINJEN	ECC fault injection enable 0: ECC fault injection disable 1: ECC fault injection enable
30:26	Reserved	Must be kept at reset value.
25:24	ECCINJDATA[1:0]	When ECCINJEN is set to 1, these bits' field can be used to control the injection of one bit or two bits error fault into the flash memory. 00: No error fault injection 01: One bit error fault injection 10: One bit error fault injection 11: Two bits error fault injection **Note:** These bit fields can only be used to inject errors into bit0 and bit1 of flash memory data.
23:16	Reserved	Must be kept at reset value.
15	ECCCORIE	One bit error correct interrupt enable. 0: One bit error correct interrupt disable. 1: One bit error correct interrupt enable.
14	ECCDETIE	Two bits errors detect interrupt enable. 0: Two bits errors detect interrupt disable. 1: Two bits errors detect interrupt enable.
13:2	Reserved	Must be kept at reset value.
1	ECCDET	Two bit errors detect flag. This bit set when two bit errors is detected. This bit is cleared by writing 1. 0: Two bits errors are not detected. 1: Two bits errors are detected.

0	ECCCOR	One bit error detected and correct flag. This bit is cleared by writing 1. 0: No bit error is detected and corrected. 1: One bit error is detected and corrected.

4.4.10. ECC address register (FMC_ECCADDR)

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
ECCADDR[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECCADDR[15:0]															
r															

Bits	Field	Descriptions
31:0	ECCADDR[31:0]	ECC error logic address (aligned to 64bit).

4.4.11. 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	Fields	Descriptions
31:0	PID[31:0]	Product ID code register These bits are read only by software. These bits are one time programmed in the process of the chip production.

5. Power management unit (PMU)

5.1. Overview

The power consumption is regarded as one of the most important issues for the devices of GD32E25x 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 the best tradeoff among the conflicting demands of CPU operating time, speed and power consumption. For GD32E25x series, there are three power domains, including V_{DD} / V_{DDA} domain, V_{CORE} domain, and Backup domain, as is shown in ***Figure 5-1. 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. Backup domain is powered from the main V_{DD} supply.

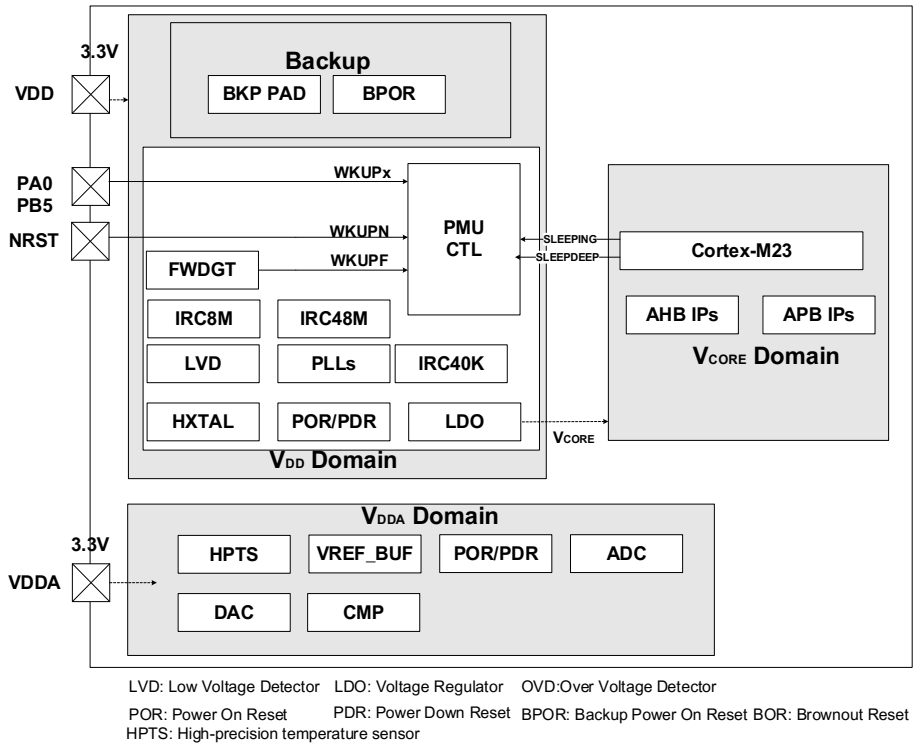
5.2. Characteristics

- Three power domains: V_{BAK}, V_{DD} / V_{DDA} and V_{CORE} power domains.
- 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 (LDO) issue an interrupt or event when the power is lower than a programmed threshold.
- LDO output voltage select for power saving.

5.3. Function overview

Figure 5-1. Power supply overview provides details on the internal configuration of the PMU and the relevant power domains.

Figure 5-1. Power supply overview



5.3.1. 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), POR / PDR (power on / down reset), FWDGT (free watchdog timer), IRC8M (Internal 8MHz RC oscillator), IRC28M (Internal 28MHz RC oscillator), IRC40K (Internal 40KHz RC oscillator), PLLs (phase locking loop), LVD (low voltage detector) etc. V_{DDA} domain includes ADC (AD converter), DAC (DA converter) , etc.

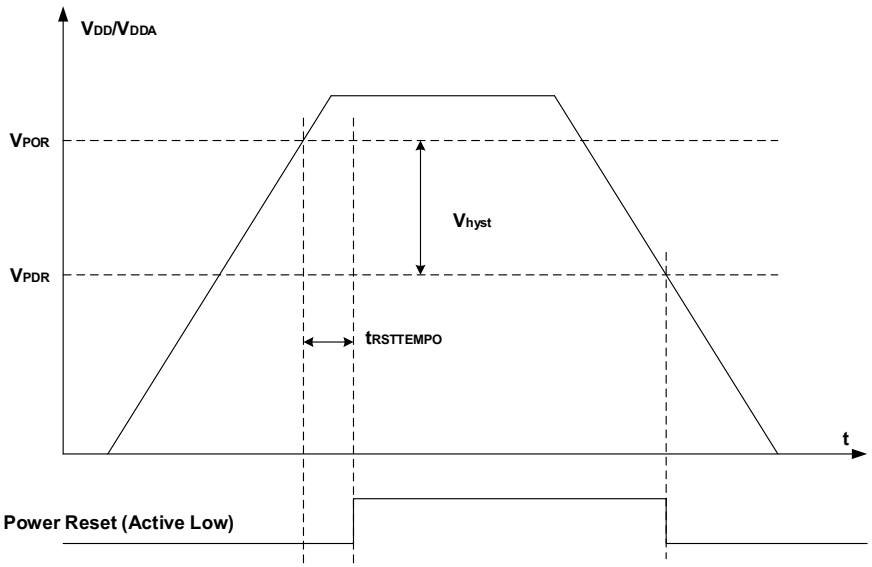
V_{DD} domain

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

The POR / PDR circuit is implemented to detect VDD / 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. The POR circuit only detects V_{DD} supply voltage, and the PDR circuit detects both VDD and VDDA supply voltages. ***Figure 5-2. Waveform of the POR/PDR*** shows the relationship between the supply voltage and the power reset signal. V_{POR}, indicates the threshold of power on reset, while V_{PDR}, means the threshold of power down reset. Refer to the datasheet for the hysteresis voltage (V_{hyst}) value.

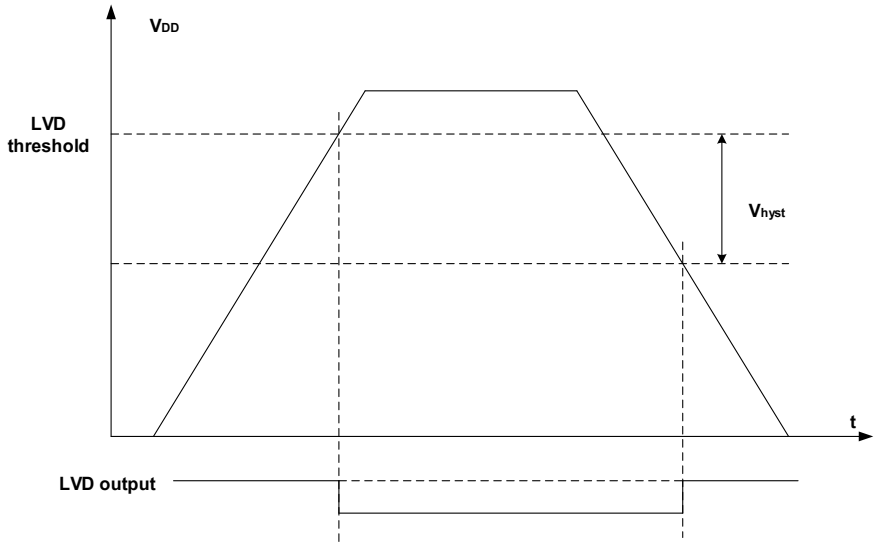
Note: V_{DDA} monitor function detected by the PDR circuit can be disabled by reset VDDA_VISOR bit in option byte register OB_USER, to reduce power consumption when users are sure that the V_{DDA} is higher than or equal to V_{DD}.

Figure 5-2. Waveform of the POR/PDR



The LVD is used to detect whether the V_{DD} 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, and LVDF bit, which in the Power status register (PMU_CS), indicates if V_{DD} 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 5-3. 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 shows the relationship between the supply voltage and the LVD signal. Refer to the datasheet for the hysteresis voltage (V_{hyst}) value..

Figure 5-3. Waveform of the LVD threshold



Backup domain

The Backup domain is powered by the VDD, the VDD supplies power for BPOR, and BREG (Backup Register).

The Backup domain reset sources include the Backup domain power-on-reset (BPOR) and the Backup Domain software reset. The Backup domain can be reset by setting the BKPRST bit in the RCU_BDCTL register.

V_{DDA} domain

Generally, digital circuits are powered by VDD, while most of analog circuits are powered by VDDA. To improve the ADC 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, if V_{DDA} is different from V_{DD}, V_{DDA} must always be higher, but the voltage difference should not exceed 0.3V.

To ensure a high accuracy on low voltage ADC and DAC, the ADC / DAC independent external reference voltage should be connected to VREFP pins. According to the different packages, VREFP pin can be connected to VDDA pin, or external reference voltage which refers to *Table 13-2. ADC input pins definition* and *Table 14-1. DAC I/O description*.

5.3.2. VCORE power domain

The main functions that include Cortex®-M23 logic, AHB / APB peripherals, the APB interfaces for the Backup domain and the V_{DD} / V_{DDA} domain, etc, are located in this power domain. Once the V_{CORE} is powered up, the POR will generate a reset sequence on the V_{CORE} power domain. If need to enter the expected 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 an expected power saving mode which will be discussed in the following section.

5.3.3. Power saving modes

After a system reset or a power reset, the GD32E25x MCU operates at full function and all power domains are active. Users can achieve lower power consumption through slowing down the system clocks (HCLK, PCLK1, PCLK2) or gating the clocks of the unused peripherals. 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®-M23. In Sleep mode, only clock of Cortex®-M23 is off. To enter the Sleep mode, it is only necessary to clear the SLEEPDEEP bit in the Cortex®-M23 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®-M23 Technical Reference Manual). The mode offers the lowest wakeup time as no time is wasted in interrupt entry or exit.

According to the SLEEPONEXIT bit in the Cortex®-M23 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 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 lowest priority ISR.

Deep-sleep mode

The Deep-sleep mode is based on the SLEEPDEEP mode of the Cortex®-M23. In Deep-sleep mode, all clocks in the V_{CORE} domain are off, and all of IRC8M, IRC28M, HXTAL and PLLs are disabled. The contents of SRAM and registers are preserved. The LDO can operate normally 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®-M23 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®-M23 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 2-3. 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 based on the SLEEPDEEP mode of the Cortex®-M23, too. In Standby mode, the whole V_{CORE} domain is powered off, the LDO is shut down, and all of IRC8M, IRC28M, HXTAL and PLL are disabled. Before entering the Standby mode, it is necessary to set the SLEEPDEEP bit in the Cortex®-M23 System Control Register, and 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 that the MCU has been in Standby mode. There are four wakeup sources for the Standby mode, including the external reset from NRST pin, the BKP tamper event, the FWDGT reset, and the rising edge on WKUP pins. 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®-M23 will execute instruction code from the 0x00000000 address.

Table 5-1. 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, IRC28M, HXTAL and PLL.	1. The V_{CORE} domain is power off. 2. Disable IRC8M, IRC28M, 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 pins 3. FWDGT reset 4. BKP
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, and WKUP pins if enabled.

5.4. PMU registers

PMU base address: 0x4000 7000

5.4.1. Control register (PMU_CTL)

Address offset: 0x00

Reset value: 0x0000 4000 (reset by 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							BKPWEN	LVDT[2:0]			LVDEN	STBRST	WURST	STBMOD	LDOLP
							rw	rw			rw	rc_w1	rc_w1	rw	rw

Bits	Fields	Descriptions
31:9	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 000：2.04V 001：2.18V 010：2.32V 011：2.46V 100：2.60V 101：2.74V 110：2.88V 111：3.02V
4	LVDEN	Low voltage detector enable 0: Disable Low Voltage Detector 1: Enable Low Voltage Detector **Note:** When LVD_LOCK bit is set to 1 in the SYSCFG_CFG2 register, LVDEN and LVDT[2:0] are read only.
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®-M23 enters SLEEPDEEP mode 1: Enter the Standby mode when the Cortex®-M23 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 **Note:** Some peripherals may work with the IRC8M clock in the Deep-sleep mode. In this case, the LDO automatically switches from the low power mode to the normal mode and remains in this mode until the peripheral stop working.

5.4.2. Control and status register (PMU_CS)

Address offset: 0x04

Reset value: 0x0000 0000 (not reset by 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		WUPEN5	Reserved				WUPEN0	Reserved					LVDF	STBF	WUF
		rw					rw						r	r	r

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13	WUPEN5	WKUP Pin5 (PB5) enable 0: Disable WKUP pin5 function 1: Enable WKUP pin5 function If WUPEN5 is set before entering the power saving mode, a rising edge on the WKUP pin5 wakes up the system from the power saving mode. As the WKUP pin5 is active high, the WKUP pin5 is internally configured to input pull down mode. And set this bit will trigger a wakeup event when the input is already high.
12:9	Reserved	Must be kept at reset value.
8	WUPEN0	WKUP Pin 0 (PA0) enable 0: Disable WKUP pin0 function 1: Enable WKUP pin0 function

		If WUPEN0 is set before entering the power saving mode, a rising edge on the WKUP pin0 wakes up the system from the power saving mode. As the WKUP pin0 is active high, the WKUP pin0 is internally configured to input pull down mode. And set 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 stopped 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 BKP wakeup event including BKP Tamper event. This bit is cleared only by a POR / PDR or by setting the WURST bit in the PMU_CTL register.

6. Backup Domain (BKP)

6.1. Overview

The backup registers located in the backup domain can be powered by V_{DD}. These backup registers are used to store and protect user application data, and they are not affected by waking up from standby mode or system resets.

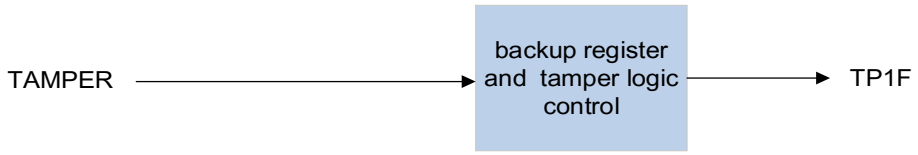
6.2. Characteristics

- Tamper sources can be chosen and tamper type is configurable.
- Five 32-bit (20 bytes total) universal backup registers which can keep data under power saving mode. Backup register will be reset if tamper event detected.

6.3. Function overview

6.3.1. Block diagram

Figure 6-1. Block diagram of BKP



The BKP unit includes:

- Tamper event / interrupt
- 32-bit backup registers

6.3.2. Tamper detection

The BKP_DATAx pin input can be used for tamper event detection under edge detection mode or level detection mode with configurable filtering setting.

backup registers (BKP_DATAx)

The BKP backup registers are located in the V_{DD} backup domain. The wake up action from Standby Mode or System Reset does not affect these registers.

These registers are only reset by detected tamper event and backup domain reset.

Tamper detection function initialization

BKP tamper detection function can be independently enabled on tamper input pin by setting corresponding TP1EN bit. Tamper detection configuration is set before enable TP1EN bit. When the tamper event is detected, the corresponding flag (TP1F) will assert. Tamper event can generate an interrupt if tamper interrupt enable (TPIE) is set. Any tamper event will reset all backup registers (BKP_DATAx).

Edge detection mode on tamper input detection

When FLT bit is set to 0x0, the tamper detection is set to edge detection mode and TP1EG bit determines the rising edge or falling edge is the detecting edge. When tamper detection is under edge detection mode, the internal pull-up resistors on the tamper detection input pin are deactivated.

Because of detecting the tamper event will reset the backup registers (BKP_DATAx), writing to the backup register should ensure that the tamper event reset and the writing operation will not occur at the same time, a recommend way to avoid this situation is disable the tamper detection before writing to the backup register and re-enable tamper detection after finish writing.

Note: Tamper detection is still running when V_{DD} power is switched off if tamper is enabled.

Level detection mode with configurable filtering on tamper input detection

When FLT bit is not reset to 0x0, the tamper detection is set to level detection mode and FLT bit determines the consecutive number of samples (2, 4 or 8) needed for valid level. When DISPU is set to 0x0(this is default), the internal pull-up resistance will pre-charge the tamper input pin before each sampling and thus larger capacitance is allowed to connect to the tamper input pin. The pre-charge duration is configured through PRCH bit. Higher capacitance needs long pre-charge time.

The time interval between each sampling is also configurable. Through adjusting the sampling frequency (FREQ), software can balance between the power consuming and tamper detection latency.

6.4. BKP interrupts

All BKP interrupts are connected to the EXTI controller.

Below steps should be followed if you want to use the BKP tamper:

1. Configure and enable the corresponding interrupt line to BKP tamper event of EXTI and set the rising edge for triggering;
2. Configure and enable the BKP tamper global interrupt;
3. Configure and enable the BKP tamper function.

Table 6-1. BKP interrupts control

Interrupt	Event flag	Control Bit	Exit Sleep	Exit Deep-sleep	Exit Standby
Tamper 1	TP1F	TPIE	Y	Y(*)	Y(*)

* Only active when BKP clock source is IRC40K.

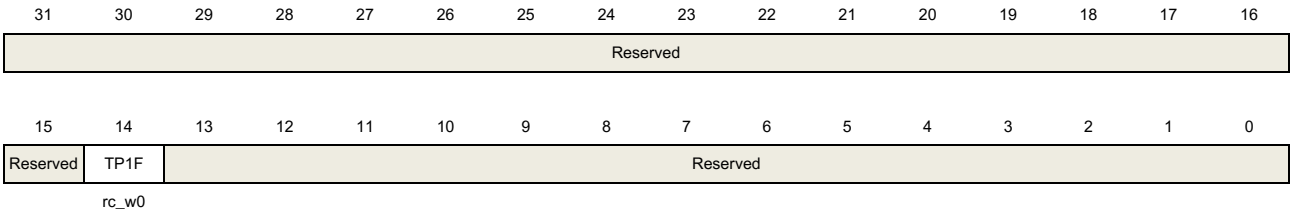
6.5. Register definition

BKP base address: 0x4000 2800

6.5.1. Status register (BKP_STAT)

Address offset: 0x0C
Backup domain reset value: 0x0000 0000

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

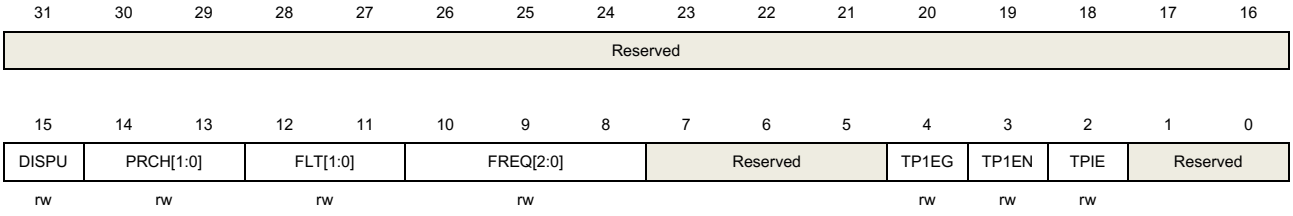


Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value
14	TP1F	TAMPER detected flag Set to 1 by hardware when tamper detection is found on tamper1 input pin. Software can clear this bit by writing 0 into this bit.
13:0	Reserved	Must be kept at reset value

6.5.2. Tamper register (BKP_TAMP)

Address offset: 0x40
Backup domain reset: 0x0000 0000
System reset: no effect

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



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15	DISPU	TAMPER pull up disable bit 0: Enable inner pull-up before sampling for pre-charge TAMPER pin 1: Disable pre-charge duration

14:13	PRCH[1:0]	Pre-charge duration time of TAMPER This setting determines the pre-charge time before each sampling. 0x0: 1 BKP clock 0x1: 2 BKP clock 0x2: 4 BKP clock 0x3: 8 BKP clock
12:11	FLT[1:0]	TAMPER filter count setting This bit determines the tamper sampling type and the number of consecutive sample. 0x0: Detecting tamper event using edge mode. Pre-charge duration is disabled automatically 0x1: Detecting tamper event using level mode.2 consecutive valid level samples will make an effective tamper event 0x2: Detecting tamper event using level mode.4 consecutive valid level samples will make an effective tamper event 0x3: Detecting tamper event using level mode.8 consecutive valid level samples will make an effective tamper event
10:8	FREQ[2:0]	Sampling frequency of tamper event detection 0x0: Sample once every 32768 BKPCLK (1Hz if BKPCLK=32.768KHz) 0x1: Sample once every 16384 BKPCLK (2Hz if BKPCLK=32.768KHz) 0x2: Sample once every 8192 BKPCLK (4Hz if BKPCLK=32.768KHz) 0x3: Sample once every 4096 BKPCLK (8Hz if BKPCLK=32.768KHz) 0x4: Sample once every 2048 BKPCLK (16Hz if BKPCLK=32.768KHz) 0x5: Sample once every 1024 BKPCLK (32Hz if BKPCLK=32.768KHz) 0x6: Sample once every 512 BKPCLK (64Hz if BKPCLK=32.768KHz) 0x7: Sample once every 256 BKPCLK (128Hz if BKPCLK=32.768KHz)
7:5	Reserved	Must be kept at reset value
4	TP1EG	Tamper 1 event trigger edge If tamper detection is in edge mode (FLT =0): 0: Rising edge triggers a tamper detection event 1: Falling edge triggers a tamper detection event If tamper detection is in level mode (FLT !=0): 0: Low level triggers a tamper detection event 1: High level triggers a tamper detection event
3	TP1EN	Tamper 1 detection enable 0: Disable tamper 1 detection function 1: Enable tamper 1 detection function
2	TPIE	Tamper detection interrupt enable 0: Disable tamper interrupt 1: Enable tamper interrupt

1:0	Reserved	Must be kept at reset value

Note: It's strongly recommended that reset the TP1EN before change the tamper configuration.

6.5.3. Backup registers (BKP_DATAx) (x=0…4)

Address offset: 0x50~0x60
Backup domain reset: 0x0000 0000
System reset: no effect
This register is writing and reading protected

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
DATA[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA[15:0]															
rw															

Bits	Fields	Descriptions
31:0	DATA[31:0]	Data These registers can be wrote or read by software. The content remains valid even in power saving mode. Tamper detection flag TPxF assertion will reset these registers. Also, when the FMC readout protection disables will reset these registers.

7. Reset and clock unit (RCU)

7.1. Reset control unit (RCTL)

7.1.1. Overview

GD32E25x reset control includes the control of three kinds of reset: power reset, system reset and backup domain reset. The power on reset, known as a cold reset, resets the full system except the backup domain during a power up. A system reset resets the processor core and peripheral IP components with the exception of the SW-DP controller and the backup domain. A 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.

7.1.2. Function overview

Power Reset

The power reset is generated by either an external reset as power on and power down reset (POR / PDR reset), or by the internal reset generator when exiting standby mode. The power reset sets all registers to their reset values except the backup domain. When exiting Standby mode, all registers within the 1.2V domain are reset to their default values, while registers outside the 1.2V domain, such as WKUP, FWDGT, and Standby mode control registers, remain unaffected. The power reset which active signal is low will be de-asserted when the internal LDO voltage regulator is ready to provide 1.2V power for GD32E25x series. 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®-M23 application interrupt and reset control register is set (SW_RSTn)
- Option byte loader reset (OBL_RSTn)
- Reset generated when entering Standby mode when resetting nRST_STDBY bit in user option bytes (OB_STDBY_RSTn)
- Reset generated when entering Deep-sleep mode when resetting nRST_DPSLP bit in user option bytes (OB_DPSLP_RSTn)

Note: The NRST pin can be configured with the option byte NRST_MDSEL in the following two modes:

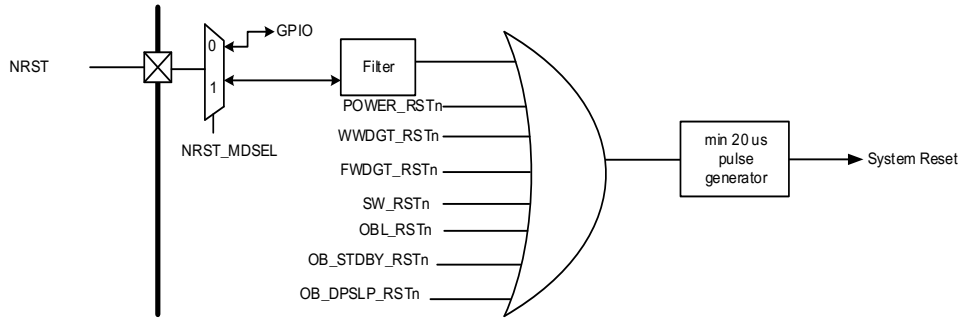
1. Input/output mode: the GPIO function of the NRST pin is not available in this mode. The reset signal can be transferred from the NRST pin to the device, causing the device to reset, the reset pulse signal can be reflected through the NRST pin, and the minimum reset pulse duration is 20 μs.
2. GPIO mode: NRST pin can only function as standard GPIO, reset function is not available, reset signal is only inside the device, not reflected in NRST pin.

Note: For the GD32E251xx, the default mode of the NRST pin is GPIO mode, whereas for the GD32E252xx, the default mode of the NRST pin is Input/Output mode.

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 7-1. 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} power on).

7.2. Clock control unit (CCTL)

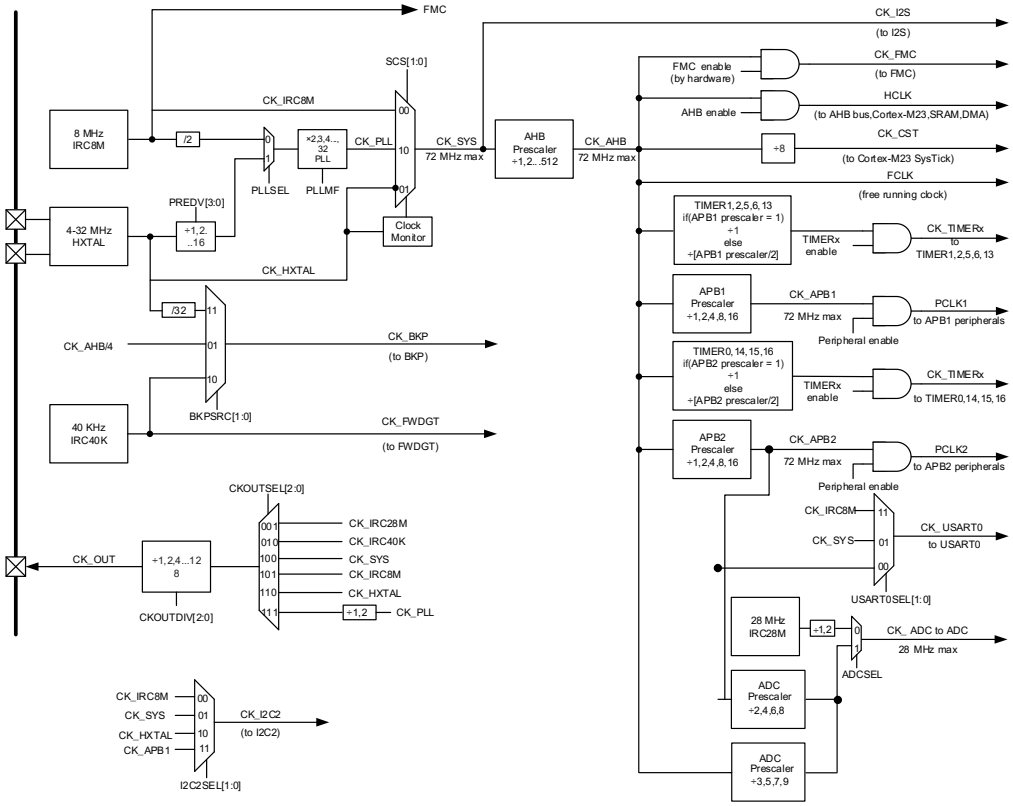
7.2.1. Overview

The clock control unit provides a range of frequencies and clock functions. These include an Internal 8 MHz RC oscillator (IRC8M), an Internal 28 MHz RC oscillator (IRC28M), a High speed crystal oscillator (HXTAL), an Internal 40KHz RC oscillator (IRC40K), a Phase Lock Loop (PLL), a HXTAL clock monitor, clock prescalers, clock multiplexers and clock gating circuitry.

The clocks of the AHB, APB and Cortex®-M23 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 72 MHz.

Figure 7-2. Clock tree



The frequency of AHB, APB2 and APB1 domains can be configured by each prescaler. The maximum frequency of the AHB, APB2 and APB1 domains is 72 MHz / 72 MHz / 72 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 ADC are clocked by the clock of APB2 divided by 2, 4, 6, 8 or by the clock of AHB divided by 3, 5, 7, 9 or IRC28M or IRC28M / 2 clock for GD32E25x series selected by ADCSEL bit in configuration register 2 (RCU_CFG2).

The USART0 is clocked by IRC8M clock or system clock or APB2 clock, which selected by USART0SEL bits in configuration register 2 (RCU_CFG2).

The I2C2 is clocked by IRC8M clock or HXTAL clock or system clock or APB1 clock, which selected by I2C2SEL bits in configuration register 2 (RCU_CFG2).

The BKP is clocked by IRC40K clock, AHB clock divided by 4 or HXTAL clock divided by 32 which select by BKPSRC bits in Backup Domain Control Register (RCU_BDCTL).

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

The TIMERs are clocked by the clock divided from CK_APB. The frequency of TIMERs clock is equal to CK_APBx or twice the CK_APBx. If the APB prescaler is 1, the timer clock frequencies are set to AHB frequency divide by 1. Otherwise, they are set to the AHB frequency divide by half of APB prescaler.

7.2.2. Characteristics

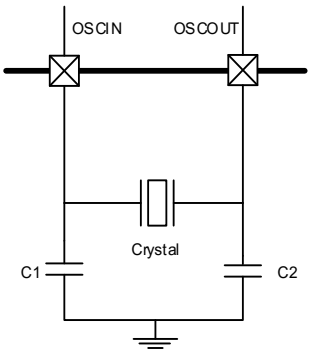
- 4 to 32 MHz High speed crystal oscillator (HXTAL).
- Internal 8 MHz RC oscillator (IRC8M).
- Internal 28 MHz RC oscillator (IRC28M).
- Internal 40 KHz RC oscillator (IRC40K).
- PLL clock source can be HXTAL or IRC8M.
- HXTAL clock monitor.

7.2.3. Function overview

High speed crystal oscillator (HXTAL)

The high speed 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 7-3. HXTAL clock source

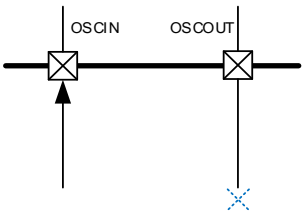


The HXTAL crystal oscillator can be switched on or off using the HXTALEN bit in the control register 0, RCU_CTL0. The HXTALSTB flag in control register 0, RCU_CTL0 indicates if the high-speed external crystal oscillator is stable. When the HXTAL is powered up, it will not be released for use until this 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 0 RCU_CTL0. During bypass mode, the signal is connected to OSCIN, and OSCOUT

remains in the suspended state, as shown in ***Figure 7-4. HXTAL clock source in bypass mode***. The CK_HXTAL is equal to the external clock which drives the OSCIN pin.

Figure 7-4. HXTAL clock source in bypass mode



Note: The HXTAL clock source of the GD32E251xx series MCUs only supports input from an active crystal oscillator and does not support a passive crystal oscillator. When using HXTAL, be sure to enable bypass mode

Internal 8 MHz RC oscillator (IRC8M)

The Internal 8 MHz 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 0, RCU_CTL0. The IRC8MSTB flag in the control register 0, RCU_CTL0 is used to indicate if the internal 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 interrupt register, RCU_INT, is set when the IRC8M becomes stable. The IRC8M clock can also be used as 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 initially wakes-up.

Phase locked loop (PLL)

The internal Phase Locked Loop, PLL, can provide 16~72 MHz clock output which is 2 ~32 multiples of a fundamental reference frequency of 4 ~ 32 MHz.

The PLL can be switched on or off by using the PLLEN bit in the control register 0, RCU_CTL0. The PLLSTB flag in the control register 0, RCU_CTL0 will indicate if the PLL clock is stable. An interrupt can be generated if the related interrupt enable bit, PLLSTBIE, in the Interrupt register, RCU_INT, is set as the PLL becomes stable.

Internal 28 MHz RC oscillator (IRC28M)

The Internal 28 MHz RC Oscillator, IRC28M, has a fixed frequency of 28 MHz and dedicated

as ADC clock. The IRC28M RC oscillator can be switched on or off using the IRC28MEN bit in the control register 1 (RCU_CTL1). The IRC28MSTB flag in the control register 1 (RCU_CTL1) is used to indicate if the internal 28M RC oscillator is stable. An interrupt can be generated if the related interrupt enable bit, IRC28MSTBIE, in the Interrupt register, RCU_INT, is set when the IRC28M becomes stable.

Internal 40 KHz RC oscillator (IRC40K)

The Internal 40 KHz 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 Interrupt register RCU_INT is set when the IRC40K becomes stable.

System clock (CK_SYS) selection

After the system reset, the default CK_SYS source will be IRC8M and can be switched to HXTAL or PLL by changing the system clock switch bits, SCS, in the 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 directly by the CK_SYS or the 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 0, RCU_CTL0. 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 Flag, CKMIF, in the 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®-M23. If the HXTAL is selected as the clock source of CK_SYS or PLL, the HXTAL failure will force the CK_SYS source to IRC8M and the PLL will be disabled automatically.

Clock output capability

The clock output capability is ranging from 40 kHz to 72 MHz. There are several clock signals can be selected via the CK_OUT Clock source selection bits, CKOUTSEL, in the 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 7-1. Clock source select

Clock source selection bits	Clock source
000	No Clock
001	CK_IRC28M
010	CK_IRC40K
011	No Clock
100	CK_SYS
101	CK_IRC8M
110	CK_HXTAL
111	CK_PLL or CK_PLL/2

The CK_OUT frequency can be reduced by a configurable binary divider, controlled by the CKOUTDIV[2:0] bits, in the configuration register 0(RCU_CFG0).

Deep-sleep mode clock control

If the USART0 clock is selected IRC8M clock in Deep-sleep mode, they are capable of open IRC8M clock or close IRC8M clock, which is used by the USART0 to wake up the Deep-sleep mode.

Voltage control

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

Table 7-2. Core domain voltage selected in Deep-sleep mode -

DSLPVS[1:0]	Deep-sleep mode voltage(V)
00	1.2
01	1.1
10	1.0
11	0.9

The RCU_DSV register are protected by voltage key register (RCU_VKEY). Only after write 0x1A2B3C4D to the RCU_VKEY register, the RCU_DSV register can be written.

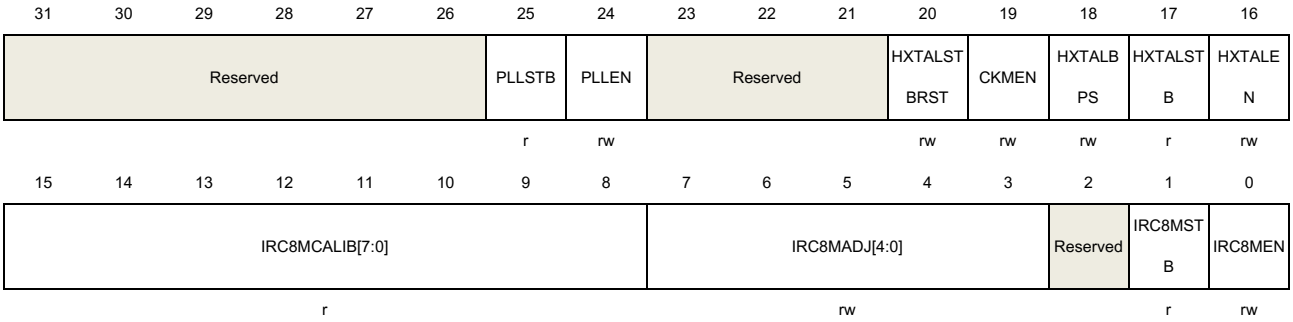
7.3. Register definition

RCU base address: 0x4002 1000

7.3.1. Control register 0 (RCU_CTL0)

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).



Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
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
23:21	Reserved	Must be kept at reset value.
20	HXTALSTBRST	HXTAL stabilization reset Set and reset by software. 0: HXTAL stabilization not clean 1: HXTAL stabilization clean
19	CKMEN	HXTAL clock monitor enable 0: Disable the external 4 ~ 32 MHz crystal oscillator (HXTAL) clock monitor 1: Enable the external 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 internal hardware will switch the system clock to be the internal high speed IRC8M RC clock. The way to recover the original system clock is by either an external reset,

		power on reset or clearing CKMIF by software. **Note:** When the HXTAL clock monitor is enabled, the hardware will automatically enable the IRC8M internal RC oscillator regardless of the control bit, IRC8MEN, state.
18	HXTALBPS	External 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	External 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	External high speed oscillator 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. Reset by hardware when entering Deep-sleep or Standby mode. 0: External 4 ~ 32 MHz crystal oscillator disabled 1: External 4 ~ 32 MHz crystal oscillator enabled
15:8	IRC8MCALIB[7:0]	Internal 8M RC oscillator calibration value register These bits are load automatically at power on.
7:3	IRC8MADJ[4:0]	Internal 8M RC oscillator clock trim adjust value These bits are set by software. The trimming value is there 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	IRC8M high speed internal 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 high speed 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

7.3.2. 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
PLLDV		CKOUTDIV[2:0]		PLLMF[4]		CKOUTSEL[2:0]		Reserved		PLLMF[3:0]				PLLPRE DV	PLLSEL
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	PLLDV	The CK_PLL divide by 1 or 2 for CK_OUT 0: CK_PLL divide by 2 for CK_OUT 1: CK_PLL divide by 1 for CK_OUT
30:28	CKOUTDIV[2:0]	The CK_OUT divider which the CK_OUT frequency can be reduced, see bits 26:24 of RCU_CFG0 for CK_OUT. 000: The CK_OUT is divided by 1 001: The CK_OUT is divided by 2 010: The CK_OUT is divided by 4 011: The CK_OUT is divided by 8 100: The CK_OUT is divided by 16 101: The CK_OUT is divided by 32 110: The CK_OUT is divided by 64 111: The CK_OUT is divided by 128
27	PLLMF[4]	Bit 4 of PLLMF register see bits 21:18 of RCU_CFG0.
26:24	CKOUTSEL[2:0]	CK_OUT clock source selection Set and reset by software. 000: No clock selected 001: Internal 28M RC oscillator clock selected 010: Internal 40K RC oscillator clock selected 011: No clock selected 100: System clock selected 101: Internal 8MHz RC oscillator clock selected 110: External high speed oscillator clock selected 111: (CK_PLL / 2) or CK_PLL selected depend on PLLDV
23:22	Reserved	Must be kept at reset value.

21:18	PLLMF[3:0]	PLL multiply factor These bits and bit 27 of RCU_CFG0 are written by software to define the PLL multiplication factor. 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 15) 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 32) **Note:** The PLL output frequency must not exceed 72 MHz.
17	PLLPREDV	HXTAL divider for PLL source clock selection. This bit is the same bit as bit PREDV[0] from RCU_CFG1. Refer to RCU_CFG1 PREDV bits description. Set and cleared by software to divide or not which is selected to PLL. 0: HXTAL clock selected 1: HXTAL / 2 clock selected
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 selected as source clock of PLL
15:14	ADCPSC[1:0]	ADC clock prescaler selection These bits and bit 31 of RCU_CFG2 are written by software to define the ADC clock prescaler. Set and cleared by software. 000: (CK_APB2 / 2) selected 001: (CK_APB2 / 4) selected 010: (CK_APB2 / 6) selected 011: (CK_APB2 / 8) selected 100: (CK_AHB / 3) selected 101: (CK_AHB / 5) selected 110: (CK_AHB / 7) selected 111: (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. 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 by HXTAL clock monitor when the HXTAL failure is detected and the HXTAL is selected as the clock source of CK_SYS or PLL. 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

7.3.3. 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	Reserved	IRC28M STBIC	PLL STBIC	HXTAL STBIC	IRC8M STBIC	Reserved	IRC40K STBIC
								w		w	w	w	w		w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		IRC28M STBIE	PLL STBIE	HXTAL STBIE	IRC8M STBIE	Reserved	IRC40K STBIE	CKMIF	Reserved	IRC28M STBIF	PLL STBIF	HXTAL STBIF	IRC8M STBIF	Reserved	IRC40K STBIF
		rw	rw	rw	rw		rw	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	Reserved	Must be kept at reset value
21	IRC28MSTBIC	IRC28M stabilization interrupt clear Write 1 by software to reset the IRC28MSTBIF flag. 0: Not reset IRC28MSTBIF flag 1: Reset IRC28MSTBIF 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	Reserved	Must be kept at reset value
16	IRC40KSTBIC	IRC40K stabilization interrupt clear Write 1 by software to reset the IRC40KSTBIF flag. 0: Not reset IRC40KSTBIF flag 1: Reset IRC40KSTBIF flag
15:14	Reserved	Must be kept at reset value.
13	IRC28MSTBIE	IRC28M stabilization interrupt enable Set and reset by software to enable/disable the IRC28M stabilization interrupt. 0: Disable the IRC28M stabilization interrupt 1: Enable the IRC28M 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	Reserved	Must be kept at reset value
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 by software when setting the CKMIC bit. 0: Clock operating normally 1: HXTAL clock stuck
6	Reserved	Must be kept at reset value.
5	IRC28MSTBIF	IRC28M stabilization interrupt flag Set by hardware when the IRC28M is stable and the IRC28MSTBIE bit is set. Reset by software when setting the IRC28MSTBIC bit. 0: No IRC28M stabilization interrupt generated 1: IRC28M stabilization interrupt generated
4	PLLSTBIF	PLL stabilization interrupt flag Set by hardware when the PLL is stable and the PLLSTBIE bit is set. Reset by software when setting the PLLSTBIC bit. 0: No PLL stabilization interrupt generated 1: PLL stabilization interrupt generated
3	HXTALSTBIF	HXTAL stabilization interrupt flag Set by hardware when the external 4 ~ 32 MHz crystal oscillator clock is stable and the HXTALSTBIE bit is set. Reset by software when setting the HXTALSTBIC bit. 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 by software when setting the IRC8MSTBIC bit. 0: No IRC8M stabilization interrupt generated 1: IRC8M stabilization interrupt generated
1	Reserved	Must be kept at reset value
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 by software when setting the IRC40KSTBIC bit. 0: No IRC40K stabilization clock ready interrupt generated 1: IRC40K stabilization interrupt generated

7.3.4. 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													TIMER16 RST	TIMER15 RST	TIMER14 RST
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	USART0 RST	Reserved	SPI0 RST	TIMER0 RST	Reserved	ADC RST	Reserved								CFGCMP RST
	rw		rw	rw		rw									rw

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18	TIMER16RST	TIMER16 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER16
17	TIMER15RST	TIMER15 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER15
16	TIMER14RST	TIMER14 reset This bit is set and reset by software. 0: No reset 1: Reset the TIMER14
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	Reserved	Must be kept at reset value.
12	SPI0RST	SPI0 reset This bit is set and reset by software. 0: No reset 1: Reset the SPI0
11	TIMER0RST	TIMER0 reset

		This bit is set and reset by software. 0: No reset 1: Reset the TIMER0
10	Reserved	Must be kept at reset value.
9	ADCRST	ADC reset This bit is set and reset by software. 0: No reset 1: Reset the ADC
8:1	Reserved	Must be kept at reset value.
0	CFGCMPRST	System configuration and comparator reset This bit is set and reset by software. 0: No reset 1: Reset system configuration and comparator

7.3.5. 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
DAC1RST	Reserved	DAC0RST	PMURST	Reserved			I2C2RST	Reserved	I2C1RST	I2C0RST	Reserved			UART1RST	Reserved
rw		rw	rw				rw		rw	rw				rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SPI1RST	Reserved		WWDGTRST	Reserved		TIMER13RST	Reserved		TIMER6RST	TIMER5RST	Reserved		TIMER2RST	TIMER1RST
	rw			rw			rw			rw	rw			rw	rw

Bits	Fields	Descriptions
31	DAC1RST	DAC1 reset This bit is set and reset by software. 0: No reset 1: Reset DAC1
30	Reserved	Must be kept at reset value
29	DAC0RST	DAC0 reset This bit is set and reset by software. 0: No reset 1: Reset DAC0
28	PMURST	Power control reset

		This bit is set and reset by software. 0: No reset 1: Reset power control unit
27:25	Reserved	Must be kept at reset value.
24	I2C2RST	I2C2 reset This bit is set and reset by software. 0: No reset 1: Reset I2C2
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 I2C1
21	I2C0RST	I2C0 reset This bit is set and reset by software. 0: No reset 1: Reset I2C0
20:18	Reserved	Must be kept at reset value.
17	UART1RST	UART1 reset This bit is set and reset by software. 0: No reset 1: Reset UART1
16:15	Reserved	Must be kept at reset value.
14	SPI1RST	SPI1 reset This bit is set and reset by software. 0: No reset 1: Reset SPI1
13:12	Reserved	Must be kept at reset value.
11	WWDGTRST	Window watchdog timer reset This bit is set and reset by software. 0: No reset 1: Reset window watchdog timer
10:9	Reserved	Must be kept at reset value.
8	TIMER13RST	TIMER13 timer reset This bit is set and reset by software. 0: No reset

		1: Reset TIMER13
7:6	Reserved	Must be kept at reset value.
5	TIMER6RST	TIMER6 timer reset This bit is set and reset by software. 0: No reset 1: Reset TIMER6
4	TIMER5RST	TIMER5 timer reset This bit is set and reset by software. 0: No reset 1: Reset TIMER5
3:2	Reserved	Must be kept at reset value.
1	TIMER2RST	TIMER2 timer reset This bit is set and reset by software. 0: No reset 1: Reset TIMER2
0	TIMER1RST	TIMER1 timer reset This bit is set and reset by software. 0: No reset 1: Reset TIMER1

7.3.6. 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).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							NEW_FEAT_EN	Reserved	PFEN	Reserved			PBEN	PAEN	Reserved
							rw		rw				rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLAEN	Reserved								CRCEN	Reserved	FMC SPEN	Reserved	SRAM SPEN	Reserved	DMAEN
rw									rw		rw		rw		rw

Bits	Fields	Descriptions
31:25	Reserved	Must be kept at reset value.
24	NEW_FEAT_EN	New feature clock enable 0: Disabled new feature clock

		1: Enabled new feature clock
23	Reserved	Must be kept at reset value.
22	PFEN	GPIO port F clock enable This bit is set and reset by software. 0: Disabled GPIO port F clock 1: Enabled GPIO port F clock
21:19	Reserved	Must be kept at reset value.
18	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
17	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
16	Reserved	Must be kept at reset value.
15	CLAEN	CLA clock enable This bit is set and reset by software. 0: Disabled CLA clock 1: Enabled CLA clock
14:7	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 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 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	Reserved	Must be kept at reset value.
0	DMAEN	DMA clock enable This bit is set and reset by software. 0: Disabled DMA clock 1: Enabled DMA clock

7.3.7. 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									DBGMCUEN	Reserved			TIMER16EN	TIMER15EN	TIMER14EN
									rw				rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	USART0EN	Reserved	SPI0EN	TIMER0EN	Reserved	ADCEN	Reserved								CFGCMPEN
	rw		rw	rw		rw									rw

Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22	DBGMCUEN	DBGMCU clock enable This bit is set and reset by software. 0: Disabled DBGMCU clock 1: Enabled DBGMCU clock
21:19	Reserved	Must be kept at reset value.
18	TIMER16EN	TIMER16 timer clock enable This bit is set and reset by software. 0: Disabled TIMER16 timer clock 1: Enabled TIMER16 timer clock
17	TIMER15EN	TIMER15 timer clock enable This bit is set and reset by software. 0: Disabled TIMER15 timer clock 1: Enabled TIMER15 timer clock
16	TIMER14EN	TIMER14 timer clock enable This bit is set and reset by software. 0: Disabled TIMER14 timer clock

		1: Enabled TIMER14 timer clock
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	Reserved	Must be kept at reset value.
12	SPI0EN	SPI0 clock enable This bit is set and reset by software. 0: Disabled SPI0 clock 1: Enabled SPI0 clock
11	TIMER0EN	TIMER0 timer clock enable This bit is set and reset by software. 0: Disabled TIMER0 timer clock 1: Enabled TIMER0 timer clock
10	Reserved	Must be kept at reset value.
9	ADCEN	ADC interface clock enable This bit is set and reset by software. 0: Disabled ADC interface clock 1: Enabled ADC interface clock
8:1	Reserved	Must be kept at reset value.
0	CFGCMPEN	System configuration and comparator clock enable This bit is set and reset by software. 0: Disabled system configuration and comparator clock 1: Enabled system configuration and comparator clock

7.3.8. 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
DAC1EN	Reserved	DAC0EN	PMUEN	Reserved			I2C2EN	Reserved	I2C1EN	I2C0EN	Reserved			UART1EN	Reserved
rw		rw	rw				rw		rw	rw				rw	

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

rw rw rw rw rw rw rw

Bits	Fields	Descriptions
31	DAC1EN	DAC1 clock enable This bit is set and reset by software. 0: Disabled DAC1 clock 1: Enabled DAC1 clock
30	Reserved	Must be kept at reset value.
29	DAC0EN	DAC0 clock enable This bit is set and reset by software. 0: Disabled DAC0 clock 1: Enabled DAC0 clock
28	PMUEN	Power interface clock enable This bit is set and reset by software. 0: Disabled Power interface clock 1: Enabled Power interface clock
27:25	Reserved	Must be kept at reset value.
24	I2C2EN	I2C2 clock enable This bit is set and reset by software. 0: Disabled I2C2 clock 1: Enabled I2C2 clock
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
20:18	Reserved	Must be kept at reset value.
17	UART1EN	UART1 clock enable This bit is set and reset by software. 0: Disabled UART1 clock 1: Enabled UART1 clock
16:15	Reserved	Must be kept at reset value.
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	Window watchdog timer clock enable This bit is set and reset by software. 0: Disabled window watchdog timer clock 1: Enabled window watchdog timer clock
10:9	Reserved	Must be kept at reset value.
8	TIMER13EN	TIMER13 timer clock enable This bit is set and reset by software. 0: Disabled TIMER13 timer clock 1: Enabled TIMER13 timer clock
7:6	Reserved	Must be kept at reset value.
5	TIMER6EN	TIMER6 timer clock enable This bit is set and reset by software. 0: Disabled TIMER6 timer clock 1: Enabled TIMER6 timer clock
4	TIMER5EN	TIMER5 timer clock enable This bit is set and reset by software. 0: Disabled TIMER5 timer clock 1: Enabled TIMER5 timer clock
3:2	Reserved	Must be kept at reset value.
1	TIMER2EN	TIMER2 timer clock enable This bit is set and reset by software. 0: Disabled TIMER2 timer clock 1: Enabled TIMER2 timer clock
0	TIMER1EN	TIMER1 timer clock enable This bit is set and reset by software. 0: Disabled TIMER1 timer clock 1: Enabled TIMER1 timer clock

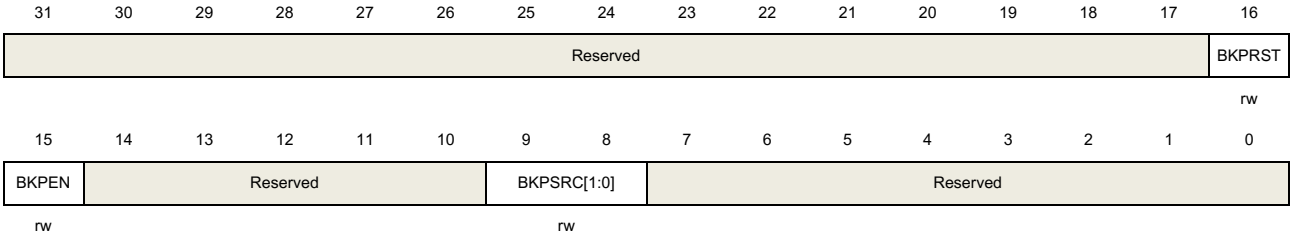
7.3.9. Backup domain control register (RCU_BDCTL)

Address offset: 0x20

Reset value: 0x0000 0000, reset by Backup domain Reset.

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

The BKPSRC and BKPEN bits of the Backup domain control register (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) has to be set.



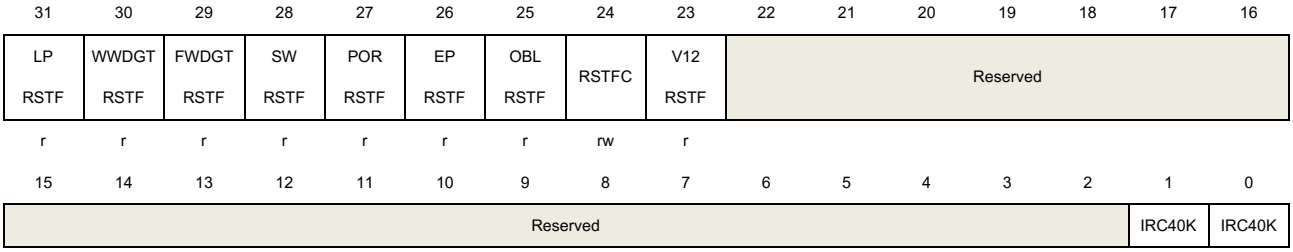
Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value
16	BKPRST	Backup domain reset This bit is set and reset by software. 0: No reset 1: Resets Backup domain
15	BKPEN	BKP clock enable This bit is set and reset by software. 0: Disabled BKP clock 1: Enabled BKP clock
14:10	Reserved	Must be kept at reset value
9:8	BKPSRC[1:0]	BKP clock entry selection Set and reset by software to control the BKP clock source. 00: No clock selected 01: (CK_AHB / 4) selected as BKP source clock 10: CK_IRC40K selected as BKP source clock 11: (CK_HXTAL / 32) selected as BKP source clock
7: 0	Reserved	Must be kept at reset value

7.3.10. Reset source /clock register (RCU_RSTSCK)

Address offset: 0x24

Reset value: 0x0C80 0000, reset flags reset by power Reset only, other reset by system reset.

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



| | | |

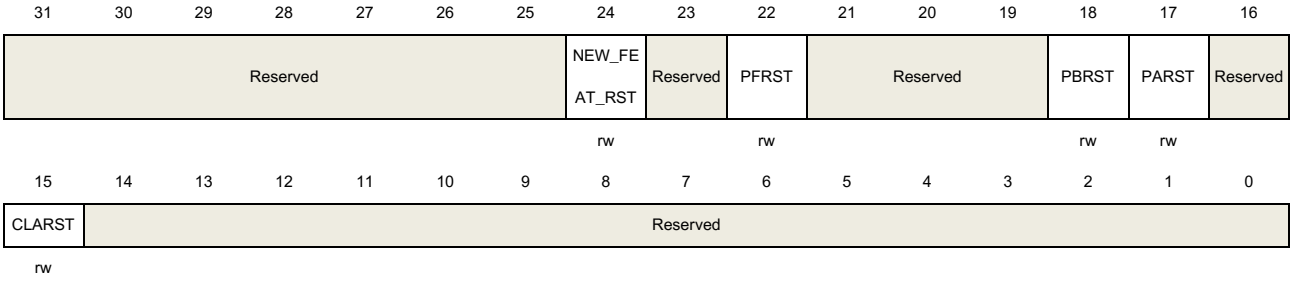
		This bit is set by software to clear all reset flags. 0: Not clear reset flags 1: Clear reset flags
23	V12RSTF	V12 domain Power reset flag Set by hardware when a V12 domain Power reset generated. Reset by writing 1 to the RSTFC bit. 0: No V12 domain Power reset generated 1: V12 domain Power reset generated
22:2	Reserved	Must be kept at reset value.
1	IRC40KSTB	IRC40K stabilization 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

7.3.11. 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
31:25	Reserved	Must be kept at reset value.
24	NEW_FEAT_RST	New feature reset This bit is set and reset by software. 0: No reset for new feature 1: Reset for new feature
23	Reserved	Must be kept at reset value.
22	PFRST	GPIO port F reset

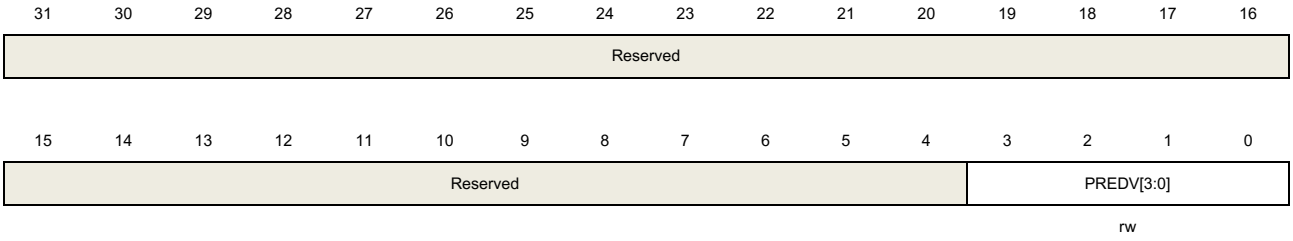
		This bit is set and reset by software. 0: No reset GPIO port F 1: Reset GPIO port F
21:19	Reserved	Must be kept at reset value.
18	PBRST	GPIO port B reset This bit is set and reset by software. 0: No reset GPIO port B 1: Reset GPIO port B
17	PARST	GPIO port A reset This bit is set and reset by software. 0: No reset GPIO port A 1: Reset GPIO port A
16	Reserved	Must be kept at reset value.
15	CLARST	CLA reset This bit is set and reset by software. 0: No reset CLA module 1: Reset CLA module
14:0	Reserved	Must be kept at reset value.

7.3.12. 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).



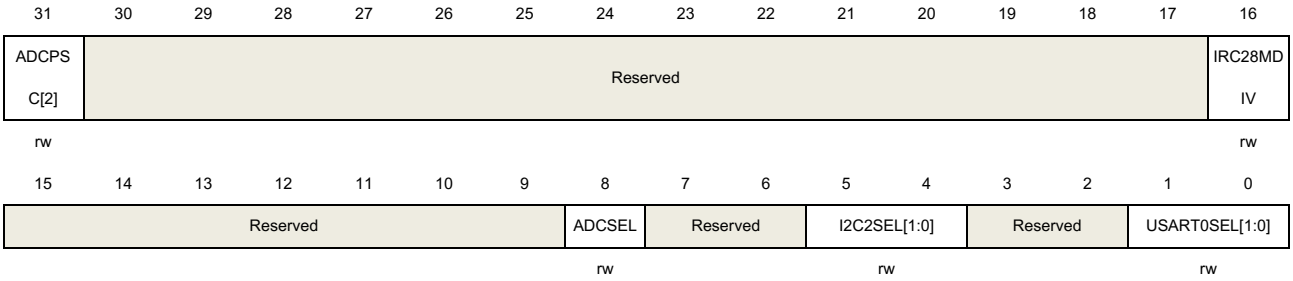
Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	PREDV[3:0]	CK_HXTAL divider previous PLL This bit is set and reset by software. These bits can be written when PLL is disabled. **Note:** The bit 0 of PREDV is same as bit 17 of RCU_CFG0, so modifying bit 17 of RCU_CFG0 also modifies bit 0 of RCU_CFG1. The CK_HXTAL is divided by (PREDV + 1).

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

7.3.13. Configuration register 2 (RCU_CFG2)

Address offset: 0x30
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	ADCPSC[2]	Bit 2 of ADCPSC see bits 15:14 of RCU_CFG0
30:17	Reserved	Must be kept at reset value.
16	IRC28MDIV	IRC28M divider or not 0: IRC28M /2 used as ADC clock 1: IRC28M used as ADC clock
15:9	Reserved	Must be kept at reset value.
8	ADCSEL	CK_ADC clock source selection This bit is set and reset by software.

		0: CK_ADC select CK_IRC28M 1: CK_ADC select CK_APB2 which is divided by 2, 4, 6, 8 or CK_AHB which is divided by 3, 5, 7, 9
7:6	Reserved	Must be kept at reset value.
5:4	I2C2SEL[1:0]	CK_I2C2 clock source selection This bit is set and reset by software. 00: CK_I2C2 select CK_IRC8M 01: CK_I2C2 select CK_SYS 10: CK_I2C2 select CK_HXTAL 11: CK_I2C2 select CK_APB1
3:2	Reserved	Must be kept at reset value.
1:0	USART0SEL[1:0]	CK_USART0 clock source selection This bit is set and reset by software. 00: CK_USART0 select CK_APB2 01: CK_USART0 select CK_SYS 10: Reserved 11: CK_USART0 select CK_IRC8M

7.3.14. Control register 1 (RCU_CTL1)

Address offset: 0x34

Reset value: 0x0000 XX80 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															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRC28MCALIB[7:0]								IRC28MADJ[4:0]					Reserved	IRC28MSTB	IRC28MEN
r								rw						r	rw

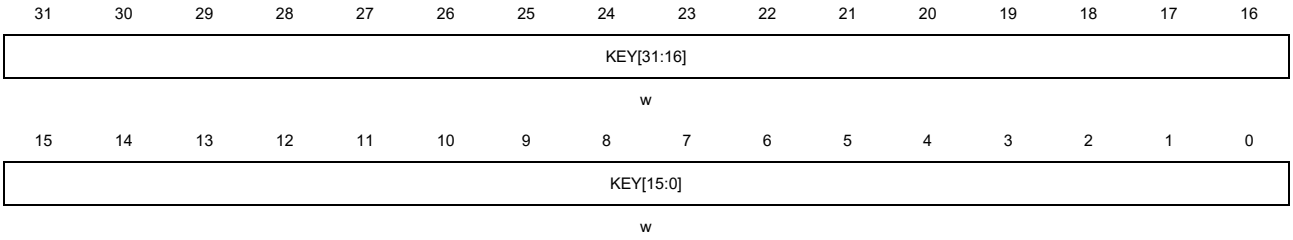
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:8	IRC28MCALIB[7:0]	Internal 28M RC Oscillator calibration value register These bits are load automatically at power on.
7:3	IRC28MADJ[4:0]	Internal 28M RC Oscillator clock trim adjust value These bits are set by software. The trimming value is these bits (IRC28MADJ) added to the IRC28MCALIB[7:0] bits. The trimming value should trim the IRC28M to 28MHz +/- 1%.

2	Reserved	Must be kept at reset value.
1	IRC28MSTB	IRC28M internal 28M RC oscillator stabilization flag Set by hardware to indicate if the IRC28M oscillator is stable and ready for use. 0: IRC28M oscillator is not stable 1: IRC28M oscillator is stable
0	IRC28MEN	IRC28M internal 28M RC oscillator enable Set and reset by software. 0: Internal 28 MHz RC oscillator disabled 1: Internal 28 MHz RC oscillator enabled

7.3.15. Voltage key register (RCU_VKEY)

Address offset: 0x100
Reset value: 0x0000 0000.

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

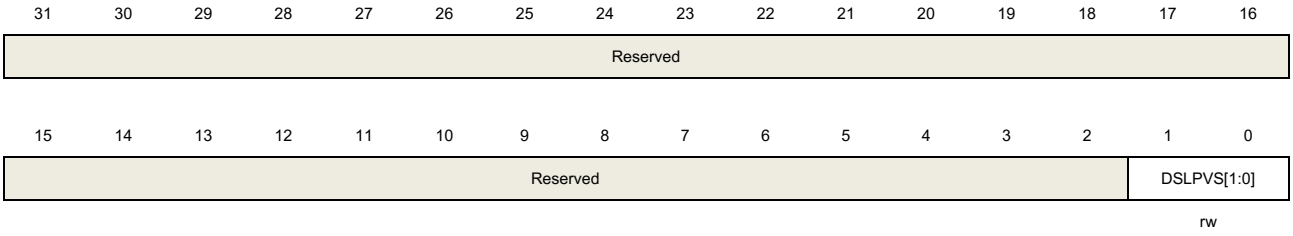


Bits	Fields	Descriptions
31:0	KEY[31:0]	The key of RCU_DSV register These bits are written only by software and read as 0. Only after writing 0x1A2B3C4D to the RCU_VKEY, the RCU_DSV register can be written.

7.3.16. Deep-sleep mode voltage register (RCU_DSV)

Offset: 0x134
Reset value: 0x0000 0000

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



Bits	Fields	Descriptions

31:2	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 1.2V in Deep-sleep mode 01: The core voltage is 1.1V in Deep-sleep mode 10: The core voltage is 1.0V in Deep-sleep mode 11: The core voltage is 0.9V in Deep-sleep mode

8. General-purpose I / O (GPIO)

8.1. Overview

There are up to 32 general purpose I / O pins (GPIO), named PA0 ~ PA15 and PB0 ~ PB11, PF0~ PF3 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. In GD32E251xx GPIO pins can be configured as functions pins with crossbar decoder decoder and dedicated digital functions.

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.
- Output drive speed selection.
- Analog input / output configuration.
- Alternate function input / output configuration.
- Port configuration lock.
- Single cycle toggle output capability.
- GPIO have the capable of state retention when system reset(exclude power reset).
- Two special interrupt vectors with direct pin interrupt sources--only for GD32E251xx.
- Crossbar decoder for pin assignment of peripheral functions--only for GD32E251xx.
- Contains direct pin interrupt source with shared interrupt vector for a total of 22--only for GD32E251xx.

8.3. Function overview

Each of the general-purpose I / O ports can be configured as GPIO inputs, GPIO outputs, AF

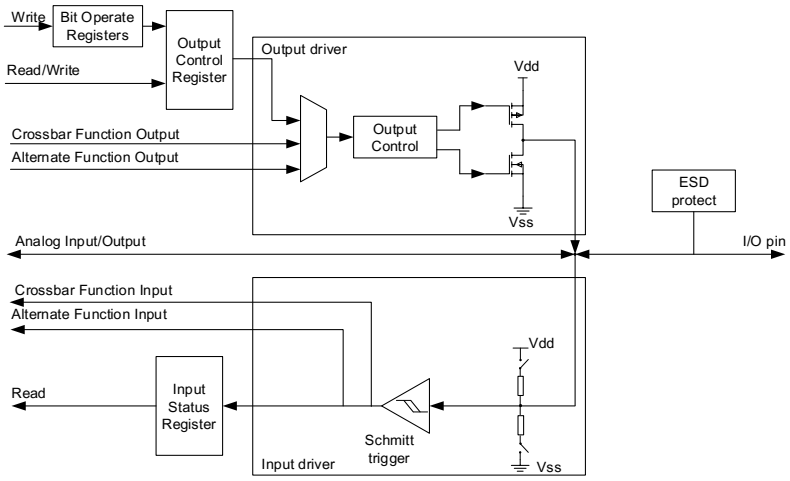
function or analog mode by GPIO 32-bit control register (GPIOx_CTL). AFIO input or output direction is decided by AFIO function after AFIO enable. When the port is output (GPIO output or AFIO output), it can be configured as push-pull or open drain mode by GPIO output mode registers (GPIOx_OMODE). And the port max speed can be configured by GPIO output speed registers (GPIOx_OSPD). Each port can be configured as floating (no pull-up and pull-down), pull-up or pull-down function by GPIO pull-up / pull-down registers (GPIOx_PUD).

Table 8-1. GPIO configuration table

PAD TYPE			CTLn	OMn	PUDn
GPIO INPUT	X	Floating	00	X	00
		Pull-up			01
		Pull-down			10
GPIO OUTPUT	Push-pull	Floating	01	0	00
		Pull-up			01
		Pull-down			10
	Open-drain	Floating		1	00
		Pull-up			01
		Pull-down			10
AFIO INPUT	X	Floating	10	X	00
		Pull-up			01
		Pull-down			10
AFIO OUTPUT	Push-pull	Floating	10	0	00
		Pull-up			01
		Pull-down			10
	Open-drain	Floating		1	00
		Pull-up			01
		Pull-down			10
ANALOG	X	X	11	X	XX

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 that input disabled without pull-up(PU) / pull-down(PD) resistors. But the Serial-Wired Debug pins are configured as AF PU / PD mode after reset:

GD32E252xx：

PA14: SWCLK in AF pull-down mode

PA13: SWDIO in AF pull-up mode

GD32E251xx:

PF2: SWCLK in AF pull-down mode

PB11: SWDIO in AF pull-up 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. When the GPIO pins are configured as input pins, the data on the external pads can be captured at every AHB 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 chooses 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 AHB write access by programming '1' to the bit operate register (GPIOx_BOP, or for clearing only GPIOx_BC, or for toggle only GPIOx_TG). The other bits will not be affected.

8.3.2. External interrupt / event lines

All ports have external interrupt capability. To use external interrupt lines, the port must be configured as input mode.

8.3.3. Alternate functions (AF)

When the port is configured as AFIO (set CTLy bits to "0b10", which is in GPIOx_CTL registers), the port is used as peripheral alternate functions. Each port has sixteen alternate functions which can be configured by GPIO alternate functions select registers (GPIOx_AFSELy(y=0,1)). The detail alternate function assignments for each port are described in the device datasheet.

8.3.4. Additional functions

Some pins have additional functions, which have priority over the configuration in the standard GPIO registers. When for ADC additional functions, the port must be configured as analog mode. When for BKP, WKUPx and oscillators additional functions, the port type is set automatically by related BKP, PMU and RCU registers. These ports can be used as normal GPIO when the additional functions disabled.

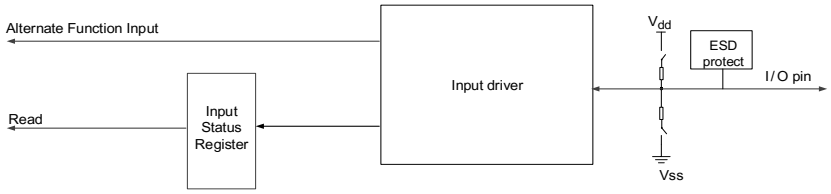
8.3.5. Input configuration

When GPIO pin is configured as input:

- The schmitt trigger input is activated.
- The weak pull-up and pull-down resistors could be chosen.
- Every AHB clock cycle the data present on the I / O pad 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



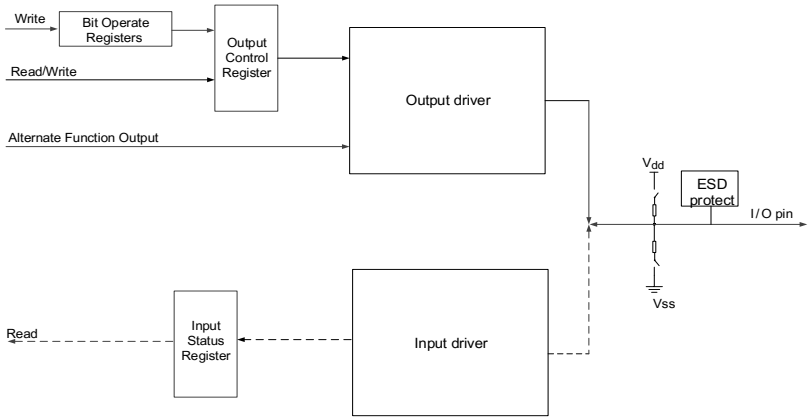
8.3.6. Output configuration

When GPIO pin is configured as output:

- The schmitt trigger input is activated.
- The weak pull-up and pull-down resistors could be chosen.
- The output buffer is enabled:
  - Open-Drain mode: The pad outputs "0" when a "0" in the output control register; while the pad leaves Hi-Z when a "1" in the output control register.
  - Push-Pull mode: The pad outputs "0" when a "0" in the output control register; while the pad outputs "1" when a "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



8.3.7. 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 de-activated.
- Read access to the port input status register gets the value "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



8.3.8. Alternate function (AF) configuration

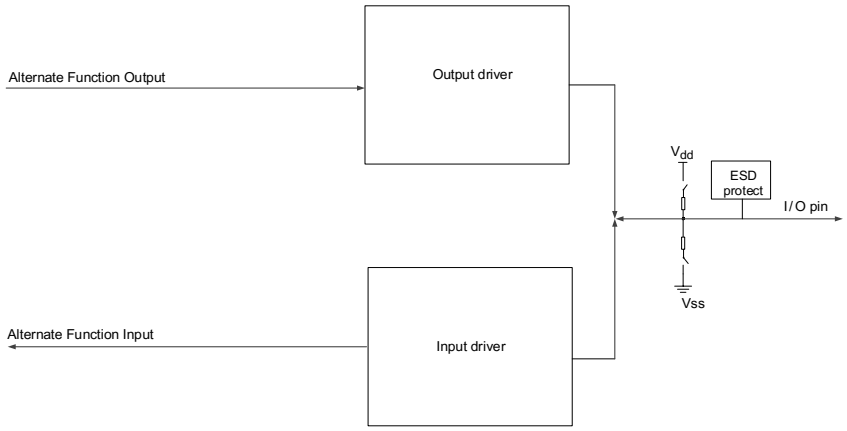
To suit for different device packages, the GPIO supports some alternate functions mapped to some other pins by software.

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 activated.
- The weak pull-up and pull-down resistors could be chosen.
- The data present on the I / O pin is sampled into the port input status register every AHB clock cycle.
- A read access to the port input status register gets the I / O state in Open-Drain mode.
- A read access to the port output control register gets the last written value in Push-Pull mode.

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



8.3.9. GPIO locking function

The locking mechanism allows the IO configuration to be protected.

The protected registers are GPIOx_CTL, GPIOx_OMODE, GPIOx_OSPD, GPIOx_PUD, GPIOx_AFSELy(y=0,1). It allows the I / O configuration to be frozen by the 32-bit locking register (GPIOx_LOCK). When the special LOCK sequence has been applied on a port bit, it is no longer able to modify the value of the port bit until the next reset. It should be recommended to be used in the configuration of driving a power module.

8.3.10. GPIO single cycle toggle function

GPIO could toggle the I / O output level in single AHB cycle by writing 1 to the corresponding

bit of GPIOx_TG register. The output signal frequency could up to the half of the AHB clock.

8.3.11. GPIO reset hold

When PIN_RSTMD bit is set in the SYSCFG_CFG1 registers, the following register values remain unchanged after system reset, and will be set to reset values when power is reset. When PIN_RSTMD bit is reset in the SYSCFG_CFG1 registers, the following register values are reset values after the system reset.

8.3.12. Priority Crossbar decoder (only for GD32E251xx)

Priority crossbar decoder is dedicated to assigning peripheral interfaces and follows priority when assigning pin to peripheral IO. The GD32E252xx does not support this function, and the peripheral interface allocation uses the AF function.The GD32E251xx supports this feature, except PB11 and PF2, if one of PA0-PA15,PB0-PB10,PF0 and PF3 is configured with AF mode, the crossbar decoder enable is masked, and the function cannot be used.

Figure 8-6. Digital functional map

GD32E251Kx	PB8	PB3	PA15	PA14	PA11	PA10	PA9	PA8	PB10	PB1	PB0	PA0	PA2	PA7	PA6	PA5	PA4	PF2
GD32E251Ex	PB8	PB3	PA15	PA13	PB10	PB1	PB0	PA2	PA7	PA6	PA5	PA4						
GD32E251Ex	CLA3OUT	EXTI_REMAP	CLA2OUT		I2C0_SDA	I2C0_CLK	CLA1OUT	CLA0OUT	DAC1_OUT1	DAC1_OUT0	DAC0_OUT1	DAC0_OUT0						
GD32E251Kx				CLA2OUT					I2C0_SDA	I2C0_SCL	CLA1OUT		CLA0OUT	DAC1_OUT1	DAC1_OUT0	DAC0_OUT1	DAC0_OUT0	SWCLK

Figure 8-6. Digital functional map shows the different package displayed digital functions, for details about how to enable dedicated digital functions, refer to the GPIO_DFEN register.

Figure 8-7. Crossbar decoder functional map

GD32E251Kx	PB9	PB8	PF0	PB5	PB4	PB3	PA15	PA14	PA13	PA12	PA11	PA10	PA9	PA8	PB2	PB10	PB1	PB0	PA0
GD32E251Ex	PB9	PB8	PF0	PB5	PB4	PB3	PA15	PA13	PA10	PA9	PB10	PB1	PB0	PA2	/	PA7	PA6	PA5	PA4
USART0_TX																			
USART0_RX																			
SPI0_SCK																			
SPI0_MISO																			
SPI0_MOSI																			
SPI0_NSS*																			
I2C1_SDA(SMBUS)																			
I2C1_SCL(SMBUS)																			
CMP0_SYNC_OUT																			
CMP0																			
CK_OUT																			
TIMER0_CH0																			
TIMER0_CH1																			
TIMER0_CH2																			
TIMER0_CH3																			
TIMER0_CH4																			
TIMER0_CH5																			
TIMER0_ETI																			
TIMER1_CH0_ETI																			
TIMER2_ETI																			
TIMER13_CH0																			
USART1-TX																			
USART1-RX																			
USART1-RTS																			
USART1-CTS																			
Pin skip settings	0 skip0	0 skip1	0 skip2	0 skip3	0 skip4	0 skip5	0 skip6	0 skip7	0 skip8	0 skip9	0 skip10	0 skip11	0 skip12	0 skip13	0 skip14	0 skip15	0 skip16	0 skip17	0 skip18

Figure 8-7. Crossbar decoder functional map shows the blue part indicates that peripheral functions can be mapped to the corresponding pin through the crossbar decoder, the pin in gray does not have the crossbar decoder function. The white part indicates that peripheral functions cannot be mapped to the corresponding pin through the crossbar decoder. The crossbar decoder follows pin priorities descending from left to right, and peripheral priorities descending from top to bottom.

The last line of pin skip setting is the configuration the GPIO_SKIP register corresponding to

each pin. If the pin needs to use the special digital / general purpose IO / analog function, the software will set the SKIPx (x = 0…18) bit in the GPIO_ SKIP register with corresponding the pin.

When a peripheral allocation pin is enabled, the unallocated pin in the leftmost blue area of the table is allocated to the peripheral. If a pin is already occupied by a higher-priority peripheral function assignment, when assigning a pin to the next peripheral, the hardware automatically skips the pin and selects the next unassigned pin.

The software configuration flow of assigning pin by using the priority crossbar decoder:

(a) If the pin assigns the analog function, configure the CTLn[1:0] (n = 0…15) bit filed into "0b11" in the GPIOx_CTL1 register. Or the pin assigns the GPIO mode, configure the CTLn[1:0] (n = 0…15) bit filed into "0b01 / 0b00" in the GPIOx_CTL1 register. Or the pin assigns the dedicated digital functions, set the corresponding bit in the GPIO_DFEN register. Then set the corresponding SKIPx (x = 0…18) bit in the GPIO_ SKIP register.
(b) In addition to analog, general IO functions,dedicated digital functions, the remaining pins can be assigned through the digital decoder, set the corresponding bit in the GPIO_DECEN register.
(c) Enable crossbar decoder by setting the DECEN bit in the GPIO_DECEN register.

8.3.13. INT0 and INT1 (only for GD32E251xx)

INT0,INT1 assign the pin independently of the crossbar decoder, and INT0,INT1 may be assigned to the pin that already have used the crossbar decoder. INT0,INT1 will monitor the assigned pins without interfering with peripherals assigned pins through the crossbar decoder.

When need assign the pin only to INT0 / INT1, set the SKIPx (x = 0…18) bit in the GPIO_SKIP register with corresponding the pin.

8.3.14. Pin match (only for GD32E251xx)

When a pin is used as input, if the input logical value of this pin & GPIO_MASK register corresponds to the value of the bit is not equal to the bit value of the GPIO_MATCH register & the bit value of the GPIO_MASK register, the pin mismatch event occurs, resulting in an interrupt.

8.3.15. Pin not match wakes up from deepsleep mode (only for GD32E251xx)

Whatever MATCHIE bit set or reset, the MCU can be wakeup from Deep-sleep mode with the pin not match interrupt. Before wakeup from Deep-sleep mode, need to configure corresponding bit in the GPIO_MASK and GPIO_MATCH registers.

8.4. Register definition

For GD32E252xx devices

GPIOA base address: 0x4800 0000

GPIOB base address: 0x4800 0400

GPIOF base address: 0x4800 1400

For GD32E251xx devices

GPIOA base address: 0x4800 0000

GPIOB base address: 0x4800 0400

GPIOSP base address: 0x4800 1000

GPIOF base address: 0x4800 1400

Note: GPIOSP base address only for GPIO_DECEN, GPIO_DFEN, GPIO_SKIP, GPIO_INTCTL, GPIO_MASK, GPIO_MATCH, GPIO_MATCHF registers.

8.4.1. Port control register (GPIOx_CTL, x=A, B, F)

Address offset: 0x00
GD32E252xx:
Reset value: 0x2800 0000 for port A, 0x0000 0000 for other.
GD32E251xx:
Reset value: 0x0000 0000 for A; 0x0080 0000 for B; 0x0000 0020 for port F.
Note: This register can be used without enabling the GPIO clock.

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]	CTL14[1:0]	CTL13[1:0]	CTL12[1:0]	CTL11[1:0]	CTL10[1:0]	CTL9[1:0]	CTL8[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
CTL7[1:0]	CTL6[1:0]	CTL5[1:0]	CTL4[1:0]	CTL3[1:0]	CTL2[1:0]	CTL1[1:0]	CTL0[1:0]
rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	CTL15[1:0]	Pin 15 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
29:28	CTL14[1:0]	Pin 14 configuration bits These bits are set and cleared by software.

		Refer to CTL0[1:0] description
27:26	CTL13[1:0]	Pin 13 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
25:24	CTL12[1:0]	Pin 12 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
23:22	CTL11[1:0]	Pin 11 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
21:20	CTL10[1:0]	Pin 10 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
19:18	CTL9[1:0]	Pin 9 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
17:16	CTL8[1:0]	Pin 8 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
15:14	CTL7[1:0]	Pin 7 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
13:12	CTL6[1:0]	Pin 6 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
11:10	CTL5[1:0]	Pin 5 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
9:8	CTL4[1:0]	Pin 4 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
7:6	CTL3[1:0]	Pin 3 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
5:4	CTL2[1:0]	Pin 2 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description

3:2	CTL1[1:0]	Pin 1 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
1:0	CTL0[1:0]	Pin 0 configuration bits These bits are set and cleared by software. 00: Input mode (reset value) 01: GPIO output mode 10: Alternate function mode 11: Analog mode

8.4.2. Port output mode register (GPIOx_OMODE, x=A, B, F)

Address offset: 0x04

GD32E252xx:

Reset value: 0x0000 0000

GD32E251xx:

Reset value: 0x0000 FFFF for port A; 0x0000 07FF for port B; 0x0000 000B for port F.

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
OM15	OM14	OM13	OM12	OM11	OM10	OM9	OM8	OM7	OM6	OM5	OM4	OM3	OM2	OM1	OM0
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	OM15	Pin 15 output mode bit These bits are set and cleared by software. Refer to OM0 description
14	OM14	Pin 14 output mode bit These bits are set and cleared by software. Refer to OM0 description
13	OM13	Pin 13 output mode bit These bits are set and cleared by software. Refer to OM0 description
12	OM12	Pin 12 output mode bit These bits are set and cleared by software. Refer to OM0 description
11	OM11	Pin 11 output mode bit

		These bits are set and cleared by software. Refer to OM0 description
10	OM10	Pin 10 output mode bit These bits are set and cleared by software. Refer to OM0 description
9	OM9	Pin 9 output mode bit These bits are set and cleared by software. Refer to OM0 description
8	OM8	Pin 8 output mode bit These bits are set and cleared by software. Refer to OM0 description
7	OM7	Pin 7 output mode bit These bits are set and cleared by software. Refer to OM0 description
6	OM6	Pin 6 output mode bit These bits are set and cleared by software. Refer to OM0 description
5	OM5	Pin 5 output mode bit These bits are set and cleared by software. Refer to OM0 description
4	OM4	Pin 4 output mode bit These bits are set and cleared by software. Refer to OM0 description
3	OM3	Pin 3 output mode bit These bits are set and cleared by software. Refer to OM0 description
2	OM2	Pin 2 output mode bit These bits are set and cleared by software. Refer to OM0 description
1	OM1	Pin 1 output mode bit These bits are set and cleared by software. Refer to OM0 description
0	OM0	Pin 0 output mode bit These bits are set and cleared by software. 0: Output push-pull mode 1: Output open-drain mode

8.4.3. Port output speed register (GPIOx_OSPD, x=A, B, F)

Address offset: 0x08

GD32E252xx：

Reset value: 0x0C00 0000 for port A; 0x0000 0000 for other.

GD32E251xx:

Reset value: 0x00C0 0000 for port B; 0x0000 0000 for other.

Note: This register can be used without enabling the GPIO clock.

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
OSPD15[1:0]	OSPD14[1:0]	OSPD13[1:0]	OSPD12[1:0]	OSPD11[1:0]	OSPD10[1:0]	OSPD9[1:0]	OSPD8[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
OSPD7[1:0]	OSPD6[1:0]	OSPD5[1:0]	OSPD4[1:0]	OSPD3[1:0]	OSPD2[1:0]	OSPD1[1:0]	OSPD0[1:0]
rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	OSPD15[1:0]	Pin 15 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
29:28	OSPD14[1:0]	Pin 14 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
27:26	OSPD13[1:0]	Pin 13 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
25:24	OSPD12[1:0]	Pin 12 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
23:22	OSPD11[1:0]	Pin 11 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
21:20	OSPD10[1:0]	Pin 10 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
19:18	OSPD9[1:0]	Pin 9 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
17:16	OSPD8[1:0]	Pin 8 output max speed bits

		These bits are set and cleared by software. Refer to OSPD0[1:0] description
15:14	OSPD7[1:0]	Pin 7 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
13:12	OSPD6[1:0]	Pin 6 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
11:10	OSPD5[1:0]	Pin 5 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
9:8	OSPD4[1:0]	Pin 4 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
7:6	OSPD3[1:0]	Pin 3 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
5:4	OSPD2[1:0]	Pin 2 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
3:2	OSPD1[1:0]	Pin 1 output max speed bits These bits are set and cleared by software. Refer to OSPD0[1:0] description
1:0	OSPD0[1:0]	Pin 0 output max speed bits These bits are set and cleared by software. x0: Output max speed 2M (reset value) 01: Output max speed 10M 11: Output max speed 50M

8.4.4. Port pull-up/down register (GPIOx_PUD, x=A, B, F)

Address offset: 0x0C

GD32E252xx:

Reset value: 0x2400 0000 for A; 0x0000 0000 for others.

GD32E251xx:

Reset value: 0x5555 5555 for port A; 0x0055 5555 for port B; when NRST_MDSEL_VALUE is 1, 0x0000 0055 for port F, when NRST_MDSEL_VALUE is 0, 0x0000 0065 for port F.

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
PUD15[1:0]	PUD14[1:0]	PUD13[1:0]	PUD12[1:0]	PUD11[1:0]	PUD10[1:0]	PUD9[1:0]	PUD8[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
PUD7[1:0]	PUD6[1:0]	PUD5[1:0]	PUD4[1:0]	PUD3[1:0]	PUD2[1:0]	PUD1[1:0]	PUD0[1:0]
rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	PUD15[1:0]	Pin 15 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
29:28	PUD14[1:0]	Pin 14 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
27:26	PUD13[1:0]	Pin 13 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
25:24	PUD12[1:0]	Pin 12 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
23:22	PUD11[1:0]	Pin 11 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
21:20	PUD10[1:0]	Pin 10 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
19:18	PUD9[1:0]	Pin 9 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
17:16	PUD8[1:0]	Pin 8 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
15:14	PUD7[1:0]	Pin 7 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
13:12	PUD6[1:0]	Pin 6 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description

11:10	PUD5[1:0]	Pin 5 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
9:8	PUD4[1:0]	Pin 4 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
7:6	PUD3[1:0]	Pin 3 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
5:4	PUD2[1:0]	Pin 2 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
3:2	PUD1[1:0]	Pin 1 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
1:0	PUD0[1:0]	Pin 0 pull-up or pull-down bits These bits are set and cleared by software. 00: Floating mode, no pull-up and pull-down (reset value) 01: With pull-up mode 10: With pull-down mode 11: Reserved

8.4.5. Port input status register (GPIOx_ISTAT, x=A, B, F)

Address offset: 0x10

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	ISTAT9	ISTAT8	ISTAT7	ISTAT6	ISTAT5	ISTAT4	ISTAT3	ISTAT2	ISTAT1	ISTAT0
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

8.4.6. Port output control register (GPIOx_OCTL, x=A, B, F)

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
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 output low 1: Pin output high **Note:** for OTC set/reset, ODR bits can be set and reset or flipped respectively by writing to the GPIOx_BSRR/GPIOX_BRR/GPIOX_BTGR register

8.4.7. Port bit operate register (GPIOx_BOP, x=A, B, F)

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
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	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
15:0	BOPy	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

Note: BOPy has a higher priority than CRy

8.4.8. Port configuration lock register (GPIOx_LOCK, x=A, B, F)

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															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 key It can only be set by using the lock key writing sequence. And is always readable. 0: GPIOx_LOCK register and the port configuration are not locked 1: GPIOx_LOCK register is locked until an MCU reset LOCK key writing sequence: Write 1->Write 0->Write 1-> Read 0-> Read 1 **Note:** The value of LKy(y=0..15) 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: Port configuration not locked 1: Port configuration locked

8.4.9. Alternate function selected register 0 (GPIOx_AFSEL0, x=A / B)

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
SEL7[3:0]				SEL6[3:0]				SEL5[3:0]				SEL4[3:0]			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL3[3:0]				SEL2[3:0]				SEL1[3:0]				SEL0[3:0]			
rw				rw				rw				rw			

Bits	Fields	Descriptions
31:28	SEL7[3:0]	Pin 7 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
27:24	SEL6[3:0]	Pin 6 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
23:20	SEL5[3:0]	Pin 5 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
19:16	SEL4[3:0]	Pin 4 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
15:12	SEL3[3:0]	Pin 3 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
11:8	SEL2[3:0]	Pin 2 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
7:4	SEL1[3:0]	Pin 1 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
3:0	SEL0[3:0]	Pin 0 alternate function selected These bits are set and cleared by software. 0000: AF0 selected (reset value) 0001: AF1 selected 0010: AF2 selected 0011: AF3 selected 0100: AF4 selected 0101: AF5 selected 0110: AF6 selected 0111: AF7 selected 0111: AF8 selected 0111: AF9 selected …

1111: AF15 selected

8.4.10. Alternate function selected register 1 (GPIOx_AFSEL1, x=A / B)

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
SEL15[3:0]				SEL14[3:0]				SEL13[3:0]				SEL12[3:0]			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL11[3:0]				SEL10[3:0]				SEL9[3:0]				SEL8[3:0]			
rw				rw				rw				rw			

Bits	Fields	Descriptions
31:28	SEL15[3:0]	Pin 15 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
27:24	SEL14[3:0]	Pin 14 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
23:20	SEL13[3:0]	Pin 13 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
19:16	SEL12[3:0]	Pin 12 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
15:12	SEL11[3:0]	Pin 1 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
11:8	SEL10[3:0]	Pin 10 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
7:4	SEL9[3:0]	Pin 9 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
3:0	SEL8[3:0]	Pin 8 alternate function selected These bits are set and cleared by software. 0000: AF0 selected (reset value)

0001: AF1 selected
0010: AF2 selected
0011: AF3 selected
0100: AF4 selected
0101: AF5 selected
0110: AF6 selected
0111: AF7 selected
1000: AF8 selected
1001: AF9 selected
…
1111: AF15 selected

8.4.11. Bit clear register (GPIOx_BC, x=A, B, F)

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
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

8.4.12. Port bit toggle register (GPIOx_TG, x=A, B, F)

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
TG15	TG14	TG13	TG12	TG11	TG10	TG9	TG8	TG7	TG6	TG5	TG4	TG3	TG2	TG1	TG0

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	TGy	Port toggle bit y(y=0..15) These bits are set and cleared by software. 0: No action on the corresponding OCTLy bit 1: Toggle the corresponding OCTLy bit

8.4.13. Crossbar enable register (GPIO_CBREN)

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												CBAREN	CUSRT1 CTSEN	CUSRT1 RTSEN	CUART1 EN
												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	CTIMER1 3EN	CTIMER2 ETIEN	CTIMER1 ETIEN	CTIMER0 ETIEN	CTIMER0MOD			CSYSCK EN	Reserved		CCMP0A EN	CCMP0E N	CSMBEN	CSPI0EN	CUSART 0EN
rw	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	CBAREN	Crossbar Enable These bits are set and cleared by software. 0: Crossbar disabled 1: Crossbar enabled
18	CUART1CTSEN	Crossbar UART1 CTS Input Enable These bits are set and cleared by software. 0: UART1 CTS unavailable at Port pin 1: UART1 CTS routed to Port pin
17	CUART1RTSEN	Crossbar UART1 RTS Output Enable These bits are set and cleared by software. 0: UART1 RTS unavailable at Port pin 1: UART1 RTS routed to Port pin
16	CUART1EN	Crossbar UART1 I/O Enable These bits are set and cleared by software. 0: UART1 I / O unavailable at Port pin

		1: UART1 TX RX routed to Port pins
15	Reserved	Must be kept at reset value.
14	CTIMER13EN	Crossbar TIMER13 Channel0 enable These bits are set and cleared by software. 0: TIMER13 CH0 unavailable at Port pin 1: TIMER13 CH0 routed to Port pin
13	CTIMER2ETIEN	Crossbar TIMER2 External triggle enable These bits are set and cleared by software. 0: TIMER2_ETR unavailable at Port pin 1: TIMER2_ETR routed to Port pin
12	CTIMER1ETIEN	Crossbar TIMER1 External Channel0 triggle enable These bits are set and cleared by software. 0: TIMER1_ETR Channel0 unavailable at Port pin 1: TIMER1_ETR Channel0 routed to Port pin
11	CTIMER0ETIEN	Crossbar TIMER0 External Counter Input Enable These bits are set and cleared by software. 0: TIMER0_ETR unavailable at Port pin 1: TIMER0_ETR routed to Port pin
10:8	CTIMER0MOD	Crossbar TIMER0 Module I/O Enable These bits are set and cleared by software. 000: All TIMER0 I/O unavailable at Port pins 001: CH0 routed to Port pin 010: CH0, CH1 routed to Port pins 011: CH0, CH1, CH2 routed to Port pins 100: CH0, CH1, CH2, CH3 routed to Port pins 101: CH0, CH1, CH2, CH3, CH4 routed to Port pins 110: CH0, CH1, CH2, CH3, CH4, CH5 routed to Port pins
7	CSYSCKEN	Crossbar SYSCLK Output Enable. These bits are set and cleared by software. 0: SYSCLK unavailable at Port pin 1: SYSCLK output routed to Port pin
6:5	Reserved	Must be kept at reset value
4	CCMP0AEN	Crossbar Comparator0 Asynchronous Output Enable. These bits are set and cleared by software 0: Asynchronous CMP0 unavailable at Port pin 1: Asynchronous CMP0 routed to Port pin
3	CCMP0EN	Crossbar Comparator0 Output Enable. These bits are set and cleared by software.

		0: CMP0 unavailable at Port pin 1: CMP0 routed to Port pin
2	CSMBEN	Crossbar SMBus I / O Enable. These bits are set and cleared by software. 0: SMBus I / O unavailable at Port pins 1: SMBus I / O routed to Port pins
1	CSPI0EN	Crossbar SPI0 I / O Enable These bits are set and cleared by software. 0: SPI0 I/O unavailable at Port pins 1: SPI0 I/O routed to Port pins
0	CUSART0EN	Crossbar USART0 I/O Enable. These bits are set and cleared by software. 0: USART0 I / O unavailable at Port pin 1: USART0 TX RX routed to Port pins

8.4.14. Digitial function enable register (GPIO_DFEN)

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
Reserved											CLA3PINEN	CLA2PINEN	CLA1PINEN	CLA0PINEN	I2C0PINEN
											rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	CLA3PINEN	CLA3 Pin Output Enable These bits are set and cleared by software. 0: Disable asynchronous output to the selected GPIO pin 1: Enable asynchronous output to the selected GPIO pin
3	CLA2PINEN	CLA2 Pin Output Enable These bits are set and cleared by software. 0: Disable asynchronous output to the selected GPIO pin 1: Enable asynchronous output to the selected GPIO pin
2	CLA1PINEN	CLA1 Pin Output Enable

These bits are set and cleared by software.
0: Disable asynchronous output to the selected GPIO pin
1: Enable asynchronous output to the selected GPIO pin

1	CLA0PINEN	CLA0 Pin Output Enable These bits are set and cleared by software. 0: Disable asynchronous output to the selected GPIO pin 1: Enable asynchronous output to the selected GPIO pin
0	I2C0PINEN	I2C0 Pin mode enable These bits are set and cleared by software. 0: Disable I2C0 to the selected GPIO pin 1: Enable I2C0 to the selected GPIO pin **Note:** AF function is not enabled, this bit enables I2C0 input and output in pin.

8.4.15. Crossbar skip register (GPIO_SKIP)

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													SKIP18	SKIP17	SKIP16
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SKIP15	SKIP14	SKIP13	SKIP12	SKIP11	SKIP10	SKIP9	SKIP8	SKIP7	SKIP6	SKIP5	SKIP4	SKIP3	SKIP2	SKIP1	SKIP0
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	SKIP18	SKIP18 value These bits are set and cleared by software. 0: PA0 (for GD32E251KX) / PA4 (for GD32E251EX) pin is not skipped by the crossbar 1: PA0 (for GD32E251KX) / PA4 (for GD32E251EX) pin is skipped by the crossbar
17	SKIP17	SKIP17 value These bits are set and cleared by software. 0: PB0 (for GD32E251KX) / PA5 (for GD32E251EX) pin is not skipped by the crossbar 1: PB0 (for GD32E251KX) / PA5 (for GD32E251EX) pin is skipped by the crossbar
16	SKIP16	SKIP16 value

These bits are set and cleared by software.
0: PB1 (for GD32E251KX) / PA6 (for GD32E251EX) pin is not skipped by the crossbar
1: PB1 (for GD32E251KX) / PA6 (for GD32E251EX) pin is skipped by the crossbar

15	SKIP15	SKIP15 value These bits are set and cleared by software. 0: PB10 (for GD32E251KX) / PA7 (for GD32E251EX) pin is not skipped by the crossbar 1: PB10 (for GD32E251KX) / PA7 (for GD32E251EX) pin is skipped by the crossbar
14	SKIP14	SKIP14 value These bits are set and cleared by software. 0: PB2 (only for GD32E251KX) pin is not skipped by the crossbar 1: PB2 (only for GD32E251KX) pin is skipped by the crossbar
13	SKIP13	SKIP13 value These bits are set and cleared by software. 0: PA8 (for GD32E251KX) / PA2 (for GD32E251EX) pin is not skipped by the crossbar 1: PA8 (for GD32E251KX) / PA2 (for GD32E251EX) pin is skipped by the crossbar
12	SKIP12	SKIP12 value These bits are set and cleared by software. 0: PA9 (for GD32E251KX) / PB0 (for GD32E251EX) pin is not skipped by the crossbar 1: PA9 (for GD32E251KX) / PB0 (for GD32E251EX) pin is skipped by the crossbar
11	SKIP11	SKIP11 value These bits are set and cleared by software. 0: PA10 (for GD32E251KX) / PB1 (for GD32E251EX) pin is not skipped by the crossbar 1: PA10 (for GD32E251KX) / PB1 (for GD32E251EX) pin is skipped by the crossbar
10	SKIP10	SKIP10 value These bits are set and cleared by software. 0: PA11 (for GD32E251KX) / PB10 (for GD32E251EX) pin is not skipped by the crossbar 1: PA11 (for GD32E251KX) / PB10 (for GD32E251EX) pin is skipped by the crossbar
9	SKIP9	SKIP9 value

		These bits are set and cleared by software. 0: PA12 (for GD32E251KX) / PA9 (for GD32E251EX) pin is not skipped by the crossbar 1: PA12 (for GD32E251KX) / PA9 (for GD32E251EX) pin is skipped by the crossbar
8	SKIP8	SKIP8 value These bits are set and cleared by software. 0: PA13 (for GD32E251KX) / PA10 (for GD32E251EX) pin is not skipped by the crossbar 1: PA13 (for GD32E251KX) / PA10 (for GD32E251EX) pin is skipped by the crossbar
7	SKIP7	SKIP7 value These bits are set and cleared by software. 0: PA14 (for GD32E251KX) / PA13 (for GD32E251EX) pin is not skipped by the crossbar 1: PA14 (for GD32E251KX) / PA13 (for GD32E251EX) pin is skipped by the crossbar
6	SKIP6	SKIP6 value These bits are set and cleared by software. 0: PA15 pin is not skipped by the crossbar 1: PA15 pin is skipped by the crossbar
5	SKIP5	SKIP5 value These bits are set and cleared by software. 0: PB3 pin is not skipped by the crossbar 1: PB3 pin is skipped by the crossbar
4	SKIP4	SKIP4 value These bits are set and cleared by software. 0: PB4 pin is not skipped by the crossbar 1: PB4 pin is skipped by the crossbar
3	SKIP3	SKIP3 value These bits are set and cleared by software. 0: PB5 pin is not skipped by the crossbar 1: PB5 pin is skipped by the crossbar
2	SKIP2	SKIP2 value These bits are set and cleared by software. 0: PF0 pin is not skipped by the crossbar 1: PF0 pin is skipped by the crossbar
1	SKIP1	SKIP1 value These bits are set and cleared by software.

		0: PB8 pin is not skipped by the crossbar 1: PB8 pin is skipped by the crossbar
0	SKIP0	SKIP0 value These bits are set and cleared by software. 0: PB9 pin is not skipped by the crossbar 1: PB9 pin is skipped by the crossbar **Note:** SKIP the correspondence between pin and crossbar see ***Figure 8-7. Crossbar decoder functional map***.

8.4.16. INT0 and INT1 control register (GPIO_INTCTL)

Address offset: 0x40

Reset value: 0x0000 0001

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								INT1PL	INT1SL[2:0]			INT0PL	INT0SL[2:0]		
								rw	rw			rw	rw		

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	INT1PL	INT1 Polarity. These bits are set and cleared by software 0: INT1 input is active low 1: INT1 input is active high
6:4	INT1SL[2:0]	INT1 Port Pin Selection. These bits are set and cleared by software. 000: Select PB9 001: Select PB8 010: Select PF0 011: Select PB5 100: Select PB4 101: Select PB3 110: Select PA15 111 Reserved
3	INT0PL	INT0 Polarity. These bits are set and cleared by software. 0: INT0 input is active low

		1: INT0 input is active high
2:0	INT0SL[2:0]	INT0 Port Pin Selection. These bits are set and cleared by software. 000: Select PB9 001: Select PB8 010: Select PF0 011: Select PB5 100: Select PB4 101: Select PB3 110: Select PA15 111: Reserved

8.4.17. Pin mask register (GPIO_MASK)

Address offset: 0x44

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										MSK21	MSK20	MSK19	MSK18	MSK17	MSK16
										rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSK15	MSK14	MSK13	MSK12	MSK11	MSK10	MSK9	MSK8	MSK7	MSK6	MSK5	MSK4	MSK3	MSK3	MSK1	MSK0
rw	rw	rw	rw	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	MSK21	MSK21 Value These bits are set and cleared by software. 0：PA3 (only for GD32E251KX) logic value is ignored and will not cause a port mismatch event 1：PA3 (only for GD32E251KX) logic value is compared to register MSK21
20	MSK20	MSK20 Value These bits are set and cleared by software. 0：PA2 (only for GD32E251KX) logic value is ignored and will not cause a port mismatch event 1：PA2 (only for GD32E251KX) logic value is compared to register MSK20
19	MSK19	MSK19 Value These bits are set and cleared by software. 0：PA1 (only for GD32E251KX) logic value is ignored and will not cause a port mismatch event

1：PA1 (only for GD32E251KX) logic value is compared to register MSK19

18	MSK18	MSK18 Value These bits are set and cleared by software. 0：PA0 (for GD32E251KX) / PA4 (for GD32E251EX) logic value is ignored and will not cause a port mismatch event 1：PA0 (for GD32E251KX) / PA4 (for GD32E251EX) logic value is compared to register MSK18
17	MSK17	MSK17 Value These bits are set and cleared by software. 0：PB0 (for GD32E251KX) / PA5 (for GD32E251EX) logic value is ignored and will not cause a port mismatch event 1：PB0 (for GD32E251KX) / PA5 (for GD32E251EX) logic value is compared to register MSK17
16	MSK16	MSK16 Value These bits are set and cleared by software. 0：PB1 (for GD32E251KX) / PA6 (for GD32E251EX) logic value is ignored and will not cause a port mismatch event 1：PB1 (for GD32E251KX) / PA6 (for GD32E251EX) logic value is compared to register MSK16
15	MSK15	MSK15 Value These bits are set and cleared by software. 0：PB10 (for GD32E251KX) / PA7 (for GD32E251EX) logic value is ignored and will not cause a port mismatch event 1：PB10 (for GD32E251KX) / PA7 (for GD32E251EX) logic value is compared to register MSK15
14	MSK14	MSK14 Value These bits are set and cleared by software. 0：PB2 (only for GD32E251KX) logic value is ignored and will not cause a port mismatch event 1：PB2 (only for GD32E251KX) logic value is compared to register MSK14
13	MSK13	MSK13 Value These bits are set and cleared by software. 0：PA8 (for GD32E251KX) / PA2 (for GD32E251EX) logic value is ignored and will not cause a port mismatch event 1：PA8 (for GD32E251KX) / PA2 (for GD32E251EX) logic value is compared to register MSK13
12	MSK12	MSK12 Value These bits are set and cleared by software. 0：PA9 (for GD32E251KX) / PB0 (for GD32E251EX) logic value is ignored and will not cause a port mismatch event

		1：PA9 (for GD32E251KX) / PB0 (for GD32E251EX) logic value is compared to register MSK12
11	MSK11	MSK11 Value These bits are set and cleared by software. 0：PA10 (for GD32E251KX) / PB1 (for GD32E251EX) logic value is ignored and will not cause a port mismatch event 1：PA10 (for GD32E251KX) / PB1 (for GD32E251EX) logic value is compared to register MSK11
10	MSK10	MSK10 Value These bits are set and cleared by software. 0：PA11 (for GD32E251KX) / PB10 (for GD32E251EX) logic value is ignored and will not cause a port mismatch event 1：PA11 (for GD32E251KX) / PB10 (for GD32E251EX) logic value is compared to register MSK10
9	MSK9	MSK9 Value These bits are set and cleared by software. 0：PA12 (for GD32E251KX) / PA9 (for GD32E251EX) logic value is ignored and will not cause a port mismatch event 1：PA12 (for GD32E251KX) / PA9 (for GD32E251EX) logic value is compared to register MSK9
8	MSK8	MSK8 Value These bits are set and cleared by software. 0：PA13 (for GD32E251KX) / PA10 (for GD32E251EX) logic value is ignored and will not cause a port mismatch event 1：PA13 (for GD32E251KX) / PA10 (for GD32E251EX) logic value is compared to register MSK8
7	MSK7	MSK7 Value These bits are set and cleared by software. 0：PA14 (for GD32E251KX) / PA13 (for GD32E251EX) logic value is ignored and will not cause a port mismatch event 1：PA14 (for GD32E251KX)/PA13 (for GD32E251EX) logic value is compared to register MSK7
6	MSK6	MSK6 Value These bits are set and cleared by software. 0：PA15 logic value is ignored and will not cause a port mismatch event 1：PA15 logic value is compared to register MSK6
5	MSK5	MSK5 Value These bits are set and cleared by software. 0：PB3 logic value is ignored and will not cause a port mismatch event

		1：PB3 logic value is compared to register MSK5
4	MSK4	MSK4 Value These bits are set and cleared by software. 0：PB4 logic value is ignored and will not cause a port mismatch event 1：PB4 logic value is compared to register MSK4
3	MSK3	MSK3 Value These bits are set and cleared by software. 0：PB5 logic value is ignored and will not cause a port mismatch event 1：PB5 logic value is compared to register MSK3
2	MSK2	MSK2 Value These bits are set and cleared by software. 0：PF0 logic value is ignored and will not cause a port mismatch event 1：PF0 logic value is compared to register MSK2
1	MSK1	MSK1 Value These bits are set and cleared by software. 0：PB8 logic value is ignored and will not cause a port mismatch event 1：PB8 logic value is compared to register MSK1
0	MSK0	MSK0 Value These bits are set and cleared by software. 0：PB9 logic value is ignored and will not cause a port mismatch event 1：PB9 logic value is compared to register MSK0

8.4.18. Pin match register (GPIO_MATCH)

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										MATCH21	MATCH20	MATCH19	MATCH18	MATCH17	MATCH16
										rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MATCH15	MATCH14	MATCH13	MATCH12	MATCH11	MATCH10	MATCH9	MATCH8	MATCH7	MATCH6	MATCH5	MATCH4	MATCH3	MATCH2	MATCH1	MATCH0
rw	rw	rw	rw	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	MATCH21	MATCH21 Value

These bits are set and cleared by software.
0：PA3 (only for GD32E251KX) logic value is compared with logic LOW
1：PA3 (only for GD32E251KX) logic value is compared with logic HIGH

20	MATCH20	MATCH20 Value These bits are set and cleared by software. 0：PA2 (only for GD32E251KX) logic value is compared with logic LOW 1：PA2 (only for GD32E251KX) logic value is compared with logic HIGH
19	MATCH19	MATCH19 Value These bits are set and cleared by software. 0：PA1 (only for GD32E251KX) logic value is compared with logic LOW 1：PA1 (only for GD32E251KX) logic value is compared with logic HIGH
18	MATCH18	MATCH18 Value These bits are set and cleared by software. 0：PA0 (for GD32E251KX) / PA4 (for GD32E251EX) logic value is compared with logic LOW 1：PA0 (for GD32E251KX) / PA4 (for GD32E251EX) logic value is compared with logic HIGH
17	MATCH17	MATCH17 Value These bits are set and cleared by software. 0：PB0 (for GD32E251KX) / PA5 (for GD32E251EX) logic value is compared with logic LOW 1：PB0 (for GD32E251KX) / PA5 (for GD32E251EX) logic value is compared with logic HIGH
16	MATCH16	MATCH16 Value These bits are set and cleared by software. 0：PB1 (for GD32E251KX) / PA6 (for GD32E251EX) logic value is compared with logic LOW 1：PB1 (for GD32E251KX) / PA6 (for GD32E251EX) logic value is compared with logic HIGH
15	MATCH15	MAT15 Value These bits are set and cleared by software. 0：PB10 (for GD32E251KX) / PA7 (for GD32E251EX) logic value is compared with logic LOW 1：PB10 (for GD32E251KX) / PA7 (for GD32E251EX) logic value is compared with logic HIGH
14	MATCH14	MATCH14 Value These bits are set and cleared by software. 0：PB2 (only for GD32E251KX) logic value is compared with logic LOW 1：PB2 (only for GD32E251KX) logic value is compared with logic HIGH

13	MATCH13	MATCH13 Value These bits are set and cleared by software. 0：PA8 (for GD32E251KX) / PA2 (for GD32E251EX) logic value is compared with logic LOW 1：PA8 (for GD32E251KX) / PA2 (for GD32E251EX) logic value is compared with logic HIGH
12	MATCH12	MATCH12 Value These bits are set and cleared by software. 0：PA9 (for GD32E251KX) / PB0 (for GD32E251EX) logic value is compared with logic LOW 1：PA9 (for GD32E251KX) / PB0 (for GD32E251EX) logic value is compared with logic HIGH
11	MATCH11	MATCH11 Value These bits are set and cleared by software. 0：PA10 (for GD32E251KX) / PB1 (for GD32E251EX) logic value is compared with logic LOW 1：PA10 (for GD32E251KX) / PB1 (for GD32E251EX) logic value is compared with logic HIGH
10	MATCH10	MATCH10 Value These bits are set and cleared by software. 0：PA11 (for GD32E251KX) / PB10 (for GD32E251EX) logic value is compared with logic LOW 1：PA11 (for GD32E251KX) / PB10 (for GD32E251EX) logic value is compared with logic HIGH
9	MATCH9	MATCH9 Value. These bits are set and cleared by software. 0：PA12 (for GD32E251KX) / PA9 (for GD32E251EX) logic value is compared with logic LOW 1：PA12 (for GD32E251KX) / PA9 (for GD32E251EX) logic value is compared with logic HIGH
8	MATCH8	MATCH8 Value These bits are set and cleared by software. 0：PA13 (for GD32E251KX) / PA10 (for GD32E251EX) logic value is compared with logic LOW 1：PA13 (for GD32E251KX)/PA10 (for GD32E251EX) logic value is compared with logic HIGH
7	MATCH7	MATCH7 Value These bits are set and cleared by software. 0：PA14 (for GD32E251KX) / PA13 (for GD32E251EX) logic value is compared with logic LOW

1：PA14 (for GD32E251KX) / PA13 (for GD32E251EX) logic value is compared with logic HIGH

6	MATCH6	MATCH6 Value These bits are set and cleared by software. 0：PA15 logic value is compared with logic LOW 1：PA15 logic value is compared with logic HIGH
5	MATCH5	MATCH5 Value These bits are set and cleared by software. 0：PB3 logic value is compared with logic LOW 1：PB3 logic value is compared with logic HIGH
4	MATCH4	MATCH4 Value These bits are set and cleared by software. 0：PB4 logic value is compared with logic LOW 1：PB4 logic value is compared with logic HIGH
3	MATCH3	MATCH3 Value These bits are set and cleared by software. 0：PB5 logic value is compared with logic LOW 1：PB5 logic value is compared with logic HIGH
2	MATCH2	MATCH2 Value These bits are set and cleared by software. 0：PF0 logic value is compared with logic LOW 1：PF0 logic value is compared with logic HIGH
1	MATCH1	MATCH1 Value These bits are set and cleared by software. 0：PB8 logic value is compared with logic LOW 1：PB8 logic value is compared with logic HIGH
0	MATCH0	MATCH0 Value These bits are set and cleared by software. 0：PB9 logic value is compared with logic LOW 1：PB9 logic value is compared with logic HIGH

8.4.19. Pin match flag register (GPIO_MATCHF)

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									MATCHIE	MATCHF 21	MATCHF 20	MATCHF 19	MATCHF 18	MATCHF 17	MATCHF 16

									rw	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MATCHF 15	MATCHF 14	MATCHF 13	MATCHF 12	MATCHF 11	MATCHF 10	MATCHF 9	MATCHF 8	MATCHF 7	MATCHF 6	MATCHF 5	MATCHF 4	MATCH 3	MATCH 2	MATCHF 1	MATCHF 0
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
31:23	Reserved	Must be kept at reset value.
22	MATCHIE	1: enable port not match interrupt 0: disable port not match interrupt
21	MATCHF21	MATCHF21 Value Port not match set 1 by hardware Clear PA3 (only for GD32E251KX) by software can clear this bit by writing 1.
20	MATCHF20	MATCHF20 Value Port not match set 1 by hardware Clear PA2 (only for GD32E251KX) by software can clear this bit by writing 1.
19	MATCHF19	MATCHFG19 Value Port not match set 1 by hardware Clear PA1 (only for GD32E251KX) by software can clear this bit by writing 1.
18	MATCHF18	MATCHF18 Value Port not match set 1 by hardware Clear PA0 (for GD32E251KX) / PA4 (for GD32E251EX) by software can clear this bit by writing 1.
17	MATCHF17	MATCHF17 Value Port not match set 1 by hardware Clear PB0 (for GD32E251KX) / PA5 (for GD32E251EX) by software can clear this bit by writing 1.
16	MATCHF16	MATCHF16 Value Port not match set 1 by hardware Clear PB1 (for GD32E251KX) / PA6 (for GD32E251EX) by software can clear this bit by writing 1.
15	MATCHF15	MATCHF15 Value Port not match set 1 by hardware Clear PB10 (for GD32E251KX) / PA7 (for GD32E251EX) by software can clear this bit by writing 1.
14	MATCHF14	MATCHF14 Value Port not match set 1 by hardware

		Clear PB2 (only for GD32E251KX) by software can clear this bit by writing 1.
13	MATCHF13	MATCHF13 Value Port not match set 1 by hardware Clear PA8 (for GD32E251KX) / PA2 (for GD32E251EX) by software clear this bit by writing 1.
12	MATCHF12	MATCHF12 Value Port not match set 1 by hardware Clear PA9 (for GD32E251KX) / PB0 (for GD32E251EX) by software can clear this bit by writing 1.
11	MATCHF11	MATCHF11 Value Port not match set 1 by hardware Clear PA10 (for GD32E251KX) / PB1 (for GD32E251EX) by software can clear this bit by writing 1.
10	MATCHF10	MATCHF10 Value Port not match set 1 by hardware Clear PA11 (for GD32E251KX) / PB10 (for GD32E251EX) by software can clear this bit by writing 1.
9	MATCHF9	MATCHF9 Value Port not match set 1 by hardware Clear PA12 (for GD32E251KX) / PA9 (for GD32E251EX) by software can clear this bit by writing 1.
8	MATCHF8	MATCHF8 Value. Port not match set 1 by hardware Clear PA13 (for GD32E251KX) / PA10 (for GD32E251EX) by software can clear this bit by writing 1.
7	MATCHF7	MATCHF7 Value Port not match set 1 by hardware Clear PA14 (for GD32E251KX) / PA13 (for GD32E251EX) by software can clear this bit by writing 1.
6	MATCHF6	MATCHF6 Value Port not match set 1 by hardware Clear PA15 by software can clear this bit by writing 1.
5	MATCHF5	MATCHF5 Value Port not match set 1 by hardware Clear PB3 by software can clear this bit by writing 1.
4	MATCHF4	MATCHF4 Value Port not match set 1 by hardware Clear PB4 by software can clear this bit by writing 1.

3	MATCHF3	MATCHF3 Value Port not match set 1 by hardware Clear PB5 by software can clear this bit by writing 1.
2	MATCHF2	MATCHF2 Value Port not match set 1 by hardware Clear PF0 by software can clear this bit by writing 1.
1	MATCHF1	MATCHF1 Value. Port not match set 1 by hardware Clear PB8 by software can clear this bit by writing 1.
0	MATCHF0	MATCHF0 Value. Port not match set 1 by hardware Clear PB9 by software can clear this bit by writing 1.

9. Timer (TIMERx)

Table 9-1. Timers (TIMERx) are devided into six sorts

TIMER	TIMER0	TIMER1/2	TIMER13	TIMER14	TIMER15/16	TIMER5/6
TYPE	Advanced	General-L0	General-L2	General-L3	General-L4	Basic
Prescaler	16-bit	16-bit	16-bit	16-bit	16-bit	16-bit
Counter	16-bit	32-bit(TIMER1) 16-bit(TIMER2)	16-bit	16-bit	16-bit	16-bit
Count mode	UP, DOWN, Center-aligned	UP, DOWN, Center-aligned	UP ONLY	UP ONLY	UP ONLY	UP ONLY
Repetition	●	×	×	●	●	×
CH Capture/ Compare	6	4	1	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:** TIMER14_TRGO **ITI1:** 0 **ITI2:** TIMER2_TRGO **ITI3:** 0

(2) TIMER1 **ITI0:** TIMER0_TRGO **ITI1:** TIMER14_TRGO **ITI2:** TIMER2_TRGO **IT3: 0**
TIMER2 **ITI0:** TIMER0_TRGO **ITI1:** 0 **ITI2:** TIMER14_TRGO **ITI3:** 0

(3) TIMER14 **ITI0:** 0 **ITI1:** TIMER2_TRGO **ITI2:** 0 **ITI3:** 0

(4) Only update events will generate a DMA request. TIMER5/6 do not have DMAS bit (DMA request source selection).

9.1. Advanced timer (TIMERx, x=0)

9.1.1. Overview

The advanced timer module (TIMER0) is a six-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.

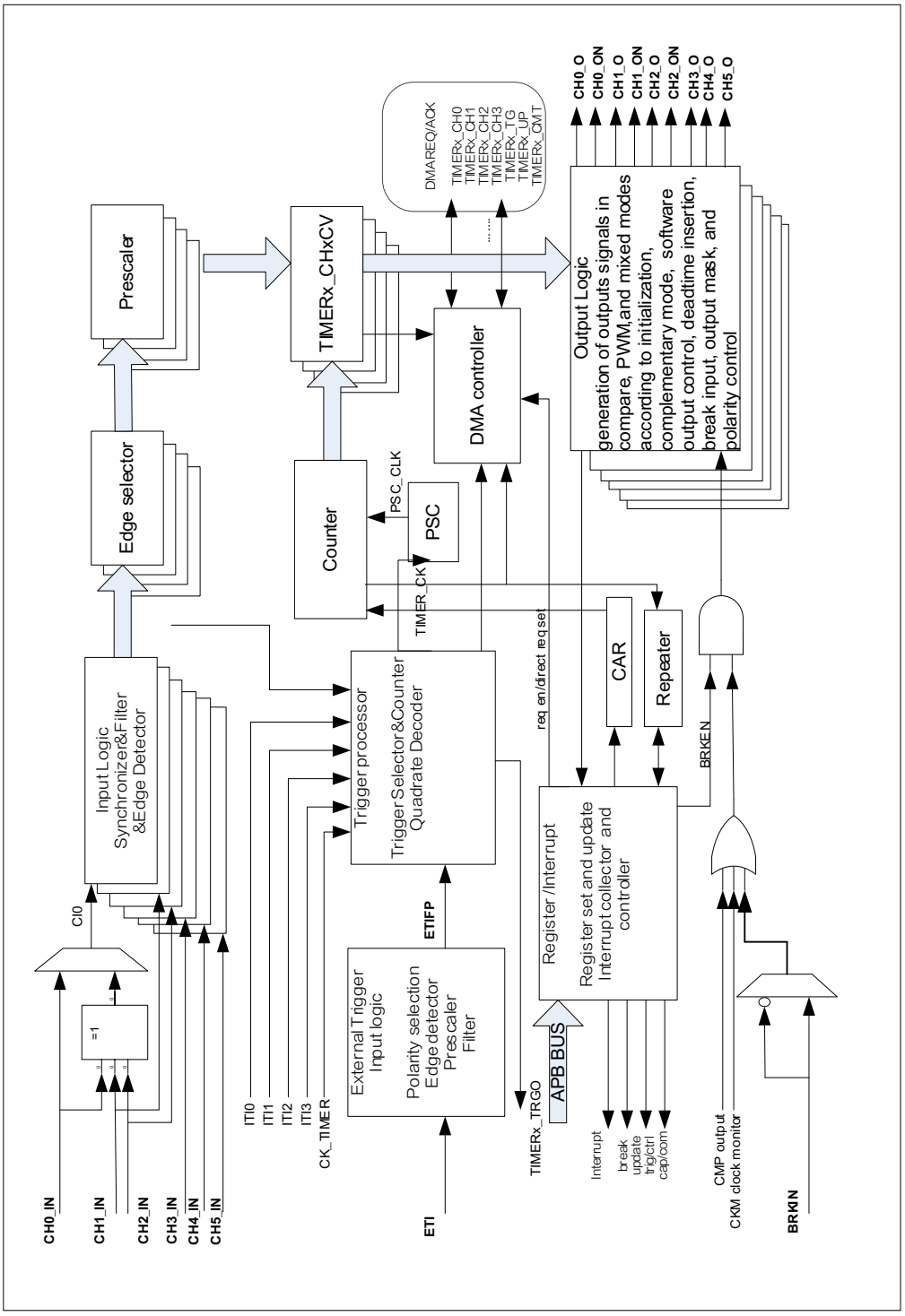
9.1.2. Characteristics

- Total channel num: 6.
- 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, commutation 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.

9.1.3. Block diagram

Figure 9-1. Advanced timer block diagram provides details of the internal configuration of the advanced timer.

Figure 9-1. Advanced timer block diagram



Note: Only TIMER0 supports CMP output as the break source input.

9.1.4. 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]).

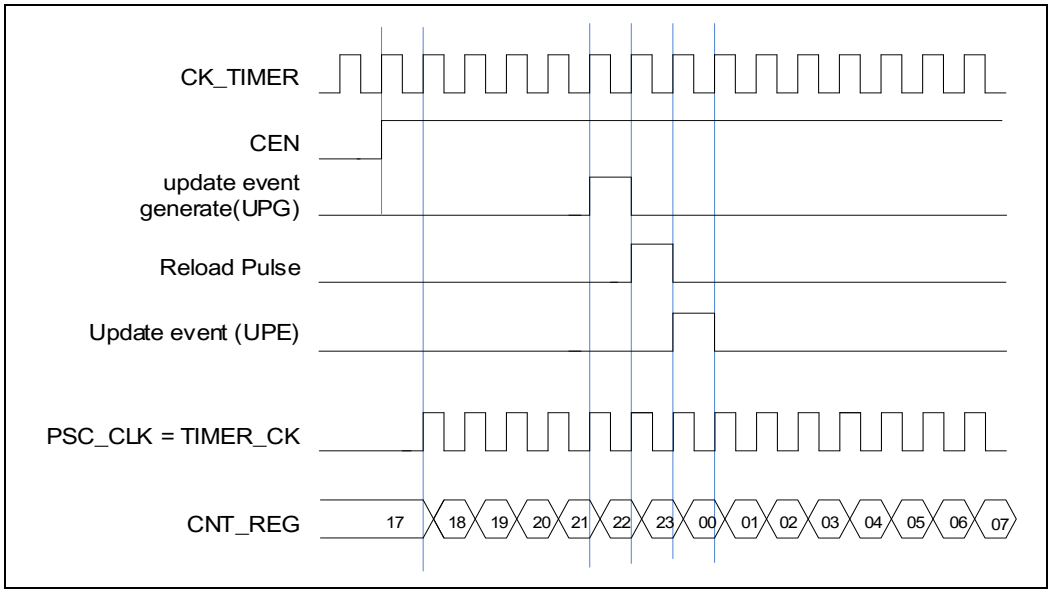
- SMC [2:0] == 3'b000. Internal clock CK_TIMER is selected as timer clock source which is from module RCU.

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 9-2. 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.

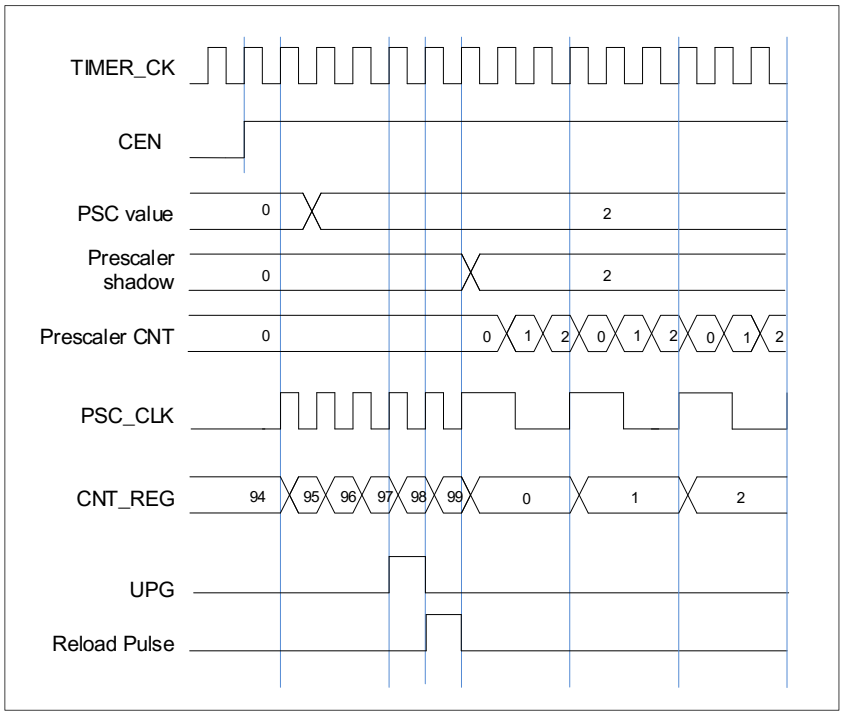
- SMC1== 1'b1 (external clock mode 1). External input ETI is selected as timer clock source (ETI)

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 9-3. 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, counter auto reload register, prescaler register) are updated.

Figure 9-4. Timing chart of up counting mode, PSC=0/2 and *Figure 9-5. Timing chart of up counting mode, change TIMERx_CAR on the go* show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 9-4. Timing chart of up counting mode, PSC=0/2

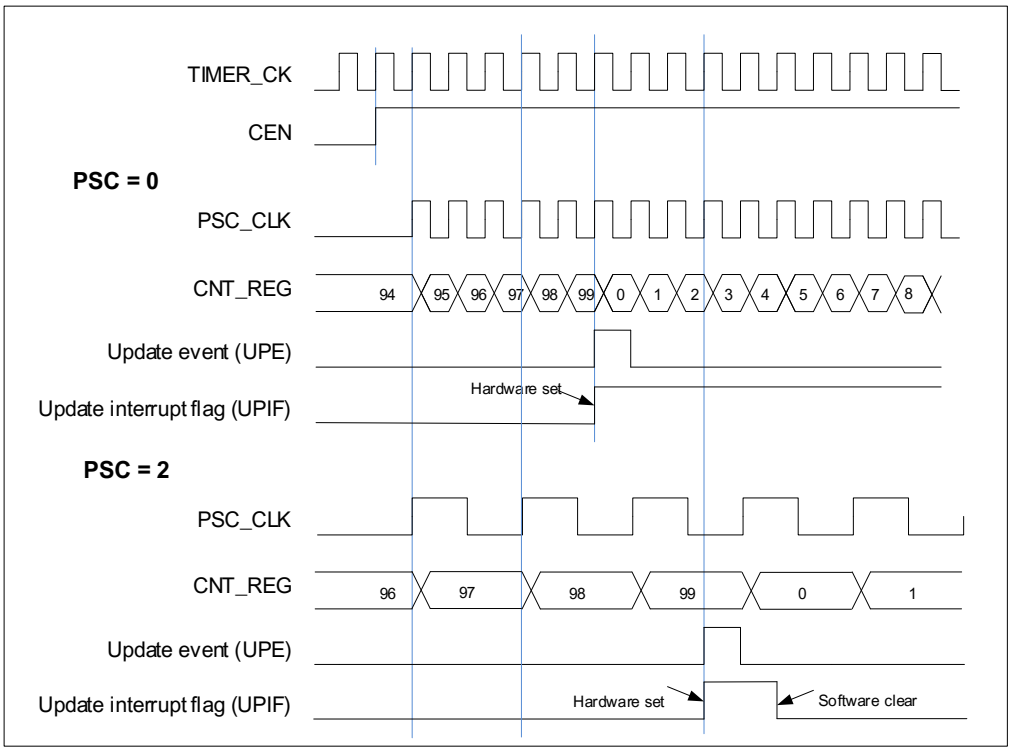
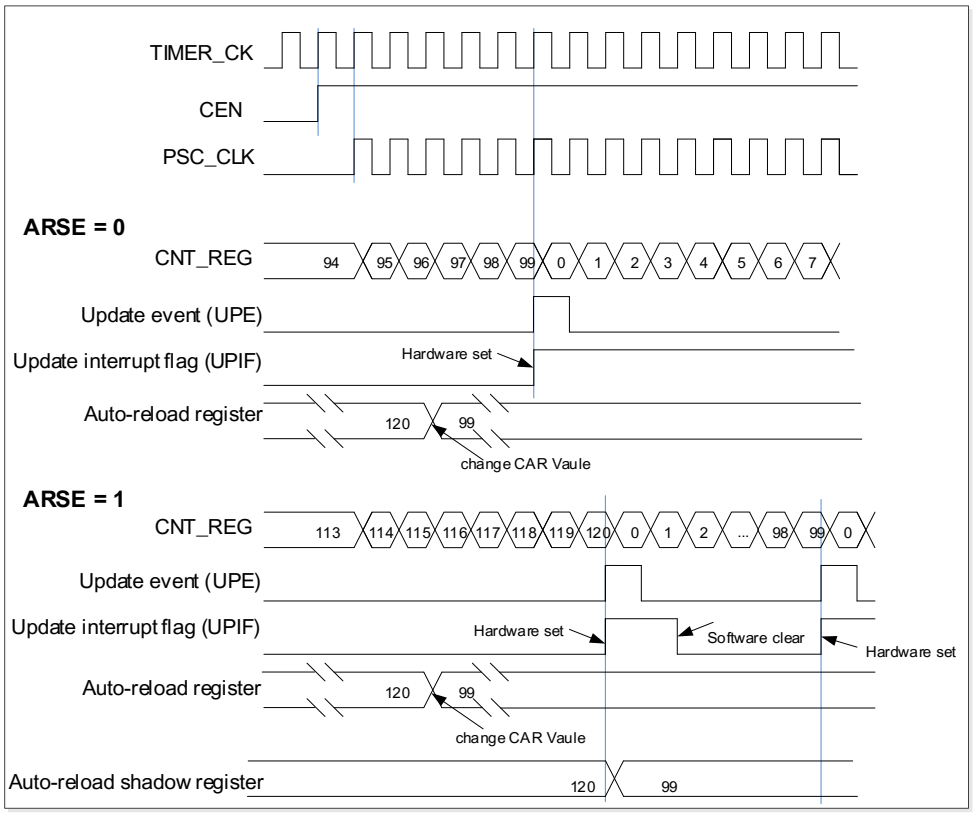


Figure 9-5. Timing chart of up counting mode, change TIMERx_CAR on the go



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 shadow registers (repetition counter, counter auto reload register, prescaler register) are updated.

Figure 9-6. Timing chart of down counting mode, PSC=0/2 and *Figure 9-7. Timing chart of down counting mode, change TIMERx_CAR on the go* show some examples of the counter behavior in different clock frequencies when TIMERx_CAR=0x99.

Figure 9-6. Timing chart of down counting mode, PSC=0/2

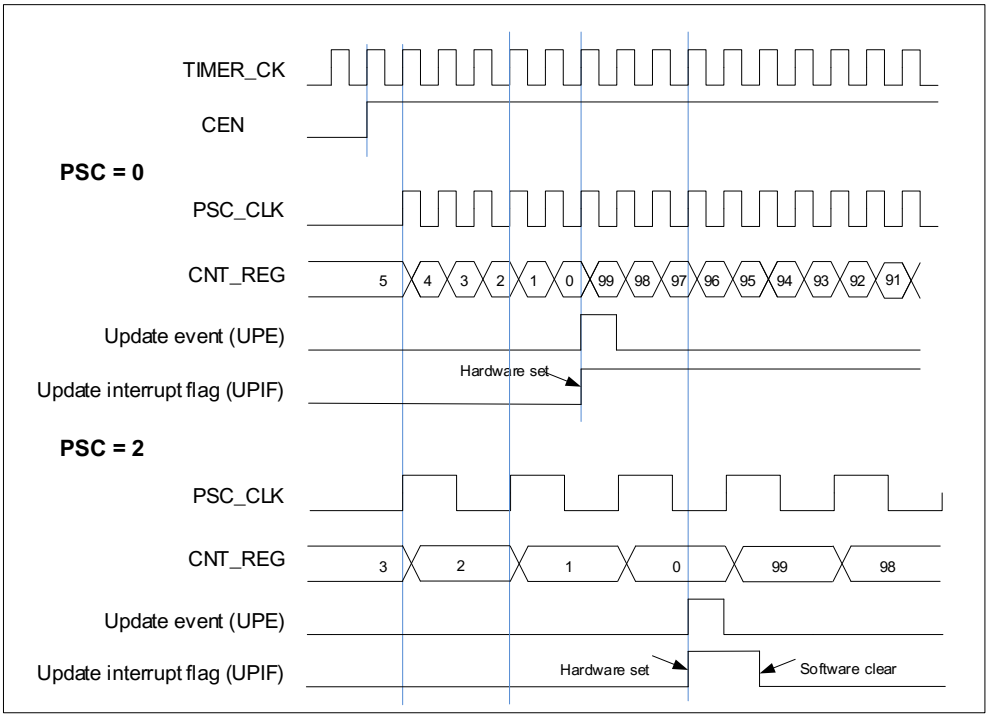
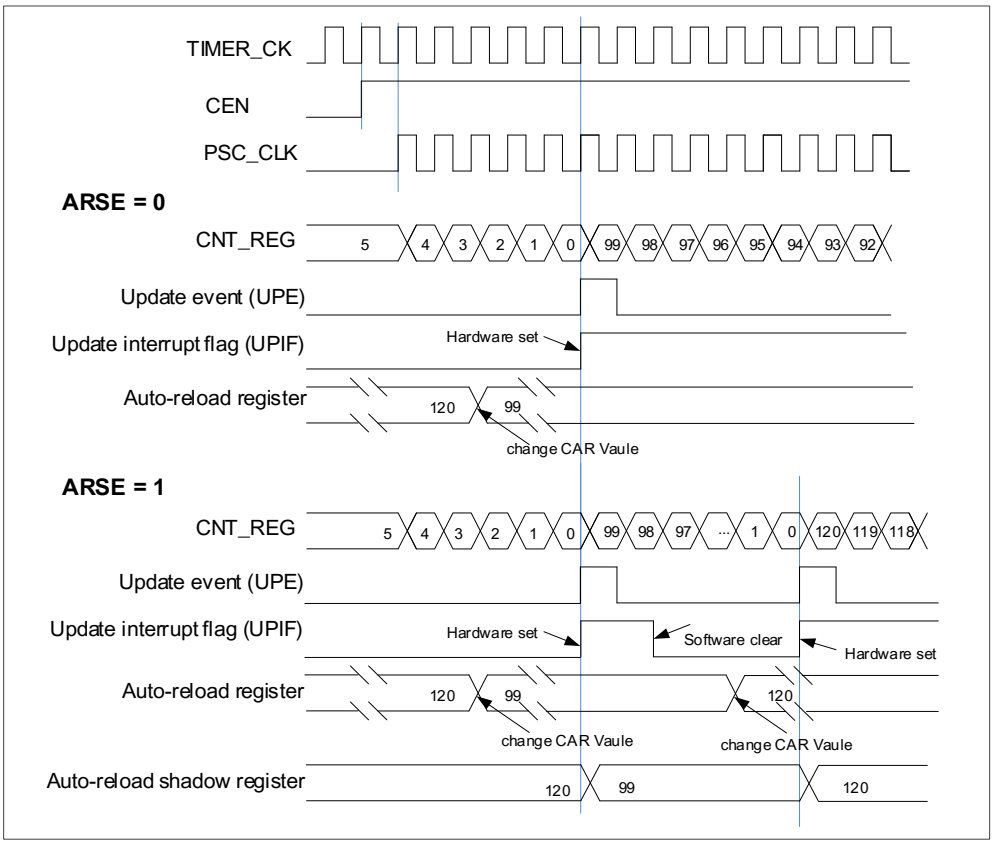


Figure 9-7. Timing chart of down counting mode, change TIMERx_CAR on the go



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.

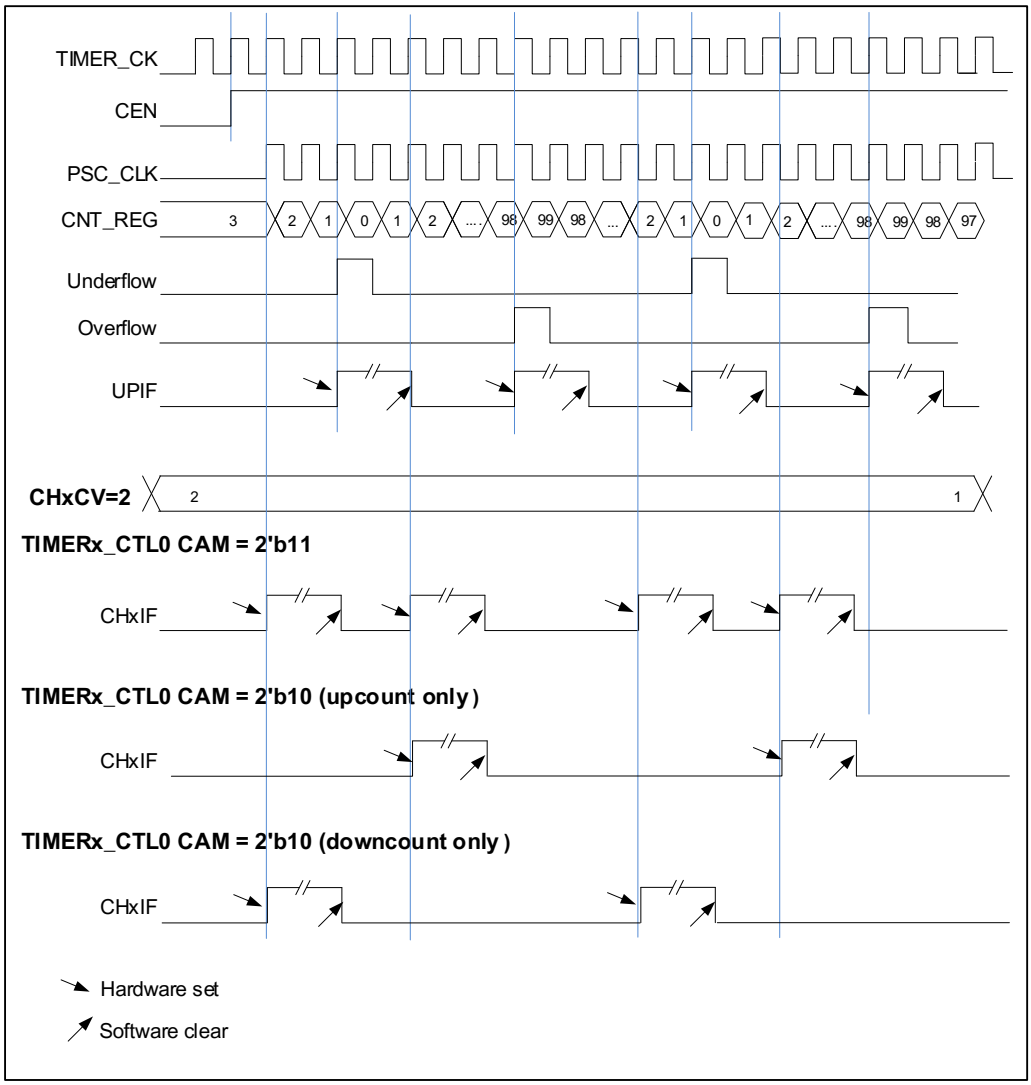
The UPIF bit in the TIMERx_INTF register can be set to 1 either when an underflow event or an overflow event occurs. While the CHxIF bit is associated with the value of CAM in TIMERx_CTL0. The details refer to ***Figure 9-8. Timing chart of center-aligned counting mode.***

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, counter auto-reload register, prescaler register) are updated.

Figure 9-8. Timing chart of center-aligned counting mode show some examples of the counter behavior when TIMERx_CAR=0x99. TIMERx_PSC=0x0

Figure 9-8. Timing chart of center-aligned counting mode



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 9-9. Repetition counter timing chart of center-aligned counting mode

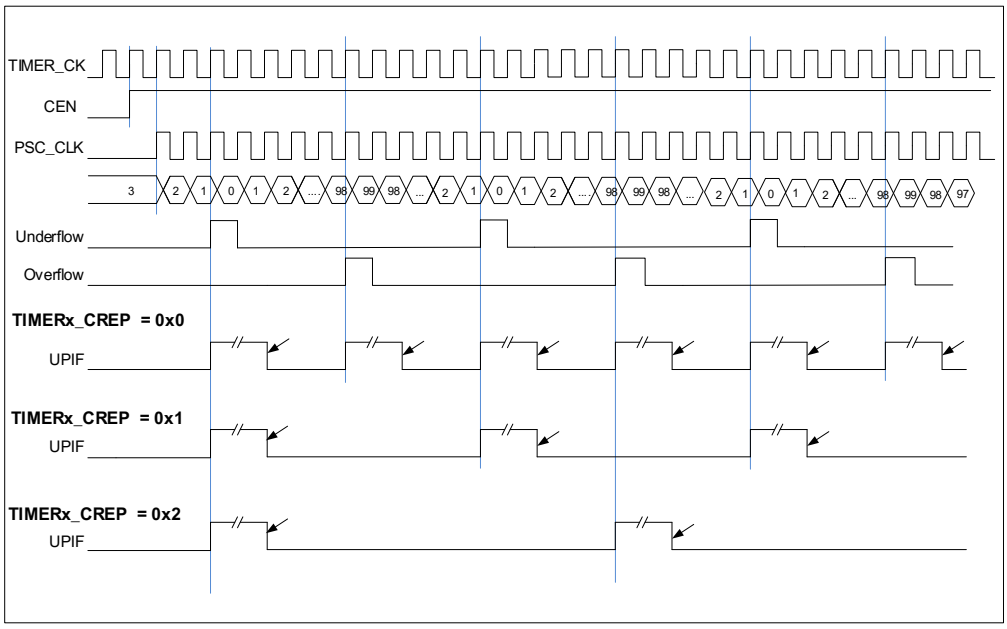


Figure 9-10. Repetition counter timing chart of up counting mode

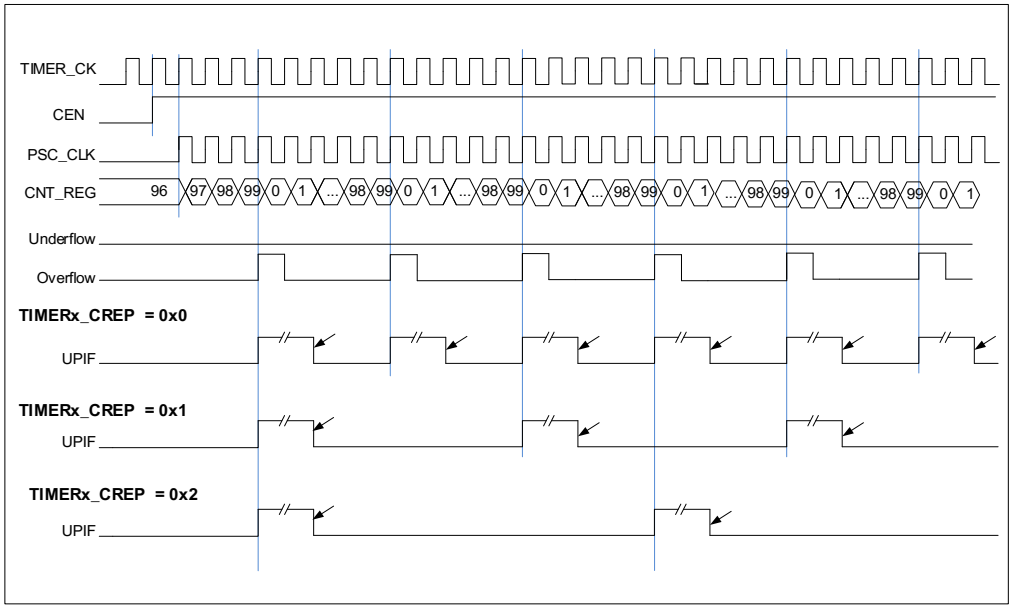
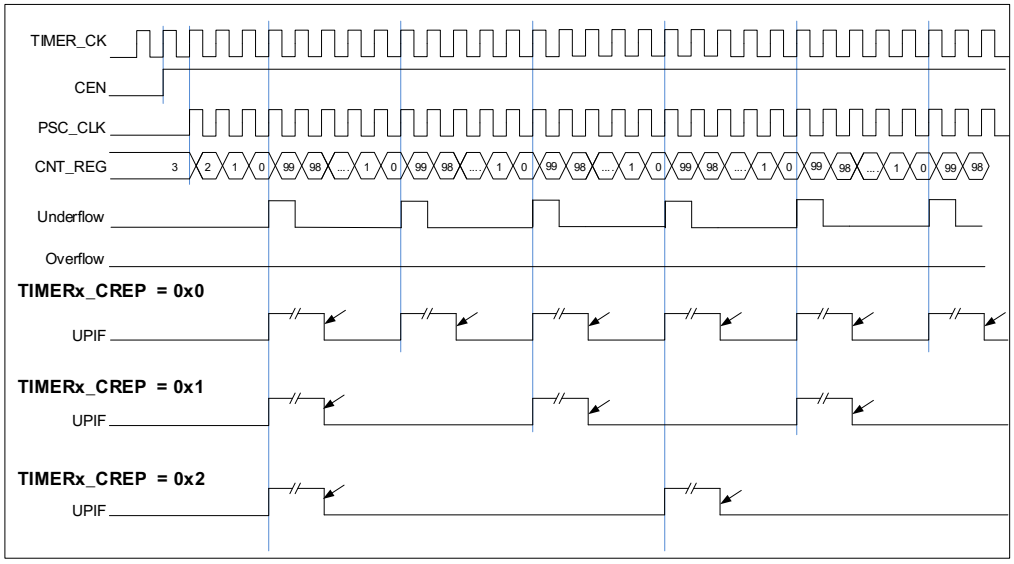


Figure 9-11. Repetition counter timing chart of down counting mode



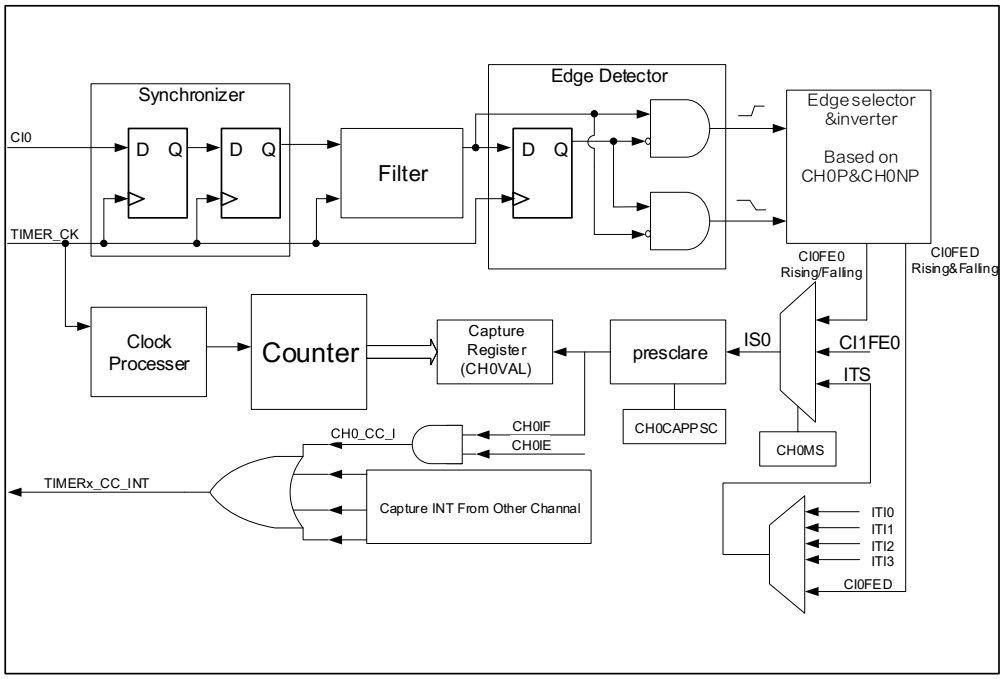
Input capture and output compare channels

The advanced timer has four independent channels which can be used as capture inputs or compare outputs. Each channel is built around a channel capture compare register including an input stage, a 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 it is enabled when CHxIE=1.

Figure 9-12. Channel input capture principle



The input signals of channelx (CIx) can be the TIMERx_CHx signal or the XOR signal of the TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 signals. First, the input signal of channel (CIx) is synchronized to TIMER_CK signal, and then sampled by a digital filter to generate a filtered input signal. Then through the edge detector, the rising or falling edge is detected by configuring CHxP bit. The input capture signal can also be selected from the input signal of other channel or the internal trigger signal by configuring CHxMS bits. The IC prescaler makes several input events generate one effective capture event. On the capture event, TIMERx_CHxCV will store the value of counter.

So, the process can be divided into several steps as below:

Step1: Filter configuration (CHxCAPFLT in TIMERx_CHCTL0).

Based on the input signal and quality of requested signal, configure compatible CHxCAPFLT.

Step2: Edge selection (CHxP/CHxNP in TIMERx_CHCTL2).

Rising edge or falling edge, choose one by configuring CHxP/CHxNP bits.

Step3: Capture source selection (CHxMS in TIMERx_CHCTL0).

As soon as selecting one input capture source by CHxMS, the channel must be set 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 to get the interrupt and DMA request.

Step5: Capture enable (CHxEN in TIMERx_CHCTL2).

Result: When the wanted input signal is captured, TIMERx_CHxCV will be set by counter's value and CHxIF is asserted. If the CHxIF is 1, the CHxOF will also be asserted. The interrupt and DMA request will be asserted or not based on the configuration of CHxIE and CHxDEN in TIMERx_DMAINTEN.

Direct generation: A DMA request or interrupt is generated by setting CHxG 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 connects to CI0 input. Select CI0 as channel 0 capture signals by setting CH0MS to 2'b01 in the channel control register (TIMERx_CHCTL0) and set capture on rising edge. Select CI0 as channel 1 capture signal by setting CH1MS to 2'b10 in the channel control register (TIMERx_CHCTL0) and set capture on falling edge. The counter is set to restart mode and is restarted on channel 0 rising edge. Then the TIMERX_CH0CV can measure the PWM period and the TIMERx_CH1CV can measure the PWM duty cycle.

- **Channel output compare function**

Figure 9-13. Channel output compare principle (with complementary output, x=0, 1, 2)

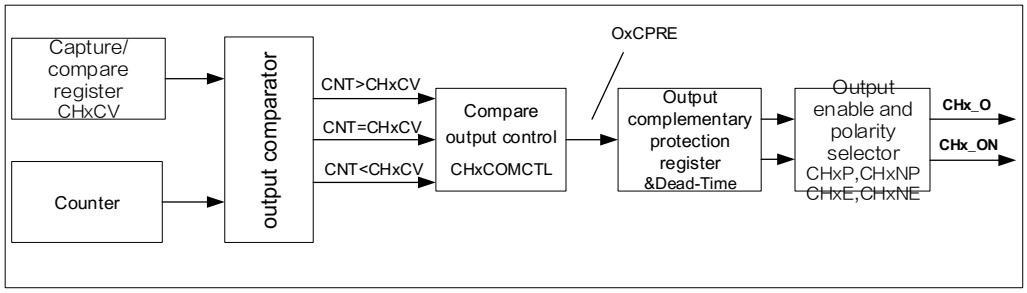


Figure 9-14. Channel output compare principle (CHx_O,x=3, 4, 5)

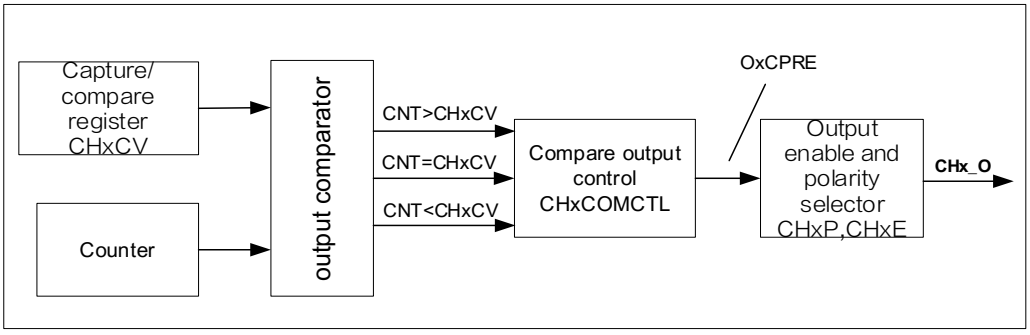


Figure 9-13. Channel output compare principle (with complementary output, x=0, 1, 2) and *Figure 9-14. Channel output compare principle (CHx_O,x=3, 4, 5)* show the principle circuit of channels output compare function. The relationship between the channel output signal CHx_O/CHx_ON and the OxCPRE signal (more details refer to *Channel output prepare signal*) is described as below: The active level of OxCPRE is high, the output level of CHx_O/CHx_ON depends on OxCPRE signal, CHxP/CHxNP bit and CHxE/CHxNE bit (please refer to the TIMERx_CHCTL2 register for more details). For examples:

1) Configure CHxP=0 (the active level of CHx_O is high, the same as OxCPRE), CHxE=1 (the output of CHx_O is enabled),

If the output of OxCPRE is active(high) level, the output of CHx_O is active(high) level;

If the output of OxCPRE is inactive(low) level, the output of CHx_O is active(low) level.

2) Configure CHxNP=0 (the active level of CHx_ON is low, contrary to OxCPRE), CHxNE=1 (the output of CHx_ON is enabled).

If the output of OxCPRE is active(high) level, the output of CHx_O is active(low) level;

If the output of OxCPRE is inactive(low) level, the output of CHx_O is active(high) level.

When CH0_O and CH0_ON are output at the same time, the specific outputs of CH0_O and CH0_ON are related to the relevant bits (ROS, IOS, POE and DTCFG bits) in the TIMERx_CCHP register. Please refer to ***Channel output complementary PWM*** for more details.

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 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 will be set and the channel (n) interrupt is generated if CHxIE = 1. And the DMA request will be asserted, if CxCDE=1.

So, the process can be divided into 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 polarity by CHxP/CHxNP.
- Enable the output by CHxEN.

Step3: Interrupt/DMA request enable configuration by CHxIE/CxCDE.

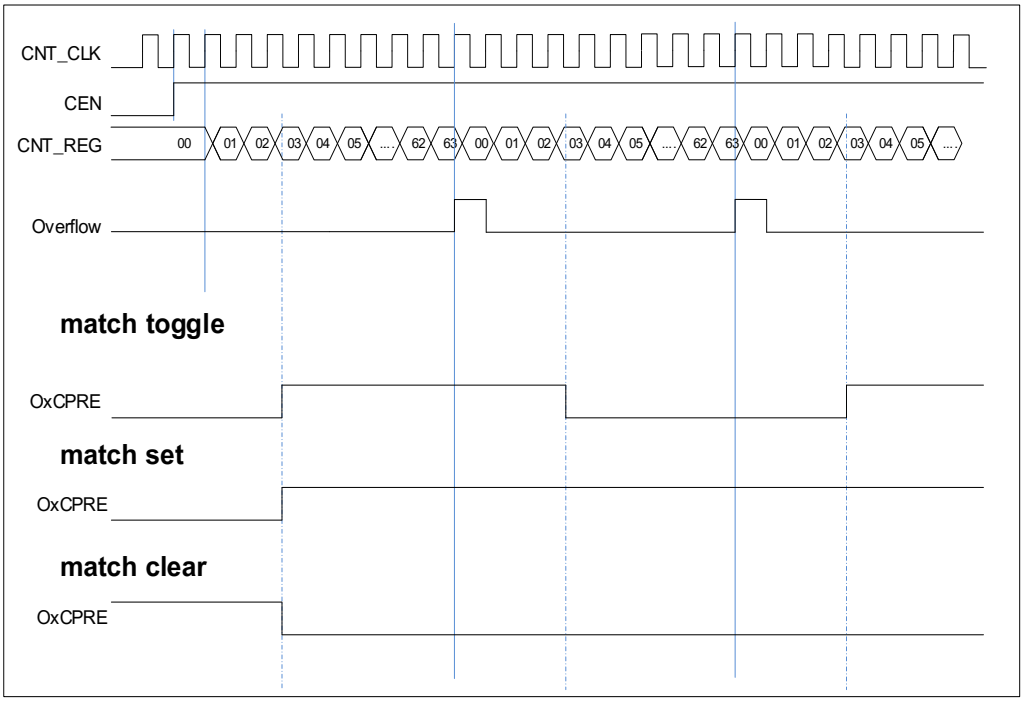
Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV.

The TIMERx_CHxCV can be changed onging to meet the expected waveform.

Step5: Start the counter by configuring CEN to 1.

Figure 9-15. Output-compare in three modes shows the three compare modes: toggle/set/clear. CAR=0x63, CHxVAL=0x3.

Figure 9-15. Output-compare in three modes



Output PWM function

In the PWM output mode (by setting the CHxCOMCTL bit to 3'b110 (PWM mode 0) or to 3'b 111(PWM mode 1)), the channel can generate PWM waveform according to the TIMERx_CAR register and TIMERx_CHxCV register.

Based on the counter mode, PWM can also be divided into EAPWM (Edge-aligned PWM) and CAPWM (Center-aligned PWM).

The EAPWM's period is determined by TIMERx_CAR and the duty cycle is determined by TIMERx_CHxCV. ***Figure 9-16. Timing chart of EAPWM*** shows the EAPWM output and interrupts waveform.

The CAPWM's period is determined by 2*TIMERx_CAR, and the duty cycle is determined by 2*TIMERx_CHxCV. ***Figure 9-17. Timing chart of CAPWM*** shows the CAPWM output and interrupts waveform.

In up counting mode, if the value of TIMERx_CHxCV is greater than the value of TIMERx_CAR, the output will be always inactive in PWM mode 0 (CHxCOMCTL=3'b110). And if the value of TIMERx_CHxCV is greater than the value of TIMERx_CAR, the output will be always active in PWM mode 1 (CHxCOMCTL=3'b111).

Figure 9-16. Timing chart of EAPWM

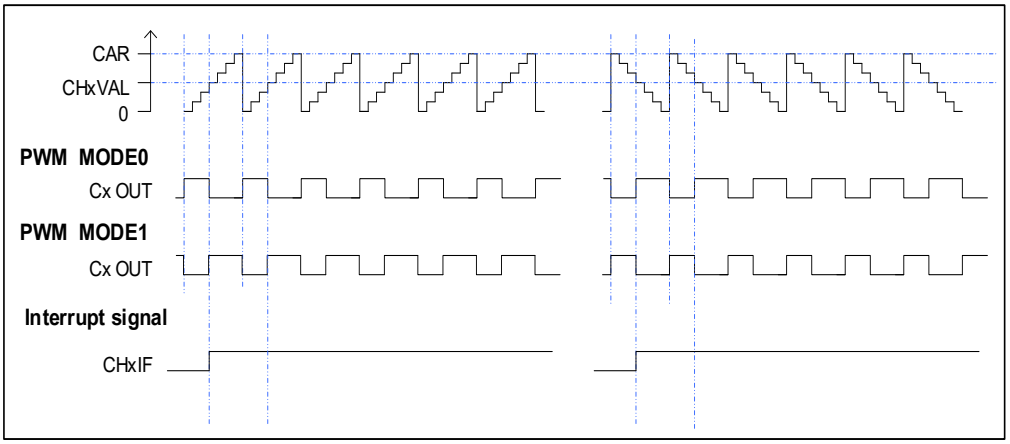
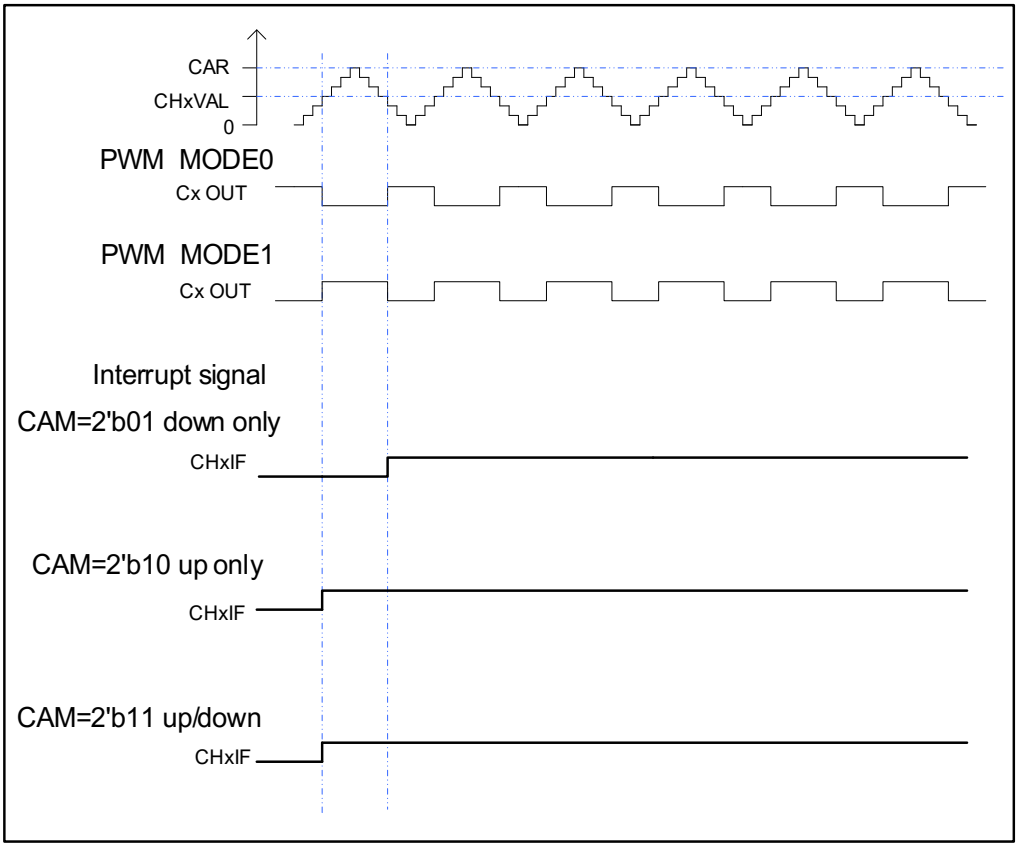


Figure 9-17. Timing chart of CAPWM



Channel output prepare signal

As is shown in *Figure 9-13. Channel output compare principle (with complementary output, x=0, 1, 2)* when TIMERx is configured in compare match output mode,a middle signal which is OxCPRE signal (Channel x output prepare signal) will be generated before the channel outputs signal. The OxCPRE signal type is defined by configuring the CHxCOMCTL

bit. The OxCPRE signal has several types of output function. These include keeping the original level by configuring the CHxCOMCTL field to 0x00, setting to high by configuring the CHxCOMCTL field to 0x01, setting to low by configuring the CHxCOMCTL field to 0x02 or toggling signal by configuring the CHxCOMCTL field to 0x03 when the counter value matches the content of the TIMERx_CHxCV register.

The PWM mode 0/PWM mode 1 output is another output type of OxCPRE which is setup by configuring 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. Refer to the definition of relative bit for more details.

Another special function of the OxCPRE signal is a forced output which can be achieved by configuring the CHxCOMCTL field to 0x04/0x05. The output can be forced to an inactive/active level irrespective of the comparison condition between the values of the counter and the TIMERx_CHxCV.

Configure the CHxCOMCEN bit to 1 in the TIMERx_CHCTL0 register, the OxCPRE signal can be forced to 0 when the ETIFP signal derived from the external ETI pin is set to a high level. The OxCPRE signal remains low until the next transition to the active state.

Channel output complementary PWM

Function of complementary is for a pair of channels, CHx_O and CHx_ON, the two output signals cannot be active at the same time. The TIMERx has 6 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, the POEN, ROS and IOS bits in the TIMERx_CCHP register, ISOx and ISOxN bits in the TIMERx_CTL1 register. The output polarity is determined by CHxP and CHxNP bits in the TIMERx_CHCTL2 register.

Table 9-2. Complementary outputs controlled by parameters

Complementary Parameters					Output Status	
POEN	ROS	IOS	CHxEN	CHxNEN	CHx_O	CHx_ON
0	0/1	0	0	0	CHx_O = ISOx CHx_ON = ISOxN CHx_O / CHx_ON output disable(1).	
				1	CHx_O = CHxP CHx_ON = CHxNP CHx_O/CHx_ON output enable. If clock is enable: CHx_O = ISOx CHx_ON = ISOxN	
			1	0		
				1		
		1	0	0	CHx_O = ISOx CHx_ON = ISOxN CHx_O / CHx_ON output disable(1).	
				1	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.	
			1	0		
				1		
1	0	0/1	0	0	CHx_O/CHx_ON = LOW CHx_O/CHx_ON output disable.	
				1	CHx_O = LOW CHx_O output disable.	CHx_ON =OxCPRE⊕(4)CHxNP CHx_ON output enable.
			1	0	CHx_O=OxCPRE⊕CHxP CHx_O output enable.	CHx_ON = LOW CHx_ON output disable.
				1	CHx_O=OxCPRE⊕CHxP CHx_O output enable.	CHx_ON =(!OxCPRE)(5)⊕CHxNP. CHx_ON output enable.
	1		0	0	CHx_O = CHxP CHx_O output "off-state".	CHx_ON = CHxNP CHx_ON output "off-state".
				1	CHx_O = CHxP CHx_O output "off-state"	CHx_ON =OxCPRE⊕CHxNP CHx_ON output enable
			1	0	CHx_O=OxCPRE⊕CHxP CHx_O output enable	CHx_ON = CHxNP CHx_ON output "off-state".

				1	CHx_O=OxCPRE⊕CHxP CHx_O output enable	CHx_ON =(!OxCPRE)⊕CHxNP CHx_ON output enable.

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

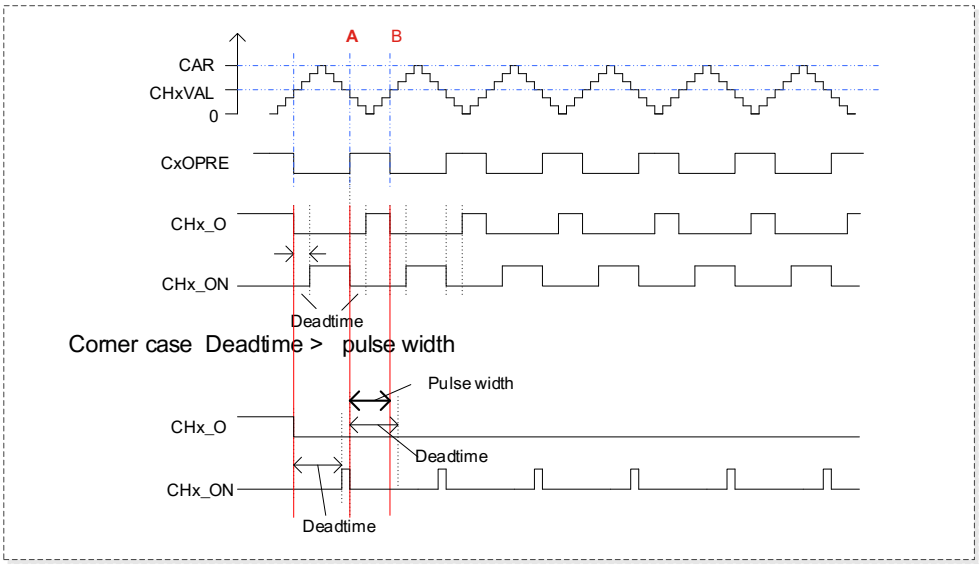
The dead time insertion is enabled when both CHxEN and CHxNEN are configured to 1'b1, it is also necessary to configure POEN to 1. The field named DTCFG defines the dead time delay that can be used for all channels except channel 3. Refer to the TIMERx_CCHP register for details about the delay time.

The dead time delay insertion ensures that two complementary signals are not active at the same time.

When the channelx match event (TIMERx counter = CHxVAL) occurs, OxCPRE will be toggled in PWM mode 0. At point A in ***Figure 9-18. Complementary output with dead time insertion***, CHx_O signal remains at the low level until the end of the dead time delay, while CHx_ON signal will be cleared at once. Similarly, at point B when the channelx match event (TIMERx counter = CHxVAL) occurs again, OxCPRE is cleared, and CHx_O signal will be cleared at once, while CHx_ON signal remains at the low level 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 duty cycle of the CHx_O signal, then the CHx_O signal is always inactive (As shown in ***Figure 9-18. Complementary output with dead time insertion***).

Figure 9-18. Complementary output with dead time insertion



Break mode

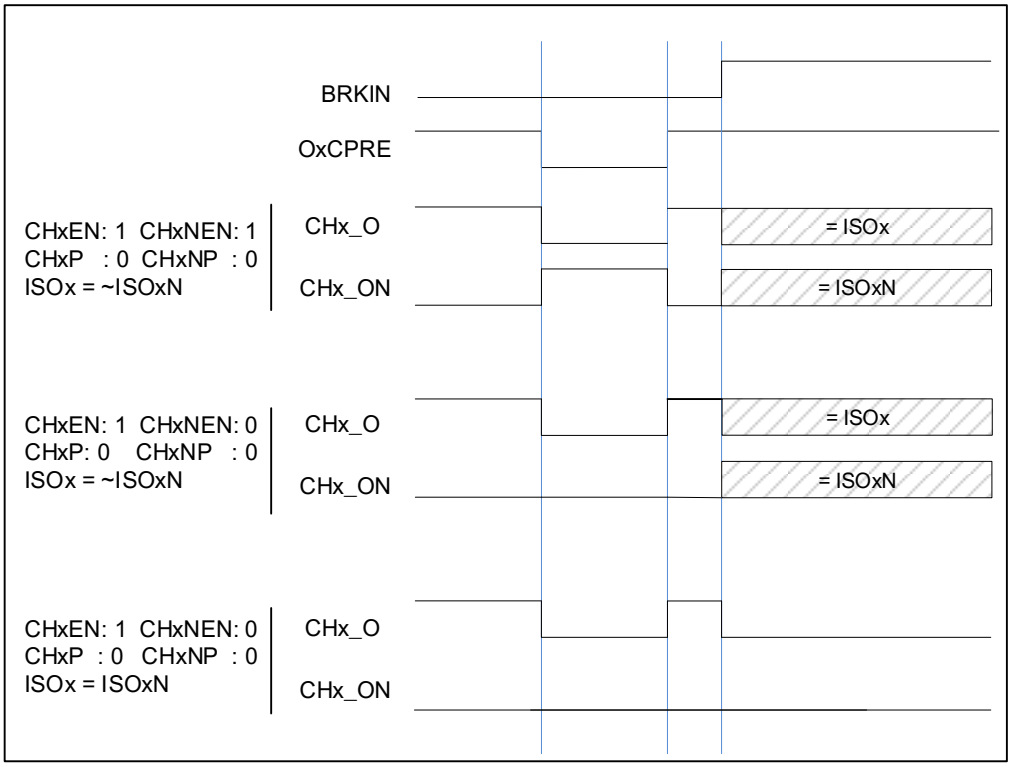
In this mode, CHx_O and CHx_ON are controlled by the POEN, IOS and ROS bits in the TIMERx_CCHP register, ISOx and ISOxN bits in the TIMERx_CTL1 register. In any case, CHx_O and CHx_ON signals cannot be set to active level at the same time. The break sources are input break pin, CMP output (only TIMER0 support) 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 polarity of BRKIN break input is setting by the BRKP bit in TIMERx_CCHP. It should be noted that the break input from CMP is valid at a high level and there is no polarity selection in the TIMER module.

When a break occurs, the POEN bit is cleared asynchronously. As soon as POEN is 0, the level of the CHx_O and CHx_ON outputs are determined by the ISOx and ISOxN bits in the TIMERx_CTL1 register. If IOS is 0, the timer releases the enable output, otherwise, the enable output remains high. The complementary outputs are first in the reset state, and then the dead time generator is reactivated 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 will be set. If BRKIE is 1, an interrupt will be generated.

Figure 9-19. Output behavior of the channel 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 with each other to generate the counter value. Setting SMC=0x01, 0x02, or 0x03 to select that the counting direction of timer is determined only by the CI0, only by the CI1, or by the CI0 and the CI1. The DIR bit is modified during the voltage level change of each direction selection source. The mechanism of changing the counter direction is shown in ***Table 9-3. Counting direction versus quadrature decoder signals***. The quadrature decoder can be regarded as an external clock with a direction selection. This means that the counter counts continuously from 0 to the counter-reload value. Therefore, users must configure the TIMERx_CAR register before the counter starts to count.

Table 9-3. Counting direction versus quadrature decoder signals

Counting mode	Level	CI0FE0		CI1FE1	
		Rising	Falling	Rising	Falling
CI0 only counting	CI1FE1=High	Down	Up	-	-
	CI1FE1=Low	Up	Down	-	-
CI1 only counting	CI0FE0=High	-	-	Up	Down
	CI0FE0=Low	-	-	Down	Up
CI0 and CI1 counting	CI1FE1=High	Down	Up	X	X
	CI1FE1=Low	Up	Down	X	X

Counting mode	Level	CI0FE0		CI1FE1	
		Rising	Falling	Rising	Falling
	CI0FE0=High	X	X	Up	Down
	CI0FE0=Low	X	X	Down	Up

Note: "-" means "no counting"; "X" means impossible.

Figure 9-20. Example of counter operation in quadrature decoder interface mode

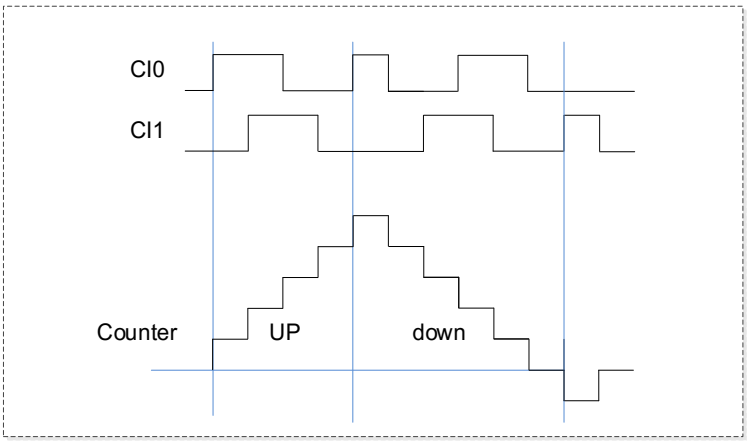
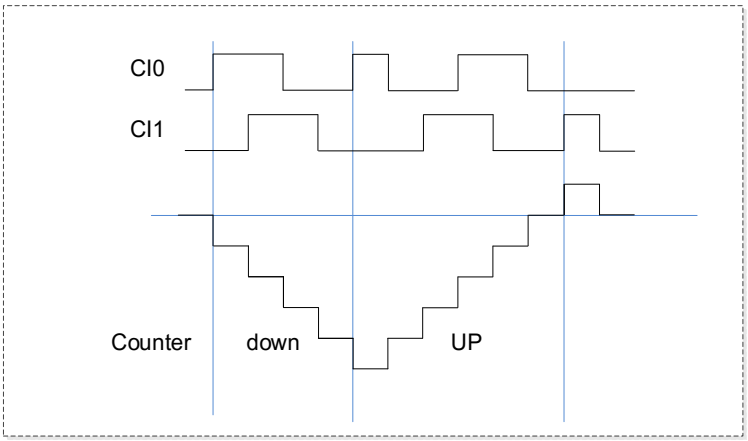


Figure 9-21. Example of quadrature decoder interface mode with CI0FE0 polarity inverted



Hall sensor function

Hall sensor is generally used to control BLDC Motor; advanced timer can support this function.

Figure 9-22. Hall sensor is used to BLDC motor show how to connect. And we can see we need two timers. First TIMER_in (Advanced/GeneralL0 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/GeneralL0 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.

TIMER_in (TIMER2) -> TIMER_out (TIMER0 ITI2)

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 9-22. Hall sensor is used to BLDC motor

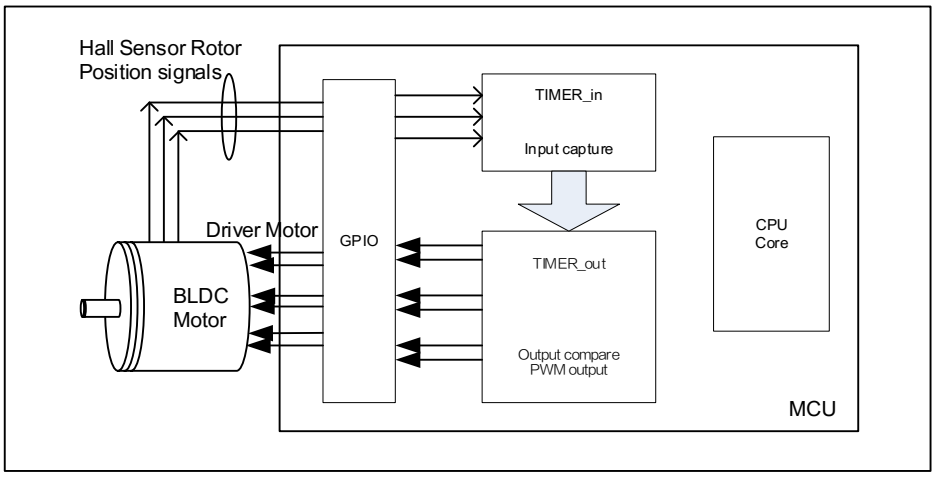
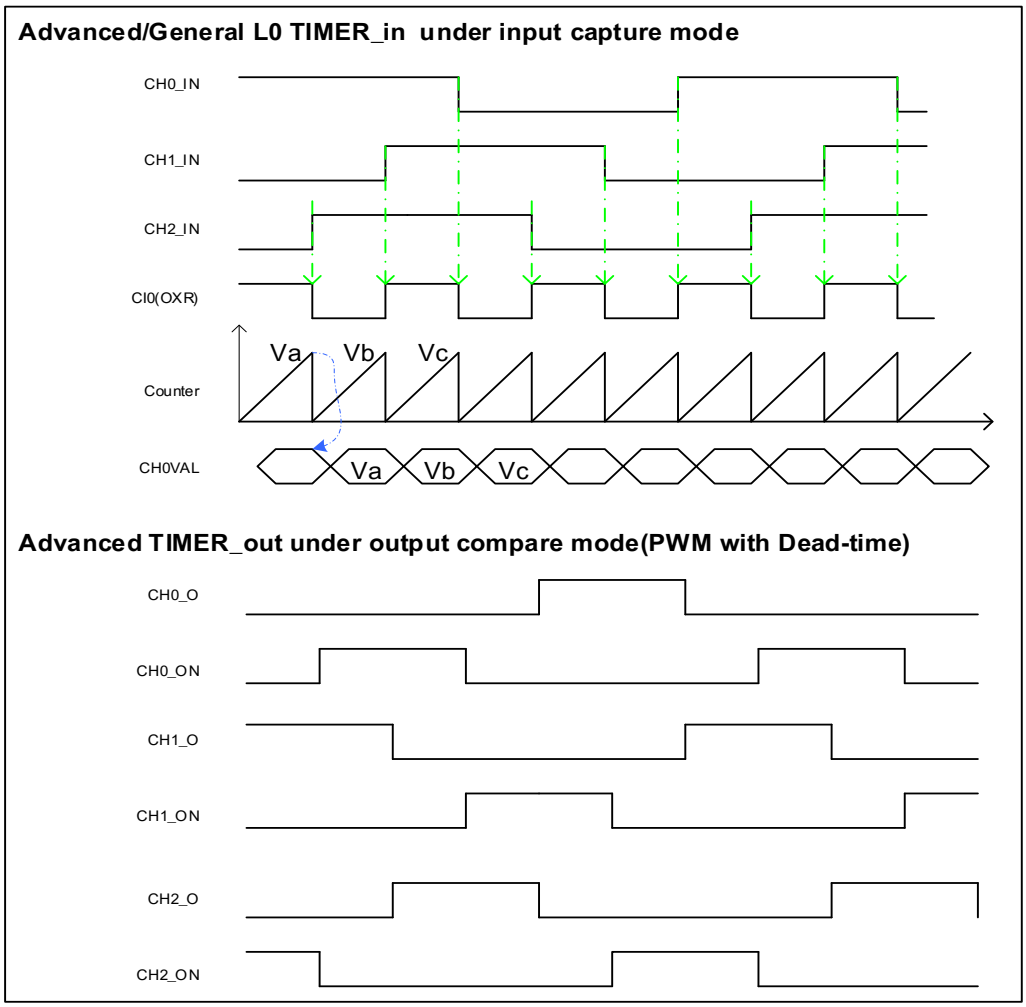


Figure 9-23. Hall sensor timing between two timers

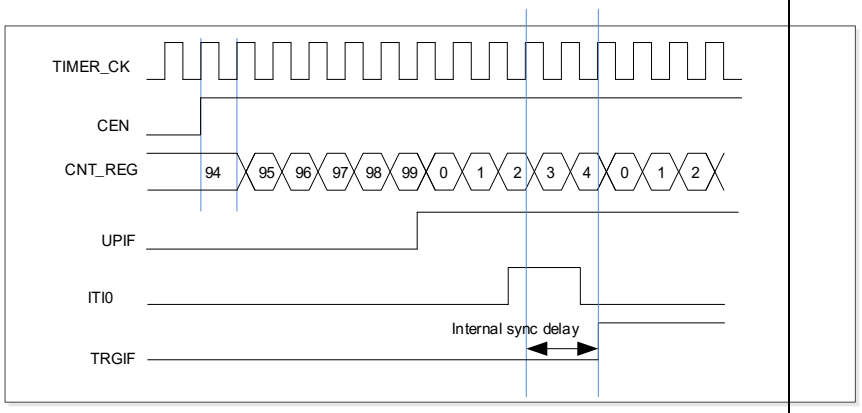


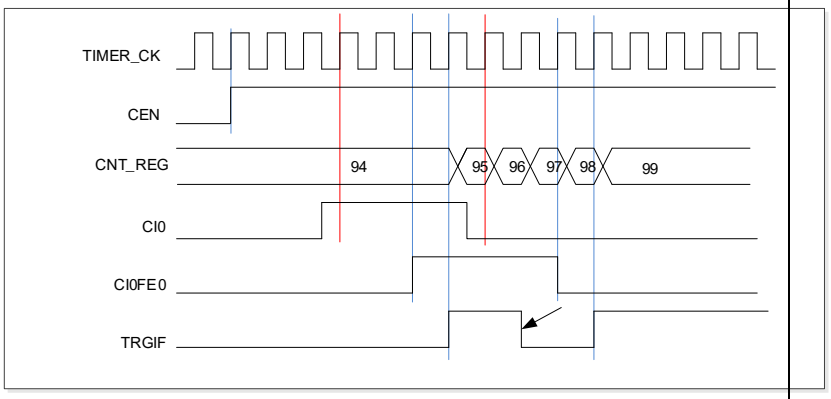
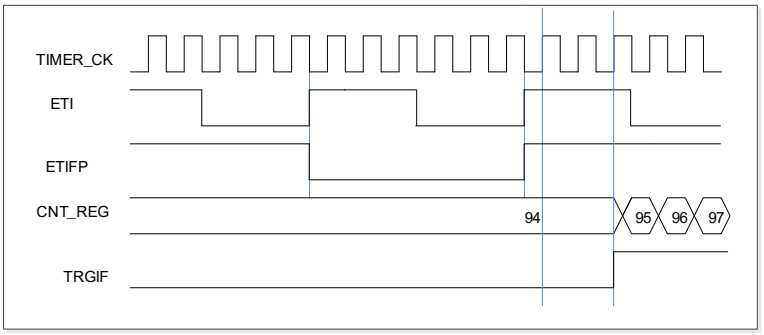
Master-slave management

The TIMERx can be synchronized with a trigger in several modes including restart mode, pause mode and event mode which is selected by the SMC[2:0] bits in the TIMERx_SMCFG register. The input trigger of these modes can be selected by the TRGS[2:0] bits in the TIMERx_SMCFG register.

Table 9-4. 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	If CI0FE0 or CI1FE1 is selected as the trigger source, configure the CHxP and CHxNP for the polarity selection and inversion. If ETIFP is selected as	For the ITIx, no filter and prescaler can be used. For the CIx, filter can be used by configuring CHxCAPFLT, no prescaler can be

<table>
<tr><th></th><th>Mode Selection</th><th>Source Selection</th><th>Polarity Selection</th><th>Filter and Prescaler</th></tr>
<tr><td></td><td></td><td>110: CI1FE1
111: ETIFP</td><td>the trigger source, configure the ETP for polarity selection and inversion.</td><td>used.
For the ETIFP, filter can be used by configuring ETFC and prescaler can be used by configuring ETPSC.</td></tr>
<tr><td rowspan="2">Exam1</td><td>Restart mode
The counter will be cleared and restart when a rising edge of trigger input comes.</td><td>TRGS[2:0] = 3'b000
ITI0 is selected.</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 9-24. Restart mode

</td></tr>
<tr><td>Exam2</td><td>Pause mode
The counter will be paused when the trigger input is low, and it will start when the trigger input is high.</td><td>TRGS[2:0]=3'b101
CI0FE0 is selected.</td><td>TI0S=0 (Non-xor) [CH0NP=0, CH0P=0]
CI0FE0 does not invert. The capture event will occur on the rising edge only.</td><td>Filter is bypassed in this example.</td></tr>
</table>

<table>
<tr><th></th><th>Mode Selection</th><th>Source Selection</th><th>Polarity Selection</th><th>Filter and Prescaler</th></tr>
<tr><td></td><td colspan="4">Figure 9-25. Pause mode

</td></tr>
<tr><td rowspan="2">Exam3</td><td>Event mode
The counter will start to count when a rising edge of trigger input comes.</td><td>TRGS[2:0] =3'b111
ETIFP is selected.</td><td>ETP = 0, the polarity of ETI does not change.</td><td>ETPSC = 1, ETI is divided by 2.
ETFC = 0, ETI does not filter.</td></tr>
<tr><td colspan="4">Figure 9-26. Event mode

</td></tr>
</table>

Single pulse mode

Single pulse mode is enabled by setting SPM in TIMERx_CTL0. If SPM is set, the counter will be cleared and stopped when the next update event occurs. In order to get a pulse waveform, the TIMERx is configured to PWM mode or compare mode by CHxCOMCTL.

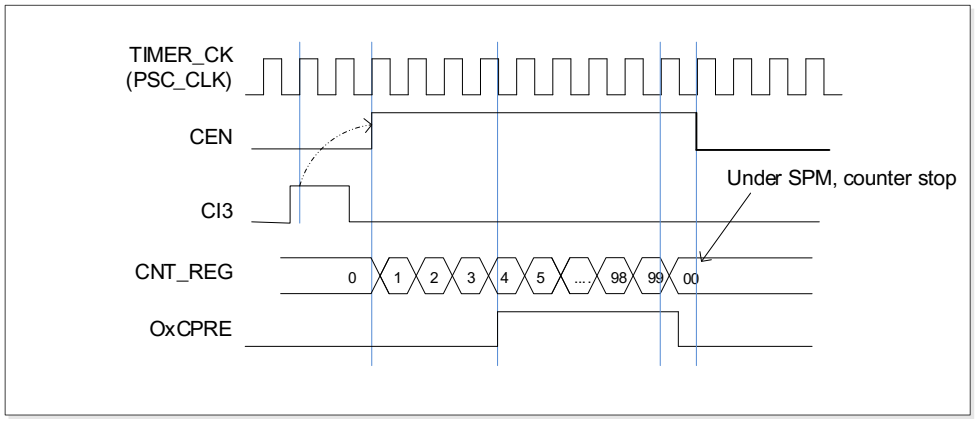
Once the timer is set to the single pulse mode, it is not necessary to configure the timer enable bit CEN in the TIMERx_CTL0 register to 1 to enable the counter. Setting the CEN bit to 1 or a trigger signal edge can generate a pulse and then keep the CEN bit at a high state until the update event occurs or the CEN bit is written to 0 by software. If the CEN bit is cleared to 0 by software, the counter will be stopped and its value will be held.

In the single pulse mode, the active edge of trigger which sets the CEN bit to 1 will enable the counter. However, there exists 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 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 the PWM mode 0 or PWM mode 1 and the trigger source is derived from the trigger signal.

Figure 9-27. Single pulse mode, TIMERx CHxCV = 4, TIMERx CAR=99 shows an example.

Figure 9-27. Single pulse mode, TIMERx_CHxCV = 4, TIMERx_CAR=99

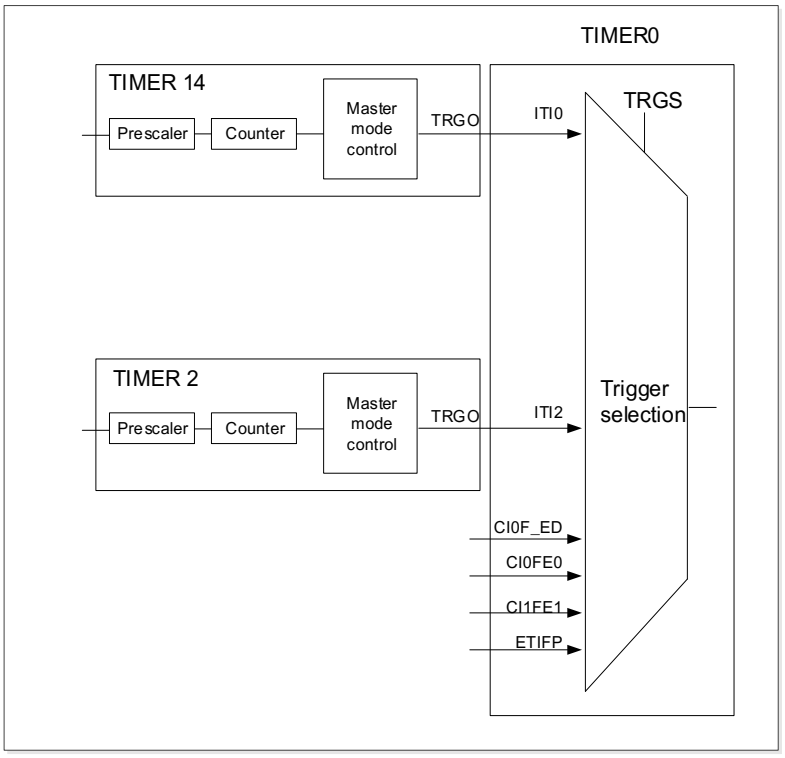


Timers interconnection

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 9-28. TIMER0 Master/Slave mode timer example shows the timer0 trigger selection when it is configured in slave mode.

Figure 9-28. TIMER0 Master/Slave mode timer example



Other interconnection examples:

- TIMER2 as prescaler for TIMER0

We configure TIMER2 as a prescaler for TIMER0. Refer to ***Figure 9-28. TIMER0 Master/Slave mode timer example*** for connections. Do as follow:

1. Configure TIMER2 in master mode and select its Update Event (UPE) as trigger output (MMC=010 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=010 in the TIMER0_SMCFG register).
4. Configure TIMER0 in external clock mode 1 (SMC=111 in TIMER0_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).

- Using an external trigger to start 2 timers synchronously

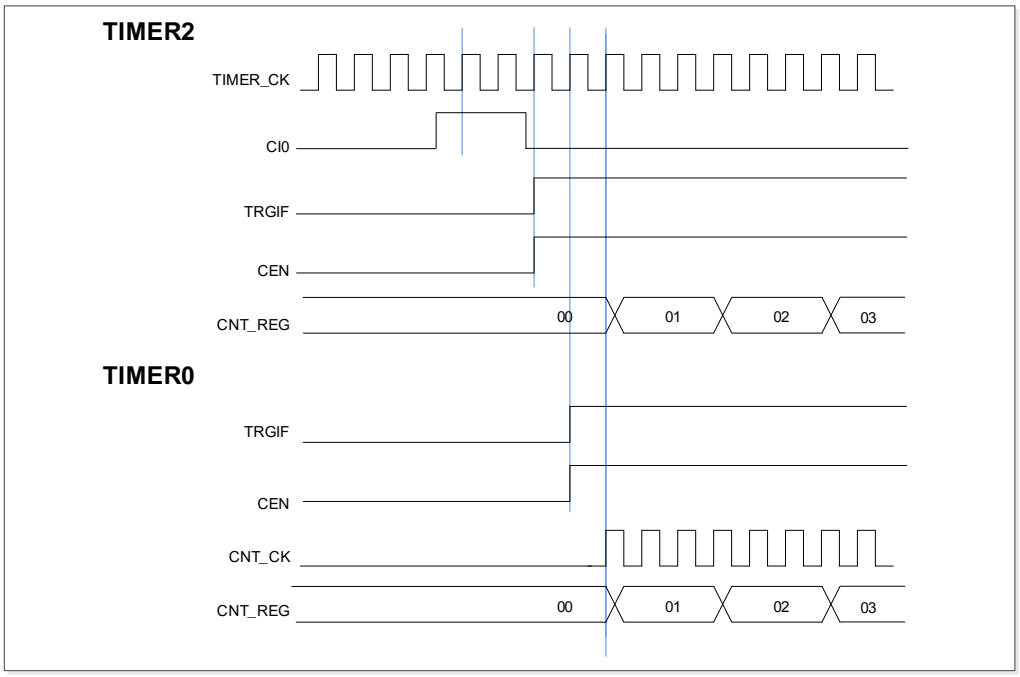
We configure the start of TIMER0 is triggered by the enable of TIMER2, and TIMER2 is triggered by its CI0 input rises edge. To ensure 2 timers start synchronously, TIMER2 must be configured in Master/Slave mode. Do as follow:

1. Configure TIMER2 slave mode to get the input trigger from CI0 (TRGS=100 in the TIMER2_SMCFG register).

2. Configure TIMER2 in event mode (SMC=110 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=010 in the TIMER0_SMCFG register).
5. Configure TIMER0 in event mode (SMC=110 in the TIMER0_SMCFG register).

When a rising edge occurs on TIMER2's CI0, two timer counters starts counting synchronously on the internal clock and both TRGIF flags are set.

Figure 9-29. Triggering TIMER0 and TIMER2 with TIMER2's CI0 input



Timer DMA mode

Timer DMA mode is the function that configures timer's register by DMA module. The relative registers are TIMERx_DMACFG and TIMERx_DMATB. Corresponding DMA request bit should be asserted to enable DMA request for internal interrupt event. TIMERx will send a request to DMA when the interrupt event occurs. DMA is configured to M2P (memory to peripheral) mode and the address of TIMERx_DMATB is configured to PADDR (peripheral base address), then DMA will access the TIMERx_DMATB. In fact, TIMERx_DMATB register 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), the timer sends only one DMA request. 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 and DMATA+0xC at the next 3 accesses to TIMERx_DMATB. In a word, one-time DMA internal interrupt event asserts, (DMATC+1) times request will be sent by TIMERx.

If one more DMA request event occurs, TIMERx will repeat the process above.

Timer debug mode

When the Cortex®-M23 is halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register is set to 1, the TIMERx counter stops.

9.1.5. TIMERx register definition (x=0)

TIMER0 base address: 0x4001 2C00

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 quadrature decoder 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, pause mode and quadrature decoder 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															ISO5
															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISO4	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		rw

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value
16	ISO5	Idle state of channel 5 output Refer to ISO0 bit
15	ISO4	Idle state of channel 4 output Refer to ISO0 bit
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, CH1 and 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 in master mode to slave timers 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 enable (for CHxEN, CHxNEN and CHxCOMCTL bits) are set (CCSE=1), these shadow registers update are controlled as below: 0: The shadow registers update by when CMTG bit is set. 1: The shadow registers update by 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 based when commutation event coming. 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]			OCRC	SMC[2:0]		
rw	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 ETIF 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_{CK_TIMER}
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 OCRC

OCPRE clear source selection

0: OCPRE clear source is connected to the OCPRE_CLR input

1: OCPRE clear source is connected to ETIFP

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.
Because CI0F_ED outputs 1 pulse for each transition on CI0F, and the pause mode checks the level of the trigger signal, when CI0F_ED is selected as the trigger input, the pause mode must not be used.

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													CH5IE	CH4IE	CH5DEN
													rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH4DEN	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	rw

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18	CH5IE	Channel 5 capture/compare interrupt enable 0: disabled 1: enabled
17	CH4IE	Channel 4 capture/compare interrupt enable 0: disabled 1: enabled
16	CH5DEN	Channel 5 capture/compare DMA request enable 0: disabled 1: enabled

15	CH4DEN	Channel 4 capture/compare DMA request enable 0: disabled 1: enabled
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															CH5OF
															rc_w0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH4OF	CH5IF	CH4IF	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	rc_w0	rc_w0	rc_w0

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	CH5OF	Channel 5 over capture flag Refer to CH0OF description
15	CH4OF	Channel 4 over capture flag Refer to CH0OF description
14	CH5IF	Channel 5 's capture/compare interrupt flag Refer to CH0IF description
13	CH4IF	Channel 4 's capture/compare interrupt flag Refer to CH0IF description
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 channel's commutation event occurs, and cleared by software 0: No channel commutation interrupt occurred 1: Channel commutation interrupt occurred
4	CH3IF	Channel 3 's capture/compare interrupt flag Refer to CH0IF description
3	CH2IF	Channel 2 's capture/compare interrupt flag Refer to CH0IF description
2	CH1IF	Channel 1 's capture/compare interrupt flag Refer to CH0IF description
1	CH0IF	Channel 0 's capture/compare interrupt flag This flag is set by hardware and cleared by software. When channel 0 is in input mode, this flag is set when a capture event occurs. When channel 0 is in output mode, this flag is set when a compare event occurs. If Channel0 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 on an update event 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						CH5G	CH4G	BRKG	TRGG	CMTG	CH3G	CH2G	CH1G	CH0G	UPG
						w	w	w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	CH5G	Channel 5's capture or compare event generation Refer to CH0G description
8	CH4G	Channel 4's capture or compare event generation Refer to CH0G description
7	BRKG	Break event generation This bit is set by software and cleared by hardware automatically. When this bit is set, the POEN bit is cleared and BRKIF flag is 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 is 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's c/c control update event
4	CH3G	Channel 3's capture or compare event generation Refer to CH0G description
3	CH2G	Channel 2's capture or compare event generation Refer to CH0G description
2	CH1G	Channel 1's capture or compare event generation Refer to CH0G description
1	CH0G	Channel 0's capture or compare event generation This bit is set by software in order to generate a capture or compare event in channel 0, it is automatically cleared by hardware. When this bit is set, the CH0IF flag is set, the corresponding interrupt or DMA request is sent if enabled. In addition, if channel 1 is configured in input mode, the current value of the counter is captured in TIMERx_CH0CV register, and the CH0OF flag is set if the CH0IF flag was already high. 0: No generate a channel 1 capture or compare event 1: Generate a channel 1 capture or compare event
0	UPG	Update event generation This bit can be set by software, and cleared by hardware automatically. When this bit is set, the counter is cleared if the center-aligned or up counting mode is selected, else (down counting) 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
CH1COM CEN	CH1COMCTL[2:0]			CH1COM SEN	CH1COM FEN	CH1MS[1:0]		CH0COM CEN	CH0COMCTL[2:0]			CH0COM SEN	CH0COM FEN	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 OCPRE clear source signal is detected as high level, the O0CPRE signal will be cleared. The OCPRE clear source signal is selected by OCRC bit in TIMERx_SMCFG register. 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 0: Channel 0 output quickly compare disable. Only the valid edge of the comparison result will generate a comparison match, and CH0_O is set to the compare level. The minimum delay from an edge on the result of the comparison to activate CH0_O output is 5 clock cycles. 1: Channel 0 output quickly compare enable. Both the valid edge of the trigger input signal and the comparison result will generate a comparison match, and CH0_O is set to the compare level.The minimum delay from an edge on the trigger input to activate CH0_O output is 3 clock cycles.
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

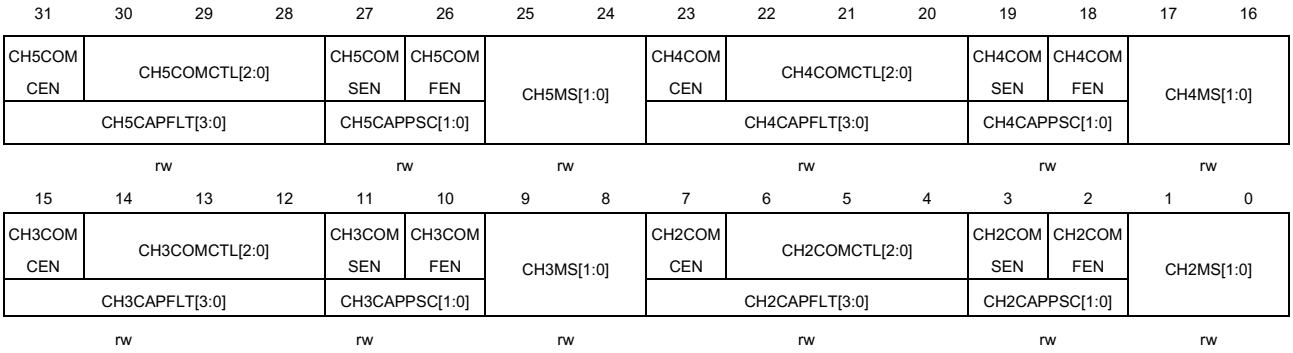
1:0 CH0MS[1:0] — Channel 0 mode selection
Same as Output compare mode

Channel control register 1 (TIMERx_CHCTL1)

Address offset: 0x1C

Reset value: 0x0000 0000

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



Output compare mode:

Bits	Fields	Descriptions
31	CH5COMCEN	Channel 5 output compare clear enable Refer to CH0COMCEN description
30:28	CH5COMCTL[2:0]	Channel 5 compare output control Refer to CH0COMCTL description
27	CH5COMSEN	Channel 5 output compare shadow enable Refer to CH0COMSEN description
26	CH5COMFEN	Channel 5 output compare fast enable Refer to CH0COMSEN description
25:24	CH5MS[1:0]	Channel 5 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. (CH5EN bit in TIMERx_CHCTL2 register is reset). 00: Channel 5 is configured as output 01: Channel 5 is configured as input, IS5 is connected to CI5FE5 10: Channel 5 is configured as input, IS5 is connected to CI4FE5 11: Channel 5 is configured as input, IS5 is connected to ITS. **Note:** When CH5MS[1:0]=11, it is necessary to select an internal trigger input through TRGS bits in TIMERx_SMCFG register.
23	CH4COMCEN	Channel 4 output compare clear enable Refer to CH0COMCEN description
22:20	CH4COMCTL[2:0]	Channel 4 compare output control Refer to CH0COMCTL description
19	CH4COMSEN	Channel 4 output compare shadow enable

		Refer to CH0COMSEN description
18	CH4COMFEN	Channel 4 output compare fast enable Refer to CH0COMSEN description
17:16	CH4MS[1:0]	Channel 4 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. (CH4EN bit in TIMERx_CHCTL2 register is reset). 00: Channel 4 is configured as output 01: Channel 4 is configured as input, IS4 is connected to CI4FE4 10: Channel 4 is configured as input, IS4 is connected to CI5FE4 11: Channel 4 is configured as input, IS4 is connected to ITS. **Note:** When CH4MS[1:0]=11, it is necessary to select an internal trigger input through TRGS bits in TIMERx_SMCFG register.
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 CH0COMSEN 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 configured as output 01: Channel 3 is configured as input, IS3 is connected to CI3FE3 10: Channel 3 is configured as input, IS3 is connected to CI2FE3 11: Channel 3 is configured as input, 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 OCPRE clear source signal is detected as high level, the O0CPRE signal will be cleared. The OCPRE clear source signal is selected by OCRC bit in TIMERx_SMCFG register. 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 controls the behavior of the output reference signal O2CPRE which

drives CH2_O and CH2_ON. O2CPRE is active high, while CH2_O and CH2_ON active level depends on CH2P and CH2NP 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 matches the output compare register TIMERx_CH2CV.
010: Clear the channel output. O2CPRE signal is forced low when the counter matches the output compare register TIMERx_CH2CV.
011: Toggle on match. O2CPRE toggles when the counter matches the output compare register TIMERx_CH2CV.
100: Force low. O2CPRE is forced low level.
101: Force high. O2CPRE is forced high level.
110: PWM mode0. When counting up, O0CPRE is active as long as the counter is smaller than TIMERx_CH0CV else inactive. When counting down, O0CPRE is inactive as long as the counter is larger than TIMERx_CH0CV else active.
111: PWM mode1. When counting up, O0CPRE is inactive as long as the counter is smaller than TIMERx_CH0CV else active. When counting down, O0CPRE is active as long as the counter is larger than TIMERx_CH0CV else inactive.
When configured in PWM mode, the O2CPRE level changes only when the output compare mode switches from "Timing mode" mode to "PWM" mode or when the result of the comparison 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 validating the shadow register only in single pulse mode (SPM bit in TIMERx_CTL0 register is set). 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 0: Channel 2 output quickly compare disable. Only the valid edge of the comparison result will generate a comparison match, and CH2_O is set to the compare level. The minimum delay from an edge on the result of the comparison to activate CH2_O output is 5 clock cycles. 1: Channel 2 output quickly compare enable. Both the valid edge of the trigger input signal and the comparison result will generate a comparison match, and CH2_O is set to the compare level.The minimum delay from an edge on the trigger input to activate CH2_O output is 3 clock cycles.

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 configured as output 01: Channel 2 is configured as input, IS2 is connected to CI2FE2 10: Channel 2 is configured as input, IS2 is connected to CI3FE2 11: Channel 2 is configured as input, 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
31:28	CH5CAPFLT[3:0]	Channel 5 input capture filter control Refer to CH0CAPFLT description
27:26	CH5CAPPSC[1:0]	Channel 5 input capture prescaler Refer to CH0CAPPSC description
25:24	CH5MS[1:0]	Channel 5 mode selection Same as Output compare mode
23:20	CH4CAPFLT[3:0]	Channel 4 input capture filter control Refer to CH0CAPFLT description
19:18	CH4CAPPSC[1:0]	Channel 4 input capture prescaler Refer to CH0CAPPSC description
17:16	CH4MS[1:0]	Channel 4 mode selection Same as Output compare mode
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 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														CH5P	CH5EN
														rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH4P	CH4EN	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	rw	rw

Bits	Fields	Descriptions
31:18	Reserved	Must be kept at reset value

17	CH5P	Channel 5 capture/compare function polarity Refer to CH0P description
16	CH5EN	Channel 5 capture/compare function enable Refer to CH0EN description
15	CH4P	Channel 4 capture/compare function polarity Refer to CH0P description
14	CH4EN	Channel 4 capture/compare function enable Refer to CH0EN description
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-filed 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 channel0. 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. [CH0NP, CH0P] will select the active trigger or capture polarity for CI0FE0 or CI1FE0. [CH0NP==0, CH0P==0]: CIxFE0's rising edge is the active signal for capture or trigger operation in slave mode. And CIxFE0 will not be inverted. [CH0NP==0, CH0P==1]: CIxFE0's falling edge 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]: CIxFE0's falling and rising edge are both the active signal for capture or trigger operation in slave mode. And CIxFE0 will be not inverted. This bit cannot be modified when PROT [1:0] bit-filed 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 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-filed indicates the current counter value. Writing to this bit-filed 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															

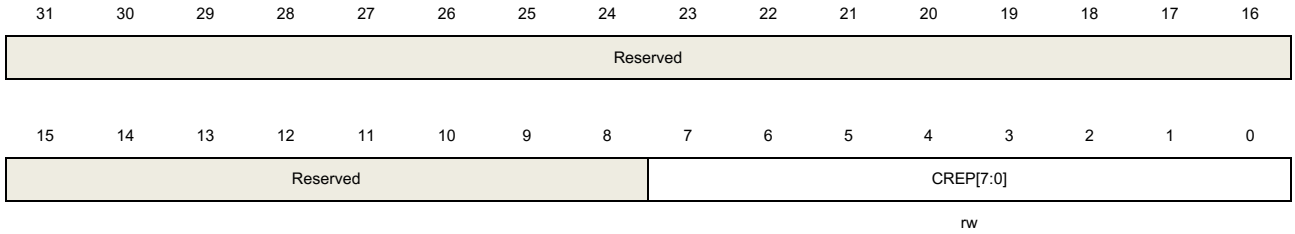
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	CARL[15:0]	Counter auto reload value This bit-filed specifies the auto reload value of the counter.

Counter repetition register (TIMERx_CREP)

Address offset: 0x30

Reset value: 0x0000 0000

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



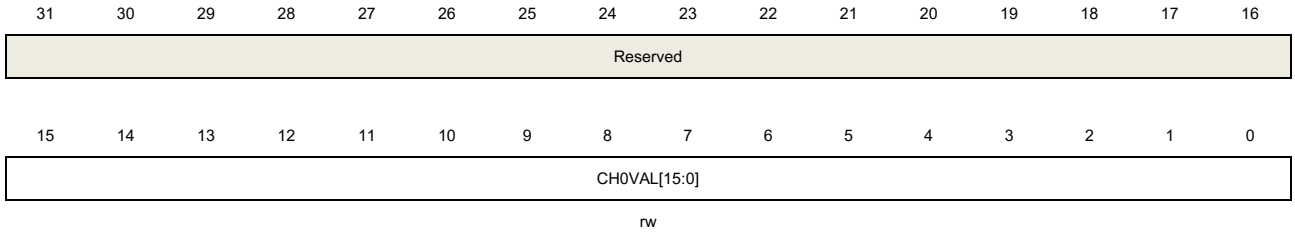
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	CREP[7:0]	Counter repetition value This bit-filed specifies the update event generation rate. Each time the repetition counter counting down to zero, an update event is generated. The update rate of the shadow registers is also affected by this bit-filed 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)



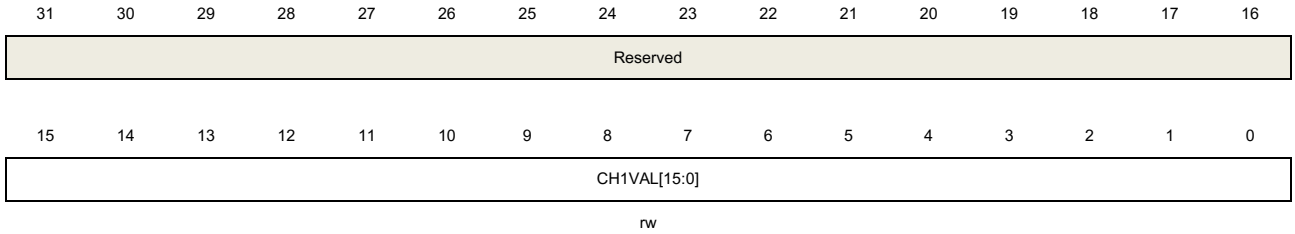
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	CH0VAL[15:0]	Capture or compare value of channel0 When channel 0 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 0 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates 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)



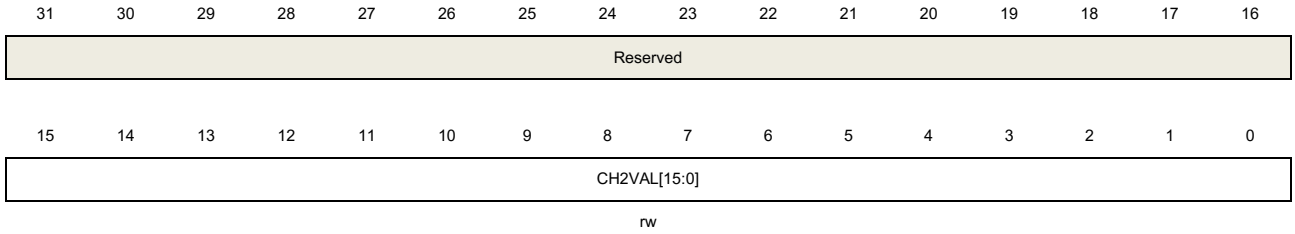
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	CH1VAL[15:0]	Capture or compare value of channel1 When channel 1 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 1 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates 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)



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	CH2VAL[15:0]	Capture or compare value of channel 2 When channel 2 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 2 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates 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 or compare value of channel 3 When channel3 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 3 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Complementary channel protection register (TIMERx_CCHP)

Address offset: 0x44

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
POEN	OAEN	BRKP	BRKEN	ROS	IOS	PROT[1:0]		DTCFG[7:0]							
rw	rw	rw	rw	rw	rw	rw		rw							

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
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.

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 9-2. 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 9-2. 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:

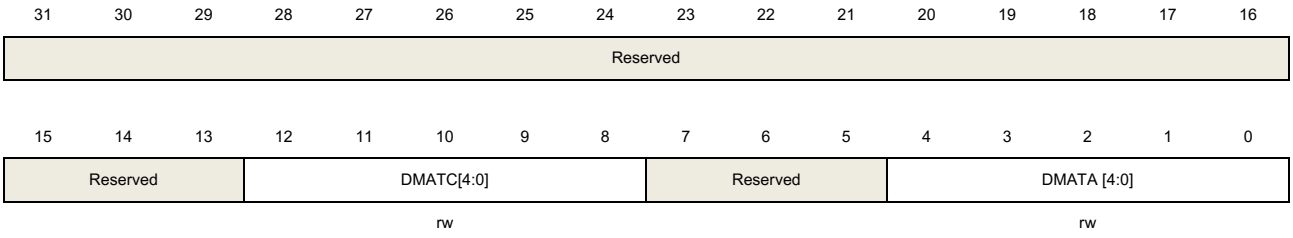
1. t_{DTS_CK} is the period of DTS_CK which is configured by CKDIV[1:0] in TIMERx_CTL0.
2. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.

DMA configuration register (TIMERx_DMACFG)

Address offset: 0x48

Reset value: 0x0000 0000

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



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 filed 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.

Channel 4 capture/compare value register (TIMERx_CH4CV)

Address offset: 0x50

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
CH4VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	CH4VAL[15:0]	Capture or compare value of channel 4 When channel4 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 4 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Channel 5 capture/compare value register (TIMERx_CH5CV)

Address offset: 0x54

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
CH5VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	CH5VAL[15:0]	Capture or compare value of channel 5 When channel5 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 5 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates 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	OUTSEL
														rw	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. 0: No effect 1: If write the CHxVAL register, the write value is same as the CHxVAL value, the write access ignored
0	OUTSEL	The output value selection This bit-field set and reset by software 0: No effect 1: If POEN and IOS are 0, the output disabled

9.2. General level0 timer (TIMERx, x=1, 2)

9.2.1. Overview

The general level0 timer module (TIMER1, 2) 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(TIMER2) or 32-bit(TIMER1) 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.

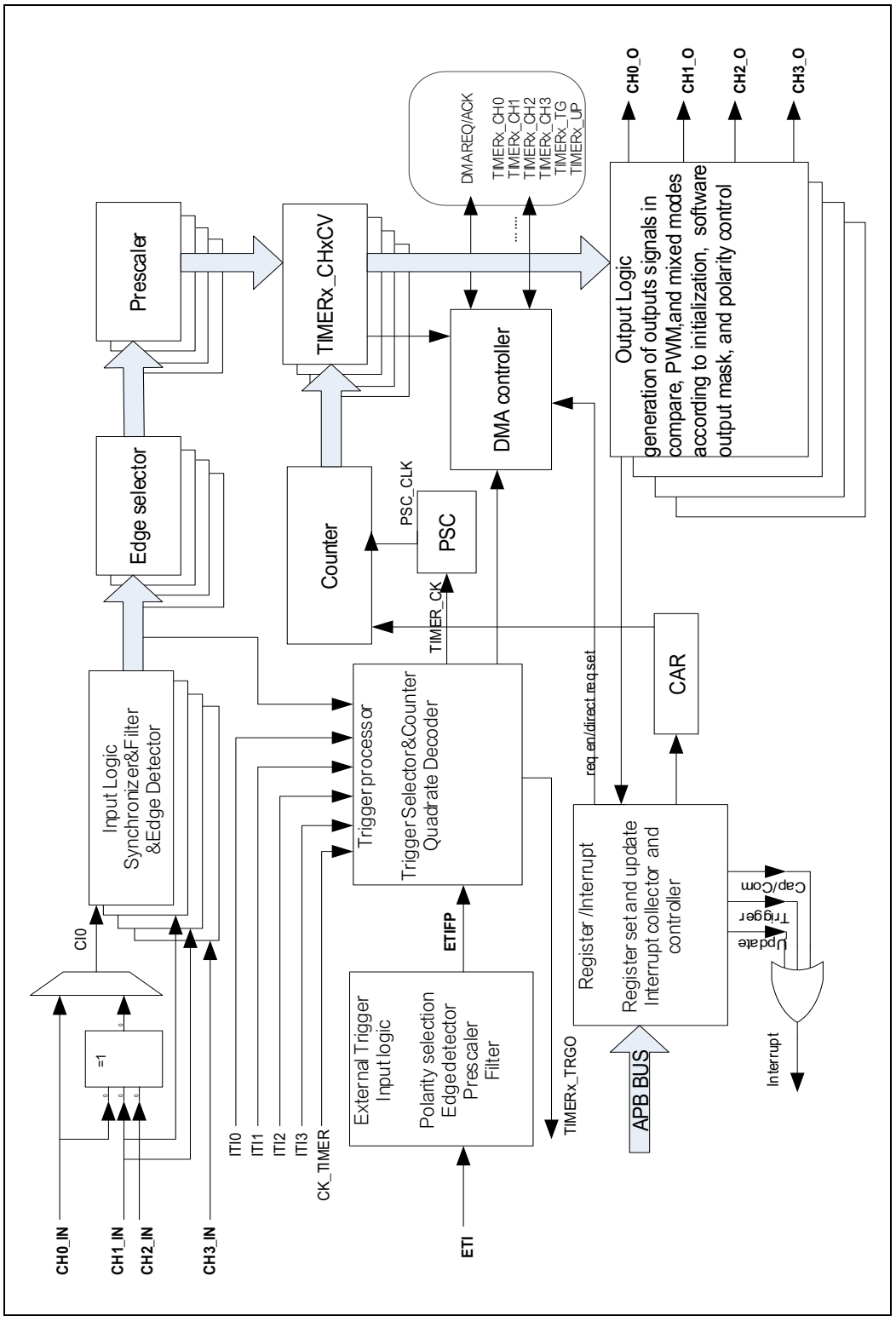
9.2.2. Characteristics

- Total channel num: 4.
- Counter width: 16 bits(TIMER2), 32 bits(TIMER1).
- 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.
- Support external enable/disable timer

9.2.3. Block diagram

Figure 9-30. General Level 0 timer block diagram provides details on the internal configuration of the general level0 timer.

Figure 9-30. General Level 0 timer block diagram



9.2.4. Function overview

Timer gate control

When the GATCTL be set to 0, the timer only controlled by CEN. Otherwise it also controlled by the external input signal INTx(INT0 for TIMER1, INT1 for TIMER2), as shown in ***Table 9-5. Timerx(x=1,2) enable options***.

When GATCTL in the TIMERx_CTL1 is 0, set the CEN bit to enable the timer.

When GATCTL in the TIMERx_CTL1 is 1, pull up the input signal INTx and set the CEN bit to enable the timer. Otherwise, the timer will be disabled.

The INTxPL bit in GPIO_INTCTL register of GPIO defines the polarity of INTx.

The INTxSL bit in GPIO_INTCTL register of GPIO selects INTx Port Pin.

Table 9-5. Timerx(x=1,2) enable options

CEN bit	GATCTL bit	INTx Port Pin	INTxPL bit	TIMER enable
0	x	x	x	Disabled
1	0	x	x	Enabled
1	1	0	0	Disabled
1	1	0	1	Enabled
1	1	1	0	Enabled
1	1	1	1	Disabled

Note: "x" means any value.

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]).

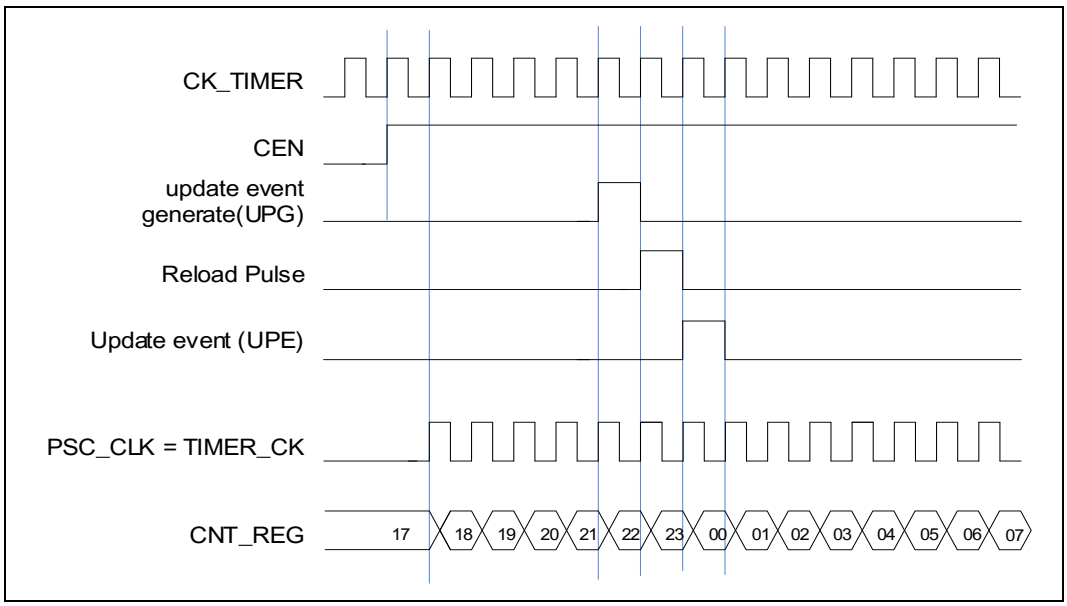
- SMC [2:0] = 3'b000. Internal clock CK_TIMER is selected as timer clock source which is from module RCU.

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 module.

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, more details will be introduced later. 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 9-31. 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.

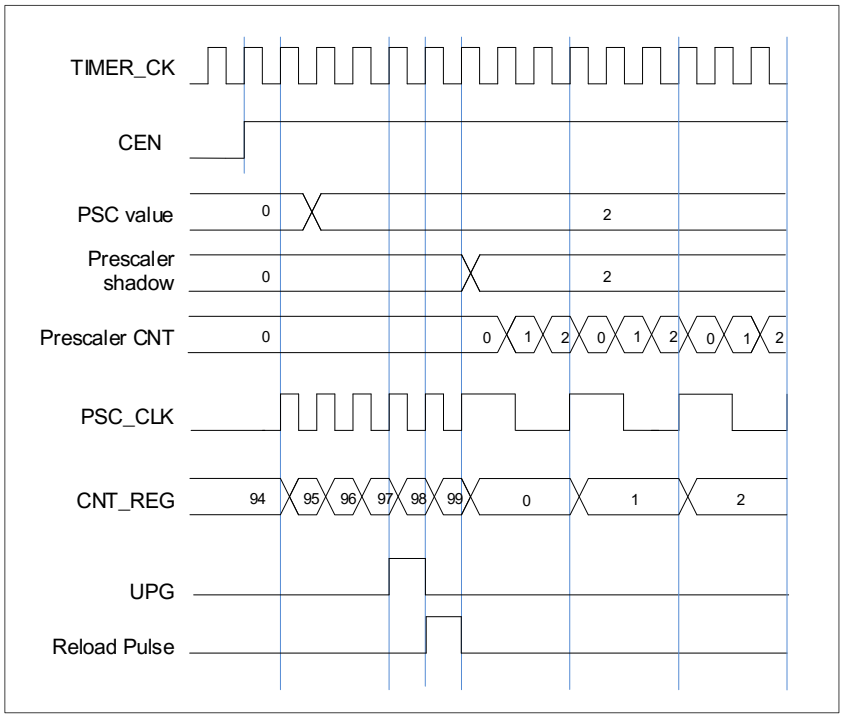
- SMC1= 1'b1 (external clock mode 1). External input ETI is selected as timer clock source.

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 setting the SMC [2:0] to 0x7 and the TRGS[2:0] to 0x7. Note that the ETI signal is derived from the ETI pin sampled by a digital filter. When the ETI signal is selected as the 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 9-32. 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.

Figure 9-33. Timing chart of up counting mode, PSC=0/2 and *Figure 9-34. Timing chart of up counting mode, change TIMERx_CAR on the go.* show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 9-33. Timing chart of up counting mode, PSC=0/2

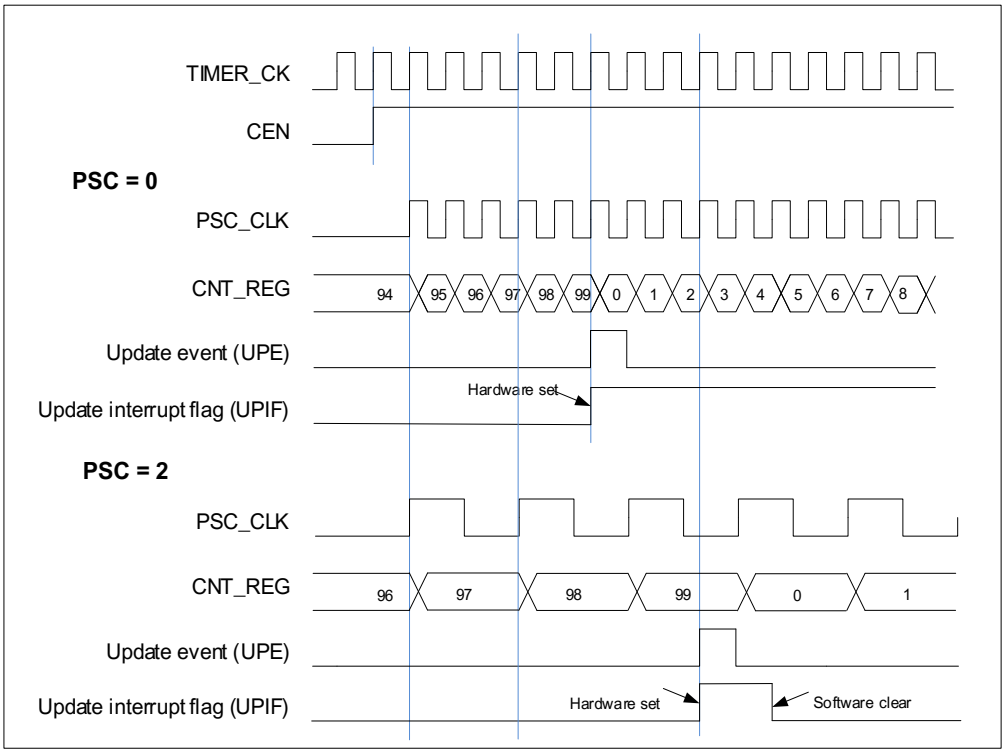
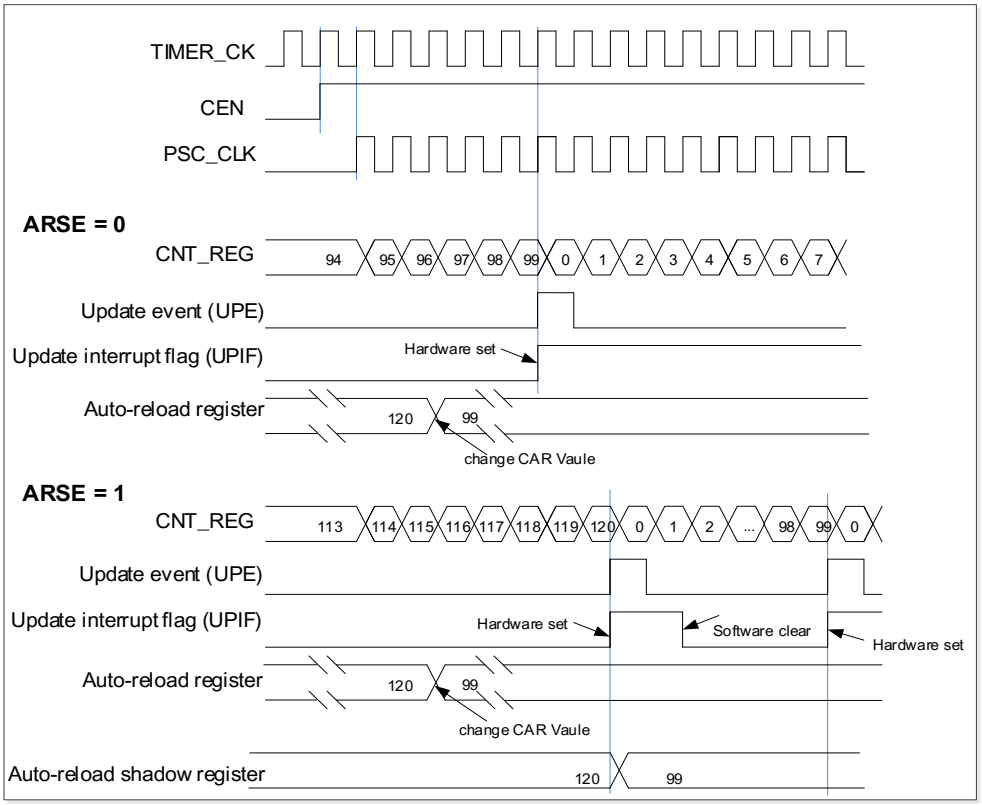


Figure 9-34. Timing chart of up counting mode, change TIMERx_CAR on the go.



Counter down counting

In this mode, the counter counts down continuously from the counter reload value, which is defined in the TIMERx_CAR register, in a count-down direction. Once the counter reaches 0, the counter restarts to count again from the counter reload value. 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 an update event will be generated.

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.

Figure 9-35. Timing chart of down counting mode, PSC=0/2 and *Figure 9-36. Timing chart of down counting mode, change TIMERx_CAR on the go.* show some examples of the counter behavior for different clock frequencies when TIMERx_CAR=0x99.

Figure 9-35. Timing chart of down counting mode, PSC=0/2

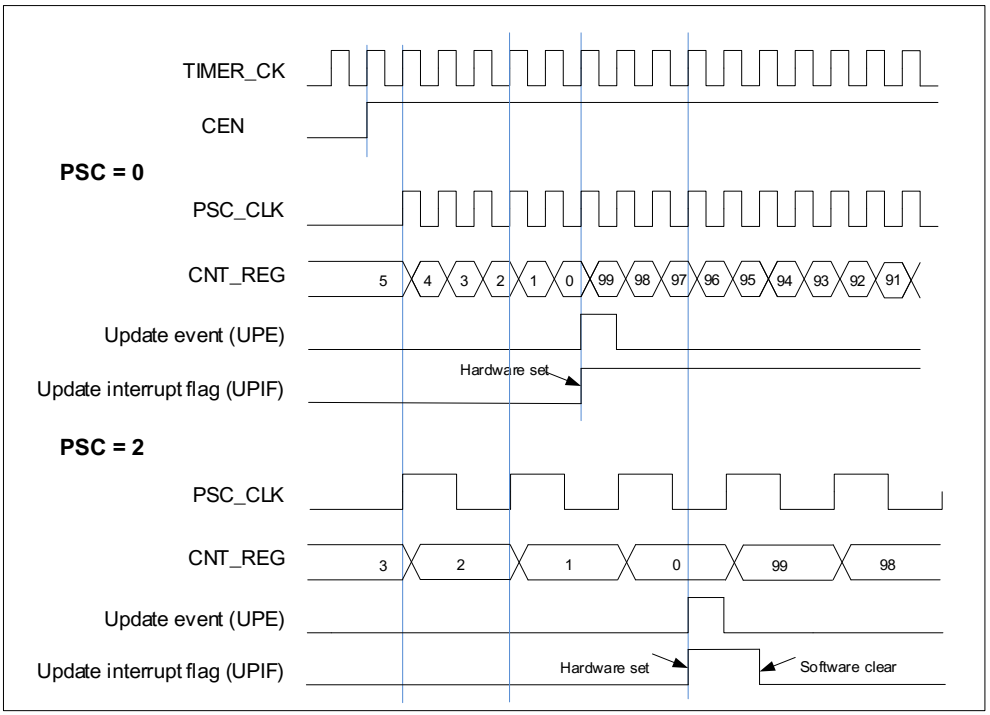
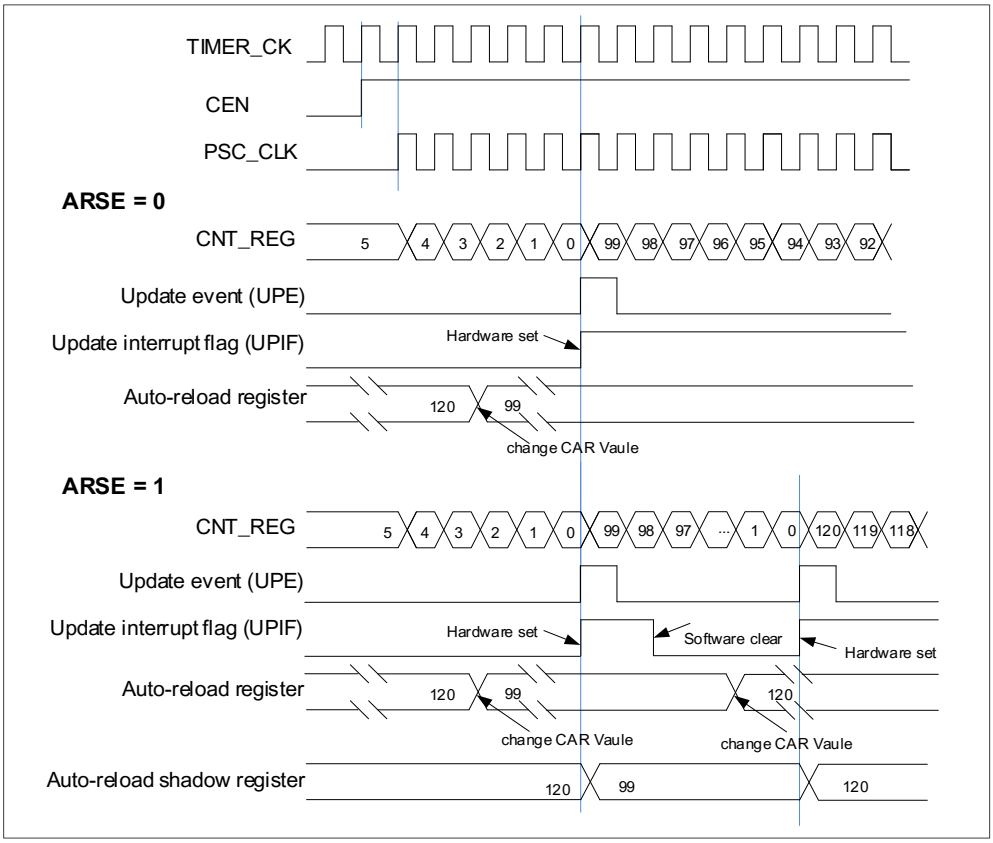


Figure 9-36. Timing chart of down counting mode, change TIMERx_CAR on the go.



Counter center-aligned counting

In the center-aligned counting 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 mode 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 generate an update event irrespective of whether the counter is counting up or down in the center-aligned counting mode.

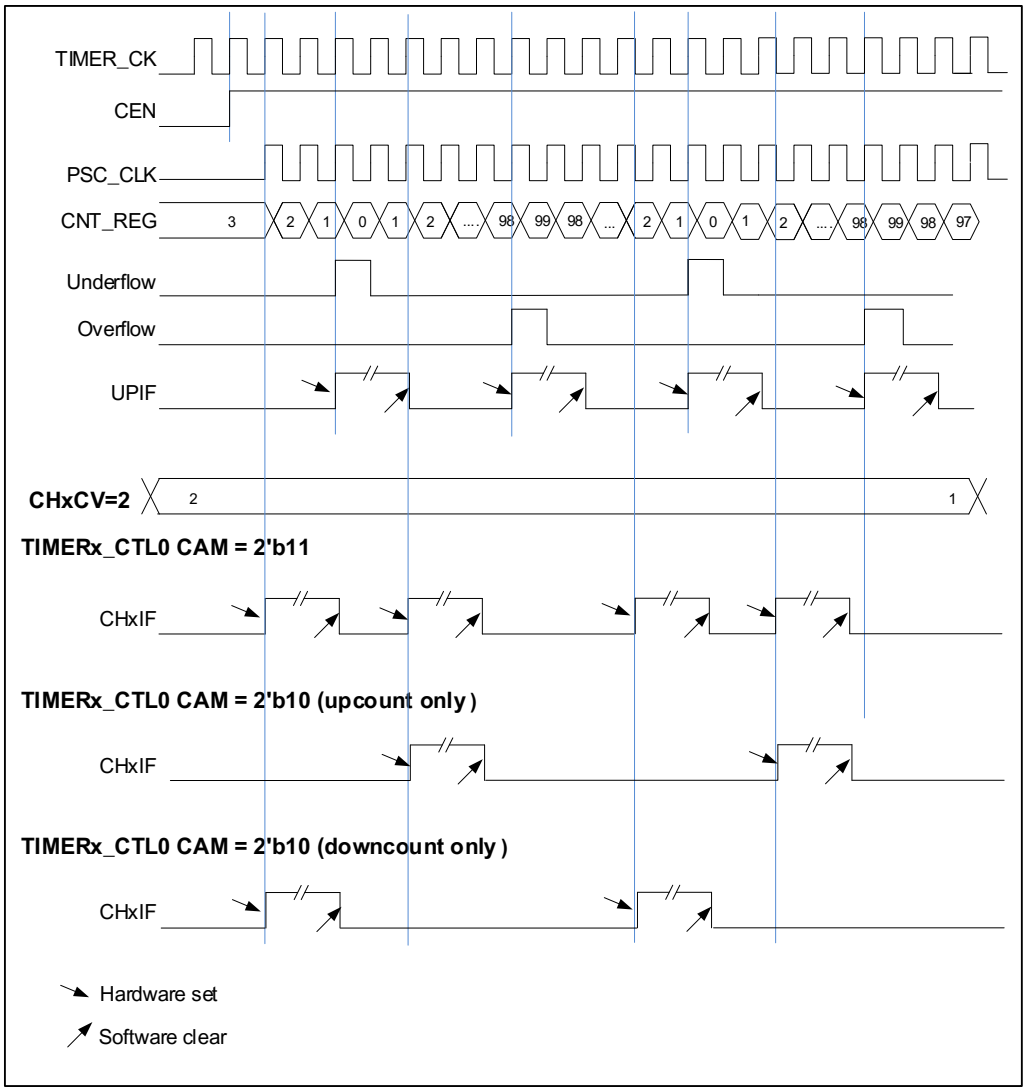
The UPIF bit in the TIMERx_INTF register will be set to 1 either when an underflow event or an overflow event occurs. While the CHxIF bit is associated with the value of CAM in TIMERx_CTL0. The details refer to ***Figure 9-37. Timing chart of center-aligned counting mode***.

If the UPDIS bit in the TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the registers (auto-reload register, prescaler register) are updated.

Figure 9-37. Timing chart of center-aligned counting mode shows the example of the counter behavior when TIMERx_CAR=0x99. TIMERx_PSC=0x0.

Figure 9-37. Timing chart of center-aligned counting mode



Input capture and output compare channels

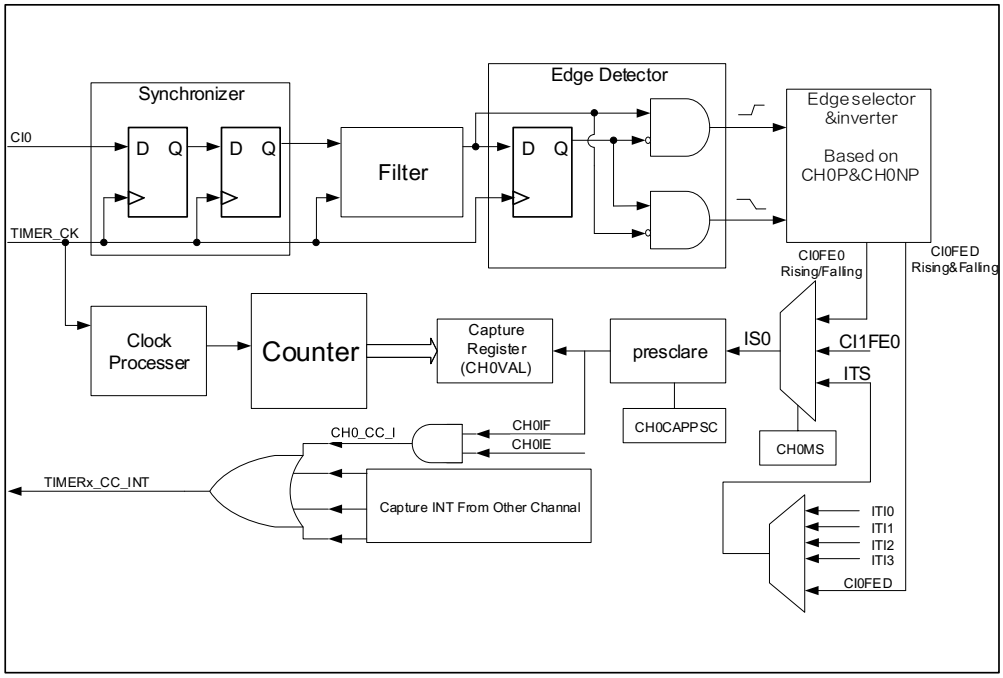
The general level0 Timer has four independent channels which can be used as capture inputs or compare match outputs. Each channel is built around a channel capture compare register including an input stage, channel controller and an output stage.

- **Channel input capture function**

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 it is enabled when CHxIE=1.

Figure 9-38. Channel input capture principle



The input signals of channelx (CIx) can be the TIMERx_CHx signal or the XOR signal of the TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 signals. First, the input signal of channel (CIx) is synchronized to TIMER_CK signal, and then sampled by a digital filter to generate a filtered input signal. Then through the edge detector, the rising or falling edge is detected by configuring CHxP bit. The input capture signal can also be selected from the input signal of other channel or the internal trigger signal by configuring CHxMS bits. The IC prescaler makes several input events generate one effective capture event. On the capture event, TIMERx_CHxCV will store the value of counter.

So, the process can be divided into several steps as below:

Step1: Filter configuration (CHxCAPFLT in TIMERx_CHCTL0).

Based on the input signal and quality of requested signal, configure compatible CHxCAPFLT.

Step2: Edge selection (CHxP/CHxNP in TIMERx_CHCTL2).

Rising edge or falling edge, choose one by configuring CHxP/CHxNP bits.

Step3: Capture source selection (CHxMS in TIMERx_CHCTL0)

As soon as selecting one input capture source by CHxMS, the channel must be set 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 to get the interrupt and DMA request.

Step5: Capture enable (CHxEN in TIMERx_CHCTL2)

Result: When the wanted input signal is captured, TIMERx_CHxCV will be set by counter's value and CHxIF is asserted. If the CHxIF is 1, the CHxOF will also be asserted. The interrupt and DMA request will be asserted or not based on the configuration of CHxIE and CHxDEN in TIMERx_DMAINTEN.

Direct generation: A DMA request or interrupt is generated by setting CHxG 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 connects to CI0 input. Select CI0 as channel 0 capture signals by setting CH0MS to 2'b01 in the channel control register (TIMERx_CHCTL0) and set capture on rising edge. Select CI0 as channel 1 capture signal by setting CH1MS to 2'b10 in the channel control register (TIMERx_CHCTL0) and set capture on falling edge. The counter is set to restart mode and is restarted on channel 0 rising edge. Then the TIMERX_CH0CV can measure the PWM period and the TIMERx_CH1CV can measure the PWM duty cycle.

- **Channel output compare function**

Figure 9-39. Channel output compare principle (x=0, 1, 2, 3)

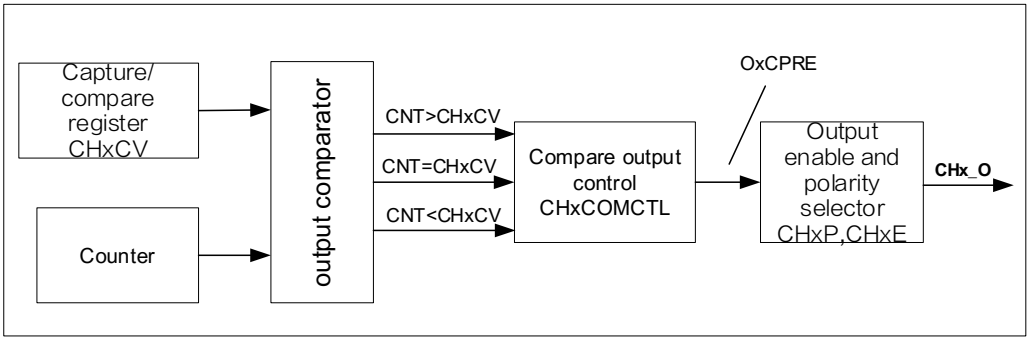


Figure 9-39. Channel output compare principle (x=0, 1, 2, 3) shows the principle circuit of channels output compare function. The relationship between the channel output signal CHx_O and the OxCPRE signal (more details refer to *Channel output prepare signal*) is described as below: The active level of OxCPRE is high, the output level of CHx_O depends on OxCPRE signal, CHxP bit and CHxE bit (please refer to the TIMERx_CHCTL2 register for more details).For example, configure CHxP=0 (the active level of CHx_O is high, the same as OxCPRE), CHxE=1 (the output of CHx_O is enabled):

If the output of OxCPRE is active(high) level, the output of CHx_O is active(high) level.

If the output of OxCPRE is inactive(low) level, the output of CHx_O is active(low) level.

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 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 will be set and the channel (n) interrupt is generated if CHxIE = 1. And the DMA request will be asserted, if CxCDE=1.

So, the process can be divided into 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 polarity by CHxP.
- Enable the output by CHxEN.

Step3: Interrupt/DMA-request enables configuration by CHxIE/CxCDE.

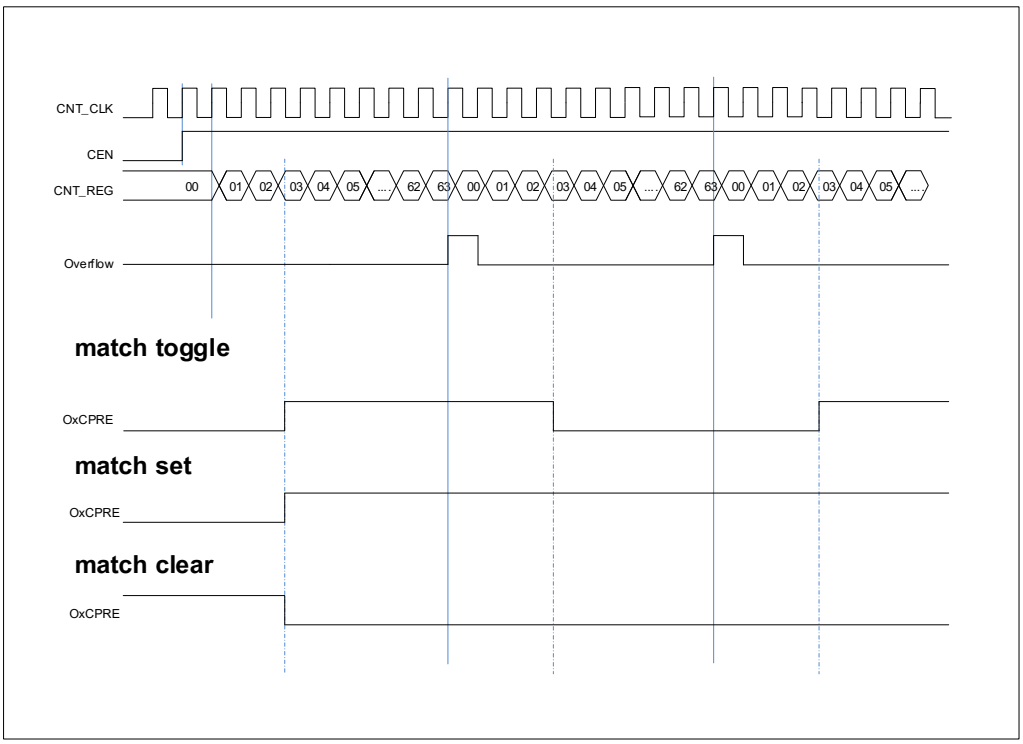
Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV.

The TIMERx_CHxCV can be changed onging to meet the expected waveform.

Step5: Start the counter by configuring CEN to 1.

The timing chart below shows the three compare modes toggle/set/clear. CAR=0x63, CHxVAL=0x3.

Figure 9-40. Output-compare under three modes



Output PWM function

In the output PWM function (by setting the CHxCOMCTL bit to 3'b110 (PWM mode 0) or to 3'b 111(PWM mode 1)), the channel can generate PWM waveform according to the TIMERx_CAR register and TIMERx_CHxCV register.

Based on the counter mode, PWM can also be divided into EAPWM (Edge-aligned PWM) and CAPWM (Center-aligned PWM).

The EAPWM's period is determined by TIMERx_CAR and the duty cycle is determined by TIMERx_CHxCV. ***Figure 9-41. Timing chart of EAPWM*** 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 9-42. Timing chart of CAPWM*** shows the CAPWM output and interrupts waveform.

In up counting mode, if the value of TIMERx_CHxCV is greater than the value of TIMERx_CAR, the output will be always inactive in PWM mode 0 (CHxCOMCTL=3'b110). And if the value of TIMERx_CHxCV is greater than the value of TIMERx_CAR, the output will be always active in PWM mode 1 (CHxCOMCTL=3'b111).

Figure 9-41. Timing chart of EAPWM

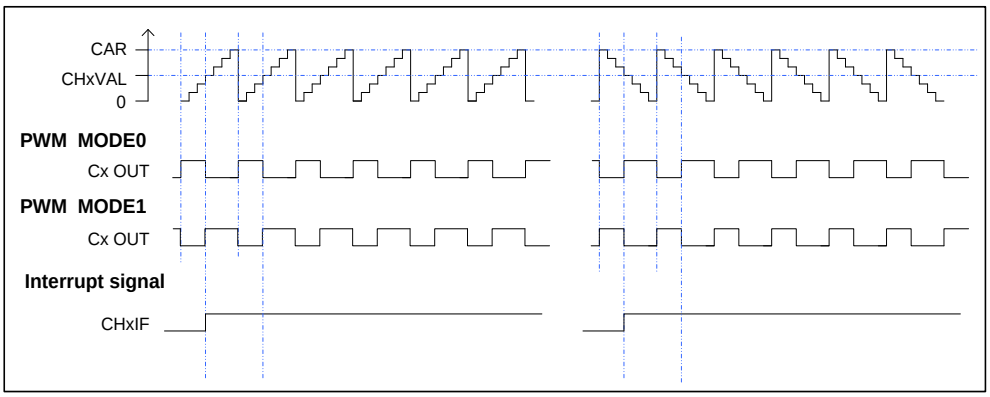
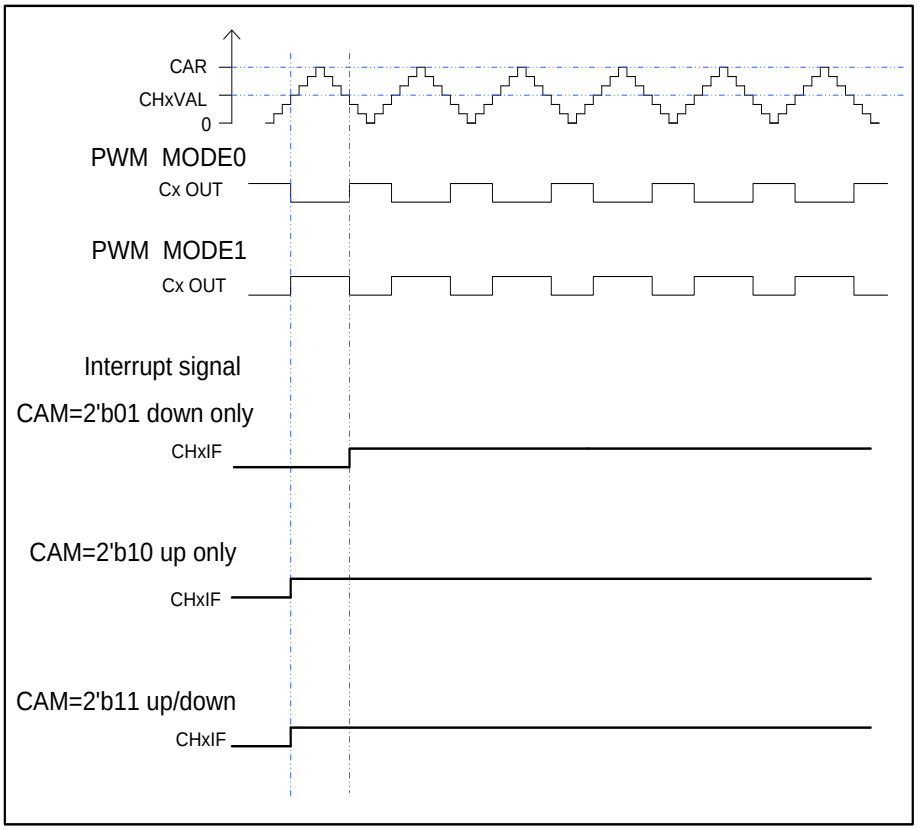


Figure 9-42. Timing chart of CAPWM



Channel output prepare signal

As is shown in ***Figure 9-39. Channel output compare principle (x=0, 1, 2, 3),*** when TIMERx is configured in compare match output mode, a middle signal which is OxCPRE signal (Channel x output prepare signal) will be generated before the channel outputs signal. The OxCPRE signal type is defined by configuring the CHxCOMCTL bit. The OxCPRE signal has several types of output function. These include keeping the original level by configuring the CHxCOMCTL field to 0x00, setting to high by configuring the CHxCOMCTL field to 0x01, setting to low by configuring the CHxCOMCTL field to 0x02 or toggling signal by configuring the CHxCOMCTL field to 0x03 when the counter value matches the content of the TIMERx_CHxCV register.

The PWM mode 0/PWM mode 1 output is another output type of OxCPRE which is setup by configuring 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. Refer to the definition of relative bit for more details.

Another special function of the OxCPRE signal is a forced output which can be achieved by configuring the CHxCOMCTL field to 0x04/0x05. The output can be forced to an inactive/active level irrespective of the comparison condition between the values of the counter and the TIMERx_CHxCV.

Configure the CHxCOMCEN bit to 1 in the TIMERx_CHCTL0 register, the OxCPRE signal can be forced to 0 when the ETIFP signal derived from the external ETI pin is set to a high level. The OxCPRE signal remains low until the next transition to the active state.

Quadrature decoder

Refer to ***Quadrature decoder***.

Hall sensor function

Refer to ***Hall sensor function***.

Master-slave management

Refer to ***Master-slave management***.

Single pulse mode

Refer to ***Single pulse mode***.

Timers interconnection

Refer to ***Timers interconnection***.

Timer DMA mode

Timer DMA mode is the function that configures timer's register by DMA module. The relative registers are TIMERx_DMACFG and TIMERx_DMATB. Corresponding DMA request bit should be asserted to enable DMA request for internal interrupt event. TIMERx will send a request to DMA when the interrupt event occurs. DMA is configured to M2P (memory to peripheral) mode and the address of TIMERx_DMATB is configured to PADDR (peripheral base address), then DMA will access the TIMERx_DMATB. In fact, TIMERx_DMATB register 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), the timer sends only one DMA request. 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 and DMATA+0xC at the next 3 accesses to TIMERx_DMATB. In a word, one-time DMA internal interrupt event asserts, (DMATC+1) times request will be sent by TIMERx.

If one more DMA request event occurs, TIMERx will repeat the process above.

Timer debug mode

When the Cortex®-M23 is halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register set to 1, the TIMERx counter stops.

9.2.5. TIMERx register definition (x=1, 2)

TIMER1 base address: 0x4000 0000

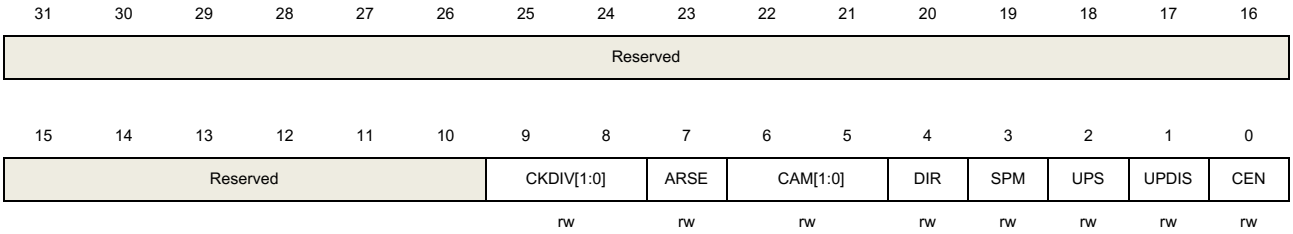
TIMER2 base address: 0x4000 0400

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
6:5	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 quadrature decoder 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, pause mode and quadrature decoder 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
GATCTL	Reserved							TI0S	MMC[2:0]			DMAS	Reserved		
								rw	rw			rw			

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15	GATCTL	Timer gate control 0: Timer enabled when CEN=1 1: Timer enabled when CEN=1 and ex_int is active
14:8	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, CH1 and 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 in master mode to slave timers 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															
rw															

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]			OCRC	SMC[2:0]		
rw	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 ETIF 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_{CK_TIMER}
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	OCRC	OCPRE clear source selection 0: OCPRE clear source is connected to the OCPRE_CLR input 1: OCPRE clear source is connected to ETIFP
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. Because CI0F_ED outputs 1 pulse for each transition on CI0F, and the pause mode checks the level of the trigger signal, when CI0F_ED is selected as the trigger input, the pause mode must not be used.

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
31:16	Reserved	Must be kept at reset value
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	CH3IF	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 's capture/compare interrupt enable

		Refer to CH0IF description
3	CH2IF	Channel 2 's capture/compare interrupt enable Refer to CH0IF description
2	CH1IF	Channel 1 's capture/compare interrupt flag Refer to CH0IF description
1	CH0IF	Channel 0 's capture/compare interrupt flag This flag is set by hardware and cleared by software. When channel 0 is in input mode, this flag is set when a capture event occurs. When channel 0 is in output mode, this flag is set when a compare event occurs. If Channel0 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 on an update event 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 is 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's capture or compare event generation

		Refer to CH0G description
3	CH2G	Channel 2's capture or compare event generation Refer to CH0G description
2	CH1G	Channel 1's capture or compare event generation Refer to CH0G description
1	CH0G	Channel 0's capture or compare event generation This bit is set by software in order to generate a capture or compare event in channel 0, it is automatically cleared by hardware. When this bit is set, the CH1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. In addition, if channel 1 is configured in input mode, the current value of the counter is captured in TIMERx_CH0CV register, and the CH0OF flag is set if the CH0IF flag was already high. 0: No generate a channel 1 capture or compare event 1: Generate a channel 1 capture or compare event
0	UPG	This bit can be set by software, and cleared by hardware automatically. When this bit is set, the counter is cleared if the center-aligned or up counting mode is selected, else (down counting) 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 OCPRE clear source signal is detected as high level, the O0CPRE signal will be cleared. The OCPRE clear source signal is selected by OCRC bit in TIMERx_SMCFG register. 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.
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 0: Channel 0 output quickly compare disable. Only the valid edge of the comparison result will generate a comparison match, and CH0_O is set to the compare level. The minimum delay from an edge on the result of the comparison to activate CH0_O output is 5 clock cycles. 1: Channel 0 output quickly compare enable. Both the valid edge of the trigger input signal and the comparison result will generate a comparison match, and CH0_O is set to the compare level.The minimum delay from an edge on the trigger input to activate CH0_O output is 3 clock cycles.
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:** 1) When CH0MS[1:0]=11, it is necessary to select an internal trigger input through TRGS bits in TIMERx_SMCFG register. 2) When the ETI function is used in TIMER1, CH0MS in TIMER1 must not be 2'b00.

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 OCPRE clear source signal is detected as high level, the O0CPRE signal will be cleared. The OCPRE clear source signal is selected by OCRC bit in TIMERx_SMCFG register. 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.
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)
2	CH2COMFEN	Channel 2 output compare fast enable 0: Channel 2 output quickly compare disable. Only the valid edge of the comparison result will generate a comparison match, and CH2_O is set to the compare level. The minimum delay from an edge on the result of the comparison to activate CH2_O output is 5 clock cycles. 1: Channel 2 output quickly compare enable. Both the valid edge of the trigger input signal and the comparison result will generate a comparison match, and CH2_O is set to the compare level.The minimum delay from an edge on the trigger input to activate CH2_O output is 3 clock cycles.
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 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
CH3NP	Reserved	CH3P	CH3EN	CH2NP	Reserved	CH2P	CH2EN	CH1NP	Reserved	CH1P	CH1EN	CH0NP	Reserved	CH0P	CH0EN
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	CH3NP	Channel 3 complementary output polarity Refer to CH0NP description
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	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	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 CH0EN 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]: CIxFE0's rising edge is the active signal for capture or trigger operation in slave mode. And CIxFE0 will not be inverted. [CH0NP==0, CH0P==1]: CIxFE0's falling edge 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]: CIxFE0's falling and rising edge are both the active signal for capture or trigger operation in slave mode. And CIxFE0 will be not 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) (x=1)

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
CNT[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

CNT[15:0]

rw

Bits	Fields	Descriptions
31:0	CNT[31:0]	This bit-filed indicates the current counter value. Writing to this bit-filed can change the value of the counter.

Counter register (TIMERx_CNT) (x=2)

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-filed indicates the current counter value. Writing to this bit-filed 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) (x=1)

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
CARL[31:16]															

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

rw

Bits	Fields	Descriptions
31:0	CARL[31:0]	Counter auto reload value This bit-filed specifies the auto reload value of the counter. **Note:** When the timer is configured in input capture mode, this register must be configured a non-zero value (such as 0xFFFFFFF) which is larger than user expected value.

Counter auto reload register (TIMERx_CAR) (x=2)

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-filed specifies the auto reload value of the counter. **Note:** When the timer is configured in input capture mode, this register must be configured a non-zero value (such as 0xFFFF) which is larger than user expected value.

Channel 0 capture/compare value register (TIMERx_CH0CV) (x=1)

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
CH0VAL[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH0VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CH0VAL[31:0]	Capture or compare value of channel0 When channel 0 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 0 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Channel 0 capture/compare value register (TIMERx_CH0CV) (x=2)

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 or compare value of channel0 When channel 0 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 0 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Channel 1 capture/compare value register (TIMERx_CH1CV) (x=1)

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
CH1VAL[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CH1VAL[31:0]	Capture or compare value of channel1 When channel 1 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 1 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Channel 1 capture/compare value register (TIMERx_CH1CV) (x=2)

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 or compare value of channel1 When channel 1 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 1 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Channel 2 capture/compare value register (TIMERx_CH2CV) (x=1)

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
CH2VAL[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CH2VAL[31:0]	Capture or compare value of channel 2 When channel 2 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 2 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Channel 2 capture/compare value register (TIMERx_CH2CV) (x=2)

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 or compare value of channel 2 When channel 2 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 2 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Channel 3 capture/compare value register (TIMERx_CH3CV) (x=1)

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
CH3VAL[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CH3VAL[31:0]	Capture or compare value of channel 3 When channel3 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 3 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Channel 3 capture/compare value register (TIMERx_CH3CV) (x=2)

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															

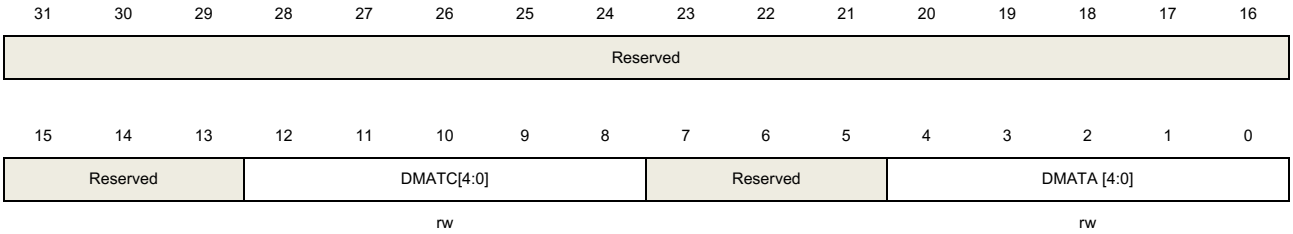
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	CH3VAL[15:0]	Capture or compare value of channel 3 When channel3 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 3 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates 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)



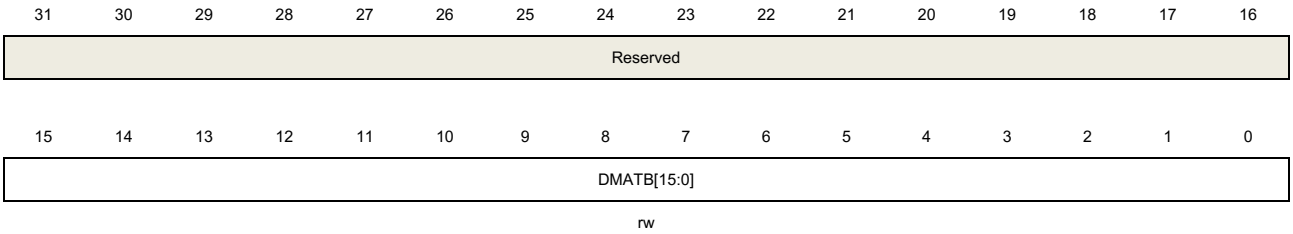
Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:8	DMATC [4:0]	DMA transfer count This filed defines the number(n) of the register that DMA will access(R/W), n = (DMATC [4:0] +1). DMATC [4:0] is from 5'b0_0000 to 5'b1_0001.
7:5	Reserved	Must be kept at reset value.
4:0	DMATA [4:0]	DMA transfer access start address This filed 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)



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.

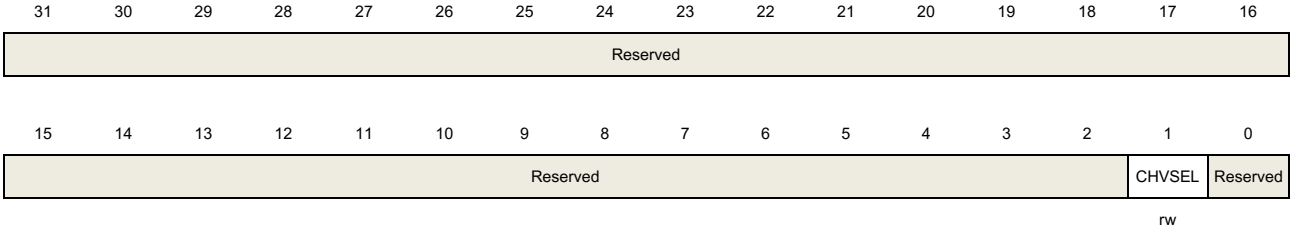
The transfer Timer is calculated by hardware, and ranges from 0 to DMATC.

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. 0: No effect. 1: If write the CHxVAL register, the write value is same as the CHxVAL value, the write access ignored.
0	Reserved	Must be kept at reset value

9.3. General level2 timer (TIMERx, x=13)

9.3.1. Overview

The general level2 timer module (TIMER13) 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.

9.3.2. Characteristics

- Total channel num: 1.
- Counter width: 16 bits.
- Source of count clock: internal clock.
- Counter mode: count up only.
- Programmable prescaler: 16 bits. Factor can be changed ongoing.
- Each channel is user-configurable:

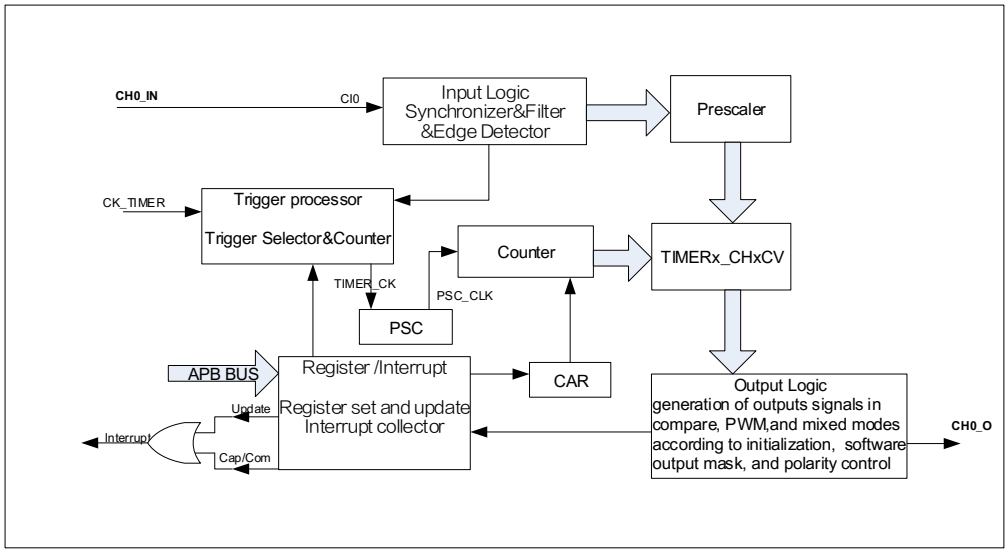
Input capture mode, output compare mode, programmable PWM mode

- Auto-reload function.
- Interrupt output on: update, compare/capture event.

9.3.3. Block diagram

Figure 9-43. General level2 timer block diagram provides details on the internal configuration of the general level2 timer.

Figure 9-43. General level2 timer block diagram



9.3.4. Function overview

Clock source configuration

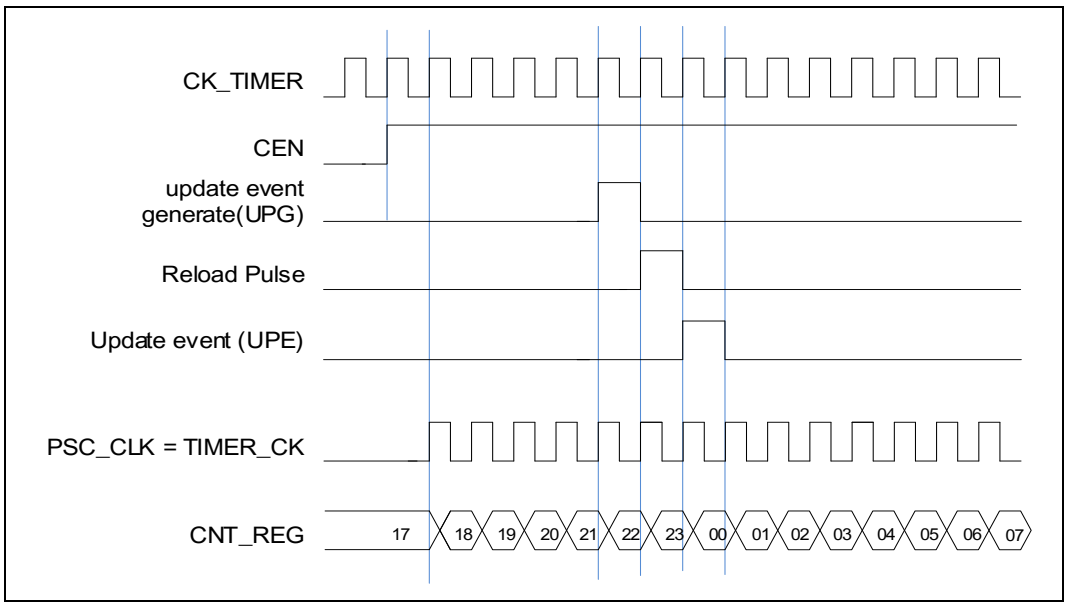
The general level2 TIMER can only being clocked by the CK_TIMER.

- 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.

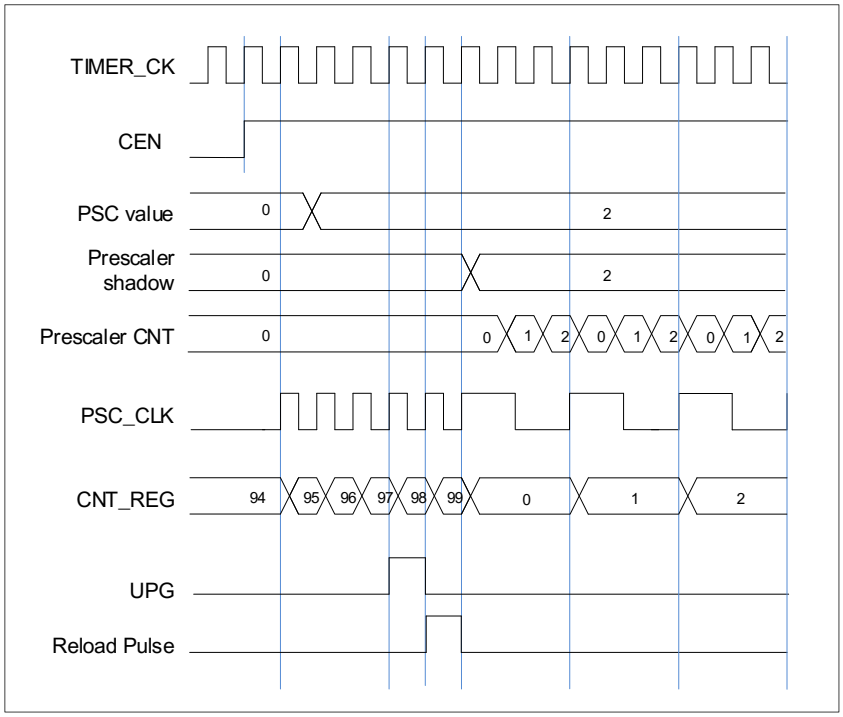
Figure 9-44. Timing chart of internal clock divided by 1



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 9-45. 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.

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 an update event will be generated.

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 9-46. Timing chart of up counting mode, PSC=0/2

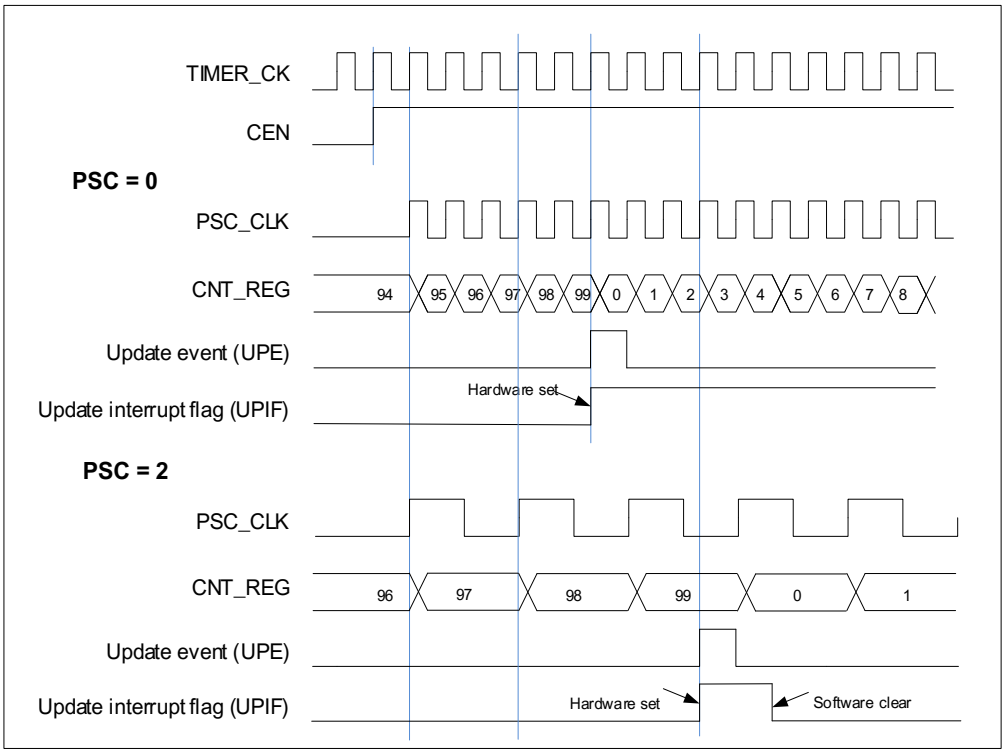
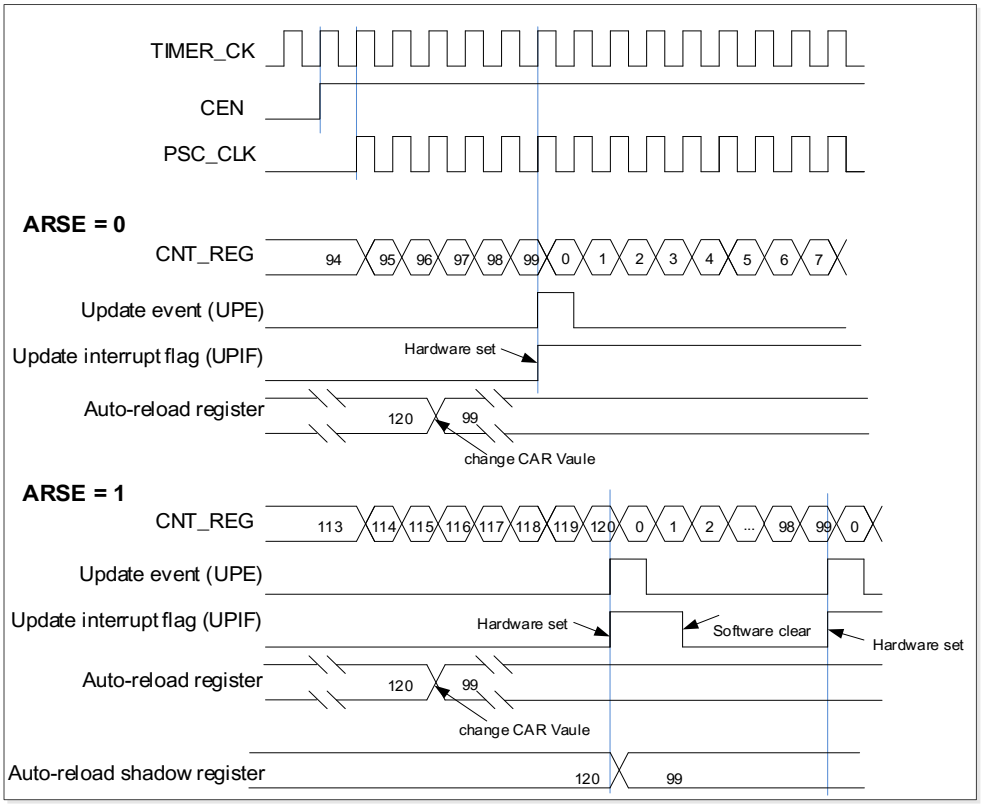


Figure 9-47. Timing chart of up counting mode, change TIMERx_CAR on the go



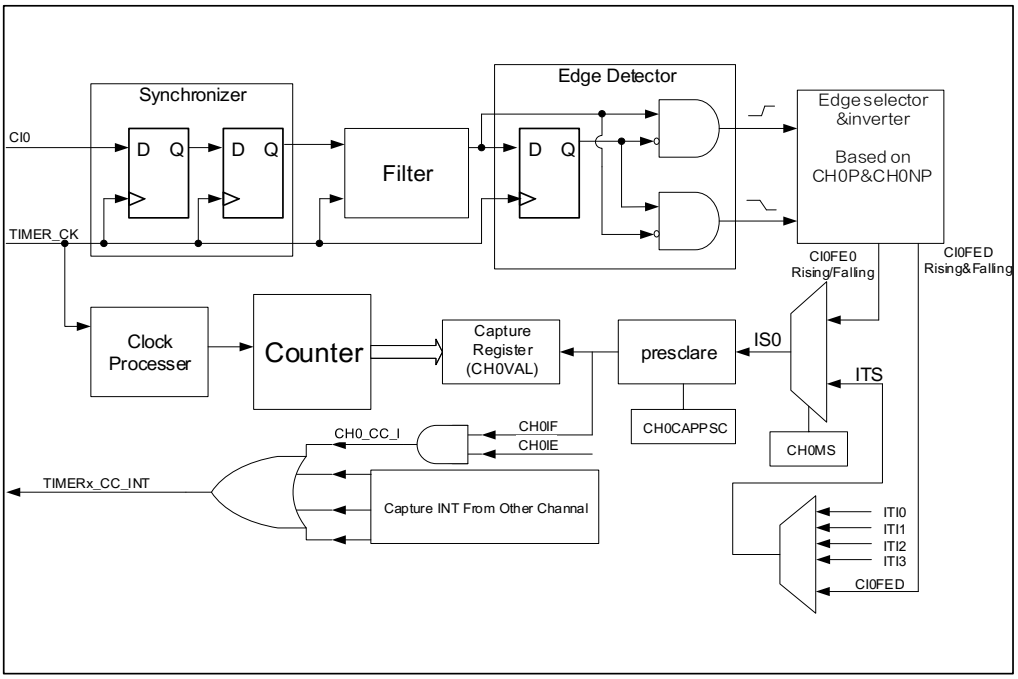
Input capture and output compare channels

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 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 it is enabled when CHxIE=1.

Figure 9-48. Channel input capture principle



First, the input signal of channel (CIx) is synchronized to TIMER_CK signal, and then sampled by a digital filter to generate a filtered input signal. Then through the edge detector, the rising or falling edge is detected by configuring CHxP bit. The input capture signal can also be selected from the input signal of other channel or the internal trigger signal by configuring CHxMS bits. The IC prescaler makes several input events generate one effective capture event. On the capture event, TIMERx_CHxCV will store the value of counter.

So, the process can be divided into several steps as below:

Step1: Filter configuration (CHxCAPFLT in TIMERx_CHCTL0).

Based on the input signal and quality of requested signal, configure compatible CHxCAPFLT.

Step2: Edge selection.(CHxP/CHxNP in TIMERx_CHCTL2).

Rising edge, falling edge or both edges (rising and falling edge), choose one by configuring CHxP/CHxNP bits.

Step3: Capture source selection (CHxMS in TIMERx_CHCTL0).

As soon as selecting one input capture source by CHxMS, the channel must be set 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 to get the interrupt and DMA request.

Step5: Capture enable (CHxEN in TIMERx_CHCTL2).

Result: When the wanted input signal is captured, TIMERx_CHxCV will be set by counter's value and CHxIF is asserted. If the CHxIF is 1, the CHxOF will also be asserted. The interrupt and DMA request will be asserted or not based on the configuration of CHxIE in TIMERx_DMAINTEN.

Direct generation: An interrupt is generated by setting CHxG directly.

- **Output compare mode**

Figure 9-49. Channel output compare principle

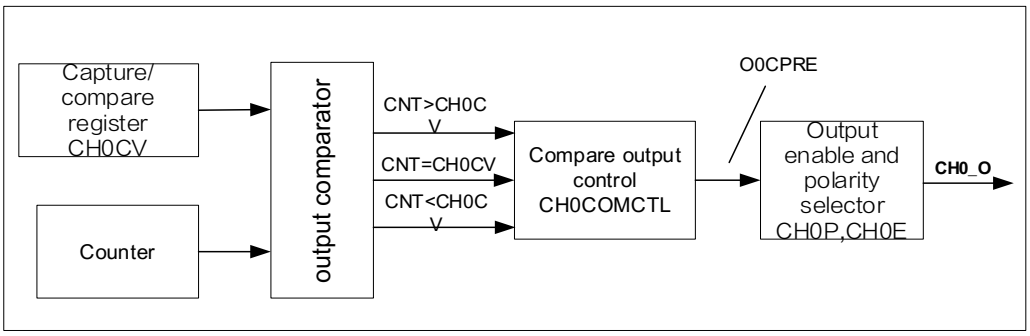


Figure 9-49. Channel output compare principle shows the principle circuit of channels output compare function. The relationship between the channel output signal CHx_O and the OxCPRE signal (more details refer to *Channel output prepare signal*) is described as below: The active level of OxCPRE is high, the output level of CHx_O depends on OxCPRE signal, CHxP bit and CHxE bit (please refer to the TIMERx_CHCTL2 register for more details).For example, configure CHxP=0 (the active level of CHx_O is high, the same as OxCPRE), CHxE=1 (the output of CHx_O is enabled):

If the output of OxCPRE is active(high) level, the output of CHx_O is active(high) level;

If the output of OxCPRE is inactive(low) level, the output of CHx_O is active(low) level.

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 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 will be set and the channel (n) interrupt is generated if CHxIE = 1.

So, the process can be divided into 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 polarity by CHxP.
- Enable the output by CHxEN.

Step3: Interrupt/DMA-request enables configuration by CHxIE.

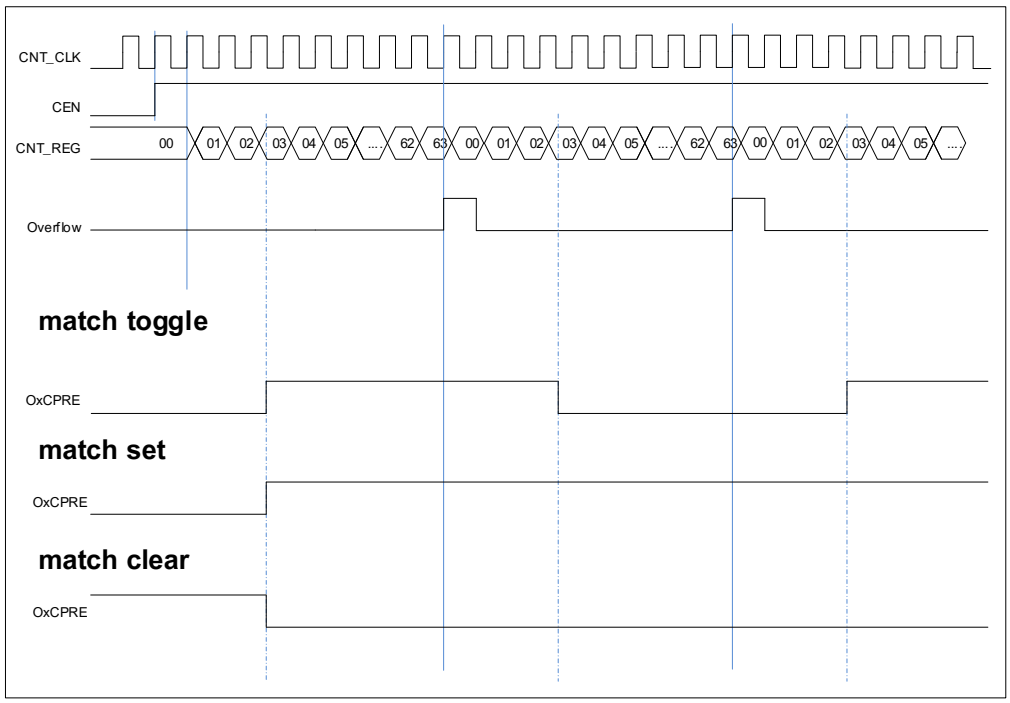
Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV.

The TIMERx_CHxCV can be changed onging to meet the expected waveform.

Step5: Start the counter by configuring CEN to 1.

The timing chart below show the three compare modes toggle/set/clear. CAR=0x63, CHxVAL=0x3

Figure 9-50. Output-compare in three modes



Output PWM function

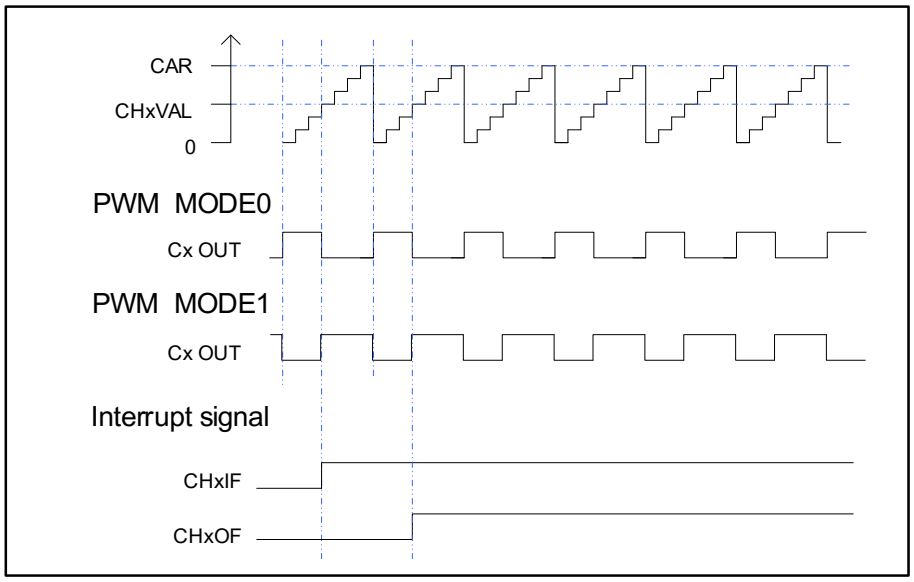
In the output PWM function (by setting the CHxCOMCTL bits to 3'b110 (PWM mode0) or to 3'b 111(PWM mode1), the channel can generate PWM waveform according to the TIMERx_CAR register and TIMERx_CHxCV register.

The period is determined by TIMERx_CAR and duty cycle is determined by TIMERx_CHxCV. *Figure 9-51. PWM mode timechart* shows the PWM output mode 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 9-51. PWM mode timechart



Channel output prepare signal

As is shown in ***Figure 9-49. Channel output compare*** , when TIMERx is configured in compare match output mode a middle signal which is OxCPRE signal (Channel x output prepare signal) will be generated before the channel outputs signal. The OxCPRE signal type is defined by configuring the CHxCOMCTL bit. The OxCPRE signal has several types of output function. These include keeping the original level by configuring the CHxCOMCTL field to 0x00, setting to high by configuring the CHxCOMCTL field to 0x01, setting to low by configuring the CHxCOMCTL field to 0x02 or toggling signal by configuring the CHxCOMCTL field to 0x03 when the counter value matches the content of the TIMERx_CHxCV register.

The PWM mode 0/PWM mode 1 output is another output type of OxCPRE which is setup by configuring 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. Refer to the definition of relative bit for more details.

Another special function of the OxCPRE signal is a forced output which can be achieved by configuring the CHxCOMCTL field to 0x04/0x05. The output can be forced to an inactive/active level irrespective of the comparison condition between the values of the counter and the TIMERx_CHxCV.

Timer debug mode

When the Cortex®-M23 halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register set to 1, the TIMERx counter stops.

9.3.5. TIMERx register definition (x=13)

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)

31	30	29	28	27	26	25	24	23	22	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	Reserved				UPS	UPDIS	CEN
						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
6:3	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, pause mode and quadrature decoder 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

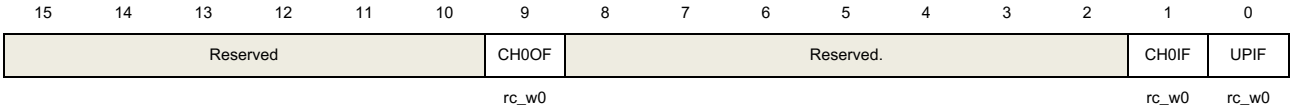
Interrupt flag register (TIMERx_INTF)

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															



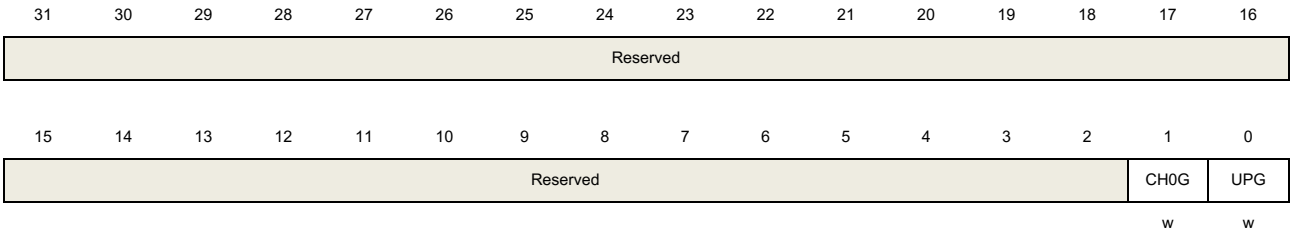
Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	CH0OF	Channel 0 over capture flag When channel 0 is configured in input mode, this flag is set by hardware when a capture event occurs while CH0IF flag has already been set. This flag is cleared by software. 0: No over capture interrupt occurred 1: Over capture interrupt occurred
8:2	Reserved	Must be kept at reset value.
1	CH0IF	Channel 0 's capture/compare interrupt flag This flag is set by hardware and cleared by software. When channel 0 is in input mode, this flag is set when a capture event occurs. When channel 0 is in output mode, this flag is set when a compare event occurs. If Channel0 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 on an update event 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)



Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.

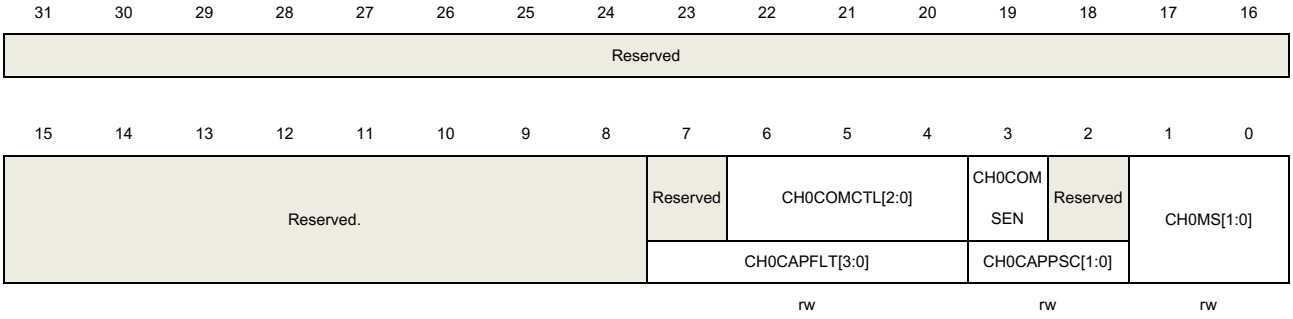
1	CH0G	Channel 0's capture or compare event generation This bit is set by software in order to generate a capture or compare event in channel 0, it is automatically cleared by hardware. When this bit is set, the CH1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. In addition, if channel 1 is configured in input mode, the current value of the counter is captured in TIMERx_CH0CV register, and the CH0OF flag is set if the CH0IF flag was already high. 0: No generate a channel 1 capture or compare event 1: Generate a channel 1 capture or compare event
0	UPG	This bit can be set by software, and cleared by hardware automatically. 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)



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	Reserved	Must be kept at reset value.
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 01: Channel 0 is programmed as input, IS0 is connected to CI0FE0 10: Reserved. 11: 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

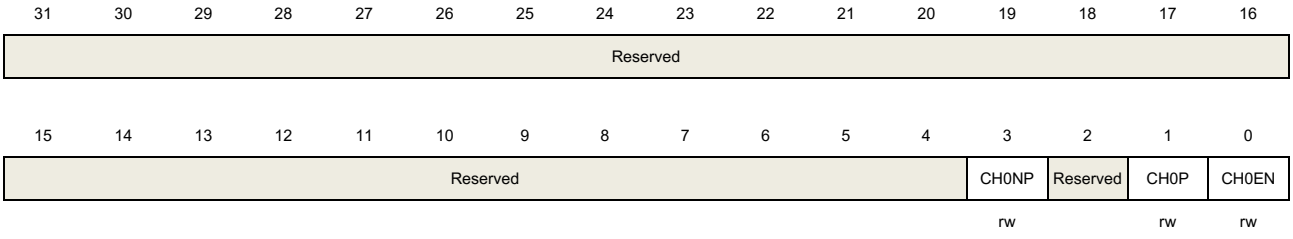
Same as output compare mode

Channel control register 2 (TIMERx_CHCTL2)

Address offset: 0x20

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value

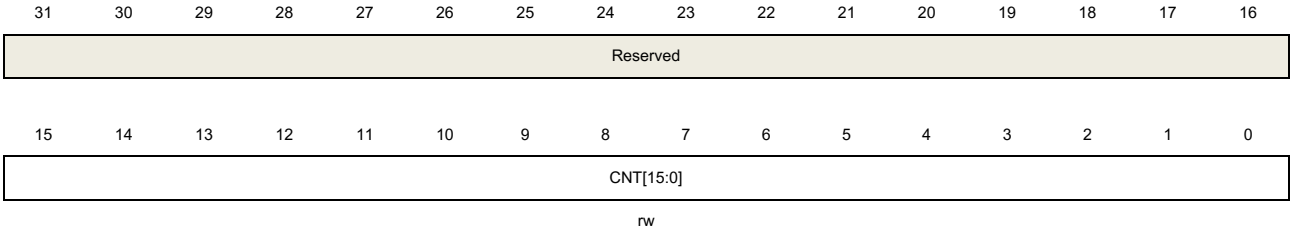
3	CH0NP	Channel 0 complementary output polarity When channel 0 is configured in output mode, this bit specifies the complementary output signal polarity. 0: Channel 0 active high 1: Channel 0 active low When channel 0 is configured in input mode, together with CH0P, this bit is used to define the polarity of CI0.
2	Reserved	Must be kept at reset value
1	CH0P	Channel 0 capture/compare 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]: CIxFE0's rising edge is the active signal for capture or trigger operation in slave mode. And CIxFE0 will not be inverted. [CH0NP==0, CH0P==1]: CIxFE0's falling edge 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]: CIxFE0's falling and rising edge are both the active signal for capture or trigger operation in slave mode. And CIxFE0 will be not 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)



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	CNT[15:0]	This bit-filed indicates the current counter value. Writing to this bit-filed 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-filed specifies the auto reload value of the counter. **Note:** When the timer is configured in input capture mode, this register must be

configured a non-zero value (such as 0xFFFF) which is larger than user expected value.

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 or compare value of channel0 When channel 0 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 0 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Channel input remap register(TIMERx_IRMP)

Address offset: 0x50

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														CI0_RMP[1:0]	
														rw	

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value
1:0	CI0_RMP[1:0]	Channel 0 input remap 00: Channel 0 input is connected to GPIO(TIMER13_CH0) 01: Reserved 10: Channel 0 input is connected to HXTAL/32 clock

11: Channel 0 input is connected to CKOUTSEL

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. 0: No effect. 1: If write the CHxVAL register, the write value is same as the CHxVAL value, the write access ignored.
0	Reserved	Must be kept at reset value.

9.4. General level3 timer (TIMERx, x=14)

9.4.1. Overview

The general level3 timer module (TIMER14) is a two-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 general level3 timer has a 16-bit counter that can be used as an unsigned counter.

In addition, the general level3 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 counters incrementing in unison.

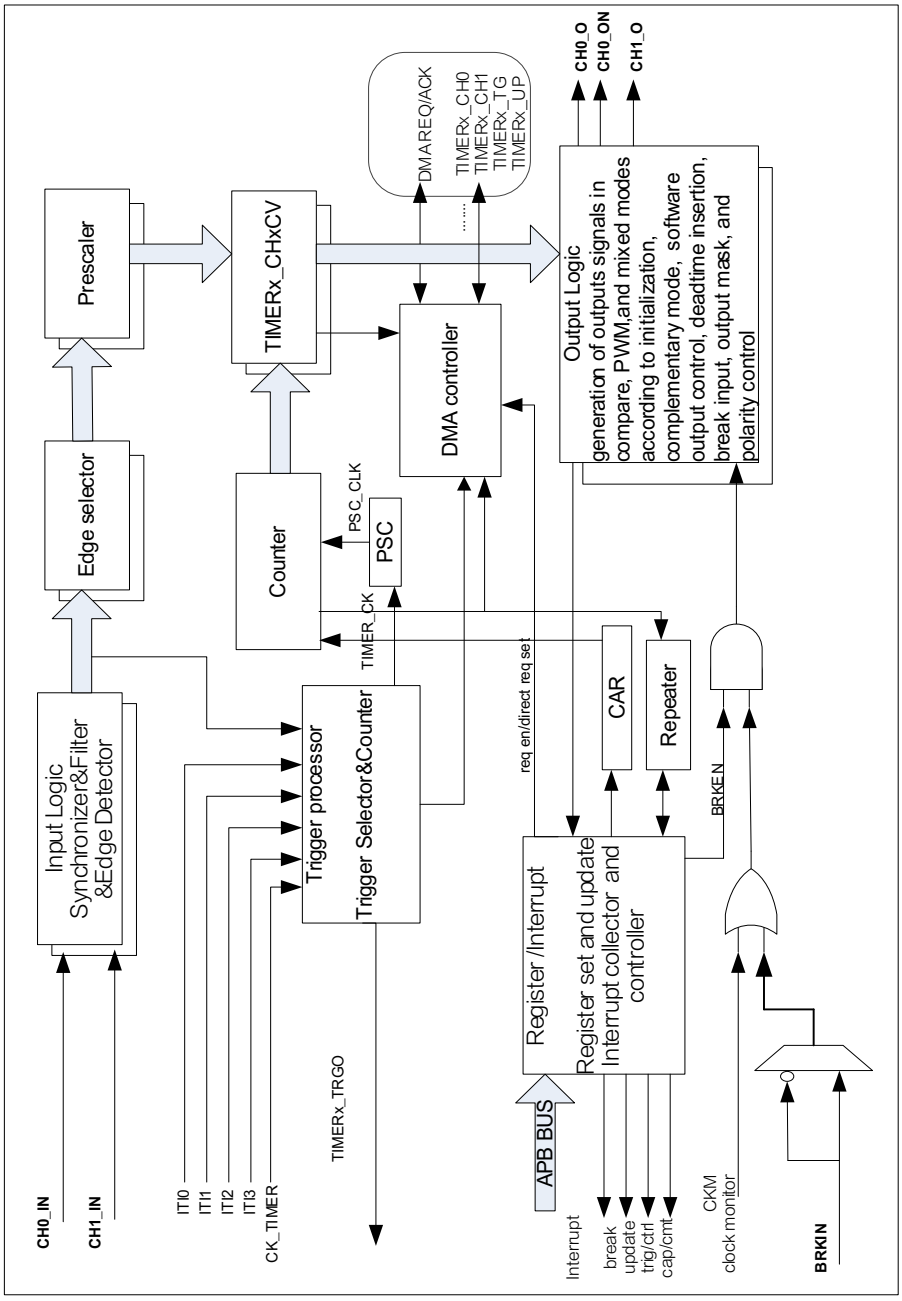
9.4.2. Characteristics

- Total channel num: 2.
- Counter width: 16 bits.
- Clock source of timer is selectable: internal clock, internal trigger, external input.
- Counter modes: count up 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, single pulse mode
- Programmable dead time insertion.
- Auto reload function.
- Programmable counter repetition function.
- Break input.
- Interrupt output or DMA request on: update event, trigger event, compare/capture event, commutation event and break input.
- Daisy chaining of timer modules allows a single timer to start multiple timers.
- Timer synchronization allows selected timers to start counting on the same clock cycle.
- Timer master-slave management.

9.4.3. Block diagram

Figure 9-52. General level3 timer block diagram provides details of the internal configuration of the general level3 timer.

Figure 9-52. General level3 timer block diagram



9.4.4. Function overview

Clock source configuration

The clock source of the advanced timer can be either the CK_TIMER or an alternate clock source controlled by SMC bits (TIMERx_SMCFG bit [2:0]).

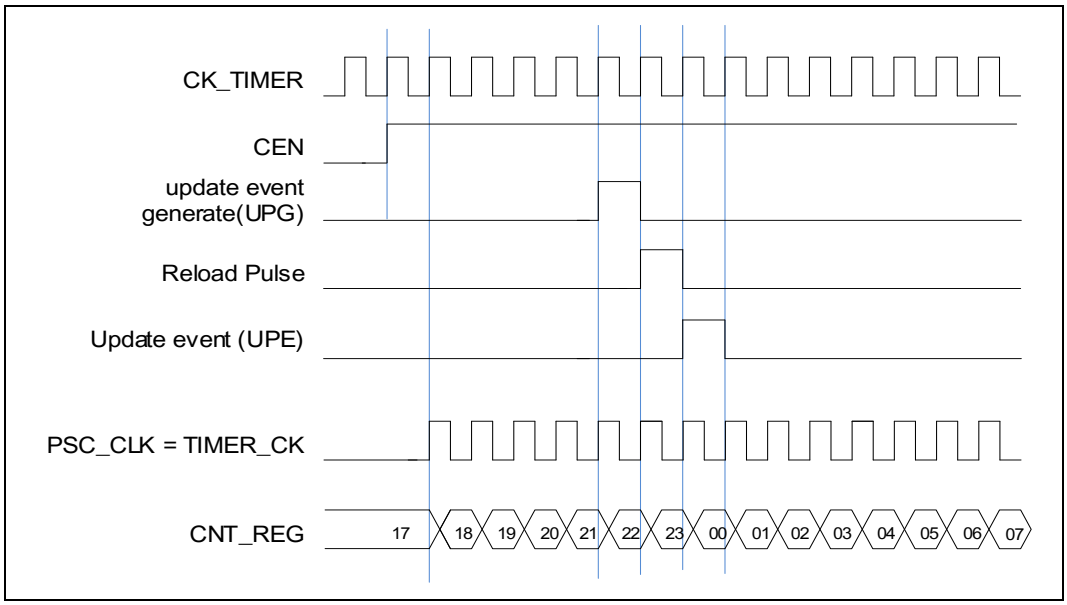
- SMC [2:0] == 3'b000. Internal clock CK_TIMER is selected as timer clock source which is from module RCU.

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 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 9-53. 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

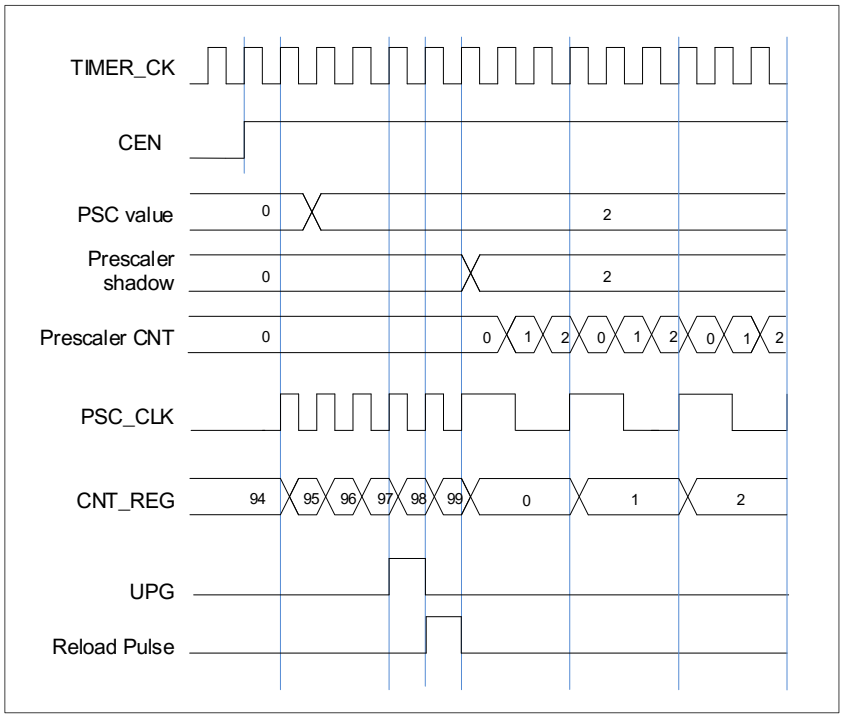
The TIMER_CK, which drives counter's prescaler to count, can be triggered by the event of rising or falling edge on the external pin TIMERx_CH0/TIMERx_CH1. This mode can be selected by setting SMC [2:0] to 0x7 and the TRGS [2:0] to 0x4, 0x5 or 0x6.

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.

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 9-54. 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, counter auto reload register, prescaler register) are updated.

Figure 9-55. 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 9-55. Timing chart of up counting mode, PSC=0/2

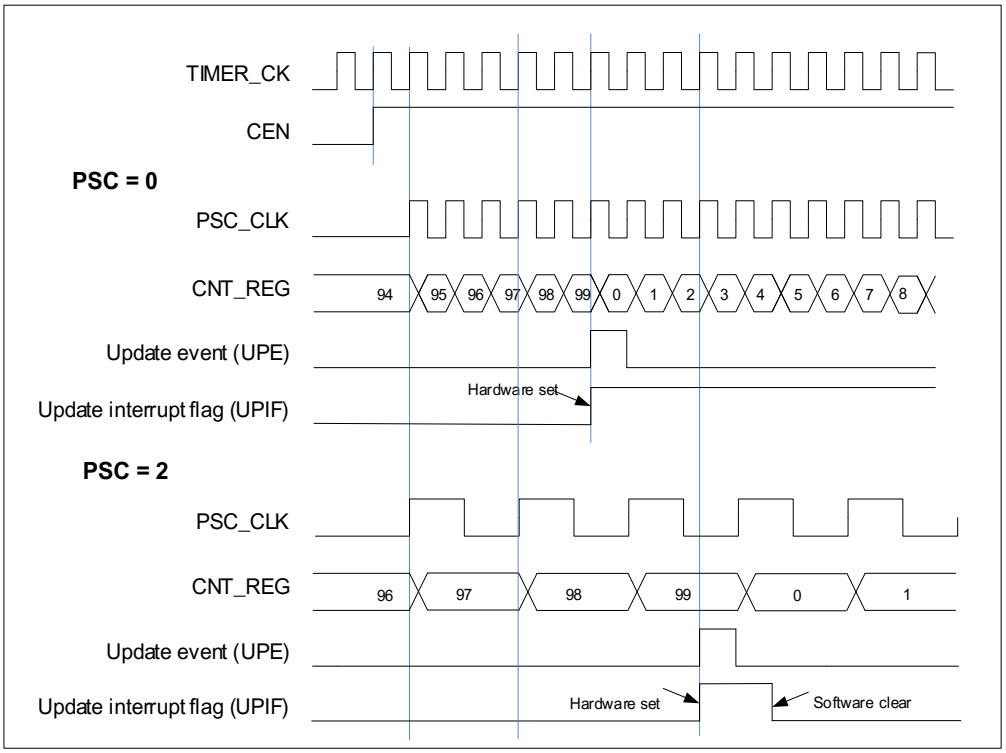
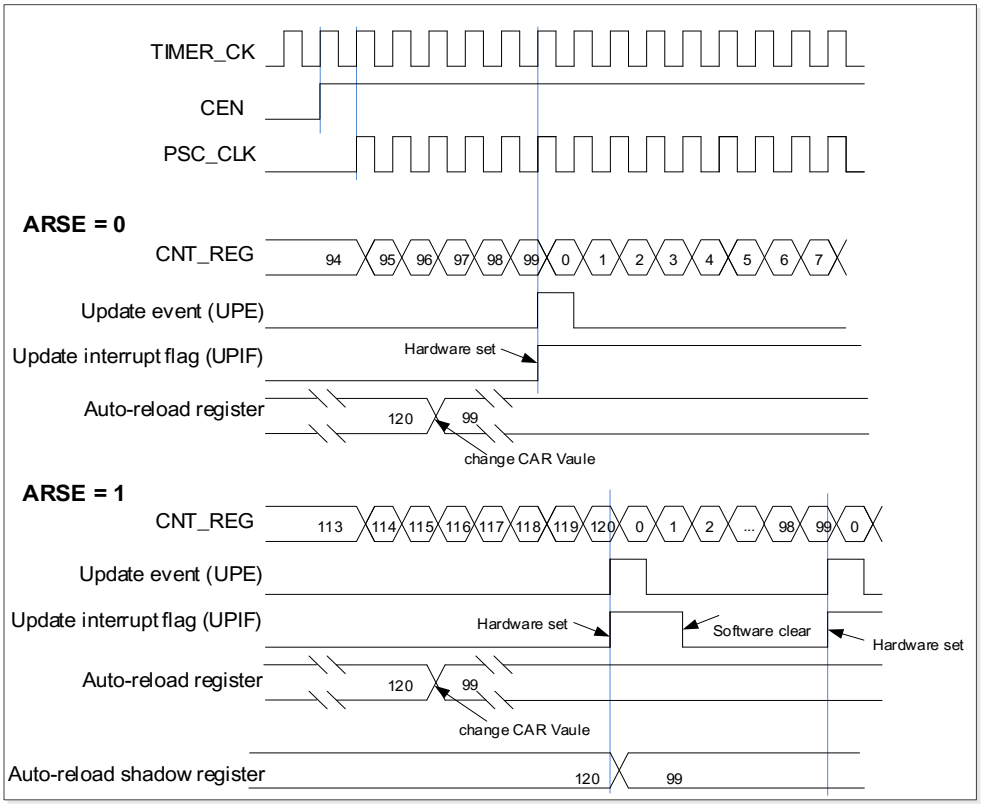


Figure 9-56. Timing chart of up counting mode, change TIMERx_CAR on the go

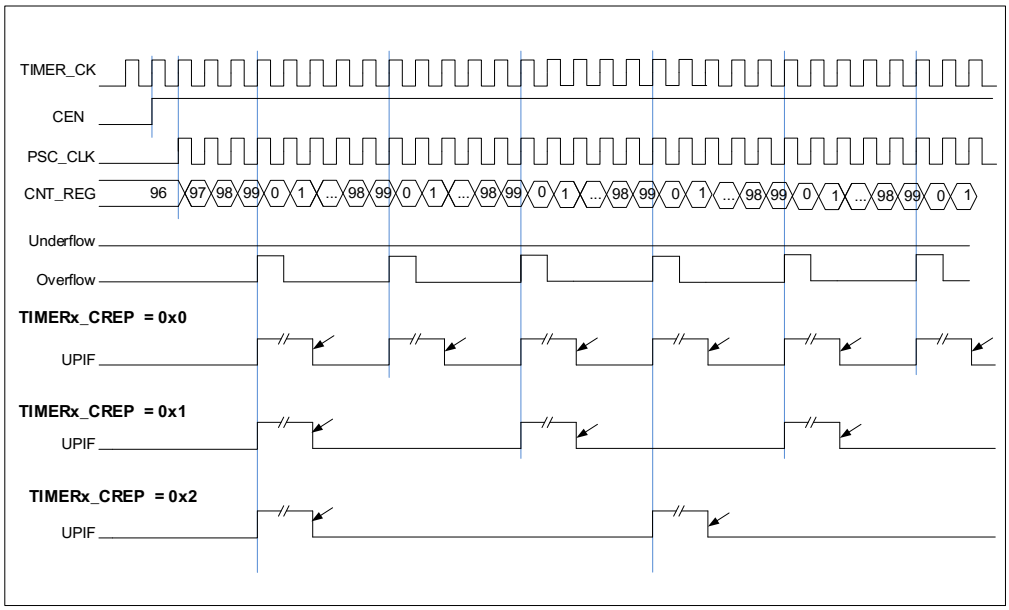


Update event (from overflow/underflow) rate configuration

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 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.

Figure 9-57. Repetition counter timing chart of up counting mode



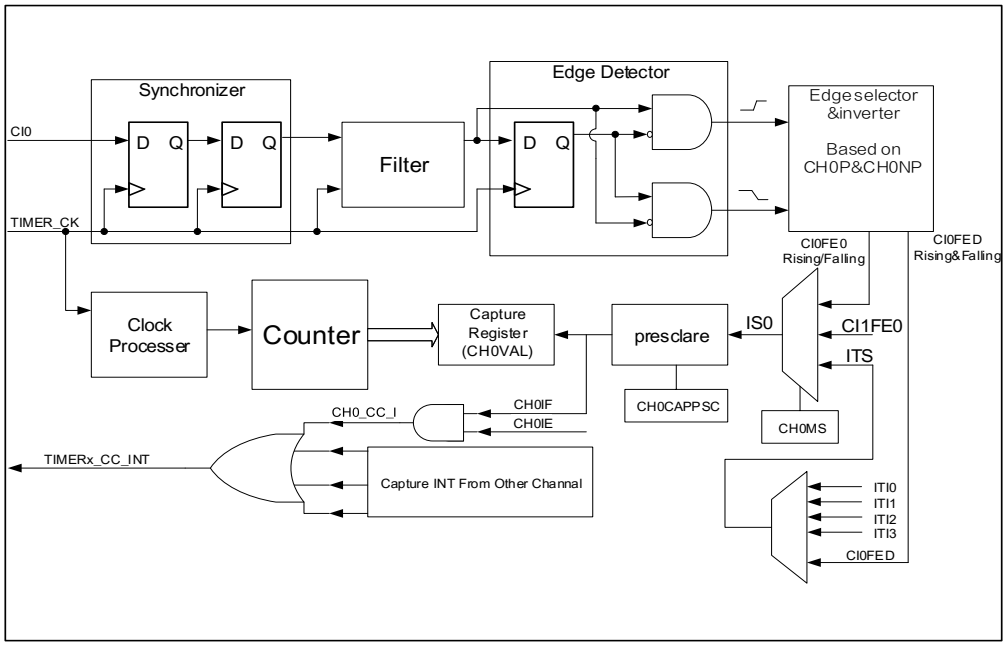
Input capture and output compare channels

The general level3 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 it is enabled when CHxIE=1.

Figure 9-58. Channel input capture principle



Channels' input signals (CIx) is the TIMERx_CHx signal. 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, CHxVAL 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**

Figure 9-59. channel output compare principle (with complementary output, x=0)

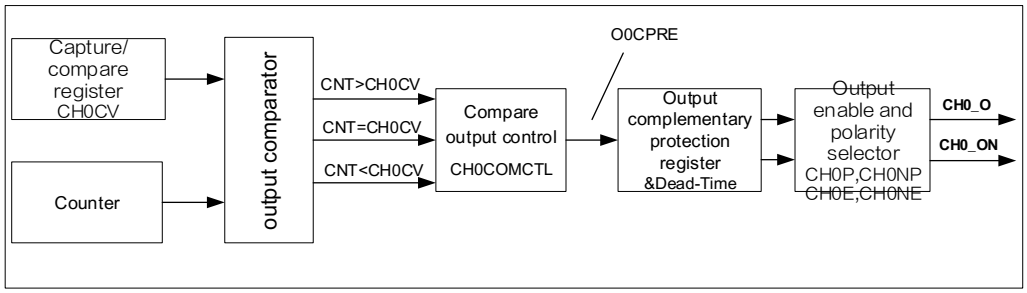


Figure 9-60. Channel output compare principle (CH1_O)

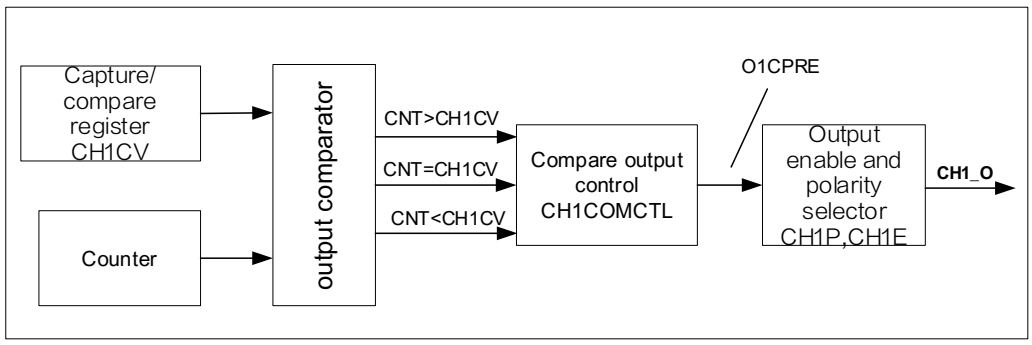


Figure 9-59. channel output compare principle (with complementary output, x=0) and *Figure 9-60. Channel output compare principle (CH1_O)* show the principle circuit of channels output compare function. The relationship between the channel output signal CHx_O/CHx_ON and the OxCPRE signal (more details refer to *Channel output prepare signal*) is described as below: The active level of OxCPRE is high, the output level of CHx_O/CHx_ON depends on OxCPRE signal, CHxP/CHxNP bit and CHxE/CHxNE bit (please refer to the TIMERx_CHCTL2 register for more details). For examples:

1) configure CHxP=0 (the active level of CHx_O is high, the same as OxCPRE), CHxE=1

(the output of CHx_O is enabled):

If the output of OxCPRE is active(high) level, the output of CHx_O is active(high) level;

If the output of OxCPRE is inactive(low) level, the output of CHx_O is active(low) level.

2) Configure CHxNP=0 (the active level of CHx_ON is low, contrary to OxCPRE), CHxNE=1 (the output of CHx_ON is enabled):

If the output of OxCPRE is active(high) level, the output of CHx_O is active(low) level;

If the output of OxCPRE is inactive(low) level, the output of CHx_O is active(high) level.

When CH0_O and CH0_ON are output at the same time, the specific outputs of CH0_O and CH0_ON are related to the relevant bits (ROS, IOS, POE and DTCFG bits) in the TIMERx_CCHP register. Please refer to ***Channel output complementary PWM*** for more details.

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 CHxDEN = 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
- Select the active high polarity by CHxP/CHxNP
- Enable the output by CHxEN

Step3: Interrupt/DMA-request enables configuration by CHxIE/ CHxDEN

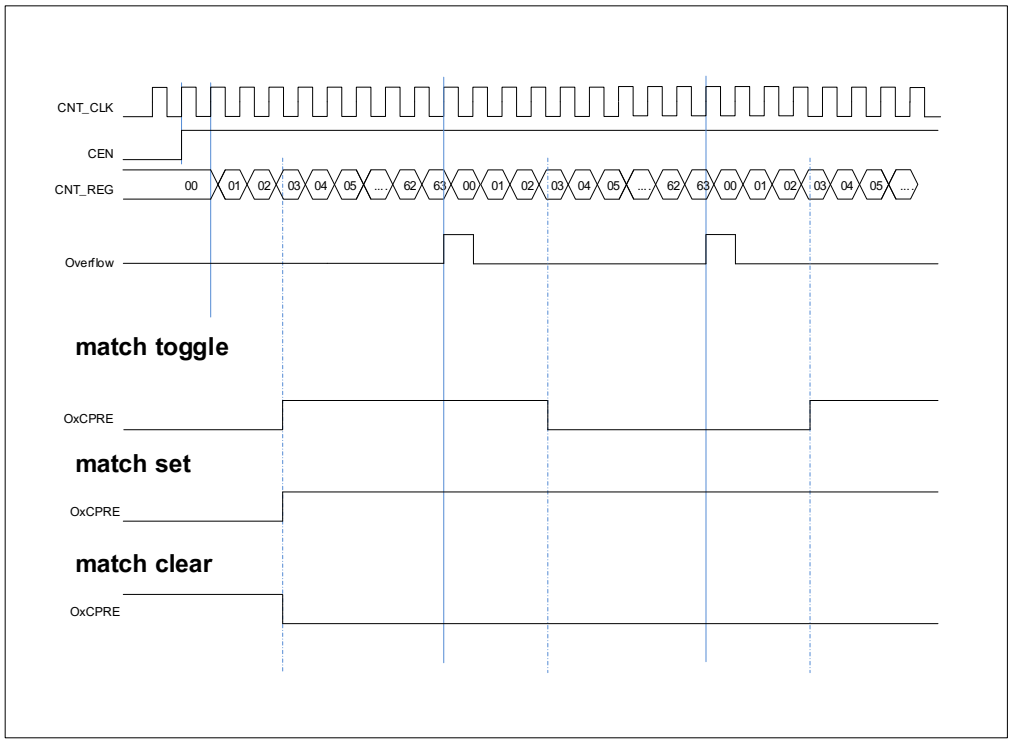
Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV

About the CHxVAL; you can change it ongoing to meet the waveform you expected.

Step5: Start the counter by CEN.

Figure 9-61. Output-compare in three modes show the three compare modes toggle/set/clear. CAR = 0x63, CHxVAL = 0x3

Figure 9-61. Output-compare in three modes



Output PWM function

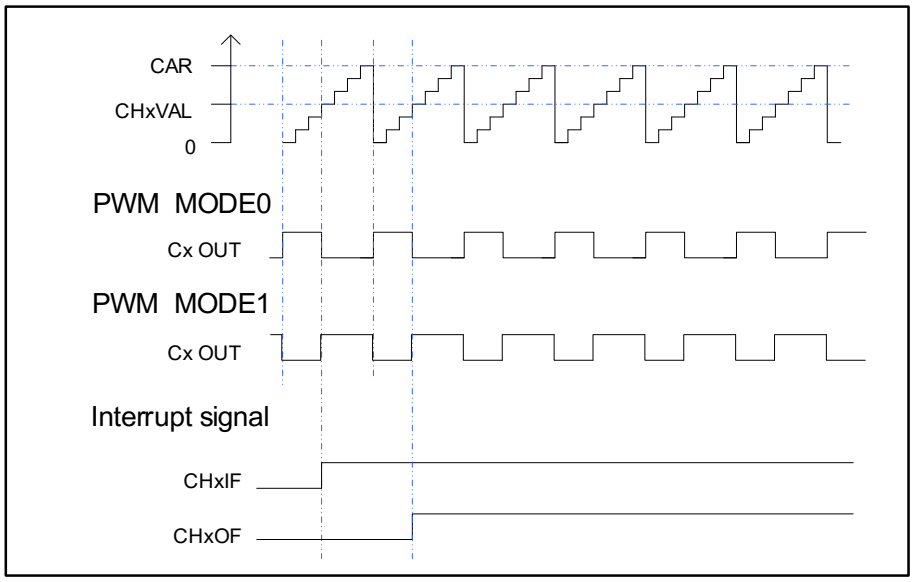
In the output PWM function (by setting the CHxCOMCTL bits to 3'b110 (PWM mode0) or to 3'b 111(PWM mode1), the channel can generate PWM waveform according to the TIMERx_CAR register and TIMERx_CHxCV register.

The period is determined by TIMERx_CAR and duty cycle is determined by TIMERx_CHxCV. *Figure 9-62. PWM mode timechart* shows the PWM output mode 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 9-62. PWM mode 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.

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 2 channels, but only the first channel 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 9-6. Complementary outputs controlled by parameters

Complementary Parameters					Output Status	
POEN	ROS	IOS	CHxEN	CHxNEN	CHx_O	CHx_ON
0	0/1	0	0	0	CHx_O = ISOx CHx_ON = ISOxN CHx_O / CHx_ON output disable(1).	
				1	CHx_O = CHxP CHx_ON = CHxNP CHx_O/CHx_ON output enable. If clock is enable: CHx_O = ISOx CHx_ON = ISOxN	
			1	0		
				1		
		1	0	0	CHx_O = ISOx CHx_ON = ISOxN CHx_O / CHx_ON output disable(1).	
				1	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.	
			1	0		
				1		
1	0	0/1	0	0	CHx_O/CHx_ON = LOW CHx_O/CHx_ON output disable.	
				1	CHx_O = LOW CHx_O output disable.	CHx_ON =OxCPRE⊕(4)CHxNP CHx_ON output enable.
			1	0	CHx_O=OxCPRE⊕CHxP CHx_O output enable.	CHx_ON = LOW CHx_ON output disable.
				1	CHx_O=OxCPRE⊕CHxP CHx_O output enable.	CHx_ON =(!OxCPRE)(5)⊕CHxNP. CHx_ON output enable.
	1		0	0	CHx_O = CHxP CHx_O output "off-state".	CHx_ON = CHxNP CHx_ON output "off-state".
				1	CHx_O = CHxP CHx_O output "off-state"	CHx_ON =OxCPRE⊕CHxNP CHx_ON output enable
			1	0	CHx_O=OxCPRE⊕CHxP CHx_O output enable	CHx_ON = CHxNP CHx_ON output "off-state".

				1	CHx_O=OxCPRE⊕CHxP CHx_O output enable	CHx_ON =(!OxCPRE)⊕CHxNP CHx_ON output enable.

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 channel 0. 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.

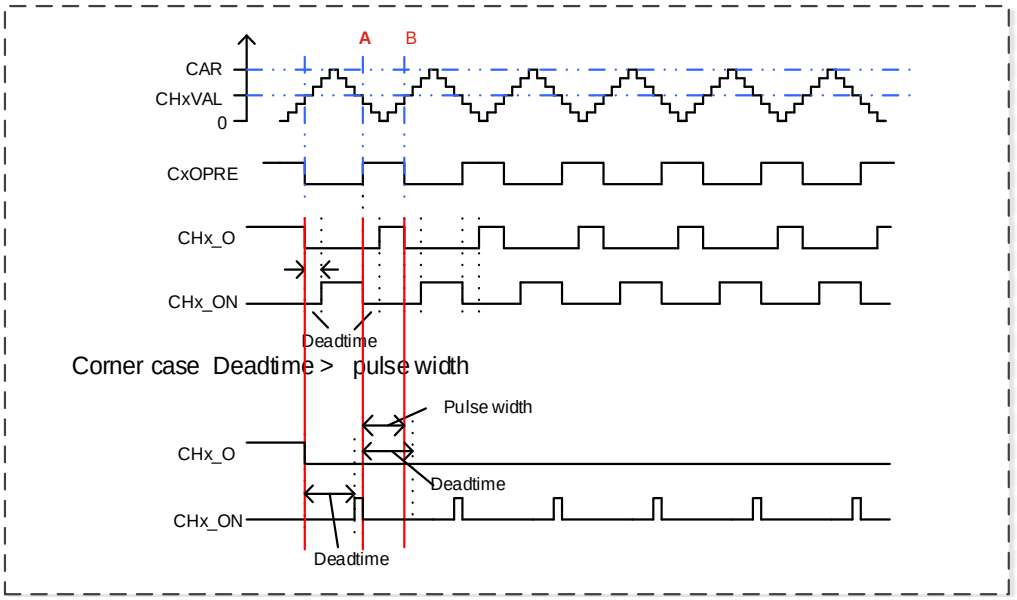
When the channel (x) match (TIMERx counter = CHxVAL) occurs, OxCPRE will be toggled because under PWM0 mode. At point A in the ***Figure 9-63. Complementary output with dead-time insertion.*** CHx_O signal remains at the low value until the end of the deadtime delay, while CHx_ON will be cleared at once. 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 9-63. 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 9-63. Complementary output with dead-time insertion.



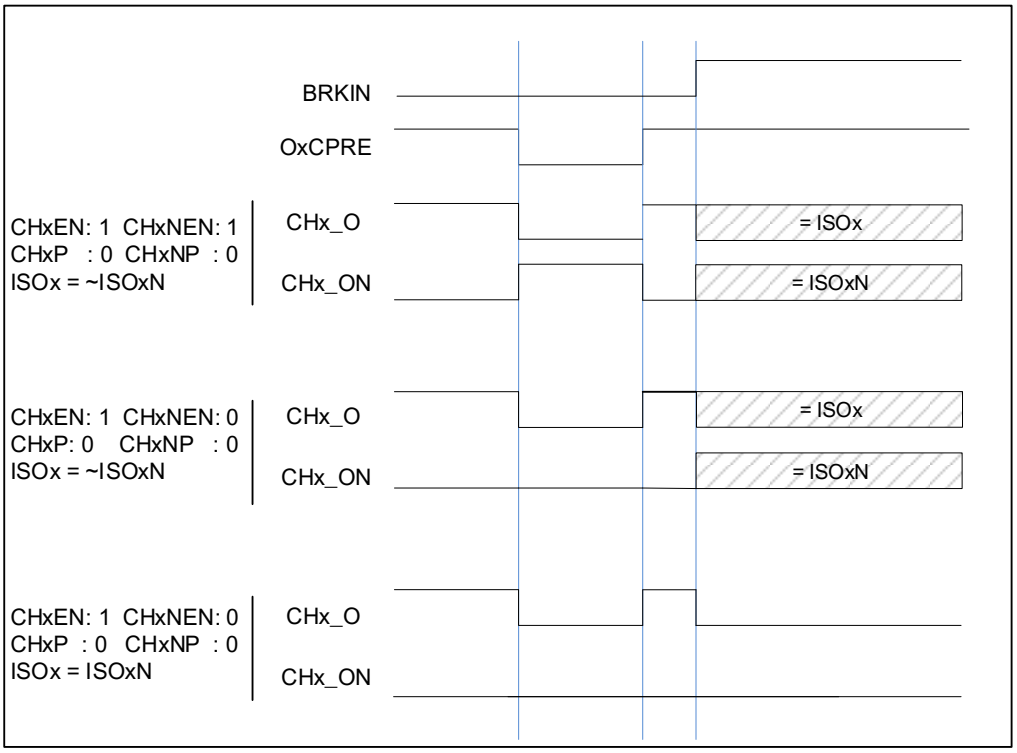
Break mode

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 9-64. Output behavior in response to a break(The break high active)

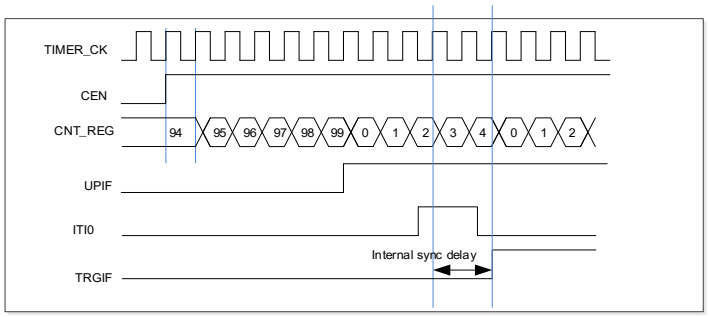
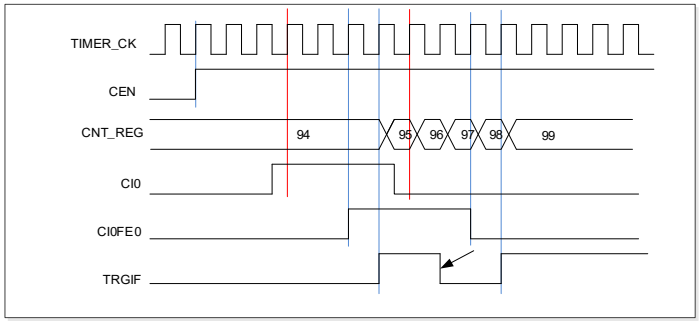
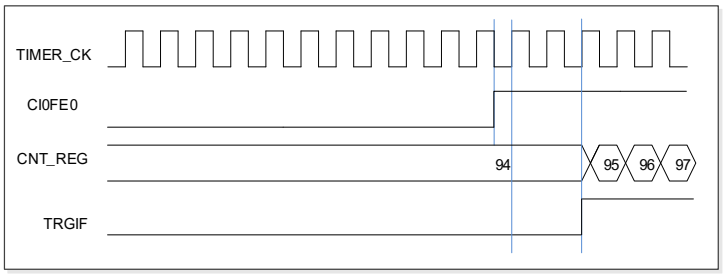


Master-slave management

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 9-7. Slave mode example table

	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	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
	rising trigger input.			
	Figure 9-65. 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.
	Figure 9-66. 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.	TI0S=0(Non-xor) [CH0NP==0, CH0P==0] no inverted.	Filter is bypass in this example.
	Figure 9-67. Event mode  			

Single pulse mode

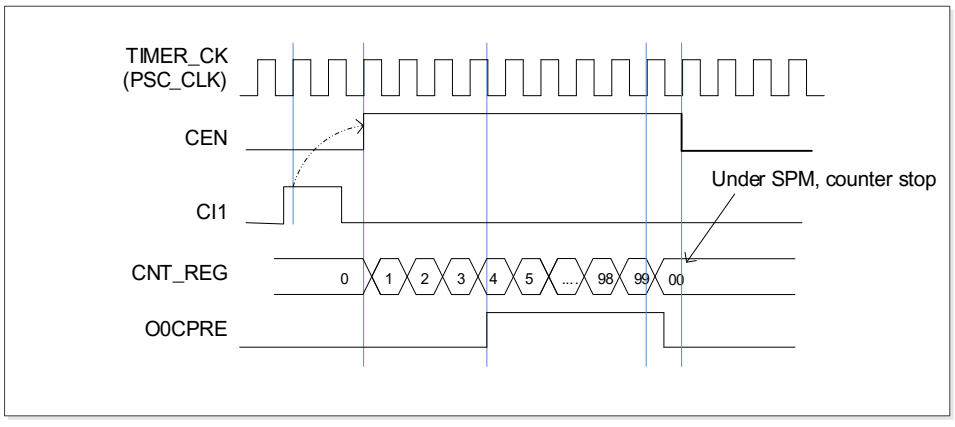
Single pulse mode is opposite to the repetitive mode, which can be enabled by setting SPM in TIMERx_CTL0. When you set SPM, the counter will be clear and stop when the next update event. In order to get pulse waveform, you can set the TIMERx to PWM mode or compare by CHxCOMCTL.

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 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 9-68. Single pulse mode TIMERx_CHxCV = 4 TIMERx_CAR=99 shows an example.

Figure 9-68. Single pulse mode TIMERx_CHxCV = 4 TIMERx_CAR=99



Timers interconnection

Refer to *Timers interconnection*

Table 9-8. TIMERx(x=14) interconnection

Slave TIMER	ITI0(TRGS = 000)	ITI1(TRGS = 001)	ITI2(TRGS = 010)	ITI3(TRGS = 011)
TIMER14	Reserved	TIMER2	Reserved	Reserved

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®-M23 halted, and the TIMERx_HOLD configuration bit in DBG_CTL1 register set to 1, the TIMERx counter stops.

9.4.5. TIMERx register definition (x=14)

TIMER14 base address: 0x4001 4000

Control register 0 (TIMERx_CTL0)

Address offset: 0x00

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						CKDIV[1:0]		ARSE	Reserved			SPM	UPS	UPDIS	CEN
						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
6:4	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, pause mode and quadrature decoder mode.

Control register 1 (TIMERx_CTL1)

Address offset: 0x04

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					ISO1	ISO0N	ISO0	Reserved	MMC[2:0]			DMAS	CCUC	Reserved	CCSE
					rw	rw	rw		rw			rw	rw		rw

Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value
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	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 in master mode to slave timers 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: Reserved 111: Reserved
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 enable (for CHxEN, CHxNEN and CHxCOMCTL bits) are set (CCSE=1), these shadow registers update are controlled as below: 0: The shadow registers update by when CMTG bit is set. 1: The shadow registers update by 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 based when commutation event coming. 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
Reserved								MSM	TRGS[2:0]			Reserved	SMC[2:0]		
								rw	rw				rw		

Bits	Fields	Descriptions
31:8	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: CI0 edge flag (CI0F_ED) 101: Channel 0 input filtered output (CI0FE0) 110: Channel 1 input filtered output (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 Mode 0. The counter counts on the rising edges of the selected

trigger.

Because CI0F_ED outputs 1 pulse for each transition on CI0F, and the pause mode checks the level of the trigger signal, when CI0F_ED is selected as the trigger input, the pause mode must not be used.

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	Reserved		CH1DEN	CH0DEN	UPDEN	BRKIE	TRGIE	CMTIE	Reserved		CH1IE	CH0IE	UPIE
	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
12:11	Reserved	Must be kept at reset value.
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
4:3	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	BRKIF	TRGIF	CMTIF	Reserved.		CH1IF	CH0IF	UPIF
					rc_w0	rc_w0		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

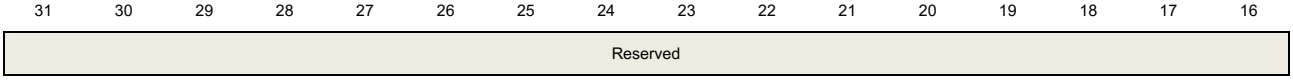
8	Reserved	Must be kept at reset value.
7	BRKIF	Break interrupt flag When the break input is inactive, the bit is set by hardware. When the break input is inactive, the bit can be cleared by software. 0: No active level break has been detected. 1: An active level has been detected.
6	TRGIF	Trigger interrupt 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 channel's commutation event occurs, and cleared by software 0: No channel commutation interrupt occurred 1: Channel commutation interrupt occurred
4:3	Reserved	Must be kept at reset value
2	CH1IF	Channel 1 's capture/compare interrupt flag Refer to CH0IF description
1	CH0IF	Channel 0 's capture/compare interrupt flag This flag is set by hardware and cleared by software. When channel 0 is in input mode, this flag is set when a capture event occurs. When channel 0 is in output mode, this flag is set when a compare event occurs. If Channel0 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 on an update event 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)



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BRKG	TRGG	CMTG	Reserved		CH1G	CH0G	UPG
								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 and cleared by hardware automatically. When this bit is set, the POEN bit is cleared and BRKIF flag is 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 is 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's c/c control update event
4:3	Reserved	Must be kept at reset value
2	CH1G	Channel 1's capture or compare event generation Refer to CH0G description
1	CH0G	Channel 0's capture or compare event generation This bit is set by software in order to generate a capture or compare event in channel 0, it is automatically cleared by hardware. When this bit is set, the CH0IF flag is set, the corresponding interrupt or DMA request is sent if enabled. In addition, if channel 1 is configured in input mode, the current value of the counter is captured in TIMERx_CH0CV register, and the CH0OF flag is set if the CH0IF flag was already high. 0: No generate a channel 1 capture or compare event 1: Generate a channel 1 capture or compare event
0	UPG	Update event generation This bit can be set by software, and cleared by hardware automatically. When this bit is set, the counter is cleared if the center-aligned or up counting mode is selected,

else (down counting) 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
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. 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 0: Channel 0 output quickly compare disable. Only the valid edge of the comparison result will generate a comparison match, and CH0_O is set to the compare level. The minimum delay from an edge on the result of the comparison to activate CH0_O output is 5 clock cycles. 1: Channel 0 output quickly compare enable. Both the valid edge of the trigger input signal and the comparison result will generate a comparison match, and CH0_O is set to the compare level.The minimum delay from an edge on the trigger input to activate CH0_O output is 3 clock cycles.
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
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 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								CH1NP	Reserved	CH1P	CH1EN	CH0NP	CH0NEN	CH0P	CH0EN
								rw		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 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.

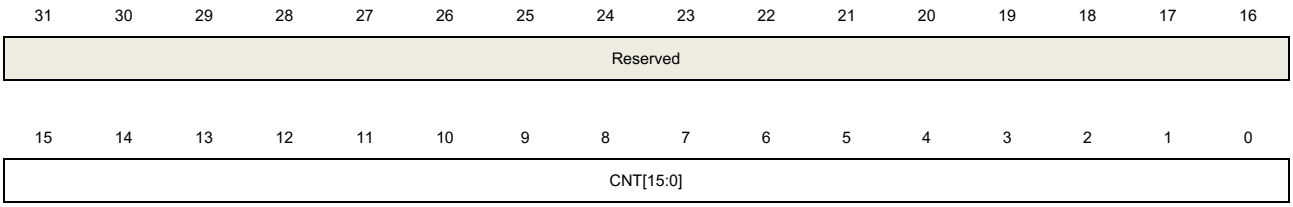
		This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 or 10.
2	CH0NEN	Channel 0 complementary output enable When channel 0 is configured in output mode, setting this bit enables the complementary output in channel0. 0: Channel 0 complementary output disabled 1: Channel 0 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. [CH0NP, CH0P] will select the active trigger or capture polarity for CI0FE0 or CI1FE0. [CH0NP==0, CH0P==0]: CIxFE0's rising edge is the active signal for capture or trigger operation in slave mode. And CIxFE0 will not be inverted. [CH0NP==0, CH0P==1]: CIxFE0's falling edge 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]: CIxFE0's falling and rising edge are both the active signal for capture or trigger operation in slave mode. And CIxFE0 will be not inverted. This bit cannot be modified when PROT [1:0] bit-filed 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 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)



rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	CNT[15:0]	This bit-filed indicates the current counter value. Writing to this bit-filed 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-filed specifies the auto reload value of the counter.

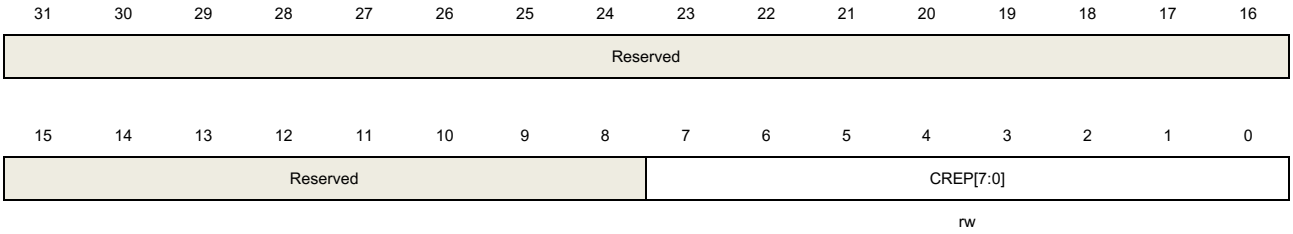
Note: When the timer is configured in input capture mode, this register must be configured a non-zero value (such as 0xFFFF) which is larger than user expected value.

Counter repetition register (TIMERx_CREP)

Address offset: 0x30

Reset value: 0x0000 0000

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



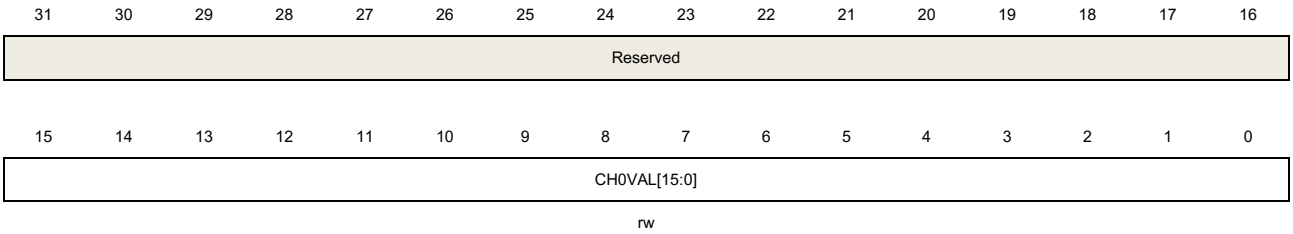
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	CREP[7:0]	Counter repetition value This bit-filed specifies the update event generation rate. Each time the repetition counter counting down to zero, an update event is generated. The update rate of the shadow registers is also affected by this bit-filed 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 or compare value of channel0 When channel 0 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only.

When channel 0 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates 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 or compare value of channel1 When channel 1 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 1 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Complementary channel protection register (TIMERx_CCHP)

Address offset: 0x44

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
POEN	OAEN	BRKP	BRKEN	ROS	IOS	PROT[1:0]		DTCFG[7:0]							
rw	rw	rw	rw	rw	rw	rw		rw							

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
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 CKM 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 9-6. 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 9-6. 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 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:

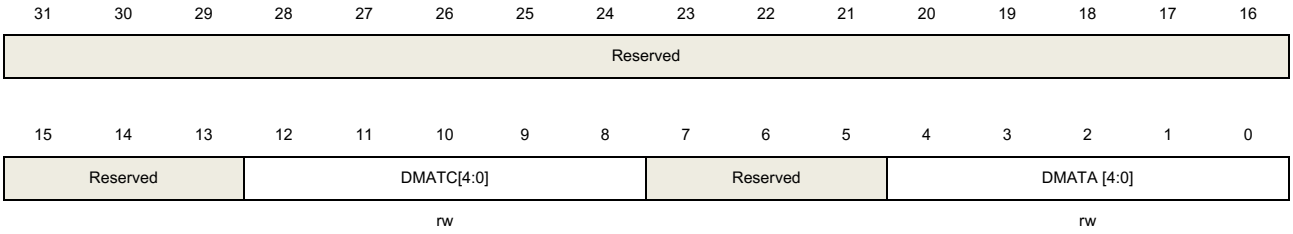
1. t_{DTS_CK} is the period of DTS_CK which is configured by CKDIV[1:0] in TIMERx_CTL0.
2. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.

DMA configuration register (TIMERx_DMACFG)

Address offset: 0x48

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:8	DMATC [4:0]	DMA transfer count This filed is defined the number of DMA will access(R/W) the register of TIMERx_DMATB
7:5	Reserved	Must be kept at reset value.
4:0	DMATA [4:0]	DMA transfer access start address This filed 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. 5'b0_0000: TIMERx_CTL0 5'b0_0001: TIMERx_CTL1 … In a word: Start Address = TIMERx_CTL0 + DMATA*4

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.

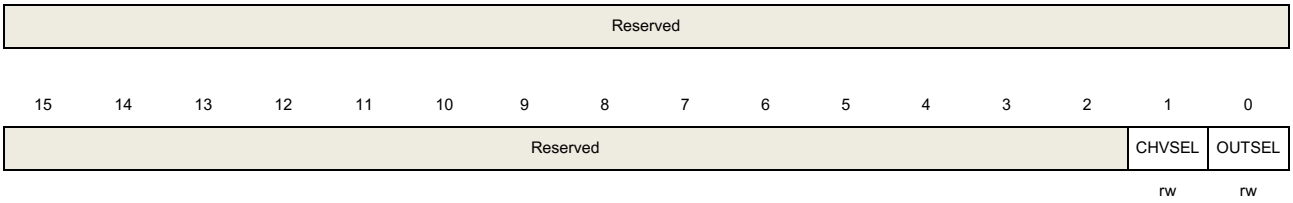
Configuration register (TIMERx_CFG)

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



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. 0: No effect. 1: If write the CHxVAL register, the write value is same as the CHxVAL value, the write access ignored.
0	OUTSEL	The output value selection This bit-field set and reset by software 0: No effect. 1: If POEN and IOS are 0, the output disabled.

9.5. General level4 timer (TIMERx, x=15, 16)

9.5.1. Overview

The general level4 timer module (TIMER15, TIMER16) is a one-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 general level4 timer has a 16-bit counter that can be used as an unsigned counter.

In addition, the general level4 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.

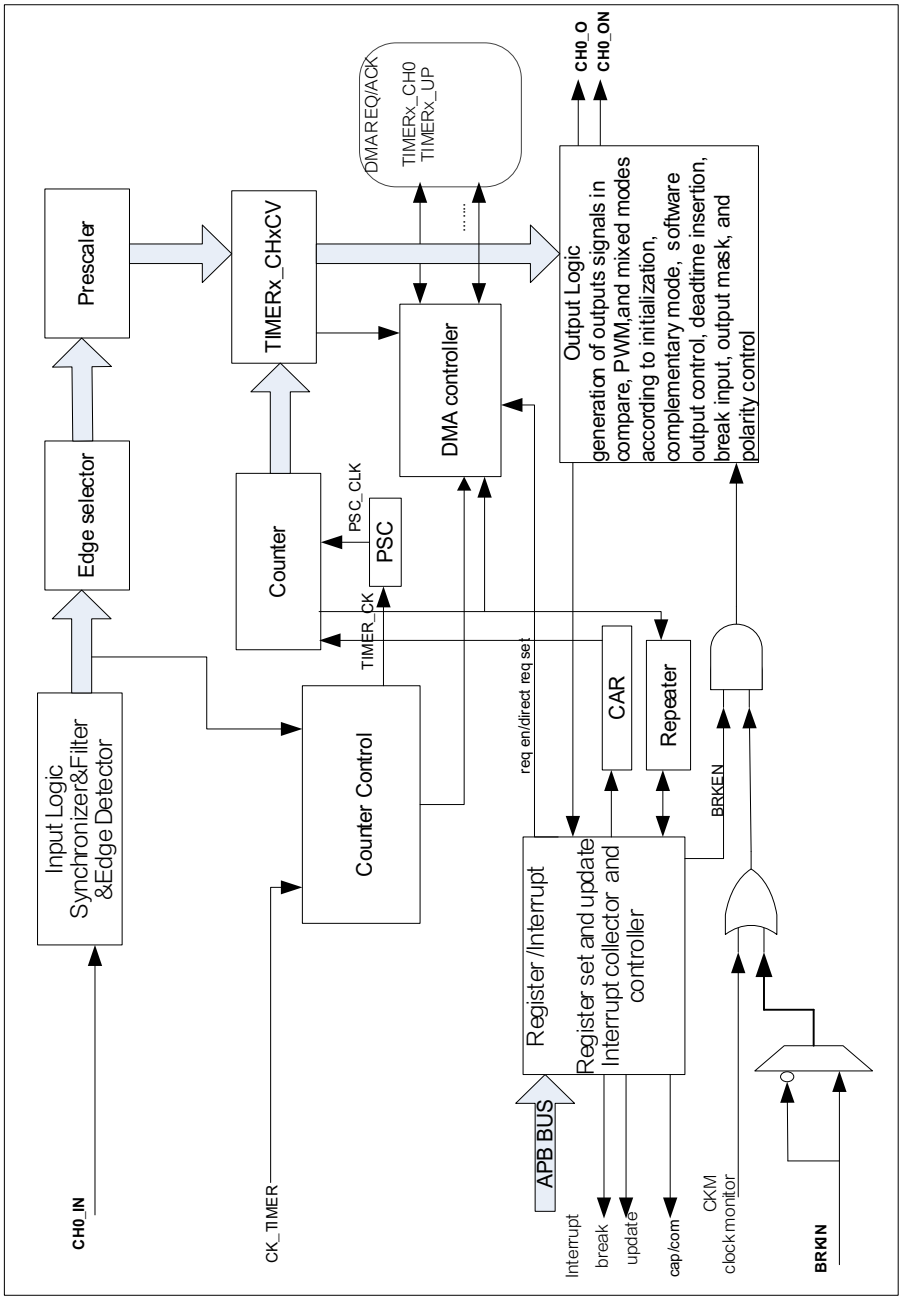
9.5.2. Characteristics

- Total channel num: 1.
- Counter width: 16 bits.
- Clock source of counter clock: internal clock.
- Counter modes: count up 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, single pulse mode
- Programmable dead time insertion.
- Auto reload function.
- Programmable counter repetition function.
- Break input.
- Interrupt output or DMA request on: update event, compare/capture event, commutation event and break input.

9.5.3. Block diagram

Figure 9-69. General level4 timer block diagram provides details of the internal configuration of the general level4 timer.

Figure 9-69. General level4 timer block diagram



9.5.4. Function overview

Clock source configuration

The general level4 TIMER can only being clocked by the CK_TIMER.

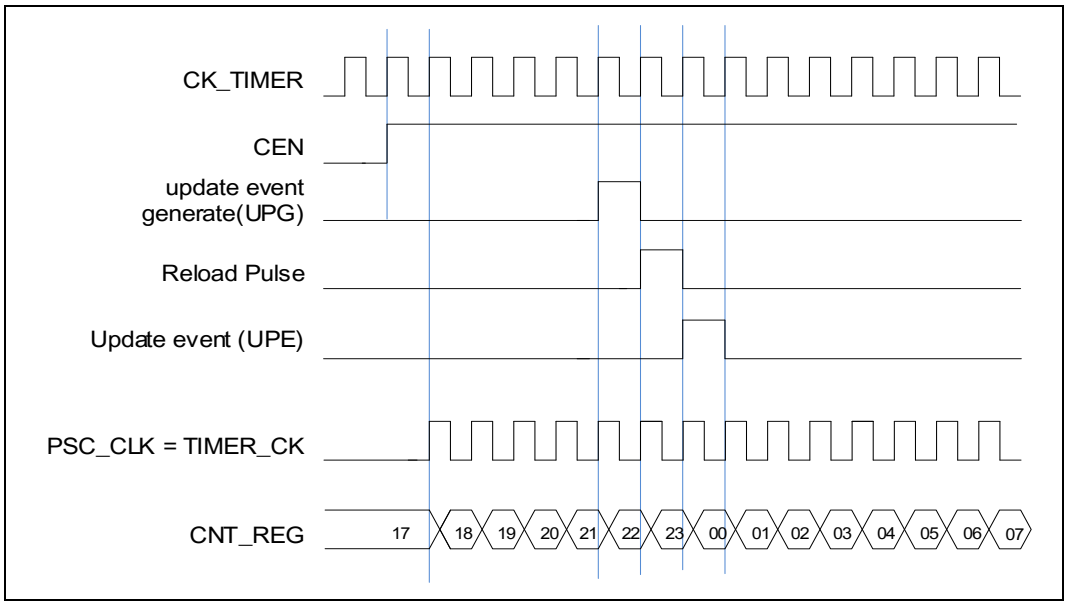
- Internal timer clock CK_TIMER which is from module RCU

The general level4 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

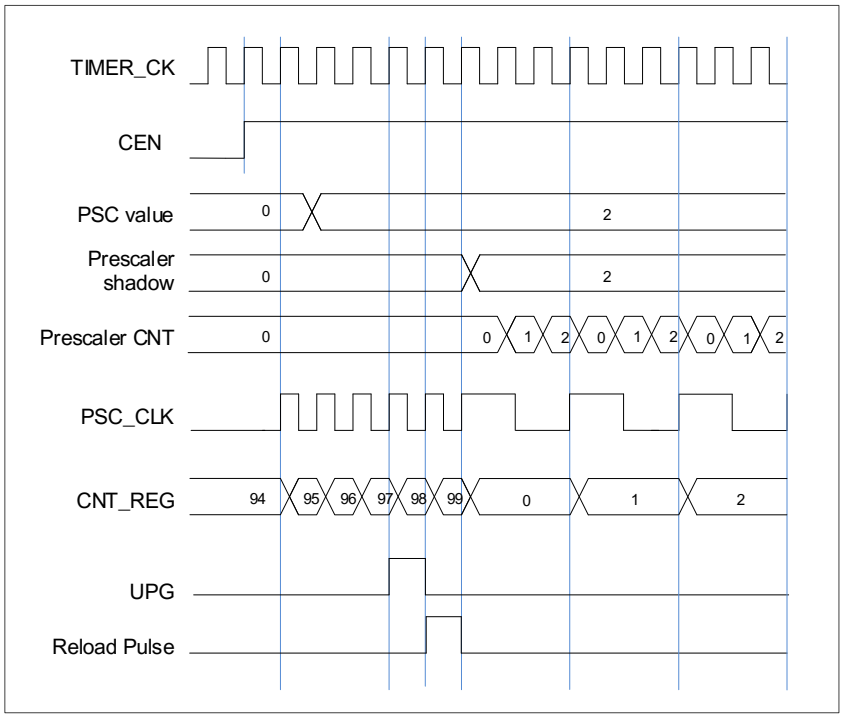
Figure 9-70. Timing chart of internal clock divided by 1



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 9-71. 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 registers (repetition counter, counter auto reload register, prescaler register) are updated.

Figure 9-72. 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 9-72. Timing chart of up counting mode, PSC=0/2

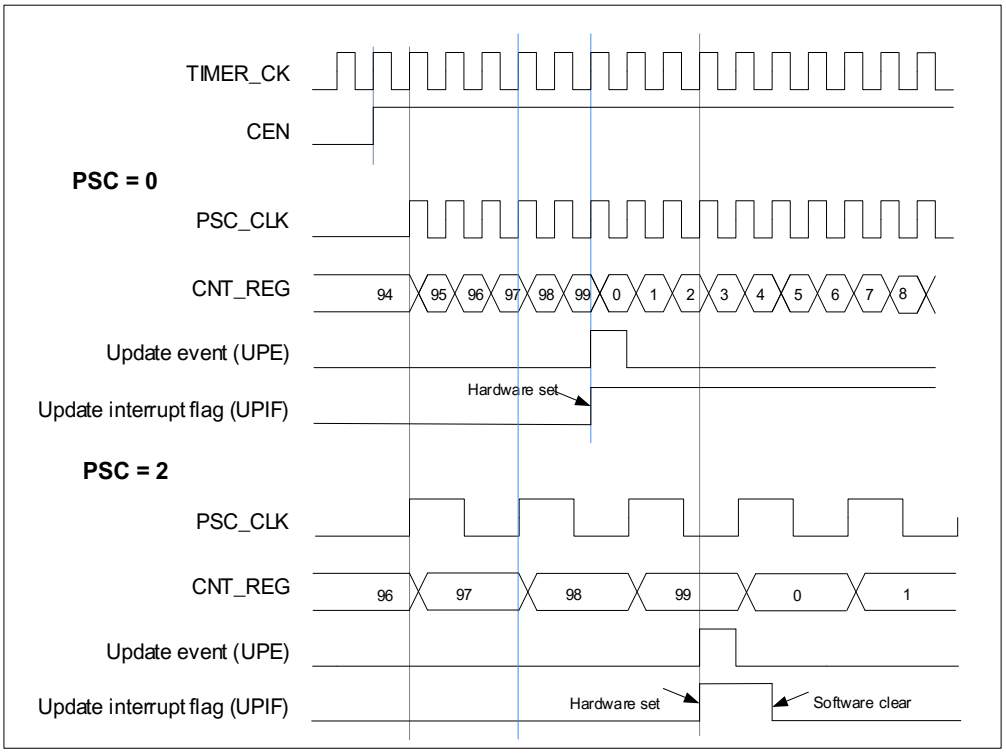
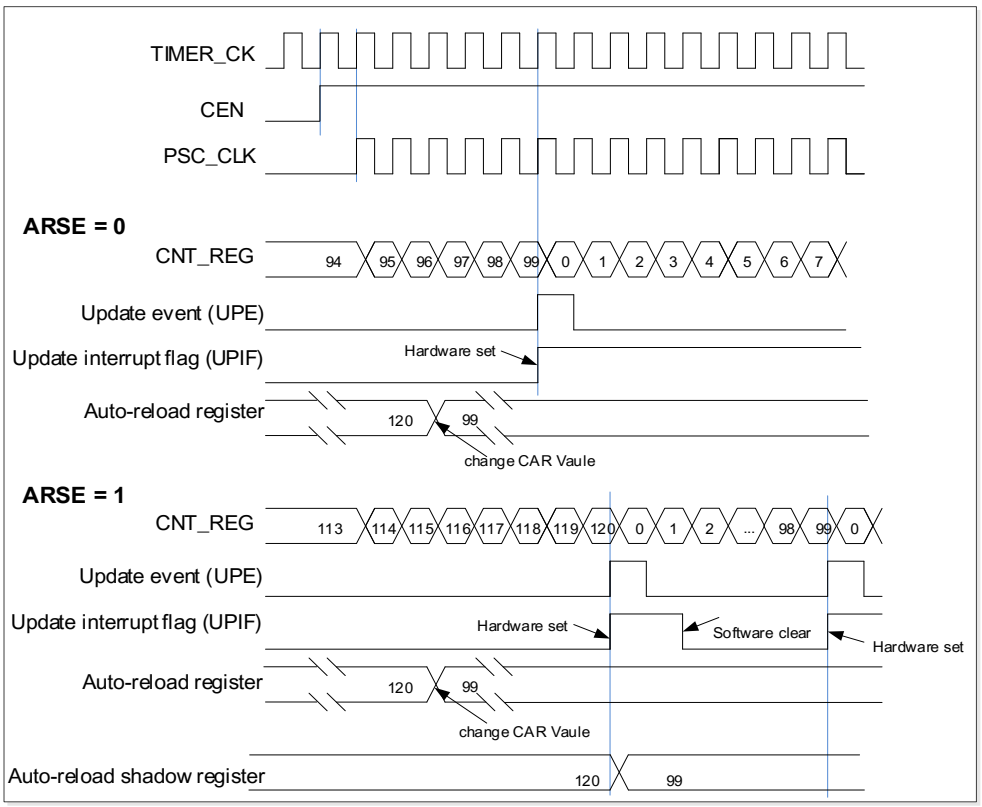


Figure 9-73. Timing chart of up counting mode, change TIMERx_CAR on the go

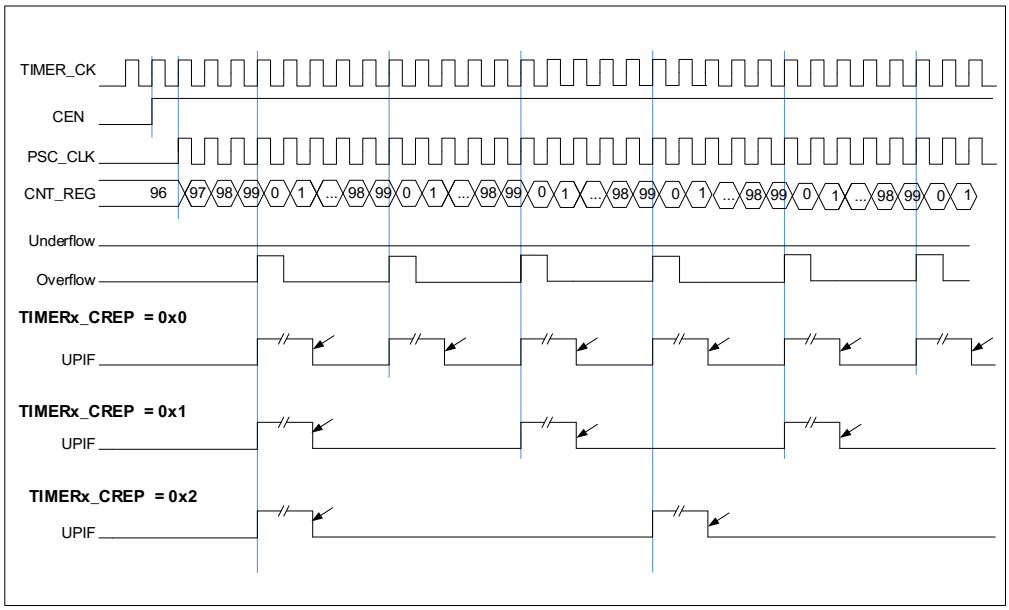


Update event (from overflow/underflow) rate configuration

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 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.

Figure 9-74. Repetition counter timing chart of up counting mode



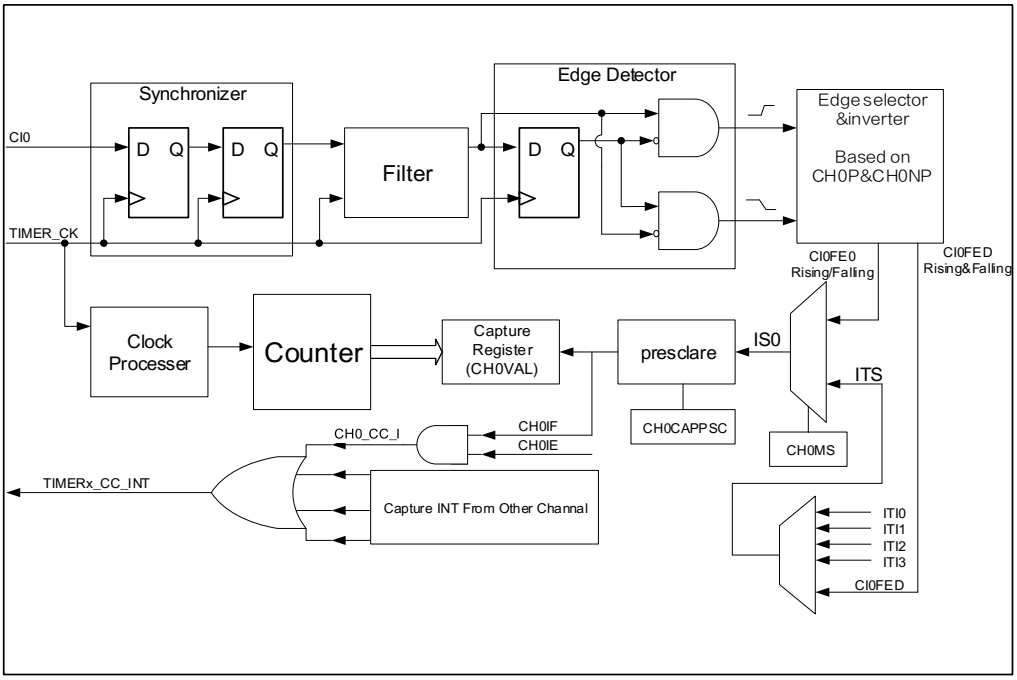
Input capture and output compare channels

The general level4 timer has one 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 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 9-75. Channel input capture principle



Channels' input signals (CIx) is the TIMERx_CHx signal. 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, CHxVAL 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 period measurement from signals on the TIMERx_CHx pins. For example, PWM signal connects to CI0 input. Select CI0 as channel 0 capture signals by setting CH0MS to 2'b01 in the channel control register (TIMERx_CHCTL0) and set capture on rising edge. The counter is set to restart mode and is restarted on channel 0 rising edge. Then the TIMERX_CH0CV can measure the PWM period.

- **Channel output compare function**

Figure 9-76. Channel output compare principle (with complementary output, x=0)

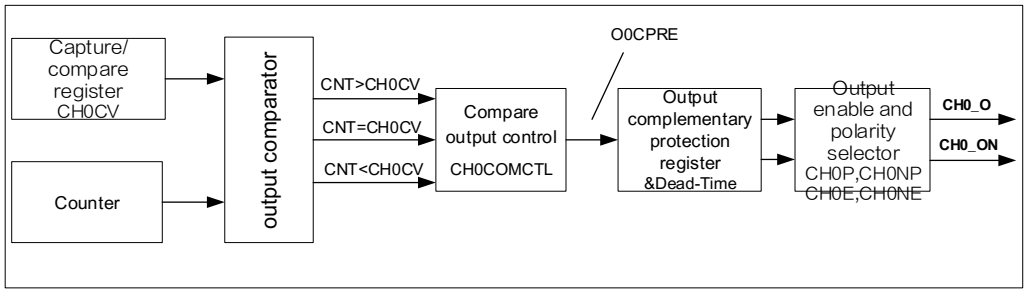


Figure 9-76. Channel output compare principle (with complementary output, x=0) show the logic circuit of output compare mode. The relationship between the channel output signal CHx_O/CHx_ON and the OxCPRE signal (more details refer to *Channel output prepare signal*) is described as below: The active level of OxCPRE is high, the output level of CHx_O/CHx_ON depends on OxCPRE signal, CHxP/CHxNP bit and CHxE/CHxNE bit (please refer to the TIMERx_CHCTL2 register for more details). For examples,

1) Configure CHxP=0 (the active level of CHx_O is high, the same as OxCPRE), CHxE=1 (the output of CHx_O is enabled):

If the output of OxCPRE is active(high) level, the output of CHx_O is active(high) level;

If the output of OxCPRE is inactive(low) level, the output of CHx_O is active(low) level.

2) Configure CHxNP=0 (the active level of CHx_ON is low, contrary to OxCPRE), CHxNE=1 (the output of CHx_ON is enabled):

If the output of OxCPRE is active(high) level, the output of CHx_O is active(low) level;

If the output of OxCPRE is inactive(low) level, the output of CHx_O is active(high) level.

When CH0_O and CH0_ON are output at the same time, the specific outputs of CH0_O and CH0_ON are related to the relevant bits (ROS, IOS, POE and DTCFG bits) in the TIMERx_CCHP register. Please refer to *Channel output complementary PWM* for more

details.

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 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 CHxDEN =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
- Enable the output by CHxEN

Step3: Interrupt/DMA-request enables configuration by CHxIE/CHxDEN

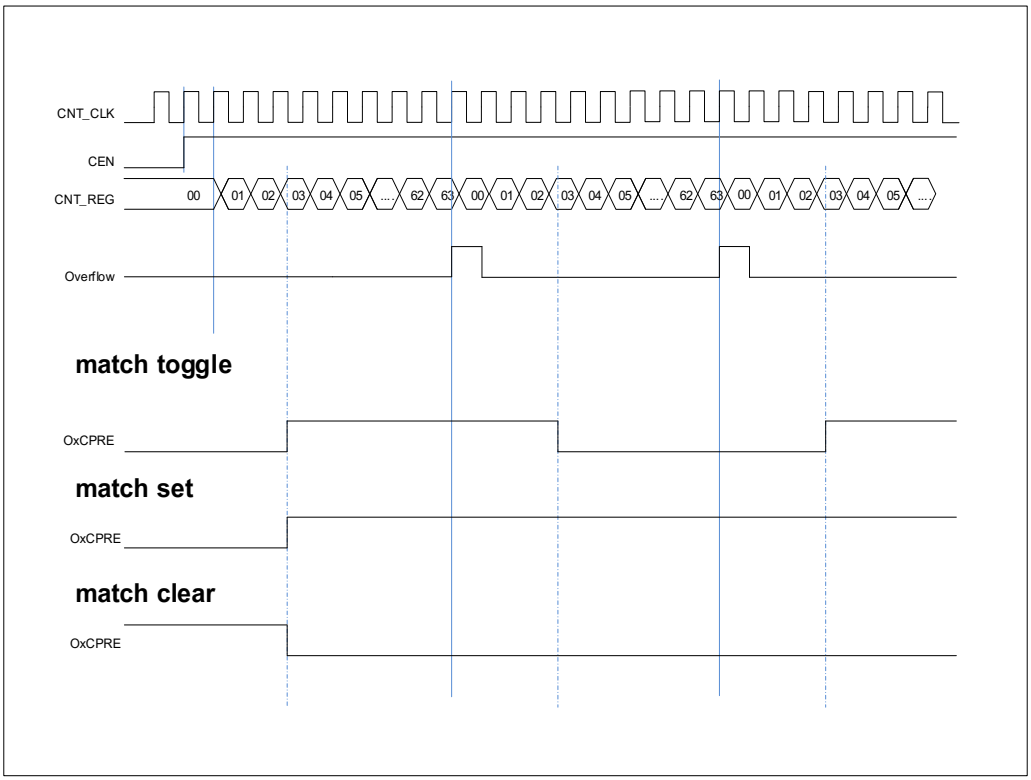
Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV

About the CHxVAL; you can change it ongoing 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 9-77. Output-compare under three modes



Output PWM function

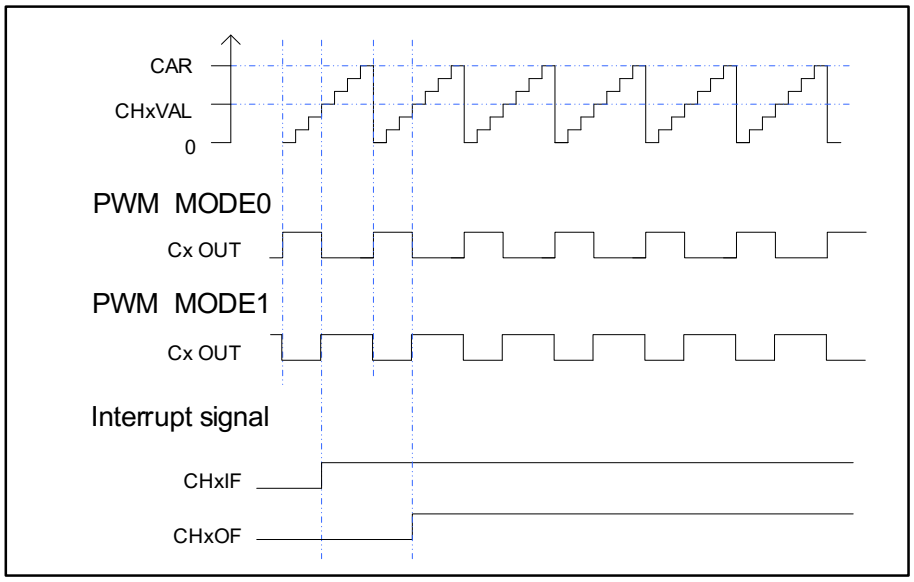
In the output PWM function (by setting the CHxCOMCTL bits to 3'b110 (PWM mode0) or to 3'b 111(PWM mode1), the channel can generate PWM waveform according to the TIMERx_CAR register and TIMERx_CHxCV register.

The period is determined by TIMERx_CAR and duty cycle is determined by TIMERx_CHxCV. ***Figure 9-78. PWM mode timechart*** shows the PWM output mode 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 9-78. PWM mode 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.

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 only 1 channel 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 9-9. Complementary outputs controlled by parameters

Complementary Parameters					Output Status	
POEN	ROS	IOS	CHxEN	CHxNEN	CHx_O	CHx_ON
0	0/1	0	0	0	CHx_O = ISOx CHx_ON = ISOxN CHx_O / CHx_ON output disable(1).	
				1	CHx_O = CHxP CHx_ON = CHxNP CHx_O/CHx_ON output enable. If clock is enable: CHx_O = ISOx CHx_ON = ISOxN	
			1	0		
				1		
		1	0	0	CHx_O = ISOx CHx_ON = ISOxN CHx_O / CHx_ON output disable(1).	
				1	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.	
			1	0		
				1		
1	0	0/1	0	0	CHx_O/CHx_ON = LOW CHx_O/CHx_ON output disable.	
				1	CHx_O = LOW CHx_O output disable.	CHx_ON =OxCPRE⊕(4)CHxNP CHx_ON output enable.
			1	0	CHx_O=OxCPRE⊕CHxP CHx_O output enable.	CHx_ON = LOW CHx_ON output disable.
				1	CHx_O=OxCPRE⊕CHxP CHx_O output enable.	CHx_ON =(!OxCPRE)(5)⊕CHxNP. CHx_ON output enable.
	1		0	0	CHx_O = CHxP CHx_O output "off-state".	CHx_ON = CHxNP CHx_ON output "off-state".
				1	CHx_O = CHxP CHx_O output "off-state"	CHx_ON =OxCPRE⊕CHxNP CHx_ON output enable
			1	0	CHx_O=OxCPRE⊕CHxP CHx_O output enable	CHx_ON = CHxNP CHx_ON output "off-state".

				1	CHx_O=OxCPRE⊕CHxP CHx_O output enable	CHx_ON =(!OxCPRE)⊕CHxNP CHx_ON output enable.

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 channel 1. 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.

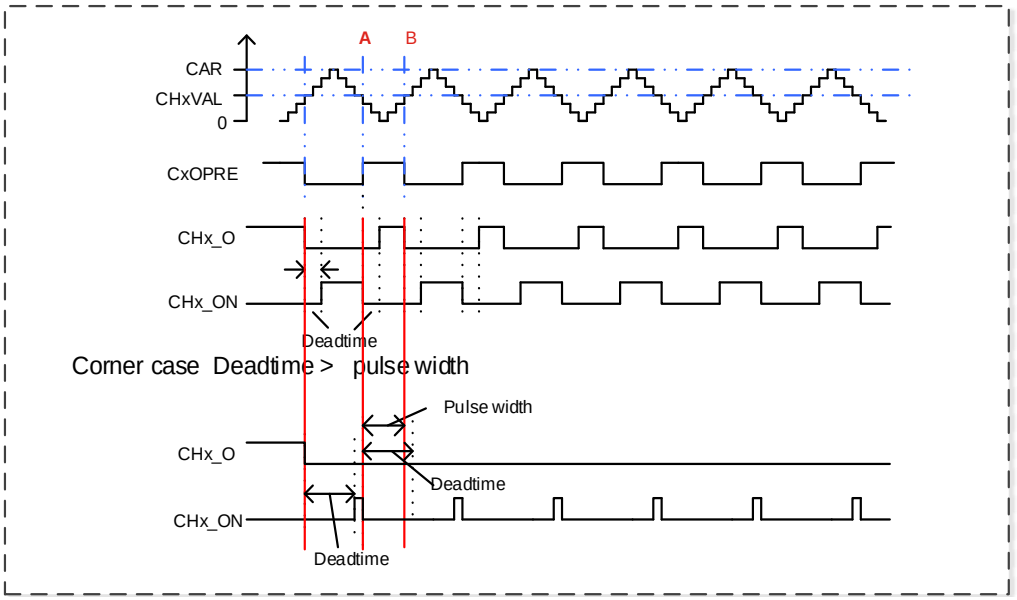
When the channel (x) match (TIMERx counter = CHxVAL) occurs, OxCPRE will be toggled because under PWM0 mode. At point A in the ***Figure 9-79. Complementary output with dead-time insertion.*** CHx_O signal remains at the low value until the end of the deadtime delay, while CHx_ON will be cleared at once. 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 9-79. 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 9-79. Complementary output with dead-time insertion.



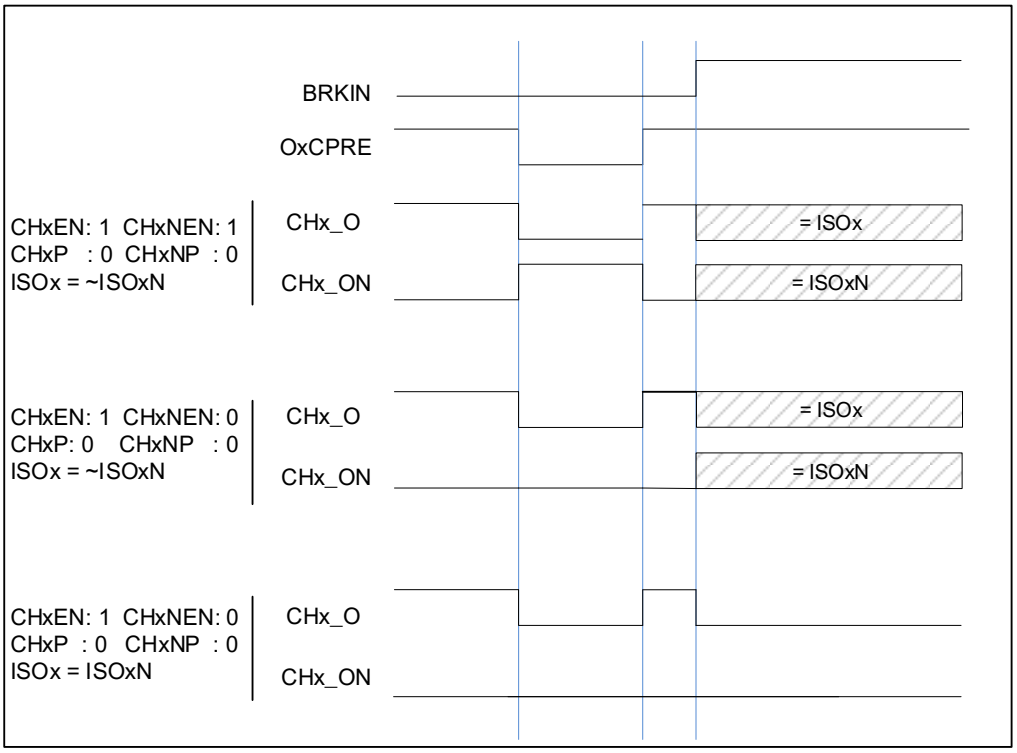
Break mode

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 9-80. Output behavior in response to a break(The break high active)



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.

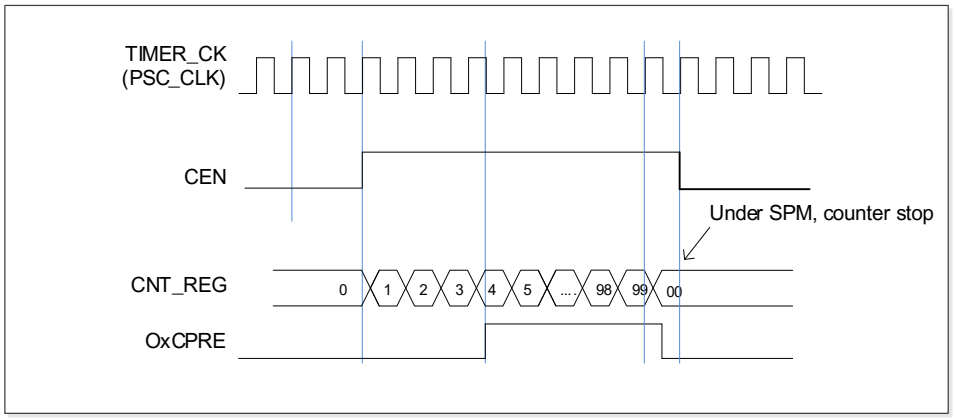
Once the timer is set to operate in the single pulse mode, it is necessary to set the timer enable bit CEN in the TIMERx_CTL0 register to 1 to enable the counter. Setting the CEN bit to 1 can generate a pulse and then keep the CEN bit at a high state until the update event occurs or the CEN bit is written 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 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 9-81. Single pulse mode TIMERx_CHxCV = 0x04 TIMERx_CAR=0x60 shows an

example.

Figure 9-81. Single pulse mode TIMERx_CHxCV = 0x04 TIMERx_CAR=0x60



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®-M23 halted, and the TIMERx_HOLD configuration bit in DBG_CTL1 register set to 1, the TIMERx counter stops.

9.5.5. TIMERx register definition (x=15, 16)

TIMER15 base address: 0x4001 4400

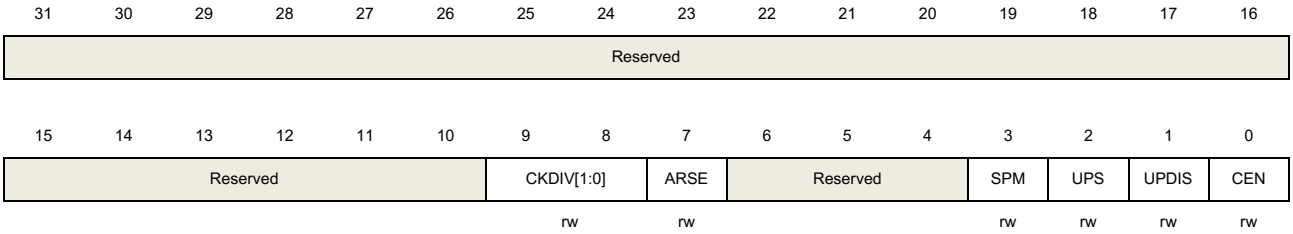
TIMER16 base address: 0x4001 4800

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, pause mode and quadrature decoder 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						ISO0N	ISO0	Reserved				DMAS	CCUC	Reserved	CCSE
						rw	rw					rw	rw		rw

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value
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.
7:4	Reserved	Must be kept at reset value
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 enable (for CHxEN, CHxNEN and CHxCOMCTL bits) are set (CCSE=1), these shadow registers update are controlled as below: 0: The shadow registers update by when CMTG bit is set. 1: The shadow registers update by 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 based when commutation event coming. When a channel does not have a complementary output, this bit has no effect.

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						CH0DEN	UPDEN	BRKIE	Reserved	CMTIE	Reserved			CH0IE	UPIE
						rw	rw	rw		rw				rw	rw

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value
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	Reserved	Must be kept at reset value
5	CMTIE	Commutation interrupt enable 0: disabled 1: enabled
4:2	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	BRKIF	Reserved	CMTIF	Reserved			CH0IF	UPIF
						rc_w0		rc_w0		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	Reserved	Must be kept at reset value.

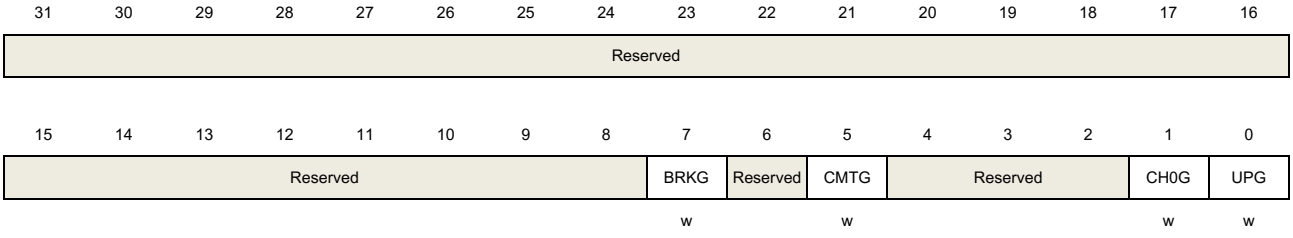
7	BRKIF	Break interrupt flag When the break input is inactive, the bit is set by hardware. When the break input is inactive, the bit can be cleared by software. 0: No active level break has been detected. 1: An active level has been detected.
6	Reserved	Must be kept at reset value
5	CMTIF	Channel commutation interrupt flag This flag is set by hardware when channel's commutation event occurs, and cleared by software 0: No channel commutation interrupt occurred 1: Channel commutation interrupt occurred
4:2	Reserved	Must be kept at reset value
1	CH0IF	Channel 0 's capture/compare interrupt flag This flag is set by hardware and cleared by software. When channel 0 is in input mode, this flag is set when a capture event occurs. When channel 0 is in output mode, this flag is set when a compare event occurs. If Channel0 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 on an update event 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)



Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value
7	BRKG	Break event generation This bit is set by software and cleared by hardware automatically. When this bit is

		set, the POEN bit is cleared and BRKIF flag is set, related interrupt or DMA transfer can occur if enabled. 0: No generate a break event 1: Generate a break event
6	Reserved	Must be kept at reset value
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's c/c control update event
4:2	Reserved	Must be kept at reset value
1	CH0G	Channel 0's capture or compare event generation This bit is set by software in order to generate a capture or compare event in channel 0, it is automatically cleared by hardware. When this bit is set, the CH0IF flag is set, the corresponding interrupt or DMA request is sent if enabled. In addition, if channel 1 is configured in input mode, the current value of the counter is captured in TIMERx_CH0CV register, and the CH0OF flag is set if the CH0IF flag was already high. 0: No generate a channel 1 capture or compare event 1: Generate a channel 1 capture or compare event
0	UPG	Update event generation This bit can be set by software, and cleared by hardware automatically. When this bit is set, the counter is cleared in up counting mode is selected. 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	Reserved	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. 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	Reserved	Must be kept at reset value.

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: Reserved. 11: 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 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												CH0NP	CH0NEN	CH0P	CH0EN
												rw	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, together with CH0P, this bit is used to define the polarity of CI0. This bit cannot be modified when PROT [1:0] bit-filed 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 channel0. 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. [CH0NP, CH0P] will select the active trigger or capture polarity for CI0FE0 or CI1FE0. [CH0NP==0, CH0P==0]: CIxFE0's rising edge is the active signal for capture or

		trigger operation in slave mode. And CIxFE0 will not be inverted. [CH0NP==0, CH0P==1]: CIxFE0's falling edge 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]: CIxFE0's falling and rising edge are both the active signal for capture or trigger operation in slave mode. And CIxFE0 will be not inverted. This bit cannot be modified when PROT [1:0] bit-filed 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 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-filed indicates the current counter value. Writing to this bit-filed 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-filed specifies the auto reload value of the counter. **Note:** When the timer is configured in input capture mode, this register must be configured a non-zero value (such as 0xFFFF) which is larger than user expected value.

Counter repetition register (TIMERx_CREP)

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															

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

rw

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	CREP[7:0]	Counter repetition value This bit-filed specifies the update event generation rate. Each time the repetition counter counting down to zero, an update event is generated. The update rate of the shadow registers is also affected by this bit-filed 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)

31	30	29	28	27	26	25	24	23	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 or compare value of channel0 When channel 0 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 0 is configured in output mode, this bit-filed contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates every update event.

Complementary channel protection register (TIMERx_CCHP)

Address offset: 0x44

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
POEN	OAEN	BRKP	BRKEN	ROS	IOS	PROT[1:0]		DTCFG[7:0]							
rw	rw	rw	rw	rw	rw	rw		rw							

Bits	Fields	Descriptions

31:16	Reserved	Must be kept at reset value
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 CKM 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 9-9. 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 9-9. 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 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.

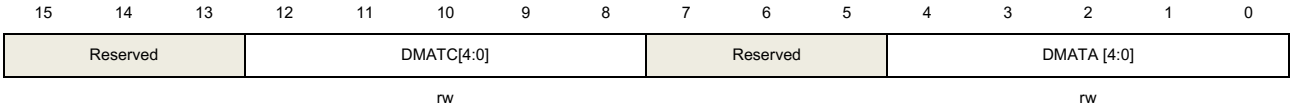
DMA configuration register (TIMERx_DMACFG)

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															

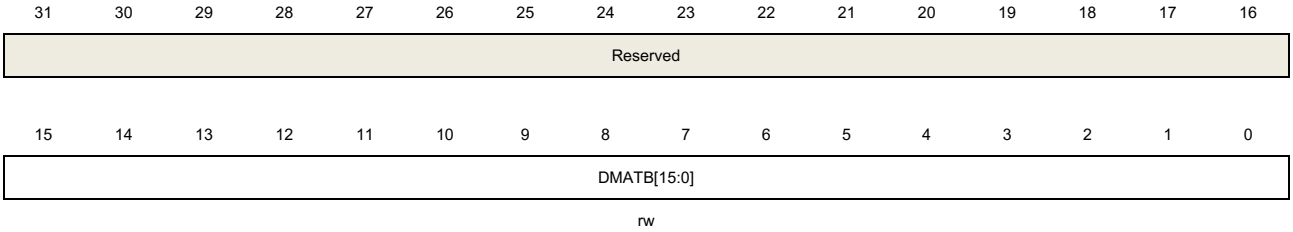


Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:8	DMATC [4:0]	DMA transfer count This filed defines the number(n) of the register that DMA will access(R/W), n = (DMATC [4:0] +1). DMATC [4:0] is from 5'b0_0000 to 5'b1_0001.
7:5	Reserved	Must be kept at reset value.
4:0	DMATA [4:0]	DMA transfer access start address This filed 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)

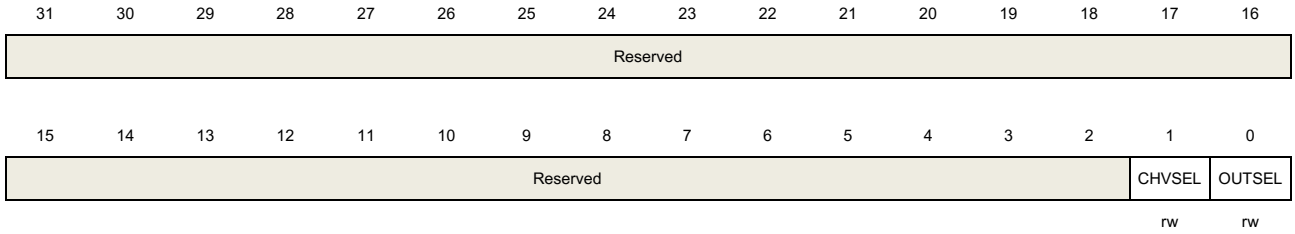


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	DMATB[15:0]	DMA transfer buffer When a read or write operation is assigned to this register, the register located at the address range (Start Addr + Transfer Timer* 4) will be accessed. The transfer Timer is calculated by hardware, and ranges from 0 to DMATC.

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. 0: No effect. 1: If write the CHxVAL register, the write value is same as the CHxVAL value, the write access ignored.
0	OUTSEL	The output value selection This bit-field set and reset by software 0: No effect. 1: If POEN and IOS are 0, the output disabled.

9.6. Basic timer (TIMERx, x=5, 6)

9.6.1. Overview

The basic timer module (TIMER5, TIMER6) reference is a 16-bit counter that can be used as an unsigned counter. The basic timer can be configured to generate DMA request.

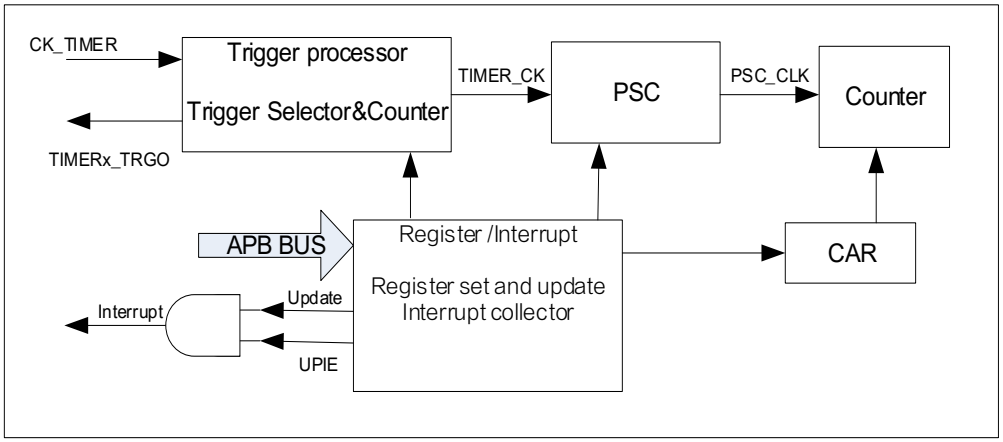
9.6.2. Characteristics

- Counter width: 16bit.
- Source of count clock is internal clock only.
- Counter modes: only count up.
- Programmable prescaler: 16 bits. Factor can be changed ongoing.
- Auto-reload function.
- Interrupt output or DMA request on update event.

9.6.3. Block diagram

Figure 9-82. Basic timer block diagram provides details on the internal configuration of the basic timer.

Figure 9-82. Basic timer block diagram



9.6.4. Function overview

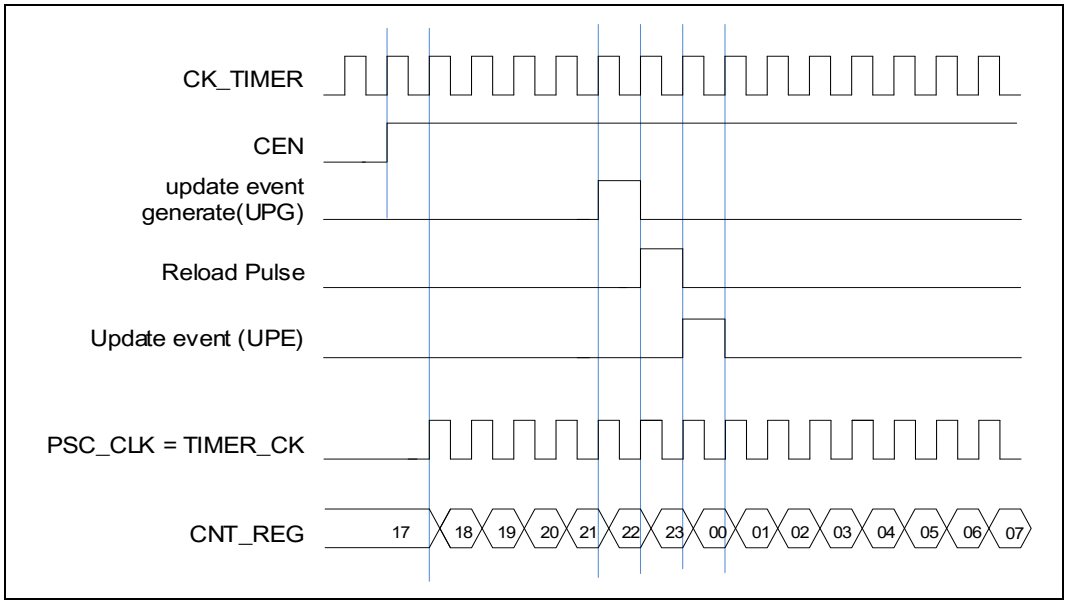
Clock source configuration

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.

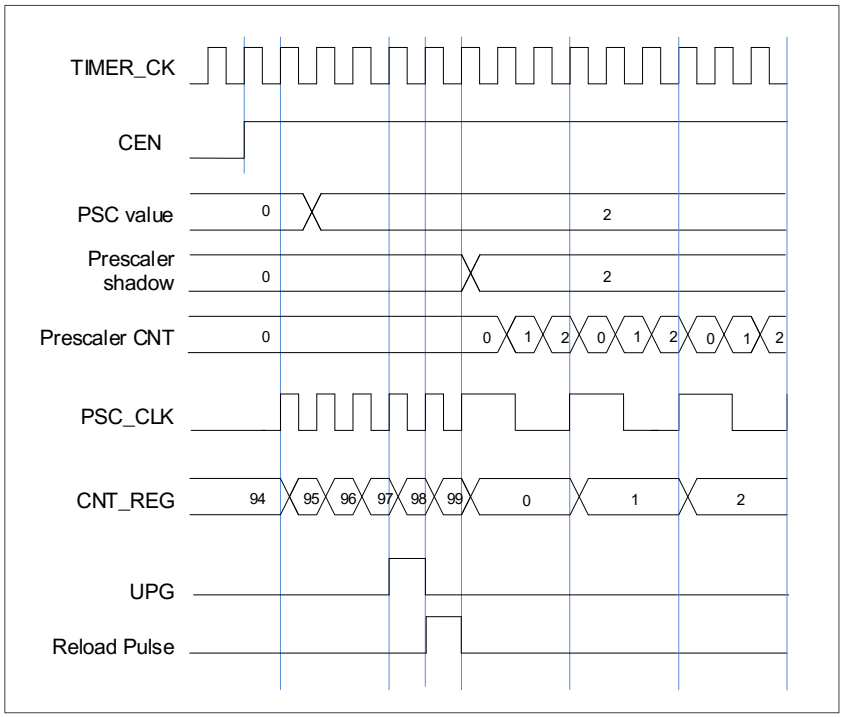
Figure 9-83. Timing chart of internal clock divided by 1



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 9-84. 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 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 9-85. Timing chart of up counting mode, PSC=0/2

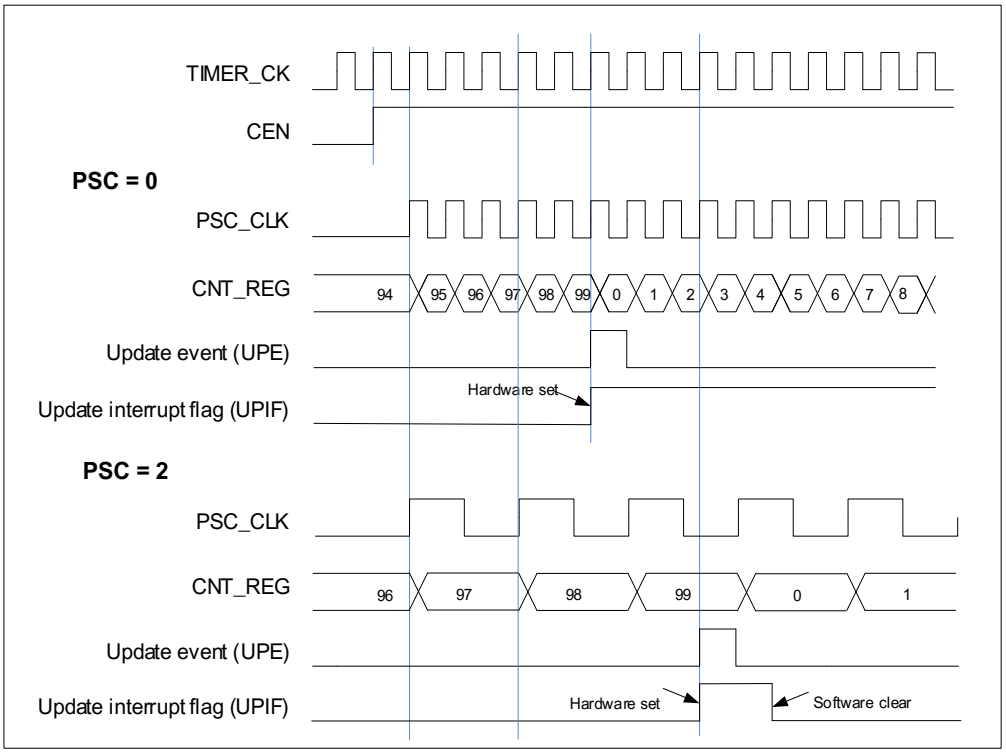
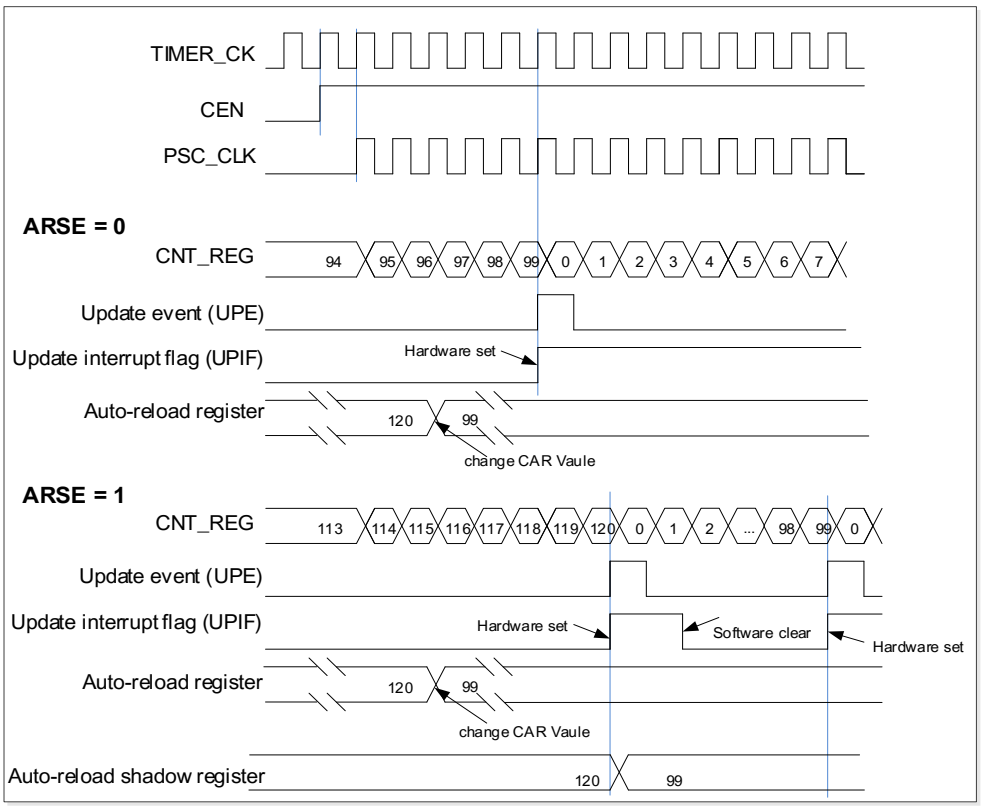


Figure 9-86. Timing chart of up counting mode, change TIMERx_CAR on the go



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®-M23 halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register set to 1, the TIMERx counter stops.

9.6.5. TIMERx registers(x=5, 6)

TIMER5 base address: 0x4000 1000

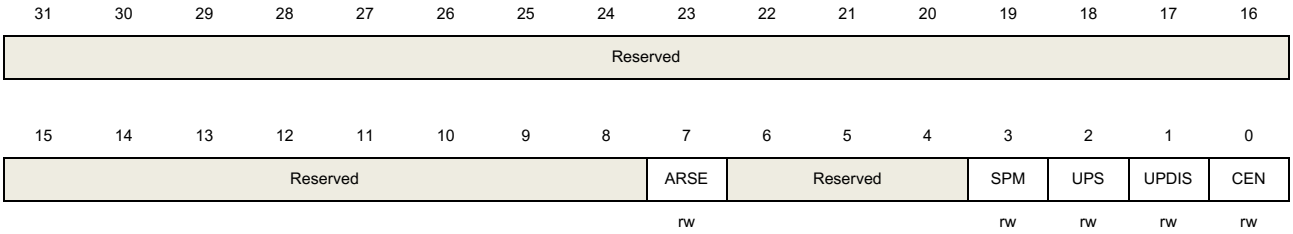
TIMER6 base address: 0x4000 1400

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

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						

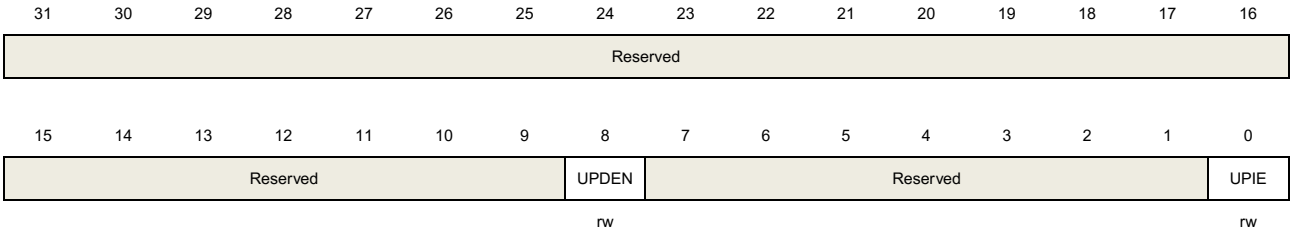
Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value
6:4	MMC[2:0]	Master mode control These bits control the selection of TRGO signal, which is sent in master mode to slave timers for synchronization function. 000: When a counter reset event occurs, a TGRO trigger signal is output. The 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)



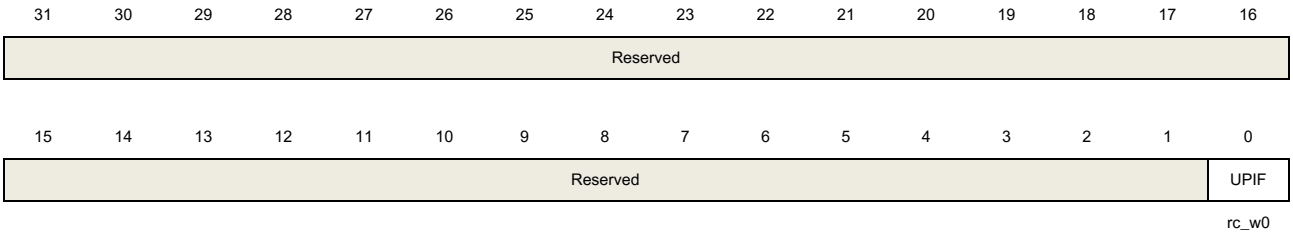
Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	UPDEN	Update DMA request enable 0: disabled 1: enabled
7:1	Reserved	Must be kept at reset value.
0	UPIE	Update interrupt enable 0: disabled 1: enabled

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 on an update event 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 cleared by hardware automatically. 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-filed indicates the current counter value. Writing to this bit-filed 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-filed specifies the auto reload value of the counter.

10. 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 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.

10.1. Free watchdog timer (FWDGT)

10.1.1. 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 or the counter is refreshed when the value of the counter is greater than the window register value. The register write protection function in free watchdog can be enabled to prevent it from changing the configuration unexpectedly.

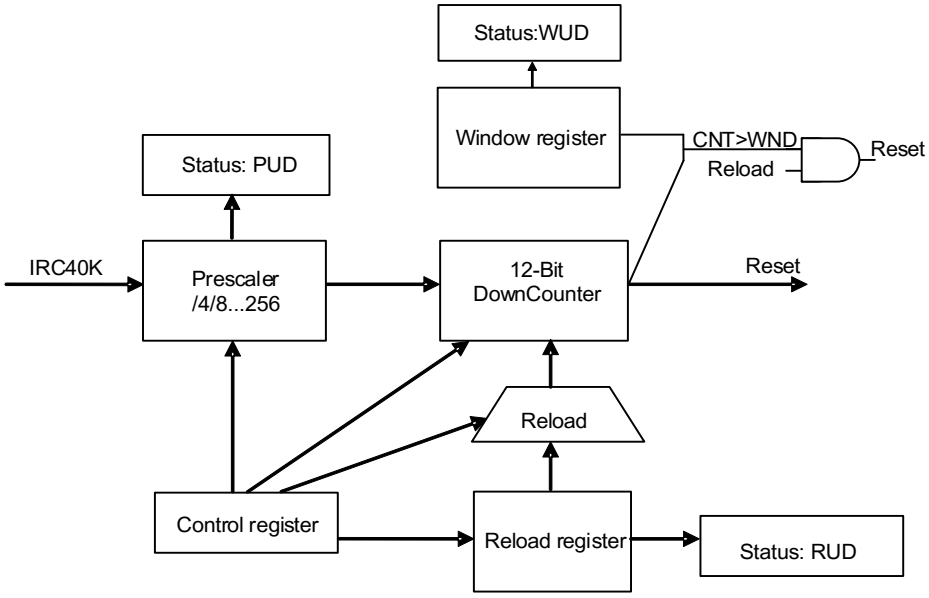
10.1.2. Characteristics

- Free-running 12-bit down counter.
- Generate reset in two conditions when FWDGT is enabled:
  - Reset when the counter reached 0.
  - The counter is refreshed when the value of the counter is greater than the window register value.
- 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.

10.1.3. Function overview

The free watchdog consists of an 8-stage prescaler and a 12-bit down counter. ***Figure 10-1. Free watchdog block diagram*** shows the functional block of the free watchdog module.

Figure 10-1. Free watchdog block diagram



The free watchdog is enabled by writing the value (0xCCCC) to the control register (FWDGT_CTL), then 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).

By setting the appropriate window in the FWDGT_WND register, the FWDGT can also work as a window watchdog timer. A reset will occur if the reload operation is performed while the counter is greater than the value stored in the window register (FWDGT_WND). The default value of the FWDGT_WND is 0x0000 0FFF, so if it is not updated, the window option is disabled. A reload operation is performed in order to reset the downcounter to the FWDGT_RLD value and the prescaler counter to generate the next reload, as soon as the window value is changed.

The free watchdog can automatically start at power on when the hardware free watchdog bit in the device option bits is set. To avoid reset, the software should reload the counter before the counter reaches 0x000.

The FWDGT_PSC register, the FWDGT_RLD register and the FWDGT_WND register. 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), status register (FWDGT_STAT) 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®-M23 core halted (Debug mode). The FWDGT stops in Debug mode if the FWDGT_HOLD bit is set.

Table 10-1. Min/max FWDGT timeout period at 40 kHz (IRC40K)

Prescaler divider	PSC[2:0] bits	Min timeout (ms) RL[11:0]= 0x000	Max timeout (ms) RL[11:0]= 0xFFF
1 / 4	000	0.1	409.6
1 / 8	001	0.2	819.2
1 / 16	010	0.4	1638.4
1 / 32	011	0.8	3276.8
1 / 64	100	1.6	6553.6
1 / 128	101	3.2	13107.2
1 / 256	110 or 111	6.4	26214.4

The FWDGT timeout can be more accurately 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 IRC32K clock cycles between two consecutive watchdog reload operations.

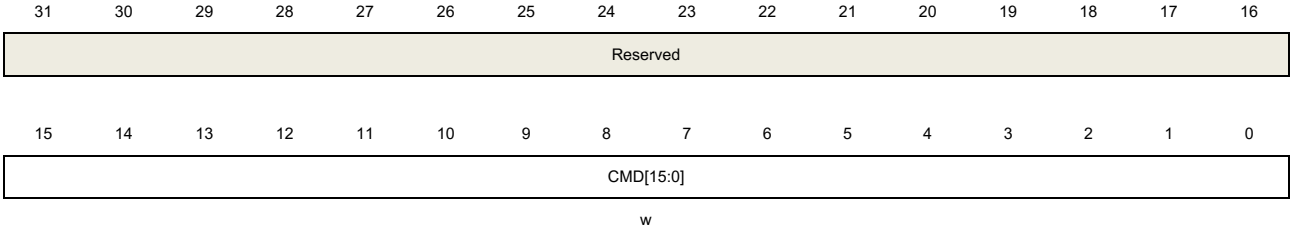
10.1.4. Register definition

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.

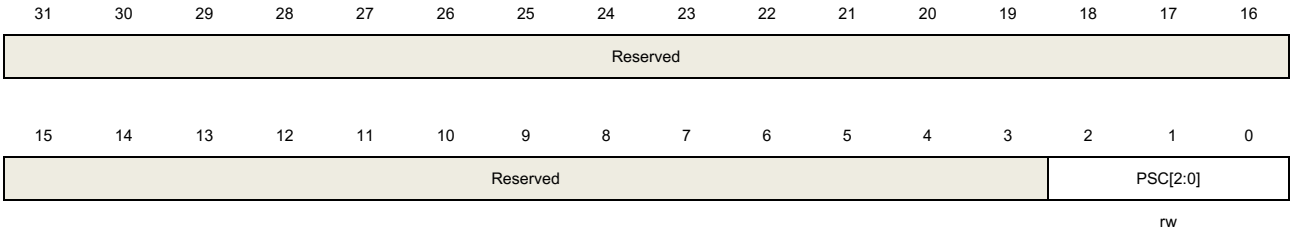


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CMD[15:0]	Write only. Several different functions are realized by writing these bits with different values. 0x5555: Disable the FWDGT_PSC, FWDGT_RLD and FWDGT_WND write protection. 0xCCCC: Start the free watchdog timer counter. 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 in 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.

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

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

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 in the FWDGT_CTL register will reload the FWDGT conter 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 RUD 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.

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													WUD	RUD	PUD
													r	r	r

Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	WUD	Watchdog counter window value update When a write operation to FWDGT_WND register ongoing, this bit is set and the value read from FWDGT_WND register is invalid.
1	RUD	Free watchdog timer counter reloads value update During a write operation to FWDGT_RLD register, this bit is set and the value read from FWDGT_RLD register is invalid.
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.

Window register (FWDGT_WND)

Address offset: 0x10

Reset value: 0x0000 0FFF

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

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

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

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	WND[11:0]	Watchdog counter window value. These bits are used to contain the high limit of the window value to be compared to the downcounter. A reset will occur if the reload operation is performed while the counter is greater than the value stored in this register. The WUD bit in the FWDGT_STAT register must be reset in order to be able to change the reload value.

These bits are write protected. Write 0x5555 in the FWDGT_CTL register before writing these bits.

If several window values are used by the application, it is mandatory to wait until WUD bit has been reset before changing the window value. However, after updating the window value it is not necessary to wait until WUD is reset before continuing code execution except in case of low-power mode entry (Before entering low-power mode, it is necessary to wait until WUD is reset).

10.2. Window watchdog timer (WWDGT)

10.2.1. 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 = 0x3F). 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.

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.

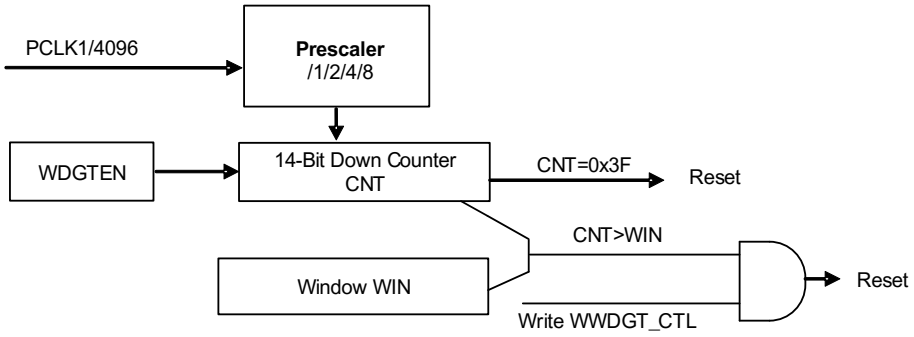
10.2.2. Characteristics

- Programmable free-running 14-bit down counter.
- Generate 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.

10.2.3. 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, or the counter is refreshed before the counter reaches the window register value.

Figure 10-2. Window watchdog timer block diagram



The window watchdog timer is always disabled after power on reset. The software starts the

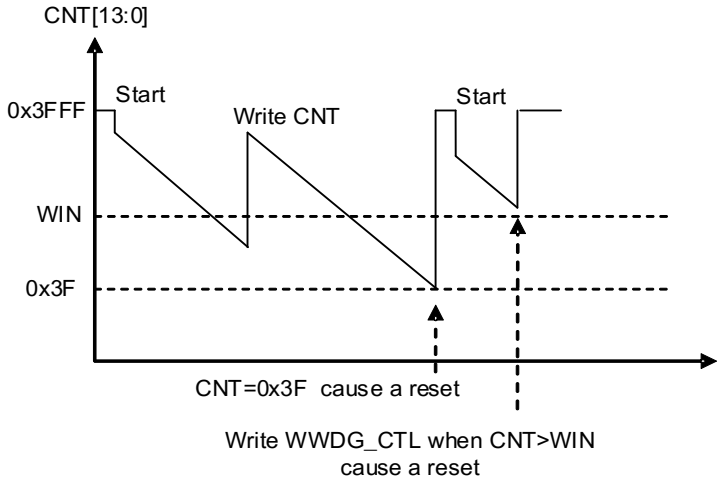
watchdog by setting the WDGTEN bit in the WWDGT_CTL register. When window watchdog timer is enabled, the counter counts down all the time, the configured value of the counter should be greater than 0x3F. The count down speed depends on the APB1 clock and the prescaler (PSC[1:0] bits in the WWDGT_CFG register).

The WIN[13:0] bits in the configuration register (WWDGT_CFG) specifies 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 analyse 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 10-3. Window watchdog timing diagram



Calculate the WWDGT timeout by using the formula below.

$$t_{WWDGT}=t_{PCLK1} \times 4096 \times 2^{PSC} \times (\text{CNT}[13:0] - 0x3F) \quad \text{(ms)} \qquad \text{(13-1)}$$

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 10-2. Min-max timeout value at 72 MHz (f_{PCLK1})

Prescaler divider	PSC[1:0]	Min timeout value	Max timeout value

		CNT[13:0] =0x40	CNT[13:0]=0x3FFF
1 / 1	00	56.89μs	0.928s
1 / 2	01	113.78μs	1.857s
1 / 4	10	227.56μs	3.714s
1 / 8	11	455.11μs	7.427s

If the WWDGT_HOLD bit in DBG module is cleared, the WWDGT continues to work even the Cortex®-M23 core halted (Debug mode). While the WWDGT_HOLD bit is set, the WWDGT stops in Debug mode.

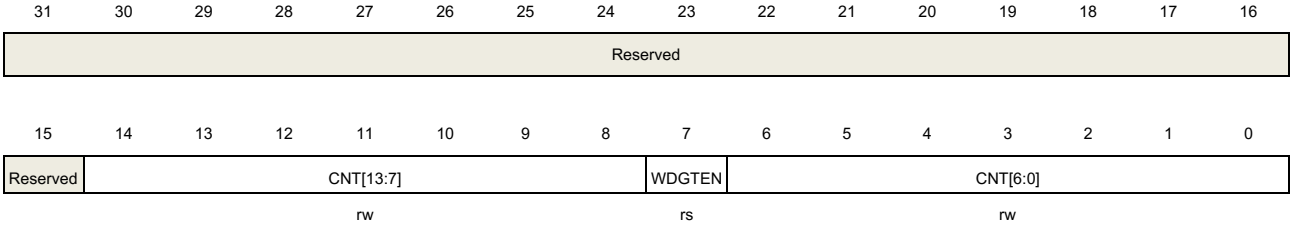
10.2.4. Register definition

WWDGT base address: 0x4000 2C00

Control register (WWDGT_CTL)

Address offset: 0x00
Reset value: 0x0000 7F7F

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

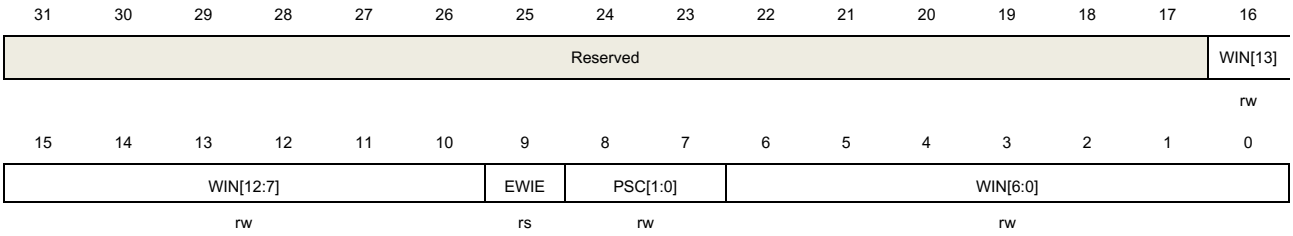


Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14:8	CNT[13:7]	The high 7-bit value of the watchdog timer counter. Refer to CNT[6:0]
7	WDGTEN	Start the Window watchdog timer. Cleared by a hardware reset. Writing 0 has no effect. 0: Window watchdog timer disabled 1: Window watchdog timer enabled
6:0	CNT[6:0]	The low 7-bit value of the watchdog timer counter. A reset occur 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: 0x0001 FC7F

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



Bits	Fields	Descriptions

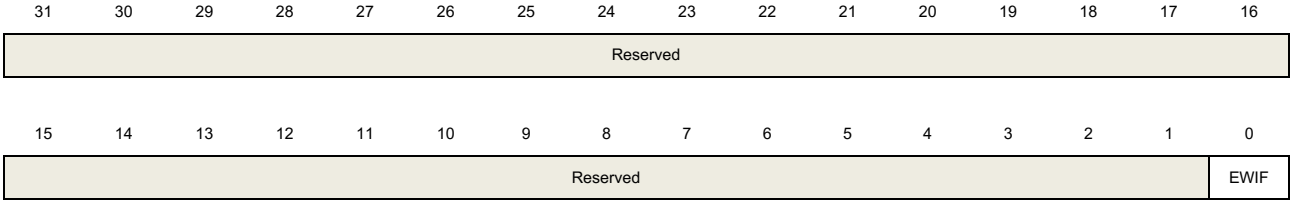
31:17	Reserved	Must be kept at reset value.
16:10	WIN[13:7]	The high 7-bit window value. Refer to WIN[6:0]
9	EWIE	Early wakeup interrupt enables. If the bit is set, an interrupt occurs when the counter reaches 0x40. It can be cleared by a hardware reset or software reset by setting the WWDGTRST bit of the RCU module. A write operation of 0 has no effect.
8:7	PSC[1:0]	Prescaler. The time base of the watchdog counter 00: (PCLK1 / 4096) / 1 01: (PCLK1 / 4096) / 2 10: (PCLK1 / 4096) / 4 11: (PCLK1 / 4096) / 8
6:0	WIN[6:0]	The low 7-bit window value. A reset occur 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 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.

11. Debug (DBG)

11.1. Overview

The GD32E25x 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®-M23. The debug system supports serial wire debug (SWD) and trace functions. The debug and trace functions refer to the following documents:

- Cortex®-M23 Technical Reference Manual
- Arm Debug Interface v5 Architecture Specification

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

11.2. SW function overview

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

11.2.1. Pin assignment

The synchronous serial wire debug (SWD) provide 2-pin SW interface, known as SW data input/output (SWDIO) and SW clock (SWCLK).

The pin assignment is as following:

SWCLK: PA14(GD32E252xx) / PF2(GD32E251xx)

SWDIO: PA13(GD32E252xx) / PB11(GD32E251xx)

If SWD not used, all 2-pin can be released to other GPIO functions. Please refer to ***General-purpose I / O (GPIO)***.

11.3. Debug hold function overview

11.3.1. Debug support for power saving mode

When the STB_HOLD bit in DBG control register 0 (DBG_CTL0) 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 generated.

When the DSLP_HOLD bit in DBG control register 0 (DBG_CTL0) 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 0 (DBG_CTL0) 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 and FWDGT

When the core is halted and the corresponding bit in DBG control register 0 or DBG control register 1 (DBG_CTL0 or DBG_CTL1) 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.

11.4. Register definition

DBG base address: 0x4001 5800

11.4.1. ID code register (DBG_ID)

Address offset: 0x00

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 0 (DBG_CTL0)

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.				TIMER13_HOLD	Reserved						TIMER6_HOLD	TIMER5_HOLD	Reserved	I2C2_HOLD	I2C1_HOLD
				rw							rw	rw		rw	rw

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

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
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.
26:21	Reserved	Must be kept at reset value.
20	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.
19	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.
18	Reserved	Must be kept at reset value
17	I2C2_HOLD	I2C2 hold bit This bit is set and reset by software. 0: no effect 1: hold the I2C2 status to avoid SMBUS timeout 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.
14:13	Reserved	Must be kept at reset value.
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.
7:3	Reserved	Must be kept at reset value.
2	STB_HOLD	Standby mode hold bit 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 bit 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 bit This bit is set and reset by software. 0: no effect 1: In the sleep mode, the clock of AHB is on.

11.4.3. Control register 1 (DBG_CTL1)

Address offset: 0x08

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.													TIMER16_HOLD	TIMER15_HOLD	TIMER14_HOLD
													rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18	TIMER16_HOLD	TIMER 16 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 16 counter for debugging when the core is halted.

17	TIMER15_HOLD	TIMER 15 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 15 counter for debugging when the core is halted.
16	TIMER14_HOLD	TIMER 14 hold bit This bit is set and reset by software. 0: no effect 1: hold the TIMER 14 counter for debugging when the core is halted.
15:0	Reserved	Must be kept at reset value.

12. Cyclic redundancy checks management unit (CRC)

12.1. Overview

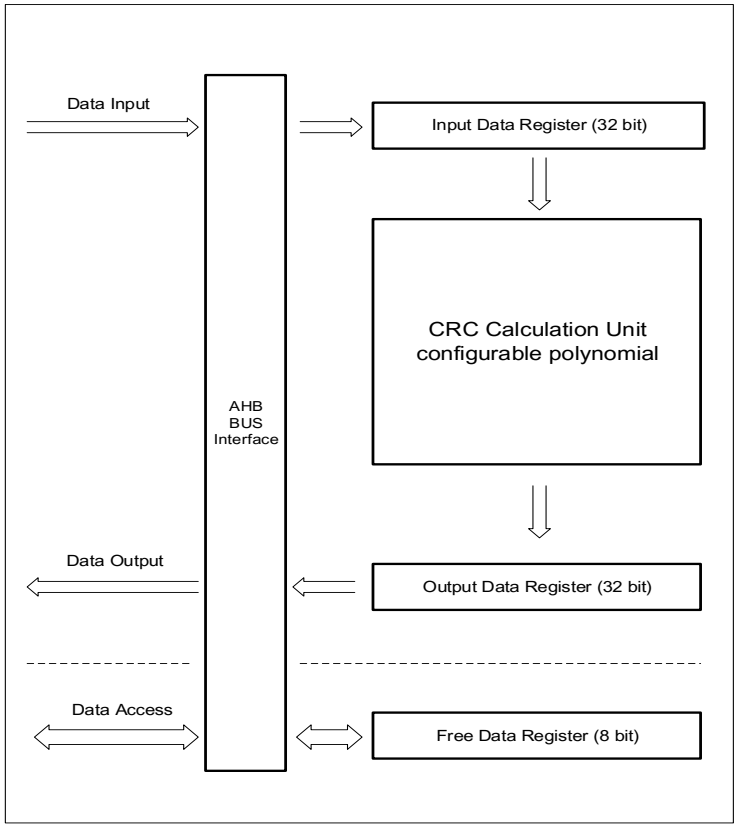
A cyclic redundancy checks (CRC) is an error-detecting code commonly used in digital networks and storage devices to detect accidental changes to raw data.

This CRC management unit can be used to calculate 7 / 8 / 16 / 32 bit CRC code within user configurable polynomial.

12.2. Characteristics

- Input data supports 7 / 8 / 16 / 32 size bit.
- Different input size for different calculation time.1 / 2 / 4 cycle for 7(8) / 16 / 32 bits.
- Input and output data can be reversed.
- User configurable polynomial size.
- User configurable initial value after CRC reset.
- Free 8-bit register is unrelated to calculation and can be used for any other goals by any other peripheral devices.

Figure 12-1. Block diagram of CRC calculation unit



12.3. Function overview

- CRC management 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 / 2 / 1 AHB clock cycles for 32 / 16 / 8(7) bit data size, during this period AHB will not be hanged because of the existence of the 32bit 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.
- Reversible function can reverse the input data and output data.

  For input data, 3 reverse types can be selected.

  Original data is 0x3456CDEF:

1) byte reverse:
   32-bit data is divided into 4 groups and reverse implement in group inside. Reversed data: 0x2C6AB3F7

2) half-word reverse:
   32-bit data is divided into 2 groups and reverse implement in group inside. Reversed data: 0x6A2CF7B3

3) word reverse:
   32-bit data is divided into 1 groups and reverse implement in group inside. Reversed data: 0xF7B36A2C

   For output data, reverse type is word reverse.
   For example: when REV_O=1, calculation result 0x3344CCDD will be converted to 0xBB3322CC.

- User configurable initial calculation data is available.
  - When RST bit is set or write operation to CRC_IDATA register, the CRC_DATA register will be automatically initialized to the value in CRC_IDATA.
- User configurable polynomial.
  - Depends on PS[1:0] bits, the valid polynomial and output bit width can be selected by user. If the polynomial is less than 32 bit, the high bits of the input data and output data is unavailable. It is strongly recommend resetting the CRC management unit after change the PS[1:0] bits or polynomial.

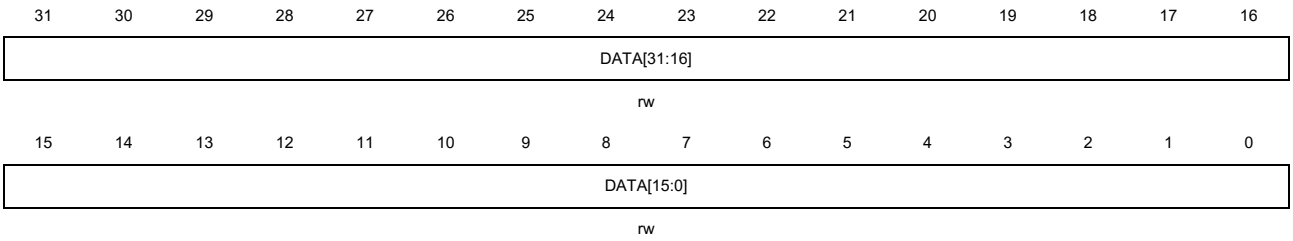
12.4. Register definition

CRC base address: 0x4002 3000

12.4.1. Data register (CRC_DATA)

Address offset: 0x00
Reset value: 0xFFFF FFFF

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

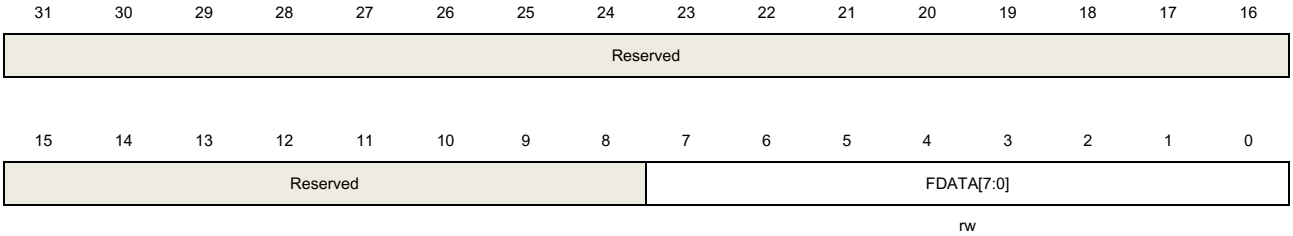


Bits	Fields	Descriptions
31:0	DATA[31:0]	CRC calculation result bits Software writes and reads. This register is used to calculate new data, and the register can be written the new data directly. Write value cannot be read because the read value is the previous CRC calculation result.

12.4.2. 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

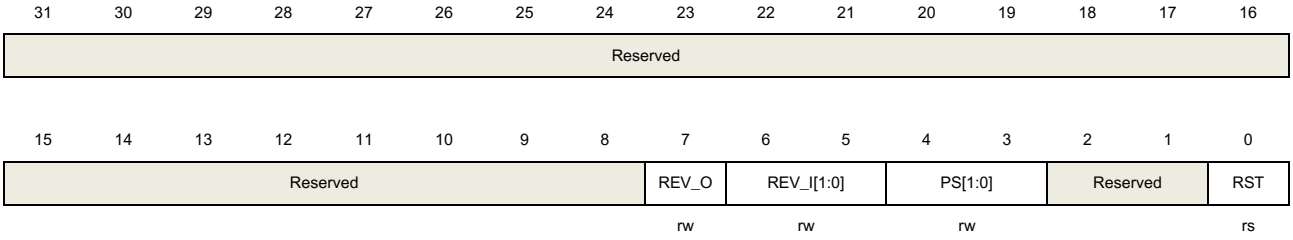
by any other peripheral. The CRC_CTL register will generate no effect to the byte.

12.4.3. Control register (CRC_CTL)

Address offset: 0x08

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	REV_O	Reverse output data value in bit order 0: Not bit reversed for output data 1: Bit reversed for output data
6:5	REV_I[1:0]	Reverse type for input data 00: Dot not use reverse for input data 01: Reverse input data with every 8-bit length 10: Reverse input data with every 16-bit length 11: Reverse input data with whole 32-bit length
4:3	PS[1:0]	Size of polynomial 00: 32 bit 01: 16 bit (POLY[15:0] is used for calculation) 10: 8 bit (POLY[7:0] is used for calculation) 11: 7 bit (POLY[6:0] is used for calculation)
2:1	Reserved	Must be kept at reset value.
0	RST	Set this bit can reset the CRC_DATA register to the value in CRC_IDATA then automatically cleared itself to 0 by hardware. This bit will take no effect to CRC_FDATA. Software writes and reads.

12.4.4. Initialization data register (CRC_IDATA)

Address offset: 0x10

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
IDATA[31:16]															
rw															

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

Bits	Fields	Descriptions
31:0	IDATA[31:0]	Configurable initial CRC data value When RST bit in CRC_CTL asserted, CRC_DATA will be programmed to this value.

12.4.5. Polynomial register (CRC_POLY)

Address offset: 0x14

Reset value: 0x04C1 1DB7

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
POLY[31:16]															
rw															

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

Bits	Fields	Descriptions
31:0	POLY[31:0]	User configurable polynomial value This value is used together with PS[1:0] bits.

13. Analog to digital converter (ADC)

13.1. Overview

A 12-bit successive approximation analog-to-digital converter module (ADC) is integrated on the MCU chip, which can sample analog signals from 20 external channels and 3 internal channels. The 23 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 alignment or the most significant bit alignment. An on-chip hardware oversample scheme improves performances and reduces the computational burden of MCU.

13.2. 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.
- Dual clock domain architecture (APB clock and ADC clock).
- Analog input channels:
  - 20 external analog inputs.
  - 1 channel for internal high precision temperature sensor (V_{SENSE}).
  - 1 channel for internal reference voltage (V_{REFINT}).
  - 1 channel for 1/2 analog power supply voltage ($V_{DDADIV2}$).
- Start-of-conversion can be initiated:
  - By software.
  - By hardware triggers.
- Operation modes:
  - Converts a single channel or scan a sequence of channels.
  - Single operation mode converts selected input channel once per trigger.
  - Continuous operation mode converts selected input channels continuously.
  - Discontinuous operation mode.
- 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-bits.
- Channel input range: V_{SSA} <= V_{IN} <= V_{REFP}

13.3. Pins and internal signals

Figure 13-1. ADC module block diagram shows the ADC block diagram. *Table 13-1. ADC internal input signals* and *Table 13-2. ADC input pins definition* give the ADC internal signals and pins description.

Table 13-1. ADC internal input signals

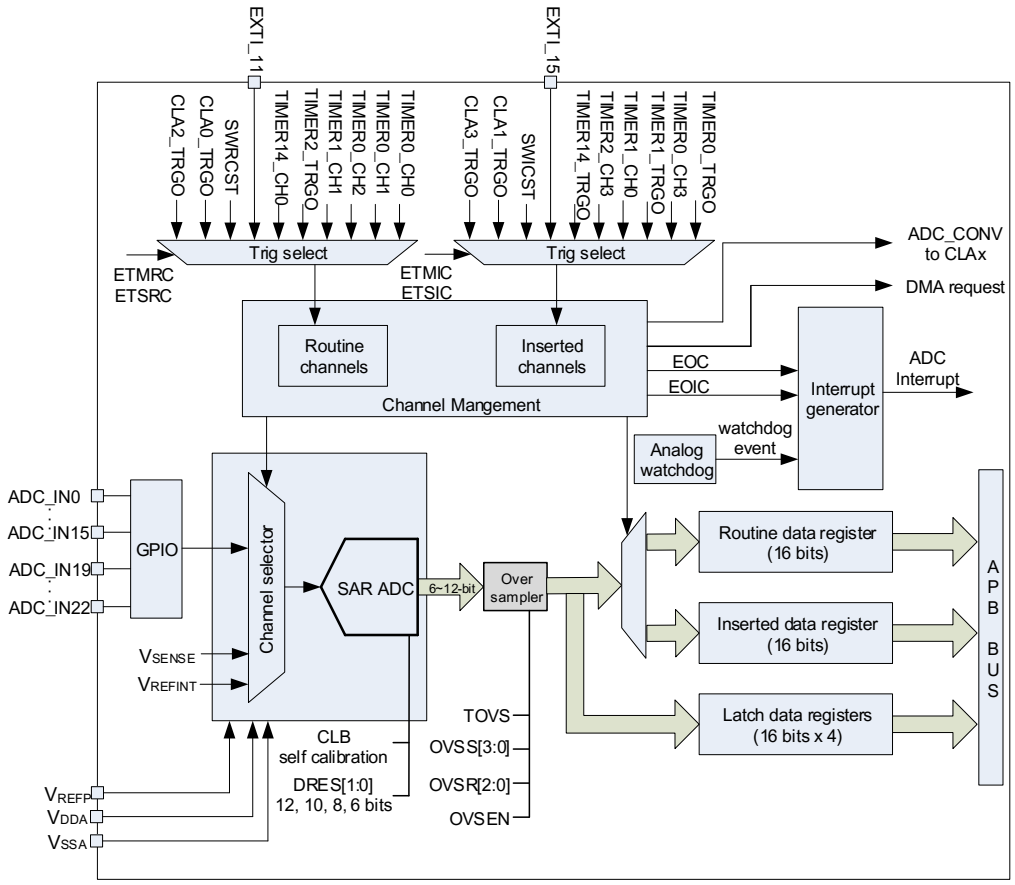
Internal signal name	Description
V_{SENSE}	Internal high precision temperature sensor output voltage
V_{REFINT}	Internal 1.2V voltage reference output voltage
$V_{DDADIV2}$	1/2 analog power supply voltage

Table 13-2. ADC input pins definition

Name	Description
V_{DDA}	Analog power supply equals to V_{DD}
V_{REFP}	The positive reference reference voltage for the ADC
V_{SSA}	Ground for analog power supply equals to V_{SS}
ADCx_IN [15:0], ADCx_IN [22:19]	Up to 20 external channels

13.4. Function overview

Figure 13-1. ADC module block diagram



13.4.1. 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 then cleared by hardware as soon as the calibration is completed.

When the ADC operating conditions change (such as supply power voltage V_{DDA}, 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 that 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.

13.4.2. Dual clock domain architecture

The ADC sub-module, with exception of the APB interface block, is feed by an ADC clock, which can be asynchronous and independent from the APB clock.

Application can reduce PLCK frequency for low power operation while still keeping optimum ADC performance.

13.4.3. ADC 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 off mode. After ADC is enabled, you need delay t_{su} time for sampling, the value of t_{su} please refer to the device datasheet.

13.4.4. Routine and inserted sequence

The ADC supports 23 channels and the sampling channels can be devided into two sequence: a routine sequence and an inserted sequence.

The routine sequence supports up to 23 channels, and each channel is called routine channel. The RL[4:0] bits in the ADC_RSQ0 register specify the total conversion sequence length. The ADC_RSQ0~ADC_RSQ3 registers specify the selected channels of the routine sequence.

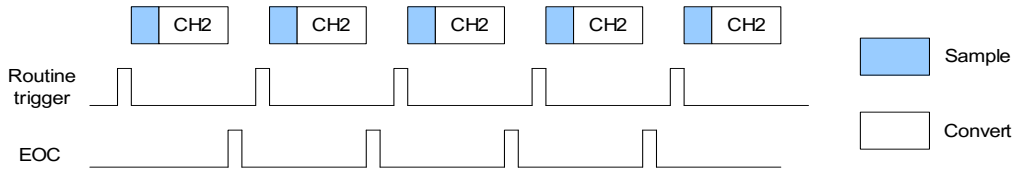
The inserted sequence supports up to 4 channels, and each channel is called inserted channel. The IL[1:0] bits in the ADC_ISQ register specify the total conversion sequence length. The ADC_ISQ register specifies the selected channels of the inserted sequence.

13.4.5. Operation modes

Single operation mode

In the single operation mode, the ADC performs conversion on the channel specified in the RSQ0[4:0] bits in ADC_RSQ2 at a routine trigger. When the ADCON is 1, the ADC samples and converts a single channel, once the corresponding software trigger or external trigger is active.

Figure 13-2. 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.

After conversion of a single inserted channel, the conversion data will be stored in the ADC_IDATA register, the EOIC and EOC will be set. An interrupt will be generated if the EOICIE or 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 the RSQ0 with the analog channel number;
3. Configure the ADC_SAMPTx register;
4. Configure the ETMRC and ETSRC bits in the ADC_CTL1 register if it is needed;
5. Set the SWRCST bit, or generate an external trigger for the routine sequence;
6. Wait for the EOC flag to be set;
7. Read the converted result from the ADC_RDATA register;
8. Clear the EOC flag by writing 0.

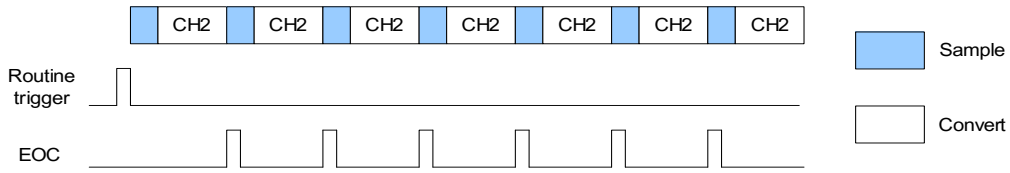
Software procedure for single conversion of inserted channel:

1. Make sure the DISIC, SM in the ADC_CTL0 register are reset;
2. Configure ISQ3 with the analog channel number;
3. Configure ADC_SAMPTx register;
4. Configure the ETMIC and ETSIC bits in the ADC_CTL1 register if in need;
5. Set the SWICST bit, or generate an external trigger for the inserted sequence;
6. Wait for the EOIC, EOC flags to be set;
7. Read the converted result in the ADC_IDATA register;
8. Clear the EOIC, EOC flags by writing 0 to it.

Continuous operation mode

The continuous operation mode will be enabled when the 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 is 1, the ADC samples and converts specified a channel, once the corresponding software trigger or external trigger is active. The conversion data will be stored in the ADC_RDATA register.

Figure 13-3. Continuous operation mode



Software procedure for continuous operation mode 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 ETMRC 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 for the EOC flag to be set.
7. Read the converted date 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.

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

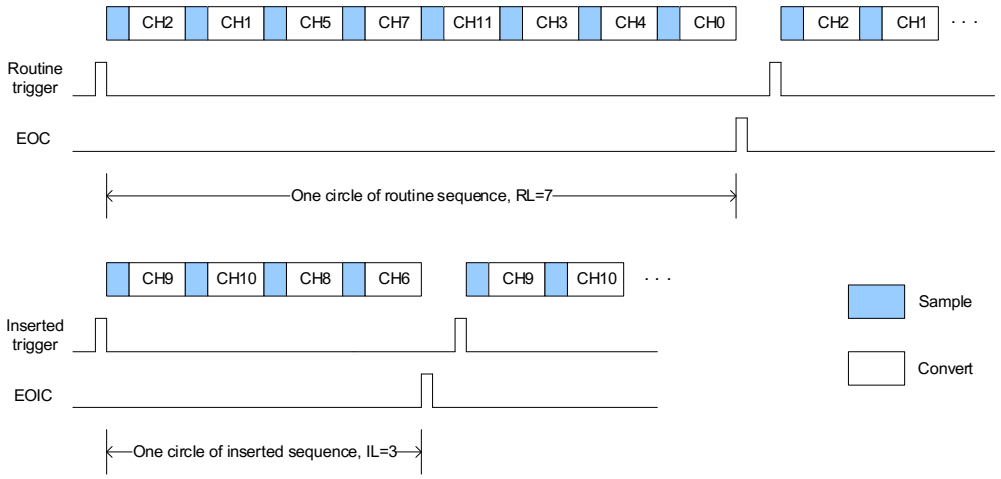
1. Set the CTN, RDMA and RDDM bits in the ADC_CTL1 register.
2. Configure RSQ0 with the analog channel number.
3. Configure ADC_SAMPTx register.
4. Configure the ETMRC and ETSRC bits in the ADC_CTL1 register in if it is needed.
5. Prepare the DMA module to transfer data from the ADC_RDATA.
6. Set the 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 the channels with a specific sequence specified in the ADC_RSQ0~ADC_RSQ3 registers or ADC_ISQ register. When the ADCON has been set high, the ADC samples and converts specified channels one by one in the routine or inserted sequence till the end of the routine or inserted sequence, once the corresponding software trigger or external trigger is active. The conversion data will be stored in the ADC_RDATA or ADC_IDATA register. After conversion of the routine or inserted sequence, the EOC or EOIC will be set. An interrupt will be generated if the EOCIE or EOICIE bit is set. The RDMA or IDMA bit in ADC_CTL1 register can be set when the routine or inserted 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 13-4. Scan operation mode, continuous disable



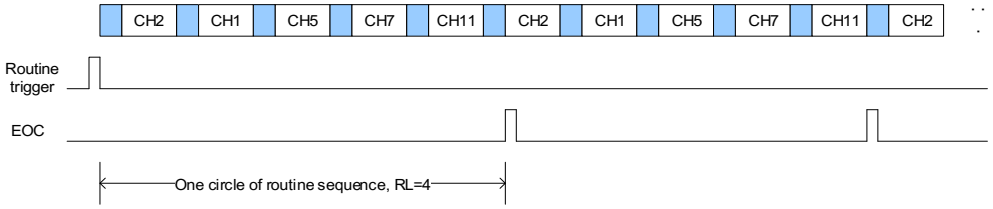
Software procedure for scan operation mode on a routine sequence:

1. Set the SM bit in the ADC_CTL0 register and the RDMA bit in the ADC_CTL1 register.
2. Configure ADC_RSQx and ADC_SAMPTx registers.
3. Configure the ETMRC and ETSRC bits in the ADC_CTL1 register if it is needed.
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.

Software procedure for scan conversion on an inserted sequence:

1. Set the SM bit in the ADC_CTL0 register and the IDMA bit in the ADC_CTL1 register;
2. Configure ADC_ISQ and ADC_SAMPTx registers;
3. Configure ETMIC and ETSIC bits in the ADC_CTL1 register if it in need;
4. Prepare the DMA module to transfer data from the ADC_IDATA;
5. Set the SWICST bit, or generate an external trigger for the inserted sequence.

Figure 13-5. Scan operation mode, continuous enable



Discontinuous operation mode

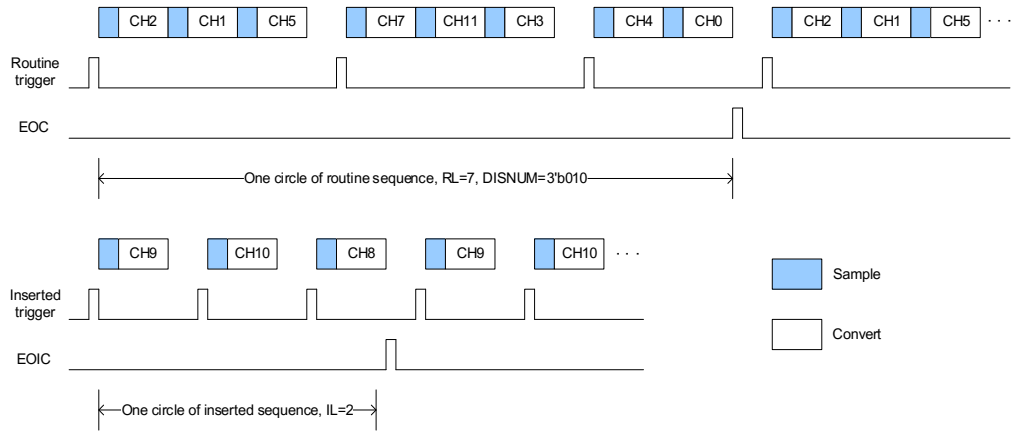
For routine sequence, the discontinuous operation mode will be enabled when the DISRC bit in the ADC_CTL0 register is set. In this mode, the ADC performs a short sequence of n conversions (n does not exceed 8) which is a part of the sequence of conversions configured in the ADC_RSQ0~ADC_RSQ3 registers. 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 coverts the next n channels selected in the ADC_RSQ0~ADC_RSQ3 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.

For inserted sequence, the discontinuous operation mode will be enabled when the DISIC bit in the ADC_CTL0 register is set. In this mode, the ADC performs one conversion which is a part of the sequence of conversions selected in the ADC_ISQ register. When the corresponding software trigger or external trigger is active, the ADC samples and coverts the next channel configured in the ADC_ISQ register until all the inserted channels in the inserted sequence are done. The EOIC will be set after every circle of the inserted sequence. An interrupt will be generated if the EOICIE bit is set.

The routine and inserted sequence cannot both work in discontinuous operation mode. Only one sequence conversion can be set in discontinuous operation mode at a time.

Figure 13-6. Discontinuous operation mode



Software procedure for discontinuous conversion on a routine sequence:

1. Set the DISRC bit in the ADC_CTL0 register and the RDMA 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 the ETMRC 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;
7. Repeat step6 if in need.

Software procedure for discontinuous conversion on an inserted sequence:

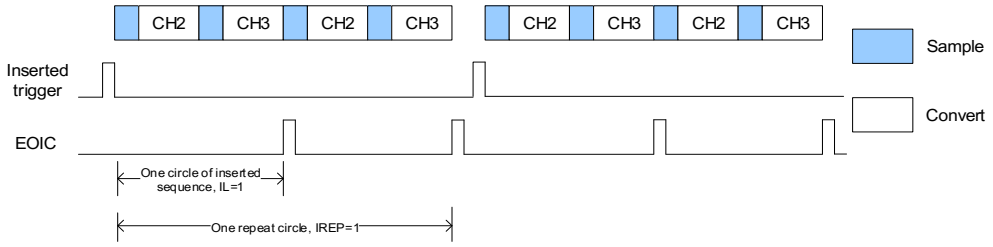
1. Set the DISIC bit in the ADC_CTL0 register and the IDMA bit in the ADC_CTL1 register;
2. Configure ADC_ISQ and ADC_SAMPTx registers;
3. Configure ETMIC and ETSIC bits in the ADC_CTL1 register if in need;
4. Prepare the DMA module to transfer data from the ADC_IDATA;
5. Set the SWICST bit, or generate an external trigger for the inserted sequence;
6. Repeat step5 if in need.

Repeat operation mode

This mode operates only in an inserted sequence. It must be used in conjunction with single

operation mode or scan operation mode. By configuring the IREP[5:0] bits in the ADC_CTL2 register, the inserted sequence of the ADC can repeatedly sample and convert IREP[5:0] + 1 times when a single software trigger or external trigger occurs.

Figure 13-7. Repeat operation mode (SM=1)



Software procedure for repeat operation mode on an inserted sequence:

1. Set the SM bit in the ADC_CTL0 register and the IDMA bit in the ADC_CTL2 register;
2. Configure the IREP[5:0] bits in the ADC_CTL2 register;
3. Configure the ADC_ISQ and ADC_SAMPTx registers;
4. Configure the ETMIC and ETSIC bits in the ADC_CTL1 register if it in need;
5. Prepare the DMA module to transfer data from the ADC_IDATA;
6. Set the SWICST bit, or generate an external trigger for the inserted sequence;

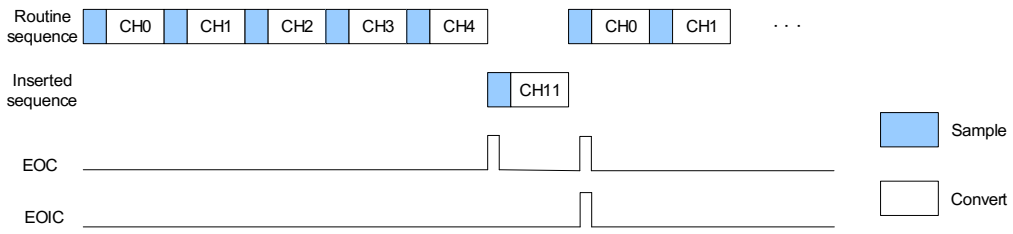
Note: When discontinuous mode on inserted sequence is enabled, the IREP[5:0] bits must keep reset value.

13.4.1. Inserted channel operation

Auto-insertion

The inserted sequence channels are automatically sampled and converted after the routine sequence channels when the ICA bit in ADC_CTL0 register is set. In this mode, external trigger on inserted channels cannot be enabled. A sequence of up to 16 conversions programmed in the ADC_RSQ0~ADC_RSQ2 and ADC_ISQ registers can be used to convert in this mode. In addition to the ICA bit, if the CTN bit is also set, the routine sequence channels and inserted sequence channels are constantly converted.

Figure 13-8. Auto-insertion, CTN = 1

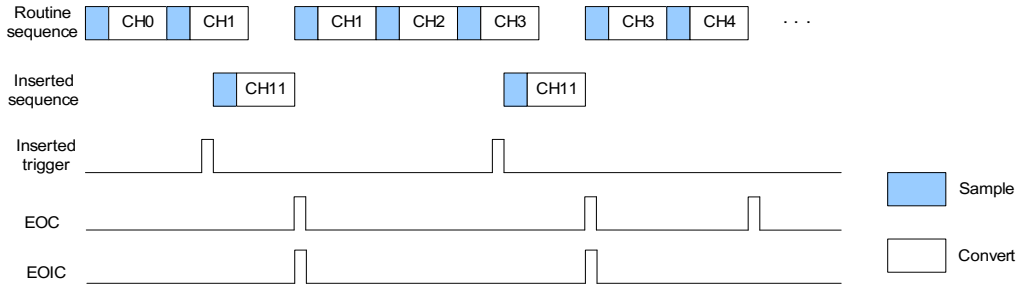


The auto insertion mode can not be enabled when the discontinuous operation mode is set.

- **Triggered insertion**

If the ICA bit in ADC_CTL0 register is cleared, the triggered insertion occurs if a software or external trigger occurs during the routine sequence channel conversion. In this situation, the ADC aborts the current conversion and starts the conversion of inserted sequence. After the inserted sequence is done, the routine channel conversion is resumed from the aborted conversion.

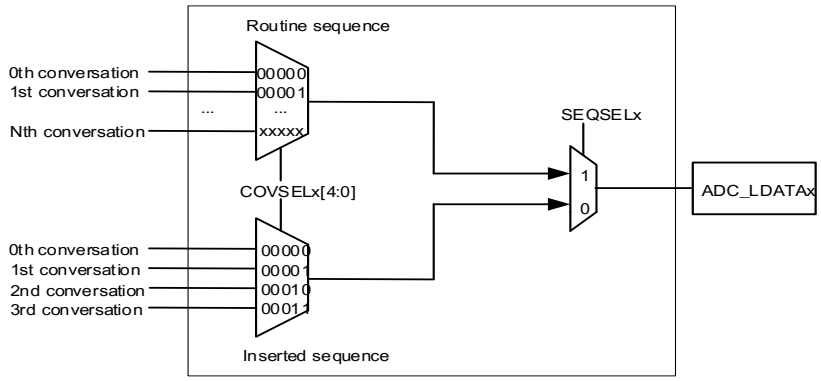
Figure 13-9. Triggered insertion



13.4.2. Convert data latch

The ADC has 4 latch data registers, ADC_LDATAx (x=0…3), which can latch the data from a completed conversion in a routine or inserted sequence. The control logic for the latch data registers is shown in ***Figure 13-10. Data latch***.

Figure 13-10. Data latch

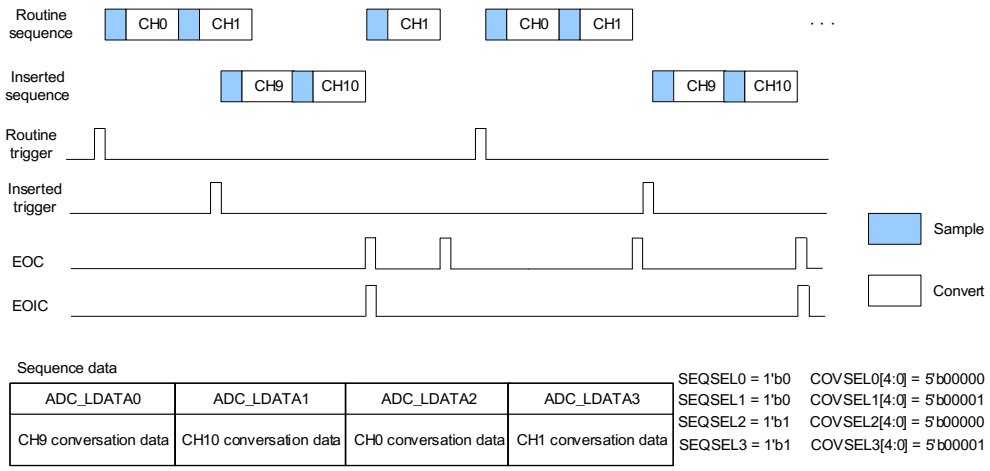


The data to be latched from either the routine or inserted sequence is selected through the SEQSELx (x=0…3) bit in the ADC_LDCTL register, while the COVSELx[4:0] bits selects which conversion result in the sequence to latch.

ADC_LDATAx (x=0…3) registers are set to store the conversion results of the inserted sequence or routine sequence and default store inserted sequence xth conversion results.

Therefore, in addition to using DMA to transfer conversion data from the ADC_RDATA or ADC_IDATA registers, data can also be directly read from the latch data registers, offering greater flexibility for consecutive conversions in a conversion sequence.

Figure 13-11. Sequence data from ADC_LDATAx



Software procedure for triggered insertion mode and read the sequence data from ADC_LDATAx:

1. Set the SM bit in the ADC_CTL0 register;
2. Configure ADC_RSQx, ADC_ISQ and ADC_SAMPTx registers;
3. Configure SEQSEL2, COVSEL2[4:0], SEQSEL3 and COVSEL3[4:0] bits in the ADC_LDCTL register for the routine sequence;
4. Configure SEQSEL0, COVSEL0[4:0], SEQSEL1 and COVSEL1[4:0] bits in the ADC_LDCTL register for the inserted sequence;
5. Configure ETMRC, ETSRC, ETMIC and ETSIC bits in the ADC_CTL1 register if it is needed;
6. Set the SWRCST bit, or generate an external trigger for the routine sequence;
7. Set the SWICST bit, or generate an external trigger for the inserted sequence;
8. Wait for EOC or EOIC flags to be set;
9. If EOC flag is set, and EOIC flag is not set, read the converted result of routine sequence in the ADC_LDATA2 and ADC_LDATA3 register;
10. Clear the EOC flag by writing 0.
11. If EOIC flag is set, read the converted result of inserted sequence in the ADC_LDATA0 and ADC_LDATA1 register;
12. Clear the EOIC, EOC flag by writing 0.

Note: If the routine sequence conversion result is read by querying the EOC flag, it should be noted that the flag will also be set at the end of the inserted sequence conversion.

13.4.3. Conversion result threshold monitor function

The analog watchdog is enabled when the RWDEN and IWDEN bit in the ADC_CTL0 register is set for routine and inserted sequence respectively. 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 selected by the RWDEN, IWDEN, WDSC and WDCHSEL [4:0] bits in ADC_CTL0 register, can be monitored by the analog watchdog.

13.4.4. Data storage mode

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

When the most significant bit alignment, the 12/10/8-bit data are aligned on a half-word, while the 6-bit data are aligned on a byte basis as shown below ***Figure 13-12. Data storage mode of 12-bit resolution***, ***Figure 13-13. Data storage mode of 10-bit resolution***, ***Figure 13-14. Data storage mode of 8-bit resolution*** and ***Figure 13-15. Data storage mode of 6-bit resolution***.

Figure 13-12. Data storage mode of 12-bit resolution

Routine channel data

0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Inserted channel data

Sign	Sign	Sign	Sign	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

DAL=0

Routine channel data

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0

Inserted channel data

Sign	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0

DAL=1

Figure 13-13. Data storage mode of 10-bit resolution

Routine channel data

0	0	0	0	0	0	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Inserted channel data

Sign	Sign	Sign	Sign	Sign	Sign	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

DAL=0

Routine channel data

D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0	0	0

Inserted channel data

Sign	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0	0

DAL=1

Figure 13-14. Data storage mode of 8-bit resolution

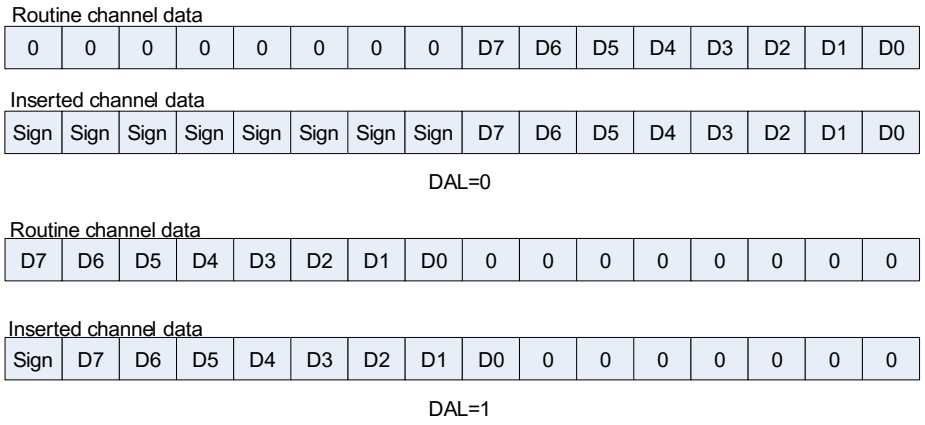
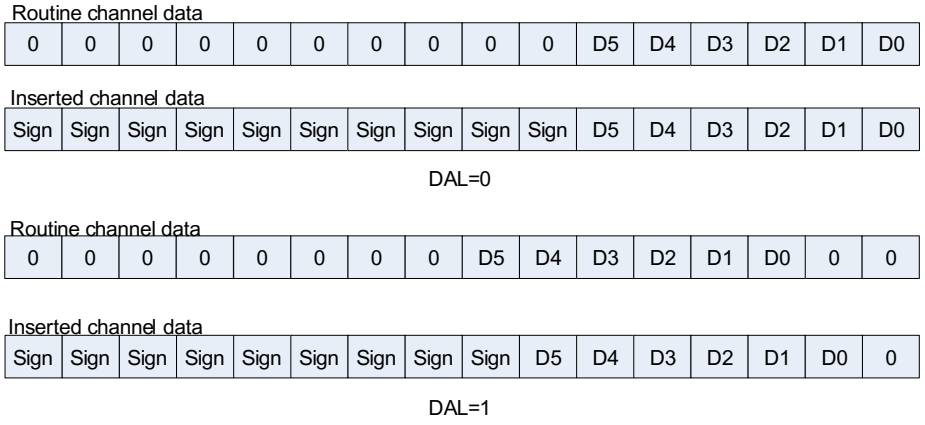


Figure 13-15. Data storage mode of 6-bit resolution



13.4.5. Sample time configuration

The number of ADC_CLK cycles which is used to sample the input voltage can be specified by the SPTn [2:0] bits in the ADC_SAMPT0, ADC_SAMPT1, and ADC_SAMPT2 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 = 28MHz and sample time is 1.5 cycles, the total conversion time is "1.5+12.5" CK_ADC cycles, that means 0.500us.

13.4.6. External trigger configuration

The routine or inserted sequence can be triggered by rising edge of external trigger inputs. The external trigger source of routine sequence is controlled by the ETSRC[3:0] bits in the ADC_CTL1 register, while the external trigger source of inserted sequence is controlled by

the ETSIC[3:0] bits in the ADC_CTL1 register

Table 13-3. External trigger for regular channels of ADC

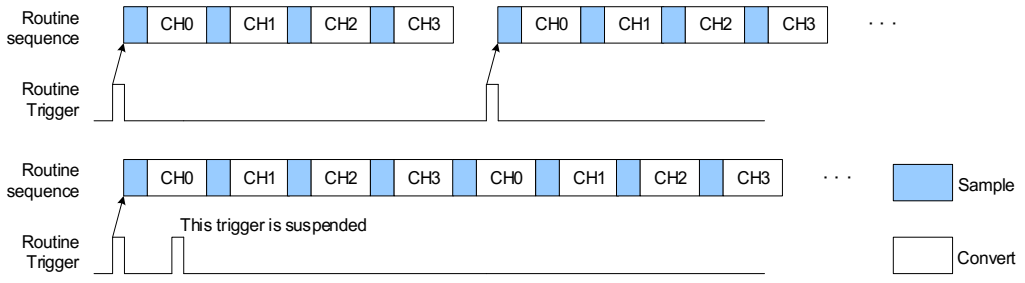
ETSRC[3:0]	Trigger source	Trigger type
0000	TIMER0_CH0	Internal on-chip signal
0001	TIMER0_CH1	
0010	TIMER0_CH2	
0011	TIMER1_CH1	
0100	TIMER2_TRGO	
0101	TIMER14_CH0	
0110	EXTI_11	External signal
0111	SWRCST	Software trigger
1000	CLA0_TRGO	Internal on-chip signal
1001	CLA2_TRGO	

Table 13-4. External trigger for inserted channels of ADC

ETSIC[3:0]	Trigger source	Trigger type
0000	TIMER0_TRGO	Internal on-chip signal
0001	TIMER0_CH3	
0010	TIMER1_TRGO	
0011	TIMER0_CH0	
0100	TIMER2_CH3	
0101	TIMER14_TRGO	
0110	EXTI_15	External signal
0111	SWICST	Software trigger
1000	CLA1_TRGO	Internal on-chip signal
1001	CLA3_TRGO	

During the routine sequence conversion, if a new routine sequence trigger request occurs, the trigger request will be suspended and executed after the current conversion is completed, as shown in *Figure 13-16. Routine trigger occurs during routine conversion.*

Figure 13-16. Routine trigger occurs during routine conversion



During the inserted sequence conversion, if new inserted sequence or routine sequence trigger requests occur, these trigger requests will also be suspended and executed after the

current conversion is completed. When both inserted sequence and routine sequence trigger requests are suspended, the ADC will prioritize the inserted sequence conversion. After the inserted sequence conversion is completed, the routine sequence conversion will then be executed.

Each sequence (inserted sequence or routine sequence) can have at most one suspended trigger request. Any subsequent trigger requests will be ignored.

13.4.7. DMA request

The DMA request, which is enabled by the RDMA bit of ADC_CTL1 or IDMA bit of ADC_CTL2 register, is used to transfer data of routine or inserted sequence for conversion of more than one channel. The ADC generates a DMA request at the end of conversion of a routine or inserted channel. When this request is received, the DMA will transfer the converted data from the ADC_RDATA or ADC_IDATA register to the destination location which is specified by the user.

13.4.8. ADC internal channels

When the HPTSEN bit in ADC_CTL1 register is set, the high-precision temperature sensor channel (ADC_IN16) is 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} (please refer to the datasheet). When this sensor is not in use, it can be set in power down mode by resetting the HPTSEN bit.

The output voltage of the temperature sensor changes linearly with temperature. Due to the diversification of production process, the deviation of the temperature change curve is different on chip to chip (refer to the device datasheet for more information).

To use the high- precison temperature sensor:

1. Configure the ADC clock (not greater than 15MHz).
2. Configure the conversion sequence and the sampling time (t_{s_temp}) for the channel.
3. Enable the temperature sensor by setting the HPTSEN bit in the ADC control register (ADC_CTL1).
4. Start the ADC conversion by setting the ADCON bit (or by trigger).
5. Read the resulting temperature data($V_{temperature}$) in the ADC data register, and get the temperature using the following formula:

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

$V_{temperature}$: The output voltage of temperature sensor.

V_{25}: The output voltage of temperature sensor at 25°C. The ADC conversion results of the temperature sensor corresponding to 25°C were calibrated during chip production. The factory-calibrated offset value is stored in the read-only area of flash, and refers to electrical

characteristics section of device datasheet.

Avg_Slope: Average slope for curve between temperatures vs. the output voltage of temperature sensor, refers to electrical characteristics section of device datasheet.

Note:

1) After the temperature sensor is enabled, it is necessary to wait for at least 3 ADC sampling cycles before the ADC conversion code value is considered valid, and the first 3 conversion data should be discarded;

2) The sampling accuracy of temperature sensor can be improved by means of hardware on-chip over-sampling or software averaging.

When the VDDADIV2EN bit in ADC_CTL1 register is set, the $V_{DDADIV2}$ channel (ADC_IN18) is enabled. The $V_{DDADIV2}$ channel is internally connected to the analog supply voltage $V_{DDA}/2$.

When the INREFEN bit in ADC_CTL1 register is set, the V_{REFINT} channel (ADC_IN17) is enabled. The internal reference voltage (V_{REFINT}) provides a stable (bandgap) voltage output for the ADC and comparators. V_{REFINT} is internally connected to the ADC input channel.

13.4.9. Programmable resolution (DRES)

The resolution is configured by programming the DRES[1:0] bits in the ADC_CTL0 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 13-5. tCONV timings depending on resolution*.

Table 13-5. t_{CONV} timings depending on resolution

DRES[1:0] bits	t_{CONV} (ADC clock cycles)	t_{CONV}(ns) at fADC=28MHz	t_{SMPL}(min) (ADC clock cycles)	t_{ADC}(ADC clock cycles)	t_{ADC}(ns) at f_{ADC}=28MHz
12	12.5	446ns	1.5	14	500ns
10	10.5	375ns	1.5	12	429ns
8	8.5	304ns	1.5	10	357ns
6	6.5	232ns	1.5	8	286ns

13.4.10. On-chip hardware oversampling

The on-chip hardwareoversampling unit, which is enabled by OVSEN bit in the ADC_OVSAMPCTL register, provides higher data resolution at the cost of lower output data rate.

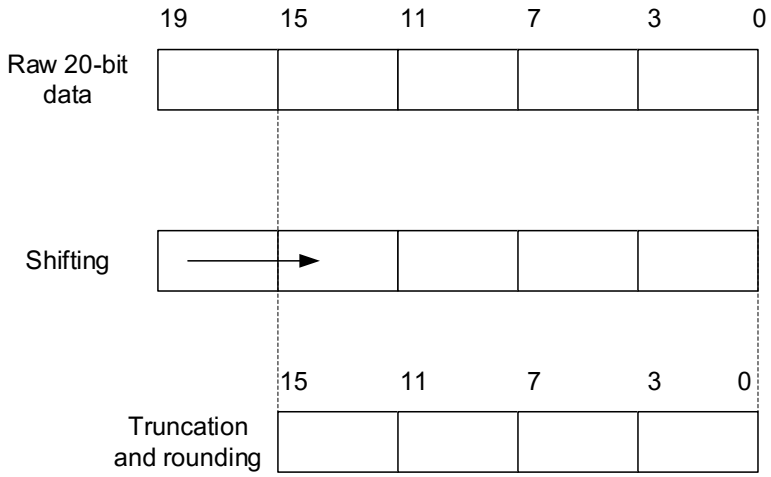
It provides a result with the following form, where N and M can be adjusted, and Dout(n) is the n-th output digital signal of the ADC:

$$\text{Result} = \frac{1}{M} * \sum_{n=0}^{n=N-1} D_{OUT}(n) \quad (13\text{-}1)$$

The on-chip hardware oversampling circuit performs the following functions: summing and bit right shifting. The oversampling ratio N is defined using the OVSR[2:0] bits in the ADC_OVSAMPCTL register. It can range from 2x to 256x. The division coefficient M consists of a right bit shift 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 13-17. 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 13-18. 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 13-18. A numerical example with 5-bit shifting and rounding

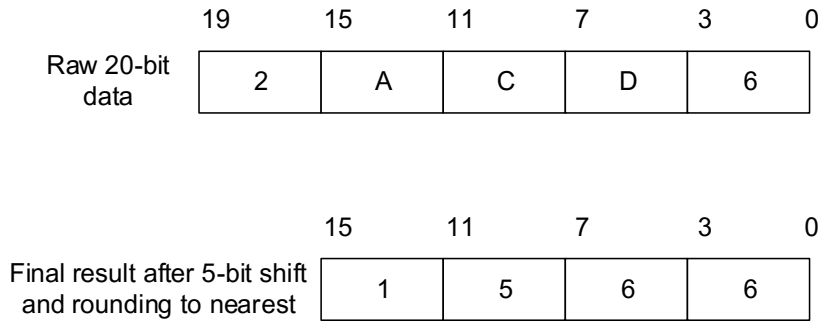


Table 13-6. Maximum output results for N and M combimations (grayed values indicates truncation) below gives the data format for the various N and M combinations, and the raw conversion data equals 0xFFF.

Table 13-6. Maximum output results for N and M combimations (grayed values indicates 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 remains equal throughout the oversampling sequence. New data is supplied every N conversions, and the equivalent delay is equal to:

$$N \times t_{ADC} = N \times (t_{SMPL} + t_{CONV}) \quad (13\text{-}2)$$

Oversampling work with ADC modes

Most of the ADC work modes are available when oversampling is enabled.

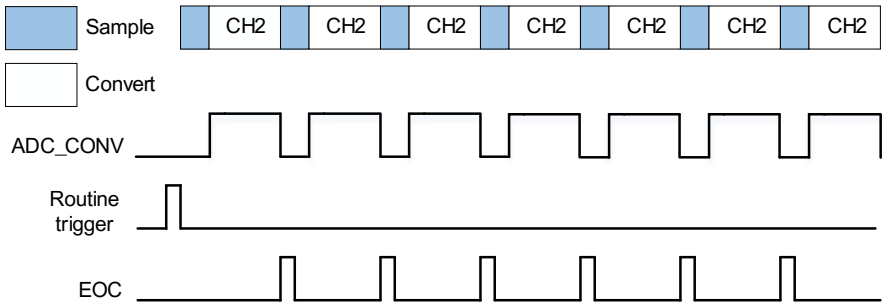
- Routine channels.
- ADC started by software or external triggers.
- Single or scan, continuous or discontinuous operation modes.
- Programmable sample time.
- Analog watchdog.

The oversampling configuration can only be changed when ADCON is reset. Make sure configuring the oversampling before setting ADCON to 1.

13.4.11. ADC convertsion signal

The ADC conversion signal ADC_CONV is held high level during ADC channel conversion and low level at other time. The ADC_CONV signal can be internally connected to the CLAx module as an input to the CLAx multiplexer.

Figure 13-19. ADC convertsion signal in continuous operation mode



13.4.12. ADC interrupts

The interrupt can be produced on one of the events:

- End of conversion for routine sequence.
- The analog watchdog event (the analog watchdog status bit is set).

Separate interrupt enable bits are available for flexibility.

13.5. Register definition

ADC base address: 0x4001 2400

13.5.1. Status register (ADC_STAT)

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											STRC	STIC	EOIC	EOC	WDE
											rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

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	STIC	Start flag of inserted sequence 0: Inserted sequence conversion is not started 1: Inserted sequence conversion is started Set by hardware when inserted sequence conversion starts. Cleared by software writing 0 to it.
2	EOIC	End flag of inserted sequence conversion 0: No end of inserted sequence conversion 1: End of inserted sequence conversion Set by hardware at the end of inserted sequence conversion. Cleared by software writing 0 to it.
1	EOC	End flag of routine or inserted sequence conversion 0: No end of routine sequence conversion 1: End of routine sequence conversion Set by hardware at the end of a routine or inserted sequence conversion. Cleared by software writing 0 to it or by reading the ADC_RDATA register.
0	WDE	Analog watchdog event flag 0: No analog watchdog event

1: Analog watchdog event

Set by hardware when the converted voltage crosses the values programmed in the ADC_WDLT and ADC_WDHT registers.

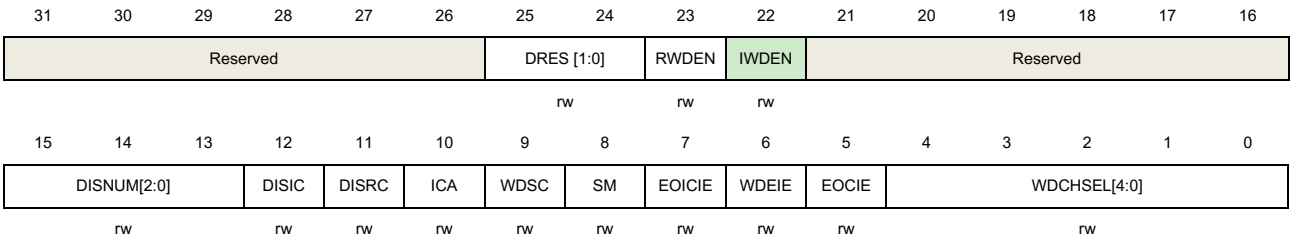
Cleared by software writing 0 to it.

13.5.2. Control register 0 (ADC_CTL0)

Address offset: 0x04

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value
25:24	DRES[1:0]	ADC resolution 00: 12bit 01: 10bit 10: 8bit 11: 6bit
23	RWDEN	Routine sequence analog watchdog enable 0: Routine sequence analog watchdog disable 1: Routine sequence analog watchdog enable
22	IWDEN	Inserted sequence analog watchdog enable 0: Inserted sequence analog watchdog disable 1: Inserted sequence analog watchdog enable
21:16	Reserved	Must be kept at reset value
15:13	DISNUM[2:0]	Number of conversions in discontinuous mode The number of channels to be converted after a trigger will be DISNUM[2:0] +1
12	DISIC	Discontinuous mode on inserted sequence 0: Discontinuous operation mode on inserted sequence disable 1: Discontinuous operation mode on inserted sequence enable
11	DISRC	Discontinuous mode on routine sequence 0: Discontinuous operation mode on routine sequence disable

		1: Discontinuous operation mode on routine sequence enable
10	ICA	Inserted sequence convert automatically 0: Inserted sequence convert automatically disable 1: Inserted sequence convert automatically enable
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	EOICIE	Interrupt enable for EOIC 0: EOIC interrupt disable 1: EOIC interrupt enable
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 channel 0 00001: ADC channel 1 00010: ADC channel 2 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 channel 15 1x000: ADC channel 16 1x001: ADC channel 17 1x010: ADC channel 18 1x011: ADC channel 19

1x100: ADC channel 20

1x101: ADC channel 21

1x11x: ADC channel 22

Other values are reserved.

Note: ADC analog inputs Channel 16, Channel 17 and Channel 18 are internally connected to the high precision temperature sensor, V_{REFINT} analog inputs and $V_{DDADIV2}$ input. Channel 19 ~ channel 22 are external PIN input.

13.5.3. 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			HPTSENSEL	VDDADIV2EN	ETSRC[3]	ETSIC[3]	HPTSEN	TSVREN	SWRCST	SWICST	ETMRC	ETSRC[2:0]			Reserved
			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
ETMIC	ETSIC [2:0]			DAL	Reserved		RDMA	Reserved				RSTCLB	CLB	CTN	ADCON
rw	rw			rw			rw					rw	rw	rw	rw

Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value
28	HPTSENSEL	High precision temperature sensor enable select. 0: The high precision temperature sensor enable through the TSVREN bit 1: The high precision temperature sensor enable through the HPTSEN bit
27	VDDADIV2EN	Channel 18 (analog supply voltage VDDA/2) enable of ADC. 0: Channel 18 of ADC disable 1: Channel 18 of ADC enable
26	ETSRC[3]	External trigger select for routine sequence. Refer to ETSRC[2:0]
25	ETSIC[3]	External trigger select for inserted sequence. Refer to ETSIC[2:0]
24	HPTSEN	Channel 16 (high precision temperature sensor) enable of ADC. 0: Channel 16 of ADC disable 1: Channel 16 of ADC enable
23	TSVREN	Channel 16 (high precision temperature sensor) and channel 17 (internal reference voltage) enable of ADC. Only channel 17 enable of ADC when HPTSENSEL = 1. 0: Channel 16 and channel 17 of ADC disable 1: Channel 16 and channel 17 of ADC enable
22	SWRCST	Software start conversion of routine sequence

		Set 1 on this bit starts the 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	SWICST	Software start conversion of inserted sequence Set 1 on this bit starts the conversion of a inserted sequence if ETSIC is 111. It is set by software and cleared by software or by hardware immediately after the conversion starts.
20	ETMRC	External trigger mode 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 These bits and the ETSRC[3] bit are written by software to define the external trigger select for routine sequence. 0000: TIMER0 CH0 0001: TIMER0 CH1 0010: TIMER0 CH2 0011: TIMER1 CH1 0100: TIMER2 TRGO 0101: TIMER14 CH0 0110: EXTI line 11 0111: SWRCST 1000: CLA0 TRGO 1001: CLA2 TRGO 1010~1111: reserved
16	Reserved	Must be kept at reset value
15	ETMIC	External trigger mode for inserted sequence 0: External trigger for inserted sequence disable 1: External trigger for inserted sequence enable
14:12	ETSIC[2:0]	External trigger select for inserted channel These bits and the ETSIC [3] bit are written by software to define the external trigger select for inserted sequence. 0000: TIMER0 TRGO 0001: TIMER0 CH3 0010: TIMER1 TRGO 0011: TIMER1 CH0 0100: TIMER2 CH3 0101: TIMER14 TRGO 0110: EXTI line15 0111: SWICST 1000: CLA1 TRGO

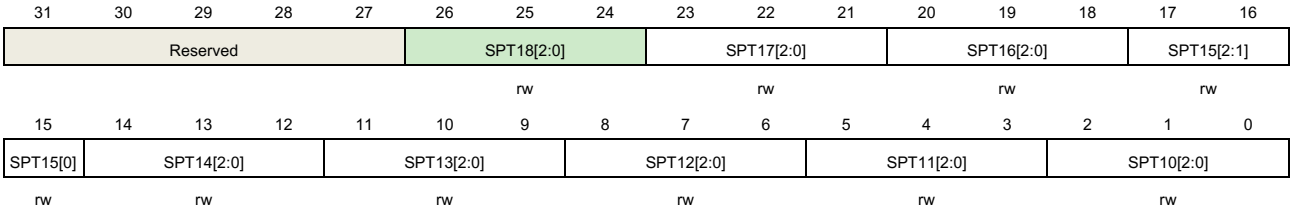
		1001: CLA3 TRGO 1010~1111: reserved
11	DAL	Data alignment 0: LSB alignment 1: MSB alignment
10:9	Reserved	Must be kept at reset value
8	RDMA	DMA request enable for routine sequence 0: DMA request disable 1: DMA request enable
7:4	Reserved	Must be kept at reset value
3	RSTCLB	Reset calibration This bit is set by software and cleared by hardware after the calibration registers are initialized. 0: Calibration register initialization done. 1: Calibration register initialization starts
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 waked 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 routine sequence conversion will start. 0: ADC disable and power down 1: ADC enable

13.5.4. Sample time register 0 (ADC_SAMPT0)

Address offset: 0x0C

Reset value: 0x0000 0000

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



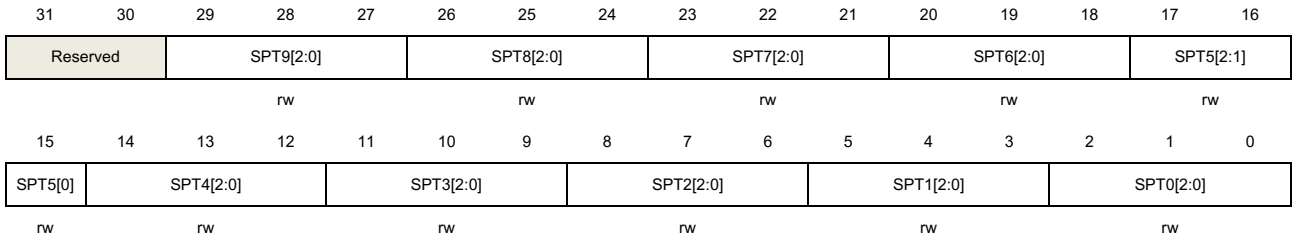
Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value
26:24	SPT18[2:0]	Refer to SPT10[2:0] description
23:21	SPT17[2:0]	Refer to SPT10[2:0] description
20:18	SPT16[2:0]	Refer to SPT10[2:0] description
17:15	SPT15[2:0]	Refer to SPT10[2:0] description
14:12	SPT14[2:0]	Refer to SPT10[2:0] 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

13.5.5. Sample time register 1 (ADC_SAMPT1)

Address offset: 0x10

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value
29:27	SPT9[2:0]	Refer to SPT0[2:0] description

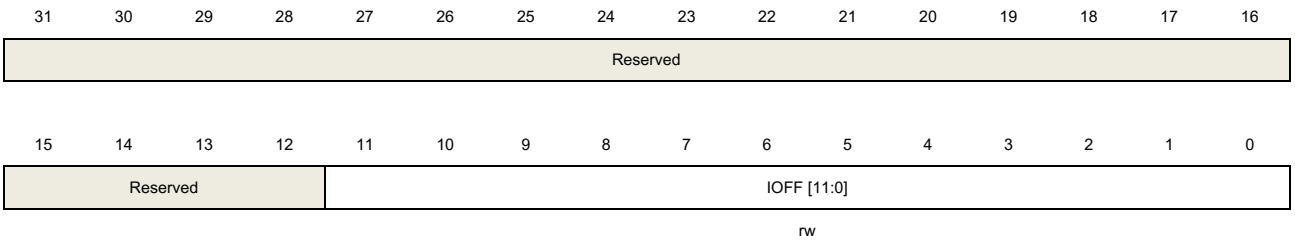
26:24	SPT8[2:0]	Refer to SPT0[2:0] description
23:21	SPT7[2:0]	Refer to SPT0[2:0] description
20:18	SPT6[2:0]	Refer to SPT0[2:0] description
17:15	SPT5[2:0]	Refer to SPT0[2:0] description
14:12	SPT4[2:0]	Refer to SPT0[2:0] description
11:9	SPT3[2:0]	Refer to SPT0[2:0] description
8:6	SPT2[2:0]	Refer 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

13.5.6. Inserted channel data offset register x (ADC_IOFFx) (x=0..3)

Address offset: 0x14 + 0x04 * x (x=0..3)

Reset value: 0x0000 0000

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



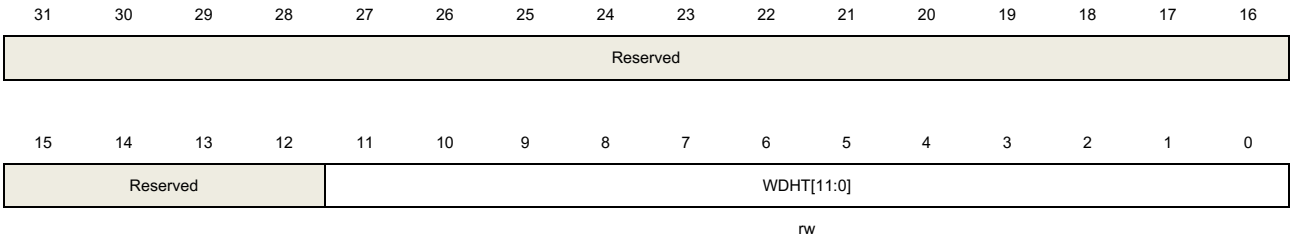
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	IOFF[11:0]	Data offset for inserted channel x. These bits will be subtracted from the inserted channel converted data, and the offset result are saved in the ADC_IDATA register.

13.5.7. Watchdog high threshold register (ADC_WDHT)

Address offset: 0x24

Reset value: 0x0000 0FFF

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



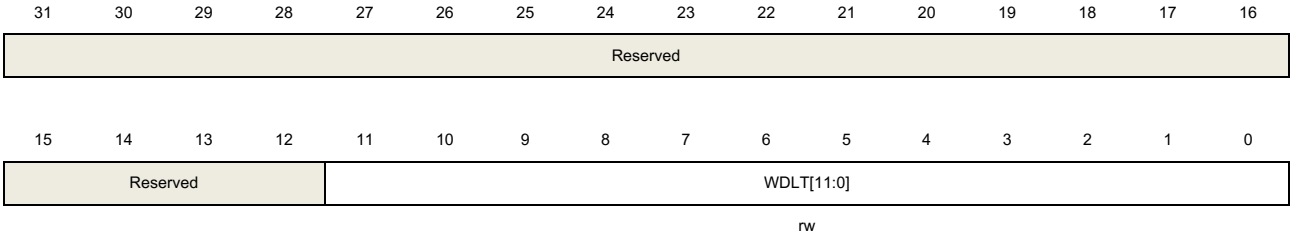
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value
11:0	WDHT[11:0]	High threshold for analog watchdog These bits define the high threshold for the analog watchdog.

13.5.8. Watchdog low threshold register (ADC_WDLT)

Address offset: 0x28

Reset value: 0x0000 0000

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



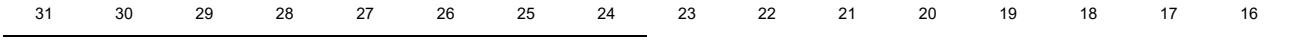
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value
11:0	WDLT[11:0]	Low threshold for analog watchdog These bits define the low threshold for the analog watchdog.

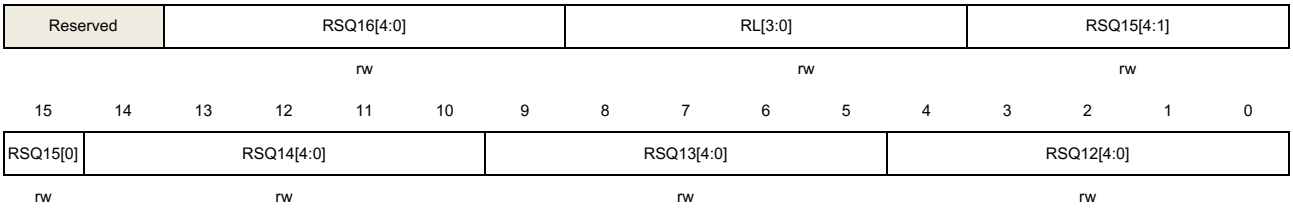
13.5.9. Routine sequence register 0 (ADC_RSQ0)

Address offset: 0x2C

Reset value: 0x0000 0000

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





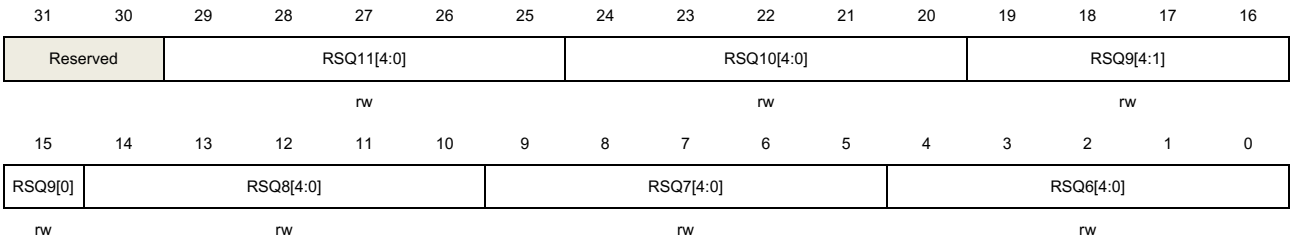
Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value
29:25	RSQ16[4:0]	Refer to RSQ0[4:0] description
24:20	RL[4:0]	Routine sequence length The total number of conversion in routine sequence equals to RL[4:0] +1.
19:15	RSQ15[4:0]	Refer to RSQ0[4:0] description
14:10	RSQ14[4:0]	Refer to 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

13.5.10. Routine sequence register 1 (ADC_RSQ1)

Address offset: 0x30

Reset value: 0x0000 0000

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

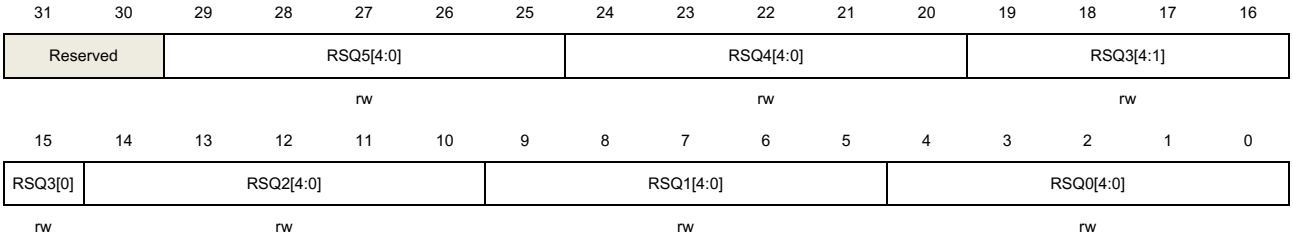


Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value
29:25	RSQ11[4:0]	Refer to RSQ0[4:0] description
24:20	RSQ10[4:0]	Refer to RSQ0[4:0] description
19:15	RSQ9[4:0]	Refer to RSQ0[4:0] description
14:10	RSQ8[4:0]	Refer to RSQ0[4:0] description
9:5	RSQ7[4:0]	Refer to RSQ0[4:0] description
4:0	RSQ6[4:0]	Refer to RSQ0[4:0] description

13.5.11. Routine sequence register 2 (ADC_RSQ2)

Address offset: 0x34
Reset value: 0x0000 0000

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

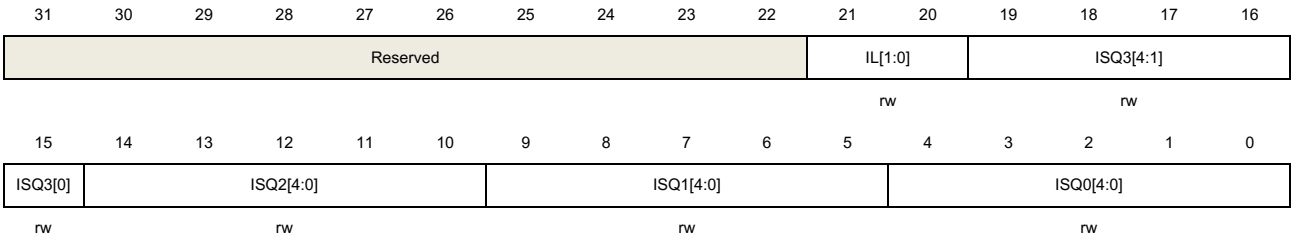


Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value
29:25	RSQ5[4:0]	Refer to RSQ0[4:0] description
24:20	RSQ4[4:0]	Refer to RSQ0[4:0] description
19:15	RSQ3[4:0]	Refer to RSQ0[4:0] description
14:10	RSQ2[4:0]	Refer to RSQ0[4:0] description
9:5	RSQ1[4:0]	Refer to RSQ0[4:0] description
4:0	RSQ0[4:0]	The channel number (0..22) is written to these bits to select a channel as the nth conversion in the routine sequence.

13.5.12. Inserted sequence register (ADC_ISQ)

Address offset: 0x38
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.
21:20	IL[1:0]	Inserted channel length.

		The total number of conversion in inserted sequence equals IL[1:0] + 1.
19:15	ISQ3[4:0]	Refer to ISQ0[4:0] description
14:10	ISQ2[4:0]	Refer to ISQ0[4:0] description
9:5	ISQ1[4:0]	Refer to ISQ0[4:0] description
4:0	ISQ0[4:0]	The channel number (0..22) is written to these bits to select a channel as the nth conversion in the inserted sequence. Unlike the routine conversion sequence, the inserted channels are converted starting from (4-IL [1:0]-1), if IL [1:0] length is less than 4. IL — Insert channel order 11 — ISQ0>>ISQ1>>ISQ2>>ISQ3 10 — ISQ1>>ISQ2>>ISQ3 01 — ISQ2>>ISQ3 00 — ISQ3

13.5.13. Latch data register x (ADC_LDATAx) (x= 0..3)

Address offset: 0x3C + 0x04 * x (x=0..3)
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
LDATAn [15:0]															
r															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	LDATAn[15:0]	Inserted or routine number n conversion data. These bits contain the number n conversion result, which is read only.

13.5.14. 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
Reserved															

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	Reserved	Must be kept at reset value
15:0	RDATA[15:0]	Routine channel data These bits contain the conversion result from routine channel, which is read only.

13.5.15. Oversampling control register (ADC_OVSAMPCTL)

Address offset: 0x80

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						TOVS	OVSS[3:0]				OVSR[2:0]			Reserved	OVSEN
						rw	rw				rw				rw

Bits	Fields	Descriptions
31:10	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

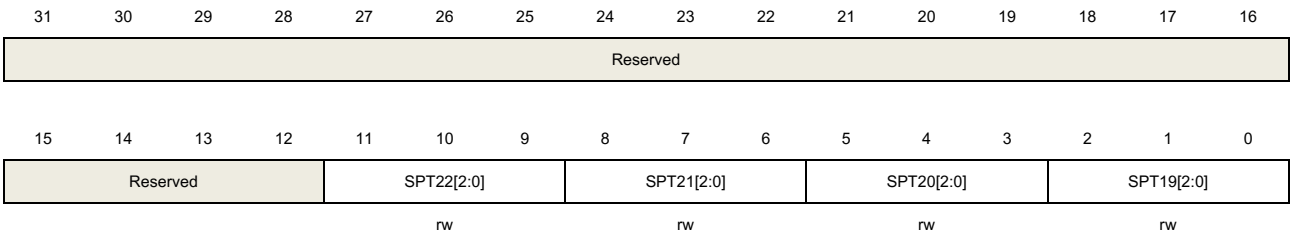
		Other: reserved **Note:** The software allows this bit to be written only when ADCON = 0 (this ensures that no conversion is in progress).
4:2	OVSR[2:0]	Oversampling ratio This bit filed defines the number of oversampling ratio. 000: 2x 001: 4x 010: 8x 011: 16x 100: 32x 101: 64x 110: 128x 111: 256x **Note:** 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	Oversampling enable This bit is set and cleared by software. 0: Oversampling disabled 1: Oversampling enabled **Note:** The software allows this bit to be written only when ADCON = 0 (this ensures that no conversion is in progress).

13.5.16. Sample time register 2 (ADC_SAMPT2)

Address offset: 0x84

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
11:9	SPT22[2:0]	Refer to SPT19[2:0] description
8:6	SPT21[2:0]	Refer to SPT19[2:0] description

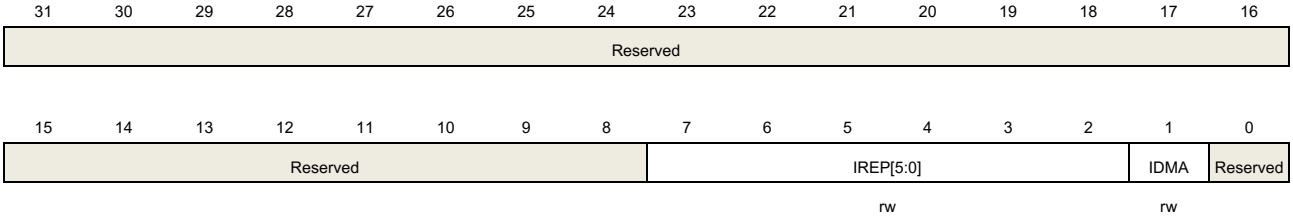
5:3	SPT20[2:0]	Refer to SPT19[2:0] description
2:0	SPT19[2:0]	Channel sampling time 000: channel sampling time is 1.5 cycles 001: channel sampling time is 7.5 cycles 010: channel sampling time is 13.5 cycles 011: channel sampling time is 28.5 cycles 100: channel sampling time is 41.5 cycles 101: channel sampling time is 55.5 cycles 110: channel sampling time is 71.5 cycles 111: channel sampling time is 239.5 cycles

13.5.17. Control register 2 (ADC_CTL2)

Address offset: 0x88

Reset value: 0x0000 0000

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



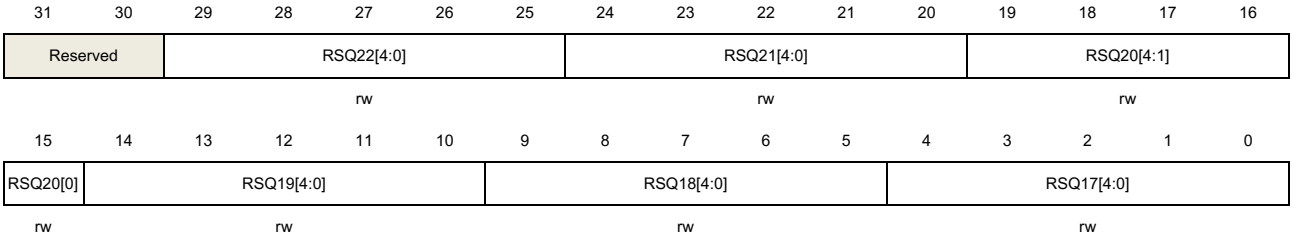
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value
7:2	IREP [5:0]	Inserted sequence repeat times. These bits control the conversion times of the inserted sequence. The total coversion times is IREP [5:0] +1. **Note:** When discontinuous mode on inserted sequence is enabled, these bits must keep reset value. 000000: 1 time. 000001: 2 times. …. 111111: 64 times.
1	IDMA	DMA request enable for inserted sequence 0: DMA request disable 1: DMA request enable
0	Reserved	Must be kept at reset value

13.5.18. Routine sequence register 3 (ADC_RSQ3)

Address offset: 0x8C

Reset value: 0x0000 0000

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



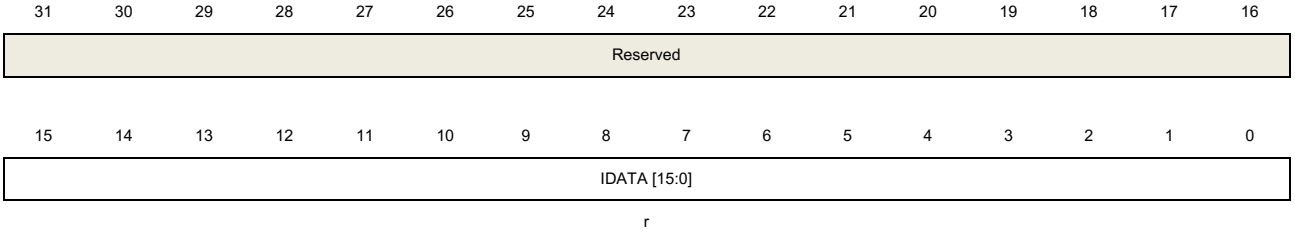
Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value
29:25	RSQ22[4:0]	Refer to RSQ0[4:0] description
24:20	RSQ21[4:0]	Refer to RSQ0[4:0] description
19:15	RSQ20[4:0]	Refer to RSQ0[4:0] description
14:10	RSQ19[4:0]	Refer to RSQ0[4:0] description
9:5	RSQ18[4:0]	Refer to RSQ0[4:0] description
4:0	RSQ17[4:0]	Refer to RSQ0[4:0] description

13.5.19. Inserted data register (ADC_IDATA)

Address offset: 0x90

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	IDATA[15:0]	Inserted channel data These bits contain the inserted channel conversion value, which is read only.

13.5.20. Latch data control register(ADC_LDCTL)

Address offset: 0x94

Reset value: 0x0001 0203

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
SEQSEL0	Reserved		COVSEL0[4:0]					SEQSEL1	Reserved		COVSEL1[4:0]				
rw			rw					rw			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEQSEL2	Reserved		COVSEL2[4:0]					SEQSEL3	Reserved		COVSEL3[4:0]				
rw			rw					rw			rw				

Bits	Fields	Descriptions
31	SEQSEL0	The ADC_LDATA0 register sequence source selection 0: select inserted sequence 1: select routine sequence **Note:** The software allows this bit to be written only when ADCON=0 (this ensures that no conversion is progress).
30:29	Reserved	Must be kept at reset value.
28:24	COVSEL0[4:0]	The ADC_LDATA0 register conversion source selection When ADC_LDATA0 register is used for sequence conversion, the nth conversion data of the inserted sequence (SEQSEL0 = 0) or the routine sequence (SEQSEL0 = 1) can be selected to be stored in ADC_LDATA0 register. 00000: The 0th conversion of the sequence. 00001: The 1st conversion of the sequence. 00010: The 2nd conversion of the sequence. … 10110: the 22th conversion of the sequence. Other values are reserved. **Note:** When used in inserted sequence (SEQSEL0 = 0), it can only be 00000,00001,00010,00011, and the COVSELn[4:0] default value is n. The software allows this bit to be written only when ADCON=0 (this ensures that no conversion is progress).
23	SEQSEL1	Refer to SEQSEL0 description.
22:21	Reserved	Must be kept at reset value.
20:16	COVSEL1[4:0]	Refer to COVSEL0[4:0] description.

15	SEQSEL2	Refer to SEQSEL0 description.
14:13	Reserved	Must be kept at reset value.
12:8	COVSEL2[4:0]	Refer to COVSEL0[4:0] description.
7	SEQSEL3	Refer to SEQSEL0 description.
6:5	Reserved	Must be kept at reset value.
4:0	COVSEL3[4:0]	Refer to COVSEL0[4:0] description.

14. Digital-to-analog converter (DAC)

14.1. Overview

The Digital-to-analog converter converts 12-bit digital data to a voltage on the external pins. The digital data can be configured in 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.

14.2. 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 trigged 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.
- Output can be configured to persist when system reset.

Figure 14-1. DAC block diagram shows the block diagram of DAC and *Table 14-1. DAC I/O description***Table 14-1. DAC I/O** gives the pin description.

Figure 14-1. DAC block diagram

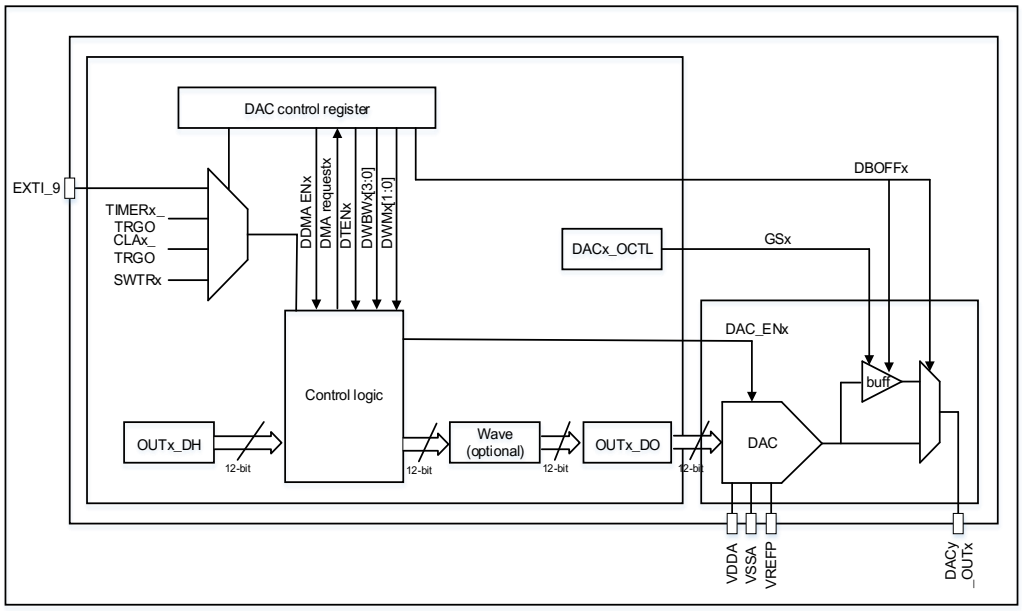


Table 14-1. DAC I/O description

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}	Postive 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 14-2. DAC triggers and outputs summary

	DAC0		DAC1	
Channel	**Channel0**	**Channel1**	**Channel0**	**Channel1**
DAC outputs connected to I / Os	PA4	PA5	PA6	PA7
DAC output buffer	•		•	
DAC software trigger	•			
DAC trigger signals from EXTI	EXTI_9			
DAC trigger signals from TIMER	TIMER0_TRGO TIMER1_TRGO TIMER2_TRGO TIMER5_TRGO TIMER6_TRGO TIMER14_TRGO			
DAC trigger	CLA0_TRGO			

	DAC0	DAC1
signals from CLA	CLA1_TRGO CLA2_TRGO CLA3_TRGO	

Note: The GPIO pins should be configured to analog mode before enable the DAC module.

14.3. Function overview

14.3.1. DAC enable

The DAC can be powered on by setting the DENx bit in the DAC_CTL0 register. A t_{WAKEUP} time is needed to startup the analog DAC submodule.

14.3.2. 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.

14.3.3. 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 fored to 4'b0000.

14.3.4. DAC trigger

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 14-3. External triggers of DAC***.

Table 14-3. External triggers of DAC

DTSELx[3:0]	Trigger Source	Trigger Type
4b'0000	TIMER5_TRGO	Hardware trigger
4b'0001	TIMER14_TRGO	
4b'0010	TIMER6_TRGO	
4b'0011	TIMER2_TRGO	
4b'0100	TIMER1_TRGO	
4b'0101	TIMER0_TRGO	
4b'0110	EXTI_9	

DTSELx[3:0]	Trigger Source	Trigger Type
4b'0111	SWTRIG	Software trigger
4b'1000	CLA0_TRGO	Hardware trigger
4b'1001	CLA1_TRGO	
4b'1010	CLA2_TRGO	
4b'1011	CLA3_TRGO	

The TIMERx_TRGO signals are generated from the timers, the CLAx_TRGO signals are generated from the CLAs, while the software trigger can be generated by setting the SWTRx bits in the DAC_SWT register.

14.3.5. 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.

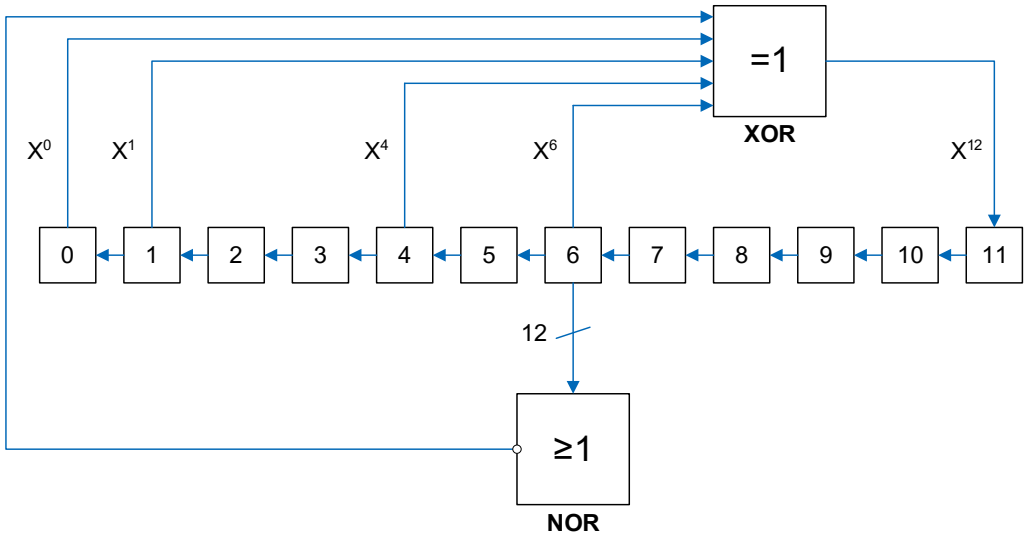
When the DAC holding data (OUTx_DH) is loaded into the DAC_OUTx_DO register, after the time $t_{SETTLING}$ which is determined by the analog output load and the power supply voltage, the analog output is valid.

14.3.6. 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.

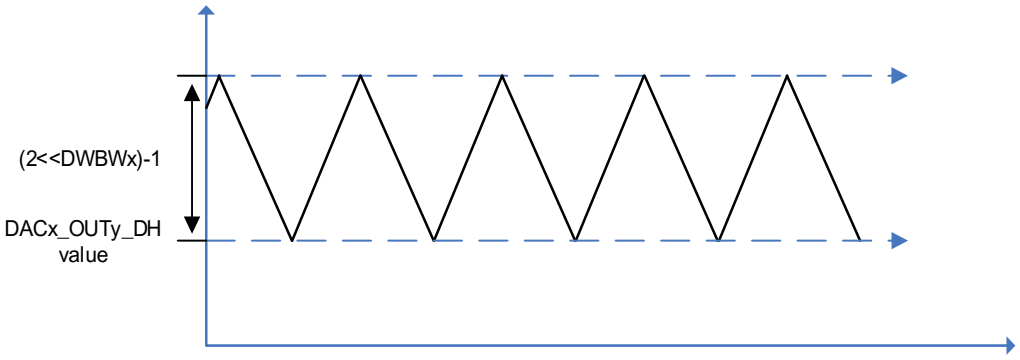
LFSR noise wave mode: there is a Linear Feedback Shift Register (LFSR) in the DAC control logic, it controls the LFSR noise signal which is added to the OUTx_DH value, and then the result is stored into the DAC_OUTx_DO register. When the configured DAC noise wave bit width is less than 12, the noise signal equals to the LSB DWBWx bits of the LFSR register, while the MSB bits are masked.

Figure 14-2. 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 14-3. DAC triangle noise wave



14.3.7. DAC output voltage

When GSx bit in DAC_OCTL register is not set or buffer off, the analog output voltages on the DAC pin are determined by the following equation:

$$V_{DAC_OUT}=V_{REFP}*OUTx_DO/4096 \quad (14\text{-}1)$$

The digital input is linearly converted to an analog output voltage and its range is 0 to V_{REFP}.

When GSx bit is set and buffer on, the analog output voltages on the DAC pin are determined by the following equation:

$$V_{DAC_OUT}=V_{REFP}*OUTx_DO/2048 \quad (14\text{-}2)$$

The digital input is linearly converted to an analog output voltage and its range is 0 to V_{DDA}.

14.3.8. 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.

If the second external trigger arrives before confirming the previous request, the new request will not be serviced, and an underrun error event occurs. The DDUDRx bit in the DAC_STAT0 register is set, an interrupt will be generated if the DDUDRIEx bit in the DAC_CTL0 register is set. The DMA request will be stalled until the DDUDRx bit is cleared.

In DAC concurrent conversion mode, only one of the DDMAENx bits should be set, so only one of the DDUDRIEx bits is used.

14.3.9. 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.

14.3.10. DAC reset persist mode

The reset persist mode can be enabled by setting the DRSTMDx bit in the DAC_CTL0 register. Output channel will persist through all resets except for power-on-reset.

When DRSTMDx is set to 1, the register DAC_CTL0 (bit DRSTMDx, DTENx, DBOFFx and DENx), OUTx_DH, OUTx_DO and DAC_OCTL can only be reset by power on reset.

14.4. Register definition

DAC0 base address: 0x4000 7400

DAC1 base address: 0x4000 CC00

14.4.1. 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
DRSTMD1	DTSEL1[3]	DDUDRIE1	DDMAEN1	DWBW1[3:0]				DWM1[1:0]		DTSEL1[2:0]			DTEN1	DBOFF1	DEN1
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
DRSTMD0	DTSEL0[3]	DDUDRIE0	DDMAEN0	DWBW0[3:0]				DWM0[1:0]		DTSEL0[2:0]			DTEN0	DBOFF0	DEN0
rw	rw	rw	rw	rw				rw		rw			rw	rw	rw

Bits	Fields	Descriptions
31	DRSTMD1	DACx_OUT1 reset mode 0: Normal mode. All resets will reset DACx_OUT1 and its associated registers. 1: Reset persist mode. DACx_OUT1 output will persist through all resets except for power-on-resets.
30	DTSEL1[3]	DACx_OUT1 trigger selection bit[3], refer to DTSEL1[2:0]
29	DDUDRIE1	DACx_OUT1 DMA Underrun Interrupt enable 0: DACx_OUT1 DMA underrun Interrupt disabled 1: DACx_OUT1 DMA underrun Interrupt enabled
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. 0000: TIMER5 TRGO 0001: TIMER14 TRGO 0010: TIMER6 TRGO 0011: TIMER2 TRGO 0100: TIMER1 TRGO 0101: TIMER0 TRGO 0110: EXTI line 9 0111: Software trigger 1000: CLA0_TRGO 1001: CLA1_TRGO 1010: CLA2_TRGO 1011: CLA3_TRGO 11xx: reserved
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	DRSTMD0	DACx_OUT0 reset mode

		0: Normal mode. All resets will reset DACx_OUT0 and its associated registers. 1: Reset persist mode. DACx_OUT0 output will persist through all resets except for power-on-resets.
14	DTSEL0[3]	DACx_OUT0 trigger selection bit 3, refer to DTSEL0[2:0]
13	DDUDRIE0	DACx_OUT0 DMA Underrun Interrupt enable 0: DACx_OUT0 DMA underrun Interrupt disabled 1: DACx_OUT0 DMA underrun Interrupt enabled
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. 0000: TIMER5 TRGO 0001: TIMER14 TRGO 0010: TIMER6 TRGO 0011: TIMER2 TRGO 0100: TIMER1 TRGO

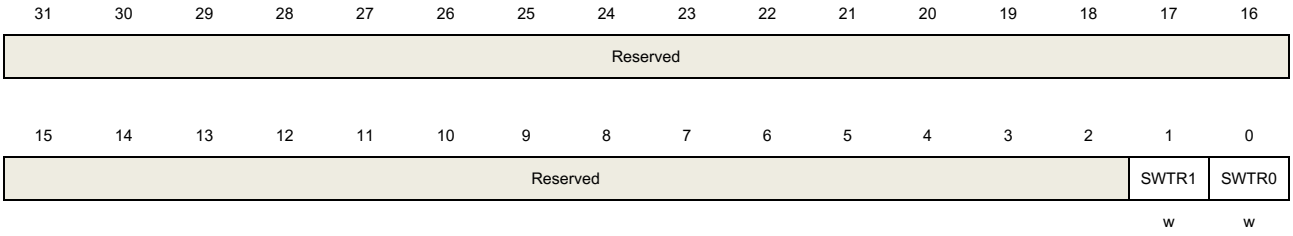
		0101: TIMER0 TRGO 0110: EXTI line 9 0111: Software trigger 1000: CLA0_TRGO 1001: CLA1_TRGO 1010: CLA2_TRGO 1011: CLA3_TRGO 11xx: reserved
2	DTEN0	DACx_OUT0 trigger enable 0: DACx_OUT0 trigger disabled 1: DACx_OUT0 trigger enabled
1	DBOFF0	DACx_OUT0 output buffer turn off 0: DACx_OUT0 output buffer turns 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

14.4.2. DACx software trigger register (DAC_SWT)

Address offset: 0x04

Reset value: 0x0000 0000

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



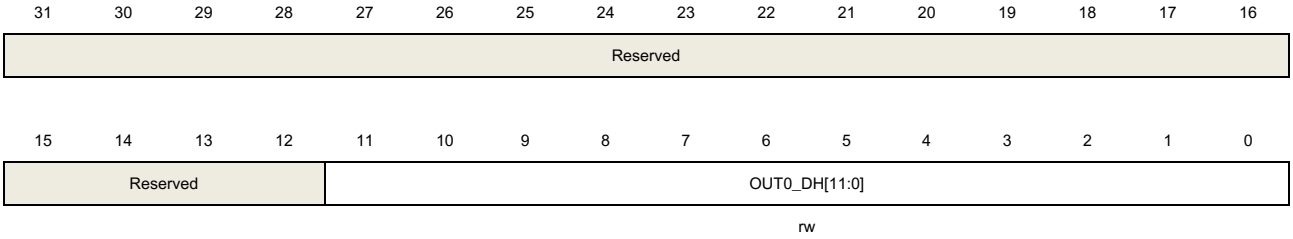
Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	SWTR1	DACx_OUT1 software trigger, cleared by hardware 0: Software trigger disabled 1: Software trigger enabled
0	SWTR0	DACx_OUT0 software trigger, cleared by hardware 0: Software trigger disabled 1: Software trigger enabled

14.4.3. 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)



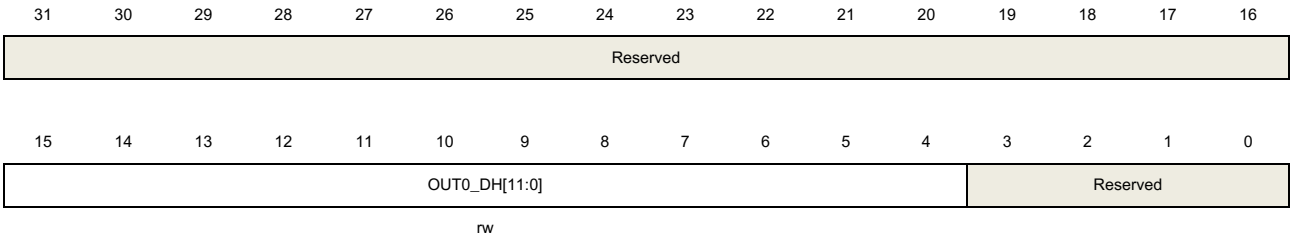
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	OUT0_DH[11:0]	DACx_OUT0 12-bit right-aligned data These bits specify the data that is to be converted by DACx_OUT0.

14.4.4. 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)



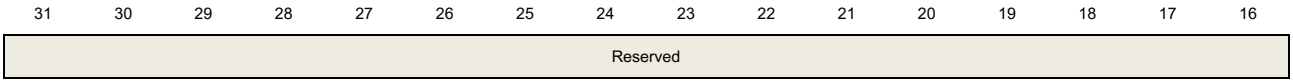
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:4	OUT0_DH[11:0]	DACx_OUT0 12-bit left-aligned data These bits specify the data that is to be converted by DACx_OUT0.
3:0	Reserved	Must be kept at reset value.

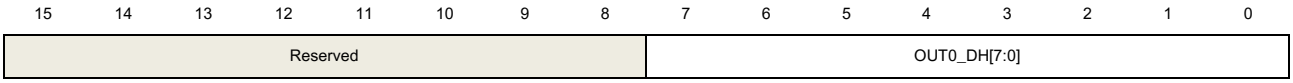
14.4.5. DACx_OUT0 8-bit right-aligned data holding register (DAC_OUT0_R8DH)

Address offset: 0x10

Reset value: 0x0000 0000

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





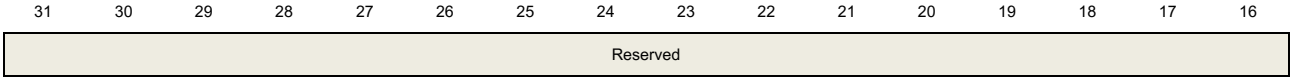
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	OUT0_DH[7:0]	DACx_OUT0 8-bit right-aligned data These bits specify the MSB 8 bits of the data that is to be converted by DACx_OUT0.

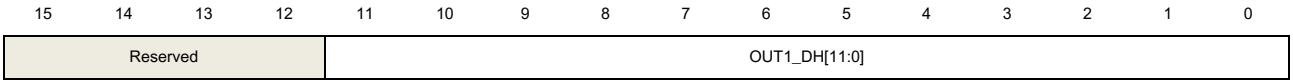
14.4.6. DACx_OUT1 12-bit right-aligned data holding register (DAC_OUT1_R12DH)

Address offset: 0x14

Reset value: 0x0000 0000

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





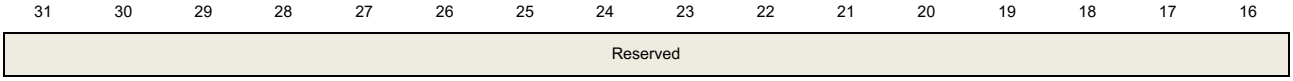
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	OUT1_DH[11:0]	DACx_OUT1 12-bit right-aligned data These bits specify the data that is to be converted by DACx_OUT1.

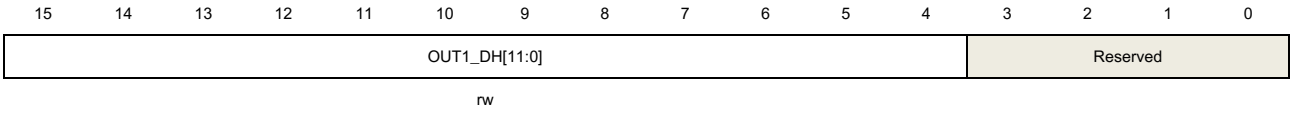
14.4.7. DACx_OUT1 12-bit left-aligned data holding register (DAC_OUT1_L12DH)

Address offset: 0x18

Reset value: 0x0000 0000

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





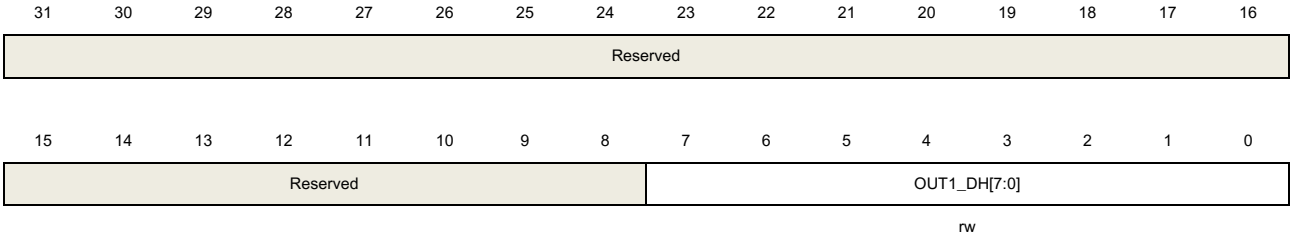
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:4	OUT1_DH[11:0]	DACx_OUT1 12-bit left-aligned data These bits specify the data that is to be converted by DACx_OUT1.
3:0	Reserved	Must be kept at reset value.

14.4.8. 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)



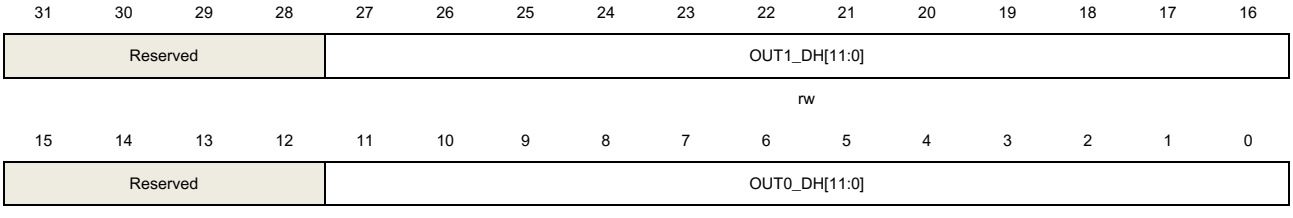
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	OUT1_DH[7:0]	DACx_OUT1 8-bit right-aligned data These bits specify the MSB bits of the data that is to be converted by DACx_OUT1.

14.4.9. 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)



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.

14.4.10. 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.

14.4.11. 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)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT1_DH [7:0]								OUT0_DH [7:0]							
rw								rw							

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.

14.4.12. 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)

31	30	29	28	27	26	25	24	23	22	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_DO [11:0]											
				r											

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, reflect the data that is being converted by DACx_OUT0.

14.4.13. 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)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				OUT1_DO [11:0]											
				r											

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, reflect the data that is being converted by DACx_OUT1.

14.4.14. DACx status register 0 (DAC_STAT0)

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		DDUDR1	Reserved												
		rc_w1													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DDUDR0	Reserved												
		rc_w1													

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29	DDUDR1	DACx_OUT1 DMA underrun flag This bit is set by hardware and cleared by software (by writing it to 1). 0: no underrun occurred 1: underrun occurred (Speed of DAC trigger is high than the DMA transfer).
28:14	Reserved	Must be kept at reset value.
13	DDUDR0	DACx_OUT0 DMA underrun flag This bit is set by hardware and cleared by software (by writing it to 1). 0: No underrun occurred 1: underrun occurred (Speed of DAC trigger is high than the DMA transfer).
12:0	Reserved	Must be kept at reset value.

14.4.15. DACx output control register (DAC_OCTL)

Address offset: 0x80

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														DCSEL1	GS1
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														DCSEL0	GS0
														rw	rw

Bits	Fields	Descriptions
31:18	Reserved	Must be kept at reset value.
17	DCSEL1	DAC0_OUT1 connect GPIO selection 0: DAC is connected to the external pin and to on chip peripherals (CMP). 1: DAC is connected to on chip peripherals (CMP) independently only if DAC0_OUT1 output buffer turns off (DBOFF1=1). Otherwise DAC is connected to the external pin and to on chip peripherals (CMP). Note: This bit is only available for DAC0.
16	GS1	DACx_OUT1 output buffer gain selection 0：1x gain 1：2x gain
15:2	Reserved	Must be kept at reset value.
1	DCSEL0	DAC0_OUT0 connect GPIO selection 0: DAC is connected to the external pin and to on chip peripherals (CMP). 1:DAC is connected to on chip peripherals (CMP) independently only if DAC0_OUT0 output buffer turns off (DBOFF0=1). Otherwise DAC is connected to the external pin and to on chip peripherals (CMP). Note: This bit is only available for DAC0.
0	GS0	DACx_OUT0 output buffer gain selection 0：1x gain 1：2x gain

15. Comparator (CMP)

15.1. Overview

The general purpose CMP can work either standalone (all terminal are available on I / Os) or together with the timers.

It can be used to wake up the MCU from low-power mode by an analog signal, provide a trigger source when an analog signal is in a certain condition, achieve some current control by working together with a PWM output of a timer and the DAC.

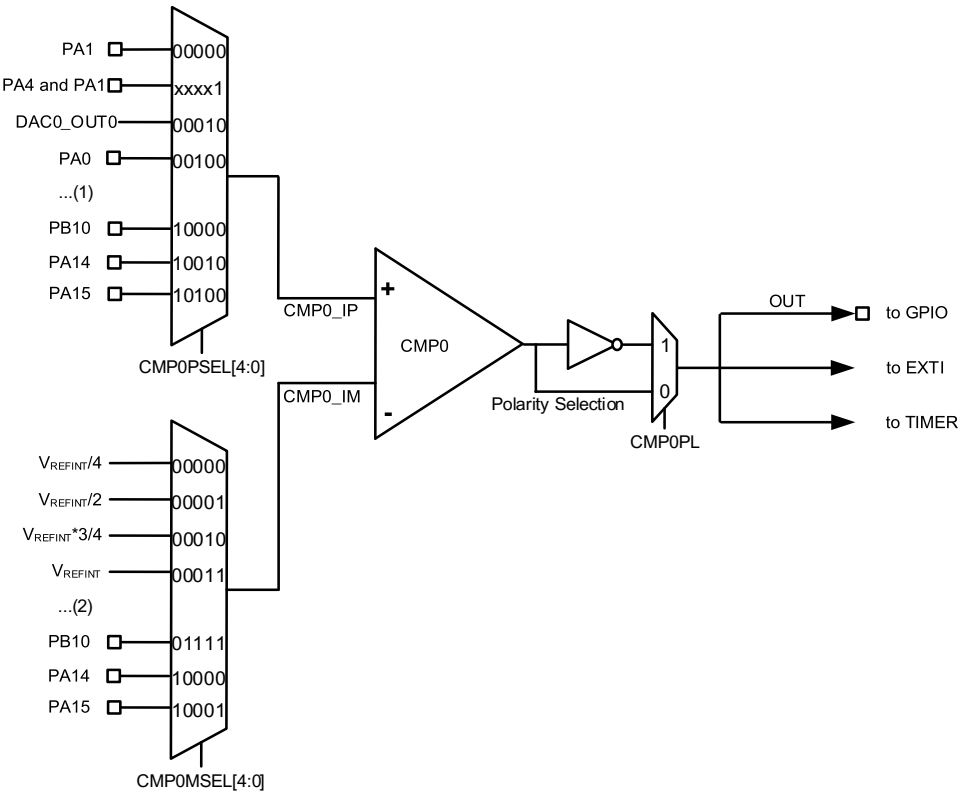
15.2. Characteristics

- Rail-to-rail comparators.
- Configurable hysteresis.
- Configurable speed and consumption.
- Configurable analog input source.
  - DAC output.
  - Multiplexed I / O pins.
  - The whole or sub-multiple values of internal reference voltage.
- Outputs to I / O.
- Outputs to timers for triggering.
- Outputs to EXTI.

15.3. Function overview

The block diagram of CMP is shown as below.

Figure 15-1. CMP block diagram



(1) Refer to CMP0PSEL[4:0] bits in CMPx_CS register.

(2) Refer to CMP0MSEL[4:0] bits in CMPx_CS register.

Note: V_{REFINT} is 1.2V.

15.3.1. CMP clock and reset

The clock of the CMP which is connected to APB bus, is synchronous with PCLK. It shares the common reset and clock enable bits with SYSCFG.

15.3.2. CMP I / O configuration

These I / Os must be configured in analog mode in the GPIOs registers before they are selected as CMP inputs.

The CMP output can be redirected internally and externally simultaneously.

Refer to pin definitions in datasheet, and the CMP output can be connected to the corresponding I/O port via the alternate function of the GPIO.

The CMP output can be redirected internally and externally simultaneously.

CMP output internally connect to the TIMER and the connections between them are as follows:

- CMP output to the TIMER input channel.
- CMP output to the TIMER break.
- CMP output to the TIMER OCPRE_CLR.

In order to work even in Deep-sleep mode, the polarity selection logic and the output redirection to the port work independently from PCLK.

Table 15-1. CMP inputs and outputs summary details the inputs and outputs of the CMP.

Table 15-1. CMP inputs and outputs summary

	CMP0
CMP non inverting inputs connected To I / Os	PA0 PA1 PA2 PA3 PA4 PB0 PB1 PB3 PB10 PA14 PA15
CMP non inverting inputs connected to internal signals	DAC0_OUT0
CMP inverting inputs connected to I / Os	PA0 PA1 PA2 PA3 PA4 PA5 PB0 PB1 PB3 PB10 PA14 PA15
CMP inverting inputs connected to internal signals	V_{REFINT} / 4 V_{REFINT} / 2 V_{REFINT} * 3 / 4 V_{REFINT} DAC0_OUT0 DAC0_OUT1
CMP outputs connected to I / Os(asynchronous	PA0

	CMP0
output)	
CMP outputs connected to I / Os(synchronous output)	PA5
CMP outputs connected to EXTI	•
CMP outputs connected to internal signals	TIMER0_CH0 TIMER1_CH3 TIMER2_CH0 TIMER0 OCPRE_CLR TIMER1 OCPRE_CLR TIMER2 OCPRE_CLR
CMP outputs(motor control protection)	TIMER0 BREAK

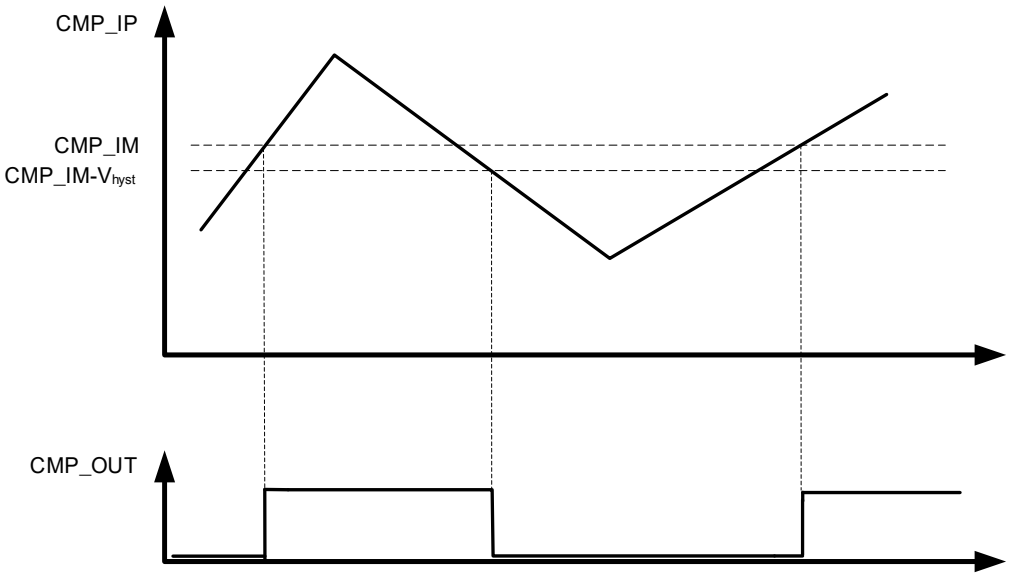
15.3.3. CMP operating mode

For a given application, there is a trade-off between the CMP power consumption versus propagation delay, which is adjusted by configuring bits CMPxM[1:0] in CMPx_CS register. The CMP works fastest with highest power consumption when CMPxM[1:0] = 2'b00, while works slowest with lowest power consumption when CMPxM[1:0] = 2'b11.

15.3.4. CMP hysteresis

In order to avoid spurious output transitions that caused by the noise signal, a programmable hysteresis is designed to force the hysteresis value by configuring CMPx_CS register. This function could be shut down if it is unnecessary.

Figure 15-2. CMP hysteresis



15.3.5. CMP register write protection

The CMP control and status register (CMPx_CS) can be protected from writing by setting CMPxLK bit to 1. The CMPx_CS register, including the CMPxLK bit will be read-only, and can only be reset by the MCU reset.

15.3.6. CMP interrupt

The CMP output is connected to the EXTI and the EXTI line is exclusive to CMP. With this function, CMP can generate either interrupt or event which could be used to exit from low-power mode.

15.3.7. CMP pin reset mode

The PIN_RSTMD bit in the system configuration register 2 (SYSCFG_CFG2) controls whether the register bits (including CMPxHST[1:0], CMPxPL, CMPxOSEL[2:0], CMPxMSEL[4:0], CMPxPSEL[4:0], CMPxM[1:0] , and CMPxEN etc.) are reset when any reset event occurs. When this bit is set, the PIN_RSTMD bit and these register bits are held in all reset events except POR.

15.4. Register definition

CMP base address: 0x4001 001C

15.4.1. CMP Control/status register (CMPx_CS)

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										CMP0PSEL[4:1]				CMP0MSEL[4:3]	
										rw				rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP0LK	CMP0O	CMP0HST[1:0]		CMP0PL	CMP0OSEL[2:0]			Reserved	CMP0MSEL[2:0]			CMP0M[1:0]		CMP0PSEL[0]	CMP0EN
rwo	r	rw		rw	rw				rw			rw		rw	rw

Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.
21:18	CMP0PSEL[4:1]	CMP0_IP input selection Refer to CMP0PSEL[0] description.
17:16	CMP0MSEL[4:3]	CMP0_IM input selection Refer to CMP0MSEL[2:0] description.
15	CMP0LK	CMP0 lock This bit can set all control bits of CMP0 as read-only. It can only be set once by software and cleared by a system reset. 0: CMPx_CS[6:0], CMPx_CS[21:8] bits are read-write 1: CMPx_CS[6:0], CMPx_CS[21:8] bits are read-only
14	CMP0O	CMP0 output state This bit is a copy of CMP0 output state, which is read only. 0: Non-inverting input below inverting input and the output is low 1: Non-inverting input above inverting input and the output is high
13:12	CMP0HST[1:0]	CMP0 hysteresis These bits are used to control the hysteresis level. 00: No hysteresis 01: Low hysteresis 10: Medium hysteresis 11: High hysteresis
11	CMP0PL	Polarity of CMP0 output

		This bit is used to select the polarity of CMP0 output. 0: Output is not inverted 1: Output is inverted
10:8	CMP0OSEL[2:0]	CMP0 output selection These bits are used to select the destination of the CMP0 output. 000: no selection 001: TIMER0 break input 010: TIMER0 CH0 input capture 011: TIMER0 OCPRE_CLR input 100: TIMER1 CH3 input capture 101: TIMER1 OCPRE_CLR input 110: TIMER2 CH0 input capture 111: TIMER2 OCPRE_CLR input **Note:** It is recommended to enable CMP first, and then configure the timer channel, when using TIMER to capture the output signal of the comparator.
7	Reserved	Must be kept at reset value.
6:4	CMP0MSEL[2:0]	CMP0_IM input selection The CMP0MSEL[2:0] and CMP0MSEL[4:3] bits are used to select the source connected to the CMP0_IM input of the CMP0. 00000: V_{REFINT} / 4 00001: V_{REFINT} / 2 00010: V_{REFINT} * 3 / 4 00011: V_{REFINT} 00100: PA4 00101: PA5 00110: PA0 00111: PA2 01000: DAC0_OUT0 01001: DAC0_OUT1 01010: PA1 01011: PA3 01100: PB0 01101: PB1 01110: PB3 01111: PB10 10000: PA14 10001: PA15 others: Reserved
3:2	CMP0M[1:0]	CMP0 mode These bits are used to control the operating mode of the CMP0 adjust the speed / consumption.

		00: High speed / full power 01: Medium speed / medium power 10: Low speed / low power 11: Very-low speed / ultra-low power
1	CMP0PSEL[0]	CMP0_IP input selection The CMP0PSEL[0] and CMP0PSEL[0][4:1] bits are used to select the source connected to the CMP0_IP input of the CMP. 00000: PA1 xxxx1: PA1 and PA4 (All need to be configured as analog pins) 00010: DAC0_OUT0 00100: PA0 00110: PA2 01000: PA3 01010: PB0 01100: PB1 01110: PB3 10000: PB10 10010: PA14 10100: PA15 others: Reserved
0	CMP0EN	CMP0 enable 0: CMP0 disabled 1: CMP0 enabled

16. Configurable Logic Array (CLA)

16.1. Overview

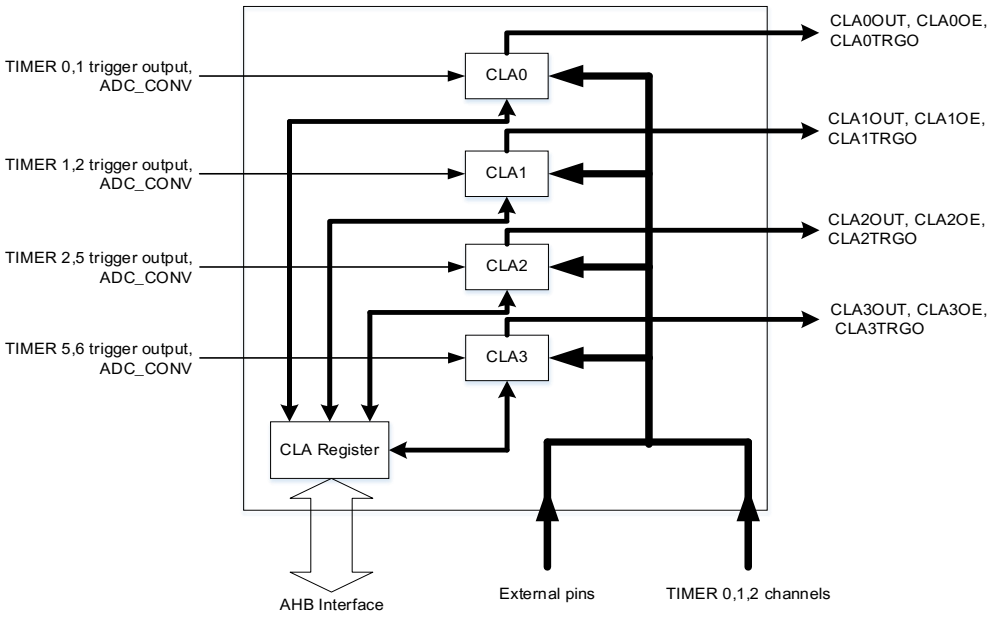
The configurable logic array provides 256 programmable digital logic operations for external pins, ADC and timers without intervention from the CPU. Four independent CLA units are implemented in this module. Each CLA unit supports configurable asynchronous and synchronous output for GPIO pins.

16.2. Characteristics

- Four independent CLA units, with two input Signal Selector(SIGS), supporting 16 input signals, including external pins, timer channels, ADC, and CLA asynchronous outputs
- A Logic-Configure-Unit(LCU) providing 256 programmable digital logic functions is implemented in each CLA units
- Programmable asynchronous and synchronous output
- CLA output can be configured to synchronize with external pins and timers.
- Four CLA units can be combined and support complicated logic operations.

16.3. Block diagram

Figure 16-1. Block diagram of CLA



16.4. Function overview

Four identical CLA units are inplemented in this module. Two SIGS are implemented in each CLA units. In addition, a LCU is included in each CLA units.

16.4.1. CLA input Signal Selector

Each CLA unit includes two Signal Selector: SIGS0 and SIGS1. The input of each SIGS include several external pins, timer channels[TIMERx_CHx], timer trigger output[TIMERx_TRGO], ADC conversion signal[ADC_CONV] and CLA units asynchronuous output[CLAx_ASYNC_OUT]. When another CLA units asynchoronous output is selected as SIGS input, a complicated combinatorial logic operation is realized. When the TIMERx_TRGO is selected as SIGS input, only the first HCLK cycle is valid, and the rest of the high TIMERx_TRGO is regarded as logic LOW.

Table 16-1. CLAxSIGS0 input selection and *Table 16-2. CLAxSIGS1 input selection* show the CLAxSIGS0 and CLAxSIGS1 input selection.

Table 16-1. CLAxSIGS0 input selection

SIGS0[3:0]	CLA0SIGS0	CLA1SIGS0	CLA2SIGS0	CLA3SIGS0
0000	CLA0_ASYNC_OUT	CLA0_ASYNC_OUT	CLA0_ASYNC_OUT	CLA0_ASYNC_OUT
0001	CLA1_ASYNC_OUT	CLA1_ASYNC_OUT	CLA1_ASYNC_OUT	CLA1_ASYNC_OUT
0010	CLA2_ASYNC_OUT	CLA2_ASYNC_OUT	CLA2_ASYNC_OUT	CLA2_ASYNC_OUT
0011	CLA3_ASYNC_OUT	CLA3_ASYNC_OUT	CLA3_ASYNC_OUT	CLA3_ASYNC_OUT
0100	TIMER1_TRGO	TIMER2_TRGO	TIMER5_TRGO	TIMER6_TRGO
0101	TIMER0_CH0	TIMER0_CH0	TIMER0_CH1	TIMER0_CH2
0110	TIMER0_CH1	TIMER0_CH3	TIMER0_CH3	TIMER1_CH0
0111	TIMER0_CH2	TIMER1_CH0	TIMER1_CH1	TIMER1_CH1
1000	CLAIN0	CLAIN4	CLAIN0	CLAIN2
1001	CLAIN2	CLAIN5	CLAIN1	CLAIN3
1010	CLAIN4	CLAIN8	CLAIN8	CLAIN6
1011	CLAIN6	CLAIN10	CLAIN9	CLAIN7
1100	CLAIN8	CLAIN12	CLAIN14	CLAIN10
1101	CLAIN10	CLAIN13	CLAIN15	CLAIN11
1110	CLAIN12	CLAIN16	CLAIN16	CLAIN18
1111	CLAIN14	CLAIN18	CLAIN17	CLAIN19

Table 16-2. CLAxSIGS1 input selection

SIGS1[3:0]	CLA0SIGS1	CLA1SIGS1	CLA2SIGS1	CLA3SIGS1
0000	CLA0_ASYNC_OUT	CLA0_ASYNC_OUT	CLA0_ASYNC_OUT	CLA0_ASYNC_OUT
0001	CLA1_ASYNC_OUT	CLA1_ASYNC_OUT	CLA1_ASYNC_OUT	CLA1_ASYNC_OUT
0010	CLA2_ASYNC_OUT	CLA2_ASYNC_OUT	CLA2_ASYNC_OUT	CLA2_ASYNC_OUT
0011	CLA3_ASYNC_OUT	CLA3_ASYNC_OUT	CLA3_ASYNC_OUT	CLA3_ASYNC_OUT

SIGS1[3:0]	CLA0SIGS1	CLA1SIGS1	CLA2SIGS1	CLA3SIGS1
0100	ADC_CONV	ADC_CONV	ADC_CONV	ADC_CONV
0101	TIMER0_CH3	TIMER0_CH1	TIMER0_CH0	TIMER0_CH0
0110	TIMER1_CH0	TIMER0_CH2	TIMER0_CH2	TIMER0_CH1
0111	TIMER1_CH1	TIMER1_CH1	TIMER1_CH0	TIMER0_CH3
1000	CLAIN1	CLAIN6	CLAIN2	CLAIN0
1001	CLAIN3	CLAIN7	CLAIN3	CLAIN1
1010	CLAIN5	CLAIN9	CLAIN10	CLAIN4
1011	CLAIN7	CLAIN11	CLAIN11	CLAIN5
1100	CLAIN9	CLAIN14	CLAIN12	CLAIN8
1101	CLAIN11	CLAIN15	CLAIN13	CLAIN9
1110	CLAIN13	CLAIN17	CLAIN18	CLAIN16
1111	CLAIN15	CLAIN19	CLAIN19	CLAIN17

16.4.2. LCU control

A LCU is implemented in each CLA unit, A LCU is implemented in each CLA unit which introduce 256 combinatorial logic functions controlled by LCUCTL [7:0] bits in the CLAx_LCUCTL register. The LCU has three inputs, which have no effect on LCU output when the CLA unit is disabled.

- Input0(IN0): SIGS0 output
- Input1(IN1): SIGS1 output
- Input2(IN2): CLA result from CLA[x-1]

If the CLA unit is disabled, the LCU inputs are force to '0'.

The 8-bit LCUCTL controls which logic function of the input 0, input 1 and input 2 can be effected on the output, as shown in *Table 16-3. LCU control*.

Table 16-3. LCU control

LCUCTL[7:0]	IN0	IN1	IN2
bit 0	0	0	0
bit 1	0	0	1
bit 2	0	1	0
bit 3	0	1	1
bit 4	1	0	0
bit 5	1	0	1
bit 6	1	1	0
bit 7	1	1	1

For example, if the logic function (IN0^IN1^IN2) is required, the LCUCTL[7:0] in CLAx_LCUCTL shoud be configured like this: when {IN0, IN1 IN2} == 3'b000, the result of (IN0^IN1^IN2) is 1'b0, so bit 0 of LCUCTL[7:0] = 0; when {IN0, IN1 IN2} == 3'b001, the result of (IN0^IN1^IN2) is 1'b1, so bit 1 of LCUCTL[7:0] = 1; when {IN0, IN1 IN2} == 3'b010, the

result of (IN0^IN1^IN2) is 1'b1, so bit 2 of LCUCTL[7:0] = 1; when {IN0, IN1 IN2} == 3'b011, the result of (IN0^IN1^IN2) is 1'b0, so bit 3 of LCUCTL[7:0] = 0; when {IN0, IN1 IN2} == 3'b100, the result of (IN0^IN1^IN2) is 1'b1, so bit 4 of LCUCTL[7:0] = 1; when {IN0, IN1 IN2} == 3'b101, the result of (IN0^IN1^IN2) is 1'b0, so bit 5 of LCUCTL[7:0] = 0; when {IN0, IN1 IN2} == 3'b110, the result of (IN0^IN1^IN2) is 1'b0, so bit 6 of LCUCTL[7:0] = 0; when {IN0, IN1 IN2} == 3'b111, the result of (IN0^IN1^IN2) is 1'b1, so bit 7 of LCUCTL[7:0] = 1. Thus, LCUCTL[7:0] = 8'b10010110.

16.4.3. CLA Output

Each CLA unit has an output to the selected GPIO pins, which can be the LCU result or the LCU result after a flip-flop, selected by the OSEL bit in the CLAx_CTL register.

The clock source and the clock polarity of the flip-flop can be selected by the CSEL[1:0] bits and CPOL bit of the CLAx_CTL register, respectively. There are four clocks which can be used as the clock source of flip-flop, they are:

- CLA[x-1] result: the LCU result of the previous CLA units (the CLA3 LCU result is sent to the CLA0).
- SIGS0 output: the output of SIGS0
- HCLK
- TIMER_TRGO: TIMER0_TRGO for CLA0, TIMER1_TRGO for CLA1, TIMER2_TRGO for CLA2, TIMER5_TRGO for CLA3.

Writing the FFRST bit in the CLAX_CTL register to "1" can reset the flip-flop output. If the flip-flop output is selected as the CLA output, it is recommended to set the FFRST bit before enable the CLA unit.

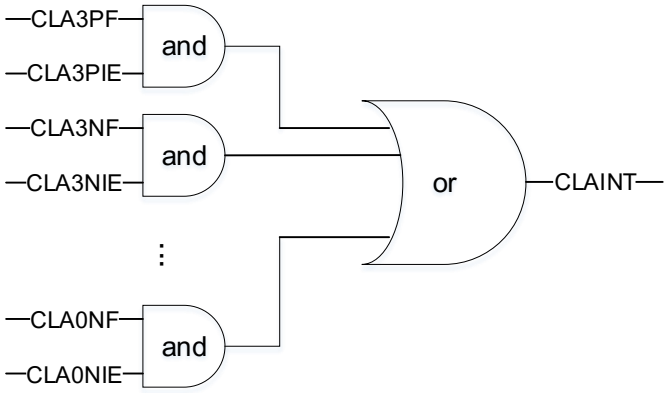
If the OEN bit is reset, the CLAxOE is reset, and the corresponding GPIO pins output function is disable.

Each CLA unit has an synchronizer, the HCLK is output to CLAx_TRGO after the synchronizer. which is sent to the ADC and DAC, as a trigger event.

16.4.4. Interrupt

Eight interrupt flags are implemented in the CLA_INTF register, including CLA3PF, CLA3NF, CLA2PF, CLA2NF, CLA1PF, CLA1NF, CLA0PF and CLA0NF. Each state flag has a dedicated enable bit in the CLA_GCTL register. The CLA interrupt logic is shown in ***Figure 16-2. CLA interrupt logic***, an interrupt can be produced when any flags is detected and the corresponding enable bit is enabled.

Figure 16-2. CLA interrupt logic



16.5. Register definition

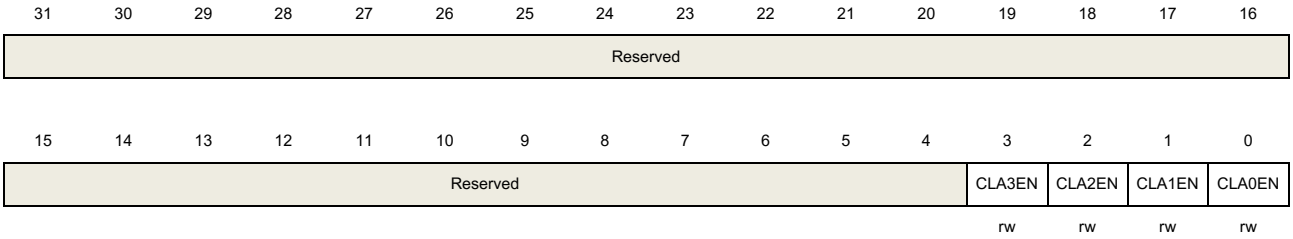
CLA base address: 0x4003 8000

16.5.1. Global control register (CLA_GCTL)

Address offset: 0x00

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	CLA3EN	CLA3 unit enable Software set and cleared. 0: CLA3 unit is disabled 1: CLA3 unit is enabled
2	CLA2EN	CLA2 unit enable Software set and cleared. 0: CLA2 unit is disabled 1: CLA2 unit is enabled
1	CLA1EN	CLA1 unit enable Software set and cleared. 0: CLA1 unit is disabled 1: CLA1 unit is enabled
0	CLA0EN	CLA0 unit enable Software set and cleared. 0: CLA0 unit is disabled 1: CLA0 unit is enabled

16.5.2. Interrupt enable register (CLA_INTE)

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								CLA3PIE	CLA3NIE	CLA2PIE	CLA2NIE	CLA1PIE	CLA1NIE	CLA0PIE	CLA0NIE
								rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	CLA3PIE	CLA3 unit posedge interrupt enable. Software set and cleared 0: CLA3 unit posedge interrupt is disabled. 1: CLA3 unit posedge interrupt is enabled. An interrupt is generated when the CLA3PF bit is set.
6	CLA3NIE	CLA3 unit negedge interrupt enable. Software set and cleared 0: CLA3 unit negedge interrupt is disabled. 1: CLA3 unit negedge interrupt is enabled. An interrupt is generated when the CLA3NF bit is set.
5	CLA2PIE	CLA2 unit posedge interrupt enable. Software set and cleared. 0: CLA2 unit posedge interrupt is disabled. 1: CLA2 unit posedge interrupt is enabled. An interrupt is generated when the CLA2PF bit is set.
4	CLA2NIE	CLA2 unit negedge interrupt enable. Software set and cleared. 0: CLA2 unit negedge interrupt is disabled. 1: CLA2 unit negedge interrupt is enabled. An interrupt is generated when the CLA2NF bit is set.
3	CLA1PIE	CLA1 unit posedge interrupt enable. Software set and cleared. 0: CLA1 unit posedge interrupt is disabled. 1: CLA1 unit posedge interrupt is enabled. An interrupt is generated when the CLA1PF bit is set.
2	CLA1NIE	CLA1 unit negedge interrupt enable. Software set and cleared. 0: CLA1 unit negedge interrupt is disabled. 1: CLA1 unit negedge interrupt is enabled. An interrupt is generated when the CLA1NF bit is set.

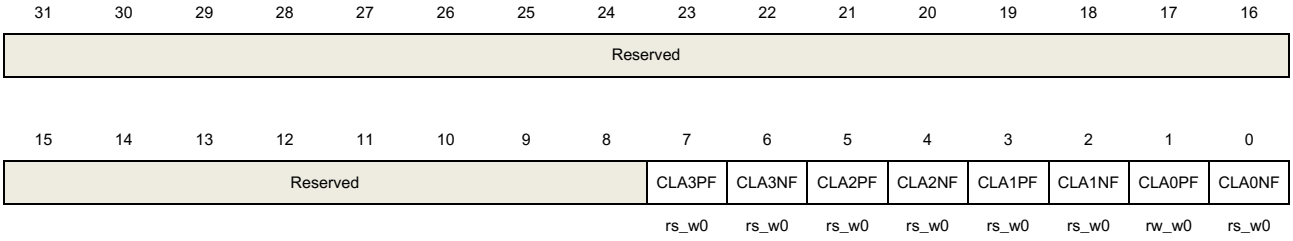
1	CLA0PIE	CLA0 unit posedge interrupt enable. Software set and cleared. 0: CLA0 unit posedge interrupt is disabled. 1: CLA0 unit posedge interrupt is enabled. An interrupt is generated when the CLA0PF bit is set.
0	CLA0NIE	CLA0 unit negedge interrupt enable. Software set and cleared. 0: CLA0 unit negedge interrupt is disabled. 1: CLA0 unit negedge interrupt is enabled. An interrupt is generated when the CLA0NF bit is set.

16.5.3. Interrupt flag register (CLA_INTF)

Address offset: 0x08

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	CLA3PF	CLA3 unit posedge flag. Hareware set and software cleared. 0: A CLA3 output posedge has not been detected. 1: A CLA3 output posedge has been detected.
6	CLA3NF	CLA3 unit negedge flag. Hareware set and software cleared. 0: A CLA3 output negedge has not been detected. 1: A CLA3 output negedge has been detected.
5	CLA2PF	CLA2 unit posedge flag. Hareware set and software cleared. 0: A CLA2 output posedge has not been detected. 1: A CLA2 output posedge has been detected.
4	CLA2NF	CLA2 unit negedge flag. Hareware set and software cleared. 0: A CLA2 output negedge has not been detected.

		1: A CLA2 output negedge has been detected.
3	CLA1PF	CLA1 unit posedge flag. Hareware set and software cleared. 0: A CLA1 output posedge has not been detected. 1: A CLA1 output posedge has been detected.
2	CLA1NF	CLA1 unit negedge flag. Hareware set and software cleared. 0: A CLA1 output negedge has not been detected. 1: A CLA1 output negedge has been detected.
1	CLA0PF	CLA0 unit posedge flag. Hareware set and software cleared. 0: A CLA0 output posedge has not been detected. 1: A CLA0 output posedge has been detected.
0	CLA0NF	CLA0 unit negedge flag. Hareware set and software cleared. 0: A CLA0 output negedge has not been detected. 1: A CLA0 output negedge has been detected.

16.5.4. Status register (CLA_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												CLA3OU	CLA2OU	CLA1OU	CLA0OU
												r	r	r	r

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	CLA3OUT	CLA3 unit output state. Hardware set and cleared. 0: The current logic level of CLA3 unit is LOW. 1: The current logic level of CLA3 unit is HIGH.
2	CLA2OUT	CLA2 unit output state. Hardware set and cleared.

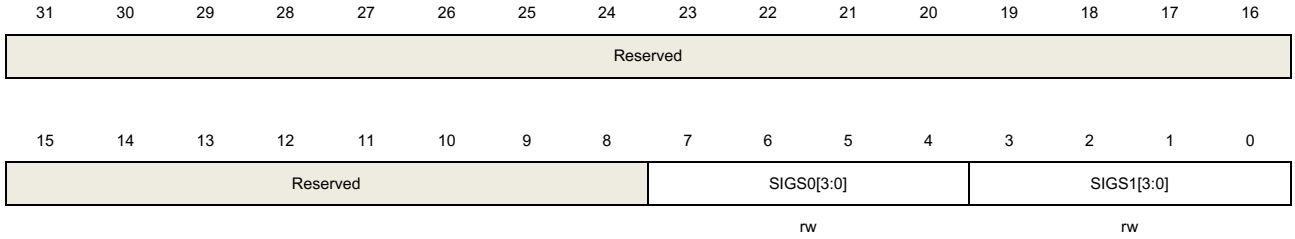
		0: The current logic level of CLA2 unit is LOW. 1: The current logic level of CLA2 unit is HIGH.
1	CLA1OUT	CLA1 unit output state. Hardware set and cleared. 0: The current logic level of CLA1 unit is LOW. 1: The current logic level of CLA1 unit is HIGH.
0	CLA0OUT	CLA0 unit output state Hardware set and cleared. 0: The current logic level of CLA0 unit is LOW. 1: The current logic level of CLA0 unit is HIGH.

16.5.5. Signal selection register (CLAx_SIGS)(x=0..3)

Address offset: 0x10 + 0x0C * x

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.
7:4	SIGS0[3:0]	Signal Selector 0 input selection. Software set and cleared. These bits select the input for Signal Selector 0. Please reference ***Table 16-1. CLAxSIGS0 input selection***.
3:0	SIGS1[3:0]	Signal Selector 1 input selection. Software set and cleared. These bits select the input for Signal Selector 1. Please reference ***Table 16-2. CLAxSIGS1 input selection***.

16.5.6. LCU control register (CLAx_LCUCTL) (x=0..3)

Address offset: 0x14 + 0x0C * x

Reset value: 0x0000 0000

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

Reserved

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

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	LCUCTL[7:0]	LCU control Software set and cleared These bits controll which logic function of the input0, input1 and input2 can be effected on the output. Examples: IN1 | IN2: LCUCTL = 8'b11101110 IN0 & (IN1 ^ IN2): LCUCTL = 8'b01100000

16.5.7. Control register (CLAx_CTL) (x=0..3)

Address offset: 0x18 + 0x0C * x

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				SWBEN0	SWBEN1	SWB0	SWB1	OSEL	OEN	Reserved		FFRST	CPOL	CSEL[1:0]	
				rw	rw	rw	rw	rw	rw			w	rw	rw	

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11	SWBEN0	Enable software bit 0 0: Disable software bit 0 1: Enable software bit 0
10	SWBEN1	Enable software bit 1 0: Disable software bit 1 1: Enable software bit 1
9	SWB0	Software bit 0 If SWBEN0 is 1, the output of CLAxSIGS0 is the value of SWB0, otherwise the output is the input selected by CLAxSIGS0.
8	SWB1	Software bit 1

		If SWBEN1 is 1, the output of CLAxSIGS1 is the value of SWB1, otherwise the output is the input selected by CLAxSIGS1.
7	OSEL	Output selection Software set and cleared 0: Flip-flop output is selected as CLAx output 1: LCU result is selected as CLAx output
6	OEN	Output enable Software set and cleared 0: CLAx output is disabled 1: CLAx output is enabed
5:4	Reserved	Must be kept at reset value
3	FFRST	Flip-flop output reset 0: no effect 1: Reset the flip-flop output asynchronously
2	CPOL	Clock polarity of Flip-flop Software set and cleared This bitfield must not be modified once the CLAx is enabled(CLAxEN = 1) 0: clock posedge is valid 1: clock negedge is valid
1:0	CSEL[1:0]	Flip-flop clock source selection Software set and cleared This bitfield must not be modified once the CLAx is enabled(CLAxEN = 1) 00: CLA[x-1] LCU result 01: SIGS0 output 10: HCLK 11: TIMER_TRGO

17. VREF

17.1. Overview

A precision internal reference circuit is inside. The precision internal reference is used to provide reference voltage for ADC/DAC, or used by off-chip circuit connecting to V_{REFP} pin.

17.2. Characteristics

The precision internal reference features are described as follows:

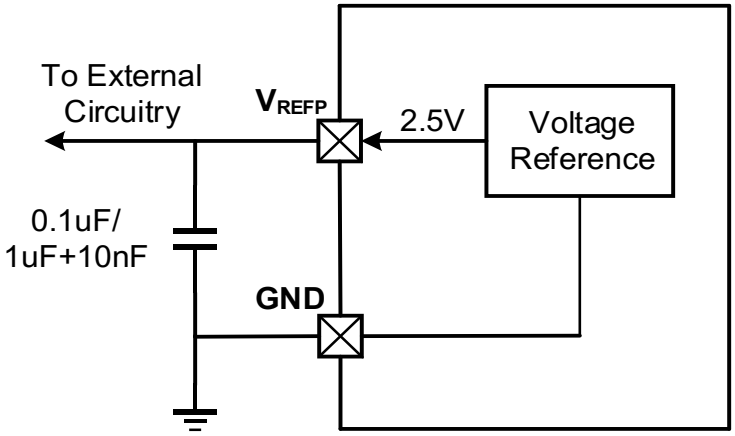
- Stable voltage, and product trimmed.
- Connects to V_{REFP} pin to source off-chip circuits.
- Provides 2.5V reference voltage.

17.3. Function overview

The precision reference includes one control signals, VREF_EN bit in SYSCFG_CFG1 register. VREF_EN is used to enable the internal reference, producing 2.5V reference voltage and connecting to V_{REFP} pin. When VREF_EN is disabled, off-chip reference voltage could be injected to V_{REFP} pin to source ADC/DAC. If there is no V_{REFP} pin (refer to datasheet), the V_{REFP} is connected to V_{DDA} and the VREF_EN bit in SYSCFG_CFG1 register must keep 0.

When using precision internal reference voltage, and a bypass capacitor about 0.1uF (or 1uF) and 10nF connected in parallel to ground is required.

Figure 17-1. Precision Reference Connection



18. Infrared ray port (IFRP)

18.1. Overview

Infrared ray port (IFRP) is used to control infrared light LED, and send out infrared data to implement infrared ray remote control.

There is no register in this module, which is controlled by TIMER15 and TIMER16. Enable the module's output by set the GPIO pin to IFRP Mode.

18.2. Characteristics

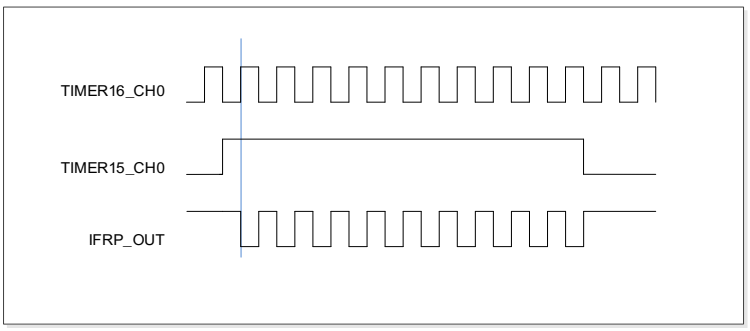
- The IFRP output signal is decided by TIMER15_CH0 and TIMER16_CH0.
- To get correct infrared ray signal, TIMER15 should generate low frequence modulation envelope signal, and TIMER16 should generate high frequence carrier signal.
- The IFRP output (PB9) can control LED interface by setting PB9MPEN in SYSCFG_CFG0.

18.3. Function overview

IFRP is a module which is able to integrate the output of TIMER15 and TIMER16 to generate an infrared ray signal.

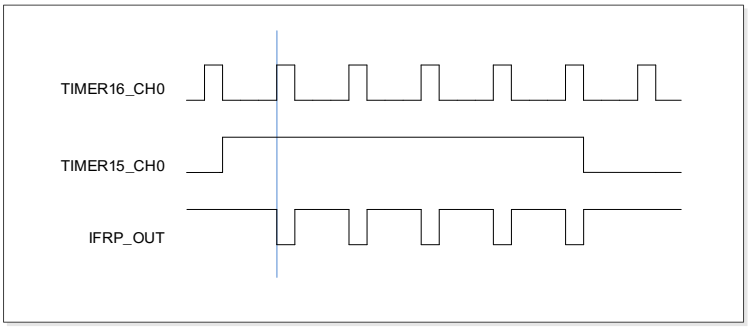
1. The TIMER15's CH0 is programed to generate the low frequence PWM signal which is the modulation envelope signal. The TIMER16's CH0 is programed to generate the high frquence PWM signal which is the carrier signal. And the channel need to be enabled before generating these signals.
2. Program the GPIO remap regisger and enable the pin.
3. Enable IFRP_OUT output to PB9 by setting PB9MPEN to IFRP Mode in SYSCFG_CFG0.

Figure 18-1. IFRP output timechart 1



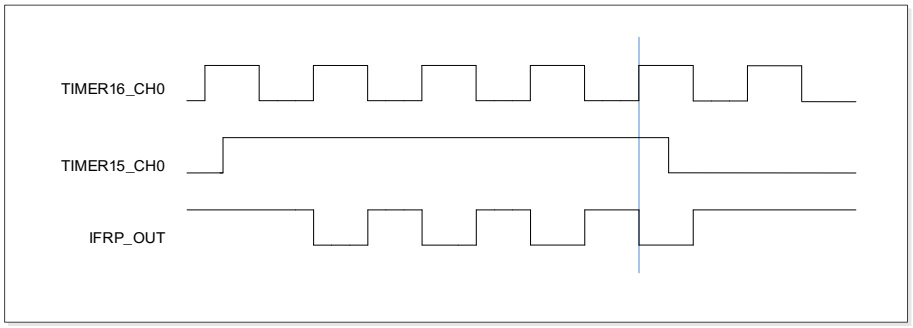
Note: IFRP_OUT has one APB clock delay from TIMER16_CH0.

Figure 18-2. IFRP output timechart 2



Note: Carrier (TIMER15_CH0)'s duty cycle can be changed, and IFRP_OUT has inverted relationship with TIMER16_CH0 when TIMER15_CH0 is high.

Figure 18-3. IFRP output timechart 3



Note: IFRP_OUT will keep the integrity of TIMER16_CH0, even if envelope signal (TIMER15_CH0) is no active.

19. Universal synchronous/asynchronous receiver / transmitter (USART)

19.1. 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 (CK_SYS, PCLK2 and CK_IRC8M) to produces a dedicated wide range baudrate 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, half-duplex mode and 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.

All USARTs support DMA function for high-speed data communication.

19.2. Characteristics

- Automatic baud rate detection.
- NRZ standard format
- Asynchronous, full duplex communication
- Half duplex single wire communications
- Receive FIFO function
- Dual clock domain:
  - Asynchronous PCLK and USART clock
  - Baud rate programming independent from the PCLK reprogramming
- Programmable baud-rate generator allowing speed up to 9 MBits/s when the clock frequency is 72 MHz and oversampling is by 8
- Fully programmable serial interface characteristics:
  - A data word (8 or 9 bits) LSB or MSB first
  - Even, odd or no-parity bit generation/detection
  - 0.5, 1, 1.5 or 2 stop bit generation
- Swappable TX/RX pin
- Configurable data polarity
- Hardware modem operations (CTS/RTS) and RS485 drive enable
- Configurable multibuffer communication using centralized DMA
- Separate enable bits for transmitter and receiver

- Parity control
  - Transmits parity bit
  - Checks parity of received data byte
- 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 line or address match detection
- Support for ModBus communication
  - Timeout feature
  - CR/LF character recognition
- Wake up from deep-sleep mode
  - By standard RBNE interrupt
  - By WUF interrupt
- Various status flags
  - Flags for transfer detection: receive buffer not empty (RBNE), receive FIFO full (RFF), transmit buffer empty (TBE), transfer complete (TC).
  - 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)
  - Flag for ModBus communication: address/character match (AMF) and receiver timeout (RTF)
  - Flags for smartcard block mode: end of block (EBF) and receiver timeout (RTF)
  - Wakeup from deep-sleep mode flag
  - Interrupt occurs at these events when the corresponding interrupt enable bits are set

While USART0 is fully implemented, UART1 is only partially implemented with the following features not supported.

- Automatic baud rate detection
- Smartcard mode
- IrDA SIR ENDEC block
- LIN mode
- Dual clock domain and wakeup from Deep-sleep mode
- Receiver timeout interrupt
- ModBus communication

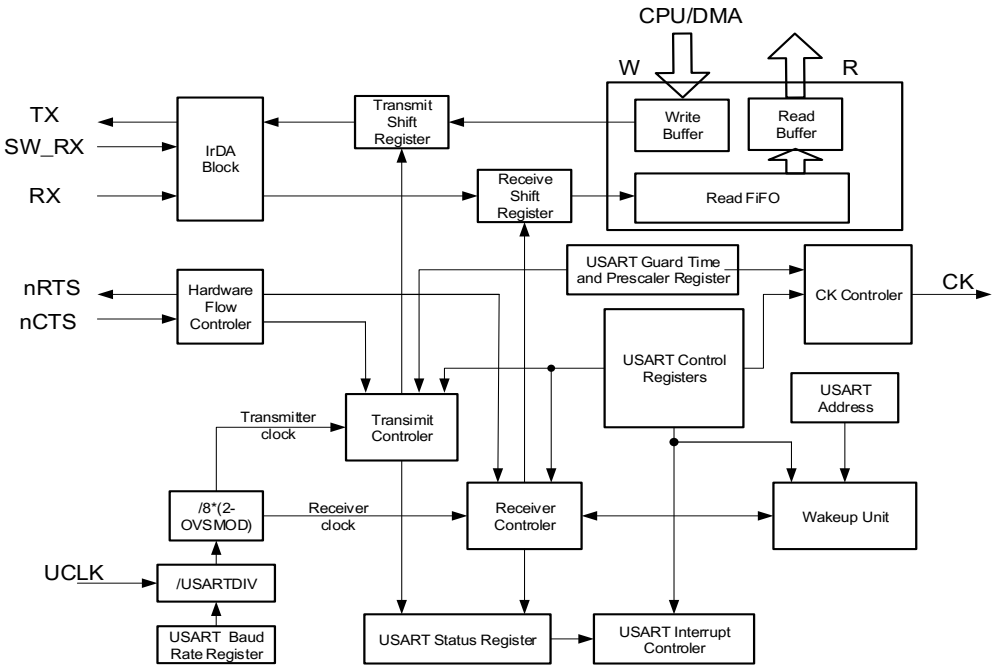
19.3. Function overview

The interface is externally connected to another device by the main pins listed in ***Table 19-1. Description of USART important pins***.

Table 19-1. 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

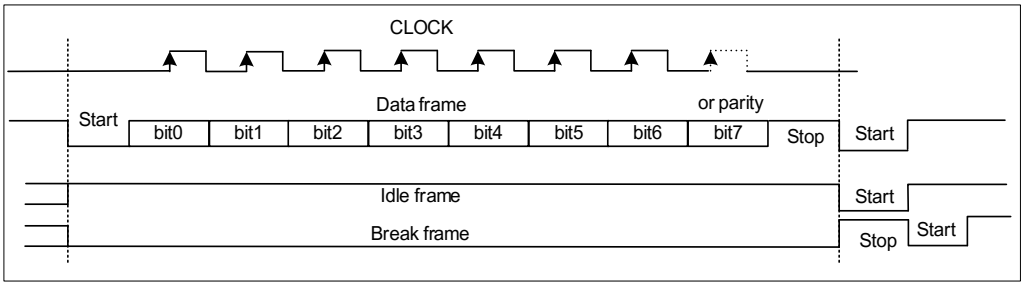
Figure 19-1. USART module block diagram



19.3.1. 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 of 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 19-2. 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 19-2. 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 and single-wire 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.

The break frame structure is a 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.

19.3.2. 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.

The baud-rate divider (USARTDIV) has the following relationship with the UCLK:

In case of oversampling by 16, the equation is:

$$\text{USARTDIV} = \frac{\text{UCLK}}{16 \times \text{Baud Rate}} \quad (19\text{-}1)$$

In case of oversampling by 8, the equation is:

$$\text{USARTDIV} = \frac{\text{UCLK}}{8 \times \text{Baud Rate}} \quad (19\text{-}2)$$

For example, when oversampled by 16:

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.

19.3.3. 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_CTL1 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_TDATA when the TBE bit in the USART_STAT register is asserted. The TBE bit is cleared by writing USART_TDATA 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_TDATA 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_TDATA 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.

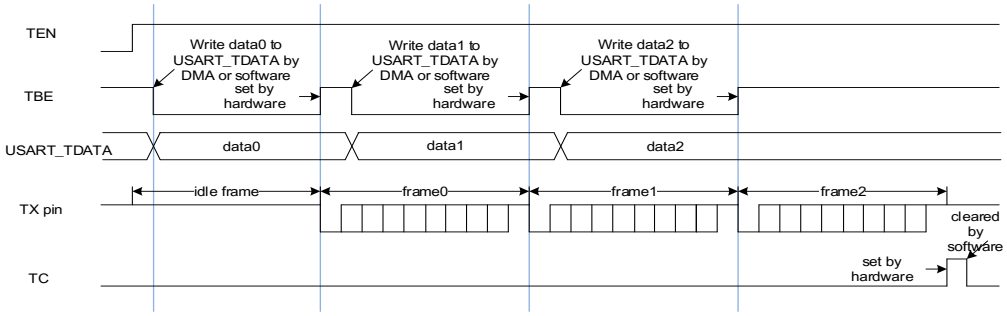
If a frame is transmitted and the TBE bit is asserted, the TC bit of the USART_STAT register will be set. An interrupt will be generated if the corresponding interrupt enable bit (TCIE) is set in the USART_CTL0 register.

The USART transmit procedure is shown in ***Figure 19-3. USART transmit procedure***. The software operating process is as follows:

1. Write the WL bit in USART_CTL0 to set the data bits length.
2. Set the STB[1:0] bits in USART_CTL1 to configure the number of stop bits.
3. Enable DMA (DENT bit) in USART_CTL2 if multibuffer communication is selected.
4. Set the baud rate in USART_BAUD.
5. Set the UEN bit in USART_CTL0 to enable the USART.
6. Set the TEN bit in USART_CTL0.
7. Wait for the TBE being asserted.
8. Write the data to the USART_TDATA register.

9. Repeat step7-8 for each data, if DMA is not enabled.
10. Wait until TC=1 to finish.

Figure 19-3. 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. The TC bit can be cleared by writing 1 to TCC bit in USART_INTC register.

The break frame is sent when the SBKCMD bit is set, and SBKCMD bit is reset after the transmission.

19.3.4. USART receiver

After power on, the USART receiver can be enabled by the following procedure:

1. Write the WL bit in USART_CTL0 to set the data bits length.
2. Set the STB[1:0] bits in USART_CTL1.
3. Enable DMA (DENR bit) in USART_CTL2 if multibuffer communication is selected.
4. Set the baud rate in USART_BAUD.
5. Set the UEN bit in USART_CTL0 to enable the USART.
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_STAT 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 is stored in the USART_STAT register.

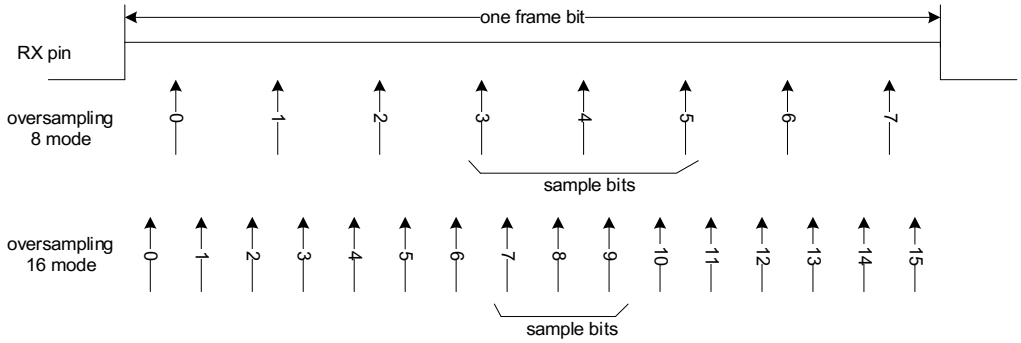
The software can get the received data by reading the USART_RDATA register directly, or through DMA. The RBNE bit is cleared by a read operation on the USART_RDATA register, whatever it is performed by software directly, or through DMA.

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. If the oversampling 8 mode is enabled, the 3rd, 4th and 5th samples are used, 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 a data bit, parity bit or stop bit, a noisy error (NERR) status 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. If the OSB bit in USART_CTL2 register is set, the receiver gets only one sample to evaluate a bit value. In this situation, no noisy error will be detected.

Figure 19-4. Oversampling method of a receive frame bit (OSB=0)



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_STAT 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_STAT register will be set. An interrupt is generated, If the receive DMA is enabled and the ERRIE bit in USART_CTL2 register is set. According to the configuration of the stop bit, there are the following situations:

- 0.5 stop bit: When 0.5 stop bit, stop bit is not sampled.
- 1 stop bit: When 1 stop bit, sampling in the middle of stop bit.
- 1.5 stop bits: When 1.5 stop bits, the 1.5 stop bits are divided into 2 parts: the 0.5 stop bit part is not sampled and sampling in the middle of 1 stop bit.
- 2 stop bits: When 2 stop bits, if a frame error is detected during the first stop bit, the frame error flag is set, the second stop bit is not checked frame error. If no frame error is detected during the first stop bit, then continue to check the second stop bit for frame error.

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_STAT 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) occurs during reception, NERR, PERR, FERR or ORERR will be set at the same time with RBNE. If the receive DMA is not enabled, when the RBNE interrupt occurs, software need to

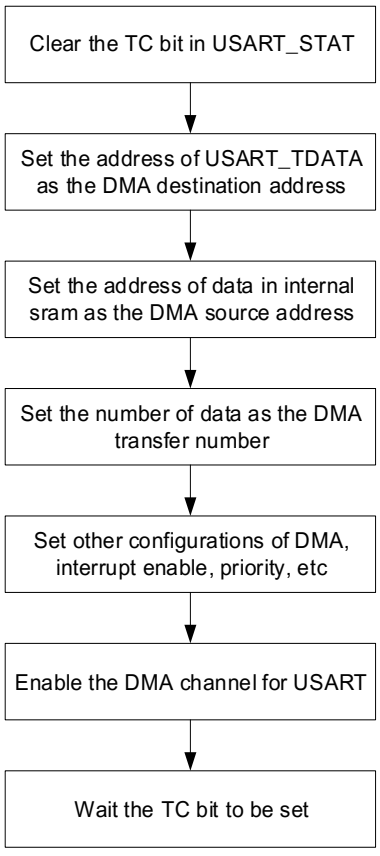
check whether there is a noise error, parity error, frame error or overrun error.

19.3.5. 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 step are shown in ***Figure 19-5. Configuration step when using DMA for USART transmission***.

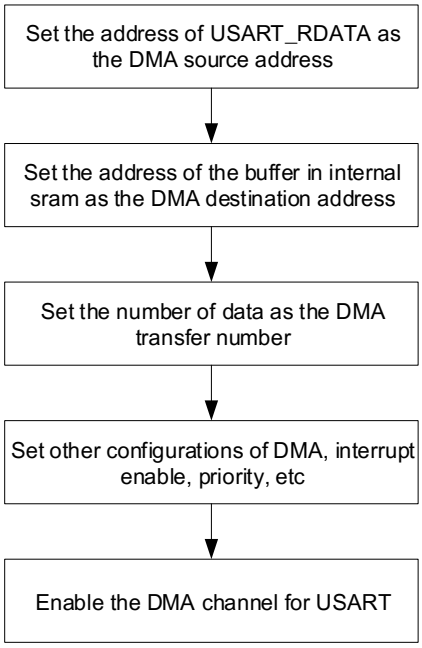
Figure 19-5. Configuration step when using DMA for USART transmission



After all of the data frames are transmitted, the TC bit in USART_STAT 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 19-6. Configuration step 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_STAT.

Figure 19-6. Configuration step when using DMA for USART reception

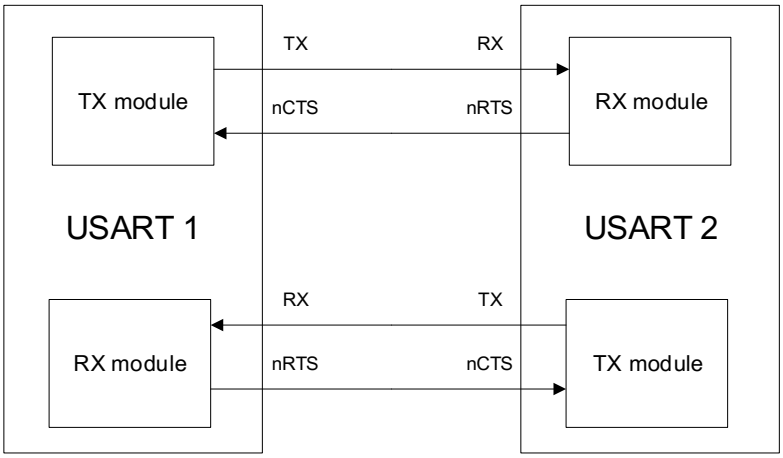


When the number of the data received by USART reaches the DMA transfer number, an end of transfer interrupt can be generated in the DMA module.

19.3.6. 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 19-7. Hardware flow control between two USARTs



RTS flow control

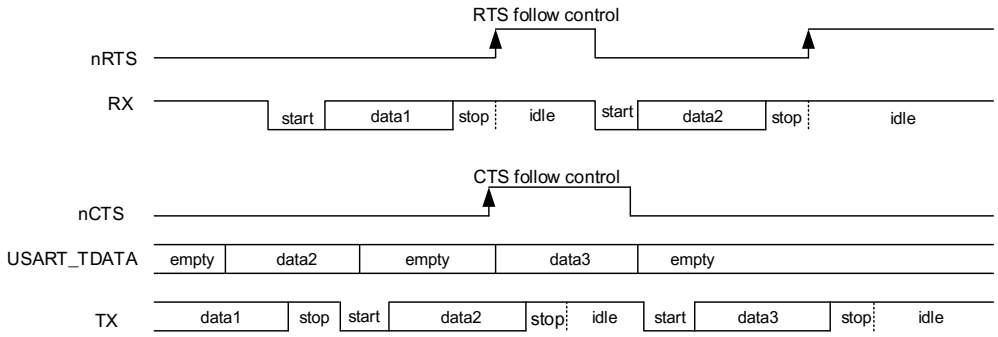
The USART receiver outputs the nRTS, which reflects the status of the receive buffer. When

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.

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_STAT 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.

Figure19-8. Hardware flow control



RS485 Driver Enable

The driver enable feature, which is enabled by setting bit DEM in the USART_CTL2 control register, allows the user to activate the external transceiver control, through the DE (Driver Enable) signal. The assertion time, which is programmed using the DEA [4:0] bits field in the USART_CTL0 control register, is the time between the activation of the DE signal and the beginning of the START bit. The de-assertion time, which is programmed using the DED [4:0] bits field in the USART_CTL0 control register, is the time between the end of the last stop bit and the de-activation of the DE signal. The polarity of the DE signal can be configured using the DEP bit in the USART_CTL2 control register.

19.3.7. 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 writing 1 to the MMCMD bit in USART_CMD register.

If a USART is in mute mode, all of the receive status bits cannot be set. The USART can also be wake 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 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 or 7 bits, which are configured by the ADDM bit of the USART_CTL1 register, of an address frame is the same as the ADDR 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_STAT register. If the LSB 4/7 bits of an address frame defers from the ADDR 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 ADDM=0, the LSB 4 bits of an address frame will be used to compare with ADDR[3:0] bits in the USART_CTL1 register.

If ADDM=1, PCEN=0 and WL=1, the LSB 8 bits of an address frame will be used to compare with ADDR[7:0] bits in the USART_CTL1 register. If ADDM=1, PCEN=1 and WL=0, the LSB 6 bits of an address frame will be used to compare with ADDR[5:0] bits in the USART_CTL1 register. If ADDM=1, PCEN=1 and WL=1, the LSB 7 bits of an address frame will be used to compare with ADDR[6:0] bits in the USART_CTL1 register. If ADDM=1, PCEN=0 and WL=0, the LSB 7 bits of an address frame will be used to compare with ADDR[6:0] bits in the USART_CTL1 register.

19.3.8. LIN mode

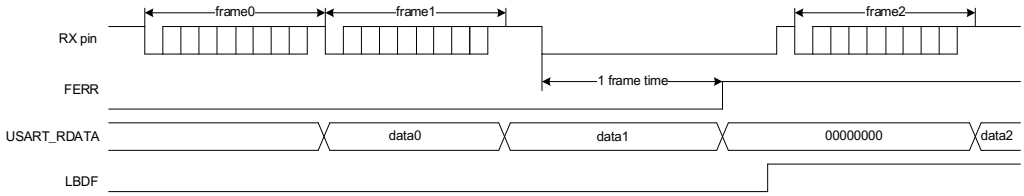
The local interconnection network mode is enabled by setting the LMEN bit in USART_CTL1. The CKEN, STB[1:0] bit in USART_CTL1 and the SCEN, HDEN, IREN bits in USART_CTL2 should be cleared in LIN mode.

When transmitting a normal data frame, the transmission procedure is the same as the normal USART mode. The data bits length can only be 8. And the break frame is 13-bit '0', 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 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_STAT is set. An interrupt occurs if the LBDIE bit in USART_CTL1 is set.

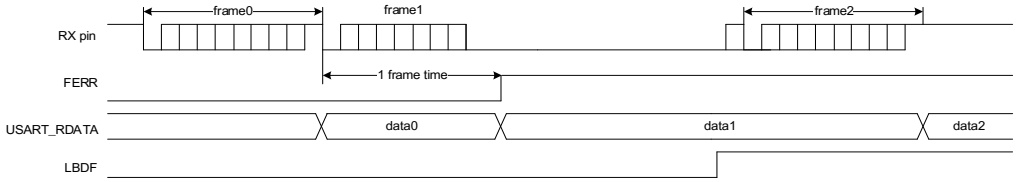
As shown in ***Figure 19-9. Break frame occurs during idle state***, if a break frame occurs during the idle state on the RX pin, the USART receiver will receive an all '0' frame, with an asserted FERR status.

Figure 19-9. Break frame occurs during idle state



As shown in ***Figure 19-10. 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 19-10. Break frame occurs during a frame



19.3.9. 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 clock output is also not activated during idle and break frame sending. 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).

The clock is synchronized with the data transmitted. The receiver in synchronous mode samples the data on the transmitter clock without any oversampling.

Figure 19-11. Example of USART in synchronous mode

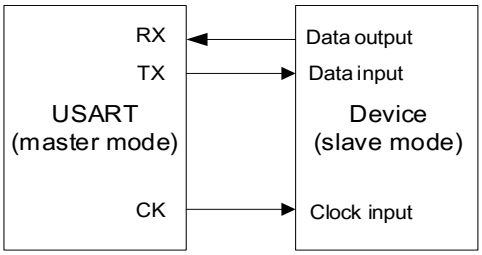
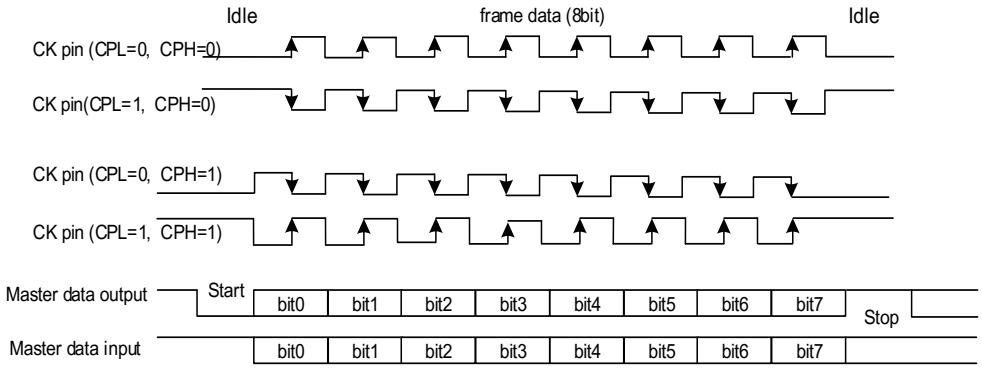


Figure 19-12. 8-bit format USART synchronous waveform (CLEN=1)

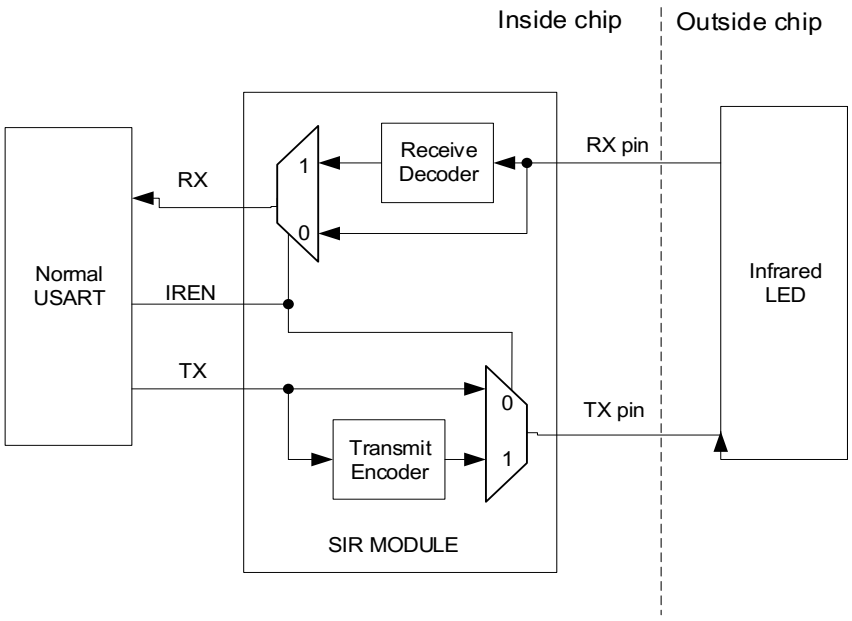


19.3.10. IrDA SIR ENDEC mode

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 19-13. IrDA SIR ENDEC module

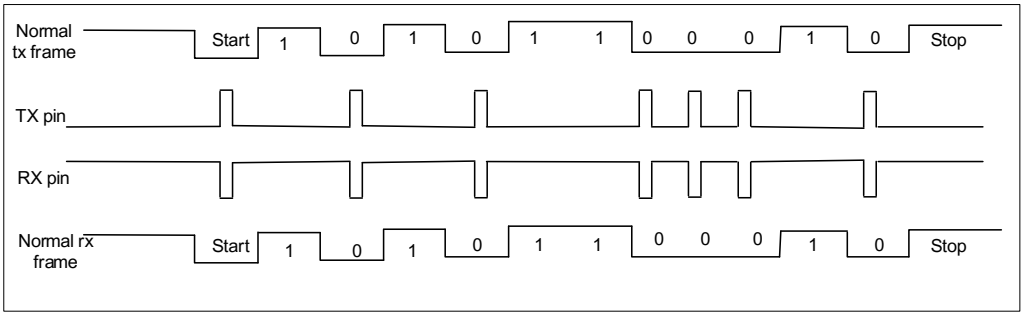


In IrDA mode, the polarity of the TX and RX pins is different. The TX pin is usually at low state, while the RX pin is usually at high state. The IrDA pins keep stable to represent the logic '1', while an infrared light pulse on the IrDA pins (a Return to Zero signal) represents the logic '0'. The pulse 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 19-14. 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.

19.3.11. 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 TX and RX pins are connected together internally. The TX pin should be configured as IO pin. The conflicts should be controlled by the software. When the TEN bit is set, the data in the data register will be sent.

19.3.12. 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 reset in smartcard mode.

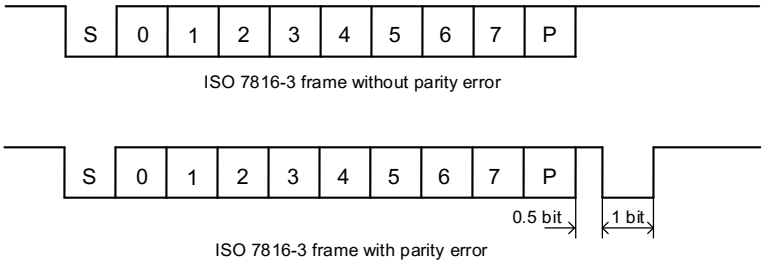
A clock is provided to the smartcard if the CKEN bit is set. The clock can be divided for other use.

The frame consists of 1 start bit, 9 data bits (1 parity bit included) and 1.5 stop bits.

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 drives a bidirectional line

that is also driven by the smartcard.

Figure 19-15. 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. A timeout

interrupt will be generated, if no answer is received from the card before the expiration of this period. If the first character is received before the expiration of the period, it is signaled by the RBNE interrupt. If DMA is used to read from the smartcard in block mode, the DMA must be enabled only after the first character is received.

In order to allow the automatic check of the maximum wait time between two consecutive characters, the USART_RT register must be programmed to the CWT (character wait time) - 11 value, which is expressed in baudtime units, after the reception of the first character (RBNE interrupt). The USART signals to the software through the RT flag and interrupt (when RTIE bit is set), if the smartcard doesn't send a new character in less than the CWT period after the end of the previous character.

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). This register field must be programmed to the minimum value (0x0), before the start of the block, when using DMA mode. With this value, an interrupt is generated after the 4th received character. The software must read the third byte as block length from the receive buffer.

In interrupt driven receive mode, the length of the block may be checked by software or by programming the BL value. However, before the start of the block, the maximum value of BL (0xFF) may be programmed. The real value will be programmed after the reception of the third character.

The total block length (including prologue, epilogue and information fields) equals BL+4. The end of the block is signaled to the software through the EBF flag and interrupt (when EBIE bit is set). The RT interrupt may occur in case of an error in the block length.

Direct and inverse convention

The smartcard protocol defines two conventions: direct and inverse.

The direct convention is defined as: LSB first, logical bit value of 1 corresponds to H state of the line and parity is even. In this case, the following control bits must be programmed: MSBF=0, DINV=0 (default values).

The inverse convention is defined as: MSB first, logical bit value 1 corresponds to an L state on the signal line and parity is even. In this case, the following control bits must be programmed: MSBF=1, DINV=1.

19.3.13. Auto baudrate detection

The USART is able to detect and automatically set the USART_BAUD register value based on the reception of one character. The auto baudrate detection is enabled by writing '1' to the ABDEN bit in USART_CTL1. There are two methods which can be chosen through the ABDM bits in the USART_CTL1 register. These methods are:

1. The USART will measure the duration of the start bit (falling edge to rising edge). In this case the receiving pattern should be any character starting with a bit at 1.
2. The USART will measure the duration of the start and of the 1st data bit. The measure is done falling edge to falling edge, ensuring a better accuracy in the case of slow signal slopes. In this case, the receiving pattern should be any character starting with 10xx bits.

After the baud rate detection is completed, the ABDF bit in the USART_STAT register will be set by hardware. Writing 1 to the ABDCMD bit in the USART_CMD register clears both the ABDF and ABDE bits, and initiates a new baud rate detection.

If the communication rate is out of range, the ABDE bit in the USART_STAT register will be set by hardware. The baud rate detection range is: bit duration between 16 and 65536 clock periods (oversampling by 16) and between 8 and 32768 clock periods (oversampling by 8).

19.3.14. ModBus communication

The USART offers basic support for the implementation of ModBus/RTU and ModBus/ASCII protocols by implementing an end of block detection.

In the ModBus/RTU mode, the end of one block is recognized by an idle line for more than 2 characters time. This function is implemented through the programmable timeout function.

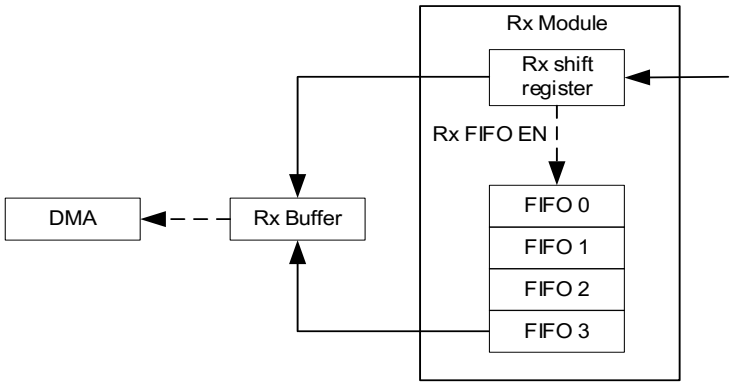
To detect the idle line, the RTEN bit in the USART_CTL1 register and the RTIE in the USART_CTL0 register must be set. The USART_RT register must be set to the value corresponding to a timeout of 2 characters time. After the last stop bit is received, when the receive line is idle for this duration, an interrupt will be generated, informing the software that the current block reception is completed.

In the ModBus/ASCII mode, the end of a block is recognized by a specific (CR/LF) character sequence. The USART manages this mechanism using the character match function by programming the LF ASCII code in the ADDR field and activating the address match interrupt (AMIE=1). When a LF has been received or can check the CR/LF in the DMA buffer, the software will be informed.

19.3.15. Receive FIFO

The receive FIFO can be enabled by setting the RFEN bit of the USART_RFCS register to avoid the overrun error when the CPU can't serve the RBNE interrupt immediately. Up to 5 frames receive data can be stored in the receive FIFO and receive buffer. The RFFINT flag will be set when the receive FIFO is full. An interrupt is generated if the RFFIE bit is set.

Figure 19-16. USART receive FIFO structure



If the software read receive data buffer in the routing of the RBNE interrupt, the RBNEIE bit should be reset at the beginning of the routing and set after all of the receive data is read out. The PERR/NERR/FERR/EBF flags should be cleared before reading a receive data out.

19.3.16. Wakeup from deep-sleep mode

The USART is able to wake up the MCU from deep-sleep mode by the standard RBNE interrupt or the WUM interrupt.

The UESM bit must be set and the USART clock must be set to IRC8M (refer to the reset and clock unit RCU section).

When using the standard RBNE interrupt, the RBNEIE bit must be set before entering deep-sleep mode.

When using the WUIE interrupt, the source of WUIE interrupt may be selected through the WUM bit fields.

DMA must be disabled before entering deep-sleep mode. Before entering deep-sleep mode, software must check that the USART is not performing a transfer, by checking the BSY flag in the USART_STAT register. The REA bit must be checked to ensure the USART is actually enabled.

When the wakeup event is detected, the WUF flag is set by hardware and a wakeup interrupt is generated if the WUIE bit is set, independently of whether the MCU is in stop or active mode.

19.3.17. USART interrupts

The USART interrupt events and flags are listed in *Table 19-3. USART interrupt requests*.

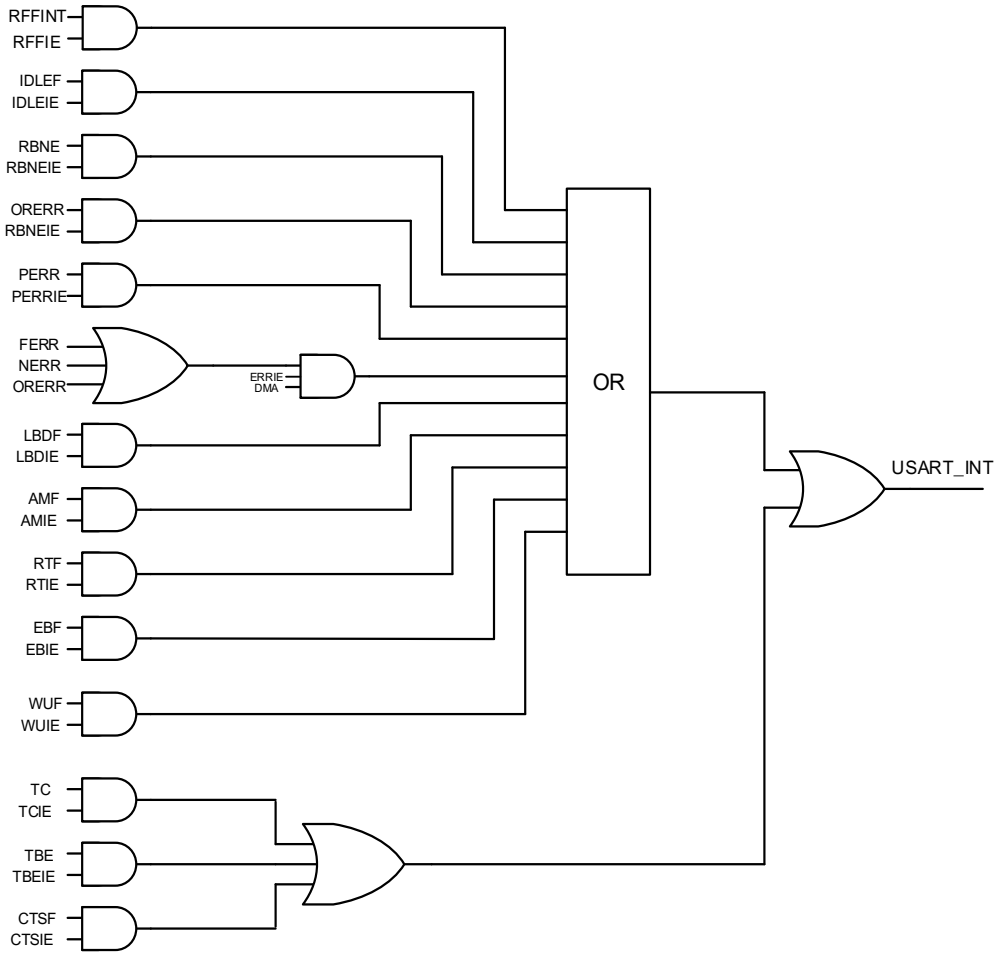
Table 19-3. USART interrupt requests

Interrupt event	Event flag	Enable Control bit
Transmit data register empty	TBE	TBEIE
CTS flag	CTSF	CTSIE

Interrupt event	Event flag	Enable Control bit
Transmission complete	TC	TCIE
Received data ready to be read	RBNE	RBNEIE
Overrun error detected	ORERR	
Receive FIFO full	RFFINT	RFFIE
Idle line detected	IDLEF	IDLEIE
Parity error flag	PERR	PERRIE
Break detected flag in LIN mode	LBDF	LBDIE
Reception errors (noise flag, overrun error, framing error) in DMA reception	NERR or ORERR or FERR	ERRIE
Character match	AMF	AMIE
Receiver timeout error	RTF	RTIE
End of block	EBF	EBIE
Wakeup from deep-sleep mode	WUF	WUIE

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 19-17. USART interrupt mapping diagram



19.4. Register definition

USART0 base address: 0x4001 3800

UART1 base address: 0x4000 4400

19.4.1. Control register 0 (USART_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				EBIE	RTIE	DEA[4:0]					DED[4:0]				
				rw	rw	rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVSMOD	AMIE	MEN	WL	WM	PCEN	PM	PERRIE	TBEIE	TCIE	RBNEIE	IDLEIE	TEN	REN	UESM	UEN
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	EBIE	End of Block interrupt enable. 0: End of Block interrupt is disabled. 1: End of Block interrupt is enabled. This bit is reserved in UART1.
26	RTIE	Receiver timeout interrupt enable. 0: Receiver timeout interrupt is disabled. 1: Receiver timeout interrupt is enabled. This bit is reserved in UART1.
25:21	DEA[4:0]	Driver Enable assertion time These bits are used to define the time between the activation of the DE (driver enable) signal and the beginning of the start bit. It is expressed in sample time units (1/8 or 1/16 bit time), which are configured by the OVSMOD bit. This bit field cannot be written when the USART is enabled (UEN=1).
20:16	DED[4:0]	Driver enable de-assertion time. These bits are used to define the time between the end of the last stop bit, in a transmitted message, and the de-activation of the DE (Driver Enable) signal. It is expressed in sample time units (1/8 or 1/16 bit time), which are configured by the OVSMOD bit. This bit field cannot be written when the USART is enabled (UEN=1).
15	OVSMOD	Oversample mode

		0: Oversampling by 16. 1: Oversampling by 8. This bit must be kept cleared in LIN, IrDA and smartcard modes. This bit field cannot be written when the USART is enabled (UEN=1).
14	AMIE	ADDR match interrupt enable. 0: ADDR match interrupt is disabled. 1: ADDR match interrupt is enabled.
13	MEN	Mute mode enable. 0: Mute mode disabled. 1: Mute mode enabled.
12	WL	Word length 0: 8 data bits 1: 9 data bits This bit field cannot be written when the USART is enabled (UEN=1).
11	WM	Wakeup method in mute mode 0: Idle line 1: Address match This bit field cannot be written when the USART is enabled (UEN=1).
10	PCEN	Parity control enable. 0: Parity control disabled. 1: Parity control enabled. This bit field cannot be written when the USART is enabled (UEN=1).
9	PM	Parity mode 0: Even parity 1: Odd parity This bit field cannot be written when the USART is enabled (UEN=1).
8	PERRIE	Parity error interrupt enable. 0: Parity error interrupt is disabled. 1: An interrupt will occur whenever the PERR bit is set in USART_STAT.
7	TBEIE	Transmitter register empty interrupt enable. 0: Interrupt is inhibited. 1: An interrupt will occur whenever the TBE bit is set in USART_STAT.
6	TCIE	Transmission complete interrupt enable. If this bit is set, an interrupt occurs when the TC bit in USART_STAT 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. 0: Read data register not empty interrupt and overrun error interrupt disabled.

		1: An interrupt will occur whenever the ORERR bit is set or the RBNE bit is set in USART_STAT.
4	IDLEIE	IDLE line detected interrupt enable. 0: IDLE line detected interrupt disabled. 1: An interrupt will occur whenever the IDLEF bit is set in USART_STAT.
3	TEN	Transmitter enable. 0: Transmitter is disabled. 1: Transmitter is enabled.
2	REN	Receiver enable 0: Receiver is disabled. 1: Receiver is enabled and begins searching for a start bit.
1	UESM	USART enable in Deep-sleep mode. 0: USART not able to wake up the MCU from Deep-sleep mode. 1: USART able to wake up the MCU from Deep-sleep mode. Providing that the clock source for the USART must be IRC8M. This bit is reserved in UART1.
0	UEN	USART enable 0: USART prescaler and outputs disabled. 1: USART prescaler and outputs enabled.

19.4.2. Control register 1 (USART_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
ADDR[7:0]								RTEN	ABDM[1:0]		ABDEN	MSBF	DINV	TINV	RINV
rw								rw	rw		rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STRP	LMEN	STB[1:0]		CKEN	CPL	CPH	CLEN	Reserved	LBDIE	LBLEN	ADDM	Reserved			
rw	rw	rw		rw	rw	rw	rw		rw	rw	rw				

Bits	Fields	Descriptions
31:24	ADDR[7:0]	Address of the USART terminal. These bits give the address of the USART terminal. In multiprocessor communication during mute mode or deep-sleep mode, this is used for wakeup with address match detection. The received frame, the MSB of which is equal to 1, will be compared to these bits. When the ADDM bit is reset, only the ADDR[3:0] bits are used to compare. In normal reception, these bits are also used for character detection. The whole

		received character (8-bit) is compared to the ADDR[7:0] value and AMF flag is set on matching. This bit field cannot be written when both reception (REN=1) and USART (UEN=1) are enabled.
23	RTEN	Receiver timeout enable. 0: Receiver timeout function disabled. 1: Receiver timeout function enabled. This bit is reserved in UART1.
22:21	ABDM[1:0]	Auto baud rate mode. 00: Falling edge to rising edge measurement (measurement of the start bit). 01: Falling edge to falling edge measurement (the received frame must be in a start 10xxxxxx frame format). 10: Reserved. 11: Reserved. This bitfield can only be written when ABDEN = 0 or the USART is disabled (UE = 0). These bits are reserved in UART1.
20	ABDEN	Auto baud rate enable. 0: Auto baud rate detection is disabled. 1: Auto baud rate detection is enabled. This bit is reserved in UART1.
19	MSBF	Most significant bit first. 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).
18	DINV	Data bit level inversion. 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).
17	TINV	TX pin level inversion. 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).
16	RINV	RX pin level inversion. 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).
15	STRP	Swap TX/RX pins. 0: The TX and RX pins functions are not swapped.

		1: The TX and RX pins functions are swapped. This bit field cannot be written when the USART is enabled (UEN=1).
14	LMEN	LIN mode enable. 0: LIN mode disabled. 1: LIN mode enabled. This bit field cannot be written when the USART is enabled (UEN=1). This bit is reserved in UART1.
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 This bit field cannot be written when the USART is enabled (UEN=1).
11	CKEN	CK pin enable. 0: CK pin disabled. 1: CK pin enabled. This bit field cannot be written when the USART is enabled (UEN=1). This bit is reserved in UART1.
10	CPL	Clock polarity. 0: Steady low value on CK pin outside transmission window in synchronous mode. 1: Steady high value on CK pin outside transmission window in synchronous mode. This bit field cannot be written when the USART is enabled (UEN=1). This bit is reserved in UART1.
9	CPH	Clock phase. 0: The first clock transition is the first data capture edge in synchronous mode. 1: The second clock transition is the first data capture edge in synchronous mode. This bit field cannot be written when the USART is enabled (UEN=1). This bit is reserved in UART1.
8	CLEN	CK length. 0: The clock pulse of the last data bit (MSB) is not output to the CK pin in synchronous mode. 1: The clock pulse of the last data bit (MSB) is output to the CK pin in synchronous mode. This bit field cannot be written when the USART is enabled (UEN=1). This bit is reserved in UART1.
7	Reserved	Must be kept at reset value.
6	LBDIE	LIN break detection interrupt enable. 0: LIN break detection interrupt is disabled.

		1: An interrupt will occur whenever the LBDF bit is set in USART_STAT. This bit is reserved in UART1.
5	LBLEN	LIN break frame length. 0: 10 bit break detection. 1: 11 bit break detection. This bit field cannot be written when the USART is enabled (UEN=1). This bit is reserved in UART1.
4	ADDM	Address detection mode. This bit is used to select between 4-bit address detection and full-bit address detection. 0: 4-bit address detection 1: Full-bit address detection. In 7-bit, 8-bit and 9-bit data modes, the address detection is done on 6-bit, 7-bit and 8-bit address (ADDR[5:0], ADDR[6:0] and ADDR[7:0]) respectively This bit field cannot be written when the USART is enabled (UEN=1).
3:0	Reserved	Must be kept at reset value.

19.4.3. Control register 2 (USART_CTL2)

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									WUIE	WUM[1:0]		SCRTNUM[2:0]			Reserved
									rw	rw		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEP	DEM	DDRE	OVRD	OSB	CTSIE	CTSEN	RTSEN	DENT	DENR	SCEN	NKEN	HDEN	IRLP	IREN	ERRIE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22	WUIE	Wakeup from deep-sleep mode interrupt enable. 0: Wakeup from deep-sleep mode interrupt is disabled. 1: Wakeup from deep-sleep mode interrupt is enabled. This bit is reserved in UART1.
21:20	WUM[1:0]	Wakeup mode from deep-sleep mode. These bits are used to specify the event which activates the WUF (wakeup from deep-sleep mode flag) in the USART_STAT register. 00: WUF active on address match, which is defined by ADDR and ADDM. 01: Reserved.

		10: WUF active on start bit. 11: WUF active on RBNE. This bit field cannot be written when the USART is enabled (UEN=1). This bit is reserved in UART1.
19:17	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 transmission error (FERR bit set) will occur after this number of automatic retransmission retries. In reception mode, reception error (RBNE and PERR bits set) will occur after this number or erroneous reception trials. When these bits are configured as 0x0, there will be no automatic retransmission in transmit mode. This bit field is only can be cleared to 0 when the USART is enabled (UEN=1), to stop retransmission. This bit is reserved in UART1.
16	Reserved	Must be kept at reset value.
15	DEP	Driver enable polarity mode. 0: DE signal is active high. 1: DE signal is active low. This bit field cannot be written when the USART is enabled (UEN=1).
14	DEM	Driver enable mode This bit is used to activate the external transceiver control, through the DE signal, which is output on the RTS pin. 0: DE function is disabled. 1: DE function is enabled. This bit field cannot be written when the USART is enabled (UEN=1).
13	DDRE	Disable DMA on reception error. 0: DMA is not disabled in case of reception error. The DMA request is not asserted to make sure the erroneous data is not transferred, but the next correct received data will be transferred. The RBNE is kept 0 to prevent overrun, but the corresponding error flag is set. This mode can be used in smartcard mode. 1: DMA is disabled following a reception error. The DMA request is not asserted until the error flag is cleared. The RBNE flag and corresponding error flag will be set. The software must first disable the DMA request (DMAR = 0) or clear RBNE before clearing the error flag. This bit field cannot be written when the USART is enabled (UEN=1).
12	OVRD	Overrun disable. 0: Overrun functionality is enabled. The ORERR error flag will be set when received data is not read before receiving new data, and the new data will be lost.

		1: Overrun functionality is disabled. The ORERR error flag will not be set when received data is not read before receiving new data, and the new received data overwrites the previous content of the USART_RDATA register. This bit field cannot be written when the USART is enabled (UEN=1).
11	OSB	One sample bit method. 0: Three sample bit method. 1: One sample bit method. This bit field cannot be written when the USART is enabled (UEN=1).
10	CTSIE	CTS interrupt enable. 0: CTS interrupt is disabled. 1: An interrupt will occur whenever the CTS bit is set in USART_STAT.
9	CTSEN	CTS enable 0: CTS hardware flow control disabled. 1: CTS hardware flow control enabled. This bit field cannot be written when the USART is enabled (UEN=1).
8	RTSEN	RTS enable 0: RTS hardware flow control disabled. 1: RTS hardware flow control enabled, data can be requested only when there is space in the receive buffer. This bit field cannot be written when the USART is enabled (UEN=1).
7	DENT	DMA enable for transmission. 0: DMA mode is disabled for transmission. 1: DMA mode is enabled for transmission.
6	DENR	DMA enable for reception. 0: DMA mode is disabled for reception. 1: DMA mode is enabled for reception.
5	SCEN	Smartcard mode enable. 0: Smartcard mode disabled. 1: Smartcard mode enabled. This bit field cannot be written when the USART is enabled (UEN=1). This bit is reserved in UART1.
4	NKEN	NACK enable in smartcard mode. 0: Disable NACK transmission when parity error. 1: Enable NACK transmission when parity error. This bit field cannot be written when the USART is enabled (UEN=1). This bit is reserved in UART1.
3	HDEN	Half-duplex enable. 0: Half duplex mode is disabled. 1: Half duplex mode is enabled.

		This bit field cannot be written when the USART is enabled (UEN=1).
2	IRLP	IrDA low-power 0: Normal mode. 1: Low-power mode. This bit field cannot be written when the USART is enabled (UEN=1).
1	IREN	IrDA mode enable 0: IrDA disabled 1: IrDA enabled This bit field cannot be written when the USART is enabled (UEN=1). This bit is reserved in UART1.
0	ERRIE	Error interrupt enable. 0: Error interrupt disabled. 1: An interrupt will occur whenever the FERR bit or the ORERR bit or the NERR bit is set in USART_STAT in multibuffer communication.

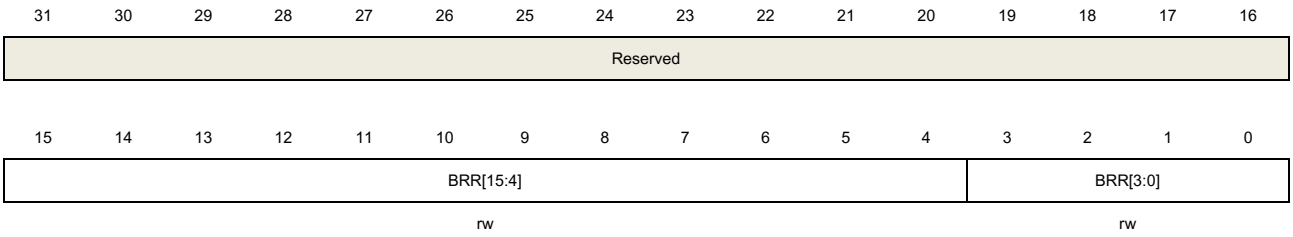
19.4.4. Baud rate generator register (USART_BAUD)

Address offset: 0x0C

Reset value: 0x0000 0000

This register cannot be written when the USART is enabled (UEN=1).

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



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:4	BRR[15:4]	Integer of baud-rate divider INTDIV = BRR[15:4]
3:0	BRR[3:0]	Fraction of baud-rate divider If OVSMOD = 0, FRADIV= BRR[3:0]. If OVSMOD = 1, FRADIV = BRR[2:0], BRR [3] must be reset.

19.4.5. Prescaler and guard time configuration register (USART_GP)

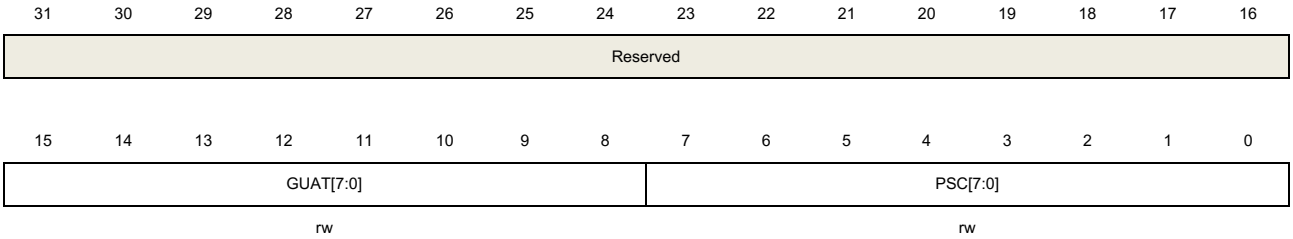
Address offset: 0x10

Reset value: 0x0000 0000

This register cannot be written when the USART is enabled (UEN=1).

This register is reserved in UART1.

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



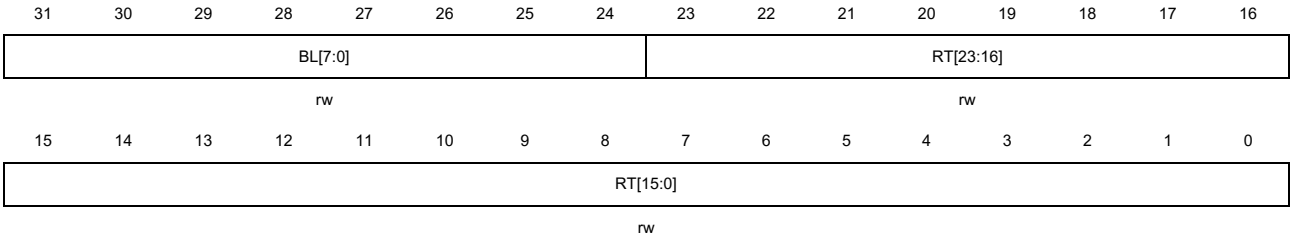
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:8	GUAT[7:0]	Guard time value in smartcard mode. This bit field cannot be written when the USART is enabled (UEN=1).
7:0	PSC[7:0]	Prescaler value for dividing the system clock. In IrDA Low-power mode, the division factor is the prescaler value. 00000000: Reserved - do not program this value. 00000001: divides the source clock by 1. 00000010: divides the source clock by 2. ... In IrDA normal mode, 00000001: can be set this value only. In smartcard mode, the prescaler value for dividing the system clock is stored in PSC[4:0] bits. And the bits of PSC[7:5] must be kept at reset value. The division factor is twice as the prescaler value. 00000: Reserved - do not program this value. 00001: divides the source clock by 2. 00010: divides the source clock by 4. 00011: divides the source clock by 6. ... This bit field cannot be written when the USART is enabled (UEN=1).

19.4.6. Receiver timeout register (USART_RT)

Address offset: 0x14
Reset value: 0x0000 0000

This bit is reserved in UART1.

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



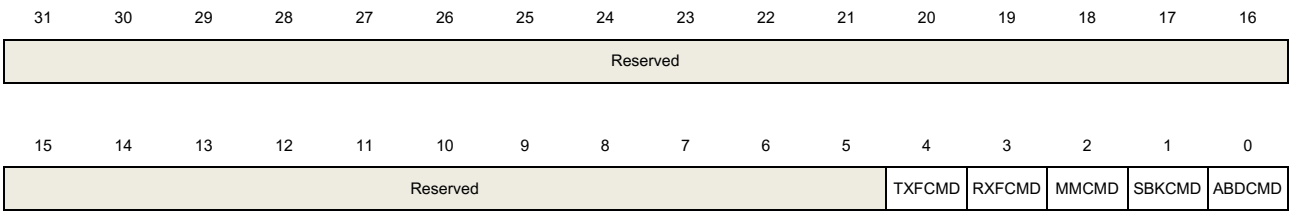
Bits	Fields	Descriptions
31:24	BL[7:0]	Block Length These bits specify the block length in smartcard T=1 reception. Its value equals the number of information characters + the length of the Epilogue Field (1-LEC/2-CRC) - 1. This value, which must be programmed only once per received block, can be programmed 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) and/or when the EBC bit is written to 1, 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. In standard mode, the RTF flag is set if no new start bit is detected for more than the RT value after the last received character. In smartcard mode, 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 the new value is lower than or equal to the counter. These bits must only be programmed once per received character. **Note:** When oversampling by 16, RT[23:0] cannot be configured as 0xFFFFFF.

19.4.7. Command register (USART_CMD)

Address offset: 0x18

Reset value: 0x0000 0000

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



w w w w w

Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	TXFCMD	Transmit data flush request. Writing 1 to this bit sets the TBE flag, to discard the transmit data. This bit is reserved in UART1.
3	RXFCMD	Receive data flush command. Writing 1 to this bit clears the RBNE flag to discard the received data without reading it.
2	MMCMD	Mute mode command. Writing 1 to this bit makes the USART into mute mode and sets the RWU flag.
1	SBKCMD	Send break command. Writing 1 to this bit sets the SBKF flag and makes the USART send a BREAK frame, as soon as the transmit machine is idle.
0	ABDCMD	Auto baudrate detection request. Writing 1 to this bit issues an automatic baud rate measurement command on the next received data frame and resets the ABDF and ABDE flags in the USART_STAT. This bit is reserved in UART1.

19.4.8. Status register (USART_STAT)

Address offset: 0x1C

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									REA	TEA	WUF	RWU	SBF	AMF	BSY
									r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABDF	ABDE	Reserved	EBF	RTF	CTS	CTSF	LBDF	TBE	TC	RBNE	IDLEF	ORERR	NERR	FERR	PERR
r	r		r	r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22	REA	Receive enable acknowledge flag. This bit, which is set/reset by hardware, reflects the receive enable state of the USART core logic. 0: The USART core receiving logic has not been enabled.

		1: The USART core receiving logic has been enabled. This bit is reserved in UART1.
21	TEA	Transmit enable acknowledge flag. This bit, which is set/reset by hardware, reflects the transmit enable state of the USART core logic. 0: The USART core transmitting logic has not been enabled. 1: The USART core transmitting logic has been enabled.
20	WUF	Wakeup from deep-sleep mode flag. 0: No wakeup from deep-sleep mode. 1: Wakeup from deep-sleep mode. An interrupt is generated if WUFIE=1 in the USART_CTL2 register and the MCU is in Deep-sleep mode. This bit is set by hardware when a wakeup event, which is defined by the WUM bit field, is detected. Cleared by writing a 1 to the WUC in the USART_INTC register. This bit can also be cleared when UESM is cleared. This bit is reserved in UART1.
19	RWU	Receiver wakeup from mute mode This bit is used to indicate if the USART is in mute mode. 0: Receiver in active mode 1: Receiver in mute mode It is cleared/set by hardware when a wakeup/mute sequence (address or IDLEIE) is recognized, which is selected by the WM bit in the USART_CTL0 register. This bit can only be set by writing 1 to the MMCMD bit in the USART_CMD register when wakeup on IDLEIE mode is selected.
18	SBF	Send break flag 0: No break character is transmitted 1: Break character will be transmitted This bit indicates that a send break character was requested. Set by software, by writing 1 to the SBKCMD bit in the USART_CMD register. Cleared by hardware during the stop bit of break transmission.
17	AMF	ADDR match flag 0: ADDR does not match the received character. 1: ADDR matches the received character, an interrupt is generated if AMIE=1 in the USART_CTL0 register. Set by hardware, when the character defined by ADDR [7:0] is received. Cleared by writing 1 to the AMC in the USART_INTC register.
16	BSY	Busy flag 0: USART reception path is idle. 1: USART reception path is working.

15	ABDF	Auto baudrate detection flag. 0: No auto baudrate detection complete. 1: Auto baudrate detection complete. Set by hardware when the automatic baud rate has been completed. Cleared by writing 1 to the ABDCMD(USART_CMD[0]) in the USART_CMD register, to request a new auto baudrate detection. This bit is reserved in UART1.
14	ABDE	Auto baudrate detection error. 0: No auto baudrate detection error occurred. 1: Auto baudrate detection error occurred. Set by hardware if the baud rate out of range. Cleared by software, by writing 1 to the ABDCMD(USART_CMD[0]) bit in the USART_CMD register. This bit is reserved in UART1.
13	Reserved	Must be kept at reset value.
12	EBF	End of block flag. 0: End of block not reached. 1: End of block (number of characters) reached. An interrupt is generated if the EBIE=1 in the USART_CTL1 register. Set by hardware when the number of received bytes (from the start of the block, including the prologue) is equal or greater than BLEN + 4. Cleared by writing 1 to EBC bit in USART_INTC register. This bit is reserved in UART1.
11	RTF	Receiver timeout flag. 0: Timeout value not reached. 1: Timeout value reached without any data reception. An interrupt is generated if RTIE bit in the USART_CTL1 register is set. Set by hardware when the RT value, programmed in the USART_RT register has lapsed without any communication. Cleared by writing 1 to RTC bit in USART_INTC register. The timeout corresponds to the CWT or BWT timings in smartcard mode. This bit is reserved in UART1.
10	CTS	CTS level This bit equals to the inverted level of the nCTS input pin. 0: nCTS input pin is in high level. 1: nCTS input pin is in low level. This bit is reserved in UART1.
9	CTSF	CTS change flag. 0: No change occurred on the nCTS status line. 1: A change occurred on the nCTS status line. An interrupt will occur if the CTSIE

		bit is set in USART_CTL2. Set by hardware when the nCTS input toggles. Cleared by writing 1 to CTSC bit in USART_INTC register. This bit is reserved in UART1.
8	LBDF	LIN break detected flag. 0: LIN break is not detected. 1: LIN break is detected. An interrupt will occur if the LBDIE bit is set in USART_CTL1. Set by hardware when the LIN break is detected. Cleared by writing 1 to LBDC bit in USART_INTC register. This bit is reserved in UART1.
7	TBE	Transmit data register empty. 0: Data is not transferred to the shift register. 1: Data is transferred to the shift register. An interrupt will occur if the TBEIE bit is set in USART_CTL0. Set by hardware when the content of the USART_TDATA register has been transferred into the transmit shift register or writing 1 to TXFCMD bit of the USART_CMD register. Cleared by a write to the USART_TDATA.
6	TC	Transmission completed 0: Transmission is not completed. 1: Transmission is complete. An interrupt will occur if the TCIE bit is set in USART_CTL0. Set by hardware if the transmission of a frame containing data is completed and if the TBE bit is set. Cleared by writing 1 to TCC bit in USART_INTC register.
5	RBNE	Read data buffer not empty. 0: Data is not received. 1: Data is received and ready to be read. An interrupt will occur if the RBNEIE bit is set in USART_CTL0. Set by hardware when the content of the receive shift register has been transferred to the USART_RDATA. Cleared by reading the USART_RDATA or writing 1 to RXFCMD bit of the USART_CMD register.
4	IDLEF	IDLE line detected flag. 0: No idle line is detected. 1: Idle line is detected. An interrupt will occur if the IDLEIE bit is set in USART_CTL0. Set by hardware when an idle line is detected. It will not be set again until the RBNE bit has been set itself.

Cleared by writing 1 to IDLEC bit in USART_INTC register.

3	ORERR	Overrun error 0: No overrun error is detected. 1: Overrun error is detected. An interrupt will occur if the RBNEIE bit is set in USART_CTL0. An interrupt will occur if the ERRIE bit is set in USART_CTL2. Set by hardware when the word in the receive shift register is ready to be transferred into the USART_RDATA register while the RBNE bit is set. Cleared by writing 1 to OREC bit in USART_INTC register.
2	NERR	Noise error flag 0: No noise error is detected. 1: Noise error is detected. An interrupt will occur if the ERRIE bit is set in USART_CTL2. Set by hardware when noise error is detected on a received frame. Cleared by writing 1 to NEC bit in USART_INTC register.
1	FERR	Frame error flag 0: No framing error is detected. 1: Frame error flag or break character is detected. An interrupt will occur if the ERRIE bit is set in USART_CTL2. Set by hardware when a de-synchronization, excessive noise or a break character is detected. This bit will be set when the maximum number of transmit attempts is reached without success (the card NACKs the data frame), when USART transmits in smartcard mode. Cleared by writing 1 to FEC bit in USART_INTC register.
0	PERR	Parity error flag. 0: No parity error is detected. 1: Parity error flag is detected. An interrupt will occur if the PERRIE bit is set in USART_CTL0. Set by hardware when a parity error occurs in receiver mode. Cleared by writing 1 to PEC bit in USART_INTC register.

19.4.9. Interrupt status clear register (USART_INTC)

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											WUC	Reserved		AMC	Reserved
											w			w	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			EBC	RTC	Reserved	CTSC	LBDC	Reserved	TCC	Reserved	IDLEC	OREC	NEC	FEC	PEC

w w w w w w w w w w

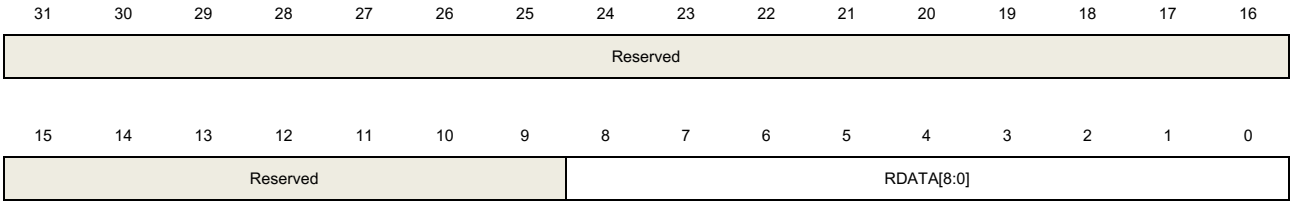
Bits	Fields	Descriptions
31:21	Reserved	Must be kept at reset value.
20	WUC	Wakeup from deep-sleep mode clear. Writing 1 to this bit clears the WUF bit in the USART_STAT register. This bit is reserved in UART1.
19:18	Reserved	Must be kept at reset value.
17	AMC	ADDR match clear. Writing 1 to this bit clears the AMF bit in the USART_STAT register.
16:13	Reserved	Must be kept at reset value.
12	EBC	End of block clear. Writing 1 to this bit clears the EBF bit in the USART_STAT register. This bit is reserved in UART1.
11	RTC	Receiver timeout clear. Writing 1 to this bit clears the RTF flag in the USART_STAT register. This bit is reserved in UART1.
10	Reserved	Must be kept at reset value.
9	CTSC	CTS change clear. Writing 1 to this bit clears the CTSF bit in the USART_STAT register.
8	LBDC	LIN break detected clear. Writing 1 to this bit clears the LBDF flag in the USART_STAT register. This bit is reserved in UART1.
7	Reserved	Must be kept at reset value.
6	TCC	Transmission complete clear. Writing 1 to this bit clears the TC bit in the USART_STAT register.
5	Reserved	Must be kept at reset value.
4	IDLEC	Idle line detected clear. Writing 1 to this bit clears the IDLEF bit in the USART_STAT register.
3	OREC	Overrun error clear. Writing 1 to this bit clears the ORERR bit in the USART_STAT register.
2	NEC	Noise detected clear. Writing 1 to this bit clears the NERR bit in the USART_STAT register.
1	FEC	Frame error flag clear. Writing 1 to this bit clears the FERR bit in the USART_STAT register.

0	PEC	Parity error clear. Writing 1 to this bit clears the PERR bit in the USART_STAT register.

19.4.10. Receive data register (USART_RDATA)

Address offset: 0x24
Reset value: Undefined

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

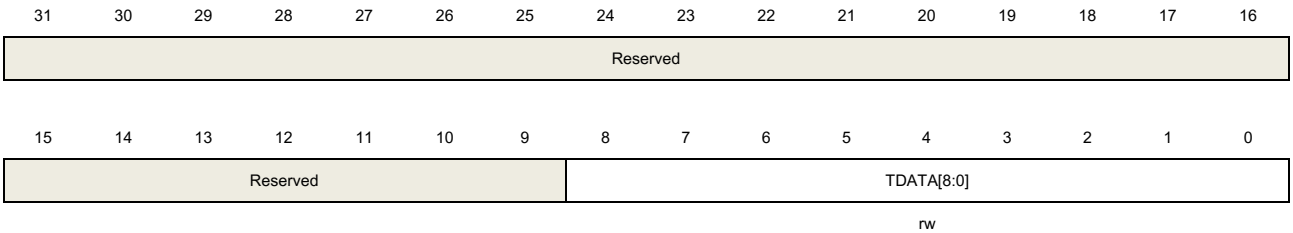


Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8:0	RDATA[8:0]	Receive data value. The received data character is contained in these bits. The value read in the MSB (bit 7 or bit 8 depending on the data length) will be the received parity bit, if receiving with the parity is enabled (PCEN bit set to 1 in the USART_CTL0 register).

19.4.11. Transmit data register (USART_TDATA)

Address offset: 0x28
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	TDATA[8:0]	Transmit data value. The transmit data character is contained in these bits. The value written in the MSB (bit 7 or bit 8 depending on the data length) will be replaced by the parity, when transmitting with the parity is enabled (PCEN bit set

to 1 in the USART_CTL0 register).

This register must be written only when TBE bit in USART_STAT register is set.

19.4.12. USART coherence control register (USART_CHC)

Address offset: 0xC0

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							EPERR	Reserved							HCM
							rc_w0								rw

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 the RBNE in status register. 1: nRTS signal is set when the last data bit (parity bit when pce is set) has been sampled.

19.4.13. USART receive FIFO control and status register (USART_RFCS)

Address offset: 0xD0

Reset value: 0x0000 0400

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
RFFINT	RFCNT[2:0]			RFF	RFE	RFFIE	RFEN	Reserved							ELNACK
r_w0	r			r	r	rw	rw								rw

Bits	Fields	Descriptions

31:16	Reserved	Must be kept at reset value.
15	RFFINT	Receive FIFO full interrupt flag.
14:12	RFCNT[2:0]	Receive FIFO counter number.
11	RFF	Receive FIFO full flag. 0: Receive FIFO not full. 1: Receive FIFO full.
10	RFE	Receive FIFO empty flag. 0: Receive FIFO not empty. 1: Receive FIFO empty.
9	RFFIE	Receive FIFO full interrupt enable. 0: Receive FIFO full interrupt disable. 1: Receive FIFO full interrupt enable.
8	RFEN	Receive FIFO enable. This bit can be set when UESM = 1. This bit field cannot be written when the USART is enabled (UEN = 1). 0: Receive FIFO disable. 1: Receive FIFO enable.
7:1	Reserved	Must be kept at reset value.
0	ELNACK	Early NACK when smartcard mode is selected. The NACK pulse occurs 1/16 bit time earlier when the parity error is detected. 0: Early NACK disable when smartcard mode is selected. 1: Early NACK enable when smartcard mode is selected. This bit is reserved in UART1.

20. Serial peripheral interface/Inter-IC sound (SPI / I2S)

20.1. 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 also supported in SPI1.

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.

20.2. Characteristics

20.2.1. SPI characteristics

- Master or slave operation with full-duplex or half-duplex or simplex mode.
- Separate transmission and reception buffer, 16 bits wide (only in SPI0).
- Separate transmission and reception 32-bit FIFO (only in SPI1).
- Data frame size can be 8 or 16 bits (only in SPI0).
- Data frame size can be 4 to 16 bits (only in SPI1).
- 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 SPI1).

20.2.2. I2S characteristics

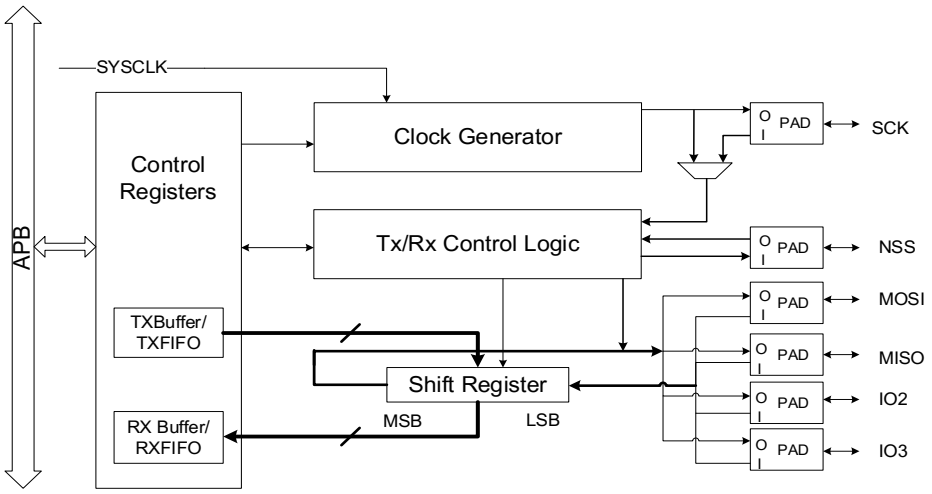
- Master or slave operation for transmission/reception.
- Four I2S standards supported: PHILIPS, MSB justified, LSB justified and PCM standard.
- Data length can be 16 bits, 24 bits or 32 bits.
- Channel length can be 16 bits or 32 bits.
- Transmission and reception using a 16 bits wide buffer.
- Audio 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.

20.3. SPI function overview

20.3.1. SPI block diagram

Figure 20-1. Block diagram of SPI



20.3.2. SPI signal description

Normal configuration (not Quad-SPI mode)

Table 20-1. 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

Pin name	Direction	Description
		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 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 SPI1). Quad-SPI mode can only work in master mode.

The SPI is connected to external devices through 6 pins in Quad-SPI mode:

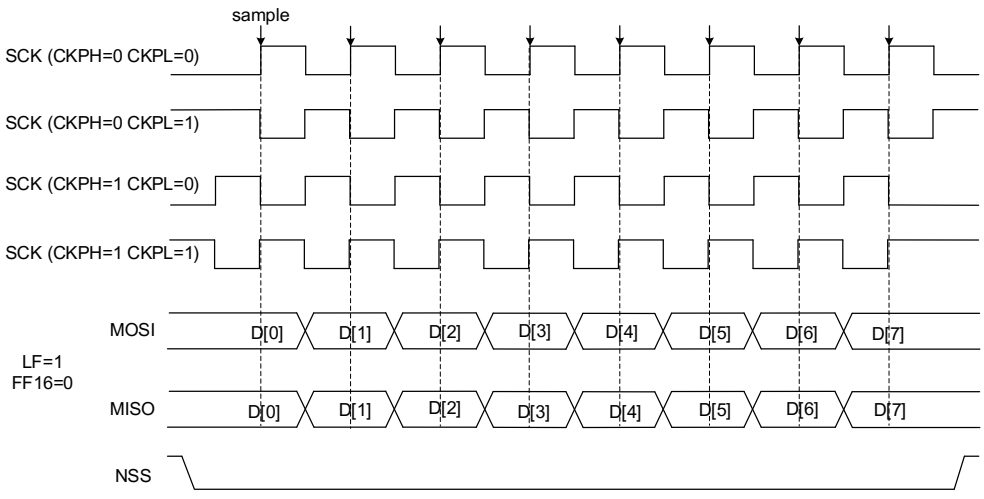
Table 20-2. Quad-SPI signal description

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

20.3.3. 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 20-2. SPI0 timing diagram in normal mode



In SPI0 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.

Data order is configured by LF bit in SPI_CTL0 register, and SPI will send the LSB first if LF=1, or the MSB if LF=0. The data order is fixed to MSB first in TI mode.

Figure 20-3. SPI1 timing diagram in normal mode

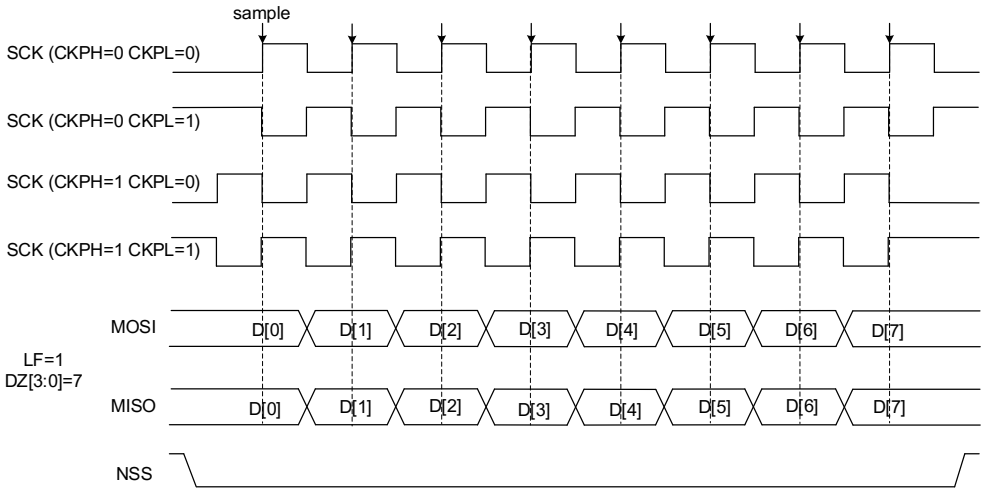
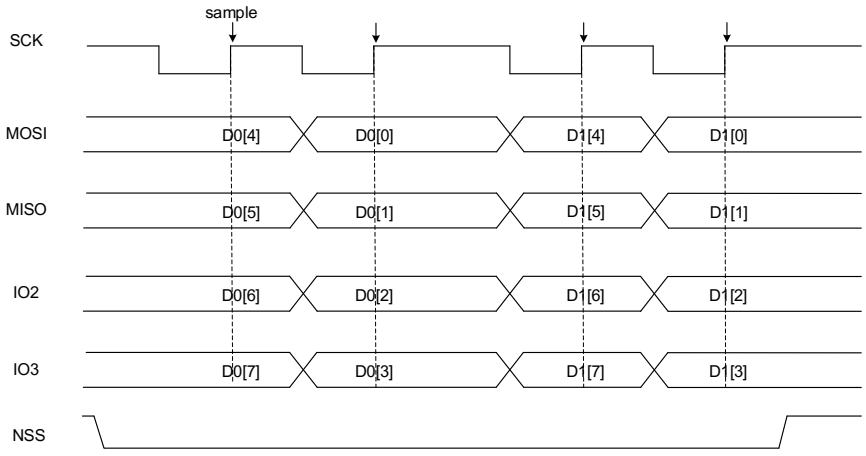


Figure 20-4. SPI timing diagram in Quad-SPI mode (CKPL=1, CKPH=1, LF=0)



In SPI1 normal mode, the length of data is configured by the DZ[3:0] bits in the SPI_CTL1 register. It can be set from 4-bit up to 16-bit length and the setting applies for both transmission and reception, and the read access to the FIFO must be aligned with the BYTEN bit in the SPI_CTL1 register. The data frame length is fixed to 8 bits in Quad-SPI mode.

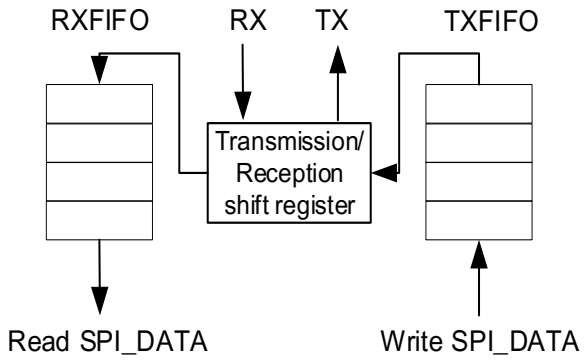
Data order is configured by LF bit in SPI_CTL0 register, and SPI will send the LSB first if LF=1, or the MSB if LF=0. The data order is fixed to MSB first in TI mode.

When the SPI_DATA register is accessed, data frames are always right-aligned into either a byte (If the data length is less than or equal to one byte) or a half-word. During communication, only bits within the data frame will be output with the clock.

20.3.4. Separate transmission and reception FIFO

The separate 32-bit reception FIFO (RXFIFO) and transmission FIFO (TXFIFO) are used in different directions for SPI data transactions, and they can enable the SPI to work in a continuous flow (only available in SPI1).

Figure 20-5. Transmission and reception FIFO



When the current TXFIFO level is less than or equal to half of its capacity, the TXFIFO is considered empty(1) and TBE is set to 1 by hardware at this time. When the TBE bit is set, writing data to the SPI_DATA register will store the data at the end of the TXFIFO. Hardware sets the RBNE bit when the RXFIFO is considered non-empty(2). When the RBNE bit is set, reading data from the SPI_DATA register will get the oldest data from the RXFIFO.

Note: (1) For SPI1, the TXFIFO empty means that the TXFIFO level is less than or equal to half of its capacity. The meaning of TXFIFO full is the opposite. Therefore, when the data frame format is not greater than 8 bits, the TXFIFO can store up to three data frames. If the TXFIFO empty or full appears below and there is no special explanation, the meaning is the same as that described here.

(2) For SPI1, the meaning of RXFIFO empty is divided into the following two conditions: If BYTEN bit in SPI_CTL1 is set, the RXFIFO empty means the RXFIFO level is less than quarter of its capacity. At this time, when the data frame format is not more than 8 bits, the RXFIFO can store up to 4 data frames. If BYTEN is cleared, the RXFIFO empty means the RXFIFO level is less than half of its capacity. The meaning of RXFIFO full is the opposite. If the RXFIFO empty or full appears below and there is no special explanation, the meaning is the same as that described here.

Data merging (Only for SPI1)

When DZ[3:0] in the SPI_CTL1 register configures the transmission data bit width to be 8 bits or less than 8 bits, by configuring the BYTEN bit in the SPI_CTL1 register to 0, the data merge transmission mode function is enabled. When DZ[3:0] in the configuration SPI_CTL1 register configures the transmission data bit width to be less than or equal to 8 bits, this function can realize that when 16-bit write access is performed to the SPI_DATA register, two data frames are sent in parallel instead of serial line method.

Similarly, at the receiving end, the receiver obtains these two data frames through a 16-bit read access to SPI_DATA, and only one RBNE event will be generated when the two frames of data are received.

Note: when an odd number of data bytes will be transferred, on the transmitter side, writing the last data frame of any odd sequence with an 8-bit access to SPI_DATA is enough. The receiver has to change BYTEN for the last data frame received in the odd sequence of frames in order to generate the RBNE event.

20.3.5. 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 transmits/receives data only when NSS level is low. In software NSS mode, NSS pin is not used.

Table 20-3. 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 20-4. 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.

20.3.6. SPI operation modes

Table 20-5. SPI operation 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 BDEN = 0 BDOEN: Don't care	MOSI: not used MISO: reception
MTB	Master transmission with bidirectional connection	MSTMOD = 1 RO = 0	MOSI: transmission MISO: not used

Mode	Description	Register configuration	Data pin usage
		BDEN = 1 BDOEN = 1	
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 20-6. A typical full-duplex connection

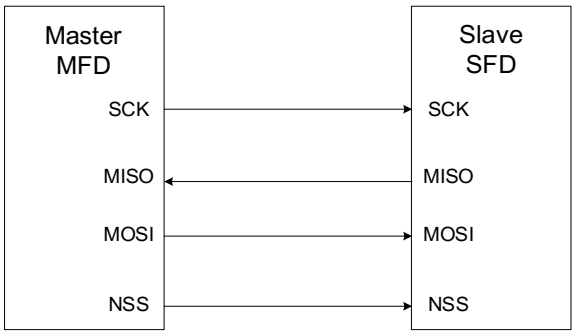


Figure 20-7. A typical simplex connection (Master: receive, Slave: transmit)

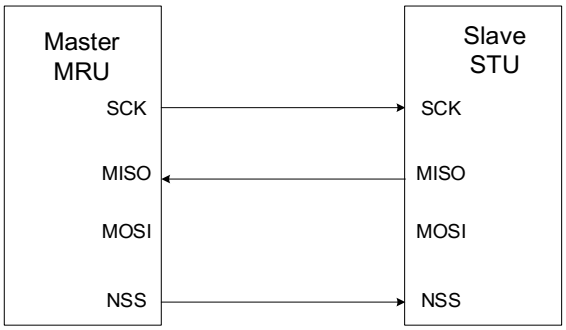


Figure 20-8. A typical simplex connection (Master: transmit only, Slave: receive)

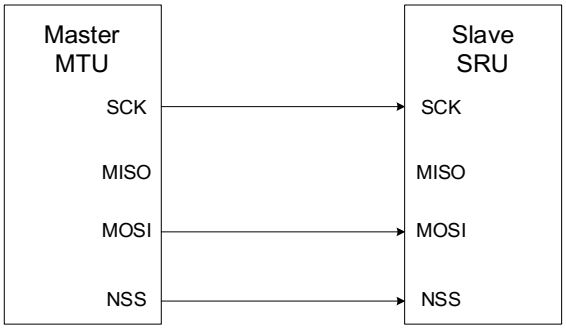
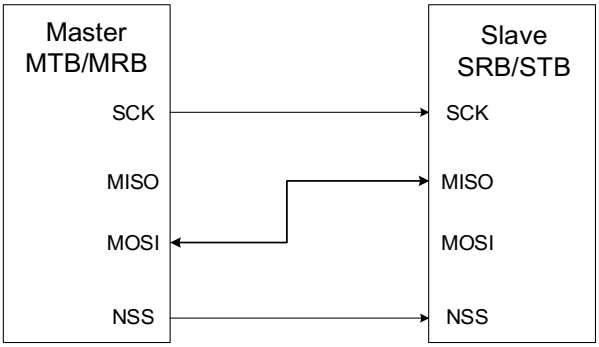


Figure 20-9. A typical bidirectional connection



Initialization sequence

SPI0:

Before transmitting 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. Configure data format (FF16 bit in the SPI_CTL0 register).
3. Configure the clock timing register (CKPL and CKPH bits in the SPI_CTL0 register).
4. Configure the frame format (LF bit in the SPI_CTL0 register).

5. Configure 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 ***Table 20-5. SPI operation modes*** section.
9. Enable the SPI (set the SPIEN bit).

Note: During communication, CKPH, CKPL, MSTMOD, PSC[2:0] and LF bits should not be changed.

SPI1:

Before transmitting 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. Configure the clock timing register (CKPL and CKPH bits in the SPI_CTL0 register).
3. Configure the frame format (LF bit in the SPI_CTL0 register).
4. Configure data format (DZ[3:0] bits in the SPI_CTL1 register) and the access size for the SPI_DATA register (BYTEN bit in the SPI_CTL1 register).
5. Configure 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 operation modes described in ***Table 20-5. SPI operation modes*** section.
9. Configure the TXDMA_ODD and RXDMA_ODD bits, according to the needs of the application.
10. If Quad-SPI mode is used, set the QMOD bit in SPI_QCTL register. Ignore this step if Quad-SPI mode is not used.
11. Enable the SPI (set the SPIEN bit).

Note: During communication, CKPH, CKPL, MSTMOD, PSC[2:0], LF and DZ[3:0] bits should not be changed.

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 transmission buffer/TXFIFO. In slave mode the transmission starts when SCK clock signal begins to toggle at SCK pin and NSS level is low, so application should ensure that data is already written into transmission buffer/TXFIFO before the transmission starts in slave mode.

When SPI begins to send a data frame, it first loads this data frame from the transmission buffer/TXFIFO to the shift register and then begins to transmit the loaded data frame. After TBE flag is set, which means the transmission buffer/TXFIFO 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 reception buffer/RXFIFO and RBNE will be set. The application should read SPI_DATA register to get the received data and this will clear the RBNE flag automatically when reception buffer/RXFIFO is empty. 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 transmission buffer/TXFIFO 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/RXFIFO 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 20-10. Timing diagram of TI master mode with discontinuous transfer

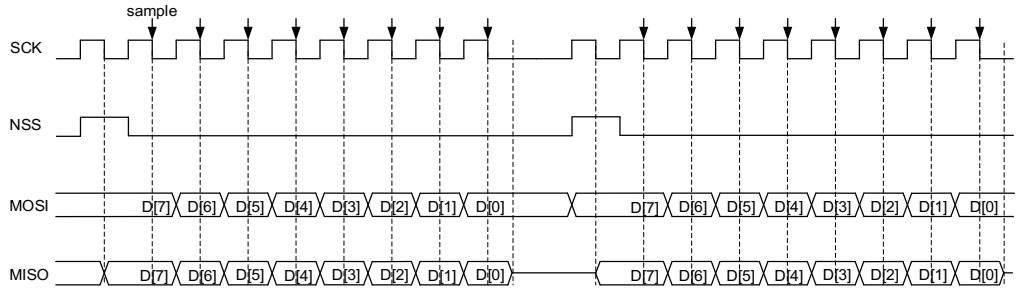
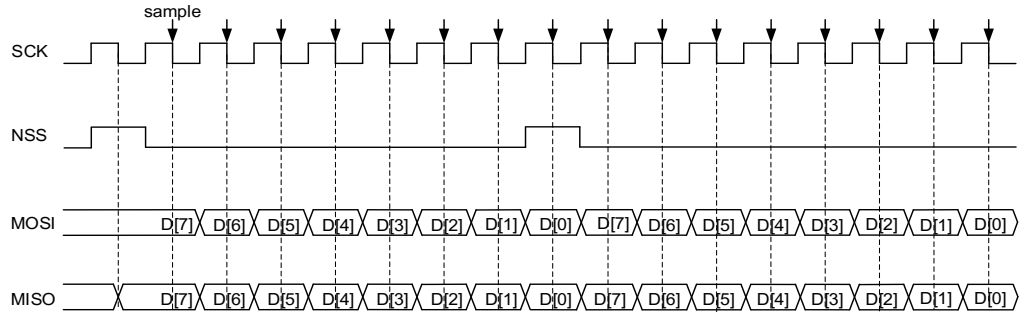
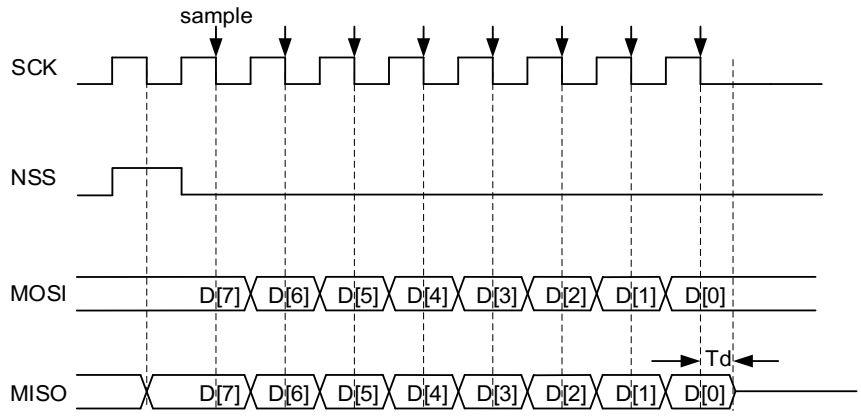


Figure 20-11. Timing diagram of TI master mode with continuous transfer



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 20-12. 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 calledT_d, T_d is decided by PSC [2:0] bits in SPI_CTL0 register.

$$T_d = \frac{T_{bit}}{2} + 5 * T_{pclk} \quad (20\text{-}1)$$

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.

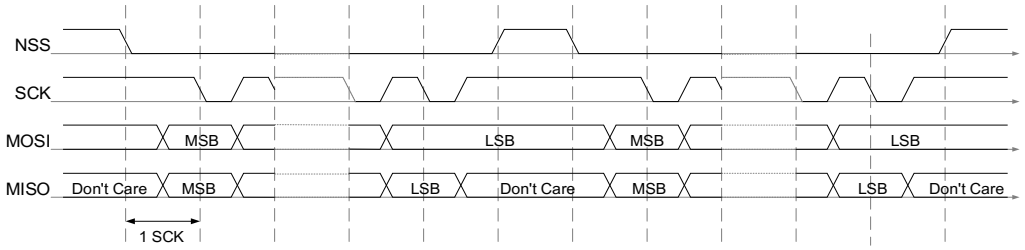
NSS pulse mode operation sequence

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 transmission buffer/TXFIFO. Multiple SCK clock cycle intervals are possible if the transfer buffer/TXFIFO 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 20-13. Timing diagram of NSS pulse with continuous transmit



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, CRCL, RO and LF in SPI_CTL0 register should be kept cleared and DZ[3:0] should be set to ensure that SPI data size is 8-bit, MSTMOD should be set to ensure that SPI is in master mode. SPIEN, PSC, CKPL and CKPH 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.

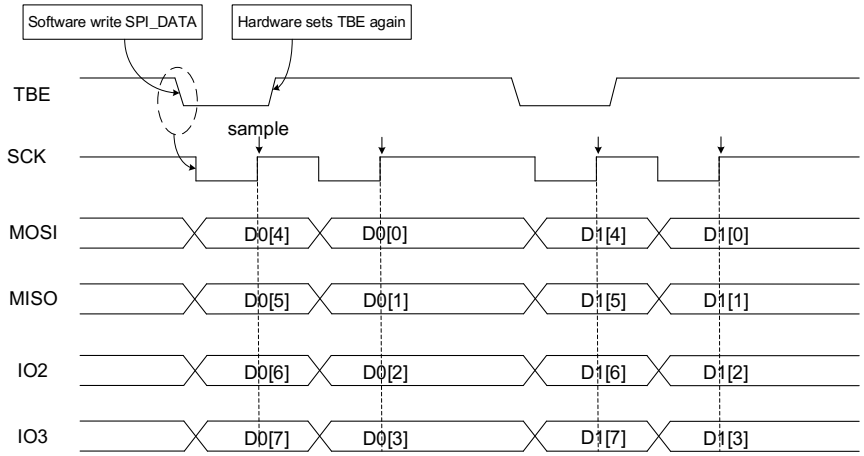
Quad write operation

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:

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 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 20-14. Timing diagram of write operation in Quad-SPI mode



Quad read operation

SPI works in quad read mode when QMOD and QRD 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.

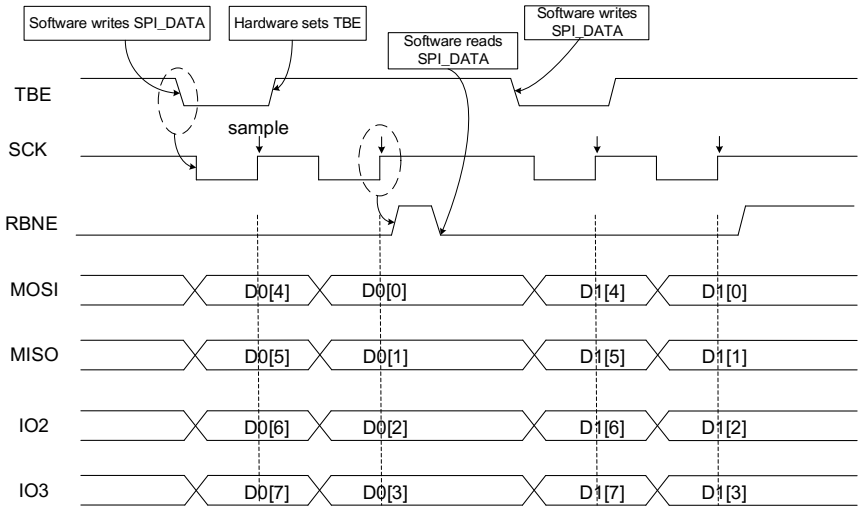
The operation flow for receiving in quad mode is shown below:

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 20-15. Timing diagram of read operation in Quad-SPI mode



SPI disabling sequence

Different sequences are used to disable the SPI in different operation modes.

MFD SFD

For SPI0, 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.

For SPI1, wait until TXLVL[1:0] = 00 and confirm TRANS = 0. Then disable the SPI by clearing SPIEN bit. At last, read data until RXTVL[1:0] = 00.

MTU MTB STU STB

For SPI0, 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.

For SPI1, wait until TXLVL[1:0] = 00 and confirm TRANS = 0. Then disable the SPI by clearing SPIEN bit.

MRU MRB

For SPI0, 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.

For SPI1, after getting the second last RBNE flag, read out this data and delay for a SCK clock time and disable the SPI by clearing SPIEN bit. At last, wait until the last RBNE flag is

set and read data until RXTVL[1:0] = 00.

SRU SRB

For SPI0, 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 completed.

For SPI1, application can disable the SPI when it doesn't want to receive data, and then confirm the TRANS=0 and read data until RXLVL[1:0] = 00.

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.

20.3.7. 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, apIO2plication should first configure DMA modules correctly, 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.

Data merging with DMA (Only for SPI1)

In the case of using DMA for data transmission, when BYTEN is set to 0 and the data length configured by DZ[3:0] is less than or equal to 8 bits and the data merging mode is enabled, the DMA will access the SPI_DATA register in 16-bit mode, automatically Complete the data transmission.

In the case that the data packetization mode is enabled and the frame number of the data frame is not an even multiple, in order to avoid the problem of one more frame of data in the last DMA transmission, the TXDMA_ODD/RXDMA_ODD bit in the SPI_CTL1 register needs

to be set to 1.

20.3.8. 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 calculate CRC for each bit transmitted and received on lines continuously, 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 transmission buffer/TXFIFO. 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.

For SPI0, if the data length is 8-bit, the CRC calculation is based on the CRC8 standard. If i the data length is 16-bit, the CRC calculation is based on the CRC16 standard. If the DMA function is enabled, the software does not need to set the CRCNT bit, and the hardware will handle the CRC transmission and verification automatically.

For SPI1, only when the data length is 8-bit or 16-bit, SPI provides CRC calculation function, and independent of the data length, it can be fixedly set to 8-bit or 16-bit CRC calculation. For all other data lengths, CRC function is invalid. CRC data exchange usually requires one or more data communication time after the end of the data sequence. For example, when setting the data length to 8 bits and doing a 16-bit CRC check, it takes two frames to send the complete CRC data. If the DMA function is enabled, the hardware will handle the CRC transmission and verification automatically, but the SPI needs to set the counter value of the DMA transmitting channel and the receiving channel. The transmitting DMA counter value is the number of data frames that do not include the CRC frame. The configuration of receiving the DMA counter value is as follows:

1. Full duplex mode: Suppose the amount of data received by SPI is L, when CRCL = 0 and DZ = 8, then the count of the DMA receive channel is L + 1, otherwise the count of the DMA receive channel is L + 2.

2. Receive only mode: DMA receive channel count is only equal to the amount of data received .After receiving data, the CRC value is obtained by reading SPI_RCRC register by software.

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).

20.3.9. SPI interrupts

Status flags

- **Transmission buffer/TXFIFO empty flag (TBE)**

This bit is set when the transmission buffer is empty or the TXFIFO level is lower or equal to 1/2 of FIFO depth, the software can write the next data to the transmission buffer/TXFIFO by writing the SPI_DATA register.

- **Reception buffer/RXFIFO not empty flag (RBNE)**

For SPI0, this bit is set when reception buffer is not empty, which means that one data is received and stored in the reception buffer, and software can read the data by reading the SPI_DATA register.

For SPI1, this bit is set depending on the BYTEN bit in the SPI_CTL1: If BYTEN = 0, the RBNE is set when the RXFIFO level is greater or equal to 1/2(16-bit). If BYTEN = 1, the RBNE is set when the RXFIFO level is greater or equal to 1/4(8-bit).

- **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 conditions

- **Configuration fault error (CONFERR)**

CONFERR is an error flag in master mode. In NSS hardware mode and 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 bit are write protection 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. For SPI0, that means the last data has not been read out and the newly incoming data is received. For SPI1, that means the RXFIFO has not enough space to store this received data. The reception buffer/RXFIFO 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.

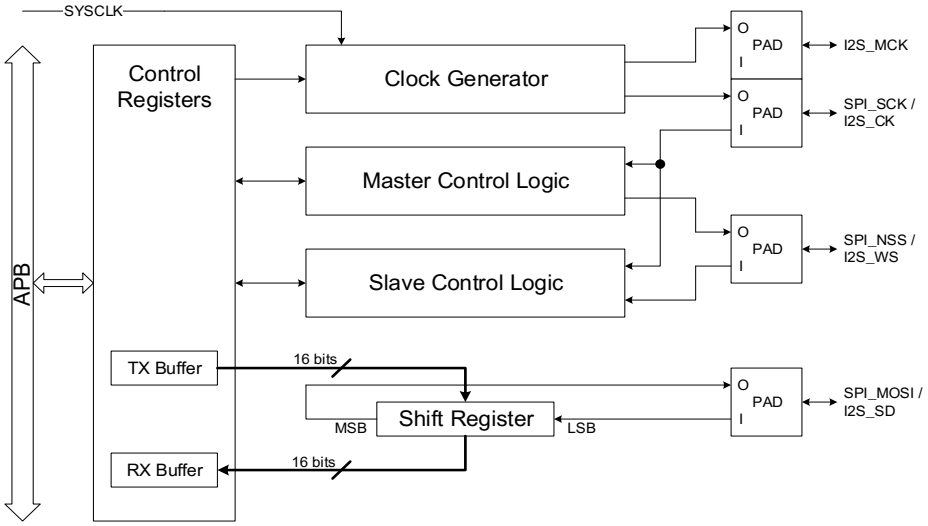
Table 20-6. SPI interrupt requests

Flag	Description	Clear method	Interrupt enable bit
TBE	Transmission buffer/TXFIFO empty	Write SPI_DATA register.	TBEIE
RBNE	Reception buffer/RXFIFO 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	

20.4. I2S function overview

20.4.1. I2S block diagram

Figure 20-16. 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 I2SCK and I2S_WS. The shift register handles the serial data transmission and reception on I2S_SD.

20.4.2. 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 equal to 256 x Fs, and Fs is the audio sampling frequency.

20.4.3. 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-multiplexed 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 bit 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-bit, 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 the data. The timing diagrams for each configuration are shown below.

Figure 20-17. I2S PHILIPS standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0)

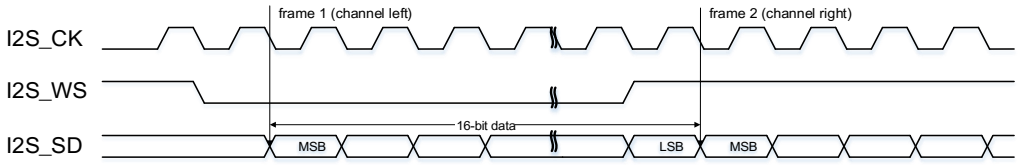
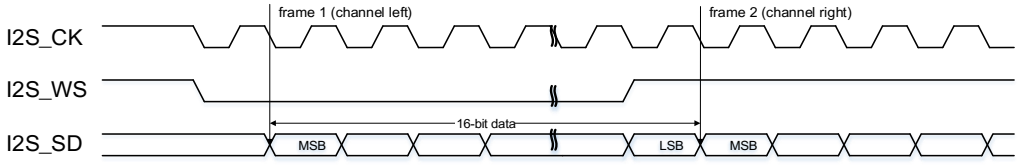


Figure 20-18. I2S PHILIPS standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1)



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 20-19. I2S PHILIPS standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

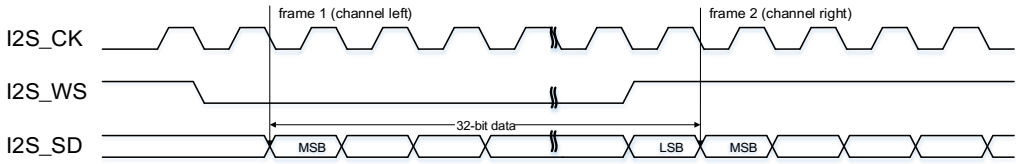
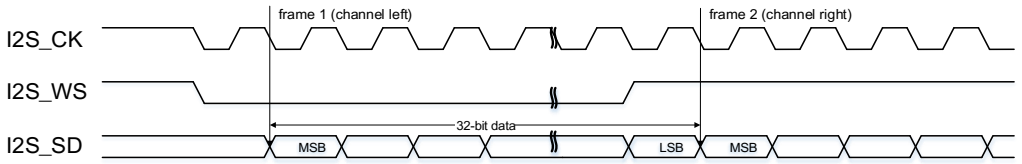


Figure 20-20. I2S PHILIPS standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1)



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 higher 16 bits, and the second one should be the lower 16 bits.

Figure 20-21. I2S PHILIPS standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

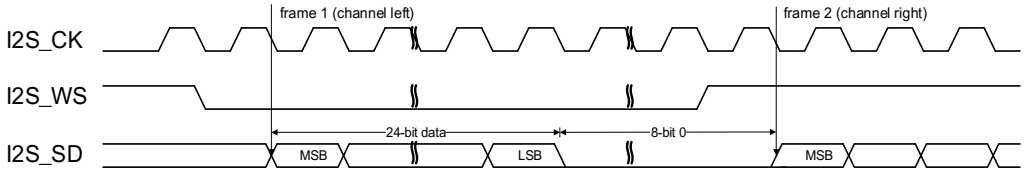
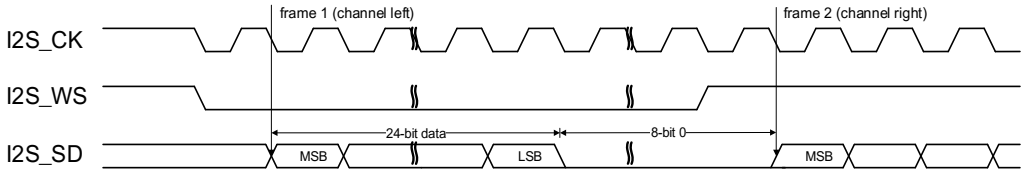


Figure 20-22. I2S PHILIPS standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1)



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 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 20-23. I2S PHILIPS standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

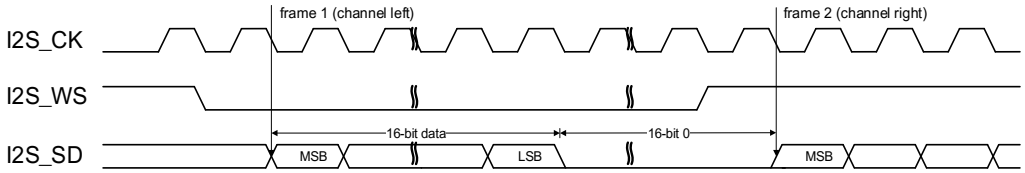
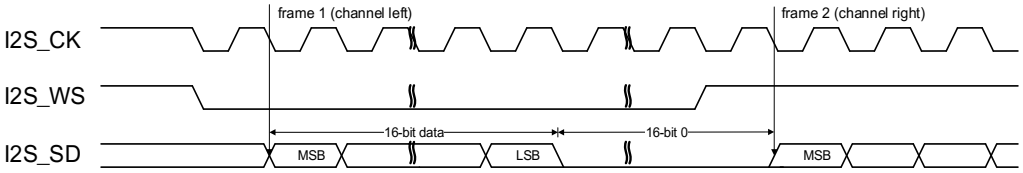


Figure 20-24. I2S PHILIPS standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



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 20-25. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0)

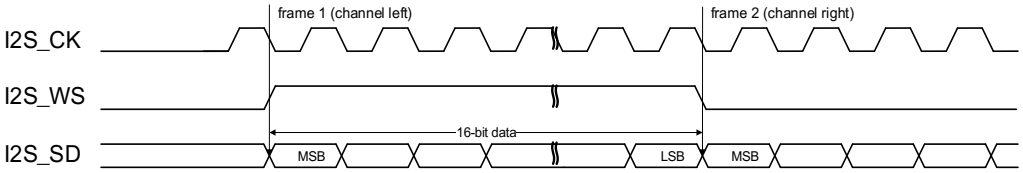


Figure 20-26. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1)

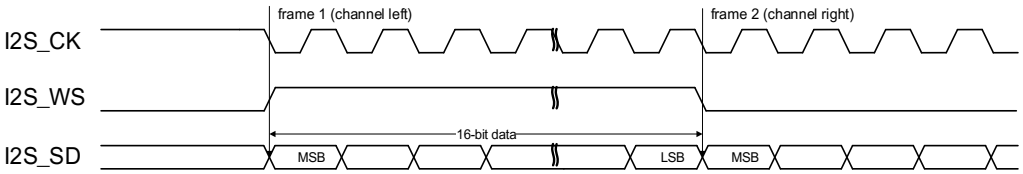


Figure 20-27. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

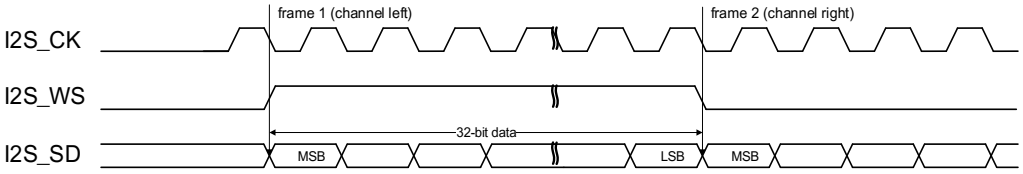


Figure 20-28. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1)

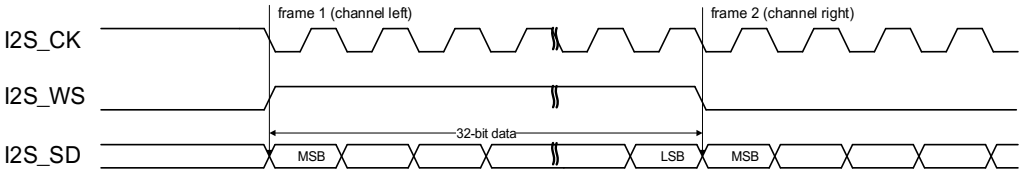


Figure 20-29. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

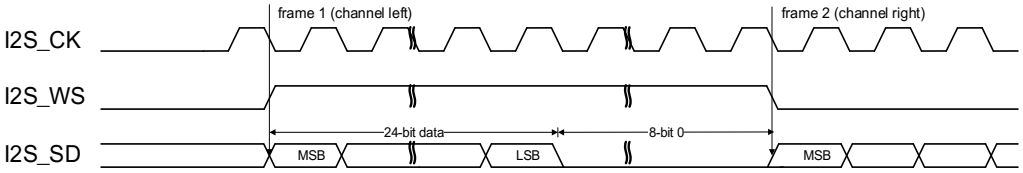


Figure 20-30. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1)

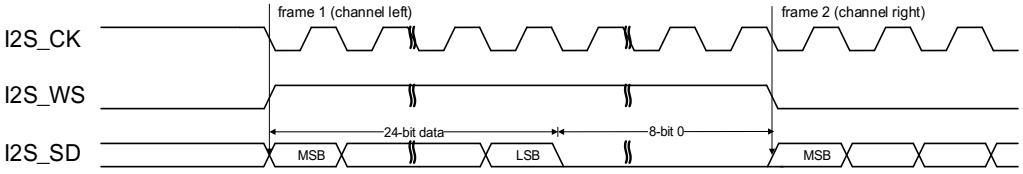


Figure 20-31. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

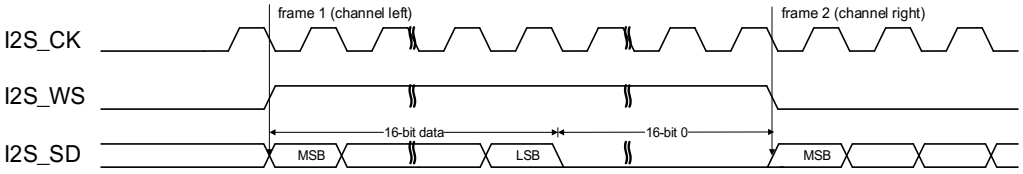
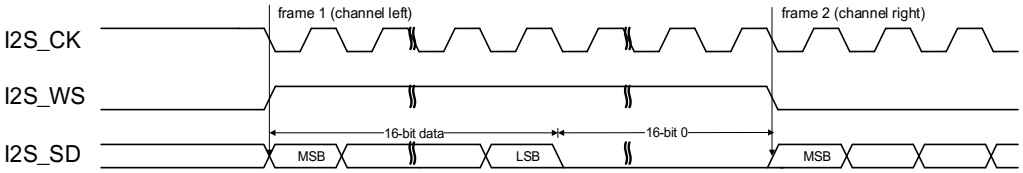


Figure 20-32. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



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 20-33. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

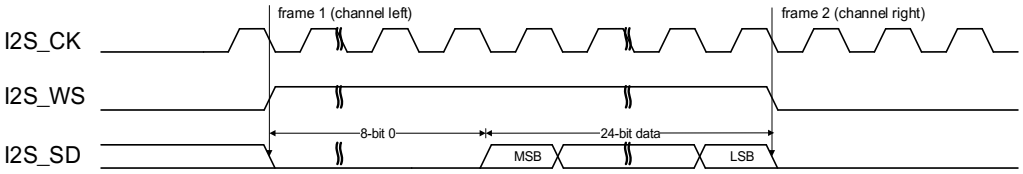
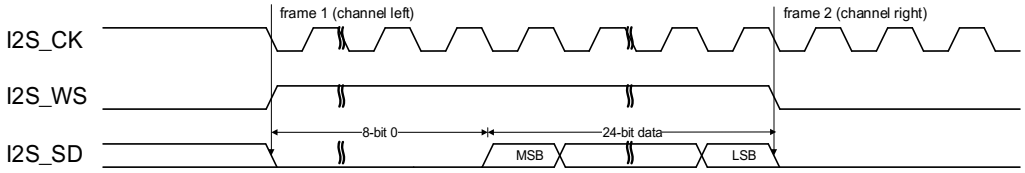


Figure 20-34. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1)



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 20-35. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

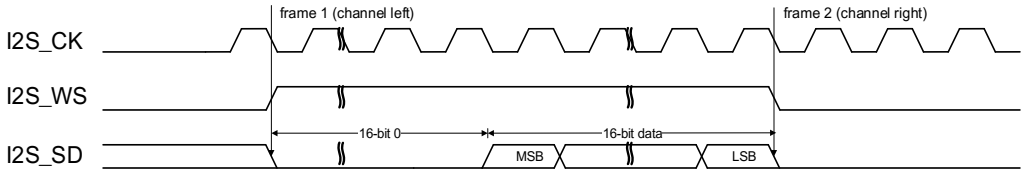
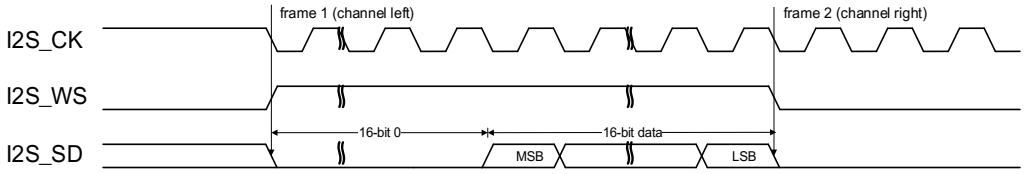


Figure 20-36. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



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 20-37. PCM standard short frame synchronization mode timing diagram

(DTLEN=00, CHLEN=0, CKPL=0)

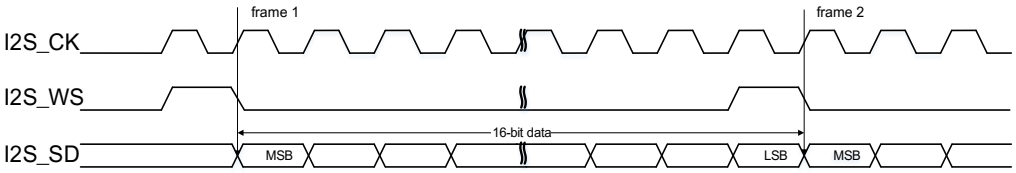


Figure 20-38. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1)

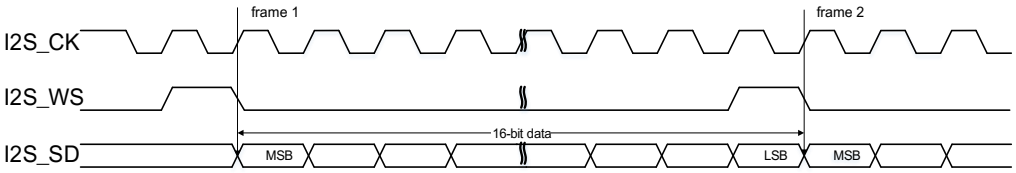


Figure 20-39. PCM standard short frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

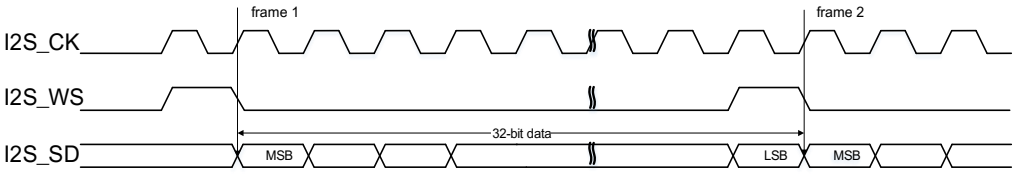


Figure 20-40. PCM standard short frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=1)

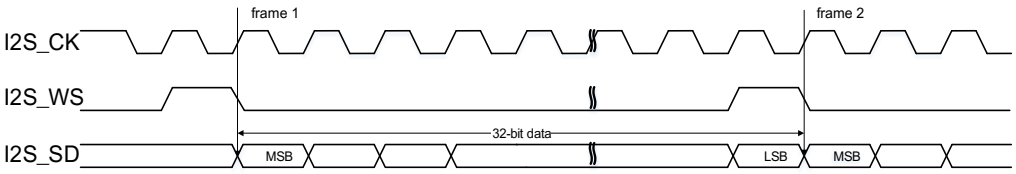


Figure 20-41. PCM standard short frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

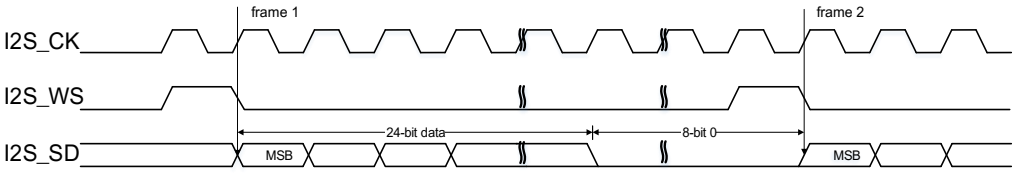


Figure20-42. PCM standard short frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=1)

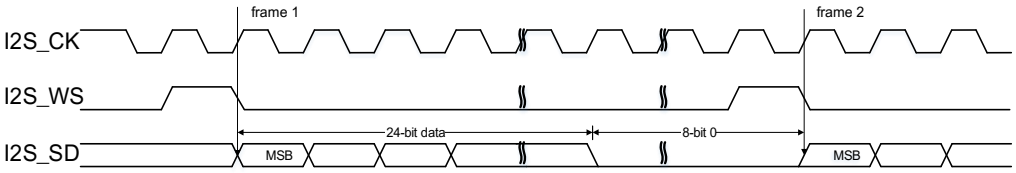


Figure 20-43. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

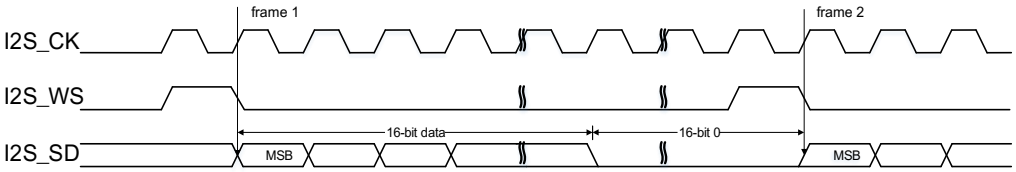
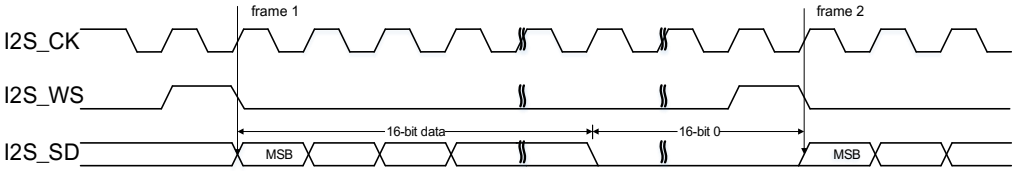


Figure 20-44. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



The timing diagrams for each configuration of the long frame synchronization mode are shown below.

Figure 20-45. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=0)

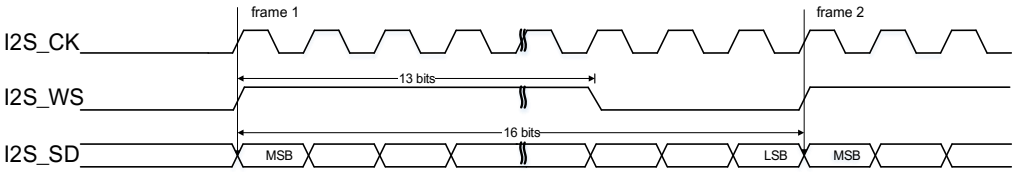


Figure20-46. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1)

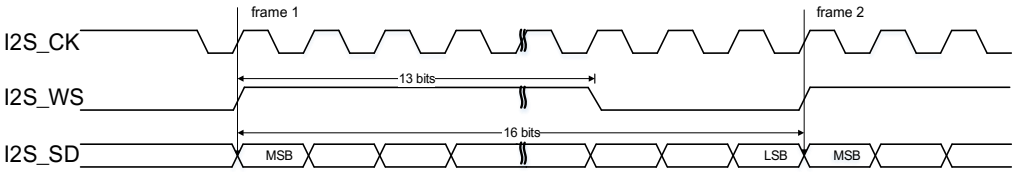


Figure 20-47. PCM standard long frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

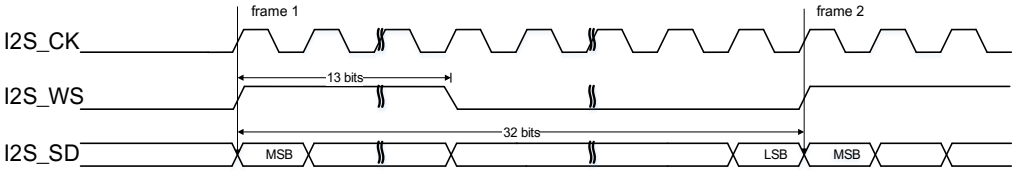


Figure 20-48. PCM standard long frame synchronization mode timing diagram

(DTLEN=10, CHLEN=1, CKPL=1)

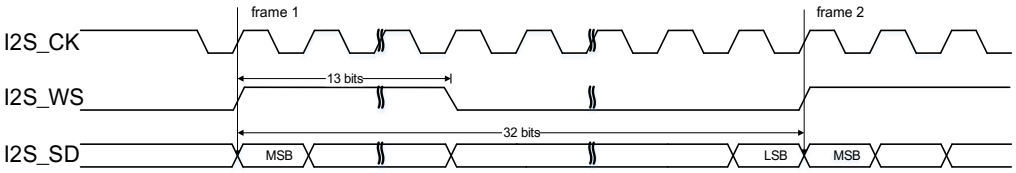


Figure 20-49. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

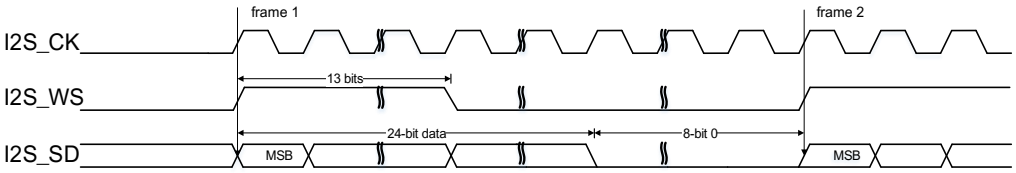


Figure 20-50. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=1)

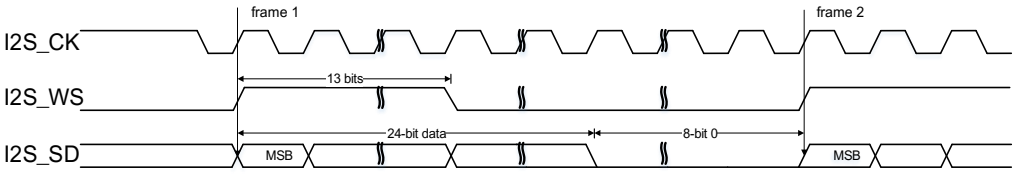


Figure 20-51. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

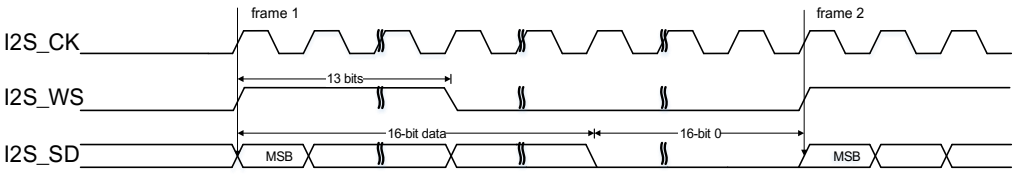
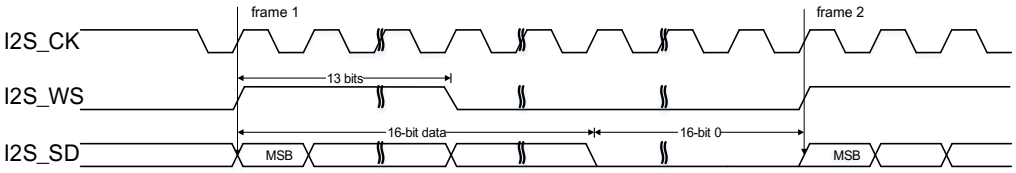
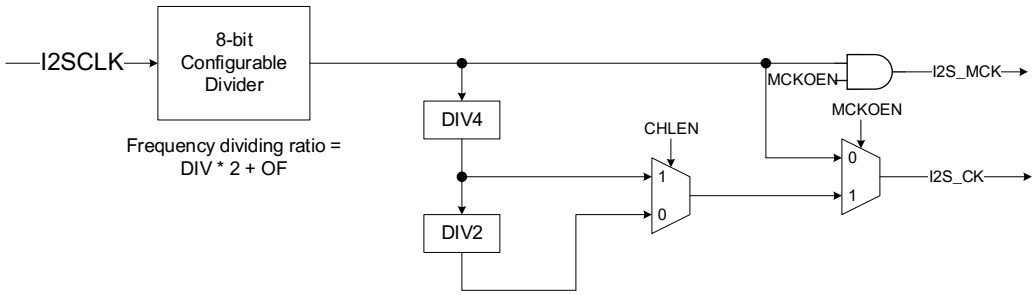


Figure 20-52. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



20.4.4. I2S clock

Figure 20-53. Block diagram of I2S clock generator



The block diagram of I2S clock generator is shown as *Figure 20-53. 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 20-7. I2S bitrate calculation formulas*.

Table 20-7. 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 20-8. Audio sampling frequency calculation formulas*.

Table 20-8. 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).

20.4.5. Operation

Operation modes

The operation mode is selected by the I2SOPMOD[1:0] 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 20-9. Direction of I2S interface signals for each operation mode*.

Table 20-9. Direction of I2S interface signals for each operation mode

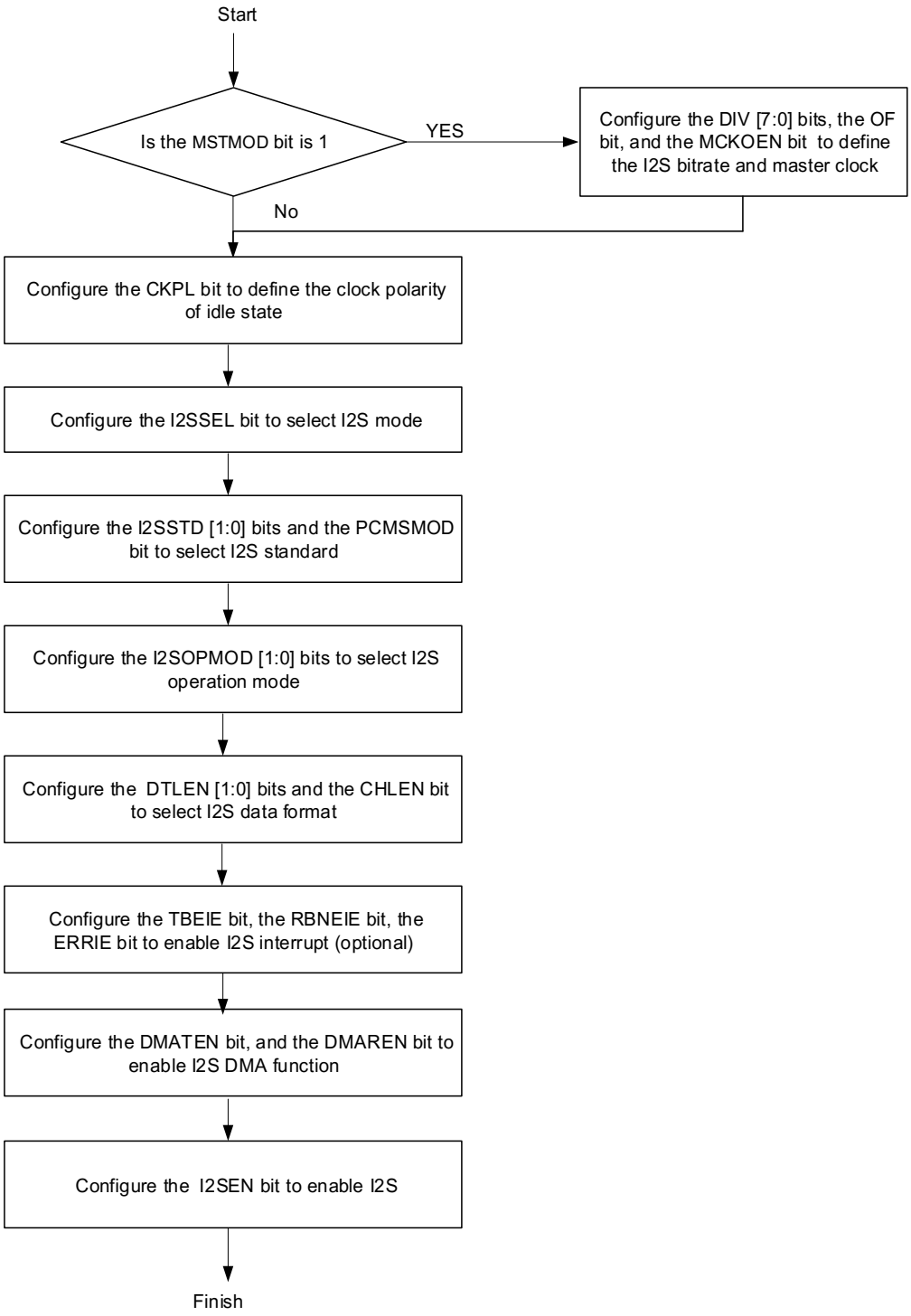
Operation mode	I2S_MCK	I2S_CK	I2S_WS	I2S_SD
Master transmission	Output or NU(1)	Output	Output	Output
Master reception	Output or NU(1)	Output	Output	Input
Slave transmission	Input or NU(1)	Input	Input	Output
Slave reception	Input or NU(1)	Input	Input	Input

1. NU means the pin is not used by I2S and can be used by other functions.

I2S initialization sequence

I2S initialization sequence is shown as *Figure 20-54. I2S initialization sequence*.

Figure 20-54. I2S initialization sequence



I2S master transmission sequence

The TBE flag is used to control the transmission sequence. As is mentioned before, the TBE flag indica tes that the transmission buffer is empty, and an interrupt will be generated if the TBEIE bit in the SPI_CTL1 register is set. At the beginning, the transmission 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 transmission 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 transmission 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.

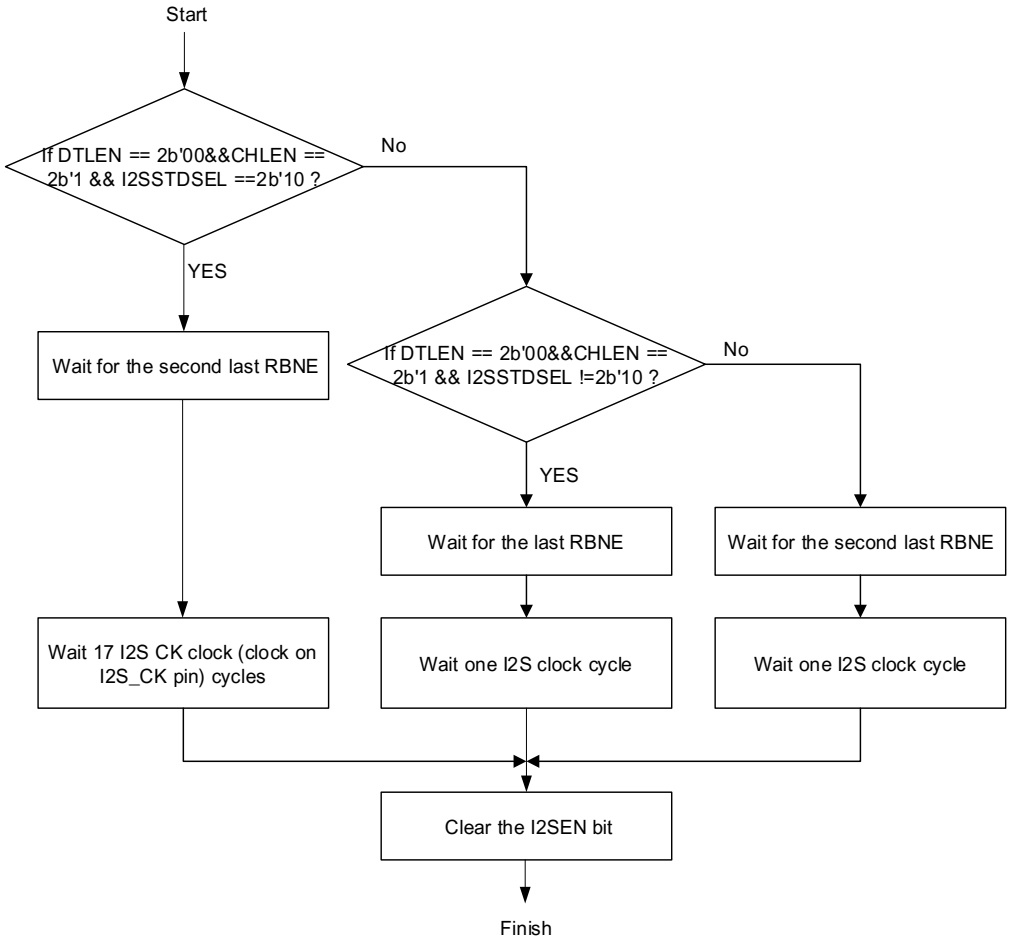
I2S master reception sequence

The RBNE flag is used to control the reception sequence. As is mentioned before, the RBNE flag indicates the reception 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 reception buffer is empty (RBNE is low). When a reception sequence finishes, the received data in the shift register is loaded into the reception 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 the channel side which 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 as below ***Figure 20-55. I2S master reception disabling sequence***.

Figure 20-55. 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.

20.4.6. DMA function

DMA function is the same as SPI mode. The only difference is that the CRC function is not available in I2S mode.

20.4.7. 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.

- **Transmission buffer empty flag (TBE)**

This bit is set when the transmission buffer is empty, the software can write the next data to the transmission buffer by writing the SPI_DATA register.

- **Reception buffer not empty flag (RBNE)**

This bit is set when reception buffer is not empty, which means that one data is received and stored in the reception 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 conditions

There are three error flags:

- **Transmission underrun error flag (TXURERR)**

This situation occurs when the transmission buffer is empty when the valid SCK signal starts in slave transmission mode.

- **Reception overrun error flag (RXORERR)**

This situation occurs when the reception buffer is full and a newly incoming data has been completely received. When overrun occurs, the data in reception 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 enabled bits are summed up in the *Table 20-10. I2S interrupt*.

Table 20-10. I2S interrupt

Interrupt flag	Description	Clear method	Interrupt enable bit
TBE	Transmission buffer empty	Write SPI_DATA register	TBEIE
RBNE	Reception 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	

20.5. Register definition

SPI0/I2S0 base address: 0x4001 3000

SPI1 base address: 0x4000 3800

20.5.1. Control register 0 (SPI_CTL0)

Address offset: 0x00
Reset value: 0x0000 0000

This register can be accessed by byte (8-bit) or 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 / CRCL	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: Disable CRC calculation. 1: Enable CRC calculation.
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 (only for SPI0) 0: 8-bit data frame format 1: 16-bit data frame format
	CRCL	CRC length (only for SPI1) 0: 8-bit crc length. 1: 16-bit crc length.
10	RO	Receive only When BDEN is cleared, this bit determines the direction of transfer. 0: Full-duplex mode 1: Receive-only mode
9	SWNSSEN	NSS software mode selection 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 has an effect 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: Disable SPI peripheral. 1: Enable SPI peripheral.
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. PCLK means PCLK1 when using SPI1.

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.

20.5.2. Control register 1 (SPI_CTL1)

Address offset: 0x04

Reset value: SPI0：0x0000 0000

SPI1：0x0000 0700

This register can be accessed by byte (8-bit) or 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	TXDMA_ODD	RXDMA_ODD	BYTEN	DZ[3:0]				TBEIE	RBNEIE	ERRIE	TMOD	NSSP	NSSDRV	DMATEN	DMAREN
	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	TXDMA_ODD	Odd bytes in TX DMA channel (only for SPI1) In data merging mode, this bit is set if the total number of data to transmit by DMA is odd. It has effect only when DMATEN is set and data merging mode enable (data size is less than or equal to 8-bit and write access to SPI_DATA is 16-bit wide). This field can be written only when SPI is disabled. 0: The total number of data to transmit by DMA is even. 1: The total number of data to transmit by DMA is odd.
13	RXDMA_ODD	Odd bytes in RX DMA channel (only for SPI1) In data merging mode, this bit is set if the total number of data to receive by DMA is odd. It has effect only when DMAREN is set and data merging mode enable (data size is less than or equal to 8-bit and write access to SPI_DATA is 16-bit wide). This field can be written only when SPI is disabled. 0: The total number of data to receive by DMA is even.

		1: The total number of data to receive by DMA is odd.
12	BYTEN	Byte access enable (only for SPI1) This bit is used to indicate the access size to FIFO, and set the threshold of the RXFIFO that generate RBNE. 0: Half-word access, and RBNE is generated when RXLVL >= 2. 1: Byte access, and RBNE is generated when RXLVL >= 1.
11:8	DZ[3:0]	Date size (only for SPI1) This field indicates the data size for transfer. 0000: Force to "0111" 0001: Force to "0111" 0010: Force to "0111" 0011: 4-bit 0100: 5-bit …… 1111: 16-bit
7	TBEIE	Transmit buffer / TXFIFO 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 / RXFIFO 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 or the CONFERR bit or the FERR bit or 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 master NSS output 1: Enable master NSS output
1	DMATEN	Transmit buffer / TXFIFO DMA enable 0: Disable transmit buffer / TXFIFO DMA 1: Enable transmit buffer / TXFIFO DMA, when the TBE bit in SPI_STAT is set, it will be a DMA request on corresponding DMA channel.

0	DMAREN	Receive buffer / RXFIFO DMA enable 0: Disable receive buffer / RXFIFO DMA 1: Enable receive buffer / RXFIFO DMA, when the RBNE bit in SPI_STAT is set, it will be a DMA request on corresponding DMA channel.

20.5.3. Status register (SPI_STAT)

Address offset: 0x08

Reset value: 0x0000 0002

This register can be accessed by byte (8-bit) or 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			TXLVL[1:0]		RXLVL[1:0]		FERR	TRANS	RXORERR	CONFERR	CRCERR	TXURERR	I2SCH	TBE	RBNE
			r		r		rc_w0	r	r	r	rc_w0	r	r	r	r

Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:11	TXLVL[1:0]	TXFIFO level (only for SPI1) 00: Empty 01: 1/4 full 10: 1/2 full 11: Full **Note:** The FIFO level here refers to the current actual storage of the FIFO. Here, the FIFO is considered full when the FIFO level is greater than 1/2.
10:9	RXLVL[1:0]	RXFIFO level (only for SPI1) 00: Empty 01: 1/4 full 10: 1/2 full 11: Full This field has no meaning when SPI is in receive-only mode with CRC function enabled. **Note:** The FIFO level here refers to the current actual storage of the FIFO. Here, the FIFO is considered full when the FIFO level is greater than 1/2.
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 is able to be cleared by writing 0.
7	TRANS	Transmitting ongoing bit 0: SPI is idle. 1: SPI 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 operation on the SPI_CTL0 register followed by a read or write access to the SPI_STAT 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 is able to be 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 / TXFIFO empty 0: Transmit buffer / TXFIFO is not empty. 1: Transmit buffer / TXFIFO is empty.
0	RBNE	Receive buffer / RXFIFO not empty

0: Receive buffer / RXFIFO is empty.

1: Receive buffer / RXFIFO is not empty.

20.5.4. Data register (SPI_DATA)

Address offset: 0x0C

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit) or 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
SPI_DATA[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	SPI_DATA[15:0]	Data transfer register. For SPI0, the hardware has two buffers, including transmission buffer and reception buffer. Write data to SPI_DATA will save the data to transmission buffer and read data from SPI_DATA will get the data from reception 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, transmission buffer and reception buffer are 8-bits. If the Data frame format is set to 16-bit data, the SPI_DATA [15:0] is used for transmission and reception, transmission buffer and reception buffer are 16-bit. For SPI1, the hardware has two FIFOs, including TXFIFO and RXFIFO. Write data to SPI_DATA will save the data to TXFIFO and read data from SPI_DATA will get the data from RXFIFO. **Note:** In fact, SPI1 hardware determines the size of each access to SPI_DATA only based on the BYTEN bit in SPI_CTL1, regardless of the size of the software's current operation.

20.5.5. CRC polynomial register (SPI_CRCPOLY)

Address offset: 0x10

Reset value: 0x0000 0007

This register can be accessed by byte (8-bit) or 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 register This register contains the CRC polynomial and it is used for CRC calculation. The default value is 0007h.

20.5.6. RX CRC register (SPI_RCRC)

Address offset: 0x14

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit) or 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															
r															

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

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. For SPI0, 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]. For SPI1, CRC function is valid only when the data length is 8 bits or 16 bits. And if the CRC length is set to 8-bit and the data size is equal to 8-bit, the CRC calculation is based on CRC8 standard, and saves the value in RCRC [7:0]. In addition to this, the 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.

20.5.7. TX CRC register (SPI_TCRC)

Address offset: 0x18

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit) or 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
TCRC[15:0]															
r															

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. For SPI0, 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]. For SPI1, CRC function is valid only when the data length is 8 bits or 16 bits. And if the CRC length is set to 8-bit and the data size is equal to 8-bit, the CRC calculation is based on CRC8 standard, and saves the value in TCRC[7:0]. In addition to this, the 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.

20.5.8. I2S control register (SPI_I2SCTL)

Address offset: 0x1C

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit) or 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 mode or I2S mode is disabled.
10	I2SEN	I2S enable 0: Disable I2S 1: Enable I2S 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 mode 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 mode 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 mode 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 mode 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

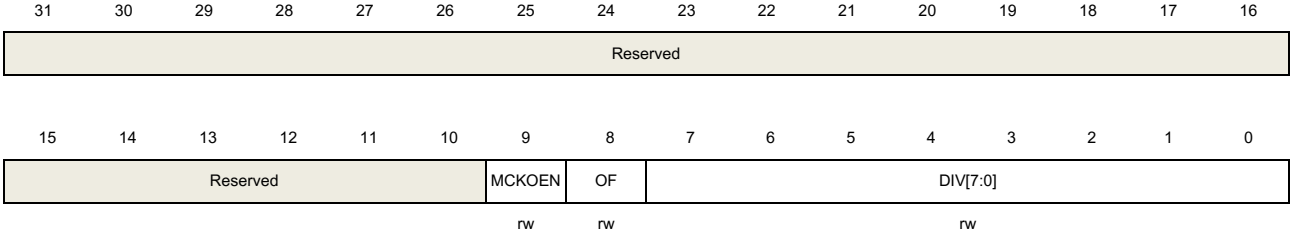
		11: Reserved These bits should be configured when I2S mode is disabled. These bits are not used in SPI mode.
0	CHLEN	Channel length 0: 16 bits 1: 32 bits The channel length must be equal to or greater than the data length. This bit should be configured when I2S mode is disabled. This bit is not used in SPI mode.

20.5.9. I2S clock prescaler register (SPI_I2SPSC)

Address offset: 0x20

Reset value: 0x0000 0002

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



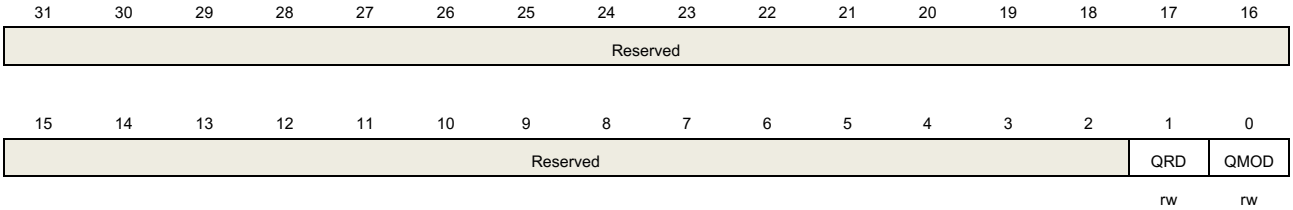
Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	MCKOEN	I2S_MCK output enable 0: Disable I2S_MCK output 1: Enable I2S_MCK output This bit should be configured when I2S mode is disabled. This bit is not used in SPI mode.
8	OF	Odd factor for the prescaler 0: Real divider value is DIV * 2 1: Real divider value is DIV * 2 + 1 This bit should be configured when I2S mode 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 mode is disabled. These bits are not used in SPI mode.

20.5.10. Quad-SPI mode control register (SPI_QCTL) of SPI1

Address offset: 0x80

Reset value: 0x0000 0000

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



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 can only be configured when the SPI is not busy (the TRANS bit is cleared). This bit is only available in SPI1.
0	QMOD	Quad-SPI mode enable. 0: SPI is in single wire mode. 1: SPI is in Quad-SPI mode. This bit can only be configured when the SPI is not busy (the TRANS bit is cleared). This bit is only available in SPI1.

21. Inter-integrated circuit interface (I2C)

21.1. Inter-integrated circuit interface (I2Cx, x=0, 1)

21.1.1. 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.

21.1.2. 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.
- Multiple 7-bit slave addresses (2 addresses with configurable mask).
- 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.

21.1.3. Function overview

Figure 21-1. I2C module block diagram below provides details of the internal configuration of the I2C interface.

Figure 21-1. I2C module block diagram

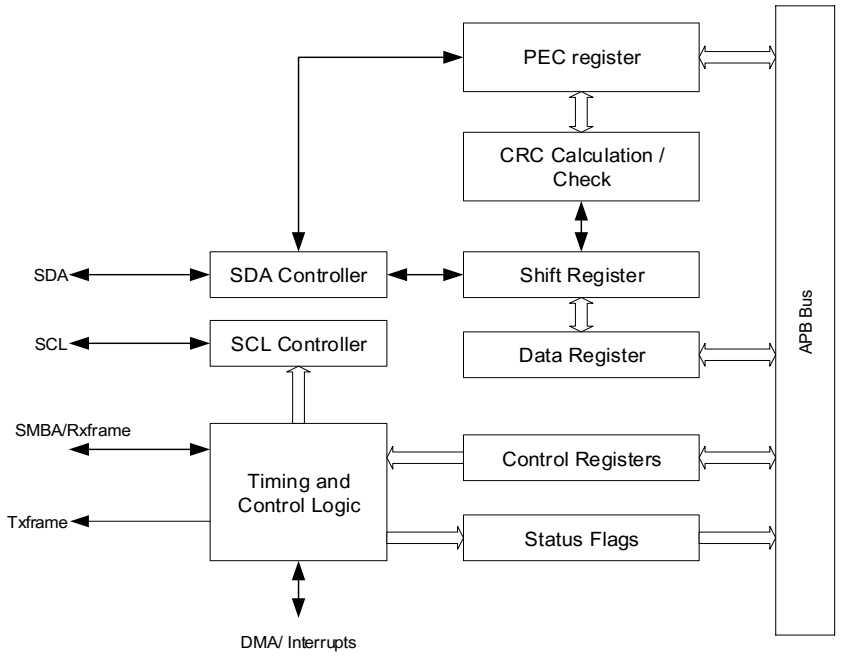


Table 21-1. Definition of I2C-bus terminology (refer to the I2C specification of PHILIPS semiconductors)

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

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.

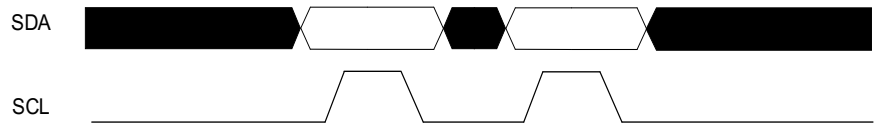
Both SDA and SCL are bidirectional lines, connected to a positive supply voltage via current-source or pull-up resistor. When the bus is free, both lines are HIGH. The output stages of devices connected to the bus must have an open-drain or open-collect to perform the wired-AND function. Data on the 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_CTL2 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}.

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 21-2. Data validation***). One clock pulse is generated for each data bit to be transferred.

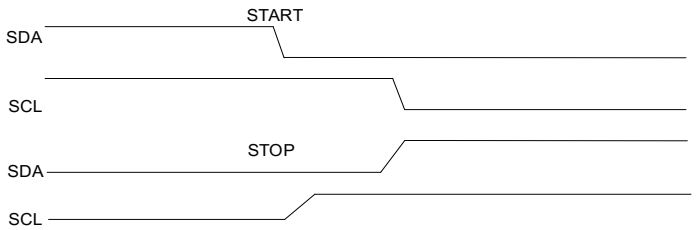
Figure 21-2. Data validation



START and STOP signal

All transmissions begin with a START and are terminated by a STOP (see ***Figure 21-3. 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 21-3. START and STOP signal



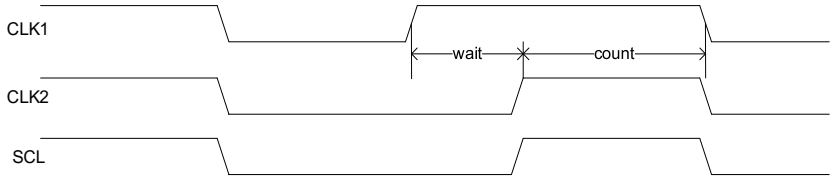
Clock synchronization

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 21-4. 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 21-4. Clock synchronization



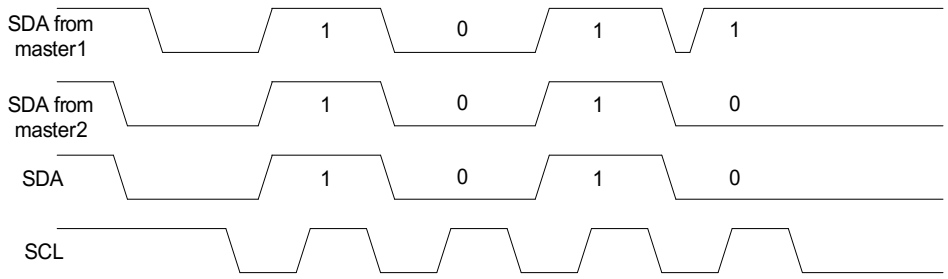
Arbitration

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 21-5. SDA line arbitration



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 21-6. I2C communication flow with 7-bit address

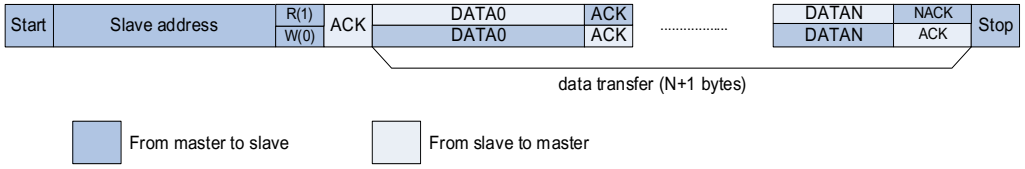


Figure 21-7. I2C communication flow with 10-bit address (Master Transmit)

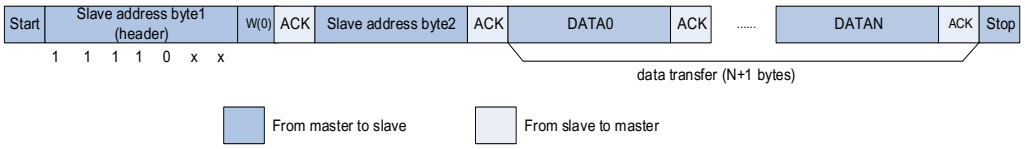
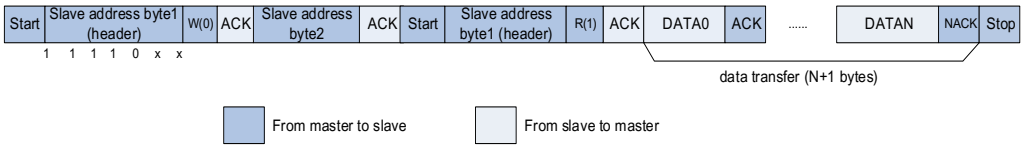


Figure 21-8. I2C communication flow with 10-bit address (Master Receive)



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.

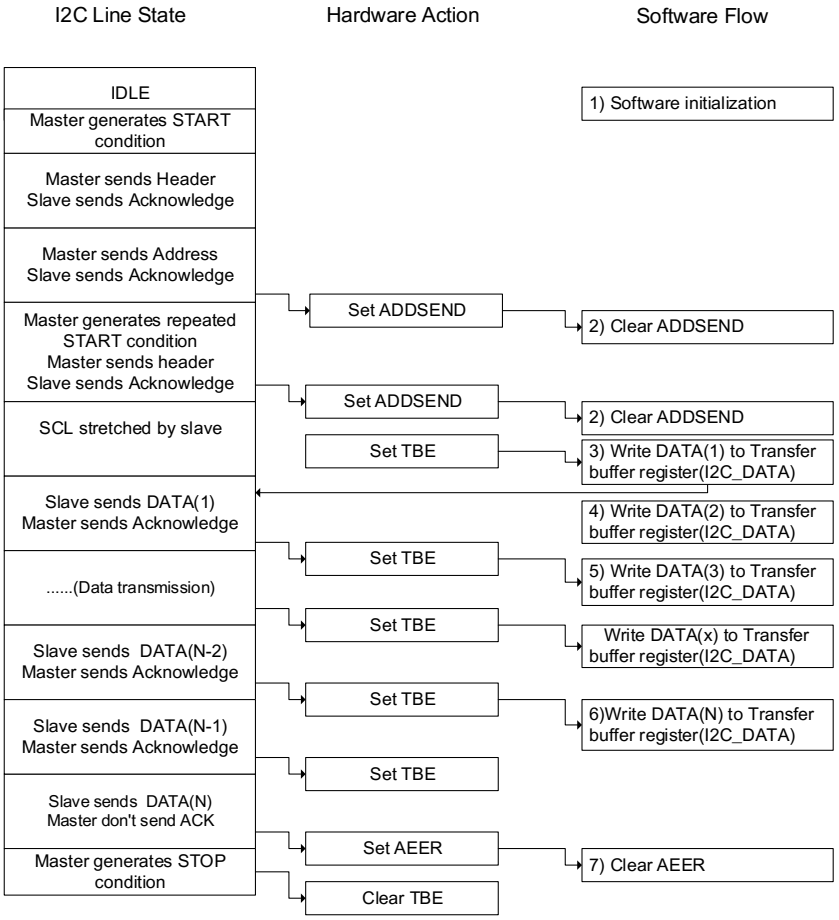
Programming model in slave transmitting mode

As is shown in ***Figure 21-9. 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 21-9. Programming model for slave transmitting (10-bit address mode)



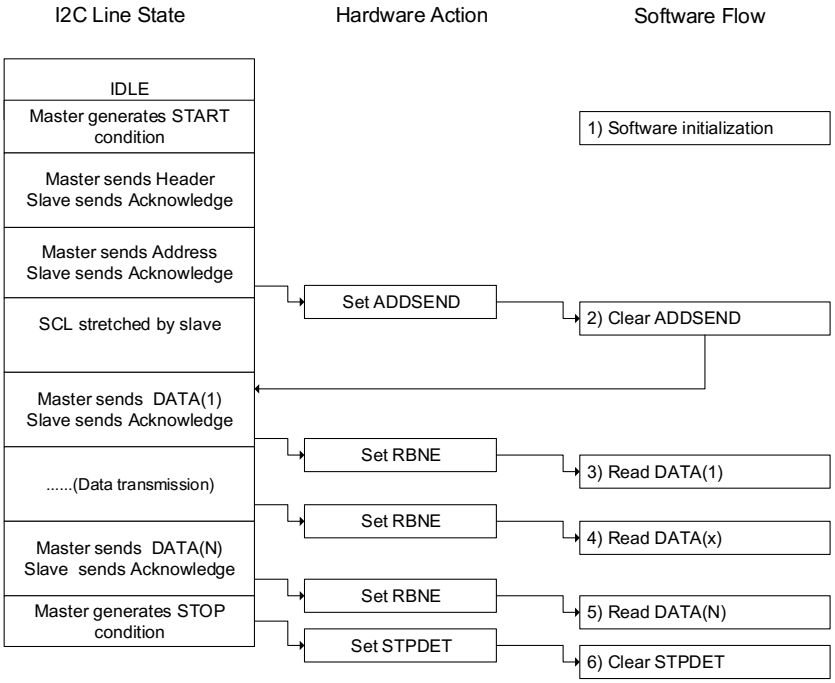
Programming model in slave receiving mode

As is shown in ***Figure 21-10. 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 21-10. Programming model for slave receiving (10-bit address mode)



Programming model in master transmitting mode

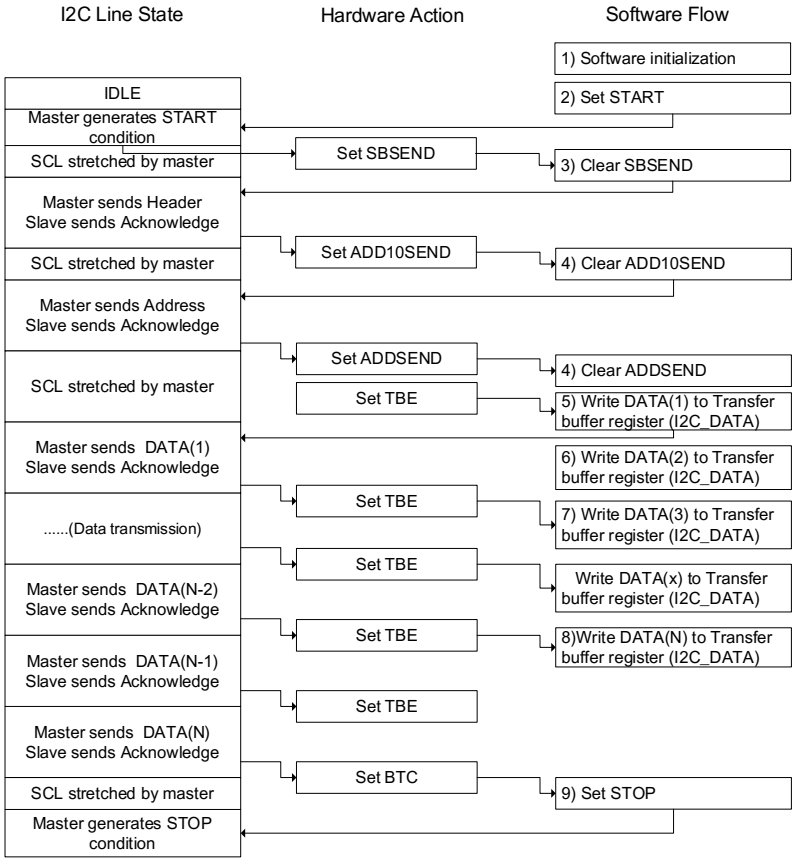
As is shown in ***Figure 21-11. Programming model for master transmitting (10-bit address mode)***, the following software procedure should be followed if users wish to make transaction in master transmitter mode:

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 21-11. Programming model for master transmitting (10-bit address mode)



Programming model in master receiving mode

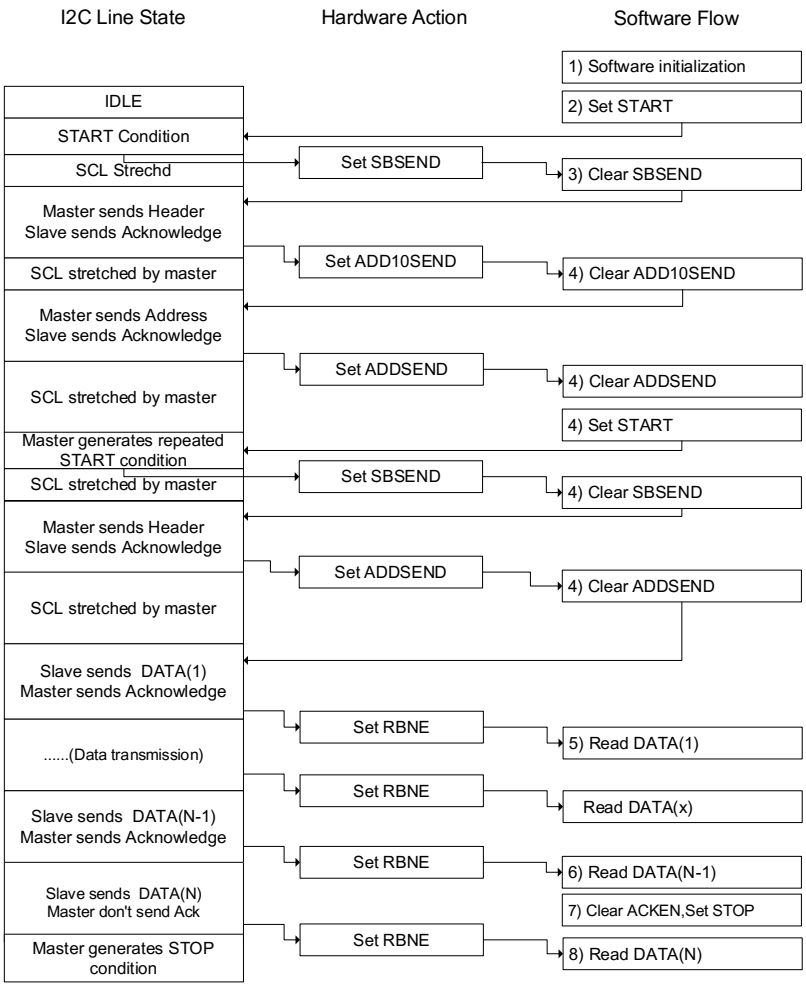
In master receiving mode, a master is responsible for generating NACK for the last byte reception and then sending a STOP signal on I2C bus. So, special attention should be paid to ensure the correct ending of data reception. Two solutions for master receiving are provided here for applications: Solution 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 21-12. 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 21-13. 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

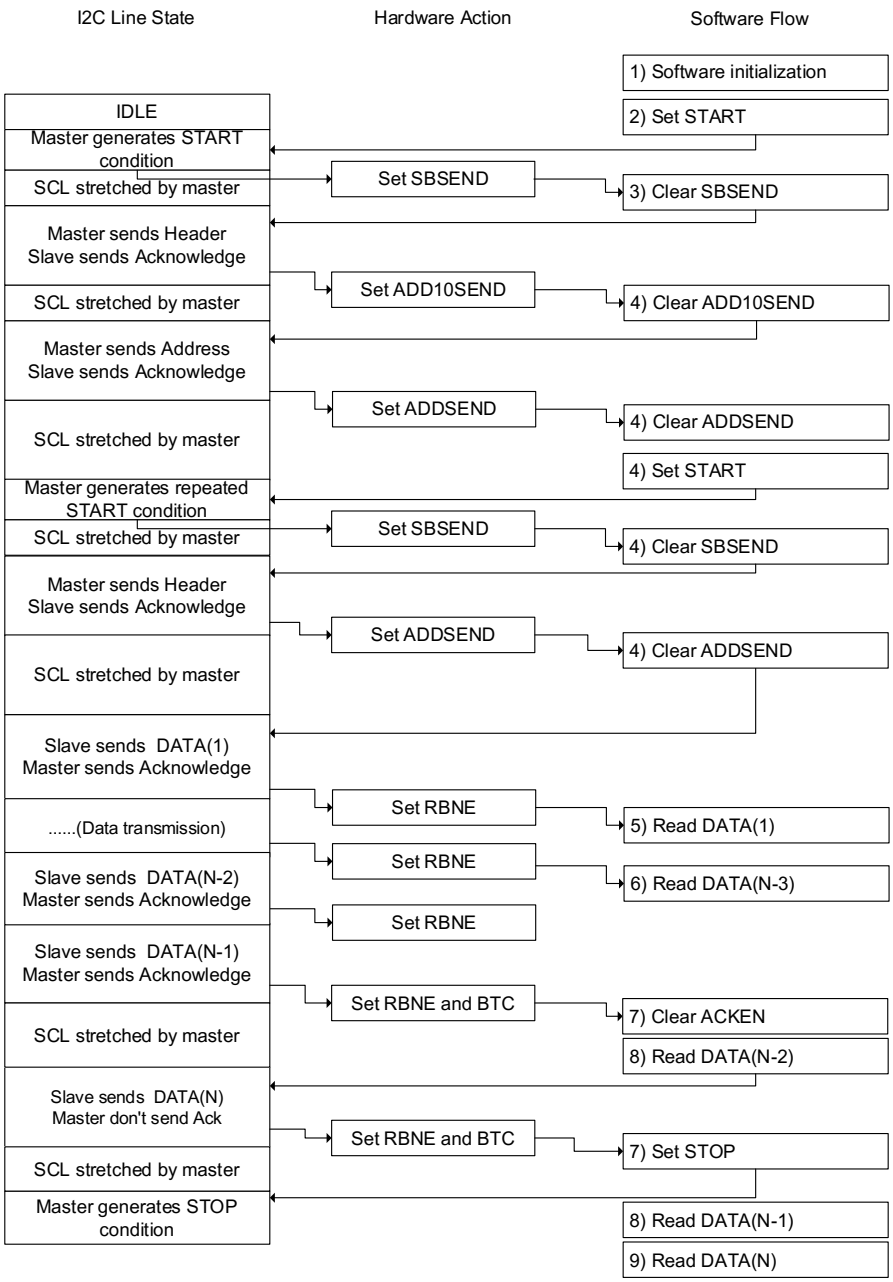
In Step4, software should reset ACKEN bit before clearing ADDSEND bit and set STOP bit after clearing ADDSEND bit. Step 5 is the last step when N=1.

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 21-13. Programming model for master receiving mode using solution B (10-bit

address mode)



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.

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.

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.

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 communication flow

The communication 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.

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.

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 21-2. 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 21-3. 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

I2C debug mode

When the microcontroller enters the debug mode (Cortex®-M23 core halted), the SMBus timeout either continues to work normally or stops, depending on the I2Cx_HOLD (x=0, 1) configuration bits in the DBG module.

21.1.4. Register definition

I2C0 base address: 0x4000 5400

I2C1 base address: 0x4000 5800

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

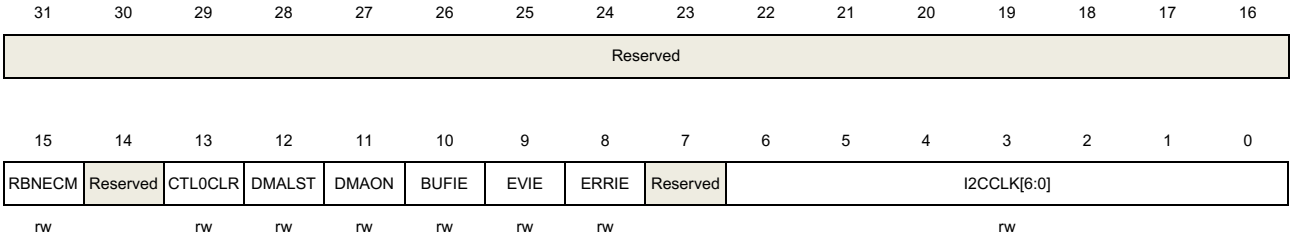
1: I2C is enabled

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
31:16	Reserved	Must be kept at reset value.
15	RBNECM	RBNE clear mode 0: RBNE can be cleared when I2C_DATA is read and BTC is cleared 1: RBNE can be cleared when I2C_DATA is read
14	Reserved	Must be kept at reset value.
13	CTL0CLR	Whether or not to clear I2C_CTL0 when I2CEN=0 0: I2C_CTL0 will be cleared when I2CEN=0 1: I2C_CTL0 will be kept when I2CEN=0
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	Reserved	Must be kept at reset value.
6:0	I2CCLK[6:0]	I2C peripheral clock frequency I2CCLK[6:0] should be the frequency of input APB1 clock in MHz which is at least 2. 0000000 - 0000001: Not allowed 0000010 - 1001000: 2 MHz~72 MHz 1001001 - 1111111: 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.

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

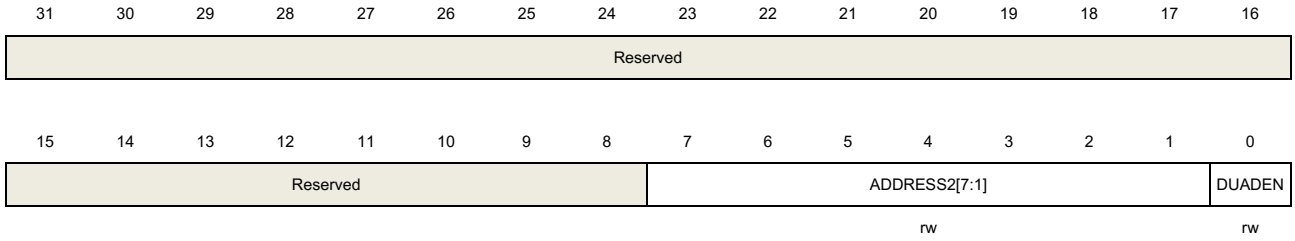
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	ADDFORMAT	Address format for the I2C slave 0: 7-bit address 1: 10-bit address
14:10	Reserved	Must be kept at reset value.
9:8	ADDRESS[9:8]	Highest two bits of a 10-bit address
7:1	ADDRESS[7:1]	7-bit address or bits 7:1 of a 10-bit address
0	ADDRESS0	Bit 0 of a 10-bit address

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).



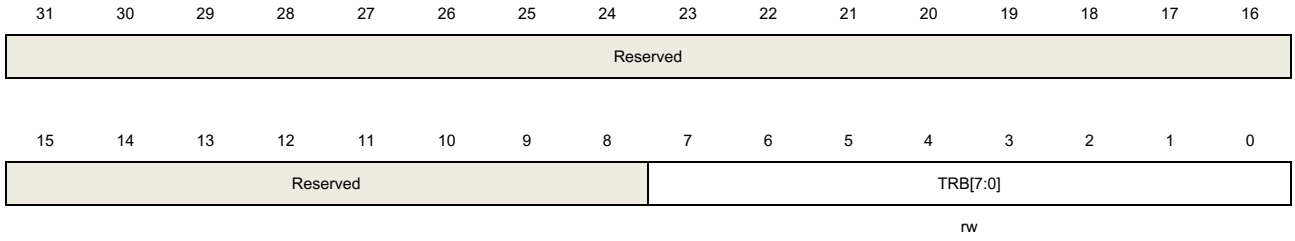
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:1	ADDRESS2[7:1]	The second I2C address for the slave in Dual-Address mode
0	DUADEN	Dual-Address mode enable 0: Dual-Address mode is disabled 1: Dual-Address mode is enabled

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).



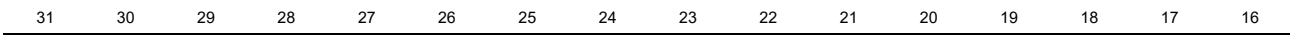
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	TRB[7:0]	Transmission or reception data buffer

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).



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

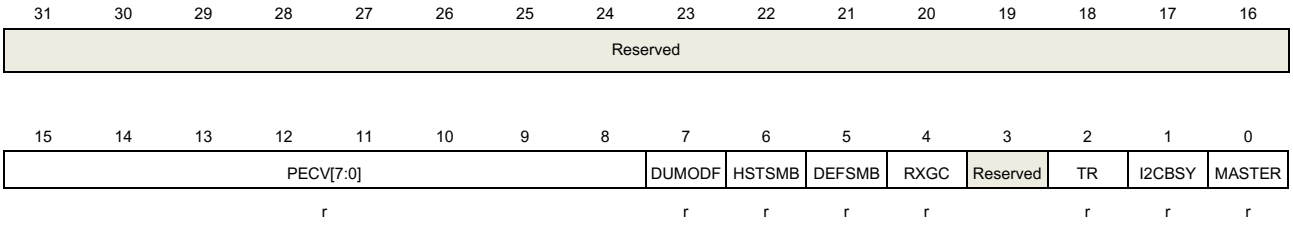
		matches with its own address in slave mode. This bit is set by hardware and cleared by reading I2C_STAT0 and reading I2C_STAT1. 0: In slave mode, no address is received or the received address does not match witih its own address. In master mode, no address is sent or address has been sent but not received 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

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

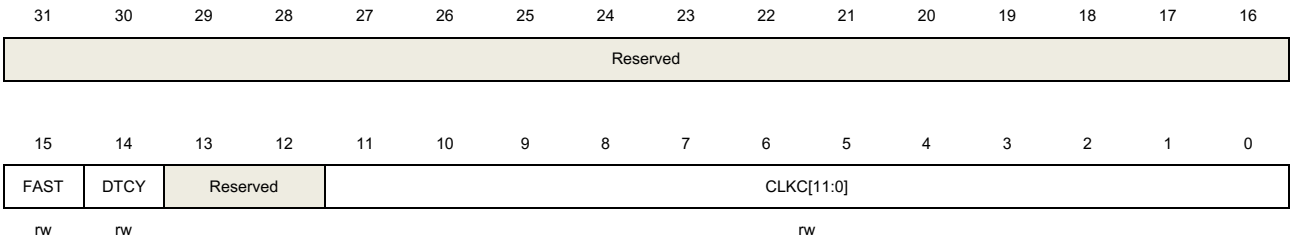
		1: The default address has been received for SMBus device
4	RXGC	General call address (0x00) received. This bit is cleared by hardware after a STOP or a START signal or I2CEN=0. 0: No general call address (0x00) received 1: General call address (0x00) received
3	Reserved	Must be kept at reset value.
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

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).



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.

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[6:0]						
									rw						

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6:0	RISETIME[6:0]	Maximum rise time in master mode The RISETIME value should be the maximum SCL rise time incremented by 1.

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

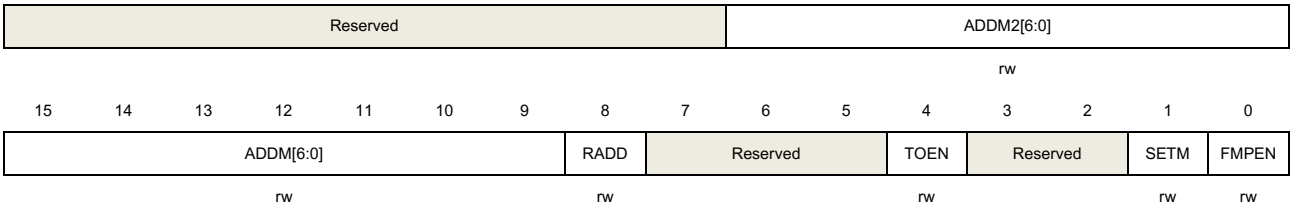
Control register 2 (I2C_CTL2)

Address offset: 0x90

Reset value: 0x0000 FE00

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



Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22:16	ADDM2[6:0]	Defines which bits of register ADDRESS2[7:1] are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in ADDM2[6:0] enables comparisons with the corresponding bit in ADDRESS2[7:1]. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address).
15:9	ADDM[6:0]	Defines which bits of register ADDRESS[7:1] are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in ADDM[6:0] enables comparisons with the corresponding bit in ADDRESS[7:1]. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address).
8	RADD	The received 7bit slave address or the [7:0] bits of the received 10bit slave address will be recorded in the transfer buffer register(I2C_DATA) when this bit is set.
7:5	Reserved	Must be kept at reset value.
4	TOEN	Timeout calculation enable. This bit is used to enable the 25ms SCL low state timeout counter when the SMBA mode is not enabled. 0: disable the SCL low state timeout counter 1: enable the SCL low state timeout counter
3:2	Reserved	Must be kept at reset value.
1	SETM	Start Early Termination Mode. This bit describe how to deal with the low state of the SCL line when I2C is in the SCL high state following the start phase. 0: terminate the SCL high state and start to count the SCL low state. (as the standard I2C protocol) 1: do the same thing as arbitration lost
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

Control and status register (I2C_CS)

Address offset: 0x94

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						STPSENDIE	STLOIE	Reserved						STPSEND	STLO
						rw	rw							rc_w0	rc_w0

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	STPSENDIE	Interrupt enable for stop condition sent 0: interrupt disable 1: interrupt enable
8	STLOIE	Interrupt enable for start lost 0: interrupt disable 1: interrupt enable
7:2	Reserved	Must be kept at reset value.
1	STPSEND	Stop condition sent out in master mode This bit is set by hardware and cleared by software write 0 0: No STOP condition sent 1: STOP condition sent
0	STLO	Start lost flag. This bit is set when low level of the SCL line occurs when I2C is in the start hold timing phase (tHD: STA) if the SETM bit is set to 1. This bit is set by hardware and cleared by software write 0 0: Start lost not occur 1: start lost occurred

Status clear register (I2C_STATC)

Address offset: 0x98

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
SRCEN	Reserved										STOPFC	ADD10SENDC	BTCC	ADDSENDC	SBSENDC
rw											w	w	w	w	w

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	SRCEN	Status register clear enable. When this bit is set, the STOPF, ADD10SEND, BTC, ADDSEND, SBSEND status bits can be cleared by writing 1 to the STOPFC, ADD10SENDC, BTCC, ADDSENDC and SBSENDC registers respectively. Once this bit is set the STOPF, ADD10SEND, BTC, ADDSEND, SBSEND status bits can not be cleared by other software sequences. 0: disable the clear function 1: enable the clear function
14:5	Reserved	Must be kept at reset value.
4	STOPFC	STOPF status clear When SRCEN bit is set to 1, software can clear the STPDET bit of I2C_STAT0 by write 1 to this bit
3	ADD10SENDC	ADD10SEND status clear When SRCEN bit is set to 1, software can clear the ADD10SEND bit of I2C_STAT0 by write 1 to this bit
2	BTCC	BTC status clear When SRCEN bit is set to 1, software can clear the BTC bit of I2C_STAT0 by write 1 to this bit
1	ADDSENDC	ADDSEND status clear When SRCEN bit is set to 1, software can clear the ADDSEND bit of I2C_STAT0 by write 1 to this bit
0	SBSENDC	Start send status clear. When SRCEN bit is set to 1, software can clear the SBSEND bit of I2C_STAT0 by write 1 to this bit

21.2. Inter-integrated circuit interface (I2C2)

21.2.1. 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), and PMBus (power management bus). It also supports multi-master I2C bus. The I2C interface provides DMA mode for users to reduce CPU overload.

21.2.2. 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.
- Multiple 7-bit slave addresses (2 addresses with configurable mask).
- Programmable setup time and hold time.
- Multi-master capability.
- Supports standard mode (up to 100 kHz), fast mode (up to 400 kHz) and fast mode plus (up to 1MHz, must enable Fm+ in SYSCFG_CFG0 register).
- Configurable SCL stretching in slave mode.
- Supports DMA mode.
- Compatible with SMBus 3.0 and PMBus 1.3.
- Optional PEC (packet error checking) generation and check.
- Programmable analog and digital noise filters.
- Wakeup from Sleep mode and Deep-sleep mode on I2C address match.
- Independent clock from PCLK.

21.2.3. Function overview

Figure 21-14. I2C module block diagram below provides details on the internal configuration of the I2C interface.

Figure 21-14. I2C module block diagram

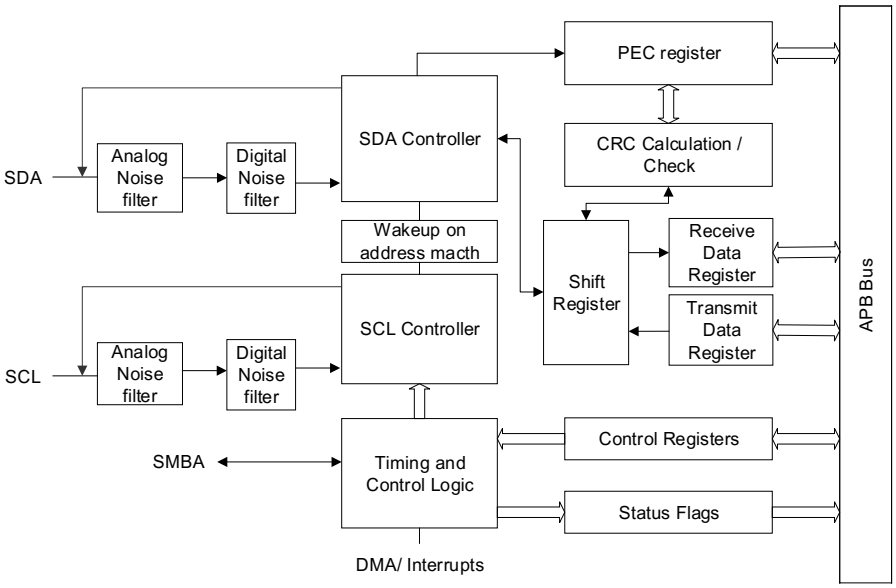


Table 21-4. Definition of I2C-bus terminology (refer to the I2C specification of PHILIPS semiconductors)

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
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

Clock requirements

The I2C clock is independent of the PCLK frequency, so that the I2C can be operated independently.

This I2C clock (I2CCLK) can be selected from the following three clock sources:

- IRC8M: internal 8 MHz RC (default value)
- PCLK1: APB1 clock
- SYSCLK: system clock
- HXTAL: High speed crystal oscillator

The I2CCLK period t_{I2CCLK} must match the conditions as follows:

- $t_{I2CCLK} < (t_{LOW} - t_{filters})/4$

- $t_{I2CCLK} < t_{HIGH}$

with:

t_{LOW}: SCL low time

t_{HIGH}: SCL high time

$t_{filters}$: When the filters are enabled, represent the delays by the analog filter and digital filter.

Analog filter delay is maximum 120ns. Digital filter delay is DNF[3:0]×t_{I2CCLK}.

The period of PCLK clock t_{PCLK} match the conditions as follows:

- $t_{PCLK} < 4/3 * t_{SCL}$

with:

t_{SCL}: the period of SCL

Note: When the I2C kernel is provided by PCLK, this clock must match the conditions for t_{I2CCLK}.

I2C communication flow

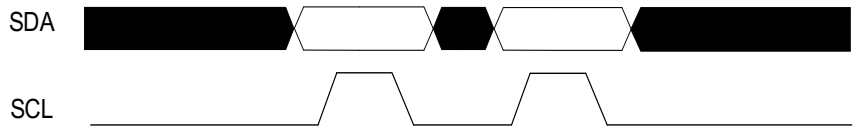
An I2C device is able to transmit or receive data whether it's a master or a slave, thus, there're 4 operation modes for an I2C device:

- Slave transmitter
- Slave receiver
- Master transmitter
- Master receiver

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 data line can only change when the clock signal on the SCL line is LOW (see ***Figure 21-15. Data validation***). One clock pulse is generated for each data bit transferred.

Figure 21-15. Data validation

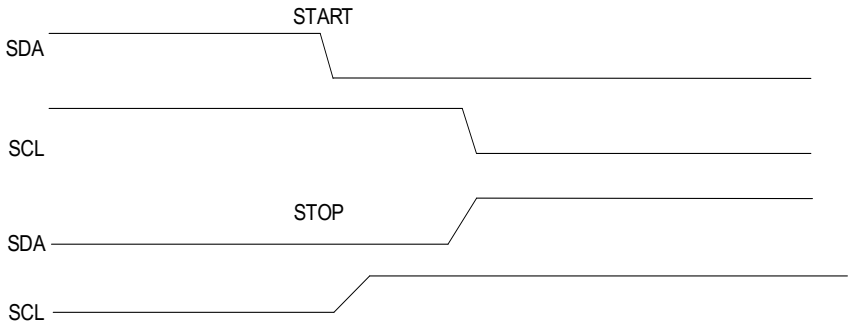


START and STOP signal

All transactions begin with a START and are terminated by a STOP (see ***Figure 21-16. 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 21-16. START and STOP signal



Each I2C device is recognized by a unique address (whether it is a microcontroller, LCD driver, memory or keyboard interface) and can operate as either a transmitter or receiver, depending on the function of the device. It operates in slave mode by default. When it generates a START signal, the interface automatically switches from slave to master. If an arbitration loss or a STOP generation occurs, then the interface switches from master to slave, allowing multimaster capability.

An I2C slave will continue to detect addresses after a START signalon I2C bus and compare the detected address with its slave address which is programmable by software. Once the two addresses match, the I2C slave will send an ACK to the I2C bus and responses 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 responses to a General Call Address (0x00). The I2C block support both 7-bit and 10-bit address modes.

Data and addresses are transferred as 8-bit bytes, MSB first. The first byte(s) following the START signalcontain the address (one in 7-bit mode, two in 10-bit mode). The address is always transmitted in master mode.

A 9th clock pulse follows the 8 clock cycles of byte transmission, during which the receiver must send an acknowledge bit to the transmitter. Acknowledge can be enabled or disabled by software.

An I2C master always initiates or end a transfer using START or STOP signaland it's also responsible for SCL clock generation.

In master mode, if AUTOEND = 1, the STOP signalis generated automatically by hardware. If AUTOEND = 0, the STOP signalgenerated by software, or the master can generate a RESTART signalto start a new transfer.

Figure 21-17. I2C communication flow with 10-bit address (Master Transmit)

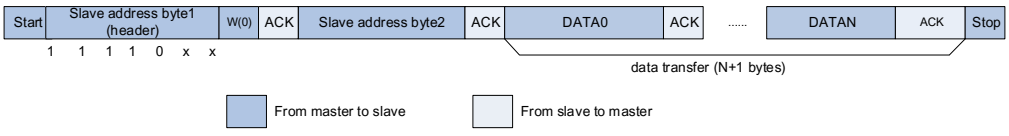


Figure 21-18. I2C communication flow with 7-bit address (Master Transmit)

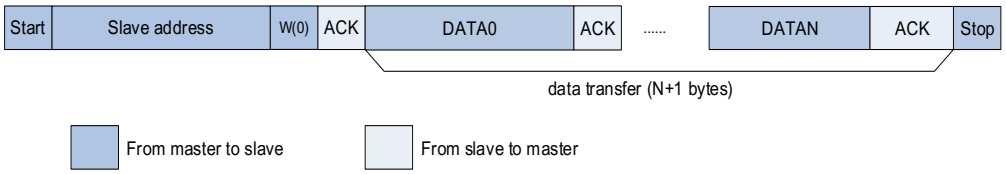
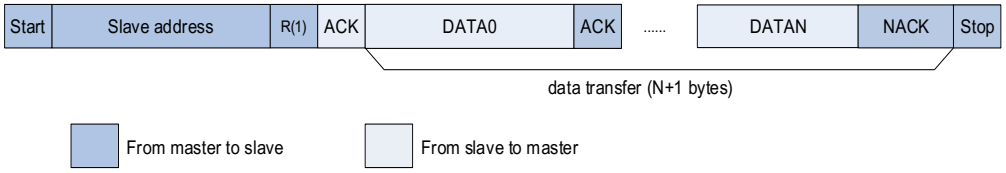


Figure 21-19. I2C communication flow with 7-bit address (Master Receive)



In 10-bit addressing mode, the HEAD10R bit can configured to decide whether the complete address sequence must be executed, or only the header to be sent. When HEAD10R = 0, the complete 10 bit address read sequence must be excuted with START + header of 10-bit address in write direction + slave address byte 2 + RESTART + header of 10-bit address in read direction, as is shown in ***Figure 21-20. I2C communication flow with 10-bit address (Master Receive when HEAD10R=0)***.

In 10-bit addressing mode, if the master reception follows a master transmission between the same master and slave, the address read sequence can be RESTART + header of 10-bit address in read direction, as is shown in ***Figure 21-21. I2C communication flow with 10-bit address (Master Receive when HEAD10R=1)***.

Figure 21-20. I2C communication flow with 10-bit address (Master Receive when HEAD10R=0)

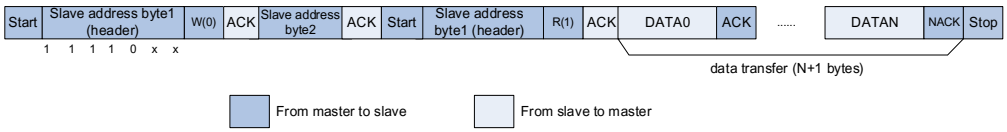
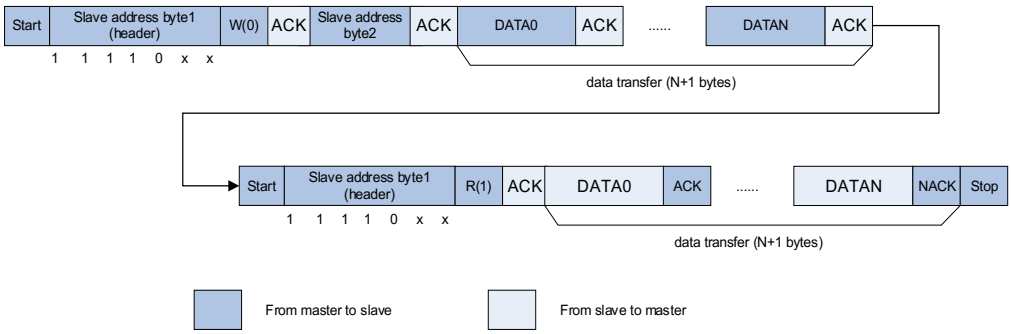


Figure 21-21. I2C communication flow with 10-bit address (Master Receive when HEAD10R=1)



Noise filter

Analog noise filter and digital noise filter are integrated in I2C peripheral, the noise filters can be configured before the I2C peripheral is enabled according to the actual requirements.

The analog noise filter is disabled by setting the ANOFF bit in I2C_CTL0 register and enabled when ANOFF is 0. It can suppress spikes with a pulse width up to 50ns in fast mode and fast mode plus.

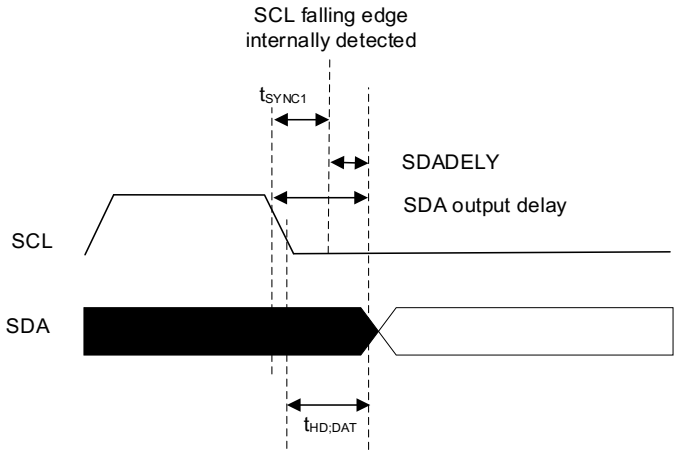
The digital noise filter can be used by configuring the DNF[3:0] bit in I2C_CTL0 register. The level of the SCL or the SDA will be changed if the level is stable for more than DNF[3:0]×t_{I2CCLK}. The length of spikes to be suppressed is configured by DNF[3:0].

I2C timings configuration

The PSC[3:0], SCLDELY[3:0] and SDADELY[3:0] bits in the I2C_TIMING register must be configured in order to guarantee a correct data hold and setup time used in I2C communication.

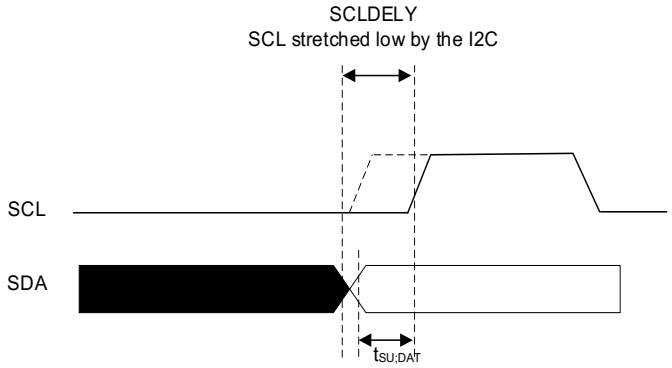
If the data is already available in I2C_TDATA register, the data will be sent on SDA after the SDADELY delay. As is shown in ***Figure 21-22. Data hold time***.

Figure 21-22. Data hold time



The SCLDELY counter starts when the data is sent on SDA output. As is shown in ***Figure 21-23. Data setup time***.

Figure 21-23. Data setup time



When the SCL falling edge is internally detected, a delay is inserted before sending SDA output. This delay is $t_{SDADELY}$=SDADELY*t_{PSC}+t_{I2CCLK} where t_{PSC}=(PSC+1)*t_{I2CCLK}. $t_{SDADELY}$ effects $t_{HD;DAT}$. The total delay of SDA output is t_{SYNC1}+{[SDADELY*(PSC+1)+1]*t_{I2CCLK}}. t_{SYNC1} depends on SCL falling slope, the delay of analog filter, the delay of digital filter and delay of SCL synchronization to I2CCLK clock. The delay of SCL synchronization to I2CCLK clock is 2 to 3 t_{I2CCLK}.

Note: When SDADELY = 0, $t_{SDADELY}$ = $t_{PSC.}$

SDADELY must match condition as follows:

- SDADELY>={t_f(max)+$t_{HD;DAT}$(min)-t_{AF}(min)-[(DNF+3)*t_{I2CCLK}]}/[(PSC+1)*t_{I2CCLK}]
- SDADELY<={$t_{HD;DAT}$(max)-t_{AF}(max)-[(DNF+4)*t_{I2CCLK}]}/[(PSC+1)*t_{I2CCLK}]

Note: t_{AF} is the delay of analog filter. The $t_{HD;DAT}$ should be less than the maximum of $t_{VD;DAT}$.

When SS = 0, after $t_{SDADELY}$ delay, the slave had to stretch the clock before the data writing to I2C_TDATA register, SCL is low during the data setup time. The setup time is $t_{SCLDELY}$=(SCLDELY+1)*t_{PSC}. $t_{SCLDELY}$ effects $t_{SU;DAT}$.

SCLDELY must match condition as follows:

- SCLDELY>={[t_r(max)+$t_{SU;DAT}$(min)]/[(PSC+1)*t_{I2CCLK}]}-1

In master mode, the SCL clock high and low levels must be configured by programming the PSC[3:0], SCLH[7:0] and SCLL[7:0] bits in the I2C_TIMING register.

When the SCL falling edge is internally detected, a delay is inserted before releasing the SCL output. This delay is t_{SCLL}=(SCLL+1)*t_{PSC} where t_{PSC}=(PSC+1)*t_{I2CCLK}. t_{SCLL} impacts the SCL low time t_{LOW}.

When the SCL rising edge is internally detected, a delay is inserted before forcing the SCL output to low level. This delay is t_{SCLH}=(SCLH+1)*t_{PSC} where t_{PSC}=(PSC+1)*t_{I2CCLK}. t_{SCLH} impacts the SCL high time t_{HIGH}.

Note: When the I2C is enabled, the timing configuration and SS mode must not be changed.

Table 21-5. Data setup time and data hold time

Symbol	Parameter	Standard mode		Fast mode		Fast mode plus		SMBus		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
$t_{HD;DAT}$	Data hold time	0	-	0	-	0	-	0.3	-	us
$t_{VD;DAT}$	Data valid time	-	3.45	-	0.9	-	0.45	-	-	
$t_{SU;DAT}$	Data setup time	250	-	100	-	50	-	250	-	ns
t_r	Rising time of SCL and SDA	-	1000	-	300	-	120	-	1000	
t_f	falling time of SCL and SDA	-	300	-	300	-	120	-	300	

I2C reset

A software reset can be performed by clearing the I2CEN bit in the I2C_CTL0 register. When a software reset is generated, the SCL and SDA are released. The communication control bits and status bits come back to the reset value. Software reset have no effect on configuration registers. The impacted register bits are START, STOP, NACKEN in I2C_CTL1 register, I2CBSY, TBE, TI, RBNE, ADDSEND, NACK, TCR, TC, STPDET, BERR, LOSTARB and OUERR in I2C_STAT register. Additionally, when the SMBus is supported, PECTRANS in I2C_CTL1 register, PECERR, TIMEOUT and SMBALT in I2C_STAT are also impacted.

In order to perform the software reset, I2CEN must be kept low during at least 3 APB clock cycles. This is ensured by writing software sequence as follows:

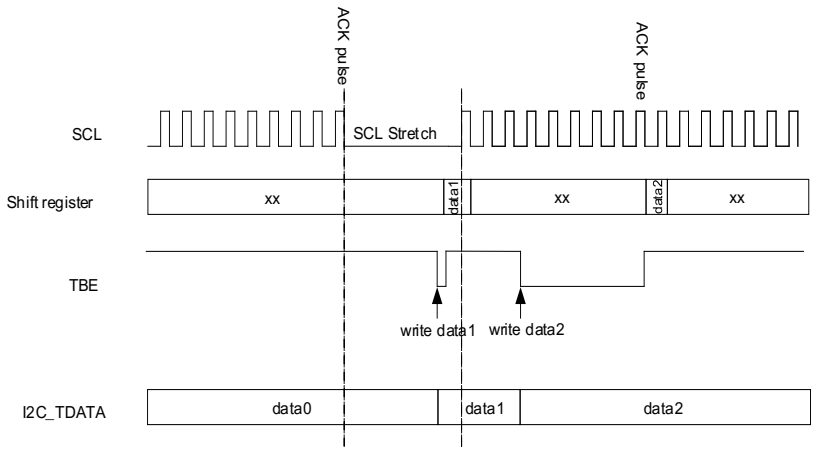
- Write I2CEN = 0
- Check I2CEN = 0
- Write I2CEN = 1

Data transfer

Data Transmission

When transmitting data, if TBE is 0, it indicates that the I2C_TDATA register is not empty, the data in I2C_TDATA register is moved to the shift register after the 9th SCL pulse. Then the data will be transmitted through the SDA line from the shift register. If TBE is 1, it indicates that the I2C_TDATA register is empty, the SCL line is stretched low until I2C_TDATA is not empty. The stretch begins after the 9th SCL pulse.

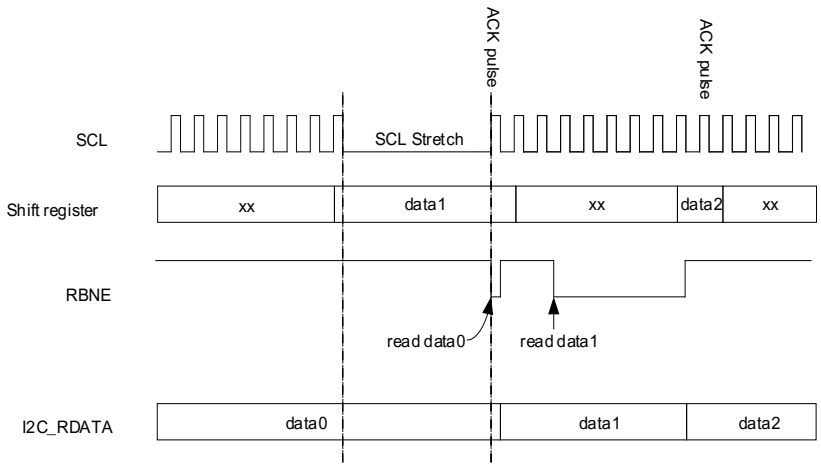
Figure 21-24. Data transmission



Data Reception

When receiving data, the data will be received in the shift register first. If RBNE is 0, the data in the shift register will move into I2C_RDATA register. If RBNE is 1, the SCL line wii be stretched until the previous received data in I2C_RDATA is read. The stretch is inserted before the acknowledge pulse.

Figure 21-25. Data reception



Reload and automatic end mode

In order to manage byte transfer and to shut down the communication in modes as is shown in ***Table 21-6. Communication modes to be shut down***, the I2C embedded a byte counter in the hardware.

Table 21-6. Communication modes to be shut down

Working mode	Action
Master mode	NACK, STOP and RESTART generation
Slave receiver mode	ACK control

Working mode	Action
SMBus mode	PEC generation/checking

The number of bytes to be transferred is configured by BYTENUM[7:0] in I2C_CTL1 register. If BYTENUM is greater than 255, or in slave byte control mode, the reload mode must be enabled by setting the RELOAD bit in I2C_CTL1 register. In reload mode, when BYTENUM counts to 0, the TCR bit will be set, and an interrupt will be generated if TCIE is set. Once the TCR flag is set, SCL is stretched. The TCR bit is cleared by writing a non-zero number in BYTENUM.

Note: The reload mode must be disabled after the last reloading of BYTENUM[7:0].

The reload mode must be disabled when the automatic end mode is enabled. In automatic end mode, the master will send a STOP signal automatically when the BYTENUM[7:0] counts to 0.

When reload mode and automatic end mode are disabled, the I2C communication process needs to be terminated by software. If the number of bytes in BYTENUM[7:0] has been transferred, the STOP bit should be set by software to generate a STOP signal, and then TC flag must be cleared.

I2C slave mode

Initialization

When works in slave mode, at least one slave address should be enabled. Slave address 1 can be programmed in I2C_SADDR0 register and slave address 2 can be programmed in I2C_SADDR1 register. ADDRESSEN in I2C_SADDR0 register and ADDRESS2EN in I2C_SADDR1 register should be set when the corresponding address is used. 7-bit address or 10-bit address can be programmed in ADDRESS[9:0] in I2C_SADDR0 register by configuring the ADDFORMAT bit in 7-bit address or 10-bit address.

The ADDM[6:0] in I2C_CTL2 register defines which bits of ADDRESS[7:1] are compared with an incoming address byte, and which bits are ignored.

The ADDMSK2[2:0] is used to mask ADDRESS2[7:1] in I2C_SADDR1 register. For details, refer to the description of ADDMSK2[2:0] in I2C_SADDR1 register.

When the I2C received address matches one of its enabled addresses, the ADDSEND will be set, and an interrupt is generated if the ADDMIE bit is set. The READDR[6:0] bits in I2C_STAT register will store the received address. And TR bit in I2C_STAT register updates after the ADDSEND is set. The bit will let the slave to know whether to act as a transmitter or receiver.

SCL line stretching

The clock stretching is used in slave mode by default (SS = 0), the SCL line can be stretched low if necessary. The SCL will be stretched in following cases.

- The SCL is stretched when the ADDSEND bit is set, and released when the ADDSEND bit is cleared.
- In slave transmitting mode, after the ADDSEND bit is cleared, the SCL will be stretched before the first data byte writing to the I2C_TDATA register. Or the SCL will be stretched before the new data is written to the I2C_TDATA register after the previous data transmission is completed.
- In slave receiving mode, a new reception is completed but the data in I2C_RDATA register has not been read.
- When SBCTL = 1 and RELOAD = 1, after the transfer of the last byte, TCR is set. Before the TCR is cleared, the SCL will be stretched.
- The I2C stretches SCL low during $[(SDADELY+SCLDELY+1)*(PSC+1)+1]*t_{I2CCLK}$ after detecting the SCL falling edge.

The clock stretching can be disabled by setting the SS bit in I2C_CTL0 register (SS = 1). The SCL will not be stretched in following cases.

- When the ADDSEND is set, the SCL will be not stretched.
- In slave transmitting mode, before the first SCL pulse, the data must be written in the I2C_TDATA register . Or else the OUERR bit in the I2C_STAT register will be set, if the ERRIE bit is set, an interrupt will be generated. When the STPDET bit is set and the first data transmission starts, OUERR bit in the I2C_STAT register will also be set.
- In slave receiving mode, before the 9th SCL pulse (ACK pulse) occurred by the next data byte, the data must be read out from the I2C_RDATA register. Or else the OUERR bit in the I2C_STAT register will be set, if the ERRIE bit is set, and an interrupt will be generated.

Slave byte control mode

In slave receiving mode, the slave byte control mode can be enabled by setting the SBCTL bit in the I2C_CTL0 register to allow byte ACK control. When SS = 1, the slave byte control mode is not allowed.

When using slave byte control mode, the reload mode must be enabled by setting the RELOAD bit in I2C_CTL1 register. In slave byte control mode, BYTENUM[7:0] in I2C_CTL1 register must be configured as 1 in the ADDSEND interrupt service routine and reloaded to 1 after each byte received. The TCR bit in I2C_STAT register will be set when a byte is received, the SCL will be stretched low by slave between the 8th and 9th clock pulses. Then the data can be read from the I2C_RDATA register, and the slave determines to send an ACK or a NACK by configuring the NACKEN bit in the I2C_CTL1 register. When the BYTENUM[7:0] is written a non-zero value, the slave will release the stretch.

When the BYTENUM[7:0] is greater than 0x1, there is no stretch between the reception of two data bytes.

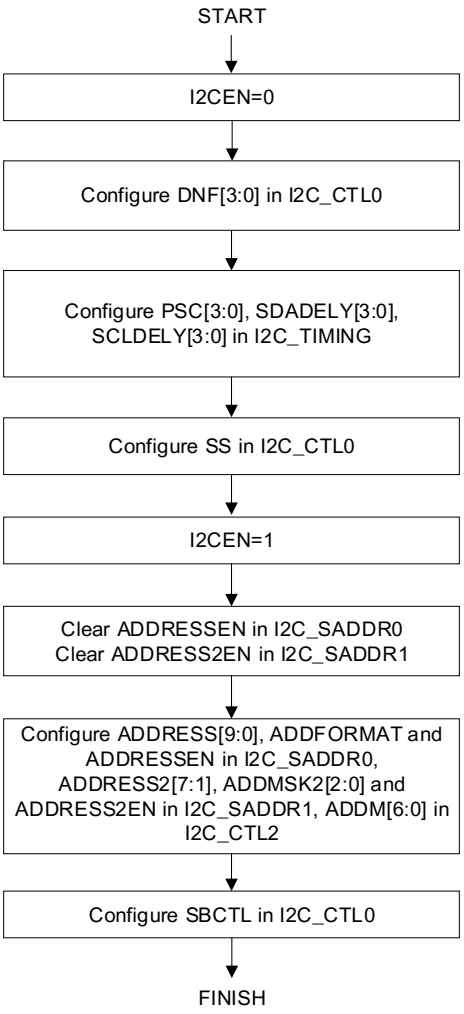
Note: The SBCTL bit can be configured in following cases:

1. I2CEN = 0.

2. The slave has not been addressed.
3. ADDSEND = 1.

Only when the ADDSEND = 1 or TCR = 1, the RELOAD bit can be modified.

Figure 21-26. I2C initialization in slave mode



Programming model in slave transmitting mode

When the I2C_TDATA register is empty, the TI bit in I2C_STAT register will be set. If the TIE bit in I2C_CTL0 register is set, an interrupt will be generated. The NACK bit in I2C_STAT register will be set when a NACK is received. And an interrupt is generated if the NACKIE bit is set in the I2C_CTL0 register. The TI bit in I2C_STAT register will not be set when a NACK is received.

The STPDET bit in I2C_STAT register will be set when a STOP is received. If the STPDETIE in I2C_CTL0 register is set, an interrupt will be generated.

When SBCTL is 0, if ADDSEND = 1, and the TBE bit in I2C_STAT register is 0, the data in I2C_TDATA register can be chosed to be transmitted or flushed. The data is flushed by setting

the TBE bit.

When SBCTL = 1, the slave works in slave byte control mode, the BYTENUM[7:0] must be configured in the ADDSEND interrupt service routine. And the number of TI events is equal to the value of BYTENUM[7:0].

When SS = 1, the SCL will not be stretched when ADDSEND bit in I2C_STAT register is set. In this case, the data in I2C_TDATA register can not be flushed in ADDSEND interrupt service routine. So the first byte to be sent must be programmed in the I2C_TDATA register previously.

- This data can be the one which is written in the last TI event of the last transfer.
- Setting the TBE bit can flush the data if it is not the one to be sent, then a new byte can be written in I2C_TDATA register. The STPDET must be 0 when the data transmission begins. Or else the OUERR bit in I2C_STAT register will be set and an underrun error occurs.
- When interrupt or DMA is used in slave transmitter, if a TI event is needed, in order to generate a TI event both the TI bit and the TBE bit must be set.

Figure 21-27. Programming model for slave transmitting when SS=0

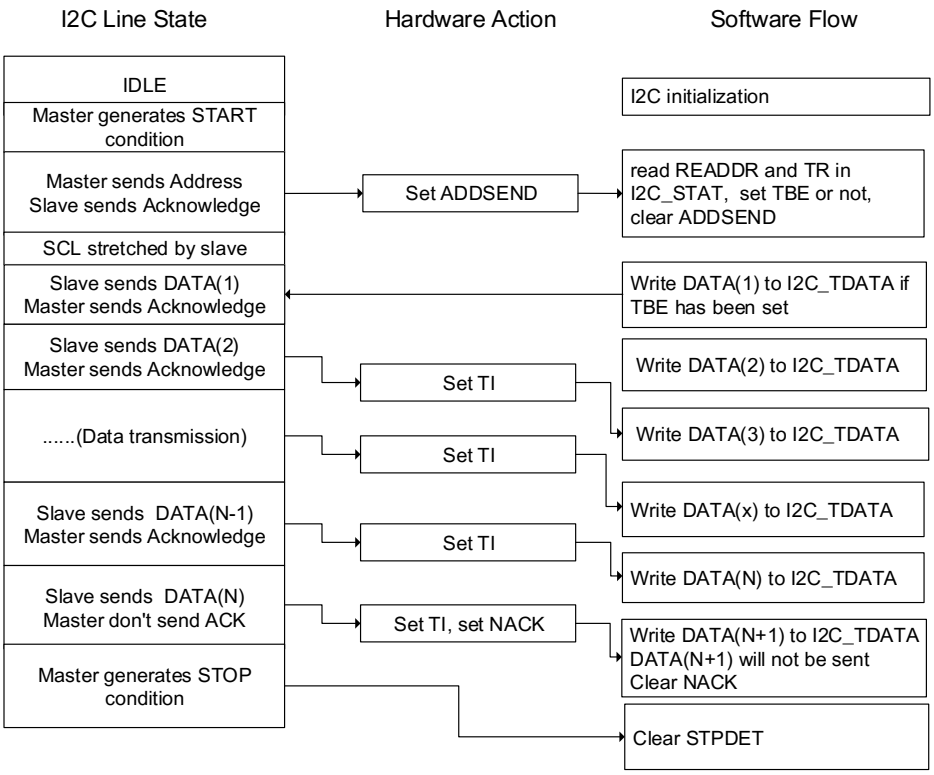
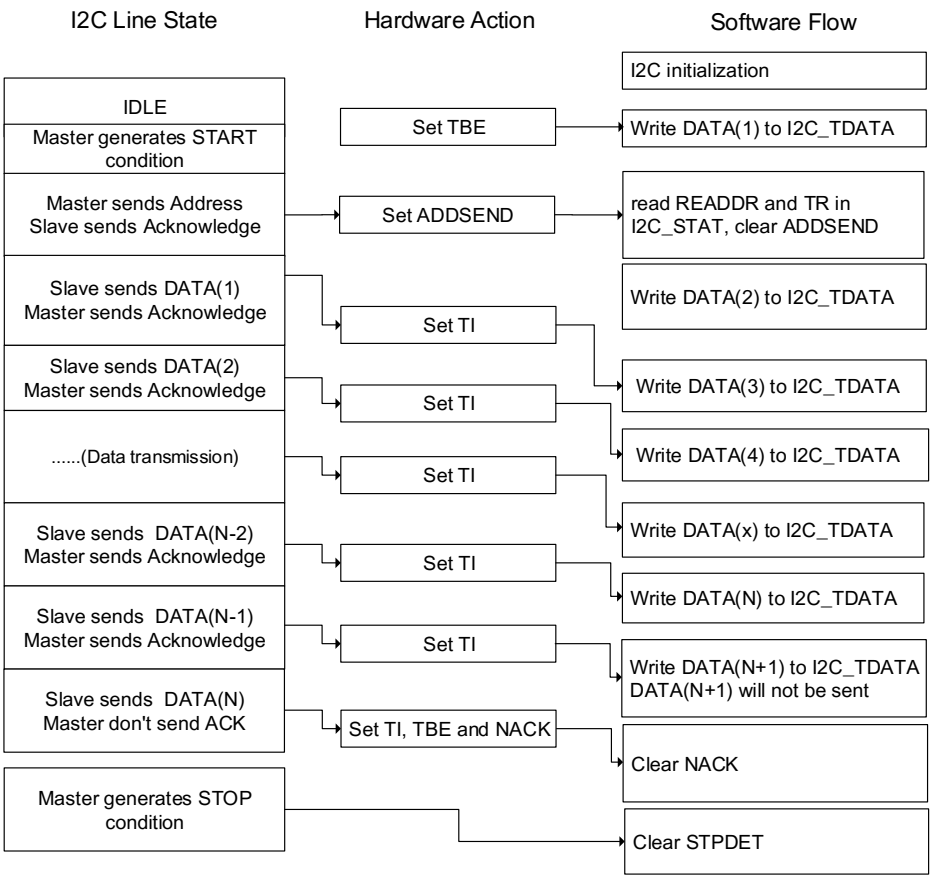


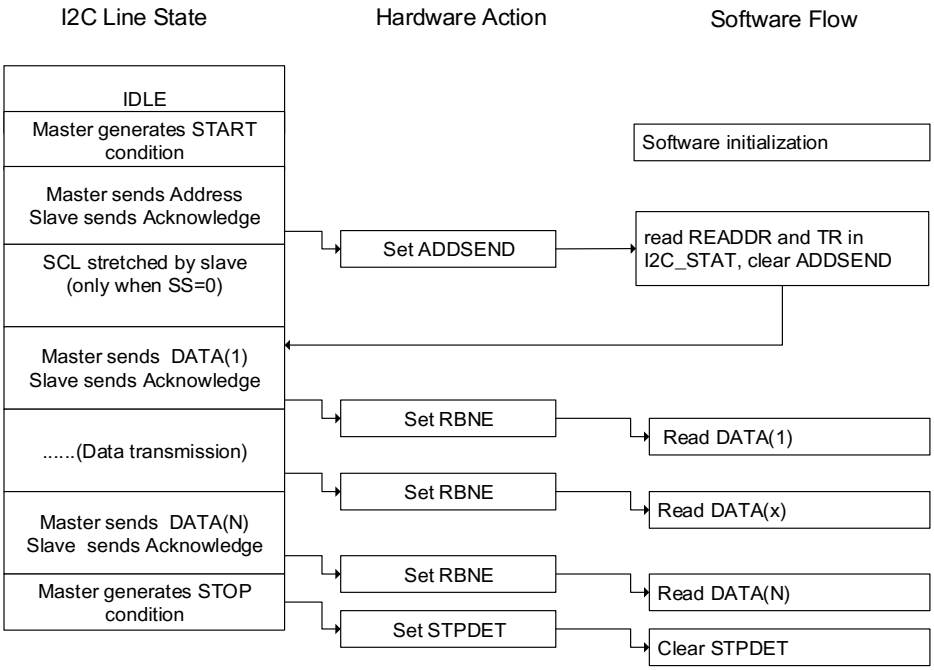
Figure 21-28. Programming model for slave transmitting when SS=1



Programming model in slave receiving mode

When the I2C_RDATA is not empty, the RBNE bit in I2C_STAT register is set, and if the RBNEIE bit in I2C_CTL0 register is set, an interrupt will be generated. When a STOP is received, STPDET will be set in I2C_STAT register. If the STPDETIE bit in I2C_CTL0 register is set, and an interrupt will be generated.

Figure 21-29. Programming model for slave receiving



I2C master mode

Initialization

The SCLH[7:0] and SCLL[7:0] in I2C_TIMING register should be configured when I2CEN is 0. In order to support multi-master communication and slave clock stretching, a clock synchronization mechanism is implemented.

The SCLL[7:0] and SCLH[7:0] are used for the low level counting and high level couting respectively. After a t_{SYNC1} delay, when the SCL low level is detected, the SCLL[7:0] starts counting, if the SCLL[7:0] in I2C_TIMING register is reached by SCLL[7:0] counter, the I2C will release the SCL clock. After a t_{SYNC2} delay, when the SCL high level is detected, the SCLH[7:0] starts counting, if the SCLH[7:0] in I2C_TIMING register is reached by SCLH[7:0] counter, the I2C will stretch the SCL clock.

So the master clock period is:

$t_{SCL}=t_{SYNC1}+t_{SYNC2}+\{[(SCLH[7:0]+1)+(SLLL[7:0]+1)]*(PSC+1)*t_{I2CCLK}\}$.

The t_{SYNC1} depends on the SCL falling slope, delay by input analog and digital noise filter and SCL synchronization with I2CCLK clock, which generally 2 to 3 I2CCLK periods. The t_{SYNC2} depends on the SCL rising slope, delay by input analog and digital noise filter and SCL synchronization with I2CCLK clock, which generally 2 to 3 I2CCLK periods. The delay by digital noise filter is DNF[3:0]*t_{I2CCLK}.

When works in master mode, the ADD10EN bit, SADDRESS[9:0] bits, TRDIR bit should be configured in I2C_CTL1 register. When the addressing mode is 10-bit in master receiving

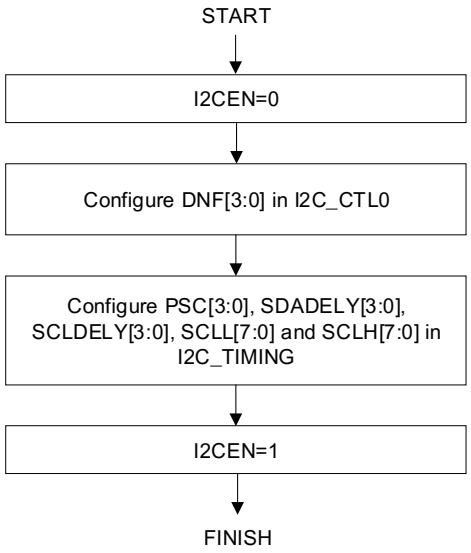
mode, the HEAD10R bit must be configured to decide whether the complete address sequence must be executed, or only the header to be sent. The number of bytes to be transferred should be configured in BYTENUM[7:0] in I2C_CTL1 register. If the number of bytes to be transferred is equal to or greater than 255, BYTENUM[7:0] should be configured as 0xFF. Then the master sends the START signal. All the bits above should be configured before the START is set. The slave address will be sent after the START signal when the I2CBSY bit I2C_STAT register is detected as 0. When the arbitration is lost, the master changes to slave mode and the START bit will be cleared by hardware. When the slave address has been sent, the START bit will be cleared by hardware.

Note: BYTENUM[7:0] can not be configured as 0.

In 10-bit addressing mode, if the master receives a NACK after the transmission of 10-bit header, the master will resend it until ACK is received. The ADDSENDC bit must be set to stop sending the slave address.

If the START bit is set, meanwhile the ADDSEND is set by addressing as a slave, the master changes to slave mode. The ADDSENDC bit must be set to clear the START bit.

Figure 21-30. I2C initialization in master mode



Programming model in master transmitting mode

In master transmitting mode, the TI bit is set after the ACK is received of each byte transmission. If the TIE bit in I2C_CTL0 register is set, an interrupt will be generated. The bytes to be transferred is programmed in BYTENUM[7:0] in I2C_CTL1 register. If the bytes to be transferred is greater than 255, RELOAD bit in I2C_CTL1 register must be set to enable the reload mode. In reload mode, when data of BYTENUM[7:0] bytes have been transferred, the TCR bit in I2C_STAT register will be set and the SCL stretches unitil BYTENUM[7:0] is modified with a non-zero value.

When a NACK is received, the TI bit will not set.

- If data of BYTENUM[7:0] bytes have been transferred and RELOAD = 0, the AUTOEND bit in I2C_CTL1 can be set to generate a STOP signal automatically. When AUTOEND is 0, the TC bit in I2C_STAT register will be set and the SCL is stretched. In this case, the master can generate a STOP signal by setting the STOP bit in the I2C_CTL1 register. Or generate a RESTART signal to start a new transfer. The TC bit is cleared when the START / STOP bit is set.
- If a NACK is received, a STOP signal is automatically generated, the NACK is set in I2C_STAT register, if the NACKIE bit is set, an interrupt will be generated.

Note: When the RELOAD bit is 1, the AUTOEND has no effect.

Figure 21-31. Programming model for master transmitting (N<=255)

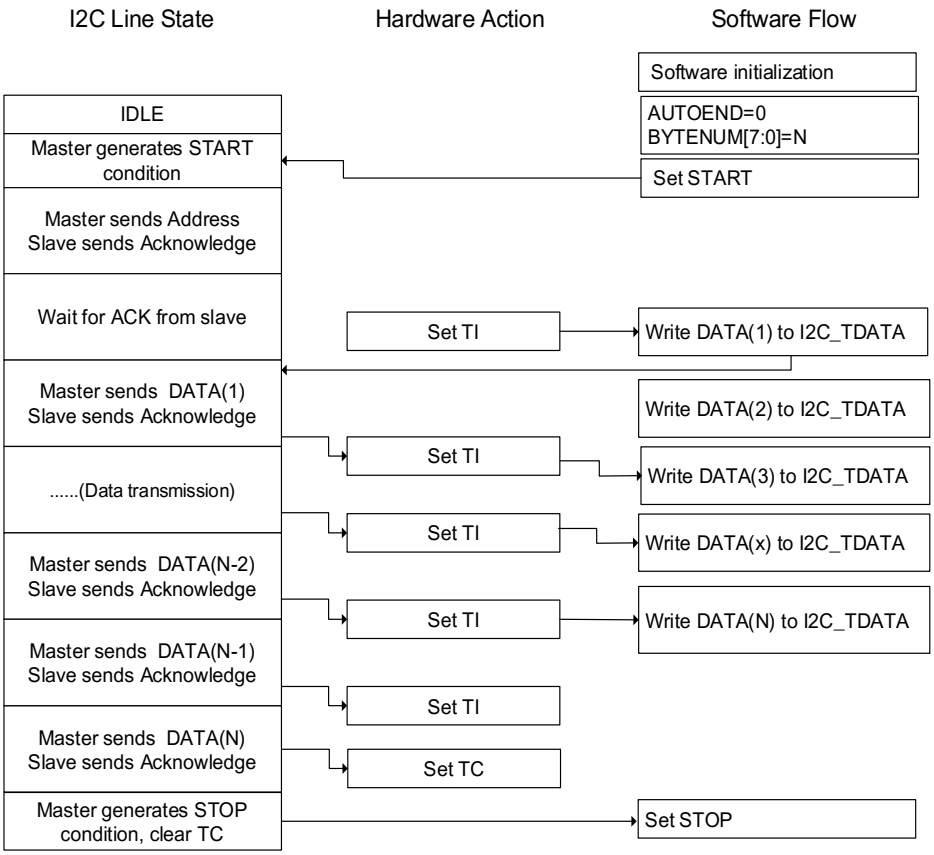
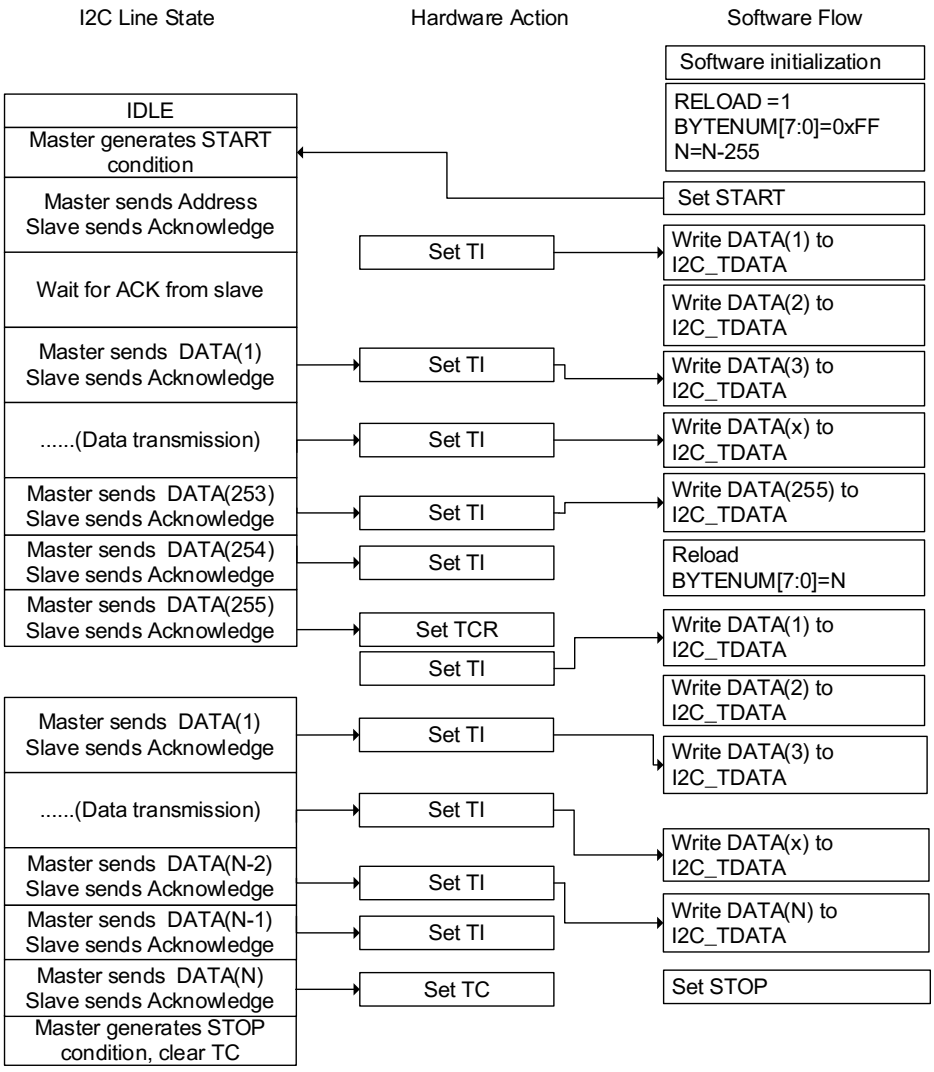


Figure 21-32. Programming model for master transmitting (N>255)



Programming model in master receiving mode

In master receiving mode, the RBNE bit in I2C_STAT register will be set when a byte is received. If the RBNEIE bit is set in I2C_CTL0 register, an interrupt will be generated. If the number of bytes to be received is greater than 255, RELOAD bit in I2C_CTL1 register must be set to enable the reload mode. In reload mode, when data of BYTENUM[7:0] bytes have been transferred, the TCR bit in I2C_STAT register will be set and the SCL stretches unitil BYTENUM[7:0] is modified with a non-zero value.

If data of BYTENUM[7:0] bytes have been transferred and RELOAD = 0, the AUTOEND bit in I2C_CTL1 can be set to generate a STOP signal automatically. When AUTOEND is 0, the TC bit in I2C_STAT register will be set and the SCL is stretched. In this case, the master can generate a STOP signal by setting the STOP bit in the I2C_CTL1 register. Or generate a RESTART signal to start a new transfer. The TC bit is cleared when the START / STOP bit is

set.

Figure 21-33. Programming model for master receiving (N<=255)

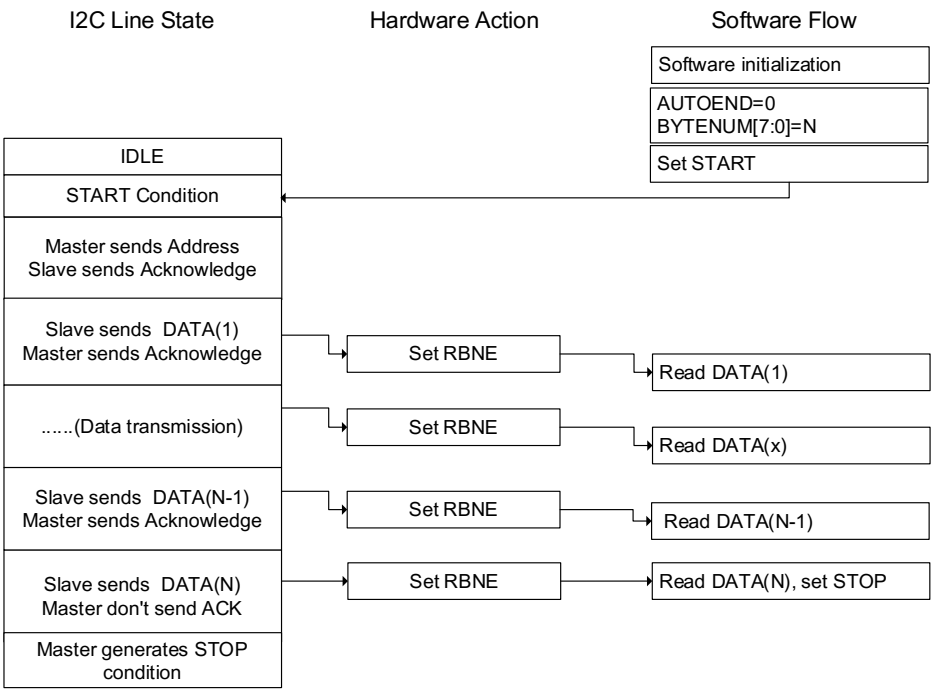
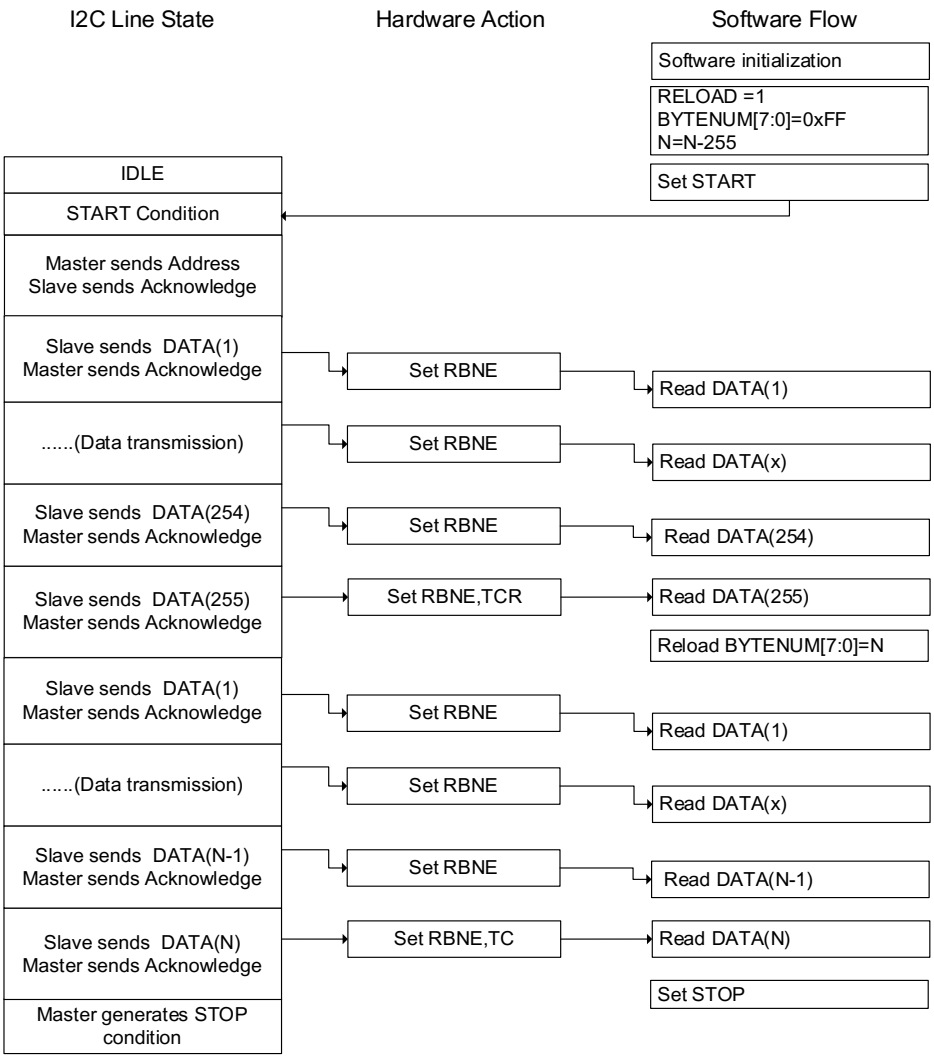


Figure 21-34. Programming model for master receiving (N>255)



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 transaction 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 uses I2C hardware and I2C hardware addressing, but adds 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 allow 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.

SMBus slave byte control

The slave byte control of SMBus receiver is the same as I2C. It allows the ACK control of each byte. The Slave Byte Control mode must be enabled by setting SBCTL bit in I2C_CTL0 register.

Host notify protocol

When the SMBHAEN bit in the I2C_CTL0 register is set, the SMBus supports the host notify protocol. In this protocol, the device acts as a master and the host as a slave, and the host will acknowledge the SMBus host address.

Time-out feature

SMBus has a time-out feature which resets devices if a communication takes too long. This explains the minimum clock frequency of 10 kHz to prevent locking up the bus. I2C can be a 'DC' bus, meaning that a slave device stretches the master clock when performing some routine while the master is accessing it. This will notify to the master that the slave is busy but does not want to lose the communication. The slave device will allow continuation after its task is completed. There is no limit in the I2C bus protocol as to how long this delay can be, whereas for a SMBus system, it would be limited to 25~35ms. 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 clear this mode. Slave devices are not allowed to hold the clock low too long.

The timeout detection can be enabled by setting TOEN and EXTOEN bits in the I2C_TIMEOUT register. The timer must be configured to guarantee that the timeout detected before the maximum time given in the SMBus specification.

The value programmed in BUSTOA[11:0] is used to check the $t_{TIMEOUT}$ parameter. To detect the SCL low level timeout, the TOIDLE bit must be 0. And the timer can be enabled by setting the TOEN bit in the I2C_TIMEOUT register, after the TOEN bit is set, the BUSTOA[11:0] and the TOIDLE bit cannot be changed. If the low level time of SCL is greater than (BUSTOA+1)*2048*t_{I2CCLK}, the TIMEOUT flag will be set in I2C_STAT register.

The BUSTOB[11:0] is used to check the $t_{LOW:SEXT}$ of the slave and the $t_{LOW:MEXT}$ of the master. The timer can be enabled by setting the EXTOEN bit in the I2C_TIMEOUT register, after the EXTOEN bit is set, the BUSTOB[11:0] cannot beS changed. If the SCL stretching time of the SMBus peripheral is greater than (BUSTOB+1)*2048*t_{I2CCLK} and within the timeout interval described in the bus idle detection section, the TIMEOUT bit in the I2C_STAT register will be set.

Packet error checking

There is a CRC-8 calculator in I2C block to perform Packet Error Checking for I2C data. A PEC (packet error code) byte is appended at the end of each transfer. The byte is calculated as CRC-8 checksum, calculated over the entire message including the address and read/write bit. The polynomial used is x^8+x^2+x+1 (the CRC-8-ATM HEC algorithm, initialized to zero).

When I2C is disabled, the PEC can be enabled by setting the PECEN bit in I2C_CTL0 register. Since the PEC transmission is managed by BYTENUM[7:0] in I2C_CTL1 register, SBCTL bit must be set when act as a slave. When PECTRANS is set and the RELOAD bit is cleared, PEC is transmitted after the BYTENUM[7:0]-1 data byte. The PECTRANS has no effect if RELOAD is set.

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. The host processes the interrupt and accesses all SMBALERT# devices through the Alert Response Address at the same time. If the SMBALERT# is pulled low by the devices, the devices will acknowledge the Alert Response Address. When SMBHAEN is 0, it is configured as a slave device, the SMBA pin will be pulled low by setting the SMBALTEN bit in the I2C_CTL0 register. Meanwhile the Alert Response Address is enabled. When SMBHAEN is 1, it is configured as a host, and the SMBALTEN is 1, as soon as a falling edge is detected on the SMBA pin, the SMBALT flag will be set in the I2C_STAT register. If the ERRIE bit is set in the I2C_CTL0 register, an interrupt will be generated. When SMBALTEN is 0, the level of ALERT line is considered high even if the SMBA pin is low. The SMBA pin can be used as a standard GPIO if SMBALTEN is 0.

Bus idle detection

If the master detects that the high level duration of the clock and data signals is greater than $t_{HIGH,MAX}$, the bus can be considered idle.

This timing parameter includes the case of a master that has been dynamically added to the bus and may not have detected a state transition on a SMBCLK or SMBDAT lines. In this case, in order to ensure that there is no ongoing transmission, the master must wait long enough.

The BUSTOA[11:0] bits must be programmed with the timer reload value to enable the t_{IDLE} check in order to obtain the t_{IDLE} parameter. To detect SCL and SDA high level timeouts, the TOIDLE bit must be set. Then setting the TOEN bit in the I2C_TIMEOUT register to enable

the timer, after the TOEN bit is set, the BUSTOA[11:0] bit and the TOIDLE bit cannot be changed. If the high level time of both SCL and SDA is greater than (BUSTOA+1)*4*t_{I2CCLK}, the TIMEOUT flag will be set in the I2C_STAT register.

SMBus slave mode

The SMBus receiver must be able to NACK each command or data it receives. For ACK control in slave mode, slave byte control mode can be enabled by setting SBCTL bit in I2C_CTL0 register.

SMBus-specific addresses should be enabled when needed. The SMBus Device Default address (0b1100 001) is enabled by setting the SMBDAEN bit in the I2C_CTL0 register. The SMBus Host address (0b0001 000) is enabled by setting the SMBHAEN bit in the I2C_CTL0 register. The Alert Response Address (0b0001 100) is enabled by setting the SMBALTEN bit in the I2C_CTL0 register.

SMBus mode

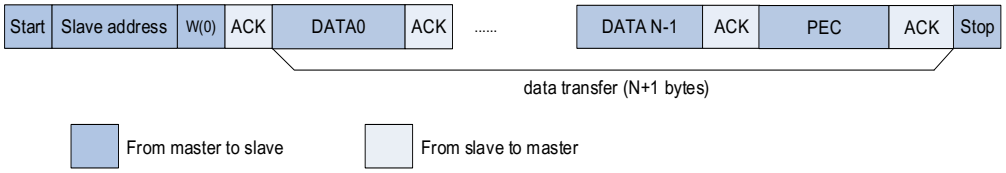
SMBus master transmitter and slave receiver

The PEC in SMBus master mode can be transmitted by setting the PECTRANS bit before setting the START bit, and the number of bytes in the BYTENUM[7:0] field must be configured. In this case, the total number of transmissions when TI interrupt occur is BYTENUM - 1. So if BYTENUM = 0x1 and PECTRANS bit is set, the data in I2C_PEC register will be transmitted automatically. If AUTOEND is 1 the SMBus master will send the STOP signal after the PEC byte automatically. If the AUTOEND is 0, the SMBus master can send a RESTART signal after the PEC. The PEC byte in I2C_PEC register will be sent after BYTENUM - 1 bytes, and the TC flag will be set after PEC is sent, then the SCL line is stretched low. The RESTART must be set in the TC interrupt routine.

When used as slave receiver, in order to allow PEC checking at the end of the number of bytes transmitted, SBCTL must be set. To configure ack control for each byte, the RELOAD must be set. In order to check the PEC byte, it is necessary to clear the RELOAD bit and set PECTRANS bit. After receiving BYTENUM-1 data, the next received byte will be compared with the data in the I2C_PEC register. If the PEC values does not match, the NACK is automatically generated. If the PEC values matches, the ACK is automatically generated, regardless of the NACKEN bit value. When PEC byte is received, it is also copied into the I2C_RDATA register, and RBNE flag will be set. If the ERRIE bit in I2C_CTL0 register is 1, when PEC value does not match, the PECERR flag will be set and the interrupt will be generated. If ACK control is not required, then PECTRANS can be set to 1 and BYTENUM can be programmed according to the number of bytes to be received.

Note: After the RELOAD bit is set, the PECTRANS cannot be changed.

Figure 21-35. SMBus master transmitter and slave receiver communication flow



SMBus master receiver and slave transmitter

If the SMBus master is required to receive PEC at the end of bytes transfer, automatic end mode can be enabled. Before sending a START signal on the bus, PECTRANS bit must be set and slave addresses must be programmed. After receiving BYTENUM - 1 data, the next received byte will be compared with the data in the I2C_PEC register automatically. A NACK is respond to the PEC byte before STOP signal.

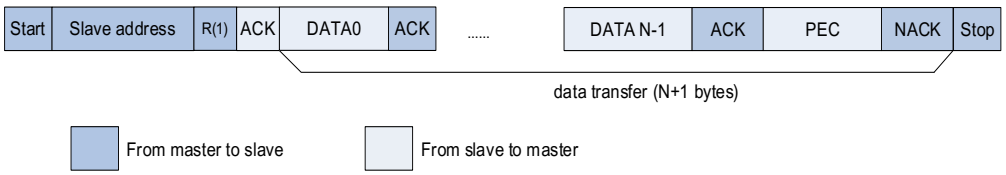
If the SMBus master receiver is required to generate a RESTART signal after receiving PEC byte, automatic end mode must be disabled. Before sending a START signal to the bus, PECTRANS bit must be set and slave addresses must be programmed. After receiving BYTENUM - 1 data, the next received byte will be compared with the data in the I2C_PEC register automatically. The TC flag will be set after PEC is sent, then the SCL line is stretched low. The RESTART can be set in the TC interrupt routine.

When used as slave transmitter, in order to allow PEC transmission at the end of BYTENUM[7:0] bytes, SBCTL must be set. If PECTRANS bit is set, the number of bytes in BYTENUM[7:0] contains PEC byte. In this case, if the number of bytes requested by the master is greater than BYTENUM - 1, the total number of TI interrupts will be BYTENUM - 1, and the data of the I2C_PEC register will be transmitted automatically.

Note:

1. After the RELOAD bit is set, the PECTRANS cannot be changed.
2. When SMBus acts as a slave transmitter and uses the PEC function, BYTENUM[7:0] must be configured before detecting the START signal.

Figure 21-36. SMBus master receiver and slave transmitter communication flow



Wakeup from power saving modes

When the address of I2C matches correctly, it can wake up MCU from Sleep mode and Deep-sleep mode. In order to wake up from these power saving modes, WUEN bit must be set in the I2C_CTL0 register and the IRC8M must be selected as the clock source for I2CCLK.

During Deep-sleep mode and Deep-sleep 1 mode, the IRC8M is switched off. The I2C interface switches the IRC8M on, and stretches SCL low until IRC8M is woken up when a START is detected. Then the IRC8M is used as the clock of I2C to receive the address. When address matching is detected, I2C stretches SCL during MCU wake-up. The SCL is released until the software clears the ADDSEND flag and the transmission proceeds normally. If the detected address does not match, IRC8M will be closed again and the MCU will not be wake up.

Only an address match interrupt (ADDMIE = 1) can wakeup the MCU. If the clock source of I2C is the system clock, or WUEN = 0, IRC8M will not switched on after receiving start signal. When wakeup from power saving mode is enabled, the digital filter must be disabled and the SS bit in I2C_CTL0 must be cleared. Before entering power saving mode, the I2C peripheral must be disabled (I2CEN = 0) if wakeup from power saving mode is disabled (WUEN = 0).

Use DMA for data transfer

As is shown in I2C slave mode and I2C master mode, each time TI or RBNE is asserted, software should write or read a byte, this may cause CPU's high overload. The DMA controller can be used to process TI and RBNE flag: each time TI or RBNE is asserted, DMA controller does a read or write operation automatically.

The DMA transmission request is enabled by setting the DENT bit in I2C_CTL0 register. The DMA reception request is enabled by setting the DENR bit in I2C_CTL0 register. In master mode, the slave address, transmission direction, number of bytes and START bit are programmed by software. The DMA must be initialized before setting the START bit. The number of bytes to be transferred is configured in the BYTENUM[7:0] in I2C_CTL1 register. In slave mode, the DMA must be initialized before the address match event or in the ADDSEND interrupt routine, before clearing the ADDSEND flag.

I2C error and interrupts

The I2C error flags are listed in *Table 21-7. I2C error flags*.

Table 21-7. I2C error flags

I2C Error Name	Description
BERR	Bus error
LOSTARB	Arbitration lost
OUERR	Overrun / Underrun flag
PECERR	CRC value doesn't match
TIMEOUT	Bus timeout in SMBus mode
SMBALT	SMBus Alert

The I2C interrupt events and flags are listed in *Table 21-8. I2C interrupt events*.

Table 21-8. I2C interrupt events

Interrupt event	Event flag	Enable control bit
I2C_RDATA is not empty during receiving	RBNE	RBNEIE
Transmit interrupt	TI	TIE
STOP signal detected in slave mode	STPDET	STPDETIE
Transfer complete reload	TCR	TCIE
Transfer complete	TC	
Address match	ADDSEND	ADDMIE
Not acknowledge received	NACK	NACKIE
Bus error	BERR	ERRIE
Arbitration Lost	LOSTARB	
Overrun / Underrun error	OUERR	
PEC error	PECERR	
Timeout error	TIMEOUT	
SMBus Alert	SMBALT	

I2C debug mode

When the microcontroller enters the debug mode (Cortex®-M23 core halted), the SMBus timeout either continues to work normally or stops, depending on the I2C2_HOLD configuration bit in the DBG module.

21.2.4. Register definition

I2C2 base address: 0x4000 C000

Control register 0 (I2C_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								PECEN	SMBALTEN	SMBDAEN	SMBHAEN	GCEN	WUEN	SS	SBCTL
								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DENR	DENT	Reserved	ANOFF	DNF[3:0]				ERRIE	TCIE	STPDETIE	NACKIE	ADDMIE	RBNEIE	TIE	I2CEN
rw	rw		rw	rw				rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23	PECEN	PEC Calculation Switch 0: PEC Calculation off 1: PEC Calculation on **Note:** If the SMBus feature is not supported, this bit is reserved and forced by hardware to '0'.
22	SMBALTEN	SMBus Alert enable 0: SMBA pin is not pulled down (device mode) or SMBus Alert pin SMBA is disabled (host mode) 1: SMBA pin is pulled down (device mode) or SMBus Alert pin SMBA is enabled (host mode) **Note:** If the SMBus feature is not supported, this bit is reserved and forced by hardware to '0'.
21	SMBDAEN	SMBus device default address enable 0: Device default address is disabled, the default address 0b1100001x will be not acknowledged. 1: Device default address is enabled, the default address 0b1100001x will be acknowledged. **Note:** If the SMBus feature is not supported, this bit is reserved and forced by hardware to '0'.
20	SMBHAEN	SMBus host address enable 0: Host address is disabled, address 0b0001000x will be not acknowledged.

		1: Host address is enabled, address 0b0001000x will be acknowledged. **Note:** If the SMBus feature is not supported, this bit is reserved and forced by hardware to '0'.
19	GCEN	Whether or not to response to a General Call (0x00) 0: Slave won't response to a General Call 1: Slave will response to a General Call
18	WUEN	Wakeup from power saving mode enable, including Deep-sleep mode and Deep-sleep 1 mode. This bit is cleared when mcu wakeup from power saving mode. 0: Wakeup from power saving mode disable. 1: Wakeup from power saving mode enable. **Note:** WUEN can be set only when DNF[3:0] = 0000.
17	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 **Note:** When in master mode, this bit must be 0. This bit can be modified when I2CEN = 0.
16	SBCTL	Slave byte control This bit is used to enable hardware byte control in slave mode. 0: Slave byte control is disabled 1: Slave byte control is enabled
15	DENR	DMA enable for reception 0: DMA is disabled for reception 1: DMA is enabled for reception
14	DENT	DMA enable for transmission 0: DMA is disabled for transmission 1: DMA is enabled for transmission
13	Reserved	Must be kept at reset value.
12	ANOFF	Analog noise filter disable 0: Analog noise filter is enabled 1: Analog noise filter is disabled **Note:** This bit can only be programmed when the I2C is disabled (I2CEN = 0).
11:8	DNF[3:0]	Digital noise filter 0000: Digital filter is disabled 0001: Digital filter is enabled and filter spikes with a length of up to 1 t_{I2CCLK} ... 1111: Digital filter is enabled and filter spikes with a length of up to15 t_{I2CCLK}

Note: These bits can only be modified when the I2C is disabled (I2CEN = 0).

7	ERRIE	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled. When BERR, LOSTARB, OUERR, PECERR, TIMEOUT or SMBALT bit is set, an interrupt will be generated.
6	TCIE	Transfer complete interrupt enable 0: Transfer complete interrupt is disabled 1: Transfer complete interrupt is enabled
5	STPDETIE	Stop detection interrupt enable 0: Stop detection (STPDET) interrupt is disabled 1: Stop detection (STPDET) interrupt is enabled
4	NACKIE	NACK received interrupt enable 0: NACK received interrupt is disabled 1: NACK received interrupt is enabled
3	ADDMIE	Address match interrupt enable in slave mode 0: Address matchinterrupt is disabled 1: Address matchnterrupt is enabled
2	RBNEIE	Receive interrupt enable 0: Receive (RBNE) interrupt is disabled 1: Receive (RBNE) interrupt is enabled
1	TIE	Transmit interrupt enable 0: Transmit (TI) interrupt is disabled 1: Transmit (TI) interrupt is enabled
0	I2CEN	I2C peripheral enable 0: I2C is disabled 1: I2C is enabled

Control register 1 (I2C_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					PECTRANS	AUTOEND	RELOAD	BYTENUM[7:0]							
					rw	rw	rw	rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NACKEN	STOP	START	HEAD10R	ADD10EN	TRDIR	SADDRESS[9:0]									

rw rw rw rw rw rw rw

Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value.
26	PECTRANS	PEC Transfer Set by software. Cleared by hardware in the following cases: When PEC byte is transferred or ADDSEND bit is set or STOP signal is detected or I2CEN = 0. 0: Don't transfer PEC value 1: Transfer PEC **Note:** This bit has no effect when RELOAD = 1, or SBCTL = 0 in slave mode. If the SMBus feature is not supported, this bit is reserved and forced by hardware to '0'.
25	AUTOEND	Automatic end mode in master mode 0: TC bit is set when the transfer of BYTENUM[7:0] bytes is completed. 1: a STOP signal is sent automatically when the transfer of BYTENUM[7:0] bytes is completed. **Note:** This bit works only when RELOAD = 0. This bit is set and cleared by software.
24	RELOAD	Reload mode 0: After the data of BYTENUM[7:0] bytes transfer, the transfer is completed. 1: After data of BYTENUM[7:0] bytes transfer, the transfer is not completed and the new BYTENUM[7:0] will be reloaded. Every time when the BYTENUM[7:0] bytes have been transferred, the TCR bit in I2C_STAT register will be set. **Note:** This bit is set and cleared by software.
23:16	BYTENUM[7:0]	Number of bytes to be transferred These bits are programmed with the number of bytes to be transferred. When SBCTL = 0, these bits have no effect. **Note:** When the START bit is set to 1, this bit cannot be modified, and the software cannot change this bit during transmission.
15	NACKEN	Generate NACK in slave mode 0: an ACK is sent after receiving a new byte. 1: a NACK is sent after receiving a new byte. **Note:** The bit can be set by software, and cleared by hardware when the NACK is sent, or when a STOP signal is detected or ADDSEND is set, or when I2CEN = 0. When PEC is enabled, whether to send an ACK or a NACK is not depend on the NACKEN bit. When SS = 1, and the OUERR bit is set, the value of NACKEN is ignored and a NACK will be sent.
14	STOP	Generate a STOP signal on I2C bus This bit is set by software and cleared by hardware when I2CEN = 0 or STOP signal is detected.

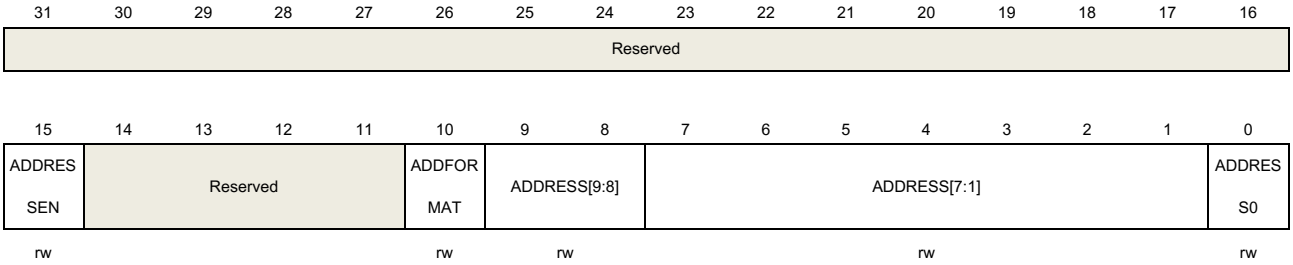
		0: STOP will not be sent 1: STOP will be sent
13	START	Generate a START signal on I2C bus This bit is set by software and cleared by hardware after the address is sent. When the arbitration is lost, or a timeout error occurred, or I2CEN = 0, this bit can also be cleared by hardware. It can be cleared by software by setting the ADDSENDC bit in I2C_STATC register. 0: START will not be sent 1: START will be sent
12	HEAD10R	10-bit address header executes read direction only in master receive mode 0: The 10 bit master receive address sequence is START + header of 10-bit address (write) + slave address byte 2 + RESTART + header of 10-bit address (read). 1: The 10 bit master receive address sequence is RESTART + header of 10-bit address (read). **Note:** When the START bit is set to 1, this bit cannot be modified, and the software cannot change this bit during transmission.
11	ADD10EN	10-bit addressing mode enable in master mode 0: 7-bit addressing in master mode 1: 10-bit addressing in master mode **Note:** When the START bit is set to 1, this bit cannot be modified, and the software cannot change this bit during transmission.
10	TRDIR	Transfer direction in master mode 0: Master transmit 1: Master receive **Note:** When the START bit is set to 1, this bit cannot be modified, and the software cannot change this bit during transmission.
9:0	SADDRESS[9:0]	Slave address to be sent SADDRESS[9:8]: Slave address bit 9:8 If ADD10EN = 0, these bits have no effect. If ADD10EN = 1, these bits should be written with bits 9:8 of the slave address to be sent. SADDRESS[7:1]: Slave address bit 7:1 If ADD10EN = 0, these bits should be written with the 7-bit slave address to be sent. If ADD10EN = 1, these bits should be written with bits 7:1 of the slave address to be sent. SADDRESS0: Slave address bit 0 If ADD10EN = 0, this bit has no effect. If ADD10EN = 1, this bit should be written with bit 0 of the slave address to be sent **Note:** When the START bit is set to 1, this bit cannot be modified, and the software cannot change this bit during transmission.

Slave address register 0 (I2C_SADDR0)

Address offset: 0x08

Reset value: 0x0000 0000

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



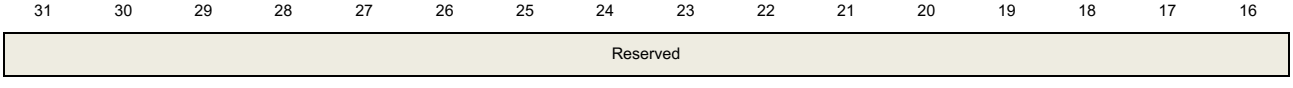
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	ADDRESSEN	I2C address enable 0: I2C address disable. 1: I2C address enable.
14:11	Reserved	Must be kept at reset value.
10	ADDFORMAT	Address mode for the I2C slave 0: 7-bit address 1: 10-bit address **Note:** When ADDRESSEN is set, this bit should not be written.
9:8	ADDRESS[9:8]	Highest two bits of a 10-bit address **Note:** When ADDRESSEN is set, these bits should not be written.
7:1	ADDRESS[7:1]	7-bit address or bits 7:1 of a 10-bit address **Note:** 1. When ADDRESSEN is set, these bits should not be written. 2. When the slave address is set to the reserved address 0b'1111000, the slave will not respond if an address match occurs.
0	ADDRESS0	Bit 0 of a 10-bit address **Note:** When ADDRESSEN is set, this bit should not be written.

Slave address register 1 (I2C_SADDR1)

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
ADDRESS2EN	Reserved				ADDMSK2[2:0]			ADDRESS2[7:1]							Reserved
rw					rw			rw							

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	ADDRESS2EN	Second I2C address enable 0: Second I2C address disable. 1: Second I2C address enable.
14:11	Reserved	Must be kept at reset value.
10:8	ADDMSK2[2:0]	ADDRESS2[7:1] mask Defines which bits of ADDRESS2[7:1] are compared with an incoming address byte, and which bits are masked (don't care). 000: No mask, all the bits must be compared. n(001~110): ADDRESS2[n:1] is masked. Only ADDRESS2[7:n+1] are compared. 111: ADDRESS2[7:1] are masked. All 7-bit received addresses are acknowledged except the reserved address (0b0000xxx and 0b1111xxx). **Note:** When ADDRESS2EN is set, these bits should not be written. If ADDMSK2 is not equal to 0, the reserved I2C addresses (0b0000xxx and 0b1111xxx) are not acknowledged even if all the bits are matched.
7:1	ADDRESS2[7:1]	Second I2C address for the slave **Note:** 1. When ADDRESS2EN sis set, these bits should not be written. 2. When the slave address is set to the reserved address 0b'1111000, the slave will not respond if an address match occurs.
0	Reserved	Must be kept at reset value.

Timing register (I2C_TIMING)

Address offset: 0x10

Reset value: 0x0000 0000

This register cannot be modified when the I2C is enabled (I2CEN = 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
PSC[3:0]				Reserved				SCLDELY[3:0]				SDADELY[3:0]			
rw								rw				rw			

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

rw rw

Bits	Fields	Descriptions
31:28	PSC[3:0]	Timing prescaler In order to generate the clock period t_{PSC} used for data setup and data hold counters, these bits are used to configure the prescaler for I2CCLK. The t_{PSC} is also used for SCL high and low level counters. $t_{PSC}=(PSC+1)*t_{I2CCLK}$
27:24	Reserved	Must be kept at reset value.
23:20	SCLDELY[3:0]	Data setup time A delay $t_{SCLDELY}$ between SDA edge and SCL rising edge can be generated by configuring these bits. And during $t_{SCLDELY}$, the SCL line is stretched low in master mode and in slave mode when SS = 0. $t_{SCLDELY}=(SCLDELY+1)*t_{PSC}$
19:16	SDADELY[3:0]	Data hold time A delay $t_{SDADELY}$ between SCL falling edge and SDA edge can be generated by configuring these bits. And during $t_{SDADELY}$, the SCL line is stretched low in master mode and in slave mode when SS = 0. 0x0: $t_{SDADELY}$= 1*t_{PSC} 0x1-0xF: $t_{SDADELY}$= SDADELY*t_{PSC}
15:8	SCLH[7:0]	SCL high period SCL high period can be generated by configuring these bits. $t_{SCLH}=(SCLH+1)*t_{PSC}$ **Note:** These bits can only be used in master mode.
7:0	SCLL[7:0]	SCL low period SCL low period can be generated by configuring these bits. $t_{SCLL}=(SCLL+1)*t_{PSC}$ **Note:** These bits can only be used in master mode.

Timeout register (I2C_TIMEOUT)

Address offset: 0x14
Reset value: 0x0000 0000

Note: If the SMBus feature is not supported, this register is reserved and forced by hardware to "0x00000000".
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
EXTOEN	Reserved			BUSTOB[11:0]											
rw				rw											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

TOEN	Reserved	TOIDLE	BUSTOA[11:0]
rw		rw	rw

Bits	Fields	Descriptions
31	EXTOEN	Extended clock timeout detection enable When a cumulative SCL stretch time is greater than $t_{LOW:EXT}$, a timeout error will be occurred. $t_{LOW:EXT}$=(BUSTOB+1)*2048*t_{I2CCLK}. 0: Extended clock timeout detection is disabled. 1: Extended clock timeout detection is enabled.
30:28	Reserved	Must be kept at reset value.
27:16	BUSTOB[11:0]	Bus timeout B Configure the cumulative clock extension timeout. In master mode, the master cumulative clock low extend time $t_{LOW:MEXT}$ is detected. In slave mode, the slave cumulative clock low extend time $t_{LOW:SEXT}$ is detected. $t_{LOW:EXT}$=(BUSTOB+1)*2048*t_{I2CCLK}. **Note:** These bits can be modified only when EXTOEN = 0.
15	TOEN	Clock timeout detection enable If the SCL stretch time greater than $t_{TIMEOUT}$ when TOIDLE = 0 or high for more than t_{IDLE} when TOIDLE = 1, a timeout error is detected. 0: SCL timeout detection is disabled 1: SCL timeout detection is enabled
14:13	Reserved	Must be kept at reset value.
12	TOIDLE	Idle clock timeout detection 0: BUSTOA is used to detect SCL low timeout 1: BUSTOA is used to detect both SCL and SDA high timeout when the bus is idle **Note:** This bit can be written only when TOEN = 0.
11:0	BUSTOA[11:0]	Bus timeout A When TOIDLE = 0, $t_{TIMEOUT}$=(BUSTOA+1)*2048*t_{I2CCLK}. When TOIDLE = 1, t_{IDLE}=(BUSTOA+1)*4*t_{I2CCLK}. **Note:** These bits can be written only when TOEN = 0.

Status register (I2C_STAT)

Address offset: 0x18

Reset value: 0x0000 0001

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

I2CBSY	Reserved	SMBALT	TIMEOUT	PECERR	OUERR	LOSTARB	BERR	TCR	TC	STPDET	NACK	ADDSEND	RBNE	TI	TBE
r		r	r	r	r	r	r	r	r	r	r	r	r	rw	rw

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:17	READDR[6:0]	Received match address in slave mode When the ADDSEND bit is set, these bits store the matched address. In the case of a 10-bit address, READDR[6:0] stores the header of the 10-bit address followed by the 2 MSBs of the address.
16	TR	Whether the I2C is a transmitter or a receiver in slave mode This bit is updated when the ADDSEND bit is set. 0: Receiver 1: Transmitter
15	I2CBSY	Busy flag This bit is set by hardware when a START signal is detected and cleared by hardware after a STOP signal. When I2CEN = 0, this bit is also cleared by hardware. 0: No I2C communication. 1: I2C communication active.
14	Reserved	Must be kept at reset value.
13	SMBALT	SMBus Alert When SMBHAEN = 1, SMBALTEN = 1, and a SMBALERT event (falling edge) is detected on SMBA pin, this bit will be set by hardware. It is cleared by software by setting the SMBALTC bit. This bit is cleared by hardware when I2CEN = 0. 0: SMBALERT event is not detected on SMBA pin 1: SMBALERT event is detected on SMBA pin **Note:** If the SMBus feature is not supported, this bit is reserved and forced by hardware to '0'.
12	TIMEOUT	TIMEOUT flag. When a timeout or extended clock timeout occurred, this bit will be set. It is cleared by software by setting the TIMEOUTC bit and cleared by hardware when I2CEN = 0. 0: no timeout or extended clock timeout occur 1: a timeout or extended clock timeout occur **Note:** If the SMBus feature is not supported, this bit is reserved and forced by hardware to '0'.
11	PECERR	PEC error This flag is set by hardware when the received PEC does not match with the content of I2C_PEC register. Then a NACK is automatically sent. It is cleared by software by setting the PECERRC bit and cleared by hardware when I2CEN = 0.

		0: Received PEC and content of I2C_PEC match 1: Received PEC and content of I2C_PEC don't match, I2C will send NACK regardless of NACKEN bit. **Note:** If the SMBus feature is not supported, this bit is reserved and forced by hardware to '0'.
10	OUERR	Overrun/Underrun error in slave mode In slave mode with SS = 1, when an overrun/underrun error occurs, this bit will be set by hardware. It is cleared by software by setting the OUERRC bit and cleared by hardware when I2CEN = 0. 0: No overrun or underrun occurs 1: Overrun or underrun occurs
9	LOSTARB	Arbitration Lost It is cleared by software by setting the LOSTARBC bit and cleared by hardware when I2CEN = 0. 0: No arbitration lost. 1: Arbitration lost occurs and the I2C block changes back to slave mode.
8	BERR	Bus error When an unexpected START or STOP signal on I2C bus is detected, a bus error occurs and this bit will be set. It is cleared by software by setting BERRC bit and cleared by hardware when I2CEN = 0. 0: No bus error 1: A bus error detected
7	TCR	Transfer complete reload This bit is set by hardware when RELOAD = 1 and data of BYTENUM[7:0] bytes have been transferred. It is cleared by software when BYTENUM[7:0] is written to a non-zero value. 0: When RELOAD = 1, transfer of BYTENUM[7:0] bytes is not completed 1: When RELOAD = 1, transfer of BYTENUM[7:0] bytes is completed
6	TC	Transfer complete in master mode This bit is set by hardware when RELOAD = 0, AUTOEND = 0 and data of BYTENUM[7:0] bytes have been transferred. It is cleared by software when START bit or STOP bit is set. 0: Transfer of BYTENUM[7:0] bytes is not completed 1: Transfer of BYTENUM[7:0] bytes is completed
5	STPDET	STOP signal detected in slave mode This flag is set by hardware when a STOP signal is detected on the bus. It is cleared by software by setting STPDETC bit and cleared by hardware when I2CEN = 0. 0: STOP signal is not detected. 1: STOP signal is detected.
4	NACK	Not Acknowledge flag

This flag is set by hardware when a NACK is received. It is cleared by software by setting NACKC bit and cleared by hardware when I2CEN = 0.
0: ACK is received.
1: NACK is received.

3 ADDSEND
Address received matches in slave mode.
This bit is set by hardware when the received slave address matched with one of the enabled slave addresses. It is cleared by software by setting ADDSENDC bit and cleared by hardware when I2CEN = 0.
0: Received address not matched
1: Received address matched

2 RBNE
I2C_RDATA is not empty during receiving
This bit is set by hardware when the received data is shift into the I2C_RDATA register. It is cleared when I2C_RDATA is read.
0: I2C_RDATA is empty
1: I2C_RDATA is not empty, software can read

1 TI
Transmit interrupt
This bit is set by hardware when the I2C_TDATA register is empty and the I2C is ready to transmit data. It is cleared when the next data to be sent is written in the I2C_TDATA register.When SS = 1, this bit can be set by software, in order to generate a TI event (interrupt if TIE = 1 or DMA request if DENT = 1).
0: I2C_TDATA is not empty or the I2C is not ready to transmit data
1: I2C_TDATA is empty and the I2C is ready to transmit data

0 TBE
I2C_TDATA is empty during transmitting
This bit is set by hardware when the I2C_TDATA register is empty. It is cleared when the next data to be sent is written in the I2C_TDATA register. This bit can be set by software in order to empty the I2C_TDATA register.
0: I2C_TDATA is not empty
1: I2C_TDATA is empty

Status clear register (I2C_STATC)

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
Reserved		SMBALTC	TIMEOUTC	PECERRC	OUERRC	LOSTARBC	BERRC	Reserved		STPDETC	NACKC	ADDSENDC	Reserved		
		w	w	w	w	w	w			w	w	w			

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13	SMBALTC	SMBus alert flag clear. Software can clear the SMBALT bit of I2C_STAT by writing 1 to this bit. **Note:** If the SMBus feature is not supported, this bit is reserved and forced by hardware to '0'.
12	TIMEOUTC	TIMEOUT flag clear. Software can clear the TIMEOUT bit of I2C_STAT by writing 1 to this bit. **Note:** If the SMBus feature is not supported, this bit is reserved and forced by hardware to '0'.
11	PECERRC	PEC error flag clear. Software can clear the PECERR bit of I2C_STAT by writing 1 to this bit. **Note:** If the SMBus feature is not supported, this bit is reserved and forced by hardware to '0'.
10	OUERRC	Overrun/Underrun flag clear. Software can clear the OUERR bit of I2C_STAT by writing 1 to this bit.
9	LOSTARBC	Arbitration Lost flag clear. Software can clear the LOSTARB bit of I2C_STAT by writing 1 to this bit.
8	BERRC	Bus error flag clear. Software can clear the BERR bit of I2C_STAT by writing 1 to this bit.
7:6	Reservced	Must be kept at reset value.
5	STPDETC	STPDET flag clear Software can clear the STPDET bit of I2C_STAT by writing 1 to this bit.
4	NACKC	Not Acknowledge flag clear Software can clear the NACK bit of I2C_STAT by writing 1 to this bit.
3	ADDSENDC	ADDSEND flag clear Software can clear the ADDSEND bit of I2C_STAT by writing 1 to this bit.
2:0	Reserved	Must be kept at reset value.

PEC register (I2C_PEC)

Address offset: 0x20

Reset value: 0x0000 0000

Note: If the SMBus feature is not supported, this register is reserved and forced by hardware to "0x00000000".

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								PECV[7:0]							
								r							

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	PECV[7:0]	Packet Error Checking Value that calculated by hardware when PEC is enabled. PECV is cleared by hardware when I2CEN = 0.

Receive data register (I2C_RDATA)

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
Reserved								RDATA[7:0]							
								r							

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	RDATA[7:0]	Receive data value

Transmit data register (I2C_TDATA)

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
Reserved								TDATA [7:0]							
								rw							

Bits	Fields	Descriptions

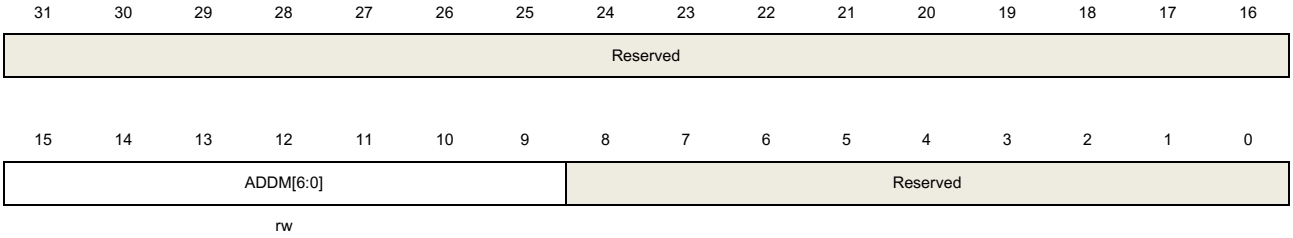
31:8	Reserved	Must be kept at reset value.
7:0	TDATA[7:0]	Transmit data value

Control register 2 (I2C_CTL2)

Address offset: 0x90

Reset value: 0x0000 FE00

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



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:9	ADDM[6:0]	Defines which bits of ADDRESS[7:1] are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in ADDM[6:0] enables comparisons with the corresponding bit in ADDRESS[7:1]. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address).
8:0	Reserved	Must be kept at reset value.

22. Appendix

22.1. List of abbreviations used in register

Table 22-1. List of abbreviations used in register

abbreviations for registers	Descriptions
read-only (r)	Software can only read this bit.
write-only (w)	Software can only write to this bit.
read/write (rw)	Software can read and write to this bit.
read/write-once (rwo)	Software can only write once to this bit and can also read it at any time. Only a reset can return the bit to its reset value.
read/write-1-to-clear (rc_w1)	Software can read as well as clear this bit by writing 1. Writing 0 has no effect on the bit value.
read/write-0-to-clear (rc_w0)	Software can read as well as clear this bit by writing 0. Writing 1 has no effect on the bit value.
read/write-to-clear (rc_w)	Software can read as well as clear this bit by writing. Writing 0 or 1 has the same effect to this bit.
write-to-clear (c_w)	Software can only clear this bit by writing. Writing 0 or 1 has the same effect to this bit.
write-1-to-clear (c_w1)	Software can only clear this bit by writing 1. Writing 0 has no effect on the bit value.
readable/read-to-clear (rc_r)	Software can read this bit. Reading this bit automatically clears it to '0'. Writing '0' has no effect on the bit value.
readable/settable (rs)	Software can read as well as set this bit to 1. Writing 0 has no effect on the bit value.
readable/read-to-set (rs_r)	Software can read this bit, and set this bit by reading. Writing has no effect on the bit value.
writable/read-to-set (ws_r)	Software can write this bit, and set this bit by reading.
readable/write-to-set (rs_w)	Software can read this bit, and set this bit by writing. Writing 0 or 1 has the same effect to this bit.
write-to-set (s_w)	Software can only set this bit by writing. Writing 0 or 1 has the same effect to this bit.
readable/write-0-to-set (rs_w0)	Software can read this bit, and set this bit by writing 0. Writing 1 has no effect on the bit value.
read-only/write-to-trigger (rt_w)	Software can read this bit. Writing 0 or 1 triggers an event but has no effect on the bit value.
read-only/write-1-to-trigger (rt_w1)	Software can only read to this bit. Writing 1 triggers the event but has no effect on the bit value.
read-only/write-0-to-	Software can only read to this bit. Writing 0 triggers the event but has no

abbreviations for registers	Descriptions
trigger (rt_w0)	effect on the bit value.
toggle (t)	Software can toggle this bit by writing 1. Writing 0 has no effect.

22.2. List of terms

Table 22-2. 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.

22.3. Available peripherals

For availability of peripherals and their number across all MCU series types, refer to the corresponding device data datasheet.

23. Revision history

Table 23-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Jun.1, 2026
1.1	1. Change "PA0-PA15, PB0-PB10, PF0, PF1 and PF3" to "PA0-PA15, PB0-PB10, PF0 and PF3" in chapter ***8.3.12. Priority Crossbar decoder (only for GD32E251xx)***. 2. Delete LDOVS related description in chapter ***5.3.3 Power saving modes***. 3. Delete LDOVS bit field in chapter ***5.4.1. Control register (PMU_CTL)***. 4. Add note in chapter ***1.4. Boot configuration***.	Aug.10, 2026

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