GigaDevice Semiconductor Inc.

GD32VW55x
RISC-V 32-bit MCU

For GD32VW553xx

User Manual

Revision 1.3

(Feb. 2025)

Table of Contents

Table of Contents **2**

List of Figures **16**

List of Tables **21**

1. System and memory architecture **23**

1.1. RISC-V processor **23**

1.2. System architecture **24**

1.3. Memory map **25**

1.3.1. On-chip SRAM memory 29

1.3.2. On-chip flash memory overview 29

1.4. Boot configuration **29**

1.5. System configuration registers (SYSCFG) **32**

1.5.1. Configuration register 0 (SYSCFG_CFG0) 32

1.5.2. EXTI sources selection register 0 (SYSCFG_EXTISS0) 32

1.5.3. EXTI sources selection register 1 (SYSCFG_EXTISS1) 33

1.5.4. EXTI sources selection register 2 (SYSCFG_EXTISS2) 34

1.5.5. EXTI sources selection register 3 (SYSCFG_EXTISS3) 34

1.5.6. I/O compensation control register (SYSCFG_CPSCTL) 35

1.5.7. SYSCFG configuration register 1 (SYSCFG_CFG1) 36

1.5.8. SYSCFG shared SRAM configuration register (SYSCFG_SCFG) 36

1.5.9. TIMER trigger selection register (SYSCFG_TIMERxCFG)(x = 0..2) 37

2. Flash memory controller (FMC) **40**

2.1. Overview **40**

2.2. Characteristics **40**

2.3. Function overview **40**

2.3.1. Flash memory architecture 40

2.3.2. Read operations 41

2.3.3. Unlock the FMC_CTL register 41

2.3.4. Page erase 42

2.3.5. Mass erase 43

2.3.6. Main flash programming 44

2.3.7. Option bytes 46

2.3.8. Security protection 47

2.3.9. Wirte protection 48

2.3.10. FLASH interrupts 49

2.4. Register definition **50**

2.4.1. Unlock key register (FMC_KEY) 50
2.4.2. Option byte unlock key register (FMC_OBKEY) 50
2.4.3. Status register (FMC_STAT) 50
2.4.4. Control register (FMC_CTL) 51
2.4.5. Address register (FMC_ADDR) 53
2.4.6. Option byte status register (FMC_OBSTAT) 53
2.4.7. Option byte register (FMC_OBR) 54
2.4.8. Option byte user value register (FMC_OBUSER) 55
2.4.9. Option byte write protection area register 0 (FMC_OBWRP0) 55
2.4.10. Option byte write protection area register 1 (FMC_OBWRP1) 56
2.4.11. NO RTDEC region register x (FMC_NODECx)(x = 0…3) 56
2.4.12. Offset region register (FMC_OFRG) 57
2.4.13. Offset value register (FMC_OFVR) 57
2.4.14. Product ID0 register (FMC_PID0) 58
2.4.15. Product ID1 register (FMC_PID1) 58
2.4.16. RF Trim register 0 (FMC_RFT0) 58
2.4.17. RF Trim register 1 (FMC_RFT1) 59
2.4.18. WIFI Trim register x (FMC_WFTx)(x = 0…15) 59

3. Electronic fuse (EFUSE) 61

3.1. Overview 61

3.2. Characteristics 61

3.3. Function overview 61

3.3.1. Block diagram 61
3.3.2. Efuse architecture 62
3.3.3. Efuse macro description 62
3.3.4. Read operation 64
3.3.5. Program operation 64

3.4. Register definition 65

3.4.1. Control and status register (EFUSE_CS) 65
3.4.2. Address register (EFUSE_ADDR) 66
3.4.3. Control register 0 (EFUSE_CTL0) 67
3.4.4. Control register 1 (EFUSE_CTL1) 67
3.4.5. Flash protection control register (EFUSE_FPCTL) 68
3.4.6. User byte control register (EFUSE_USERCTL) 69
3.4.7. EFUSE reserved register x (EFUSE_RESx) (x = 0…2) 70
3.4.8. Firmware AES key register x (EFUSE_AESKEYx) (x = 0…3) 70
3.4.9. RoTPK key register x (EFUSE_ROTPKKEYx) (x = 0…7) 70
3.4.10. Product UID register x (EFUSE_PUIDx) (x = 0…3) 71
3.4.11. HUK key register x (EFUSE_HUKKEYx) (x = 0…3) 71
3.4.12. User data register x (EFUSE_USER_DATAx) (x = 0…7) 71
3.4.13. Boot address register (EFUSE_BOOTADDR) 72

4. Power management unit (PMU) 73
4.1. Overview 73
4.2. Characteristics 73
4.3. Function overview 74
4.3.1. Backup domain 74
4.3.2. V_{DD} / V_{DDA} power domain 75
4.3.3. 1.1V power domain 77
4.3.4. Power saving modes 77
4.4. Register definition 83
4.4.1. Control register 0 (PMU_CTL0) 83
4.4.2. Control and status register 0 (PMU_CS0) 84
4.4.3. Control register 1 (PMU_CTL1) 86
4.4.4. Control and status register 1 (PMU_CS1) 88
4.4.5. Parameter register 0 (PMU_PAR0) 89
4.4.6. Parameter register 1 (PMU_PAR1) 90
4.4.7. Parameter register 2 (PMU_PAR2) 90
4.4.8. RF Control register (PMU_RFCTL) 91
4.4.9. RF timer parameter register (PMU_RFPAR) 92
4.4.10. PMU interrupt flag register(PMU_INTF) 93
4.4.11. PMU interrupt enable register(PMU_INTEN) 93
4.4.12. PMU interrupt clear register(PMU_INTC) 94
5. Reset and clock unit (RCU) 95
5.1. Reset control unit (RCTL) 95
5.1.1. Overview 95
5.1.2. Function overview 95
5.2. Clock control unit (CCTL) 96
5.2.1. Overview 96
5.2.2. Characteristics 98
5.2.3. Function overview 98
5.3. Register definition 103
5.3.1. Control register (RCU_CTL) 103
5.3.2. PLL register (RCU_PLL) 105
5.3.3. Clock configuration register 0 (RCU_CFG0) 105
5.3.4. Clock interrupt register (RCU_INT) 108
5.3.5. AHB1 reset register (RCU_AHB1RST) 110
5.3.6. AHB2 reset register (RCU_AHB2RST) 111
5.3.7. AHB3 reset register (RCU_AHB3RST) 112
5.3.8. APB1 reset register (RCU_APB1RST) 113
5.3.9. APB2 reset register (RCU_APB2RST) 114
5.3.10. AHB1 enable register (RCU_AHB1EN) 116
5.3.11. AHB2 enable register (RCU_AHB2EN) 118

5.3.12. AHB3 enable register (RCU_AHB3EN) 118
5.3.13. APB1 enable register (RCU_APB1EN) 119
5.3.14. APB2 enable register (RCU_APB2EN) 121
5.3.15. AHB1 sleep mode enable register (RCU_AHB1SPEN) 122
5.3.16. Backup domain control register (RCU_BDCTL) 123
5.3.17. Reset source / clock register (RCU_RSTSCK) 124
5.3.18. PLLDIG clock configuration register 0 (RCU_PLLDIGCFG0) 126
5.3.19. Clock configuration register 1 (RCU_CFG1) 126
5.3.20. Additional clock control register (RCU_ADDCTL) 128
5.3.21. PLLDIG clock configuration register 1 (RCU_PLLDIGCFG1) 129
5.3.22. Voltage key register (RCU_VKEY) 129
5.3.23. Deep-sleep mode voltage register (RCU_DSV) 130

6. Interrupt / event controller (EXTI) 131

6.1. Overview 131

6.2. Characteristics 131

6.3. Function overview 131

6.4. External interrupt and event block diagram 134

6.5. External interrupt and event function overview 134

6.6. Register definition 137

6.6.1. Interrupt enable register (EXTI_INTEN) 137
6.6.2. Event enable register (EXTI_EVEN) 137
6.6.3. Rising edge trigger enable register (EXTI_RTEN) 138
6.6.4. Falling edge trigger enable register (EXTI_FTEN) 138
6.6.5. Software interrupt event register (EXTI_SWIEV) 139
6.6.6. Pending register (EXTI_PD) 140

7. General-purpose and alternate-function I/Os (GPIO and AFIO) 141

7.1. Overview 141

7.2. Characteristics 141

7.3. Function overview 141

7.3.1. GPIO pin configuration 143
7.3.2. External interrupt / event lines 143
7.3.3. Alternate functions (AF) 143
7.3.4. Additional functions 143
7.3.5. Input configuration 144
7.3.6. Output configuration 144
7.3.7. Analog configuration 145
7.3.8. Alternate function (AF) configuration 145
7.3.9. GPIO locking function 146
7.3.10. GPIO I/O compensation cell 146
7.3.11. GPIO single cycle toggle function 146

7.4. Register definition 147
7.4.1. Port control register (GPIOx_CTL, x = A…C) 147
7.4.2. Port output mode register (GPIOx_OMODE, x = A...C) 148
7.4.3. Port output speed register (GPIOx_OSPD, x = A...C) 150
7.4.4. Port pull-up/pull-down register (GPIOx_PUD, x = A…C) 152
7.4.5. Port input status register (GPIOx_ISTAT, x = A...C) 154
7.4.6. Port output control register (GPIOx_OCTL, x = A...C) 154
7.4.7. Port bit operate register (GPIOx_BOP, x = A…C) 155
7.4.8. Port configuration lock register (GPIOx_LOCK, x = A…C) 155
7.4.9. Alternate function selected register 0 (GPIOx_AFSEL0, x = A...C) 156
7.4.10. Alternate function selected register 1 (GPIOx_AFSEL1, x = A…C) 157
7.4.11. Bit clear register (GPIOx_BC, x = A...C) 158
7.4.12. Port bit toggle register (GPIOx_TG, x = A…C) 159
8. Cyclic redundancy checks management unit (CRC) 160
8.1. Overview 160
8.2. Characteristics 160
8.3. Function overview 161
8.4. Register definition 162
8.4.1. Data register (CRC_DATA) 162
8.4.2. Free data register (CRC_FDATA) 162
8.4.3. Control register (CRC_CTL) 163
9. True random number generator (TRNG) 164
9.1. Overview 164
9.2. Characteristics 164
9.3. Function overview 164
9.3.1. Operation flow 165
9.3.2. Error flags 165
9.4. Register definition 166
9.4.1. Control register (TRNG_CTL) 166
9.4.2. Status register (TRNG_STAT) 166
9.4.3. Data register (TRNG_DATA) 167
10. Direct memory access controller (DMA) 169
10.1. Overview 169
10.2. Characteristics 169
10.3. Block diagram 170
10.4. Function overview 170
10.4.1. Peripheral handshake 172
10.4.2. Data process 173

10.4.3. Address generation 178
10.4.4. Circular mode 179
10.4.5. Switch-buffer mode 179
10.4.6. Transfer flow controller 180
10.4.7. Transfer operation 180
10.4.8. Transfer finish 181
10.4.9. Channel configuration 183
10.5. Interrupts 184
10.5.1. Flag 184
10.5.2. Exception 185
10.5.3. Error 186
10.6. Register definition 188
10.6.1. Interrupt flag register 0 (DMA_INTF0) 188
10.6.2. Interrupt flag register 1 (DMA_INTF1) 189
10.6.3. Interrupt flag clear register 0 (DMA_INTC0) 189
10.6.4. Interrupt flag clear register 1 (DMA_INTC1) 190
10.6.5. Channel x control register (DMA_CHxCTL) (x = 0..7) 191
10.6.6. Channel x counter register (DMA_CHxCNT) (x = 0..7) 195
10.6.7. Channel x peripheral base address register (DMA_CHxPADDR) (x = 0..7) 195
10.6.8. Channel x memory 0 base address register (DMA_CHxM0ADDR) (x = 0..7) 196
10.6.9. Channel x memory 1 base address register (DMA_CHxM1ADDR) (x = 0..7) 196
10.6.10. Channel x FIFO control register (DMA_CHxFCTL) (x = 0..7) 197
11. Debug (DBG) 199
11.1. Overview 199
11.2. JTAG function overview 199
11.2.1. Pin assignment 199
11.2.2. JTAG daisy chained structure 199
11.2.3. Debug reset 200
11.3. Debug hold function overview 200
11.3.1. Debug support for power saving mode 200
11.3.2. Debug support for TIMER, I2C, WWDGT, FWDGT and RTC 200
11.4. Register definition 201
11.4.1. ID code register (DBG_ID) 201
11.4.2. Control register 0 (DBG_CTL0) 201
11.4.3. Control register 1 (DBG_CTL1) 202
11.4.4. Control register 2 (DBG_CTL2) 203
12. Analog to digital converter (ADC) 205
12.1. Overview 205
12.2. Characteristics 205

12.3. Pins and internal signals 206
12.4. Function overview 207
12.4.1. ADC clock 207
12.4.2. ADCON switch 207
12.4.3. Routine sequence 207
12.4.4. Operation modes 208
12.4.5. Conversion result threshold monitor function 211
12.4.6. Data storage mode 211
12.4.7. Sampling time configuration 212
12.4.8. External trigger configuration 212
12.4.9. DMA request 213
12.4.10. Overflow detection 213
12.4.11. ADC internal channels 214
12.4.12. Programmable resolution (DRES) 214
12.4.13. On-chip hardware oversampling 215
12.4.14. ADC interrupts 216
12.5. Register definition 217
12.5.1. Status register (ADC_STAT) 217
12.5.2. Control register 0 (ADC_CTL0) 218
12.5.3. Control register 1 (ADC_CTL1) 219
12.5.4. Sample time register 0 (ADC_SAMPT0) 221
12.5.5. Sample time register 1 (ADC_SAMPT1) 222
12.5.6. Watchdog high threshold register (ADC_WDHT) 222
12.5.7. Watchdog low threshold register (ADC_WDLT) 223
12.5.8. Routine sequence register 0 (ADC_RSQ0) 223
12.5.9. Routine sequence register 1 (ADC_RSQ1) 224
12.5.10. Routine sequence register 2 (ADC_RSQ2) 224
12.5.11. Routine data register (ADC_RDATA) 225
12.5.12. Oversampling control register (ADC_OVSAMPCTL) 225
12.5.13. Commom control register (ADC_CCTL) 227
13. Watchdog timer (WDGT) 228
13.1. Free watchdog timer (FWDGT) 228
13.1.1. Overview 228
13.1.2. Characteristics 228
13.1.3. Function overview 228
13.1.4. Register definition 231
13.2. Window watchdog timer (WWDGT) 234
13.2.1. Overview 234
13.2.2. Characteristics 234
13.2.3. Function overview 234
13.2.4. Register definition 237

14. Real time clock (RTC) 239
14.1. Overview 239
14.2. Characteristics 239
14.3. Function overview 240
14.3.1. Block diagram 240
14.3.2. Clock source and prescalers 240
14.3.3. Shadow registers introduction 241
14.3.4. Configurable and field maskable alarm 241
14.3.5. Configurable periodic auto-wakeup counter 242
14.3.6. RTC initialization and configuration 242
14.3.7. Calendar reading 243
14.3.8. Resetting the RTC 245
14.3.9. RTC shift function 245
14.3.10. RTC reference clock detection 246
14.3.11. RTC coarse digital calibration 246
14.3.12. RTC smooth digital calibration 247
14.3.13. Time-stamp function 249
14.3.14. Tamper detection 249
14.3.15. Calibration clock output 250
14.3.16. Alarm output 251
14.3.17. RTC power saving mode management 251
14.3.18. RTC interrupts 251
14.4. Register definition 253
14.4.1. Time register (RTC_TIME) 253
14.4.2. Date register (RTC_DATE) 253
14.4.3. Control register (RTC_CTL) 254
14.4.4. Status register (RTC_STAT) 257
14.4.5. Prescaler register (RTC_PSC) 259
14.4.6. Wakeup timer register (RTC_WUT) 259
14.4.7. Coarse calibration register (RTC_COSC) 260
14.4.8. Alarm 0 time and date register (RTC_ALRM0TD) 261
14.4.9. Alarm 1 time and date register (RTC_ALRM1TD) 262
14.4.10. Write protection key register (RTC_WPK) 263
14.4.11. Sub second register (RTC_SS) 263
14.4.12. Shift function control register (RTC_SHIFTCTL) 264
14.4.13. Time of time stamp register (RTC_TTS) 264
14.4.14. Date of time stamp register (RTC_DTS) 265
14.4.15. Sub second of time stamp register (RTC_SSTS) 266
14.4.16. High resolution frequency compensation register (RTC_HRFC) 266
14.4.17. Tamper register (RTC_TAMP) 267
14.4.18. Alarm 0 sub second register (RTC_ALRM0SS) 269
14.4.19. Alarm 1 sub second register (RTC_ALRM1SS) 270

14.4.20. Backup registers (RTC_BKPx) (x = 0…19) 271

15. Timer (TIMERx) 273

15.1. Advanced timer (TIMERx, x=0) 274

15.1.1. Overview 274

15.1.2. Characteristics 274

15.1.3. Block diagram 275

15.1.4. Function overview 275

15.1.5. TIMERx registers(x=0) 303

15.2. General level0 timer (TIMERx, x=1, 2) 330

15.2.1. Overview 330

15.2.2. Characteristics 330

15.2.3. Block diagram 330

15.2.4. Function overview 331

15.2.5. TIMERx registers(x=1, 2) 347

15.3. General level4 timer (TIMERx, x=15,16) 370

15.3.1. Overview 370

15.3.2. Characteristics 370

15.3.3. Block diagram 370

15.3.4. Function overview 371

15.3.5. TIMERx registers(x=15, 16) 385

15.4. Basic timer (TIMERx, x=5) 401

15.4.1. Overview 401

15.4.2. Characteristics 401

15.4.3. Block diagram 401

15.4.4. Function overview 401

15.4.5. TIMERx registers(x=5) 405

16. Universal synchronous/asynchronous receiver /transmitter (USART) 410

16.1. Overview 410

16.2. Characteristics 410

16.3. Function overview 412

16.3.1. USART frame format 412

16.3.2. Baud rate generation 413

16.3.3. USART transmitter 414

16.3.4. USART receiver 415

16.3.5. Use DMA for data buffer access 417

16.3.6. Hardware flow control 418

16.3.7. Multi-processor communication 419

16.3.8. LIN mode 420

16.3.9. Synchronous mode 421

16.3.10. IrDA SIR ENDEC mode 422

16.3.11. Half-duplex communication mode 424
16.3.12. Smartcard (ISO7816-3) mode 424
16.3.13. ModBus communication 426
16.3.14. Receive FIFO 426
16.3.15. Wakeup from deep-sleep mode 427
16.3.16. USART interrupts 427
16.4. Register definition 430
16.4.1. Control register 0 (USART_CTL0) 430
16.4.2. Control register 1 (USART_CTL1) 432
16.4.3. Control register 2 (USART_CTL2) 435
16.4.4. Baud rate generator register (USART_BAUD) 438
16.4.5. Prescaler and guard time configuration register (USART_GP) 438
16.4.6. Receiver timeout register (USART_RT) 439
16.4.7. Command register (USART_CMD) 440
16.4.8. Status register (USART_STAT) 441
16.4.9. Interrupt status clear register (USART_INTC) 444
16.4.10. Receive data register (USART_RDATA) 446
16.4.11. Transmit data register (USART_TDATA) 446
16.4.12. USART coherence control register (USART_CHC) 447
16.4.13. USART receive FIFO control and status register (USART_RFCS) 447
17. Inter-integrated circuit interface (I2C) 449
17.1. Overview 449
17.2. Characteristics 449
17.3. Function overview 449
17.3.1. Clock requirements 450
17.3.2. I2C communication flow 451
17.3.3. Noise filter 454
17.3.4. I2C timings 454
17.3.5. Software reset 456
17.3.6. Data transfer 456
17.3.7. I2C slave mode 459
17.3.8. I2C master mode 464
17.3.9. SMBus support 469
17.3.10. SMBus mode 472
17.3.11. Wakeup from Deep-sleep mode 474
17.3.12. Use DMA for data transfer 474
17.3.13. I2C error and interrupts 475
17.3.14. I2C debug mode 475
17.4. Register definition 476
17.4.1. Control register 0 (I2C_CTL0) 476
17.4.2. Control register 1 (I2C_CTL1) 478

17.4.3. Slave address register 0 (I2C_SADDR0) 480
17.4.4. Slave address register 1 (I2C_SADDR1) 481
17.4.5. Timing register (I2C_TIMING) 482
17.4.6. Timeout register (I2C_TIMEOUT) 483
17.4.7. Status register (I2C_STAT) 484
17.4.8. Status clear register (I2C_STATC) 487
17.4.9. PEC register (I2C_PEC) 488
17.4.10. Receive data register (I2C_RDATA) 488
17.4.11. Transmit data register (I2C_TDATA) 488
17.4.12. Control register 2 (I2C_CTL2) 489

18. Serial peripheral interface (SPI) 490

18.1. Overview 490

18.2. Characteristics 490

18.2.1. SPI characteristics 490

18.3. SPI function overview 490

18.3.1. SPI block diagram 490
18.3.2. SPI signal description 491
18.3.3. SPI clock timing and data format 491
18.3.4. NSS function 492
18.3.5. SPI operating modes 493
18.3.6. DMA function 499
18.3.7. CRC function 499
18.3.8. SPI interrupts 500

18.4. Register definition 502

18.4.1. Control register 0 (SPI_CTL0) 502
18.4.2. Control register 1 (SPI_CTL1) 504
18.4.3. Status register (SPI_STAT) 505
18.4.4. Data register (SPI_DATA) 506
18.4.5. CRC polynomial register (SPI_CRCPOLY) 506
18.4.6. RX CRC register (SPI_RCRC) 507
18.4.7. TX CRC register (SPI_TCRC) 508

19. Quad-SPI interface (QSPI) 509

19.1. Overview 509

19.2. Characteristics 509

19.3. Function overview 509

19.3.1. QSPI block diagram 509
19.3.2. QSPI command format 510
19.3.3. QSPI signal line modes 512
19.3.4. CSN and SCK 512

19.4. Operating modes 512

19.4.1. Normal mode 513
19.4.2. Read polling mode 514
19.4.3. Memory map mode 515
19.5. QSPI configuration 515
19.5.1. Flash configuration 515
19.5.2. IP configuration 515
19.6. Send instruction only once 516
19.7. Error and interrupts 516
19.8. Register definition 518
19.8.1. Control register (QSPI_CTL) 518
19.8.2. Device configuration register (QSPI_DCFG) 520
19.8.3. Status register (QSPI_STAT) 521
19.8.4. Status clear register (QSPI_STATC) 522
19.8.5. Data length register (QSPI_DTLEN) 523
19.8.6. Transfer configuration register (QSPI_TCFG) 524
19.8.7. Address register (QSPI_ADDR) 526
19.8.8. Alternate bytes register (QSPI_ALTE) 526
19.8.9. Data register (QSPI_DATA) 526
19.8.10. Status mask register (QSPI_STATMK) 527
19.8.11. Status match register (QSPI_STATMATCH) 527
19.8.12. Interval register (QSPI_INTERVAL) 528
19.8.13. Timeout register (QSPI_TMOUT) 528
19.8.14. FIFO flush register (QSPI_FLUSH) 529
20. Cryptographic Acceleration Unit (CAU) 530
20.1. Overview 530
20.2. Characteristics 530
20.3. CAU data type and initialization vectors 531
20.3.1. Data type 531
20.3.2. Initialization vectors 532
20.4. Cryptographic acceleration processor 532
20.4.1. DES / TDES cryptographic acceleration processor 533
20.4.2. AES cryptographic acceleration processor 537
20.5. Operating modes 544
20.6. CAU DMA interface 545
20.7. CAU interrupts 545
20.8. CAU suspended mode 546
20.9. Register definition 548
20.9.1. Control register (CAU_CTL) 548

20.9.2. Status register 0 (CAU_STAT0) 549
20.9.3. Data input register (CAU_DI) 550
20.9.4. Data output register (CAU_DO) 551
20.9.5. DMA enable register (CAU_DMAEN) 551
20.9.6. Interrupt enable register (CAU_INTEN) 552
20.9.7. Status register 1 (CAU_STAT1) 552
20.9.8. Interrupt flag register (CAU_INTF) 553
20.9.9. Key registers (CAU_KEY0…3(H / L)) 553
20.9.10. Initial vector registers (CAU_IV0…1(H / L)) 556
20.9.11. GCM or CCM mode context switch register x (CAU_GCMCCMCTXSx) (x = 0…7) 557
20.9.12. GCM mode context switch register x (CAU_GCMCTXSx) (x = 0…7) 558

21. Hash Acceleration Unit (HAU) 559

21.1. Overview 559

21.2. Characteristics 559

21.3. HAU data type 559

21.4. HAU core 561

21.4.1. Automatic data padding 561
21.4.2. Digest computing 562
21.4.3. Hash mode 563
21.4.4. HMAC mode 563

21.5. HAU suspended mode 563

21.5.1. Transfer data by CPU 564
21.5.2. Transfer data by DMA 564

21.6. HAU interrupt 565

21.6.1. Input FIFO interrupt 565
21.6.2. Calculation completion interrupt 565

21.7. Register definition 566

21.7.1. HAU control register (HAU_CTL) 566
21.7.2. HAU data input register (HAU_DI) 567
21.7.3. HAU configuration register (HAU_CFG) 568
21.7.4. HAU data output register (HAU_DO0…7) 569
21.7.5. HAU interrupt enable register (HAU_INTEN) 571
21.7.6. HAU status and flag register (HAU_STAT) 571
21.7.7. Context switch register x (HAU_CTXSx) (x = 0...53) 572

22. Public Key Cryptographic Acceleration Unit (PKCAU) 573

22.1. Overview 573

22.2. Characteristics 573

22.3. Function overview 573

22.3.1. Operands 574

22.3.2. RSA algorithm 574
22.3.3. ECC algorithm 576
22.3.4. Integer arithmetic operations 577
22.3.5. Elliptic curve operations in Fp domain 587
22.3.6. PKCAU operation process 593
22.3.7. Processing times 594
22.3.8. Status, errors and interrupts 595
22.4. Register definition 597
22.4.1. Control register (PKCAU_CTL) 597
22.4.2. Status register (PKCAU_STAT) 598
22.4.3. Status clear register (PKCAU_STATC) 599
23. Infrared ray port (IFRP) 601
23.1. Overview 601
23.2. Characteristics 601
23.3. Function overview 601
24. Wireless 603
24.1. Overview 603
24.2. Wi-Fi 603
24.2.1. Characteristics 603
24.3. BLE 604
24.3.1. Characteristics 604
24.4. Radio 605
25. Appendix 606
25.1. List of abbreviations used in register 606
25.2. List of terms 606
25.3. Available peripherals 607
26. Revision history 608

List of Figures

Figure 1-1. GD32VW55x system architecture 25
Figure 2-1. Process of page erase operation 43
Figure 2-2. Process of mass erase operation 44
Figure 2-3. Process of word program operation 46
Figure 3-1. Block diagram of Efuse controller 61
Figure 4-1. Power supply overview 74
Figure 4-2. Waveform of the POR / PDR 76
Figure 4-3. Waveform of the LVD threshold 76
Figure 4-4. RF sequence 80
Figure 5-1. The system reset circuit 96
Figure 5-2. Clock tree 97
Figure 5-3. HXTAL clock source 98
Figure 5-4. HXTAL clock source in bypass mode 99
Figure 6-1. Block diagram of EXTI 134
Figure 7-1. Basic structure of a general-pupose I/O 142
Figure 7-2. Basic structure of Input configuration 144
Figure 7-3. Basic structure of Output configuration 145
Figure 7-4. Basic structure of Analog configuration 145
Figure 7-5. Basic structure of Alternate function configuration 146
Figure 8-1. Block diagram of CRC calculation unit 160
Figure 9-1. TRNG block diagram 164
Figure 10-1. Block diagram of DMA 170
Figure 10-2. Data stream for three transfer modes 171
Figure 10-3. Handshake mechanism 172
Figure 10-4. Data packing/unpacking when PWIDTH = ‘00’ 177
Figure 10-5. Data packing / unpacking when PWIDTH = ‘01’ 178
Figure 10-6. Data packing/unpacking when PWIDTH = ‘10’ 178
Figure 10-7. DMA operation of switch-buffer mode 180
Figure 10-8. System connection of DMA 187
Figure 12-1. ADC module block diagram 207
Figure 12-2. Single operation mode 208
Figure 12-3. Continuous conversion mode 209
Figure 12-4. Scan operation mode, continuous disable 210
Figure 12-5. Scan operation mode, continuous enable 210
Figure 12-6. Discontinuous operation mode 210
Figure 12-7. Data storage mode of 12-bit resolution 211
Figure 12-8. Data storage mode of 10-bit resolution 211
Figure 12-9. Data storage mode of 8-bit resolution 212
Figure 12-10. Data storage mode of 6-bit resolution 212
Figure 12-11. 20-bit to 16-bit result truncation 215

Figure 12-12. A numerical example with 5-bit shifting and rounding 216
Figure 13-1. Free watchdog timer block diagram 229
Figure 13-2. Window watchdog timer block diagram 235
Figure 13-3. Window watchdog timing diagram 236
Figure 14-1. Block diagram of RTC 240
Figure 15-1. Advanced timer block diagram 275
Figure 15-2. Normal mode, internal clock divided by 1 276
Figure 15-3. Timing chart of PSC value change from 0 to 2 277
Figure 15-4. Timing chart of up counting mode, PSC=0/2 278
Figure 15-5. Timing chart of up counting mode, change TIMERx_CAR ongoing 278
Figure 15-6. Timing chart of down counting mode, PSC=0/2 279
Figure 15-7. Timing chart of down counting mode, change TIMERx_CAR ongoing 280
Figure 15-8. Timing chart of center-aligned counting mode 281
Figure 15-9. Repetition timechart for center-aligned counter 282
Figure 15-10. Repetition timechart for up-counter 282
Figure 15-11. Repetition timechart for down-counter 283
Figure 15-12. Channel input capture principle 284
Figure 15-13. Channel output compare principle (with complementary output, x=0,1,2) 285
Figure 15-14. Channel output compare principle (CH3_O) 285
Figure 15-15. Output-compare in three modes 287
Figure 15-16. Timing chart of EAPWM 288
Figure 15-17. Timing chart of CAPWM 288
Figure 15-18. Complementary output with dead-time insertion. 291
Figure 15-19. Output behavior of the channel in response to a break (the break high active) 292
Figure 15-20. Counter behavior with CI0FE0 polarity non-inverted in mode 2 293
Figure 15-21. Counter behavior with CI0FE0 polarity inverted in mode 2 293
Figure 15-22. Hall sensor is used to BLDC motor 294
Figure 15-23. Hall sensor timing between two timers 295
Figure 15-24. Restart mode 296
Figure 15-25. Pause mode 296
Figure 15-26. Event mode 297
Figure 15-27. Single pulse mode, TIMERx_CHxCV = 4, TIMERx_CAR=99 298
Figure 15-28. Trigger mode of TIMER0 controlled by enable signal of TIMER2 299
Figure 15-29. Trigger mode of TIMER0 controlled by update signal of TIMER2 299
Figure 15-30. Pause mode of TIMER0 controlled by enable signal of TIMER2 300
Figure 15-31. Pause mode of TIMER0 controlled by O0CPREF signal of TIMER2 300
Figure 15-32. Trigger TIMER0 and TIMER2 by the CI0 signal of TIMER2 301
Figure 15-33. General Level 0 timer block diagram 331
Figure 15-34. Normal mode, internal clock divided by 1 332
Figure 15-35. Timing chart of PSC value change from 0 to 2 333
Figure 15-36. Timing chart of up counting mode, PSC=0/2 334
Figure 15-37. Timing chart of up counting mode, change TIMERx_CAR ongoing 334
Figure 15-38. Timing chart of down counting mode,PSC=0/2 335

Figure 15-39. Timing chart of down counting mode, change TIMERx_CAR ongoing 336
Figure 15-40. Timing chart of center-aligned counting mode 337
Figure 15-41. Channel input capture principle 338
Figure 15-42. Channel output compare principle (x=0,1,2,3) 339
Figure 15-43. Output-compare in three modes 340
Figure 15-44. Timing chart of EAPWM 341
Figure 15-45. Timing chart of CAPWM 341
Figure 15-46. Restart mode 343
Figure 15-47. Pause mode 344
Figure 15-48. Event mode 344
Figure 15-49. Single pulse mode, TIMERx_CHxCV = 4, TIMERx_CAR=99 345
Figure 15-50. General level4 timer block diagram 371
Figure 15-51. Timing chart of internal clock divided by 1 372
Figure 15-52. Timing chart of PSC value change from 0 to 2 372
Figure 15-53. Up-counter timechart, PSC=0/2 373
Figure 15-54. Up-counter timechart, change TIMERx_CAR on the go 374
Figure 15-55. Repetition timechart for up-counter 375
Figure 15-56. Channel input capture principle 376
Figure 15-57. Channel output compare principle (with complementary output, x=0) 377
Figure 15-58. Output-compare under three modes 378
Figure 15-59. PWM mode timechart 379
Figure 15-60. Complementary output with dead-time insertion. 381
Figure 15-61. Output behavior in response to a break(The break high active) 382
Figure 15-62. Single pulse mode TIMERx_CHxCV = 0x04 TIMERx_CAR=0x99 383
Figure 15-63. Basic timer block diagram 401
Figure 15-64. Timing chart of internal clock divided by 1 402
Figure 15-65. Timing chart of PSC value change from 0 to 2 402
Figure 15-66. Timing chart of up counting mode, PSC=0/2 403
Figure 15-67. Timing chart of up counting mode, change TIMERx_CAR ongoing 404
Figure 16-1. USART module block diagram 412
Figure 16-2. USART character frame (8 bits data and 1 stop bit) 412
Figure 16-3. USART transmit procedure 415
Figure 16-4. Oversampling method of a receive frame bit (OSB=0) 416
Figure 16-5. Configuration step when using DMA for USART transmission 417
Figure 16-6. Configuration step when using DMA for USART reception 418
Figure 16-7. Hardware flow control between two USARTs 418
Figure 16-8. Hardware flow control 419
Figure 16-9. Break frame occurs during idle state 421
Figure 16-10. Break frame occurs during a frame 421
Figure 16-11. Example of USART in synchronous mode 422
Figure 16-12. 8-bit format USART synchronous waveform (CLEN=1) 422
Figure 16-13. IrDA SIR ENDEC module 423
Figure 16-14. IrDA data modulation 423
Figure 16-15. ISO7816-3 frame format 424

Figure 16-16. USART receive FIFO structure 427
Figure 16-17. USART interrupt mapping diagram 429
Figure 17-1. I2C module block diagram 450
Figure 17-2. Data validation 451
Figure 17-3. START and STOP condition 452
Figure 17-4. I2C communication flow with 10-bit address (Master Transmit) 452
Figure 17-5. I2C communication flow with 7-bit address (Master Transmit) 453
Figure 17-6. I2C communication flow with 7-bit address (Master Receive) 453
Figure 17-7. I2C communication flow with 10-bit address (Master Receive when HEAD10R=0) 453
Figure 17-8. I2C communication flow with 10-bit address (Master Receive when HEAD10R=1) 453
Figure 17-9. Data hold time 454
Figure 17-10. Data setup time 455
Figure 17-11. Data transmission 457
Figure 17-12. Data reception 457
Figure 17-13. I2C initialization in slave mode 461
Figure 17-14. Programming model for slave transmitting when SS=0 462
Figure 17-15. Programming model for slave transmitting when SS=1 463
Figure 17-16. Programming model for slave receiving 464
Figure 17-17. I2C initialization in master mode 465
Figure 17-18. Programming model for master transmitting (N<=255) 466
Figure 17-19. Programming model for master transmitting (N>255) 467
Figure 17-20. Programming model for master receiving (N<=255) 468
Figure 17-21. Programming model for master receiving (N>255) 469
Figure 17-22. SMBus Master Transmitter and Slave Receiver communication flow 473
Figure 17-23. SMBus Master Receiver and Slave Transmitter communication flow 474
Figure 18-1. Block diagram of SPI 490
Figure 18-2. SPI timing diagram in normal mode 492
Figure 18-3. A typical full-duplex connection 495
Figure 18-4. A typical simplex connection (Master: Receive, Slave: Transmit) 495
Figure 18-5. A typical simplex connection (Master: Transmit only, Slave: Receive) 495
Figure 18-6. A typical bidirectional connection 495
Figure 18-7. Timing diagram of TI master mode with discontinuous transfer 497
Figure 18-8. Timing diagram of TI master mode with continuous transfer 497
Figure 18-9. Timing diagram of TI slave mode 498
Figure 19-1 QSPI diagram 510
Figure 19-2 QSPI command format 511
Figure 20-1. DATAM No swapping and Half-word swapping 531
Figure 20-2. DATAM Byte swapping and Bit swapping 532
Figure 20-3. CAU diagram 532
Figure 20-4. DES / TDES ECB encryption 534
Figure 20-5. DES / TDES ECB decryption 535
Figure 20-6. DES / TDES CBC encryption 536

Figure 20-7. DES / TDES CBC decryption 537
Figure 20-8. AES ECB encryption 538
Figure 20-9. AES ECB decryption 538
Figure 20-10. AES CBC encryption 539
Figure 20-11. AES CBC decryption 540
Figure 20-12. Counter block structure 540
Figure 20-13. AES CTR encryption / decryption 541
Figure 21-1. DATAM No swapping and Half-word swapping 560
Figure 21-2. DATAM Byte swapping and Bit swapping 560
Figure 21-3. HAU block diagram 561
Figure 22-1. PKCAU module block diagram 574
Figure 22-2. Flow chart of RSA algorithm 575
Figure 22-3. Flow chart of ECDSA sign 576
Figure 22-4. Flow chart of ECDSA verification 577
Figure 22-5. Arithmetic addition 578
Figure 22-6. Arithmetic subtraction 579
Figure 22-7. Arithmetic multiplication 579
Figure 22-8. Arithmetic comparison 580
Figure 22-9. Modular reduction 580
Figure 22-10. Modular addition 581
Figure 22-11. Modular subtraction 582
Figure 22-12. Montgomery parameter calculation 582
Figure 22-13. Mutual mapping between Montgomery domain and natural domain 583
Figure 22-14. Montgomery multiplication 584
Figure 22-15. Modular exponentiation of normal mode 584
Figure 22-16. Modular exponentiation of fast mode 585
Figure 22-17. Modular inversion 585
Figure 22-18. RSA CRT exponentiation 586
Figure 22-19. Point on elliptic curve Fp check 588
Figure 22-20. ECC scalar multiplication of normal mode 589
Figure 22-21. ECC scalar multiplication of fast mode 589
Figure 22-22. ECDSA sign 591
Figure 22-23. ECDSA verification 592
Figure 23-1. IFRP output timechart 1 601
Figure 23-2. IFRP output timechart 2 602
Figure 23-3. IFRP output timechart 3 602

List of Tables

Table 1-1. The interconnection relationship of the AHB interconnect matrix 24
Table 1-2. Memory map of GD32VW55x devices 26
Table 1-3. BOOT0 modes 30
Table 1-4. BOOT1 modes 30
Table 1-5. Boot address modes 30
Table 2-1. Base address and size for flash memory 40
Table 2-2. Option bytes 47
Table 2-3. Flash operation under different protection levels 48
Table 2-4. Flash interrupt requests 49
Table 3-1. Efuse address mapping 62
Table 3-2. System parameters 62
Table 4-1. Time in RF sequence 81
Table 4-2. Power saving mode summary 82
Table 5-1. Clock output 0 source select 101
Table 5-2. Clock output 1 source select 101
Table 5-3. 1.1V domain voltage selected in deep-sleep mode 102
Table 6-1. Interrupt vector table 132
Table 6-2. EXTI source 135
Table 7-1. GPIO configuration table 142
Table 10-1. Transfer mode 171
Table 10-2. Peripheral requests to DMA 173
Table 10-3. CNT configuration 174
Table 10-4. FIFO counter critical value configuration rules 176
Table 10-5. DMA interrupt events 184
Table 12-1. ADC internal input signals 206
Table 12-2. ADC input pins definition 206
Table 12-3. External trigger modes 212
Table 12-4. External trigger source for ADC 212
Table 12-5. t_{CONV} timings depending on resolution 214
Table 12-6. Maximum output results for N and M combimations (grayed values indicates truncation) 216
Table 13-1. Min/max FWDGT timeout period at 32 kHz (IRC32K) 229
Table 13-2. Min-max timeout value at 42 MHz (f_{PCLK1}) 236
Table 14-1. RTC power saving mode management 251
Table 14-2. RTC interrupts control 251
Table 15-1. Timers (TIMERx) are divided into four sorts 273
Table 15-2. Complementary outputs controlled by parameters 290
Table 15-3. Counting direction in different quadrature decoder mode 293
Table 15-4. Examples of slave mode 295
Table 15-5. Examples of slave mode 342

Table 15-6. Complementary outputs controlled by parameters 380
Table 16-1. Description of USART important pins 412
Table 16-2. Configuration of stop bits 413
Table 16-3. USART interrupt requests 427
Table 17-1. Definition of I2C-bus terminology (refer to the I2C specification of Philips semiconductors) 450
Table 17-2. Data setup time and data hold time 456
Table 17-3. Communication modes to be shut down 458
Table 17-4. I2C configuration 458
Table 17-5. SMBus with PEC configuration 471
Table 17-6. I2C error flags 475
Table 17-7. I2C interrupt events 475
Table 18-1. SPI signal description 491
Table 18-2. NSS function in slave mode 492
Table 18-3. NSS function in master mode 493
Table 18-4. SPI operating modes 493
Table 18-5. SPI interrupt requests 501
Table 19-1. QSPI signal description 509
Table 19-2. QSPI command description 511
Table 19-3. QSPI singnal line modes 512
Table 19-4. AHB write access mode and number of bytes add to FIFO 514
Table 19-5. TERR and AHB error conditions 516
Table 19-6. QSPI interrupt events 516
Table 22-1. Parameters of RSA algorithm 575
Table 22-2. Integer arithmetic operations 577
Table 22-3. Range of parameters used by RSA CRT exponentiation operation 587
Table 22-4. Elliptic curve operations in Fp domain 587
Table 22-5. Range of parameters used by point on elliptic curve Fp check 588
Table 22-6. Range of parameters used by ECC scalar multiplication 590
Table 22-7. Range of parameters used by ECDSA sign 591
Table 22-8. Range of parameters used by ECDSA verification 593
Table 22-9. Modular exponentiation computation times 594
Table 22-10. ECC scalar multiplication computation times 595
Table 22-11. ECDSA signature average computation times 595
Table 22-12. ECDSA verification average computation times 595
Table 22-13. Montgomery parameters average computation times 595
Table 22-14. PKCAU interrupt requests 596
Table 25-1. List of abbreviations used in register 606
Table 25-2. List of terms 606
Table 26-1. Revision history 608

1. System and memory architecture

The devices of GD32VW55x series devices are 32-bit general-purpose microcontrollers based on the Nuclei N307 processor. The N307 processor is based on the RSIC-V architecture instruction set, hereinafter referred to as the RISC-V processor. The RISC-V processor includes two AHB buses known as I-Cache bus and System bus. All memory accesses of the RISC-V processor are executed on the two buses according to the different purposes and the target memory spaces. The memory organization uses a Harvard architecture, pre-defined memory map and up to 4 GB of memory space, making the system flexible and extendable.

1.1. RISC-V processor

RISC-V CPU target for embedded applications that require low energy consumption and small area, which is compliant to RISC-V architecture with several efficient micro-architecture features, including simple dynamic branch prediction, instruction pre-fetch buffers and I-cache. Visit **http://user.nucleisys.com** for more information about the N307 core. It supports 64 general purpose registers (GPRs):

- 3-pipeline stages, using state-of-the-art processor micro-architecture to deliver the best-of-class performance efficiency and lowest cost.
- Machine (M) and User (U) Privilege levels support.
- Non-maskable interrupt (NMI) support.
- Support dynamic branch predictor.
- Configurable instruction prefetch logic, which can prefetch subsequent two instructions to hide the instruction memory access latency.
- Support WFI (Wait For Interrupt) and WFE (Wait For Event) scheme to enter sleep mode.
- Interrupt priority levels configurable and programmable.
- Enhancement of vectored interrupt handling for real-time performance.
- Support interrupt preemption with priority.
- Support interrupt tail chaining.
- Standard 4-wire JTAG debug port and 2-wire cJTAG debug port.
- Support interactive debug functionalities.
- Support 8 triggers for hardware breakpoint.
- RV32I / M / A / F / D / C / P / B instruction extensions support.
- Support two-level sleep modes: shallow sleep mode, and deep sleep mode.
- Support 64-bits widereal-time counter (can be used as System Tick).
- Support Physical Memory Protection (PMP) to protect the memory, 8 entries.
- Support Instruction Cache (I-Cache), 2-way associative, cache line size 32 bytes, total 32KB.
- Support single / double precision FPU.
- Support 2 cycle floating point MAC.
- Support packed-SIMD DSP.

1.2. System architecture

A 32-bit multilayer bus is implemented in the GD32VW55x devices, which enables parallel access paths between multiple masters and slaves in the system. The multilayer bus consists of an AHB interconnect matrix, three AHB buses and two APB buses. The interconnection relationship of the AHB interconnect matrix is shown below. In the following table, “1” indicates the corresponding master is able to access the corresponding slave through the AHB interconnect matrix, while the ‘0’ means the corresponding master cannot access the corresponding slave through the AHB interconnect matrix.

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

	F-C BUS	S BUS	DMAM	WIFI	DMAP
FMC	1	1	1	0	1
SRAM0	1	1	1	1	1
AHB1	0	1	1	0	1
QSPI	1	1	1	1	1
AHB2	0	1	1	0	1
SRAM1	1	1	1	1	1
SRAM2	1	1	1	1	1
SRAM3	1	1	1	1	1
APB1	0	1	1	0	1
APB2	0	1	1	0	1
BLE	1	1	1	1	1

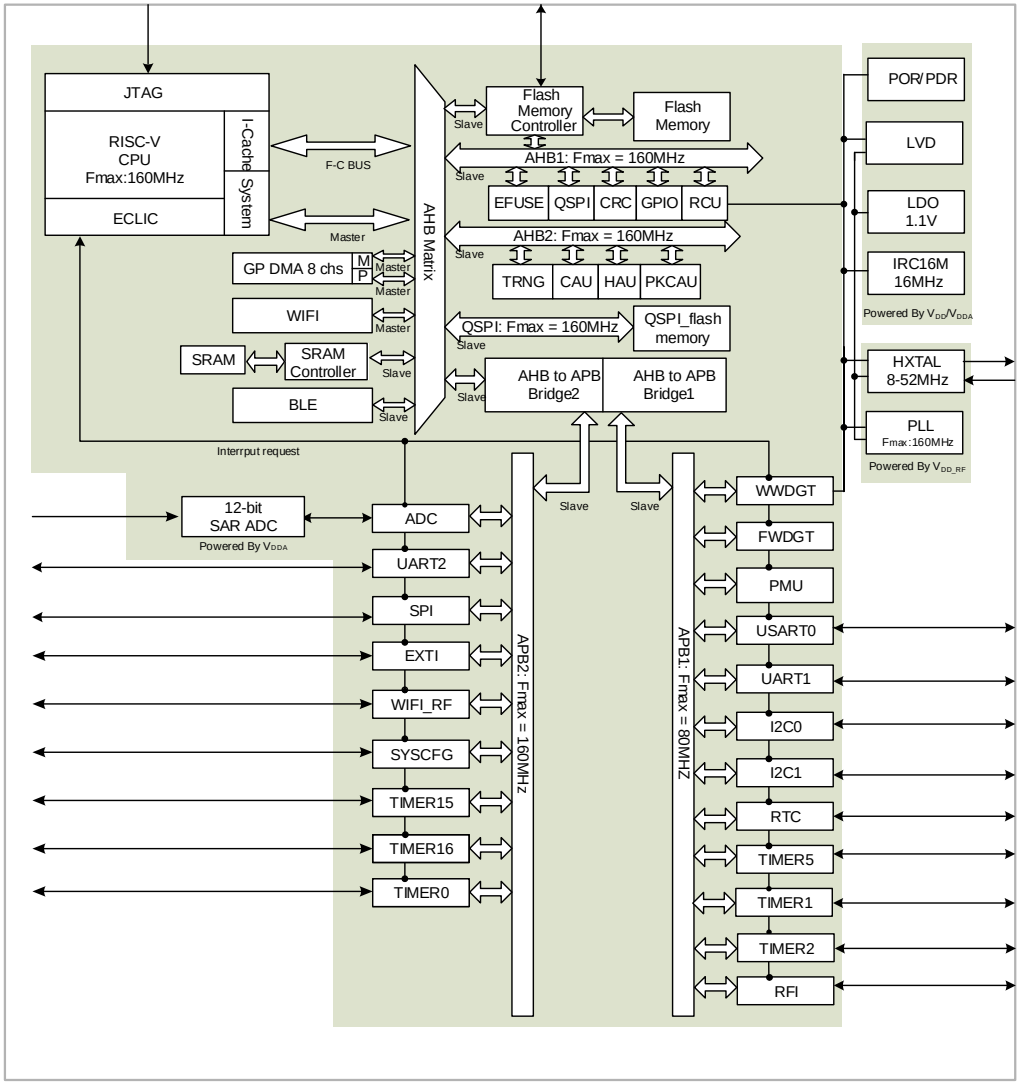
As is shown above, there are several masters connected with the AHB interconnect matrix, including F-CBUS, SBUS, DMAM, DMAP and WIFI. F-CBUS is the code bus of the RISC-V core. F-CBUS is used for instruction fetch to the internal memories when cache is enabled, the target of F-CBUS are the internal Flash, the external memory (QSPI_flash), BLE and internal SRAMs (SRAM0, SRAM1, SRAM2 and SRAM3). Similarly, SBUS is the system bus of the RISC-V core, which is used for instruction / vector fetches, data loading / storing and debugging access of the system regions. The System regions include the internal SRAM region, the external memory (QSPI_flash), BLE and the Peripheral region. DMAM is the memory bus of DMA, which is used by the DMA to perform transfer to/from memories, and the targets of this bus are the internal Flash, internal SRAMs, the external memory (QSPI_flash), BLE and the peripheral region. DMAP is the peripheral bus of DMA, which is used by the DMA to access AHB peripherals or to perform memory-to-memory transfers.The targets of this bus are the AHB and APB peripherals, plus data memories: the internal Flash, internal SRAMs (SRAM0, SRAM1,SRAM2 and SRAM3), BLE and external memory (QSPI_flash). WIFI is the bus connects the AHB master interface of the WIFI to the BusMatrix, the targets of this bus are the SRAMs (SRAM0, SRAM1, SRAM2 and SRAM3), BLE and the external memory (QSPI_flash).

There are also several slaves connected with the AHB interconnect matrix, including FMC,

SRAM0, SRAM1, SRAM2, SRAM3, AHB1, AHB2, APB1, APB2, QSPI and BLE. FMC is the bus interface of the flash memory controller. SRAM0~ SRAM3 is on-chip static random access memories. AHB1 is the AHB bus connected with all of the AHB1 slaves. AHB2 is the AHB bus connected with AHB2 slaves. QSPI is the bus interface of external memory (QSPI_flash). BLE is the bus interface of BLE. While APB1 and APB2 are the two APB buses connected with all of the APB slaves. The two APB buses connect with all the APB peripherals. APB1 is limited to 80Mhz, APB2 is limited to 160Mhz.

These are interconnected using a multilayer AHB bus architecture as shown in ***Figure 1-1. GD32VW55x system architecture***.

Figure 1-1. GD32VW55x system architecture



1.3. Memory map

The RISC-V processor is structured in Harvard architecture which can use separate buses to fetch instructions and load/store data. The instruction code and data are both located in the same memory address space but in different address ranges. Program memory, data memory,

registers and I/O ports are organized within the same linear 4-Gbyte address space. The maximum address range of the RISC-V is 4-Gbyte due to its 32-bit bus address width. Additionally, a pre-defined memory map is provided by the RISC-V processor to reduce the software complexity of repeated implementation of different device vendors. In the map, some regions are used by the RISC-V system peripherals which can not be modified. However, the other regions are available to the vendors. ***Table 1-2. Memory map of GD32VW55x devices*** shows the memory map of the GD32VW55x devices, including Code, SRAM, peripheral, and other pre-defined regions. Almost each peripheral is allocated 1KB of space. This allows simplifying the address decoding for each peripheral.

Table 1-2. Memory map of GD32VW55x devices

Pre-defined Regions	Bus	Address	Peripherals
External device	QSPI	0x9800 0000 – 0xD0FF FFFF	Reserved
		0x9000 0000 - 0x97FF FFFF	QSPI_FLASH (MEM)
		0x7000 0000 - 0x8FFF FFFF	Reserved
		0x6000 0000 - 0x67FF FFFF	Reserved
Peripheral	AHB2	0x4C06 3000 - 0x4FFF FFFF	Reserved
		0x4C06 1000 - 0x4C06 2FFF	PKCAU
		0x4C06 0C00 - 0x4C06 0FFF	Reserved
		0x4C06 0800 - 0x4C06 0BFF	TRNG
		0x4C06 0400 - 0x4C06 07FF	HAU
		0x4C06 0000 - 0x4C06 03FF	CAU
		0x4C05 0400 - 0x4C05 FFFF	Reserved
		0x4C05 0000 - 0x4C05 03FF	Reserved
		0x4C04 0000 - 0x4C04 FFFF	Reserved
		0x4C00 0000 - 0x4C03 FFFF	Reserved
	AHB1	0x4904 0000 - 0x4BFF FFFF	Reserved
		0x4900 0000 - 0x4903 FFFF	Reserved
		0x400B 1000 - 0x48FF FFFF	Reserved
		0x400B 0800 - 0x400B 0FFF	Reserved
		0x400B 0400 - 0x400B 07FF	Reserved
		0x400B 0000 - 0x400B 03FF	Reserved
		0x400A 1000 - 0x400A FFFF	Reserved
		0x400A 0C00 - 0x400A 0FFF	Reserved
		0x400A 0800 - 0x400A 0BFF	Reserved
		0x400A 0400 - 0x400A 07FF	Reserved
		0x400A 0000 - 0x400A 03FF	Reserved
		0x4008 0400 - 0x4009 FFFF	Reserved
		0x4008 0000 - 0x4008 03FF	Reserved
		0x4003 0000 - 0x4007 FFFF	WIFI
		0x4002 BC00 - 0x4002 FFFF	Reserved
		0x4002 B000 - 0x4002 BBFF	Reserved

Pre-defined Regions	Bus	Address	Peripherals
		0x4002 A000 - 0x4002 AFFF	Reserved
		0x4002 8000 - 0x4002 9FFF	Reserved
		0x4002 6800 - 0x4002 7FFF	Reserved
		0x4002 6400 - 0x4002 67FF	Reserved
		0x4002 6000 - 0x4002 63FF	DMA
		0x4002 5C00 - 0x4002 5FFF	Reserved
		0x4002 5800 - 0x4002 5BFF	QSPI_FLASH(REG)
		0x4002 5400 - 0x4002 57FF	Reserved
		0x4002 5000 - 0x4002 53FF	Reserved
		0x4002 4000 - 0x4002 4FFF	Reserved
		0x4002 3C00 - 0x4002 3FFF	Reserved
		0x4002 3800 - 0x4002 3BFF	RCU
		0x4002 3400 - 0x4002 37FF	Reserved
		0x4002 3000 - 0x4002 33FF	CRC
		0x4002 2C00 - 0x4002 2FFF	Reserved
		0x4002 2800 - 0x4002 2BFF	EFUSE
		0x4002 2400 - 0x4002 27FF	Reserved
		0x4002 2000 - 0x4002 23FF	FMC
		0x4002 1C00 - 0x4002 1FFF	Reserved
		0x4002 1800 - 0x4002 1BFF	Reserved
		0x4002 1400 - 0x4002 17FF	Reserved
		0x4002 1000 - 0x4002 13FF	Reserved
		0x4002 0C00 - 0x4002 0FFF	Reserved
		0x4002 0800 - 0x4002 0BFF	GPIOC
		0x4002 0400 - 0x4002 07FF	GPIOB
		0x4002 0000 - 0x4002 03FF	GPIOA
	APB2	0x4001 8800 - 0x4001 FFFF	Reserved
		0x4001 8400 - 0x4001 87FF	TIMER16
		0x4001 8000 - 0x4001 83FF	TIMER15
		0x4001 7C00 - 0x4001 7FFF	Reserved
		0x4001 7800 - 0x4001 7BFF	WIFI_RF
		0x4001 6800 - 0x4001 77FF	Reserved
		0x4001 6000 - 0x4001 67FF	Reserved
		0x4001 5800 - 0x4001 5FFF	Reserved
		0x4001 5400 - 0x4001 57FF	Reserved
		0x4001 4C00 - 0x4001 53FF	Reserved
		0x4001 4800 - 0x4001 4BFF	Reserved
		0x4001 4400 - 0x4001 47FF	Reserved
		0x4001 4000 - 0x4001 43FF	Reserved
		0x4001 3C00 - 0x4001 3FFF	EXTI

Pre-defined Regions	Bus	Address	Peripherals
		0x4001 3800 - 0x4001 3BFF	SYSCFG
		0x4001 3400 - 0x4001 37FF	Reserved
		0x4001 3000 - 0x4001 33FF	SPI
		0x4001 2C00 - 0x4001 2FFF	Reserved
		0x4001 2400 - 0x4001 2BFF	Reserved
		0x4001 2000 - 0x4001 23FF	ADC
		0x4001 1400 - 0x4001 1FFF	Reserved
		0x4001 1000 - 0x4001 13FF	UART2
		0x4001 0800 - 0x4001 0FFF	Reserved
		0x4001 0400 - 0x4001 07FF	Reserved
		0x4001 0000 - 0x4001 03FF	TIMER0
	APB1	0x4000 D000 - 0x4000 FFFF	Reserved
		0x4000 CC00 - 0x4000 CFFF	RFI
		0x4000 7400 - 0x4000 CBFF	Reserved
		0x4000 7000 - 0x4000 73FF	PMU
		0x4000 6C00 - 0x4000 6FFF	Reserved
		0x4000 5C00 - 0x4000 6BFF	Reserved
		0x4000 5800 - 0x4000 5BFF	I2C1
		0x4000 5400 - 0x4000 57FF	I2C0
		0x4000 4C00 - 0x4000 53FF	Reserved
		0x4000 4800 - 0x4000 4BFF	USART0
		0x4000 4400 - 0x4000 47FF	UART1
		0x4000 4000 - 0x4000 43FF	Reserved
		0x4000 3C00 - 0x4000 3FFF	Reserved
		0x4000 3800 - 0x4000 3BFF	Reserved
		0x4000 3400 - 0x4000 37FF	Reserved
		0x4000 3000 - 0x4000 33FF	FWDGT
		0x4000 2C00 - 0x4000 2FFF	WWDGT
		0x4000 2800 - 0x4000 2BFF	RTC
		0x4000 2400 - 0x4000 27FF	Reserved
		0x4000 2000 - 0x4000 23FF	Reserved
		0x4000 1C00 - 0x4000 1FFF	Reserved
		0x4000 1800 - 0x4000 1BFF	Reserved
		0x4000 1400 - 0x4000 17FF	Reserved
		0x4000 1000 - 0x4000 13FF	TIMER5
		0x4000 0C00 - 0x4000 0FFF	Reserved
		0x4000 0800 - 0x4000 0BFF	Reserved
		0x4000 0400 - 0x4000 07FF	TIMER2
		0x4000 0000 - 0x4000 03FF	TIMER1
SRAM	AHB	0x2101 0000 - 0x3FFF FFFF	Reserved

Pre-defined Regions	Bus	Address	Peripherals
		0x2100 0000 - 0x2100 FFFF	BLE
		0x2005 0000 - 0x20FF FFFF	Reserved
		0x2003 0000 - 0x2004 FFFF	SRAM3 (96KB + shared 32KB)
		0x2002 0000 - 0x2002 FFFF	SRAM2 (64KB)
		0x2001 0000 - 0x2001 FFFF	SRAM1 (64KB)
		0x2000 0000 - 0x2000 FFFF	SRAM0 (64KB)
Code	AHB	0x1000 0000 - 0x1FFF FFFF	External memories remap
		0x0FFC 0100 - 0x0FFF FFFF	Reserved
		0x0FFC 0000 - 0x0FFC 00FF	EFUSE (256 bytes)
		0x0BF8 0000 –0x0FFB FFFF	Reserved
		0x0BF4 0000 - 0x0BF7 FFFF	ROM(256KB)
		0x0A07 0000 - 0x0BF3 FFFF	Reserved
		0x0A04 0000 - 0x0A06 FFFF	Reserved
		0x0A02 0000 - 0x0A03 FFFF	Reserved
		0x0A01 0000 - 0x0A01 FFFF	Reserved
		0x0A00 0000 - 0x0A00 FFFF	Reserved
		0x0840 0000 - 0x09FF FFFF	Reserved
		0x0800 0000 - 0x083F FFFF	Flash memory
		0x0000 0000 - 0x07FF FFFF	External memories remap

1.3.1. On-chip SRAM memory

The GD32VW55x series of devices contain up to 288 KB + shared 32KB on-chip SRAM (SRAM0 64 KB, SRAM1 64 KB, SRAM2 64 KB, SRAM3 96KB + shared 32KB) which starts at address 0x2000 0000. It supports byte, half-word (16 bits), and word (32 bits) accesses.

1.3.2. On-chip flash memory overview

The devices provide up to 4096 KB of on-chip flash memory which is organized into 1024 pages with 4KB capacity per page, and 256KB information block for the boot loader.

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

1.4. Boot configuration

At startup, BOOT0 value and BOOT1 value are used to select the boot address. BOOT0 value and BOOT1 value are determined by configurations shown in ***Table 1-3. BOOT0 modes*** and ***Table 1-4. BOOT1 modes*** respectively.

- The BOOT0 value may come from the BOOT0 pin or from the value of SWBOOT0 bit in the EFUSE_CTL0 register to free the GPIO pad if needed.

- The BOOT1 value may come from the PB1 pin or from the value of SWBOOT1 bit in the EFUSE_CTL0 register to free the GPIO pad if needed.

Table 1-3. BOOT0 modes

SWBOOT0	EFBOOT0	BOOT0 PC8 pin	BOOT0
0	-	0	0
0	-	1	1
1	0	-	0
1	1	-	1

Table 1-4. BOOT1 modes

SWBOOT1	EFBOOT1	BOOT1 PB1 pin	BOOT1
0	-	0	0
0	-	1	1
1	0	-	0
1	1	-	1

Refer to ***Table 1-5. Boot address modes*** for boot address.

When BOOT0 value is 0:

- The boot address is selected according to EFSB bit value in EFUSE_CTL0 register.

When BOOT0 value is 1:

- When the EFBOOTLK bit in the EFUSE_CTL0 register is 0, the boot address is selected according to BOOT1 value.

Table 1-5. Boot address modes

EFBOOTLK	BOOT0	BOOT1	EFSB	Boot address	Boot area
-	0	-	0	0x08000000	SIP Flash
-	0	-	1	0x0BF46000	secure boot
0	1	0	-	0x0BF40000	Bootloader / ROM
0	1	1	-	0x20000000	SRAM
1	1	-	-	0x0BF40000	Bootloader / ROM

The BOOTx (x=0/1) value (either coming from the pin or the EFBOOTx bit) is latched upon reset release. It is up to the user to set BOOTx values to select the required boot mode. The BOOTx pin or EFBOOTx bit (depending on the EFBOOTLK and SWBOOTx bit value in the EFUSE_CTL0 register) is also re-sampled when exiting from Standby mode. Consequently, they must be kept in the required Boot mode configuration in Standby mode. After startup delay, the selection of the boot area is done before releasing the processor reset.

The embedded bootloader is located in the System memory, which is used to reprogram the Flash memory. The bootloader can be activated through one of the following serial interfaces:

USART0(PB15 and PA8), UART1(PA4 and PA5), UART2(PA6 and PA7).

1.5. System configuration registers (SYSCFG)

SYSCFG base address: 0x4001 3800

1.5.1. Configuration register 0 (SYSCFG_CFG0)

Address offset: 0x00

Reset value: 0x0000 000X (X indicates BOOT_MODE[1:0] may be any value according to the BOOT0 and BOOT1 pins)

This register 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														BOOT_MODE[1:0]	
														r	

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1:0	BOOT_MODE[1:0]	Boot mode (Refer to ***Boot configuration*** for details) Bit0 is mapping to the BOOT0 value, the value of bit1 is the opposite of the BOOT1_n option bit value. x0: Boot from Main Flash 01: Boot from the system 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.

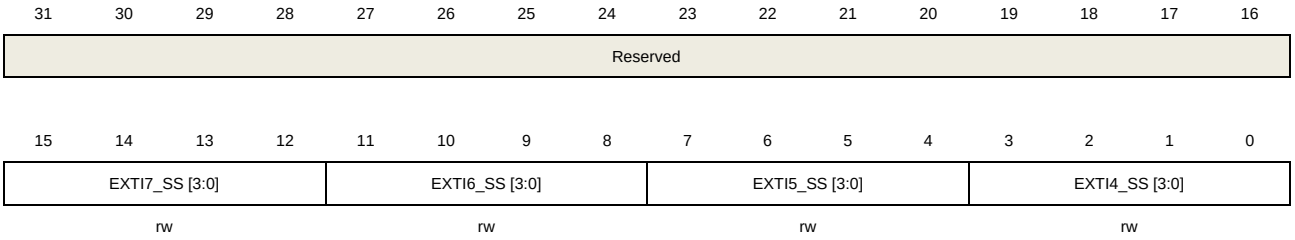
15:12	EXTI3_SS[3:0]	EXTI 3 sources selection 0000: PA3 pin 0001: PB3 pin Other configurations are reserved.
11:8	EXTI2_SS[3:0]	EXTI 2 sources selection 0000: PA2 pin 0001: PB2 pin Other configurations are reserved.
7:4	EXTI1_SS[3:0]	EXTI 1 sources selection 0000: PA1 pin 0001: PB1 pin Other configurations are reserved.
3:0	EXTI0_SS[3:0]	EXTI 0 sources selection 0000: PA0 pin 0001: PB0 pin Other configurations are reserved.

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 0000: PA7 pin Other configurations are reserved.
11:8	EXTI6_SS[3:0]	EXTI 6 sources selection 0000: PA6 pin Other configurations are reserved.
7:4	EXTI5_SS[3:0]	EXTI 5 sources selection 0000: PA5 pin

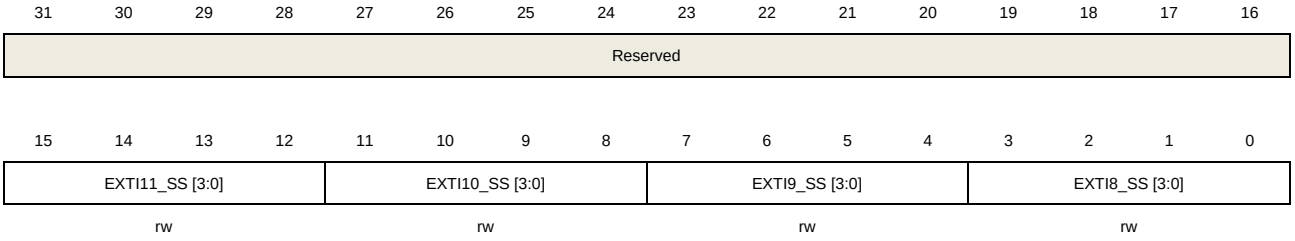
		Other configurations are reserved.
3:0	EXTI4_SS[3:0]	EXTI 4 sources selection 0000: PA4 pin 0001: PB4 pin Other configurations are reserved.

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 0000: PA11 pin 0001: PB11 pin Other configurations are reserved.
11:8	EXTI10_SS[3:0]	EXTI 10 sources selection 0000: PA10 pin Other configurations are reserved.
7:4	EXTI9_SS[3:0]	EXTI 9 sources selection 0000: PA9 pin Other configurations are reserved.
3:0	EXTI8_SS[3:0]	EXTI 8 sources selection 0000: PA8 pin 0001: PC8 pin Other configurations are reserved.

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 0000: PA15 pin 0001: PB15 pin 0010: PC15 pin Other configurations are reserved.
11:8	EXTI14_SS[3:0]	EXTI 14 sources selection 0000: PA14 pin 0001: PC14 pin Other configurations are reserved.
7:4	EXTI13_SS[3:0]	EXTI 13 sources selection 0000: PA13 pin 0001: PB13 pin 0010: PC13 pin Other configurations are reserved.
3:0	EXTI12_SS[3:0]	EXTI 12 sources selection 0000: PA12 pin 0001: PB12 pin Other configurations are reserved.

1.5.6. I/O compensation control register (SYSCFG_CPSCTL)

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							CPS_RDY	Reserved							CPS_EN
							r								rw

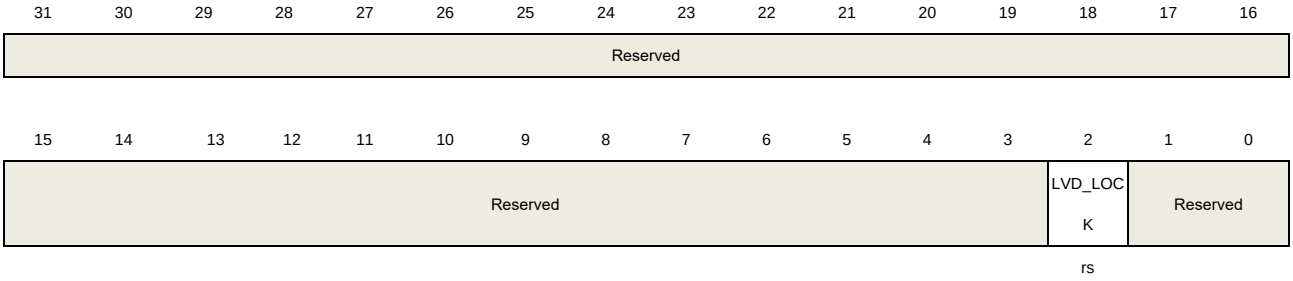
Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	CPS_RDY	Compensation cell ready flag 0: I/O compensation cell not ready 1: I/O compensation cell ready
7:1	Reserved	Must be kept at reset value
0	CPS_EN	Compensation cell power-down 0: I/O compensation cell power-down mode 1: I/O compensation cell enabled

1.5.7. SYSCFG configuration register 1 (SYSCFG_CFG1)

Address offset: 0x54

Reset value: 0x0000 0000

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



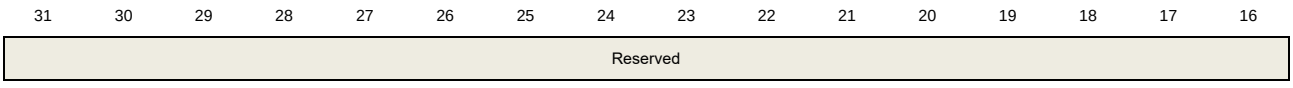
Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	LVD_LOCK	LVD lockup bit This bit is set by software and cleared only by a system reset. It can be used to lock the LVD connection to TIMER0/15/16 break input. 0: LVD interrupt disconnected from TIMER0/15/16 break input. 1: LVD interrupt connected to TIMER0/15/16 break input.
1:0	Reserved	Must be kept at reset value.

1.5.8. SYSCFG shared SRAM configuration register (SYSCFG_SCFG)

Address offset: 0x68

Reset value: 0x0000 0001

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															SOWNSEL
															rw

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	SOWNSEL	Shared SRAM ownership select This bit is used to control the ownership of the shared 32K SRAM. 0: Wireless 1: Core

1.5.9. TIMER trigger selection register (SYSCFG_TIMERxCFG)(x = 0..2)

Address offset: 0x100 + 0x4 * x
Reset value: 0x0000 0000

TSCFG0[3:0], TSCFG1[3:0]..TSCFG6[3:0] are mutually exclusive and cannot be configured at the same time. When TSCFG7[3:0] is used, ensure that TSCFGy[3:0](y = 0..6) is zero. Otherwise, the channel input source selected by TSCFGy[3:0] prevails.

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSCFG7[3:0]				TSCFG6[3:0]				TSCFG5[3:0]				TSCFG4[3:0]			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSCFG3[3:0]				TSCFG2[3:0]				TSCFG1[3:0]				TSCFG0[3:0]			
rw				rw				rw				rw			

Bits	Fields	Descriptions
31:28	TSCFG7[3:0]	ITS trigger source configuration ITS trigger source configuration, include channels' input signals (CIx) and internal trigger input (ITIx). 0000: Reserved 0001: Internal trigger input 0 (ITI0) 0010: Internal trigger input 1 (ITI1) 0011: Internal trigger input 2 (ITI2) 0100: Internal trigger input 3 (ITI3) 0101: CI0 edge flag (CI0F_ED) Others: Reserved These bits must not be changed when slave mode is enabled. **Note:** When TSCFG7[3:0] is used, TSCFGy[3:0](y = 0..6) should be zero, otherwise, the ITS trigger cooperate with TSCFGy[3:0], and input source depend on

TSCFGy[3:0].

27:24	TSCFG6[3:0]	External clock mode 0 configuration This bit-field specifies which signal is selected as the trigger input, which is used to synchronize the counter. 0000: External clock mode0 disable 0001: Internal trigger input 0 (ITI0) 0010: Internal trigger input 1 (ITI1) 0011: Internal trigger input 2 (ITI2) 0100: Internal trigger input 3 (ITI3) 0101: CI0 edge flag (CI0F_ED) 0110: channel 0 input Filtered output (CI0FE0) 0111: channel 1 input Filtered output (CI1FE1) 1000: External trigger input filter output(ETIFP) Others: Reserved These bits must not be changed when slave mode is enabled.
23:20	TSCFG5[3:0]	Event mode configuration This bit-field specifies which signal is selected as the trigger input, which is used to synchronize the counter. 0000: Event mode disable 0001: Internal trigger input 0 (ITI0) 0010: Internal trigger input 1 (ITI1) 0011: Internal trigger input 2 (ITI2) 0100: Internal trigger input 3 (ITI3) 0101: CI0 edge flag (CI0F_ED) 0110: channel 0 input Filtered output (CI0FE0) 0111: channel 1 input Filtered output (CI1FE1) 1000: External trigger input filter output(ETIFP) Others: Reserved These bits must not be changed when slave mode is enabled.
19:16	TSCFG4[3:0]	Pause mode configuration This bit-field specifies which signal is selected as the trigger input, which is used to synchronize the counter. 0000: Pause mode disable 0001: Internal trigger input 0 (ITI0) 0010: Internal trigger input 1 (ITI1) 0011: Internal trigger input 2 (ITI2) 0100: Internal trigger input 3 (ITI3) 0101: Reserved 0110: channel 0 input Filtered output (CI0FE0) 0111: channel 1 input Filtered output (CI1FE1) 1000: External trigger input filter output(ETIFP) Others: Reserved

		These bits must not be changed when slave mode is enabled.
15:12	TSCFG3[3:0]	Restart mode configuration This bit-field specifies which signal is selected as the trigger input, which is used to synchronize the counter. 0000: Restart mode disable 0001: Internal trigger input 0 (ITI0) 0010: Internal trigger input 1 (ITI1) 0011: Internal trigger input 2 (ITI2) 0100: Internal trigger input 3 (ITI3) 0101: CI0 edge flag (CI0F_ED) 0110: channel 0 input Filtered output (CI0FE0) 0111: channel 1 input Filtered output (CI1FE1) 1000: External trigger input filter output(ETIFP) Others: Reserved These bits must not be changed when slave mode is enabled.
11:8	TSCFG2[3:0]	Quadrature decoder mode 2 configuration 0000: Quadrature decoder mode 2 disable Others: The counter counts on both CI0FE0 and CI1FE1 edge, while the direction depends on each other. These bits must not be changed when slave mode is enabled.
7:4	TSCFG1[3:0]	Quadrature decoder mode 1 configuration 0000: Quadrature decoder mode 1 disable Others: The counter counts on CI1FE1 edge, while the direction depends on CI0FE0 level. These bits must not be changed when slave mode is enabled.
3:0	TSCFG0[3:0]	Quadrature decoder mode 0 configuration 0000: Quadrature decoder mode 0 disable Others: The counter counts on CI0FE0 edge, while the direction depends on CI1FE1 level. These bits must not be changed when slave mode is enabled.

2. Flash memory controller (FMC)

2.1. Overview

The flash memory controller, FMC, provides all the necessary functions for the on-chip Flash Memory. There also provide page erase, mass erase, and word program for flash memory.

2.2. Characteristics

- Two memory organizations:
  - Main : Up to 4MB (typical : 2MB) of on-chip flash memory for instruction and data.
  - Information block : 256K bytes bootloader
- The on-chip flash pagesize is 4KB.
- Supports RTDEC (decrypt by AES real-time) fuction.
- Word programming, page erase and mass erase operation.
- Flash security protection to prevent illegal code / data access.
- Page erase / program protection to prevent unexpected operation.

2.3. Function overview

2.3.1. Flash memory architecture

The flash memory consists of no more than 4MB main flash, which is organized into 1024 pages with 4KB capacity per page, and 256KB information block for the boot loader. Each page can be erased individually. The ***Table 2-1. Base address and size for flash memory*** shows the details of flash organization.

Table 2-1. Base address and size for flash memory

Block	Name	Address range	size(bytes)
Main flash block	Page 0	0x0800 0000 - 0x0800 0FFF	4KB
	Page 1	0x0800 1000 - 0x0800 1FFF	4KB
	Page 2	0x0800 2000 - 0x0800 2FFF	4KB
	. . .	. . .	. . .
	Page 1022	0x083F E000 - 0x083F EFFF	4KB
	Page 1023	0x083F F000 - 0x083F FFFF	4KB
Bootloader	normal	0x0BF4 0000 - 0x0BF4 5FFF	24KB
	secure boot	0x0BF4 6000 - 0x0BF4 DFFF	32KB
	secure ROM	0x0BF4 E000 - 0x0BF7 FFFF	200KB

Note: The bootloader block cannot be programmed or erased by user.

2.3.2. Read operations

The flash can be addressed directly as a common memory space.

RTDEC function

RTDEC function means that when reading data from flash, it can be decrypted in real time according to the EFUSE module's configuration of AES algorithm. (Data written to flash is encrypted already). There have on-time decrypt function when AESEN bit in EFUSE_USERCTL register is set. This is implements by hardware on-time and invisible by software. The AES key use AESKEY [127:0] bits in EFUSE_AESKEY to set.

Fetch data use SIP flash. Start address is 0x08000000. Size is 4MB. It can support RTDEC function.

NO-RTDEC function

Up to four areas can be configured without RTDEC function by FMC_NODECx (x=0,1,2,3), even if AESEN bit in EFUSE_USERCTL register is set.

Read add offset fuction

In order to meet the needs of Wi-Fi OTA fuction, the read offset function can be configured to increase the bus initial address by add an offset and then read it from FMC. After setting the read offset value and area by configuring FMC_OFVR and FMC_OFRG, reading the value of the source address is equivalent to reading the value of the add offset address. If user need to configure other functions, the RTDEC area configuration needs to use the source address.

Note: 1. Adding offset area does not support configure into NO-RTDEC area.
2. The offset function only supports read operation, not support program operation and erase operation.

2.3.3. Unlock the FMC_CTL register

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

The OBWEN bit in the FMC_CTL are still protected even the FMC_CTL is unlocked. The

unlocking sequence consists of two write operations, which are writing 0x45670123 and 0xCDEF89AB to the FMC_OBKEY register. Then the hardware sets the OBWEN bit in the FMC_CTL register to 1. The software can reset OBWEN bit to 0 to protect the FMC_NODECx (x=0,1,2,3) / FMC_OFRG / FMC_OFVR registers.

Note: 1. As long as the wrong key is written to FMC_CTL, a bus error will be generated.
2. Write wrong key to OBKEY does not generate a bus error.

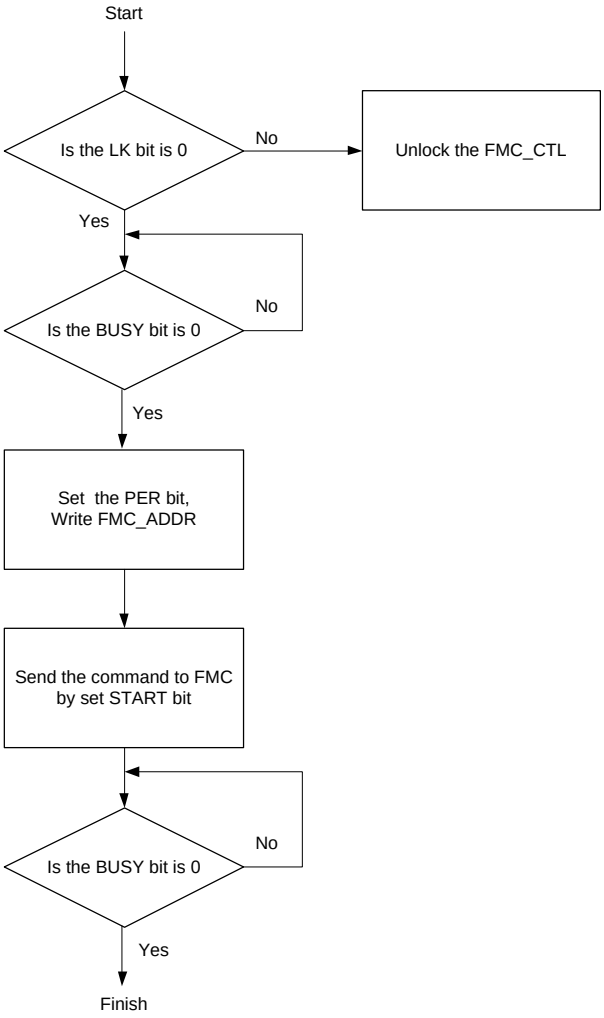
2.3.4. Page erase

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

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

When the operation is executed successfully, the ENDF bit in the FMC_STAT register is set, and an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set. Note that a correct target page address must be confirmed. Otherwise, the software may run out of control if the target erase page is being used to fetch codes or access data. The FMC will not provide any notification when that happens. Additionally, the page erase operation will be ignored on erase / program protected pages. In this condition, a flash operation error interrupt will be triggered by the FMC if the ENDIE bit in the FMC_CTL register is set. The software can check the WPERR bit in the FMC_STAT register to detect this condition in the interrupt handler. The ***Figure 2-1. Process of page erase operation*** shows the page erase operation flow.

Figure 2-1. Process of page erase operation



2.3.5. Mass erase

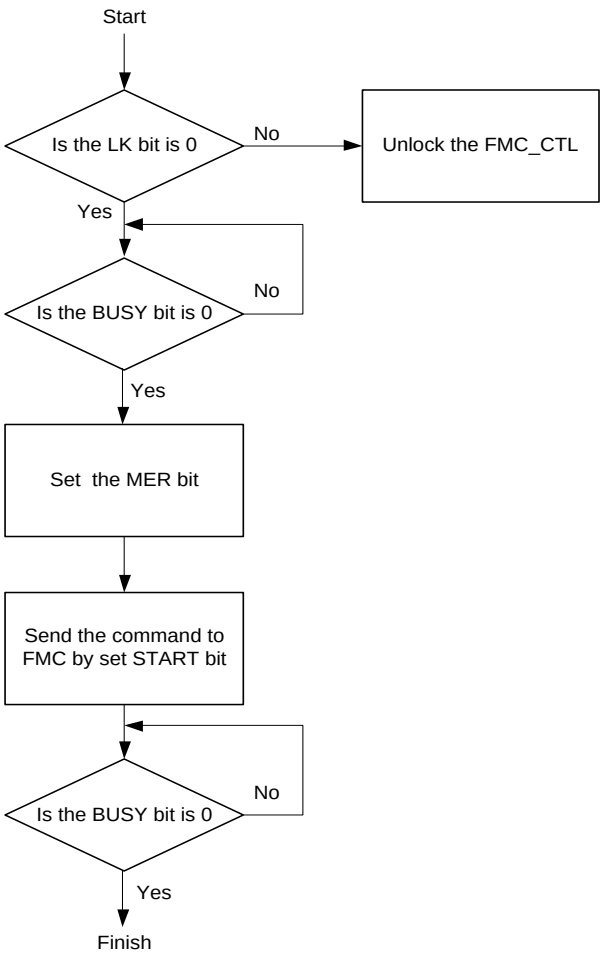
The FMC provides a complete erase function which is used to initialize the main flash block contents. This erase can affect entire flash block by setting the MER bit to 1 in the FMC_CTL register. The following steps show the mass erase register access sequence.

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

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

The ***Figure 2-2. Process of mass erase operation*** indicates the mass erase operation flow.

Figure 2-2. Process of mass erase operation



2.3.6. Main flash programming

The FMC provides a 32-bit word programming function by SBUS which is used to modify the main flash memory contents.

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

- Unlock the FMC_CTL register if necessary.
- Check the BUSY bit in the FMC_STAT register to confirm that no flash memory operation is in progress (BUSY equals to 0). Otherwise, wait until the operation has finished.
- Set the PG bit in the FMC_CTL register.
- Write the data to be programed by SBUS with desired absolute address (0x08XX XXXX).
- Wait until all the operations have been finished by checking the value of the BUSY bit in the FMC_STAT register.
- Read and verify the flash memory using a SBUS access if required.

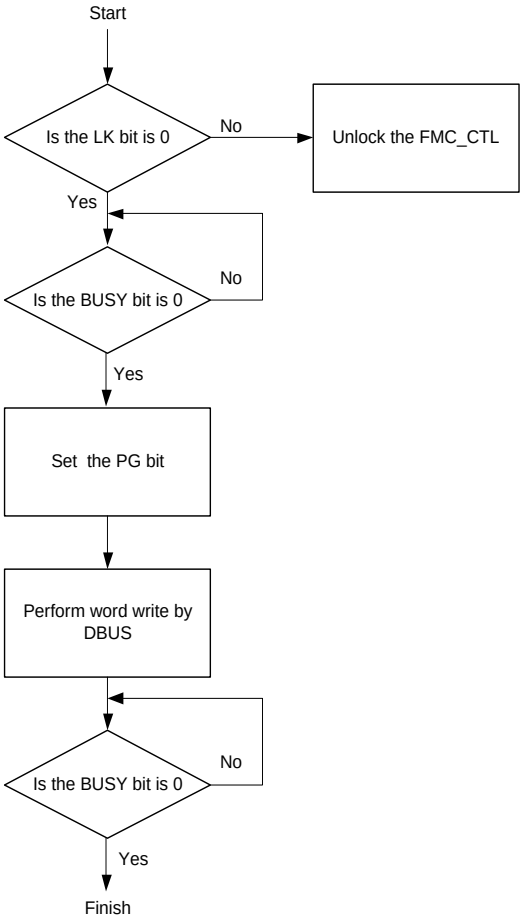
When the operation is executed successfully, the ENDF bit in the FMC_STAT register is set, and an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL register is set. Note that there are some program error need caution:

Each word can be programmed only one time after erase and before next erase. Note that the PG bit must be set before the word programming operation.

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

In 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 WPERR bit in the FMC_STAT register to detect which condition occurred in the interrupt handler. The ***Figure 2-3. Process of word program operation*** shows the word programming operation flow.

Figure 2-3. Process of word program operation



Note: 1. In order to meet Wi-Fi OTA requirements, support continuous program when set PG bit.

2. If user do not do read to check operation before program, it can realize the wired-AND of the data to be programmed and the data in flash.

3. In order to achieve the fastest programming speed, user need to follow the following points when programming. (1) 32-bit program and the address is continuous. (2) Program continuously on the bus, there must be no read flash between two write operation. (3) The interval between two write operations cannot exceed $64xT_{hclk}$ (flash clock is hclk/2).

2.3.7. Option bytes

Option bytes description

The option bytes description is shown in the ***Table 2-2. Option bytes***.The option bytes are configured according to the requirements of the application.

Table 2-2. Option bytes

Name	Register map
12：SRAM1_RST 11：NRST_DPSLP 10：NRST_STDBY 9：NWDG_HW [7:0]：SPC[7:0]	***Option byte register (FMC_OBR)***
[31:0]：USER[31:0]	***Option byte user value register (FMC_OBUSER)***
[25:16]：WRP0_EPAGE[9:0] [15:0]：WRP0_SPAGE[9:0]	***Option byte write protection area register 0 (FMC_OBWRP0)***
[25:16]：WRP1_EPAGE[9:0] [15:0]：WRP1_SPAGE[9:0]	***Option byte write protection area register 1 (FMC_OBWRP1)***

Option bytes modify

To modify the user options value, follow the procedure below:

- Unlock the FMC_CTL register if necessary.
- Check the BUSY bit in the FMC_STAT register to confirm that no flash memory operation is in progress (BUSY equals to 0)
- Unlock the option bytes operation bits in the FMC_CTL register if necessary.
- Wait until the OBWEN bit is set in the FMC_CTL register.
- Write the desired options value in the options registers.
- Set the OBSTART bit in the FMC_CTL register.
- Wait until all the operations have finished by checking the value of the BUSY bit in the FMC_STAT register.
- Set the OBRLD option bit or launch a system reset to start option bytes loading.

2.3.8. Security protection

Security protection

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

Protection level 0: no protection

When SPC[7:0] bits in FMC_OBR is set to 0xAA, after the system is reset, the flash memory will be in no protection state. The main flash are accessible by all operations. The SRAM1 and the backup registers are also accessible by all operations.

Protection level 1: read protection

When SPC[7:0] bits in FMC_OBR is set to to any value except 0xAA, after the system is reset,

the flash memory will be in protection level 1 state.

• **User mode:** Code executing in user mode (boot flash) can access flash main memory, SRAM1 and backup registers with all operations (read, erase and program).

• **Debug, boot RAM and boot loader modes**: In debug mode or when code is running from boot RAM or boot loader, the flash main memory, the backup registers and the SRAM1 are totally inaccessible. In these boot modes an intrusion is detected and a read or write access to the flash or SRAM1 generates a bus error and a core exception.

Table 2-3. Flash operation under different protection levels

Access type	Fetch	Read	Write	Page erase
No protection, protection level-1 no intrusion(1)	OK		no write protection pages: OK write protection pages: WI and WPERR flag set	
Protection level-1 with instruction(2)	Bus error			write invalid, WPERR flag set

Note: 1. Level 1 no intrusion = when booting from user flash and no debug access. 2. Level 1 with intrusion = when debug / boot RAM / boot bootloader access is detected.

Modify security protection level

It is easy to move from security protection level 0 to level 1 by changing the value of the SPC[7:0] in FMC_OBR to any value except for 0xAA.

When the SPC[7:0] is programmed to the value 0xAA to move from level 1 to level 0, a mass erase of the flash main memory is performed. The backup registers and all SRAMs are also erased.

2.3.9. Wirte protection

The FMC provides a write protection function to prevent erase / program function on the flash memory. The page erase or program will not be accepted by the FMC on protected pages. If the page erase or program command is sent to the FMC on a protected page, the WPERR bit in the FMC_STAT register will be set by the FMC. If the WPERR bit is set and the ERRIE bit is also set to 1 to enable the corresponding interrupt, then the flash operation error interrupt will be triggered by the FMC to draw the attention of the CPU. The write-protected area defined in the option bytes (FMC_OBR) is also write-protected.

2.3.10. FLASH interrupts

Interrupts: end of operation / operation error.

Table 2-4. Flash interrupt requests

Flag	Description	Clear method	Interrupt enable bit
ENDF	end of operation	Write 1 to corresponding bit in FMC_STAT register	ENDIE
WPERR	erase / program on protected pages		ERRIE

2.4. Register definition

FMC base address: 0x4002 2000

2.4.1. Unlock key register (FMC_KEY)

Address offset: 0x04

Reset value: 0x0000 0000

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

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

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

2.4.2. Option byte unlock key register (FMC_OBKEY)

Address offset: 0x08

Reset value: 0x0000 0000

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

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

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

2.4.3. Status register (FMC_STAT)

Address offset: 0x0C

Reset value: 0x0000 0000

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

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

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

Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5	ENDF	End of operation flag bit When the operation executed successfully, this bit is set by hardware. The software can clear it by writing 1.
4	WPERR	Erase / Program protection error flag bit When erase / program on protected pages, this bit is set by hardware. The software can clear it by writing 1.
3:1	Reserved	Must be kept at reset value.
0	BUSY	The flash is busy bit When the operation is in progress, this bit is set to 1. When the operation is end or an error is generated, this bit is cleared to 0.

2.4.4. Control register (FMC_CTL)

Address offset: 0x10

Reset value: 0x0000 0080

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

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	OBRLD	Option byte reload bit This bit is set by software 0: No effect 1: force option byte reload.
14	OBSTART	Option bytes modification start bit

		0: No effect 1: Trigger an option bytes operation. This bit can only be written if OBWEN bit is set. This bit is only set by software, and is cleared when the BUSY bit is cleared in FMC_STAT.
13	Reserved	Must be kept at reset value.
12	ENDIE	End of operation interrupt enable bit This bit is set or cleared by software 0: No interrupt generated by hardware. 1: End of operation interrupt enable
11	Reserved	Must be kept at reset value.
10	ERRIE	Error interrupt enable bit This bit is set or cleared by software 0: No interrupt generated by hardware. 1: Error interrupt enable
9	OBWEN	FMC_OBR / FMC_OBUSER / FMC_OBWRPx(x=0,1) / FMC_NODECx(x = 0…3) / FMC_OFRG / FMC_OFVR write 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 sequence written to the FMC_KEY register. This bit can be set by software.
6	START	Send erase command to FMC This bit is set by software to send erase command to FMC. This bit is cleared by hardware when the BUSY bit is cleared.
5:4	Reserved	Must be kept at reset value.
3	WTPG	WIFI trim program command bit This bit is set or clear by software 0: no effect 1: WIFI trim program command
2	MER	Main flash mass erase command bit This bit is set or cleared by software 0: No effect 1: Main flash mass erase command
1	PER	Main flash page erase command bit This bit is set or clear by software 0: No effect

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

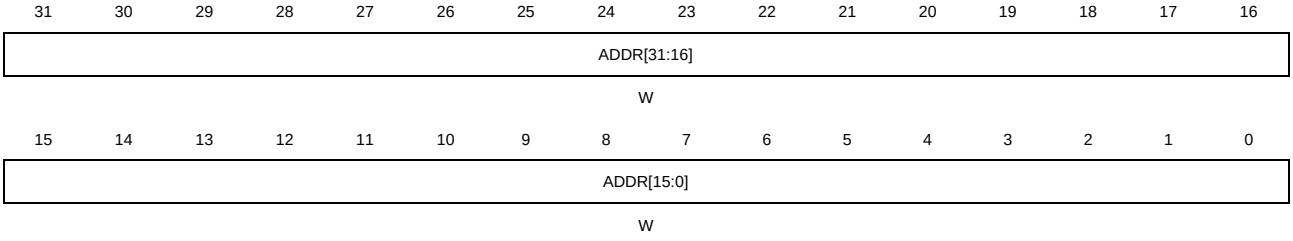
Note: This register should be reset after the corresponding flash operation completed.

2.4.5. Address register (FMC_ADDR)

Address offset: 0x14

Reset value: 0x0000 0000

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



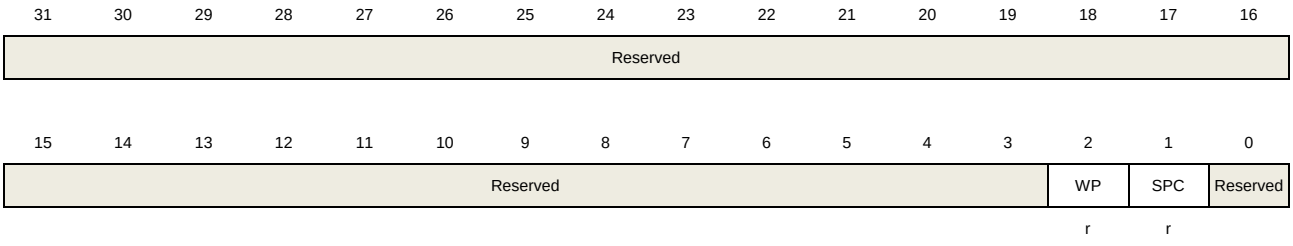
Bits	Fields	Descriptions
31:0	ADDR[31:0]	Flash erase / program command address bits These bits are configured by software. ADDR bits are the address of flash to be erased / programmed.

2.4.6. Option byte status register (FMC_OBSTAT)

Address offset: 0x1C

Reset value: 0x0XXX XXXX.

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



Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	WP	EFUSE config 32k write protection state 0: write protection is reset

		1: write protection is set
1	SPC	Security protection level 1 state 0: Protection level 1 is reset 1: Protection level 1 is set
0	Reserved	Must be kept at reset value.

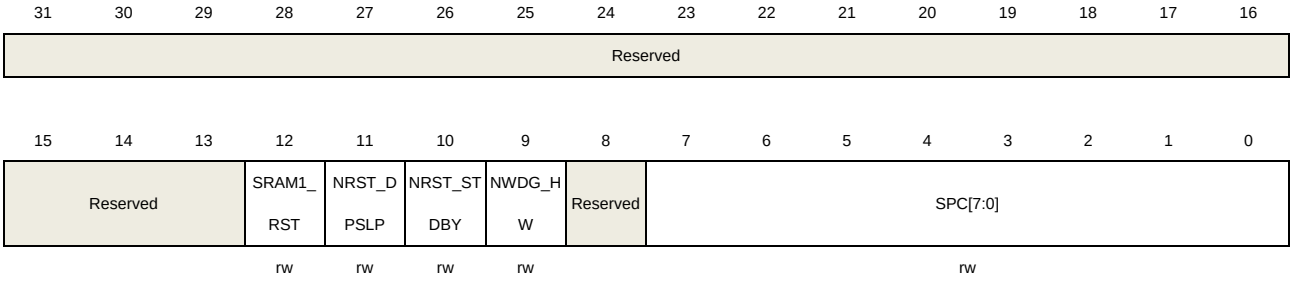
2.4.7. Option byte register (FMC_OBR)

Address offset: 0x40

Reset value: 0xXXXX XXXX (Register bits 0 to 31 are loaded with values from flash memory when OBRLD is set or system reset. The loading condition of the SRAM1_RST bit must be power-on reset.)

This register can not be written if OBWEN bit is set.

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



Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12	SRAM1_RST	SRAM1 reset enable bit 0: No effect 1: Clear SRAM1 data automatically. Effective after system reset.
11	NRST_ DPSLP	Deepsleep entry reset option status bit 0: A reset is generated when entering Deepsleep mode 1: No reset generated when entering Deepsleep mode
10	NRST_STDBY	Standby entry reset option status bit 0: A reset is generated when entering Standby mode 1: No reset generated when entering Standby mode
9	NWDG_HW	Watchdog status bit 0: Watchdog is hardware watchdog 1: Watchdog is software watchdog.
8	Reserved	Must be kept at reset value.
7:0	SPC[7:0]	Option byte security protection value. Effective after system reset.

Note: The security protection of the flash memory is subject to this bits field.

2.4.8. Option byte user value register (FMC_OBUSER)

Address offset: 0x44
Reset value: 0xXXXX XXXX (Register bits 0 to 31 are loaded with values from flash memory when OBRLD is set or system reset.)

This register can not be written if OBWEN bit is set.

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

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

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

Bits	Fields	Descriptions
31:0	USER[31:0]	Option byte USER value.

2.4.9. Option byte write protection area register 0 (FMC_OBWRP0)

Address offset: 0x48
Reset value: 0xXXXX XXXX (Register bits 0 to 31 are loaded with values from flash memory when OBRLD is set or system reset.)

This register can not be written if OBWEN bit is set.

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						WRP0_EPAGE[9:0]									
						rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						WRP0_SPAGE[9:0]									
						rw									

Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25:16	WRP0_EPAGE[9:0]	End page of write protection area 0
15:10	Reserved	Must be kept at reset value.
9:0	WRP0_SPAGE[9:0]	Start page of write protection area 0

2.4.10. Option byte write protection area register 1 (FMC_OBWRP1)

Address offset: 0x4C
Reset value: 0xXXXX XXXX (Register bits 0 to 31 are loaded with values from flash memory when OBRLD is set or system reset.)

This register can not be written if OBWEN bit is set.

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						WRP1_EPAGE[9:0]									
						rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						WRP1_SPAGE[9:0]									
						rw									

Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25:16	WRP1_EPAGE[9:0]	End page of write protection area 1
15:10	Reserved	Must be kept at reset value.
9:0	WRP1_SPAGE[9:0]	Start page of write protection area 1

2.4.11. NO RTDEC region register x (FMC_NODECx)(x = 0…3)

Address offset: 0x70 + 0x4 * x
Reset value: 0x0000 03FF

This register can not be written if OBWEN bit is set.

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						NODECx_EPAGE[9:0]									
						rw									

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

Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25:16	NODECx_EPAGE[9:0]	End page of NODEC region x (x=0,1,2,3).
15:10	Reserved	Must be kept at reset value.

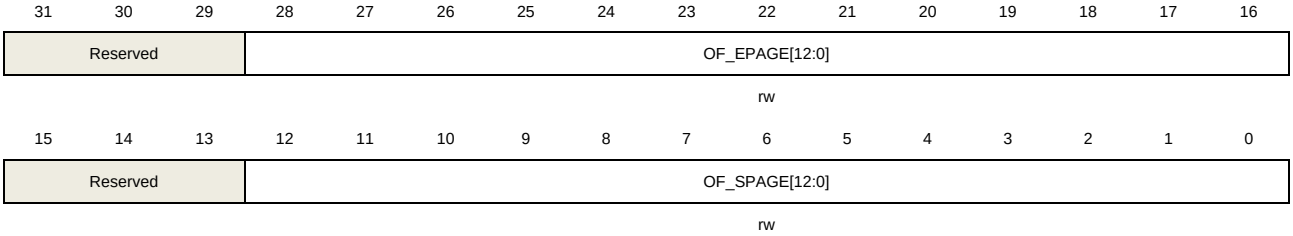
9:0	NODECx_SPAGE[9:0]	Start page of NODEC region x (x=0,1,2,3).

2.4.12. Offset region register (FMC_OFRG)

Address offset: 0x80
Reset value: 0x0000 1FFF

This register can not be written if OBWEN bit is set.

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



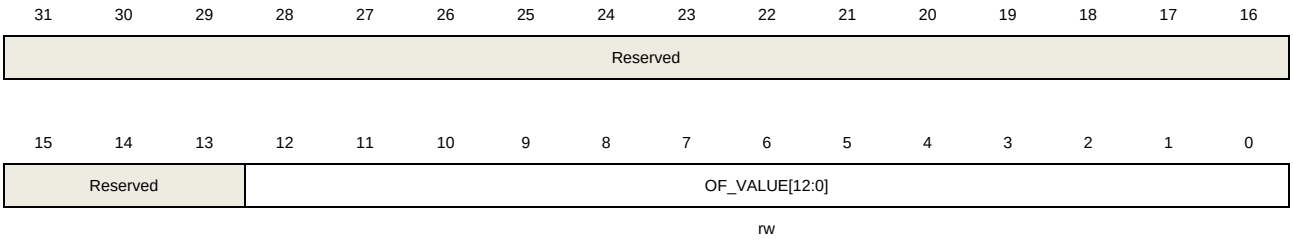
Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28:16	OF_EPAGE[12:0]	End page of offset region.
15:13	Reserved	Must be kept at reset value.
12:0	OF_SPAGE[12:0]	Start page of offset region

2.4.13. Offset value register (FMC_OFVR)

Address offset: 0x84
Reset value: 0x0000 0000

This register can not be written if OBWEN bit is set.

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



Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:0	OF_VALUE[12:0]	Offset value

2.4.14. Product ID0 register (FMC_PID0)

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
PID0[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PID0[15:0]															
r															

Bits	Fields	Descriptions
31:0	PID0[31:0]	Product reserved ID code register These bits are read only by software. These bits are unchanged constant after power on. These bits are one time program when the chip produced.

2.4.15. Product ID1 register (FMC_PID1)

Address offset: 0x104

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
Reserved															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PID1[15:0]															
r															

Bits	Field	Descriptions
31:0	PID1[15:0]	Product reserved ID code register These bits are read only by software. These bits are unchanged constant after power on. These bits are one time program when the chip produced.

2.4.16. RF Trim register 0 (FMC_RFT0)

Address offset: 0x108

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
BLETXCAL[7:0]								WIFITXCAL[7:0]							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THECAL[7:0]								PABIAST1[3:0]				PABIAST0[3:0]			
r								r				r			

Bits	Field	Descriptions
31:24	BLETXCAL[7:0]	BLE transmit power calibration value
23:16	WIFITXCAL[7:0]	WIFI transmit power calibration value
15:8	THECAL[7:0]	Thermal meter calibration value
7:4	PABIAST1[3:0]	The PA bias fine tune value.
3:0	PABIAST0[3:0]	The PA(Power Amplifier) bias coarse tune value.

2.4.17. RF Trim register 1 (FMC_RFT1)

Address offset: 0x10C
Reset value: 0x0000 00XX

This register 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								EFUSEID[3:0]				WIFIRXGCAL[3:0]			
								r				r			

Bits	Field	Descriptions
31:8	Reserved	Must be kept at reset value.
7:4	EFUSEID[3:0]	The EFUSE version ID.
3:0	WIFIRXGCAL[3:0]	The WIFI receive gain calibration value.

2.4.18. WIFI Trim register x (FMC_WFTx)(x = 0…15)

Address offset: 0x200 + 0x4 * x
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
WIFI_TRIM[31:16]															
rw															

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

Bits	Field	Descriptions
31:0	WIFI_TRIM[31:0]	After the system is reset, it is loaded from the flash. After the WTPG is set to 1, it can be program, and the corresponding flash cannot be erased. In the register program process, the WTPG bit is set to 1. Until the BUSY bit is 0, it indicates the end of programing. These bits are one time program when the chip produced, and each program a register (32bit). Update the value of program after system reset.

3. Electronic fuse (EFUSE)

3.1. Overview

The Efuse controller has Efuse macro that store system paramters. As a non-volatile unit of storage, the bit of Efuse macro cannot be restored to 0 once it is programmed to 1. According to the software opration, the Efuse controller can program all bits in the system parameters.

3.2. Characteristics

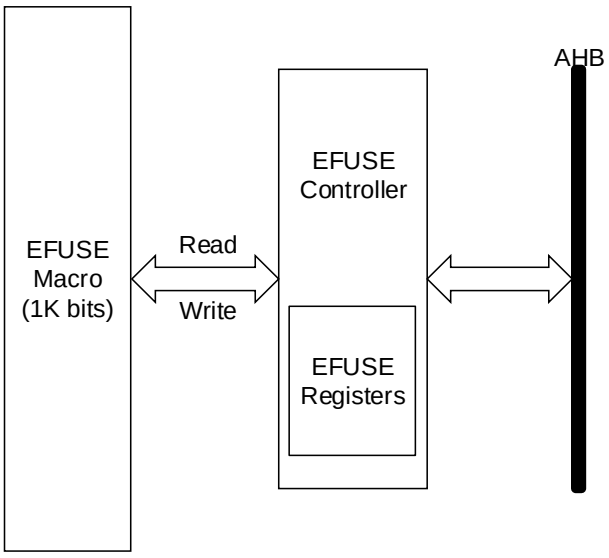
- One-time programmable nonvolatile Efuse storage cells organized as 128*8 bit.
- All bits in the Efuse cannot be rollback from 1 to 0.
- Can only be accessed through corresponding register.

3.3. Function overview

3.3.1. Block diagram

Efuse controller implements the Efuse macro read-program control logic. Efuse controller contains Efuse_1Kb module that instantiates Efuse macro.

Figure 3-1. Block diagram of Efuse controller



3.3.2. Efuse architecture

The Efuse consists of up to 1024 bits storage cells organized into 128 bytes. Efuse uses 7-bit address encoding. The following table ***Table 3-1. Efuse address mapping*** shows the address.

Table 3-1. Efuse address mapping

ADDR [6:0]	Efuse byte
000_0000	Efuse[0]
000_0001	Efuse[1]
000_0010	Efuse[2]
...	...
111_1111	Efuse[127]

3.3.3. Efuse macro description

The Efuse macro stores 10 system parameters, every system parameter has different width. The following table ***Table 3-2. System parameters*** shows the details of each Efuse byte.

Table 3-2. System parameters

Parameter	Width /B	Start(1) address	Program-protected	Read-protected	Description	Note
Efuse control 0	1B	7'd0	Can write multiple times, but can not rollback	Read out after system reset and keep unchanged, bus readable	Control bytes of relevant parameters required for MCU startup. For more details, refer to ***Control register 0 (EFUSE_CTL0)***	User defined
Efuse control 1	1B	7'd1	Can write multiple times, but can not rollback	Read out after system reset and keep unchanged, bus readable	For more details, refer to ***Control register 1 (EFUSE_CTL1***)	User defined
Flash protection	1B	7'd2	Can write multiple times, but can not rollback	Read out after system reset and keep unchanged, bus readable	The option of flash protection. For more details, refer to ***Flash protection control register (EFUSE_FPCTL)***	User defined
User control	1B	7'd3	Can write multiple times, but can not rollback	Read out after system reset and keep unchanged, bus readable	The option of user control function For more details, refer to ***User byte control register (EFUSE_USERCTL)***	User defined
reserved	12B	7'd4	Can write	Reserve for	For more details, refer to	User

Parameter	Width /B	Start(1) address	Program-protected	Read-protected	Description	Note
			multiple times, but can not rollback	further use	***EFUSE reserved register x (EFUSE_RESx)***	defined
AES key	16B	7'd16	Once-time	Read out after system reset and keep unchanged, bus readable	The AES key used to encrypt the firmware image For more details, refer to ***Firmware AES key register x (EFUSE_AESKEYx)***	User defined
RoTPK or its hash	32B	7'd32	Once-time	Only can be read out by ROM after system reset	Root of Trust Public Key（ECC's public key/the HASH result of the RSA's Public Key） For more details, refer to ***RoTPK key register x (EFUSE_ROTPKKEYx)***	User defined
MCU UID	16B	7'd64	Can not modify	Read out after system reset and keep unchanged, bus readable	The Unique ID of MCU For more details, refer to ***Product UID register x (EFUSE_PUIDx)***	CP
HUK	16B	7'd80	Can not modify	Read out after system reset and keep unchanged, bus readable	Hardware Unique Key, which provides the RoT (Root of Trust) for confidentiality. For more details, refer to ***HUK key register x (EFUSE_HUKKEYx)***	CP
User data	32B	7'd 96	Write once after system reset	Read out after system reset, bus readable	User defined data For more details, refer to ***User data register x (EFUSE_USER_DATAx)***	User defined

Note:

(1) 7-bit address encoding.

(2) System parameters must be read by its size. And it is also recommended to write according to its size too.

3.3.4. Read operation

The value of the Efuse can only be accessed through the corresponding register. After system reset, the Efuse value take effect and reloaded to corresponding register. The following steps show the register access sequence of the Efuse reading operation.

1. Clear the RDIF bit in EFUSE_CS register if it is set, and make sure there is no overstep boundary error.
2. Reset the EFRW bit in EFUSE_CS register.
3. Write the desired Efuse address and size to EFUSE_ADDR register.
4. Set the EFSTR bit EFUSE_CS register.
5. Wait until the reading operation has been finished by checking the RDIF bit in EFUSE_CS register.
6. Read the register value corresponding to the Efuse.

When the read operation is executed successfully, the RDIF in EFUSE_CS register is set, and an interrupt will be triggered by EFUSE if the RDIE bit in the EFUSE_CS register is set.

3.3.5. Program operation

The value of the Efuse can only be modified through the corresponding register. The following steps show the register access sequence of the Efuse writing operation.

1. Clear the PGIF bit in EFUSE_CS register if it is set, and make sure there are no overstep boundary error.
2. SET the EFRW bit in EFUSE_CS register.
3. Write the desired Efuse address and size to EFUSE_ADDR register.
4. Write the data to the corresponding register.
5. Set the EFSTR bit EFUSE_CS register.
6. Wait until the writing operation has been finished by checking the PGIF bit in EFUSE_CS register.

When the write operation is executed successfully, the PGIF in EFUSE_CS register is set, and an interrupt will be triggered by Efuse if the PGIE bit in the EFUSE_CS register is set. It should be noted that the address and size of the written data must match the corresponding fuse register. If not match, OVBERIF bit in the EFUSE_CS register will be set, and an interrupt will be triggered by Efuse if the OVBERIE bit in the EFUSE_CS register is set.

3.4. Register definition

EFUSE base address: 0x4002 2800

3.4.1. Control and status register (EFUSE_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					OVBERIC	RDIC	PGIC	Reserved	OVBERIE	RDIE	PGIE	Reserved	OVBERIF	RDIF	PGIF
					rc_w1	rc_w1	rc_w1		rw	rw	rw		r	r	r

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

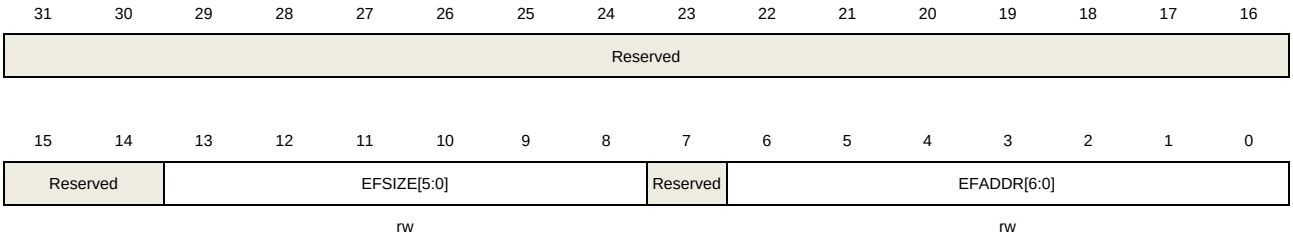
Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value.
26	OVBERIC	Clear bit for overstep boundary error interrupt flag 0: No effect 1: Clear error flag
25	RDIC	Clear bit for read operation completed interrupt flag 0: No effect 1: Clear read operation completed interrupt flag
24	PGIC	Clear bit for program operation completed interrupt flag 0: No effect 1: Clear program operation completed interrupt flag
23	Reserved	Must be kept at reset value.
22	OVBERIE	Enable bit for overstep boundary error interrupt 0: Disable the overstep boundary error interrupt 1: Enable the overstep boundary error interrupt
21	RDIE	Enable bit for read operation completed interrupt 0: Disable the read operation completed interrupt 1: Enable the read operation completed interrupt
20	PGIE	Enable bit for program operation completed interrupt 0: Disable the program operation completed interrupt 1: Enable the program operation completed interrupt
19	Reserved	Must be kept at reset value.

18	OVBERIF	Overstep boundary error flag 0: No overstep boundary error occurred 1: Overstep boundary error has occurred
17	RDIF	Read operation complete flag 0: Read Efuse operation not completed 1: Read Efuse operation completed
16	PGIF	Program operation completed flag 0: Program Efuse operation not completed 1: Program Efuse operation completed
15:2	Reserved	Must be kept at reset value.
1	EFRW	The selection of Efuse operation 0: Read Efuse 1: Write Efuse This bit cannot be modified when the EFSTR bit is 1
0	EFSTR	Start Efuse operation This bit is set by software and cleared by hardware 0: No effect 1: Start read or write Efuse operation

3.4.2. Address register (EFUSE_ADDR)

Address offset: 0x04
Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13:8	EFSIZE[5:0]	Read or write Efuse data size
7	Reserved	Must be kept at reset value.
6:0	EFADDR[6:0]	Read or write Efuse data start address

Note: This register cannot be modified when the EFSTR bit is 1.

3.4.3. Control register 0 (EFUSE_CTL0)

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										SWBOOT0	EFBOOT0	SWBOOT1	EFBOOT1	EFBOOTLK	EFSB
										rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5	SWBOOT0	Efuse BOOT0 bit enable 0: Select boot0_pad as BOOT0 output 1: Select Efuse_boot0 as BOOT0 output
4	EFBOOT0	Efuse BOOT0 0: Efuse_boot0 = 0 1: Efuse_boot0 = 1
3	SWBOOT1	Efuse BOOT1 bit enable 0: Select boot1_pad as BOOT1 output 1: Select Efuse_boot1 as BOOT1 output
2	EFBOOT1	Efuse BOOT1 0: Efuse_boot1 = 0 1: Efuse_boot1 = 1
1	EFBOOTLK	EFUSE_CTL0 register bits[5:2] lock bit 0: Unlock EFUSE_CTL register bits[5:2], these bits can be modify 1: Lock EFUSE_CTL register bits[5:2], these bits can not be modify
0	EFSB	Startup from secure boot This bit needs to work with the bits[5:1] to decide how to startup 0: Startup from Flash 1: Startup from secure boot

3.4.4. Control register 1 (EFUSE_CTL1)

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								VFCERT	VFIMG	Reserved			ROTLK	NDBG	Reserved
								rw	rw				rw	rw	

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	VFCERT	Verify firmware certificate 0: Disable firmware certificate verification 1: Enable firmware certificate verification
6	VFIMG	Verify firmware image 0: Disable firmware image verification 1: Enable firmware image verification
5:3	Reserved	Must be kept at reset value.
2	ROTLK	EFUSE_ROTPKKEY register lock bit 0: Unlock EFUSE_ROTPKKEY register, these bytes can be modified. 1: Lock EFUSE_ROTPKKEY register, these bytes can not be modified.
1	NDBG	Debugging permission setting 0: Unlimited debugging 1: Can not debug
0	Reserved	Must be kept at reset value.

3.4.5. Flash protection control register (EFUSE_FPCTL)

Address offset: 0x10

Reset value: 0x0000 00X0

This register 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													FP[2:0]		
													rw		

Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.

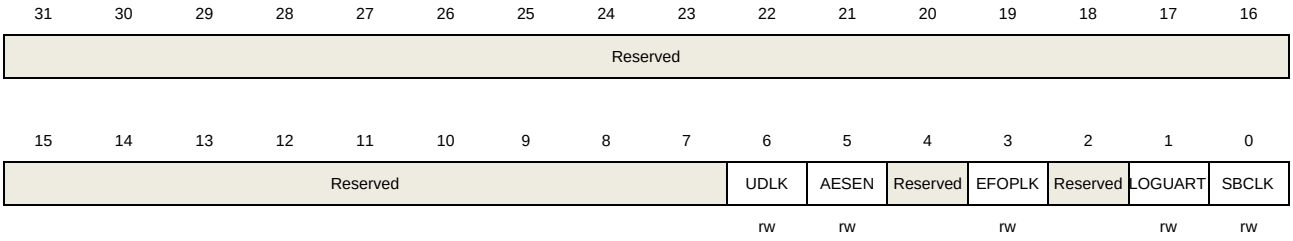
2:0	FP[2:0]	Efuse flash protection value Bit2: 0~32K write protection Bit1: Reserved Bit0: Read protection level 1

3.4.6. User byte control register (EFUSE_USERCTL)

Address offset: 0x14

Reset value: 0x0000 0006

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



Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	UDLK	EFUSE_USER_DATA register lock bit 0: Unlock EFUSE_USER_DATA register, and register can be modified. 1: Lock EFUSE_USER_DATA register, and register can not be modified.
5	AESEN	Lock EFUSE_AESKEY register and enable AES decrypt function 0: Disable AES decrypt and unlock EFUSE_AESKEY register and AES key can be modified. 1: Enable AES decrypt and lock EFUSE_AESKEY register and AES key can not be modified.
4	Reserved	Must be kept at reset value.
3	EFOPLK	EFUSE_FPCTL and EFUSE_USERCTL register lock bit 0: Unlock EFUSE_FPCTL and EFUSE_USERCTL register, these bytes can be modified. 1: Lock EFUSE_FPCTL and EFUSE_USERCTL register, these bytes can not be modified.
2	Reserved	Must be kept at reset value.
1	LOGUART	Secure Boot Log UART selection. 0: UART2 (PA6/PA7) 1: UART1 (PA4/PA5)
0	SBCLK	Secure boot clock source selection. 0: IRC16M

1: External HXTAL

3.4.7. EFUSE reserved register x (EFUSE_RESx) (x = 0…2)

Address offset: 0x18 + 0x4 * 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
RES[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES[15:0]															
rw															

Bits	Fields	Descriptions
31:0	RES[31:0]	Efuse reserved bytes.

3.4.8. Firmware AES key register x (EFUSE_AESKEYx) (x = 0…3)

Address offset: 0x24 + 0x4 * 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
AESKEY[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AESKEY[15:0]															
rw															

Bits	Fields	Descriptions
31:0	AESKEY[31:0]	Efuse AES key value.

3.4.9. RoTPK key register x (EFUSE_ROTPKKEYx) (x = 0…7)

Address offset: 0x34 + 0x4 * 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
RKEY[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

RKEY[15:0]
rw

Bits	Fields	Descriptions
31:0	RKEY[31:0]	Efuse RoTPK or its HASH value.

3.4.10. Product UID register x (EFUSE_PUIDx) (x = 0…3)

Address offset: 0x54 + 0x4 * 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
UID[31:16]															
r															

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

Bits	Fields	Descriptions
31:0	UID[31:0]	Efuse MCU UID value.

3.4.11. HUK key register x (EFUSE_HUKKEYx) (x = 0…3)

Address offset: 0x64 + 0x4 * 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
HKEY[31:16]															
r															

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

Bits	Fields	Descriptions
31:0	HKEY[31:0]	Efuse HUK key value.

3.4.12. User data register x (EFUSE_USER_DATAx) (x = 0…7)

Address offset: 0x74 + 0x4 * 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
USERDATA[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USERDATA[15:0]															
rw															

Bits	Fields	Descriptions
31:0	USERDATA[31:0]	Efuse USER_DATA value.

3.4.13. Boot address register (EFUSE_BOOTADDR)

Address offset: 0x124

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

Bits	Fields	Descriptions
31:0	BOOTADDR[31:0]	Boot from the address.

4. Power management unit (PMU)

4.1. Overview

The power consumption is regarded as one of the most important issues for the devices of GD32VW55x series. According to the power management unit (PMU), provides six types of power saving modes, including Sleep, Deep-sleep, Standby, SRAM_sleep, Wi-Fi_sleep and BLE_sleep 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 GD32VW55x devices, there are three power domains, including V_{DD} / V_{DDA} domain, 1.1V domain and backup domain, as is shown in the ***Figure 4-1. 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 1.1V domain power. Backup domain is powered from the main V_{DD} supply.

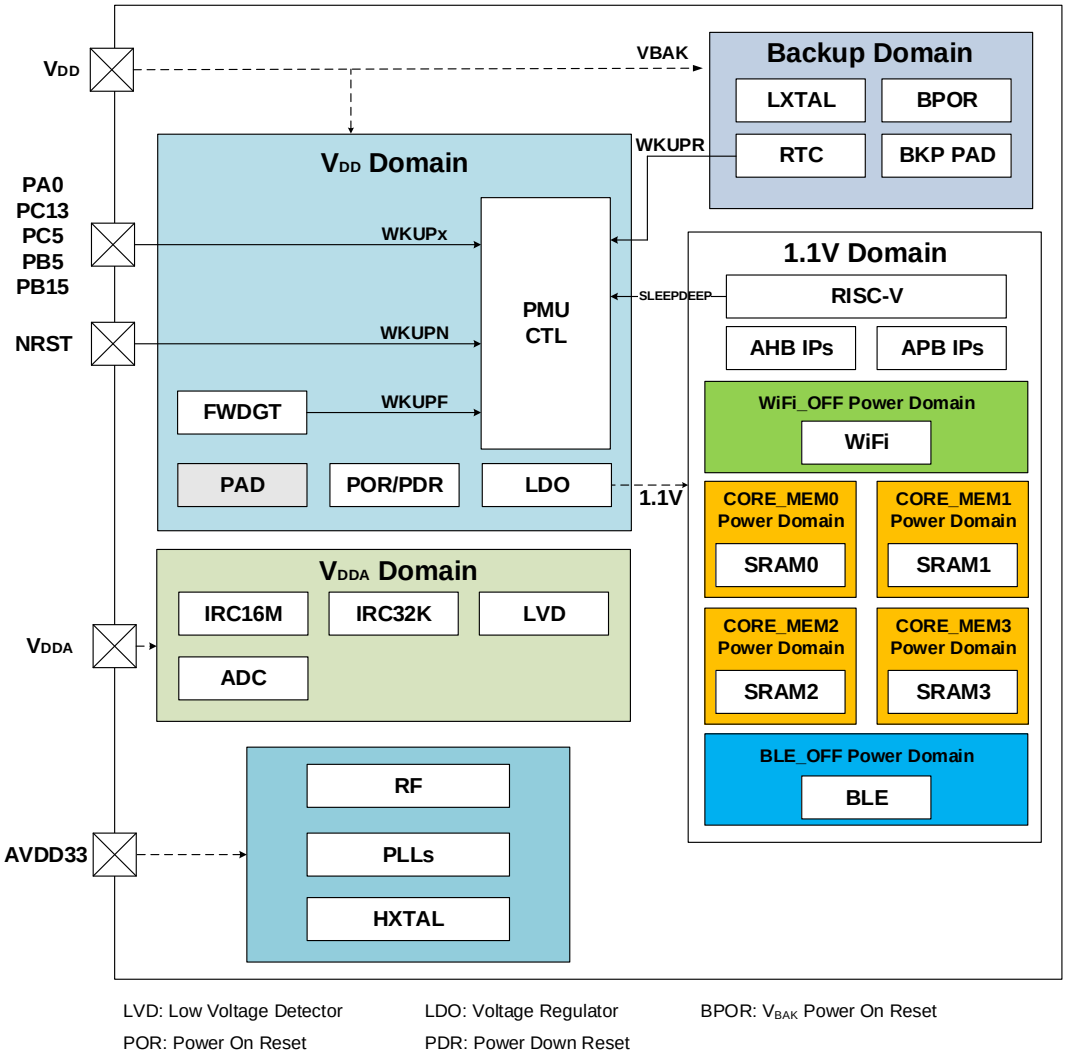
4.2. Characteristics

- Three power domains: V_{BAK}, V_{DD} / V_{DDA} and 1.1V power domains.
- Six power saving modes: Sleep, Deep-sleep, Standby, SRAM_sleep, Wi-Fi_sleep and BLE_sleep modes.
- Internal Voltage regulator (LDO) supplies around 1.1V voltage source for 1.1V domain.
- V_{DD} Low Voltage Detector(LVD) can issue an interrupt or event when the power is lower than a programmed threshold.
- LDO output voltage select for power saving.
- Ultra power saving for low-driver mode in Deep-sleep mode.
- SRAM0/SRAM1/SRAM2/SRAM3 can be power-off.
- Wi-Fi_off domain can be power-off.
- BLE_off domain can be power-off.

4.3. Function overview

Figure 4-1. Power supply overview provides details on the internal configuration of the PMU and the relevant power domains.

Figure 4-1. Power supply overview



LVD: Low Voltage Detector
LDO: Voltage Regulator
BPOR: V_{BAK} Power On Reset
POR: Power On Reset
PDR: Power Down Reset

4.3.1. Backup domain

The Backup domain is powered by the V_{DD}, and the V_{BAK} pin which drives Backup Domain, power supply for RTC unit, LXTAL oscillator, BPOR，and three pads, including PC13 to PC15.

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

The clock source of the Real Time Clock (RTC) circuit can be derived from the Internal 32KHz RC oscillator (IRC32K) or the Low Speed Crystal oscillator (LXTAL), or HXTAL clock divided by 1-32. Before entering the power saving mode by executing the WFI / WFE instruction, the RISC-V can setup the RTC register with an expected alarm time and enable the alarm function to achieve the RTC alarm event. After entering the power saving mode for a certain amount of time, the RTC alarm will wake up the device when the time match event occurs. The details of the RTC configuration and operation will be described in the ***Real time clock (RTC)***.

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

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

Note: Since PC13, PC14, PC15 can only be obtained by a small current, the speed of GPIOs PC13 to PC15 should not exceed 2MHz when they are in output mode (maximum load: 30pF).

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

V_{DD} / V_{DDA} domain includes two parts: V_{DD} domain and V_{DDA} domain.

V_{DD} domain includes HXTAL (high speed crystal oscillator), LDO (voltage regulator), POR / PDR (power on / down reset), FWDGT (free watchdog timer), all pads, etc.

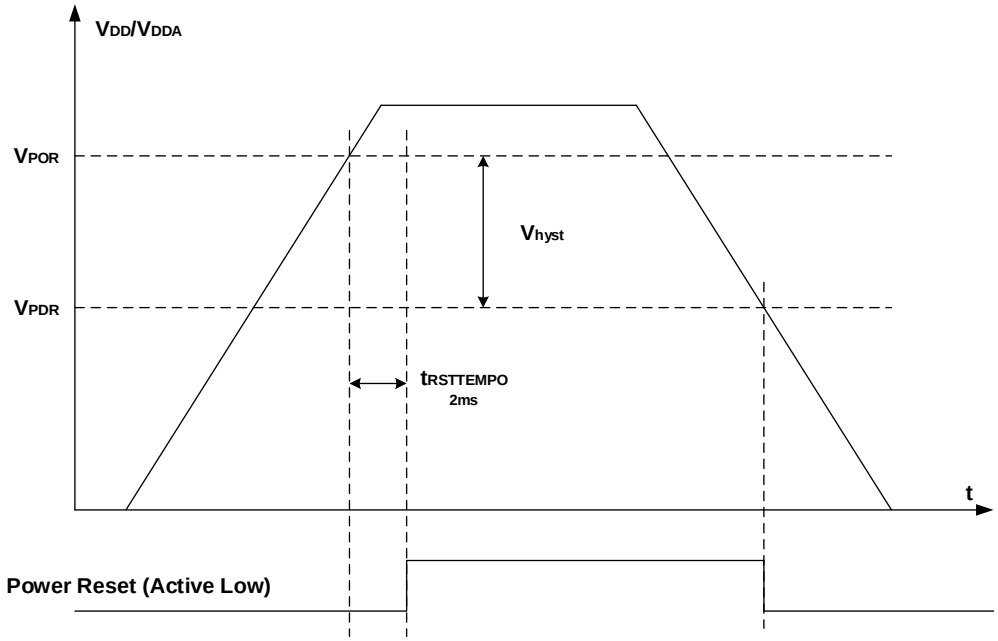
V_{DDA} domain includes ADC (AD converter), IRC16M (internal 16MHz RC oscillator), IRC32K (internal 32KHz RC oscillator), PLLs (phase locking loop), LVD (low voltage detector), etc.

V_{DD} domain

The LDO, which is implemented to supply power for the 1.1V domain, is always enabled after reset. It can be configured to operate in three different status, including sleep mode (full power on), deep-sleep mode (on or low power), and in the standby mode (power off).

The POR / PDR circuit is implemented to detect V_{DD} / V_{DDA} and generate the power reset signal which resets the whole chip except backup power domain when the supply voltage is lower than the specified threshold. ***Figure 4-2. Waveform of the POR / PDR*** shows the relationship between the supply voltage and the power reset signal. V_{POR}, which typical value is refer to GD32VW55x datasheet, indicates the threshold of power on reset, while V_{PDR}, which typical value is refer to GD32VW55x datasheet, means the threshold of power down reset. The hysteresis voltage (V_{hyst}) is refer to GD32VW55x datasheet.

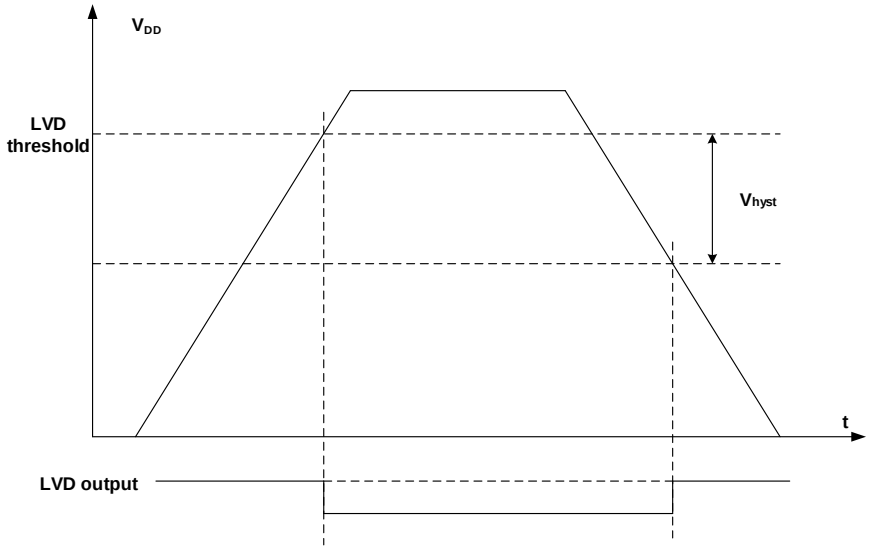
Figure 4-2. Waveform of the POR / PDR



V_{DDA} domain

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 PMU_CTL0 register. The LVD is enabled by setting the LVDEN bit, and LVDF bit, which in the PMU_CS0 register, indicates if V_{DD} / V_{DDA} is higher or lower than the LVD threshold. This event is internally connected to the EXTI line 16 and can generate an interrupt if it is enabled through the EXTI registers. ***Figure 4-3. 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. The hysteresis voltage (V_{hyst}) is refer to GD32VW55x datasheet.

Figure 4-3. Waveform of the LVD threshold



Generally, digital circuits are powered by V_{DD}, while most of analog circuits are powered by V_{DDA}. To improve the ADC conversion accuracy, the independent power supply V_{DDA} is implemented to achieve better performance of analog circuits. V_{DDA} can be externally connected to V_{DD} through the external filtering circuit that avoids noise on V_{DDA}, and V_{SSA} should be connected to V_{SS} through the specific circuit independently. Otherwise, when the V_{DD} and V_{DDA} are provided by different power supplies, the difference between V_{DD} and V_{DDA} during power-up and running time should not exceed 0.3V.

4.3.3. 1.1V power domain

The main functions that include RISC-V logic, AHB/APB peripherals, the APB interfaces for the Backup domain and the V_{DD}/V_{DDA} domain, SRAM_OFF power domain, BLE_OFF power domain, Wi-Fi_OFF power domain, etc, are located in this power domain. Once the 1.1V is powered up, the POR will generate a reset sequence on the 1.1V 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.

Wi-Fi_OFF domain

Refer to Wi-Fi spec, the typical work state is as follows:

1. Power off when standby mode.
2. Power on when cotrolled by register in run / sleep / deepsleep mode.
3. Power off (default) when controlled by register in run / sleep / deepsleep mode.

CORE_MEM0/1/2/3 domain

In GD32VW55x, the CORE_MEM0/1/2/3 power domain is defined for 48KB SRAM0 (first 16KB of SRAM0 cannot be power-off) / 64KB SRAM1 / 64KB SRAM2 / 128KB SRAM3. When run / sleep / deep-sleep mode, the SRAMs can power-off. The typical work state is as follows:

1. Power off when standby mode.
2. Power on when cotrolled by register in run / sleep / deepsleep mode.
3. Power off when controlled by register in run / sleep / deepsleep mode.

BLE_OFF domain

The typical work state is as follows:

1. Power off when standby mode.
2. Power on when cotrolled by register in run / sleep / deepsleep mode.
3. Power off (default) when controlled by register in run / sleep / deep_sleep mode.

4.3.4. Power saving modes

After a system reset or a power reset, the GD32VW55x 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 and PCLK2) or gating the clocks of the unused peripherals or configuring the LDO output voltage by LDOVS bits in PMU_CTL0 register. The LDOVS bits should be configured only when the PLL is off. Besides, six power saving modes are provided to achieve even lower power consumption, they are sleep mode, deep-sleep mode, standby, SRAM_sleep, BLE_sleep and Wi-Fi_sleep mode.

Sleep mode

The sleep mode is corresponding to the SLEEPING mode of the RISC-V. In sleep mode, only clock of RISC-V is off. To enter the sleep mode, it is only necessary to clear the CSR_SLEEPVALUE bit in the RISC-V 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. The mode offers the lowest wakeup time as no time is wasted in interrupt entry or exit.

Deep-sleep mode

The deep-sleep mode is based on the SLEEPDEEP mode of the RISC-V. In deep-sleep mode, all clocks in the 1.1V domain are off, and all of IRC16M, HXTAL and PLLs are disabled. The contents of SRAM0/1/2/3 and registers are preserved. The LDO can operate normally or in low power mode depending on the LDOLP bit in the PMU_CTL0 register. Before entering the Deep-sleep mode, it is necessary to set the CSR_SLEEPVALUE bit in the RISC-V System Control Register, and clear the STBMOD bit in the PMU_CTL0 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. When exiting the Deep-sleep mode, the IRC16M is selected as the system clock. Notice that an additional wakeup delay will be incurred if the LDO operates in low power mode.

The low-driver mode in deep-sleep mode can be entered by configuring the LDEN[1:0], LDNP, LDLP, LDOLP bits in the PMU_CTL0 register. The low-driver mode provides lower drive capability, and the low-power mode take lower power.

Normal-driver & Normal-power: The Deep-sleep mode is not in low-driver mode by configure LDEN[1:0] to 00 in the PMU_CTL0 register, and not in low-power mode depending on the LDOLP bit reset in the PMU_CTL0 register.

Normal-driver & Low-power: The Deep-sleep mode is not in low-driver mode by configure LDEN[1:0] to 00 in the PMU_CTL0 register. The low-power mode enters depending on the LDOLP bit set in the PMU_CTL0 register.

Low-driver & Normal-power: The low-driver mode in Deep-sleep mode when the LDO in normal-power mode depending on the LDOLP bit reset in the PMU_CTL0 register enters by configure LDEN[1:0] to 0b11 and LDNP to 1 in the PMU_CTL0 register.

Low-driver & Low-power: The low-driver mode in Deep-sleep mode when the LDO in low-

power mode depending on the LDOLP bit set in the PMU_CTL0 register enters by configure LDEN[1:0] to 0b11 and LDLP to 1 in the PMU_CTL0 register.

No Low-driver: The Deep-sleep mode is not in low-driver mode by configure LDEN[1:0] to 00 in the PMU_CTL0 register.

Note: In order to enter deep-sleep mode smoothly, all EXTI line pending status (in the EXTI_PD register) and RTC alarm / timestamp / tamper / auto wakeup flag must be reset. 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 RISC-V, too. In standby mode, the whole 1.1V domain is power off, the LDO is shut down, and all of IRC16M, HXTAL and PLLs are disabled. Before entering the standby mode, it is necessary to set the CSR_SLEEPVALUE bit in the RISC-V System Control Register, and set the STBMOD bit in the PMU_CTL0 register, and clear WUF bit in the PMU_CS0 register. Then, the device enters the standby mode after a WFI or WFE instruction is executed, and the STBF status flag in the PMU_CS0 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 RTC alarm / time stamp / tamper / auto wakeup events, 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 SRAM0 / SRAM1 / SRAM2 / SRAM3 and registers in 1.1V power domain are lost in standby mode. When exiting from the standby mode, a power-on reset occurs and the RISC-V will execute instruction code from the 0x00000000 address.

SRAM_sleep mode

When at least one of SRAM0 / SRAM1 / SRAM2 / SRAM3 is powered off, set the SRAMxPSLEEP (x = 0/1/2/3) bit in PMU_CTL1 register, then corresponding SRAMx (x = 0/1/2/3) will enter power off state (wait for several PCLK clocks, SRAM can completely power off and enter the SRAM sleep mode).

When the SRAMxPWAKE (x = 0/1/2/3) bit in PMU_CTL1 register is set, the SRAMx (x = 0/1/2/3) will be powered on.

SRAM0 / SRAM1 / SRAM2 / SRAM3 can be configured power on or power off when in run / sleep / deep_sleep mode.

SRAM0 / SRAM1 / SRAM2 / SRAM3 are power off when in standby mode.

BLE_sleep mode

When BLE enter BLE_sleep mode, BLE_OFF domain power off.

When exit from BLE_sleep mode, BLE is active mode, all BLE power domain power on.

Wi-Fi_sleep mode

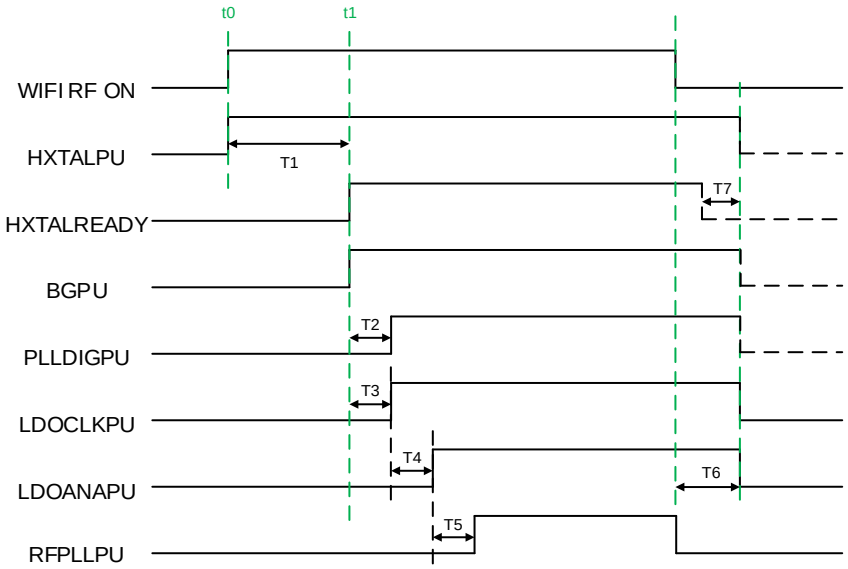
The Wi-Fi_sleep mode can enter by software (set WPEN bit to 1 and set WPSLEEP bit to 1), or by hardware (driven by Wi-Fi hardware signal sleep_wl when WPEN is 1). This mode can exit by clearing WPEN bit to 0, or by setting WPEN bit to 1 then setting WPSLEEP bit to 1, or by hardware (driven by Wi-Fi hardware signal wake_wl when WPEN is 1).

When Wi-Fi enter Wi-Fi_sleep mode, Wi-Fi_OFF domain power off.

When exit from Wi-Fi_sleep mode, Wi-Fi is active mode, all Wi-Fi power domain power on.

The RF sequence is the interface of RF module, the ***Figure 4-4. RF sequence*** shows the RF sequence.

Figure 4-4. RF sequence



HXTALPU, HXTALREADY and PLLDIGPU are bits of RCU_CTL register; BGPU, LDOCLKPU, LDOANAPU and RFPLLPU are bits of RCU_CFG1 register.

When PLLDIGPU is 1, PLLDIGOSEL[1:0] bits in RCU_PLLDIGCFG0 register should not change, and no rising edge should appear on PLLDIGEN bit of RCU_CTL register.

- If RFSWEN bit is 0, choose hardware mode to configure RF sequence:

  When set WPSLEEP bit to 1, or Wi-Fi hardware signal sleep_wl is valid, RFPLL/XTAL/BG/RF ANA clocks will be automatically closed according to ***Table 4-1. Time in RF sequence*** value in order of ***Figure 4-4. RF sequence***;

  When set WPWAKE bit to 1, or Wi-Fi hardware signal wake_wl is valid, RFPLL/XTAL/BG/RF ANA clocks will be automatically opened according to ***Table 4-1. Time in RF sequence*** value in order of ***Figure 4-4. RF sequence***.

- If RFSWEN bit is 1, choose software mode to configure RF sequence:

  RFPLL/XTAL/BG/RF ANA clocks will be opened or closed by configuring RCU related

registers(recommend to configure in order of ***Figure 4-4. RF sequence***. If related registers are not configured, the clocks will remain as before).

$$t1 = t0 + T1 \quad (5\text{-}1)$$

Table 4-1. Time in RF sequence

Name	Time	Discription
T1	HXTAL mode: 1ms. External supply mode (HXTAL bypass mode): 1us. (Note that the external supply clock is already ready)	HXTAL ready time
T2	1us	BandGap start time
T3	1us	-
T4	1us	Power up interval
T5	1us	Power up interval
T6	1us	-
T7	1us	The reserved time of simulated closing output clock of HXTAL

If HXTAL is selected to output clock, when HXTALREADY and HXTALEN in RCU_CTL register are 1, the output clock will be stable within a HXTAL period.

Table 4-2. Power saving mode summary

Mode	Sleep	Deep-sleep	Standby	SRAM_sleep	BLE_sleep	Wi-Fi_sleep
Description	Only CPU clock is off	All clocks in the 1.1V domain are off Disable IRC16M, HXTAL and PLLs	The 1.1V domain is power off Disable IRC16M, HXTAL and PLLs	at least one of SRAM1 / SRAM2 / SRAM3 is power off	BLE_OFF is power off	Wi-Fi_OFF is power off
LDO Status	On (normal power mode)	On (normal or low power mode, normal or low driver mode)	Off	On (normal or low power mode, normal or low driver mode)	On (normal or low power mode, normal or low driver mode)	On (normal or low power mode, normal or low driver mode)
Configuration	SLEEPDEEP = 0	SLEEPDEEP = 1 STBMOD = 0	SLEEPDEEP = 1 STBMOD = 1, WURST = 1	SRAMxPSLEEP = 1 (x = 1/2/3)	BLEPSLEEP = 1	1. WPEN = 1, WPSLEEP = 1 2. Or hardware signal sleep_wl valid when WPEN = 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. RTC	SRAMxPWAKE = 1(x = 1/2/3)	BLEPWAKE = 1	1. WPWAKE = 1 when WPEN = 1 2. Or clear WPEN = 0 3. Or hardware signal wake_wl valid when WPEN = 1
Wakeup Latency	None	IRC16M wakeup time, LDO wakeup time added if LDO is in low power mode	Power on sequence	100ns	520ns	2100ns

4.4. Register definition

PMU base address: 0x4000 7000

4.4.1. Control register 0 (PMU_CTL0)

Address offset: 0x00

Reset value: 0x0000 0000 (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												LDEN[1:0]		Reserved	
												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				LDNP	LDLP	Reserved	BKPWEN	LVDT[2:0]			LVDEN	STBRST	WURST	STBMOD	LDOLP
				rw	rw		rw	rw			rw	rc_w1	rc_w1	rw	rw

Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19:18	LDEN[1:0]	Low-driver mode enable in Deep-sleep mode 00: Low-driver mode disable in Deep-sleep mode 01: Reserved 10: Reserved 11: Low-driver mode enable in Deep-sleep mode.
17:12	Reserved	Must be kept at reset value.
11	LDNP	Low-driver mode when use normal power LDO 0: normal driver when use normal power LDO 1: Low-driver mode enabled when LDEN is 11 and use normal power LDO
10	LDLP	Low-driver mode when use low power LDO. 0: normal driver when use low power LDO 1: Low-driver mode enabled when LDEN is 11 and use low power LDO
9	Reserved	Must be kept at reset value.
8	BKPWEN	Backup Domain write enable 0: Disable write access to the registers in Backup domain 1: Enable write access to the registers in Backup domain After reset, any write access to the registers in Backup domain is ignored. This bit has to be set to enable write access to these registers.
7:5	LVDT[2:0]	Low Voltage Detector Threshold 000: 2.1V 001: 2.3V

		010: 2.4V 011: 2.6V 100: 2.7V 101: 2.9V 110: 3.0V 111: 3.1V
4	LVDEN	Low Voltage Detector Enable 0: Disable Low Voltage Detector 1: Enable Low Voltage Detector **Note:** When LVD_LOCK bit is set to 1 in the SYSCFG_CFG1 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 RISC-V enters SLEEPDEEP mode 1: Enter the Standby mode when the RISC-V 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

4.4.2. Control and status register 0 (PMU_CS0)

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												LDRF[1:0]		Reserved	
												rc_w1			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	LDOVSRF	Reserved		WUPEN3	WUPEN2	WUPEN1	WUPEN0	Reserved					LVDF	STBF	WUF
	r			rw	rw	rw	rw						r	r	r

Bits	Fields	Descriptions

31:20	Reserved	Must be kept at reset value.
19:18	LDRF[1:0]	Low-driver mode ready flag These bits are set by hardware when enter deep-sleep mode and the LDO in Low-driver mode. These bits are cleared by software when write 11. 00: normal driver in deep-sleep mode 01: Reserved 10: Reserved 11: Low-driver mode in Deep-sleep mode
17:15	Reserved	Must be kept at reset value.
14	LDOVSRF	LDO voltage select ready flag 0: LDO voltage select not ready 1: LDO voltage select ready
13:12	Reserved	Must be kept at reset value.
11	WUPEN3	WKUP Pin3 (PA12) enable 0: Disable WKUP pin3 function 1: Enable WKUP pin3 function If WUPEN3 is set before entering the power saving mode, a rising edge on the WKUP pin3 wakes up the system from the power saving mode. As the WKUP pin3 is active high, the WKUP pin3 is internally configured to input pull down mode. And set this bit will trigger a wakup event when the input is aready high.
10	WUPEN2	WKUP Pin2 (PA7) enable 0: Disable WKUP pin2 function 1: Enable WKUP pin2 function If WUPEN2 is set before entering the power saving mode, a rising edge on the WKUP pin2 wakes up the system from the power saving mode. As the WKUP pin2 is active high, the WKUP pin2 is internally configured to input pull down mode. And set this bit will trigger a wakup event when the input is aready high.
9	WUPEN1	WKUP Pin1 (PA15) enable 0: Disable WKUP pin1 function 1: Enable WKUP pin1 function If WUPEN1 is set before entering the power saving mode, a rising edge on the WKUP pin1 wakes up the system from the power saving mode. As the WKUP pin1 is active high, the WKUP pin1 is internally configured to input pull down mode. And set this bit will trigger a wakup event when the input is aready high.
8	WUPEN0	WKUP Pin0 (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 wakup event when the input is aready 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_CTL0 register.
0	WUF	Wakeup Flag 0: No wakeup event has been received 1: Wakeup event occurred from the WKUP pins or the RTC wakeup event including RTC Tamper event, RTC alarm event，RTC Time Stamp event or RTC Wakeup This bit is cleared only by a POR / PDR or by setting the WURST bit in the PMU_CTL0 register.

4.4.3. Control register 1 (PMU_CTL1)

Address offset: 0x08

Reset value: 0x0024 0002 (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						BLE_WAKEUP_REQ	BLEPWAKE	BLEPSLEEP	Reserved	BLE_SRAM_RET	SRAM0PWAKE	SRAM0PSLEEP	WIFI_SRAM_RET	WIFI_LPDS_ON	RETDIS
						rw	rw	rw		rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SRAM3PWAKE	SRAM3PSLEEP	Reserved		SRAM2PWAKE	SRAM2PSLEEP	Reserved		SRAM1PWAKE	SRAM1PSLEEP	Reserved	WPWAKE	WPSLEEP	WPEN	Reserved
	rw	rw			rw	rw			rw	rw		rw	rw	rw	

Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25	BLE_WAKEUP_REQ	BLE wakeup request Reset value is 0
24	BLEPWAKE	Setting this bit by software will wakeup BLE. Clear by hardware.

23	BLEPSLEEP	BLE go to sleep when setting this bit by software. Clear by hardware.
22	Reserved	Must be kept at reset value.
21	BLE_SRAM_RET	BLE SRAM enter retention mode when deepsleep and BLE_sleep. 0: Disable retention mode 1: Enable retention mode
20	SRAM0PWAKE	Setting this bit by software will wakeup SRAM0. Clear by hardware.
19	SRAM0PSLEEP	SRAM0 go to sleep when setting this bit by software. Clear by hardware.
18	WIFI_SRAM_RET	Wi-Fi SRAM enter retention mode when deepsleep and Wi-Fi_sleep. 0: Disable retention mode 1: Enable retention mode
17	WIFI_LPDS_ON	Wi-Fi auto low-power deepsleep flow on 0: Exit Wi-Fi auto low-power deepsleep flow 1: Launch Wi-Fi auto low-power deepsleep flow
16	RETDIS	No retention register when Wi-Fi power-off. 0: Enable retention mode 1: Disable retention mode
15	Reserved	Must be kept at reset value.
14	SRAM3PWAKE	Setting this bit by software will wakeup SRAM3. Clear by hardware.
13	SRAM3PSLEEP	SRAM3 go to sleep when setting this bit by software. Clear by hardware.
12:11	Reserved	Must be kept at reset value.
10	SRAM2PWAKE	Setting this bit by software will wakeup SRAM2. Clear by hardware.
9	SRAM2PSLEEP	SRAM2 go to sleep when setting this bit by software. Clear by hardware.
8:7	Reserved	Must be kept at reset value.
6	SRAM1PWAKE	Setting this bit by software will wakeup SRAM1. Clear by hardware.
5	SRAM1PSLEEP	SRAM1 go to sleep when setting this bit by software. Clear by hardware.
4	Reserved	Must be kept at reset value.
3	WPWAKE	Wi-Fi wakeup Only when WPEN bit is 1, set this bit by software will wakeup Wi-Fi. Or by clear WPEN bit (change from 1 to 0) can wakeup Wi-Fi. Clear by hardware.
2	WPSLEEP	Wi-Fi go to sleep by software only when WPEN bit is 1. Clear by hardware.
1	WPEN	Wi-Fi power enable (reset:1) (a global reset on Wi-Fi is needed after switching this bit) 0: Wi-Fi_OFF domain power on. When the Wi-Fi_OFF is power off, clear this bit will wakeup Wi-Fi (Note that the IRC16M clock should be at work)

		1: Wi-Fi_OFF domain power off when Wi-Fi sleep, and power on when Wi-Fi wakeup.(when this bit is 1, Wi-Fi power on / off can be set by software or hardware)
0	Reserved	Must be kept at reset value.

4.4.4. Control and status register 1 (PMU_CS1)

Address offset: 0x0C

Reset value: 0x8000 0010

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														BLEPS_ACTIVE	BLEPS_SLEEP
														r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SRAM3PS_ACTIVE	SRAM3PS_SLEEP	Reserved		SRAM2PS_ACTIVE	SRAM2PS_SLEEP	SRAM0PS_ACTIVE	SRAM0PS_SLEEP	SRAM1PS_ACTIVE	SRAM1PS_SLEEP	BLE_POWER_STATE	WPS_ACTIVE	WPS_SLEEP	Reserved	
	r				r	r	r	r	r	r	r	r	r		

Bits	Fields	Descriptions
31:18	Reserved	Must be kept at reset value.
17	BLEPS_ACTIVE	BLE is in active state. Read only.
16	BLEPS_SLEEP	BLE is in sleep state. Read only.
15	Reserved	Must be kept at reset value.
14	SRAM3PS_ACTIVE	SRAM3 is in active state. Read only.
13	SRAM3PS_SLEEP	SRAM3 is in sleep state. Read only.
12:11	Reserved	Must be kept at reset value.
10	SRAM2PS_ACTIVE	SRAM2 is in active state. Read only.
9	SRAM2PS_SLEEP	SRAM2 is in sleep state. Read only.
8	SRAM0PS_ACTIVE	SRAM0 is in active state. Read only.
7	SRAM0PS_SLEEP	SRAM0 is in sleep state. Read only.
6	SRAM1PS_ACTIVE	SRAM1 is in active state. Read only.
5	SRAM1PS_SLEEP	SRAM1 is in sleep state. Read only.
4	BLE_POWER_STATE	BLE power on or power off 0: BLE can enter SLEEP

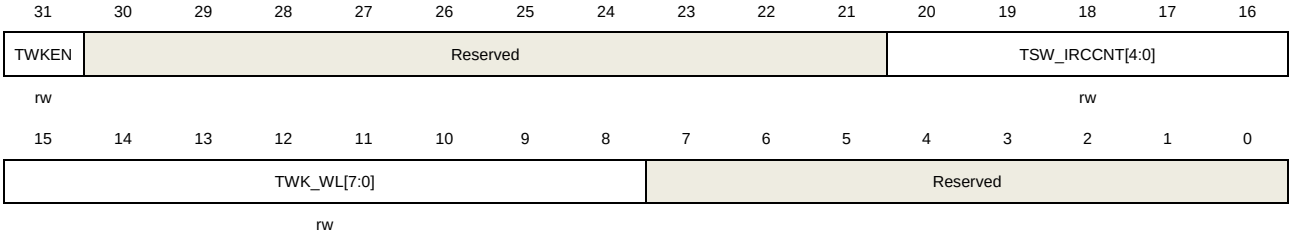
		1: BLE can wakeup
3	WPS_ACTIVE	Wi-Fi is in active state. Read only.
2	WPS_SLEEP	Wi-Fi is in sleep state. Read only.
1:0	Reserved	Must be kept at reset value.

4.4.5. Parameter register 0 (PMU_PAR0)

Address offset: 0x10

Reset value: 0x000A 2000

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



Bits	Fields	Descriptions
31	TWKEN	User SW value when wake up Wi-Fi_OFF or not. 0: Use HW ack signal when wake up Wi-Fi_OFF. 1: Use SW value when wake up Wi-Fi_OFF, the value is set by TWK_WL.
30:21	Reserved	Must be kept at reset value.
20:16	TSW_IRCCNT[4:0]	When enter deepsleep, switch to IRC16M clock. The default is 22 IRC16M clock. 0x00-0x11: 22 IRC16M CLK 0x12: 23 IRC16M CLK 0x13: 24 IRC16M CLK 0x14: 25 IRC16M CLK 0x15: 26 IRC16M CLK 0x16: 27 IRC16M CLK 0x17: 28 IRC16M CLK 0x18: 29 IRC16M CLK 0x19: 30 IRC16M CLK 0x1A: 31 IRC16M CLK 0x1B: 32 IRC16M CLK 0x1C: 33 IRC16M CLK 0x1D: 34 IRC16M CLK 0x1E: 35 IRC16M CLK 0x1F: 36 IRC16M CLK
15:8	TWK_WL[7:0]	Wakeup time of Wi-Fi_OFF domain. Clock step and max value is 64 us.

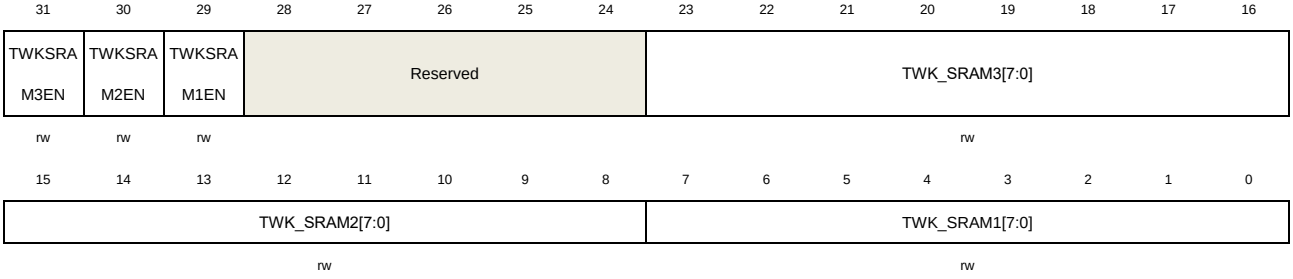
7:0	Reserved	Must be kept at reset value.

4.4.6. Parameter register 1 (PMU_PAR1)

Address offset: 0x14

Reset value: 0x0020 2020

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



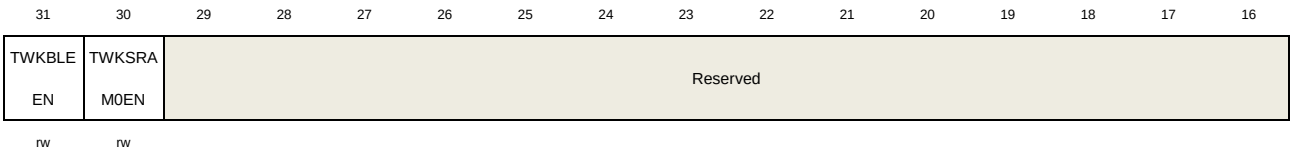
Bits	Fields	Descriptions
31	TWKSRAM3EN	User SW value when wakeup SRAM3 or not. 0: Use HW ack signal when wakeup SRAM3. 1: Use SW value when wakeup SRAM3, the value is set by TWK_SRAM3
30	TWKSRAM2EN	User SW value when wakeup SRAM2 or not. 0: Use HW ack signal when wakeup SRAM2. 1: Use SW value when wakeup SRAM2, the value is set by TWK_SRAM2
29	TWKSRAM1EN	User SW value when wakeup SRAM1 or not. 0: Use HW ack signal when wakeup SRAM1. 1: Use SW value when wakeup SRAM1, the value is set by TWK_SRAM1.
28:24	Reserved	Must be kept at reset value.
23:16	TWK_SRAM3[7:0]	Wakeup time of SRAM3 domain. 4 clock step and max value is 64 us.
15:8	TWK_SRAM2[7:0]	Wakeup time of SRAM2 domain. 4 clock step and max value is 64 us.
7:0	TWK_SRAM1[7:0]	Wakeup time of SRAM1 domain. 4 clock step and max value is 64 us.

4.4.7. Parameter register 2 (PMU_PAR2)

Address offset: 0x18

Reset value: 0x0000 2020

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



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TWK_BLE[7:0]								TWK_SRAM0[7:0]							
rw								rw							

Bits	Fields	Descriptions
31	TWKBLEEN	User SW value when wakeup BLE or not 0: Use HW ack signal when wakeup BLE. 1: Use SW value when wakeup BLE, the value is set by TWK_BLE.
30	TWKSRAM0EN	User SW value when wakeup SRAM0 or not 0: Use HW ack signal when wakeup SRAM0. 1: Use SW value when wakeup SRAM0, the value is set by TWK_SRAM0.
29:16	Reserved	Must be kept at reset value.
15:8	TWK_BLE[7:0]	Wakeup time of BLE domain. 4 clock step and max value is 64us.
7:0	TWK_SRAM0[7:0]	Wakeup time of SRAM0 domain. 4 clock step and max value is 64us.

4.4.8. RF Control register (PMU_RFCTL)

Address offset: 0x20

Reset value: 0x0000 0000

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									MCU_STAE[2:0]			RF_STATE[3:0]			
										r				r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										BLESWEN	MCU_PLL DOWN	MCU_PLL UP	RFFC	RFFS	RFSWEN
										rw	rw	rw	rw	rw	rw

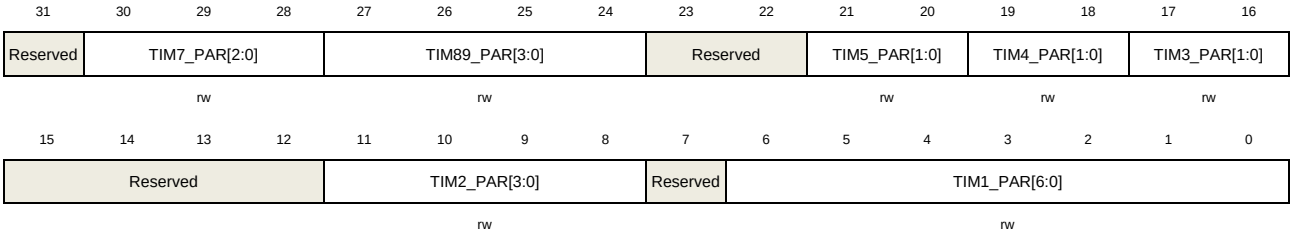
Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22:20	MCU_STATE[2:0]	MCU status 000: MCU_IDLE 011: MCU_RUN Others: reserved
19:16	RF_STATE[3:0]	RF status 0000: IDLE 1000: RFRUN Others: reserved
15:6	Reserved	Must be kept at reset value.

5	BLESWEN	0: enable RF by BLE hardware 1: enable RF by software, and RF is not affected by Bluetooth on or off
4	MCU_PLLDOWN	Software set or clear Software force close, close MCU PLL power. When wireless is in active state, set this bit shall not work.
3	MCU_PLLUP	Software set or clear(must > 2 IRC16M clock). Hardware clear. Software force open, open MCU PLL power.
2	RFFC	Software set or clear (must > 2 IRC16M clock). Hardware clear Software force close, close RF power, force to do the hardware RF sequence shutdown process.
1	RFFS	Software set or clear (must > 2 IRC16M clock). Hardware clear Software force start, open RF power, force to do the hardware RF sequence opening process.
0	RFSWEN	0: RF sequence configured automatically by hardware according to ***Figure 4-4. RF sequence*** (default) 1: RF sequence configured by software.

4.4.9. RF timer parameter register (PMU_RFPAR)

Address offset: 0x24
Reset value: 0x472A 0164

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



Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:28	TIM7_PAR[2:0]	Step:250ns, Default: 1us max:2us
27:24	TIM89_PAR[3:0]	0.1us step, default: 0.5us, max 1us
23:22	Reserved	Must be kept at reset value.
21:20	TIM5_PAR[1:0]	0.5us step, default :1us ,max: 1.5us
19:18	TIM4_PAR[1:0]	0.5us step, default :1us ,max: 1.5us
17:16	TIM3_PAR[1:0]	0.5us step, default :1us ,max: 1.5us

15:12	Reserved	Must be kept at reset value.
11:8	TIM2_PAR[3:0]	1 us step, default 1us, max: 16us
7	Reserved	Must be kept at reset value.
6:0	TIM1_PAR[6:0]	0.1ms step, default 1ms. Max: 12.7ms。 XTAL bypass mode: 125ns step default 1.25us.

4.4.10. PMU interrupt flag register(PMU_INTF)

Address offset: 0x2C
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														BLE_PS_RISEF	BLE_PS_FALLF
														r	r

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	BLE_PS_RISEF	0: No egde changes in BLE_POWER_STATUS is detected. 1: A rising edge of BLE_POWER_STATUS is detected which indicates BLE requests to wake
0	BLE_PS_FALLF	0: No egde changes in BLE_POWER_STATUS is detected. 1: Indicate a falling edge of BLE_POWER_STATUS is detected which indicates BLE is ready to sleep.

4.4.11. PMU interrupt enable register(PMU_INTEN)

Address offset: 0x30
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														BLE_PS_RISE_EN	BLE_PS_FALL_EN

rw rw

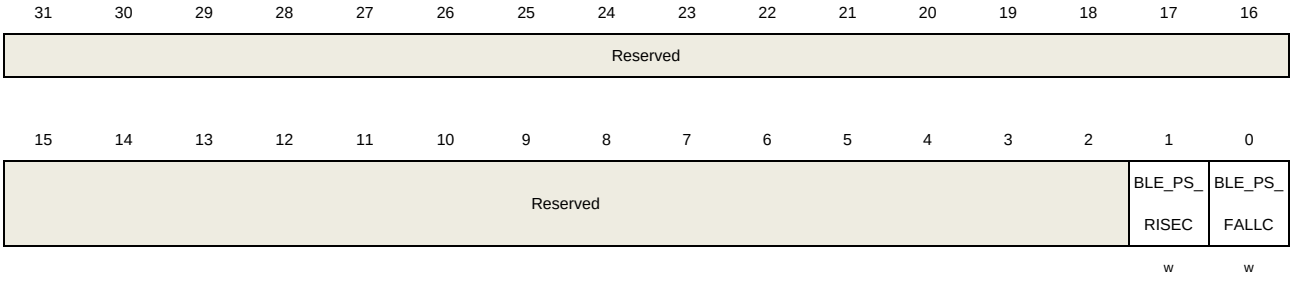
Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	BLE_PS_RISE_EN	0: Disable BLE_PS_RISE interrupt 1: Enable BLE_PS_RISE interrupt
0	BLE_PS_FALL_EN	0: Disable BLE_PS_fall interrupt 1: Enable BLE_PS_fall interrupt

4.4.12. PMU interrupt clear register(PMU_INTC)

Address offset: 0x34

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	BLE_PS_RISEFC	Set this bit to 1 clears BLE_PS_RISEF. Read this bit return 0.
0	BLE_PS_FALLFC	Set this bit to 1 clears BLE_PS_FALLF. Read this bit return 0.

5. Reset and clock unit (RCU)

5.1. Reset control unit (RCTL)

5.1.1. Overview

GD32VW55x reset control includes the control of three kinds of reset: power reset, system reset and backup domain reset. The power reset, known as a cold reset, resets the full system. The system reset resets the processor core and peripheral IP components except for the JTAG controller and the backup domain. The backup domain reset resets the backup domain. These resets can be triggered by an external signal, internal events and the reset generators. More information about these resets will be described in the following sections.

5.1.2. Function overview

Power reset

The power reset is generated by either an external reset as power on 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. The power reset whose active signal is low, it will be de-asserted when the internal LDO voltage regulator is ready to provide 1. power. The reset service routine vector is fixed at address 0x0000_0004 in the memory map.

System reset

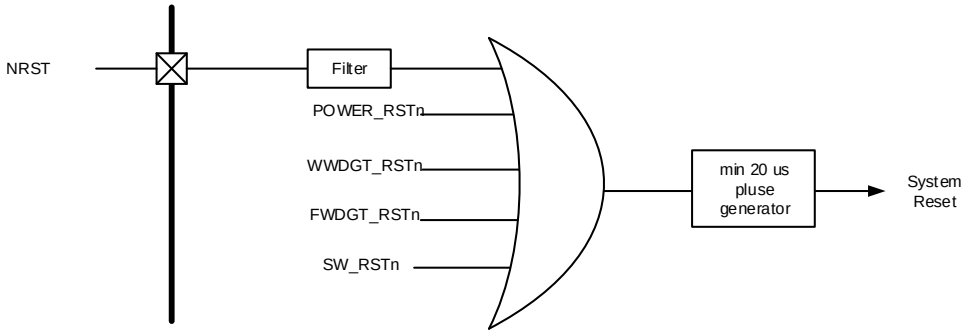
A system reset is generated by the following events:

- A power reset (POWER_RSTn).
- A external pin reset (NRST).
- A window watchdog timer reset (WWDGT_RSTn).
- A free watchdog timer reset (FWDGT_RSTn).
- The SYSRESETREQ bit in RISC-V application interrupt and reset control register is set (SW_RSTn).

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

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

Figure 5-1. The system reset circuit



Backup domain reset

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

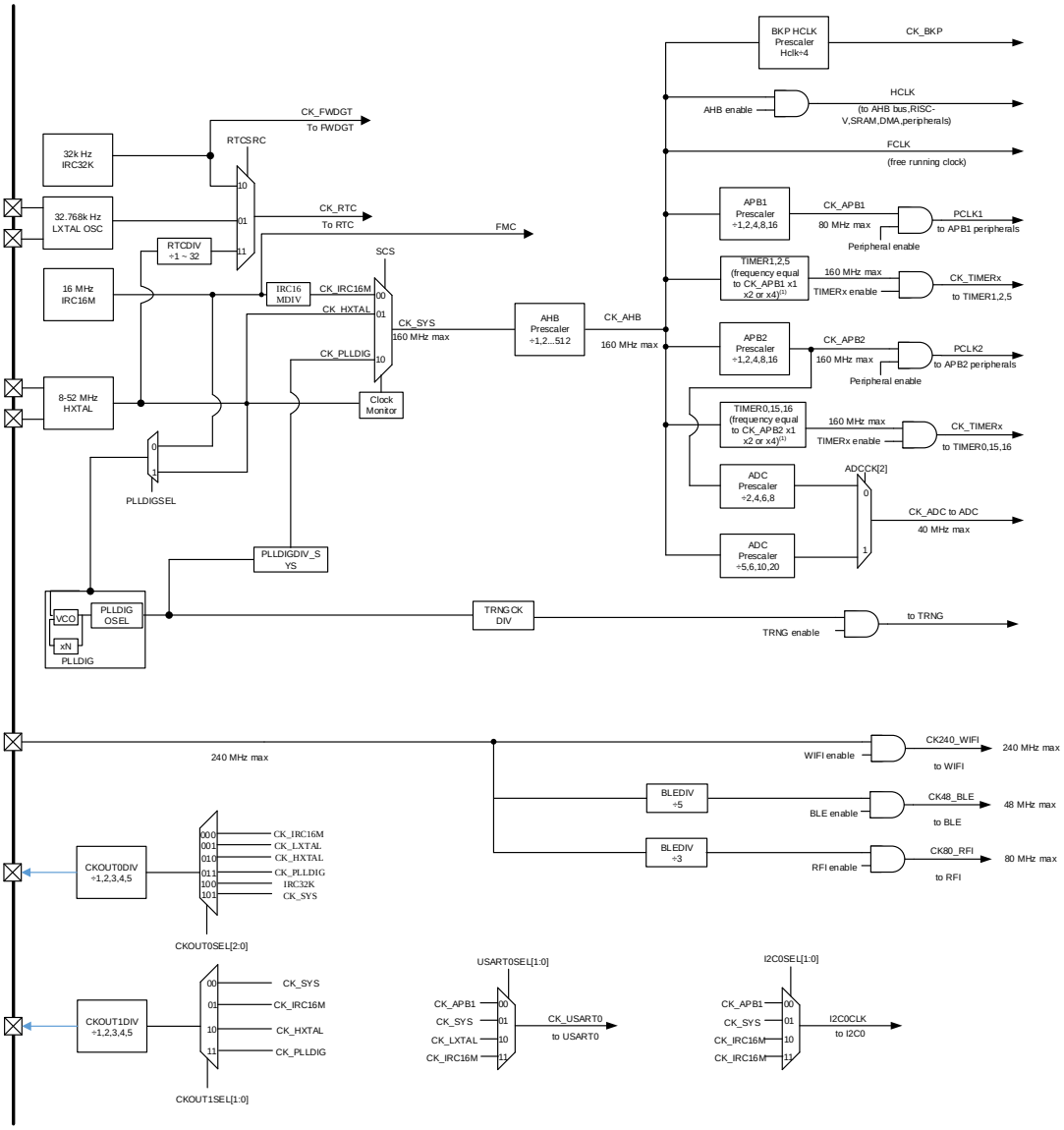
5.2. Clock control unit (CCTL)

5.2.1. Overview

The clock control unit provides a range of frequencies and clock functions. These include a internal 16 MHz RC oscillator (IRC16M), a high speed crystal oscillator (HXTAL), a low speed internal 32 KHz RC oscillator (IRC32K), a low speed crystal oscillator (LXTAL), a phase lock loop digital (PLLDIG), a HXTAL clock monitor, clock prescalers, clock multiplexers and clock gating circuitry.

The clocks of the AHB, APB and RISC-V are derived from the system clock (CK_SYS) which can source from the IRC16M, HXTAL or PLLDIG. The maximum operating frequency of the system clock (CK_SYS) can be up to 160 MHz. The free watchdog timer has independent clock source (IRC32K), and real time clock (RTC) uses the IRC32K, LXTAL or HXTAL divided by RTCDIV (in RCU_CFG0 register) as its clock source.

Figure 5-2. Clock tree



(1) Please refer to TIMERSEL bit in RCU_CFG1 for detail.

The frequency of AHB, APB2 and the APB1 domains can be configured by each prescaler. The maximum frequency of the AHB and the APB2 / APB1 domains is 160 MHz / 160 MHz / 80 MHz. The RISC-V system timer (SysTick) external clock is clocked with the AHB clock (HCLK) divided by 8. The SysTick can work either with this clock or with the AHB clock (HCLK), configurable in the systick control and status register.

The ADCs are clocked by the clock of APB2 divided by 2, 4, 6, 8 or by the clock of AHB divided by 5, 6, 10, 20, which defined by ADCCK in ADC_CCTL register.

The TIMERs are clocked by the clock divided from CK_AHB. The frequency of TIMERs clock is equal to CK_APBx, twice the CK_APBx or four times the CK_APBx. Please refer to TIMERSEL bit in RCU_CFG1 for detail.

The TRNG is clocked by the clock of PLLDIG.

The RTC is clocked by LXTAL clock or IRC32K clock or HXTAL clock divided by 1 to 32 (defined by RTCDIV bits in RCU_CFG0) which select by RTCSRC bit in backup domain control register (RCU_BDCTL). After the RTC select HXTAL clock divided by 1 to 32 (defined by RTCDIV bits in RCU_CFG0), the clock disappeared when the 1.1V core domain power off. After the RTC select IRC32K, the clock disappeared when V_{DD} power off. When the RTC select LXTAL, the clock disappeared when V_{DD} power off.

The FWDGT is clocked by IRC32K clock, which is forced on when FWDGT started.

The BLE is clocked by 240 MHz RF clock, which can be divided by 5 or 3 to 48 MHz or 80 MHz.

The USART0 is clocked by IRC16M clock or LXTAL clock or System clock or APB1 clock, which selected by USART0SEL bits in RCU_CFG1 register.

The I2C0 is clocked by IRC16M clock or System clock or APB1 clock, which selected by I2C0SEL bits in RCU_CFG1 register.

5.2.2. Characteristics

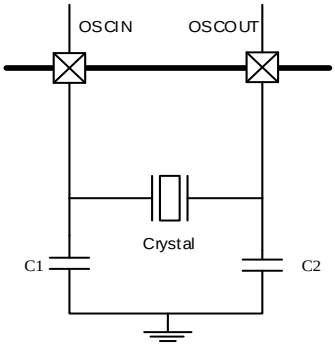
- 8 to 52 MHz High Speed crystal oscillator (HXTAL).
- Internal 16 MHz RC oscillator (IRC16M).
- 32,768 Hz Low Speed crystal oscillator (LXTAL).
- Internal 32 KHz RC oscillator (IRC32K).
- PLLDIG clock source can be HXTAL or IRC16M.
- HXTAL clock monitor.

5.2.3. Function overview

High speed crystal oscillator (HXTAL)

The high speed external crystal oscillator (HXTAL), which has a frequency from 8 to 52 MHz, produces a highly accurate clock source for use as the system clock. A crystal with a specific frequency must be connected and located close to the two HXTAL pins. The external resistor and capacitor components connected to the crystal are necessary for proper oscillation.

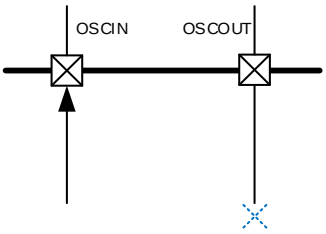
Figure 5-3. HXTAL clock source



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

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

Figure 5-4. HXTAL clock source in bypass mode



Internal 16M RC oscillators (IRC16M)

The internal 16 MHz RC oscillator, IRC16M, has a fixed frequency of 16 MHz and is the default clock source selection for the CPU when the device is powered up. The IRC16M oscillator provides a lower cost type clock source as no external components are required. The IRC16M RC oscillator can be switched on or off using the IRC16MEN bit in the control register RCU_CTL. The IRC16MSTB flag in the control register RCU_CTL is used to indicate if the internal 16 MHz RC oscillator is stable. The start-up time of the IRC16M oscillator is shorter than the HXTAL crystal oscillator. An interrupt can be generated if the related interrupt enable bit, IRC16MSTBIE, in the clock interrupt register RCU_INT, is set when the IRC16M becomes stable. The IRC16M clock can also be used as the system clock source or the PLLDIG input clock.

The frequency accuracy of the IRC16M 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 PLLDIG is the system clock source, to minimize the time required for the system to recover from the Deep-sleep Mode, the hardware forces the IRC16M clock to be the system clock when the system initially wakes-up.

If USART0 and I2C0 select IRC16M as function clock, the IRC16M will be force on according to USART0 and I2C0 function.

Phase locked loop digital (PLLDIG)

There is a internal phase locked loop digital, PLLDIG. The PLLDIG could be used to generator system clock (no more than 160 MHz) and division-clock which used to TRNG.

The PLLDIG can be switched on or off by using the PLLDIGEN / PLLDIGPU bit in the RCU_CTL register. The PLLDIGSTB flag in the RCU_CTL register will indicate if the PLLDIG clock is stable. An interrupt can be generated if the related interrupt enable bit, PLLDIGSTBIE, in the RCU_INT register, is set as the PLLDIG becomes stable.

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

Low speed crystal oscillator (LXTAL)

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

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

Internal 32K RC oscillator (IRC32K)

The internal RC oscillator has a frequency of about 32 KHz and is a low power clock source for the real time clock circuit or the free watchdog timer. The IRC32K offers a low cost clock source as no external components are required. The IRC32K RC oscillator can be switched on or off by using the IRC32KEN bit in the reset source / clock register (RCU_RSTSCK). The IRC32KSTB flag in the reset source / clock register RCU_RSTSCK will indicate if the IRC32K clock is stable. An interrupt can be generated if the related interrupt enable bit IRC32KSTBIE in the clock interrupt register (RCU_INT) is set when the IRC32K becomes stable.

System clock (CK_SYS) selection

After the system reset, the default CK_SYS source will be IRC16M and can be switched to HXTAL or CK_PLLDIG by changing the system clock switch bits, SCS, in the clock configuration register 0, RCU_CFG0. When the SCS value is changed, the CK_SYS will continue to operate using the original clock source until the target clock source is stable. When a clock source is directly or indirectly (by PLLDIG) used as the CK_SYS, 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, RFCKMEN, in the control register (RCU_CTL). This function should be enabled after the HXTAL start-up delay and disabled when the HXTAL is stopped. Once the HXTAL failure is detected, the HXTAL will be automatically disabled. The HXTAL clock stuck interrupt flag, CKMIF, in the clock interrupt register, RCU_INT, will be set and the HXTAL failure event will be generated. This failure interrupt is connected to the Non-Maskable interrupt, NMI, of the RISC-V. If the HXTAL is selected as the clock source of CK_SYS or PLLDIG and CK_PLLDIG used as system clock, the HXTAL failure will force the CK_SYS source to IRC16M and the PLLDIG will be disabled automatically. If the HXTAL is selected as the clock source of PLLDIG, the HXTAL failure will force the PLLDIG closed automatically.

Clock output capability

The clock output capability is ranging from 32 KHz to 160 MHz. There are several clock signals can be selected via the CK_OUT0 clock source selection bits, CKOUT0SEL, in the clock configuration register 0 (RCU_CFG0). The corresponding GPIO pin should be configured in the properly alternate function I/O (AFIO) mode to output the selected clock signal. The CK_OUT1 is seleced by CKOUT1SEL, in the clock configuration register 0 (RCU_CFG0).

Table 5-1. Clock output 0 source select

Clock Source 0 Selection bits	Clock Source
000	CK_IRC16M
001	CK_LXTAL
010	CK_HXTAL
011	CK_PLLDIG
100	CK_IRC32K
101	CK_SYS
110	Reserved
111	Reserved

Table 5-2. Clock output 1 source select

Clock Source 1 Selection bits	Clock Source
00	CK_SYS
01	CK_IRC16M
10	CK_HXTAL
11	CK_PLLDIG

The CK_OUT0 frequency can be reduced by a configurable binary divider, controlled by the CKOUT0DIV bits, in the clock configuration register (RCU_CFG0).

The CK_OUT1 frequency can be reduced by a configurable binary divider, controlled by the CKOUT1DIV bits, in the clock configuration register (RCU_CFG0).

Voltage control

The 1.1V domain voltage in Deep-sleep mode can be controlled by DSLPVS[1:0] bit in the Deep-sleep mode voltage register (RCU_DSV). 1.1V domain voltage selected in deep-sleep mode.

Table 5-3. 1.1V domain voltage selected in deep-sleep mode

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

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

5.3. Register definition

RCU base address: 0x4002 3800

5.3.1. Control register (RCU_CTL)

Address offset: 0x00
Reset value: 0x0040 xx83 where x is undefined.

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HXTALREADY	Reserved		HXTALPU	Reserved				PLLDIGSTB	RFCKMEN	PLLDIGEN	PLLDIGPU	Reserved	HXTALBPS	HXTALSTB	HXTALEN
rw			rw					r	rw	rw	rw	rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRC16MCALIB[7:0]								IRC16MADJ[4:0]					IRC16MFON	IRC16MSTB	IRC16MEN
r								rw					rw	r	rw

Bits	Fields	Descriptions
31	HXTALREADY	High speed crystal oscillator ready, set by software, which can be written when HXTALEN is off. 0: HXTAL is not ready. 1: HXTAL is ready.
30:29	Rservered	Must be kept at reset value.
28	HXTALPU	High speed crystal oscillator (HXTAL) power up, which can be written when HXTALEN is off. Set and reset by software. Reset by hardware when entering Deep-sleep or standby mode. 0: High speed crystal oscillator power down. 1: High speed crystal oscillator power up.
27:24	Rservered	Must be kept at reset value.
23	PLLDIGSTB	PLLDIG clock stabilization flag Set by hardware to indicate if the PLLDIG output clock is stable and ready for use. 0: PLLDIG is not stable. 1: PLLDIG is stable.
22	RFCKMEN	HXTAL clock monitor enable, check RF XTAL 0: Disable the high speed crystal oscillator (HXTAL) clock monitor. 1: Enable the high speed crystal oscillator (HXTAL) clock monitor.
21	PLLDIGEN	PLLDIG enable, this bit cannot be reset if the PLLDIG clock is used as the system

		clock Set and reset by software. Reset by hardware when entering Deep-sleep or standby mode. 0: PLLDIG disable. 1: PLLDIG enable.
20	PLLDIGPU	PLLDIG power up, this bit cannot be reset if the PLLDIG clock is used as the system clock Set and reset by software. Reset by hardware when entering Deep-sleep or standby mode. 0: PLLDIG power down. 1: PLLDIG power up.
19	Rservered	Must be kept at reset value.
18	HXTALBPS	High speed crystal oscillator (HXTAL) clock bypass mode enable The HXTALBPS bit can be written only if the HXTALEN and HXTALPU both are 0. 0: Disable the HXTAL bypass mode. 1: Enable the HXTAL bypass mode in which the HXTAL output clock is equal to the input clock.
17	HXTALSTB	High speed crystal oscillator (HXTAL) clock stabilization flag Set by hardware to indicate if the HXTAL oscillator is stable and ready for use. 0: HXTAL oscillator is not stable. 1: HXTAL oscillator is stable.
16	HXTALEN	High speed crystal oscillator (HXTAL) enable Set and reset by software. This bit cannot set disabled if the HXTAL clock is used as the system clock, or if the PLLDIG is used as the system clock with the HXTAL used as the source clock of the PLLDIG. If HXTAL clock as the system clock, hardware limitations can't disable this bit. When PLLDIG clock as the system clock, and HXTAL as PLLDIG input clock, if this bit is forced to disabled, the system clock will automatically jump to IRC16M system clock. Reset by hardware when entering Deep-sleep or standby mode. Reset by hardware when CKM check fail. 0: High speed crystal oscillator disabled. 1: High speed crystal oscillator enabled.
15:8	IRC16MCALIB[7:0]	Internal 16 MHz RC Oscillator calibration value register These bits are load automatically at power on.
7:3	IRC16MADJ[4:0]	Internal 16 MHz RC oscillator clock trim adjust value These bits are set by software. The trimming value is these bits (IRC16MADJ) added to the IRC16MCALIB[7:0] bits. The trimming value should trim the IRC16M to 16 MHz ± 1%.
2	IRC16MRFON	Internal 16 MHz RC RF differential clock signal enable Set and reset by software. This bit can be set if IRC16MEN is set to 1. 0: Internal 16 MHz RC RF differential clock disable.

		1: Internal 16 MHz RC RF differential clock enable.
1	IRC16MSTB	IRC16M internal 16MHz RC oscillator stabilization flag Set by hardware to indicate if the IRC16M oscillator is stable and ready for use. 0: IRC16M oscillator is not stable. 1: IRC16M oscillator is stable.
0	IRC16MEN	Internal 16 MHz RC oscillator enable Set and reset by software. This bit cannot be reset if the IRC16M 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 RFCKMEN is set. 0: Internal 16 MHz RC oscillator disabled. 1: Internal 16 MHz RC oscillator enabled.

5.3.2. PLL register (RCU_PLL)

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
Reserved															

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	PLLDIGSEL	PLLDIG clock source selection Set and reset by software to control the PLLDIG clock source. 0: IRC16M clock selected as source clock of PLLDIG. 1: HXTAL clock selected as source clock of PLLDIG.
14:0	Reserved	Must be kept at reset value.

5.3.3. Clock configuration register 0 (RCU_CFG0)

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

CKOUT1SEL[1:0]	CKOUT1DIV[2:0]	CKOUT0DIV[2:0]	CKOUT0SEL[2:0]	RTCDIV[4:0]
rw	rw	rw	rw	rw

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

Bits	Fields	Descriptions
31:30	CKOUT1SEL[1:0]	CKOUT1 clock source selection Set and reset by software. 00: Sysclk selected. 01: Internal 16M RC Oscillator clock selected. 10: High speed crystal oscillator clock (HXTAL) selected. 11: CK_PLLDIG clock selected.
29:27	CKOUT1DIV[2:0]	The CK_OUT1 divider which the CK_OUT1 frequency can be reduced See bits 31:30 of RCU_CFG0 for CK_OUT1. 0xx: The CK_OUT1 is divided by 1. 100: The CK_OUT1 is divided by 2. 101: The CK_OUT1 is divided by 3. 110: The CK_OUT1 is divided by 4. 111: The CK_OUT1 is divided by 5.
26:24	CKOUT0DIV[2:0]	The CK_OUT0 divider which the CK_OUT0 frequency can be reduced See bits 23:21 of RCU_CFG0 for CK_OUT0. 0xx: The CK_OUT0 is divided by 1. 100: The CK_OUT0 is divided by 2. 101: The CK_OUT0 is divided by 3. 110: The CK_OUT0 is divided by 4. 111: The CK_OUT0 is divided by 5.
23:21	CKOUT0SEL[2:0]	CKOUT0 clock source selection Set and reset by software. 000: Internal 16M RC Oscillator clock selected. 001: Low speed crystal oscillator clock (LXTAL) selected. 010: High speed crystal oscillator clock (HXTAL) selected. 011: CK_PLLDIG clock selected. 100: Internal 32K RC Oscillator clock selected. 101: SYSTEM clock selected. 110 / 111: Reserved.
20:16	RTCDIV[4:0]	RTC clock divider factor Set and reset by software. These bits is used to generator clock for RTC (no more than 1MHz) from HXTAL clock. 00000: CK_HXTAL / 1.

		00001: CK_HXTAL / 2. 00010: CK_HXTAL / 3. 00011: CK_HXTAL / 4. … 11111: CK_HXTAL / 32.
15:13	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.
12:10	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.
9:8	Reserved	Must be kept at reset value.
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_IRC16M as the CK_SYS source. 01: select CK_HXTAL as the CK_SYS source. 10: select CK_PLLDIG as the CK_SYS source. 11: no clock for CK_SYS source.
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 IRC16M when leaving Deep-sleep and

standby mode or HXTAL failure is detected by HXTAL clock monitor when HXTAL is selected directly or indirectly as the clock source of CK_SYS. Does not take effect when 2'b11 is written. SCS will keep the previous value.
00: select CK_IRC16M as the CK_SYS source.
01: select CK_HXTAL as the CK_SYS source.
10: select CK_PLLDIG as the CK_SYS source.
11: no clock for CK_SYS source.

5.3.4. Clock interrupt register (RCU_INT)

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								CKMIC	PLLDIGSTBIC	Reserved		HXTALSTBIC	IRC16MSTBIC	LXTALSTBIC	IRC32KSTBIC
								w	w			w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	PLLDIGSTBIE	Reserved		HXTALSTBIE	IRC16MSTBIE	LXTALSTBIE	IRC32KSTBIE	CKMIF	PLLDIGSTBIF	Reserved		HXTALSTBIF	IRC16MSTBIF	LXTALSTBIF	IRC32KSTBIF
	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	PLLDIGSTBIC	PLLDIG stabilization interrupt clear Write 1 by software to reset the PLLDIGSTBIF flag. 0: Not reset PLLDIGSTBIF flag. 1: Reset PLLDIGSTBIF flag.
21:20	Reserved	Must be kept at reset value.
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	IRC16MSTBIC	IRC16M stabilization interrupt clear Write 1 by software to reset the IRC16MSTBIF flag. 0: Not reset IRC16MSTBIF flag.

		1: Reset IRC16MSTBIF flag.
17	LXTALSTBIC	LXTAL stabilization interrupt clear Write 1 by software to reset the LXTALSTBIF flag. 0: Not reset LXTALSTBIF flag. 1: Reset LXTALSTBIF flag.
16	IRC32KSTBIC	IRC32K Stabilization interrupt clear Write 1 by software to reset the IRC32KSTBIF flag. 0: Not reset IRC32KSTBIF flag. 1: Reset IRC32KSTBIF flag.
15	Reserved	Must be kept at reset value.
14	PLLDIGSTBIE	PLLDIG stabilization interrupt enable Set and reset by software to enable / disable the PLLDIG stabilization interrupt. 0: Disable the PLLDIG stabilization interrupt. 1: Enable the PLLDIG stabilization interrupt.
13:12	Reserved	Must be kept at reset value.
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	IRC16MSTBIE	IRC16M stabilization interrupt enable Set and reset by software to enable / disable the IRC16M stabilization interrupt. 0: Disable the IRC16M stabilization interrupt. 1: Enable the IRC16M stabilization interrupt.
9	LXTALSTBIE	LXTAL stabilization interrupt enable LXTAL stabilization interrupt enable / disable control. 0: Disable the LXTAL stabilization interrupt. 1: Enable the LXTAL stabilization interrupt.
8	IRC32KSTBIE	IRC32K stabilization interrupt enable IRC32K stabilization interrupt enable / disable control. 0: Disable the IRC32K stabilization interrupt. 1: Enable the IRC32K stabilization interrupt.
7	CKMIF	HXTAL clock stuck interrupt flag Set by hardware when the HXTAL clock is stuck. Reset when setting the CKMIC bit by software. 0: Clock operating normally. 1: HXTAL clock stuck.
6	PLLDIGSTBIF	PLLDIG stabilization interrupt flag Set by hardware when the PLLDIG is stable and the PLLDIGSTBIE bit is set.

		Reset when setting the PLLDIGSTBIC bit by software. 0: No PLLDIG stabilization interrupt generated. 1: PLLDIG stabilization interrupt generated.
5:4	Reserved	Must be kept at reset value.
3	HXTALSTBIF	HXTAL stabilization interrupt flag Set by hardware when the High speed 8~ 52 MHz crystal oscillator clock is stable and the HXTALSTBIE bit is set. Reset when setting the HXTALSTBIC bit by software. 0: No HXTAL stabilization interrupt generated. 1: HXTAL stabilization interrupt generated.
2	IRC16MSTBIF	IRC16M stabilization interrupt flag Set by hardware when the Internal 16 MHz RC oscillator clock is stable and the IRC16MSTBIE bit is set. Reset when setting the IRC16MSTBIC bit by software. 0: No IRC16M stabilization interrupt generated. 1: IRC16M stabilization interrupt generated.
1	LXTALSTBIF	LXTAL stabilization interrupt flag Set by hardware when the low speed 32,768 Hz crystal oscillator clock is stable and the LXTALSTBIE bit is set. Reset when setting the LXTALSTBIC bit by software. 0: No LXTAL stabilization interrupt generated. 1: LXTAL stabilization interrupt generated.
0	IRC32KSTBIF	IRC32K stabilization interrupt flag Set by hardware when the internal 32 KHz RC oscillator clock is stable and the IRC32KSTBIE bit is set. Reset when setting the IRC32KSTBIC bit by software. 0: No IRC32K stabilization clock ready interrupt generated. 1: IRC32K stabilization interrupt generated.

5.3.5. AHB1 reset register (RCU_AHB1RST)

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
BLERST	Rserved									DMARST	Reserved				
rw										rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		WIFIRST	CRCRST	Reserved									PCRST	PBRST	PARST

rw rw rw rw rw

Bits	Fields	Descriptions
31	BLERST	BLE reset This bit is set and reset by software. 0: No reset. 1: Reset the BLE.
30:22	Reserved	Must be kept at reset value.
21	DMARST	DMA reset This bit is set and reset by software. 0: No reset. 1: Reset the DMA.
20:14	Reserved	Must be kept at reset value.
13	WIFIRST	WIFI reset This bit is set and reset by software. 0: No reset. 1: Reset the WIFI.
12	CRCRST	CRC reset This bit is set and reset by software. 0: No reset. 1: Reset the CRC.
11:3	Reserved	Must be kept at reset value.
2	PCRST	GPIO port C reset This bit is set and reset by software. 0: No reset. 1: Reset the GPIO port C.
1	PBRST	GPIO port B reset This bit is set and reset by software. 0: No reset. 1: Reset the GPIO port B.
0	PARST	GPIO port A reset This bit is set and reset by software. 0: No reset. 1: Reset the GPIO port A.

5.3.6. AHB2 reset register (RCU_AHB2RST)

Address offset: 0x14

Reset value: 0x0000 0000

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TRNGRST	HAURST	CAURST	PKCAURST	Reserved		
									rw	rw	rw	rw			

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	TRNGRST	TRNG reset This bit is set and reset by software. 0: No reset. 1: Reset the TRNG.
5	HAURST	HAU reset This bit is set and reset by software. 0: No reset. 1: Reset the HAU.
4	CAURST	CAU reset This bit is set and reset by software. 0: No reset. 1: Reset the CAU.
3	PKCAURST	PKCAU reset This bit is set and reset by software. 0: No reset. 1: Reset the PKCAU.
2:0	Reserved	Must be kept at reset value.

5.3.7. AHB3 reset register (RCU_AHB3RST)

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															

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

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	QSPIRST	QSPI reset This bit is set and reset by software. 0: No reset. 1: Reset the QSPI.
0	Reserved	Must be kept at reset value.

5.3.8. APB1 reset register (RCU_APB1RST)

Address offset: 0x20

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			PMURST	Reserved					I2C1RST	I2C0RST	Reserved		USART0RST	UART1RST	Reserved
			rw						rw	rw			rw	rw	

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

Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28	PMURST	PMU reset This bit is set and reset by software. 0: No reset. 1: Reset the PMU.
27:23	Reserved	Must be kept at reset value.
22	I2C1RST	I2C1 reset This bit is set and reset by software. 0: No reset. 1: Reset the I2C1.
21	I2C0RST	I2C0 reset This bit is set and reset by software. 0: No reset. 1: Reset the I2C0.

20:19	Reserved	Must be kept at reset value
18	USART0RST	USART0 reset This bit is set and reset by software. 0: No reset. 1: Reset the USART0.
17	UART1RST	UART1 reset This bit is set and reset by software. 0: No reset. 1: Reset the UART1.
16	Reserved	Must be kept at reset value.
15	RFIRST	RFI reset This bit is set and reset by software. 0: No reset. 1: Reset the RFI.
14:12	Reserved	Must be kept at reset value.
11	WWDGTRST	WWDGT reset This bit is set and reset by software. 0: No reset. 1: Reset the WWDGT.
10:5	Reserved	Must be kept at reset value.
4	TIMER5RST	TIMER5 reset This bit is set and reset by software. 0: No reset. 1: Reset the TIMER5.
3:2	Reserved	Must be kept at reset value.
1	TIMER2RST	TIMER2 reset This bit is set and reset by software. 0: No reset. 1: Reset the TIMER2.
0	TIMER1RST	TIMER1 reset This bit is set and reset by software. 0: No reset. 1: Reset the TIMER1.

5.3.9. APB2 reset register (RCU_APB2RST)

Address offset: 0x24

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
RFRST	Reserved												TIMER16RST	TIMER15RST	Reserved
rw													rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SYSCFGRST	Reserved	SPIRST	Reserved			ADCRST	Reserved			UART2RST	Reserved			TIMER0RST
	rw		rw				rw				rw				rw

Bits	Fields	Descriptions
31	RFRST	RF reset This bit is set and reset by software. 0: No reset. 1: Reset the RF.
30:19	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.
16:15	Reserved	Must be kept at reset value.
14	SYSCFGRST	SYSCFG reset This bit is set and reset by software. 0: No reset. 1: Reset the SYSCFG.
13	Reserved	Must be kept at reset value.
12	SPIRST	SPI reset This bit is set and reset by software. 0: No reset. 1: Reset the SPI.
11:9	Reserved	Must be kept at reset value.
8	ADCRST	ADC reset This bit is set and reset by software. 0: No reset.

		1: Reset the ADC.
7:5	Reserved	Must be kept at reset value.
4	UART2RST	UART2 reset This bit is set and reset by software. 0: No reset. 1: Reset the UART2.
3:1	Reserved	Must be kept at reset value.
0	TIMER0RST	TIMER0 reset This bit is set and reset by software. 0: No reset. 1: Reset the TIMER0.

5.3.10. AHB1 enable register (RCU_AHB1EN)

Address offset: 0x30

Reset value: 0x000F 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
BLEEN	Reserved									DMAEN	Reserved	SRAM3EN	SRAM2EN	SRAM1EN	SRAM0EN
rw										rw		rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	WIFIRUNEN	WIFIEN	CRCEN	Reserved									PCEN	PBEN	PAEN
	rw	rw	rw										rw	rw	rw

Bits	Fields	Descriptions
31	BLEEN	BLE clock enable This bit is set and reset by software. 0: Disabled BLE clock. 1: Enabled BLE clock.
30:22	Reserved	Must be kept at reset value.
21	DMAEN	DMA clock enable This bit is set and reset by software. 0: Disabled DMA clock. 1: Enabled DMA clock.
20	Reserved	Must be kept at reset value.
19	SRAM3EN	SRAM3 clock enable

		This bit is set and reset by software. 0: Disabled SRAM3 clock. 1: Enabled SRAM3 clock.
18	SRAM2EN	SRAM2 clock enable This bit is set and reset by software. 0: Disabled SRAM2 clock. 1: Enabled SRAM2 clock.
17	SRAM1EN	SRAM1 clock enable This bit is set and reset by software. 0: Disabled SRAM1 clock. 1: Enabled SRAM1 clock.
16	SRAM0EN	SRAM0 clock enable This bit is set and reset by software. 0: Disabled SRAM0 clock. 1: Enabled SRAM0 clock.
15	Reserved	Must be kept at reset value.
14	WIFIRUNEN	WIFIRUNEN clock enable This bit is set and reset by software. If WIFIEN is 0, this bit doesn't work. 0: Disabled WIFIRUNEN clock. 1: Enabled WIFIRUNEN clock.
13	WIFIEN	WIFI Module clock enable This bit is set and reset by software. 0: Disabled WIFI clock. 1: Enabled WIFI clock.
12	CRCEN	CRC clock enable This bit is set and reset by software. 0: Disabled CRC clock. 1: Enabled CRC clock.
11:3	Reserved	Must be kept at reset value.
2	PCEN	GPIO port C clock enable This bit is set and reset by software. 0: Disabled GPIO port C clock. 1: Enabled GPIO port C clock.
1	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.
0	PAEN	GPIO port A clock enable

This bit is set and reset by software.

0: Disabled GPIO port A clock.

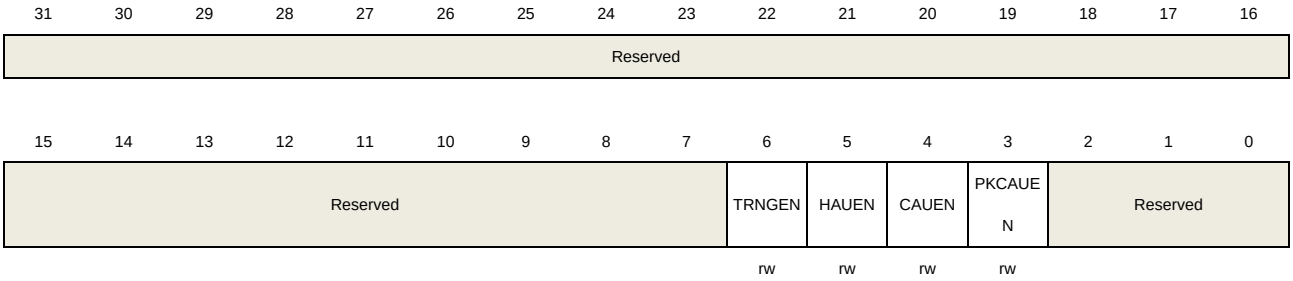
1: Enabled GPIO port A clock.

5.3.11. AHB2 enable register (RCU_AHB2EN)

Address offset: 0x34

Reset value: 0x0000 0000

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



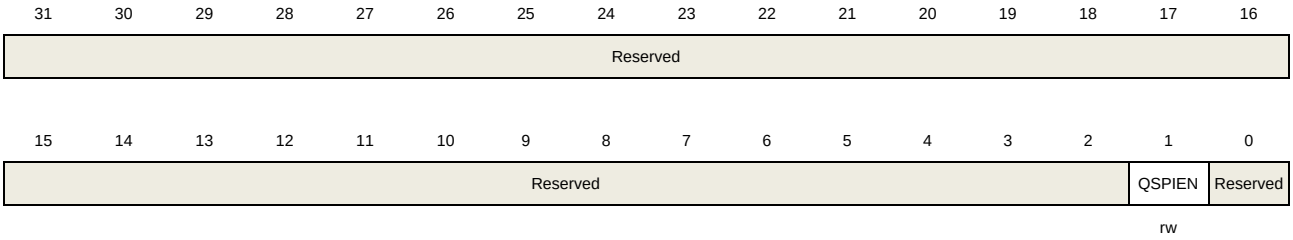
Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	TRNGEN	TRNG clock enable This bit is set and reset by software. 0: Disabled TRNG clock. 1: Enabled TRNG clock.
5	HAUEN	HAU clock enable This bit is set and reset by software. 0: Disabled HAU clock. 1: Enabled HAU clock.
4	CAUEN	CAU clock enable This bit is set and reset by software. 0: Disabled CAU clock. 1: Enabled CAU clock.
3	PKCAUEN	PKCAU clock enable This bit is set and reset by software. 0: Disabled PKCAU clock. 1: Enabled PKCAU clock.
2:0	Reserved	Must be kept at reset value.

5.3.12. AHB3 enable register (RCU_AHB3EN)

Address offset: 0x38

Reset value: 0x0000 0000

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



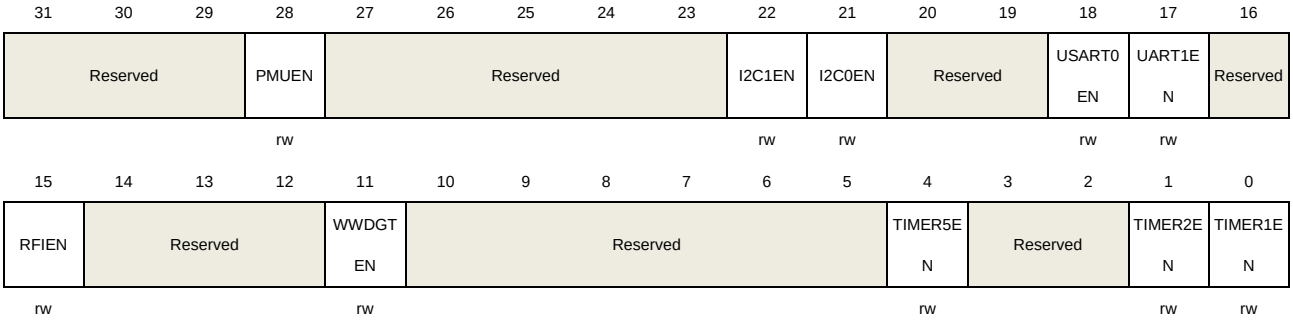
Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	QSPIEN	QSPI clock enable This bit is set and reset by software. 0: Disabled QSPI clock. 1: Enabled QSPI clock.
0	Reserved	Must be kept at reset value.

5.3.13. APB1 enable register (RCU_APB1EN)

Address offset: 0x40

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28	PMUEN	PMU clock enable This bit is set and reset by software. 0: Disabled PMU clock. 1: Enabled PMU clock.
27:23	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.
20:19	Reserved	Must be kept at reset value.
18	USART0EN	USART0 clock enable This bit is set and reset by software. 0: Disabled USART0 clock. 1: Enabled USART0 clock.
17	UART1EN	UART1 clock enable This bit is set and reset by software. 0: Disabled UART1 clock. 1: Enabled UART1 clock.
16	Reserved	Must be kept at reset value.
15	RFIEN	RFI clock enable This bit is set and reset by software. 0: Disabled RFI clock. 1: Enabled RFI clock.
14:12	Reserved	Must be kept at reset value.
11	WWDGTEN	WWDGT clock enable This bit is set and reset by software. 0: Disabled WWDGT clock. 1: Enabled WWDGT clock.
10:5	Reserved	Must be kept at reset value.
4	TIMER5EN	TIMER5 clock enable This bit is set and reset by software. 0: Disabled TIMER5 clock. 1: Enabled TIMER5 clock.
3:2	Reserved	Must be kept at reset value.
1	TIMER2EN	TIMER2 clock enable This bit is set and reset by software. 0: Disabled TIMER2 clock. 1: Enabled TIMER2 clock.
0	TIMER1EN	TIMER1 clock enable

This bit is set and reset by software.

0: Disabled TIMER1 clock.

1: Enabled TIMER1 clock.

5.3.14. APB2 enable register (RCU_APB2EN)

Address offset: 0x44

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	RFEN	RF clock enable This bit is set and reset by software. 0: Disabled RF clock. 1: Enabled RF clock.
30:19	Reserved	Must be kept at reset value.
18	TIMER16EN	TIMER16 clock enable This bit is set and reset by software. 0: Disabled TIMER16 clock. 1: Enabled TIMER16 clock.
17	TIMER15EN	TIMER15 clock enable This bit is set and reset by software. 0: Disabled TIMER15 clock. 1: Enabled TIMER15 clock.
16:15	Reserved	Must be kept at reset value.
14	SYSCFGEN	SYSCFG clock enable This bit is set and reset by software. 0: Disabled SYSCFG clock. 1: Enabled SYSCFG clock.
13	Reserved	Must be kept at reset value.
12	SPIEN	SPI clock enable

		This bit is set and reset by software. 0: Disabled SPI clock. 1: Enabled SPI clock.
11:9	Reserved	Must be kept at reset value.
8	ADCEN	ADC clock enable This bit is set and reset by software. 0: Disabled ADC clock. 1: Enabled ADC clock.
7:5	Reserved	Must be kept at reset value.
4	UART2EN	UART2 clock enable This bit is set and reset by software. 0: Disabled UART2 clock. 1: Enabled UART2 clock.
3:1	Reserved	Must be kept at reset value.
0	TIMER0EN	TIMER0 clock enable This bit is set and reset by software. 0: Disabled TIMER0 clock. 1: Enabled TIMER0 clock.

5.3.15. AHB1 sleep mode enable register (RCU_AHB1SPEN)

Address offset: 0x50

Reset value: 0x0000 8000

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
FMCSPEN	Reserved														
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	FMCSPEN	FMC clock enable when sleep mode This bit is set and reset by software. 0: Disabled FMC clock when sleep mode 1: Enabled FMC clock when sleep mode

14:0	Reserved	Must be kept at reset value.

5.3.16. Backup domain control register (RCU_BDCTL)

Address offset: 0x70

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

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

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCEN	Reserved					RTCSRC[1:0]		Reserved			LXTALDRI[1:0]		LXTALBPS	LXTALSTB	LXTALEN
rw						rw					rw		rw	r	rw

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	BKPRST	Backup domain reset This bit is set and reset by software. 0: No reset. 1: Resets backup domain.
15	RTCEN	RTC clock enable This bit is set and reset by software. 0: Disabled RTC clock. 1: Enabled RTC clock.
14:10	Reserved	Must be kept at reset value.
9:8	RTCSRC[1:0]	RTC clock entry selection Set and reset by software to control the RTC clock source. Once the RTC clock source has been selected, it cannot be changed anymore unless the backup domain is reset. 00: No clock selected. 01: CK_LXTAL selected as RTC source clock. 10: CK_IRC32K selected as RTC source clock. 11: (CK_HXTAL / RTCDIV) selected as RTC source clock, please refer to RTCDIV bits in RCU_CFG0 register.
7:5	Reserved	Must be kept at reset value.

4:3	LXTALDRI[1:0]	LXTAL drive capability Set and reset by software. Backup domain reset resets this value. 00: Lower driving capability. 01: High driving capability. 10: Higher driving capability. 11: Highest driving capability (reset value). **Note**: The LXTALDRI is not in bypass mode.
2	LXTALBPS	LXTAL bypass mode enable Set and reset by software. 0: Disable the LXTAL Bypass mode. 1: Enable the LXTAL Bypass mode.
1	LXTALSTB	Low speed crystal oscillator stabilization flag Set by hardware to indicate if the LXTAL output clock is stable and ready for use. 0: LXTAL is not stable. 1: LXTAL is stable.
0	LXTALEN	LXTAL enable Set and reset by software. 0: Disable LXTAL. 1: Enable LXTAL.

5.3.17. Reset source / clock register (RCU_RSTSCK)

Address offset: 0x74

Reset value: 0x0C00 0000, all reset flags reset by power reset only, RSTFC / IRC32KEN reset by system reset.

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPRSTF	WWDGTRSTF	FWDGTRSTF	SWRSTF	PORRSTF	EPRSTF	Reserved	RSTFC	Reserved							
r	r	r	r	r	r		rw								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														IRC32KSTB	IRC32KEN
														r	rw

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

		1: Low-power management reset generated.
30	WWDGTRSTF	Window watchdog timer reset flag Set by hardware when a window watchdog timer reset generated. Reset by writing 1 to the RSTFC bit. 0: No window watchdog reset generated. 1: Window watchdog reset generated.
29	FWDGTRSTF	Free watchdog timer reset flag Set by hardware when a free watchdog timer reset generated. Reset by writing 1 to the RSTFC bit. 0: No free watchdog timer reset generated. 1: Free Watchdog timer reset generated.
28	SWRSTF	Software reset flag Set by hardware when a software reset generated. Reset by writing 1 to the RSTFC bit. 0: No software reset generated. 1: Software reset generated.
27	PORRSTF	Power reset flag Set by hardware when a Power reset generated. Reset by writing 1 to the RSTFC bit. 0: No Power reset generated. 1: Power reset generated.
26	EPRSTF	External pin reset flag Set by hardware when an external pin reset generated. Reset by writing 1 to the RSTFC bit. 0: No external pin reset generated. 1: External pin reset generated.
25	Reserved	Must be kept at reset value.
24	RSTFC	Reset flag clear This bit is set by software to clear all reset flags. 0: Not clear reset flags. 1: Clear reset flags.
23:2	Reserved	Must be kept at reset value.
1	IRC32KSTB	IRC32K stabilization flag Set by hardware to indicate if the IRC32K output clock is stable and ready for use. 0: IRC32K is not stable. 1: IRC32K is stable.
0	IRC32KEN	IRC32K enable Set and reset by software. 0: Disable IRC32K.

1: Enable IRC32K.

5.3.18. PLLDIG clock configuration register 0 (RCU_PLLDIGCFG0)

Address offset: 0x84

Reset value: 0x0B00 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
PLLDIGDIV_SYS[5:0]						PLLDIGOSEL[1:0]		Reserved							
rw						rw									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31:26	PLLDIGDIV_SYS[5:0]	PLLDIG clock divider factor for system clock Set and reset by software to control the PLLDIG clock divider factor for system clock. 000000: PLLDIG clock divided by 1 for system clock. 000001: PLLDIG clock divided by 2 for system clock. 000010: PLLDIG clock divided by 3 for system clock. … 111111: PLLDIG clock divided by 64 for system clock.
25:24	PLLDIGOSEL[1:0]	PLLDIG output frequency select 00: selected 192Mhz as PLLDIG output frequency. 01: selected 240Mhz as PLLDIG output frequency. 10: selected 320Mhz as PLLDIG output frequency. 11: selected 480Mhz as PLLDIG output frequency.
23:0	Reserved	Must be kept at reset value.

5.3.19. Clock configuration register 1 (RCU_CFG1)

Address offset: 0x8C

Reset value: 0x0030 0600

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
USART0SEL[1:0]		Reserved		I2C0SEL[1:0]		Reserved	TIMERSEL	Reserved		LDO_ANA_LQB	LDO_CLK_LQB	BGPU	LDOCLKPU	LDOANAPU	RFPLLPU
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

RFPLLLOCK	RFPLLCALEN	Reserved	BGVBIT[2:0]	IRC16MDIV[8:0]
ro	rw		rw	rw

Bits	Fields	Descriptions
31:30	USART0SEL[1:0]	USART0 Clock Source Selection Set and reset by software to control the USART0 clock source. 00: CK_APB1 selected as USART0 source clock. 01: CK_SYS selected as USART0 source clock. 10: CK_LXTAL selected as USART0 source clock. 11: CK_IRC16M selected as USART0 source clock.
29:28	Reserved	Must be kept at reset value.
27:26	I2C0SEL[1:0]	I2C0 Clock Source Selection Set and reset by software to control the I2C0 clock source. 00: CK_APB1 selected as I2C0 source clock. 01: CK_SYS selected as I2C0 source clock. 1x: CK_IRC16M selected as I2C0 source clock.
25	Reserved	Must be kept at reset value.
24	TIMERSEL	TIMER clock selection This bit is set and reset by software. This bit defined all timer clock selection. 0: If APB1PSC / APB2PSC in RCU_CFG0 register is 0b0xx (CK_APBx = CK_AHB) or 0b100 (CK_APBx = CK_AHB / 2), the TIMER clock is equal to CK_AHB (CK_TIMERx = CK_AHB). Or else, the TIMER clock is twice the corresponding APB clock (TIMER in APB1 domain: CK_TIMERx = 2 x CK_APB1; TIMER in APB2 domain: CK_TIMERx = 2 x CK_APB2). 1: If APB1PSC / APB2PSC in RCU_CFG0 register is 0b0xx (CK_APBx = CK_AHB), 0b100 (CK_APBx = CK_AHB / 2), or 0b101 (CK_APBx = CK_AHB / 4), the TIMER clock is equal to CK_AHB (CK_TIMERx = CK_AHB). Or else, the TIMER clock is four times the corresponding APB clock (TIMER in APB1 domain: CK_TIMERx = 4 x CK_APB1, TIMER in APB2 domain: CK_TIMERx = 4 x CK_APB2).
23:22	Reserved	Must be kept at reset value.
21	LDO_ANA_LQB	Analog LDO current bias mode selection 0: Analog LDO high current bias mode. 1: Analog LDO low current bias mode.
20	LDO_CLK_LQB	Clock LDO current bias mode selection 0: Clock LDO high current bias mode. 1: Clock LDO low current bias mode.
19	BGPU	BandGap power on enable When PLLDIGEN is 1, this bit can’t be written 0.

		0: BandGap power down. 1: BandGap power on.
18	LDOCLKPU	LDO clock power on enable for RF / ADC 0: LDO clock power down. 1: LDO clock power on.
17	LDOANAPU	LDO analog power on enable for RF filter 0: LDO analog power down. 1: LDO analog power on.
16	RFPLLPU	RFPLL power on enable 0: RFPLL power down. 1: RFPLL power on.
15	RFPLLLOCK	RF PLL lock 0: RFPLL is not locked. 1: RFPLL is locked.
14	RFPLLCALEN	RF PLL calculation enable 0: RF PLL calculation disable. 1: RF PLL calculation enable.
13:12	Reserved	Must be kept at reset value.
11:9	BGVBIT[2:0]	BandGap Power adjust, which can't be written when HXTALEN or PLLDIGEN is on.
8:0	IRC16MDIV[8:0]	IRC16M clock divider factor for system clock It can't be written when system clock select IRC16M or IRC16MEN.. 00000000: IRC16M clock divided by 1. 00000001: IRC16M clock divided by 2. 00000010: IRC16M clock divided by 3. … 11111111: IRC16M clock divided by 512.

5.3.20. Additional clock control register (RCU_ADDCTL)

Address offset: 0x90

Reset value: 0x0000 0000

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

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

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

rw

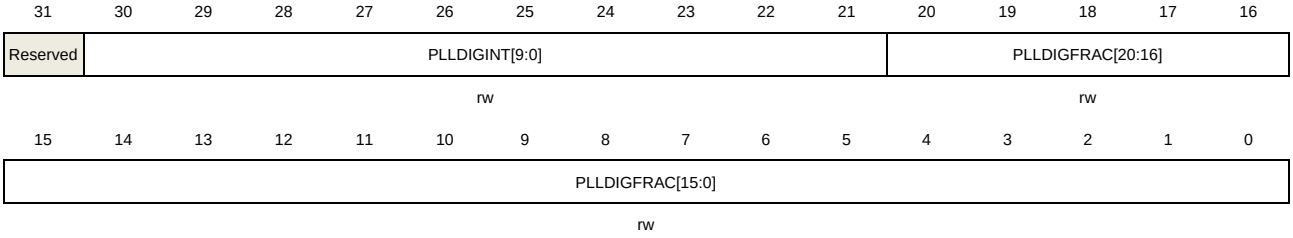
Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5:1	TRNGCKDIV[4:0]	PLLDIG clock divider factor for TRNG clock Set and reset by software to control the PLLDIG clock divider factor for TRNG clock. 00000: PLLDIG clock divided by 1 for TRNG clock. 00001: PLLDIG clock divided by 2 for TRNG clock. 00010: PLLDIG clock divided by 3 for TRNG clock. … 11111: PLLDIG clock divided by 32 for TRNG clock.
0	Reserved	Must be kept at reset value.

5.3.21. PLLDIG clock configuration register 1 (RCU_PLLDIGCFG1)

Address offset: 0x94
Reset value: 0x0780 0000

If use the WIFI, need to configure the RF clock source * frequency doubling factor = 960 MHz, to guarantee the WIFI function is normal. RF clock frequency should meet 960 MHz, RF clock source from HXTAL or IRC16M. PLLDIGINT assignment for the integer part of "960 MHz / RF clock source", PLLDIGFRAC assignment for fractional part of "960 MHz / RF clock source".

This register can be only accessed by word (32-bit)



Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:21	PLLDIGINT[9:0]	Control PLLDIG frequency doubling factor for integer part. **Note**: The value of PLLDIGINT[9:0] should be no less than 0x08.
20:0	PLLDIGFRAC[20:0]	Control PLLDIG frequency doubling factor for fractional part.

5.3.22. 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)

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

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

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

5.3.23. Deep-sleep mode voltage register (RCU_DSV)

Address offset: 0x134

Reset value: 0x0000 0000

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

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

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

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1:0	DSLPVS[1:0]	Deep-sleep mode voltage selection. These bits are set and reset by software. Deep-sleep mode voltage selection，LDO need to be configured to low drive mode. **Note**: If it is not configured to low drive mode, LDO cannot output voltage. 00: The core voltage is 1.1V in Deep-sleep mode. 01: The core voltage is 1.0V in Deep-sleep mode. 10: The core voltage is 0.9V in Deep-sleep mode. 11: The core voltage is 0.9V in Deep-sleep mode.

6. Interrupt / event controller (EXTI)

6.1. Overview

RISC-V integrates the Enhancement Core-Local Interrupt Controller (ECLIC) for efficient interrupts processing. ECLIC is designed to provide low-latency, vectored, pre-emptive interrupts for RISC-V systems. When activated, the ECLIC subsumes and replaces the existing RISC-V local interrupt scheme (CLINT). The CLIC design has a base design that requires minimal hardware, but supports additional extensions to provide hardware acceleration. The goal of the ECLIC design is to provide support for a variety of software ABI and interrupt models, without complex hardware that can impact high-performance processor implementations. It's tightly coupled to the processor core. You can read the Technical Reference Manual of RISC-V for more details about ECLIC.

EXTI (interrupt / event controller) contains up to 25 independent edge detectors and generates interrupt requests or events to the processor. The EXTI has three trigger types: rising edge, falling edge and both edges. Each edge detector in the EXTI can be configured and masked independently.

6.2. Characteristics

- Up to 75 maskable peripheral interrupts for GD32VW55x series.
- 4 bits interrupt priority configuration—16 priority levels.
- Support exception pre-emption and tail-chaining.
- Wake up system from power saving mode.
- Up to 25 independent edge detectors in EXTI.
- Three trigger types: rising, falling and both edges.
- Software interrupt or event trigger.
- Trigger sources configurable.

6.3. Function overview

The RISC-V processor and the Enhancement Core-Local Interrupt Controller (ECLIC) prioritize and handle all interrupts in machine mode.

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 interrupt types.

Table 6-1. Interrupt vector table

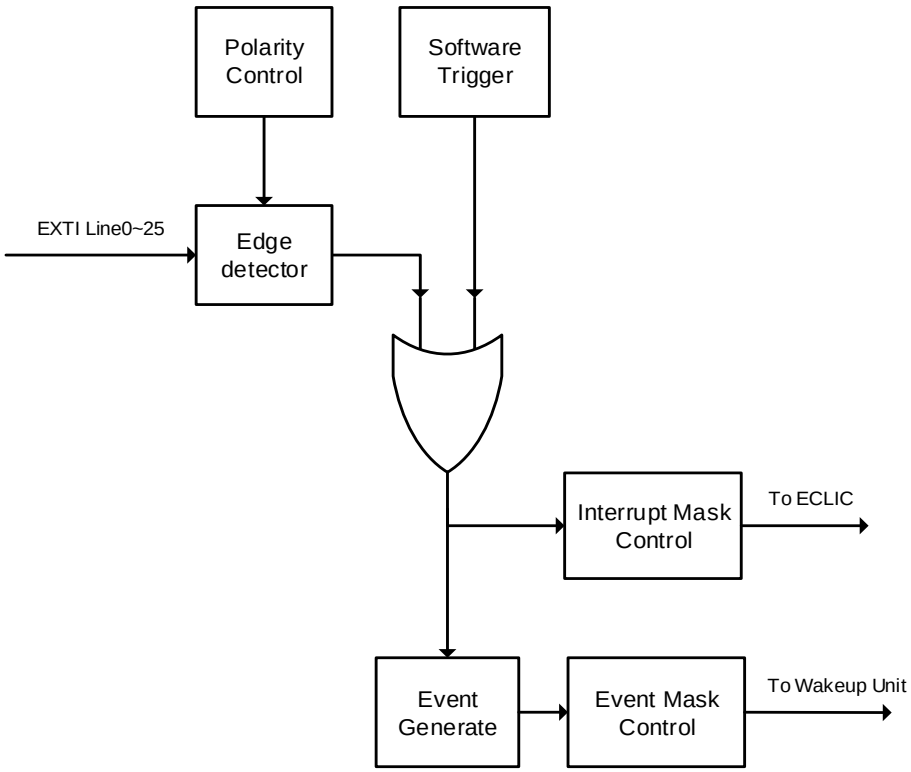
Vector number	Interrupt description	Vector address
3	CLIC_INT_SFT	0x0000_000C
7	CLIC_INT_TMR	0x0000_001C
19	WWDGT interrupt	0x0000_004C
20	LVD from EXTI interrupt	0x0000_0050
21	RTC tamper and timestamp from EXTI interrupt	0x0000_0054
22	RTC wakeup from EXTI interrupt	0x0000_0058
23	FMC global interrupt	0x0000_005C
24	RCU global interrupt	0x0000_0060
25	EXTI line0 interrupt	0x0000_0064
26	EXTI line1 interrupt	0x0000_0068
27	EXTI line2 interrupt	0x0000_006C
28	EXTI line3 interrupt	0x0000_0070
29	EXTI line4 interrupt	0x0000_0074
30	DMA channel0 global interrupt	0x0000_0078
31	DMA channel1 global interrupt	0x0000_007C
32	DMA channel2 global interrupt	0x0000_0080
33	DMA channel3 global interrupt	0x0000_0084
34	DMA channel4 global interrupt	0x0000_0088
35	DMA channel5 global interrupt	0x0000_008C
36	DMA channel6 global interrupt	0x0000_0090
37	DMA channel7 global interrupt	0x0000_0094
38	ADC interrupt	0x0000_0098
39~41	Reserved	0x0000_009C - 0x0000_00A4
42	EXTI line[5:9] interrupts	0x0000_00A8
43	TIMER0 Break interrupt	0x0000_00AC
44	TIMER0 update interrupt	0x0000_00B0
45	TIMER0 commutation and trigger interrupt	0x0000_00B4
46	TIMER0 capture compare interrupt	0x0000_00B8
47	TIMER1 global interrupt	0x0000_00BC
48	TIMER2 global interrupt	0x0000_00C0
49	Reserved	0x0000_00C4
50	I2C0 event interrupt	0x0000_00C8
51	I2C0 error interrupt	0x0000_00CC
52	I2C1 event interrupt	0x0000_00D0
53	I2C1 error interrupt	0x0000_00D4
54	SPI global interrupt	0x0000_00D8
55	Reserved	0x0000_00DC
56	USART0 global interrupt	0x0000_00E0

Vector number	Interrupt description	Vector address
57	UART1 global interrupt	0x0000_00E4
58	UART2 global interrupt	0x0000_00E8
59	EXTI line[10:15] interrupts	0x0000_00EC
60	RTC alarm from EXTI interrupt	0x0000_00F0
61~62	Reserved	0x0000_00F4 - 0x0000_00F8
63	TIMER15 global interrupt	0x0000_00FC
64	TIMER16 global interrupt	0x0000_0100
65~69	Reserved	0x0000_0104 - 0x0000_0114
70	I2C0 wakeup interrupt	0x0000_0118
71	USART0 wakeup interrupt	0x0000_011C
72	Reserved	0x0000_0120
73	TIMER5 global interrupt	0x0000_0124
74	WIFI protocol trigger interrupt	0x0000_0128
75	WIFI MAC interrupt	0x0000_012C
76	WIFI TX interrupt	0x0000_0130
77	WIFI RX interrupt	0x0000_0134
78~82	Reserved	0x0000_0138- 0x0000_0148
83	LA interrupt	0x0000_014C
84	WIFI wakeup from EXTI interrupt	0x0000_0150
85	BLE wakeup from EXTI interrupt	0x0000_0154
86	Platform(PLF) wakeup from EXTI interrupt	0x0000_0158
87~94	ISO bluetooth timestamp interrupt0~7	0x0000_015C- 0x0000_0178
95	PMU interrupt	0x0000_017C
96~97	Reserved	0x0000_0180- 0x0000_0184
98	CAU global interrupt	0x0000_0188
99	HAU / TRNG global interrupts	0x0000_018C
100	Reserved	0x0000_0190
101	WIFI interrupt	0x0000_0194
102	SW triggered interrupt	0x0000_0198
103	Fine timer target interrupt	0x0000_019C
104	Timestamp target 1 interrupt	0x0000_01A0
105	Timestamp target 2 interrupt	0x0000_01A4
106	Timestamp target 3 interrupt	0x0000_01A8
107	Encryption engine interrupt	0x0000_01AC
108	Sleep mode interrupt	0x0000_01B0

Vector number	Interrupt description	Vector address
109	Half slot interrupt	0x0000_01B4
110	FIFO activity interrupt	0x0000_01B8
111	Error interrupt	0x0000_01BC
112	Frequency selection interrupt	0x0000_01C0
113	EFUSE global interrupt	0x0000_01C4
114	QSPI global interrupt	0x0000_01C8
115	PKCAU global interrupt	0x0000_01CC

6.4. External interrupt and event block diagram

Figure 6-1. Block diagram of EXTI



6.5. External interrupt and event function overview

The EXTI contains up to 25 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 9 lines from internal modules which refers to ***Table 6-2. 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 RISC-V processor fully implements the Wait For Interrupt (WFI) instruction. EXTI can be used to wake up processor and the whole system when some expected event occurs, such as a special GPIO pin toggling or RTC alarm.

Hardware Trigger

Hardware trigger may be used to detect the voltage change of external or internal signals. The software should follow these steps to use this function:

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

Software Trigger

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

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

Table 6-2. EXTI source

EXTI line number	Source
0	PA0 / PB0
1	PA1 / PB1
2	PA2 / PB2
3	PA3 / PB3
4	PA4 / PB4
5	PA5
6	PA6
7	PA7
8	PA8 / PC8
9	PA9
10	PA10
11	PA11 / PB11
12	PA12 / PB12
13	PA13 / PB13 / PC13

EXTI line number	Source
14	PA14 / PC14
15	PA15 / PB15 / PC15
16	LVD
17	RTC alarm
18	Reserved
19	WIFI wakeup
20	RTC tamper and timestamp
21	RTC wakeup
22	I2C0 wakeup
23	USART0 wakeup
24	BLE wakeup
25	PLF wakeup

6.6. Register definition

EXTI base address: 0x4001 3C00

6.6.1. Interrupt enable register (EXTI_INTEN)

Address offset: 0x00
Reset value: 0x0000 0000

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						INTEN25	INTEN24	INTEN23	INTEN22	INTEN21	INTEN20	INTEN19	Reserved	INTEN17	INTEN16
						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
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.
25:19	INTENx	Interrupt enable bit x (x = 19…25) 0: Interrupt from linex is disabled 1: Interrupt from linex is enabled
18	Reserved	Must be kept at reset value.
17:0	INTENx	Interrupt enable bit x (x = 0…17) 0: Interrupt from linex is disabled 1: Interrupt from linex is enabled

6.6.2. 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	EVEN23	EVEN22	EVEN21	EVEN20	EVEN19	Reserved	EVEN17	EVEN16
						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
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.

25:19	EVENx	Event enable bit x (x = 19…25) 0: Event from linex is disabled 1: Event from linex is enabled
18	Reserved	Must be kept at reset value.
17:0	EVENx	Event enable bit x (x = 0…17) 0: Event from linex is disabled 1: Event from linex is enabled

6.6.3. 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	RTEN23	RTEN22	RTEN21	RTEN20	RTEN19	Reserved	RTEN17	RTEN16
						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
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.
25:19	RTENx	Rising edge trigger enable (x = 19…25) 0: Rising edge of linex is invalid 1: Rising edge of linex is valid as an interrupt / event request
18	Reserved	Must be kept at reset value.
17:0	RTENx	Rising edge trigger enable (x = 0…17) 0: Rising edge of linex is invalid 1: Rising edge of linex is valid as an interrupt / event request

6.6.4. 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	FTEN23	FTEN22	FTEN21	FTEN20	FTEN19	Reserved	FTEN17	FTEN16
						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

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.
25:19	FTENx	Falling edge trigger enable (x = 19...25) 0: Falling edge of linex is invalid 1: Falling edge of linex is valid as an interrupt / event request
18	Reserved	Must be kept at reset value.
17:0	FTENx	Falling edge trigger enable (x = 0...17) 0: Falling edge of linex is invalid 1: Falling edge of linex is valid as an interrupt / event request

6.6.5. 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	SWIEV23	SWIEV22	SWIEV21	SWIEV20	SWIEV19	Reserved	SWIEV17	SWIEV16
						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
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.
25:19	SWIEVx	Interrupt / event software trigger (x = 19...25) 0: Writing 0 has no effect. 1: Writing '1' to this bit when it is at “0” will trigger the EXTI linex software interrupt / event request. This bit is cleared by clearing the corresponding PDx bit in the EXTI_PD register
18	Reserved	Must be kept at reset value.
17:0	SWIEVx	Interrupt / event software trigger (x = 0...17) 0: Writing 0 has no effect. 1: Writing '1' to this bit when it is at “0” will trigger the EXTI linex software interrupt / event request. This bit is cleared by clearing the corresponding PDx bit in the EXTI_PD register

6.6.6. 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	PD23	PD22	PD21	PD20	PD19	Reserved	PD17	PD16
						rc_w1	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.
25:19	PDx	Interrupt pending status (x = 19…25) 0: EXTI linex is not triggered 1: EXTI linex is triggered This bit is cleared to 0 by writing 1 to it.
18	Reserved	Must be kept at reset value.
17:0	PDx	Interrupt pending status (x = 0…17) 0: EXTI linex is not triggered 1: EXTI linex is triggered This bit is cleared to 0 by writing 1 to it.

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

7.1. Overview

There are up to 29 general purpose I/O pins (GPIO), named PA0 ~ PA15, PB0 ~ PB4, PB11 ~ PB13, PB15, PC8 and PC13 ~ PC15 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 regardless of the AF input or output pins.

Each of the GPIO pins can be configured by software as output (push-pull or open-drain), input, peripheral alternate function or analog mode. Each GPIO pin can be configured as pull-up, pull-down or no pull-up / pull-down. All GPIOs are high-current capable except for analog mode.

7.2. Characteristics

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

7.3. 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 configuration registers (GPIOx_CTL). When select AF function, the pad input or output is decided by selected AF function output 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

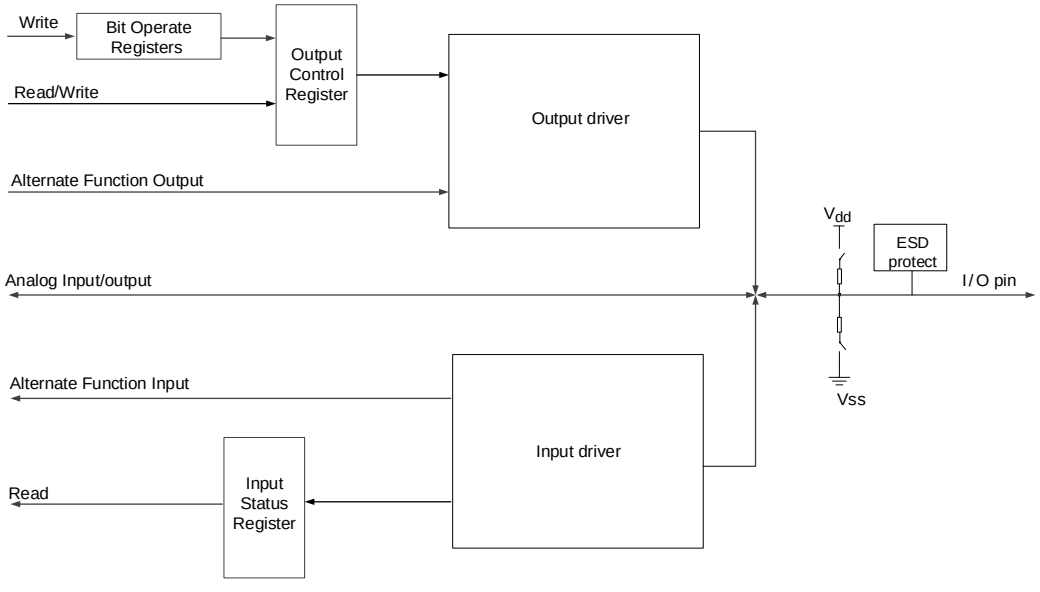
as floating (no pull-up and pull-down), pull-up or pull-down function by GPIO pull-up / pull-down registers (GPIOx_PUD).

Table 7-1. GPIO configuration table

PAD TYPE			CTLy	OMy	PUDy
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 7-1. Basic structure of a general-pupose I/O shows the basic structure of an I/O port bit.

Figure 7-1. Basic structure of a general-pupose I/O



7.3.1. GPIO pin configuration

During or just after the reset period, the alternative functions are all inactive and the GPIO ports are configured into the input floating mode that input disabled without Pull-Up(PU) / Pull-Down(PD) resistors. But the JTAG pins are in input PU / PD mode after reset:

PA15: JTDI in PU mode
PA14: JTCK in PD mode
PA13: JTMS in PU mode
PB4: NJTRST in PU mode
PB3: JTDI in Floating mode

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

When the GPIO pins are configured as output pins, 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 bit level, the user can modify only one 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.

7.3.2. External interrupt / event lines

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

7.3.3. 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 can be configured by GPIO alternate functions selected registers (GPIOx_AFSELz (z = 0,1)). The detail alternate function assignments for each port are in the device datasheet.

7.3.4. Additional functions

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

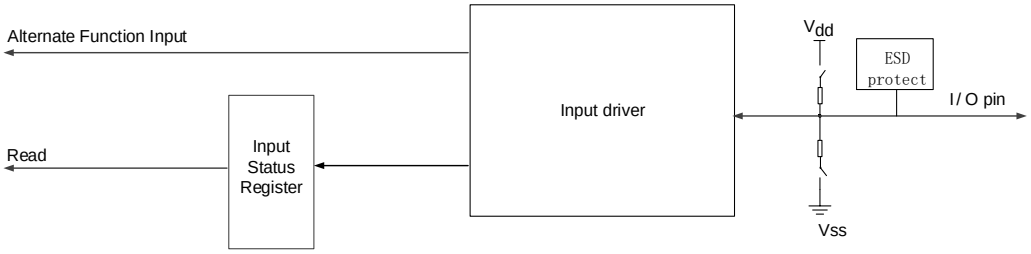
7.3.5. Input configuration

When GPIO pin is configured as Input:

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

Figure 7-2. Basic structure of Input configuration shows the input configuration.

Figure 7-2. Basic structure of Input configuration



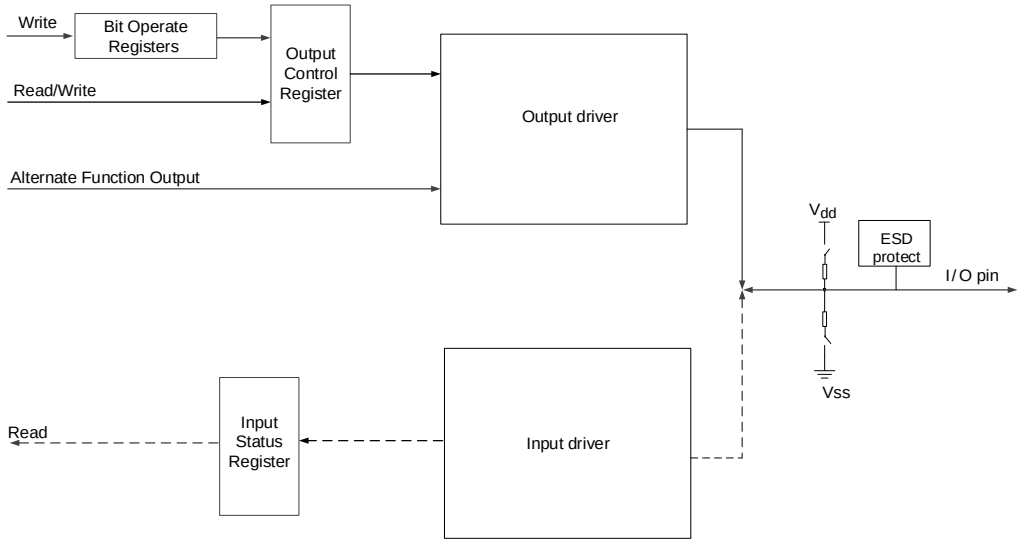
7.3.6. Output configuration

When GPIO pin is configured as output:

- The schmitt trigger input is enabled.
- The weak pull-up and pull-down resistors could be chosen.
- The output buffer is enabled.
- Open Drain Mode: The pad output low level 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 output low level when a “0” in the output control register; while the pad output high level 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 7-3. Basic structure of Output configuration shows the output configuration.

Figure 7-3. Basic structure of Output configuration



7.3.7. 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 disabled
- The port input status register of this I/O port bit is “0”

Figure 7-4. Basic structure of Analog configuration shows the analog configuration.

Figure 7-4. Basic structure of Analog configuration

7.3.8. Alternate function (AF) configuration

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

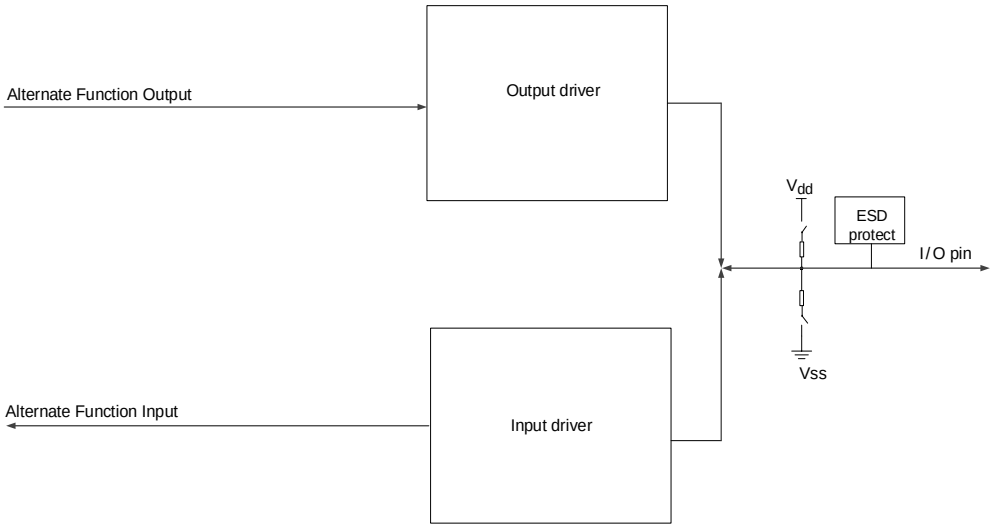
When be configured as alternate function:

- 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 enabled.
- The weak pull-up and pull-down resistors could be chosen.
- The I/O pin data is stored into the port input status register every AHB clock.
- A read access to the port input status register gets the I/O state.

- A read access to the port output control register gets the last written value.

Figure 7-5. Basic structure of Alternate function configuration shows the alternate function configuration.

Figure 7-5. Basic structure of Alternate function configuration



7.3.9. 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 and GPIOx_AFSELz (z=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 occurred on LKK bit in GPIOx_LOCK register and the LKy bit is set in GPIOx_LOCK register, the corresponding port is locked and the corresponding port configuration cannot be modified until the next reset. It recommended to be used in the configuration of driving a power module.

7.3.10. GPIO I/O compensation cell

By default, the I/O compensation cell is not used. However, when the I/O port output speed is more than 50MHz, it is recommended to use the compensation cell for slew rate control to reduce the I/O noise on power supply.

When the compensation cell is enabled, a complete flag CPS_RDY is set to indicate that the compensation cell is ready and can be used.

7.3.11. 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.

7.4. Register definition

GPIOA base address: 0x4002 0000
GPIOB base address: 0x4002 0400
GPIOC base address: 0x4002 0800

7.4.1. Port control register (GPIOx_CTL, x = A…C)

Address offset: 0x00
Reset value: GPIOA_CTL 0xA800 0000
GPIOB_CTL 0x0000 0280
GPIOC_CTL 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read 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: GPIO Input mode (reset value) 01: GPIO output mode 10: Alternate function mode. 11: Analog mode (Input and Output)

7.4.2. Port output mode register (GPIOx_OMODE, x = A...C)

Address offset: 0x04

Reset value: 0x0000 0000

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

This register can only be read 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 (reset value) 1: Output open-drain mode

7.4.3. Port output speed register (GPIOx_OSPD, x = A...C)

Address offset: 0x08

Reset value: GPIOA_OSPD 0x0C00 0000
GPIOB_OSPD 0x0000 00C0
GPIOC_OSPD 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read 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. 00: Output speed level 0 (reset state) 01: Output speed level 1 10: Output speed level 2 11: Output speed level 3

7.4.4. Port pull-up/pull-down register (GPIOx_PUD, x = A…C)

Address offset: 0x0C

Reset value: GPIOA_PUD 0x6400 0000
GPIOB_PUD 0x0000 0100
GPIOC_PUD 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read 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

7.4.5. Port input status register (GPIOx_ISTAT, x = A...C)

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	Pin input status (y = 0…15) These bits are set and cleared by hardware. 0: Input signal low 1: Input signal high

7.4.6. Port output control register (GPIOx_OCTL, x = A...C)

Address offset: 0x14
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read 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	Pin output control (y = 0…15) These bits are set and cleared by software. 0: Pin output low 1: Pin output high

7.4.7. Port bit operate register (GPIOx_BOP, x = A…C)

Address offset: 0x18
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit), half-word (16-bit), byte (8-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	Pin Clear bit y (y = 0...15) These bits are set and cleared by software. 0: No action on the corresponding OCTLy bit 1: Clear the corresponding OCTLy bit to 0
15:0	BOPy	Pin Set bit y (y = 0…15) These bits are set and cleared by software. 0: No action on the corresponding OCTLy bit 1: Set the corresponding OCTLy bit to 1

7.4.8. Port configuration lock register (GPIOx_LOCK, x = A…C)

Address offset: 0x1C
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															LKK
															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LK15	LK14	LK13	LK12	LK11	LK10	LK9	LK8	LK7	LK6	LK5	LK4	LK3	LK2	LK1	LK0

rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	LKK	Lock sequence key It can only be setted using the Lock Key Writing Sequence. And can always be read. 0: GPIO_LOCK register is not locked and the port configuration is not locked. 1: GPIO_LOCK register is locked until an MCU reset. LOCK key configuration sequence Write 1→Write 0→Write 1→Read 0→Read 1 **Note:** The value of LK[15:0] must hold during the LOCK Key Writing sequence
15:0	LKy	Port Lock bit y(y = 0...15) These bits are set and cleared by software. 0: The corresponding bit port configuration is not locked 1: The corresponding bit port configuration is locked when LKK bit is “1”

7.4.9. Alternate function selected register 0 (GPIOx_AFSEL0, x = A...C)

Address offset: 0x20
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read 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 … 1111: AF15 selected

7.4.10. Alternate function selected register 1 (GPIOx_AFSEL1, x = A…C)

Address offset: 0x24

Reset value: 0x0000 0000

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

This register can only be read 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 11 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 … 1111: AF15 selected

7.4.11. Bit clear register (GPIOx_BC, x = A...C)

Address offset: 0x28

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit), half-word (16-bit), byte (8-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	Pin 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

7.4.12. Port bit toggle register (GPIOx_TG, x = A…C)

Address offset: 0x2C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit), half-word (16-bit), byte (8-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	Pin 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. Cyclic redundancy checks management unit (CRC)

8.1. Overview

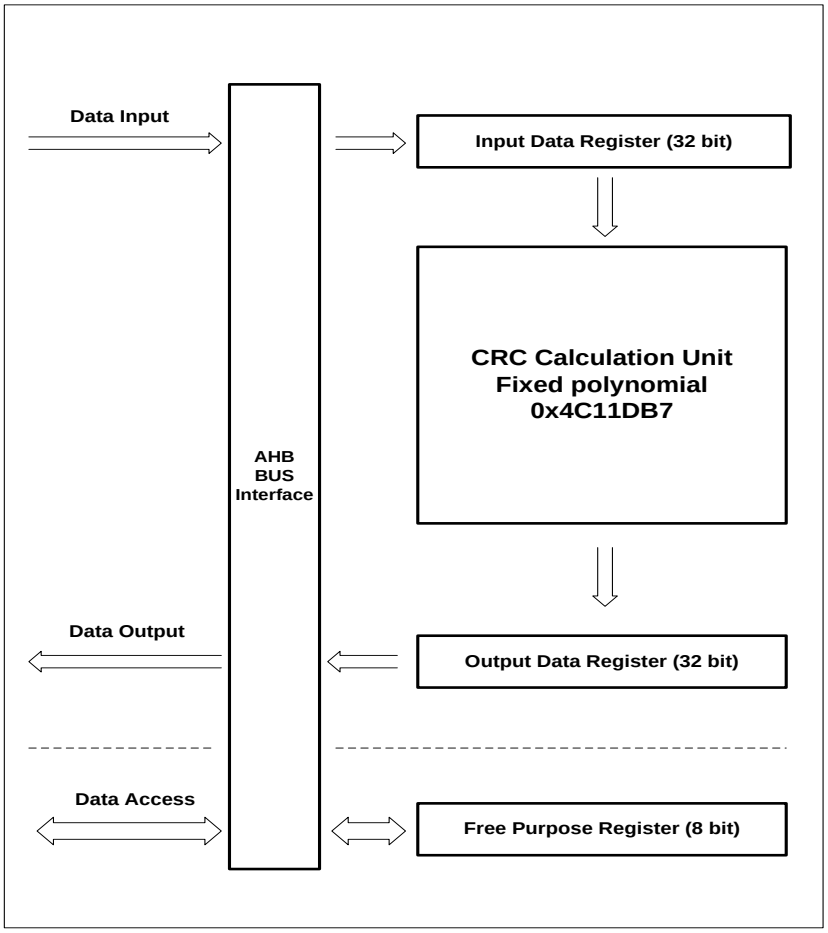
A cyclic redundancy check (CRC) is an error-detecting code commonly used in digital networks and storage devices to detect accidental changes to raw data.

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

8.2. Characteristics

- 32-bit data input and 32-bit data output. Calculation period is 4 AHB clock cycles for 32-bit input data size from data entered to the calculation result available.
- Free 8-bit register is unrelated to calculation and can be used for any other goals by any other peripheral devices.
- Fixed polynomial: 0x4C11DB7
  $X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^{8}+X^{7}+X^{5}+X^{4}+X^{2}+X+1$
  This 32-bit CRC polynomial is a common polynomial used in Ethernet.

Figure 8-1. Block diagram of CRC calculation unit



8.3. Function overview

- CRC calculation unit is used to calculate the 32-bit raw data, and CRC_DATA register will receive the raw data and store the calculation result.
  - If the CRC_DATA register has not been cleared by software setting the CRC_CTL register, the new input raw data will be calculated based on the result of previous value of CRC_DATA.
  - During CRC calculation AHB will not be hanged because of the existence of the 32-bit input buffer.
- This module supplies an 8-bit free register CRC_FDATA.
  - CRC_FDATA is unrelated to the CRC calculation, any value you write in will be read out at anytime.

8.4. Register definition

CRC access base address: 0x4002 3000

8.4.1. Data register (CRC_DATA)

Address offset: 0x00
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
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]	CRC calculation result bits Software writes and reads. This register is used to calculate new data, and the register can be written the new data directly. Written value cannot be read because the read value is the previous CRC calculation result.

8.4.2. Free data register (CRC_FDATA)

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								FDATA[7:0]							
								rw							

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

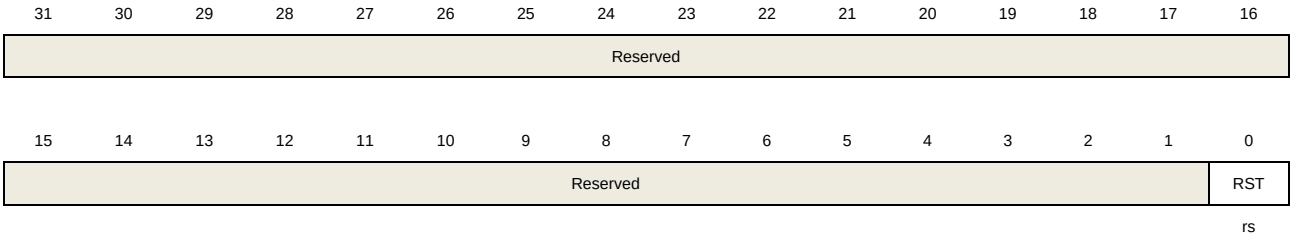
by any other peripheral. The CRC_CTL register will take no effect to the byte.

8.4.3. Control register (CRC_CTL)

Address offset: 0x08

Reset value: 0x0000 0000

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



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

9. True random number generator (TRNG)

9.1. Overview

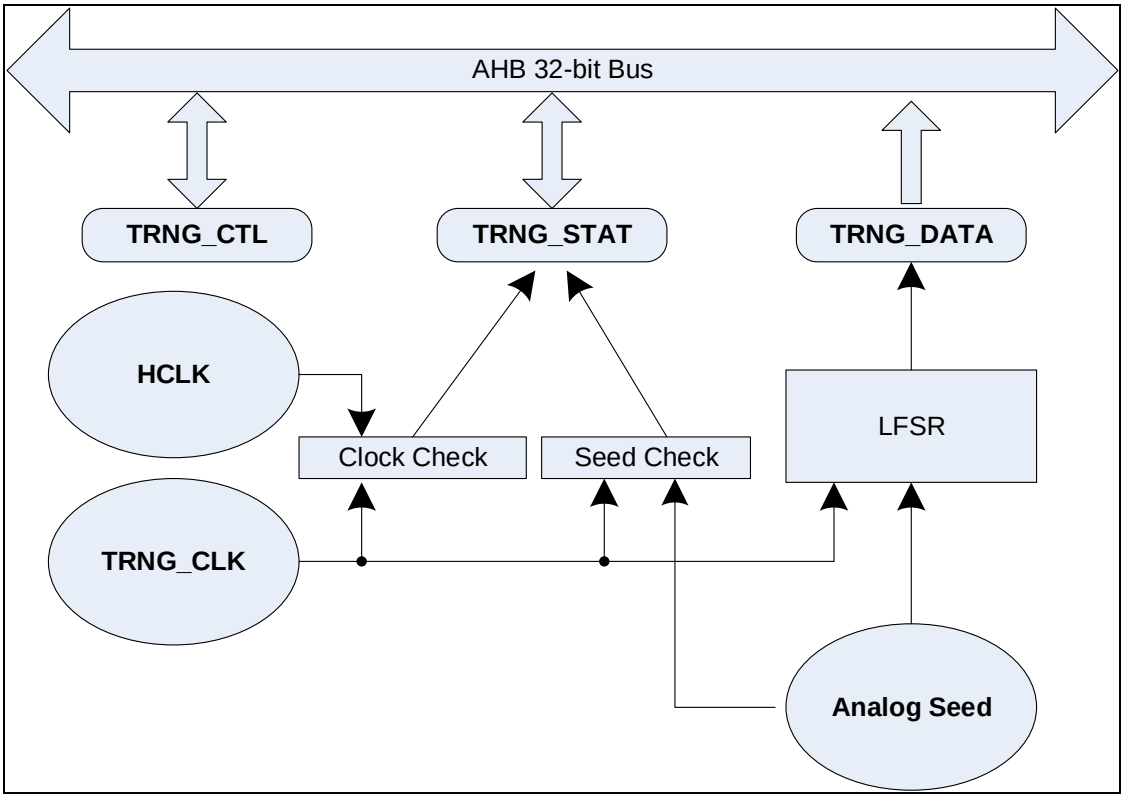
The true random number generator (TRNG) module can generate a 32-bit random value by using continuous analog noise.

9.2. Characteristics

- About 40 periods of TRNG_CLK are needed between two consecutive random numbers.
- Disable TRNG module will significantly reduce the chip power consumption.
- 32-bit random value seed is generated from analog noise, so the random number is a true random number.

9.3. Function overview

Figure 9-1. TRNG block diagram



The random number seed comes from analog circuit. This analog seed is then plugged into a linear feedback shift register (LFSR), where a 32-bit width random number is generated.

The analog seed is generated by several ring oscillators. The LFSR is driven by a configurable TRNG_CLK (refer to ***Reset and clock unit (RCU)*** chapter), so that the quality of the generated random number depends on TRNG_CLK exclusively, no matter what HCLK frequency was set or not.

The 32-bit value of LFSR will transfer into TRNG_DATA register after a sufficient number of seeds have been sent to the LFSR.

At the same time, the analog seed and TRNG_CLK clock are monitored. When an analog seed error or a clock error occurs, the corresponding status bit in TRNG_STAT will be set and an interrupt will generate if the IE bit in TRNG_CTL is set.

9.3.1. Operation flow

The following steps are recommended for using TRNG block:

- Enable the interrupt as necessary, so that when a random number or an error occurs, an interrupt will be generated.
- Enable the TRNGEN bit.
- When an interrupt occurs, check the status register TRGN_STAT, if SEIF=0, CEIF=0 and DRDY=1, then the random value in the data register could be read.

As required by the FIPS PUB 140-2, the first random data in data register should be saved but not be used. Every subsequent new random data should be compared to the previously random data. The data can only be used if it is not equal to the previously one.

9.3.2. Error flags

(1) Clock error

When the TRNG_CLK frequency is lower than the 1/16 of HCLK, the CECS and CEIF bit will be set. In this case, the application should check TRNG_CLK and HCLK frequency configurations and then clear CEIF bit. Clock error will not impact the previous random data.

(2) Seed error

When the analog seed is not changed or always changing during 64 TRNG_CLK periods, the SECS and SEIF bit will be set. In this case, the random data in data register should not be used. The application needs to clear the SEIF bit, then clear and set TRNGEN bit for restarting the TRNG.

9.4. Register definition

TRNG base address: 0x4C06 0800

9.4.1. Control register (TRNG_CTL)

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												IE	TRNGEN	Reserved	
												rw	rw		

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	IE	Interrupt enabled bit. This bit controls the generation of an interrupt when DRDY, SEIF or CEIF was set. 0: disable TRNG interrupt 1: enable TRNG interrupt
2	TRNGEN	TRNG enabled bit. 0: disable TRNG module (reduce power consuming) 1: enable TRNG module
1:0	Reserved	Must be kept at reset value.

9.4.2. Status register (TRNG_STAT)

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									SEIF	CEIF	Reserved		SECS	CECS	DRDY
									rc_w0	rc_w0			r	r	r

Bits	Fields	Descriptions

31:7	Reserved	Must be kept at reset value.
6	SEIF	Seed error interrupt flag This bit will be set if more than 64 consecutive same bit or more than 32 consecutive 01(or 10) changing are detected. 0: No fault detected 1: Seed error has been detected. The bit is cleared by writing 0.
5	CEIF	Clock error interrupt flag This bit will be set if TRNG_CLK frequency is lower than 1/16 HCLK frequency. 0: No fault detected 1: Clock error has been detected. The bit is cleared by writing 0.
4:3	Reserved	Must be kept at reset value.
2	SECS	Seed error current status 0: Seed error is not detected at current time. In case of SEIF=1 and SECS=0, it means seed error has been detected before but now is recovered. 1: Seed error is detected at current time if more than 64 consecutive same bits or more than 32 consecutive 01(or 10) changing are detected.
1	CECS	Clock error current status 0: Clock error is not detected at current time. In case of CEIF=1 and CECS=0, it means clock error has been detected before but now is recovered. 1: Clock error is detected at current time. TRNG_CLK frequency is lower than 1/16 HCLK frequency.
0	DRDY	Random data ready status bit. This bit is cleared by reading the TRNG_DATA register and set when a new random number is generated. 0: The content of TRNG data register is not available. 1: The content of TRNG data register is available.

9.4.3. Data register (TRNG_DATA)

Address offset: 0x08
Reset value: 0x0000 0000

Application must make sure DRDY is set before reading this register.

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

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

Bits	Fields	Descriptions
31:0	TRNDATA[31:0]	32-bit random data

10. Direct memory access controller (DMA)

10.1. Overview

The direct memory access (DMA) controller provides a hardware method of transferring data between peripherals and/or memory without intervention from the MCU, thereby increasing system performance by off-loading the MCU from copying large amounts of data and avoiding frequent interrupts to serve peripherals needing more data or having available data.

Two AHB master interfaces and eight four-word depth 32-bit width FIFOs are presented in DMA controller, which achieves a high DMA transmission performance. There are 8 independent channels in the DMA controller. Each channel is assigned a specific or multiple target peripheral devices for memory access request management. Two arbiters respectively for memory and peripheral are implemented inside to handle the priority among DMA requests.

Both the DMA controller and the RISC-V core implement data access through the system bus. An arbitration mechanism is implemented to solve the competition between these two masters. When the same peripheral is targeted, the MCU access will be suspended for some specific bus cycles. A round-robin scheduling algorithm is utilized in the bus matrix to guaranty at least half the bandwidth to the MCU.

10.2. Characteristics

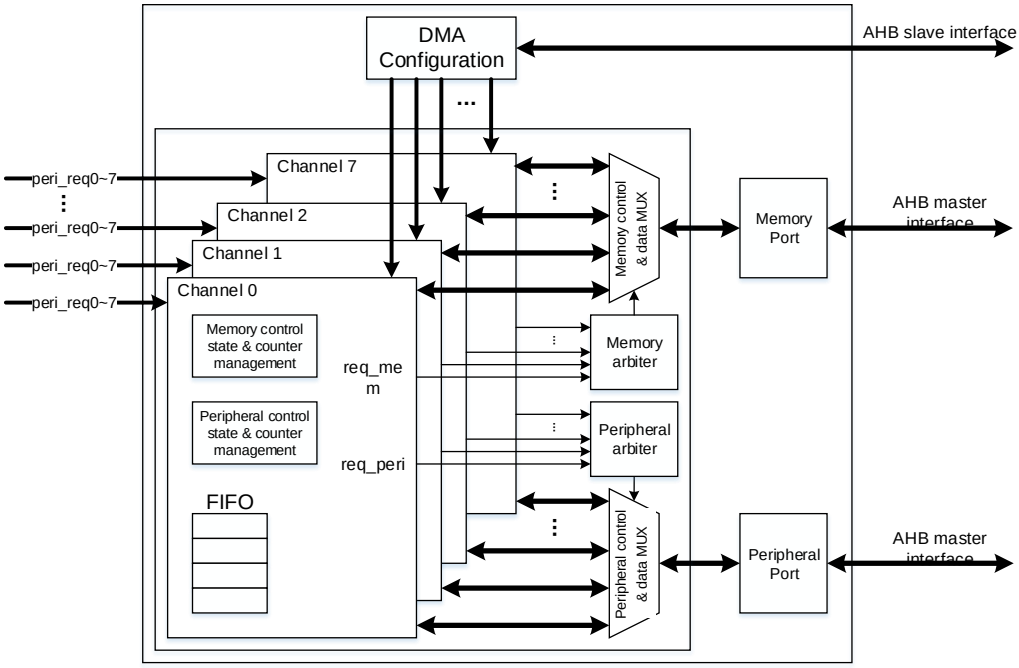
- Two AHB master interface for transferring data, and one AHB slave interface for programming DMA.
- 8 channels for DMA, up to 8 peripherals per channel with fixed hardware peripheral requests.
- Support independent single, 4, 8, 16-beat incrementing burst memory and peripheral transfer.
- Support switch-buffer transmission between peripheral and memory.
- Software DMA channel priority (low, medium, high, ultra high) and hardware DMA channel priority (DMA channel 0 has the highest priority and DMA channel 7 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 three transfer modes:
  - Read from memory and write to peripheral.
  - Read from peripheral and write to memory.
  - Read from memory and write to memory.
- Both DMA and peripheral can be configured as flow controller:

- DMA: Programmable length of data to be transferred, max to 65535.
- Peripheral: The last request signal given to DMA from peripheral determines the end of transfer.

- Support two data processing modes by use of the four-word depth 32-bit width FIFOs:
  - Multi-data mode: Pack/Unpack data when memory transfer width are different from peripheral transfer width.
  - Single-data mode: Read data from source when FIFO is empty and write data to destination when one data has been pushed into FIFO.
- One separate interrupt per channel with five types of event flags.
- Support interrupt enable and clear.

10.3. Block diagram

Figure 10-1. Block diagram of DMA



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

- DMA configuration through AHB slave interface.
- Data access through two AHB master interfaces respectively for memory access and peripheral access.
- Two arbiters inside to manage multiple peripheral requests coming at the same time.
- Channel data management to control data packing / unpacking and counting.

10.4. Function overview

The DMA controller transfers data from one address to another without CPU intervention. It

supports multiple data sizes, burst types, address generation algorithm, priority levels and several transfer modes to allow for flexible application by configuring the corresponding bits in DMA registers. All the DMA registers can be 32-bit configured through AHB slave interface.

Three transfer modes are supported, including peripheral-to-memory, memory-to-peripheral and memory-to-memory, which is determined by the TM bits in the DMA_CHxCTL register, as listed in ***Table 10-1. Transfer mode***.

Table 10-1. Transfer mode

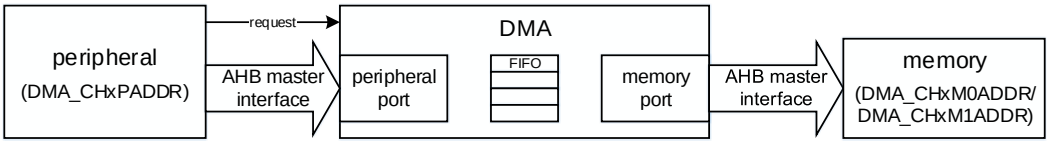
Transfer mode	TM[1:0]	Source	Destination
Peripheral to memory	00	DMA_CHxPADDR	DMA_CHxM0ADDR / DMA_CHxM1ADDR1
Memory to peripheral	01	DMA_CHxM0ADDR / DMA_CHxM1ADDR	DMA_CHxPADDR
Memory to memory	10	DMA_CHxPADDR	DMA_CHxM0ADDR / DMA_CHxM1ADDR

Note:

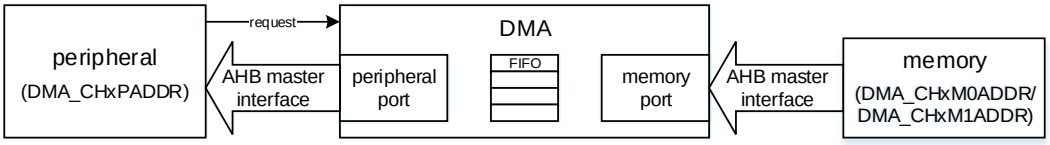
1. The MBS bit in DMA_CHxCTL register determines which is selected as the memory buffer address in DMA_CHxM0ADDR and DMA_CHxM1ADDR register. For more information, refer to section ***Switch-buffer mode***.
2. The TM bits in DMA_CHxCTL register are forbidden to be configured to 0b11, or the channel will be automatically disabled.

Figure 10-2. Data stream for three transfer modes

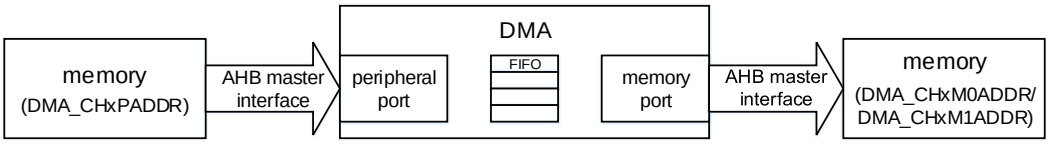
- peripheral-to-memory



- memory-to-peripheral



- memory-to-memory



As shown in ***2. The TM bits in DMA_CHxCTL register*** are forbidden to be configured to 0b11, or the channel will be automatically disabled.

Figure 10-2. Data stream for three transfer modes, Two AHB master interfaces are implemented in each DMA respectively for memory and peripheral.

- Memory to peripheral: read data from memory through AHB master interface for memory, and write data to peripheral through AHB master interface for peripheral.
- Peripheral to memory: read data from peripheral through AHB master interface for peripheral, and write data to memory through AHB master interface for memory.
- Memory to memory: read data from memory through AHB master interface for peripheral, and write data to another memory through AHB master interface for memory.

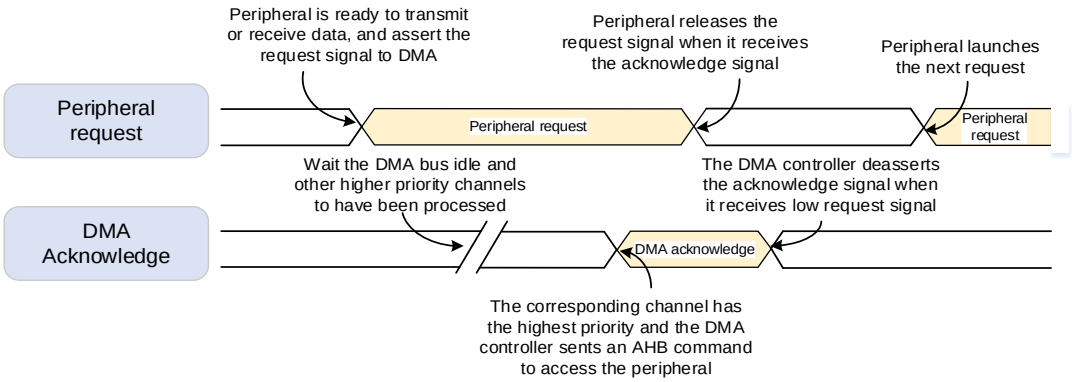
10.4.1. Peripheral handshake

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

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

Figure 10-3. Handshake mechanism shows how the handshake mechanism works between the DMA controller and peripherals.

Figure 10-3. Handshake mechanism



DMA has 8 channels, with fixed multiple peripheral requests. The PERIEN bits in the DMA_CHxCTL register determine which peripheral request signal connects to each DMA channel. The peripheral requests mapping of DMA is listed in *Table 10-2. Peripheral requests to DMA*.

As listed in the *Table 10-2. Peripheral requests to DMA*, a peripheral request can be connected to two different DMA channels. It is forbidden to simultaneously enable these two DMA channels with selecting the same peripheral request. For example, I2C0_RX is connected to channel 0 and channel 5. When the PERIEN bits in the DMA_CH0CTL register are configured to 0b001 and the PERIEN bits in the DMA_CH5CTL register are configured to 0b001, enable these two channels and responsing the request from I2C0 at the same time will cause transmission error.

Table 10-2. Peripheral requests to DMA

Channel		Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
PERIEN[2:0]	000	ADC	TIMER1_TG	TIMER2_TG	TIMER2_CH1	ADC	•	TIMER0_CH0 TIMER0_CH1 TIMER0_CH2	•
	001	I2C0_RX	•	TIMER2_UP	TIMER2_CH2	•	I2C0_RX	I2C0_TX	I2C0_TX
	010	•	•	TIMER2_CH0	TIMER2_CH3	•	CAU_OUT	CAU_IN	HAU_IN
	011	SPI_RX	TIMER1_CH2 TIMER1_UP	SPI_RX	SPI_TX	SPI_TX	TIMER1_CH0	TIMER1_CH1 TIMER1_CH3	TIMER1_UP TIMER1_CH3
	100	UART1_RX	UART1_TX	USART0_RX	•	USART0_RX	UART2_RX	UART2_TX	USART0_TX
	101	QSPI	QSPI	TIMER15_CH0 TIMER15_UP	TIMER16_CH0 TIMER16_UP	•	•	TIMER15_CH0 TIMER15_UP	TIMER16_CH0 TIMER16_UP
	110	TIMER0_TG	TIMER0_CH0	TIMER0_CH1	TIMER0_CH0	TIMER0_CH3 TIMER0_TG TIMER0_CMT	TIMER0_UP	TIMER0_CH2	•
	111	•	TIMER5_UP	I2C1_RX	I2C1_RX	•	•	•	I2C1_TX

10.4.2. Data process

Arbitration

Two arbiters are implemented in DMA respectively for memory and peripheral port. When two or more requests are received at the same time, 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.

Transfer width, burst and counter

Transfer width

PWIDTH and MWIDTH in the DMA_CHxCTL register indicate the data width of a peripheral and memory transfer seperately. The DMA supports 8-bit, 16-bit and 32-bit transfer width. In multi-data mode, if PWIDTH is not equal to MWIDTH, the DMA can automatically packs / unpacks data to achieve an integrated and correct data transfer operation. In single-data mode, MWIDTH is automatically locked as PWIDTH by hardware immediately after enable the DMA channel.

Transfer burst type

PBURST and MBURST in the DMA_CHxCTL register indicate the burst type of a peripheral and memory transfer seperately. The DMA supports single burst, 4-beat, 8-beat and 16-beat

incrementing burst for peripheral port and memory port. In single-data mode, only single burst type is supported and PBURST and MBURST are automatically locked as '00' by hardware immediately after enable the DMA channel.

In peripheral-to-memory or memory-to-peripheral mode, if PBURST is different from '00', DMA responses a increasing burst transfer of 4, 8, 16-beat based on the PBURST bits for each peripheral request. If the remaining bytes number of data item to be transferred is less than the bytes number needed for a burst transfer, the remaining data items are transferred in single transaction.

AMBA protocol specifies that bursts must not cross a 1KB address boundary, or else a transfer error will be responsed to the master. In the DMA, the peripheral burst transfer crossing a 1KB address boundary is decomposed to 4, 8 or 16 single transactions depend on the PBURST bits, as the same as the memory burst transfer.

Transfer counter

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. If the peripheral is configured as the flow controller, the CNT bits are forced to '0xFFFF' immediately after enabling the channel whatever the CNT bits are. During the transmission, the CNT bits indicate the remaining number of data items to be transferred.

The CNT bits are related to peripheral transfer width, the number of data bytes to be transferred is the CNT bits multiplied by the byte number of the peripheral transfer width. For example, if the PWIDTH bits are equal to '11', and the number of data bytes to be transferred is CNT×4. The CNT bits is decreased by 1 when a single or a beat of the burst peripheral transfer (the source memory transfer in the memory-to-memory mode) has been completed even if the transfer mode is peripheral-to-memory or memory-to-memory.

When configuring the CNT bits, the following rules must be respected to guarantee a good DMA operation:

If the circular mode is disabled by clearing the CMEN bit in the DMA_CHxCTL register, the rules to configure the CNT bits in the DMA_CHxCNT register based on the transfer width are listed in the ***Table 10-3. CNT configuration***.

The number of data bytes must be an integer multiple of the memory transfer width to guarantee an integrated single memory transfer.

Note: The number of data bytes does not need to be an interger multiple of the bytes number of a memory burst transfer or a peripheral burst number if the PBURST or/and MBURST bits are not equal to '00'. The remaining data not enough for a burst transfer are transferred can be divided into single transaction automatically.

Table 10-3. CNT configuration

PWIDTH	MWIDTH	CNT
8-bit	16-bit	Multiple of 2

8-bit	32-bit	Multiple of 4
16-bit	32-bit	Multiple of 2
Others		Any value

If the circular mode is enabled by setting the CMEN bit in the DMA_CHxCTL register. The number of data bytes must be an integer multiple of the byte number of a peripheral burst transfer and a memory burst transfer to gurantee an integrated memory and peripheral burst transfer:

$CNT / PBURST_beats$ must be an integer.

$(CNT \times PWIDTH_bytes) / (MBURST_beats \times MWIDTH_bytes)$ must be an integer.

in the above formula:

- PWIDTH_bytes is the byte number of the peripheral transfer width, 1 for 8-bit, 2 for 16-bit and 4 for 32-bit.
- PBURST_beats is the beat number of a peripheral burst transfer, 1 for single burst, 4 for INCR4, 8 for INCR8 and 16 for INCR16.
- MWIDTH_bytes is the byte number of the peripheral transfer width, 1 for 8-bit, 2 for 16-bit and 4 for 32-bit.
- MBURST_beats is the beat number of a peripheral burst transfer, 1 for single burst, 4 for INCR4, 8 for INCR8 and 16 for INCR16.

For example:

1. If PWIDTH is 16-bit, PBURST is INCR4, MWIDTH is 8-bit and MBURST is INCR16, $CNT/4$ and $(CNT\times2)/(1\times16)$ must be an integer, so the CNT bits must be configured to the multiple of 8.
2. If the If PWIDTH is 8-bit, PBURST is INCR16, MWIDTH is 16-bit and MBURST is INCR4, $CNT/16$ and $(CNT\times1)/(2\times4)$ must be an integer, so the CNT bits must be configured to the multiple of 16.

Note: When the switch-buffer mode is enabled by setting the SBMEN bit in the DMA_CHxCTL register, the circular mode is enabled automatically by hardware, and the above rules must also be respected.

FIFO

A four-word depth 32-bit FIFO is implemented as a data buffer for each DMA channel. Data reading from the source address is stored in the FIFO temporarily and transmitted to the destination through the destination port. Two data processing modes are supported depend on the FIFO configuration, including single-data mode and multi-data mode. When the transfer mode is memory-to-memory, only multi-data mode is supported to implement the DMA data processing.

Multi-data mode

The multi-data mode is selected by configuring the MDMEN bit in the DMA_CHxFCTL register to '1'.

In this mode, the DMA responds the source request when there is enough FIFO space for a source transfer, pushing the data reading from the source address into the FIFO. If the destination is a peripheral, the DMA responds the peripheral request when there is enough FIFO data for a peripheral burst transfer. If the memory is configured as the destination, the FIFO counter critical value configured in the FCCV bits of the DMA_CHxFCTL register controls the memory data processing. Only when the FIFO counter is reached the critical value, the data in the FIFO are entirely popped and written into the memory address.

To gurantee a good DMA behavior, the FIFO counter critical value (FCCV bits in the DMA_CHxFCTL register) must be an integer multiple of a memory burst transfer to ensure there is enough data for memory burst transfers. The configuration rules of the FIFO counter critical value depending on memory transfer width and memory burst types are listed in ***Table 10-4. FIFO counter critical value configuration rules***.

Table 10-4. FIFO counter critical value configuration rules

MWIDTH	MBURST	FIFO counter critical value			
		1-word	2-word	3-word	4-word
8-bit	single	4 single transactions	8 single transactions	12 single transactions	16 single transactions
	INCR4	1 burst transaction	2 burst transactions	3 burst transactions	4 burst transactions
	INCR8	ERROR	1 burst transaction	ERROR	2 burst transactions
	INCR16	ERROR	ERROR	ERROR	1 burst transaction
16-bit	single	2 single transactions	4 single transactions	6 single transactions	8 single transactions
	INCR4	ERROR	1 burst transaction	ERROR	2 burst transactions
	INCR8	ERROR	ERROR	ERROR	1 burst transaction
	INCR16	ERROR	ERROR	ERROR	ERROR
32-bit	single	1 single transaction	2 single transactions	3 single transactions	4 single transactions
	INCR4	ERROR	ERROR	ERROR	1 burst transactions
	INCR8	ERROR	ERROR	ERROR	ERROR
	INCR16	ERROR	ERROR	ERROR	ERROR

Note: When the transfer mode is peripheral-to-memory, if the PBURST_beats×PWIDTH_bytes = 16, the FIFO counter critical value must not be equal to '10'. When receiving a peripheral request, DMA initiates a peripheral burst transfer to entirely fill the FIFO. Then DMA lanches memory burst transfers to pop three words from the FIFO depending on the FIFO counter critical value and a word is still remained in the FIFO. There is no enough space for a peripheral burst transfer and the FIFO counter critical value is not reached, which make DMA transfer frozen.

Single-data mode

The single-data mode is selected by configuring the MDMEN bit in the DMA_CHxFCTL register to '0'. In this mode, only single transfer is supported to implement the DMA data access, and the FIFO counter critical value configured in the FCCV bits of the DMA_CHxFCTL register has no meaning.

In single-data mode, DMA responds the source request only when the FIFO is empty, pushing the data reading from the source address into the FIFO whatever the source transfer width is. When the FIFO is not empty, DMA responds the destination request, popping the data from the FIFO and writing it to the destination address.

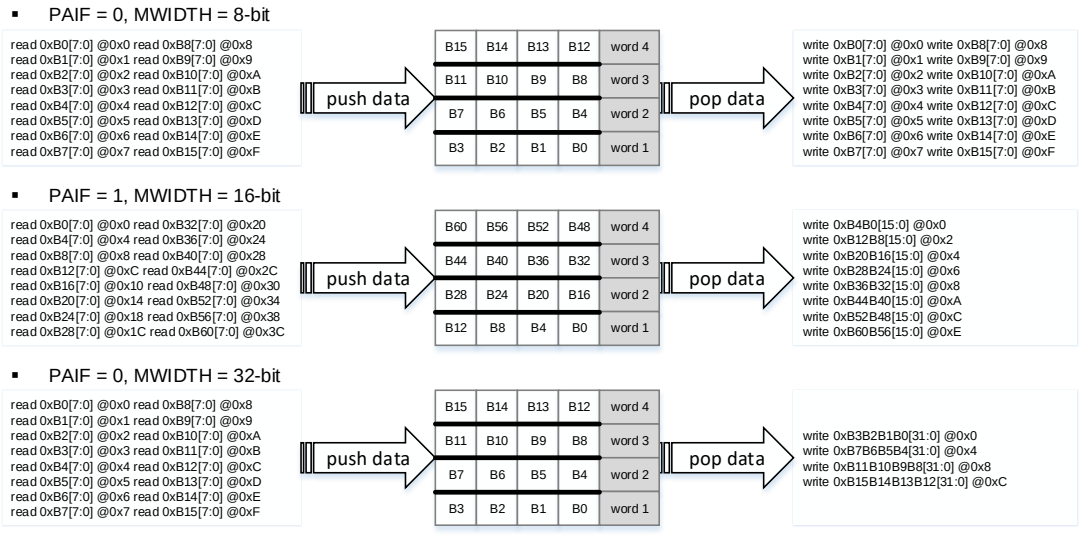
Pack / Unpack

In single-data mode, the MWIDTH bits are equal to the PWIDTH bits by force, data packing / unpacking is not needed.

In multi-data mode, the independent PWIDTH and MWIDTH bits configuration are supported for flexible DMA transfer. When the PWIDTH bits and the MWIDTH bits are not equal, DMA reading access and writing access are executed in different transfer width, and DMA packs / unpacks the data automatically. In DMA transfer operation, only little-endian addressing for both memory and peripheral is supported.

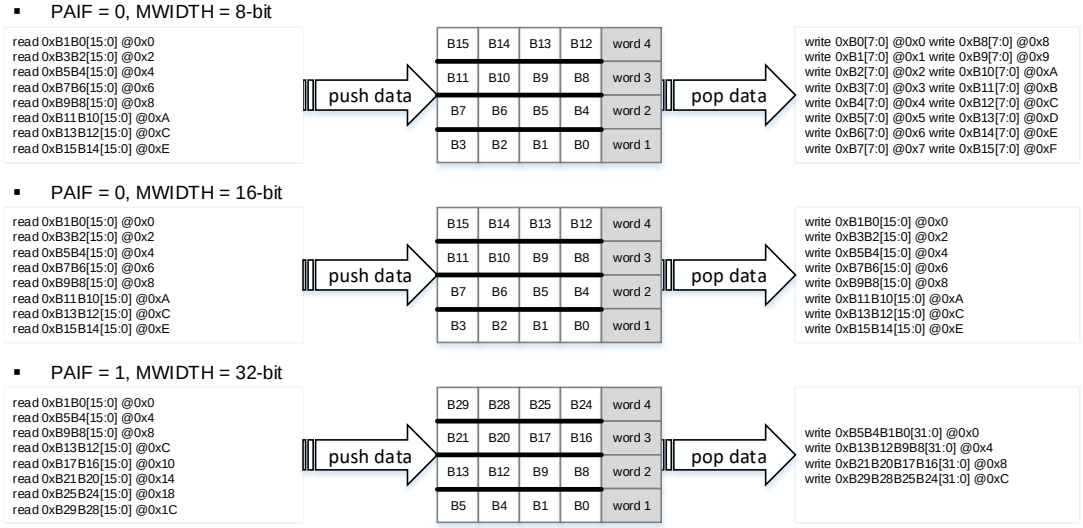
Suppose the CNT bits are 16, the PWIDTH bits are equal to '00', and both PNAGA and MNAGA are set. The DMA transfer operations for different MWIDTH are shown in the ***Figure 10-4. Data packing/unpacking when PWIDTH = '00'***.

Figure 10-4. Data packing/unpacking when PWIDTH = '00'



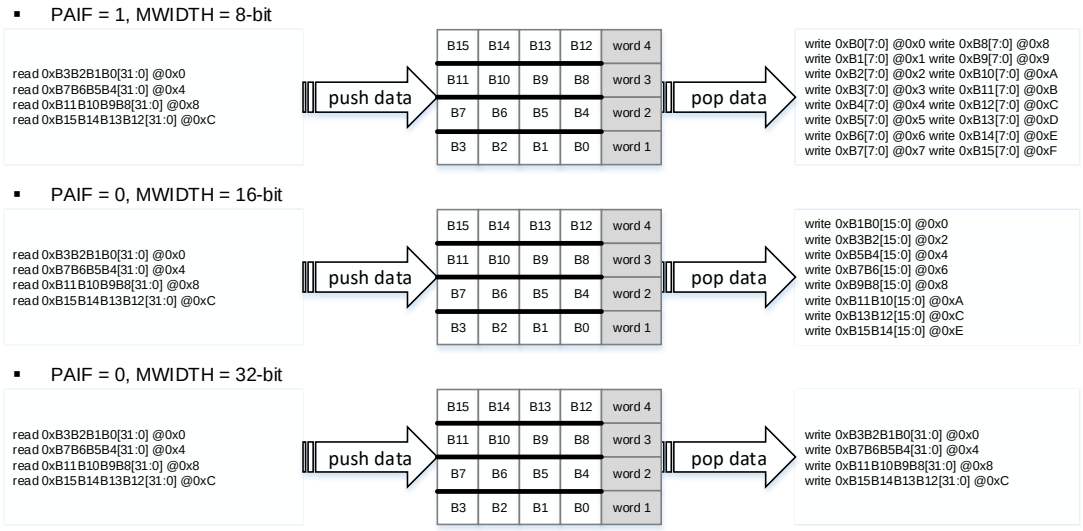
- Suppose the CNT bits are 8, the PWIDTH bits are equal to '01', and both PNAGA and MNAGA are set. The DMA transfer operations for different MWIDTH are shown in the ***Figure 10-5. Data packing / unpacking when PWIDTH =*** **'01'**.

Figure 10-5. Data packing / unpacking when PWIDTH = '01'



- Suppose DMA_CHxCNT is 4, the PWIDTH bits are equal to '10', and both PNAGA and MNAGA are set. The DMA transfer operations for different MWIDTH are shown in the ***Figure 10-6. Data packing/unpacking when PWIDTH = '10'***.

Figure 10-6. Data packing/unpacking when PWIDTH = '10'



10.4.3. 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_CHxM0ADDR, and DMA_CHxM1ADDR).

In the increasing mode, the next address is euqal to the current address plus 1 or 2 or 4, depending on the transfer data width. In multi-data mode with PBURST in the DMA_CHxCTL

register different from '00', if PAIF in the DMA_CHxCTL register is enabled, the next peripheral address increment is fixed to 4, and has nothing to do with the peripheral transfer data width. The PAIF has no meaning to the memory address generation.

Note: If PAIF in the DMA_CHxCTL register is enabled, the peripheral base address configured in the DMA_CHxPADDR register must be 32-bit alignment.

10.4.4. Circular mode

Circular mode is implemented to handle continue peripheral requests. The CMEN bit in the DMA_CHxCTL register is used to enable / disable the circular mode. Circular mode is available only when DMA controls the transfer flow. When the peripheral is selected as the transfer flow controller by setting the TFCS, the circular mode is automatically disabled immediately after the channel is enabled.

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 a transfer error is detected or the CHEN bit in the DMA_CHxCTL register is cleared.

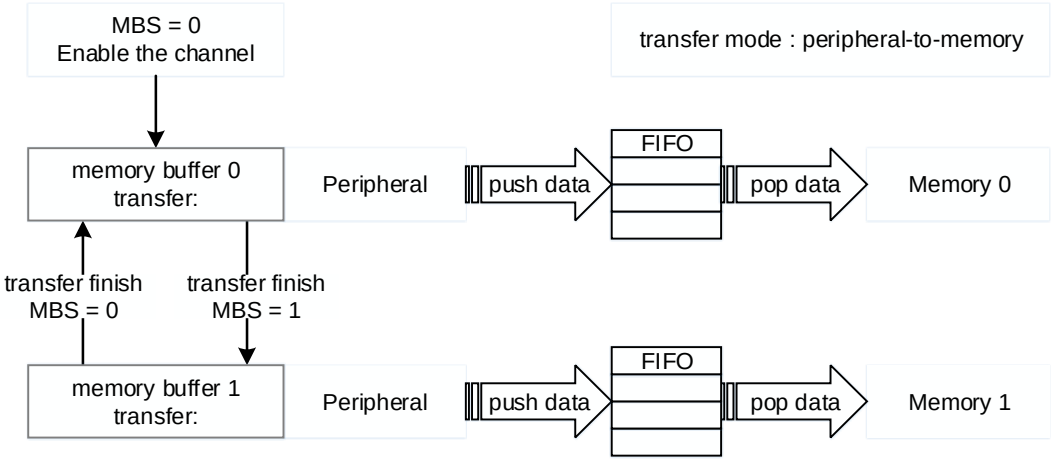
10.4.5. Switch-buffer mode

Similar to circular mode, switch-buffer mode is also implemented to handle continues peripheral requests. The SBMEN bit in the DMA_CHxCTL register is used to enable/disable the switch-buffer mode. When the switch-buffer mode is enabled, the circular mode is automatically enabled immediately after the channel is enabled. Switch-buffer mode is only available when the transfer mode is peripheral-to-memory or memory-to-peripheral. When the transfer mode is memory-to-memory, the switch-buffer mode is automatically disabled immediately after the channel is enabled.

Switch-buffer mode is supported with two memory buffers and the base address of the two memory buffers are separately configured in the DMA_CHxM0ADDR and DMA_CHxM1ADDR register. In switch-buffer mode, the DMA memory pointer switches from the current memory buffer to another at the end of every DMA transfer. During the DMA transmission, the memory buffer not being processed by DMA can be accessed by other AHB masters. In switch-buffer mode, the base address of the memory buffer not accessed by DMA can be updated even if the channel is enabled.

The MBS bit in the DMA_CHxCTL register is configured to select which memory buffer is accessed by DMA at the first DMA transfer before the channel is enabled. In switch-buffer mode, this bit switches automatically between '0' and '1' at the end of every DMA transfer, and can be used as a flag indicating the current memory buffer accessed by DMA during the transmission. The DMA operation of switch-buffer mode is shown in ***Figure 10-7. DMA operation of switch-buffer mode***.

Figure 10-7. DMA operation of switch-buffer mode



10.4.6. Transfer flow controller

The transfer flow controller controls the number of data items to be transferred. The TFCS bit in the DMA_CHxCTL register determines which of DMA and peripheral is selected to control the transfer flow.

- DMA as transfer flow controller: The CNT bits in the DMA_CHxCNT register determine the number of data items to be transferred. The CNT bits must be configured before the channel is enabled.
- Peripheral as transfer flow controller: The CNT bits configured in the DMA_CHxCNT register before the channel is enabled have no meaning and these bits are force to '0xFFFF' immediately after the channel is enabled. The peripheral determines when to finish the DMA transfer by informing a last request signal to DMA.

Note: When the transfer mode is memory-to-memory, the transfer flow controller is fixed to be DMA whatever the TFCS bit is configured to.

10.4.7. Transfer operation

Three transfer modes are supported to implement the data transfer, including peripheral-to-memory, memory-to-peripheral and memory-to-memory. Memory and peripheral can be configured as source and destination relatively.

Memory transfer

- Peripheral-to-memory mode:
  - In single-data mode, when the FIFO is not empty, DMA initiates a single memory transfer and writes data into the corresponding memory address.
  - In multi-data mode, when the FIFO counter reaches the critical value, DMA starts single or burst memory transfers to entirely fetch the FIFO data and write to the memory.

- Memory-to-peripheral mode:
  - In single-data mode, when the channel is enabled, DMA starts a single memory transfer and pushes the reading data into the FIFO immediately. During the transmission, the memory transfer is initiated only when the FIFO is empty.
  - In multi-data mode, when the channel is enabled, DMA starts several single or burst transfers to fill up the FIFO whether the peripheral request is asserted or not. During the transmission, the memory transfer is initiated once when there is enough space for it in the FIFO.
- Memory-to-memory mode: Only the multi-data mode is supported. When the FIFO counter reaches the critical value, DMA starts single or burst memory transfers to entirely fetch the FIFO data and write to the memory.

Peripheral transfer

- Peripheral-to-memory mode: When receiving a peripheral request and there is enough space in the FIFO for a peripheral transfer, DMA starts a peripheral transfer and pushes the reading data into the FIFO.
- Memory-to-peripheral mode: When receiving a peripheral request and there is enough data in the FIFO for a peripheral transfer, DMA starts a peripheral transfers to fetch the FIFO data and write to the peripheral.
- Memory-to-memory mode: Only the multi-data mode is supported. When the channel is enabled, DMA starts several peripheral transfers to fill up the FIFO. During the transmission, the peripheral transfer is initiated once when there is enough space for it in the FIFO.

10.4.8. Transfer finish

The DMA transfer is finished automatically and the FTFIFx bit in the DMA_INTF0 or DMA_INTF1 register is set when one of the following situations occurs:

- Transfer completion
- Software clear
- Error detection

Transfer completion

When enabled, the DMA begins to transfer data between peripheral and memory. After the pre-programmed number of data items has been transferred successfully, the DMA transfer is completed and the CHEN bit is automatically cleared in the DMA_CHxCTL register.

- Peripheral-to-memory mode: If DMA is the transfer flow controller, when the CNT bits reach zero and the contents of the FIFO have been entirely transferred into the memory, an end of transfer is generated. If peripheral is the transfer flow controller, the DMA transfer is completed when the last peripheral request has been responded and the contents of the FIFO have been entirely transferred into the memory.

- Memory-to-peripheral mode: If DMA is the transfer flow controller, when the CNT bits in the DMA_CHxCNT register reach zero, an end of transfer is achieved. If peripheral is the transfer flow controller, the DMA transfer is completed when the last peripheral request has been responded.
- Memory-to-memory: only DMA can be the transfer flow controller. When the CNT bits reach zero and the contents of the FIFO have been entirely transferred into the memory, an end of transfer is generated.

Software clear

The DMA transfer can be stopped by clearing the CHEN bit in the DMA_CHxCTL register by software. After the software cleared operation, the CHEN bit is still read as 1 to indicate that there are memory or peripheral transfers still active or the remaining data in the FIFO need to be transferred.

- Peripheral-to-memory: After the software cleared operation, the peripheral transfer is stopped when the current single or burst transfer is completed. To ensure that the data read from peripheral can be entirely transferred into the memory, the memory transfer continues to be active until the FIFO is empty. If the remaining byte number in the FIFO is not enough for a burst memory transfer, these data items are transferred in single transaction. If the remaining byte number is less than the memory transfer width, these data items are still written in memory transfer width with MSBs filled with zero. The software can read the CNT bits to calculate the number of valid data items in the memory. After the contents of the FIFO have been entirely transferred into the memory, the CHEN bit is cleared automatically by hardware and the FTFIFx bit in the DMA_INTF0 or DMA_INTF1 register is set.
- Memory-to-peripheral: After the software cleared operation, the DMA transfer is stopped when the current memory and peripheral transfer are completed. Then the CHEN bit is cleared and the FTFIFx bit is set.
- Memory-to-memory: The same as the peripheral-to-memory mode with the source memory transfer is implemented through the peripheral port.

Error detection

Three types error can disable the DMA transfer:

- FIFO error: When a wrong FIFO configuration is detected, the DMA channel is disabled immediately without starting any transfers. In this situation, the FTFIFx is not asserted. For more information about the FIFO error, refer to section ***Error***.
- Bus error: When the memory or peripheral port attempts to access an address beyond the access scope, a bus error is detected and the DMA transfer is stopped immediately without setting the FTFIFx. If this error is aroused by the peripheral port, the CNT bits are still decreased by 1. For more information about the bus error, refer to section ***Error***.
- Register access error: In switch-buffer mode, an access error is detected when a write command is active on the memory base address register which is being accessed by

DMA. When this error occurs, the DMA operation is the same as it after the CHEN bit software cleared. For more information about the register access error, refer to section ***Error***.

10.4.9. 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 or wait the current DMA transfer finished. When the CHEN bit is read as '0', configuring and starting a new DMA transfer is allowed.
2. Clear the FTFIFx bit in the DMA_INTF0 or DMA_INTF1 register, or a new DMA transfer can not be re-enabled.
3. Configure the TM bits in the DMA_CHxCTL register to set the transfer mode.
4. Configure the PERIEN bits in the DMA_CHxCTL register to select the target peripheral. If the transfer mode is memory-to-memory, the PERIEN bits have no meaning and this step can be skipped.
5. Configure the memory and peripheral burst types, the target memory buffer, switch-buffer mode, priority of the channel, memory and peripheral transfer width, memory and peripheral address generation algorithm, circular mode, the transfer flow controller in the DMA_CHxCTL register.
6. Configure multi-data mode, and the FCCV bits to set the FIFO counter critical value if multi-data mode is enabled in the DMA_CHxFCTL register.
7. Configure the enable bit for full transfer finish interrupt, half transfer finish interrupt, transfer access error interrupt, single-data mode exception interrupt in the DMA_CHxCTL register and the enable bit for FIFO error and exception interrupt in the DMA_CHxFCTL register.
8. Configure the DMA_CHxPADDR register for setting the peripheral base address.
9. If the switch-buffer mode is enabled, configure the DMA_CHxM0ADDR and DMA_CHxM1ADDR register for setting the memory base address. If only one memory buffer is to be used, configure the DMA_CHxM0ADDR or DMA_CHxM1ADDR corresponding with the MBS bit in the DMA_CHxCTL register.
10. Configure the DMA_CHxCNT register to set the total transfer data number.
11. Configure the CHEN bit with '1' in the DMA_CHxCTL register to enable the channel.

When restarting the suspended DMA transfer, it is recommended to respect the following steps:

1. Read the CHEN bit and ensure the DMA suspend operation has been completed. When the CHEN bit is read as '0', restarting the DMA transfer is allowed.
2. Clear the FTFIFx bit in the DMA_INTF0 or DMA_INTF1 register, or the DMA transfer can not be re-enabled.
3. Read the DMA_CHxCNT register to obtain the number of the remaining data items and calculate the number of the data items had already been transferred.
4. Configure the DMA_CHxPADDR register to update the peripheral address pointer.

5. Configure the DMA_CHxM0ADDR or the DMA_CHxM1ADDR register to update the memory address pointer.
6. Configure the DMA_CHxCNT with the number of the remaining data items.
7. Configure the CHEN bit with '1' in the DMA_CHxCTL register to restart the channel.

10.5. Interrupts

Each DMA channel has a dedicated interrupt. There are five interrupt events connected to each interrupt, including full transfer finish interrupt, half transfer finish interrupt, transfer access error interrupt, single-data mode exception interrupt, and FIFO error and exception interrupt. A DMA channel interrupt may be produced when any interrupt event occurs on the channel.

Each interrupt event has a dedicated flag bit in the DMA_INTF0 or DMA_INTF1 register, a dedicated clear bit in the DMA_INTC0 and DMA_INTC1 register, and a dedicated enable bit in the DMA_CHxCTL and CHxFCTL register, as described in the ***Table 10-5. DMA interrupt events***.

Table 10-5. DMA interrupt events

Interrupt event	Flag bit	Enable bit	Clear bit
	DMA_INTF0 or DMA_INTF1	DMA_CHxCTL or DMA_CHxFCTL	DMA_INTC0 or DMA_INTC1
Full transfer finish	FTFIF	FTFIE	FTFIFC
Half transfer finish	HTFIF	HTFIE	HTFIFC
Transfer access error	TAEIF	TAEIE	TAEIFC
Single-data mode exception	SDEIF	SDEIE	SDEIFC
FIFO error and exception	FEEIF	FEEIE	FEEIFC

These five events can be divided into three types:

- Flag: Full transfer finish flag and half transfer finish flag.
- Exception: Single-data mode exception and FIFO exception.
- Error: Transfer access error and FIFO error.

When the exception events occur, the DMA transmission is not affected and continues transferring normally. When the error events are detected, the DMA transmission is stopped. These three types of event are described in detail in the following sections.

10.5.1. Flag

Two flag events are supported, including full transfer finish flag and half transfer finish flag.

The full transfer finish flag is asserted, when one of the following situations occurs:

- The CNT bits reach zero when DMA is the transfer flow controller.
- When peripheral is the transfer flow controller, the last request is responded completely and the contents of the FIFO are entirely written into the memory in peripheral-to-memory mode.
- When the channel is disabled by software before the end of the transfer, the current memory and peripheral is completed and the contents of the FIFO are entirely written into the memory in peripheral-to-memory or memory-to-memory mode.
- When the channel is disabled because of register access error before the end of the transfer, the current memory and peripheral is completed and the contents of the FIFO are entirely written into the memory in peripheral-to-memory or memory-to-memory mode.

When the full transfer finish flag is asserted and the enabled bit for the full transfer finish interrupt is set, an interrupt is generated.

The half transfer finish flag is asserted, only when DMA is the transfer flow controller and half of the CNT bits are transferred. If peripheral is the transfer flow controller, DMA does not know when half of data items has been transferred and the half transfer finish flag will stay zero.

When the half transfer finish flag is asserted and the enabled bit for the half transfer finish interrupt is set, an interrupt is generated.

10.5.2. Exception

Two exception events are supported, including single-data mode exception and FIFO exception. These exceptions have no effect on the DMA transmission.

Single-data mode exception

This exception can be detected only when the single-data mode is enabled and the transfer mode is peripheral-to-memory. When a peripheral request is valid and the FIFO is not empty, but DMA can not get the bus authorization, then, the single data mode exception flag(SDEIFx) will be set.

When the single-data mode exception is asserted and the enabled bit for the single-data mode exception interrupt is set, an interrupt is generated.

FIFO exception

When a FIFO underrun or a FIFO overrun condition occurs, the FIFO exception is asserted. This exception can be detected only when the transmission is between peripheral and memory.

In peripheral-to-memory mode, when a peripheral request is valid and there is not enough space in the FIFO for the single or burst peripheral transfer, a FIFO overrun condition is detected. This peripheral request is not responded until the FIFO space is enough, and the accuracy of the data transmission will not be destroyed.

In memory-to-peripheral mode, when a peripheral request is valid and there is not enough data in the FIFO for the single or burst peripheral, a FIFO underrun condition is detected. This peripheral request is not responded until the data number in the FIFO is enough, and the accuracy of the data transmission will not be destroyed.

When the FIFO exception is asserted and the enabled bit for the FIFO error and exception interrupt is set, an interrupt is generated.

10.5.3. Error

FIFO error and transfer access error (including the register access error and bus error) can be detected during the DMA transmission, and the transmission can be stopped when one of the errors occurs.

FIFO error

For a good DMA operation, when the multi-data mode is enabled, the right and wrong configurations of the FIFO counter critical value corresponding with the memory transfer width and memory burst types are listed in ***Table 10-4. FIFO counter critical value configuration rules***.

If a wrong configuration is detected after enable the channel, a FIFO error is generated and the channel is disabled immediately without starting any transfers.

When the FIFO error is asserted and the enabled bit for the FIFO error and exception interrupt is set, an interrupt is generated.

Register access error

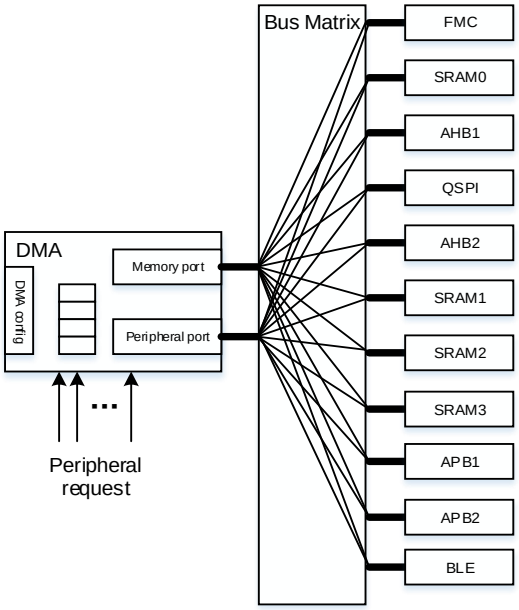
The register access error is detected only when the switch-buffer is enabled. If the software attempts to update a memory address register currently accessed by the DMA controller, a register access error is detected. For example, when the memory 0 buffer is the current source or destination, a write access on the DMA_CHxM0ADDR register could produce a register access error. When a register access error occurs, the DMA transmission is stopped when the current memory and peripheral transfer are completed and the valid FIFO data are entirely drained into the memory if needed.

When the register access error is asserted and the enabled bit for the transfer access error and exception interrupt is set, an interrupt is generated.

Bus error

When the address accessed by the DMA controller is beyond the allowed area, a response error will be received and the channel is disabled immediately. The allowed and forbidden access region for DMA is shown in ***Figure 10-8. System connection of DMA***.When the bus error is asserted and the enabled bit for the transfer access error and exception interrupt is set, an interrupt is generated.

Figure 10-8. System connection of DMA



10.6. Register definition

DMA base address: 0x4002 6000

10.6.1. Interrupt flag register 0 (DMA_INTF0)

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				FTFIF3	HTFIF3	TAEIF3	SDEIF3	Reserved	FEEIF3	FTFIF2	HTFIF2	TAEIF2	SDEIF2	Reserved	FEEIF2
				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
Reserved				FTFIF1	HTFIF1	TAEIF1	SDEIF1	Reserved	FEEIF1	FTFIF0	HTFIF0	TAEIF0	SDEIF0	Reserved	FEEIF0
				r	r	r	r		r	r	r	r	r		r

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27/21/11/5	FTFIFx	Full Transfer finish flag of channel x (x=0…3) Hardware set and software cleared by configuring DMA_INTC0 register. 0: Transfer has not finished on channel x 1: Transfer has finished on channel x
26/20/10/4	HTFIFx	Half transfer finish flag of channel x (x=0…3) Hardware set and software cleared by configuring DMA_INTC0 register. 0: Half number of transfer has not finished on channel x 1: Half number of transfer has finished on channel x
25/19/9/3	TAEIFx	Transfer access error flag of channel x (x=0…3) Hardware set and software cleared by configuring DMA_INTC0 register. 0: Transfer access error has not occurred on channel x 1: Transfer access error has occurred on channel x
24/18/8/2	SDEIFx	Single data mode exception of channel x (x=0…3) Hardware set and software cleared by configuring DMA_INTC0 register. 0: Single data mode exception has not occurred on channel x 1: Single data mode exception has occurred on channel x
23/17/7/1	Reserved	Must be kept at reset value.
22/16/6/0	FEEIFx	FIFO error and exception of channel x (x=0…3) Hardware set and software cleared by configuring DMA_INTC0 register. 0: FIFO error or exception has not occurred on channel x 1: FIFO error or exception has occurred on channel x

10.6.2. Interrupt flag register 1 (DMA_INTF1)

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				FTFIF7	HTFIF7	TAEIF7	SDEIF7	Reserved	FEEIF7	FTFIF6	HTFIF6	TAEIF6	SDEIF6	Reserved	FEEIF6
				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
Reserved				FTFIF5	HTFIF5	TAEIF5	SDEIF5	Reserved	FEEIF5	FTFIF4	HTFIF4	TAEIF4	SDEIF4	Reserved	FEEIF4
				r	r	r	r		r	r	r	r	r		r

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27/21/11/5	FTFIFx	Full Transfer finish flag of channel x (x=4…7) Hardware set and software cleared by configuring DMA_INTC1 register. 0: Transfer has not finished on channel x 1: Transfer has finished on channel x
26/20/10/4	HTFIFx	Half transfer finish flag of channel x (x=4…7) Hardware set and software cleared by configuring DMA_INTC1 register. 0: Half number of transfer has not finished on channel x 1: Half number of transfer has finished on channel x
25/19/9/3	TAEIFx	Transfer access error flag of channel x (x=4…7) Hardware set and software cleared by configuring DMA_INTC1 register. 0: Transfer access error has not occurred on channel x 1: Transfer access error has occurred on channel x
24/18/8/2	SDEIFx	Single data mode exception of channel x (x=4…7) Hardware set and software cleared by configuring DMA_INTC1 register. 0: Single data mode exception has not occurred on channel x 1: Single data mode exception has occurred on channel x
23/17/7/1	Reserved	Must be kept at reset value.
22/16/6/0	FEEIFx	FIFO error and exception of channel x (x=4…7) Hardware set and software cleared by configuring DMA_INTC1 register. 0: FIFO error or exception has not occurred on channel x 1: FIFO error or exception has occurred on channel x

10.6.3. Interrupt flag clear register 0 (DMA_INTC0)

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				FTFIFC3	HTFIFC3	TAEIFC3	SDEIFC3	Reserved	FEEIFC3	FTFIFC2	HTFIFC2	TAEIFC2	SDEIFC2	Reserved	FEEIFC2
				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
Reserved				FTFIFC1	HTFIFC1	TAEIFC1	SDEIFC1	Reserved	FEEIFC1	FTFIFC0	HTFIFC0	TAEIFC0	SDEIFC0	Reserved	FEEIFC0
				w	w	w	w		w	w	w	w	w		w

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27/21/11/5	FTFIFCx	Clear bit for Full transfer finish flag of channel x (x=0…3) 0: No effect 1: Clear full transfer finish flag
26/20/10/4	HTFIFCx	Clear bit for half transfer finish flag of channel x (x=0…3) 0: No effect 1: Clear half transfer finish flag
25/19/9/3	TAEIFCx	Clear bit for ransfer access error flag of channel x (x=0…3) 0: No effect 1: Clear transfer access error flag
24/18/8/2	SDEIFCx	Clear bit for single data mode exception of channel x (x=0…3) 0: No effect 1: Clear single data mode exception flag
23/17/7/1	Reserved	Must be kept at reset value.
22/16/6/0	FEEIFCx	Clear bit for FIFO error and exception of channel x (x=0…3) 0: No effect 1: Clear FIFO error and exception flag

10.6.4. Interrupt flag clear register 1 (DMA_INTC1)

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				FTFIFC7	HTFIFC7	TAEIFC7	SDEIFC7	Reserved	FEEIFC7	FTFIFC6	HTFIFC6	TAEIFC6	SDEIFC6	Reserved	FEEIFC6
				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
Reserved				FTFIFC5	HTFIFC5	TAEIFC5	SDEIFC5	Reserved	FEEIFC5	FTFIFC4	HTFIFC4	TAEIFC4	SDEIFC4	Reserved	FEEIFC4
				w	w	w	w		w	w	w	w	w		w

Bits	Fields	Descriptions

31:28	Reserved	Must be kept at reset value.
27/21/11/5	FTFIFCx	Clear bit for full transfer finish flag of channel x (x=4…7) 0: No effect 1: Clear full transfer finish flag
26/20/10/4	HTFIFCx	Clear bit for half transfer finish flag of channel x (x=4…7) 0: No effect 1: Clear half transfer finish flag
25/19/9/3	TAEIFCx	Clear bit for transfer access error flag of channel x (x=4…7) 0: No effect 1: Clear transfer access error flag
24/18/8/2	SDEIFCx	Clear bit for single data mode exception of channel x (x=4…7) 0: No effect 1: Clear single data mode exception flag
23/17/7/1	Reserved	Must be kept at reset value.
22/16/6/0	FEEIFCx	Clear bit for FIFO error and exception of channel x (x=4…7) 0: No effect 1: Clear FIFO error and exception flag

10.6.5. Channel x control register (DMA_CHxCTL) (x = 0..7)

Address offset: 0x10 + 0x18 × 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				PERIEN[2:0]			MBURST[1:0]		PBURST[1:0]		Reserved	MBS	SBMEN	PRIO[1:0]	
					rw		rw		rw			rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAIF	MWIDTH[1:0]		PWIDTH[1:0]		MNAGA	PNAGA	CMEN	TM[1:0]		TFCS	FTFIE	HTFIE	TAEIE	SDEIE	CHEN
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.
27:25	PERIEN[2:0]	Peripheral enable Software set and cleare. 000: Enable peripheral 0 001: Enable peripheral 1 010: Enable peripheral 2 011: Enable peripheral 3 100: Enable peripheral 4

		101: Enable peripheral 5 110: Enable peripheral 6 111: Enable peripheral 7 These bits can NOT be written when CHEN is ‘1’.
24:23	MBURST[1:0]	Transfer burst type of memory Software set and cleare. 00: single burst 01: INCR4 (4-beat incrementing burst) 10: INCR8 (8-beat incrementing burst) 11: INCR16 (16-beat incrementing burst) These bits can NOT be written when CHEN is ‘1’. These bits are automatically locked as ‘00’ by hardware immdiately after enable CHEN if MDMEN in the DMA_CHxFCTL register is configured to ‘0’.
22:21	PBURST[1:0]	Transfer burst type of peripheral Software set and cleare. 00: single burst 01: INCR4 (4-beat incrementing burst) 10: INCR8 (8-beat incrementing burst) 11: INCR16 (16-beat incrementing burst) These bits can NOT be written when CHEN is ‘1’. These bits are automatically locked as ‘00’ by hardware immediately after enable CHEN if MDMEN in the DMA_CHxFCTL register is configured to ‘0’.
20	Reserved	Must be kept at reset value.
19	MBS	Memory buffer select Hardware and software set, Hardware and software clear. 0: Memory 0 is selected as memory transfer area 1: Memory 1 is selected as memory transfer area This bit can NOT be written when CHEN is ‘1’. During the transmission, this bit can be set and cleared by hardware at the end of transfer to indicate which memory buffer is being accessed by DMA.
18	SBMEN	Switch-buffer mode enable Software set and clear. 0: Disable switch-buffer mode 1: Enable switch-buffer mode This bit can NOT be written when CHEN is ‘1’.
17:16	PRIO[1:0]	Priority level Software set and clear. 00: Low 01: Medium 10: High 11: Ultra high

		These bits can NOT be written when CHEN is ‘1’.
15	PAIF	Peripheral address increment fixed Software set and clear. 0: The peripheral address increment is determined by PWIDTH 1: The peripheral address increment is fixed to 4 This bit can NOT be written when CHEN is ‘1’. During the transmission, when PNAGA is configured to ‘0’, this bit has no effect. These bits are automatically locked as ‘0’ by hardware immediately after enable CHEN if MDMEN in the DMA_CHxFCTL register is configured to ‘0’ or PBURST are not equal to ‘00’.
14:13	MWIDTH[1:0]	Transfer width of memory Software set and clear. 00: 8-bit 01: 16-bit 10: 32-bit 11: Reserved These bits can NOT be written when CHEN is ‘1’. These bits are automatically locked as PWIDTH by hardware immediately after enable CHEN if MDMEN in the DMA_CHxFCTL register is configured to ‘0’.
12:11	PWIDTH[1:0]	Transfer width of peripheral Software set and clear. 00: 8-bit 01: 16-bit 10: 32-bit 11: Reserved These bits can NOT be written when CHEN is ‘1’.
10	MNAGA	Next address generation algorithm of memory Software set and clear 0: Fixed address mode 1: Increasing address mode This bit can NOT be written when CHEN is ‘1’.
9	PNAGA	Next address generation algorithm of peripheral Software set and clear 0: Fixed address mode 1: Increasing address mode This bit can NOT be written when CHEN is ‘1’.
8	CMEN	Circular mode enable Software set and clear. 0: Disable circular mode. 1: Enable circular mode This bit can NOT be written when CHEN is ‘1’.

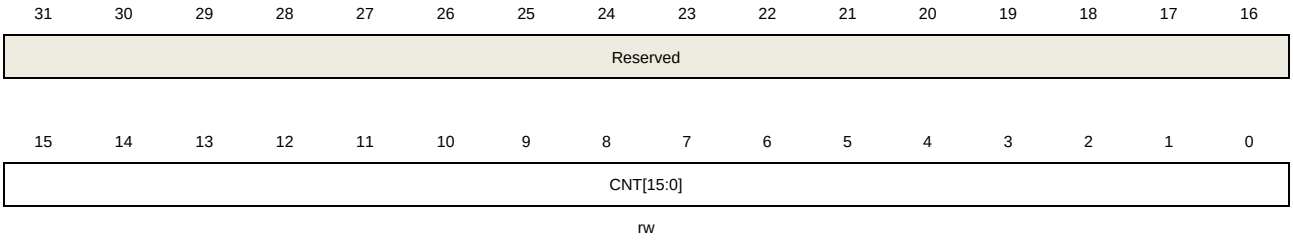
		This bit is automatically locked as ‘0’ by hardware immediately after enable CHEN if TFCS is configured to ‘1’. This bit is automatically locked as ‘1’ by hardware immediately after enable CHEN if SBMEN is configured to ‘1’.
7:6	TM[1:0]	Transfer mode Software set and clear. 00: Read from peripheral and write to memory 01: Read from memory and write to peripheral 10: Read from memory and write to memory 11: Reserved These bits can NOT be written when CHEN is ‘1’.
5	TFCS	Transfer flow controller select Software set and clear. 0: DMA is selected as the transfer flow controller 1: Peripheral is selected as the transfer flow controller This bit can NOT be written when CHEN is ‘1’.
4	FTFIE	Enable bit for full transfer finish interrupt Software set and clear. 0:Disable full transfer finish interrupt 1:Enable full transfer finish interrupt
3	HTFIE	Enable bit for half transfer finish interrupt Software set and clear. 0: Disable half transfer finish interrupt 1: Enable half transfer finish interrupt
2	TAEIE	Enable bit for tranfer access error interrupt Software set and clear. 0: Disable tranfer access error interrupt 1: Enable tranfer access error interrupt
1	SDEIE	Enable bit for single data mode exception interrupt Software set and clear. 0: Disable single data mode exception interrupt 1: Enable single data mode exception interrupt
0	CHEN	Channel enable Software set, hardware clear. 0: Disable channel 1: Enable channel When this bit is asserted, the DMA transfer is started. This bit is automaticly cleared when one of the following situations occurs: When the transfer of channel is fully finished. When a wrong FIFO configuration or a transfer access error is detected.

After a software clear operation, this bit is still read as 1 to indicate that there are memory or peripheral transfers still active until hardware has terminated all activity, at which point this bit is read as 0. Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.

10.6.6. Channel x counter register (DMA_CHxCNT) (x = 0..7)

Address offset: 0x14 + 0x18 × x
Reset value: 0x0000 0000

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

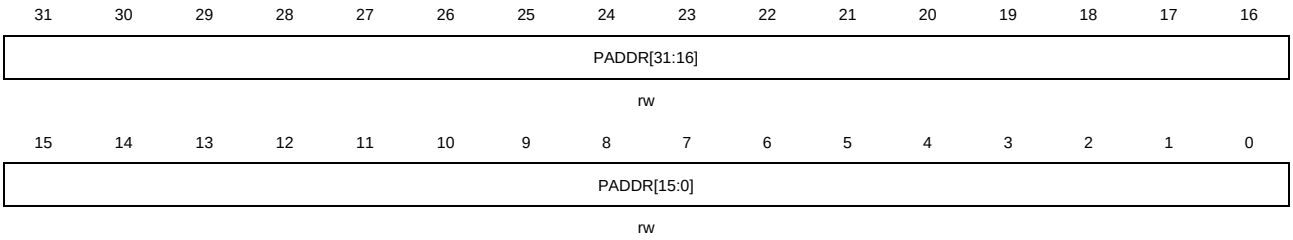


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CNT[15:0]	Transfer counter These bits can NOT be written when CHEN in the DMA_CHxCTL register is '1'. These bits are related to PWIDTH. During the transmission, These bits signify the number of remaining data to be transferred. After each DMA peripheral transfer, CNT is decreamented by 1. If CMEN or SBMEN in the DMA_CHxCTL register is configured to '1', CNT can be reloaded automatically to the original value at the end of transfer.

10.6.7. Channel x peripheral base address register (DMA_CHxPADDR) (x = 0..7)

Address offset: 0x18 + 0x18 × x
Reset value: 0x0000 0000

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



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.
Note：If PAIF in the DMA_CHxCTL register is enable, these bits must be configured to 32-bit alignment.

10.6.8. Channel x memory 0 base address register (DMA_CHxM0ADDR) (x = 0..7)

Address offset: 0x1C + 0x18 × 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
M0ADDR[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M0ADDR[15:0]															
rw															

Bits	Fields	Descriptions
31:0	M0ADDR[31:0]	Memory 0 base address When MBS in the DMA_CHxCTL register is read as to ‘0’, these bits specific the memory base address accessed by DMA during the transmission. These bits can NOT be written when CHEN in the DMA_CHxCTL register is ‘1’ and MBS in the DMA_CHxCTL register is read as ‘0’. When memory 0 is selected as memory transfer area and 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 memory 0 is selected as memory transfer area and 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.

10.6.9. Channel x memory 1 base address register (DMA_CHxM1ADDR) (x = 0..7)

Address offset: 0x20 + 0x18 × 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
M1ADDR[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

M1ADDR[15:0]
rw

Bits	Fields	Descriptions
31:0	M1ADDR[31:0]	Memory 1 base address When MBS in the DMA_CHxCTL register is read as to ‘1’, these bits specific the memory base address accessed by DMA during the transmission. These bits can NOT be written when CHEN in the DMA_CHxCTL register is ‘1’ and MBS in the DMA_CHxCTL register is read as ‘1’. When memorty 1 is selected as memory tranfer area and 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 memorty 1 is selected as memory tranfer area and 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.

10.6.10. Channel x FIFO control register (DMA_CHxFCTL) (x = 0..7)

Address offset: 0x24 + 0x18 × x

Reset value: 0x0000 0021

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FEEIE	Reserved	FCNT[2:0]			MDMEN	FCCV[1:0]	
								rw		r			rw	rw	

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	FEEIE	Enable bit for FIFO error and exception interrupt Software set and clear. 0: Disable FIFO error and exception interrupt 1: Enable FIFO error and exception interrupt
6	Reserved	Must be kept at reset value.
5:3	FCNT[2:0]	FIFO counter Hardware set and clear. 000: The FIFO is not empty and the data is less than 1 word 001: FIFO data is equal or more than 1 word and less than 2 words 010: FIFO data is equal or more than 2 words and less than 3 words 011: FIFO data is equal or more than 3 words and less than 4 words

		100: FIFO Empty 101: FIFO Full 110~111: Reserved These bits specific the number of data stored in FIFO during the transmission. When MDMEN is configured to '0', these bits has no meaning.
2	MDMEN	Multi-data mode enable Software set and clear. 0: Disable Multi-data mode 1: Enable Multi-data mode These bits can NOT be written when CHEN in the DMA_CHxCTL register is '1'. These bits are automatically locked as '1' by hardware immediately after enable CHEN in the DMA_CHxCTL register if TM in the DMA_CHxCTL register is configured to '10'.
1:0	FCCV[1:0]	FIFO counter critical value Software set and clear 00: One word 01: Two Words 10: Three Words 11: Four Words These bits can NOT be written when CHEN in the DMA_CHxCTL register is '1'. When MDMEN is configured to '0', these bits has no meaning.

11. Debug (DBG)

11.1. Overview

The GD32VW55x series provide a variety of debug feature. They are implemented with a standard configuration of the RISC-V module together with a daisy chained standard TAP controller. Debug functions are integrated into the RISC-V. The debug system supports standard JTAG debug. The debug refer to the following documents:

- RISC-V External Debug Support Version 0.13

The DBG hold unit helps debugger to debug power saving mode, TIMER, I2C, WWDGT, FWDGT and RTC. When corresponding bit is set, provide clock when in power saving mode or hold the state for TIMER, WWDGT, FWDGT, I2C or RTC.

11.2. JTAG function overview

Debug capabilities can be accessed by a debug tool via JTAG interface.

11.2.1. Pin assignment

The JTAG interface provides 5-pin standard JTAG, known as JTAG clock (JTCK), JTAG mode selection (JTMS), JTAG data input (JTDI), JTAG data output (JTDO) and JTAG reset (NJTRST, active low).

The pin assignment are:

PA15: JTDI

PA14: JTCK

PA13: JTMS

PB4: NJTRST

PB3: JTDO

By default, 5-pin standard JTAG debug mode is chosen after reset. Users can also use JTAG function without NJTRST pin, then the PB4 can be used to other GPIO functions. (NJTRST tied to 1 by hardware). The pin assignment for debugging with cJTAG are: PA14: JTCK, PA13: JTMS, other pins (PA15, PB4, PB3) can be used for normal GPIO functions. If JTAG not used, all 5-pin can be released to other GPIO functions. Please refer to ***General-purpose and alternate-function I/Os (GPIO and AFIO)***.

11.2.2. JTAG daisy chained structure

The RISC-V JTAG TAP is connected to a Boundary-Scan (BSD) JTAG TAP. The BSD JTAG

IR is 5-bit width, and the RISC-V JTAG IR is also 5-bit width.

The BSD JTAG IDCODE is 0x790007A3.

11.2.3. Debug reset

The System reset initializes the majority of the RISC-V. The NJTRST reset can reset JTAG TAP controller only.

11.3. Debug hold function overview

11.3.1. Debug support for power saving mode

When 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_IRC16M. When exit the standby mode, a system reset generated.

When 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_IRC16M.

When 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, FWDGT and RTC

When the core halted and the corresponding bit in DBG control register 1(DBG_CTL1/2) is set, the following behaved.

For TIMER, the timer counters stopped and hold for debug.

For I2C, SMBUS timeout hold for debug.

For WWDGT or FWDGT, the counter clock stopped for debug.

For RTC, the counter is stopped for debugging.

11.4. Register definition

DEBUG base address: 0xE0044000

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 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.															

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

Bits	Fields	Descriptions
31:3	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: At the standby mode, the clock of AHB bus and system clock are provided by CK_IRC16M, a system reset generated when exit standby mode

1	DSLP_HOLD	Deep-sleep mode hold bit This bit is set and reset by software 0: No effect 1: At the Deep-sleep mode, the clock of AHB bus and system clock are provided by CK_IRC16M
0	SLP_HOLD	Sleep mode hold bit This bit is set and reset by software 0: No effect 1: At 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.									I2C1_HOLD	I2C0_HOLD	Reserved.				
									rw	rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved.			FWDGT_HOLD	WWDGT_HOLD	RTC_HOLD	Reserved					TIMER5_HOLD	Reserved		TIMER2_HOLD	TIMER1_HOLD
			rw	rw	rw						rw			rw	rw

Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22	I2C1_HOLD	I2C1 hold bit This bit is set and reset by software 0: No effect 1: Hold the I2C1 SMBUS timeout for debug when core halted
21	I2C0_HOLD	I2C0 hold bit This bit is set and reset by software 0: No effect 1: Hold the I2C0 SMBUS timeout for debug when core halted
20:13	Reserved	Must be kept at reset value
12	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.

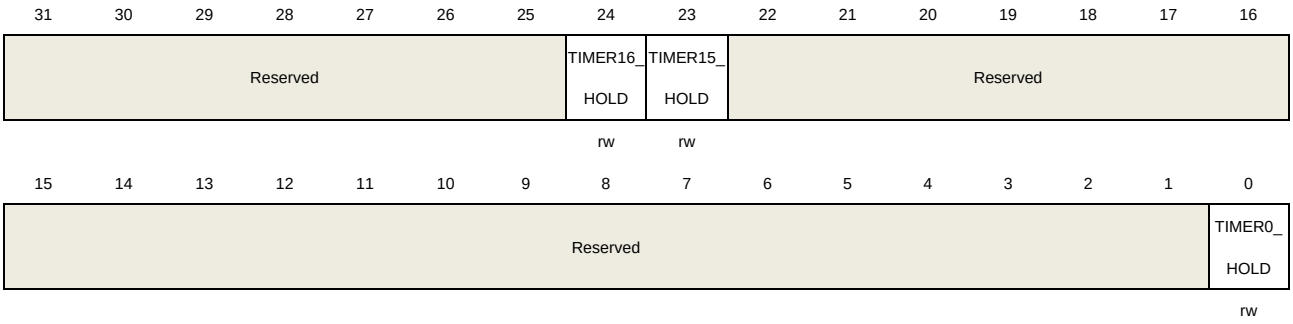
11	WWDGT_HOLD	WWDGT hold bit This bit is set and reset by software. 0: No effect 1: Hold the WWDGT counter clock for debugging when the core is halted.
10	RTC_HOLD	RTC hold bit This bit is set and reset by software. 0: No effect 1: Hold the RTC counter for debugging when the core is halted.
9:5	Reserved	Must be kept at reset value
4	TIMER5_HOLD	TIMER5 hold bit This bit is set and reset by software. 0: No effect 1: Hold the TIMER5 counter for debugging when the core is halted.
3:2	Reserved	Must be kept at reset value
1	TIMER2_HOLD	TIMER2 hold bit This bit is set and reset by software. 0: No effect 1: Hold the TIMER2 counter for debugging when the core is halted.
0	TIMER1_HOLD	TIMER1 hold bit This bit is set and reset by software. 0: No effect 1: Hold the TIMER1 counter for debugging when the core is halted.

11.4.4. Control register 2 (DBG_CTL2)

Address offset: 0x0C

Reset value: 0x0000 0000; power reset only

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



Bits	Fields	Descriptions
31:25	Reserved	Must be kept at reset value.

24	TIMER16_HOLD	TIMER16 hold bit This bit is set and reset by software. 0: No effect 1: Hold the TIMER16 counter for debugging when the core is halted.
23	TIMER15_HOLD	TIMER15 hold bit This bit is set and reset by software. 0: No effect 1: Hold the TIMER15 counter for debugging when the core is halted.
22:1	Reserved	Must be kept at reset value.
0	TIMER0_HOLD	TIMER0 hold bit This bit is set and reset by software 0: No effect 1: Hold the TIMER0 counter for debug when core halted

12. Analog to digital converter (ADC)

12.1. Overview

A 12-bit successive approximation analog-to-digital converter module(ADC) is integrated on the MCU chip, which can sample analog signals from 9 external channels and 2 internal channels. The 11 ADC sampling channels all support a variety of operation modes. After sampling and conversion, the conversion results can be stored in the corresponding data registers according to the least significant bit(LSB) alignment or the most significant(MSB) bit alignment. An on-chip hardware oversample scheme improves performances and reduces the computational burden of MCU.

12.2. Characteristics

- High performance:
  - ADC sampling rsolution:12-bit, 10-bit, 8-bit or 6-bit.
  - ADC sampling rate: 3 MSPs for 12-bit resolution, 3.5 MSPs for 10-bit resolution, 4.2 MSPs for 8-bit resolution, 5.25 MSPs for 6-bit resolution, faster sampling rate can be obtained by lowering the resolution.
  - Programmable sampling time.
  - Data storage mode: the most significant bit and the least significant bit.
  - DMA support for routine channels.
- Analog input channels:
  - 9 external analog inputs.
  - 1 channel for internal temperature sensor (VSENSE).
  - 1 channel for internal reference voltage (VREFINT).
- Start-of-conversion can be initiated:
  - By software.
  - By hardware triggers.
- Operation modes:
  - Convert a single channel or scan a sequence of channels.
  - Single operation mode converts selected input channel once per trigger.
  - Continuous operation mode converts selected input channels continuously.
  - Discontinuous operation mode.
- Interrupt generation
  - At the end of conversions.
  - Analog watchdog event.
  - Overflow event.
- Analog watchdog
- Oversampling.
  - 16-bit data register.

- Oversampling ratio adjustable from 2x to 256x.
- Programmable data shift up to 8-bits.

- ADC input range: $0 \leq V_{IN} \leq V_{DDA}$.

12.3. Pins and internal signals

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

Table 12-1. ADC internal input signals

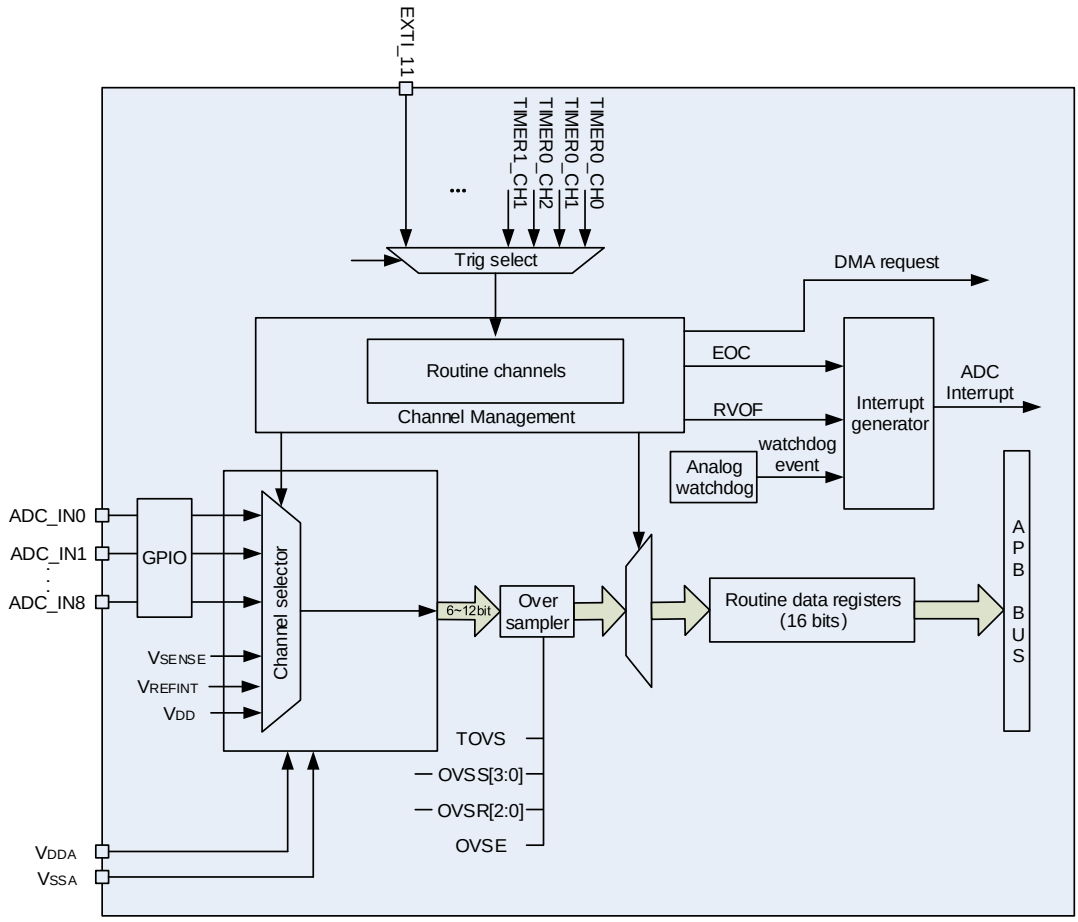
Internal signal name	Description
V_{SENSE}	Internal temperature sensor output voltage
V_{REFINT}	Internal voltage reference output voltage

Table 12-2. ADC input pins definition

Name	Remarks
V_{DDA}	Analog power supply equals to VDD
V_{SSA}	Ground for analog power supply equals to VSS
ADCx_IN[8:0]	Up to 9 external channels

12.4. Function overview

Figure 12-1. ADC module block diagram



12.4.1. ADC clock

The CK_ADC clock is synchronous with the AHB and APB2 clock and provided by the clock controller. The maximum frequency is 42MHz. ADC clock can be divided and configured by RCU controller.

12.4.2. ADCON switch

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 be enter power-off mode

12.4.3. Routine sequence

The channel management circuit can organize the sampling conversion channels into a sequence: routine sequence. The routine sequence supports up to 11 channels, and each

channel is called routine channel.

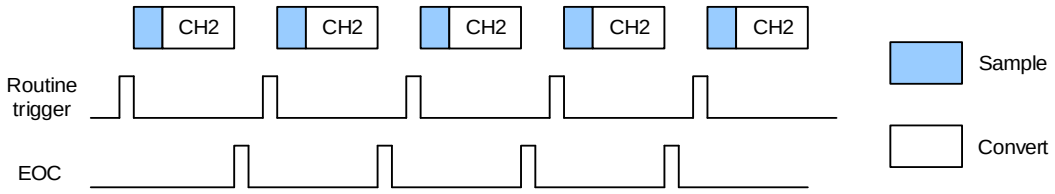
The RL[3:0] bits in the ADC_RSQ0 register specify the total conversion sequence length. The ADC_RSQ0~ADC_RSQ2 registers specify the selected channels of the routine sequence.

12.4.4. 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. When the ADCON is 1, the ADC samples and converts a single channel, once the corresponding software trigger or external trigger is active.

Figure 12-2. Single operation mode



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

Software procedure for a single operation mode of a routine channel:

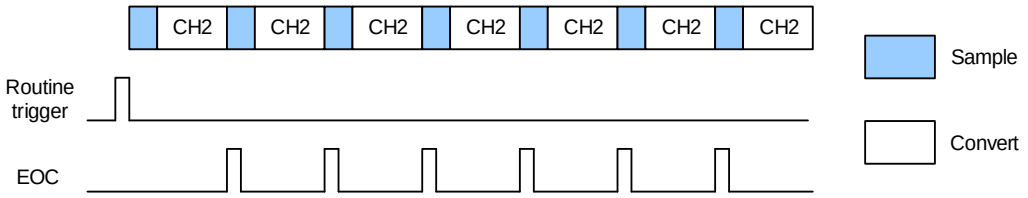
1. Make sure the DISRC, SM in the ADC_CTL0 register and CTN bit in the ADC_CTL1 register are reset;
2. Configure the RSQ0 with the analog channel number;
3. Configure the ADC_SAMPTx register;
4. Configure the ETMRC and ETSRC bits in the ADC_CTL1 register if it is needed;
5. Set the SWRCST bit, or generate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

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

Continuous operation mode

The continuous operation mode will be enabled when 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 12-3. Continuous conversion 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 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. After a delay of one CK_ADC, read the converted date in the ADC_RDATA register;
8. Clear the EOC flag by writing 0;
9. Repeat steps 6~8 as soon as the conversion is in need.

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

To avoid checking, DMA can be used to transfer the converted data:

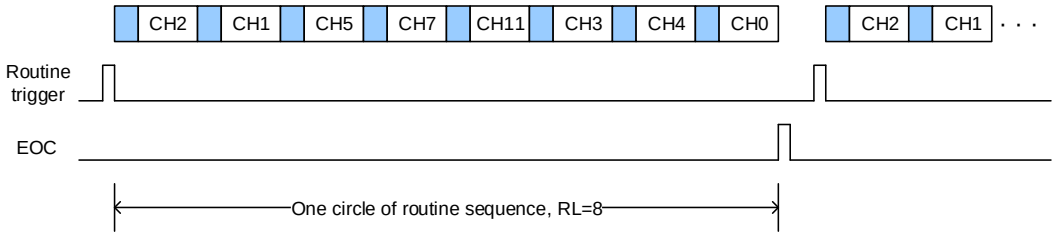
1. Set the CTN and DMA bits in the ADC_CTL1 register;
2. Configure RSQ0 with the analog channel number;
3. Configure ADC_SAMPTx register;
4. Configure the ETMRC and ETSRC bits in the ADC_CTL1 register in if it is needed;
5. Prepare the DMA module to transfer data from the ADC_RDATA;
6. Set the SWRCST bit, or generate an external trigger for the routine sequence.

Scan operation mode

The scan operation mode will be enabled when the SM bit in the ADC_CTL0 register is set. In this mode, the ADC performs conversion on all channels with a specific routine sequence specified in the ADC_RSQ0~ADC_RSQ2 registers. When the ADCON is 1, the ADC samples and converts specified channels one by one in the routine sequence till the end of the sequence, once the corresponding software trigger or external trigger is active. The conversion data will be stored in the ADC_RDATA register. After conversion of the routine sequence, the EOC will be set. An interrupt will be generated if the EOCIE bit is set. The DMA bit in ADC_CTL1 register must be set when the routine sequence works in scan mode.

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

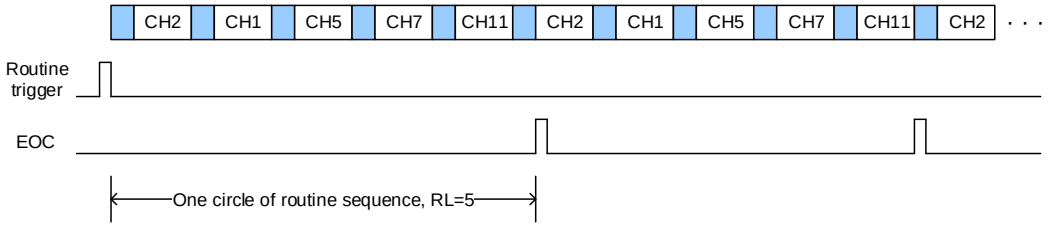
Figure 12-4. Scan operation mode, continuous disable



Software procedure for scan operation mode on a routine sequence:

1. Set the SM bit in the ADC_CTL0 register and the DMA bit in the ADC_CTL1 register;
2. Configure ADC_RSQx and ADC_SAMPTx registers;
3. Configure 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;
6. Wait for the EOC flag to be set;
7. Clear the EOC flag by writing 0.

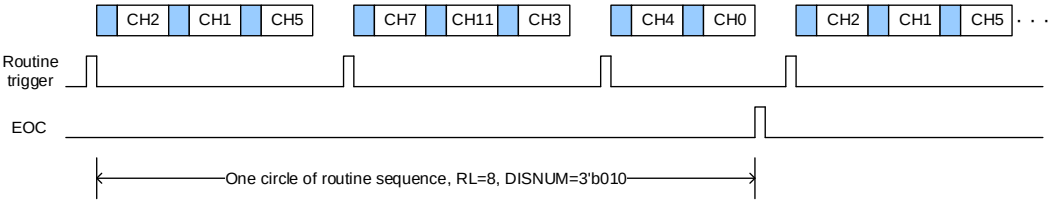
Figure 12-5. Scan operation mode, continuous enable



Discontinuous operation mode

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 selected in the ADC_RSQ0~ADC_RSQ2 registers. The value of n is configured by the DISNUM[2:0] bits in the ADC_CTL0 register. When the corresponding software trigger or external trigger is active, the ADC samples and coverts the next n channels configured in the ADC_RSQ0~ADC_RSQ2 registers until all the channels of routine sequence are done. The EOC will be set after every circle of the routine sequence. An interrupt will be generated if the EOCIE bit is set.

Figure 12-6. Discontinuous operation mode



Software procedure for discontinuous operation mode on a routine sequence:

1. Set the DISRC bit in the ADC_CTL0 register and the DMA bit in the ADC_CTL1 register;

2. Configure DISNUM [2:0] bits in the ADC_CTL0 register;
3. Configure ADC_RSQx and ADC_SAMPTx registers;
4. Configure the ETMRC and ETSRC bits in the ADC_CTL1 register 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;
7. Repeat step6 if it is needed;
8. Wait the EOC flag to be set;
9. Clear the EOC flag by writing 0.

12.4.5. Conversion result threshold monitor function

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

12.4.6. 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 is aligned on a byte, which are shown as ***Figure 12-7. Data storage mode of 12-bit resolution***, ***Figure 12-8. Data storage mode of 10-bit resolution***, ***Figure 12-9. Data storage mode of 8-bit resolution***, ***Figure 12-10. Data storage mode of 6-bit resolution***.

Figure 12-7. Data storage mode of 12-bit resolution

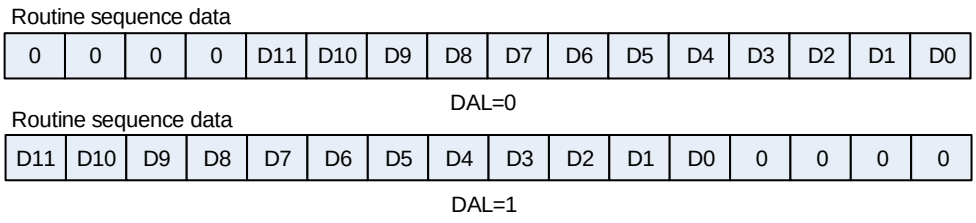


Figure 12-8. Data storage mode of 10-bit resolution

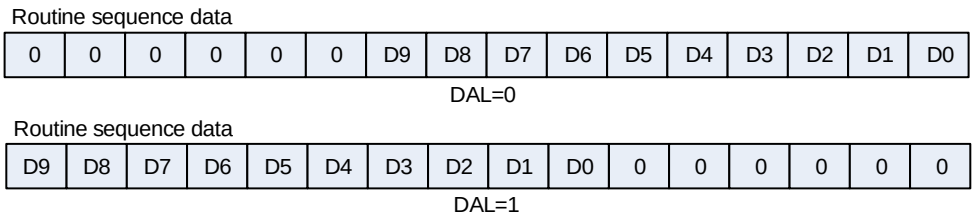


Figure 12-9. Data storage mode of 8-bit resolution

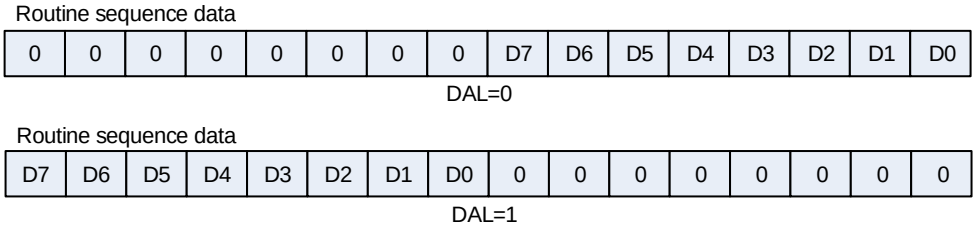
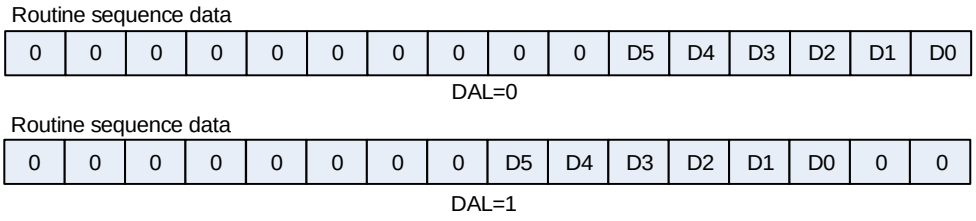


Figure 12-10. Data storage mode of 6-bit resolution



12.4.7. Sampling time configuration

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

For example, when the CK_ADC = 42MHz and sample time is 2.5cycles, the total conversion time is “2.5+12” CK_ADC cycles, that means 0.345us.

12.4.8. External trigger configuration

The conversion of routine sequence can be triggered by rising / falling edge of external trigger inputs. The ETMRC[1:0] and ETMIC[1:0] bits in the ADC_CTL1 register control the trigger modes of routine sequence respectively. The external trigger source of routine channel group is controlled by the ETSRC[3:0] bits in the ADC_CTL1 register.

Table 12-3. External trigger modes

ETMRC[1:0]	Trigger mode
00	External trigger disable
01	Rising edge of external trigger enable
10	Falling edge of external trigger enable
11	Rising and falling edge of external trigger enable

Table 12-4. External trigger source for ADC

ETSRC[3:0]	Trigger Source	Trigger Type
0000	TIMER0_CH0	Hardware trigger
0001	TIMER0_CH1	
0010	TIMER0_CH2	

ETSRC[3:0]	Trigger Source	Trigger Type
0011	TIMER1_CH1	
0100	TIMER1_CH2	
0101	TIMER1_CH3	
0110	TIMER1_TRGO	
0111	TIMER2_CH0	
1000	TIMER2_TRGO	
1001	TIMER2_CH2	
1010	TIMER15_CH0	
1011	TIMER16_CH0	
1100	Reserved	
1101	Reserved	
1110	TIMER5_TRGO	
1111	EXTI_11	

The selection of the external triggers can be changed on the fly, while no trigger event occurs due to this change.

12.4.9. DMA request

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

12.4.10. Overflow detection

Overflow detection is enabled when DMA is enabled or EOCM bit in ADC_CTL1 is set. An overflow event occurs when a routine conversion is done before the prior routine data has been read out. The ROVF bit of the ADC_STAT is set. Overflow interrupt is generated if the ROVFIE bit in the ADC_CTL0 is set.

It is recommended to reinitialize the DMA module to recover the ADC from ROVF state. To ensure the routine converted data are transferred correctly, the internal state machine is reset. The ADC conversion will be stalled until the ROVF bit is cleared.

Software procedure for recovering the ADC from ROVF state:

1. Clear DMA bit of ADC_CTL1 to 0;
2. Clear ADCON bit of ADC_CTL1 to 0;
3. Clear CHEN bit of DMA_CHxCTL to 0 with reinit DMA module;
4. Clear ROVF bit of ADC_STAT to 0;
5. Set CHEN bit of DMA_CHxCTL to 1;
6. Set DMA bit of ADC_CTL1 to 1;
7. Set ADCON bit of ADC_CTL1 to 1;

8. Wait T(setup);
9. Start conversion with software or trigger.

12.4.11. ADC internal channels

When the TSVREN bit in ADC_CCTL register is set, the temperature sensor channel (ADC_IN9) and V_{REFINT} channel (ADC_IN10) are enabled. The temperature sensor can be used to measure the ambient temperature of the device. The sensor output voltage can be converted into a digital value by ADC. The sampling time for the temperature sensor is recommended to be set to at least 17.1μs. When this sensor is not in use, it can be set in power down mode by resetting the TSVREN bit.

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

The internal reference voltage (V_{REFINT}) provides a stable (bandgap) voltage output for the ADC and comparators. V_{REFINT} is internally connected to the ADC_IN10 input channel.

To use the temperature sensor:

1. Configure the conversion sequence (ADC_IN9) and the sampling time (17.1μs) for the channel.
2. Enable the temperature sensor by setting the TSVREN bit in ADC_CCTL.
3. Start the ADC conversion by setting the ADCON bit or by the triggers.
4. Read the internal temperature sensor output voltage ($V_{temperature}$), and get the temperature with the following equation:

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

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

Avg_Slope: Average Slope for curve between Temperature vs. internal temperature sensor output voltage, the typical value is refer to the datasheet.

12.4.12. Programmable resolution (DRES)

The resolution is configured by programming the DRES[1:0] bits in the ADC_OVSAMPCTL register. For applications that do not require high data accuracy, lower resolution allows faster conversion time. The DRES[1:0] bits must only be changed when the ADCON bit is reset. Lower resolution reduces the conversion time needed for the successive approximation steps as shown in *Table 12-5. tCONV timings depending on resolution.*

Table 12-5. t_{CONV} timings depending on resolution

DRES[1:0] bits	t_{CONV} (ADC clock	t_{CONV}(ns) at f_{ADC}=42MHz	t_{SMPL}(min) (ADC clock	t_{ADC} (ADC clock	t_{ADC}(us) at f_{ADC}=42MHz

	cycles)		cycles)	cycles)	
12	12	310 ns	2	14	333 ns
10	10	262 ns	2	12	286 ns
8	8	214 ns	2	10	238 ns
6	6	167 ns	2	8	190 ns

12.4.13. 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.

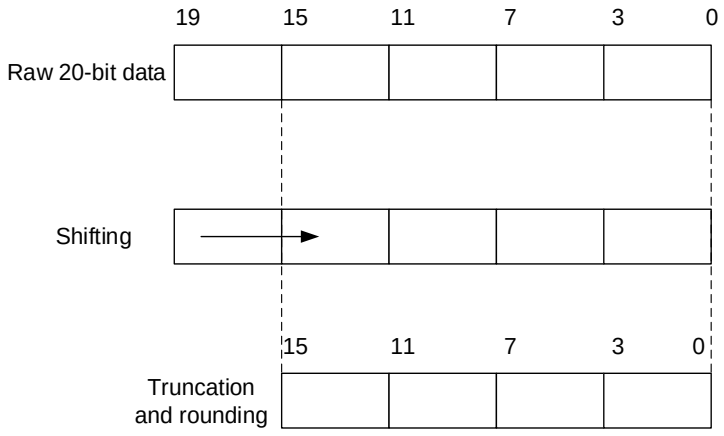
It provides a result with the following form, where N and M can be adjusted, and Dout(n) is the n-th output digital signal of the ADC:

$$Result=\frac{1}{M}*\sum_{n=0}^{n=N-1} D_{OUT}(n) \quad (14\text{-}1)$$

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 12-11. 20-bit to 16-bit result truncation



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

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

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

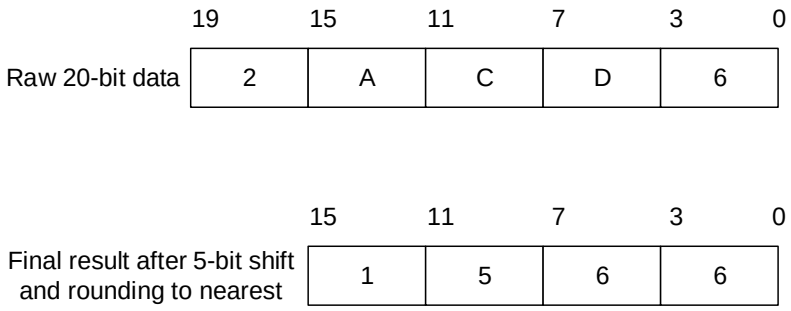


Table 12-6. Maximum output results for N and M combimations (grayed values indicates truncation) below gives the data format for the various N and M combinations, and the raw conversion data equals 0xFFF.

Table 12-6. 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 is maintained the same as that of standard conversion mode during the whole oversampling sequence. New data are provided every N conversion, with an equivalent delay equal to:

$$N \times t_{ADC} = N \times (t_{SMPL} + t_{CONV}) \quad (14\text{-}2)$$

12.4.14. 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).
- Overflow event.

12.5. Register definition

ADC base address: 0x4001 2000

12.5.1. 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										ROVF	STRC	Reserved		EOC	WDE
										rc_w0	rc_w0			rc_w0	rc_w0

Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5	ROVF	Routine data register overflow 0: Routine data register not overflow 1: Routine data register overflow This bit is set by hardware when the routine data registers are overflow, in single mode or multi mode. This flag is only set when DMA is enabled or end of conversion mode is set to 1(EOCM=1). The recent routine data is lost when this bit is set. Cleared by software writing 0 to it.
4	STRC	Start flag of routine sequence conversion 0: Conversion is not started 1: Conversion is started Set by hardware when routine sequence conversion starts. Cleared by software writing 0 to it.
3:2	Reserved	Must be kept at reset value.
1	EOC	End flag of routine sequence conversion 0: No end of routine sequence 1: End of routine sequence Set by hardware at the end of a routine sequence conversion. Cleared by software writing 0 to it or by reading the ADC_RDATA register.
0	WDE	Analog watchdog event flag 0: 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.

12.5.2. Control register 0 (ADC_CTL0)

Address offset: 0x04

Reset value: 0x0000 0000

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

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

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

Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value.
26	ROVFIE	Interrupt enable for ROVF 0: ROVF interrupt disable 1: ROVF interrupt enable
25:24	DRES[1:0]	ADC data resolution 00: 12bit; 01: 10bit; 10: 8bit; 11: 6bit
23	RWDEN	Routine channel analog watchdog enable 0: Routine channel analog watchdog disable 1: Routine channel analog watchdog enable
22:16	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 in routine sequence.
12	Reserved	Must be kept at reset value.
11	DISRC	Discontinuous mode on routine channels 0: Discontinuous operation mode disable 1: Discontinuous operation mode enable
10	Reserved	Must be kept at reset value.
9	WDSC	When in scan mode, analog watchdog is effective on a single channel

		0: All channels have analog watchdog function 1: A single channel has analog watchdog function
8	SM	Scan mode 0: Scan operation mode disable 1: Scan operation mode enable
7	Reserved	Must be kept at reset value.
6	WDEIE	Interrupt enable for WDE 0: Interrupt disable 1: Interrupt enable
5	EOCIE	Interrupt enable for EOC 0: EOC interrupt disable 1: EOC 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 Other values are reserved. **Note:** ADC analog inputs Channel 9 and Channel 10 are internally connected to the temperature sensor and V_{REFINT}.

12.5.3. 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	SWRCST	ETMRC[1:0]		ETSRC[3:0]				Reserved							
	rw	rw		rw											

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

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	SWRCST	Software start conversion of routine sequence. Setting 1 on this bit starts a conversion of a routine sequence. It is set by software and cleared by software or by hardware after the conversion starts.
29:28	ETMRC[1:0]	External trigger mode for routine channel 00: External trigger for routine channel disable 01: Rising edge of external trigger for routine channel enable 01: Falling edge of external trigger for routine channel enable 11: Rising and falling edge of external trigger for routine channel enable
27:24	ETSRC[3:0]	External trigger select for routine channel 0000: TIMER0 CH0 0001: TIMER0 CH1 0010: TIMER0 CH2 0011: TIMER1 CH1 0100: TIMER1 CH2 0101: TIMER1 CH3 0110: TIMER1 TRGO 0111: TIMER2 CH0 1000: TIMER2 TRGO 1001: TIMER2 CH2 1010: TIMER15 CH0 1011: TIMER16 CH0 1100: Reserved 1101: Reserved 1110: TIMER5 TRGO 1111: EXTI line 11
23:12	Reserved	Must be kept at reset value.
11	DAL	Data alignment 0: right alignment 1: left alignment
10	EOCM	End of conversion mode 0: Only at the end of a sequence of routine conversions,the EOC bit is set. Overflow detection is disabled unless DMA=1. 1: At the end of each routine conversion,the EOC bit is set. Overflow is detected automatically
9	DDM	DMA disable mode This bit configure the DMA disable mode for single ADC mode 0: The DMA engine is disabled after the end of transfer signal from DMA controller is detected.

		1: When DMA=1, the DMA engine issues a request at end of each routine conversion.
8	DMA	DMA request enable for routine channel. 0: DMA request disable 1: DMA request enable
7:2	Reserved	Must be kept at reset value.
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. 0: ADC disable and power down 1: ADC enable

12.5.4. Sample time register 0 (ADC_SAMPT0)

Address offset: 0x0C
Reset value: 0x0000 0000

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SPT10[3:0]				SPT9[3:0]				SPT8[3:0]			
				rw				rw				rw			

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value
11:8	SPT10[3:0]	refer to SPT8[3:0] description
7:4	SPT9[3:0]	refer to SPT8[3:0] description
3:0	SPT8[3:0]	Channel sample time 0000: 1.5 cycles 0001: 2.5 cycles 0010: 14.5 cycles 0011: 27.5 cycles 0100: 55.5 cycles 0101: 83.5 cycles 0110: 111.5 cycles

0111: 143.5 cycles

1000：479.5 cycles

12.5.5. Sample time register 1 (ADC_SAMPT1)

Address offset: 0x10

Reset value: 0x0000 0000

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SPT7[3:0]				SPT6[3:0]				SPT5[3:0]				SPT4[3:0]			
rw				rw				rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPT3[3:0]				SPT2[3:0]				SPT1[3:0]				SPT0[3:0]			
rw				rw				rw				rw			

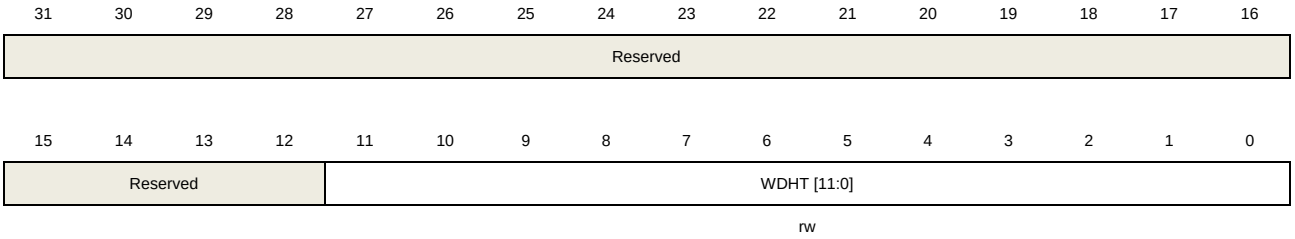
Bits	Fields	Descriptions
31:28	SPT7[3:0]	Refer to SPT0[3:0] description
27:24	SPT6[3:0]	Refer to SPT0[3:0] description
23:20	SPT5[3:0]	Refer to SPT0[3:0] description
19:16	SPT4[3:0]	Refer to SPT0[3:0] description
15:12	SPT3[3:0]	Refer to SPT0[3:0] description
11:8	SPT2[3:0]	Refer to SPT0[3:0] description
7:4	SPT1[3:0]	Refer to SPT0[3:0] description
3:0	SPT0[3:0]	Channel sample time 0000: 1.5 cycles 0001: 2.5 cycles 0010: 14.5 cycles 0011: 27.5 cycles 0100: 55.5 cycles 0101: 83.5 cycles 0110: 111.5 cycles 0111: 143.5 cycles 1000：479.5 cycles

12.5.6. Watchdog high threshold register (ADC_WDHT)

Address offset: 0x24

Reset value: 0x0000 0FFF

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



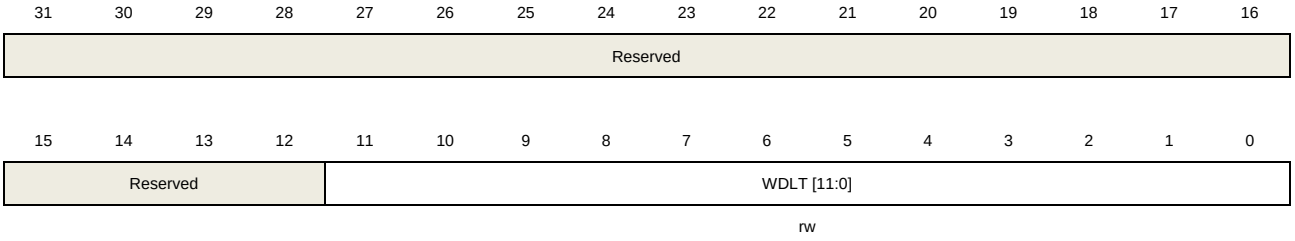
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	WDHT[11:0]	High threshold for analog watchdog These bits define the high threshold for the analog watchdog.

12.5.7. Watchdog low threshold register (ADC_WDLT)

Address offset: 0x28

Reset value: 0x0000 0000

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



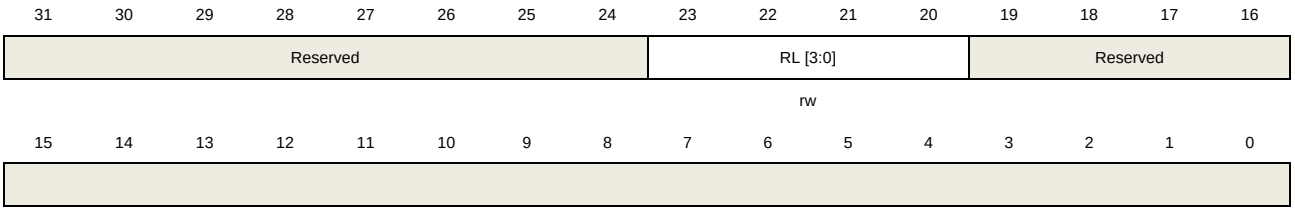
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	WDLT[11:0]	Low threshold for analog watchdog These bits define the low threshold for the analog watchdog.

12.5.8. Routine sequence register 0 (ADC_RSQ0)

Address offset: 0x2C

Reset value: 0x0000 0000

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



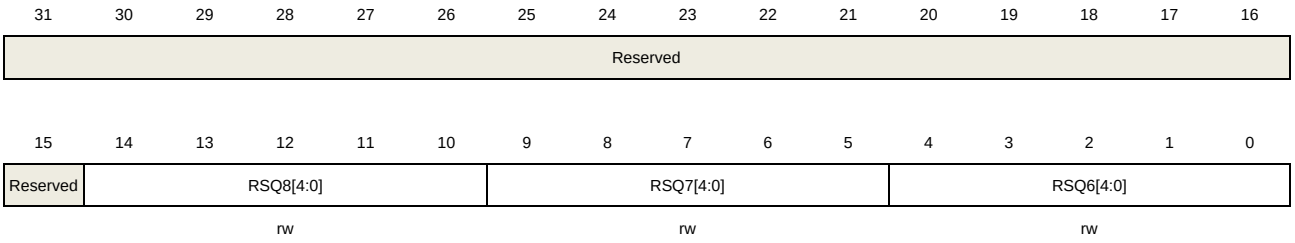
Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:20	RL[3:0]	Routine sequence length The total number of conversion in Routine sequence equals to RL[3:0] +1.
19:0	Reserved	Must be kept at reset value.

12.5.9. Routine sequence register 1 (ADC_RSQ1)

Address offset: 0x30

Reset value: 0x0000 0000

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



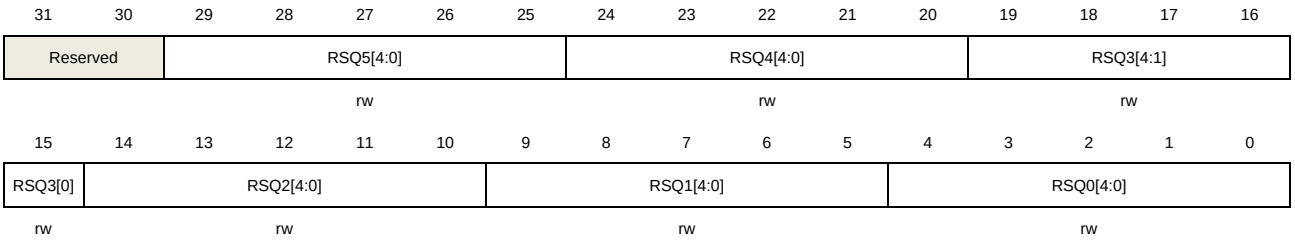
Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14:10	RSQ8[4:0]	Refer to RSQ0[4:0] description
9:5	RSQ7[4:0]	Refer to RSQ0[4:0] description
4:0	RSQ6[4:0]	Refer to RSQ0[4:0] description

12.5.10. 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..10) is written to these bits to select a channel as the nth conversion in the routine sequence.

12.5.11. Routine data register (ADC_RDATA)

Address offset: 0x4C
Reset value: 0x0000 0000

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
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 routine channel conversion value, which is read only.

12.5.12. 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:** Software is allowed to write this bit only when ADCON=0 (which ensures that no conversion is ongoing).
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:** Software is allowed to write this bit only when ADCON =0 (which ensures that no conversion is ongoing).
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:** Software is allowed to write this bit only when ADCON =0 (which ensures that no conversion is ongoing).
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:** Software is allowed to write this bit only when ADCON =0 (which ensures that no conversion is ongoing).

12.5.13. Commom control register (ADC_CCTL)

Address offset: 0x304

Reset value: 0x0000 0000

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

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

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

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23	TSVREN	Channel 9 (temperature sensor) and 10 (internal reference voltage) enable of ADC. 0: Channel 9 and 10 of ADC disable 1: Channel 9 and 10 of ADC enable
22:19	Reserved	Must be kept at reset value.
18:16	ADCCK[2:0]	ADC clock These bits configure the ADC clock. 000: PCLK2 / 2 001: PCLK2 / 4 010: PCLK2 / 6 011: PCLK2 / 8 100: HCLK / 5 101: HCLK / 6 110: HCLK / 10 111: HCLK / 20
15:0	Reserved	Must be kept at reset value.

13. 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 (or an interrupt in window watchdog timer) when the internal counter reaches a given value. The watchdog timer counter can be stopped while the processor is in the debug mode.

13.1. Free watchdog timer (FWDGT)

13.1.1. Overview

The free watchdog timer (FWDGT) has free clock source (IRC32K). Thereupon the FWDGT can operate even if the main clock fails. It's suitable for the situation that requires an independent environment and lower timing accuracy.

The free watchdog timer causes a reset when the internal down counter reaches 0. The register write protection function in free watchdog timer can be enabled to prevent it from changing the configuration unexpectedly.

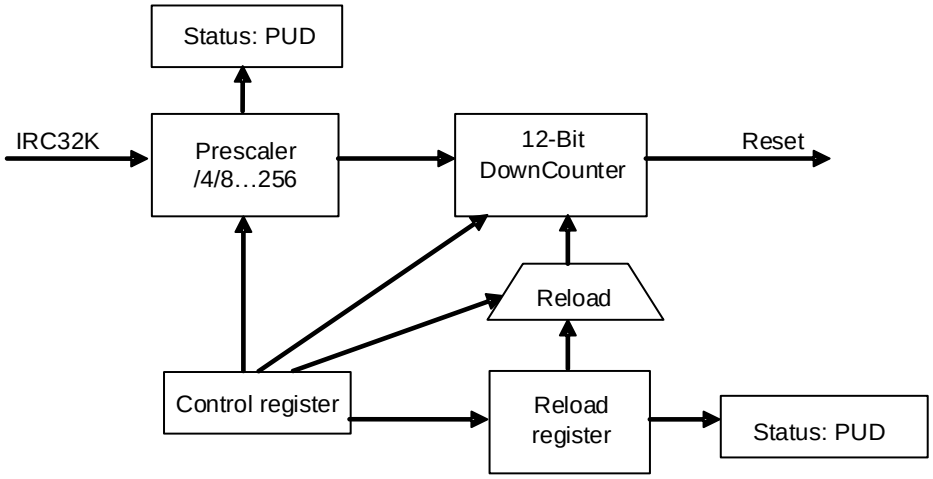
13.1.2. Characteristics

- Free-running 12-bit downcounter.
- Reset when the downcounter reaches 0, if the watchdog is enabled.
- Free clock source，FWDGT can operate even if the main clock fails such as in standby and Deep-sleep modes.
- Hardware free watchdog timer bit, automatically start the FWDGT at power on.
- FWDGT debug mode, the FWDGT can stop or continue to work in debug mode.

13.1.3. Function overview

The free watchdog timer consists of an 8-stage prescaler and a 12-bit down-counter.***Figure 13-1. Free watchdog timer block diagram*** shows the functional block of the free watchdog timer module.

Figure 13-1. Free watchdog timer block diagram



The free watchdog timer is enabled by writing the value 0xCCCC to the control register (FWDGT_CTL), then the counter starts counting down. When the counter reaches the value 0x000, there will be a reset.

The counter can be reloaded by writing the value (0xAAAA) to the FWDGT_CTL register at anytime. The reload value comes from the FWDGT_RLD register. The software can prevent the watchdog reset by reloading the counter before the counter reaches the value 0x000.

The FWDGT_PSC register and the FWDGT_RLD register are write protected. Before writing these registers, the software should write the value (0x5555) to the FWDGT_CTL register. These registers will be protected again by writing any other value to the FWDGT_CTL register. When an update operation of the prescaler register (FWDGT_PSC) or the reload value register (FWDGT_RLD) is ongoing, the status bits in the FWDGT_STAT register are set.

If the FWDGT_HOLD bit in DBG module is cleared, the FWDGT continues to work even the RISC-V core halted (Debug mode). While the FWDGT stops in Debug mode if the FWDGT_HOLD bit is set.

Table 13-1. Min/max FWDGT timeout period at 32 kHz (IRC32K)

Prescaler divider	PSC[2:0] bits	Min timeout (ms) RLD[11:0]=0x000	Max timeout (ms) RLD[11:0]=0xFFF
1/4	000	0.03125	511.90625
1/8	001	0.03125	1023.78125
1/16	010	0.03125	2047.53125
1/32	011	0.03125	4095.03125
1/64	100	0.03125	8190.03125
1/128	101	0.03125	16380.03125
1/256	110 or 111	0.03125	32760.03125

The FWDGT timeout can be more accurate by calibrating the IRC32K.

Note： When after the execution of dog reload operation, if the MCU needs enter the

deepsleep / standby mode immediately, more than 3 IRC32K clock interval must be inserted in the middle of reload and deepsleep / standby mode commands by software setting.

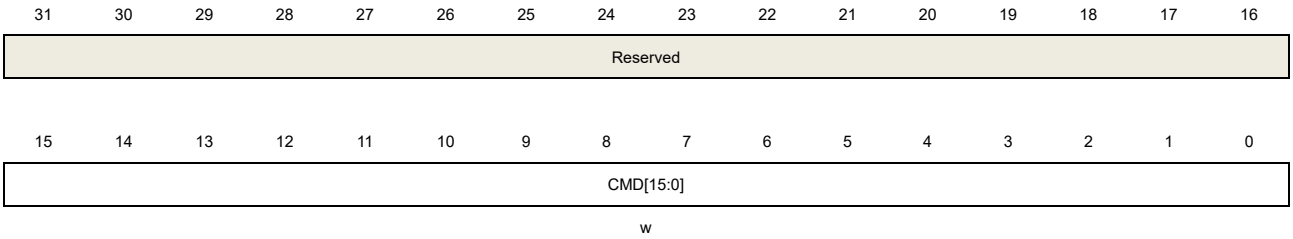
13.1.4. Register definition

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).

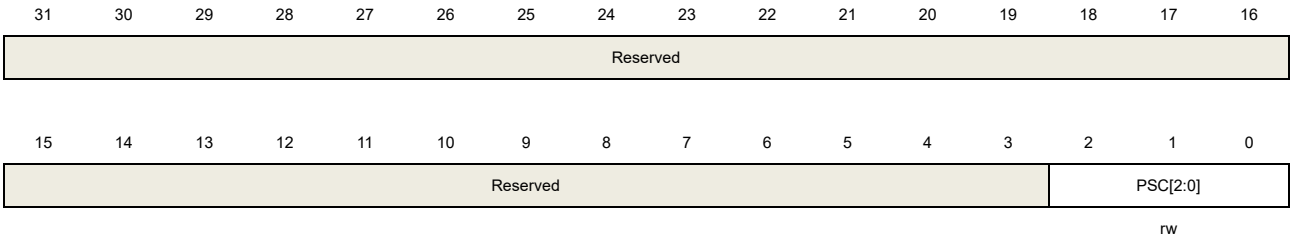


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CMD[15:0]	Write only. Several different fuctions are realized by writing these bits with different values: 0x5555: Disable the FWDGT_PSC and FWDGT_RLD write protection 0xCCCC: Start the free watchdog timer counter. When the counter reduces to 0, the free watchdog timer 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).



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 is reset before changing the prescaler value. However, after updating the prescaler value it is not necessary to wait until PUD is reset before continuing code execution.

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).

31	30	29	28	27	26	25	24	23	22	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 counter with the RLD value. These bits are write-protected. Write 0x5555 in 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 is reset before changing the reload value. However, after updating the reload value it is not necessary to wait until RUD is reset before continuing code execution.

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).

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

Reserved

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

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	RUD	Free watchdog timer counter reload value update During a write operation to FWDGT_RLD register, this bit is set and the value read from FWDGT_RLD register is invalid. This bit is reset by hardware after the update operation of FWDGT_RLD register.
0	PUD	Free watchdog timer prescaler value update During a write operation to FWDGT_PSC register, this bit is set and the value read from FWDGT_PSC register is invalid. This bit is reset by hardware after the update operation of FWDGT_PSC register.

13.2. Window watchdog timer (WWDGT)

13.2.1. Overview

The window watchdog timer (WWDGT) is used to detect system failures due to software malfunctions. After the window watchdog timer starts, the value of down counter reduces progressively. The watchdog timer causes a reset when the counter reached 0x3F (the CNT[6] bit has been cleared). The watchdog timer also causes a reset when the counter is refreshed before the counter reached the window register value. So the software should refresh the counter in a limited window. The window watchdog timer generates an early wakeup status flag when the counter reaches 0x40 or refreshes before the counter reaches the window value. Interrupt occurs if it is enabled.

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.

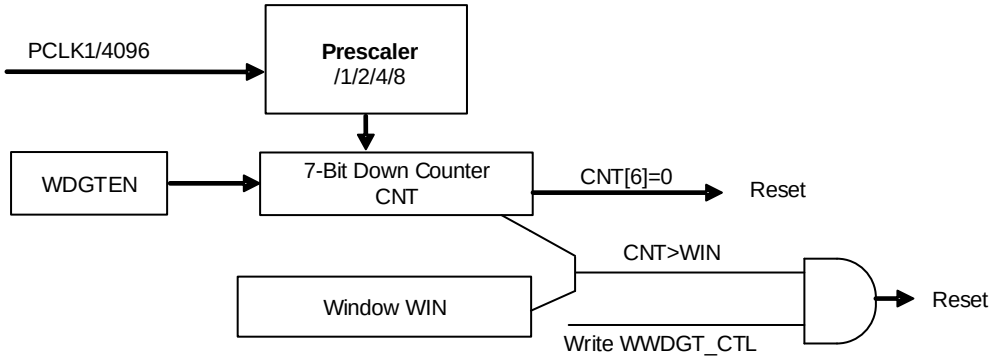
13.2.2. Characteristics

- Programmable free-running 7-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): if the watchdog is started and the interrupt is enabled, the interrupt occurs when the counter reaches 0x40 or refreshes before it reaches the window value.
- WWDGT debug mode, the WWDGT can stop or continue to work in debug mode.

13.2.3. Function overview

If the window watchdog timer is enabled (set the WDGTEN bit in the WWDGT_CTL), the watchdog timer cause a reset when the counter reaches 0x3F (the CNT[6] bit has been cleared), or the counter is refreshed before the counter reaches the window register value.

Figure 13-2. Window watchdog timer block diagram



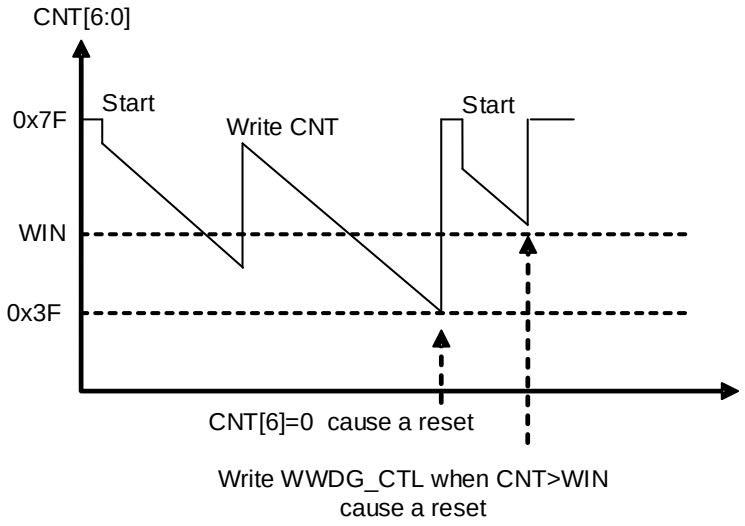
The watchdog is always disabled after power on reset. The software starts the watchdog by setting the WDGTEN bit in the WWDGT_CTL register. When window watchdog timer is enabled, the counter counts down all the time, the configured value of the counter should be greater than 0x3F(it implies that the CNT[6] bit should be set). The CNT[5:0] determine the maximum time interval between two reloading. The count down speed depends on the APB1 clock and the prescaler (PSC[1:0] bits in the WWDGT_CFG register).

The WIN[6:0] bits in the configuration register (WWDGT_CFG) 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 or the counter is refreshed before it reaches the window value. 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 13-3. Window watchdog timing diagram



Calculate the WWDGT timeout by using the formula below.

$$t_{WWDGT}=t_{PCLK1} \times 4096 \times 2^{PSC} \times (CNT[5:0]+1) \quad (ms) \qquad (13\text{-}1)$$

where:

t_{WWDGT}: WWDGT timeout

t_{PCLK1}: APB1 clock period measured in ms

Table 13-2. Min-max timeout value at 42 MHz (fPCLK1) shows the minimum and maximum values of the t_{WWDGT}.

Table 13-2. Min-max timeout value at 42 MHz (f_{PCLK1})

Prescaler divider	PSC[1:0]	Min timeout value CNT[6:0] =0x40	Max timeout value CNT[6:0]=0x7F
1/1	00	97.52 µs	6.24 ms
1/2	01	195.04 µs	12.48 ms
1/4	10	390.08 µs	24.96 ms
1/8	11	780.16 µs	49.92 ms

If the WWDGT_HOLD bit in DBG module is cleared, the WWDGT continues to work even the RISC-V core halted (Debug mode). While the WWDGT_HOLD bit is set, the WWDGT stops in Debug mode.

13.2.4. Register definition

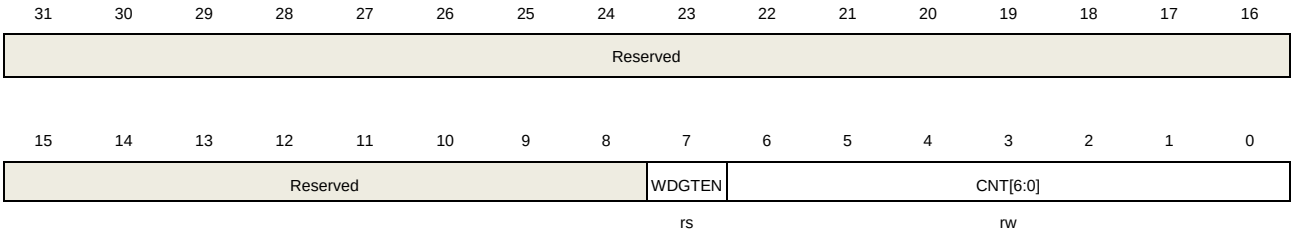
WWDG base address: 0x4000 2C00

Control register (WWDGT_CTL)

Address offset: 0x00

Reset value: 0x0000 007F

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



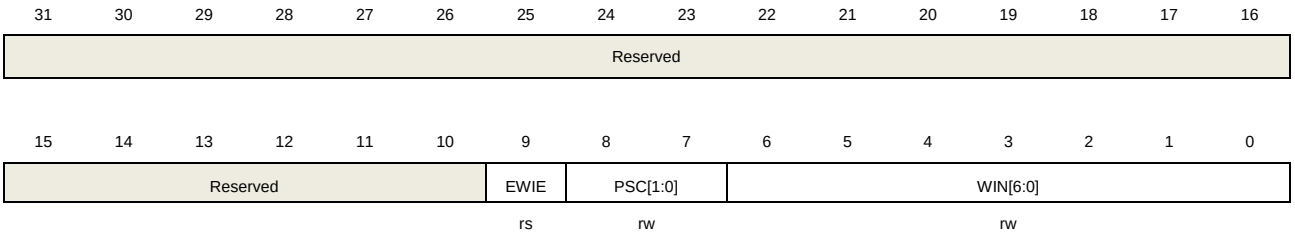
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	WDGTEN	Start the Window watchdog timer. Cleared by a hardware reset. Writing 0 has no effect. 0: Window watchdog timer disabled 1: Window watchdog timer enabled
6:0	CNT[6:0]	The value of the watchdog timer counter. A reset occur when 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: 0x0000 007F

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



Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	EWIE	Early wakeup interrupt enable. If the bit is set, an interrupt occurs when the counter reaches 0x40. It can be cleared by a hardware reset or software clock reset. A write

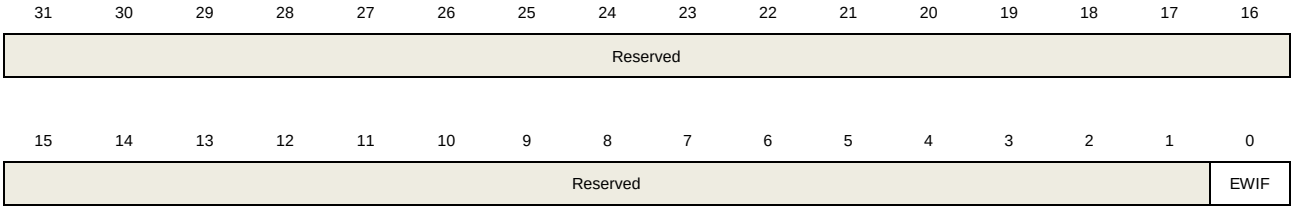
		operation of 0 has no effect.
8:7	PSC[1:0]	Prescaler. The time base of the watchdog counter 00: (PCLK1 / 4096) / 1 01: (PCLK1 / 4096) / 2 10: (PCLK1 / 4096) / 4 11: (PCLK1 / 4096) / 8
6:0	WIN[6:0]	The Window value. A reset occur if the watchdog counter (CNT bits in WWDGT_CTL) is written when the value 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 half-word(16-bit) or word(32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	EWIF	Early wakeup interrupt flag. When the counter reaches 0x40 or refreshes before it reaches the window value, 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.

14. Real time clock (RTC)

14.1. Overview

The RTC provides a time which includes hour / minute / second / sub-second and a calendar includes year / month / day / week day. The time and calendar are expressed in BCD code except sub-second. Sub-second is expressed in binary code. Hour adjust for daylight saving time. Working in power saving mode and smart wakeup is software configurable. Support improving the calendar accuracy using extern accurate low frequency clock.

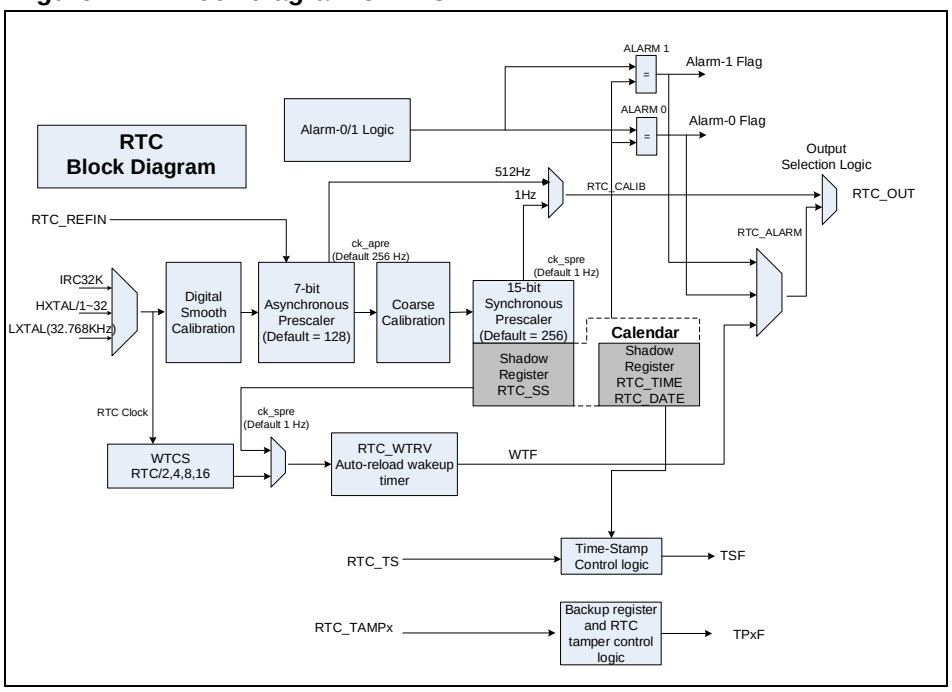
14.2. Characteristics

- Daylight saving compensation supported by software.
- External high-accurate low frequency (50Hz or 60Hz) clock used to achieve higher calendar accuracy performed by reference clock detection option function.
- Atomic clock adjust(max adjust accuracy is 0.95PPM) for calendar calibration performed by digital calibration function.
- Sub-second adjustment by shift function.
- Time-stamp function for saving event time.
- Two Tamper sources can be chosen and tamper type is configurable.
- Programmable calendar and two field maskable alarms.
- Maskable interrupt source:
  - Alarm 0 and Alarm 1
  - Time-stamp detection
  - Tamper detection
  - Auto wakeup event
- Twenty 32-bit (80 bytes total) universal backup registers which can keep data under power saving mode. Backup register will be reset if tamper event detected.

14.3. Function overview

14.3.1. Block diagram

Figure 14-1. Block diagram of RTC



The RTC unit includes:

- Alarm event / interrupt.
- Tamper event / interrupt.
- 32-bit backup registers.
- Optional RTC output function:
  - 512Hz (default prescale) : PC13 / PA3 / PA8
  - 1Hz (default prescale) : PC13 / PA3 / PA8
  - Alarm event (polarity is configurable) : PC13 / PA3 / PA8
  - Automatic wakeup event (polarity is configurable) : PC13 / PA3 / PA8
- Optional RTC input function:
  - Time stamp event detection (RTC_TS) : PC13
  - Tamper 0 event detection (RTC_TAMP0) : PC13
  - Tamper 1 event detection (RTC_TAMP0) : PA0
  - Reference clock input RTC_REFIN (50 or 60 Hz)

It is possible to output RTC_OUT on PA3 or PA8 pin thanks to OUT2EN bit in RTC_CTL[31].

In addition, the TAMPER 1 alternate function corresponds to PA0 pin.

14.3.2. Clock source and prescalers

RTC unit has three independent clock sources: LXTAL, IRC32K and HXTAL with divided by

1 ~ 32(configured in RCU_CFG0 register).

In the RTC unit, there are two prescalers used for implementing the calendar and other functions. One prescaler is a 7-bit asynchronous prescaler and the other is a 15-bit synchronous prescaler. Asynchronous prescaler is mainly used for reducing power consumption. The asynchronous prescaler is recommended to set as high as possible if both prescalers are used.

The frequency formula of two prescalers is shown as below:

$$f_{ck_apre} = \frac{f_{rtcclk}}{FACTOR_A + 1} \quad (14\text{-}1)$$

$$f_{ck_spre} = \frac{f_{ck_apre}}{FACTOR_S + 1} = \frac{f_{rtcclk}}{(FACTOR_A + 1)*(FACTOR_S + 1)} \quad (14\text{-}2)$$

The ck_apre clock is used to driven the RTC_SS down counter which stands for the time left to next second in binary format and when it reaches 0 it will automatically reload FACTOR_S value. The ck_spre clock is used to driven the calendar registers. Each clock will make second plus one.

14.3.3. Shadow registers introduction

BPSHAD control bit decides the location when APB bus accesses the RTC calendar register RTC_DATE, RTC_TIME and RTC_SS. By default, the BPSHAD is cleared, and APB bus accesses the shadow calendar registers. Shadow calendar registers is updated with the value of real calendar registers every two RTC clock and at the same time RSYNF bit will be set once. This update mechanism is not performed in Deep-Sleep mode and Standby mode. When exiting these modes, software must clear RSYNF bit and wait it is asserted (the max wait time is 2 RTC clock) before reading calendar register under BPSHAD=0 situation.

Note: When reading calendar registers (RTC_SS, RTC_TIME, RTC_DATE) under BPSHAD=0, the frequency of the APB clock (fapb) must be at least 7 times the frequency of the RTC clock (frtcclk).

System reset will reset the shadow calendar registers.

14.3.4. Configurable and field maskable alarm

RTC alarm function is divided into some fields and each has a maskable bit.

RTC alarm function can be enabled or disabled by ALRMxEN bit in RTC_CTL. If all the alarm fields value match the corresponding calendar value when ALRMxEN=1, the Alarm flag will be set.

Note: FACTOR_S in the RTC_PSC register must be larger than 3 if MSKS bit reset in RTC_ALRMxTD.

If a field is masked, the field is considered as matched in logic. If all the fields have been masked, the Alarm Flag will assert 3 RTC clock later after ALRMxEN is set.

14.3.5. Configurable periodic auto-wakeup counter

In the RTC block, there is a 16-bit down counter designed to generate periodic wakeup flag.

This function is enabled by set the WTEN to 1 and can be running in power saving mode.

Two clock sources can be chose for the down counter:

1) RTC clock divided by 2 / 4 / 8 / 16

   Assume RTC clock comes from LXTAL (32.768 KHz), this can periodically assert wakeup interrupt from 122us to 32s under the resolution down to 61us.

2) Internal clock ck_spre

   Assume ck_spre is 1Hz, this can periodically assert wakeup interrupt from 1s to 36 hours under the resolution down to 1s.

   - WTCS[2:1] = 0b10.This will make period to be 1s to 18 hours
   - WTCS[2:1] = 0b11.This will make period to be 18 to 36 hours

When this function is enabled, the down counter is running. When it reaches 0, the WTF flag is set and the wakeup counter is automatically reloaded with RTC_WUT value.

When WTF asserts, software must then clear it.

If WTIE is set and this counter reaches 0, a wakeup interrupt will make system exit from the power saving mode. System reset has no influence on this function.

14.3.6. RTC initialization and configuration

RTC register write protection

BKPWEN bit in the PMU_CTL register is cleared in default, so writing to RTC registers needs setting BKPWEN bit ahead of time.

After power-on reset, most of RTC registers are write protected. Unlocking this protection is the first step before writing to them.

Following below steps will unlock the write protection:

1. Write ‘0xCA’ into the RTC_WPK register.
2. Write ‘0x53’ into the RTC_WPK register.

Writing a wrong value to RTC_WPK will make write protection valid again.

After backup domain reset, some of the RTC registers are write-protected: RTC_TIME, RTC_DATE, RTC_PSC, RTC_COSC, RTC_HRFC, RTC_SHIFTCTL, the bit INITM in RTC_STAT and the bits CS, S1H ,A1H，REFEN in RTC_CTL.

Calendar initialization and configuration

The prescaler and calendar value can be programmed by the following steps:

1. Enter initialization mode (by setting INITM=1) and polling INITF bit until INITF=1.
2. Program both the asynchronous and synchronous prescaler factors in RTC_PSC register.
3. Write the initial calendar values into the shadow calendar registers (RTC_TIME and RTC_DATE), and use the CS bit in the RTC_CTL register to configure the time format (12 or 24 hours).
4. Exit the initialization mode (by setting INITM=0).

About 4 RTC clock cycles later, real calendar registers will load from shadow registers and calendar counter restarts.

Note: Reading calendar register (BPSHAD=0) after initialization, software should confirm the RSYNF bit to 1.

YCM flag indicates whether the calendar has been initialized by checking the year field of calendar.

Daylight saving Time

RTC unit supports daylight saving time adjustment through S1H, A1H and DSM bit.

S1H and A1H can subtract or add 1 hour to the calendar when the calendar is running.S1H and A1H operation can be tautologically set and DSM bit can be used to recording this adjust operation. After setting the S1H / A1H, subtract / add 1 hour will perform when next second comes.

Alarm function operation process

To avoid unexpected alarm assertion and metastable state, alarm function has an operation flow:

1. Disable alarm (by resetting ALRMxEN in RTC_CTL).
2. Set the alarm registers needed(RTC_ALRMxTD / RTC_ALRMxSS).
3. Enable alarm function (by setting ALRMxEN in the RTC_CTL).

14.3.7. Calendar reading

Reading calendar registers under BPSHAD=0

When BPSHAD=0, calendar value is read from shadow registers. For the existence of synchronization mechanism, a basic request has to meet: the APB1 bus clock frequency must be equal to or greater than 7 times the RTC clock frequency.APB1 bus clock frequency lower than RTC clock frequency is not allowed in any case whatever happens.

When APB1 bus clock frequency is not equal to or greater than 7 times the RTC clock frequency, the calendar reading flow should be obeyed:

1. Reading calendar time register and date register twice.
2. If the two values are equal, the value can be seen as the correct value.

3. If the two values are not equal, a third reading should performed.
4. The third value can be seen as the correct value.

RSYNF is asserted once every 2 RTC clock and at this time point, the shadow registers will be updated to current time and date.

To ensure consistency of the 3 values (RTC_SS, RTC_TIME, and RTC_DATE), below consistency mechanism is used in hardware:

1. Reading RTC_SS will lock the updating of RTC_TIME and RTC_DATE.
2. Reading RTC_TIME will lock the updating of RTC_DATE.
3. Reading RTC_DATE will unlock updating of RTC_TIME and RTC_DATE.

If the software wants to read calendar in a short time interval(smaller than 2 RTCCLK periods), RSYNF must be cleared by software after the first calendar read, and then the software must wait until RSYNF is set again before next reading.

In below situations, software should wait RSYNF bit asserted before reading calendar registers (RTC_SS, RTC_TIME, and RTC_DATE):

1. After a system reset
2. After an initialization
3. After shift function

Especially that software must clear RSYNF bit and wait it asserted before reading calendar register after wakeup from power saving mode.

Reading calendar registers under BPSHAD=1

When BPSHAD=1, RSYNF is cleared and maintains as 0 by hardware so reading calendar registers does not care about RSYNF bit. Current calendar value is read from real-time calendar counter directly. The benefit of this configuration is that software can get the real current time without any delay after wakeup from power saving mode (Deep-sleep / Standby Mode).

Because of no RSYNF bit periodic assertion, the results of the different calendar registers (RTC_SS / RTC_TIME / RTC_DATE) might not be coherent with each other when clock ck_apre edge occurs between two reading calendar registers.

In addition, if current calendar register is changing and at the same time the APB bus reading calendar register is also performing, the value of the calendar register read out might be not correct.

To ensure the correctness and consistency of the calendar value, software must perform reading operation as this: read all calendar registers continuously, if the last two values are the same, the data is coherent and correct.

14.3.8. Resetting the RTC

There are two reset sources used in RTC unit: system reset and backup domain reset.

System reset will affect calendar shadow registers and some bits of the RTC_STAT. When system reset is valid, the bits or registers mentioned before are reset to the default value.

Backup domain reset will affect the following registers and system reset will not affect them:

- RTC current real-time calendar registers
- RTC Control register (RTC_CTL)
- RTC Prescaler register (RTC_PSC)
- RTC Wakeup timer register (RTC_WUT)
- RTC Coarse calibration register (RTC_COSC)
- RTC High resolution frequency compensation register (RTC_HRFC)
- RTC Shift control register (RTC_SHIFTCTL)
- RTC Time stamp registers (RTC_SSTS / RTC_TTS / RTC_DTS)
- RTC Tamper register (RTC_TAMP)
- RTC Backup registers (RTC_BKPx)
- RTC Alarm registers (RTC_ALRMxSS / RTC_ALRMxTD)

The RTC unit will go on running when system reset occurs or enter power saving mode, but if backup domain reset occurs, RTC will stop counting and all registers will reset.

14.3.9. RTC shift function

When there is a remote clock with higher degree of precision and RTC 1Hz clock (ck_spre) has an offset (in a fraction of a second) with the remote clock, RTC unit provides a function named shift function to remove this offset and thus make second precision higher.

RTC_SS register indicates the fraction of a second in binary format and is down counting when RTC is running. Therefore by adding the SFS[14:0] value to the synchronous prescaler counter SSC[15:0] or by adding the SFS[14:0] value to the synchronous prescaler counter SSC[15:0] and at the same time set A1S bit can delay or advance the time when next second arrives.

The maximal RTC_SS value depends on the FACTOR_S value in RTC_PSC. The higher FACTOR_S, the higher adjust precision.

Because of the 1Hz clock (ck_spre) is generated by FACTOR_A and FACTOR_S, the higher FACTOR_S means the lower FACTOR_A, then more power consuming.

Note: Before using shift function, the software must check the MSB of SSC in RTC_SS (SSC[15]) and confirm it is 0.

After writing RTC_SHIFTCTL register, the SOPF bit in RTC_STAT will be set at once. When shift operation is complete, SOPF bit is cleared by hardware. System reset does not affect SOPF bit.

Shift operation only works correctly when REFEN=0.

Software must not write to RTC_SHIFTCTL if REFEN=1.

14.3.10. RTC reference clock detection

RTC reference clock detection is another way to increase the precision of RTC second. To enable this function, you should have an external clock source (50Hz or 60 Hz) which is more precise than LXTAL clock source.

After enabling this function (REFEN=1), each 1Hz clock (ck_spre) edge is compared to the nearest RTC_REFIN clock edge. In most cases, the two clock edges are aligned every time. But when two clock edges are misaligned for the reason of LXTAL poor precision, the RTC reference clock detection function will shift the 1Hz clock edge a little to make next 1Hz clock edge aligned to reference clock edge.

When REFEN=1, a time window is applied at every second update time different detection state will use different window period.

7 ck_apre window is used when detecting the first reference clock edge and 3 ck_apre window is used for the edge aligned operation.

Whatever window used, the asynchronous prescaler counter will be forced to reload when the reference clock is detected in the window. When the two clock (ck_spre and reference clock) edges are aligned, this reload operation has no effect for 1Hz clock. But when the two clock edge are not aligned, this reload operation will shift ck_spre clock edge a bit to make the ck_spre(1Hz) clock edge aligned to the reference clock edge.

When reference detection function is running while the external reference clock is removed (no reference clock edge found in 3 ck_apre window), the calendar updating still can be performed by LXTAL clock only. If the reference clock is recovered later, detection function will use 7 ck_apre window to identify the reference clock and use 3 ck_apre window to adjust the 1Hz clock (ck_spre) edge.

Note: Software must configure the FACTOR_A=0x7F and FACTOR_S=0xFF before enabling reference detection function (REFEN=1).

Reference detection function does not work in Standby Mode and must not be used with coarse digital function.

14.3.11. RTC coarse digital calibration

There are two digital methods can be chose for calibration: coarse digital calibration and smooth digital calibration. These two types cannot be used together.

Coarse digital calibration can be used to add or mask ck_apre clock cycles at the output of the asynchronous prescaler.

When COSD=0, 2 ck_apre cycles are added every minute for the first 2xCOSS minutes. The

effect of such configuration will make calendar to be updated sooner.

When COSD=1, 1 ck_apre cycle is removed every minute for the first 2xCOSS minutes. The effect of such configuration will make calendar to be updated later.

Only in initialization mode can configure coarse calibration and the function starts after clearing INITM bit. The full calibration window lasts 64 minutes. The first 2xCOSS minutes of this 64-minute window are take adjust.

About 2PPM resolution is taken for negative calibration and about 4PPM resolution is taken for positive calibration.

Note: The calibration can be performed either on LXTAL or HXTAL clock. If FACTOR_A < 6, the calibration may not work correctly.

Example:

FACTOR_A and FACTOR_S are default values. LXTAL is the RTC clock source and frequency is 32.768 KHz.

During a calibration window (64 minutes), the ck_apre clock frequency is only adjusted in the first 2 x COSS minutes. If COSS=1, this means only the first 2 minutes of 64 minutes will make adjustment.

The calibration step therefore has the effect of adding 512 or subtracting 256 oscillator cycles for each calibration window (64min x 60s / min x 32768cycles / s). In another word, this is equivalent to +4.069PPM or -2.035PPM per calibration step. Then for one month running, the minimum calibration step is +10.5 or -5.27 seconds and the maximum calibration step is +5.45 to -2.72 minutes.

14.3.12. RTC smooth digital calibration

RTC smooth calibration function is a way to calibrate the RTC frequency based on RTC clock in a configurable period time.

This calibration is equally executed in a period time and the cycle number of the RTC clock in the period time will be added or subtracted. The resolution of the calibration is about 0.954PPM with the range from -487.1PPM to +488.5PPM.

The calibration period time can be configured to the 220 / 219 / 218 RTC clock cycles which stands for 32 / 16 / 8 seconds if RTC input frequency is 32.768 KHz.

The High resolution frequency compensation register (RTC_HRFC) specifies the number of RTCCLK clock cycles to be calibrated during the period time:

So using CMSK can mask clock cycles from 0 to 511 and thus the RTC frequency can be reduced by up to 487.1PPM.

To increase the RTC frequency the FREQI bit can be set. If FREQI bit is set, there will be 512 additional cycles to be added during period time which means every 211 / 210 / 29(32 / 16 / 8 seconds) RTC clock insert one cycle.

So using FREQI can increase the RTC frequency by 488.5PPM.

The combined using of CMSK and FREQI can adjust the RTC cycles from -511 to +512 cycles in the period time which means the calibration range is -487.1PPM to +488.5PPM with a resolution of about 0.954PPM.

When calibration function is running, the output frequency of calibration is calculated by the following formula:

$$f_{cal} = f_{rtcclk} \times \left(1 + \frac{FREQI \times 512 - CMSK}{2^N + CMSK - FREQI \times 512}\right) \quad (14\text{-}3)$$

Note: N=20 / 19 / 18 for 32 / 16 / 8 seconds window period

Calibration when FACTOR_A < 3

When asynchronous prescaler value (FACTOR_A) is set to less than 3, software should not set FREQI bit to 1 when using calibration function. FREQI setting will be ignored when FACTOR_A < 3.

When the FACTOR_A is less than 3, the FACTOR_S value should be set to a value less than the nominal value. Assuming that RTC clock frequency is nominal 32.768 KHz, the corresponding FACTOR_S should be set as following rule:

FACTOR_A = 2: 2 less than nominal FACTOR_S (8189 with 32.768 KHz)

FACTOR_A = 1: 4 less than nominal FACTOR_S (16379 with 32.768 KHz)

FACTOR_A = 0: 8 less than nominal FACTOR_S (32759 with 32.768 KHz)

When the FACTOR_A is less than 3, CMSK is 0x100，the formula of calibration frequency is as follows:

$$f_{cal} = f_{rtcclk} \times \left(1 + \frac{256 - CMSK}{2^N + CMSK - 256}\right) \quad (14\text{-}4)$$

Note: N = 20 / 19 / 18 for 32 / 16 / 8 seconds window period

Verifying the RTC calibration

Calibration 1Hz output is provided to assist software to measure and verify the RTC precision.

Up to 2 RTC clock cycles measurement error may occur when measuring the RTC frequency over a limited measurement period. To eliminate this measurement error the measurement period should be the same as the calibration period.

- When the calibration period is 32 seconds(this is default configuration)

Using exactly 32s period to measure the accuracy of the calibration 1Hz output can guarantee the measure is within 0.477PPM (0.5 RTCCLK cycles over 32s)

- When the calibration period is 16 seconds(by setting CWND16 bit)

In this configuration, CMSK[0] is fixed to 0 by hardware. Using exactly 16s period to measure the accuracy of the calibration 1Hz output can guarantee the measure is within 0.954PPM (0.5 RTCCLK cycles over 16s)

- When the calibration period is 8 seconds(by setting CWND8 bit)

In this configuration, CMSK[1:0] is fixed to 0 by hardware. Using exactly 8s period to measure the accuracy of the calibration 1Hz output can guarantee the measure is within 1.907PPM (0.5 RTCCLK cycles over 8s)

Re-calibration on-the-fly

When the INITF bit is 0, software can update the value of RTC_HRFC using following steps:

1) Wait the SCPF=0.
2) Write the new value into RTC_HRFC register.
3) After 3 ck_apre clocks, the new calibration settings take effect.

14.3.13. Time-stamp function

Time-stamp function is performed on RTC_TS pin and is enabled by control bit TSEN.

When a time-stamp event occurs on RTC_TS pin, the calendar value will be saved in time-stamp registers (RTC_DTS / RTC_TTS / RTC_SSTS) and the time-stamp flag (TSF) is set to 1 by hardware. Time-stamp event can generate an interrupt if time-stamp interrupt enable (TSIE) is set.

Time-stamp registers only record the calendar at the first time time-stamp event occurs which means that time-stamp registers will not change when TSF=1.

To extend the time-stamp event source, one optional feature is provided: tamper function can also be considered as time-stamp function if TPTS is set.

Note: When the time-stamp event occurs, TSF is set 2 ck_apre cycles delay because of synchronization mechanism.

14.3.14. Tamper detection

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

RTC backup registers (RTC_BKPx)

The RTC backup registers are located in the VDD 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

RTC tamper detection function can be independently enabled on tamper input pin by setting corresponding TPxEN bit. Tamper detection configuration is set before enable TPxEN bit. When the tamper event is detected, the corresponding flag (TPxF) will assert. Tamper event

can generate an interrupt if tamper interrupt enable (TPIE) is set.

The backup registers are reset when a tamper detection event occurs except if the TAMPxNOER bit is set in the RTC_TAMP register. The backup registers and the device secrets erased by tamp_erase signal can be reset by software by setting the BKERASE bit in the RTC_TAMP register

Timestamp on tamper event

The TPTS bit can control whether the tamper detection function is used as time-stamp function. If the bit is set to 1, the TSF bit will be set when the tamper event detected as if “enable” the time-stamp function. Whatever the TPTS bit is, the TPxF will assert when tamper event detected.

Edge detection mode on tamper input detection

When FLT bit is set to 0x0, the tamper detection is set to edge detection mode and TPxEG 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 (RTC_BKPx), 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: Tamper0 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.

14.3.15. Calibration clock output

Calibration clock can be output on the PC13 / PA3 / PA8 if COEN bit is set to 1.

When the COS bit is set to 0(this is default) and asynchronous prescaler is set to 0x7F(FACTOR_A), the frequency of RTC_CALIB is f_{rtcclk} / 64.When the RTCCLK is 32.768KHz,

RTC_CALIB output is corresponding to 512Hz.It's recommend to using rising edge of RTC_CALIB output for there may be a light jitter on falling edge.

When the COS bit is set to 1, the RTC_CALIB frequency is:

$$f_{rtc_calib} = \frac{f_{rtcclk}}{(FACTOR_A+1)\times(FACTOR_S+1)} \quad (14\text{-}5)$$

When the RTCCLK is 32.768 KHz, RTC_CALIB output is corresponding to 1Hz if prescaler are default values.

14.3.16. Alarm output

When OS control bits are not reset, RTC_ALARM alternate function output is enabled. This function will directly output the content of alarm flag or auto wakeup flag bit in RTC_STAT.

The OPOL bit in RTC_CTL can configure the polarity of the alarm or auto wakeup flag output which means that the RTC_ALARM output is the opposite of the corresponding flag bit or not.

14.3.17. RTC power saving mode management

Table 14-1. RTC power saving mode management

Mode	Active in Mode	Exit Mode
Sleep	Yes	RTC interrupts
Deep-Sleep	Yes: if clock source is LXTAL or IRC32K	RTC alarm / tamper event / timestamp event / wake up
Standby	Yes: if clock source is LXTAL or IRC32K	RTC alarm / tamper event / timestamp event / wake up

14.3.18. RTC interrupts

All RTC interrupts are connected to the EXTI controller.

Below steps should be followed if you want to use the RTC alarm / tamper / timestamp / auto wakeup interrupt:

1) Configure and enable the corresponding interrupt line to RTC alarm / tamper / ti-mestamp / auto wakeup event of EXTI and set the rising edge for triggering.
2) Configure and enable the RTC alarm / tamper / timestamp / auto wakeup interrupt.
3) Configure and enable the RTC alarm / tamper / timestamp / auto wakeup function.

Table 14-2. RTC interrupts control

Interrupt	Event flag	Control bit	Clear interrupt flag	Exit sleep	Exit deep-sleep and standby

Alarm 0	ALRM0F	ALRM0IE	write 1 in ALRM0FC	Y	Y(*)
Alarm 1	ALRM1F	ALRM1IE	write 1 in ALRM1FC	Y	Y(*)
Wakeup	WTF	WTIE	write 1 in WTFC	Y	Y(*)
Timestamp	TSF	TSIE	write 1 in TSFC	Y	Y(*)
Tamper x	TPxF	TPxIE	Write 1 in TPxFC	Y	Y(*)

Note: (*)Only active when RTC clock source is LXTAL or IRC32K.

14.4. Register definition

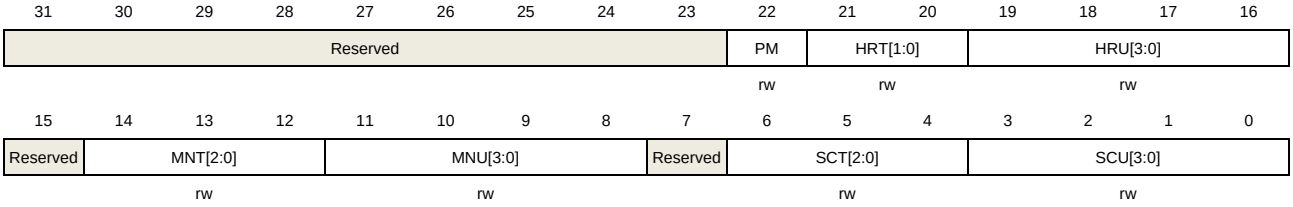
RTC base address: 0x4000 2800

14.4.1. Time register (RTC_TIME)

Address offset: 0x00
System reset value: 0x0000 0000 when BPSHAD = 0.
Not affected when BPSHAD = 1.

This register is write protected and can only be written in initialization state.

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



Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22	PM	AM / PM mark 0: AM or 24-hour format 1: PM
21:20	HRT[1:0]	Hour tens in BCD code
19:16	HRU[3:0]	Hour units in BCD code
15:	Reserved	Must be kept at reset value.
14:12	MNT[2:0]	Minute tens in BCD code
11:8	MNU[3:0]	Minute units in BCD code
7	Reserved	Must be kept at reset value.
6:4	SCT[2:0]	Second tens in BCD code
3:0	SCU[3:0]	Second units in BCD code

14.4.2. Date register (RTC_DATE)

Address offset: 0x04
System reset value: 0x0000 2101 when BPSHAD = 0.
Not affected when BPSHAD = 1.

This register is write protected and can only be written in initialization state.

This register 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								YRT[3:0]				YRU[3:0]			
								rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DOW[2:0]			MONT	MONU[2:0]				Reserved		DAYT		DAYU			
rw			rw	rw						rw		rw			

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:20	YRT	Year tens in BCD code
19:16	YRU[3:0]	Year units in BCD code
15:13	DOW[2:0]	Days of the week 0x0: Reserved 0x1: Monday … 0x7: Sunday
12	MONT	Month tens in BCD code
11:8	MONU[2:0]	Month units in BCD code
7:6	Reserved	Must be kept at reset value.
5:4	DAYT[1:0]	Day tens in BCD code
3:0	DAYU[3:0]	Day units in BCD code

14.4.3. Control register (RTC_CTL)

Address offset: 0x08

System reset: not affected

Backup domain reset value: 0x0000 0000

This register is writing 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
OUT2EN	Reserved							COEN	OS[1:0]		OPOL	COS	DSM	S1H	A1H
rw								rw	rw		rw	rw	rw	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSIE	WTIE	ALRM1IE	ALRM0IE	TSEN	WTEN	ALRM1EN	ALRM0EN	CCEN	CS	BPSHAD	REFEN	TSEG	WTCS[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		

Bits	Fields	Descriptions
31	OUT2EN	RTC_OUT pin select 0: RTC_OUT output to PC13

		1: RTC_OUT output to PA3 or PA8
30:24	Reserved	Must be kept at reset value.
23	COEN	Calibration output enable 0: Disable calibration output 1: Enable calibration output
22:21	OS[1:0]	Output selection This bit is used for selecting flag source to output 0x0: Disable output RTC_ALARM 0x1: Enable alarm0 flag output 0x2: Enable alarm1 flag output 0x3: Enable wakeup flag output
20	OPOL	Output polarity This bit is used to invert output RTC_ALARM 0: Disable invert output RTC_ALARM 1: Enable invert output RTC_ALARM
19	COS	Calibration output selection Valid only when COEN=1 and prescalers are at default values 0: Calibration output is 512 Hz 1: Calibration output is 1Hz
18	DSM	Daylight saving mark This bit is flexible used by software. Often can be used to recording the daylight saving hour adjustment.
17	S1H	Subtract 1 hour(winter time change) One hour will be subtracted from current time if it is not 0 0: No effect 1: 1 hour will be subtracted at next second change time.
16	A1H	Add 1 hour(summer time change) One hour will be added from current time 0: No effect 1: 1 hour will be added at next second change time
15	TSIE	Time-stamp interrupt enable 0: Disable time-stamp interrupt 1: Enable time-stamp interrupt
14	WTIE	Auto-wakeup timer interrupt enable 0: Disable auto-wakeup timer interrupt 1: Enable auto-wakeup timer interrupt
13	ALRM1IE	RTC alarm-1 interrupt enable 0: Disable alarm interrupt

		1: Enable alarm interrupt
12	ALRM0IE	RTC alarm-0 interrupt enable 0: Disable alarm interrupt 1: Enable alarm interrupt
11	TSEN	Time-stamp function enable 0: Disable time-stamp function 1: Enable time-stamp function
10	WTEN	Auto-wakeup timer function enable 0: Disable function 1: Enable function
9	ALRM1EN	Alarm-1 function enable 0: Disable alarm function 1: Enable alarm function
8	ALRM0EN	Alarm-0 function enable 0: Disable alarm function 1: Enable alarm function
7	CCEN	Coarse calibration function enable 0: Disable function 1: Enable function **Note:** FACTOR_A must be greater than 6 before enabled and can only be written in initialization state.
6	CS	Clock System 0: 24-hour format 1: 12-hour format **Note:** Can only be written in initialization state
5	BPSHAD	Shadow registers bypass control 0: Reading calendar from shadow registers 1: Reading calendar from current real-time calendar **Note:** If frequency of APB1 clock is less than seven times the frequency of RTCCLK, this bit must set to 1.
4	REFEN	Reference clock detection function enable 0: Disable reference clock detection function 1: Enable reference clock detection function **Note:** Can only be written in initialization state and FACTOR_S must be 0x00FF
3	TSEG	Valid event edge of time-stamp 0: rising edge is valid event edge for time-stamp event 1: falling edge is valid event edge for time-stamp event
2:0	WTCS[2:0]	Auto-wakeup timer clock selection

0x0:RTC Clock divided by 16
0x1:RTC Clock divided by 8
0x2:RTC Clock divided by 4
0x3:RTC Clock divided by 2
0x4:0x5: ck_spre (default 1Hz) clock
0x6:0x7: ck_spre (default 1Hz) clock and 2^{16} is added to wake-up counter.

14.4.4. Status register (RTC_STAT)

Address offset: 0x0C
System reset: Only INITM, INITF and RSYNF bits are set to 0. Others are not affected
Backup domain reset value: 0x0000 0007

This register is writing 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
Reserved															SCPF
															r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TP1F	TP0F	TSOVRF	TSF	WTF	ALRM1F	ALRM0F	INITM	INITF	RSYNF	YCM	SOPF	WTWF	ALRM1WF	ALRM0WF
	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rw	r	rc_w0	r	r	r	r	r

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	SCPF	Smooth calibration pending flag Set to 1 by hardware when software writes to RTC_HRFC and set to 0 by hardware when smooth calibration window is completed.
15	Reserved	Must be kept at reset value.
14	TP1F	RTC_TAMP1 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	TP0F	RTC_TAMP0 detected flag Set to 1 by hardware when tamper detection is found on tamper0 input pin. Software can clear this bit by writing 0 into this bit.
12	TSOVRF	Time-stamp overflow flag This bit is set by hardware when a time-stamp event is detected if TSF bit is set before. Cleared by software writing 0.
11	TSF	Time-stamp flag Set by hardware when time-stamp event is detected. Cleared by software writing 0.

10	WTF	Wakeup timer flag Set by hardware when wakeup timer decreased to 0. Cleared by software writing 0. This flag must be cleared at least 1.5 RTC Clock periods before WTF is set to 1 again.
9	ALRM1F	Alarm-1 occurs flag Set to 1 by hardware when current time/date matches the time/date of alarm 1 setting value. Cleared by software writing 0.
8	ALRM0F	Alarm-0 occurs flag Set to 1 by hardware when current time/date matches the time/date of alarm 0 setting value. Cleared by software writing 0.
7	INITM	Enter initialization mode 0: Free running mode 1: Enter initialization mode for setting calendar time / date and prescaler. Counter will stop under this mode.
6	INITF	Initialization state flag Set to 1 by hardware and calendar register and prescaler can be programmed in this state. 0: Calendar registers and prescaler register cannot be changed 1: Calendar registers and prescaler register can be changed
5	RSYNF	Register synchronization flag Set to 1 by hardware every 2 RTCCLK which will copy current calendar time / date into shadow register. Initialization mode (INITM), shift operation pending flag (SOPF) or bypass mode (BPSHAD) will clear this bit. This bit is also can be cleared by software writing 0. 0: Shadow register are not yet synchronized 1: Shadow register are synchronized
4	YCM	Year configuration mark Set by hardware if the year field of calendar date register is not the default value 0. 0: Calendar has not been initialized 1: Calendar has been initialized
3	SOPF	Shift function operation pending flag 0: No shift operation is pending 1: Shift function operation is pending
2	WTWF	Wakeup timer write enable flag 0: Wakeup timer update is not allowed 1: Wakeup timer update is allowed

1	ALRM1WF	Alarm 1 configuration can be write flag Set by hardware if alarm register can be wrote after ALRM1EN bit has reset. 0: Alarm registers programming is not allowed 1: Alarm registers programming is allowed
0	ALRM0WF	Alarm 0 configuration can be write flag Set by hardware if alarm register can be wrote after ALRM0EN bit has reset. 0: Alarm registers programming is not allowed. 1: Alarm registers programming is allowed.

14.4.5. Prescaler register (RTC_PSC)

Address offset: 0x10

System reset: not effected

Backup domain reset value: 0x007F 00FF

This register is write protected and can only be written in initialization state.

This register 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									FACTOR_A[6:0]						
									rw						

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

Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22:16	FACTOR_A[6:0]	Asynchronous prescaler factor ck_apre frequency = RTCCLK frequency / (FACTOR_A+1)
15	Reserved	Must be kept at reset value.
14:0	FACTOR_S[14:0]	Synchronous prescaler factor ck_spre frequency = ck_apre frequency / (FACTOR_S+1)

14.4.6. Wakeup timer register (RTC_WUT)

Address offset: 0x14

System reset: not effected

Backup domain reset value: 0x0000 FFFF

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

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

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

rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	WTRV[15:0]	Auto-wakeup timer reloads value. Every (WTRV[15:0]+1) ck_wut period the WTF bit is set after WTEN=1.The ck_wut is selected by WTCS[2:0] bits. **Note:** This configure case is forbidden: WTRV=0x0000 with WTCS[2:0]=0b011. This register can be written only when WTWF=1.

14.4.7. Coarse calibration register (RTC_COSC)

Address offset: 0x18

System reset: not effect

Backup domain reset value: 0x0000 0000

This register is write protected and can only be written in initialization state.

This register 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								COSD	Reserved		COSS[4:0]				
								rw			rw				

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	COSD	Coarse calibration direction 0: Increase calendar update frequency 1: Decrease calendar update frequency
6:5	Reserved	Must be kept at reset value.
4:0	COSS[4:0]	Coarse calibration step When COSD=0: 0x00:+0 PPM 0x01:+4 PPM(approximate value) 0x02:+8 PPM(approximate value) .... 0x1F:+126 PPM(approximate value) When COSD=1: 0x00:-0 PPM 0x01:-2 PPM(approximate value) 0x02:-4 PPM(approximate value) …

0x1F:-63 PPM(approximate value)

14.4.8. Alarm 0 time and date register (RTC_ALRM0TD)

Address offset: 0x1C

System reset: not effect

Backup domain reset value: 0x0000 0000

This registe is write protected and can only be written in initialization state.

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSKD	DOWS	DAYT[1:0]		DAYU[3:0]				MSKH	PM	HRT[1:0]		HRU[3:0]			
rw	rw	rw		rw				rw	rw	rw		rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSKM	MNT[2:0]			MNU[3:0]				MSKS	SCT[2:0]			SCU[3:0]			
rw	rw			rw				rw	rw			rw			

Bits	Fields	Descriptions
31	MSKD	Alarm date mask bit 0: Not mask date / day field 1: Mask date / day field
30	DOWS	Day of the week selected 0: DAYU[3:0] indicates the date units 1: DAYU[3:0] indicates the week day and DAYT[1:0] has no means.
29:28	DAYT[1:0]	Date tens in BCD code
27:24	DAYU[3:0]	Date units or week day in BCD code
23	MSKH	Alarm hour mask bit 0: Not mask hour field 1: Mask hour field
22	PM	AM / PM flag 0: AM or 24-hour format 1: PM
21:20	HRT[1:0]	Hour tens in BCD code
19:16	HRU[3:0]	Hour units in BCD code
15	MSKM	Alarm minutes mask bit 0: Not mask minutes field 1: Mask minutes field
14:12	MNT[2:0]	Minutes tens in BCD code
11:8	MNU[3:0]	Minutes units in BCD code

7	MSKS	Alarm second mask bit 0: Not mask second field 1: Mask second field
6:4	SCT[2:0]	Second tens in BCD code
3:0	SCU[3:0]	Second units in BCD code

14.4.9. Alarm 1 time and date register (RTC_ALRM1TD)

Address offset: 0x20

System reset: not effect

Backup domain reset value: 0x0000 0000

This register is write protected and can only be written in initialization state.

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSKD	DOWS	DAYT[1:0]		DAYU[3:0]				MSKH	PM	HRT[1:0]		HRU[3:0]			
rw	rw	rw		rw				rw	rw	rw		rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSKM	MNT[2:0]			MNU[3:0]				MSKS	SCT[2:0]			SCU[3:0]			
rw	rw			rw				rw	rw			rw			

Bits	Fields	Descriptions
31	MSKD	Alarm date mask bit 0: Not mask date / day field 1: Mask date / day field
30	DOWS	Day of the week selected 0: DAYU[3:0] indicates the date units 1: DAYU[3:0] indicates the week day and DAYT[3:0] has no means.
29:28	DAYT[1:0]	Day tens in BCD code
27:24	DAYU[3:0]	Day units or week day in BCD code
23	MSKH	Alarm hour mask bit 0: Not mask hour field 1: Mask hour field
22	PM	AM / PM flag 0: AM or 24-hour format 1: PM
21:20	HRT[1:0]	Hour tens in BCD code
19:16	HRU[3:0]	Hour units in BCD code
15	MSKM	Alarm minutes mask bit 0: Not mask minutes field

		1: Mask minutes field
14:12	MNT[2:0]	Minutes tens in BCD code
11:8	MNU[3:0]	Minutes units in BCD code
7	MSKS	Alarm second mask bit 0: Not mask second field 1: Mask second field
6:4	SCT[2:0]	Second tens in BCD code
3:0	SCU[3:0]	Second units in BCD code

14.4.10. Write protection key register (RTC_WPK)

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								WPK[7:0]							
								w							

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	WPK[7:0]	Key for write protection

14.4.11. Sub second register (RTC_SS)

Address offset: 0x28

System reset value: 0x0000 0000 when BPSHAD = 0.

Not affected when BPSHAD = 1.

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

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	SSC[15:0]	Sub second value

This value is the counter value of synchronous prescaler. Second fraction value is calculated by the below formula:
Second fraction = (FACTOR_S - SSC) / (FACTOR_S + 1)

14.4.12. Shift function control register (RTC_SHIFTCTL)

Address offset: 0x2C
System reset: not effect
Backup Reset value: 0x0000 0000

This register is writing protected and can only be wrote when SOPF=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
A1S	Reserved														
w															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SFS[14:0]														
	w														

Bits	Fields	Descriptions
31	A1S	One second add 0: Not add 1 second 1: Add 1 second to the clock / calendar. This bit is jointly used with SFS field to add a fraction of a second to the clock.
30:15	Reserved	Must be kept at reset value.
14:0	SFS[14:0]	Subtract a fraction of a second The value of this bit will add to the counter of synchronous prescaler. When only using SFS, the clock will delay because the synchronous prescaler is a down counter: Delay (seconds) = SFS / (FACTOR_S + 1) When jointly using A1S and SFS, the clock will advance: Advance (seconds) = (1 - (SFS / (FACTOR_S + 1)))

Note: Writing to this register will cause RSYNF bit to be cleared.

14.4.13. Time of time stamp register (RTC_TTS)

Address offset: 0x30
Backup domain reset value: 0x0000 0000
System reset: no effect

This register will record the calendar time when TSF is set to 1.Reset TSF bit will also clear this register.

This register 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	PM	HRT[1:0]	HRU[3:0]
	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	MNT[2:0]			MNU[3:0]				Reserved	SCT[2:0]			SCU[3:0]			
	r			r					r			r			

Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22	PM	AM / PM mark 0:AM or 24-hour format 1:PM
21:20	HRT[1:0]	Hour tens in BCD code
19:16	HRU[3:0]	Hour units in BCD code
15	Reserved	Must be kept at reset value.
14:12	MNT[2:0]	Minute tens in BCD code
11:8	MNU[3:0]	Minute units in BCD code
7	Reserved	Must be kept at reset value.
6:4	SCT[2:0]	Second tens in BCD code
3:0	SCU[3:0]	Second units in BCD code

14.4.14. Date of time stamp register (RTC_DTS)

Address offset: 0x34
Backup domain reset value: 0x0000 0000
System reset: no effect

This register will record the calendar date when TSF is set to 1.Reset TSF bit will also clear this register.

This register 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
DOW[2:0]			MONT	MONU[3:0]				Reserved		DAYT[1:0]		DAYU[3:0]			
r			r	r						r		r			

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:13	DOW[2:0]	Days of the week

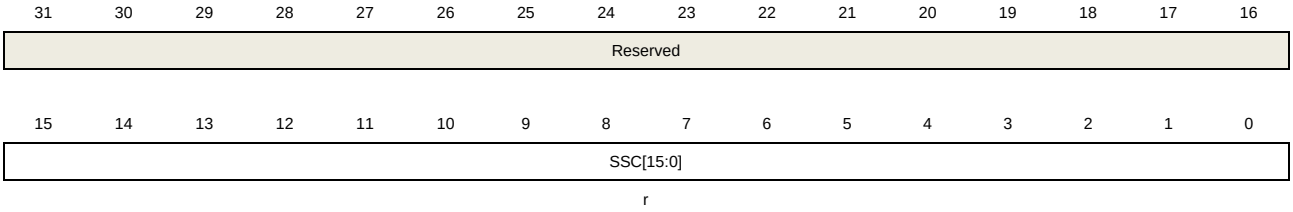
12	MONT	Month tens in BCD code
11:8	MONU[3:0]	Month units in BCD code
7:6	Reserved	Must be kept at reset value.
5:4	DAYT[1:0]	Day tens in BCD code
3:0	DAYU[3:0]	Day units in BCD code

14.4.15. Sub second of time stamp register (RTC_SSTS)

Address offset: 0x38
Backup domain reset: 0x0000 0000
System reset: no effect

This register will record the calendar date when TSF is set to 1.Reset TSF bit will also clear this register.

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



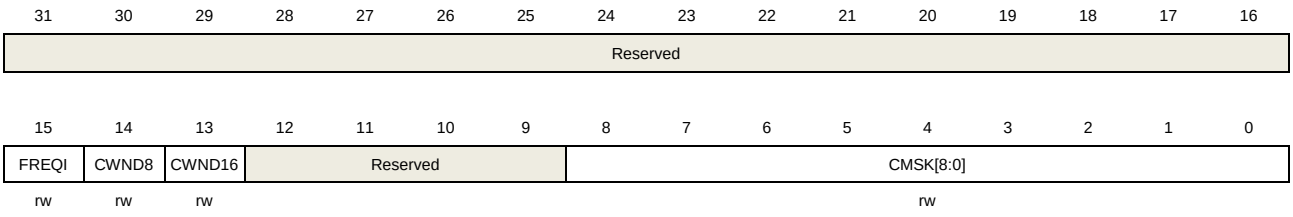
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	SSC[15:0]	Sub second value This value is the counter value of synchronous prescaler when TSF is set to 1.

14.4.16. High resolution frequency compensation register (RTC_HRFC)

Address offset: 0x3C
Backup domain reset: 0x0000 0000
System Reset: no effect

This register is write protected.

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



Bits	Fields	Descriptions

31:16	Reserved	Must be kept at reset value.
15	FREQI	Increase RTC frequency by 488.5PPM 0: No effect 1: One RTCCLK pulse is inserted every 2^{11} pulses. This bit should be used in conjunction with CMSK bit. If the input clock frequency is 32.768KHz, the number of RTCCLK pulses added during 32s calibration window is (512 * FREQI) - CMSK
14	CWND8	Frequency compensation window 8 second selected 0: No effect 1: Calibration window is 8 second **Note:** When CWND8=1, CMSK[1:0] are stuck at “00”.
13	CWND16	Frequency compensation window 16 second selected 0: No effect 1: Calibration window is 16 second **Note:** When CWND16=1, CMSK[0] are stuck at “0”.
12:9	Reserved	Must be kept at reset value.
8:0	CMSK[8:0]	Calibration mask number The number of mask pulse out of 2^{20} RTCCLK pulse. This feature will decrease the frequency of calendar with a resolution of 0.9537 PPM.

14.4.17. Tamper register (RTC_TAMP)

Address offset: 0x40

Backup domain reset: 0x0000 0000

System reset: no effect

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BKERASE	Reserved										TP1NOER	TP0NOER	AOT	Reserved	
rw											rw	rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DISPU	PRCH[1:0]		FLT[1:0]		FREQ[2:0]			TPTS	Reserved		TP1EG	TP1EN	TPIE	TP0EG	TP0EN
rw	rw		rw		rw			rw			rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31	BKERASE	Backup registers erase Writing ‘1’ to this bit reset the backup registers. Writing 0 has no effect. This bit is always read as 0.
30:21	Reserved	Must be kept at reset value.
20	TP1NOER	Tamper 1 no erase 0: Tamper 1 event erases the backup registers.

		1: Tamper 1 event does not erase the backup registers
19	TP0NOER	Tamper 0 no erase 0: Tamper 0 event erases the backup registers. 1: Tamper 0 event does not erase the backup registers
18	AOT	RTC_ALARM output type 0: Open-drain output type 1: Push-pull output type
17:16	Reserved	Must be kept at reset value.
15	DISPU	RTC_TAMPx pull up disable bit 0: Enable inner pull-up before sampling for pre-charge RTC_TAMPx pin 1: Disable pre-charge duration
14:13	PRCH[1:0]	Pre-charge duration time of RTC_TAMPx This setting determines the pre-charge time before each sampling. 0x0: 1 RTC clock 0x1: 2 RTC clock 0x2: 4 RTC clock 0x3: 8 RTC clock
12:11	FLT[1:0]	RTC_TAMPx 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 RTCCLK(1Hz if RTCCLK=32.768KHz) 0x1: Sample once every 16384 RTCCLK(2Hz if RTCCLK=32.768KHz) 0x2: Sample once every 8192 RTCCLK(4Hz if RTCCLK=32.768KHz) 0x3: Sample once every 4096 RTCCLK(8Hz if RTCCLK=32.768KHz) 0x4: Sample once every 2048 RTCCLK(16Hz if RTCCLK=32.768KHz) 0x5: Sample once every 1024 RTCCLK(32Hz if RTCCLK=32.768KHz) 0x6: Sample once every 512 RTCCLK(64Hz if RTCCLK=32.768KHz) 0x7: Sample once every 256 RTCCLK(128Hz if RTCCLK=32.768KHz)
7	TPTS	Make tamper function used for timestamp function 0:No effect

		1:TSF is set when tamper event detected even TSEN=0
6:5	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	TP0EG	Tamper 0 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
0	TP0EN	Tamper 0 detection enable 0: Disable tamper 0 detection function 1: Enable tamper 0 detection function

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

14.4.18. Alarm 0 sub second register (RTC_ALRM0SS)

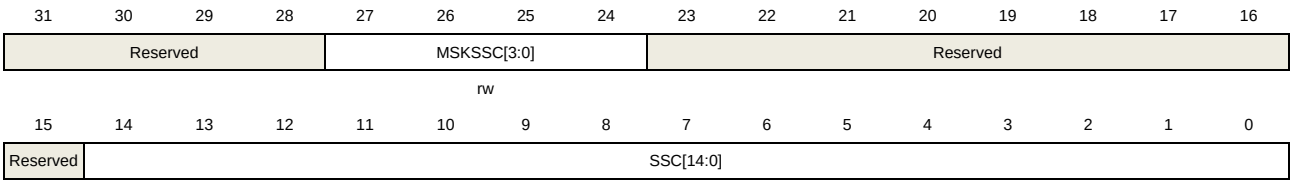
Address offset: 0x44

Backup domain reset: 0x0000 0000

System reset: no effect

This register is write protected and can only be wrote when ALRM0EN=0 or INITM=1.

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



rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:24	MSKSSC[3:0]	Mask control bit of SSC 0x0: Mask alarm sub second setting. The alarm asserts at every second time point if all the rest alarm fields are matched. 0x1: SSC[0] is to be compared and all others are ignored 0x2: SSC[1:0] is to be compared and all others are ignored 0x3: SSC[2:0] is to be compared and all others are ignored 0x4: SSC[3:0] is to be compared and all others are ignored 0x5: SSC[4:0] is to be compared and all others are ignored 0x6: SSC[5:0] is to be compared and all others are ignored 0x7: SSC[6:0] is to be compared and all others are ignored 0x8: SSC[7:0] is to be compared and all others are ignored 0x9: SSC[8:0] is to be compared and all others are ignored 0xA: SSC[9:0] is to be compared and all others are ignored 0xB: SSC[10:0] is to be compared and all others are ignored 0xC: SSC[11:0] is to be compared and all others are ignored 0xD: SSC[12:0] is to be compared and all others are ignored 0xE: SSC[13:0] is to be compared and all others are ignored 0xF: SSC[14:0] is to be compared and all others are ignored **Note:** The bit 15 of synchronous counter (SSC[15] in RTC_SS) is never compared.
23:15	Reserved	Must be kept at reset value.
14:0	SSC[14:0]	Alarm sub second value This value is the alarm sub second value which is to be compared with synchronous prescaler counter SSC. Bit number is controlled by MSKSSC bits.

14.4.19. Alarm 1 sub second register (RTC_ALRM1SS)

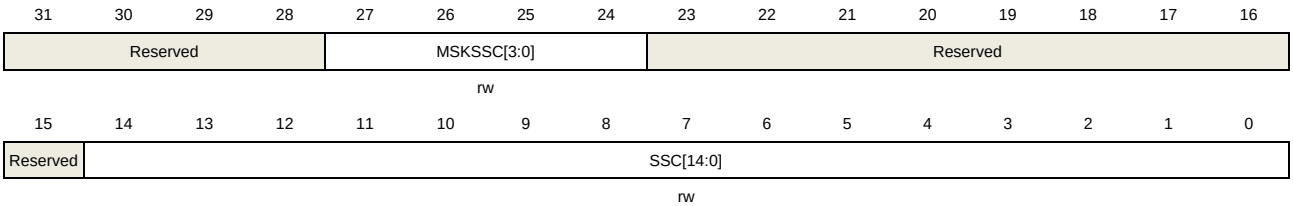
Address offset: 0x48

Backup domain reset: 0x0000 0000

System reset: no effect

This register is write protected and can only be wrote when ALRM1EN=0 or INITM=1.

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



Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:24	MSKSSC[3:0]	Mask control bit of SSC 0x0: Mask alarm sub second setting. The alarm asserts at every second time point if all the rest alarm fields are matched. 0x1: SSC[0] is to be compared and all others are ignored 0x2: SSC[1:0] is to be compared and all others are ignored 0x3: SSC[2:0] is to be compared and all others are ignored 0x4: SSC[3:0] is to be compared and all others are ignored 0x5: SSC[4:0] is to be compared and all others are ignored 0x6: SSC[5:0] is to be compared and all others are ignored 0x7: SSC[6:0] is to be compared and all others are ignored 0x8: SSC[7:0] is to be compared and all others are ignored 0x9: SSC[8:0] is to be compared and all others are ignored 0xA: SSC[9:0] is to be compared and all others are ignored 0xB: SSC[10:0] is to be compared and all others are ignored 0xC: SSC[11:0] is to be compared and all others are ignored 0xD: SSC[12:0] is to be compared and all others are ignored 0xE: SSC[13:0] is to be compared and all others are ignored 0xF: SSC[14:0] is to be compared and all others are ignored **Note:** The bit 15 of synchronous counter (SSC[15] in RTC_SS) is never compared.
23:15	Reserved	Must be kept at reset value.
14:0	SSC[14:0]	Alarm sub second value This value is the alarm sub second value which is to be compared with synchronous prescaler counter SSC. Bit number is controlled by MSKSSC bits.

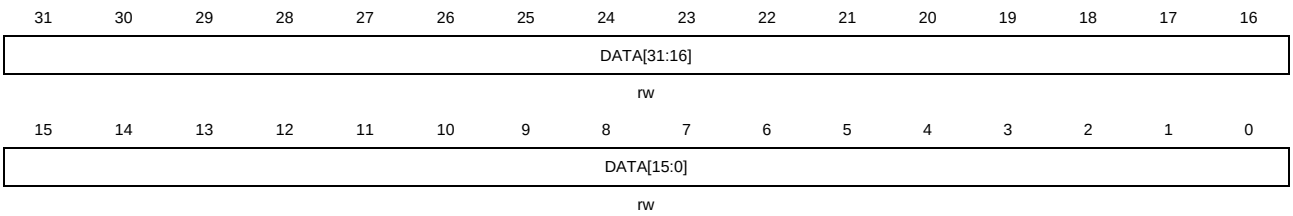
14.4.20. Backup registers (RTC_BKPx) (x = 0…19)

Address offset: 0x70 + 0x04 * x

Backup domain reset: 0x0000 0000

System reset: no effect

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



Bits	Fields	Descriptions
31:0	DATA[31:0]	Data These registers can be wrote or read by software. Tamper detection flag TPxF

assertion will reset these registers.

15. Timer (TIMERx)

Table 15-1. Timers (TIMERx) are divided into four sorts

TIMER	TIMER0	TIMER1/2	TIMER15/16	TIMER5
TYPE	Advanced	General-L0	General-L4	Basic
Prescaler	16-bit	16-bit	16-bit	16-bit
Counter	16-bit	32-bit(TIMER1/2)	16-bit	16-bit
Count mode	UP,DOWN, Center-aligned	UP,DOWN, Center-aligned	UP ONLY	UP ONLY
Repetition	●	×	●	×
CH Capture/ Compare	4	4	1	0
Complementary & Dead-time	●	×	●	×
Break	●	×	●	×
Single Pulse	●	●	●	●
Quadrature Decoder	●	●	×	×
Master-slave management	●	●	×	×
Inter connection	●(1)	●(2)	×	×
DMA	●	●	●	●(3)
Debug Mode	●	●	●	●

(1) TIMER0 **ITI0:** 0 **ITI1:** TIMER1_TRGO **ITI2:** TIMER2_TRGO **ITI3:** 0

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

TIMER2 **ITI0:** TIMER0_TRGO **ITI1:** TIMER1_TRGO **ITI2:** 0 **ITI3:** 0

(3) Only update events will generate a DMA request. Note that TIMER5 do not have DMA configuration registers.

15.1. Advanced timer (TIMERx, x=0)

15.1.1. Overview

The advanced timer module (Timer0) is a four-channel timer that supports both input capture and output compare. They can generate PWM signals to control motor or be used for power management applications. The advanced timer has a 16-bit counter that can be used as an unsigned counter.

In addition, the advanced timers can be programmed and be used for counting, their external events can be used to drive other timers.

Timer also includes a dead-time Insertion module which is suitable for motor control applications.

Timer and timer are completely independent with each other, but they may be synchronized to provide a larger timer with their counters incrementing in unison.

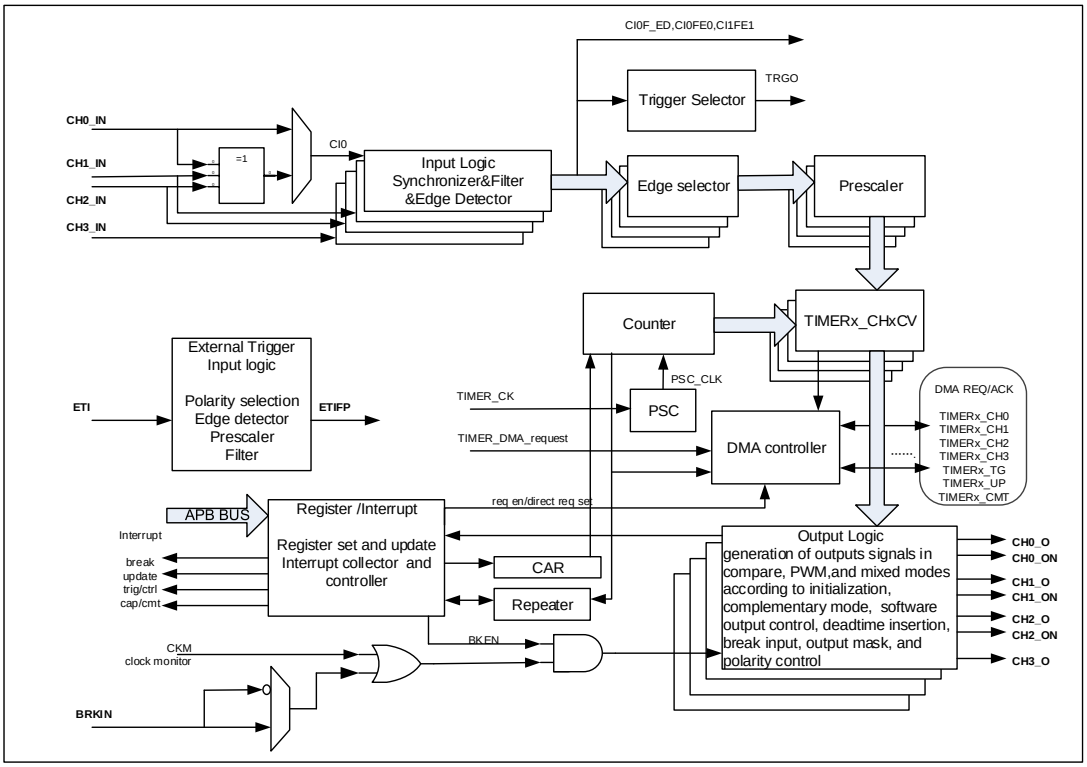
15.1.2. Characteristics

- Total channel num: 4.
- Counter width: 16 bit.
- Source of counter clock is selectable:
  internal clock, internal trigger, external input, external trigger.
- Multiple counter modes: count up, count down, count up/down.
- Quadrature Decoder: used to track motion and determine both rotation direction and position.
- Hall sensor: for 3-phase motor control.
- Programmable prescaler: 16 bit. The factor can be changed on the go.
- 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, trigger event, compare/capture event, and break input.
- Daisy chaining of timer modules allows a single timer to initiate multiple timers.
- Timer synchronization allows selected timers to start counting on the same clock cycle.
- Timer master-slave management.

15.1.3. Block diagram

Figure 15-1. Advanced timer block diagram provides details of the internal configuration of the advanced timer.

Figure 15-1. Advanced timer block diagram



15.1.4. 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 TSCFGy[3:0] in SYSCFG_TIMER0CFG(y=0,1…6). When alternate clock source is used, the SYSCFG clock must be enabled.

- TSCFGy[3:0] = 4'b0000 in SYSCFG_TIMER0CFG(y=0,1…6). Internal clock CK_TIMER is selected as timer clock source which is from module RCU.

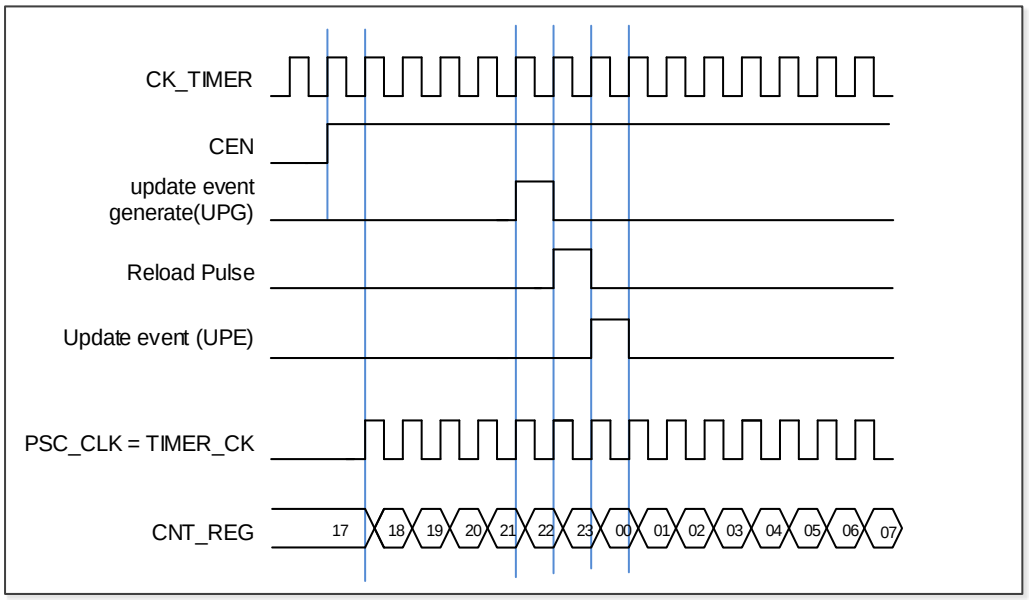
The default clock source is the CK_TIMER for driving the counter prescaler when TSCFGy[3:0] = 4'b0000 in SYSCFG_TIMER0CFG(y=0,1…6). When the CEN is set, the CK_TIMER will be divided by PSC value to generate PSC_CLK.

In this mode, the TIMER_CK which drives counter's prescaler to count is equal to CK_TIMER which is from RCU module.

- if TSCFGy[3:0] !=4'b0000 in SYSCFG_TIMER0CFG(y=0,1,2,6), the prescaler is clocked by other clock sources selected in the TSCFG6[3:0] register, more details will be

introduced later. When the TSCFGy[3:0] (y=3,4,5) are setting to an available value, the internal clock TIMER_CK is the counter prescaler driving clock source.

Figure 15-2. Normal mode, internal clock divided by 1



- TSCFG6[3:0] are setting to an available value (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 TSCFG6[3:0] to 0x5, 0x6, 0x7.

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 TSCFG6[3:0] to 0x1,0x2,0x3,0x4.

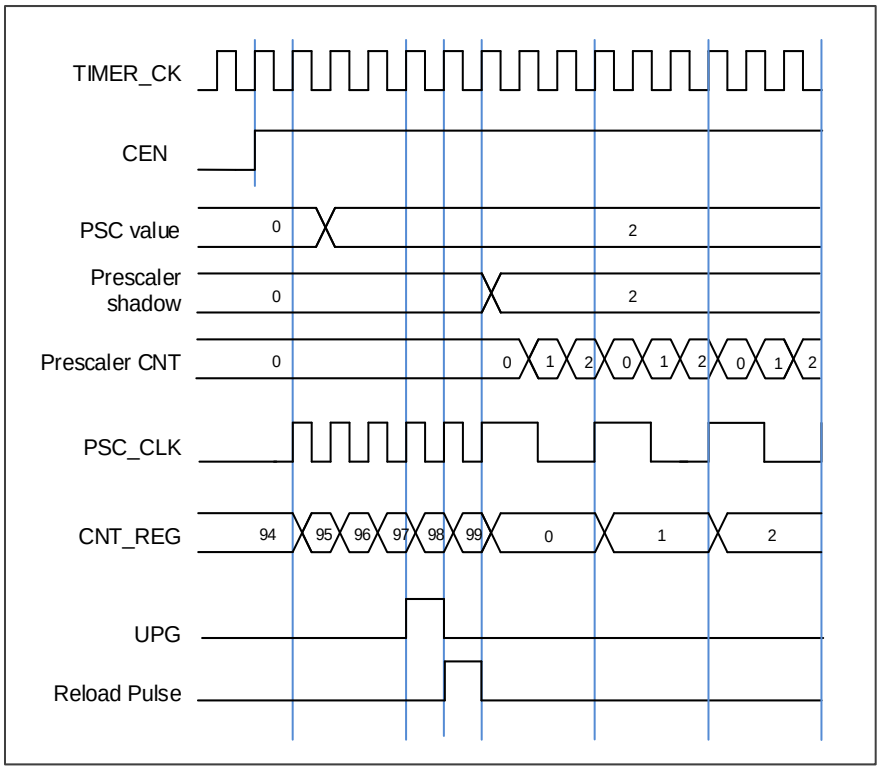
- SMC1= 1'b1 (external clock mode 1). External input ETI is selected as timer clock source.

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 TSCFG6[3:0] to 0x8. 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 15-3. 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 15-4. Timing chart of up counting mode, PSC=0/2 and ***Figure 15-5. Timing chart of up counting mode, change TIMERx_CAR ongoing*** show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 15-4. Timing chart of up counting mode, PSC=0/2

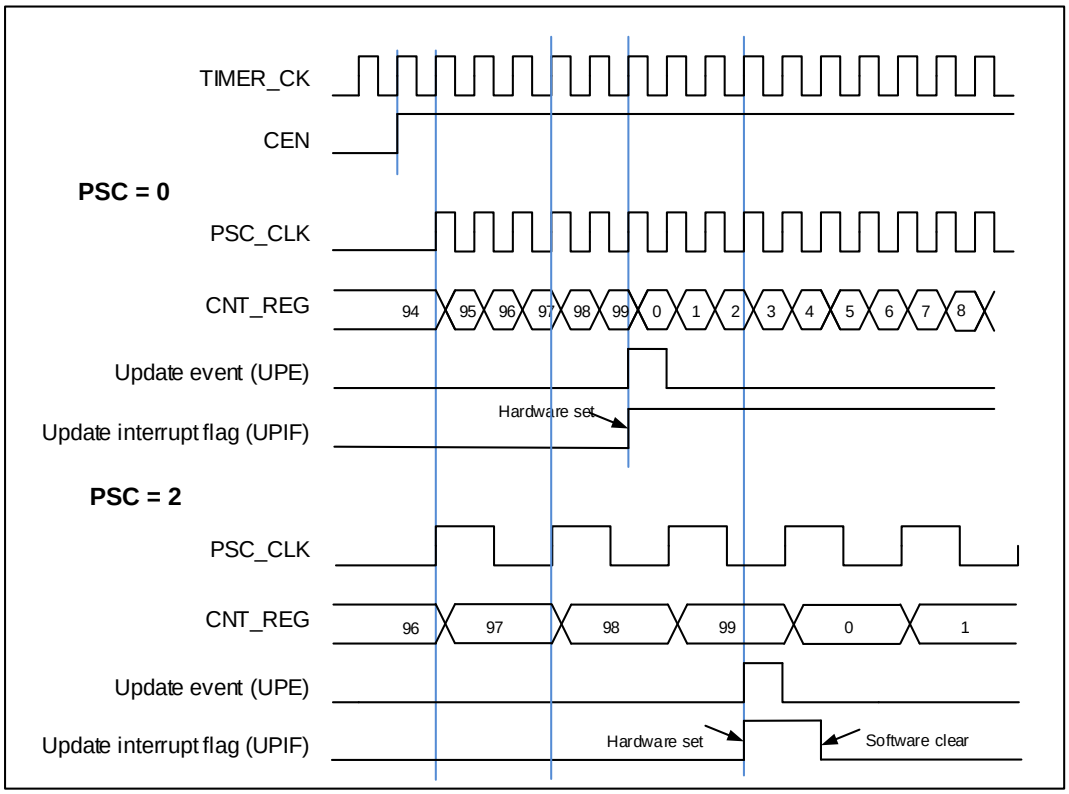
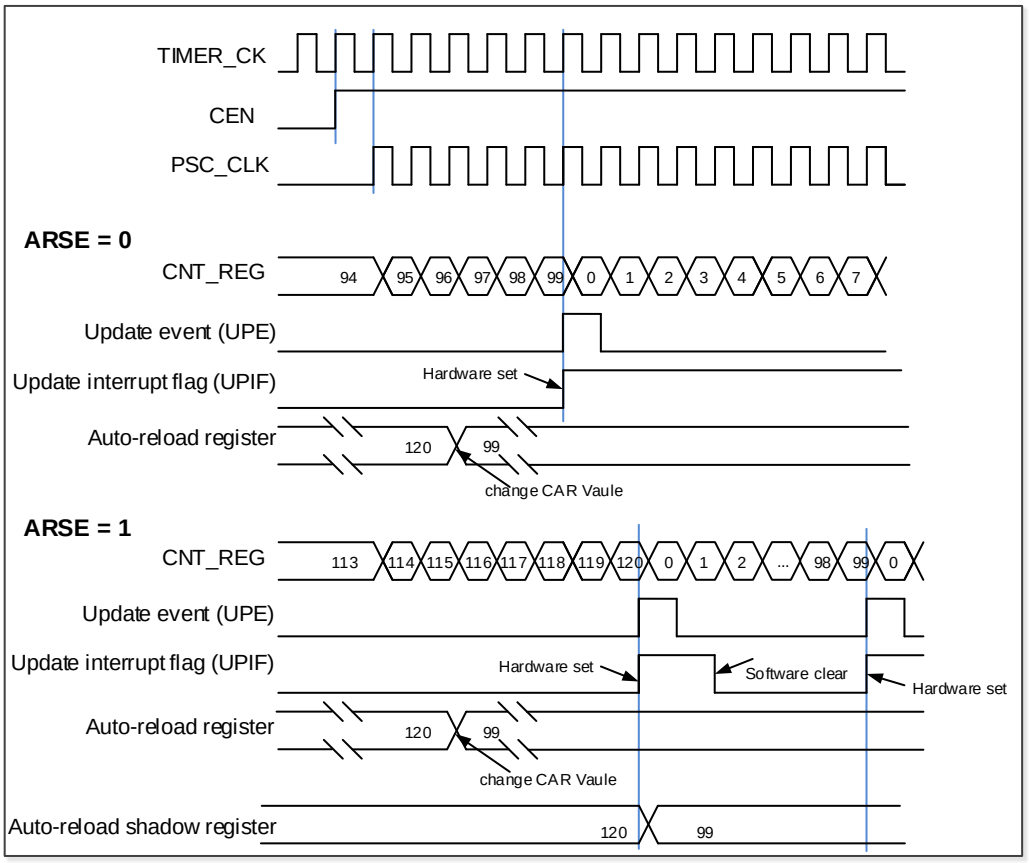


Figure 15-5. Timing chart of up counting mode, change TIMERx_CAR ongoing



Counter down counting

In this mode, the counter counts down continuously from the counter-reload value, which is defined in the TIMERx_CAR register, to 0 in a count-down direction. Once the counter reaches to 0, the counter 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 15-6. Timing chart of down counting mode, PSC=0/2 and *Figure 15-7. Timing chart of down counting mode, change TIMERx_CAR ongoing* show some examples of the counter behavior in different clock frequencies when TIMERx_CAR=0x99.

Figure 15-6. Timing chart of down counting mode, PSC=0/2

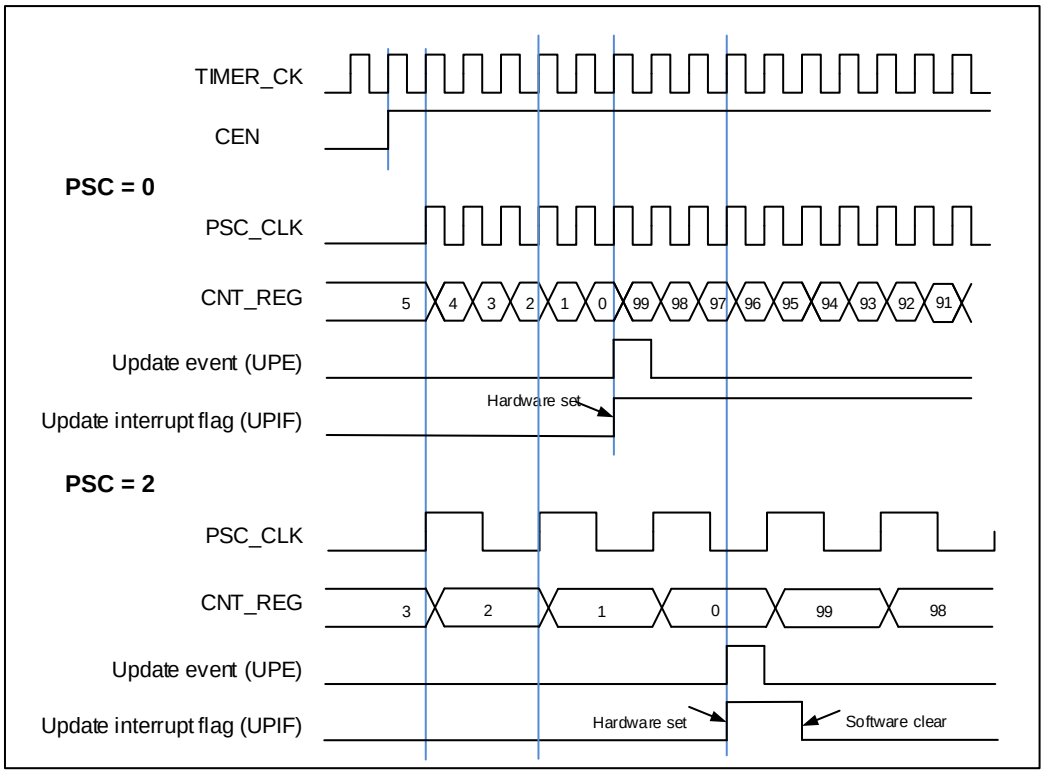
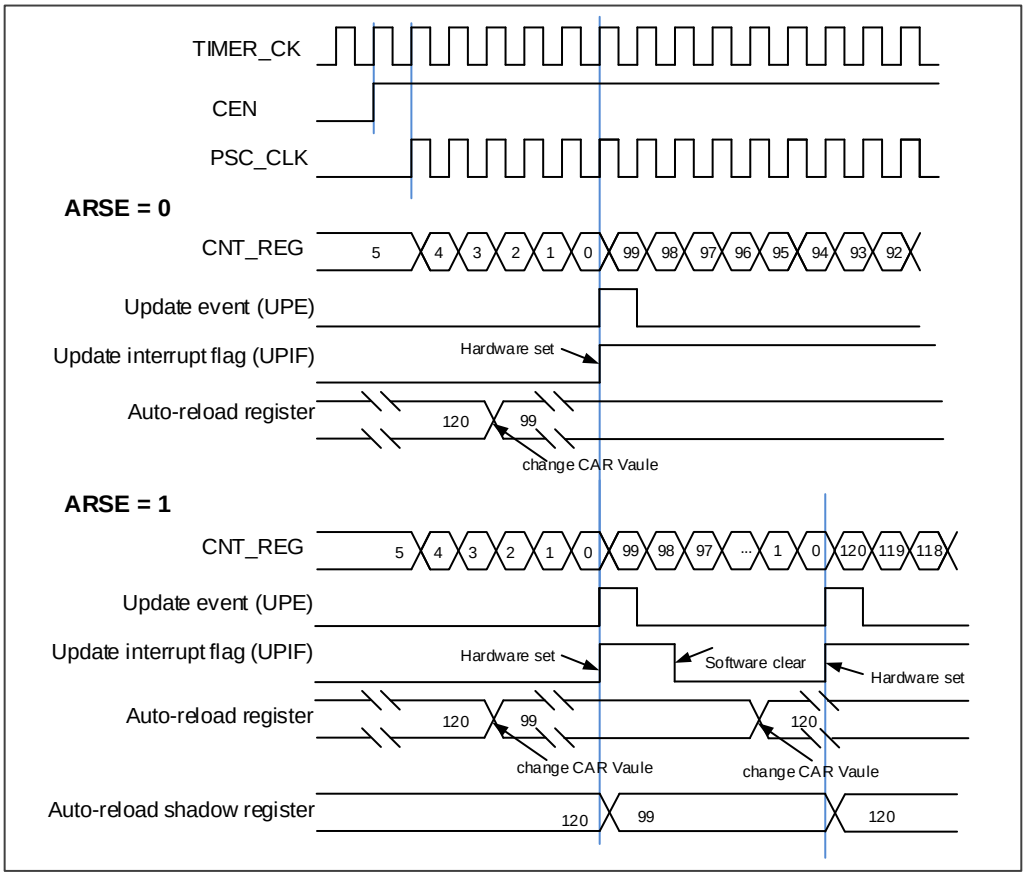


Figure 15-7. Timing chart of down counting mode, change TIMERx_CAR ongoing



Counter center-aligned counting

In this mode, the counter counts up from 0 to the counter-reload value and then counts down to 0 alternatively. The Timer module generates an overflow event when the counter counts to the counter-reload value subtract 1 in the up-counting direction and generates an underflow event when the counter counts to 1 in the down-counting 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.

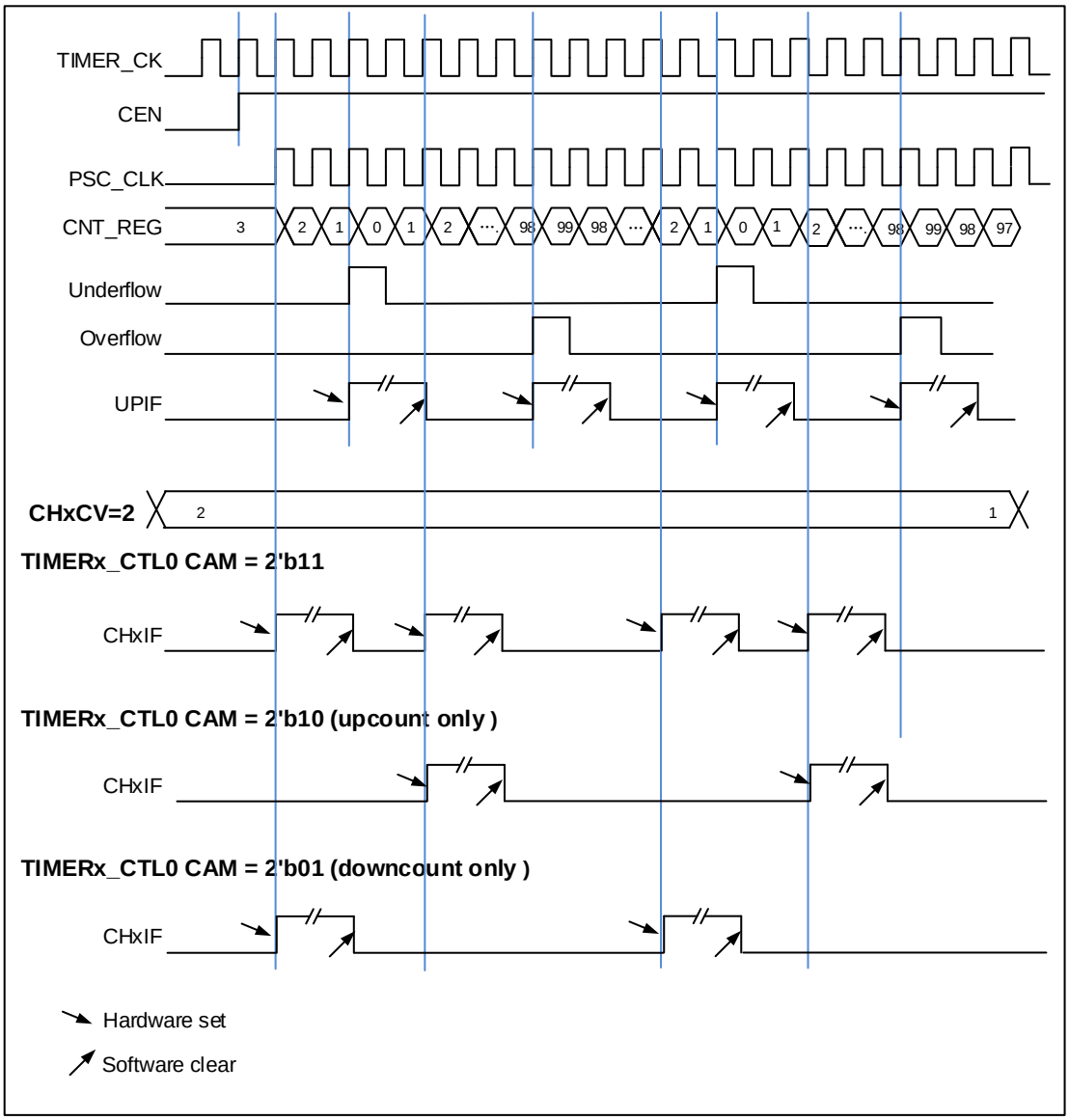
The UPIF bit in the TIMERx_INTF register can be set to 1 either when an underflow event or an overflow event occurs. While the CHxIF bit is associated with the value of CAM in TIMERx_CTL0. The details refer to ***Figure 15-8. Timing chart of center-aligned counting mode.***

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 15-8. Timing chart of center-aligned counting mode show some examples of the counter behavior when TIMERx_CAR=0x99. TIMERx_PSC=0x0

Figure 15-8. 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 15-9. Repetition timechart for center-aligned counter

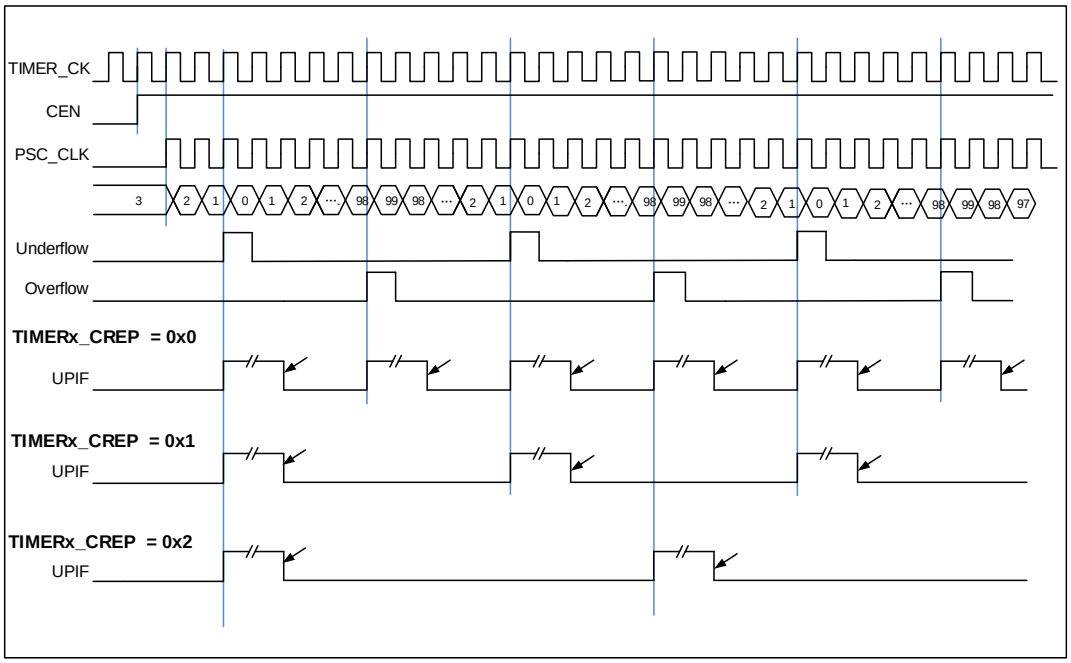


Figure 15-10. Repetition timechart for up-counter

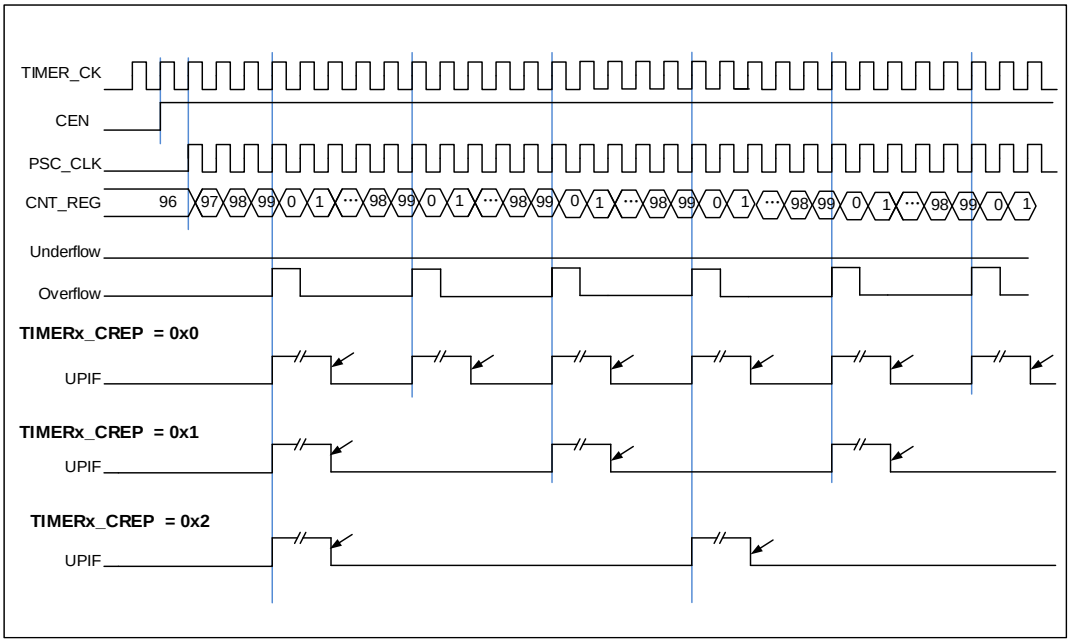
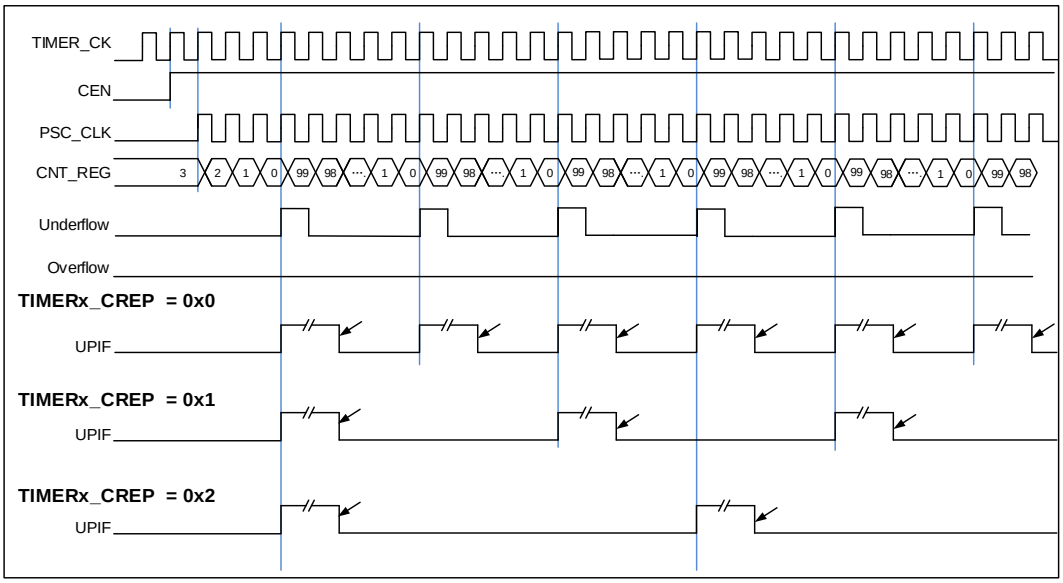


Figure 15-11. Repetition timechart for down-counter



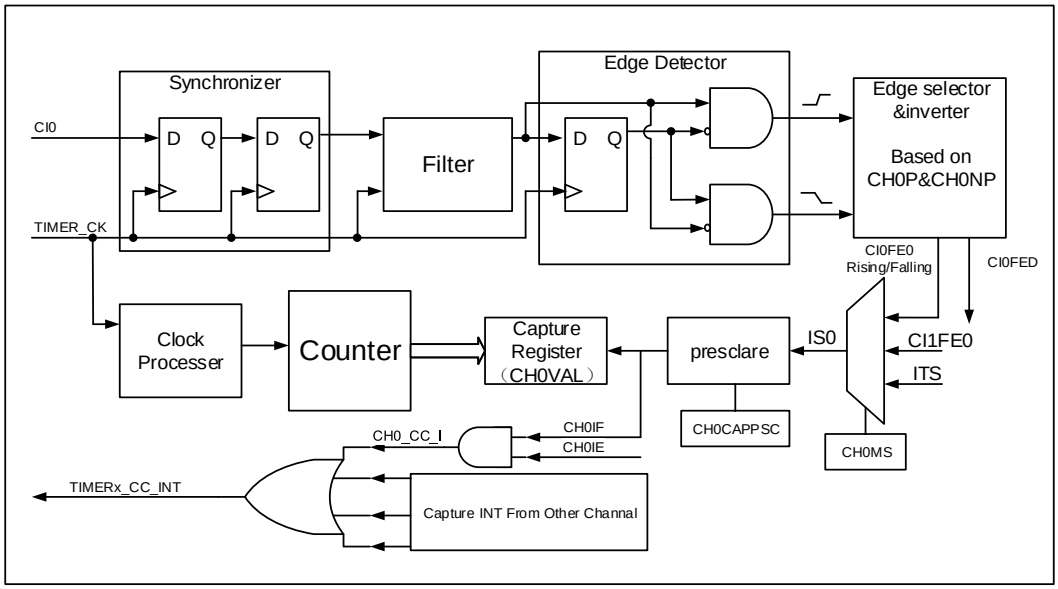
Input capture and output compare channels

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

■ Channel input capture function

Channel input capture function allows the channel to perform measurements such as pulse timing, frequency, period, duty cycle and so on. The input stage consists of a digital filter, a channel polarity selection, edge detection and a channel prescaler. When a selected edge occurs on the channel input, the current value of the counter is captured into the TIMERx_CHxCV register, at the same time the CHxIF bit is set and the channel interrupt is generated if enabled by CHxIE = 1.

Figure 15-12. Channel input capture principle



One of channels' input signals (CIx) can be chosen from the TIMERx_CHx signal or the Excusive-OR function of the TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 signals. First, the channel input signal (CIx) is synchronized to TIMER_CK domain, and then sampled by a digital filter to generate a filtered input signal. Then through the edge detector, the rising and falling edge are detected. You can select one of them by CHxP. One more selector is for the other channel and trig, controlled by CHxMS. The IC_prescaler make several the input event generate one effective capture event. On the capture event, TIMERx_CHxCV will restore the value of counter.

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

Step1: Filter configuration. (CHxCAPFLT in TIMERx_CHCTL0)

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

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

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

Step3: Capture source selection. (CHxMS in TIMERx_CHCTL0)

As soon as you select one input capture source by CHxMS, you have set the channel to input mode (CHxMS! =0x0) and TIMERx_CHxCV cannot be written any more.

Step4: Interrupt enable. (CHxIE and CHxDEN in TIMERx_DMAINTEN)

Enable the related interrupt enable; you can got the interrupt and DMA request.

Step5: Capture enables. (CHxEN in TIMERx_CHCTL2)

Result: when you wanted input signal is got, TIMERx_CHxCV will be set by counter's value. And CHxIF is asserted. If the CHxIF is high, the CHxOF will be asserted also. The interrupt and DMA request will be asserted based on the configuration of CHxIE and CHxDEN in TIMERx_DMAINTEN

Direct generation: if you want to generate a DMA request or Interrupt, you can set CHxG by software directly.

The channel input capture function can be also used for pulse width measurement from signals on the TIMERx_CHx pins. For example, PWM signal connect to CI0 input. Select channel 0 capture signals to CI0 by setting CH0MS to 2'b01 in the channel control register (TIMERx_CHCTL0) and set capture on rising edge. Select channel 1 capture signal to CI0 by setting CH1MS to 2'b10 in the channel control register (TIMERx_CHCTL0) and set capture on falling edge. The counter set to restart mode and restart on channel 0 rising edge. Then the TIMERX_CH0CV can measure the PWM period and the TIMERx_CH1CV can measure the PWM duty.

- **Channel output compare function**

Figure 15-13. Channel output compare principle (with complementary output, x=0,1,2)

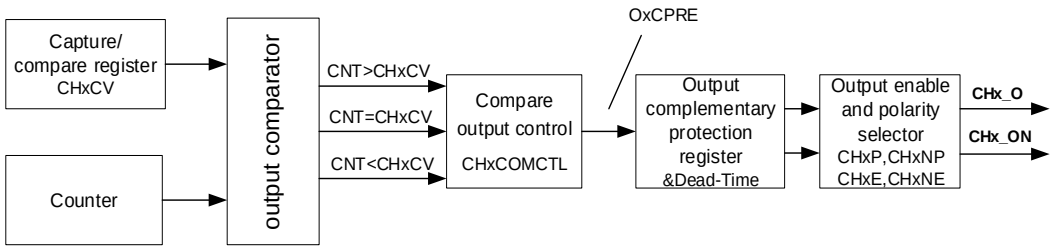


Figure 15-14. Channel output compare principle (CH3_O)

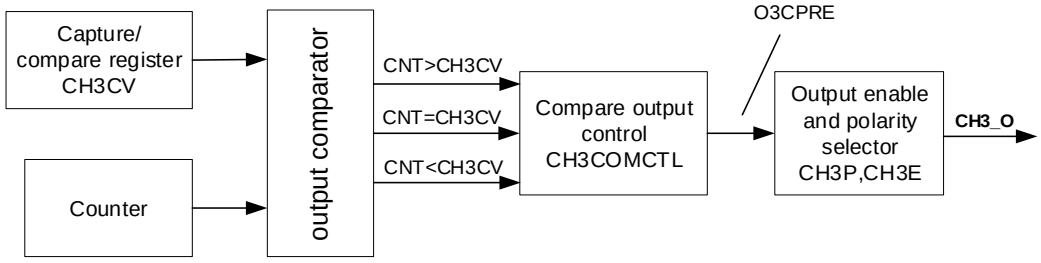


Figure 15-13. Channel output compare principle (with complementary output, x=0,1,2) and ***Figure 15-14. Channel output compare principle (CH3_O)*** show the principle circuit of channels output compare function. The relationship between the channel output signal CHx_O/CHx_ON and the OxCPRE signal (more details refer to ***Channel output prepare signal***) is described as blew: The active level of O0CPRE is high, the output level of CH0_O/CH0_ON depends on OxCPRE signal, CHxP/CHxNP bit and CH0E/CH0NE 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 outputs complementary*** 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 is 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 to several steps as below:
Step1: Clock Configuration. Such as clock source, clock prescaler and so on.

Step2: Compare mode configuration.

- Set the shadow enable mode by CHxCOMSEN
- Set the output mode (Set/Clear/Toggle) by CHxCOMCTL.
- Select the active high polarity by CHxP/CHxNP
- Enable the output by CHxEN

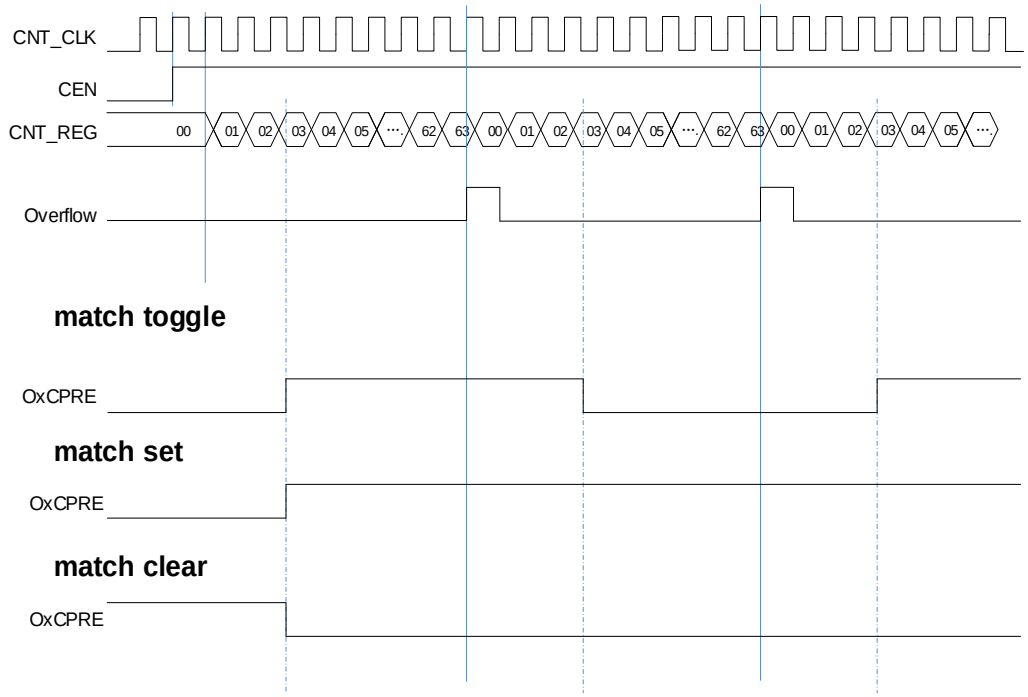
Step3: Interrupt/DMA-request enables configuration by CHxIE/CxCDE

Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV
About the TIMERx_CHxCV; you can change it on the go to meet the waveform you expected.

Step5: Start the counter by CEN.

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

Figure 15-15. 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 registers and TIMERx_CHxCV registers.

Based on the counter mode, we can also divide PWM into EAPWM (Edge aligned PWM) and CAPWM (Centre aligned PWM).

The EAPWM period is determined by TIMERx_CAR and duty cycle is determined by TIMERx_CHxCV.***Figure 15-16. Timing chart of EAPWM*** shows the EAPWM output and interrupts waveform.

The CAPWM period is determined by 2*TIMERx_CAR, and duty cycle is by 2*TIMERx_CHxCV. ***Figure 15-17. Timing chart of CAPWM*** shows the CAPWM output and interrupts waveform.

If TIMERx_CHxCV is greater than TIMERx_CAR, the output will be always active under PWM mode0 (CHxCOMCTL==3'b110).
And if TIMERx_CHxCV is equal to zero, the output will be always inactive under PWM mode0 (CHxCOMCTL==3'b110).

Figure 15-16. Timing chart of EAPWM

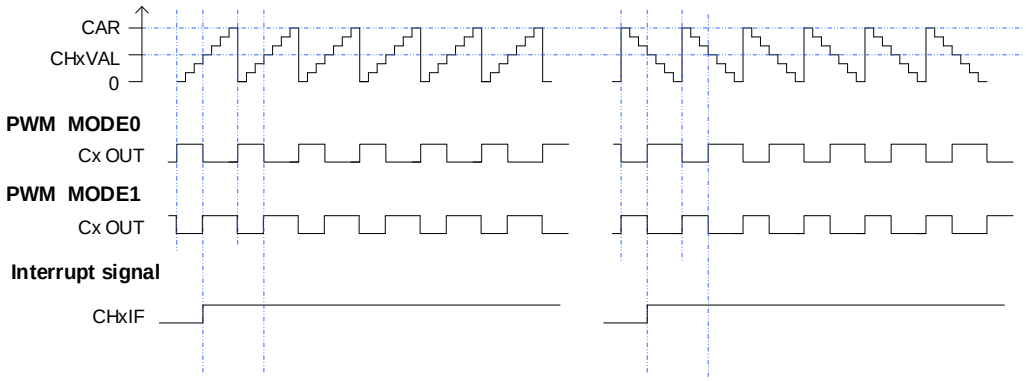
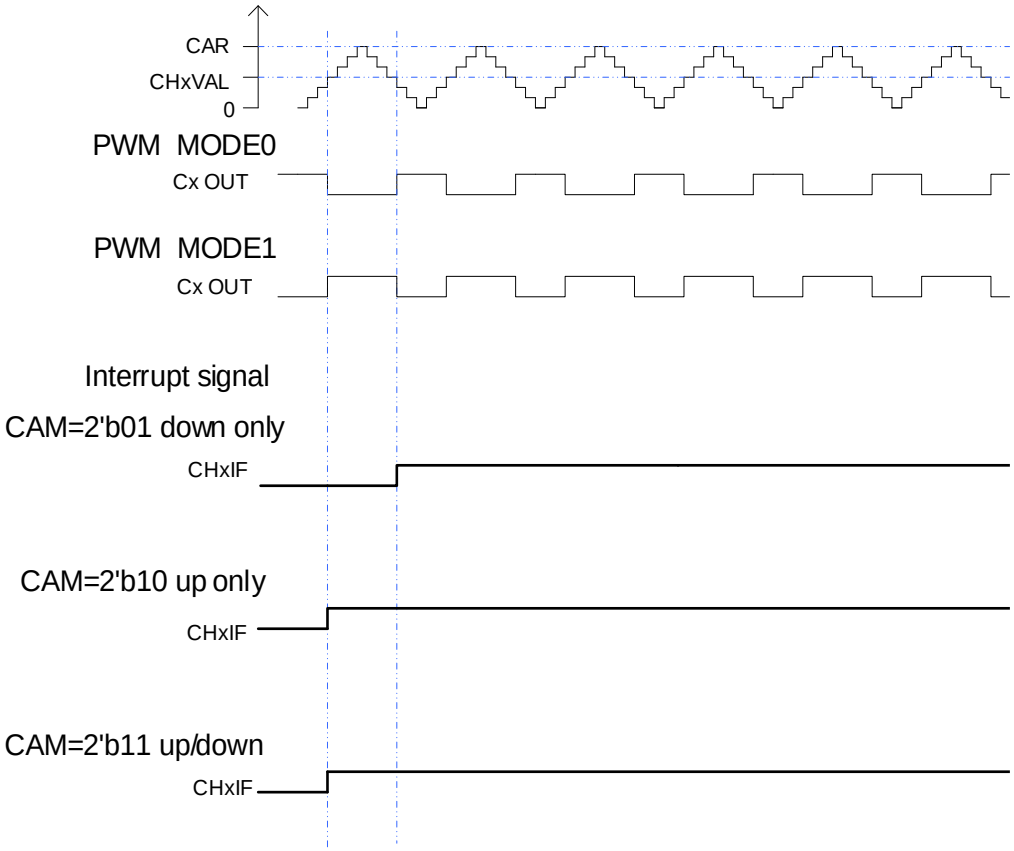


Figure 15-17. Timing chart of CAPWM



Channel output prepare signal

As is shown in ***Figure 15-13. Channel output compare principle (with complementary output, x=0,1,2)***,when the TIMERx is used in the compare match output mode, the OxCPRE signal (Channel x Output prepare signal) is defined by setting the CHxCOMCTL filed. The OxCPRE signal has several types of output function. These include, keeping the original level

by setting the CHxCOMCTL field to 0x00, set to 1 by setting the CHxCOMCTL field to 0x01, set to 0 by setting the CHxCOMCTL field to 0x02 or signal toggle by setting the CHxCOMCTL field to 0x03 when the counter value matches the content of the TIMERx_CHxCV register.

The PWM mode 0 and PWM mode 1 outputs are also another kind of OxCPRE output which is setup by setting the CHxCOMCTL field to 0x06/0x07. In these modes, the OxCPRE signal level is changed according to the counting direction and the relationship between the counter value and the TIMERx_CHxCV content. With regard to a more detail description refer to the relative bit definition.

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

The OxCPRE signal can be forced to 0 when the ETIFE signal is derived from the external ETI pin and when it is set to a high level by setting the CHxCOMCEN bit to 1 in the TIMERx_CHCTL0 register. The OxCPRE signal will not return to its active level until the next update event occurs.

Channel outputs complementary PWM

Function of complementary is for a pair of CHx_O and CHx_ON. Those two output signals cannot be active at the same time. The TIMERx has 4 channels, but only the first three channels have this function. The complementary signals CHx_O and CHx_ON are controlled by a group of parameters: the CHxEN and CHxNEN bits in the TIMERx_CHCTL2 register and the POEN, ROS, IOS, ISOx and ISOxN bits in the TIMERx_CCHP and TIMERx_CTL1 registers. The outputs polarity is determined by CHxP and CHxNP bits in the TIMERx_CHCTL2 register.

Table 15-2. 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 / CHx_ON = LOW CHx_O / CHx_ON output disable(1).	
				1	CHx_O/ CHx_ON output “off-state” (2): the CHx_O/ CHx_ON output inactive level firstly: CHx_O = CHxP, CHx_ON = CHxNP; If the clock for deadtime generator is present, after a deadtime: CHx_O = ISOx, CHx_ON = ISOxN. (3)	
			1	0		
				1		
		1	x	x	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	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 all channels expect for channel 3. The detail about the delay time, refer to the register TIMERx_CCHP.

The dead time delay insertion ensures that no two complementary signals drive the active state at the same time.

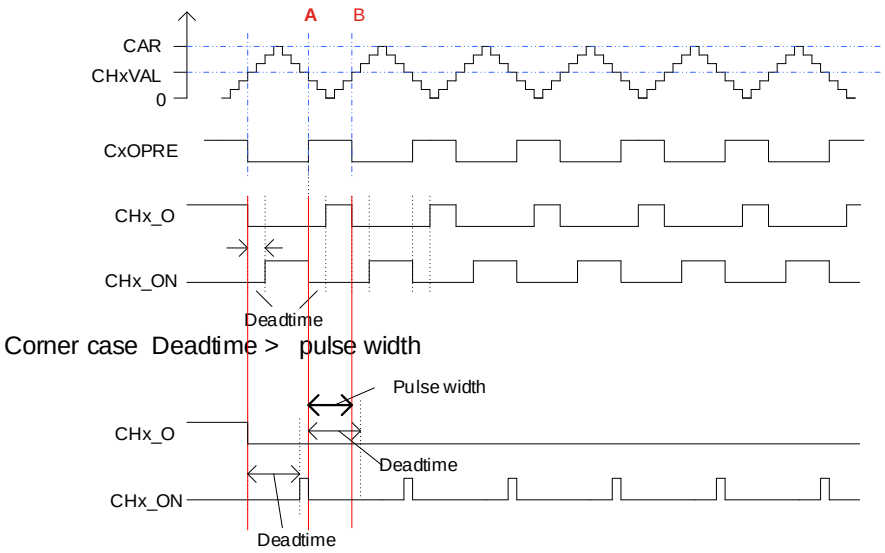
When the channel (x) match (TIMERx counter = CHxVAL) occurs, OxCPRE will be toggled because under PWM0 mode. At point A in the *Figure 15-18. Complementary output with dead-time insertion.* CHx_O signal remains at the low value until the end of the deadtime delay, while CHx_ON will be cleared at once. 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 15-18. 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 15-18. Complementary output with dead-time insertion.



Break mode

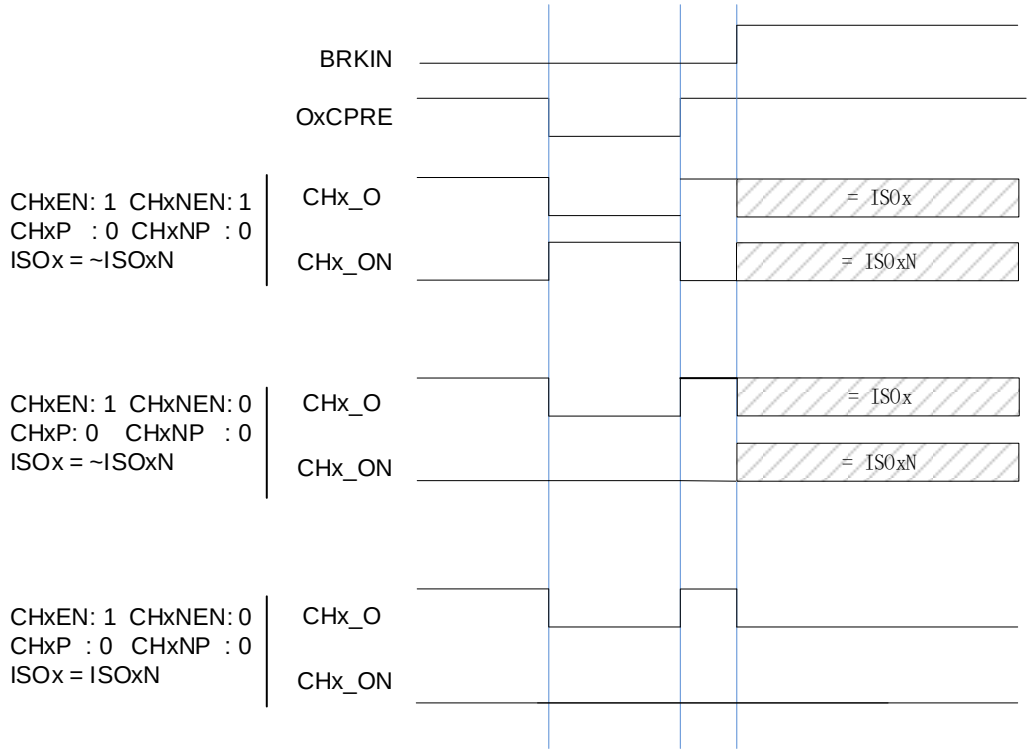
In this function, the output CHx_O and CHx_ON are controlled by the POEN, IOS and ROS bits in the TIMERx_CCHP register, ISOx and ISOxN bits in the TIMERx_CTL1 register and

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 15-19. 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 to control the counter value. Setting TSCFGy[3:0] != 4'b0000 (y=0,1,2) 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 for changing the counter direction is shown in ***Table 15-3. Counting direction in different***

quadrature decoder mode. The quadrature decoder can be regarded as an external clock with a directional selection. This means that the counter counts continuously in the interval between 0 and the counter-period value. Therefore, TIMERx_CAR register must be configured before the counter starts to count.

Table 15-3. Counting direction in different quadrature decoder mode

Counting mode	Level	CI0FE0		CI1FE1	
		Rising	Falling	Rising	Falling
CI0 only counting	CI1FE1=1	Down	Up	-	-
	CI1FE1=0	Up	Down	-	-
CI1 only counting	CI0FE0=1	-	-	Up	Down
	CI0FE0=0	-	-	Down	Up
CI0 and CI1 counting	CI1FE1=1	Down	Up	X	X
	CI1FE1=0	Up	Down	X	X
	CI0FE0=1	X	X	Up	Down
	CI0FE0=0	X	X	Down	Up

Note: "-" means "no counting"; "X" means impossible. ”0” means “low level”, ”1” means “high level”.

Figure 15-20. Counter behavior with CI0FE0 polarity non-inverted in mode 2

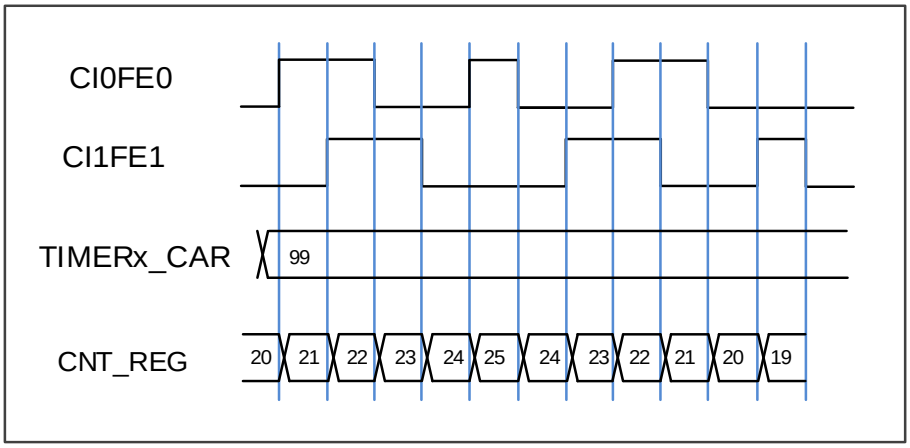
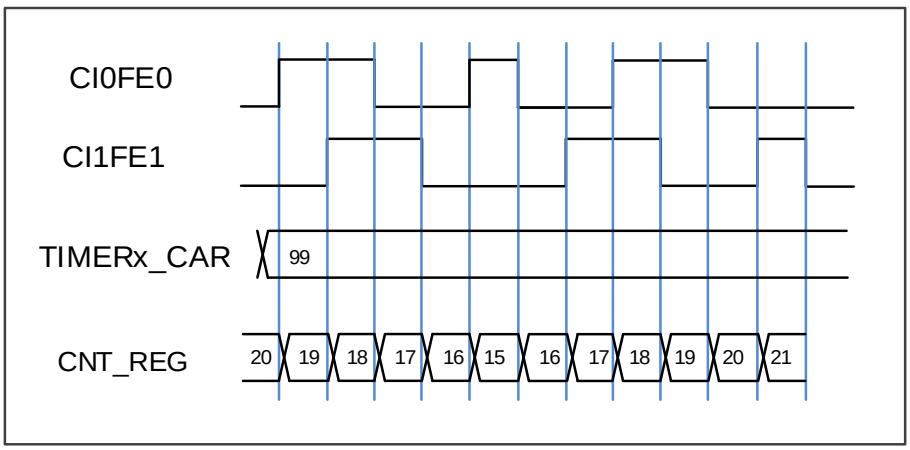


Figure 15-21. Counter behavior with CI0FE0 polarity inverted in mode 2



Hall sensor function

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

Figure 15-22. 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 using TSCFGy[3:0] in SYSCFG_TIMERxCFG register, TIMER_in and TIMER_out can be connected. TIMER_out will generate PWM signal to control BLDC motor's speed based on the ITRx. Then, the feedback circuit is finished, also you change configuration to fit your request.

About the TIMER_in, it need have input XOR function, so you can choose from Advanced/General L0 TIMER.

And TIMER_out need have functions of complementary and Dead-time, so only advanced timer can be chosen. Else, based on the timers' internal connection relationship, pair's timers can be selected. For example:

TIMER_in (TIMER1) -> TIMER_out (TIMER0 ITI1)

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 15-22. Hall sensor is used to BLDC motor

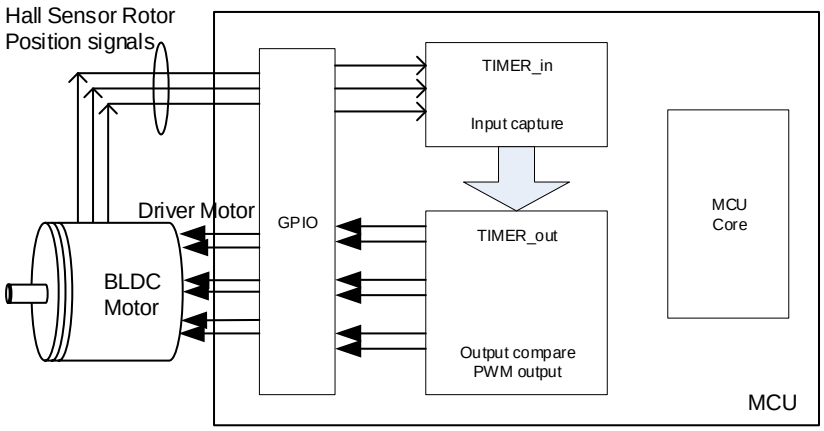
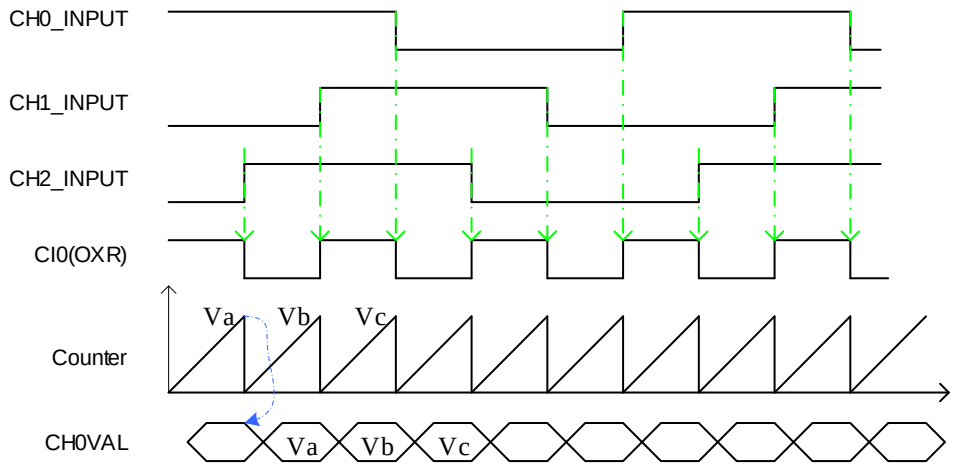
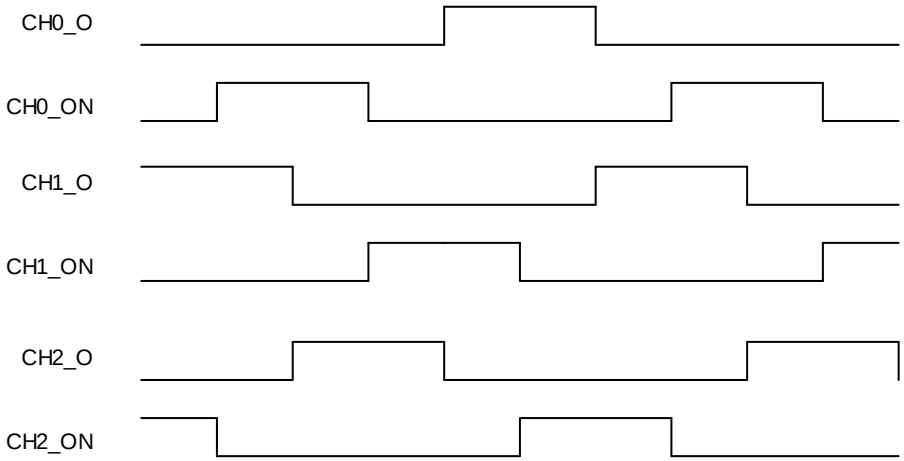


Figure 15-23. Hall sensor timing between two timers

Advanced/General L0 TIMER_in under input capture mode



Advanced TIMER_out under output compare mode(PWM with Dead-time)

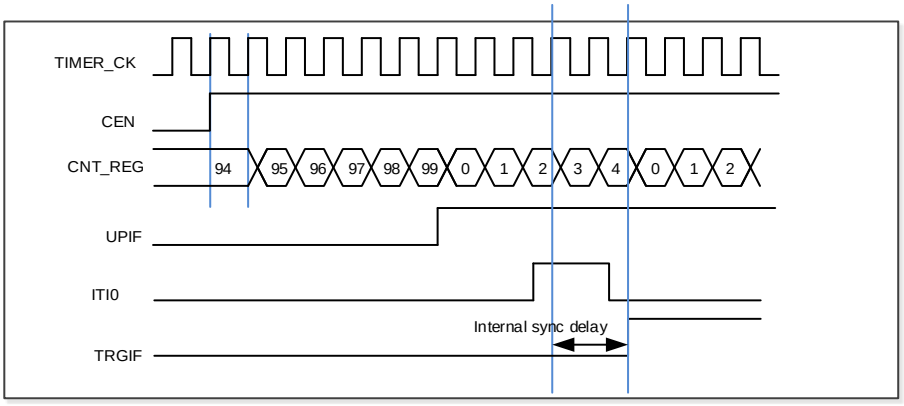
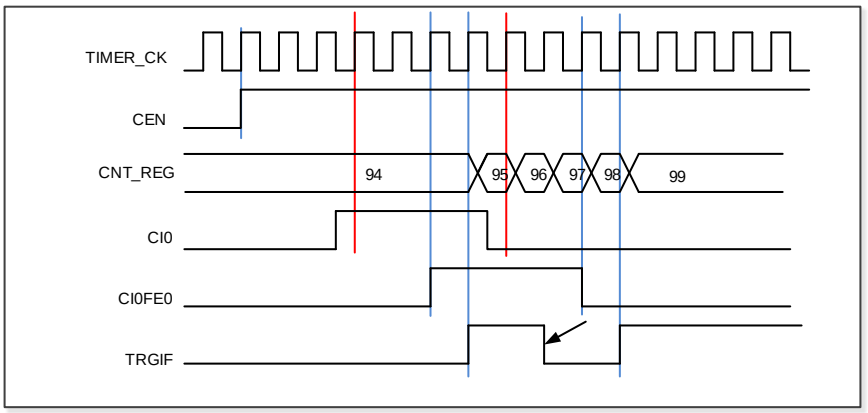


Master-slave management

The TIMERx can be synchronized with a trigger in several modes including restart mode, pause mode and event mode which is selected by the TSCFGy[3:0] in SYSCFG_TIMERxCFG(y=3,4 ,5).

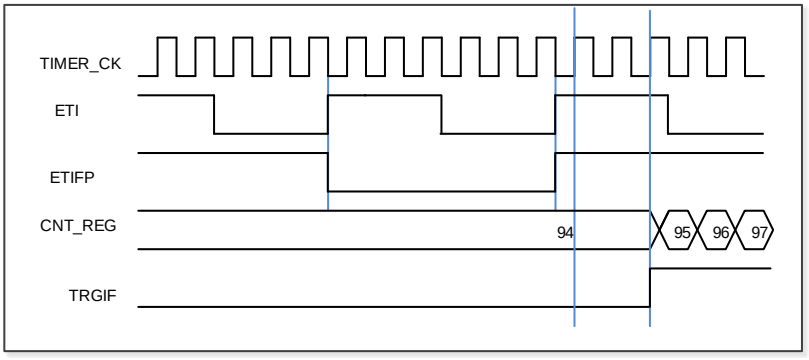
Table 15-4. Examples of slave mode

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
LIST	TSCFGy[3:0] y=3 (restart mode) y=4 (pause mode) y=5 (event mode)	TSCFGy[3:0] 0001: ITI0 0010: ITI1 0011: ITI2 0100: ITI3 0101: CI0F_ED 0110: 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

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
		0111: CI1FE1 1000: ETIFP	the trigger source, configure the ETP for polarity selection and inversion.	used. For the ETIFP, filter can be used by configuring ETFC and prescaler can be used by configuring ETPSC.
Exam1	**Restart mode** The counter will be cleared and restart when a rising edge of trigger input comes.	TSCFG3[3:0] = 4'b 0001.ITI0 is selected.	For ITI0, no polarity selector can be used.	For the ITI0, no filter and prescaler can be used.
	Figure 15-24. Restart mode 			
Exam2	**Pause mode** The counter will be paused when the trigger input is low, and it will start when the trigger input is high.	TSCFG4[3:0] = 4'b 0110 CI0FE0 is selected.	TI0S=0 (Non-xor) [CH0NP=0, CH0P=0] CI0FE0 does not invert. The capture event will occur on the rising edge only.	Filter is bypassed in this example.
	Figure 15-25. Pause mode 			

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
Exam3	**Event mode** The counter will start to count when a rising edge of trigger input comes.	TSCFG5[3:0] = 4'b 1000 ETIFP is selected.	ETP = 0, the polarity of ETI does not change.	ETPSC = 1, ETI is divided by 2. ETFC = 0, ETI does not filter.

Figure 15-26. Event mode



Single pulse mode

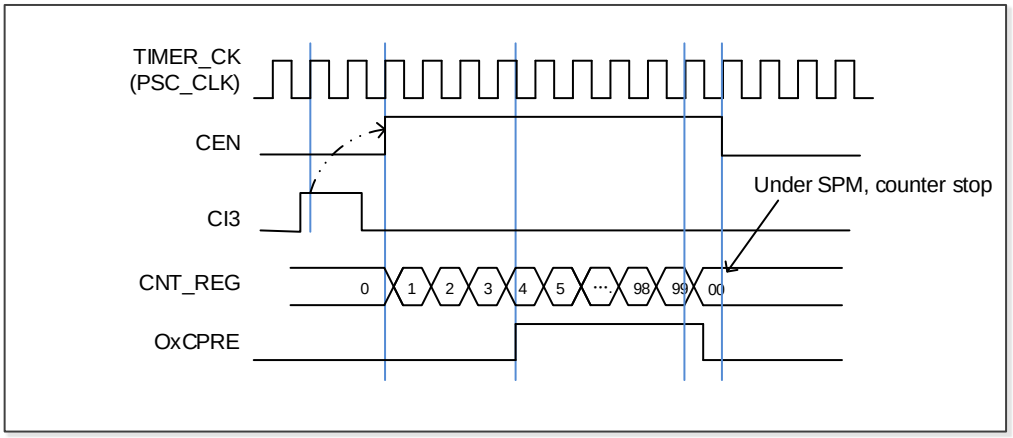
Single pulse mode is opposite to the repetitive mode, which can be enabled by setting SPM in TIMERx_CTL0. When you set SPM, the counter will be clear and stop when the next update event. In order to get pulse waveform, you can set the TIMERx to PWM mode or compare by CHxCOMCTL.

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

In the single pulse mode, the trigger active edge which sets the CEN bit to 1 will enable the counter. However, there exist several clock delays to perform the comparison result between the counter value and the TIMERx_CHxCV value. In order to reduce the delay to a minimum value, the user can set the CHxCOMFEN bit in each TIMERx_CHCTL0/1 register. After a trigger rising occurs in the single pulse mode, the OxCPRE signal will immediately be forced to the state which the OxCPRE signal will change to, as the compare match event occurs without taking the comparison result into account. The CHxCOMFEN bit is available only when the output channel is configured to operate in the PWM0 or PWM1 output mode and the trigger source is derived from the trigger signal.

Figure 15-27. Single pulse mode, TIMERx CHxCV = 4, TIMERx CAR=99 shows an example.

Figure 15-27. Single pulse mode, TIMERx_CHxCV = 4, TIMERx_CAR=99



Timers interconnection

The timers can be internally connected for timer chaining or synchronization. This can be implemented by configuring one timer to operate in the master mode while configuring another timer to be in the slave mode. The following figures show several examples of trigger selection for the master mode and slave mode.

Some interconnection examples:

- TIMER2 as the prescaler for TIMER0

1. Configure TIMER2 in master mode and select its update event (UPE) as trigger output (MMC=3'b010 in the TIMER2_CTL1 register). Then TIMER2 drives a periodic signal on each counter overflow.
2. Configure TIMER2 period (TIMER2_CAR register).
3. Configure TIMER0 in external clock mode 1 and select the TIMER0 input trigger source from TIMER2 (TSCFG6[3:0] = 0001 in the_SYSCFG_TIMERxCFG register).
4. Start TIMER0 by writing '1 in the CEN bit (TIMER0_CTL0 register).Start TIMER2 by writing '1 in the CEN bit (TIMER2_CTL0 register).

- Start TIMER0 with TIMER2's enable/update signal

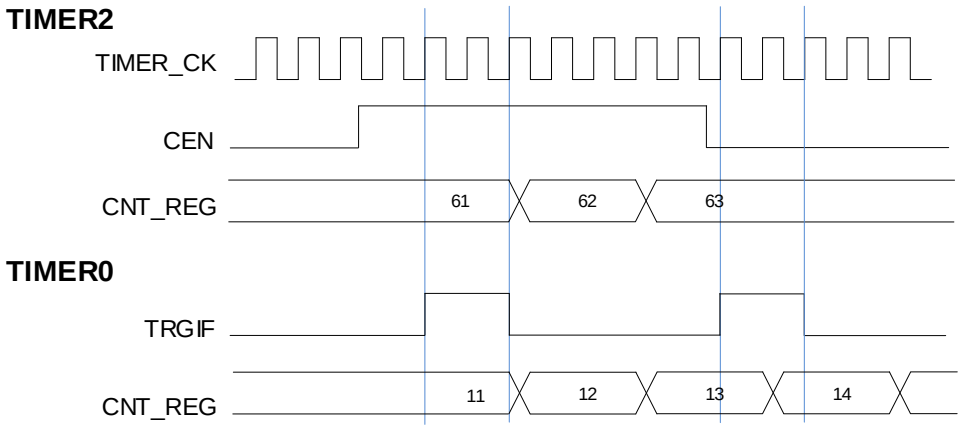
First, enable TIMER0 with the enable signal of TIMER2. Refer to ***Figure 15-28. Trigger mode of TIMER0 controlled by enable signal of TIMER2***. TIMER0 starts counting from its current value with the divided internal clock after being triggered by TIMER2 enable signal output.

When TIMER0 receives the trigger signal, its CEN bit is set and the counter counts until TIMER0 is disabled. Both clock frequency of the counters are divided by 3 from TIMER_CK ($f_{PSC_CLK} = f_{TIMER_CK}/3$). Steps are shown as follows:

1. Configure TIMER2 in master mode to send its enable signal as trigger output (MMC=3'b001 in the TIMER2_CTL1 register).
2. Configure TIMER2 master mode to send its enable signal as trigger output, and configure TIMER0 to select the input trigger from TIMER2 (TSCFG5[3:0] = 0011 in the SYSCFG_TIMERxCFG register).

3. Start TIMER2 by writing 1 to the CEN bit (TIMER2_CTL0 register).

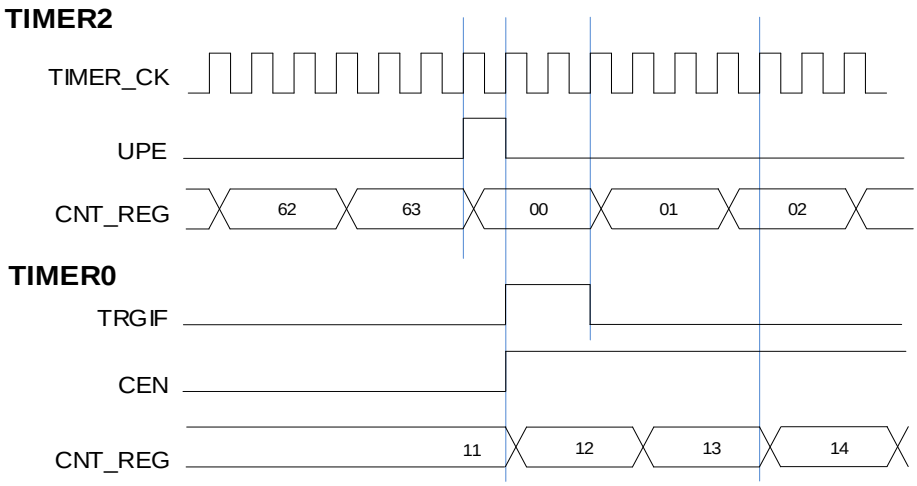
Figure 15-28. Trigger mode of TIMER0 controlled by enable signal of TIMER2



In this example, the update event can also be used as trigger source instead of enable signal. Refer to ***Figure 15-29. Trigger mode of TIMER0 controlled by update signal of TIMER2***. Steps are shown as follows:

1. Configure TIMER2 in master mode to send its update event (UPE) as trigger output (MMC=3'b010 in the TIMER2_CTL1 register).
2. Configure the TIMER2 period (TIMER2_CARL registers).
3. Configure TIMER0 to get the input trigger from TIMER2, configure TIMER0 in event mode. (TSCFG5[3:0] = 0011 in the_ SYSCFG_TIMERxCFG register).
4. Start TIMER2 by writing '1' to the CEN bit (TIMER2_CTL0 register).

Figure 15-29. Trigger mode of TIMER0 controlled by update signal of TIMER2



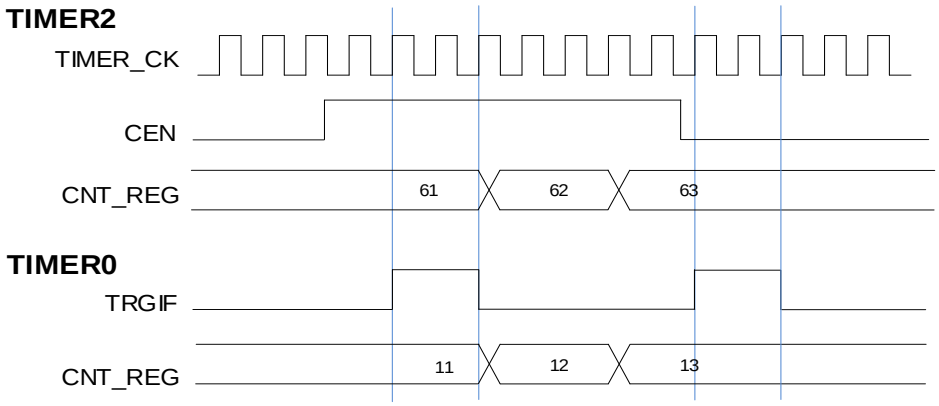
- Enable TIMER0 to count with the enable/O0CPRE signal of TIMER2.

In this example, TIMER0 is enabled with the enable signal of TIMER2. Refer to ***Figure 15-30. Pause mode of TIMER0 controlled by enable signal of TIMER2*** TIMER0 counts with the divided internal clock only when TIMER2 is enabled. Both clock frequency of the counters are

divided by 3 from TIMER_CK ($f_{PSC_CLK} = f_{TIMER_CK}/3$). Steps are shown as follows:

1. Configure TIMER2 in master mode and output enable signal as trigger output (MMC=3'b001 in the TIMER2_CTL1 register).
2. Configure TIMER0 to get the input trigger from TIMER2, configure TIMER0 in pause mode (TSCFG5 [3:0] = 0011 in the_ SYSCFG_TIMERxCFG register).
3. Enable TIMER0 by writing '1' to the CEN bit (TIMER0_CTL0 register).
4. Start TIMER2 by writing '1' to the CEN bit (TIMER2_CTL0 register).
5. Stop TIMER2 by writing '0' to the CEN bit (TIMER2_CTL0 register).

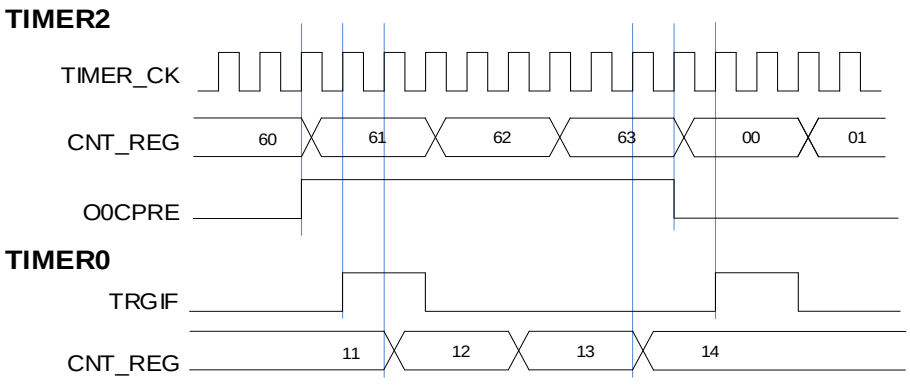
Figure 15-30. Pause mode of TIMER0 controlled by enable signal of TIMER2



In this example, O0CPRE can also be used as trigger source instead of enable signal output. Steps are shown as follows:

1. Configure TIMER2 in master mode and O0CPRE as trigger output (MMS=3'b100 in the TIMER2_CTL1 register).
2. Configure the TIMER2 O0CPRE waveform (TIMER2_CHCTL0 register).
3. Configure TIMER0 to get the input trigger from TIMER2, configure TIMER0 in pause mode (TSCFG5[3:0] = 0011 in the_ SYSCFG_TIMERxCFG register).
4. Enable TIMER0 by writing '1' to the CEN bit (TIMER0_CTL0 register).
5. Start TIMER2 by writing '1' to the CEN bit (TIMER2_CTL0 register).

Figure 15-31. Pause mode of TIMER0 controlled by O0CPREF signal of TIMER2



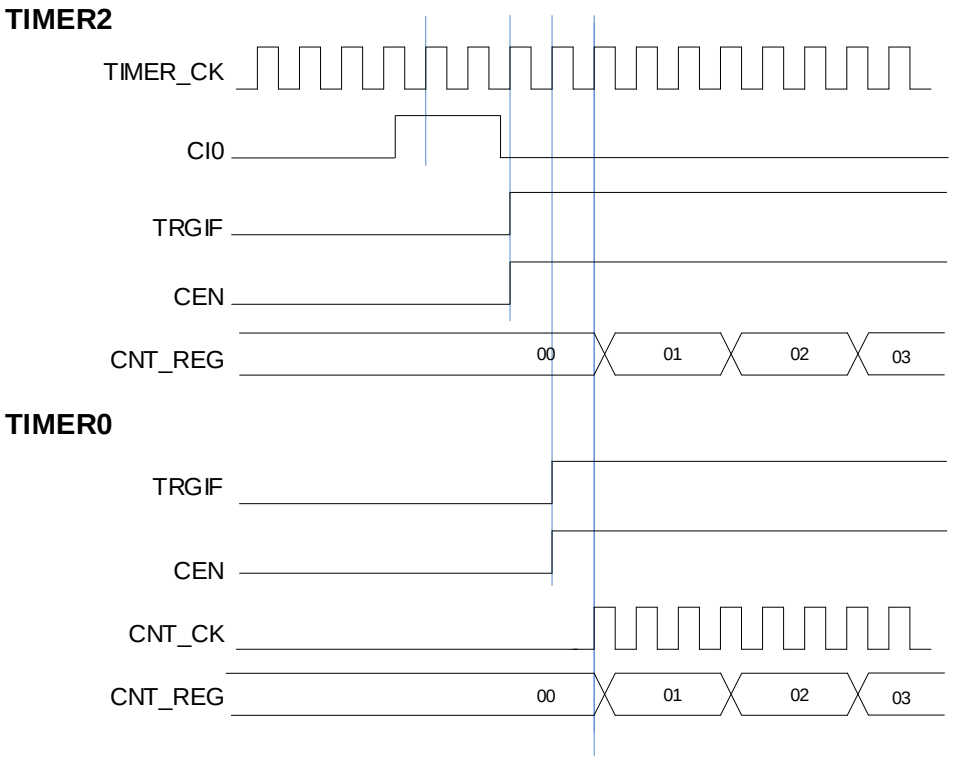
- Using an external trigger to start two timers synchronously.

The start of TIMER0 is triggered by the enable signal of TIMER2, and TIMER2 is triggered by its CI0 input rising edge. To ensure that two timers start synchronously, TIMER2 must be configured in master/slave mode. Steps are shown as follows:

1. Configure TIMER2 in event mode and select the CI0F_ED as TIMER2 input trigger source (TSCFG5[3:0] = 4b'0101 in the_SYSCFG_TIMERxCFG register).
2. Configure TIMER2 in master/slave mode by writing MSM=1 (TIMER2_SMCFG register).
3. Configure TIMER0 in event mode and select the TIMER2 as TIMER0 input trigger source (TSCFG5[3:0] = 4b'0011 in the_SYSCFG_TIMERxCFG register).

When the CI0 signal of TIMER2 generates a rising edge, two timer counters start counting synchronously with the internal clock and both TRGIF flags are set.

Figure 15-32. Trigger TIMER0 and TIMER2 by the CI0 signal of TIMER2



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 RISCV halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register is set to 1, the TIMERx counter stops.

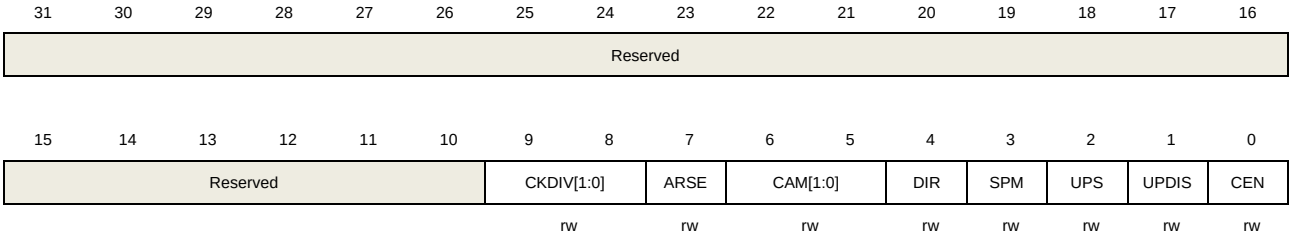
15.1.5. TIMERx registers(x=0)

TIMER0 access base address: 0x4001 0000

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 encoder mode, this bit is read only.
3	SPM	Single pulse mode. 0: Single pulse mode disable. The counter continues after update event. 1: Single pulse mode enable. The counter counts until the next update event occurs.
2	UPS	Update source This bit is used to select the update event sources by software. 0: These events generate update interrupts or DMA requests: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: This event generates update interrupts or DMA requests: The counter generates an overflow or underflow event
1	UPDIS	Update disable. This bit is used to enable or disable the update event generation. 0: Update event enable. When an update event occurs, the corresponding shadow registers are loaded with their preloaded values. These events generate update event: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: Update event disable. **Note:** When this bit is set to 1, setting UPG bit or 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 encoder mode.

Control register 1 (TIMERx_CTL1)

Address offset: 0x04
Reset value: 0x0000 0000

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ISO3	ISO2N	ISO2	ISO1N	ISO1	ISO0N	ISO0	TI0S	MMC[2:0]			DMAS	CCUC	Reserved	CCSE
	rw	rw	rw	rw	rw	rw	rw	rw	rw			rw	rw		rw

Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14	ISO3	Idle state of channel 3 output Refer to ISO0 bit
13	ISO2N	Idle state of channel 2 complementary output Refer to ISO0N bit
12	ISO2	Idle state of channel 2 output Refer to ISO0 bit
11	ISO1N	Idle state of channel 1 complementary output Refer to ISO0N bit
10	ISO1	Idle state of channel 1 output Refer to ISO0 bit
9	ISO0N	Idle state of channel 0 complementary output 0: When POEN bit is reset, CH0_ON is set low. 1: When POEN bit is reset, CH0_ON is set high This bit can be modified only when PROT [1:0] bits in TIMERx_CCHP register is 00.
8	ISO0	Idle state of channel 0 output 0: When POEN bit is reset, CH0_O is set low. 1: When POEN bit is reset, CH0_O is set high The CH0_O output changes after a dead-time if CH0_ON is implemented. This bit can be modified only when PROT [1:0] bits in TIMERx_CCHP register is 00.
7	TI0S	Channel 0 trigger input selection 0: The TIMERx_CH0 pin input is selected as channel 0 trigger input. 1: The result of combinational XOR of TIMERx_CH0, 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: Reset. When the UPG bit in the TIMERx_SWEVG register is set or a reset is generated by the slave mode controller, a TRGO pulse occurs. And in the latter case, the signal on TRGO is delayed compared to the actual reset. 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	Reserved						
rw	rw	rw		rw				rw							

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15	ETP	External trigger polarity This bit specifies the polarity of ETI signal 0: ETI is active at rising edge or high level . 1: ETI is active at falling edge or low level .
14	SMC1	Part of SMC for enable External clock mode1. In external clock mode 1, the counter is clocked by any active edge on the ETIFP signal. 0: External clock mode 1 disabled 1: External clock mode 1 enabled. When the slave mode is configured as restart mode, pause mode or event mode, the timer can still work in the external clock 1 mode by setting this bit. But the TSCFGy[3:0](y=3,4,5) bits must not be 1000 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 SYSCFG_TIMERxCFG register.
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
6:0	Reserved	Must be kept at reset value.

DMA and interrupt enable register (TIMERx_DMAINTEN)

Address offset: 0x0C
Reset value: 0x0000 0000

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TRGDEN	CMTDEN	CH3DEN	CH2DEN	CH1DEN	CH0DEN	UPDEN	BRKIE	TRGIE	CMTIE	CH3IE	CH2IE	CH1IE	CH0IE	UPIE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14	TRGDEN	Trigger DMA request enable 0: disabled

		1: enabled
13	CMTDEN	Commutation DMA request enable 0: disabled 1: enabled
12	CH3DEN	Channel 3 capture/compare DMA request enable 0: disabled 1: enabled
11	CH2DEN	Channel 2 capture/compare DMA request enable 0: disabled 1: enabled
10	CH1DEN	Channel 1 capture/compare DMA request enable 0: disabled 1: enabled
9	CH0DEN	Channel 0 capture/compare DMA request enable 0: disabled 1: enabled
8	UPDEN	Update DMA request enable 0: disabled 1: enabled
7	BRKIE	Break interrupt enable 0: disabled 1: enabled
6	TRGIE	Trigger interrupt enable 0: disabled 1: enabled
5	CMTIE	commutation interrupt enable 0: disabled 1: enabled
4	CH3IE	Channel 3 capture/compare interrupt enable 0: disabled 1: enabled
3	CH2IE	Channel 2 capture/compare interrupt enable 0: disabled 1: enabled
2	CH1IE	Channel 1 capture/compare interrupt enable 0: disabled 1: enabled

1	CH0IE	Channel 0 capture/compare interrupt enable 0: disabled 1: enabled
0	UPIE	Update interrupt enable 0: disabled 1: enabled

Interrupt flag register (TIMERx_INTF)

Address offset: 0x10
Reset value: 0x0000 0000

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CH3OF	CH2OF	CH1OF	CH0OF	Reserved	BRKIF	TRGIF	CMTIF	CH3IF	CH2IF	CH1IF	CH0IF	UPIF
			rc_w0	rc_w0	rc_w0	rc_w0	.	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12	CH3OF	Channel 3 over capture flag Refer to CH0OF description
11	CH2OF	Channel 2 over capture flag Refer to CH0OF description
10	CH1OF	Channel 1 over capture flag Refer to CH0OF description
9	CH0OF	Channel 0 over capture flag When channel 0 is configured in input mode, this flag is set by hardware when a capture event occurs while CH0IF flag has already been set. This flag is cleared by software. 0: No over capture interrupt occurred 1: Over capture interrupt occurred
8	Reserved	Must be kept at reset value.
7	BRKIF	Break interrupt flag When the break input is inactive, the bit is set by hardware. When the break input is inactive, the bit can be cleared by software. 0: No active level break has been detected.

1: An active level has been detected.

6	TRGIF	Trigger interrupt flag This flag is set on trigger event and cleared by software. When in pause mode, both edges on trigger input generates a trigger event, otherwise, only an active edge on trigger input can generates a trigger event. 0: No trigger event occurred. 1: Trigger interrupt occurred.
5	CMTIF	Channel commutation interrupt flag This flag is set by hardware when 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 Channel 0 is set to input mode, this bit will be reset by reading TIMERx_CH0CV. 0: No Channel 0 interrupt occurred 1: Channel 0 interrupt occurred
0	UPIF	Update interrupt flag This bit is set by hardware 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	BRKG	TRGG	CMTG	CH3G	CH2G	CH1G	CH0G	UPG
	w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	BRKG	Break event generation This bit is set by software 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
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 working only if an internal trigger input is selected, through TSCFG7[3:0] bit-field in SYSCFG_TIMERxCFG (x=0) register.
7	CH0COMCEN	Channel 0 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the O0CPRE signal will be cleared. 0: Channel 0 output compare clear disable 1: Channel 0 output compare clear enable
6:4	CH0COMCTL[2:0]	Channel 0 compare output control This bit-field specifies the compare output mode of the the output prepare signal O0CPRE. In addition, the high level of O0CPRE is the active level, and CH0_O and CH0_ON channels polarity depends on CH0P and CH0NP bits. 000: Timing mode. The O0CPRE signal keeps stable, independent of the comparison between the register TIMERx_CH0CV and the counter TIMERx_CNT. 001: Set the channel output. O0CPRE signal is forced high when the counter is equals to the output compare register TIMERx_CH0CV. 010: Clear the channel output. O0CPRE signal is forced low when the counter is equals to the output compare register TIMERx_CH0CV. 011: Toggle on match. O0CPRE toggles when the counter is equals to the output compare register TIMERx_CH0CV. 100: Force low. O0CPRE is forced to low level. 101: Force high. O0CPRE is forced to high level. 110: PWM mode0. When counting up, O0CPRE is high when the counter is smaller than TIMERx_CH0CV, and low otherwise. When counting down, O0CPRE is low when the counter is larger than TIMERx_CH0CV, and high otherwise. 111: PWM mode1. When counting up, O0CPRE is low when the counter is smaller than TIMERx_CH0CV, and high otherwise. When counting down, O0CPRE is high when the counter is larger than TIMERx_CH0CV, and low otherwise. If configured in PWM mode, the O0CPRE level changes only when the output compare mode is adjusted from “Timing” mode to “PWM” mode or the comparison result changes. This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH0MS bit-filed is 00(COMPARE MODE).
3	CH0COMSEN	Channel 0 compare output shadow enable When this bit is set, the shadow register of TIMERx_CH0CV register, which updates at each update event, will be enabled. 0: Channel 0 output compare shadow disable 1: Channel 0 output compare shadow enable The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1) This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH0MS bit-filed is 00.
2	CH0COMFEN	Channel 0 output compare fast enable

When this bit is set, the effect of an event on the trigger in input on the capture/compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH0_O is set to the compare level independently from the result of the comparison.
0: Channel 0 output quickly compare disable.
1: Channel 0 output quickly compare enable.

1:0 CH0MS[1:0] Channel 0 I/O mode selection
This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH0EN bit in TIMERx_CHCTL2 register is reset).).
00: Channel 0 is programmed as output mode
01: Channel 0 is programmed as input mode, IS0 is connected to CI0FE0
10: Channel 0 is programmed as input mode, IS0 is connected to CI1FE0
11: Channel 0 is programmed as input mode, IS0 is connected to ITS
Note: When CH0MS[1:0]=11, it is working only if an internal trigger input is selected, through TSCFG7[3:0] bit-field in SYSCFG_TIMERxCFG (x=0) 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

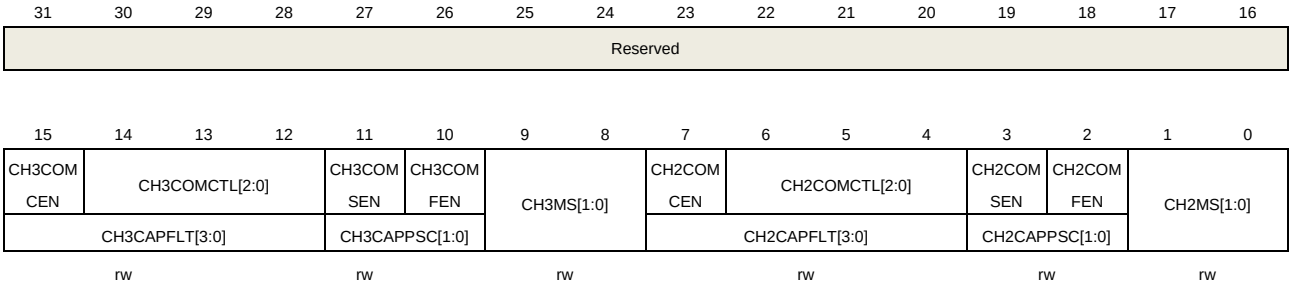
Same as Output compare mode

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
31:16	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 working only if an internal trigger input is selected, through TSCFG7[3:0] bit-field in SYSCFG_TIMERxCFG (x=0) register.
7	CH2COMCEN	Channel 2 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the O2CPRE signal will be cleared. 0: Channel 2 output compare clear disable 1: Channel 2 output compare clear enable
6:4	CH2COMCTL[2:0]	Channel 2 compare output control This bit-field specifies the compare output mode of the the output prepare signal O0CPRE. In addition, the high level of O0CPRE is the active level, and CH0_O and CH0_ON channels polarity depends on CH0P and CH0NP bits. 000: Timing mode. The O2CPRE signal keeps stable, independent of the comparison between the output compare register TIMERx_CH2CV and the counter TIMERx_CNT. 001: Set the channel output. O2CPRE signal is forced high when the counter is equals to the output compare register TIMERx_CH2CV. 010: Clear the channel output. O2CPRE signal is forced low when the counter is equals to the output compare register TIMERx_CH2CV. 011: Toggle on match. O2CPRE toggles when the counter is equals to the output compare register TIMERx_CH2CV. 100: Force low. O2CPRE is forced to low level. 101: Force high. O2CPRE is forced to high level. 110: PWM mode 0. When counting up, O2CPRE is high when the counter is smaller than TIMERx_CH2CV, and low otherwise. When counting down, O2CPRE is low when the counter is larger than TIMERx_CH2CV, and high otherwise. 111: PWM mode 1. When counting up, O2CPRE is low when the counter is smaller than TIMERx_CH2CV, and high otherwise. When counting down, O2CPRE is high

		when the counter is larger than TIMERx_CH2CV, and low otherwise. If configured in PWM mode, the O2CPRE level changes only when the output compare mode is adjusted from “Timing” mode to “PWM” mode or the comparison result changes. This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH2MS bit-filed is 00(COMPARE MODE).
3	CH2COMSEN	Channel 2 compare output shadow enable When this bit is set, the shadow register of TIMERx_CH2CV register, which updates at each update event will be enabled. 0: Channel 2 output compare shadow disable 1: Channel 2 output compare shadow enable The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1) This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 11 and CH0MS bit-filed is 00.
2	CH2COMFEN	Channel 2 output compare fast enable When this bit is set, the effect of an event on the trigger in input on the capture/compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH2_O is set to the compare level independently from the result of the comparison. 0: Channel 2 output quickly compare disable. 1: Channel 2 output quickly compare enable.
1:0	CH2MS[1:0]	Channel 2 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH2EN bit in TIMERx_CHCTL2 register is reset).). 00: Channel 2 is programmed as output mode 01: Channel 2 is programmed as input mode, IS2 is connected to CI2FE2 10: Channel 2 is programmed as input mode, IS2 is connected to CI3FE2 11: Channel 2 is programmed as input mode, IS2 is connected to ITS. **Note:** When CH2MS[1:0]=11, it is working only if an internal trigger input is selected, through TSCFG7[3:0] bit-field in SYSCFG_TIMERxCFG (x=0) 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	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

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	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															

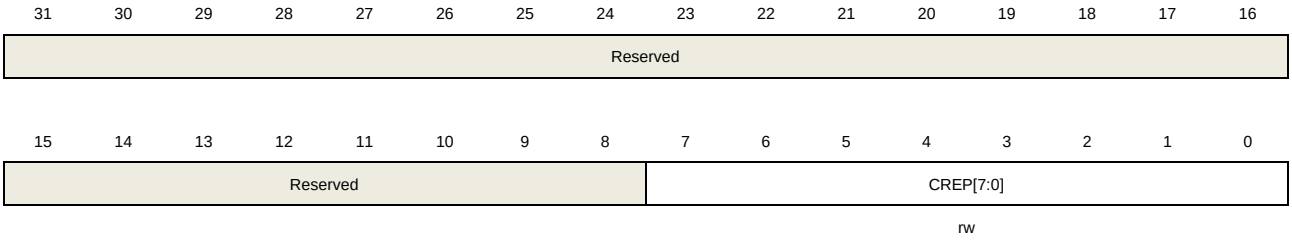
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CARL[15:0]	Counter auto reload value This bit-filed specifies the auto reload value of the counter.

Counter repetition register (TIMERx_CREP)

Address offset: 0x30

Reset value: 0x0000 0000

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



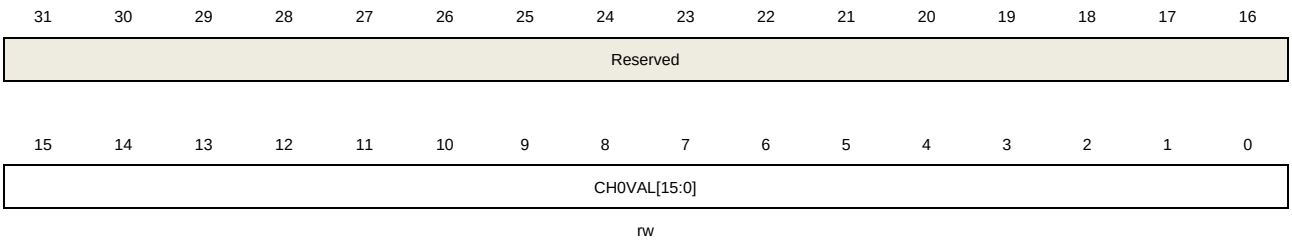
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	CREP[7:0]	Counter repetition value This bit-filed specifies the update event generation rate. Each time the repetition counter counting down to zero, an update event is generated. The update rate of the shadow registers is also affected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

31	30	29	28	27	26	25	24	23	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)

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)

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 input polarity This bit specifies the polarity of the BRKIN input signal. 0: BRKIN input active low 1; BRKIN input active high
12	BRKEN	Break input 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 15-2. 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 15-2. 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

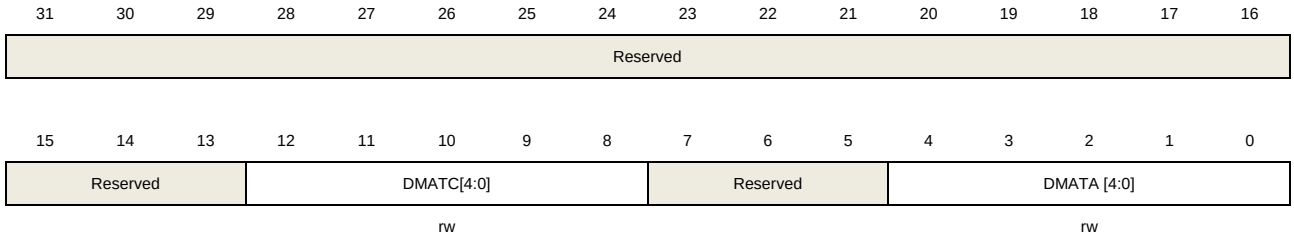
Note:

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).

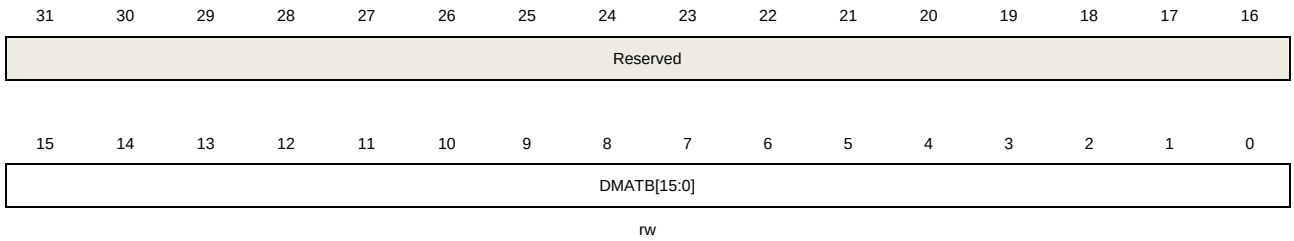


Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:8	DMATC [4:0]	DMA transfer count This filed defines the number(n) of the register that DMA will access(R/W), n = (DMATC [4:0] +1). DMATC [4:0] is from 5'b0_0000 to 5'b1_0001.
7:5	Reserved	Must be kept at reset value.
4:0	DMATA [4:0]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



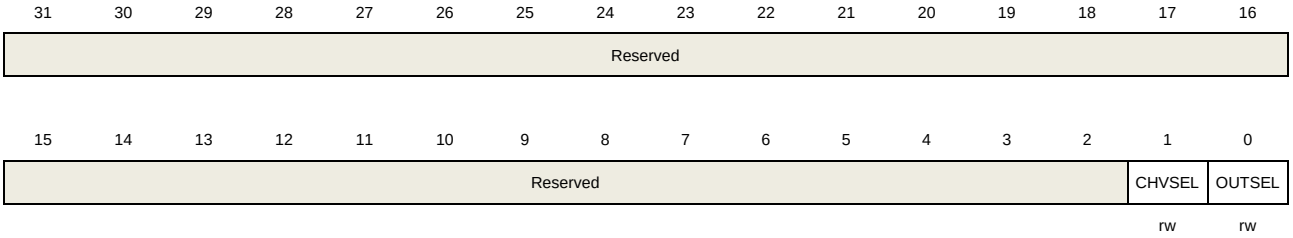
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	DMATB[15:0]	DMA transfer buffer When a read or write operation is assigned to this register, the register located at the address range (Start Addr + Transfer Timer* 4) will be accessed. The transfer Timer is calculated by hardware, 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 is 0, the output disabled

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

15.2.1. Overview

The general level0 timer module (Timer1, 2) is a four-channel timer that supports input capture, output compare. They can generate PWM signals to control motor or be used for power management applications. The general level0 time reference is 32-bit(TIMER1/2) that can be used as an unsigned counter.

In addition, the general level0 timers can be programmed and be used to count or time external events that drive other timers.

Timer and timer are completely independent, but there may be synchronized to provide a larger timer with their counters incrementing in unison.

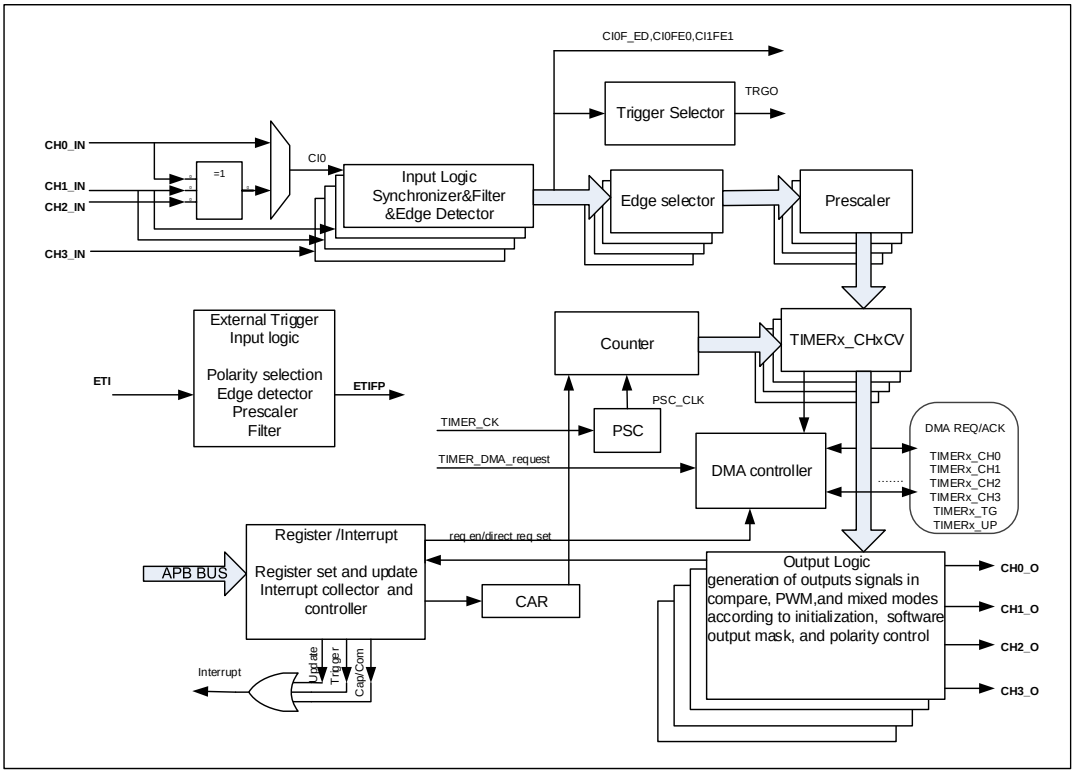
15.2.2. Characteristics

- Total channel num: 4.
- Counter width: 32bit.
- Source of count clock is selectable:
  internal clock, internal trigger, external input, external trigger.
- Multiple counter modes: count up, count down, count up/down.
- Quadrature decoder: used to track motion and determine both rotation direction and position.
- Hall sensor: for 3-phase motor control.
- Programmable prescaler: 16 bit. Factor can be changed on the go.
- Each channel is user-configurable:
  Input capture mode, output compare mode, programmable PWM mode, single pulse mode
- Auto-reload function.
- Interrupt output or DMA request on: update, trigger event, and compare/capture event.
- Daisy chaining of timer modules to allow a single timer to initiate multiple timing events.
- Timer synchronization allows selected timers to start counting on the same clock cycle.
- Timer master-slave management.

15.2.3. Block diagram

Figure 15-33. General Level 0 timer block diagram provides details on the internal configuration of the general level0 timer.

Figure 15-33. General Level 0 timer block diagram



15.2.4. Function overview

Clock source configuration

The general level0 TIMER has the capability of being clocked by either the CK_TIMER or an alternate clock source controlled by TSCFGy[3:0] in SYSCFG_TIMER0CFG(y=0,1…6). When alternate clock source is used, the SYSCFG clock must be enabled.

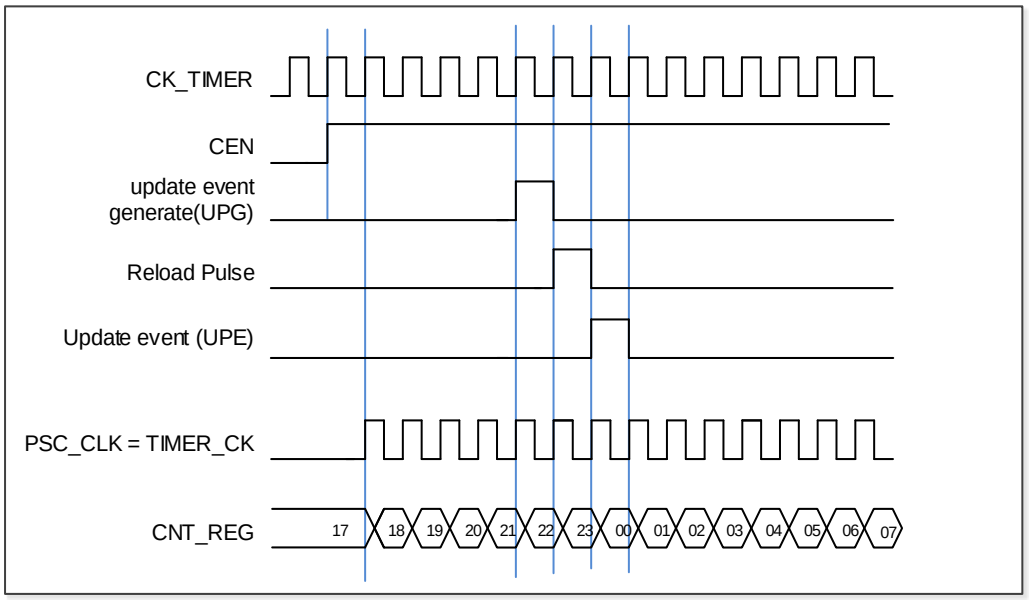
- TSCFGy[3:0] = 4'b0000 in SYSCFG_TIMER0CFG(y=0,1…6). Internal clock CK_TIMER is selected as timer clock source which is from module RCU.

The default clock source is the CK_TIMER for driving the counter prescaler when TSCFGy[3:0] = 4'b0000 in SYSCFG_TIMER0CFG(y=0,1…6). 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 TSCFGy[3:0] !=4'b0000 in SYSCFG_TIMER0CFG(y=0,1,2,6), the prescaler is clocked by other clock sources selected in the TSCFG6[3:0] register, more details will be introduced later. When the TSCFGy[3:0] (y=3,4,5) are setting to an available value, the internal clock TIMER_CK is the counter prescaler driving clock source.

Figure 15-34. Normal mode, internal clock divided by 1



- TSCFG6[3:0] are setting to an available value (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 TSCFG6[3:0] to 0x5, 0x6, 0x7.

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 TSCFG6[3:0] to 0x1,0x2,0x3,0x4.

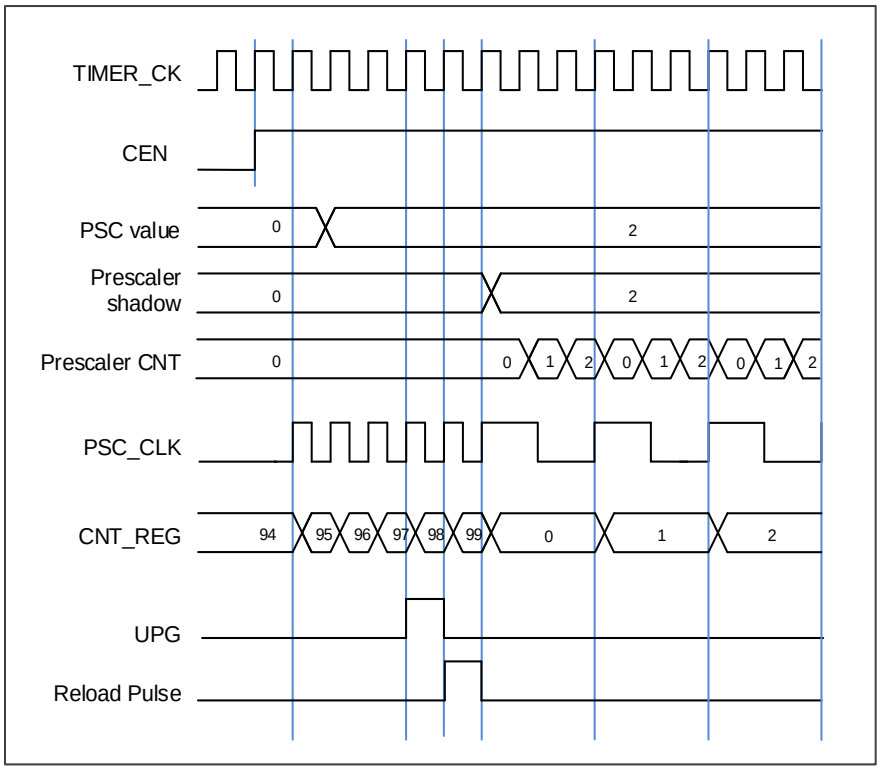
- SMC1= 1'b1 (external clock mode 1). External input ETI is selected as timer clock source.

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 TSCFG6[3:0] to 0x8. 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 15-35. 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 restarts to count once again from 0. 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 15-36. Timing chart of up counting mode, PSC=0/2 and ***Figure 15-37. Timing chart of up counting mode, change TIMERx_CAR ongoing*** show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 15-36. Timing chart of up counting mode, PSC=0/2

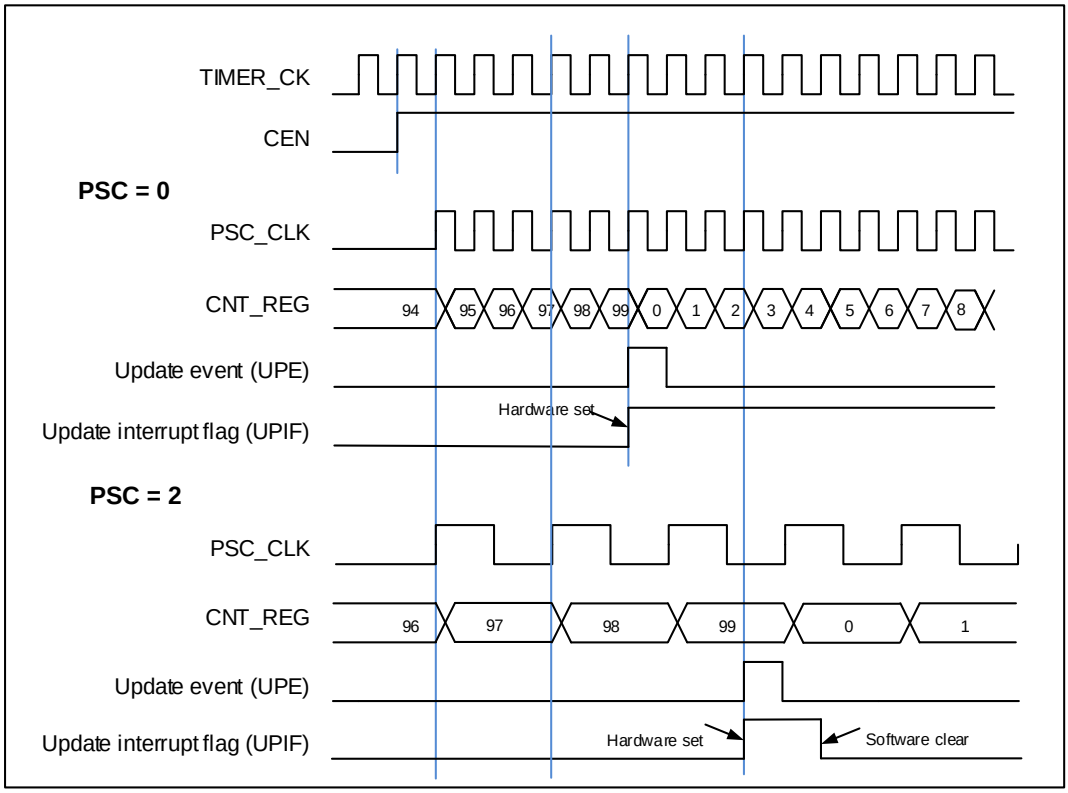
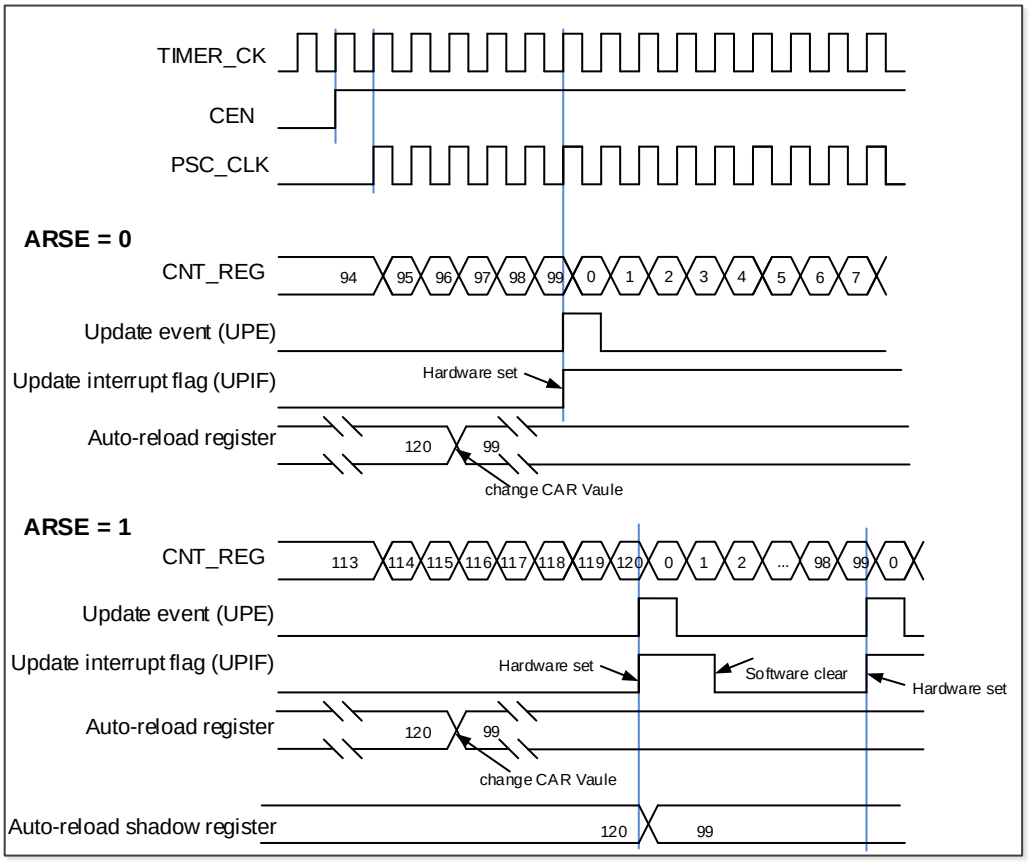


Figure 15-37. Timing chart of up counting mode, change TIMERx_CAR ongoing



Counter down counting

In this mode, the counter counts down continuously from the counter-reload value, which is defined in the TIMERx_CAR register, to 0 in a count-down direction. Once the counter reaches to 0, the counter restarts to count again from the counter-reload value. The update event is generated at each counter underflow. The counting direction bit DIR in the TIMERx_CTL0 register should be set to 1 for the down-counting mode.

When the update event is set by the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to the counter-reload value and generates an update event.

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

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

Figure 15-38. Timing chart of down counting mode,PSC=0/2 and *Figure 15-39. Timing chart of down counting mode, change TIMERx_CAR ongoing* show some examples of the counter behavior for different clock frequencies when TIMERx_CAR=0x99.

Figure 15-38. Timing chart of down counting mode,PSC=0/2

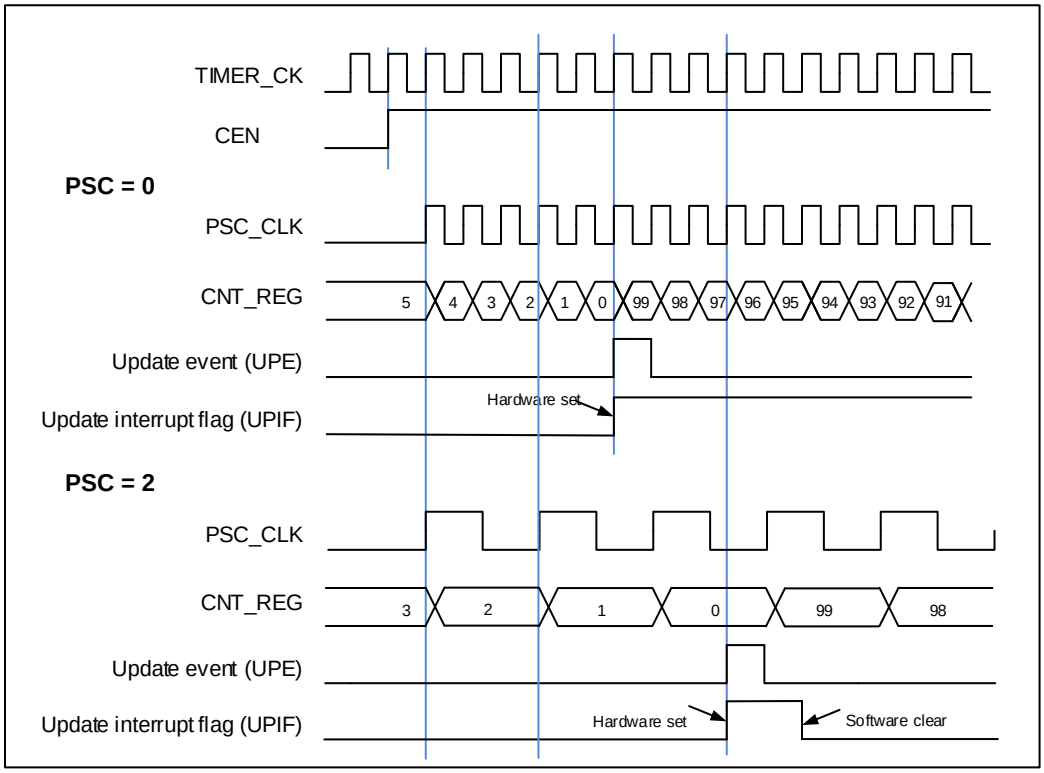
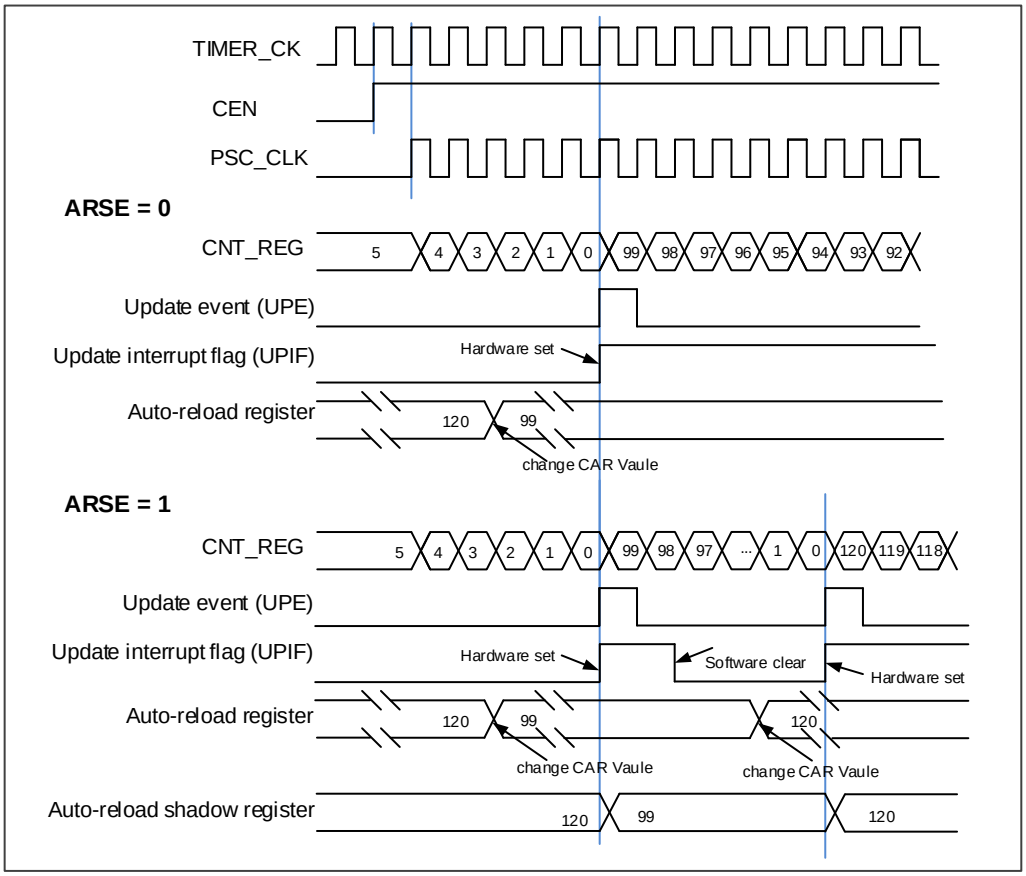


Figure 15-39. Timing chart of down counting mode, change TIMERx_CAR ongoing



Counter center-aligned counting

In this mode, the counter counts up from 0 to the counter-reload value and then counts down to 0 alternatively. The Timer module generates an overflow event when the counter counts to the counter-reload value subtract 1 in the up-counting direction and generates an underflow event when the counter counts to 1 in the down-counting mode. The counting direction bit DIR in the TIMERx_CTL0 register is read-only and indicates the counting direction when in the center-aligned mode.

Setting the UPG bit in the TIMERx_SWEVG register will initialize the counter value to 0 and generates an update event irrespective of whether the counter is counting up or down in the center-align counting mode.

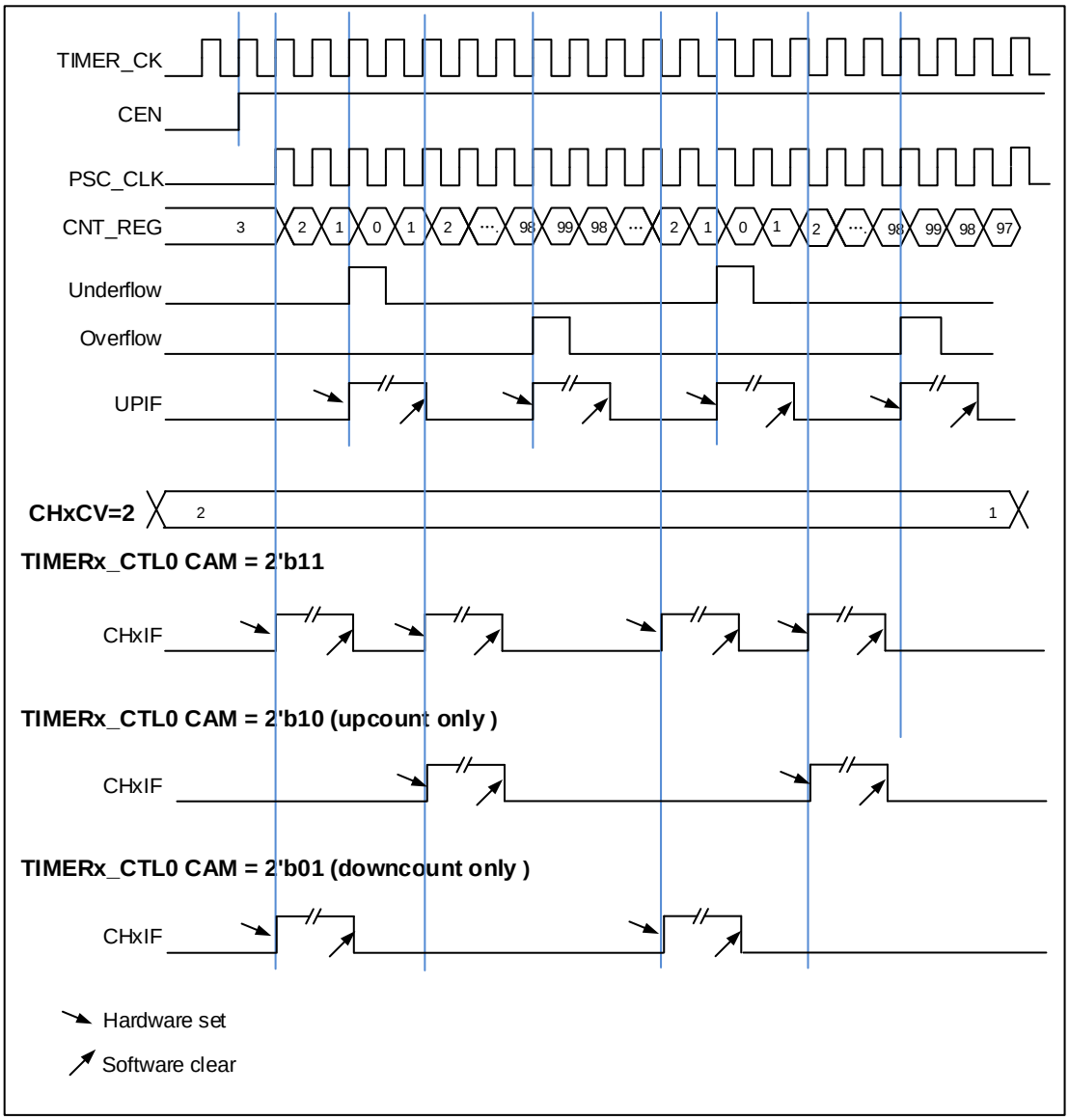
The UPIF bit in the TIMERx_INTF register can be set to 1 either when an underflow event or an overflow event occurs. While the CHxIF bit is associated with the value of CAM in TIMERx_CTL0. The details refer to ***Figure 15-40. Timing chart of center-aligned counting mode***

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

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

Figure 15-40. Timing chart of center-aligned counting mode show some examples of the counter behavior when TIMERx_CAR=0x99. TIMERx_PSC=0x0

Figure 15-40. Timing chart of center-aligned counting mode



Input capture and output compare channels

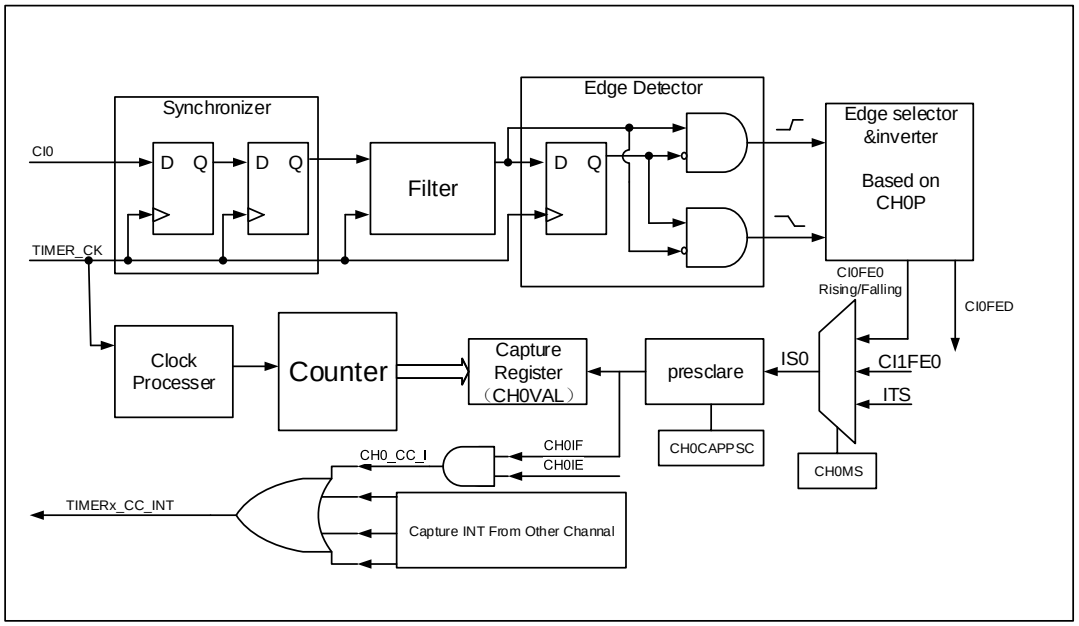
The general level0 Timer has four independent channels which can be used as capture inputs or compare match outputs. Each channel is built around a channel capture compare register including an input stage, channel controller and an output stage.

- 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 enabled by CHxIE = 1.

Figure 15-41. Channel input capture principle



One of channels' input signals (CIx) can be chosen from the TIMERx_CHx signal or the Excusive-OR function of the TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 signals. First, the channel input signal (CIx) is synchronized to TIMER_CK domain, and then sampled by a digital filter to generate a filtered input signal. Then through the edge detector, the rising and falling edge are detected. You can select one of them by CHxP. One more selector is for the other channel and trig, controlled by CHxMS. The IC_prescaler make several the input event generate one effective capture event. On the capture event, TIMERx_CHxCV will restore the value of Counter.

So the process can be divided to several steps as below:

Step1: Filter Configuration. (CHxCAPFLT in TIMERx_CHCTL0)

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

Step2: Edge Selection. (CHxP in TIMERx_CHCTL2)

Rising or falling edge, choose one by CHxP.

Step3: Capture source Selection. (CHxMS in TIMERx_CHCTL0)

As soon as you select one input capture source by CHxMS, you have set the channel to input mode (CHxMS!=0x0) and TIMERx_CHxCV cannot be written any more.

Step4: Interrupt enable. (CHxIE and CHxDEN in TIMERx_DMAINTEN)

Enable the related interrupt enable; you can got the interrupt and DMA request.

Step5: Capture enables. (CHxEN in TIMERx_CHCTL2)

Result: When you wanted input signal is got, TIMERx_CHxCV will be set by counter's value. And CHxIF is asserted. If the CHxIF is high, the CHxOF will be asserted also. The interrupt and DMA request will be asserted based on the your configuration of CHxIE and CHxDEN in

TIMERx_DMAINTEN

Direct generation: If you want to generate a DMA request or interrupt, you can set CHxG by software directly.

The 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 15-42. Channel output compare principle (x=0,1,2,3)

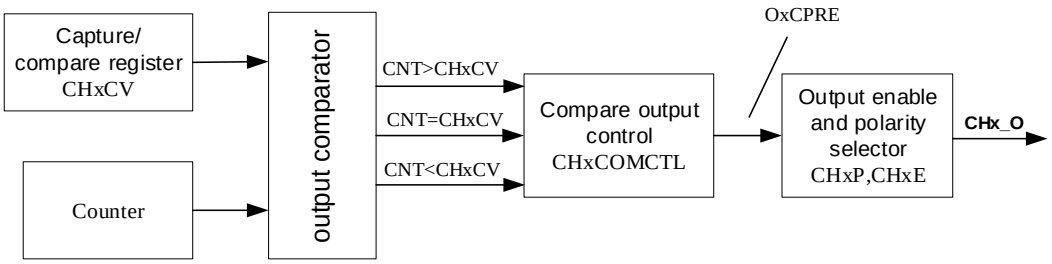


Figure 15-42. Channel output compare principle (x=0,1,2,3) shows the logic circuit of output compare mode. The relationship between the channel output signal CHx_O and the OxCPRE signal (more details refer to *Channel output prepare signal*) is described as blew: The active level of O0CPRE is high, the output level of CH0_O depends on OxCPRE signal, CHxP bit and CH0P 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 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 CxCDE=1.

So the process can be divided to several steps as below:

Step1: Clock configuration. Such as clock source, clock prescaler and so on.

Step2: Compare mode configuration.

* Set the shadow enable mode by CHxCOMSEN
* Set the output mode (Set/Clear/Toggle) by CHxCOMCTL.
* Select the active high polarity by CHxP
* Enable the output by CHxEN

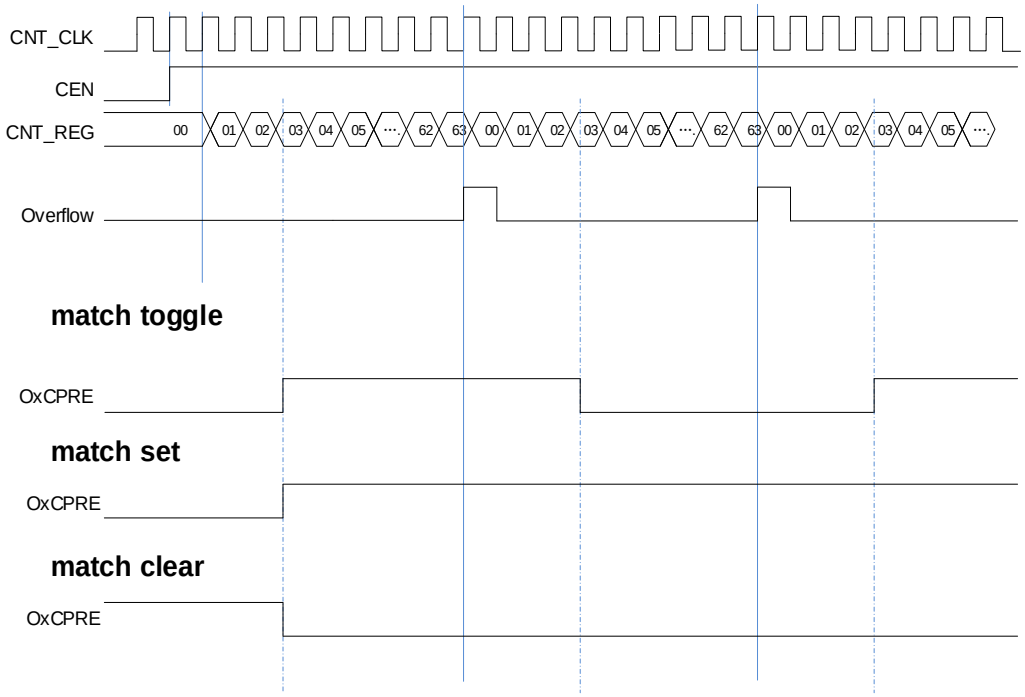
Step3: Interrupt/DMA-request enables configuration by CHxIE/CxCDE

Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV.
About the CHxVAL, you can change it on the go to meet the waveform you expected.

Step5: Start the counter by CEN.

Figure 15-43. Output-compare in three modes show the three compare modes toggle/set/clear. CAR=0x63, CHxVAL=0x3

Figure 15-43. 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 outputs PWM waveform according to the TIMERx_CAR registers and TIMERx_CHxCV registers.

Based on the counter mode, we have can also divide PWM into EAPWM (Edge aligned PWM) and CAPWM (Centre aligned PWM).

The EAPWM period is determined by TIMERx_CAR and duty cycle is by TIMERx_CHxCV. *Figure 15-44. 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 15-45. Timing chart of CAPWM*** shows the CAPWM output and interrupts waveform.

If TIMERx_CHxCV is greater than TIMERx_CAR, the output will be always active under PWM mode0 (CHxCOMCTL==3'b110).
And if TIMERx_CHxCV is equal to zero, the output will be always inactive under PWM mode0 (CHxCOMCTL==3'b110).

Figure 15-44. Timing chart of EAPWM

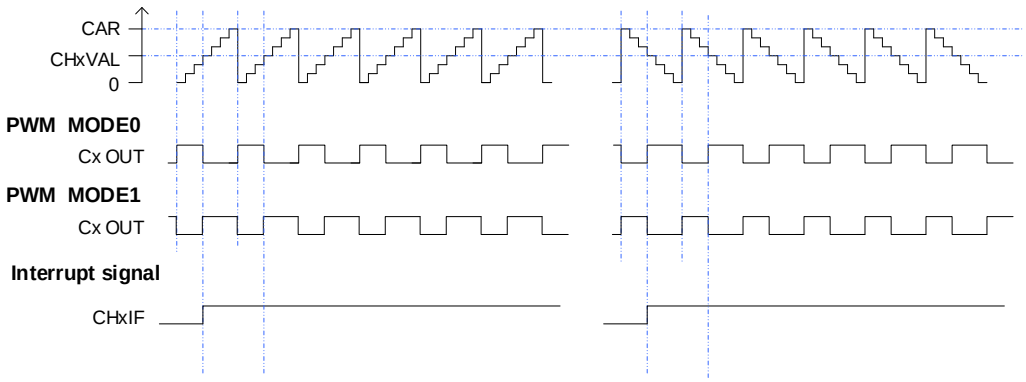
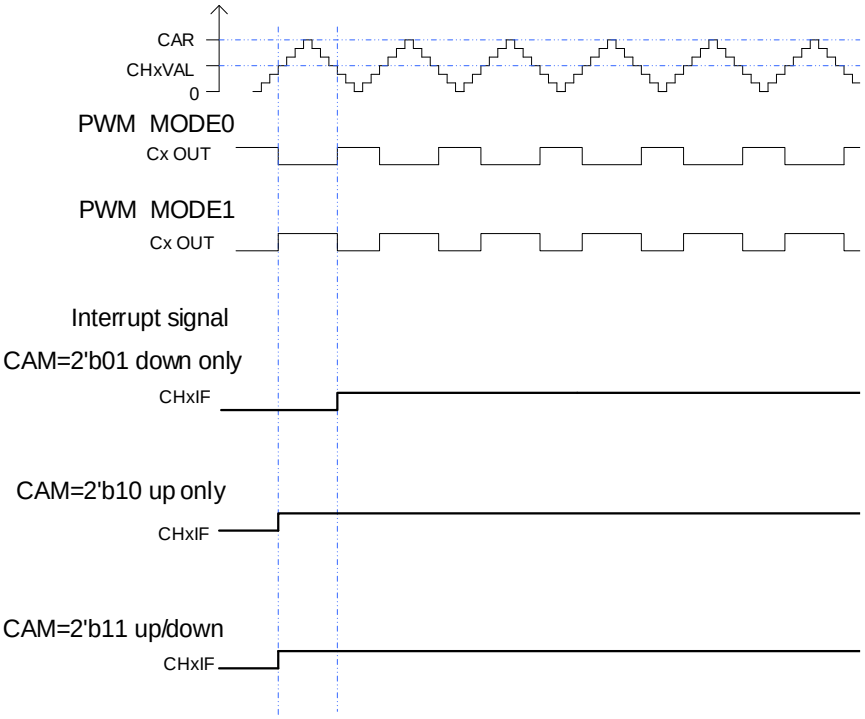


Figure 15-45. Timing chart of CAPWM



Channel output prepare signal

As is shown in ***Figure 15-42. Channel output compare principle (x=0,1,2,3)***, when the TIMERx is used in the compare match output mode, the OxCPRE signal (Channel x Output prepare signal) is defined by setting the CHxCOMCTL filed. The OxCPRE signal has several types of output function. These include, keeping the original level by setting the CHxCOMCTL field to 0x00, set to 1 by setting the CHxCOMCTL field to 0x01, set to 0 by setting the CHxCOMCTL field to 0x02 or signal toggle by setting the CHxCOMCTL field to 0x03 when the counter value matches the content of the TIMERx_CHxCV register.

The PWM mode 0 and PWM mode 1 outputs are also another kind of OxCPRE output which is setup by setting the CHxCOMCTL field to 0x06/0x07. In these modes, the OxCPRE signal level is changed according to the counting direction and the relationship between the counter value and the TIMERx_CHxCV content. With regard to a more detail description refer to the relative bit definition.

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

The OxCPRE signal can be forced to 0 when the ETIFE signal is derived from the external ETI pin and when it is set to a high level by setting the CHxCOMCEN bit to 1 in the TIMERx_CHCTL0 register. The OxCPRE signal will not return to its active level until the next update event occurs.

Quadrature decoder

Refer to ***Quadrature decoder***.

Hall sensor function

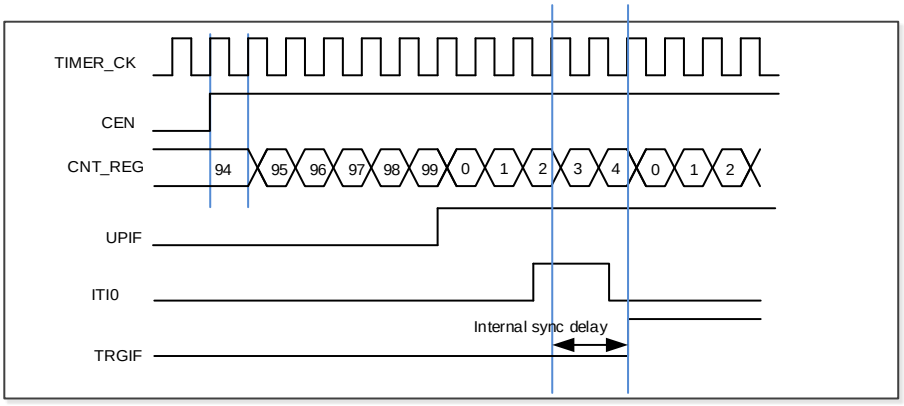
Refer to ***Hall sensor function***.

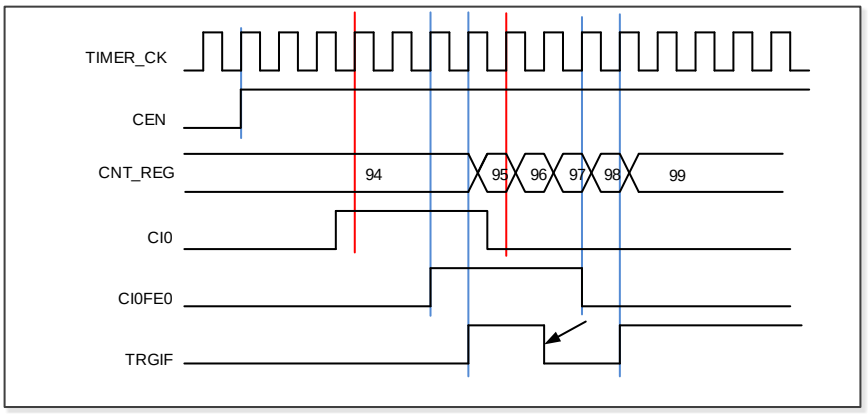
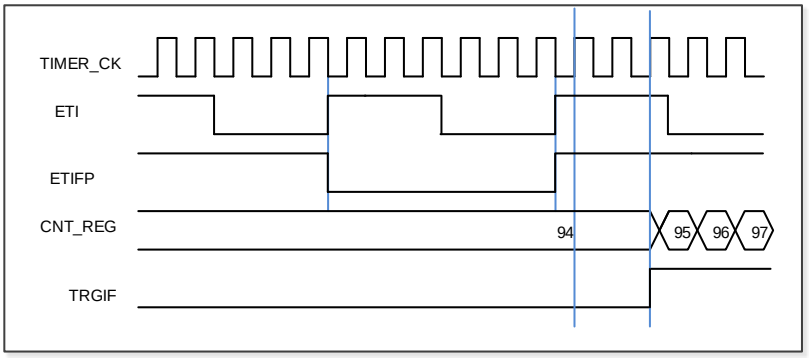
Master-slave management

The TIMERx can be synchronized with a trigger in several modes including restart mode, pause mode and event mode which is selected by the TSCFGy[3:0] in SYSCFG_TIMERxCFG(y=3,4 ,5).

Table 15-5. Examples of slave mode

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
LIST	TSCFGy[3:0] y=3 (restart mode) y=4 (pause mode) y=5 (event mode)	TSCFGy[3:0] 0001: ITI0 0010: ITI1 0011: ITI2 0100: ITI3	If CI0FE0 or CI1FE1 is selected as the trigger source, configure the CHxP and CHxNP for the polarity selection	For the ITIx, no filter and prescaler can be used. For the CIx, filter can be used by configuring

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
		0101: CI0F_ED 0110: CI0FE0 0111: CI1FE1 1000: ETIFP	and inversion. If ETIFP is selected as the trigger source, configure the ETP for polarity selection and inversion.	CHxCAPFLT, no prescaler can be used. For the ETIFP, filter can be used by configuring ETFC and prescaler can be used by configuring ETPSC.
Exam1	**Restart mode** The counter will be cleared and restart when a rising edge of trigger input comes.	TSCFG3[3:0] = 4'b 0001.ITI0 is selected.	For ITI0, no polarity selector can be used.	For the ITI0, no filter and prescaler can be used.
	Figure 15-46. Restart mode 			
Exam2	**Pause mode** The counter will be paused when the trigger input is low, and it will start when the trigger input is high.	TSCFG4[3:0] = 4'b 0110 CI0FE0 is selected.	TI0S=0 (Non-xor) [CH0NP=0, CH0P=0] CI0FE0 does not invert. The capture event will occur on the rising edge only.	Filter is bypassed in this example.

<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 15-47. 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>TSCFG5[3:0] = 4’b 1000
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 15-48. Event mode
 
</td></tr>
</table>

Single pulse mode

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

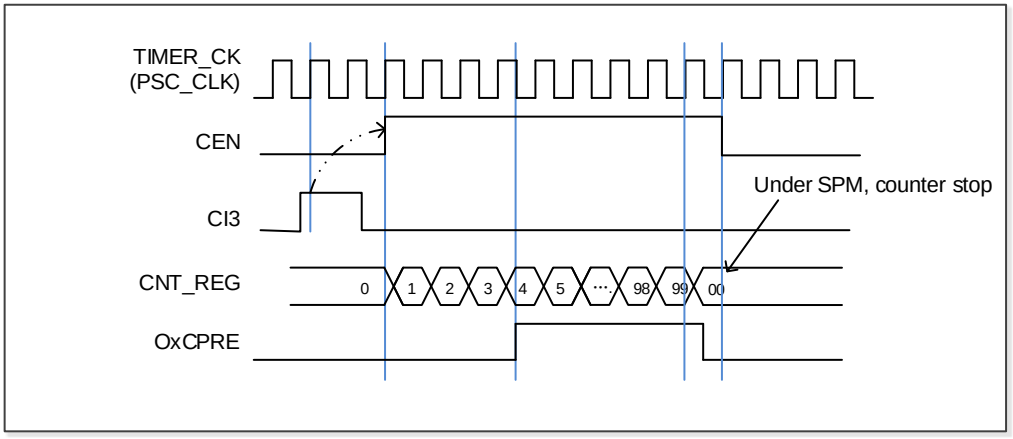
Once the timer is set to operate in the single pulse mode, it is not necessary to set the timer enable bit CEN in the TIMERx_CTL0 register to 1 to enable the counter. The trigger to generate a pulse can be sourced from the trigger signals edge or by setting the CEN bit to 1 using software. Setting the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

In the single pulse mode, the trigger active edge which sets the CEN bit to 1 will enable the

counter. However, there exist several clock delays to perform the comparison result between the counter value and the TIMERx_CHxCV value. In order to reduce the delay to a minimum value, the user can set the CHxCOMFEN bit in each TIMERx_CHCTL0/1 register. After a trigger rising occurs in the single pulse mode, the OxCPRE signal will immediately be forced to the state which the OxCPRE signal will change to, as the compare match event occurs without taking the comparison result into account. The CHxCOMFEN bit is available only when the output channel is configured to operate in the PWM0 or PWM1 output mode and the trigger source is derived from the trigger signal.

Figure 15-49. Single pulse mode, TIMERx_CHxCV = 4, TIMERx_CAR=99 shows an example.

Figure 15-49. Single pulse mode, TIMERx_CHxCV = 4, TIMERx_CAR=99



Timers interconnection

Refer to *Advanced timer (TIMERx, x=0)*.

Timer DMA mode

Timer's DMA mode is the function that configures timer's register by DMA module. The relative registers are TIMERx_DMACFG and TIMERx_DMATB; Of course, you have to enable a DMA request which will be asserted by some internal interrupt event. When the interrupt event was asserted, TIMERx will send a request to DMA, which is configured to M2P mode and PADDR is TIMERx_DMATB, then DMA will access the TIMERx_DMATB. In fact, register TIMERx_DMATB is only a buffer; timer will map the TIMERx_DMATB to an internal register, appointed by the field of DMATA in TIMERx_DMACFG . If the field of DMATC in TIMERx_DMACFG is 0(1 transfer), then the timer's DMA request is finished. While if TIMERx_DMATC is not 0, such as 3(4 transfers), then timer will send 3 more requests to DMA, and DMA will access timer's registers DMASAR+0x4, DMASAR+0x8, DMASAR+0xc at the next 3 accesses to TIMERx_DMATB. In one word, one time DMA internal interrupt event assert, DMATC+1 times request will be send by TIMERx.

If one more time DMA request event coming, TIMERx will repeat the process as above.

Timer debug mode

When the RISCV halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register set to 1, the TIMERx counter stops.

15.2.5. TIMERx registers(x=1, 2)

TIMER1 access base address: 0x4000 0000
TIMER2 access 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).

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKDIV[1:0]		ARSE	CAM[1:0]		DIR	SPM	UPS	UPDIS	CEN
						rw		rw	rw		rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	CKDIV[1:0]	Clock division The CKDIV bits can be configured by software to specify division factor between the CK_TIMER and the dead-time and digital filter sample clock (DTS). 00: $f_{DTS}=f_{CK_TIMER}$ 01: $f_{DTS}= f_{CK_TIMER}$ /2 10: $f_{DTS}= f_{CK_TIMER}$ /4 11: Reserved
7	ARSE	Auto-reload shadow enable 0: The shadow register for TIMERx_CAR register is disabled 1: The shadow register for TIMERx_CAR register is enabled
6:5	CAM[1:0]	Counter aligns mode selection 00: No center-aligned mode (edge-aligned mode). The direction of the counter is specified by the DIR bit. 01: Center-aligned and counting down assert mode. The counter counts under center-aligned and channel is configured in output mode (CHxMS=00 in TIMERx_CHCTL0 register). Only when counting down, CHxF bit can be set. 10: Center-aligned and counting up assert mode. The counter counts under center-aligned and channel is configured in output mode (CHxMS=00 in TIMERx_CHCTL0 register). Only when counting up, CHxF bit can be set. 11: Center-aligned and counting up/down assert mode. The counter counts under center-aligned and channel is configured in output mode (CHxMS=00 in TIMERx_CHCTL0 register). Both when counting up and counting down, CHxF bit

		can be set. After the counter is enabled, cannot be switched from 0x00 to non 0x00.
4	DIR	Direction 0: Count up 1: Count down If the timer work in center-aligned mode or encoder mode, this bit is read only.
3	SPM	Single pulse mode. 0: Single pulse mode disable. The counter continues after update event. 1: Single pulse mode enable. The counter counts until the next update event occurs.
2	UPS	Update source This bit is used to select the update event sources by software. 0: These events generate update interrupts or DMA requests: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: This event generates update interrupts or DMA requests: The counter generates an overflow or underflow event
1	UPDIS	Update disable. This bit is used to enable or disable the update event generation. 0: Update event enable. When an update event occurs, the corresponding shadow registers are loaded with their preloaded values. These events generate update event: The UPG bit is set The counter generates an overflow or underflow event The restart mode generates an update event. 1: Update event disable. **Note:** When this bit is set to 1, setting UPG bit or 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 encoder mode.

Control register 1 (TIMERx_CTL1)

Address offset: 0x04
Reset value: 0x0000 0000

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

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

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TI0S	MMC[2:0]			DMAS	Reserved		
								rw	rw			rw			

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	TI0S	Channel 0 trigger input selection 0: The TIMERx_CH0 pin input is selected as channel 0 trigger input. 1: The result of combinational XOR of TIMERx_CH0, 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: Reset. When the UPG bit in the TIMERx_SWEVG register is set or a reset is generated by the slave mode controller, a TRGO pulse occurs. And in the latter case, the signal on TRGO is delayed compared to the actual reset. 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.

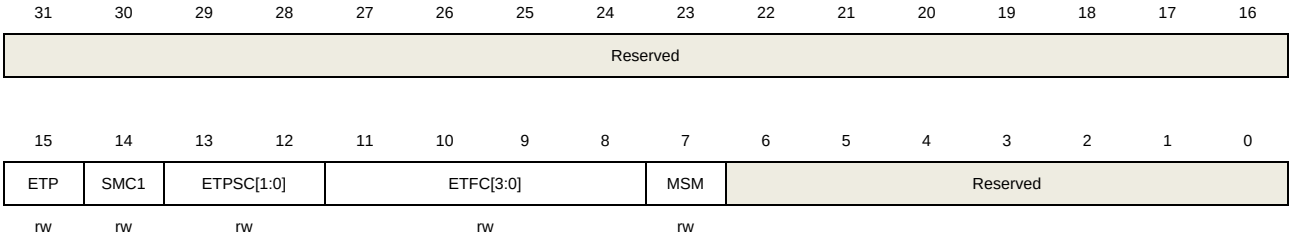
2:0	Reserved	Must be kept at reset value.

Slave mode configuration register (TIMERx_SMCFG)

Address offset: 0x08

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15	ETP	External trigger polarity This bit specifies the polarity of ETI signal 0: ETI is active at rising edge or high level . 1: ETI is active at falling edge or low level .
14	SMC1	Part of SMC for enable External clock mode1. In external clock mode 1, the counter is clocked by any active edge on the ETIFP signal. 0: External clock mode 1 disabled 1: External clock mode 1 enabled. When the slave mode is configured as restart mode, pause mode or event mode, the timer can still work in the external clock 1 mode by setting this bit. But the TSCFGy[3:0](y=3,4,5) bits must not be 1000 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 SYSCFG_TIMERxCFG register.
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

6:0 Reserved Must be kept at reset value.

DMA and interrupt enable register (TIMERx_DMAINTEN)

Address offset: 0x0C

Reset value: 0x0000 0000

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

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

Reserved	TRGDEN	Reserved	CH3DEN	CH2DEN	CH1DEN	CH0DEN	UPDEN	Reserved	TRGIE	Reserved	CH3IE	CH2IE	CH1IE	CH0IE	UPIE
	rw		rw	rw	rw	rw	rw		rw		rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14	TRGDEN	Trigger DMA request enable 0: disabled 1: enabled
13	Reserved	Must be kept at reset value.
12	CH3DEN	Channel 3 capture/compare DMA request enable 0: disabled 1: enabled
11	CH2DEN	Channel 2 capture/compare DMA request enable 0: disabled 1: enabled
10	CH1DEN	Channel 1 capture/compare DMA request enable 0: disabled 1: enabled
9	CH0DEN	Channel 0 capture/compare DMA request enable 0: disabled 1: enabled
8	UPDEN	Update DMA request enable 0: disabled 1: enabled
7	Reserved	Must be kept at reset value.
6	TRGIE	Trigger interrupt enable 0: disabled 1: enabled
5	Reserved	Must be kept at reset value.
4	CH3IE	Channel 3 capture/compare interrupt enable 0: disabled 1: enabled
3	CH2IE	Channel 2 capture/compare interrupt enable 0: disabled 1: enabled
2	CH1IE	Channel 1 capture/compare interrupt enable 0: disabled

		1: enabled
1	CH0IE	Channel 0 capture/compare interrupt enable 0: disabled 1: enabled
0	UPIE	Update interrupt enable 0: disabled 1: enabled

Interrupt flag register (TIMERx_INTF)

Address offset: 0x10
Reset value: 0x0000 0000

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CH3OF	CH2OF	CH1OF	CH0OF	Reserved		TRGIF	Reserved	CH3IF	CH2IF	CH1IF	CH0IF	UPIF
			rc_w0	rc_w0	rc_w0	rc_w0			rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12	CH3OF	Channel 3 over capture flag Refer to CH0OF description
11	CH2OF	Channel 2 over capture flag Refer to CH0OF description
10	CH1OF	Channel 1 over capture flag Refer to CH0OF description
9	CH0OF	Channel 0 over capture flag When channel 0 is configured in input mode, this flag is set by hardware when a capture event occurs while CH0IF flag has already been set. This flag is cleared by software. 0: No over capture interrupt occurred 1: Over capture interrupt occurred
8:7	Reserved	Must be kept at reset value.
6	TRGIF	Trigger interrupt flag This flag is set on trigger event and cleared by software. When in pause mode, both edges on trigger input generates a trigger event, otherwise, only an active edge on

		trigger input can generates a trigger event. 0: No trigger event occurred. 1: Trigger interrupt occurred.
5	Reserved	Must be kept at reset value.
4	CH3IF	Channel 3 ‘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 Channel 0 is set to input mode, this bit will be reset by reading TIMERx_CH0CV. 0: No Channel 0 interrupt occurred 1: Channel 0 interrupt occurred
0	UPIF	Update interrupt flag This bit is set by hardware 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 working only if an internal trigger input is selected, through TSCFG7[3:0] bit-field in SYSCFG_TIMERxCFG(x=1,2) register.
7	CH0COMCEN	Channel 0 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the O0CPRE signal will be cleared. 0: Channel 0 output compare clear disable 1: Channel 0 output compare clear enable
6:4	CH0COMCTL[2:0]	Channel 0 compare output control This bit-field specifies the compare output mode of the the output prepare signal O0CPRE. In addition, the high level of O0CPRE is the active level, and CH0_O and CH0_ON channels polarity depends on CH0P and CH0NP bits. 000: Timing mode. The O0CPRE signal keeps stable, independent of the comparison between the register TIMERx_CH0CV and the counter TIMERx_CNT. 001: Set the channel output. O0CPRE signal is forced high when the counter is equals to the output compare register TIMERx_CH0CV. 010: Clear the channel output. O0CPRE signal is forced low when the counter is equals to the output compare register TIMERx_CH0CV. 011: Toggle on match. O0CPRE toggles when the counter is equals to the output compare register TIMERx_CH0CV. 100: Force low. O0CPRE is forced to low level. 101: Force high. O0CPRE is forced to high level.

110: PWM mode0. When counting up, O0CPRE is high when the counter is smaller than TIMERx_CH0CV, and low otherwise. When counting down, O0CPRE is low when the counter is larger than TIMERx_CH0CV, and high otherwise.
111: PWM mode1. When counting up, O0CPRE is low when the counter is smaller than TIMERx_CH0CV, and high otherwise. When counting down, O0CPRE is high when the counter is larger than TIMERx_CH0CV, and low otherwise.
If configured in PWM mode, the O0CPRE level changes only when the output compare mode is adjusted from “Timing” mode to “PWM” mode or the comparison result changes.

3	CH0COMSEN	Channel 0 compare output shadow enable When this bit is set, the shadow register of TIMERx_CH0CV register, which updates at each update event, will be enabled. 0: Channel 0 output compare shadow disable 1: Channel 0 output compare shadow enable The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1)
2	CH0COMFEN	Channel 0 output compare fast enable When this bit is set, the effect of an event on the trigger in input on the capture/compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH0_O is set to the compare level independently from the result of the comparison. 0: Channel 0 output quickly compare disable. 1: Channel 0 output quickly compare enable.
1:0	CH0MS[1:0]	Channel 0 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH0EN bit in TIMERx_CHCTL2 register is reset).). 00: Channel 0 is programmed as output mode 01: Channel 0 is programmed as input mode, IS0 is connected to CI0FE0 10: Channel 0 is programmed as input mode, IS0 is connected to CI1FE0 11: Channel 0 is programmed as input mode, IS0 is connected to ITS **Note:** When CH0MS[1:0]=11, it is working only if an internal trigger input is selected, through TSCFG7[3:0] bit-field in SYSCFG_TIMERxCFG (x=1,2) register.

Input capture mode:

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	CH1CAPFLT[3:0]	Channel 1 input capture filter control Refer to CH0CAPFLT description
11:10	CH1CAPPSC[1:0]	Channel 1 input capture prescaler

Refer to CH0CAPPSC description

9:8 CH1MS[1:0] Channel 1 mode selection

Same as Output compare mode

7:4 CH0CAPFLT[3:0] Channel 0 input capture filter control

The CI0 input signal can be filtered by digital filter and this bit-field configure the filtering capability.

Basic principle of digital filter: continuously sample the CI0 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level.

The filtering capability configuration is as follows:

CH0CAPFLT [3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{CK_TIMER}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	
4'b0110	6	$f_{DTS}/4$
4'b0111	8	
4'b1000	6	$f_{DTS}/8$
4'b1001	8	
4'b1010	5	$f_{DTS}/16$
4'b1011	6	
4'b1100	8	
4'b1101	5	$f_{DTS}/32$
4'b1110	6	
4'b1111	8	

3:2 CH0CAPPSC[1:0] Channel 0 input capture prescaler

This bit-field specifies the factor of the prescaler on channel 0 input. The prescaler is reset when CH0EN bit in TIMERx_CHCTL2 register is clear.

00: Prescaler disable, input capture occurs on every channel input edge

01: The input capture occurs on every 2 channel input edges

10: The input capture occurs on every 4 channel input edges

11: The input capture occurs on every 8 channel input edges

1:0 CH0MS[1:0] Channel 0 mode selection

Same as Output compare mode

Channel control register 1 (TIMERx_CHCTL1)

Address offset: 0x1C

Reset value: 0x0000 0000

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3COMCEN	CH3COMCTL[2:0]			CH3COMSEN	CH3COMFEN	CH3MS[1:0]		CH2COMCEN	CH2COMCTL[2:0]			CH2COMSEN	CH2COMFEN	CH2MS[1:0]	
CH3CAPFLT[3:0]				CH3CAPPSC[1:0]				CH2CAPFLT[3:0]				CH2CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	CH3COMCEN	Channel 3 output compare clear enable Refer to CH0COMCEN description
14:12	CH3COMCTL[2:0]	Channel 3 compare output control Refer to CH0COMCTL description
11	CH3COMSEN	Channel 3 output compare shadow enable Refer to CH0COMSEN description
10	CH3COMFEN	Channel 3 output compare fast enable Refer to CH0COMFEN description
9:8	CH3MS[1:0]	Channel 3 mode selection This bit-field specifies the direction of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH3EN bit in TIMERx_CHCTL2 register is reset). 00: Channel 3 is programmed as output mode 01: Channel 3 is programmed as input mode, IS3 is connected to CI3FE3 10: Channel 3 is programmed as input mode, IS3 is connected to CI2FE3 11: Channel 3 is programmed as input mode, IS3 is connected to ITS. **Note:** When CH3MS[1:0]=11, it is working only if an internal trigger input is selected, through TSCFG7[3:0] bit-field in SYSCFG_TIMERxCFG (x=1,2) register.
7	CH2COMCEN	Channel 2 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the O2CPRE signal will be cleared. 0: Channel 2 output compare clear disable 1: Channel 2 output compare clear enable
6:4	CH2COMCTL[2:0]	Channel 2 compare output control This bit-field specifies the compare output mode of the the output prepare signal O0CPRE. In addition, the high level of O0CPRE is the active level, and CH0_O and CH0_ON channels polarity depends on CH0P and CH0NP bits.

		000: Timing mode. The O2CPRE signal keeps stable, independent of the comparison between the output compare register TIMERx_CH2CV and the counter TIMERx_CNT. 001: Set the channel output. O2CPRE signal is forced high when the counter is equals to the output compare register TIMERx_CH2CV. 010: Clear the channel output. O2CPRE signal is forced low when the counter is equals to the output compare register TIMERx_CH2CV. 011: Toggle on match. O2CPRE toggles when the counter is equals to the output compare register TIMERx_CH2CV. 100: Force low. O2CPRE is forced to low level. 101: Force high. O2CPRE is forced to high level. 110: PWM mode 0. When counting up, O2CPRE is high when the counter is smaller than TIMERx_CH2CV, and low otherwise. When counting down, O2CPRE is low when the counter is larger than TIMERx_CH2CV, and high otherwise. 111: PWM mode 1. When counting up, O2CPRE is low when the counter is smaller than TIMERx_CH2CV, and high otherwise. When counting down, O2CPRE is high when the counter is larger than TIMERx_CH2CV, and low otherwise. If configured in PWM mode, the O2CPRE level changes only when the output compare mode is adjusted from “Timing” mode to “PWM” mode or the comparison result changes.
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 When this bit is set, the effect of an event on the trigger in input on the capture/compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH2_O is set to the compare level independently from the result of the comparison. 0: Channel 2 output quickly compare disable. 1: Channel 2 output quickly compare enable.
1:0	CH2MS[1:0]	Channel 2 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active. (CH2EN bit in TIMERx_CHCTL2 register is reset).). 00: Channel 2 is programmed as output mode 01: Channel 2 is programmed as input mode, IS2 is connected to CI2FE2 10: Channel 2 is programmed as input mode, IS2 is connected to CI3FE2

11: Channel 2 is programmed as input mode, IS2 is connected to ITS.

Note: When CH2MS[1:0]=11, it is working only if an internal trigger input is selected, through TSCFG7[3:0] bit-field in SYSCFG_TIMERxCFG (x=1,2) 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=1,2)

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
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.

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=1,2)

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
CARL[31:16]															

rw

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

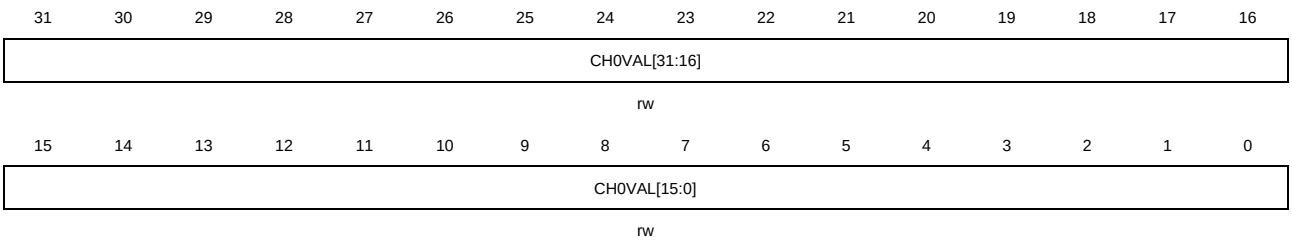
rw

Bits	Fields	Descriptions
31:0	CARL[31:0]	Counter auto reload value This bit-filed specifies the auto reload value of the counter.

Channel 0 capture/compare value register (TIMERx_CH0CV) (x=1,2)

Address offset: 0x34
Reset value: 0x0000 0000

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

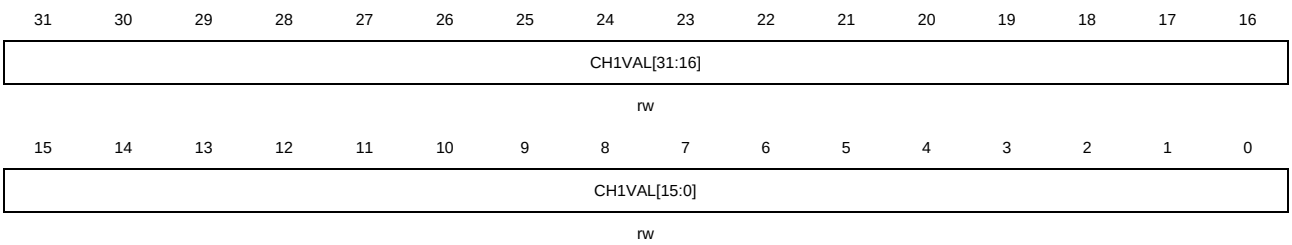


Bits	Fields	Descriptions
31:0	CH0VAL[31:0]	Capture or compare value of channel0 When channel 0 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 0 is configured in output mode, this bit-filed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

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

Address offset: 0x38
Reset value: 0x0000 0000

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



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

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
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 3 capture/compare value register (TIMERx_CH3CV) (x=1,2)

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
CH3VAL[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3VAL[15:0]															
rw															

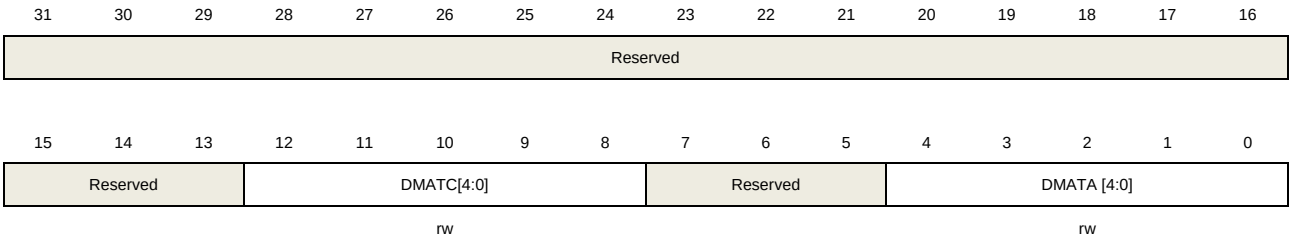
Bits	Fields	Descriptions
31:0	CH3VAL[31:0]	Capture or compare value of channel 3 When channel3 is configured in input mode, this bit-filed indicates the counter value corresponding to the last capture event. And this bit-filed is read-only. When channel 3 is configured in output mode, this bit-filed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

shadow register updates every update event.

DMA configuration register (TIMERx_DMACFG)

Address offset: 0x48
Reset value: 0x0000 0000

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

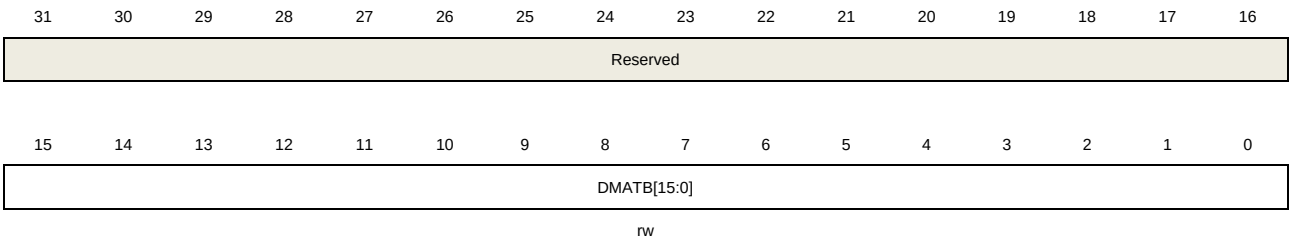


Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:8	DMATC [4:0]	DMA transfer count This filed defines the number(n) of the register that DMA will access(R/W), n = (DMATC [4:0] +1). DMATC [4:0] is from 5'b0_0000 to 5'b1_0001.
7:5	Reserved	Must be kept at reset value.
4:0	DMATA [4:0]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

Channel input remap register (TIMERx_IRMP)(x=2)

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				CI3_RMP[1:0]		Reserved									
				rw											

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value
11:10	CI3_RMP[1:0]	Channel 3 input remap 00: Channel 3 input is connected to GPIO(TIMER2_CH3) 01: Channel 3 input is connected to IRC32K 10: Channel 3 input is connected to LXTAL 11: Channel 3 input is connected to CKOUT
9:0	Reserved	Must be kept at reset value

Configuration register (TIMERx_CFG)

Address offset: 0xFC
Reset value: 0x0000 0000

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

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

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

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	CHVSEL	Write CHxVAL register selection This bit-field set and reset by software. 1: If write the CHxVAL register, the write value is same as the CHxVAL value, the write access ignored 0: No effect
0	Reserved	Must be kept at reset value.

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

15.3.1. 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.

15.3.2. Characteristics

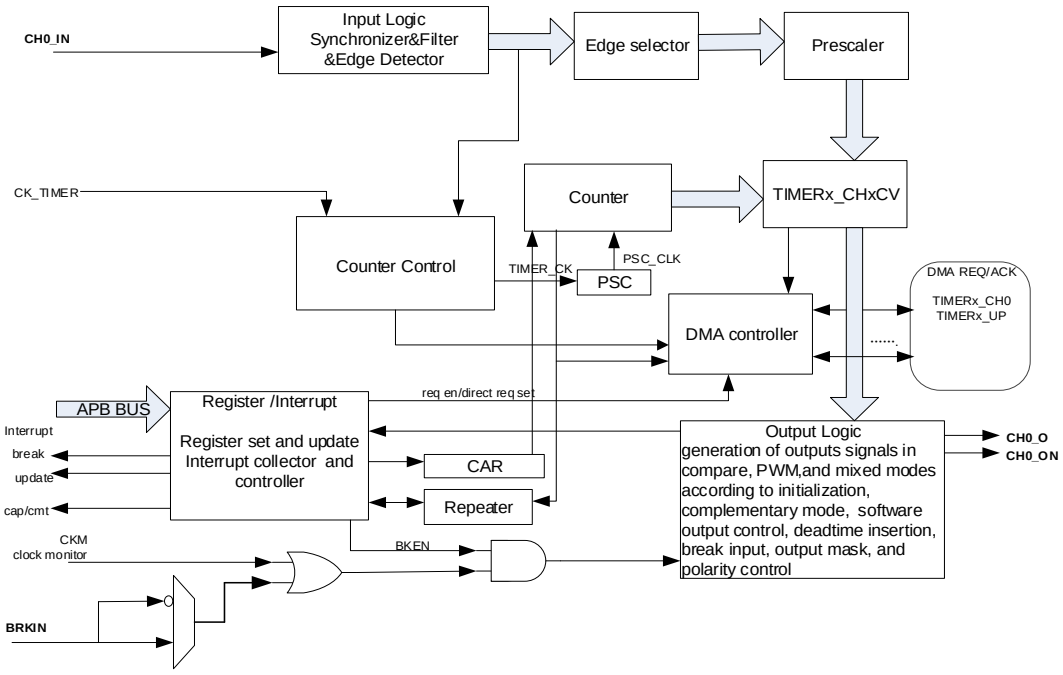
- Total channel num: 1.
- Counter width: 16 bits.
- Source of counter clock: internal clock.
- Counter modes: count up only.
- Programmable prescaler: 16 bit.The factor can be changed on the go.
- Each channel is user-configurable:
  input capture mode, output compare mode, programmable PWM mode, single pulse mode
- Programmable dead time insertion.
- Auto reload function.
- Programmable counter repetition function.
- Break input.
- Interrupt output or DMA request on: update, compare/capture event, and break input.

15.3.3. Block diagram

Figure 15-50. General level4 timer block diagram provides details of the internal configuration

of the general level4 timer.

Figure 15-50. General level4 timer block diagram



15.3.4. Function overview

Clock source configuration

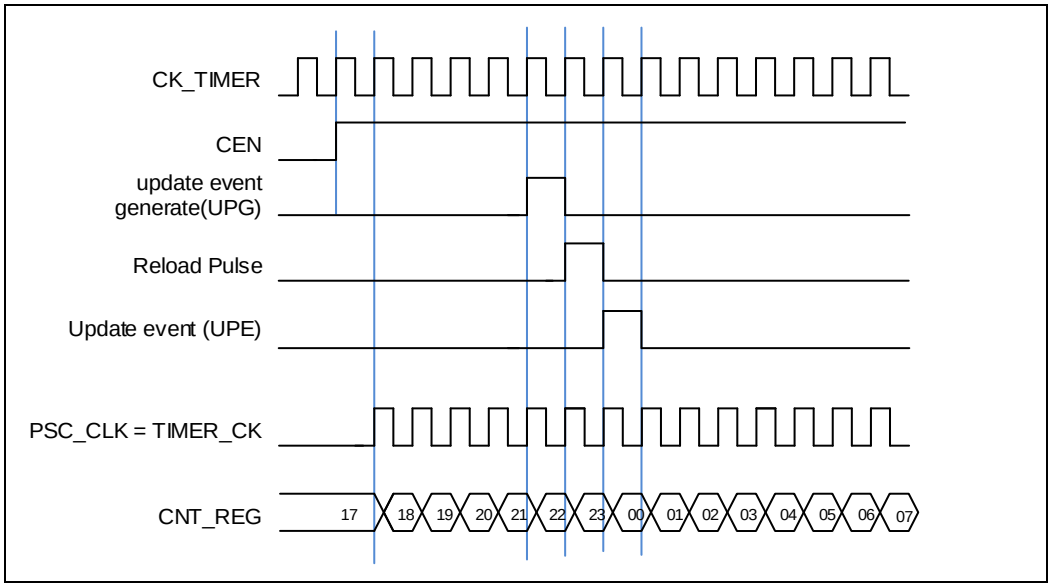
The general level4 TIMER can only being clocked by the CK_TIMER.

- 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

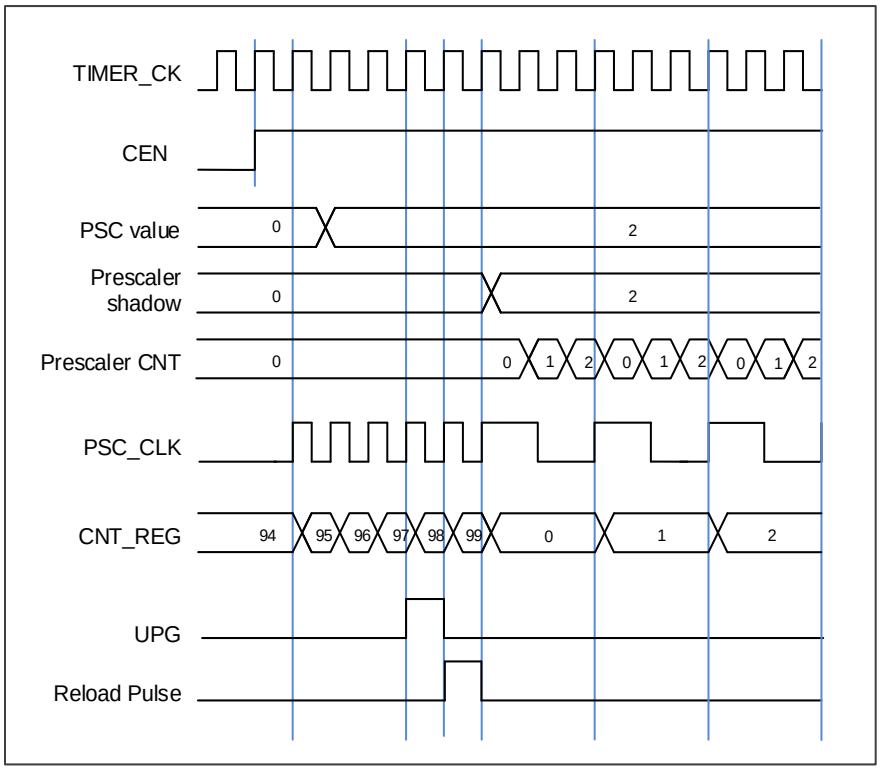
Figure 15-51. Timing chart of internal clock divided by 1



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 15-52. 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.

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 15-53. Up-counter timechart, PSC=0/2 show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 15-53. Up-counter timechart, PSC=0/2

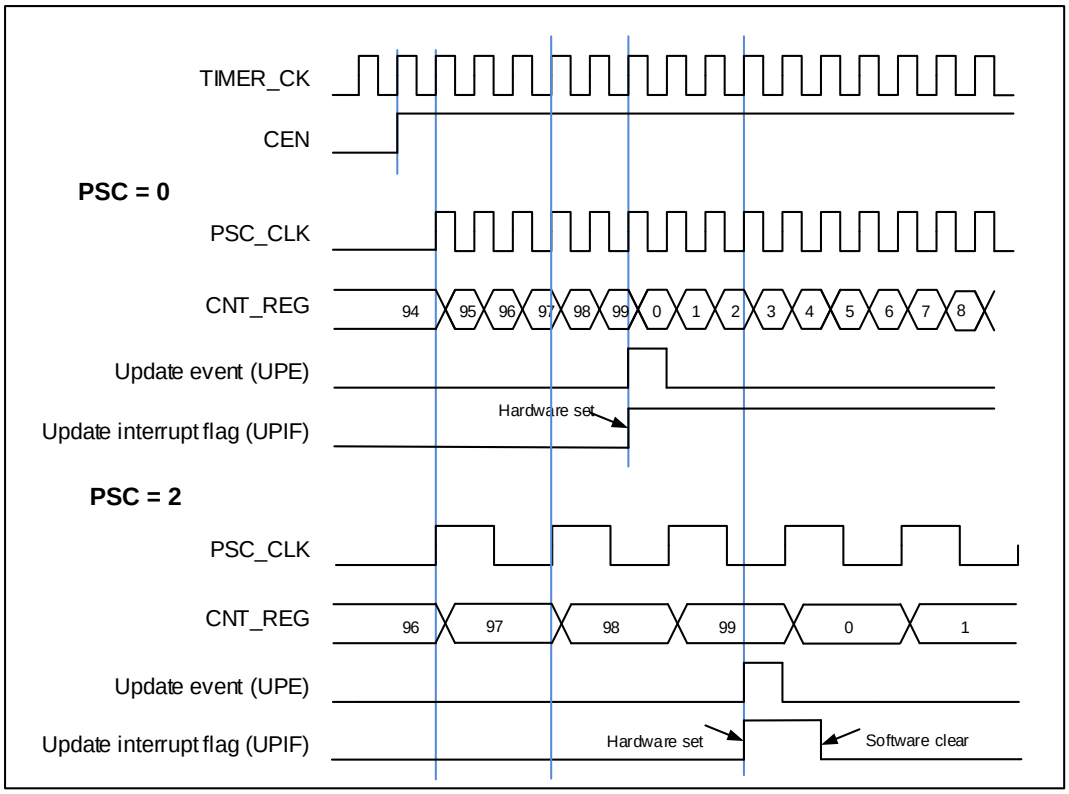
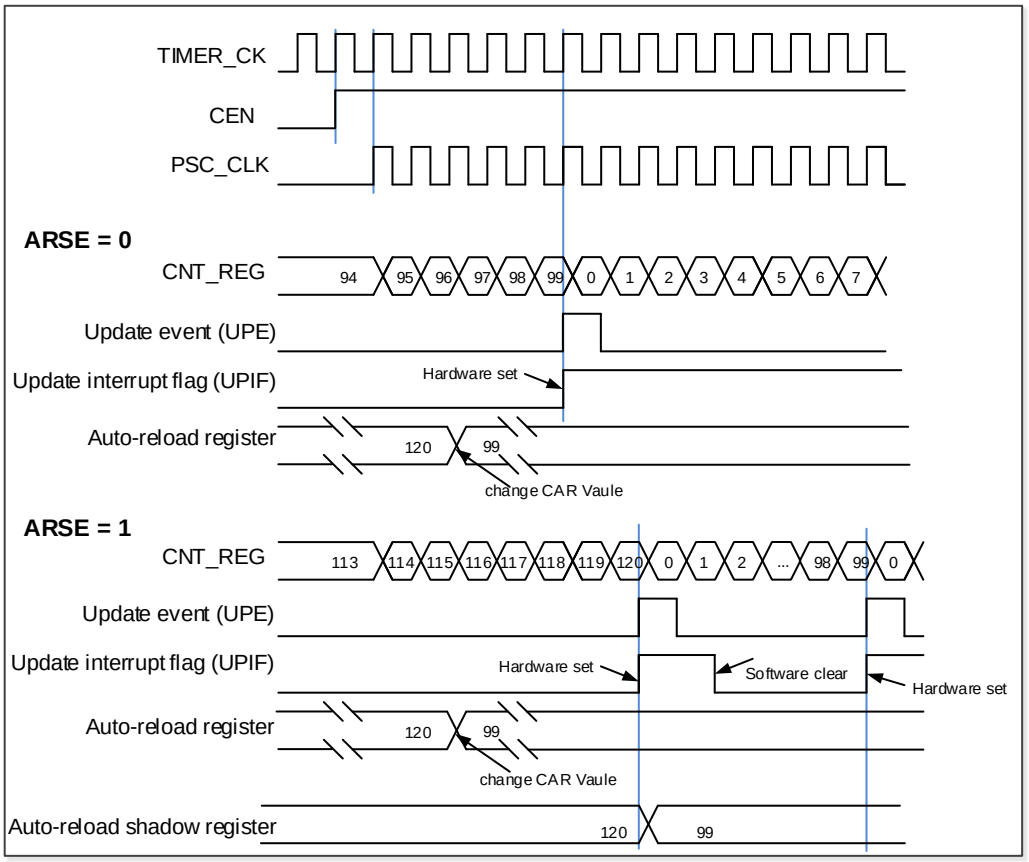


Figure 15-54. Up-counter timechart, change TIMERx_CAR on the go

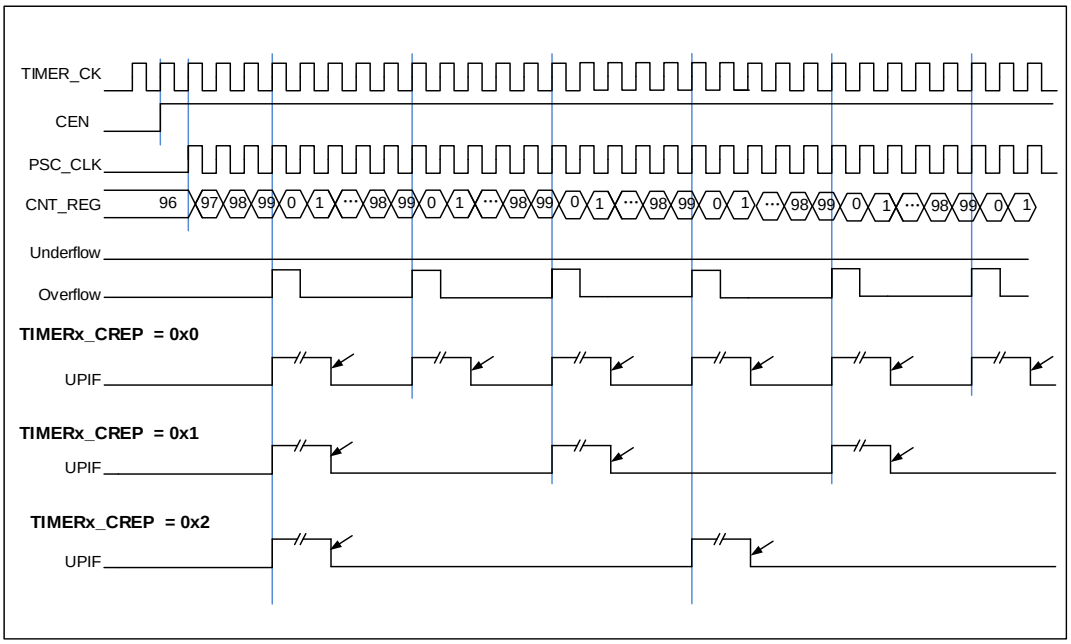


Update event (from overflow/underflow) rate configuration

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 15-55. Repetition timechart for up-counter



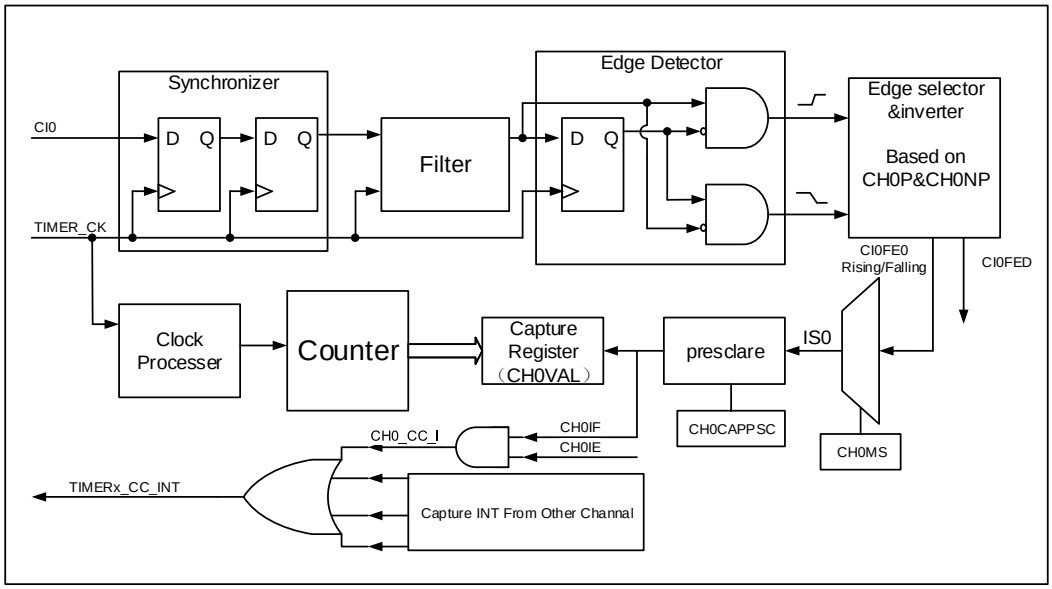
Input capture and output compare channels

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, duty cycle and so on. The input stage consists of a digital filter, a channel polarity selection, edge detection and a channel prescaler. When a selected edge occurs on the channel input, the current value of the counter is captured into the TIMERx_CHxCV register, at the same time the CHxIF bit is set and the channel interrupt is generated if enabled by CHxIE = 1.

Figure 15-56. 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.

Channel output compare function

Figure 15-57. Channel output compare principle (with complementary output, x=0)

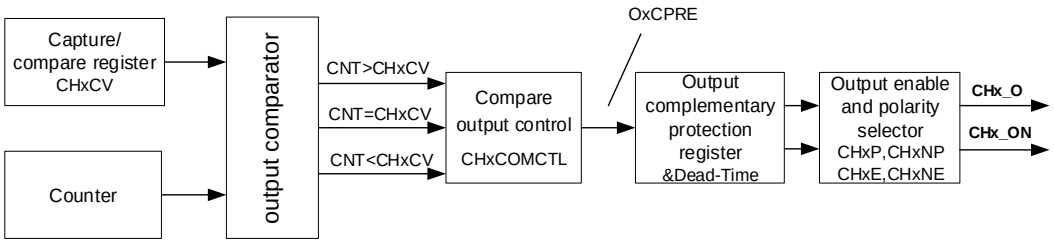


Figure 15-57. Channel output compare principle (with complementary output, x=0) show the principle circuit of channels output compare function. The relationship between the channel output signal CHx_O/CHx_ON and the OxCPRE signal (more details refer to *Channel output prepare signal*) is described as 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 *Outputs complementary* 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
* Set the output mode (Set/Clear/Toggle) by CHxCOMCTL.
* Select the active high polarity by CHxP/CHxNP

* Enable the output by CHxEN

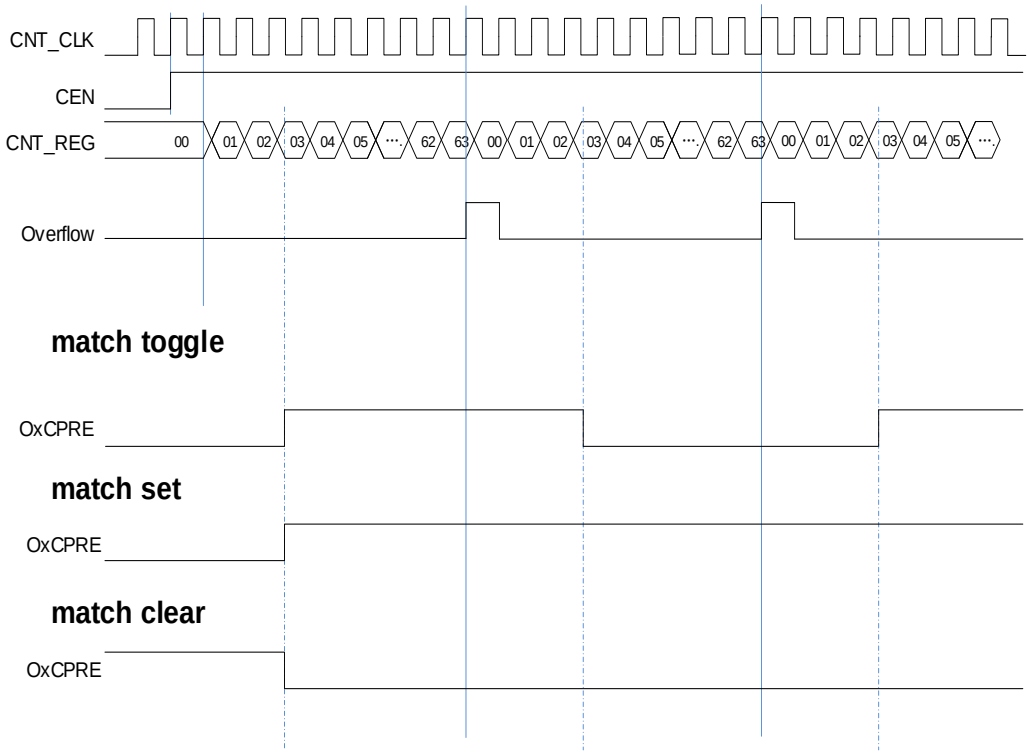
Step3: Interrupt/DMA-request enables configuration by CHxIE/CHxDEN

Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV
About the CHxVAL; you can change it on the go to meet the waveform you expected.

Step5: Start the counter by CEN.

Figure 15-58. Output-compare under three modes show the three compare modes toggle/set/clear. CAR=0x63, CHxVAL=0x3

Figure 15-58. Output-compare under three modes



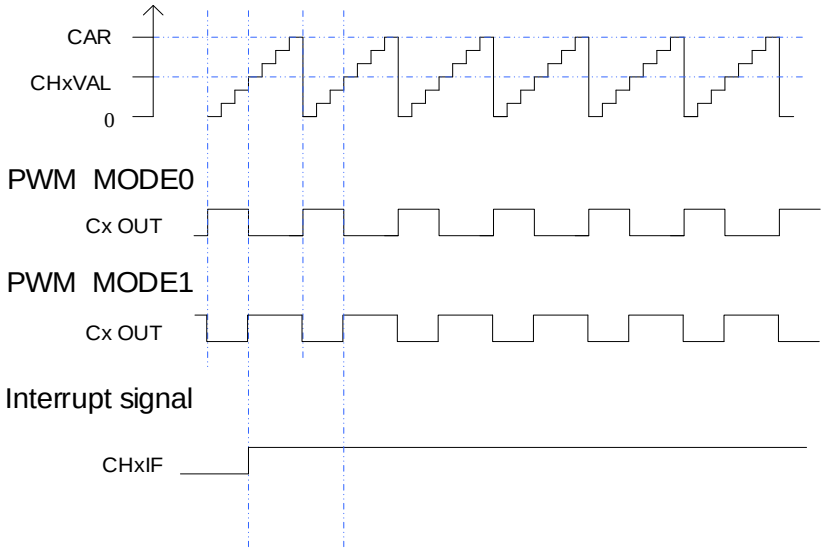
Output PWM function

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

The period is determined by TIMERx_CAR and duty cycle is determined by TIMERx_CHxCV. ***Figure 15-59. 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 15-59. PWM mode timechart



Channel output prepare signal

As is shown in ***Figure 15-57. Channel output compare principle (with complementary output, x=0).***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.

Outputs complementary

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 15-6. 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 / CHx_ON = LOW CHx_O / CHx_ON output disable(1).	
				1	CHx_O/ CHx_ON output “off-state” (2): the CHx_O/ CHx_ON output inactive level firstly: CHx_O = CHxP, CHx_ON = CHxNP; If the clock for deadtime generator is present, after a deadtime: CHx_O = ISOx, CHx_ON = ISOxN. (3)	
			1	0		
				1		
		1	x	x	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	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.

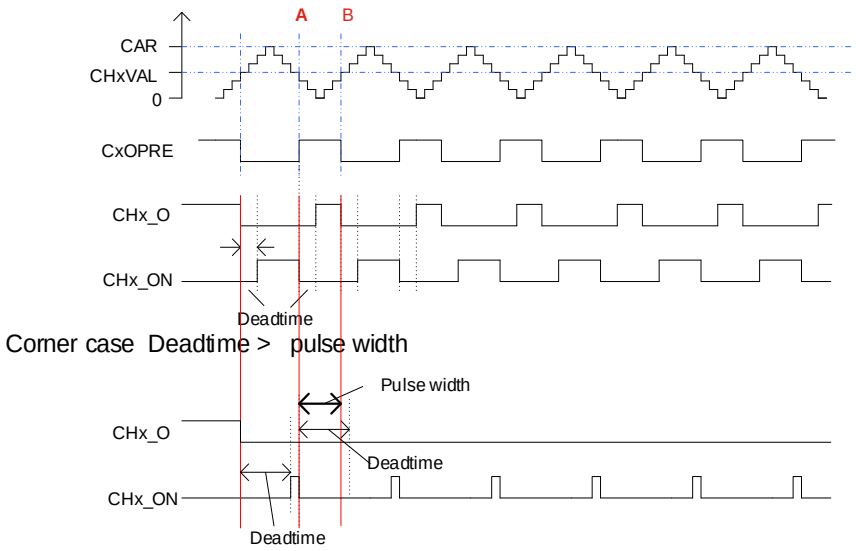
When the channel (x) match (TIMERx counter = CHxVAL) occurs, OxCPRE will be toggled because under PWM0 mode. At point A in the ***Figure 15-60. Complementary output with dead-time insertion.*** CHx_O signal remains at the low value until the end of the deadtime delay, while CHx_ON will be cleared at 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 15-60. 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 15-60. Complementary output with dead-time insertion.



Break mode

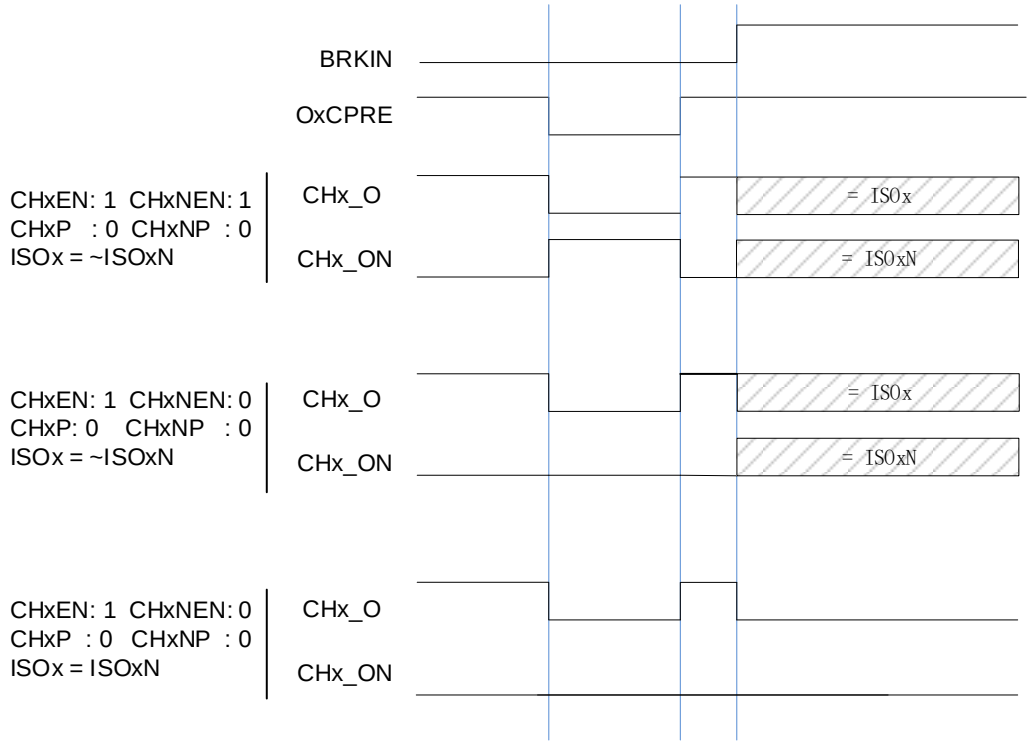
In this mode, the output CHx_O and CHx_ON are controlled by the POEN, IOS and ROS bits in the TIMERx_CCHP register, ISOx and ISOxN bits in the TIMERx_CTL1 register and cannot be set both to active level when break occurs. The break sources are input break pin and

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 15-61. 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.

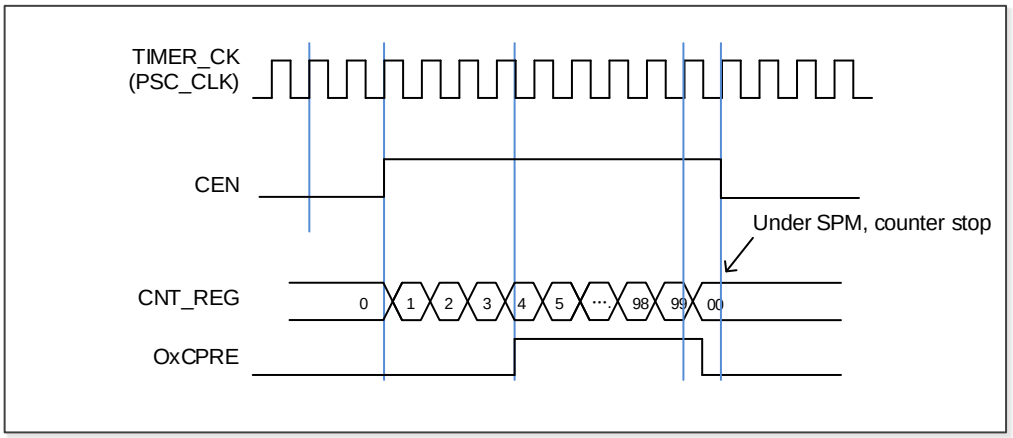
Once the timer is set to operate in the single pulse mode, it is necessary to set the timer enable bit CEN in the TIMERx_CTL0 register to 1 to enable the counter. Setting the CEN bit to 1 can generate a pulse and then keep the CEN bit at a high state until the update event occurs or the CEN bit is written 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 15-62. Single pulse mode TIMERx_CHxCV = 0x04 TIMERx_CAR=0x shows an example.

Figure 15-62. Single pulse mode TIMERx_CHxCV = 0x04 TIMERx_CAR=0x99



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

15.3.5. TIMERx registers(x=15, 16)

TIMER15 access base address: 0x4001 8000

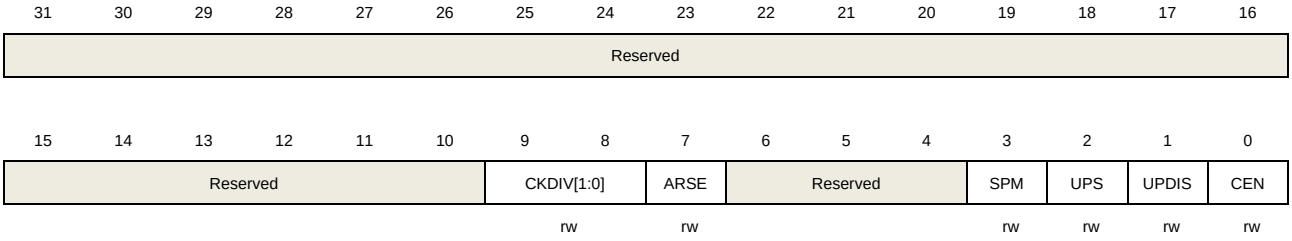
TIMER16 access base address: 0x4001 8400

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 and the CEN bit will be reset.
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 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 encoder mode.

Control register 1 (TIMERx_CTL1)

Address offset: 0x04
Reset value: 0x0000 0000

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

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

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

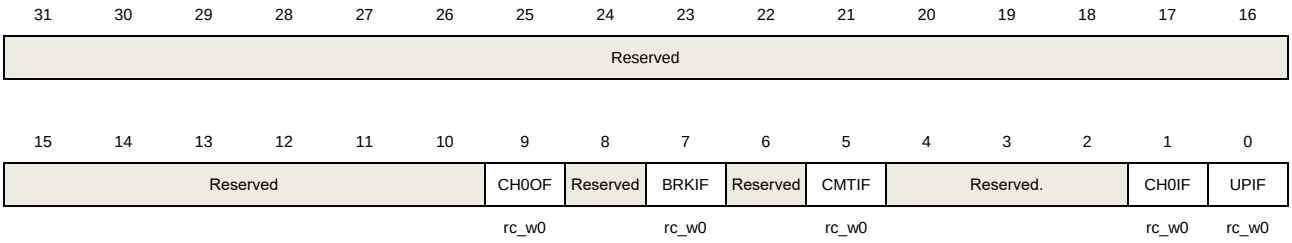
Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value
9	CH0DEN	Channel 0 capture/compare DMA request enable 0: disabled 1: enabled

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)



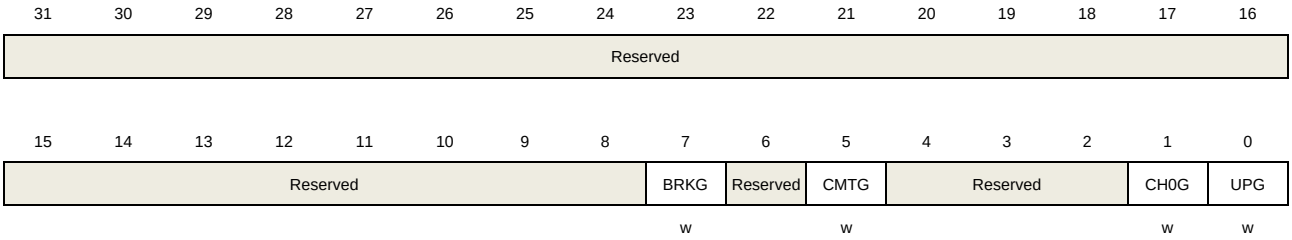
Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value
9	CH0OF	Channel 0 over capture flag When channel 0 is configured in input mode, this flag is set by hardware when a capture event occurs while CH0IF flag has already been set. This flag is cleared by software. 0: No over capture interrupt 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 Channel 0 is set to input mode, this bit will be reset by reading TIMERx_CH0CV. 0: No Channel 0 interrupt occurred 1: Channel 0 interrupt occurred
0	UPIF	Update interrupt flag This bit is set by hardware 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 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								Reserved	CH0COMCTL[2:0]			CH0COMSEN	CH0COMFEN	CH0MS[1:0]	
								CH0CAPFLT[3:0]				CH0CAPPSC[1:0]			
									rw			rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value
6:4	CH0COMCTL[2:0]	Channel 0 compare output control This bit-field specifies the compare output mode of the the output prepare signal O0CPRE. In addition, the high level of O0CPRE is the active level, and CH0_O and CH0_ON channels polarity depends on CH0P and CH0NP bits. 000: Timing mode. The O0CPRE signal keeps stable, independent of the comparison between the register TIMERx_CH0CV and the counter TIMERx_CNT. 001: Set the channel output. O0CPRE signal is forced high when the counter is equals to the output compare register TIMERx_CH0CV. 010: Clear the channel output. O0CPRE signal is forced low when the counter is equals to the output compare register TIMERx_CH0CV. 011: Toggle on match. O0CPRE toggles when the counter is equals to the output compare register TIMERx_CH0CV. 100: Force low. O0CPRE is forced to low level. 101: Force high. O0CPRE is forced to high level. 110: PWM mode0. When counting up, O0CPRE is high when the counter is smaller than TIMERx_CH0CV, and low otherwise. When counting down, O0CPRE is low when the counter is larger than TIMERx_CH0CV, and high otherwise. 111: PWM mode1. When counting up, O0CPRE is low when the counter is smaller than TIMERx_CH0CV, and high otherwise. When counting down, O0CPRE is high when the counter is larger than TIMERx_CH0CV, and low otherwise. If configured in PWM mode, the O0CPRE level changes only when the output compare mode is adjusted from “Timing” mode to “PWM” mode or the comparison result changes. 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 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)

31	30	29	28	27	26	25	24	23	22	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

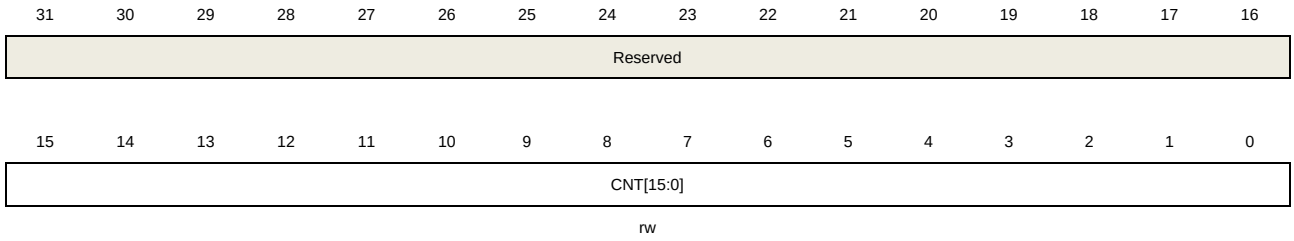
Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value
3	CH0NP	Channel 0 complementary output polarity When channel 0 is configured in output mode, this bit specifies the complementary output signal polarity. 0: Channel 0 complementary output high level is active level 1: Channel 0 complementary 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)



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)

31	30	29	28	27	26	25	24	23	22	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. 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 15-6. 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 15-6. 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:

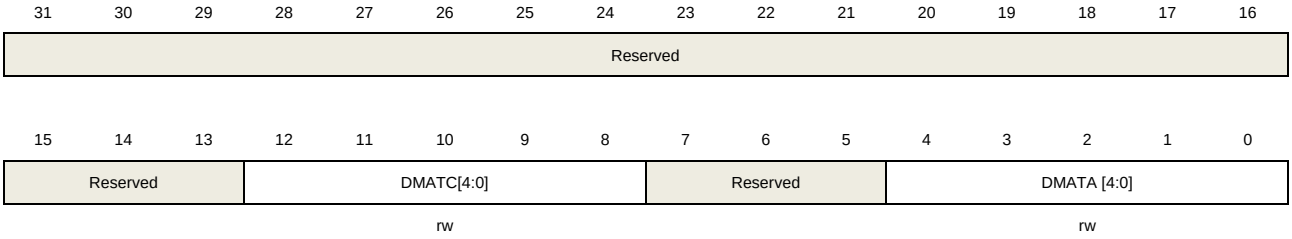
1. t_{DTS_CK} is the period of DTS_CK which is configured by CKDIV[1:0] in TIMERx_CTL0.
2. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.

DMA configuration register (TIMERx_DMACFG)

Address offset: 0x48

Reset value: 0x0000 0000

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



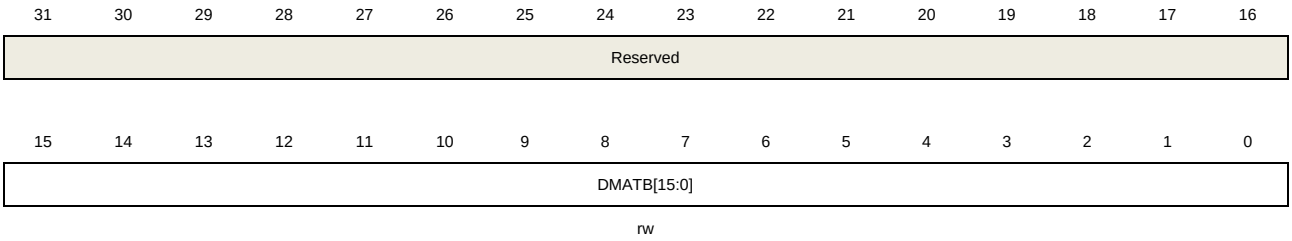
Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
12:8	DMATC [4:0]	DMA transfer count This filed defines the number(n) of the register that DMA will access(R/W), n = (DMATC [4:0] +1). DMATC [4:0] is from 5'b0_0000 to 5'b1_0001.
7:5	Reserved	Must be kept at reset value.
4:0	DMATA [4:0]	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)



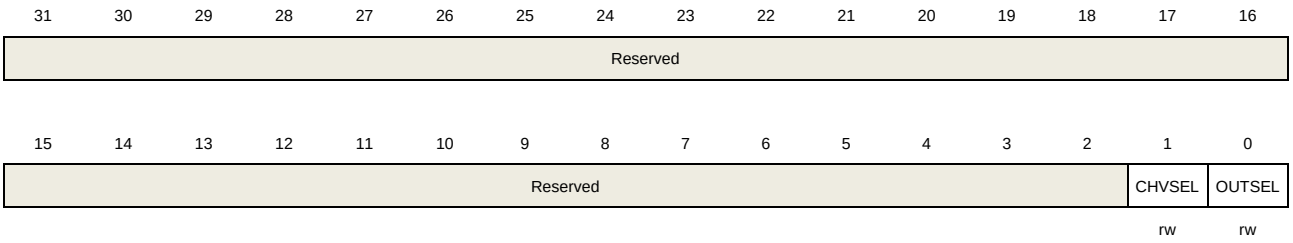
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	DMATB[15:0]	DMA transfer buffer When a read or write operation is assigned to this register, the register located at the address range (Start Addr + Transfer Timer* 4) will be accessed. The transfer Timer is calculated by hardware, 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. 1: If write the CHxVAL register, the write value is same as the CHxVAL value, the write access ignored 0: No effect
0	OUTSEL	The output value selection This bit-field set and reset by software 1: If POEN and IOS is 0, the output disabled 0: No effect

15.4. Basic timer (TIMERx, x=5)

15.4.1. Overview

The basic timer module (Timer5) reference is a 16-bit counter that can be used as an unsigned counter. The basic timer can be configured to generate DMA request.

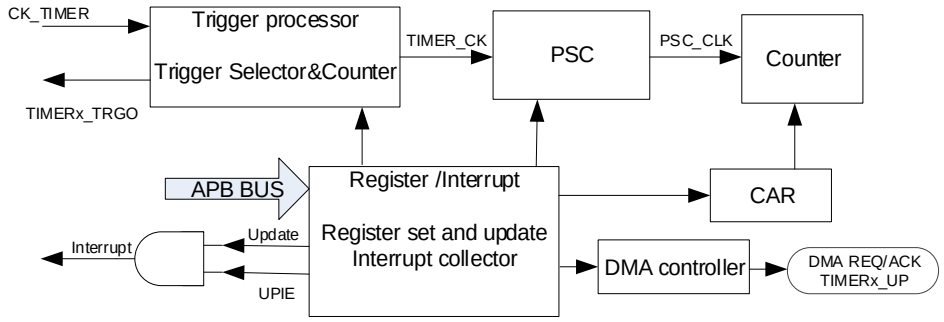
15.4.2. Characteristics

- Counter width: 16bit.
- Source of count clock is internal clock only.
- Multiple counter modes: count up.
- Programmable prescaler: 16 bit. Factor can be changed on the go.
- Auto-reload function.
- Interrupt output or DMA request on update event.

15.4.3. Block diagram

Figure 15-63. Basic timer block diagram provides details on the internal configuration of the basic timer.

Figure 15-63. Basic timer block diagram



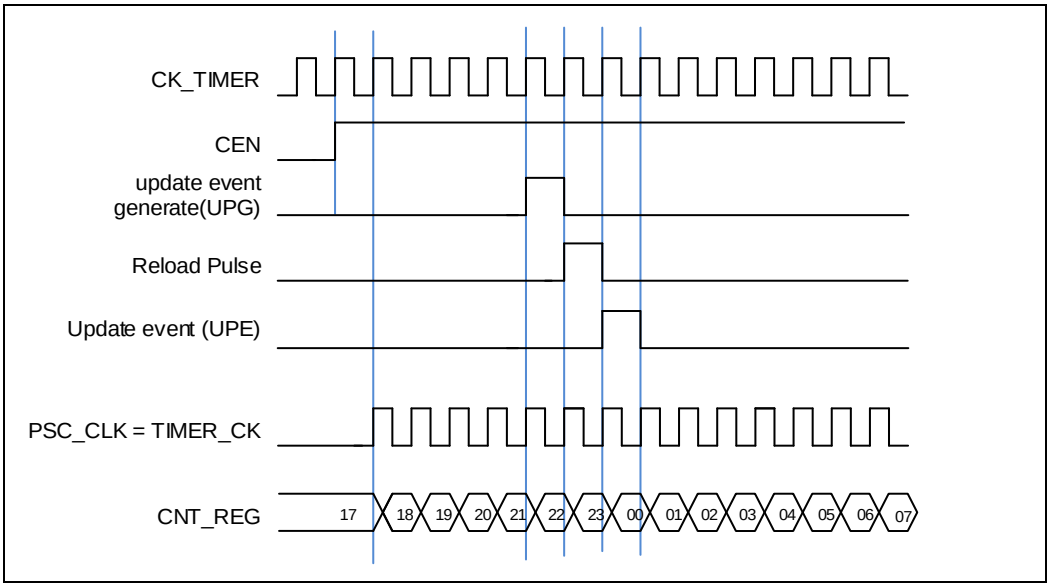
15.4.4. Function overview

Clock source configuration

The basic TIMER can only being 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.

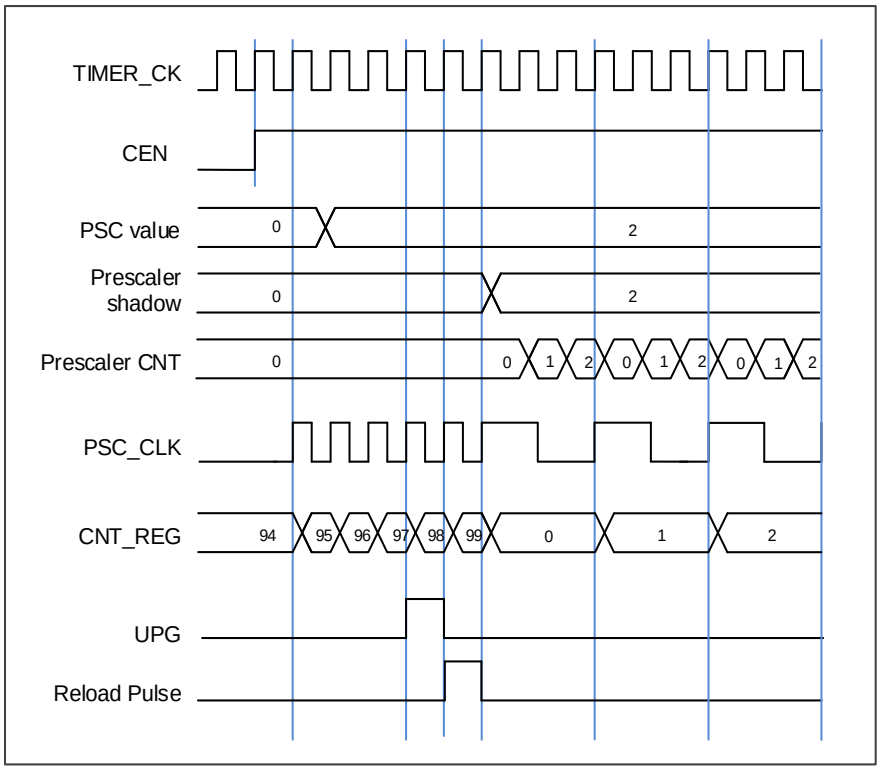
Figure 15-64. Timing chart of internal clock divided by 1



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 15-65. Timing chart of PSC value change from 0 to 2



Counter up counting

In this mode, the counter counts up continuously from 0 to the counter-reload value, which is defined in the TIMERx_CAR register, in a count-up direction. Once the counter reaches the counter reload value, the counter will start counting up from 0 again.The update event is generated at each counter overflow.

When the update event is set by the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to 0 and generates an update event.

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

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

The following figures show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 15-66. Timing chart of up counting mode, PSC=0/2

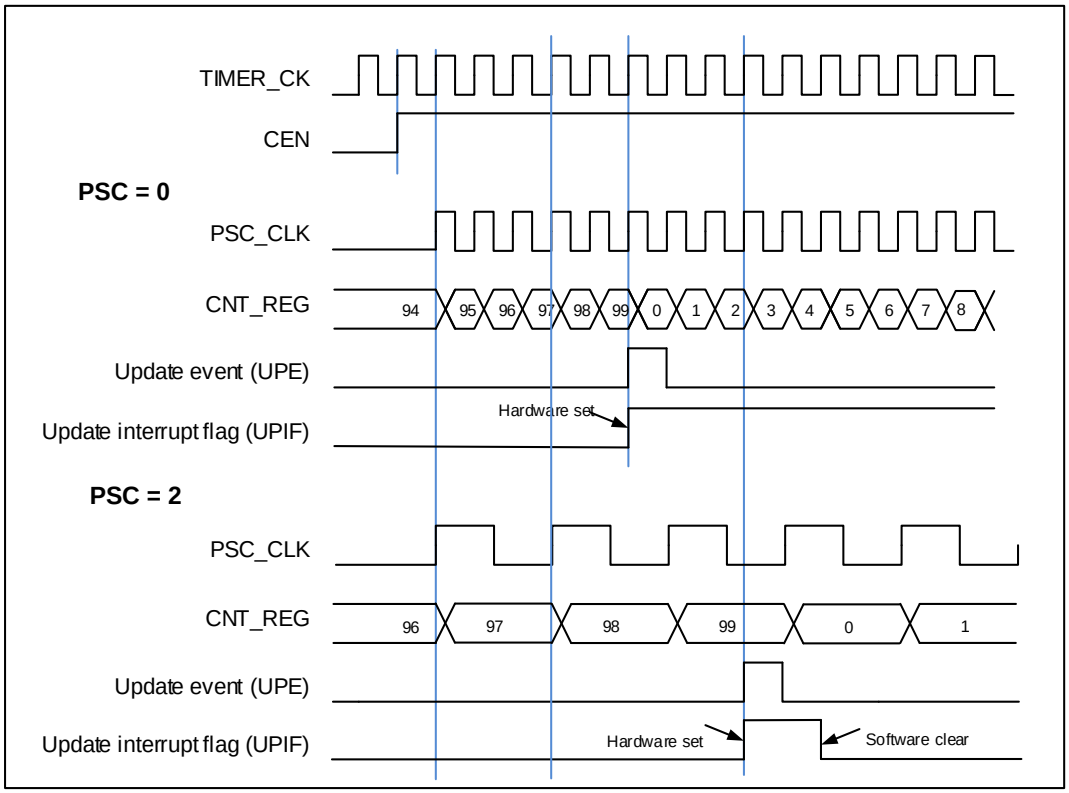
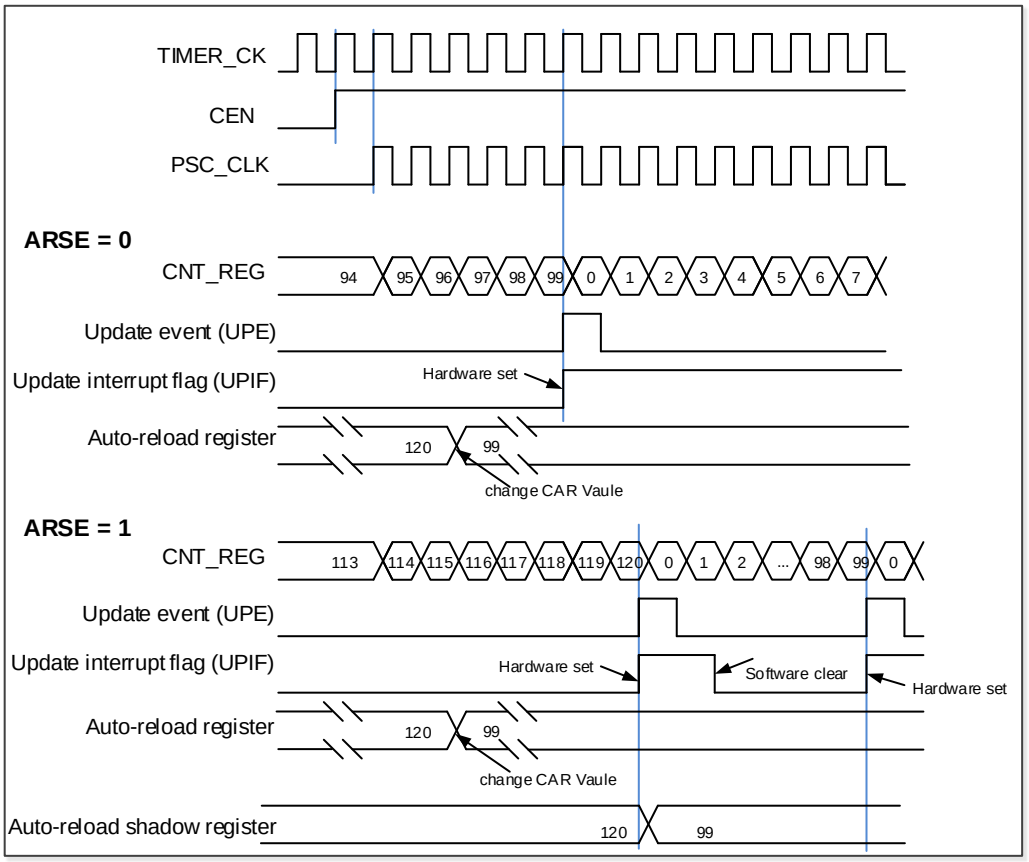


Figure 15-67. Timing chart of up counting mode, change TIMERx_CAR ongoing



Single pulse mode

Single pulse mode is opposite to the repetitive mode, which can be enabled by setting SPM in TIMERx_CTL0. When you set SPM, the counter will be clear and stop when the next update event.

Once the timer is set to operate in the single pulse mode, it is necessary to set the timer enable bit CEN in the TIMERx_CTL0 register to 1 to enable the counter, then the CEN bit keeps at a high state until the update event occurs or the CEN bit is written to 0 by software. If the CEN bit is cleared to 0 using software, the counter will be stopped and its value held.

Timer debug mode

When the RISCV halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register set to 1, the TIMERx counter stops.

15.4.5. TIMERx registers(x=5)

TIMER5 access base address: 0x4000 1000

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								ARSE	Reserved			SPM	UPS	UPDIS	CEN
								rw				rw	rw	rw	rw

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						

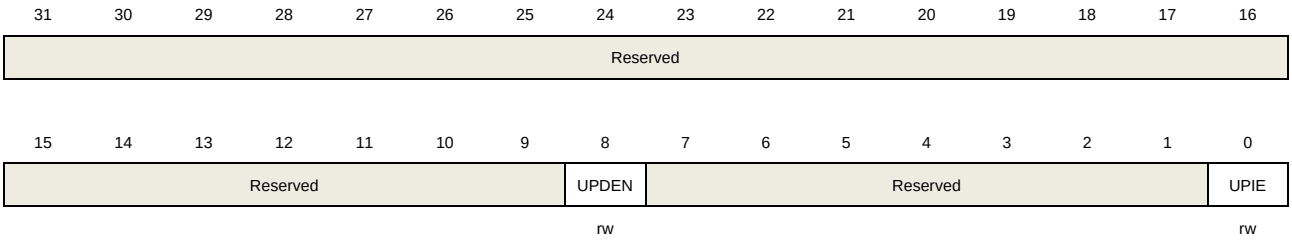
Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6:4	MMC[2:0]	Master mode control These bits control the selection of TRGO signal, which is sent in master mode to slave timers for synchronization function. 000: When a counter reset event occurs, a TRGO trigger signal is output. The 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.
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).

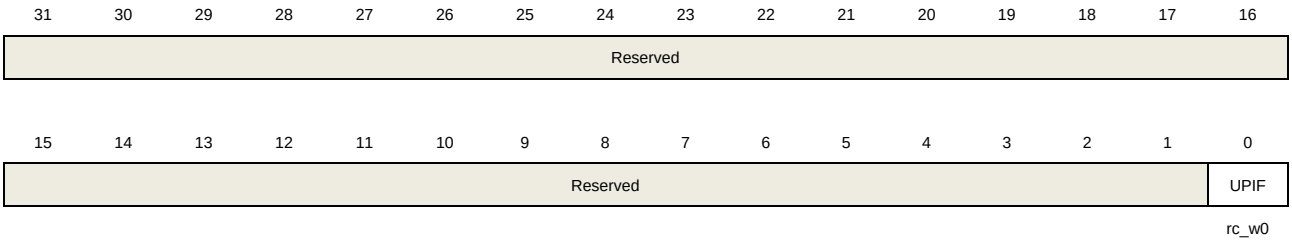


Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	UPDEN	Update DMA request enable 0: disabled 1: enabled
7:1	Reserved	Must be kept at reset value.
0	UPIE	Update interrupt enable 0: disabled 1: enabled

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.

16. Universal synchronous/asynchronous receiver /transmitter (USART)

16.1. Overview

The Universal Synchronous / Asynchronous Receiver/Transmitter (USART) provides a flexible serial data exchange interface. Data frames can be transferred in full duplex or half duplex mode, synchronously or asynchronously through this interface. A programmable baud rate generator divides the UCLK (PCLK1, PCLK2 and CK_USART0 available only for USART0) 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.

16.2. Characteristics

- NRZ standard format.
- Asynchronous, full duplex communication.
- Half duplex single wire communications.
- Receive FIFO function.
- Dual clock domain:
  - Asynchronous PCLK and USART clock independent of PCLK.
  - Baud rate programming independent from the PCLK reprogramming.
- Programmable baud-rate generator allowing speed up to 20 MBits/s when the clock frequency is 160 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 and UART2 is only partially implemented with the following features not supported.

- Smartcard mode.
- IrDA SIR ENDEC block.
- LIN mode.
- Dual clock domain and wakeup from deep-sleep mode.
- Receiver timeout interrupt.
- ModBus communication.
- Synchronous mode.
- Hardware flow control

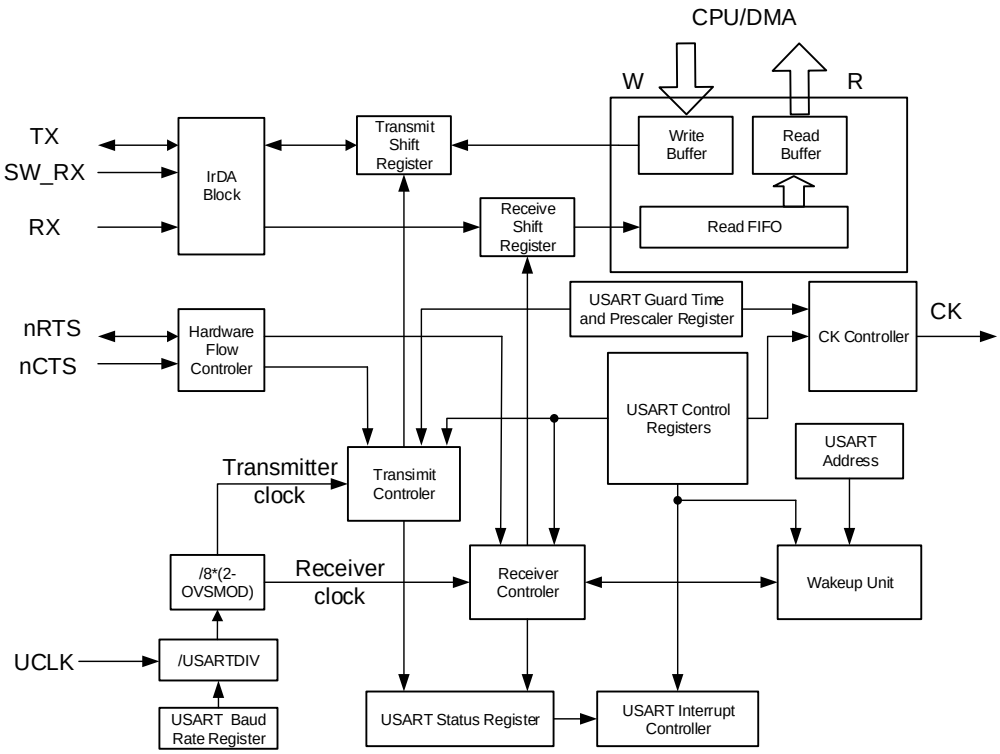
16.3. Function overview

The interface is externally connected to another device by the main pins listed in *Table 16-1. Description of USART important pins*.

Table 16-1. 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

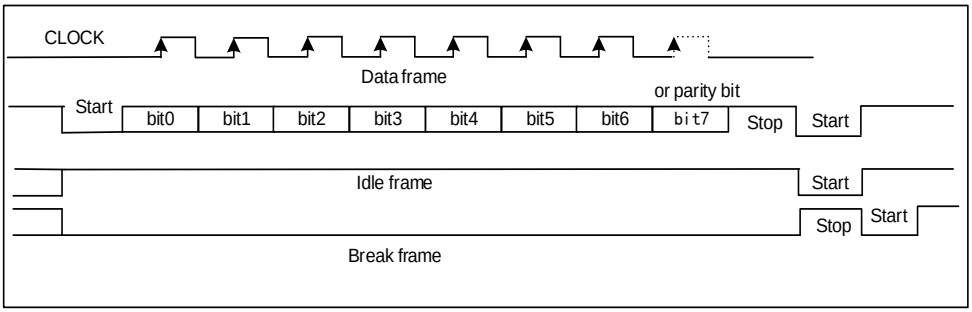
Figure 16-1. USART module block diagram



16.3.1. USART frame format

The USART frame starts with a start bit and ends up with a number of stop bits. The length of the data frame is configured by the WL bit in the USART_CTL0 register. The last data bit can be used as parity check bit by setting the PCEN bit in USART_CTL0 register. When the WL bit is reset, the parity bit is the 7th bit. When the WL bit is set, the parity bit is the 8th bit. The method of calculating the parity bit is selected by the PM bit in USART_CTL0 register.

Figure 16-2. 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 16-2. 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.

16.3.2. 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 clock:

In case of oversampling by 16, the equation is:

$$\text{USARTDIV} = \frac{\text{UCLK}}{16 \times \text{Baud Rate}} \quad (16\text{-}1)$$

In case of oversampling by 8, the equation is:

$$\text{USARTDIV} = \frac{\text{UCLK}}{8 \times \text{Baud Rate}} \quad (16\text{-}2)$$

For example, when oversampled by 16:

1. Get USARTDIV by caculating the value of USART_BAUD:
   If USART_BAUD=0x21D, then INTDIV=33 (0x21), FRADIV=13 (0xD).
   USARTDIV=33+13/16=33.81.

2. Get the value of USART_BAUD 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_BAUD=0x1E6.

   Note: If the roundness of FRADIV is 16 (overflow), the carry must be added to the integer part.

16.3.3. 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.

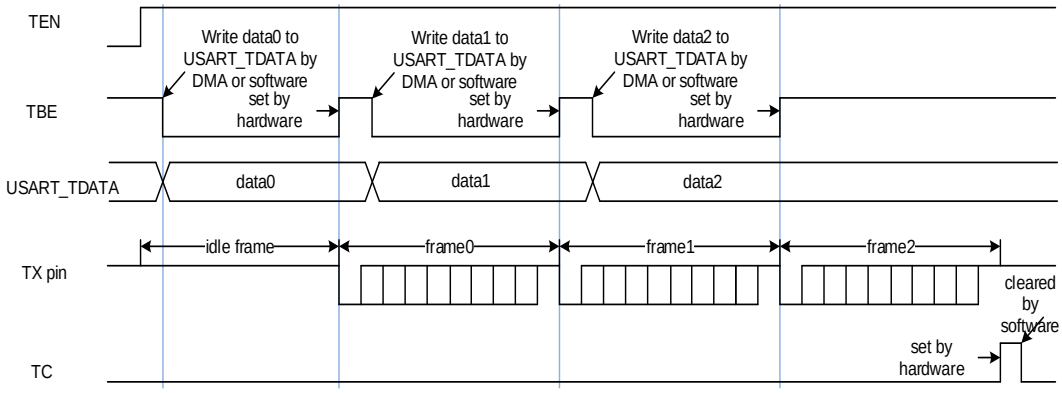
After power on, the TBE bit is high by default. Data can be written to the USART_TDATA when the TBE bit in the USART_STAT register is asserted. The TBE bit is cleared by writing USART_TDATA register and it is set by hardware after the data is put into the transmit shift register. If a data is written 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 16-3. 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 16-3. 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.

16.3.4. 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 are stored in the USART_STAT register.

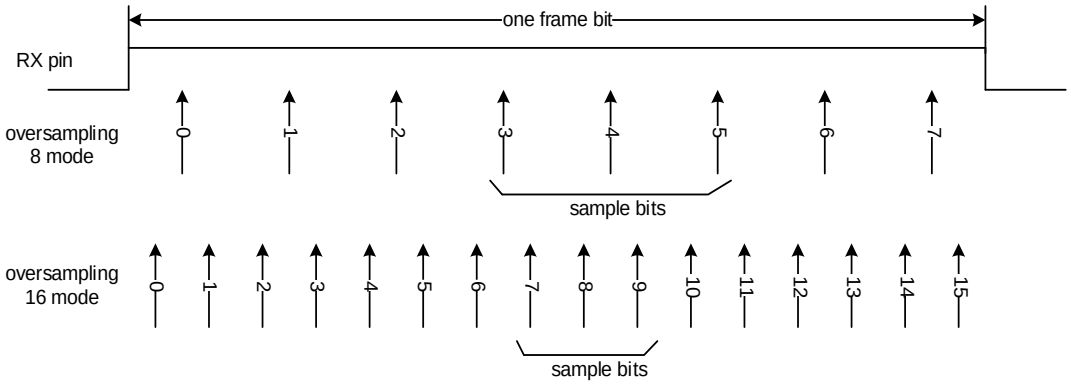
The software can get the received data by reading the USART_RDATA register directly, or through DMA. The RBNE bit is cleared by a read operation on the USART_RDATA register, whatever it is performed by software directly, or through DMA.

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 start bit, 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 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 16-4. 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 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 ERRIE bit in USART_CTL2 register is set, or if the RBNEIE is set.

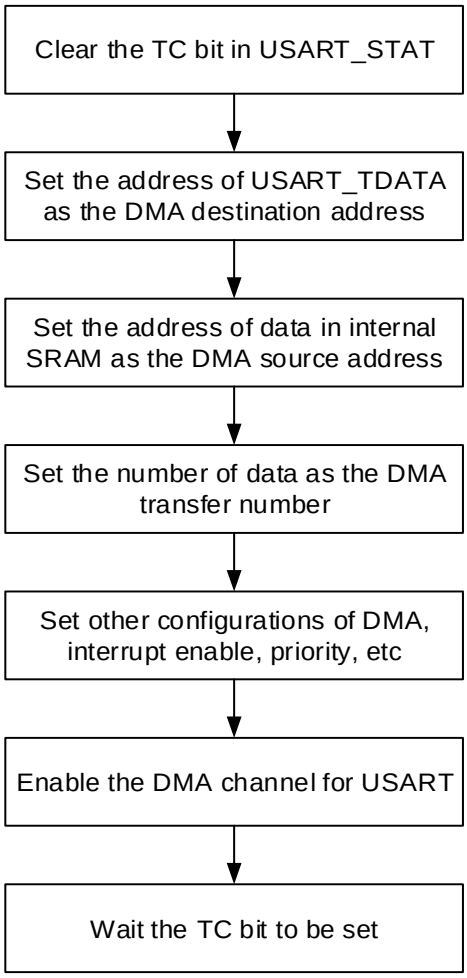
If a noise error (NERR), parity error (PERR), frame error (FERR) or overrun error (ORERR) occurs during reception, NERR, PERR, FERR or ORERR will be set at the same time with RBNE. If the receive DMA is not enabled, when the RBNE interrupt occurs, software need to check whether there is a noise error, parity error, frame error or overrun error.

16.3.5. 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 16-5. Configuration step when using DMA for USART transmission***.

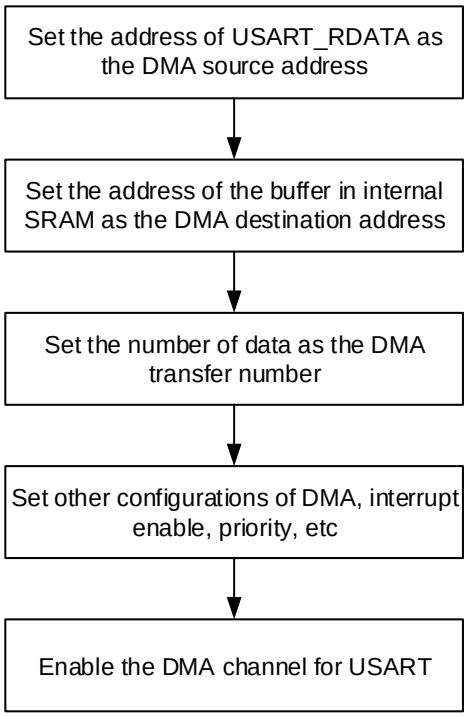
Figure 16-5. Configuration step when using DMA for USART transmission



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 16-6. 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 16-6. Configuration step when using DMA for USART reception

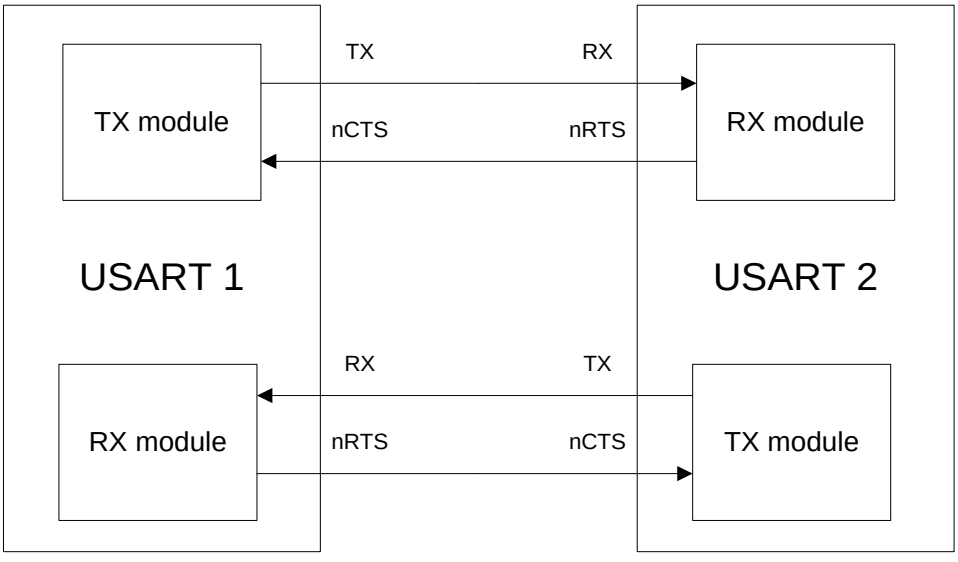


When the number of the data received by USART reaches the DMA transfer number, an end of transfer interrupt can be generated in the DMA module.

16.3.6. 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 16-7. Hardware flow control between two USARTs



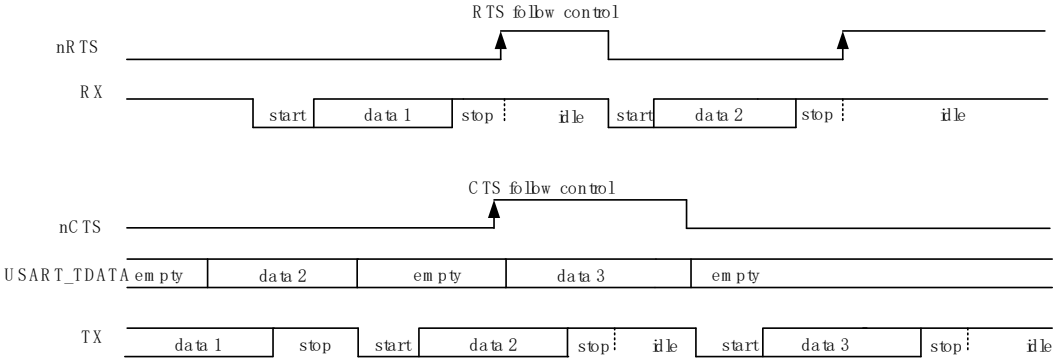
RTS flow control

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.

Figure 16-8. 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.

16.3.7. 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. If the RWU bit is reset, an idle frame is detected on the RX pin, the IDLEF bit inUSART_STAT will be set. If the RWU bit is set, an idle frame is detected on the RX pin, the hardware clears the RWU bit and exits the mute mode. When it is woken up by an idle frame, the IDLEF bit in USART_STAT will be set.

When the WM bit of in USART_CTL0 register is set, the MSB bit of a frame is detected as the address flag. If the address flag is high, the frame is treated as an address frame. If the address flag is low, the frame is treated as a data frame. If the LSB 4 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 the PCEN bit in USART_CTL0 is set, the MSB bit will be checked as the parity bit, and the bit preceding the MSB bit is detected as the address bit. If the ADDM bit is set and the receive frame is a 7bit data, the LSB 6 bits will be compared with ADDR[5:0]. If the ADDM bit is set and the receive frame is a 9bit data, the LSB 8 bits will be compared with ADDR[7:0].

16.3.8. LIN mode

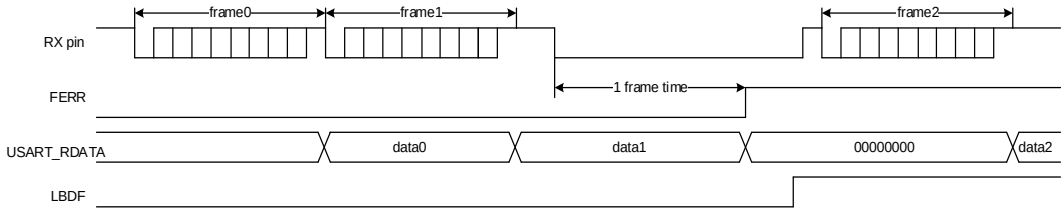
The local interconnection network mode is enabled by setting the LMEN bit in USART_CTL1. The CKEN, STB[1:0] bit in USART_CTL1 and the SCEN, HDEN, IREN bits in USART_CTL2 should be cleared in LIN mode.

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.

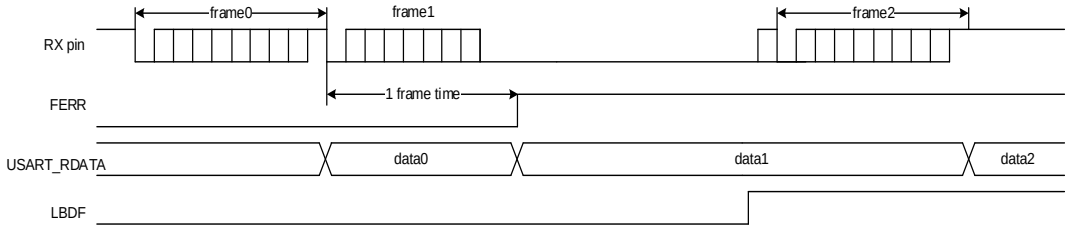
As shown in ***Figure 16-9. Break frame occurs during idle state***, if a break frame occurs during the idle state on the RX pin, the USART receiver will receive an all ‘0’ frame, with an asserted FERR status.

Figure 16-9. Break frame occurs during idle state



As shown in ***Figure 16-10. 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 16-10. Break frame occurs during a frame



16.3.9. 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 16-11. Example of USART in synchronous mode

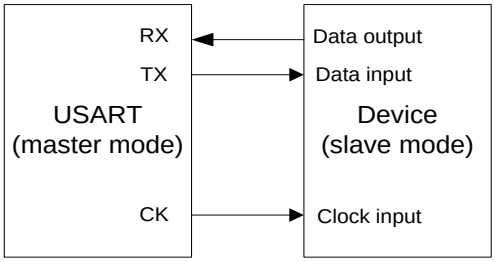
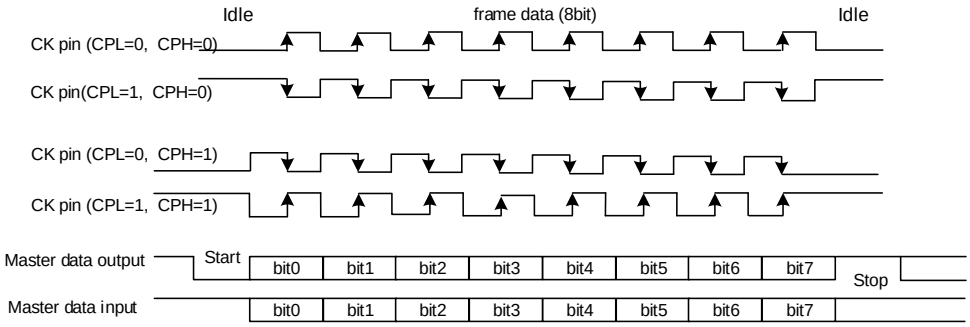


Figure 16-12. 8-bit format USART synchronous waveform (CLEN=1)

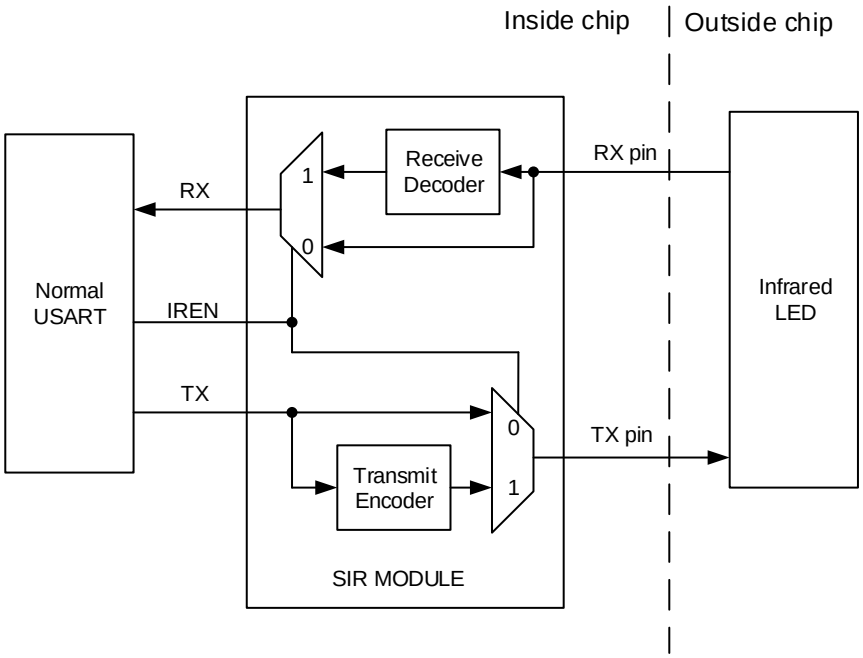


16.3.10. IrDA SIR ENDEC mode

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.

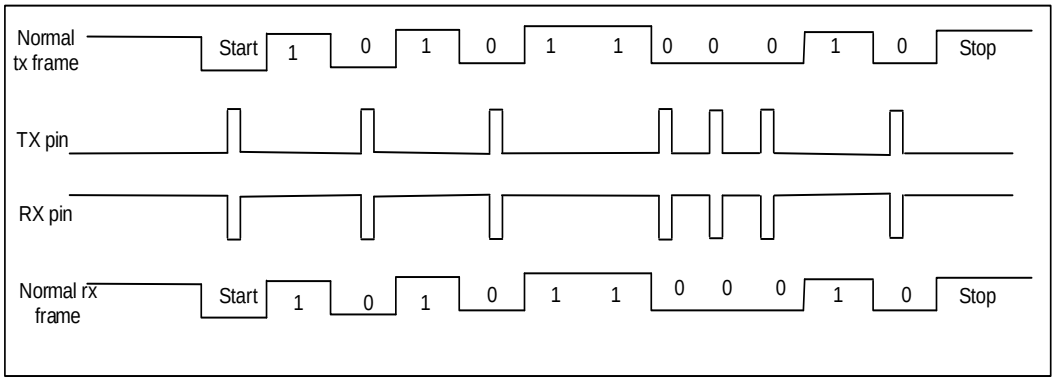
Figure 16-13. IrDA SIR ENDEC module



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 16-14. 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. 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.

16.3.11. 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.

16.3.12. Smartcard (ISO7816-3) mode

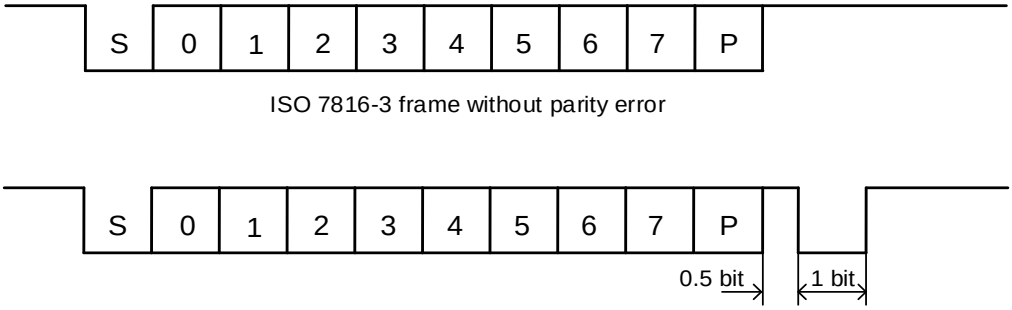
The smartcard mode is an asynchronous mode, which is designed to support the ISO7816-3 protocol. Both the character (T=0) mode and the block (T=1) mode are supported. The smartcard mode is enabled by setting the SCEN bit in USART_CTL2. The LMEN bit in USART_CTL1 and HDEN, IREN bits in USART_CTL2 should 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 16-15. 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 2.5 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 high 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 low state on the signal line and parity is even. In this case, the following control bits must be programmed: MSBF=1, DINV=1.

16.3.13. 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.

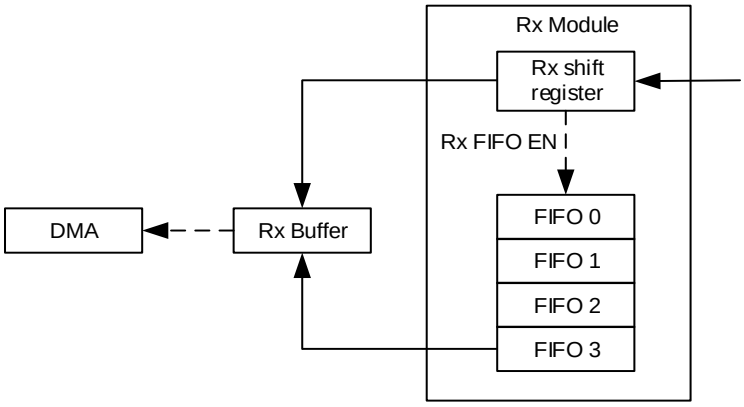
To detect the idle line, the RTEN bit in the USART_CTL1 register and the RTIE in the USART_CTL0 register must be set. The USART_RT register must be set to the value corresponding to a timeout of 2 characters time. After the last stop bit is received, when the receive line is idle for this duration, an 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.

16.3.14. 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 16-16. 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.

16.3.15. 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 IRC16M or LXTAL (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.

16.3.16. USART interrupts

The USART interrupt events and flags are listed in ***Table 16-3. USART interrupt requests***.

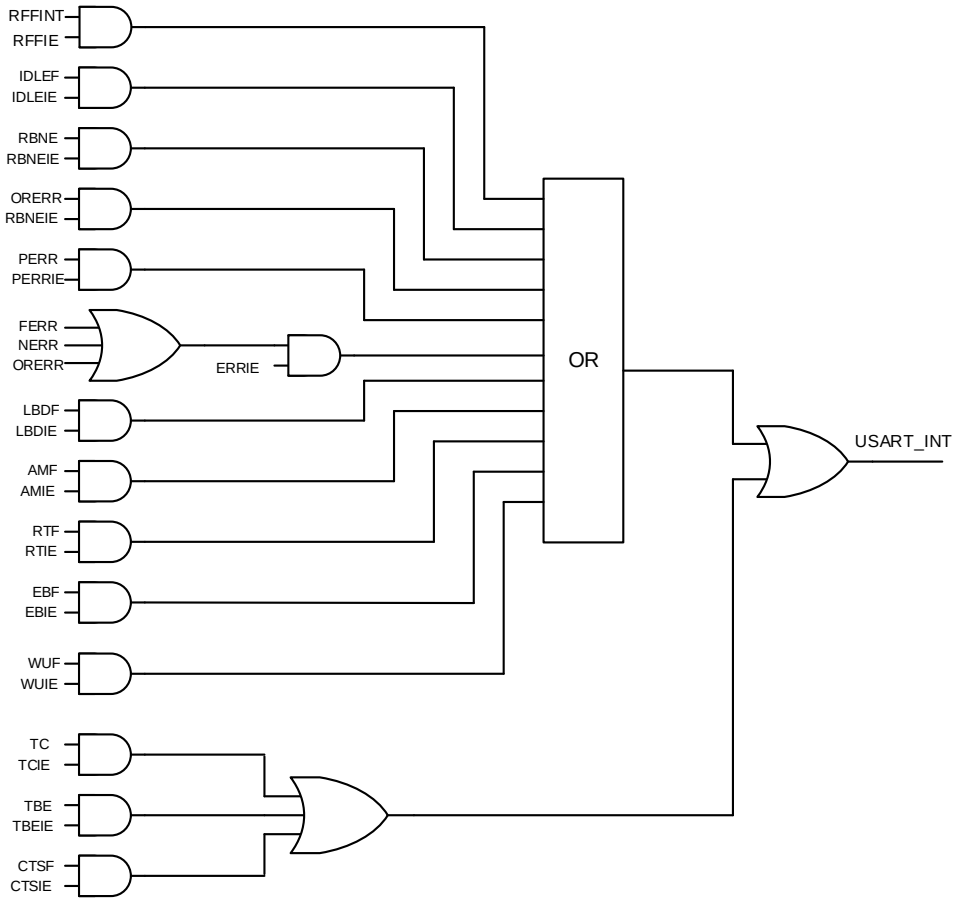
Table 16-3. USART interrupt requests

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)	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 16-17. USART interrupt mapping diagram



16.4. Register definition

USART0 base address: 0x4000 4800

UART1 base address: 0x4000 4400

UART2 base address: 0x4001 1000

16.4.1. 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 and UART2.
26	RTIE	Receiver timeout interrupt enable 0: Receiver timeout interrupt is disabled 1: Receiver timeout interrupt is enabled This bit is reserved in UART1 and UART2.
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 IRC16M or LXTAL. This bit is reserved in UART1 and UART2.
0	UEN	USART enable 0: USART prescaler and outputs disabled 1: USART prescaler and outputs enabled

16.4.2. 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	Reserved			MSBF	DINV	TINV	RINV
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 and UART2.
22:20	Reserved	Must be kept at reset value.
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 and UART2.
13:12	STB[1:0]	STOP bits length 00: 1 Stop bit 01: 0.5 Stop bit 10: 2 Stop bits 11: 1.5 Stop bit

		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 and UART2.
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).
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).
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)
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 and UART2.
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 and UART2.
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.

16.4.3. 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 and UART2.
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 and UART2.
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 and UART2.

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	Mask DMA request 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 when reception error, but the corresponding error flag is set. This mode can be used in smartcard mode 1: The DMA request is not asserted in case of reception error 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).

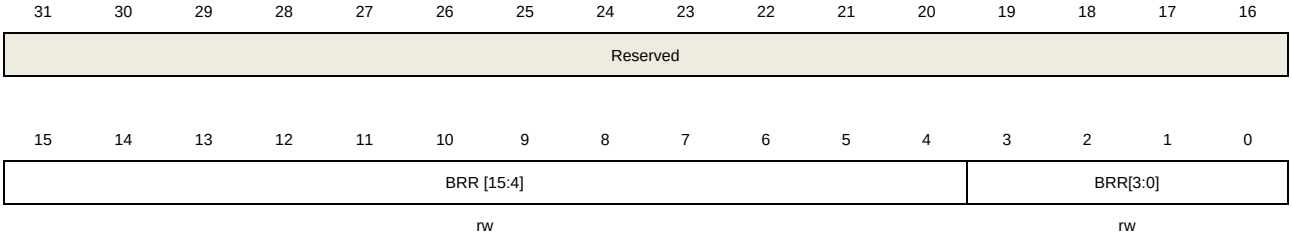
8	RTSEN	RTS enable 0: RTS hardware flow control disabled 1: RTS hardware flow control enabled, data can be requested only when there is space in the receive buffer This bit field cannot be written when the USART is enabled (UEN=1).
7	DENT	DMA enable for transmission 0: DMA mode is disabled for transmission 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 and UART2.
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 and UART2.
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 and UART2.
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

16.4.4. Baud rate generator register (USART_BAUD)

Address offset: 0x0C
Reset value: 0x0000 0000

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

This register cannot be written when the USART is enabled (UEN=1).



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.

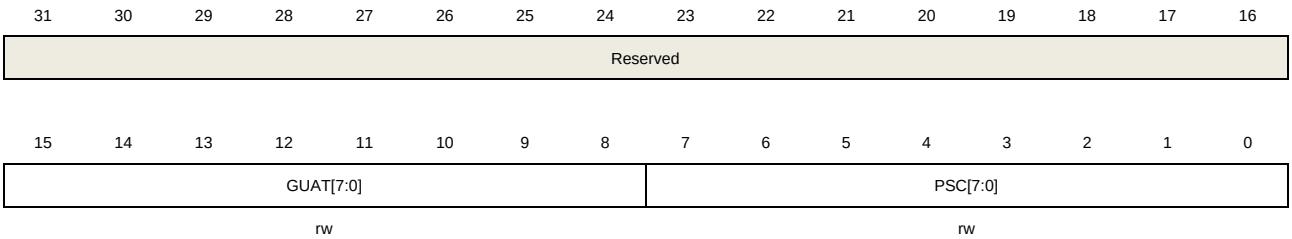
16.4.5. Prescaler and guard time configuration register (USART_GP)

Address offset: 0x10
Reset value: 0x0000 0000

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

This register cannot be written when the USART is enabled (UEN=1).

This register is reserved in UART1 and UART2.



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).

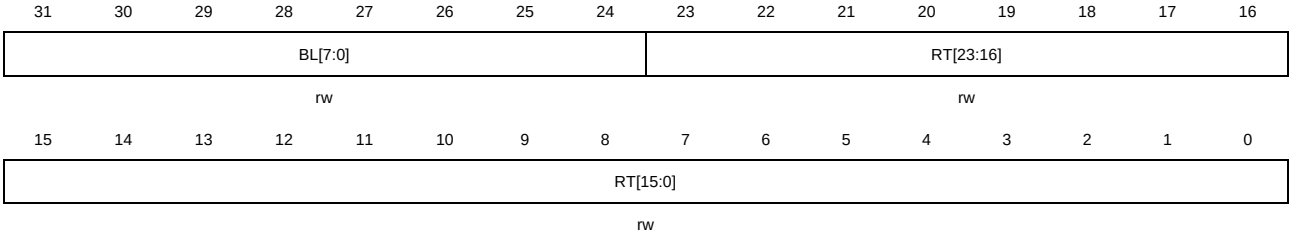
16.4.6. Receiver timeout register (USART_RT)

Address offset: 0x14

Reset value: 0x0000 0000

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

This register is reserved in UART1 and UART2.



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.

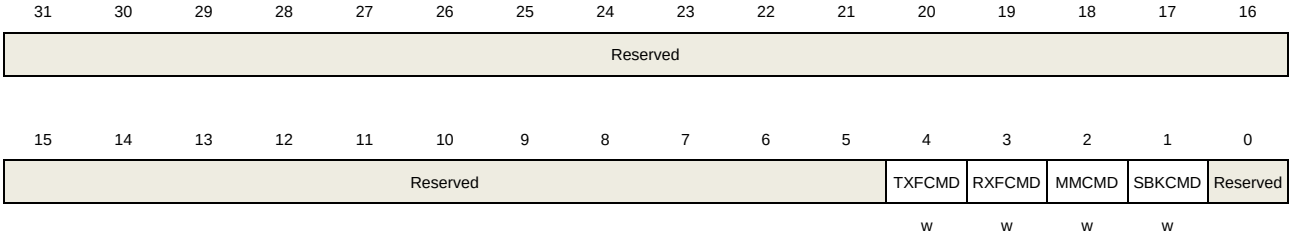
23:0	RT[23:0]	Receiver timeout threshold These bits specify receiver timeout value in terms of number of baud clocks. In standard mode, the RTF flag is set if no new start bit is detected for more than the RT value after the last received character. In smartcard mode, the CWT and BWT are implemented 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.

16.4.7. Command register (USART_CMD)

Address offset: 0x18

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	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 and UART2.
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	Reserved	Must be kept at reset value.

16.4.8. 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
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

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
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 and UART2.
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
15:13	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 and UART2.
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 and UART2.
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
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.
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 and UART2.
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 Ooverrun error is detected 1: Overrun error is detected. An interrupt will occur if the RBNEIE bit is set in USART_CTL0. In multibuffer communication, 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. In multibuffer communication, 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. In multibuffer communication, 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.

16.4.9. 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 and UART2.
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 and UART2.
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 and UART2.
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 and UART2.
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

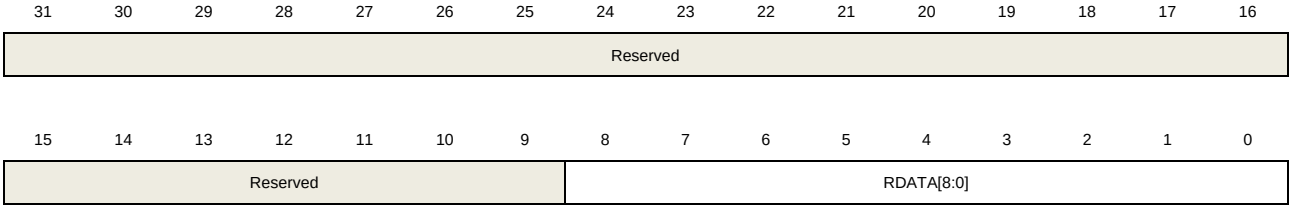
Writing 1 to this bit clears the PERR bit in the USART_STAT register.

16.4.10. Receive data register (USART_RDATA)

Address offset: 0x24

Reset value: Undefined

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



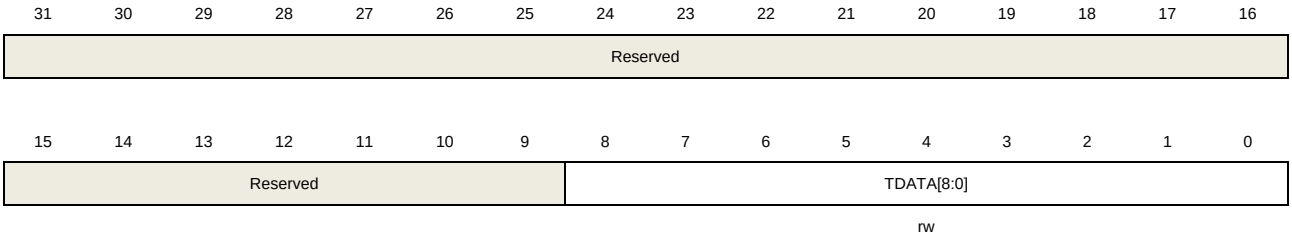
Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8:0	RDATA[8:0]	Receive data value The received data character is contained in these bits. The value read in the MSB (bit 7 or bit 8 depending on the data length) will be the received parity bit, if receiving with the parity is enabled (PCEN bit set to 1 in the USART_CTL0 register).

16.4.11. 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).

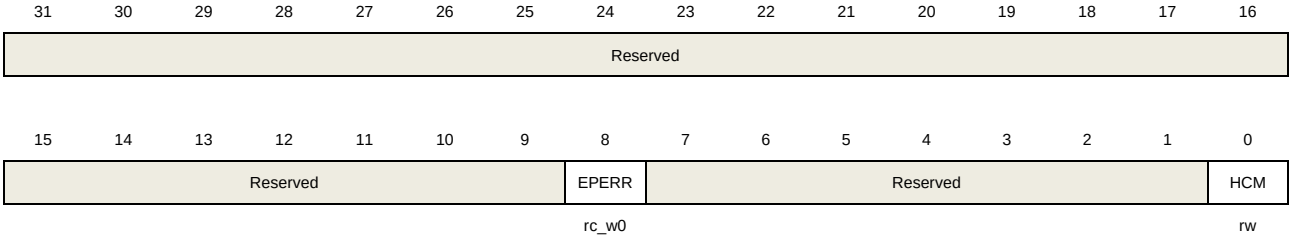
This register must be written only when TBE bit in USART_STAT register is set.

16.4.12. USART coherence control register (USART_CHC)

Address offset: 0xC0

Reset value: 0x0000 0000

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



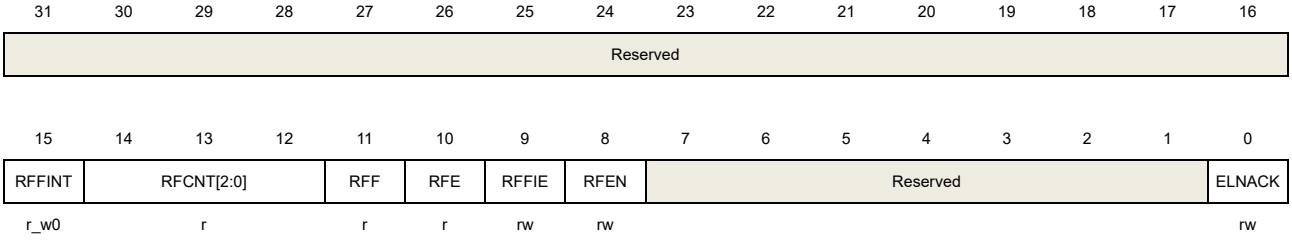
Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	EPERR	Early parity error flag. This flag will be set as soon as the parity bit has been detected, which is before RBNE flag. This flag is cleared by writing 0. 0: No parity error is detected 1: Parity error is detected.
7:1	Reserved	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.

16.4.13. 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).



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. 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 NACKdisable when smartcard mode is selected 1:Early NACKenable when smartcard mode is selected This bit is reserved in UART1 and UART2.

17. Inter-integrated circuit interface (I2C)

17.1. 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.

17.2. 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 address with configurable mask).
- Programmable setup time and hold time.
- Multi-master capability.
- Supports standard mode (up to 100 kHz) and fast mode (up to 400 kHz) and fast mode plus (up to 1MHz).
- Configurable SCL stretching in slave mode.
- Supports DMA mode.
- SMBus 3.0 and PMBus 1.3 compatible.
- Optional PEC (packet error checking) generation and check.
- Programmable analog and digital noise filters.
- Wakeup from Deep-sleep mode on I2C0 address match.
- Independent clock from PCLK.

17.3. Function overview

Figure 17-1. I2C module block diagram below provides details on the internal configuration of the I2C interface.

Figure 17-1. I2C module block diagram

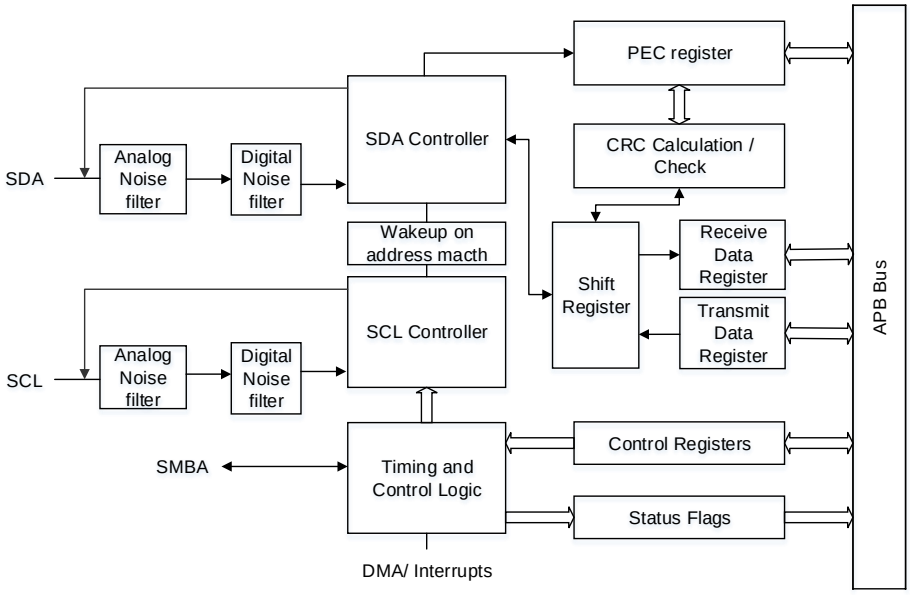


Table 17-1. Definition of I2C-bus terminology (refer to the I2C specification of Philips semiconductors)

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
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

17.3.1. Clock requirements

The I2C0 clock is independent of the PCLK frequency, so that the I2C0 can be operated independently.

This I2C0 clock (I2CCLK) can be selected from the following three clock sources:

- PCLK1: APB1 clock (default value)
- IRC16M: internal 16 MHz RC
- SYSCLK: system clock

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 260ns. Digital filter delay is DNF[3:0] x tI2CCLK

The period of PCLK clock tPCLK 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}.

17.3.2. I2C communication flow

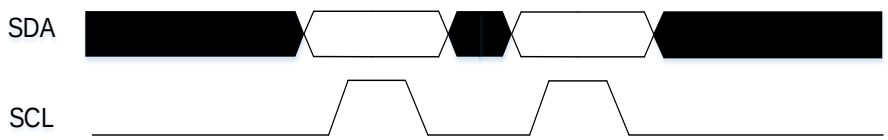
An I2C device is able to transmit or receive data whether it's a master or a slave, thus, there're 4 operation modes for an I2C device:

- 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 17-2. Data validation***). One clock pulse is generated for each data bit transferred.

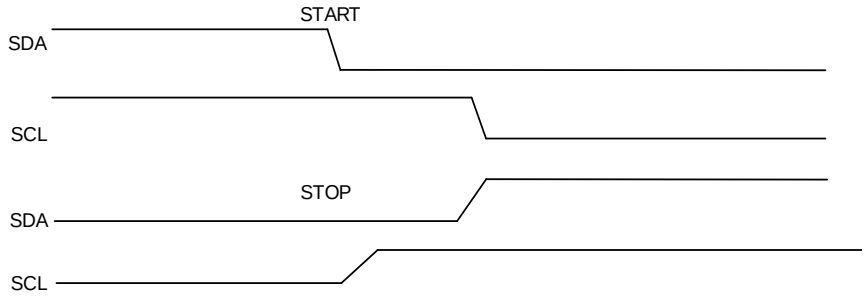
Figure 17-2. Data validation



START and STOP condition

All transactions begin with a START (S) and are terminated by a STOP (P) (see ***Figure 17-3. START and STOP condition***). A HIGH to LOW transition on the SDA line while SCL is HIGH defines a START condition. A LOW to HIGH transition on the SDA line while SCL is HIGH defines a STOP condition.

Figure 17-3. START and STOP condition



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 condition, 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 condition on 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 condition contain 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 condition and it's also responsible for SCL clock generation.

In master mode, if AUTOEND=1, the STOP condition is generated automatically by hardware. If AUTOEND=0, the STOP condition generated by software, or the master can generate a RESTART condition to start a new transfer.

Figure 17-4. I2C communication flow with 10-bit address (Master Transmit)

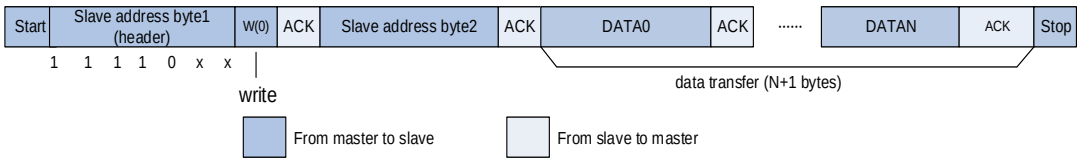


Figure 17-5. I2C communication flow with 7-bit address (Master Transmit)

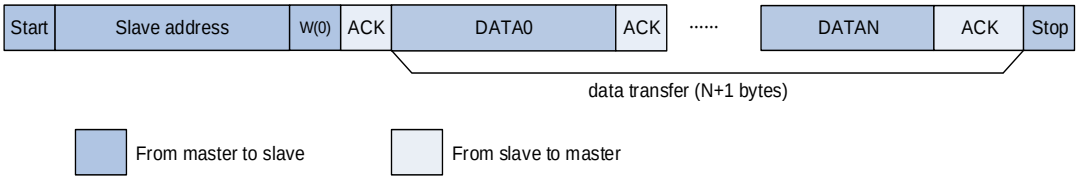
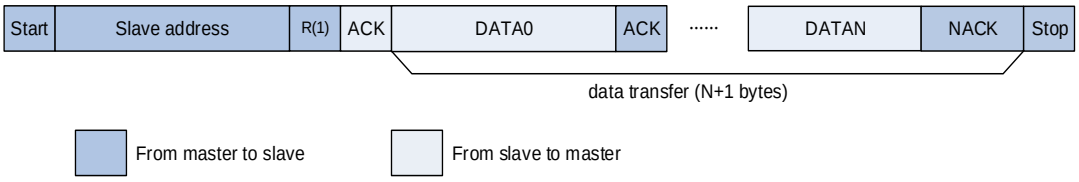


Figure 17-6. I2C communication flow with 7-bit address (Master Receive)



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 17-7. 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 17-8. I2C communication flow with 10-bit address (Master Receive when HEAD10R=1)***.

Figure 17-7. I2C communication flow with 10-bit address (Master Receive when HEAD10R=0)

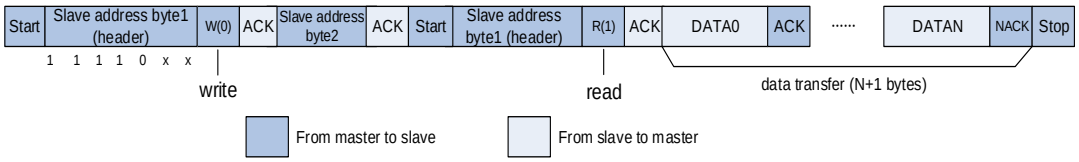
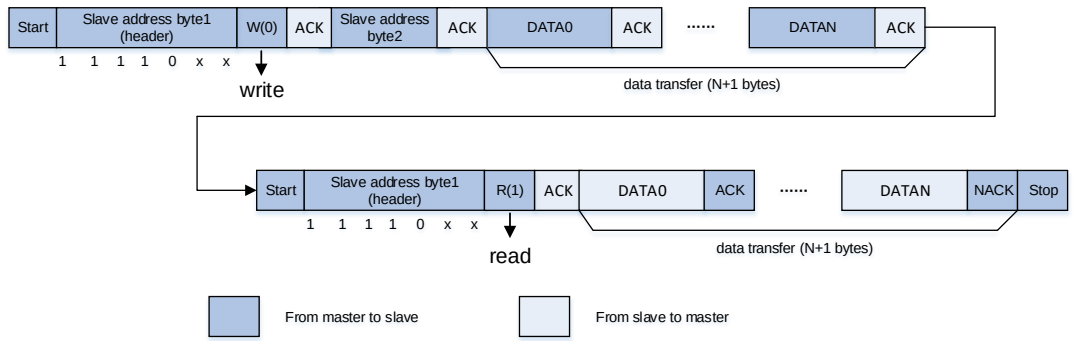


Figure 17-8. I2C communication flow with 10-bit address (Master Receive when HEAD10R=1)



17.3.3. Noise filter

The noise filters must be configured before setting the I2CEN bit in I2C_CTL0 register if it is necessary. The analog noise filter is present on the SDA and SCL inputs by default. The analog filter requires the suppression of spikes with a pulse width up to 50ns in fast mode and fast mode plus. The analog filter can be disabled by setting the ANOFF bit in I2C_CTL0 register.

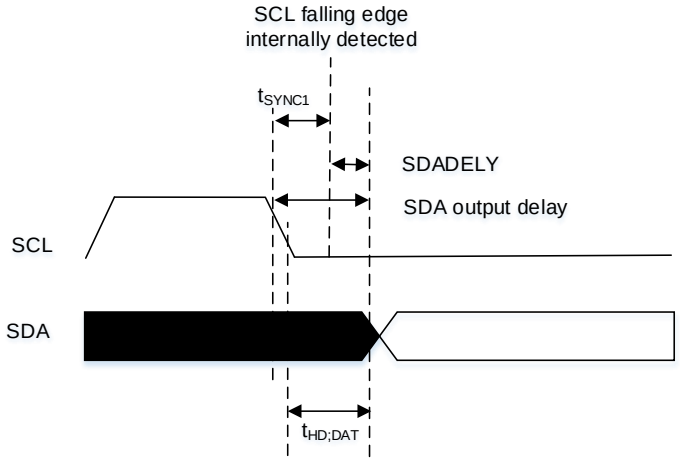
The digital filter can be used by configuring the DNF[3:0] bit in I2C_CTL0 register. When the digital filter is enabled, the level of the SCL or the SDA line is internally changed only if it remains stable for more than DNF[3:0] x t_{I2CCLK}. This allows to suppress spikes with a programmable length of 1 to 15 of t_{I2CCLK}.

17.3.4. I2C timings

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.

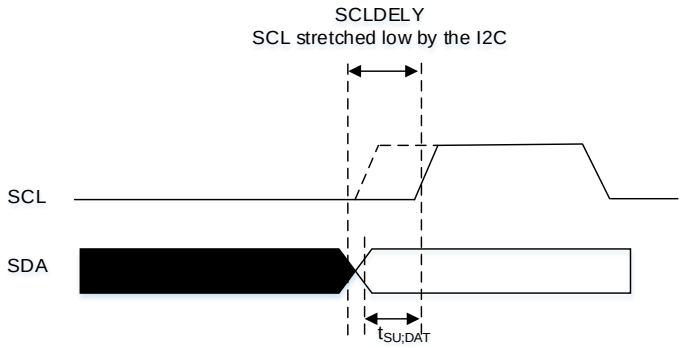
If the data is already available in I2C_TDATA register, the data will be sent on SDA after the SDADELY delay. As is shown in ***Figure 17-9. Data hold time***.

Figure 17-9. Data hold time



The SCLDELY counter starts when the data is sent on SDA output. As is shown in ***Figure 17-10. Data setup time***.

Figure 17-10. 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}.

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: 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_{SUDAT}(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 17-2. 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	

17.3.5. Software 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

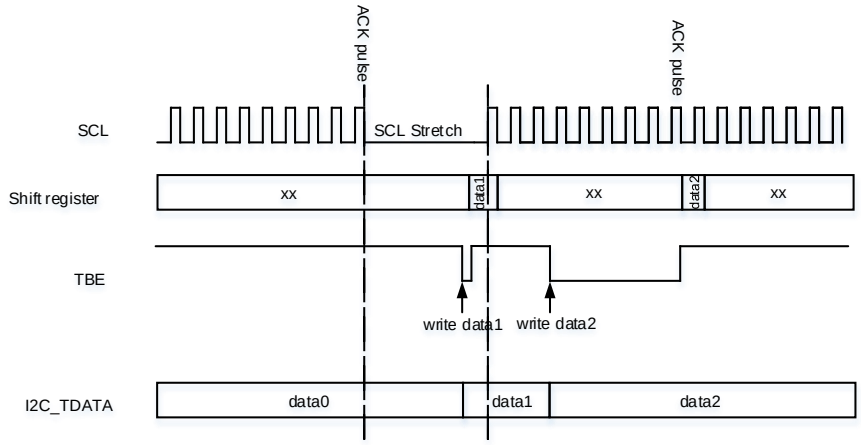
17.3.6. Data transfer

The data transfer is managed through data registers and shift register.

Data Transmission

If the I2C_TDATA register is not empty, that is, TBE=0, its content is moved to the shift register after the 9th SCL pulse (Acknowledge pulse). The shift register content is shifted out on SDA line. If TBE=1 means that no data is written to I2C_TDATA, SCL line is stretched low until I2C_TDATA is written. The stretch is done after the 9th SCL pulse.

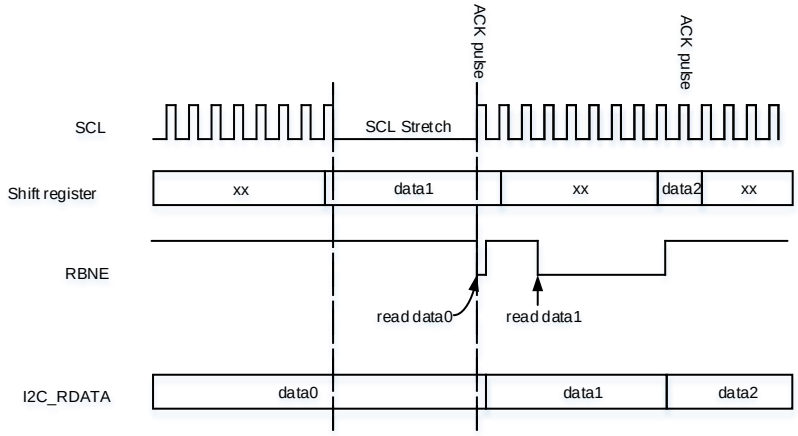
Figure 17-11. Data transmission



Data Reception

When receiving data, the SDA input fills the shift register. After the 8th SCL pulse, the complete data byte is received. If RBNE=0 (I2C_RDATA register is empty), the data in the shift register is moved into I2C_RDATA register. If RBNE=1 indicates that the previous received data byte has not been read, the SCL line is stretched low until I2C_RDATA is read. The stretch is inserted between the 8th and 9th SCL pulse (before Acknowledge pulse).

Figure 17-12. Data reception



Hardware transfer management

In order to manage byte transfer and to shut down the communication in modes as is shown in ***Table 17-3. Communication modes to be shut down***, the I2C embedded a byte counter in the hardware.

Table 17-3. Communication modes to be shut down

Working mode	Action
Master mode	NACK, STOP and RESTART generation
Slave receiver mode	ACK control
SMBus mode	PEC generation/checking

The byte counter is always used in master mode. It is disabled in slave mode by default, but it can be enabled by software by setting the SBCTL (slave byte control) bit in the I2C_CTL0 register.

The number of bytes to be transferred is programmed in the BYTENUM[7:0] bit field of the I2C_CTL1 register. If the number of bytes to be transferred (BYTENUM) is greater than 255, or if a receiver wants to control the acknowledge value of a received data byte, the reload mode must be selected by setting the RELOAD bit in the I2C_CTL1 register. In this mode, TCR flag is set when the number of bytes programmed in BYTENUM has been transferred, and an interrupt is generated if TCIE is set. Once the TCR flag is set, SCL is stretched. When BYTENUM is written to a non-zero value, TCR is cleared by software.

When the BYTENUM counter is reloaded with the last number of bytes, RELOAD bit must be cleared.

When RELOAD = 0 in master mode, the counter can be used in two modes:

- **Automatic end mode** (AUTOEND = 1 in the I2C_CTL1 register). In this mode, once the number of bytes programmed in the BYTENUM[7:0] bit field has been transferred, the master automatically sends a STOP condition. Note that when the RELOAD bit is set, the AUTOEND bit has no effect.
- **Software end mode** (AUTOEND = 0 in the I2C_CTL1 register). In this mode, once the number of bytes programmed in the BYTENUM[7:0] bit field has been transferred, the TC flag is set and an interrupt is generated if the TCIE bit is set. As long as the TC flag is set, the SCL signal is stretched. When the START or STOP bit is set in the I2C_CTL1 register, the TC flag is cleared by software. This mode must be used when the master wants to send a RESTART condition.

Table 17-4. I2C configuration

Function	SBCTL bit	RELOAD bit	AUTOEND bit
Master Tx/Rx + BYTENUM + STOP	x	0	1
Master Tx/Rx + BYTENUM + RESTART	x	0	0
Slave Tx/Rx all received bytes ACKed	0	x	x
Slave Rx with ACK control	1	1	x

17.3.7. 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.
- After SCL falling edge detection, the I2C stretches SCL low during

  [(SDADELY+SCLDELY+1) x (PSC+1) + 1] x tI2CCLK.

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.

- The SCL will be not stretched while the ADDSEND is set.

- In slave transmitting mode, the data should be written in the I2C_TDATA register before the first SCL pulse corresponding to its transfer occurs. 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, the data must be read from the I2C_RDATA register before the 9th SCL pulse (ACK pulse) occurred by the next data byte. 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 order to get control of each byte, 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 slave stretchs the SCL low between the 8th and 9th clock pulses. Then the data can be read from the I2C_RDATA register, and the slave determined 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. The ACK or NACK is sent and the next byte can be received.

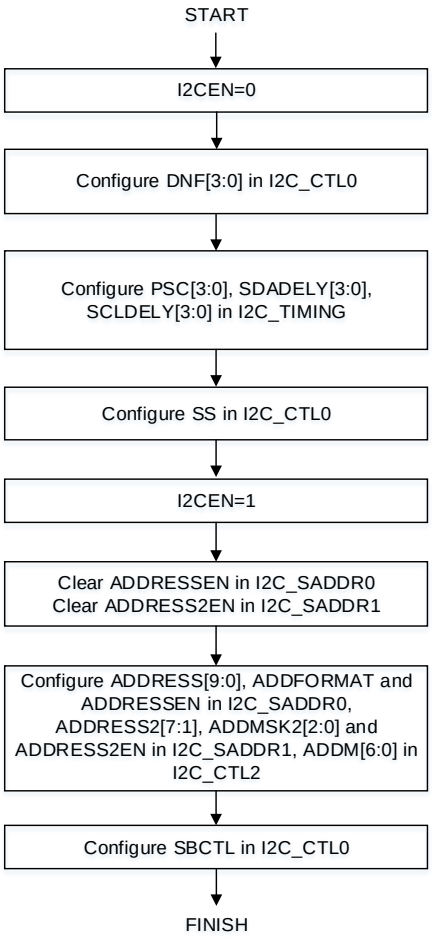
When the BYTENUM[7:0] is greater than 0x1, there is no stretch between the reception of two data bytes.

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 17-13. I2C initialization in slave mode



Slave transmitter

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 data byte to be sent must be programmed in the I2C_TDATA register previously.

- This data can be the data written in the last TI event of the last transfer.
- If the data is not the one to be sent, setting the TBE bit can flush the data, then a new byte can be written in I2C_TDATA register. Then the STPDET bit should be cleared. 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 17-14. Programming model for slave transmitting when SS=0

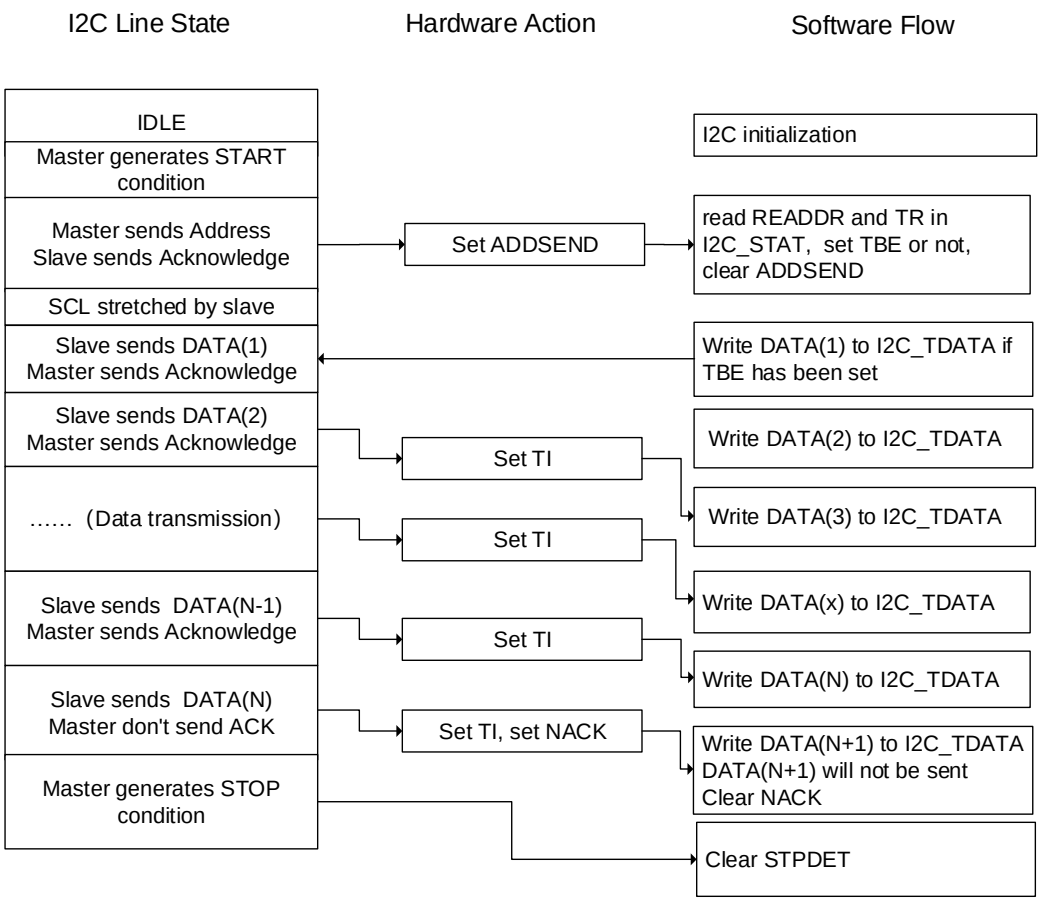
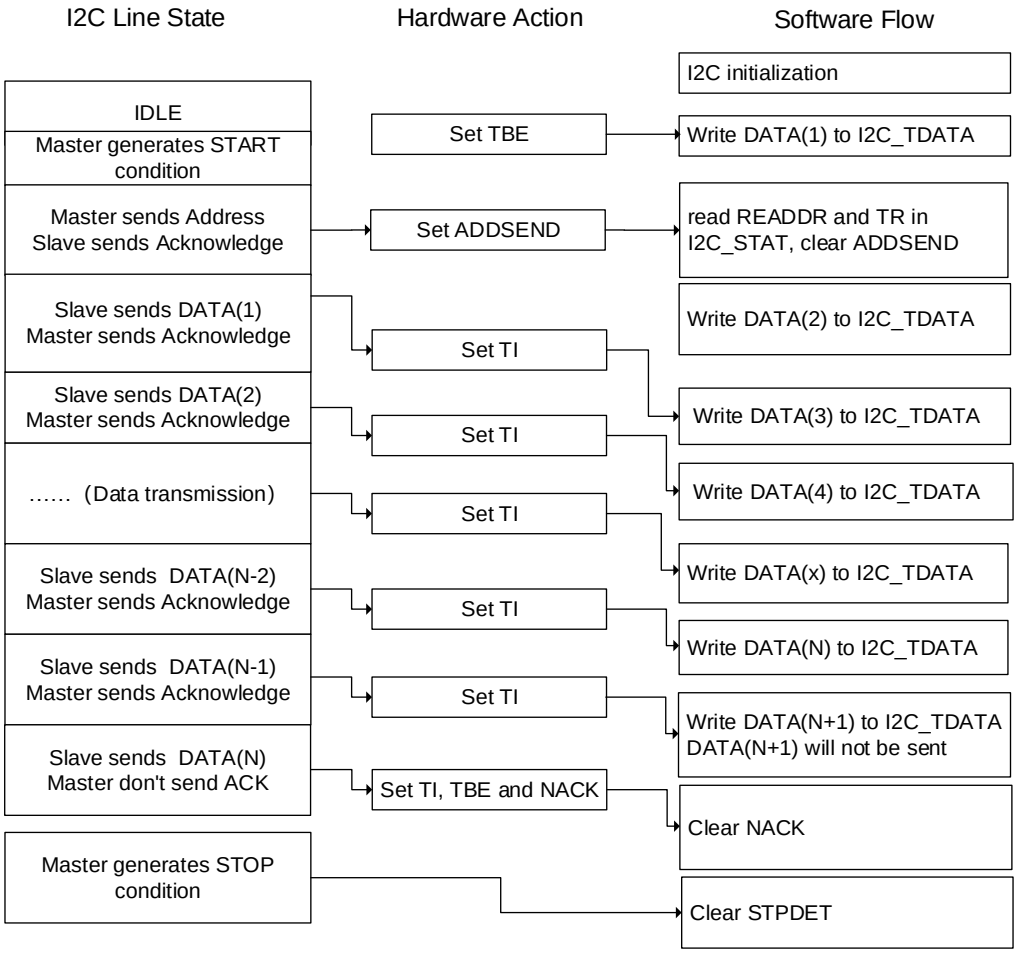


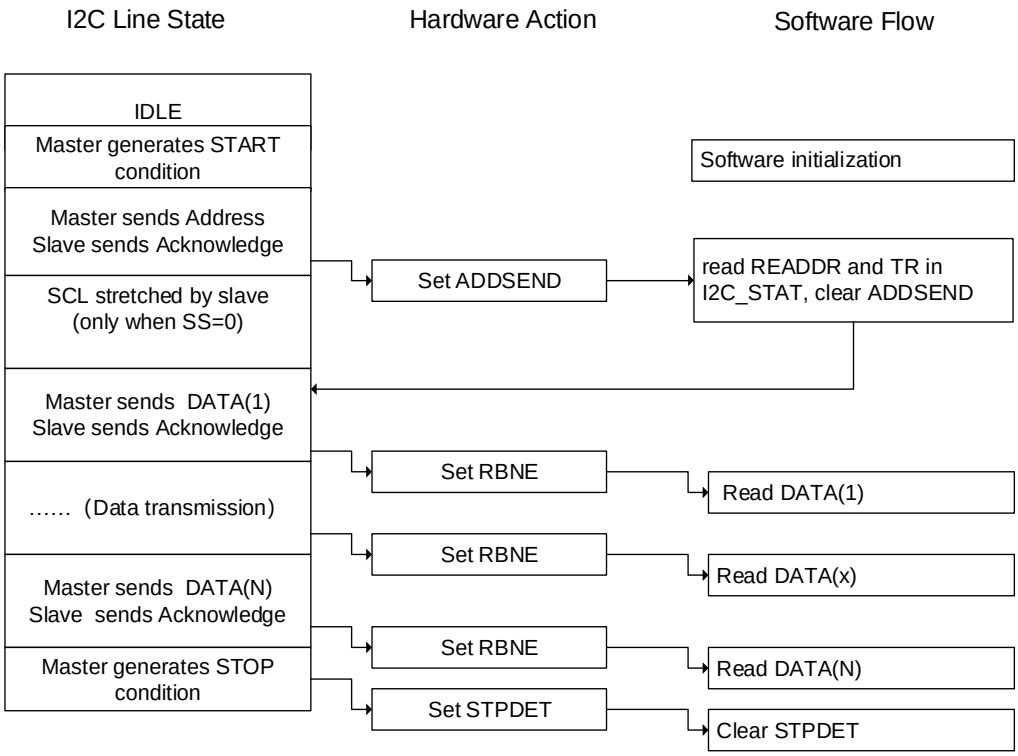
Figure 17-15. Programming model for slave transmitting when SS=1



Slave receiver

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 17-16. Programming model for slave receiving



17.3.8. 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.

For clock synchronization, the low level of the clock is counted starting from the SCL low level internal detection by the SCLL[7:0] counter, the high level of the clock is counted by the SCLH[7:0] counter, starting from the SCL high level internal detection.

The I2C detects its SCL low level after a t_{SYNC1} delay depending on the SCL falling edge, SCL input analog and digital noise filter and SCL synchronization to the I2CCLK clock. If the SCLL[7:0] value in I2C_TIMING register is reached by the the SCLL[7:0] counter, the I2C will release the SCL clock.

The I2C detects its SCL high level after a t_{SYNC2} delay depending on the SCL rising edge, SCL input analog and digital noise filter and SCL synchronization to I2CCLK clock. If the SCLH[7:0] value in I2C_TIMING register is reached by the the SCLH[7:0] counter, the I2C will stretch the SCL clock.

So the master clock period is: $t_{SCL} = t_{SYNC1} + t_{SYNC2} + \{[(SCLH[7:0]+1) + (SCLL[7:0]+1)] \times$

(PSC+1) x t_{I2CCLK}}

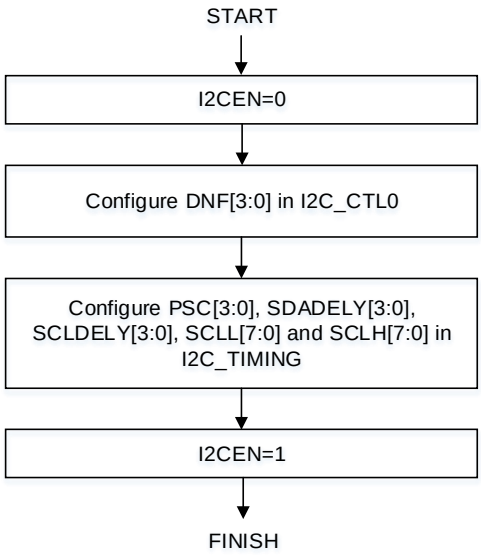
The t_{SYNC1} depends on the SCL falling slope, delay by input analog and digital noise filter and SCL synchronization with I2CCLK clock, which generally 2 to 3 I2CCLK periods. The t_{SYNC2} depends on the SCL rising slope, delay by input analog and digital noise filter and SCL synchronization with I2CCLK clock, which generally 2 to 3 I2CCLK periods. The delay by digital noise filter is DNF[3:0] x 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 condition. All the bits above should be configured before the START is set. The slave address will be sent after the START condition 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.

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 17-17. I2C initialization in master mode



Master transmitter

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_CTL0 register. If the bytes to be transferred is greater than 255, RELOAD bit in I2C_CTL0 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 condition 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 condition by setting the STOP bit in the I2C_CTL1 register. Or generate a RESTART condition to start a new transfer. The TC bit is cleared when the START/STOP bit is set.
- If a NACK is received, a STOP condition 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 17-18. Programming model for master transmitting (N<=255)

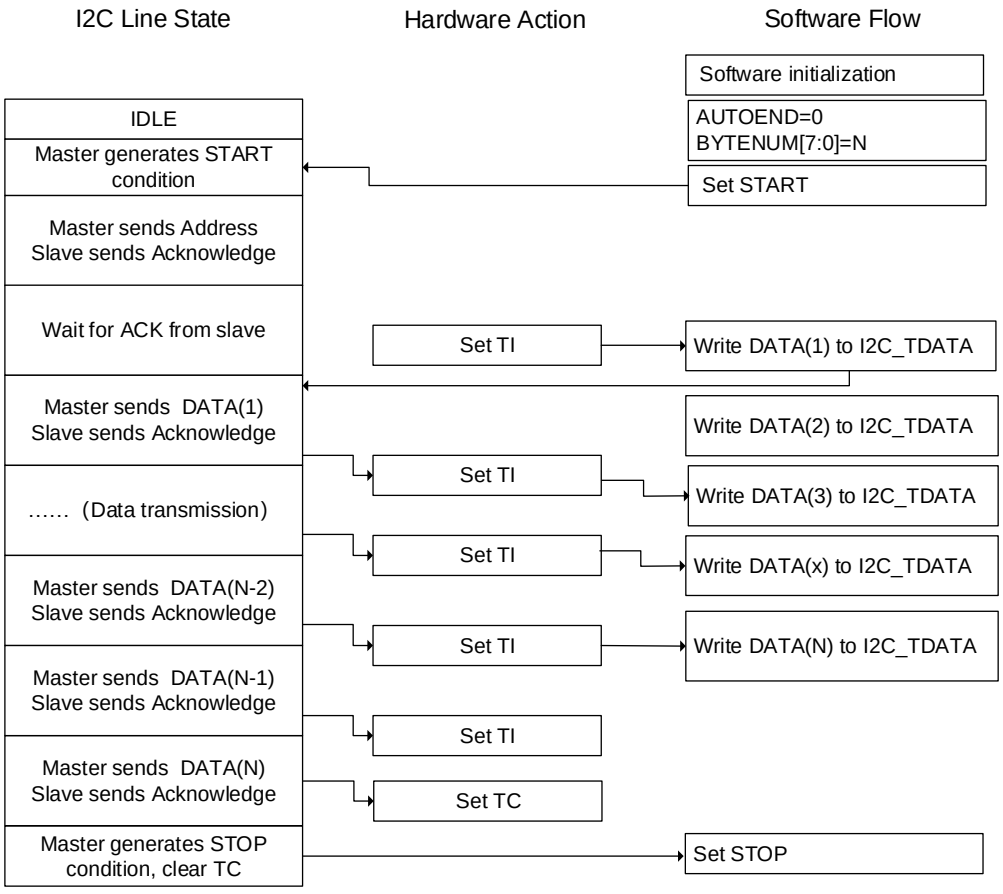
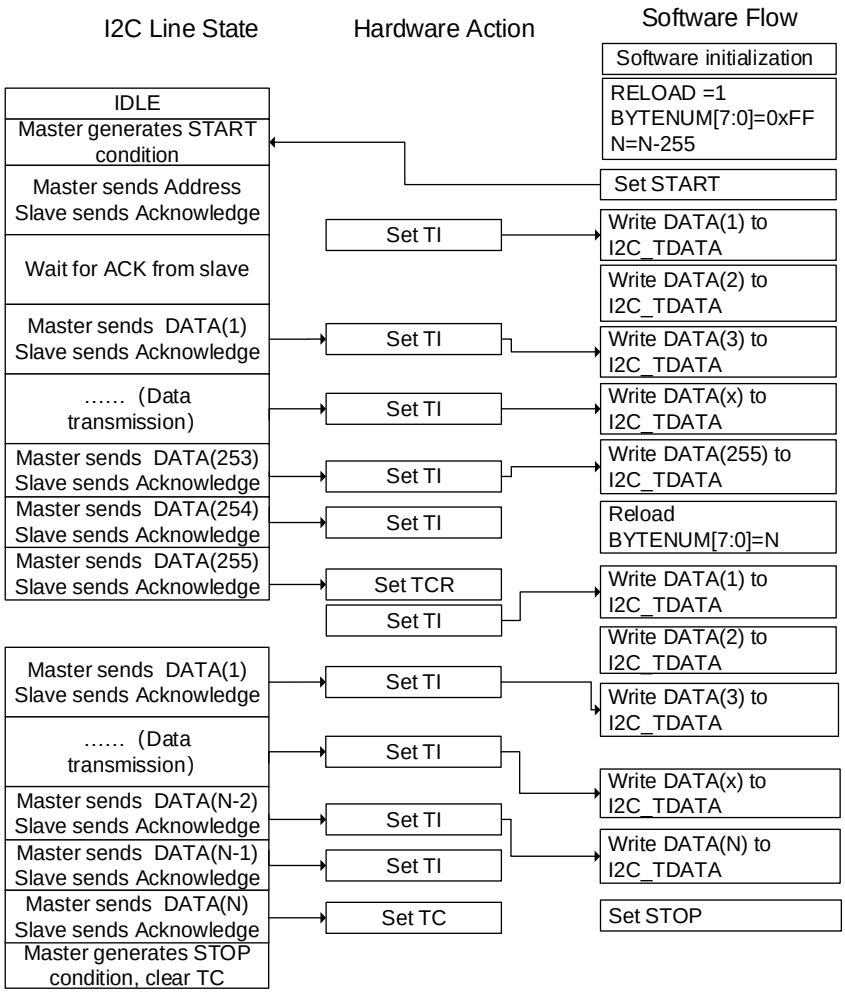


Figure 17-19. Programming model for master transmitting (N>255)



Master receiver

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_CTL0 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 condition 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 condition by setting the STOP bit in the I2C_CTL1 register. Or generate a RESTART condition to start a new transfer. The TC bit is cleared when the START bit is set.

Figure 17-20. Programming model for master receiving (N<=255)

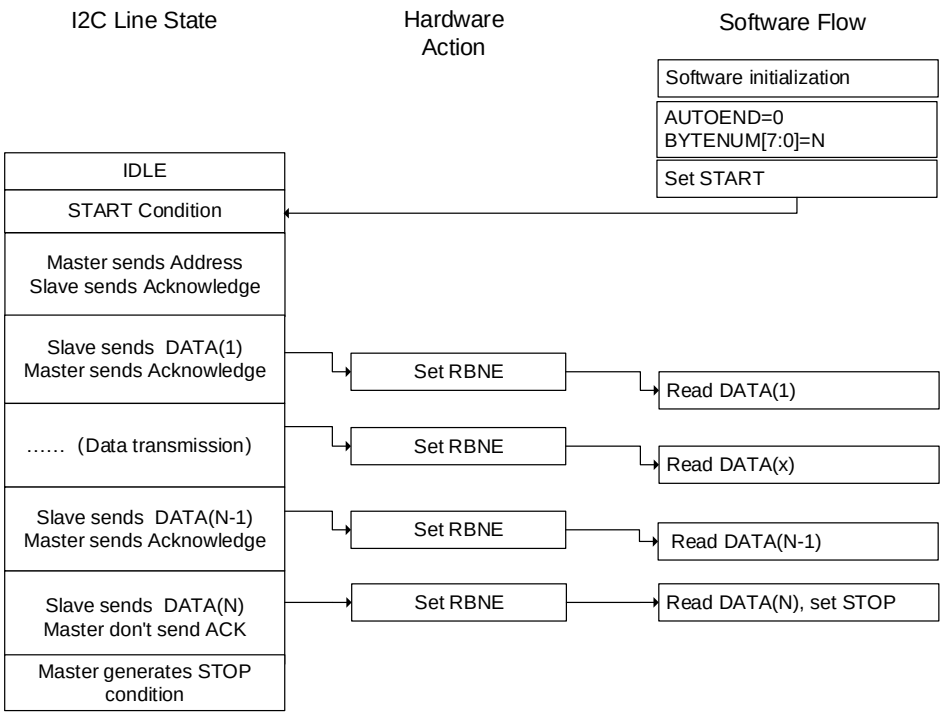
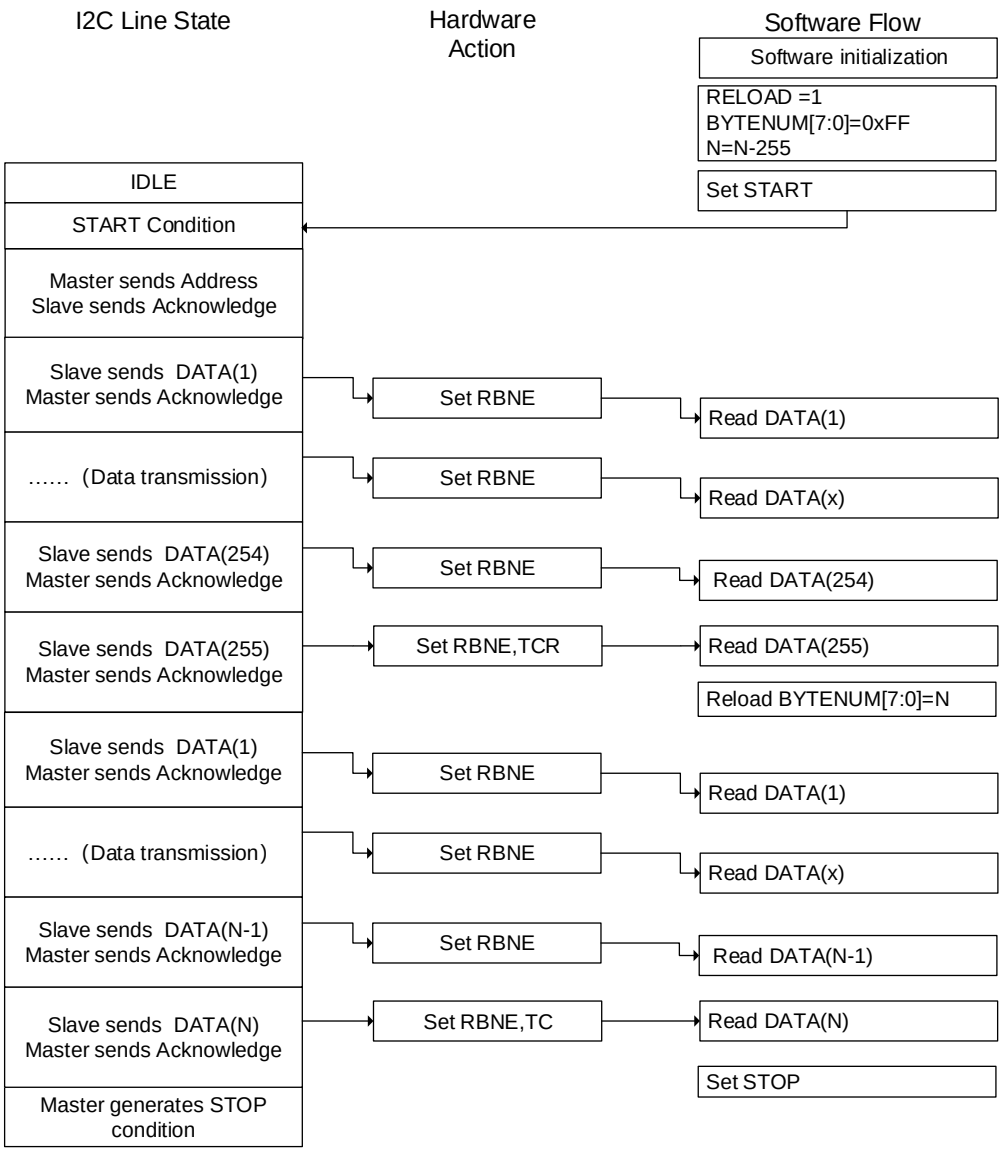


Figure 17-21. Programming model for master receiving (N>255)



17.3.9. 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 both those protocols there is a very useful distinction made between a System Host and all the other devices in the system that can have the names and functions of masters or slaves.

Received command and data acknowledge control

A SMBus receiver must be able to NACK each received command or data. In order to allow the ACK control in slave mode, 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. The host will acknowledge the SMBus Host address. When this protocol is used, the device acts as a master and the host as a slave.

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.

- $t_{TIMEOUT}$ check

In order to enable the $t_{TIMEOUT}$, the BUSTOA[11:0] must be programmed with the timer to check the $t_{TIMEOUT}$ parameter. To detect SCL low level timeout, the TOIDLE bit must be configured to "0". Then set TOEN in the I2C_TIMEOUT register to enable the timer. If the low level time of SCL is greater than (BUSTOA + 1) x 2048 x t_{I2CCLK}, the TIMEOUT flag is set in the I2C_STAT register.

Note: After the TOEN bit is set, the BUSTOA[11:0] and the TOIDLE bit cannot be changed.

- $t_{LOW:SEXT}$ and $t_{LOW:MEXT}$ check

By configuring a 12-bit BUSTOB timer, the $t_{LOW:SEXT}$ of the slave and the $t_{LOW:MEXT}$ of the master can be checked. Since the standard only specifies the maximum value, the same value can be chose for the both. Then enable the timer by setting the EXTOEN bit in the I2C_TIMEOUT register. If the SCL stretching time of the SMBus peripheral is longer than (BUSTOB + 1) x 2048 x 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.

Note: After the TOEN bit is set, the BUSTOB[11:0] cannot be changed.

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).

Setting the PECEN bit in the I2C_CTL0 register will enable the PEC calculation. The PEC transfer can be managed by the hardware byte counter: BYTENUM[7:0] in the I2C_CTL1 register. The PECEN bit must be configured before enabling the I2C.

Since the PEC transmission is managed by hardware byte counter, SBCTL bit must be set when connecting SMBus in slave mode. 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.

Table 17-5. SMBus with PEC configuration

Mode	SBCTL bit	RELOAD bit	AUTOEND bit	PECTRANS bit
Master Tx/Rx BYTENUM + PEC+ STOP	x	0	1	1
Master Tx/Rx BYTENUM + PEC + RESTART	x	0	0	1
Slave Tx/Rx with PEC	1	0	x	1

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. Only the device(s) which pulled SMBALERT# low will acknowledge the Alert Response Address. When SMBHAEN is 0, it is configured as a slave device, the SMBA pin is 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 is 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 ALERT line is considered high even if the external 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 peripheral supports hardware bus idle detection.

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 set TOEN in the I2C_TIMEOUT register to enable the timer. If the high level time of both SCL and SDA is greater than (BUSTOA + 1) x 2048 x t_{I2CCLK}, the TIMEOUT flag is set in the I2C_STAT register.

Note: After the TOEN bit is set, the BUSTOA[11:0] bit and the TOIDLE bit cannot be changed.

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 (0b0001100) is enabled by setting the SMBALTEN bit in the I2C_CTL0 register.

17.3.10. SMBus mode

SMBus Master Transmitter and Slave Receiver

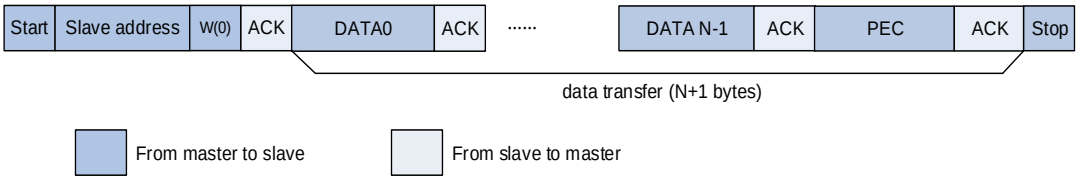
The PEC in SMBus master mode can be transmitted by setting the PECTRANS bit before setting the START bit, and the number of bytes in the BYTENUM[7:0] field must be configured. In this case, the total number of transmissions when TI interrupt occur is BYTENUM-1. So if

BYTENUM=0x1 and PECTRANS bit is set at the same time, the contents of the I2C_PEC register are automatically transferred. If the automatic end mode is selected (AUTOEND=1), the SMBus master automatically sends the STOP condition after the PEC byte. If the automatic end mode is not selected (AUTOEND=0), the SMBus master can send a RESTART condition after the PEC. The I2C_PEC register content 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 to enable the RELOAD mode. 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 contents of the I2C_PEC register. If the comparison does not match, the NACK is automatically generated; if the comparison matches, the ACK is automatically generated, regardless of the ACK bit value. When PEC byte is received, it is also copied into the I2C_RDATA register like other data, and RBNE flag will be set. If the ERRIE bit in I2C_CTL0 register is 1, when PEC 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 17-22. 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 chose (AUTOEND=1). Before sending a START condition 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 contents of the I2C_PEC register automatically. A NACK is respond to the PEC byte before STOP condition.

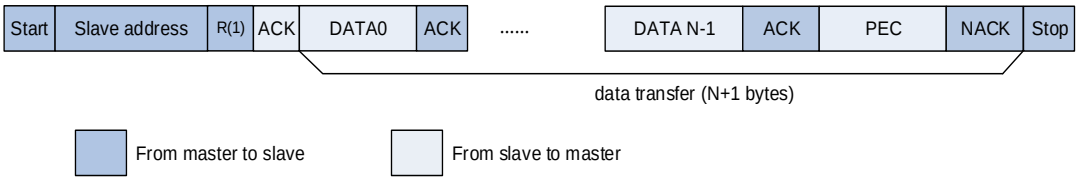
If the SMBus master receiver is required to generate a RESTART signal after receiving PEC byte, the software mode (AUTOEND = 0) must be selected. Before sending a START signal to the bus, PECTRANS bit must be set and slave addresses must be programmed. After receiving BYTENUM-1 data, the next received byte will be compared with the contents of 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 contents of the I2C_PEC register will be transmitted automatically.

Note: After the RELOAD bit is set, the PECTRANS cannot be changed.

Figure 17-23. SMBus Master Receiver and Slave Transmitter communication flow



17.3.11. Wakeup from Deep-sleep mode

When the address of I2C matches correctly, it can wake up from MCU Deep-sleep mode (APB clock is off). In order to wake up from Deep-sleep mode, WUEN bit must be set in the I2C_CTL0 register and the IRC16M must be selected as the clock source for I2CCLK. During Deep-sleep mode, the IRC16M is switched off. The I2C interface switches the IRC16M on, and stretches SCL low until IRC16M is woken up when a START is detected. Then the IRC16M 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, IRC16M 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, IRC16M will not switched on after receiving start signal. When wakeup from Deep-sleep mode is enabled, the digital filter must be disabled and the SS bit in I2C_CTL0 must be cleared. Before entering Deep-sleep mode (I2CEN=0), the I2C peripheral must be disabled if wakeup from Deep-sleep mode is disabled (WUEN =0).

Note: Only address match of I2C0 can wakeup MCU from Deep-sleep mode.

17.3.12. 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.

17.3.13. I2C error and interrupts

The I2C error flags are listed in *Table 17-6. I2C error flags*.

Table 17-6. 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 17-7. I2C interrupt events*.

Table 17-7. I2C interrupt events

Interrupt event	Event flag	Enable control bit
I2C_RDATA is not empty during receiving	RBNE	RBNEIE
Transmit interrupt	TI	TIE
STOP condition 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	

17.3.14. I2C debug mode

When the microcontroller enters the debug mode (RISC-V core halted), the SMBus timeout either continues to work normally or stops, depending on the I2Cx_HOLD configuration bits in the DBG module.

17.4. Register definition

I2C0 base address: 0x4000 5400

I2C1 base address: 0x4000 5800

17.4.1. Control register 0 (I2C_CTL0)

Address offset: 0x00

Reset value: 0x0000 0000

This register can 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
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)
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.
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.
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 Deep-sleep mode enable 0: Wakeup from Deep-sleep mode disable. 1: Wakeup from Deep-sleep mode enable. **Note:** WUEN can be set only when DNF[3:0] = 0000. This bit is reserved in I2C1.
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 These bits are used to configure the digital noise filter on SDA and SCL input. The digital filter will filter spikes with a length of up to DNF[3:0] * t_{I2CCLK} 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} 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	Not acknowledge received interrupt enable 0: Not acknowledge (NACK) received interrupt is disabled 1: Not acknowledge (NACK) received interrupt is enabled
3	ADDMIE	Address match interrupt enable in slave mode 0: Address match (ADDSEND) interrupt is disabled 1: Address match (ADDSEND) interrupt 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

17.4.2. Control register 1 (I2C_CTL1)

Address offset: 0x04

Reset value: 0x0000 0000

This register can 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 condition 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.
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 condition 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. 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:** These bits should not be modified when the START bit is set.
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 condition 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 condition on I2C bus This bit is set by software and cleared by hardware when I2CEN=0 or STOP condition is detected. 0: STOP will not be sent 1: STOP will be sent
13	START	Generate a START condition 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

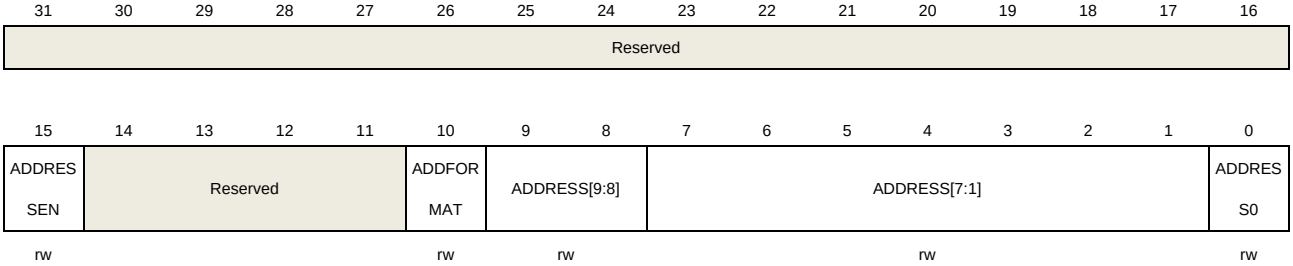
		1: START will be sent
12	HEAD10R	10-bit address header executes read direction only in master receive mode 0: The 10 bit master receive address sequence is START + header of 10-bit address (write) + slave address byte 2 + RESTART + header of 10-bit address (read). 1: The 10 bit master receive address sequence is RESTART + header of 10-bit address (read). **Note:** When the START bit is set, this bit can not be changed.
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, this bit can not be modified.
10	TRDIR	Transfer direction in master mode 0: Master transmit 1: Master receive **Note:** When the START bit is set, this bit can not be modified.
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 have 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, the bit filed can not be modified.

17.4.3. Slave address register 0 (I2C_SADDR0)

Address offset: 0x08

Reset value: 0x0000 0000

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



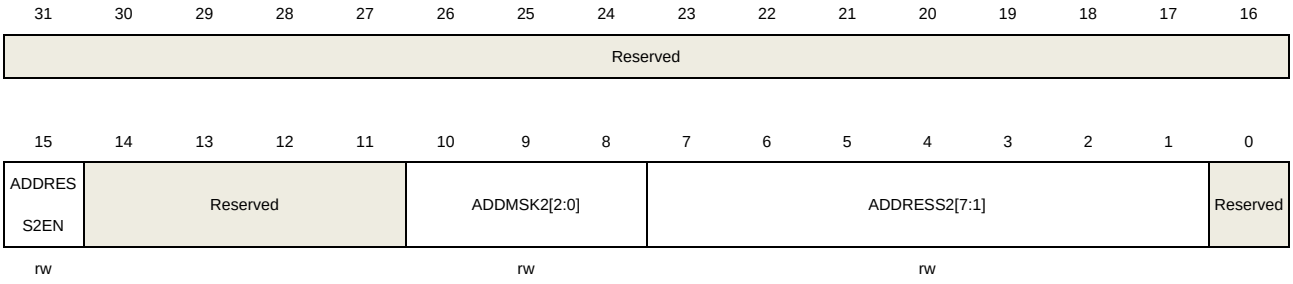
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	ADDRESSEN	I2C address enable 0: I2C address disable. 1: I2C address enable.
14:11	Reserved	Must be kept at reset value.
10	ADDFORMAT	Address mode for the I2C slave 0: 7-bit address 1: 10-bit address **Note:** When ADDRESSEN is set, this bit should not be written.
9:8	ADDRESS[9:8]	Highest two bits of a 10-bit address **Note:** When ADDRESSEN is set, this bit should not be written.
7:1	ADDRESS[7:1]	7-bit address or bits 7:1 of a 10-bit address **Note:** When ADDRESSEN is set, this bit should not be written.
0	ADDRESS0	Bit 0 of a 10-bit address **Note:** When ADDRESSEN is set, this bit should not be written.

17.4.4. Slave address register 1 (I2C_SADDR1)

Address offset: 0x0C

Reset value: 0x0000 0000

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



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

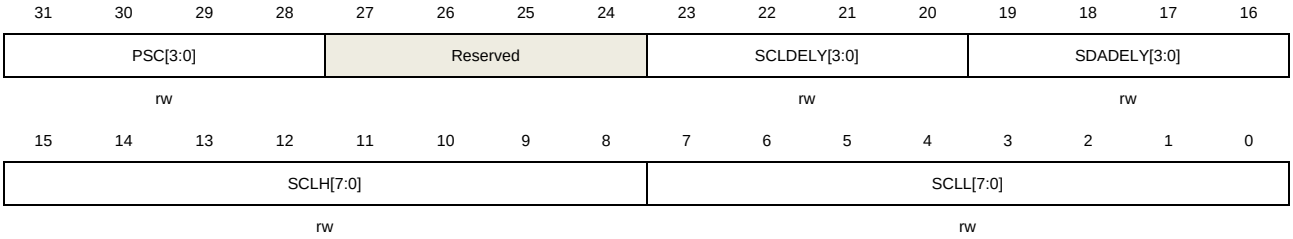
		Defines which bits of ADDRESS2[7:1] are compared with an incoming address byte, and which bits are masked (don't care). 000: No mask, all the bits must be compared. n(001~110): ADDRESS2[n:0] is masked. Only ADDRESS2[7:n+1] are compared. 111: ADDRESS2[7:1] are masked. All 7-bit received addresses are 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:** When ADDRESS2EN is set, these bits should not be written.
0	Reserved	Must be kept at reset value.

17.4.5. Timing register (I2C_TIMING)

Address offset: 0x10

Reset value: 0x0000 0000

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



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) x 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) x 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. $t_{SDADELY}$= SDADELY x t_{PSC}
15:8	SCLH[7:0]	SCL high period SCL high period can be generated by configuring these bits. t_{SCLH} = (SCLH+1) x 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) x t_{PSC} **Note:** These bits can only be used in master mode.

17.4.6. Timeout register (I2C_TIMEOUT)

Address offset: 0x14

Reset value: 0x0000 0000

This register can 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) x 2048 x 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	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) x 2048 x 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	Bus timeout A When TOIDLE=0, $t_{TIMEOUT}$= (BUSTOA +1) x 2048 x t_{I2CCLK} When TOIDLE=1, t_{IDLE} = (BUSTOA +1) x 4 x t_{I2CCLK} **Note:** These bits can be written only when TOEN =0.

17.4.7. Status register (I2C_STAT)

Address offset: 0x18

Reset value: 0x0000 0001

This register can 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 condition is detected and cleared by hardware after a STOP condition. 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
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
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.
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 condition 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 condition detected in slave mode This flag is set by hardware when a STOP condition is detected on the bus. It is cleared by software by setting STPDETC bit and cleared by hardware when I2CEN=0. 0: STOP condition is not detected. 1: STOP condition 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

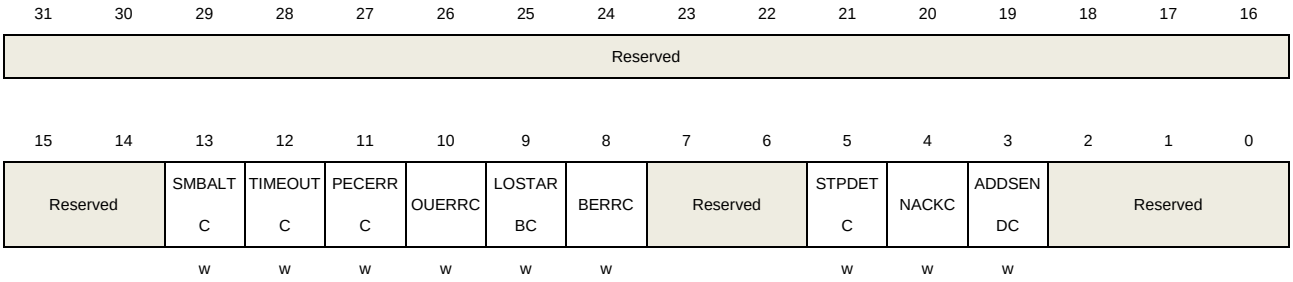
1: I2C_TDATA is empty

17.4.8. Status clear register (I2C_STATC)

Address offset: 0x1C

Reset value: 0x0000 0000

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



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
12	TIMEOUTC	TIMEOUT flag clear. Software can clear the TIMEOUT bit of I2C_STAT by writing 1 to this bit
11	PECERRC	PEC error flag clear. Software can clear the PECERR bit of I2C_STAT by writing 1 to this bit
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 write 1 to this bit
4	NACKC	Not Acknowledge flag clear Software can clear the NACK bit of I2C_STAT by write 1 to this bit
3	ADDSENDC	ADDSEND flag clear

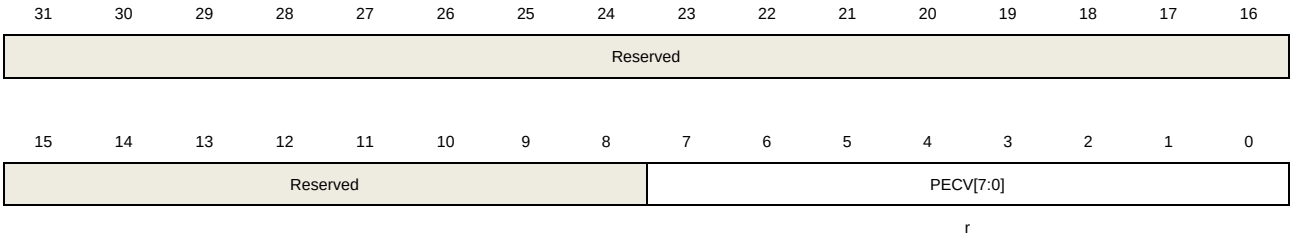
		Software can clear the ADDSEND bit of I2C_STAT by write 1 to this bit
2:0	Reserved	Must be kept at reset value.

17.4.9. PEC register (I2C_PEC)

Address offset: 0x20

Reset value: 0x0000 0000

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



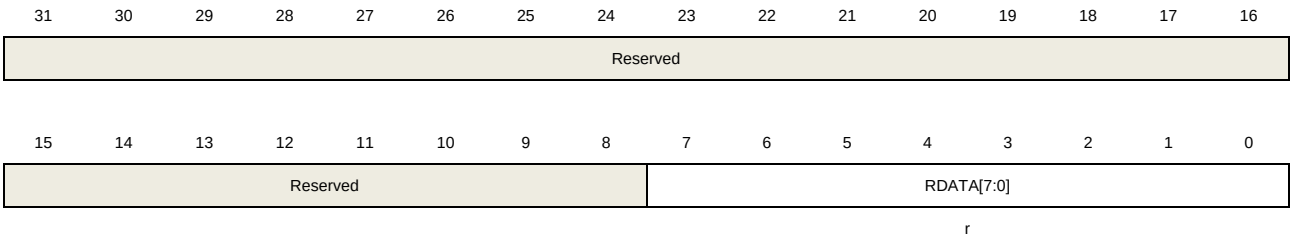
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	PECV[7:0]	Packet Error Checking Value that calculated by hardware when PEC is enabled. PECV is cleared by hardware when I2CEN = 0.

17.4.10. Receive data register (I2C_RDATA)

Address offset: 0x24

Reset value: 0x0000 0000

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



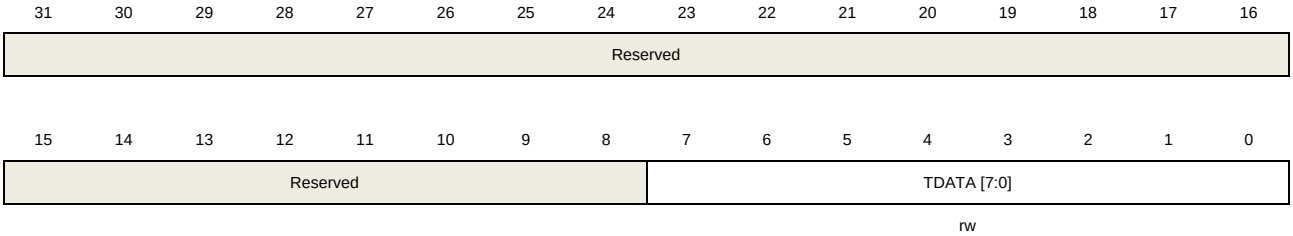
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	RDATA[7:0]	Receive data value

17.4.11. Transmit data register (I2C_TDATA)

Address offset: 0x28

Reset value: 0x0000 0000

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



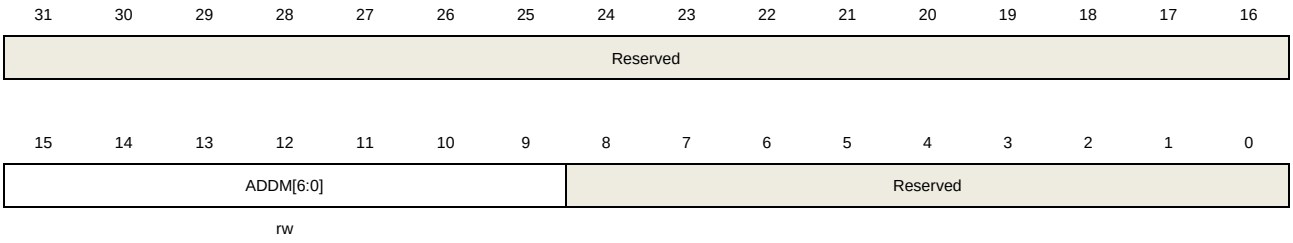
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	TDATA[7:0]	Transmit data value

17.4.12. Control register 2 (I2C_CTL2)

Address offset: 0x90

Reset value: 0x0000 0000

This register can 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.

18. Serial peripheral interface (SPI)

18.1. Overview

The SPI module can communicate with external devices using the SPI 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.

18.2. Characteristics

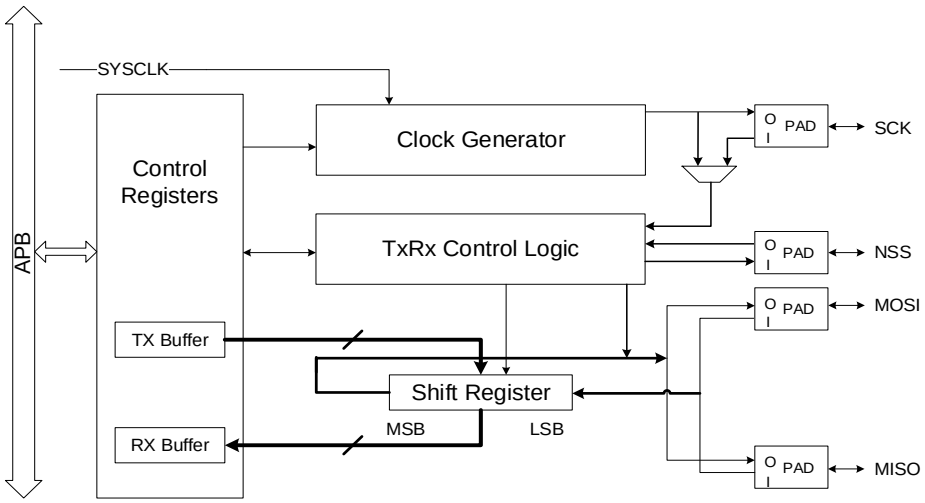
18.2.1. SPI characteristics

- Master or slave operation with full-duplex, half-duplex or simplex mode.
- Separate transmit and receive buffer, 16 bits wide.
- Data frame size can be 8 or 16 bits.
- Bit order can be LSB or MSB.
- Software and hardware NSS management.
- Hardware CRC calculation, transmission and checking.
- Transmission and reception using DMA.
- SPI TI mode supported.

18.3. SPI function overview

18.3.1. SPI block diagram

Figure 18-1. Block diagram of SPI



18.3.2. SPI signal description

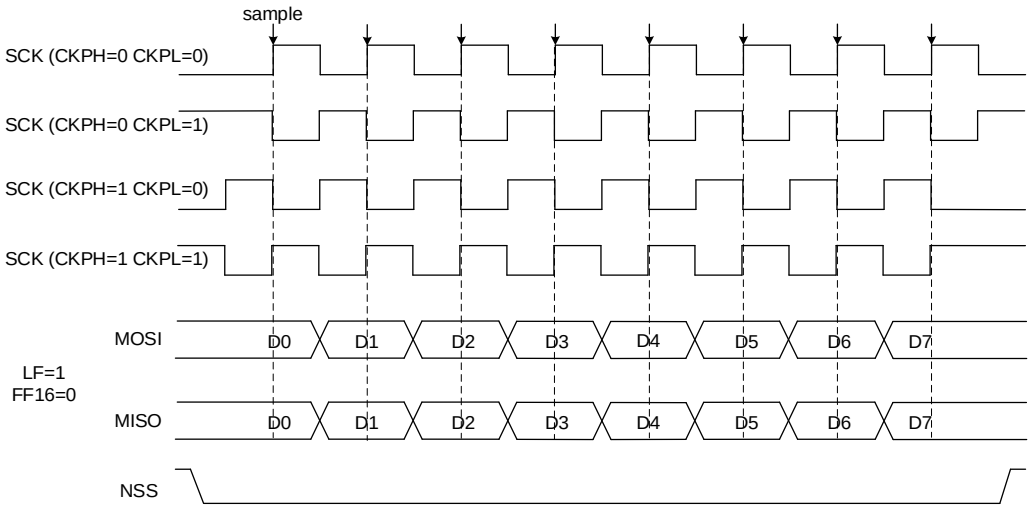
Table 18-1. SPI signal description

Pin name	Direction	Description
SCK	I/O	Master: SPI clock output Slave: SPI clock input
MISO	I/O	Master: Data reception line Slave: Data transmission line Master with bidirectional mode: Not used Slave with bidirectional mode: Data transmission and reception line.
MOSI	I/O	Master: Data transmission line Slave: Data reception line Master with bidirectional mode: Data transmission and reception line. Slave with bidirectional mode: Not used
NSS	I/O	Software NSS mode: Not used Master in hardware NSS mode: when NSSDRV=1, it is NSS output, suitable for single master application; when NSSDRV=0, it is NSS input, suitable for multi-master application. Slave in hardware NSS mode: NSS input, as a chip select signal for slave.

18.3.3. 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 18-2. SPI timing diagram in normal mode



In normal mode, the length of data is configured by the FF16 bit in the SPI_CTL0 register. Data length is 16 bits if FF16=1, otherwise is 8 bits.

Data order is configured by the LF bit in SPI_CTL0 register, and SPI will first send the LSB first if LF=1, or the MSB first if LF=0. The data order is fixed to MSB first in TI mode.

18.3.4. NSS function

Slave mode

When slave mode is configured (MSTMOD=0), SPI gets NSS level from NSS pin in hardware NSS mode (SWNSSEN = 0) or from SWNSS bit in software NSS mode (SWNSSEN = 1), and SPI transmits/receives data only when NSS level is low. In software NSS mode, NSS pin is not used.

Table 18-2. 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 18-3. 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.

18.3.5. SPI operating modes

Table 18-4. SPI operating modes

Mode	Description	Register configuration	Data pin usage
MFD	Master full-duplex	MSTMOD = 1 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Transmission MISO: Reception
MTU	Master transmission with unidirectional connection	MSTMOD = 1 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Transmission MISO: Not used

Mode	Description	Register configuration	Data pin usage
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 BDEN = 1 BDOEN = 1	MOSI: Transmission MISO: Not used
MRB	Master reception with bidirectional connection	MSTMOD = 1 RO = 0 BDEN = 1 BDOEN = 0	MOSI: Reception MISO: Not used
SFD	Slave full-duplex	MSTMOD = 0 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Reception MISO: Transmission
STU	Slave transmission with unidirectional connection	MSTMOD = 0 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Not used MISO: Transmission
SRU	Slave reception with unidirectional connection	MSTMOD = 0 RO = 1 BDEN = 0 BDOEN: Don't care	MOSI: Reception MISO: Not used
STB	Slave transmission with bidirectional connection	MSTMOD = 0 RO = 0 BDEN = 1 BDOEN = 1	MOSI: Not used MISO: Transmission
SRB	Slave reception with bidirectional connection	MSTMOD = 0 RO = 0 BDEN = 1 BDOEN = 0	MOSI: Not used MISO: Reception

Figure 18-3. A typical full-duplex connection

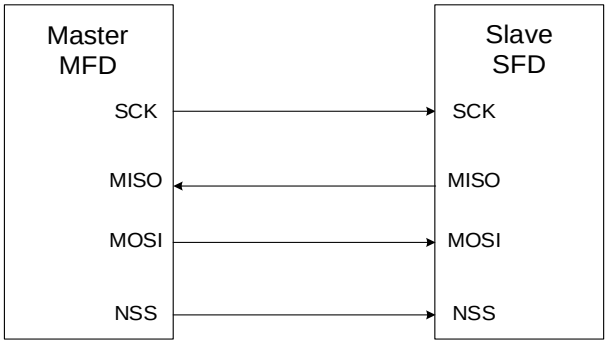


Figure 18-4. A typical simplex connection (Master: Receive, Slave: Transmit)

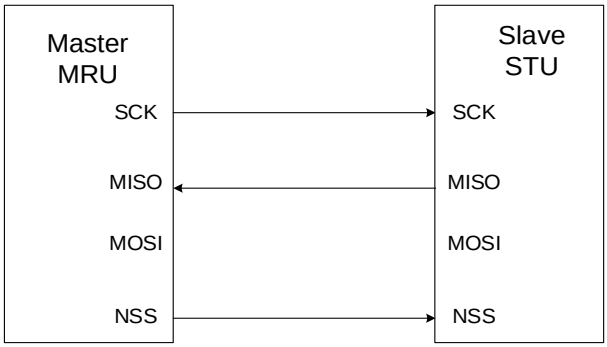


Figure 18-5. A typical simplex connection (Master: Transmit only, Slave: Receive)

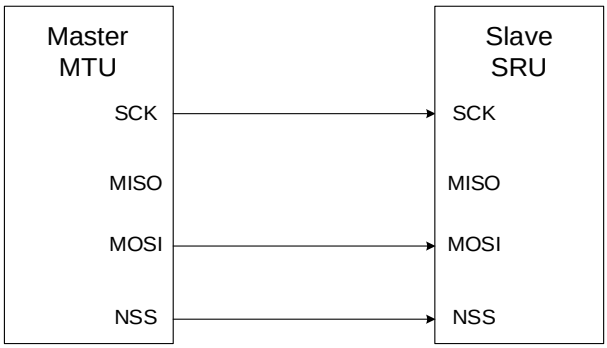
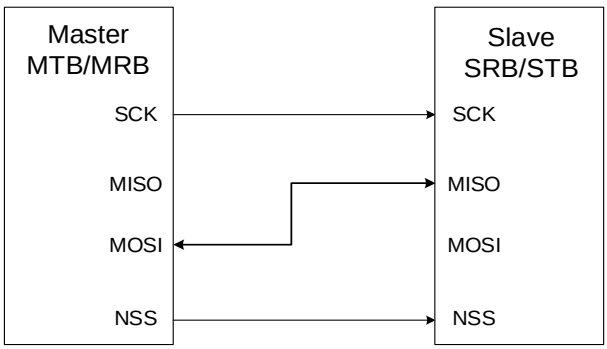


Figure 18-6. A typical bidirectional connection



SPI initialization sequence

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. Program data format (FF16 bit in the SPI_CTL0 register).
3. Program the clock timing register (CKPL and CKPH bits in the SPI_CTL0 register).
4. Program the frame format (LF bit in the SPI_CTL0 register).
5. Program the NSS mode (SWNSSEN and NSSDRV bits in the SPI_CTL0 register) according to the application's demand as described above in ***NSS function*** section.
6. If TI mode is used, set TMOD bit in SPI_CTL1 register, otherwise, ignore this step.
7. Configure MSTMOD, RO, BDEN and BDOEN depending on the operating modes described in ***SPI operating modes*** section.
8. Enable the SPI (set the SPIEN bit).

Note: CKPH, CKPL, MSTMOD, PSC[2:0], LF bits should not be changed during communication.

SPI basic transmission and reception sequence

Transmission sequence

After the initialization sequence, the SPI is enabled and stays at idle state. In master mode, the transmission starts when the application writes a data into the transmit buffer. In slave mode the transmission starts when SCK clock signal at SCK pin begins to toggle and NSS level is low, so application should ensure that data is already written into transmit buffer before the transmission starts in slave mode.

When SPI begins to send a data frame, it first loads this data frame from the data buffer to the shift register and then begins to transmit the loaded data frame, TBE (transmit buffer empty) flag is set after the first bit of this frame is transmitted. After TBE flag is set, which means the transmit buffer is empty, the application should write SPI_DATA register again if it has more data to transmit.

In master mode, software should write the next data into SPI_DATA register before the transmission of current data frame is completed if it desires to generate continuous transmission.

Reception sequence

After the last valid sample clock, the incoming data will be moved from shift register to the receive buffer and RBNE (receive buffer not empty) will be set. The application should read SPI_DATA register to get the received data and this will clear the RBNE flag automatically. In MRU and MRB modes, hardware continuously sends clock signal to receive the next data

frame, while in full-duplex master mode (MFD), hardware only receives the next data frame when the transmit buffer is not empty.

SPI operation sequence in different modes (Not TI 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 regardless of the RBNE and OVRE bits.

The master reception mode (MRU or MRB) is different from the reception sequence of full-duplex mode. In MRU or MRB mode, after SPI is enabled, the SPI continuously generates SCK until the SPI is disabled. So the application should ignore the TBE flag and read out reception buffer in time after the RBNE flag is set, otherwise a data overrun fault will occur.

The slave reception mode (SRU or SRB) is similar to the reception sequence of full-duplex mode regardless of the TBE flag.

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 18-7. Timing diagram of TI master mode with discontinuous transfer

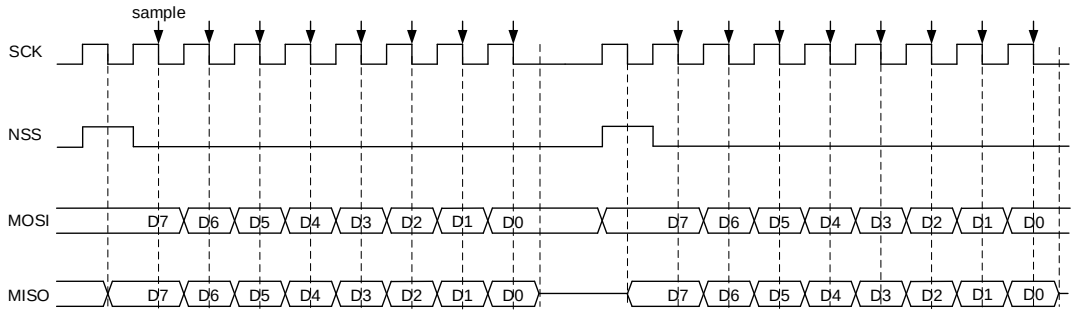
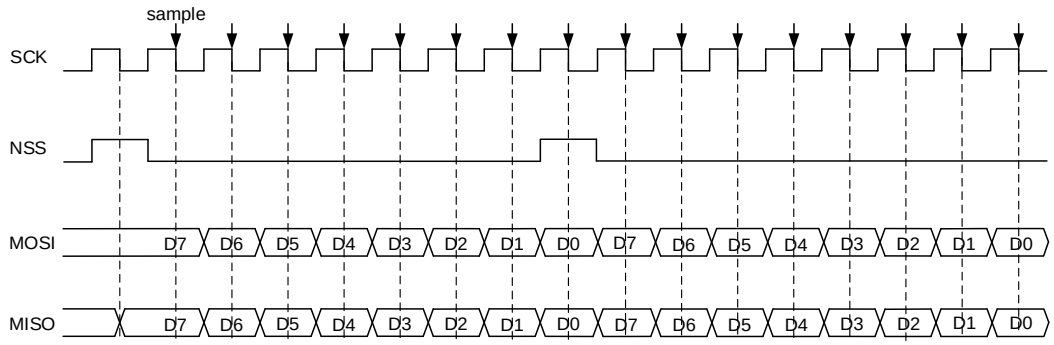
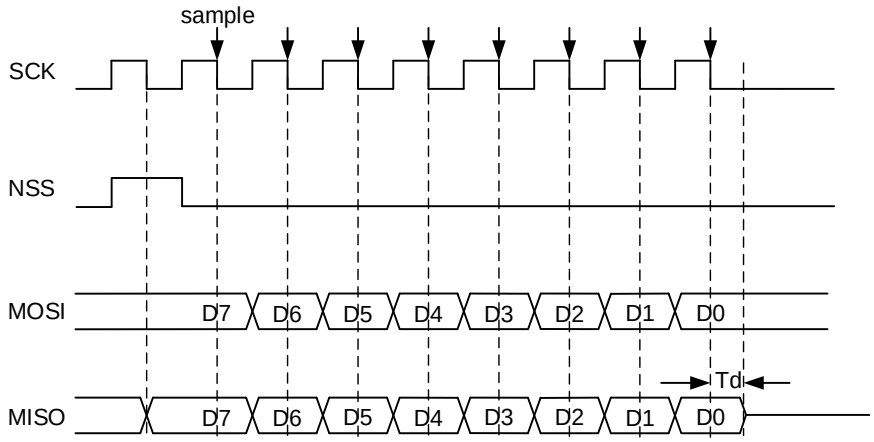


Figure 18-8. Timing diagram of TI master mode with continuous transfer



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 18-9. Timing diagram of TI slave mode



In slave TI mode, after the last rising edge of SCK in transfer, the slave begins to transmit the LSB bit of the last data byte, and after a half-bit time, the master begins to sample the line. To make sure that the master samples the right value, the slave should continue to drive this bit after the falling sample edge of SCK for a period of time before releasing the pin. This time is called T_d. T_d is decided by PSC[2:0] bits in SPI_CTL0 register.

$$T_d = \frac{T_{bit}}{2} + 5 * T_{pclk} \quad (18\text{-}1)$$

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.

SPI disabling sequence

Different sequences are used to disable the SPI in different operation modes.

MFD SFD

Wait for the last RBNE flag and then receive the last data. Confirm that TBE=1 and TRANS=0. At last, disable the SPI by clearing SPIEN bit.

MTU MTB STU STB

Write the last data into SPI_DATA and wait until the TBE flag is set and then wait until the TRANS flag is cleared. Disable the SPI by clearing SPIEN bit.

MRU MRB

After getting the second last RBNE flag, read out this data and delay for a SCK clock time and then, disable the SPI by clearing SPIEN bit. Wait until the last RBNE flag is set and read out the last data.

SRU SRB

Application can disable the SPI when it doesn't want to receive data, and then wait until the TRANS=0 to ensure the ongoing transfer completes.

TI mode

The disabling sequence of TI mode is the same as the sequences described above.

18.3.6. DMA function

The DMA frees the application from data writing and reading process during transfer, to improve the system efficiency.

DMA function in SPI is enabled by setting DMATEN and DMAREN bits in SPI_CTL1 register. To use DMA function, application should first correctly configure DMA modules, then configure SPI module according to the initialization sequence, at last enable SPI.

After being enabled, if DMATEN is set, SPI will generate a DMA request each time when TBE=1, then DMA will acknowledge to this request and write data into the SPI_DATA register automatically. If DMAREN is set, SPI will generate a DMA request each time when RBNE=1, then DMA will acknowledge to this request and read data from the SPI_DATA register automatically.

18.3.7. CRC function

There are two CRC calculators in SPI: one for transmission and the other for reception. The CRC calculation uses the polynomial defined in SPI_CRCPOLY register.

Application can enable the CRC function by setting CRCEN bit in SPI_CTL0 register. The CRC calculators continuously calculate CRC for each bit transmitted and received on lines, and the calculated CRC values can be read from SPI_TCRC and SPI_RCRC registers.

To transmit the calculated CRC value, application should set the CRCNT bit in SPI_CTL0 register after the last data is written to the transmit buffer. In full-duplex mode (MFD or SFD), when the SPI transmits a CRC and prepares to check the received CRC value, the SPI treats the incoming data as a CRC value. In reception mode (MRB, MRU, SRU and SRB), the application should set the CRCNT bit after the second last data frame is received. When CRC checking fails, the CRCERR flag will be set.

If 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.

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.

18.3.8. SPI interrupts

Status flags

- Transmit buffer empty flag (TBE)

This bit is set when the transmit buffer is empty, the software can write the next data to the transmit buffer by writing the SPI_DATA register.

- Receive buffer not empty flag (RBNE)

This bit is set when receive buffer is not empty, which means that one data is received and stored in the receive buffer, and software can read the data by reading the SPI_DATA register.

- SPI transmitting ongoing flag (TRANS)

TRANS is a status flag to indicate whether the transfer is ongoing or not. It is set and cleared by hardware and not controlled by software. This flag doesn't generate any interrupt.

Error flags

- Configuration fault error (CONFERR)

CONFERR is an error flag in master mode. In NSS hardware mode and if the NSSDRV is not enabled, the CONFERR is set when the NSS pin is pulled low. In NSS software mode, the CONFERR is set when the SWNSS bit is 0. When the CONFERR is set, the SPIEN bit and the MSTMOD bit are cleared by hardware, the SPI is disabled and the device is forced into slave mode.

The SPIEN and MSTMOD bits are write protected until the CONFERR is cleared. The CONFERR bit of the slave cannot be set. In a multi-master configuration, the device can be in slave mode with CONFERR bit set, which means there might have been a multi-master conflict for system control.

- Rx overrun error (RXORERR)

The RXORERR bit is set if a data is received when the RBNE is set. That means, the last data has not been read out and the newly incoming data is received. The receive buffer contents won't be covered with the newly incoming data, so the newly incoming data is lost.

- Format error (FERR)

In slave TI mode, the slave also monitors the NSS signal and set an error flag if it detects an

incorrect NSS behavior, for example: toggles at the middle bit of a byte.

- CRC error (CRCERR)

When the CRCEN bit is set, the CRC calculation result of the received data in the SPI_RCRC register is compared with the received CRC value after the last data, the CRCERR is set when they are different.

Table 18-5. SPI interrupt requests

Flag	Description	Clear method	Interrupt enable bit
TBE	Transmit buffer empty	Write SPI_DATA register.	TBEIE
RBNE	Receive buffer not empty	Read SPI_DATA register.	RBNEIE
CONFERR	Configuration fault error	Read or write SPI_STAT register, then write SPI_CTL0 register.	ERRIE
RXORERR	Rx overrun error	Read SPI_DATA register, then read SPI_STAT register.	
CRCERR	CRC error	Write 0 to CRCERR bit	
FERR	TI Mode Format Error	Write 0 to FERR bit	

18.4. Register definition

SPI base address: 0x4001 3000

18.4.1. Control register 0 (SPI_CTL0)

Address offset: 0x00
Reset value: 0x0000 0000

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BDEN	BDOEN	CRCEN	CRCNT	FF16	RO	SWNSSEN	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 information transfers between the MOSI pin in master and the MISO pin in slave.
14	BDOEN	Bidirectional transmit output enable When BDEN is set, this bit determines the direction of transfer. 0: Work in receive-only mode 1: Work in transmit-only mode
13	CRCEN	CRC calculation enable 0: CRC calculation is disabled 1: CRC calculation is enabled
12	CRCNT	CRC next transfer 0: Next transfer is data 1: Next transfer is CRC value (TCRC) When the transfer is managed by DMA, CRC value is transferred by hardware. This bit should be cleared. In full-duplex or transmit-only mode, set this bit after the last data is written to SPI_DATA register. In receive-only mode, set this bit after the second last data is received.
11	FF16	Data frame format 0: 8-bit data frame format

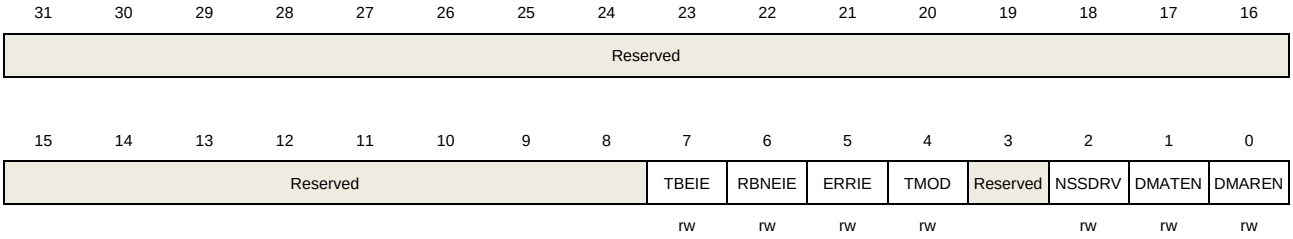
		1: 16-bit data frame format
10	RO	Receive only mode When BDEN is cleared, this bit determines the direction of transfer. 0: Full-duplex mode 1: Receive-only mode
9	SWNSSEN	NSS software mode enable 0: NSS hardware mode. The NSS level depends on NSS pin. 1: NSS software mode. The NSS level depends on SWNSS bit. This bit has no meaning in SPI TI mode.
8	SWNSS	NSS pin selection in NSS software mode 0: NSS pin is pulled low 1: NSS pin is pulled high This bit effects only when the SWNSSEN bit is set. This bit has no meaning in SPI TI mode.
7	LF	LSB first mode 0: Transmit MSB first 1: Transmit LSB first This bit has no meaning in SPI TI mode.
6	SPIEN	SPI enable 0: SPI peripheral is disabled 1: SPI peripheral is enabled
5:3	PSC[2:0]	Master clock prescaler selection 000: PCLK/2 100: PCLK/32 001: PCLK/4 101: PCLK/64 010: PCLK/8 110: PCLK/128 011: PCLK/16 111: PCLK/256 PCLK means PCLK2.
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

18.4.2. Control register 1 (SPI_CTL1)

Address offset: 0x04

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	TBEIE	Transmit buffer empty interrupt enable 0: TBE interrupt is disabled. 1: TBE interrupt is enabled. An interrupt is generated when the TBE bit is set.
6	RBNEIE	Receive buffer not empty interrupt enable 0: RBNE interrupt is disabled. 1: RBNE interrupt is enabled. An interrupt is generated when the RBNE bit is set.
5	ERRIE	Errors interrupt enable 0: Error interrupt is disabled. 1: Error interrupt is enabled. An interrupt is generated when the CRCERR bit, the CONFERR bit, or the RXORERR bit is set.
4	TMOD	SPI TI mode enable 0: SPI TI mode disabled. 1: SPI TI mode enabled.
3	Reserved	Must be kept at reset value.
2	NSSDRV	Drive NSS output 0: NSS output is disabled. 1: NSS output is enabled. If the NSS pin is configured as output, the NSS pin is pulled low in master mode when SPI is enabled. If the NSS pin is configured as input, the NSS pin should be pulled high in master mode, and this bit has no effect.
1	DMATEN	Transmit buffer DMA enable 0: Transmit buffer DMA is disabled. 1: Transmit buffer DMA is enabled, when the TBE bit in SPI_STAT is set, there will be a DMA request on corresponding DMA channel.

0	DMAREN	Receive buffer DMA enable 0: Receive buffer DMA is disabled. 1: Receive buffer DMA is enabled, when the RBNE bit in SPI_STAT is set, there will be a DMA request on corresponding DMA channel.

18.4.3. Status register (SPI_STAT)

Address offset: 0x08

Reset value: 0x0000 0002

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							FERR	TRANS	RXORERR	CONFERR	CRCERR	Reserved		TBE	RBNE
							rc_w0	r	r	r	rc_w0			r	r

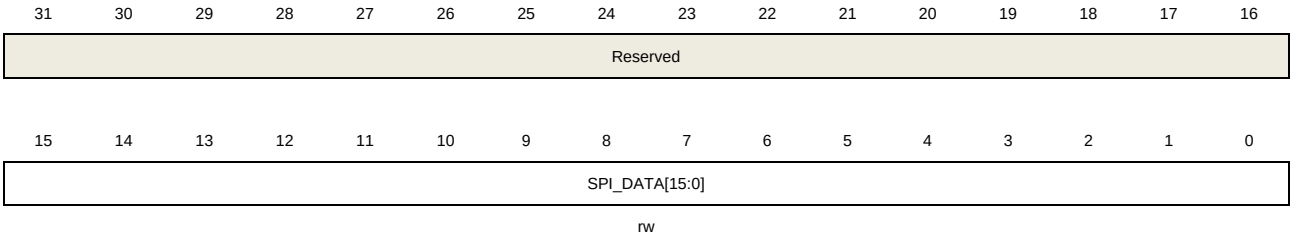
Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	FERR	Format error SPI TI Mode: 0: No TI mode format error 1: TI mode format error occurs This bit is set by hardware and cleared by writing 0.
7	TRANS	Transmitting ongoing bit 0: SPI is idle. 1: SPI is currently transmitting and/or 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 or write operation on the SPI_STAT register followed by a write access to the SPI_CTL0 register.
4	CRCERR	SPI CRC error bit

		0: The SPI_RCRC value is equal to the received CRC data at last. 1: The SPI_RCRC value is not equal to the received CRC data at last. This bit is set by hardware and cleared by writing 0.
3:2	Reserved	Must be kept at reset value.
1	TBE	Transmit buffer empty 0: Transmit buffer is not empty 1: Transmit buffer is empty
0	RBNE	Receive buffer not empty 0: Receive buffer is empty 1: Receive buffer is not empty

18.4.4. Data register (SPI_DATA)

Address offset: 0x0C
Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	SPI_DATA[15:0]	Data transfer register The hardware has two buffers, including transmit buffer and receive buffer. Write data to SPI_DATA will save the data to transmit buffer and read data from SPI_DATA will get the data from receive buffer. When the data frame format is set to 8-bit data, the SPI_DATA [15:8] is forced to 0 and the SPI_DATA[7:0] is used for transmission and reception, transmit buffer and receive buffer are 8-bit. If the data frame format is set to 16-bit data, the SPI_DATA[15:0] is used for transmission and reception, transmit buffer and receive buffer are 16-bit.

18.4.5. CRC polynomial register (SPI_CRCPOLY)

Address offset: 0x10
Reset value: 0x0000 0007

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

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

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CRCPOLY[15:0]	CRC polynomial value These bits contain the CRC polynomial and they are used for CRC calculation. The default value is 0007h.

18.4.6. RX CRC register (SPI_RCRC)

Address offset: 0x14
Reset value: 0x0000 0000

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

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

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

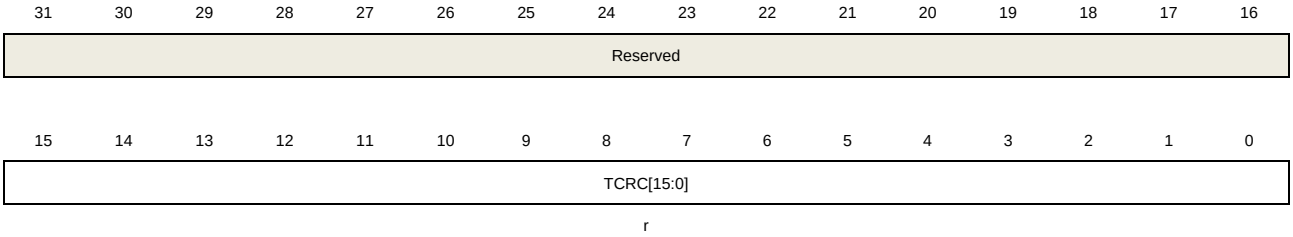
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	RCRC[15:0]	RX CRC value When the CRCEN bit of SPI_CTL0 is set, the hardware computes the CRC value of the received bytes and saves them in RCRC register. If the data frame format is set to 8-bit data, CRC calculation is based on CRC8 standard, and saves the value in RCRC[7:0], when the data frame format is set to 16-bit data, CRC calculation is based on CRC16 standard, and saves the value in RCRC[15:0]. The hardware computes the CRC value after each received bit, when the TRANS is set, a read to this register could return an intermediate value. This register is reset when the CRCEN bit in SPI_CTL0 register or the SPIxRST bit in RCU reset register is set.

18.4.7. TX CRC register (SPI_TCRC)

Address offset: 0x18

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	TCRC[15:0]	TX CRC value When the CRCEN bit of SPI_CTL0 is set, the hardware computes the CRC value of the transmitted bytes and saves them in TCRC register. If the data frame format is set to 8-bit data, CRC calculation is based on CRC8 standard, and saves the value in TCRC[7:0], when the data frame format is set to 16-bit data, CRC calculation is based on CRC16 standard, and saves the value in TCRC[15:0]. The hardware computes the CRC value after each transmitted bit, when the TRANS is set, a read to this register could return an intermediate value. The different frame formats (LF bit of the SPI_CTL0) will get different CRC values. This register is reset when the CRCEN bit in SPI_CTL0 register or the SPIxRST bit in RCU reset register is set.

19. Quad-SPI interface (QSPI)

19.1. Overview

The QSPI is a specialized interface that communicate with flash memories. This interface supports single, dual or quad SPI FLASH. It can operate in normal mode, read polling mode and memory map mode.

19.2. Characteristics

- Three functional modes: normal mode (address extend), read polling mode and memory map mode.
- Fully programmable command format for both normal mode and memory map mode.
- Integrated FIFO for transmission/reception.
- 8, 16, or 32-bit data accesses.
- DMA channel for normal mode.

19.3. Function overview

19.3.1. QSPI block diagram

6 signals are used to interface with an external flash memory. The description is shown in ***Table 19-1. QSPI signal description***.

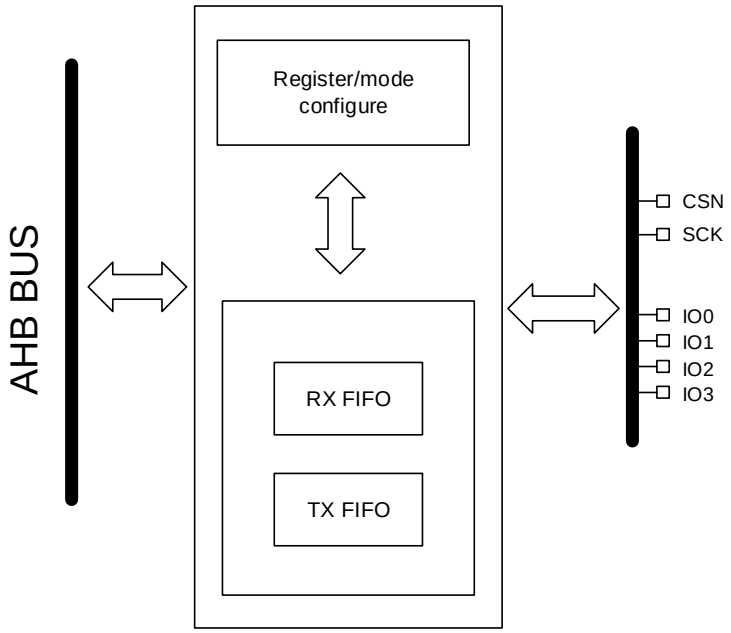
Table 19-1. QSPI signal description

Pin name	Direction	Description
CSN	O	chip select output (active low)
SCK	O	clock output
IO0/SO	I/O	single mode:data output dual mode: data intput or output quad mode: data intput or output
IO1/SI	I/O	single mode:data input dual mode: data intput or output quad mode: data intput or output
IO2	I/O	single mode: connect WP pin of flash, control “write protect” function dual mode: connect WP pin of flash, control “write protect” function quad mode: data intput or output
IO3	I/O	single mode: connect HOLD pin of flash, control “hold”

Pin name	Direction	Description
		function dual mode: connect HOLD pin of flash, control “hold” function quad mode: data intput or output

*Figure 19-1 QSPI diagram*shows the block diagram of the QSPI unit.

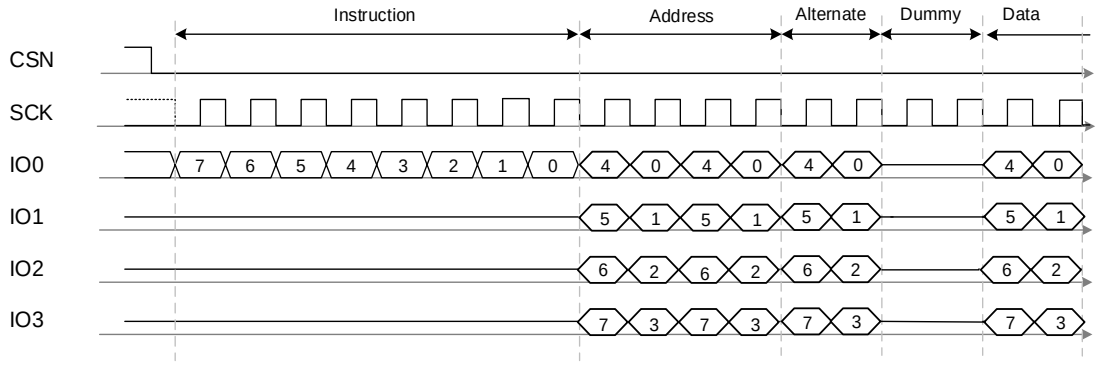
Figure 19-1 QSPI diagram



19.3.2. QSPI command format

The QSPI communicates with the flash memory using commands in various formats. There are totally 5 phases which can be included or not: instruction, address, alternate byte, dummy and data. Any phase can be skipped, but it must contain at least one of the instruction phase, address phase, alternate byte phase, and data phase. This is guaranteed by the software, and the hardware design has no protection method. In addition, the most-significant-bit always occupies the highest IO line number.

Figure 19-2 QSPI command format



The commands and the corresponding configuration are shown in ***Table 19-2. QSPI command description***.

Table 19-2. QSPI command description

Command	Send information	Configuration	Note
instruction	8-bit instruction	QSPI_TCFG register defines the instruction and signal line mode	-
address	1-4 bytes of address	QSPI_ADDR register defines the information of address. QSPI_TCFG register defines the number of address and signal line mode	-
Alternate byte	1-4 alternate-bytes	QSPI_ALTE register defines the information of alternate bytes, QSPI_TCFG register defines the number of alternate-bytes and signal line mode	-
dummy	0-31 cycles	QSPI_TCFG register defines the cycles and DATAMOD field (QSPI_TCFG register) defines the dummy signal line mode	given without any data being transferred for external flash, in order to wait flash prepare data
data	any number of bytes	In normal mode, QSPI_DTLEN register defines the number of bytes. DATAMOD filed (QSPI_TCFG register) defines the data signal line mode, and the configuration of DATAMOD = 00 must only be used in normal write mode	In memory map mode, the number of bytes to be transmitted is specified as single AHB bus access operation, these could be 8, 16 or 32 read/write access, corresponding to 1, 2, or 4 bytes.

Note: Signal line mode can be no command, single line mode, dual line mode, or quad line mode.

19.3.3. QSPI signal line modes

Each of the instruction, address, alternate-byte, or data phase can be configured separately into signal line modes by setting IMOD / ADDRMOD / ALTEMOD / DATAMOD.

Table 19-3. QSPI singnal line modes

Signal line modes		Single line mode	Dual line mode	Quad line mode
Configure filed	IMOD	01 or 00	10 or 00	11 or 00
	ADDRMOD			
	ALTEMOD			
	DATAMOD			
Pins	IO0 (SO)	Output	Input: data read (high impedance) output: all other phases	Input: data read (high impedance) Output: all other phases.
	IO1 (SI)	Input (high inpedance)		
	IO2	Output 0 (deactivate “write protect function”)		
	IO3	Output 1 (deactivate “hold” function)		
Description		In dummy phase when DATAMOD = 01, IO0 output, IO1 input (high inpedance)	In dummy phase when DATAMOD = 10, IO0 / IO1 are always high-impedance.	In dummy phase when DATAMOD = 11, IO0 / IO1 / IO2 / IO3 are always high-impedance.

IO2 / IO3 are used only in quad mode, and if none of the 5 phases is configured in quad mode, IO2 / IO3 will be released and can be used for other functions even when QSPI is enabled.

19.3.4. CSN and SCK

The default value of CSN is high, and it falls before a command begins and rises as soon as it finishes.

SCK output signal is a gate signal from internal sck, where the internal sck is present all the time.

CSN falls one SCK cycle before the first valid rising SCK edge, and rises on SCK cycle after the final valid rising SCK edge.

When the FIFO stays empty in a write command phase, or full in a read command phase, SCK will be stalled and keep low until the FIFO can work again. At this moment if the CSN is high, SCK will rises back up one half of a SCK cycle after the rising edge of the CSN.

19.4. Operating modes

The QSPI can operate in normal mode, read polling mode and memory map mode. In normal mode, all operations are performed depends on QSPI registers. In read polling mode, the values of status registers in external flash memory are periodically read and checked. In memory map mode, the external flash memory is mapped to the microcontroller address

space (range from 0x9000 0000 to 0x97FF FFFF) and can be accessed as an internal memory.

19.4.1. Normal mode

The write operation in normal mode is selected by configuring the FMOD[1:0] in QSPI_TCFG register to “00”. The data to be transmitted is written into QSPI_DATA. The read operation in normal mode is selected by setting the FMOD[1:0] in QSPI_TCFG register to “01”, and the data to be received is read from QSPI_DATA.

The DTLEN[31:0] in QSPI_DTLEN register defines the number of bytes to be transferred. If DTLEN is 0xFFFF FFFF, the number of bytes to be transferred is considered undefined, and the transmission continues until the memory size boundary is reached as specified by FMSZ[4:0] in QSPI_DCFG register. If DTLEN is 0xFFFF FFFF and FMSZ[4:0] is configured as 0x1F, the flash memory capacity is 4GB, the transmission will continue indefinitely until a request of abort is occurred or the QSPI is disabled.

The transfer complete flag TC will be set when the number of bytes programmed in DTLEN has been transferred. If the DTLEN is 0xFFFF FFFF, TC will be set when the transmitted/received byte number equals to the external memory size defined by FMSZ[4:0]. An interrupt is generated if TCIE and TC are both set. The TC bit is cleared by writing 1 to the TCC bit in QSPI_STATC register.

Initialize a command sequence

The command sequence starts immediately after the last information is provided by software according to communication requirement.

When neither address nor data are required, the sequence starts immediately after QSPI_TCFG has been accessed.

When address is required and no data is required, the sequence starts after QSPI_ADDR has been accessed.

When both address and data are required in normal write mode, the command sequence starts after QSPI_DATA has been accessed.

FIFO

A FIFO 16 bytes is implemented to transfer data. In normal write mode, the relationship between the AHB write access mode and the number of bytes add to FIFO is shown in ***Table 19-4. AHB write access mode and number of bytes add to FIFO***.

Table 19-4. AHB write access mode and number of bytes add to FIFO

AHB write access mode	Number of bytes add to FIFO
32-bit	4 bytes
16-bit	2 bytes
8-bit	1 byte

Note: When the AHB write access mode is 8-bit or 16-bit, the least significant bytes are valid in QSPI_DATA register.

FIFO threshold is defined by FTL[3:0] in QSPI_CTL register, in normal read mode, when the amount of bytes in the FIFO is equal or above the defined threshold, FIFO threshold flag FT in QSPI_STAT register will be set. FT is also set after data phase is completed if FIFO is not empty. In normal write mode, when the amount of the empty bytes in the FIFO is above the threshold, FT will be set.

An interrupt is generated if both FTIE and FT is set. If DMA is enabled, a DMA request is generated by FT, until this flag is cleared.

In normal read mode, when the FIFO becomes full, the QSPI temporarily stop SCK clock to avoid overrun. The reading sequence is not resumed until more than 4 bytes are available in FIFO.

19.4.2. Read polling mode

The read polling mode can be selected by configuring the FMOD[1:0] to “10”. In read polling mode, the QSPI periodically starts a read command with up to 4-bytes data. The received data can be bit-wise masked and compared with a defined data content. If a match happens, an interrupt will be generated if RPMFIE is set.

The read polling access sequence is started the same as it of normal read mode. The BUSY bit keeps 1 between periodic intervals.

Polling match mode configured by RPMM controls the comparison match mode.If RPMM = 0, the AND mode is selected. In this mode, status match flag RPMF will be set only when there is a match on all the unmasked bits. While if RPMM = 1, the OR mode is selected. In this mode, RPMF will be set if there is a match on any of the unmasked bit.

In read polling mode, if RPMS is set, read polling sequence will stop when a match is detected, and the BUSY flag is cleared at the end of data phase. Otherwise, the periodic sequence always continues until ABORT bit is set or the QSPI is disabled.

In read polling mode, FIFO is bypassed and the status bytes read are stored in QSPI_DATA, and the status bytes stored are not affected by the MASK control field. Contents in QSPI_DATA is renewed at the beginning of data phase if there is any.

FT is set at the end of data phase, where the external flash memory status bytes are considered read, and it is cleared when QSPI_DATA is read.

19.4.3. Memory map mode

The memory map mode can be selected by configuring the FMOD[1:0] to “11”. In memory map mode, the external flash memory is considered as internal memory, no more than 128MB can be addressed even if the external memory is larger. The memory map mode can not access the address no more than 128MB but outside the range which is specified by FMSZ. Or else, an error will be generated. If the AHB master is the CPU, a hard fault interrupt will be generated. If the AHB master is the DMA, a transfer error will be generated, and the corresponding DMA channel is disabled by hardware.

In this mode, byte, half-word, and word single or burst access are supported.

The memory map mode supports prefetch function on sequential address accessing. The QSPI load the data at the following address before accessing it, if the next access is indeed at the next address, the access with be faster since the data is already prefetched. Otherwise, the read sequence is restarted, pulling CSN low before the read sequence starts.

When the FIFO is full, the output of SCK will be stopped, and the CSN is low during this period. If the TMOUTEN bit in QSPI_CTL register is set, CSN will be pulled high when the duration of the low period reaches the number of SCK clock cycles specified in QSPI_TMOUT register.

At the beginning of a transfer, BUSY goes high before CSN falls, and is cleared when a timeout occurred or abort / disable is issued.

19.5. QSPI configuration

19.5.1. Flash configuration

The configuration in QSPI_DCFG register can be used to specify the characteristics of the external flash memory, so that the QSPI interface can work consistently.

The size of the external memory is defined by FMSZ[4:0] in QSPI_DCFG register. FMSZ + 1 is the number of address bits in the flash memory. The maximum of the flash capacity can be up to 4GB in normal mode.

CSHC[2:0] defines the chip select high time, it specifies the minimum number of SCK cycles that CSN must stay high between two command sequences.

19.5.2. IP configuration

The configuration in QSPI_CTL register can be used to specify the characteristics of the QSPI IP.

PSC[7:0] in QSPI_CTL register indicates the clock prescaler division factor.

SSAMPLE is used to define which SCK edge is used to sample data. By default, the QSPI samples data one half of a SCK cycle after the external flash drives. However, it may be

beneficial to sample data later because of the external signal delays. The sample edge can be shifted half one of SCK cycle using SSAMPLE bit.

The DMA request is enabled by setting the DMAEN bit in QSPI_STAT register. The FIFO threshold level is configured in FTL[3:0] bits in QSPI_CTL register.

19.6. Send instruction only once

Sending instruction only once mode is enabled by setting the SIOO bit in QSPI_TCFG register, this function is valid for all functional modes. If SIOO bit is set, the instruction is sent only once after QSPI_TCFG has been accessed. The subsequent command sequence will skip the instruction phase before the next configuration of QSPI_TCFG register.

19.7. Error and interrupts

TERR and AHB error will be generated if one of the conditions occurs in ***Table 19-5. TERR and AHB error conditions***.

Table 19-5. TERR and AHB error conditions

Error name	Condition
TERR	1. In normal mode or read polling mode, a wrong address has been programed in QSPI_ADDR register according to the address defined by FMSZ. 2. In normal mode, the address (ADDR) plus data length (DTLEN) is greater than external memory size.
AHB error	1. In memory map mode, out of range access is done by AHB master, or when the QSPI is disabled. 2. AHB master is accessing the memory mapped space while the memory map mode is not enabled.

The QSPI interrupt event and flags are listed in ***Table 19-6. QSPI interrupt events***.

Table 19-6. QSPI interrupt events

Interrupt event	Event Flag	Interrupt enable bit	Clear method
FIFO threshold interrupt	FT	FTIE	By hardware
Transfer complete interrupt	TC	TCIE	Set TCC bit in QSPI_STATC register
Transfer error interrupt	TERR	TERRIE	Set TERRC bit in QSPI_STATC register
Timeout interrupt	TMOUT	TMOUTIE	Set TMOUTC bit in QSPI_STATC register
Status match interrupt	RPMF	RPMFIE	Set RPMFC bit in QSPI_STATC register

19.8. Register definition

QSPI base address: 0x4002 5800

19.8.1. Control register (QSPI_CTL)

Address offset: 0x00
Reset value: 0x0000 0010

This register 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[7:0]								RPMM	RPMS	Reserved	TMOUTIE	RPMFIE	FTIE	TCIE	TERRIE
rw								rw	rw		rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCKDVALUE[3:0]				FTL[3:0]				Reserved	SCKDEN	SSAMPLE[1:0]		TMOUTEN	DMAEN	ABORT	QSPIEN
rw				rw					rw	rw		rw	rw	w1s	rw

Bits	Fields	Descriptions
31:24	PSC[7:0]	The clock prescaler of AHB clock for generating SCK. The relationship between the frequency of SCK and the AHB is $f_{SCK}=f_{AHB}/(PSC+1)$. 0: $f_{SCK}=f_{AHB}$ 1: $f_{SCK}=f_{AHB}/2$ 2: $f_{SCK}=f_{AHB}/3$ ... 255: $f_{SCK}=f_{AHB}/256$ When the clock division factor is odd, the duty cycle of SCK is not 50%. And the low level is one cycle longer than the high level. These bits can be modified only when BUSY = 0.
23	RPMM	Read polling match mode 0: AND match mode. If all the unmasked bits received from the flash memory match with the corresponding bits in QSPI_STATMATCH register, RPMF will be set. 1: OR match mode. If any one of the unmasked bits received from the Flash memory matches its corresponding bit in the QSPI_STATMATCH register, RPMF will be set. This bit can be modified only when BUSY = 0.
22	RPMS	Read polling mode stop This bit determines if read polling is stopped after a match. 0: Read polling mode stops when ABORT bit is set or the QSPI is disabled. 1: Read polling mode stops when a match happens. This bit can be modified only when BUSY = 0.
21	Reserved	Must be kept at reset value.

20	TMOUTIE	Timeout interrupt enable 0: disable timeout interrupt 1: enable timeout interrupt
19	RPMFIE	Read polling mode match interrupt enable 0: disable read polling mode match interrupt 1: enable read polling mode match interrupt
18	FTIE	FIFO threshold interrupt enable 0: disable FIFO threshold interrupt 1: enable FIFO threshold interrupt
17	TCIE	Transfer complete interrupt enable 0: disable transfer complete interrupt 1: enable transfer complete interrupt
16	TERRIE	Transfer error interrupt enable 0: disable transfer error interrupt 1: enable transfer error interrupt
15:12	SCKDVALUE[3:0]	SCK delay value. These bits only useful when SCKDEN is enabled and SSAMPLE is configured.
11:8	FTL[3:0]	FIFO threshold level These bits are useful in normal mode, the threshold number of bytes in the FIFO that will cause the FIFO threshold flag to be set. In normal write mode (FMOD = 00): 0: FT is set if there are 1 or more free bytes available to be written to in the FIFO 1: FT is set if there are 2 or more free bytes available to be written to in the FIFO ... 15: FT is set if there are 16 free bytes available to be written to in the FIFO In normal read mode (FMOD = 01): 0: FT is set if there are 1 or more valid bytes that can be read from the FIFO 1: FT is set if there are 2 or more valid bytes that can be read from the FIFO ... 15: FT is set if there are 16 valid bytes that can be read from the FIFO If DMAEN = 1, then the DMA controller for the corresponding channel must be disabled before changing the FTL value.
7	Reserved	Must be kept at reset value.
6	SCKDEN	SCK delay enable when read data from flash, it is only useful when SSAMPLE is 1. 0: disable SCK delay 1: enable SCK delay
5:4	SSAMPLE[1:0]	Sample delay QSPI samples data 1/2 SCK clock cycle after flash memory drives data by default. Taking the delay of external signals into account, thes bits can be configured to

		allow the data to be sampled later. 0: No delay 1: 1/2 cycle delay 2: 1 cycle delay 3: Reserved These bits can be modified only when BUSY = 0.
3	TMOUTEN	Timeout counter enable In memory map mode (FMOD = 11). When setting this bit, the chip select output (CSN) will be high if there is no access after a certain amount of time which specified by TMOUTCYC[15:0]. 0: Disable the timeout counter, the chip select (CSN) will remain low after an access in memory map mode. 1: Enable the timeout counter, the chip select (CSN) will be high if there is no access to flash memory after TMOUTCYC[15:0] cycles in memory map mode. This bit can be modified only when BUSY = 0.
2	DMAEN	DMA enable DMA can be used to transfer data in normal mode.. And when FT is set, DMA transfers initiates. 0: disable DMA 1: enable DMA
1	ABORT	Abort request This bit is used to stop the current command. It is automatically cleared once the abort is completed. In read polling mode or memory map mode, when setting this bit, the RPMS bit or the DMAEN bit will be cleared. 0: No abort request 1: Abort request
0	QSPIEN	the QSPI enable 0: disable QSPI 1: enable QSPI

19.8.2. Device configuration register (QSPI_DCFG)

Address offset: 0x04

Reset value: 0x001F 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											FMSZ[4:0]				
													rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	CSHC[2:0]	Reserved	CKMOD
	rw		rw

Bits	Fields	Descriptions
31:21	Reserved	Must be kept at reset value.
20:16	FMSZ[4:0]	Flash memory size These bits defines the size of external memory. And the number of bytes is $2^{[FMSZ+1]}$. FMSZ+1 is the number of address bits in the flash memory. In normal mode, the flash memory capacity can be up to 4GB, while it is limited to 128MB in memory map mode. These bits can be modified only when BUSY = 0.
15:11	Reserved	Must be kept at reset value.
10:8	CSHC[2:0]	Chip select high cycle The chip select (CSN) must stay high for at least CSHC+1 SCK cycles between two command sequences. 0: CSN must stay high for at least 1 SCK cycle between two flash memory command sequences. 1: CSN must stay high for at least 2 SCK cycles between two flash memory command sequences. ... 7: CSN must stay high for at least 8 SCK cycles between two flash memory command sequences. These bits can be modified only when BUSY = 0.
7:1	Reserved	Must be kept at reset value.
0	CKMOD	This bit indicates the SCK level when QSPI is free. 0: SCK must stay low when CSN is high. 1: SCK must stay high when CSN is high. This bit can be modified only when BUSY = 0.

19.8.3. Status register (QSPI_STAT)

Address offset: 0x08
Reset value: 0x0000 0004

This register 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			FL[4:0]					Reserved		BUSY	TMOUT	RPMF	FT	TC	TERR
			rw							r	r	r	r	r	r

Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:8	FL[4:0]	FIFO level These bits is used to configure the number of valid bytes which are being stored in the FIFO in normal mode. In memory map mode and in read polling mode, FL is 0.
7:6	Reserved	Must be kept at reset value.
5	BUSY	Busy flag This bit is set when a command is transferring. In normal mode, this bit will be cleared once the command phase is completed and if in read opreation of normal mode, FIFO also needs be empty.
4	TMOUT	Timeout flag When the TMOUTEN is set and there is no access to flash memory after TMOUTCYC[15:0] cycles. This bit will be set.
3	RPMF	Read polling match flag This bit is set in read polling mode when the unmasked received data matches the expected value in QSPI_STATMATCH register.
2	FT	FIFO threshold flag In normal mode, when the FIFO threshold has been reachedor if the FIFO is not empty after the last read operation from the flash memory, this bit will be set. And it is cleared by hardware as soon as the threshold condition is not true. In read polling mode, this bit will be set. When the status register is read by the external flash, and it is cleared when the QSPI_DATA register is read.
1	TC	Transfer complete flag This bit will be set when the number of data programmed in QSPI_DTLEN register has been transmitted in normal mode or when abort operation is completed.
0	TERR	Transfer error flag In normal mode, when an invalid address has been accessed. This bit will be set

19.8.4. Status clear register (QSPI_STATC)

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											TMOUTC	RPMFC	Reserved	TCC	TERRC

w w w w

Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	TMOUTC	Clear timeout flag Writing 1 to this bit clears the TMOUT bit in the QSPI_STAT register.
3	RPMFC	Clear read polling mode match flag Writing 1 to this bit clears the RPMF bit in the QSPI_STAT register.
2	Reserved	Must be kept at reset value.
1	TCC	Clear transfer complete flag Writing 1 to this bit clears the TC bit in the QSPI_STAT register.
0	TERRC	Clear transfer error flag Writing 1 to this bit clears the TERR bit in the QSPI_STAT register.

19.8.5. Data length register (QSPI_DTLEN)

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
DTLEN[31:16]															
rw															

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

Bits	Fields	Descriptions
31:0	DTLEN[31:0]	Data length These bits specify the number of data to be retrieved (value+1) in normal and read polling modes. When in read polling mode, the value of these bits should no greater than 3 (indicating 4 bytes). When these bits are configured as 0xFFFF FFFF in normal mode, it means undefined length, the QSPI will continue transferring data until the end of memoryaccording to FMSZ[4:0] in QSPI_DCFG register. 0x0000 0000: the data length to be transferred is 1 byte 0x0000 0001: the data length to be transferred is 2 bytes 0x0000 0002: the data length to be transferred is 3 bytes 0x0000 0003: the data length to be transferred is 4 bytes ... 0xFFFF FFFD: the data length to be transferred is 4,294,967,294 (4G-2) bytes 0xFFFF FFFE: the data length to be transferred is 4,294,967,295 (4G-1) bytes

0xFFFF FFFF: undefined length, all bytes defined by FMSZ[4:0] in QSPI_DCFG register until the end of flash memorywill be transferred. If FMSZ[4:0] is 0x1F, it will continue reading indefinitely.

When in memory map mode, these bits have no effect.

These bits can be modified only when BUSY = 0.

19.8.6. Transfer configuration register (QSPI_TCFG)

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			SIOO	FMOD[1:0]		DATAMOD[1:0]		Reserved	DUMYC[4:0]					ALTESZ[1:0]	
			rw	rw		rw			rw					rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALTEMOD[1:0]		ADDRSZ[1:0]		ADDRMOD[1:0]		IMOD[1:0]		INSTRUCTION[7:0]							
rw		rw		rw		rw		rw							

Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28	SIOO	Send instruction only once mode When IMOD is 00, this bit has no effect. 0: Send instruction on every command sequence 1: Send instruction only for the first command sequence These bits can be modified only when BUSY = 0.
27:26	FMOD[1:0]	Functional mode 00: write operation in normal mode 01: read operation in normal mode 10: read polling mode 11: Memory map mode These bits can be modified when the DMA controller for the corresponding channel is disabled if DMAEN is 1. These bits can be modified only when BUSY = 0.
25:24	DATAMOD[1:0]	Data mode These bits define the data phase's mode of operation. It is also specify the dummy phase mode of operation. 00: No data 01: Data on a single line 10: Data on two lines 11: Data on four lines These bits can be modified only when BUSY = 0.

23	Reserved	Must be kept at reset value.
22:18	DUMYC[4:0]	Number of dummy cycles These bits define the duration of the dummy phase. These bits can be modified only when BUSY = 0.
17:16	ALTESZ[1:0]	Alternate bytes size 00: 8-bit alternate byte 01: 16-bit alternate bytes 10: 24-bit alternate bytes 11: 32-bit alternate bytes These bits can be modified only when BUSY = 0.
15:14	ALTEMOD[1:0]	Alternate bytes mode 00: No alternate bytes 01: Alternate bytes on a single line 10: Alternate bytes on two lines 11: Alternate bytes on four lines These bits can be modified only when BUSY = 0.
13:12	ADDRSZ[1:0]	Address size 00: 8-bit address 01: 16-bit address 10: 24-bit address 11: 32-bit address These bits can be modified only when BUSY = 0.
11:10	ADDRMOD[1:0]	Address mode 00: No address 01: Address on a single line 10: Address on two lines 11: Address on four lines These bits can be modified only when BUSY = 0.
9:8	IMOD[1:0]	Instruction mode 00: No instruction 01: Instruction on a single line 10: Instruction on two lines 11: Instruction on four lines These bits can be modified only when BUSY = 0.
7:0	INSTRUCTION[7:0]	Instruction Command information to be send to the flash memory. These bits can be modified only when BUSY = 0.

19.8.7. Address register (QSPI_ADDR)

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
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]	Address to be send to the external Flash memory When BUSY=0 or in memory mapped mode, writing values to these bits will be ignored.

19.8.8. Alternate bytes register (QSPI_ALTE)

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
ALTE[31:16]															
rw															

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

Bits	Fields	Descriptions
31:0	ALTE[31:0]	Alternate Bytes Optional data to be send to the flash memory follow by address. These bits can be modified only when BUSY = 0

19.8.9. Data register (QSPI_DATA)

Address offset: 0x20
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
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 that will be interacting with flash memory. In write operation of normal mode, the data written to this register will be stored in the FIFO before sending to the flash memory. If the FIFO is full, a write operation will be stopped and wait until the FIFO has enough space. In read operation of normal mode, the data received from the flash memory can be read in this register. If the bytes in the FIFO are less than the requested bytes by the read command, and BUSY=1, the read operation will be stopped and wait until enough data is stored in the FIFO or until the transfer is completed. In read polling mode, this register contains the last data read from the flash memory.

19.8.10. Status mask register (QSPI_STATMK)

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
MASK[31:16]															

rw

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

rw

Bits	Fields	Descriptions
31:0	MASK[31:0]	Status mask in read polling mode Mask of the status bytes received from the flash memory in read polling mode. For bit n of MASK[31:0]: 0: Bit n of the data received can not match 1: Bit n of the data received must match These bits can be modified only when BUSY = 0.

19.8.11. Status match register (QSPI_STATMATCH)

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

MATCH[31:16]
rw

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

Bits	Fields	Descriptions
31:0	MATCH[31:0]	Status match in read polling mode Expected value to be compared with the value in the QSPI_STATMK register to get a match. These bits can be modified only when BUSY = 0.

19.8.12. Interval register (QSPI_INTERVAL)

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															
rw															

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

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	INTERVAL[15:0]	Interval cycle Number of SCK cycles between two read commands in read polling mode. These bits can be modified only when BUSY = 0.

19.8.13. Timeout register (QSPI_TMOUT)

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															
rw															

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

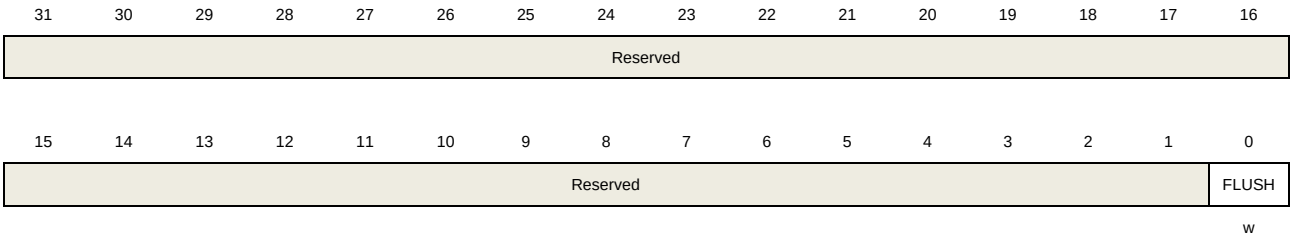
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	TMOUTCYC[15:0]	Timeout cycle When the FIFO is full in memory map mode, these bits indicate how many SCK cycles the QSPI waits for next access. In this duration, CSN keeps low. These bits can be modified only when BUSY = 0.

19.8.14. FIFO flush register (QSPI_FLUSH)

Address offset: 0x34

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	FLUSH	Used to flush all qspi interal fifo.

20. Cryptographic Acceleration Unit (CAU)

20.1. Overview

The cryptographic acceleration unit (CAU) is used to encipher and decipher data with DES, Triple-DES or AES (128, 192, or 256) algorithms. It is fully compliant implementation of the following standards:

- The Data Encryption Standard (DES) and the Triple Data Encryption Algorithm (TDEA) are announced by Federal Information Processing Standards Publication (FIPS) 46-3, October 25, 1999. It follows the American National Standards Institute (ANSI) X9.52 standard.
- The Advanced Encryption Standard (AES) is announced by Federal Information Processing Standards Publication 197, November 26, 2001.

DES / TDES / AES algorithms with different key sizes are supported to perform data encryption and decryption in the CAU in multiple modes.

The CAU is a 32-bit peripheral, DMA transfer is supported and data can be accessed in the input and output FIFO.

20.2. Characteristics

- DES, TDES and AES encryption / decryption algorithms are supported.
- Multiple modes are supported respectively in DES, TDES and AES, including Electronic codebook (ECB), Cipher block chaining (CBC), Counter mode (CTR), Galois / counter mode (GCM), Galois message authentication code mode (GMAC), Counter with CBC-MAC (CCM), Cipher Feedback mode (CFB) and Output Feedback mode (OFB).
- DMA transfer for incoming and outgoing data is supported.

DES / TDES

- Supports the ECB and CBC chaining algorithms.
- Two 32-bit initialization vectors (IV) are used in CBC mode.
- 8*32-bit input and output FIFO.
- Multiple data types are supported, including No swapping, Half-word swapping Byte swapping and Bit swapping.
- Data are transferred by DMA, CPU during interrupts, or without both of them.

AES

- Supports the ECB, CBC, CTR, GCM, GMAC, CCM, CFB and OFB chaining algorithms.
- Supports 128-bit, 192-bit and 256-bit keys.
- Four 32-bit initialization vectors (IV) are used in CBC, CTR, GCM, GMAC, CCM, CFB

and OFB modes.

- 8*32-bit input and output FIFO.
- Multiple data types are supported, including No swapping, Half-word swapping Byte swapping and Bit swapping.
- Data can be transferred by DMA, CPU during interrupts, or without both of them.

20.3. CAU data type and initialization vectors

20.3.1. Data type

The cryptographic acceleration unit receives data of 32 bits at a time, while they are processed in 64 / 128 bits for DES / AES algorithms. For each data block, according to the data type, the data could be bit / byte / half-word / no swapped before they are transferred into the cryptographic acceleration processor. The same swapping operation should be also performed on the processor output data before they are collected. Note the least-significant data always occupies the lowest address location no matter which data type is configured, because the system memory is little-endian.

Figure 20-1. DATAM No swapping and Half-word swapping and *Figure 20-2. DATAM Byte swapping and Bit swapping* illustrate the 128-bit AES block data swapping according to different data types. (For DES, the data block is two 32-bit words, please refer to the first two words data swapping in the figure).

Figure 20-1. DATAM No swapping and Half-word swapping

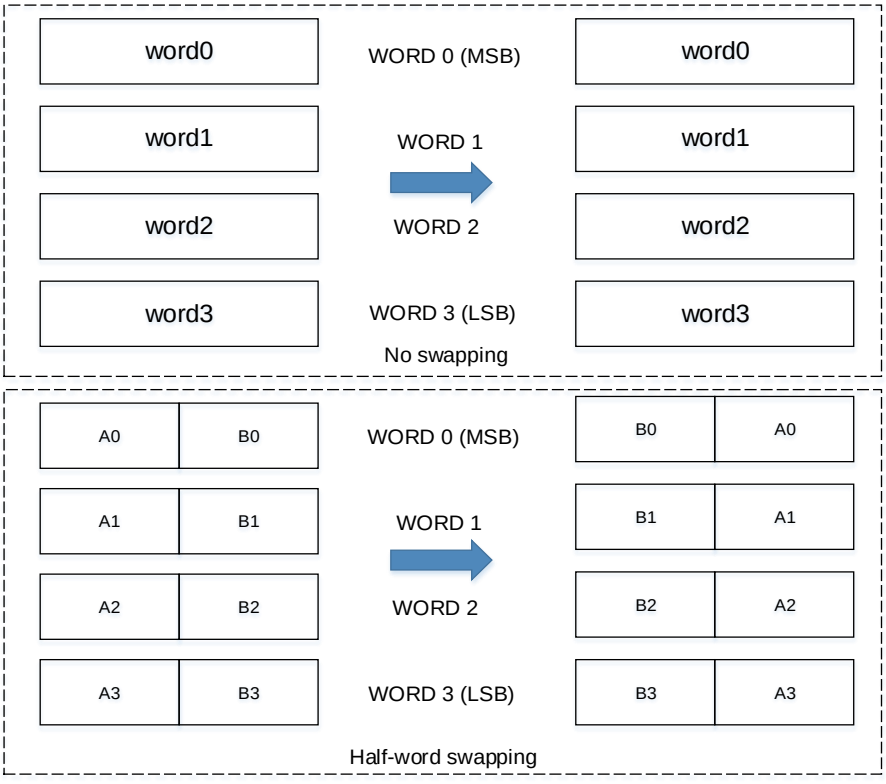
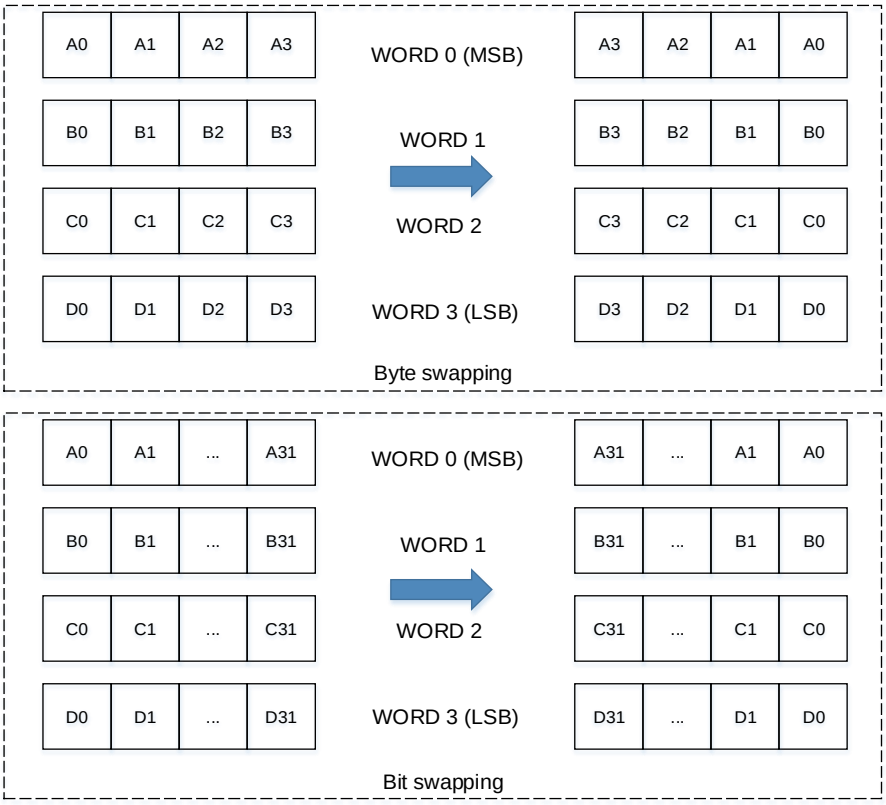


Figure 20-2. DATAM Byte swapping and Bit swapping



20.3.2. Initialization vectors

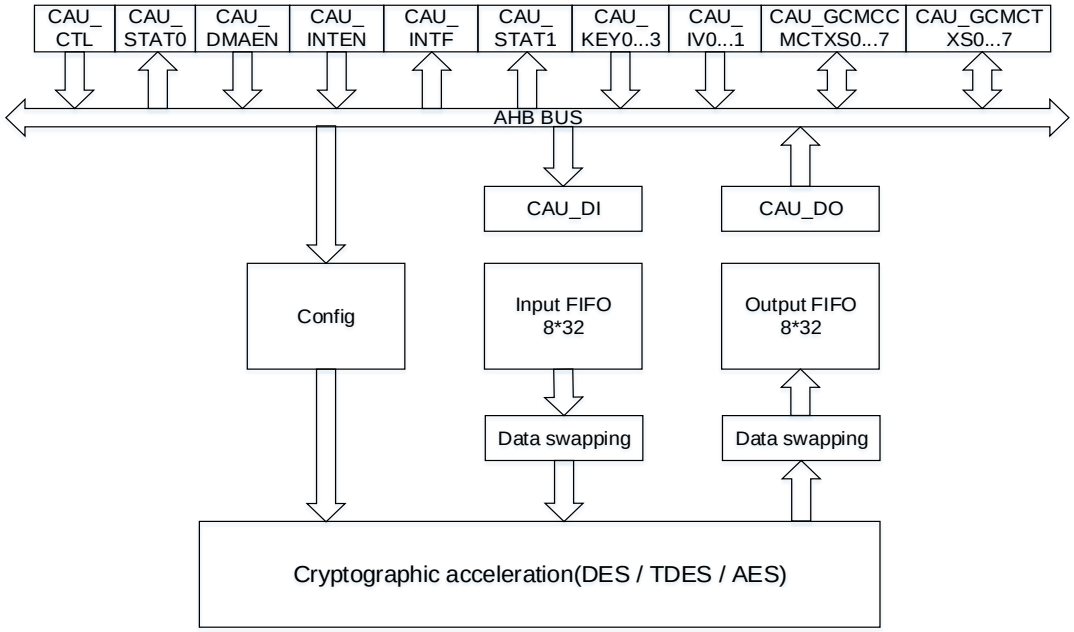
The initialization vectors are used in CBC, CTR, GCM, GMAC, CCM, CFB and OFB modes to XOR with data blocks. They are independent of plaintext and ciphertext, and the DATAM value will not affect them. Note the initialization vector registers CAU_IV0,1(H / L) can only be written when BUSY is 0, otherwise the write operations are invalid.

20.4. Cryptographic acceleration processor

The cryptographic acceleration unit implements DES and AES acceleration processors, which are detailed described in section ***DES / TDES cryptographic acceleration processor*** and ***AES cryptographic acceleration processor***.

Figure 20-3. CAU diagram shows the block diagram of the cryptographic acceleration unit.

Figure 20-3. CAU diagram



20.4.1. DES / TDES cryptographic acceleration processor

The DES / TDES cryptographic acceleration processor contains the DES algorithm (DEA), cryptographic keys (DES algorithm needs 1 key and TDES algorithm needs 3 keys), and initialization vectors in CBC mode.

DES / TDES key

[KEY1] is used in DES and [KEY3 KEY2 KEY1] are used in TDES respectively.
When TDES algorithm is configured, three different keying options are allowed:

1. Three same keys

The three keys KEY3, KEY2 and KEY1 are completely equal, which means KEY3 = KEY2 = KEY1. FIPS PUB 46-3 – 1999 (and ANSI X9.52 -1998) refers to this option. It is easy to understand that this mode is equivalent to DES.

2. Two different keys

In this option, KEY2 is different from KEY1, and KEY3 is equal to KEY1, which means, KEY1 and KEY2 are independent while KEY3 = KEY1. FIPS PUB 46-3 – 1999 (and ANSI X9.52 – 1998) refers to this option.

3. Three different keys

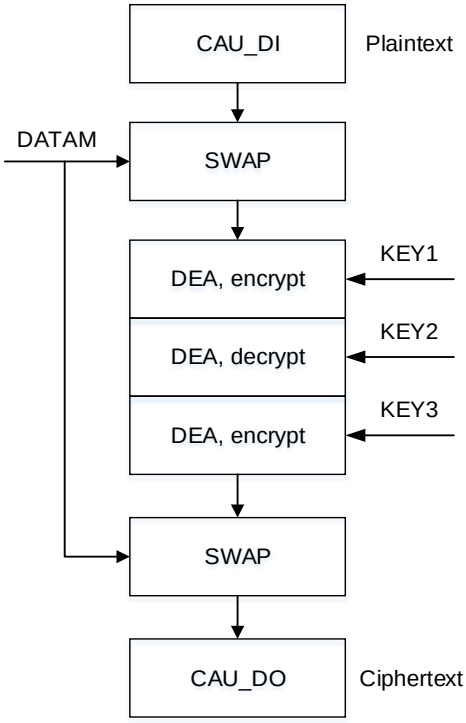
In this option, KEY1, KEY2 and KEY3 are completely independent. FIPS PUB 46-3 -1999 (and ANSI X9.52 – 1998) refers to this option.

More information of the thorough explanation of the key used in the DES / TDES please refer to FIPS PUB 46-3 (and ANSI X9.52 -1998), and the explanation process is omitted in this manual.

DES / TDES ECB encryption

The 64-bit input plaintext is first obtained after data swapping according to the data type. When the TDES algorithm is configured, the input data block is read in the DEA and encrypted using KEY1. The output is fed back directly to next DEA and then decrypted using KEY2. After that, the output is fed back directly to the last DEA and encrypted with KEY3. The output after above processes is then swapped back according to the data type again, and a 64-bit ciphertext is produced. When the DES algorithm is configured, the result of the first DEA encrypted using KEY1 is swapped directly according to the data type, and a 64-bit ciphertext is produced. The procedure of DES / TDES ECB mode encryption is illustrated in ***Figure 20-4. DES / TDES ECB encryption***.

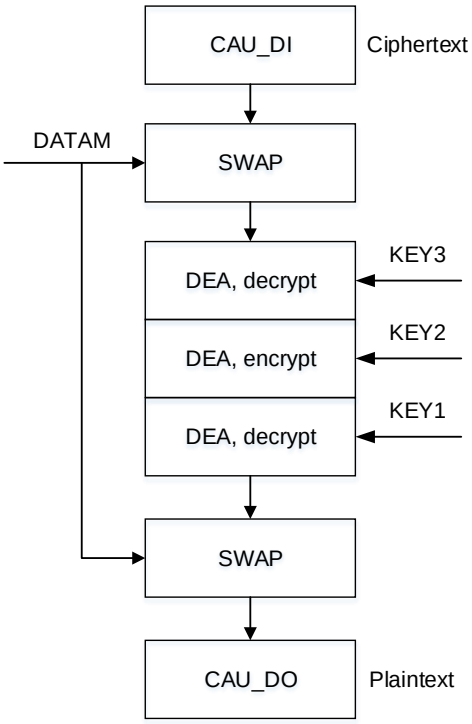
Figure 20-4. DES / TDES ECB encryption



DES / TDES ECB decryption

The 64-bit input ciphertext is first obtained after data swapping according to the data type. When the TDES algorithm is configured, the input data block is read in the DEA and decrypted using KEY3. The output is fed back directly to next DEA and then encrypted using KEY2. After that, the output is fed back directly to the last DEA and decrypted with KEY1. The output after above process is then swapped back according to the data type again, and a 64-bit plaintext is produced. When the DES algorithm is configured, the result of the first DEA decrypted using KEY1 is swapped directly according to the data type, and a 64-bit plaintext is produced. The procedure of DES / TDES ECB mode decryption is illustrated in ***Figure 20-5. DES / TDES ECB decryption***.

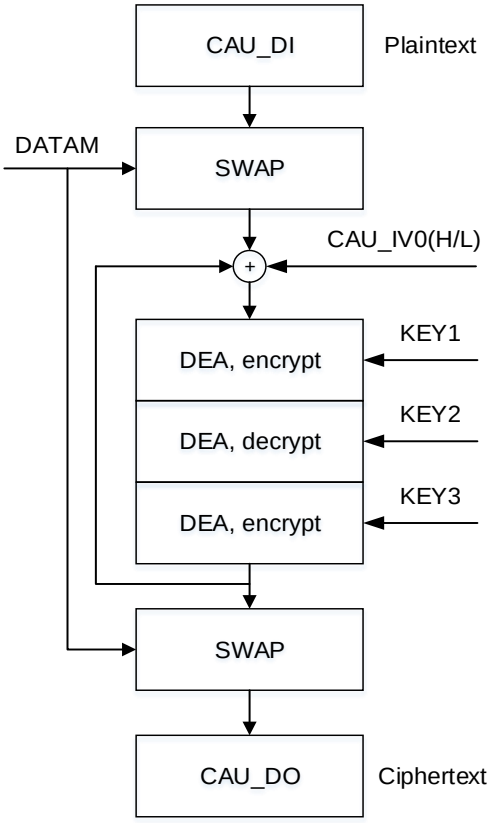
Figure 20-5. DES / TDES ECB decryption



DES / TDES CBC encryption

The input data of the DEA block in CBC mode consists of two aspects: the input plaintext after data swapping according to the data type, and the initialization vectors. When the TDES algorithm is configured, the XOR result of the swapped plaintext data block and the 64-bit initialization vector CAU_IV0..1 is read in the DEA and encrypted using KEY1. The output is fed back directly to next DEA and then decrypted using KEY2. After that, the output is fed back directly to the last DEA and encrypted with KEY3. The result is then used as the next initialization vector and exclusive-ORed with the next plaintext data block to process next encryption. The above operations are repeated until the last plaintext block is encrypted. Note if the plaintext message does not consist of an integral number of data blocks, the final partial data block should be encrypted in a specified manner. At last, the output ciphertext is also obtained after data swapping according to the data type. When the DES algorithm is configured, the state and process of the second and third block of DEA should be omitted. The procedure of DES / TDES CBC mode encryption is illustrated in ***Figure 20-6. DES / TDES CBC encryption***.

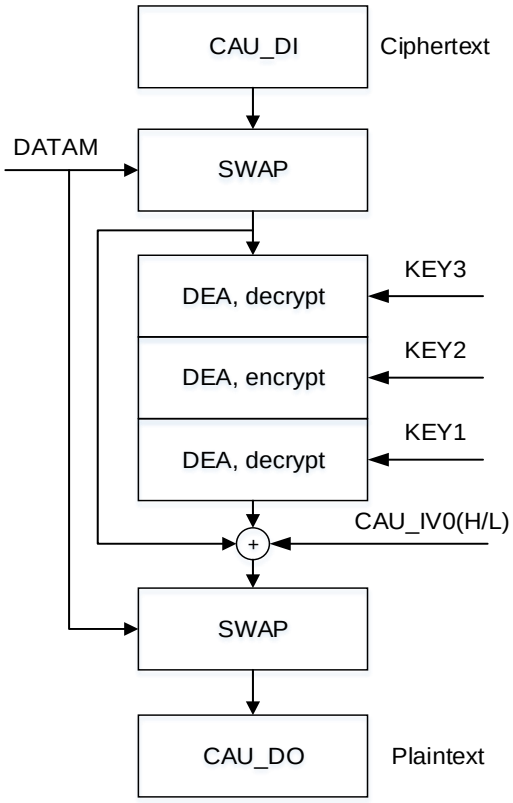
Figure 20-6. DES / TDES CBC encryption



DES / TDES CBC decryption

In DES / TDES CBC decryption, when the TDES algorithm is configured, the first ciphertext block is used directly after data swapping according to the data type, it is read in the DEA and decrypted using KEY3. The output is fed back directly to next DEA and then encrypted using KEY2. After that, the output is fed back directly to the last DEA and decrypted with KEY1. The first result of above process is then XORed with the initialization vector which is the same as that used during encryption. At the same time, the first ciphertext is then used as the next initialization vector and exclusive-ORed with the next result after DEA blocks. The above operations are repeated until the last ciphertext block is decrypted. Note if the ciphertext message does not consist of an integral number of data blocks, the final partial data block should be decrypted in a specified manner same to that in encryption. At last, the output plaintext is also obtained after data swapping according to the data type. When the DES algorithm is configured, the state and process of the second and third block of DEA should also be omitted. The procedure of DES / TDES CBC mode decryption is illustrated in ***Figure 20-7. DES / TDES CBC decryption***.

Figure 20-7. DES / TDES CBC decryption



20.4.2. AES cryptographic acceleration processor

The AES cryptographic acceleration processor consists of three components, including the AES algorithm (AEA), multiple keys and the initialization vectors or Nonce.

Three lengths of AES keys are supported: 128, 192 and 256 bits, and different initialization vectors or nonce are used depends on the operation mode.

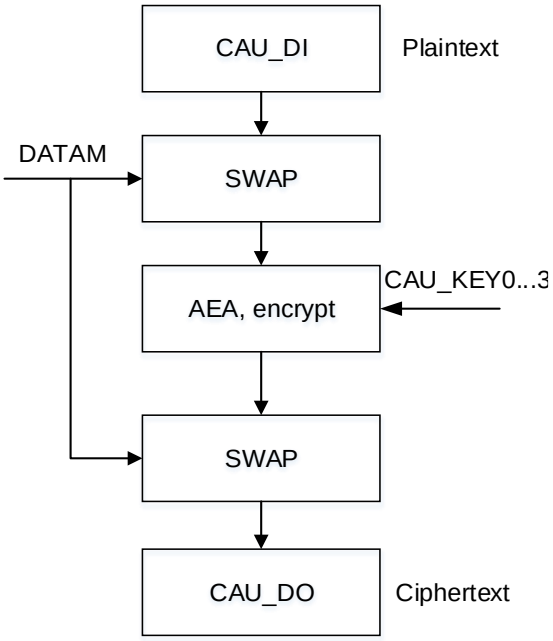
The AES key is used as [KEY3 KEY2] when the key size is configured as 128, [KEY3 KEY2 KEY1] when the key size is configured as 192 and [KEY3 KEY2 KEY1 KEY0] when the key size is configured as 256.

The thorough explanation of the key used in the AES is provided in FIPS PUB 197 (November 26, 2001), and the explanation process is omitted in this manual.

AES-ECB mode encryption

The 128-bit input plaintext is first obtained after data swapping according to the data type. The input data block is read in the AEA and encrypted using the 128, 192 or 256 -bit key. The output after above process is then swapped back according to the data type again, and a 128-bit ciphertext is produced and stored in the out FIFO. The procedure of AES ECB mode encryption is illustrated in ***Figure 20-8. AES ECB encryption***.

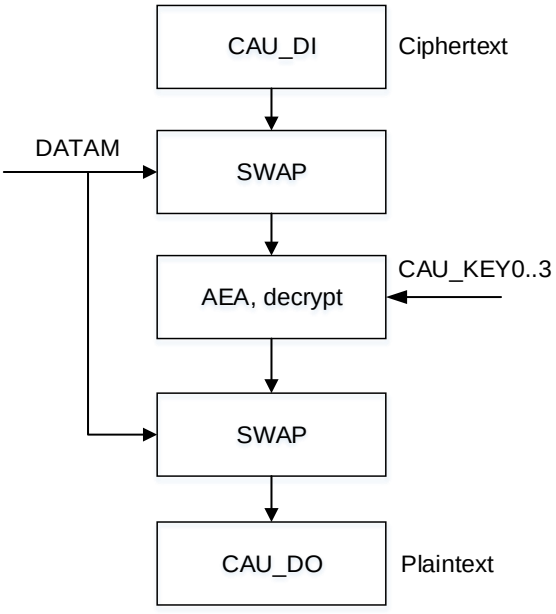
Figure 20-8. AES ECB encryption



AES-ECB mode decryption

First of all, the key derivation must be completed to prepare the decryption keys, the input key of the key schedule is the same to that used in encryption. The last round key obtained from the above operation is then used as the first round key in the decryption. After the key derivation, the 128-bit input ciphertext is first obtained after data swapping according to the data type. The input data block is read in the AEA and decrypted using keys prepared above. The output is then swapped back according to the data type again, and a 128-bit plaintext is produced. The procedure of AES ECB mode decryption is illustrated in ***Figure 20-9. AES ECB decryption***.

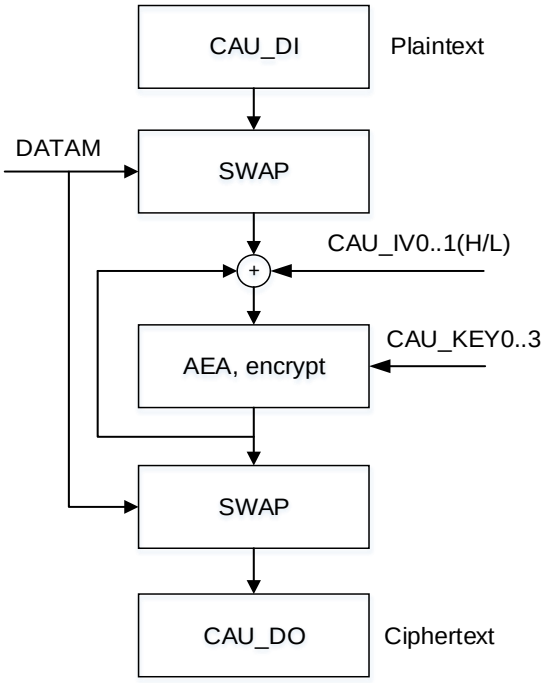
Figure 20-9. AES ECB decryption



AES-CBC mode encryption

The input data of the AEA block in CBC mode consists of two aspects: the input plaintext after data swapping according to the data type, and the initialization vectors. The XOR result of the swapped plaintext data block and the 128-bit initialization vector CAU_IV0..1 is read in the AEA and encrypted using the 128-, 192-, 256-bit key. The result is then used as the next initialization vector and exclusive-ORed with the next plaintext data block to process next encryption. The above operations are repeated until the last plaintext block is encrypted. Note if the plaintext message does not consist of an integral number of data blocks, the final partial data block should be encrypted in a specified manner. At last, the output ciphertext is also obtained after data swapping according to the data type. The procedure of AES CBC mode encryption is illustrated in ***Figure 20-10. AES CBC encryption***.

Figure 20-10. AES CBC encryption

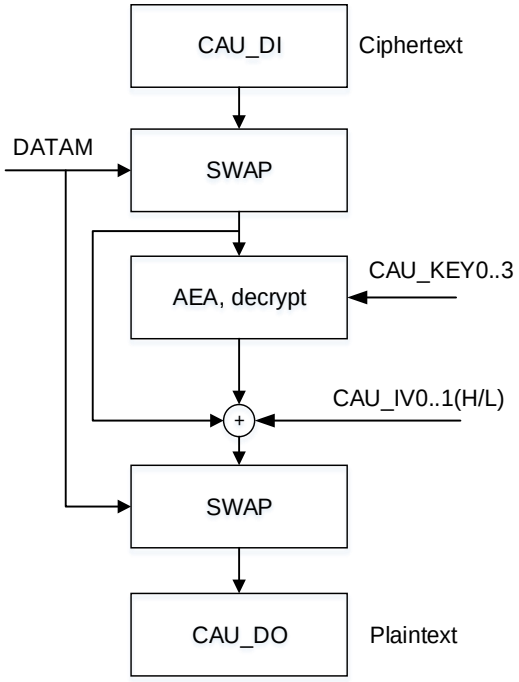


AES-CBC mode decryption

Similar to that in AES-ECB mode decryption, the key derivation also must be completed first to prepare the decryption keys, the input of the key schedule should be the same to that used in encryption. The last round key obtained from the above operation is then used as the first round key in the decryption. After the key derivation, the 128-bit input ciphertext is first obtained after data swapping according to the data type. The input data block is read in the AEA and decrypted using keys prepared above. At the same time, the first ciphertext is then used as the next initialization vector and exclusive-ORed with the next result after AEA blocks (The first initialization is obtained directly from the CAU_IV0..1 registers). The above operations are repeated until the last ciphertext block is decrypted. Note if the ciphertext message does not consist of an integral number of data blocks, the final partial data block should be decrypted in a specified manner same to that in encryption. At last, the output

plaintext is also obtained after data swapping according to the data type. The procedure of AES CBC mode decryption is illustrated in ***Figure 20-11. AES CBC decryption***.

Figure 20-11. AES CBC decryption



AES-CTR mode

In counter mode, a counter is used in addition with a nonce value to be encrypted and decrypted in AEA, and the result will be used for the XOR operation with the plaintext or the ciphertext. As the counter is incremented from the same initialized value for each block in encryption and decryption, the key schedules during the encryption and decryption are the same. Then decryption operation acts exactly in the same way as the encryption operation. Only the 32-bit LSB of the 128-bit initialization vector represents the counter, which means the other 96 bits are unchanged during the operation, and the initial value should be set to 1. Nonce is 32-bit single-use random value and should be updated to each communication block. And the 64-bit initialization vector is used to ensure that a given value is used only once for a given key. ***Figure 20-12. Counter block structure*** illustrates the counter block structure and ***Figure 20-13. AES CTR encryption / decryption*** shows the AES CTR encryption / decryption.

Figure 20-12. Counter block structure

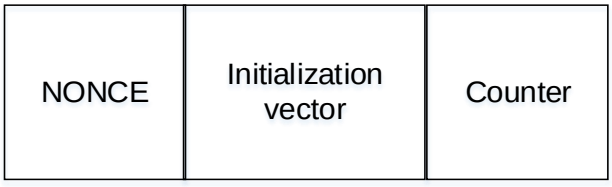
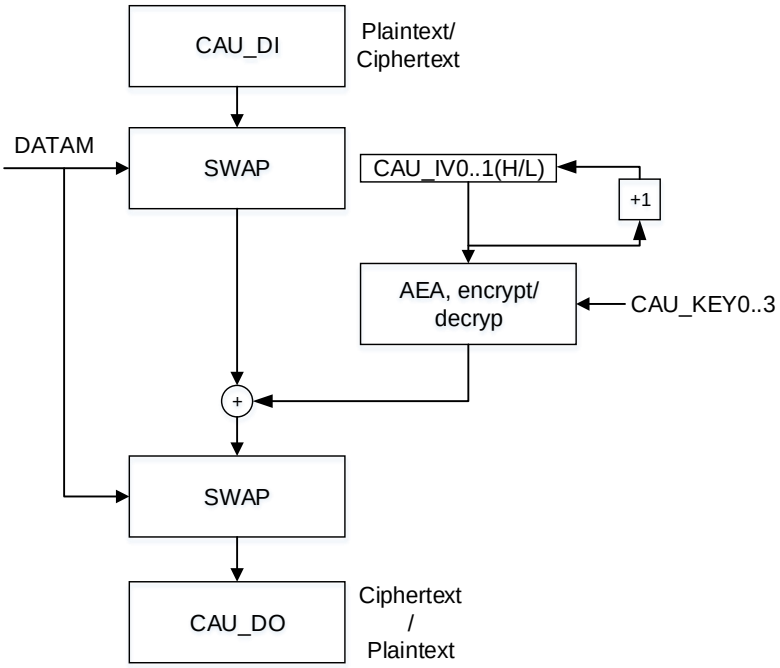


Figure 20-13. AES CTR encryption / decryption



AES-GCM mode

The AES Galois / counter mode (GCM) can be used to encrypt or authenticate message, then ciphertext and tag can be obtained. This algorithm is based on AES CTR mode to ensure confidentiality. A multiplier over a fixed finite field is used to generate the tag.

In this mode, four steps are required to perform an encryption / decryption:

1. GCM prepare phase

   The hash key is calculated and saved internally to be used later.

   (a) Clear the CAUEN bit to make sure CAU is disabled.
   (b) Configure the ALGM[3:0] bits to ‘1000’.
   (c) Configure GCM_CCMPH[1:0] bits to ‘00’.
   (d) Configure key registers and initialization vectors.
   (e) Enable CAU by writing 1 to CAUEN bit.
   (f) Wait until CAUEN bit is cleared by hardware, and then enable CAU again for following phases.

2. GCM AAD (additional authenticated data) phase

   This phase must be performed after GCM prepare phase and also precede the encryption / decryption phase. In this phase, data is authenticated but not protected.

   (g) Configure GCM_CCMPH[1:0] bits to ‘01’.
   (h) Write data into CAU_DI register, INF and IEM flags can be used to determine if the input FIFO can receive data. The size of the AAD must be a multiple of 128bits. DMA can also be used.

(i) Repeat (h) until all AAD data are supplied, wait until BUSY bit is cleared.

3. GCM encryption / decryption phase

   This phase must be performed after GCM AAD phase. In this phase, the message is authenticated and encrypted / decrypted.

   (j) Configure GCM_CCMPH[1:0] bits to '10'.
   (k) Configure the computation direction in CAUDIR.
   (l) Write payload data into CAU_DI register, INF and IEM flags can be used to determine if the input FIFO can receive data. ONE and OFU flags can be used to check if the output FIFO is not empty. If the output FIFO is not empty, read the CAU_DO register. DMA can also be used.
   (m) Repeat (l) step until all payload blocks are processed.

4. GCM tag phase

   In this phase, the final authentication tag is generated.

   (n) Configure GCM_CCMPH[1:0] bits to '11'.
   (o) Write the input into the CAU_DI register, 4 times write operation is needed. The input consists of the AAD size (64bits) and the payload data size (64bits).
   (p) Wait until the ONE flag is set to 1, and then read CAU_DO 4 times. The output corresponds to the authentication tag.
   (q) Disable the CAU.

Note: The key should be prepared at the beginning when a decryption is performed.

AES-GMAC mode

The AES Galois message authentication code mode is also supported to authenticate the message. It is processing based on the AES-GCM mode, while the encryption / decryption phase is by-passed.

AES-CCM mode

The AES combined cipher machine mode, which is similar to AES-GCM mode, also allows encrypting and authenticating message. It is also based on AES-CTR mode to ensure confidentiality. In this mode, AES-CBC is used to generate a 128-bit tag.

The CCM standard (RFC 3610 Counter with CBC-MAC (CCM) dated September 2003) defines particular encoding rules for the first authentication block (B0 in the standard). In particular, the first block includes flags, a nonce and the payload length expressed in bytes. The CCM standard also specifies another format for encryption / decryption, called A or counter. The counter is incremented during the encryption / decryption phase and its 32 LSB bits are initialized to '1' during the tag generation (A0 packet in the CCM standard).

Note: The formatting operation of B0 packet should be handled by software.

In this mode, four steps are required to perform an encryption / decryption:

1. CCM prepare phase

   In this phase, B0 packet (the first packet) is programmed into the CAU_DI register. CAU_DO never contain data in this phase.

   (a) Clear the CAUEN bit to make sure CAU is disabled.
   (b) Configure the ALGM[3:0] bits to '1001'.
   (c) Configure GCM_CCMPH[1:0] bits to '00'.
   (d) Configure key registers and initialization vectors.
   (e) Enable CAU by writing 1 to CAUEN bit.
   (f) Program the B0 packet into the CAU_DI.
   (g) Wait until CAUEN is cleared by hardware, and then enable CAU again for following phases.

2. CCM AAD (additional authenticated data) phase

   This phase must be performed after CCM prepare phase and also precede the encryption / decryption phase. In this phase, CAU_DO never contain data in this phase.

   This phase can be by-passed if there is no additional authenticated data.

   (h) Configure GCM_CCMPH[1:0] bits to '01'
   (i) Write data into CAU_DI register, INF and IEM flags can be used to determine if the input FIFO can receive data. The size of the AAD must be a multiple of 128 bits. DMA can also be used.
   (j) Repeat (i) until all AAD data are supplied, wait until BUSY bit is cleared

3. CCM encryption / decryption phase

   This phase must be performed after CCM AAD phase. In this phase, the message is authenticated and encrypted / decrypted.

   Like GCM, the CCM chaining mode can be applied on a message composed only by plaintext authenticated data (that is, only AAD, no payload). Note that this way of using CCM is not called CMAC (it is not similar to GCM / GMAC).

   (k) Configure GCM_CCMPH[1:0] bits to '10'
   (l) Configure the computation direction in CAUDIR
   (m) Write data into CAU_DI register, INF and IEM flags can be used to determine if the input FIFO can receive data. ONE and OFU flags can be used to check if the output FIFO is not empty. If so, read the CAU_DO register. DMA can also be used.
   (n) Repeat (m) step until all payload blocks are processed.

4. CCM tag phase

   In this phase, the final authentication tag is generated.

   (o) Configure GCM_CCMPH[1:0] bits to '11'
   (p) Write the 128 bit input into the CAU_DI register, 4 times of write operation to CAU_DI is needed. The input is the A0 value.

(q) Wait until the ONE flag is set to 1, and then read CAU_DO 4 times. The output corresponds to the authentication tag.

(r) Disable the CAU

AES-CFB mode

The Cipher Feedback (CFB) mode is a confidentiality mode that features the feedback of successive ciphertext segments into the input blocks of the forward cipher to generate output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and the decryption process is similar to the encryption described before.

AES-OFB mode

The Output Feedback (OFB) mode is a confidentiality mode that features the iteration of the forward cipher on an initialization vectors (IV) to generate a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and the decryption process is similar to the encryption described before.

20.5. Operating modes

Encryption

1. Disable the CAU by resetting the CAUEN bit in the CAU_CTL register.
2. Select and configure the key length with the KEYM bits in the CAU_CTL register if AES algorithm is chosen.
3. Configure the CAU_KEY0..3(H / L) registers according to the algorithm.
4. Configure the DATAM bit in the CAU_CTL register to select the data swapping type.
5. Configure the algorithm (DES / TDES / AES) and the chaining mode (ECB / CBC / CTR / GCM / GMAC / CCM / CFB / OFB) by writing the ALGM[3:0] bit in the CAU_CTL register.
6. Configure the encryption direction by writing 0 to the CAUDIR bit in the CAU_CTL register.
7. Configure the initialization vectors by writing the CAU_IV0..1 registers.
8. Flush the input FIFO and output FIFO by configure the FFLUSH bit in the CAU_CTL register when CAUEN is 0.
9. Enable the CAU by set the CAUEN bit as 1 in the CAU_CTL register.
10. If the INF bit in the CAU_STAT0 register is 1, then write data blocks into the CAU_DI register. The data can be transferred by DMA / CPU during interrupts / no DMA or interrupts.
11. Wait for ONE bit in the CAU_STAT0 register is 1 then read the CAU_DO registers. The output data can also be transferred by DMA / CPU during interrupts / no DMA or interrupts.
12. Repeat steps 10, 11 until all data blocks has been encrypted.

Decryption

1. Disable the CAU by resetting the CAUEN bit in the CAU_CTL register.
2. Select and configure the key length with the KEYM bits in the CAU_CTL register if AES algorithm is chosen.
3. Configure the CAU_KEY0..3(H / L) registers according to the algorithm.
4. Configure the DATAM bit in the CAU_CTL register to select the data swapping type.
5. Configure the ALGM[3:0] bits to “0111” in the CAU_CTL register to complete the key derivation.
6. Enable the CAU by set the CAUEN bit as 1.
7. Wait until the BUSY and CAUEN bit return to 0 to make sure that the decryption keys are prepared.
8. Configure the algorithm (DES / TDES / AES) and the chaining mode (ECB / CBC / CTR / GCM / GMAC / CCM / CFB / OFB) by writing the ALGM[3:0] bit in the CAU_CTL register.
9. Configure the decryption direction by writing 1 to the CAUDIR bit in the CAU_CTL register.
10. Configure the initialization vectors by writing the CAU_IV0..1 registers.
11. Flush the input FIFO and output FIFO by configure the FFLUSH bit in the CAU_CTL register when CAUEN is 0.
12. Enable the CAU by set the CAUEN bit as 1 in the CAU_CTL register.
13. If the INF bit in the CAU_STAT0 register is 1, then write data blocks into the CAU_DI register. The data can be transferred by DMA / CPU during interrupts / no DMA or interrupts.
14. Wait for ONE bit in the CAU_STAT0 register is 1, then read the CAU_DO registers. The output data can also be transferred by DMA / CPU during interrupts / no DMA or interrupts.
15. Repeat steps 13, 14 until all data blocks has been decrypted.

20.6. CAU DMA interface

The DMA can be used to transfer data blocks with the interface of the cryptographic acceleration unit. The operations can be controlled by the CAU_DMAEN register. DMAIEN is used to enable the DMA request during the input phase, then a word is written into CAU_DI from DMA. DMAOEN is used to enable the DMA request during the output phase, then a word is read from the CAU.

Single and Burst transfers are both supported to ensure the data transfer if the number of words is not an integral multiple of burst size. Note the DMA controller should be configured to perform burst of 4 words or less to make sure no data will be lost. DMA channel for output data has a higher priority than that channel for input data so that the output FIFO can be empty earlier than that the input FIFO is full.

20.7. CAU interrupts

There are two types of interrupt registers in CAU, which are CAU_STAT1 and CAU_INTF. In

CAU, the interrupt is used to indicate the situation of the input and output FIFO.

Any of input and output FIFO interrupt can be enabled or disabled by configuring the Interrupt Enable register CAU_INTEN. Value 1 of the register enable the interrupts.

Input FIFO interrupt

The input FIFO interrupt is asserted when the number of words in the input FIFO is less than four words, then ISTA is asserted. And if the input FIFO interrupt is enabled by IINTEN with a 1 value, the IINTF is also asserted. Note if the CAUEN is low, then the ISTA and IINTF are also always low.

Output FIFO interrupt

The output FIFO interrupt is asserted when the number of words in the output FIFO is more than one words, then OSTA is asserted. And if the output FIFO interrupt is enabled by OINTEN with a 1 value, the OINTF is also asserted. Note Unlike that of Input FIFO interrupt, the value of CAUEN will never affect the situation of OSTA and OINTF.

20.8. CAU suspended mode

It is possible to suspend a data block if another new data block with a higher priority needs to be processed in CAU. The following steps can be performed to complete the encryption / decryption acceleration of the suspended data blocks.

When DMA transfer is used:

1. Stop the current input transfer. Clear the DMAIEN bit in the CAU_DMAEN register.
2. When it is DES or AES, wait until both the input and output FIFO are both empty if the input FIFO is not empty (IEM = 0), then write a word of data into CAU_DI register, do as so until the IEM is checked to be 1, then wait until the BUSY bit is cleared, so that the next data block will not be affected by the last one. Case of TDES is similar to that of AES except that it does not need to wait until the input FIFO is empty.
3. Stop the output transfer by clearing the DMAOEN bit in the CAU_DMAEN register. And disable the CAU by clearing the CAUEN bit in the CAU_CTL register.
4. Save the configuration, including the key size, data type, operation mode, direction, GCM CCM phase and the key values. When it is CBC, CTR, GCM, GMAC, CCM, CFB or OFB chaining mode, the initialization vectors should also be stored. When it is GCM, GMAC, or CCM mode, the context switch registers CAU_GCMCCMCTXSx (x = 0..7) and CAU_GCMCTXSx (x = 0..7) should also be stored.
5. Configure and process the new data block.
6. Restore the process before. Configure the CAU with the parameters stored before, and prepare the key and initialization vectors, and the context switch registers CAU_GCMCCMCTXSx (x = 0..7) and CAU_GCMCTXSx (x = 0..7) should also be restored. Then enable CAU by setting the CAUEN bit in the CAU_CTL register.

When data transfer is done by CPU access to CAU_DI and CAU_DO:

1. When the data transfer is done by CPU access, then wait for the fourth read of the CAU_DO register and before the next CAU_DI write access so that the message is suspended at the end of a block processing.
2. Disable the CAU by clearing the CAUEN bit in the CAU_CTL register.
3. Save the configuration, including the key size, data type, operation mode, direction, GCM CCM phase and the key values. When it is CBC, CTR, GCM, GMAC, CCM, CFB or OFB chaining mode, the initialization vectors should also be stored. When it is GCM, GMAC, or CCM mode, the context switch registers CAU_GCMCCMCTXSx (x = 0..7) and CAU_GCMCTXSx (x = 0..7) should also be stored.
4. Configure and process the new data block.
5. Restore the process before. Configure the CAU with the parameters stored before, and prepare the key and initialization vectors, and the context switch registers CAU_GCMCCMCTXSx (x = 0..7) and CAU_GCMCTXSx (x = 0..7) should also be restored. Then enable CAU by setting the CAUEN bit in the CAU_CTL register.

20.9. Register definition

CAU secure access base address: 0x4C06 0000

20.9.1. Control register (CAU_CTL)

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												ALGM[3]	Reserved	GCM_CCMPH[1:0]	
												rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAUEN	FFLUSH	Reserved				KEYM[1:0]		DATAM[1:0]		ALGM[2:0]			CAUDIR	Reserved	
rw	w					rw		rw		rw			rw		

Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19	ALGM[3]	Encryption / decryption algorithm mode bit 3
18	Reserved	Must be kept at reset value.
17:16	GCM_CCMPH[1:0]	GCM CCM phase 00: prepare phase 01: AAD phase 10: encryption / decryption phase 11: tag phase
15	CAUEN	CAU Enable 0: CAU is disabled 1: CAU is enabled **Note:** the CAUEN can be cleared automatically when the key derivation (ALGM = 0111b) is finished or the AES-GCM or AES-CCM prepare phase finished.
14	FFLUSH	Flush FIFO 0: No effect 1: When CAUEN = 1, flush the input and output FIFO Reading this bit always returns 0
13:10	Reserved	Must be kept at reset value.
9:8	KEYM[1:0]	AES key size mode configuration, must be configured when BUSY = 0 00: 128-bit key length 01: 192-bit key length 10: 256-bit key length

		11: never use
7:6	DATAM[1:0]	Data swapping type mode configuration, must be configured when BUSY = 0 00: No swapping 01: Half-word swapping 10: Byte swapping 11: Bit swapping
5:3	ALGM[2:0]	Encryption / decryption algorithm mode bit 0 to bit 2 These bits and bit 19 of CAU_CTL must be configured when BUSY = 0 0000: TDES-ECB with CAU_KEY1, 2, 3. Initialization vectors (CAU_IV0..1) are not used 0001: TDES-CBC with CAU_KEY1, 2, 3. Initialization vectors (CAU_IV0) is used to XOR with data blocks 0010: DES-ECB with only CAU_KEY1 Initialization vectors (CAU_IV0..1) are not used 0011: DES-CBC with only CAU_KEY1 Initialization vectors (CAU_IV0) is used to XOR with data blocks 0100: AES-ECB with CAU_KEY0, 1, 2, 3. Initialization vectors (CAU_IV0..1) are not used 0101: AES-CBC with CAU_KEY0, 1, 2, 3. Initialization vectors (CAU_IV0..1) are used to XOR with data blocks 0110: AES_CTR with CAU_KEY0, 1, 2, 3. Initialization vectors (CAU_IV0..1) are used to XOR with data blocks In this mode, encryption and decryption are same, then the CAUDIR is disregarded. 0111: AES key derivation for decryption mode. The input key must be same to that used in encryption. The BUSY bit is set until the process has been finished, and CAUEN is then cleared. 1000: Galois Counter Mode (GCM). This algorithm mode is also used for GMAC algorithm. 1001: Counter with CBC-MAC (CCM). 1010: Cipher Feedback (CFB) mode 1011: Output Feedback (OFB) mode
2	CAUDIR	CAU direction, must be configured when BUSY = 0 0: Encryption 1: Decryption
1:0	Reserved	Must be kept at reset value.

20.9.2. Status register 0 (CAU_STAT0)

Address offset: 0x04

Reset value: 0x0000 0003

This register 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											BUSY	OFU	ONE	INF	IEM
											r	r	r	r	r

Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	BUSY	Busy bit 0: No processing. This is because: - CAU is disabled by CAUEN = 0 or the processing has been completed. - No enough data or no enough space in the input / output FIFO to perform a data block 1: CAU is processing data or key derivation.
3	OFU	Output FIFO is full 0: Output FIFO is not full 1: Output FIFO is full
2	ONE	Output FIFO is not empty 0: Output FIFO is empty 1: Output FIFO is not empty
1	INF	Input FIFO is not full 0: Input FIFO is full 1: Input FIFO is not full
0	IEM	Input FIFO is empty 0: Input FIFO is not empty 1: Input FIFO is empty

20.9.3. Data input register (CAU_DI)

Address offset: 0x08
Reset value: 0x0000 0000

The data input register is used to transfer plaintext or ciphertext blocks into the input FIFO for processing. The MSB is firstly written into the FIFO and the LSB is the last one. If the CAUEN is 0 and the input FIFO is not empty, when it is read, then the first data in the FIFO is popped out and returned. If the CAUEN is 1, the returned value is undefined. Once it is read, then the FIFO must be flushed.

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

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

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

Bits	Fields	Descriptions
31:0	DI[31:0]	Data input Write these bits will write data to IN FIFO, read these bits will return IN FIFO value if CAUEN is 0, or it will return an undefined value

20.9.4. Data output register (CAU_DO)

Address offset: 0x0C
Reset value: 0x0000 0000

The data output register is a read only register. It is used to receive plaintext or ciphertext results from the output FIFO. Similar to CAU_DI, the MSB is read at first while the LSB is read at last.

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

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

Bits	Fields	Descriptions
31:0	DO[31:0]	Data output These bits are read only, read these bits return OUT FIFO value.

20.9.5. DMA enable register (CAU_DMAEN)

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														DMAOEN	DMAIEN
														rw	rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	DMAOEN	DMA output enable 0: DMA for OUT FIFO data is disabled 1: DMA for OUT FIFO data is enabled
0	DMAIEN	DMA input enable 0: DMA for IN FIFO data is disabled 1: DMA for IN FIFO data is enabled

20.9.6. Interrupt enable register (CAU_INTEN)

Address offset: 0x14

Reset value: 0x0000 0000

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

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

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

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	OINTEN	OUT FIFO interrupt enable 0: OUT FIFO interrupt is disable 1: OUT FIFO interrupt is enable
0	IINTEN	IN FIFO interrupt enable 0: IN FIFO interrupt is disable 1: IN FIFO interrupt is enable

20.9.7. Status register 1 (CAU_STAT1)

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															

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

Reserved	OSTA	ISTA
	r	r

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	OSTA	OUT FIFO interrupt status 0: OUT FIFO interrupt status not pending 1: OUT FIFO interrupt status pending
0	ISTA	IN FIFO interrupt status 0: IN FIFO interrupt not pending 1: IN FIFO interrupt flag pending

20.9.8. Interrupt flag register (CAU_INTF)

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														OINTF	IINTF
														r	r

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	OINTF	OUT FIFO enabled interrupt flag 0: OUT FIFO Interrupt not pending 1: OUT FIFO Interrupt pending
0	IINTF	IN FIFO enabled interrupt flag 0: IN FIFO Interrupt not pending 1: IN FIFO Interrupt pending when CAUEN is 1

20.9.9. Key registers (CAU_KEY0…3(H / L))

Address offset: 0x20 to 0x3C
Reset value: 0x0000 0000

This registers have to be accessed by word (32-bit), and all of them must be written when BUSY is 0.

In DES mode, only CAU_KEY1 is used.

In TDES mode, CAU_KEY1, CAU_KEY2 and CAU_KEY3 are used.

In AES-128 mode, KEY2H[31:0] || KEY2L[31:0] is used as AES_KEY[0:63], and KEY3H[31:0] || KEY3L[31:0] is used as AES_KEY[64:127].

In AES-192 mode, KEY1H[31:0] || KEY1L[31:0] is used as AES_KEY[0:63], KEY2H[31:0] || KEY2L[31:0] is used as AES_KEY[64:127], and KEY3H[31:0] || KEY3L[31:0] is used as AES_KEY[128:191].

In AES-256 mode, KEY0H[31:0] || KEY0L[31:0] is used as AES_KEY[0:63], KEY1H[31:0] || KEY1L[31:0] is used as AES_KEY[64:127], KEY2H[31:0] || KEY2L[31:0] is used as AES_KEY[128:191], and KEY3H[31:0] || KEY3L[31:0] is used as AES_KEY[192:255].

NOTE: "||" is a concatenation operator. For example, X || Y denotes the concatenation of two bit strings X and Y.

CAU_KEY0H

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY0H[31:16]															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY0H[15:0]															
w															

CAU_KEY0L

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY0L[31:16]															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY0L[15:0]															
w															

CAU_KEY1H

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY1H[31:16]															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY1H[15:0]															

w

CAU_KEY1L

Address offset: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY1L[31:16]															
w															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY1L[15:0]															
w															

CAU_KEY2H

Address offset: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY2H[31:16]															
w															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY2H[15:0]															
w															

CAU_KEY2L

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY2L[31:16]															
w															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY2L[15:0]															
w															

CAU_KEY3H

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY3H[31:16]															
w															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY3H[15:0]															
w															

CAU_KEY3L

Address offset: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY3L[31:16]															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY3L[15:0]															
w															

Bits	Fields	Descriptions
31:0	KEY0...3(H / L)	The key for DES, TDES, AES

20.9.10. Initial vector registers (CAU_IV0…1(H / L))

Address offset: 0x40 to 0x4C

Reset value: 0x0000 0000

This registers have to be accessed by word (32-bit), and all of them must be written when BUSY is 0.

In DES / TDES mode, IV0H is the leftmost bits, and IV0L is the rightmost bits of the initialization vectors.

In AES mode, IV0H is the leftmost bits, and IV1L is the rightmost bits of the initialization vectors.

CAU_IV0H

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV0H[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IV0H[15:0]															
rw															

CAU_IV0L

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV0L[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IV0L[15:0]															
rw															

CAU_IV1H

Address offset: 0x48

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV1H[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IV1H[15:0]															
rw															

CAU_IV1L

Address offset: 0x4C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV1L[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IV1L[15:0]															
rw															

Bits	Fields	Descriptions
31:0	IV0...1(H / L)	The initialization vector for DES, TDES, AES

20.9.11. GCM or CCM mode context switch register x (CAU_GCMCCMCTXSx) (x = 0…7)

Address offset: 0x50 to 0x6C

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
CTXx[31:16]															
rw															

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

Bits	Fields	Descriptions

31:0	CTXx[31:0]	The internal status of the CAU core. Read and save the register data when a high-priority task is coming to be processed, and restore the saved data back to the registers to resume the suspended processing. **Note:** These registers are used only when GCM, GMAC, or CCM mode is selected.

20.9.12. GCM mode context switch register x (CAU_GCMCTXSx) (x = 0…7)

Address offset: 0x70 to 0x8C
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
CTXx[31:16]															
rw															

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

Bits	Fields	Descriptions
31:0	CTXx[31:0]	The internal status of the CAU core. Read and save the register data when a high-priority task is coming to be processed, and restore the saved data back to the registers to resume the suspended processing. **Note:** These registers are used only when GCM or GMAC mode is selected.

21. Hash Acceleration Unit (HAU)

21.1. Overview

The hash acceleration unit is used for information security. The secure hash algorithm (SHA-1, SHA-224, SHA-256), the message-digest algorithm (MD5) and the keyed-hash message authentication code (HMAC) algorithm are supported for various applications. The digest will be computed and the length is 160 / 224 / 256 / 128 bits for a message up to (2^{64} - 1) bits computed by SHA-1, SHA-224, SHA-256 and MD5 algorithms respectively. In HMAC algorithm, SHA-1, SHA-224, SHA-256 or MD5 will be called twice as hash functions and authenticating messages can be produced.

The HAU is fully compliant implementation of the following standards:

- Federal Information Processing Standards Publication 180-2 (FIPS PUB 180-2)
- Secure Hash Standard specifications (SHA-1, SHA-224, SHA-256)
- Internet Engineering Task Force Request for Comments number 1321 (IETF RFC 1321) specifications (MD5)

21.2. Characteristics

- 32-bit AHB slave peripheral.
- High performance of computation of hash algorithms.
- Little-endian data representation.
- Multiple data types are supported, including no swapping, half-word swapping, byte swapping, and bit swapping with 32-bit data words.
- Automatic data padding to fill the 512-bit message block for digest computation.
- DMA transfer is supported.
- Hash / HMAC process suspended mode.

21.3. HAU data type

The hash acceleration unit receives data words of 32 bits at a time, while they are processed in 512-bits blocks. For each input word, according to the data type, the data could be bit / byte / half-word / no swapped before they are transferred into the hash acceleration core. The same swapping operation should be also performed on the core output data before they are collected. Note the least-significant data always occupies the lowest address location no matter which data type is configured, because the system memory is little-endian. However, the computation of SHA-1, SHA-224 and SHA-256 are big-endian.

Figure 21-1. DATAM No swapping and Half-word swapping and ***Figure 21-2. DATAM Byte swapping and Bit swapping*** illustrate the data swapping according to different data

types.

Figure 21-1. DATAM No swapping and Half-word swapping

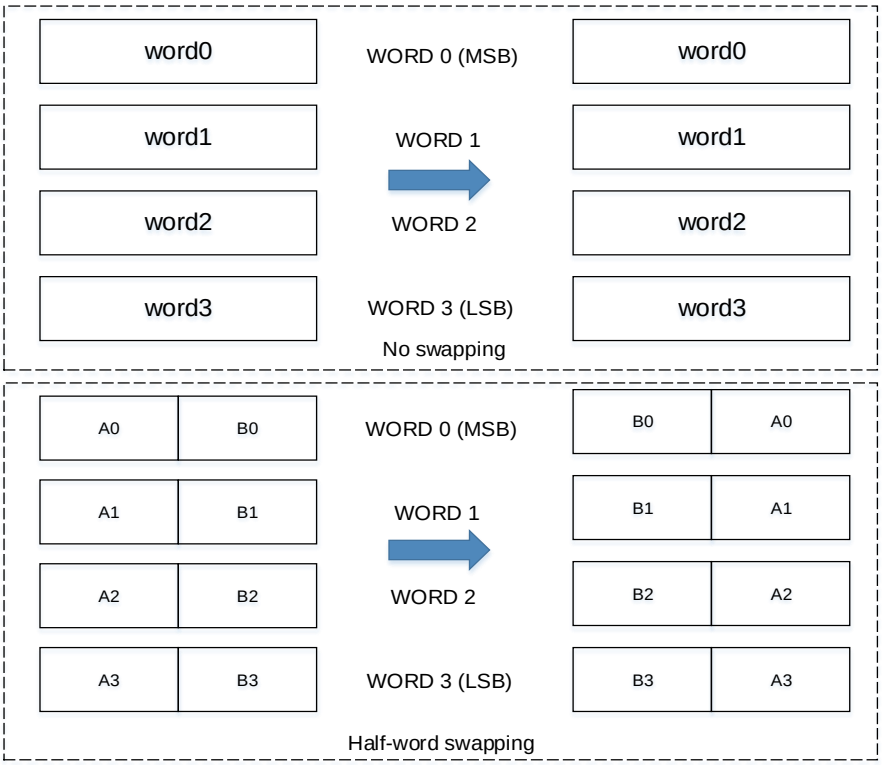
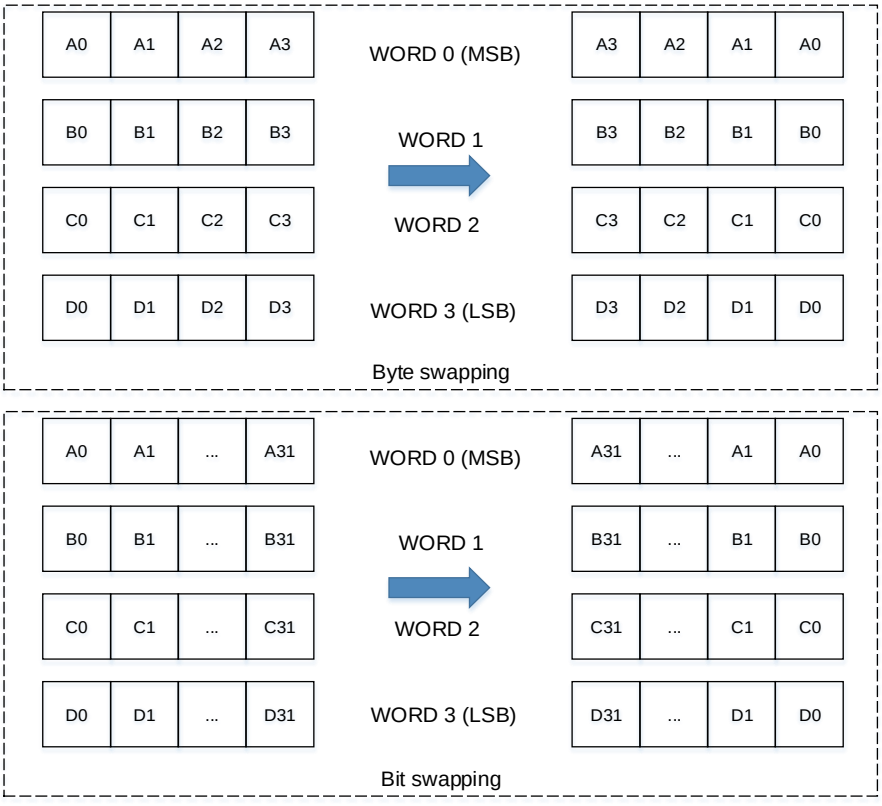


Figure 21-2. DATAM Byte swapping and Bit swapping

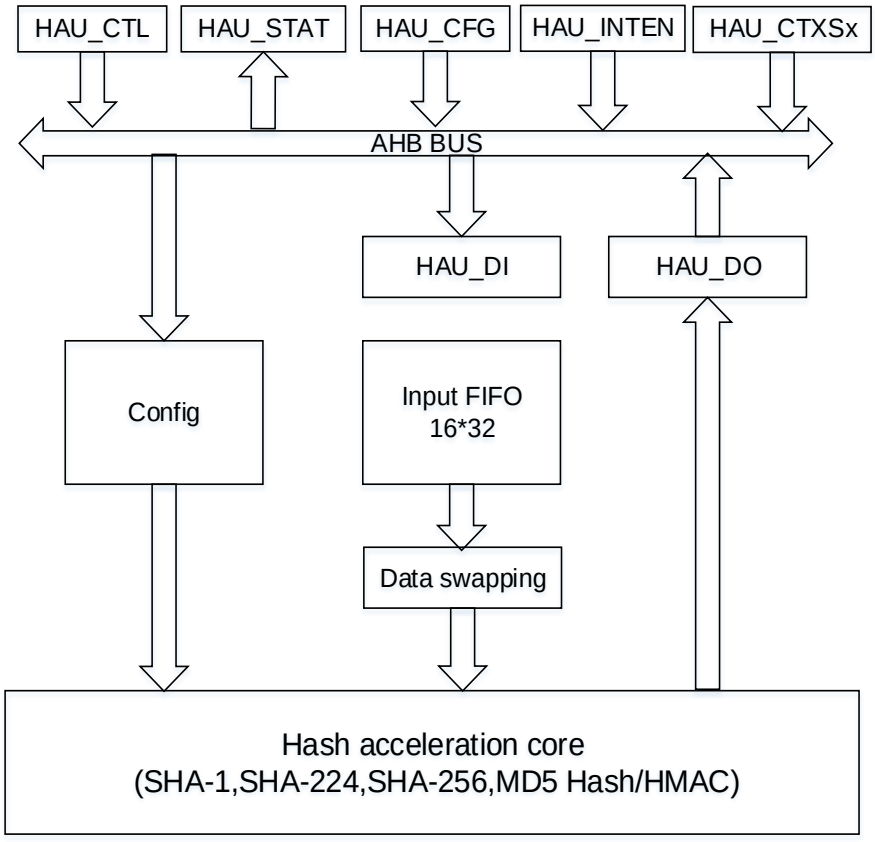


21.4. HAU core

The hash acceleration unit is used to compute condensed information of input messages with secure hash algorithms. The digest result has a length of 160 / 224 / 256 / 128 bits for a message up to (264-1) bits computed by SHA-1, SHA-224, SHA256 and MD5 algorithms respectively. It can be used to generate or verify the signature of a message with a higher efficiency because of the much simpler of the information.

A message which need to be processed in the HAU should be considered as bit information. And the length is the number of bits of the message. The information security is ensured because that, to find the original message using the digest is computationally impossible and, the result will be completely different with any change to the input message.

Figure 21-3. HAU block diagram



21.4.1. Automatic data padding

The input message should be padded first so that the number of bits in the input of the HAU core can be an integral multiple of 512. First of all, a “1” is added to follow the last bit of the input message, and then several “0” should be padded to ensure the result modulo 512 is 448, at last, a 64-bit length information of input is added.

After the message padding is correctly performed, the VBL bits in the HAU_CFG register is

configured as the 64-bit length value above, and CALEN bit in the HAU_CFG register can be set 1 to start the calculation of the digest of the last block.

Data Padding Example: The input message is “HAU”, which ASCII hexadecimal code is:

484155

Then the VBL bits in the HAU_CFG register is set as decimal 24 because of the valid bit length. A “1” is added at bit location 24 then, and several “0” are padded so that the result modulo 512 is 448, the hexadecimal result is as follows:

48415580 00000000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000

After that, a 64-bit length information of the input message is padded, which hexadecimal value is 18, and the final result will be:

48415580 00000000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000018

21.4.2. Digest computing

After data padding, for each block calculation of HAU, 512 bits are written into the HAU core by DMA or CPU. To start the processing of the HAU core, the peripheral must obtain the information as to whether the HAU_DI register contains the last bits of the message or not. This can be confirmed with the status of the input FIFO and the HAU_DI register.

When DMA is used to transfer data:

The status of the block transfer is automatically interpreted with the information from the DMA controller. And padding and digest computation are performed automatically as if CALEN bit in the HAU_CFG register is set as 1.

Note: If hash message is large files and multiple DMA transfers are needed, then MDS bit should be set as 1. And the VBL bits need to be set before the transfer. The CALEN bit is not set automatically after an intermediate DMA transfer completed. Only when the last DMA transfer is processing, the MDS bit is cleared so that the CALEN bit is automatically set after data transferring.

Otherwise, the MDS bit is set as 0. And the CALEN bit is set automatically after a DMA transfer. Also, VBL bits need to set before the DMA transfer.

When CPU is used to transfer data without DMA:

- The intermediate block computing can be started when HAU_DI is filled with another new word of the next block.
- The last block computing can be started when CALEN bit in the HAU_CFG register is 1.

21.4.3. Hash mode

The hash mode is selected when the HMS bit in the HAU_CTL register is set as 0. And when the START bit in the HAU_CTL register is 1, SHA-1, SHA-224, SHA-256 and MD5 mode computation is chosen by the ALGM bits.

After a message block of 512 bit has been received through the HAU_DI register and the input FIFO, the processor starts the calculation with the information from DMA or the status of the CALEN bit.

The results can be finally read from the HAU_DO0..7 registers.

21.4.4. HMAC mode

HMAC mode is used for message authentication with a unique key chosen by the user. More information about the HMAC specifications please refer to “HMAC: keyed-hashing for message authentication, H. Krawczyk, M. Bellare, R. Canetti, February 1997”.

The HMAC algorithm can be represented as:

HMAC(input) = HASH[((key | opad) XOR 0x5c) | HASH(((key | ipad) XOR 0x36) | input)]

where “ipad” and “opad” are used to extend the key to 512 bits with several “0” and “|” is the concatenation operator.

There are four different phases in the HMAC mode:

1. Configure the HMS bit in the HAU_CTL register as 1 and set the ALGM bits as the desired algorithm. If the key size is longer than 64 bytes, then the KLM bit in the HAU_CTL register should also be set. After that, start the HAU core by set the START bit.
2. The key is used as the input message to complete the calculation in HASH mode.
3. The new key used for the inner hush function is elaborated when the last word is accessed and computation has started.
4. After the first hash round, HAU core starts to receive the key for the outer hash function, usually, the outer hash function uses the same new key as the inner hash function. And when the last word of the key is entered and computation starts, the results are available in the HAU_DO registers.

21.5. HAU suspended mode

It is possible to suspend HASH or HMAC operation to perform a high-prior task first, then after the high-prior task is finished, resume the suspended operation.

When suspending the current task, it is necessary to save the context of the current task from registers to memory, and then the task can be resumed by restoring the context from memory to the HAU registers.

The following steps can be performed to complete the HAU process of the suspended data blocks.

21.5.1. Transfer data by CPU

1. Stop the current data transmission and calculation. Wait for BUSY = 0, and wait for DIF = 1 when NWIF[3:0] is larger than 0 (do not wait for DIF = 1 when NWIF[3:0] is 0).Only when no data block is processing, users can save context.
2. Save the configuration. Save the content of HAU_INTEN, HAU_CFG, HAU_CTL, HAU_CTXS0 to HAU_CTXS37 (HAU_CTXS0 to HAU_CTXS53 when HMAC operation is processing) registers to memory.
3. Configure and process the new message.
4. Restore the process before. Restore the content from memory to HAU_INTEN, HAU_CFG and HAU_CTL registers.
5. Resume the message calculation. Set START bit of HAU_CTL register to 1, to restart a new message digest calculation.
6. Resume the previous core state. Restore the content from memory to HAU_CTXS0 ~ HAU_CTXS37 (HAU_CTXS0 ~ HAU_CTXS53 when HMAC operation is to be resumed) registers.
7. Continue the operation from where it suspended before.

21.5.2. Transfer data by DMA

1. Wait for BUSY = 0, then if the CCF bit of HAU_STAT register is set, the proceeding context switch is no longer need, otherwise wait for BUSY = 1 again.
2. Stop the current data transfer. Disable DMA1 channel 7 data transmission, then clear DMAE bit of HAU_CTL register to disable DMA request.
3. Save the current configuration. Wait for BUSY = 0, then if the CCF bit of HAU_STAT register is set, the proceeding context switch is no longer need, otherwise save the content of HAU_INTEN, HAU_CFG, HAU_CTL, HAU_CTXS0 to HAU_CTXS37 (HAU_CTXS0 to HAU_CTXS53 when HMAC operation is processing) registers to memory.
4. Configure and process the new message.
5. Restore the process before. Restore the content from memory to HAU_INTEN, HAU_CFG and HAU_CTL registers.
6. Resume DMA channel transmission. Reconfigure the DMA channel to transfer data.
7. Resume the message calculation. Set START bit of HAU_CTL register to 1, to restart a new message digest calculation.

8. Resume the previous core state. Restore the content from memory to HAU_CTXS0 ~ HAU_CTXS37 (HAU_CTXS0 ~ HAU_CTXS53 when HMAC operation is to be resumed) registers.
9. Set DMAE bit of HAU_CTL register to 1, continue the operation from where it suspended before.

Note: If the value of NWIF[3:0] bits of HAU_CTL register is 0, it means the context switch occurs between two data blocks, at the time when the previous block is completely processed, and the next block has not been pushed into input FIFO, so there is no need to save and restore HAU_CTXS22 ~ HAU_CTXS37 registers.

21.6. HAU interrupt

There are two types of interrupt registers in HAU, which are both in HAU_STAT register. In HAU, the interrupt is used to indicate the situation of the input FIFO and the status of whether the digest calculation is completed.

Any of interrupts can be enabled or disabled by configuring the HAU interrupt enable register HAU_INTEN. Value 1 of the register bits enable the interrupts.

21.6.1. Input FIFO interrupt

When the processing of data pushed in the input FIFO is completed, then DIF is asserted. If input FIFO interrupt is enabled, when DIF is asserted, input FIFO interrupt will be asserted.

21.6.2. Calculation completion interrupt

When the digest calculation is finished, then CCF is asserted. If calculation completion interrupt is enabled, when CCF is asserted, calculation completion interrupt will be asserted.

21.7. Register definition

HAU base address: 0x4C06 0400

21.7.1. HAU control register (HAU_CTL)

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													ALGM[1]	Reserved	KLM
													rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		MDS	DINE	NWIF[3:0]				ALGM[0]	HMS	DATAM[1:0]		DMAE	START	Reserved	
		rw	r	r				rw	rw	rw		rw	w		

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18	ALGM[1]	Algorithm selection bit 1
17	Reserved	Must be kept at reset value.
16	KLM	Key length mode 0: Key length ≤ 64 bytes 1: Key length > 64 bytes **Note:** This bit must be changed when no computation is processing.
15:14	Reserved	Must be kept at reset value.
13	MDS	Multiple DMA Selection Set this bit if hash message is large files and multiple DMA transfers are needed. 0: Single DMA transfers needed and CALEN bit is automatically set at the end of a DMA transfer 1: Multiple DMA transfers needed and CALEN bit is not automatically set at the end of a DMA transfer
12	DINE	DI register not empty 0: The DI register is empty 1: The DI register is not empty **Note:** This bit is cleared when START bit or CALEN bit is set as 1.
11:8	NWIF[3:0]	Number of words in the input FIFO **Note:** These bits are cleared when START bit set or a digest calculation starts (CALEN bit is set as 1, or DMA end of transfer)

7	ALGM[0]	Algorithm selection bit 0 This bit and bit 18 of CTL are written by software to select the SHA-1, SHA-224, SHA256 or the MD5 algorithm: 00: Select SHA-1 algorithm 01: Select MD5 algorithm 10: Select SHA224 algorithm 11: Select SHA256 algorithm
6	HMS	HAU mode selection, must be changed when no computation is processing 0: HASH mode selected 1: HMAC mode selected. If the key length is longer than 64 bytes, then KLM bit must also be set
5:4	DATAM[1:0]	Data type mode Defines the format of the data entered into the HAU_DI register: 00: No swapping. The data written to HAU_DI is direct write to FIFO without swapping. 01: Half-word swapping. The data written into HAU_DI need half-word swapping before write to FIFO. 10: Bytes swapping. The data written into HAU_DI need bytes swapping before write to FIFO. 11: Bit swapping. The data written into HAU_DI need bytes swapping before write to FIFO.
3	DMAE	DMA enable 0: DMA disabled 1: DMA enabled **Note:** 1. This bit is cleared when transferring the last data of the message, but not cleared because of START. 2. When DMA is transferring, writing 0 to this bit will not stop the current transfer until the transfer is completed or START is set as 1.
2	START	Start the digest calculation 1: Start the digest of a new message 0: No effect **Note:** Reading this bit always returns 0.
1:0	Reserved	Must be kept at reset value.

21.7.2. HAU data input register (HAU_DI)

Address offset: 0x04
Reset value: 0x0000 0000

The data input register is used to transfer message with 512-bit blocks into the input FIFO for processing. Any new write operation to this register will be extended while the digest calculation is in process until it has been finished.

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

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

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

Bits	Fields	Descriptions
31:0	DI[31:0]	Message data input When write to these registers, the current content pushed to IN FIFO and new value updates. When read, returns the current content.

21.7.3. HAU configuration register (HAU_CFG)

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							CALEN	Reserved			VBL[4:0]				
							w				rw				

Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	CALEN	Digest calculation enable 0: No calculation 1: Start data padding with VBL prepared previously. Start the calculation of the last digest **Note:** Reading this bit always returns 0.
7:5	Reserved	Must be kept at reset value.
4:0	VBL[4:0]	Valid bits length in the last word 0x00: All 32 bits of the last data written to HAU_DI after data swapping are valid 0x01: Only bit [31] of the last data written to HAU_DI after data swapping are valid 0x02: Only bits [31:30] of the last data written to HAU_DI after data swapping are valid 0x03: Only bits [31:29] of the last data written to HAU_DI after data swapping are valid ...

0x1F: Only bits [31:1] of the last data written to HAU_DI after data swapping are valid

Note: These bits must be configured before setting the CALEN bit.

21.7.4. HAU data output register (HAU_DO0…7)

Reset value: 0x0000 0000
This register has to be accessed by word (32-bit)

The data output registers are read only registers. They are used to receive results from the output FIFO. And they are reset by the START bit. Any read access when calculating will be extended until the calculation is completed.

In SHA-1 mode, HAU_DO0…4 are used.
In MD5 mode, HAU_DO0…3 are used.
In SHA-224 mode, HAU_DO0…6 are used.
In SHA-256 mode, HAU_DO0…7 are used.

HAU_DO0

Address offset: 0x0C and 0x310

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DO0[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DO0[15:0]															
r															

HAU_DO1

Address offset: 0x10 and 0x314

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DO1[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DO1[15:0]															
r															

HAU_DO2

Address offset: 0x14 and 0x318

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DO2[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DO2[15:0]															

r

HAU_DO3

Address offset: 0x18 and 0x31C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DO3[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DO3[15:0]															
r															

HAU_DO4

Address offset: 0x1C and 0x320

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DO4[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DO4[15:0]															
r															

HAU_DO5

Address offset: 0x324

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DO5[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DO5[15:0]															
r															

HAU_DO6

Address offset: 0x328

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DO6[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DO6[15:0]															
r															

HAU_DO7

Address offset: 0x32C

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

DO7[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DO7[15:0]															
r															

Bits	Fields	Descriptions
31:0	DO0..7[31:0]	Message digest result of hash algorithm

21.7.5. HAU interrupt enable register (HAU_INTEN)

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														CCIE	DIIE
														rw	rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	CCIE	Calculation completion interrupt enable 0: Calculation completion interrupt is disabled 1: Calculation completion interrupt is enabled
0	DIIE	Data input interrupt enable 0: Data input interrupt is disabled 1: Data input interrupt is enabled

21.7.6. HAU status and flag register (HAU_STAT)

Address offset: 0x24

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												BUSY	DMAS	CCF	DIF
												r	r	rc_w0	rc_w0

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	BUSY	Busy bit 0: No processing 1: Data block is in process
2	DMAS	DMA status 0: DMA is disabled (DMAE = 0) and no transfer is processing 1: DMA is enabled (DMAE = 1) or a transfer is processing
1	CCF	Digest calculation completion flag 0: Digest calculation is not completed 1: Digest calculation is completed
0	DIF	Data input flag 0: A data is written to data input register 1: A data processing is completed (only the data in input FIFO will be processed)

21.7.7. Context switch register x (HAU_CTXSx) (x = 0...53)

Address offset: 0xF8 + 0x04 × x, (x = 0...53)

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
CTXx[31:16]															
rw															

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

Bits	Fields	Descriptions
31:0	CTXx[31:0]	The complete internal status of the HAU core. Read and save the register data when a high-priority task is coming to be processed, and restore the saved data back to the registers to resume the suspended processing.

22. Public Key Cryptographic Acceleration Unit (PKCAU)

22.1. Overview

Public key encryption is also called asymmetric encryption, asymmetric encryption algorithms use different keys for encryption and decryption. The Public Key Cryptographic Acceleration Unit (PKCAU) can accelerate RSA (Rivest, Shamir and Adleman), Diffie-Hellmann (DH key exchange) and ECC (elliptic curve cryptography) in GF(p) (Galois domain). These operations are performed in the Montgomery domain to improve computational efficiency.

22.2. Characteristics

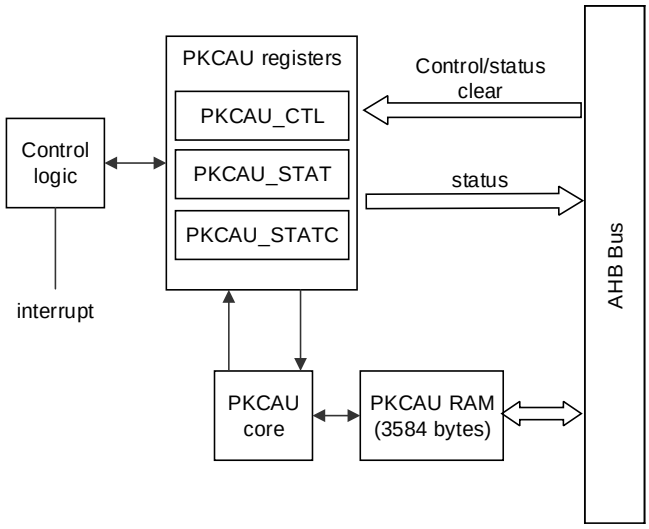
- Support RSA/DH algorithms with up to 3136 bits of operands.
- Support ECC algorithm with up to 640 bits of operands.
- RSA modular exponentiation, RSA CRT exponentiation.
- ECC scalar multiplication, check point on elliptic curve.
- ECDSA (Elliptic Curve Digital Signature Algorithm) signature and verification.
- Support Montgomery multiplication, accelerate RSA, DH and ECC operations.
- Embedded RAM of 3584 bytes.
- Conversion between the Montgomery domain and the natural domain.
- PKCAU is a 32 bit peripheral, only 32-bit access is supported.

22.3. Function overview

The Public Key Cryptographic Acceleration Unit (PKCAU) is used for accelerating RSA, DH and ECC operations in GF(p) domain. The PKCAU module contains PKCAU RAM, PKCAU core, and PKCAU registers. The PKCAU RAM is used to store the parameters required by the operation and save the calculation result after the calculation is completed.

Figure 22-1. PKCAU module block diagram below provides details on the internal configuration of the PKCAU interface.

Figure 22-1. PKCAU module block diagram



22.3.1. Operands

If the RSA operand size is ROS, the modulus length is ML, then the data size is ROS = (ML/32+1) words. If the ECC operand size is EOS, the prime modulus length is ML, then the data size is EOS = (ML/32+1) words.

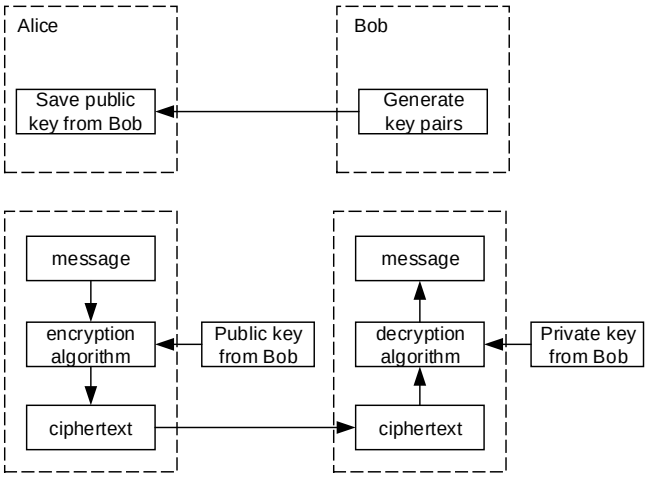
The PKCAU supports RSA/DH algorithms with up to 3136 bits (98 words) of operands and ECC algorithm with up to 640 bits (20 words) of operands.The maximum ROS is 99 words, and the maximum EOS is 21 words.

When writing the input parameters to the PKCAU RAM, a word 0x00000000 must be added. The PKCAU RAM is little-endian. For example, when writing the input parameter x_p of ECC P256 for ECC scalar multiplication to PKCAU RAM, the modulus length is 8 words, address offset 0x55C stores the lowest byte, address offset 0x578 stores the highest byte. Address offset 0x57C stores word 0x00000000.

22.3.2. RSA algorithm

RSA algorithm is a common public key cryptography algorithm and the most widely used asymmetric cryptography algorithm. The RSA algorithm flow is shown in ***Figure 22-2. Flow chart of RSA algorithm***.

Figure 22-2. Flow chart of RSA algorithm



A complete public key crypto system includes key pairs (public and private keys), encryption algorithms and decryption algorithms.

Generate key pairs of RSA

1. Select two large primes p and q (p≠q);
2. Calculate n=p×q, n is the modulus of public and private keys;
3. Calculate L=∅ (n)=(p-1)(q-1), ∅ (n) is euler function;
4. Select e, 1<e<L, e and L must be relatively prime;
5. Calculate d, 1<d<L and e*d mod L = 1.

The parameters show in ***Table 22-1. Parameters of RSA algorithm*** can be obtained by the above calculation.

Table 22-1. Parameters of RSA algorithm

Parameters	Description
n	modulus
e	public exponent
d	private exponent
(n,e)	public key
(n,d)	private key

RSA encryption

Bob generates a key pair that conforms to RSA algorithm standard, including public key and private key. Bob sends the public key to Alice, and holds the private key. Alice can encrypt the message m through the public key from Bob and obtain the ciphertext c. Then Alice sends the ciphertext to Bob. The ciphertext is $c = m^e \bmod n$.

RSA decryption

After receiving the ciphertext, Bob decrypts the ciphertext to get the plaintext by the private

key. The decryption process is $m = c^d \bmod n$.

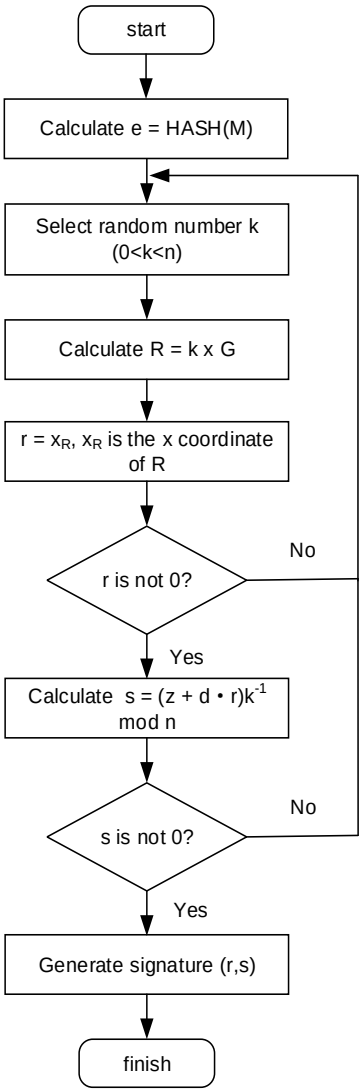
22.3.3. ECC algorithm

Suppose the message is M, d is the private key, G is the base point of the chosen elliptic curve, Q is a point of the chosen elliptic curve,with a prime order n. The hash function is HASH(), z is the Ln leftmost bits of HASH (M), where Ln is the bit length of the order n. The ECDSA sign and verification are detailed as follows:

ECDSA sign

The signature result of ECDSA consists of r and s. The process to generate ECDSA signature is shown in ***Figure 22-3. Flow chart of ECDSA sign***.

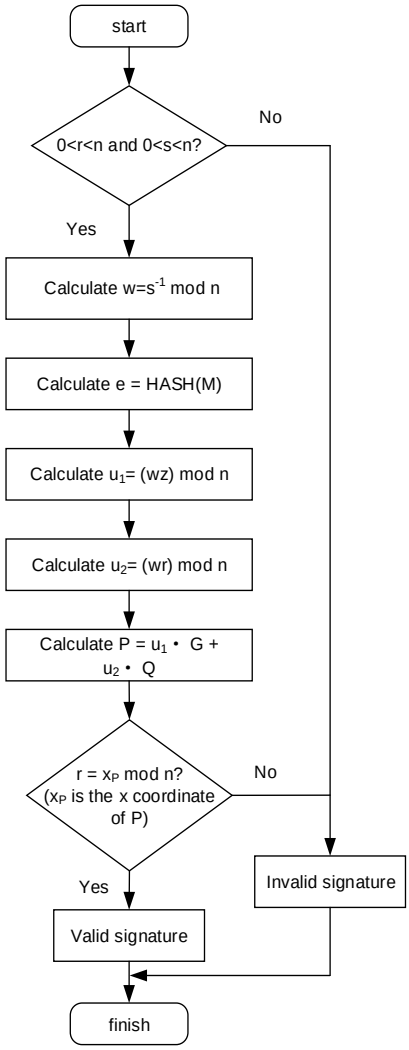
Figure 22-3. Flow chart of ECDSA sign



ECDSA verification

Before verifying the signature, be sure to get the signer's public key, message, and signature. The process to generate ECDSA signature is shown in ***Figure 22-4. Flow chart of ECDSA verification***.

Figure 22-4. Flow chart of ECDSA verification



Note: The HASH in the above diagram is the agreed cryptographic hash function.

22.3.4. Integer arithmetic operations

The integer arithmetic operation can be selected by configuring the MODSEL[5:0] in PKCAU_CTL register. The operation modes to be selected is shown in ***Table 22-2. Integer arithmetic operations***.

Table 22-2. Integer arithmetic operations

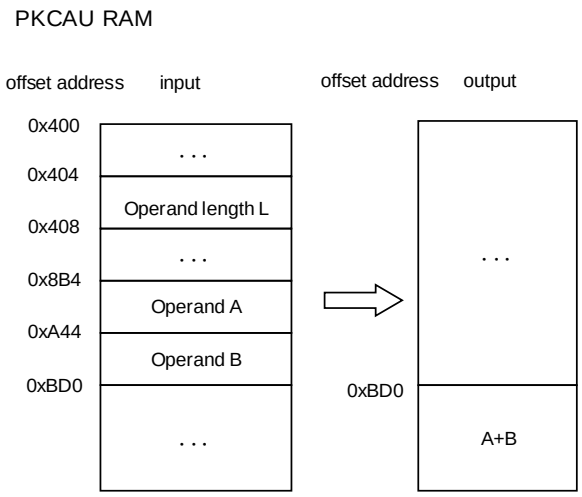
MODSEL[5:0]	Operation modes
000000	calculate Montgomery parameter and then modular exponentiation

MODSEL[5:0]	Operation modes
000001	calculate Montgomery parameter only
000010	modular exponentiation (the Montgomery parameter must be preloaded)
000111	RSA CRT exponentiation
001000	Modular inversion
001001	Arithmetic addition
001010	Arithmetic subtraction
001011	Arithmetic multiplication
001100	Arithmetic comparison
001101	Modular reduction
001110	Modular addition
001111	Modular subtraction
010000	Montgomery multiplication

Arithmetic addition

The arithmetic addition operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "001001". The operation declaration is shown in ***Figure 22-5. Arithmetic addition***. The operation result is “result = A+B”.

Figure 22-5. Arithmetic addition



$0 \le A < 2^{L}$, $0 \le B < 2^{L}$, $0 \le result < 2^{L+1}$, $0 < L \le 3136$.

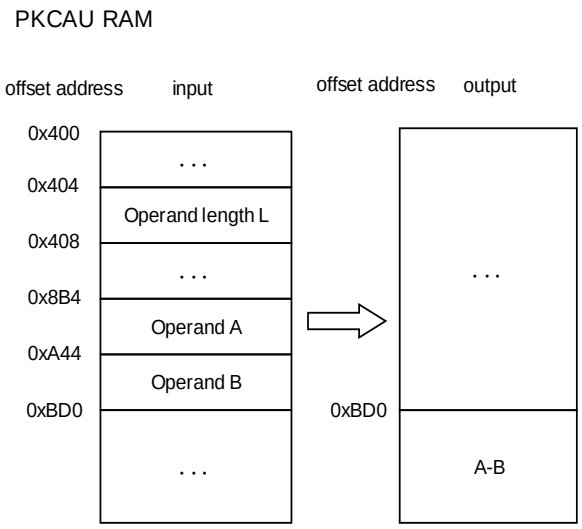
Arithmetic subtraction

The arithmetic subtraction operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "001010". The operation declaration is shown in ***Figure 22-6. Arithmetic subtraction***.

If A≥B, the operation result is “result = A-B”;

If A<B, the operation result is “result = $A-B+2^{L+ceil(L\%32)}$”

Figure 22-6. Arithmetic subtraction

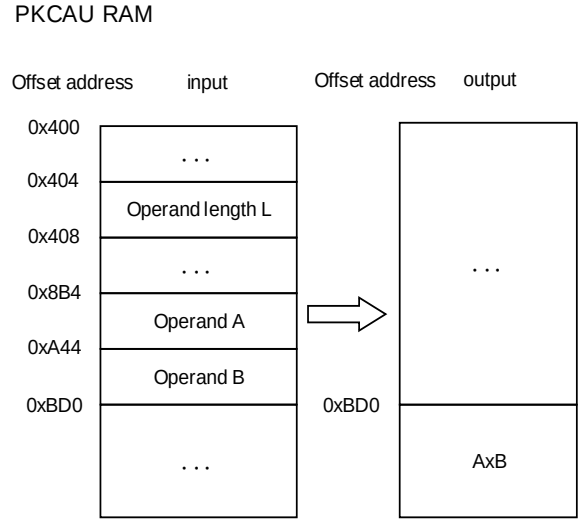


$0 \le A < 2^L$, $0 \le B < 2^L$, $0 \le result < 2^L$, $0 < L \le 3136$.

Arithmetic multiplication

The arithmetic multiplication operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "001011". The operation declaration is shown in ***Figure 22-7. Arithmetic multiplication***. The operation result is “result = A×B”.

Figure 22-7. Arithmetic multiplication



$0 \le A < 2^L$, $0 \le B < 2^L$, $0 \le result < 2^{2L}$, $0 < L \le 3136$.

Arithmetic comparison

The arithmetic comparison operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "001100". The operation declaration is shown in ***Figure 22-8. Arithmetic***

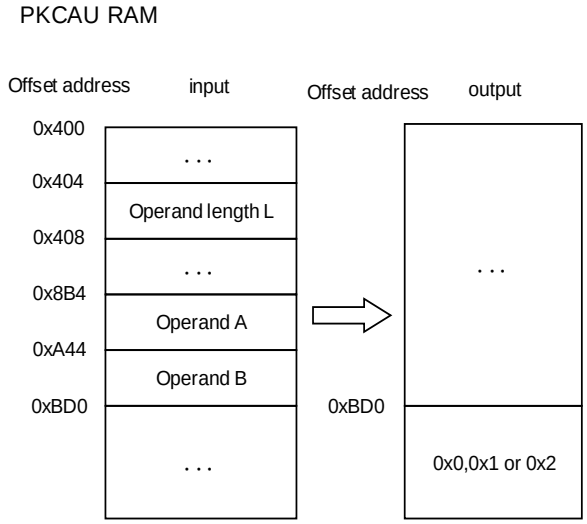
comparison.

If A=B, the operation result is “result =0x0”;

If A>B, the operation result is “result =0x1”;

If A<B, the operation result is “result =0x2”.

Figure 22-8. Arithmetic comparison

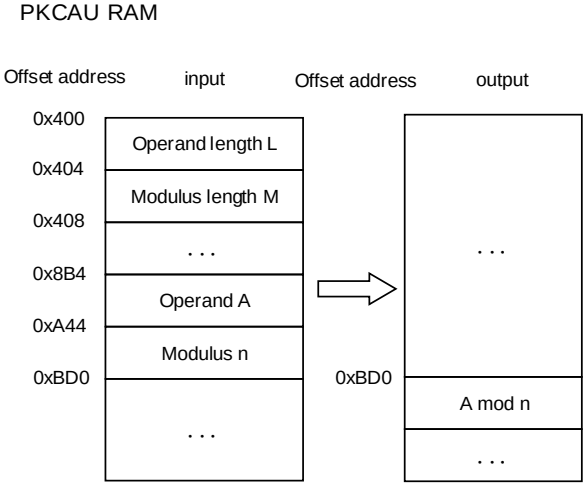


$0 \le A < 2^L$, $0 \le B < 2^L$, result =0x0, 0x01, 0x2, $0 < L \le 3136$.

Modular reduction

The modular reduction operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "001101". The operation declaration is shown in ***Figure 22-9. Modular reduction***. The operation result is “result = A mod n”.

Figure 22-9. Modular reduction

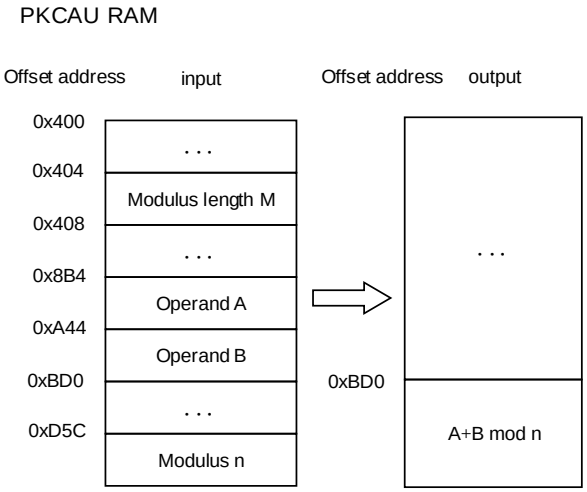


$0 < L \le 3136$, $0 < M \le 3136$, $0 \le A < 2^L$, $0 < n < 2^M$, $0 \le result < n$.

Modular addition

The modular addition operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "001110". The operation declaration is shown in ***Figure 22-10. Modular addition***. The operation result is “result = A+B mod n”.

Figure 22-10. Modular addition



$0 \le A < n$, $0 \le B < n$, $0 \le result < n$, $0 < n < 2^M$, $0 < M \le 3136$.

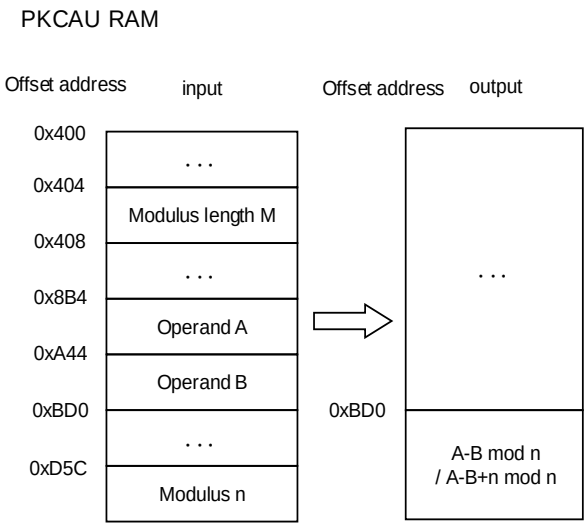
Modular subtraction

The modular subtraction operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "001111". The operation declaration is shown in ***Figure 22-11. Modular subtraction***.

If $A \ge B$, the operation result is “result = A-B mod n”;

If $A < B$, the operation result is “result = A-B+n mod n”.

Figure 22-11. Modular subtraction



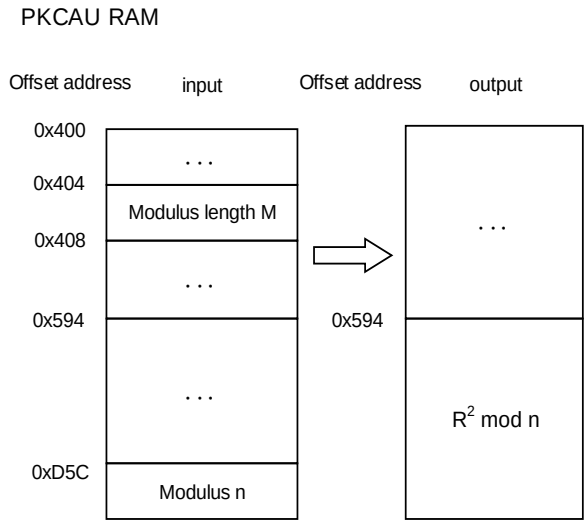
$0 \le A < n$, $0 \le B < n$, $0 \le result < n$, $0 < n < 2^M$, $0 < M \le 3136$.

Montgomery parameter calculation

PKCAU needs to use the Montgomery parameter (R^2 mod n) to convert the operands to Montgomery residue system representation.

The Montgomery parameter calculation operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "000001". The operation declaration is shown in ***Figure 22-12. Montgomery parameter calculation***.

Figure 22-12. Montgomery parameter calculation



$0 < M \le 3136$, $1 < n < 2^M$(n must be odd integer).

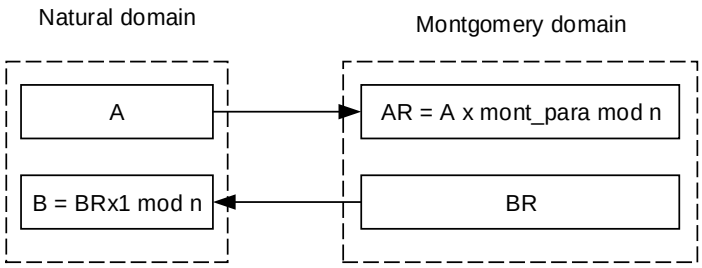
Montgomery multiplication

Suppose A, B and C are in natural domain. “x” function is Montgomery multiplication operation. The two main uses of this opreation are as follows:

1. Mutual mapping between Montgomery domain and natural domain.

As is shown in ***Figure 22-13. Mutual mapping between Montgomery domain and natural domain***, if A is an integer in natural domain, the Montgomery parameter mont_para is R^2 mod n, the result AR = A x mont_para mod n is A in Montgomery domain. In turn, If BR is an integer in Montgomery domain, the calculation result B = BRx1 mod n is in natural domain.

Figure 22-13. Mutual mapping between Montgomery domain and natural domain



2. Perform a modular multiplication operation A x B mod n.

(1) Calculate Montgomery parameter mont_para = R^2 mod n.

(2) Calculate AR = A x mont_para mod n, the output is in Montgomery domain.

(3) Calculate AB = AR x B mod n, the output is in natural domain.

Multiple modular multiplication A x B x C mod n.

(1) Calculate Montgomery parameter mont_para = R^2 mod n.

(2) Calculate AR = A x mont_para mod n, the output is in Montgomery domain.

(3) Calculate BR = B x mont_para mod n, the output is in Montgomery domain.

(4) Calculate ABR = AR x BR mod n, the output is in Montgomery domain.

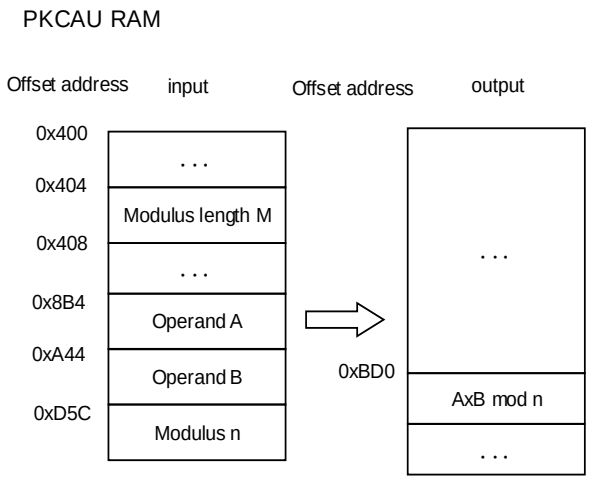
(5) Calculate CR = C x mont_para mod n, the output is in Montgomery domain.

(6) Calculate ABCR = ABR x CR mod n, the output is in Montgomery domain.

(7) Calculate ABC= ABCR x 1 mod n, the output is in natural domain.

The Montgomery multiplication operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "010000". The operation declaration is shown in ***Figure 22-14. Montgomery multiplication***.

Figure 22-14. Montgomery multiplication



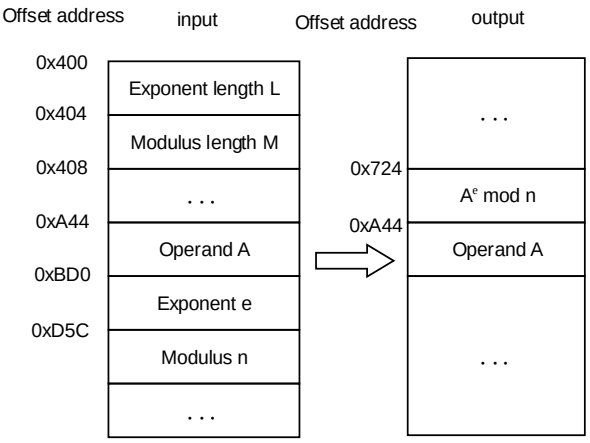
$0 \le A < n$, $0 \le B < n$, $0 < n < 2^M$, $0 < M \le 3136$ (n must be odd integer).

Modular exponentiation

Normal mode

The Modular exponentiation of normal mode operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "000000". The operation declaration is shown in ***Figure 22-15. Modular exponentiation of normal mode***. The operation result is “result = A^e mod n”.

Figure 22-15. Modular exponentiation of normal mode



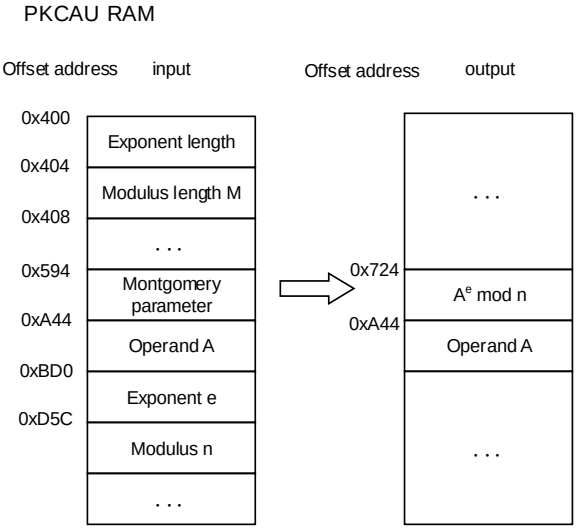
$0 < L \le 3136$, $0 < M \le 3136$, $0 \le A < n$, $0 \le e < 2^L$, $0 \le result < n$, $1 < n < 2^M$ (n must be odd integer).

Fast mode

The Modular exponentiation of fast mode operation is selected by configuring MODSEL[5:0]

in PKCAU_CTL register as "000010". The operation declaration is shown in ***Figure 22-16. Modular exponentiation of fast mode***. The operation result is “result = A^e mod n”.

Figure 22-16. Modular exponentiation of fast mode

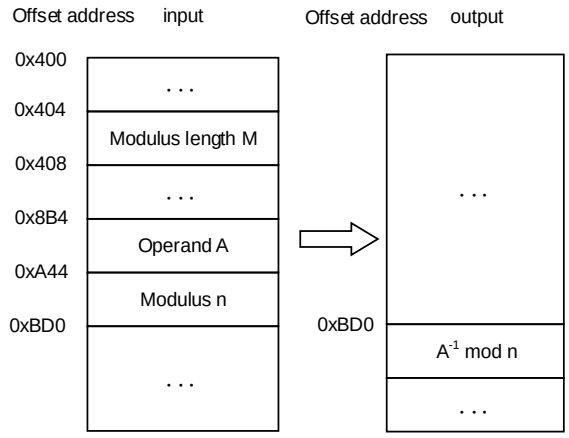


0≤A<n, 0≤e<n, 0≤result<n, 0<n<2^M, 0<M≤3136, 0<Montgomery parameter (R^2 mod n)<n.

Modular inversion

The Modular exponentiation of fast mode operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "001000". The operation declaration is shown in ***Figure 22-17. Modular inversion***. The operation result is “result = A^{-1} mod n”.

Figure 22-17. Modular inversion



0<A<n, 0<result<n, 0<n<2^M, 0<M≤3136.

Note:

1. If the modulus n is prime, for all values of A that satisfy the condition “1≤A<n”, the modular inversion output is valid.

2. If the modulus n is not prime, only when the greatest common divisor of A and n is 1, the modular inversion output is valid.

RSA CRT exponentiation

The RSA CRT exponentiation operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "000111".

p and q are part of the private key, and are primes

$d_P = d \bmod (p-1)$

$d_Q = d \bmod (q-1)$

$q_{inv} = q^{-1} \bmod p$

These parameters above allow the recipient to compute the exponentiation $m = A^d \pmod{pq}$ more efficiently as follows:

$m = A^d \pmod{pq}$

$m_1 = A^{dP} \bmod p$

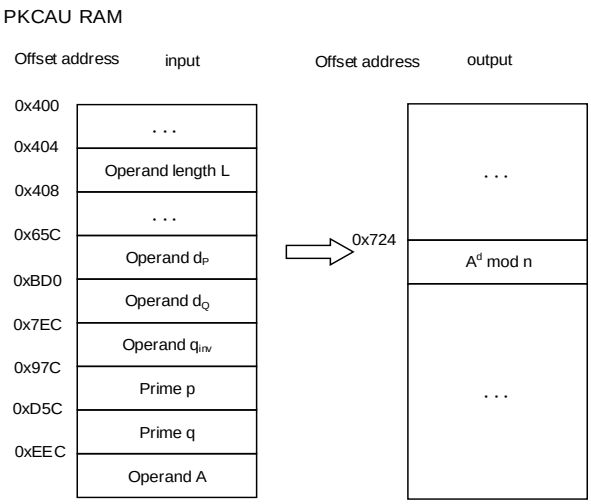
$m_2 = A^{dQ} \bmod p$

$h = q_{inv}(m_1 - m_2) \bmod p,\ m_1 > m_2$

$m = m_2 + hq$

The operation declaration is shown in ***Figure 22-18. RSA CRT exponentiation***. The operation result is "result = A^d mod pq".

Figure 22-18. RSA CRT exponentiation



The range of parameters used by RSA CRT exponentiation operation is shown in ***Table 22-3. Range of parameters used by RSA CRT exponentiation operation***.

Table 22-3. Range of parameters used by RSA CRT exponentiation operation

Parameters		Range
Input	Operand d_P	$0 \le d_P < 2^{L/2}$
	Operand d_Q	$0 \le d_Q < 2^{L/2}$
	Operand q_{inv}	$0 < q_{inv} < 2^{L/2}$
	Prime p	$0 < p < 2^{L/2}$
	Prime q	$0 < q < 2^{L/2}$
	Operand A	$0 \le A < 2^{L}$
Output	result = A^d mod pq	0≤result<pq

22.3.5. Elliptic curve operations in Fp domain

The Elliptic curve operation mode in Fp domain can be selected by configuring the MODSEL[5:0] in PKCAU_CTL register. The operation modes to be selected is shown in ***Table 22-4. Elliptic curve operations in Fp domain***.

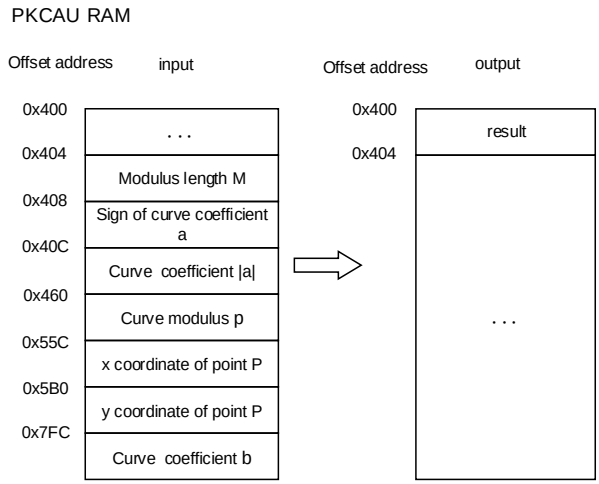
Table 22-4. Elliptic curve operations in Fp domain

MODSEL[5:0]	Operation modes
100000	Montgomery parameter computation then ECC scalar multiplication
100010	ECC scalar multiplication only (Montgomery parameter must be loaded first)
100100	ECDSA sign
100110	ECDSA verification
101000	Point on elliptic curve Fp check

Point on elliptic curve Fp check

The operation is used to check whether P(x,y) is on the $y^2 = x^3+ax+b \bmod p$ in prime domain, where A and B are curve coefficients. The point on elliptic curve check operation in Fp domain is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "101000". The operation declaration is shown in ***Figure 22-19. Point on elliptic curve Fp check***. If the operation result is 0, it indicates that point P is on the elliptic curve. Or else, it indicates that point P is not on the elliptic curve.

Figure 22-19. Point on elliptic curve Fp check



The range of parameters used by point on elliptic curve Fp check operation is shown in ***Table 22-5. Range of parameters used by point on elliptic curve Fp check***.

Table 22-5. Range of parameters used by point on elliptic curve Fp check

Parameters		Range
Input	Modulus length M	0<M≤640
	Sign of curve coefficient a	0x0: positive 0x1: negative
	Curve coefficient |a|	Absolute value |a|<p
	Curve coefficient b	Absolute value |b|<p
	Curve modulus p	Odd prime 0<p≤2^M
	x coordinate of point P	x<p
	y coordinate of point P	y<p

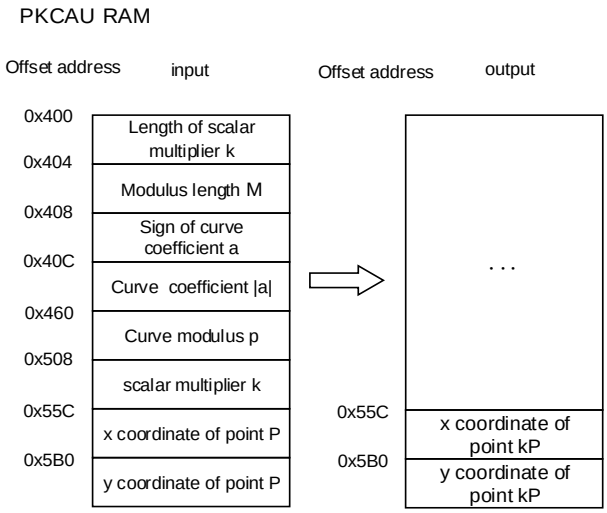
ECC scalar multiplication

ECC scalar multiplication operation is ak×P(x_P,y_P), where P is a point on the elliptic curve in the prime domain Fp. The operation result is also on the elliptic curve, or a point at infinity.

Normal mode

The ECC scalar multiplication operation in normal mode is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "100000". The operation declaration is shown in ***Figure 22-20. ECC scalar multiplication of normal mode***.

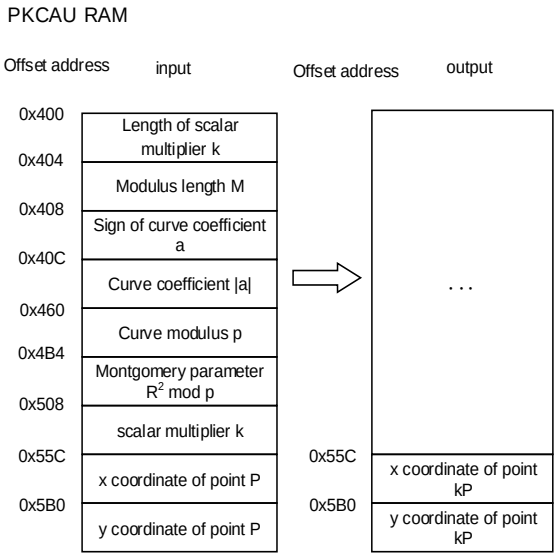
Figure 22-20. ECC scalar multiplication of normal mode



Fast mode

The ECC scalar multiplication operation in fast mode is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "100010". The operation declaration is shown in ***Figure 22-21. ECC scalar multiplication of fast mode***.

Figure 22-21. ECC scalar multiplication of fast mode



The range of parameters used by point on elliptic curve Fp check operation is shown in ***Table 22-6. Range of parameters used by ECC scalar multiplication***.

Table 22-6. Range of parameters used by ECC scalar multiplication

Parameters		Range
input	Length of scalar multiplier k (LEN)	0<LEN≤640
	Modulus length M	0<M≤640
	Sign of curve coefficient a	0x0: positive 0x1: negative
	Curve coefficient |a|	Absolute value |a|<p
	Curve modulus p	Odd prime $0<p\leq2^{M}$
	scalar multiplier k	$0\leq k<2^{LEN}$ (k < n, and n is the curve prime order)
	x coordinate of point P	$x_P<p$
	y coordinate of point P	$y_P<p$
output	x coordinate of point kP	x<p
	y coordinate of point kP	y<p

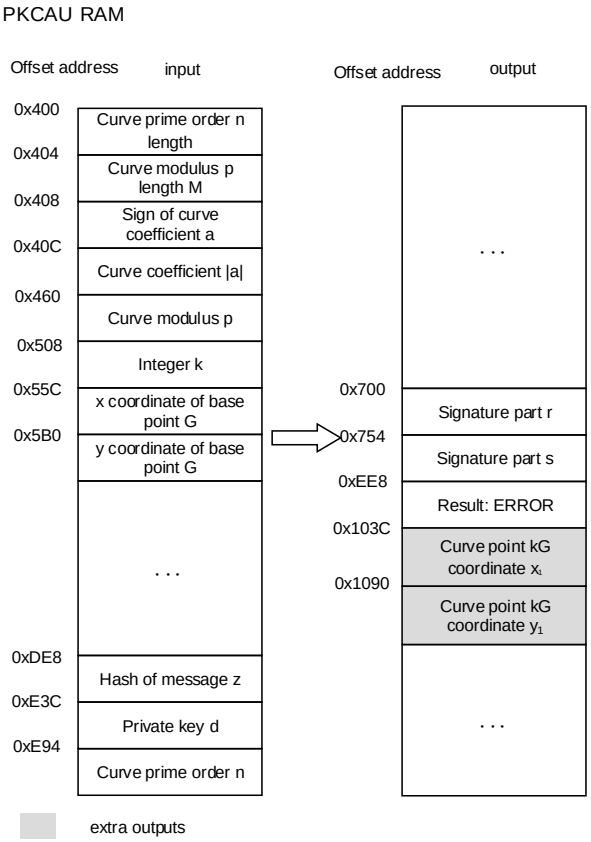
If k=0, the output is a point at infinity. When k is a multiple of curve prime order n, the output will also be a point at infinity. In this module, output is (0, 0) if the result is a point at infinity.

If k<0, the absolute value of k replaces k as the scalar multiplier for ECC scalar multiplication. After the computation is completed, -P = (x, -y) can be used to compute the final result of y.

ECDSA sign

The ECDSA sign operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "100100". The operation declaration is shown in ***Figure 22-22. ECDSA sign***.

Figure 22-22. ECDSA sign



The range of parameters used ECDSA sign operation is shown in ***Table 22-7. Range of parameters used by ECDSA sign***.

Table 22-7. Range of parameters used by ECDSA sign

Parameters		Range
input	Curve prime order n length (LEN)	0<LEN≤640
	Curve modulus p length (M)	0<M≤640
	Sign of curve coefficient a	0x0: positive 0x1: negative
	Curve coefficient |a|	Absolute value |a|<p
	Curve modulus p	Odd prime 0<p<2^M
	Integer k	0≤k<2^{LEN}
	x coordinate of base point G	x<p
	y coordinate of base point G	y<p
	Hash of message z	z<2^{LEN}
	Private key d	positive integer d < n
	Curve prime order n	prime n<2^{LEN}

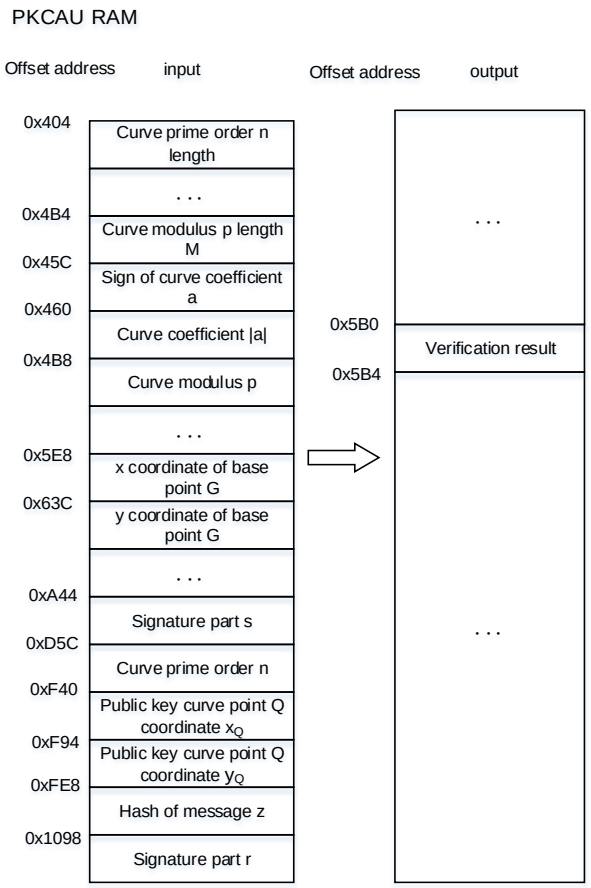
Parameters		Range
output	Signature part r	0<r<n
	Signature part s	0<s<n
	Signature result: ERROR	0x0: no error 0x1: Signature part r is 0 0x2: Signature part s is 0
	Curve point kG coordinate x_1	$0 \le x_1 < n$
	Curve point kG coordinate y_1	$0 \le y_1 < n$

If the error output is not 0, the content in PKCAU RAM will be cleared by hardware to avoid leaking private key related information.

ECDSA verification

The ECDSA verification operation is selected by configuring MODSEL[5:0] in PKCAU_CTL register as "100110". The operation declaration is shown in ***Figure 22-23. ECDSA verification***.

Figure 22-23. ECDSA verification



The range of parameters used ECDSA verification operation is shown in ***Table 22-8. Range***

of parameters used by ECDSA verification.

Table 22-8. Range of parameters used by ECDSA verification

Parameters		Range
input	Curve prime order n length (LEN)	0<LEN≤640
	Curve modulus p length (M)	0<M≤640
	Sign of curve coefficient a	0x0: positive 0x1: negative
	Curve coefficient |a|	Absolute value |a|<p
	Curve modulus p	Odd prime 0<p<2^M
	x coordinate of base point G	x<p
	y coordinate of base point G	y<p
	Public key curve point Q coordinate x_Q	x_Q<p
	Public key curve point Q coordinate y_Q	y_Q<p
	Signature part r	0<r<n
	Signature part s	0<s<n
	Hash of message z	z<2^{LEN}
	Curve prime order n	prime n<2^{LEN}
output	verification result	0x0: verification valid Not 0x0: verification invalid

22.3.6. PKCAU operation process

The PKCAU can be enabled by setting the PKCAUEN bit in PKCAU_CTL register. When the PKCAU is performing a calculation, the PKCAUEN should not be cleared, or else the ongoing operation is terminated and the content in PKCAU RAM will not be guaranteed.

When PKCAUEN is 0, the application can still access PKCAU RAM through the AHB interface.

Operation process in normal mode

The flows below applies to all operations listed in MODSEL[5:0] bits in the PKCAU_CTL register.

1. After system reset, the PKCAU RAM is to be erased from the beginning to the end. During this period, the BUSY bit in PKCAU_STAT register is set. All operation to PKCAU RAM should not carry out until the BUSY bit is 0;

2. Load the initial data into PKCAU RAM at offset address 0x400;
3. Specify the operation to be performed in MODSEL[5:0] bits in PKCAU_CTL register, then set START bit in PKCAU_CTL register;
4. Wait for the ENDF bit set in PKCAU_STAT register;
5. Read calculation result from the PKCAU RAM, then clear the ENDF bit by setting the ENDFC bit in PKCAU_STATC register.

Operation process in fast mode

The fast mode computes the Montgomery parameter only once when calculating many operations with the same modulus. When the operation is performed, the pre-calculated Montgomery parameter is loaded to perform the calculation.

The flow of operation in fast mode is as follows:

1. Load the initial data into PKCAU RAM at offset address 0x400;
2. Select the Montgomery parameter calculation mode by configuring the MODSEL[5:0] as “000001”, then set START bit in PKCAU_CTL register;
3. Wait for the ENDF bit set in PKCAU_STAT register;
4. Read the Montgomery parameter from the PKCAU RAM, then clear the ENDF bit by setting the ENDFC bit in PKCAU_STATC register;
5. Load the initial data and Montgomery parameter into PKCAU RAM;
6. Specify the operation to be performed in MODSEL[5:0] bits in PKCAU_CTL register, then set START bit in PKCAU_CTL register;
7. Wait for the ENDF bit set in PKCAU_STAT register;
8. Read calculation result from the PKCAU RAM, then clear the ENDF bit by setting the ENDFC bit in PKCAU_STATC register.

22.3.7. Processing times

The following tables summarize the PKCAU computation times, expressed in clock cycles.

Table 22-9. Modular exponentiation computation times

Exponent length (in bits)	Mode	Operand length (in bits)		
		1024	2048	3072
1024	Normal	6780000	-	-
	Fast	6701000	-	-
	CRT	1853000	-	-
2048	Normal	-	52196000	-
	Fast	-	51910000	-

Exponent length (in bits)	Mode	Operand length (in bits)		
		1024	2048	3072
	CRT	-	13651000	-
3072	Normal	-	-	182783000
	Fast	-	-	181953000
	CRT	-	-	44905000

Table 22-10. ECC scalar multiplication computation times

Mode	Modulus length (in bits)					
	160	192	256	320	384	512
Normal	626000	951000	1997000	3617000	5762000	13134000
Fast	623000	946000	1990000	3607000	5749000	13111000

Table 22-11. ECDSA signature average computation times

Modulus length (in bits)					
160	192	256	320	384	512
634000	966000	2029000	3648000	5833000	13177000

Table 22-12. ECDSA verification average computation times

Modulus length (in bits)					
160	192	256	320	384	512
1261000	1901000	3997000	7225000	11477000	26287000

Table 22-13. Montgomery parameters average computation times

Modulus length (in bits)								
160	192	256	320	384	512	1024	2048	3072
3873	4658	7109	10330	14526	22301	79116	284359	626909

22.3.8. Status, errors and interrupts

There are several status and error flags in PKCAU, and interrupt may be asserted from these flags by setting some register bits.

- Access address error (ADDRERR)

When the accessed address exceeds the expected range of PKCAU RAM, the address error flag ADDRERR bit in PKCAU_STAT register will be set. If the ADDRERRIE bit is set in PKCAU_CTL register, an interrupt will be generated. The ADDRERR bit can be cleared by setting the ADDRERRC bit in PKCAU_STATC register.

- RAM error (RAMERR)

The PKCAU core is using the PKCAU RAM while an AHB access to the PKCAU RAM occurs, the RAM error flag RAMERR bit in PKCAU_STAT register will be set. If it is an AHB read, it returns 0, an AHB write will be ignored. If the RAMERRIE bit is set in PKCAU_CTL register, an interrupt will be generated. The RAMERR bit can be cleared by setting the RAMERRC bit in PKCAU_STATC register.

- End of operation flag (ENDF)

When the operation specified in MODSEL[5:0] bits in the PKCAU_CTL register is completed, the ENDF bit will be set. If the ENDIE bit in PKCAU_CTL register is set, an interrupt will be generated. The ENDF bit can be cleared by setting the ENDFC bit in PKCAU_STATC register. The ENDF bit can also be cleared automatically if another operation is carried out by set the START bit.

The PKCAU interrupt events and flags are listed in ***Table 22-14. PKCAU interrupt requests***.

Table 22-14. PKCAU interrupt requests

Interrupt event	Event flag	Flag clear	Enable control bit
Access address error	ADDRERR	ADDRERRC	ADDRERRIE
RAM error	RAMERR	RAMERRC	RAMERRIE
Operation end flag	ENDF	ENDFC	ENDIE

22.4. Register definition

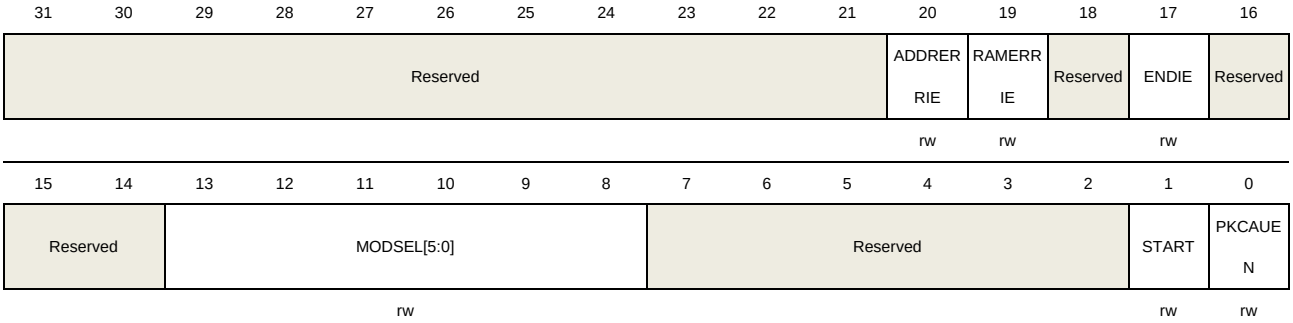
PKCAU base address: 0x4C06 1000

22.4.1. Control register (PKCAU_CTL)

Address offset: 0x00

Reset value: 0x0000 0000

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



Bits	Fields	Descriptions
31:21	Reserved	Must be kept at reset value.
20	ADDRERRIE	Address error interrupt enable 0: Address error interrupt disable 1: Address error interrupt enable
19	RAMERRIE	RAM error interrupt enable 0: RAM error interrupt disable 1: RAM error interrupt enable
18	Reserved	Must be kept at reset value.
17	ENDIE	End of operation interrupt enable 0: End of operation interrupt disable 1: End of operation interrupt enable
16:14	Reserved	Must be kept at reset value.
13:8	MODSEL[5:0]	PKCAU operation mode selection 000000: Montgomery parameter computation then modular exponentiation 000001: Montgomery parameter computation only 000010: Modular exponentiation only (Montgomery parameter must be loaded first) 000111: RSA CRT exponentiation 001000: Modular inversion 001001: Arithmetic addition

		001010: Arithmetic subtraction 001011: Arithmetic multiplication 001100: Arithmetic comparison 001101: Modular reduction 001110: Modular addition 001111: Modular subtraction 010000: Montgomery multiplication 100000: Montgomery parameter computation then ECC scalar multiplication 100010: ECC scalar multiplication only (Montgomery parameter must be loaded first) 100100: ECDSA sign 100110: ECDSA verification 101000: Point on elliptic curve Fp check Other values are reserved.
7:2	Reserved	Must be kept at reset value.
1	START	PKCAU starts operation This bit is set by software to start the PKCAU operation which is specified in MODSEL[5:0] in PKCAU_CTL register. When the BUSY bit in PKCAU_STAT register is 1, writing 1 to this bit will be ignored.
0	PKCAUEN	PKCAU enable 0: PKCAU disable 1: PKCAU enable

22.4.2. Status register (PKCAU_STAT)

Address offset: 0x04
Reset value: 0x0000 0000

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											ADDRERR	RAMERR	Reserved	ENDF	BUSY
											r	r		r	r

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

Bits	Fields	Descriptions
31:21	Reserved	Must be kept at reset value.
20	ADDRERR	Address error.

		0: No address error. 1: The accessed address exceeds the expected range of PKCAU RAM, an address error occurs.
19	RAMERR	PKCAU RAM error 0: No PKCAU RAM error. 1: When the PKCAU core is using the RAM, AHB accesses the PKCAU RAM, a PKCAU RAM error occurs.
18	Reserved	Must be kept at reset value.
17	ENDF	End of PKCAU operation When the operation executed completely, this bit is set by hardware.
16	BUSY	Busy flag When the START bit in PKCAU_CTL register is set, this bit is set by hardware. When the PKCAU operation is completed, this bit is cleared by hardware.
15:0	Reserved	Must be kept at reset value.

22.4.3. Status clear register (PKCAU_STATC)

Address offset: 0x08

Reset value: 0x0000 0000

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

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

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

Bits	Fields	Descriptions
31:21	Reserved	Must be kept at reset value.
20	ADDRERRC	Address error flag clear. Software can clear the ADDRERR bit in PKCAU_STAT by writing 1 to this bit.
19	RAMERRC	PKCAU RAM error flag clear. Software can clear the RAMERR bit in PKCAU_STAT by writing 1 to this bit.
18	Reserved	Must be kept at reset value.
17	ENDFC	End of PKCAU operation flag clear.

		Software can clear the ENDF bit in PKCAU_STAT by writing 1 to this bit.
16:0	Reserved	Must be kept at reset value.

23. Infrared ray port (IFRP)

23.1. 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. The IFRP_OUT pin can be configured by GPIO alternate function selected register.

23.2. Characteristics

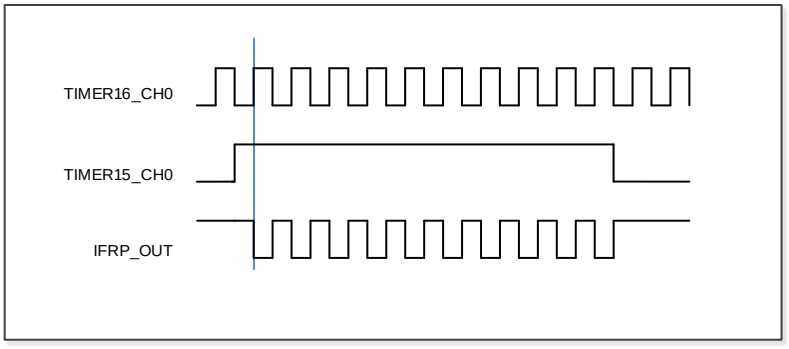
- The IFRP output signal is decided by TIMER15_CH0 and TIMER16_CH0
- To get correct infrared ray signal, TIMER15 should generate low frequence modulation envelope signal, and TIMER16 should generate high frequence carrier signal

23.3. Function overview

IFRP is a module which is able to integrate the output of TIMER15 and TIMER16 to generate an infrared ray signal.

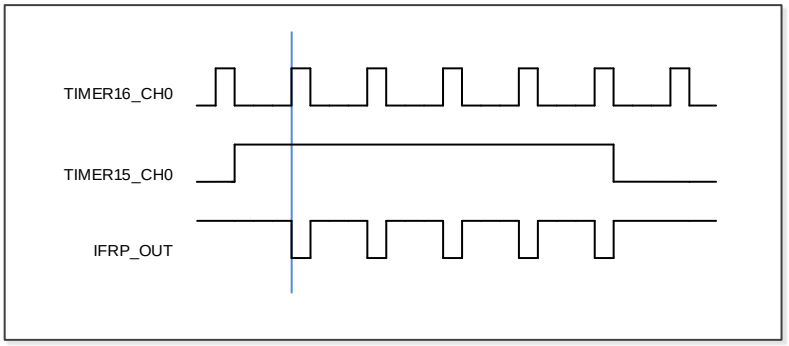
1. The TIMER15's CH0 is programed to generate the low frequence PWM signal which is the modulation envelope signal. The TIMER16's CH0 is programed to generate the high frquency PWM signal which is the carrier signal. And the channel need to be enabled before generating these signals.
2. Program the GPIO alternate function selected register and enable the IFRP_OUT pin.

Figure 23-1. IFRP output timechart 1



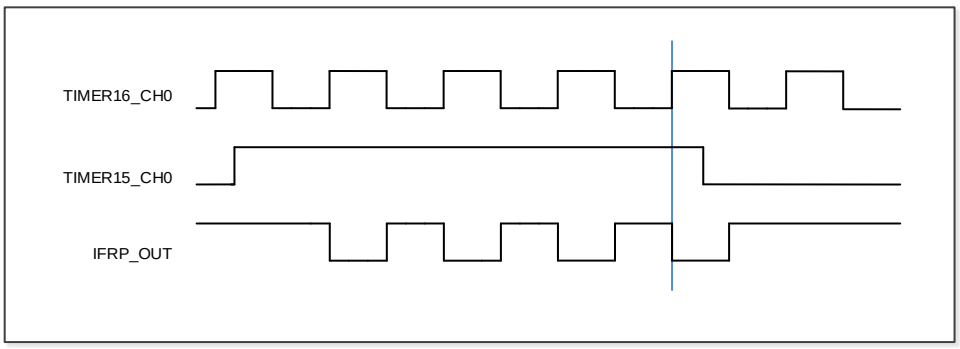
Note: IFRP_OUT has one APB clock delay from TIMER16_CH0.

Figure 23-2. 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 23-3. IFRP output timechart 3



Note: IFRP_OUT will keep the integrity of TIMER16_CH0, even if evelope signal (TIMER15_CH0) is no active.

24. Wireless

24.1. Overview

The GD32VW55x is a highly integrated Wireless System-on-Chip (SoC) that includes an RISC-V processor with a single stream Wi-Fi6 MAC and PHY, BLE5 link layer and modem, RF transceiver, a power amplifier (PA), and a receive low-noise amplifier (LNA). It is an optimized SoC designed for a broad array of smart devices for Internet of Things (IoT) applications.

24.2. Wi-Fi

24.2.1. Characteristics

Standards Supported

- 802.11b / g / n /ax compatible.
- 802.11e QoS Enhancement (WMM).
- 802.11i (WPA, WPA2, WPA3). Open, shared key, and pair-wise key authentication services.
- WiFi WPS.
- WiFi Direct.
- Integrated TCP / IP protocol.

Wi-Fi MAC

- Target Wake up Time (TWT) operation.
- Two NAV.
- Multiple BSSID operation.
- OFDMA-based random access.
- Spatial reuse.
- Transmission and reception of aggregated MPDUs (A-MPDU) for high throughput.
- Support for immediate ACK and Block-ACK policies.
- Support for power management schemes, including WMM power-save, power-save multi-poll (PSMP), and multiphase PSMP operation.
- Interframe space timing support, including RIFS.
- Support for RTS / CTS and CTS-to-self frame sequences for protecting frame exchanges
- Back-off counters in hardware for supporting multiple priorities as specified in the WMM specification.
- Timing synchronization function (TSF), network allocation vector (NAV) maintenance, and target beacon transmission time (TBTT) generation in hardware.

- Hardware engine for AES-CCMP, legacy WPA TKIP, legacy WEP ciphers, and support for key management.
- Programmable independent basic service set (IBSS) or infrastructure basic service set or Access Point functionality.

Wi-Fi PHY

- Single antenna 1x1 stream in 20MHz channels.
- 20M bandwidth.
- MU-OFDMA in UL and DL as a non-AP STA.
- DL MU-MIMO as a non-AP STA.
- Beamforming as a beamformee.
- Rx STBC scheme (1 spatial stream and 2 space-time streams).
- Mid-amble.
- DCM.
- All guard interval (0.8 / 1.6 / 3.2us).
- Support of 802.11ax MCS up to MCS9 with Max phy rate as 114.7Mbps.
- Per packet TX power control.
- Advanced channel estimation / equalization, automatic gain control, CCA, carrier/symbol recovery, and frame detection.
- Digital calibration algorithms to handle CMOS RF chip process, voltage, and temperature (PVT) variations.
- Per-packet channel quality and signal-strength measurements.
- Compliance with FCC and other worldwide regulatory requirements.

24.3. BLE

24.3.1. Characteristics

Standards Supported

- BLE5.2.

BLE Linker Layer

- Support multiple simulateneous hardware connection
- Advertising Extension
- High duty cycle non-connectable advertising
- Channel selection algorithm #2
- Support multiple simultaneous BLE connections

BLE Modem

- High speed 2M PHY.

- Long range coded PHY.
- Data rate: 125, 500, 1000 and 2000kbps.

24.4. Radio

Radio is shared between Wi-Fi and BLE.

- Fractional-N for multiple reference clock support.
- Integrated PA with power control.
- Optimized Tx gain distribution for linearity and noise performance.
- Direct conversion architecture.
- On-chip gain selectable LNA with optimized noise figure.
- High dynamic range AGC.

25. Appendix

25.1. List of abbreviations used in register

Table 25-1. List of abbreviations used in register

abbreviations for registers	Descriptions
read/write (rw)	Software can read and write to this bit.
read-only (r)	Software can only read this bit.
write-only (w)	Software can only write to this bit. Reading this bit returns the reset value.
read/clear write 1 (rc_w1)	Software can read as well as clear this bit by writing 1. Writing 0 has no effect on the bit value.
read/clear write 0 (rc_w0)	Software can read as well as clear this bit by writing 0. Writing 1 has no effect on the bit value.
toggle (t)	Software can toggle this bit by writing 1. Writing 0 has no effect.
read-only/set by write 1 (rt_w1)	Software can only read to this bit. Writing 1 trigges the event but has no effect on the bit value.
read/set (rs)	Software can read as well as set this bit to 1. Writing 0 has no effect on the bit value.
read/clear by read (rc_r)	Software can read this bit. Reading this bit automatically clears it to '0'. Writing '0' has no effect on the bit value.
read/set by read (rs_r)	Software can read this bit, and set this bit by reading. Writing has no effect on the bit value.
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/clear write (rc_w)	Software can read as well as clear this bit by writing. Writing 0 or 1 has the same effect to this bit.
read-only/write trigger (rt_w)	Software can read this bit. Writing 0 or 1 triggers an event but has no effect on the bit value.

25.2. List of terms

Table 25-2. 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	ICP is the ability to program the Flash memory of a microcontroller using the

Glossary	Descriptions
programming)	JTAG 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.

25.3. Available peripherals

For availability of peripherals and their number across all MCU series types, refer to the corresponding device data datasheet.

26. Revision history

Table 26-1. Revision history

Revision No.	Description	Date
1.0	1. Initial Release.	Oct.13, 2023
1.1	1. Change descriptions of DDRE bit in USART_CTL2 register, refers to ***Control register 2 (USART_CTL2)***. 2. Modify ***Channel input remap register (TIMERx_IRMP)(x=2)*** register bits and the description. 3. Modify figure ***Figure 5-2. Clock tree*** and add notes. 4. Modify descriptions in ***Boot configuration***.	Dec.30, 2023
1.2	1. Modify descriptions in ***BLE Modem***. 2. Add descriptions of CH0IF bit in TIMERx_INTF register. 3. Modify the pause mode field of ***TIMER trigger selection register (SYSCFG_TIMERxCFG)(x = 0..2)*** register, in which the 00101 value descriptions to reserved. 4. Modify ***Analog to digital converter (ADC)*** module descriptions. 5. Modify ***Timer (TIMERx)*** module descriptions.	Mar.25, 2024
1.3	1. Modify ***Table 1-2. Memory map of GD32VW55x devices***, delete unaccessable address. 2. Modify SCPF bit description of ***Status register (RTC_STAT)***. 3. Modify Software procedure description in ***Single operation mode*** and ***Continuous operation mode*** chapter for ADC module. 4. Modify bit 17, 19, 20 description in ***Control register (PKCAU_CTL)*** for PKCAU module. 5. Modify sequence chart of Counter center-aligned counting mode in ***Advanced timer (TIMERx, x=0)*** and ***General level0 timer (TIMERx, x=1, 2)*** chapter for TIMER module. 6. Delete chapter ***Device electronic signature***. 7. Modify ***Table 12-2. ADC input pins definition***. 8. Delete characteristics of “ADC supply requirements: 1.62V to 3.6V, and typical power supply voltage is 3.3V” in chapter 12.2. 9. Modify ***Figure 4-1. Power supply overview*** and ***VDD / VDDA power domain*** contents.	Feb.13, 2025

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 under the intellectual property laws and treaties of the People’s Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. 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.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do release the Company as well as it’s suppliers and/or distributors from any claim, damage, or other liability arising from or related to all Unintended Uses of the Products. Customers shall indemnify and hold the Company as well as it’s 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 Products.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.

© 2025 GigaDevice – All rights reserved