User Manual

MM32F0010

32-Bit Microcontroller Based on ARM® Cortex® M0

Version: 1.00

MindMotion reserves the right to change the relevant information without prior notification.

Contents

1 Memory and Bus Architecture 1
1.1 System architecture 1
1.2 Memory organization 3
1.2.1 Introduction 3
1.2.2 Memory mapping and register addressing 3
1.3 Embedded SRAM 7
1.4 Flash memory overview 7
2 Embedded Flash Memory (FLASH) 8
2.1 Key Flash features 8
2.2 Flash memory function description 9
2.2.1 Flash memory organization 9
2.2.2 Read operation of the Flash memory 10
2.2.3 Flash write and erase operations 10
2.3 Memory protection 17
2.3.1 Read protection 17
2.3.2 Write protection of main memory 18
2.3.3 Option byte write protection 18
2.4. Flash interrupt 18
2.5 Option byte description 18
2.6. Flash register descriptions 20
2.6.1 Flash access control register (FLASH_ACR) 20
2.6.2 FPEC key register (FLASH_KEYR) 21
2.6.3 Flash OPTKEY register (FLASH_OPTKEYR) 21
2.6.4 Flash status register (FLASH_SR) 22
2.6.5 Flash control register (FLASH_CR) 23
2.6.6 Flash address register (FLASH_AR) 24
2.6.7 Option byte register (FLASH_OBR) 25
2.6.8 Write protection register (FLASH_WRPR) 26
3 Cyclic Redundancy Check (CRC) Calculation Unit 27
3.1 CRC introduction 27
3.2 CRC main features 27
3.3 CRC function introduction 28

3.4 CRC registers 28
3.4.1 CRC data register (CRC_DR) 28
3.4.2 CRC independent data register (CRC_IDR) 29
3.4.3 CRC control register (CRC_CTRL) 29
4 Power Control (PWR) 30
4.1 Power supplies 30
4.1.1 Independent A/D converter supply and reference voltage 30
4.1.2 Voltage regulator 31
4.2 Power supply supervisor 31
4.2.1 Power-on reset (POR)/power-down reset (PDR) 31
4.2.2 Programmable voltage detector (PVD) 31
4.3 Low-power modes 32
4.3.1 Slowing down system clocks 33
4.3.2 Peripheral clock gating 33
4.3.3 Sleep mode 33
4.3.4 Stop mode 34
4.3.5 Standby mode 35
4.4 Power control registers 36
4.4.1 Power control register (PWR_CR) 36
4.4.2 Power control/status register (PWR_CSR) 38
5 Reset and Clock Control (RCC) 40
5.1 Reset 40
5.1.1 System reset 40
5.1.2 Power reset 40
5.2 Clocks 41
5.2.1 HSE clock 43
5.2.2 HSI clock 44
5.2.3 LSI clock 44
5.2.4 System clock (SYSCLK) selection 44
5.2.5 Clock security system (CSS) 45
5.2.6 Watchdog clock 45
5.2.7 Clock-out capability 45
5.3 RCC register file and memory mapping description 45
5.3.1 Clock control register (RCC_CR) 46

5.3.2 Clock configuration register (RCC_CFGR) 47
5.3.3 Clock interrupt register (RCC_CIR) 49
5.3.4 APB1 peripheral reset register (RCC_APB1RSTR) 52
5.3.5 AHB peripheral clock enable register (RCC_AHBENR) 53
5.3.6 APB1 peripheral clock enable register (RCC_APB1ENR) 55
5.3.7 Control status register (RCC_CSR) 57
5.3.8 AHB peripheral reset register (RCC_AHBRSTR) 59
5.3.9 System configuration register (RCC_SYSCFG) 59
6 General-Purpose I/Os (GPIOs) 61
6.1 GPIO functional description 61
6.1.1 General-purpose I/O (GPIO) 63
6.1.2 Atomic bit set or reset 63
6.1.3 External interrupt/wakeup lines 63
6.1.4 Alternate functions (AF) 63
6.1.5 Software remapping of I/O alternate functions 64
6.1.6 GPIO locking mechanism 64
6.1.7 Input configuration 64
6.1.8 Output configuration 64
6.1.9 Alternate function configuration 65
6.1.10 Analog input configuration 66
6.1.11 GPIO configurations for device peripherals 67
6.2. Alternate function I/O and debug configuration 68
6.2.1 OSC_IN/OSC_OUT 69
6.2.2 SWD alternate function remapping 69
6.3 GPIO register description 69
6.3.1 Port configuration register low (GPIOx_CRL) (x = A..B) 69
6.3.2 Port configuration register high (GPIOx_CRH) (x = A..B) 70
6.3.3 Port input data register (GPIOx_IDR) (x = A..B) 71
6.3.4 Port output data register (GPIOx_ODR) (x = A..B) 72
6.3.5 Port bit set/reset register (GPIOx_BSRR) (x = A..B) 72
6.3.6 Port bit reset register (GPIOx_BRR) (x = A..B) 72
6.3.7 Port configuration lock register (GPIOx_LCKR) (x = A..B) 73
6.3.8 Port output open drain control register (GPIOx_DCR) (x = A..B) 74
6.3.9 Port alternate function register low (GPIOx_AFRL) (x = A..B) 74

6.3.10 Port alternate function register high (GPIOx_AFRH) (x = A..B) 75
7 Interrupts and Events (EXTI) 77
7.1 Nested vectored interrupt controller 77
7.1.1 SysTick calibration value register 77
7.1.2 Interrupt and exception vectors 77
7.2 External interrupt/event controller (EXTI) 79
7.2.1 Main features 79
7.2.2 Block diagram 80
7.2.3 Wakeup event management 80
7.2.4 Functional description 80
7.2.5 External interrupt/event line mapping 81
7.3 EXTI register description 82
7.3.1 Interrupt mask register (EXTI_IMR) 83
7.3.2 Event mask register (EXTI_EMR) 83
7.3.3 Rising trigger selection register (EXTI_RTSR) 84
7.3.4 Falling trigger selection register (EXTI_FTSR) 84
7.3.5 Software interrupt event register (EXTI_SWIER) 85
7.3.6 Pending register (EXTI_PR) 85
8 Analog-to-digital converter (ADC) 87
8.1 ADC introduction 87
8.2 ADC main features 87
8.3 System block diagram 88
8.4 ADC functional description 88
8.4.1 ADC on-off control 89
8.4.2 Channel selection 90
8.5 Arbitrary channel operation mode 90
8.5.1 Single conversion mode 90
8.5.2 One-cycle scan mode 91
8.5.3 Continuous scan mode 91
8.6 Data alignment 93
8.6.1 Programmable resolution 93
8.6.2 Programmable sampling time 93
8.7 Conversion on external trigger 93
8.8 Internal reference voltage 94

8.9 Monitoring AD conversion result in the window comparator mode 94
8.10 ADC register description 94
8.10.1 A/D data register (ADC_ADDATA) 96
8.10.2 A/D configuration register (ADC_ADCFG) 96
8.10.3 A/D control register (ADC_ADCR) 98
8.10.4 A/D window comparison register (ADC_ADCMPR) 100
8.10.5 A/D status register (ADC_ADSTA) 101
8.10.6 A/D data register (ADC_ADDR0 ~ 8) 102
8.10.7 A/D arbitrary channel selection register (ADC_CHANY0) 103
8.10.8 A/D arbitrary channel selection register 1 (ADC_CHANY1) 104
8.10.9 A/D arbitrary channel configuration register (ADC_ANY_CFG) 105
8.10.10 A/D arbitrary channel control register (ADC_ANY_CR) 105
9 Advanced-Control Timer (TIM1) 107
9.1 TIM1 introduction 107
9.2 Main features 107
9.3 Functional description 108
9.3.1 Time-base unit 108
9.3.2 Counter modes 110
9.3.3 Repetition counter 118
9.3.4 Clock selection 120
9.3.5 Capture/compare channels 122
9.3.6 Input capture mode 124
9.3.7 PWM input mode 125
9.3.8 Forced output mode 126
9.3.9 Output compare mode 127
9.3.10 PWM mode 128
9.3.11 Complementary outputs and dead-time insertion 131
9.3.12 Using the break function 133
9.3.13 6-step PWM generation 135
9.3.14 One-pulse mode 136
9.3.15 Encoder interface mode 138
9.3.16 Timer input XOR function 140
9.3.17 Interfacing with Hall sensors 140
9.3.18 TIMx and external trigger synchronization 142

9.3.19 Timer synchronization 144
9.3.20 Debug mode 144
9.4 Register description 144
9.4.1 Control register 1 (TIMx_CR1) 145
9.4.2 Control register 2 (TIMx_CR2) 146
9.4.3 Slave mode control register (TIMx_SMCR) 149
9.4.4 Interrupt enable register (TIMX_DIER) 150
9.4.5 Status register (TIMx_SR) 152
9.4.6 Event generation register (TIMx_EGR) 154
9.4.7 Capture/compare mode register 1 (TIMx_CCMR1) 156
9.4.8 Capture/compare mode register 2 (TIMx_CCMR2) 160
9.4.9 Capture/compare enable register (TIMx_CCER) 162
9.4.10 Counter (TIMx_CNT) 165
9.4.11 Prescaler (TIMx_PSC) 165
9.4.12 Auto-reload register (TIMx_ARR) 165
9.4.13 Repetition counter register (TIMx_RCR) 166
9.4.14 Capture/compare register 1 (TIMx_CCR1) 166
9.4.15 Capture/compare register 2 (TIMx_CCR2) 167
9.4.16 Capture/compare register 3 (TIMx_CCR3) 167
9.4.17 Capture/compare register 4 (TIMx_CCR4) 168
9.4.18 Break and dead-time register (TIMx_BDTR) 169
9.4.19 Capture/compare mode register 3 (TIMx_CCMR3) 171
9.4.20 Capture/compare register 5 (TIMx_CCR5) 172
9.4.21 PWM phase-shift update request enable register (TIMx_PDER) 172
9.4.22 PWM phase-shift downcounting capture/compare register (CCRxFALL) 173
10 16-Bit General-Purpose Timer (TIM3) 175
10.1. TIMx introduction 175
10.2. TIMx main features 175
10.3. TIMx functional description 176
10.3.1 Time-base unit 176
10.3.2 Counter modes 178
10.3.3 Clock selection 186
10.3.4 Capture/compare channel 188
10.3.5 Input capture mode 190

10.3.6 PWM input mode 191
10.3.7 Forced output mode 192
10.3.8 Output compare mode 192
10.3.9 PWM mode 193
10.3.10 One-pulse mode 196
10.3.11 Encoder Interface mode 197
10.3.12 Timer input XOR function 200
10.3.13 Timers and external trigger synchronization 200
10.3.14 Timer synchronization 202
10.3.15 Debug mode 206
10.4 TIMx register description 206
10.4.1 Control register 1 (TIMx_CR1) 207
10.4.2 Control register 2 (TIMx_CR2) 209
10.4.3 Slave mode control register (TIMx_SMCR) 210
10.4.4 Interrupt enable register (TIMx_DIER) 212
10.4.5 Status register (TIMx_SR) 212
10.4.6 Event generation register (TIMx_EGR) 214
10.4.7 Capture/compare mode register 1 (TIMx_CCMR1) 215
10.4.8 Capture/compare mode register 2 (TIMx_CCMR2) 220
10.4.9 Capture/compare enable register (TIMx_CCER) 221
10.4.10 Counter (TIMx_CNT) 223
10.4.11 Prescaler (TIMx_PSC) 223
10.4.12 Auto-reload register (TIMx_ARR) 224
10.4.13 Capture/compare register 1 (TIMx_CCR1) 224
10.4.14 Capture/compare register 2 (TIMx_CCR2) 225
10.4.15 Capture/compare register 3 (TIMx_CCR3) 225
10.4.16 Capture/compare register 4 (TIMx_CCR4) 226
11 Basic Timer (TIM14) 227
11.1 TIM14 introduction 227
11.2 TIM14 main features 227
11.3 TIM14 functional description 228
11.3.1 Time-base unit 228
11.3.2 Counting mode 230
11.3.3 Clock source 232

11.3.4 Capture/compare channel 233
11.3.5 Input capture mode 235
11.3.6 Forced output mode 235
11.3.7 Output compare mode 236
11.3.8 PWM mode 237
11.3.9 Debug mode 238
11.4 TIM14 register description 238
11.4.1 Control register 1 (TIM14_CR1) 239
11.4.2 Interrupt enable register (TIM14_DIER) 240
11.4.3 Status register (TIM14_SR) 240
11.4.4 Event generation register (TIM14_EGR) 241
11.4.5 Capture/compare mode register 1 (TIM14_CCMR1) 242
11.4.6 Capture/compare enable register (TIM14_CCER) 246
11.4.7 Counter (TIM14_CNT) 248
11.4.8 Prescaler (TIM14_PSC) 248
11.4.9 Auto-reload register (TIM14_ARR) 248
11.4.10 Capture/compare register 1 (TIM14_CCR1) 248
11.4.11 Break and dead-time register (TIM14_BDTR) 249
12 Independent Watchdog (IWDG) 250
12.1 Independent watchdog (IWDG) introduction 250
12.2 IWDG main features 250
12.3 IWDG functional description 250
12.3.1 Hardware watchdog 251
12.3.2 Register access protection 251
12.3.3 Debug mode 252
12.4 IWDG register description 252
12.4.1 Key register (IWDG_KR) 252
12.4.2 Prescaler register (IWDG_PR) 253
12.4.3 Reload register (IWDG_RLR) 254
12.4.4 Status register (IWDG_SR) 254
12.4.5 IWDG control register (IWDG_CR) 255
12.4.6 IWDG interrupt generate register (IWDG_IGEN) 256
12.4.7 Counter register (IWDG_CNT) 256
13 Window Watchdog (WWDG) 258

13.1 WWDG introduction 258
13.2 WWDG main features 258
13.3 WWDG functional description 258
13.4 How to program the watchdog timeout 260
13.5 Debug mode 261
13.6 WWDG register description 261
13.6.1 Control register (WWDG_CR) 261
13.6.2 Configuration register (WWDG_CFGR) 262
13.6.3 Status register (WWDG_SR) 262
14 Serial Peripheral Interface (SPI) 264
14.1 SPI introduction 264
14.2 Main features 264
14.3 SPI functional description 265
14.3.1 General description 265
14.3.2 SPI in slave mode 269
14.3.3 SPI in master mode 269
14.3.4 Status flags 270
14.3.5 Baud rate configuration 271
14.4 Register file and memory mapping description 271
14.4.1 Transmit data register (SPI_TXREG) 272
14.4.2 Receive data register (SPI_RXREG) 272
14.4.3 Current status register (SPI_CSTAT) 273
14.4.4 Interrupt status register (SPI_INTSTAT) 274
14.4.5 Interrup enable register (SPI_INTEN) 275
14.4.6 Interrupt clear register (SPI_INTCLR) 276
14.4.7 Global control register (SPI_GCTL) 278
14.4.8 Common control register (SPI_CCTL) 279
14.4.9 Baud rate generator (SPI_SPBRG) 281
14.4.10 Receive data number register (SPI_RXDNR) 281
14.4.11 Slave chip select register (SPI_NSSR) 282
14.4.12 Data control register (SPI_EXTCTL) 282
15 Inter-Integrated Circuit (I2C) Interface 284
15.1 I2C introduction 284
15.2 I2C main features 284

15.3 I2C protocol 284
15.3.1 Start and Stop conditions 284
15.3.2 Slave addressing protocol 285
15.3.3 Transmitting and receiving protocol 286
15.3.4 Tx buffer management and Start, Stop, and Restart generation 288
15.3.5 Multiple master arbitration 290
15.3.6 Clock synchronization 290
15.3.7 SCL configuration 291
15.4 I2C operation mode 292
15.4.1 Slave mode 293
15.4.2 Master mode 295
15.4.3 I2C abort transfer 296
15.5 I2C interrupts 297
15.6 I2C register description 297
15.6.1 I2C control register (I2C_CR) 299
15.6.2 I2C target address register (I2C_TAR) 302
15.6.3 I2C Slave address register (I2C_SAR) 302
15.6.4 I2C Data command register (I2C_DR) 303
15.6.5 Standard mode I2C clock SCL high count register (I2C_SSHR) 303
15.6.6 Standard mode I2C clock SCL low count register (I2C_SSLR) 303
15.6.7 Fast mode I2C clock SCL high count register (I2C_FSHR) 304
15.6.8 Fast mode I2C clock SCL low count register (I2C_FSLR) 304
15.6.9 I2C interrupt status register (I2C_ISR) 304
15.6.10 I2C interrupt mask register (I2C_IMR) 305
15.6.11 I2C RAW interrupt status register (I2C_RAWISR) 305
15.6.12 I2C receive threshold register (I2C_RXTLR) 307
15.6.13 I2C transmit threshold register (I2C_TXTLR) 308
15.6.14 I2C combined and independent interrupt clear register (I2C_ICR) 308
15.6.15 I2C RX_UNDER interrupt clear register (I2C_RX_UNDER) 308
15.6.16 I2C RX_OVER interrupt clear register (I2C_RX_OVER) 309
15.6.17 I2C TX_OVER interrupt clear register (I2C_TX_OVER) 309
15.6.18 I2C RD_REQ interrupt clear register (I2C_RD_REQ) 309
15.6.19 I2C TX_ABRT interrupt clear register (I2C_TX_ABRT) 310
15.6.20 I2C RX_DONE interrupt clear register (I2C_RX_DONE) 310

15.6.21 I2C ACTIVITY interrupt clear register (I2C_ACTIV) 310
15.6.22 I2C STOP_DET interrupt clear register (I2C_STOP) 311
15.6.23 I2C START_DET interrupt clear register (I2C_START) 311
15.6.24 I2C GEN_CALL interrupt clear register (I2C_GC) 311
15.6.25 I2C enable register (I2C_ENR) 311
15.6.26 I2C status register (I2C_SR) 312
15.6.27 I2C transmit FIFO level register (I2C_TXFLR) 313
15.6.28 I2C receive FIFO level register (I2C_RXFLR) 313
15.6.29 I2C SDA hold time register (I2C_HOLD) 314
15.6.30 I2C SDA setup time register (I2C_SETUP) 314
15.6.31 I2C general call ACK register (I2C_GCR) 314
15.6.32 I2C slave address mask register (I2C_SLVMASK) 315
15.6.33 I2C slave receive address register (I2C_SLVRCVADDR) 315
16 Universal Asynchronous Receiver Transmitter (UART) 316
16.1 UART introduction 316
16.2 UART main features 316
16.3 UART functional description 317
16.3.1 UART character description 318
16.3.2 Transmitter 319
16.3.3 Receiver 321
16.3.4 9-bit data communication 322
16.3.5 Multiprocessor communication 322
16.3.6 Single-wire half-duplex communication 323
16.3.7 Smartcard 323
16.3.8 Fractional baud rate generator 325
16.3.9 Sampling 325
16.3.10 Parity control 326
16.3.11 Automatic baud rate detection 326
16.4 UART interrupt requests 327
16.5 UART register description 327
16.5.1 UART transmit data register (UART_TDR) 328
16.5.2 UART receive data register (UART_RDR) 329
16.5.3 UART current status register (UART_CSR) 329
16.5.4 UART interrupt status register (UART_ISR) 330

16.5.5 UART interrupt enable register (UART_IER) 331
16.5.6 UART interrupt clear register (UART_ICR) 333
16.5.7 UART global control register (UART_GCR) 334
16.5.8 UART common control register (UART_CCR) 335
16.5.9 UART baud rate register (UART_BRR) 337
16.5.10 UART fractional baud rate register (UART_FRA) 338
16.5.11 UART receive address register (UART_RXADDR) 338
16.5.12 UART receive mask register (UART_RXMASK) 339
16.5.13 UART SCR register (UART_SCR) 339
16.5.14 UART IDLE data length register (UART_IDLR) 340
16.5.15 UART ABRCR automatic baud rate control register (UART_ABRCR) 340
17 System Configuration Controller (SYSCFG) 342
17.1 SYSCFG register description 342
17.1.1 SYSCFG configuration register (SYSCFG_CFGR) 342
17.1.2 External interrupt configuration register 1 (SYSCFG_EXTICR1) 343
17.1.3 External interrupt configuration register 2 (SYSCFG_EXTICR2) 344
17.1.4 External interrupt configuration register 3 (SYSCFG_EXTICR3) 344
17.1.5 External interrupt configuration register 4 (SYSCFG_EXTICR4) 345
17.1.6 PAD configuration register (SYSCFG_PADHYS) 345
18 Device Electronic Signature (Device) 347
18.1 Memory size registers 347
18.1.1 Unique device ID register (96 bits) 347
18.2 UID register description 347
18.2.1 Unique ID (UID1) 348
18.2.2 Unique ID (UID2) 348
18.2.3 Unique ID (UID3) 348
18.2.4 Unique ID (UID4) 349
19 Debug Support (DBG) 350
19.1 Overview 350
19.2 Pinout and debug port pins 351
19.2.1 SWD debug port pins 351
19.2.2 Internal pull-up and pull-down on SWD pins 351
19.3 ID codes and locking mechanism 351
19.3.1 MCU device ID code 351

19.3.2 Cortex JEDEC-106 ID code 352
19.4 SW debug port 352
19.4.1 SW protocol introduction 352
19.4.2 SW protocol sequence 353
19.4.3 SW-DP state machine (Reset, idle states, ID code) 354
19.4.4 DP and AP read/write accesses 354
19.4.5 SW-DP registers 354
19.4.6 SW-AP registers 355
19.5 MCU debug component (MCUDBG) 355
19.5.1 Debug support for low-power modes 355
19.5.2 Debug MCU configuration register 356
19.6 DBG register description 356
19.6.1 DBG control register (DBG_CR) 356
20 Revision History 359

Illustration

Figure 1. System architecture 2
Figure 2. Programming procedures 12
Figure 3. Flash register page erase procedures 13
Figure 4. Flash register mass erase procedures 14
Figure 5. Option byte programming procedures 16
Figure 6. Option byte erase procedure 16
Figure 7. CRC calculation unit block diagram 27
Figure 8. Power supply block diagram 30
Figure 9. Power-on reset/power-down reset waveform 31
Figure 10. PVD thresholds 32
Figure 11. Reset circuit 41
Figure 12. Clock tree 42
Figure 13. Clock sources 43
Figure 14. Basic structure of a standard I/O port bit 62
Figure 15. Input floating/pull up/pull down configurations 64
Figure 16. Output configuration 65

Figure 17. Alternate function configuration 66
Figure 18. High impedance-analog input configuration 67
Figure 19. External interrupt/event controller block diagram 80
Figure 20. External interrupt GPIO mapping 82
Figure 21. ADC system block diagram 88
Figure 22. ADC block diagram 89
Figure 23. Channel conversion timing diagram in Single conversion mode 90
Figure 24. Channel conversion timing diagram in One-cycle conversion mode 91
Figure 25. Channel conversion timing diagram in Continuous scan mode 92
Figure 26. Timing diagram with dynamically updated configuration, in Continuous scan mode 93
Figure 27. Data alignment style 93
Figure 28. Advanced-control timer block diagram 108
Figure 29. Counter timing diagram with prescaler division change from 1 to 2 109
Figure 30.Counter timing diagram with prescaler division change from 1 to 4 110
Figure 31. Counter timing diagram, internal clock divided by 1 111
Figure 32. Counter timing diagram, internal clock divided by 2 111
Figure 33. Counter timing diagram, internal clock divided by 4 111
Figure 34. Counter timing diagram, internal clock divided by N 112
Figure 35. Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded) 112
Figure 36. Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded) 113
Figure 37. Counter timing diagram, internal clock divided by 1 113
Figure 38. Counter timing diagram, internal clock divided by 2 114
Figure 39. Counter timing diagram, internal clock divided by 4 114
Figure 40. Counter timing diagram, internal clock divided by N 114
Figure 41. Counter timing diagram, update event when repetition counter is not used 115
Figure 42. Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6 116
Figure 43. Counter timing diagram, internal clock divided by 2 116
Figure 44. Counter timing diagram, internal clock divided by 4, TIMx_ARR=0x03 117
Figure 45. Counter timing diagram, internal clock divided by N 117
Figure 46. Counter timing diagram, update event with ARPE=1 (counter underflow) 118
Figure 47. Counter timing diagram, update event with ARPE=1 (counter overflow) 118

Figure 48. Update rate examples depending on mode and TIMx_RCR register settings 120
Figure 49. Control circuit in normal mode, internal clock divided by 1 121
Figure 50. TI2 external clock connection example 121
Figure 51. Control circuit in external clock mode 1 122
Figure 52. Capture/compare channel (example: channel 1 input stage) 123
Figure 53. Capture/compare channel 1 main circuit 123
Figure 54. Capture/compare channel 1 main circuit 124
Figure 55. Output stage of capture/compare channel (channel 4) 124
Figure 56. PWM input mode timing 126
Figure 57. Output compare mode, toggle on OC1 128
Figure 58. Edge-aligned PWM waveforms (ARR = 8) 129
Figure 59. Center-aligned PWM waveforms (ARR = 8) 130
Figure 60. Phase shift diagram 131
Figure 61. Complementary output with dead-time insertion 132
Figure 62. Dead-time waveforms with delay greater than the negative pulse 132
Figure 63. Dead-time waveforms with delay greater than the positive pulse 133
Figure 64. Output behavior in response to a break 135
Figure 65. 6-step generation, COM example (OSSR=1) 136
Figure 66. Example of one pulse mode 137
Figure 67. Example of counter operation in encoder interface mode 139
Figure 68. Example of encoder interface mode with IC1FP1 polarity inverted 140
Figure 69. Example of Hall sensor interface 141
Figure 70. Control circuit in reset mode 142
Figure 71. Control circuit in gated mode 143
Figure 72. Control circuit in trigger mode 143
Figure 73. General-purpose timer block diagram 176
Figure 74. Counter timing diagram with prescaler division change from 1 to 2 177
Figure 75. Counter timing diagram with prescaler division change from 1 to 4 178
Figure 76. Counter timing diagram, internal clock divided by 1 179
Figure 77. Counter timing diagram, internal clock divided by 2 179
Figure 78. Counter timing diagram, internal clock divided by 4 179

Figure 79. Counter timing diagram, internal clock divided by N 180
Figure 80. Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded) 180
Figure 81. Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded) 181
Figure 82. Counter timing diagram, internal clock divided by 1 182
Figure 83. Counter timing diagram, internal clock divided by 2 182
Figure 84. Counter timing diagram, internal clock divided by 4 182
Figure 85. Counter timing diagram, internal clock divided by N 183
Figure 86. Counter timing diagram, update event when repetition counter is not used 183
Figure 87. Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6 184
Figure 88. Counter timing diagram, internal clock divided by 2 185
Figure 89. Counter timing diagram, internal clock divided by 4, TIMx_ARR=0x03 185
Figure 90. Counter timing diagram, internal clock divided by N 185
Figure 91. Counter timing diagram, update event with ARPE=1 (counter underflow) 186
Figure 92. Counter timing diagram, update event with ARPE=1 (counter overflow) 186
Figure 93. Control circuit in normal mode, internal clock divided by 1 187
Figure 94. TI2 external clock connection example 187
Figure 95. Control circuit in external clock mode 1 188
Figure 96. Capture/compare channel (example: channel 1 input stage) 189
Figure 97.Capture/compare channel 1 main circuit 189
Figure 98. Output stage of capture/compare channel (channel 1) 190
Figure 99. PWM input mode timing 192
Figure 100. Output compare mode, toggle on OC1 193
Figure 101. Edge-aligned PWM waveforms (ARR = 8) 194
Figure 102. Center-aligned PWM waveforms (ARR = 8) 195
Figure 103. Example of one pulse mode 196
Figure 104. Example of counter operation in encoder interface mode 199
Figure 105. Example of encoder interface mode with IC1FP1 polarity inverted 199
Figure 106. Control circuit in reset mode 200
Figure 107. Control circuit in gated mode 201
Figure 108. Control circuit in trigger mode 202
Figure 109. Master/Slave timer example 202

Figure 110.Gating timer 3 with OC1REF of timer 1 203
Figure 111. Gating timer 3 with enable of timer 1 204
Figure 112. Triggering timer 3 with update of timer 1 205
Figure 113. Triggering timer 3 with enable of timer 1 205
Figure 114. Triggering timer 1 and 3 with timer 1 TI1 input 206
Figure 115. Basic timer block diagram 228
Figure 116. Counter timing diagram with prescaler division change from 1 to 2 229
Figure 117. Counter timing diagram with prescaler division change from 1 to 4 229
Figure 118. Counter timing diagram, internal clock divided by 1 230
Figure 119. Counter timing diagram, internal clock divided by 2 231
Figure 120. Counter timing diagram, internal clock divided by 4 231
Figure 121. Counter timing diagram, internal clock divided by N 231
Figure 122. Counter timing diagram, update event when ARPE=0 (TIM14_ARR not preloaded) 232
Figure 123. Counter timing diagram, update event when ARPE=1 (TIM14_ARR preloaded).. 232
Figure 124. Control circuit in normal mode, internal clock divided by 1 233
Figure 125. Capture/compare channel (example: channel 1 input stage) 233
Figure 126. Capture/compare channel 1 main circuit 234
Figure 127. Output stage of capture/compare channel (channel 1) 234
Figure 128. Output compare mode, toggle on OC1 237
Figure 129. Edge-aligned PWM waveforms (ARR = 8) 238
Figure 130. Independent watchdog block diagram 251
Figure 131. Watchdog block diagram 259
Figure 132. Window watchdog timing diagram 260
Figure 133. SPI block diagram 265
Figure 134. Single master/single slave application 266
Figure 135. Data clock timing diagram 268
Figure 136. Start and Stop conditions 285
Figure 137. 7-bit address format 285
Figure 138. 10-bit address format 286
Figure 139. Master transmit protocol 287

Figure 140. Master receive protocol 287
Figure 141. Start Byte transfer 288
Figure 142. DR register 289
Figure 143. Master transmit - empty Tx FIFO 289
Figure 144. Master receive - empty Tx FIFO 289
Figure 145. Multiple master arbitration 290
Figure 146. Multiple master clock synchronization 291
Figure 147. SCL master clock generation 291
Figure 148. SCL master clock synchronization 292
Figure 149. I2C function block diagram 293
Figure 150. I2C interface master flow chart 296
Figure 151. Interrupt mechanism 297
Figure 152. UART block diagram 318
Figure 153. UART timing 319
Figure 154. Status bit change during transmission 320
Figure 155. UART block diagram 324
Figure 156. UART block diagram 325
Figure 157. RX pin sampling scheme 326
Figure 158. Block diagram of MM32 series and CPU level debug support 348

Table

Table 1. Memory mapping 3
Table 2. Flash module organization 9
Table 3. Flash memory read protection status 17
Table 4. Flash memory read unprotection status 18
Table 5. Flash interrupt request 18
Table 6. Option byte format 18
Table 7. Option byte organization 19
Table 8. Description of the option bytes 19
Table 9. Flash register overview 20
Table 10. Overview of CRC registers 28
Table 11. Low-power mode summary 32

Table 12. SLEEP NOW mode 33
Table 13. SLEEP ON EXIT mode 34
Table 14. Stop mode 35
Table 15. Standby mode 35
Table 16. Overview of power control registers 36
Table 17. RCC register overview 45
Table 18. Port bit configuration table 62
Table 19. Output MODE bits 62
Table 20. Advanced timers TIM1 67
Table 21. General-purpose timers TIM3/14 67
Table 22. UARTs 67
Table 23. SPI 68
Table 24. I2C 68
Table 25. ADC 68
Table 26. Other IOs 68
Table 27. Debug interface signals 69
Table 28. Overview of GPIO registers 69
Table 29. Vector table for this product series 77
Table 30. Overview of EXTI registers 83
Table 31. Overview of ADC registers 94
Table 32. Counting direction versus encoder signals 138
Table 33. TIM1 register overview 144
Table 34. TIMx internal trigger connection 150
Table 35. Output control bits for complementary OCx and OCxN channels with break feature 164
Table 36. Counting direction versus encoder signals 198
Table 37. Overview of TIMx registers 206
Table 38. TIMx internal trigger connection 212
Table 39. Output control bit for standard Ocx channels 223
Table 40. Overview of TIM14 registers 238
Table 41. Output control bit for standard OCx channels 247
Table 42. IWDG timeout period (input clock at 40 KHz (LSI)) 251

Table 43. Overview of IWDG registers 252
Table 44. Overview of WWDG registers 261
Table 45. SPI status 270
Table 46. Baud rate formula 271
Table 47. Overview of SPI registers 271
Table 48. First bytes for I2C 286
Table 49. Setting and clearing the interrupt bits 297
Table 50. Overview of I2C register description 298
Table 51. DISSLAVE(bit 6) and MASTER(bit 0) settings 302
Table 52. UART interrupt requests 327
Table 53. Overview of UART registers 327
Table 54. Overview of SYSCFG registers 342
Table 55. Overview of memory size register description 347
Table 56. SWJ debug port pins 351
Table 58. ID codes 352
Table 59. Packet request (8-bits) 353
Table 60. ACK response (3 bits) 353
Table 61. ACK response (33 bits) 353
Table 63. Overview of DBG registers 356
Table 64. Revision history 359

1 Memory and Bus Architecture

Memory and Bus Architecture

1.1 System architecture

The main system consists of:

- One master:
  - CPU core system bus (S-bus)
- Three slaves:
  - Internal SRAM
  - Internal Flash memory
  - Bridges from AHB to APB (APBx), which connect all the APB peripherals

These are interconnected using a multilayer AHB bus architecture as shown in Figure 1:

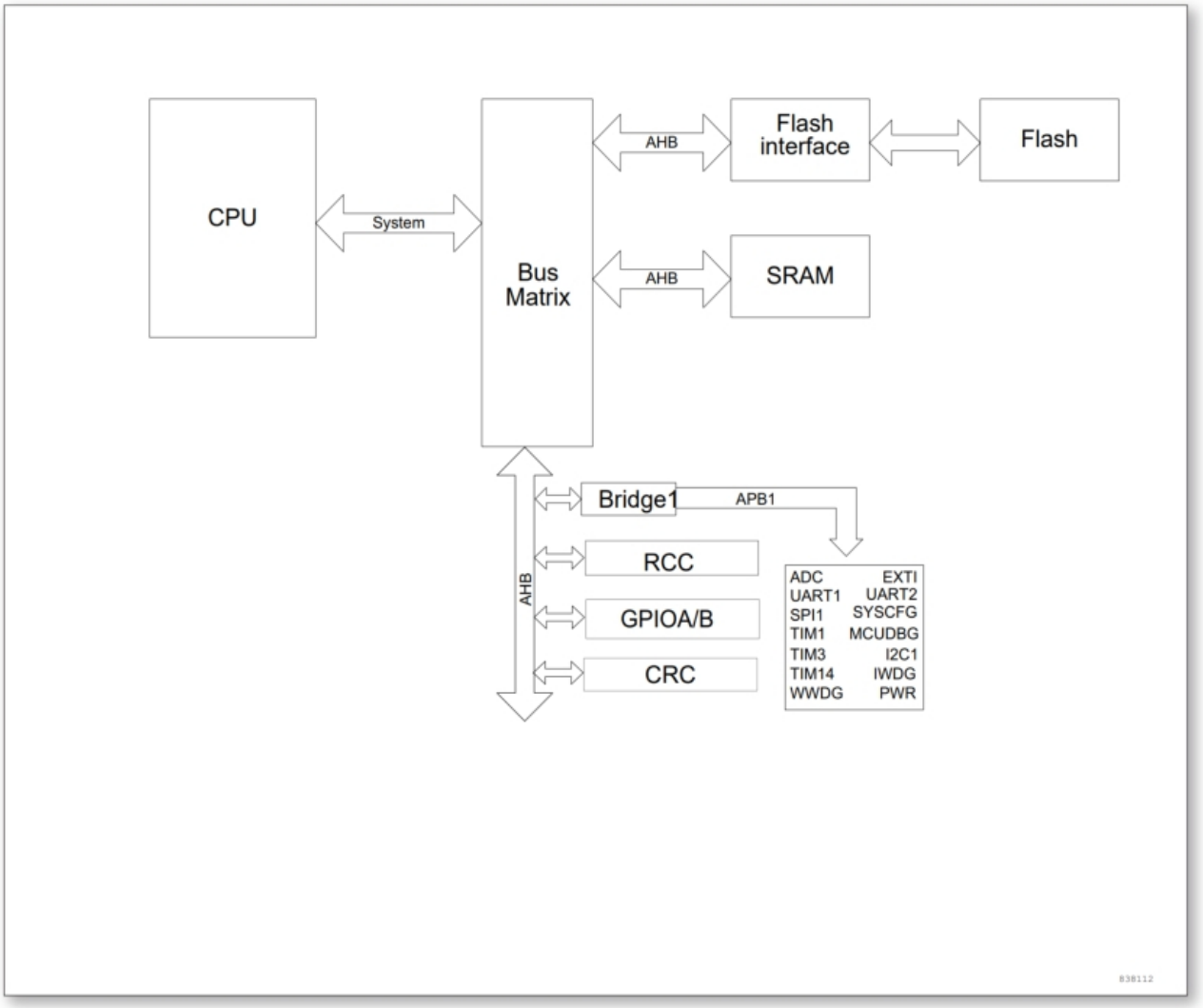


Figure 1. System architecture

System bus

This bus connects the system bus of the CPU core (peripherals bus) to a BusMatrix which manages the arbitration of the core.

BusMatrix

The BusMatrix manages the access of the core system bus. The BusMatrix is composed of the master module bus and the slave module bus.

AHB peripherals are connected on system bus through a BusMatrix.

AHB2APB bridges - APB

The AHB to APB bridges provide full synchronous connections between the AHB and the APB buses. After each device reset, all peripheral clocks are disabled (except for the SRAM and Flash). Before using a peripheral you have to enable its clock in the RCC_AHBENR or RCC_APB1ENR register.

Note: When a 8- or 16-bit access is performed on an APB register, the access is automatically transformed into a 32-bit access: the bridge automatically scales up the 16- or 8-bit data to feed the 32-bit vector.

1.2 Memory organization

1.2.1 Introduction

Program memory, data memory, registers and I/O ports are organized within the same linear 4-Gbyte address space.

The bytes are coded in memory in Little Endian format. The lowest address byte in a word is considered the word's least significant byte and the highest address byte the most significant.

The addressable memory space is divided into 8 main blocks, each of 512 MB. All the other memory areas that are not allocated to on-chip memories and peripherals are considered "Reserved" address space. For details, refer to the memory mapping and register addressing section and the peripheral sections.

1.2.2 Memory mapping and register addressing

See the memory mapping diagram corresponding to each peripheral section for a comprehensive diagram of the memory mapping.

The table below gives the start addresses of all built-in peripherals.

Table 1. Memory mapping

Bus	Boundary Address	Size	Peripheral	Remark
Flash	0x0000 0000–0x0000 3FFF	16 KB	Main Flash memory, system memory or SRAM, depending on the BOOT configuration	
	0x0000 4000–0x07FF FFFF	~ 127 MB	Reserved	
	0x0800 0000–0x0800 3FFF	16 KB	Main Flash memory	
	0x0800 0000–0x1FFD FFFF	~ 383 MB	Reserved	
	0x1FFE 0000–0x1FFE 01FF	0.5 KB	Reserved	
	0x1FFE 0200–0x1FFE 0FFF	3 KB	Reserved	
	0x1FFE 1000–0x1FFE 1BFF	3 KB	Reserved	
	0x1FFE 1C00–0x1FFF F3FF	~ 256 MB	Reserved	
	0x1FFF F400–0x1FFF F7FF	1 KB	System memory	
	0x1FFF F800–0x1FFF F80F	16 B	Option bytes	
	0x1FFF F810–0x1FFF FFFF	~2 KB	Reserved	

Bus	Boundary Address	Size	Peripheral	Remark
SRAM	0x2000 0000–0x2000 07FF	2 KB	SRAM	
	0x2000 0700–0x2FFF FFFF	~ 255 MB	Reserved	
APB1	0x4000 0000–0x4000 03FF	1 KB	Reserved	
	0x4000 0400–0x4000 07FF	1 KB	TIM3	
	0x4000 0800–0x4000 0BFF	8 KB	Reserved	
	0x4000 2800–0x4000 2BFF	1 KB	Reserved	
	0x4000 2C00–0x4000 2FFF	1 KB	WWDG	
	0x4000 3000–0x4000 33FF	1 KB	IWDG	
	0x4000 3400–0x4000 37FF	1 KB	Reserved	
	0x4000 3800–0x4000 3BFF	1 KB	Reserved	
	0x4000 4000–0x4000 43FF	1 KB	Reserved	
	0x4000 4400–0x4000 47FF	1 KB	UART2	
	0x4000 4800–0x4000 4BFF	3 KB	Reserved	
	0x4000 5400–0x4000 57FF	1 KB	I2C1	
	0x4000 5800–0x4000 6BFF	5 KB	Reserved	
	0x4000 6C00–0x4000 6FFF	1 KB	Reserved	

Bus	Boundary Address	Size	Peripheral	Remark
APB1	0x4000 7000–0x4000 73FF	1 KB	PWR	
	0x4000 7400–0x4000 FFFF	35 KB	Reserved	
	0x4001 0000–0x4001 03FF	1 KB	SYSCFG	
	0x4001 0400–0x4001 07FF	1 KB	EXTI	
	0x4001 0800–0x4001 23FF	7 KB	Reserved	
	0x4001 2400–0x4001 27FF	1 KB	ADC1	
	0x4001 2800–0x4001 2BFF	1 KB	Reserved	
	0x4001 2C00–0x4001 2FFF	1 KB	TIM1	
	0x4001 3000–0x4001 33FF	1 KB	SPI1	
	0x4001 3400–0x4001 37FF	1 KB	DBGMCU	
	0x4001 3800–0x4001 3BFF	1 KB	UART1	
	0x4001 3C00–0x4001 3FFF	1 KB	Reserved	
	0x4001 4000–0x4001 43FF	1 KB	TIM14	
	0x4001 4400–0x4001 47FF	1 KB	Reserved	
	0x4001 4800–0x4001 4BFF	1 KB	Reserved	
	0x4001 4C00–0x4001 7FFF	13 KB	Reserved	
AHB	0x4002 0000–0x4002 03FF	1 KB	Reserved	
	0x4002 0400–0x4002 0FFF	3 KB	Reserved	
	0x4002 1000–0x4002 13FF	1 KB	RCC	
	0x4002 1400–0x4002 1FFF	3 KB	Reserved	
	0x4002 2000–0x4002 23FF	1 KB	Flash Interface	
	0x4002 2400–0x4002 2FFF	3 KB	Reserved	
	0x4002 3000–0x4002 33FF	1 KB	CRC	
	0x4002 3400–0x47FF FFFF	~ 127 MB	Reserved	

Bus	Boundary Address	Size	Peripheral	Remark
AHB	0x4800 0000–0x4800 03FF	1 KB	GPIOA	
	0x4800 0400–0x4800 07FF	1 KB	GPIOB	
	0x4800 0800–0x4800 0BFF	1 KB	Reserved	
	0x4800 0C00–0x4800 0FFF	1 KB	Reserved	
	0x4800 1000–0x5FFF FFFF	~ 384 MB	Reserved	

1.3 Embedded SRAM

A static SRAM up to 2K bytes is embedded. It can be accessed as bytes (8 bits), half-words (16 bits) or full words (32 bits). The SRAM start address is 0x2000 0000.

- The SRAM on the data bus reaches up to 2K bytes. It can be accessed by the CPU through the fastest system clock without inserting any latency.

1.4 Flash memory overview

The Flash memory has two different storage areas:

- Main Flash memory block, which includes the application and the user data zones (if necessary)
- Information block, which includes four parts:
  - Option bytes — containing the hardware and storage protection user configuration options.
  - System memory — refer to the embedded Flash memory section.

The Flash memory interface executes commands and data access based on the AHB protocol. Its prefetch buffer function enables the acceleration of the code execution by the CPU.

2 Embedded Flash Memory (FLASH)

Embedded Flash Memory (FLASH)

2.1 Key Flash features

- The Flash memory is up to 16K bytes
- Memory organization:
  - Main Flash memory block: Up to 4K bytes (4K × 32 bits)
  - Information block:
    - System memory: Up to 1K bytes (1K × 8 bits)
    - Option bytes: Up to 2 × 8 bits

The Flash memory interface features:

- Read interface with prefetch buffer (2 x 32-bit words)
- Option byte loader
- Flash Program/Erase operation
- Read/Write protection

(FLASH)

Module	Name	Address	Size (byte)
Main memory block	Page 0	0x0800 0000 - 0x0800 03FF	1K
	Page 1	0x0800 0400 - 0x0800 07FF	1K
	Page 2	0x0800 0800 - 0x0800 0BFF	1K
	Page 3	0x0800 0C00 - 0x0800 0FFF	1K
	…	…	…
	…	…	…
	Page 15	0x0800 3C00 - 0x0800 3FFF	1K
Information block	System memory	0x1FFF F400 - 0x1FFF F7FF	1K
	Option bytes	0x1FFF F800 - 0x1FFF F80F	16
Flash memory interface registers	FLASH_ACR	0x4002 2000 - 0x4002 2003	4
	FLASH_KEYR	0x4002 2004 - 0x4002 2007	4
	FLASH_OPTKEYR	0x4002 2008 - 0x4002 200B	4
	FLASH_SR	0x4002 200C - 0x4002 200F	4
	FLASH_CR	0x4002 2010 - 0x4002 2013	4
	FLASH_AR	0x4002 2014 - 0x4002 2017	4
	Reserved	0x4002 2018 - 0x4002 201B	4
	FLASH_OBR	0x4002 201C - 0x4002 201F	4
	Reserved	0x4002 2020 - 0x4002 2023	4

- Low power consumption mode

2.2 Flash memory function description

2.2.1 Flash memory organization

The Flash memory is organized as 32-bit wide memory cells that can be used for storing both code and data. The main Flash memory block is partitioned by 16 pages (1K bytes per page) or 4 sectors (4K bytes per sector), with write protection on a sector-by-sector basis (refer to the memory protection part).

Table 2. Flash module organization

2.2.2 Read operation of the Flash memory

The embedded Flash module can be addressed directly, as a common memory space. Any read operation that accesses the content of the Flash module should go through a dedicated judgmental process.

Instruction and data fetches are performed through the AHB bus in the way that is specified as the option in the Flash access control register (FLASH_ACR).

- Instruction fetch: CPU operation speed can be accelerated after enabling the prefetch buffer.
- Latency: The number of wait bits for correct reading.

Instruction fetch

The CPU fetches the instructions over the AHB bus. The prefetch block aims at increasing the efficiency of instruction fetch.

Prefetch buffer

Prefetch buffer (2 x 64-bit blocks): The buffer is enabled automatically after each reset. A block can be completely updated on a single read from the Flash memory as the size of each block (32 bits) matches the bandwidth of the Flash memory. The implementation of this prefetch buffer makes a faster CPU execution possible. The CPU can fetch one word up to 32 bits at one time, as it fetches one instruction at a time with the next instruction readily available in the prefetch buffer.

Prefetch controller

The prefetch controller decides to access the Flash memory depending on the available space in the prefetch buffer. The controller initiates a read request when there is at least one block free in the prefetch buffer. After reset, the default state of the prefetch buffer is on. The prefetch buffer should be switched on/off only when SYSCLK is lower than 24MHz and no prescaler is applied on the AHB clock (SYSCLK must be equal to HCLK). The prefetch buffer is usually switched on/off during the initialization routine, while the MCU is running on the internal 8MHz oscillator.

Note: The latency for prefetch buffer must be kept on when using a prescaler different from 1 on the AHB clock.

Access latency

In order to maintain the correct reading from the Flash memory, the speed ratio of the prefetch controller has to be specified in the LATENCY[2:0] of the Flash access control register. This value gives the number of cycles needed to insert between each access of the Flash memory and next access. After reset, the value is zero, ie. no wait cycle is inserted.

2.2.3 Flash write and erase operations

The embedded Flash memory supports the in-circuit programming as well as the in-application programming.

The in-circuit programming (ICP) method refers to updating the contents of the Flash memory in the circuit by loading the user codes into the microcontroller. ICP offers a quick and efficient method and eliminates unnecessary socketing of devices during chip programming.

In contrast to the ICP method, in-application programming (IAP) can use any communication interface supported by the MCU (I/Os, USB, UART, I2C, SPI, etc.)

to download programs or data. IAP allows the user to re-program the application while the application is running. Nevertheless, part of the application has to be previously programmed in the Flash memory using ICP.

The write and erase operations can be performed within the whole operating voltage range of the product. They are performed by the following 7 registers:

- Key register (FLASH_KEYR)
- Option byte key register (FLASH_OPRKEYR)
- Flash control register (FLASH_CR)
- Flash status register (FLASH_SR)
- Flash address register (FLASH_AR)
- Option byte register (FLASH_OBR)
- Write protection register (FLASH_WRPR)

An ongoing Flash memory write operation will not block the CPU as long as the CPU does not access the Flash memory. That is to say, during a write/erase operation to the Flash memory, any attempt to read the Flash memory will stall the bus. The read operation will proceed correctly once the write/erase operation has completed. This means that instruction or data fetches cannot be made while a write/erase operation is ongoing.

For write/erase operations on the Flash memory, the internal oscillator (HSI) must be ON.

Unlocking the Flash memory

After reset, the Flash memory is protected in default to prevent unexpected erase operation. The FLASH_CR register is not allowed to be rewritten, unless an unlocking sequence is executed for the FLASH_CR register to enable access permission to the FLASH_CR. This sequence consists of two write operations:

- Write Key 1 = 0x45670123
- Write Key 2 = 0xCDEF89AB

Any wrong sequence locks up the FLASH_CR register until the next reset.

Also a bus error is returned on a wrong key sequence, which will then trigger a hardware error response. This is done immediately if KEY1 is wrong, or if KEY1 has been correctly written but KEY2 is wrong.

Main Flash memory programming

The main Flash memory can be programmed 16 bits at a time. One programming operation is done when a half-word (16 bits) is written into the corresponding address with the PG bit of the FLASH_CR to be 1. Any attempt to write data that are not half-word long will result in a hardware error response.

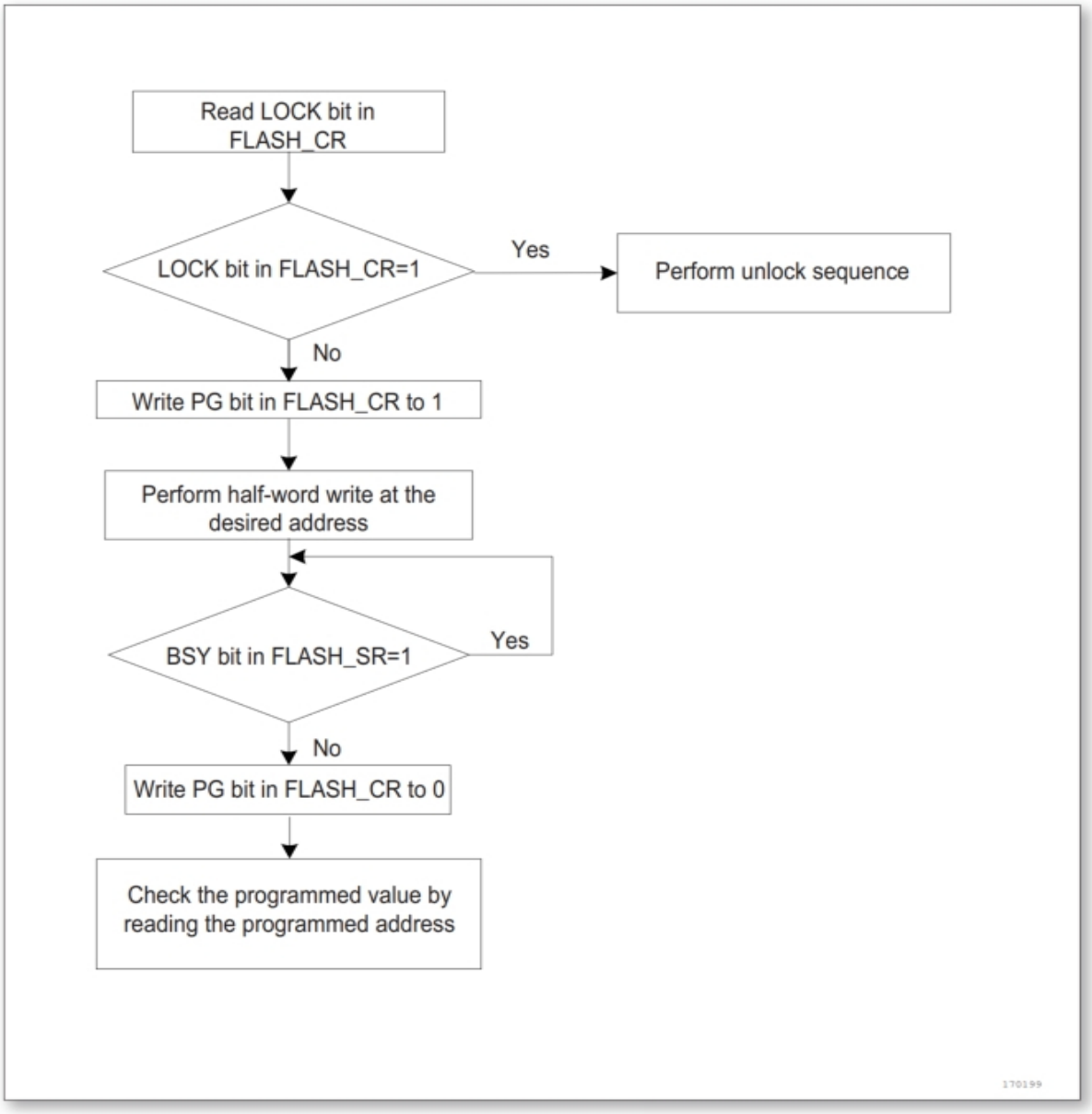


Figure 2. Programming procedures

The Flash memory interface prefetches the bytes to be programmed and checks whether they are all set to 1. If not, the program operation is skipped and a programming error warning is issued by the PGERR bit in the FLASH_SR register.

If the address to be programmed is write-protected by the FLASH_WRPR register, the program operation is skipped and a programming error warning is issued, either. The end of the program operation is indicated by the EOP bit in the FLASH_SR register.

The main Flash memory programming sequence in standard mode is as follows:

- Check that last operation is finished by checking the BSY bit in the FLASH_SR register
- Set the PG bit in the FLASH_CR register
- Perform the data write in a unit of half-word at the desired address
- Wait for the BSY bit in the FLASH_SR register to return to zero
- Read the data and verify

Note: These registers are not to be written when the BSY bit of the FLASH_SR register is set to 1.

Flash memory erase

The Flash memory can be erased page by page or completely as mass erase.

Page erase

To erase a page, the procedure below should be followed:

- Check that last operation is finished by checking the BSY bit in the FLASH_SR register
- Set the PER bit to 1 in the FLASH_CR register
- Write the FLASH_AR register to select a page to erase
- Set the STRT bit to 1 in the FLASH_CR register
- Wait for the BSY bit in the FLASH_SR register to return to zero
- Read the erased page and verify

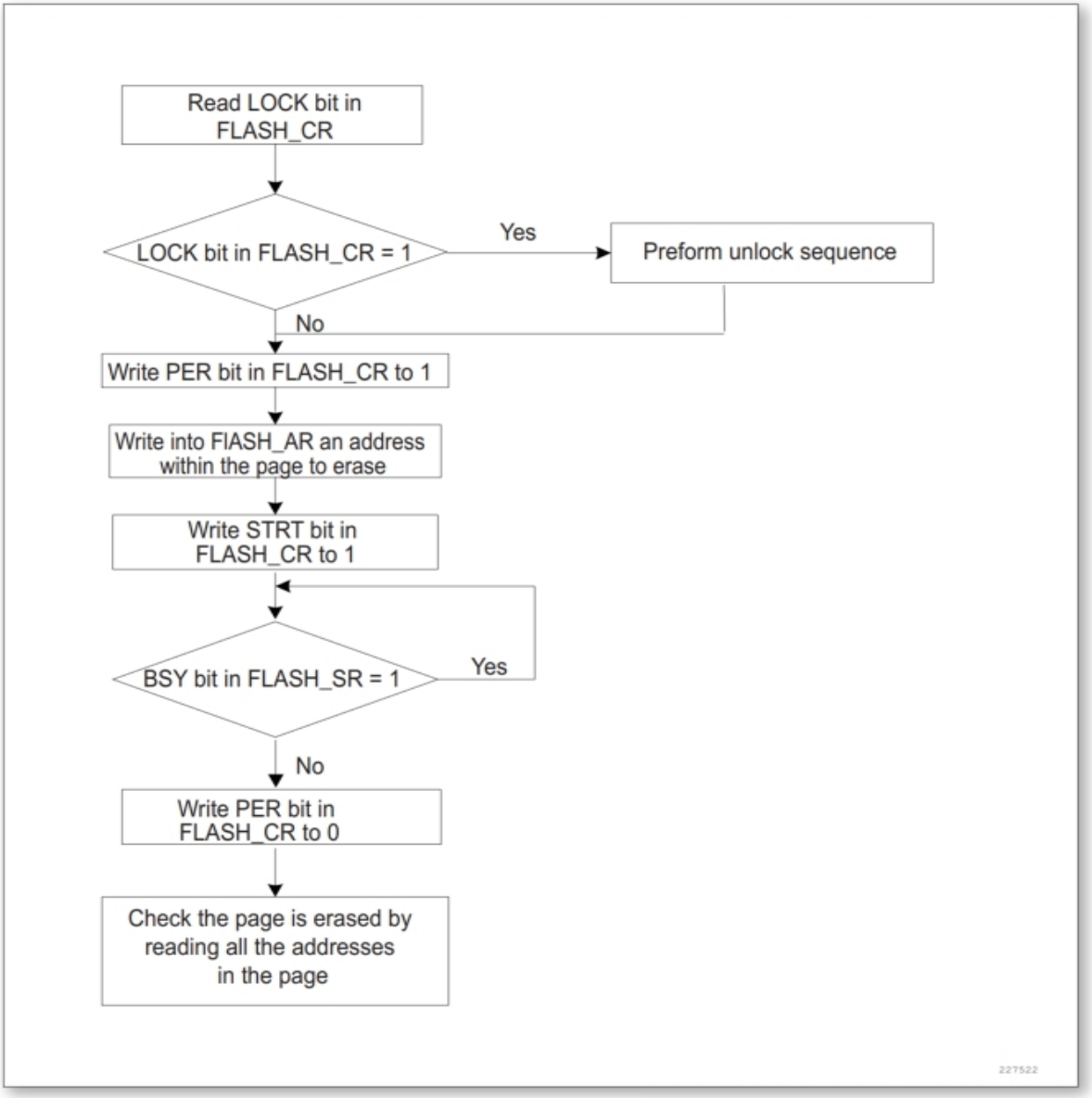


Figure 3. Flash register page erase procedures

Mass Erase

The Mass Erase command can be used to completely erase the whole Flash user area.

The information block is unaffected by this instruction. The specific sequence is as follows:

- Check that last operation is finished by checking the BSY bit in the FLASH_SR register
- Set the MER bit to 1 in the FLASH_CR register
- Set the STRT bit to 1 in the FLASH_CR register
- Wait for the BSY bit to return to zero
- Read all the pages and verify

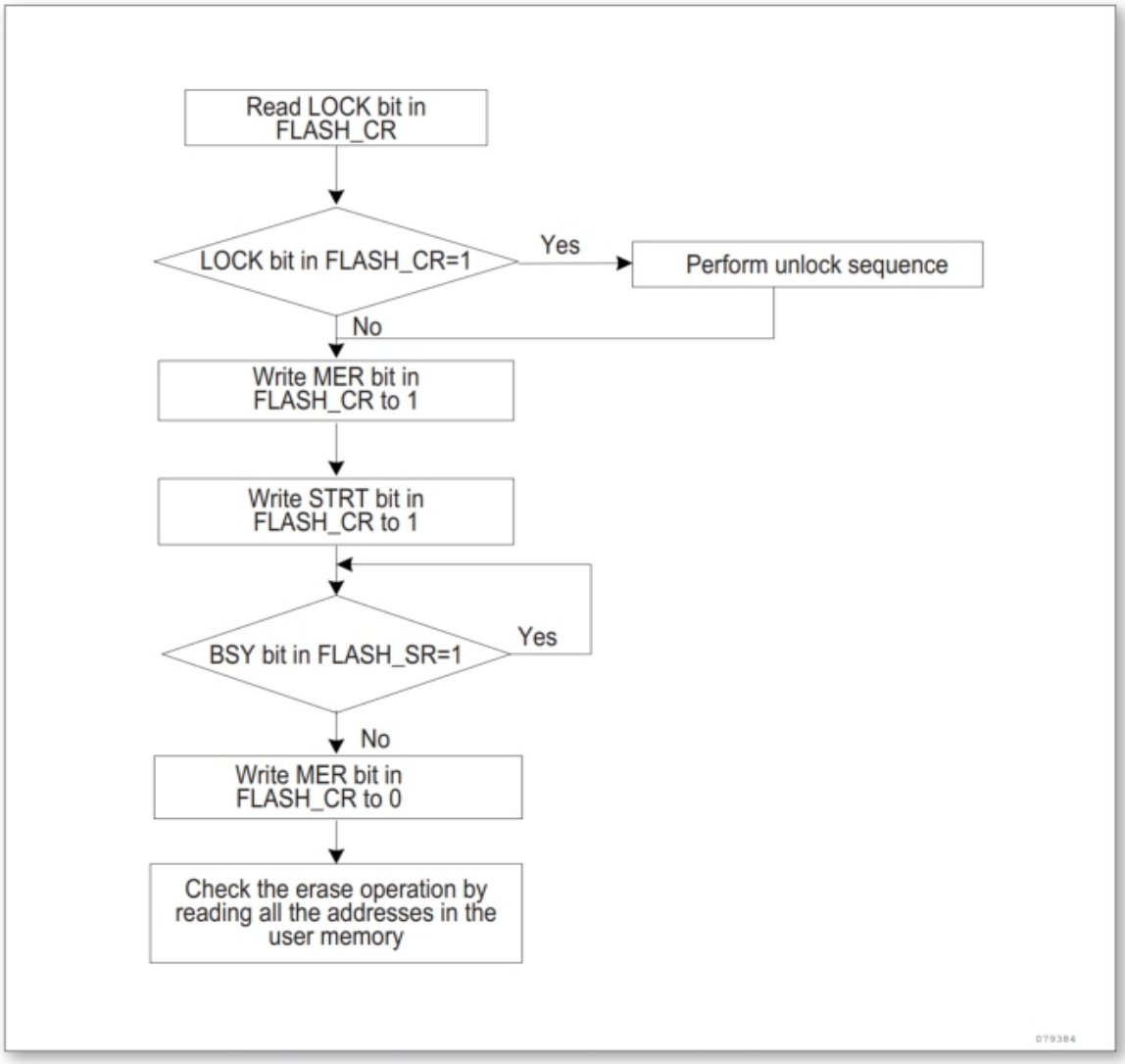


Figure 4. Flash register mass erase procedures

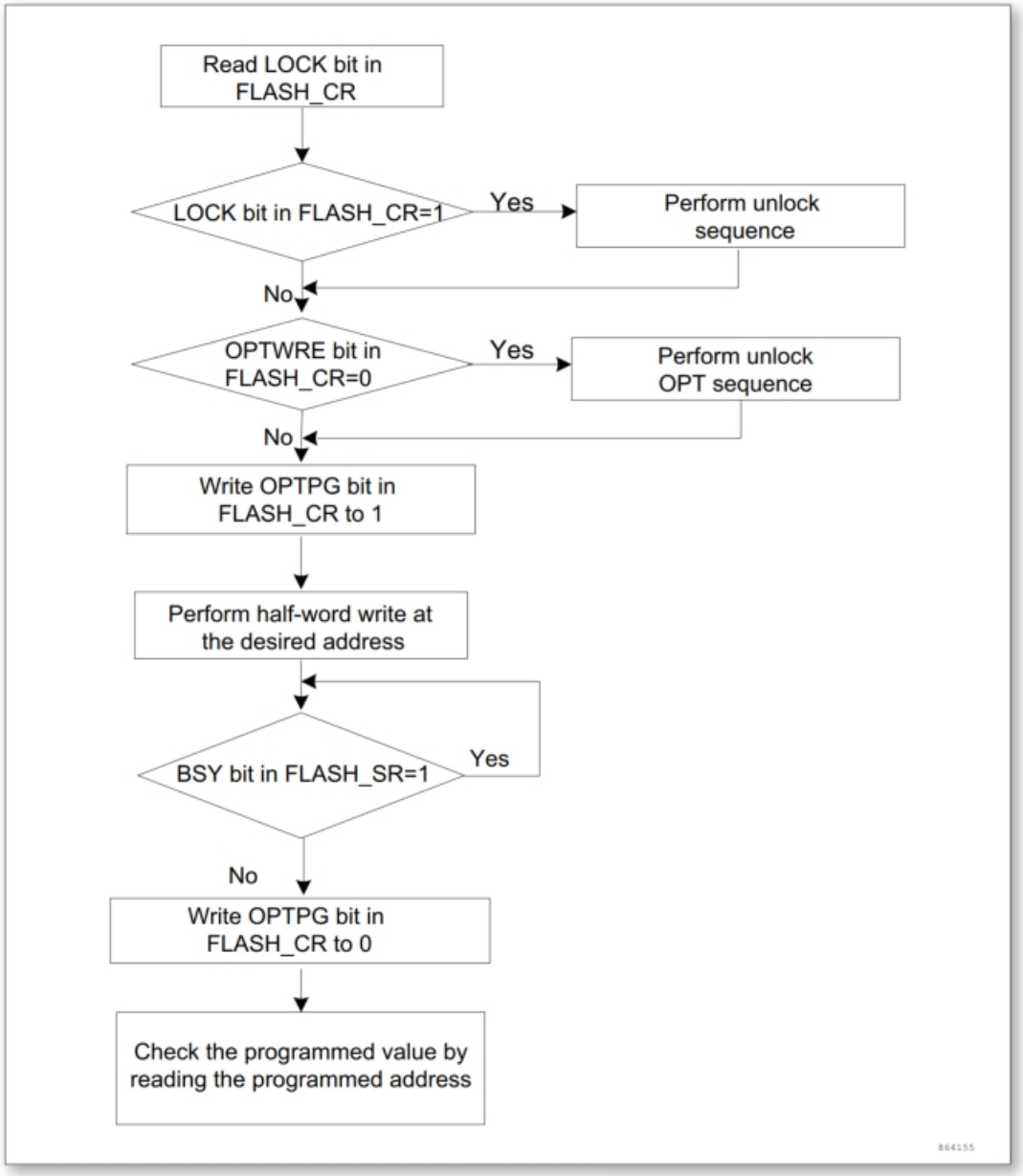
Option byte programming

The option bytes are programmed differently from normal user addresses, including two for write protection and one for hardware configuration. After unlocking the access permission to the Flash, the writing of keys into the FLASH_OPTKEYR register has to be performed. Once it is done, the OPTWRE bit in the FLASH_CR register will be set to '1'. Then the user has to set the OPTPG bit in the FLASH_CR register and perform a half-word write operation at the desired address. Similarly, the option bytes are checked to confirm they are set as 1. If not, the relevant operation is skipped and a error warning is issued by the PGERR bit in the FLASH_SR register. The end of the program operation is indicated by the EOP bit in the FLASH_SR register.

The option byte is 16-bit data. The significant data is the 8 lower bits, while the 8 higher bits are the complement of the 8 lower bits. During the programming process, the user has to calculate the complement of the 8 lower bits and consider it as the 8 higher bits first, and write it together with the 8 lower bits as a half-word. Ensure that the value written into the option byte always makes the 8 higher bits and the 8 lower bits as the radix-minus-one complements to each other. The sequence is as follows:

- Check that last operation is finished by checking the BSY bit in the FLASH_SR register
- Unlock the OPTWRE bit in the FLASH_CR register
- Set the OPTPG bit to 1 in the FLASH_CR register
- Write the data (half-word) to the desired address
- Wait for the BSY bit to return to zero
- Read the data and verify that a mass erase is automatically performed when the protection option byte is changed from protected to unprotected.

  If the user wants to change an option other than the protection option, then the mass erase is not performed. This mechanism is used to protect the content of the Flash memory.



(FLASH)

Figure 5. Option byte programming procedures

Erase procedure

The option byte erase sequence is as follows:

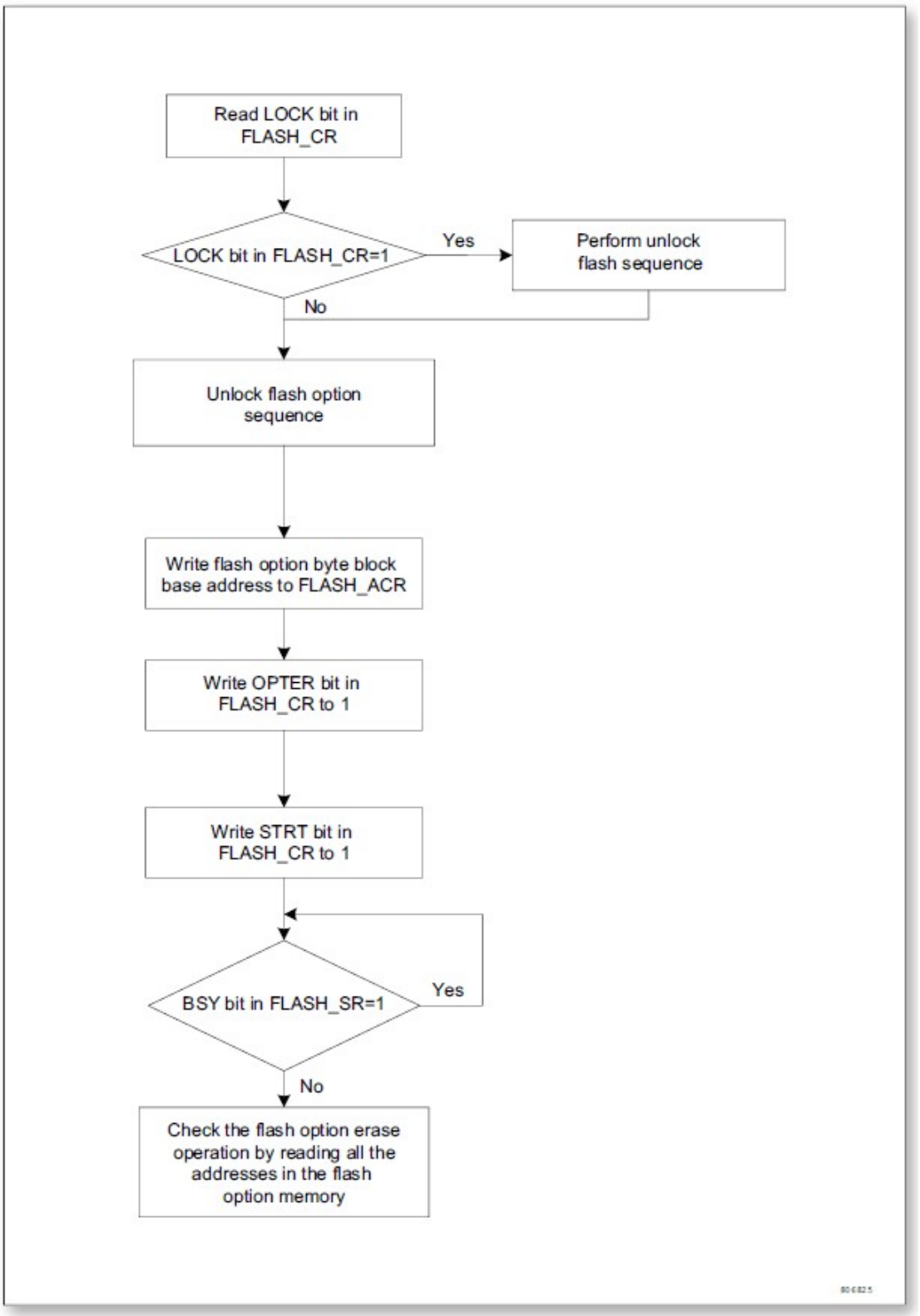


Figure 6. Option byte erase procedure

2.3 Memory protection

The user area of the Flash memory can be protected against read by untrusted code. The Flash memory can also be protected against unwanted erase due to program runaway.

The write-protection granularity is then of one sector (four pages).

2.3.1 Read protection

The read protection is activated by setting the RDP option to a half word and then, by re-applying a system POR (power-on reset) to reload new RDPs.

Note: If the read protection is set while the debugger is still connected through SWD/JTAG, apply a POR instead of a system reset (without debugger connection).

Set as read protection

Refer to the half-word programming method in the option byte area, and write a half word for the RDP option in sequence to the corresponding address

1. Set the Flash AR address value to 0x1FFFF800 and execute the block erase
2. Write the desired value of 0x807F to 0x1FFFF800
3. Reset the desired system via software, or turn off the power before turning on it again

Once corresponding values have been written to the protection byte:

- Main Flash memory read access is not allowed except for the user code (when booting from main Flash memory itself with the non-debug mode).
- After the read protection is activated, the Flash is inaccessible in the debug mode (sram boot and debug modes). The flash's own programs should not occupy the 4KB space.
- Pages 0-3 are automatically write-protected. The rest of the memory can be programmed by the code executed from the main Flash memory (for IAP or data storage, etc.), but it is protected against write/erase (but not against mass erase) in debug mode or when booting from the embedded SRAM.
- All features for loading code into and executing code from the embedded SRAM via SWD are still active. Booting from embedded SRAM via SWD is also allowed and this can be used to disable the read protection.
- When executing code from the embedded SRAM to access the main Flash memory, Flash memory accesses through the DMA, SWV (serial wire viewer), SWD (serial wire debug), ETM () and boundary scan are not allowed.

The Flash memory is protected when the RDPs contain the following values:

Table 3. Flash memory read protection status

RDP Byte Value	Read Protection Status
RDP=0x807F @0x1FFFF800	Protected

Note: If the corresponding address value for the option byte block is not 0xFFFF, the erase of option byte block should be executed first. Erasing the option byte block will not trigger a automatic mass erase or change the protection status.

Unprotection

To disable the read protection from the embedded SRAM:

Refer to the half-word programming method in the option byte area, and write a half word for the RDP option in sequence to the corresponding address:

1. Set the Flash AR address value to 0x1FFFF800 and execute the block erase
2. Write the desired value of 0x5AA5 to 0x1FFFF800 to trigger a mass erase of the main Flash memory. At this stage the read protection is not activated.

Apply a POR reset to reload the option bytes, to disable the read protection.

Table 4. Flash memory read unprotection status

RDP Byte Value	Read Protection Status
Erase the 0x1FFFF800 option block Write RDP=0x5AA5 @0x1FFFF800 Trigger the main Flash mass erase for 0x08000000	Protection is disabled

2.3.2 Write protection of main memory

The write protection is controlled in a unit of sector (4 pages). The WRP bit in the option bytes should be configured, and then the system reset will load the new option bytes to enable the protection. If a write or an erase operation is performed on a protected sector, The WRPRTERR flag bit is set on the FLASH_SR.

2.3.3 Option byte write protection

The option byte blocks are always read-accessible and write-protected by default. To gain write access (program/erase) to the option byte blocks, a sequence of keys (same as for lock) has to be written into the OPTKEYR. It then gives write access to the option byte blocks and this is indicated by OPTWRE bit in the FLASH_CR register. Write access can be disabled by clearing the bit.

2.4. Flash interrupt

Table 5. Flash interrupt request

Interrupt Event	Event Symbol	Enable Control Bit
End of operation	EOP	EOPIE
Write protection error	WRPRTERR	ERRIE
Programming error	PGERR	ERRIE

2.5 Option byte description

Option bytes are configured by the user depending on the application requirements. As an example, the watchdog may be selected in hardware or software mode.

A 32-bit word is split up as follows in the option bytes:

Table 6. Option byte format

Bits 31 ~ 24	Bits 23 ~ 16	Bits 15 ~ 8	Bits 7 ~ 0
complemented option byte 1	Option byte 1	complemented option byte 0	Option byte 0

Note: Complements must be implemented by the user application software.

The organization of these option bytes inside the option byte block is as below.

The option bytes can be read from the memory locations listed in the table below or from the option byte register (FLASH_OBR).

Note: The new programmed option bytes (user, read/write protection) take effect after a system reset.

Table 7. Option byte organization

Address	[31:24]	[23:16]	[15:8]	[7:0]
0x1FFF F800	nUSER	USER	nRDP	RDP
0x1FFF F804	nData1	Data1	nData0	Data0
0x1FFF F808			nWRP0	WRP0
0x1FFF F80C				

Table 8. Description of the option bytes

Flash memory address	Option bytes
0x1FFF F800	Bits [31:24] nUSER Bits [23:16] USER: User option byte (stored in FLASH_OBR[9:2]) This byte is used to configure the following features: Select the watchdog event: hardware or software Note: Only bit [16] is used, and bits [23:17] are not used. Bit 16: WDG_SW 0: hardware watchdog 1: software watchdog Bits [15:8]: nRDP Bits [7:0]: RDP: read protection option byte The read protection helps the user protect the code stored in Flash memory. It is activated by setting the RDP option byte. When this option byte is programmed to a correct value (RDP = 0x807F), read access to the Flash memory is forbidden. (The result of RDP enabled/disabled is stored in FLASH_OBR[1].)
0x1FFF F804	Datax: two bytes for user data The address can be programmed using the option byte programming procedure. Bits [31:24]: nData1 Bits [23:16]: Data1 (stored in FLASH_OBR[25:18]) Bits [15:8]: nData0 Bits [7:0]: Data0 (stored in FLASH_OBR[17:10])

Flash memory address	Option bytes
0x1FFF F808	WRPx: Flash memory write protection option bytes Bits [15:8]: nWRP0 Bits [7:0]: WRP0 (stored in FLASH_WRPR[7:0]) Each bit of the option byte WRPx is used to protect 4 pages in main memory. 0: write protection active 1: write protection not active In total, one user option byte is used to protect the 32-Kbyte main Flash memory. WRP0: write-protects for pages 0 to 31

On every system reset, the option byte loader (OBL) reads the information block and stores the data into the option byte register (FLASH_OBR). Each option bit also has its complement in the information block. During option bit loading, by verifying the option bit and its complement, it is possible to check that the loading has correctly taken place. If this is not the case, an option byte error (OPTERR) is generated. When an error occurs, the corresponding option byte is forced to 0xFF. The verification is disabled when the option byte and its complement are both equal to 0xFF (after-erase state).

All option bits (but not their complements) are available to configure the microcontroller. The option byte registers are accessible in read mode by the CPU.

2.6. Flash register descriptions

Table 9. Flash register overview

Offset	Acronym	Register Name	Reset	Section
0x00	FLASH_ACR	Flash access control register	0x00000038	Section 2.6.1
0x04	FLASH_KEYR	FPEC key register	0x00000000	Section 2.6.2
0x08	FLASH_OPTKEYR	Flash OPTKEY register	0x00000000	Section 2.6.3
0x0C	FLASH_SR	Flash status register	0x00000000	Section 2.6.4
0x10	FLASH_CR	Flash control register	0x00000080	Section 2.6.5
0x14	FLASH_AR	Flash address register	0x00000000	Section 2.6.6
0x1C	FLASH_OBR	Option byte register	0x03FFFC1C	Section 2.6.7
0x20	FLASH_WRPR	Write protection register	0xFFFFFFFF	Section 2.6.8

2.6.1 Flash access control register (FLASH_ACR)

Address offset: 0x00

Reset value: 0x0000 0038

(FLASH)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										PRFTBS	PRFTBE	Res.	LATENCY		
										r	rw		rw	rw	rw

Bit	Field	Type	Reset	Description
31:5	Reserved		0x0000 0001	Reserved，and Bit 5 is always read as 1.
4	PRFTBE	rw	0x01	Prefetch buffer enable 0: prefetch buffer is disabled 1: prefetch buffer is enabled
3	Reserved			Reserved and always read as 1.
2:0	LATENCY	rw	0x00	Latency These bits represent the ratio of the SYSCLK (system clock) period to the Flash access time. 000: zero wait state, if 0 < SYSCLK ≤ 24MHz 001: one wait state, if 24MHz < SYSCLK ≤ 48MHz

2.6.2 FPEC key register (FLASH_KEYR)

Address offset: 0x04

Reset value: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:0	FKEYR	w	0x0000 0000	FPEC key (Flash key) These bits represent the keys to unlock the FPEC.

Note: These bits are all write-only and will return a 0 when read.

2.6.3 Flash OPTKEY register (FLASH_OPTKEYR)

Address offset: 0x08
Reset value: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31 : 0	OPTKEYR	w	0x0000 0000	Option byte key These bits represent the keys to unlock the OPTWRE.

Note: These bits are all write-only and will return a 0 when read.

2.6.4 Flash status register (FLASH_SR)

Address offset: 0x0C

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										EOP	WRPRTERR	Res.	PGERR	Res.	BSY
										rc_w1	rc_w1		rc_w1		r

Bit	Field	Type	Reset	Description
31:6	Reserved			Reserved, always read as 0.
5	EOP	rc_w1	0x00	End of operation Set by hardware to '1' when a Flash operation (programming/erase) is completed. Its status can be cleared by writing a '1' to it. Note: EOP status is set at the end of each successful program or erase operation.
4	WRPRTERR	rc_w1	0x00	Write protection error Set by hardware to '1' when a write-protected address of the Flash memory to be programmed. Its status can be cleared by writing a '1' to it.
3	Reserved			Reserved and always read as 0.
2	PGERR	rc_w1	0x00	Programming error Set by hardware to '1' when an address to be programmed contains a value different from '0xFFFF'. Its status can be cleared by writing a '1' to it. Note: The STRT bit in the FLASH_CR register should be cleared before starting a programming operation.

Bit	Field	Type	Reset	Description
1	Reserved			Reserved and always read as 0.
0	BSY	r	0x00	Busy This indicates that a Flash operation is in progress. This is set to '1' on the beginning of a Flash operation and cleared (as '0') when the operation finishes or when an error occurs.

2.6.5 Flash control register (FLASH_CR)

Address offset: 0x10

Reset value: 0x0000 0080

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			EOPIE	Res.	ERRIE	OPTWRE	Res.	LOCK	STRT	OPTER	OPTPG	Res.	MER	PER	PG
			rw		rw	rc_w0		rw	rw	rw	rw		rw	rw	rw

Bit	Field	Type	Reset	Description
31:13	Reserved			Reserved, always read as 0.
12	EOPIE	rw	0x00	End of operation interrupt enable This bit enables the interrupt generation when the EOP bit in the FLASH_SR register goes to '1'. 0: interrupt generation disabled 1: interrupt generation enabled
11	Reserved			Reserved and always read as 0.
10	ERRIE	rw	0x00	Error interrupt enable This bit enables the interrupt generation on an FPEC error (when PGERR/WRPRTERR are set to '1' in the FLASH_SR register). 0: interrupt generation disabled 1: interrupt generation enabled
9	OPTWRE	rc_w0	0x00	Option byte write enable When set to '1', the option bytes can be programmed. This bit is set to '1' on writing the correct key sequence to the FLASH_OPTKEYR register. Writing '0' to it by software can clear this bit.
8	Reserved			Reserved and always read as 0.

Bit	Field	Type	Reset	Description
7	LOCK	rw	0x01	Lock Write to ‘1’ only. When it is set to ‘1’, it indicates that the FPEC and FLASH_CR are locked. This bit is cleared to ‘0’ by hardware after detecting the correct unlock sequence. In the event of unsuccessful unlock operation, this bit is not allowed to be changed until the next system reset.
6	STRT	rw	0x00	Start This bit triggers an ERASE operation when set to ‘1’. This bit can be set to ‘1’ only by software and cleared to ‘0’ automatically when the BSY bit is set to ‘1’.
5	OPTER	rw	0x00	Option byte erase Option byte erase chosen.
4	OPTPG	rw	0x00	Option byte programming Option byte programming chosen.
3	Reserved			Reserved and always read as 0.
2	MER	rw	0x00	Mass erase Erase of all user pages chosen.
1	PER	rw	0x00	Page erase Page erase chosen.
0	PG	rw	0x00	Programming Flash programming chosen.

2.6.6 Flash address register (FLASH_AR)

Address offset: 0x014

Reset value: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31 : 0	FAR	w	0x0000 0000	Flash Address Chooses the address to program when programming is ongoing, or a page to erase when Page Erase is selected. Note: Write access to this register is blocked when the BSY bit in the FLASH_SR register is set to '1'.

Updated by hardware with the currently/last used address. For Page Erase operations, this register should be updated to indicate the page to erase.

2.6.7 Option byte register (FLASH_OBR)

Address offset: 0x1C

Reset value: 0x03FF FC1C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						Data1								Data0	
						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
Data0						Reserved					nRST _STDBY	nRST _STOP	WDG _SW	RDPRT.	OPTERR
r	r	r	r	r	r								r	r	r

Bit	Field	Type	Reset	Description
31:26	Reserved			Reserved, always read as 0.
25:18	Data1	r	0xFF	Data1
17:10	Data0	r	0xFF	Data0
9:5	Reserved			Reserved
4	nRST_STDBY	r	0x01	Reset event when entering the Standby mode 0: generate a reset event when entering the Standby mode 1: do not generate a reset event when entering the Standby mode
3	nRST_STOP	r	0x01	Reset event when entering the Stop mode 0: generate a reset event when entering the Stop mode 1: do not generate a reset event when entering the Stop mode
2	WDG_SW	r	0x01	Select the watchdog event 0: hardware watchdog 1: software watchdog

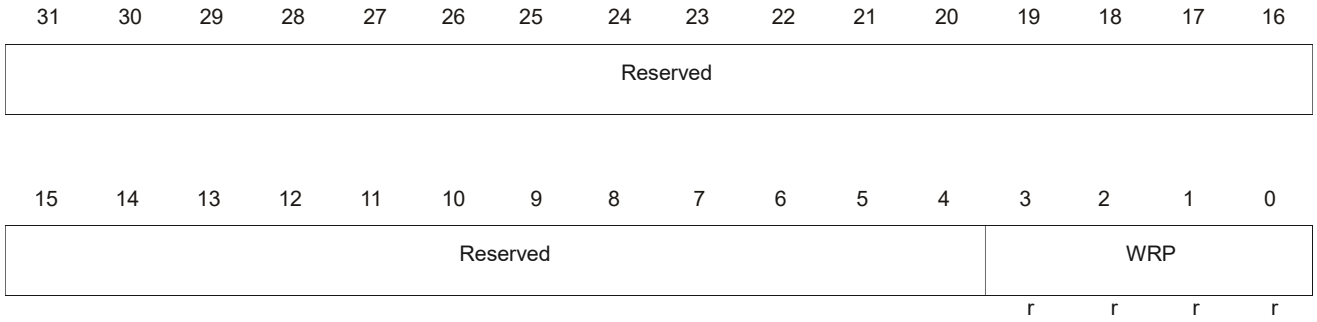
Bit	Field	Type	Reset	Description
1	RDPRT	r	0	Read protection level status When set to '1', this indicates that the Flash memory is read-protected. Note: This bit is read-only.
0	OPTERR	r	0x00	Option byte error When set to '1' , this indicates that the option byte and its complement do not match. Note: This bit is read-only.

The value used to reset this register is related to the value written to the option byte. The value used to reset the OPTERR bit is related to the comparison between the option byte and its complement when loading the option byte.

2.6.8 Write protection register (FLASH_WRPR)

Address offset: 0x20

Reset value: 0xFFFF FFFF



Bit	Field	Type	Reset	Description
31 31:4	Reserved			Reserved, always read as 0.
3:0	WRP	r	0x0F	Write protection This register contains the write-protection option bytes loaded by the OBL. 0: write protection active 1: write protection not active Note: These bits are read-only.

3 Cyclic Redundancy Check (CRC) Calculation Unit

Cyclic Redundancy Check (CRC) Calculation Unit

3.1 CRC introduction

The CRC (cyclic redundancy check) calculation unit is used to get a CRC calculation result from a 32-bit full word based on a fixed generator polynomial. Among other applications, CRC-based techniques are used to verify data transmission or storage correctness or storage integrity. In the scope of the EN/IEC60335-1 standard, they offer a means of verifying the Flash memory integrity. The CRC calculation unit helps compute a signature of the software during runtime, to be compared with a reference signature generated at linktime and stored at a given memory location.

3.2 CRC main features

- Uses CRC-32 (Ethernet) polynomial: 0x4C11DB7

  $X_{32} + X_{26} + X_{23} + X_{22} + X_{16} + X_{12} + X_{11} + X_{10} + X_8 + X_7 + X_5 + X_4 + X_2 + X + 1$

- Single input/output 32-bit data register
- CRC computation done in 3 AHB clock cycles (HCLK)
- General-purpose 8-bit register (can be used for temporary storage)

The CRC calculation unit block diagram is shown below.

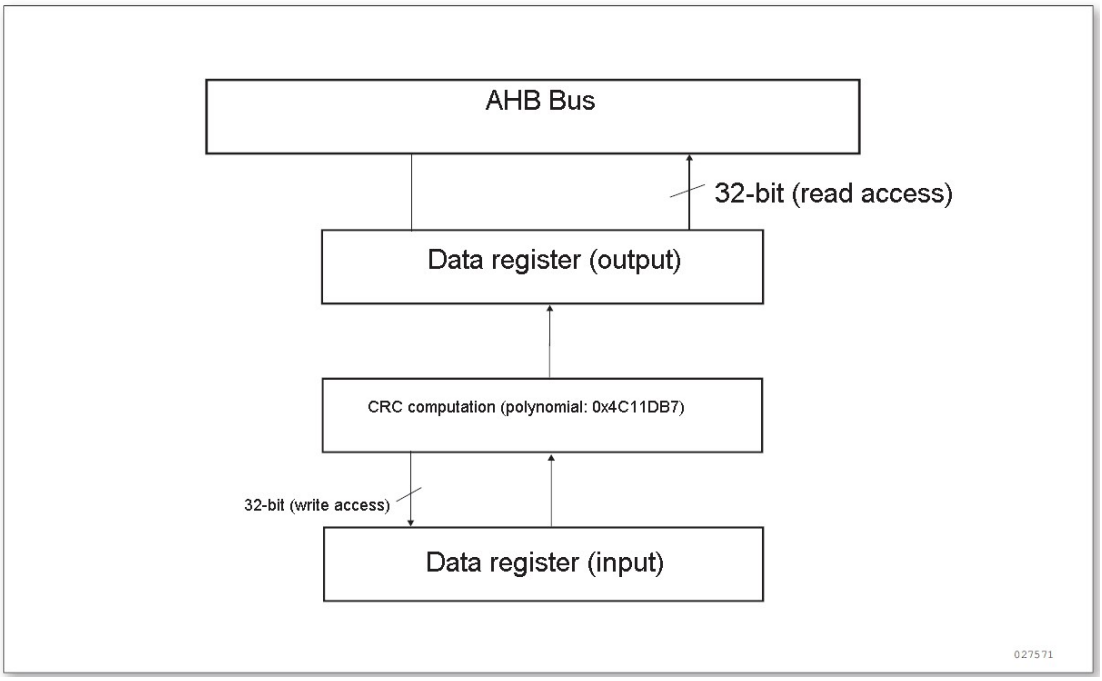


Figure 7. CRC calculation unit block diagram

3.3 CRC function introduction

The CRC calculation unit mainly consists of a single 32-bit data register, which:

- is used as an input register to enter new data in the CRC calculator (when writing into the register)
- holds the result of the previous CRC calculation (when reading the register)

Each write operation into the data register creates a combination of the previous CRC calculation value and the new one (CRC computation is done on the whole 32-bit data word, and not byte per byte).

The write operation is stalled until the end of the CRC computation, thus allowing back-to-back write accesses or consecutive write and read accesses against the CRC_DR register.

The CRC_DR register can be reset to 0xFFFF FFFF with the RESET bit in the CRC_CTRL register.

This operation does not affect the contents of the CRC_IDR register.

3.4 CRC registers

The CRC calculation unit contains two data registers and a control register.

Table 10. Overview of CRC registers

Offset	Acronym	Register Name	Reset	Section
0x00	CRC_DR	CRC data register	0xFFFFFFFF	Section 3.4.1
0x04	CRC_IDR	CRC independent data register	0x00000000	Section 3.4.2
0x08	CRC_CTRL	CRC control register	0x00000000	Section 3.4.3

3.4.1 CRC data register (CRC_DR)

Address offset: 0x00

Reset value: 0xFFFF FFFF

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

Bit	Field	Type	Reset	Description
31:0	DR	rw	0xFFFF FFFF	DR: data register bits Read as an input register to return the CRC calculation result when writing new data into the CRC calculator.

3.4.2 CRC independent data register (CRC_IDR)

Address offset: 0x04

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31:8	Reserved			always read as 0.
7:0	IDR	rw	0x00	IDR: general-purpose 8-bit data register bits Can be used as a temporary storage location for one byte. This register is not affected by CRC resets generated by the RESET bit in the CRC_CTRL register.

Note: This register is not involved in the CRC calculation, thus it can be used as a storage for any data.

3.4.3 CRC control register (CRC_CTRL)

Address offset: 0x08

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31:1	Reserved			always read as 0.
0	RESET	w	0x00	RESET: CRC reset calculation unit Sets the data register to 0xFFFF FFFF. This bit can only be set to ‘1’, and it is automatically cleared to ‘0’ by hardware.

4 Power Control (PWR)

Power Control (PWR)

4.1 Power supplies

The chip requires a 2.0-to-5.5 V operating voltage supply (V_{DD}). An embedded regula tor is used to supply the required core power.

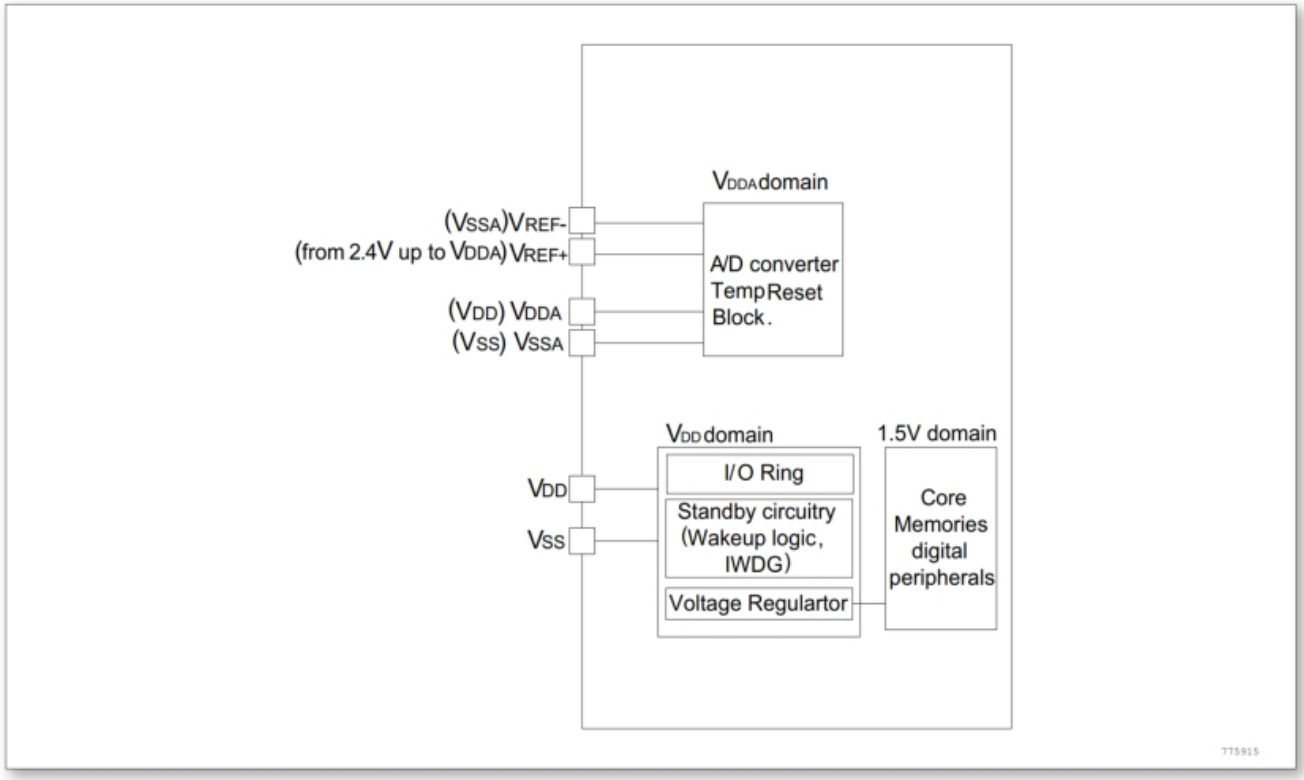


Figure 8. Power supply block diagram

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

4.1.1 Independent A/D converter supply and reference voltage

To improve conversion accuracy, the ADC has an independent power supply which can be separately filtered and shielded from spikes on the PCB.

- The ADC supply input is available on a V_{DDA} pin.
- An isolated supply ground connection is provided on pin V_{SSA}.

When the V_{REF-} pin is available (according to package), it must be tied to V_{SSA}.

4.1.2 Voltage regulator

The voltage regulator is always enabled after reset. It works in three different modes de pending on the application modes.

• In Run mode, the regulator supplies power to the 1.5V domain (core, memories and peripherals)
normally.

• In Stop mode, the regulator supplies low-power to the 1.5V domain, preserving contents of registers
and SRAM.

• In Standby mode, the regulator is powered off. The contents of the registers and SRAM are all lost.

4.2 Power supply supervisor

4.2.1 Power-on reset (POR)/power-down reset (PDR)

The chip has an integrated POR/PDR circuitry that allows proper operation starting from/d own to 2.0V of supply.

The device remains in Reset mode when V_{DD}/V_{DDA} is below a specified threshold, V_{POR}/V_{PDR}, without the need for an external reset circuitry. For more details concerning the powe r-on/power-down reset, refer to the electrical characteristics of the datasheet.

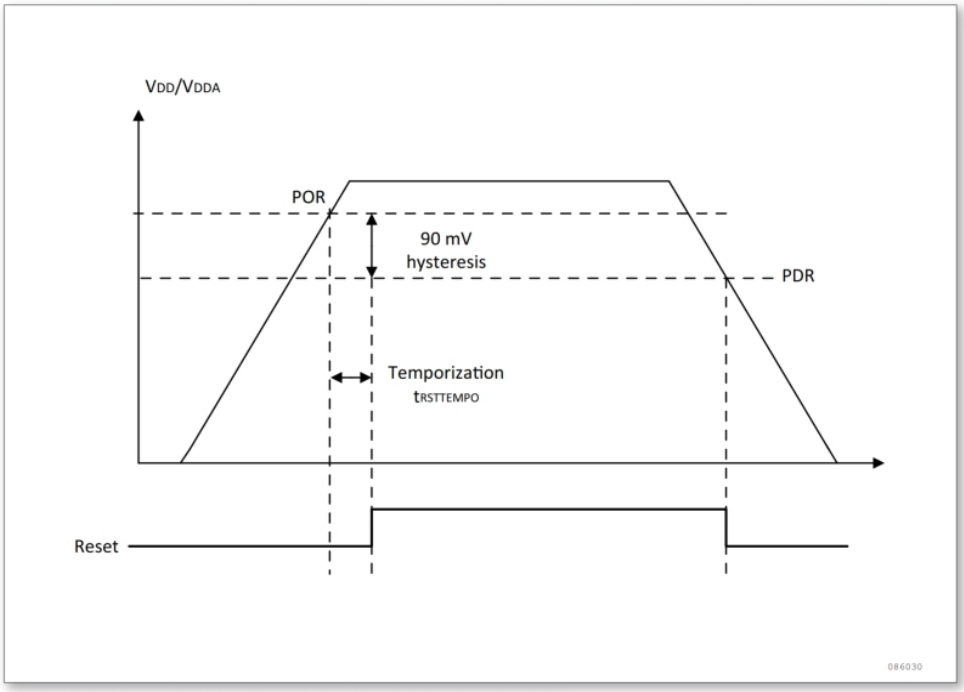


Figure 9. Power-on reset/power-down reset waveform

4.2.2 Programmable voltage detector (PVD)

The PVD can be used by the user to monitor the power supply by comparing V_{DD} to the PLS bits in the power control register (PWR_CR). These bits can select the threshold of the monitored voltage. The PVD is enabled by setting the PVDE bit.

A PVDO flag is available, in the power control/status register (PWR_CSR), to indicate if V_{DD} is higher or lower than the PVD voltage threshold. This event is internally connected to the EXTI line16 and can generate an interrupt if the interrupt is enabled through the EXTI registers. The PVD interrupt can be generated when V_{DD} drops below the PVD threshold or when V_{DD} rises above the PVD threshold depending on EXTI line16 rising/falling edge configuration. As an example the service routine could perform emergency shutdown tasks.

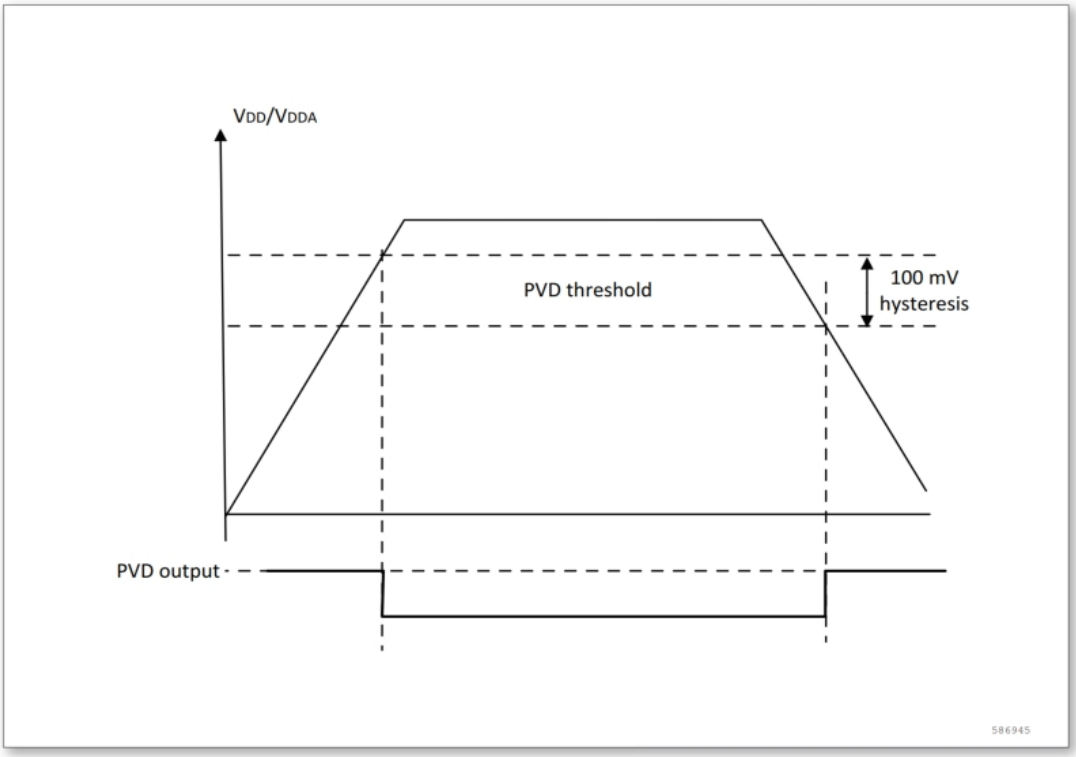


Figure 10. PVD thresholds

4.3 Low-power modes

By default, the microcontroller is in Run mode after a system or a power reset. Several low-power modes are available to save power when the CPU does not need to be kept running, for example when waiting for an external event. It is up to the user to select the mode that gives the best compromise between low-power consumption, short startup time and available wakeup sources.

The chip features three low-power modes:

- Sleep mode (CPU stops working, but all peripherals including CPU peripherals like NVIC, SysTick, etc. are kept running)
- Stop mode (all clocks are stopped while the SRAM and register contents are kept intact)
- Standby mode (the core power is off and the contents of the registers and SRAM are all lost) In addition, the power consumption in Run mode can be reduced by one of the following means:
- Slowing down the system clock frequency
- Turning off the clocks on the APB and AHB buses when they are unused

Table 11. Low-power mode summary

Mode	Entry	Wakeup	Effect on 1.5V domain clocks	Effect on V_{DD} domain clocks	Voltage regulator
SLEEP (SLEEP NOW or SLEEP ON EXIT)	WFI (Wait for Interrupt)	Any interrupt	The CPU clock is off. Other clocks and ADC clock are not affected.	None	ON
	WFE (Wait for Event)	Wakeup event			
Stop	PDDS bit SLEEPDEEP bit WFI or WFE	Any EXTI (configured in the EXTI registers)	All 1.5V domain clocks OFF	HSI and HSE oscillatorsOFF	ON
Standby	PDDS bit SLEEPDEEP bit WFI or WFE	WKUP pin rising edge, external reset in NRST pin, IWDG reset			OFF

4.3.1 Slowing down system clocks

In Run mode, the speed of any system clock (SYSCLK, HCLK, PCLK1) can be reduced

by programming the prescaler registers. These prescalers can also be used to slow down peripheral clocks before entering Sleep mode.

For more details refer to: Clock configuration register (RCC_CFGR).

4.3.2 Peripheral clock gating

In Run mode, the HCLK and PCLKx for individual peripherals and memories can be stopped at any time to reduce power consumption.

To further reduce power consumption in Sleep mode the peripheral clocks can be disabled prior to executing the WFI or WFE instructions.

Peripheral clock gating is controlled by the AHB peripheral clock enable register (RCC_AHBENR) and APB1 peripheral clock enable register (RCC_APB1ENR).

4.3.3 Sleep mode

Entering Sleep mode

The Sleep mode is entered by executing the WFI or WFE instructions. Two options are available to select the Sleep mode entry mechanism, depending on the SLEEPONEXIT bit in the CPU System Control register:

- SLEEP-NOW: if the SLEEPONEXIT bit is cleared, the MCU enters Sleep mode as soon as WFI or WFE instruction is executed.
- SLEEP-ON-EXIT: if the SLEEPONEXIT bit is set, the MCU enters Sleep mode as soon as it exits the lowest priority interrupt service routine.

In the Sleep mode, all I/O pins keep the same state as in the Run mode.

Refer to Table 12 and Table 13 for details on how to enter Sleep mode.

Table 12. SLEEP NOW mode

SLEEP NOW mode	Description
Entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - SLEEPDEEP = 0 - SLEEPONEXIT = 0 Refer to the CPU system Control register.
Exit	If WFI was used for sleep mode entry: Interrupt: Refer to the interrupt vector table If WFE was used for sleep mode entry: Wakeup event: Refer to wakeup event management
Wakeup latency	None

Table 13. SLEEP ON EXIT mode

SLEEP ON EXIT mode	Description
Entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - SLEEPDEEP = 0 - SLEEPONEXIT = 1 Refer to the CPU system Control register.
Exit	If WFI was used for sleep mode entry: Interrupt or clear the CPU control register bit 1 If WFE was used for sleep mode entry: Wakeup event: Refer to wakeup event management
Wakeup latency	None

4.3.4 Stop mode

The Stop mode is based on the CPU deepsleep mode combined with peripheral clock gating. In Stop mode, the voltage regulator can be configured in normal mode, all clocks in the 1.5V domain are stopped, and the HSI and the HSE oscillators are disabled. SRAM and register contents are preserved.

In Stop mode, all I/O pins keep the same state as in the Run mode.

Entering Stop mode
Refer to Table 14 for details on how to enter the Stop mode.

The following features can be selected by programming individual control bits:

- Independent watchdog (IWDG): the IWDG is started by writing to its key register or by hardware option.
- Internal oscillator (LSI oscillator): this is configured by the LSION bit in the control/status register (RCC_CSR). The ADC can also consume power during the Stop mode, unless it is disabled before entering it. To disable it, the ADEN bit in the ADC_ADCFG register must be written to 0. Other unused GPIOs also consume power unless they are configured with analog inputs.

Exiting Stop mode
Refer to Table 14 for details on how to exit the Stop mode.

When exiting Stop mode by issuing an interrupt or a wakeup event, the HSI oscillator is selected as system clock. The clock frequency is the HSI divided by 6. When the voltage regulator operates in normal power consumption mode, an additional startup delay is incurred when waking up from Stop mode.

Table 14. Stop mode

Stop mode	Description
Entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - Set SLEEPDEEP bit in CPU system control register - Clear PDDS bit in power control register (PWR_CR) - Switch the system clock to LSI or HSI Note: To enter Stop mode, all flags of EXTI request bits (pending register (EXTI_PEND)) must be cleared. Otherwise, the Stop mode entry procedure is ignored and program execution continues.
Exit	WFI (Wait for Interrupt) while: Any EXTI line configured in interrupt mode (the corresponding EXTI vector must be enabled in the NVIC). Refer to the interrupt vector table. WFE (Wait for Event) while: Any EXTI line configured in event mode. Refer to the wakeup event management.
Wakeup latency	LSI or HSI wakeup time

4.3.5 Standby mode

The Standby mode allows to achieve the lowest power consumption. It is based on the CPU deepsleep mode, with the voltage regulator disabled. The 1.5 V domain is consequently powered off. The HSI oscillator and the HSE oscillator are also switched off. SRAM and register contents are lost. Only standby circuits remain powered.

Entering Standby mode

Refer to Table 15 for details on how to enter the Standby mode.

In Standby mode, the following features can be selected by setting individual control bits:

- Independent watchdog (IWDG): the IWDG is started by writing to its key register or by hardware option.
- Internal oscillator (LSI oscillator): this is configured by the LSION bit in the control/status register (RCC_CSR).

Exiting Standby mode

The microcontroller exits the Standby mode when an external reset (NRST pin), an IWDG reset, or a rising edge on the WKUP pin occurs. All registers are reset after wakeup from Standby except for power control/status register (PWR_CSR). After waking up from Standby mode, program execution restarts in the same way as after a reset (vector reset is fetched, etc.). The power control/status register (PWR_CSR) indicates that the core exits the Standby mode.

Refer to Table 15 for more details on how to exit Standby mode.

Table 15. Standby mode

Standby mode	Description
Entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - Set SLEEPDEEP bit in CPU system control register - Set PDDS bit in power control register (PWR_CR) - Clear WUF bit in power control/status register (PWR_CR)
Exit	WKUP pin rising edge, external reset in NRST pin, IWDG reset.
Wakeup latency	For voltage regulator startup in the reset phase

I/O states in Standby mode

In Standby mode, all I/O pins are high impedance except:

- Reset pin (PA14)
  Before entering the Standby mode, GPIO should be configured by software to floating inputs and the SFT_NRST_RMP bit of RCC_SYSCFG to 1. An external pull-up resistor is required, and low level is at least kept at 4us.
- WKUP pin (PA0)
  If EWUP isn't available, the PA0 is an analog port.
  If EWUP is available, the PA0 is a floating input.

Debug mode

By default, the debug connection is lost if the application puts the MCU in Stop or Standby mode while the debug features are used. This is due to the fact that the CPU core is no longer clocked.

However, by setting some configuration bits in the DBGMCU_CR register, the software can be debugged even when using the low-power modes extensively. For more details, refer to debug support for low-power modes.

4.4 Power control registers

Table 16. Overview of power control registers

Offset	Acronym	Register Name	Reset	Section
0x00	PWR_CR	Power control register	0x00000000	Section 4.4.1
0x04	PWR_CSR	Power control/status register	0x00000000	Section 4.4.2

4.4.1 Power control register (PWR_CR)

Address offset: 0x00

Reset value: 0x0000 0000 (cleared by wakeup from Standby mode)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			PLS				Reserved				PVDE	CSBF	CWUF	PDDS	Res.
			rw	rw	rw	rw					rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
31:13	Reserved			always read as 0.
12:9	PLS	rw	0x00	PVD level selection These bits are used to select the voltage threshold detected by the power voltage detector 0000: 1.8V 0100: 3.0V 1000: 4.2V 0001: 2.1V 0101: 3.3V 1001: 4.5V 0010: 2.4V 0110: 3.6V 1010: 4.8V 0011: 2.7V 0111: 3.9V Others: reserved Note: Refer to the electrical characteristics of the datasheet for more details.
8:5	Reserved			always read as 0
4	PVDE	rw	0x00	Power voltage detector enable 1: PVD enabled 0: PVD disabled
3	CSBF	w	0x00	Clear standby flag always read as 0 1: clear the SBF Standby Flag (write) 0: no effect
2	CWUF	w	0x00	Clear wakeup flag always read as 0 1: clear the WUF wakeup flag after 2 system clock cycles (write) 0: no effect
1	PDDS	rw	0x00	Power down deepsleep 1: enter Standby mode when the CPU enters deepsleep 0: enter Stop mode when the CPU enters deepsleep
0	LPDS	rw	0x00	Low-power deepsleep It works together with the PDDS bit when PDDS = 0. 1: voltage regulator in low-power mode during Stop mode 0: voltage regulator in normal power consumption mode during Stop mode During Stop mode, the current under the condition that LPDS = 1 is smaller than LPDS = 0. For details, refer to the corresponding chip datasheet.

4.4.2 Power control/status register (PWR_CSR)

Address offset: 0x04

Reset value: 0x0000 0000 (not cleared by wakeup from Standby mode)

Additional APB cycles are needed to read this register versus a standard APB read.

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							EWUP	Reserved					PVDO	SBF	WUF
							rw						rw	rw	r

Bit	Field	Type	Reset	Description
31:9	Reserved			always read as 0.
8	EWUP	rw	0x00	Enable WKUP pin 1 = WKUP pin is used for wakeup from Standby mode and forced in input pull down configuration (rising edge on WKUP pin wakes up the system from Standby mode) 0 = WKUP pin is used for general purpose I/O. An event on the WKUP pin does not wake up the CPU from Standby mode. Note: This bit is cleared by a system reset.
7:3	Reserved			always read as 0
2	PVDO	rw	0x00	PVD output This bit is valid only if PVD is enabled by the PVDE bit. 1 = V_{DD}/V_{DDA} is lower than the PVD threshold selected by the PLS 0 = V_{DD}/V_{DDA} is higher than the PVD threshold selected by the PLS Note: The PVD is stopped by Standby mode. For this reason, this bit is equal to 0 after Standby or reset until the PVDE bit is set.
1	SBF	rw	0x00	Standby flag This bit is set by hardware and cleared only by a POR/PDR (power on reset/power down reset) or by setting the CSBF bit in the power control register (PWR_CR). 1 = system entering the Standby mode 0 = system not in the Standby mode

Bit	Field	Type	Reset	Description
0	WUF	r	0x00	Wakeup flag This bit is set by hardware and cleared only by a POR/PDR (power on reset/power down reset) or by setting the CWUF bit in the power control register (PWR_CR). 0 = no wakeup event occurred Note: An additional event is detected if the WKUP pin is enabled (by setting the EWUP bit) when the WKUP pin level is already high.

5 Reset and Clock Control (RCC)

Reset and Clock Control (RCC)

5.1 Reset

There are two types of reset, defined as system reset and power-on reset.

5.1.1 System reset

A system reset sets all registers to their reset values except the reset flags in the clock control register (RCC_CSR), LSI oscillator enable flags, standby and wakeup flags in the power control register (PWR_CSR), external reset flags of software mapping in the system configuration register, and the DBG control register (DBG_CR).

A system reset is generated when one of the following events occurs:

1. A low level on the nRST pin (external reset)
2. Window watchdog end of count condition (WWDG reset)
3. Independent watchdog end of count condition (IWDG reset)
4. A software reset (SW reset)
5. A CPU deadlock reset
6. A PVD reset

The reset source can be identified by checking the reset flags in the RCC_CSR control status register.

Note: A system reset doesn't influence the nRST configuration if the SFT_NRST_RMP bit of RCC_SYSCFG is set to '1', PA14 is mapped to the external reset of nRST, and the low level is at least 4us.

Software reset

A software reset is available if the SYSRESETREQ bit in the CPU application interrupt and reset control register (AIRCR) is set to '1'.

5.1.2 Power reset

A power reset is generated when one of the following events occurs:

1. Power-on/power-down reset (POR/PDR reset)
2. When exiting Standby mode

A power reset sets all registers to their reset values.

The reset source in Figure 11 will act on the RESET pin and it is always kept at a low level during the reset phase. The RESET service routine vector is fixed at address 0x0000_0004.

(RCC)

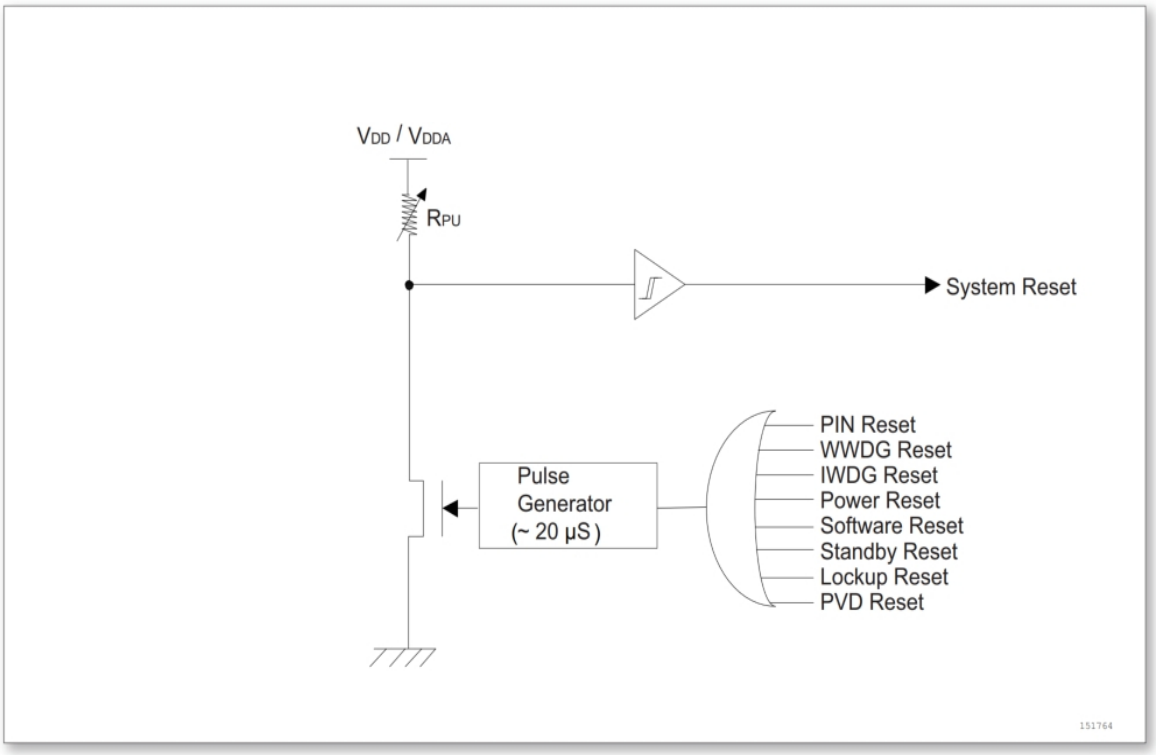


Figure 11. Reset circuit

5.2 Clocks

Four different clock sources can be used to drive the system clock (SYSCLK):

- HSI oscillator clock
- HSE oscillator clock
- LSI clock
- HSI oscillator clock divided by 6

Each clock source can be switched on or off independently when it is not used, to optimize system power consumption.

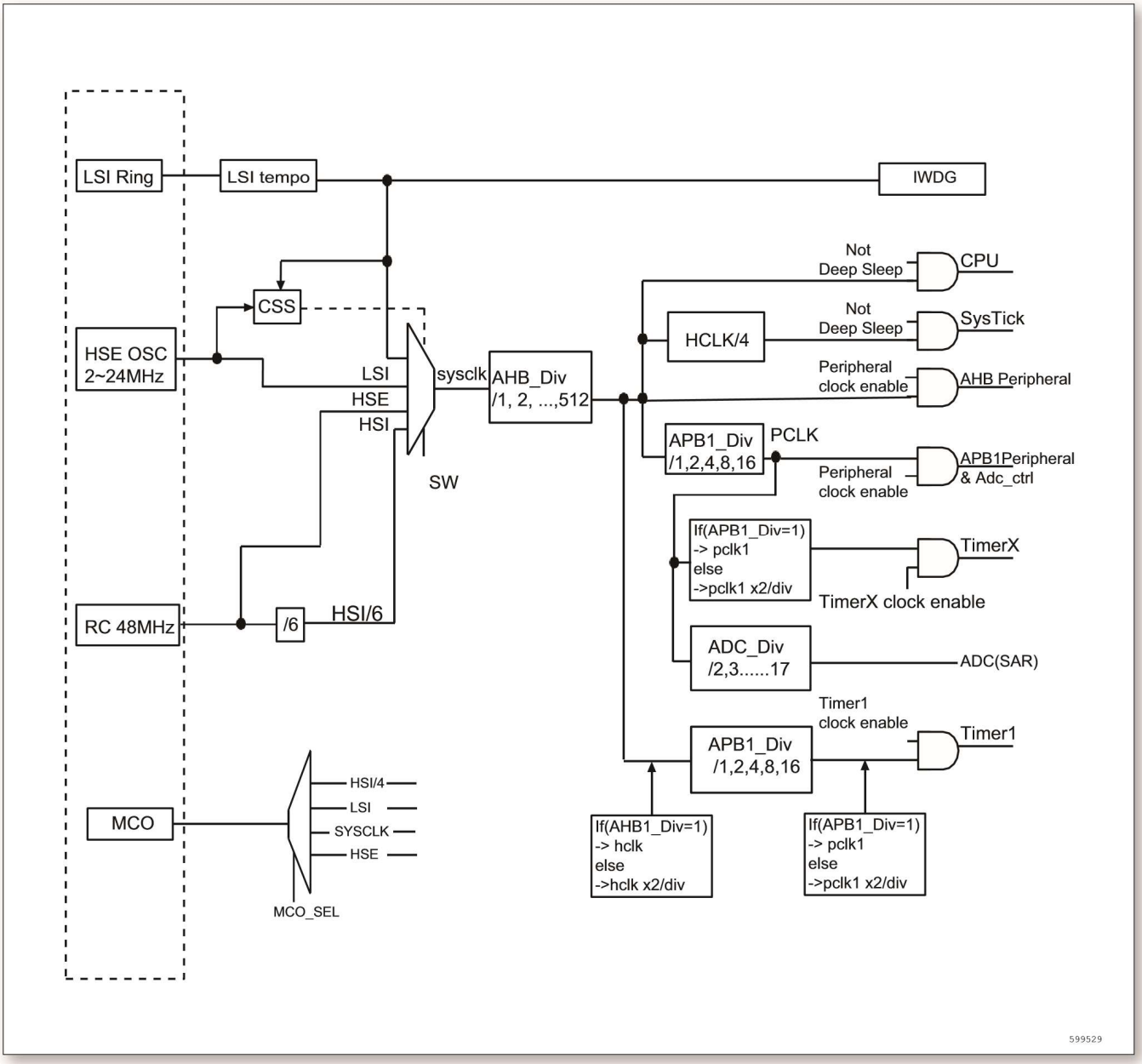


Figure 12. Clock tree

Several prescalers allow the configuration of the AHB frequency and the low speed APB (APB1) domains. The maximum frequency of the AHB and the APB1 domains is 48MHz.

The RCC feeds the CPU system timer (SysTick) external clock with the AHB clock divided by 4. The SysTick can work either with this clock or with the AHB clock, configurable in the SysTick control and status register. The ADCs are clocked by the APB1 clock divided.

The timer clock frequencies are automatically fixed by hardware. There are two cases:

1. if the APB prescaler coefficient is 1, the timer clock frequencies are set to the same frequency as that of the APB bus to which the timers are connected.
2. otherwise, they are set to twice (×2) the frequency of the APB bus to which the timers are connected.

FCLK acts as CPU free-running clock.

(RCC)

5.2.1 HSE clock

The high speed external clock signal (HSE) can be generated from two clock sources below:

- HSE external crystal/ceramic resonator
- HSE user external clock

The crystal/ceramic resonator and the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and startup stabilization time. The loading capacitance values must be adjusted according to the selected oscillator.

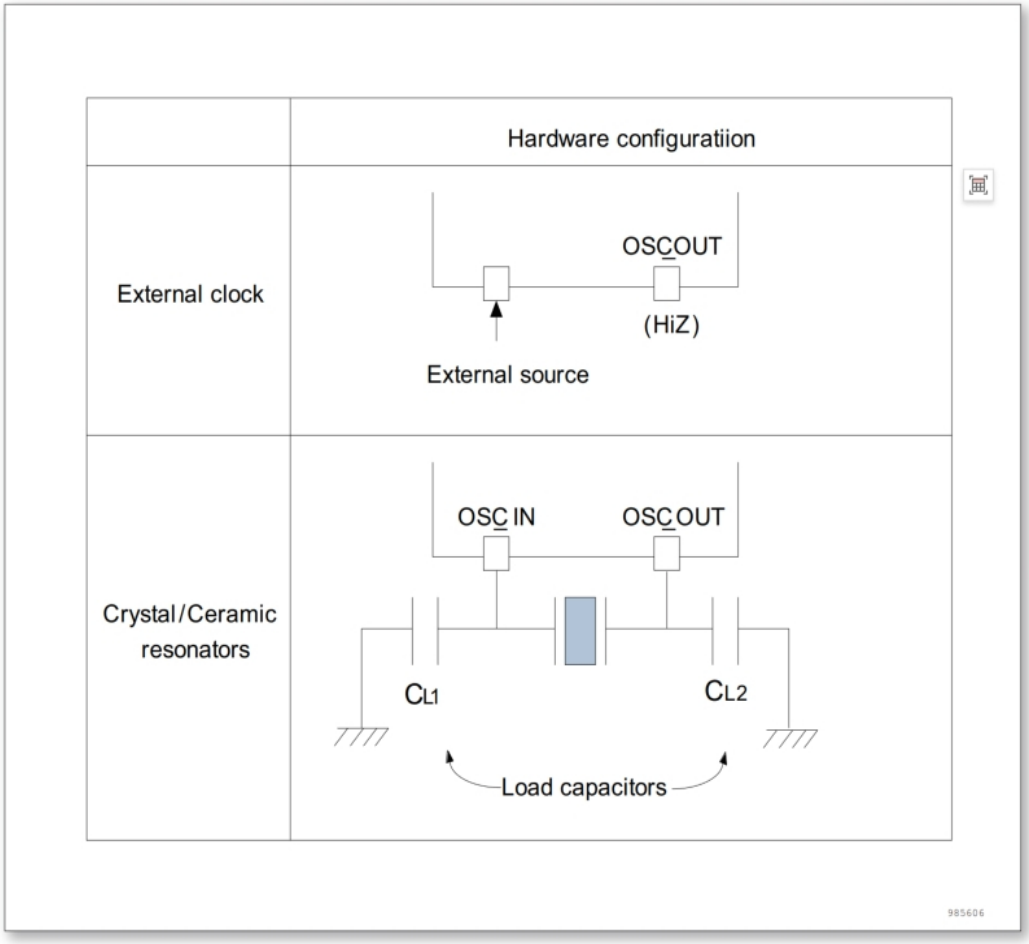


Figure 13. Clock sources

External source (HSE bypass)

In this mode, an external clock source must be provided. It can have a frequency of up to 24MHz. You select this mode by setting the HSEBYP and HSEON bits in the clock control register. The external clock signal (square or sinus with ~50% duty cycle) has to drive the OSC_IN pin while the OSC_OUT pin should not be connected. The PB0/PB1 must be set as an analog port.

External crystal/ceramic resonator (HSE crystal)

The external oscillator has the advantage of producing a very accurate rate on the main clock. The associated hardware configuration is shown in Figure 13. For more details, refer to the electrical characteristics of the datasheet.

The HSERDY flag in the clock control register (RCC_CR) indicates if the high-speed external oscillator is stable or not. At startup, the clock is not released until this bit is set to ‘1’ by hardware. An interrupt can be generated if enabled in the clock interrupt register (RCC_CIR).

The HSE crystal can be switched on and off using the HSEON bit in the clock control register (RCC_CR).

5.2.2 HSI clock

The HSI clock signal is generated from an internal HSI oscillator and can be used directly as a system clock. The HSI oscillator has the advantage of providing a system clock source without any external components. It also has a faster startup time than the HSE crystal oscillator however, even with calibration its frequency is less accurate.

Calibration

Oscillator frequencies of chips can vary because of manufacturing process variations, this is why the HSI clock frequency of each chip is factory calibrated for 1% accuracy at 25°C. After reset, the factory calibration value is loaded in the HSICAL flag of the clock control register.

The HSIRDY flag in the clock control register indicates if the HSI oscillator is stable or not. At clock startup, the HSI oscillator output clock is not released until this bit is set by hardware to ‘1’. The HSI oscillator can be switched on and off using the HSION bit in the clock control register.

The HSI clock can also be used as a backup source if the HSE crystal oscillator fails. Refer to section 5.2.5.

5.2.3 LSI clock

The LSI oscillator acts as an low-power clock source that can be kept running in Stop and Standby mode for the independent watchdog (IWDG) and auto-wakeup unit (AWU). The LSI clock frequency is around 40kHz. For more details, refer to the electrical characteristics section of the datasheets.

LSI oscillator can be switched on and off using the LSION bit in the control/status register (RCC_CSR). The LSIRDY flag in the control/status register (RCC_CSR) indicates if the low-speed internal oscillator is stable or not. At startup, the clock is not released until this bit is set to ‘1’ by hardware. A LSI interrupt request can be generated if enabled in the clock interrupt register (RCC_CIR).

5.2.4 System clock (SYSCLK) selection

After a system reset, the HSI oscillator divided by 6 is selected as system clock. When a clock source is used directly as system clock, it is not possible to stop it.

A switch from one clock source to another occurs only if the target clock source is ready (clock stable after startup delay). If a clock source which is not yet ready is selected, the system clock switch will not occur. A switch occurs until the target clock source is ready.

Status bits in the clock configuration register (RCC_CFGR) indicate which clock(s) is (are) ready and which clock is currently used as system clock.

5.2.5 Clock security system (CSS)

Clock security system can be activated by software. In this case, the clock detector is enabled after the HSE oscillator startup delay, and disabled when the HSE clock is stopped

If a failure is detected on the HSE clock, the HSE oscillator is automatically disabled. A clock failure event is sent to the break input of the advanced timer (TIM1) and a clock security interrupt CSS is generated, allowing the software to perform rescue operations. The CSS interrupt is linked to the

CPU NMI (Non-Maskable Interrupt) exception vector.

Note: Once the CSS is enabled and if the HSE clock fails, the CSS interrupt occurs and an NMI is automatically generated. The NMI will be executed indefinitely unless the CSS interrupt pending bit is cleared. As a consequence, in the NMI ISR user must clear the CSS interrupt by setting the CSSC bit in the clock interrupt register (RCC_CIR).

If the HSE oscillator is used directly or indirectly as the system clock, a detected failure causes a switch of the system clock to the HSI oscillator and the disabling of the HSE oscillator.

5.2.6 Watchdog clock

If the independent watchdog (IWDG) is started by either hardware option or software access, the LSI oscillator is forced ON and cannot be disabled. After the LSI oscillator temporization, the clock is provided to the IWDG.

5.2.7 Clock-out capability

The microcontroller allows the clock to be output onto the external MCO pin.

The registers of the corresponding GPIO port must be configured with related functions. One of 4 clock signals can be selected as the MCO clock.

- SYSCLK
- HSI/4
- HSE
- LSI

The clock selection is controlled by the MCO[2:0] bits of the clock configuration register (RCC_CFGR).

5.3 RCC register file and memory mapping description

Table 17. RCC register overview

Offset	Acronym	Register Name	Reset	Section
0x00	RCC_CR	Clock control register	0x0000XX01	Section 5.3.1
0x04	RCC_CFGR	Clock configuration register	0x00000000	Section 5.3.2
0x08	RCC_CIR	Clock interrupt register	0x00000000	Section 5.3.3
0x10	RCC_APB1RSTR	APB1 peripheral reset register	0x00000000	Section 5.3.4

Offset	Acronym	Register Name	Reset	Section
0x14	RCC_AHBENR	AHB peripheral clock enable register	0x00000014	Section 5.3.5
0x1C	RCC_APB1ENR	APB1 peripheral clock enable register	0x00000000	Section 5.3.6
0x24	RCC_CSR	Control status register	0x08000000	Section 5.3.7
0x28	RCC_AHBRSTR	AHB peripheral reset register	0x00000000	Section 5.3.8
0x40	RCC_SYSCFG	System configuration register	0x020F0003	Section 5.3.9

5.3.1 Clock control register (RCC_CR)

Address offset: 0x00

Reset value: 0x0000 XX01

Access: no wait state, word, half-word and byte access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CSSON	HSEBYP	HSERDY	HSEON
												rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													HSITEN	HSIRDY	HSION
													rw	r	rw

Bit	Field	Type	Reset	Description
31:20	Reserved			always read as 0.
19	CSSON	rw	0x00	CSSON: Clock security system enable Set to ‘1’ or cleared as ‘0’ by software to enable the clock detector. 0: clock detector OFF 1: clock detector ON if the external oscillator is ready
18	HSEBYP	rw	0x00	HSEBYP: External high-speed clock bypass Set to ‘1’ by software to bypass the external crystal oscillator. The HSEBYP bit can be written only if the external oscillator is disabled. 0: external oscillator not bypassed 1: external crystal oscillator bypassed
17	HSERDY	r	0x00	HSERDY: External high-speed clock ready flag is set to ‘1’ by hardware to indicate that the external clock is stable. 0: external clock not ready 1: external clock ready

Bit	Field	Type	Reset	Description
16	HSEON	rw	0x00	HSEON: External high-speed clock enable Set to '1' or cleared as '0' by software. Cleared as '0' by hardware to stop the external clock when entering Stop or Standby mode. This bit cannot be cleared as '0' if the external clock is used or selected as the system clock. 0: HSE oscillator OFF 1: HSE oscillator ON
15: 3	Reserved			always read as 0
2	HSITEN	rw	0x00	HSITEN: Set to '1' or cleared as '0' by software to enable the internal high-speed clock temperature calibration. 0: internal HSI clock not automatically calibrated with temperature 1: internal HSI clock automatically calibrated with temperature
1	HSIRDY	r	0x00	HSIRDY: Internal high-speed clock ready flag Set to '1' by hardware to indicate that internal clock is stable. After the HSION bit is cleared, HSIRDY goes low after 3 AHB clock cycles. 0: internal high-speed clock not ready 1: internal high-speed clock ready
0	HSION	rw	0x01	HSION: Internal high-speed clock enable Set to '1' or cleared as '0' by software. Set to '1' by hardware to force the internal oscillator ON when leaving Stop or Standby mode or in case of failure of the external clock used as system clock. This bit cannot be cleared as '0' if the internal high-speed clock is used directly or indirectly as system clock or is selected to become the system clock. 0: internal high-speed clock OFF 1: internal high-speed clock ON

5.3.2 Clock configuration register (RCC_CFGR)

Address offset: 0x04

Reset value: 0x0000 0000

Access: no wait state, word, half-word and byte access

1 or 2 wait states inserted only if the access occurs during clock source switch.

(RCC)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					PPRE1			HPRE				SWS		SW	
					rw	rw	rw	rw	rw	rw	rw	r	r	rw	rw

Bit	Field	Type	Reset	Description
31 : 27	Reserved			always read as 0
26 : 24	MCO	rw	0x00	MCO: Microcontroller clock output Set to ‘1’ or cleared as ‘0’ by software. 0xx: no clock output; 010: LSI clock output; 011: reserved 100: system clock (SYSCLK) output; 101: HSI clock divided by 4 output; 110: HSE clock output; 111: reserved Note: 1. This clock output may have some truncated cycles at startup or during MCO clock source switching. 2. When the system clock is selected to output to the MCO pin, make sure that this clock does not exceed 50MHz (the maximum IO frequency).
23:11	Reserved			always read as 0
10: 8	PPRE1	rw	0x00	PPRE1: APB1 prescale coefficient Set to ‘1’ or cleared as ‘0’ by software to control the prescale coefficient of the APB1 low-speed clock (PCLK1). 0xx: HCLK not divided 100: HCLK divided by 2 101: HCLK divided by 4 110: HCLK divided by 8 111: HCLK divided by 16

Bit	Field	Type	Reset	Description
7: 4	HPRE	rw	0x00	HPRE: AHB prescale coefficient Set to '1' or cleared as '0' by software to control the prescale coefficient of the AHB clock. 0xxx: SYSCLK not divided 1000: SYSCLK divided by 2 1001: SYSCLK divided by 4 1010: SYSCLK divided by 8 1011: SYSCLK divided by 16 1100: SYSCLK divided by 64 1101: SYSCLK divided by 128 1110: SYSCLK divided by 256 1111: SYSCLK divided by 512 Note: 1. The prefetch buffer must be kept on when using a prescale coefficient greater than 1 on the AHB clock. Refer to Reading the Flash memory section for more details.
3: 2	SWS	r	0x00	SWS: system clock switch status 00: HSI divided by 6 used as system clock; 01: HSE used as system clock; 10: HSI used as system clock; 11: LSI used as system clock;
1: 0	SW	rw	0x00	SW: system clock switch Set to '1' or cleared as '0' by software to select SYSCLK source. Set by hardware to force HSI divided by 6 to be the system clock when leaving Stop and Standby mode or in case of failure of the HSE used as system clock. 00: HSI divided by 6 used as system clock; 01: HSE used as system clock; 10: HSI used as system clock; 11: LSI used as system clock.

5.3.3 Clock interrupt register (RCC_CIR)

Address offset: 0x08
Reset value: 0x0000 0000
Access: no wait cycle, word, half-word and byte access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								CSSC	Reserved			HSE RDYC	HSI RDYC	Res.	LSI RDYC
								rc_w1				rc_w1	rc_w1		rc_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				HSE RDYIE	HSI RDYIE	Res.	LSI RDYIE	CSSF	Reserved			HSE RDYF	HSI RDYF	Res.	LSI RDYF
				rw	rw		rw	r				r	r		r

Bit	Field	Type	Reset	Description
31:24	Reserved			always read as 0.
23	CSSC	rc_w1	0x00	CSSC: clock security system interrupt clear Set to '1' by software to clear the CSSF security system interrupt flag. 0: no action 1: clear CSSF security system interrupt flag
22:20	Reserved			always read as 0
19	HSERDYC	rc_w1	0x00	HSERDYC: HSE ready interrupt clear Set to '1' by software to clear the HSE ready interrupt flag HSERDYF. 0: no action 1: clear HSE ready interrupt flag HSERDYF
18	HSIRDYC	rc_w1	0x00	HSIRDYC: HSI ready interrupt clear Set to '1' by software to clear the HSI ready interrupt flag HSIRDYF. 0: no action 1: clear HSI ready interrupt flag HSIRDYF
17	Reserved			always read as 0
16	LSIRDYC	rc_w1	0x00	LSIRDYC: LSI ready interrupt clear Set to '1' by software to clear the LSI ready interrupt flag LSIRDYF. 0: no action 1: clear LSI ready interrupt flag LSIRDYF
15: 12	Reserved			always read as 0
11	HSERDYIE	rw	0x00	HSERDYIE:HSE ready interrupt enable Set to '1' or cleared as '0' by software to enable/disable interrupt caused by the external oscillator stabilization. 0: HSE ready interrupt disabled 1: HSE ready interrupt enabled
10	HSIRDYIE	rw	0x00	HSIRDYIE: HSI ready interrupt enable Set to '1' or cleared as '0' by software to enable/disable interrupt caused by the internal oscillator stabilization. 0: HSI ready interrupt disabled 1: HSI ready interrupt enabled

Bit	Field	Type	Reset	Description
9	Reserved			always read as 0
8	LSIRDYIE	rw	0x00	LSIRDYIE: LSI ready interrupt enable Set to '1' or cleared as '0' by software to enable/disable interrupt caused by the internal 40KHz oscillator stabilization. 0: LSI ready interrupt disabled 1: LSI ready interrupt enabled
7	CSSF	r	0x00	CSSF: clock security system interrupt flag) Set to '1' by hardware when a failure is detected in the external oscillator. It is cleared by software setting the CSSC bit to '1'. 0: no clock security interrupt caused by HSE clock failure 1: clock security interrupt caused by HSE clock failure
6: 4	Reserved			always read as 0
3	HSERDYF	r	0x00	HSERDYF: HSE ready interrupt flag Set to '1' by hardware when external low speed clock becomes stable. It is cleared by software setting the HSERDYC bit to '1'. 0: no clock ready interrupt caused by the external oscillator 1: clock ready interrupt caused by the external oscillator
2	HSIRDYF	r	0x00	HSIRDYF: HSI ready interrupt flag Set to '1' by hardware when internal high speed clock becomes stable. It is cleared by software setting the HSIRDYC bit to '1'. 0: no clock ready interrupt caused by the internal HSI oscillator 1: clock ready interrupt caused by the internal HSI oscillator
1	Reserved			always read as 0
0	LSIRDYF	r	0x00	LSIRDYF: LSI ready interrupt flag Set to '1' by hardware when internal low speed clock becomes stable. It is cleared by software setting the LSIRDYC bit to '1'. 0: no clock ready interrupt caused by the internal 40KHz oscillator; 1: clock ready interrupt caused by the internal 40KHz oscillator.

5.3.4 APB1 peripheral reset register (RCC_APB1RSTR)

Address offset: 0x10

Reset value: 0x0000 0000

Access: no wait cycle, word, half-word and byte access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	SYSCFG	DBGMCU	PWR	Reserved						I2C1	Reserved			UART2	UART1
	rw	rw	rw							rw				rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			SPI1	WWDG	Res.	ADC1	Reserved					TIM14	TIM1	TIM3	Res.
			rw	rw		rw						rw	rw	rw	

Bit	Field	Type	Reset	Description
31	Reserved			always read as 0
30	SYSCFG	rw	0x00	SYSCFG: system configuration register reset Set to '1' or cleared as '0' by software. 0: no action 1: reset SYSCFG
29	DBGMCU	rw	0x00	DBGMCU: system configuration register reset (DBGMCU reset) Set to '1' or cleared as '0' by software. 0: no action 1: reset DBGMCU
28	PWR	rw	0x00	PWR: power interface reset) Set to '1' or cleared as '0' by software. 0: no action 1: reset power interface
27:22	Reserved			always read as 0
21	I2C1	rw	0x00	I2C1: I2C1 reset Set to '1' or cleared as '0' by software. 0: no action 1: reset I2C1
20:18	Reserved			always read as 0
17	UART2	rw	0x00	UART2: UART2 reset Set to '1' or cleared as '0' by software. 0: no action 1: reset UART2

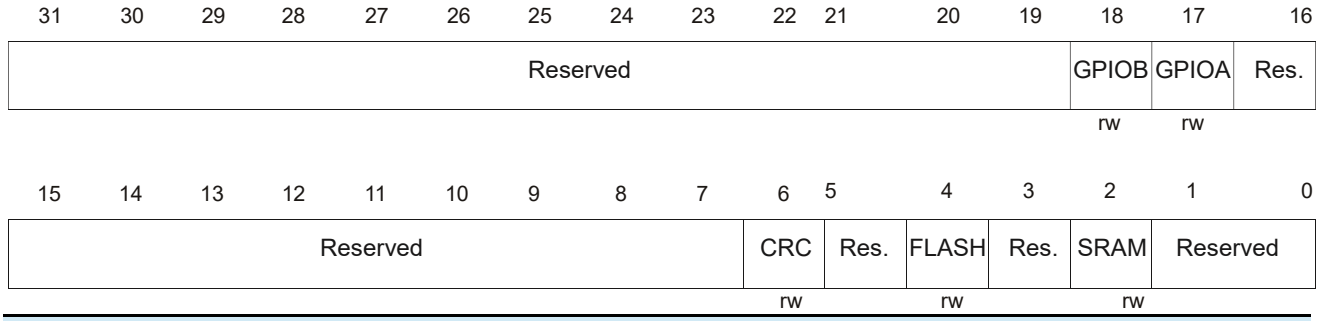
Bit	Field	Type	Reset	Description
16	UART1	rw	0x00	UART1: UART1 reset Set to '1' or cleared as '0' by software. 0: no action 1: reset UART1
15:13	Reserved			always read as 0
12	SPI1	rw	0x00	SPI1: SPI1 reset Set to '1' or cleared as '0' by software. 0: no action 1: reset SPI1
11	WWDG	rw	0x00	WWDG: window watchdog reset) Set to '1' or cleared as '0' by software. 0: no action 1: reset window watchdog
10	Reserved			always read as 0
9	ADC1	rw	0x00	ADC1: ADC1 interface reset Set to '1' or cleared as '0' by software. 0: no action 1: reset ADC1 interface
8:4	Reserved			always read as 0
3	TIM14	rw	0x00	TIM14: Timer 14 reset Set to '1' or cleared as '0' by software. 0: no action 1: reset TIM14 timer
2	TIM1	rw	0x00	TIM1: Timer 1 reset Set to '1' or cleared as '0' by software. 0: no action 1: reset TIM1 timer
1	TIM3	rw	0x00	TIM3: Timer 3 reset Set to '1' or cleared as '0' by software. 0: no action 1: reset TIM3 timer
0	Reserved			always read as 0

5.3.5 AHB peripheral clock enable register (RCC_AHBENR)

Address offset: 0x14

Reset value: 0x0000 0014

Access: no wait cycle, word, half-word and byte access



Bit	Field	Type	Reset	Description
31:19	Reserved			always read as 0.
18	GPIOB	rw	0x00	GPIOB: GPIOB clock enable 0: GPIOB clock disabled 1: GPIOB clock enabled
17	GPIOA	rw	0x00	GPIOA: GPIOA clock enable 0: GPIOA clock disabled 1: GPIOA clock enabled
16:7	Reserved			always read as 0
6	CRC	rw	0x00	CRC: CRC clock enable Set to ‘1’ or cleared as ‘0’ by software. 0: CRC clock disabled 1: CRC clock enabled
5	Reserved			always read as 0

Bit	Field	Type	Reset	Description
4	FLASH	rw	0x01	FLASH: FLASH clock enable 0: FLASH clock disabled 1: FLASH clock enabled
3	Reserved			always read as 0
2	SRAM	rw	0x01	SRAM: SRAM interface clock enable 0: SRAM clock disabled 1: SRAM clock enabled
0:1	Reserved			always read as 0

5.3.6 APB1 peripheral clock enable register (RCC_APB1ENR)

Address offset: 0x1C

Reset value: 0x0000 0000

Access: no wait cycle, word, half-word and byte access
Note: if the peripheral clock is OFF, the software cannot read the value of the peripheral register.

31	30	29	28	27	26 25	24	23	22	21	20	19	18	17	16
Res.	SYSCFG	DBGMCU	PWR	Reserved					I2C1	Reserved			UART2	UART1
	rw	rw	rw						rw				rw	rw

15	14	13	12	11	10 9	8	7	6	5	4	3	2	1	0
Reserved			SPI1	WWDG	Res.	ADC1	Reserved				TIM14	TIM1	TIM3	Res.
			rw	rw		rw					rw	rw	rw	

Bit	Field	Type	Reset	Description
31	Reserved			always read as 0
30	SYSCFG	rw	0x00	SYSCFG: system configuration register clock enable Set to ‘1’ or cleared as ‘0’ by software. 0: no action 1: reset SYSCFG
29	DBGMCU	rw	0x00	DBGMCU: DBGMCU enable Set to ‘1’ or cleared as ‘0’ by software. 0: no action 1: reset DBGMCU
28	PWR	rw	0x00	PWR: power interface clock enable Set to ‘1’ or cleared as ‘0’ by software. 0: no action 1: reset power interface

Bit	Field	Type	Reset	Description
27:22	Reserved			always read as 0
21	I2C1	rw	0x00	I2C1: I2C1 clock enable Set to '1' or cleared as '0' by software. 0: no action 1: reset I2C1
20:18	Reserved			always read as 0
17	UART2	rw	0x00	UART2: UART2 clock enable Set to '1' or cleared as '0' by software. 0: no action 1: reset UART2
16	UART1	rw	0x00	UART1: UART1 clock enable Set to '1' or cleared as '0' by software. 0: no action 1: reset UART1
15:13	Reserved			always read as 0
12	SPI1	rw	0x00	SPI1: SPI1 clock enable Set to '1' or cleared as '0' by software. 0: no action 1: reset SPI1
11	WWDG	rw	0x00	WWDG: window watchdog clock reset Set to '1' or cleared as '0' by software. 0: no action 1: reset window watchdog
10	Reserved			always read as 0
9	ADC1	rw	0x00	ADC1: ADC1 interface clock enable Set to '1' or cleared as '0' by software. 0: no action 1: reset ADC1 interface
8:4	Reserved			always read as 0
3	TIM14	rw	0x00	TIM14: Timer 14 clock enable Set to '1' or cleared as '0' by software. 0: no action 1: reset TIM14 timer
2	TIM1	rw	0x00	TIM1: Timer 1 clock enable Set to '1' or cleared as '0' by software. 0: no action 1: reset TIM1 timer

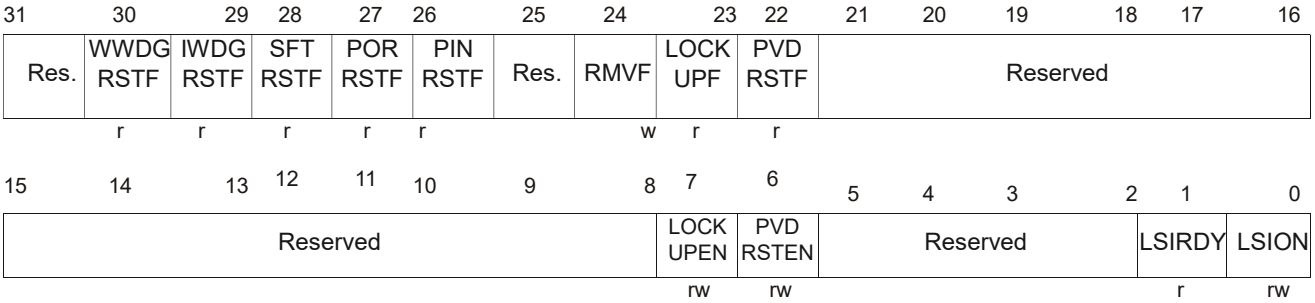
Bit	Field	Type	Reset	Description
1	TIM3	rw	0x00	TIM3: Timer 3 clock enable Set to ‘1’ or cleared as ‘0’ by software. 0: no action 1: reset TIM3 timer
0	Reserved			always read as 0

5.3.7 Control status register (RCC_CSR)

Address offset: 0x24

Reset value: 0x0800 0000

Access: 0 to 3 wait cycles, word, half-word and byte access
Wait states are inserted in the case of successive accesses to this register.



Bit	Field	Type	Reset	Description
31	Reserved			always read as 0
30	WWDGRSTF	r	0x00	WDGRSTF: window watchdog reset flag Set to ‘1’ by hardware when a window watchdog reset occurs. It is only cleared by resetting the power or writing to the RMVF bit via software. 0: no window watchdog reset occurred 1: window watchdog reset occurred
29	IWDGRSTF	r	0x00	IWDGRSTF: independent watchdog reset flag Set to ‘1’ by hardware when an independent watchdog reset from VDD domain occurs. It is only cleared by resetting the power or writing to the RMVF bit via software. 0: no independent watchdog reset occurred 1: independent watchdog reset occurred
28	SFTRSTF	r	0x00	SFTRSTF: software reset flag Set to ‘1’ by hardware when a software reset occurs. It is only cleared by resetting the power or writing to the RMVF bit via software. 0: no software reset occurred 1: software reset occurred

Bit	Field	Type	Reset	Description
27	PORRSTF	r	0x01	PORRSTF: POR/PDR reset flag Set to '1' by hardware when a a POR/PDR reset occurs. It is only cleared by resetting the power or writing to the RMVF bit via software. 0: no POR/PDR reset occurred 1: POR/PDR reset occurred
26	PINRSTF	r	0x00	PINRSTF: nRST PIN reset flag (PIN reset flag) Set to '1' by hardware when a reset from the nRST pin occurs. It is only cleared by resetting the power or writing to the RMVF bit via software. 0: no reset from nRST pin occurred 1: reset from nRST pin occurred
25	Reserved			always read as 0
24	RMVF	w	0x00	RMVF: remove reset flag Set to '1' by software to clear the reset flags. 0: no action 1: clear the reset flags
23	LOCKUPF	r	0x00	LOCKUPF: CPU lockup reset flag Set to '1' by hardware when a reset from the CPU lockup occurs. It is only cleared by resetting the power or writing to the RMVF bit via software. 0: no reset from the CPU deadlock occurred 1: reset from the CPU deadlock occurred
22	PVDRSTF	r	0x00	PVDRSTF: PVD reset flag Set to '1' by hardware when a PVD reset occurs. It is only cleared by resetting the power or writing to the RMVF bit via software. 0: no PVD reset occurred 1: PVD reset occurred
21:8	Reserved			always read as 0
7	LOCKUPEN	rw	0x00	LOCKUPEN: CPU lockup reset enable 0: CPU lockup reset disabled 1: CPU lockup reset enabled
6	PVDRSTEN	rw	0x00	PVDRSTEN: PVD reset enable 0: PVD reset disabled 1: PVD reset enabled
5:2	Reserved			always read as 0
1	LSIRDY	r	0x00	LSIRDY: internal low-speed oscillator ready Set to '1' or cleared as '0' by software to indicate when the internal RC 40KHz oscillator is stable. After the LSION bit is cleared, LSIRDY goes low after 3 AHB clock cycles. 0: internal 40KHz oscillator not ready 1: internal 40 KHz oscillator ready

Bit	Field	Type	Reset	Description	Bit
0	LSION	rw	0x00	LSION: internal low-speed oscillator enable) Set to ‘1’ or cleared as ‘0’ by software, or cleared by resetting the power. 0: internal 40 KHz oscillator OFF 1: internal 40 KHz oscillator ON	

5.3.8 AHB peripheral reset register (RCC_AHBRSTR)

Address offset: 0x28

Reset value: 0x0000 0000

Access: 0 to 3 wait cycles, word, half-word and byte access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													GPIOB	GPIOA	Res.
													rw	rw	

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

Bit	Field	Type	Reset	Description
31: 19	Reserved			always read as 0
18	GPIOB	rw	0x00	GPIOB: GPIOB reset Set to ‘1’ or cleared as ‘0’ by software. 0: no action 1: reset GPIOB
17	GPIOA	rw	0x00	GPIOA: GPIOA reset Set to ‘1’ or cleared as ‘0’ by software. 0: no action 1: reset GPIOA
16: 0	Reserved			always read as 0

5.3.9 System configuration register (RCC_SYSCFG)

Address offset: 0x40

Reset value: 0x020F 0003

Access: 0 to 3 wait cycles, word, half-word and byte access

(RCC)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OSC_LPF_EN	Reserved											SFT_NRST_RMP	SECTOR_1K_CFG	PROG_CHECK_EN
	rw												rw	rw	rw

Bit	Field	Type	Reset	Description
31: 15	Reserved			Reserved.
14	OSC_LPF_EN	rw	0x00	OSC_LPF_EN: external crystal oscillator low pass filter enable 0: disabled 1: enabled
3: 13	Reserved			Reserved.
2	SFT_NRST_RMP	rw	0x00	SFT_NRST_RMP: software mapping of nRST. This bit is only cleared when the power is reset or when it is set to '0' by software. 1: PA14 mapped to nRST 0: no action Note: The SFT_NRST_RMP bit of RCC_SYSCFG is set to '1', PA14 is mapped to the external reset of nRST, and the low level is at least 4us.
1	SECTOR_1K_CFG	rw	0x01	SECTOR_1K_CFG: size for Flash erase. 1: 1K bytes 0: 512 bytes
0	PROG_CHECK_EN	rw	0x01	PROG_CHECK_EN: check whether the data in the Flash is FF while writing the Flash.(hardware is fixed to 1) 1: check 0: not to check

6 General-Purpose I/Os (GPIOs)

General-Purpose I/Os (GPIOs)

6.1 GPIO functional description

Each port bit of the general-purpose IO (GPIO) ports, can be individually configured by software in several modes.

- Input floating
- Input pull-up
- Input pull-down
- Analog input
- Output open-drain
- Output push-pull
- Alternate function push-pull
- Alternate function open-drain

Each I/O port bit is freely programmable, however the I/O port registers have to be accessed as 32-bit words (half-word or byte accesses are not allowed).

The purpose of the GPIOx_BSRR and GPIOx_BRR registers is to allow atomic read/modify accesses to any of the GPIO registers. This way, there is no risk that an IRQ occurs between the read and the modify access.

The figure below shows the basic structure of an I/O port bit.

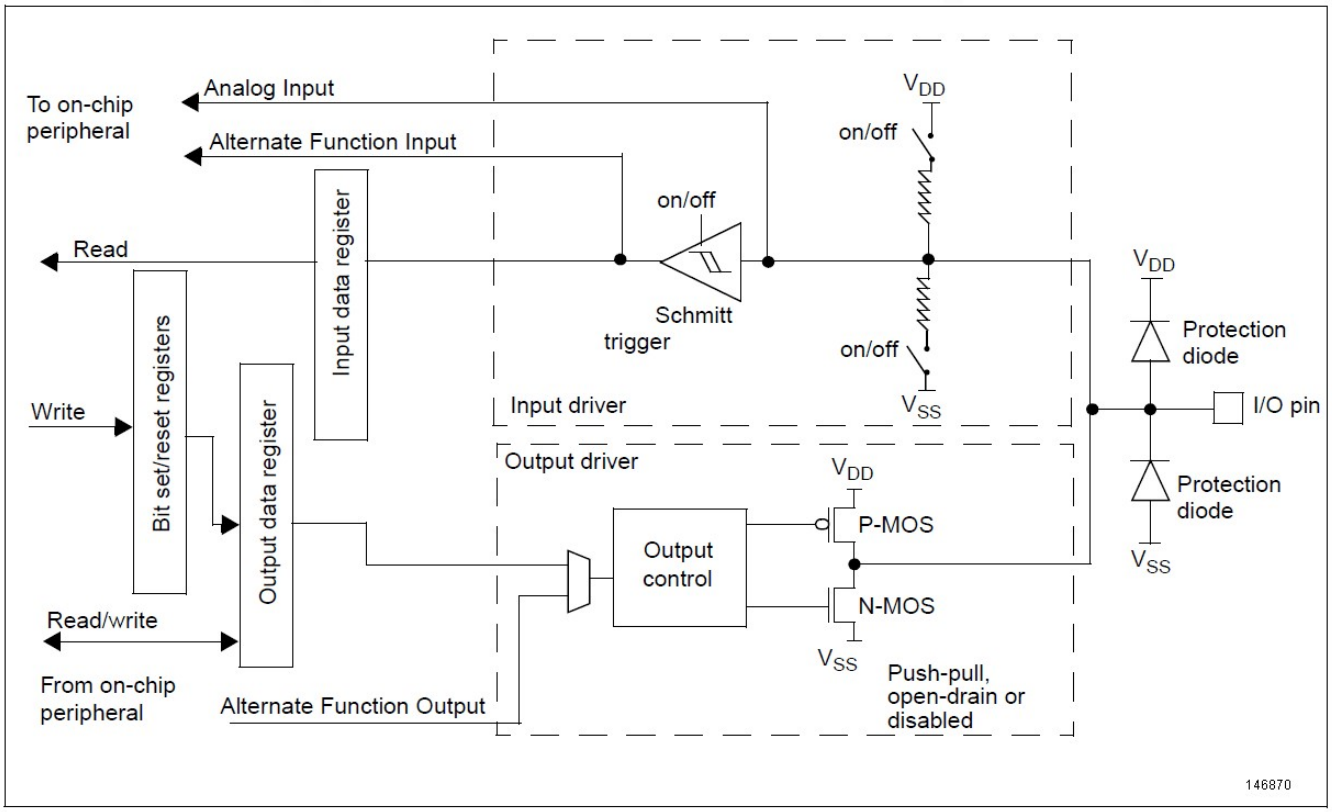


Figure 14. Basic structure of a standard I/O port bit

Table 18. Port bit configuration table

Configuration mode		CNF1	CNF0	MODE1	MODE0	PxODR register
General purpose output	Push-pull	0	0	01		0 or 1
	Open-drain		1			0 or 1
Alternate function output	Push-pull	1	0			not used
	Open-drain		1			not used
Input	Analog input	0	0	00		not used
	Input floating		1			not used
	Input pull-down	1	0			0
	Input pull-up					1

Table 19. Output MODE bits

MODE[1:0]	Meaning
00	Input
01	Output

6.1.1 General-purpose I/O (GPIO)

During and just after reset, the alternate functions are not active and the I/O ports are configured in Input Floating mode (CNFx[1:0] = 01, MODEx[1:0] = 00).

The SWD pins are in input pull-up (Pu)/pull-down (PD) after reset:

- PA14: SWCLK in PD
- PA13: SWDIO in PU

All GPIO ports except debug ports are in the floating state after reset.

When configured as output, the value written to the output data register (GPIOx_ODR) is output on the I/O pin. It is possible to use the output driver in Push-Pull mode or Open-Drain mode (only the N-MOS is activated when outputting 0).

All GPIO pins have an internal weak pull-up and weak pull-down that can be activated or not when configured as input.

6.1.2 Atomic bit set or reset

There is no need for the software to disable interrupts when programming the GPIOx_ODR at bit level.

This is achieved by programming to '1' the Bit Set/Reset register (GPIOx_BSRR, or for reset only GPIOx_BRR) to select the bits to modify.

The unselected bits will not be modified.

6.1.3 External interrupt/wakeup lines

All ports have external interrupt capability. To use external interrupt lines, the port must be configured in input mode. For more information on external interrupts, refer to Section 6.3: External interrupt/event controller (EXTI).

6.1.4 Alternate functions (AF)

It is necessary to program the Port Bit Configuration register before using a default alternate function.

- For alternate function inputs, the port must be configured in Input mode (floating, pull-up or pull-down) and the input pin must be driven externally.

Note: It is also possible to emulate the AF input pin by software by programming the GPIO controller. In this case, the port should be configured in Alternate Function Output mode. And obviously, the corresponding port should not be driven externally as it will be driven by the software using the GPIO controller.

- For alternate function outputs, the port must be configured in Alternate Function Output mode (Push-Pull or Open-Drain).
- For bidirectional Alternate Functions, the port bit must be configured in Alternate Function Output mode (Push-Pull or Open-Drain). In this case the input driver is configured in input floating mode.

If a port bit is configured as Alternate Function Output, this disconnects the output register and connects the pin to the output signal of an on-chip peripheral. If software configures a GPIO pin as Alternate Function Output, but peripheral is not activated, its output is not specified.

6.1.5 Software remapping of I/O alternate functions

In that case, the alternate functions are no longer mapped to their original assignations.

6.1.6 GPIO locking mechanism

The locking mechanism allows the IO configuration to be frozen. When the LOCK sequence has been applied on a port bit, it is no longer possible to modify the value of the port bit until the next reset.

6.1.7 Input configuration

When the I/O port is programmed as Input:

- The Output Buffer is disabled
- The Schmitt Trigger Input is activated
- The weak pull-up and pull-down resistors are activated or not depending on input configuration (pull-up, pull-down or floating)
- A read access to the Input Data register obtains the I/O State

The figure below shows the input configuration of the I/O port bit.

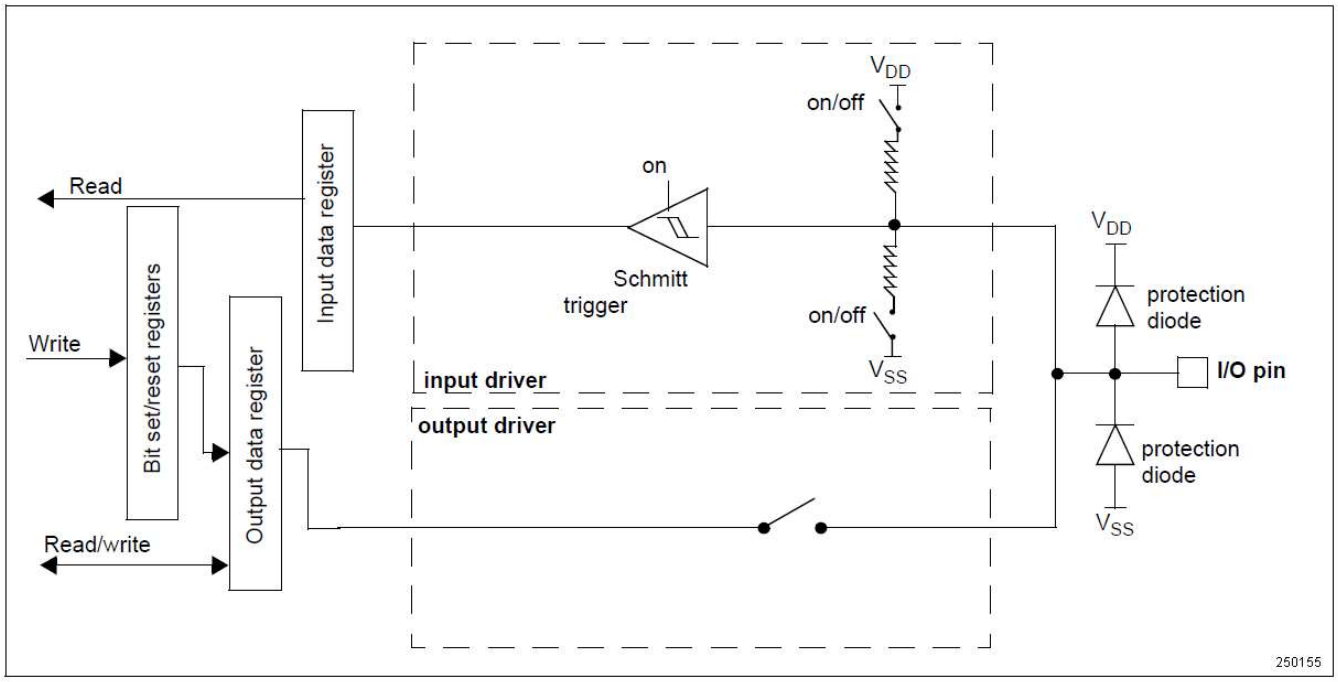


Figure 15. Input floating/pull up/pull down configurations

6.1.8 Output configuration

When the I/O port is configured as output:

- The Output Buffer is enabled
  - Open Drain Mode: A “0” in the Output register activates the N-MOS while a “1” in the Output register leaves the port in Hi-Z (the P-MOS is never activated)
  - Push-Pull Mode: A “0” in the Output register activates the N-MOS while a “1” in the Output register activates the P-MOS

- The Schmitt Trigger Input is activated
- The weak pull-up and pull-down resistors are disabled.
- A read access to the Input Data register gets the I/O state in open drain mode
- A read access to the Output Data register gets the last written value in push-pull mode

The figure below shows the output configuration of the I/O port bit:

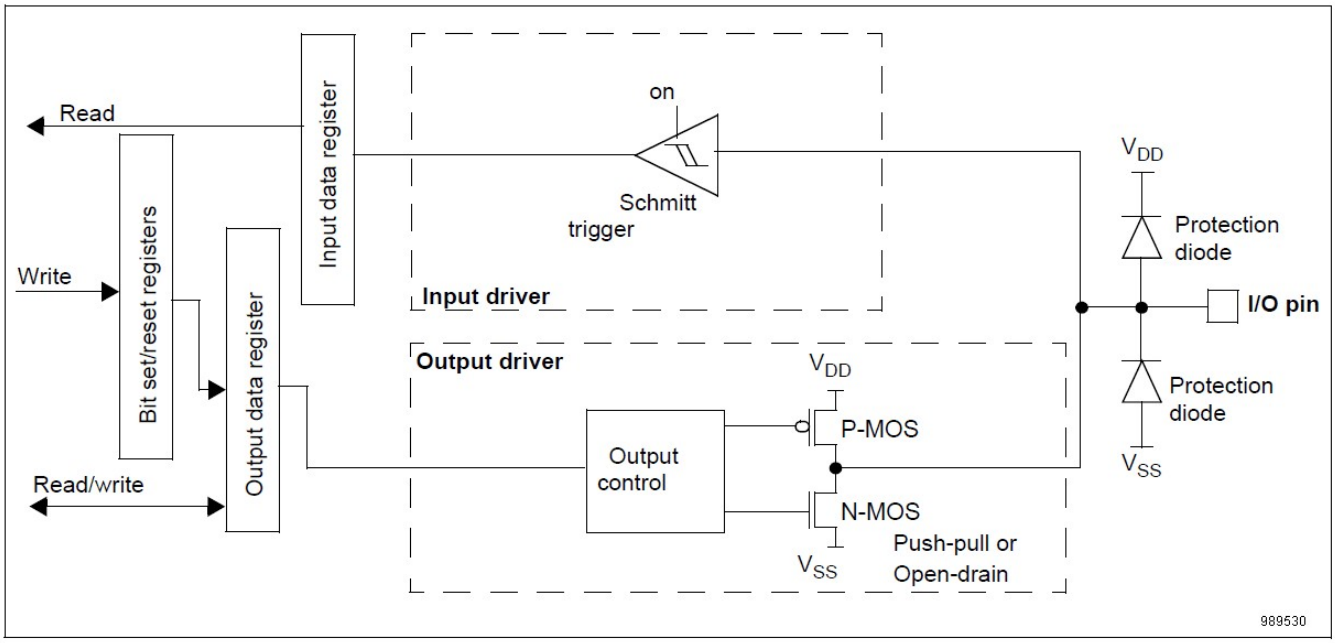


Figure 16. Output configuration

6.1.9 Alternate function configuration

When the I/O port is configured as alternate function:

- The Output Buffer is turned on in Open Drain or Push-Pull configuration
- The Output Buffer is driven by the signal coming from the peripheral (alternate function output)
- The Schmitt Trigger Input is activated
- The weak pull-up and pull-down resistors are disabled
- A read access to the Input Data register gets the I/O state in open drain mode
- A read access to the Output Data register gets the last written value in push-pull mode

The figure below shows the alternate function configuration of the I/O port bit. Also, refer to AFIO registers for further information.

A set of Alternate Function I/O registers allows the user to remap some alternate functions to different pins.

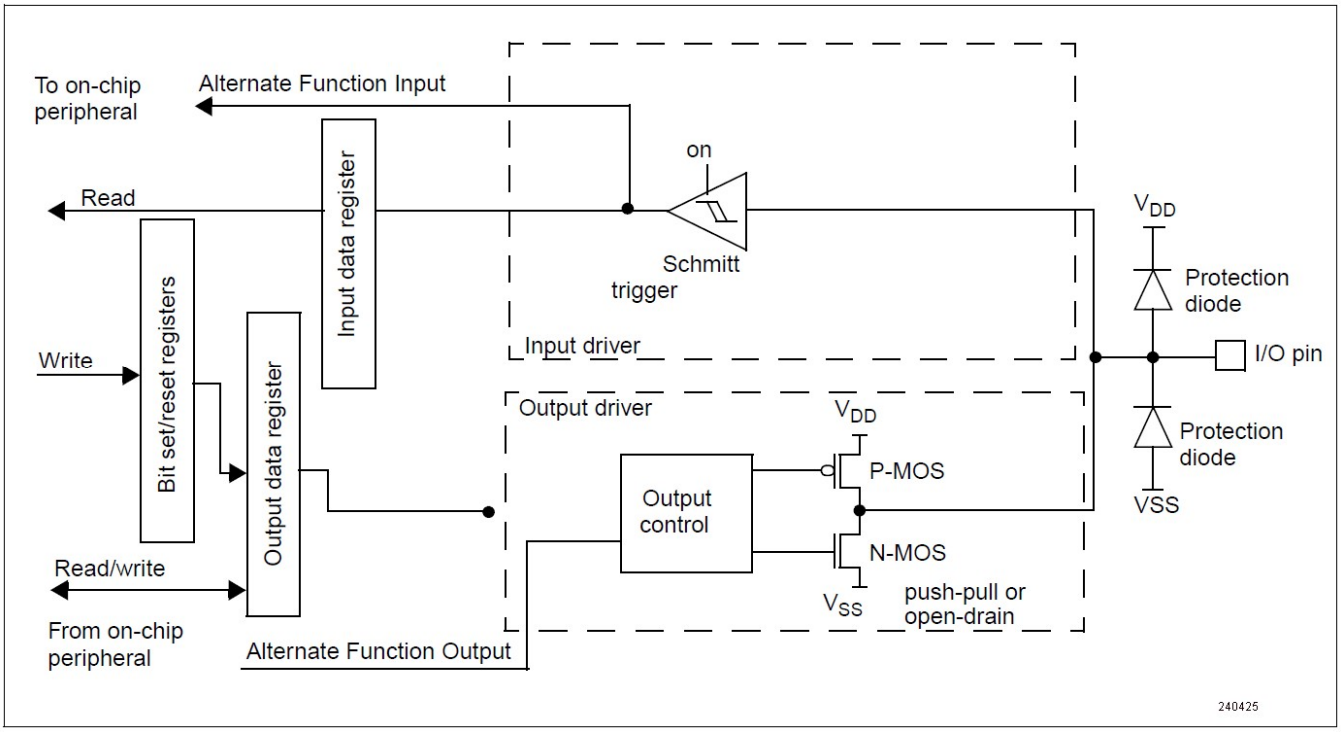


Figure 17. Alternate function configuration

6.1.10 Analog input configuration

When the I/O port is configured as analog input configuration:

- The Output Buffer is disabled.
- The Schmitt Trigger Input is de-activated providing zero consumption for every analog value of the I/O pin. The output of the Schmitt Trigger is forced to ‘0’.
- The weak pull-up and pull-down resistors are disabled.
- A read access to the Input Data register gets the value ‘0’.

The figure below shows the high impedance input configuration of the I/O port bit.

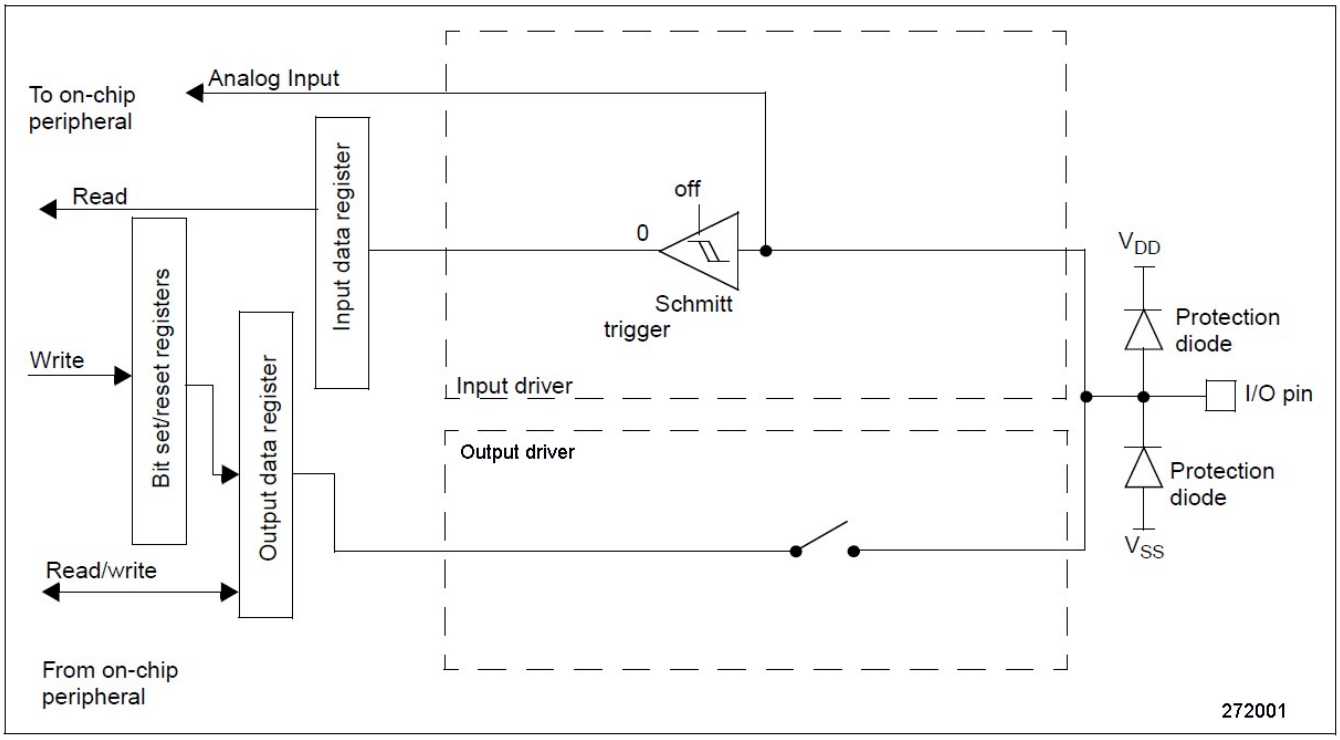


Figure 18. High impedance-analog input configuration

6.1.11 GPIO configurations for device peripherals

The following tables give the GPIO configurations of the device peripherals:

Table 20. Advanced timers TIM1

TIM1 pin	Configuration	GPIO configuration
TIM1_CHx	Input capture channel x	Input floating
	Output compare channel x	Alternate function output push-pull
TIM1_CHxN	Complementary output channel x	Alternate function output push-pull
TIM1_BKIN	Break input	Input floating

Table 21. General-purpose timers TIM3/14

TIM3/14 pin	Configuration	GPIO configuration
TIM3/14_CHx	Input capture channel x	Input floating
	Output compare channel x	Alternate function output push-pull

Table 22. UARTs

UART pin	Configuration	GPIO configuration
UARTx_TX	Data sent via serial port	Alternate function output push-pull
UARTx_RX	Data received via serial port	Input floating/Input pull-up

Table 23. SPI

SPI pin	Configuration	GPIO configuration
SPIx_SCK	Master mode	Alternate function output push-pull
	Slave mode	Input floating
SPIx_MOSI	Full duplex/master	Alternate function output push-pull
	Full duplex/slave	Input floating/Input pull-up
SPIx_MISO	Full duplex/master	Input floating/Input pull-up
	Full duplex/slave	Alternate function output push-pull
SPIx_NSS	Hardware master/slave	Input floating/Input pull-up/Input pull-down
	Hardware master/NSS output enabled	Alternate function output push-pull
	Software mode	Not used. Can be used as a GPIO

Table 24. I2C

I2C pin	Configuration	GPIO configuration
I2Cx_SCL	I2C clock	Alternate function output open drain
I2Cx_SDA	I2C data	Alternate function output open drain

Table 25. ADC

ADC pin	GPIO configuration
ADC	Analog input

Table 26. Other IOs

Pins	Configuration	GPIO configuration
MCO	Clock output	Alternate function output push-pull
External input lines	External interrupt input	Input floating/Input pull-up/Input pull-down

6.2. Alternate function I/O and debug configuration

To optimize the number of peripherals available, it is possible to remap some alternate functions to some other pins. It is achieved by setting the alternate function register (AFR). In this case, the alternate functions are no longer mapped to their original assignations.

6.2.1 OSC_IN/OSC_OUT

The external oscillator pins OSC_IN/OSC_OUT can be used as general-purpose I/O PD0/PD1 by turning off OSC enable bit first and then setting the alternate function register (AFR).

PB0/PB1 must be set as an analog port when used as OSC_IN/OSC_OUT.

Note: The external interrupt/event function is not remapped.

6.2.2 SWD alternate function remapping

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

Table 27. Debug interface signals

Alternate function	GPIO port
SWDIO	PA13
SWCLK	PA14

To optimize the number of free GPIOs during debugging, this mapping can be configured in different ways by setting the alternate function remapping and alternate function register.

6.3 GPIO register description

Table 28. Overview of GPIO registers

Offset	Acronym	Register Name	Reset	Section
0x00	GPIOx_CRL	Port configuration register low	0x44444444	Section 6.3.1
0x04	GPIOx_CRH	Port configuration register high	0x44444444	Section 6.3.2
0x08	GPIOx_IDR	Port input data register	0x0000XXXX	Section 6.3.3
0x0C	GPIOx_ODR	Port output data register	0x00000000	Section 6.3.4
0x10	GPIOx_BSRR	Port bit set/reset register	0x00000000	Section 6.3.5
0x14	GPIOx_BRR	Port bit reset register	0x00000000	Section 6.3.6
0x18	GPIOx_LCKR	Port configuration lock register	0x00000000	Section 6.3.7
0x1C	GPIOx_DCR	Port output open drain control register	0x00000000	Section 6.3.8
0x20	GPIOx_AFRL	Port alternate function register low	0xFFFFFFFF	Section 6.3.9
0x24	GPIOx_AFRH	Port alternate function register high	0xFFFFFFFF	Section 6.3.10

6.3.1 Port configuration register low (GPIOx_CRL) (x = A..B)

Address offset: 0x00
Reset value: 0x4444 4444

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16
CNF7	MODE7	CNF6	MODE6	CNF5	MODE5	CNF4	MODE4
rw	rw	rw	rw	rw	rw	rw	rw

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
CNF3	MODE3	CNF2	MODE2	CNF1	MODE1	CNF0	MODE0
rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31: 30 27: 26 23: 22 19: 18 15: 14 11: 10 7: 6 3: 2	CNFy	rw	0x01	Port x configuration bits (0…7) These bits are written by software to configure the corresponding I/O port. Refer to Table 18: Port bit configuration table. In input mode (MODE = 00): 00: Analog input mode 01: Input floating mode 10: Input pull-up/pull-down mode 11: Reserved In output mode (MODE > 00): 00: General purpose output push-pull 01: General purpose output open-drain 10: Alternate function output push-pull 11: Alternate function output open-drain
29: 28 25: 24 21: 20 17: 16 13: 12 9: 8 5: 4 1: 0	MODEy	rw	0x00	Port x mode bits (y = 0…7) These bits are written by software to configure the corresponding I/O port. Refer to Table 18: Port bit configuration table. 00: Input mode (reset state) 01: Output mode 10: Reserved 11: Reserved

6.3.2 Port configuration register high (GPIOx_CRH) (x = A..B)

Address offset: 0x04

Reset value: 0x4444 4444

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16
CNF15	MODE15	CNF14	MODE14	CNF13	MODE13	CNF12	MODE12
rw	rw	rw	rw	rw	rw	rw	rw

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
CNF11	MODE11	CNF10	MODE10	CNF9	MODE9	CNF8	MODE8

Bit	Field	Type	Reset	Description
31: 30 27: 26 23: 22 19: 18 15: 14 11: 10 7: 6 3: 2	CNFy	rw	0x01	Port x configuration bits (8…15) These bits are written by software to configure the corresponding I/O port. Refer to Table 18: Port bit configuration table. In input mode (MODE = 00): 00: Analog input mode 01: Input floating mode 10: Input pull-up/pull-down mode 11: Reserved In output mode (MODE[1: 0] > 00): 00: General purpose output push-pull 01: General purpose output open-drain 10: Alternate function output push-pull 11: Alternate function output open-drain
29: 28 25: 24 21: 20 17: 16 13: 12 9: 8 5: 4 1: 0	MODEy	rw	0x00	Port x mode bits (y = 8…15) These bits are written by software to configure the corresponding I/O port. Refer to Table 18: Port bit configuration table. 00: Input mode (reset state) 01: Output mode 10: Reserved 11: Reserved

6.3.3 Port input data register (GPIOx_IDR) (x = A..B)

Address offset: 0x08
Reset value: 0x0000 XXXX

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

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

Bit	Field	Type	Reset	Description
31: 16	Reserved			always read as 0.
15: 0	IDRy	r	0xXXXX	Port input data (y = 0..15) These bits are read only and can be accessed in Word (16-bit) mode only. They contain the input value of the corresponding I/O port.

6.3.4 Port output data register (GPIOx_ODR) (x = A..B)

Address offset: 0x0C

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31: 16	Reserved			always read as 0.
15: 0	ODRy	rw	0x0000	Port output data (y = 0..15) These bits are read only and can be accessed in Word (16-bit) mode only. Note: The ODR bits can be individually set and cleared by writing to the GPIOx_BSRR (x = A…B).

6.3.5 Port bit set/reset register (GPIOx_BSRR) (x = A..B)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
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
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bit	Field	Type	Reset	Description
31: 16	BRy	w	0x0000	Port x Reset bit y (y = 0…15) These bits are write-only and can be accessed in Word (32-bit) mode only. 0: no effect on the corresponding ODRy bit 1: clear the corresponding ODRy bit as ‘0’
15: 0	BSy	w	0x0000	Port x Set bit y (y = 0…15) These bits are write-only and can be accessed in Word (32-bit) mode only. 0: no effect on the corresponding ODRy bit 1: set the corresponding ODRy bit to ‘1’

6.3.6 Port bit reset register (GPIOx_BRR) (x = A..B)

Address offset: 0x14

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31: 16	Reserved			always read as 0
15: 0	BRy	w	0x0000	Port x Reset bit y (y = 0…15) These bits are write-only and can be accessed in Word (16-bit) mode only. 0: no effect on the corresponding ODRy bit 1: clear the corresponding ODRy bit as ‘0’

6.3.7 Port configuration lock register (GPIOx_LCKR) (x = A..B)

This register is used to lock the configuration of the port bits when a correct write sequence is applied to bit 16 (LCKK). The value of bits [15:0] is used to lock the configuration of the GPIO. During the write sequence, the value of LCKR[15:0] must not change. When the LOCK sequence has been applied on a port bit it is no longer possible to modify the value of the port bit until the next reset.

Each lock bit freezes the corresponding 4 bits of the control register (CRL, CRH).

Address offset: 0x18

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31: 17	Reserved			always read as 0
16	LCKK	rw	0x00	Lock key This bit can be read anytime. It can only be modified using the Lock Key Writing Sequence. 0: port configuration lock key not active 1: port configuration lock key active. GPIOx_LCKR register is locked until the next reset. LOCK key writing sequence: Write 1-> Write 0-> Write 1-> Read 0-> Read 1 (last read is optional but confirms that the lock is active) Note: During the LOCK Key Writing sequence, the value of LCK must not change. Any error in the lock key writing sequence will abort the lock.
15: 0	LCKy	rw	0x00	Port x Lock bit y (y = 0…15) These bits are read write but can only be written when the LCKK bit is 0. 0: port configuration not locked 1: port configuration locked

6.3.8 Port output open drain control register (GPIOx_DCR) (x = A..B)

Address offset: 0x1C

Reset value: 0x0000 0000

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16
PX15	PX14	PX13	PX12	PX11	PX10	PX9	PX8
rw	rw	rw	rw	rw	rw	rw	rw

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
PX7	PX6	PX5	PX4	PX3	PX2	PX1	PX0
rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31: 2	PX15-PX1	rw	0x00	see PX0
1: 0	PX0	rw	0x00	PX0[1: 0]: 2'b11: output open-drain mode with port pull-up 2'b01: output open-drain mode with port pull-down 2'bx0: output open-drain mode without port pull-up/pull-down

6.3.9 Port alternate function register low (GPIOx_AFRL) (x = A..B)

Address offset: 0x20

Reset value: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFR7				AFR6				AFR5				AFR4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFR3				AFR2				AFR1				AFR0			
rw				rw				rw				rw			

Bit	Field	Type	Reset	Description
31: 0	AFRy	rw	0xFFFF FFFF	Port x alternate function bit y (y = 0…7) These bits can be accessed by software to configure the IO alternate function. 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: Reserved 0110: Reserved 0111: Reserved

6.3.10 Port alternate function register high (GPIOx_AFRH) (x = A..B)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFR15				AFR14				AFR13				AFR12			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFR11				AFR10				AFR9				AFR8			
rw				rw				rw				rw			

Bit	Field	Type	Reset	Description

31: 0	AFRy	rw	0x0000 0000	Port x alternate function bit y (y = 8…15) These bits can be accessed by software to configure the IO alternate function. 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: Reserved 0110: Reserved 0111: Reserved

7 Interrupts and Events (EXTI)

Interrupt and Events (EXTI)

7.1 Nested vectored interrupt controller

Features

- All interrupts are maskable (except NMI)
- 16 programmable priority levels (4 bits of interrupt priority are used)
- Low-latency exception and interrupt handling
- Power management control
- Implementation of system control registers

The nested vectored interrupt controller (NVIC) and the processor core interface are closely coupled, which enables low latency interrupt processing and efficient processing of late arriving interrupts.

All interrupts including the core exceptions are managed by the NVIC. For more information on exceptions and NVIC programming, refer to CPU Technical Reference Manual.

7.1.1 SysTick calibration value register

The SysTick calibration value is set to 6000, which gives a reference time base of 1ms with the SysTick clock set to 6MHz (HCLK/4, HCLK = 24MHz).

7.1.2 Interrupt and exception vectors

The following table is the vector table for this product series.

Table 29. Vector table for this product series

Position	Priority	Type of priority	Name	Description	Address
	-	-	-	Reserved	0x0000_0000
	-3	fixed	Reset	Reset	0x0000_0004
	-2	fixed	NMI	Non maskable interrupt The RCC Clock Security System (CSS) is linked to the NMI vector.	0x0000_0008
	-1	fixed	HardFault	All types of fault	0x0000_000C
	0	settable	MemManage	Memory management	0x0000_0010

(EXTI)

Position	Priority	Type of priority	Name	Description	Address
	1	settable	BusFault	Pre-fetch fault, memory access fault	0x0000_0014
	2	settable	UsageFault	Undefined instruction or illegal state	0x0000_0018
	-	-	-	Reserved	0x0000_001C ~ 0x0000_002B
	3	settable	SVCall	System service call via SWI instruction	0x0000_002C
	4	settable	DebugMonitor	Debug Monitor	0x0000_0030
	-	-	-	Reserved	0x0000_0034
	5	settable	PendSV	Pendable request for system service	0x0000_0038
	6	settable	SysTick	System tick timer	0x0000_003C
0	7	settable	WWDG_IWDG/EXTI17	Window watchdog and independent watchdog interrupts EXTI line [17] interrupt	0x0000_0040
1	8	settable	PVD	PVD through EXTI16 detection interrupt	0x0000_0044
2	9	settable	-	Reserved	0x0000_0048
3	10	settable	FLASH	Flash global interrupt	0x0000_004C
4	11	settable	RCC	RCC global interrupt	0x0000_0050
5	12	settable	EXTI0_1	EXTI line [1: 0] interrupt	0x0000_0054
6	13	settable	EXTI2_3	EXTI line [3: 2] interrupt	0x0000_0058
7	14	settable	EXTI4_15	EXTI line [15: 4] interrupt	0x0000_005C
8	15	settable	-	Reserved	0x0000_0060
9	16	settable	-	Reserved	0x0000_0064
10	17	settable	-	Reserved	0x0000_0068
11	18	settable	-	Reserved	0x0000_006C
12	19	settable	ADC1	ADC1 interrupt	0x0000_0070
13	20	settable	TIM1_BRK_UP_TRG_COM	TIM1 brake, update, trigger, and COM interrupt	0x0000_0074
14	21	settable	TIM1_CC	TIM1 capture compare interrupt	0x0000_0078
15	22	settable	-	Reserved	0x0000_007C
16	23	settable	TIM3	TIM3 global interrupt	0x0000_0080
17	24	settable	-	Reserved	0x0000_0084

(EXTI)

Position	Priority	Type of priority	Name	Description	Address
18	25	settable	-	Reserved	0x0000_0088
19	26	settable	TIM14	TIM14 global interrupt	0x0000_008C
20	27	settable	-	Reserved	0x0000_0090
21	28	settable	-	Reserved	0x0000_0094
22	29	settable	-	Reserved	0x0000_0098
23	30	settable	I2C1	I2C1 global interrupt	0x0000_009C
24	31	settable	-	Reserved	0x0000_00A0
25	32	settable	SPI1	SPI1 global interrupt	0x0000_00A4
26	33	settable	-	Reserved	0x0000_00A8
27	34	settable	UART1	UART1 global interrupt	0x0000_00AC
28	35	settable	UART2	UART2 global interrupt	0x0000_00B0
29	36	settable	-	Reserved	0x0000_00B4
30	37	settable	-	Reserved	0x0000_00B8

7.2 External interrupt/event controller (EXTI)

The external interrupt/event controller (EXTI) manages external and internal asynchronous events/interrupts and generates corresponding event requests to CPU/interrupt controller and wakeup requests to power management.

Edge detectors generating event/interrupt requests. Each input line can be independently configured to select the input type (pulse or pending) and the corresponding trigger event (rising or falling or both). Each input line can also be masked independently. A pending register maintains the status line of the interrupt requests.

7.2.1 Main features

The EXTI controller main features are the following:

- Independent trigger and mask on each interrupt/event line
- Dedicated status bit for each interrupt line
- Generation of software event/interrupt requests
- Detection of external signal with pulse width lower than APB1 clock period. Refer to the electrical characteristics section of the datasheet for details on this parameter.

(EXTI)

7.2.2 Block diagram

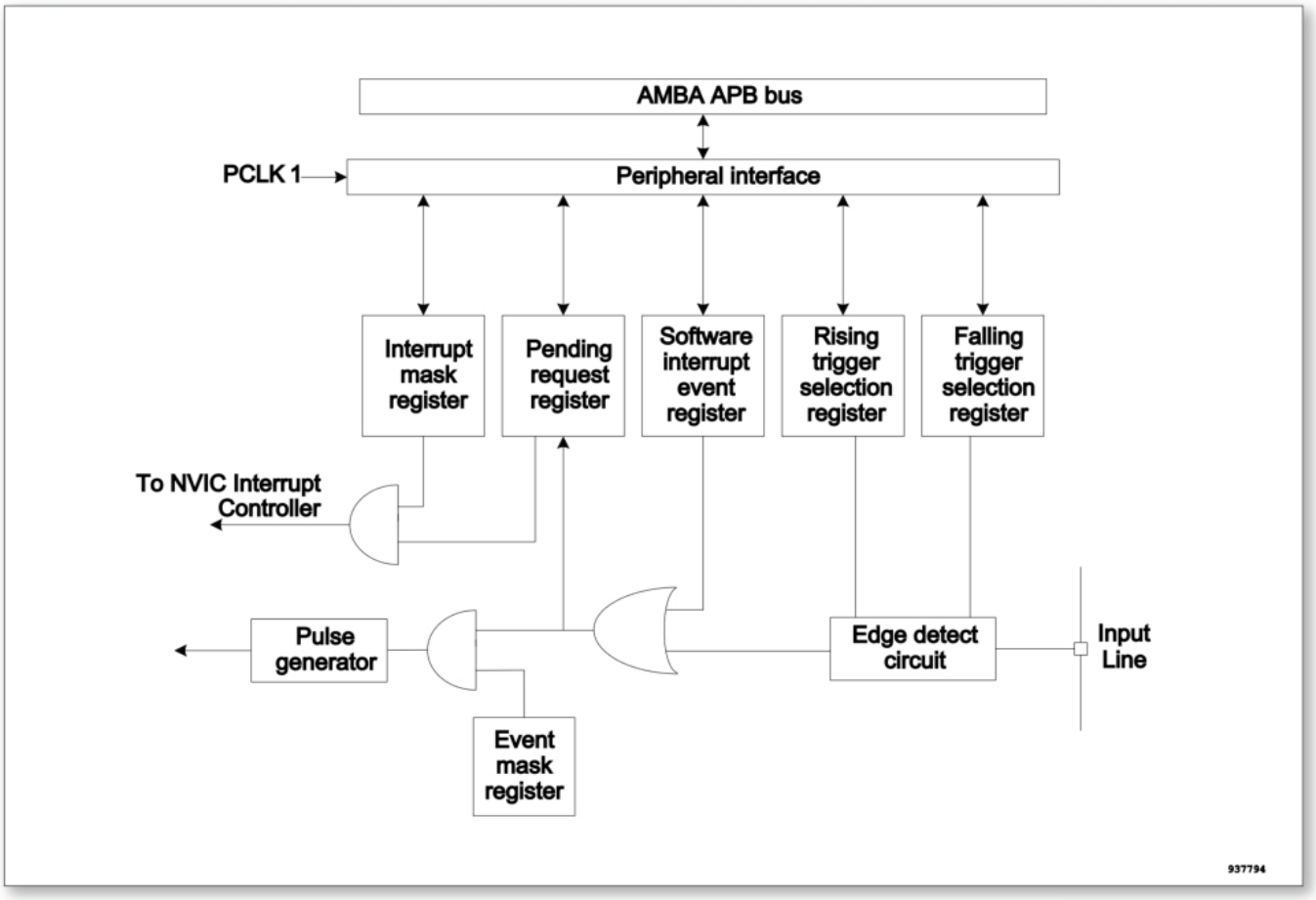


Figure 19. External interrupt/event controller block diagram

7.2.3 Wakeup event management

It is possible to wake up the core (WFE) by handling external or internal events. The wakeup event can be generated either by:

- enabling an interrupt in the peripheral control register but not in the NVIC, and enabling the SEVONPEND bit in the CPU system control register.

  When the CPU resumes from WFE, the peripheral interrupt pending bit and the peripheral NVIC interrupt channel pending bit (in the NVIC interrupt clear pending register) have to be cleared.
- or configuring an external or internal EXTI line in event mode. When the CPU resumes from WFE, it is not necessary to clear the peripheral interrupt pending bit or the NVIC interrupt channel pending bit as the pending bit corresponding to the event line is not set.

To use an external I/O port as a wakeup event, refer to functional description below.

7.2.4 Functional description

To generate the interrupt, the interrupt line should be configured and enabled. This is done by programming the two trigger registers with the desired edge detection and by enabling the interrupt request by writing a ‘1’ to the corresponding bit in the interrupt mask register. When the selected edge occurs on the external interrupt line, an interrupt

request is generated. The pending bit corresponding to the interrupt line is also set to ‘1’. This interrupt request is reset by writing a ‘1’ in the pending register.

To generate the event, the event line should be configured and enabled. This is done by programming the two trigger registers with the desired edge detection and by enabling the event request by writing a ‘1’ to the corresponding bit in the event mask register. When the selected edge occurs on the event line, an event request pulse is generated. The corresponding pending bit is not set to ‘1’.

An interrupt/event request can also be generated by software by writing a ‘1’ in the software interrupt/event register.

Hardware interrupt selection

To configure multiple lines as interrupt sources, use the following procedure:

- Configure the mask bits of the Interrupt lines (EXTI_IMR)
- Configure the Trigger Selection bits of the Interrupt lines (EXTI_RTSR and EXTI_FTSR)
- Configure the enable and mask bits that control the NVIC interrupt channel mapped to the External Interrupt Controller (EXTI) so that a request coming from interrupt lines can be correctly acknowledged.

Hardware event selection

To configure multiple lines as event sources, use the following procedure:

- Configure the mask bits of Event lines (EXTI_EMR)
- Configure the Trigger Selection bits of the Event lines (EXTI_RTSR and EXTI_FTSR)

Software interrupt/event selection

Multiple lines can be configured as software interrupt/event lines. The following is the procedure to generate a software interrupt.

- Configure the mask bits of Interrupt/Event lines (EXTI_IMR, EXTI_EMR)
- Set the required bit of the software interrupt register (EXTI_SWIER)

7.2.5 External interrupt/event line mapping

The GPIOs are connected to the 16 external interrupt/event lines in the following manner:

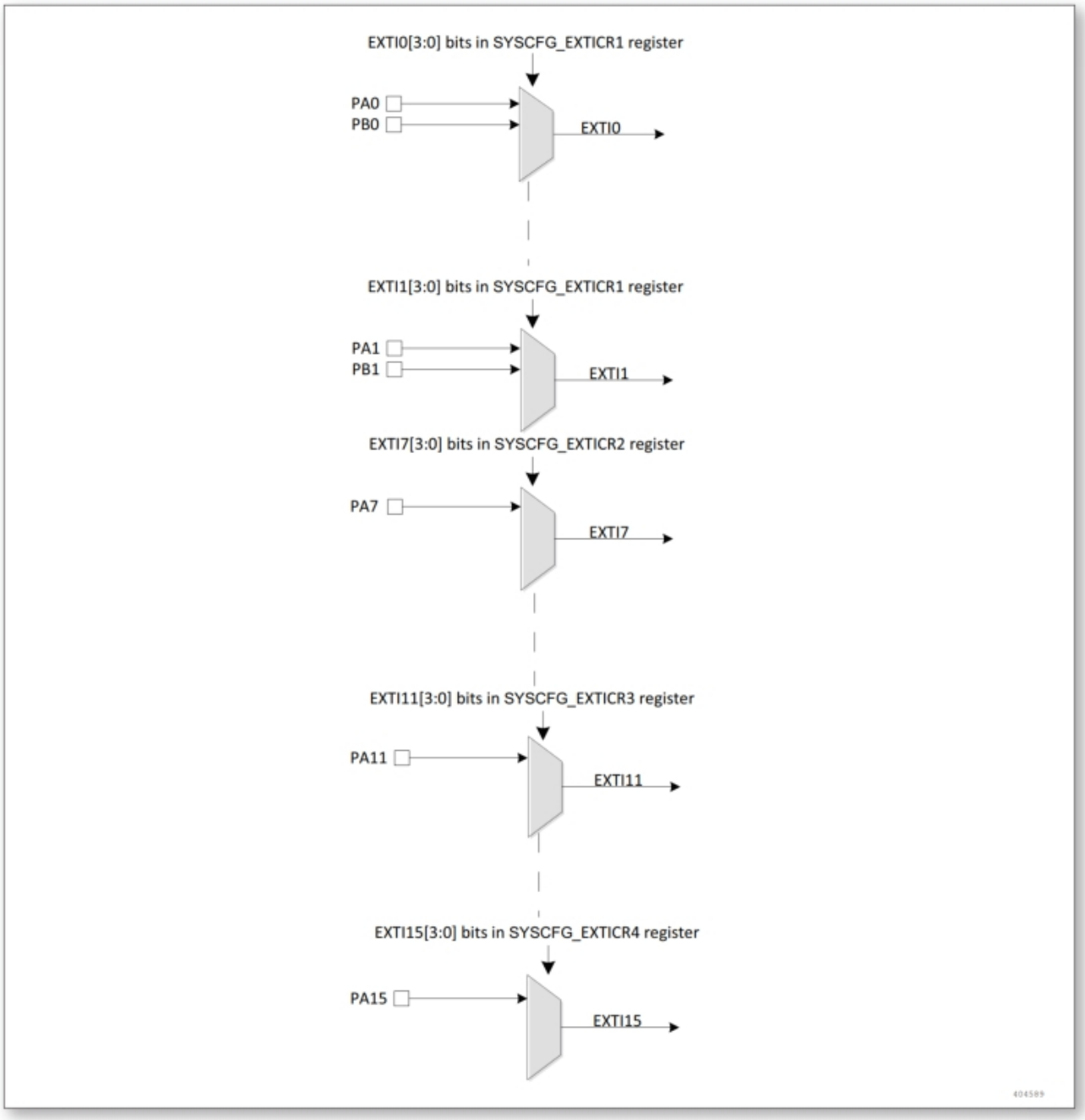


Figure 20. External interrupt GPIO mapping

Note: The corresponding GPIOs in the above diagram may vary due to the actual chip package and they are subject to the actual chip.

The other EXTI/event controllers are connected as follows:

- EXTI line 16 is connected to the PVD output
- EXTI line 17 is connected to the IWDG output

7.3 EXTI register description

Table 30. Overview of EXTI registers

Offset	Acronym	Register Name	Reset	Section
0x00	EXTI_IMR	Interrupt mask register	0x00000000	Section 7.3.1
0x04	EXTI_EMR	Event mask register	0x00000000	Section 7.3.2
0x08	EXTI_RTSR	Rising trigger selection register	0x00000000	Section 7.3.3
0x0C	EXTI_FTSR	Falling trigger selection register	0x00000000	Section 7.3.4
0x10	EXTI_SWIER	Software interrupt event register	0x00000000	Section 7.3.5
0x14	EXTI_PR	Pending register	0x00000000	Section 7.3.6

7.3.1 Interrupt mask register (EXTI_IMR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														IMR17	IMR16
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IMR15	IMR14	IMR13	IMR12	IMR11	IMR10	IMR9	IMR8	IMR7	IMR6	IMR5	IMR4	IMR3	IMR2	IMR1	IMR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:18	Reserved			always read as 0.
17:0	IMRx	rw	0x00	Interrupt Mask on line x 1: interrupt request from line x is not masked 0: interrupt request from line x is masked

7.3.2 Event mask register (EXTI_EMR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														EMR17	EMR16
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EMR15	EMR14	EMR13	EMR12	EMR11	EMR10	EMR9	EMR8	EMR7	EMR6	EMR5	EMR4	EMR3	EMR2	EMR1	EMR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:18	Reserved			always read as 0.

Bit	Field	Type	Reset	Description
17:0	EMRx	rw	0x00	Event Mask on line x 1: event request from line x is not masked 0: event request from line x is masked

7.3.3 Rising trigger selection register (EXTI_RTSR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														TR17	TR16
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TR15	TR14	TR13	TR12	TR11	TR10	TR9	TR8	TR7	TR6	TR5	TR4	TR3	TR2	TR1	TR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:18	Reserved			always read as 0.
17:0	TRx	rw	0x00	Rising trigger event configuration bit of line x 1: rising trigger enabled (for Event and Interrupt) for input line x. 0: rising trigger disabled (for Event and Interrupt) for input line x.

7.3.4 Falling trigger selection register (EXTI_FTSR)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														TR17	TR16
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TR15	TR14	TR13	TR12	TR11	TR10	TR9	TR8	TR7	TR6	TR5	TR4	TR3	TR2	TR1	TR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

(EXTI)

Bit	Field	Type	Reset	Description
23:18	Reserved			always read as 0.
17:0	TRx	rw	0x00	Falling trigger event configuration bit of line x 1: falling trigger enabled (for Event and Interrupt) for input line x. 0: falling trigger disabled (for Event and Interrupt) for input line x.

7.3.5 Software interrupt event register (EXTI_SWIER)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														SWIER17	SWIER16
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWIER15	SWIER14	SWIER13	SWIER12	SWIER11	SWIER10	SWIER9	SWIER8	SWIER7	SWIER6	SWIER5	SWIER4	SWIER3	SWIER2	SWIER1	SWIER0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit		Type	Reset	Description
23:18	Reserved			always read as 0.
17:0	SWIERx	rw	0x00	Software interrupt on line x Writing a '1' to this bit when it is set to '0' sets the corresponding pending bit in the EXTI_PR. At that time, an interrupt generates if the interrupt is enabled in the EXTI_INTMASK and EXTI_EVNTMASK. Note: This bit is cleared as '0' by clearing the corresponding bit of EXTI_PEND (by writing a 1 into the bit).

7.3.6 Pending register (EXTI_PR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														PR17	PR16
														rc_w1	rc_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
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

(EXTI)

Bit	Field	Type	Reset	Description
23:18	Reserved			always read as 0.
17:0	PRx	rc_w1	0x00	Pending bit 1: selected trigger request occurred 0: no trigger request occurred This bit is set to ‘1’ when the selected edge event arrives on the external interrupt line. This bit is cleared by writing a ‘1’ into the bit or changing the polarity of edge detection.

8 Analog-to-digital converter (ADC)

Analog-to-digital converter (ADC)

8.1 ADC introduction

The 12-bit ADC is a successive approximation analog-to-digital converter (SAR A/ converter).

A/D converter supports multiple operation modes: single conversion, continuous conversion, or automatic scan through selected channel. The ADC converter can be started by software, external pin trigger, and individual timers.

The window comparator (analog watchdog) allows the application to detect if the input voltage goes outside the user-defined high or low thresholds.

The ADC input clock is generated from the PCLK1 clock divided by a prescaler and it must not exceed 15MHz.

8.2 ADC main features

- SAR ADC with up to 12-bit programmable resolution; no less than 8 multiplexed external input channels and 1 internal channel
- Conversion rate up to 1Msps
- Support normal operation mode:
  - Single conversion mode: A/D converter does one conversion in the specified channel
  - One-cycle scan mode: A/D converter does one-cycle conversion in all specified channels (from low sequence number channel to high sequence number channel, or from high sequence number channel to low sequence number channel)
  - Continuous scan mode: A/D converter does continuous one-cycle conversions until the software stops A/D conversion.

    A/D conversion must stop if the conversion channel needs to be changed in the midway. The conversion can restart after configuring the corresponding register.
- Support arbitrary channel operation mode:
  - Single conversion mode: A/D converter does one conversion in the specified channel
  - One-cycle scan mode: A/D converter does one-cycle conversion in all specified channels (in any sequence)
  - Continuous scan mode: A/D converter does continuous one-cycle conversions until the software stops A/D conversion. A/D conversion doesn't have to stop if the conversion channel needs to be changed. The conversion in a new channel will start in the next scan cycle after configuring the corresponding register.
- Support the configuration of channel sampling time and resolution via software
- A/D conversion start condition:

  - By software
  - By external trigger with the configuration of delay via software
  - Timer1/3 matching or TRGO signal, external EXTI signal source
- Analog watchdog for comparing the conversion result with the specified value; an interrupt generation can be set by the user when the conversion result matches with the specified value.

8.3 System block diagram

The ADC system block diagram is shown below:

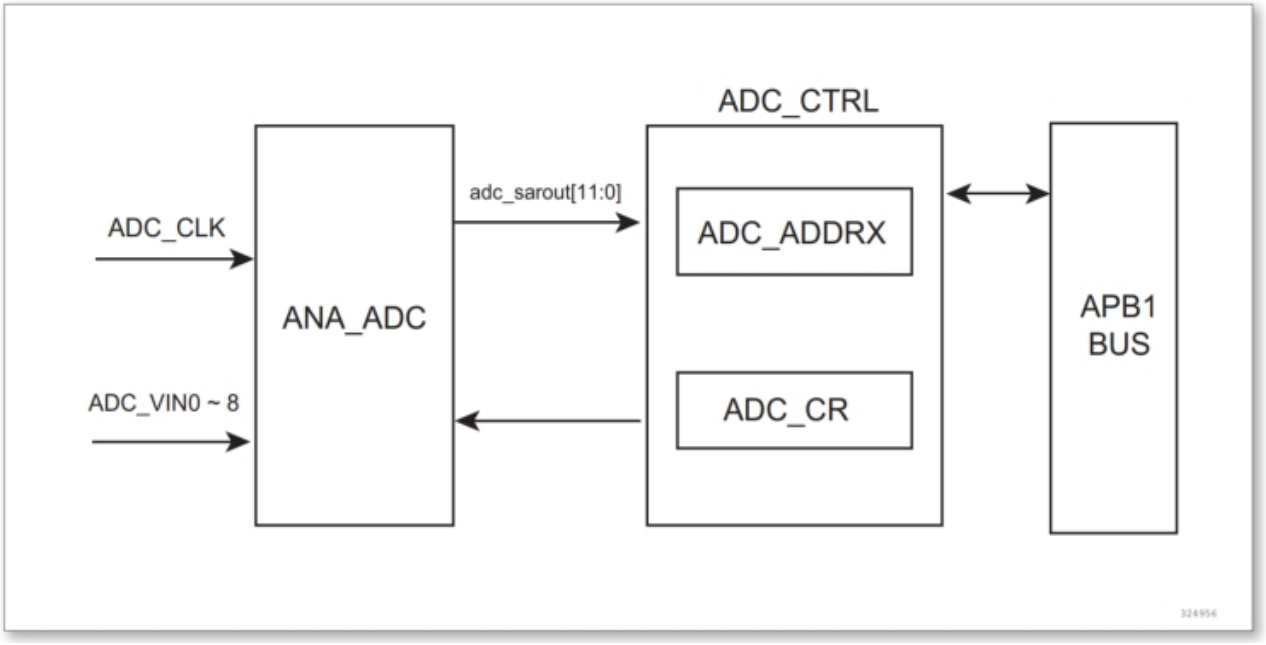


Figure 21. ADC system block diagram

8.4 ADC functional description

The figure below shows a single ADC block diagram.

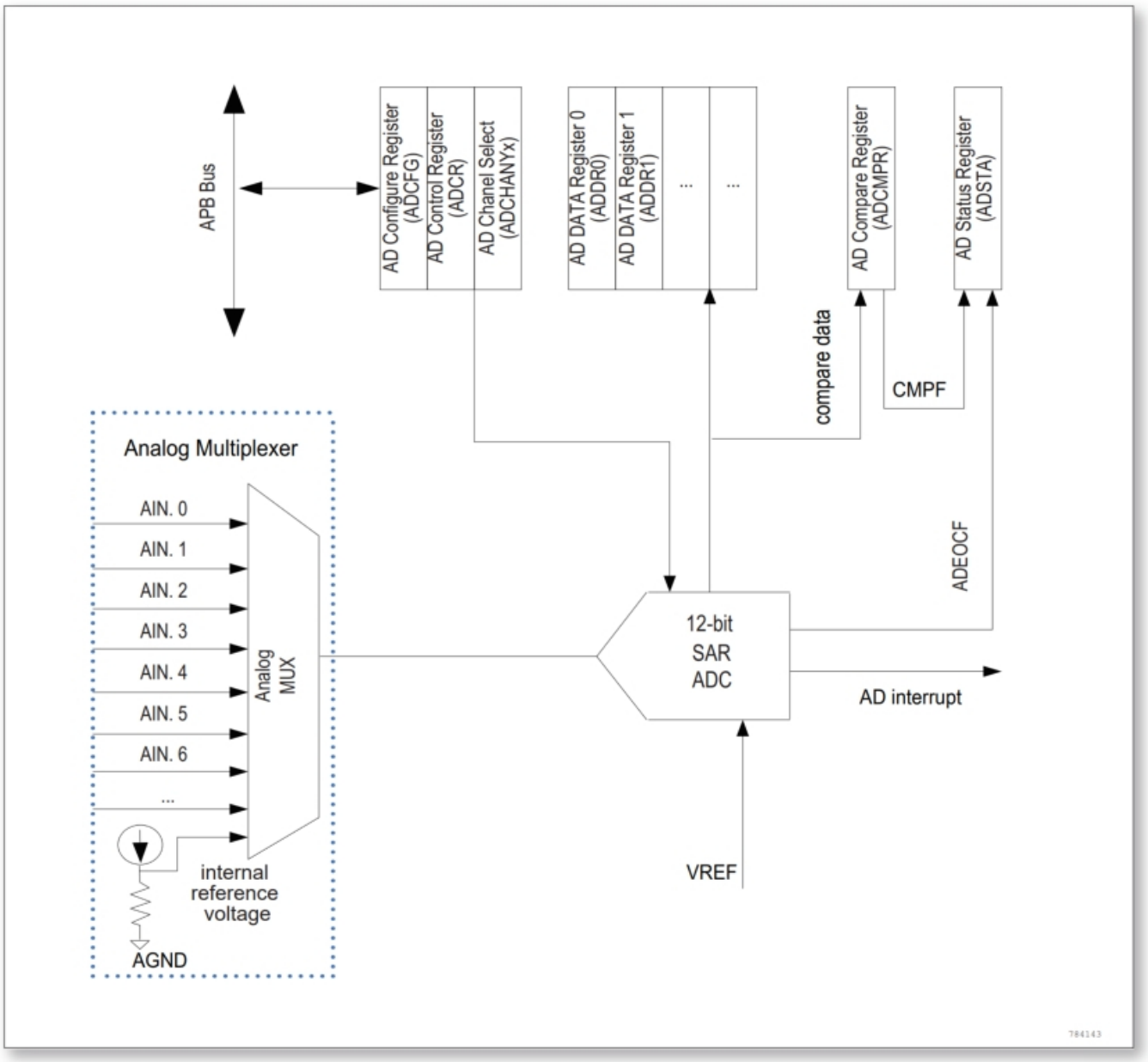


Figure 22. ADC block diagram

Note: V_SENSOR (internal reference voltage) is in the ADC AIN8 channel.

8.4.1 ADC on-off control

The ADC can be powered-on by setting the ADON bit in the ADCFG register. When the ADEN bit is set for the first time, it wakes up the ADC from Power Down mode.

Conversion starts when ADST bit is set for the ADCR register after an ADC power-up delay has elapsed (around 200ns).

The conversion can be stopped by resetting the ADST bit, and the ADC put in power down mode by resetting the ADEN bit.

8.4.2 Channel selection

ADC1 has 8 multiplexed channels including external input channels and internal 1.2V reference voltage channel. Each external input channel has an independent enable bit that can be set by CHANY_NUM, ADC_CHANY0 and ADC_CHANY1.

8.5 Arbitrary channel operation mode

8.5.1 Single conversion mode

In Single conversion mode the A/D converter does one conversion in the corresponding channel. The specific procedures are as follows:

- Set by software the ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 registers, the conversion channel, and the CHANY_MDEN bit. (only CHANY_SEL0 has to be set for the single conversion mode)
- Set the ADST bit in the ADCR register by software, external trigger input, or timer overflow and start the A/D conversion.
- After A/D conversion, the converted data is stored in the data registers (ADDATA and ADDRn).
- After A/D conversion, set the ADIF bit of the status register ADSTA to '1'. At this time, an AD EOC (End Of Conversion) interrupt request is generate if the ADIE bit in the control register ADCR is set to '1'.
- ADST bit remains 1 during the A/D conversion period. Once the A/D channel sampling is complete, the ADST is automatically cleared and the A/D converter enters the idle mode.
- If the software updates ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 during the A/D conversion period, the hardware will not update these settings immediately. The updates will be applied when the conversion of currently configured channels is complete, waiting for setting ADST by software next time.

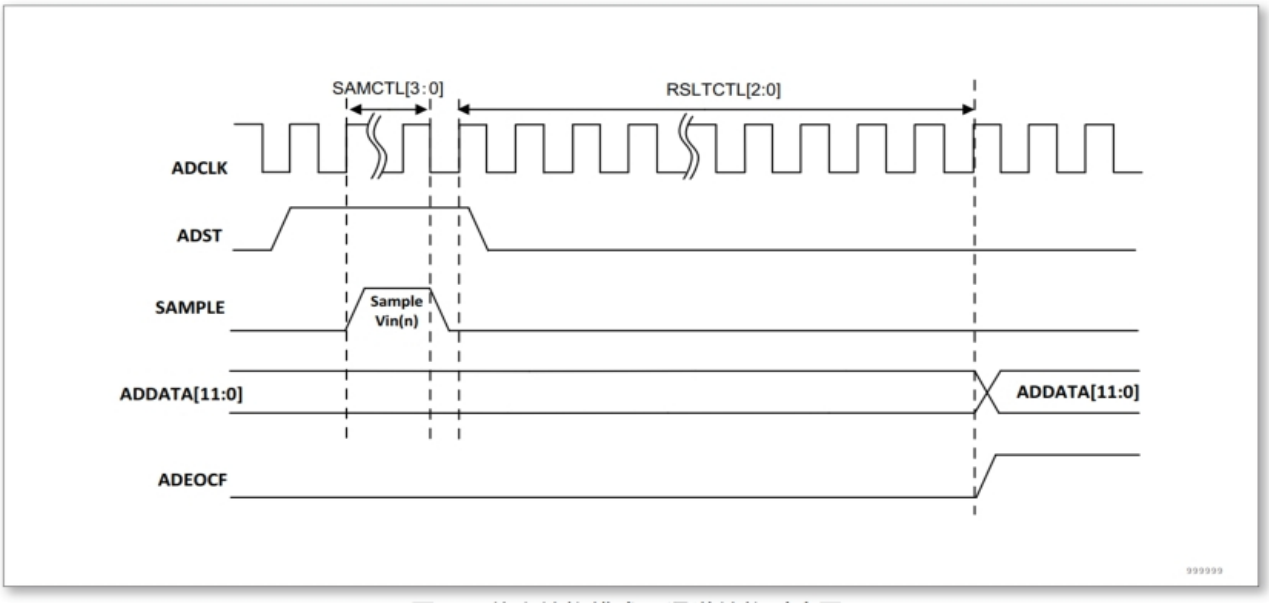


Figure 23. Channel conversion timing diagram in Single conversion mode

8.5.2 One-cycle scan mode

In One-cycle scan mode the A/D converter does one conversion according to the software configuration. The specific procedures are as follows:

- Set by software the ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 registers, desired channels and quantity, and the CHANY_MDEN bit.
- Set the ADST bit in the ADCR register by software or external trigger with the configuration of trigger delay via software. A/D conversion direction is from CHANY_SEL0 to CHANY_SEL8. The number of conversion channels is configured by CHANY_NUM.

  The number is arbitrarily configured from CHANY_SEL0 to CHANY_SEL8, which can be identical or not.
- After the end of A/D conversion in each channel, the converted values will be loaded to the data register of corresponding channel in order. The ADIF EOC flag is set. At this time, an AD EOC interrupt request is generate if the ADIE bit in the control register ADCR is set to ‘1’.
- Once the final A/D channel sampling is complete, the ADST is automatically cleared and the A/D converter enters the idle mode.
- If the software updates ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 during the A/D conversion period, the hardware will not update these settings immediately. The updates will be applied when the conversion of currently configured channels is complete, waiting for setting ADST by software next time.

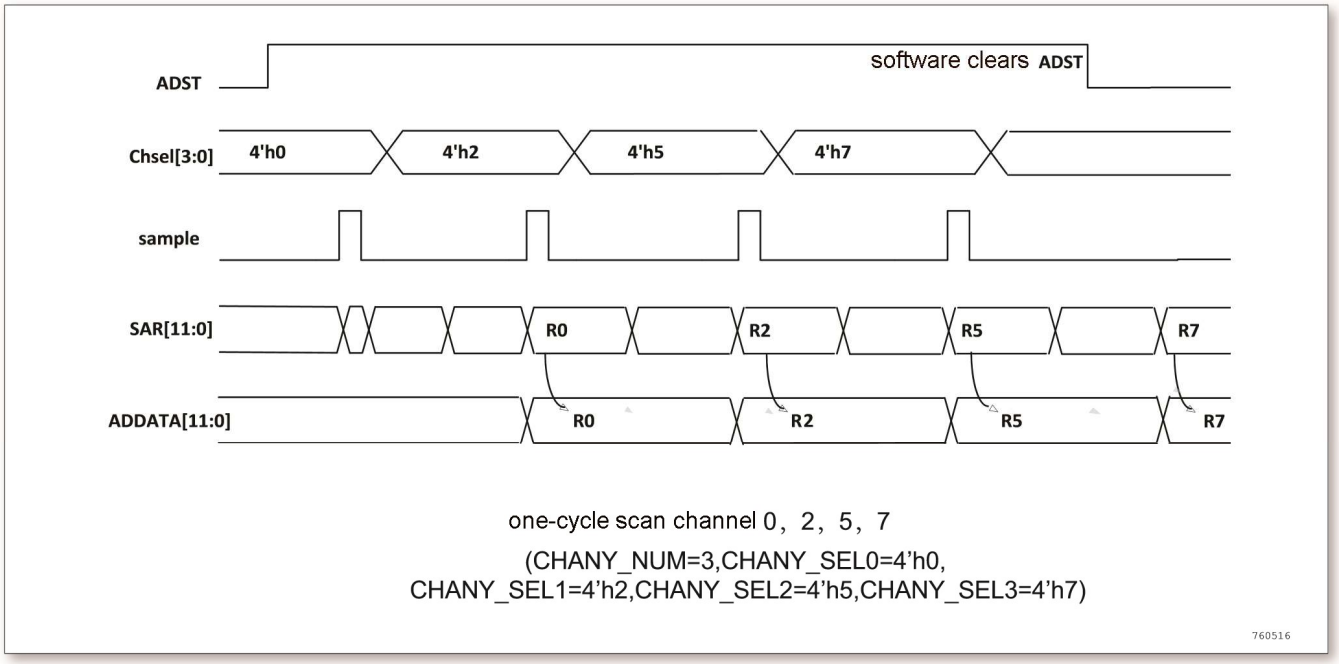


Figure 24. Channel conversion timing diagram in One-cycle conversion mode

8.5.3 Continuous scan mode

In continuous scan mode the A/D converter does continuous conversions according to the software configuration until the software stops conversion. The specific procedures are as follows:

- Set by software the ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 registers, desired channels and quantity, and the CHANY_MDEN bit.
- Set the ADST bit in the ADCR register by software or external trigger with the configuration of trigger delay via software. A/D conversion direction is from CHANY_SEL0 to CHANY_SEL8. The number of conversion channels is configured by CHANY_NUM.

The number is arbitrarily configured from CHANY_SEL0 to CHANY_SEL8, which can be identical or not.

- After the end of A/D conversion in each channel, the converted values will be loaded to the data register of corresponding channel in order. The ADIF EOC flag is set. At this time, an AD EOC interrupt request is generate if the ADIE bit in the control register ADCR is set to '1'.
- The A/D conversion continues as long as the ADST bit remains 1. Once the ADST bit is cleared and the current A/D conversion is complete, the A/D converter enters the idle mode.
- If the software updates ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 during the A/D conversion period, the hardware will not update these settings immediately. The updates will be applied when the conversion of current configured channels is complete, i.e. A conversion in the new channel starts in the next scan cycle.

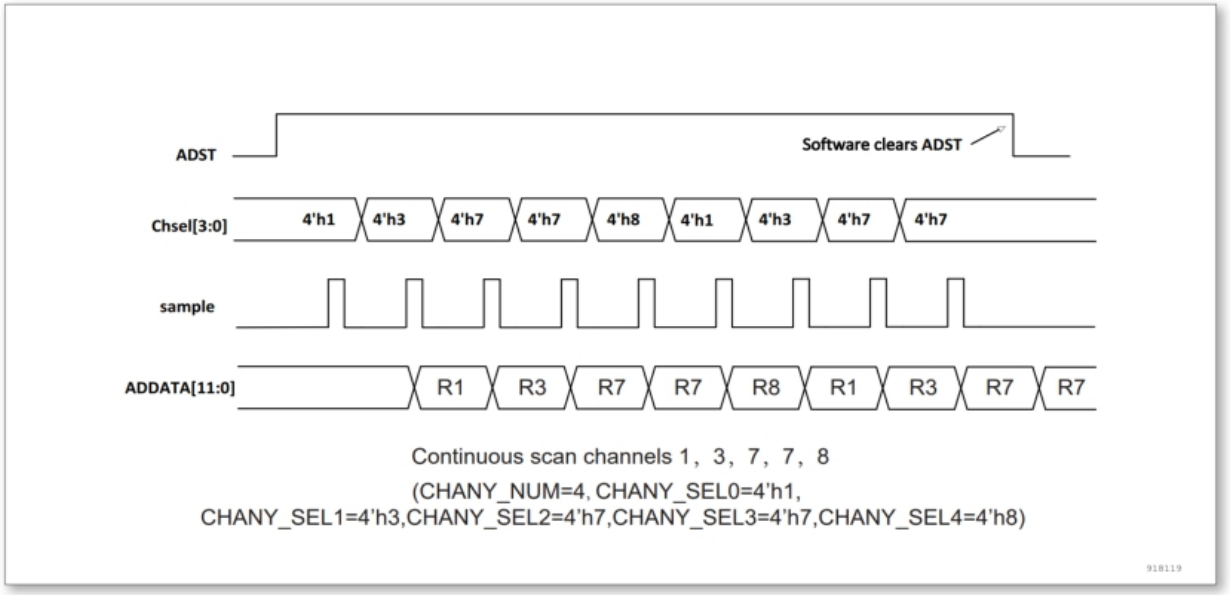


Figure 25. Channel conversion timing diagram in Continuous scan mode

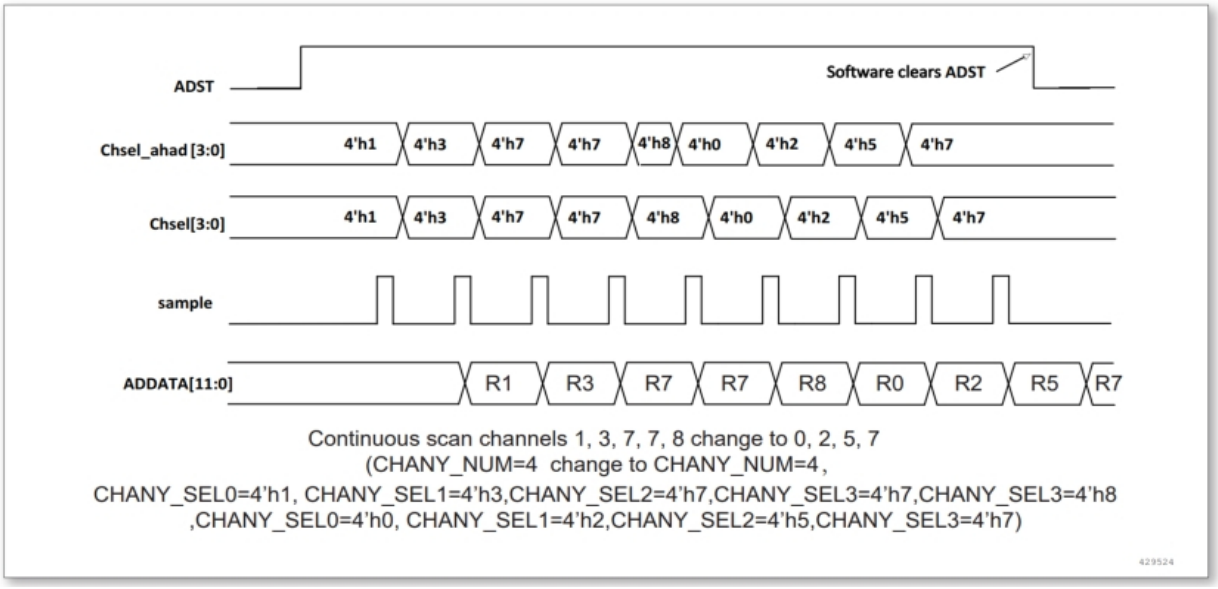


Figure 26. Timing diagram with dynamically updated configuration, in Continuous scan mode

8.6 Data alignment

ALIGN bit in the ADCR register selects the alignment of data stored after conversion. Data can be left or right aligned as shown in figure below.



Figure 27. Data alignment style

8.6.1 Programmable resolution

The ADC conversion significance can be modified using the RSLTCTL[2:0] bits in the ADC_CFG register to accelerate the data conversion speed. The significant data bits are aligned from the higher bits of 12-bit data.

8.6.2 Programmable sampling time

The ADC clock ADCLK is generated from the PCLK 1 divided by a prescaler. The prescale coefficient can be determined by setting the ADCPRE bit in the ADCFG register, i.e. PCLK 1/(N + 2) divided by a prescaler to be the ADC clock. ADC samples the input voltage for a number of ADC_CLK cycles which can be modified using the SAMCTL[3:0] bits in the ADC_CFG registers.

Set the ADC resolution as n bits (n=8,9,10,11,12). The sampling cycles of each channel is m.

The sampling frequency is calculated as follows:

$F_{sample} = F_{ADCLK} / (m + n + 0.5)$.

Suppose the resolution is set to 12 bits, and the sampling cycles of each channel is 2.5T, then $F_{sample} = F_{ADCLK}/15$.

The total conversion time is calculated as follows:

T_{CONV} = Sampling time + 12.5 conversion cycles

Example:

With an ADCCLK = 15MHz and a sampling time of 2.5 cycles:

T_{CONV} = 2.5 + 12.5 = 15 cycles = 1μs

8.7 Conversion on external trigger

ADC conversion can be triggered by an external event (e.g. timer capture, EXTI line). If the TRGEN bit in the ADCR register is set then external events are able to trigger a conversion. An external trigger source can be selected by setting the TRGSEL bit. Refer to relevant bit description of AD control register (ADCR.TRGSEL) for details of external trigger source selection. Refer to relevant bit description of AD control register (ADCR.TRGSHIFT) for details of external trigger delay control. After a trigger signal is generated, the sampling starts after a delay of N PCLK1 clock cycles. In case of trigger scan mode, only the sampling in the first channel is delayed and the rest start immediately after the last sampling is complete.

8.8 Internal reference voltage

The internal signal source channel of ADC is connected to an internal reference voltage of 1.2V. This channel converts the 1.2V reference voltage output to a digital value.

The internal reference voltage has an independent enable bit that can be enabled or disabled by setting the corresponding bit in the register.

8.9 Monitoring AD conversion result in the window comparator mode

An upper limit comparison register and a lower limit comparison register are provided in the comparison mode. The monitoring channel can be selected by setting the CMPCH bit via software.

Providing CPMHDATA ≥ CPMLDATA, and the comparison result is greater than or equal to the specified CMPHDATA value or less than the specified CMPLDATA value in the ADCMPR register, the ADWIF bit in the status register ADSTA is set to 1.

Providing CPMHDATA < CPMLDATA, and the comparison result is equal to the specified CPMHDATA value or between two specified values, the ADWIF bit in the status register ADSTA is set to 1.

An interrupt request is generate if the ADWIE bit in the control register ADCR is set.

8.10 ADC register description

Table 31. Overview of ADC registers

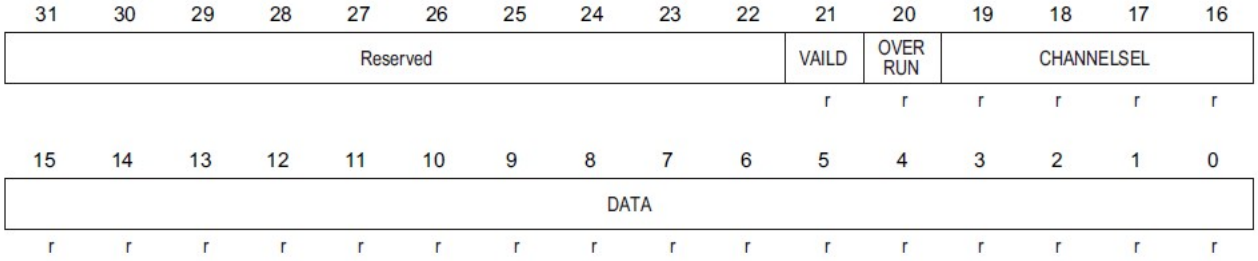
Offset	Acronym	Register Name	Reset	Section
0x00	ADC_ADDATA	A/D data register	0x00000000	Section 8.10.1
0x04	ADC_ADCFG	A/D configuration register	0x00000000	Section 8.10.2
0x08	ADC_ADCR	A/D control register	0x00000000	Section 8.10.3
0x10	ADC_ADCMPR	A/D window compare register	0x00000000	Section 8.10.4
0x14	ADC_ADSTA	A/D status register	0x00000000	Section 8.10.5
0x18~ 0x38	ADC_ADDR 0 ~ 8	A/D data register	0x00000000	Section 8.10.6

Offset	Acronym	Register Name	Reset	Section
0x5C	ADC_CHANY0	A/D arbitrary channel selection register 0	0x00000000	Section 8.10.7
0x60	ADC_CHANY1	A/D arbitrary channel selection register 1	0x00000000	Section 8.10.8
0x64	ADC_ANY_CFG	A/D arbitrary channel configuration register	0x00000000	Section 8.10.9
0x68	ADC_ANY_CR	A/D arbitrary channel control register	0x00000000	Section 8.10.10

8.10.1 A/D data register (ADC_ADDATA)

Address offset: 0x00

Reset value: 0x0000 0000



Bit	Field	Type	Reset	Description
31:22	Reserved			Reserved, always read as 0.
21	VALID	r	0x00	Valid flag (read-only) 1 = valid DATA[11:0] bit data 0 = invalid DATA[11:0] bit data This bit is set after the corresponding analog channel conversion is complete. This bit is cleared by hardware after reading the ADDATA register.
20	OVERRUN	r	0x00	Data overrun flag (read-only) 1 = DATA[11:0] data is overrun 0 = DATA[11:0] data is the last conversion result Before the new conversion result is loaded to the register, OVERRUN is set to 1 if the DATA[11:0] is not read. This bit is cleared by hardware after reading the ADDATA register.
19:16	CHANNELSEL	r	0x00	4 bits show the channel corresponding to the current data (Channel selection) 0000 = conversion data of channel 0 0001 = conversion data of channel 1 0010 = conversion data of channel 2 0011 = conversion data of channel 3 0100 = conversion data of channel 4 0101 = conversion data of channel 5 0110 = conversion data of channel 6 0111 = conversion data of channel 7 1000 = conversion data of channel 8 Others: invalid
15:0	DATA	r	0x00	12-bit A/D conversion result (Transfer data) Data can be left or right aligned as configured.

8.10.2 A/D configuration register (ADC_ADCFG)

Address offset: 0x04

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	ADCPRE	SAMCTL				RSLTCTL			ADCPRE			VSEN	Res.	ADWEN	ADEN
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw

Bit	Field	Type	Reset	Description
31 : 15	Reserved			Reserved and always read as 0.
14	ADCPRE	rw	0x00	ADC prescaler As the least significant bit of ADCPRE[3:0], used together with Bit[6:4]
13 : 10	SAMCTL	rw	0x00	Channel x Sample time selection These bits are used to independently select the sample time of each channel. In a sample cycle, the channel selection bit should remain unchanged. 0000: 2.5 cycles 0100: 42.5 cycles 0001: 8.5 cycles 0101: 56.5 cycles 0010: 14.5 cycles 0110: 72.5 cycles 0011: 29.5 cycles 0111: 240.5 cycles 1000: 3.5 cycles 1001: 4.5 cycles 1010: 5.5 cycles 1011: 6.5 cycles 1100: 7.5 cycles Others: reserved
9:7	RSLTCTL	rw	0x00	ADCx conversion data resolution selection 000: 12 significant bits 001: 11 significant bits 010: 10 significant bits 011: 9 significant bits 100: 8 significant bits
6:4	ADCPRE	rw	0x00	ADC prescaler Set to ‘1’ or cleared as ‘0’ by software to determine ADC clock frequency. PCLK2 divided by (([6:4],[14]) + 2) as the ADC clock
3	VSEN	rw	0x00	Internal reference voltage enable (Voltage Sensor enable) 1: Internal voltage sensor enabled 0: Internal voltage sensor disabled
2	Reserved			Reserved and always read as 0.
1	ADWEN	rw	0x00	A/D window comparator enable (ADC window comparison enable) 1= A/D window comparator enabled 0= A/D window comparator disabled

Bit	Field	Type	Reset	Description
0	ADEN	rw	0x00	A/D conversion enable (ADC enable) 1 = enabled 0 = disabled

8.10.3 A/D control register (ADC_ADCR)

Address offset: 0x08

Reset value: 0x0000 0000



Bit	Field	Type	Reset	Description
31:26	Reserved			Reserved, always read as 0.
25:24	TRG_EDGE	rw	0x00	Trigger edge selection 00: Double-edge trigger 01: Falling trigger register 10: Rising edge trigger 11: Trigger blocked
23:22	Reserved			Reserved and always read as 0.
21:19	TRGSHIFT	rw	0x00	External trigger shift sample After a trigger signal is generated, the sampling starts after a delay of N PCLK2 clock cycles. In case of trigger scan mode, the sampling in the rest channels starts immediately after the last sampling is complete. 0: no delay 1: 4 cycles 2: 16 cycles 3: 32 cycles 4: 64 cycles 5: 128 cycles 6: 256 cycles 7: 512 cycles
18 :17	TRGSEL	rw	0x00	External trigger selection used together with Bit[6:4]

Bit	Field	Type	Reset	Description
15:12	CMPCH	rw	0x00	Window comparison channel selection 0000 = conversion result for selecting comparison channel 0 0001 = conversion result for selecting comparison channel 1 0010 = conversion result for selecting comparison channel 2 0011 = conversion result for selecting comparison channel 3 0100 = conversion result for selecting comparison channel 4 0101 = conversion result for selecting comparison channel 5 0110 = conversion result for selecting comparison channel 6 0111 = conversion result for selecting comparison channel 7 1000 = conversion result for selecting internal comparison reference voltage 1111 = all scan channels Others: invalid
11	ALIGN	rw	0x00	Data alignment 0: right aligned 1: left aligned
10 : 9	ADMD	rw	0x00	A/D conversion mode (ADC mode) 00: single conversion 01: one-cycle conversion 10: continuous scan When the conversion mode is changed, the software has to disable the ADST bit.
8	ADST	rw	0x00	A/D conversion start 1 = conversion start 0 = end of conversion or entering the idle mode There are two ways of setting ADST: In the single or one-cycle mode, ADST is automatically cleared by hardware after conversion. In continuous scan mode the A/D converter does continuous conversions until the software writes '0' to the bit or the system is reset.
7	Reserved			Reserved and always read as 0.

Bit	Field	Type	Reset	Description
6:4	TRGSEL	rw	0x00	External trigger selection, bits [18:17,6:4] Select external trigger source 00101: TIM1_CC4 and CC5 00000: TIM1_CC1 00001: TIM1_CC2 00010: TIM1_CC3 00011: Reserved 00100: TIM3_TRGO 00101: Reserved 00110: TIM3_CC1 00111: EXTI line 11 01000: TIM1_TRGO 01001: Reserved 01010: Reserved 01001: Reserved 01100: TIM3_CC4 01101: Reserved 01110: Reserved 01111: EXTI line 15 10000: TIM1_CC4 10001: TIM1_CC5 Others: invalid
3	Reserved			Reserved and always read as 0.
2	TRGEN	rw	0x00	External hardware trigger source (External trigger enable) 1 = enable A/D conversion by external trigger signal 0 = not enable A/D conversion by external trigger signal
1	ADWIE	rw	0x00	A/D window comparator interrupt enable (ADC window comparison enable) 1= A/D window comparator interrupt enabled 0= A/D window comparator interrupt disabled
0	ADIE	rw	0x00	ADC interrupt enable 1 = ADC interrupt enabled 0 = ADC interrupt disabled An A/D EOC interrupt request is generated if the ADIF bit is set.

8.10.4 A/D window comparison register (ADC_ADCMPR)

Address offset: 0x10

Reset value: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:28	Reserved			Reserved, always read as 0.
27:16	CMPHDATA	rw	0x00	Compare data high limit The 12-bit value will be compared with the conversion result from a specified channel.
15:12	Reserved			Reserved and always read as 0.
11:0	CMPLDATA	rw	0x00	Compare data low limit The 12-bit value will be compared with the conversion result from a specified channel.

8.10.5 A/D status register (ADC_ADSTA)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			OVERRUN									Reserved			VALID
			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
VALID								CHANNEL				Res.	BUSY	ADWIF	ADIF
r	r	r	r	r	r	r	r	r	r	r	r		r	rc_w1	rc_w1

Bit	Field	Type	Reset	Description
31:29	Reserved			Reserved, always read as 0.
28:20	OVERRUN	r	0x00	Data overrun flag for channel 0 to channel 8 (Overrun flag) Read-only. Channel 8 is the voltage sensor channel.
19:17	Reserved			Reserved and always read as 0.

Bit	Field	Type	Reset	Description
16:8	VALID	r	0x00	Valid flag for channel 0 to channel 8 (Valid flag) Read-only. Channel 8 is the voltage sensor channel.
7:4	CHANNEL	r	0x00	Current conversion channel These 4 bits indicate the channel in conversion when BUSY = 1. These 4 bits indicate the channel for the next conversion when BUSY = 0.
3	Reserved			Reserved and always read as 0.
2	BUSY	r	0x00	Busy/Idle 1 = A/D converter is busy 0 = A/D converter is idle
1	ADWIF	rc_w1	0x00	ADC window comparator interrupt flag Providing CPMHDATA ≥ CPMLDATA, and the selected ADC channel comparison result is greater than or equal to the specified CMPHDATA value or less than the specified CMPLDATA value in the ADCMPR register, the ADWIF bit in the status register ADSTA is set to 1. Providing CPMHDATA < CPMLDATA, and the selected ADC channel comparison result is equal to the specified CPMHDATA value or between two specified values, the ADWIF bit in the status register ADSTA is set to 1. This flag bit is cleared by writing ‘1’.
0	ADIF	rc_w1	0x00	ADC convert complete flag This bit is set by hardware at the end of channel group conversion, and cleared by software. 1 = A/D conversion complete 0 = A/D converter incomplete This flag bit is cleared by writing ‘1’.

8.10.6 A/D data register (ADC_ADDR0 ~ 8)

Address offset: 0x18 − 0x38

Reset value: 0x0000 0000



Bit	Field	Type	Reset	Description
31:22	Reserved			Reserved, always read as 0.
21	VALID	r	0x00	Valid flag (read-only) 1 = valid DATA[11:0] bit data 0 = invalid DATA[11:0] bit data This bit is set after the corresponding analog channel conversion is complete. This bit is cleared by hardware after reading the ADDATA register.
20	OVERRUN	r	0x00	Data overrun flag (read-only) 1 = DATA [11:0] data is overrun 0 = DATA [11:0] data is the last conversion result Before the new conversion result is loaded to the register, OVERRUN is set to '1' if the DATA[11:0] is not read. This bit is cleared by hardware after reading the ADDATA register.
19:16	Reserved			Reserved and always read as 0.
15:0	DATA	r	0x00	12-bit A/D conversion result in the channel (Transfer data) Data can be left or right aligned as configured.

8.10.7 A/D arbitrary channel selection register (ADC_CHANY0)

Address offset: 0x5C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CHANY_SEL7				CHANY_SEL6				CHANY_SEL5				CHANY_SEL4			
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
CHANY_SEL3				CHANY_SEL2				CHANY_SEL1				CHANY_SEL0			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

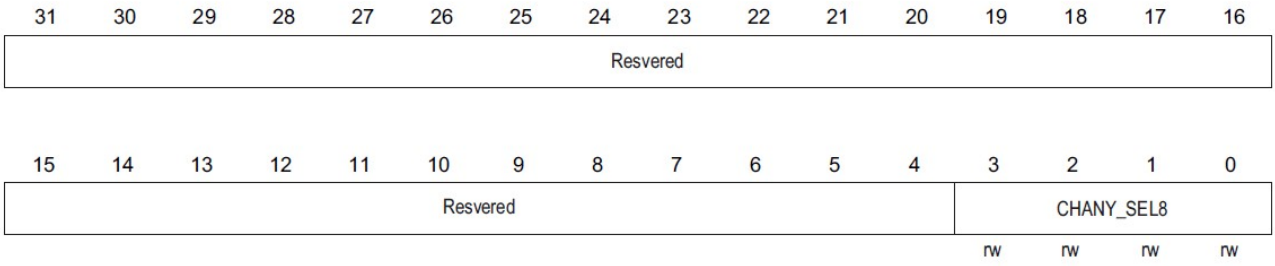
Bit	Field	Type	Reset	Description
31:28	CHANY_SEL7	rw	0x00	can be configured to any channel from channel 0 to channel 8.
27:24	CHANY_SEL6	rw	0x00	can be configured to any channel from channel 0 to channel 8.
23:20	CHANY_SEL5	rw	0x00	can be configured to any channel from channel 0 to channel 8.
19:16	CHANY_SEL4	rw	0x00	can be configured to any channel from channel 0 to channel 8.
15:12	CHANY_SEL3	rw	0x00	can be configured to any channel from channel 0 to channel 8.
11:8	CHANY_SEL2	rw	0x00	can be configured to any channel from channel 0 to channel 8.
7:4	CHANY_SEL1	rw	0x00	can be configured to any channel from channel 0 to channel 8.
3:0	CHANY_SEL0	rw	0x00	can be configured to any channel from channel 0 to channel 8.

Note: In the one-cycle or continuous scan modes, the ADC_CHANY0 shadow register is started by hardware. Before ADC starts to work, writing to ADC_CHANY0 by software also writes to its shadow register. When ADC is working, only the shadow register is updated if the ADC_CHANY0 value is changed. Besides that, when ADC starts to convert the last channel, the value of the shadow register will update to ADC_CHANY0, so as to complete the dynamic channel switch.

8.10.8 A/D arbitrary channel selection register 1 (ADC_CHANY1)

Address offset: 0x60

Reset value: 0x0000 0000



Bit	Field	Type	Reset	Description
31 : 4	Reserved			Reserved and always read as 0.
3:0	CHANY_SEL8	rw	0x00	can be configured to any channel from channel 0 to channel 8.

Note: In the one-cycle or continuous scan modes, the ADC_CHANY1 shadow register is started by hardware. Before ADC starts to work, writing to ADC_CHANY1 by software also writes to its shadow register. When ADC is working, only the shadow register is updated if the ADC_CHANY1 value is changed.

Besides that, when ADC starts to convert the last channel, the value of the shadow register will update to ADC_CHANY1, so as to complete the dynamic channel switch.

8.10.9 A/D arbitrary channel configuration register (ADC_ANY_CFG)

Address offset: 0x64

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31:4	Reserved			Reserved, always read as 0.
3:0	CHANY_NUM	rw	0x00	Number of channels: 0: 0 channel 1: 0 ~ 1 channel 2: 0 ~ 2 channels ... 7: 0 ~ 7 channels 8: 0 ~ 8 channels

Note: In the one-cycle or continuous scan modes, the ADC_NUM shadow register is started by hardware. Before ADC starts to work, writing to ADC_NUM by software also writes to its shadow register. When ADC is working, only the shadow register is updated if the ADC_NUM value is changed. Besides that, when ADC starts to convert the last channel, the value of the shadow register will update to ADC_NUM, so as to complete the dynamic channel switch.

8.10.10 A/D arbitrary channel control register (ADC_ANY_CR)

Address offset: 0x68

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31:1	Reserved			Reserved, always read as 0.
0	CHANY_MDEN	rw	0x00	Arbitrary channel configuration mode enable bit: 1: enabled 0: disabled When this bit is enabled, the function of ADC channel configuration become changed. There are two controlling parts: CHANY_NUM is used to configure the number of channels from channel 0 to channel 8. CHANY_SEL0 ~ CHANY_SEL8 is used to configure them as an arbitrary ADC channel.

Note: In the arbitrary mode plus the one-cycle/continuous scan mode, the ADST bit in the ADC_ADCR must be disabled before closing the ADC. Then the user should judge whether the BUSY bit in the ADC_ADSTA is 0. That is to say, the CHANY_MDEN bit in the ADC_ANY_CR should be disabled when the ADC conversion is complete.

9 Advanced-Control Timer (TIM1)

Advanced-Control Timer (TIM1)

9.1 TIM1 introduction

The advanced-control timer (TIM1) consists of a 16-bit auto-reload counter driven by a programmable prescaler.

It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare, PWM, complementary PWM with dead-time insertion).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers.

The advanced-control (TIM1) and general-purpose (TIMx) timers are completely independent, and do not share any resources. They can be synchronized together as described in the general-purpose timer synchronization section.

9.2 Main features

TIM1 timer features include:

- 16-bit up, down, up/down auto-reload counter.
- 16-bit programmable prescaler allowing dividing (also “on the fly”) the counter clock frequency either by any factor between 1 and 65536.
- Up to 4 independent channels for:
  - Input capture
  - Output compare
  - PWM generation (Edge and Center-aligned Mode)
  - One-pulse mode output
- Complementary outputs with programmable dead-time
- Synchronization circuit to control the timer with external signals and to interconnect several timers together
- Repetition counter to update the timer registers only after a given number of cycles of the counter
- Break input to put the timer’s output signals in reset state or in a known state
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes
- Trigger input for external clock or cycle-by-cycle current management

(TIM1)

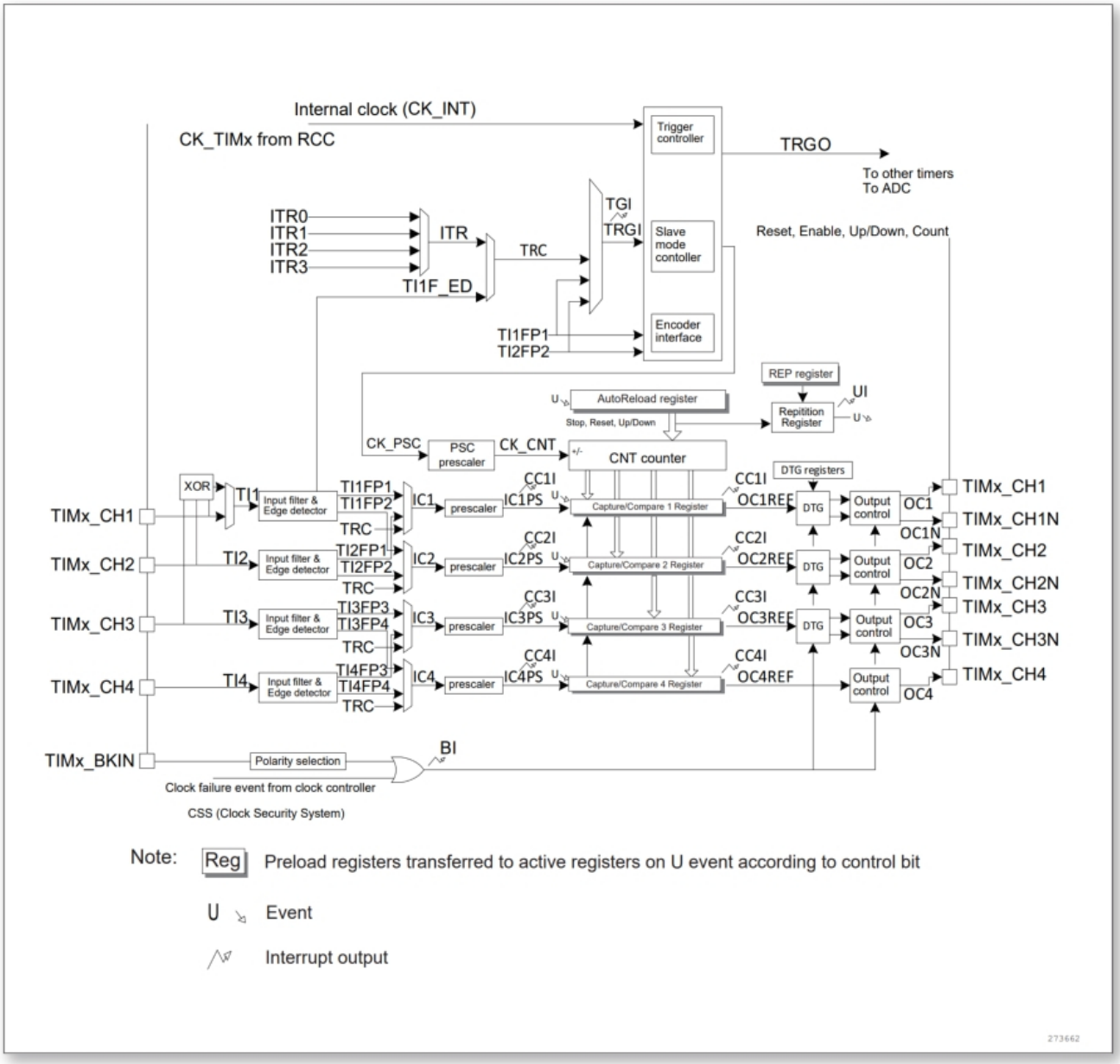


Figure 28. Advanced-control timer block diagram

9.3 Functional description

9.3.1 Time-base unit

The main block of the programmable advanced-control timer is a 16-bit counter with its related auto-reload register. The counter can count up, down or both up and down. The counter clock can be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)

(TIM1)

- Repetition counter register (TIMx_RCR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIMx_CR1 register. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software. The generation of the update event is described in detailed for each configuration.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIMx_CR1 register is set (refer also to the slave mode controller description to get more details on counter enabling).

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR1 register.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

The figures below give some examples of the counter behavior when the prescaler ratio is changed on the fly:

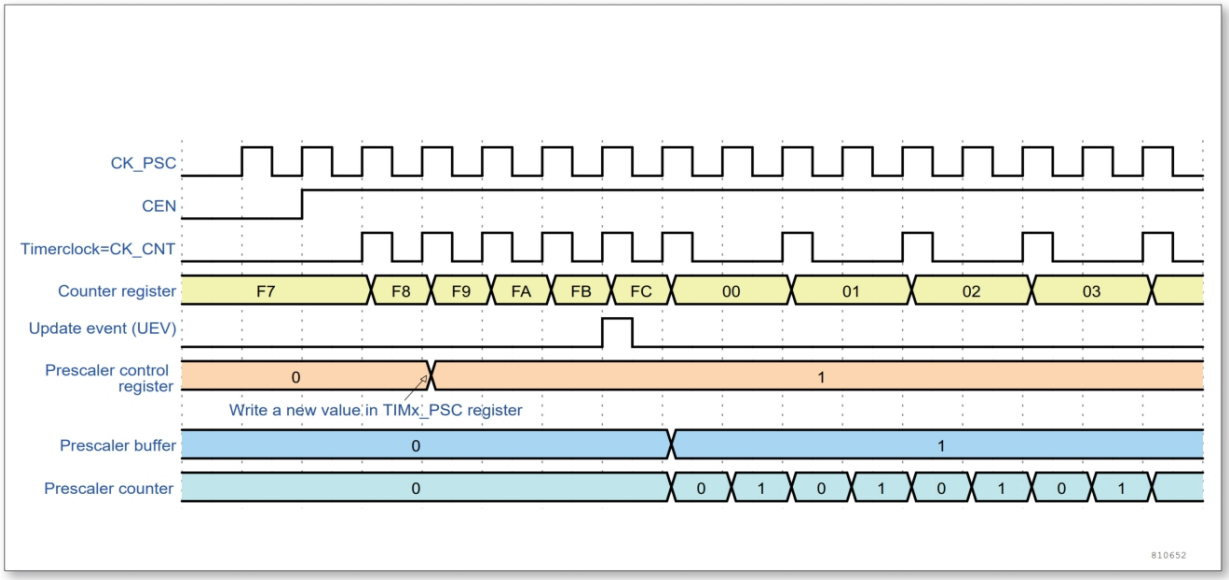


Figure 29. Counter timing diagram with prescaler division change from 1 to 2

(TIM1)

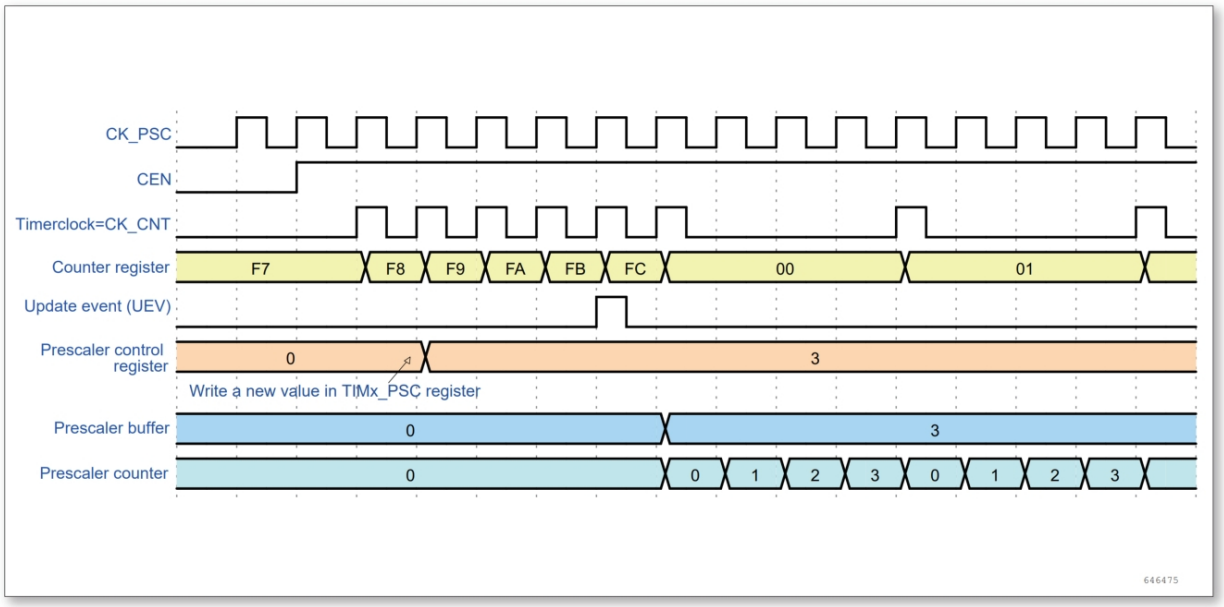


Figure 30.Counter timing diagram with prescaler division change from 1 to 4

9.3.2 Counter modes

Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the TIMx_ARR register), then restarts from 0 and generates a counter overflow event.

If the repetition counter is used, the update event (UEV) is generated after upcounting is repeated for the number of times programmed in the repetition counter register (TIMx_RCR). Else the update event is generated at each counter overflow.

Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV event can be disabled by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change) when an update event should be generated. In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag by hardware (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture mode.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set by hardware (depending on the URS bit).

- The repetition counter is reloaded with the content of TIMx_RCR register.
- The auto-reload shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR = 0x36.

(TIM1)

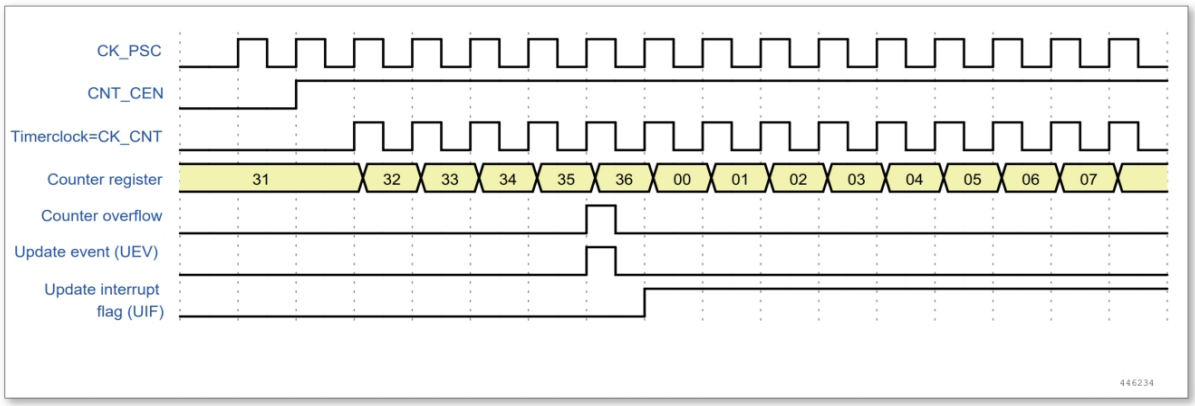


Figure 31. Counter timing diagram, internal clock divided by 1

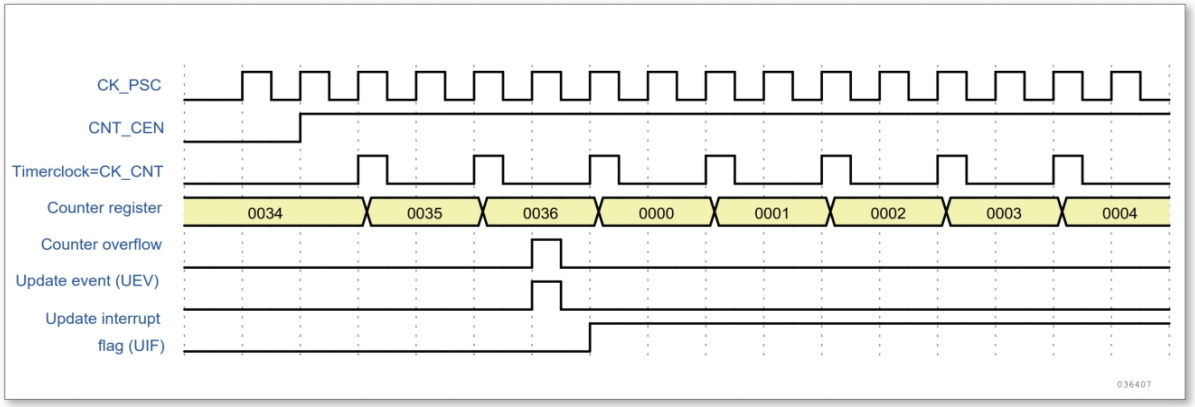


Figure 32. Counter timing diagram, internal clock divided by 2

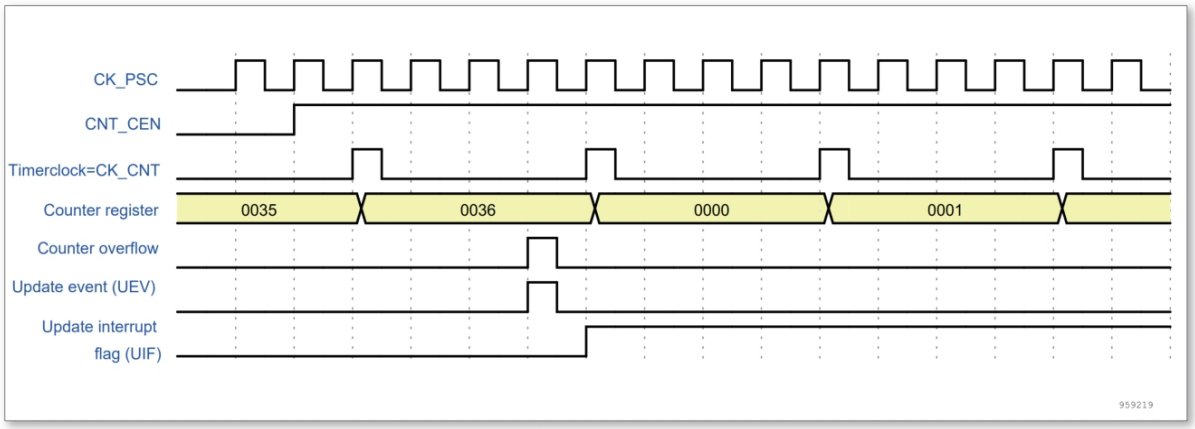


Figure 33. Counter timing diagram, internal clock divided by 4

(TIM1)

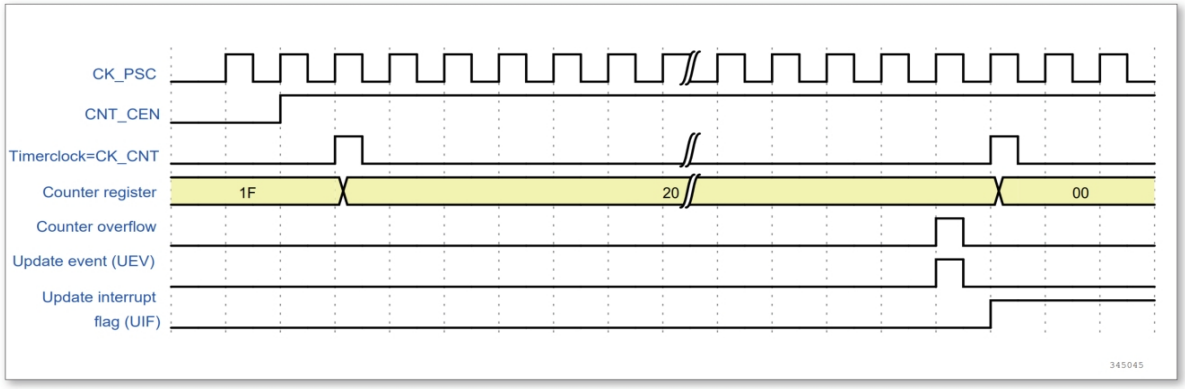


Figure 34. Counter timing diagram, internal clock divided by N

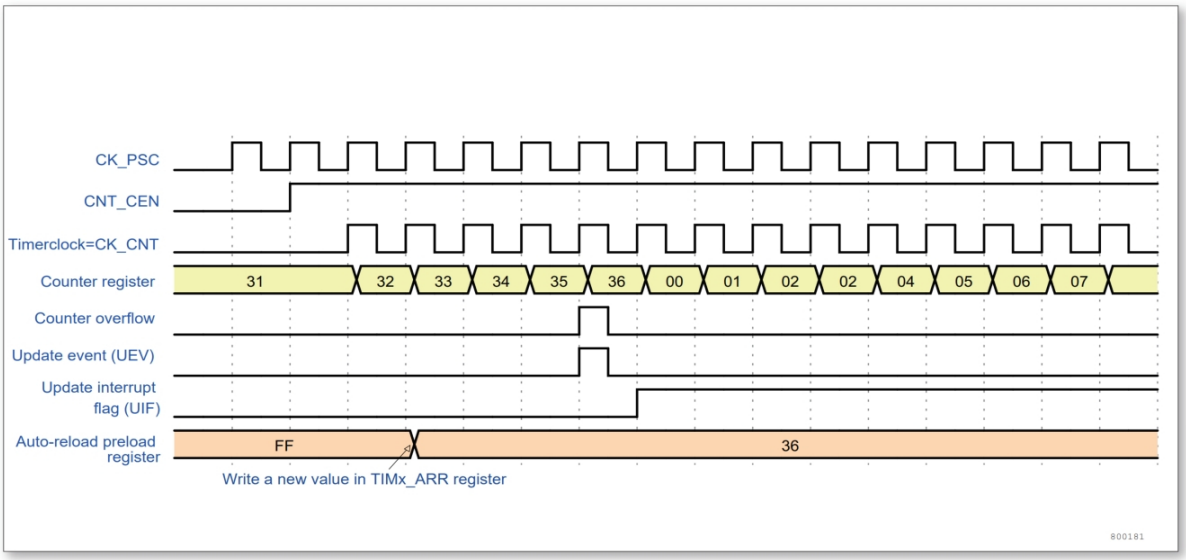
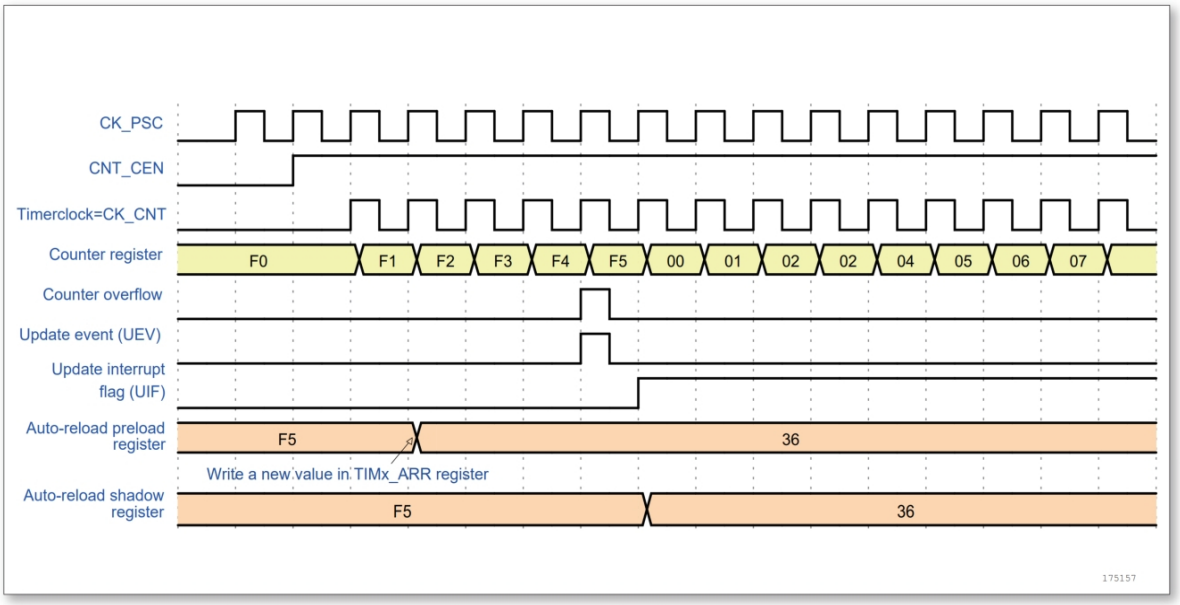


Figure 35. Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)



(TIM1)

Figure 36. Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

Downcounting mode

In downcounting mode, the counter counts from the auto-reload value (content of the TIMx_ARR register) down to 0, then restarts from the auto-reload value and generates a counter underflow event.

If the repetition counter is used, the update event (UEV) is generated after downcounting is repeated for the number of times programmed in the repetition counter register (TIMx_RCR). Else the update event is generated at each counter underflow.

Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV update event can be disabled by setting the UDIS bit in TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescale rate doesn't change).

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set

(depending on the URS bit).

- The repetition counter is reloaded with the content of TIMx_RCR register.
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIMx_ARR register).

Note that the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR = 0x36.

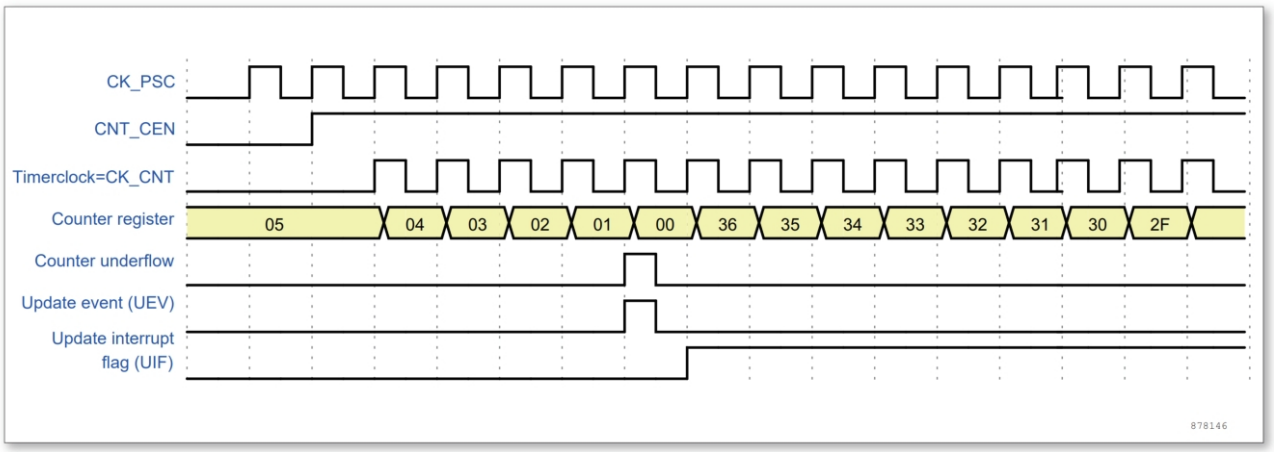


Figure 37. Counter timing diagram, internal clock divided by 1

(TIM1)

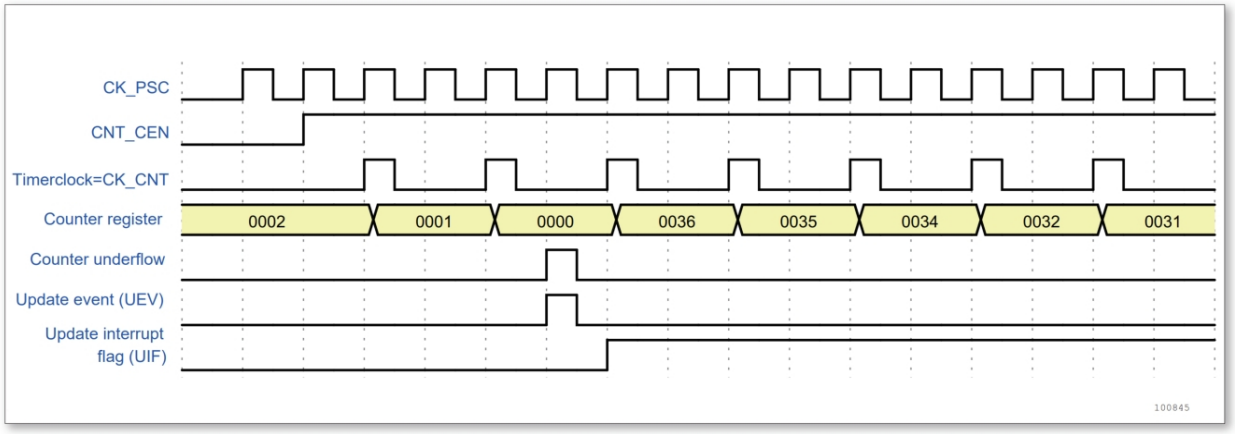


Figure 38. Counter timing diagram, internal clock divided by 2

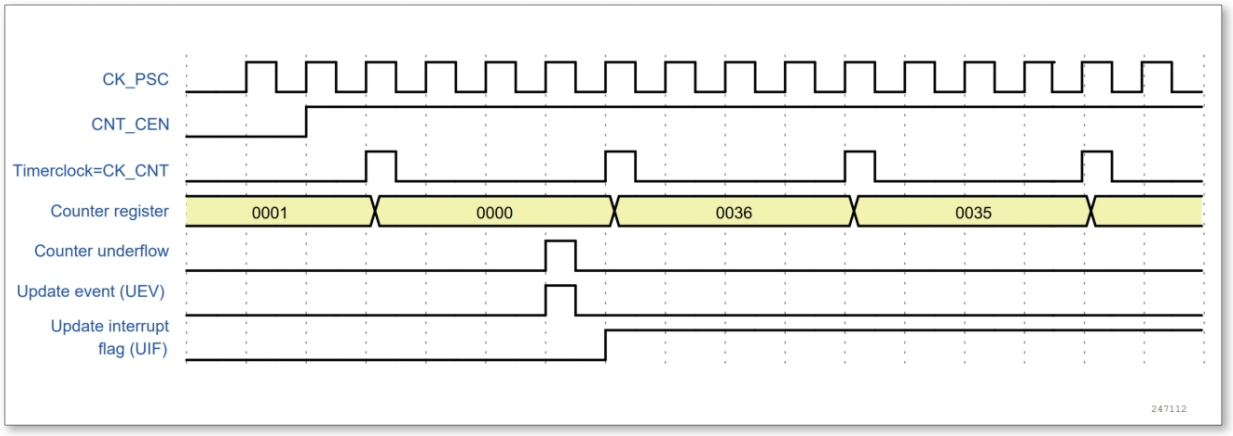


Figure 39. Counter timing diagram, internal clock divided by 4

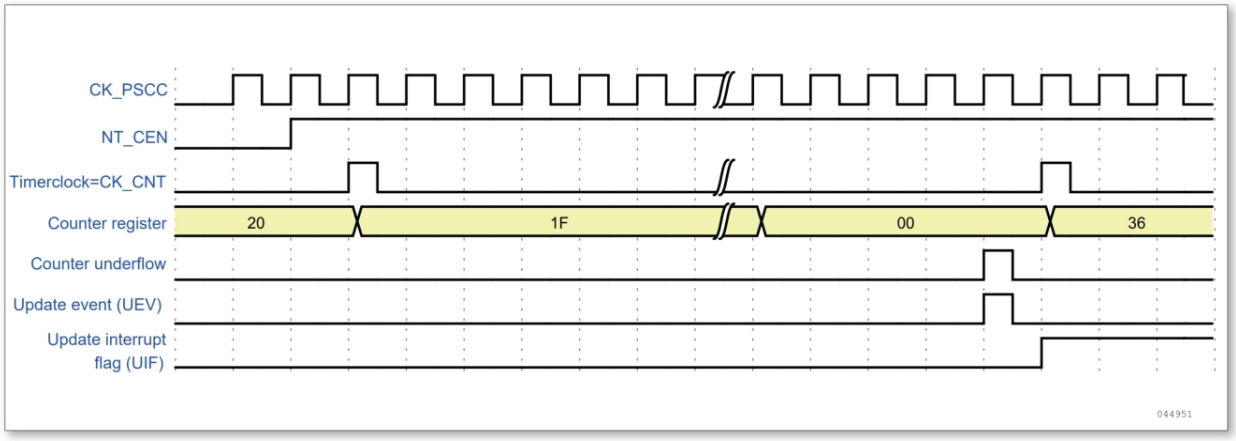


Figure 40. Counter timing diagram, internal clock divided by N

(TIM1)

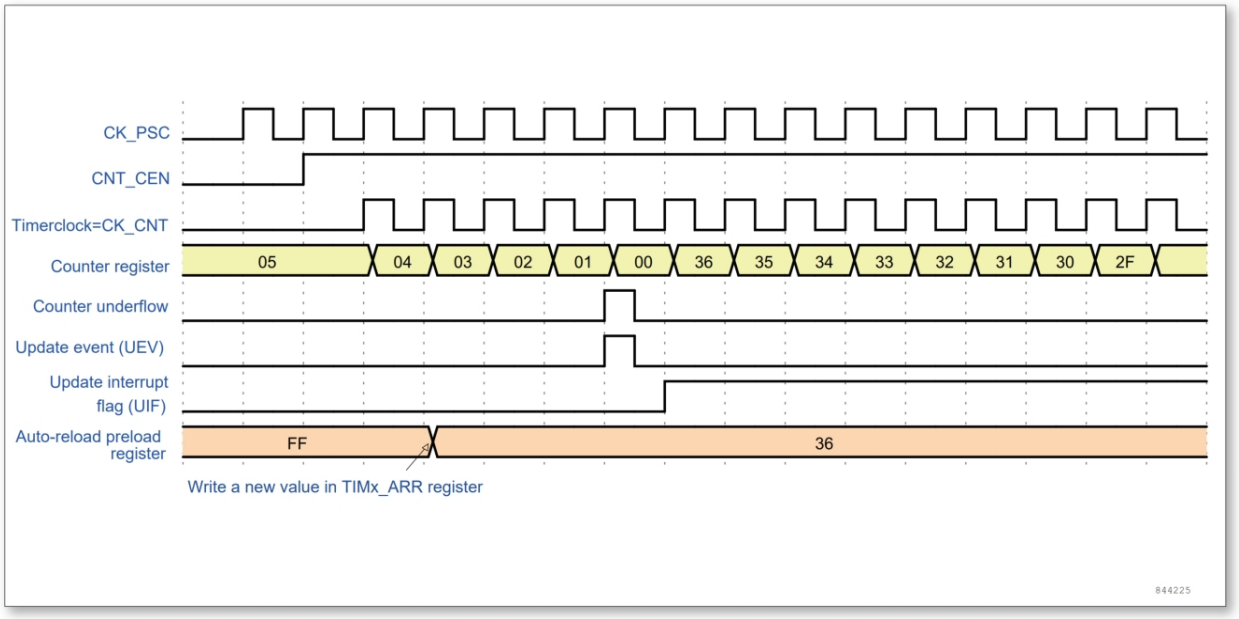


Figure 41. Counter timing diagram, update event when repetition counter is not used

Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the TIMx_ARR register) - 1, generates a counter overflow event, then counts from the auto-reload value down to 1 and generates a counter underflow event. Then it restarts counting from 0.

In this mode, the DIR direction bit in the TIMx_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter.

The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller). In this case, the counter restarts counting from 0, as well as the counter of the prescaler.

The UEV update event can be disabled by setting the UDIS bit in TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter continues counting up and down, based on the current auto-reload value.

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit).

- The repetition counter is reloaded with the content of TIMx_RCR register.
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIMx_ARR register).

Note that if the update source is a counter overflow, the auto-reload is updated before the counter is reloaded, so that the next period is the expected one (the counter is loaded with the new value).

The following figures show some examples of the counter behavior for different clock frequencies:

(TIM1)

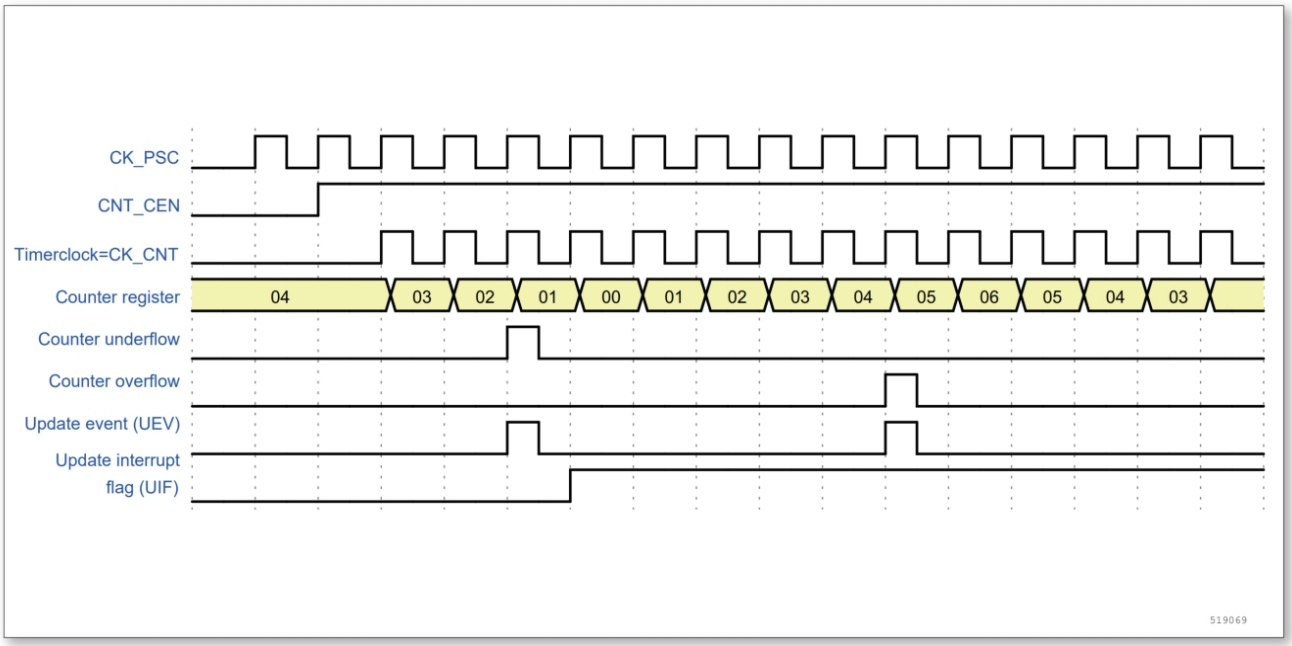


Figure 42. Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6

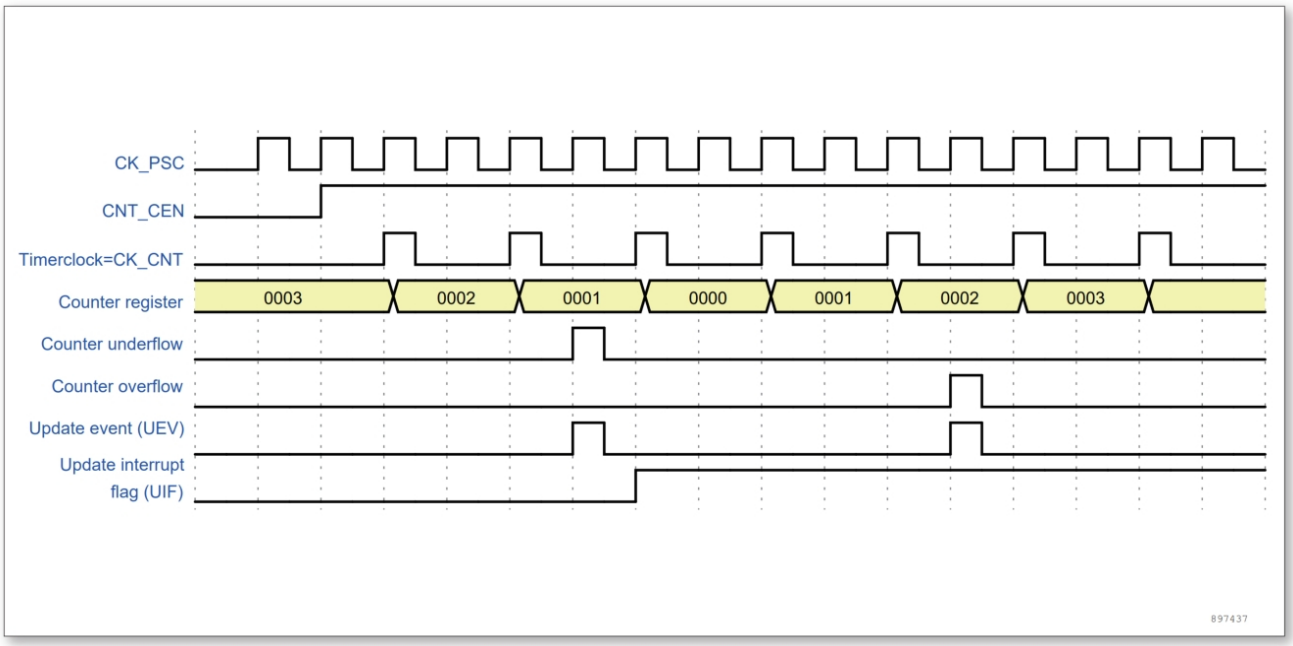


Figure 43. Counter timing diagram, internal clock divided by 2

(TIM1)

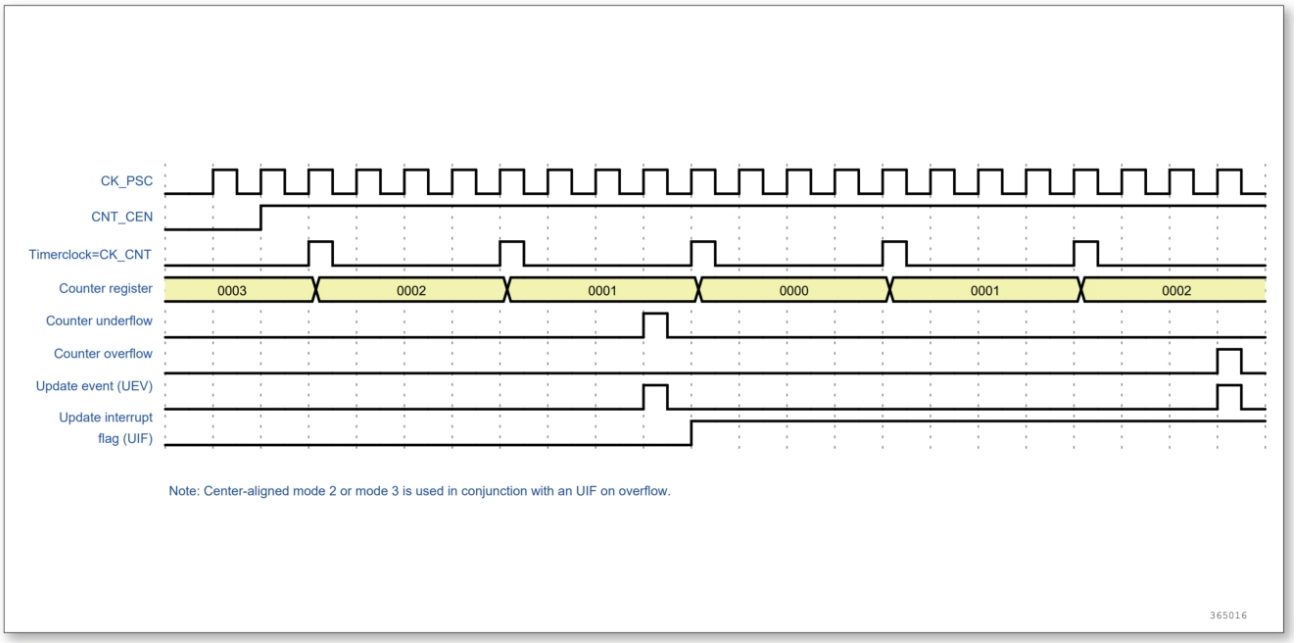


Figure 44. Counter timing diagram, internal clock divided by 4, TIMx_ARR=0x03

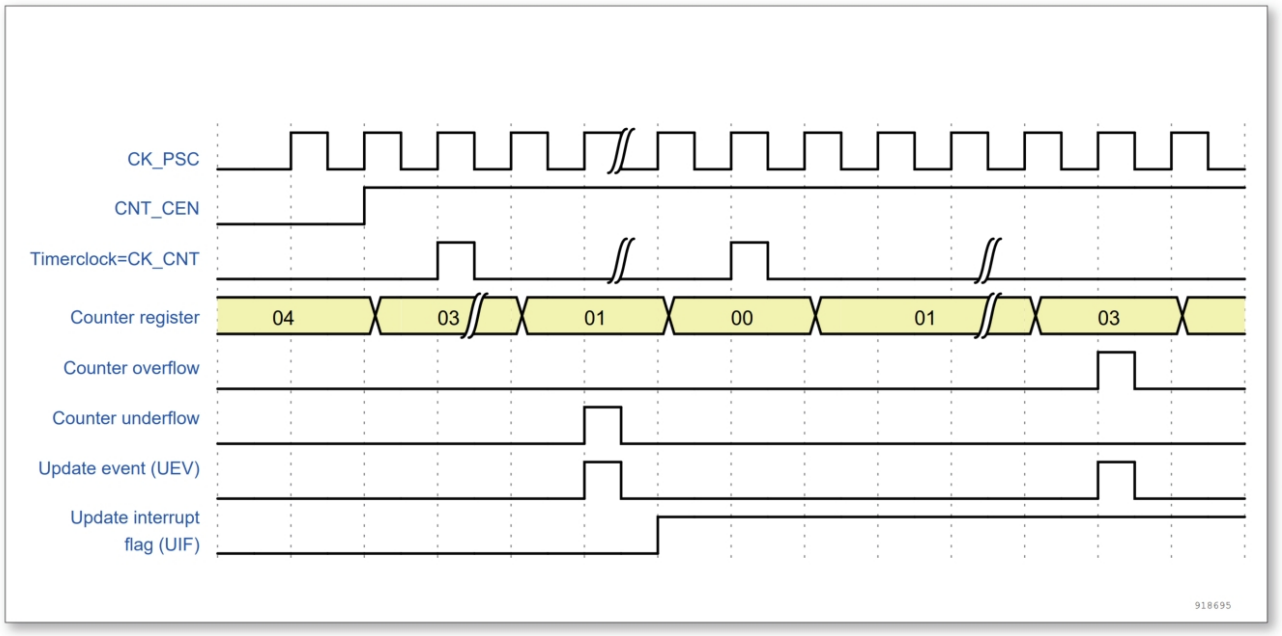


Figure 45. Counter timing diagram, internal clock divided by N

(TIM1)

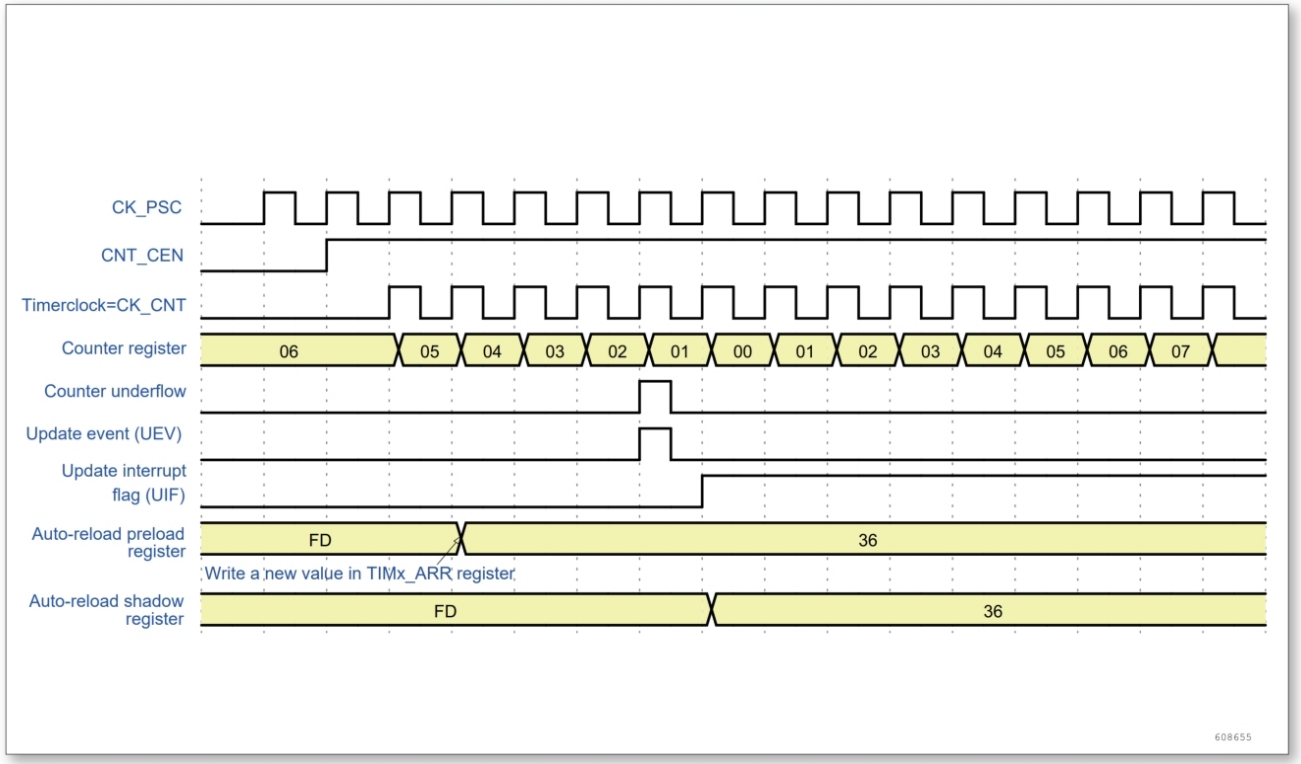


Figure 46. Counter timing diagram, update event with ARPE=1 (counter underflow)

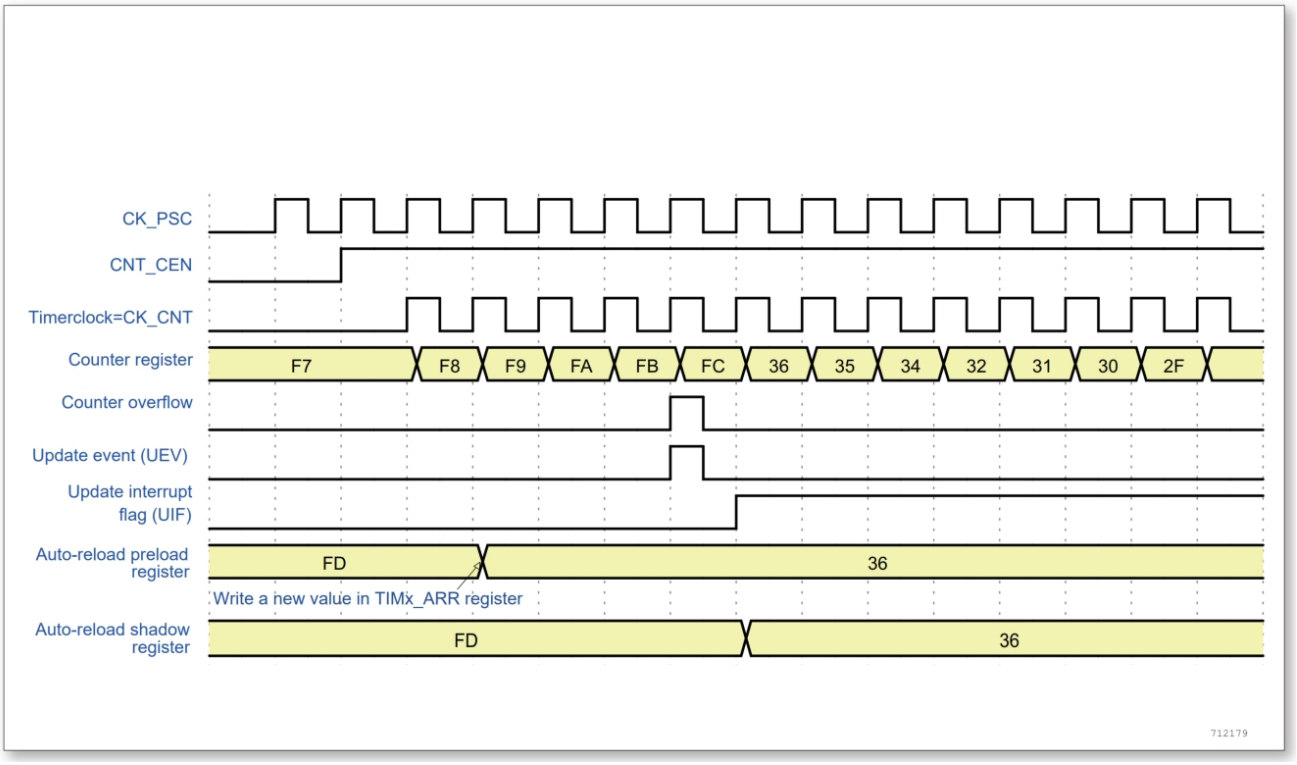


Figure 47. Counter timing diagram, update event with ARPE=1 (counter overflow)

9.3.3 Repetition counter

'Time-base unit' describes how the update event (UEV) is generated with respect to the counter overflows/underflows. It is actually generated only when the repetition counter has reached zero. This can be useful when generating PWM signals.

(TIM1)

This means that data are transferred from the preload registers to the shadow registers (TIMx_ARR auto-reload register, TIMx_PSC preload register, but also TIMx_CCRx capture/compare registers in compare mode) every N counter overflows or underflows, where N is the value in the TIMx_RCR repetition counter register.

The repetition counter is decremented:

- At each counter overflow in upcounting mode,
- At each counter underflow in downcounting mode,
- At each counter overflow and at each counter underflow in center-aligned mode. Although this limits the maximum number of repetition to 128 PWM cycles, it makes it possible to update the duty cycle twice per PWM period. When refreshing compare registers only once per PWM period in center-aligned mode, maximum resolution is 2xTck, due to the symmetry of the pattern.

The repetition counter is an auto-reload type; the repetition rate is maintained as defined by the TIMx_RCR register value (refer to Figure 48). When the update event is generated by software (by setting the UG bit in TIMx_EGR register) or by hardware through the slave mode controller, it occurs immediately whatever the value of the repetition counter is and the repetition counter is reloaded with the content of the TIMx_RCR register.

The value of the repetition counter is written to a new REP_CNT register (Repetition counter value of real-time writing) in real time. This is used in the repetition counter modification mode, to move the update interrupt flag (UIF) to left by (REP-REP_CNT) phases (to right when the subtracted value is negative) by shifting UIF detection point in real time. These bits should be written after the update event UG is generated (note writing to REP_CNT before the update event is generated makes the displacement invalid).

(TIM1)

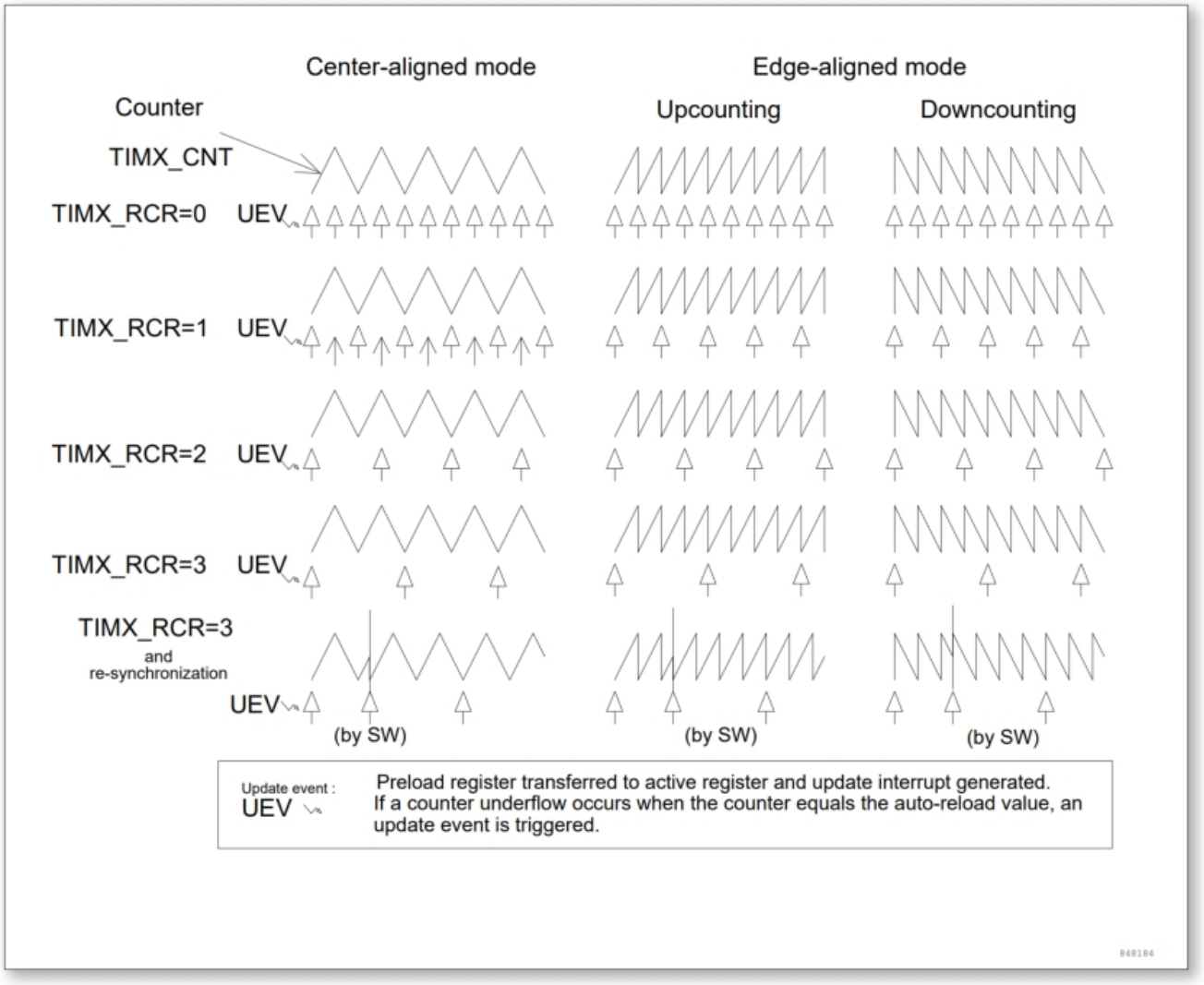


Figure 48. Update rate examples depending on mode and TIMx_RCR register settings

9.3.4 Clock selection

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT).
- External clock mode 1: external input pin (TIx).
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, the user can configure Timer 1 to act as a prescaler for Timer 2.

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS=000), then the CEN, DIR (in the TIMx_CR1 register) and UG bits (in the TIMx_EGR register) are actual control bits and can be changed only by software (except UG which remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

The figure below shows the behavior of the control circuit and the upcounter in normal mode, without prescaler.

(TIM1)

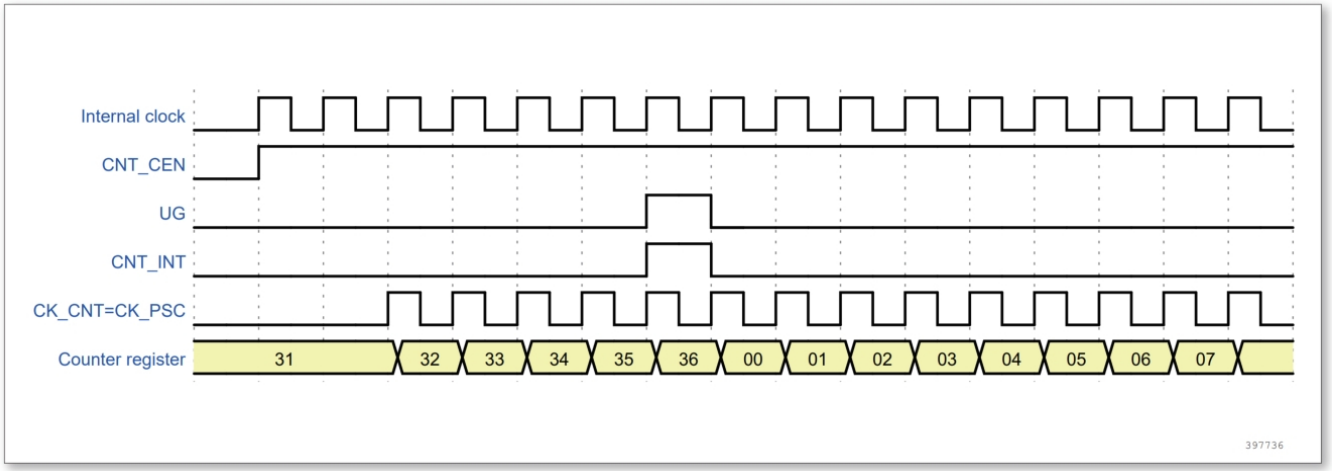


Figure 49. Control circuit in normal mode, internal clock divided by 1

External clock source mode 1

This mode is selected when SMS=111 in the TIMx_SMCR register. The counter can count at each rising or falling edge on a selected input.

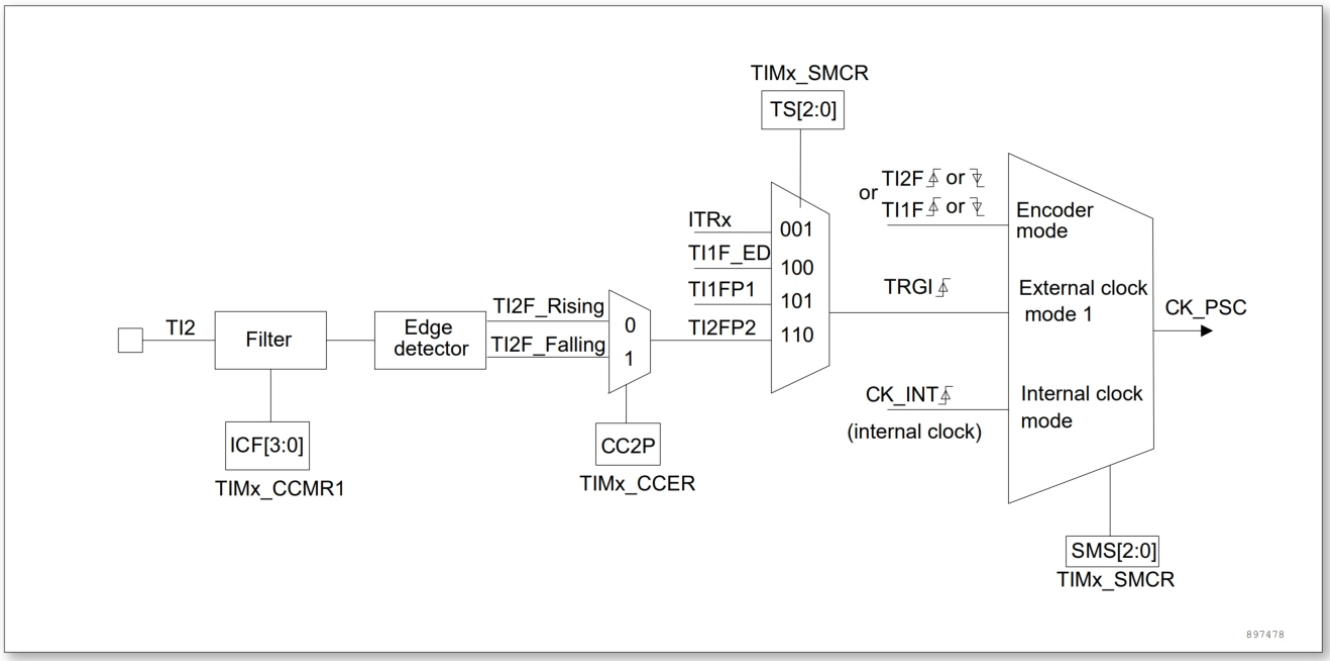


Figure 50. TI2 external clock connection example

For example, to configure the upcounter to count in response to a rising edge on the TI2 input, use the following procedure:

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = 01 in the TIMx_CCMR1 register.
2. Configure the input filter duration by writing the IC2F[3:0] bits in the TIMx_CCMR1 register (if no filter is needed, keep IC2F=0000).
3. Select rising edge polarity by writing CC2P = 0 in the TIMx_CCER register.
4. Configure the timer in external clock mode 1 by writing SMS = 111 in the TIMx_SMCR register.
5. Select TI2 as the trigger input source by writing TS = 110 in the TIMx_SMCR register.
6. Enable the counter by writing CEN = 1 in the TIMx_CR1 register.

Note: The capture prescaler is not used for triggering, so the user does not need to configure it.

(TIM1)

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set. The delay between the rising edge on TI2 and the actual clock of the counter is due to the resynchronization circuit on TI2 input.

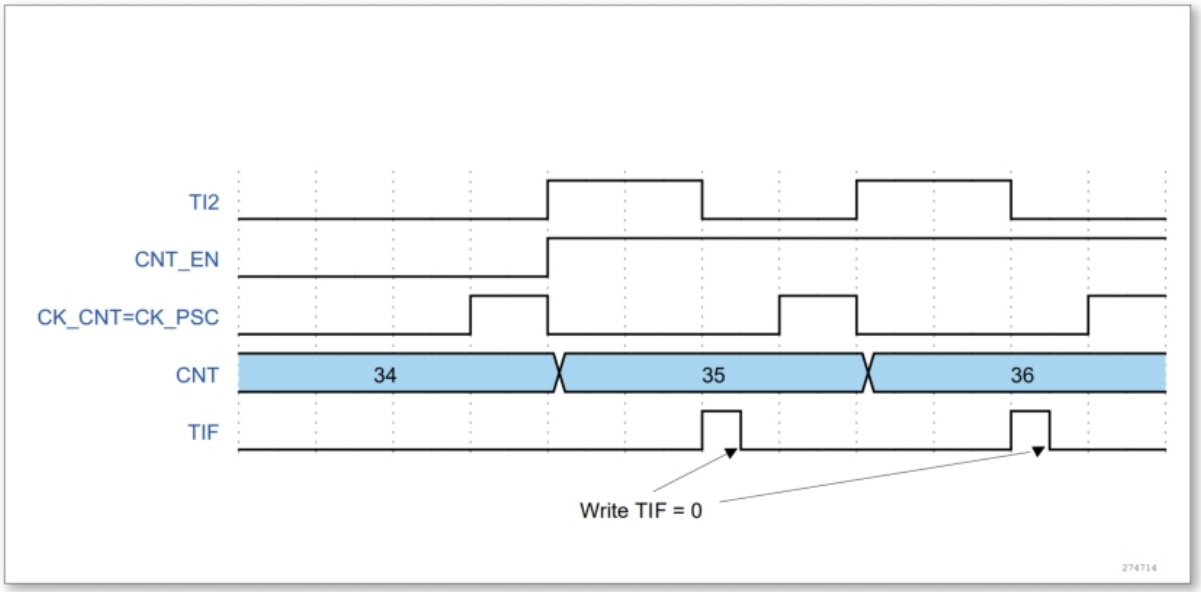


Figure 51. Control circuit in external clock mode 1

9.3.5 Capture/compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), a input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

Figure 52 to Figure 55 give an overview of one Capture/Compare channel.

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

(TIM1)

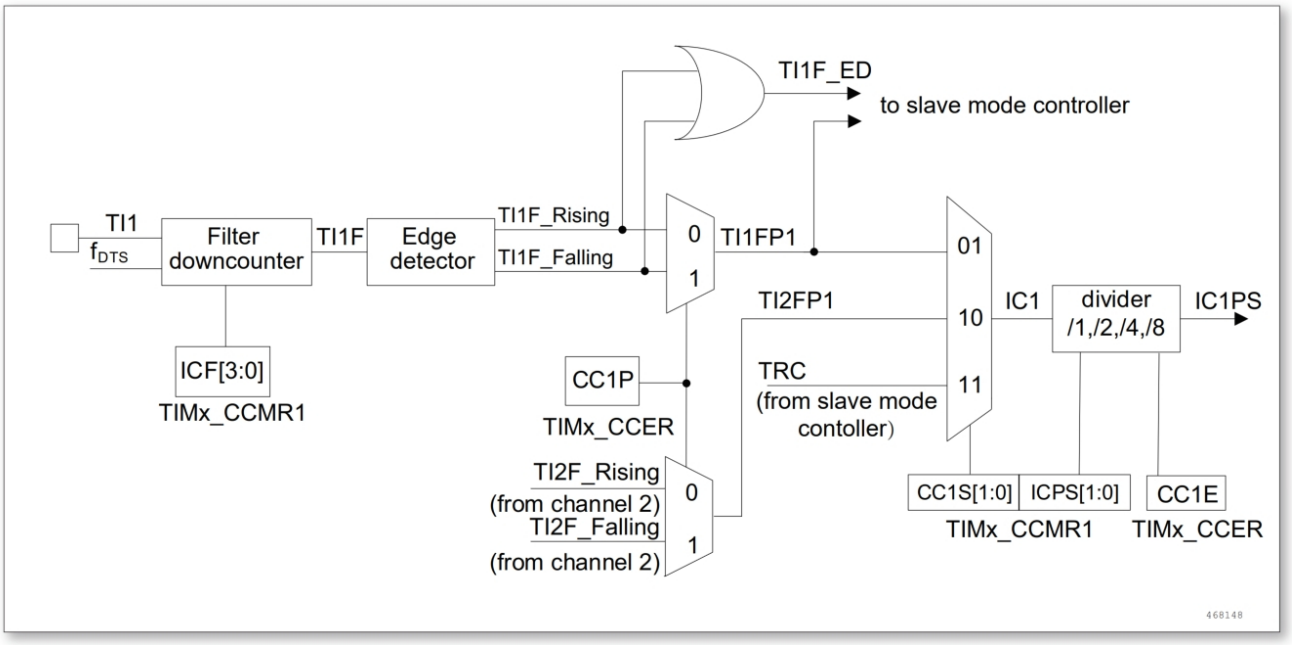


Figure 52. Capture/compare channel (example: channel 1 input stage)

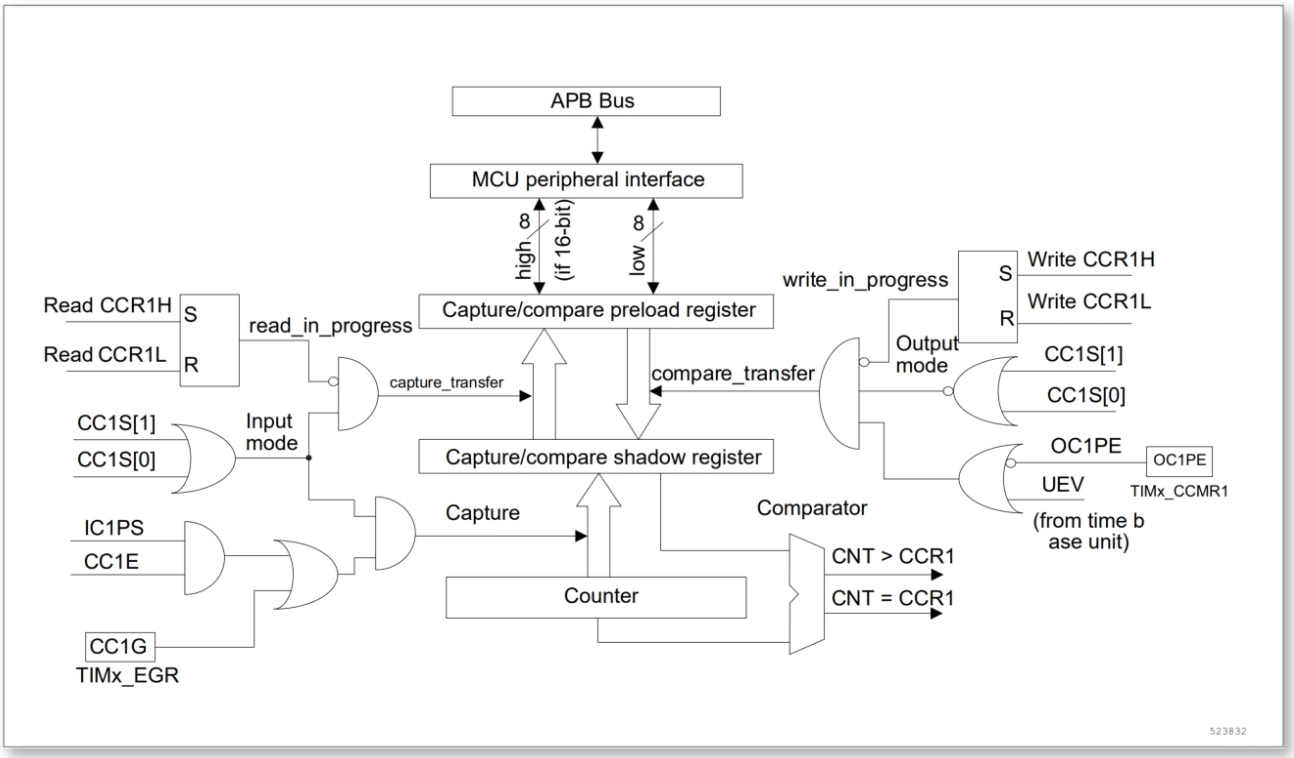


Figure 53. Capture/compare channel 1 main circuit

(TIM1)

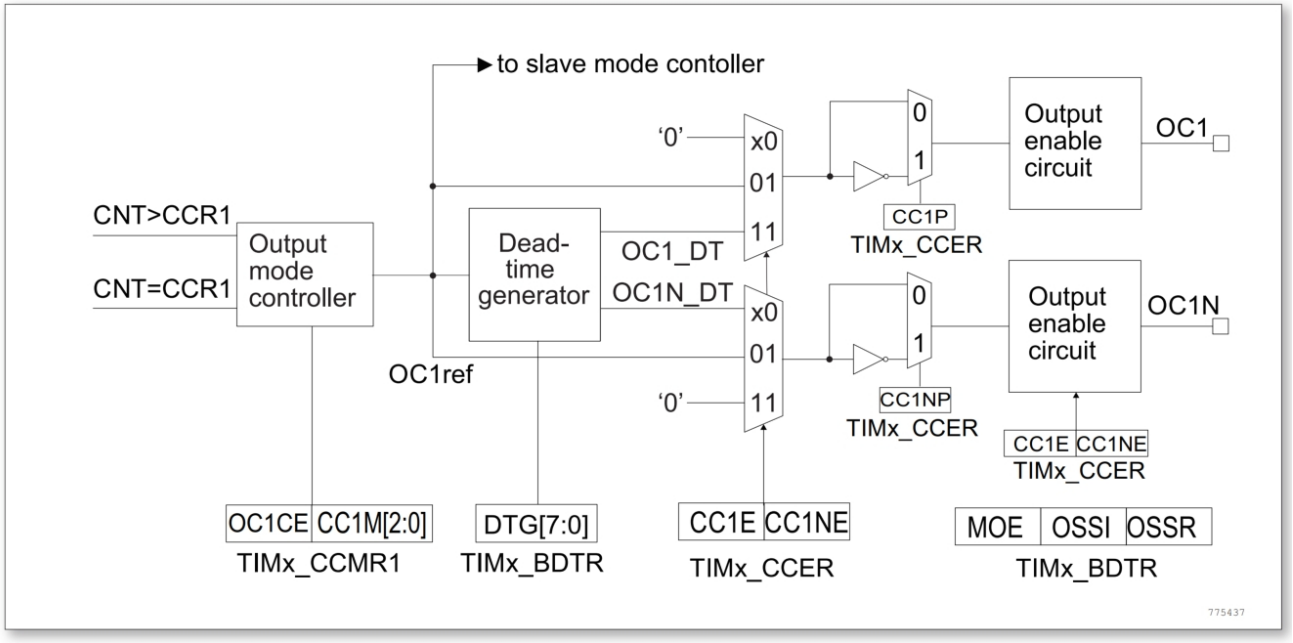


Figure 54. Capture/compare channel 1 main circuit

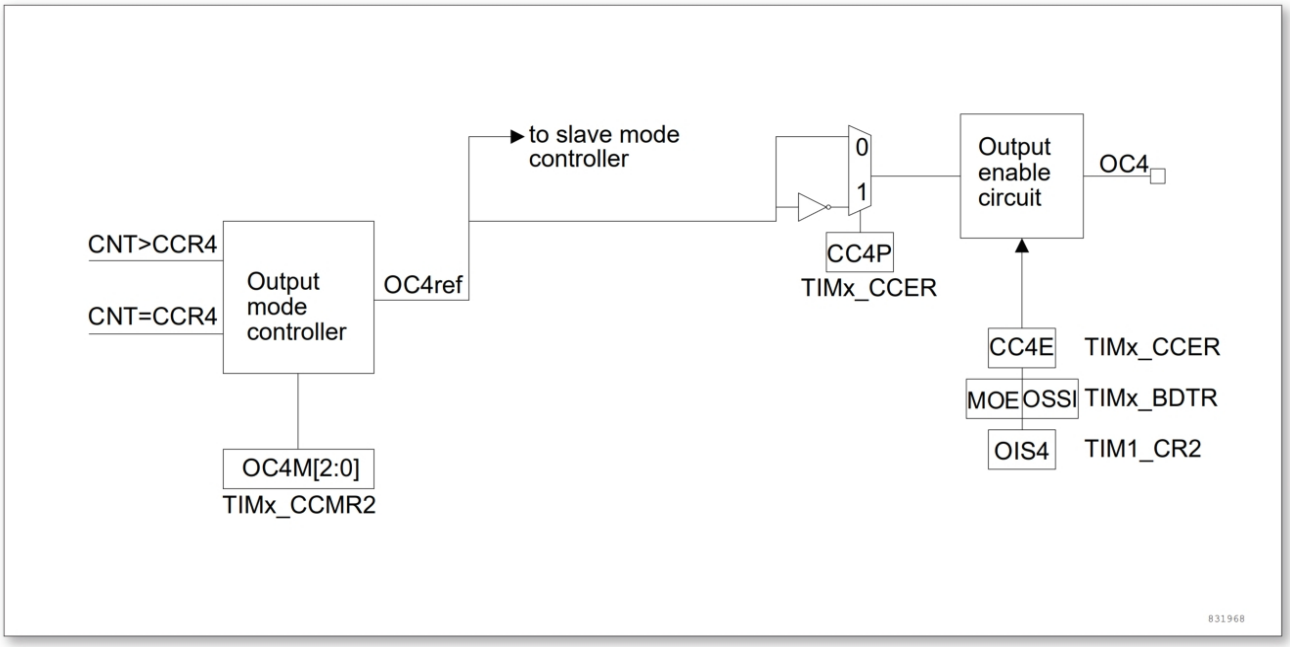


Figure 55. Output stage of capture/compare channel (channel 4)

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register. In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

9.3.6 Input capture mode

In Input capture mode, the Capture/Compare registers (TIMx_CCRx) are used to latch the value of the counter after an edge is detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIMx_SR register) is set to '1' and an interrupt can be sent if it is enabled. If a capture occurs while the CCxIF flag was already

(TIM1)

high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when written to '0'. The following example shows how to capture the counter value in TIMx_CCR1 when TI1 input rises. To do this, use the following procedure:

- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIMx_CCR1 register becomes read-only.
- Program the needed input filter duration with respect to the input signal (by programming ICxF bits in the TIMx_CCMRx register if the input is a TIx input). Let's imagine that, when toggling, the input signal is not stable during at most five internal clock cycles. We must program a filter duration longer than these five clock cycles. We can validate an edge transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency).

  Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIMx_CCER register (rising edge in this case).
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit to '1' in the TIMx_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active level transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set to '1' if at least two consecutive captures occurred whereas the

  CC1IF flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

9.3.7 PWM input mode

This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode. For example, user can measure the period (in TIMx_CCR1 register) and the duty cycle (in TIMx_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):
- Select the active input for TIMx_CCR1: write the CC1S bits to 01 in the TIMx_CCMR1 register (TI1 selected).

(TIM1)

- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to ‘0’ (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2P bit to ‘1’ (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMx_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to ‘1’ in the TIMx_CCER register.

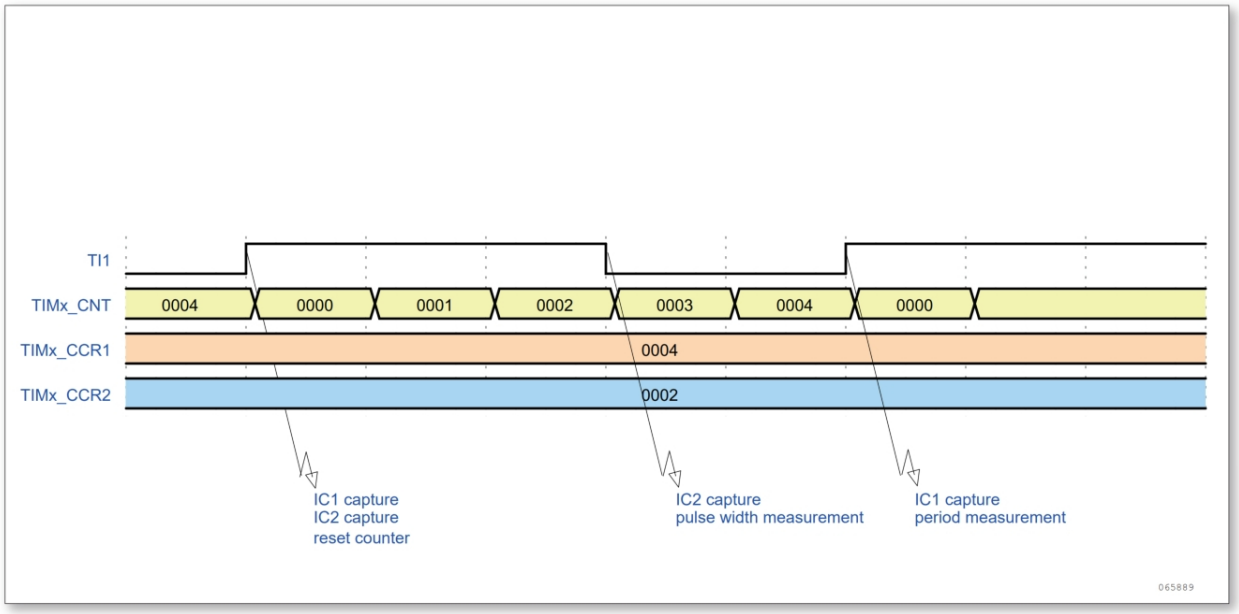


Figure 56. PWM input mode timing

The PWM input mode can be used only with the TIMx_CH1/TIMx_CH2 signals due to the fact that only TI1FP1 and TI2FP2 are connected to the slave mode controller.

9.3.8 Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compare signal (OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, the user just needs to write 101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx gets opposite value to CCxP polarity bit.

For example: CCxP = 0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bit to 100 in the TIMx_CCMRx register.

Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt requests can be sent accordingly. This is described in the output compare mode section below.

(TIM1)

9.3.9 Output compare mode

This function is used to control an output waveform or indicate when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bit in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCxM = 000), be set active (OCxM = 001), be set inactive (OCxM = 010) or can toggle (OCxM = 011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCxIE bit in the TIMx_DIER register).

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output.

The synchronization resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

The output compare mode is configured as follows:

- Select the counter clock (internal, external, prescaler).
- Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
- Set the CCxIE bit if an interrupt request is to be generated.
- Select the output mode. For example:
  - Write OCxM = 011 to toggle OCx output pin when CNT matches CCRx
  - Write OCxPE = 0 to disable preload register
  - Write CCxP = 0 to select active high polarity
  - Write CCxE = 1 to enable the output
- Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE=’0’, else TIMx_CCRx shadow register is updated only at the next update event UEV). An example is given in the figure below.

(TIM1)

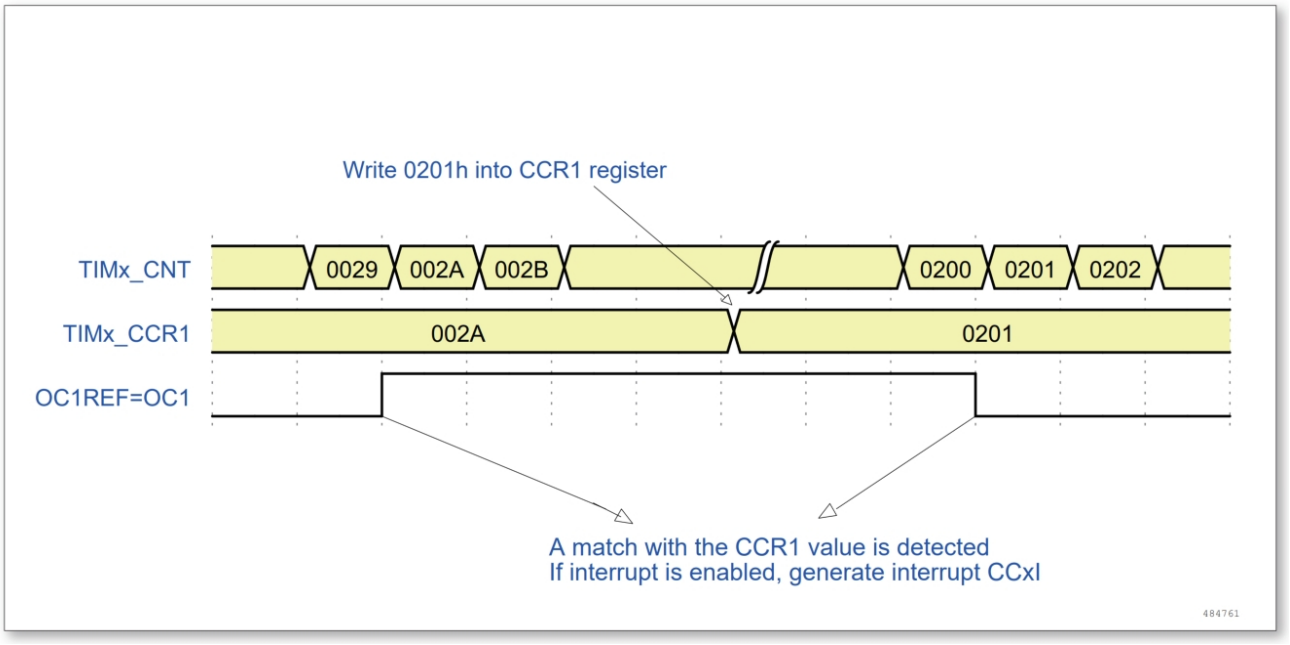


Figure 57. Output compare mode, toggle on OC1

9.3.10 PWM mode

Pulse Width Modulation mode allows generating a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bit in the TIMx_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMx_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, the user must initialize all the registers by setting the UG bit in the TIMx_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. OCx output is enabled by a combination of the CCxE, CCxNE, MOE, OSSI and OSSR bits (TIMx_CCER and TIMx_BDTR registers). Refer to the TIMx_CCER register description for more details. In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared to determine whether TIMx_CCRx $\leqslant$ TIMx_CNT or TIMx_CNT $\leqslant$ TIMx_CCRx (depending on the direction of the counter). The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIMx_CR1 register.

PWM edge-aligned mode

Upcounting configuration

Upcounting is active when the DIR bit in the TIMx_CR1 register is low. Refer to section 9.3.2.

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx; else it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR), then OCxREF is held at '1'. If the compare value is 0 then OCxRef is held at '0'. Figure 129 shows some edge-aligned PWM waveforms in an example where TIMx_ARR = 8.

(TIM1)

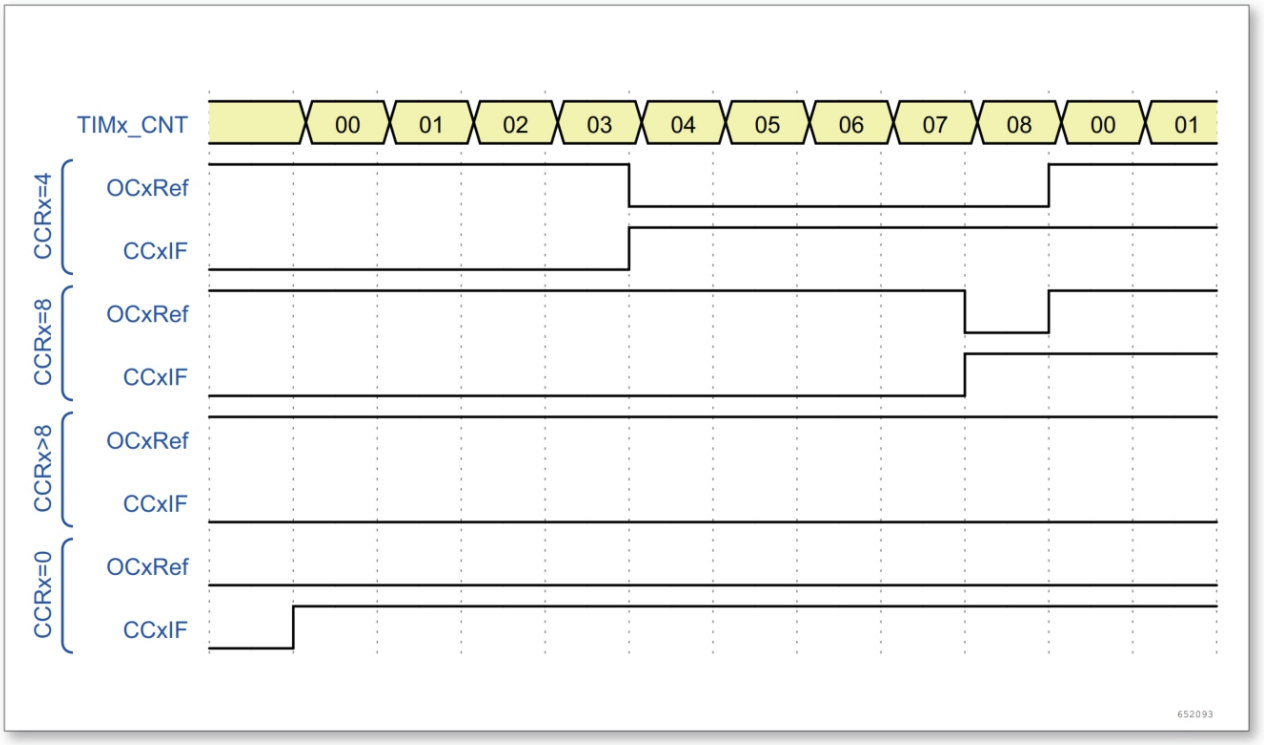


Figure 58. Edge-aligned PWM waveforms (ARR = 8)

Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high. Refer to Section 9.3.2. In PWM mode 1, the reference signal OCxRef is low as long as TIMx_CNT > TIMx_CCRx; else it becomes high. If the compare value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, then OCxREF is held at '1'.

0% PWM is not possible in this mode.

PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are different from '00' (all the remaining configurations having the same effect on the OCxRef/OCx signals). The compare flag is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the TIMx_CR1 register is updated by hardware and must not be changed by software. Refer to Center-aligned mode in Section 9.3.2.

The figure below shows some center-aligned PWM waveforms in an example where:

- TIMx_ARR = 8
- PWM mode is the PWM mode 1
- The flag is set when the counter counts down corresponding to the center-aligned mode 1 selected for CMS = 01 in TIMx_CR1 register.

(TIM1)

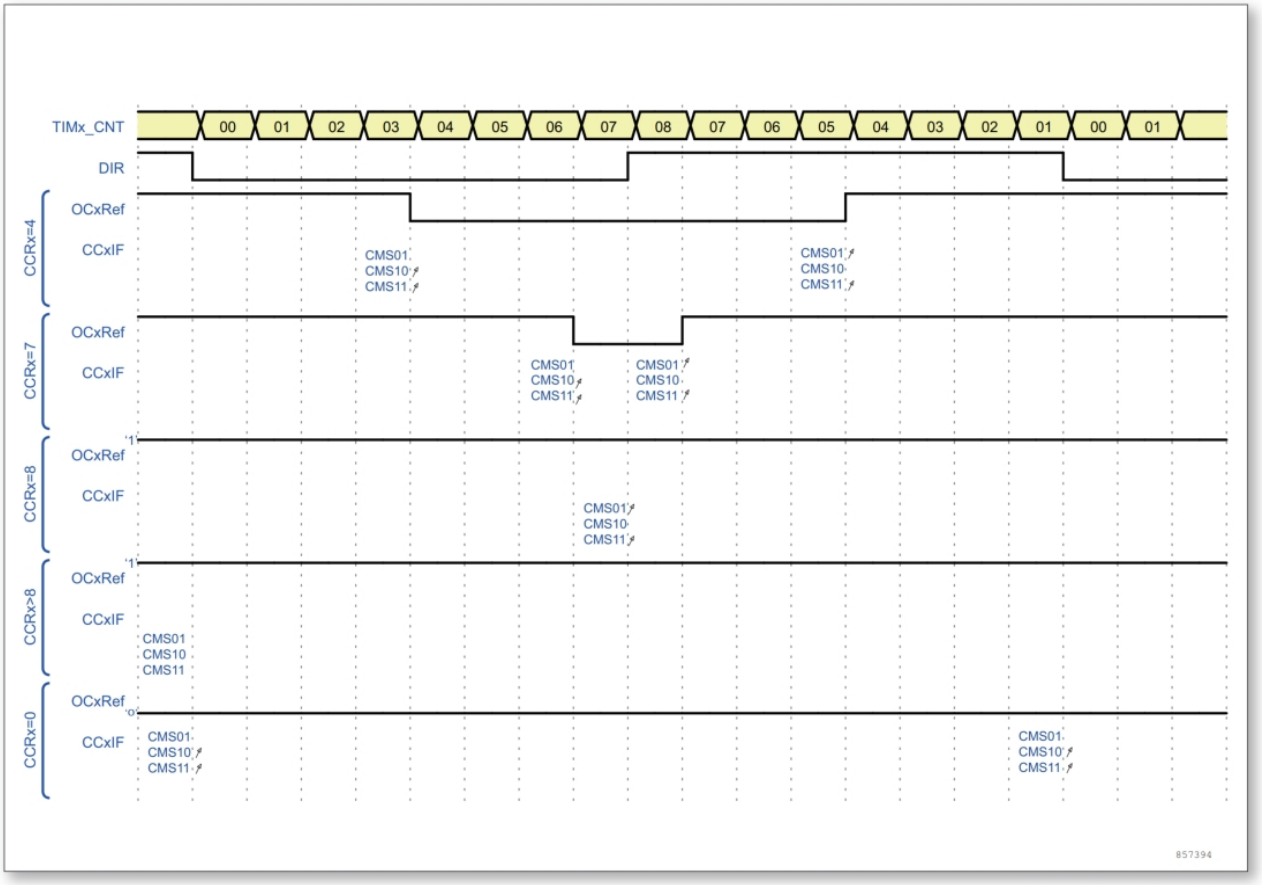


Figure 59. Center-aligned PWM waveforms (ARR = 8)

Phase shift in the PWM center-aligned mode

PDER (channel x output PWM phase shift enable bit) and CCRxFALL (channel x capture/compare value when downcounting in the PWM center-aligned mode) are added to allow 5 channels to output PWM phase shift. To realize PWM outputs with programmable phase-shift waveform (left shift or right shift as required), the user should enable the PWM phase-shift function in the PDER register and set the CCRxFALL and CCRx.

(TIM1)

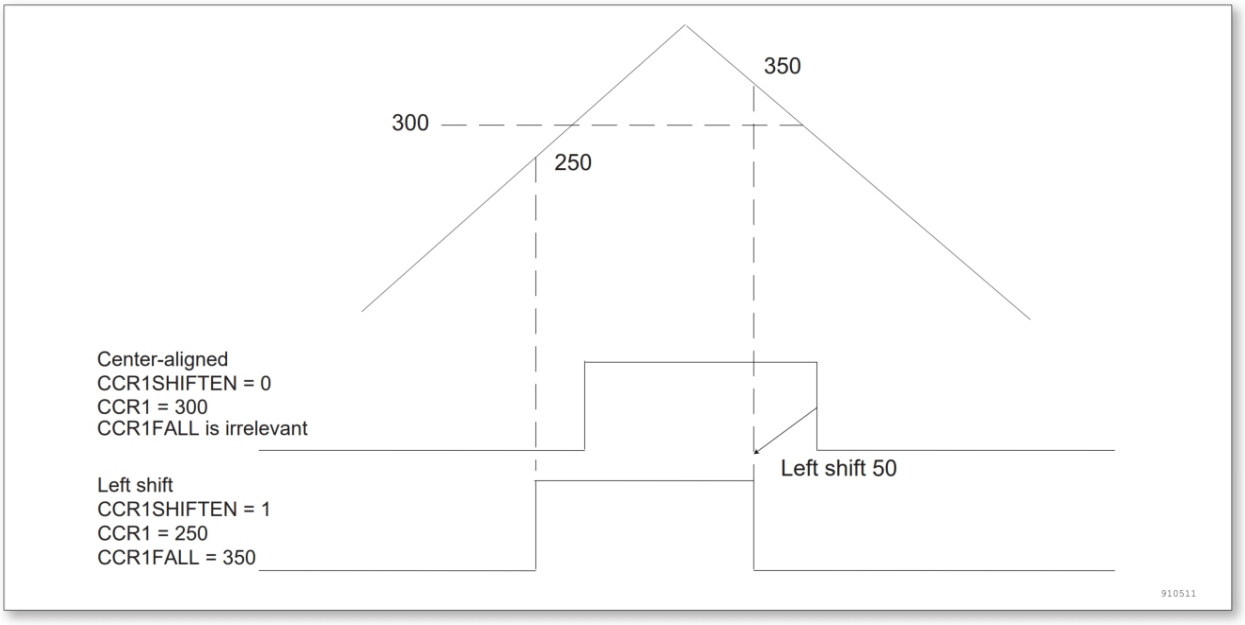


Figure 60. Phase shift diagram

Hints on using center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIMx_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:
  - The direction is not updated if the user writes a value in the counter greater than the auto-reload value (TIMx_CNT > TIMx_ARR).
  - For example, if the counter was counting up, it will continue to count up.
  - The direction is updated if the user writes 0 or writes the TIMx_ARR value in the counter but no Update Event UEV is generated.
- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIMx_EGR register) just before starting the counter and not to write the counter while it is running.

9.3.11 Complementary outputs and dead-time insertion

The advanced-control timer (TIM1) can output two complementary signals and manage the switching-off and the switching-on instants of the outputs. This time is generally known as dead-time and it has to be adjusted depending on the devices connected to the outputs and their characteristics (intrinsic delays of level-shifters, delays due to power switches...)

User can select the polarity of the outputs (main output OCx or complementary OCxN) independently for each output. This is done by writing to the CCxP and CCxNP bits in the TIMx_CCER register.

The complementary signals OCx and OCxN are activated by a combination of several control bits: the CCxE and CCxNE bits in the TIMx_CCER register and the MOE, OISx, OISxN, OSSI and OSSR bits in the TIMx_BDTR and TIMx_CR2 registers. For more details, refer to Table 35: Output control bits for complementary OCx and OCxN

(TIM1)

channels with break feature. In particular, the dead-time is activated when switching to the IDLE state (MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. There is a 10-bit dead-time generator in each channel. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

- The OCx output signal is the same as the reference signal except for the rising edge, which is delayed relative to the reference rising edge.
- The OCxN output signal is the opposite of the reference signal except for the rising edge, which is delayed relative to the reference falling edge.

  If the delay is greater than the width of the active output (OCx or OCxN) then the corresponding pulse is not generated.

The following figures show the relationships between the output signals of the dead-time generator and the reference signal OCxREF. (we suppose CCxP = 0, CCxNP = 0, MOE = 1, CCxE = 1 and CCxNE = 1 in these examples)

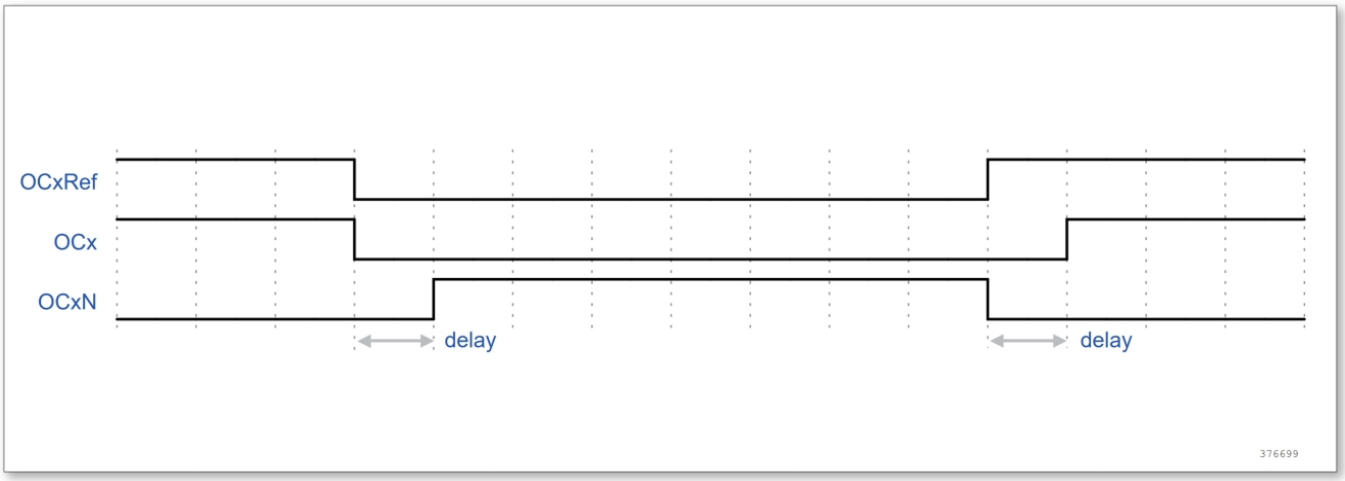


Figure 61. Complementary output with dead-time insertion

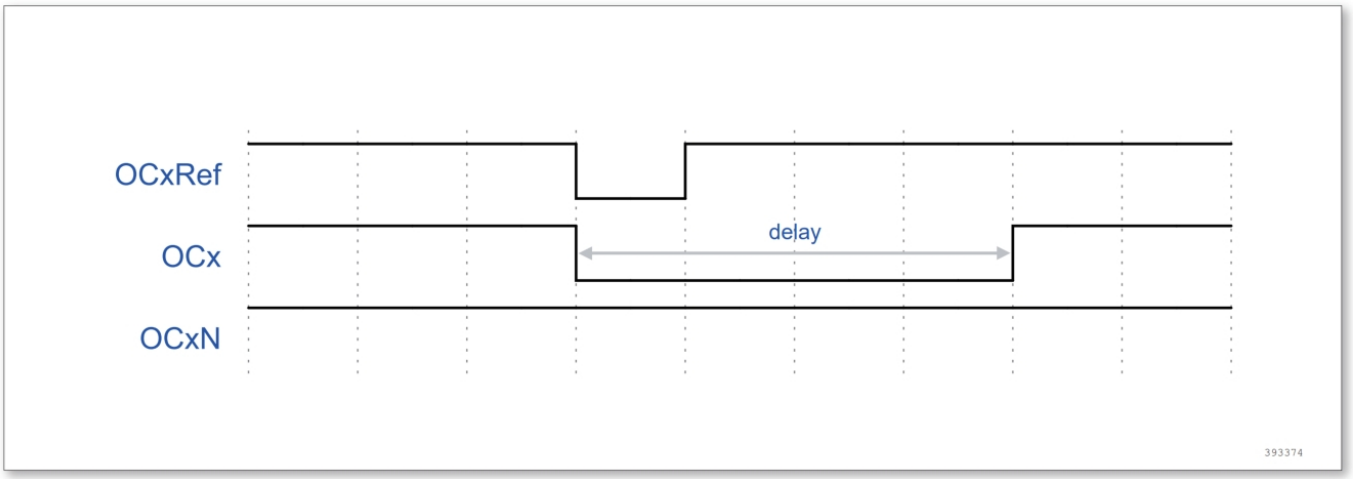


Figure 62. Dead-time waveforms with delay greater than the negative pulse

(TIM1)

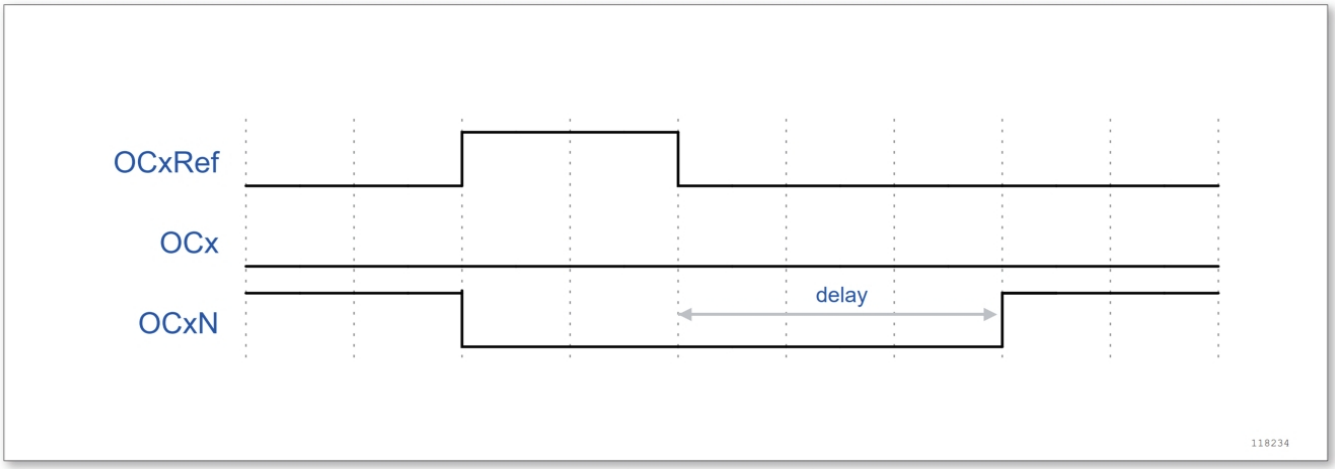


Figure 63. Dead-time waveforms with delay greater than the positive pulse

The dead-time delay is the same for each of the channels and is programmable with the DTG bits in the TIMx_BDTR register. Refer to the delay calculation in the Section 9.4.18.

Re-directing OCxREF to OCx or OCxN

In output mode (forced, output compare or PWM), OCxREF can be re-directed to the OCx output or to OCxN output by configuring the CCxE and CCxNE bits in the TIMx_CCER register.

This allows the user to send a specific waveform (such as PWM or static active level) on one output while the complementary remains at its inactive level. Other possibilities are to have both outputs at inactive level or both outputs active and complementary with dead-time.

Note: When only OCxN is enabled (CCxE = 0, CCxNE = 1), it is not complemented and becomes active as soon as OCxREF is high. For example, if CCxNP = 0 then OCxN = OCxREF. On the other hand, when both OCx and OCxN are enabled (CCxE = CCxNE = 1), OCx becomes active when OCxREF is high whereas OCxN is complemented and becomes active when OCxREF is low.

9.3.12 Using the break function

When using the break function, the output enable signals and inactive levels are modified according to corresponding control bits (MOE, OSSI and OSSR bits in the TIMx_BDTR register, OISx and OISxN bits in the TIMx_CR2 register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time. Refer to output control bits for complementary OCx and OCxN channels with break feature in the table of registers.

The break source can be either the break input pin or a clock failure event, generated by the Clock Security System (CSS), from the Reset Clock Controller.

When exiting from reset, the break circuit is disabled and the MOE bit is low. User can enable the break function by setting the BKE bit in the TIMx_BDTR register. The break input polarity can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time.

Because MOE falling edge can be asynchronous, a resynchronization circuit has been inserted between the actual signal (acting on the outputs) and the synchronous control bit (accessed in the TIMx_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if MOE is written to 1 whereas

(TIM1)

it was low, a delay (dummy instruction) must be inserted before reading it correctly. This is because the user writes an asynchronous signal, but reads a synchronous signal.

When a break occurs (selected level on the break input):

- The MOE bit is cleared asynchronously, putting the outputs in inactive state, idle state or in reset state (selected by the OSSI bit). This feature functions even if the MCU oscillator is off.
- Each output channel is driven with the level programmed in the OISx bit in the TIMx_CR2 register as soon as MOE=0. If OSSI = 0 then the timer releases the enable output else the enable output remains high.
- When complementary outputs are used:
  - The outputs are first put in reset state inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
  - If the timer clock is still present, then the dead-time generator is reactivated in order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that because of the resynchronization on MOE, the dead-time duration is a bit longer than usual (around 2 CK_TIM clock cycles).
  - If OSSI = 0 then the timer releases the enable outputs, else the enable outputs remain; or the enable outputs become high as soon as one of the CCxE or CCxNE bits is high.
- An interrupt can be generated if the BIE bit in the TIMx_DIER register is set and the break status flag (BIF bit in the TIMx_SR register) is set to ‘1’.
- If the AOE bit in the TIMx_BDTR register is set, the MOE bit is automatically set again at the next update event UEV. This can be used to perform a regulation, for instance. Else, MOE remains low until it is written to ‘1’ again. In this case, it can be used for security and the break input can be connected to an alarm from power drivers, thermal sensors or any security components.

Note: The break inputs is acting on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software).

In the meantime, the status flag BIF cannot be cleared.

The break can be generated by the BRK input which has a programmable polarity and an enable bit BKE in the TIMx_BDTR register.

In addition to the break input and the output management, a write protection has been implemented inside the break circuit to safeguard the application. It allows freezing the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). The user can choose from three levels of protection selected by the LOCK bits in the TIMx_BDTR register. Refer to Section 9.4.8. The LOCK bits can be written only once after an MCU reset.

The figure below shows an example of behavior of the outputs in response to a break:

(TIM1)

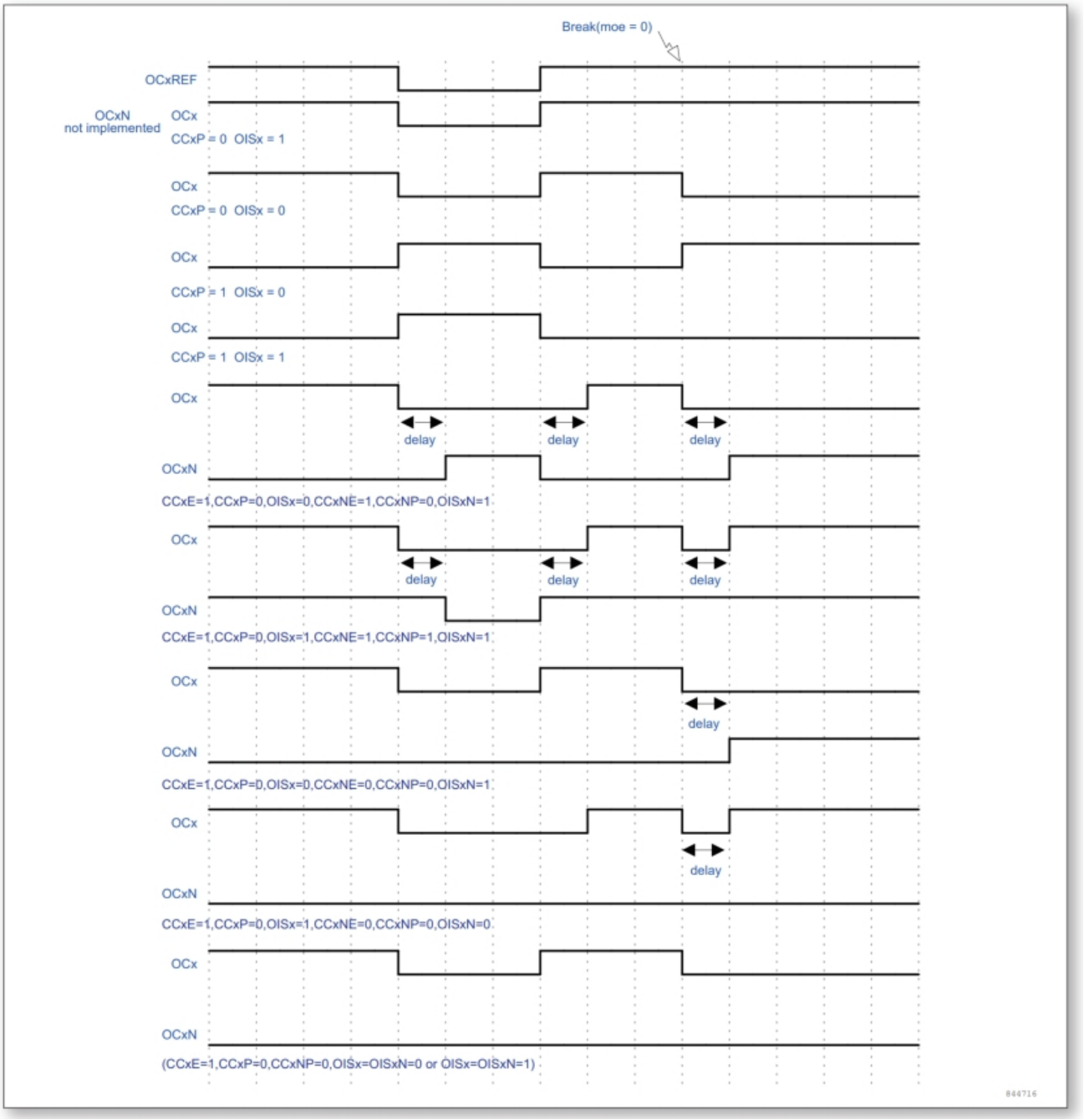


Figure 64. Output behavior in response to a break

9.3.13 6-step PWM generation

When complementary outputs are used on a channel, preload bits are available on the OCxM, CCxE and CCxNE bits. The preload bits are transferred to the shadow bits at the COM commutation event. The user can thus program in advance the configuration for the next step and change the configuration of all the channels at the same time. COM can be generated by software by setting the COM bit in the TIMx_EGR register or by hardware (on TRGI rising edge).

A flag is set when the COM event occurs (COMIF bit in the TIMx_SR register), which can generate an interrupt (if the COMIE bit is set in the TIMx_DIER register).

(TIM1)

The figure below describes the behavior of the OCx and OCxN outputs when a COM event occurs, in 3 different examples of programmed configurations.

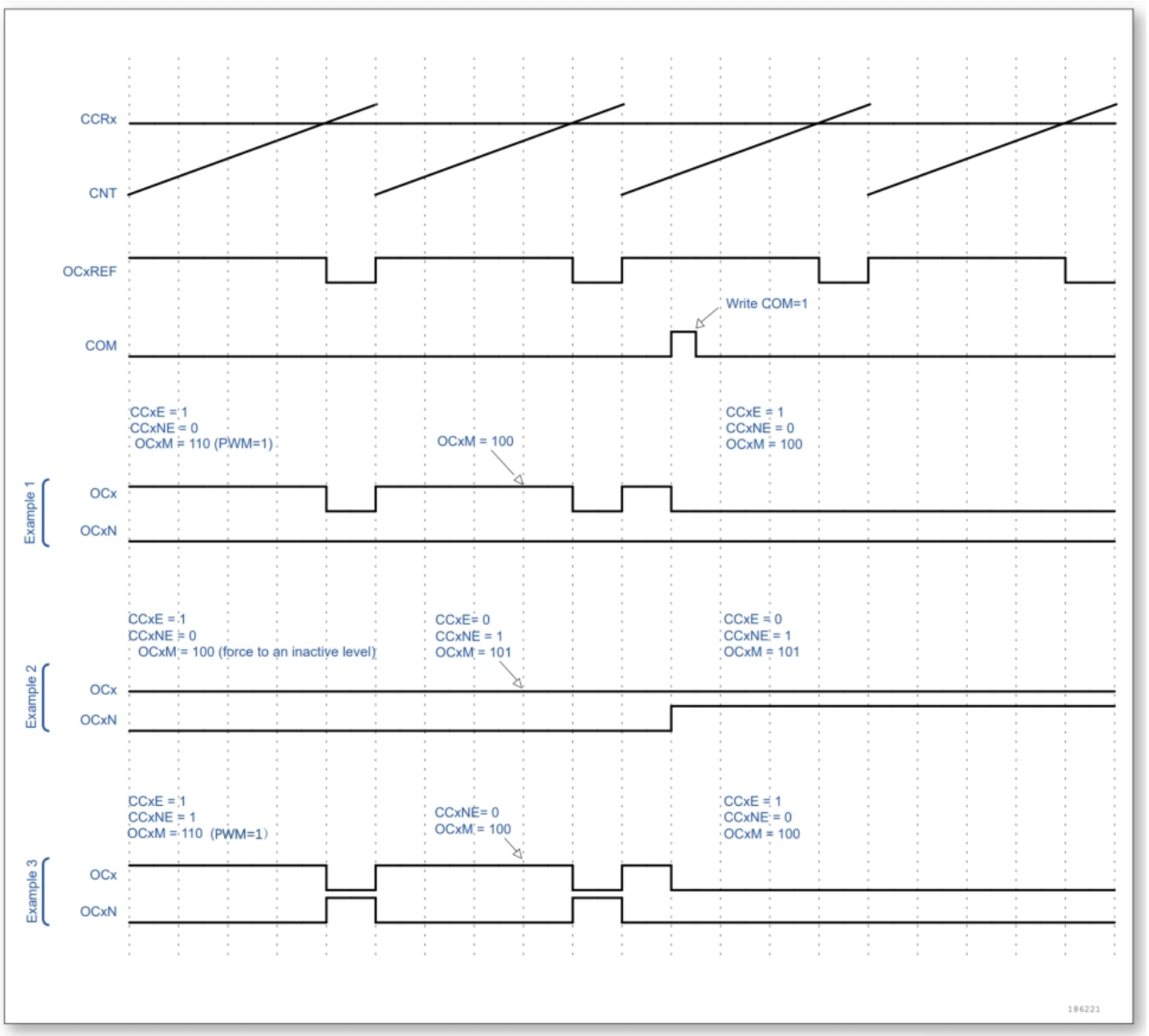


Figure 65. 6-step generation, COM example (OSSR=1)

9.3.14 One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. Select One-pulse mode by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV.

A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger),

the configuration must be:

- In upcounting: CNT < CCRx ≤ ARR (in particular, 0 < CCRx)
- In downcounting: CNT > CCRx

(TIM1)

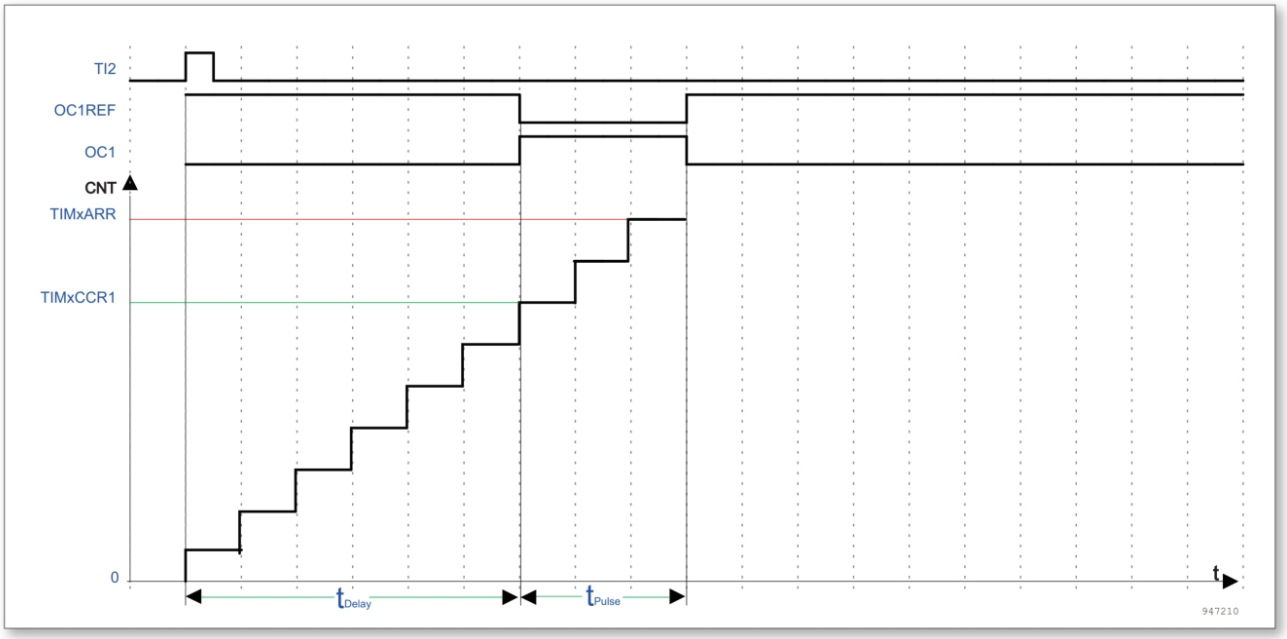


Figure 66. Example of one pulse mode

For example the user may want to generate a positive pulse on OC1 with a length of t_{PULSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI2 input pin.

Let's use TI2FP2 as trigger 1:

- Map TI2FP2 to TI2 by writing CC2S to '01' in the TIMx_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P to '0' in the TIMx_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS to '110' in the TIMx_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to '110' in the TIMx_SMCR register (trigger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The t_{DELAY} is defined by the value written in the TIMx_CCR1 register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let us say the user wants to build a waveform with a transition from '0' to '1' when a compare match occurs and a transition from '1' to '0' when the counter reaches the auto-reload value.

  To do this, enable PWM mode 2 by writing OC1M = 111 in the TIMx_CCMR1 register. The user can optionally enable the preload registers by writing OC1PE = 1 in the TIMx_CCMR1 register and ARPE in the TIMx_CR1 register. In this case the compare value must be written in the TIMx_CCR1 register, the auto-reload value in the TIMx_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to '1' in this example.

In our example, the DIR and CMS bits in the TIMx_CR1 register should be low.

The user only wants one pulse, so '1' must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0).

Particular case: OCx fast enable:

(TIM1)

In One-pulse mode, the edge detection on TIx input sets the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations and it limits the minimum delay t_{DELAY} min we can get.

If the user wants to output a waveform with the minimum delay, the OCxFE bit in the TIMx_CCMRx register must be set. Then OCxRef (and OCx) directly responses to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2 mode.

9.3.15 Encoder interface mode

To select Encoder Interface mode, write SMS to '001' in the TIMx_SMCR register if the counter is counting on TI2 edges only, SMS to '010' if it is counting on TI1 edges only and SMS to '011' if it is counting on both TI1 and TI2 edges.

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIMx_CCER register. When needed, the user can program the input filter as well. The two inputs TI1 and TI2 are used to interface to an incremental encoder.

Refer to Table 32. The counter is clocked by each valid transition on TI1FP1 or TI2FP2 (TI1 and TI2 after input filter and polarity selection, TI1FP1=TI1 if not filtered and not inverted, TI2FP2=TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit in TIMx_CR1 register written to '1'). The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence, the counter counts up or down, the DIR bit in the TIMx_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIMx_ARR register (0 to ARR or ARR down to 0 depending on the direction). So user must configure TIMx_ARR before starting. In the same way, the capture, compare, prescaler, repetition counter, trigger output features continue to work as normal.

In this mode, the counter is modified automatically following the speed and the direction of the incremental encoder and its content, therefore, always represents the encoder's position. The count direction correspond to the rotation direction of the connected sensor. The following table summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Table 32. Counting direction versus encoder signals

Active edge	**Level on opposite signal (TI1FP1 for TI2, TI2FP2 for TI1)**	**TI1FP1 signal**		**TI2FP2 signal**	
		Rising	**Falling**	**Rising**	**Falling**
Counting on TI1 only	High	Down	Up	No Count	No Count
	Low	Up	Down	No Count	No Count
Counting on TI2 only	High	No Count	No Count	Up	Down
	Low	No Count	No Count	Down	Up
Counting on TI1 and TI2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

(TIM1)

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally be used to convert the encoder's differential outputs to digital signals. This greatly increases noise immunity. The third encoder output which indicates the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset. The following figure gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points.

For this example we assume that the configuration is the following:

- CC1S = '01' (TIMx_CCMR1 register, IC1FP1 mapped on TI1).
- CC2S = '01' (TIMx_CCMR2 register, IC2FP2 mapped on TI2).
- CC1P = '0' (TIMx_CCER register, IC1FP1 non-inverted, IC1FP1 = TI1).
- C2P = '0' (TIMx_CCER register, IC2FP2 non-inverted, IC2FP2=TI2).
- SMS = '011' (TIMx_SMCR register, both inputs are active on both rising and falling edges).
- CEN = '1' (TIMx_CR1 register, counter enabled).

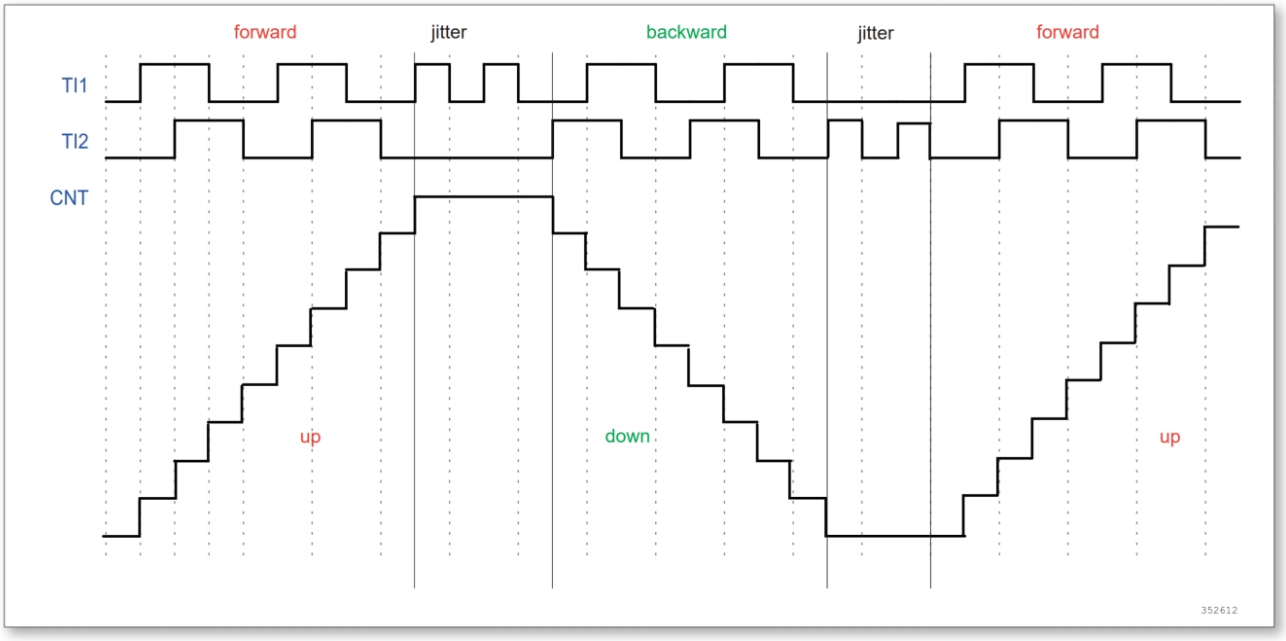


Figure 67. Example of counter operation in encoder interface mode

The following figure gives an example of counter behavior when IC1FP1 polarity is inverted (same configuration as above except CC1P = '1').

(TIM1)

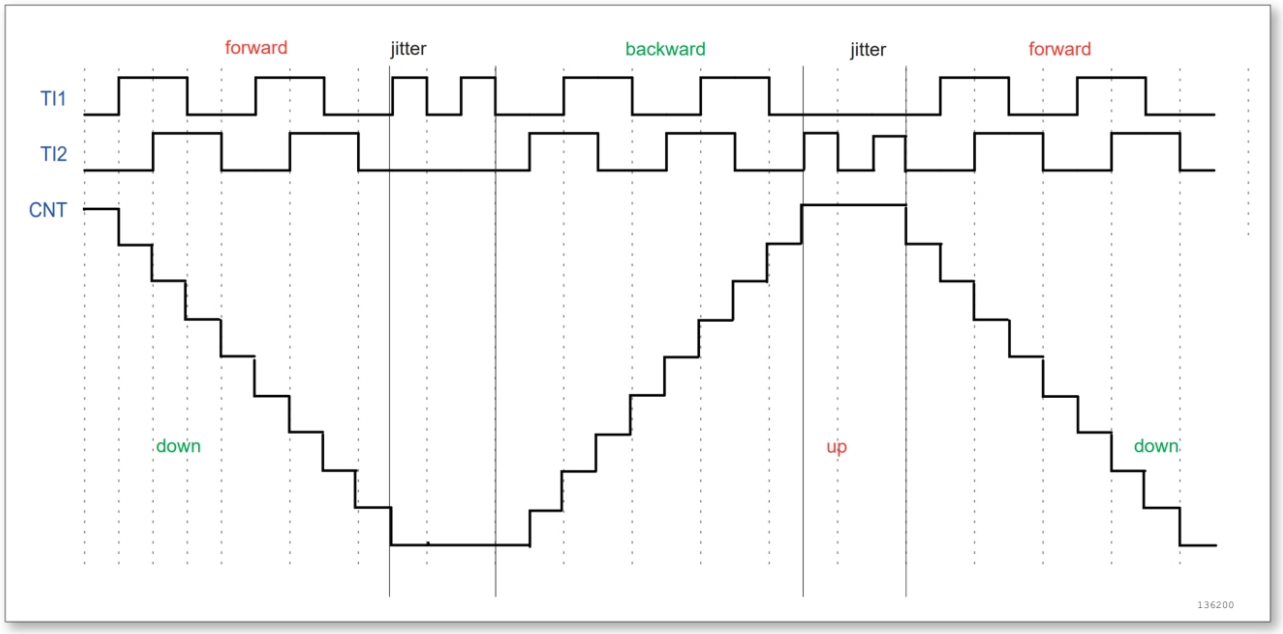


Figure 68. Example of encoder interface mode with IC1FP1 polarity inverted

The timer, when configured in Encoder Interface mode provides information on the sensor's current position. The user can obtain dynamic information (speed, acceleration, deceleration) by measuring the period between two encoder events using a second timer configured in capture mode. The output of the encoder which indicates the mechanical zero can be used for this purpose. Depending on the time between two events, the counter can also be read at regular times. This can be done by latching the counter value into a third input capture register if available (then the capture signal must be periodic and can be generated by another timer).

9.3.16 Timer input XOR function

The TI1S bit in the TIMx_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIMx_CH1, TIMx_CH2 and TIMx_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture. An example of this feature used to interface Hall sensors is given in Section 9.3.17.

9.3.17 Interfacing with Hall sensors

This is done using the advanced-control timer (TIM1) to generate PWM signals to drive the motor and another general-purpose timer TIMx (TIM2 or TIM3) referred to as "interfacing timer" in Figure 69.

The "interfacing timer" captures the 3 timer input pins (CC1, CC2, and CC3) connected through a XOR to the TI1 input channel (selected by setting the TI1S bit in the TIMx_CR2 register).

The slave mode controller is configured in reset mode; the slave input is TI1F_ED. Thus, each time one of the 3 inputs toggles, the counter restarts counting from 0. This creates a time base triggered by any change on the Hall inputs.

On the "interfacing timer", capture/compare channel 1 is configured in capture mode, capture signal is TRC (see Figure 52). The captured value, which corresponds to the time elapsed between 2 changes on the inputs, gives information about motor speed.

(TIM1)

The “interfacing timer” can be used in output mode to generate a pulse which changes the configuration of the channels of the advanced-control timer (TIM1) (by triggering a COM event). The TIM1 timer is used to generate PWM signals to drive the motor. To do this, the “interfacing timer” channel must be programmed so that a positive pulse is generated after a programmed delay (in output compare or PWM mode). This pulse is sent to the advanced-control timer (TIM1) through the TRGO output.

Example: the user wants to change the PWM configuration of the advanced-control timer TIMx after a programmed delay each time a change occurs on the Hall inputs connected to the TIMx timer.

- Configure 3 timer inputs ORed to the TI1 input channel by writing the TI1S bit in the TIMx_CR2 register to ‘1’.
- Program the time base: write the TIMx_ARR to the max value (the counter must be cleared by the TI1 change). Set the prescaler to get a maximum counter period longer than the time between 2 changes on the sensors.
- Program channel 1 in capture mode (TRC selected): write the CC1S bits in the TIMx_CCMR1 register to ‘01’. The user can also program the digital filter if needed.
  • Program channel 2 in PWM 2 mode with the desired delay: write the OC2M bits to ‘111’ and the CC2S bits to ‘00’ in the TIMx_CCMR1 register.
- Select OC2REF as trigger output on TRGO: write the MMS bits in the TIMx_CR2 register to ‘101’.

In the advanced-control timer TIM1, the right ITR input must be selected as trigger input, the timer is programmed to generate PWM signals, the capture/compare control signals are preloaded (CCPC = 1 in the TIMx_CR2 register) and the COM event is controlled by the trigger input (CCUS = 1 in the TIMx_CR2 register). The PWM control bits (CCxE, OCxM) are written after a COM event for the next step (this can be done in an interrupt subroutine generated by the rising edge of OC2REF).

The following figure describes this example:

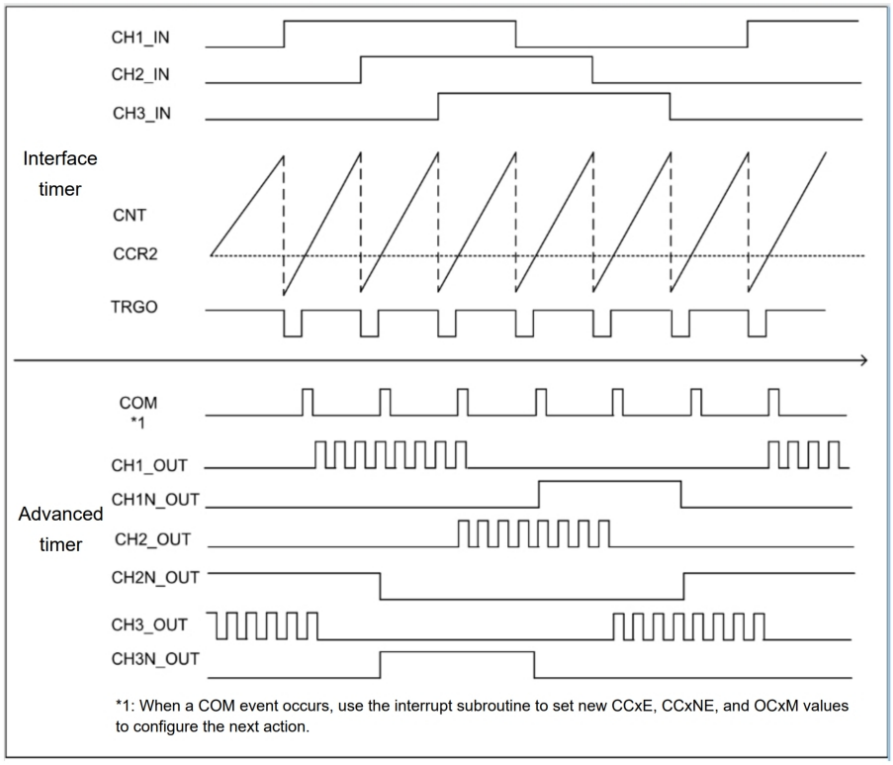


Figure 69. Example of Hall sensor interface

(TIM1)

9.3.18 TIMx and external trigger synchronization

The TIMx timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

Slave mode: Reset mode

The counter and its prescaler can be reinitialized in response to an event on a trigger input.

Moreover, if the URS bit from the TIMx_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIMx_ARR, TIMx_CCRx) are updated.

In the following example, the upcounter is cleared in response to a rising edge on TI1 input:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC1S bits select the input capture source only, i.e. CC1S = 01 in the TIMx_CCMR1 register. Write CC1P = 0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS = 100 in TIMx_SMCR register. Select TI1 as the input source by writing TS = 101 in TIMx_SMCR register.
- Start the counter by writing CEN = 1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. In the meantime, the trigger flag is set (TIF bit in the TIMx_SR register) and an interrupt request can be sent if the TIE bit (interrupt enable) is set in TIMx_DIER register.

The following figure shows this behavior when the auto-reload register TIMx_ARR = 0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

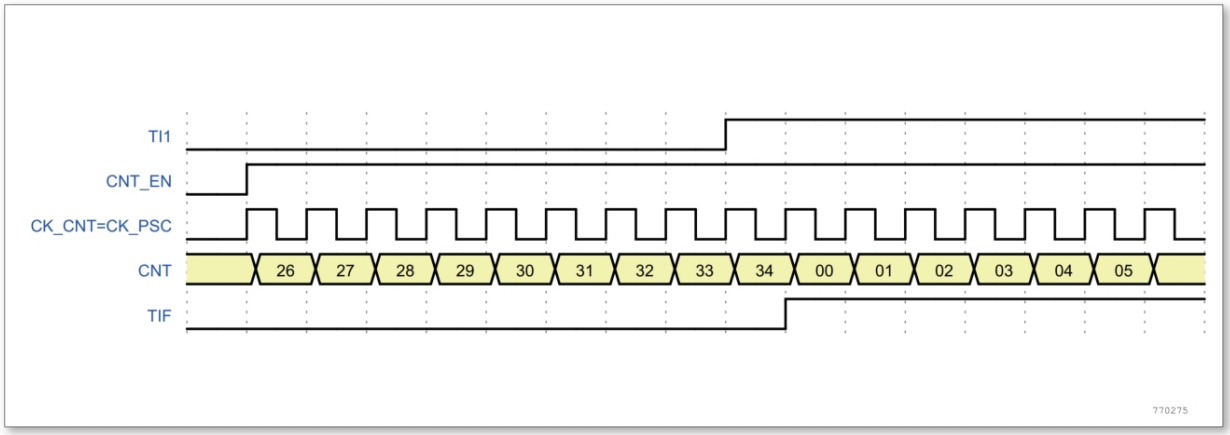


Figure 70. Control circuit in reset mode

Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so the user does not need to configure it. The CC1S bits select the input capture source only. Write CC1S = 01 in TIMx_CCMR1 register. Write CC1P = 1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS = 101 in TIMx_SMCR register. Select TI1 as the input source by writing TS = 101 in TIMx_SMCR register.

(TIM1)

- Enable the counter by writing CEN = 1 in the TIMx_CR1 register (in gated mode, the counter doesn't start if CEN = 0, whatever is the trigger input level).

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

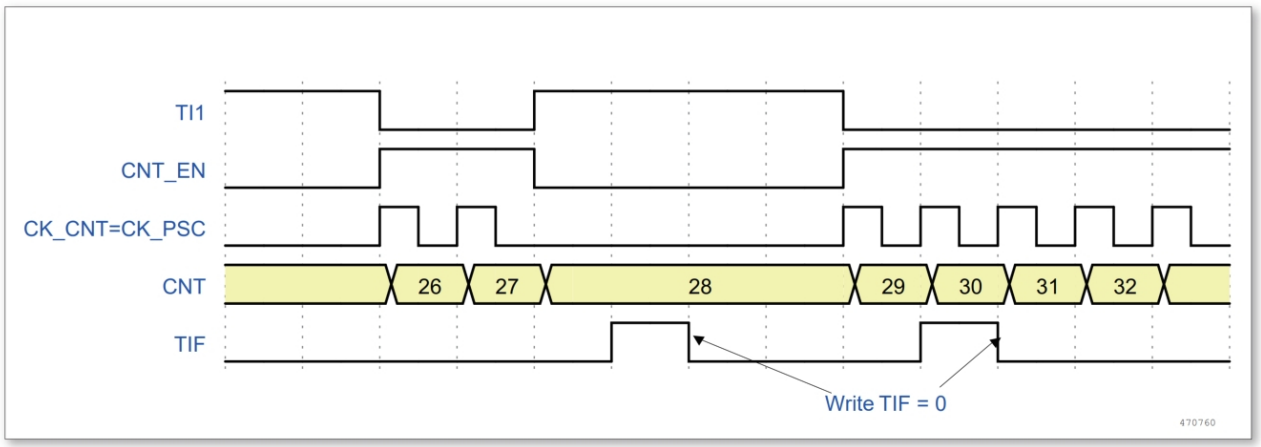


Figure 71. Control circuit in gated mode

Slave mode: Trigger mode

The counter can start in response to an event on a selected input.

In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we don't need any filter, so we keep IC2F = 0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC2S bits are configured to select the input capture source only. Write CC2P = 1 to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS = 110 in TIMx_SMCR register. Select TI2 as the input source by writing TS = 110 in TIMx_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set. The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

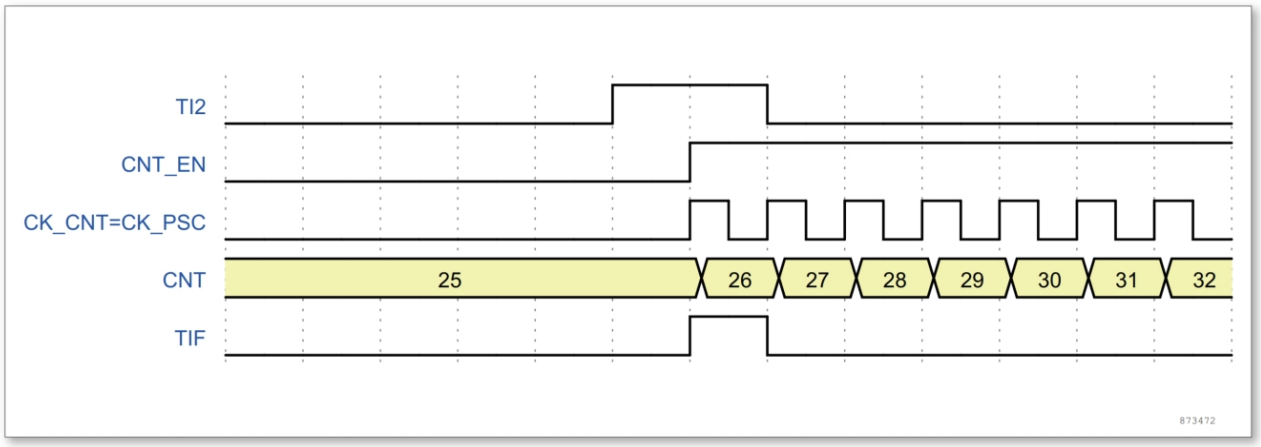


Figure 72. Control circuit in trigger mode

(TIM1)

9.3.19 Timer synchronization

The TIM timers are linked together internally for timer synchronization or chaining. Refer to Section TIMx for details.

9.3.20 Debug mode

When the microcontroller enters debug mode (CPU core halted), the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module. For more details, refer to subsequent debug section.

9.4 Register description

Table 33. TIM1 register overview

Offset	Acronym	Register Name	Reset	Section
0x00	TIMx_CR1	Control register 1	0x00000000	Section 9.4.1
0x04	TIMx_CR2	Control register 2	0x00000000	Section 9.4.2
0x08	TIMx_SMCR	Slave mode control register	0x00000000	Section 9.4.3
0x0C	TIMX_DIER	Interrupt enable register	0x00000000	Section 9.4.4
0x10	TIMx_SR	Status register	0x00000000	Section 9.4.5
0x14	TIMx_EGR	Event generate register	0x00000000	Section 9.4.6
0x18	TIMx_CCMR1	Capture/compare mode register 1	0x00000000	Section 9.4.7
0x1C	TIMx_CCMR2	Capture/compare mode register 2	0x00000000	Section 9.4.8
0x20	TIMx_CCER	Capture/compare enable register	0x00000000	Section 9.4.9
0x24	TIMx_CNT	Counter	0x00000000	Section 9.4.10
0x28	TIMx_PSC	Prescaler	0x00000000	Section 9.4.11
0x2C	TIMx_ARR	Auto-reload register	0x00000000	Section 9.4.12
0x30	TIMx_RCR	Repetition counter register	0x00000000	Section 9.4.13
0x34	TIMx_CCR1	Capture/compare register 1	0x00000000	Section 9.4.14
0x38	TIMx_CCR2	Capture/compare register 2	0x00000000	Section 9.4.15
0x3C	TIMx_CCR3	Capture/compare register 3	0x00000000	Section 9.4.16
0x40	TIMx_CCR4	Capture/compare register 4	0x00000000	Section 9.4.17
0x44	TIMx_BDTR	Brake and dead-time register	0x00000000	Section 9.4.18
0x5C	TIMx_PDER	PWM phase-shift update request enable register	0x00000000	Section 9.4.21
0x60 ~ 0x70	CCRxFALL	PWM phase-shift downcounting capture/compare register	0x00000000	Section 9.4.22

(TIM1)

9.4.1 Control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKD		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:10	Reserved			Reserved, always read as 0
9:8	CKD	rw	0x00	Clock division These 2 bits indicate the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock used by the dead-time generators and the digital filters (TIx). 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: Reserved, do not program this value
7	ARPE	rw	0x00	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6:5	CMS	rw	0x00	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set both when the counter is counting up or down. It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN = 1). See center-aligned PWM waveforms for details.

(TIM1)

Bit	Field	Type	Reset	Description
4	DIR	rw	0x00	Direction 0: Counter used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
3	OPM	rw	0x00	One pulse mode 0: Counter is not stopped at update event. 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	rw	0x00	Update request source This bit is set by software to select the UEV event sources. 0: Any of the following events generate an update interrupt if enabled. These events can be: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt if enabled.
1	UDIS	rw	0x00	Update disable This bit is set by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: - Counter overflow/underflow - Setting the UG bit to '1' - Update generation through the slave mode controller with the subsequent update of shadow registers 1: UEV disabled. The Update event is not generated, shadow registers keep their values (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set to '1' or if a hardware reset is received from the slave mode controller.
0	CEN	rw	0x00	Counter enable 0: Counter disabled 1: Counter enabled Note: External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware.

9.4.2 Control register 2 (TIMx_CR2)

Address offset: 0x04

Reset value: 0x0000 0000

(TIM1)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS			Res.	CCUS	Res.	CCPC
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw

Bit	Field	Type	Reset	Description
31: 15	Reserved			Reserved, always read as 0.
14	OIS4	rw	0x00	Output Idle state 4 (OC4 output). Refer to OIS1 bit.
13	OIS3N	rw	0x00	Output Idle state 3 (OC3N output). Refer to OIS1N bit.
12	OIS3	rw	0x00	Output Idle state 3 (OC3 output). Refer to OIS1 bit.
11	OIS2N	rw	0x00	Output Idle state 2 (OC2N output). Refer to OIS1N bit.
10	OIS2	rw	0x00	Output Idle state 2 (OC2 output). Refer to OIS1 bit.
9	OIS1N	rw	0x00	Output Idle state 1 (OC1N output) 0: OC1N = 0 after a dead-time when MOE = 0 1: OC1N = 1 after a dead-time when MOE = 0 Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).
8	OIS1	rw	0x00	Output Idle state 1 (OC1 output) 0: OC1 = 0 (after a dead-time if OC1N is implemented) when MOE = 0 1: OC1 = 1 (after a dead-time if OC1N is implemented) when MOE = 0 Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).
7	TI1S	rw	0x00	TI1 selection 0: The TIMx_CH1 pin is connected to TI1 input 1: The TIMx_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)

(TIM1)

Bit	Field	Type	Reset	Description
6:4	MMS	rw	0x00	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the TIMx_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 001: Enable - the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIMx_SMCR register). 010: Update - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output (TRGO) send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. 100: Compare - OC1REF signal is used as trigger output (TRGO) 101: Compare - OC2REF signal is used as trigger output (TRGO) 110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
3	Reserved			Reserved and always read as 0.
2	CCUS	rw	0x00	Capture/compare control update selection 0: When capture/compare control bits are preloaded (CCPC = 1), they are updated by setting the COM bit only 1: When capture/compare control bits are preloaded (CCPC = 1), they are updated by setting the COM bit or when a rising edge occurs on TRGI Note: This bit acts only on channels that have a complementary output.
1	Reserved			Reserved and always read as 0.

(TIM1)

Bit	Field	Type	Reset	Description
0	CCPC	rw	0x00	Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded 1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when the COM bit is set. Note: This bit acts only on channels that have a complementary output.

9.4.3 Slave mode control register (TIMx_SMCR)

Address offset: 0x08

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.								MSM	TS			Res.	SMS		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw

Bit	Field	Type	Reset	Description
15:8	Reserved			Reserved, always read as 0.
7	MSM	rw	0x00	Master/slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.
6:4	TS	rw	0x00	Trigger selection This bit-field selects the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0) 001: Internal Trigger 1 (ITR1) 010: Internal Trigger 2 (ITR2) 011: Internal Trigger 3 (ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) See the following table for more details on ITRx. Note: These bits must be changed only when they are not used (e.g. when SMS=000) to avoid wrong edge detections at the transition.
3	Reserved			Reserved and always read as 0.

(TIM1)

Bit	Field	Type	Reset	Description
2:0	SMS	rw	0x00	Slave mode selection When external signals are selected, the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control register description). 000: Slave mode disabled - if CEN = 1, then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 - Counter counts up/down on TI2FP2 edge depending on TI1FP1 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. 011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 110: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 - Rising edges of the selected trigger (TRGI) clock the counter. Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS = 100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.

Table 34. TIMx internal trigger connection

Slave TIM	ITR0	ITR1	ITR2	ITR3
TIM1	None	None	TIM3	None
TIM3	TIM1	None	None	None

9.4.4 Interrupt enable register (TIMX_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

(TIM1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															CC5IE
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31: 17	Reserved			Reserved, always read as 0.
16	CC5IE	rw	0x00	Capture/Compare 5 interrupt enable 0: Capture/Compare 5 interrupt disabled 1: Capture/Compare 5 interrupt enabled
15: 8	Reserved			Reserved and always read as 0.
7	BIE	rw	0x00	Break interrupt enable 0: Break interrupt disabled 1: Break interrupt enabled
6	TIE	rw	0x00	Trigger interrupt enable 0: Trigger interrupt disabled 1: Trigger interrupt enabled
5	COMIE	rw	0x00	COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
4	CC4IE	rw	0x00	Capture/Compare 4 interrupt enable 0: Capture/Compare 4 interrupt disabled 1: Capture/Compare 4 interrupt enabled
3	CC3IE	rw	0x00	Capture/Compare 3 interrupt enable 0: Capture/Compare 3 interrupt disabled 1: Capture/Compare 3 interrupt enabled
2	CC2IE	rw	0x00	Capture/Compare 2 interrupt enable 0: Capture/Compare 2 interrupt disabled 1: Capture/Compare 2 interrupt enabled
1	CC1IE	rw	0x00	Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
0	UIE	rw	0x00	Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

(TIM1)

9.4.5 Status register (TIMx_SR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															CC5IF
															rc_w0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4OF	CC3OF	CC2OF	CC1OF	Res.	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			rc_w0	rc_w0	rc_w0	rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bit	Field	Type	Reset	Description
31:17	Reserved			Reserved, always read as 0
16	CC5IF	rc_w0	0x00	Capture/Compare 5 interrupt flag Refer to CC1IF description.
15:13	Reserved			Reserved and always read as 0.
12	CC4OF	rc_w0	0x00	Capture/Compare 4 overcapture flag Refer to CC1OF description.
11	CC3OF	rc_w0	0x00	Capture/Compare 3 overcapture flag Refer to CC1OF description.
10	CC2OF	rc_w0	0x00	Capture/Compare 2 overcapture flag refer to CC1OF description.
9	CC1OF	rc_w0	0x00	Capture/Compare 1 overcapture flag This flag is set to '1' by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set to 1.
8	Reserved			Reserved and always read as 0.
7	BIF	rc_w0	0x00	Break interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input.

(TIM1)

Bit	Field	Type	Reset	Description
6	TIF	rc_w0	0x00	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode, both edges in case gated mode is selected). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
5	COMIF	rc_w0	0x00	COM interrupt flag This flag is set by hardware on COM event (when capture/compare control bits - CCxE, CCxNE, OCxM - have been updated). It is cleared by software. 0: No COM event occurred 1: COM interrupt pending.
4	CC4IF	rc_w0	0x00	Capture/Compare 4 interrupt flag Refer to CC1IF description.
3	CC3IF	rc_w0	0x00	Capture/Compare 3 interrupt flag Refer to CC1IF description.
2	CC2IF	rc_w0	0x00	Capture/Compare 2 interrupt flag Refer to CC1IF description.
1	CC1IF	rc_w0	0x00	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIMx_CR1 register description). It is cleared by software. 0: No match. 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in (copied to) TIMx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)

(TIM1)

Bit	Field	Type	Reset	Description
0	UIF	rc_w0	0x00	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update event pending. This bit is set by hardware when the registers are updated: - At overflow or underflow regarding the repetition counter value (update if REP_CNT = 0) and if the UDIS = 0 in the TIMx_CR1 register. - When CNT is reinitialized by software using the UG bit in TIMx_EGR register, if URS=0 and UDIS=0 in the TIMx_CR1 register. - When CNT is reinitialized by a trigger event (refer to the synchronous control register description), if URS=0 and UDIS=0 in the TIMx_CR1 register.

9.4.6 Event generation register (TIMx_EGR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															CC5G
															w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
								w	w	w	w	w	w	w	w

Bit	Field	Type	Reset	Description
31:17	Reserved			Reserved, always read as 0
16	CC5G	w	0x00	Capture/Compare 5 generation Refer to CC1G description.
15:8	Reserved			Reserved and always read as 0.
7	BG	w	0x00	Break generation This bit is set by software in order to generate a brake event, it is automatically cleared by hardware. 0: No action 1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt can occur if enabled.

(TIM1)

Bit	Field	Type	Reset	Description
6	TG	w	0x00	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt can occur if enabled.
5	COMG	w	0x00	Capture/Compare control update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note: This bit acts only on channels that have a complementary output.
4	CC4G	w	0x00	Capture/Compare 4 generation Refer to CC1G description.
3	CC3G	w	0x00	Capture/Compare 3 generation Refer to CC1G description.
2	CC2G	w	0x00	Capture/Compare 2 generation Refer to CC1G description.
1	CC1G	w	0x00	Capture/Compare 1 generation This bit is set by software in order to generate a capture/compare event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as output: CC1IF flag is set, corresponding interrupt is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	w	0x00	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR = 0 (upcounting), else it takes the auto-reload value (TIMx_ARR) if DIR = 1 (downcounting).

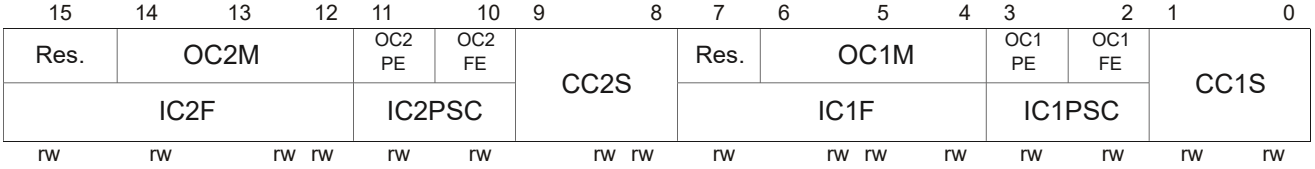
(TIM1)

9.4.7 Capture/compare mode register 1 (TIMx_CCMR1)

Address offset: 0x18

Reset value: 0x0000

The channels can be used in input (capture mode) or in output (compare mode). The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function in output mode. For a given bit, OCxx describes its function when the channel is configured in output, ICxx describes its function when the channel is configured in input. So the user must take care that the same bit can have a different meaning for the input stage and for the output stage.



Output compare mode:

Bit	Field	Type	Reset	Description
15	Reserved			Reserved, always read as 0.
14:12	OC2M	rw	0x00	Output compare 2 mode
11	OC2PE	rw	0x00	Output compare 2 preload enable
10	OC2FE	rw	0x00	Output compare 4 fast enable
9:8	CC2S	rw	0x00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output. 01: CC2 channel is configured as input, IC2 is mapped on TI2. 10: CC2 channel is configured as input, IC2 is mapped on TI1. 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = 0 in TIMx_CCER).
7	Reserved			Reserved and always read as 0.

(TIM1)

Bit	Field	Type	Reset	Description
6:4	OC1M	rw	0x00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the OC1REF. 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle. OC1REF level toggles when TIMx_CCR1=TIMx_CNT. 100: Force inactive level. OC1REF is forced low. 101: Force active level. OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as TIMx_CNT < TIMx_CCR1, else inactive. In downcounting, channel 1 is inactive (OC1REF = 0) as long as TIMx_CNT > TIMx_CCR1, else active (OC1REF = 1). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1, else active. In downcounting, channel 1 is active as long as TIMx_CNT > TIMx_CCR1, else inactive. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = 00 (the channel is configured in output). Note 2: In PWM mode 1 or 2, the OC1REF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.
3	OC1PE	rw	0x00	Output compare 1 preload enable 0: Preload function of TIMx_CCR1 register disabled. TIMx_CCR1 register can be written at anytime, the new value is taken in account immediately. 1: Preload function of TIMx_CCR1 register enabled. Read/Write operations access the preload register. TIMx_CCR1 preload value is loaded in the active register at each update event. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = 00 (the channel is configured in output). Note 2: The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIMx_CR1 register). Else the behavior is not guaranteed.

(TIM1)

Bit	Field	Type	Reset	Description
2	OC1FE	rw	0x00	Output compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S	rw	0x00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register). Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIMx_CCER).

Input capture mode:

Bit	Field	Type	Reset	Description
15:12	IC2F	rw	0x00	Input capture 2 filter
11:10	IC2PSC	rw	0x00	Input capture 2 prescaler
9:8	CC2S	rw	0x00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = 0 in TIMx_CCER).

(TIM1)

Bit	Field	Type	Reset	Description
7:4	IC1F	rw	0x00	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to generate a transition on the output: 0000: No filter, sampling is done at f_{DTS} 1000: $f_{SAMPLING}=f_{DTS/8}$, N = 6 0001: $f_{SAMPLING}=f_{CK\_INT}$, N = 2 1001: $f_{SAMPLING}=f_{DTS/8}$, N = 8 0010: $f_{SAMPLING}=f_{CK\_INT}$, N = 4 1010: $f_{SAMPLING}=f_{DTS/16}$, N = 5 0011: $f_{SAMPLING}=f_{CK\_INT}$, N = 8 1011: $f_{SAMPLING}=f_{DTS/16}$, N = 6 0100: $f_{SAMPLING}=f_{DTS/2}$, N = 6 1100: $f_{SAMPLING}=f_{DTS/16}$, N = 8 0101: $f_{SAMPLING}=f_{DTS/2}$, N = 8 1101: $f_{SAMPLING}=f_{DTS/32}$, N = 5 0110: $f_{SAMPLING}=f_{DTS/4}$， N = 6 1110: $f_{SAMPLING}=f_{DTS/32}$, N = 6 0111: $f_{SAMPLING}=f_{DTS/4}$， N = 8 1111: $f_{SAMPLING}=f_{DTS/32}$, N = 8
3:2	IC1PSC	rw	0x00	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E = 0 (TIMx_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S	rw	0x00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIMx_CCER).

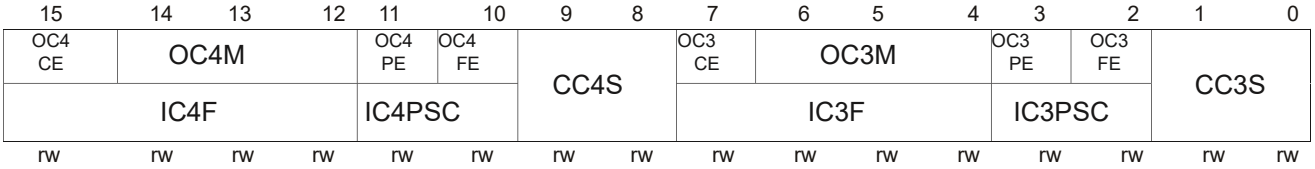
(TIM1)

9.4.8 Capture/compare mode register 2 (TIMx_CCMR2)

Address offset: 0x1C

Reset value: 0x0000

Refer to the above CCMR1 register description.



Output compare mode

Bit	Field	Type	Reset	Description
15	OC4CE	rw	0x00	Output compare 4 clear enable
14:12	OC4M	rw	0x00	Output compare 4 mode
11	OC4PE	rw	0x00	Output compare 4 preload enable
10	OC4FE	rw	0x00	Output compare 4 fast enable
9:8	CC4S	rw	0x00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = 0 in TIMx_CCER).
7	OC3CE	rw	0x00	Output compare 3 clear enable
6:4	OC3M	rw	0x00	Output compare 3 mode
3	OC3PE	rw	0x00	Output compare 3 preload enable
2	OC3FE	rw	0x00	Output compare 3 fast enable

(TIM1)

Bit	Field	Type	Reset	Description
1:0	CC3S	rw	0x00	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = 0 in TIMx_CCER).

Input capture mode

Bit	Field	Type	Reset	Description
15:12	IC4F	rw	0x00	Input capture 4 filter
11:10	IC4PSC	rw	0x00	Input capture 4 prescaler
9:8	CC4S	rw	0x00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = 0 in TIMx_CCER).
7:4	IC3F	rw	0x00	Input capture 3 filter
3:2	IC3PSC	rw	0x00	Input capture 3 prescaler
1:0	CC3S	rw	0x00	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = 0 in TIMx_CCER).

(TIM1)

9.4.9 Capture/compare enable register (TIMx_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	Res.	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15	CC4NP	rw	0x00	Capture/Compare 4 complementary output polarity Refer to CC1NP description.
14	Reserved			Reserved and always read as 0.
13	CC4P	rw	0x00	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	rw	0x00	Capture/Compare 4 output enable Refer to CC1E description.
11	CC3NP	rw	0x00	Capture/Compare 3 complementary output polarity Refer to CC1NP description.
10	CC3NE	rw	0x00	Capture/Compare 3 complementary output enable Refer to CC1NE description.
9	CC3P	rw	0x00	Capture/Compare 3 output polarity Refer to CC1NE description.
8	CC3E	rw	0x00	Capture/Compare 3 output enable Refer to CC1E description.
7	CC2NP	rw	0x00	Capture/Compare 2 complementary output polarity Refer to CC1NP description.
6	CC2NE	rw	0x00	Capture/Compare 2 complementary output enable Refer to CC1NE description.
5	CC2P	rw	0x00	Capture/Compare 2 output polarity Refer to CC1P description.
4	CC2E	rw	0x00	Capture/Compare 2 output enable Refer to CC1E description.
3	CC1NP	rw	0x00	Capture/Compare 1 complementary output polarity 0: OC1N active high 1: OC1N active low Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = 00 (the channel is configured in output).

(TIM1)

Bit	Field	Type	Reset	Description
2	CC1NE	rw	0x00	Capture/Compare 1 complementary output enable 0: Off - OC1N is not active. OC1N level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. 1: On - OC1N signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits.
1	CC1P	rw	0x00	Capture/Compare 1 output polarity CC1 channel configured as output: 0: OC1 active high 1: OC1 active low CC1 channel configured as input: This bit selects whether the inverted signal of IC1 or IC1 is used for trigger or capture operations. 0: non-inverted: capture is done on a rising edge of IC1. When used as external trigger, IC1 is non-inverted. 1: inverted: capture is done on a falling edge of IC1. When used as external trigger, IC1 is inverted. Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).
0	CC1E	rw	0x00	Capture/Compare 1 output enable CC1 channel configured as output: 0: Off - OC1 is not active. OC1 level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. 1: On - OC1 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the TIMx_CCR1 register or not. 0: capture disabled 1: capture enabled

(TIM1)

Table 35. Output control bits for complementary OCx and OCxN channels with break feature

Control bits					Output states[1]	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx output state	OCxN output state
1	X	0	0	0	Output Disabled (not driven by the timer), OCx = 0, OCx_EN = 0	Output Disabled (not driven by the timer), OCxN = 0, OCxN_EN = 0
		0	0	1	Output Disabled (not driven by the timer), OCx = 0, OCx_EN = 0	OCxREF + Polarity, OCxN = OCxREF xor CCxNP, OCxN_EN = 1
		0	1	0	OCxREF + Polarity, OCx = OCxREF xor CCxP, OCx_EN = 1	Output Disabled (not driven by the timer), OCxN = 0, OCxN_EN = 0
		0	1	1	OCxREF + Polarity + dead-time, OCx_EN = 1	Complementary to OCxREF + Polarity + dead-time, OCxN_EN = 1
		1	0	0	Output Disabled (not driven by the timer), OCx = CCxP, OCx_EN = 0	Output Disabled (not driven by the timer), OCxN = CCxNP, OCxN_EN = 0
		1	0	1	Off-State (output enabled with inactive state) OCx = CCxP, OCx_EN = 1	OCxREF + Polarity, OCxN = OCxREF xor CCxNP, OCxN_EN = 1
		1	1	0	OCxREF + Polarity, OCx = OCxREF xor CCxP, OCx_EN = 1	Off-State (output enabled with inactive state) OCxN = CCxNP, OCxN_EN = 1
		1	1	1	OCxREF + Polarity + dead-time, OCx_EN = 1	Complementary to OCxREF + Polarity + dead-time, OCxN_EN = 1
0	0	X	0	0	Output Disabled (not driven by the timer), Asynchronously: OCx = CCxP, OCx_EN = 0, OCxN = CCxNP, OCxN_EN = 0; Then if the clock is present: OCx=OISx and OCxN=OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state.	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0	Off-State (output enabled with inactive state) Asynchronously: OCx = CCxP, OCx_EN = 1, OCxN = CCxNP, OCxN_EN = 1; Then if the clock is present: OCx=OISx and OCxN=OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state.	
	1		0	1		
	1		1	0		
	1		1	1		

(TIM1)

1. When both outputs of a channel are not used (CCxE = CCxNE = 0), the OISx, OISxN, CCxP and CCxNP bits must be kept cleared.

Note: The state of the external I/O pins connected to the complementary OCx and OCxN channels depends on the OCx and OCxN channel state and the GPIO registers.

9.4.10 Counter (TIMx_CNT)

Address offset: 0x24

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x0000	Counter value

9.4.11 Prescaler (TIMx_PSC)

Address offset: 0x28

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:0	PSC	rw	0x0000	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC}/(PSC + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through slave controller when configured in “reset mode”).

9.4.12 Auto-reload register (TIMx_ARR)

Address offset: 0x2C

Reset value: 0x0000

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

(TIM1)

Bit	Field	Type	Reset	Description
15:0	ARR	rw	0x0000	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to Section 9.3.1: Time-base unit for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

9.4.13 Repetition counter register (TIMx_RCR)

Address offset: 0x30

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:8	REP_CNT	rw	0x00	Repetition counter value of real-time writing These bits are used in the repetition counter modification mode, to move the update interrupt flag (UIF) to left by (REP-REP_CNT) phases (to right when the subtracted value is negative) by shifting UIF detection point in real time. These bits should be written after the update event UG is generated (note writing to REP_CNT before the update event is generated makes the displacement invalid).
7:0	REP	rw	0x00	Repetition counter value These bits allow the user to set-up the update rate of the compare registers (i.e. Periodic transfers from preload to active registers) when preload registers are enabled, as well as the update interrupt generation rate, if this interrupt is enabled. Each time the REP_CNT related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event U_RC, any write to the TIMx_RCR register is not taken in account until the next repetition update event. It means in PWM mode, (REP+1) corresponds to: - the number of PWM periods in edge-aligned mode - the number of half PWM period in center-aligned mode.

9.4.14 Capture/compare register 1 (TIMx_CCR1)

Address offset: 0x34

Reset value: 0x0000

(TIM1)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:0	CCR1	rw	0x0000	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded immediately if the preload feature is not selected in the TIMx_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. If channel CC1 is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

9.4.15 Capture/compare register 2 (TIMx_CCR2)

Address offset: 0x38

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:0	CCR2	rw	0x0000	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value). It is loaded immediately if the preload feature is not selected in the TIMx_CCMR2 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC2 output. If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2).

9.4.16 Capture/compare register 3 (TIMx_CCR3)

Address offset: 0x3C

Reset value: 0x0000

(TIM1)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:0	CCR3	rw	0x0000	Capture/Compare 3 value If channel CC3 is configured as output: CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value). It is loaded immediately if the preload feature is not selected in the TIMx_CCMR3 register (bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC3 output. If channel CC3 is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3).

9.4.17 Capture/compare register 4 (TIMx_CCR4)

Address offset: 0x40

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:0	CCR4	rw	0x0000	Capture/Compare 4 value If channel CC4 is configured as output: CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value). It is loaded immediately if the preload feature is not selected in the TIMx_CCMR4 register (bit OC4PE). Else the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC4 output. If channel CC4 is configured as input: CCR4 is the counter value transferred by the last input capture 4 event (IC4).

(TIM1)

9.4.18 Break and dead-time register (TIMx_BDTR)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															DOE
															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK		DTG							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Note: As the bits AOE, BKP, BKE, OSSI, OSSR and DTG can be write-locked depending on the LOCK configuration, it can be necessary to configure all of them during the first write access to the TIMx_BDTR register.

Bit	Field	Type	Reset	Description
31:17	Reserved			Reserved, always read as 0.
16	DOE	rw	0x00	Direct output enable This bit is enabled when the break input is active and MOE is cleared. 1: outputs in idle state directly, without any delay of dead-time 0: when the brake input is generated, the outputs turn into idle state after a dead-time
15	MOE	rw	0x00	Main output enable This bit is cleared asynchronously by hardware as soon as the break input is active. It is cleared as ‘0’ by software or automatically set to ‘1’ depending on the AOE bit configuration. It is acting only on the channels which are configured in output. 0: OC and OCN outputs are disabled or forced to idle state. 1: OC and OCN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in TIMx_CCER register). See OC/OCN enable description for more details (Section 9.4.9: capture/compare enable register (TIMx_CCER)).

Bit	Field	Type	Reset	Description
14	AOE	rw	0x00	Automatic output enable 0: MOE can be set only by software 1: MOE can be set by software or automatically at the next update event (if the break input is not be active) Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
13	BKP	rw	0x00	Break polarity 0: Break input BRK is active low 1: Break input BRK is active high Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
12	BKE	rw	0x00	Break enable 0: Break inputs (BRK and BRK_ACTH) disabled 1: Break inputs (BRK and BRK_ACTH) enabled Note: This bit cannot be modified when LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
11	OSSR	rw	0x00	Off-state selection for Run mode This bit is used when MOE = 1 on channels having a complementary output. OSSR is not implemented if no complementary output is implemented in the timer. See OC/OCN enable description for more details (Section 9.4.9: capture/compare enable register (TIMx_CCER)). 0: When the timer is inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0). 1: When the timer is inactive, OC/OCN outputs are enabled with their inactive level as soon as CCxE = 1 or CCxNE = 1. Then, OC/OCN enable output signal = 1. Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).
10	OSSI	rw	0x00	Off-state selection for Idle mode This bit is used when MOE = 0 on channels configured as outputs. See OC/OCN enable description for more details (Section 9.4.9: capture/compare enable register (TIMx_CCER)). 0: When the timer is inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0). 1: When the timer is inactive, OC/OCN outputs are forced first with their idle level as soon as CCxE = 1 or CCxNE = 1. Then, OC/OCN enable output signal = 1) Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).

Bit	Field	Type	Reset	Description
9:8	LOCK	rw	0x00	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF - No bit is write protected. 01: LOCK Level 1 = DTG, BKE, BKP and AOE bits in TIMx_BDTR register, OISx and OISxN bits in TIMx_CR2 register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIMx_CCER register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in TIMx_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written. Note: The LOCK bits can be written only once after the system reset. Once the TIMx_BDTR register has been written, their content is frozen until the next reset.
7:0	DTG	rw	0x00	Dead-time generator setup This bit-field defines the duration of the dead-time inserted between the complementary outputs. Assuming that DT corresponds to this duration. DTG[7: 5] = 0xx: DT = (DTG[7: 0] + 1) × t_{dtg}, t_{dtg} = t_{DTS}; DTG[7: 5] = 10x: DT = (DTG[5: 0] + 1 + 64) × t_{dtg}, t_{dtg} = 2 × t_{DTS}; DTG[7: 5] = 110: DT = (DTG[4: 0] + 1 + 32) × t_{dtg}, t_{dtg} = 8 × t_{DTS}; DTG[7: 5] = 111: DT = (DTG[4: 0] + 1 + 32) × t_{dtg}, t_{dtg} = 16 × t_{DTS}; Example if t_{DTS}=125ns (8MHz), dead-time possible values are: 125ns to 15875ns by 125ns steps, 16µs to 31750ns by 250ns steps, 32µs to 63µs by 1µs steps, 64µs to 126µs by 2µs steps Note: This bit-field can not be modified as long as LOCK level 1, 2 or 3 have been programmed (LOCK bits in TIMx_BDTR register).

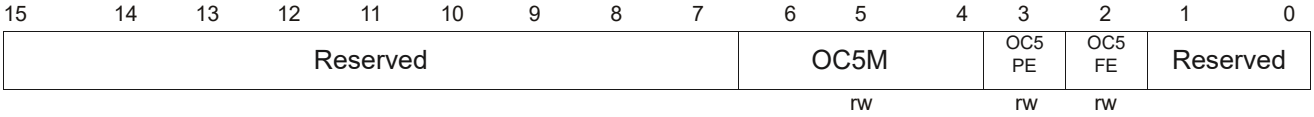
9.4.19 Capture/compare mode register 3 (TIMx_CCMR3)

Address offset: 0x54

Reset value: 0x0000

(TIM1)

The channels can only be used in output (compare mode).



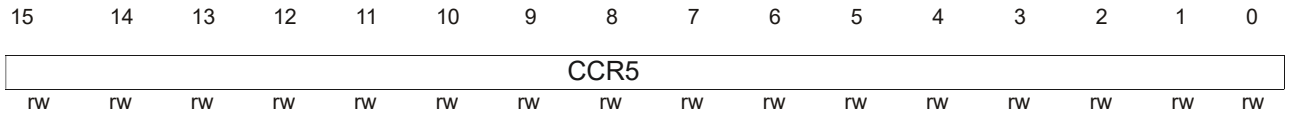
Output compare mode:

Bit	Field	Type	Reset	Description
15:7	Reserved			Reserved and always read as 0.
6:4	OC5M	rw	0x00	Output compare 5 mode
3	OC5PE	rw	0x00	Output compare 5 preload enable
2	OC5FE	rw	0x00	Output compare 5 fast enable
1:0	Reserved			Reserved and always read as 0.

9.4.20 Capture/compare register 5 (TIMx_CCR5)

Address offset: 0x58

Reset value: 0x0000



Bit	Field	Type	Reset	Description
15:0	CCR5	rw	0x0000	Capture/Compare 5 value CC5 channel can only be configured as output: CCR5 is the value to be loaded in the actual capture/compare 5 register (preload value). It is loaded immediately if the preload feature is not selected in the TIMx_CCMR3 register (bit OC5PE). Else the preload value is copied in the active capture/compare 5 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC5 output.

9.4.21 PWM phase-shift update request enable register (TIMx_PDER)

Address offset: 0x5C
Reset value: 0x0000

(TIM1)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										CCR5_SHIFT_EN	CCR4_SHIFT_EN	CCR3_SHIFT_EN	CCR2_SHIFT_EN	CCR1_SHIFT_EN	Res.
										rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
15:6	Reserved			Reserved, always read as 0
5	CCR5_SHIFTEN	rw	0x00	Allow channel 5 to output PWM phase shift enable bit 0: disable channel 5 to output PWM phase shift 1: enable channel 5 to output PWM phase shift For details, see the phase shift description for CCRx_FALL register.
4	CCR4_SHIFTEN	rw	0x00	Allow channel 4 to output PWM phase shift enable bit 0: disable channel 4 to output PWM phase shift 1: enable channel 4 to output PWM phase shift For details, see the phase shift description for CCRx_FALL register.
3	CCR3_SHIFTEN	rw	0x00	Allow channel 3 to output PWM phase shift enable bit 0: disable channel 3 to output PWM phase shift 1: enable channel 3 to output PWM phase shift For details, see the phase shift description for CCRx_FALL register.
2	CCR2_SHIFTEN	rw	0x00	Allow channel 2 to output PWM phase shift enable bit 0: disable channel 2 to output PWM phase shift 1: enable channel 2 to output PWM phase shift For details, see the phase shift description for CCRx_FALL register.
1	CCR1_SHIFTEN	rw	0x00	Allow channel 1 to output PWM phase shift enable bit 0: disable channel 1 to output PWM phase shift 1: enable channel 1 to output PWM phase shift For details, see the phase shift description for CCRx_FALL register.
0	Reserved			Reserved and always read as 0.

9.4.22 PWM phase-shift downcounting capture/compare register (CCRxFALL)

Address offset: 0x60 ~ 0x70

Reset value: 0x0000

(TIM1)

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

Bit	Field	Type	Reset	Description
15:0	CCRxFALL	rw	0x0000	Channel x capture/compare value when downcounting in the PWM center-aligned mode PWM phase-shift function: To realize PWM outputs with programmable phase-shift waveform (left shift or right shift as required), the user should enable the PWM phase-shift function in the PDER register and set the CCRxFALL and CCRx.

10 16-Bit General-Purpose Timer (TIM3)

16-Bit General-Purpose Timer (TIM3)

10.1. TIMx introduction

The general-purpose timers consist of a 16-bit auto-reload counter driven by a programmable prescaler. They may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare and PWM). Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers.

The TIMx timers are completely independent, and do not share any resources. They can be synchronized together.

10.2. TIMx main features

General-purpose TIMx timer features include:

- 16-bit up, down, up/down auto-reload counter.
- 16-bit programmable prescaler allowing dividing (also “on the fly”) the counter clock frequency either by any factor between 1 and 65536.
- Up to 3 independent channels for:
  - Input capture
  - Output comparison
  - PWM generation (Edge and Center-aligned Mode)
  - One-pulse mode output
- Synchronization circuit to control the timer with external signals and to interconnect several timers together
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes
- Trigger input for external clock or cycle-by-cycle current management

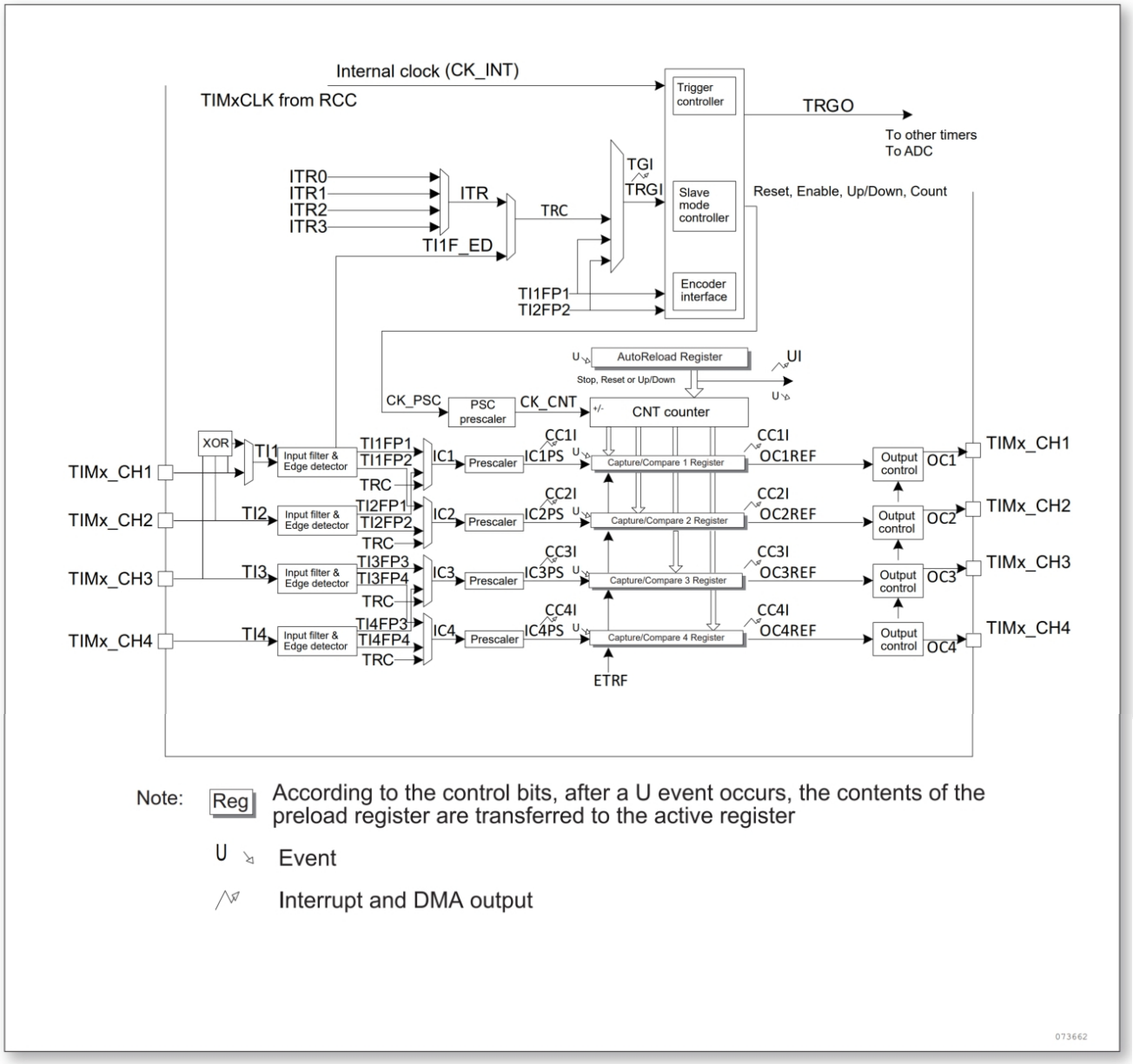


Figure 73. General-purpose timer block diagram

10.3. TIMx functional description

10.3.1 Time-base unit

The main block of the programmable general-purpose timer is a 16-bit counter with its related auto-reload register. The counter can count up, down or both up and down. The counter clock can be divided by a prescaler. The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running. The time-base unit includes:

- Counter register (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIMx_CR1 register. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software. The generation of the update event is described in detailed for each configuration.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIMx_CR1 register is set (refer also to the slave mode controller description to get more details on counter enabling).

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered.

The new prescaler ratio is taken into account at the next update event.

The figures below give some examples of the counter behavior when the prescaler ratio is changed on the fly:

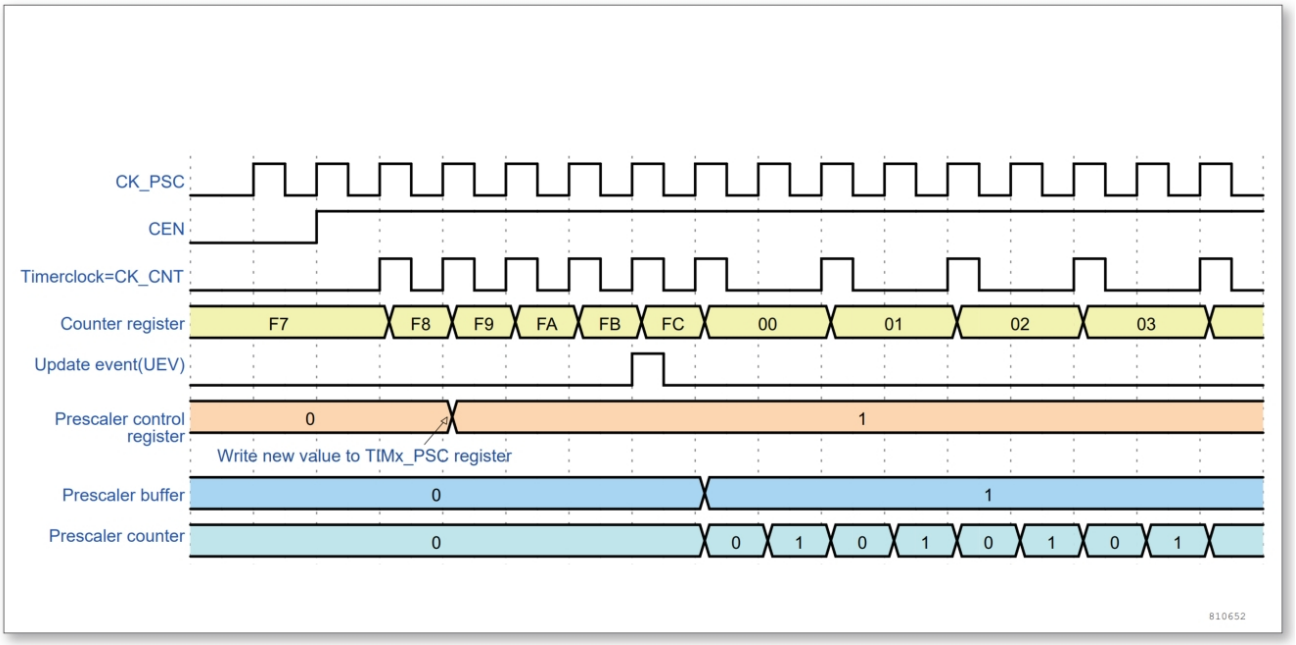


Figure 74. Counter timing diagram with prescaler division change from 1 to 2

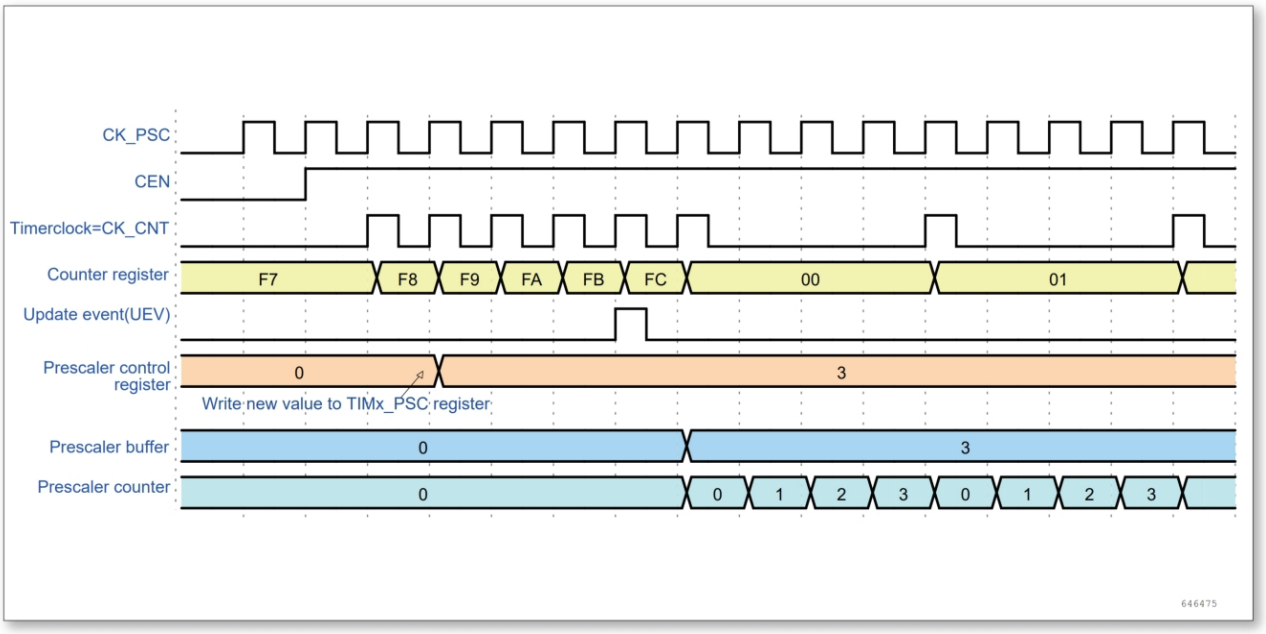


Figure 75. Counter timing diagram with prescaler division change from 1 to 4

10.3.2 Counter modes

Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the TIMx_ARR counter), then restarts from 0 and generates a counter overflow event.

An Update event can be generated at each counter overflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller).

The UEV event can be disabled by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change) when an update event should be generated. In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag by hardware (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture mode.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set by hardware (depending on the URS bit).

- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).
- The auto-reload shadow register is updated with the preload value (TIMx_ARR).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR = 0x36:

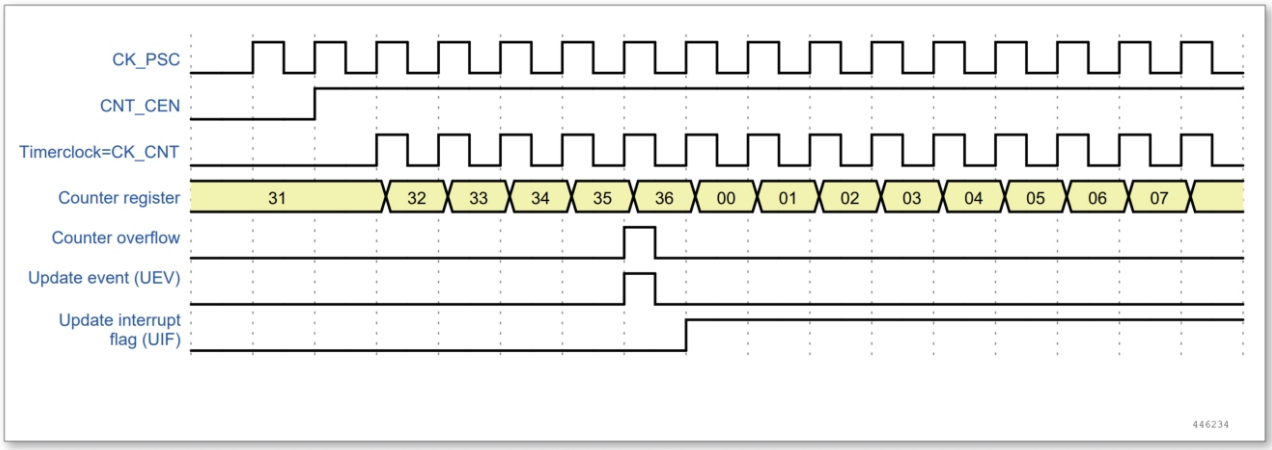


Figure 76. Counter timing diagram, internal clock divided by 1

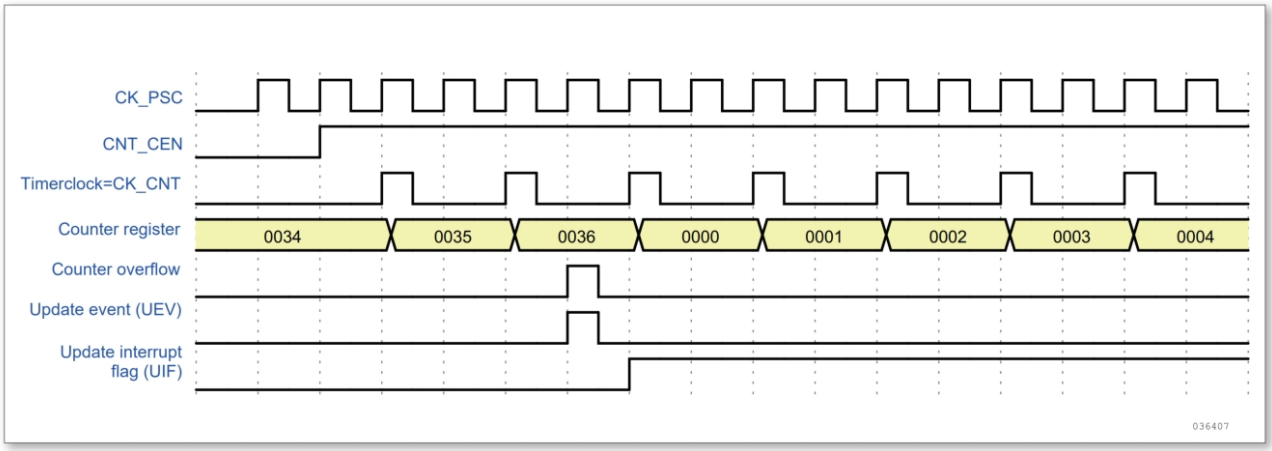


Figure 77. Counter timing diagram, internal clock divided by 2

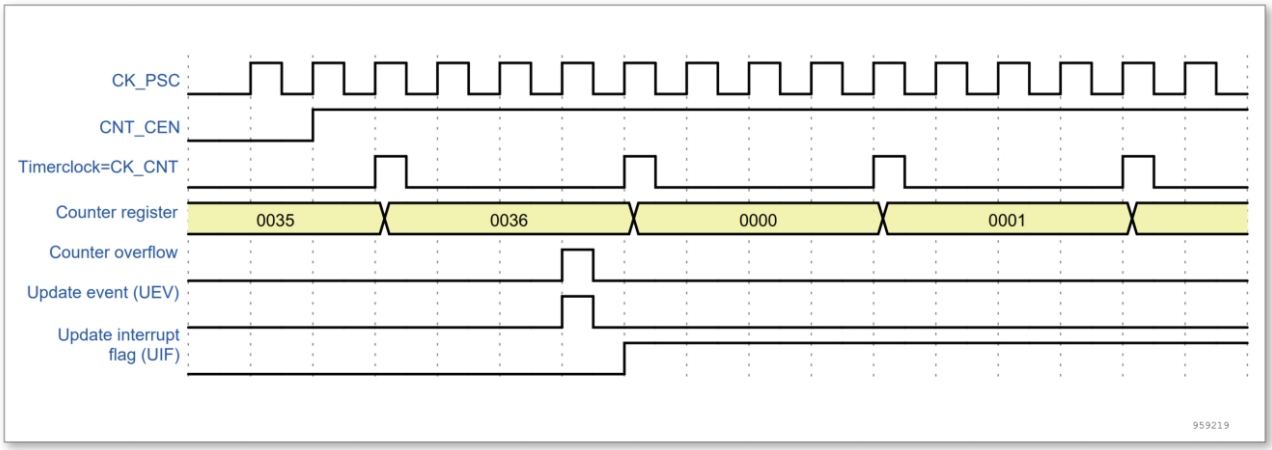


Figure 78. Counter timing diagram, internal clock divided by 4

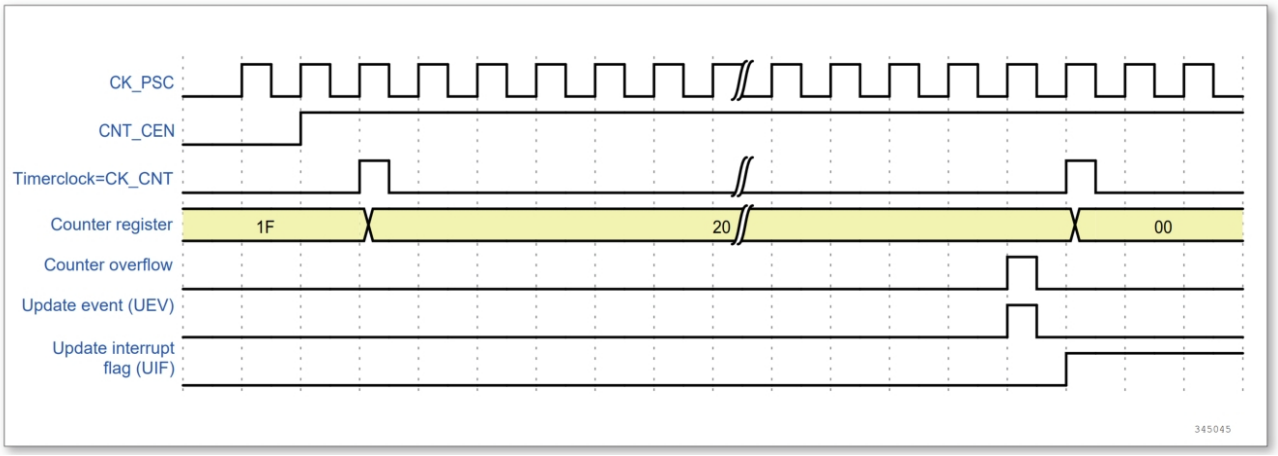


Figure 79. Counter timing diagram, internal clock divided by N

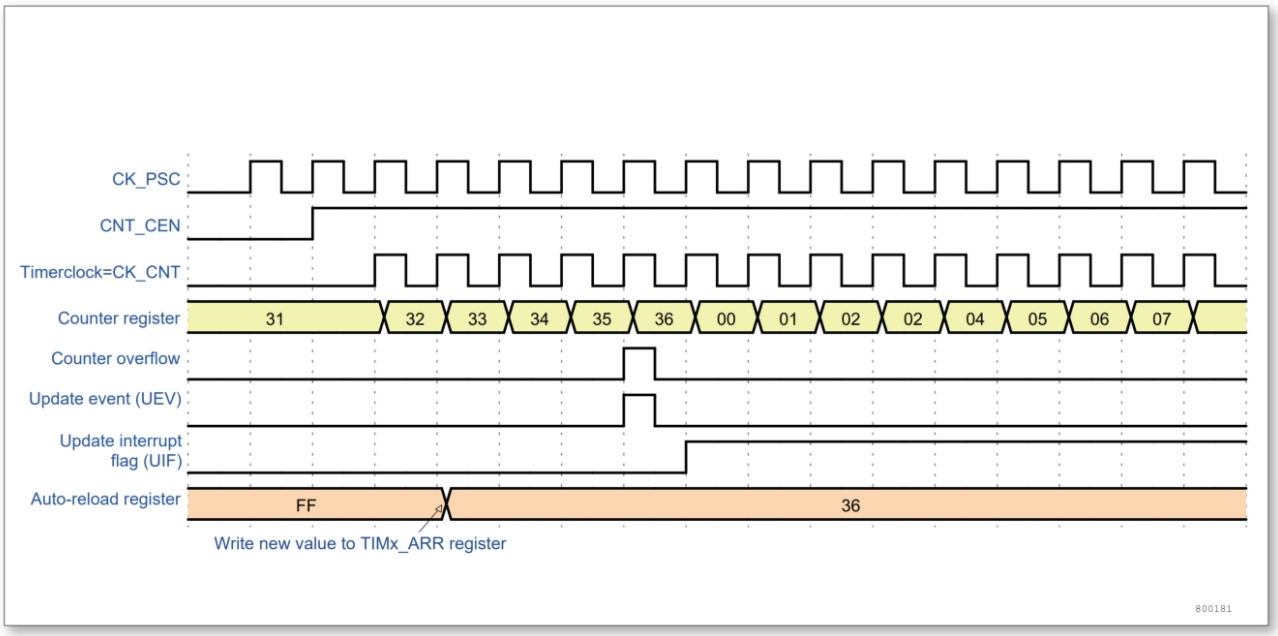


Figure 80. Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

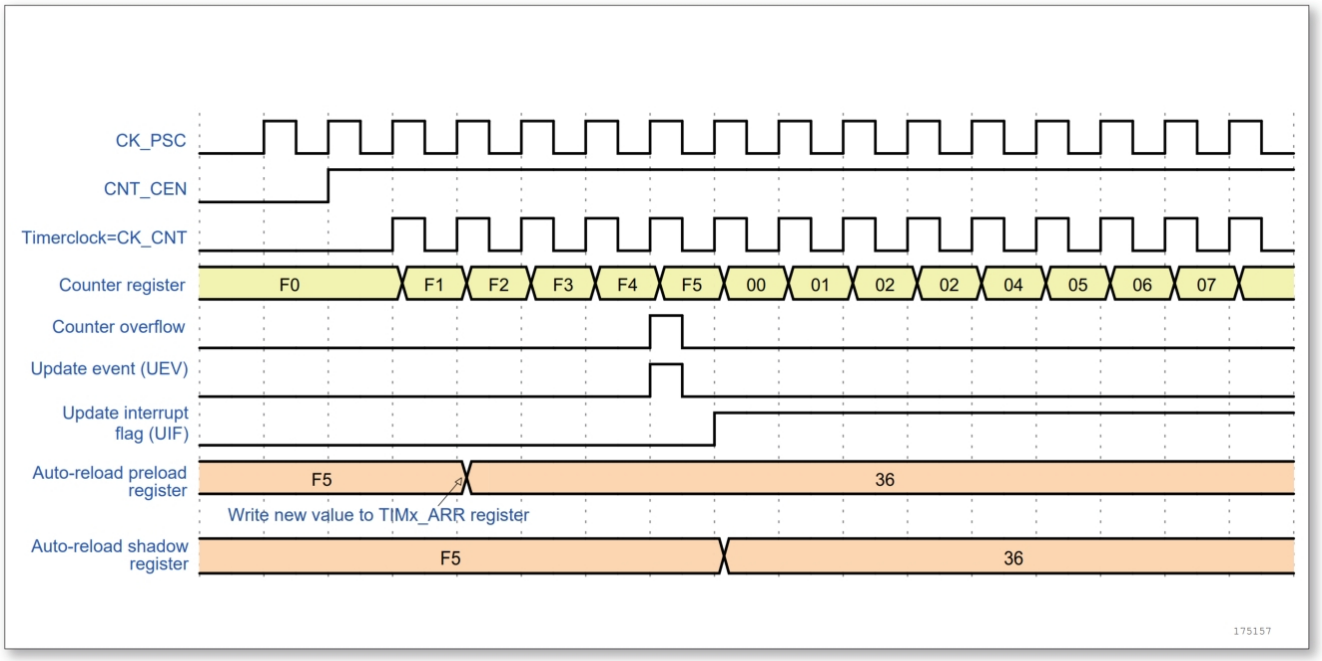


Figure 81. Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

Downcounting mode

In downcounting mode, the counter counts from the auto-reload value (content of the TIMx_ARR register) down to 0, then restarts from the auto-reload value and generates a counter underflow event.

An Update event can be generated at each counter overflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller).

The UEV update event can be disabled by setting the UDIS bit in TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescale rate doesn't change).

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit).

- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIMx_ARR register).

Note that the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR = 0x36.

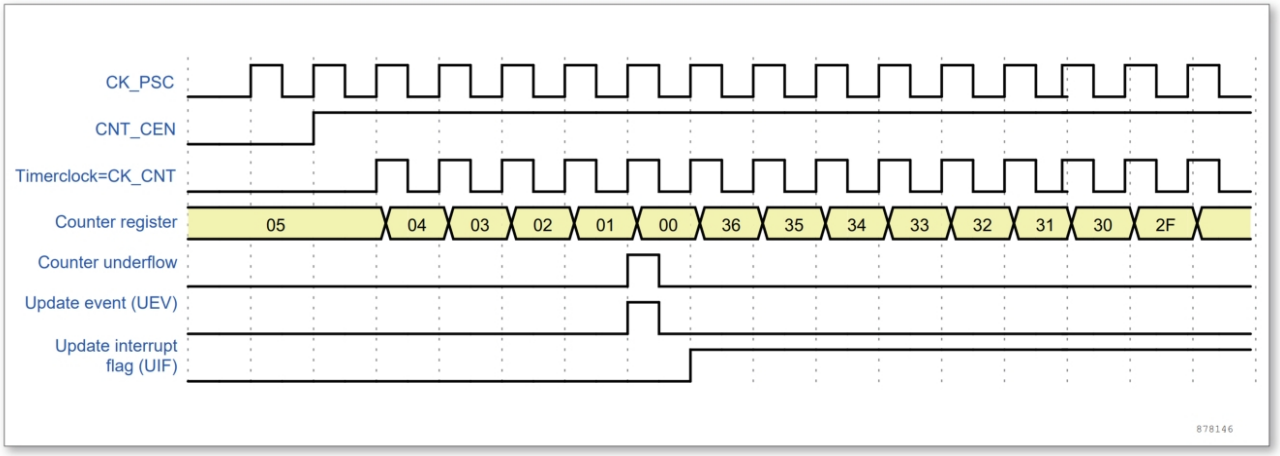


Figure 82. Counter timing diagram, internal clock divided by 1

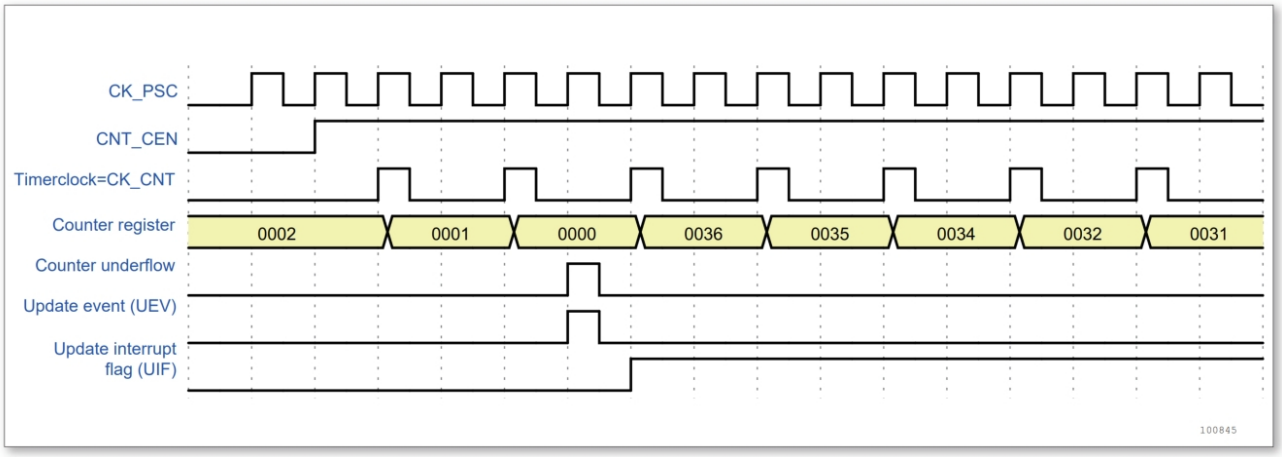


Figure 83. Counter timing diagram, internal clock divided by 2

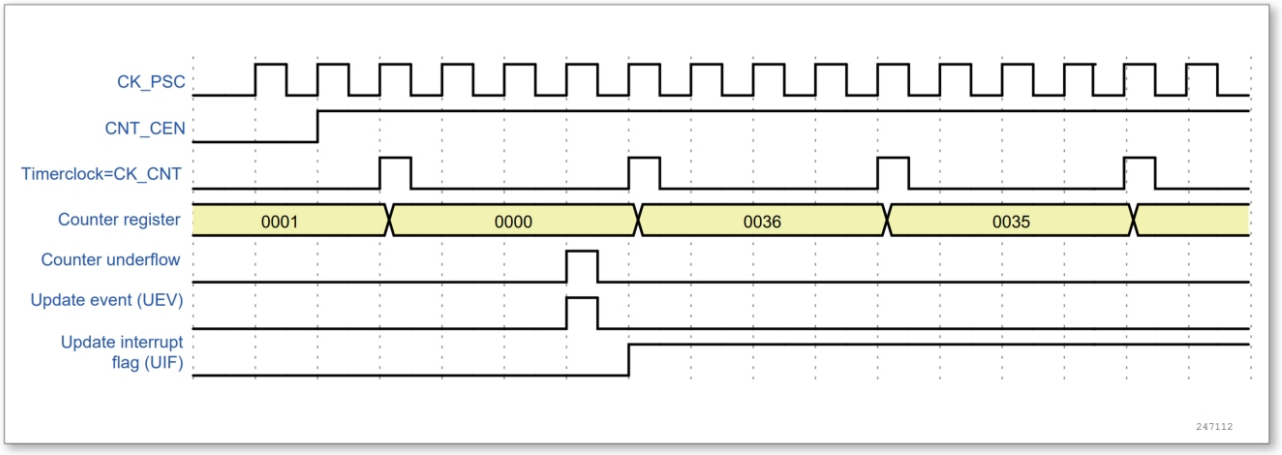


Figure 84. Counter timing diagram, internal clock divided by 4

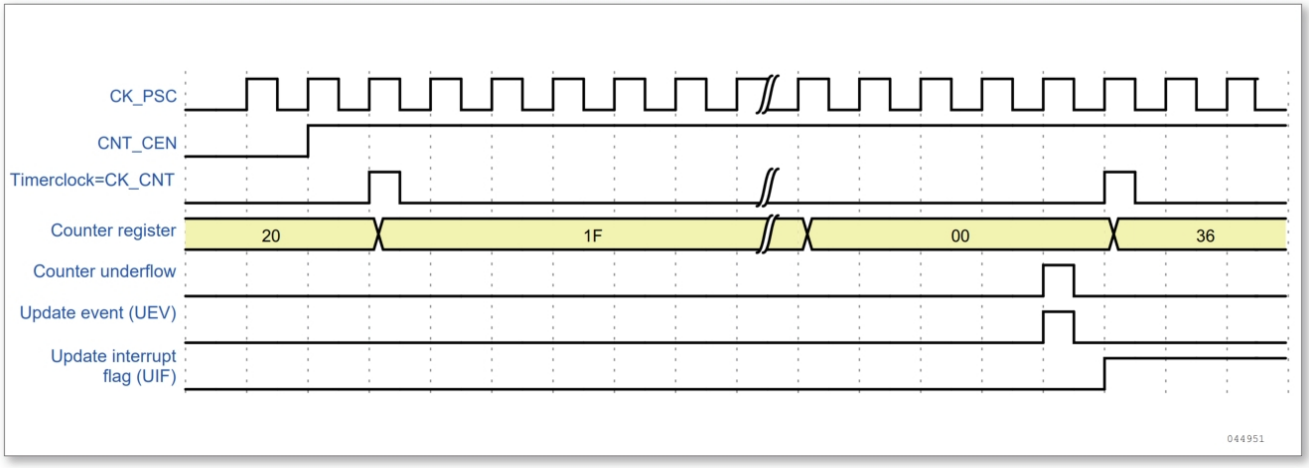


Figure 85. Counter timing diagram, internal clock divided by N

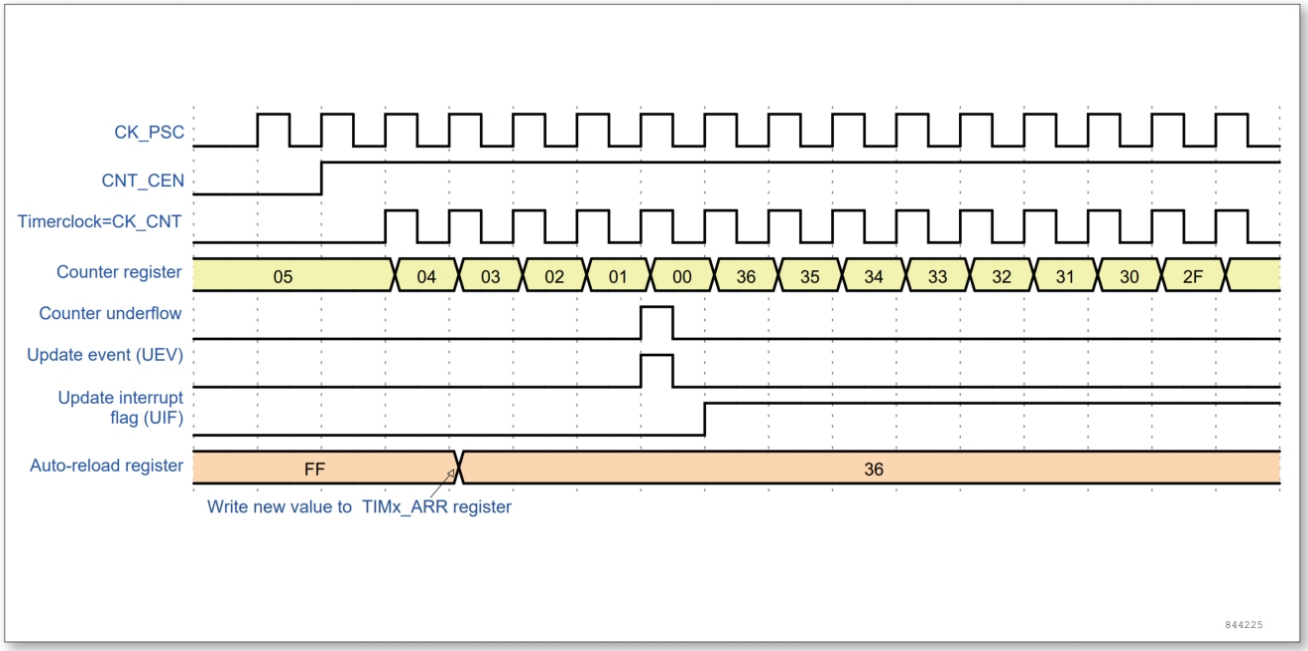


Figure 86. Counter timing diagram, update event when repetition counter is not used

Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the TIMx_ARR register) – 1, generates a counter overflow event, then counts from the auto-reload value down to 1 and generates a counter underflow event. Then it restarts counting from 0.

In this mode, the DIR direction bit in the TIMx_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter. The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller). In this case, the counter restarts counting from 0, as well as the counter of the prescaler.

The UEV update event can be disabled by setting the UDIS bit in TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter continues counting up and down, based on the current auto-reload value.

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt

is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit).

- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register.
- The auto-reload active register is updated with the preload value (content of the TIMx_ARR register).

Note that if the update source is a counter overflow, the auto-reload is updated before the counter is reloaded, so that the next period is the expected one (the counter is loaded with the new value).

The following figures show some examples of the counter behavior for different clock frequencies:

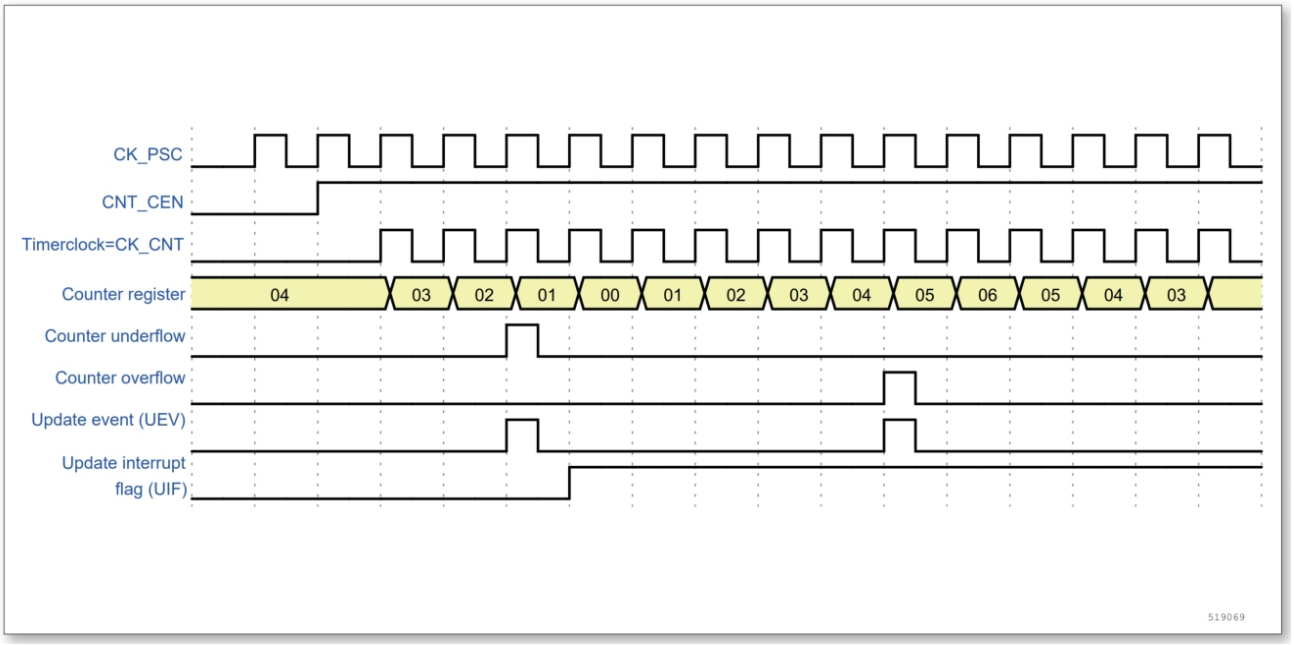


Figure 87. Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6

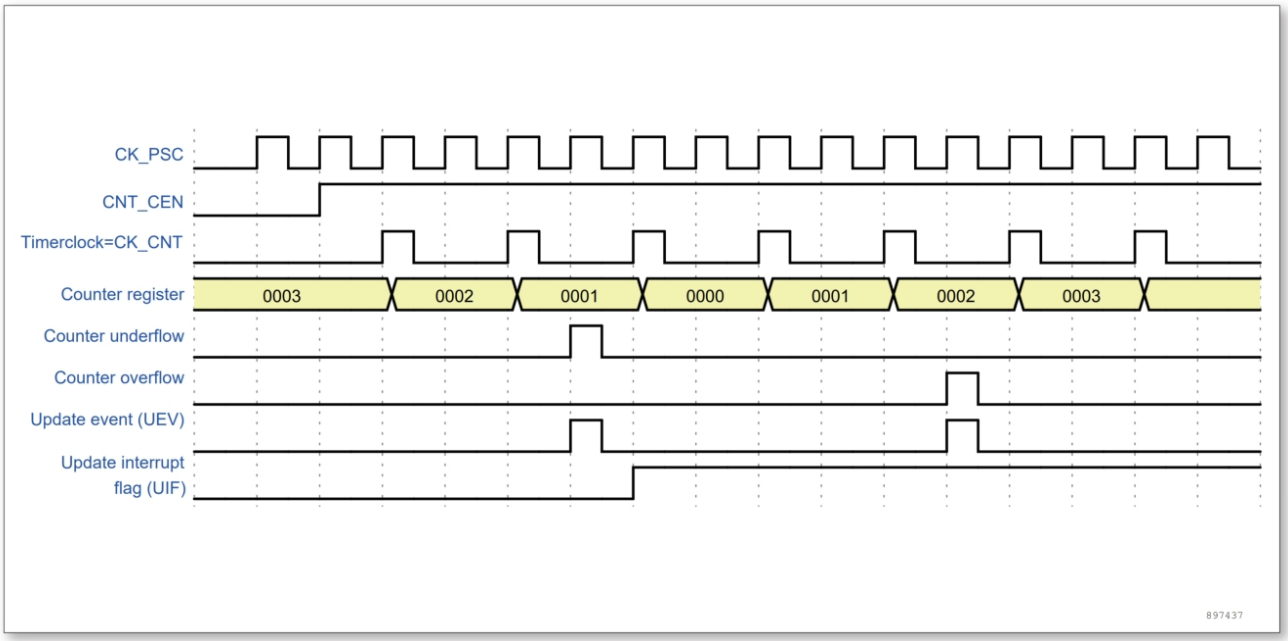


Figure 88. Counter timing diagram, internal clock divided by 2

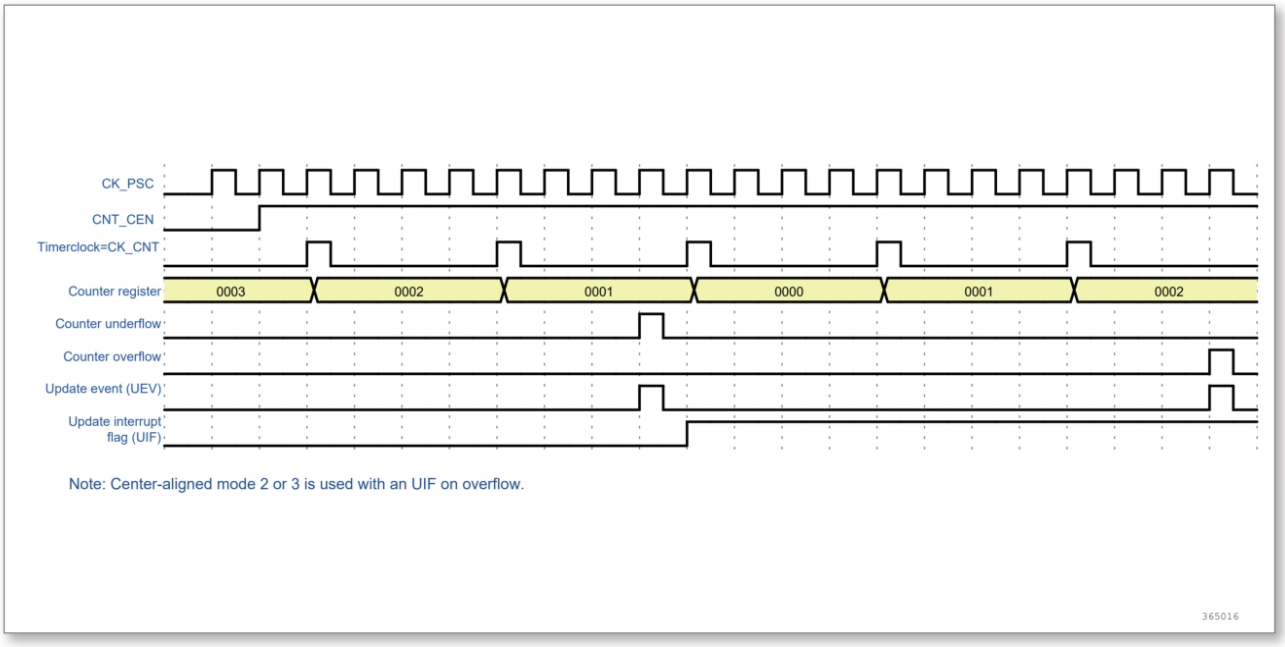


Figure 89. Counter timing diagram, internal clock divided by 4, TIMx_ARR=0x03

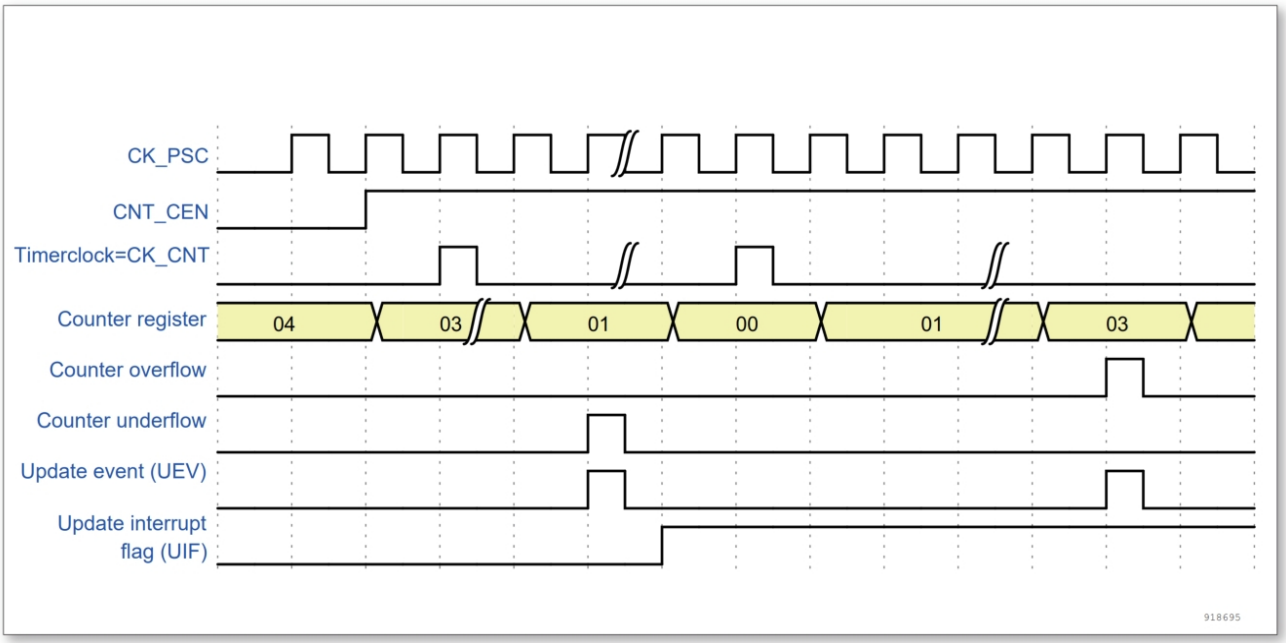


Figure 90. Counter timing diagram, internal clock divided by N

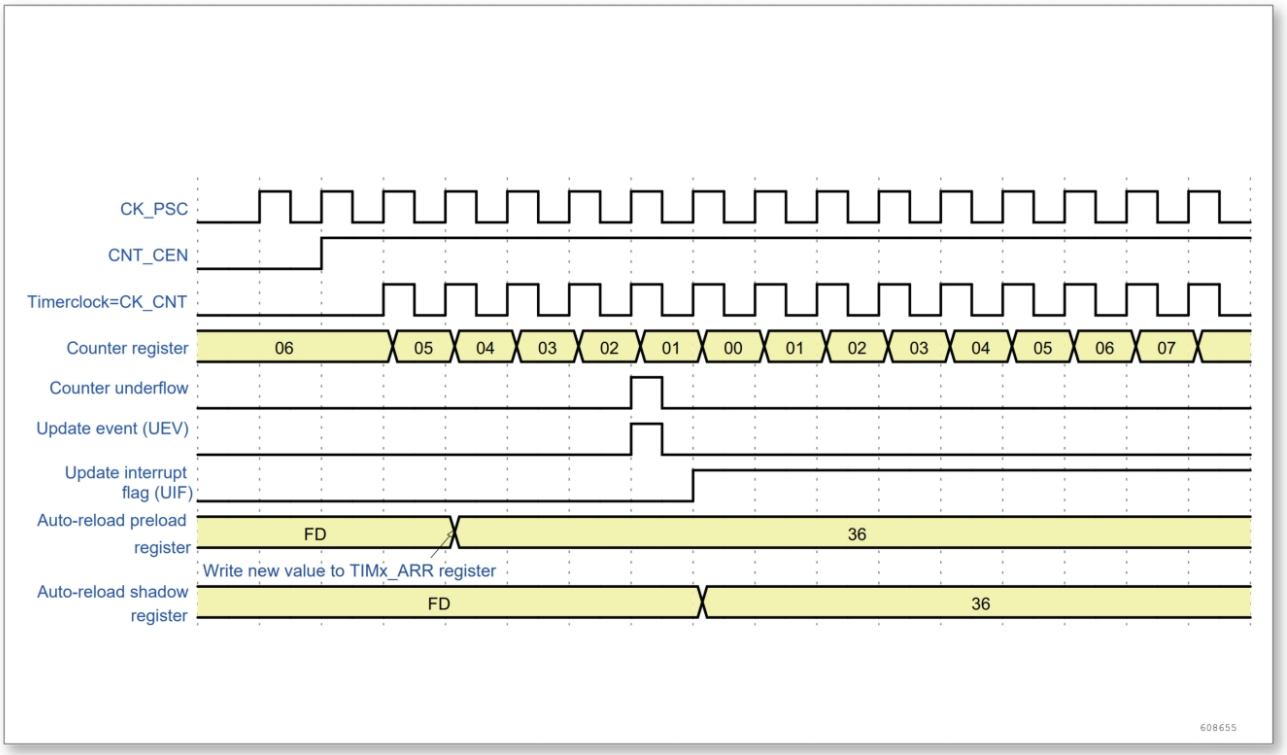


Figure 91. Counter timing diagram, update event with ARPE=1 (counter underflow)

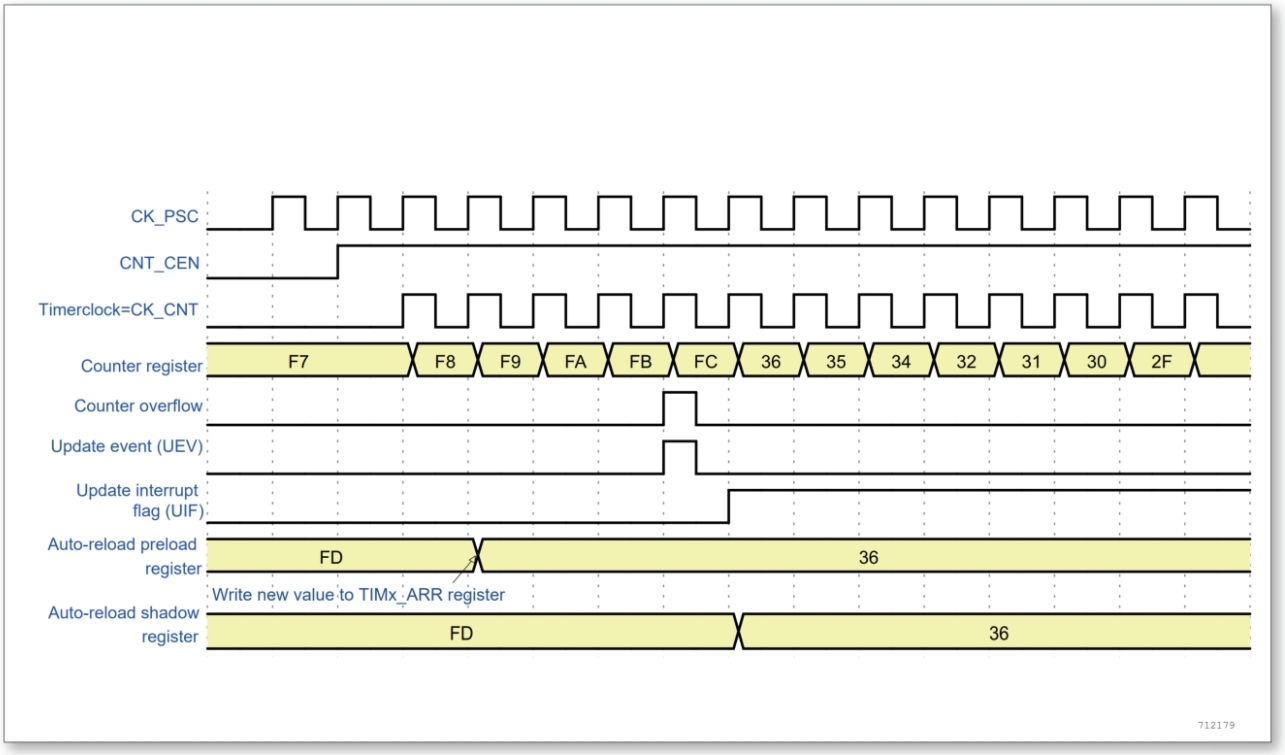


Figure 92. Counter timing diagram, update event with ARPE=1 (counter overflow)

10.3.3 Clock selection

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT).
- External clock mode 1: external input pin (TIx).

- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, the user can configure Timer 1 to act as a prescaler for Timer 3.

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS = 000), then the CEN, DIR (in the TIMx_CR1 register) and UG bits (in the TIMx_ EGR register) are actual control bits and can be changed only by software (except UG which remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

The figure below shows the behavior of the control circuit and the upcounter in normal mode, without prescaler.

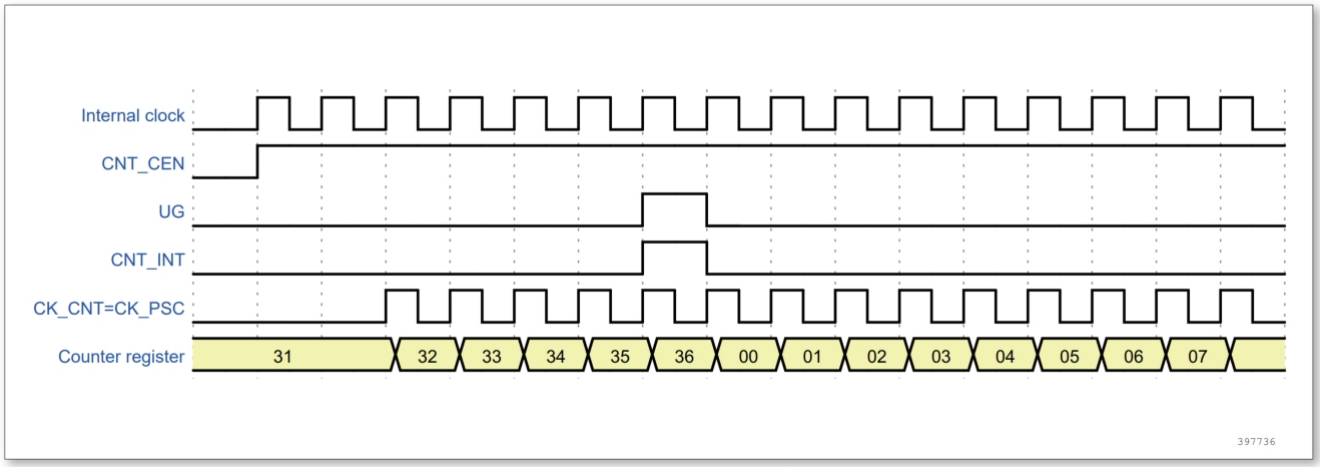


Figure 93. Control circuit in normal mode, internal clock divided by 1

External clock source mode 1

This mode is selected when SMS=111 in the TIMx_SMCR register. The counter can count at each rising or falling edge on a selected input.

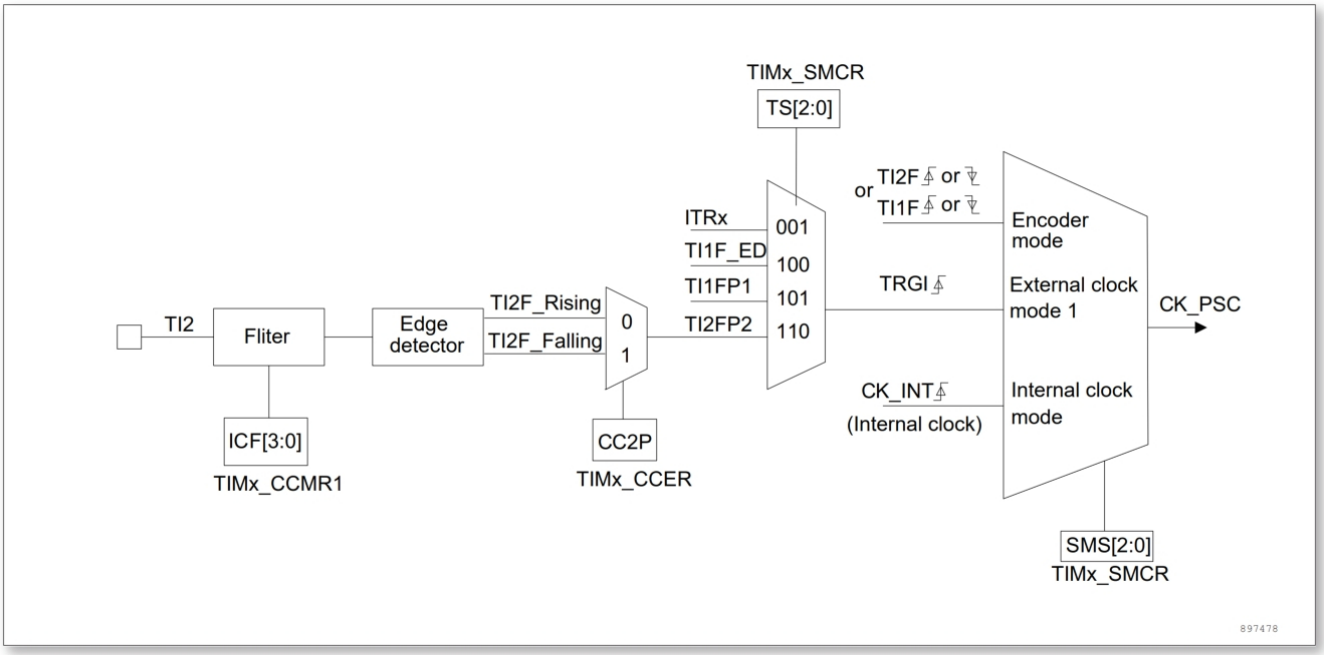


Figure 94. TI2 external clock connection example

For example, to configure the upcounter to count in response to a rising edge on the TI2 input, use the following procedure:

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = 01 in the TIMx_CCMR1 register.
2. Configure the input filter duration by writing the IC2F[3:0] bits in the TIMx_CCMR1 register (if no filter is needed, keep IC2F = 0000).

   Note: The capture prescaler is not used for triggering, so the user does not need to configure it.
3. Select rising edge polarity by writing CC2P = 0 in the TIMx_CCER register.
4. Configure the timer in external clock mode 1 by writing SMS = 111 in the TIMx_SMCR register.
5. Select TI2 as the trigger input source by writing TS = 110 in the TIMx_SMCR register.
6. Enable the counter by writing CEN = 1 in the TIMx_CR1 register.

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge on TI2 and the actual clock of the counter is due to the resynchronization circuit on TI2 input.

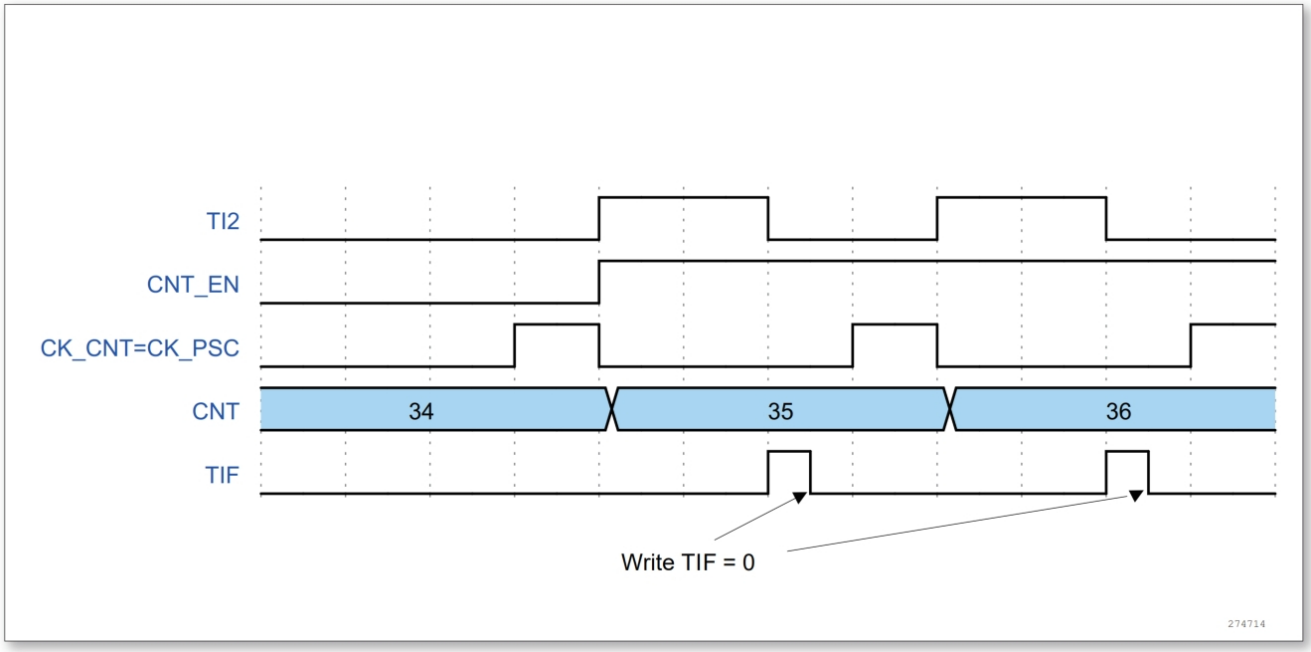


Figure 95. Control circuit in external clock mode 1

10.3.4 Capture/compare channel

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), a input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The following figures give an overview of one Capture/Compare channel. The input stage samples the corresponding TIx input to generate a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

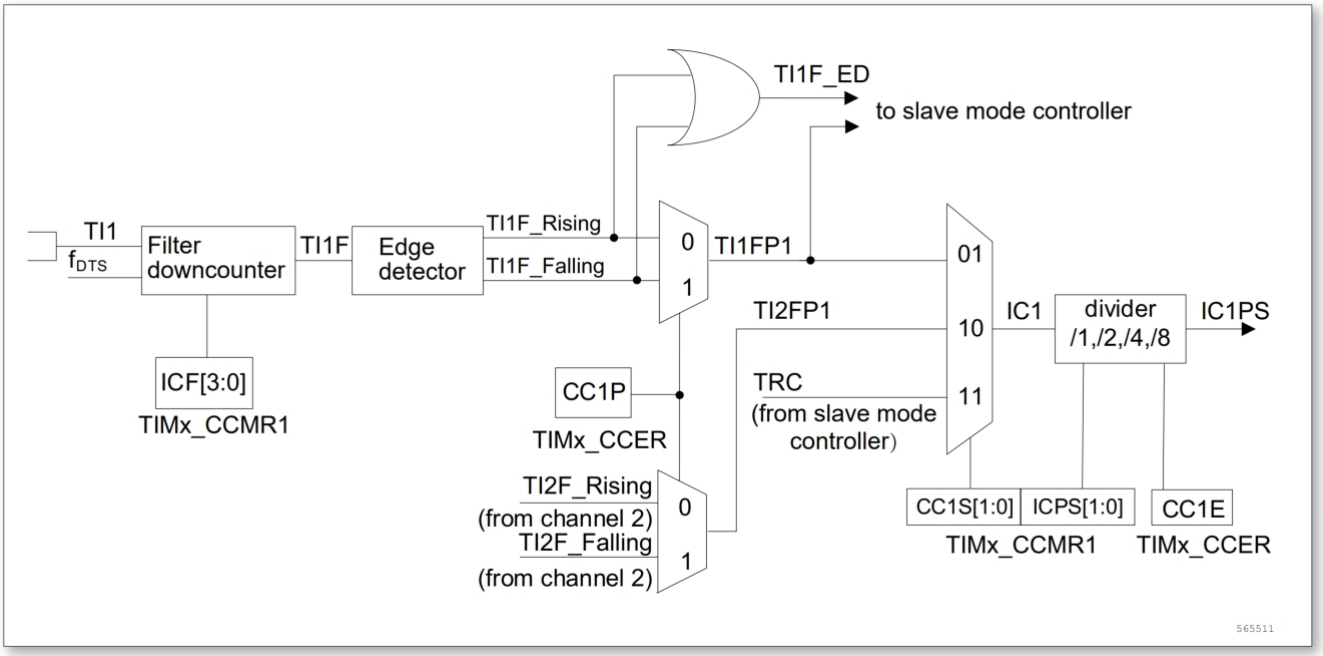


Figure 96. Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

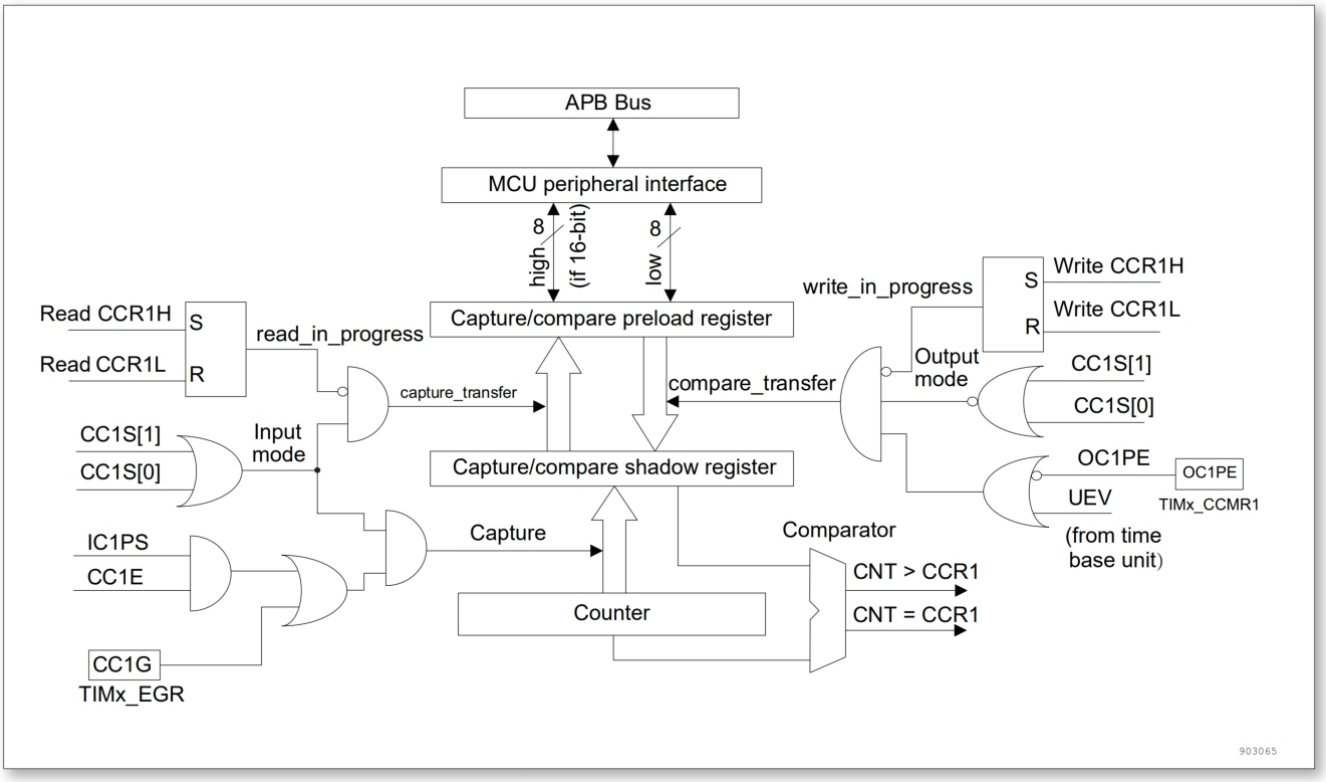


Figure 97.Capture/compare channel 1 main circuit

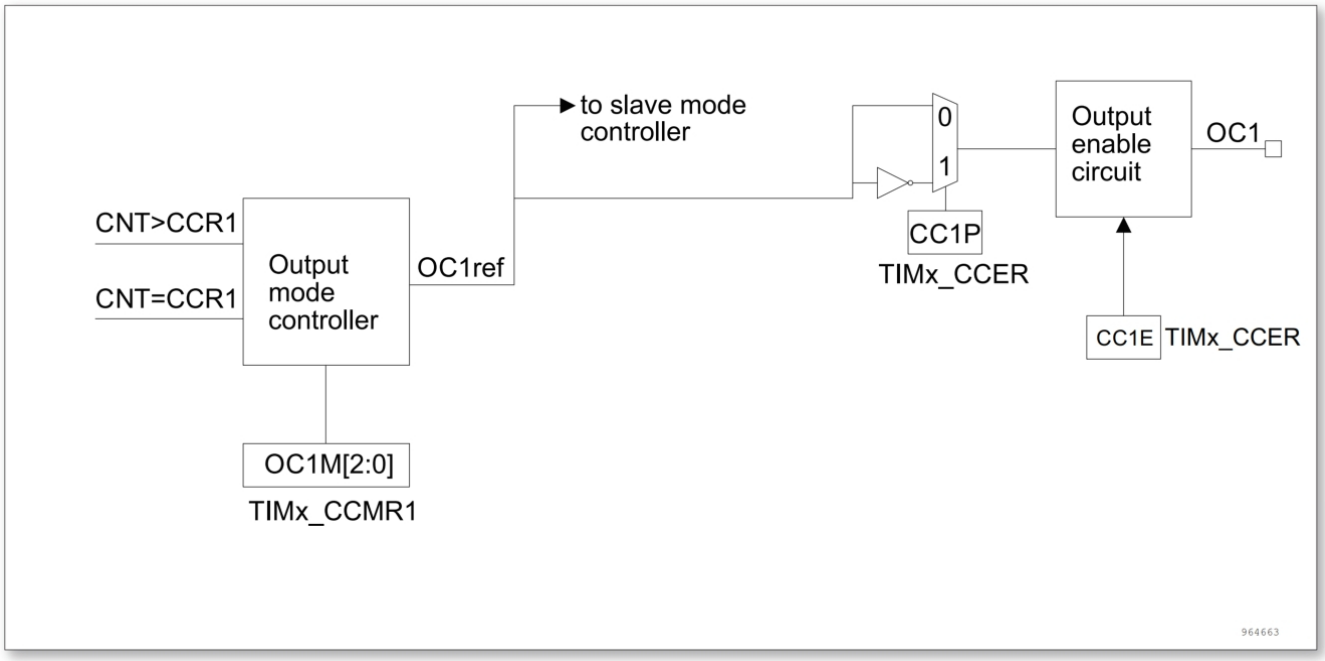


Figure 98. Output stage of capture/compare channel (channel 1)

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register. In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

10.3.5 Input capture mode

In Input capture mode, the Capture/Compare registers (TIMx_CCRx) are used to latch the value of the counter after an edge is detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIMx_SR register) is set and an interrupt can be sent if it is enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when written to '0'. The following example shows how to capture the counter value in TIMx_CCR1 when TI1 input rises. To do this, use the following procedure:

- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TM1_CCR1 register becomes read-only.
- Program the needed input filter duration with respect to the input signal (by programming ICxF bits in the TIMx_CCMRx register if the input is a TIx input). Let's imagine that, when toggling, the input signal is not stable during at most five internal clock cycles. We must program a filter duration longer than these five clock cycles. We can validate an edge transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at f_{DTS} frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIMx_CCER register (rising edge in this case).

- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit to '1' in the TIMx_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active level transition.
- CC1IF flag is set (interrupt flag). CC1IF flag was not cleared if at least two consecutive captures occurred.
- CC1OF is also set to '1'.
- An interrupt is generated depending on the CC1IE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

10.3.6 PWM input mode

This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on the same TIx input
- These 2 ICx signals are active on edges with opposite polarity
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode

For example, the user can measure the period (in TIMx_CCR1 register) and the duty cycle (in TIMx_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):

- Select the active input for TIMx_CCR1: write the CC1S bits to 01 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to '0' (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2P bit to '1' (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMx_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to '1' in the TIMx_CCER register.

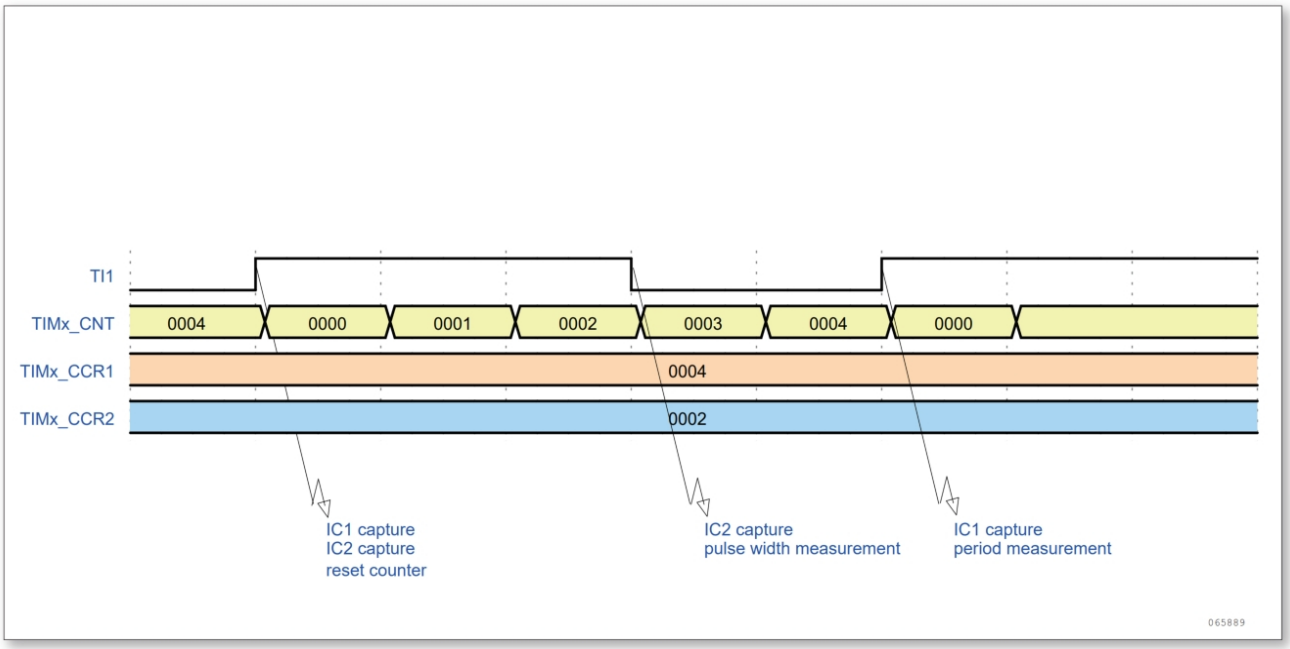


Figure 99. PWM input mode timing

The PWM input mode can be used only with the TIMx_CH1/TIMx_CH2 signals due to the fact that only TI1FP1 and TI2FP2 are connected to the slave mode controller.

10.3.7 Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compare signal (OCxREF and then OCx) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, the user just needs to write 101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx gets opposite value to CCxP polarity bit.

For example: CCxP = 0 (OCx active high) => OCx is forced to high level. The OCxREF signal can be forced low by writing the OCxM bit to 100 in the TIMx_CCMRx register. Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt requests can be sent accordingly. This is described in the output compare mode section below.

10.3.8 Output compare mode

This function is used to control an output waveform or indicate when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bit in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCxM = 000), be set active (OCxM = 001), be set inactive (OCxM = 010) or can toggle (OCxM = 011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).

- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register. In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The synchronization resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

The output compare mode is configured as follows:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example, the user must write OCxM = 011, OCxPE = 0, CCxP = 0 and CCxE = 1 to toggle OCx output pin when CNT matches CCRx, CCRx preload is not used, OCx is enabled and active high.
5. Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE = 0, else TIMx_CCRx shadow register is updated only at the next update event UEV). An example is given in the following figure.

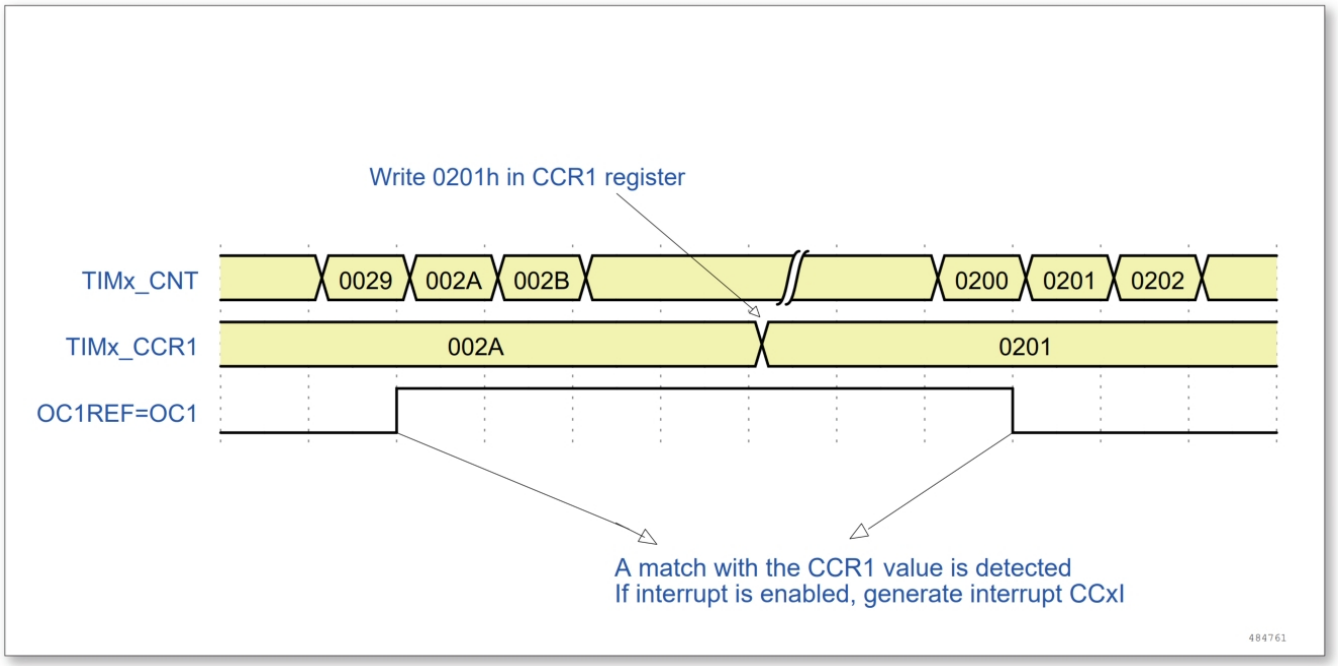


Figure 100. Output compare mode, toggle on OC1

10.3.9 PWM mode

Pulse Width Modulation mode allows generating a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing ‘110’ (PWM mode 1) or ‘111’ (PWM mode 2) in the OCxM bit in the TIMx_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMx_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, the user must initialize all the registers by setting the UG bit in the TIMx_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. OCx output is enabled by using the CCxE bit in the TIMx_CCER register. For details, refer to the TIMx_CCER register description.

In PWM mode (1 or 2), TIMx_CNT and TIM1_CCRx are always compared to determine whether TIM1_CCRx ≤ TIM1_CNT or TIM1_CNT ≤ TIM1_CCRx (depending on the direction of the counter).

The OCxREF signal can be generated when:

- the result of the comparison changes
- the output compare mode (OCxM bit in the TIMx_CCMRx register) switches from "frozen" mode (no comparison, OCxM = 000) to "PWM" mode (OCxM = 110 or 111).

Therefore, the PWM output can be forced on the fly by software. The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIMx_CR1 register.

PWM edge-aligned mode

Upcounting configuration

Upcounting is active when the DIR bit in the TIMx_CR1 register is low.

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx; else it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR), then OCxREF is held at '1'. If the compare value is 0 then OCxRef is held at '0'. The following figure shows some edge-aligned PWM waveforms in an example where TIMx_ARR = 8.

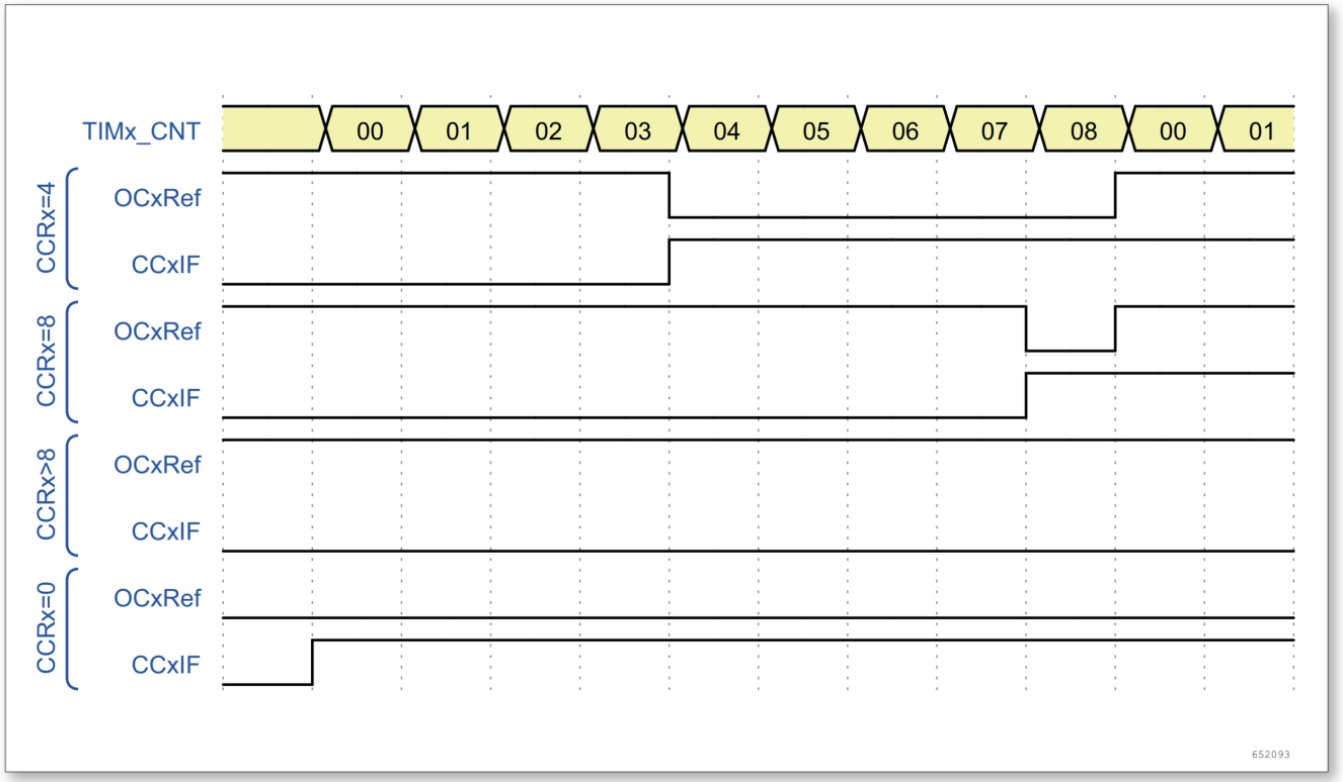


Figure 101. Edge-aligned PWM waveforms (ARR = 8)

Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high.

In PWM mode 1, the reference signal OCxRef is low as long as TIMx_CNT > TIMx_CCRx; else it becomes high. If the compare value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, then OCxREF is held at ‘1’.

0% PWM is not possible in this mode.

PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are different from ‘00’ (all the remaining configurations having the same effect on the OCxRef/OCx signals). The compare flag is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the TIMx_CR1 register is updated by hardware and must not be changed by software. Refer to center-aligned mode section.

The figure below shows some center-aligned PWM waveforms in an example where:

- TIMx_ARR = 8
- PWM mode is the PWM mode 1
- The flag is set when the counter counts down corresponding to the center-aligned mode 1 selected for CMS = 01 in TIMx_CR1 register.

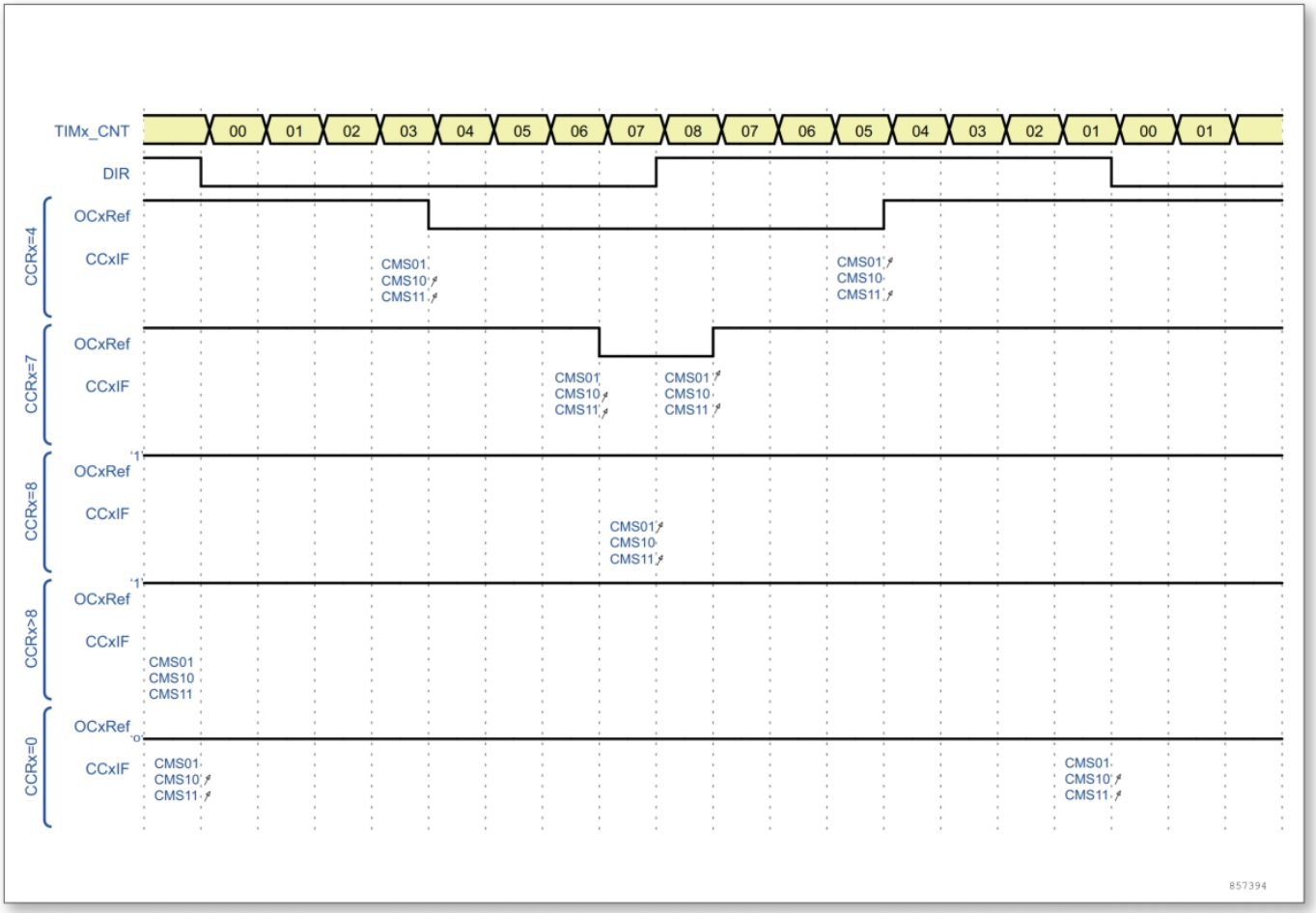


Figure 102. Center-aligned PWM waveforms (ARR = 8)

Hints on using center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIMx_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:
  - The direction is not updated if the user writes a value in the counter greater than the auto-reload value (TIMx_CNT > TIMx_ARR). For example, if the counter was counting up, it will continue to count up.
  - The direction is updated if the user writes 0 or writes the TIMx_ARR value in the counter but no Update Event UEV is generated.
- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIMx_EGR register) just before starting the counter and not to write the counter while it is running.

10.3.10 One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. Select One-pulse mode by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV.

A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

- In upcounting: CNT < CCRx ≤ ARR (in particular, 0 < CCRx)
- In downcounting: CNT > CCRx

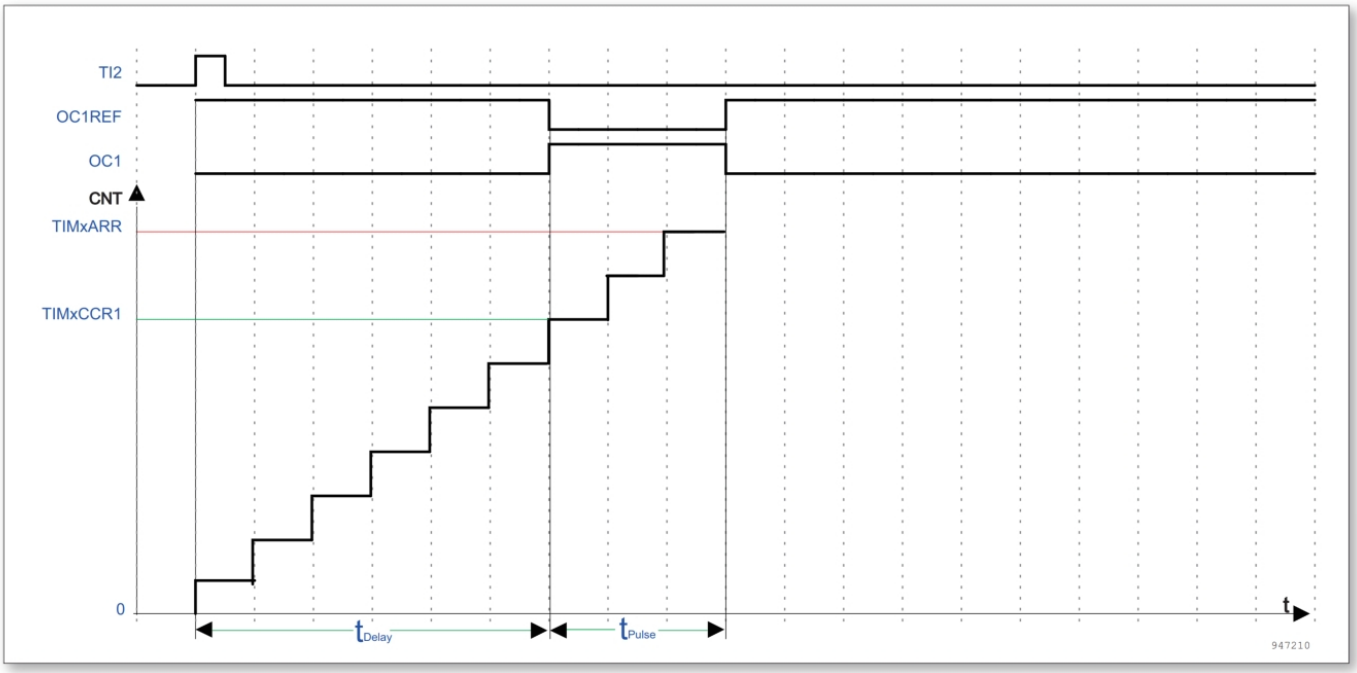


Figure 103. Example of one pulse mode

For example the user may want to generate a positive pulse on OC1 with a length of t_{PULSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI2 input pin.

Let's use TI2FP2 as trigger 1:

- Map TI2FP2 to TI2 by writing CC2S to '01' in the TIMx_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P to '0' in the TIMx_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS to '110' in the TIMx_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to '110' in the TIMx_SMCR register (trigger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The t_{DELAY} is defined by the value written in the TIMx_CCR1 register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let us say the user wants to build a waveform with a transition from '0' to '1' when a compare match occurs and a transition from '1' to '0' when the counter reaches the auto-reload value.

  To do this, enable PWM mode 2 by writing OC1M = 111 in the TIMx_CCMR1 register. The user can optionally enable the preload registers by writing OC1PE = 1 in the TIMx_CCMR1 register and ARPE in the TIMx_CR1 register. In this case the compare value must be written in the TIMx_CCR1 register, the auto-reload value in the TIMx_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to '1' in this example.

In our example, the DIR and CMS bits in the TIMx_CR1 register should be low.

The user only wants one pulse, so '1' must be written in the OPM bit in the TIMx_CR1 register to stop the counter

at the next update event (when the counter rolls over from the auto-reload value back to 0).

Particular case: OCx fast enable:

In One-pulse mode, the edge detection on TIx input sets the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations and it limits the minimum delay t_{DELAY} min we can get.

If the user wants to output a waveform with the minimum delay, the OCxFE bit in the TIMx_CCMRx register must be set. Then OCxRef (and OCx) are forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2 mode.

10.3.11 Encoder Interface mode

To select Encoder Interface mode, write SMS to '001' in the TIMx_SMCR register if the counter is counting on TI2 edges only, SMS to '010' if it is counting on TI1 edges only and SMS to '011' if it is counting on both TI1 and TI2 edges.

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIMx_CCER register. When needed, the user can program the input filter as well.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Refer to table below. The counter is clocked by each valid transition on TI1FP1 or TI2FP2 (TI1 and TI2 after input filter and polarity selection, TI1FP1=TI1 if not filtered and not inverted, TI2FP2=TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit in TIMx_CR1 register written to '1'). The sequence of transitions of the two inputs is

evaluated and generates count pulses as well as the direction signal. Depending on the sequence, the counter counts up or down, the DIR bit in the TIMx_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIMx_ARR register (0 to ARR or ARR down to 0 depending on the direction). So user must configure TIMx_ARR before starting. In the same way, the capture, compare, prescaler, trigger output features continue to work as normal.

In this mode, the counter is modified automatically following the speed and the direction of the incremental encoder and its content, therefore, always represents the encoder's position. The count direction correspond to the rotation direction of the connected sensor. The following table summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Table 36. Counting direction versus encoder signals

Active edge	Level on opposite signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI1FP2 signal	
		Rising	Falling	Rising	Falling
Counting on TI1 only	High	Down	Up	No Count	No Count
	Low	Up	Down	No Count	No Count
Counting on TI2 only	High	No Count	No Count	Up	Down
	Low	No Count	No Count	Down	Up
Counting on TI1 and TI2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally be used to convert the encoder's differential outputs to digital signals. This greatly increases noise immunity. The third encoder output which indicates the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

The following figure gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points. For this example we assume that the configuration is the following:

- C1S = '01' (TIMx_CCMR1 register, IC1FP1 mapped on TI1).
- CC2S = '01' (TIMx_CCMR2 register, IC2FP2 mapped on TI2).
- CC1P = '0' (TIMx_CCER register, IC1FP1 non-inverted, IC1FP1 = TI1).
- CC2P = '0' (TIMx_CCER register, IC2FP2 non-inverted, IC2FP2 = TI2).
- SMS = '011' (TIMx_SMCR register, both inputs are active on both rising and falling edges).
- CEN = '1' (TIMx_CR1 register, counter enabled).

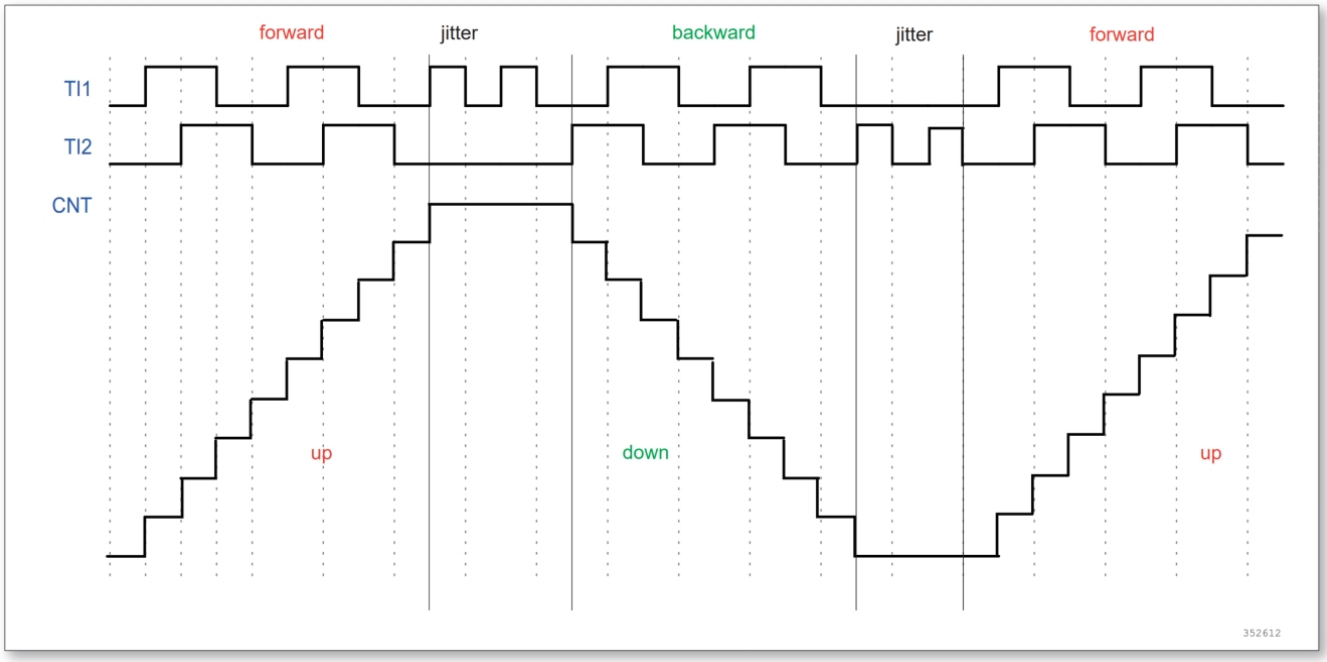


Figure 104. Example of counter operation in encoder interface mode

The following figure gives an example of counter behavior when IC1FP1 polarity is inverted (same configuration as above except CC1P = ’1’).

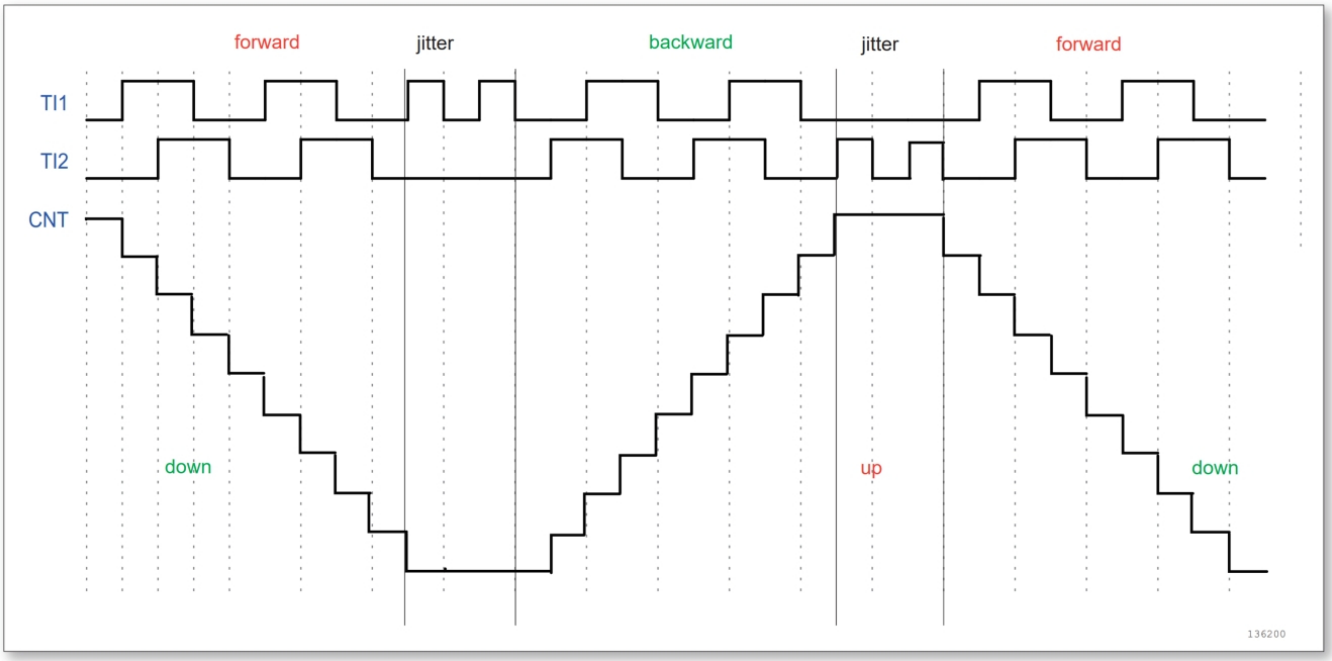


Figure 105. Example of encoder interface mode with IC1FP1 polarity inverted

The timer, when configured in Encoder Interface mode provides information on the sensor’s current position. The user can obtain dynamic information (speed, acceleration, deceleration) by measuring the period between two encoder events using a second timer configured in capture mode. The output of the encoder which indicates the mechanical zero can be used for this purpose. Depending on the time between two events, the counter can also be read at regular times. This can be done by latching the counter value into a third input capture register if available (then the capture signal must be periodic and can be generated by another timer).

10.3.12 Timer input XOR function

The TI1S bit in the TIMx_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIMx_CH1, TIMx_CH2 and TIMx_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture. An example of this feature used to interface Hall sensors is given in Section Advanced-Control Timer.

10.3.13 Timers and external trigger synchronization

The TIMx timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

Slave mode: Reset mode

The counter and its prescaler can be reinitialized in response to an event on a trigger input. Moreover, if the URS bit from the TIMx_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIMx_ARR, TIMx_CCRx) are updated.

- In the following example, the upcounter is cleared in response to a rising edge on TI1 input
- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC1S bits select the input capture source only, i.e. CC1S = 01 in the TIMx_CCMR1 register. Write CC1P = 0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS = 100 in TIMx_SMCR register. Select TI1 as the input source by writing TS = 101 in TIMx_SMCR register.
- Enable the counter by writing CEN = 1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. In the meantime, the trigger flag is set (TIF bit in the TIMx_SR register) and an interrupt request can be sent depending on the TIE (interrupt enable) bit in TIMx_DIER register.

The following figure shows this behavior when the auto-reload register TIMx_ARR = 0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

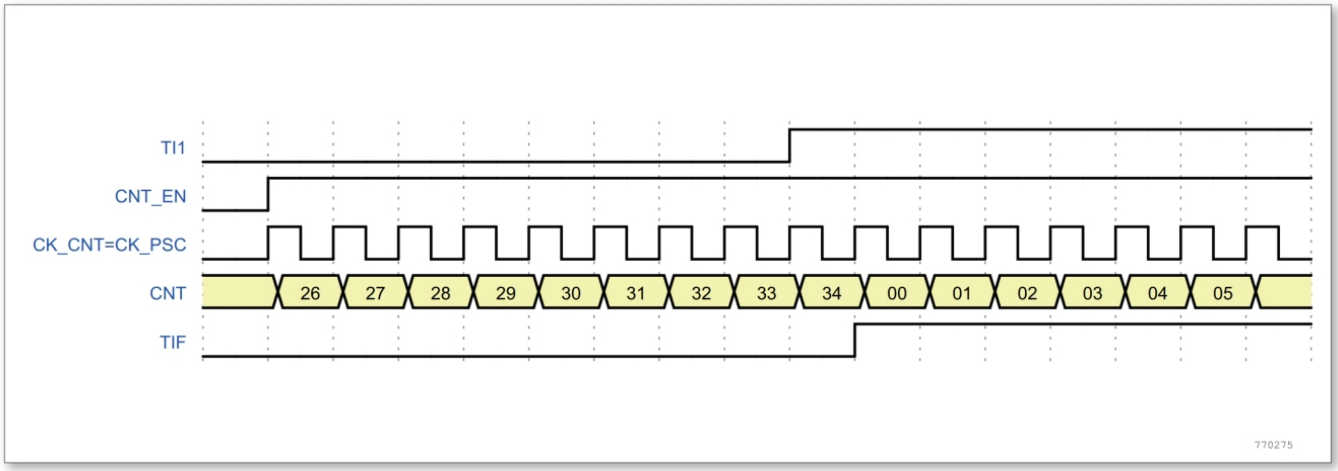


Figure 106. Control circuit in reset mode

Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so the user does not need to configure it. The CC1S bits select the input capture source only. Write CC1S = 01 in TIMx_CCMR1 register. Write CC1P = 1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS = 101 in TIMx_SMCR register. Select TI1 as the input source by writing TS = 101 in TIMx_SMCR register.
- Enable the counter by writing CEN = 1 in the TIMx_CR1 register (in gated mode, the counter doesn't start if CEN = 0, whatever is the trigger input level).

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

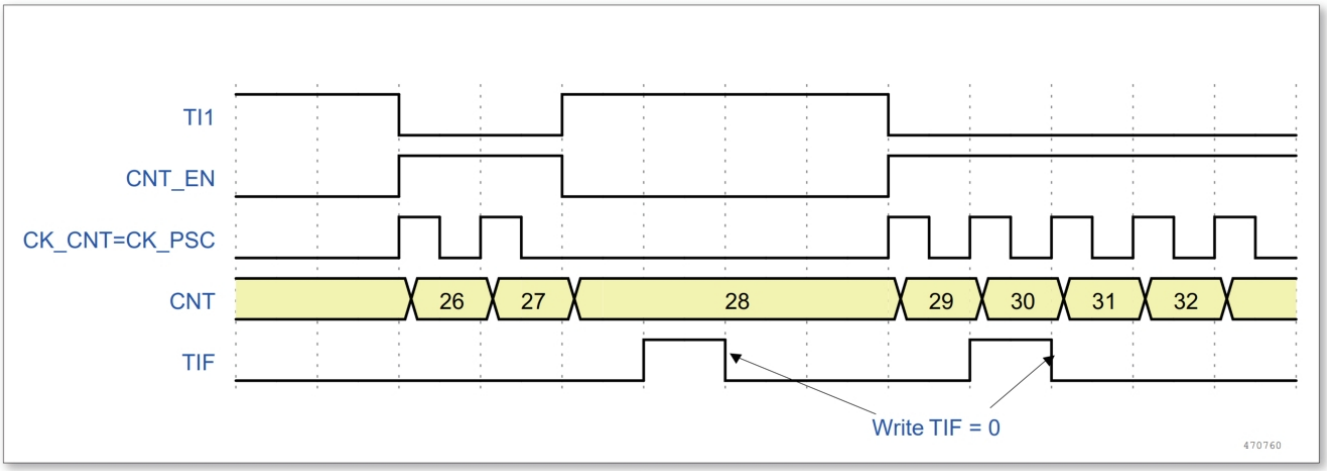


Figure 107. Control circuit in gated mode

Slave mode: Trigger mode

The counter can start in response to an event on a selected input.

In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we don't need any filter, so we keep IC2F = 0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC2S bits select the input capture source only. Write CC2S = 01 in TIMx_CCMR1 register. Write CC1P = 1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS = 110 in TIMx_SMCR register. Select TI2 as the input source by writing TS = 110 in TIMx_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set.

The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

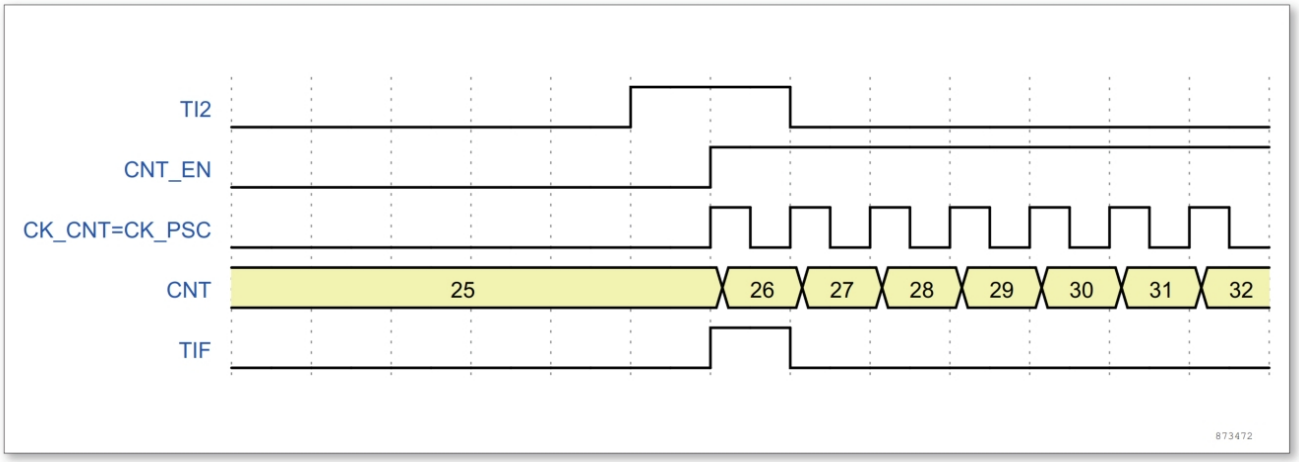


Figure 108. Control circuit in trigger mode

10.3.14 Timer synchronization

The TIMx timers are linked together internally for timer synchronization or chaining. When one Timer is configured in Master mode, it can reset, start, stop or clock the counter of another Timer configured in Slave mode.

The figure below presents an overview of the trigger selection and the master mode selection blocks.

Using one timer as prescaler for another timer

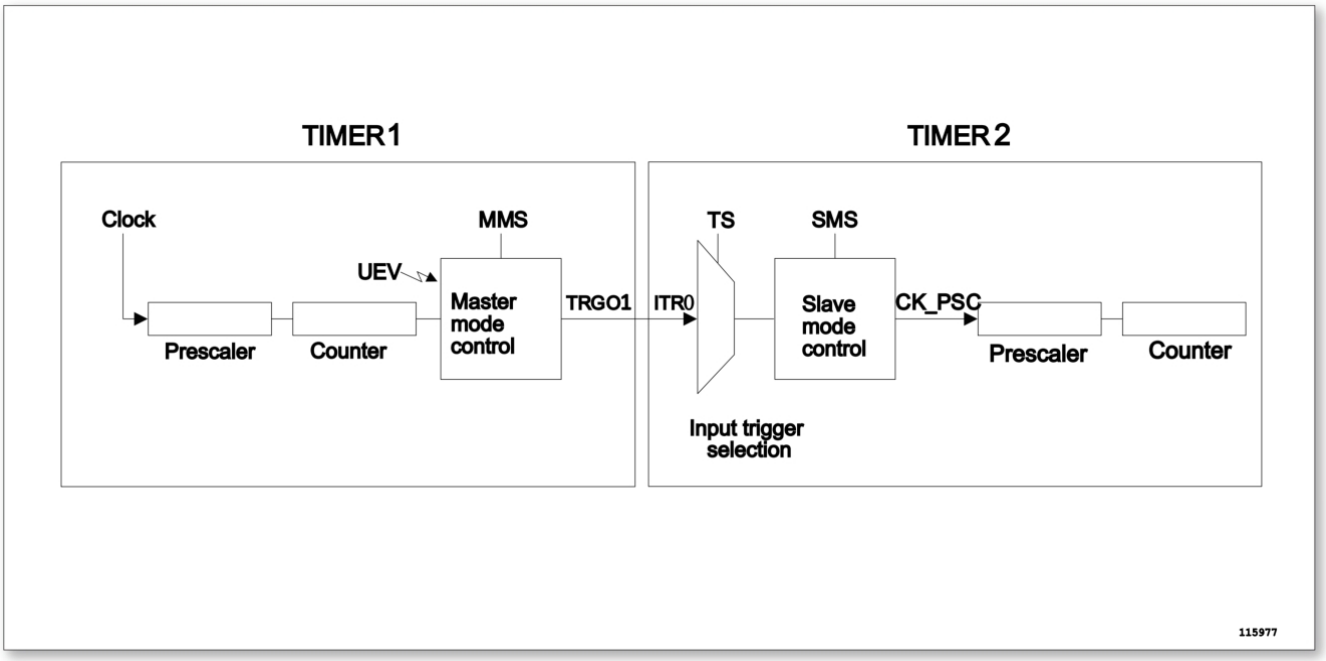


Figure 109. Master/Slave timer example

For example, the user can configure Timer 1 to act as a prescaler for Timer 3. Refer to the above diagram and perform the following operations:

- Configure Timer 1 in master mode so that it outputs a periodic trigger signal on each update event UEV.

  If you write MMS = 010 in the TIM1_CR2 register, a rising edge is output on TRGO1 each time an update event is generated.
- To connect the TRGO1 output of Timer 1 to Timer 3, Timer 3 must be configured in slave mode using ITR1 as internal trigger. You select this through the TS bits in the TIM3_SMCR register (writing TS = 000).

- Then you put the slave mode controller in external clock mode 1 (write SMS = 111 in the TIM3_SMCR register). This causes Timer 3 to be clocked by the rising edge of the periodic Timer 1 trigger signal (which corresponds to the Timer 1 counter overflow).
- Finally both timers must be enabled by setting their respective CEN bits (TIMx_CR1 register).

Note: If OCx is selected on Timer 1 as trigger output (MMS = 1xx), its rising edge is used to clock the counter of Timer 3.

Using one timer to enable another timer

In this example, we control the enable of Timer 3 with the output compare of Timer 1. Refer to the diagram below. Timer 3 counts on the divided internal clock only when OC1REF of Timer 1 is high. Both counter clock frequencies are divided by 3 by the prescaler compared to CK_INT ($f_{CK\_CNT} = f_{CK\_INT/3}$).

- Configure Timer 1 master mode to send its Output compare 1 Reference (OC1REF) signal as trigger output (MMS = 100 in the TIM1_CR2 register).
- Configure the Timer 1 OC1REF waveform (TIM1_CCMR1 register).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 001 in the TIM3_SMCR register).
- Configure Timer 3 in gated mode (SMS = 101 in TIM3_SMCR register).
- Enable Timer 3 by writing '1' in the CEN bit (TIM3_CR1 register).
- Start Timer 1 by writing '1' in the CEN bit (TIM1_CR1 register).

Note: The counter 3 clock is not synchronized with counter 1, this mode only affects the Timer 3 counter enable signal.

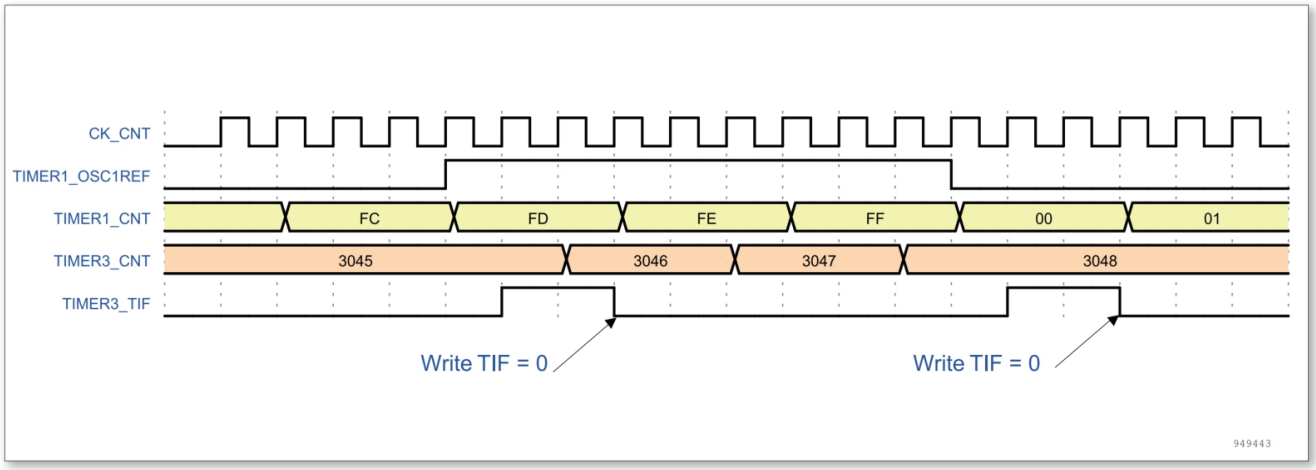


Figure 110.Gating timer 3 with OC1REF of timer 1

In the above example, the Timer 3 counter and prescaler are not initialized before being started. So they start counting from their current value. It is possible to start from a given value by resetting both timers before starting Timer 1. You can then write any value you want in the timer counters. The timers can easily be reset by writing the UG bit in the TIMx_EGR registers.

In the next example, we synchronize Timer 1 and Timer 3. Timer 1 is the master and starts from 0. Timer 3 is the slave and starts from 0xE7. The prescaler ratio is the same for both timers. Timer 3 stops when Timer 1 is disabled by writing '0' to the CEN bit in the TIM1_CR1 register.

- Configure Timer 1 master mode to send its Output compare 1 Reference (OC1REF) signal as trigger output (MMS = 100 in the TIM1_CR2 register).
- Configure the Timer 1 OC1REF waveform (TIM1_CCMR1 register).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register).
- Configure Timer 3 in gated mode (SMS = 101 in TIM3_SMCR register).
- Reset Timer 1 by writing ‘1’ in UG bit (TIM1_EGR register).
- Reset Timer 3 by writing ‘1’ in UG bit (TIM3_EGR register).
- Initialize Timer 3 to 0xE7 by writing ‘0xE7’ in the timer 3 counter (TIM3_CNT).
- Enable Timer 3 by writing ‘1’ in the CEN bit (TIM3_CR1 register).
- Start Timer 1 by writing ‘1’ in the CEN bit (TIM1_CR1 register).
- Stop Timer 1 by writing ‘0’ in the CEN bit (TIM1_CR1 register).

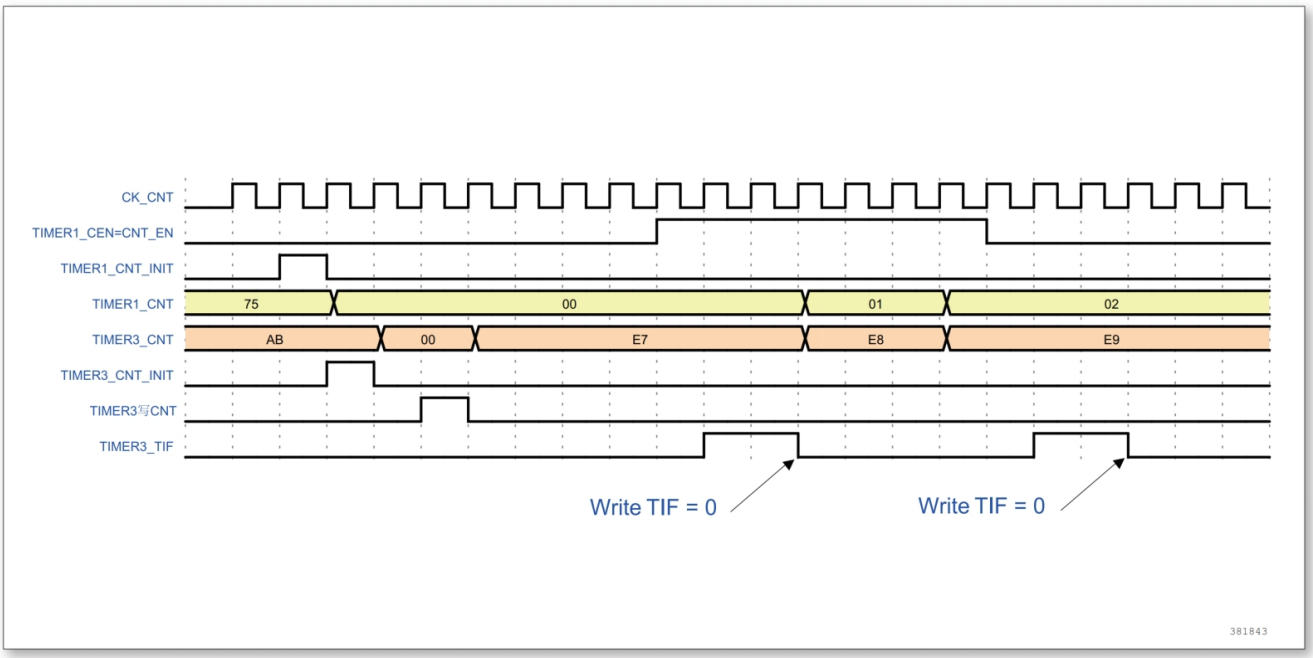


Figure 111. Gating timer 3 with enable of timer 1

Using one timer to start another timer

In this example, we set the enable of Timer 3 with the update event of Timer 1. Refer to the diagram below. Timer 3 starts counting from its current value (which can be nonzero) on the divided internal clock as soon as the update event is generated by Timer 1. When Timer 3 receives the trigger signal, its CEN bit is automatically set to ‘1’ and the counter counts until we write ‘0’ to the CEN bit in the TIM3_CR1 register. Both counter clock frequencies are divided by 3 by the prescaler compared to CK_INT ($f_{CK\_CNT} = f_{CK\_INT/3}$).

- Configure Timer 1 master mode to send its Update Event (UEV) as trigger output (MMS = 010 in the TIM1_CR2 register).
- Configure the Timer 1 period (TIM1_ARR register).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register).
- Configure Timer 3 in trigger mode (SMS = 110 in TIM3_SMCR register).
- Start Timer 1 by writing ‘1’ in the CEN bit (TIM1_CR1 register).

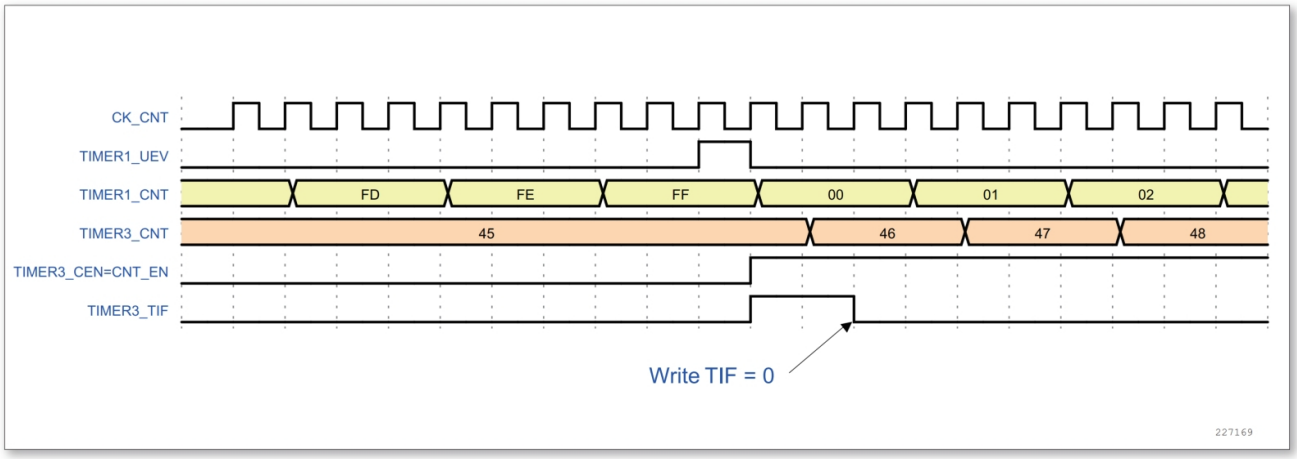


Figure 112. Triggering timer 3 with update of timer 1

As in the previous example, the user can initialize both counters before starting counting. The diagram below shows the behavior with the same configuration as 0 but in trigger mode instead of gated mode (SMS = 110 in the TIM3 SMCR register).

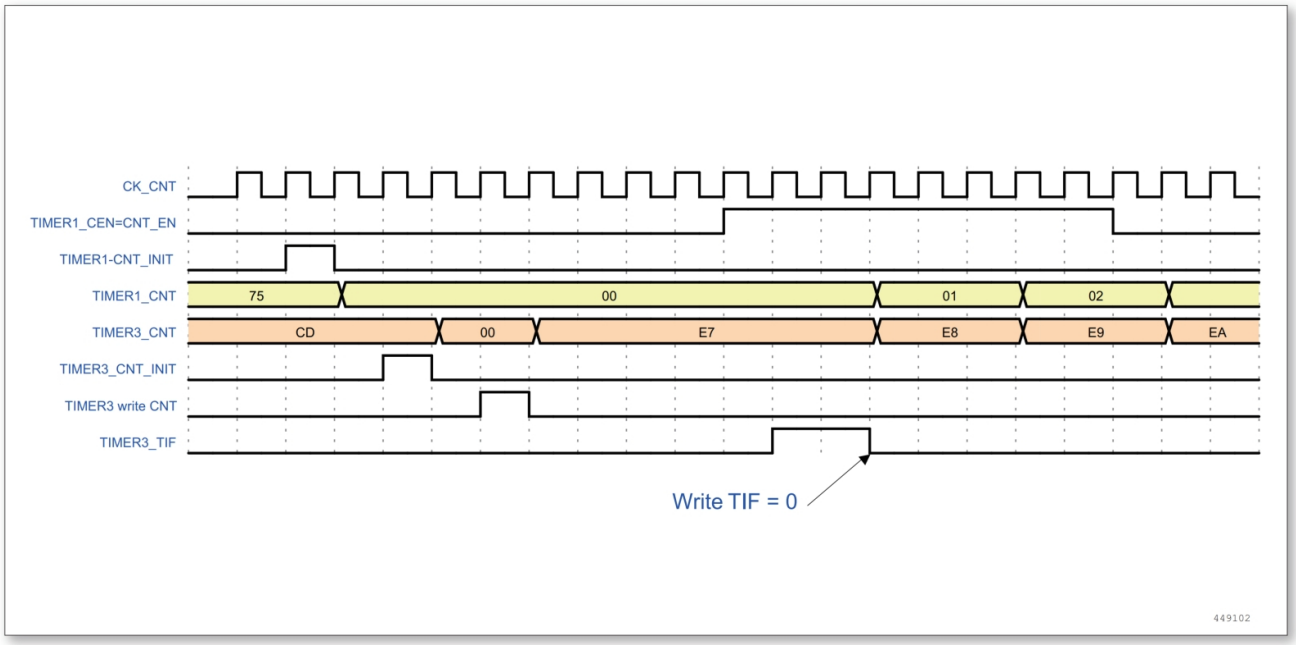


Figure 113. Triggering timer 3 with enable of timer 1

Using one timer as prescaler for another timer

In this example, we use Timer 1 as prescaler for Timer 3. We configure as follows:

- Configure Timer 1 master mode to send its Update Event (UEV) as trigger output (MMS = 010 in the TIM1_CR2 register). Then a periodic signal can be generated at each counter overflow.
- Configure the Timer 1 period (TIM1_ARR register).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register).
- Configure Timer 3 in external clock mode (SMS = 111 in TIM3_SMCR register).
- Start Timer 3 by writing ‘1’ in the CEN bit (TIM1_CR2 register).
- Start Timer 1 by writing ‘1’ in the CEN bit (TIM1_CR1 register).

Starting 2 timers synchronously in response to an external trigger

In this example, we set the enable of Timer 1 when its TI1 input rises, and the enable of Timer 3 with the enable of Timer 1. To ensure the counters are aligned, Timer 1 must be configured in Master/Slave mode (slave with respect to TI1, master with respect to Timer 3):

- Configure Timer 1 master mode to send its Enable as trigger output (MMS = 001 in the TIM1_CR2 register).
- Configure Timer 1 slave mode to get the input trigger from TI1 (TS = 100 in the TIM1_SMCR register).
- Configure Timer 1 in trigger mode (SMS = 110 in the TIM1_SMCR register).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register).
- Configure Timer 3 in trigger mode (SMS = 110 in TIM3_SMCR register).

When a rising edge occurs on TI1 (Timer 1), both counters starts counting synchronously on the internal clock and both TIF flags are set.

Note: In this example both timers are initialized before starting (by setting their respective UG bits). Both counters starts from 0, but you can easily insert an offset between them by writing any of the counter registers (TIMx_CNT). You can see from the following diagram that there is a delay between CNT_EN and CK_PSC on Timer 1 in the master/slave mode.

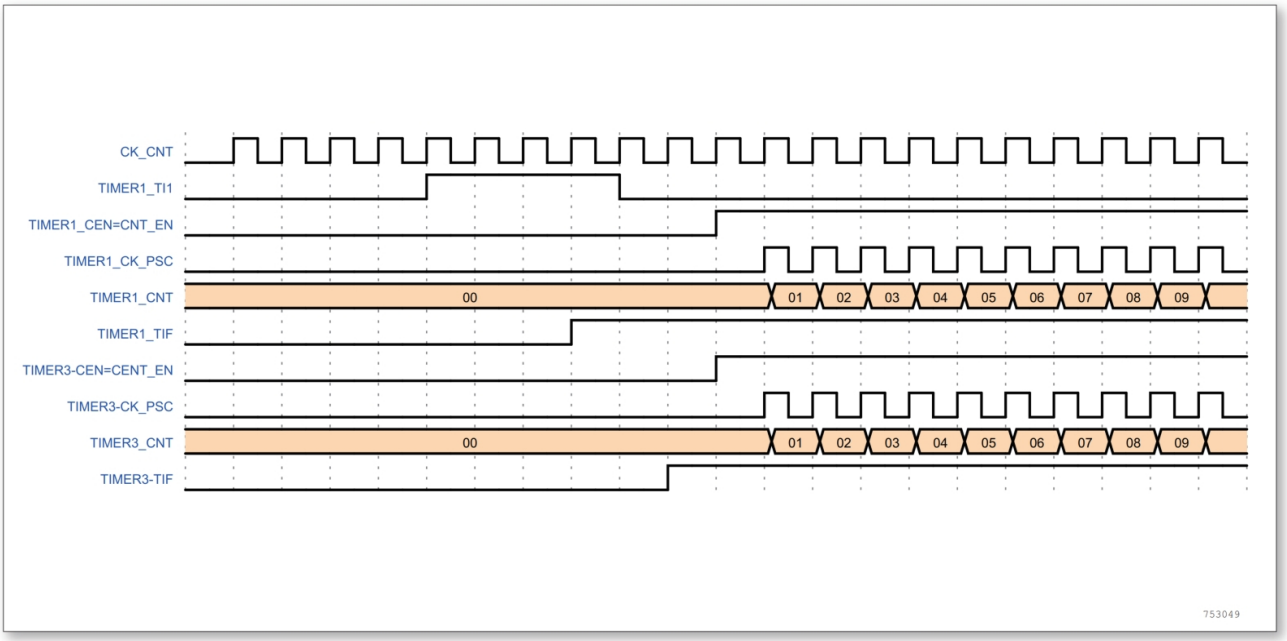


Figure 114. Triggering timer 1 and 3 with timer 1 TI1 input

10.3.15 Debug mode

When the microcontroller enters debug mode (CPU core - halted), the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module. For more details, refer to debug module section.

10.4 TIMx register description

The peripheral registers can be accessed by half-words (16-bit) or words (32-bit).

Table 37. Overview of TIMx registers

Offset	Acronym	Register Name	Reset	Section
0x00	TIMx_CR1	Control register 1	0x00000000	Section 10.4.1
0x04	TIMx_CR2	Control register 2	0x00000000	Section 10.4.2
0x08	TIMx_SMCR	Slave mode control register	0x00000000	Section 10.4.3
0x0C	TIMx_DIER	Interrupt enable register	0x00000000	Section 10.4.4
0x10	TIMx_SR	Status register	0x00000000	Section 10.4.5
0x14	TIMx_EGR	Event generate register	0x00000000	Section 10.4.6
0x18	TIMx_CCMR1	Capture/compare mode register 1	0x00000000	Section 10.4.7
0x1C	TIMx_CCMR2	Capture/compare mode register 2	0x00000000	Section 10.4.8
0x20	TIMx_CCER	Capture/compare enable register 1	0x00000000	Section 10.4.9
0x24	TIMx_CNT	Counter	0x00000000	Section 10.4.10
0x28	TIMx_PSC	Prescaler	0x00000000	Section 10.4.11
0x2C	TIMx_ARR	Auto-reload register	0x00000000	Section 10.4.12
0x34	TIMx_CCR1	Capture/compare register 1	0x00000000	Section 10.4.13
0x38	TIMx_CCR2	Capture/compare register 2	0x00000000	Section 10.4.14
0x3C	TIMx_CCR3	Capture/compare register 3	0x00000000	Section 10.4.15
0x40	TIMx_CCR4	Capture/compare register 4	0x00000000	Section 10.4.16

10.4.1 Control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKD		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:10	Reserved			Reserved, read as 0
9:8	CKD	rw	0x00	Clock division These 2 bits indicate the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock used by the dead-time generators and the digital filters (TIx). 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: Reserved, do not program this value

Bit	Field	Type	Reset	Description
7	ARPE	rw	0x00	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6:5	CMS	rw	0x00	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set both when the counter is counting up. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN = 1)
4	DIR	rw	0x00	Direction 0: Counter used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
3	OPM	rw	0x00	One pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	rw	0x00	Update request source This bit is set by software to select the UEV event sources. 0: Any of the following events generate an update interrupt if enabled. These events can be: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt if enabled.

Bit	Field	Type	Reset	Description
1	UDIS	rw	0x00	Update disabled This bit is set by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: - Counter overflow/underflow - Setting the UG bit to '1' - Update generation through the slave mode controller with the subsequent update of shadow registers 1: UEV disabled. The Update event is not generated, shadow registers keep their values (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set to '1' or if a hardware reset is received from the slave mode controller.
0	CEN	rw	0x00	Counter enable 0: Counter disabled 1: Counter enabled Note: External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware.

10.4.2 Control register 2 (TIMx_CR2)

Address offset: 0x04

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TI1S	MMS			Reserved			
								rw	rw						

Bit	Field	Type	Reset	Description
15: 8	Reserved			Reserved, read as 0
7	TI1S	rw	0x00	TI1 selection 0: The TIMx_CH1 pin is connected to TI1 input 1: The TIMx_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)
6:4	MMS	rw	0x00	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the TIMx_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset.

Bit	Field	Type	Reset	Description
				001: Enable - the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIMx_SMCR register). 010: Update - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output (TRGO) send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. 100: Compare - OC1REF signal is used as trigger output (TRGO) 101: Compare - OC2REF signal is used as trigger output (TRGO) 110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
3: 0	Reserved			Reserved and read as 0.

10.4.3 Slave mode control register (TIMx_SMCR)

Address offset: 0x08

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								MSM	TS			Res.	SMS		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw

Bit	Field	Type	Reset	Description
15: 8	Reserved			Reserved, read as 0
7	MSM	rw	0x00	Master/slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.

Bit	Field	Type	Reset	Description
6:4	TS	rw	0x00	Trigger selection This bit-field selects the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0) 001: Internal Trigger 1 (ITR1) 010: Internal Trigger 2 (ITR2) 011: Internal Trigger 3 (ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) See the following table for more details on ITRx. Note: These bits must be changed only when they are not used (e.g. when SMS=000) to avoid wrong edge detections at the transition.
3	Reserved			Reserved and read as 0.
2:0	SMS	rw	0x00	Slave mode selection When external signals are selected, the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control register description). 000: Slave mode disabled - if CEN = 1, then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 - Counter counts up/down on TI2FP2 edge depending on TI1FP1 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. 011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 110: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 - Rising edges of the selected trigger (TRGI) clock the counter. Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS = 100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.

Table 38. TIMx internal trigger connection

Slave TIM	ITR0 (TS = 000)	ITR1 (TS = 001)	ITR2 (TS = 010)	ITR3 (TS = 011)
TIM3	TIM1	No	No	No

10.4.4 Interrupt enable register (TIMx_DIER)

Address offset: 0x0C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
									rw		rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15: 7	Reserved			Reserved, read as 0
6	TIE	rw	0x00	Trigger interrupt enable 0: Trigger interrupt disabled 1: Trigger interrupt enabled
5	Reserved			Reserved and read as 0.
4	CC4IE	rw	0x00	Capture/Compare 4 interrupt enable 0: Capture/Compare 4 interrupt disabled 1: Capture/Compare 4 interrupt enabled
3	CC3IE	rw	0x00	Capture/Compare 3 interrupt enable 0: Capture/Compare 3 interrupt disabled 1: Capture/Compare 3 interrupt enabled
2	CC2IE	rw	0x00	Capture/Compare 2 interrupt enable 0: Capture/Compare 2 interrupt disabled 1: Capture/Compare 2 interrupt enabled
1	CC1IE	rw	0x00	Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
0	UIE	rw	0x00	Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

10.4.5 Status register (TIMx_SR)

Address offset: 0x10

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.				CC3OF	CC2OF	CC1OF	Res.		TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
				rc_w0	rc_w0	rc_w0			rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bit	Field	Type	Reset	Description
15:12	Reserved			Reserved, read as 0
11	CC3OF	rc_w0	0x00	Capture/Compare 3 overcapture flag Refer to CC1OF description.
10	CC2OF	rc_w0	0x00	Capture/Compare 2 overcapture flag refer to CC1OF description.
9	CC1OF	rc_w0	0x00	Capture/Compare 1 overcapture flag This flag is set to '1' by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set to 1.
8: 7	Reserved			Reserved and read as 0.
6	TIF	rc_w0	0x00	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode, both edges in case gated mode is selected). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
5	Reserved			Reserved and read as 0.
4	CC4IF	rc_w0	0x00	Capture/Compare 4 interrupt flag Refer to CC1IF description.
3	CC3IF	rc_w0	0x00	Capture/Compare 3 interrupt flag Refer to CC1IF description.
2	CC2IF	rc_w0	0x00	Capture/Compare 2 interrupt flag Refer to CC1IF description.

Bit	Field	Type	Reset	Description
1	CC1IF	rc_w0	0x00	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIMx_CR1 register description). It is cleared by software. 0: No match. 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in (copied to) TIMx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	rc_w0	0x00	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update event pending. This bit is set by hardware when the registers are updated: - At overflow or underflow regarding the repetition counter value (update if REP_CNT = 0) and if the UDIS = 0 in the TIMx_CR1 register. - When CNT is reinitialized by software setting the UG bit in TIMx_EGR register, if URS=0 and UDIS=0 in the TIMx_CR1 register. - When CNT is reinitialized by a trigger event (refer to the synchronous control register description), if URS=0 and UDIS=0 in the TIMx_CR1 register.

10.4.6 Event generation register (TIMx_EGR)

Address offset: 0x14
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.									TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
									w	w	w	w	w	w	w

Bit	Field	Type	Reset	Description
15:7	Reserved			Reserved, read as 0
6	TG	w	0x00	Trigger generation This bit is set by software in order to generate a brake event, it is automatically cleared by hardware. 0: No action

Bit	Field	Type	Reset	Description
				1: The TIF flag is set in TIMx_SR register. Related interrupt can occur if enabled.
5:4	Reserved			Reserved and read as 0.
4	CC4G	w	0x00	Capture/Compare 4 generation Refer to CC1G description.
3	CC3G	w	0x00	Capture/Compare 3 generation Refer to CC1G description.
2	CC2G	w	0x00	Capture/Compare 2 generation Refer to CC1G description.
1	CC1G	w	0x00	Capture/Compare 1 generation This bit is set by software in order to generate a capture/compare event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as output: CC1IF flag is set, corresponding interrupt is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	w	0x00	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR = 0 (upcounting), else it takes the auto-reload value (TIMx_ARR) if DIR = 1 (downcounting).

10.4.7 Capture/compare mode register 1 (TIMx_CCMR1)

Address offset: 0x18

Reset value: 0x0000

The channels can be used in input (capture mode) or in output (compare mode). The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function in output mode. For a given bit, OCxx describes its function when the channel is configured in output, ICxx describes its function when the channel is configured in input. So the user must take care that the same bit can have a different meaning for the input stage and for the output stage.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OC2M			OC2PE	OC2FE	CC2S		Res.	OC1M			OC1PE	OC1FE	CC1S	
IC2F				IC2PSC				IC1F				IC1PSC			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Output compare mode:

Bit	Field	Type	Reset	Description
15	Reserved			Reserved, read as 0
14:12	OC2M	rw	0x00	Output compare 2 mode
11	OC2PE	rw	0x00	Output compare 2 preload enable
10	OC2FE	rw	0x00	Output compare 4 fast enable
9:8	CC2S	rw	0x00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output. 01: CC2 channel is configured as input, IC2 is mapped on TI2. 10: CC2 channel is configured as input, IC2 is mapped on TI1. 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = 0 in TIMx_CCER).
7	Reserved			Reserved and read as 0.

Bit	Field	Type	Reset	Description
6:4	OC1M	rw	0x00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the OC1REF. 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle. OC1REF level toggles when TIMx_CCR1=TIMx_CNT. 100: Force inactive level. OC1REF is forced low. 101: Force active level. OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as TIMx_CNT < TIMx_CCR1, else inactive. In downcounting, channel 1 is inactive (OC1REF = 0) as long as TIMx_CNT > TIMx_CCR1, else active (OC1REF = 1). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1, else active. In downcounting, channel 1 is active as long as TIMx_CNT > TIMx_CCR1, else inactive. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = 00 (the channel is configured in output). Note 2: In PWM mode 1 or 2, the OC1REF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.

Bit	Field	Type	Reset	Description
3	OC1PE	rw	0x00	Output compare 1 preload enable 0: Preload function of TIMx_CCR1 register disabled. TIMx_CCR1 register can be written at anytime, the new value is taken in account immediately. 1: Preload function of TIMx_CCR1 register enabled. Read/Write operations access the preload register. TIMx_CCR1 preload value is loaded in the active register at each update event. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = 00 (the channel is configured in output). Note 2: The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIMx_CR1 register). Else the behavior is not guaranteed.
2	OC1FE	rw	0x00	Output compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S	rw	0x00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIMx_CCER).

Input capture mode:

Bit	Field	Type	Reset	Description
15:12	IC2F	rw	0x00	Input capture 2 filter
11:10	IC2PSC	rw	0x00	Input capture 2 prescaler

Bit	Field	Type	Reset	Description
9:8	CC2S	rw	0x00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC2 channel is configured as output. 01: CC2 channel is configured as input, IC2 is mapped on TI2. 10: CC2 channel is configured as input, IC2 is mapped on TI1. 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = 0 in TIMx_CCER).
7:4	IC1F	rw	0x00	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to generate a transition on the output: 0000: No filter, sampling is done at f_{DTS} 1000: $f_{SAMPLING}=f_{DTS/8}$, N = 6 0001: $f_{SAMPLING}=f_{CK\_INT}$, N = 2 1001: $f_{SAMPLING}=f_{DTS/8}$, N = 8 0010: $f_{SAMPLING}=f_{CK\_INT}$, N = 4 1010: $f_{SAMPLING}=f_{DTS/16}$, N = 5 0011: $f_{SAMPLING}=f_{CK\_INT}$, N = 8 1011: $f_{SAMPLING}=f_{DTS/16}$, N = 6 0100: $f_{SAMPLING}=f_{DTS/2}$, N = 6 1100: $f_{SAMPLING}=f_{DTS/16}$, N = 8 0101: $f_{SAMPLING}=f_{DTS/2}$, N = 8 1101: $f_{SAMPLING}=f_{DTS/32}$, N = 5 0110: $f_{SAMPLING}=f_{DTS/4}$， N = 6 1110: $f_{SAMPLING}=f_{DTS/32}$, N = 6 0111: $f_{SAMPLING}=f_{DTS/4}$， N = 8 1111: $f_{SAMPLING}=f_{DTS/32}$, N = 8
3:2	IC1PSC	rw	0x00	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E = 0 (TIMx_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events

Bit	Field	Type	Reset	Description
1:0	CC1S	rw	0x00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIMx_CCER).

10.4.8 Capture/compare mode register 2 (TIMx_CCMR2)

Address offset: 0x1C

Reset value: 0x0000

Refer to the above CCMR1 register description

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OC4M			OC4PE	OC4FE	CC4S		OC3CE	OC3M			OC3PE	OC3FE	CC3S	
								IC3F				IC3PSC			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Output compare mode

Bit	Field	Type	Reset	Description
15	Reserved			Reserved and read as 0.
14:12	OC4M	rw	0x00	Output compare 4 mode
11	OC4PE	rw	0x00	Output compare 4 preload enable
10	OC4FE	rw	0x00	Output compare 4 fast enable
9:8	CC4S	rw	0x00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = 0 in TIMx_CCER).

Bit	Field	Type	Reset	Description
7	OC3CE	rw	0x00	Output compare 3 clear enable
6:4	OC3M	rw	0x00	Output compare 3 mode
3	OC3PE	rw	0x00	Output compare 3 preload enable
2	OC3FE	rw	0x00	Output compare 3 fast enable
1:0	CC3S	rw	0x00	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = 0 in TIMx_CCER).

Input capture mode

Bit	Field	Type	Reset	Description
15:8	Reserved			Reserved, read as 0
7:4	IC3F	rw	0x00	Input capture 3 filter
3:2	IC3PSC	rw	0x00	Input capture 3 prescaler
1:0	CC3S	rw	0x00	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = 0 in TIMx_CCER).

10.4.9 Capture/compare enable register (TIMx_CCER)

Address offset: 0x20

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	Res.	CC4P	CC4E	CC3NP	Res.	CC3P	CC3E	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
rw		rw	rw	rw	rw	rw	rw	rw		rw	rw	rw		rw	rw

Bit	Field	Type	Reset	Description
15	CC4NP	rw	0x00	Capture/Compare 4 comple mentary output polarity Refer to CC1NP description.
14	Reserved			Reserved and always read as 0.
13	CC4P	rw	0x00	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	rw	0x00	Capture/Compare 4 output enable Refer to CC1E description.
11	CC3NP	rw	0x00	Capture/Compare 3 complementary output polarity Refer to CC1NP description.
10	Reserved			Reserved and always read as 0.
9	CC3P	rw	0x00	Capture/Compare 3 output polarity Refer to CC1P description.
8	CC3E	rw	0x00	Capture/Compare 3 output enable Refer to CC1E description.
7	CC2NP	rw	0x00	Capture/Compare 2 complementary output polarity Refer to CC1NP description.
6	Reserved			Reserved and always read as 0.
5	CC2P	rw	0x00	Capture/Compare 2 output polarity Refer to CC1P description.
4	CC2E	rw	0x00	Capture/Compare 2 output enable Refer to CC1E description.
3	CC1NP	rw	0x00	Capture/Compare 1 complementary output polarity 0: OC1N active high 1: OC1N active low Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = 00 (the channel is configured in output).
2	Reserved			Reserved and always read as 0.
1	CC1P	rw	0x00	Capture/Compare 1 output polarity CC1 channel configured as output: 0: OC1 active high 1: OC1 active low CC1 channel configured as input: This bit selects whether IC1 or the inverted signal of IC1 is used for trigger or capture operations. 0: non-inverted: capture is done on a rising edge of IC1. When used as external trigger, IC1 is non-inverted. 1: inverted: capture is done on a falling edge of IC1. When used as external trigger, IC1 is inverted. Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).

Bit	Field	Type	Reset	Description
0	CC1E	rw	0x00	Capture/Compare 1 output enable CC1 channel configured as output: 0: Off - OC1 is not active. OC1 level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. 1: On - OC1 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the TIMx_CCR1 register or not. 0: capture disabled 1: capture enabled

Table 39. Output control bit for standard Ocx channels

CCxE bit	OCx output state
0	Output disabled (OCx = 0, OCx_EN = 0)
1	OCx = OCxREF + Polarity, OCx_EN = 1

Note: The states of the external I/O pins connected to the standard OCx channels depend on the state of the OCx channel and the GPIO and AFIO registers.

10.4.10 Counter (TIMx_CNT)

Address offset: 0x24

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15：0	CNT	rw	0x0000	Counter value

10.4.11 Prescaler (TIMx_PSC)

Address offset: 0x28

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:0	PSC	rw	0x0000	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC}/(PSC + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through slave controller when configured in “reset mode”).

10.4.12 Auto-reload register (TIMx_ARR)

Address offset: 0x2C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:0	ARR	rw	0x0000	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to the Section 10.3.1 for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

10.4.13 Capture/compare register 1 (TIMx_CCR1)

Address offset: 0x34

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:0	CCR1	rw	0x0000	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded immediately if the preload feature is not selected in the TIMx_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. If channel CC1 is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

10.4.14 Capture/compare register 2 (TIMx_CCR2)

Address offset: 0x38

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:0	CCR2	rw	0x0000	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value). It is loaded immediately if the preload feature is not selected in the TIMx_CCMR2 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC2 output. If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2).

10.4.15 Capture/compare register 3 (TIMx_CCR3)

Address offset: 0x3C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:0	CCR3	rw	0x0000	Capture/Compare 3 value If channel CC3 is configured as output: CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value). It is loaded immediately if the preload feature is not selected in the TIMx_CCMR3 register (bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC3 output. If channel CC3 is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3).

10.4.16 Capture/compare register 4 (TIMx_CCR4)

Address offset: 0x40

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:0	CCR4	rw	0x0000	Capture/Compare 4 value If channel CC4 is configured as output: CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value). It is loaded immediately if the preload feature is not selected in the TIMx_CCMR4 register (bit OC4PE). Else the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC4 output. If channel CC4 is configured as input: CCR4 is the counter value transferred by the last input capture 4 event (IC4).

11 Basic Timer (TIM14)

Basic Timer (TIM14)

11.1 TIM14 introduction

The basic timer TIM14 consists of a 16-bit auto-reload counter driven by a programmable prescaler.

It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare and PWM).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers.

The basic timer TIM14 is completely independent, and does not share any resources.

11.2 TIM14 main features

- 16-bit auto-reload counter
- 16-bit programmable prescaler allowing dividing (also “on the fly”) the counter clock frequency either by any factor between 1 and 65536.
- Independent channels for:
  - Input capture
  - Output comparison
  - PWM generation (Edge-aligned Mode)
- Interrupt generation on the following events:
  - Update: counter overflow, counter initialization (by software)
  - Input capture
  - Output comparison

(TIM14)

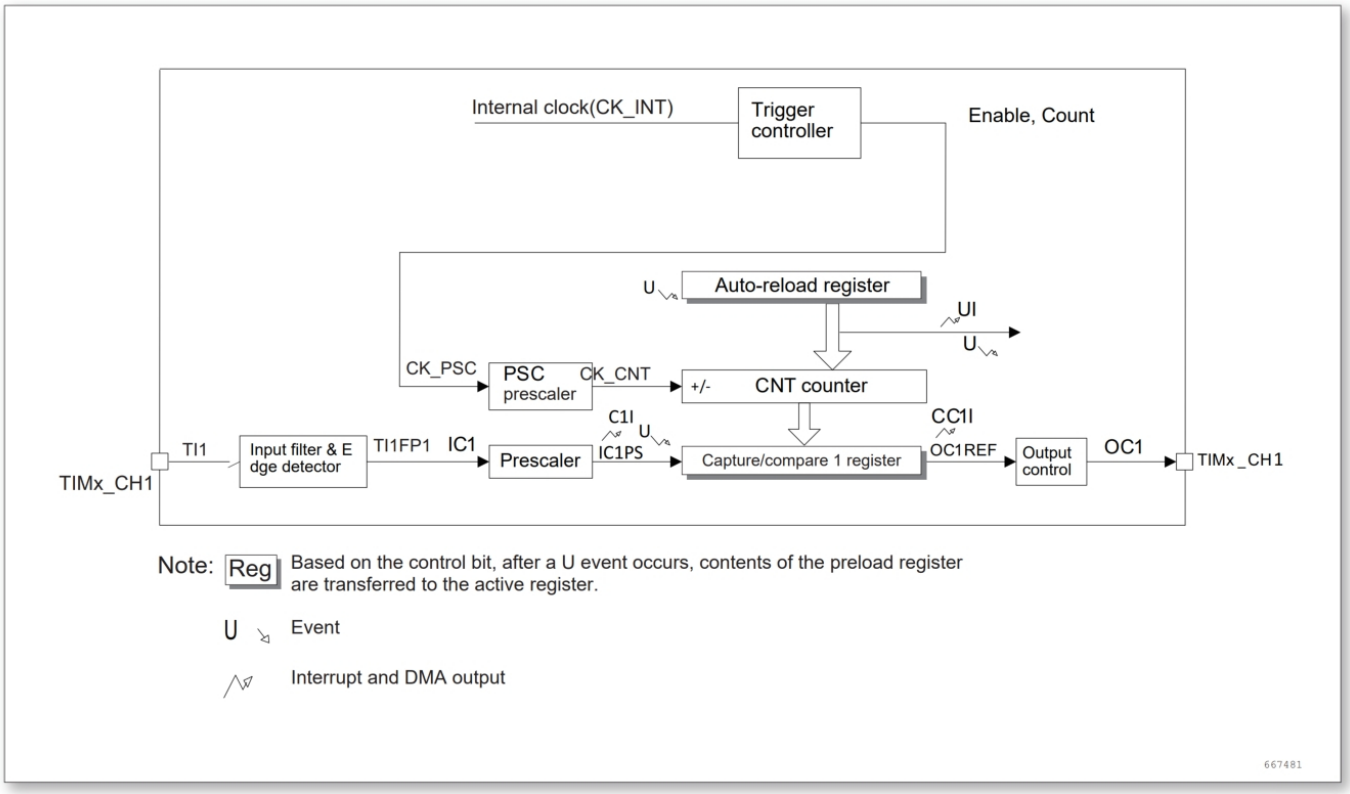


Figure 115. Basic timer block diagram

11.3 TIM14 functional description

11.3.1 Time-base unit

The main block of the programmable basic timer is a 16-bit counter with its related auto-reload register. The counter can count up. The counter clock can be divided by a prescaler. The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running. The time-base unit includes:

- Counter register (TIM14_CNT)
- Prescaler register (TIM14_PSC)
- Auto-reload register (TIM14_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIM14_CR1 register. The update event is sent when the counter reaches the overflow value and if the UDIS bit equals 0 in the TIM14_CR1 register. It can also be generated by software. The generation of the update event is described in detailed for each configuration.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIM14_CR1 register is set (refer also to the slave mode controller description to get more details on counter enabling).

Note that the actual counter enable signal CNT_EN is set 1 clock cycle after CEN.

(TIM14)

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIM14_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

The figures below give some examples of the counter behavior when the prescaler ratio is changed on the fly.

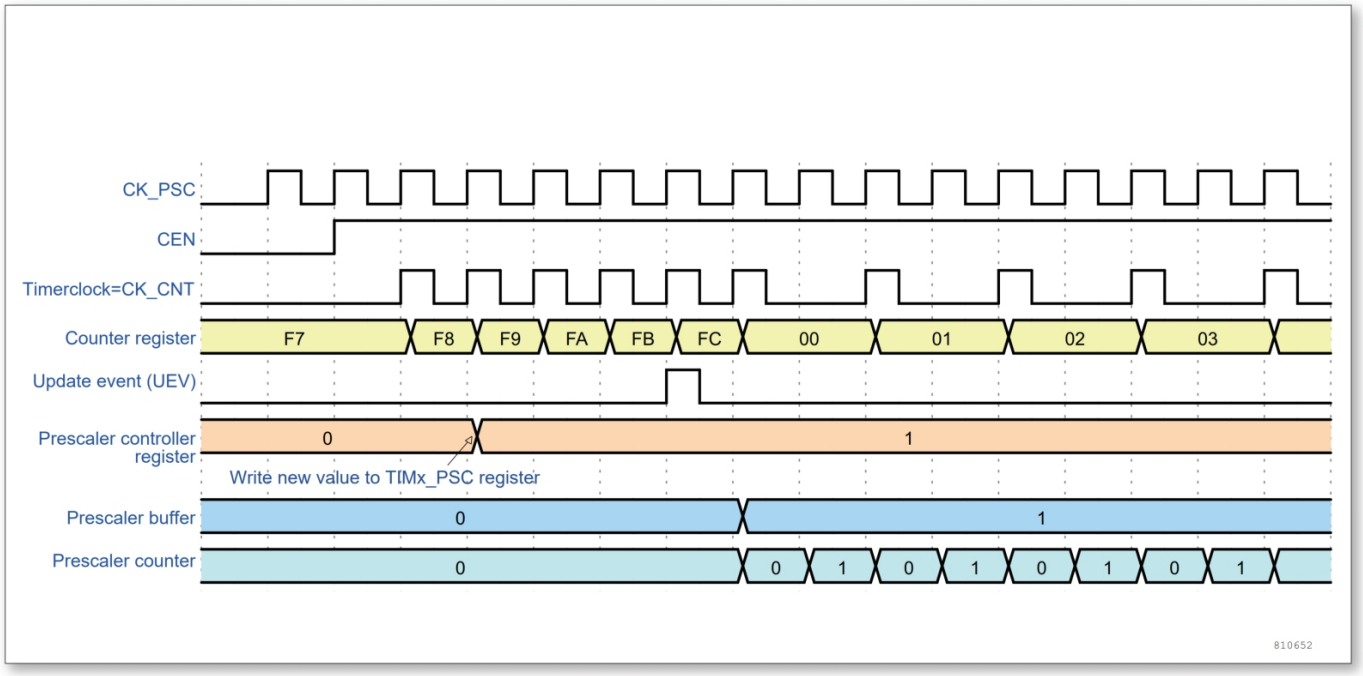


Figure 116. Counter timing diagram with prescaler division change from 1 to 2

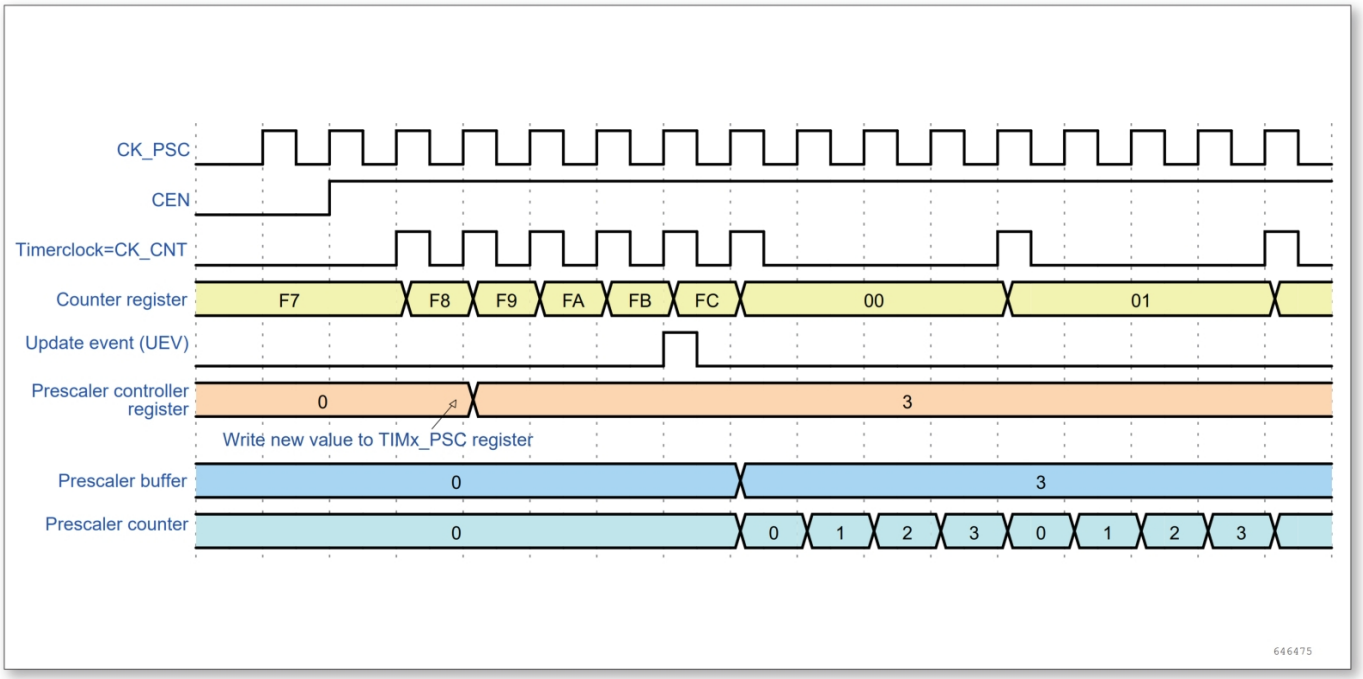


Figure 117. Counter timing diagram with prescaler division change from 1 to 4

(TIM14)

11.3.2 Counting mode

Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the TIM14_ARR counter), then restarts from 0 and generates a counter overflow event.

An update event can be generate by setting the UG bit in the TIM14_EGR register.

The UEV event can be disabled by setting the UDIS bit in the TIM14_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change) when an update event should be generated. In addition, if the URS bit (update request selection) in TIM14_CR1 register is set, setting the UG bit generates an update event UEV, but the UIF flag is not set by hardware (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter in the capture mode.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM14_SR register) is set by hardware (depending on the URS bit).

- The auto-reload shadow register is updated with the preload value (TIM14_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIM14_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIM14_ARR = 0x36:

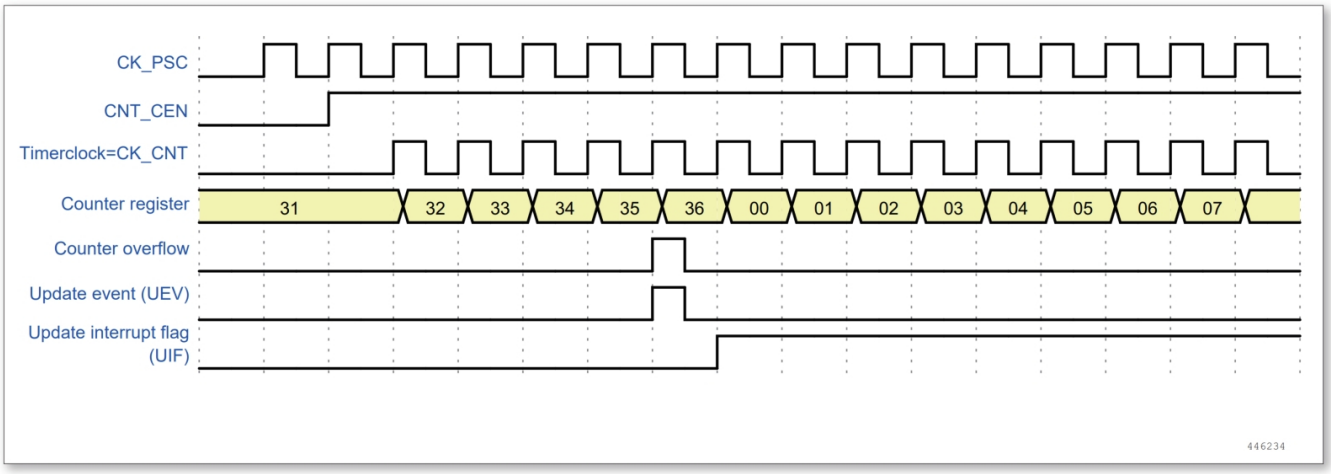


Figure 118. Counter timing diagram, internal clock divided by 1

(TIM14)

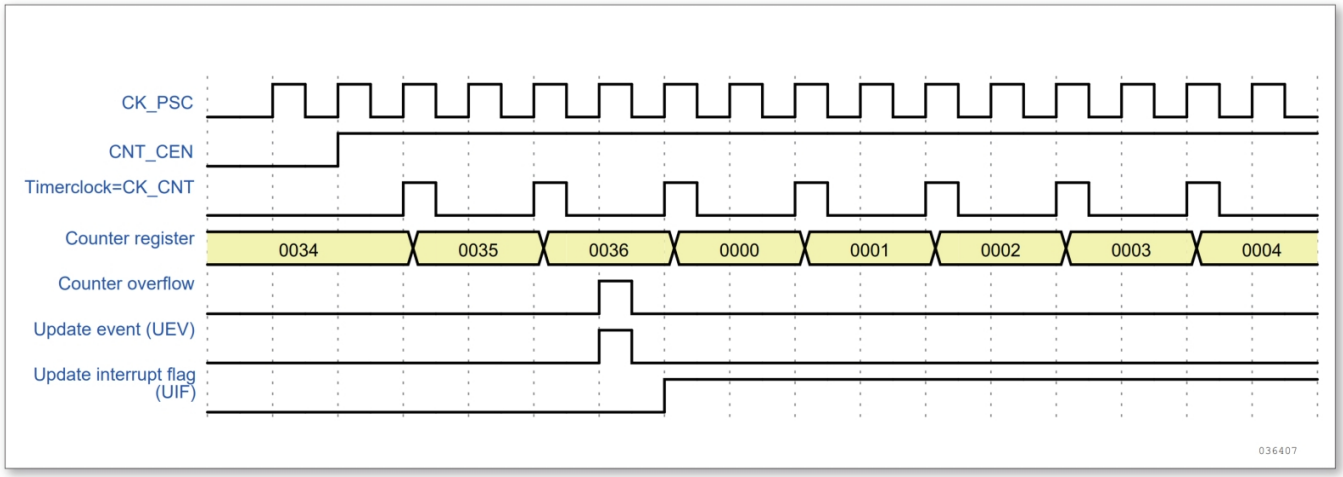


Figure 119. Counter timing diagram, internal clock divided by 2

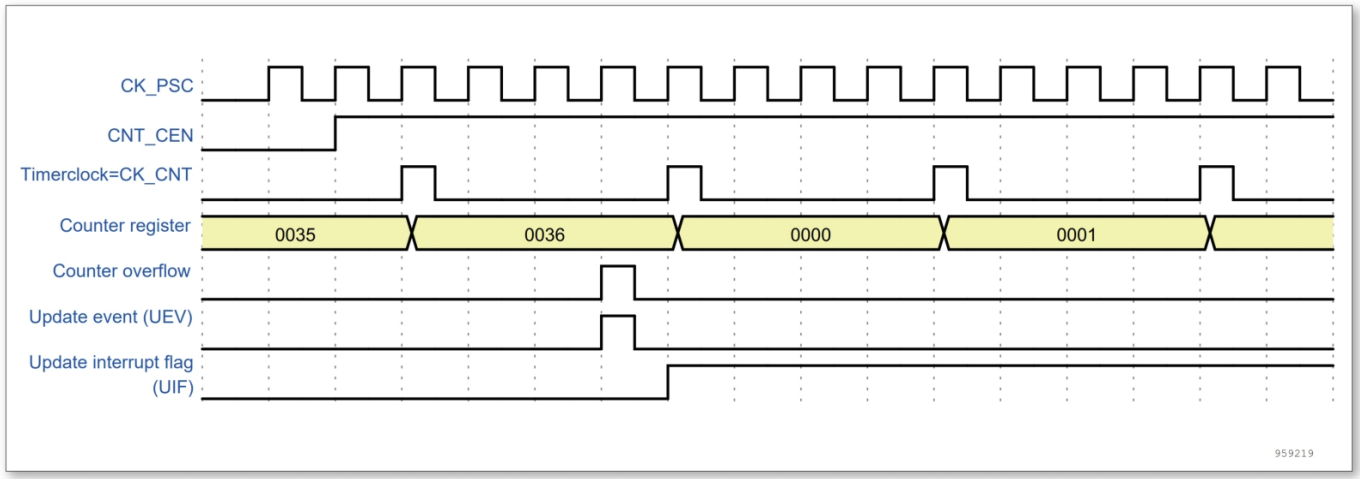


Figure 120. Counter timing diagram, internal clock divided by 4

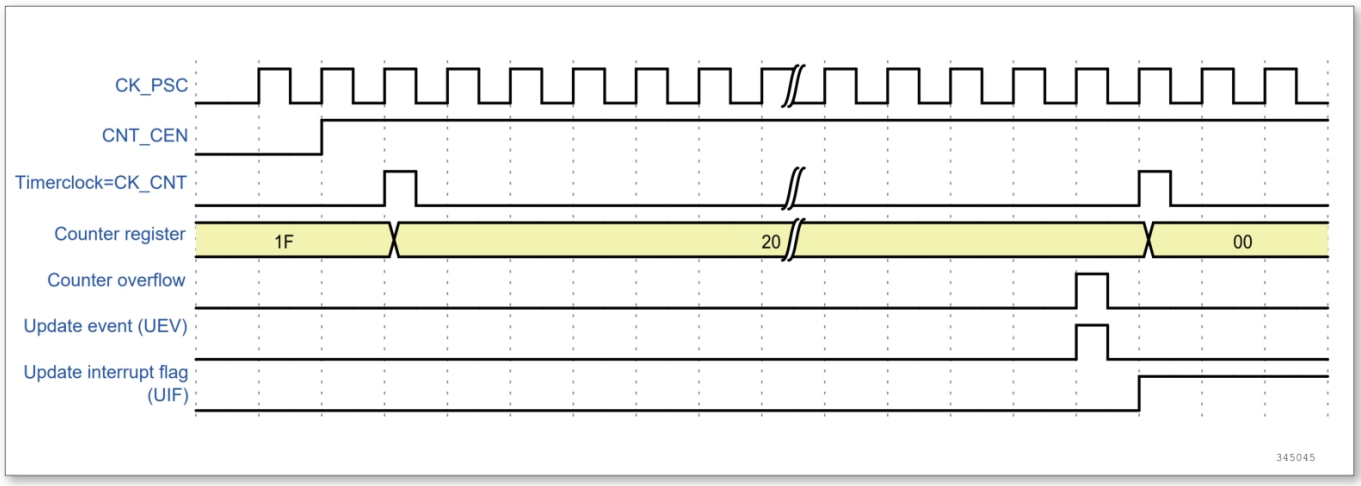


Figure 121. Counter timing diagram, internal clock divided by N

(TIM14)

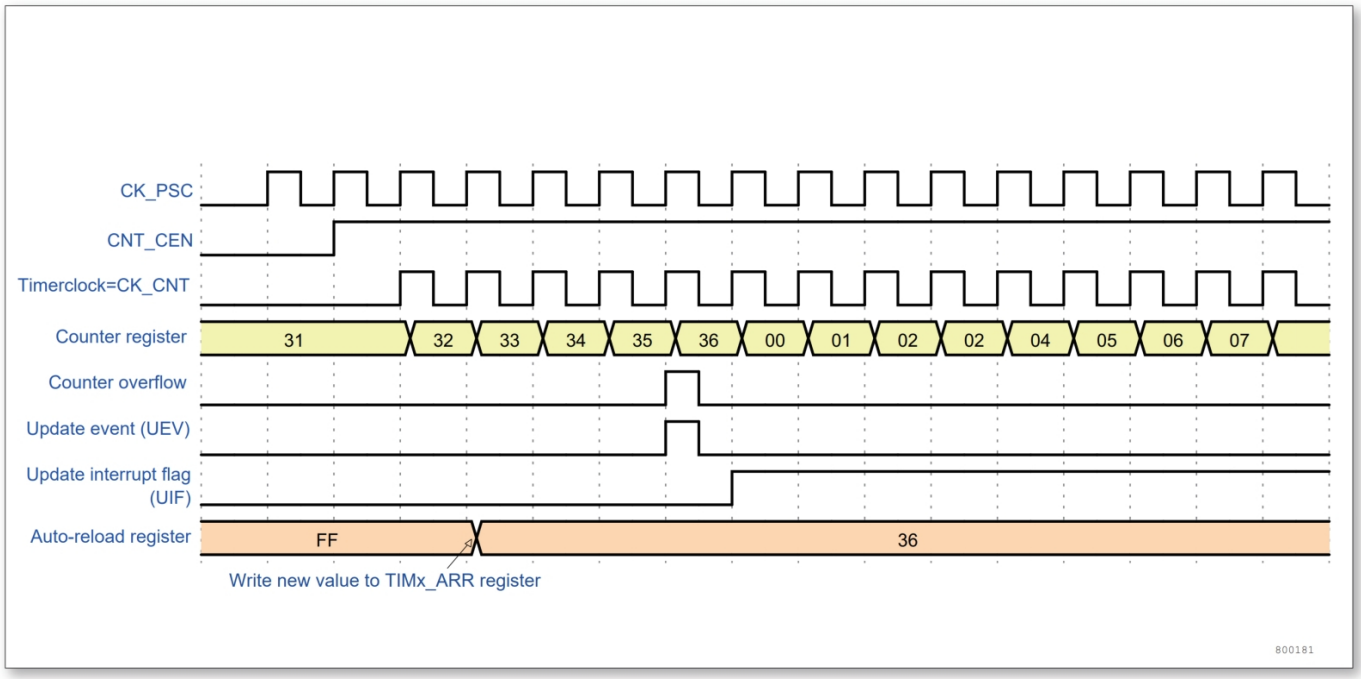


Figure 122. Counter timing diagram, update event when ARPE=0 (TIM14_ARR not preloaded)

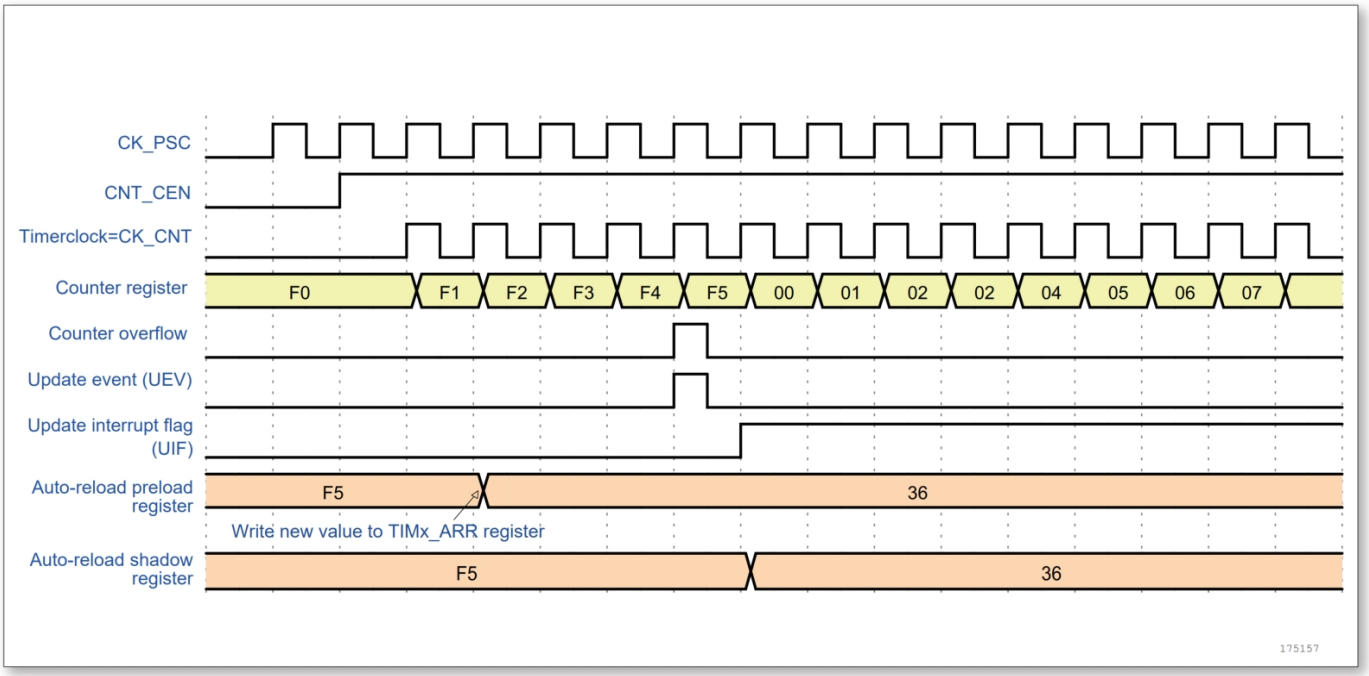


Figure 123. Counter timing diagram, update event when ARPE=1 (TIM14_ARR preloaded)

11.3.3 Clock source

The counter clock is provided by the Internal clock (CK_INT) source.

The CEN (in the TIM14_CR1 register) and UG bits (in the TIM14_EGR register) are actual control bits and can be changed only by software (except for UG that remains cleared automatically). As soon as the CEN bit is written to ‘1’, the prescaler is clocked by the internal clock CK_INT.

The figure below shows the behavior of the control circuit and the upcounter in normal mode, without prescaler.

(TIM14)

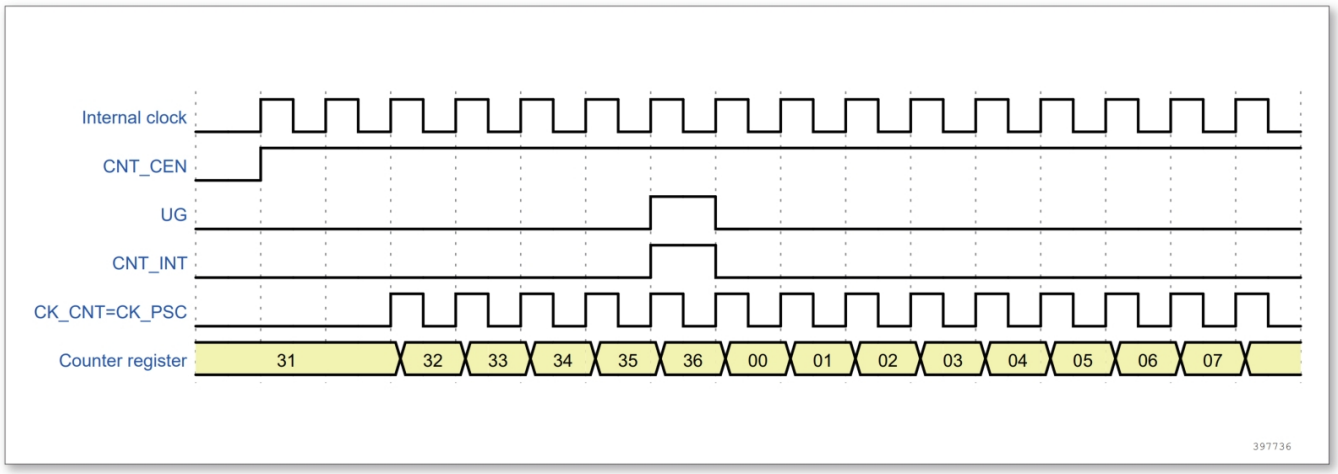


Figure 124. Control circuit in normal mode, internal clock divided by 1

11.3.4 Capture/compare channel

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), a input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The following figures give an overview of one Capture/Compare channel. The input stage samples the corresponding TIx input to generate a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

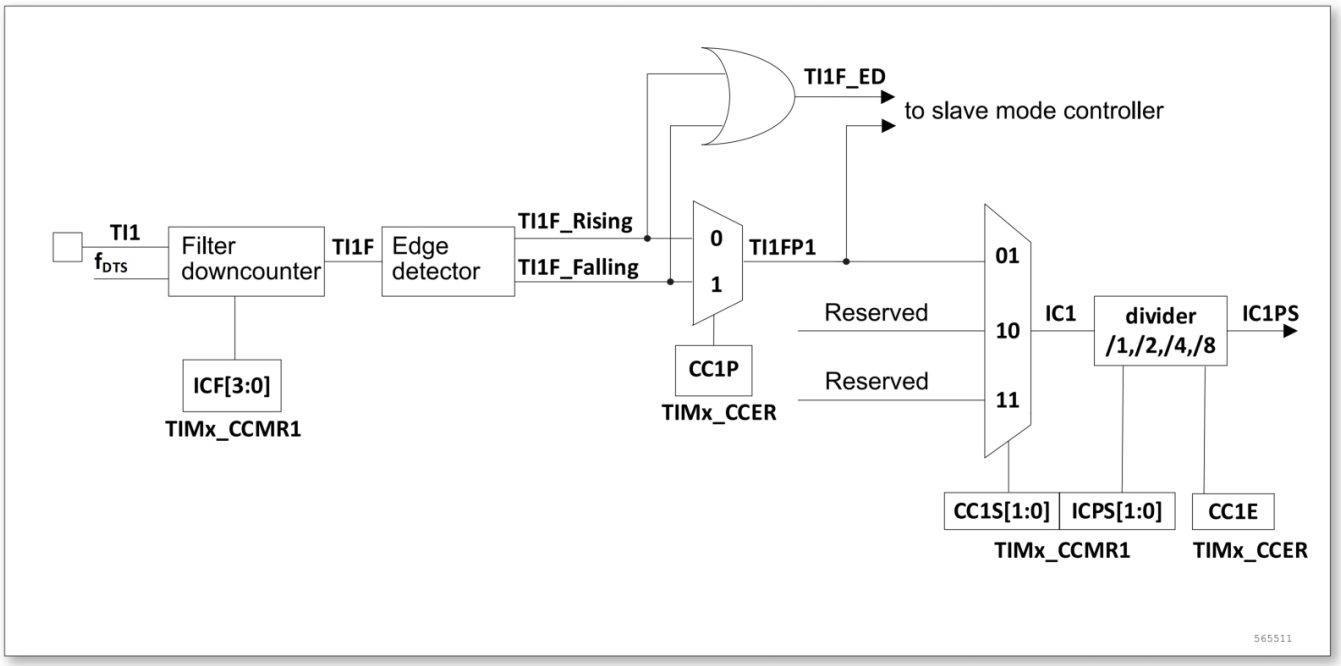


Figure 125. Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference: OCxREF (active high). The polarity acts at the end of the chain.

(TIM14)

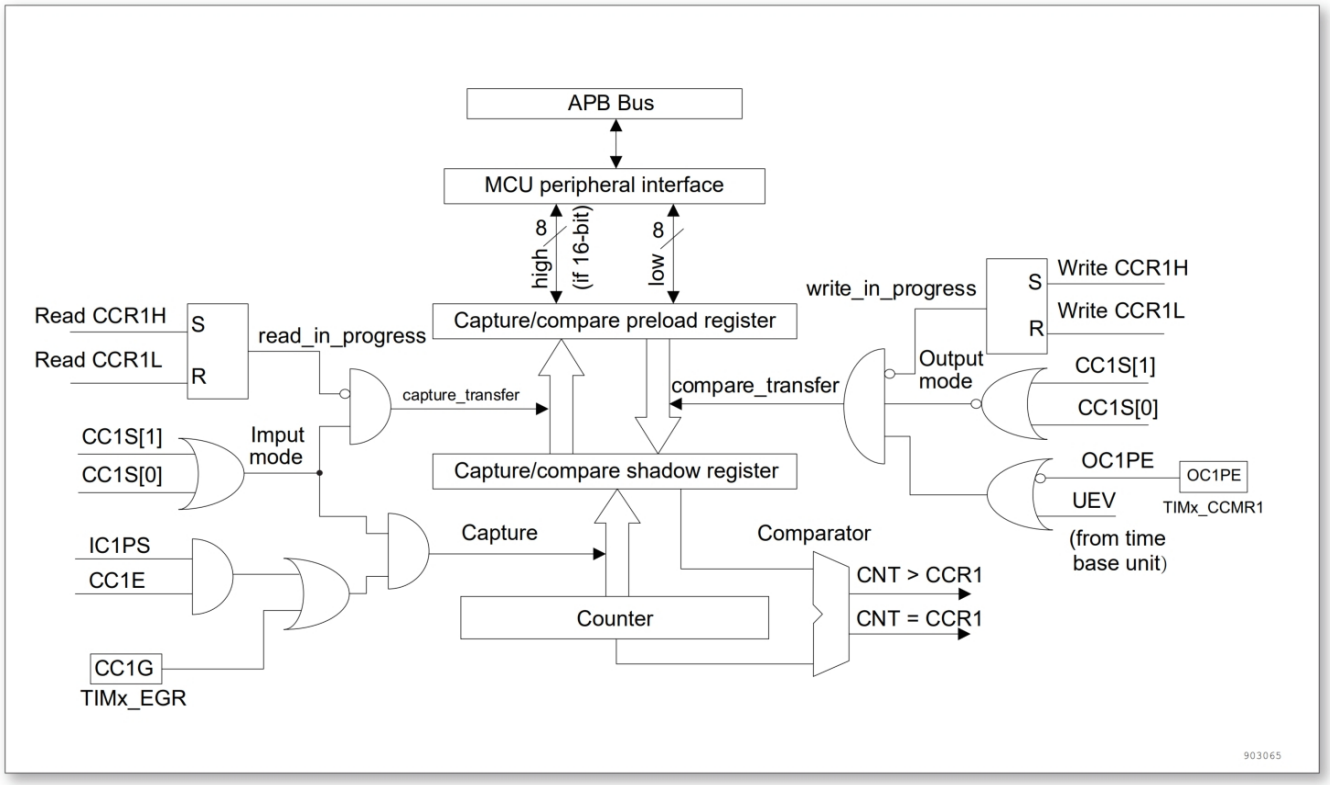


Figure 126. Capture/compare channel 1 main circuit

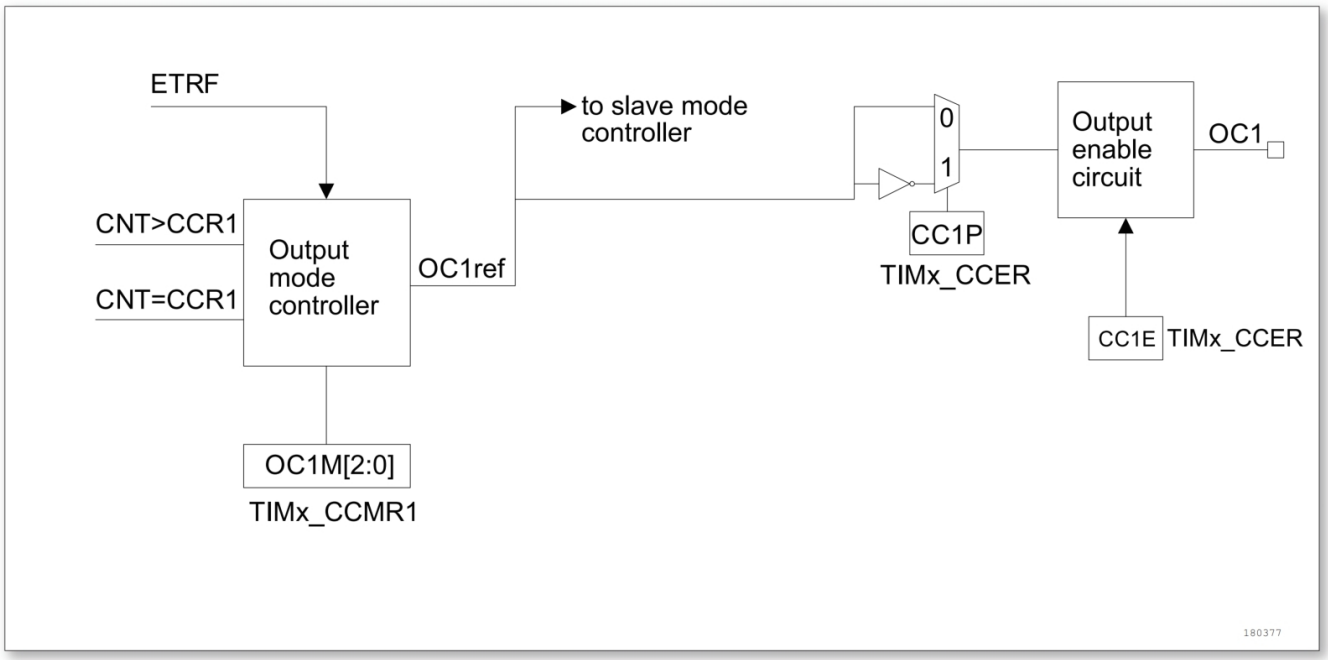


Figure 127. Output stage of capture/compare channel (channel 1)

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register. In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

11.3.5 Input capture mode

In Input capture mode, the Capture/Compare registers (TIM14_CCRx) are used to latch the value of the counter after an edge is detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIM14_SR register) is set and an interrupt can be sent if it is enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIM14_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIM14_CCRx register. CCxOF is cleared when written to '0'. The following example shows how to capture the counter value in TIM14_CCR1 when TI1 input rises. To do this, use the following procedure:

1. Select the active input: TIM14_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIM14_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIM14_CCR1 register becomes read-only.
2. Program the needed input filter duration with respect to the input signal (by programming ICxF bits in the TIM14_CCMRx register if the input is a TIx input). Let's imagine that, when toggling, the input signal is not stable during at most five internal clock cycles. We must program a filter duration longer than these five clock cycles. We can validate an edge transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at f_{DTS} frequency). Then write IC1F bits to 0011 in the TIM14_CCMR1 register.
3. Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 and CC1NP to 0 in the TIM14_CCER register (rising edge in this case).
4. Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIM14_CCMR1 register).
5. Enable capture from the counter into the capture register by setting the CC1E bit to '1' in the TIM14_CCER register.
6. If needed, enable the related interrupt request by setting the CC1IE bit in the TIM14_DIER register.

When an input capture occurs:

- The TIM14_CCR1 register gets the value of the counter on the active level transition. CC1IF flag is set (interrupt flag). CC1OF is also set to '1' if at least two consecutive captures occurred whereas the CC1IF flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: Input capture interrupt requests can be generated by software by setting the corresponding CCxG bit in the TIM14_EGR register.

11.3.6 Forced output mode

In output mode (CCxS bits = 00 in the TIM14_CCMRx register), each output compare signal (OCxREF and then OCx) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCxREF/OCx) to its active level, the user just needs to write 101 in the OCxM bits in the corresponding TIM14_CCMRx register. Thus OCXREF

is forced high (OCxREF is always active high) and OCx gets opposite value to CCxP polarity bit. For example: CCxP = 0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bit to 100 in the TIM14_CCMRx register. Anyway, the comparison between the TIM14_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt requests can be sent accordingly. This is described in the output compare mode section below.

11.3.7 Output compare mode

This function is used to control an output waveform or indicate when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

1. Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bit in the TIM14_CCMRx register) and the output polarity (CCxP bit in the TIM14_CCER register). The output pin can keep its level (OCxM = 000), be set active (OCxM = 001), be set inactive (OCxM = 010) or can toggle (OCxM = 011) on match.
2. Sets a flag in the interrupt status register (CCxIF bit in the TIM14_SR register).
3. Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIM14_DIER register).

The TIM14_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIM14_CCMRx register. In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The synchronization resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

The output compare mode is configured as follows:

- Select the counter clock (internal, external, prescaler).
- Write the desired data in the TIM14_ARR and TIM14_CCRx registers.
- Set the CCxIE bit if an interrupt request is to be generated.
- Select the output mode,
  - Write OCxM = 011 to toggle OCx output pin when CNT matches CCRx
  - Write OCxPE = 0 to disable preload
  - Write CCxP = 0 to select active high
  - Write CCxE = 1 to enable output
- Enable the counter by setting the CEN bit in the TIM14_CR1 register.

The TIM14_CCRx register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE = 0, else TIMx_CCRx shadow register is updated only at the next update event UEV). An example is given in the following figure.

(TIM14)

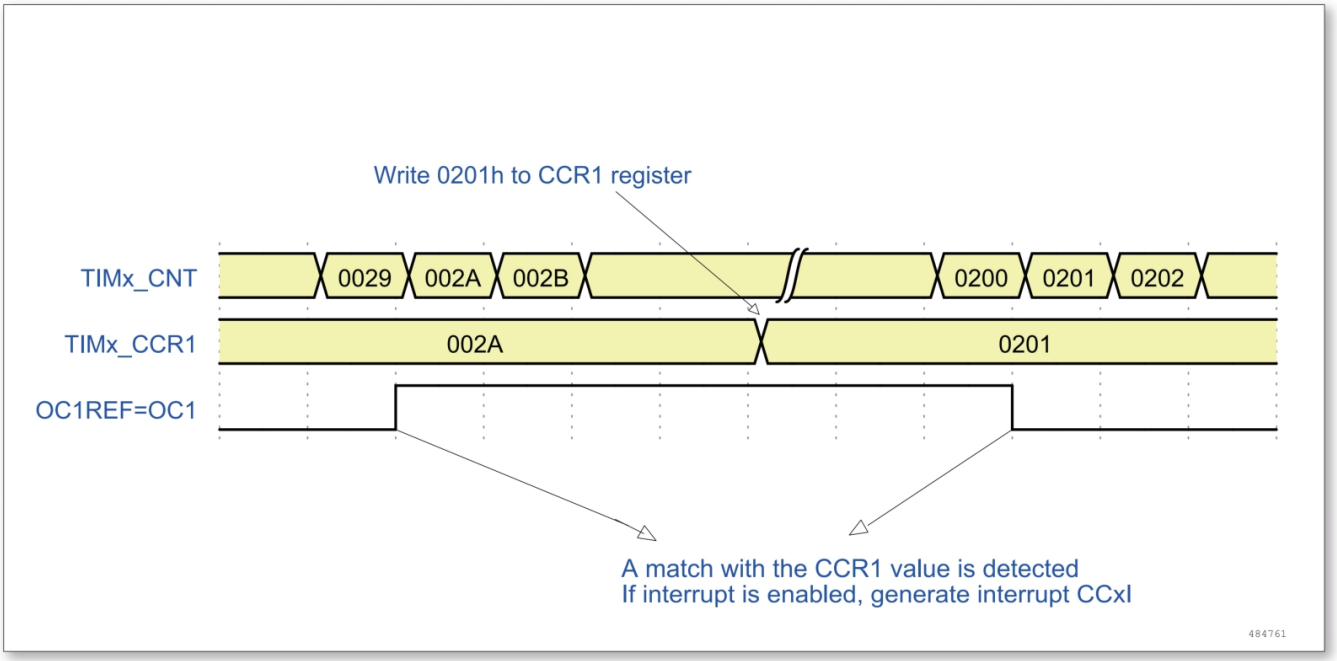


Figure 128. Output compare mode, toggle on OC1

11.3.8 PWM mode

Pulse Width Modulation mode allows generating a signal with a frequency determined by the value of the TIM14_ARR register and a duty cycle determined by the value of the TIM14_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bit in the TIM14_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIM14_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIM14_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, the user must initialize all the registers by setting the UG bit in the TIM14_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIM14_CCER register. It can be programmed as active high or active low. OCx output is enabled by using the CCxE bit in the TIM14_CCER register. For details, refer to the TIM14_CCERx register description.

In PWM mode (1 or 2), TIM14_CNT and TIM14_CCRx are always compared to determine whether TIM14_CNT ≤ TIM14_CCRx.

The reason is that this counter is counting up and can only generate PWM in edge-aligned mode.

PWM edge-aligned mode

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIM14_CNT < TIM14_CCRx; else it becomes low. If the compare value in TIM14_CCRx is greater than the auto-reload value (in TIM14_ARR), then OCxREF is held at '1'. If the compare value is 0 then OCxREF is held at '0'. The following figure shows some edge-aligned PWM waveforms in an example where TIM14_ARR = 8.

(TIM14)

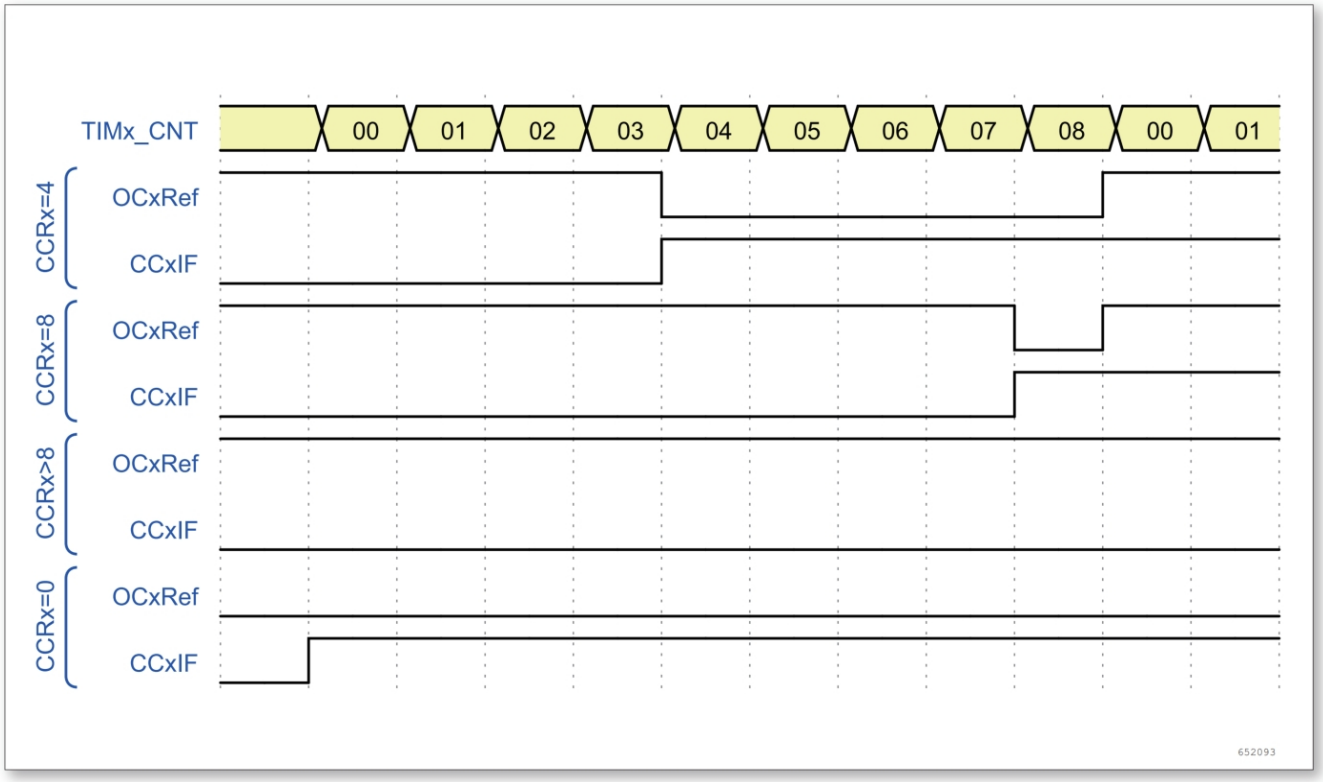


Figure 129. Edge-aligned PWM waveforms (ARR = 8)

11.3.9 Debug mode

When the microcontroller enters debug mode (Cortex®-M0 core halted), the TIM14 counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

11.4 TIM14 register description

Table 40. Overview of TIM14 registers

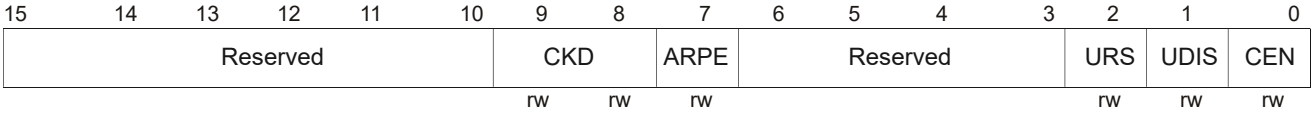
Offset	Acronym	Register Name	Reset	Section
0x00	TIM14_CR1	Control register 1	0x00000000	Section 11.4.1
0x0C	TIM14_DIER	Interrupt enable register	0x00000000	Section 11.4.2
0x10	TIM14_SR	Status register	0x00000000	Section 11.4.3
0x14	TIM14_EGR	Event generate register	0x00000000	Section 11.4.4
0x18	TIM14_CCMR1	Capture/compare mode register 1	0x00000000	Section 11.4.5
0x20	TIM14_CCER	Capture/compare enable register	0x00000000	Section 11.4.6
0x24	TIM14_CNT	Counter	0x00000000	Section 11.4.7
0x28	TIM14_PSC	Prescaler	0x00000000	Section 11.4.8
0x2C	TIM14_ARR	Auto-reload register	0x00000000	Section 11.4.9

Offset	Acronym	Register Name	Reset	Section
0x34	TIM14_CCR1	Capture/compare register 1	0x00000000	Section 11.4.10
0x44	TIM14_BDTR	Brake and dead-time register	0x00000000	Section 11.4.11

11.4.1 Control register 1 (TIM14_CR1)

Address offset: 0x00

Reset value: 0x0000



Bit	Field	Type	Reset	Description
15:10	Reserved			Reserved, read as 0
9:8	CKD	rw	0x00	Clock division These 2 bits indicate the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling frequency used by the dead-time generators and the digital filters (ETR, TIx). 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: Reserved, do not program this value
7	ARPE	rw	0x00	Auto-reload preload enable 0: TIM14_ARR register is not buffered 1: TIM14_ARR register is buffered
6:3	Reserved			Reserved and read as 0.
2	URS	rw	0x00	Update request source This bit is set by software to select the UEV event sources. 0: The Update (UEV) event is generated by one of the following events if UEV is enabled: - Counter overflow - Setting the UG bit - Update generation through the slave mode controller 1: Only counter overflow generates an update interrupt if UEV is enabled.

Bit	Field	Type	Reset	Description
1	UDIS	rw	0x00	Update disable This bit is set by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: - Counter overflow - Setting the UG bit Shadow registers are updated subsequently. 1: UEV disabled. The Update event is not generated, shadow registers (ARR, PSC, CCRx) keep their values. However the counter and the prescaler are reinitialized if the UG bit is set.
0	CEN	rw	0x00	Counter enable 0: Counter disabled 1: Counter enabled

11.4.2 Interrupt enable register (TIM14_DIER)

Address offset: 0x0C
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														CC1IE	UIE
														rw	rw

Bit	Field	Type	Reset	Description
15: 2	Reserved			Reserved, read as 0
1	CC1IE	rw	0x00	Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
0	UIE	rw	0x00	Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

11.4.3 Status register (TIM14_SR)

Address offset: 0x10

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CC1OF	Reserved							CC1IF	UIF
						rc_w0								rc_w0	rc_w0

Bit	Field	Type	Reset	Description
15:10	Reserved			Reserved, read as 0
9	CC1OF	rc_w0	0x00	Capture/Compare 1 overcapture flag This flag is set to '1' by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The counter value has been captured in TIM14_CCR1 register while CC1IF flag was already set to 1.
8: 2	Reserved			Reserved and read as 0.
1	CC1IF	rc_w0	0x00	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value. It is cleared by software. 0: No match. 1: The content of the counter TIM14_CNT matches the content of the TIM14_CCR1 register. When the contents of TIM14_CCR1 are greater than the contents of TIM14_ARR, the CC1IF bit goes high on the counter overflow. If channel CC1 is configured as input: This bit is set to '1' by hardware on a capture. It is cleared by software or by reading the TIM14_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in (copied to) TIM14_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	rc_w0	0x00	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: - At overflow and if the UDIS = 0 in the TIM14_CR1 register; - When CNT is reinitialized by software using the UG bit in TIM14_EGR register, if URS=0 and UDIS=0 in the TIM14_CR1 register.

11.4.4 Event generation register (TIM14_EGR)

Address offset: 0x14
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														CC1G	UG
														w	w

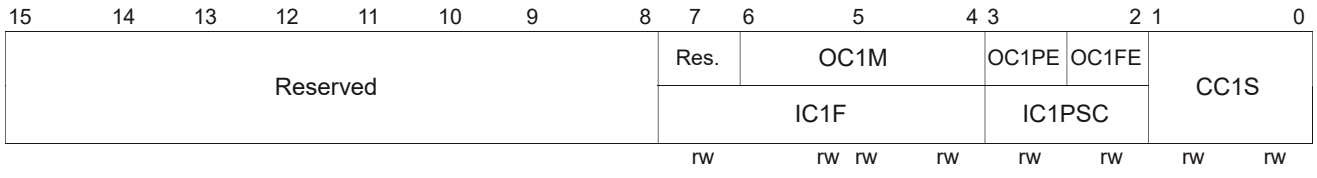
Bit	Field	Type	Reset	Description
15:2	Reserved			Reserved, read as 0
1	CC1G	w	0x00	Capture/Compare 1 generation This bit is set by software in order to generate a capture/compare event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as output: CC1IF flag is set, corresponding interrupt is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIM14_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	w	0x00	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared.

11.4.5Capture/compare mode register 1 (TIM14_CCMR1)

Address offset: 0x18

Reset value: 0x0000

The channels can be used in input (capture mode) or in output (compare mode). The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function in output mode. For a given bit, OCxx describes its function when the channel is configured in output, ICxx describes its function when the channel is configured in input. So the user must take care that the same bit can have a different meaning for the input stage and for the output stage.



Bit	Field	Type	Reset	Description
15: 7	Reserved			Reserved, read as 0
6:4	OC1M	rw	0x00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 is derived. OC1REF is active high whereas OC1 active level depends on CC1P bit. 000: Frozen. The comparison between the output compare register TIM14_CCR1 and the counter TIM14_CNT has no effect on the OC1REF. 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIM14_CNT matches the capture/compare register 1 (TIM14_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIM14_CNT matches the capture/compare register 1 (TIM14_CCR1). 011: Toggle. OC1REF level toggles when TIM14_CCR1=TIM14_CNT. 100: Force inactive level. OC1REF is forced low. 101: Force active level. OC1REF is forced high. 110: PWM mode 1 - channel 1 is active as long as TIM14_CNT < TIM14_CCR1, else inactive. 111: PWM mode 2 - channel 1 is inactive as long as TIM14_CNT < TIM14_CCR1, else active. Note: In PWM mode 1 or 2, the OC1REF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.
3	OC1PE	rw	0x00	Output compare 1 preload enable 0: Preload function of TIM14_CCR1 register disabled. TIM14_CCR1 register can be written at anytime, the new value is taken in account immediately. 1: Preload function of TIM14_CCR1 register enabled. Read/Write operations access the preload register. TIM14_CCR1 preload value is loaded in the active register at each update event. Note: The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIM14_CR1 register). Else the behavior is not guaranteed.

Bit	Field	Type	Reset	Description
2	OC1FE	rw	0x00	Output compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S	rw	0x00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: reserved 11: reserved Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIM14_CCER).

Input capture mode:

Bit	Field	Type	Reset	Description
15: 8	Reserved			Reserved, read as 0

Bit	Field	Type	Reset	Description
7:4	IC1F	rw	0x00	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to generate a transition on the output: 0000: No filter, sampling is done at f_{DTS} 1000: $f_{SAMPLING}=f_{DTS/8}$, N = 6 0001: $f_{SAMPLING}=f_{CK\_INT}$, N = 2 1001: $f_{SAMPLING}=f_{DTS/8}$, N = 8 0010: $f_{SAMPLING}=f_{CK\_INT}$, N = 4 1010: $f_{SAMPLING}=f_{DTS/16}$, N = 5 0011: $f_{SAMPLING}=f_{CK\_INT}$, N = 8 1011: $f_{SAMPLING}=f_{DTS/16}$, N = 6 0100: $f_{SAMPLING}=f_{DTS/2}$, N = 6 1100: $f_{SAMPLING}=f_{DTS/16}$, N = 8 0101: $f_{SAMPLING}=f_{DTS/2}$, N = 8 1101: $f_{SAMPLING}=f_{DTS/32}$, N = 5 0110: $f_{SAMPLING}=f_{DTS/4}$, N = 6 1110: $f_{SAMPLING}=f_{DTS/32}$, N = 6 0111: $f_{SAMPLING}=f_{DTS/4}$, N = 8 1111: $f_{SAMPLING}=f_{DTS/32}$, N = 8
3:2	IC1PSC	rw	0x00	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E = 0 (TIM14_CCER regsiter). 00: no prescaler, capture is done each time an edge is detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events

Bit	Field	Type	Reset	Description
1:0	CC1S	rw	0x00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: reserved 11: reserved Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIM14_CCER).

11.4.6 Capture/compare enable register (TIM14_CCER)

Address offset: 0x20
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												CC1NP	Res.	CC1P	CC1E
												rw		rw	rw

Bit	Field	Type	Reset	Description
15:4	Reserved			Reserved, read as 0
3	CC1NP	rw	0x00	Capture/Compare 1 complementary output polarity CC1 channel configured as output: CC1NP must be kept cleared (CC1NP = 0) CC1 channel configured as input: CC1NP is used in conjunction with CC1P to define TI1FP1 polarity (refer to CC1P description).
2	Reserved			Reserved and read as 0.

Bit	Field	Type	Reset	Description
1	CC1P	rw	0x00	Capture/Compare 1 output polarity CC1 channel configured as output: 0: OC1 active high 1: OC1 active low CC1 channel configured as input: CC1P/CC1NP bits select TI1FP1 and TI2FP1 polarity for trigger or capture operations (whether IC1 or the inverted signal of IC1 is used). 00: non-inverted/rising edge Capture is done on TIxFP1 rising edge (capture mode); TIxFP1 is not inverted. 01: inverted/falling edge edge Capture is done on TIxFP1 falling edge (capture mode); TIxFP1 is inverted. 10: reserved, do not use this configuration. 11: non-inverted/both edges Capture is done on both TIxFP1 rising and falling edges (capture mode); TIxFP1 is not inverted. Note: These bits can not be modified as long as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = 00 (the channel is configured in output).
0	CC1E	rw	0x00	Capture/Compare 1 output enable CC1 channel configured as output: 0: Off - OC1 is not active. 1: On - OC1 signal is output on the corresponding output pin. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the register TIM14_CCR1 or not. 0: capture disabled 1: capture enabled

Table 41. Output control bit for standard OCx channels

CCxE bit	OCx output state
0	Output disabled (OCx = 0, OCx_EN = 0)
1	OCx = OCxREF + Polarity, OCx_EN = 1

Note: The states of the external I/O pins connected to the standard OCx channels depend on the state of the OCx channel and the GPIO registers.

11.4.7 Counter (TIM14_CNT)

Address offset: 0x24

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x0000	Counter value

11.4.8 Prescaler (TIM14_PSC)

Address offset: 0x28

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:0	PSC	rw	0x0000	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC}/(PSC + 1). PSC contains the value to be loaded in the active prescaler register at each update event

11.4.9 Auto-reload register (TIM14_ARR)

Address offset: 0x2C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:0	ARR	rw	0x0000	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to the section of Time-base unit for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

11.4.10 Capture/compare register 1 (TIM14_CCR1)

Address offset: 0x34

Reset value: 0x0000

(TIM14)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:0	CCR1	rw	0x0000	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded immediately if the preload feature is not selected in the TIM14_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM14_CNT and signalled on OC1 output. If channel CC1 is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

11.4.11 Break and dead-time register (TIM14_BDTR)

Address offset: 0x44

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15	MOE	rw	0x00	Main output enable 0: outputs disabled. 1: outputs are enabled if their respective enable bits are set (CCxE, CCxNE in TIM14_CCER register). See OC/OCN enable description for more details (Section 11.4.6: capture/compare enable register (TIM14_CCER)).
14:0	Reserved			Reserved and always read as 0.

12 Independent Watchdog (IWDG)

Independent Watchdog (IWDG)

12.1 Independent watchdog (IWDG) introduction

The device has embedded an independent watchdog which offers a combination of high safety level, timing accuracy and flexibility of use. The independent watchdog serves to detect and resolve malfunctions due to software failure, and to trigger system reset when the counter reaches a given timeout value.

The IWDG is clocked by its own dedicated low-speed clock (LSI) and thus stays active even if the main clock fails.

The IWDG is best suited to applications which require the watchdog to run as a totally independent process outside the main application, but have lower timing accuracy constraints.

12.2 IWDG main features

- Free-running downcounter
- Clocked from an independent oscillator (can operate in Standby and Stop modes)
- Reset (if watchdog activated) when the downcounter value of 0x0000 is reached

12.3 IWDG functional description

The figure below shows the functional blocks of the independent watchdog module.

When the independent watchdog is started by writing the value 0xCCCC in the Key register (IWDG_KR), the counter starts counting down from the reset value of 0xFFF. When it reaches the end of count value (0x000), a reset signal is generated (IWGD_RESET).

Whenever the key value 0xAAAA is written in the IWDG_KR key register, the IWDG_RLR value is reloaded in the counter and the watchdog reset is prevented.

(IWDG)

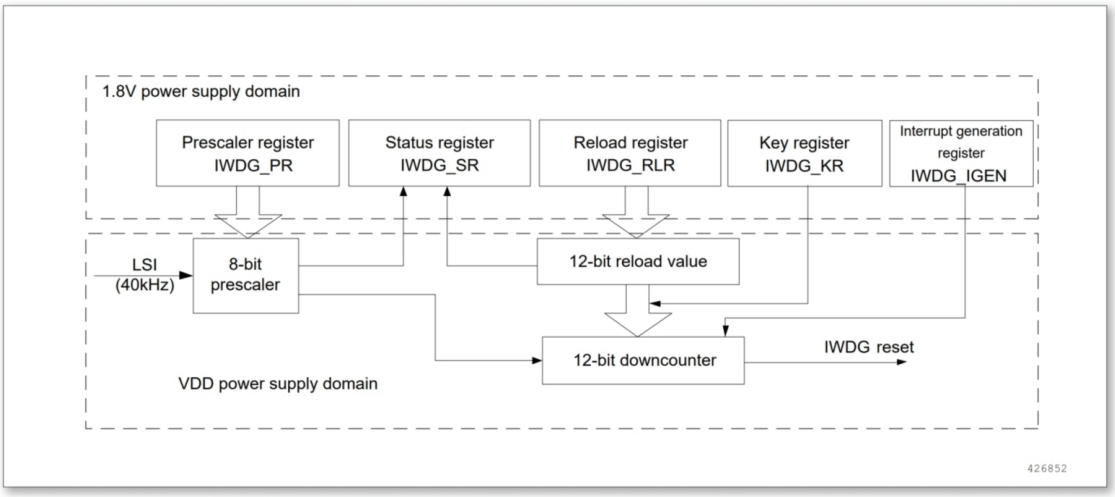


Figure 130. Independent watchdog block diagram

Note: The watchdog function is implemented in the V_{DD} voltage domain, still functional in Stop and Standby modes.

Table 42. IWDG timeout period (input clock at 40 KHz (LSI))

Prescale coefficient	PR[2:0] bits	Min timeout RL[11:0]=0x000	Max timeout
/4	0	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6
/128	5	3.2	13107.2
/256	(6 or 7)	6.4	26214.4

Note: These timings are given for a 40 kHz clock but the MCU internal oscillator frequency can vary from 30KHz to 60KHz.

Although this may be true that the oscillator frequency is accurate, the precise timing still depends on the phrase difference between the APB interface clock and the oscillator clock. As a consequence, one complete oscillator cycle will always be uncertain.

12.3.1 Hardware watchdog

If the “Hardware watchdog” feature is enabled through the device option bytes (refer to the section of Embedded Flash Memory), the watchdog is automatically enabled at power-on, and will generate a system reset unless the Key register is written by the software before the counter reaches end of count.

12.3.2 Register access protection

Write access to the IWDG_PR, IWDG_RLR, and IWDG_IGEN registers is protected. To modify them, first write the code 0x5555 in the IWDG_KR register. A write access to this register with a different value will break the sequence and register access will be

protected again. This implies that it is the same case of the reload operation (writing 0xAAAA).

A status register is available to indicate that an update of the prescaler or the down-counter value is on going.

12.3.3 Debug mode

When the microcontroller enters debug mode (CPU core halted), the IWDG counter either continues to work normally or stops, depending on DBG_IWDG_STOP configuration bit in DBG module. For more details, refer to debug module section.

12.4 IWDG register description

Table 43. Overview of IWDG registers

Offset	Acronym	Register Name	Reset	Section
0x00	IWDG_KR	Key register	0x00000000	Section 12.4.1
0x04	IWDG_PR	Prescaler register	0x00000000	Section 12.4.2
0x08	IWDG_RLR	Reload register	0x00000FFF	Section 12.4.3
0x0C	IWDG_SR	Status register	0x00000000	Section 12.4.4
0x10	IWDG_CR	Control register	0x00000000	Section 12.4.5
0x14	IWDG_IGEN	Interrupt generate register	0x00000FFF	Section 12.4.6
0x18	IWDG_CNT	Counter register	0x00000000	Section 12.4.7

12.4.1 Key register (IWDG_KR)

Address offset: 0x00

Reset value: 0x0000 0000 (reset by Standby mode)

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

Bit	Field	Type	Reset	Description
31:16	Reserved			always read as 0.
15:0	KEY	w	0x0000	Key value (write-only register, read as0x0000) These bits must be written by software at regular intervals with the key value 0xAAAA, otherwise the watchdog generates a reset when the counter reaches 0. Writing the key value 0x5555 to enable access to the IWDG_PR, IWDG_RLR, IWDG_IGEN and IWDG_CR[IRQ_SEL] registers.Writing the key value 0xCCCC starts the watchdog.

12.4.2 Prescaler register (IWDG_PR)

Address offset: 0x04

Reset value: 0x0000 0000 (reset by Standby mode)

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

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

Bit	Field	Type	Reset	Description
31:3	Reserved			always read as 0.
2:0	PR	rw	0x00	Prescaler divider These bits are write access protected. They are set to select the prescaler dividers of the counter clock. The PVU bit in the IWDG_SR register must be reset in order to be able to change the prescaler dividers. 000: prescaler divider = 4 100: prescaler divider = 64 001: prescaler divider = 8 101: prescaler divider = 128 010: prescaler divider = 16 110: prescaler divider = 256 011: prescaler divider = 32 111: prescaler divider = 256 Note: Reading this register returns the prescaler value from the V_{DD} voltage domain. This value may not be valid if a write operation is ongoing on this register. For this reason, the value read from this register is valid only when the PUV bit in the IWDG_SR register is reset.

12.4.3 Reload register (IWDG_RLR)

Address offset: 0x08

Reset value: 0x0000 0FFF (reset by Standby mode)

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

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

Bit	Field	Type	Reset	Description
31:12	Reserved			always read as 0.
11:0	RL	rw	0xFFF	Watchdog counter reload value These bits are write access protected. They serve to define the reload value of the watchdog counter. This value will be loaded to the counter each time 0xAAAA is written in the IWDG_KR register. Subsequently, the watchdog counter counts down from this value. The watchdog timeout period is a function of this value and the clock prescaler. Note: Reading this register returns the prescaler value from the V_{DD} voltage domain. This value may not be valid if a write operation is ongoing on this register. For this reason, the value read from this register is valid only when the RUV bit in the IWDG_SR register is reset.

12.4.4 Status register (IWDG_SR)

Address offset: 0x0C

Reset value: 0x0000 0000 (not reset by Standby mode)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												UPDATE	IVU	RVU	PVU
												r	r	r	r

Bit	Field	Type	Reset	Description
31:4	Reserved			always read as 0.
3	UPDATE	r	0x00	Watchdog reload value update flag This bit is set by hardware to indicate that 0xAAAA is written into the IWDG_KR register. It is reset by hardware when the reload value of the watchdog is written into the counter and the counter is updated.
2	IVU	r	0x00	Watchdog Interrupt Generate value update This bit is set by hardware to indicate that an update of the interrupt generate value is ongoing. It is reset by hardware when the interrupt generate value update operation is completed in the V_{DD} voltage domain (takes up to 5 oscillator cycles at 40KHz). The interrupt generate value can be updated only when IVU bit is reset.
1	RVU	r	0x00	Watchdog counter reload value update This bit is set by hardware to indicate that an update of the reload value is ongoing. It is reset by hardware when the reload value update operation is completed in the V_{DD} voltage domain (takes up to 5 oscillator cycles at 40KHz). The reload value can be updated only when RVU bit is reset.
0	PVU	r	0x00	Watchdog prescaler value update This bit is set by hardware to indicate that an update of the prescaler value is ongoing. It is reset by hardware when the prescaler value update operation is completed in the V_{DD} voltage domain (takes up to 5 oscillator cycles at 40KHz). The prescaler value can be updated only when RVU bit is reset.

Note: If several reload values, prescaler values or interrupt generate values are used by application, it is mandatory to wait until RVU bit is reset before changing the reload value, to wait until PVU bit is reset before changing the prescaler value and to wait until IVU bit is reset before changing the interrupt generate value. However, after updating the prescaler and/or the reload value, it is not necessary to wait until RVU or PVU is reset before continuing code execution (even in case of low-power mode entry, the write operation is taken into account and will complete).

12.4.5 IWDG control register (IWDG_CR)

Address offset: 0x10

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31:2	Reserved			Reserved, always read as 0
1	IRQ_CLR	rw	0x00	IWDG interrupt clear 1: interrupt cleared 0: no action
0	IRQ_SEL	rw	0x00	IWDG overflow operation select 1: interrupt generation enabled after overflow 0: reset generation enabled after overflow

12.4.6 IWDG interrupt generate register (IWDG_IGEN)

Address offset: 0x14

Reset value: 0x0000 0FFF

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

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

Bit	Field	Type	Reset	Description
31:12	Reserved			Reserved, always read as 0.
11:0	IGEN	rw	0xFFF	IWDG Interrupt Generate value) These bits are write access protected. They serve to define the interrupt threshold of the watchdog counter. An interrupt is generated when the watchdog counter counts down until it reaches the threshold.

12.4.7 Counter register (IWDG_CNT)

Address offset: 0x18

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT								PS							
r								r							

Bit	Field	Type	Reset	Description
31:20	Reserved			Reserved and always read as 0.
19:8	CNT	r	0x00	Current value of IWDG counter.
7:0	PS	r	0x00	Current value of IWDG prescaler counter.

13 Window Watchdog (WWDG)

Window Watchdog (WWDG)

13.1 WWDG introduction

The window watchdog is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. The watchdog circuit generates an MCU reset on expiry of a programmed time period, unless the program refreshes the contents of the downcounter before the T6 bit becomes cleared. An MCU reset is also generated by the reload counter when the downcounter value is greater than the window value. This implies that the counter must be refreshed in a limited time window.

13.2 WWDG main features

- Programmable free-running downcounter
- Conditional reset:
  - Reset (if watchdog activated) when the downcounter value becomes less than 0x40.
  - Reset (if watchdog activated) if the downcounter is reloaded outside the window.
- The early wakeup interrupt (EWI) is triggered when the downcounter reaches 0x40 (if enabled and the watchdog activated). It is applied to reload counter to avoid WWDG reset.

13.3 WWDG functional description

If the watchdog is activated (the WDGA bit is set in the WWDG_CR register) and when the 7-bit (T[6:0]) downcounter rolls over from 0x40 to 0x3F (T6 becomes cleared), it initiates a reset. If the software reloads the counter while the counter value is greater than the value stored in the window register, then a reset is generated.

(WWDG)

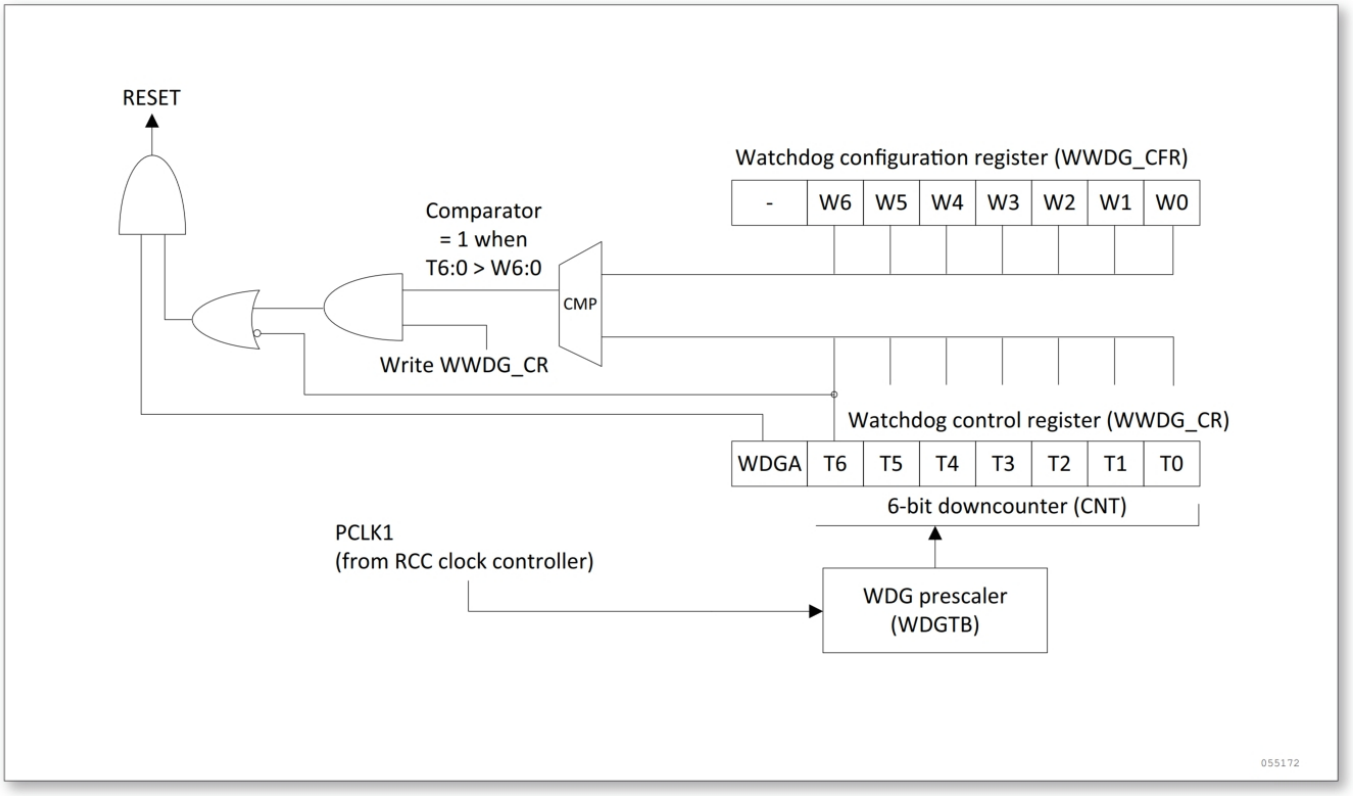


Figure 131. Watchdog block diagram

The application program must write in the WWDG_CR register at regular intervals during normal operation to prevent an MCU reset. This operation must occur only when the counter value is lower than the window register value. The value to be stored in the WWDG_CR register must be between 0xFF and 0xC0.

- Enabling the watchdog

The watchdog is always disabled after a reset. It is enabled by setting the WDGA bit in the WWDG_CR register, then it cannot be disabled again except by a reset.

- Controlling the downcounter

This downcounter is free-running, counting down even if the watchdog is disabled. When the watchdog is enabled, the T6 bit must be set to prevent generating an immediate reset.

The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset. The timing varies between a minimum and a maximum value due to the unknown status of the prescaler when writing to the WWDG_CR register.

The Configuration register (WWDG_CFR) contains the high limit of the window: To prevent a reset, the downcounter must be reloaded when its value is lower than the window register value and greater than 0x3F. The above figure describes the operation process of the window register.

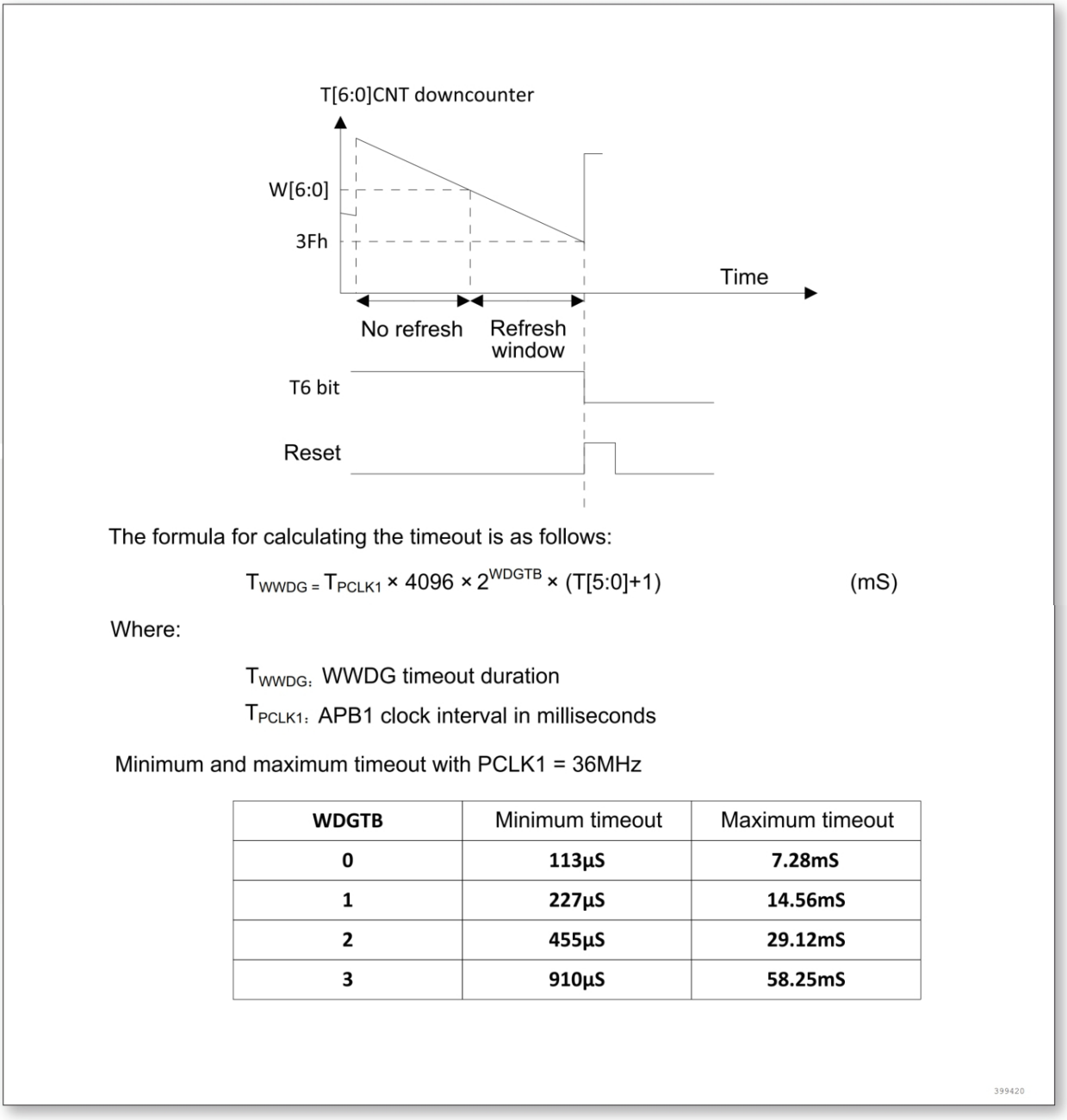
Another way to reload counter is to use the Early Wakeup Interrupt (EWI). The EWI interrupt is enabled by setting the WEI bit in the WWDG_CFR register. When the downcounter reaches the value 0x40, an EWI interrupt is generated and the corresponding interrupt service routine (ISR) can be used to set a counter value greater than 0x40, so as to avoid WWDG reset. The EWI interrupt is cleared by writing '0' to the EWIF bit in the WWDG_SR register.

Note: The T6 bit can be used to generate a software reset (the WDGA bit is set and the T6 bit is cleared).

13.4 How to program the watchdog timeout

The following figure shows the linear relationship between the increments loaded into the watchdog counter (CNT) and the time delay of the watchdog (in mS). This figure can be used as a reference for quick calculations, regardless of the time deviation. Please use the formula provided in the figure below when a higher accuracy is required.

Warning: When writing to the WWDG_CR register, always write 1 in the T6 bit to avoid generating an immediate reset.



The formula for calculating the timeout is as follows:

$$T_{WWDG} = T_{PCLK1} \times 4096 \times 2^{WDGTB} \times (T[5:0]+1) \quad (mS)$$

Where:

T_{WWDG}: WWDG timeout duration

T_{PCLK1}: APB1 clock interval in milliseconds

Minimum and maximum timeout with PCLK1 = 36MHz

WDGTB	Minimum timeout	Maximum timeout
0	**113µS**	**7.28mS**
1	**227µS**	**14.56mS**
2	**455µS**	**29.12mS**
3	**910µS**	**58.25mS**

Figure 132. Window watchdog timing diagram

13.5 Debug mode

When the microcontroller enters debug mode (CPU core halted), the WWDG counter either continues to work normally or stops, depending on DBG_WWDG_STOP configuration bit in DBG module. For more details, refer to debug module section.

13.6 WWDG register description

Table 44. Overview of WWDG registers

Offset	Acronym	Register Name	Reset	Section
0x00	WWDG_CR	Control register	0x0000007F	Section 13.6.1
0x04	WWDG_CFGR	Configuration register	0x0000007F	Section 13.6.2
0x08	WWDG_SR	Status register	0x00000000	Section 13.6.3

13.6.1 Control register (WWDG_CR)

Address offset: 0x00

Reset value: 0x0000 007F

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

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

Bit	Field	Type	Reset	Description
31:8	Reserved			Reserved, always read as 0.
7	WDGA	rw	0x00	Activation bit This bit is set by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog can generate a reset. 0: Watchdog disabled 1: Watchdog enabled
6:0	T	rw	0x7F	7 - bit counter (MSB to LSB) These bits contain the value of the watchdog counter. It is decremented by 1 every (4096 x 2^{WDGTB}) PCLK1 cycles. A watchdog reset is produced when the counter value is decremented from 40h to 3Fh (T6 becomes cleared).

13.6.2 Configuration register (WWDG_CFGR)

Address offset: 0x04

Reset value: 0x0000 007F

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

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

Bit	Field	Type	Reset	Description
31:10	Reserved			Reserved, always read as 0.
9	EWI	rw	0x00	Early wakeup interrupt When set, an interrupt occurs whenever the counter reaches the value 40h. This interrupt is only cleared by hardware after a reset.
8:7	WDGTB	rw	0x00	Timer base The time base of the prescaler can be modified as follows: 00: CK Counter Clock (PCLK1 div 4096) div 1 01: CK Counter Clock (PCLK1 div 4096) div 2 10: CK Counter Clock (PCLK1 div 4096) div 4 11: CK Counter Clock (PCLK1 div 4096) div 8
6:0	WINDOW	rw	0x7F	7-bit window value These bits contain the window value to be compared to the downcounter.

13.6.3 Status register (WWDG_SR)

Address offset: 0x08

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															EWIF
															rc_w0

Bit	Field	Type	Reset	Description
31:1	Reserved			Reserved, always read as 0.
0	EWIF	rc_w0	0x00	Early wakeup interrupt flag This bit is set by hardware when the counter has reached the value 40h. It must be cleared by software by writing '0'. A write of '1' in the bit has no effect. This bit is also set if the interrupt is not enabled.

14 Serial Peripheral Interface (SPI)

Serial Peripheral Interface (SPI)

14.1 SPI introduction

The serial peripheral interface (SPI) generally supports for communications across the boards, e.g. expanding serial Flash and ADC. Many devices form IC manufactures are compatible with SPIs.

The SPI allows full-duplex, synchronous, serial communication of MCU with external devices. The communication can be implemented by the application software through the status query and SPI interrupt.

14.2 Main features

- Fully compatible with SPI of Motorola
- Full-duplex synchronous transfers on three lines
- 16-bit programmable baud rate generator
- Master and Slave modes supported
- 8-byte FIFO transmission and reception
- Max SPI clock is PCLK/2 in the master mode (PCLK is the APB clock); max SPI clock is PCLK/4 in the slave mode
- Programmable clock polarity and phase
- Programmable data order with MSB-first or LSB-first shifting
- Operations of one master and multiple slaves are supported
- Simultaneous transmission and reception of 1- to 32-bit data
- 1- to 32-bit data transmission and reception are only supported in the LSB mode instead of the MSB mode, except for the 8-bit data
- Transmission and reception buffer, 8 bits of data size respectively, are supported
- Interrupt trigger capability
  - Tx buffer is empty
  - Tx buffer and Tx shift register are both empty
  - Tx end underflow
  - Reception of effective bytes
  - Rx buffer overflow
  - Rx buffer full
  - Reception of a specified number of bytes in the master mode

14.3 SPI functional description

14.3.1 General description

The block diagram of the SPI is shown in the diagram below.

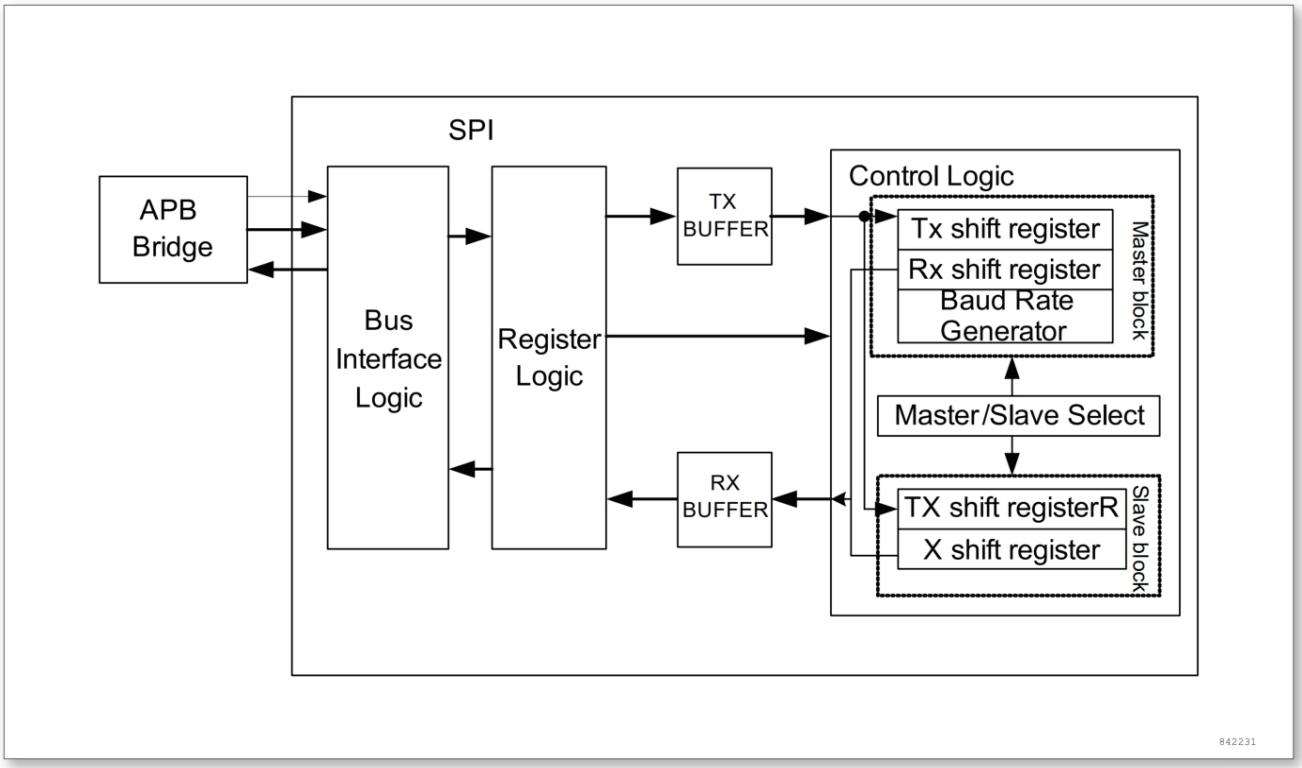


Figure 133. SPI block diagram

SPI supports simultaneous transmission and reception of 1- to 32-bit data. SPI can be configured in Slave mode or in Master mode in a Master environment. Four possible timing relationships may be chosen by configuring the clock polarity CPOL and clock phase CPHA. The programmable data order is MSB-first or LSB-first shifting.

The same clock is adopted in the transmission and reception. Data is output on the rising edge or falling edge of the clock, while latched on the opposite effective edge of SCLK. SPI serves to swap data, thus the data must be read at the end of the transfer, even though it is not significant data. In the SPI mode, the clock phases and polarities of the Master and the Slave which communicates with the Master must be identical.

Usually, the SPI is connected to external devices through four pins:

- MISO: Master In/Slave Out pin. This pin can be used to transmit data in slave mode and receive data in master mode.
- MOSI: Master Out/Slave In pin. This pin can be used to transmit data in master mode and receive data in slave mode.
- SCK: Serial Clock output for SPI masters and input for SPI slaves.
- NSS: Slave select. This is an optional pin to select a master/slave device. This pin acts as a ‘chip select’ to let the SPI master communicate with specific slaves individually and to avoid contention on the data lines. Slave NSS pins can be driven as standard IO ports on the master device. The NSS pin may also be used as an output if enabled and driven

(SPI)

low if the SPI is in master configuration. In this manner, all SPI NSS pins connected to the Master NSS pin see a low level.

An example of interconnections between a single master and a single slave is illustrated in the following figure.

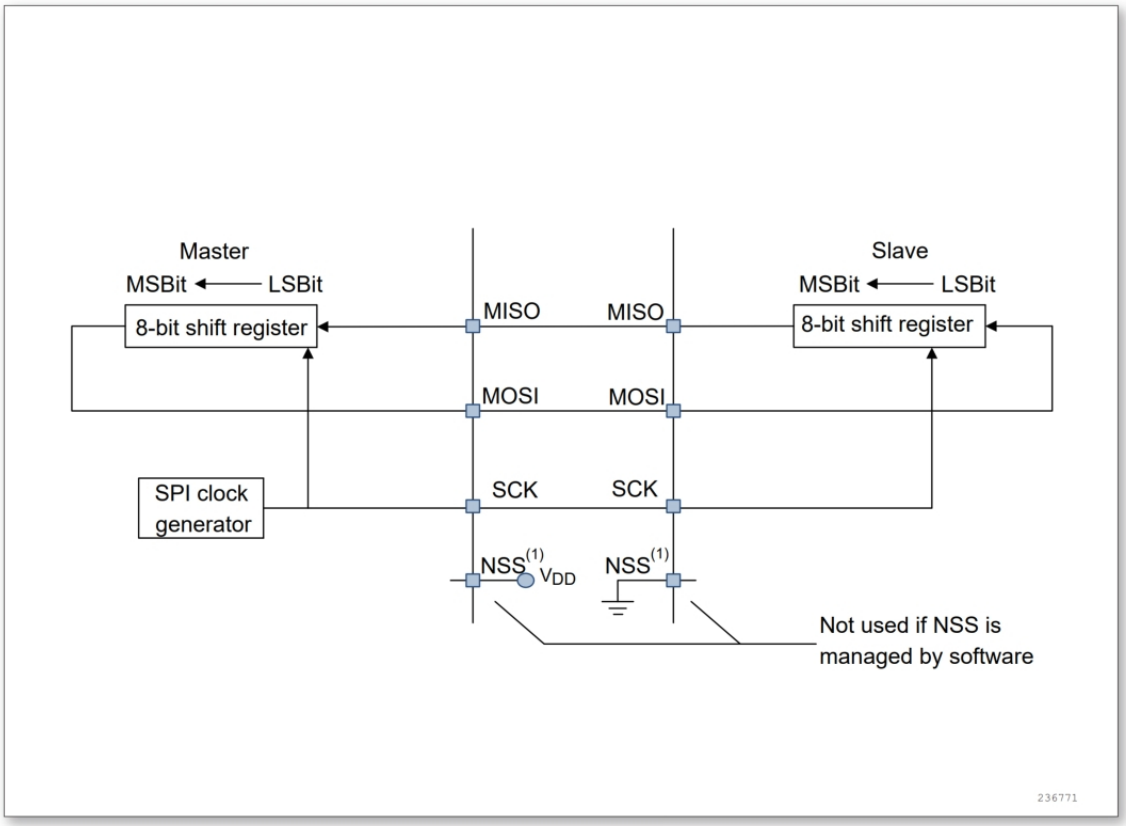


Figure 134. Single master/single slave application

The MOSI pins are connected together and the MISO pins are connected together. In this way data is transferred serially between master and slave (most significant bit first).

The communication is always initiated by the master. When the master device transmits data to a slave device via the MOSI pin, the slave device responds via the MISO pin. This implies full-duplex communication with both data out and data in synchronized with the same clock signal (which is provided by the master device via the SCK pin).

Clock signal phase and polarity

The CPOL and CPHA bits in the SPI_CCTL register can generate four possible combinations of timing relationships. The CPOL (clock polarity) bit controls the idle state level of the SCK clock when no data is being transferred. This bit affects both master and slave modes. If CPOL is reset, the SCK pin has a low-level idle state or a low level between two transmissions. If CPOL is set, the SCK pin has a high-level idle state or a high level between two transmissions.

If the CPHA (clock phase) bit is set, the first data bit is latched on the occurrence of the first clock edge (falling edge if the CPOL bit is set, rising edge if the CPOL bit is reset) of the SCK clock. Concurrently, the first received data bit is captured. SPI changes the serial data at the first transition of SCK clock in the transmission (at this point the clock shifts in the opposite direction of the idle state) and captures the data on the next edge.

If the CPHA (clock phase) bit is reset, the first data bit is latched on the occurrence of the second clock edge (rising edge if the CPOL bit is set, falling edge if the CPOL bit is reset)

of the SCK clock. Concurrently, the first received data bit is captured. SPI captures the serial data at the first transition of SCK clock in the transmission (at this point the clock shifts in the opposite direction of the idle state) and changes the data on the next edge.

The combination of the CPOL (clock polarity) and CPHA (clock phase) bits selects the data capture clock edge. The diagram 135 shows an SPI transfer with the four combinations of the CPHA and CPOL bits. The diagram may be interpreted as a master or slave timing diagram where the SCK pin, the MISO pin,

the MOSI pin are directly connected between the master and the slave device.

High-speed transmission

With regard to board-level time delay sensitivity in the high-speed transmission mode, the TXEDGE and RXEDGE control bits in the SPI_CCTL register conduct the time adjustment for transmission phase and reception sampling.

- In the Slave mode: Tx data is sent to the data bus immediately when TXEDGE = 1 (suitable for the high-speed mode, SPBRG = 4);

  Tx data is sent to the data bus after one effective clock edge when TXEDGE = 0 (suitable for the low-speed mode, SPBRG

  > 4).

- In the Master mode: Data is captured on the end edge of clock in the Tx data bits when RXEDGE = 1 (suitable for the high-speed mode);

  data is captured in the middle of the Tx data bits when RXEDGE = 0.

1. Prior to changing the CPOL/CPHA bits, the SPI must be disabled by resetting the SPIEN bit.
2. Master and slave must be programmed with the same timing mode.
3. The idle state of SCK must correspond to the polarity selected in the SPI_CCTL register (by pulling up SCK in the idle state if CPOL = 1 or pulling down SCK in the idle state if CPOL = 0).

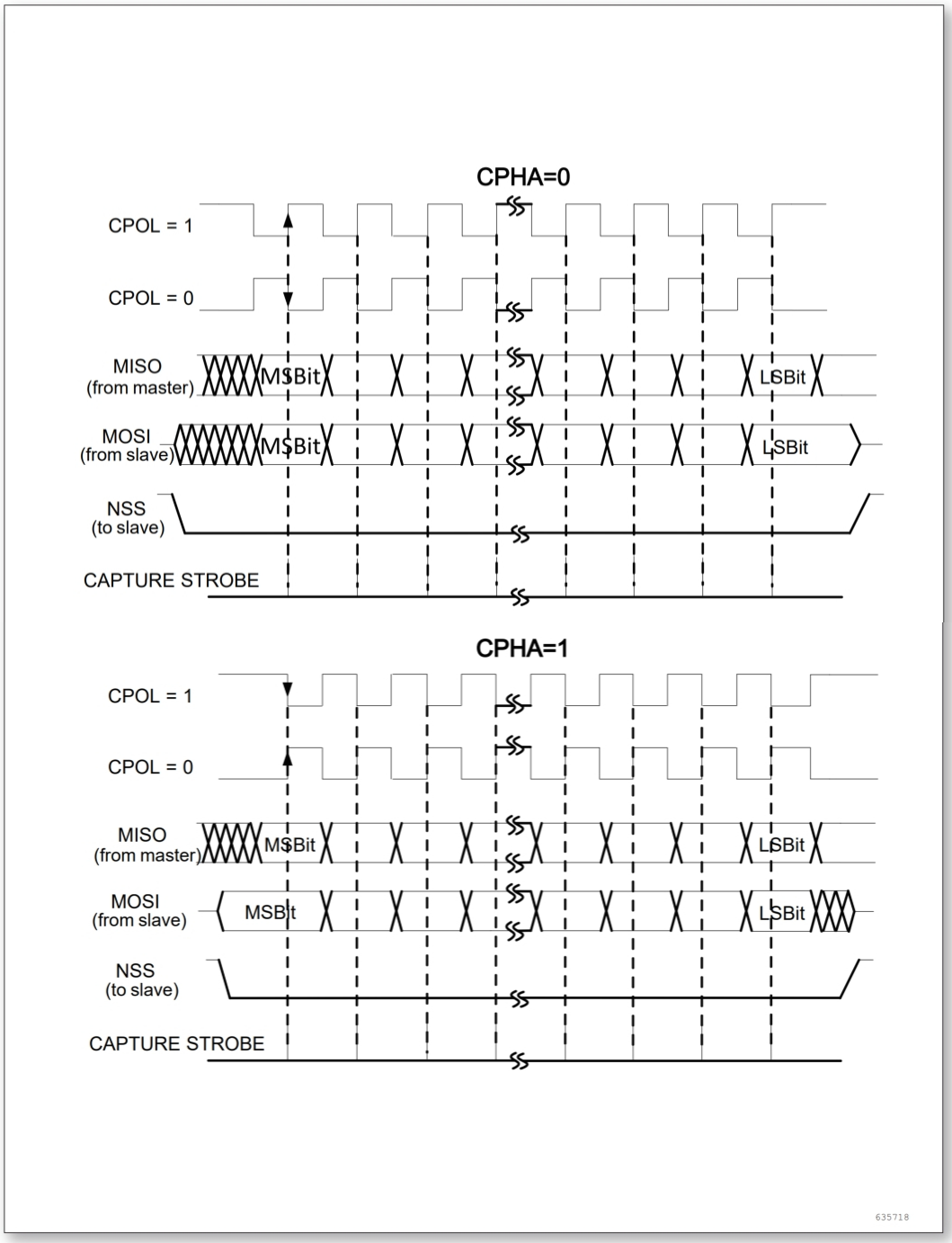


Figure 135. Data clock timing diagram

Data frame format

Data can be shifted out either MSB-first or LSB-first depending on the value of the LSBFE bit in the SPI_CCTL register.

Each data frame is 7 or 8 bits long depending on the SPILEN bit in the SPI_CCTL register. The selected data frame format is applicable for transmission and/or reception.

Moreover, 1 to 32 bits of data frame should be configured in the SPI_EXTCTL register. This configuration should be coupled with the following settings: DW8_32 bit set to 1 in the SPI_GCTL register, LSBFE bit as well as SPILEN bit set to 1 in the SPI_CCTL register.

14.3.2 SPI in slave mode

In the slave configuration, the serial clock is received on the SCK pin from the master device. The value set in the SPI_SPBRG register does not affect the data transfer rate.

Procedure

1. Set the SPILEN bit to define 7- or 8-bit data frame format.
2. Select the CPOL and CPHA bits to define phase relationships between the data transfer and the serial clock. For correct data transfer, the CPOL and CPHA bits must be configured in the same way in the slave device and the master device.
3. The frame format (MSB-first or LSB-first depending on the value of the LSBFE bit in the SPI_CCTL register) must be the same as the master device.
4. Clear the MDOE bit and set the SPIEN bit to assign corresponding pins to work in the SPI mode. In this configuration, the MOSI pin is a data input and the MISO pin is a data output.

Transmit sequence

The data byte is parallel-loaded into the Tx buffer during a write cycle.

The transmit sequence begins when the slave device receives the clock signal and the first data bit is received on its MOSI pin. The first bit is then sent. The remaining bits are loaded into the shift-register. The TX_INTF flag in the SPI_INTSTAT register is set on the transfer of data from the Tx Buffer to the shift register and an interrupt is generated if the TXIEN bit in the SPI_INTEN register is set.

Receive sequence

For the receiver, when data transfer is complete:

- The data in shift register is transferred to Rx Buffer and the RX_INTF flag is set in the SPI_INTSTAT register.
- An Interrupt is generated if the RXIEN bit is set in the SPI_INTEN register.

After the last sampling clock edge, the RXNE bit is set, a copy of the data byte received in the shift register is moved to the Rx buffer. When the SPI_RXREG register is read, the SPI peripheral returns this buffered value.

14.3.3 SPI in master mode

In the master configuration, the serial clock is generated on the SCK pin.

Procedure

1. Use the SPI_SPBRG register to define the serial clock baud rate
2. Select the CPOL and CPHA bits to define phase relationships between the data transfer and the serial clock.
3. Set the SPILEN bit to define 7- or 8-bit data frame format.
4. Configure the LSBFE bit in the SPI_CCTL register to define the frame format.
5. If the data will be only received and not transmitted, the SPI_RNDNR register should be configured to define the number of bytes to be received.
6. The MDOE and SPIEN bits must be set.

In this configuration, the MOSI pin is a data output and the MISO pin is a data input. The NSS is the Slave select signal output.

Transmit sequence

The transmit sequence begins when a byte is written in the Tx Buffer. The data byte is parallel-loaded into the shift register (from the internal bus) during the first bit transmission and then shifted out serially to the MOSI pin. MSB first or LSB first depends on theLSBFE bit in the SPI_CCTL register. The TX_INTF flag is set on the transfer of data from the Tx Buffer to the shift register and an interrupt is generated if the TXIEN bit in the SPI_INTEN register is set.

Receive sequence

For the receiver, when data transfer is complete:

- The data in shift register is transferred to Rx Buffer and the RX_INTF flag is set in the SPI_INTSTAT register.
- An Interrupt is generated if the RXIEN bit is set in the SPI_INTEN register.

After the last sampling clock edge, the RXNE bit is set, a copy of the data byte received in the shift register is moved to the Rx buffer. When the SPI_RXREG register is read, the SPI peripheral returns this buffered value.

If the data will be only received and not transmitted, the RXMATCH_INTF bit should be set on the transfer completion of bytes defined by RXDNR, which implies that all the data is received and the clock signal is no longer sent in the Master mode.

14.3.4 Status flags

To ensure the convenient operation of the software, four current status flags and seven interrupt status flags are provided for the application to monitor the state of the SPI bus. The current status flags are read-only, automatically set and reset by hardware. Interrupt status flags are set when an event occurs. A CPU interrupt is generated when the flags are enabled. They are cleared by software.

SPI has one 8-byte Tx buffer and one 8-byte Rx buffer. According to the DW8_32 bit in the SPI_GCTL, the CPU can read/write one or four bytes at one time. According to the DW8_32 bit configuration, the Tx buffer and Rx buffer have one status flag of one byte or effective entry, respectively.

Table 45. SPI status

Category	Status flags	Buffer and signal status
Interrupt status	TX_INTF	According to the DW8_32 bit configuration, the minimum space of one effective entry is provided for completing one write operation on the Tx data register.
	RX_INTF	According to the DW8_32 bit configuration, the minimum space of one effective entry is provided for completing one read operation on the Rx data register.
	UNDERRUN_INTF	Tx buffer is empty and transmission is repeated.
	RXOERR_INTF	Rx buffer is not empty and overridden.
	RXMATCH_INTF	Not empty. Last entry is transferred to the Rx buffer.
	RXFULL_INTF	Rx buffer is full and not able to receive new data.
	TXEPT_INTF	Tx buffer is empty and not able to transmit data.

(SPI)

Category	Status flags	Buffer and signal status
Current status	RXAVL_4BYTE	Rx buffer contains effective data beyond 4 bytes.
	TXFULL	Tx buffer is full.
	TXEPT	Tx buffer is empty.
	RXAVL	Rx buffer is not empty and able to receive at least one byte.

When the TXTLF = 00 in the SPI_GCTL register, the TX_INTF is set if the Tx buffer has the space not less than one idle entry; when the TXTLF = 01, the TX_INTF is set if the Tx buffer has over 50% idle space.

When the RXTLF = 00 in the SPI_GCTL register, the RX_INTF is set if the Rx buffer has not less than one effective entry;

when the RXTLF = 01, the RX_INTF is set if the Rx buffer contains effective data accounting for over 50%.

14.3.5 Baud rate configuration

The baud rate is the generated frequency of SCLK, generally the frequency division of PCLK. BRG is a 16-bit baud rate generator.

The SPBREG register controls the counting cycle of the 16-bit counter.

If expected baud rate and f_{pclk} (frequency of APB module) are provided, the formula shown in the following table can be used to calculate an approximate value to assign the SPBRG register, where X represents the value for SPBRG register (2 ~ 65535).

Table 46. Baud rate formula

Mode	Formula
SPI mode	Baud rate = f_{pclk}/X

14.4 Register file and memory mapping description

Table 47. Overview of SPI registers

Offset	Acronym	Register Name	Reset	Section
0x00	SPI_TXREG	Tx data register	0x00000000	Section 14.4.1
0x04	SPI_RXREG	Rx data register	0x00000000	Section 14.4.2
0x08	SPI_CSTAT	Current status register	0x00000001	Section 14.4.3
0x0C	SPI_INTSTAT	Interrupt status register	0x00000000	Section 14.4.4
0x10	SPI_INTEN	Interrupt enable register	0x00000000	Section 14.4.5
0x14	SPI_INTCLR	Interrupt clear register	0x00000000	Section 14.4.6
0x18	SPI_GCTL	Global control register	0x00000004	Section 14.4.7

Offset	Acronym	Register Name	Reset	Section
0x1C	SPI_CCTL	Common control register	0x00000008	Section 14.4.8
0x20	SPI_SPBRG	Baud rate generator	0x00000002	Section 14.4.9
0x24	SPI_RXDNR	Rx data number register	0x00000001	Section 14.4.10
0x28	SPI_NSSR	Slave select register	0x000000FF	Section 14.4.11
0x2C	SPI_EXTCTL	Data control register	0x00000008	Section 14.4.12

14.4.1 Transmit data register (SPI_TXREG)

Address offset: 0x00

Reset value: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:0	TXREG	rw	0x0000 0000	Transmit data register The effective number of bits is controlled by the DW8_32 bit. 0: only low 8 bits are effective 1: TXREG[31：0] bits are all effective

14.4.2 Receive data register (SPI_RXREG)

Address offset: 0x04

Reset value: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:0	RXREG	r	0x0000 0000	Receive data register The effective number of bits is controlled by the DW8_32 bit. 0: only low 8 bits are effective 1: RXREG[31：0] bits are all effective. This register is read-only and not writable.

14.4.3 Current status register (SPI_CSTAT)

Address offset: 0x08

Reset value: 0x0000 0001

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				RXFADDR				TXFADDR				RXAVL_4BYTE	TXFULL	RXAVL	TXEPT
				r	r	r	r	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31:12	Reserved			always read as 0
11:8	RXFADDR	r	0x00	number of effective data in the current Rx buffer
7:4	TXFADDR	r	0x00	number of effective data in the current Tx buffer
3	RXAVL_4BYTE	r	0x00	Receive available 4 byte data message 1: Rx buffer contains data beyond 4 bytes. 0: Rx buffer contains data less than 4 bytes.
2	TXFULL	r	0x00	Transmitter FIFO full status bit 1: Tx buffer is full. 0: Tx buffer is not full.
1	RXAVL	r	0x00	Receive available byte data message This bit is set when the Rx buffer receives a complete byte. 1: Rx buffer has received an effective byte. 0: Rx buffer is empty. This bit is read-only, automatically set and reset by hardware.
0	TXEPT	r	0x01	Transmitter empty bit 1: Tx buffer and Tx shift register are both empty. 0: Tx buffer is not empty. This bit is read-only, automatically set and reset by hardware.

14.4.4 Interrupt status register (SPI_INTSTAT)

Address offset: 0x0C

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TXEPT_INTF	RXFULL_INTF	RX MATCH _INFT	RXO ERR_ INFT	UNDE RRUN _INTF	RX_ INTF	TX_ INTF
									r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31:7	Reserved			always read as 0.
6	TXEPT_INTF	r	0x00	Transmitter empty interrupt flag bit This bit is set automatically by hardware. The TXEPT_ICLR bit is cleared by writing to the INTCLR register. 1: Tx buffer and Tx shift register are both empty. 0: Transmitter is not empty. Note: This bit is an interrupt status signal whereas TXEPT is a status signal.
5	RXFULL_INTF	r	0x00	RX FIFO full interrupt flag bit This bit is set automatically by hardware. The RXFULL_ICLR bit is cleared by writing to the INTCLR register. 1: RX buffer is full. 0: RX buffer is not full.
4	RXMATCH_ INTF	r	0x00	Receive data match the RXDNR number, the receive process will be completed and generate the interrupt This bit is set automatically by hardware. The RXMATCH_ICLR bit is cleared by writing to the INTCLR register. 1: A number of bytes specified by the RXDNR register are received. 0: A number of bytes specified by the RXDNR register are not received.
3	RXOERR_ INTF	r	0x00	Receive overrun error interrupt flag bit) This bit is set automatically by hardware. The RXOERR_ICLR bit is cleared by writing to the INTCLR register. 1: overrun error 0: no overrun error

Bit	Field	Type	Reset	Description
2	UNDERRUN_INTF	r	0x00	SPI underrun interrupt flag bit in the Slave mode This bit is set automatically by hardware. The UNDERRUN_ICLR bit is cleared by writing to the INTCLR register. 1: underrun error 0: no underrun error
1	RX_INTF	r	0x00	Receive data available interrupt flag bit This bit is set automatically by hardware. The RX_ICLR bit is cleared by writing to the INTCLR register. This bit is set when the Rx buffer receives a complete byte. 1: Rx buffer has received data of effective byte. 0: Rx buffer is empty.
0	TX_INTF	r	0x00	Transmit FIFO available interrupt flag bit (one byte is sent) This bit is automatically set by hardware and reset when the Tx buffer is not empty. 1: Tx buffer is valid. 0: Tx buffer is invalid.

14.4.5 Interrup enable register (SPI_INTEN)

Address offset: 0x10

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TXEPT_IEN	RXFULL_IEN	RX MATCH_IEN	RXO ERR_IEN	UNDE RRUN_IEN	RX_IEN	TX_IEN
									rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:7	Reserved			always read as 0.
6	TXEPT_IEN	rw	0x00	Transmit empty interrupt enable bit 1: interrupt enabled 0: interrupt disabled
5	RXFULL_IEN	rw	0x00	Receive FIFO full interrupt enable bit 1: interrupt enabled 0: interrupt disabled
4	RXMATCH_IEN	rw	0x00	Receive data match the RXDNR number, the receive process will be completed and enable the interrupt 1: interrupt enabled 0: interrupt disabled
3	RXOERR_IEN	rw	0x00	Receive overrun error interrupt enable bit 1: interrupt enabled 0: interrupt disabled
2	UNDERRUN_IEN	rw	0x00	Transmitter underrun interrupt enable bit (SPI slave mode only) 1: interrupt enabled 0: interrupt disabled
1	RX_IEN	rw	0x00	Receive FIFO interrupt enable bit 1: interrupt enabled 0: interrupt disabled
0	TX_IEN	rw	0x00	Transmit FIFO empty interrupt enable bit 1: interrupt enabled 0: interrupt disabled

14.4.6 Interrupt clear register (SPI_INTCLR)

Address offset: 0x14

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TXEPT_ICLR	RXFULL_ICLR	RX MATCH_ICLR	RXO ERR_ICLR	UNDE RRUN_ICLR	RX_ICLR	TX_ICLR
									w	w	w	w	w	w	r

Bit	Field	Type	Reset	Description
31:7	Reserved			always read as 0
6	TXEPT_ICLR	w	0x00	Transmitter empty interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
5	RXFULL_ICLR	w	0x00	Receiver buffer full interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
4	RXMATCH_ICLR	w	0x00	Receive data match the RXDNR number interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
3	RXOERR_ICLR	w	0x00	Receive overrun error interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
2	UNDERRUN_ICLR	w	0x00	Transmitter underrun interrupt clear bit (SPI slave mode only) 1: interrupt cleared 0: interrupt uncleared
1	RX_ICLR	w	0x00	Receive interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
0	TX_ICLR	r	0x00	Transmitter FIFO empty interrupt clear bit 1: interrupt cleared 0: interrupt uncleared

14.4.7 Global control register (SPI_GCTL)

Address offset: 0x18

Reset value: 0x0000 0004

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			NSS TOG	DW8_32	NSS	Reserved					RXEN	TXEN	MODE	INTEN	SPIEN
			rw	rw	rw						rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:13	Reserved			always read as 0
12	NSSTOG	rw	0x00	Slave select automatic toggle 1: NSS signal toggles automatically on the completion of data transfer 0: NSS signal does not toggle Note: This bit is only valid in the Master mode.
11	DW8_32	rw	0x00	Valid byte or doubleword data select signal 0: only low 8 bits are effective 1: 32 bits are all effective Note: The access is allowed in the specified data format.
10	NSS	rw	0x00	NSS select signal that from software or hardware 0: controlled by the NSSR register 1: controlled by hardware automatically during data transfer
9: 5	Reserved			always read as 0
4	RXEN	rw	0x00	Receive enable bit 1: receive enabled 0: receive disabled Meanwhile, Rx buffer can be cleared. Note: When SPI is working only in the Master receive mode, txen must be set to 0.

Bit	Field	Type	Reset	Description
3	TXEN	rw	0x00	Transmit enable bit 1: transmit enabled 0: transmit disabled Meanwhile, Tx buffer can be cleared. Note: Transmission and reception are simultaneous in the Master mode.
2	MODE	rw	0x01	Master mode bit 1: Master mode (the internal BRG generates the serial clock) 0: Slave mode (the external host generates the serial clock)
1	INTEN	rw	0x00	SPI interrupt enable bit 1: SPI interrupt enabled 0: SPI interrupt disabled
0	SPIEN	rw	0x00	SPI select bit 0 : SPI disabled (reset status) 1 : SPI enabled

14.4.8 Common control register (SPI_CCTL)

Address offset: 0x1C

Reset value: 0x0000 0008

(SPI)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									CPHASEL	TXEDGE	RXEDGE	SPILEN	LSBFE	CPOL	CPHA
									rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:7	Reserved			always read as 0.
6	CPHASEL	rw	0x00	CPHA polarity invert select 1: CPHA is configured as inverted. When CPHA =1, the first data bit capture starts on the second clock edge. When CPHA =0, the first data bit capture starts on the first clock edge. 0: CPHA configuration is not changed.
5	TXEDGE	rw	0x00	Transmit data edge select (Slave mode) 1: Tx data is sent to the data bus immediately (suitable for the high-speed mode, SPBRG = 4). 0: Tx data is sent to the data bus after one effective clock edge (suitable for the low-speed mode, SPBRG > 4).
4	RXEDGE	rw	0x00	Receive data edge select (Master mode) 1: Data is captured on the end edge of clock in the Tx data bits (suitable for the high-speed mode). 0: Data is captured in the middle of the Tx data bits.
3	SPILEN	rw	0x01	SPI character length bit This bit is enabled after the DW8_32 is set (DW8_32 = 0). 1: 8 characters (default) 0: 7 characters
2	LSBFE	rw	0x00	LSBFE: LSB first enable bit 1: least significant bit first in the data Tx or Rx 0: most significant bit first in the data Tx or Rx
1	CPOL	rw	0x00	Clock polarity select bit 1: clock has a high level when idle (between two transmissions). 0: clock has a low level when idle (between two transmissions).
0	CPHA	rw	0x00	Clock phase select bit 1: the first data bit capture starts on the first clock edge. 0: the first data bit capture starts on the second clock edge.

14.4.9 Baud rate generator (SPI_SPBRG)

Address offset: 0x20

Reset value: 0x0000 0002

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

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

Bit	Field	Type	Reset	Description
31:16	Reserved			always read as 0.
15:0	SPBRG	rw	0x0002	SPI baud rate control register for baud rate Baud rate formula: Baud rate = f_{pclk}/SPBRG (f_{pclk} is the APB clock frequency) Note: Never write ‘0’ or ‘1’ to this register.

14.4.10 Receive data number register (SPI_RXDNR)

Address offset: 0x24

Reset value: 0x0000 0001

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

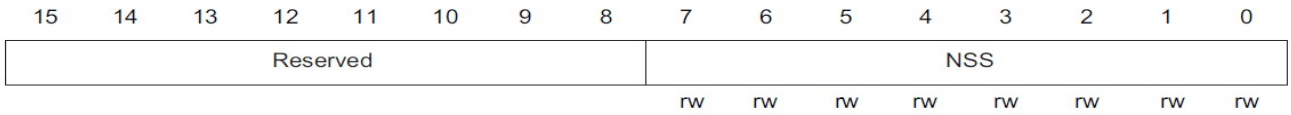
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDNR															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			always read as 0.
15:0	RXDNR	rw	0x0001	The register is used to hold a count of to be received bytes in next receive process. The value of this register is valid when the SPI is in the Master receive mode. The default value is 1. It can be written by MCU. Note: Never write '0' to this register.

14.4.11 Slave chip select register (SPI_NSSR)

Address offset: 0x28

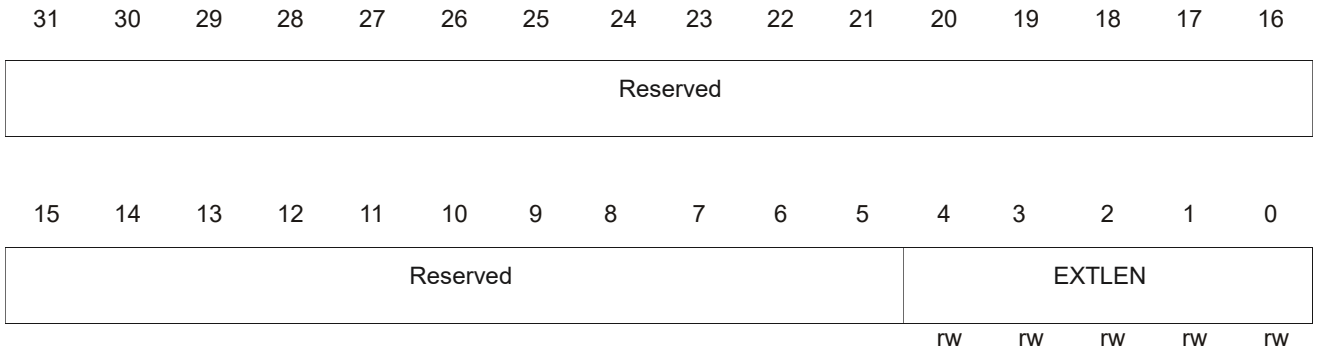
Reset value: 0x0000 00FF



Bit	Field	Type	Reset	Description
15:8	Reserved			
7: 0	NSS	rw	0xFF	Chip select output signal in Master mode Valid in the lower bit, invalid in the Slave mode 0: slave is selected 1: slave is not selected

14.4.12 Data control register (SPI_EXTCTL)

Address offset: 0x2C
Reset value: 0x0000 0008



Bit	Field	Type	Reset	Description
31:5	Reserved			always read as 0.
4:0	EXTLEN	rw	0x08	Control the SPI character length 0 0000: 32 bits 0 0001: 1 bits 0 0010: 2 bits 0 0011: 3 bits … 1 1100: 28 bits 1 1101: 29 bits 1 1110: 30 bits 1 1111: 31 bits Note: This bit is valid only when the DW8_32 bit is set to ‘0’ in the SPI_GCTL register, LSBFE bit as well as SPILEN bit set to ‘1’ in the SPI_CCTL register.

15 Inter-Integrated Circuit (I2C) Interface

Inter-Integrated Circuit (I2C) Interface

15.1 I2C introduction

I2C (inter-integrated circuit) bus interface serves as an interface between the microcontroller and the serial I2C bus. It provides multimaster capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.

The I2C bus is a two-wire serial interface, where the serial data line (SDA) and the serial clock line (SCL) pass information between the devices connected to the bus. Each device is addressable by a unique address and can operate as a transmitter or receiver. Besides a transmitter or receiver, the device can also be considered as a master or slave when it conducts data transfer. A master is a device that initializes the data transfer on the bus and generates transportable clock signals. At this point, any addressable device is considered as a slave.

I2C can operate in the Standard mode (data transfer rate between 0 to 100Kbps) and Fast mode (data transfer rate up to 400Kbps).

15.2 I2C main features

- Parallel-bus I2C bus protocol converter
- Half-duplex synchronous operation
- Master and Slave modes supported
- 7-bit/10-bit address supported
- Standard mode (100Kbps) and Fast mode (400Kbps) supported
- Start, Stop, and Restart generation, alongside Acknowledge signal detection
- A sole master supported in the Master mode
- Respective 2-byte Tx and Rx FIFOs
- Spike-free circuit added in the SCLI and SDAI
- Interrupt and query operations supported
- I2C slave multi-addressing capability supported (see the register description for details)

15.3 I2C protocol

15.3.1 Start and Stop conditions

When the bus is in the idle state, SCL and SDA are pulled up by an external pull-up resistor simultaneously.The data transfer initialized in the Master begins with a Start

condition. A high to low transition on the SDA line while SCL is high defines a Start condition. The data transfer in the Master is terminated by by the Master sending a Stop condition. A low to high transition on the SDA line while SCL is high defines a Stop condition. A timing diagram of Start and Stop conditions is shown as below. SDA must maintain steady during the data transfer when SCL = 1.

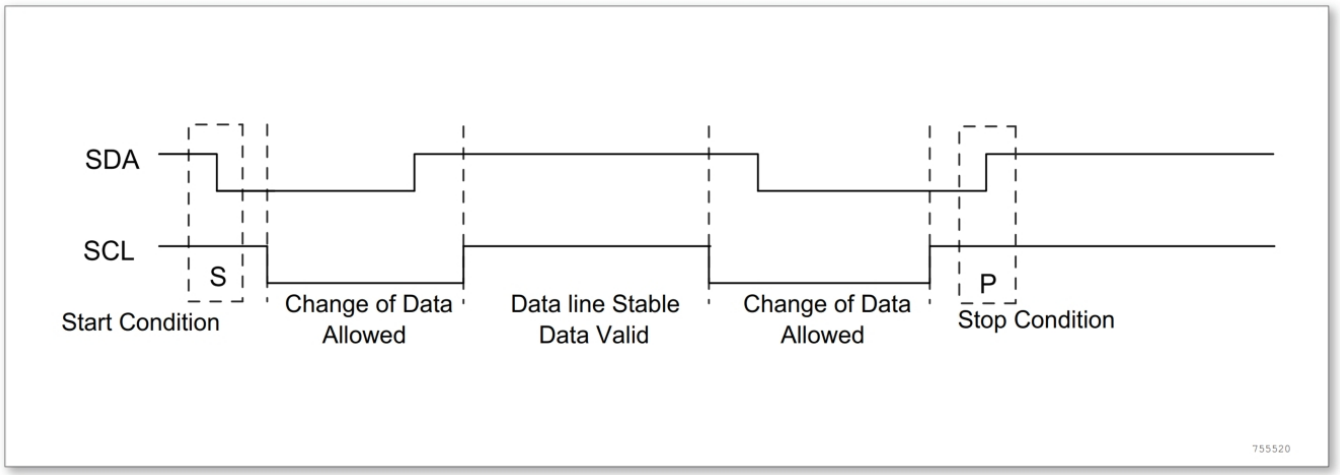


Figure 136. Start and Stop conditions

15.3.2 Slave addressing protocol

I2C has two address formats: 7-bit address format and 10-bit address format.

7-bit address format

The following figure shows the slave address is formed from the first 7 bits (bit 7:1) of one byte sent following a Start condition (S).

The least significant bit (bit 0) determines the direction of the message. If bit 0 = 0, then the Master writes data to the Slave; if bit 0 = 1, then the Master reads data from the Slave.

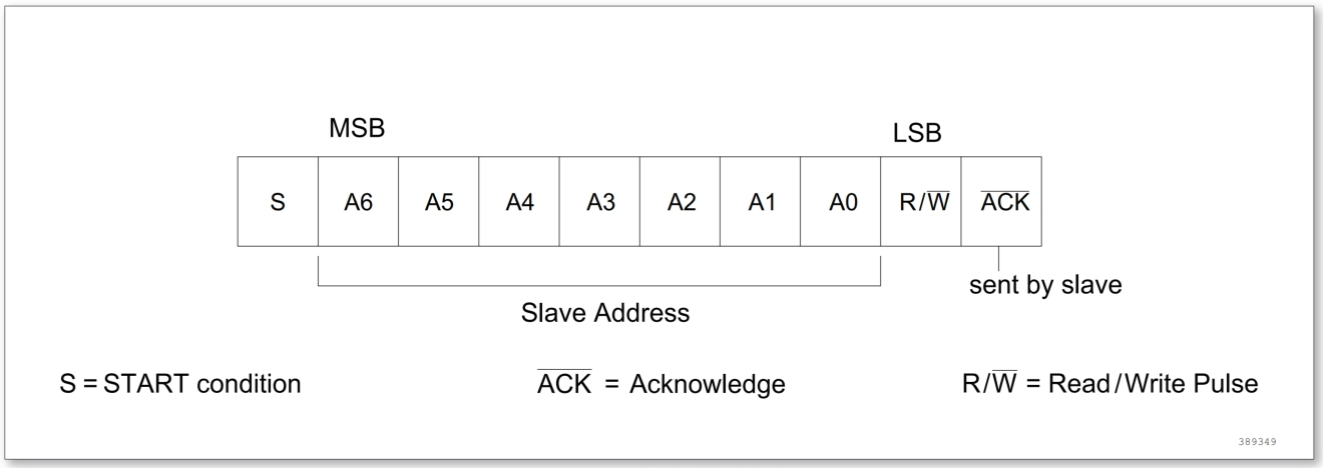


Figure 137. 7-bit address format

10-bit address format

During 10-bit addressing, two bytes are transferred to set the 10-bit address. The transfer of the first byte contains the following bit definition. The first five bits (bits 7:3) notify the slaves that this is a 10-bit transfer followed by the next two bits (bits 2:1), which set the

slaves address bits 9:8, and the LSB bit (bit 0) determines the direction of the message (R/W). The second byte transferred sets lower eight bits of the 10-bit address.

For details, see the diagram below:

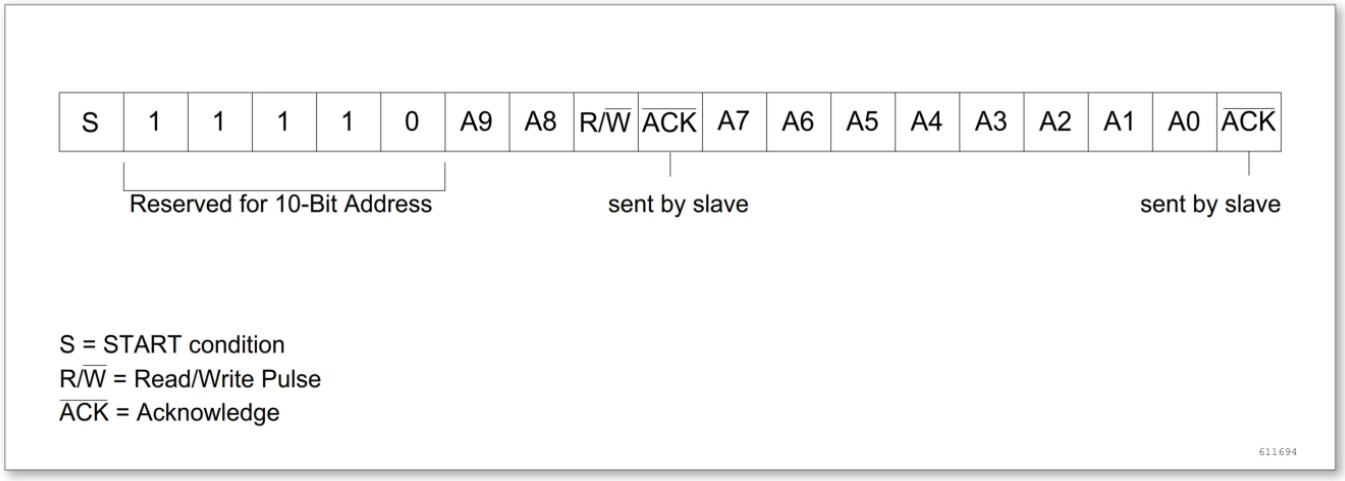


Figure 138. 10-bit address format

The following table defines the special purposes and reserved addresses of first bytes for I2C:

Table 48. First bytes for I2C

Slave Address	R/W Bit	Description
0000 000	0	General call address. I2C places the data in the receive buffer and issues a general call interrupt.
0000 000	1	Start byte.
0000 001	X	CBUS address. I2C interface ignores these accesses.
0000 010	X	Reserved
0000 011	X	Reserved
0000 1xx	X	Reserved
1111 1xx	X	Reserved
1111 0xx	X	10-bit slave addressing.

15.3.3 Transmitting and receiving protocol

The master can initiate data transmission and reception to or from the bus, acting as either a master-transmitter or master-receiver. A slave responds to requests from the master to either transmit data or receive data, acting as either a slave-transmitter or slave-receiver.

Master transmit and Slave receive

All data is transmitted in byte format, with no limit on the number of bytes per data transfer. After the master sends the address and R/W bit or the master transmits a byte of data to the slave, the slave-receiver must respond with the acknowledge signal (ACK). When a slave-receiver does not respond with an ACK pulse, the master aborts the transfer by issuing a Stop condition. If a slave is not able to respond, it slave must leave the SDA line high so that the master can abort the transfer.

If the master-transmitter is transmitting data as shown in the following figure, then the slave-receiver responds to the master-transmitter with an ACK pulse after every byte of data is received.

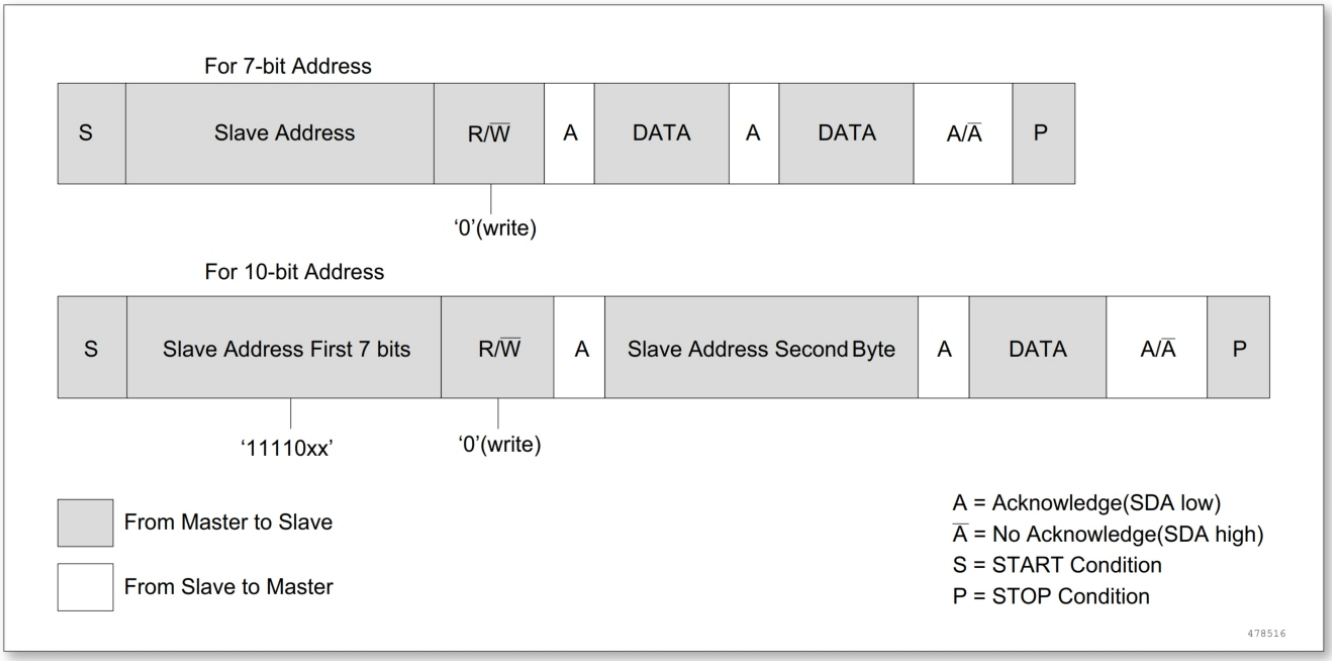


Figure 139. Master transmit protocol

Master receive and Slave transmit

If the master is receiving data as shown in the following figure, then the master responds to the slave-transmitter after a byte of data has been received, except for the last byte. This is the way the master-receiver notifies the slave-transmitter that this is the last byte. The slave-transmitter relinquishes the SDA line after detecting the No Acknowledge (NACK) bit so that the master can issue a Stop condition.

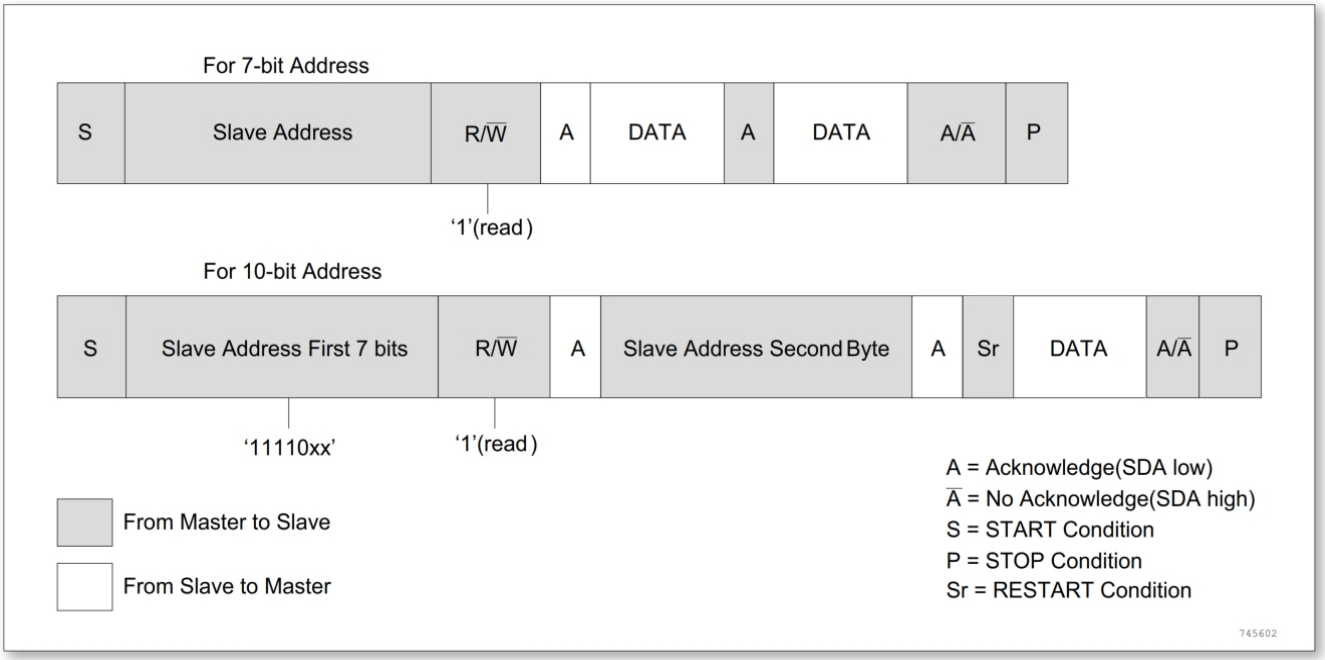


Figure 140. Master receive protocol

When a master does not want to relinquish the bus with a Stop condition, the master can issue a Restart condition. This is identical to a Start condition except it occurs after the ACK. Operating in Master mode, the I2C interface can then communicate with the same slave using a transfer of a different direction.

Start Byte transfer protocol

The Start Byte transfer protocol is set up for systems that do not have a dedicated I2C hardware module. When I2C module is set as a master, it supports the generation of Start Byte transfers at the beginning of every transfer in case a slave device requires it.

This protocol consists of seven zeros followed by a 1, as illustrated in the following figure. This allows the processor that is polling the bus to under-sample the address phase until the processor detects a 0. Once the processor detects a 0, it switches from the under sampling rate to the correct rate of the master.

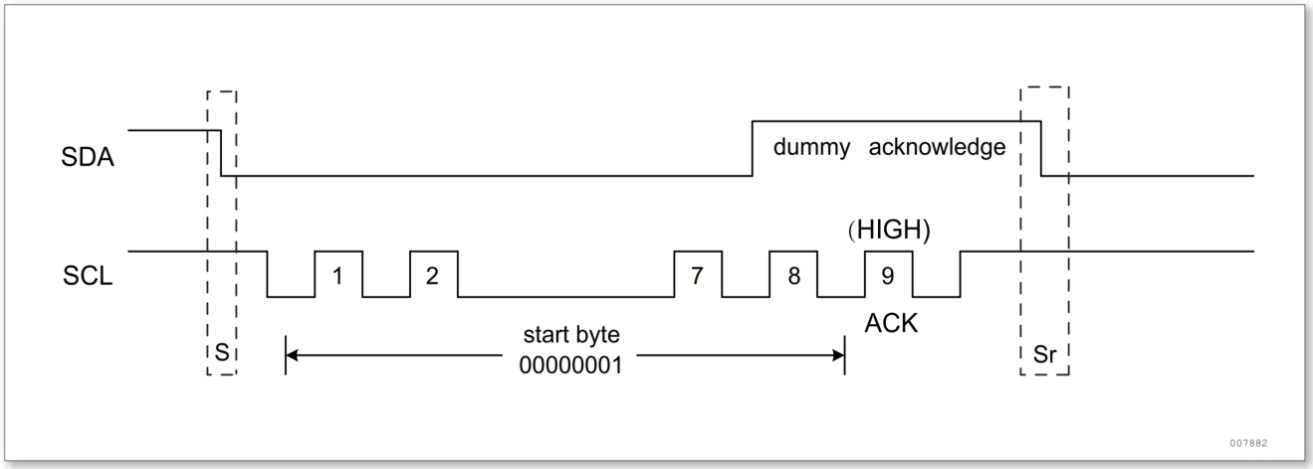


Figure 141. Start Byte transfer

The Start Byte has the following procedure:

1. Master generates a Start condition
2. Master transmits the Start byte (0000 0001)
3. Master transmits the ACK clock pulse (ACK)
4. No slave responds with an ACK pulse
5. Master generates a Restart condition (RESTART)

A hardware I2C receiver does not respond to the Start Byte because it is a reserved address and resets after the Restart condition is generated.

15.3.4 Tx buffer management and Start, Stop, and Restart generation

Operating in Master mode, I2C module generates a Stop condition once the Tx buffer is empty. If the RESTART generation function is enabled (RESTART = 1), then a Restart condition is generated when the transfer direction switches from read/write to write/read. If the Restart condition is disabled, then a Start condition is generated following a Stop condition.

The figure below shows bits of a DR register.

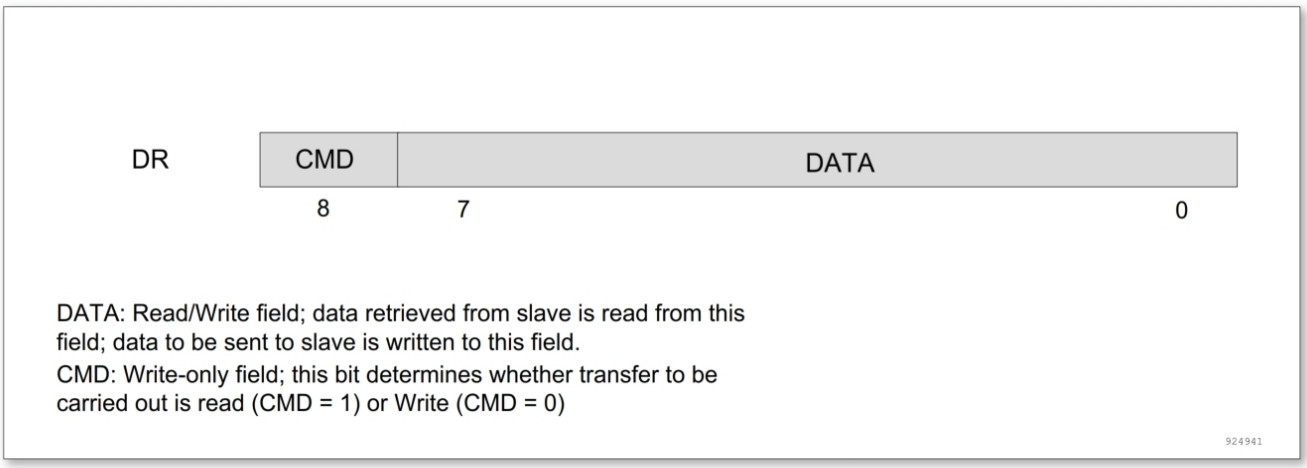


Figure 142. DR register

The following timing diagram illustrates the behavior of I2C module operating in the master transmit mode when Tx FIFO becomes empty.

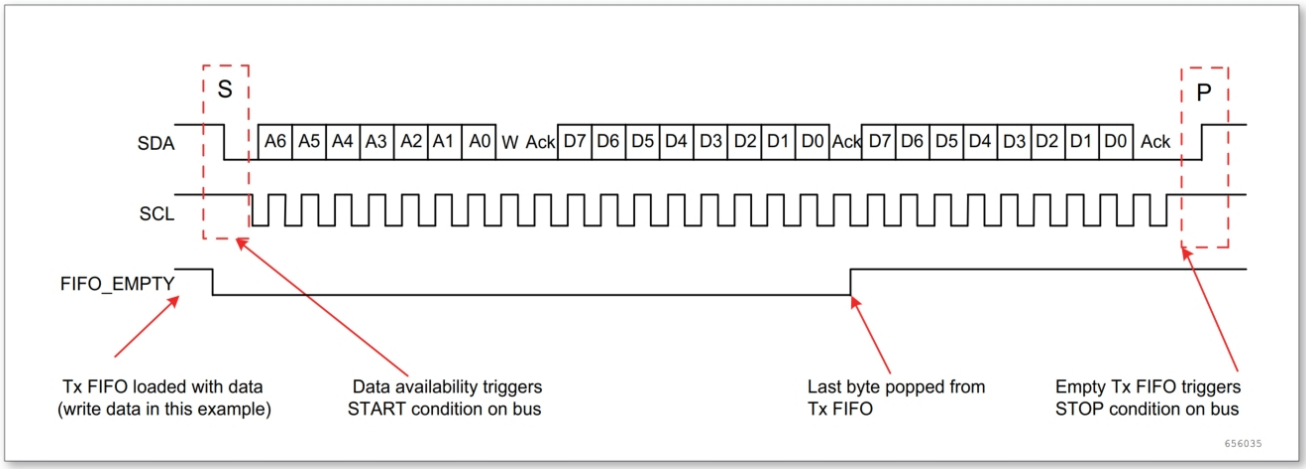


Figure 143. Master transmit - empty Tx FIFO

The following timing diagram illustrates the behavior of I2C module operating in the master receive mode when Tx FIFO becomes empty.

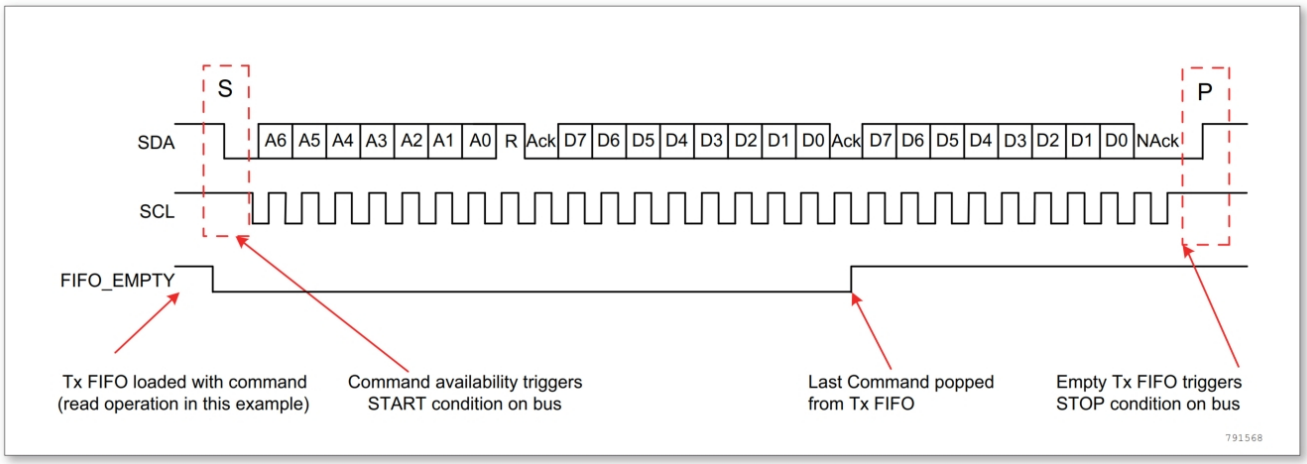


Figure 144. Master receive - empty Tx FIFO

15.3.5 Multiple master arbitration

The I2C bus allows multiple masters to reside on the same bus. If there are several masters on the same I2C-bus, there is an arbitration procedure if they try to take control of the bus at the same time. Through the arbitration, only one master can take control of the bus with intact messages. Once a master has control of the bus, no other master can take control until the first master sends a Stop condition and places the bus in an idle state.

Arbitration takes place on the SDA line, while the SCL line is in the high level. If there are two or more masters trying to send messages to the bus, the master, which transmits a 1 while the other master transmits 0, loses arbitration. The master that lost arbitration can continue to generate clock pulses until the end of the byte transfer. If all masters are addressing the same device, the arbitration could go into the data phase.

Upon detecting that it has lost arbitration, the I2C interface stops generating SCL signals.

The following figure illustrates the timing of two masters arbitrating on the bus.

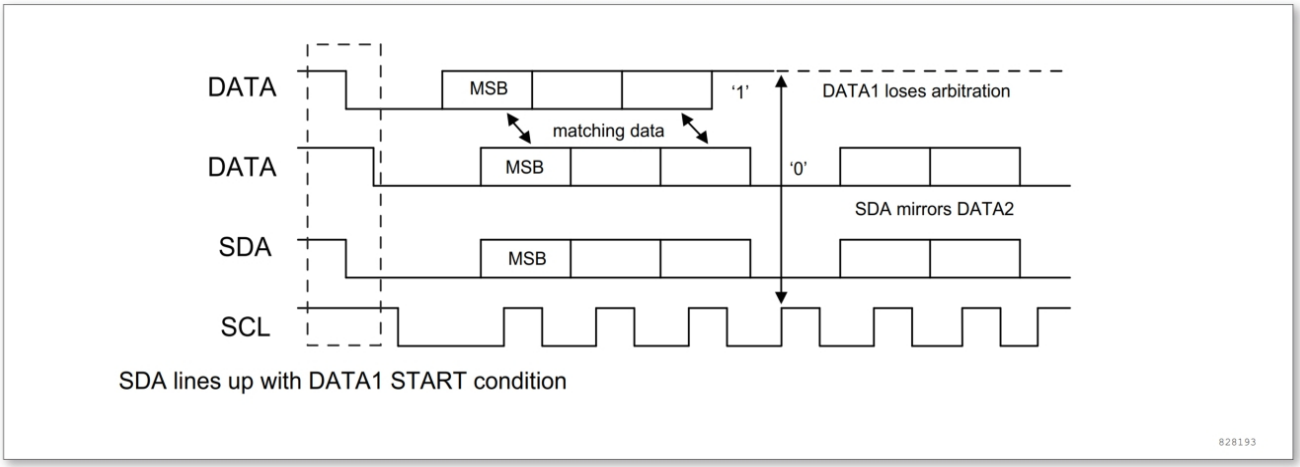


Figure 145. Multiple master arbitration

15.3.6 Clock synchronization

When two or more masters try to transfer information on the bus at the same time, they must arbitrate and synchronize the SCL clock. All masters generate their own clocks to transfer messages. Data is only valid during the high period of clock. Clock synchronization is performed using the wired-AND connection to the SCL signal. When the master transitions the SCL clock to 0, the master starts counting the low time of the SCL clock and transitions the SCL clock to 1 at the beginning of the next clock period. However, if another master is holding the SCL line to 0, then the master goes into a wait state until the SCL clock transitions to 1.

All masters then count off their high time, and the master with the shortest high time transitions the SCL to 0. The masters then counts out their low times and the one with the longest low time forces the other masters into a wait state. Therefore, a synchronized SCL clock is generated, which is illustrated in the following figure.

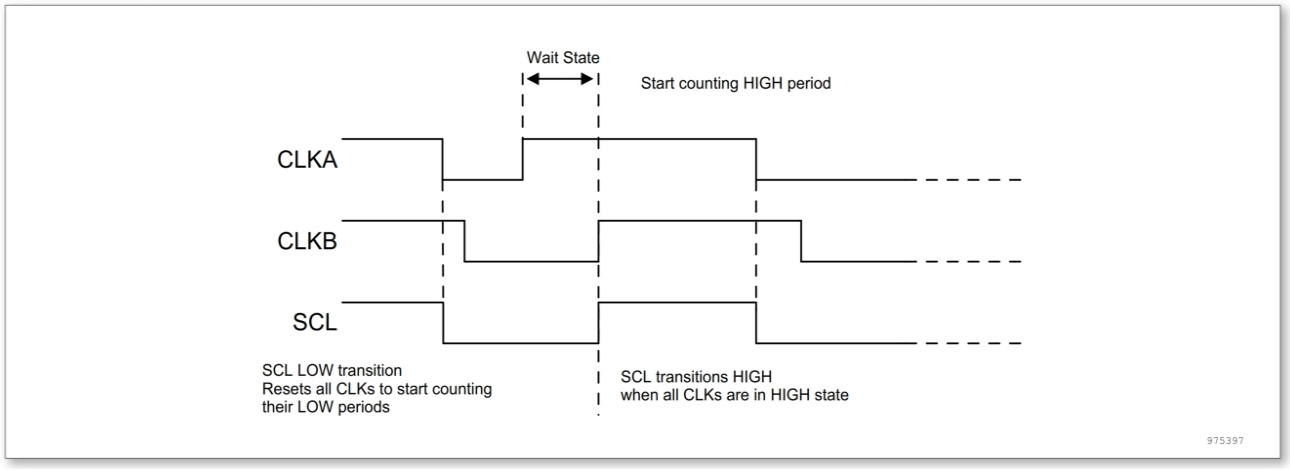


Figure 146. Multiple master clock synchronization

15.3.7 SCL configuration

For I2C level configuration, refer to the following:

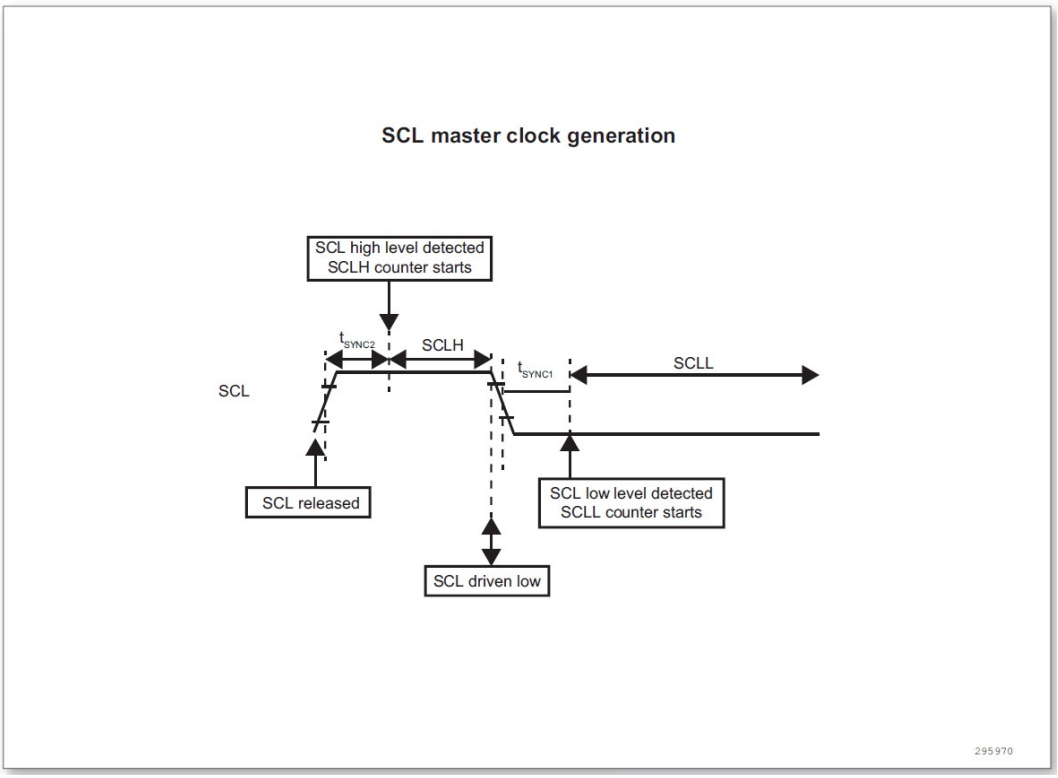


Figure 147. SCL master clock generation

SCLH = (SSHR + 12) x I2CCLK + tsyn1

SCLL = (SSLR + 1) x I2CCLK + tsyn2

Note: tsyn1 is equal to 0 ~ 1 I2CCLK and tsyn2 is equal to 0 ~ 1 I2CCLK

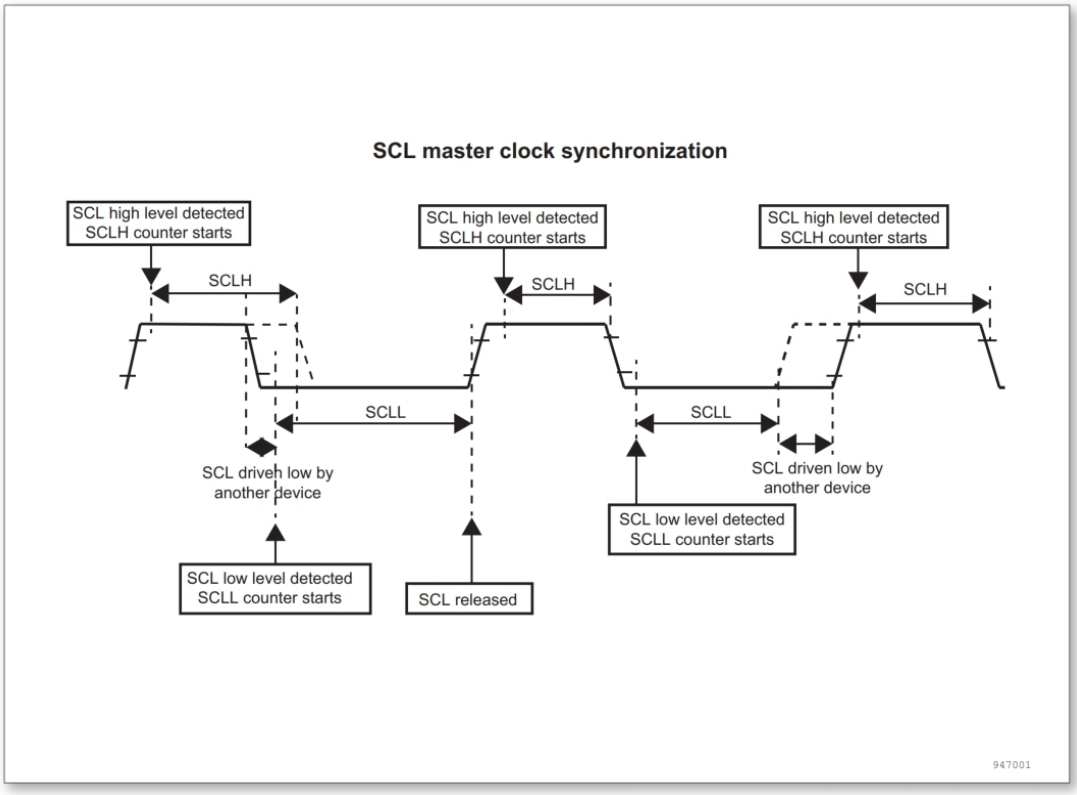


Figure 148. SCL master clock synchronization

SCLH = (FSHR + 12) x I2CCLK + tsyn1

SCLL = (FSLR + 1) x I2CCLK + tsyn2

Note: tsyn1 is equal to 2 ~ 3 I2CCLK and tsyn2 is equal to 2 ~ 3 I2CCLK.

15.4 I2C operation mode

I2C interface can operate in one of the four following modes:

- Slave transmitter mode
- Slave receiver mode
- Master transmitter mode
- Master receiver mode

Note: I2C interface module can only operate in the Master or Slave mode but it cannot operate in both modes at the same time. Therefore, make sure that the bit 6 (DISSLAVE) and bit 0 (MASTER) in the CR register are not set to 0 and 1 (or 1 and 0 respectively).

The I2C functional block diagram is shown as below:

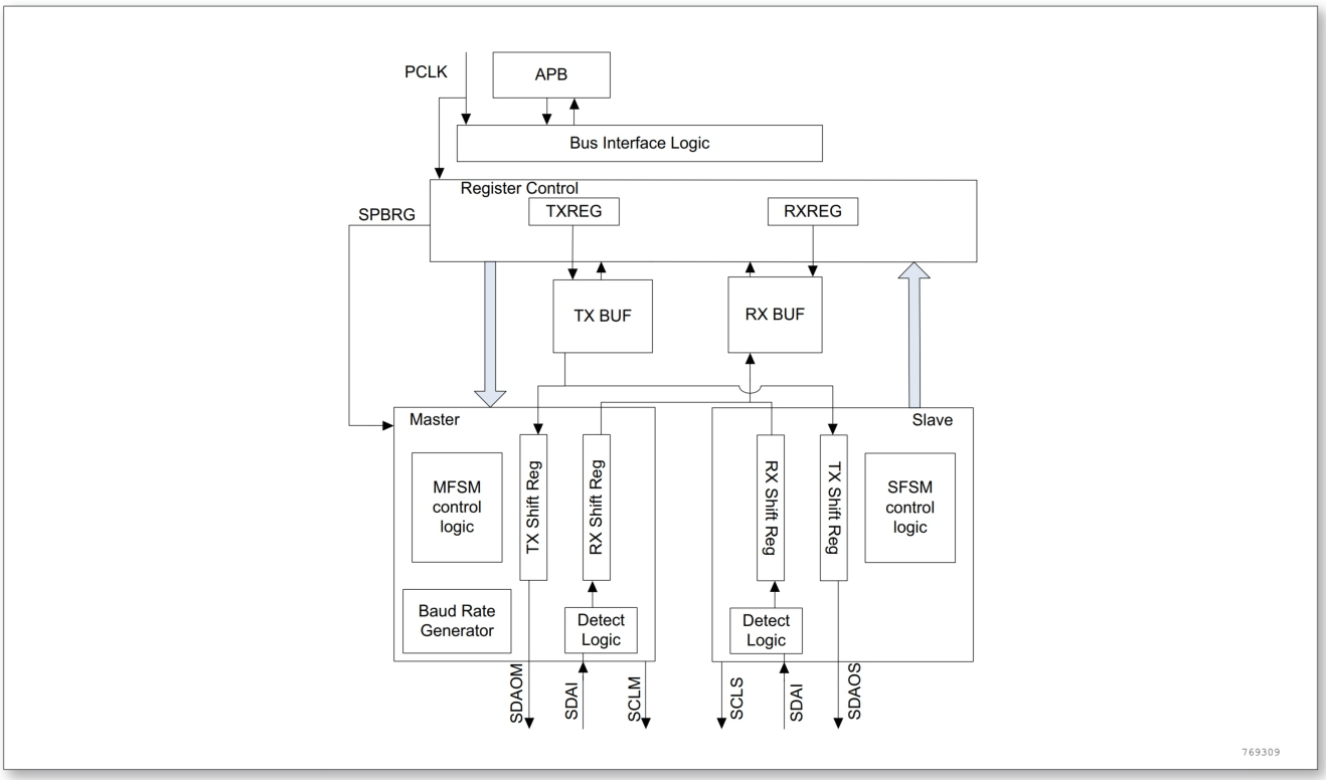


Figure 149. I2C function block diagram

15.4.1 Slave mode

The program flow chart of the Slave mode is described as follows:

Initial configuration

1. Disable I2C by writing a 0 to bit 0 of the ENR register.
2. Initialize the SAR register to set the slave address. This is the address to which the I2C interface responds.
3. Set the CR register to specify which type of addressing is supported (7- or 10-bit by setting bit 3). Write a 0 into bit 6 (DISSLAVE) and a 0 to bit 0 (MASTER) in the CR register.
4. Enable the I2C interface module by writing a 1 to bit 0 of the ENR register.

Slave-transmitter operation for a single byte

When another I2C master device on the bus addresses the I2C interface and requests data, the I2C interface acts as a slave-transmitter and the following steps occur:

1. Another I2C master device initiates an I2C transfer with an sending address that matches the slave address in the SAR register.
2. The I2C interface acknowledges the sent address and recognizes the direction of the transfer to indicate that it is acting as a slave-transmitter.
3. The I2C interface asserts the RD_REQ interrupt (bit 5 of the RAWISR register) and pulls down the SCL line. The bus stays in the wait state until the software responses.

If the RD_REQ interrupt has been masked (register IMR[5] = 0), then it is recommended that CPU does periodic query of the RAWISR register.

1. Setting bit 5 of the RAWISR register to 1 can be treated as the equivalent of the RD_REQ interrupt being asserted.
2. Software must then act to satisfy the I2C transfer requirements.
3. The timing intervals are generally 10 times the SCL clock period. For example, for 400Kbps, the timing interval is 25us.
4. If there is any data remaining in the Tx FIFO before receiving the read request, the I2C interface asserts a TX_ABRT interrupt (RAWISR[6]) to flush the old data from the Tx FIFO.
5. Software writes data to the DR register and writes a 0 in bit 8.
6. Software must clear the RD_REQ and TX_ABRT interrupts of the RAWISR register.
7. The I2C interface relinquishes SCL and transmits the byte.
8. The master controls the I2C bus by issuing a Restart condition or releases the bus by issuing a Stop condition.

Slave-receiver operation for a single byte

When another master device addresses the I2C interface and sends data, the I2C interface acts as a slave-receiver and the following steps occur:

1. Another I2C master device initiates an I2C transfer with an sending address that matches the slave address in the SAR register.
2. The I2C interface acknowledges the sent address and recognizes the direction of the transfer to indicate that it is acting as a slave-receiver.
3. I2C interface receives the transmitted data from master and places it in the receive buffer.
4. I2C interface asserts the RX_FULL interrupt (RAWISR[2]). If the RX_FULL interrupt has been masked (IMR[2] = 0), then it is recommended that the CPU does periodic query of the SR register. Reads of the SR register, with bit 3 (RFNE) set at 1, must then be treated as the equivalent of the RX_FULL interrupt being asserted.
5. Software may read the byte from the DR register (bits 7:0) to get the data that was received.
6. The master controls the I2C bus by issuing a Restart condition or releases the bus by issuing a Stop condition.

Slave-transfer operation for bulk transfers

In the standard I2C protocol, all transactions are single byte transactions and the program responds to a master read request by writing one byte into the slave's Tx FIFO. When a slave (slave-transmitter) receives a read request (RD_REQ) from the master (master-receiver), there should be at least one entry placed into the slave-transmitter's Tx FIFO. The I2C interface module is designed to handle more data in the Tx FIFO so that subsequent read requests can receive that data without raising an interrupt. Ultimately, this eliminates latencies significantly being incurred by the interrupt for data each time.

This mode only occurs when I2C interface is acting as a slave-transmitter. If the master acknowledges the data sent by the slave-transmitter and there is no data in the slave's TX FIFO, the I2C interface will pull down the SCL line of I2C bus until the read request interrupt (RD_REQ) is generated; moreover, it will not relinquish the SCL line unless the data in the TX FIFO is prepared.

If the RX_REQ interrupt has been masked (ISR[5] = 0), then it is recommended that the software does periodic query/reads of the RAWISR register. Reads of

RAWISR[5] that return 1 must be treated as the equivalent of the RX_REQinterrupt.

The RD_REQ interrupt is raised upon a read request, and like interrupts, must be cleared when exiting the interrupt service handling routine (ISR). The ISR allows you to either write 1 byte or more than 1 byte into the TX FIFO. During the transmission of these bytes to the master, if the master acknowledges the last byte, then the slave must raise the RD_REQ interrupt request again because the master is requesting for more data.

If the master is to receive n bytes from the I2C interface but the program wrote a number of bytes larger than n to the Tx FIFO, then when the slave finishes sending the requested n bytes, it clears the Tx FIFO and ignores any excess bytes.

15.4.2 Master mode

Initial configuration

1. Disable the I2C interface by setting ENR[0] = 0.
2. Set the I2C operation speed mode (Stand mode or Fast mode) by configuring bit 2:1 in the CR register. Meanwhile, make sure bit 6(DISSLAVE) = 1 and bit 0(MASTER) = 1.
3. Write to the TAR register the address of the I2C device. Set this register so that a general call address or a Start Byte command can be configured.
4. Enable the I2C interface by writing a 1 in ENR[0].
5. Now write the transfer direction and data to be sent to the DR register. If the DR register is written before the I2C interface is enabled, the data and commands are lost as the buffers are kept cleared when the I2C interface is not enabled.

Through these steps, I2C interface will generate a Start condition and send the address byte data to the I2C bus.

Master transmit and Master receive

The I2C interface supports switching back and forth between reading and writing dynamically. When transmitting data, write the data to be written to the lower byte of the I2C Rx/Tx data buffer and Command register (DR). The CMD bit should be written to 0 for write operations. Subsequently, a read command does not require the configuration of lower byte of the DR register, and a 1 should be written to the CMD bit. If the sent FIFO is empty, the I2C module pulls down SCL until the next command is written into the sent FIFO.

Program flow chart

The following flow chart shows a programming model when I2C interface is set as a master:

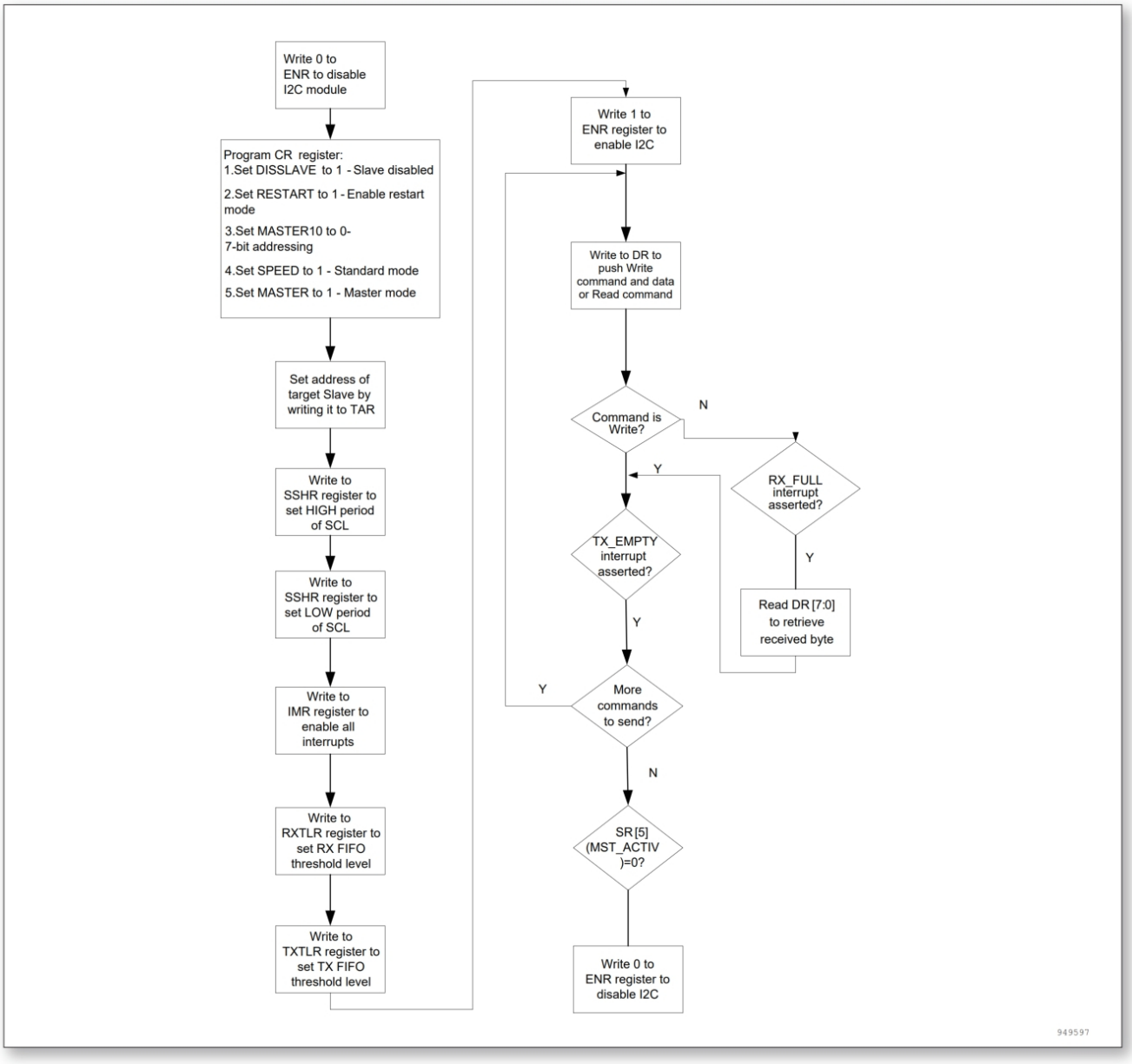


Figure 150. I2C interface master flow chart

15.4.3 I2C abort transfer

The ABRT control bit of the ENR register allows the software to relinquish the I2C bus before completing the issued transfer commands from the Tx FIFO.

In response to an ABORT request, the I2C module issues the Stop condition over the I2C bus, alongside Tx FIFO flush. Aborting the transfer is allowed only in master mode of operation.

Procedure

1. Stop filling the Tx FIFO (DR) with new commands.
2. Set ABRT bit of the ENR register to 1.
3. Wait for the TX_ABRT interrupt.

15.5 I2C interrupts

The table below gives the list of I2C interrupt bits, as well as the way to set and clear them. Some bits are set by hardware and cleared by software, others are set and cleared by hardware.

Table 49. Setting and clearing the interrupt bits

Interrupt bit	Set by hardware/Cleared by software	Set and cleared by hardware
GEN_CALL	√	x
START_DET	√	x
STOP_DET	√	x
ACTIVITY	√	x
RX_DONE	√	x
TX_ABRT	√	x
RD_REQ	√	x
TX_EMPTY	x	√
TX_OVER	√	x
RX_FULL	x	√
RX_OVER	√	x
RX_UNDER	√	x

The figure below describes the interrupt bit is set by hardware and cleared by software in the interrupt register.

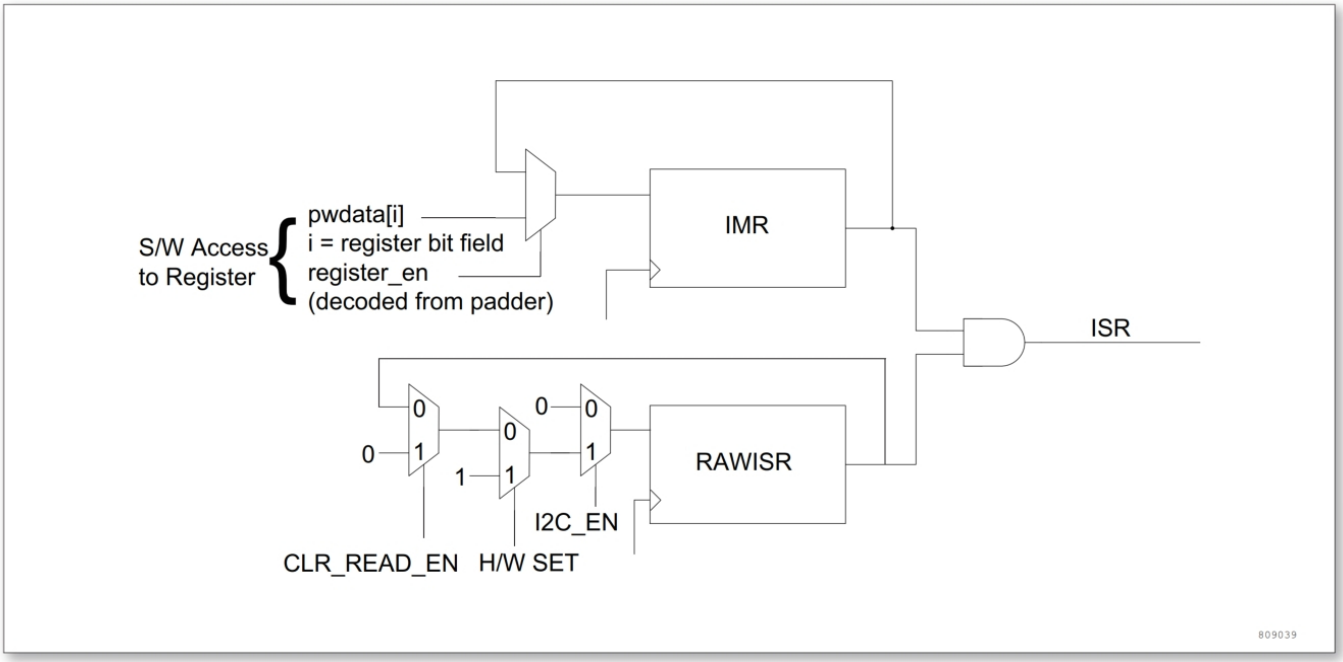


Figure 151. Interrupt mechanism

15.6 I2C register description

Table 50. Overview of I2C register description

Offset	Acronym	Register Name	Reset	Section
0x00	I2C_CR	I2C control register	0x0000007F	Section 15.6.1
0x04	I2C_TAR	I2C target address register	0x00000055	Section 15.6.2
0x08	I2C_SAR	I2C slave address register	0x00000055	Section 15.6.3
0x10	I2C_DR	I2C data command register	0x00000001	Section 15.6.4
0x14	I2C_SSHR	Standard mode I2C clock SCL high count register	0x00000190	Section 15.6.5
0x18	I2C_SSLR	Standard mode I2C clock SCL low count register	0x000001D6	Section 15.6.6
0x1C	I2C_FSHR	Fast mode I2C clock SCL high count register	0x0000003C	Section 15.6.7
0x20	I2C_FSLR	Fast mode I2C clock SCL low count register	0x00000082	Section 15.6.8
0x2C	I2C_ISR	I2C interrupt status register	0x00000000	Section 15.6.9
0x30	I2C_IMR	I2C interrupt mask register	0x000008FF	Section 15.6.10
0x34	I2C_RAWISR	I2C RAW interrupt status register	0x00000000	Section 15.6.11
0x38	I2C_RXTLR	I2C receive threshold register	0x00000000	Section 15.6.12
0x3C	I2C_TXTLR	I2C transmit threshold register	0x00000000	Section 15.6.13
0x40	I2C_ICR	I2C combined and independent interrupt clear register	0x00000000	Section 15.6.14
0x44	I2C_RX_UNDER	I2C RX_UNDER interrupt clear register	0x00000000	Section 15.6.15

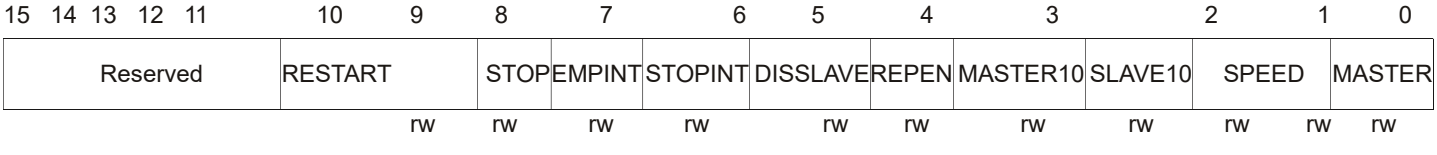
UM_MM32F0010_Ver1.00

Offset	Acronym	Register Name	Reset	Section
0x48	I2C_RX_OVER	I2C RX_OVER interrupt clear register	0x00000000	Section 15.6.16
0x4C	I2C_TX_OVER	I2C TX_OVER interrupt clear register	0x00000000	Section 15.6.17
0x50	I2C_RD_REQ	I2C RD_REQ interrupt clear register	0x00000000	Section 15.6.18
0x54	I2C_TX_ABRT	I2C TX_ABRT interrupt clear register	0x00000000	Section 15.6.19
0x58	I2C_RX_DONE	I2C RX_DONE interrupt clear register	0x00000000	Section 15.6.20
0x5C	I2C_ACTIV	I2C ACTIVITY interrupt clear register	0x00000000	Section 15.6.21
0x60	I2C_STOP	I2C STOP_DET interrupt clear register	0x00000000	Section 15.6.22
0x64	I2C_START	I2C START_DET interrupt clear register	0x00000000	Section 15.6.23
0x68	I2C_GC	I2C GEN_CALL interrupt clear register	0x00000000	Section 15.6.24
0x6C	I2C_ENR	I2C enable register	0x00000000	Section 15.6.25
0x70	I2C_SR	I2C status register	0x00000006	Section 15.6.26
0x74	I2C_TXFLR	I2C transmit FIFO level register	0x00000000	Section 15.6.27
0x78	I2C_RXFLR	I2C receive FIFO level register	0x00000000	Section 15.6.28
0x7C	I2C_HOLD	I2C SDA hold time register	0x00000001	Section 15.6.29
0x94	I2C_SETUP	I2C SDA setup time register	0x00000064	Section 15.6.30
0x98	I2C_GCR	I2C general call ACK register	0x00000001	Section 15.6.31
0xB0	I2C_SLVMASK	I2C slave address mask register	0x000003FF	Section 15.6.32
0xB4	I2C_SLVRCVADDR	I2C slave receive address register	0x00000000	Section 15.6.33

15.6.1 I2C control register (I2C_CR)

Address offset: 0x00

Reset value: 0x007F



Bit	Field	Type	Reset	Description
15:11	Reserved			always read as 0.
10	RESTART	rw	0x00	This bit controls whether a RESTART is issued before transmission or reception. It is valid only when the IC_EMPTYFIFO_HOLD_MASTER_EN is set to '1'. 1: If RESTART is '1', a RESTART signal is issued before the data is sent/received (according to the value of CMD), regardless of whether or not the transfer direction is changing from the previous command. If RESTART is '0', a START is issued instead, followed by a STOP signal. 0: If RESTART is 1, a RESTART is issued only if the transfer direction is changing from the previous command. If RESTART is '0', a START is issued instead, followed by a STOP signal.
9	STOP		0x00	STOP: This bit controls whether a STOP is issued after transmission or reception. It is valid only when the IC_EMPTYFIFO_HOLD_MASTER_EN is set to '1'. 1: STOP is issued after this byte, regardless of whether or not the Tx FIFO is empty. If the Tx FIFO is not empty, the master immediately tries to start a new transfer by issuing a START and arbitrating for the bus. 0: STOP is not issued after this byte, regardless of whether or not the Tx FIFO is empty. The master continues the current transfer by sending/receiving data bytes according to the value of the CMD bit. If the Tx FIFO is empty, the master holds the SCL low and stalls the bus until Tx FIFO receives new data.
8	EMPINT	rw	0x00	This bit controls whether a TX_EMPTY interrupt is generated. For details, refer to the RAWISR register.
7	STOPINT	rw	0x00	This bit controls whether a STOP interrupt is generated in the Slave mode. 1: STOP interrupt is generated only when the address is matched. 0: STOP interrupt is generated regardless of whether or not the address is matched. This bit is only applicable in the Slave mode. Note: During general call addressing, if this bit is set, the slave doesn't generate a STOP interrupt. The STOP interrupt is generated only when the sent address is matched with the slave address.

Bit	Field	Type	Reset	Description
6	DISSLAVE	rw	0x01	This bit controls whether I2C has its slave disabled. 0: slave enabled 1: slave disabled
5	REPEN	rw	0x01	This bit determines whether RESTART conditions may be sent when acting as a master. 0: disabled 1: enabled When RESTART is disabled, I2C interface acting as a master is prohibited from performing the following functions: Send a Start Byte Change transfer direction in the combined format Read operation with a 10-bit address Replace Restart condition with a Stop and a subsequent Start condition. If the above operations are performed, it will result in setting 6(TX_ABRT) bit of RAWISR register to 1.
4	MASTER10	rw	0x01	Address mode when I2C acts as a master 0: 7-bit address mode 1: 10-bit address mode
3	SLAVE10	rw	0x01	When acting as a slave, this bit controls whether the I2C responds to 7- or 10-bit addresses 0: 7-bit addressing. The I2C interface ignores transactions that involve 10-bit addressing; for 7-bit addressing, only the lower 7 bits of the SAR register are compared. 1: 10-bit addressing. The I2C responds to only 10-bit addressing transfers that match the full 10 bits of the SAR register.
2:1	SPEED	rw	0x03	These bits control at which speed the I2C operates. The setting is valid only when the I2C operates in the Master mode. 1: Standard mode (0 ~ 100Kbps) 2: Fast mode (400Kbps)
0	MASTER	rw	0x01	This bit controls whether the I2C master is enabled. 0: master disabled 1: master enabled

DISSLAVE(bit 6) and MASTER(bit 0) are set according to the table below:

Table 51. DISSLAVE(bit 6) and MASTER(bit 0) settings

DISSLAVE CR[6]	MASTER CR[0]	Status
0	0	Slave device
0	1	Wrong configuration
1	0	Wrong configuration
1	1	Master device

15.6.2 I2C target address register (I2C_TAR)

Address offset: 0x04
Reset value: 0x0055

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

Bit	Field	Type	Reset	Description
15:12	Reserved			always read as 0.
11	SPECIAL	rw	0x00	This bit indicates whether software performs a special command (General Call or Start Byte command). 0: Ignore bit 10(GC) and use ADDR bit normally. 1: Perform special I2C command as specified in GC bit.
10	GC	rw	0x00	If bit 11(SPECIAL) is set to 1, then this bit indicates whether a General Call or Start byte command is to be performed by the I2C. 0: General Call address After issuing a General Call, only writes may be performed. The I2C interface remains working in General Call mode until the SPECIAL(bit 11) is cleared. 1: START BYTE command
9:0	ADDR	rw	0x55	This is the target address for any master transaction. When transmitting a General Call, these bits are ignored. To generate a START BYTE command, the CPU needs to write only once into these bits.

15.6.3 I2C Slave address register (I2C_SAR)

Address offset: 0x08
Reset value: 0x0055

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

Bit	Field	Type	Reset	Description
15:10	Reserved			always read as 0
9:0	ADDR	rw	0x55	The bit holds the slave address when the I2C interface is operating as a slave. For 7-bit addressing, only ADDR [6:0] is valid.

15.6.4 I2C Data command register (I2C_DR)

Address offset: 0x10
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:9	Reserved			always read as 0.
8	CMD	w	0x00	This bit controls whether a read or a write is performed in the Master mode. 1: Read 0: Write When a command is entered in the TX FIFO, this bit distinguishes the write and read commands. In slave-receiver mode, write operation to this bit is ignored. In slave-transmitter mode, a “0” indicates that data of DR register is to be transmitted.
7:0	DAT	rw	0x00	This register contains the data to be transmitted or received on the I2C bus.

15.6.5 Standard mode I2C clock SCL high count register (I2C_SSHR)

Address offset: 0x14

Reset value: 0x0190

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

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x0190	This register sets the SCL clock high-period for I2C interface in the Standard mode. Note: This register is programmed to a value between 0 and 65525, because I2C interface uses a 16-bit counter to flag an I2C bus idle condition when this counter reaches a value of SSHR + 10.

15.6.6 Standard mode I2C clock SCL low count register (I2C_SSLR)

Address offset: 0x18

Reset value: 0x01D6

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

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x01D6	This register sets the minimum SCL clock low-period to 8 for I2C interface in the Standard mode.

15.6.7 Fast mode I2C clock SCL high count register (I2C_FSHR)

Address offset: 0x1C

Reset value: 0x003C

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

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x003C	This register sets the SCL clock high-period for I2C interface in the Fast mode. When I2Coperates in the Standard mode, this register is read-only and returns 0. The minimum value is 6.

15.6.8 Fast mode I2C clock SCL low count register (I2C_FSLR)

Address offset: 0x20

Reset value: 0x0082

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

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x0082	This register sets the SCL clock low-period for I2C interface in the Fast mode. When I2Coperates in the Standard mode, this register is read-only and returns 0. The minimum value is 8.

15.6.9 I2C interrupt status register (I2C_ISR)

Address offset: 0x2C
Reset value: 0x0000

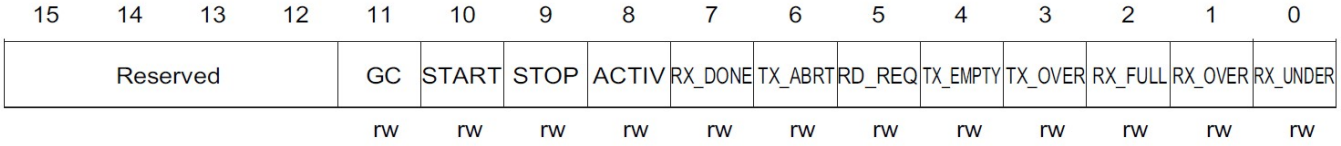
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				GC	START	STOP	ACTIV	RX_DONE	TX_ABRT	RD_REQ	TX_EMPTY	TX_OVER	RX_FULL	RX_OVER	RX_UNDER
				r	r	r	r	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
15:12	Reserved			always read as 0
11:0	ISR	r	0x0000	Refer to RAWISR register for detailed descriptions of these bits.

15.6.10 I2C interrupt mask register (I2C_IMR)

Address offset: 0x30

Reset value: 0x08FF



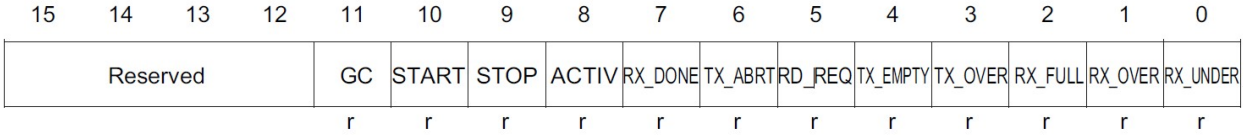
Bit	Field	Type	Reset	Description
15:12	Reserved			always read as 0
11:0	IMR	rw	0x08FF	These bits mask their corresponding ISR bits.

15.6.11 I2C RAW interrupt status register (I2C_RAWISR)

Address offset: 0x34

Reset value: 0x0000

Unlike the ISR register, RAWISR is not masked.



Bit	Field	Type	Reset	Description
15:12	Reserved			always read as 0.
11	GC	r	0x00	General call Set when a General Call address is received. It is cleared either by disabling I2C interface or when the CPU reads the GC register. I2C stores the received data in the Rx buffer.
10	START	r	0x00	Start condition detection This bit is set to 1 once it detects that a START or RESTART condition has occurred on the I2C interface regardless of whether I2C interface is operating in Slave or Master mode.

Bit	Field	Type	Reset	Description
9	STOP	r	0x00	Stop condition detection The status of this bit differs based on the STOPINT status in the CR register. When STOPINT = 0 This bit is set to 1 once it detects that a STOP condition has occurred on the I2C interface regardless of whether I2C interface is operating in Slave or Master mode. In Slave mode, a STOP interrupt is generated irrespective of whether the address is matched or not. When STOPINT = 1 In Master mode (MASTER = 1), this bit indicates whether a STOP condition has occurred on the I2C interface. In Slave mode (MASTER = 0), a STOP interrupt is generated only if the slave address is matched successfully.
8	ACTIV	r	0x00	I2C interface activity This bit captures I2C module activity and stays set until it is cleared by four ways: Disabling the I2C interface Reading the ACTIV register Reading the ICR register System reset Once this bit is set, it stays set unless one of the four methods is used to clear it. Even if the I2C is idle, this bit remains set until cleared.
7	RX_DONE	r	0x00	Slave transmit done When the I2C is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done.
6	TX_ABRT	r	0x00	Transmit abort This bit is set to 1 when I2C interface, as an I2C transmitter, is unable to complete the data transfer in the FIFO. Note:) The I2C flushes the Tx FIFO and Rx FIFO when there is a transmit abort. The Tx FIFO remains in this flushed state until the TX_ABRT register is read. Once this read is performed, the Tx FIFO is then ready to accept new data from the APB bus.

Bit	Field	Type	Reset	Description
5	RD_REQ	r	0x00	Read request This bit is set to 1 when I2C is acting as a slave and another master is attempting to read data from the I2C interface. The I2C interface holds the bus in a wait state (SCL = 0) until this interrupt is serviced, which means that the I2C interface, as a slave, has been addressed by another master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the DR register. This bit is set to 0 just after the processor reads the RD_REQ register.
4	TX_EMPTY	r	0x00	Transmit buffer empty The status of this bit differs based on the EMPINT status in the CR register. If EMPINT = 0, this bit is set to 1 when the transmit buffer is at or below the threshold value If EMPINT = 1, this bit is set to 1 when the transmit buffer is at or below the threshold value and the most recent data transmission from the internal shift register is completed. It is automatically cleared by hardware when the transmit buffer is not empty.
3	TX_OVER	r	0x00	Transmit buffer over When the transmit buffer is full, the processor writes new data and causes overflow, which sets the bit to 1.
2	RX_FULL	r	0x00	Receive buffer not empty Set when the receive buffer is not empty. It is automatically cleared by hardware when the receive buffer is empty.
1	RX_OVER	r	0x00	Receive buffer over Set if the receive buffer is full and an additional byte is received. The I2C interface acknowledges this, but new data will be lost.
0	RX_UNDER	r	0x00	Receive buffer under Set to 1 if the processor attempts to read the DR register when RX FIFO is empty.

15.6.12 I2C receive threshold register (I2C_RXTLR)

Address offset: 0x38

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:8	Reserved			always read as 0
7:0	TL	rw	0x00	Receive FIFO threshold level This bit controls whether the RX_FULL interrupt is triggered.

15.6.13 I2C transmit threshold register (I2C_TXTLR)

Address offset: 0x3C
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:8	Reserved			always read as 0
7:0	TL	rw	0x00	Transmit FIFO threshold level This bit controls whether the TX_EMPTY interrupt is triggered.

15.6.14 I2C combined and independent interrupt clear register (I2C_ICR)

Address offset: 0x40
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0.
0	ICR	rc_r	0x00	Reading this register clears all combined and independent interrupts. This bit only clears software clearable interrupts, instead of hardware clearable interrupts.

15.6.15 I2C RX_UNDER interrupt clear register (I2C_RX_UNDER)

Address offset: 0x44
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0
0	RX_UNDER	rc_r	0x00	Reading this register clears the RX_UNDER interrupt (RAWISR[0]).

15.6.16 I2C RX_OVER interrupt clear register (I2C_RX_OVER)

Address offset: 0x48
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0
0	RX_OVER	rc_r	0x00	Reading this register clears the RX_OVER interrupt (RAWISR[1]).

15.6.17 I2C TX_OVER interrupt clear register (I2C_TX_OVER)

Address offset: 0x4C
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0
0	TX_OVER	rc_r	0x00	Reading this register clears the TX_OVER interrupt (RAW_ISR[3]).

15.6.18 I2C RD_REQ interrupt clear register (I2C_RD_REQ)

Address offset: 0x50

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0
0	RD_REQ	rc_r	0x00	Reading this register clears the RD_REQ interrupt (RAW_ISR[5]).

15.6.19 I2C TX_ABRT interrupt clear register (I2C_TX_ABRT)

Address offset: 0x54
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0.
0	TX_ABRT	rc_r	0x00	Reading this register clears the TX_ABRT interrupt (RAWISR6]). This also releases the TX FIFO from the flushed/reset state, allowing more writes to the TX FIFO.

15.6.20 I2C RX_DONE interrupt clear register (I2C_RX_DONE)

Address offset: 0x58
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0
0	RX_DONE	rc_r	0x00	Reading this register clears the RX_DONE interrupt (RAWISR[7]).

15.6.21 I2C ACTIVITY interrupt clear register (I2C_ACTIV)

Address offset: 0x5C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0.
0	ACTIV	rc_r	0x00	Reading this register clears the ACTIV interrupt (RAW-ISR[8]) if the I2C bus is not active anymore. If the I2C is still active, the ACTIV interrupt bit continues to be set. It is automatically cleared by hardware if the I2C module is disabled and if there is no further activity on the I2C bus. By reading this register, you get status of the ACTIV(bit 8) in the RAWISR register.

15.6.22 I2C STOP_DET interrupt clear register (I2C_STOP)

Address offset: 0x60
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0
0	STOP	rc_r	0x00	Reading this register clears the STOP interrupt (RAWISR[9]).

15.6.23 I2C START_DET interrupt clear register (I2C_START)

Address offset: 0x64
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0
0	START	rc_r	0x00	Reading this register clears the START interrupt (RAWISR[10]).

15.6.24 I2C GEN_CALL interrupt clear register (I2C_GC)

Address offset: 0x68
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0
0	GC	rc_r	0x00	Reading this register clears the GC interrupt (RAWISR[11]).

15.6.25 I2C enable register (I2C_ENR)

Address offset: 0x6C
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:2	Reserved			always read as 0.
1	ABORT	rw	0x00	I2C transfer abort 0: ABORT not initiated or ABORT done 1: ABORT operation in progress The software can abort the I2C transfer by setting this bit when the I2C module acts as a master. The software cannot clear the ABORT bit once set. In response to an ABORT, the I2C controller issues a STOP and flushes the Tx FIFO after completing the current transfer, then sets the TX_ABRT interrupt after the abort operation. The ABORT bit is cleared automatically after the abort operation.
0	ENABLE	rw	0x00	I2C mode enable 0: I2C module disabled (TX and RX FIFOs are held in an erased state) 1: I2C module enabled

15.6.26 I2C status register (I2C_SR)

Address offset: 0x70

Reset value: 0x0006

This is a read-only register used to indicate the current transfer status and FIFO status. None of the bits in this register request an interrupt.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									SLV_ACTIV	MST_ACTIV	RFF	RFNE	TFE	TFNE	ACTIV
									r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
15:7	Reserved			always read as 0.
6	SLV _ACTIV	r	0x00	Slave FSM activity status 0: Slave FSM is in IDLE state so the Slave part of I2C is not active 1: Slave FSM is not in IDLE state so the Slave part of I2C is active
5	MST _ACTIV	r	0x00	Master FSM activity status 0: Master FSM is in IDLE state so the Master part of I2C is not active 1: Master FSM is not in IDLE state so the Master part of I2C is active
4	RFF	r	0x00	Receive FIFO completely full 0: Receive FIFO is not full 1: Receive FIFO is full

Bit	Field	Type	Reset	Description
3	RFNE	r	0x00	Receive FIFO not empty 0: Receive FIFO is empty 1: Receive FIFO is not empty
2	TFE	r	0x01	Transmit FIFO completely empty 0: Transmit FIFO is not empty 1: Transmit FIFO is empty
1	TFNF	r	0x01	Transmit FIFO not full 0: Transmit FIFO is full 1: Transmit FIFO is not full
0	ACTIV	r	0x00	I2C activity status This bit is an ORed result of MST_ACTIV bit and SLV_ACTIV bit.

15.6.27 I2C transmit FIFO level register (I2C_TXFLR)

Address offset: 0x74
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:2	Reserved			always read as 0
1:0	CNT	r	0x00	Contains the number of valid data entries in the transmit FIFO (0~2)

15.6.28 I2C receive FIFO level register (I2C_RXFLR)

Address offset: 0x78
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:2	Reserved			always read as 0
1:0	CNT	r	0x00	Contains the number of valid data entries in the receive FIFO (0~2)

15.6.29 I2C SDA hold time register (I2C_HOLD)

Address offset: 0x7C
Reset value: 0x0000 0001

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

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

Bit	Field	Type	Reset	Description
31:24	Reserved			always read as 0
23:16	RX_HOLD	rw	0x00	Sets the SDA hold time in units of APB1 system clock period, when I2C device acts as a receiver.
15:0	TX_HOLD	rw	0x01	Sets the SDA hold time in units of APB1 system clock period, when I2C device acts as a transmitter.

15.6.30 I2C SDA setup time register (I2C_SETUP)

Address offset: 0x94
Reset value: 0x0064

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

Bit	Field	Type	Reset	Description
15:8	Reserved			always read as 0.
7:0	CNT	rw	0x64	SDA setup time It is recommended that if the required delay is 1000ns, then for an APB1 clock frequency of 10MHz, this register should be programmed to a value of 11. The minimum value of this register is 2.

15.6.31 I2C general call ACK register (I2C_GCR)

Address offset: 0x98
Reset value: 0x0001

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

Bit	Field	Type	Reset	Description
15:1	Reserved			always read as 0.
0	GC	rw	0x01	ACK general call 1: I2C responds with an ACK when it receives a General Call. 0: I2C neither responds with an ACK nor generates an interrupt, when it receives a General Call.

15.6.32 I2C slave address mask register (I2C_SLVMASK)

Address offset: 0xB0
Reset value: 0x03FF

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

Bit	Field	Type	Reset	Description
15:10	Reserved			always read as 0.
9:0	MASK	rw	0x03FF	Slave address mask 1: a corresponding bit in the SAR register requires comparison 0: a corresponding bit in the SAR register is masked, not requiring comparison

15.6.33 I2C slave receive address register (I2C_SLVRCVADDR)

Address offset: 0xB4
Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15:10	Reserved			always read as 0
9:0	ADDR	r	0x0000	Address actually received by the slave

16 Universal Asynchronous Receiver Transmitter (UART)

Universal Asynchronous Receiver Transmitter (UART)

16.1 UART introduction

The universal asynchronous receiver transmitter (UART) offers a flexible means of full-duplex data exchange with external equipment requiring an industry standard NRZ asynchronous serial data format. The UART offers a very wide range of baud rates using a fractional baud rate generator. It supports synchronous one-way communication and half-duplex single wire communication.

16.2 UART main features

- Support asynchronous RS-232S protocol and comply with industrial standard 16550
- Full duplex, asynchronous communications
- Fractional baud rate generator systems
- Common programmable transmit and receive baud rates
- Separate transmit and 16×8 receive FIFO registers
- Embedded 1-byte Tx and Rx FIFOs
- Data transmission and reception in the way of least significant bit first
- A start bit followed by data bits, with length of 5, 6, 7, or 8 bits, plus stop bits in the end In addition, an optional parity bit (with odd or even select) can be put between the data bits and stop bits.
- The bit 9 can be configured for synchronization frame.
- Generation and detection of hardware odd or even verification
- Generation and detection of line breaks
- Generation and detection of idle lines
- BRK transmission and reception based on LIN protocol
- Support exchange of Tx and Rx signals and inversion of Tx and Rx
- Support baud rate adaption function
- Support following interrupt sources:
  - Tx BUFFER empty
  - Rx data valid
  - Rx buffer overrun
  - Framing error
  - Parity error
  - Rx break frame
  - Tx shift register complete
  - Tx break frame complete
  - Rx synchronous frame
  - Idle frame complete

- Automatic end of baud rate
- Automatic baud rate error

16.3 UART functional description

Any UART bidirectional communication requires a minimum of two pins: Receive Data In (RX) and Transmit Data Out (TX).

RX: Receive Data Input is the serial data input. Oversampling techniques are used for data recovery by discriminating between incoming data and noise.

TX: Transmit Data Output. When the transmitter is disabled, the output pin returns to its IO port configuration. When the transmitter is enabled and nothing is to be transmitted, the TX pin is at high level.

- The bus is in the idle state prior to transmission or reception.
- A start bit
- A data word (5, 6, 7 or 8 bits), least significant bit first
- 0.5,1, 1.5, 2 stop bits indicating that the data frame is complete
- This interface uses a fractional baud rate generator - with a 16-bit integer and 4-bit fraction representation.
- By enabling the SWAP bit in the UART_GCR register, the interface swaps signals from Tx and Rx ends.
- By enabling the RX_TOG/ TX_TOG bit in the UART_GCR register, the interface inverts signals from Tx/Rx ends.

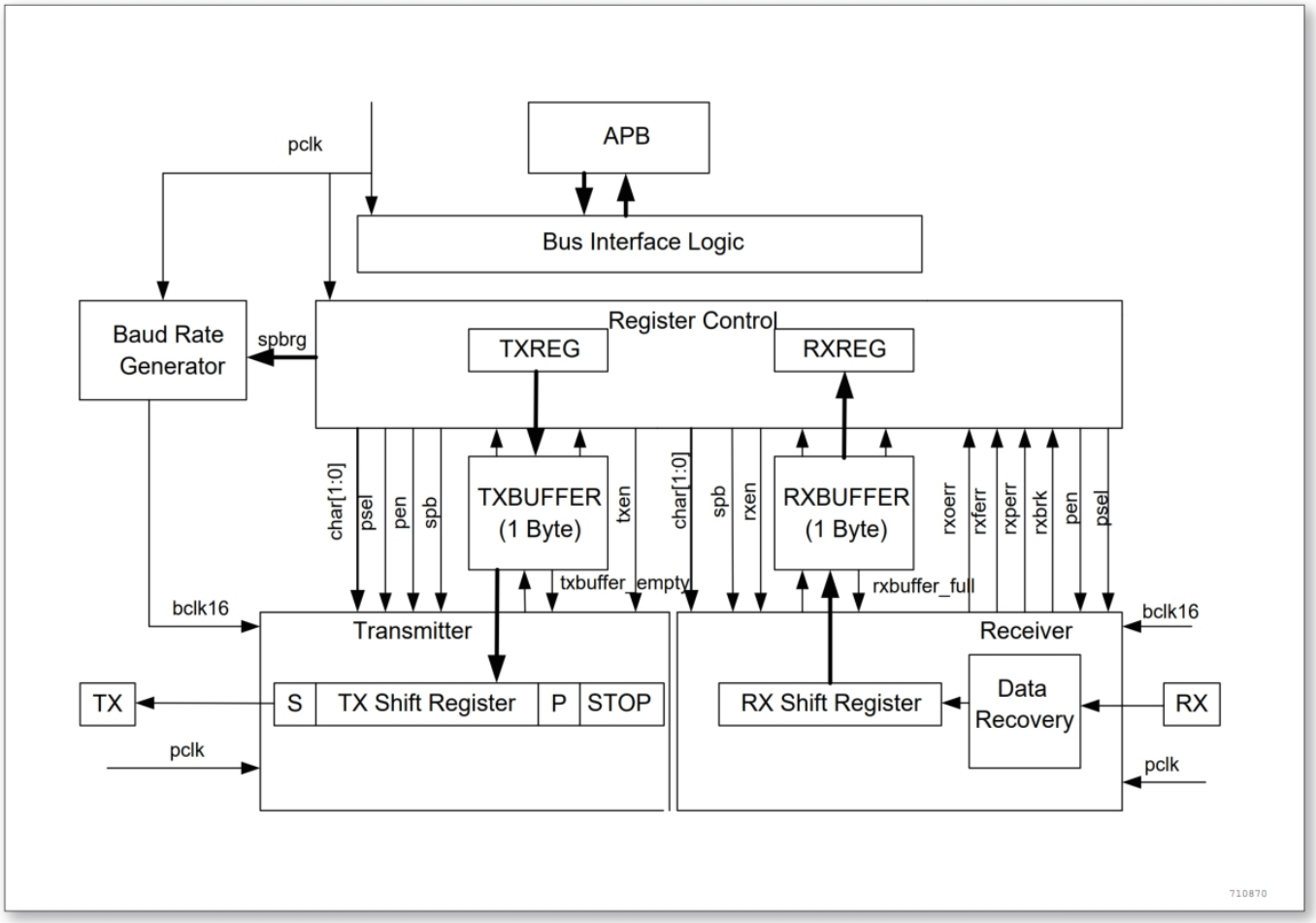


Figure 152. UART block diagram

16.3.1 UART character description

Word length may be selected as being 5 to 8 bits by programming the CHAR bit in the UART_CCR register. The TX pin is in low state during the start bit. It is in high state during the stop bit.

An Idle character is interpreted as an entire data frame of “1”s followed by the start bit of the next frame which contains data (The number of “1” bits will include the number of stop bits).

A Break character is interpreted on receiving “0”s for a frame period (including the period of stop bits which are also “0”s). At the end of the break frame, the transmitter inserts 1 stop bit (“1”) again to acknowledge the start bit.

Transmission and reception are driven by a common baud rate generator, the clock for each is generated when the enable bit is set respectively for the transmitter and receiver.

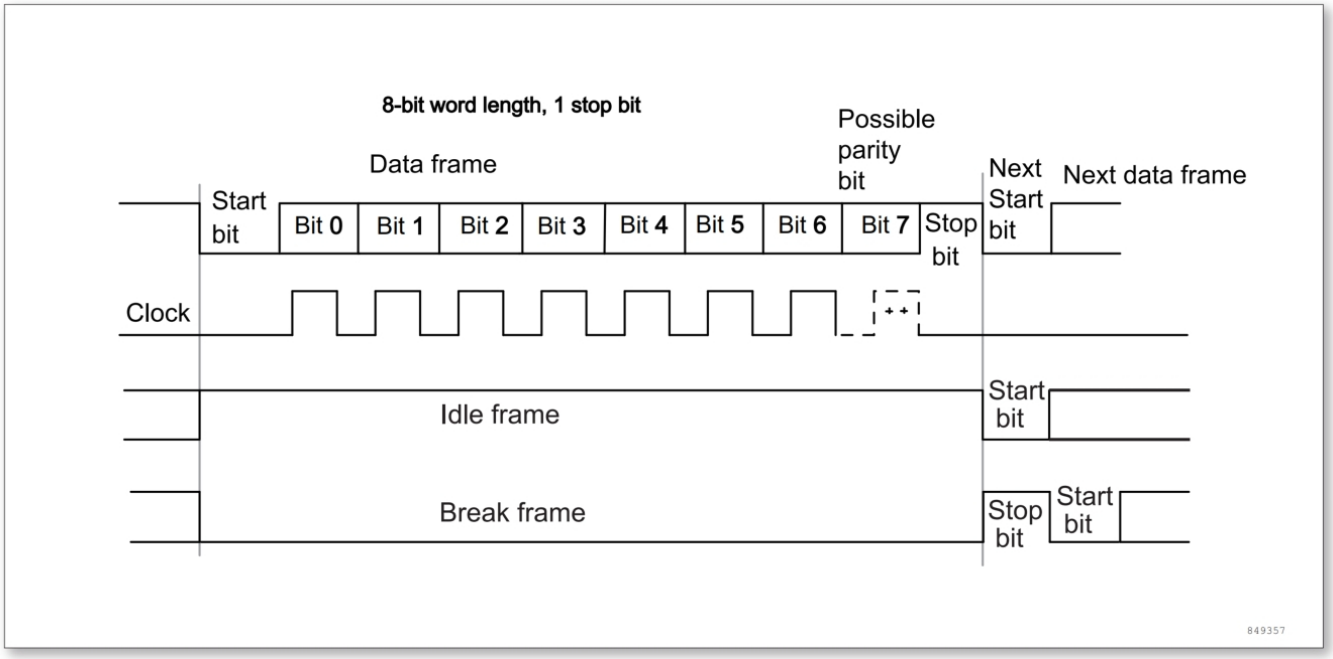


Figure 153. UART timing

16.3.2 Transmitter

The transmitter can send 5- to 8-bit data depending on the CHAR bit status. When the transmit enable bit (TXEN) is set, the data in the transmit shift register is output on the TX pin.

Character transmission

During a UART transmission, data shifts out least significant bit first on the TX pin. In this mode, the UART_TDR register consists of a buffer between the internal bus and the transmit shift register.

Every character is preceded by a start bit at low level. The character is followed by a configurable number of stop bits.

The TXEN bit should not be reset during transmission of data. Resetting this bit during the transmission will corrupt the data on the TX pin as the baud rate counters will get frozen. The current data being transmitted will be lost.

Configurable stop bits

The number of stop bits to be transmitted with every character can be programmed in bit SPB.

Break frame is comprised of start bit, data bits, optional parity bit and stop bits. Its length is not fixed, depending on the various settings of data bits, parity bit and stop bits. Once the break frame is received, the RXBRK_INTF bit in the interrupt status register is set to 1.

Procedure

1. Enable the UART by writing the UARTEN bit in UART_GCR to 1.
2. Program the CHAR bit in UART_CCR to define the word length.
3. Program the number of stop bits in SPB of UART_CCR.
4. Set the TXEN bit in UART_GCR.

5. Select the desired baud rate using UART_BRR and UART_FRA registers.
6. Write the data to send in the UART_TDR register (this clears the TX_INTF bit). Repeat this for each data to be transmitted in case of single buffer.

Single byte communication

The TX_INTF bit is always cleared by a write to the data register. The TX_INTF bit is set by hardware and it indicates:

- The data has been moved from TDR to the shift register and the data transmission has started.
- The TDR register is emptied.
- The next data can be written in the UART_TDR register without overwriting the previous data.

This flag generates an interrupt if the TXIEN bit is set. When UART transmission is taking place, a write instruction to the UART_TDR register stores the data in the TDR register and copies it in the shift register at the end of the current transmission.

When no UART transmission is taking place, a write instruction to the UART_TDR register places the data directly in the shift register, the data transmission starts, and the TX_INTF bit is immediately set. Meanwhile, the TXBUF_EMPTY bit in the UART_CSR is also set. If a frame is transmitted (after the stop bit) and no more data is written to UART_TDR (TDR register is empty), then TXC is set, indicating all transmissions are complete.

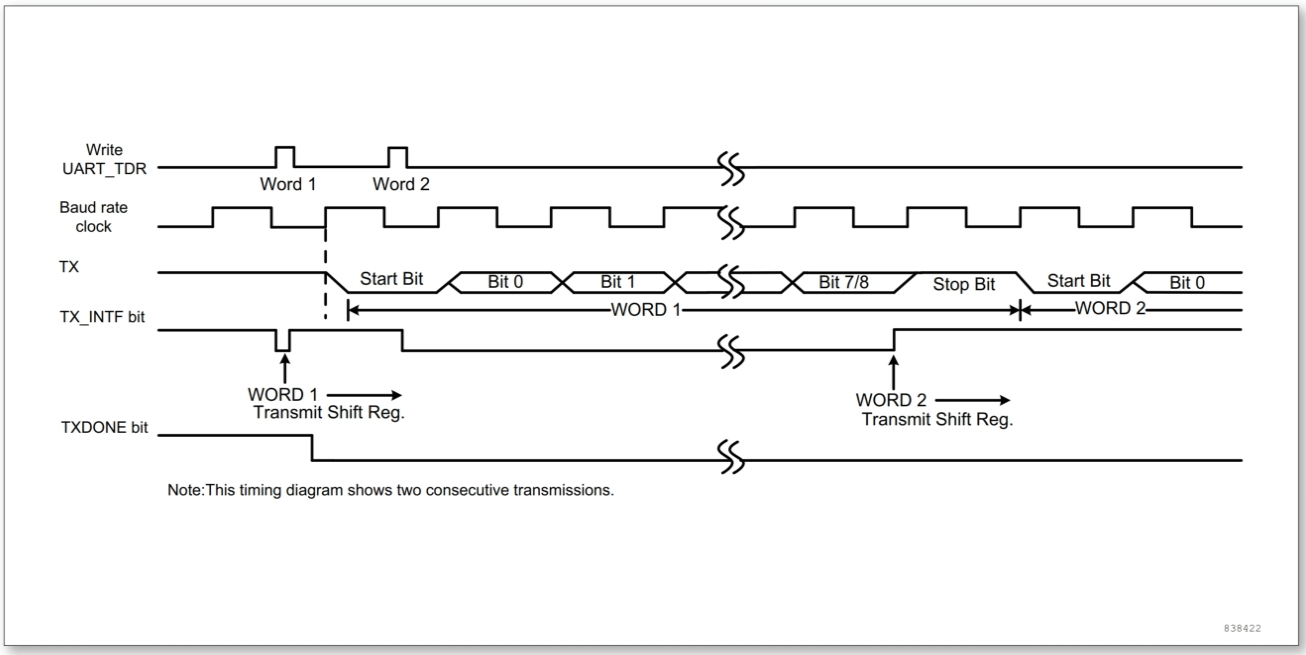


Figure 154. Status bit change during transmission

Break characters

Setting the BRK bit transmits a break character. If the BRK bit is set to '1', a break character is sent on the TX line after completing the current character transmission. This bit is reset by hardware when the break character is completed (during the stop bit of the break character).

The UART inserts a logic 1 bit at the end of the last break frame to guarantee the recognition of the start bit of the next frame.

Idle characters

After TXEN is enabled, setting the UART_IDLR bit to 1 drives the UART to send an idle

frame before the subsequent data transmission. The length of the idle frame is written to the UART_IDL register.

16.3.3 Receiver

Character reception

During a UART reception, data shifts in least significant bit first through the RX pin. In this mode, the UART_RDR register consists of a buffer between the internal bus and the receive shift register.

Procedure:

1. Enable the UART by writing the UARTEN bit in UART_GCR register to 1.
2. Program the CHAR bit in UART_CCR to define the word length.
3. Program the number of stop bits in SPB of UART_CCR.
4. Select the desired baud rate using UART_BRR and UART_FRA registers.
5. Set the RXEN bit in UART_GCR. This enables the receiver which begins searching for a start bit.

When a character is received,

- The RX_INTF bit is set. It indicates that the content of the shift register is transferred to the RDR. In other words, data has been received and can be read (as well as its associated error flags).
- An interrupt is generated if the RXIEN bit is set.
- The error flags can be set if a frame error or an overrun error has been detected during reception.
- The software reads the UART_RDR register. The RX_INTF bit must be cleared before the end of the reception of the next character.

The RXEN bit should not be reset while receiving data. If the RXEN bit is cleared during reception, the reception of the current byte will be lost.

Break character

When a break frame is received, the UART will set an RXBRK_INTF interrupt.

Idle character

If the UART_IDLR is not cleared, UART will set the RXIDL_INTF interrupt when an idle frame is received.

Overrun error

An overrun error occurs when a character is received when UART_RDR has not been read.

When an overrun error occurs:

- The RXOERR_INTF bit is set.
- The RDR content will not be lost. The previous data is available when a read to UART_RDR is performed.
- The shift register will be overwritten. Subsequently, any data received is lost.
- An interrupt is generated if the RXOERREN bit is set.

Framing error

A framing error is detected when the stop bit is not recognized or received at the expected time. When the framing error is detected:

- The RXFERR_INTF bit is set by hardware.
- The invalid data is not transferred from the Shift register to the UART_RDR register.
- An interrupt is generated if the RXFERREN bit is set.

16.3.4 9-bit data communication

If the B8EN control bit in the UART_CCR register, then the UART is able to transmit and receive 9-bit data. Note: Parity enable bit (PEN) is invalid after enabling B8E.

During data transmission, the B8TXD should be set before the data is written to the transmit register UART_TDR. B8TXD, acting as MSB of data to be sent, is delivered together with the UART_TDR value. If the B8TOG is set and the B8TXD is identical to the B8POL, then the data is used as the address frame or synchronization frame. The B8TXD will toggle automatically at the end of transmission.

It is not required to set the B8TXD as an invalid level in the subsequent data transfer.

During data reception, the most significant bit of the received data can be obtained by reading the B8RXD bit in the register. If the received B8RXD is identical to the B8POL, the RXB8_INTF bit in the interrupt status register UART_ISR will be set.

16.3.5 Multiprocessor communication

There is a possibility of performing multiprocessor communication with the UART (several UARTs connected in a network). For instance, one of the UARTs can be the master, its TX output is connected to the RX input of the other UART slaves.

Respective TX outputs of UART slaves are logically ANDed together and connected to the RX input of the master.

In multiprocessor configurations, it is often desirable that only the intended message recipient should actively receive the subsequent message contents, thus reducing redundant UART service overhead for all non addressed receivers.

The non addressed devices may be placed in mute mode by means of the muting function. In mute mode:

- None of the reception status bits can be set.
- All the receive interrupts are inhibited.
- The RWU bit in UART_CCR register is set to 1. RWU can be controlled automatically by hardware or written by the software under certain conditions.

The UART can enter or exit from mute mode using one of two methods, depending on the WAKE bit in the UART_CCR register:

- Idle bus detection if the WAKE bit is reset;
- Address Mark detection if the WAKE bit is set.

Idle bus detection (WAKE = 0)

The UART enters mute mode when the RWU bit is written to 1. It wakes up when an Idle frame is detected. Then the RWU bit is cleared by hardware but the interrupt status flag (RX_INTF) is not set. RWU can also be written to 0 by software.

Address mark detection (WAKE = 1)

In this mode, bytes are recognized as addresses if their MSB is ‘B8POL’ else they are considered as data. In an address byte, the address of the targeted receiver is compared by the receiver with its own address. The address and mask bit of the receiver are is programmed in the UART_RXADDR and UART_RXMASK registers.

The UART enters mute mode when a byte is received which does not match its programmed address. In this case, the RWU bit is set by hardware. Neither the interrupt flag (RX_INTF) is set nor an interrupt is issued for this reception, as the UART would have entered mute mode.

It exits from mute mode when a byte is received which matches the programmed address in the receiver. Then the RWU bit is cleared and subsequent bytes are received normally. The interrupt status flag (RX_INTF) is set for the matched address character since the RWU bit has been cleared.

16.3.6 Single-wire half-duplex communication

The single-wire half-duplex mode is selected by setting the HDSEL bit in the UART_SCR register. In this mode, the SCEN bit in the UART_SCR register must be kept cleared.

The UART can be configured to follow a single-wire half-duplex protocol. In single-wire half-duplex mode, the TX and RX pins are connected internally. The selection between half and full-duplex communication is made with a control bit ‘HALF DUPLEX SEL’ (HDSEL in UART_SCR).

As soon as HDSEL is written to 1:

- RX is no longer used
- TX is always released when no data is transmitted. Thus, it acts as a standard I/O in idle or in reception. It means that the I/O must be configured as floating input (or output high open-drain) when not driven by the UART.

Apart from this, the communications are similar to what is done in normal UART mode. The conflicts on the line must be managed by the software (by the use of a centralized arbiter, for instance). In particular, the transmission is never blocked by hardware and continues to occur as soon as a data entry is written in the data register while the TXEN bit is set.

16.3.7 Smartcard

The Smartcard mode is selected by setting the SCEN bit in the UART_SCR register.

The Smartcard interface is designed to support Smartcard asynchronous protocol as defined in the ISO7816-3 standard. The UART should be configured as:

- 8 bits plus parity: where CHAR = 11 and PEN = 1 in the UART_CCR register
- 1.5 stop bits when transmitting and receiving: where SPB1 = 1 and SPB0 = 1 in the UART_CCR register.

The figure below shows examples of what can be seen on the data line with and without parity error.

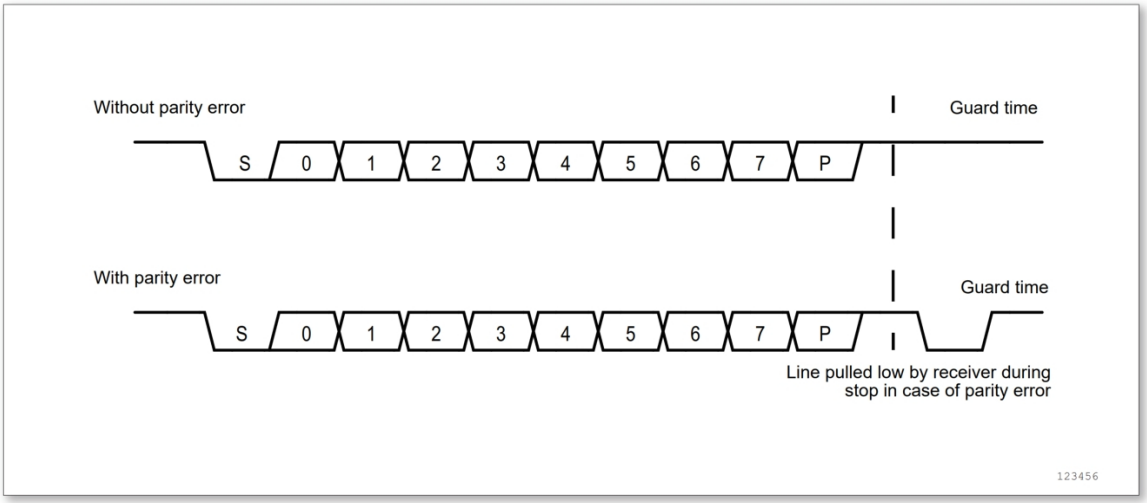


Figure 155. UART block diagram

When connected to a smartcard, the UART TX output drives a bidirectional line of the smartcard. To do this, RX and TX must be connected to the same I/O. During the transmission of start bit and data bytes, the Tx enable bit (TXEN) of the transmitter is set. This bit is released (weak pull-up) during the transmission of stop bits. Therefore, the receiver can hold the data line low in case of parity error. If the TXEN is not used and TX is held high during the transmission of stop bits, then the receiver can drive this line if TX is configured as open drain.

Smartcard is a single wire half duplex communication protocol.

- Transmission of data from the transmit shift register is delayed by a minimum of 1/2 baud clock. In normal operation, a full transmit shift register will start shifting on the next baud clock edge. In Smartcard mode, this transmission is further delayed by a guaranteed 1/2 baud clock.
- If a parity error is detected during reception of a data frame programmed with a 0.5 or 1.5 stop bit period, the transmit line is pulled low for a baud clock period after the completion of the receive frame (i.e. the end of stop bits). This is to indicate to the Smartcard that the data transmitted to UART has not been correctly received. This NACK signal (pulling transmit line low for 1 baud clock) will cause a framing error on the transmitter side (configured with 1.5 stop bits). The application can handle re-sending of data according to the protocol. A parity error is ‘NACK’ed by the receiver if the SCAEN control bit is set, otherwise a NACK is not transmitted.
- The assertion of the TXC flag can be delayed by programming the Guard Time register. In normal operation, TXC is asserted when the transmit shift register is empty and no further transmit requests are outstanding. In Smartcard mode, an empty transmit shift register triggers the guard time counter to count up to the programmed value in the Guard Time register. TXC is forced low during this time. When the guard time counter reaches the value in the Guard Time register, TXC is asserted high.
- The de-assertion of TXC flag is unaffected by Smartcard mode.
- If a framing error is detected by the transmitter (due to a NACK from the receiver), the NACK will not be detected as a start bit by the receive block of the transmitter. According to the ISO protocol, the duration of the received NACK can be 1 or 2 baud clock periods.
- On the receiver side, if a parity error is detected and a NACK is transmitted, the receiver will not detect the NACK as a start bit.

Note: 1. A break character is not significant in Smartcard mode. A 00h data with a framing error will be treated as data and not as a break.

2. No IDLE frame is transmitted when toggling the TXEN bit. The IDLE frame is not defined by the ISO protocol.

The figure below details how the NACK signal is sampled by the UART. In this example, the UART transmits a data and is configured with 1.5 stop bits. The receiver part of the UART is enabled in order to check the integrity of the data and the NACK signal.

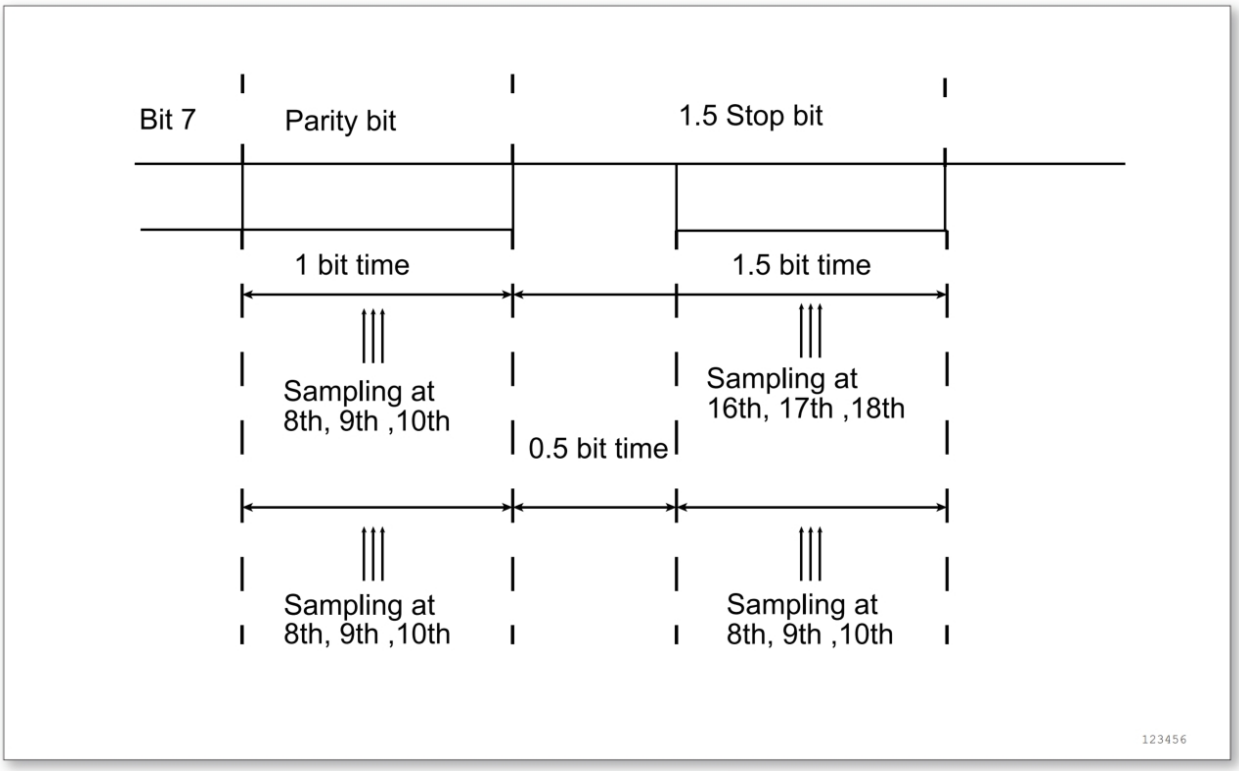


Figure 156. UART block diagram

16.3.8 Fractional baud rate generator

The corresponding baud rate can be set using the BRR and FRA registers. Refer to the formula below:

$$f_{baudrate} = \frac{f_{PCLK}}{16 \times UARTDIV}$$

$$UARTDIV = BRR + \frac{FRA}{16}$$

This leads to:

$$f_{baudrate} = \frac{f_{PCLK}}{16 \times BRR + FRA}$$

The minimum value of BRR register is 1.

16.3.9 Sampling

Since there is no individual clock for asynchronous communications, it is required that the receiver is synchronous. UART has one detection circuit to obtain correct character data

from the receive pin ‘RX’. UART adopts the x16 data baud rate ‘bclk16’ clock for data sampling from the RX pin. Each data entry is sampled by 16 clocks, from which the sampling values of the 7, 8 and 9 bits are used.

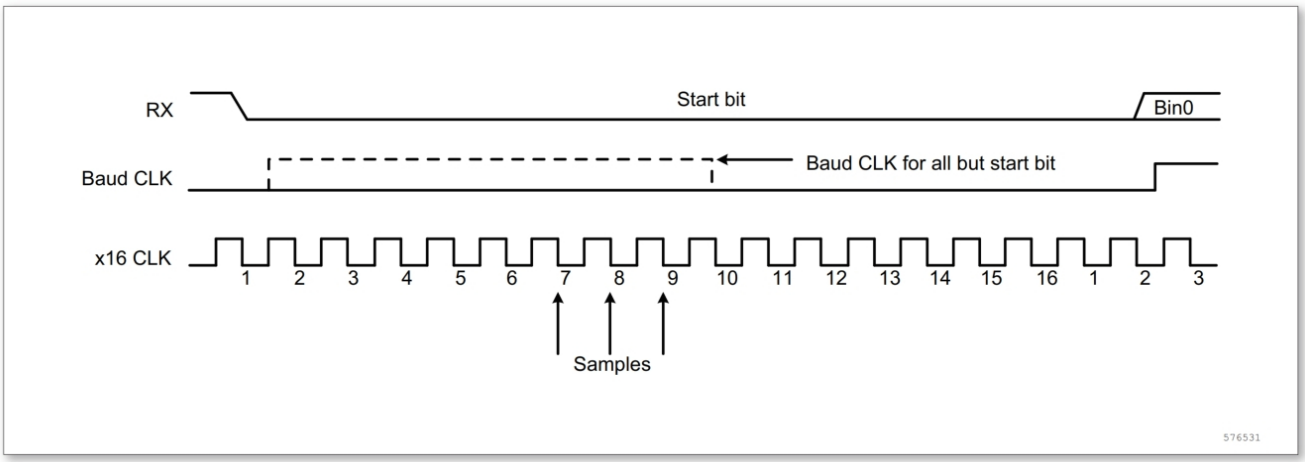


Figure 157. RX pin sampling scheme

16.3.10 Parity control

Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PEN bit in the UART_CCR register. The invalid data is not transferred from the Shift register to the UART_RDR register in case of parity error.

Even parity: the parity bit is calculated to obtain an even number of “1s” inside the frame and the parity bit.

Ex: data = 00110101; 4 bits set => parity bit will be 0 if even parity is selected (PSEL bit in UART_CCR = 1).

Odd parity: the parity bit is calculated to obtain an odd number of “1s” inside the frame and the parity bit.

Ex: data = 00110101; 4 bits set => parity bit will be 1 if odd parity is selected (PSEL bit in UART_CCR = 0).

Transmission mode**:** If the PEN bit is set in UART_CCR, then the parity bit is sent after the data in the data register is transmitted (even number of “1s” if even parity is selected or an odd number of “1s” if odd parity is selected). If the parity check fails, the RXPERR_INTF flag is set in the UART_ISR register and an interrupt is generated if RXPERREN is set in advance.

Note: The parity check is invalid if the 9bit function is enabled.

16.3.11 Automatic baud rate detection

UART can detect and set the UART_BRR register automatically according to the signal it has received.

The baud rate is automatically configured as follows:

1. Set the Former_edge and Latter_edge in the ABRCR register to automatically select the former edge and the latter edge of baud rate.
2. Set the Abr_bitcnt in the ABRCR register to select the automatic detection of baud rate bit length.
3. Set the Abr_en in the ABRCR register to enable the automatic detection of baud rate.

Enable ABRENDIEN to generate an automatic baud rate completion interrupt by the automatic end of the baud rate.

Enable ABRERRIEN to generate an automatic baud rate error interrupt in case of an automatic baud rate bit error or detection overrun error.

Ex:

1. Set Former_edge = 1, Latter_edge=0, Abr_bitcnt=2, Abr_en=1

When the data 0xEF has been received, the baud rate will be detected automatically and the UART_BRR register will be reset. The Rxfifo will receive the remaining data 0x0E.

2. Set Former_edge = 0, Latter_edge=1, Abr_bitcnt=2, Abr_en=1

When the data 0xF8 has been received, the baud rate will be detected automatically and the UART_BRR register will be reset. The Rxfifo will receive the remaining data 0x1F.

Note: If Former_edge = 1 and the LSB bit of received data is not 1, then a reception error will occur. No data will be in the Rxfifo.

16.4 UART interrupt requests

Table 52. UART interrupt requests

Interrupt Event	Interrupt Status	Enable Bit
Tx buffer empty	TX_INTF	TXIEN
Effective data received	RX_INTF	RXIEN
Tx shift register complete	TXC_INTF	TXC_EN
Rx overrun error	RXOERR_INTF	RXOERREN
Parity error	RXPERR_INTF	RXPERREN
Framing error	RXFERR_INTF	RXFERREN
Rx break frame	RXBRK_INTF	RXBRKEN
Tx break frame	TXBRK_INTF	TXBRK_EN
Rx synchronous frame	RXB8_INTF	RXB8_EN
Rx idle frame	RXIDL_INTF	RXIDLEN
Automatic end of baud rate	ABREND_INTF	ABRENDIEN
Automatic baud rate error	ABRERR_INTF	ABRERRIEN

These events generate an interrupt if the corresponding interrupt Enable Control Bit is set.

16.5 UART register description

Table 53. Overview of UART registers

Offset	Acronym	Register Name	Reset	Section
0x00	UART_TDR	UART transmit data register	0x00000000	Section 16.5.1
0x04	UART_RDR	UART receive data register	0x00000000	Section 16.5.2
0x08	UART_CSR	UART current status register	0x00000009	Section 16.5.3
0x0C	UART_ISR	UART interrupt status register	0x00000000	Section 16.5.4
0x10	UART_IER	UART interrupt enable register	0x00000000	Section 16.5.5
0x14	UART_ICR	UART interrupt clear register	0x00000000	Section 16.5.6
0x18	UART_GCR	UART global control register	0x00000000	Section 16.5.7
0x1C	UART_CCR	UART common control register	0x00000000	Section 16.5.8
0x20	UART_BRR	UART baud rate register	0x00000001	Section 16.5.9
0x24	UART_FRA	UART fractional baud rate register	0x00000000	Section 16.5.10
0x28	UART_RXADDR	UART receive address register	0x00000000	Section 16.5.11
0x2C	UART_RXMASK	UART receive mask register	0x000000FF	Section 16.5.12
0x30	UART_SCR	UART SCR register	0x00000000	Section 16.5.13
0x34	UART_IDLR	UART IDLE data length register	0x0000000C	Section 16.5.14
0x38	UART_ABRCR	UART ABRCR automatic baud rate control register	0x00000000	Section 16.5.15

16.5.1 UART transmit data register (UART_TDR)

Address offset: 0x00
Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31:8	Reserved			always read as 0
7:0	TXREG	rw	0x00	Transmit data register

16.5.2 UART receive data register (UART_RDR)

Address offset: 0x04
Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31:8	Reserved			always read as 0
7:0	RXREG	r	0x00	Receive data register This is a read-only register.

16.5.3 UART current status register (UART_CSR)

Address offset: 0x08
Reset value: 0x0000 0009

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												TXBUF_EMPTY	TXFULL	RXAVL	TXC
												r	r	r	r

Bit	Field	Type	Reset	Description
31:4	Reserved			always read as 0.
3	TXBUF_ EMPTY	r	0x01	Transmit buffer empty flag bit 1: Transmit buffer is empty 0: Transmit buffer is not empty
2	TXFULL	r	0x00	Transmit buffer full flag bit 1: Transmit buffer is full 0: Transmit buffer is not full

Bit	Field	Type	Reset	Description
1	RXAVL	r	0x00	Receive valid data flag bit This bit is set when the Rx buffer has received a complete byte. 1: Rx buffer has received a complete and effective byte. 0: Rx buffer is empty.
0	TXC	r	0x01	Transmit complete flag bit 1: Tx buffer and Tx shift register are both empty. 0: Tx buffer is not empty.

16.5.4 UART interrupt status register (UART_ISR)

Address offset: 0x0C
Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				ABRERR_INTF	ABREND_INTF	RXIDLE_INTF	RXB8_INTF	TX BRK_INTF	RX BRK_INTF	RX FERR_INTF	RXP ERR_INTF	RXO ERR_INTF	TXC_INTF	RX_INTF	TX_INTF
				r	r	r	r	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31:12	Reserved			always read as 0.
11	ABRERR_INTF	r	0x00	UART auto baud rate error interrupt flag bit 1: Auto baud rate error 0: No auto baud rate error
10	ABREND_INTF	r	0x00	UART auto baud rate end interrupt flag bit 1: Auto baud rate is at the end 0: Automatic baud rate is not at the end
9	RXIDLE_INTF	r	0x00	UART receive frame idle interrupt flag bit After stop bits, RX pin has received several (number of UART_IDLR) high levels. 1: Idle frame detected 0: No idle frame
8	RXB8_ INTF	r	0x00	UART synchronization frame interrupt flag bit In 9-bit communication mode, the RXB8_INT is set once the 9 bit of received data is identical to the CCR.B8POL register. This bit can be treated as an interrupt request signal. 1: Synchronization frame received 0: Synchronization frame not received

Bit	Field	Type	Reset	Description
7	TXBRK_INTF	r	0x00	UART break frame transmission complete interrupt flag bit 1: Transmission of shift register break frame data is complete 0: Shift register is empty or shift transmission is ongoing Note: The break frame cannot be sent continuously.
6	RXBRK_INTF	r	0x00	UART receive frame break interrupt flag bit After abnormal stop bits, RX pin has received no less than 10 low levels. 1: Break frame detected 0: No break frame
5	RXFERR_INTF	r	0x00	Frame error interrupt flag bit) A frame error occurs when an abnormal stop bit is detected. 1: A frame error detected 0: no frame error
4	RXPERR_INTF	r	0x00	Parity error interrupt flag bit 1: Parity error detected 0: No parity error
3	RXOERR_INTF	r	0x00	Receive overflow error interrupt flag bit 1: Overrun error received 0: No overrun error
2	TXC_INTF	r	0x00	UART Tx shift register complete interrupt flag bit 1: Transmission of shift register data is complete 0: Shift register is empty or shift transmission is ongoing Note: The flag bit is related with the guard time.
1	RX_INTF	r	0x00	Receive valid data interrupt flag bit This bit is set when the Rx buffer receives a complete byte. 1: Rx buffer has received data of effective byte. 0: Rx buffer is empty.
0	TX_INTF	r	0x00	Transmit buffer empty interrupt flag bit 1: Transmit buffer is empty 0: Transmit buffer is not empty

16.5.5 UART interrupt enable register (UART_IER)

Address offset: 0x10

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				ABRERRIEN	ABRENDIEN	RXIDLEN	RXB8_IEN	TXBRK_IEN	RXBRKEN	RXFERREN	RXPERREN	RXOERREN	TXC_IEN	RXIEN	TXIEN
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:12	Reserved			always read as 0.
11	ABRERRIEN	rw	0x00	Auto baud rate error interrupt enable bit 1: interrupt enabled 0: interrupt disabled
10	ABRENDIEN	rw	0x00	Auto baud rate end interrupt enable bit 1: interrupt enabled 0: interrupt disabled
9	RXIDLEN	rw	0x00	Receive frame idle interrupt enable bit 1: interrupt enabled 0: interrupt disabled
8	RXB8_IEN	rw	0x00	UART synchronization frame interrupt enable control bit 1: Rx synchronization frame interrupt enabled 0: Rx synchronization frame interrupt disabled
7	TXBRK_IEN	rw	0x00	UART break frame transmission complete interrupt enable control bit 1: Tx break frame complete interrupt enabled 0: Tx break frame complete interrupt disabled
6	RXBRKEN	rw	0x00	UART receive frame break interrupt enable bit 1: interrupt enabled 0: interrupt disabled
5	RXFERREN	rw	0x00	Frame error interrupt enable bit 1: interrupt enabled 0: interrupt disabled
4	RXPERREN	rw	0x00	Parity error interrupt enable bit 1: interrupt enabled 0: interrupt disabled
3	RXOERREN	rw	0x00	Receive overflow error interrupt enable bit 1: interrupt enabled 0: interrupt disabled

Bit	Field	Type	Reset	Description
2	TXC_IEN	rw	0x00	UART Tx shift register complete interrupt enable control bit 1: Transmission of shift register data is complete 0: Shift register is empty or shift transmission is ongoing
1	RX_IEN	rw	0x00	Receive buffer interrupt enable bit 1: interrupt enabled 0: interrupt disabled
0	TX_IEN	rw	0x00	Transmit buffer empty interrupt enable bit 1: interrupt enabled 0: interrupt disabled

16.5.6 UART interrupt clear register (UART_ICR)

Address offset: 0x14
Reset value: 0x0000 0000

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

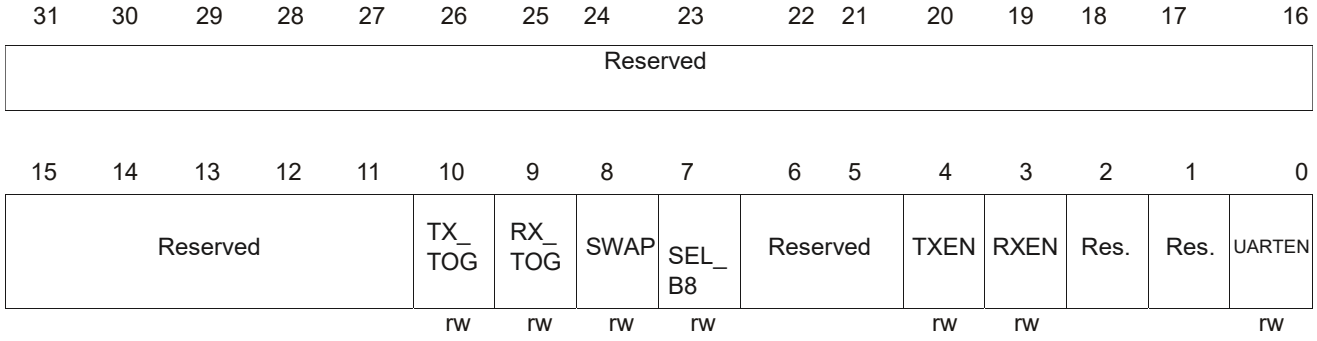
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				ABRERR CLR	ABREND CLR	RXIDL CLR	RXB8 _CLR	TXBRK _CLR	RX BRK CLR	RX FEER CLR	RXP ERR CLR	RXO ERR CLR	TXC_ ICLR	RX _ICLR	TX _ICLR
				w	w	w	w	w	w	w	w	w	w	w	w

Bit	Field	Type	Reset	Description
31:12	Reserved			always read as 0.
11	ABRERRCLR	w	0x00	UART auto baud rate error interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
10	ABRENDCLR	w	0x00	UART auto baud rate end interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
9	RXIDLCLR	w	0x00	UART receive frame idle interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
8	RXB8_CLR	w	0x00	UART synchronization frame interrupt flag clear control bit 1: Rx synchronization frame interrupt flag cleared 0: No action

Bit	Field	Type	Reset	Description
7	TXBRK_CLR	w	0x00	UART break frame transmission complete interrupt flag clear control bit 1: Break frame transmission complete interrupt flag cleared 0: No action
6	RXBRKCLR	w	0x00	UART receive frame break interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
5	RXFERRCLR	w	0x00	Frame error interrupt enable bit 1: interrupt cleared 0: interrupt uncleared
4	RXPERRCLR	w	0x00	Parity error interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
3	RXOERRCLR	w	0x00	Receive overflow error interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
2	TXC_ICLR	w	0x00	Transmit complete interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
1	RX_ICLR	w	0x00	Receive interrupt clear bit 1: interrupt cleared 0: interrupt uncleared
0	TX_ICLR	w	0x00	Transmit buffer empty interrupt clear bit 1: interrupt cleared 0: interrupt uncleared

16.5.7 UART global control register (UART_GCR)

Address offset: 0x18
Reset value: 0x0000 0000



Bit	Field	Type	Reset	Description
31:11	Reserved			always read as 0.
10	TX_TOG	rw	0x00	Transmit toggle bit 1: Transmit toggle enabled 0: invalid
9	RX_TOG	rw	0x00	Receive toggle bit 1: Receive toggle enabled 0: invalid
8	SWAP	rw	0x00	Swap between input and output 1: Input and output are swapped 0: invalid Note: Once the SWAP bit is set, the MODE in the GPIOx_CRL register needs to be changed, e.g. the original input mode changes to the output mode.
7	SEL_B8	rw	0x00	This bit selects whether B8 data transmission or reception is valid or not. 1: B8 data transfer 0: invalid When the B8EN bit is enabled in the UART_CCR, 9-bit data is allowed to transmit; the data with a length of 'CHAR bit +1' is allowed to receive.
6:5	Reserved			always read as 0
4	TXEN	rw	0x00	Enable transmit 1: Transmission enabled. 0: Transmission disabled. TX BUFFER can be cleared.
3	RXEN	rw	0x00	Enable receive 1: Reception enabled. 0: Reception disabled. RX BUFFER can be cleared.
2	Reserved			always read as 0
1	Reserved			always read as 0
0	UARTEN	rw	0x00	UART mode selection bit 1: UART mode enabled 0: UART mode disabled

16.5.8 UART common control register (UART_CCR)

Address offset: 0x1C

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	LIN	WAKE	RWU	B8EN	B8TOG	B8POL	B8TXD	B8RXD	SPB1	CHAR		BRK	SPB0	PSEL	PEN
	rw	rw	rw	rw	rw	rw	rw	r	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:15	Reserved			always read as 0.
14	LIN	rw	0x00	UART LIN protocol transfer break frame enable bit 1: LIN protocol valid 0: LIN protocol invalid
13	WAKE	rw	0x00	Wakeup method This bit determines the way to wake up UART. 1: Address flag wakeup 0: Idle bus wakeup Note: UART should have received one byte before it is placed in the mute mode. Otherwise, it will not be waken up by idle bus detection.
12	RWU	rw	0x00	Receive wakeup This bit determines whether to place the UART in mute mode. It can be set or reset by software. When a wakeup sequence arrives, the hardware will clear it automatically. 1: Receiver in mute mode 0: Receiver in normal mode If the Address flag wakeup is set, it can not be revised by software, if the receive buffer is not empty.
11	B8EN	rw	0x00	UART synchronization frame bit 9 enable control bit The enable bit is checked when this bit is enabled. PEN is then invalid. 1: Synchronization frame bit 9 transmission enabled 0: Synchronization frame bit 9 transmission disabled
10	B8TOG	rw	0x00	UART synchronization frame transmission bit 9 auto toggle control bit 1: Bit 9 auto toggle enabled 0: Bit 9 auto toggle disabled Note: When B8TXD value is identical to B8POL value and the register configuration is complete, the first transmitted data is the default address bit and the second data begins to toggle.
9	B8POL	rw	0x00	UART synchronization frame bit 9 polarity control bit 1: Synchronization frame bit 9 high level enabled 0: Synchronization frame bit 9 low level enabled
8	B8TXD	rw	0x00	UART synchronization frame transmitted data bit 9 1: Tx synchronization frame bit 9 is high level 0: Tx synchronization frame bit 9 is low level

Bit	Field	Type	Reset	Description
7	B8RXD	r	0x00	UART synchronization frame received data bit 9 Read-only. 1: Rx synchronization frame bit 9 is high level 0: Rx synchronization frame bit 9 is low level
6	SPB1	rw	0x00	Stop selection bit, which sets the number of stop bits in conjunction with SPB0.
5:4	CHAR	rw	0x00	UART data width bit 00: 5 bits 01: 6 bits 10: 7 bits 11: 8 bits
3	BRK	rw	0x00	UART transmit frame break 1: serial forced output of logic '0' (frame break) 0: break disabled
2	SPB0	rw	0x00	Stop bit selection Set the number of Tx stop bits. SPB1, SPB0: 00, 1 stop bits SPB1, SPB0: 01, 2 stop bits SPB1, SPB0: 10, 0.5 stop bits SPB1, SPB0: 11, 1.5 stop bits
1	PSEL	rw	0x00	Parity selection bit When enabled, this bit selects whether even parity or odd parity is used. 1: even parity 0: odd parity
0	PEN	rw	0x00	Parity enable bit 1: Tx and Rx parity check enabled. 0: Parity check disabled.

16.5.9 UART baud rate register (UART_BRR)

Address offset: 0x20
Reset value: 0x0000 0001

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
DIV_Mantissa															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			always read as 0.
15:0	DIV_Mantissa	rw	0x0001	Integer part of UARTDIV These 16 bits define the integer part of the UART frequency divider division factor (UARTDIV). The minimum value of DIV_Mantissa is 1.

16.5.10 UART fractional baud rate register (UART_FRA)

Address offset: 0x24

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31:4	Reserved			always read as 0.
3:0	DIV_Fraction	rw	0x00	Fraction part of UARTDIV These 4 bits define the fraction part of the UART frequency divider division factor (UARTDIV).

16.5.11 UART receive address register (UART_RXADDR)

Address offset: 0x28

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31:8	Reserved			always read as 0.
7:0	RXADDR	rw	0x00	Local address match with the UART synchronization frame data. When RXMASK = 0xFF, RXB8_INTF is generated when the received synchronization frame data is identical to the local address match. Address 0 is the general call address. A response will be generated when this address is received.

16.5.12 UART receive mask register (UART_RXMASK)

Address offset: 0x2C

Reset value: 0x0000 00FF

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

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

Bit	Field	Type	Reset	Description
31:8	Reserved			always read as 0.
7:0	RXMASK	rw	0xFF	If all data bits are ‘0’, a synchronization frame break request is generated when any data is received. If a data bit is ‘1’, a synchronization frame break request is generated when the RDR matches with the corresponding bit in the RXADDR.

16.5.13 UART SCR register (UART_SCR)

Address offset: 0x30

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			HDSEL	SCFCNT								Res.	NACK	SCAEN	SCEN
			rw	rw	rw	rw	rw	rw	rw	rw	rw		r	rw	rw

Bit	Field	Type	Reset	Description
31:13	Reserved			Reserved, always read as 0.
12	HDSEL	rw	0x00	Single-wire half-duplex mode selection. 1: Enable the half-duplex mode 0: Disable the half-duplex mode
11:4	SCFCNT	rw	0x00	ISO7816 guarding counter. If the Tx data is at low level in the period of guarding counter, it is not allowed to be the start bit of next data. 0 = 16 baud rate count time 15..1 = 15..1 time
3	Reserved			Reserved and always read as 0.
2	NACK	r	0x00	Master receive frame ACK bit
1	SCAEN	rw	0x00	ISO7816 check auto ACK bit 1: Auto ACK enabled 0: Auto ACK disabled
0	SCEN	rw	0x00	ISO7816 enable control bit 1: ISO7816 function enabled 0: ISO7816 function disabled

16.5.14 UART IDLE data length register (UART_IDLR)

Address offset: 0x34
Reset value: 0x0000 000C

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

Bit	Field	Type	Reset	Description
31:16	Reserved			Reserved and always read as 0.
15:0	IDLR	rw	0x000C	UART idle data length register The data length is not 0.

16.5.15 UART ABRCR automatic baud rate control register (UART_ABRCR)

Address offset: 0x38
Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											Latter_edge	Former_edge	Abr_bitcnt		Abren
											rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:5	Reserved			Reserved, always read as 0.
4	LATTER_EDGE	rw	0x00	Select the latter edge of baud rate automatically 1: Rising edge 0: Falling edge
3	FORMER_EDGE	rw	0x00	Select the former edge of baud rate automatically 1: Rising edge 0: Falling edge
2:1	ABR_BITCNT	rw	0x00	Automatic baud rate length detection. This bit detects the length of bits between the former edge and the latter edge. 11:1 bit 10: 2 bits 01: 4 bits 00: 8 bits
0	ABREN	rw	0x00	Auto baud rate enable. This bit enables the auto baud rate only when the UART is idle. After it is enabled, it detects the edge of received signal. After the auto baud rate detection, the hardware set the UART_BRR and UART_FRA registers automatically. 1: Auto baud rate enabled 0: Auto baud rate disabled

17 System Configuration Controller (SYSCFG)

System Configuration Controller (SYSCFG)

The devices feature a set of system configuration registers. The main purposes of the registers are the following:

- Managing the external interrupt line connection to the GPIOs.
- Remapping the memory located at the beginning of the code area
- Configuring the external interrupt pins

17.1 SYSCFG register description

Table 54. Overview of SYSCFG registers

Offset	Acronym	Register Name	Reset	Section
0x00	SYSCFG_CFGR	SYSCFG configuration register	0x0000000X	Section 17.1.1
0x08	SYSCFG_EXTICR1	External interrupt configuration register 1	0x00000000	Section 17.1.2
0x0C	SYSCFG_EXTICR2	External interrupt configuration register 2	0x00000000	Section 17.1.3
0x10	SYSCFG_EXTICR3	External interrupt configuration register 3	0x00000000	Section 17.1.4
0x14	SYSCFG_EXTICR4	External interrupt configuration register 4	0x00000000	Section 17.1.5
0x18	SYSCFG_PADHYS	PAD configuration register	0x00000000	Section 17.1.6

17.1.1 SYSCFG configuration register (SYSCFG_CFGR)

This register has two control bits that are used to configure the storage area type of memory start address 0x0000 0000. These bits are configured by software to bypass the BOOT selection.After reset, these bits take the value selected by the actual BOOT mode configuration.

Address offset: 0x00

Reset value: 0x0000 000X (X is the control bit selected by the actual BOOT mode configuration)

(SYSCFG)

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

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

Bit	Field	Type	Reset	Description
31:2	Reserved			always read as 0.
1:0	MEM_MODE	rw	0x00	Memory mapping selection bits These bits are set and cleared by software. They control the memory internal mapping at address 0x0000 0000. After reset, these bits take on the value selected by the BOOT0 pin configuration and nBOOT1 bit. x0: Main Flash memory mapped at 0x0000 0000 01: System Flash memory mapped at 0x0000 0000 11: Embedded RAM mapped at 0x0000 0000

17.1.2 External interrupt configuration register 1 (SYSCFG_EXTICR1)

Address offset: 0x08

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI3				EXTI2				EXTI1				EXTI0			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			always read as 0.
15:0	EXTIx	rw	0x00	EXTI x configuration bits (x = 0 to 3) These bits are written and read by software to select the input source for the EXTIx external interrupt. 0000: PA[x] pin 0001: PB[x] pin 0010: reserved 0011: reserved

17.1.3 External interrupt configuration register 2 (SYSCFG_EXTICR2)

Address offset: 0x0C

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI7				EXTI6				EXTI5				EXTI4			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			always read as 0.
15:0	EXTIx	rw	0x00	EXTI x configuration bits (x = 4 to 7) These bits are written and read by software to select the input source for the EXTIx external interrupt. 0000: PA[x] pin 0001: PB[x] pin 0010: reserved 0011: reserved

17.1.4 External interrupt configuration register 3 (SYSCFG_EXTICR3)

Address offset: 0x10

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI11				EXTI10				EXTI9				EXTI8			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			always read as 0.
15:0	EXTIx	rw	0x00	EXTI x configuration bits (x = 8 to 11) These bits are written and read by software to select the input source for the EXTIx external interrupt. 0000: PA[x] pin 0001: PB[x] pin 0010: reserved 0011: reserved

17.1.5 External interrupt configuration register 4 (SYSCFG_EXTICR4)

Address offset: 0x14

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI15				EXTI14				EXTI13				EXTI12			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			always read as 0.
15:0	EXTIx	rw	0x00	EXTI x configuration bits (x = 12 to 15) These bits are written and read by software to select the input source for the EXTIx external interrupt. 0000: PA[x] pin 0001: PB[x] pin 0010: reserved 0011: reserved

17.1.6 PAD configuration register (SYSCFG_PADHYS)

Address offset: 0x18

Reset value: 0x0000 0000

(SYSCFG)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															I2C1_MODE_SEL
															rw

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

Bit	Field	Type	Reset	Description
31:17	Reserved			always read as 0.
16	I2C1_MODE_SEL	rw	0x00	I2C1 port mode selection bit 0: Open-drain mode 1: Push-pull mode
15:0	Reserved			always read as 0

18 Device Electronic Signature (Device)

Device Electronic Signature (Device)

The electronic signature is stored in the system memory area in the Flash memory module, and can be read using the JTAG, SWD or the CPU.

It contains factory-programmed identification data that allow the user firmware or other external devices to automatically match its interface to the characteristics of the microcontroller.

18.1 Memory size registers

18.1.1 Unique device ID register (96 bits)

The unique device identifier is ideally suited:

- for use as serial numbers (for example USB string serial numbers or other end applications)
- for use as security keys in order to increase the security of code in Flash memory while using and combining this unique ID with software encryption-decryption algorithm before programming the internal Flash memory
- to activate secure boot processes

The 96-bit unique device identifier provides a reference number which is unique for any microcontroller and in any context. These bits can never be altered by the user.

The 96-bit unique device identifier can also be read in single bytes (8 bits)/half-words (16 bits)/words (32 bits) in different ways.

18.2 UID register description

Table 55. Overview of memory size register description

Offset	Acronym	Register Name	Reset	Section
0x00	UID1	Unique ID	0xXXXXXXXX	Section 18.2.1
0x02	UID2	Unique ID	0xXXXXXXXX	Section 18.2.2
0x04	UID3	Unique ID	0xXXXXXXXX	Section 18.2.3
0x08	UID4	Unique ID	0xXXXXXXXX	Section 18.2.4

18.2.1 Unique ID (UID1)

Base address: 0x1FFF F7E8

Address offset: 0x00

Read-only and factory-programmed

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

Bit	Field	Type	Reset	Description
15:0	U_ID	r		U_ID: 15:0 unique ID bits This field value is also reserved for a future feature.

18.2.2 Unique ID (UID2)

Address offset: 0x02

Read-only and factory-programmed

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

Bit	Field	Type	Reset	Description
15:0	U_ID	r		U_ID: 31:16 unique ID bits This field value is also reserved for a future feature.

18.2.3 Unique ID (UID3)

Address offset: 0x04

Read-only and factory-programmed

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

Bit	Field	Type	Reset	Description
31:0	U_ID	r		U_ID: 63:32 unique ID bits This field value is also reserved for a future feature.

18.2.4 Unique ID (UID4)

Address offset: 0x08

Read-only and factory-programmed

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

Bit	Field	Type	Reset	Description
31:0	U_ID	r		U_ID: 95:64 unique ID bits This field value is also reserved for a future feature.

19 Debug Support (DBG)

Debug Support (DBG)

19.1 Overview

This series is built around a core which contains hardware extensions for advanced debugging features. The debug extensions allow the core to be stopped either on a given instruction fetch (breakpoint) or data access (watchpoint). When stopped, the core's internal state and the system's external state may be examined. Once examination is complete, the core and the peripherals may be restored and program execution resumed.

The debug features are used by the debugger host when connecting to and debugging the microcontroller.

Supports:

- Serial wire debug port

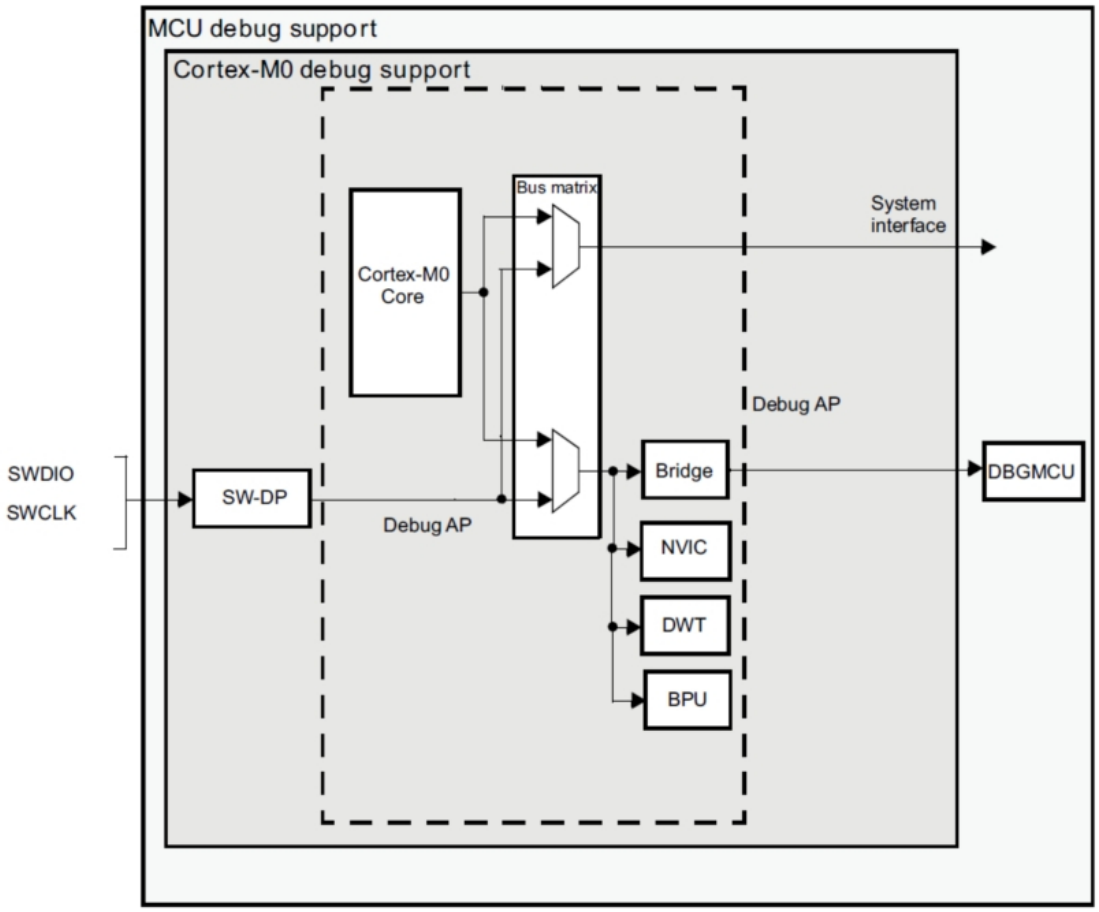


Figure 158. Block diagram of MM32 series and CPU level debug support

The CPU core provides integrated on-chip debug support. It is comprised of:

- SW-DP: serial wire debug port

(DBG)

- AHP-AP: AHB access port
- ITM: instrumentation trace macrocell
- FPB: Flash patch breakpoint
- DWT: Data watchpoint trigger
- TPUI: Trace port unit interface

19.2 Pinout and debug port pins

This chip is available in various packages with different numbers of available pins. As a result, some functionality related to pin availability may differ between packages.

19.2.1 SWD debug port pins

Two general-purpose I/O ports from the chip are used as pins for the SW-DP. These pins are available on all packages.

Table 56. SWJ debug port pins

SWJ-DP Pin Name	SW Debug Port		Pin Assignment
	Type	Debug Assignment	
SW debug port	Input/Output	Serial Wire Data Input/Output	PA13
SWCLK	Input	Serial Wire Clock	PA14

19.2.2 Internal pull-up and pull-down on SWD pins

It is necessary to ensure that the SWD input pins are not floating since they are directly connected to D triggers to control the debug mode features. Special care must be taken with the SWCLK pin which is directly connected to the clock of some of these D triggers.

To avoid any uncontrolled IO levels, the device embeds internal pull-ups and pull-downs on the SWD input pins.

- SWDIO: internal pull-up
- SWCLK: input pull-down

The software can then use these I/Os as standard GPIOs.

19.3 ID codes and locking mechanism

There are several ID codes inside the MCU.

19.3.1 MCU device ID code

The MCU integrates an MCU ID code. This ID identifies the MCU part number and the die revision. It is part of the DBG_MCU component and is mapped on the external APB bus. This code is accessible using the SW debug port (two pins) or by the user software.

DBGMCU_IDCODE

(DBG)

Address: 0x40013400

Only 32-bits access supported. Read-only.

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

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

Table 57

bits 31: 0	DEV_ID: Device identifier

19.3.2 Cortex JEDEC-106 ID code

The CPU integrates a JEDEC-106 ID code. It is located in the 4KB ROM table mapped on the internal APB bus at address 0xE00FF000_0xE00FFFFF.

The table below shows the ID codes of this series:

Table 58. ID codes

ID name	Chip
DEV_ID	0x8C4350D1
CPU TAP SW ID	0x0BB11477

19.4 SW debug port

19.4.1 SW protocol introduction

This synchronous serial protocol uses two pins:

- SWCLK: clock from host to target
- SWDIO: bidirectional data signal

The protocol allows two banks of registers (DPACC registers and APACC registers) to be read and written to.

Bits are transferred LSB-first on the wire.

For SWDIO is bidirectional, the line must be pulled-up on the board (100 K□ recommended).

Each time the direction of SWDIO changes in the protocol, a turnaround time is inserted where the line is not driven by the host nor the target. By default, this turnaround time is one bit time, however this can be adjusted by configuring the SWCLK frequency.

19.4.2 SW protocol sequence

Each sequence consists of three phases:

- Packet request (8 bits) transmitted by the host
- Acknowledge response (3 bits) transmitted by the target
- Data transfer phase (33 bits) transmitted by the host or the target

Table 59. Packet request (8-bits)

Bit	Name	Description
0	Start	Must be “1”
1	APnDP	0: DP Access 1: AP Access
2	RnW	0: Write Request 1: Read Request
4:3	A[3:2]	Address field of the DP or AP registers
5	Parity	Parity bit of preceding bits
6	Stop	0
7	Park	Not driven by the host. Must be read as “1” by the target because of the pull-up.

Refer to the CPU Technical Reference Manual for a detailed description of DPACC and APACC registers. The packet request is always followed by the turnaround time (default 1 bit) where neither the host nor target drives the line.

Table 60. ACK response (3 bits)

Bit	Name	Description
0 .. 2	ACK	001: FAULT 010: WAIT 100: OK

The ACK Response must be followed by a turnaround time only if it is a READ transaction or if a WAIT or FAULT acknowledge has been received.

Table 61. ACK response (33 bits)

Bit	Name	Description
0 .. 31	WDATA/RDATA	Write or Read data
32	Parity	Parity bit of the 32 data bits

The DATA transfer must be followed by a turnaround time only if it is a READ transaction.

19.4.3 SW-DP state machine (Reset, idle states, ID code)

The State Machine of the SW-DP has an internal ID code which identifies the SW-DP. It follows the JEP-106 standard.

The SW-DP state machine is inactive until the debugger reads this ID code.

- The SW-DP state machine is in RESET STATE either after power-on reset, or after the DP has switched from to SWD or after the line is high for more than 50 cycles.
- The state machine switches to the IDLE STATE if the line is low for at least two cycles when the state machine is in the RESET state.
- After RESET state, it is mandatory to first enter into an IDLE state AND to perform a READ access of the DP-SW ID register. Otherwise, the debugger will receive a FAULT acknowledge response on another transactions.

19.4.4 DP and AP read/write accesses

- Read accesses to the DP are not posted: the data can be obtained by the debugger immediately (if ACK = OK) or can be delayed (if ACK = WAIT).
- Read accesses to the AP are posted. This means that the result of the access is returned on the next transfer. If the next access to be done is NOT an AP access, then the DP-RDBUFF register must be read to obtain the result.
- The READOK flag of the DP-CTRL/STAT register is updated on every AP read access or RDBUFF read request to notify the debugger if the AP read access was successful.
- The SW-DP implements a write buffer (for both DP or AP writes), that enables it to accept a write operation even when other transactions are still outstanding. If the write buffer is full, the acknowledge response obtained by the debugger is “WAIT”. IDCODE read or CTRL/STAT read or ABORT write are accepted even if the write buffer is full.
- Because of the asynchronous clock domains SWCLK and HCLK, two extra SWCLK cycles are needed after a write transaction (after the parity bit) to make the write effective internally. These cycles should be applied while driving the line low (IDLE state). This is particularly important when writing the CTRL/STAT for a power-up request. Otherwise, the next transaction (requiring a power-up) occurs immediately and leads to failure.

19.4.5 SW-DP registers

Access to these registers are initiated when APnDP = 0.

Table 62

A[3:2]	Read/Write	CTRLSEL bit in the SELECT register	Register	Description
00	Read		IDCODE	Fixed at 0x1BA01477 (identifies the SW-DP).
00	Write		ABORT	
01	Read/Write	0	DP-CTRL /STAT	Request a system or debug power-up; configure the transfer operation for AP accesses; control the compare and verify operations; read some status flags (overrun, power-up acknowledges).

A[3:2]	Read/Write	CTRLSEL bit in the SELECT register	Register	Description
01	Read/Write	1	WIRE CONTROL	Configure the physical serial port protocol (like the duration of the turnaround time).
10	Read		READ RESEND	Enables recovery of the data from a corrupted debugger transfer, without repeating the original AP transfer.
10	Write		SELECT	Select the current access port and the active 4-words register window.
11	Read/Write		READ BUFFER	This register is useful because AP accesses are posted (the result of a read AP request is available on the next AP transaction). This read buffer captures data from the AP, presented as the result of a previous read, without initiating a new transaction.

19.4.6 SW-AP registers

Access to these registers are initiated when APnDP=1.

The access address of AP registers is comprised of two parts:

- The value A[3:2]
- The current value of the DP SELECT register

19.5 MCU debug component (MCUDBG)

The MCU debug component helps the debugger provide support for:

- Low-power modes
- Clock control for timers and watchdog during a breakpoint
- Control of the trace pins assignment

19.5.1 Debug support for low-power modes

To enter low-power mode, the instruction WFI or WFE must be executed. The MCU implements several low-power modes which can either deactivate the CPU clock or reduce the power of the CPU. The core does not allow FCLK or HCLK to be turned off during a debug session. As these are required for the debugger connection, during a debug, they must remain active. The MCU integrates special means to allow the user to debug software in low-power modes.

For this, the debugger host must first set some debug configuration registers to change the low-power mode behavior.

- In Sleep mode, DBG_SLEEP bit of DBGMCU_CR register must be previously set by the debugger. This will feed HCLK with the same clock that is provided to FCLK (system clock previously configured by the software).
- In Stop mode, the bit DBG_STOP must be previously set by the debugger. This will enable the internal oscillator clock to feed FCLK and HCLK in STOP mode.

19.5.2 Debug MCU configuration register

This register allows the configuration of the MCU under DEBUG.

This concerns:

- Low-power mode support
- Timer and watchdog counter support
- Trace pin assignment

This DBGMCU_CR is mapped on the external APB bus at base address 0x4001 3404. It is asynchronously reset by the PORESET (and not the system reset). It can be written by the debugger under system reset.

If the debugger host does not support these features, it is still possible for the user software to write to these registers.

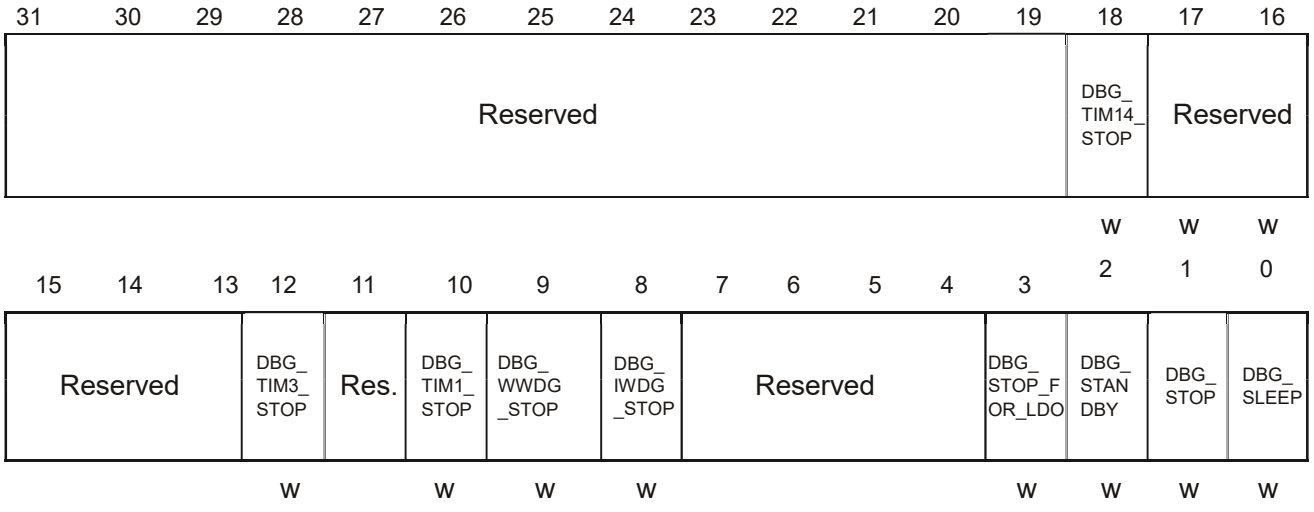
19.6 DBG register description

Table 63. Overview of DBG registers

Offset	Acronym	Register Name	Reset	Section
0x00	DBG	DBG control register	0x00000000	Section 19.6.1

19.6.1 DBG control register (DBG_CR)

POR Reset: 0x0000 0000 (not reset by system reset)



Bit	Field	Type	Reset	Description
31:19	Reserved			always read as 0.
18	DBG_TIM14_ STOP	w	0x00	TIM14 counter stopped when core is halted 0: The involved Timer Counter still behaves normally 1: The involved Timer Counter is halted

(DBG)

Bit	Field	Type	Reset	Description
17:13	Reserved			always read as 0
12	DBG_TIM3_STOP	w	0x00	TIM3 counter stopped when core is halted 0: The involved Timer Counter still behaves normally 1: The involved Timer Counter is halted
11	Reserved			always read as 0
10	DBG_TIM1_STOP	w	0x00	TIM1 counter stopped when core is halted 0: The involved Timer Counter is fed even if the core is halted 1: The involved Timer Counter is stopped when the core is halted
9	DBG_WWDG_STOP	w	0x00	Debug window watchdog stopped when core is halted 0: The window watchdog counter clock continues even if the core is halted 1: The window watchdog counter clock is stopped when the core is halted
8	DBG_IWDG_STOP	w	0x00	Debug independent watchdog stopped when core is halted 0: The watchdog counter clock continues even if the core is halted 1: The watchdog counter clock is stopped when the core is halted
7:4	Reserved			always read as 0
3	DBG_STOP_FOR_LDO	w	0x00	Debug Stop mode STOP LDO
2	DBG_STANDBY	w	0x00	Debug Standby mode 0: (FCLK = Off, HCLK = Off) The whole digital part is unpowered. From software point of view, exiting from Standby is identical than fetching reset vector (except a few status bits indicated that the MCU is exiting Standby) 1: (FCLK= Off, HCLK=Off) In this case, the digital part is not unpowered and FCLK and HCLK are provided by the internal RL oscillator which remains active. In addition, the MCU generates a system reset during Standby mode so that exiting from Standby is identical than fetching from reset.

(DBG)

Bit	Field	Type	Reset	Description
1	DBG_STOP	w	0x00	Debug Stop mode 0: (FCLK = Off, HCLK = Off) In STOP mode, the clock controller disables all clocks (including HCLK and FCLK). When exiting from STOP mode, the clock configuration is identical to the one after RESET (HSI clock divided by 4 is used as the PLL input clock). Consequently, the software does not have to reprogram the clock controller. When exiting STOP mode, the software must reprogram the clock controller, if HSE clock is used as the PLL input clock. 1: (FCLK=Off, HCLK=Off) In this case, when entering STOP mode, FCLK and HCLK are clocked by the internal oscillator. When exiting STOP mode, the software must reprogram the clock controller.
0	DBG_SLEEP	w	0x00	Debug Sleep mode 0: (FCLK = Off, HCLK = Off) In Sleep mode, FCLK is clocked by the system clock as previously configured while HCLK is disabled. In Sleep mode, the clock controller configuration is not reset. Consequently, when exiting from Sleep mode, the software does not need to reconfigure the clock controller. 1: (FCLK = On, HCLK = On) In this case, when entering Sleep mode, HCLK is fed by the same clock that is provided to FCLK (system clock as previously configured).

20 Revision History

Revision History

Table 64. Revision history

Date	Revision	Changes
2020/08/12	Rev1.00	Initial release.