

N32G032 series

32-bit ARM Cortex[®]-M0 microcontroller

User manual

Contents

1 Abbreviations in the text	28
1.1 List of abbreviations for registers	28
1.2 Available peripherals	28
2 Memory and bus architecture	29
2.1 System architecture	29
2.1.1 Bus architecture	29
2.1.2 Bus address mapping	31
2.1.3 Boot management	34
2.2 Memory system	35
2.2.1 FLASH specification	35
2.2.2 SRAM	46
2.2.3 FLASH register description	47
3 Power control (PWR)	53
3.1 General description	53
3.1.1 Power supply.....	54
3.1.2 Power supply supervisor	55
3.2 Power modes	56
3.2.1 LPRUN mode	58
3.2.2 SLEEP mode.....	58
3.2.3 STOP mode	59
3.2.4 PD mode	59
3.3 Debug support	60
3.3.1 Low power mode debug support.....	60
3.3.2 Peripheral debug support	60
3.4 PWR registers.....	60
3.4.1 PWR register overview	60
3.4.2 Power control register (PWR_CTRL).....	61
3.4.3 Power control status register (PWR_CTRLSTS).....	62
3.4.4 Power control register 2 (PWR_CTRL2).....	63
3.4.5 Power control register 3 (PWR_CTRL3).....	64
3.4.6 Power control register 4 (PWR_CTRL4).....	64
3.4.7 Power control register 5 (PWR_CTRL5).....	65
3.4.8 Power control register 6 (PWR_CTRL6).....	66
3.4.9 Debug control register (DBG_CTRL)	66
4 Reset and clock control (RCC).....	69
4.1 Reset Control Unit.....	69
4.1.1 Power reset.....	69
4.1.2 System reset	69
4.2 Clock control unit.....	70

4.2.1	Clock Tree Diagram.....	72
4.2.2	HSE clock	73
4.2.3	HSI clock	74
4.2.4	PLL clock.....	74
4.2.5	LSE clock.....	75
4.2.6	LSI clock.....	75
4.2.7	LSI Calibration	75
4.2.8	System clock (SYSCLK) selection	76
4.2.9	Clock security system (CLKSS)	76
4.2.10	RTC clock	76
4.2.11	Watchdog clock.....	76
4.2.12	LPUART clock	77
4.2.13	LPTIME clock	77
4.2.14	Clock output(MCO).....	77
4.3	RCC registers	78
4.3.1	RCC register overview.....	78
4.3.2	Clock control register (RCC_CTRL).....	79
4.3.3	Clock configuration register (RCC_CFG)	80
4.3.4	Clock interrupt register (RCC_CLKINT).....	83
4.3.5	APB2 peripheral reset register (RCC_APB2PRST)	85
4.3.6	APB1 peripheral reset register (RCC_APB1PRST)	87
4.3.7	AHB peripheral clock enable register (RCC_AHBCLKEN).....	89
4.3.8	APB2 peripheral clock enable register (RCC_APB2PCLKEN).....	90
4.3.9	APB1 peripheral clock enable register (RCC_APB1PCLKEN).....	92
4.3.10	Low speed clock control register (RCC_LSCTRL).....	94
4.3.11	Control/status register (RCC_CTRLSTS)	95
4.3.12	AHB peripheral reset register (RCC_AHBPRST).....	97
4.3.13	Clock configuration register 2(RCC_CFG2)	98
4.3.14	EMC control register 3 (RCC_EMCCTRL).....	100
5	GPIO and AFIO	104
5.1	Summary.....	104
5.2	Function description	105
5.2.1	I/O mode configuration.....	105
5.2.2	Status after reset.....	109
5.2.3	Individual bit setting and bit clearing.....	110
5.2.4	External interrupt /wakeup line.....	110
5.2.5	Alternate function	110
5.2.6	I/O configuration of peripherals.....	124
5.2.7	GPIO Locking mechanism.....	126
5.3	GPIO Registers	126
5.3.1	GPIO register overview	127
5.3.2	GPIO port mode description register (GPIOx_PMODE).....	128
5.3.3	GPIO port type definition (GPIOx_POTYPE)	128
5.3.4	GPIO slew rate configuration register (GPIOx_SR).....	129

5.3.5	GPIO port pull-up/pull-down register (GPIOx_PUPD).....	129
5.3.6	GPIO port input data register (GPIOx_PID).....	130
5.3.7	GPIO port output data register (GPIOx_POD).....	130
5.3.8	GPIO port bit set/clear register (GPIOx_PBSC).....	131
5.3.9	GPIO port configuration lock register (GPIOx_PLOCK).....	132
5.3.10	GPIO alternate function low register (GPIOx_AFL).....	133
5.3.11	GPIO alternate function high register (GPIOx_AFH).....	133
5.3.12	GPIO port bit clear register (GPIOx_PBC).....	134
5.3.13	GPIO driver strength configuration register (GPIOx_DS).....	135
5.4	AFIO Registers.....	135
5.4.1	AFIO register overview.....	135
5.4.2	AFIO configuration register (AFIO_CFG).....	136
5.4.3	AFIO external interrupt configuration register 1 (AFIO_EXTI_CFG1).....	137
5.4.4	AFIO external interrupt configuration register 2 (AFIO_EXTI_CFG2).....	138
5.4.5	AFIO external interrupt configuration register 3 (AFIO_EXTI_CFG3).....	138
5.4.6	AFIO external interrupt configuration register 4 (AFIO_EXTI_CFG4).....	139
6	Interrupts and events.....	141
6.1	Nested vectored interrupt controller.....	141
6.1.1	SysTick calibration value register.....	141
6.1.2	Interrupt and exception vectors.....	141
6.2	Extended Interrupt/Event Controller (EXTI).....	143
6.2.1	Introduction.....	143
6.2.2	Main features.....	143
6.2.3	Functional description.....	144
6.2.4	EXTI line mapping.....	145
6.3	EXTI Registers.....	147
6.3.1	EXTI register overview.....	147
6.3.2	Interrupt mask register(EXTI_IMASK).....	147
6.3.3	Event mask register(EXTI_EMASK).....	148
6.3.4	Rising edge trigger selection register(EXTI_RT_CFG).....	148
6.3.5	Falling edge trigger selection register(EXTI_FT_CFG).....	148
6.3.6	Software interrupt enable register(EXTI_SWIE).....	149
6.3.7	Interrupt request pending register(EXTI_PEND).....	149
6.3.8	RTC Timestamp trigger source selection register (EXTI_TS_SEL).....	150
7	DMA controller.....	151
7.1	Introduction.....	151
7.2	Main features.....	151
7.3	Block diagram.....	152
7.4	Function description.....	152
7.4.1	DMA operation.....	152
7.4.2	Channel priority and arbitration.....	153
7.4.3	DMA channels and number of transfers.....	153
7.4.4	Programmable data bit width, alignment and endians.....	153

7.4.5	Peripheral/Memory address incrementation	155
7.4.6	Channel configuration procedure.....	155
7.4.7	Flow control.....	156
7.4.8	Circular mode	156
7.4.9	Error management.....	156
7.4.10	Interrupt	157
7.4.11	DMA request mapping.....	157
7.5	DMA registers	158
7.5.1	DMA register overview.....	158
7.5.2	DMA interrupt status register (DMA_INTSTS)	160
7.5.3	DMA interrupt flag clear register (DMA_INTCLR).....	160
7.5.4	DMA channel x configuration register (DMA_CHCFGx).....	161
7.5.5	DMA channel x transfer number register (DMA_TXNUMx)	163
7.5.6	DMA channel x peripheral address register (DMA_PADDRx).....	163
7.5.7	DMA channel x memory address register (DMA_MADDRx)	164
7.5.8	DMA channel x channel request select register (DMA_CHSELx).....	164
8	CRC calculation unit	166
8.1	CRC introduction.....	166
8.2	CRC main features.....	166
8.2.1	CRC32 module	166
8.2.2	CRC16 module	166
8.3	CRC function description	167
8.3.1	CRC32	167
8.3.2	CRC16	167
8.4	CRC registers	168
8.4.1	CRC register overview.....	168
8.4.2	CRC32 data register (CRC_CRC32DAT).....	168
8.4.3	CRC32 independent data register (CRC_CRC32IDAT).....	168
8.4.4	CRC32 control register (CRC_CRC32CTRL).....	169
8.4.5	CRC16 control register (CRC_CRC16CTRL).....	169
8.4.6	CRC16 input data register (CRC_CRC16DAT)	170
8.4.7	CRC cyclic redundancy check code register (CRC_CRC16D)	170
8.4.8	LRC result register (CRC_LRC).....	171
9	Advanced-control timers (TIM1 and TIM8)	172
9.1	TIM1 and TIM8 introduction	172
9.2	Main features of TIM1 and TIM8	172
9.3	TIM1 and TIM8 function description	173
9.3.1	Time-base unit.....	173
9.3.2	Counter mode.....	174
9.3.3	Repetition counter	179
9.3.4	Clock selection	181
9.3.5	Capture/compare channels	185
9.3.6	Input capture mode	188

9.3.7	PWM input mode	189
9.3.8	Forced output mode	190
9.3.9	Output compare mode.....	190
9.3.10	PWM mode.....	191
9.3.11	One-pulse mode	194
9.3.12	Clearing the OCxREF signal on an external event	196
9.3.13	Complementary outputs with dead-time insertion	196
9.3.14	Break function.....	198
9.3.15	Debug mode.....	200
9.3.16	TIMx and external trigger synchronization	200
9.3.17	Timer synchronization	204
9.3.18	6-step PWM generation	204
9.3.19	Encoder interface mode	205
9.3.20	Interfacing with Hall sensor	208
9.4	TIMx register description(x=1, 8).....	210
9.4.1	Register Overview	210
9.4.2	Control register 1 (TIMx_CTRL1)	211
9.4.3	Control register 2 (TIMx_CTRL2)	213
9.4.4	Slave mode control register (TIMx_SMCTRL).....	215
9.4.5	DMA/Interrupt enable registers (TIMx_DINTEN).....	217
9.4.6	Status registers (TIMx_STS).....	219
9.4.7	Event generation registers (TIMx_EVTGEN).....	220
9.4.8	Capture/compare mode register 1 (TIMx_CCMOD1)	222
9.4.9	Capture/compare mode register 2 (TIMx_CCMOD2)	225
9.4.10	Capture/compare enable registers (TIMx_CCEN).....	226
9.4.11	Counters (TIMx_CNT).....	229
9.4.12	Prescaler (TIMx_PSC)	229
9.4.13	Auto-reload register (TIMx_AR)	229
9.4.14	Repeat count registers (TIMx_REPCNT)	230
9.4.15	Capture/compare register 1 (TIMx_CCDAT1).....	230
9.4.16	Capture/compare register 2 (TIMx_CCDAT2).....	231
9.4.17	Capture/compare register 3 (TIMx_CCDAT3).....	231
9.4.18	Capture/compare register 4 (TIMx_CCDAT4).....	232
9.4.19	Break and Dead-time registers (TIMx_BKDT).....	232
9.4.20	DMA Control register (TIMx_DCTRL)	234
9.4.21	DMA transfer buffer register (TIMx_DADDR)	235
9.4.22	Capture/compare mode registers 3(TIMx_CCMOD3)	235
9.4.23	Capture/compare register 5 (TIMx_CCDAT5).....	236
9.4.24	Capture/compare register 6 (TIMx_CCDAT6).....	236
10	General-purpose timers (TIM3 and TIM4)	238
10.1	General-purpose timers introduction	238
10.2	Main features of General-purpose timers	238
10.3	General-purpose timers description	239
10.3.1	Time-base unit.....	239

10.3.2	Counter mode.....	240
10.3.3	Clock selection.....	245
10.3.4	Capture/compare channels.....	249
10.3.5	Input capture mode.....	251
10.3.6	PWM input mode.....	252
10.3.7	Forced output mode.....	253
10.3.8	Output compare mode.....	253
10.3.9	PWM mode.....	255
10.3.10	One-pulse mode.....	257
10.3.11	Clearing the OCxREF signal on an external event.....	259
10.3.12	Debug mode.....	259
10.3.13	TIMx and external trigger synchronization.....	260
10.3.14	Timer synchronization.....	260
10.3.15	Encoder interface mode.....	264
10.3.16	Interfacing with Hall sensor.....	266
10.4	TIMx register description(x=3 and 4).....	266
10.4.1	Register Overview.....	266
10.4.2	Control register 1 (TIMx_CTRL1).....	268
10.4.3	Control register 2 (TIMx_CTRL2).....	269
10.4.4	Slave mode control register (TIMx_SMCTRL).....	270
10.4.5	DMA/Interrupt enable registers (TIMx_DINTEN).....	273
10.4.6	Status registers (TIMx_STS).....	274
10.4.7	Event generation registers (TIMx_EVTGEN).....	275
10.4.8	Capture/compare mode register 1 (TIMx_CCMOD1).....	276
10.4.9	Capture/compare mode register 2 (TIMx_CCMOD2).....	280
10.4.10	Capture/compare enable registers (TIMx_CCEN).....	281
10.4.11	Counters (TIMx_CNT).....	282
10.4.12	Prescaler (TIMx_PSC).....	283
10.4.13	Auto-reload register (TIMx_AR).....	283
10.4.14	Capture/compare register 1 (TIMx_CCDAT1).....	283
10.4.15	Capture/compare register 2 (TIMx_CCDAT2).....	284
10.4.16	Capture/compare register 3 (TIMx_CCDAT3).....	284
10.4.17	Capture/compare register 4 (TIMx_CCDAT4).....	285
10.4.18	DMA Control register (TIMx_DCTRL).....	285
10.4.19	DMA transfer buffer register (TIMx_DADDR).....	286
11	Basic timers (TIM6).....	288
11.1	Basic timers introduction.....	288
11.2	Main features of Basic timers.....	288
11.3	Basic timers description.....	288
11.3.1	Time-base unit.....	288
11.3.2	Counter mode.....	289
11.3.3	Clock selection.....	292
11.3.4	Debug mode.....	292
11.4	TIMx register description(x=6).....	292

11.4.1	Register overview	292
11.4.2	Control Register 1 (TIMx_CTRL1).....	293
11.4.3	DMA/Interrupt Enable Registers (TIMx_DINTEN)	294
11.4.4	Status Registers (TIMx_STS).....	295
11.4.5	Event Generation registers (TIMx_EVTGEN).....	295
11.4.6	Counters (TIMx_CNT).....	296
11.4.7	Prescaler (TIMx_PSC).....	296
11.4.8	Automatic reload register (TIMx_AR)	296
12	Low Power Timer (LPTIM)	298
12.1	Introduction	298
12.2	Main features.....	298
12.3	Function description	299
12.3.1	Block diagram.....	299
12.3.2	LPTIM Reset and Clock	299
12.3.3	Glitch filter	300
12.3.4	Prescaler.....	300
12.3.5	Trigger multiplexer	301
12.3.6	Operating mode	302
12.3.7	Timeout function.....	303
12.3.8	Waveform generation.....	304
12.3.9	Register update	305
12.3.10	Counter mode.....	306
12.3.11	Timer enable	306
12.3.12	Encoder mode	306
12.3.13	Non-orthogonal encoder mode	308
12.3.14	LPTIM interrupts.....	309
12.4	LPTIM registers	310
12.4.1	LPTIM register overview.....	310
12.4.2	LPTIM interrupt and status register (LPTIM_INTSTS).....	310
12.4.3	LPTIM interrupt clear register (LPTIM_INTCLR)	311
12.4.4	LPTIM interrupt enable register (LPTIM_INTEN).....	312
12.4.5	LPTIM configuration register (LPTIM_CFG).....	313
12.4.6	LPTIM control register (LPTIM_CTRL)	315
12.4.7	LPTIM compare register (LPTIM_COMP).....	316
12.4.8	LPTIM auto-reload register (LPTIM_ARR).....	317
12.4.9	LPTIM counter register (LPTIM_CNT).....	317
13	Independent watchdog (IWDG)	318
13.1	Introduction	318
13.2	Main features.....	318
13.3	Function description	319
13.3.1	Register access protection.....	319
13.3.2	Debug mode.....	320
13.4	User interface	320

13.4.1	Operating process	320
13.5	IWDG registers.....	321
13.5.1	IWDG register overview.....	321
13.5.2	IWDG key register (IWDG_KEY)	321
13.5.3	IWDG pre-scaler register (IWDG_PREDIV).....	321
13.5.4	IWDG reload register (IWDG_RELV)	322
13.5.5	IWDG status register (IWDG_STS).....	323
14	Window watchdog (WWDG).....	324
14.1	Introduction	324
14.2	Main features.....	324
14.3	Function description	324
14.4	Timing for refresh watchdog and interrupt generation	325
14.5	Debug mode	326
14.6	User interface	326
14.6.1	WWDG configuration flow	326
14.7	WWDG registers	326
14.7.1	WWDG register overview	326
14.7.2	WWDG control register (WWDG_CTRL).....	327
14.7.3	WWDG config register (WWDG_CFG)	327
14.7.4	WWDG status register (WWDG_STS)	328
15	Analog to digital conversion (ADC).....	329
15.1	Introduction	329
15.2	Main features	329
15.3	Function Description.....	329
15.3.1	ADC clock	331
15.3.2	ADC switch control.....	332
15.3.3	Channel selection	332
15.3.4	Internal channel	332
15.3.5	Single conversion mode.....	332
15.3.6	Continuous conversion mode.....	333
15.3.7	Timing diagram.....	333
15.3.8	Analog watchdog	333
15.3.9	Scan mode	334
15.3.10	Injection channel management	334
15.3.11	Discontinuous mode	335
15.4	Data aligned	336
15.5	Programmable channel sampling time	336
15.6	Externally triggered conversion	337
15.7	DMA requests	337
15.8	Temperature sensor	338
15.8.1	Temperature sensor using flow	338
15.9	ADC interrupt.....	339
15.10	ADC registers.....	339

15.10.1	ADC register overview	339
15.10.2	ADC status register (ADC_STS).....	340
15.10.3	ADC control register 1 (ADC_CTRL1)	342
15.10.4	ADC control register 2 (ADC_CTRL2)	344
15.10.5	ADC sampling time register 1 (ADC_SAMPT1)	345
15.10.6	ADC sampling time register 2 (ADC_SAMPT2)	346
15.10.7	ADC sampling time register 3 (ADC_SAMPT3)	347
15.10.8	ADC injected channel data offset register x (ADC_JOFFSETx) (x=1..4).....	347
15.10.9	ADC watchdog high threshold register (ADC_WDGHIGH)	348
15.10.10	ADC watchdog low threshold register (ADC_WDGLow).....	348
15.10.11	ADC regular sequence register 1 (ADC_RSEQ1)	348
15.10.12	ADC regular sequence register 2 (ADC_RSEQ2).....	349
15.10.13	ADC regular sequence register 3 (ADC_RSEQ3)	349
15.10.14	ADC Injection sequence register (ADC_JSEQ).....	350
15.10.15	ADC injection data register x (ADC_JDATx) (x= 1..4).....	351
15.10.16	ADC regulars data register (ADC_DAT)	351
15.10.17	ADC control register 3 (ADC_CTRL3).....	351
16	Comparator (COMP).....	353
16.1	COMP system connection block diagram	353
16.2	COMP features.....	354
16.3	COMP configuration process	355
16.4	COMP working mode.....	356
16.4.1	Window mode.....	356
16.4.2	Independent comparator	356
16.5	Comparator interconnection	356
16.6	Interrupt	358
16.7	COMP register.....	358
16.7.1	COMP register overview	358
16.7.2	COMP interrupt Enable register (COMP_INTEN).....	359
16.7.3	COMP interrupt register (COMP_INTSTS).....	360
16.7.4	COMP window mode enable register(COMP_WINMODE).....	360
16.7.5	COMP lock register(COMP_LOCK).....	361
16.7.6	COMP1 control register (COMP1_CTRL).....	361
16.7.7	COMP1 filter control register (COMP1_FILC).....	363
16.7.8	COMP1 filter frequency division register (COMP1_FILP).....	363
16.7.9	COMP2 control register (COMP2_CTRL).....	364
16.7.10	COMP2 filter control register (COMP2_FILC).....	366
16.7.11	COMP2 filter frequency division register (COMP2_FILP).....	366
16.7.12	COMP3 control register (COMP3_CTRL).....	367
16.7.13	COMP3 filter control register (COMP3_FILC).....	368
16.7.14	COMP3 filter frequency division register (COMP3_FILP).....	369
16.7.15	COMP reference input compare voltage register (COMP_INVREF).....	369
17	I²C interface	371

17.1 Introduction	371
17.2 Main features.....	371
17.3 Function description	372
17.3.1 SDA and SCL line control	372
17.3.2 Software communication process	373
17.3.3 Error conditions description.....	382
17.3.4 DMA application	383
17.3.5 Packet error check.....	384
17.3.6 SMBus	385
17.4 Debug mode	387
17.5 Interrupt request	387
17.6 I2C registers	388
17.6.1 I2C register overview	388
17.6.2 I2C Control register 1 (I2C_CTRL1)	389
17.6.3 I2C Control register 2 (I2C_CTRL2)	391
17.6.4 I2C Own address register 1 (I2C_OADDR1).....	392
17.6.5 I2C Own address register 2 (I2C_OADDR2).....	393
17.6.6 I2C Data register (I2C_DAT)	393
17.6.7 I2C Status register 1 (I2C_STS1)	393
17.6.8 I2C Status register 2 (I2C_STS2)	397
17.6.9 I2C Clock control register (I2C_CLKCTRL).....	398
17.6.10 I2C Rise time register (I2C_TMRISE).....	399
18 Universal synchronous asynchronous receiver transmitter (USART).....	400
18.1 Introduction	400
18.2 Main features.....	400
18.3 Functional block diagram.....	401
18.4 Function description	401
18.4.1 USART frame format.....	402
18.4.2 Transmitter.....	403
18.4.3 Receiver	405
18.4.4 Generation of fractional baud rate	408
18.4.5 Receiver's tolerance clock deviation	409
18.4.6 Parity control	410
18.4.7 DMA application	411
18.4.8 Hardware flow control.....	413
18.4.9 Multiprocessor communication	414
18.4.10 Synchronous mode.....	416
18.4.11 Single-wire half-duplex mode	418
18.4.12 IrDA SIR ENDEC mode.....	419
18.4.13 LIN mode.....	420
18.4.14 Smartcard mode (ISO7816).....	422
18.5 Interrupt request	424
18.6 Mode support.....	425
18.7 USART register	425

18.7.1	USART register overview	425
18.7.2	USART Status register (USART_STS).....	426
18.7.3	USART Data register (USART_DAT).....	428
18.7.4	USART Baud rate register (USART_BRCF).....	428
18.7.5	USART control register 1 register (USART_CTRL1)	429
18.7.6	USART control register 2 register (USART_CTRL2).....	430
18.7.7	USART control register 3 register (USART_CTRL3).....	432
18.7.8	USART guard time and prescaler register (USART_GTP).....	433
19	Low power universal asynchronous receiver transmitter (LPUART).....	435
19.1	Introduction	435
19.2	Main features.....	435
19.3	Functional block diagram.....	437
19.4	Function description	437
19.4.1	LPUART frame format	438
19.4.2	Transmitter.....	438
19.4.3	Receiver	440
19.4.4	Fractional baud rate generation.....	442
19.4.5	Parity control	443
19.4.6	DMA application	444
19.4.7	Hardware flow control.....	446
19.4.8	Low power wake up.....	447
19.5	Interrupt request	448
19.6	LPUART registers.....	448
19.6.1	LPUART register overview	448
19.6.2	LPUART status register (LPUART_STS)	449
19.6.3	LPUART interrupt enable register (LPUART_INTEN)	450
19.6.4	LPUART control register (LPUART_CTRL).....	450
19.6.5	LPUART baud rate configuration register 1 (LPUART_BRCFG1)	452
19.6.6	LPUART data register (LPUART_DAT).....	452
19.6.7	LPUART baud rate configuration register 2 (LPUART_BRCFG2)	453
19.6.8	LPUART wake up data register (LPUART_WUDAT).....	453
20	Serial peripheral interface/Inter-IC Sound (SPI/ I2S)	454
20.1	SPI introduction	454
20.2	SPI and I2S main features	454
20.2.1	SPI features.....	454
20.2.2	I2S features.....	455
20.3	SPI function description.....	456
20.3.1	General description.....	456
20.3.2	SPI work mode	459
20.3.3	Status flag	465
20.3.4	Turn off the SPI	466
20.3.5	SPI communication using DMA.....	467
20.3.6	CRC calculation.....	468

20.3.7	Error flag.....	468
20.3.8	SPI interrupt.....	469
20.4	I2S function description.....	470
20.4.1	Supported audio protocols	471
20.4.2	Clock generator.....	477
20.4.3	I2S send and receive sequence.....	478
20.4.4	Status flag	480
20.4.5	Error flag.....	481
20.4.6	I2S interrupt.....	481
20.4.7	DMA function.....	482
20.5	SPI and I2S register description	482
20.5.1	SPI register overview.....	482
20.5.2	SPI control register 1 (SPI_CTRL1) (not used in I2S mode)	482
20.5.3	SPI control register 2 (SPI_CTRL2).....	485
20.5.4	SPI status register (SPI_STS)	485
20.5.5	SPI data register (SPI_DAT).....	486
20.5.6	SPI CRC polynomial register (SPI_CRCPOLY) (not used in I ² S mode).....	487
20.5.7	SPI RX CRC register (SPI_CRCRDAT) (not used in I ² S mode).....	487
20.5.8	SPI TX CRC register (SPI_CRCTDAT)	488
20.5.9	SPI_I ² S configuration register (SPI_I2SCFG)	488
20.5.10	SPI_I ² S prescaler register (SPI_I2SPREDIV)	490
21	Real-time clock (RTC)	491
21.1	Description.....	491
21.2	Specification	491
21.3	RTC function description.....	492
21.3.1	RTC block diagram.....	492
21.3.2	GPIOs of RTC.....	493
21.3.3	RTC register write protection	493
21.3.4	RTC clock and prescaler	493
21.3.5	RTC calendar	494
21.3.6	Calendar initialization and configuration.....	494
21.3.7	Calendar reading.....	495
21.3.8	Calibration clock output.....	495
21.3.9	Programmable alarms	496
21.3.10	Alarm configuration.....	496
21.3.11	Alarm output.....	496
21.3.12	Periodic automatic wakeup.....	496
21.3.13	Wakeup timer configuration	497
21.3.14	Timestamp function	497
21.3.15	Tamper detection	498
21.3.16	Daylight saving time configuration	498
21.3.17	RTC sub-second register shift.....	498
21.3.18	RTC digital clock precision calibration	499
21.3.19	RTC low power mode.....	500

21.4 RTC Registers.....	500
21.4.1 RTC register overview	500
21.4.2 RTC Calendar Time Register (RTC_TSH)	501
21.4.3 RTC Calendar Date Register (RTC_DATE)	502
21.4.4 RTC Control Register (RTC_CTRL)	502
21.4.5 RTC Initial Status Register (RTC_INITSTS)	505
21.4.6 RTC Prescaler Register (RTC_PRE)	507
21.4.7 RTC Wakeup Timer Register (RTC_WKUPT).....	507
21.4.8 RTC Alarm A Register (RTC_ALARM_A).....	508
21.4.9 RTC Alarm B Register (RTC_ALARM_B).....	509
21.4.10 RTC Write Protection register (RTC_WRP).....	510
21.4.11 RTC Sub-second Register (RTC_SUBS).....	510
21.4.12 RTC Shift Control Register (RTC_SCTRL).....	510
21.4.13 RTC Timestamp Time Register (RTC_TST)	511
21.4.14 RTC Timestamp Date Register (RTC_TSD).....	512
21.4.15 RTC Timestamp Sub-second Register (RTC_TSSS).....	512
21.4.16 RTC Calibration Register (RTC_CALIB).....	513
21.4.17 RTC Tamper Configuration Register (RTC_TMPCFG)	513
21.4.18 RTC Alarm A sub-second register (RTC_ALRMAS).....	515
21.4.19 RTC Alarm B sub-second register (RTC_ALRMBSS).....	516
22 Operational amplifier (OPAMP)	518
22.1 OPAMP features.....	518
22.1.1 OPAMP function description	519
22.2 OPAMP working mode.....	519
22.2.1 OPAMP external amplification mode	519
22.2.2 OPAMP follower mode.....	520
22.2.3 OPAMP internal programmable gain (PGA) mode	521
22.2.4 OPAMP independent write protection	522
22.2.5 OPAMP TIMER controls the switching mode.....	522
22.3 OPAMP registers	522
22.3.1 OPAMP registers overview.....	522
22.3.2 OPAMP Control Status Register (OPAMP_CS)	524
22.3.3 OPAMP Lock Register (OPAMP_LOCK).....	525
23 Beeper	526
23.1 Introduction	526
23.2 Function description	526
23.3 Beeper registers.....	526
23.3.1 Beeper register overview	526
23.3.2 Beeper control register (BEEPER_CTRL)	526
24 Arithmetic units (HDIV and SQRT).....	528
24.1 Introduction to HDIV and SQRT	528
24.2 HDIV and SQRT function description	528

24.3 HDIV registers	528
24.3.1 HDIV register overview.....	528
24.3.2 HDIV control status register (HDIV_CTRLSTS).....	529
24.3.3 HDIV dividend register (HDIV_DIVIDEND).....	529
24.3.4 HDIV divisor register (HDIV_DIVISOR).....	530
24.3.5 HDIV quotient register (HDIV_QUOTIENT).....	530
24.3.6 HDIV remainder register (HDIV_REMAINDER)	530
24.3.7 HDIV divide by zero register (HDIV_DIVBY0).....	531
24.4 Sqrt registers.....	531
24.4.1 Sqrt register overview.....	531
24.4.2 Sqrt control status register (Sqrt_CTRLSTS).....	531
24.4.3 Sqrt Radicand register (Sqrt_RADICAND).....	532
24.4.4 Sqrt square root register (Sqrt_ROOT)	532
25 Controller area network (CAN).....	534
25.1 Introduction to CAN.....	534
25.2 Main features of CAN	534
25.3 CAN overall introduction	534
25.3.1 CAN module.....	535
25.3.2 CAN working mode.....	535
25.3.3 Send mailbox	537
25.3.4 Receiving filter	537
25.3.5 Receive FIFO.....	537
25.3.6 CAN Test mode	538
25.3.7 CAN Debugging mode	541
25.4 CAN function description.....	541
25.4.1 Send processing	541
25.4.2 Time triggered communication mode	542
25.4.3 Non-automatic retransmission mode	542
25.4.4 Receiving management.....	542
25.4.5 Identifier filtering.....	544
25.4.6 Message storage.....	547
25.4.7 Bit time characteristic	548
25.5 CAN interrupt.....	551
25.5.1 Error management	552
25.5.2 Bus-Off recovery	552
25.6 CAN Configuration Flow.....	552
25.7 CAN Register File.....	554
25.7.1 Register Description	554
25.7.2 CAN register address overview	554
25.7.3 CAN control and status register.....	558
25.7.4 CAN mailbox register.....	568
25.7.5 CAN filter register	574
26 Cryptographic Algorithm Hardware Acceleration Engine (SAC)	579

27 Debug support (DBG)	580
27.1 Overview	580
27.2 SWD function	581
27.2.1 Pin assignment	581
28 Unique device serial number (UID)	582
28.1 Introduction	582
28.2 UID register	582
28.3 UCID register	582
28.4 DBGMCU_ID register	582
29 Version history	584
30 Notice	585

List of Tables

Table 2-1 List of peripheral register addresses	32
Table 2-2 List of boot mode.....	34
Table 2-3 Flash bus address list	35
Table 2-4 Option byte list	39
Table 2-5 Read protection configuration list.....	41
Table 2-6 Flash read-write-erase ⁽¹⁾ permission control table	42
Table 2-7 FLASH register overview	47
Table 3-1 Power modes.....	56
Table 3-2 Peripheral running status	57
Table 3-3 PWR register overview	60
Table 4-1 RCC register overview.....	78
Table 5-1 I/O port configuration table	105
Table 5-2 I/O List of functional features of the pin.....	106
Table 5-3 I/O List of functional features of the pin.....	111
Table 5-4 TIM1 alternate function I/O remapping.....	111
Table 5-5 TIM8 alternate function I/O remapping.....	111
Table 5-6 TIM3 alternate function I/O remapping.....	112
Table 5-7 TIM4 alternate function I/O remapping.....	113
Table 5-8 LPTIM alternate function I/O remapping	114
Table 5-9 CAN alternate function I/O remapping.....	114
Table 5-10 USART1 alternate function I/O remapping	115
Table 5-11 USART2 alternate function I/O remapping	115
Table 5-12 UART5 alternate function I/O remapping	116
Table 5-13 UART6 alternate function I/O remapping	116
Table 5-14 LPUART1 alternate function I/O remapping.....	116
Table 5-15 LPUART2 alternate function I/O remapping.....	117
Table 5-16 I2C1 alternate function I/O remapping	117
Table 5-17 I2C2 alternate function I/O remapping	118
Table 5-18 SPI1/I2S1 alternate function I/O remapping.....	119
Table 5-19 SPI2 alternate function I/O remapping	119

Table 5-20 SPI3 alternate function I/O remapping	120
Table 5-21 COMP1 alternate function I/O remapping.....	120
Table 5-22 COMP2 alternate function I/O remapping.....	120
Table 5-23 COMP3 alternate function I/O remapping.....	120
Table 5-24 BEEPER1 alternate function I/O remapping	121
Table 5-25 BEEPER2 alternate function I/O remapping	121
Table 5-26 EVENTOUT alternate function I/O remapping.....	121
Table 5-27 RTC alternate function I/O remapping.....	122
Table 5-28 PVD alternate function I/O remapping.....	122
Table 5-29 RCC alternate function I/O remapping	122
Table 5-30 OSC_IN/OSC_OUT alternate function I/O remapping	122
Table 5-31 OSC32 alternate function remapping.....	122
Table 5-32 OSC alternate function remapping.....	123
Table 5-33 ADC external trigger injection conversion alternate function remapping.....	123
Table 5-34 ADC external trigger regular conversion alternate function remapping	123
Table 5-35 TIM4_CH2 alternate function remapping.....	123
Table 5-36 IO Signal Dual Voltage Levels Configuration	124
Table 5-37 ADC	124
Table 5-38 PVD	124
Table 5-39 TIM1/TIM8.....	124
Table 5-40 TIM3 and LPTIM	124
Table 5-41 CAN.....	124
Table 5-42 USART	124
Table 5-43 LPUART	125
Table 5-44 I2C	125
Table 5-45 SPI	125
Table 5-46 COMP.....	125
Table 5-47 BEEPER	126
Table 5-48 Other.....	126
Table 5-49 GPIO register overview	127
Table 5-50 AFIO register overview	135
Table 6-1 Vector table	141

Table 6-2 EXTI register overview	147
Table 7-1 Programmable data width and endian operation (when PINC = MINC = 1).....	153
Table 7-2 Flow control table.....	156
Table 7-3 DMA interrupt request.....	157
Table 7-4 DMA request mapping.....	158
Table 7-5 DMA register overview	158
Table 8-1 CRC register overview.....	168
Table 9-1 Counting direction versus encoder signals.....	206
Table 9-2 Register overview	210
Table 9-3 TIMx internal trigger connection.....	217
Table 9-4 Output control bits of complementary OCx and OCxN channels with break function.....	228
Table 10-1 Counting direction versus encoder signals.....	264
Table 10-2 Register overview	266
Table 10-3 TIMx internal trigger connection.....	273
Table 10-4 Output control bits of standard OCx channel.....	282
Table 11-1 Register overview	292
Table 12-1 Pre-scaler division ratios.....	300
Table 12-2 6 trigger inputs corresponding to LPTIM_CFG.TRGSEL[2:0] bits	301
Table 13-1 IWDG counting maximum and minimum reset time.....	320
Table 13-2 IWDG register overview.....	321
Table 14-1 Maximum and minimum counting time of WWDG	326
Table 14-2 WWDG register overview	326
Table 15-1 ADC pins	330
Table 15-2 Analog watchdog channel selection.....	334
Table 15-3 Right-aligned data.....	336
Table 15-4 Left-aligned data.....	336
Table 15-5 ADC is used for external triggering of regular channels.....	337
Table 15-6 ADC is used for external triggering of injection channels	337
Table 15-7 ADC interrupt	339
Table 15-8 ADC register overview	339
Table 16-1 COMP register overview	358
Table 17-1 Comparison between SMBus and I2C.....	385

Table 17-2 I ² C interrupt request.....	387
Table 17-3 I2C register overview.....	388
Table 18-1 Stop bit configuration	403
Table 18-2 Data sampling for noise detection.....	408
Table 18-3 Error calculation when setting baud rate.....	409
Table 18-4 when DIV_Decimal = 0. Tolerance of USART receiver.....	410
Table 18-5 when DIV_Decimal != 0. Tolerance of USART receiver	410
Table 18-6 Frame format	410
Table 18-7 USART interrupt request	424
Table 18-8 USART mode setting ⁽¹⁾	425
Table 18-9 USART register overview.....	425
Table 19-1 Data sampling for noise detection.....	442
Table 19-2 Parity frame format.....	444
Table 19-3 LPUART interrupt requests	448
Table 19-4 LPUART register overview	448
Table 20-1 SPI interrupt request	469
Table 20-2 Use the standard 8MHz HSE clock to get accurate audio frequency.....	478
Table 20-3 I ² S interrupt request	481
Table 20-4 SPI register overview.....	482
Table 21-1 RTC feature support.....	491
Table 21-2 RTC register overview	500
Table 22-1 OPAMP register overview	522
Table 23-1 Beeper register overview	526
Table 24-1 HDIV register overview.....	528
Table 24-2 SQRT register overview.....	531
Table 25-1 Examples of filter numbers.....	546
Table 25-2 Send mailbox register list	547
Table 25-3 Receive mailbox register list.....	548
Table 25-4 CAN register overview	554
Table 27-1 Debug port pin	581
Table 28-1 DBGMCU_ID bit description.....	582

List of Figures

Figure 2-1 Bus architecture.....	30
Figure 2-2 Bus address map.....	32
Figure 3-1 Power supply block diagram.....	54
Figure 3-2 Power on reset/power down reset waveform.....	55
Figure 3-3 PVD threshold diagram.....	56
Figure 4-1 System reset generation.....	70
Figure 4-2 Clock Tree.....	72
Figure 4-3 HSE clock source.....	73
Figure 4-4 PLL clock configuration.....	75
Figure 5-1 Basic structure of I/O ports.....	105
Figure 5-2 Input floating / pull-up / pull-down configuration mode.....	107
Figure 5-3 Output mode.....	108
Figure 5-4 Alternate function mode.....	109
Figure 5-5 Analog function mode with high impedance.....	109
Figure 6-1 External interrupt/event controller block diagram.....	144
Figure 6-2 External interrupt generic I/O mapping.....	145
Figure 7-1 DMA block diagram.....	152
Figure 8-1 CRC calculation unit block diagram.....	167
Figure 9-1 Block diagram of TIM1 and TIM8.....	173
Figure 9-2 Counter timing diagram with prescaler division change from 1 to 4.....	174
Figure 9-3 Timing diagram of up-counting. The internal clock divider factor = $2/N$	175
Figure 9-4 Timing diagram of the up-counting, update event when ARPEN=0/1.....	176
Figure 9-5 Timing diagram of the down-counting, internal clock divided factor = $2/N$	177
Figure 9-6 Timing diagram of the Center-aligned, internal clock divided factor = $2/N$	178
Figure 9-7 A center-aligned sequence diagram that includes counter overflows and underflows (ARPEN = 1).....	178
Figure 9-8 Repeat count sequence diagram in down-counting mode.....	180
Figure 9-9 Repeat count sequence diagram in up-counting mode.....	181
Figure 9-10 Repeat count sequence diagram in center-aligned mode.....	181
Figure 9-11 Control circuit in normal mode, internal clock divided by 1.....	182
Figure 9-12 TI2 external clock connection example.....	183

Figure 9-13 Control circuit in external clock mode 1 184

Figure 9-14 External trigger input block diagram..... 184

Figure 9-15 Control circuit in external clock mode 2 185

Figure 9-16 Capture/compare channel (example: channel 1 input stage) 186

Figure 9-17 Capture/compare channel 1 main circuit..... 187

Figure 9-18 Output part of channelx (x= 1,2,3, take channel 1 as example) 187

Figure 9-19 Output part of channelx (x= 4)..... 188

Figure 9-20 PWM input mode timing..... 190

Figure 9-21 Output compare mode, toggle on OC1 191

Figure 9-22 Center-aligned PWM waveform (AR=8) 193

Figure 9-23 Edge-aligned PWM waveform (APR=8) 194

Figure 9-24 Example of One-pulse mode 195

Figure 9-25 Clearing the OCxREF of TIMx..... 196

Figure 9-26 Complementary output with dead-time insertion 197

Figure 9-27 Output behavior in response to a break 200

Figure 9-28 Control circuit in reset mode 201

Figure 9-29 Control circuit in Trigger mode..... 202

Figure 9-30 Control circuit in Gated mode..... 203

Figure 9-31 Control circuit in Trigger Mode + External Clock Mode2 204

Figure 9-32 6-step PWM generation, COM example (OSSR=1) 205

Figure 9-33 Example of counter operation in encoder interface mode 206

Figure 9-34 Encoder interface mode example with IC1FP1 polarity inverted 207

Figure 9-35 Example of Hall sensor interface 209

Figure 10-1 Block diagram of TIMx (x=3 and 4) 239

Figure 10-2 Counter timing diagram with prescaler division change from 1 to 4 240

Figure 10-3 Timing diagram of up-counting. The internal clock divider factor = 2/N 241

Figure 10-4 Timing diagram of the up-counting, update event when ARPEN=0/1 242

Figure 10-5 Timing diagram of the down-counting, internal clock divided factor = 2/N 243

Figure 10-6 Timing diagram of the Center-aligned, internal clock divided factor =2/N 244

Figure 10-7 A center-aligned sequence diagram that includes counter overflows and underflows (ARPEN = 1).... 244

Figure 10-8 Control circuit in normal mode, internal clock divided by 1..... 246

Figure 10-9 TI2 external clock connection example..... 247

Figure 10-10 Control circuit in external clock mode 1 248

Figure 10-11 External trigger input block diagram..... 248

Figure 10-12 Control circuit in external clock mode 2 249

Figure 10-13 Capture/compare channel (example: channel 1 input stage) 249

Figure 10-14 Capture/compare channel 1 main circuit..... 250

Figure 10-15 Output part of channelx (x = 1,2,3,4;take channel 4 as an example) 251

Figure 10-16 PWM input mode timing..... 253

Figure 10-17 Output compare mode, toggle on OC1 254

Figure 10-18 Center-aligned PWM waveform (AR=8)..... 256

Figure 10-19 Edge-aligned PWM waveform (APR=8) 257

Figure 10-20 Example of One-pulse mode 258

Figure 10-21 Control circuit in reset mode 259

Figure 10-22 Block diagram of timer interconnection..... 260

Figure 10-23 TIM3 gated by OC1REF of TIM1 261

Figure 10-24 TIM3 gated by enable signal of TIM1 262

Figure 10-25 Trigger TIM3 with an update of TIM1 263

Figure 10-26 Triggers timers 1 and 3 using the TII input of TIM1 264

Figure 10-27 Example of counter operation in encoder interface mode 265

Figure 11-1 Block diagram of TIMx (x = 6) 288

Figure 11-2 Counter timing diagram with prescaler division change from 1 to 4 289

Figure 11-3 Timing diagram of up-counting. The internal clock divider factor = 2/N 290

Figure 11-4 Timing diagram of the up-counting, update event when ARPEN=0/1 291

Figure 11-5 Control circuit in normal mode, internal clock divided by 1 292

Figure 12-1 LPTIM Diagram..... 299

Figure 12-2 Glitch filter timing diagram..... 300

Figure 12-3 LPTIM output waveform, Continuous counting mode configuration 302

Figure 12-4 PTIM output waveform, single counting mode configuration 303

Figure 12-5 LPTIM output waveform, Single counting mode configuration and Set-once mode activated 303

Figure 12-6 Waveform generation 305

Figure 12-7 Encoder mode counting sequence 308

Figure 12-8 Input waveforms of Input1 and Input2 when the decoder module is working normally..... 309

Figure 12-9 Input1 and Input2 input waveforms when decoder module is not working 309

Figure 13-1 Functional block diagram of the independent watchdog module 319

Figure 14-1 Watchdog block diagram..... 324

Figure 14-2 Refresh window and interrupt timing of WWDG 325

Figure 15-1 Block diagram of a single ADC 330

Figure 15-2 ADC clock..... 331

Figure 15-3 Timing diagram 333

Figure 15-4 Injection conversion delay 335

Figure 15-5 Temperature sensor and VREFINT Diagram of the channel..... 338

Figure 16-1 Comparator System Connection Diagram..... 354

Figure 17-1 I2C functional block diagram..... 374

Figure 17-2 I2C bus protocol..... 374

Figure 17-3 Slave transmitter transfer sequence diagram 377

Figure 17-4 Slave receiver transfer sequence diagram 378

Figure 17-5 Master transmitter transfer sequence diagram..... 380

Figure 17-6 Master receiver transfer sequence diagram 382

Figure 18-1 USART block diagram 401

Figure 18-2 word length = 8 setting..... 402

Figure 18-3 word length = 9 setting..... 403

Figure 18-4 configuration stop bit 404

Figure 18-5 TXC/TXDE changes during transmission..... 405

Figure 18-6 Start bit detection 406

Figure 18-7 Transmission using DMA 412

Figure 18-8 Reception using DMA..... 413

Figure 18-9 hardware flow control between two USART 413

Figure 18-10 RTS flow control..... 414

Figure 18-11 CTS flow controls 414

Figure 18-12 Mute mode using idle line detection 415

Figure 18-13 Mute mode detected using address mark..... 416

Figure 18-14 USART synchronous transmission example 417

Figure 18-15 USART data clock timing example (WL=0)..... 417

Figure 18-16 USART data clock timing example (WL=1)..... 418

Figure 18-17 RX data sampling / holding time..... 418

Figure 18-18 IrDASIRENDEC-Block diagram 420

Figure 18-19 IrDA data Modulation (3/16)-normal mode 420

Figure 18-20 Break detection in LIN mode (11-bit break length-the LINBDL bit is set)..... 421

Figure 18-21 Break detection and framing error detection in LIN mode 422

Figure 18-22 ISO7816-3 Asynchronous Protocol..... 423

Figure 18-23 Use 1.5 stop bits to detect parity errors 424

Figure 19-1 LPUART block diagram..... 437

Figure 19-2 frame format..... 438

Figure 19-3 TXC changes during transmission 440

Figure 19-4 Data sampling for noise detection 442

Figure 19-5 Sending using DMA..... 445

Figure 19-6 Receiving with DMA 446

Figure 19-7 Hardware flow control between two LPUART 446

Figure 19-8 RTS flow control 447

Figure 19-9 CTS flow control..... 447

Figure 20-1 SPI block diagram 456

Figure 20-2 Selective management of hardware/software 457

Figure 20-3 Master and slave applications 458

Figure 20-4 Data clock timing diagram 459

Figure 20-5 Schematic diagram of the change of TE/RNE/BUSY when the host is continuously transmitting in full duplex mode..... 460

Figure 20-6 Schematic diagram of TE/BUSY change when the host transmits continuously in one-way only mode 461

Figure 20-7 Schematic diagram of RNE change when continuous transmission occurs in receive-only mode (BIDIRMODE = 0 and RONLY = 1)..... 461

Figure 20-8 Schematic diagram of the change of TE/RNE/BUSY when the slave is continuously transmitting in full duplex mode..... 463

Figure 20-9 Schematic diagram of TE/BUSY change during continuous transmission in slave unidirectional transmit-only mode 463

Figure 20-10 Schematic diagram of TE/BUSY change when BIDIRMODE = 0 and RONLY = 0 are transmitted discontinuously. 465

Figure 20-11 Transmission using DMA..... 467

Figure 20-12 Reception using DMA..... 468

Figure 20-13 I²S block diagram 470

Figure 20-14 I²S Philips protocol waveform (16/32-bit full precision, CLKPOL = 0)..... 472

Figure 20-15 I²S Philips protocol standard waveform (24-bit frame, CLKPOL = 0)..... 472

Figure 20-16 I²S Philips protocol standard waveform (16-bit extended to 32-bit packet frame, CLKPOL = 0)..... 472

Figure 20-17 The MSB is aligned with 16-bit or 32-bit full precision, CLKPOL = 0..... 474

Figure 20-18 MSB aligns 24-bit data, CLKPOL = 0 474

Figure 20-19 MSB-aligned 16-bit data is extended to 32-bit packet frame, CLKPOL = 0 474

Figure 20-20 LSB alignment 16-bit or 32-bit full precision, CLKPOL = 0..... 475

Figure 20-21 LSB aligns 24-bit data, CLKPOL = 0 475

Figure 20-22 LSB aligned 16-bit data is extended to 32-bit packet frame, CLKPOL = 0..... 476

Figure 20-23 PCM standard waveform (16 bits) 476

Figure 20-24 PCM standard waveform (16-bit extended to 32-bit packet frame)..... 477

Figure 20-25 I²S clock generator structure 477

Figure 20-26 Audio sampling frequency definition 478

Figure 21-1 RTC block diagram 492

Figure 22-1 Block diagram of OPAMP connection diagram 519

Figure 22-2 OPAMP external amplification mode..... 520

Figure 22-3 OPAMP follower mode 521

Figure 22-4 Internal programmable gain mode 522

Figure 25-1 Topology of CAN network..... 535

Figure 25-2 CAN working mode 537

Figure 25-3 Single CAN block diagram 538

Figure 25-4 loopback mode 539

Figure 25-5 silent mode 540

Figure 25-6 loopback silent mode..... 540

Figure 25-7 Send mailbox status..... 542

Figure 25-8 Receive FIFO status 543

Figure 25-9 Filter bit width setting-register organization 545

Figure 25-10 Examples of filter mechanisms 547

Figure 25-11 Bit sequence 549

Figure 25-12 Various CAN frames 550

Figure 25-13 Event flag and interrupt generation 551

Figure 25-14 CAN error state diagram 552

Figure 27-1 N32G032 level and Cortex®-M0 level debugging block diagram..... 580

1 Abbreviations in The Text

1.1 Describes the List of Abbreviations Used in the Register Table

The following abbreviations are used in the description of registers:

read/write(rw)	Software can read and write these bits.
read-only(r)	Software can only read these bits.
write-only(w)	Software can only write this bit, and reading this bit will return the reset value.
read/clear(rc_w1)	Software can read this bit or clear it by writing '1', and writing '0' has no effect on this bit.
read/clear(rc_w0)	Software can read this bit or clear it by writing '0', and writing '1' has no effect on this bit.
read/clear by read(rc_r)	Software can read this bit. Reading this bit will automatically clear it to '0'. Writing '0' has no effect on this bit.
read/set(rs)	Software can read or set this bit. Writing '0' has no effect on this bit.
read-only write trigger(rt_w)	Software can read this bit and write '0' or '1' to trigger an event, but it has no effect on this bit value.
toggle(t)	Software can only flip this bit by writing '1', and writing '0' has no effect on this bit.
Reserved(Res.)	Reserved bit, must be kept at reset value.

1.2 Available Peripherals

For all models of N32G032 microcontroller series, the existence and number of a peripheral, please refer to the data sheet of the corresponding model.

2 Memory and Bus Architecture

2.1 System Architecture

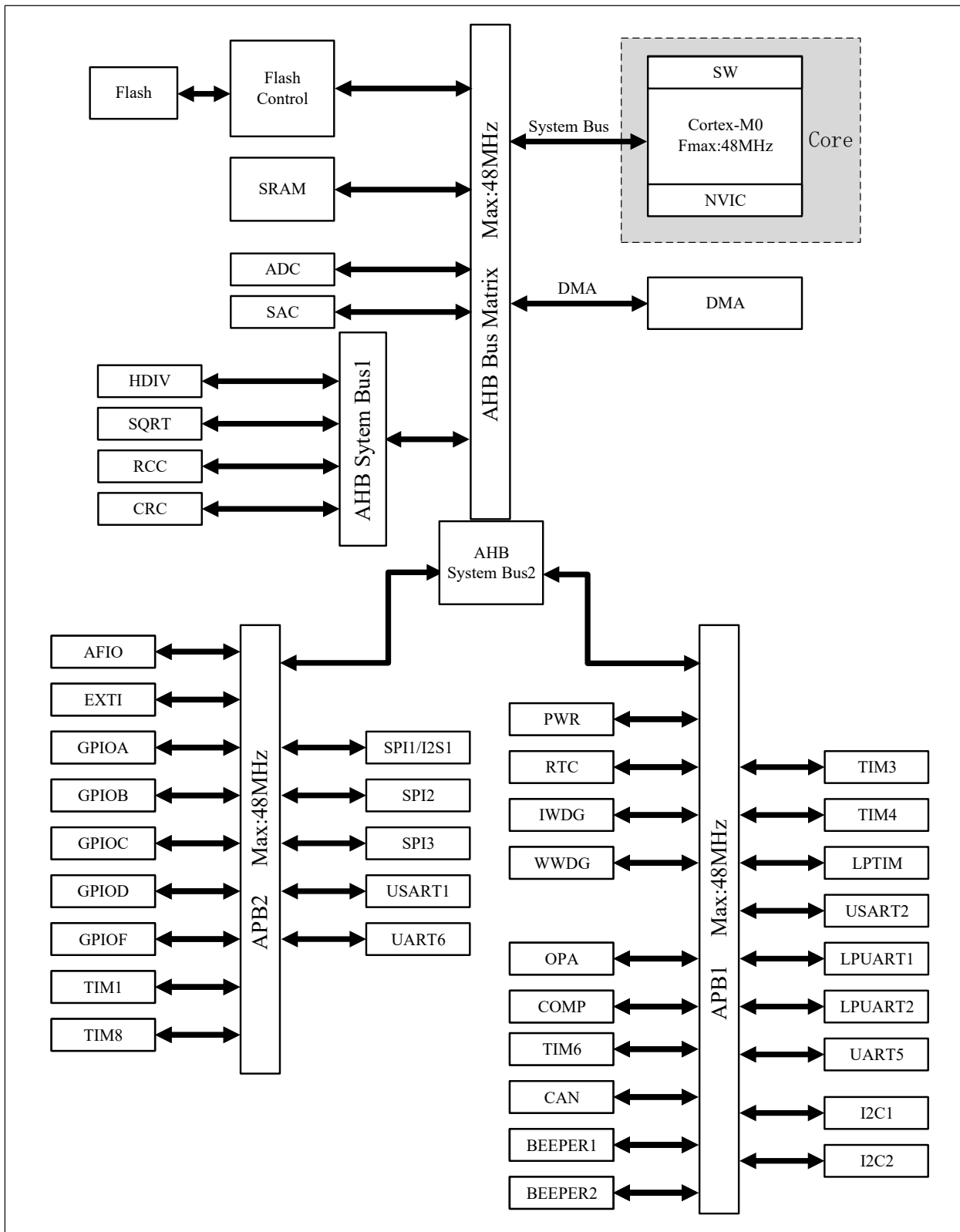
2.1.1 Bus Architecture

The main system consists of the following parts:

- Two main drive units:
 - Cortex[®]-M0 core system bus
 - General purpose DMA
- Six passive units
 - Embedded SRAM
 - Embedded Flash memory
 - ADC
 - AHB to AHB bridge, it connects some AHB devices
 - AHB to APB bridge (AHB2APB_{x,x=1,2}), which connects all APB devices

These are connected to each other through a multi-level AHB bus architecture, as shown in Figure 2-1:

Figure 2-1 Bus Architecture



- CPU System bus: This bus connects to the Cortex[®]-M0 kernel Sbus to bus matrix, used for instruction pre-fetch, data load (constant load and debug access) and AHB/APB peripheral access.
- DMA bus: The DMA's AHB master interface is connected to the bus matrix, which coordinates the kernel and DMA

access to SRAM, Flash and peripherals.

- The bus matrix coordinates access arbitration between the kernel system bus and the DMA master bus. The arbitration uses a round robin algorithm. The bus matrix consists of two driver components (CPU system bus, DMA bus) and six slave components (Flash memory interface, SRAM, ADC, and AHB system bus 1/2). Some AHB peripherals are connected to system bus 1 through the bus matrix, and system bus 2 is connected to two AHB2APB Bridges.
- The system consists of two AHB2APB Bridges, specifically AHB2APB1 and AHB2APB2. APB1 contain 20 APB peripherals with the maximum speed of PCLK being 48MHz. APB2 contains 14 APB peripherals with the maximum speed of PCLK being 48MHz.

2.1.2 Bus Address Mapping

The address mapping includes all AHB and APB peripherals: AHB peripherals, APB1 peripherals, APB2 peripherals, Flash, SRAM, System Memory etc. The specific mapping is as follows:

Figure 2-2 Bus Address Map

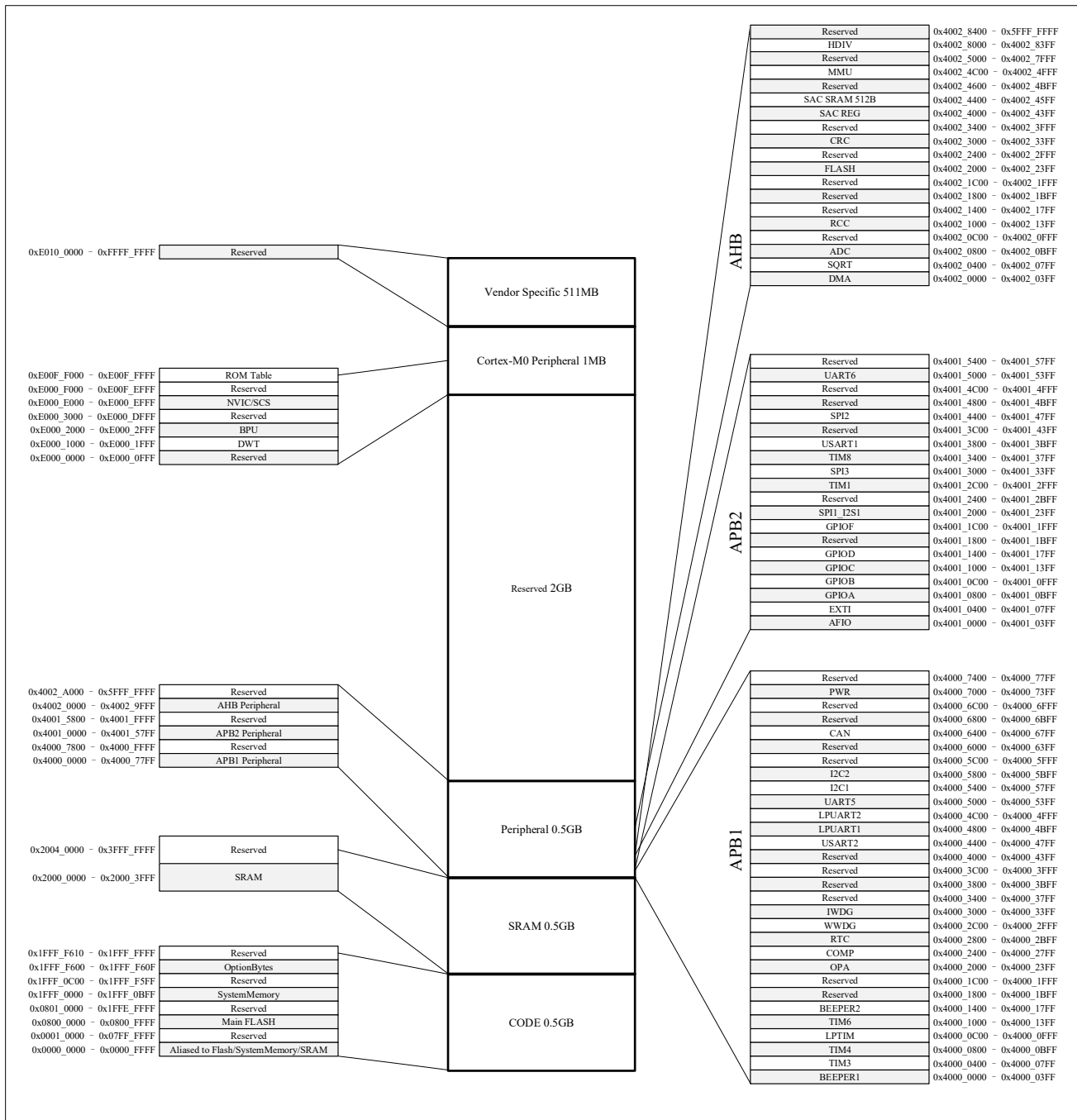


Table 2-1 List of Peripheral Register Addresses

Address Range	Peripherals	Bus
0x4002_8400 - 0x5FFF_FFFF	Reserved	AHB
0x4002_8000 - 0x4002_83FF	HDIV	
0x4002_3400 - 0x4002_7FFF	Reserved	
0x4002_3000 - 0x4002_33FF	CRC	
0x4002_2400 - 0x4002_2FFF	Reserved	
0x4002_2000 - 0x4002_23FF	Flash	
0x4002_1400 - 0x4002_1FFF	Reserved	

Address Range	Peripherals	Bus	
0x4002_1000 – 0x4002_13FF	RCC		
0x4002_0C00 – 0x4002_0FFF	Reserved		
0x4002_0800 – 0x4002_0BFF	ADC		
0x4002_0400 – 0x4002_07FF	SQRT		
0x4002_0000 – 0x4002_03FF	DMA		
0x4001_8000 – 0x4001_FFFF	Reserved		APB2
0x4001_5400 – 0x4001_7FFF	Reserved		
0x4001_5000 – 0x4001_53FF	UART6		
0x4001_4800 – 0x4001_4FFF	Reserved		
0x4001_4400 – 0x4001_47FF	SPI2		
0x4001_3C00 – 0x4001_43FF	Reserved		
0x4001_3800 – 0x4001_3BFF	USART1		
0x4001_3400 – 0x4001_37FF	TIM8		
0x4001_3000 – 0x4001_33FF	SPI3		
0x4001_2C00 – 0x4001_2FFF	TIM1		
0x4001_2400 – 0x4001_2BFF	Reserved		
0x4001_2000 – 0x4001_23FF	SPI1_I2S		
0x4001_1C00 – 0x4001_1FFF	GPIOF		
0x4001_1800 – 0x4001_1BFF	Reserved		
0x4001_1400 – 0x4001_17FF	GPIOD		
0x4001_1000 – 0x4001_13FF	GPIOC		
0x4001_0C00 – 0x4001_0FFF	GPIOB		
0x4001_0800 – 0x4001_0BFF	GPIOA		
0x4001_0400 – 0x4001_07FF	EXTI		
0x4001_0000 – 0x4001_03FF	AFIO		
0x4000_7400 – 0x4000_FFFF	Reserved	APB1	
0x4000_7000 – 0x4000_73FF	PWR		
0x4000_6800 – 0x4000_6FFF	Reserved		
0x4000_6400 – 0x4000_67FF	CAN		
0x4000_5C00 – 0x4000_63FF	Reserved		
0x4000_5800 – 0x4000_5BFF	I2C2		
0x4000_5400 – 0x4000_57FF	I2C1		
0x4000_5000 – 0x4000_53FF	UART5		
0x4000_4C00 – 0x4000_4FFF	LPUART2		
0x4000_4800 – 0x4000_4BFF	LPUART1		
0x4000_4400 – 0x4000_47FF	USART2		
0x4000_3400 – 0x4000_43FF	Reserved		
0x4000_3000 – 0x4000_33FF	IWDG		
0x4000_2C00 – 0x4000_2FFF	WWDG		
0x4000_2800 – 0x4000_2BFF	RTC		
0x4000_2400 – 0x4000_27FF	COMP		
0x4000_2000 – 0x4000_23FF	OPAMP		

Address Range	Peripherals	Bus
0x4000_1800 – 0x4000_1FFF	Reserved	
0x4000_1400 – 0x4000_17FF	BEEPER2	
0x4000_1000 – 0x4000_13FF	TIM6	
0x4000_0C00 – 0x4000_0FFF	LPTIM	
0x4000_0800 – 0x4000_0BFF	TIM4	
0x4000_0400 – 0x4000_07FF	TIM3	
0x4000_0000 – 0x4000_03FF	BEEPER1	

2.1.3 Boot Management

Boot address

During system startup, you can select the BOOT mode after the reset through the BOOT0 pin and the user option byte configuration. The value of the BOOT pin will be re-sampled after the system is reset or exits from the PD mode. After the startup delay, the CPU fetches the top-of-stack value from address 0x0000_0000 and executes the code from the reset vector address indicated by address 0x0000_0004. Because of the Cortex[®]-M0 always gets the top-of-stack value and reset vector from addresses 0x0000_0000 and 0x0000_0004, so boot is only suitable for booting the CODE block, and address remapping is designed for boot space. There are three boot modes to choose from:

- Boot from Main Flash:
 - Main Flash memory is mapped to the boot space (0x0000_0000);
 - Main Flash memory is accessible in two address areas, 0x0000_0000 or 0x0800_0000;
- Boot from System Memory:
 - System Memory is mapped to boot space (0x0000_0000);
 - System Memory can be accessed in two address areas, 0x0000_0000 or 0x1FFF_0000;
- Boot from the embedded SRAM:
 - The embedded SRAM is mapped to boot space (0x0000_0000);
 - The embedded SRAM is accessible in two address areas, 0x0000_0000 or 0x2000_0000;

2.1.3.1 Boot configuration

Three different BOOT modes can be selected through the BOOT0 pin and the user option byte configuration.

Table 2-2 List of Boot Mode

Boot Mode Select Pin				Boot Mode	Specifies The Start Address For Accessing Memory Space In Boot Mode		
nBOOT1	nBOOT0	BOOT0 pin	nSWBOOT0		Main Flash	System Memory	SRAM
X	X	0	1	Main Flash start	0x0000_0000	0x1FFF_0000	0x2000_0000
X	1	X	0		0x0800_0000		
1	X	1	1	System Memory Start	0x08000000	0x0000_0000 0x1FFF_0000	0x2000_0000
1	0	X	0				
0	X	1	1	SRAM start	0x08000000	0x1FFF_0000	0x0000_0000

Boot Mode Select Pin				Boot Mode	Specifies The Start Address For Accessing Memory Space In Boot Mode		
nBOOT1	nBOOT0	BOOT0 pin	nSWBOOT0		Main Flash	System Memory	SRAM
0	0	X	0			0x2000_0000	

Note: BOOT0 and GPIO are multiplexed, and input drop - down is used by default during power-on.

Embedded bootloader

The embedded bootloader is stored in System Memory for reprogramming Flash memory via USART1. The USART1 interface can operate not only with an external clock (HSE) but also with an internal 8MHz oscillator (HSI). For further details, please refer to “The Embedded Boot Loader Manual”.

2.2 Memory System

The program memory, data memory, registers and I/O ports are organized in the same 4GB linear address space. Data bytes are stored in the memory in Little Endian format. The lowest numbered byte in a word is regarded as the least significant byte of the word, while the highest numbered byte is the most significant byte. The specifications of program memory and data memory are as follows.

2.2.1 FLASH Specification

The Flash consists of a main Flash memory block and an information block, which are described below: (The capacity value in the following description does not include ECC)

- The maximum main memory block is 64KB, also known as main Flash memory, which contains 128 Pages for storing and running user programs and storing data.
- The information block is 5KB, including 10 Pages, and consists of system memory block (3KB), system configuration block (1.5KB) and option byte block (0.5KB).
 - The system memory block is 3KB, which contains 6 Pages, and is used for storing and running the bootloader (BOOT).
 - The system configuration block is 1.5KB, including 3 Pages.
 - The option byte block is 0.5KB, containing 1 Page, with an effective space of 14B, Both the BOOT programs and user programs can read, write and erase this area.

Flash memory organization

Both the main memory block and the information block are allocated to bus address spaces.

Table 2-3 Flash Bus Address List

Memory Area	Page Name	Address Range	Size
Main memory block	Page 0	0x0800_0000 – 0x0800_01FF	0.5 KB
	Page 1	0x0800_0200 – 0x0800_03FF	0.5 KB
	Page 2	0x0800_0400 – 0x0800_05FF	0.5 KB
	⋮	⋮	⋮

Memory Area	Page Name	Address Range	Size
	Page 127	0x0800_FE00 – 0x0800_FFFF	0.5 KB
Information block	System memory block	0x1FFF_0000 – 0x1FFF_0BFF	3KB
	System configuration area	0x1FFF_F000 – 0x1FFF_F5FF	1.5 KB
	Option byte area	0x1FFF_F600 – 0x1FFF_F60D	14B
Flash memory block interface registers	FLASH_AC	0x4002_2000 – 0x4002_2003	4B
	FLASH_KEY	0x4002_2004 – 0x4002_2007	4B
	FLASH_OPTKEY	0x4002_2008 – 0x4002_200B	4B
	FLASH_STS	0x4002_200C – 0x4002_200F	4B
	FLASH_CTRL	0x4002_2010 – 0x4002_2013	4B
	FLASH_ADD	0x4002_2014 – 0x4002_2017	4B
	Reserved	0x4002_2018 – 0x4002_201B	4B
	FLASH_OB	0x4002_201C – 0x4002_201F	4B
	FLASH_WRP	0x4002_2020 – 0x4002_2023	4B
	FLASH_ECC	0x4002_2024 – 0x4002_2027	4B

The Flash memory is organized as 32-bit wide memory units, which can store codes and data constants.

Information block is divided into three parts:

- The system memory block is used to store the bootloader in the system memory. The bootloader uses USART1 serial interface to program the Flash memory.
- System configuration block contains basic information about the chip.
- The option byte block, writing to main memory and information block is managed by embedded Flash programming/erasing controller.

There are two ways to protect Flash memory from illegal access (read, write and erase):

- Page write protection (WRP)
- Read protection (RDP)

When the Flash memory write operation is executed, any read operation to the Flash memory will stall the bus, and the read operation can only be performed correctly after the write operation is completed. This means that code or data fetches cannot be made while a program/erase operation is ongoing.

When performing Flash programming(write or erase), the internal RC oscillator (HSI) must be turned on.

Note: in the low power consumption mode, all Flash memory operations are suspended.

Read and write operation

The Flash operation only supports 32-bit operation, and the Flash should be erased before the write operation, and the minimum block size for erasing is one Page, which is 0.5KB. Write operation is divided into erasing and programming phases.

When reading Flash, the number of waiting cycles for reading can be configured by the register. When using, it needs to be calculated in combination with the clock frequency of system(SYSCLK). For example, when $\text{SYSCLK} \leq 18\text{MHz}$, the minimum number of waiting periods is 0; When $18\text{MHz} < \text{SYSCLK} \leq 36\text{MHz}$, the minimum number of waiting periods is 1; When $36\text{MHz} < \text{SYSCLK} \leq 48\text{MHz}$, the minimum number of waiting periods is 2.

Note: whether number of wait periods is not zero, enable prefetch buffer can improve overall efficiency.

Unlock Flash

After reset, the Flash module is protected and cannot be written into the FLASH_CTRL register to prevent accidental operation of Flash memory due to electrical disturbances and other reasons. By writing a specific sequence of key values into the FLASH_KEY register, you can unlock the FLASH_CTRL register. The specific sequence is: first, writing KEY1 = 0x45670123 to the FLASH_KEY register, and then write KEY2 = 0xCDEF89AB to the FLASH_KEY register.

If there is an error in sequence or key value, a bus error will be returned and the FLASH_CTRL register will be locked until the next reset. Software can check the FLASH_CTRL.LOCK bit to confirm whether the Flash is unlocked. If normal locking is required, software can set the FLASH_CTRL.LOCK bit to 1. After that, the Flash can be unlocked by writing the correct key sequence to the FLASH_KEY register.

Erase and program

2.2.1.1.1 Erase of main memory block

The main memory block can be erased page by page or completely(Mass Erase)

Page Erase

Page Erase process:

1. Check the FLASH_STS.BUSY bit to confirm that there are no other Flash operations in progress;
2. Set the FLASH_CTRL.PER bit to '1';
3. Select the page to be erased with the FLASH_ADD register;
4. Set the FLASH_CTRL.START bit to '1';
5. Wait for the FLASH_STS.BUSY bit to change to '0';
6. Read out the content of the erased page and verify it.

Mass Erase

Mass Erase process:

1. Check the FLASH_STS.BUSY bit to confirm that there are no other Flash operations in progress;
2. Set the FLASH_CTRL.MER bit to '1';
3. Set the FLASH_CTRL.START bit to '1';
4. Wait for the FLASH_STS.BUSY bit to change to '0';
5. Read out all pages and verify.

2.2.1.1.2 Main memory block programming

The main memory block can be programmed 32 bits at a time. When the FLASH_CTRL.PG bit is '1', writing a word into a Flash address will start programming once; Writing any half word of data will result in a bus error. During the programming process (the FLASH_STS.BUSY bit is '1'), any operation of reading or writing the Flash memory will cause the CPU to pause until the end of the Flash programming.

Main memory programming process:

1. Check the FLASH_STS.BUSY bit to confirm that there are no other Flash operations in progress;

2. Set the FLASH_CTRL.PG bit to '1';
3. Write the word to be programmed at the specified address;
4. Wait for the FLASH_STS.BUSY bit to change to '0';
5. Read the written address and verify the data.

Note: when the FLASH_STS.BUSY bit is '1', you cannot write to any register.

2.2.1.2 Option byte erase and programming

The option byte area is programmed differently from the main memory block. The number of option bytes is limited 7 bytes (2 bytes for write protection, 2 bytes for read protection, 1 byte for configuration and 2 bytes for storing user data). After unlocking the Flash, you must write KEY1 and KEY2 respectively (refer to Section 0) to the FLASH_OPTKEY register, and then set the FLASH_CTRL.OPTWE bit to '1'. At this time, the option byte area can be programmed: set the FLASH_CTRL.OPTPG bit to '1' and then write a word to the specified address.

When programming the word in the option byte area, use the low byte in the half-word and automatically calculate the high byte (the high byte is the complement of the low byte), before starting the programming operation, this ensures that the option byte and its complement are always correct.

Option byte erase process:

1. Check the FLASH_STS.BUSY bit to confirm that there are no other Flash operations in progress;
2. Unlock the FLASH_CTRL.OPTWE bit;
3. Set the FLASH_CTRL.OPTER bit to '1';
4. Set the FLASH_CTRL.START bit to '1';
5. Wait for the FLASH_STS.BUSY bit to change to '0';
6. Read the erased option byte and verify it.

Option byte area programming process:

1. Check the FLASH_STS.BUSY bit to confirm that there are no other Flash operations in progress;
2. Unlock the FLASH_CTRL.OPTWE bit;
3. Set the FLASH_CTRL.OPTPG bit to '1';
4. Writing the word to be programmed to the specified address;
5. Wait for the FLASH_STS.BUSY bit to change to '0';
6. Read the written address and verify the data.

ECC function

The Flash module supports the ECC functionality, enabling 1-bit error detection and correction. ECC encoding and decoding (error correction and error detection) are performed automatically by the hardware. If an error is detected, an error is set and an interrupt is generated.

Instruction prefetching

The Flash module supports the instruction prefetch function with the prefetch buffer size of 8B. Through instruction prefetching, the instruction execution efficiency of CPU can be improved. The instruction prefetch function can be

configured to be enabled or disabled through the register, and it is enabled by default.

Option byte

The option byte block is mainly used to configure read-write protection, boot mode configuration, software/hardware watchdog and reset options when the system is in power-down or stop mode. It consists of 7 options bytes: 2 bytes for write protection, 2 bytes for read protection, 1 byte for configuration option, 2 bytes defined by user. The option byte block also contains the complement codes corresponding to these 7 option bytes. These complement codes need to be automatically calculated by hardware when the option bytes are written and used for verification when the option bytes are read.

By default, the option byte block is always readable and write-protected. To write (program/erase) the option byte block, first unlock the Flash, then unlock the option byte: write the correct key-value sequence (KEY1 = 0x45670123, KEY2 = 0xCDEF89AB) in the FLASH_OPTKEY, and then write operation to the option byte block will be allowed. If the sequence is wrong or the key value is wrong, a bus error will be returned and the option byte will be locked until the next reset. To lock the option byte, write '0' to the FLASH_CTRL.OPTWE bit by software. The option byte can be unlocked by writing the correct key-value sequence in the FLASH_OPTKEY.

After each system reset, the option byte data is read out from the option byte block and stored it in the option byte register (FLASH_OB/FLASH_WRP) with read-only property. At the same time, the option byte complement data read out together will be used to verify whether the option byte data is correct. If it does not match, an option byte error flag (FLASH_OB.OBERR) will be generated. When an option byte error occurs, the corresponding option byte is forced to 0xFF. When the option byte and its complement are both 0xFF (the state after erasing), the above verification steps are skipped and verification is not required.

Table 2-4 Option Byte List

Address	[31:24] Corresponding Complement Code	[23:16] Option Byte	[15:8] Corresponding Complement Code	[7:0] Option Byte
0x1FFF_F600	nUSER	USER	nRDP1	RDP1
0x1FFF_F604	nData1	Data1	nData0	Data0
0x1FFF_F608	nWRP1	WRP1	nWRP0	WRP0
0x1FFF_F60C	Reserved	Reserved	nRDP2	RDP2

- Read protection L1 level option byte: RDP1
 - Protect the code stored in the Flash memory;
 - When the correct value is written, it is not allowed to read the Flash memory;
 - The result of whether RDP1 is turned on or not can be inquired through FLASH_OB[1];
- User configuration options: USER
 - The USER [7:6]: Reserved;
 - USER[5]: nSWBOOT0_SEL configuration option, which can be queried by FLASH_OB[7]
 - 0: nBOOT0 configuration option used for BOOT mode selection
 - 1: BOOT0 Pin is used to select the BOOT mode

- USER[4]: nBOOT1 configuration option, which can be queried by FLASH_OB[6]
- USER[3]: nBOOT0 configuration option, which can be queried by FLASH_OB[5]
- USER[2]: nRST_PD configuration option, which can be queried through FLASH_OB[4]
 - 0: A reset occurs when entering PD mode
 - 1: No reset occurs when entering PD mode
- USER[1]: nRST_STOP configuration option, which can be queried through FLASH_OB[3]
 - 0: A reset occurs when entering STOP mode
 - 1: No reset occurs when entering the STOP mode
- USER[0]: WDG_SW configuration option, which can be queried through FLASH_OB[2]
 - 0: Hardware watchdog
 - 1: Software watchdog
- 2 bytes of user data: Datax
 - Data1 (stored in FLASH_OB[25:18]);
 - Data0 (stored in FLASH_OB [17:10]);
- Write protection option byte: WRP0 ~ 1, which can be written through the register FLASH_WRP [15:0] query
 - WRP0: Write protection for pages 0 to 63, bit[0] corresponds to Page (0 to 7)..., bit[7] corresponds to Page (56~63);
 - WRP1: Write protection for pages 64~127, bit[0] corresponds to Page (64~71)..., bit[7] corresponds to Page (120~127);
- Read protection L2 level option byte: RDP2
 - Add protection function on the basis of L1, refer to detailed description of read protection in section 2.2.1.9;
 - The result of whether RDP2 is turned on or not can be inquired through FLASH_OB [31];

Write protect

Write protection can be configured for all pages in the Flash main memory block (maximum 64KB) to prevent accidental write operations caused by program crashes or electrical disturbances. The basic unit of write protection is as follows: for Page 0 to 127, every 8 pages is a basic protection unit. Write protection can be configured by setting WRP0 to 1 in the option byte block; After each configuration, a system reset is required for the configured value to be reloaded to take effect. If a protected page is programmed or erased, a protection error flag will be returned in the FLASH_STS.

The system information block contains the following blocks:

- The system memory block (3KB) in the system information block stores the boot program and cannot be changed.
- The system configuration block (1.5KB) in the system information block stores the basic information of the chip and cannot be changed.
- The option byte block (0.5KB) in the system information block stores the user-configurable option byte information. The write protection of the option byte block is achieved by writing 0 to the FLASH_CTRL.OPTWE bit by software, and after that, you can write the correct key value sequence to FLASH_OPTKEY to release the write protection of

the option byte.

Read protection

The user code in Flash can be protected against unauthorized reading by setting read protection. Read protection is set by configuring RDP bytes in the option byte block. Three different read protection levels can be configured, as shown in the following Table

Table 2-5 Read Protection Configuration List

Read Protection Status	RDP1	nRDP1	nRDP2	RDP2
L1 level	0xFF	0xFF	RDP2! = 0xCC nRDP2! = 0x33	
Unprotected	0xA5	0x5A	RDP2! = 0xCC nRDP2! = 0x33	
L2 level	0XX	0XX	0x33	0xCC
L1 level	Not the above three configurations			

- L0 level:
 - In unprotected state, (RDP1 == 0xA5 & nRDP1 == 0x5A) && (RDP2!= 0xCC | nRDP2!= 0x33);
 - The main memory block and option byte block can be read arbitrarily;
 - The main memory block and option byte block can be programmed and erased, with configurable read/write protection;
- L1 level:
 - The corresponding $\sim ((RDP1 == 0xA5 \& nRDP1 == 0x5A) \&\& (RDP2!= 0xCC | nRDP2!= 0x33)) | (RDP2 == 0xCC \& nRDP2 == 0x33)$;
 - Only the read operation of the main memory block from the user code is allowed, that is, the read operation of the main memory block is permitted only, when the program is started from the main Flash memory in non debugging mode;
 - All main memory pages can be programmed by code executing in main Flash memory (for functions such as IAP or data storage)
 - All main memory pages do not allow write or erase operations in debug mode or after booting from embedded SRAM (except for mass erase);
 - All pages are not allowed to write or erase in debug mode or after booting from embedded SRAM (except for mass erase);
 - The function of loading code into the embedded SRAM through JTAG/SWD remain effective, and it can be started from the embedded SRAM through JTAG/SWD, which can be used to remove read protection;
 - When the read-protected option byte is reprogrammed to the unprotected L0 level, all the main memory block will be automatically erased, and the process is as follows: (Erasing the option byte block will not result in automatic whole erasing operation, because the result of erasing is 0xFF, which is equivalent to remaining in the protection state of L1 level)
 1. Write the correct key value sequence into FLASH_OPTKEY to unlock the option byte block;
 2. The bus initiates a command to erase the entire option byte area (Page erase);
 3. Bus write 0xA5 to read protection option byte;

- 4. Automatically erase all main memory block by hardware;
- 5. Automatically write 0xA5 to read protection option byte by hardware;
- 6. When the system is reset (such as software reset, etc.), the option byte block (including the new RDP value 0xA5) will be reloaded into the system, and the read protection will be released;
- L2 level: Except that SRAM boot disabled, debug mode disabled, option byte write/page erase disabled and the protection level cannot be modified (irreversible), other features are the same as L1 level. The L2 level is realized by configuring another option byte RDP2. No matter what the value of RDP1 is, as long as it satisfies (RDP2==0xCC & nRDP2==0x33), it is L2 level.

Permission Protection

- Flash main memory area permissions:
 - Under L0 level: the main memory block can be read; the main memory block can be configured with the write protection attribute of each Page for programming and page erasing;
 - Under L1/2 level:
 1. When executed in SWD debug mode or after booting from embedded SRAM, all Pages do not allow (W/R/PE) operations;
 2. The 0th to 7th pages are automatically write-protected, and other pages can be programmed through the code executed in the main Flash memory block (implementing functions such as IAP or data storage);
 3. Other Pages can configure the write protection property of each Page;
 4. When the L1 level is rewritten to the L0 level, all Flash main memory blocks will be automatically erased;
- Flash option byte block permission:
 - Under the L0/L1 level, access is allowed (W/R/PE);
 - Under the L2 level: except that the debug mode is disabled, read-only access to the Flash option byte block is allowed;
- Flash system memory block permissions:
 - Only the code executed in the system memory block is allowed to access (W/R/PE); Flash and SRAM execution code or through the debugging interface are not allowed to access;
 - At the L1/L2 level, the SRAM boot jumps to the system memory execution code and does not allow access (W/R/PE). access otherwise allowed (W/R/PE);
- Flash system configuration block
 - User information: The read operation can only be performed after the system memory is started or executed by jumping to the system memory;
 - ID information: readable at L0 level, access prohibited in SWD debug mode at L1/L2 level, access prohibited at L1 level after sram is started, refer to Table 2-6 for details;

Table 2-6 Flash Read-Write-erase⁽¹⁾ Permission Control Table

Protect	Boot Mode	Main Flash	Changing a
---------	-----------	------------	------------

Level	Perform user	SWD	Main Flash	System Memory	SRAM	Protection Level
	Access block					
L0 level	First 4KB of Flash main memory block	Read-Write-Erase	Read-Write-Erase	Read-Write-Erase	Read-Write-Erase	Change to L1 or L2 is allowed
	Last 4KB of Flash main memory block	Read-Write-Erase	Read-Write-Erase	Read-Write-Erase	Read-Write-Erase	
	Flash main memory block mass erase	Allow	Allow	Allow	Allow	
	Flash option byte block	Read-Write-Erase	Read-Write-Erase	Read-Write-Erase	Read-Write-Erase	
	Flash system memory block	prohibit	prohibit	Read-Write-Erase	prohibit	
	SRAM (All)	Read and write	Read and write	Read and write	Read and write	
L1 level	First 4KB of Flash main memory block	Prohibit	Read-only	Read-only	Read-only	L0 or L2 is allowed. When changed to L0, the main memory block is automatically erased.
	Last 4KB of Flash main memory block	Prohibit	Read-Write-Erase	Read-Write-Erase	Read-Write-Erase	
	Flash main memory block mass erase	Allow	Allow	Allow	Allow	
	Flash option byte block	Read-Write-Erase	Read-Write-Erase	Read-Write-Erase	Read-Write-Erase	
	Flash system memory block	Prohibit	Prohibit	Read-write-erase	Prohibit	
	SRAM (All)	Read and write	Read and write	Read and write	Read and write	
L2 level	First 4KB of Flash main memory block	SWD The interface is disabled.	Read-only	Read-only	Read-only	No modification is allowed.
	Last 4KB of Flash main memory block		Read-write-erase	Read-write-erase	Read-write-erase	
	Flash main memory		Allow	Allow	Allow	

	block mass erase					
	Flash option byte block		Read-only	Read-only	Read-only	
	Flash system memory block		Prohibit	Read-write-erase	Prohibit	
	SRAM (All)		Read and write	Read and write	Read and write	
protect level	Boot mode	SRAM				Changing a Protection Level
	Perform user Access to block	SWD	Main Flash	System Memory	SRAM	
L0 level	First 4KB of Flash main memory block	Read-write-erase	Read-write-erase	Read-write-erase	Read-write-erase	Change to L1 or L2 is allowed
	Last 4KB of Flash main memory block	Read-write-erase	Read-write-erase	Read-write-erase	Read-write-erase	
	Flash main memory block mass erase	Allow	Allow	Allow	Allow	
	Flash option byte block	Read-write-erase	Read-write-erase	Read-write-erase	Read-write-erase	
	Flash system memory block	Prohibit	Prohibit	Read-write-erase	Prohibit	
	SRAM (All)	Read and write	Read and write	Read and write	Read and write	
L1 level	First 4KB of Flash main memory block	Prohibit	Read-only	Read-only	Prohibit	L0 or L2 is allowed. When changed to L0, the main memory block is automatically erased.
	Last 4KB of Flash main memory block	Prohibit	Read-write-erase	Read-write-erase	Prohibit	
	Flash main memory block mass erase	Allow	Allow	Allow	Allow	
	Flash option byte block	Read-write-erase	Read-write-erase	Read-write-erase	Read-write-erase	

	Flash system memory block	Prohibit	Prohibit	Prohibit	Prohibit	
	SRAM (All)	Read and write	Read and write	Read and write	Read and write	
L2 level	First 4KB of Flash main memory block	L2 protection level, cannot boot from SRAM				No modification is allowed. SWD is banned.
	Last 4KB of Flash main memory block					
	Flash main memory block mass erase					
	Flash option byte block					
	Flash system memory block					
	SRAM (All)					
protect level	Boot mode	System Memory				Changing a Protection Level
	Perform user Access to block	SWD	Jump to Flash	System Memory Start the	Jump to SRAM	
L0 level	First 4KB of Flash main memory block	Read-write-erase	Read-write-erase	Read-write-erase	Read-write-erase	Change to L1 or L2 is allowed
	Last 4KB of Flash main memory block	Read-write-erase	Read-write-erase	Read-write-erase	Read-write-erase	
	Flash main memory block mass erase	Allow	Allow	Allow	Allow	
	Flash option byte block	Read-write-erase	Read-write-erase	Read-write-erase	Read-write-erase	
	Flash system memory block	Prohibit	Prohibit	Read-write-erase	Prohibit	

	SRAM (All)	Read and write	Read and write	Read and write	Read and write	
L1 level	First 4KB of Flash main memory block	Prohibit	Read-only	Read-only	Read-only	L0 or L2 is allowed. When changed to L0, the main memory block is automatically erased.
	Last 4KB of Flash main memory block	Prohibit	Read-write-erase	Read-write-erase	Read-write-erase	
	Flash main memory block mass erase	Allow	Allow	Allow	Allow	
	Flash option byte block	Read-write-erase	Read-write-erase	Read-write-erase	Read-write-erase	
	Flash system memory block	Prohibit	Prohibit	Read-write-erase	Prohibit	
	SRAM (All)	Read and write	Read and write	Read and write	Read and write	
L2 level	First 4KB of Flash main memory block	SWD The interface is disabled.	Read-only	Read-only	Read-only	No modification allowed
	Last 4KB of Flash main memory block		Read-write-erase	Read-write-erase	Read-write-erase	
	Flash main memory block mass erase		Allow	Allow	Allow	
	Flash option byte block		Read-only	Read-only	Read-only	
	Flash system memory block		Prohibit	Read-write-erase	Prohibit	
	SRAM (All)		Read and write	Read and write	Read and write	

Note:⁽¹⁾Erase here refers to Flash page erase.

2.2.2 SRAM

SRAM is mainly used for code running, storing variables and data or stacks in the process of program execution, with a maximum capacity of 16KB.

SRAM supports read-write access of byte, half-word and word.

SRAM supports code execution, and can run programs at full speed in SRAM. The maximum address range of SRAM is

from 0x2000 0000 to 0x2000 3FFF.

SRAM cannot retain data in PD mode; In other operating modes (Run, Lprun, Sleep, and Stop), data is retained normally.

The main features are as follows:

- The maximum total capacity is 16KB
- Support byte/half word/word reading and writing
- CPU/DMA can access SRAM.
- The CPU bus can be remapped to access the SRAM, allowing programs to execute directly from the SRAM at the full speed of the CPU.

2.2.3 FLASH Register Description

All register operations must be performed in words (32 bits).

FLASH register overview

Table 2-7 Flash Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																					
000h	FLASH_AC	Reserved																								PRFTBFS	PRFTBFE	Reserved	LATENCY																									
	Reset Value																									1	1		0	0	0																							
004h	FLASH_KEY	FKEY																																																				
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																						
008h	FLASH_OPTKEY	OPTKEY																																																				
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																						
00Ch	FLASH_STS	Reserved																								ECCERR	Reserved	EOP	WRPERR	Reserved	PGERR	Reserved	BUSY																					
	Reset Value																									0		0	0		0	Reserved	0																					
010h	FLASH_CTRL	Reserved																		ECCERRITE	EOPITE	Reserved	ERRITE	OPTWE	Reserved	LOCK	START	OPTER	OPTPG	Reserved	MER	PER	PG																					
	Reset Value																			0	0		0	0		1	0	0	0		0	0	0																					
014h	FLASH_ADD	FADD																																																				
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																						
018h	Reserved																																																					
01Ch	FLASH_OB	RDPRTE	Reserved				Data1						Data0						Not Used	nSWBOOT0	nBOOT1	nBOOT0	nRST_PD	nRST_STOP	WDG_SW	RDPRTE	OBERR																											
	Reset Value	0					1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1																					
020h	FLASH_WRP	Reserved																		WRPT																																		
	Reset Value																			1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1					
024h	FLASH_ECC	Reserved																								ECC																												
	Reset Value																									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

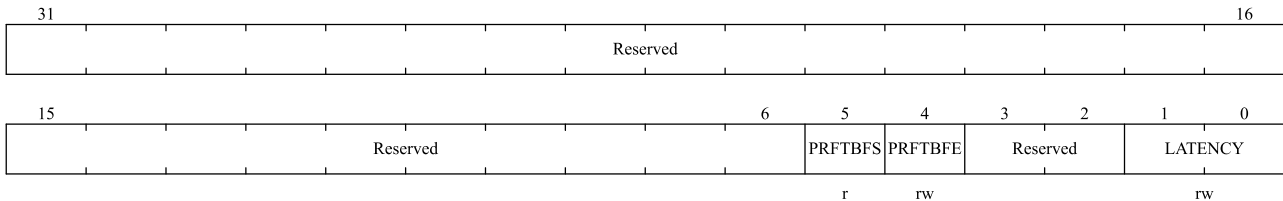
FLASH control and status register

For abbreviations related to register descriptions, please refer to Section 1.1.

2.2.3.1.1 FLASH access control register (FLASH_AC)

Address offset: 0x00

Reset value: 0x0000 0030

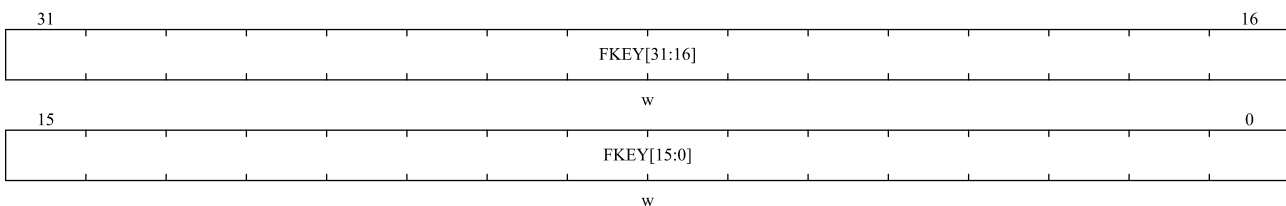


Bit Field	Name	Description
31:6	Reserved	Reserved, the reset value must be maintained.
5	PRFTBFS	Pre-fetch buffer status This bit indicates the state of the pre-fetch buffer 0: The pre-fetch buffer is disabled. 1: The pre-fetch buffer is enabled.
4	PRFTBFE	The pre-fetch buffer is enabled 0: Disables the pre-fetch buffer. 1: Enables the pre-fetch buffer.
3:2	Reserved	Reserved, the reset value must be maintained
1:0	LATENCY	Time delay These bits represent the ratio of the SYSCLK (system clock) cycle to the Flash access time 00: Zero periodic delay, when $0 < \text{SYSCLK} \leq 18\text{MHz}$ 01: A periodic delay, when $18\text{MHz} < \text{SYSCLK} \leq 36\text{MHz}$ 10: Two periodic delay, when $36\text{MHz} < \text{SYSCLK} \leq 48\text{MHz}$ 11: Reserved

2.2.3.1.2 FLASH key register (FLASH_KEY)

Address offset: 0x04

Reset value: 0xXXXX XXXX

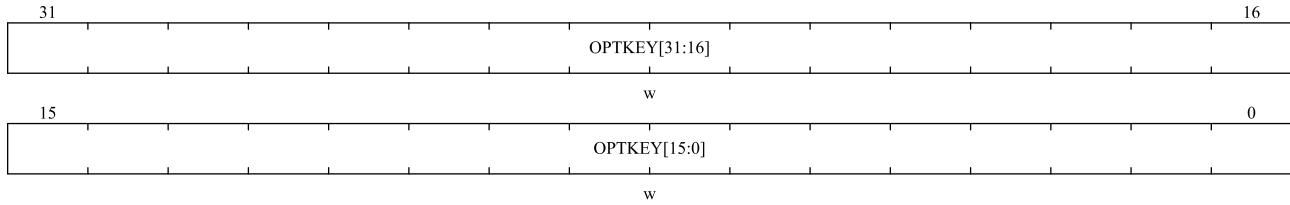


Bit Field	Name	Description
31:0	FKEY	Used to unlock the FLASH_CTRL.LOCK bit.

2.2.3.1.3 FLASH OPTKEY register (FLASH_OPTKEY)

Address offset: 0x08

Reset value: 0xXXXX XXXX

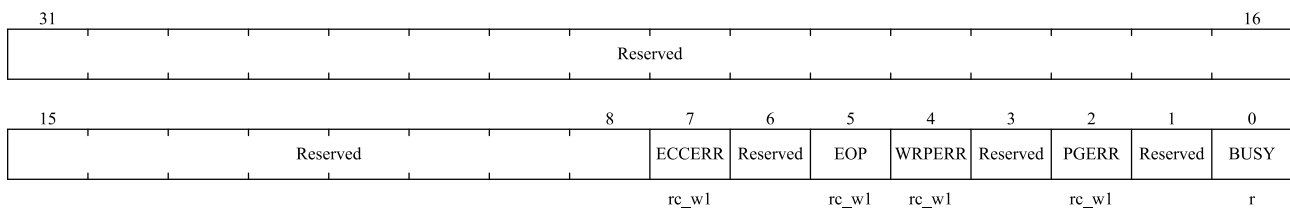


Bit Field	Name	Description
31:0	OPTKEY	Used to unlock the FLASH_CTRL.OPTWE bit.

2.2.3.1.4 FLASH status register (FLASH_STS)

Address offset: 0x0C

Reset value: 0x0000 0000



Bit Field	Name	Description
31:8	Reserved	Reserved, the reset value must be maintained
7	ECCERR	ECC error Error reading FLASH, hardware set this to '1', write '1' to clear this state.
6	Reserved	Reserved, the reset value must be maintained
5	EOP	End of the operation When the Flash operation (programming/erasing) is complete, the hardware sets this to '1' and writing '1' clears this state. <i>Note: EOP status is set for each successful programming or erasure.</i>
4	WRPERR	Write protection error When attempting to program a write protected Flash address, hardware sets this to '1' and writing '1' clears this state.
3	Reserved	Reserved, the reset value must be maintained
2	PGERR	Programming errors When trying to program to an address whose content is not '0xFFFF_FFFF', the hardware sets this to '1'. Writing '1' clears this state. <i>Note: Before programming, the FLASH_CTRL.START bit must be cleared.</i>
1	Reserved	Reserved, the reset value must be maintained
0	BUSY	Busy This bit indicates that a Flash operation is in progress. This bit is set to '1' at the

Bit Field	Name	Description
		start of the Flash operation; This bit is cleared to '0' at the end of the operation or when an error occurs.

2.2.3.1.5 FLASH control register (FLASH_CTRL)

Address offset: 0x10

Reset value: 0x0000 0080

Reserved															
31															16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ECCERR ITE	EOPITE	Reserved	ERRITE	OPTWE	Reserved	LOCK	START	OPTER	OPTPG	Reserved	MER	PER	PG
		rw	rw		rw	rw		rw	rw	rw	rw		rw	rw	rw

Bit Field	Name	Description
31:14	Reserved	Reserved, the reset value must be maintained
13	ECCERRITE	ECC error interrupt This bit allows interrupts to occur when the FLASH_STS.ECCERR bit changes to '1'. 0: Forbid interruption. 1: Interrupts are allowed.
12	EOPITE	Allow operation completion interrupt. This bit allows an interrupt to be generated when the FLASH_STS.EOP bit becomes '1'. 0: Forbid interruption. 1: Interrupts are allowed.
11	Reserved	Reserved, the reset value must be maintained
10	ERRITE	Allow error status to be interrupted This bit allows interrupts in the event of Flash errors (when FLASH_STS.PGERR/ FLASH_STS.WRPERR is set to '1'). 0: Forbid interruption. 1: Interrupts are allowed.
9	OPTWE	Allows option bytes to be written When the bit is '1', programmatic manipulation of the option byte is allowed. When the correct key sequence is written to the FLASH_OPTKEY register, the bit is set to '1'. Software can clear this bit.
8	Reserved	Reserved, the reset value must be maintained
7	LOCK	Lock This bit can only be written as '1'. When the bit is '1', Flash and FLASH_CTRL are locked. Hardware clears this bit to '0' after detecting a correct unlock sequence. After an unsuccessful unlock operation, this bit cannot be changed until the next system reset.

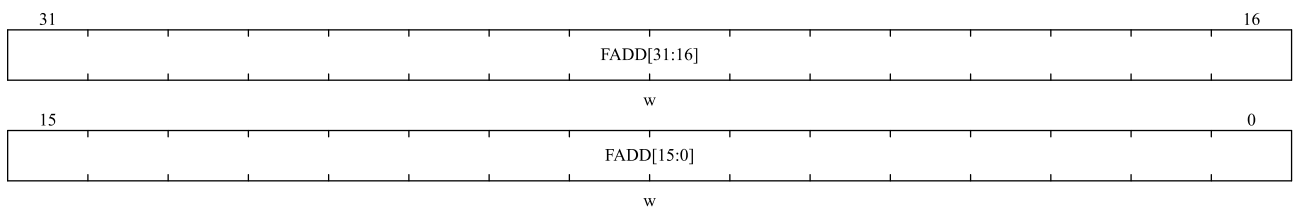
Bit Field	Name	Description
6	START	Start An erase operation is triggered when the bit is '1'. This bit can only be set by software to '1' and cleared to '0' when FLASH_STS.BUSY changes to '1'.
5	OPTER	Erase option bytes 0: Disable option bytes erase mode; 1: Enable option bytes erase mode.
4	OPTPG	Program option bytes 0: Disable option bytes program mode; 1: Enable option bytes program mode.
3	Reserved	Reserved, the reset value must be maintained
2	MER	Mass erase 0: Disable mass erase mode; 1: Enable mass erase mode.
1	PER	Page erase 0: Disable mass erase mode; 1: Enable mass erase mode.
0	PG	Program 0: Disable Program mode; 1: Enable Program mode.

Note: Please refer to Section 0 for programming and erasing.

2.2.3.1.6 FLASH address register (FLASH_ADD)

Address offset: 0x14

Reset value: 0x0000 0000

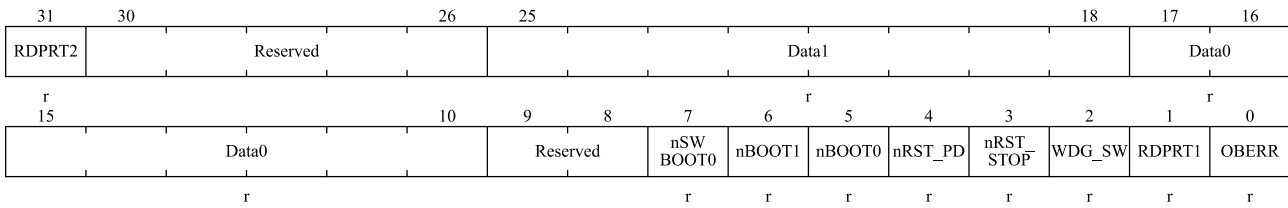


Bit Field	Name	Description
31:0	FADD	Flash memory address Select the address to program when programming and the page to erase when erasing. <i>Note: When the FLASH_STS.BUSY bit is '1', this register cannot be written.</i>

2.2.3.1.7 FLASH option byte register (FLASH_OB)

Address offset: 0x1C

Reset value: 0x03FF FFFC

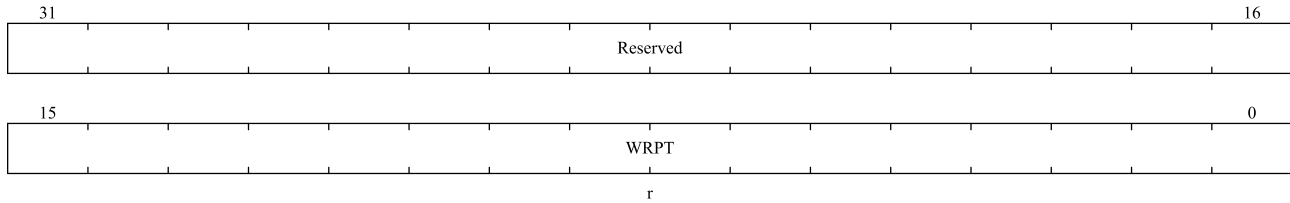


Bit Field	Name	Description
31	RDPRT2	Read protection level L2 0: Read protection L2 is disabled. 1: Read protection L2 is enabled. <i>Note: This bit is read-only.</i>
30:26	Reserved	Reserved, the reset value must be maintained
25:18	Data1[7:0]	Data1 <i>Note: This bit is read-only.</i>
17:10	Data0[7:0]	Data0 <i>Note: This bit is read-only.</i>
9:8	Reserved	Reserved, the reset value must be maintained
7	nSWBOOT0	For the usage rules, refer to Section 2.1.3.1 Boot configuration.
6	nBOOT1	For the usage rules, refer to Section 2.1.3.1 Boot configuration.
5	nBOOT0	For the usage rules, refer to Section 2.1.3.1 Boot configuration.
4	nRST_PD	The Power Down mode is used to reset the configuration 0: A reset occurs immediately after the system enters the Power Down mode. Even if the system enters the Power Down mode, the system will be reset instead of entering the Power Down mode. 1: The system does not reset after entering the Power Down mode. <i>Note: This bit is read-only.</i>
3	nRST_STOP	Enter the STOP mode to reset the configuration 0: A reset occurs immediately after entering the STOP mode. Even if the process of entering the STOP mode is executed, the system will be reset instead of entering the STOP mode. 1: No reset is generated after the STOP mode is entered. <i>Note: This bit is read-only.</i>
2	WDG_SW	Watchdog Settings 0: Hardware watchdog. 1: Software watchdog. <i>Note: This bit is read-only.</i>
1	RDPRT1	Read protection level L1 0: The L1 level of read protection is disabled. 1: L1 read protection is enabled. <i>Note: This bit is read-only.</i>
0	OBERR	Option byte error When this bit is '1', the option byte does not match its inverse. <i>Note: This bit is read-only.</i>

2.2.3.1.8 FLASH write protection register (FLASH_WRP)

Address offset: 0x20

Reset value: 0x0000 FFFF

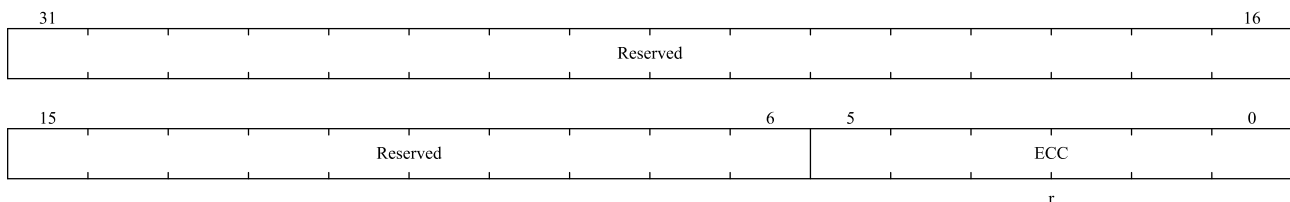


Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained
15:0	WRPT	Write protect This register contains the write protection option byte loaded by option byte area. 0: Write protection takes effect. 1: Write protection is invalid. <i>Note: These bits are read-only.</i>

2.2.3.1.9 FLASH ECC register (FLASH_ECC)

Address offset: 0x24

Reset value: 0x0000 0000



Bit Field	Name	Description
31:6	Reserved	Reserved, the reset value must be maintained
5:0	ECC	After writing a word to a 32-bit Flash address, the corresponding higher 6-bit ECC value.

3 Power Control (PWR)

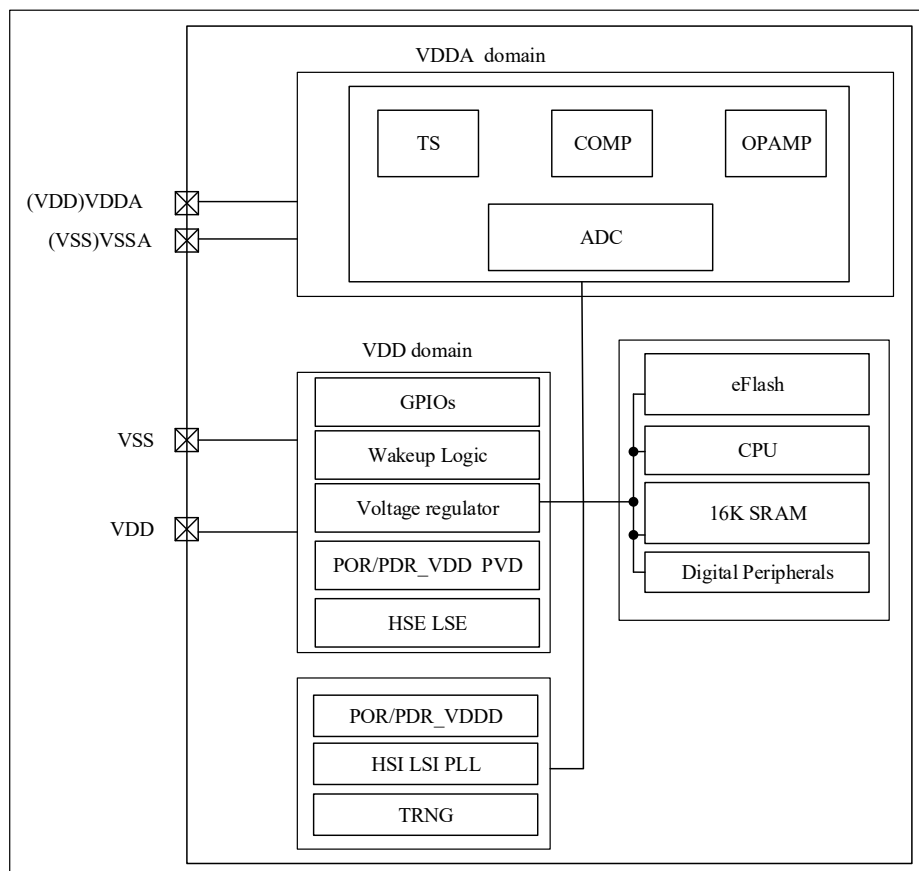
3.1 General Description

The PWR is power management unit to control status of different modules in different power modes. Its major function is to control MCU to enter different power modes and wakeup when events or interrupts happen. The MCU supports the following modes: RUN, LPRUN, SLEEP, STOP and PD (Power Down). PWR controls voltage regulator, Clock sources, Resets and Flash/SRAM/GPIO status in different power modes.

3.1.1 Power Supply

- MCU has an external V_{DD} supply. Embedded voltage regulator is used to supply the internal 1.5V digital power supply. Voltage regulator has two modes, which are normal mode and low power mode.
 - V_{DD} domain: The voltage input range is 1.8V~5.5V, which mainly provides power input for MR, IO and clock reset system.
 - V_{DDA} domain: The voltage input range is 1.8V~5.5V, which powers most analog peripherals. For details, please refer to the electrical characteristics section of the relevant datasheet.
 - V_{DDA} and V_{SSA} must be connected to V_{DD} and V_{SS} respectively.
- Voltage regulator operates in several different modes, depending on the application:
 - RUN mode: The voltage regulator provides power in normal power mode.
 - LPRUN mode: The voltage regulator provides power in normal power mode.
 - SLEEP mode: The voltage regulator provides power in normal power mode.
 - STOP mode: The voltage regulator provides power in low power mode and the output voltage can be configured to 1.5V or 1.2V by software.
 - PD mode: The voltage regulator is turned off.

Figure 3-1 Power Supply Block diagram

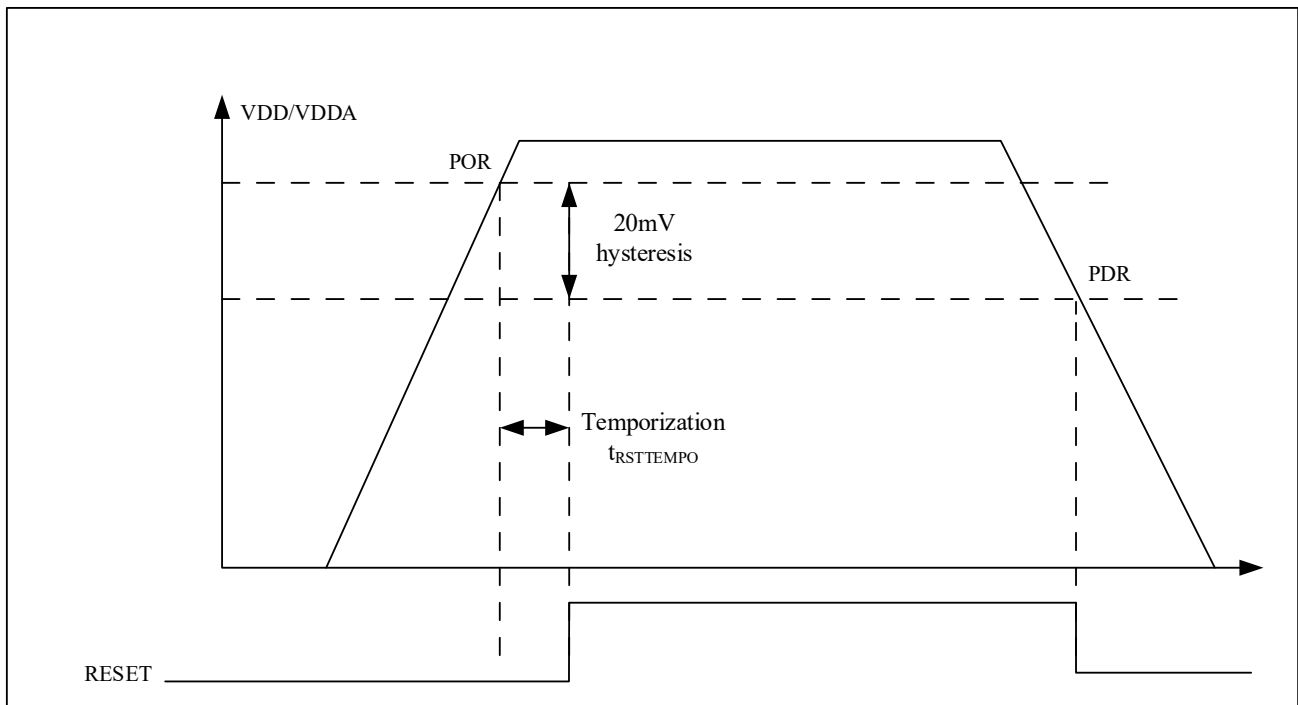


3.1.2 Power Supply Supervisor

Power on reset (POR) and power down reset (PDR)

Power on reset (POR) and power down reset (PDR) circuits are integrated inside the chip. When V_{DD}/V_{DDA} is below the specified limit voltage V_{POR}/V_{PDR} , the system remains in a reset state without the need for an external reset circuit. Refer to the electrical characteristics section of the data sheet for details on power-on and power-off resets.

Figure 3-2 Power on Reset/Power Down Reset Waveform

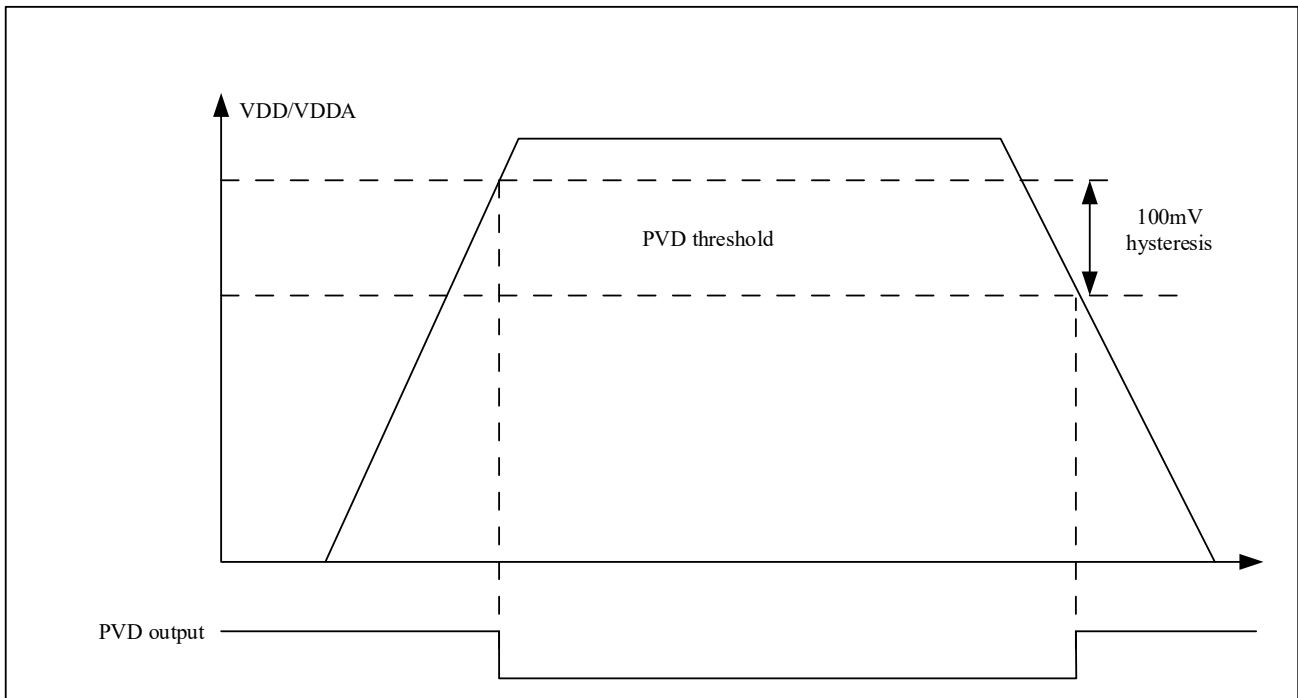


Programmable voltage detector (PVD)

The PVD monitors the power supply by comparing the V_{DD} voltage with the relevant bits in the power control register (PWR_CTRL). PWR_CTRL.PLS select the threshold of the monitoring voltage. Enable PVD by setting the PWR_CTRL.PVDEN.

The PWR_CTRLSTS.PVDO flag is used to indicate whether the V_{DD} is above/below the PVD voltage threshold. This event is connected internally to EXTI line16 and produces an interrupt if the interrupt is enabled in the external interrupt register. A PVD break occurs when the V_{DD} drops below the PVD threshold and/or when the V_{DD} rises above the PVD threshold, according to the rise/fall edge trigger setting of EXTI line 16. For example, this feature can be used to perform emergency shutdown tasks.

Figure 3-3 PVD Threshold Diagram



3.2 Power Modes

The MCU has five power modes: RUN, LPRUN, SLEEP, STOP and PD. Different mode has different performance and power consumption. A summary of MCU power modes is shown below.

Table 3-1 Power Modes

Mode	Condition	Enter	Exit
RUN	CPU running, all peripherals configurable.	Power up, system reset, or wakeup from other power modes.	Enter LPRUN, SLEEP, STOP and PD mode.
LPRUN from SRAM	CPU running at LSI or LSE, PLL off, all other peripherals configurable. Voltage regulator in normal mode. Flash enter deep standby mode.	software control	software control
LPRUN from FLASH	CPU running at LSI or LSE, PLL off, all other peripherals configurable. Voltage regulator in normal mode.	software control	software control
SLEEP	CPU sleep, all peripherals configurable, regulator in normal mode, all digital peripherals powered. Interrupts & Events can wakeup CPU.	WFI CPU returns from ISR WFE	Any interrupts wakeup event.
STOP	CPU deep SLEEP, peripherals clock disabled. Voltage regulator in LP mode. Flash enter deep standby mode.	WFI/WFE: 1) SCB_SCR.SLEEPDEEP	Any interrupts wakeup event through EXTI, NRST, IWDG.

Mode	Condition	Enter	Exit
	HSE/HSI/PLL disabled. LSE/LSI configurable. RTC//LPUART/LPTIM/COMP optional. SRAM/All register retention. All IO retention. After waking up, HSI is enabled.	= 1, no pending interrupts/events. 2) PWR_CTRL.PDSTP = 0	
PD	Voltage regulator OFF, all clocks OFF, most IO output High-Z. NRST/PA0_WKUP0/PC13_WKUP1/PA2_WKUP2 can wake up the chip.	WFI/WFE: 1) SCB_SCR.SLEEPDEEP = 1, no pending interrupt/event. 2) PWR_CTRL.PDSTP = 1	WKUP0/1/2 rising or falling edge, NRST reset.

Note: in STOP mode, after wakeup, the code can continue performing from the stop position.

The operating enabled status of different modules in different power consumption modes are shown in the following table:

Table 3-2 Peripheral Running Status

Peripheral	Run/Active	Sleep	Low Power Run	Stop Mode		Power Down Mode	
				Status	Wakeup Capability	Status	Wakeup Capability
Cortex®-M0	Y	-	Y	-	-	-	-
FLASH	O	O	O	O	-	-	-
SRAM8KB	Y	Y	Y	Y	-	-	-
POR/PDR	Y	Y	Y	Y	Y	-	-
PVD	O	O	O	O	O	-	-
DMA	O	O	O	-	-	-	-
USART/UART	O	O	O	-	-	-	-
LPUART	O	O	O	O	O	-	-
I2C	O	O	O	-	-	-	-
SPI	O	O	O	-	-	-	-
CAN	O	O	O	-	-	-	-
RTC	O	O	O	O	O	-	-
TIMER	O	O	O	-	-	-	-
LPTIMER	O	O	O	O	O	-	-
HSE	O	O	-	-	-	-	-
HSI	O	O	-	-	-	-	-
LSE	O	O	O	O	-	-	-
LSI	O	O	O	O	-	-	-
PLL	O	O	-	-	-	-	-
IWDG	O	O	O	O	O	-	-
WWDG	O	O	O	-	-	-	-
ADC	O	O	-	-	-	-	-
Temperature Sensor	O	O	O	-	-	-	-
OPA	O	O	O	-	-	-	-

COMP1	O	O	O	O	O		
SysTick timer	O	O	O	-	-		
CRC	O	O	O	-	-	-	-
HDIV/SQRT	O	O	O	-	-	-	-
GPIOs	O	O	O	O	O	-	3 pins

Notes:

- (1) Y: Yes (Enable), O: Optional (Disabled by default, Enabled by software), -: Not available.
- (2) The pins that can wake up from the PD are PA0 (WKUP0), PC13 (WKUP1), PA2 (WKUP2), and NRST.

3.2.1 Low Power Run Mode

In LOW POWER RUN(LPRUN) mode, the system clock source can be configured as LSI or LSE by `RCC_LSCTRL.LPRUNCLKSEL`; the PLL is turned off; all peripherals are configurable; the voltage regulator operates in normal mode. When executing programs in SRAM, `PWR_CTRL4.LPRUNFLH` can be configured to make the Flash enter deep standby mode. It cannot be configured if executing in FLASH.

After entering LPRUN mode, the user cannot configure `RCC_CFG.SCLKSW[2:0]` to switch the system clock.

Entering LPRUN mode

Before entering the LPRUN mode, user needs to turn on the LSI or LSE, and select the system clock source in the LPRUN mode through `RCC_LSCTRL.LPRUNCLKSEL`, and then configure `PWR_CTRL4.LPRUNEN = 1`, user can configure the to enter the deep standby mode as required. Note that a write key is required to enable write protection before accessing the `PWR_CTRL4` register.

Exiting LPRUN mode

The software sets `PWR_CTRL4.LPRUNEN = 0` to exit LPRUN mode, user can turn off LSI or LSE and switch system clock as needed. After exiting LPRUN mode, Flash automatically returns to normal mode.

3.2.2 SLEEP Mode

The CPU stops and all peripherals including peripherals around the Cortex[®]-M0 core (such as NVIC, SysTick, etc.) can run and wake up the CPU when an interrupt or event occurs.

Entering SLEEP mode

Entering SLEEP mode by executing WFI (Wait For Interrupt) or WFE (Wait For Event) instruction with `SCB_SCR.SLEEPDEEP = 0`. Depending on the `SCB_SCR.SLEEPONEXIT`, there are two options for SLEEP mode entry:

- SLEEP-NOW: If `SCB_SCR.SLEEPONEXIT = 0`, the WFI or WFE instruction is executed immediately, and the system enters sleep mode immediately.
- SLEEP-ON-EXIT: If `SCB_SCR.SLEEPONEXIT = 1`, the system immediately enters sleep mode when exiting from the lowest priority ISR.

Exiting SLEEP mode

If the WFI instruction is used to enter the SLEEP mode, any NVIC interrupts can wake up the device from the SLEEP mode.

If the WFE instruction is used to enter the SLEEP mode, MCU will exit the SLEEP mode immediately when the event occurs. Wake-up events can be generated in the following ways:

- Enable an interrupt in the peripheral control register instead of NVIC, and enable the SCB_SCR.SEVONPEND. When MCU resumes from WFE, the peripheral interrupt pending bit and the peripheral NVIC interrupt channel suspend bit (in NVIC interrupt clear pending register) must be cleared.
- Configure an external or internal EXTI event mode. When the MCU wakes up, it is not necessary to clear the peripheral interrupt pending bit and the peripheral NVIC interrupt channel pending bit (in the NVIC interrupt clear pending register) because the pending bit corresponding to the event line is not set. This mode provides the shortest wake-up time because there is no time spent on interrupt entry or exit.

3.2.3 STOP Mode

STOP mode is based on the Cortex®-M0 deep sleep mode combined with peripheral clock gating. The voltage regulator operates in low power mode. The output voltage can be configured as 1.5V/1.2V. The HSE/HSI/PLL are turned off. The LSE/LSI can be configured to turn on. All GPIO states, 8KB SRAM and all registers retention. All IO, IWDG and PVD can be used to wake up the CPU. Or other peripherals (RTC/LPUART/LPTIM/COMP) can be configured to wake up. After waking up, the HSI is turned on, and the code starts from where it hangs. Flash is in deep sleep mode.

Entering STOP mode

When entering the STOP mode, user needs to set SCB_SCR.SLEEPDEEP = 1 and PWR_CTRL.PDSTP = 0.

If a Flash operation is in progress, entering STOP mode will be delayed until the memory access is completed.

If the access to the APB area is in progress, entering STOP mode will be delayed until the APB access is completed.

In STOP mode, the following peripherals are available:

- Independent Watchdog (IWDG) optional: Once enabled, it will keep counting until a reset is generated.
- RTC optional: It can be enable by RCC_LSCTRL.RTCEN.
- LPUART/LPTIM/COMP1/PVD peripheral can wake up.
- Internal RC oscillator (LSI RC) optional: It can be turned on by RCC_LSCTRL.LSIEN.
- External 32.768kHz crystal oscillator (LSE OSC) optional: It can be turned on by RCC_LSCTRL.LSEEN.

ADC should be disabled when entering STOP mode to avoid unnecessary power consumption.

Note: If the application needs to disable the external clock before entering the stop mode, it must first switch the system clock to HSI and then deassert RCC_CTRL.HSEEN bit. If RCC_CTRL.HSEEN bit remains asserted and the external clock (external oscillator) is removed when entering the stop mode, the clock safety system (CSS) function must be enabled to detect any external oscillator failure.

Exiting STOP mode

When an interrupt or wake-up event wakes up from STOP mode, the HSI RC oscillator is selected as the system clock, codes resumed from suspended location. Since the voltage regulator is in low-power mode in STOP mode, it consumes more startup time. In addition, users can configure PWR_CTRL4.FLASHWKUP = 1 before entering STOP to shorten the wake-up time of Flash.

3.2.4 Power Down Mode

PD (Power Down) mode is based on Cortex®-M0 deep sleep mode, which can achieve lower power consumption. In this mode, the CPU, all peripherals, voltage regulator, HSE/HSI/PLL/LSE/LSI clock sources and all digital power supplies are

turned off. Except for NRST/PA0/PC13/PA2, most IO ports output High-Z.

Entering PD mode

When entering the PD mode, user needs to set SCB_SCR.SLEEPDEEP = 1 and PWR_CTRL.PDSTP = 1.

If a Flash operation is in progress, entering STOP mode will be delayed until the memory access is completed.

If the access to the APB area is in progress, entering the STOP mode will be delayed until the APB access is completed.

Exiting PD mode

The MCU exits PD mode when external reset (NRST pin), rising/falling edge of WKUP pin event occurs. And all registers will be reset after waking up.

After waking up from PD mode, code execution is same as power on (boot pin is detected, reset vector initialization, etc.).

3.3 Debug Support

By default, if the application puts the MCU in STOP or PD mode while using the debug feature, the debug connection is lost. This is due to the Cortex[®]-M0 core losing its clock.

However, by setting some configuration bits in the DBG_CTRL register, it is possible to debug the software even when in STOP and PD modes. If these register bits are configured, the voltage regulator and HSI will not be disabled or turned off.

3.3.1 Low Power Mode Debug Support

When debugging in low-power mode, ensure that the FCLK of the core is turned on to provide the necessary clock for core debugging. Users can debug the MCU in low power mode according to specific operations (guarantee the output of FCLK in low power mode). For specific operations and features, please refer to the description of DBG_CTRL.PD and DBG_CTRL.STOP in Chapter 3.4.9.

3.3.2 Peripheral Debug Support

In addition to supporting debug in low power mode, it also supports some peripherals to stop working in debug state (TIM1, TIM3, TIM4, TIM6, TIM8, I2C1, I2C2, IWDG, WWDG, CAN). For specific operations and features, please refer to the description of the other bit fields of the DBG_CTRL register in Chapter 3.4.9.

3.4 PWR Registers

3.4.1 PWR Register Overview

Table 3-3 PWR Register Overview

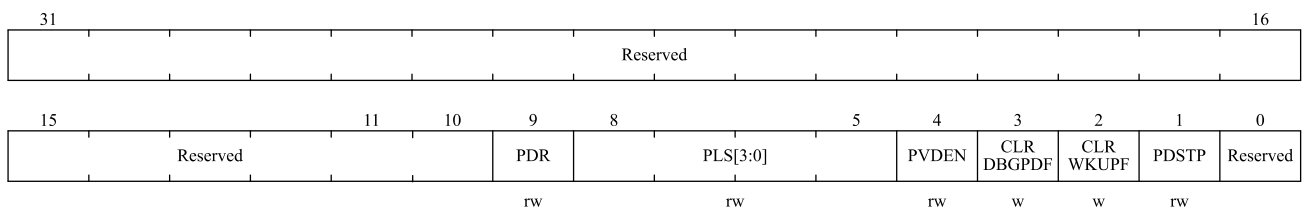
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																			
0x00	PWR_CTRL	Reserved																						PDR	PLS[3:0]				PVDEN	CLRBGPDF	CLRWKUPF	PDSTP	Reserved																			
	Reset value																							1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	PWR_CTRLSTS	Reserved																						WKUPPOL	WKUP2EN	WKUP1EN	WKUP0EN	Reserved										PVDO	DEGPDF	WKUPF												
	Reset value																							1	0	0	0											0	0	0												

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
0x08	PWR_CTRL2	Reserved																					IWDGRSTEN	Reserved															
	Reset value																						1																
0x14	PWR_CTRL3	Reserved																					LSIEN	Reserved															
	Reset value																						0																
0x20	PWR_CTRL4	Reserved																					LPRUNSTS	LPRUNFLH	LPRUNEN	Reserved			STBELH	FLHWKUP									
	Reset value																						0	1	0				0	0									
0x24	PWR_CTRL5	Reserved																					SLPMRSEL[1:0]			Reserved													
	Reset value																						0 0 1																
0x28	PWR_CTRL6	Reserved																					SLPMREN[1:0]			Reserved													
	Reset value																						0 0 0																
0x30	DBG_CTRL	Reserved												TIM8STP	Reserved	TIM6STP	LPTIMSTP	I2C2TIMOUT	I2C1TIMOUT	Reserved	TIM5STP	Reserved	TIM1STP	WWDGSTP	IWDGSTP	Reserved				PD	STOP	Reserved							
	Reset value													0		0	0	0	0		0		0	0	0					0	0								

3.4.2 Power Control Register (PWR_CTRL)

Address offset: 0x00

Reset value: 0x0000 0200



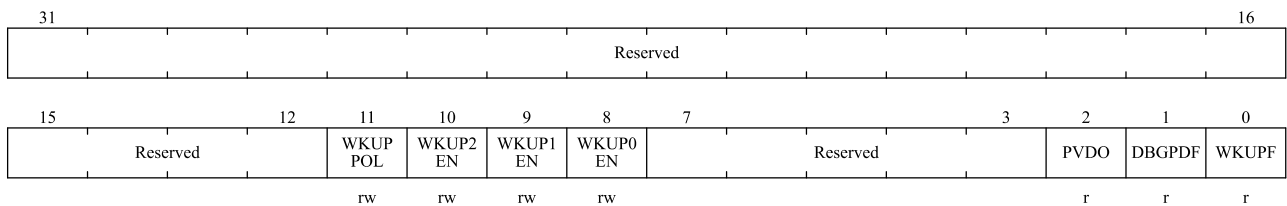
Bit Field	Name	Description								
31:10	Reserved	Reserved, the reset value must be maintained.								
9	PDR	Tune V _{DDD} PDR trigger level during STOP mode. 0: V _{DDD} PDR trigger at 1.0V 1: V _{DDD} PDR trigger at 1.2V Only V _{DDD} POR/PDR can reset this bit.								
8:5	PLS[3:0]	PVD level selection. PVD threshold is controlled below: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>PWR_CTRL.PLS</th> <th>Voltage</th> </tr> </thead> <tbody> <tr> <td>0000</td> <td>1.8v</td> </tr> <tr> <td>0001</td> <td>2.0v</td> </tr> <tr> <td>0010</td> <td>2.2v</td> </tr> </tbody> </table>	PWR_CTRL.PLS	Voltage	0000	1.8v	0001	2.0v	0010	2.2v
PWR_CTRL.PLS	Voltage									
0000	1.8v									
0001	2.0v									
0010	2.2v									

Bit Field	Name	Description																										
		<table border="1"> <tr><td>0011</td><td>2.4v</td></tr> <tr><td>0100</td><td>2.6v</td></tr> <tr><td>0101</td><td>2.8v</td></tr> <tr><td>0110</td><td>3.0v</td></tr> <tr><td>0111</td><td>3.2v</td></tr> <tr><td>1000</td><td>3.4v</td></tr> <tr><td>1001</td><td>3.6v</td></tr> <tr><td>1010</td><td>3.8v</td></tr> <tr><td>1011</td><td>4.0v</td></tr> <tr><td>1100</td><td>4.2v</td></tr> <tr><td>1101</td><td>4.6v</td></tr> <tr><td>1110</td><td>4.8v</td></tr> <tr><td>1111</td><td>5.0v</td></tr> </table>	0011	2.4v	0100	2.6v	0101	2.8v	0110	3.0v	0111	3.2v	1000	3.4v	1001	3.6v	1010	3.8v	1011	4.0v	1100	4.2v	1101	4.6v	1110	4.8v	1111	5.0v
0011	2.4v																											
0100	2.6v																											
0101	2.8v																											
0110	3.0v																											
0111	3.2v																											
1000	3.4v																											
1001	3.6v																											
1010	3.8v																											
1011	4.0v																											
1100	4.2v																											
1101	4.6v																											
1110	4.8v																											
1111	5.0v																											
4	PVDEN	Enable of Power voltage detector. Software control. 0: PVD disabled 1: PVD enabled																										
3	CLRDBGPDF	Clear DBGPD mode flag. Always read as 0. 0: No effect. 1: Clear PWR_CTRLSTS.DBGPDF bit flag after 2 System clock cycles. (write)																										
2	CLRWKUPF	Clear wake up flag. Always read as 0. 0: No effect. 1: Clear PWR_CTRLSTS.WKUPF bit flag after 2 System clock cycles. (write)																										
1	PDSTP	Enter STOP/PD mode selection Software will set and clear this bit. 0: Enter STOP mode when CPU enters deep-sleep mode. 1: Enter PD mode when CPU enters deep-sleep mode.																										
0	Reserved	Reserved, the reset value must be maintained.																										

3.4.3 Power Control Status Register (PWR_CTRLSTS)

Address offset: 0x04

Reset value: 0x0000 0800



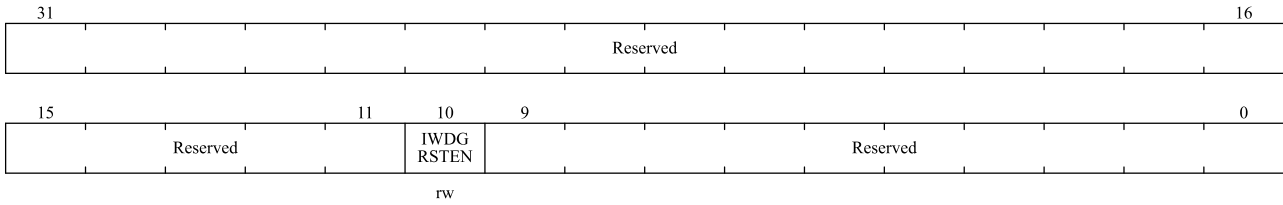
Bit Field	Name	Description
31:12	Reserved	Reserved, the reset value must be maintained.

Bit Field	Name	Description
11	WKUPPOL	Wakeup polarity for PA0/PA2/PC13. To wakeup PD mode by using rising edge or falling edge. Make sure disable wakeup enable before changing polarity value. 0: Falling edge 1: Rising edge
10	WKUP2EN	Enable PA2_WKUP pin Software can set and clear this bit. 0: WKUP pin is used for general purpose I/O. An event on the WKUP pin does not wakeup the device from PD mode. 1: WKUP pin is used for wakeup from PD mode. <i>Note: This bit is reset by V_{DD} POR/PDR Reset only.</i>
9	WKUP1EN	Enable PC13_WKUP pin Software can set and clear this bit. 0: WKUP pin is used for general purpose I/O. An event on the WKUP pin does not wakeup the device from PD mode. 1: WKUP pin is used for wakeup from PD mode. <i>Note: This bit is reset by V_{DD} POR/PDR Reset only.</i>
8	WKUP0EN	Enable PA0_WKUP pin Software can set and clear this bit. 0: WKUP pin is used for general purpose I/O. An event on the WKUP pin does not wakeup the device from PD mode. 1: WKUP pin is used for wakeup from PD mode. <i>Note: This bit is reset by V_{DD} POR/PDR Reset only.</i>
7:3	Reserved	Reserved, the reset value must be maintained.
2	PVDO	PVD output. Hardware will set and clear this bit. It is valid only if PWR_CTRL.PVDEN = 1. 0: V _{DD} /V _{DDA} is higher than the PVD threshold selected with PWR_CTRL.PLS[3:0] 1: V _{DD} /V _{DDA} is lower than the PVD threshold selected with PWR_CTRL.PLS[3:0]
1	DBGPDF	DBGPD mode status bit. When entering DBGPD mode, hardware sets this bit to '1'. Hardware clears this bit when software sets PWR_CTRL.CLRDBGPDF = 1. Only V _{DD} POR/PDR can reset this bit. 0: Chip never entered DBGPD mode 1: Chip has entered DBGPD mode
0	WKUPF	DBGPD mode wake-up status bit. This bit is set by hardware after WKUP pin wakes up DBGPD mode. Hardware clears this bit when software sets PWR_CTRL.CLRWKUPF = 1. Only V _{DD} POR/PDR can reset this bit. 0: No wakeup event occurred 1: A wakeup event was received from the WKUP pin

3.4.4 Power Control Register 2 (PWR_CTRL2)

Address offset: 0x08

Reset value: 0x0000 0400

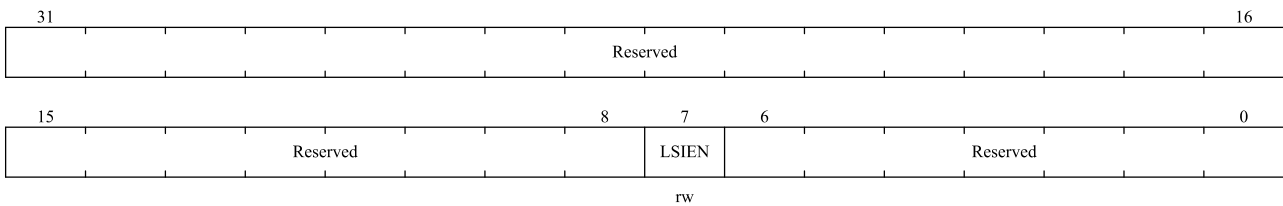


Bit Field	Name	Description
31:11	Reserved	Reserved, the reset value must be maintained.
10	IWDGRSTEN	Independent watchdog reset enable. 0: Independent watchdog cannot generate reset to RCC. 1: Independent watchdog can generate reset to RCC.
9:0	Reserved	Reserved, the reset value must be maintained.

3.4.5 Power Control Register 3 (PWR_CTRL3)

Address offset: 0x14

Reset value: 0x0000 037F



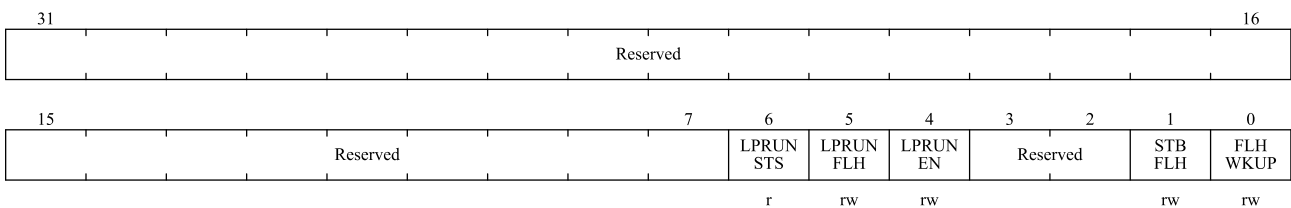
Bit Field	Name	Description
31:8	Reserved	Reserved, the reset value must be maintained.
7	LSIEN	Control PWR to enable LSI. 0: PWR continues requesting LSI clock after system enter STOP mode 1: PWR stops requesting LSI clock after system enter STOP mode
6:0	Reserved	Reserved, the reset value must be maintained.

3.4.6 Power Control Register 4 (PWR_CTRL4)

Address offset: 0x20

Reset value: 0x0000 0020

This register is write protected. Before each software write operation to this register, it must first write the key 0x0175_3603 to this register to release the write protection.

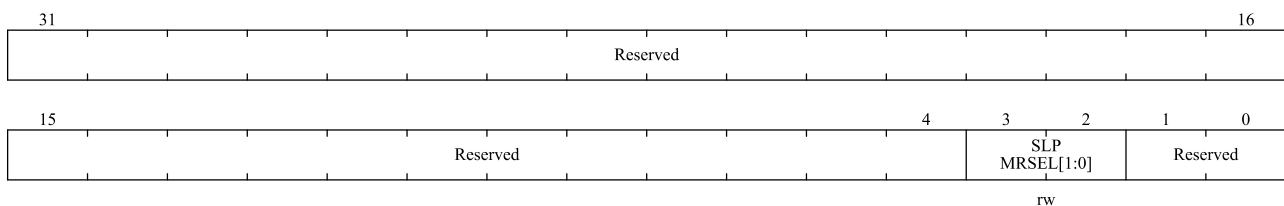


Bit Field	Name	Description
31:7	Reserved	Reserved, the reset value must be maintained.
6	LPRUNSTS	LPRUN mode entry status. This bit is set and cleared by hardware. 0: System exited LPRUN mode 1: System in LPRUN mode
5	LPRUNFLH	Flash low-power control for LPRUN mode. 0: Put Flash to deep standby after chip enters LPRUN mode 1: Keep Flash to normal working state after chip enters LPRUN mode
4	LPRUNEN	LPRUN mode enable This bit is set and cleared by software and can also be cleared by hardware in STOP and PD modes. 0: System exits LPRUN mode 1: System enters LPRUN mode
3:2	Reserved	Reserved, the reset value must be maintained.
1	STBFLH	Flash deep standby mode enable (RUN, can be configured in SLEEP mode) This bit is set and cleared by software and can also be cleared by hardware in STOP and PD modes. 0: Flash back to normal mode 1: Flash enters deep standby mode
0	FLHWKUP	Flash fast wakeup enable 0: Disable fast wake-up when system exits from STOP (wake-up time ~10us) 1: Enable fast wake-up when system exits from STOP (wake-up time ~5us)

3.4.7 Power Control Register 5 (PWR_CTRL5)

Address offset: 0x24

Reset value: 0x0000 0007



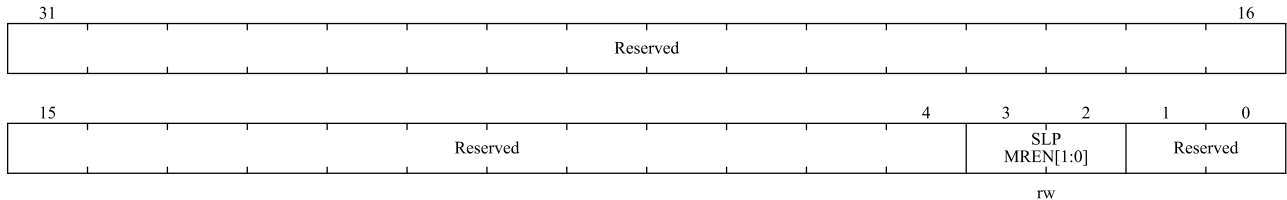
Bit Field	Name	Description
31:4	Reserved	Reserved, the reset value must be maintained.
3:2	SLPMRSEL	V _{DDD} output voltage selection after system enters STOP mode. Before configuring these bits, software must first configure PWR_CTRL6.SLPMREN = '11'. 00: Reserved 01: V _{DDD} output voltage is 1.5V 10: Reserved 11: V _{DDD} output voltage is 1.2V Only V _{DDD} POR/PDR can reset this bit.

Bit Field	Name	Description
1:0	Reserved	Reserved, the reset value must be maintained.

3.4.8 Power Control Register 6 (PWR_CTRL6)

Address offset: 0x28

Reset value: 0x0000 0000



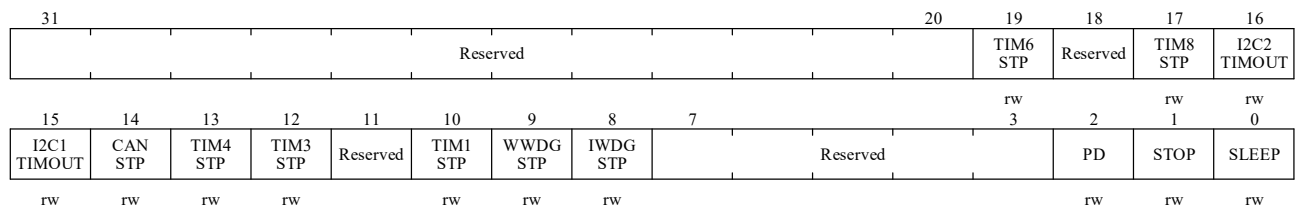
Bit Field	Name	Description
31:4	Reserved	Reserved, the reset value must be maintained.
3:2	SLPMREN	V _{DDD} output voltage selection enable 00: After entering STOP mode, V _{DDD} output voltage remains at 1.5V 01: Reserved 10: Reserved 11: After entering STOP mode, V _{DDD} output voltage is controlled by PWR_CTRL5.SLPMRSEL Only V _{DDD} POR/PDR can reset this bit.
1:0	Reserved	Reserved, the reset value must be maintained.

3.4.9 Debug Control Register (DBG_CTRL)

Address offset: 0x30

Reset value: 0x0000 0000

Only V_{DDD} POR/PDR can reset this register. Only after connecting to the Debugger, software can write access to this register.



Bit Field	Name	Description
31:20	Reserved	Reserved, the reset value must be maintained.
19	TIM6STP	TIM6 stops working when core enters debug state. This bit is set or cleared by software. 0: The counter of TIM6 works normally 1: The counter of TIM6 stops working
18	Reserved	Reserved, the reset value must be maintained.

Bit Field	Name	Description
17	TIM8STP	TIM8 stops working when core enters debug state. This bit is set or cleared by software. 0: The counter of TIM8 works normally 1: The counter of TIM8 stops working
16	I2C2TIMOUT	I2C2 stops SMBUS timeout mode when core is stopped. This bit is set or cleared by software. 0: Same as normal mode operation 1: Frozen the timeout control of SMBUS
15	I2C1TIMOUT	I2C1 stops SMBUS timeout mode when core is stopped. This bit is set or cleared by software. 0: Same as normal mode operation 1: Frozen the timeout control of SMBUS
14	CANSTP	CAN stops working when core enters debug state. This bit is set or cleared by software. 0: CAN still operate normally 1: The CAN receive register does not continue to receive data
13	TIM4STP	TIM4 stops working when core enters debug state. This bit is set or cleared by software. 0: The counter of TIM4 works normally 1: The counter of TIM4 stops working
12	TIM3STP	TIM3 stops working when core enters debug state. This bit is set or cleared by software. 0: The counter of TIM3 works normally 1: The counter of TIM3 stops working
11	Reserved	Reserved, the reset value must be maintained.
10	TIM1STP	TIM1 stops working when core enters debug state. This bit is set or cleared by software. 0: The counter of TIM1 works normally 1: The counter of TIM1 stops working
9	WWDGSTP	WWDG stops working when core enters debug state. This bit is set or cleared by software. 0: The counter of WWDG works normally 1: The counter of WWDG stops working
8	IWDGSTP	IWDG stops working when core enters debug state. This bit is set or cleared by software. 0: The counter of IWDG works normally 1: The counter of IWDG stops working
7:3	Reserved	Reserved, the reset value must be maintained.

Bit Field	Name	Description
2	PD	<p>Debug PD mode control.</p> <p>This bit is set or cleared by software.</p> <p>0: (FCLK off, HCLK off) system enters PD mode, digital circuit part is unpowered. From a software point of view, exiting PD mode is the same as a power-on reset.</p> <p>1: (FCLK on, HCLK on) system enters DBGPD mode, digital circuit part is powered, and FCLK clock is provided by the internal RC oscillator. In addition, the microcontroller exits DBGPD mode by generating a system reset.</p>
1	STOP	<p>Debug STOP mode control.</p> <p>This bit is set or cleared by software.</p> <p>0: (FCLK off, HCLK off) system enters STOP mode, clock controller disables all clocks (including HCLK and FCLK). When exiting STOP mode, the configuration of the clock is the same as after reset (Microcontroller is clocked by the 8MHz internal RC oscillator (HSI)). Therefore, software must reconfigure the clock control system to enable PLL, external oscillator, etc.</p> <p>1: (FCLK on, HCLK on) system enters DBGSTOP mode, FCLK clock is provided by the internal RC oscillator. When exiting DBGSTOP mode, the software must reconfigure the clock control system to enable PLL, external oscillator, etc. (same operation as when configuring this bit to 0).</p>
0	SLEEP	<p>Debug SLEEP mode.</p> <p>Set or cleared by software.</p> <p>0: (FCLK on, HCLK off) In SLEEP mode, FCLK is provided by the previously configured system clock, and HCLK is off. Since SLEEP mode does not reset the configured clock system, software does not need to reconfigure the clock system when exiting from SLEEP mode.</p> <p>1: (FCLK on, HCLK on) In DBGSLEEP mode, both FCLK and HCLK clocks are provided by the previously configured system clock.</p>

4 Reset and Clock Control (RCC)

4.1 Reset Control Unit

N32G032 supports the following two types of reset:

- Power Reset
- System Reset

4.1.1 Power Reset

A power reset occurs in the following circumstances:

- Power-on/ Power-down reset (POR/PDR reset).
- When exiting PD mode.

A power reset sets all registers to their reset values (refer to Figure 3-1).

The reset source will act on the NRST pin, which remains low during reset. The reset entry vector is fixed at address 0x0000_0004. For more details, refer to Table 6-1.

4.1.2 System Reset

Except for the following registers, a system reset will reset all registers to their reset states:

- RCC_CTRL.HSEIOSEL
- RCC_CTRLSTS
- RCC_EMCTRL
- RCC_LSCTRL.LSEBP
- PWR_CTRL.PDR
- PWR_CTRL5
- PWR_CTRL6
- DBG_CTRL

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

- A low level on the NRST pin (external reset)
- Window watchdog(WWDG reset)
- Independent watchdog (IWDG reset)
- Software reset (SW reset)
- Low power management reset
- MMU protection reset
- RAM parity error reset
- EMC reset

The reset source can be identified by checking the reset flags in the Control/Status register (RCC_CTRLSTS).

Software reset

A software reset can be generated by setting the SYSRESETREQ bit in Cortex®-M0 Application Interrupt and Reset Control Register. Refer to Cortex®-M0 technical reference manual for further information.

Low-power management reset

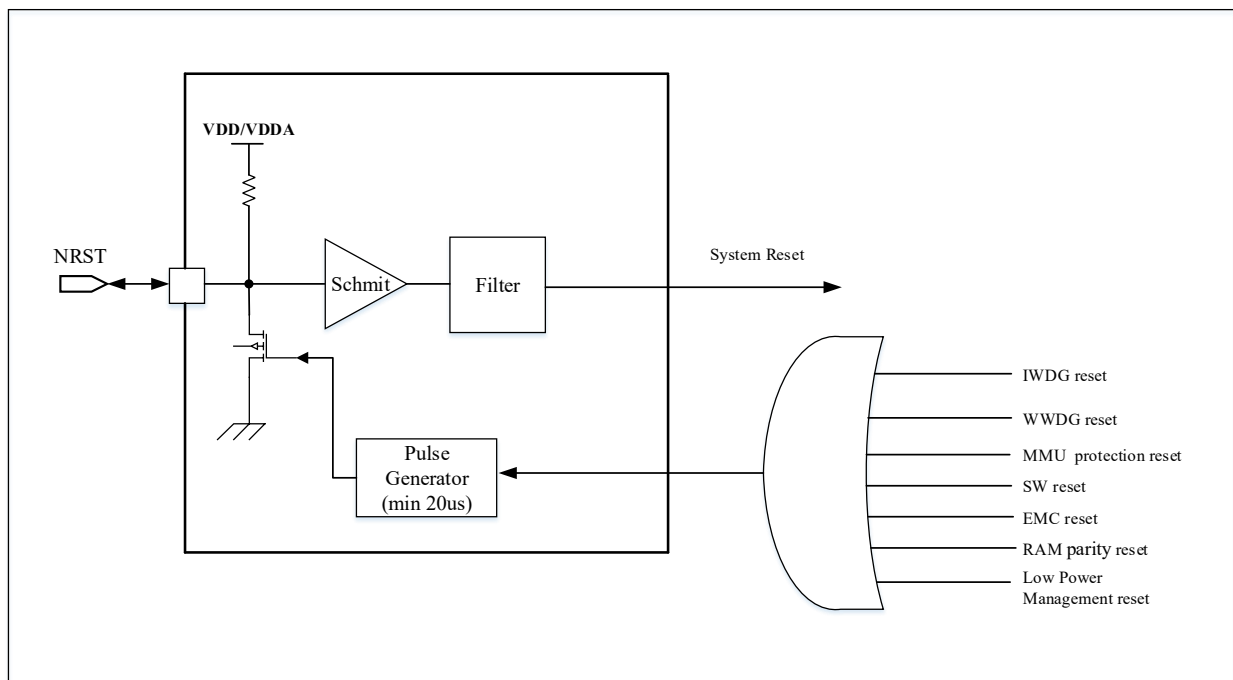
Low-power management reset can be generated by using the following methods:

- Low-power management reset generated when entering PD mode: This reset is enabled by resetting the nRST_PD bit in User Option Bytes. In this case, whenever a PD mode entry sequence is successfully executed, the system is reset instead of entering PD mode.
- Low-power management reset generated when entering STOP modes: This reset is enabled by resetting the nRST_STOP bit in User Option Bytes. In this case, whenever a STOP mode entry sequence is successfully executed, the system is reset instead of entering STOP mode.

The system reset signal provided to the chip is output on the NRST pin. The pulse generator guarantees a minimum reset pulse duration of 20µs for each reset source (external or internal). For external reset, the reset pulse is generated while the NRST pin is asserted low.

The Figure below shows the system reset generation circuit.

Figure 4-1 System Reset Generation



4.2 Clock Control unit

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

- HSI oscillator clock
- HSE oscillator clock
- PLL clock

- LSI oscillator clock
- LSE oscillator clock

The devices have the following two secondary clock sources:

- LSI: 30 kHz low-speed internal RC can be used to drive independent watchdog (IWDG) and drive IWDG,RTC,LPTIMER and LPUART through program selection. Used for Auto-wakeup from STOP mode.
- LSE: 32.768 kHz low-speed external crystal can also be used to drive RTC,LPTIMER and LPUART through program selection.

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

The frequency of the AHB, APB (APB1 and APB2) domains can be configured by the user through multiple prescalers. The maximum allowable frequency of the AHB domain, APB1 domain and APB2 domain is 48MHz.

All peripheral clocks are derived from the system clock (SYSCLK) except in the following cases:

- ADC clock is obtained by dividing the AHB/PLL clock.
- LPUART1 and LPUART2 operating clock can come from one of the following six sources, which can be configured by software:
 - HSI clock
 - HSE Clock
 - LSI clock
 - LSE clock
 - SYSCLK system clock
 - APB1 clock (PCLK)
- LPTIMER operating clock can come from one of the following six sources, which can be configured by software:
 - HSI clock
 - HSE Clock
 - LSI clock
 - LSE clock
 - COMP1_OUT
 - APB1 clock (PCLK)
- RTCCLK clock source can be provided by HSE/128, LSE or LSI clock.
 - LSE clock
 - LSI clock
 - HSE clock divided by 128
- IWDG clock source is LSI oscillator.
- Flash memory programming interface clock source is always the HSI clock

RCC provides external clock for Cortex system timer (SysTick): AHB clock (HCLK) divided by 8. Either the above clock or the Cortex clock (HCLK) to drive the SysTick can be selected by programming the SysTick control and status registers.

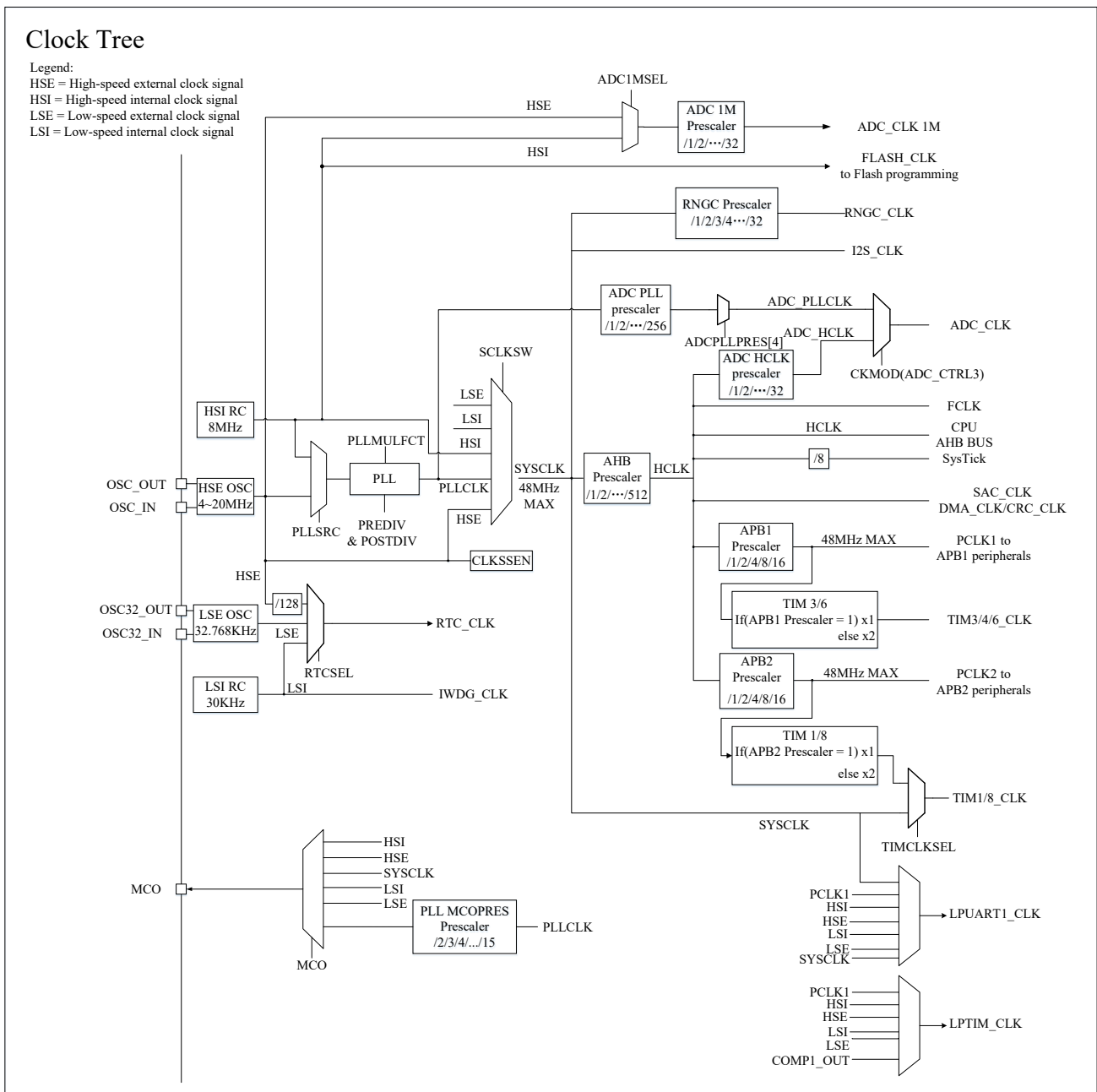
Timer clock frequency distribution is automatically set by hardware in the following 2 cases:

- If the APB prescaler is 1, the timer clock frequency is the same as the APB frequency where the timer resides.
- If the APB prescaler is not 1, the timer clock frequency is twice the APB frequency where the timer is located.

FCLK is the free running clock of the Cortex[®]-M0. See ARM's Cortex[®]-M0 Technical Reference Manual for details.

4.2.1 Clock Tree Diagram

Figure 4-2 Clock Tree



- The maximum frequency available for the system clock is 48MHz.

- For more details about the internal and external clock source characteristics, please refer to the "Electrical Characteristics" section in the product datasheet.

4.2.2 HSE Clock

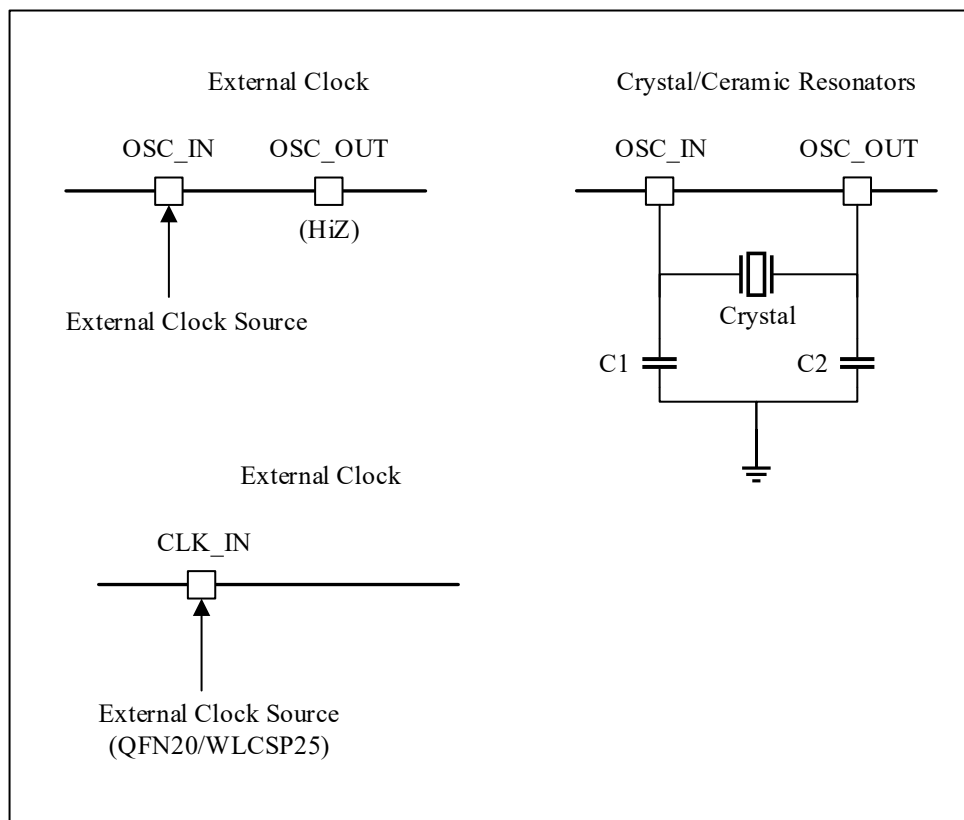
The high-speed external clock signal (HSE) can be generated from the following two clock sources:

- HSE external crystal/ceramic resonator
- HSE user external clock(input through the PA0 or PF0 pin)

In HSE bypass mode or crystal mode, RCC_CTRL.HSEEN needs to be set to 1. If RCC_CTRL.HSEEN=0, HSE will be turned off.

To reduce distortion of the clock output and shorten the start-up stabilize time, the crystal/ceramic resonator and load capacitor must be placed as close as possible to the oscillator pins of the chip. The loading capacitance value must be adjusted according to the chosen oscillator.

Figure 4-3 HSE Clock Source



External clock source (HSE bypass mode)

In this mode, an external clock source must be provided. Its frequency can be up to 20MHz. Users can select this mode by setting the RCC_CTRL.HSEIOSEL and RCC_CTRL.HSEEN bits. When PF0 is used as an external clock signal (square wave, sine wave or triangular wave with 50% duty cycle), it must be connected to the OSC_IN pin, and the OSC_OUT pin must be floating (Hi-Z). See Figure 4-3.

External crystal/ceramic resonator (HSE crystal mode)

The 4 to 20MHz external oscillator has the advantage of producing a more accurate main clock for the system. The associated hardware configuration is shown in Figure 4-3. For more details, please refer to the electrical characteristics

section of the datasheet.

The `RCC_CTRL.HSERDF` bit indicates whether the high-speed external oscillator is stable or not. At startup, the clock is not released until this bit is set by hardware. An interrupt can be generated if enabled the Clock Interrupt Register (`RCC_CLKINT`).

HSE clock can be switched on and off by setting the `RCC_CTRL.HSEEN` bit.

If the user needs to change the configuration of `RCC_CTRL.HSEIOSEL` during operation, it should be configured before enabling `RCC_CTRL.HSEEN`.

4.2.3 HSI Clock

The HSI (High Speed Internal) clock signal is generated by an internal 8MHz RC oscillator and can be directly used as the system clock or PLL input. The HSI RC oscillator can provide a clock source without any external devices. It also has a shorter startup time than the HSE crystal oscillator. However, even with calibration the frequency is less accurate.

RC oscillator frequencies can vary from one chip to another due to manufacturing process variations. Therefore, the HSI clock frequency of the chip has been calibrated to 1% (25°C) before leaving the factory.

If the user application is subject to voltage or temperature variations, this may affect the accuracy of the RC oscillator.

The `RCC_CTRL.HSIRDF` bit flag indicates if the HSI RC oscillator is stable. At startup, the HSI RC output clock is not released until this bit is set by hardware. HSI clock can be switched on and off using the `RCC_CTRL.HSIEN` bit.

If the HSE crystal oscillator fails, the HSI clock can be used as a backup source. Refer to section 4.2.9 clock security system (`CLKSS`).

4.2.4 PLL Clock

The internal PLL can be used to multiply the HSI RC output clock or the HSE crystal output clock. Refer to Figure 4-4. The settings of the PLL (select HSI or HSE as the input clock of the PLL, select the multiplier, select the prescaler and the postscaler) must be completed before it is activated. Once the PLL is activated, these parameters cannot be changed. The PLL can be configured using control bits in `RCC_CTRL` and `RCC_CFG` registers.

When switching the input clock source of the PLL, the original clock source must be turned off after configuring the new clock source (through the clock configuration register bit `RCC_CFG.PLLSRC`).

If the PLL interrupt is enabled in the clock interrupt register, an interrupt request can be generated when the PLL is ready.

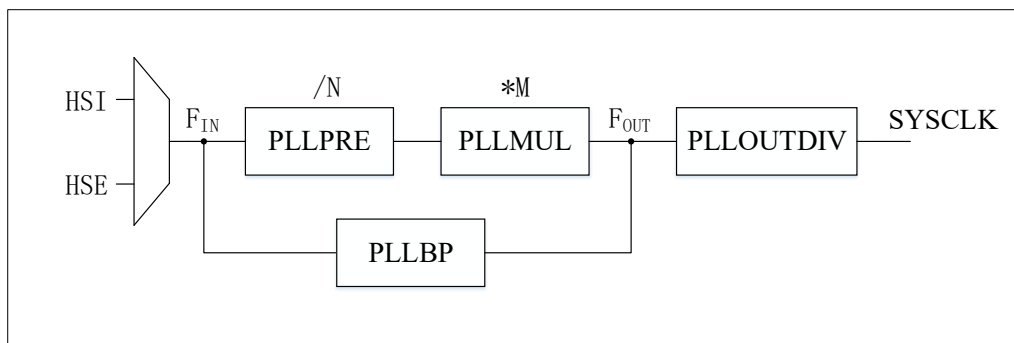
The input frequency F_{IN} range is from 4 to 20MHz.

PLL VCO(F_{OUT}) frequency range requires $48\text{MHz} \leq F_{IN} * M/N < 72\text{MHz}$.

The system frequency is derived from the PLL VCO frequency divided by the postscale factor, and the software needs to be configured correctly to avoid `SYSCLOCK` exceeding 48MHz.

In addition, users can also configure `RCC_CTRL.PLLBP` to bypass the prescaler and frequency multiplication function of PLL.

Figure 4-4 PLL Clock Configuration



4.2.5 LSE Clock

The Low Speed External clock signal (LSE) can be generated from the following two clock sources:

- LSE crystal/ceramic resonator
- LSE external clock(bypass)

LSE crystal clock source

The LSE crystal is a 32.768 kHz low-speed external crystal or ceramic resonator. It provides a low-power and accurate clock source for real-time clock or other timing functions.

The LSE clock can be turned on and off by setting the `RCC_LSCTRL.LSEEN` bit.

The `RCC_LSCTRL.LSERD` bit flag indicates if the LSE clock is stable. At startup, the LSE output clock is not released until this bit is set by hardware. An interrupt can be generated if enabled the Clock Interrupt Register (`RCC_CLKINT`).

LSE external clock source(bypass)

In this mode, an external clock source with a frequency of up to 1 MHz can be provided. Users can select this mode by setting the `RCC_LSCTRL.LSEBP` (when `RCC_LSCTRL.LSEEN` disable) bits. The external clock signal(Square, Sine or Triangle) with 50% duty cycle must be connected to the `OSC32_IN` pin, and the `OSC32_OUT` pin must be left floating (Hi-Z).

4.2.6 LSI Clock

The LSI RC can provide clock for IWDG and AWU in STOP mode. The LSI clock frequency is about 30KHz. Please refer to the electrical characteristics section of the datasheet for further information.

The LSI clock can be turned on or off by the `RCC_CTRLSTS.LSIEN` bit.

The `RCC_CTRLSTS.LSIRD` bit flag indicates if the LSI clock is stable. At startup, the clock is not released until this bit is set by hardware. An interrupt can be generated if enabled the Clock Interrupt Register (`RCC_CLKINT`).

4.2.7 LSI Calibration

The low-speed internal oscillator LSI can be calibrated to compensate for its frequency offset, resulting in an RTC time base with acceptable accuracy, as well as an independent watchdog(IWDG) timeout (when these peripherals use the LSI as a clock source).

Calibration can be achieved by measuring the LSI clock frequency using the input clock of the TIM4 (`TIM4_CLK`).

Measurements are guaranteed with HSE accuracy, and software can adjust the RTC's 20-bit prescaler to obtain the exact RTC clock base, as well as calculate to obtain the exact Independent Watchdog (IWDG) timeout.

The LSI calibration steps are as follows:

1. Turn on TIM4 and set channel 2 to input capture mode;
2. Set the AFIO_CFG.TIM4CH2_RMP bit to 1 to internally connect the LSI to channel 2 of the TIM4;
3. Measure the LSI clock frequency via TIM4 capture/compare events or interrupts;
4. Setting the 20-bit prescaler based on the measurement results and the desired RTC time base and independent watchdog timeout.

4.2.8 System Clock (SYSCLK) Selection

After the system reset, the HSI oscillator is selected as the system clock. The HSI cannot be turned off when the clock source is used as the system clock directly or indirectly through PLL.

A switch from one clock source to another occurs only when the target clock source is ready (after startup delay or PLL locked). If the selected clock source is not ready, the switching of the system clock will not operate until the clock source is ready.

RCC_CFG.SCLKSW[1:0] are used to select the system clock source. Status bits in RCC_CTRL and RCC_LSCTRL indicate which clock is ready, and RCC_CFG indicates which clock is currently used as the system clock.

4.2.9 Clock Security System (CLKSS)

The clock security system can be activated by software by setting the RCC_CTRL.CLKSS bit. Once activated, the clock detector is enabled after the startup delay of the HSE oscillator, and disabled when the HSE clock is turned off.

If the HSE clock fails, the HSE oscillator will be automatically turned off, and a clock failure event will be sent to the break input of the advanced timers (TIM1 and TIM8), and the Clock Security System Interrupt (CLKSSIF) will be generated, allowing the software to execute rescue operations. The CLKSSIF is connected to the NMI (Non-Maskable Interrupt) interrupt of the Cortex[®]-M0.

Once the CSS is activated and the HSE clock fails, the CSS interrupt is generated and the NMI is automatically generated. The NMI will be executed continuously until the CSS interrupt pending bit is cleared. Therefore, it is necessary to clear the CSS interrupt by setting the RCC_CLKINT.CLKSSICLR bit in the NMI handler.

If the HSE oscillator is directly or indirectly used as the system clock (indirectly means: it is used as the PLL input clock, and the PLL clock is used as the system clock), the clock failure will result in the system clock switching to the HSI oscillator and the external HSE oscillator being disabled. If the HSE clock (divided or not) is selected as PLL input clock, the PLL will be turned off upon HSE clock failure.

4.2.10 RTC Clock

By programming RCC_LSCTRL.RTCSEL[1:0] bits, the RTCCLK can be provided by the HSE/128, LSE, or LSI clocks.

4.2.11 Watchdog Clock

If the independent watchdog has been started by hardware option or software, the LSI oscillator will be forced to turn on and cannot be disabled. The clock is supplied to the IWDG after the LSI oscillator is stable.

4.2.12 LPUART Clock

In normal operating mode, the LPUART clock supports six clock sources: HSI, HSE, LSI, LSE, SYS_CLK and LPUART_PCLK. Since HSI, HSE, SYSCLK and PCLK will be turned off in low power mode, software should switch the LPUART clock to LSI or LSE before entering low power mode..

4.2.13 LPTIME Clock

The LPTIME clock in normal operating mode supports six clock sources: HSI, HSE, LSI, LSE, PCLK1 and COMP1_OUT. In low-power mode since HSI, HSE, PCLK will be turned off, the software should switch the LPTIMER clock to LSI, LSE or COMP1_OUT before entering low-power mode.

HSI, HSE, LSI, LSE, PCLK1 switch to each other without burrs, software should make sure both clocks are turned on before switching.

When switching from HSI, HSE, LSI, LSE, PCLK1 to COMP1_OUT, software cooperation is required:

1. Make sure the clock is turned on before switching.
2. Set COMP1_CTRL.EN = 0 to ensure that COMP1 is turned off.
3. Set RCC_APB1PCLKEN.LPTIMEN = 1 to enable LPTIM.
4. Set RCC_CFG2.LPTIMSEL = 5 to select COMP1_OUT as clock source
5. Set COMP1_CTRL.EN = 1 to enable COMP1

4.2.14 Clock Output(MCO)

The microcontroller clock output (MCO) capability allows the clock signal to be output to the external MCO pin.

The corresponding GPIO port register must be configured for the corresponding function. The following six clock signals can be selected as the MCO clock:

- SYSCLK
- HSI
- HSE
- LSI
- LSE
- PLL clock division

The clock selection is controlled by RCC_CFG.MCO[2:0] bits.

4.3 RCC Registers

4.3.1 RCC Register Overview

Table 4-1 RCC Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																												
000h	RCC_CTRL	Reserved				HSEPADSEL[1:0]		PLLRDF		PLLEN		PLLOUTEN		PLLBP		Reserved		CLKSSEN		Reserved		HSERDF		HSEEN		Reserved												Reserved		HSIRDIF		HSIEN																			
	Reset Value	0				0		0		0		1		0		0		0		0		0		0		0		0												1		1																			
004h	RCC_CFG	MCORES[3:0]				MCO[2:0]				PLLSRC		PLLOUTDIV[1:0]		PLLPRE[1:0]		PLLMULFCT[3:0]				SCLKSTS2		APB2PRES[2:0]				APB1PRES[2:0]				AHBPRES[3:0]				SCLKSTS				SCLKSW[2:0]																							
	Reset Value	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																											
008h	RCC_CLKINT	Reserved												CLKSSI CLR		Reserved		PERRCLR		PLLRDCLR		HSERDCLR		HSIRDCLR		LSERDCLR		LSIRDCLR		Reserved		RAMCERRRST		RAMCERRIEN		PLLRDIEN		HSERDIEN		HSIRDIEN		LSERDIEN		LSIRDIEN		CLKSSIF		Reserved		PERRF		PLLRDIF		HSERDIF		HSIRDIF		LSERDIF		LSIRDIF	
	Reset Value	0												0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0							
00Ch	RCC_APB2PRST	Reserved																UART6RST		Reserved		USART1RST		TIM8RST		TIM1RST		SPI3RST		SPI2RST		SPI1RST		Reserved		IOPFRST		Reserved		IOPDRST		IOPCRST		IOPBRST		IOPARST		Reserved		AFIORST											
	Reset Value	0																0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0											
010h	RCC_APB1PRST	Reserved		PWRRST		Reserved		CANRST		Reserved				I2C2RST		I2C1RST		UART5RST		LPUART4RST		LPUART3RST		USART2RST		Reserved				WWDGRST		Reserved				BEEP2RST		BEEP1RST		TIM6RST		LPTIM5RST		TIM4RST		TIM3RST		Reserved													
	Reset Value	0		0		0		0				0		0		0		0		0		0		0		0				0		0				0		0		0		0		0		0															
014h	RCC_AHBCLKEN	Reserved																ADCIEN		SACEN		Reserved		RNGCEN		Reserved		HDIVEN		CRCEN		HSQRTEN		FLITFEN		Reserved		SRAMEN		Reserved		DMA1EN																			
	Reset Value	0																0		0		0		0		0		0		0		0		0		0		0		0		0																			
018h	RCC_APB2PCKEN	Reserved																UART6EN		Reserved		USARTIEN		TIM8EN		TIMIEN		SPI3EN		SPI2EN		SPI1EN		Reserved		IOPPEN		Reserved		IOPDEN		IOPCEN		IOPBEN		IOPAEN		Reserved		AFIOEN											
	Reset Value	0																0		0		0		0		0		0		0		0		0		0		0		0		0		0		0															
01Ch	RCC_APB1PCKEN	OPAEN		Reserved		PWREN		Reserved		CANEN		Reserved				I2C2EN		I2C1EN		UART5EN		LPUART4EN		LPUART3EN		USART2EN		Reserved				WWDGEN		Reserved		COMPILTEN		COMPEN		Reserved		BP2CLKEN		BP1CLKEN		TIM6EN		LPTIM5EN		TIM4EN		TIM3EN		Reserved							
	Reset Value	0		0		0		0		0				0		0		0		0		0		0		0		0				0		0		0		0		0		0		0		0		0													
020h	RCC_LSCTRL	Reserved																Reserved		LPRUNCLKSEL		RTCRST		RTCEN		Reserved		RTCSEL[1:0]		LSEBYP		LSERD		LSEEN		LSIRD		LSIEN																							
	Reset Value	0																0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0											
024h	RCC_CTRLSTS	Reserved																EMCCLPRSTF		EMCGBRSTF		LPWRRSTF		WWDGRSTF		IWDGRSTF		SFTIRSTF		PORRSTF		PINRSTF		MMURSTF		RAMRSTF		RMRSTF																							
	Reset Value	0																0		0		0		0		0		0		0		0		0		0		0		0		0		0		0															

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
028h	RCC_AHBPRST	Reserved																				ADCIRST	SACRST	Reserved	RANGRST	Reserved	HDIRVST	Reserved	HSQRTRST	Reserved							
	Reset Value																					0	0		0		0		0								
02Ch	RCC_CFG2	TIMCLK	LPUART4SEL[2:0]				LPUART3SEL[2:0]				RANGEN	LPTIM5SEL[2:0]				RANGPRE[4:0]				ADC1MPRES[4:0]				ADC1MSEL	Reserved	ADCPLLPRES[4:0]				ADCHPRES [3:0]							
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0		0	0	0	0	0	0	0	0	0			
030h	RCC_EMCCTRL	Reserved										GBRST3	GBRST2	GBRST1	GBRST0	GBNRST3	GBNRST2	GBNRST1	GBNRST0	CLPRST3	CLPRST2	CLPRST1	CLPRST0	GBDET3	GBDET2	GBDET1	GBNDET0	CLPDET3	CLPDET2	CLPDET1	CLPDET0						
	Reset Value											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

4.3.2 Clock Control Register (RCC_CTRL)

Address offset: 0x00

Reset value: 0x0080 0003

31	28	27	26	25	24	23	22	21	20	19	18	17	16		
Reserved				HSEIOSEL[1:0]	PLL RDF	PLLEN	PLL OUTEN	PLLBP	Reserved			CLKSSEN	Reserved	HSERDF	HSEEN
				rw	r	rw	rw	rw				rw		r	rw
Reserved												HSIRD F	HSIEN		
												r	rw		

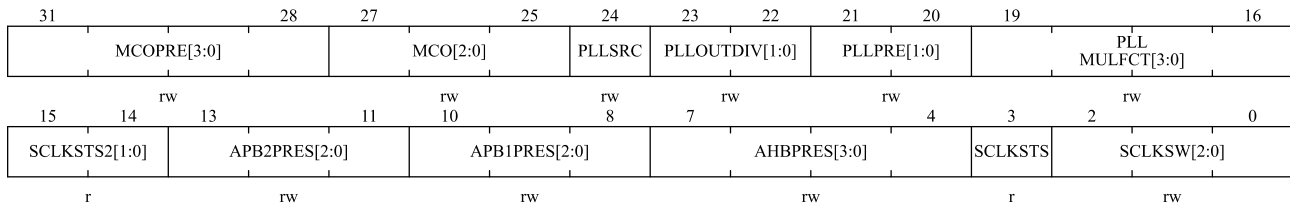
Bit Field	Name	Description
31:28	Reserved	Reserved, the reset value must be maintained
27:26	HSEIOSEL[1:0]	HSE OSC_IN pin selection 00:Using PA0 digital input for HSE OSC_IN, HSE shall be configured for external clock mode; 01:Using PF0 digital input for HSE OSC_IN, HSE should be configured for external clock mode; 10:Using PF0 analog input for HSE OSC_IN, HSE shall be configured for external crystal mode; 11:Reserved This bit cannot be written when HSEEN = 1 This bit cannot be reset by a system reset
25	PLL R DF	PLL clock ready flag Set by hardware once PLL is ready. 0: PLL is not ready 1: PLL is ready
24	PLLEN	PLL enable Set and cleared by software. When entering the LPRUN,STOP or PD mode, it is cleared by hardware. This bit cannot be cleared when PLL is used as the system clock. 0: Disable PLL

Bit Field	Name	Description
		1: Enable PLL
23	PLLOUTEN	PLL clock output enable bit. 0: PLL clock output is disabled 1: PLL clock output is enabled
22	PLLBP	PLL bypass mode 0: $F_{OUT} = F_{IN} * M/N$ (PLL VCO frequency) 1: $F_{OUT} = F_{IN}$ (bypass output)
21:20	Reserved	Reserved, the reset value must be maintained.
19	CLKSSEN	Clock security system enable Set and cleared by software. 0: Disable the clock detector 1: Enable the clock detector if the HSE oscillator is ready
18	Reserved	Reserved, the reset value must be maintained.
17	HSERDF	External high-speed clock ready flag Set by hardware once HSE is ready. This bit is cleared 6 HSE clock cycles after the HSEEN bit is cleared. 0: HSE is not ready 1: HSE is ready
16	HSEEN	External high-speed clock enable Set and cleared by software. When entering the LPRUN,STOP or PD mode, it is cleared by hardware. This bit cannot be cleared when HSE is used as the system clock. 0: Disable HSE oscillator 1: Enable HSE oscillator
15:2	Reserved	Reserved, the reset value must be maintained.
1	HSIRDF	Internal high-speed clock ready flag Set by hardware once HSI is stable. After the HSIEN bit is cleared, it takes 6 internal 8 MHz oscillator clock cycles to go low.
0	HSIEN	Internal high-speed clock enable Set and cleared by software. This bit cannot be cleared when HSI is used as the system clock. When returning from LPRUN,STOP or PD mode or HSE failure occurs, set by hardware to enable the HSI oscillator. 0: Disable HSI oscillator 1: Enable HSI oscillator

4.3.3 Clock Configuration Register (RCC_CFG)

Address offset: 0x04

Reset value: 0x2000 0000



Bit Field	Name	Description
31:28	MCOPRES[3:0]	MCO prescaler Set and cleared by software. 0010: PLL clock divided by 2 0011: PLL clock divided by 3 0100: PLL clock divided by 4 0101: PLL clock divided by 5 0110: PLL clock divided by 6 0111: PLL clock divided by 7 1000: PLL clock divided by 8 1001: PLL clock divided by 9 1010: PLL clock divided by 10 1011: PLL clock divided by 11 1100: PLL clock divided by 12 1101: PLL clock divided by 13 1110: PLL clock divided by 14 1111: PLL clock divided by 15 Other values: not allowed
27:25	MCO[2:0]	Microcontroller clock output selection Set and cleared by software. 000: no clock 001: LSI clock 010: LSE clock 011: System clock (SYSCLK) 100: HSI clock 101: HSE clock 110: Clock after PLL prescaler <i>Note: This clock output may be truncated at startup or during MCO clock source switching. When the system clock is selected to output to the MCO pin, the output clock frequency must not exceed the maximum I/O speed (For details of the maximum frequency of the I/O port, refer to the data sheet).</i>
24	PLLSRC	PLL clock source. Set or cleared by software, this bit can be written only when the PLL is turned off. 0: HSI clock is used as PLL input clock 1: HSE clock is used as PLL input clock
23:22	PLLOUTDIV[1:0]	The PLL outputs the clock division value. Set and clear through software. 00: no prescaler

Bit Field	Name	Description
		01: divided by 2 10: divided by 3 11: divided by 4
21:20	PLLPRE[1:0]	PLL prescaler 4MHz <= F _{IN} /N <= 20MHz This bit can only be written when the PLL is off 00: PLL input clock divided by 1 01: PLL input clock divided by 2 10: PLL input clock divided by 3 11: PLL input clock divided by 4
19:16	PLLMULFCT[3:0]	PLL multiplication factor. Set and clear through software. This bit can only be written when the PLL is off. F _{OUT} = F _{IN} *M /N The actual PLL M value should be this register value + 3, M = PLLMULFCT + 3. 0: M = 3 1: M = 4 2: M = 5 ... 15: M = 18
15:14	SCLKSTS2[1:0]	Use with SCLKSTS bit
13:11	APB2PRES[2:0]	APB high-speed (APB2) prescaler Set and cleared by software to configure the division factor of APB2 clock (PCLK2). Make sure that PCLK2 does not exceed 48MHz. 0xx: HCLK not divided 100: HCLK divided by 2 101: HCLK divided by 4 110: HCLK divided by 8 111: HCLK divided by 16
10:8	APB1PRES[2:0]	APB low-speed (APB1) prescaler Set and cleared by software to configure the division factor of the APB1 clock (PCLK1). Make sure that PCLK1 does not exceed 48MHz. 0xx: HCLK not divided 100: HCLK divided by 2 101: HCLK divided by 4 110: HCLK divided by 8 111: HCLK divided by 16
7:4	AHBPRES[3:0]	AHB prescaler Set and cleared by software to configure the division factor of the AHB clock (HCLK). 0xxx: SYSCLK not divided 1000: SYSCLK divided by 2 1001: SYSCLK divided by 4 1010: SYSCLK divided by 8

Bit Field	Name	Description
		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
3	SCLKSTS	System clock switch status, used together with SCLKSTS2[1:0] bit Set and cleared by hardware to indicate which clock source is used as system clock 000: HSI oscillator used as system clock 001: HSE oscillator used as system clock 010: PLL used as system clock 011: LSE used as system clock 100: LSI used as system clock
2:0	SCLKSW[2:0]	System clock switch Set and cleared by software to select the system clock source. Set by hardware to force HSI selection when exiting from the STOP mode, or when the HSE oscillator fails and CLKSSSEN is enabled. 000: HSI selected as the system clock 001: HSE selected as the system clock 010: PLL selected as the system clock 011: LSE selected as the system clock 100: LSI selected as the system clock

4.3.4 Clock Interrupt Register (RCC_CLKINT)

Address offset: 0x08

Reset value: 0x0000 0000

31										24				23	22	21	20	19	18	17	16
Reserved										CLKSSI CLR	Reserved	PERR CLR	PLLRDI CLR	HSERDI CLR	HSIRDI CLR	LSERDI CLR	LSIRDI CLR				
										w		w	w	w	w	w	w	w			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Reserved	RAM CERRRST	RAM CERRIEN	PLLRDI EN	HSERDI EN	HSIRDI EN	LSERDI EN	LSIRDI EN	CLKSSIF	Reserved	RAMCPIF	PLLRDIF	HSERDIF	HSIRDIF	LSERDIF	LSIRDIF						
	rw	rw	rw	rw	rw	rw	rw	r		r	r	r	r	r	r						

Bit Field	Name	Description
31:24	Reserved	Reserved, the reset value must be maintained.
23	CLKSSICLR	Clock security system interrupt clear Set by the software to clear the CLKSSIF flag. 0: No effect 1: Clear the CLKSSIF flag
22	Reserved	Reserved, the reset value must be maintained.
21	PERRCLR	PERRCLR: Clears PERR interrupts. This bit is set by the software to clear PERRF. 0: No impact. 1: PERRF cleared

Bit Field	Name	Description
20	PLLRDICLR	PLL ready interrupt clear Set by the software to clear the PLLRDIF flag. 0: No effect 1: Clear the PLLRDIF flag
19	HSERDICLR	HSE ready interrupt clear Set by the software to clear the HSERDIF flag. 0: Not used 1: Clear HSERDIF flag
18	HSIRDICLR	HSI ready interrupt clear Set by the software to clear the HSIRDIF flag. 0: Not used 1: Clear the HSIRDIF flag
17	LSERDICLR	LSE ready interrupt clear Set by the software to clear the LSERDIF flag. 0: Not used 1: Clear LSERDIF flag
16	LSIRDICLR	LSI ready interrupt clear Set by software to clear the LSIRDIF flag. 0: Not used 1: Clear the LSIRDIF flag
15	Reserved	Reserved, the reset value must be maintained.
14	RAMCERRRST	RAMC parity error reset enabled 1: RAMC generates a reset when it detects a parity error 0: The reset is not generated when RAMC detects a parity error
13	RAMCERRIEN	Enable RAMC parity error interrupt 1: Interrupts when RAMC detects a parity error 0: No interrupt is generated when RAMC detects a parity error
12	PLLRDIEN	PLL ready interrupt enable Set and cleared by software to enable and disable PLL ready interrupt 0: Disable PLL ready interrupt 1: Enable PLL ready interrupt
11	HSERDIEN	HSE ready interrupt enable Set and cleared by software to enable and disable HSE ready interrupt. 0: Disable HSE ready interrupt 1: Enable HSE Ready Interrupt
10	HSIRDIEN	HSI ready interrupt enable Set and cleared by software to enable and disable HSI ready interrupt. 0: Disable HSI ready interrupt 1: Enable HSI ready interrupt
9	LSERDIEN	LSE ready interrupt enable Set and cleared by software to enable and disable LSE ready interrupt. 0: Disable LSE ready interrupt 1: Enable LSE ready interrupt

Bit Field	Name	Description
8	LSIRDIEN	LSI ready interrupt enable Set and cleared by software to enable and disable LSI ready interrupt. 0: Disable LSI ready interrupt 1: Enable LSI ready interrupt
7	CLKSSIF	Clock security system interrupt flag Set by hardware when a failure is detected in the external HSE oscillator. 0: No clock security system interrupt caused by HSE clock failure 1: Clock security system interrupt caused by HSE clock failure
6	Reserved	Reserved, the reset value must be maintained.
5	RAMCPIF	RAMC parity interrupt status. Set by hardware, set by software to clear PERRCLR 1: RAMC parity error occurs 0: No RAMC parity error occurs
4	PLLRDIF	PLL ready interrupt flag This bit is set by hardware when PLLRDIEN is set and PLL clock is ready. This bit is cleared by software by setting the PLLRDICLR bit. 0: No clock ready interrupt caused by PLL lock 1: Clock ready interrupt caused by PLL lock
3	HSERDIF	HSE ready interrupt flag Set by hardware when HSERDIEN is set and the HSE clock is ready. This bit is cleared by software by setting the HSERDICLR bit. 0: No clock ready interrupt caused by HSE oscillator 1: Clock ready interrupt caused by HSE oscillator
2	HSIRDIF	HSI ready interrupt flag Set by hardware when HSIRDIEN is set and the HSI clock is ready. This bit is cleared by software by setting the HSERDICLR bit. 0: No clock ready interrupt caused by HSI oscillator 1: Clock ready interrupt caused by HSI oscillator
1	LSERDIF	LSE ready interrupt flag Set by hardware when LSERDIEN is set and the LSE clock is ready. This bit is cleared by the software by setting the LSERDICLR bit. 0: No clock ready interrupt caused by LSE oscillator 1: Clock ready interrupt caused by LSE oscillator
0	LSIRDIF	LSI ready interrupt flag Set by the hardware when LSIRDIEN is set and the LSI clock is ready. This bit is cleared by software by setting the LSIRDICLR bit. 0: No clock ready interrupt caused by LSI oscillator 1: Clock ready interrupt caused by LSI oscillator

4.3.5 APB2 Peripheral Reset Register (RCC_APB2PRST)

Address offset: 0x0c

Reset value: 0x0000 0000

Reserved														UART6 RST	Reserved
Reserved	USART1 RST	TIM8RST	TIM1RST	SPI3RST	SPI2RST	SPI1RST	Reserved	IOPFRST	Reserved	IOPDRST	IOPCRST	IOPBRST	IOPARST	Reserved	AFIORST
	rw	rw	rw	rw	rw	rw		rw		rw	rw	rw	rw		rw

Bit Field	Name	Description
31:18	Reserved	Reserved, the reset value must be maintained.
17	UART6RST	UART6 reset Set and cleared by software. 0: Clear reset 1: Reset UART6
16:15	Reserved	保留，必须保持复位值。
14	USART1RST	USART1 reset Set and cleared by software. 0: Clear reset 1: Reset USART1
13	TIM8RST	TIM8 reset Set and cleared by software. 0: Clear reset 1: Reset TIM8
12	TIM1RST	TIM1 reset Set and cleared by software. 0: Clear reset 1: Resets TIM1
11	SPI3RST	SPI3 reset Set and cleared by software. 0: Clear reset 1: Reset SPI3
10	SPI2RST	SPI2 reset Set and cleared by software. 0: Clear reset 1: Reset SPI2
9	SPI1RST	SPI1 reset Set and cleared by software. 0: Clear reset 1: Reset SPI1
8	Reserved	Reserved, the reset value must be maintained.
7	IOPFRST	GPIO port F reset Set and cleared by software. 0: Clear reset 1: Reset GPIO port F
6	Reserved	Reserved, the reset value must be maintained.

Bit Field	Name	Description
5	IOPDRST	GPIO port D reset Set and cleared by software. 0: Clear reset 1: Reset GPIO port D
4	IOPCRST	GPIO port C reset Set and cleared by software. 0: Clear reset 1: Reset GPIO port C
3	IOPBRST	GPIO port B reset Set and cleared by software. 0: Clear reset 1: Reset GPIO port B
2	IOPARST	GPIO port A reset Set and cleared by software. 0: Clear reset 1: Reset GPIO port A
1	Reserved	Reserved, the reset value must be maintained.
0	AFIORST	Alternate function IO reset Set and cleared by software. 0: Clear reset 1: Reset alternate function IO

4.3.6 APB1 Peripheral Reset Register (RCC_APB1RST)

Address offset: 0x10

Reset value: 0x0000 0000

31	Reserved	29	28	PWR RST	27	Reserved	26	CAN RST	25	Reserved	23	22	I2C2RST	21	I2C1RST	20	UART5 RST	19	LPUART2 RST	18	LPUART1 RST	17	USART2 RST	16	Reserved
15	Reserved	12	11	rw	10	rw	7	6	5	4	3	2	1	0	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
				WWDG RST		Reserved		BEEP2 RST	BEEP1 RST	TIM6RST	LPTIM RST	TIM4RST	TIM3RST	Reserved	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Bit Field	Name	Description
31:29	Reserved	Reserved, the reset value must be maintained
28	PWRRST	Power interface reset Set and cleared by software. 0: Clear reset 1: Reset the power interface
27	Reserved	Reserved, the reset value must be maintained
26	CANRST	CAN reset Set and cleared by software. 0: Clear reset

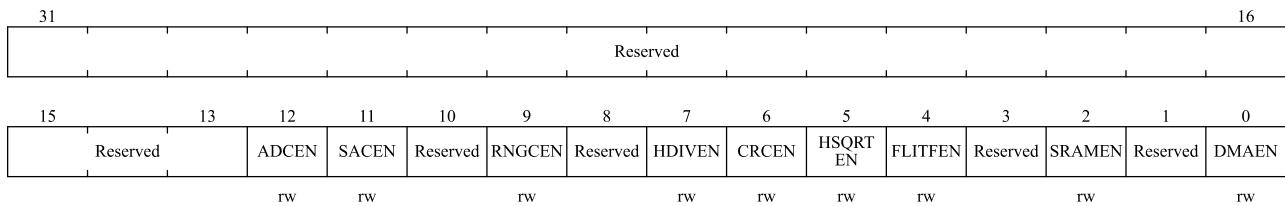
Bit Field	Name	Description
		1: Reset CAN
25:23	Reserved	Reserved, the reset value must be maintained
22	I2C2RST	I2C2 reset Set and cleared by software. 0: Clear reset 1: Reset I2C2
21	I2C1RST	I2C1 reset Set and cleared by software. 0: Clear reset 1: Reset I2C1
20	UART5RST	UART5 reset Set and cleared by software. 0: Clear reset 1: Reset UART5
19	LPUART2RST	LPUART2 reset Set and cleared by software. 0: Clear reset 1: Reset LPUART2
18	LPUART1RST	LPUART1 reset Set and cleared by software. 0: Clear reset 1: Reset LPUART1
17	USART2RST	USART2 reset Set and cleared by software. 0: Clear reset 1: Reset USART2
16:12	Reserved	Reserved, the reset value must be maintained
11	WWDGRST	Window watchdog reset Set and cleared by software. 0: Clear reset 1: Reset window watchdog
10:7	Reserved	Reserved, the reset value must be maintained
6	BEEP2RST	Beeper2 reset Set and cleared by software. 0: Clear reset 1: Reset Beeper2
5	BEEP1RST	Beeper1 reset Set and cleared by software. 0: Clear reset 1: Reset Beeper1
4	TIM6RST	TIM6 timer reset Set and cleared by software. 0: Clear reset

Bit Field	Name	Description
		1: Reset TIM6 timer
3	LPTIMRST	LPTIMR timer reset Set and cleared by software. 0: Clear reset 1: Reset LPTIM
2	TIM4RST	TIM4 timer reset Set and cleared by software. 0: Clear reset 1: Reset TIM4 timer
1	TIM3RST	TIM3 timer reset Set and cleared by software. 0: Clear reset 1: Reset TIM3 timer
0	Reserved	Reserved, the reset value must be maintained.

4.3.7 AHB Peripheral Clock Enable Register (RCC_AHBPCLEN)

Address offset: 0x14

Reset value: 0x0000 0014



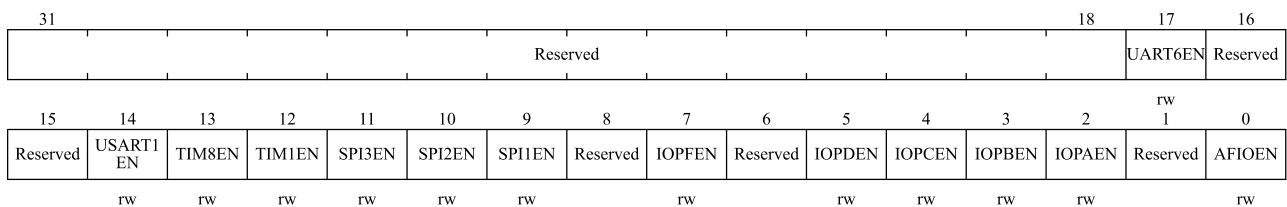
Bit Field	Name	Description
31:13	Reserved	Reserved, the reset value must be maintained.
12	ADCEN	ADC clock enable Set and cleared by software. 0: Disable ADC clock 1: Enable ADC clock
11	SACEN	SAC clock enable Set and cleared by software. 0: Disable SAC clock 1: Enable SAC clock
10	Reserved	Reserved, the reset value must be maintained.
9	RNGCEN	RNG clock enable Set and cleared by software. 0: Disable RNG clock 1: Enable RNG clock
8	Reserved	Reserved, the reset value must be maintained.
7	HDIVEN	HDIV clock enable

Bit Field	Name	Description
		Set and cleared by software. 0: Disable the HDIV clock 1: Enable the HDIV clock.
6	CRCEN	CRC clock enable Set and cleared by software. 0: Disable CRC clock 1: Enable CRC clock
5	HSQRTEN	HSQRT clock enable Set and cleared by software. 0: Disable HSQRT clock 1: Enable HSQRT clock
4	FLITFEN	Flash interface clock enable Set and cleared by software. 0: Disable the Flash interface clock 1: Enable the Flash interface clock
3	Reserved	Reserved, the reset value must be maintained.
2	SRAMEN	SRAM clock enable Set and cleared by software. 0: SRAM clock disabled in sleep mode 1: SRAM clock enabled in sleep mode
1	Reserved	Reserved, the reset value must be maintained.
0	DMAEN	DMA clock enable Set and cleared by software. 0: Disable DMA clock 1: Enable DMA clock

4.3.8 APB2 Peripheral Clock Enable Register (RCC_APB2PCLKEN)

Address offset: 0x18

Reset value: 0x0000 0000



Bit Field	Name	Description
31:18	Reserved	Reserved, the reset value must be maintained.
17	UART6EN	UART6 clock enable Set and cleared by software. 0: Disable UART6 clock 1: Enable UART6 clock

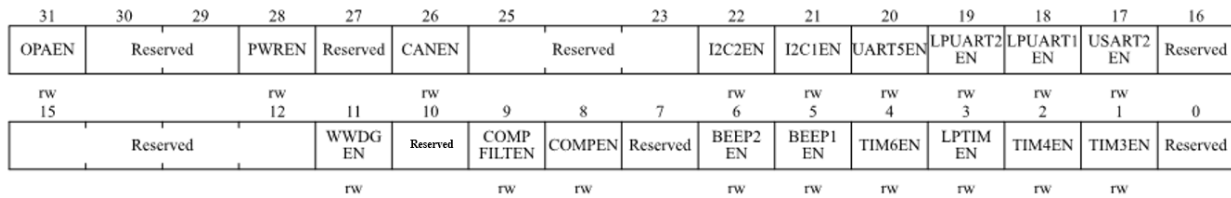
Bit Field	Name	Description
16:15	Reserved	Reserved, the reset value must be maintained.
14	USART1EN	USART1 clock enable Set and cleared by software. 0: Disable USART1 clock 1: Enable USART1 clock
13	TIM8EN	TIM8 Clock Enable Set and cleared by software. 0: Disable TIM8 clock 1: Enable TIM8 clock
12	TIM1EN	TIM1 clock enable Set and cleared by software. 0: Disable TIM1 clock 1: Enable TIM1 clock
11	SPI3EN	SPI3 clock enable Set and cleared by software. 0: Disable SPI3 clock 1: Enable SPI3 clock
10	SPI2EN	SPI2 clock enable Set and cleared by software. 0: Disable SPI2 clock 1: Enable SPI2 clock
9	SPI1EN	SPI1 clock enable Set and cleared by software. 0: Disable SPI1 clock 1: Enable SPI1 clock
8	Reserved	Reserved, the reset value must be maintained.
7	IOPFEN	GPIO port F clock enable Set and cleared by software. 0: Disable the clock of GPIO port F 1: Enable the clock of GPIO port F
6	Reserved	Reserved, the reset value must be maintained.
5	IOPDEN	GPIO port D clock enable Set and cleared by software. 0: Disable the clock of GPIO port D 1: Enable the clock of GPIO port D
4	IOPCEN	GPIO port C clock enable Set and cleared by software. 0: Disable the clock of GPIO port C 1: Enable the clock of GPIO port C
3	IOPBEN	GPIO port B clock enable Set and cleared by software. 0: Disable the clock of GPIO port B 1: Enable the clock of GPIO port B

Bit Field	Name	Description
2	IOPAEN	GPIO port A clock enable Set and cleared by software. 0: Disable the clock of GPIO port A 1: Enable the clock of GPIO port A
1	Reserved	Reserved, the reset value must be maintained.
0	AFIOEN	Alternate function IO clock enable Set and cleared by software. 0: Disable the alternate function IO clock 1: Enable the alternate function IO clock

4.3.9 APB1 Peripheral Clock Enable Register (RCC_APB1PCLKEN)

Address offset: 0x1c

Reset value: 0x0000 0000



Bit Field	Name	Description
31	OPAMPEN	OPAMP clock enable Set and cleared by software. 0: Disable OPAMP clock 1: Enable OPAMP clock
30:29	Reserved	Reserved, the reset value must be maintained
28	PWREN	Power interface clock enable Set and cleared by software. 0: Disable the power interface clock 1: Enable the power interface clock
27	Reserved	Reserved, the reset value must be maintained
26	CANEN	CAN clock enable Set and cleared by software. 0: Disable CAN clock 1: Enable CAN clock
25:23	Reserved	Reserved, the reset value must be maintained
22	I2C2EN	I2C2 clock enable Set and cleared by software. 0: Disable I2C2 clock 1: Enable I2C2 clock
21	I2C1EN	I2C1 clock enable

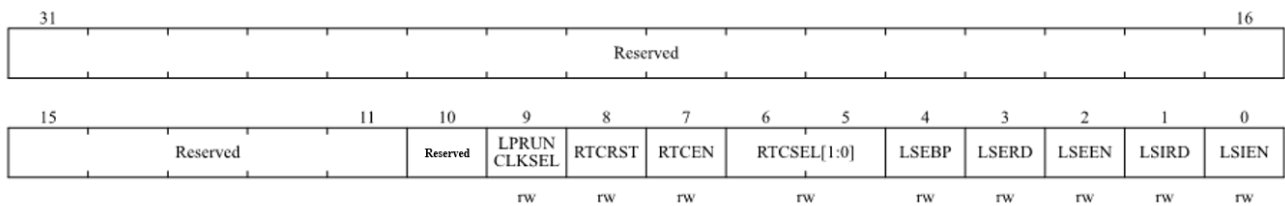
Bit Field	Name	Description
		Set and cleared by software. 0: Disable I2C1 clock 1: Enable I2C1 clock
20	UART5EN	UART5 clock enable Set and cleared by software. 0: Disable UART5 clock 1: Enable UART5 clock
19	LPUART2EN	LPUART2 clock enable Set and cleared by software. 0: Disable LPUART2 clock 1: Enable LPUART2 clock
18	LPUART1EN	LPUART1 clock enable Set and cleared by software. 0: Disable LPUART1 clock 1: Enable LPUART1 clock
17	USART2EN	USART2 clock enable Set and cleared by software. 0: Disable USART2 clock 1: Enable USART2 clock
16:12	Reserved	Reserved, the reset value must be maintained
11	WWDGEN	Window watchdog clock enable Set and cleared by software. 0: Disable WWDG clock 1: Enable WWDG clock
10	Reserved	Reserved, the reset value must be maintained
9	COMPFILTEN	Comparator filter clock enable 0: Disable the comparator filter clock 1: Enable the comparator filter clock
8	COMPEN	Comparator clock enable 0: Disable the comparator clock 1: Enable the comparator clock
7	Reserved	Reserved, the reset value must be maintained
6	BEEPER2EN	BEEPER2 clock enable Set and cleared by software. 0: Disable BEEPER2 clock 1: Enable BEEPER2 clock
5	BEEPER1EN	BEEPER1 clock enable Set and cleared by software. 0: Disable BEEPER1 clock 1: Enable BEEPER1 clock
4	TIM6EN	TIM6 timer clock enable Set and cleared by software. 0: Disable TIM6 timer clock

Bit Field	Name	Description
		1: Enable TIM6 timer clock
3	LPTIMEN	LPTIM timer clock enable Set and cleared by software. 0: Disable LPTIM timer clock 1: Enable LPTIM timer clock
2	TIM4EN	TIM4 timer clock enable Set and cleared by software. 0: Disable TIM4 timer clock 1: Enable TIM4 timer clock
1	TIM3EN	TIM3 timer clock enable Set and cleared by software. 0: Disable TIM3 timer clock 1: Enable TIM3 timer clock
0	Reserved	Reserved, the reset value must be maintained

4.3.10 Low Speed Clock Control Register (RCC_LSCTRL)

Address offset: 0x20

Reset value: 0x0000 0003



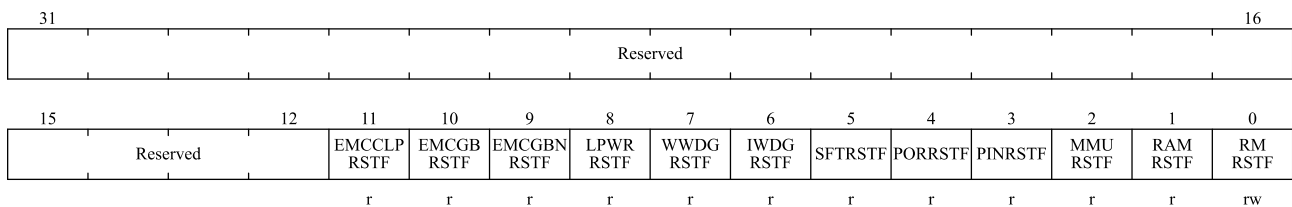
Bit Field	Name	Description
31:10	Reserved	Reserved, the reset value must be maintained
9	LPRUNCLKSEL	LPRUN clock selection Set and clear by software 0: LPRUN mode selects LSI clock 1: LPRUN mode selects LSE clock
8	RTCRST	RTC software reset 0: Clear reset 1: Reset RTC
7	RTCEN	RTC clock enable Set and clear by software 0: RTC clock disabled 1: RTC clock is enabled
6:5	RTCSEL[1:0]	RTC clock source selection. The software sets these bits to select the RTC clock source. 00: no clock 01: Select LSE oscillator as RTC clock

Bit Field	Name	Description
		10: Select LSI oscillator as RTC clock 11: Select the HSE oscillator divided by 128 as the RTC clock
4	LSEBP	The external low-speed clock oscillator is bypassed. Set by software to bypass LSE. At this time LSEEN needs to be set 0 0: LSE clock is not bypassed 1: LSE clock is bypassed This bit cannot be reset by system reset
3	LSERD	External low-speed oscillator ready flag Set by hardware once LSE is ready. After LSIEN is cleared, LSIRD is cleared after 6 external low-speed oscillator clock cycles. 0: LSE is not ready 1: LSE is ready
2	LSEEN	External low-speed oscillator enable Set and cleared by software 0: Disable LSE oscillator 1: Enable LSE oscillator
1	LSIRD	Internal low-speed ready oscillator flag Set by hardware once LSI is ready. After LSIEN is cleared, LSIRD is cleared after 3 internal low-speed oscillator clock cycles. 0: LSI is not ready 1: LSI is ready
0	LSIEN	Internal low-speed oscillator enable Set and cleared by software 0: Disable LSI oscillator 1: Enable LSI oscillator

4.3.11 Control/Status Register (RCC_CTRLSTS)

Address offset: 0x24

Reset value: 0x0000 0018



Bit Field	Name	Description
31:12	Reserved	Reserved, the reset value must be maintained
11	EMCCLPRSTF	EMCCLAMP reset flag Set by hardware when EMCCLAMP is reset. It is cleared by writing RMRSTF bit or por_rst_n reset. 0: No EMCCLAMP reset occurred 1: EMCCLAMP reset occurred

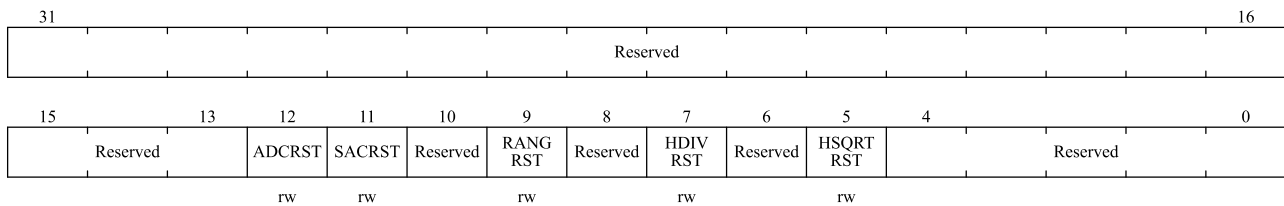
Bit Field	Name	Description
10	EMCGBRSTF	EMCGB reset flag Set by hardware when EMCGB is reset. It is cleared by writing RMRSTF bit or por_rst_n reset. 0: No EMCGB reset occurred 1: EMCGB reset occurred
9	EMCGBNRSTF	EMCGBN reset flag Set by hardware when EMCGBN is reset. It is cleared by writing RMRSTF bit or por_rst_n reset. 0: No EMCGBN reset occurred 1: EMCGBN reset occurred
8	LPWRRSTF	Low-power reset flag Set by hardware at Low-power reset. It is cleared by writing RMRSTF bit or por_rst_n reset. 0: No Low-power reset occurred 1: Low-power reset occurred
7	WWDGRSTF	Window watchdog reset flag Set by hardware when an Window watchdog reset occurs It is cleared by writing RMRSTF bit or por_rst_n reset. 0: No Window watchdog reset occurred 1: Window watchdog reset occurred
6	IWDGRSTF	Independent watchdog reset flag Set by hardware when an independent watchdog reset occurs It is cleared by writing RMRSTF bit or por_rst_n reset. 0: No independent watchdog reset occurred 1: Independent watchdog reset occurred
5	SFTRSTF	Software reset flag Set by hardware when a software reset occurs. It is cleared by writing RMRSTF bit or por_rst_n reset. 0: No software reset occurred 1: Software reset occurred
4	PORRSTF	POR/PDR reset flag Set by hardware when POR/PDR is reset. Cleared by writing to the RMRSTF bit 0: No POR/PDR reset occurred 1: POR/PDR reset occurred
3	PINRSTF	External pin reset flag Set by hardware when a reset from the NRST pin occurs. It is cleared by writing RMRSTF bit or por_rst_n reset. 0: No NRST pin reset occurred 1: NRST pin reset occurred
2	MMURSTF	MMU reset flag Set by hardware when MMU reset occurs. It is cleared by writing RMRSTF bit or por_rst_n reset.

Bit Field	Name	Description
		0: No MMU reset occurred 1: MMU reset occurred
1	RAMRSTF	RAM reset flag Set by hardware when RAM reset occurs. It is cleared by writing RMRSTF bit or por_rst_n reset. 0: No RAM reset occurred 1: RAM reset occurred
0	RMRSTF	REMOVE reset flag Set and clear by software 0: No effect 1: Clear these reset flags

4.3.12 AHB Peripheral Reset Register (RCC_AHBPRST)

Address offset: 0x28

Reset value: 0x0000 0000



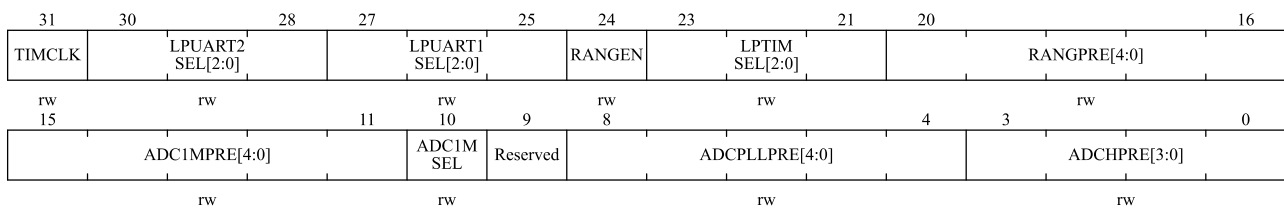
Bit Field	Name	Description
31:13	Reserved	Reserved, the reset value must be maintained.
12	ADCRST	ADC reset Set and cleared by software. 0: Clear reset 1: Reset ADC
11	SACRST	SAC reset Set and cleared by software. 0: Clear reset 1: Reset SAC
10	Reserved	Reserved, the reset value must be maintained.
9	RANGRST	RNG reset Set and cleared by software. 0: Clear reset 1: Reset RNG
8	Reserved	Reserved, the reset value must be maintained.
7	HDIVRST	HDIV reset Set and cleared by software. 0: Clear reset 1: Reset HDIV

Bit Field	Name	Description
6	Reserved	Reserved, the reset value must be maintained.
5	HSQRTRST	HSQRT reset Set and cleared by software. 0: Clear reset 1: Reset HSQRT
4:0	Reserved	Reserved, the reset value must be maintained.

4.3.13 Clock Configuration Register 2(RCC_CFG2)

Address offset: 0x2c

Reset value: 0x0000 3800



Bit Field	Name	Description
31	TIMCLK	TIM1/8 clock source selection Set and cleared by software. 0: PCLK2 is selected as TIM1/8 clock source if APB2 prescaler is 1. Otherwise, PCLK2 × 2 is selected. 1: SYSCLK input clock is selected as TIM1/8 clock source.
30:28	LPUART2SEL[2:0]	LPUART2 clock source selection Set and cleared by software. 000: APB1 clock is selected 001: System clock is selected 010: HSI clock is selected 011: HSE clock is selected 100: LSI clock is selected 101: LSE clock is selected Others: reserved.
27:25	LPUART1SEL[2:0]	LPUART1 clock source selection Set and cleared by software. 000: APB1 clock is selected 001: System clock is selected 010: HSI clock is selected 011: HSE clock is selected 100: LSI clock is selected 101: LSE clock is selected Others: reserved.
24	RANGEN	RNGEN: RNG Analog Interface Clock Enable Set and cleared by software.

Bit Field	Name	Description
		0: TRNG analog interface clock disabled 1: TRNG analog interface clock enable
23:21	LPTIMSEL[2:0]	LPTIM clock source selection Set and cleared by software. 000: select APB1 clock 001: Select HSI clock 010: Select HSE clock 011: Select LSI clock 100: Select LSE clock 101: Select COMP output Others: Not allowed, and no clock will be generated
20:16	RANGPRE[4:0]	RANGPRE: RANG Pre-Crossover Controls the prescaling factor by setting and clearing it in software. 00000: SYSCLK source clock divided by 1 00001: SYSCLK source clock divided by 2 00010: SYSCLK source clock divided by 3 ... 11110: SYSCLK source clock divided by 31 11111: SYSCLK source clock divided by 32
15:11	ADC1MPRE[4:0]	ADC 1M clock prescaler Set and cleared by software to configure the division factor of ADC 1M clock source. 00000: ADC 1M clock source not divided 00001: ADC 1M clock source divided by 2 00010: ADC 1M clock source divided by 3 ... 11110: ADC 1M clock source divided by 31 11111: ADC 1M clock source divided by 32
10	ADC1MSEL	ADC 1M clock source selection Set and cleared by software. 0: HSI oscillator clock selected as the input clock of ADC 1M 1: HSE oscillator clock selected as the input clock of ADC 1M <i>Note: When switching the ADC 1M clock source, you need to ensure that the HSI clock is turned on</i>
9	Reserved	Reserved, the reset value must be maintained.
8:4	ADCPLLPRE[4:0]	ADC PLL divider Set and cleared by software to configure the division factor from the PLL clock to the ADC. 0xxxx: ADC PLL clock is disabled 10000: PLL clock not divided 10001: PLL clock divided by 2 10010: PLL clock divided by 3 10011: PLL clock divided by 4 10100: PLL clock divided by 6

Bit Field	Name	Description
		10101: PLL clock divided by 8 10110: PLL clock divided by 10 10111: PLL clock divided by 12 11000: PLL clock divided by 16 11001: PLL clock divided by 32 11010: PLL clock divided by 64 11011: PLL clock divided by 128 Others: PLL clock divided by 256
3:0	ADCHPRE[3:0]	ADC HCLK prescaler Set and cleared by software to configure the division factor from the HCLK clock to the ADC. 0000: HCLK clock divided by 1 0001: HCLK clock divided by 2 0010: HCLK clock divided by 3 0011: HCLK clock divided by 4 0100: HCLK clock divided by 6 0101: HCLK clock divided by 8 0110: HCLK clock divided by 10 0111: HCLK clock divided by 12 1000: HCLK clock divided by 16 Others: HCLK clock divided by 32

4.3.14 EMC Control Register 3 (RCC_EMCCTRL)

Address offset: 0x30

Reset value: 0x0000 0000

31	Reserved							24	23	22	21	20	19	18	17	16
							GBRST3	GBRST2	GBRST1	GBRST0	GBNRST3	GBNRST2	GBNRST1	GBNRST0		
							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	
CLPRST3	CLPRST2	CLPRST1	CLPRST0	GBDET3	GBDET2	GBDET1	GBDET0	GBNDET3	GBNDET2	GBNDET1	GBNDET0	CLPDET3	CLPDET2	CLPDET1	CLPDET0	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit Field	Name	Description
31:24	Reserved	Reserved, the reset value must be maintained.
23	GBRST3	GB3 reset Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
22	GBRST2	GB2 reset Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
21	GBRST1	GB1 reset

Bit Field	Name	Description
		Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
20	GBRST0	GB0 reset Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
19	GBNRST3	GBN3 reset Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
18	GBNRST2	GBN2 reset Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
17	GBNRST1	GBN1 reset Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
16	GBNRST0	GBN0 reset Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
15	CLPRST3	EMC clamp3 reset Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
14	CLPRST2	EMC clamp2 reset Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
13	CLPRST1	EMC clamp1 reset Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
12	CLPRST0	EMC clamp0 reset Set and cleared by software. and reset by por_rst_n 0: Disable reset request 1: Enable reset request
11	GBDET3	GB3 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection 1: Enable detection

Bit Field	Name	Description
10	GBDET2	GB2 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection 1: Enable detection
9	GBDET1	GB1 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection 1: Enable detection
8	GBDET0	GB0 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection 1: Enable detection
7	GBNDET3	GBN3 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection 1: Enable detection
6	GBNDET2	GBN2 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection 1: Enable detection
5	GBNDET1	GBN1 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection 1: Enable detection
4	GBNDET0	GBN0 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection 1: Enable detection
3	CLPDET3	EMC Clamp3 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection 1: Enable detection
2	CLPDET2	EMC Clamp2 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection 1: Enable detection
1	CLPDET1	EMC Clamp1 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection 1: Enable detection
0	CLPDET0	EMC Clamp0 detection enable Set and cleared by software. and reset by por_rst_n 0: Disable detection

Bit Field	Name	Description
		1: Enable detection

5 GPIO and AFIO

5.1 Summary

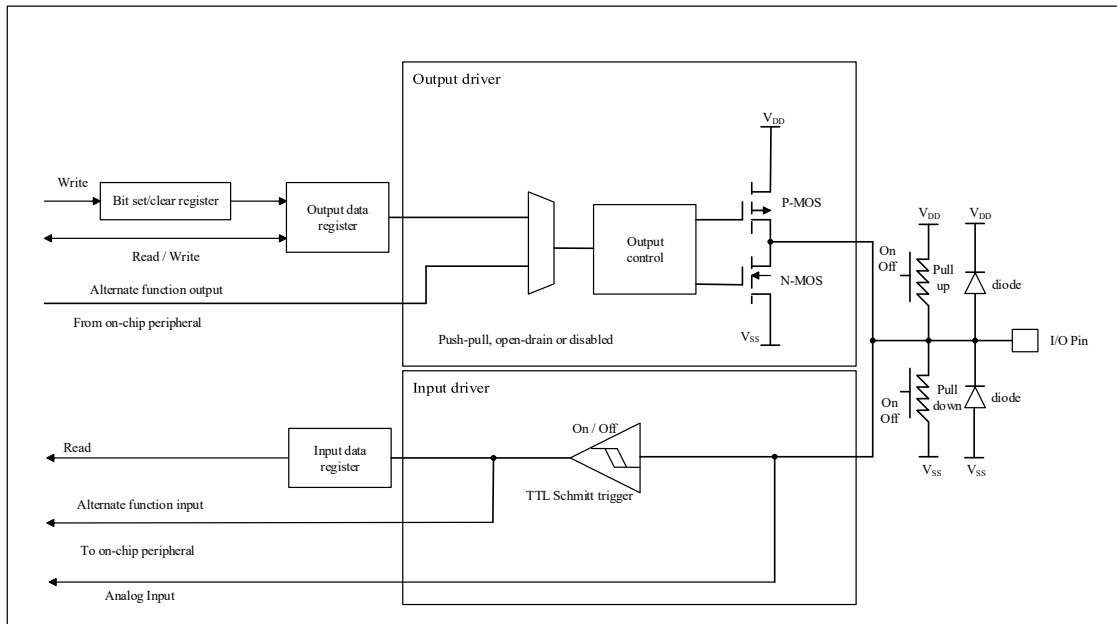
GPIO (general purpose input/output) and AFIO are available for flexible pin configuration. The chip supports up to 56 GPIOs, divided into 5 groups (GPIOA/GPIOB/GPIOC/ GPIOD/ GPIOF). GPIOA, GPIOB and GPIOC each have 16 pins, GPIOD has 1 pin and GPIOF has 7 pins. Each GPIO pin can be configured by software as output (push-pull or open drain), input (floating or pull-up or pull-down) or alternate peripheral function ports (output/input). Except for the analog input pins, other GPIO pins have the ability to pass through the large current.

GPIO ports have the following characteristics:

- Each GPIO port can be individually configured into multiple modes by software
 - Input floating
 - Input pull-up
 - Input pull-down
 - Analog function
 - Open-drain output and pull-up/pull-down capability
 - Push-pull output and pull-up/pull-down capability
 - Push-pull alternate function and pull-up/pull-down capability
 - Open-drain alternate function and pull-up/pull-down capability
- Individual bit set or bit clear function
- All I/O supports external interrupt function
- All I/O supports low power mode wake-up, rising or falling edge configurable
 - 16 EXTI lines can be used to wake up from SLEEP or STOP mode, and all I/Os can be reused as EXTI
 - PA0/PC13/PA2 three wake-up I/O can be used to wake up from PD mode, the maximum I/O filter time is 1µs
- Support software remapping I/O alternate function
- Support GPIO lock mechanism, lock states cleared by reset
- Each I/O port bit can be programmed arbitrarily, but I/O port registers must be accessed as 32-bit words (16-bit half-word or 8-bit byte access is not allowed).

The following figure shows the basic structure of I/O ports.

Figure 5-1 Basic Structure of I/O Ports



5.2 Function Description

5.2.1 I/O Mode Configuration

The I/O port mode can be configured through the registers GPIOx_PMODE, GPIOx_POTYPE and GPIOx_PUPD (x=A,B,C,F). The configuration in different operation modes is shown in the following table:

Table 5-1 I/O Port Configuration Table

PMODE[1:0]	POTYPE	PUPD[1:0]		I/O Configuration
01	0	0	0	General-purpose output push-pull
	0	0	1	General-purpose output push-pull + pull-up
	0	1	0	General-purpose output push-pull + pull-down
	0	1	1	Reserved
	1	0	0	General-purpose output open-drain
	1	0	1	General-purpose output open-drain + pull-up
	1	1	0	General-purpose output open-drain + pull-down
	1	1	1	Reserved
10	0	0	0	Alternate function + push-pull
	0	0	1	Alternate function + push-pull + pull-up
	0	1	0	Alternate function + push-pull + pull-down
	0	1	1	Reserved
	1	0	0	Alternate function open-drain
	1	0	1	Alternate function open-drain + pull-up
	1	1	0	Alternate function open-drain + pull-down
	1	1	1	Reserved
00	x	0	0	Input floating

PMODE[1:0]	POTYPE	PUPD[1:0]		I/O Configuration
	x	0	1	Input pull-up
	x	1	0	Input pull-down
	x	1	1	Reserved
11	x	0	0	Analog
	x	0	1	Reserved
	x	1	0	
	x	1	1	

In addition, the GPIOx_DS.DS_y bit can be used to configure the high/low drive strength, and the GPIOx_SR.SR_y bit can be used to configure the high/low slew rate.

The input and output characteristics of I/O under different configurations are shown in the following table:

Table 5-2 I/O List of Functional Features of The Pin

Feature	GPIO Input	GPIO Output	Analog Function	Alternate Function
Output buffer	Disabled	Enabled	Disabled	Configuration according to peripheral function
Schmitt trigger	Enabled	Enabled	Disabled, Output is forced to 0	Enable
PULL UP/DOWN/FLOATING	Configured	Configured	Disabled	Configuration according to peripheral function
OPEN DRAIN	Disabled	Can be configured, GPIO output 0 when output data is "0", high resistance of GPIO when "1"	Disabled	Can be configured, GPIO output 0 when output data is "0", high resistance of GPIO when "1"
PUSH PULL MODE	Disabled	Can be configured, GPIO output 0 when output data is "0", GPIO output 1 when output data is "1"	Disabled	Can be configured, GPIO output 0 when output data is "0", GPIO output 1 when output data is "1"
Input data register (I/O status)	Readable	Readable	Reads out 0	Readable
Output data register (Output value)	Invalid	Readable and written	Invalid	Readable

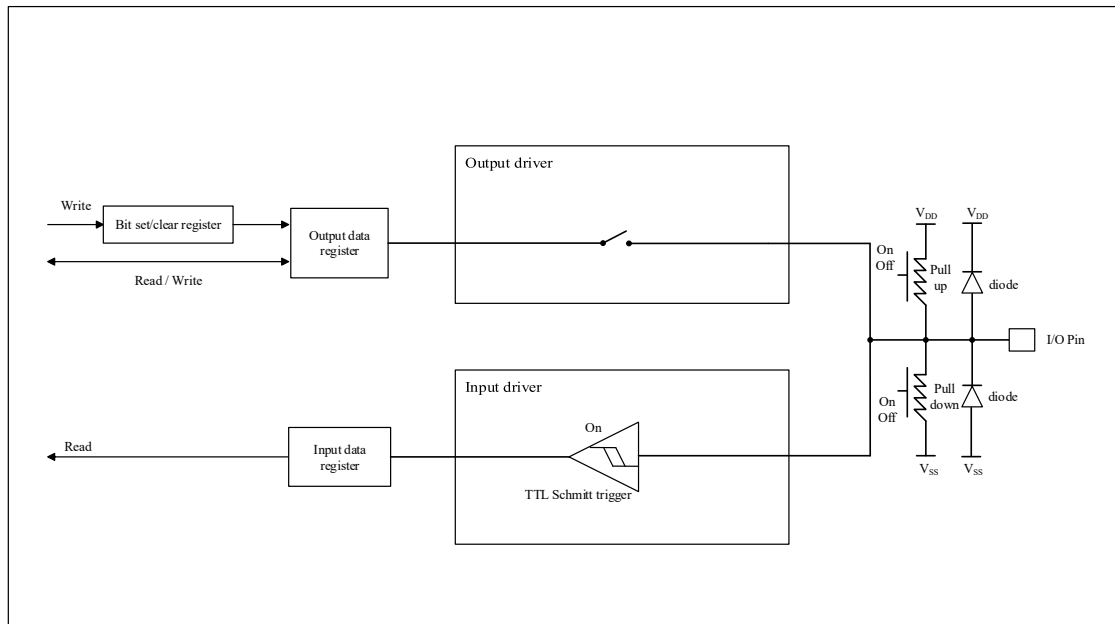
Input mode

When the I / O port is configured as input mode:

- Output buffer is disabled
- The schmitt trigger input is activated.
- Whether the pull-up and pull-down resistors are connected depends on the configuration of the GPIOx_PUPD register.
- The data appearing on the I/O pins is sampled into the input data register on every APB2 clock

- Read access to the input data register provides the I/O status

Figure 5-2 Input Floating/Pull-up/Pull-down Configuration Mode.

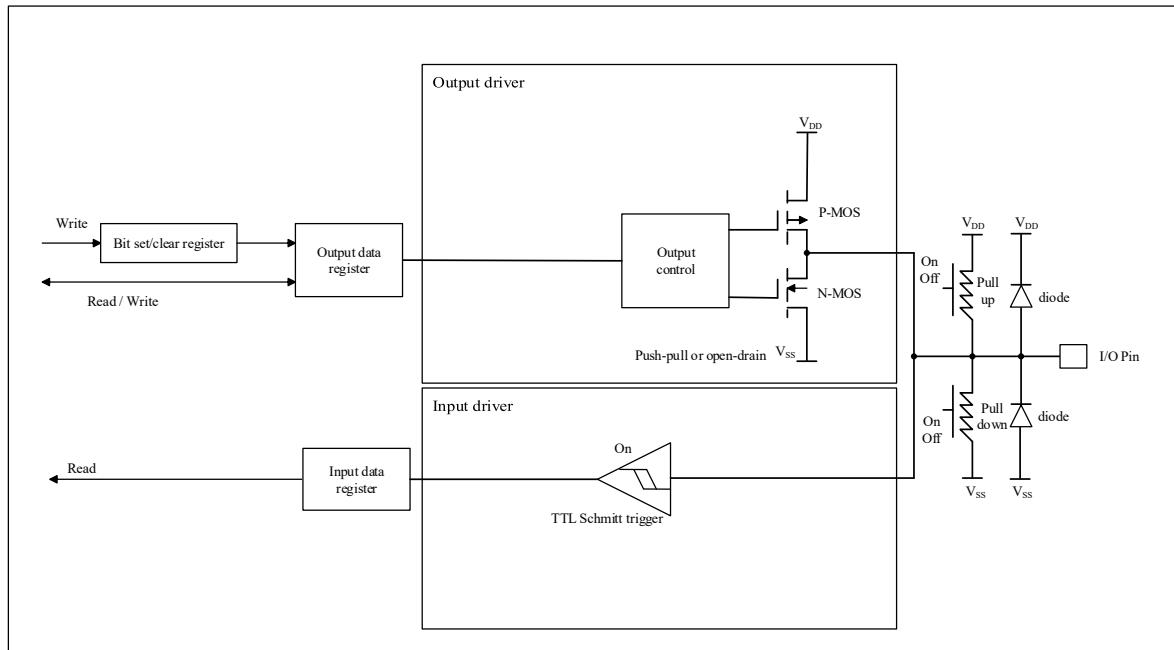


Output mode

When the I/O port is configured as output mode:

- The schmitt trigger input is activated.
- Whether the pull-up and pull-down resistors are connected depends on the configuration of the GPIOx_PUPD register
- The output buffer is activated
 - Open drain mode: '0' on the output register activates the N-MOS, the pin outputs a low level. while '1' on the output register puts the port in a high resistance state (PMOS is never activated).
 - Push-pull mode: '0' on the output register activates the N-MOS, the pin outputs a low level. While '1' on the output register activates the P-MOS, the pin outputs a high level.
- The data appearing on the I/O pins is sampled into the input data register every APB2 clock.
- Read access to the input data register to get I/O status.
- Read access to the output data register to get the last written value.

Figure 5-3 Output Mode

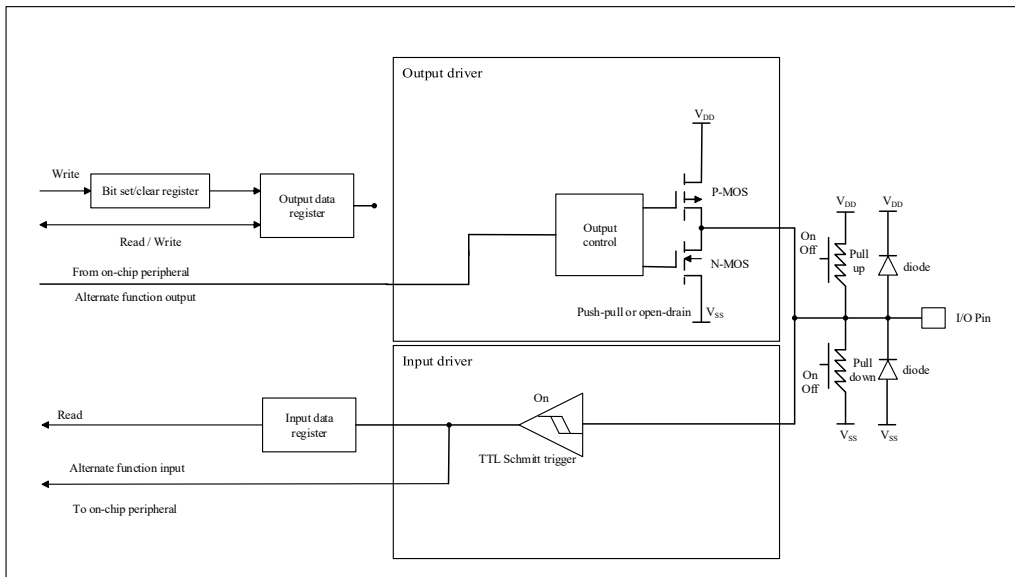


Alternate function mode

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

- The schmitt trigger input is activated.
- Whether the pull-up and pull-down resistors are connected, depending on the configuration of the GPIOx_PUPD register.
- In open-drain or push-pull configuration, the output buffer is controlled by the peripheral.
- Signal from built-in peripherals drives output buffers.
- At each APB2 clock cycle, the data appearing on the I/O pin is sampled into the input data register.
- Read access to the input data register to get I/O status.
- Read access to the output data register to get the last written value.

Figure 5-4 Alternate Function Mode

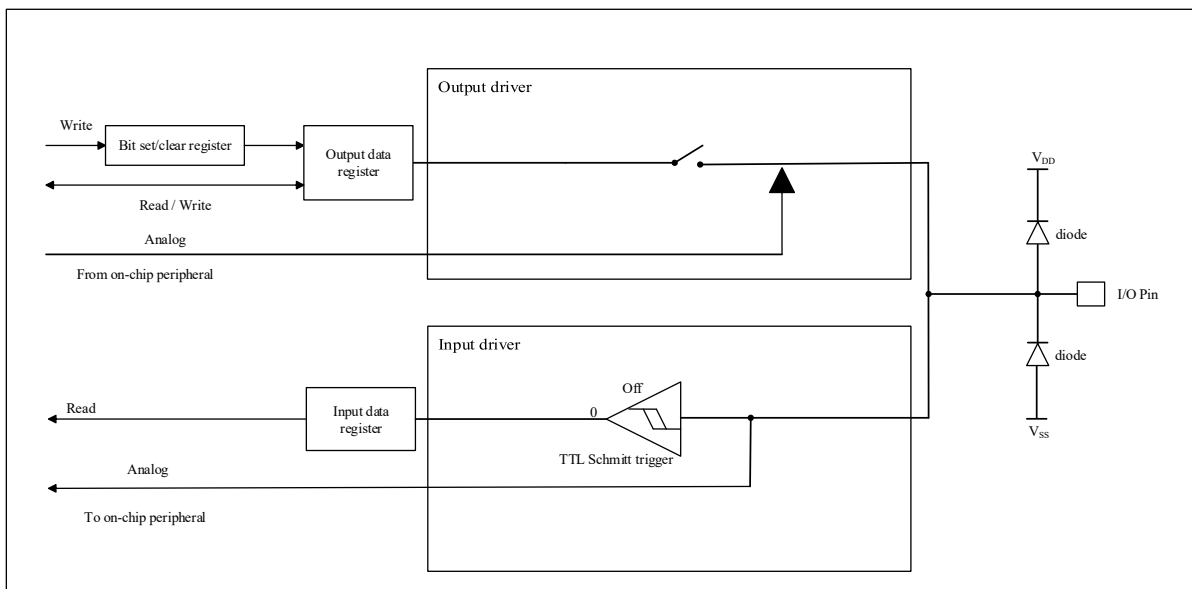


Analog function mode

When the I/O Port is configured as analog function mode:

- The pull-up and pull-down resistors are disabled.
- Read access to the input data register gets the value “0”.
- The output buffer is disabled.
- Schmitt trigger input is disabled and output value is forced to '0' (achieves zero consumption on each analog I/O pin).

Figure 5-5 Analog Function Mode with High Impedance



5.2.2 Status After Reset

During and just after reset, the alternate function is not turned on, and the I/O port is configured as analog function mode (GPIOx_PMODE.PMODEx [1:0] = 11b). However, there are the following exceptions to the signal.

- NRST default non-GPIO functions:
 - NRST pull-up input
- After reset, the default configuration of the pins related to the debugging system is the SWD interface I/O configuration:
 - PA14: SWCLK in input pull-down mode
 - PA13: SWDIO in input pull-up mode
- PA0 and PF0:
 - PA0 and PF0 default input floating mode.
 - PF0 are multiplexed to OSC_IN.
- PF2/BOOT0:
 - During the chip startup process, PF2 as BOOT0 functions, and the default input pull-down mode. After the chip is initialized, it is used as a general-purpose GPIO and is configured as an analog function mode.

5.2.3 Individual Bit Setting and Bit Clearing

By writing '1' to the bit in the set register (GPIOx_PBSC) and reset register (GPIOx_PBC), the individual bit operation of the data register (GPIOx_POD) can be realized, and one or more bits can be set/reset. The bit written with '1' is set or cleared accordingly, and the bit not written with '1' will not be changed. The software does not need to disable interrupts, and is completed in a single APB2 write operation.

5.2.4 External Interrupt/Wakeup Line

All ports have external interrupt capability, which can be configured in the EXTI module:

- The port must be configured in input mode.
- All ports can be configured for SLEEP/STOP mode wake-up, supporting rising or falling edge configurable.
- PA0/PC13/PA2 can be used for PD mode wake-up, with independent wake-up enable, supporting configurable rising edge/falling triggers, need to be configured before entering PD mode.
- General purpose I/O ports are connected to 16 external interrupt/event lines as shown in Figure 6-2, configured by registers AFIO_EXTI_CFGx.

5.2.5 Alternate Function

When I/O ports are configured in alternate function mode. The bit configuration register of port(GPIOx_AFL/ GPIOx_AFH, GPIOx_PMODE, GPIOx_POTYPE and GPIOx_PUPD) must be configured before use. Alternate input or output is determined by the peripheral.

Software remapping I/O alternate function

To expand the flexibility of alternate peripheral functions under different device packages, some peripheral alternate functions can be remapped to other pins. Each I/O has up to 16 alternate functions (AF0~AF15). After reset, except for PA13 and PA14, AFSELY is selected as AF15 by default. The I/O alternate function can be remapped by software configuring the corresponding registers (GPIOx_AFL/ GPIOx_AFH).

At this time, the alternate functions are no longer mapped to their original pins (For the I/O alternate function of the

peripheral, if it is remapped to a different pin, then the input is remapping choose one of multiple, and the output will be connected to the remapped position, and the original position will be disconnected).

SWD alternate function I/O remapping

Table 5-3 I/O List of Functional Features of The Pin

Alternate Function	I/O	Remapping
SWDIO	PA13	AF0
SWCLK	PA14	AF0

TIMx alternate function I/O remapping

5.2.5.1.1 TIM1 alternate function I/O remapping

Table 5-4 TIM1 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
TIM1_ETR	PA12	AF2
TIM1_BKIN	PA2	AF3
	PA6	AF1
	PB12	AF1
TIM1_CH1	PA4	AF3
	PA8	AF2
TIM1_CH2	PA3	AF3
	PA9	AF2
TIM1_CH3	PA5	AF3
	PA10	AF2
TIM1_CH4	PA11	AF2
TIM1_CH1N	PA7	AF5
	PB13	AF1
TIM1_CH2N	PA5	AF4
	PB0	AF1
	PB14	AF1
TIM1_CH3N	PB1	AF1
	PB15	AF1

5.2.5.1.2 TIM8 alternate function I/O remapping

Table 5-5 TIM8 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
TIM8_ETR	PA0	AF10
	PA4	AF13
	PA5	AF1
TIM8_BKIN	PA9	AF1
	PA10	AF1
	PB4	AF1

Alternate Function	I/O	Remapping
	PB5	AF1
TIM8_CH1	PA0	AF3
	PA5	AF2
	PA6	AF4
	PA8	AF9
	PB8	AF5
	PB12	AF5
TIM8_CH2	PA1	AF2
	PA7	AF1
	PB0	AF9
	PB3	AF1
	PB9	AF5
	PB13	AF5
TIM8_CH3	PA2	AF2
	PA10	AF9
	PB6	AF2
	PB11	AF5
	PB14	AF5
TIM8_CH4	PA3	AF2
	PB1	AF9
	PB7	AF7
	PB15	AF7
TIM8_CH1N	PA9	AF5
	PB6	AF5
TIM8_CH2N	PA8	AF1
	PB7	AF5
TIM8_CH3N	PB5	AF5
	PB15	AF5

5.2.5.1.3 TIM3 alternate function I/O remapping

Table 5-6 Tim3 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
TIM3_ETR	PA1	AF10
	PA7	AF12
	PD2	AF0
TIM3_CH1	PA2	AF5
	PA6	AF2
	PA10	AF3
	PA10	AF5
	PB4	AF2
	PC6	AF1
TIM3_CH2	PA3	AF5

Alternate Function	I/O	Remapping
	PA7	AF2
	PA9	AF3
	PB5	AF2
	PC7	AF1
TIM3_CH3	PB0	AF2
	PC8	AF1
TIM3_CH4	PB1	AF2
	PC9	AF1

5.2.5.1.4 TIM4 alternate function I/O remapping

Table 5-7 TIM4 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
TIM4_ETR	PB10	AF1
TIM4_CH1	PA4	AF2
TIM4_CH2	PA7	AF4
	PB15	AF6
TIM4_CH3	PB1	AF5
TIM4_CH4	PB2	AF5

LPTIM alternate function I/O remapping
Table 5-8 LPTIM Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
LPTIM_ETR	PA4	AF12
	PA6	AF9
	PA13	AF3
	PB6	AF8
	PC3	AF0
LPTIM_IN1	PA0	AF7
	PA4	AF11
	PA8	AF10
	PB1	AF8
	PB5	AF8
	PC0	AF0
LPTIM_IN2	PA1	AF9
	PA5	AF10
	PB7	AF8
	PC2	AF0
LPTIM_OUT	PA7	AF9
	PA9	AF9
	PA14	AF3
	PB2	AF8
	PB4	AF8
	PC1	AF0
	PB14	AF1

CAN alternate function I/O remapping
Table 5-9 CAN Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
CAN_RX	PA5	AF9
	PB11	AF9
	PC5	AF2
	PF1	AF4
CAN_TX	PA4	AF9
	PB12	AF9
	PC4	AF2
	PF0	AF4

USARTx alternate function I/O remapping
5.2.5.1.5 USART1 alternate function I/O remapping
Table 5-10 USART1 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
USART1_CTS	PA0	AF1
	PA11	AF4
	PB12	AF1
USART1_RTS	PA1	AF1
	PA12	AF4
USART1_TX	PA2	AF1
	PA9	AF4
	PA13	AF6
	PA14	AF4
	PB6	AF4
	PB9	AF4
USART1_RX	PA3	AF1
	PA10	AF4
	PA13	AF4
	PA15	AF4
	PB7	AF4
USART1_CK	PA4	AF1
	PA8	AF4
	PF1	AF2

5.2.5.1.6 USART2 alternate function I/O remapping
Table 5-11 USART2 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
USART2_CTS	PA0	AF4
	PA7	AF10
USART2_RTS	PA1	AF4
	PB0	AF4
USART2_TX	PA2	AF4
	PA9	AF10
	PA14	AF1
	PB6	AF1
USART2_RX	PA0	AF5
	PA3	AF4
	PA10	AF10
	PA13	AF1
	PA15	AF1

Alternate Function	I/O	Remapping
USART2_CK	PB7	AF3
	PA4	AF4
	PA8	AF6
	PB1	AF3
	PF1	AF5

5.2.5.1.7 UART5 alternate function I/O remapping

Table 5-12 Uart5 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
UART5_TX	PB0	AF5
	PB3	AF4
	PC12	AF6
UART5_RX	PB1	AF6
	PB4	AF4
	PD2	AF2

5.2.5.1.8 UART6 alternate function I/O remapping

Table 5-13 UART6 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
UART6_TX	PA4	AF6
	PA11	AF6
	PC0	AF4
UART6_RX	PA5	AF6
	PA12	AF6
	PC1	AF4

LPUARTx alternate function I/O remapping

5.2.5.1.9 LPUART1 alternate function I/O remapping

Table 5-14 LPUART1 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
LPUART1_CTS	PA6	AF5
	PB13	AF4
LPUART1_RTS	PB1	AF4
	PB14	AF4
	PD2	AF1
LPUART1_TX	PA1	AF5
	PA2	AF6
	PA4	AF10
	PA6	AF6
	PA14	AF6
	PB3	AF5

Alternate Function	I/O	Remapping
	PB6	AF3
	PB10	AF4
	PC4	AF4
	PC10	AF4
LPUART1_RX	PA0	AF11
	PA3	AF6
	PA7	AF6
	PA13	AF2
	PB4	AF5
	PB7	AF9
	PB11	AF4
	PC5	AF4
PC11	AF4	

5.2.5.1.10 LPUART2 alternate function I/O remapping

Table 5-15 LPUART2 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
LPUART2_CTS	PB7	AF2
LPUART2_RTS	PA15	AF6
LPUART2_TX	PA0	AF6
	PB5	AF3
	PC10	AF5
LPUART2_RX	PA1	AF6
	PB6	AF3
	PB7	AF1
	PC11	AF5

I2Cx alternate function I/O remapping

5.2.5.1.11 I2C1 alternate function I/O remapping

Table 5-16 I2c1 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
I2C1_SCL	PA4	AF7
	PA9	AF6
	PB6	AF6
	PB8	AF6
	PB10	AF6
	PF1	AF1
	PF6	AF1
I2C1_SDA	PA10	AF6
	PA13	AF7
	PB7	AF6

Alternate Function	I/O	Remapping
	PB9	AF6
	PB11	AF6
	PF0	AF1
	PF7	AF1
I2C1_SMBA	PA1	AF7
	PA14	AF7
	PB2	AF6
	PB5	AF6

5.2.5.1.12 I2C2 alternate function I/O remapping

Table 5-17 I2C2 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
I2C2_SCL	PA6	AF7
	PA9	AF7
	PA11	AF7
	PB10	AF7
	PB13	AF7
	PF6	AF0
I2C2_SDA	PA7	AF7
	PA10	AF7
	PA12	AF7
	PB11	AF7
	PB14	AF7
	PF7	AF0
I2C2_SMBA	PB2	AF7

SPI/I2S alternate function I/O remapping
5.2.5.1.13 SPI1/I2S1 alternate function I/O remapping
Table 5-18 SPI1/I2S1 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
SPI1_I2S1_NSS_WS	PA1	AF0
	PA4	AF0
	PA15	AF0
	PB12	AF0
SPI1_I2S1_SCLK_CK	PA0	AF0
	PA5	AF0
	PA13	AF5
	PB3	AF0
	PB13	AF0
SPI1_I2S1_MISO_MCK	PA3	AF0
	PA4	AF5
	PA6	AF0
	PA14	AF5
	PB0	AF6
	PB4	AF0
	PB14	AF0
SPI1_I2S1_MOSI_SD	PA2	AF0
	PA5	AF5
	PA7	AF0
	PB1	AF0
	PB5	AF0
	PB10	AF0
	PB15	AF0

5.2.5.1.14 SPI2 alternate function I/O remapping
Table 5-19 SPI2 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
SPI2_NSS	PB9	AF0
	PB12	AF8
SPI2_SCLK	PB10	AF2
	PB13	AF8
SPI2_MISO	PB14	AF8
	PC2	AF1
SPI2_MOSI	PB15	AF8
	PC3	AF1

5.2.5.1.15 SPI3 alternate function I/O remapping
Table 5-20 SPI3 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
SPI3_NSS	PA8	AF0
	PF7	AF4
SPI3_SCLK	PA9	AF0
	PB0	AF0
	PF6	AF4
SPI3_MISO	PA10	AF0
	PA12	AF0
	PC10	AF6
SPI3_MOSI	PA11	AF0
	PA15	AF5
	PC9	AF6

COMPx alternate function I/O remapping
5.2.5.1.16 COMP1 alternate function I/O remapping
Table 5-21 Comp1 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
COMP1_OUT	PA0	AF8
	PA6	AF8
	PA9	AF8
	PA11	AF8
	PA13	AF8
	PA15	AF8

5.2.5.1.17 COMP2 alternate function I/O remapping
Table 5-22 COMP2 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
COMP2_OUT	PA2	AF8
	PA4	AF8
	PA7	AF8
	PA10	AF8
	PA12	AF8
	PA14	AF8

5.2.5.1.18 COMP3 alternate function I/O remapping
Table 5-23 Comp3 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
COMP3_OUT	PB1	AF7
	PC8	AF5
	PF0	AF5

BEEPERx alternate function I/O remapping

5.2.5.1.19 BEEPER1 alternate function I/O remapping

Table 5-24 Beeper1 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
BEEPER1_OUT	PA6	AF11
	PC1	AF7
BEEPER1_N_OUT	PA7	AF11
	PC2	AF7

5.2.5.1.20 BEEPER2 alternate function I/O remapping

Table 5-25 BEEPER2 Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
BEEPER2_OUT	PB6	AF11
BEEPER2_N_OUT	PB7	AF11

EVENTOUT alternate function I/O remapping

Table 5-26 Eventout Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
EVENTOUT	PA1	AF3
	PA6	AF10
	PA7	AF13
	PA8	AF8
	PA6~PA8	AF3
	PA11~PA12	AF3
	PA15	AF3
	PB0	AF3
	PB0	AF7
	PB3~PB4	AF3
	PB9	AF3
	PB11~PB12	AF3
	PC0~PC4	AF3
	PF4~PF5	AF3

RTC alternate function I/O remapping

PC13 can be used as RTC TAMPER1 intrusion detection pin, RTC Timestamp, RTC output (RTC alarm, Wakeup event or calibration output (256Hz or 1Hz)).

PA0 can be used as RTC TAMPER2 intrusion detection pin.

PA10 or PB15 can be used as RTC REFCLKIN reference clock input pin.

Table 5-27 RTC Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
RTC_TS	PC13	AF1
RTC_TAMP1	PC13	AF2
RTC_TAMP2	PA0	AF9
RTC_REFIN	PA10	AF11
	PB15	AF9
RTC_OUT	PC13	AF3

PVD RCC alternate function I/O remapping
Table 5-28 PVD Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
PVD_IN	PB7	AF10

RCC alternate function I/O remapping
Table 5-29 Rcc Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
MCO	PA8	AF5
	PA9	AF11

OSC_IN/OSC_OUT alternate function I/O remapping
Table 5-30 OSC_IN/OSC_OUT Alternate Function I/O Remapping

Alternate Function	I/O	Remapping
OSC_IN	PA0	AF2
	PF0	AF0
OSC_OUT	PF1	AF0

Use OSC32_IN/OSC32_OUT pins as GPIO ports PC14/PC15

PC14~PC15 two pins can be used as GPIO mode or alternate function mode:

- PC14 and PC15 can be used for GPIO or LSE (OSC32_IN, OSC32_OUT) pins.
- Turn on the LSE function by setting the RCC_LSCTRL.LSEEN and RCC_LSCTRL.LSEBP bits.

Table 5-31 Osc32 Alternate Function Remapping

PC14 and PC15	Condition	PAD Mode Configure
GPIO mode	It can only be used in GPIO mode when the LSE is turned off and the 1.5V power supply is turned off without entering the low power mode (PD).	The mode of the GPIO is determined by the application
LSE crystal mode	RCC_LSCTRL.LSEEN bit is enabled, alternate mode is on	Analog function mode
LSE external clock mode	RCC_LSCTRL.LSEEN bit is disabled, RCC_LSCTRL.LSEBP is enabled, alternate mode is enabled	Input pull-down

The default is analog function mode; PC14 and PC15 decide which mode and I/O function they are in according to RCC_LSCTRL.LSEEN, RCC_LSCTRL.LSEBP, chip mode signal, GPIOx_PMODE, GPIOx_POTYPE and

GPIOx_PUPD.

Remap OSC_IN/OSC_OUT alternate function

PF0 and PF1 can be used for GPIO or HSE (OSC_IN, OSC_OUT) pins and other analog or digital signal pins.

- Configure the mode of PF0/PF1 by setting registers such as GPIOF_PMODE, GPIOF_POTYPE and GPIOF_PUPD.
- Remap the I/O alternate function of PF0/PF1 by setting the GPIOF_AFL.AFSEL0 [3:0] bits and GPIOF_AFL.AFSEL1[3:0] bits.
- Enable the HSE function by setting the RCC_CTRL.HSEEN bit.

Table 5-32 OSC Alternate Function Remapping

PF0	Condition	PAD Mode Configure
GPIO mode	Configure general purpose output or input mode and GPIOF_PID and GPIOF_POD registers.	The mode of the GPIO is determined by the application
HSE crystal mode or other analog alternate mode	Reset to analog input HSE, OPA alternate mode	Analog function mode
HSE external clock or other digital alternate mode	HSE clock, I2C1, USART1/2 alternate mode	Mode is determined by the application

ADC external trigger alternate function remapping

The external trigger source of ADC injection conversion and regular conversion supports remapping, see the AFIO_CFG register.

Table 5-33 ADC External Trigger Injection Conversion Alternate Function Remapping

Alternate Function	ADC_ETRI = 0	ADC_ETRI = 1
ADC external trigger injection conversion	ADC external trigger injection conversion is connected to EXTI (0-15), EXTIx can be selected by AFIO_CFG.EXTI_ETRI[3:0] configuration	ADC external trigger injection conversion is connected to TIM8_CH4

Table 5-34 ADC External Trigger Regular Conversion Alternate Function Remapping

Alternate Function	ADC_ETRR = 0	ADC_ETRR = 1
ADC external trigger injection conversion	ADC external trigger regular conversion is connected to EXTI (0-15), EXTIx can be selected by AFIO_CFG.EXTI_ETRI[3:0] configuration	ADC external trigger regular conversion is connected to TIM8_TRGO

TIM4_CH2 alternate function I/O remapping

Table 5-35 TIM4_CH2 Alternate Function Remapping

Alternate Function	TIM4CH2_RMP = 0	TIM4CH2_RMP = 1
TIM4_CH2	TIM4_CH2 is connected to PA7.	The LSI internal clock is connected to the input of TIM4_CH2 for calibration purposes

I2C IO Signal Dual Voltage Levels Configuration

The I/O voltage level signals of I2C support two configuration options: V_{DD} voltage level or low level (signal level at 1.8V).

Table 5-36 I/O Signal Dual Voltage Levels Configuration

I2C2_LV_EN	RW	1'b0	I2C2 I/O Signal Low-Level Enable: 0: V _{DD} level enabled 1: Low level enabled
I2C1_LV_EN	RW	1'b0	I2C1 PAD Signal Low-Level Enable: 0: V _{DD} level enabled 1: Low level enabled

5.2.6 I/O Configuration Of Peripherals

Table 5-37 ADC

ADC	PAD Configuration
ADC	Analog function mode

Table 5-38 PVD

PVD	PAD Configuration
PVD_IN	Analog function mode

Table 5-39 TIM1/TIM8

TIM Pin	Configuration	PAD Configuration Mode
TIM1/8_CHx	Channel x input capture	Alternate function push-pull
	Output compare channel x	Alternate function push-pull
TIM1/8_CHxN	Complementary output channel x	Alternate function push-pull
TIM1/8_BKIN	Brake input	Alternate function push-pull
TIM1/8_ETR	External trigger clock input	Alternate function push-pull

Table 5-40 Tim3 And Lptim

TIM3/4/LPTIM Pin	Configuration	PAD Configuration Mode
TIM3/4/LPTIM_CHx	Input capture channel x	Alternate function push-pull
	Output compare channel x	Alternate function push-pull
TIM3/4/LPTIM_ETR	External trigger clock input	Alternate function push-pull

Table 5-41 CAN

CAN Pin	PAD Configuration Mode
CAN_TX	Alternate function push-pull
CAN_RX	Alternate function push-pull(no pull-down or pull-up)

Table 5-42 USART

USART Pin	Configuration	PAD Configuration
USARTx_TX	full duplex mode	Alternate function push-pull
	Half duplex mode	Alternate function open-drain
USARTx_RX	Full duplex mode	Alternate function push-pull(no pull-down or pull-up/pull-up)
	Half duplex mode	Unused, can be used as general I/O.
USARTx_CK	Synchronous mode	Alternate function push-pull

USART Pin	Configuration	PAD Configuration
USARTx_RTS	Hardware flow control	Alternate function push-pull
USARTx_CTS	Hardware flow control	Alternate function push-pull(no pull-down or pull-up/pull-up)

Table 5-43 LPUART

LPUART Pin	Configuration	PAD Configuration
LPUART_TX	Full duplex mode	Alternate function push-pull
LPUART_RX	Full duplex mode	Alternate function push-pull(no pull-down or pull-up/pull-up)
LPUART_RTS	Hardware flow control	Alternate function push-pull
LPUART_CTS	Hardware flow control	Alternate function push-pull(no pull-down or pull-up /pull-up)

Table 5-44 I2C

I2C Pin	Configuration	PAD Configuration
I2Cx_SCL	I2C clock	Alternate function open-drain
I2Cx_SDA	I2C data	Alternate function open-drain
I2Cx_SMBA	SMBA data	Alternate function open-drain

Table 5-45 SPI

SPI Pin	Configuration	PAD Configuration
SPIx_SCLK	Master mode	Alternate function push-pull
	Slave mode	Alternate function push-pull
SPIx_MOSI	Full duplex mode / Master mode	Alternate function push-pull
	Full duplex mode / Slave mode	Alternate function push-pull(no pull-down or pull-up/pull-up)
	Simple bidirectional data line / Master mode	Alternate function push-pull
	Simple bidirectional data line / Slave mode	Unused, can be used as general I/O.
SPIx_MISO	Full duplex mode / Master mode	Alternate function push-pull(no pull-down or pull-up/pull-up)
	Full duplex mode / Slave mode	Alternate function push-pull
	Simple bidirectional data line / Master mode	Unused, can be used as general I/O.
	Simple bidirectional data line / Slave mode	Alternate function push-pull
SPIx_NSS	Hardware Master/ Slave mode	Alternate function push-pull(no pull-down or pull-up/pull-up/pull-down)
	Hardware Master /NSS output enable	Alternate function push-pull (When acting as the master, NSS can choose idle high impedance or idle as 1)
	Software mode	Unused, can be used as general I/O.

Table 5-46 COMP

COMP Pin	PAD Configuration
COMP_OUT	Alternate function push-pull

COMP_IN	Analog function mode
---------	----------------------

Table 5-47 BEEPER

BEEPER Pin	PAD Configuration
BEEPER_OUT	Alternate function push-pull
BEEPER_N_OUT	Alternate function push-pull

Table 5-48 Other

Pin	Alternate Function	PAD Configuration
EVENTOUT	Event output	Alternate function push-pull
RTC_TAMP1/RTC_TAMP2	Intrusion event input	Alternate function push-pull Hardware forced pull-up
RTC_TS	RTC timestamp	Alternate function push-pull
RTC_REFIN	RTC reference clock input	Alternate function push-pull
RTC_OUT	RTC output	Alternate function push-pull
MCO	Clock output	Alternate function push-pull
EXTI input line	External interrupt input	Input floating or input with pull-up or input with pull-down

5.2.7 GPIO Locking Mechanism

The locking mechanism is used to freeze the I/O configuration to prevent accidental changes. When a lock procedure is executed on a port bit, the configuration of the port cannot be changed until the next reset, refer to the port configuration lock register GPIOx_PLOCK.

- Only after the GPIOx_PLOCK.PLOCKK is operated in the correct sequence w1->w0->w1->r0 (where r0 must be), GPIOx_PLOCK[16] will become 1; after that, it will only become 0 after a system reset.
- GPIOx_PLOCK.PLOCK[15:0] can only be modified when GPIOx_PLOCK.PLOCKK = 0 (that is, when unlocked).
- GPIOx_PLOCK.PLOCK is only written simultaneously with non-zero GPIOx_PLOCK.PLOCK[15:0], then the sequence w1->w0->w1->r0 is valid; During the sequence writing process, GPIOx_PLOCK.PLOCK[15:0] remains unchanged.
- As long as GPIOx_PLOCK.PLOCKK = 0, the bits of GPIOx_PMODE/GPIOx_POTYPE/GPIOx_PUPD/GPIOx_AFL/GPIOx_AFH can be modified, they are not affected by GPIOx_PLOCK.PLOCK[15:0] configuration.
- GPIOx_PLOCK.PLOCKK = 1, GPIOx_PMODE/GPIOx_POTYPE/GPIOx_PUPD/GPIOx_AFL/GPIOx_AFH are controlled by GPIOx_PLOCK.PLOCK[15:0]. Corresponding to GPIOx_PLOCK.PLOCKy (y = 0...15) = 1, it is a lock configuration and cannot be modified; PLOCKy = 0, it can be modified.
- If the lock sequence operation is wrong, then it must be redone (w1-> w0-> w1-> r0) to initiate the lock operation again.

5.3 GPIO Registers

All register operations must be performed in 32-bit words.

5.3.1 GPIO Register Overview

GPIOA base address: 0x40010800

GPIOB base address: 0x40010C00

GPIOC base address: 0x40011000

GIOD base address: 0x40014000

GPIOF base address: 0x40011C00

Table 5-49 GPIO Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																					
000h	GPIOx_PMODE	x=A,B,C,D,F		PMODE15 [1:0]		PMODE14 [1:0]		PMODE13 [1:0]		PMODE12 [1:0]		PMODE11 [1:0]		PMODE10 [1:0]		PMODE9 [1:0]		PMODE8 [1:0]		PMODE7 [1:0]		PMODE6 [1:0]		PMODE5 [1:0]		PMODE4 [1:0]		PMODE3 [1:0]		PMODE2 [1:0]		PMODE1 [1:0]		PMODE0 [1:0]																				
	Reset Value	x=A	1	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0																				
		x=B,C	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1																			
		x=D	Reserved																																																			
x=F	Reserved																								1		1		Reserved		1		1		Reserved		0		0		0		0		0		0		0		0		0	
004h	GPIOx_POTYPE	x=A,B,C,D,F		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved																				
	Reset Value	x=A,B,C	Reserved																																																			
		x=D	Reserved																																																			
		x=F	Reserved																								0		0		0		0		0		0		0		0		0		0		0		0		0		0	
008h	GPIOx_SR	x=A,B,C,D,F		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved																				
	Reset Value	x=A,B,C	Reserved																																																			
		x=D	Reserved																																																			
		x=F	Reserved																								1		1		1		1		1		1		1		1		1		1		1		1		1		1	
00Ch	GPIOx_PUPD	x=A,B,C,D,F		PUPD15		PUPD14		PUPD13		PUPD12		PUPD11		PUPD10		PUPD9		PUPD8		PUPD7		PUPD6		PUPD5		PUPD4		PUPD3		PUPD2		PUPD1		PUPD0																				
	Reset Value	x=A	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																				
		x=B,C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																			
		x=D	Reserved																																																			
x=F	Reserved																								0		0		0		0		0		0		0		0		0		0		0		0		0		0		0	
010h	GPIOx_PID	x=A,B,C,D,F		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved																				
	Reset Value	x=A,B,C	Reserved																																																			
		x=D	Reserved																																																			
		x=F	Reserved																								0		0		0		0		0		0		0		0		0		0		0		0		0		0	
014h	GPIOx_POD	x=A,B,C,D,F		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved																				
	Reset Value	x=A,B,C	Reserved																																																			
		x=D	Reserved																																																			
		x=F	Reserved																								0		0		0		0		0		0		0		0		0		0		0		0		0		0	
018h	GPIOx_PBCS	x=A,B,C,D,F		PBC15		PBC14		PBC13		PBC12		PBC11		PBC10		PBC9		PBC8		PBC7		PBC6		PBC5		PBC4		PBC3		PBC2		PBC1		PBC0																				
	Reset Value	x=A,B,C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																				
		x=D	Reserved																																																			
		x=F	Reserved																								0		0		0		0		0		0		0		0		0		0		0		0		0		0	
01Ch	GPIOx_PLOCK	x=A,B,C,D,F		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved																				
	Reset Value	x=A,B,C	Reserved																																																			
		x=D	Reserved																																																			
		x=F	Reserved																								0		0		0		0		0		0		0		0		0		0		0		0		0		0	
020h	GPIOx_AFL	x=A,B,C,D,F		AFSEL7[3:0]				AFSEL6[3:0]				AFSEL5[3:0]				AFSEL4[3:0]				AFSEL3[3:0]				AFSEL2[3:0]				AFSEL1[3:0]				AFSEL0[3:0]																						
	Reset Value	x=A,B,C	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1																				
		x=D	Reserved																																																			
		x=F	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1																			
024h	GPIOx_AFH	x=A,B,C,D,F		AFSEL15[3:0]				AFSEL14[3:0]				AFSEL13[3:0]				AFSEL12[3:0]				AFSEL11[3:0]				AFSEL10[3:0]				AFSEL9[3:0]				AFSEL8[3:0]																						
	Reset Value	x=A	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																				
		x=B,C	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1																			
		x=D	Reserved																																																			
x=F	Reserved																								0		0		0		0		0		0		0		0		0		0		0		0		0		0		0	
028h	GPIOx_PBC	x=A,B,C,D,F		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved																				
	Reset Value	x=A,B,C	Reserved																																																			

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																						
	x=D	Reserved																									0	Reserved																											
	x=F	Reserved																									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
02Ch	GPIOx_DS	x=A,B,C,D,F	Reserved															DS15	DS14	DS13	DS12	DS11	DS10	DS9	DS8	DS7	DS6	DS5	DS4	DS3	DS2	DS1	DS0																						
		x=A,B,C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																								
	Reset Value	x=D	Reserved																									0	Reserved																										
		x=F	Reserved																									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

5.3.2 GPIO Port Mode Description Register (GPIOx_PMODE)

Offset address : 0x00

Reset value : 0xEBFF FFFC (x=A); 0xFFFF FFFF (x=B,C,D); 0xFFFF FFFC (x=F)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PMODE15[1:0]	PMODE14[1:0]	PMODE13[1:0]	PMODE12[1:0]	PMODE11[1:0]	PMODE10[1:0]	PMODE9[1:0]	PMODE8[1:0]								
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMODE7[1:0]	PMODE6[1:0]	PMODE5[1:0]	PMODE4[1:0]	PMODE3[1:0]	PMODE2[1:0]	PMODE1[1:0]	PMODE0[1:0]								
rw		rw		rw		rw		rw		rw		rw		rw	

Bit Field	Name	Description
31:30	PMODEy[1:0]	Mode of port GPIOx (x = A,B,C,D,F) pin PINy: 00: Input mode 01: General output mode 10: Alternate function mode 11: Analog function mode <i>Note: when x = A,B,C, y = 0...15;</i> <i>When x = D, y = 2, the remaining bits are reserved, and the reserved bits are read-only;</i> <i>When x = F, y = 0, 1, 2, 4, 5, 6, 7, the remaining bits are reserved, and the reserved bits are read-only.</i>
29:28		
27:26		
25:24		
23:22		
21:20		
19:18		
17:16		
15:14		
13:12		
11:10		
9:8		
7:6		
5:4		
3:2		
1:0		

5.3.3 GPIO Port Type Definition (GPIOx_POTYPE)

Offset address : 0x04

Reset value : 0x0000 0000 (x=A,B,C,D,F)

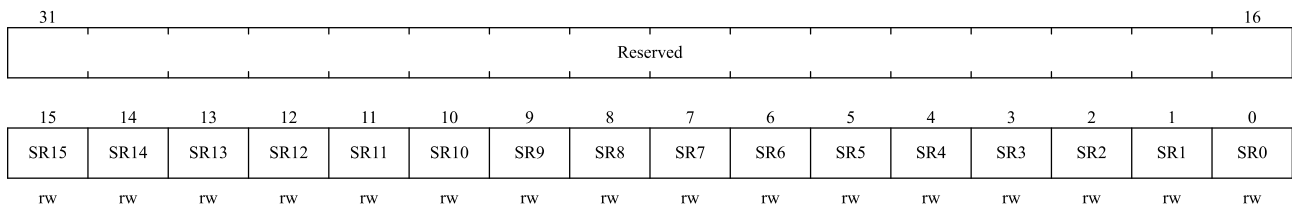
31	Reserved														16
POT15	POT14	POT13	POT12	POT11	POT10	POT9	POT8	POT7	POT6	POT5	POT4	POT3	POT2	POT1	POT0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit Field	Name	Description
31:16	Reserved	Reserved,the reset value must be maintained
15:0	POTy	Output type of port GPIOx (x = A,B,C,D,F) pin PINy: 0: Output push-pull mode (state after reset) 1: Output open-drain mode <i>Note: when x = A,B,C, y = 0...15;</i> <i>When x = D, y = 2, the remaining bits are reserved, and the reserved bits are read-only;</i> <i>When x = F, y = 0, 1, 2, 4, 5, 6, 7, the remaining bits are reserved, and the reserved bits are read-only.</i>

5.3.4 GPIO Slew Rate Configuration Register (GPIOx_SR)

Offset address : 0x08

Reset value : 0x0000 FFFF (x= A,B,C,D,F)

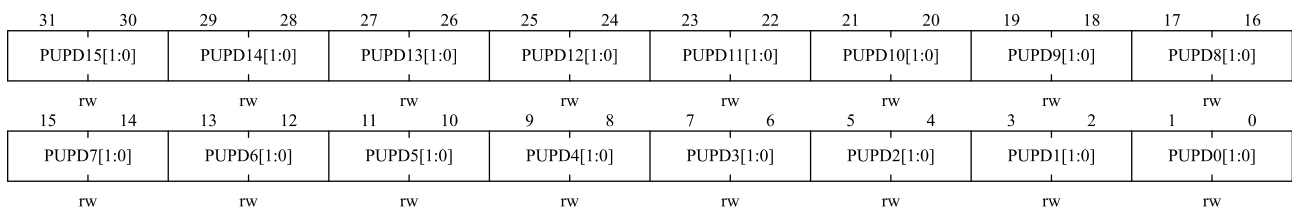


Bit Field	Name	Description
31:16	Reserved	Reserved,the reset value must be maintained
15:0	SRy	Slew rate configuration bits for port GPIOx (x = A,B,C,D,F) pin PINy 0: Fast slew rate 1: Slow slew rate <i>Note: when x = A,B,C, y = 0...15;</i> <i>When x = D, y = 2, the remaining bits are reserved, and the reserved bits are read-only;</i> <i>When x = F, y = 0, 1, 2, 4, 5, 6, 7, the remaining bits are reserved, and the reserved bits are read-only.</i>

5.3.5 GPIO Port Pull-up/Pull-down Register (GPIOx_PUPD)

Offset address : 0x0C

Reset value : 0x2400 0000 (x= A); 0x0000 0000 (x= B,C,D,F)

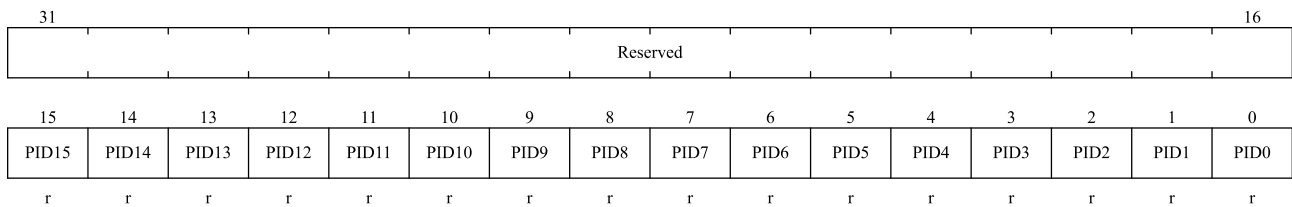


Bit Field	Name	Description
31:30	PUPDy[1:0]	Pull-up and pull-down mode of port GPIOx (x = A,B,C,D,F) pin PINy: 00: no pull-up/pull-down 01: Pull up 10: Pull down 11: Reserved <i>Note: when x = A,B,C, y = 0...15; When x = D, y = 2, the remaining bits are reserved, and the reserved bits are read-only; When x = F, y = 0, 1, 2, 4, 5, 6, 7, the remaining bits are reserved, and the reserved bits are read-only.</i>
29:28		
27:26		
25:24		
23:22		
21:20		
19:18		
17:16		
15:14		
13:12		
11:10		
9:8		
7:6		
5:4		
3:2		
1:0		

5.3.6 GPIO Port Input Data Register (GPIOx_PID)

Offset address : 0x10

Reset value : 0x0000 0000 (x=A,B,C,D,F)



Bit Field	Name	Description
31:16	Reserved	Reserved,the reset value must be maintained
15:0	PIDy	Input data of port GPIOx (x = A,B,C,D,F) pin PINy These bits are read-only, and the read value is the state of the corresponding I/O port. <i>Note: when x = A,B,C, y = 0...15; When x = D, y = 2, the remaining bits are reserved, and the reserved bits are read-only; When x = F, y = 0, 1, 2, 4, 5, 6, 7, the remaining bits are reserved, and the reserved bits are read-only.</i>

5.3.7 GPIO Port Output Data Register (GPIOx_POD)

Offset address : 0x14

Reset value : 0x0000 0000 (x=A,B,C,D,F)

Reserved															
31															16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained
15:0	PODy	Output data of port GPIOx (x = A,B,C,D,F) pin PINy These bits are readable or writable by software, and the corresponding POD bits can be independently set/cleared. <i>Note: when x = A,B,C, y = 0...15;</i> <i>When x = D, y = 2, the remaining bits are reserved, and the reserved bits are read-only;</i> <i>When x = F, y = 0, 1, 2, 4, 5, 6, 7, the remaining bits are reserved, and the reserved bits are read-only.</i>

5.3.8 GPIO Port Bit Set/Clear Register (GPIOx_PBSC)

Offset address : 0x18

Reset value : 0x0000 0000 (x=A,B,C,D,F)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PBC15	PBC14	PBC13	PBC12	PBC11	PBC10	PBC9	PBC8	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
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
PBS15	PBS14	PBS13	PBS12	PBS11	PBS10	PBS9	PBS8	PBS7	PBS6	PBS5	PBS4	PBS3	PBS2	PBS1	PBS0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

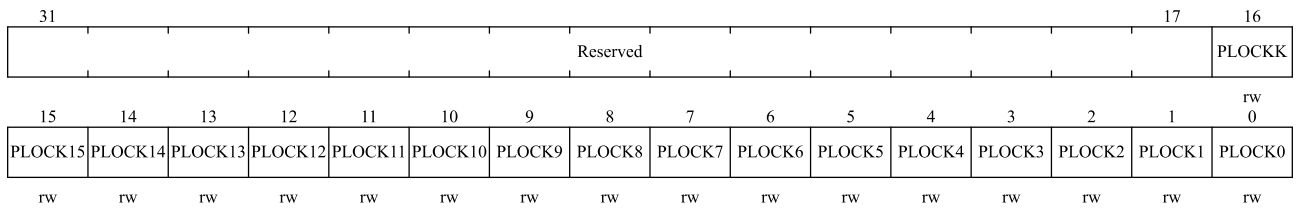
Bit Field	Name	Description
31:16	PBCy	Clear bit y of port GPIOx (x = A,B,C,D,F) These bits can only be written. 0: Does not affect the corresponding PODy bit 1: Clear the corresponding PODy bit to 0 <i>Note: If the corresponding bits of PBSy and PBCy are set at the same time, the PBSy bit works.</i> <i>Note: when x = A,B,C, y = 0...15;</i> <i>When x = D, y = 2, the remaining bits are reserved, and the reserved bits are read-only;</i> <i>When x = F, y = 0, 1, 2, 4, 5, 6, 7, the remaining bits are reserved, and the reserved bits are read-only.</i>
15:0	PBSy	Set bit y of port GPIOx (x = A,B,C,F) These bits can only be written. 0: Does not affect the corresponding PODy bit 1: Set the corresponding PODy bit to 1

Bit Field	Name	Description
		<i>Note: when $x = A,B,C$, $y = 0...15$;</i> When $x = D$, $y = 2$, the remaining bits are reserved, and the reserved bits are read-only; When $x = F$, $y = 0, 1, 2, 4, 5, 6, 7$, the remaining bits are reserved, and the reserved bits are read-only.

5.3.9 GPIO Port Configuration Lock Register (GPIOx_PLOCK)

Offset address : 0x1C

Reset value : 0x0000 0000 (x=A,B,C,D,F)

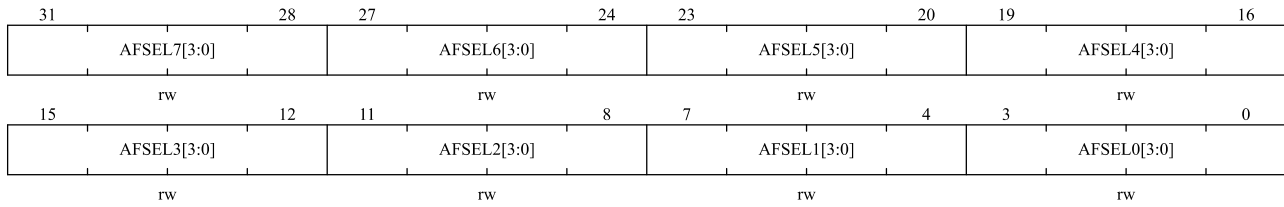


Bit Field	Name	Description
31:17	Reserved	Reserved, the reset value must be maintained
16	PLOCKK	Lock key This bit can be read at any time, it can only be modified by the lock key write sequence. 0: Port configuration lock key is not active 1: The port configuration lock key bit is activated, and the GPIOx_PLOCK register is locked before the next system reset. Lock key write sequence: write 1 -> write 0 -> write 1 -> read 0 -> (read 1) The last read 1 can be omitted, but can be used to confirm that the lock key has been activated. <i>Note: The value of PLOCK[15:0] cannot be changed while operating the lock key write sequence. Any errors in the operation lock key write sequence will not activate the lock key.</i>
15:0	PLOCKy	Configuration lock bit for port GPIOx ($x = A,B,C,D,F$) pin PINy These bits are readable and writable but can only be written when the PLOCKK bit is 0. 0: Do not lock the configuration of the port 1: Lock the configuration of the port <i>Note: when $x = A,B,C$, $y = 0...15$;</i> When $x = D$, $y = 2$, the remaining bits are reserved, and the reserved bits are read-only; When $x = F$, $y = 0, 1, 2, 4, 5, 6, 7$, the remaining bits are reserved, and the reserved bits are read-only.

5.3.10 GPIO Alternate Function Low Register (Gpiox_AFL)

Offset address : 0x20

Reset value : 0xFFFF FFFF (x= A,B,C,D,F)

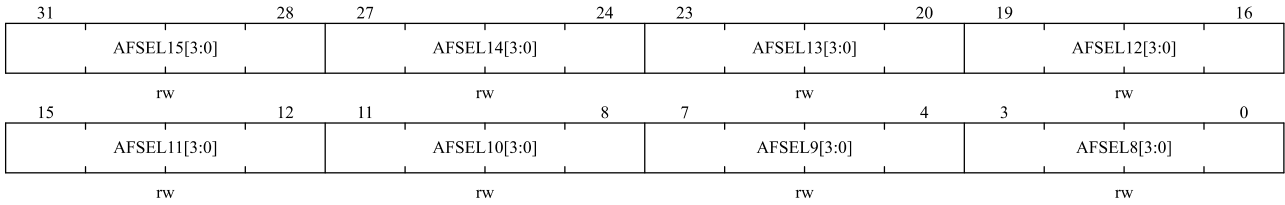


Bit Field	Name	Description	
31:28	AFSELY[3:0]	Alternate function configuration bits for port GPIOx (x = A,B,C,D,F) pins PINy (y = 0...7)	
27:24			
23:20			0000: AF0
19:16			0001: AF1
15:12			0010: AF2
11:8			0011: AF3
7:4			0100: AF4
3:0			0101: AF5
			0110: AF6
			0111: AF7
			1000: AF8
			1001: AF9
			1010: AF10
			1011: AF11
			1100: AF12
			1101: AF13
	1110: AF14		
	1111: AF15		
	<i>Note: when x = A,B,C, y = 0...7;</i>		
	<i>When x = D, y = 2, the remaining bits are reserved, and the reserved bits are read-only;</i>		
	<i>When x = F, y = 0, 1, 2, 4, 5, 6, 7, the remaining bits are reserved, and the reserved bits are read-only.</i>		

5.3.11 GPIO Alternate Function High Register (GPIOx_AFH)

Offset address : 0x24

Reset value : 0xF00F FFFF (x = A); 0xFFFF FFFF (x =B,C,D,F)

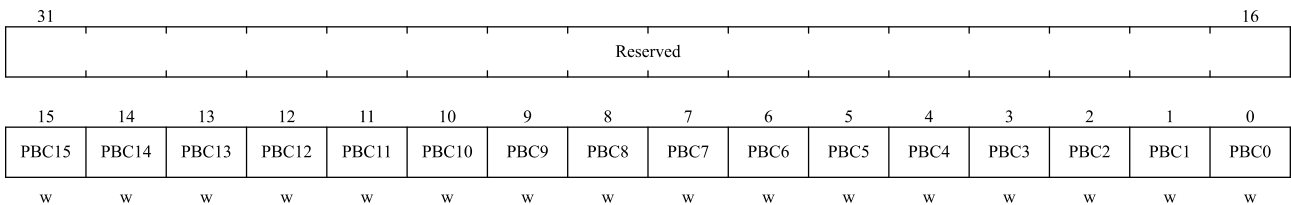


Bit Field	Name	Description
31:28 27:24 23:20 19:16 15:12 11:8 7:4 3:0	AFSELy[3:0]	Alternate Function Configuration Bits for Port GPIOx (x = A,B,C,D,F) Pins PINy (y = 8...15) 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15 <i>Note: when x = A,B,C, y = 8...15; When x = D, F, all bits are reserved, and the reserved bits are read-only;</i>

5.3.12 GPIO Port Bit Clear Register (GPIOx_PBC)

Offset address : 0x28

Reset value : 0x0000 0000 (x=A,B,C,D,F)



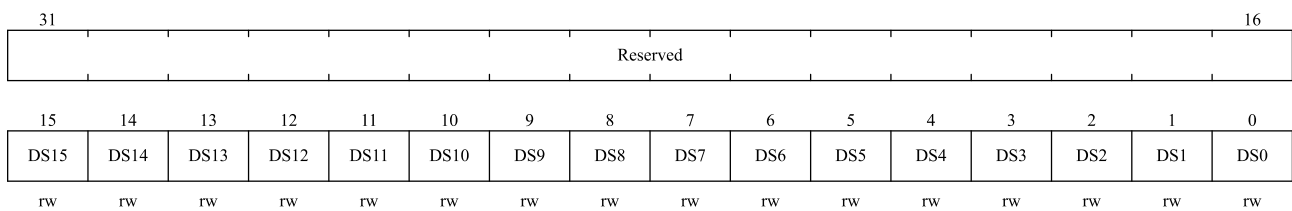
Bit Field	Name	Description
31:16	Reserved	Reserved,the reset value must be maintained
15:0	PBCy	Clear bit y of port GPIOx (x = A,B,C,D,F) These bits can only be written. 0: No effect on the corresponding POCy bit

Bit Field	Name	Description
		1: Clear the corresponding PODy bit to 0 <i>Note: when x = A,B,C, y = 0...15;</i> <i>When x = D, y = 2, the remaining bits are reserved, and the reserved bits are read-only;</i> <i>When x = F, y = 0, 1, 2, 4, 5, 6, 7, the remaining bits are reserved, and the reserved bits are read-only.</i>

5.3.13 GPIO Driver Strength Configuration Register (GPIOx_DS)

Offset address : 0x2C

Reset value : 0x0000 0000 (x=A,B,C,D,F)



Bit Field	Name	Description
31:16	Reserved	Reserved,the reset value must be maintained
15:0	DSy	Drive capability configuration bits for port GPIOx (x = A,B,C,D,F) pins PINy 0: High drive capability (16mA(5V)/8mA(3.3V)/4mA(1.8V)) 1: Low drive capability (8mA(5V)/4mA(3.3V)/2mA(1.8V)) <i>Note: when x = A,B,C, y = 0...15;</i> <i>When x = D, y = 2, the remaining bits are reserved, and the reserved bits are read-only;</i> <i>When x = F, y = 0, 1, 2, 4, 5, 6, 7, the remaining bits are reserved, and the reserved bits are read-only.</i>

5.4 AFIO Registers

5.4.1 AFIO Register Overview

AFIO base address: 0x40010000

Table 5-50 AFIO Register Overview

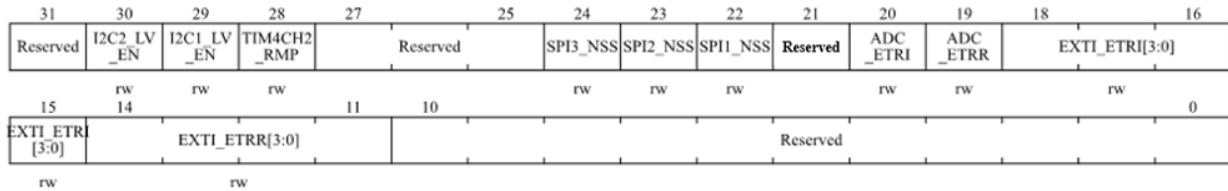
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0														
000h	AFIO_CFG	Reserved	I2C2_LV_EN	I2C1_LV_EN	TIM4CH2_RMP	Reserved			SPI3_NSS	SPI2_NSS	SPI1_NSS	Reserved	ADC_ETRI	ADC_ATTR	EXTI_ETRI [3:0]			EXTI_ETRR [3:0]			Reserved																										
	Reset Value		0	0	0				0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0														
008h	AFIO_EXTI_CFG1	Reserved																EXTI3_CFG [3:0]			EXTI2_CFG [3:0]			EXTI1_CFG [3:0]			EXTI0_CFG [3:0]																				
	Reset Value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
00Ch	AFIO_EXTI_CFG2	Reserved																EXTI7_CFG [3:0]			EXTI6_CFG [3:0]			EXTI5_CFG [3:0]			EXTI4_CFG [3:0]																				
	Reset Value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
010h	AFIO_EXTI_CFG3	Reserved																EXTI11_CFG [3:0]			EXTI10_CFG [3:0]			EXTI9_CFG [3:0]			EXTI8_CFG [3:0]																				
	Reset Value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0															
	Reset Value	Reserved																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
014h	AFIO_EXTI_CFG4	Reserved																EXTI15_CFG [3:0]				EXTI14_CFG [3:0]				EXTI13_CFG [3:0]				EXTI12_CFG [3:0]																		
	Reset Value	Reserved																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

5.4.2 AFIO Configuration Register (AFIO_CFG)

Offset address : 0x00

Reset value : 0x0000 0000



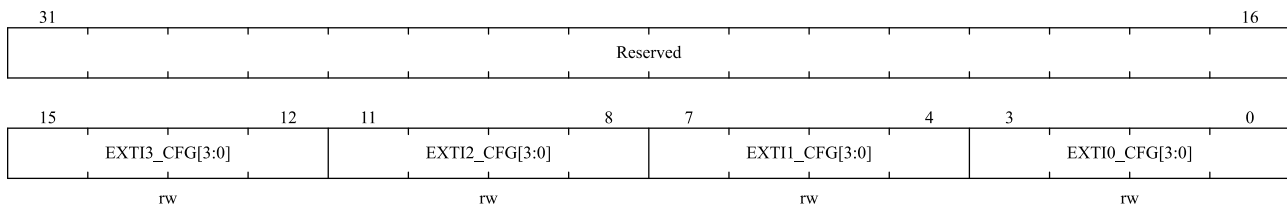
Bit Field	Name	Description
31	Reserved	Reserved,the reset value must be maintained
30	I2C2_LV_EN	I2C2 I/O Signal Low-Level Enable: 0: V _{DD} level enabled 1: Low level enabled
29	I2C1_LV_EN	I2C1 I/O Signal Low-Level Enable: 0: V _{DD} level enabled 1: Low level enabled
28	TIM4CH2_RMP	TIM4 Channel 2 Internal Remapping: This bit can be set to '1' or '0' by software. It controls the internal mapping of TIM4_CH2. 0: TIM4_CH2 is connected to PA7 1: LSI internal clock is connected to TIM4_CH2 input for calibration of LSI Reserved, must be kept at reset value.
27:25	Reserved	Reserved,the reset value must be maintained
24	SPI3_NSS	NSS mode selection bit of SPI3 (NSS is configured in AFIO push-pull mode). 0: NSS is high impedance when idle 1: NSS is high level when idle
23	SPI2_NSS	NSS mode selection bit of SPI2 (NSS is configured in AFIO push-pull mode). 0: NSS is high impedance when idle 1: NSS is high level when idle
22	SPI1_NSS	NSS mode selection bit of SPI1 (NSS is configured in AFIO push-pull mode). 0: NSS is high impedance when idle 1: NSS is high level when idle
21	Reserved	Reserved,the reset value must be maintained
20	ADC_ETRI	ADC injection conversion external trigger remapping This bit can be set to '1' or '0' by software. It controls the trigger input connected to the external trigger for ADC injection conversion.

Bit Field	Name	Description
		0: ADC injection conversion external trigger is connected to EXTI (0-15) 1: ADC injection conversion external trigger is connected to TIM8_CH4.
19	ADC_ETRR	ADC rugular conversion external trigger remapping This bit can be set to '1' or '0' by software. It controls the trigger input connected to the ADC rugular conversion external trigger. 0: ADC rugular conversion external trigger is connected to EXTI (0-15) 1: The ADC rugular conversion external trigger is connected to TIM8_TRGO.
18:15	EXTI_ETRI[3:0]	Select EXTI Line injection to convert external trigger remapping 0000: select EXTI0 injection conversion external trigger 0001: Select EXTI1 injection to convert external trigger ... 1111: Select EXTI15 injection conversion external trigger
14:11	EXTI_ETRR[3:0]	Select EXTI Line rugular to convert external trigger remapping 0000: Select EXTI0 rugular to convert external trigger 0001: Select EXTI1 rugular to convert external trigger ... 1111: Select EXTI15 rugular to convert external trigger
10:0	Reserved	Reserved,the reset value must be maintained

5.4.3 AFIO External Interrupt Configuration Register 1 (AFIO_EXTI_CFG1)

Offset address : 0x08

Reset value : 0x0000 0000



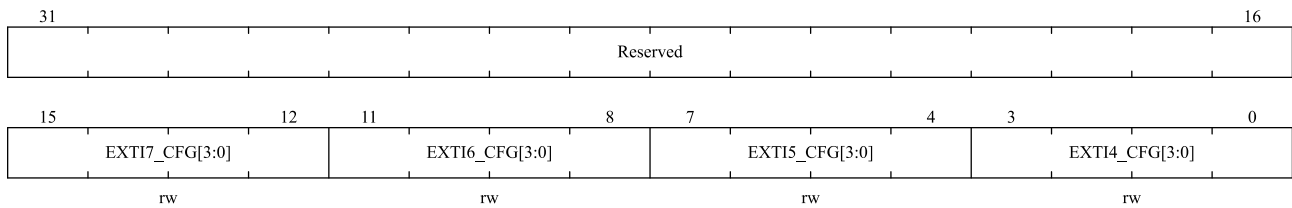
Bit Field	Name	Description
31:30	Reserved	Reserved,the reset value must be maintained
15:0	EXTIx_CFG[3:0]	EXTIx configuration (x = 0...3) These bits are readable and writable by software and are used to select the input source for the EXTIx external interrupt. EXTI0 configuration: 0000: PA0 pin 0001: PB0 pin 0010: reserved 0101: PF0 pin Other: reserved EXTI1 configuration: 0000: PA1 pin 0001: PB1 pin 0010: reserved 0101: PF1 pin Other: reserved EXTI2 configuration: 0000: PA2 pin 0001: PB2 pin 0010: reserved

Bit Field	Name	Description
		0101: PF2 pin Other: reserved EXTI3 configuration: 0000: PA3 pin 0001: PB3 pin 0010: reserved 0101: reserved Other: reserved

5.4.4 AFIO External Interrupt Configuration Register 2 (AFIO_EXTI_CFG2)

Offset address : 0x0C

Reset value : 0x0000 0000

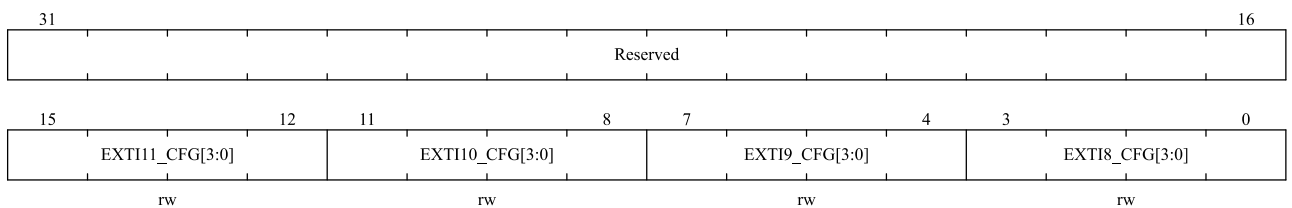


Bit Field	Name	Description
31:30	Reserved	Reserved, the reset value must be maintained
15:0	EXTIx_CFG[3:0]	EXTIx configuration (x = 4..7) These bits are readable and writable by software and are used to select the input source for the EXTIx external interrupt. EXTI4 configuration: 0000: PA4 pin 0001: PB4 pin 0010: reserved 0101: reserved Other: reserved EXTI5 configuration: 0000: PA5 pin 0001: PB5 pin 0010: reserved 0101: reserved Other: reserved EXTI6 configuration: 0000: PA6 pin 0001: PB6 pin 0010: reserved 0101: PF6 pin Other: reserved EXTI7 configuration: 0000: PA7 pin 0001: PB7 pin 0010: reserved 0101: PF7 pin Other: reserved

5.4.5 AFIO External Interrupt Configuration Register 3 (AFIO_EXTI_CFG3)

Offset address : 0x10

Reset value : 0x0000 0000

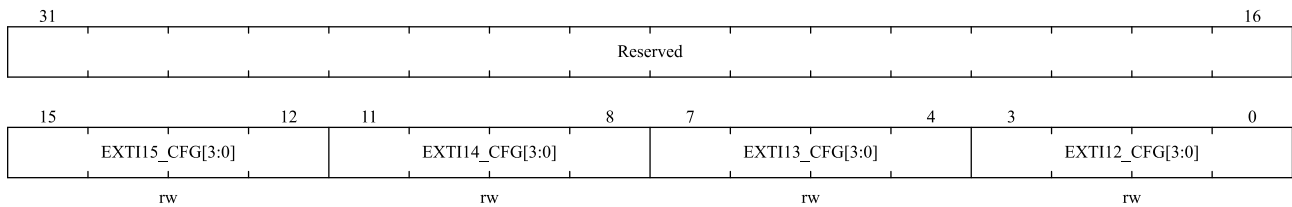


Bit Field	Name	Description
31:30	Reserved	Reserved,the reset value must be maintained
15:0	EXTIx_CFG[3:0]	<p>EXTIx configuration (x = 8...11)</p> <p>These bits are readable and writable by software and are used to select the input source for the EXTIx external interrupt.</p> <p>EXTI8 configuration:</p> <p>0000: PA8 pin 0001: PB8 pin 0010: reserved 0101: reserved Other: reserved</p> <p>EXTI9 configuration:</p> <p>0000: PA9 pin 0001: PB9 pin 0010: reserved 0101: reserved Other: reserved</p> <p>EXTI10 configuration:</p> <p>0000: PA10 pin 0001: PB10 pin 0010: reserved 0101: reserved Other: reserved</p> <p>EXTI11 configuration:</p> <p>0000: PA11 pin 0001: PB11 pin 0010: reserved 0101: reserved Other: reserved</p>

5.4.6 AFIO External Interrupt Configuration Register 4 (AFIO_EXTI_CFG4)

Offset address : 0x14

Reset value : 0x0000 0000



Bit Field	Name	Description
31:30	Reserved	Reserved,the reset value must be maintained
15:0	EXTIx_CFG[3:0]	<p>EXTIx configuration (x = 12...15)</p> <p>These bits are readable and writable by software and are used to select the input source for the EXTIx external interrupt.</p> <p>EXTII2 configuration:</p> <p>0000: PA12 pin 0001: PB12 pin 0010: reserved 0101: reserved Other: reserved</p> <p>EXTII3 configuration:</p> <p>0000: PA13 pin 0001: PB13 pin 0010: PC13 pin 0101: reserved Other: reserved</p> <p>EXTII4 configuration:</p> <p>0000: PA14 pin 0001: PB14 pin 0010: PC14 pin 0101: reserved Other: reserved</p> <p>EXTII5 configuration:</p> <p>0000: PA15 pin 0001: PB15 pin 0010: PC15 pin</p>

Bit Field	Name	Description
		0101: reserved Other: reserved

6 Interrupts and Events

6.1 Nested Vectored Interrupt Controller

Features

- 32 maskable interrupt channels (excluding 16 Cortex[®]-M0 interrupt lines);
- 4 programmable priorities (using 2 interrupt priorities);
- Low latency exception and interrupt handling;
- Power management control;
- The implementation of system control register;

The nested vectored interrupt controller (NVIC) is closely linked to the processor core, enabling low latency interrupt processing and efficient processing of late interrupts. The nested vectored interrupt controller manages interrupts including core exceptions.

6.1.1 SysTick calibration value Register

The system tick calibration value is fixed at 6000. When the system tick clock is set to 6MHz (the maximum value of HCLK/8), 1ms time base is generated.

6.1.2 Interrupt and Exception Vectors

Table 6-1 Vector Table

Position	Priority	Priority Type	Name	Description	Address
-	-	-	-	Reserved	0x0000 0000
-	-3	Fixed	Reset	Reset	0x0000 0004
-	-2	Fixed	NMI	Non-maskable interrupt.RCC clock security system (CSS) is connected to the NMI vector.	0x0000 0008
-	-1	Fixed	HardFault	All types of errors (fault)	0x0000 000C
-	3	Settable	SVCall	System services invoked by SWI directives	0x0000 002C
-	5	Settable	PendSV	System service requests that can be pending	0x0000 0038
-	6	Settable	SysTick	System tick timer	0x0000 003C
0	7	Settable	WWDG	Window watchdog interrupt	0x0000 0040
1	8	Settable	PVD	PVD interrupt connected to EXTI line 16	0x0000 0044
2	9	Settable	RTC	RTC interrupt connected to EXTI lines 17, 19 and 20	0x0000 0048
3	10	Settable	MMU	MMU global interrupt	0x0000 004C
4	11	Settable	FLASH	Flash global interrupt	0x0000 0050
5	12	Settable	RCC	RCC global interrupt	0x0000 0054

Position	Priority	Priority Type	Name	Description	Address
6	13	Settable	EXTI0_1	The EXTI line [1:0] interrupt	0x0000 0058
7	14	Settable	EXTI2_3	The EXTI line [3:2] interrupt	0x0000 005C
8	15	Settable	EXTI4_15	The EXTI line [15:4] interrupt	0x0000 0060
9	16	Settable	Reserved	Reserved	0x0000 0064
10	17	Settable	SAC	SAC interrupt	0x0000 0068
11	18	Settable	DMA_CH1_2_3_4	DMA channel 1/2/3/4 interrupt	0x0000 006C
12	19	Settable	DMA_CH5_6_7_8	DMA channel 5/6/7/8 interrupt	0x0000 0070
13	20	Settable	TIM1_BRK_UP_TRG_COM	TIM1 brakes, updates, triggers and communication interrupt	0x0000 0074
14	21	Settable	TIM1_CC	TIM1 capture comparison interrupt	0x0000 0078
15	22	Settable	CAN	CAN interrupt	0x0000 007C
16	23	Settable	TIM3	TIM3 global interrupt	0x0000 0080
17	24	Settable	TIM4	TIM4 global interrupt	0x0000 0084
18	25	Settable	TIM8_BRK_UP_TRG_COM	TIM8 brakes, updates, triggers and communication interrupt	0x0000 0088
19	26	Settable	TIM8_CC	TIM8 capture comparison interrupt	0x0000 008C
20	27	Settable	LPTIM/TIM6	LPTIM (connected to EXTI line 23) /TIM6 global interrupt	0x0000 0090
21	28	Settable	ADC	ADC global interrupt	0x0000 0094
22	29	Settable	SPI2/SPI3	SPI2/SPI3 global interrupt	0x0000 0098
23	30	Settable	I2C1	I2C1 global interruption	0x0000 009C
24	31	Settable	I2C2	I2C2 global interrupt	0x0000 00A0
25	32	Settable	SPI1	SPI1 global interrupt	0x0000 00A4
26	33	Settable	HDIV/SQRT	Divider/square operator global interrupt	0x0000 00A8
27	34	Settable	RAMC_ERR	RAMC_ERR global interrupt	0x0000 00AC
28	35	Settable	USART1/USART2	USART1/USART2 global interrupt	0x0000 00B0
29	36	Settable	LPUART1/ LPUART2	LPUART1/ LPUART2 global interrupt (connected to EXTI line 22)	0x0000 00B4
30	37	Settable	UART5/ UART6	UART5/ UART6 global interrupt	0x0000 00B8
31	38	Settable	COMP1/COMP2/COMP3	COMP1(connected to EXTI line 18) /COMP2/COMP3 interrupt	0x0000 00BC

6.2 Extended Interrupt/Event Controller (EXTI)

6.2.1 Introduction

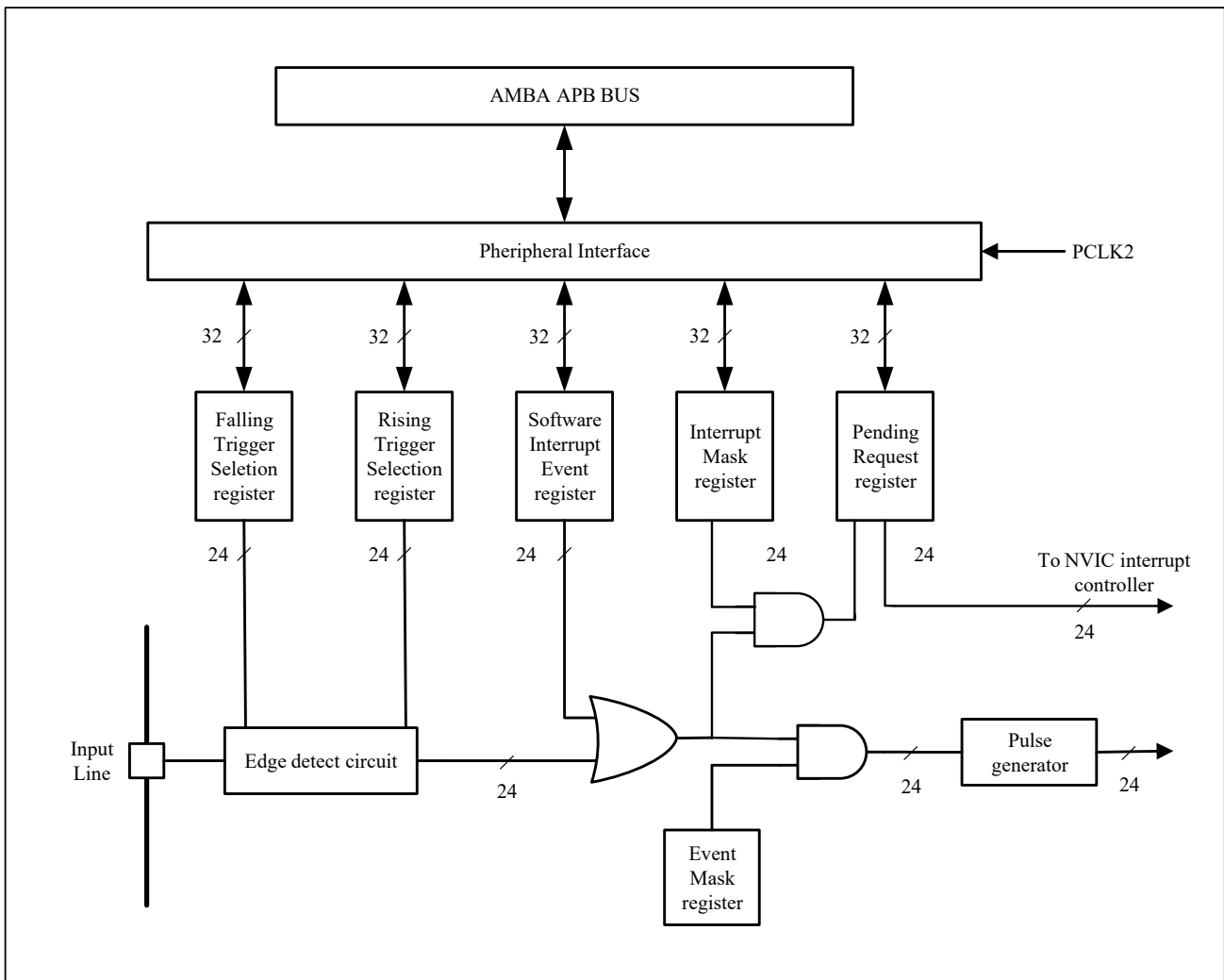
The extended interrupt/event controller contains 24 edge detection circuits that trigger interrupt/event triggers. Each input line can be independently configured with pulse or pending input types, and three trigger event types including rising edge, falling edge or double edges, which can also be independently masked. Interrupt requests that hold the state line in the pending register can be cleared by writing '1' in the corresponding bit of the pending register.

6.2.2 Main Features

The main features of EXTI controller are as follows:

- Supports 24 software interrupt/event requests
- Interrupts/events corresponding to each input line can be configured to trigger or mask independently
- Each interrupt line has an independent state bit
- Support for pulse or pending input types
- 3 trigger events are supported: rising edge, falling edge, and double edges
- Can wake up MUC to exit low power mode

Figure 6-1 External Interrupt/Event Controller Block Diagram



6.2.3 Functional Description

The EXTI contains 24 interrupt lines, 16 lines from I/O pins and 8 lines from internal modules. To trigger interrupts, the NVIC interrupt channel of the extended interrupt controller must be configured to enable the appropriate interrupt lines. Select rising edge, falling edge, or double edges trigger event types by edge trigger configuration registers EXTI_RT_CFG and EXTI_FT_CFG, and write '1' to the corresponding bit of interrupt masking register EXTI_IMASK to allow interrupt requests. When a preset edge trigger polarity is detected on the external interrupt line, an interrupt request is generated and the corresponding pending bit is set to '1'. Writing '1' to the corresponding bit of the pending register clears the interrupt request.

To generate events, the corresponding event line must be configured and enabled. According to the desired edge detection polarity, set up the rise/fall edge trigger configuration register, while writing '1' in the corresponding bit of the event masking register to allow interrupt requests. When a preset edge occurs on an event line, an event request pulse is generated and the corresponding pending bit is not set to '1'.

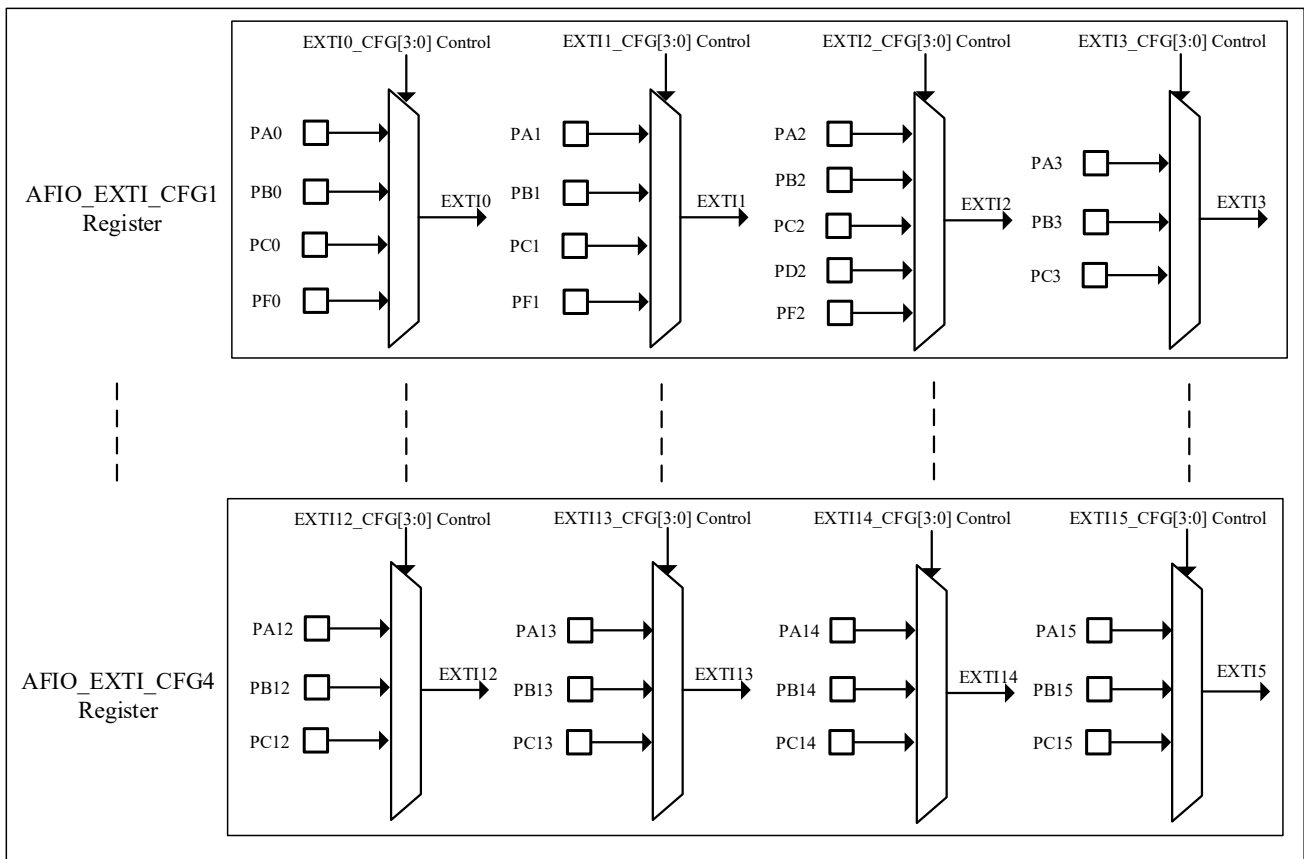
In addition, interrupt/event requests can also be generated by software by writing a '1' in the software interrupt/event register.

- Hardware interrupt configuration, select and configure 24 lines as interrupt sources as required:

- Configure the mask bit (EXTI_IMASK) for 24 interrupt lines.
- Configure the Trigger Selection bits of the selected interrupt line(EXTI_RT_CFG and EXTI_FT_CFG);
- Configure the enable and mask bits of the NVIC interrupt channel corresponding to the extended interrupt controller so that the requests in the 24 interrupt lines can be correctly responded to.
- Hardware event configuration: Select 24 lines as event sources as required:
 - Configure the mask bit (EXTI_EMASK) for 24 event lines.
 - Configure the Trigger Selection bits for the selected event line (EXTI_RT_CFG and EXTI_FT_CFG).
- Software interrupt/event configuration, select 24 lines as software interrupt/event lines as required:
 - Configure 24 interrupt/event line mask bits (EXTI_IMASK and EXTI_EMASK).
 - Configure the request bit of the software interrupt event register (EXTI_SWIE).

6.2.4 EXTI Line Mapping

Figure 6-2 External Interrupt Generic I/O Mapping



To configure external interrupts/events on the GPIO line using AFIO_EXTI_CFG1~4, the AFIO clock must be enabled first. Universal I/O ports are connected to 16 external interrupt/event lines as shown above. The connection mode of the other 8 EXTI lines is as follows:

- EXTI line 16 is connected to the PVD output
- EXTI line 17 is connected to the RTC alarm

- EXTI line 18 is connected to the COMP1 wake up event
- EXTI line 19 is connected to the RTC tamper for detection or timestamp wake up event
- EXTI line 20 is connected to the RTC wake up event
- EXTI line 21 is reserve
- EXTI line 22 is connected to the LPUART1/2 wake up event
- EXTI line 23 is connected to the LPTIM wake up event

6.3 EXTI Registers

EXTI base address: 0x40010400

6.3.1 EXTI Register Overview

Table 6-2 Exti Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
000h	EXTI_IMASK	Reserved									IMASK23	IMASK22	IMASK21	IMASK20	IMASK19	IMASK18	IMASK17	IMASK16	IMASK15	IMASK14	IMASK13	IMASK12	IMASK11	IMASK10	IMASK9	IMASK8	IMASK7	IMASK6	IMASK5	IMASK4	IMASK3	IMASK2	IMASK1	IMASK0	
	Reset Value										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
004h	EXTI_EMASK	Reserved									EMASK23	EMASK22	EMASK21	EMASK20	EMASK19	EMASK18	EMASK17	EMASK16	EMASK15	EMASK14	EMASK13	EMASK12	EMASK11	EMASK10	EMASK9	EMASK8	EMASK7	EMASK6	EMASK5	EMASK4	EMASK3	EMASK2	EMASK1	EMASK0	
	Reset Value										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
008h	EXTI_RT_CFG	Reserved									RT_CFG23	RT_CFG22	RT_CFG21	RT_CFG20	RT_CFG19	RT_CFG18	RT_CFG17	RT_CFG16	RT_CFG15	RT_CFG14	RT_CFG13	RT_CFG12	RT_CFG11	RT_CFG10	RT_CFG9	RT_CFG8	RT_CFG7	RT_CFG6	RT_CFG5	RT_CFG4	RT_CFG3	RT_CFG2	RT_CFG1	RT_CFG0	
	Reset Value										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00Ch	EXTI_FT_CFG	Reserved									FT_CFG23	FT_CFG22	FT_CFG21	FT_CFG20	FT_CFG19	FT_CFG18	FT_CFG17	FT_CFG16	FT_CFG15	FT_CFG14	FT_CFG13	FT_CFG12	FT_CFG11	FT_CFG10	FT_CFG9	FT_CFG8	FT_CFG7	FT_CFG6	FT_CFG5	FT_CFG4	FT_CFG3	FT_CFG2	FT_CFG1	FT_CFG0	
	Reset Value										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
010h	EXTI_SWIE	Reserved									SWIE23	SWIE22	SWIE21	SWIE20	SWIE19	SWIE18	SWIE17	SWIE16	SWIE15	SWIE14	SWIE13	SWIE12	SWIE11	SWIE10	SWIE9	SWIE8	SWIE7	SWIE6	SWIE5	SWIE4	SWIE3	SWIE2	SWIE1	SWIE0	
	Reset Value										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
014h	EXTI_PEND	Reserved									PEND23	PEND22	PEND21	PEND20	PEND19	PEND18	PEND17	PEND16	PEND15	PEND14	PEND13	PEND12	PEND11	PEND10	PEND9	PEND8	PEND7	PEND6	PEND5	PEND4	PEND3	PEND2	PEND1	PEND0	
	Reset Value										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
018h	EXTI_TS_SEL	Reserved																								TSSEL[3:0]									
	Reset Value																									0	0	0	0						

6.3.2 Interrupt Mask Register(EXTI_IMASK)

Address offset : 0x00

Reset value : 0x00000000

31	Reserved												24	23	22	21	20	19	18	17	16	
												IMASK23	IMASK22	IMASK21	IMASK20	IMASK19	IMASK18	IMASK17	IMASK16			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
IMASK15	IMASK14	IMASK13	IMASK12	IMASK11	IMASK10	IMASK9	IMASK8	IMASK7	IMASK6	IMASK5	IMASK4	IMASK3	IMASK2	IMASK1	IMASK0							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw							

Bit Field	Name	Describe
31:24	Reserved	Reserved, the reset value must be maintained.
23:0	IMASKx	Interrupt mask on line x. (x is 0,1,2...19,23) 0: Mask the interrupt request from line x;

		1: open the interrupt request from line x
--	--	-------------------------------------------

6.3.3 Event Mask Register(EXTI_EMASK)

Address offset : 0x04

Reset value : 0x00000000

31									24	23	22	21	20	19	18	17	16
Reserved										EMASK23	EMASK22	EMASK21	EMASK20	EMASK19	EMASK18	EMASK17	EMASK16
										rw	rw	rw	rw	rw	rw	rw	rw
EMASK15	EMASK14	EMASK13	EMASK12	EMASK11	EMASK10	EMASK9	EMASK8	EMASK7	EMASK6	EMASK5	EMASK4	EMASK3	EMASK2	EMASK1	EMASK0		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit Field	Name	Describe
31:24	Reserved	Reserved, the reset value must be maintained.
23:0	EMASKx	Event masking on line x. (x is 0,1,2...19,23) 0: Masking the event request from line x; 1: open the event request from line x

6.3.4 Rising Edge Trigger Selection Register(EXTI_RT_CFG)

Address offset : 0x08

Reset value : 0x00000000

31									24	23	22	21	20	19	18	17	16
Reserved										RT_CFG23	RT_CFG22	RT_CFG21	RT_CFG20	RT_CFG19	RT_CFG18	RT_CFG17	RT_CFG16
										rw	rw	rw	rw	rw	rw	rw	rw
RT_CFG15	RT_CFG14	RT_CFG13	RT_CFG12	RT_CFG11	RT_CFG10	RT_CFG9	RT_CFG8	RT_CFG7	RT_CFG6	RT_CFG5	RT_CFG4	RT_CFG3	RT_CFG2	RT_CFG1	RT_CFG0		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit Field	Name	Describe
31:24	Reserved	Reserved, the reset value must be maintained.
23:0	RT_CFGx	The rising edge on line x triggers the configuration bit. (x is 0,1,2...19,23) 0: Disables rising edge trigger (interrupts and events) on input line x. 1: Enable rising edge trigger (interrupts and events) on input line x.

6.3.5 Falling Edge Trigger Selection Register(EXTI_FT_CFG)

Address offset : 0x0C

Reset value : 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								FT_CFG23	FT_CFG22	FT_CFG21	FT_CFG20	FT_CFG19	FT_CFG18	FT_CFG17	FT_CFG16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FT_CFG15	FT_CFG14	FT_CFG13	FT_CFG12	FT_CFG11	FT_CFG10	FT_CFG9	FT_CFG8	FT_CFG7	FT_CFG6	FT_CFG5	FT_CFG4	FT_CFG3	FT_CFG2	FT_CFG1	FT_CFG0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit Field	Name	Describe
31:24	Reserved	Reserved, the reset value must be maintained.
23:0	FT_CFGx	The falling edge on line x triggers the configuration bit. (x is 0,1,2...19,23) 0: Disables falling edge trigger (interrupts and events) on input line x. 1: Enable falling edge trigger (interrupts and events) on input line x is allowed.

6.3.6 Software Interrupt Enable Register(EXTI_SWIE)

Address offset : 0x10

Reset value : 0x00000000

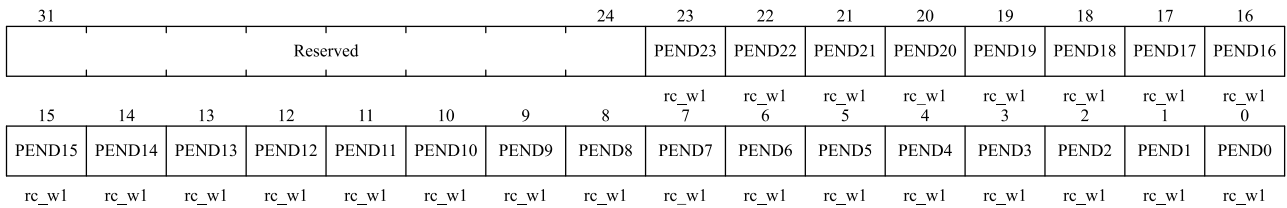
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								SWIE23	SWIE22	Reserved	SWIE20	SWIE19	SWIE18	SWIE17	SWIE16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWIE15	SWIE14	SWIE13	SWIE12	SWIE11	SWIE10	SWIE9	SWIE8	SWIE7	SWIE6	SWIE5	SWIE4	SWIE3	SWIE2	SWIE1	SWIE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit Field	Name	Describe
31:24	Reserved	Reserved, the reset value must be maintained.
23:0	SWIE _x	Software interrupt on line x. (x is 0,1,2...19,23) When the bit is '0', writing '1' sets the corresponding pending bit in EXTI_PEND. If this interrupt is allowed in EXTI_IMASK and EXTI_EMASK, an interrupt will be generated. <i>Note: This bit can be cleared to '0' by writing '1' to clear the corresponding bit of EXTI_PEND.</i>

6.3.7 Interrupt Request Pending Register(EXTI_PEND)

Address offset : 0x14

Reset value : 0x00000000

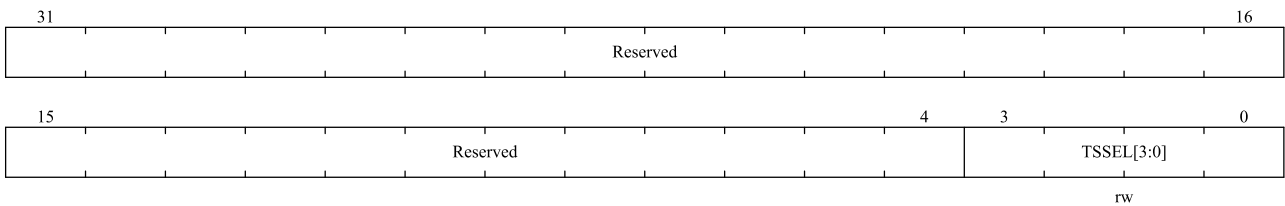


Bit Field	Name	Describe
31:24	Reserved	Reserved, the reset value must be maintained.
23:0	PENDx	Hang bit on line x. (x is 0,1,2...19,23) 0: No pending request occurred. 1: A pending trigger request has occurred. This bit is set to '1' when a selected edge trigger event occurs on the external interrupt line. It can be cleared by writing '1' to the bit, or by changing the polarity of the edge detection.

6.3.8 RTC Timestamp Trigger Source Selection Register (EXTI_TS_SEL)

Address offset : 0x18

Reset value : 0x00000000



Bit Field	Name	Describe
31:4	Reserved	Reserved, the reset value must be maintained.
3:0	TSSEL[3:0]	Select external interrupt input as trigger source of timestamp event. 0: Select EXTI0 as the trigger source of timestamp event; 1: select EXTI1 as the trigger source of timestamp event; 15: Select EXTI15 as the trigger source of timestamp events.

7 DMA Controller

7.1 Introduction

The DMA controller can access totally 5 AHB slaves: Flash, SRAM, ADC, APB1 and APB2. DMA Controller is controlled by CPU to perform fast data transfer from source to destination. After configuration, data can be transferred without CPU intervention. Thus, CPU can be released for other computation/control tasks or save overall system power consumption.

The main architecture of the MCU is a multi-layer AHB-Lite bus structure with round-robin arbitration scheme. DMA and CPU core can access different slaves in parallel or same slaves sequentially.

DMA controller has 8 logic channels. Each logic channel is to serve memory access requests from single or multiple peripherals. Internal arbiter controls the priority of different DMA channels.

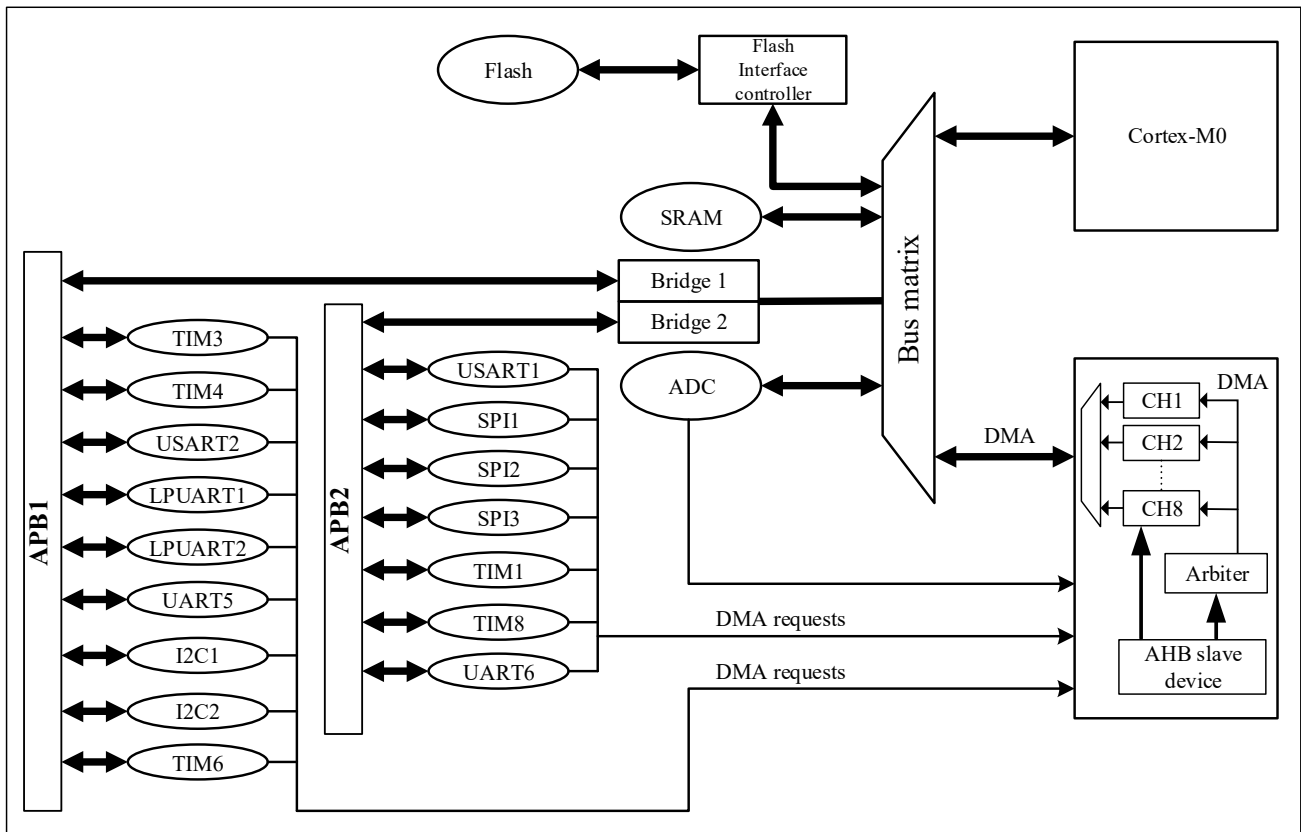
7.2 Main Features

DMA main features:

- 8 DMA channels which can be configured independently.
- Each DMA channel supports hardware requests and software triggers to initiate transfer, and is configured by software.
- Each DMA channel has dedicated software priority level (DMA_CHCFGx.PRIOLVL [1:0] bits, corresponding to 4 levels of priority) which can be configured individually. Channels with the same software priority level will further compare hardware index (channel number) to decide final priority (lower index number channel will has higher priority).
- Configurable source and destination size. Address setting should correspond to data size.
- Configurable circular transfer mode for each channel.
- Each channel has 3 independent event flags and interrupts (Transfer complete, Half transfer, Transfer error), and 1 global interrupt flag (set by logical or of 3 events).
- Support three transfer types which are Memory-to-Memory, Memory-to-Peripheral and Peripheral-to-Memory.
- Access totally 5 AHB slaves: Flash, SRAM, ADC, APB1 and APB2.
- Configurable data transmit number (0~65535).

7.3 Block Diagram

Figure 7-1 DMA Block Diagram



7.4 Function Description

DMA controller and Cortex[®]-M0 core share the same system data bus. When CPU and DMA access the same destination (RAM or peripheral) at the same time, DMA request will suspend CPU from accessing the system bus for several cycles, and the bus arbiter will perform round-robin scheduling. This allows the CPU to get at least half of the system bus (memory or peripheral) bandwidth.

7.4.1 DMA Operation

A DMA request can be triggered by hardware peripherals or software, and the DMA controller processes the request according to the priority level of the channel. The data is read from the source address according to the configured transfer address and bit width, and then the read data is stored in the destination address space. After one operation, the controller calculates the number of remaining transfers and updates the source address and the destination address of the next transfer.

Each DMA data transfer consists of three operations:

- Data access: determine the source address (DMA_PADDRx or DMA_MADDRx) according to the transfer direction and read data from the source address.
- Data storage: determine the destination address (DMA_PADDRx or DMA_MADDRx) according to the transfer direction and store the read data into the destination address space.

- Calculate the number of outstanding operations, perform a decrement operation of the DMA_TXNUMx register, and update the source and destination addresses of the next operation.

7.4.2 Channel Priority and Arbitration

The DMA uses an arbitration strategy to handle multiple requests from different channels. The priority of each channel is programmable in the channel control register (DMA_CHCFGx).

4 levels of priority:

- Very high priority
- High priority
- Medium priority
- Low priority

By default, channel with lower index has higher priority if the programmed priority is the same.

For memory to memory transfer, re-arbitration is carried on after 4 transfer operations.

For transfer related to periphery, re-arbitration is carried on after each transfer operation.

7.4.3 DMA Channels and Number Of Transfers

Each channel can perform DMA transfer between the peripheral register at the specified address and the memory address. The number of data transferred by DMA is programmable, and the maximum supported value is 65535. The DMA_TXNUM register is decremented after each transfer.

7.4.4 Programmable Data Bit Width, Alignment and Endians

Peripheral and memory transfer data bit width supports byte, half-word and word, which can be programmed through DMA_CHCFGx.PSIZE and DMA_CHCFGx.MSIZE.

When DMA_CHCFGx.PSIZE and DMA_CHCFGx.MSIZE are different, the DMA module aligns the data according to the below.

Table 7-1 Programmable Data Width and Endian Operation (When PINC = MINC = 1)

Source Width (bit)	Destination Width (bit)	Number of Transfer (byte)	Source: Address/Data	Transfer Operations (R: Read, W: Write)	Destination: Address/Data
8	8	4	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3	1: R B0 [7:0] @0x0, W B0 [7:0] @0x0 2: R B1 [7:0] @0x1, W B1 [7:0] @0x1 3: R B2 [7:0] @0x2, W B2 [7:0] @0x2 4: R B3 [7:0] @0x3, W B3 [7:0] @0x3	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3
8	16	4	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3	1: R B0 [7:0] @0x0, W 00B0 [15:0] @0x0 2: R B1 [7:0] @0x1, W 00B1 [15:0] @0x2 3: R B2 [7:0] @0x2, W 00B2 [15:0] @0x4 4: R B3 [7:0] @0x3, W 00B3 [15:0] @0x6	0x0 / 00B0 0x2 / 00B1 0x4 / 00B2 0x6 / 00B3

Source Width (bit)	Destination Width (bit)	Number of Transfer (byte)	Source: Address/Data	Transfer Operations (R: Read, W: Write)	Destination: Address/Data
8	32	4	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3	1: R B0 [7:0] @0x0, W 000000B0 [31:0] @0x0 2: R B1 [7:0] @0x1, W 000000B1 [31:0] @0x4 3: R B2 [7:0] @0x2, W 000000B2 [31:0] @0x8 4: R B3 [7:0] @0x3, W 000000B3 [31:0] @0xC	0x0 / 000000B0 0x4 / 000000B1 0x8 / 000000B2 0xC / 000000B3
16	8	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: R B1B0 [15:0] @0x0, W B0 [7:0] @0x0 2: R B3B2 [15:0] @0x2, W B2 [7:0] @0x1 3: R B5B4 [15:0] @0x4, W B4 [7:0] @0x2 4: R B7B6 [15:0] @0x6, W B6 [7:0] @0x3	0x0 / B0 0x1 / B2 0x2 / B4 0x3 / B6
16	16	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: R B1B0 [15:0] @0x0, W B1B0 [15:0] @0x0 2: R B3B2 [15:0] @0x2, W B3B2 [15:0] @0x2 3: R B5B4 [15:0] @0x4, W B5B4 [15:0] @0x4 4: R B7B6 [15:0] @0x6, W B7B6 [15:0] @0x6	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6
16	32	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: R B1B0 [15:0] @0x0, W 0000B1B0 [31:0] @0x0 2: R B3B2 [15:0] @0x2, W 0000B3B2 [31:0] @0x4 3: R B5B4 [15:0] @0x4, W 0000B5B4 [31:0] @0x8 4: R B7B6 [15:0] @0x6, W 0000B7B6 [31:0] @0xC	0x0 / 0000B1B0 0x4 / 0000B3B2 0x8 / 0000B5B4 0xC / 0000B7B6
32	8	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC	1: R B3B2B1B0 [31:0] @0x0, W B0 [7:0] @0x0 2: R B7B6B5B4 [31:0] @0x4, W B4 [7:0] @0x1 3: R BBBAB9B8 [31:0] @0x8, W B8 [7:0] @0x2 4: R BFBEBDBC [31:0] @0xC, W BC [7:0] @0x3	0x0 / B0 0x1 / B4 0x2 / B8 0x3 / BC
32	16	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC	1: R B3B2B1B0 [31:0] @0x0, W B1B0 [15:0] @0x0 2: R B7B6B5B4 [31:0] @0x4, W B5B4 [15:0] @0x2 3: R BBBAB9B8 [31:0] @0x8, W B9B8 [15:0] @0x4 4: R BFBEBDBC [31:0] @0xC, W BDBC [15:0] @0x6	0x0 / B1B0 0x2 / B5B4 0x4 / B9B8 0x6 / BDBC
32	32	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC	1: R B3B2B1B0 [31:0] @0x0, W B3B2B1B0 [31:0] @0x0 2: R B7B6B5B4 [31:0] @0x4, W B7B6B5B4 [31:0] @0x4 3: R BBBAB9B8 [31:0] @0x8, W BBBAB9B8 [31:0] @0x8 4: R BFBEBDBC [31:0] @0xC, W BFBEBDBC [31:0] @0xC	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC

Notice:

DMA always provide full 32-bits data to `HWDATA[31:0]` no matter what destination size it is (`HSIZE` still follows destination size setting for device supports byte/half-word operation). The `HWDATA[31:0]` follows the following rules:

- When source size is smaller than destination size, DMA pads the MSB with 0 until their sizes match and duplicates it to be 32 bits. E.g., source is 8 bits data `0x55` and destination size is 16 bits. DMA fills the source data with 0 to make it 16 bits and become `0x0055`, then duplicate it to 32-bit data `0x0055_0055` and provide to `HWDATA[31:0]`; (if destination size is 32-bit then DMA will only pad source data with 0).
- When source size is larger or equal to destination size and smaller than 32 bits, DMA duplicates source data to 32 bits data. E.g., source data is 8 bits data `0x1F`, `HWDATA[31:0] = 0x1F1F_1F1F`. if source data is 16 bits data `0x2345`, then `HWDATA[31:0] = 0x2345_2345`.

This ensures peripherals that only support word operation won't generate bus error and the desired data can still move to the place we want with extra bits i.e. 0 padding. If user wants to configure an 8-bit register but is aligned to a 32-bit address boundary, the source size should be set to 8 bits and destination to 32 bits so extra bits will be padded with 0.

7.4.5 Peripheral/Memory Address Incrementation

DMA_CHCFGx.PINC and DMA_CHCFGx.MINC respectively control whether the peripheral address and memory address are enabled in auto-increment mode. The software cannot write (can read) the address register during transfer.

- In auto-increment mode, the next address to be transferred is automatically increased according to the data bit width (1, 2 or 4) after each transfer. The address of the first transfer is stored in DMA_PADDRx or DMA_MADDRx register.
- In fixed mode, the address is always fixed to the initial address.

At the end of transfer (i.e. the transfer count changes to 0), different processes will be carried out according to whether the current operation is under circular mode or not.

- In non-circular mode, DMA stops after the transfer is completed. To start a new DMA transfer, need to rewrite the transfer number in the DMA_TXNUMx register with the DMA channel disabled.
- In circular mode, at the end of a transfer, the content of the DMA_TXNUMx register will be automatically reloaded to its initial value, and the current internal peripheral or memory address register will also be reloaded to the initial base address set by the DMA_PADDRx or DMA_MADDRx register.

7.4.6 Channel Configuration Process

The detail configuration process is as below:

1. Configure interrupt mask bits, 1: enable interrupts, 0 disable interrupts.
2. Configure channel peripheral address and memory address and transfer direction.
3. Configure channel priority, 0: lowest, 3: highest.
4. Configure peripheral and memory address increment.
5. Configure channel transfer block size.
6. If necessary, configure circular mode.
7. If it is memory to memory, configure MEM2MEM mode (Note: to configure DMA to work in M2M mode, user needs to set corresponding channel select value to reserved value, e.g., 47).
8. Repeat step 1~8 on channel 1~8 and finally.
9. Enable corresponding channel.

If software is used to serve interrupt, the software must enquire interrupt status register to check which interrupt occurred (software needs to write 1 to interrupt flag clear bit to clear the corresponding interrupt). Before enable channel, all interrupts corresponding to the channel should be cleared.

If the interrupt is transfer complete interrupt, software can configure the next transfer, or report to user this channel transformation is done.

Note: DMA user privilege management only supports that the user of the DMA configuration register is the same user as the DMA-enabled user, otherwise it will cause DMA transfer errors to occur.

7.4.7 Flow Control

Three major flow controls are supported:

- Memory to memory
- Memory to peripheral
- Peripheral to memory

Flow control is controlled by two register bits in each DMA channel configuration register. Flow control is used to control source/destination and direction of DMA channel.

Table 7-2 Flow Control Table

DMA_CHCFGx.MEM2MEM	DMA_CHCFGx.DIR	Source	Destination	Transfer
1	x	Memory	Memory	AHB read to AHB write, can do back2back transfer
0	1	Memory	AHB Peripheral	AHB read to AHB write, single transfer
			APB Peripheral	AHB read to APB write, single transfer
0	0	AHB Peripheral	Memory	AHB read to AHB write, single transfer
		APB Peripheral		APB read to AHB write, single transfer

7.4.8 Circular Mode

The circular mode is used to process circular buffers and continuous data transmission (such as ADC scan mode). The DMA_CHCFGx.CIRC is used to enable this function. When the circular mode is activated, if the number of data to be transferred becomes 0, it will automatically be restored to the initial value when configuring the channel, and the DMA operation will continue.

If the user wants to turn off the circular mode, the user needs to write 0 to DMA_CHCFGx.CHEN to disable the DMA channel, and then write 0 to DMA_CHCFGx.CIRC (when DMA_CHCFGx.CHEN is 1, other bits in the DMA_CHCFGx register cannot be rewritten).

7.4.9 Error Management

DMA access to a reserved address space will cause DMA transmission errors. When an error occurs, the transfer error flag is set, and the hardware automatically clears the current DMA channel enable bit (DMA_CHCFGx.CHEN), and the channel operation is stopped. If the transfer error interrupt enable bit is set in the DMA_CHCFGx register, an interrupt will be

generated.

7.4.10 Interrupt

- Transfer complete interrupt:

An interrupt is generated when channel data transfer is completed. Interrupt is a level signal. Each channel has its dedicated interrupt, interrupt mask control and interrupt status bit. interrupt status bit is cleared when interrupt flag clear bit is set.

- Half transfer interrupt:

An interrupt is generated when half of the channel data is transferred. Interrupt is a level signal. Each channel has its dedicated interrupt, interrupt mask control and interrupt status bit. interrupt status bit is cleared when interrupt flag clear bit is set.

- Transfer error interrupt:

An interrupt is generated when bus returned error. Interrupt is a level signal. Each channel has its dedicated interrupt, interrupt mask control and interrupt status bit. interrupt status bit is cleared when interrupt flag clear bit is set.

Table 7-3 DMA Interrupt Request

Interrupt Event	Event Flag Bit	Enable Control Bit
Half transfer	HTXF	HTXIE
Transfer complete	TXCF	TXCIE
Transfer error	ERRF	ERRIE

7.4.11 DMA Request Mapping

Totally there are 35 DMA requests from all the peripherals. To have better support with full flexibility, register bits can be used to select which DMA request is mapped to which DMA channel. The table below shows the mapping scheme of peripherals' DMA request to DMA controller's DMA channels.

Table 7-4 Dma Request Mapping

DMA Channel Select	Peripheral DMA Request	DMA Channel Select	Peripheral DMA Request
sel = 0	adc_dma	sel = 24	Tim1_ch2
sel = 1	Usart1_tx	sel = 25	Tim1_ch3
sel = 2	Usart1_rx	sel = 26	Tim1_ch4
sel = 3	Usart2_tx	sel = 27	Tim1_com
sel = 4	Usart2_rx	sel = 28	Tim1_up
sel = 5	Lpuart1_tx	sel = 29	Tim1_trig
sel = 6	Lpuart1_rx	sel = 30	Tim8_ch1
sel = 7	Lpuart2_tx	sel = 31	Tim8_ch2
sel = 8	Lpuart2_rx	sel = 32	Tim8_ch3
sel = 9	Usart5_tx	sel = 33	Tim8_ch4
sel = 10	Usart5_rx	sel = 34	Tim8_com
sel = 11	Usart6_tx	sel = 35	Tim8_up
sel = 12	Usart6_rx	sel = 36	Tim8_trig
sel = 13	Spi1_tx	sel = 37	Tim3_ch1
sel = 14	Spi1_rx	sel = 38	Tim3_ch3
sel = 15	Spi2_tx	sel = 39	Tim3_ch4
sel = 16	Spi2_rx	sel = 40	Tim3_up
sel = 17	Spi3_tx	sel = 41	Tim3_trig
sel = 18	Spi3_rx	sel = 42	Tim4_ch1
sel = 19	I2c1_tx	sel = 43	Tim4_ch2
sel = 20	I2c1_rx	sel = 44	Tim4_ch3
sel = 21	I2c2_tx	sel = 45	Tim4_up
sel = 22	I2c2_rx	sel = 46	TIM6
sel = 23	Tim1_ch1		

7.5 DMA Registers

7.5.1 DMA Register Overview

Table 7-5 DMA Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
000h	DMA_INTSTS	ERRF8	HTXF8	TXCF8	GLBF8	ERRF7	HTXF7	TXCF7	GLBF7	ERRF6	HTXF6	TXCF6	GLBF6	ERRF5	HTXF5	TXCF5	GLBF5	ERRF4	HTXF4	TXCF4	GLBF4	ERRF3	HTXF3	TXCF3	GLBF3	ERRF2	HTXF2	TXCF2	GLBF2	ERRF1	HTXF1	TXCF1	GLBF1								
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								
004h	DMA_INTCLR	CERRE8	CHTXF8	CTXCF8	CGLBF8	CERRF7	CHTXF7	CTXCF7	CGLBF7	CERRF6	CHTXF6	CTXCF6	CGLBF6	CERRF5	CHTXF5	CTXCF5	CGLBF5	CERRF4	CHTXF4	CTXCF4	CGLBF4	CERRF3	CHTXF3	CTXCF3	CGLBF3	CERRF2	CHTXF2	CTXCF2	CGLBF2	CERRF1	CHTXF1	CTXCF1	CGLBF1								
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								
008h	DMA_CHCFG1	Reserved																		MEM2MEM	PRIORVL[1:0]	MSIZE[1:0]	PSIZE[1:0]	MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN										
	Reset Value	0																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00Ch	DMA_TXNUM1	Reserved																		NDTX[15:0]																					
	Reset Value	0																		0																					
010h	DMA_PADDR1	ADDR[31:0]																																							
	Reset Value	0																																							
014h	DMA_MADDR1	ADDR[31:0]																																							
	Reset Value	0																																							
018h	DMA_CHSEL1	Reserved																								CH_SEL[5:0]															

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0													
01Ch	DMA_CHCFG2	Reserved														MEM2MEM	PRIOLVL[1:0]	MSIZE[1:0]	PSIZE[1:0]	MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN	0	0	0	0	0	0													
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
020h	DMA_TXNUM2	Reserved														NDTX[15:0]												0																		
024h	DMA_PADDR2	ADDR[31:0]																																												
	Reset Value	0																																												
028h	DMA_MADDR2	ADDR[31:0]																																												
	Reset Value	0																																												
02Ch	DMA_CHSEL2	Reserved														CH_SEL[5:0]												0																		
	Reset Value	0														0												0																		
030h	DMA_CHCFG3	Reserved														MEM2MEM	PRIOLVL[1:0]	MSIZE[1:0]	PSIZE[1:0]	MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN	0	0	0	0	0	0													
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
034h	DMA_TXNUM3	Reserved														NDTX[15:0]												0																		
038h	DMA_PADDR3	ADDR[31:0]																																												
	Reset Value	0																																												
03Ch	DMA_MADDR3	ADDR[31:0]																																												
	Reset Value	0																																												
040h	DMA_CHSEL3	Reserved														CH_SEL[5:0]												0																		
	Reset Value	0														0												0																		
044h	DMA_CHCFG4	Reserved														MEM2MEM	PRIOLVL[1:0]	MSIZE[1:0]	PSIZE[1:0]	MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN	0	0	0	0	0	0													
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
048h	DMA_TXNUM4	Reserved														NDTX[15:0]												0																		
04Ch	DMA_PADDR4	ADDR[31:0]																																												
	Reset Value	0																																												
050h	DMA_MADDR4	ADDR[31:0]																																												
	Reset Value	0																																												
054h	DMA_CHSEL4	Reserved														CH_SEL[5:0]												0																		
	Reset Value	0														0												0																		
058h	DMA_CHCFG5	Reserved														MEM2MEM	PRIOLVL[1:0]	MSIZE[1:0]	PSIZE[1:0]	MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN	0	0	0	0	0	0													
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
05Ch	DMA_TXNUM5	Reserved														NDTX[15:0]												0																		
060h	DMA_PADDR5	ADDR[31:0]																																												
	Reset Value	0																																												
064h	DMA_MADDR5	ADDR[31:0]																																												
	Reset Value	0																																												
068h	DMA_CHSEL5	Reserved														CH_SEL[5:0]												0																		
	Reset Value	0														0												0																		
06Ch	DMA_CHCFG6	Reserved														MEM2MEM	PRIOLVL[1:0]	MSIZE[1:0]	PSIZE[1:0]	MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN	0	0	0	0	0	0													
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
070h	DMA_TXNUM6	Reserved														NDTX[15:0]												0																		
074h	DMA_PADDR6	ADDR[31:0]																																												
	Reset Value	0																																												
078h	DMA_MADDR6	ADDR[31:0]																																												
	Reset Value	0																																												
07Ch	DMA_CHSEL6	Reserved														CH_SEL[5:0]												0																		
	Reset Value	0														0												0																		
080h	DMA_CHCFG7	Reserved														MEM2MEM	PRIOLVL[1:0]	MSIZE[1:0]	PSIZE[1:0]	MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN	0	0	0	0	0	0													
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
084h	DMA_TXNUM7	Reserved														NDTX[15:0]												0																		
088h	DMA_PADDR7	ADDR[31:0]																																												
	Reset Value	0																																												
08Ch	DMA_MADDR7	ADDR[31:0]																																												
	Reset Value	0																																												
090h	DMA_CHSEL7	Reserved														CH_SEL[5:0]												0																		
	Reset Value	0														0												0																		

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0														
094h	DMA_CHCFG8	Reserved														MEM2MEM	PRIOLVL[1:]	0	MSIZE[1:0]	PSIZE[1:0]	MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN	0																		
	Reset Value															0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
098h	DMA_TXNUM8	Reserved														NDTX[15:0]																															
	Reset Value															0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
09Ch	DMA_PADDR8	ADDR[31:0]																																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0											
0A0h	DMA_MADDR8	ADDR[31:0]																																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0										
0A4h	DMA_CHSEL8	Reserved																								CH_SEL[5:0]																					
	Reset Value																									0	0	0	0	0	0																
	Reset Value																																														

7.5.2 DMA Interrupt Status Register (DMA_INTSTS)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ERRF8	HTXF8	TXCF8	GLBF8	ERRF7	HTXF7	TXCF7	GLBF7	ERRF6	HTXF6	TXCF6	GLBF6	ERRF5	HTXF5	TXCF5	GLBF5
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
ERRF4	HTXF4	TXCF4	GLBF4	ERRF3	HTXF3	TXCF3	GLBF3	ERRF2	HTXF2	TXCF2	GLBF2	ERRF1	HTXF1	TXCF1	GLBF1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit Field	Name	Description
31/27/23/19/15/11/7/3	ERRF _x	Transfer error flag for channel x (x=1...8). Hardware sets this bit when transfer error happen. This bit is cleared by software by writing '1' to DMA_INTCLR.CERRF _x bit. 0: Transfer error no happened on channel x. 1: Transfer error happened on channel x.
30/26/22/18/14/10/6/2	HTXF _x	Half transfer flag for channel x (x=1...8). Hardware sets this bit when half transfer is done. This bit is cleared by software by writing '1' to DMA_INTCLR.CHTXF _x bit. 0: Half transfer not yet done on channel x. 1: Half transfer was done on channel x.
29/25/21/17/13/9/5/1	TXCF _x	Transfer complete flag for channel x (x=1...8). Hardware sets this bit when transfer is done. This bit is cleared by software by writing '1' to DMA_INTCLR.CTXCF _x bit. 0: Transfer not yet done on channel x. 1: Transfer was done on channel x.
28/24/20/16/12/8/4/0	GLBF _x	Global flag for channel x (x=1...8). Hardware sets this bit when any interrupt events happen in this channel. This bit is cleared by software by writing '1' to DMA_INTCLR.CGLBF _x bit. 0: No transfer error, half transfer or transfer done event happen on channel x. 1: One of transfer error, half transfer or transfer done event happen on channel x.

7.5.3 DMA Interrupt Flag Clear Register (DMA_INTCLR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CERRF8	CHTXF8	CTXCF8	CGLBF8	CERRF7	CHTXF7	CTXCF7	CGLBF7	CERRF6	CHTXF6	CTXCF6	CGLBF6	CERRF5	CHTXF5	CTXCF5	CGLBF5
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
CERRF4	CHTXF4	CTXCF4	CGLBF4	CERRF3	CHTXF3	CTXCF3	CGLBF3	CERRF2	CHTXF2	CTXCF2	CGLBF2	CERRF1	CHTXF1	CTXCF1	CGLBF1
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit Field	Name	Description
31/27/23/19/15/11/7/3	CERRF _x	Clear transfer error flag for channel x (x=1...8). Software can set this bit to clear ERRF of corresponding channel. 0: No action. 1: Reset DMA_INTSTS.ERRF bit of corresponding channel.
30/26/22/18/14/10/6/2	CHTXF _x	Clear half transfer flag for channel x (x=1...8). Software can set this bit to clear HTXF of corresponding channel. 0: No action. 1: Reset DMA_INTSTS.HTXF bit of corresponding channel.
29/25/21/17/13/9/5/1	CTXCF _x	Clear transfer complete flag for channel x (x=1...8). Software can set this bit to clear TXCF of corresponding channel. 0: No action. 1: Reset DMA_INTSTS.TXCF bit of corresponding channel.
28/24/20/16/12/8/4/0	CGLBF _x	Clear global event flag for channel x (x=1...8). Software can set this bit to clear GLBF of corresponding channel. 0: No action. 1: Reset DMA_INTSTS.GLBF bit of corresponding channel.

7.5.4 DMA Channel X Configuration Register (DMA_CHCFG_x)

Note: The x is channel number, x = 1...8

Address offset: 0x08+20 * (x-1)

Reset value: 0x0000 0000

31	Reserved													16	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	MEM2 MEM	PRIOLVL[1:0]	MSIZE[1:0]	PSIZE[1:0]	MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN			
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit Field	Name	Description
31:15	Reserved	Reserved, the reset value must be maintained.
14	MEM2MEM	Memory to memory mode. Software can configure this channel to memory to memory transfer when it is not yet enabled. 0: Channel transfer between memory and peripheral.

Bit Field	Name	Description
		1: Channel set to memory to memory transfer.
13:12	PRIOLVL[1:0]	Channel priority. Software can program channel priority when channel is not enable. 00: Low 01: Medium 10: High 11: Very high
11:10	MSIZE[1:0]	Memory data size. Software can configure data size read/write from/to memory address. 00: 8-bits 01: 16-bits 10: 32-bits 11: Reserved
9:8	PSIZE[1:0]	Peripheral data size. Software can configure data size read/write from/to peripheral address. 00: 8-bits 01: 16-bits 10: 32-bits 11: Reserved
7	MINC	Memory increment mode. Software can enable/disable memory address increment mode. 0: Memory address won't increase with each transfer. 1: Memory address increase with each transfer.
6	PINC	Peripheral increment mode. Software can enable/disable peripheral address increment mode. 0: Peripheral address won't increase with each transfer. 1: Peripheral address increase with each transfer.
5	CIRC	Circular mode. Software can set/clear this bit. 0: Channel will stop after one round of transfer. 1: Channel configure as circular mode.
4	DIR	Data transfer direction Software can set/clear this bit. 0: Data transfer from Peripheral to Memory 1: Data transfer from Memory to Peripheral.
3	ERRIE	Transfer error interrupt enable. Software can enable/disable transfer error interrupt. 0: Disable transfer error interrupt of channel x. 1: Enable transfer error interrupt of channel x.
2	HTXIE	Half transfer interrupt enable. Software can enable/disable half transfer interrupt. 0: Disable half transfer interrupt of channel x. 1: Enable half transfer interrupt of channel x.

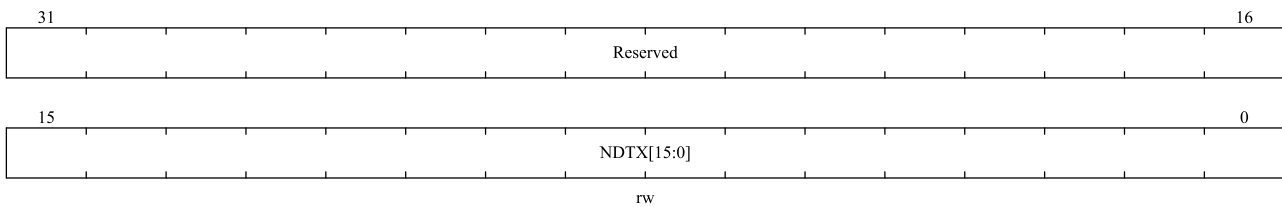
Bit Field	Name	Description
1	TXCIE	Transfer complete interrupt enable. Software can enable/disable transfer complete interrupt. 0: Disable transfer complete interrupt of channel x. 1: Enable transfer complete interrupt of channel x.
0	CHEN	Channel enable. Software can set/reset this bit. 0: Disable channel. 1: Enable channel.

7.5.5 DMA Channel X Transfer Number Register (DMA_TXNUMx)

Note: The x is channel number, x = 1...8

Address offset: 0x0C+20 * (x-1)

Reset value: 0x0000 0000



Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained.
15:0	NDTX	Number of data to transfer. Number of data to be transferred (0~65535). Software can read/write the number of transfers when channel is disable and it will be read only after channel enable. Every successful transfer of corresponding DMA channel will decrease this register by 1. If circular mode is enable, it will automatically reload pre-set value when it reach zero. Otherwise it will keep at zero and reset channel enable.

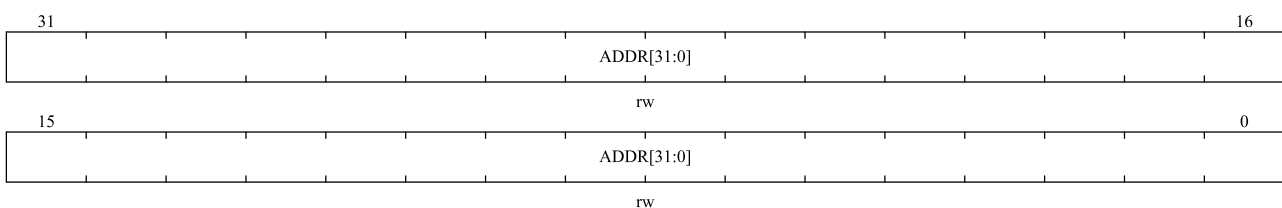
7.5.6 DMA Channel X Peripheral Address Register (DMA_PADDRx)

Note: The x is channel number, x = 1...8

Address offset: 0x10+20 * (x-1)

Reset value: 0x0000 0000

This register can only be written if the channel is disabled (DMA_CHCFGx.CHEN = 0).



Bit Field	Name	Description
31:0	ADDR	Peripheral address. Peripheral starting address for DMA to read/write from/to. Increment of address will be decided by DMA_CHCFGx.PSIZE. With DMA_CHCFGx.PSIZE equal to 01, DMA ignores bit 0 of PADDR and if DMA_CHCFGx.PSIZE equal to 10 DMA will ignore bit [1:0] of PADDR.

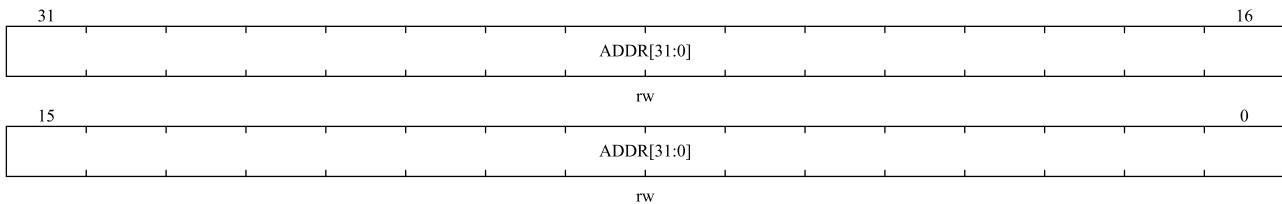
7.5.7 DMA Channel X Memory Address Register (DMA_MADDRx)

Note: The x is channel number, x = 1...8

Address offset: 0x14+20 * (x-1)

Reset value: 0x0000 0000

This register can only be written if the channel is disabled (DMA_CHCFGx.CHEN = 0).



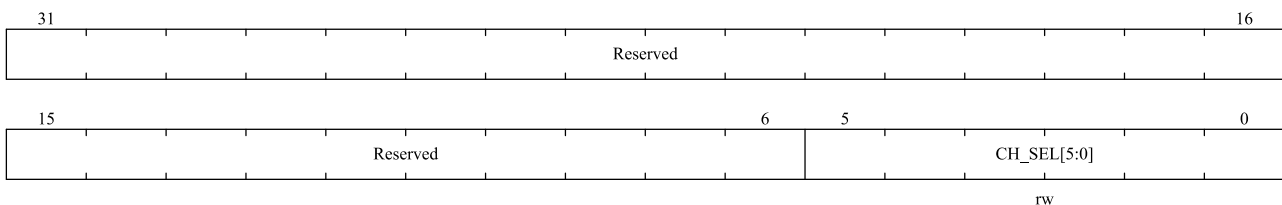
Bit Field	Name	Description
31:0	ADDR	ADDR Memory address. Memory starting address for DMA to read/write from/to. Increment of address will be decided by DMA_CHCFGx.MSIZE. With DMA_CHCFGx.MSIZE equal to 01, DMA ignores bit 0 of MADDR and if DMA_CHCFGx.MSIZE equal to 10 DMA will ignore bit [1:0] of MADDR.

7.5.8 DMA Channel X Channel Request Select Register (DMA_CHSELx)

Note: The x is channel number, x = 1...8

Address offset: 0x18+20 * (x-1)

Reset value: 0x0000 0000



Bit Field	Name	Description
31:6	Reserved	Reserved, the reset value must be maintained.
5:0	CH_SEL[5:0]	DMA channel request selection 0x00: adc_dma

Bit Field	Name	Description
		0x2E: TIM6 For the mapping of peripheral DMA requests to DMA input request channel numbers, please refer to Table 7-4

8 CRC Calculation Unit

8.1 CRC Introduction

This module integrates the functions of CRC32 and CRC16, and the cyclic redundancy check (CRC) calculation unit obtains any CRC calculation result according to a fixed generator polynomial. In other applications, CRC technology is mainly used to verify the correctness and integrity of data transmission or data storage. EN/IEC 60335-1 provides a method to verify the integrity of Flash memory. CRC calculation unit can calculate the signature of the software when the program is running, then compare it with the reference signature generated during connection, and then store it in the specified memory space.

8.2 CRC Main Features

8.2.1 CRC32 Module

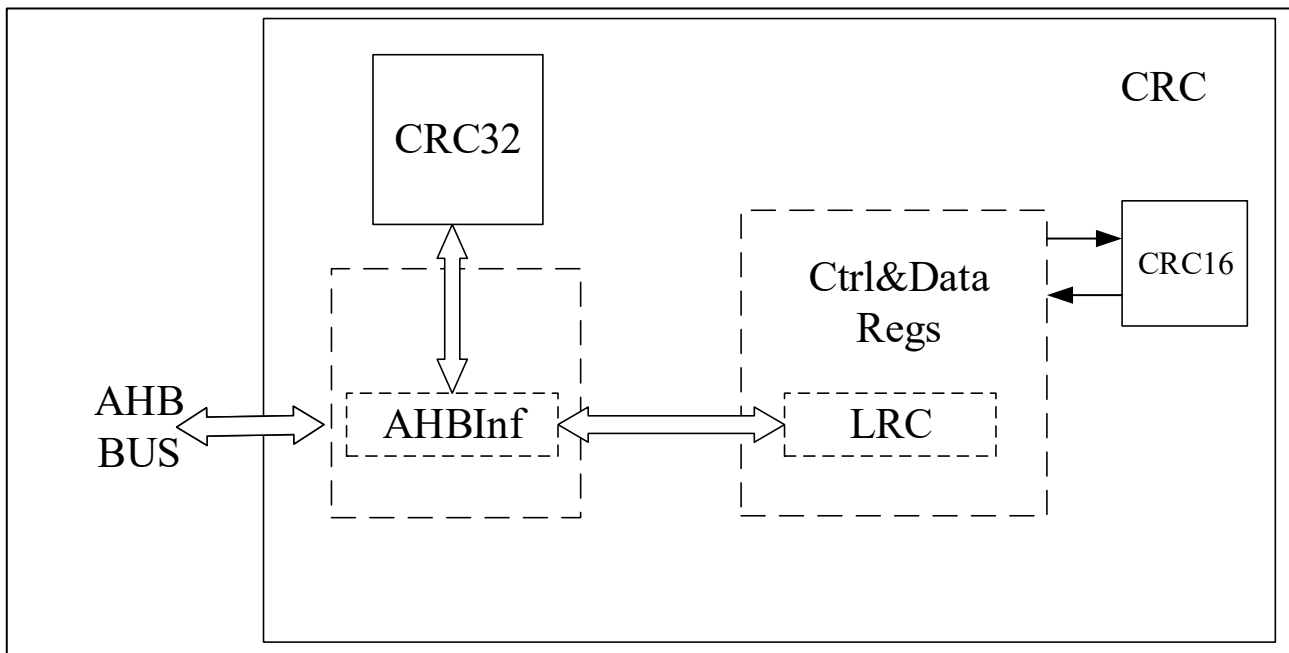
- $CRC32(X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X+1)$
- 32 bits of data to be checked and 32 bits of output check code.
- CRC calculation time: 1 AHB clock cycles (HCLK)
- General-purpose 8-bit register (can be used to store temporary data)
- Checksum initial value: 0xFFFFFFFF
- resulting XOR value: 0
- Input data is not inverted
- Output data is not inverted

8.2.2 CRC16 Module

- $CRC16(X^{16}+X^{15}+X^2+1)$
- There are 8 bits of data to be checked and 16 bits of output check code.
- CRC calculation time: 1 AHB clock cycle (HCLK)
- The verification initial value can be configured, and the size end of the data to be verified can be configured.
- Support 8bit LRC check value generation

The following figure is the block diagram of CRC unit.

Figure 8-1 CRC Calculation Unit Block Diagram



8.3 CRC Function Description

8.3.1 CRC32

CRC calculation unit contains one 32-bit data register:

- Writing this register to input CRC data.
- Reading this register to get the calculated CRC result.

Every writing operation to this data register triggers the calculation of this new data with the previous calculation result (CRC calculation is performed on the whole 32-bit word rather than byte by byte).

Supports back-to-back writes or sequential write-read operations.

CRC_CRC32DAT can be re-initialized to 0xFFFFFFFF by setting CRC_CRC32CTRL.RESET. This operation does not affect the data in register CRC_CRC32IDAT.

8.3.2 CRC16

CRC_CRC16CTRL.ENDHL controls Little Endian format or Big Endian format.

To clear the result of the last CRC operation, set CRC_CRC16CTRL.CLR to 1 or CRC_CRC16D to 0.

The initial value of CRC calculation can be configured by writing the CRC_CRC16D register. By default, the initial value is the result of the last calculation.

LRC calculation is the same as CRC calculation. Both are carried out at the same time. CRC or LRC can be read out depending on needs. If the initial value needs to be set, the LRC register should be configured first.

8.4 CRC Registers

8.4.1 CRC Register Overview

The following table lists the registers and reset values of CRC.

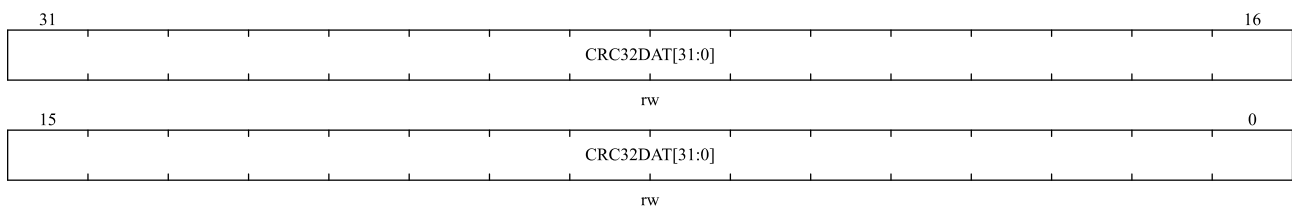
Table 8-1 CRC Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
000h	CRC32DAT	CRC32DAT[31:0]																																
	Reset Value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
004h	CRC32IDAT	Reserved																								CRC32IDAT[7:0]								
	Reset Value																									0	0	0	0	0	0	0	0	
008h	CRC32CTRL	Reserved																																
	Reset Value																																	
00Ch	CRC16CTRL	Reserved																												CLR	ENDHL	Reserved		
	Reset Value																													0	0			
010h	CRC16DAT	Reserved																								CRC16DAT[7:0]								
	Reset Value																									0	0	0	0	0	0	0	0	0
014h	CRC16D	Reserved														CRC16D[15:0]																		
	Reset Value															0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
018h	LRC	Reserved																								LRCDAT[7:0]								
	Reset Value																									0	0	0	0	0	0	0	0	0

8.4.2 CRC32 Data Register (CRC_CRC32DAT)

Address offset: 0x00

Reset value: 0xFFFF FFFF

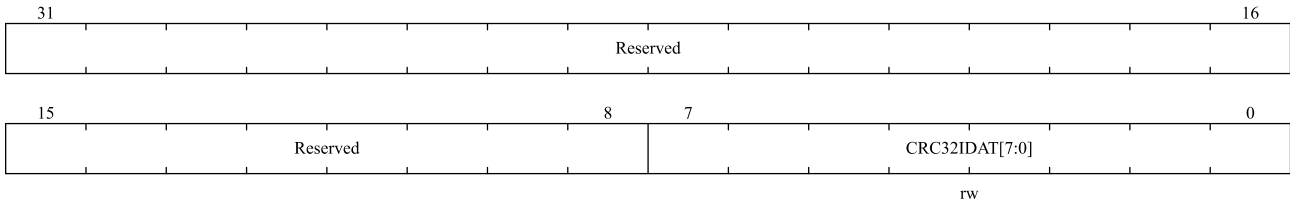


Bit Field	Name	Description
31:0	CRC32DAT[31:0]	CRC32 Data register. The written data is the CRC value to be checked. The read data is the CRC calculation result. Only 32-bit operations are supported.

8.4.3 CRC32 Independent Data Register (CRC_CRC32IDAT)

Address offset: 0x04

Reset value: 0x0000 0000



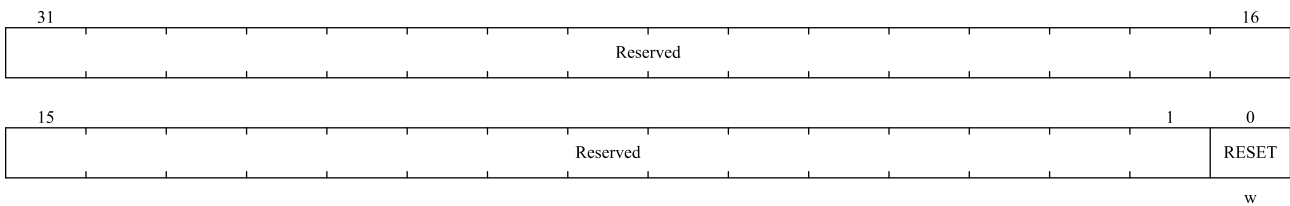
Bit Field	Name	Description
31:8	Reserved	Reserved,the reset value must be maintained
7:0	CRC32IDAT[7:0]	Independent 8-bit data register. General 8 bits data register. It is for temporary stored 1-byte data. CRC_ CRC32CTRL.RESET bit reset signal will not impact this register.

Note: This register is not a part of CRC calculation and can be used to store any data.

8.4.4 CRC32 Control Register (CRC_ CRC32CTRL)

Address offset: 0x08

Reset value: 0x0000 0000

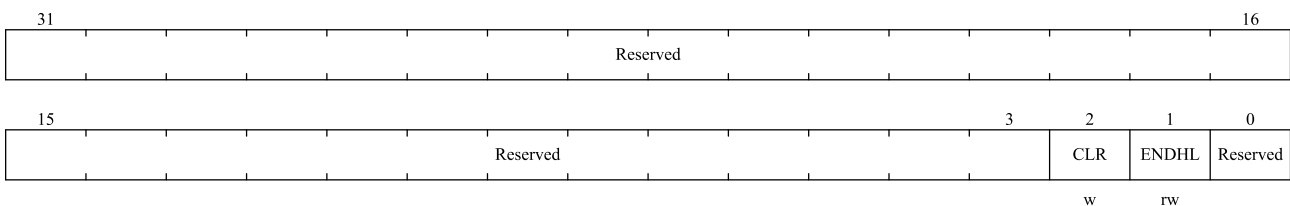


Bit Field	Name	Description
31:1	Reserved	Reserved,the reset value must be maintained
0	RESET	RESET signal. It can reset CRC32 module and set data register to be 0xFFFF_FFFF. This reset can only write 1, and hardware will clear to 0 automatically.

8.4.5 CRC16 Control Register (CRC_ CRC16CTRL)

Address offset: 0x0C

Reset value: 0x0000 0000



Bit Field	Name	Description
31:3	Reserved	Reserved,the reset value must be maintained

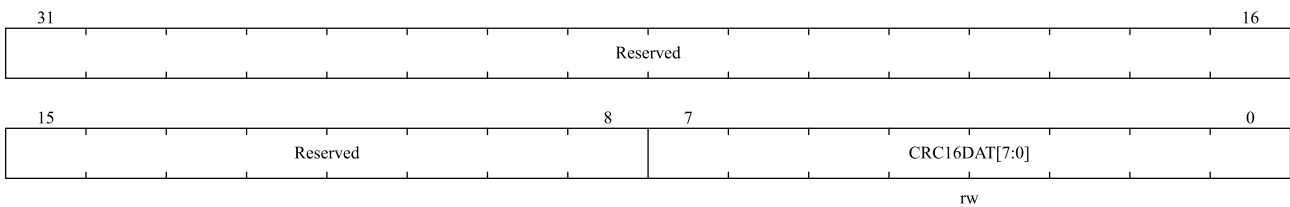
Bit Field	Name	Description
2	CLR	Clear CRC16 results. 0: Not clear. 1: Clear to default value 0x0000. Set this bit to 1 will only maintain 1 clock cycle, hardware will clear automatically. (Software read always 0).
1	ENDHL	Data to be verified start to calculate from MSB or LSB. 0: From MSB to LSB 1: From LSB to MSB This bit is only for data to be verified.
0	Reserved	Reserved,the reset value must be maintained

Note: 8-bits, 16-bits and 32-bits operations are supported.

8.4.6 CRC16 Input Data Register (CRC_CRC16DAT)

Address offset: 0x10

Reset value: 0x0000 0000



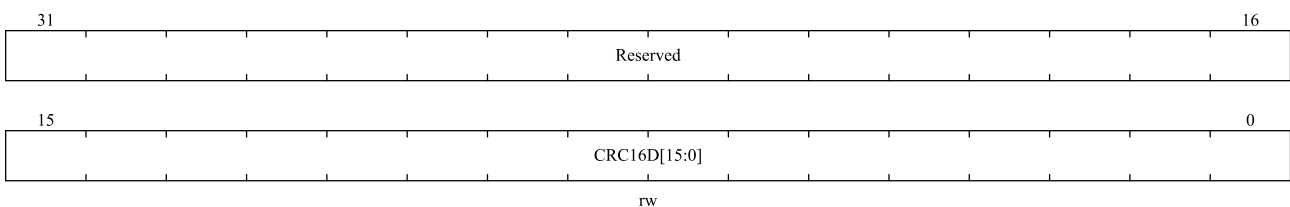
Bit Field	Name	Description
31:8	Reserved	Reserved,the reset value must be maintained
7:0	CRC16DAT[7:0]	Data to be verified.

Note: 8-bits, 16-bits and 32-bits operations are supported.

8.4.7 CRC Cyclic Redundancy Check Code Register (CRC_CRC16D)

Address offset: 0x14

Reset value: 0x0000 0000



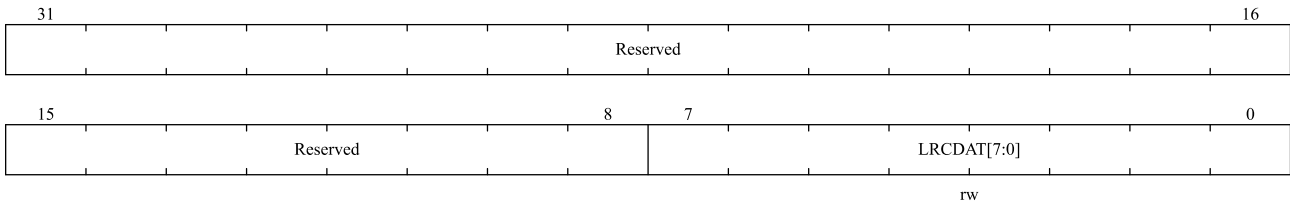
Bit Field	Name	Description
31:16	Reserved	Reserved,the reset value must be maintained
15:0	CRC16D[15:0]	16-bit value of cyclic redundancy result data. Every time the software writes the CRC16DAT register, the 16-bit calculated data from CRC16 is updated in this register.

Note: 8-bits, 16-bits and 32-bits operations are supported (8-bit operations must be performed twice in a row to ensure that 16-bit initial values are configured properly)

8.4.8 LRC Result Register (CRC_LRC)

Address offset: 0x18

Reset value: 0x0000 0000



Bit Field	Name	Description
31:8	Reserved	Reserved, the reset value must be maintained
7:0	LRCDAT[7:0]	LRC check value register. Software need to write initial value before use. And then each writing data to CRCDR will be XOR with LCR register value. The result will be stored in LRC. Software read the result. It should be cleared before next use.

9 Advanced-control Timers (TIM1 and TIM8)

9.1 TIM1 and TIM8 Introduction

The advanced control timers (TIM1 and TIM8) are mainly used in the following scenarios: counting the input signals, measuring the pulse lengths of the input signal and generating the output waveform.

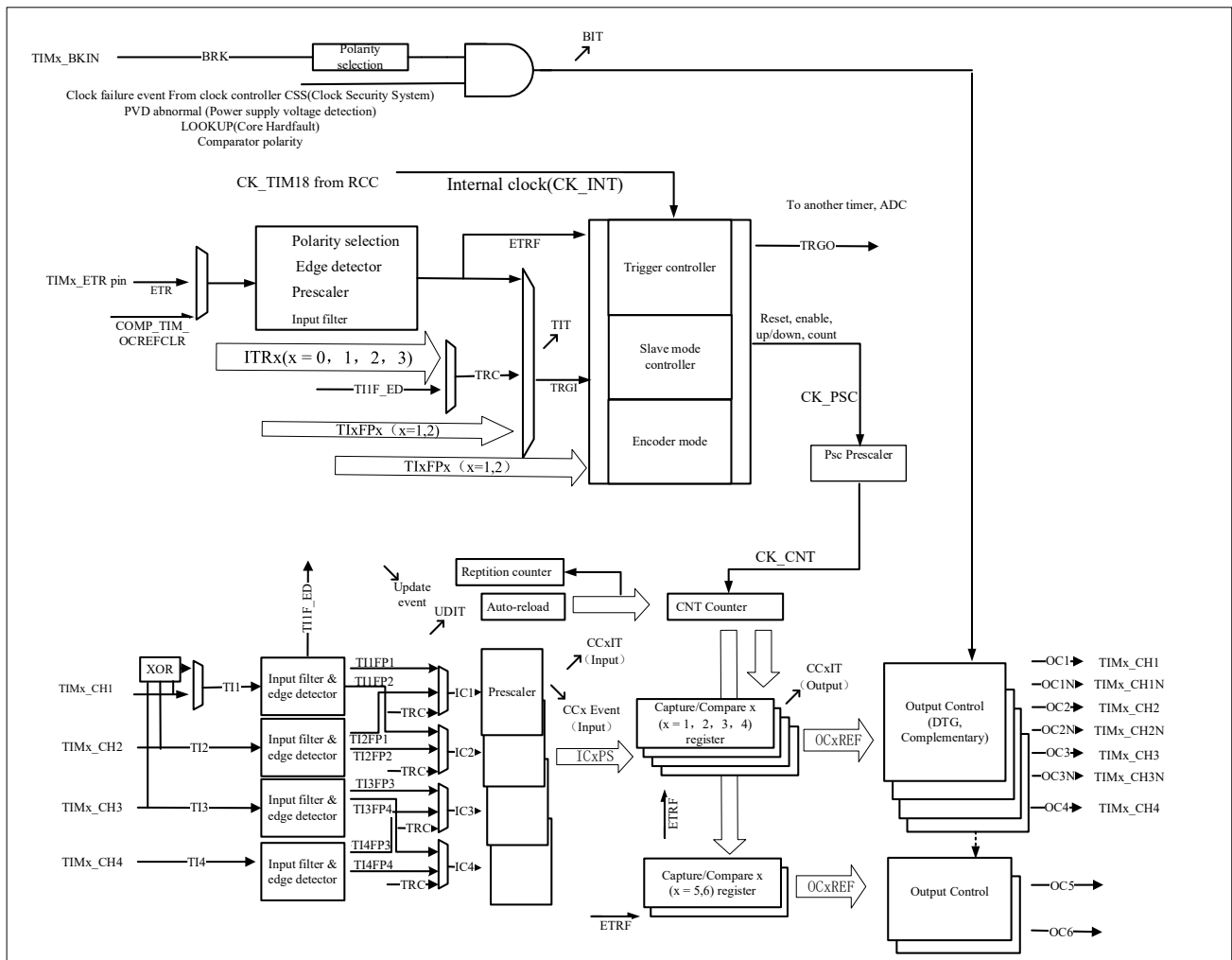
The advanced timer has complementary output functions, dead-time insertion and break function. They are suitable for motor control.



9.2 Main Features of TIM1 and TIM8

- 16-bit auto-reload counters. (It can realize up-counting, down-counting, up/down counting).
- 16-bit programmable prescaler. (The prescaler factor can be configured with any value between 1 and 65536)
- Programmable repetition counter
- Up to 4 capture/compare channels:
 - Input capture;
 - Output comparison;
 - PWM output
 - One-pulse mode output;
- PWM triggered ADC sampling:
- The trigger time point is software-configurable throughout the PWM cycle.
- Complementary outputs with programmable dead-time
 - For TIM1 and TIM8, channel 1,2,3 support this feature
- Timer can be controlled by external signal
- Timers can be linked together internally for timer synchronization or chaining
- Allows the repeat counter of the timer register to be updated after a specified number of counter cycles;
- The Break input signal can reset the timer output signal to either a reset state or a known state;
- An interrupt/DMA occurs when the following events occur:
 - Update event: Counter overflow up/overflow down, counter initialization (triggered by software or internal/external);
 - Trigger events (counters start, stop, initialize, or count internally/externally triggered);
 - Input capture;
 - Output comparison;
 - Break signal input;
- Incremental (quadrature) encoder interface: used for tracking motion and resolving rotation direction and position

- Hall sensor interface: used to do three-phase motor control
- The trigger input as an external clock or periodic current management

Figure 9-1 Block Diagram of TIM1 and TIM8



 *The event*
 *Interrupt and DMA output*
The capture channel 1 input can come from IOM or comparator output

9.3 TIM1 and TIM8 Function Description

9.3.1 Time-base Unit

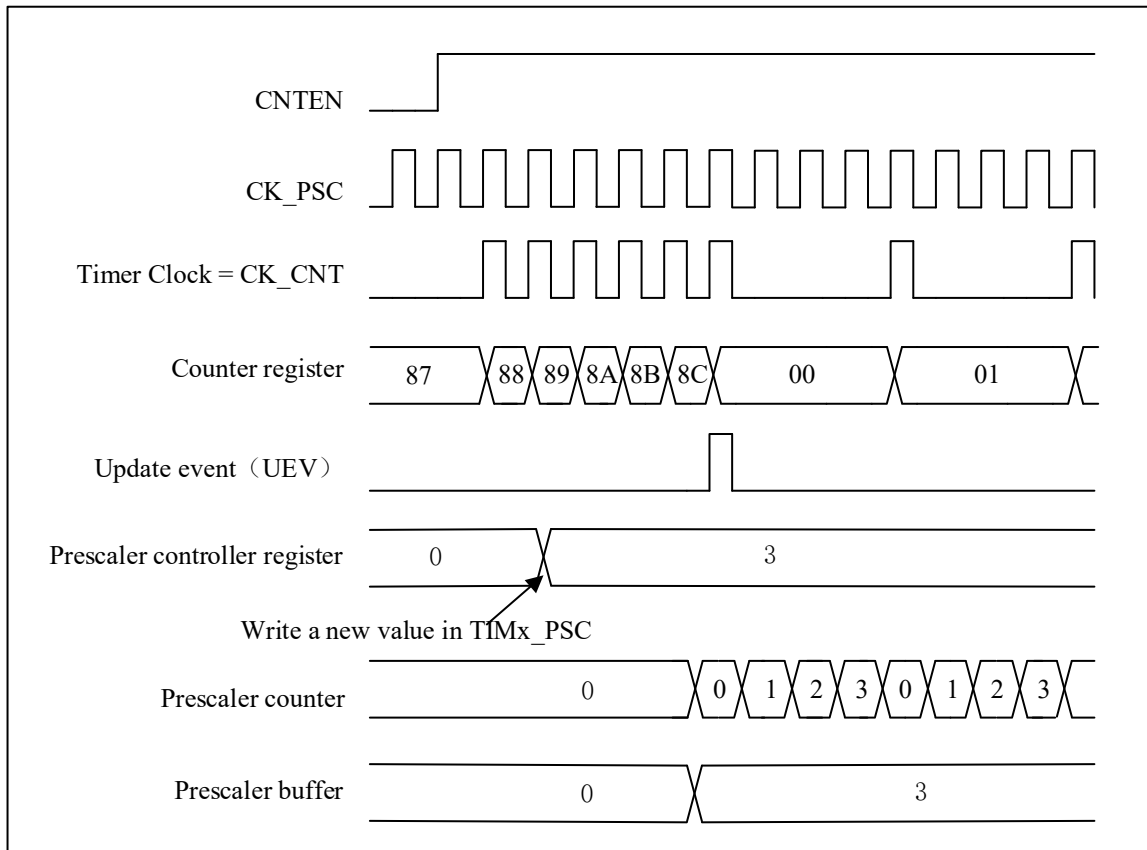
The time-base unit of advanced-control timers mainly includes: prescaler, counter, auto-reload and repetition counter. When the time base unit is operating, the software can read and write the corresponding registers (**TIMx_PSC**, **TIMx_CNT**, **TIMx_AR** and **TIMx_REPCNT**) at any time.

Depending on the setting of the auto-reload preload enable bit (**TIMx_CTRL1.ARPEN**), the content of the preload register is transferred to the shadow register immediately or at each update event UEV. When **TIMx_CTRL1.UPDIS=0**, a counter overflow/underflow or software setting **TIMx_EVTGEN.UDGN** will generate an update event. The counter starts counting one clock cycle after the **TIMx_CTRL1.CNTEN** bit is set.

Prescaler description

The TIMx_PSC register consists of a 16-bit counter that can be used to divide the counter clock frequency by any factor between 1 and 65536. It can be changed on the fly as it is buffered. The new prescaler ratio is taken into account at the next update event.

Figure 9-2 Counter Timing Diagram with Prescaler Division Change from 1 to 4



9.3.2 Counter Mode

Up-counting mode

In up-counting mode, the counter will count from 0 to the value of the register TIMx_AR, then restart from 0 and a counter overflow event is generated.

when the TIMx_CTRL1.UPRS bit (select update request) and the TIMx_EVTGEN.UDGN bit are set, the update event (UEV) will be generated, and TIMx_STS.UDITF not be set by hardware, therefore, no update interrupts or update DMA requests will be generated. This is to avoid generating the update interrupt when clearing the counter.

Depending on the TIMx_CTRL1.UPRS, when an update event occurs, all registers are updated and the TIMx_STS.UDITF is set:

- The repetition counter is reloaded with the content of the TIMx_REPCNT
- Update auto-reload shadow registers are updated with preload value(TIMx_AR), when TIMx_CTRL1.ARPEN = 1.
- The prescaler shadow register is reloaded with the preload value(TIMx_PSC).

Setting TIMx_CTRL1.UPDIS=1 is to avoid updating the shadow registers when new values are written to the preload registers.

When an update event is generated, the counter will still be cleared and the prescaler counter will also be set to 0 (but the prescaler value will remain unchanged).

The figure below shows some examples of the counter behavior for different prescaler factors in the up-counting mode.

Figure 9-3 Timing Diagram of Up-counting. The Internal Clock Divider Factor = 2/N

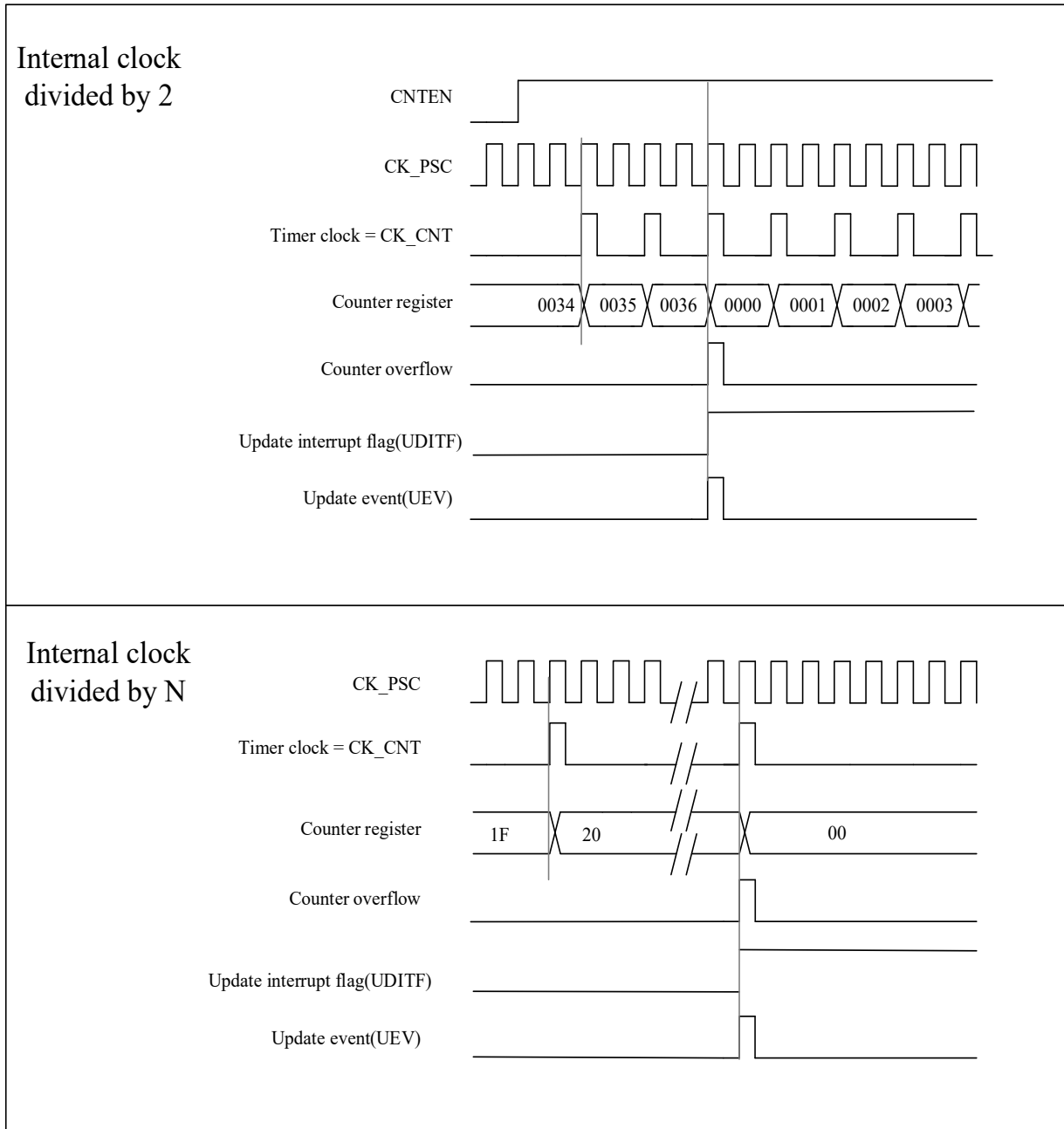
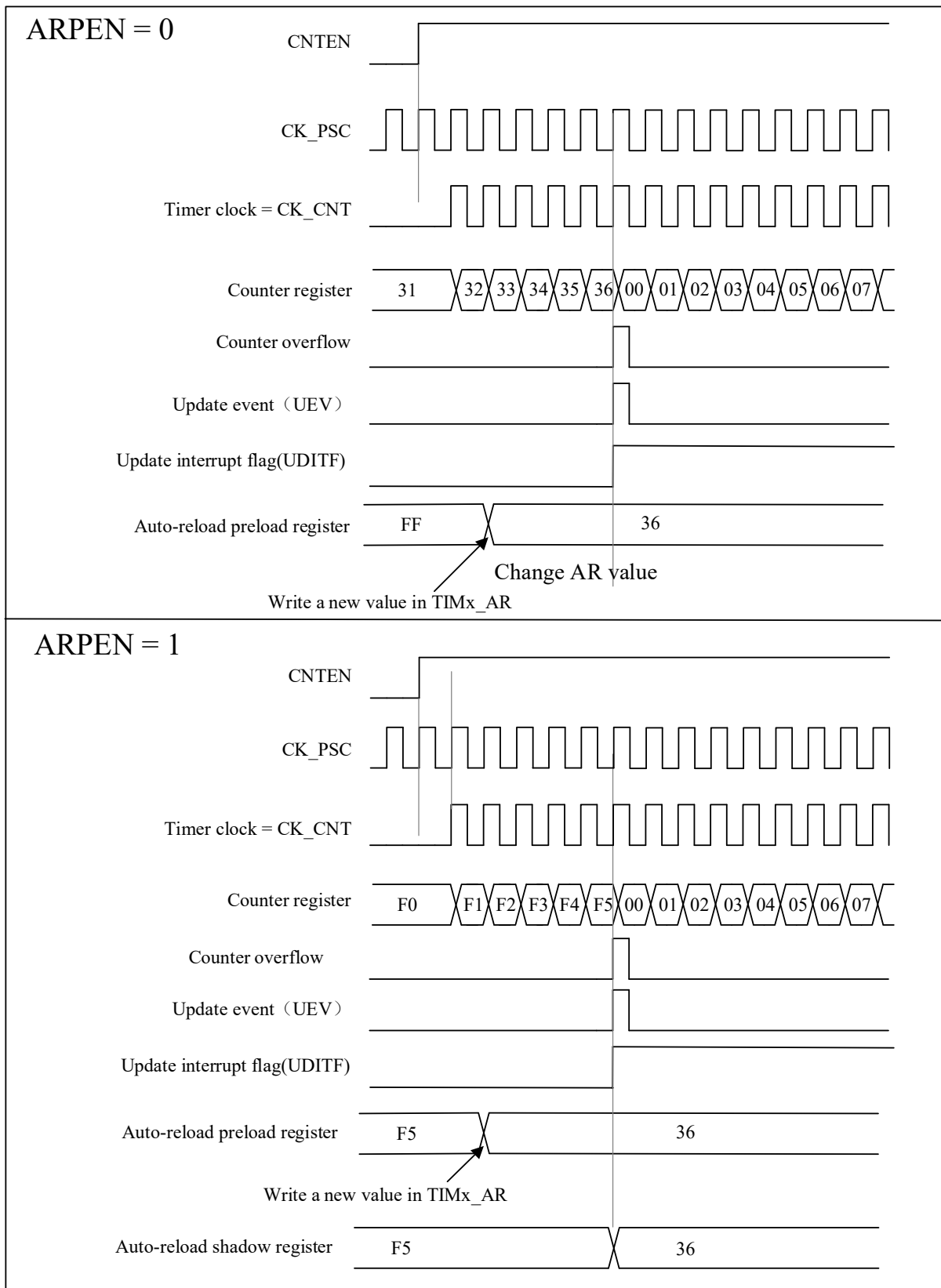


Figure 9-4 Timing Diagram of The Up-counting, Update Event When ARPEN=0/1



Down-counting mode

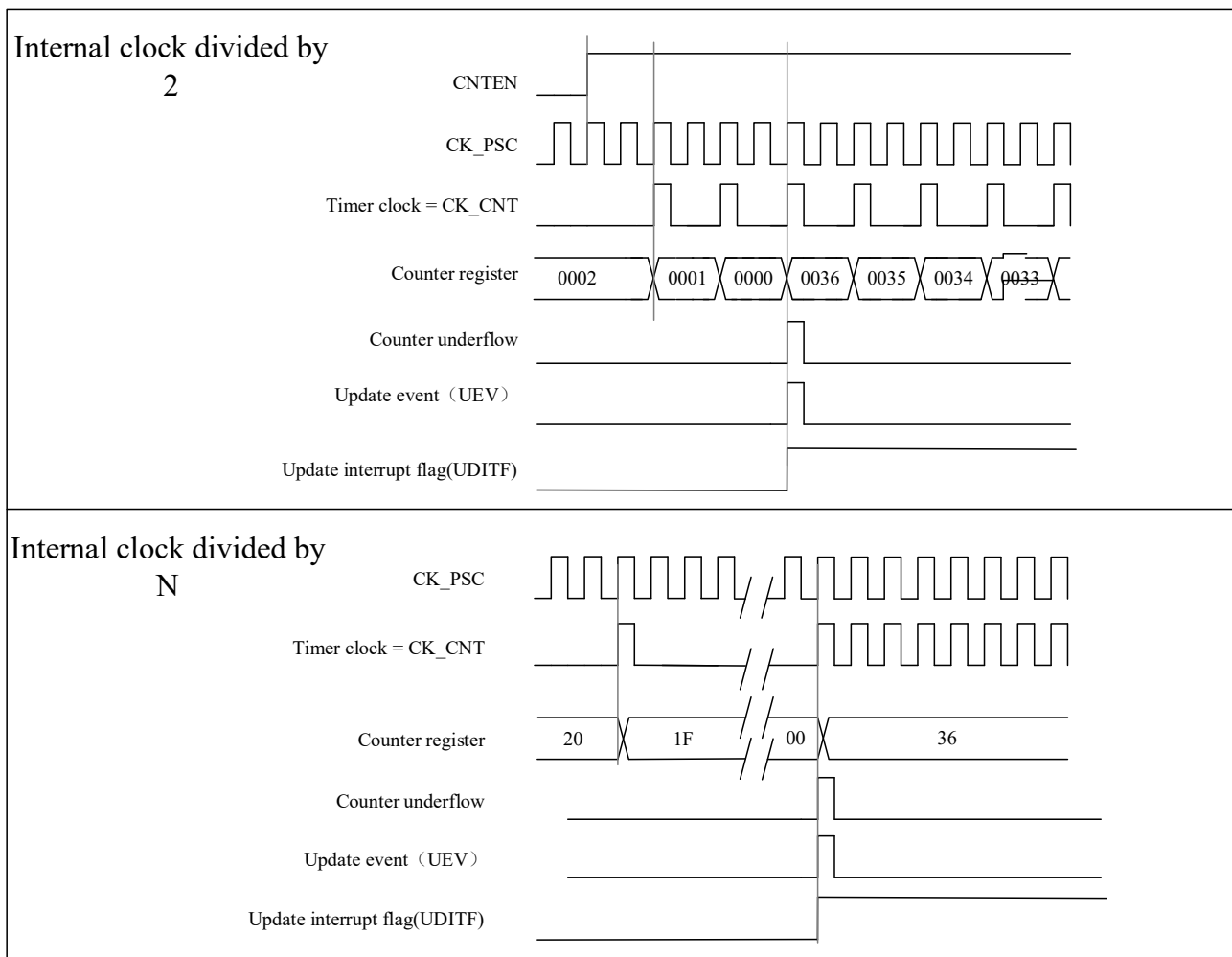
In down-counting mode, the counter will decrement from the value of the register TIMx_AR to 0, then restart from the

auto-reload value and generate a counter underflow event.

The process of configuring update events and updating registers in down-counting mode is the same as in up-counting mode, refer to Section 0.

The figure below shows some examples of the counter behavior and the update flags for different division factors in the down-counting mode.

Figure 9-5 Timing Diagram of The Down-counting, Internal Clock Divided Factor = 2/N



Center-aligned mode

In center-aligned mode, the counter increments from 0 to the value (TIMx_AR) – 1, a counter overflow event is generated, Then counts down from the auto-reload value (TIMx_AR) to 1 and generates a counter underflow event. Then the counter restarts from 0 and starts counting up again.

In this mode, the TIMx_CTRL1.DIR direction bits have no effect and the count direction is updated and specified by hardware. Center-aligned mode is active when the TIMx_CTRL1.CAMSEL bit is not equal to "00".

An update event can be generated at each counter overflows and at each counter underflows. Alternatively, an update event can also be generated by setting the TIMx_EVTGEN.UDGN bit (either by software or using a slave mode controller). In this case, the counter restarts counting from 0, as well as the counter of prescaler.

Note: if an update is generated due to a counter overflow, the auto-reload value will be updated before the counter is

reloaded.

Figure 9-6 Timing Diagram of The Center-Aligned, Internal Clock Divided Factor =2/N

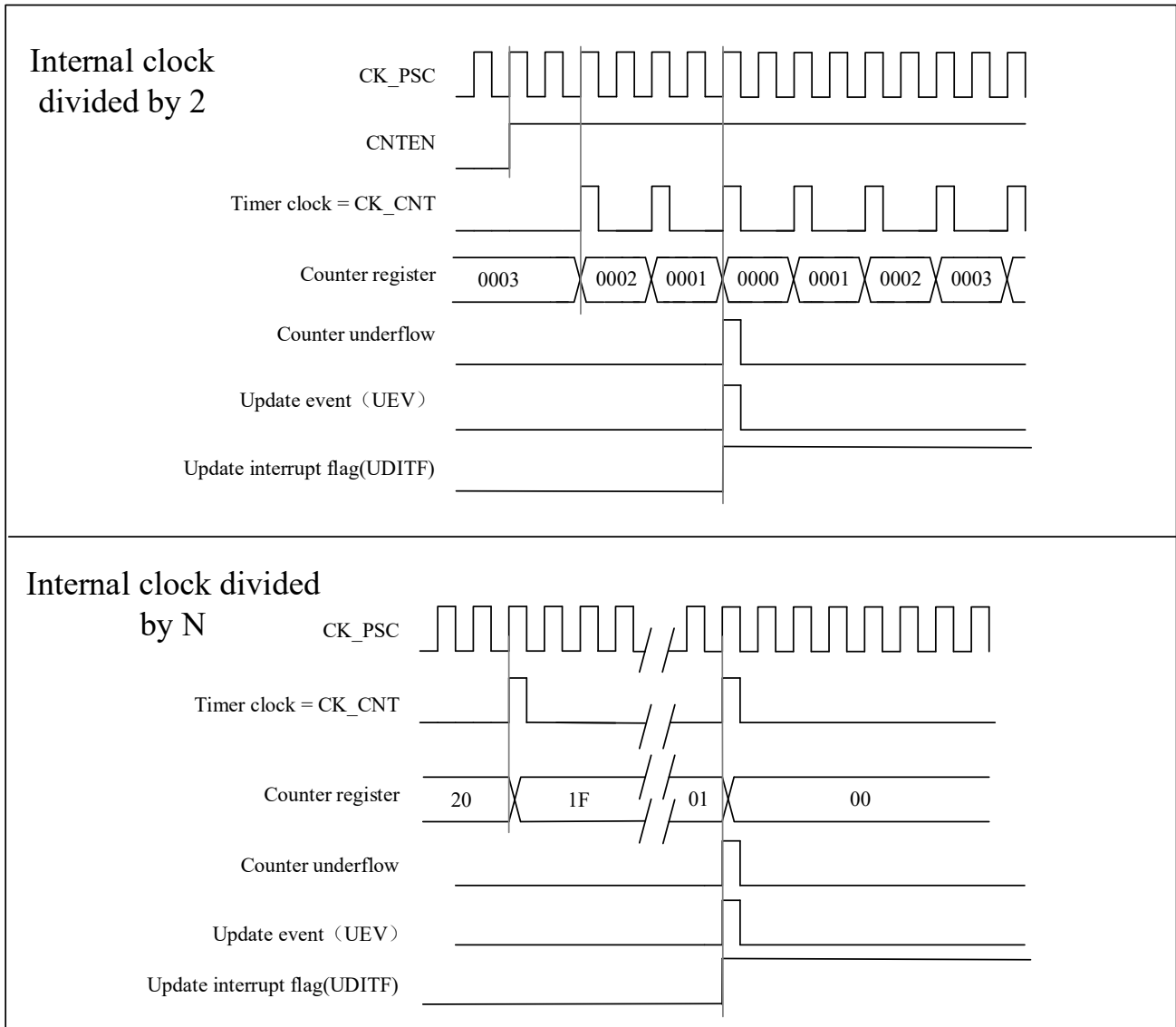
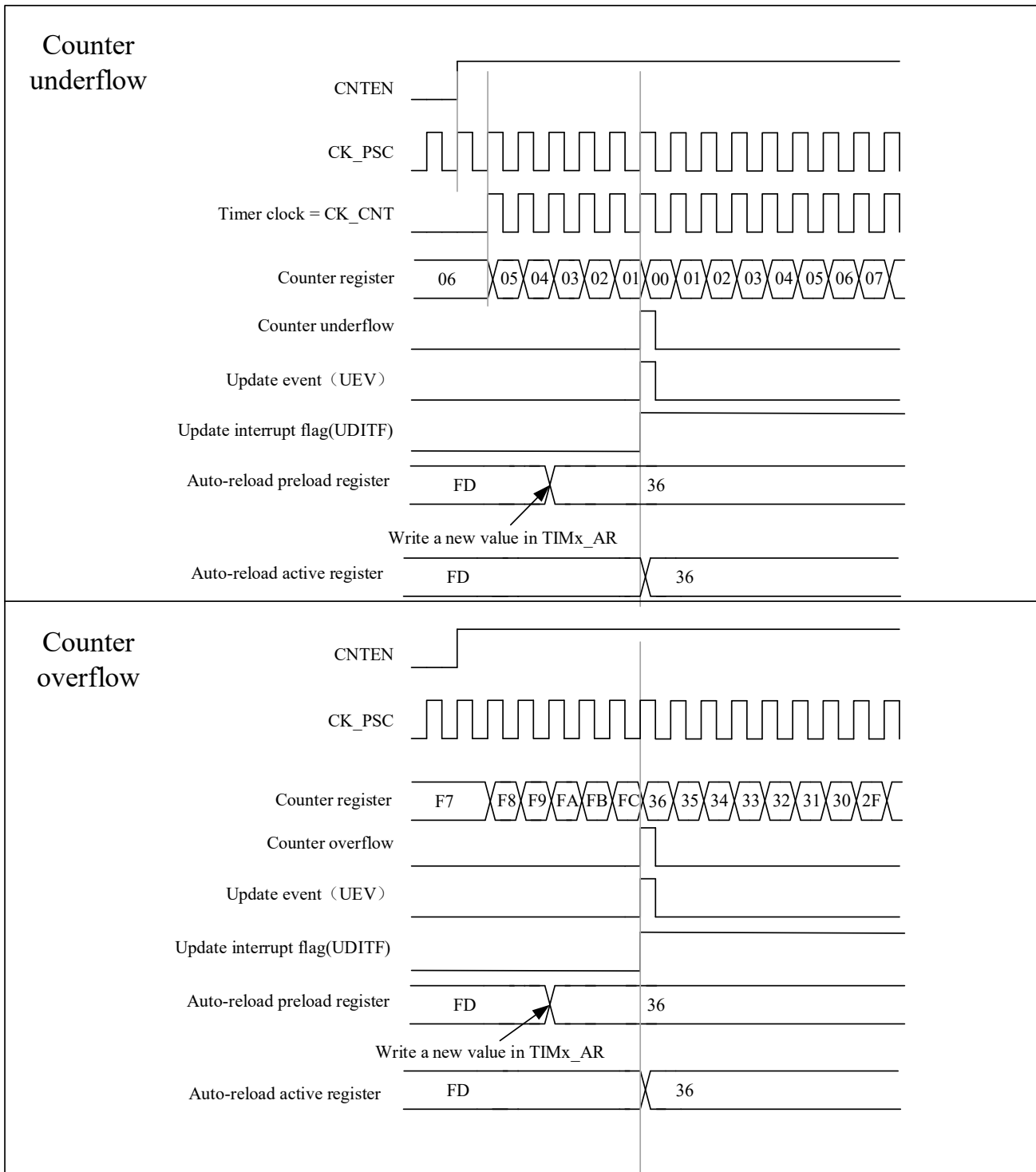


Figure 9-7 A Center-Aligned Sequence Diagram That Includes Counter Overflows and Underflows (ARPEN

= 1)



9.3.3 Repetition Counter

The time basic unit of Section 9.3.1 describes the conditions for generating an update event (UEV). An update event (UEV) is generated only when the repetition counter has reached zero, which is valuable for generating PWM signals.

This means that data is transferred from the preload registers to the shadow registers every N+1 counter overflows or underflows, where N is the value in the TIMx_REPCNT.

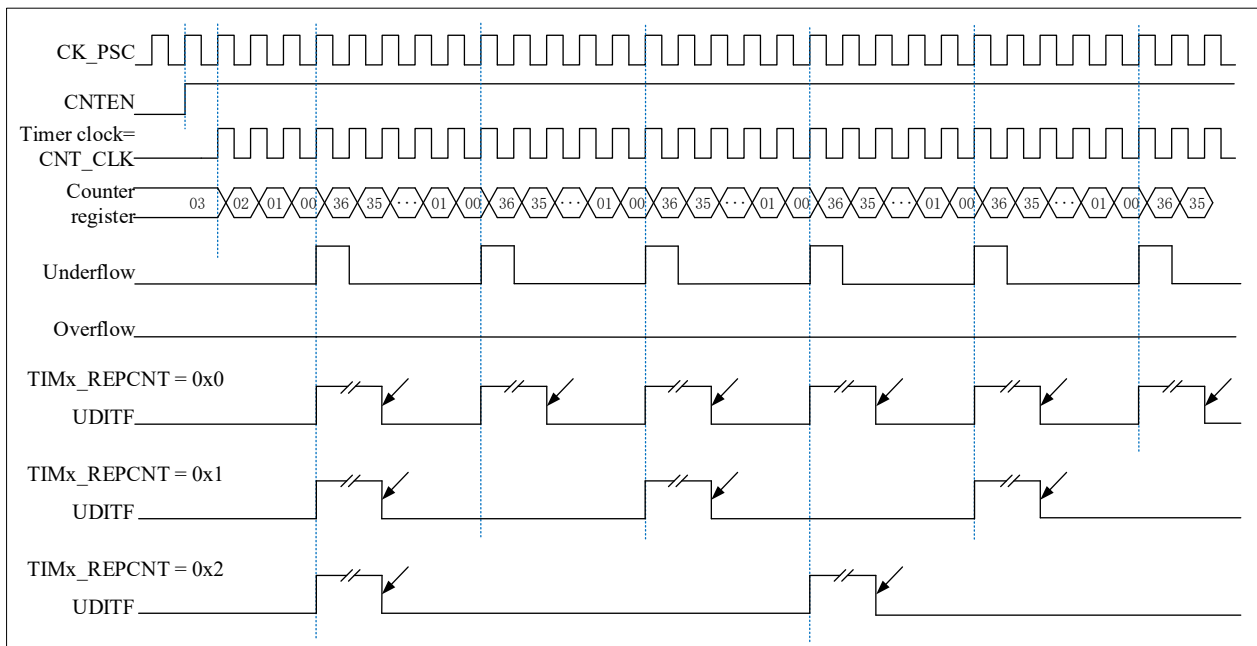
The repetition counter is decremented:

- In up-counting mode, each time the counter reaches the maximum value, an overflow occurs.
- In down-counting mode, each time the counter decrements to the minimum value, an underflow occurs.
- In center-aligned mode, each time the counter overflows or underflows.

Its repetition rate is defined by the value of the TIMx_REPCNT register.

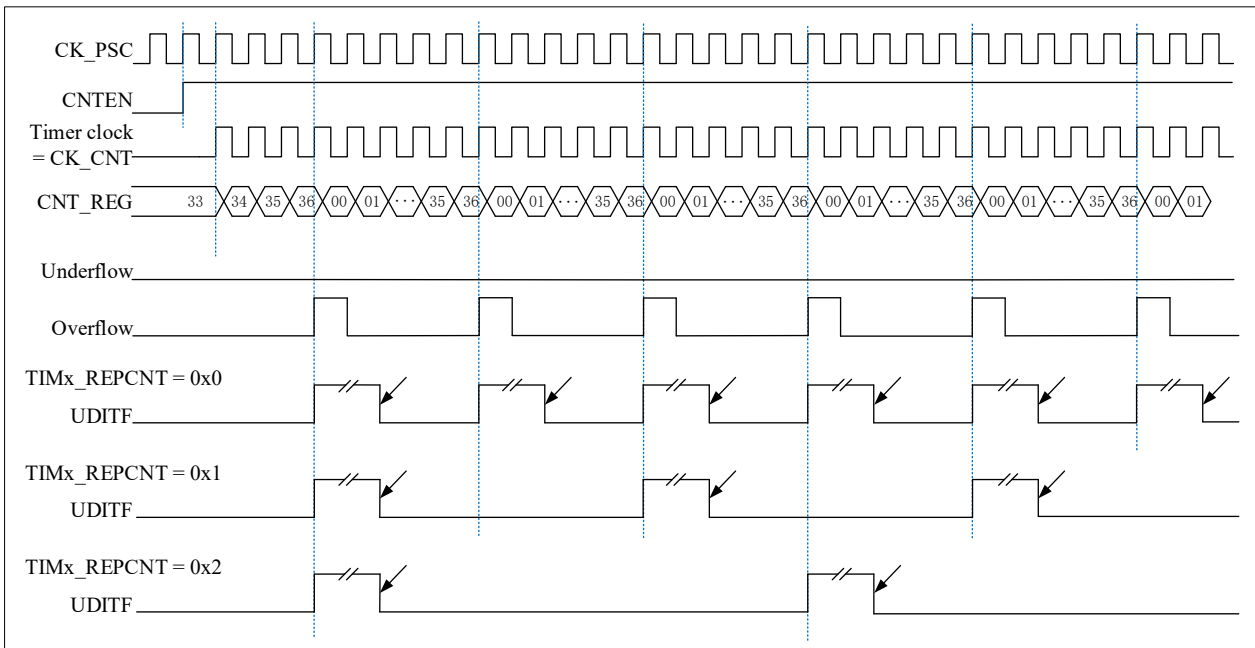
Repetition counters is an autoreload type. The update event (generated by setting TIMx_EVTGEN.UDGN or hardware through slave mode controller) occurs immediately whatever the value of the repetition counter is.

Figure 9-8 Repeat Count Sequence Diagram in Down-counting Mode



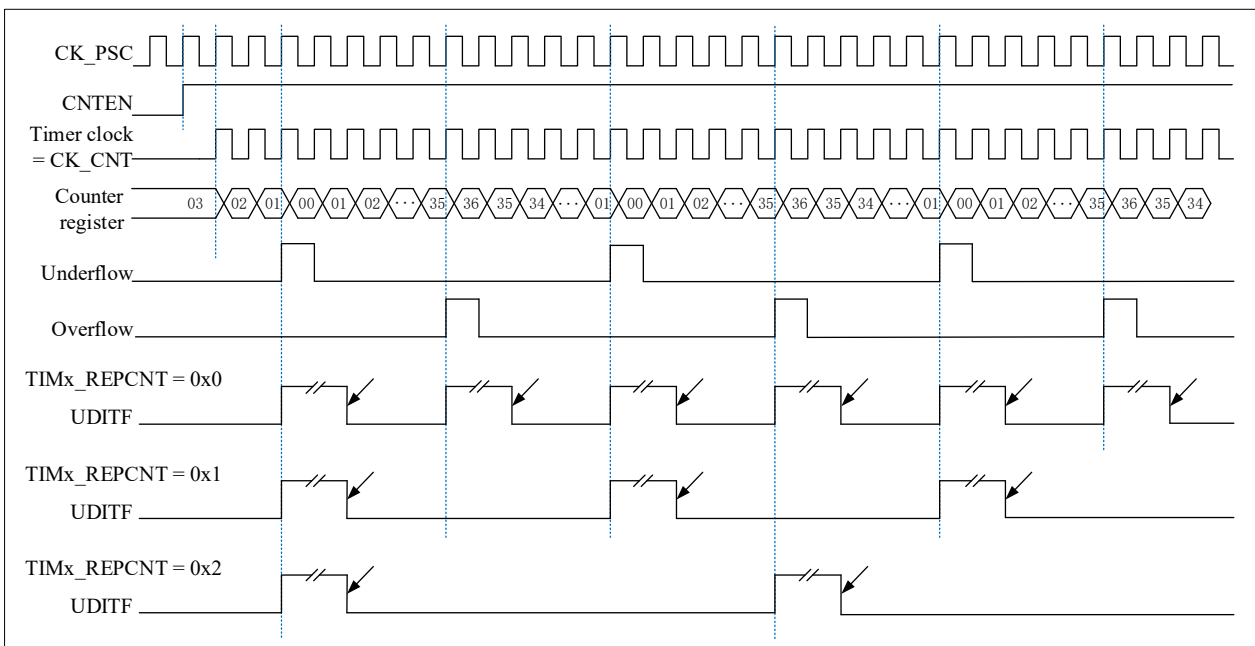
↙ Software clear

Figure 9-9 Repeat Count Sequence Diagram in Up-counting Mode



Software clear

Figure 9-10 Repeat Count Sequence Diagram in Center-aligned Mode



Software clear

9.3.4 Clock Selection

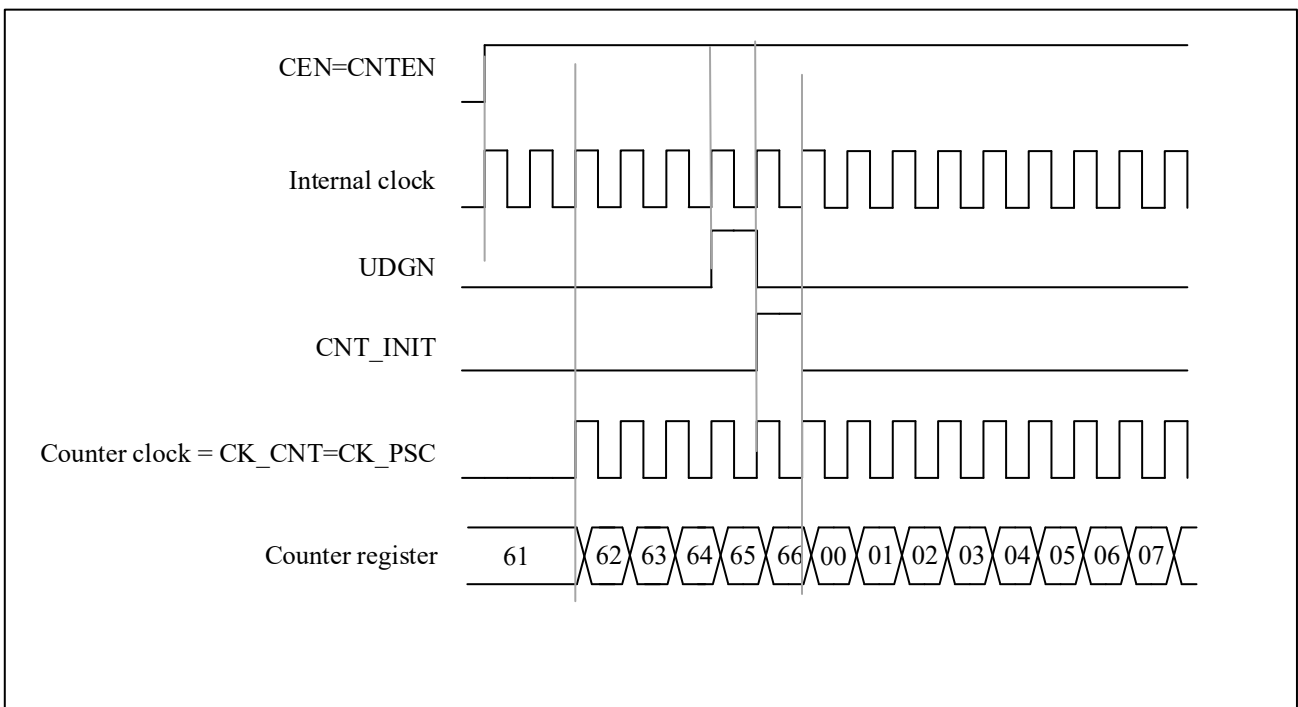
- The internal clock of advanced-control timers : CK_INT

- Two kinds of external clock mode :
 - External input pin
 - External trigger input (ETR)
- Internal trigger input (ITRx): one timer is used as a prescaler for another timer.

Internal clock source (CK_INT)

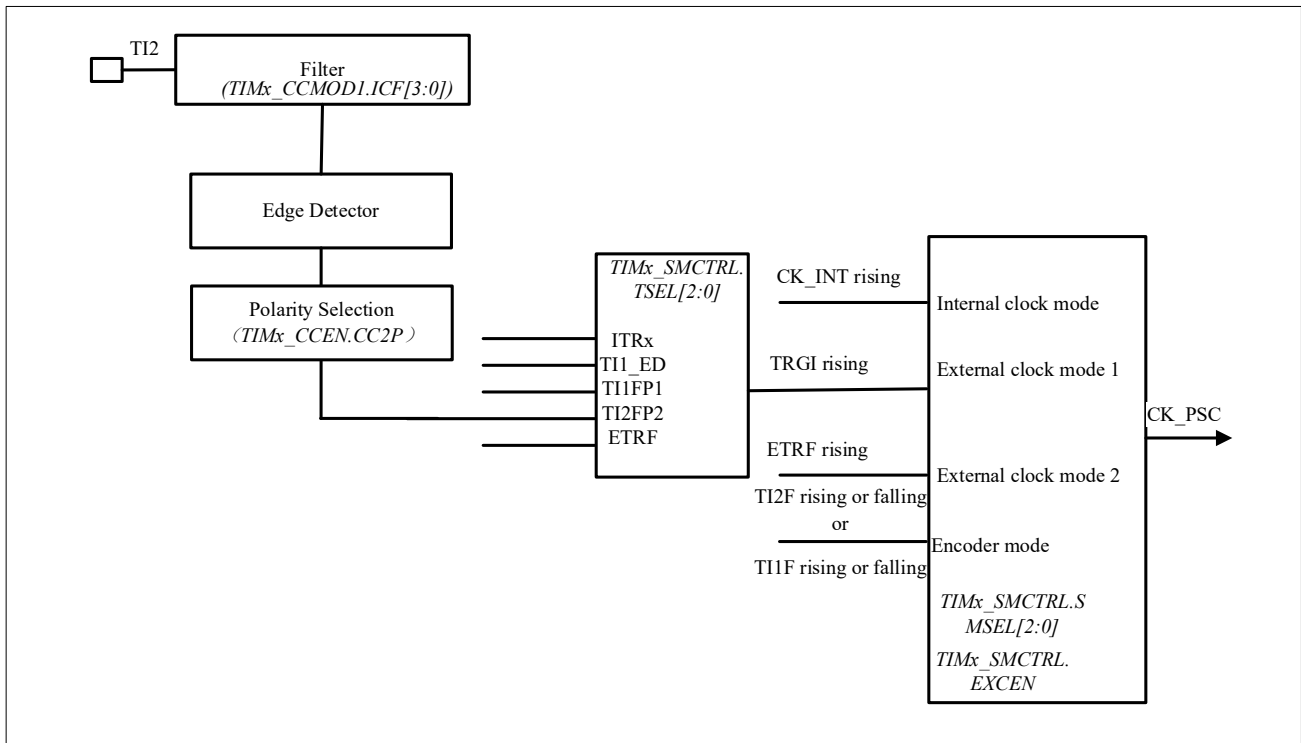
When the TIMx_SMCTRL.SMSEL is equal to “000”, the slave mode controller is disabled. The three control bits (TIMx_CTRL1.CNTEN, TIMx_CTRL1.DIR, TIMx_EVTGEN.UDGN) can only be changed by software (except TIMx_EVTGEN.UDGN, which remains cleared automatically). It is provided that the TIMx_CTRL1.CNTEN bit is written as '1' by software, the clock source of the prescaler is provided by the internal clock CK_INT.

Figure 9-11 Control Circuit in Normal Mode, Internal Clock Divided by 1



External clock source mode 1

Figure 9-12 TI2 External Clock Connection Example



This mode is selected by configuring `TIMx_SMCTRL.SMSEL=111`. The counter can be configured to count on the rising or falling edge of the clock at the selected input.

For example, to configure up-counting mode to count on the rising edge of the clock at the TI2 input, the configuration steps are as follows:

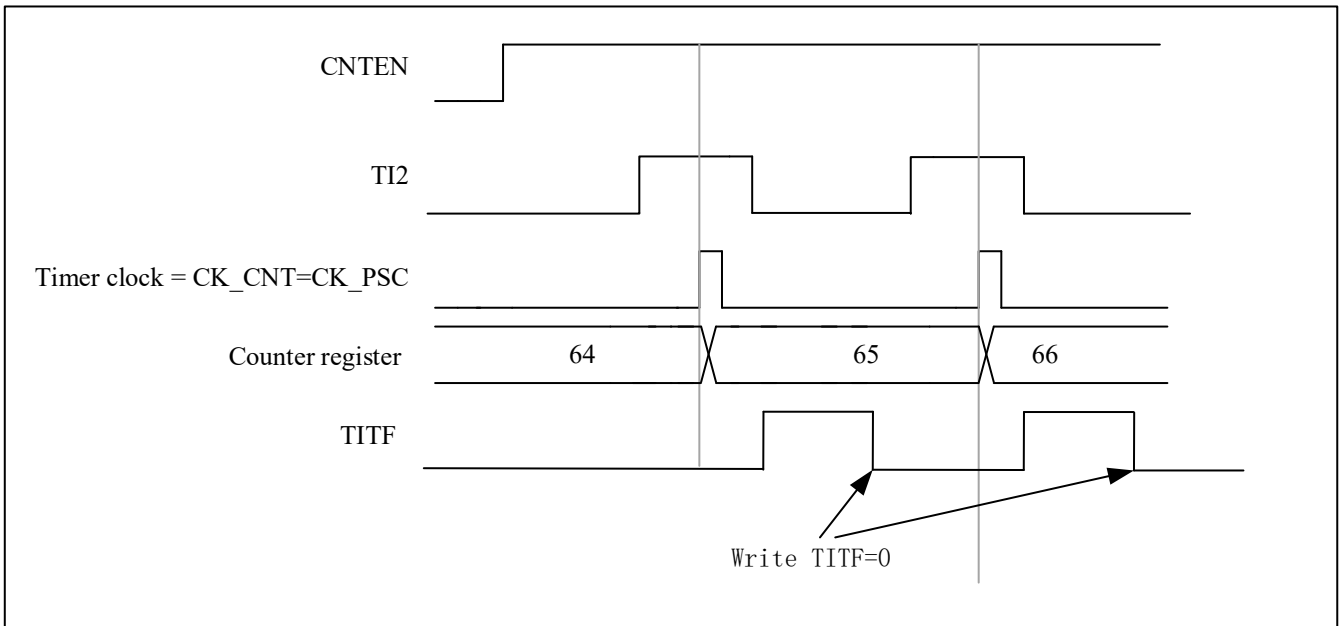
1. Configure `TIMx_CCMOD1.CC2SEL` equal to '01', CC2 channel is configured as input, IC2 is mapped to TI2
2. Configure `TIMx_CCEN.CC2P` equal to '0', select clock rising edge polarity
3. Configure `TIMx_CCMOD1.IC2F[3:0]` to select input filter bandwidth(if filter is not needed, keep IC2F bit at '0000')
4. Configure `TIMx_SMCTRL.SMSEL` equal to '111', select timer external clock mode 1
5. Configure `TIMx_SMCTRL.TSEL` equal to '110', select TI2 as the trigger input source
6. Configure `TIMx_CTRL1.CNTEN` equal to '1' to start the counter

Note: The capture prescaler is not used for triggering, so it does not need to be configured

When the rising edge of the timer clock occurs at `TI2=1`, the counter counts once and the `TIMx_STS.TITF` flag is pulled high.

The delay between the rising edge of TI2 and the actual clock of the counter depends on the resynchronization circuit at the input of TI2.

Figure 9-13 Control Circuit in External Clock Mode 1

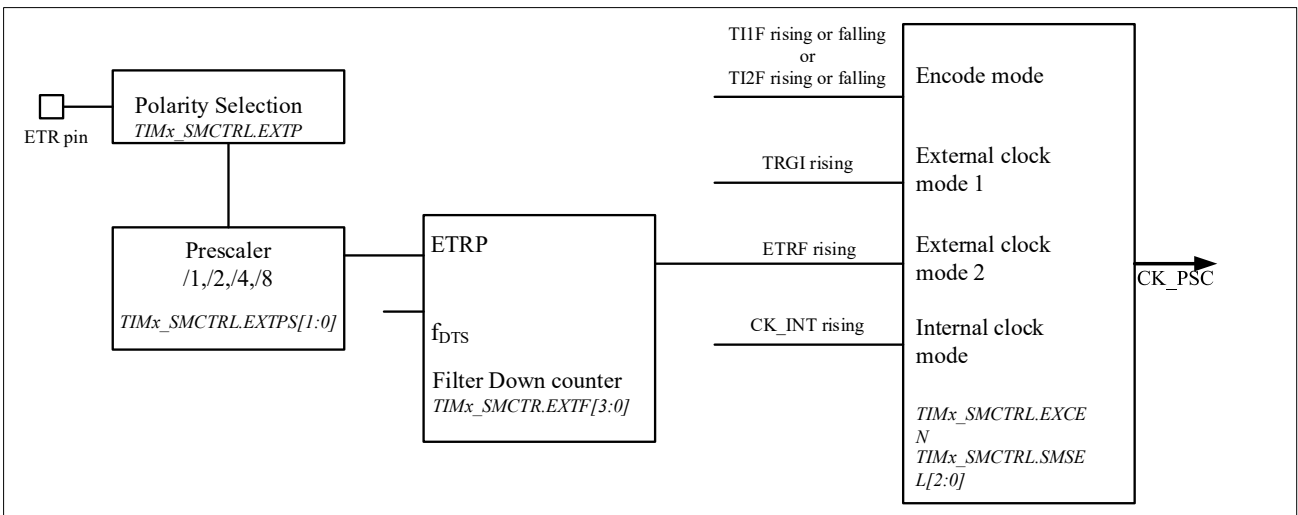


External clock source mode 2

This mode is selected by `TIMx_SMCTRL .EXCEN` equal to 1. The counter can count on every rising or falling edge of the external trigger input ETR.

The following figure is a schematic diagram of the external trigger input module in external clock source mode 2

Figure 9-14 External Trigger Input Block Diagram



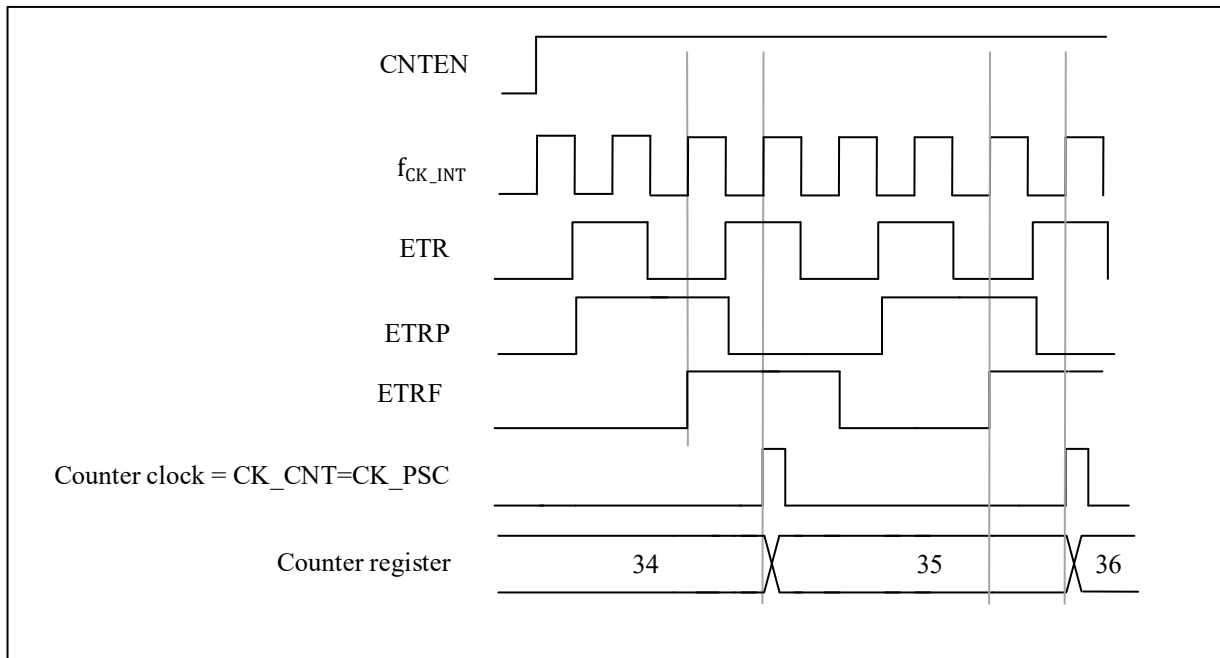
For example, use the following configuration steps to make the up counter count every 2 rising edges on ETR.

1. As no filter is needed in this case, setting `TIMx_SMCTRL .EXTF[3:0]` equal to '0000'
2. Configure the prescaler by setting `TIMx_SMCTRL.EXTPS[1:0]` equal to '01'
3. Select the polarity on ETR pin by setting `TIMx_SMCTRL.EXTP` equal to '0', The rising edge of ETR is valid
4. External clock mode 2 is selected by setting `TIMx_SMCTRL .EXCEN` equal to '1'

5. Enable the counter by setting TIMx_CTRL1.CNTEN equal to '1'

The counter counts every 2 rising edges of ETR. The delay between the rising edge of ETR and the actual clock of the counter is due to a resynchronization circuit on the ETR signal.

Figure 9-15 Control Circuit in External Clock Mode 2

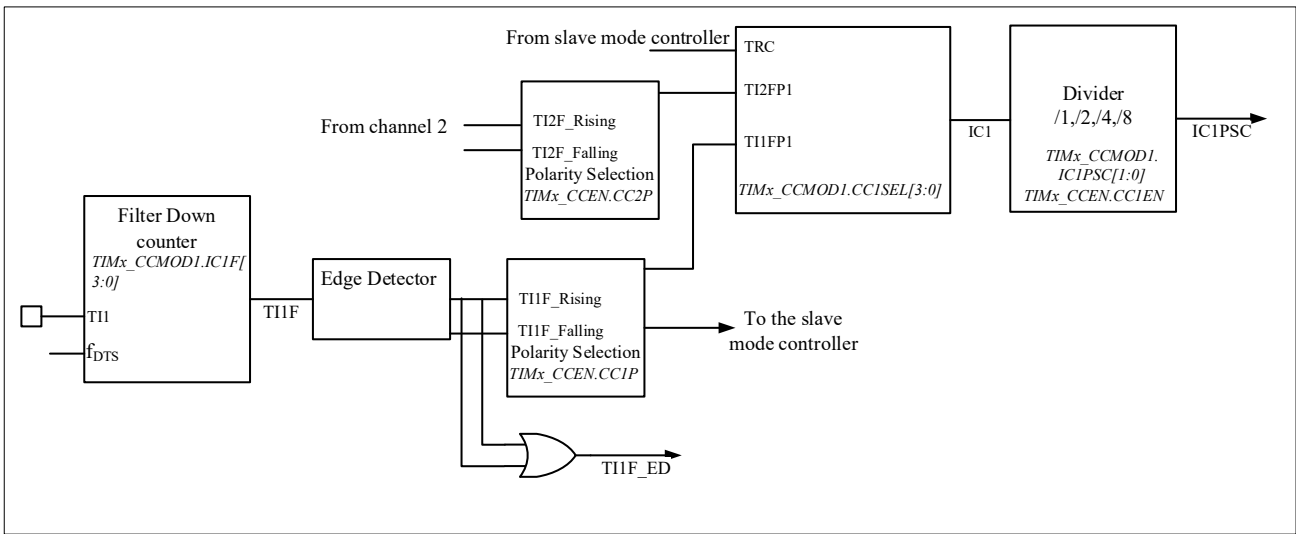


9.3.5 Capture/Compare Channels

The capture/compare channels include capture/compare registers and shadow registers. The input stage consists of digital filters, multiplexers and prescalers. The output section includes comparators and output controls.

The input signal T_{ix} is sampled and filtered to generate the signal T_{ixF}. Then, a signal (T_{ixF}_rising or T_{ixF}_falling) is generated by the edge detector of the polarity select function, the polarity of which is selected by the TIMx_CCEN.CC_xP bits. This signal can be used as a trigger input for the slave mode controller. At the same time, the signal IC_x is sent to the capture register after prescaler. The following figure shows a block diagram of a capture/compare channel.

Figure 9-16 Capture/Compare Channel (Example: Channel 1 Input Stage)



The output stage generates an intermediate waveform OCxRef (active high) as reference. The polarity acts at the end of the chain.

Figure 9-17 Capture/Compare Channel 1 Main Circuit

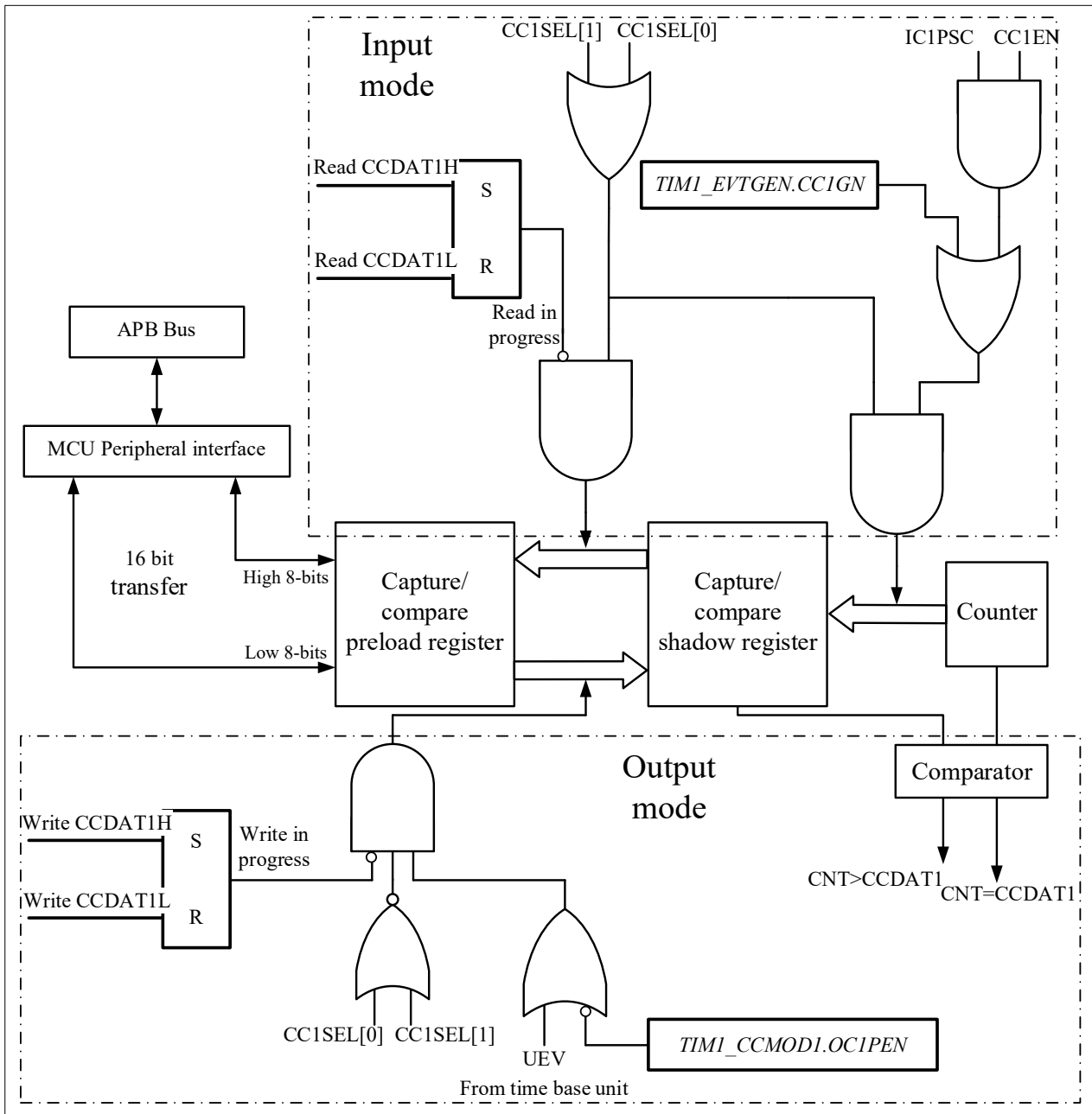


Figure 9-18 Output Part of Channelx (X= 1,2,3, Take Channel 1 As Example)

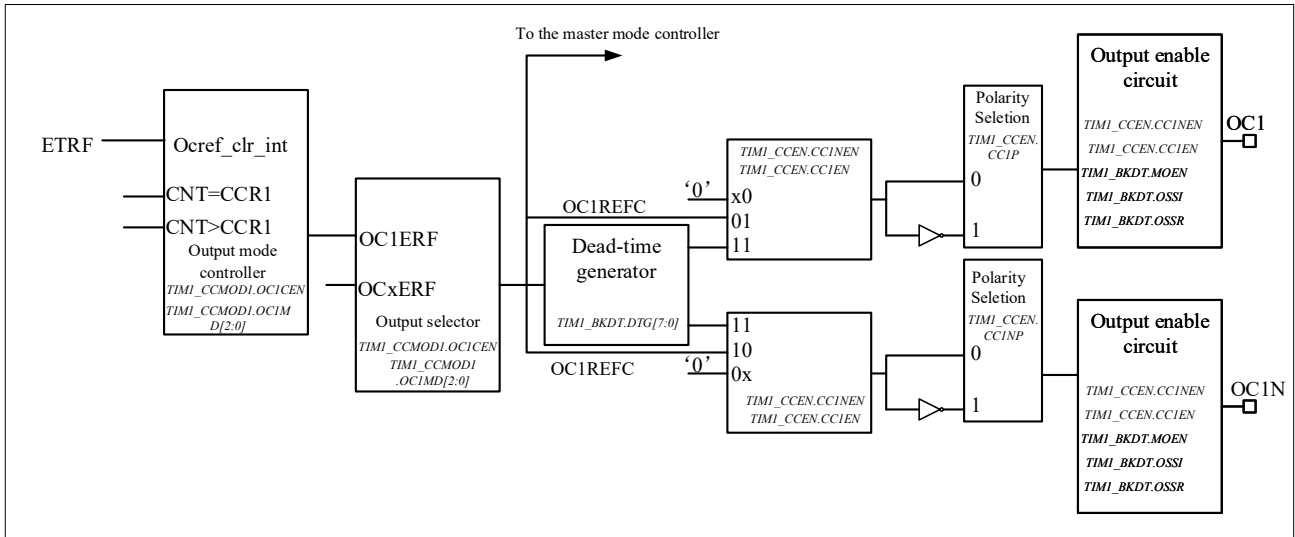
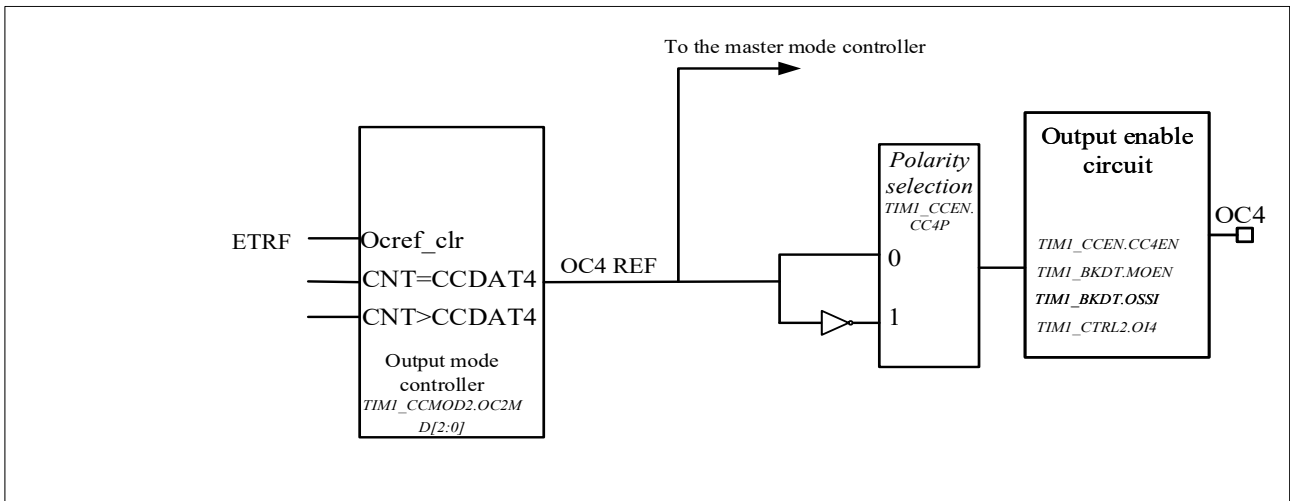


Figure 9-19 Output Part of Channelx (X= 4)



Reads and writes always access the preload registers when capturing/comparing. The two specific operating processes are as follows:

In capture mode, the capture is actually done in the shadow register, which is copied into the preload register.

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

9.3.6 Input Capture Mode

In input capture mode, the TIMx_CCxDATx registers are used to latch the counter value after the ICx signal detects.

There is a capture interrupt flag TIMx_STS.CCxITF, which can trigger an interrupt or DMA request when the corresponding interrupt enable is pulled high.

The TIMx_STS.CCxITF bit is set by hardware when a capture event occurs and is cleared by software or by reading the TIMx_CCxDATx register.

The overcapture flag TIMx_STS.CCxOCF is set equal to 1 when the counter value is captured in the TIMx_CCxDATx register and TIMx_STS.CCxITF is already pulled high. TIMx_STS.CCxOCF is cleared by writing 0 to it.

To achieve a rising edge of the TI1 input to capture the counter value into the TIMx_CCDA1 register, the configuration flow is as follows:

1. Select the valid input: Configure TIMx_CCMOD1.CC1SEL to '01'. At this time, the input is the CC1 channel, and IC1 is mapped to TI1.
2. Define the input filter duration required for programming: Define the sampling frequency of the TI1 input and the length of the digital filter by configuring the TIMx_CCMODx.ICxF bits. Example: If the input signal jitters up to 5 internal clock cycles, we must program a filter duration longer than 5 clock cycles. When 8 consecutive samples (sampled at f_{DTS} frequency) with the new level are detected, we can validate the transition on TI1. Then configure TIMx_CCMOD1.IC1F to '0011'.
3. Select the rising edge as the valid transition polarity on the TI1 channel by configuring TIMx_CCEN.CC1P=0.
4. Configure the input prescaler. In this example, configure TIMx_CCMOD1.IC1PSC= '00' to disable the prescaler because we want to capture every valid transition.
5. Enable capture by configuring TIMx_CCEN.CC1EN = '1'.

If you want to enable DMA request, you can configure TIMx_DINTEN.CC1DEN=1. If you want to enable related interrupt request, you can configure TIMx_DINTEN.CC1IEN bit=1

9.3.7 PWM Input Mode

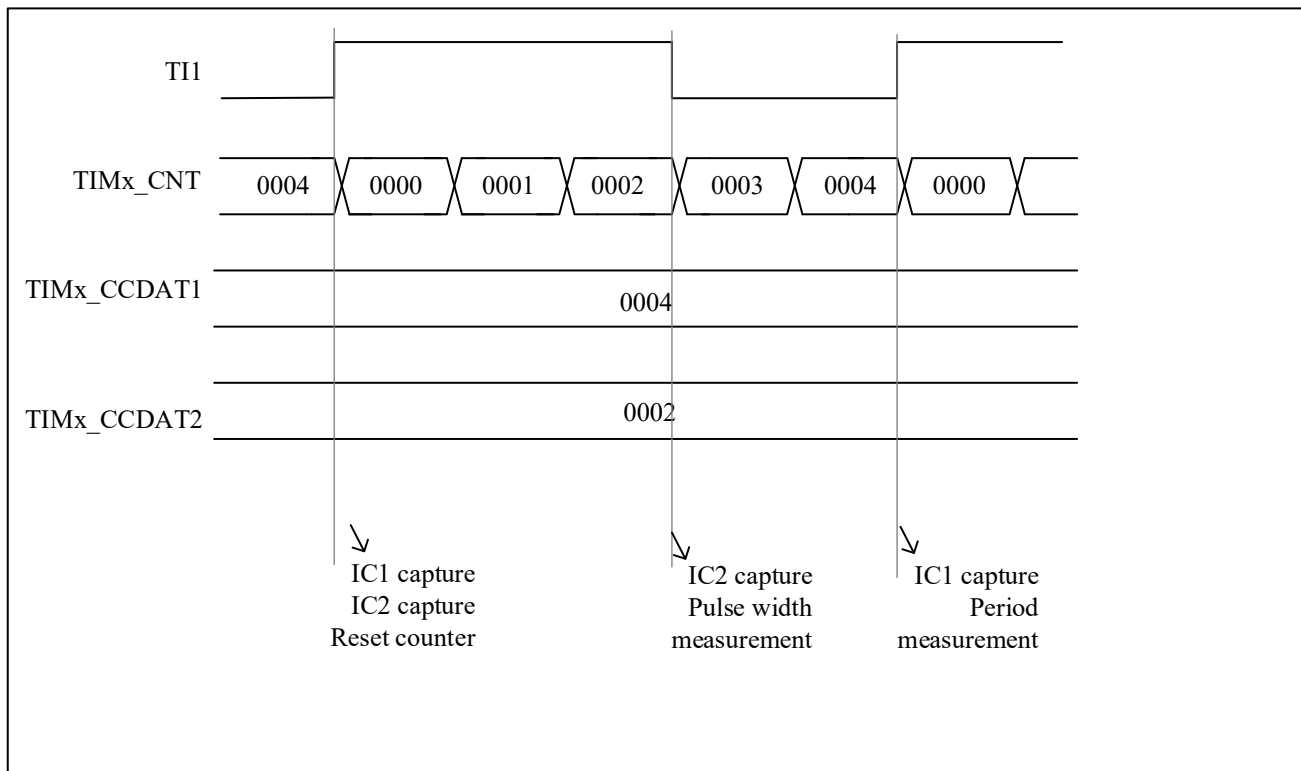
There are some differences between PWM input mode and normal input capture mode, including:

- Two ICx signals are mapped to the same TIx input.
- Two ICx signals are active on edges of opposite polarity.
- Select one of two TIxFP signals as trigger input.
- The slave mode controller is configured in reset mode.

For example, the following configuration flow can be used to know the period and duty cycle of the PWM signal on TI1 (It depends on the frequency of CK_INT and the value of the prescaler).

1. Configure TIMx_CCMOD1.CC1SEL equal to '01' to select TI1 as active input for TIMx_CCDA1.
2. Configure TIMx_CCEN.CC1P equal to '0' to select the active polarity of filtered timer input 1 (TI1FP1), active on the rising edge.
3. Configure TIMx_CCMOD1.CC2SEL equal to '10' to select TI1 as valid input for TIMx_CCDA2.
4. Configure TIMx_CCEN.CC2P equal to 1 to select the valid polarity of filtered timer input 2 (TI1FP2), active on the falling edge.
5. Configure TIMx_SMCTRL.TSEL=101 to select filtered timer input 1 (TI1FP1) as valid trigger input.
6. Configure TIMx_SMCTRL.SMSEL=100 to configure the slave mode controller to reset mode.
7. Configure TIMx_CCEN.CC1EN=1 and TIMx_CCEN.CC2EN=1 to enable capture.

Figure 9-20 PWM Input Mode Timing



Because of only filter timer input 1 (TI1FP1) and filter timer input 2 (TI2FP2) are connected to the slave mode controller, the PWM input mode can only be used with the TIMx_CH1/TIMx_CH2 signals.

9.3.8 Forced Output Mode

In output mode (TIMx_CCMODx.CCxSEL=00), the output compare signal can be forced to active or inactive level directly by software.

TIMx_CCMODx.OCxMD=101 can be set to force the output compare signal to active level. And the OCxREF will be forced high, OCx get the opposite value to CcxP polarity bit. On the other hand, user can set TIMx_CCMODx.OCxMD=100 to force the output compare signal to inactive level.

In this mode, the values of the TIMx_CCDATx shadow register and the counter still comparing with each other.

The comparison between the output compare register TIMx_CCDATx and the counter TIMx_CNT has no effect on OCxREF. And the flag still can be set. Therefore, the interrupt and DMA requests still can be sent.

9.3.9 Output Compare Mode

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

When the capture/compare register and the counter have the same value, the output compare function's operations are as follow:

- TIMx_CCMODx.OCxMD is for output compare mode, and TIMx_CCEN.CCxP is for output polarity. When the compare matches, if set TIMx_CCMODx.OCxMD=000, the output pin will keep its level; if set TIMx_CCMODx.OCxMD=001, the output pin will be set active; if set TIMx_CCMODx.OCxMD=010, the output pin will be set inactive; if set TIMx_CCMODx.OCxMD=011, the output pin will be set to toggle.
- Set TIMx_STS.CCxITF.

- If user set TIMx_DINTEN.CCxIEN, a corresponding interrupt will be generated.
- If user set TIMx_DINTEN.CCxDEN and set TIMx_CTRL2.CCDSEL to select DMA request, the DMA request will be sent.

User can set TIMx_CCMODx.OCxPEN to choose capture/compare shadow register using capture/compare preload registers (TIMx_CCDAx) or not.

The timing resolution is one count of the counter.

In one-pulse mode, the output compare mode can also be used to output a single pulse.

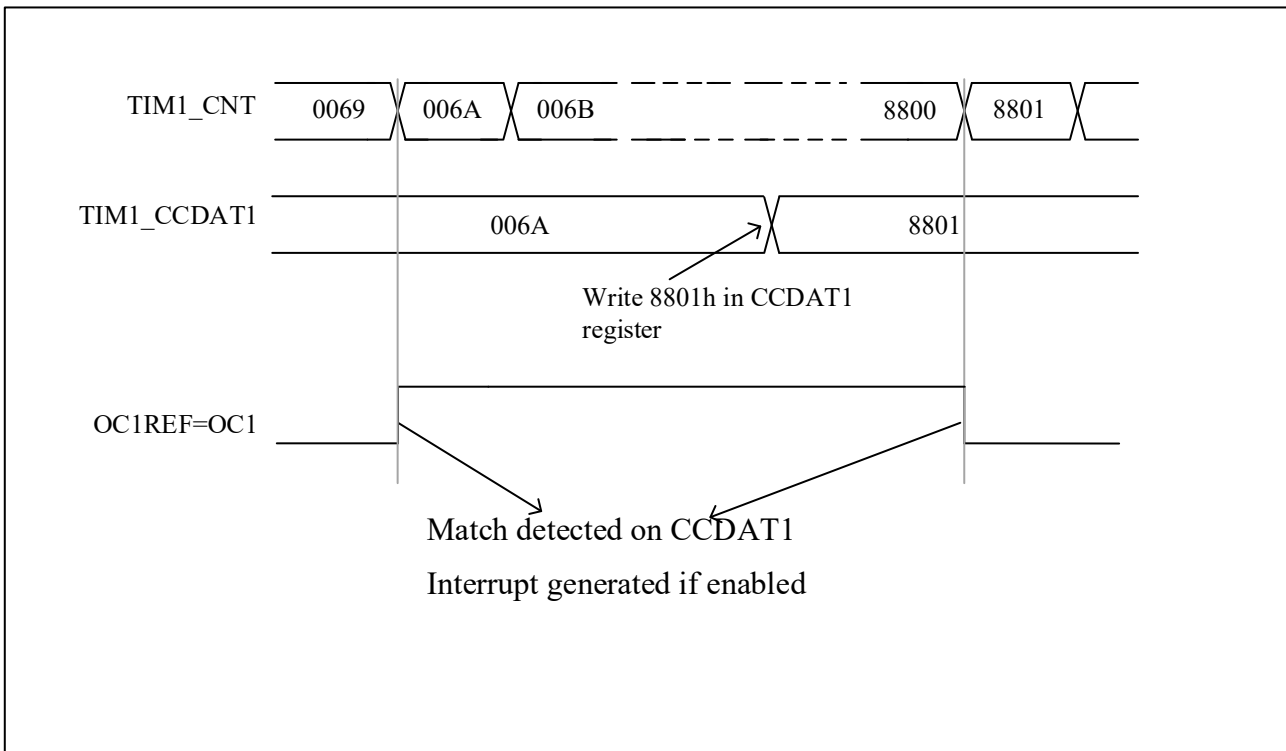
Here are the configuration steps for output compare mode:

1. First of all, user should select the counter clock.
2. Secondly, set TIMx_AR and TIMx_CCDAx with desired data.
3. Set TIMx_DINTEN.CCxIEN if user need to generate an interrupt..
4. Then select the output mode by set TIMx_CCEN.CCxP, TIMx_CCMODx.OCxMD, TIMx_CCEN.CCxEN, etc.
5. At last, set TIMx_CTRL1.CNTEN to enable the counter.

User can update the output waveform by setting TIMx_CCDAx at any time, as long as the preload register is not enabled. Otherwise the TIMx_CCDAx shadow register will be updated at the next update event.

Here is an example.

Figure 9-21 Output Compare Mode, Toggle on OC1



9.3.10 PWM Mode

Pulse width modulation mode is used to generate a signal with a frequency determined by the value of the TIMx_AR

register and a duty cycle determined by the value of the TIMx_CCxDATx register. And depending on the value of TIMx_CTRL1.CAMSEL, the TIM can generate PWM signal in edge-aligned mode or center-aligned mode.

User can select PWM mode 1 or PWM mode 2 by setting TIMx_CCMODx. OCxMD=110 or setting TIMx_CCMODx. OCxMD=111. To enable preload register, user must set corresponding TIMx_CCMODx.OCxPEN. Then set TIMx_CTRL1.ARPEN to auto-reload preload register.

User can set polarity of OCx by setting TIMx_CCEN.CCxP. User need to set the combination of the value of CCxEN, CCxNEN, MOEN, OSSI, and OSSR in TIMx_CCEN and TIMx_BKDT to enable the output of OCx.

The values of TIMx_CNT and TIMx_CCxDATx are always compared with each other when the TIM is under PWM mode.

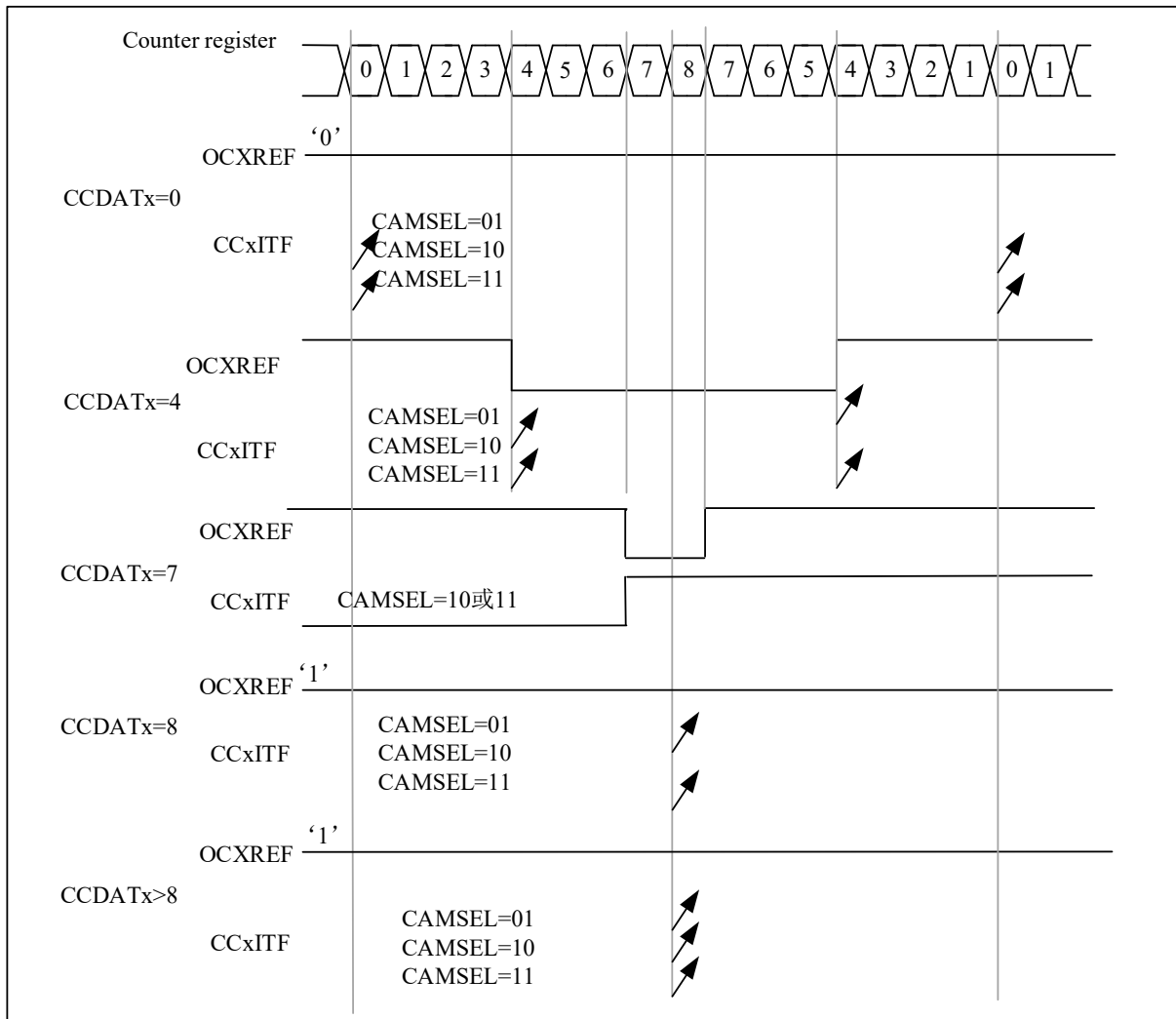
Only when an update event occurs, the preload register will be transferred to the shadow register. Therefore user must reset all the registers by setting TIMx_EVTGEN.UDGN before the counter starts counting..

PWM center-aligned mode

The PWM center-aligned mode is active when setting TIMx_CTRL1.CAMSEL equal to 01, 10 or 11,. The setting of the compare flag depends on the value of TIMx_CTRL1.CAMSEL. There are three kinds of situation that the compare flag is set, only when the counter counts up, only when the counter counts down, or when the counter counts up and counts down. The TIMx_CTRL1.DIR is updated by hardware and must not be changed by software.

The example of center-aligned PWM waveforms is as follow, and the setting of the waveform are: TIMx_AR=8, PWM mode 1, the compare flag is set when the counter counts down corresponding to TIMx_CTRL1. CAMSEL=01.

Figure 9-22 Center-aligned PWM Waveform (AR=8)



When using center-aligned mode, users should pay attention to the following considerations:

- It depends on the value of TIMx_CTRL1.DIR that the counter counts up or down. Caution that the DIR and CAMSEL bits should not be changed at the same time.
- User should not write the counter while running in center-aligned mode, otherwise it will cause unexpected results. Here are some examples:
 - If the value written into the counter is 0 or the value of TIMx_AR, the direction will be updated but the update event will not be generated.
 - If the value written into the counter is greater than the value of auto-reload, the direction will not be updated.
- For safety reasons, it is recommended that users set TIMx_EVTGEN.UDGN to generate an update by software before starting the counter, and do not write the counter while it is running.

PWM edge-aligned mode

There are two kinds of configuration in edge-aligned mode, up-counting and down-counting.

- **Up-counting**

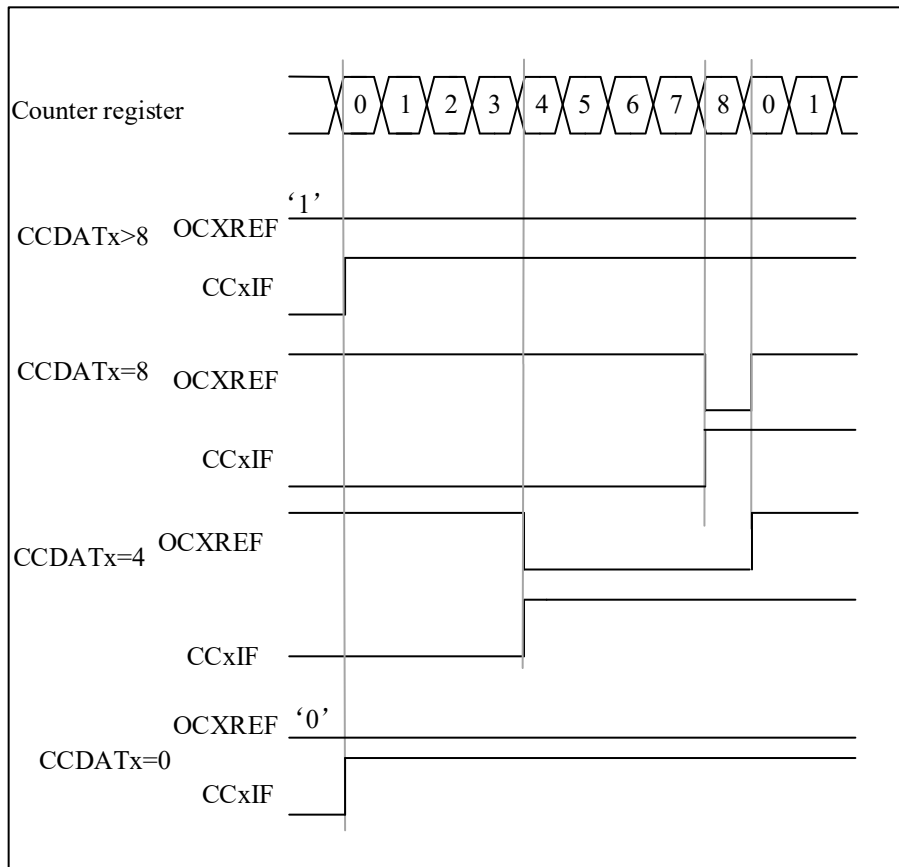
User can set TIMx_CTRL1.DIR=0 to make counter counts up.

Example for PWM mode 1.

When TIMx_CNT < TIMx_CCDA Tx, the reference PWM signal OCxREF is high. Otherwise it will be low. If the compare value in TIMx_CCDA Tx is greater than the auto-reload value, the OCxREF will remains 1. Conversely, if the compare value is 0, the OCxREF will remains 0.

When TIMx_AR=8, the PWM waveform is as follow.

Figure 9-23 Edge-aligned PWM Waveform (APR=8)



- **Down-counting**

User can set TIMx_CTRL1.DIR=1 to make counter counts down.

Example for PWM mode 1.

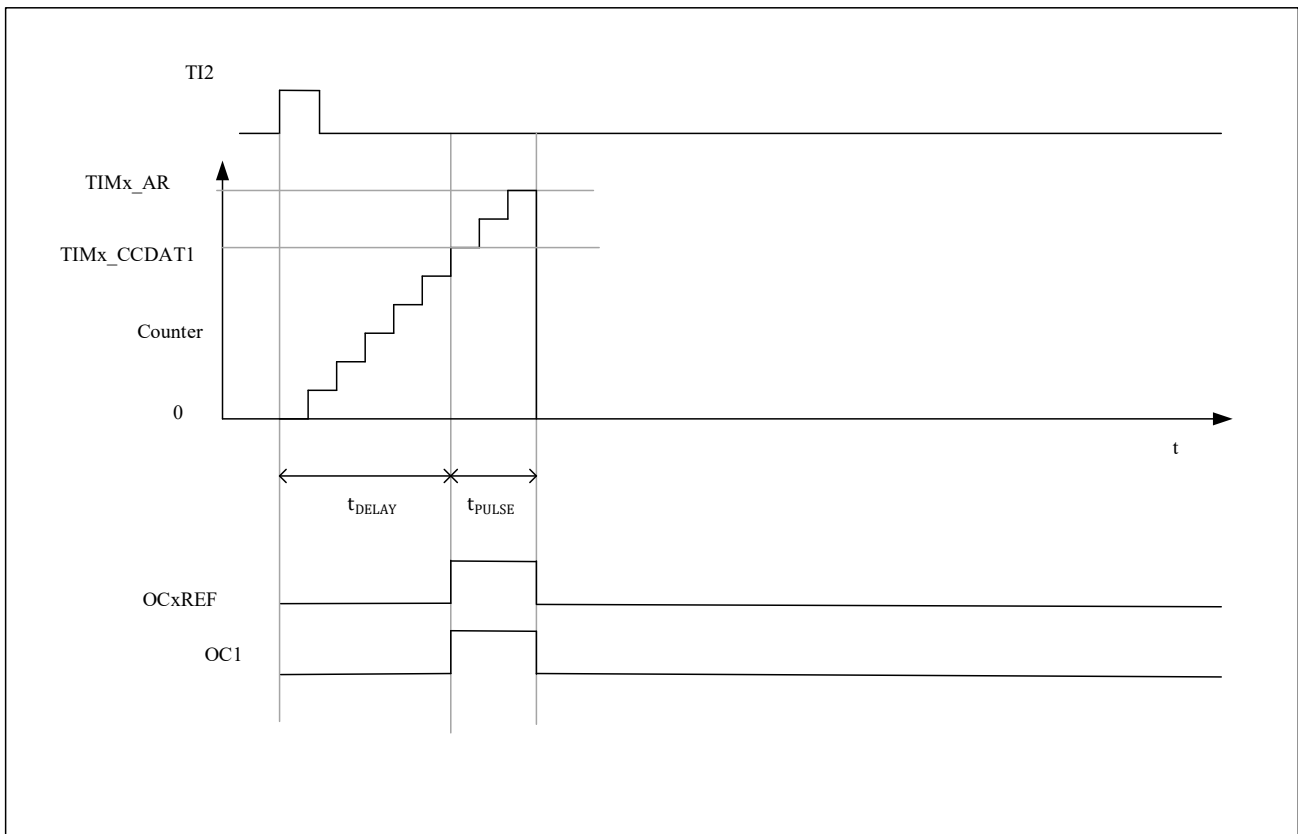
When TIMx_CNT > TIMx_CCDA Tx, the reference PWM signal OCxREF is low. Otherwise it will be high. If the compare value in TIMx_CCDA Tx is greater than the auto-reload value, the OCxREF will remains 1.

Note: If the nth PWM cycle CCDA Tx shadow register >= AR value, the shadow register value of CCDA Tx in the (n+1)th PWM cycle is 0. At the moment when the counter is 0 in the (n+1)th PWM cycle, although the value of the counter = CCDA Tx shadow register = 0 and OCxREF = '0', no compare event will be generated.

9.3.11 One-pulse Mode

In the one-pulse mode (ONEPM), a trigger signal is received, and a pulse tPULSE with a programmable pulse length is generated after a programmable delay tDELAY. The output mode needs to be configured as output compare mode or PWM mode. After selecting one-pulse mode, the counter will stop counting after the update event UEV is generated.

Figure 9-24 Example of One-pulse Mode



The following is an example of the one-pulse mode:

A rising edge trigger is detected from the TI2 input, and a pulse with a length of t_{PULSE} is generated on OC1 after a delay of t_{DELAY} .

1. Counter configuration: count up, counter $TIMx_CNT < TIMx_CCDAT1 \leq TIMx_AR$;
2. TI2FP2 is mapped to TI2, $TIMx_CCMOD1.CC2SEL = '01'$; TI2FP2 is configured for rising edge detection, $TIMx_CCEN.CC2P = '0'$;
3. TI2FP2 acts as the trigger (TRGI) of the slave mode controller and starts the counter, $TIMx_SMCTRL.TSEL = '110'$, $TIMx_SMCTRL.SMSEL = '110'$ (trigger mode);
4. The count value to be delayed (t_{DELAY}) is written to **TIMx_CCDAT1**, $TIMx_AR - TIMx_CCDAT1$ is the count value of the pulse length t_{PULSE} ;
5. Configure $TIMx_CTRL1.ONEPM = 1$ to enable single pulse mode, configure $TIMx_CCMOD1.OC1MD = '111'$ to select PWM2 mode;
6. Wait for an external trigger event on TI2, and a one pulse waveform will be output on OC1;

Special case: OCx fast enable:

In one-pulse mode, an edge is detected through the TIX input, and triggers the start of the counter to count to the comparison value and output a pulse. These operations limit the minimum delay t_{DELAY} that can be achieved.

You can set $TIMx_CCMODx.OCxFEN = 1$ to turn on OCx fast enable, the OCxREF signal will be forced to be converted to the same level as the comparison match occurs immediately after triggering the rising edge, no matter what the

comparison result is. OCxFEN fast enable takes effect only when the channel mode is configured for PWM1 and PWM2 modes.

9.3.12 Clearing The Ocxref Signal on An External Event

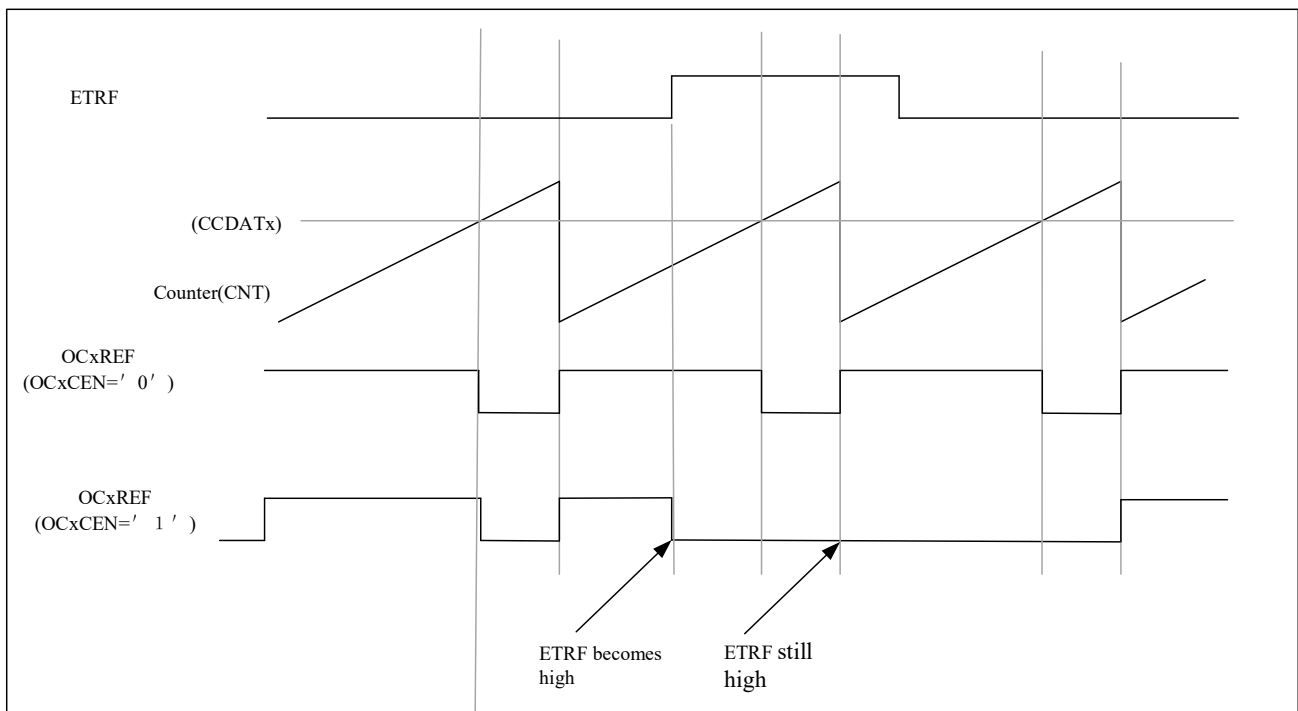
If setting TIMx_CCMODx.OCxCEN=1, high level of ETRF input can be used to driven the OCxREF signal to low and the OCxREF signal will remains low until the next UEV happens. Only output compare and PWM modes can use this function. This cannot be used when it is in forced mode.

For example: The ETR signal is connected to the output of the comparator to control the current. The operation for ETR should be as follow:

1. Set TIMx_SMCTRL.EXTPS=00 to disable the external trigger prescaler.
2. Set TIMx_SMCTRL.EXCEN=0 to disable the external clock mode 2.
3. Set TIMx_SMCTRL.EXTP and TIMx_SMCTRL.EXTF to configure the external trigger polarity and external trigger filter according to the need.

Example for the case that the behavior of OCxREF signal for different value of OCxCEN when ETRF input becomes high. Timer is set to be in PWM mode in this case.

Figure 9-25 Clearing The Ocxref of Timx



9.3.13 Complementary Outputs with Dead-time Insertion

The advanced-control timer can output two complementary signals, and manage the switching-off and switching-on instants of outputs. This is called dead-time. it has to be adjusted depending on the devices connected to the outputs and their characteristics.

The polarity of outputs is set by TIMx_CCEN.CCxP and TIMx_CCEN.CCxNP. And this selection is independently for each output.

The complementary signals OCx and OCxN are set by the combination of several control bits, which are TIMx_CCEN.CCxEN, TIMx_CCEN.CCxNEN, TIMx_BKDT.MOEN, TIMx_CTRL2.OIx, TIMx_CTRL2.OIxN, TIMx_BKDT.OSSI, and TIMx_BKDT.OSSR. the dead-time will be activated when switching to the IDLE state,.

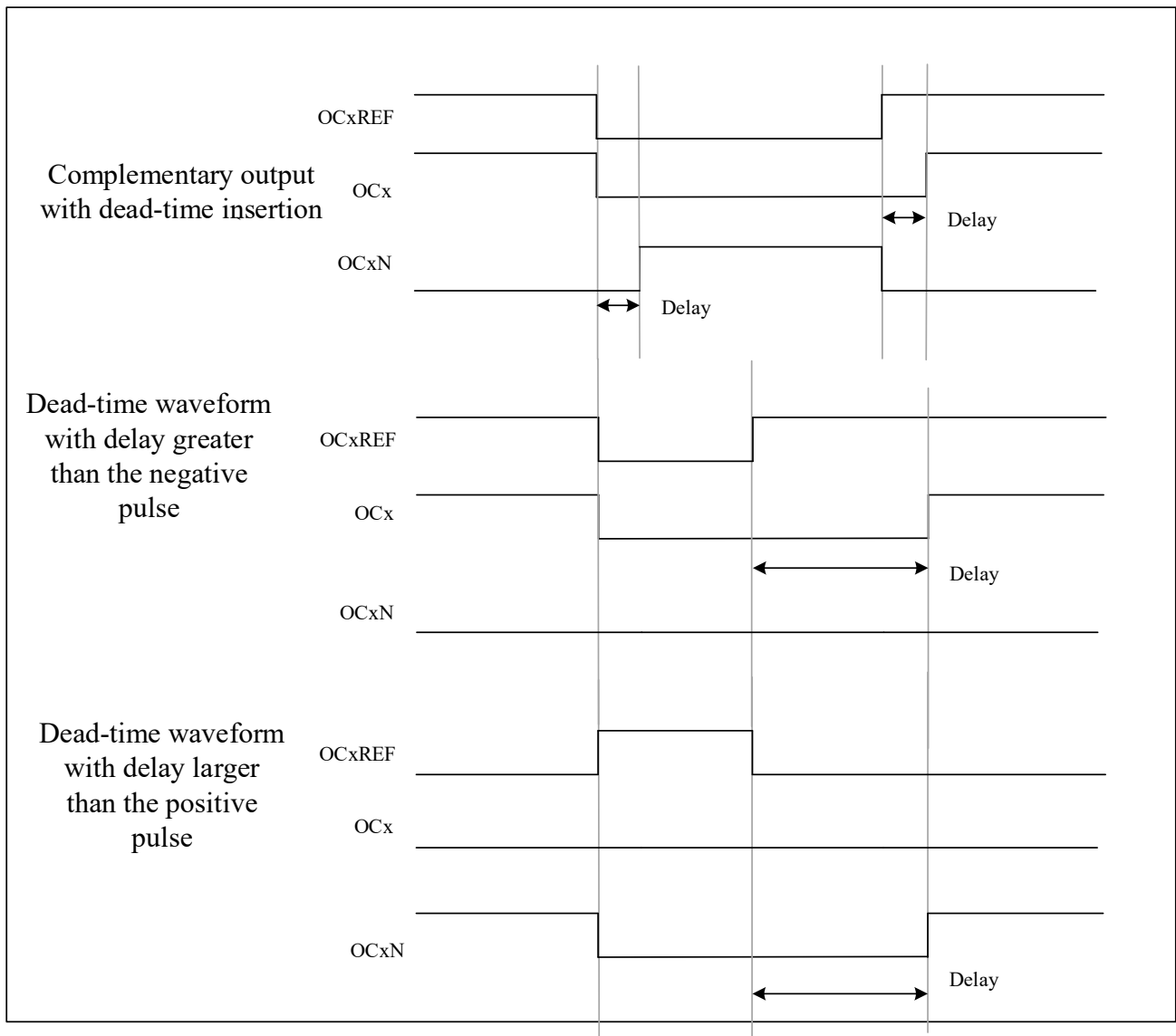
when TIMx_CCEN.CCxEN and TIMx_CCEN.CCxNEN are set at the same time, a dead-time will be insert. If there is a break circuit, the TIMx_BKDT.MOEN should be set too. There are 10-bit dead-time generators for each channel.

Reference waveform OCxREF can generates 2 outputs OCx and OCxN. If OCx and OCxN are active high, the OCx output signal is the same as the reference signal and the OCxN output signal is the opposite of the reference signal. However, OCx output signal will be delayed relative to the reference rising edge, and the OCxN output signal will be delayed relative to the reference falling edge. If the delay is greater than the width of the active OCx or OCxN output, the corresponding pulse will not generated.

The relationships between the output signals of the dead-time generator and the reference signal OCxREF are as follow.

Assume that TIMx_CCEN.CCxP=0, TIMx_CCEN.CCxNP=0, TIMx_BKDT.MOEN=1, TIMx_CCEN.CCxEN=1, TIMx_CCEN.CCxNEN=1.

Figure 9-26 Complementary Output with Dead-time Insertion



TIMx_BKDT.DTGN is set to program the dead-time delay for each of the channels.

Redirecting OCxREF to OCx or OCxN

In output mode, OCxREF can be re-directed to the OCx output or to OCxN output by setting TIMx_CCEN.CCxEN and TIMx_CCEN.CCxNEN. Here are two ways to use this function. When the complementary remains at its inactive level, user can use this function to send a specific waveform, such as PWM or static active level. User can also use this function to set both outputs in their inactive level or both outputs active and complementary with dead-time.

If TIMx_CCEN.CCxEN=0 and TIMx_CCEN.CCxNEN=1, it will not complemented, and OCxN will become active when OCxREF is high. On the other hand, if TIMx_CCEN.CCxEN=1 and TIMx_CCEN.CCxNEN=1, OCx will become active when OCxREF is high. On the contrary, OCxN will become active when OCxREF is low.

9.3.14 Break Function

The output enable signals and inactive levels will be modified by setting the corresponding control bits when using the break function. The output of OCx and OCxN cannot at the active level at the same time no matter when, that is $(CCxP \wedge OIx) \wedge (CCxN \wedge OIxN) = 0$.

When multiple break signals are enabled, each break signal constitutes an OR logic. Here are some signal which can be the source of breaking.

- The break input pin
- A clock failure event, generated by the clock security system in the clock controller.
- A PVD failure event.
- Core Hardfault event.
- The output signal of the comparator (configured in the comparator module, high level break).
- By software through the TIMx_EVTGEN.BGN.

The break circuit will be disable after reset. And the MOEN bit will be low. TIMx_BKDT.BKEN can be set to enable the break function. The polarity of break input signal can be selected by setting TIMx_BKDT.BKP. The TIMx_BKDT.BKEN and TIMx_BKDT.BKP can be modified at the same time. After setting the TIMx_BKDT.BKEN and TIMx_BKDT.BKP, there is 1 APB clock cycle delay before the configuration take effect. Therefore, user need to wait 1 APB clock cycle to read back the written bit value.

The falling edge of MOEN can be asynchronous, so there set a resynchronization circuit between the actual signal and the synchronous control bit. This circuit will cause a delay between the asynchronous and the synchronous signal. If TIMx_BKDT.MOEN is set whereas it was low, a delay must be inserted before reading the value. Because an asynchronous signal was written but user read the synchronous signal.

The behaviors that after a break occurs are as follow:

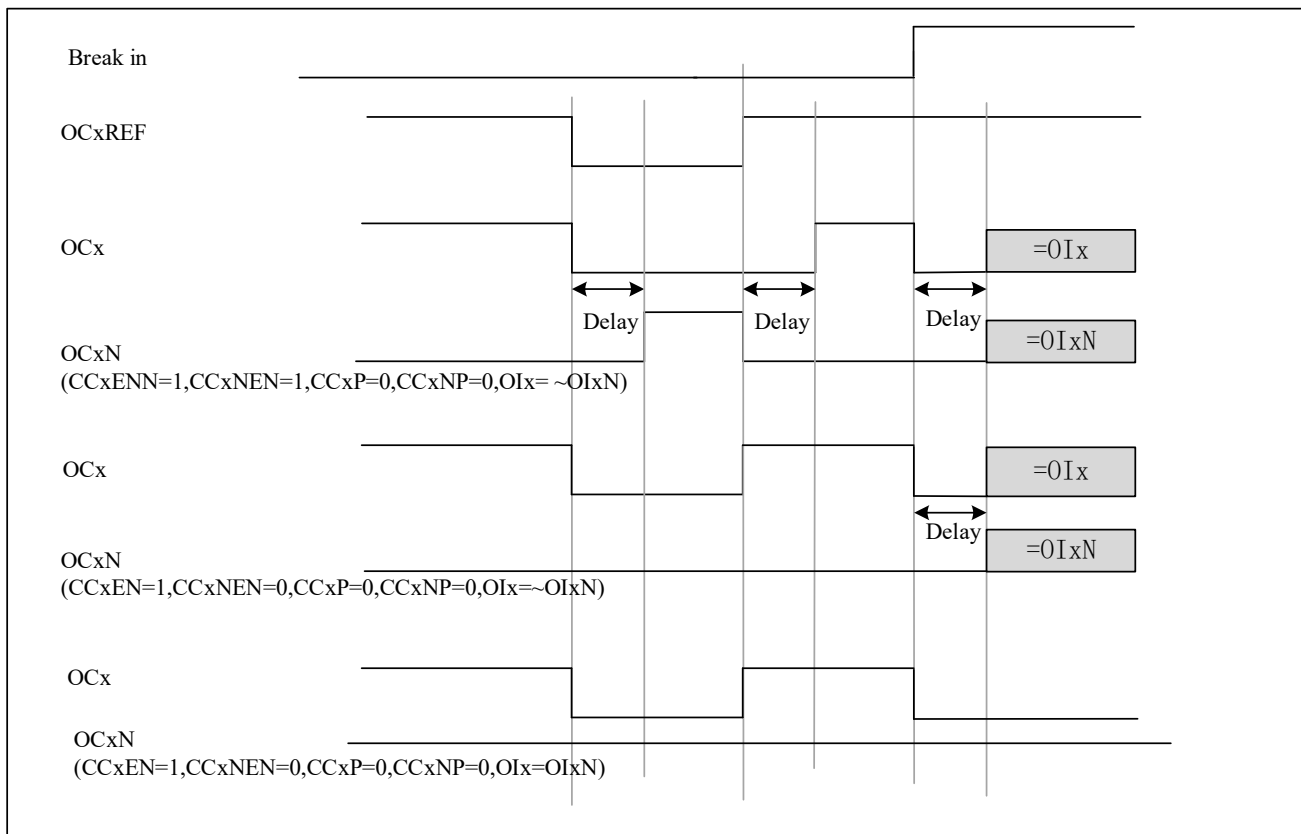
- TIMx_BKDT.MOEN will be cleared asynchronously, then the outputs will be put in inactive state, idle state or reset state. The state of output is selected by setting TIMx_BKDT.OSSI. This will take effect even if the MCU oscillator is off.
- Once TIMx_BKDT.MOEN=0, each output channel will be driven with the level programmed in TIMx_CTRL2.OIx. Timer will release the enable outputs(taken over by GPIO controller) if TIMx_BKDT.OSSI=0, otherwise it will remains high.

- If user choose to use complementary outputs, the behaviors of TIM are as follow
 - The outputs will be set in reset state first depending on the polarity. It is an asynchronous option so it still works even if there is no clock provided to the timer.
 - The dead-time generator will be reactivated if the timer clock is still provided, and drive the outputs according to the value of TIMx_CTRL2.OIx and TIMx_CTRL2.OIxN after the dead-time when $(CCxP \wedge OIx) \wedge (CCxNP \wedge OIxN) = 0$, that is, the OCx and OCxN still cannot be driven to active level at the same time. Note that the dead-time will be longer than usual because of the resynchronization on MOEN (almost 2 cycles of ck_tim).
 - Timer will release the output control if TIMx_BKDT.OSSI=0. Otherwise, if the enable output was high, it will remain high. If it was low, it will become high when TIMx_CCEN.CCxEN or TIMx_CCEN.CCxNEN is high.
- If TIMx_DINTEN.BIEN=1, an interrupt will be generated when TIMx_STS.BITF=1.
- If user set TIMx_BKDT.AOEN, the TIMx_BKDT.MOEN will be set automatically when the next UEV happened. User can use this to regulate. If user did not set TIMx_BKDT.AOEN, the TIMx_BKDT.MOEN will remain low until been set 1 again. At this situation, user can use this for security. User can connect the break input to thermal sensors, alarm for power drivers, or other security components.
- When the break input is active, TIMx_BKDT.MOEN cannot be set automatically or by software at the same time, and the TIMx_STS.BITF cannot be cleared. Because the break inputs are active on level.

To insure the security of application, the break circuit has the write protection function, and there is break input and output management too. It allow user to freeze some parameters, such as dead-time duration, OCx/OCxN polarities and state when disabled, OCxMD configurations, break enable and polarity. User can choose one of the 3 levels of protection to use by setting TIMx_BKDT.LCKCFG. However, the TIMx_BKDT.LCKCFG can only be written once after an MCU reset.

An example for output behavior in response to a break is as follow

Figure 9-27 Output Behavior in Response to A Break



9.3.15 Debug Mode

When the microcontroller is in debug mode (the Cortex[®]-M0 core halted), depending on the DBG_CTRL.TIMx_STOP configuration in the PWR module, the TIMx counter can either continue to operation normally or stop. For more details refer to Section 3.3.2.

9.3.16 Timx and External Trigger Synchronization

TIMx timers can be synchronized by a trigger in slave modes (reset, trigger and gated).

Slave mode: Reset mode

In reset mode, the trigger event can reset the counter and the prescaler updates the preload registers TIMx_AR, TIMx_CCDATx, and generates the update event UEV (TIMx_CTRL1.UPRS=0).

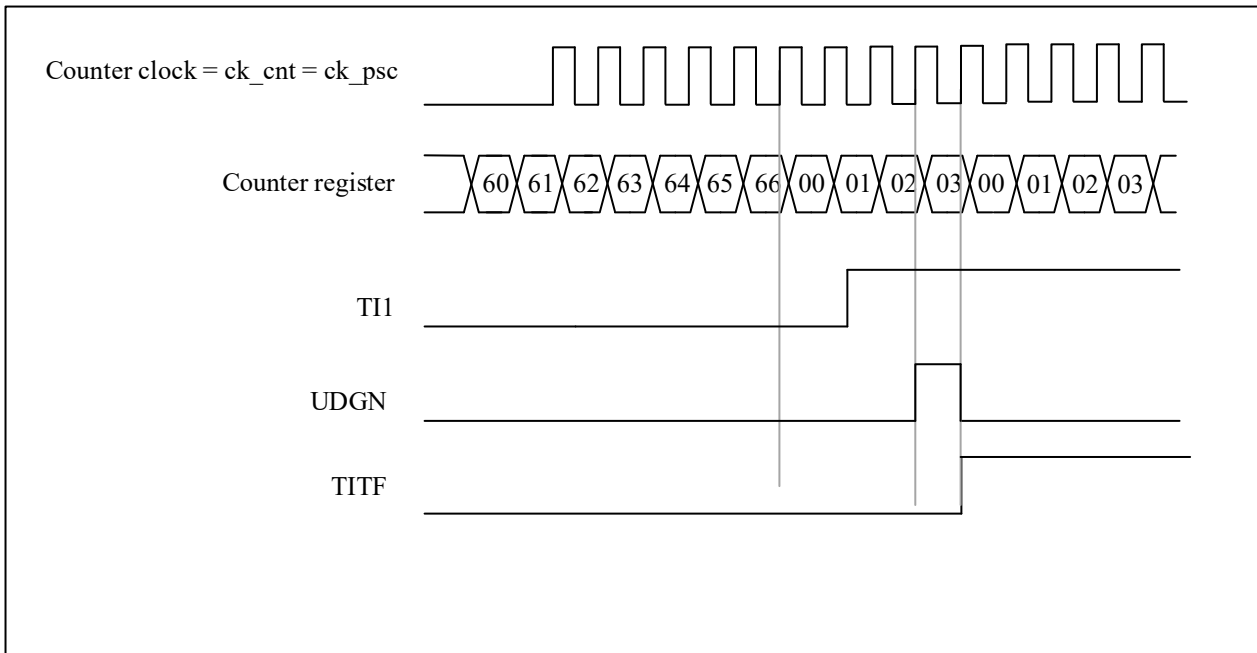
The following is an example of a reset mode:

1. Channel 1 is configured as input to detect the rising edge of TI1 (TIMx_CCMOD1.CC1SEL=01, TIMx_CCEN.CC1P=0);
2. The slave mode is selected as reset mode (TIMx_SMCTRL.SMSEL=100), and the trigger input is selected as TI1 (TIMx_SMCTRL.TSEL=101);
3. Start counter (TIMx_CTRL1.CNTEN = 1).

After starting the timer, when TI1 detects a rising edge, the counter resets and restarts counting, and the trigger flag is set (TIMx_STS.TITF=1);

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 9-28 Control Circuit in Reset Mode



Slave mode: trigger mode

In trigger mode, the trigger event (rising edge/falling edge) of the input port can trigger the counter to start counting.

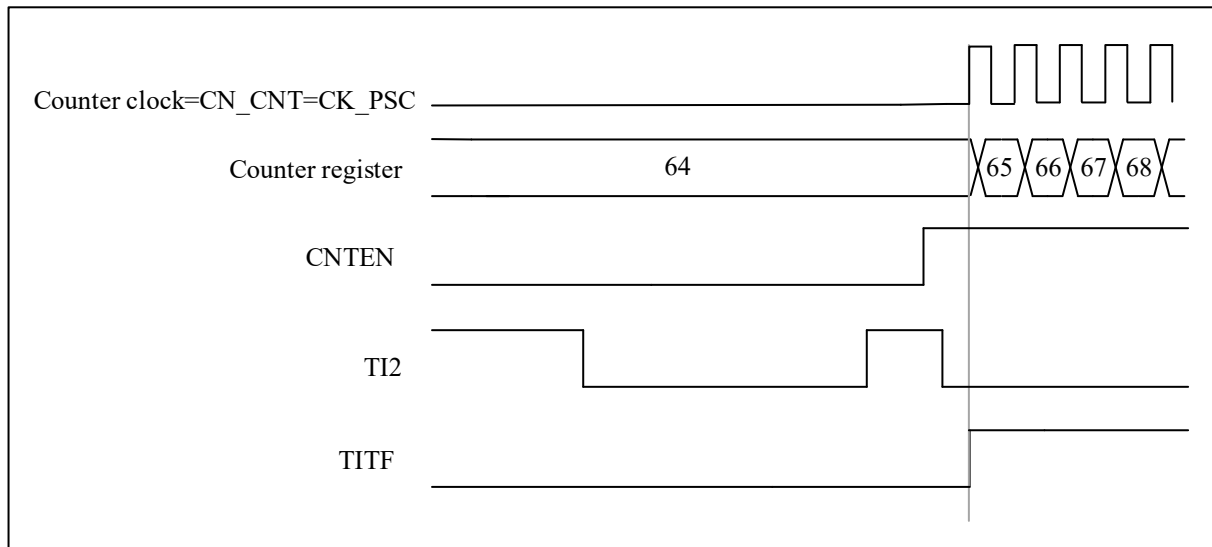
The following is an example of a trigger pattern:

1. Channel 2 is configured as input to detect the rising edge of TI2 (TIMx_CCMOD1.CC2SEL=01, TIMx_CCEN.CC2P=0);
2. Select the slave mode as the trigger mode (TIMx_SMCTRL.SMSEL=110), select TI2 for trigger input (TIMx_SMCTRL.TSEL=110);

When a rising edge is detected on TI2 , the counter starts counting, and the trigger flag is set (TIMx_STS.TITF=1);

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 9-29 Control Circuit in Trigger Mode



Slave mode: gated mode

In gated control mode, the level polarity of the input port can control whether the counter counts or not.

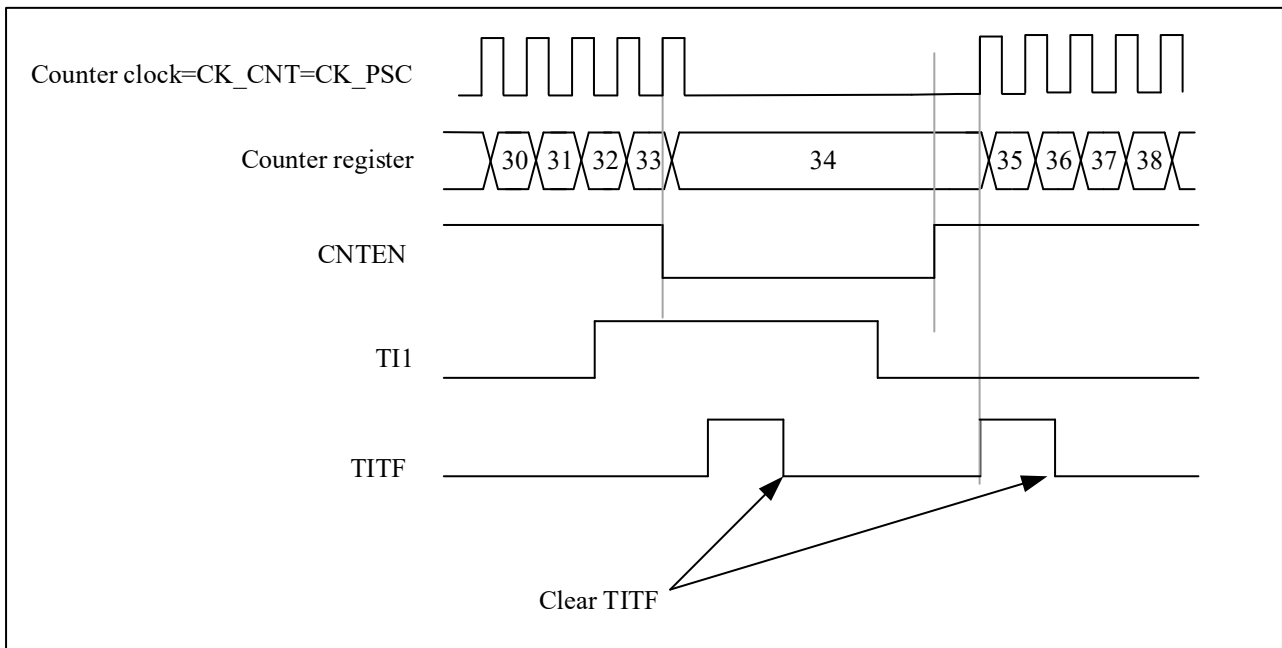
The following is an example of a gated pattern:

1. Channel 1 is configured as input detection active low on TI1 (TIMx_CCMOD1.CC1SEL=01, TIMx_CCEN.CC1P=1);
2. Select the slave mode as the gated mode (TIMx_SMCTRL.SMSEL=101), and select TI1 as the trigger input (TIMx_SMCTRL.TSEL=101);
3. Start counter(TIMx_CTRL1.CNTEN=1).

When TI1 detects that the level changes from low to high, the counter stops counting, and when TI1 detects that the level changes from high to low, the counter starts counting, and the trigger flag will be set (TIMx_STS.TITF=1) when it starts or stops counting;

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 9-30 Control Circuit in Gated Mode



Slave mode: trigger mode + external clock mode 2

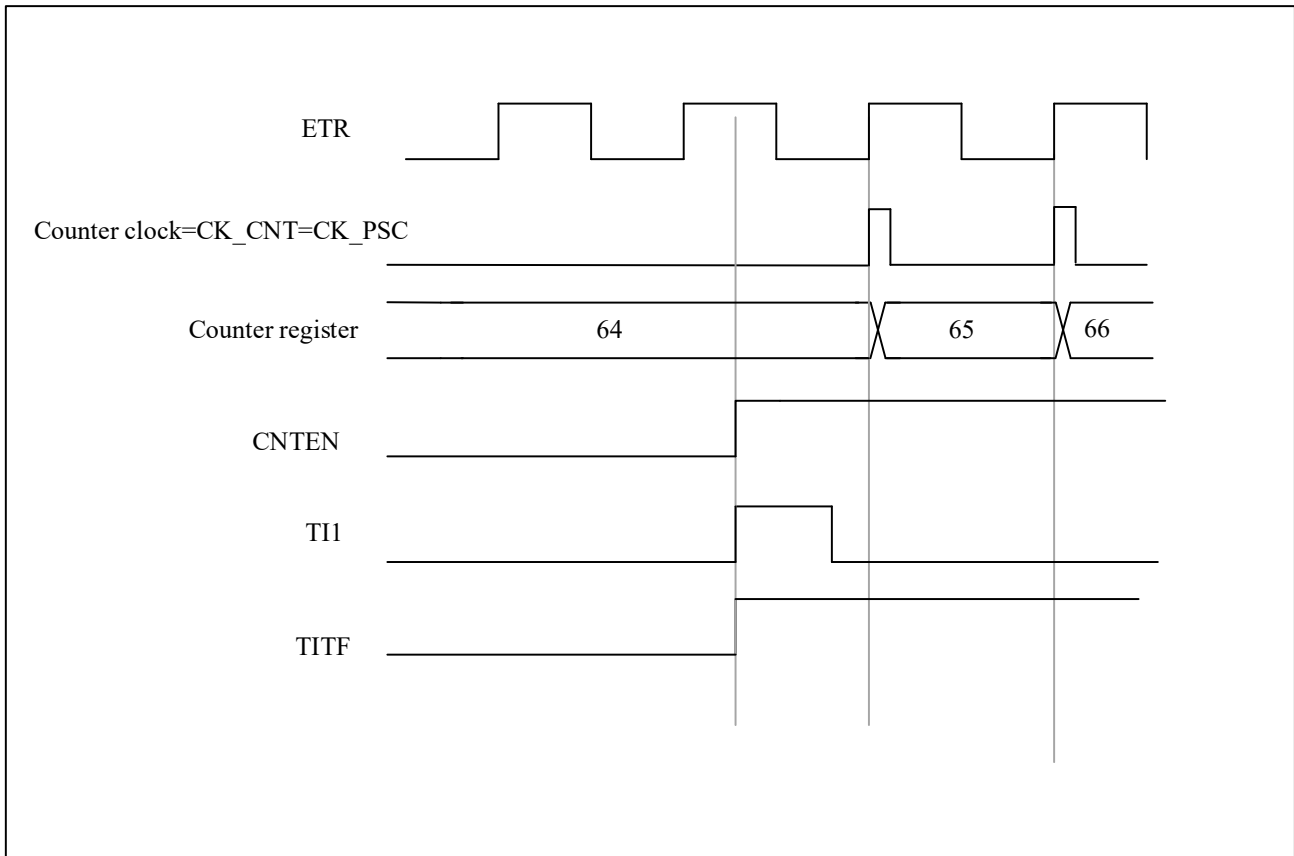
In reset mode, trigger mode and gated mode, the counter clock can be selected as external clock mode 2, and the ETR signal is used as the external clock source input. At this time, the trigger selection needs to select non-ETRF (TIMx_SMCTRL.TSEL=111).

Here is an example:

1. Channel 1 is configured as input to detect the rising edge of TI1 (TIMx_CCMOD1.CC1SEL=01, TIMx_CCEN.CC1P=0);
2. Enable external clock mode 2 (TIMx_SMCTRL.EXCEN=1), select rising edge for external trigger polarity (TIMx_SMCTRL.EXTP=0), select slave mode as trigger mode (TIMx_SMCTRL.SMSEL=110), select TI1 for trigger input (TIMx_SMCTRL.TSEL=101);

When TI1 detects a rising edge, the counter starts counting on the rising edge of ETR, and the trigger flag is set (TIMx_STS.TITF=1);

Figure 9-31 Control Circuit in Trigger Mode + External Clock Mode2



9.3.17 Timer Synchronization

All TIM timers are internally connected for timer synchronization or chaining. For more details, refer to Section 10.3.14.

9.3.18 Generating Six-step PWM Output

In order to modify the configuration of all channels at the same time, the configuration of the next step can be set in advance (the preloaded bits are OCxMD, CCxEN and CCxNEN). When a COM commutation event occurs, the OCxMD, CCxEN, and CCxNEN preload bits are transferred to the shadow register bits.

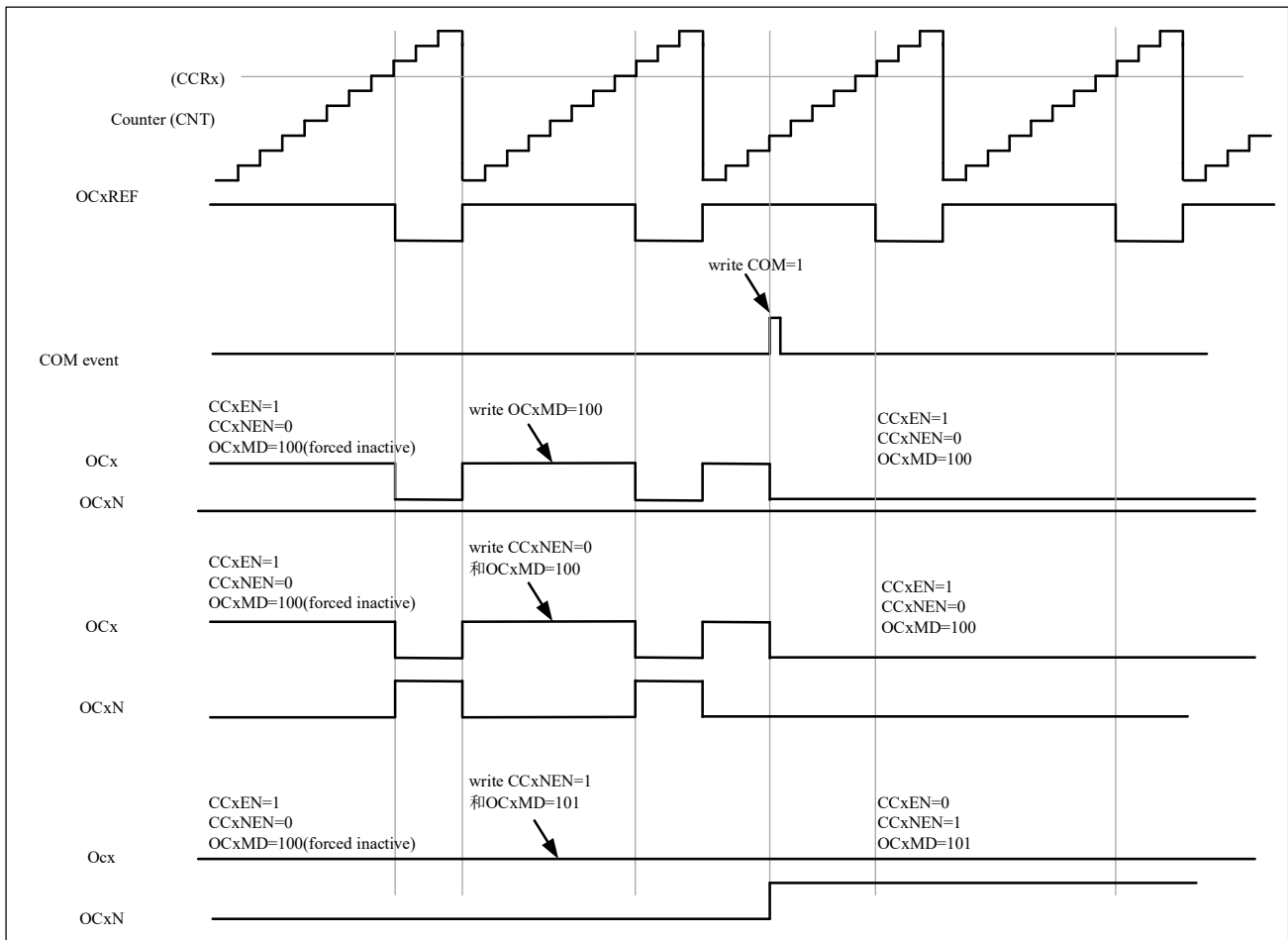
Methods to generate a COM commutation event:

1. Setting TIMx_EVTGEN.CCUDGN by software;
2. Generated by hardware on the rising edge of TRGI;

When a COM commutation event occurs, the TIMx_STS.COMITF flag will be set, enabling interrupts (TIMx_DINTEN.COMIEN) will trigger interrupts, and enabling DMA requests (TIMx_DINTEN.COMDEN) will generate DMA requests.

The following figure shows the output timing diagram of OCx and OCxN when a COM commutation event occurs in three different configurations:

Figure 9-32 6-Step PWM Generation, COM Example (OSSR=1)



9.3.19 Encoder Interface Mode

The encoder uses two inputs TI1 and TI2 as the interface and the counter counts on every edge change on TI1FP1 or TI2FP2. The counting direction is automatically controlled by hardware TIMx_CTRL1.DIR. There are three types of encoder counting modes:

1. The counter only counts on the edge of TI1, TIMx_SMCTRL.SMSEL = '001';
2. The counter only counts on the edge of TI2, TIMx_SMCTRL.SMSEL = '010';
3. The counter counts on the edges of TI1 and TI2 at the same time, TIMx_SMCTRL.SMSEL = '011';

The encoder interface is equivalent to using an external clock with direction selection, and the counter only counts continuously between 0 and the auto-reload value (TIMx_AR.AR [15:0]). Therefore, it is necessary to configure the auto-reload register TIMx_AR in advance.

Note: Encoder mode and external clock mode 2 are not compatible and must not be selected together.

The relationship between the counting direction and the encoder signal is shown in Table 9-1 Counting Direction Versus Encoder Signals:

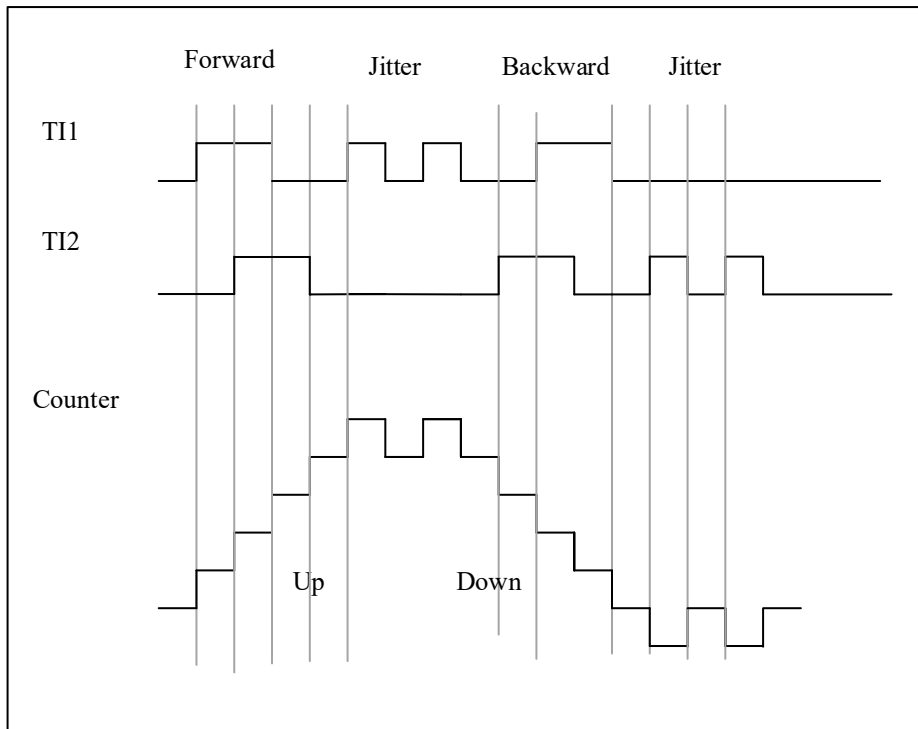
Table 9-1 Counting Direction Versus Encoder Signals

Active Edge	Level On Opposite Signals (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 Signal		TI2FP2 Signal	
		Rising	Falling	Rising	Falling
Counting only at TI1	High	Counting down	Counting up	Don't count	Don't count
	Low	Counting up	Counting down	Don't count	Don't count
Counting only at TI2	High	Don't count	Don't count	Counting up	Counting down
	Low	Don't count	Don't count	Counting down	Counting up
Counting on TI1 and TI2	High	Counting down	Counting up	Counting up	Counting down
	Low	Counting up	Counting down	Counting down	Counting up

Here is an example of an encoder with dual edge select for triggering to suppress input jitter:

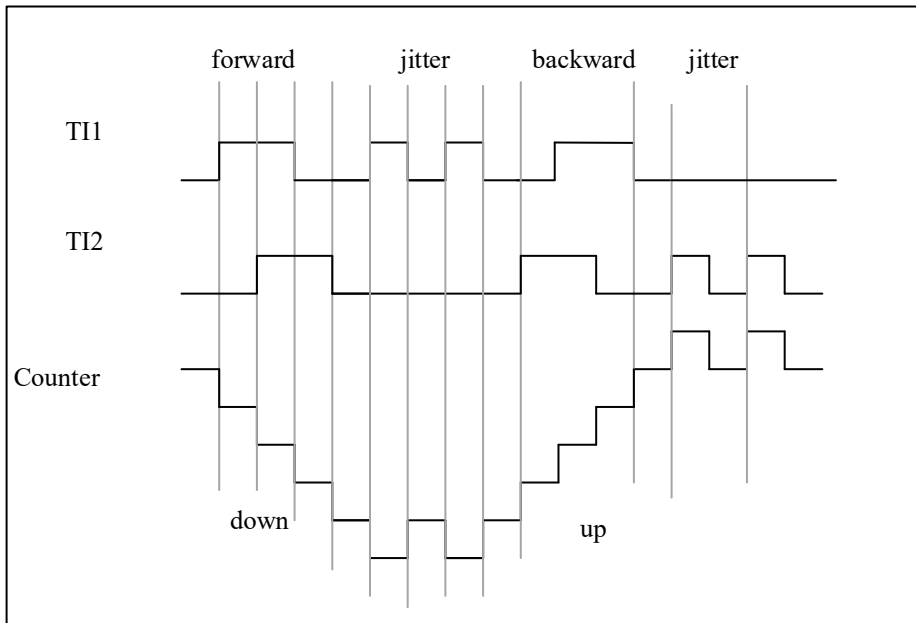
1. IC1FP1 is mapped to TI1 (TIMx_CCMOD1.CC1SEL= '01'), IC1FP1 is not inverted (TIMx_CCEN.CC1P= '0');
2. IC1FP2 is mapped to TI2 (TIMx_CCMOD2.CC2SEL= '01'), IC2FP2 is not inverted (TIMx_CCEN.CC2P= '0');
3. The input is active at both rising and falling edges (TIMx_SMCTRL.SMSEL = '011');
4. Enable counter TIMx_CTRL1.CNTEN= '1';

Figure 9-33 Example of Counter Operation in Encoder Interface Mode



The following figure shows the example of counter behavior when IC1FP1 polarity is inverted (CC1P= '1', other configurations are the same as above)

Figure 9-34 Encoder Interface Mode Example with IC1FP1 Polarity Inverted



9.3.20 Interfacing with Hall Sensor

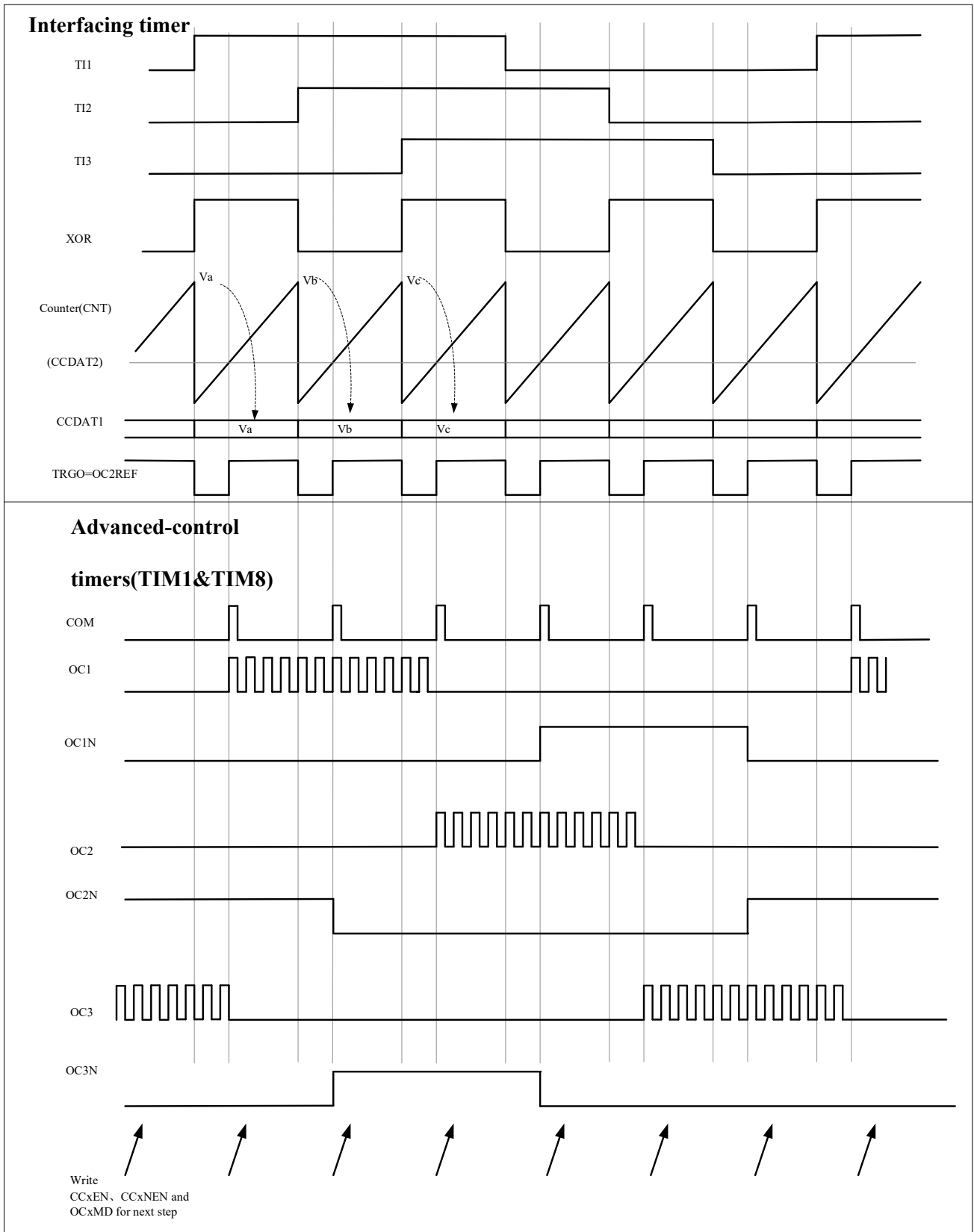
The Hall sensor is connected to the three input pins (CC1, CC2 and CC3) of the timer, then select the XOR function to pass the inputs of TIMx_CH1, TIMx_CH2 and TIMx_CH3 through the XOR gate as the output of TI1 to channel 1 for capture signal.

The timer needs to be configured as the reset mode in slave mode (TIMx_SMCTRL.SMSEL= '100'); the edge of the trigger select TI1 triggers TI1F_ED (TIMx_SMCTRL.TSEL= '100'), any change in the Hall 3 inputs will trigger the counter to recount, so it is used as a time base; the capture/compare channel 1 is configured to capture the TRC signal in capture mode (TIMx_CCMOD1.CC1SEL= '11'), which is used to calculate the two input time intervals, thereby reflecting the motor speed.

Select timer channel 2 to output pulses to the advanced timer, trigger the COM event of the advanced timer, and update the control bits of the output PWM. The trigger selection of the advanced timer needs to select the corresponding internal trigger signal (TIMx_SMCTRL.TSEL="ITRx"), the capture/compare preload control bit needs to be configured to support preload (TIMx_CTRL2.CCPCTL=1) and support the rising edge of TRGI Trigger an update (TIMx_CTRL2.CCUSEL=1).

This example is shown in the following figure.

Figure 9-35 Example of Hall Sensor Interface



9.4 Timx Register Description(x=1, 8)

For abbreviations used in registers, refer to Section 1.1

All register operations must be performed in half word (16-bits) or one word (32-bits).

9.4.1 Register Overview

Table 9-2 Register Overview

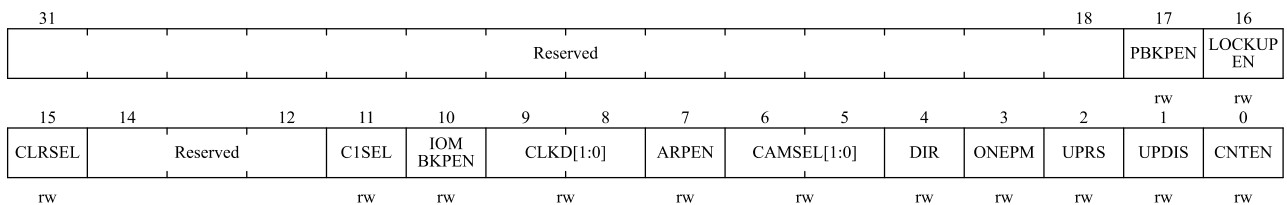
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0									
000h	TIMx_CTRL1	Reserved														PBKPEN	LBKPEN	CLRSEL	Reserved	Reserved	Reserved	CISEL	IOMBKPEN	CLKD[1:0]		ARPEN	CAMSEL[1:0]		DIR	ONEPM	UPRS	UPDIS	CNTEN									
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
004h	TIMx_CTRL2	Reserved														O16	Reserved	O15	Reserved	O14	O13N	O13	O12N	O12	O11N	O11	TI1SEL	MMSEL[2:0]		CCDSEL	CCUSEL	Reserved	CCPCTL									
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
008h	TIMx_SMCTRL	Reserved														EXTP	EXCEN	EXTPS[1:0]		EXTF[3:0]			MSMD	TSEL[2:0]		Reserved	SMSSEL[2:0]															
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00Ch	TIMx_DINTEN	Reserved														TDEN	COMDEN	CC4DEN	CC3DEN	CC2DEN	CC1DEN	UDEN	BIEN	TIEN	COMEN	CC4IEN	CC3IEN	CC2IEN	CC1IEN	UIEN												
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
010h	TIMx_STS	Reserved														CC6ITF	CC5ITF	Reserved		CC4OCF	CC3OCF	CC2OCF	CC1OCF	Reserved	BITF	TITF	COMITF	CC4ITF	CC3ITF	CC2ITF	CC1ITF	UDITF										
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
014h	TIMx_EVTGEN	Reserved														BGN	TGN	CCUDGN	CC4GN	CC3GN	CC2GN	CC1GN	UDGN																			
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0																
018h	TIMx_CCMOD1 Output compare mode	Reserved														OC2CEN	OC2MD[2:0]		OC2PEN	OC2FEN	CC2SEL[1:0]	OC1CEN	OC1MD[2:0]		OC1PEN	OC1FEN	CC1SEL[1:0]															
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0										
018h	TIMx_CCMOD1 Input capture mode	Reserved														IC2F[3:0]		IC2PSC[1:0]	CC2SEL[1:0]	IC1F[3:0]	IC1PSC[1:0]	CC1SEL[1:0]																				
	Reset Value	0														0	0	0	0	0	0	0	0	0	0																	
01Ch	TIMx_CCMOD2 Output compare mode	Reserved														OC4CEN	OC4MD[2:0]		OC4PEN	OC4FEN	CC4SEL[1:0]	OC3CEN	OC3MD[2:0]		OC3PEN	OC3FEN	CC3SEL[1:0]															
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
01Ch	TIMx_CCMOD2 Input capture mode	Reserved														IC4F[3:0]		IC4PSC[1:0]	CC4SEL[1:0]	IC3F[3:0]	IC3PSC[1:0]	CC3SEL[1:0]																				
	Reset Value	0														0	0	0	0	0	0	0	0	0	0																	
020h	TIMx_CCEN	Reserved														CC6P	CC6EN	Reserved	CC5P	CC5EN	Reserved	CC4P	CC4EN	CC3NP	CC3NEN	CC3P	CC3EN	CC2NP	CC2NEN	CC2P	CC2EN	CC1NP	CC1NEN	CC1P	CC1EN							
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
024h	TIMx_CNT	Reserved															CNT[15:0]																
	Reset Value	0															0																
028h	TIMx_PSC	Reserved															PSC[15:0]																
	Reset Value	0															0																
02Ch	TIMx_AR	Reserved															AR[15:0]																
	Reset Value	1															1																
030h	TIMx_REPCNT	Reserved															REPCNT[7:0]																
	Reset Value	0															0																
034h	TIMx_CCDAT1	Reserved															CCDAT1[15:0]																
	Reset Value	0															0																
038h	TIMx_CCDAT2	Reserved															CCDAT2[15:0]																
	Reset Value	0															0																
03Ch	TIMx_CCDAT3	Reserved															CCDAT3[15:0]																
	Reset Value	0															0																
040h	TIMx_CCDAT4	Reserved															CCDAT4[15:0]																
	Reset Value	0															0																
044h	TIMx_BKDT	Reserved															MOEN	AOEN	BKP	BKEN	OSSR	OSSI	LCKCFG[1:0]	DTGN[7:0]									
	Reset Value	0															0	0	0	0	0	0	0	0									
048h	TIMx_DCTRL	Reserved															DBLEN[4:0]				Reserved			DBADDR[4:0]									
	Reset Value	0															0				0			0									
04Ch	TIMx_DADDR	Reserved															BURST[15:0]																
	Reset Value	0															0																
054h	TIMx_CCMOD3	Reserved															OC6CEN	OC6MD[2:0]		OC6PEN	OC6FEN	Reserved			OC5CEN	OC5MD[2:0]		OC5PEN	OC5FEN	Reserved			
	Reset Value	0															0	0		0	0	0			0	0		0	0	0			
058h	TIMx_CCDAT5	Reserved															CCDAT5[15:0]																
	Reset Value	0															0																
05Ch	TIMx_CCDAT6	Reserved															CCDAT6[15:0]																
	Reset Value	0															0																

9.4.2 Control Register 1 (TIMx_CTRL1)

Offset address: 0x00

Reset value: 0x0000 0000



Bit Field	Name	Description
31:18	Reserved	Reserved, the reset value must be maintained
17	PBKPEN	PVD as BKP enable 0: Disable 1: Enable

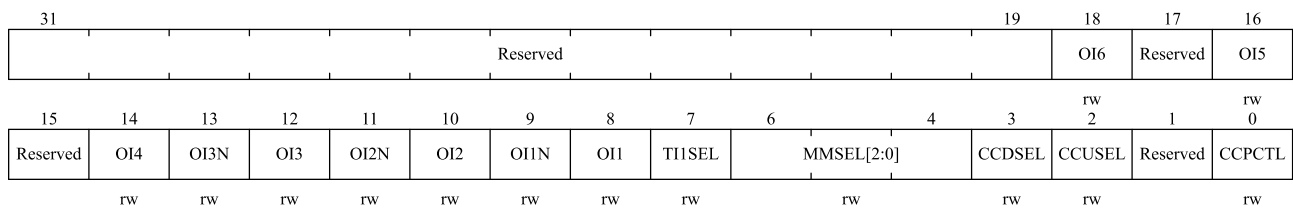
Bit Field	Name	Description
16	LBKPEN	LockUp as BKP enable 0: Disable 1: Enable
15	CLRSEL	OCxREF clear selection 0: Select the external OCxREF clear from ETR 1: Select the internal OCxREF clear from COMP(COMP1/COMP2/COMP3)
14:12	Reserved	Reserved, the reset value must be maintained
11	C1SEL	Channel 1 selection 0: Select external CH1 signal from IOM 1: Select internal CH1 signal from COMP(COMP1/COMP2/COMP3)
10	IOMBKPEN	Enabling IOM as BKP 0: Enable 1: Disable
9:8	CLKD[1:0]	Clock division CLKD[1:0] indicates the division ratio between CK_INT (timer clock) and DTS (clock used for dead-time generator and digital filters (ETR, Tlx)) 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 use this configuration
7	ARPEN	ARPEN: Auto-reload preload enable 0: Shadow register disable for TIMx_AR register 1: Shadow register enable for TIMx_AR register
6:5	CAMSEL[1:0]	Center-aligned mode selection 00: Edge-aligned mode. TIMx_CTRL1.DIR specifies up-counting or down-counting. 01: Center-aligned mode 1. The counter counts in center-aligned mode, and the output compare interrupt flag bit is set to 1 when down-counting. 10: Center-aligned mode 2. The counter counts in center-aligned mode, and the output compare interrupt flag bit is set to 1 when up-counting. 11: Center-aligned mode 3. The counter counts in center-aligned mode, and the output compare interrupt flag bit is set to 1 when up-counting or down-counting. <i>Note: Switching from edge-aligned mode to center-aligned mode is not allowed when the counter is still enabled (TIMx_CTRL1.CNTEN = 1).</i>
4	DIR	Direction 0: Up-counting 1: Down-counting <i>Note: This bit is read-only when the counter is configured in center-aligned mode or encoder mode.</i>
3	ONEPM	One-pulse mode 0: Disable one-pulse mode, the counter counts are not affected when an update event occurs. 1: Enable one-pulse mode, the counter stops counting when the next update event occurs (clearing TIMx_CTRL1.CNTEN bit)
2	UPRS	Update request source

Bit Field	Name	Description
		<p>This bit is used to select the UEV event sources by software.</p> <p>0: If update interrupt or DMA request is enabled, any of the following events will generate an update interrupt or DMA request:</p> <ul style="list-style-type: none"> – Counter overflow/underflow – The TIMx_EVTGEN.UDGN bit is set – Update generation from the slave mode controller <p>1: If update interrupt or DMA request is enabled, only counter overflow/underflow will generate update interrupt or DMA request</p>
1	UPDIS	<p>Update disable</p> <p>This bit is used to enable/disable the Update event (UEV) events generation by software.</p> <p>0: Enable UEV. UEV will be generated if one of following condition been fulfilled:</p> <ul style="list-style-type: none"> – Counter overflow/underflow – The TIMx_EVTGEN.UDGN bit is set – Update generation from the slave mode controller <p>Shadow registers will update with preload value.</p> <p>1: UEV disabled. No update event is generated, and the shadow registers (AR, PSC, and CCDATx) keep their values. If the TIMx_EVTGEN.UDGN bit is set or a hardware reset is issued by the slave mode controller, the counter and prescaler are reinitialized.</p>
0	CNTEN	<p>Counter Enable</p> <p>0: Disable counter</p> <p>1: Enable counter</p> <p><i>Note: external clock, gating mode and encoder mode can only work after TIMx_CTRL1.CNTEN bit is set in the software. Trigger mode can automatically set TIMx_CTRL1.CNTEN bit by hardware.</i></p>

9.4.3 Control Register 2 (TIMx_CTRL2)

Offset address: 0x04

Reset value: 0x0000 0000



Bit Field	Name	Description
31:19	Reserved	Reserved, the reset value must be maintained
18	OI6	Output idle state 6 (OC6 output). See TIMx_CTRL2.OI1 bit.
17	Reserved	Reserved, the reset value must be maintained
16	OI5	Output idle state 5 (OC5 output). See TIMx_CTRL2.OI1 bit.
15	Reserved	Reserved, the reset value must be maintained
14	OI4	Output idle state 4 (OC4 output). See TIMx_CTRL2.OI1 bit.

Bit Field	Name	Description
13	OI3N	Output idle state 3 (OC3N output). See TIMx_CTRL2.OI1N bits.
12	OI3	Output idle state 3 (OC3 output). See TIMx_CTRL2.OI1 bit.
11	OI2N	Output idle state 2 (OC2N output). See TIMx_CTRL2.OI1N bits.
10	OI2	Output idle state 2 (OC2 output). See TIMx_CTRL2.OI1 bit.
9	OI1N	Output Idle state 1 (OC1N Output) 0: When TIMx_BKDT.MOEN = 0, after dead-time OC1N = 0 1: When TIMx_BKDT.MOEN = 0, after dead-time OC1N = 1
8	OI1	Output Idle state 1 0: When TIMx_BKDT.MOEN = 0, if OC1N is implemented, after dead-time OC1 = 0 1: When TIMx_BKDT.MOEN = 0, if OC1N is implemented, after dead-time OC1 = 1
7	TI1SEL	TI1 selection 0: TIMx_CH1 pin connected to TI1 input. 1: TIMx_CH1, TIMx_CH2, and TIMx_CH3 pins are XOR connected to the TI1 input.
6:4	MMSEL[2:0]	Master Mode Selection These 3 bits (TIMx_CTRL2.MMSEL [2:0]) are used to select the synchronization information (TRGO) sent to the slave timer in the master mode. Possible combinations are as follows: 000: Reset –When the TIMx_EVTGEN.UDGN is set or a reset is generated by the slave mode controller, a TRGO pulse occurs. And in the latter case, the signal on TRGO is delayed compared to the actual reset. 001: Enable - The TIMx_CTRL1.CNTEN bit is used as the trigger output (TRGO). Sometimes you need to start multiple timers at the same time or enable slave timer for a period of time. The counter enable signal is set when TIMx_CTRL1.CNTEN bit is set or the trigger input in gated mode is high. 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 (refer to the description of the TIMx_SMCTRL.MSMD bit). 010: Update - The update event is selected as the trigger output (TRGO). For example, a master timer clock can be used as a slave timer prescaler. 011: Compare pulse - Triggers the output to send a positive pulse (TRGO) when the TIMx_STS.CC1ITF is to be set (even if it is already high), when a capture or a comparison succeeds. 100: Compare - OC1REF signal is used as the trigger output (TRGO). 101: Compare - OC2REF signal is used as the trigger output (TRGO). 110: Compare - OC3REF signal is used as the trigger output (TRGO). 111: Compare - OC4REF signal is used as the trigger output (TRGO).
3	CCDSEL	Capture/compare DMA selection 0: When a CCx event occurs, a DMA request for CCx is sent. 1: When an update event occurs, a DMA request for CCx is sent.
2	CCUSEL	Capture/compare control update selection 0: If TIMx_CTRL2.CCPCTL = 1, they can only be updated by setting CCUDGN bits 1: If TIMx_CTRL2.CCPCTL = 1, they can be updated by setting CCUDGN bits or a rising edge on TRGI. <i>Note: This bit only applied to channels with complementary outputs.</i>

Bit Field	Name	Description
1	Reserved	Reserved, the reset value must be maintained
0	CCPCTL	Capture/ Compare preloaded control 0: No preloading of CCxEN, CCxNEN and OCxMD bits occurs. 1: Preloading of CCxEN, CCxNEN and OCxMD bits occurs. they are updated only when a commutation event COM occurs (TIMx_EVTGEN.CCUDGN bit set or rising edge on TRGI depending on CCUSEL bit) <i>Note: This bit only applied to channels with complementary outputs.</i>

9.4.4 Slave Mode Control Register (TIMx_SMCTRL)

Offset address: 0x08

Reset value: 0x0000

15	14	13	12	11	8	7	6	4	3	2	0
EXTP	EXCEN	EXTPS[1:0]		EXTF[3:0]		MSMD		TSEL[2:0]	Reserved		SMSEL[2:0]
rw	rw	rw		rw		rw		rw			rw

Bit Field	Name	Description
15	EXTP	External trigger polarity This bit is used to select whether the trigger operation is to use ETR or the inversion of ETR. 0: ETR active at high level or rising edge. 1: ETR active at low level or falling edge.
14	EXCEN	External clock enable This bit is used to enable external clock mode 2, and the counter is driven by any active edge on the ETRF signal in this mode. 0: External clock mode 2 disable. 1: External clock mode 2 enable. <i>Note 1: When external clock mode 1 and external clock mode 2 are enabled at the same time, the input of the external clock is ETRF.</i> <i>Note 2: The following slave modes can be used simultaneously with external clock mode 2: reset mode, gated mode and trigger mode; However, TRGI cannot connect to ETRF (TIMx_SMCTRL.TSEL ≠ '111').</i> <i>Note 3: Setting the TIMx_SMCTRL.EXCEN bit has the same effect as selecting external clock mode 1 and connecting TRGI to ETRF (TIMx_SMCTRL.SMSEL = 111 and TIMx_SMCTRL.TSEL = 111).</i>
13:12	EXTPS[1:0]	External trigger prescaler The frequency of the external trigger signal ETRP must be at most 1/4 of TIMxCLK frequency. When a faster external clock is input, a prescaler can be used to reduce the frequency of ETRP. 00: Prescaler disable 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8

Bit Field	Name	Description
11:8	EXTF[3:0]	<p>External trigger filter</p> <p>These bits are used to define the frequency at which the ETRP signal is sampled and the bandwidth of the ETRP digital filtering. In effect, the digital filter is an event counter that generates a validate output after consecutive N events are recorded.</p> <p>0000: No filter, sampling at f_{DTS}</p> <p>0001: $f_{SAMPLING} = f_{CK_INT}$, $N = 2$</p> <p>0010: $f_{SAMPLING} = f_{CK_INT}$, $N = 4$</p> <p>0011: $f_{SAMPLING} = f_{CK_INT}$, $N = 8$</p> <p>0100: $f_{SAMPLING} = f_{DTS}/2$, $N = 6$</p> <p>0101: $f_{SAMPLING} = f_{DTS}/2$, $N = 8$</p> <p>0110: $f_{SAMPLING} = f_{DTS}/4$, $N = 6$</p> <p>0111: $f_{SAMPLING} = f_{DTS}/4$, $N = 8$</p> <p>1000: $f_{SAMPLING} = f_{DTS}/8$, $N = 6$</p> <p>1001: $f_{SAMPLING} = f_{DTS}/8$, $N = 8$</p> <p>1010: $f_{SAMPLING} = f_{DTS}/16$, $N = 5$</p> <p>1011: $f_{SAMPLING} = f_{DTS}/16$, $N = 6$</p> <p>1100: $f_{SAMPLING} = f_{DTS}/16$, $N = 8$</p> <p>1101: $f_{SAMPLING} = f_{DTS}/32$, $N = 5$</p> <p>1110: $f_{SAMPLING} = f_{DTS}/32$, $N = 6$</p> <p>1111: $f_{SAMPLING} = f_{DTS}/32$, $N = 8$</p>
7	MSMD	<p>Master/ Slave mode</p> <p>0: No action</p> <p>1: Events on the trigger input (TRGI) are delayed to allow a perfect synchronization between the current timer (via TRGO) and its slaves. This is useful when several timers are required to be synchronized to a single external event.</p>
6:4	TSEL[2:0]	<p>Trigger selection</p> <p>These 3 bits are used to select the trigger input of the synchronous counter.</p> <p>000: Internal trigger 0 (ITR0) 100: TI1 edge detector (TI1F_ED)</p> <p>001: Internal trigger 1 (ITR1) 101: Filtered timer input 1 (TI1FP1)</p> <p>010: Internal trigger 2 (ITR2) 110: Filtered timer input 2 (TI2FP2)</p> <p>011: Internal trigger 3 (ITR3) 111: External triggered Input (ETRF)</p> <p>For more details on ITRx, refer to Table 9-3 below.</p> <p><i>Note: These bits must be changed only when not in use (e. g. TIMx_SMCTRL.SMSEL=000) to avoid false edge detection at the transition.</i></p>
3	Reserved	Reserved, the reset value must be maintained

Bit Field	Name	Description
2:0	SMSEL[2:0]	<p>Slave mode selection</p> <p>When an external signal is selected, the active edge of the trigger signal (TRGI) is linked to the selected external input polarity (refer to input control register and control register description)</p> <p>000: Disable slave mode. If TIMx_CTRL1.CNTEN = 1, the prescaler is driven directly by the internal clock.</p> <p>001: Encoder mode 1. According to the level of TI2FP2, the counter up-counting or down-counting on the edge of TI1FP1.</p> <p>010: Encoder mode 2. According to the level of TI1FP1, the counter up-counting or down-counting on the edge of TI2FP2.</p> <p>011: Encoder mode 3. According to the input level of another signal, the counter up-counting or down-counting on the edges of TI2FP1 and TI2FP2.</p> <p>100: Reset mode. On the rising edge of the selected trigger input (TRGI), the counter is reinitialized and the shadow register is updated.</p> <p>101: Gated mode. When the trigger input (TRGI) is high, the clock of the counter is enabled. Once the trigger input becomes low, the counter stops counting, but is not reset. In this mode, the start and stop of the counter are controlled.</p> <p>110: Trigger mode. When a rising edge occurs on the trigger input (TRGI), the counter is started but not reset. In this mode, only the start of the counter is controlled.</p> <p>111: External clock mode 1. The counter is clocked by the rising edge of the selected trigger input (TRGI).</p> <p><i>Note: Do not use gated mode if TI1F_ED is selected as the trigger input (TIMx_SMCTRL.TSEL=100). This is because TI1F_ED outputs a pulse for each TI1F transition, whereas gated mode checks the level of the triggered input.</i></p>

Table 9-3 TIMx Internal Trigger Connection

Slave Timer	ITR0 (TSEL = 000)	ITR1 (TSEL = 001)	ITR2 (TSEL = 010)	ITR3 (TSEL = 011)
TIM1	NA	NA	TIM3	TIM4
TIM8	TIM1	NA	TIM4	NA

9.4.5 DMA/Interrupt Enable Registers (TIMx_DINTEN)

Offset address: 0x0C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TDEN	COMDEN	CC4DEN	CC3DEN	CC2DEN	CC1DEN	UDEN	BIEN	TIEN	COMIEN	CC4IEN	CC3IEN	CC2IEN	CC1IEN	UIEN
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

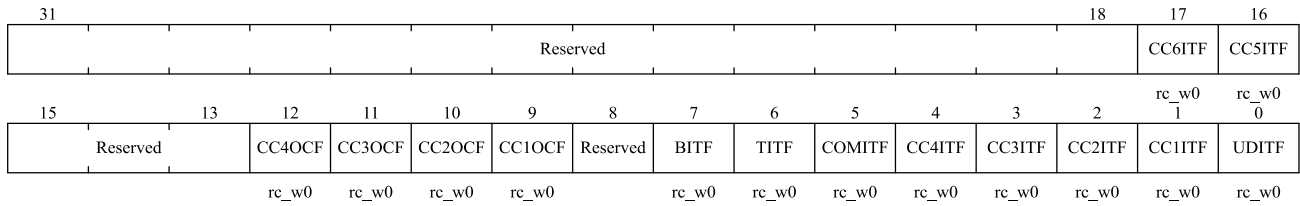
Bit Field	Name	Description
15	Reserved	Reserved, the reset value must be maintained
14	TDEN	<p>Trigger DMA request enable</p> <p>0: Disable trigger DMA request</p> <p>1: Enable trigger DMA request</p>

Bit Field	Name	Description
13	COMDEN	COM DMA request enable 0: Disable COM DMA request 1: Enable COM DMA request
12	CC4DEN	Capture/Compare 4 DMA request enable 0: Disable capture/compare 4 DMA request 1: Enable capture/compare 4 DMA request
11	CC3DEN	Capture/Compare 3 DMA request enable 0: Disable capture/compare 3 DMA request 1: Enable capture/compare 3 DMA request
10	CC2DEN	Capture/Compare 2 DMA request enable 0: Disable capture/compare 2 DMA request 1: Enable capture/compare 2 DMA request
9	CC1DEN	Capture/Compare 1 DMA request enable 0: Disable capture/compare 1 DMA request 1: Enable capture/compare 1 DMA request
8	UDEN	Update DMA request enable 0: Disable update DMA request 1: Enable update DMA request
7	BIEN	Break interrupt enable 0: Disable break interrupt 1: Enable break interrupt
6	TIEN	Trigger interrupt enable 0: Disable trigger interrupt 1: Enable trigger interrupt
5	COMIEN	COM interrupt enable 0: Disable COM interrupt 1: Enable COM interrupt
4	CC4IEN	Capture/Compare 4 interrupt enable 0: Disable capture/compare 4 interrupt 1: Enable capture/compare 4 interrupt
3	CC3IEN	Capture/Compare 3 interrupt enable 0: Disable capture/compare 3 interrupt 1: Enable capture/compare 3 interrupts
2	CC2IEN	Capture/Compare 2 interrupt enable 0: Disable capture/compare 2 interrupt 1: Enables capture/compare 2 interrupts
1	CC1IEN	Capture/Compare 1 interrupt enable 0: Disable capture/compare 1 interrupt 1: Enables capture/comparing 1 interrupt
0	UIEN	Update interrupt enable 0: Disable update interrupt 1: Enables update interrupt

9.4.6 Status Registers (TIMx_STS)

Offset address: 0x10

Reset value: 0x0000 0000



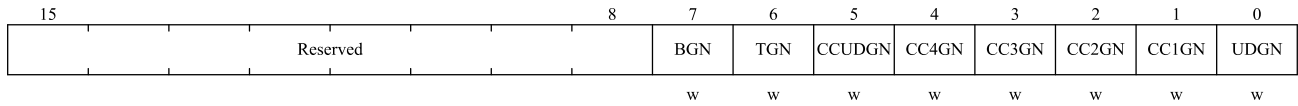
Bit Field	Name	Description
31: 18	Reserved	Reserved, the reset value must be maintained
17	CC6ITF	Capture/Compare 6 interrupt flag See TIMx_STS.CC1ITF description.
16	CC5ITF	Capture/Compare 5 interrupt flag See TIMx_STS.CC1ITF description.
15: 13	Reserved	Reserved, the reset value must be maintained
12	CC4OCF	Capture/Compare 4 overcapture flag See TIMx_STS.CC1OCF description.
11	CC3OCF	Capture/Compare 3 overcapture flag See TIMx_STS.CC1OCF description.
10	CC2OCF	Capture/Compare 2 overcapture flags See TIMx_STS.CC1OCF description.
9	CC1OCF	Capture/Compare 1 overcapture flag This bit is set by hardware only when the corresponding channel is configured in input capture mode. Cleared by software writing 0. 0: No overcapture occurred 1: TIMx_STS.CC1ITF was already set when the value of the counter has been captured in the TIMx_CCDAT1 register.
8	Reserved	Reserved, the reset value must be maintained
7	BITF	Break interrupt flag This bit is set by hardware once the brake input is active. This bit is cleared by software when the brake input becomes inactive. 0: No break event occurred 1: An active level has been detected
6	TITF	Trigger interrupt flag This bit is set by hardware when an active edge is detected on the TRGI input when the slave mode controller is in a mode other than gated. This bit is set by hardware when any edge in gated mode is detected. This bit is cleared by software. 0: No trigger event occurred 1: Trigger interrupt occurred

Bit Field	Name	Description
5	COMITF	<p>COM interrupt flag</p> <p>This bit is set by hardware once a COM event is generated (when TIMx_CCEN.CCxEN, TIMx_CCEN.CCxNEN, TIMx_CCMOD1.OCxMD have been updated). This bit is cleared by software.</p> <p>0: No COM event occurred</p> <p>1: COM interrupt pending</p>
4	CC4ITF	<p>Capture/Compare 4 interrupt flag</p> <p>See TIMx_STS.CC1ITF description.</p>
3	CC3ITF	<p>Capture/Compare 3 interrupt flag</p> <p>See TIMx_STS.CC1ITF description.</p>
2	CC2ITF	<p>Capture/Compare 2 interrupt flag</p> <p>See TIMx_STS.CC1ITF description.</p>
1	CC1ITF	<p>Capture/Compare 1 interrupt flag</p> <p>When the corresponding channel of CC1 is in output mode:</p> <p>Except in center-aligned mode, this bit is set by hardware when the counter value is the same as the compare value (refer to TIMx_CTRL1.CAMSEL bit description). This bit is cleared by software.</p> <p>0: No match occurred.</p> <p>1: The value of TIMx_CNT is the same as the value of TIMx_CCDAT1.</p> <p>When the value of TIMx_CCDAT1 is greater than the value of TIMx_AR, the TIMx_STS.CC1ITF bit will go high if the counter overflows (in up-counting and up/down-counting modes) and underflows in down-counting mode.</p> <p>When the corresponding channel of CC1 is in input mode:</p> <p>This bit is set by hardware when the capture event occurs. This bit is cleared by software or by reading TIMx_CCDAT1.</p> <p>0: No input capture occurred.</p> <p>1: Input capture occurred. Counter value has captured in the TIMx_CCDAT1. An edge with the same polarity as selected has been detected on IC1.</p>
0	UDITF	<p>Update interrupt flag</p> <p>This bit is set by hardware when an update event occurs under the following conditions:</p> <ul style="list-style-type: none"> – When TIMx_CTRL1.UPDIS = 0, and repeat counter value overflow or underflow (An update event is generated when the repeat counter equals 0). – When TIMx_CTRL1.UPRS = 0, TIMx_CTRL1.UPDIS = 0, and set the TIMx_EVTGEN.UDGN bit by software to reinitialize the CNT. – When TIMx_CTRL1.UPRS = 0, TIMx_CTRL1.UPDIS = 0, and the counter CNT is reinitialized by the trigger event. (See TIMx_SMCTRL Register description) <p>This bit is cleared by software.</p> <p>0: No update event occurred</p> <p>1: Update interrupt occurred</p>

9.4.7 Event Generation Registers (TIMx_EVTGEN)

Offset address: 0x14

Reset values: 0x0000



Bit Field	Name	Description
15: 8	Reserved	Reserved, the reset value must be maintained
7	BGN	<p>Break generation</p> <p>This bit can generate a brake event when set by software. And at this time TIMx_BKDT.MOEN = 0, TIMx_STS.BITF = 1, if the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated. This bit is automatically cleared by hardware.</p> <p>0: No action 1: Generated a break event</p>
6	TGN	<p>Trigger generation</p> <p>This bit can generate a trigger event when set by software. And at this time TIMx_STS.TITF = 1, if the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated. This bit is automatically cleared by hardware.</p> <p>0: No action 1: Generated a trigger event</p>
5	CCUDGN	<p>Capture/Compare control update generation</p> <p>This bit is set by software. And if TIMx_CTRL2.CCPCTL = 1 at this time, the CCxEN, CCxNEN and OCxMD bits are allowed to be updated. This bit is automatically cleared by hardware.</p> <p>0: No action 1: Generated a COM event</p> <p><i>Note: This bit is only valid for channels with complementary outputs.</i></p>
4	CC4GN	<p>Capture/Compare 4 generation</p> <p>See TIMx_EVTGEN.CC1GN description.</p>
3	CC3GN	<p>Capture/Compare 3 generation</p> <p>See TIMx_EVTGEN.CC1GN description.</p>
2	CC2GN	<p>Capture/Compare 2 generation</p> <p>See TIMx_EVTGEN.CC1GN description.</p>
1	CC1GN	<p>Capture/Compare 1 generation</p> <p>This bit can generate a capture/compare event when set by software. This bit is automatically cleared by hardware.</p> <p>When the corresponding channel of CC1 is in output mode: The TIMx_STS.CC1ITF flag will be pulled high, if the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated.</p> <p>When the corresponding channel of CC1 is in input mode: TIMx_CC1DAT1 will capture the current counter value, and the TIMx_STS.CC1ITF flag will be pulled high, if the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated. If The TIMx_STS.CC1ITF is already pulled high, pull TIMx_STS.CC1OCF high.</p> <p>0: No action 1: Generated a CC1 capture/compare event</p>

Bit Field	Name	Description
0	UDGN	Update generation This bit can generate an update event when set by software. And at this time the counter will be reinitialized, the prescaler counter will be cleared, the counter will be cleared in center-aligned or up-counting mode, but take TIMx_AR in down-counting mode the value of the register. This bit is automatically cleared by hardware. 0: No action 1: Generated an update event

9.4.8 Capture/Compare Mode Register 1 (TIMx_CCMOD1)

Offset address: 0x18

Reset value: 0x0000

Channels can be used for input (capture mode) or output (compare mode), and the direction of the channel is defined by the corresponding CCxSEL bit. The other bits of the register act differently in input and output modes. OCx describes the function of a channel in output mode, ICx describes the function of a channel in input mode. Hence, please note that the same bit can have different meanings for output mode and for input mode.

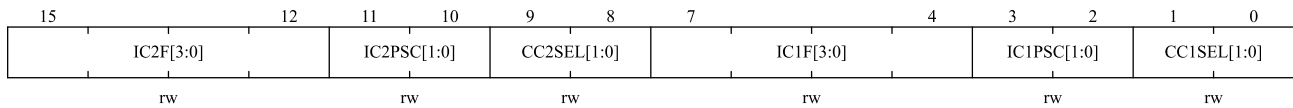
Output compare mode:

15	14	12	11	10	9	8	7	6	4	3	2	1	0
OC2CEN	OC2M[2:0]		OC2PEN	OC2FEN	CC2SEL[1:0]		OC1CEN	OC1M[2:0]		OC1PEN	OC1FEN	CC1SEL[1:0]	
rw	rw		rw	rw	rw		rw	rw		rw	rw	rw	

Bit Field	Name	Description
15	OC2CEN	Output Compare 2 clear enable
14:12	OC2MD[2:0]	Output Compare 2 mode
11	OC2PEN	Output Compare 2 preload enable
10	OC2FEN	Output Compare 2 fast enable
9:8	CC2SEL[1:0]	Capture/compare 2 selection These bits are used to select the input/output and input mapping of the channel 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 only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC2SEL is writable only when the channel is off (TIMx_CCEN.CC2EN = 0).</i>
7	OC1CEN	Output Compare 1 clear enable 0: OC1REF is not affected by ETRF input level 1: OC1REF is cleared immediately when the ETRF input level is detected as high
6:4	OC1MD[2:0]	Output Compare 1 mode These bits are used to manage the output reference signal OC1REF, which determines the values of OC1 and OC1N, and is valid at high levels, while the active levels of OC1 and OC1N depend on the TIMx_CCEN.CC1P and TIMx_CCEN.CC1NP bits. 000: Frozen. Comparison between TIMx_CCDAT1 register and counter TIMx_CNT has no effect on OC1REF signal.

Bit Field	Name	Description
		<p>001: Set channel 1 to the active level on match. When TIMx_CCDAT1 = TIMx_CNT, OC1REF signal will be forced high.</p> <p>010: Set channel 1 as inactive level on match. When TIMx_CCDAT1 = TIMx_CNT, OC1REF signal will be forced low.</p> <p>011: Toggle. When TIMx_CCDAT1 = TIMx_CNT, OC1REF signal will be toggled.</p> <p>100: Force to inactive level. OC1REF signal is forced low.</p> <p>101: Force to active level. OC1REF signal is forced high.</p> <p>110: PWM mode 1 - In up-counting mode, if TIMx_CNT < TIMx_CCDAT1, OC1REF signal of channel 1 is high, otherwise it is low. In down-counting mode, if TIMx_CNT > TIMx_CCDAT1, OC1REF signal of channel 1 is low, otherwise it is high.</p> <p>111: PWM mode 2 - In up-counting mode, if TIMx_CNT < TIMx_CCDAT1, OC1REF signal of channel 1 is low, otherwise it is high. In down-counting mode, if TIMx_CNT > TIMx_CCDAT1, OC1REF signal of channel 1 is high, otherwise it is low.</p> <p><i>Note 1: In PWM mode 1 or PWM mode 2, the OC1REF level changes only when the comparison result changes or when the output compare mode is switched from frozen mode to PWM mode.</i></p>
3	OC1PEN	<p>Output Compare 1 preload enable</p> <p>0: Disable preload function of TIMx_CCDAT1 register. Supports write operations to TIMx_CCDAT1 register at any time, and the written value is effective immediately.</p> <p>1: Enable preload function of TIMx_CCDAT1 register. Only read and write operations to preload registers. When an update event occurs, the value of TIMx_CCDAT1 is loaded into the active register.</p> <p><i>Note 1: Only when TIMx_CTRL1.ONEPM = 1 (In one-pulse mode), PWM mode can be used without verifying the preload register, otherwise no other behavior can be predicted.</i></p>
2	OC1FEN	<p>Output Compare 1 fast enable</p> <p>This bit is used to speed up the response of the CC output to the trigger input event.</p> <p>0: CC1 behaves normally depending on the counter and CCDAT1 values, even if the trigger is ON. The minimum delay for activating CC1 output when an edge occurs on the trigger input is 5 clock cycles.</p> <p>1: An active edge of the trigger input acts like a comparison match on CC1 output. Therefore, OC is set to the comparison level regardless of the comparison result. The delay time for sampling the trigger input and activating the CC1 output is reduced to 3 clock cycles.</p> <p>OCxFEN only works if the channel is configured in PWM1 or PWM2 mode.</p>
1: 0	CC1SEL[1:0]	<p>Capture/compare 1 selection</p> <p>These bits are used to select the input/output and input mapping of the channel</p> <p>00: CC1 channel is configured as output</p> <p>01: CC1 channel is configured as input, IC1 is mapped on TI1</p> <p>10: CC1 channel is configured as input, IC1 is mapped on TI2</p> <p>11: CC1 channels are configured as inputs and IC1 is mapped to TRC. This mode is only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL.</p> <p><i>Note: CC1SEL is writable only when the channel is off (TIMx_CCEN.CC1EN = 0).</i></p>

Input capture mode:



Bit Field	Name	Description
15:12	IC2F[3:0]	Input Capture 2 Filter
11:10	IC2PSC[1:0]	Input Capture 2 Prescaler
9:8	CC2SEL[1:0]	Capture/Compare 2 selection These bits are used to select the input/output and input mapping of the channel 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 only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC2SEL is writable only when the channel is off (TIMx_CCEN.CC2EN = 0).</i>
7:4	IC1F[3:0]	Input Capture 1 filter These bits are used to define sampling frequency of TI1 input and the length of digital filter. The digital filter is an event counter that generates an output transition after N events are recorded. 0000: No filter, sampling at f_{DTS} frequency 0001: $f_{SAMPLING} = f_{CK_INT}$, $N = 2$ 0010: $f_{SAMPLING} = f_{CK_INT}$, $N = 4$ 0011: $f_{SAMPLING} = f_{CK_INT}$, $N = 8$ 0100: $f_{SAMPLING} = f_{DTS}/2$, $N = 6$ 0101: $f_{SAMPLING} = f_{DTS}/2$, $N = 8$ 0110: $f_{SAMPLING} = f_{DTS}/4$, $N = 6$ 0111: $f_{SAMPLING} = f_{DTS}/4$, $N = 8$ 1000: $f_{SAMPLING} = f_{DTS}/8$, $N = 6$ 1001: $f_{SAMPLING} = f_{DTS}/8$, $N = 8$ 1010: $f_{SAMPLING} = f_{DTS}/16$, $N = 5$ 1011: $f_{SAMPLING} = f_{DTS}/16$, $N = 6$ 1100: $f_{SAMPLING} = f_{DTS}/16$, $N = 8$ 1101: $f_{SAMPLING} = f_{DTS}/32$, $N = 5$ 1110: $f_{SAMPLING} = f_{DTS}/32$, $N = 6$ 1111: $f_{SAMPLING} = f_{DTS}/32$, $N = 8$
3:2	IC1PSC[1:0]	Input Capture 1 prescaler These bits are used to select the ratio of the prescaler for IC1 (CC1 input). When TIMx_CCEN.CC1EN = 0, the prescaler will be reset. 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	CC1SEL[1:0]	Capture/Compare 1 selection These bits are used to select the input/output and input mapping of the channel

Bit Field	Name	Description
		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 to TRC. This mode is only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC1SEL is writable only when the channel is off (TIMx_CCEN.CC1EN = 0).</i>

9.4.9 Capture/Compare Mode Register 2 (TIMx_CCMOD2)

Offset address: 0x1C

Reset value: 0x0000

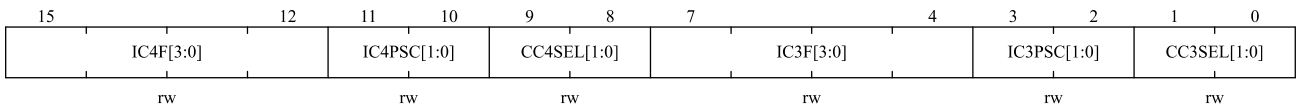
See the description of the CCMOD1 register above

Output comparison mode:

15	14	12	11	10	9	8	7	6	4	3	2	1	0
OC4CEN	OC4MD[2:0]	OC4PEN	OC4FEN	CC4SEL[1:0]	OC3CEN	OC3MD[2:0]	OC3PEN	OC3FEN	CC3SEL[1:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit Field	Name	Description
15	OC4CEN	Output compare 4 clear enable
14:12	OC4MD[2:0]	Output compare 4 mode
11	OC4PEN	Output compare 4 preload enable
10	OC4FEN	Output compare 4 fast enable
9:8	CC4SEL[1:0]	Capture/Compare 4 selection These bits are used to select the input/output and input mapping of the channel 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 only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC4SEL is writable only when the channel is off (TIMx_CCEN.CC4EN = 0).</i>
7	OC3CEN	Output compare 3 clear enable
6:4	OC3MD[2:0]	Output compare 3 mode
3	OC3PEN	Output compare 3 preload enable
2	OC3FEN	Output compare 3 fast enable
1:0	CC3SEL[1:0]	Capture/Compare 3 selection These bits are used to select the input/output and input mapping of the channel 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped to TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped to TRC. This mode is only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC3SEL is writable only when the channel is off (TIMx_CCEN.CC3EN = 0).</i>

Input capture mode:

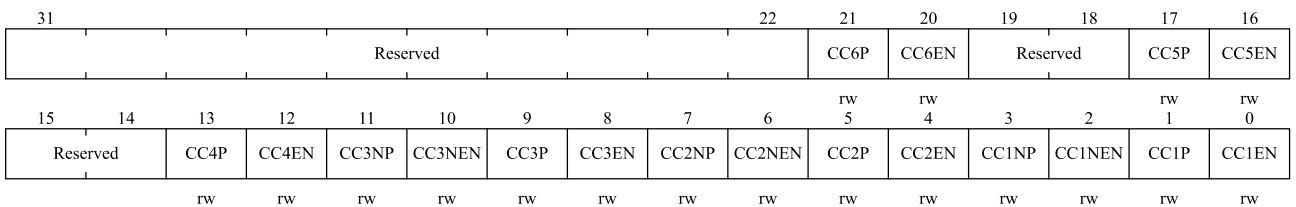


Bit Field	Name	Description
15:12	IC4F[3:0]	Input Capture 4 filter
11:10	IC4PSC[1:0]	Input Capture 4 Prescaler
9:8	CC4SEL[1:0]	Capture/Compare 4 selection These bits are used to select the input/output and input mapping of the channel 00: CC4 channel is configured as output 01: CC4 channel is configured 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 only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC4SEL is writable only when the channel is off (TIMx_CCEN.CC4EN = 0).</i>
7:4	IC3F[3:0]	Input Capture 3 filter
3:2	IC3PSC[1:0]	Input Capture 3 Prescaler
1:0	CC3SEL[1:0]	Capture/compare 3 selection These bits are used to select the input/output and input mapping of the channel 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped to TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped to TRC. This mode is only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC3SEL is writable only when the channel is off (TIMx_CCEN.CC3EN = 0).</i>

9.4.10 Capture/Compare Enable Registers (TIMx_CCEN)

Offset address: 0x20

Reset value: 0x0000 0000



Bit Field	Name	Description
31:22	Reserved	Reserved, the reset value must be maintained
21	CC6P	Capture/Compare 6 output polarity See TIMx_CCEN.CC1P description.
20	CC6EN	Capture/Compare 6 output enable See TIMx_CCEN.CC1EN description.
19: 18	Reserved	Reserved, the reset value must be maintained

Bit Field	Name	Description
17	CC5P	Capture/Compare 5 output polarity See TIMx_CCEN.CC1P description.
16	CC5EN	Capture/Compare 5 output enable See TIMx_CCEN.CC1EN description.
15:14	Reserved	Reserved, the reset value must be maintained
13	CC4P	Capture/Compare 4 output polarity See TIMx_CCEN.CC1P description.
12	CC4EN	Capture/Compare 4 output enable See TIMx_CCEN.CC1EN description.
11	CC3NP	Capture/Compare 3 Complementary output polarity See TIMx_CCEN.CC1NP description.
10	CC3NEN	Capture/Compare 3 complementary output enable See TIMx_CCEN.CC1NEN description.
9	CC3P	Capture/Compare 3 output polarity See TIMx_CCEN.CC1P description.
8	CC3EN	Capture/Compare 3 output enable See TIMx_CCEN.CC1EN description.
7	CC2NP	Capture/Compare 2 complementary output polarity See TIMx_CCEN.CC1NP description.
6	CC2NEN	Capture/Compare 2 complementary output enable See TIMx_CCEN.CC1NEN description.
5	CC2P	Capture/Compare 2 output polarity See TIMx_CCEN.CC1P description.
4	CC2EN	Capture/Compare 2 output enable See TIMx_CCEN.CC1EN description.
3	CC1NP	Capture/Compare 1 complementary output polarity 0: OC1N active high 1: OC1N active low
2	CC1NEN	Capture/Compare 1 complementary output enable 0: Disable - Disable output OC1N signal. The level of OC1N depends on the value of these bits TIMx_BKDT.MOEN, TIMx_BKDT.OSSI, TIMx_BKDT.OSSR, TIMx_CTRL2.OI1, TIMx_CTRL2.OI1N and TIMx_CCEN.CC1EN. 1: Enable - Enable output OC1N signal. The level of OC1N depends on the value of these bits TIMx_BKDT.MOEN, TIMx_BKDT.OSSI, TIMx_BKDT.OSSR, TIMx_CTRL2.OI1, TIMx_CTRL2.OI1N and TIMx_CCEN.CC1EN.
1	CC1P	Capture/Compare 1 output polarity When the corresponding channel of CC1 is in output mode: 0: OC1 active high 1: OC1 active low When the corresponding channel of CC1 is in input mode: At this time, this bit is used to select whether IC1 or the inverse signal of IC1 is used as the trigger or capture signal.

Bit Field	Name	Description
		0: non-inverted: Capture action occurs when IC1 generates a rising edge. When used as external trigger, IC1 is non-inverted. 1: inverted: Capture action occurs when IC1 generates a falling edge. When used as external trigger, IC1 is inverted. <i>Note: If TIMx_BKDT.LCKCFG = 3 or 2, these bits cannot be modified.</i>
0	CC1EN	Capture/Compare 1 output enable When the corresponding channel of CC1 is in output mode: 0: Disable - Disable output OC1 signal. The level of OC1 depends on the value of these bits TIMx_BKDT.MOEN, TIMx_BKDT.OSSI, TIMx_BKDT.OSSR, TIMx_CTRL2.OI1, TIMx_CTRL2.OI1N and TIMx_CCEN.CC1NEN. 1: Enable - Enable output OC1 signal. The level of OC1N depends on the value of these bits TIMx_BKDT.MOEN, TIMx_BKDT.OSSI, TIMx_BKDT.OSSR, TIMx_CTRL2.OI1, TIMx_CTRL2.OI1N and TIMx_CCEN.CC1NEN. When the corresponding channel of CC1 is in input mode: At this time, this bit is used to disable/enable the capture function. 0: Disable capture 1: Enable capture

Table 9-4 Output Control Bits of Complementary Ocx and Ocxn Channels with Break Function

Control Bits					Output State ¹⁾	
MOEN	OSSI	OSSR	CCxEN	CCxNEN	OCx Output State	OCxN Output State
1	X	0	0	0	Output disabled (not driven by timer) OCx=0,OCx_EN=0	Output disabled (not driven by timer) OCxN=0,OCxN_EN=0
		0	0	1	Output disabled (not driven by 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 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 timer) OCx=CCxP,OCx_EN=0	Output disabled (not driven by 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

Control Bits					Output State ¹⁾	
MOEN	OSSI	OSSR	CCxEN	CCxNEN	OCx Output State	OCxN Output State
0	0	X	0	0	Output disabled (not driven by timer)	
	0		0	1	Asynchronously: $OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0;$	
	0		1	0	Then if the clock is present: $OCx=OIx$ and $OCxN=OIxN$ after a dead-time, when $(CCxP \wedge OIx) \wedge (CCxNP \wedge OIxN) \neq 0.$	
	0		1	1		
	1		0	0	Off-state (Output enabled with inactive state)	
	1		0	1	Asynchronously: $OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1;$	
	1		1	0	Then if the clock is present: $OCx=OIx$ and $OCxN=OIxN$ after a dead-time, when $(CCxP \wedge OIx) \wedge (CCxNP \wedge OIxN) \neq 0.$	
	1		1	1		

Note:

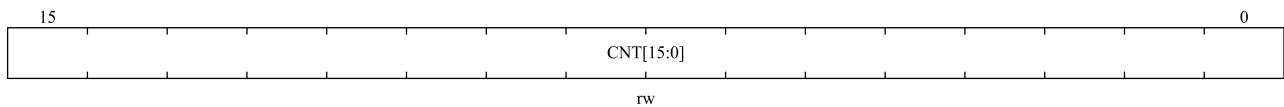
1. If both outputs of a channel are not used ($CCxEN = CCxNEN = 0$), $OIx, OIxN, CCxP$ and $CCxNP$ must all be cleared.

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

9.4.11 Counters (TIMx_CNT)

Offset address: 0x24

Reset value: 0x0000

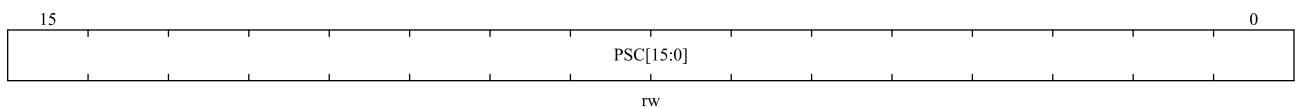


Bit Field	Name	Description
15:0	CNT[15:0]	Counter value

9.4.12 Prescaler (TIMx_PSC)

Offset address: 0x28

Reset value: 0x0000

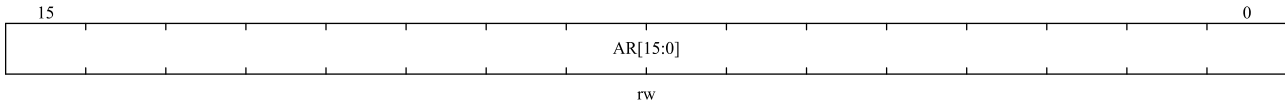


Bit Field	Name	Description
15:0	PSC[15:0]	Prescaler value Counter clock $f_{CK_CNT} = f_{CK_PSC} / (PSC [15:0] + 1).$ Each time an update event occurs, the PSC value is loaded into the active prescaler register.

9.4.13 Auto-reload Register (TIMx_AR)

Offset address: 0x2C

Reset values: 0xFFFF

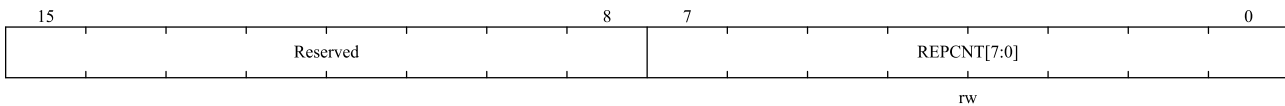


Bit Field	Name	Description
15:0	AR[15:0]	Auto-reload value These bits define the value that will be loaded into the actual auto-reload register. See Section 9.3.1 for more details. When the TIMx_AR.AR [15:0] value is null, the counter does not work.

9.4.14 Repeat Count Registers (TIMx_REPCNT)

Offset address: 0x30

Reset value: 0x0000

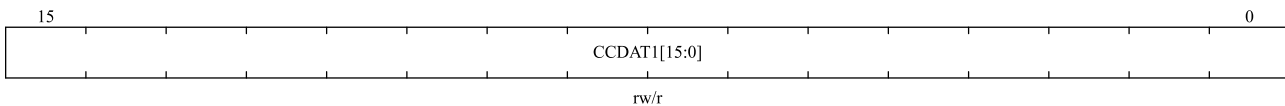


Bit Field	Name	Description
15:8	Reserved	Reserved, the reset value must be maintained
7:0	REPCNT[7:0]	Repetition counter value Repetition counter is used to generate the update event or update the timer registers only after a given number (N+1) cycles of the counter, where N is the value of TIMx_REPCNT.REPCNT . The repetition counter is decremented at each counter overflow in up-counting mode, at each counter underflow in down-counting mode or at each counter overflow and at each counter underflow in center-aligned mode. Setting the TIMx_EVTGEN.UDGN bit will reload the content of TIMx_REPCNT.REPCNT and generate an update event.

9.4.15 Capture/Compare Register 1 (TIMx_CC DAT1)

Offset address: 0x34

Reset value: 0x0000



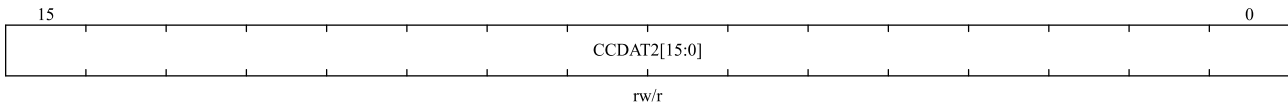
Bit Field	Name	Description
15:0	CCDAT1[15:0]	Capture/Compare 1 value 1) CC1 channel is configured as output: CCDAT1 contains the value to be compared to the counter TIMx_CNT, signaling on the OC1 output. If the preload feature is not selected in TIMx_CCMOD1.OC1PEN bit, the written value is immediately transferred to the active register. Otherwise, this preloaded value is transferred to the active register only when an update event occurs. 2) CC1 channel is configured as input:

Bit Field	Name	Description
		CCDAT1 contains the counter value transferred by the last input capture 1 event (IC1). When configured as input mode, register CCDAT1 and CCDDAT1 are only readable. When configured as output mode, register CCDAT1 and CCDDAT1 are readable and writable.

9.4.16 Capture/Compare Register 2 (TIMx_CCDA2)

Offset address: 0x38

Reset value: 0x0000

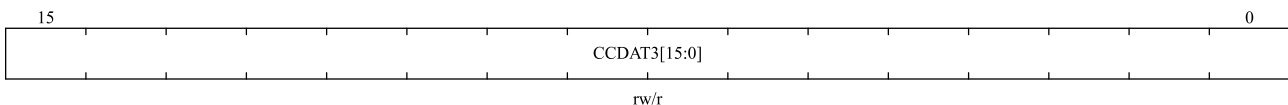


Bit Field	Name	Description
15:0	CCDAT2[15:0]	<p>Capture/Compare 2 values</p> <p>1) CC2 channel is configured as output: CCDAT2 contains the value to be compared to the counter TIMx_CNT, signaling on the OC2 output. If the preload feature is not selected in TIMx_CCMOD1.OC2PEN bit, the written value is immediately transferred to the active register. Otherwise, this preloaded value is transferred to the active register only when an update event occurs.</p> <p>2) CC2 channel is configured as input: CCDAT2 contains the counter value transferred by the last input capture 2 event (IC2). When configured as input mode, register CCDAT2 and CCDDAT2 are only readable. When configured as output mode, register CCDAT2 and CCDDAT2 are readable and writable.</p>

9.4.17 Capture/Compare Register 3 (TIMx_CCDA3)

Offset address: 0x3C

Reset value: 0x0000



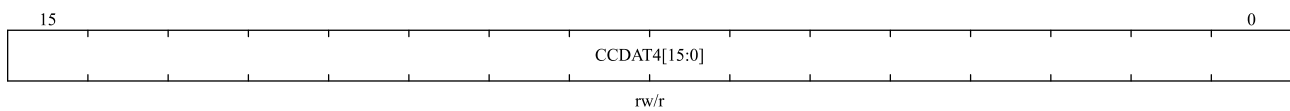
Bit Field	Name	Description
15:0	CCDAT3[15:0]	<p>Capture/Compare 3 value</p> <p>1) CC3 channel is configured as output: CCDAT3 contains the value to be compared to the counter TIMx_CNT, signaling on the OC3 output. If the preload feature is not selected in TIMx_CCMOD2.OC3PEN bit, the written value is immediately transferred to the active register. Otherwise, this preloaded value is transferred to the active register only when an update event occurs.</p> <p>2) CC3 channel is configured as input:</p>

Bit Field	Name	Description
		CCDAT3 contains the counter value transferred by the last input capture 3 event (IC3). When configured as input mode, register CCDAT3 and CCDDAT3 are only readable. When configured as output mode, register CCDAT3 and CCDDAT3 are readable and writable.

9.4.18 Capture/Compare Register 4 (TIMx_CCDA4)

Offset address: 0x40

Reset value: 0x0000

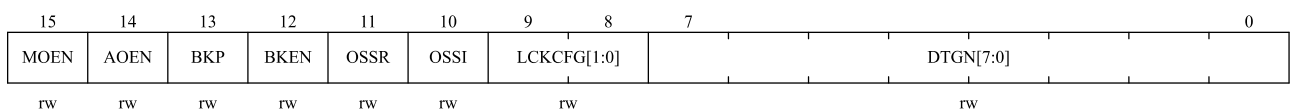


Bit Field	Name	Description
15:0	CCDAT4[15:0]	Capture/Compare 4 value 1) CC4 channel is configured as output: CCDAT4 contains the value to be compared to the counter TIMx_CNT, signaling on the OC4 output. If the preload feature is not selected in TIMx_CCMOD2.OC4PEN bit, the written value is immediately transferred to the active register. Otherwise, this preloaded value is transferred to the active register only when an update event occurs. 2) CC4 channel is configured as input: CCDAT4 contains the counter value transferred by the last input capture 4 event (IC4). When configured as input mode, register CCDAT4 and CCDDAT4 are only readable. When configured as output mode, register CCDAT4 and CCDDAT4 are readable and writable.

9.4.19 Break and Dead-time Registers (TIMx_BKDT)

Offset address: 0x44

Reset value: 0x0000



Note: AOEN, BKP, BKEN, OSSI, OSSR, and DTGN [7:0] bits can all be write protected depending on the LOCK configuration, and it is necessary to configure all of them on the first write to the TIMx_BKDT register.

Bit Field	Name	Description
15	MOEN	Main Output enable This bit can be set by software or hardware depending on the TIMx_BKDT.AOEN bit, and is asynchronously cleared to '0' by hardware once the brake input is active. It is only valid for

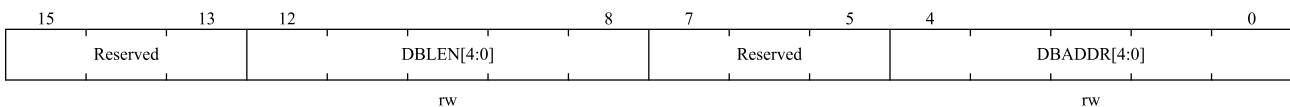
Bit Field	Name	Description
		channels configured as outputs. 0: OC and OCN outputs are disabled or forced to idle state. 1: OC and OCN outputs are enabled if TIMx_CCEN.CCxEN or TIMx_CCEN.CCxNEN bits are set. For more details, refer to Section 9.4.10 Capture/Compare enable registers (TIMx_CCEN).
14	AOEN	Automatic output enable 0: Only software can set TIMx_BKDT.MOEN; 1: Software sets TIMx_BKDT.MOEN; or if the break input is not active, when the next update event occurs, hardware automatically sets TIMx_BKDT.MOEN.
13	BKP	Break input polarity 0: Low level of the brake input is valid 1: High level of the brake input is valid <i>Note: Any write to this bit requires an APB clock delay to take effect.</i>
12	BKEN	Break enable 0: Disable brake input (BRK and CCS clock failure events) 1: Enable brake input (BRK and CCS clock failure events) <i>Note: Any write to this bit requires an APB clock delay to take effect.</i>
11	OSSR	Off-state Selection for Run Mode This bit is used when TIMx_BKDT.MOEN=1 and the channel is a complementary output. The OSSR bit does not exist in timer without complementary outputs. 0: When inactive, OCx/OCxN outputs are disabled (OCx/OCxN enable output signal = 0) 1: When inactive, OCx/OCxN outputs are enabled with their inactive level as soon as CCxEN = 1 or CCxNEN = 1. Then, OCx/OCxN enable output signal = 1 For more details, See Section 9.4.10, capture/compare enablement registers (TIMx_CCEN).
10	OSSI	Off-state Selection for Idle Mode This bit is used when TIMx_BKDT.MOEN=0 and the channels configured as outputs. 0: When inactive, OCx/OCxN outputs are disabled (OCx/OCxN enable output signal = 0) 1: When inactive, OCx/OCxN outputs are enabled with their idle level as soon as CCxEN = 1 or CCxNEN = 1. Then, OCx/OCxN enable output signal = 1 For more details, See Section 9.4.10, capture/compare enablement registers (TIMx_CCEN).
9:8	LCKCFG[1:0]	Lock Configuration These bits offer a write protection against software errors. 00: – No write protected. 01: – LOCK Level 1 TIMx_BKDT.DTGN, TIMx_BKDT.BKEN, TIMx_BKDT.BKP, TIMx_BKDT.AOEN, TIMx_CTRL2.OIx, TIMx_CTRL2.OIxN bits enable write protection. 10: – LOCK Level 2 Except for register write protection in LOCK Level 1 mode, TIMx_CCEN.CCxP and TIMx_CCEN.CCxNP (If the corresponding channel is configured in output mode), TIMx_BKDT.OSSR and TIMx_BKDT.OSSI bits also enable write protection. 11:

Bit Field	Name	Description
		<p>– LOCK Level 3</p> <p>Except for register write protection in LOCK Level 2, TIMx_CCMODx.OCxMD and TIMx_CCMODx.OCxPEN bits (If the corresponding channel is configured in output mode) also enable write protection.</p> <p><i>Note: After the system reset, the LCKCFG bit can only be written once. Once written to the TIMx_BKDT register, LCKCFG will be protected until the next reset.</i></p>
7:0	DTGN [7:0]	<p>Dead-time Generator</p> <p>These bits define the dead-time duration between inserted complementary outputs. The relationship between the DTGN value and the dead time is as follows::</p> <p>DTGN[7:5] = 0xx: $dead\ time = DTGN[7:0] \times (t_{DTS})$</p> <p>DTGN[7:5] = 10x: $dead\ time = (64 + DTGN[5:0]) \times (2 \times t_{DTS})$</p> <p>DTGN[7:5]=110: $dead\ time = (32 + DTGN[4:0]) \times (8 \times t_{DTS})$</p> <p>DTGN [then] = 111: $dead\ time = (32 + DTGN [4:0]) \times (16 \times t_{DTS})$</p> <p>t_{DTS} value see TIMx_CTRL1.CLKD [1:0].</p>

9.4.20 DMA Control Register (TIMx_DCTRL)

Offset address: 0x48

Reset value: 0x0000



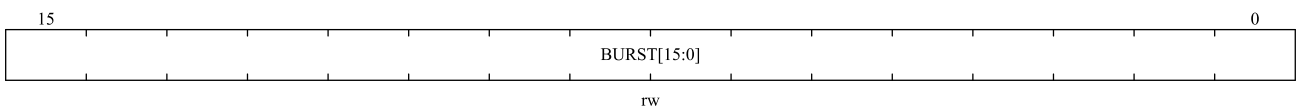
Bit Field	Name	Description
15:9	Reserved	Reserved, the reset value must be maintained, kept at 0.
12:8	DBLEN[4:0]	<p>DMA Burst Length</p> <p>This bit field defines the number DMA will accesses (write/read) TIMx_DADDR register.</p> <p>00000: 1 time transfer 00001: 2 times transfers 00010: 3 times transfers ... 10001: 18 times transfers</p>
7:5	Reserved	Reserved, the reset value must be maintained.
4:0	DBADDR[4:0]	<p>DMA Base Address</p> <p>This bit field defines the first address where the DMA accesses the TIMx_DADDR register.</p> <p>When access is done through the TIMx_DADDR first time, this bit-field specifies the address you just access. And then the second access to the TIMx_DADDR, you will access the address of “DMA Base Address + 4”</p>

Bit Field	Name	Description
		00000: TIMx_CTRL1, 00001: TIMx_CTRL2, 00010: TIMx_SMCTRL, ... 10001: TIMx_BKDT 10010: TIMx_DCTRL

9.4.21 DMA Transfer Buffer Register (TIMx_DADDR)

Offset address: 0x4C

Reset value: 0x0000

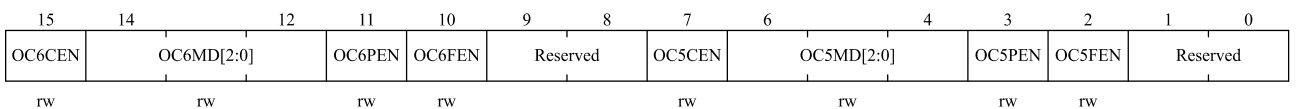


Bit Field	Name	Description
15:0	BURST[15:0]	DMA access buffer. When a read or write operation is assigned to this register, the register located at the address range (DMA base address + DMA burst length × 4) will be accessed. DMA base address = The address of TIMx_CTRL1 + TIMx_DCTRL.DBADDR * 4; DMA burst len = TIMx_DCTRL.DBLEN + 1. Example: If TIMx_DCTRL.DBLEN = 0x3(4 transfers), TIMx_DCTRL.DBADDR = 0xD (TIMx_CC DAT1), DMA data length = half word, DMA memory address = buffer address in SRAM, DMA peripheral address = TIMx_DADDR address. When an event occurs, TIMx will send requests to the DMA, and transfer data 4 times. For the first time, DMA access to the TIMx_DADDR register will be mapped to access TIMx_CC DAT1 register; For the second time, DMA access to the TIMx_DADDR register will be mapped to access TIMx_CC DAT2 register; For the fourth time, DMA access to the TIMx_DADDR register will be mapped to access TIMx_CC DAT4 register;

9.4.22 Capture/Compare Mode Registers 3(TIMx_CCMOD3)

Offset address: 0x54

Reset value: 0x0000



Bit Field	Name	Description
15	OC6CEN	Output compare 6 clear enable

Bit Field	Name	Description
14:12	OC6MD[2:0]	Output compare 6 mode
11	OC6PEN	Output compare 6 preload enable
10	OC6FEN	Output compare 6 fast enable
9:8	Reserved	Reserved, the reset value must be maintained
7	OC5CEN	Output compare 5 clear enable
6:4	OC5MD[2:0]	Output compare 5 mode
3	OC5PEN	Output compare 5 Preload enable
2	OC5FEN	Output compare 5 fast enable
1: 0	Reserved	Reserved, the reset value must be maintained

9.4.23 Capture/Compare Register 5 (TIMx_CC DAT5)

Offset address: 0x58

Reset value: 0x0000

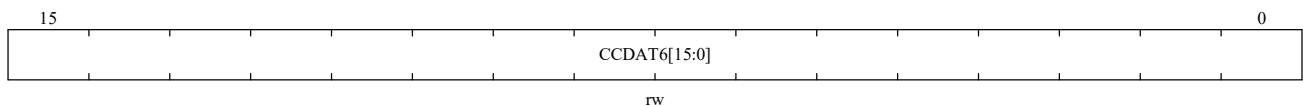


Bit Field	Name	Description
15:0	CCDAT5[15:0]	<p>Capture/Compare 5 value</p> <ul style="list-style-type: none"> CC5 channel can only configured as output: <p>CCDAT5 contains the value to be compared to the counter TIMx_CNT, signaling on the OC5 output.</p> <p>If the preload feature is not selected in TIMx_CCMOD3.OC5PEN bit, the written value is immediately transferred to the active register. Otherwise, this preloaded value is transferred to the active register only when an update event occurs.</p> <p>TIM1_CC5 and TIM8_CC5 is used for comparator blanking.</p>

9.4.24 Capture/Compare Register 6 (TIMx_CC DAT6)

Offset address: 0x5C

Reset value: 0x0000



Bit Field	Name	Description
15:0	CCDAT6[15:0]	<p>Capture/Compare 6 value</p> <ul style="list-style-type: none"> • CC6 channel can only configured as output: <p>CCDAT6 contains the value to be compared to the counter TIMx_CNT, signaling on the OC6 output.</p> <p>If the preload feature is not selected in TIMx_CCMOD3.OC6PEN bit, the written value is immediately transferred to the active register. Otherwise, this preloaded value is transferred to the active register only when an update event occurs.</p> <p>TIM1_CC6 for OPAMP switch.</p>

10 General-purpose Timers (TIM3 and TIM4)

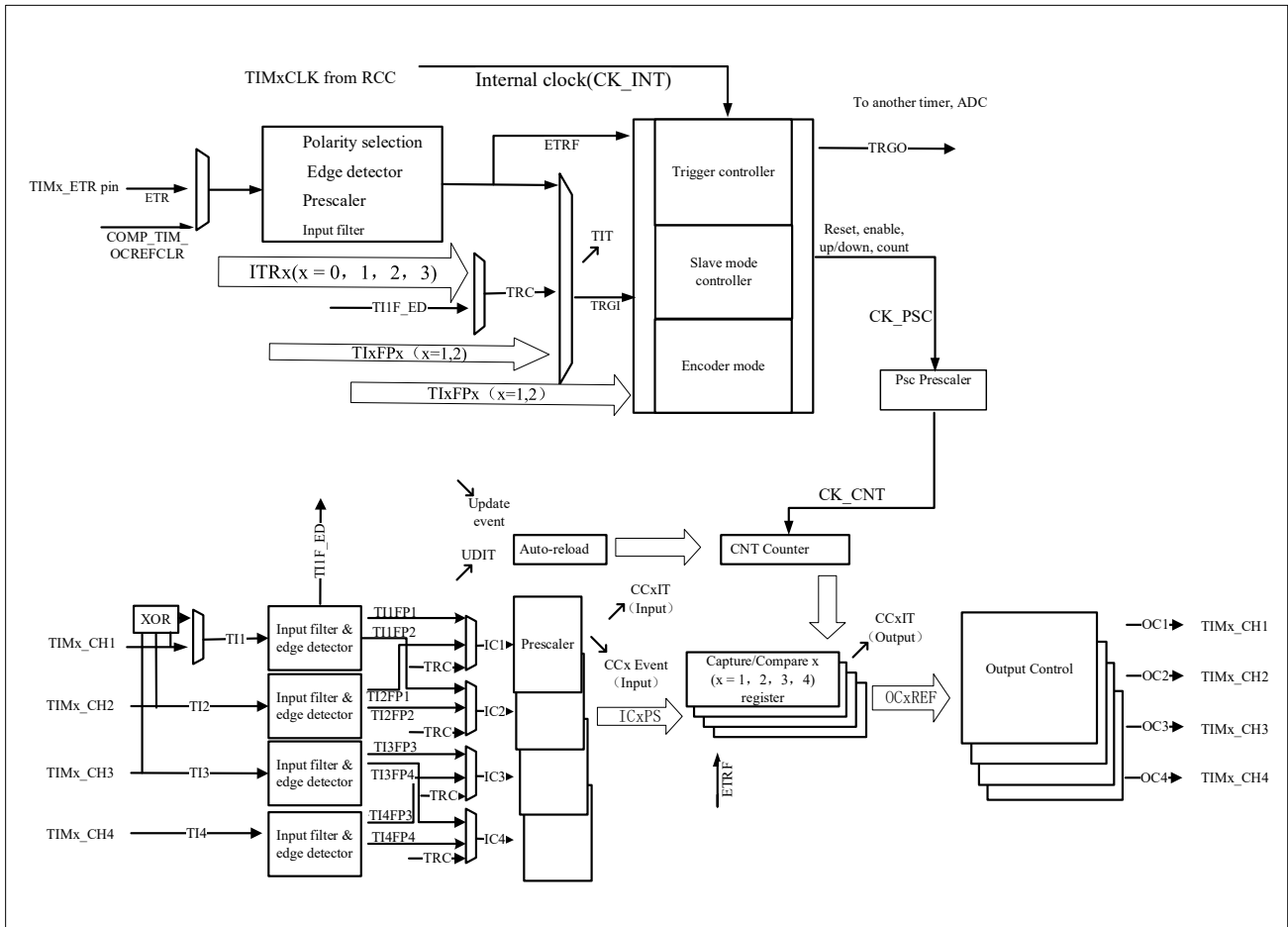
10.1 General-purpose Timers Introduction

The general-purpose timers (TIM3 and TIM4) are mainly used in the following scenarios: counting the input signals, measuring the pulse length of the input signals and generating the output waveform, etc.

10.2 Main Features of General-purpose Timers

- 16-bit auto-reload counters. (It can realize up-counting, down-counting, up/down counting)
- 16-bit programmable prescaler. (The prescaler factor can be configured with any value between 1 and 65536)
- Up to 4 channels
 - Input capture
 - Output compare
 - PWM generation (edge or center-aligned mode)
 - Single-pulse mode output
- Channel's operation modes: PWM output, output compare, one-pulse mode output, input capture
- The events that generate the interrupt/DMA are as follows:
 - Update event
 - Trigger event
 - Input capture
 - Output compare
- Timer can be controlled by external signal
- Timers can be linked together internally for timer synchronization or chaining
- Incremental (quadrature) encoder interface: used for tracking motion and resolving rotation direction and position
- Hall sensor interface: used to do three-phase motor control
- Supports capturing the signal of the internal comparator output

Figure 10-1 Block Diagram of TIMx (x=3 and 4)



↘ The event ↗ Interrupt and DMA output

The capture channel 1/2/3 input can come from IOM or comparator output

10.3 General-purpose Timers Description

10.3.1 Time-base Unit

The time-base unit mainly includes: prescaler, counter and auto-reload. When the time base unit is operating, the software can read and write the corresponding registers (TIMx_PSC, TIMx_CNT and TIMx_AR) at any time.

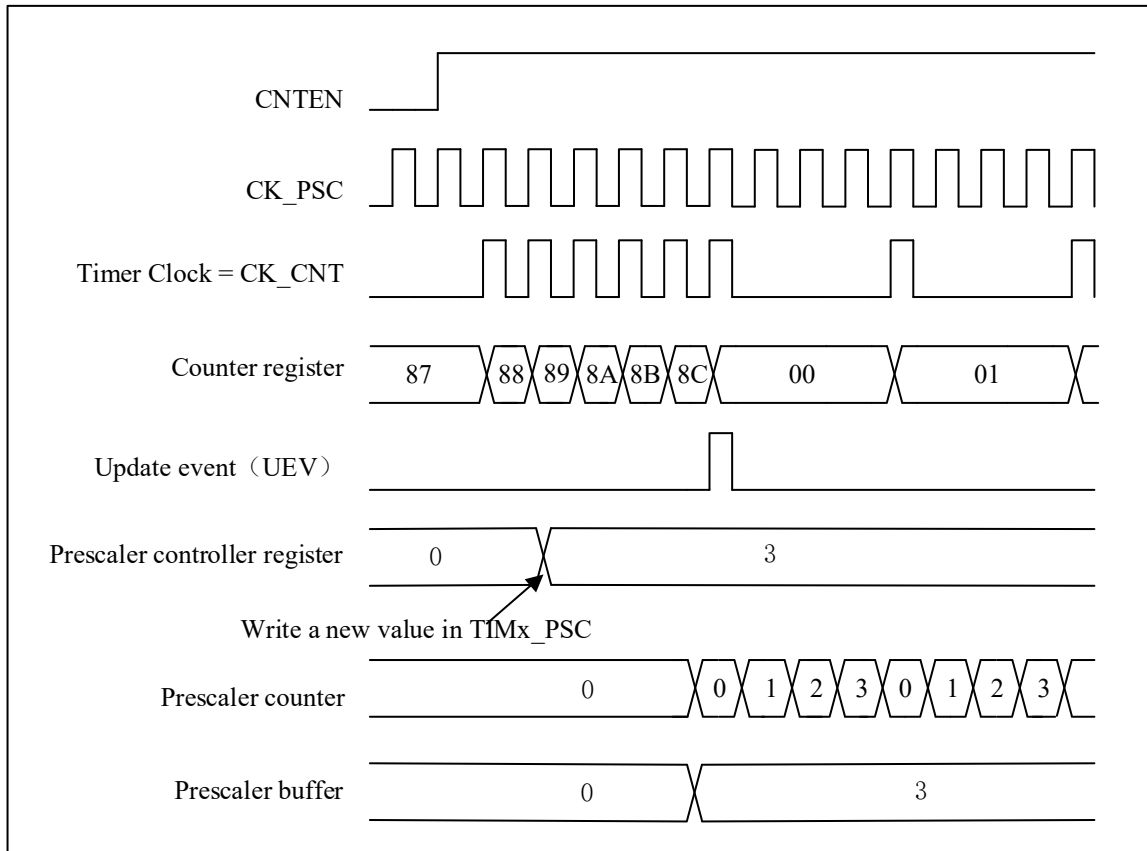
Depending on the setting of the auto-reload preload enable bit (TIMx_CTRL1.ARPEN), the value of the preload register is transferred to the shadow register immediately or at each update event UEV. An update event is generated when the counter reaches the overflow/underflow condition and it can be generated by software when TIMx_CTRL1.UPDIS=0. The counter CK_CNT is valid only when the TIMx_CTRL1.CNTEN bit is set. The counter starts counting one clock cycle after the TIMx_CTRL1.CNTEN bit is set.

Prescaler description

The TIMx_PSC register consists of a 16-bit counter that can be used to divide the counter clock frequency by any factor between 1 and 65536. It can be changed on the fly as it is buffered. The new prescaler ratio is taken into account at the

next update event.

Figure 10-2 Counter Timing Diagram with Prescaler Division Change from 1 To 4



10.3.2 Counter Mode

Up-counting mode

In up-counting mode, the counter will count from 0 to the value of the register TIMx_AR, then restart from 0, and a counter overflow event is generated.

When the TIMx_CTRL1.UPRS bit (select update request) and the TIMx_EVTGEN.UDGN bit are set, the update event (UEV) is generated, and TIMx_STS.UDITF will not be set by hardware, therefore, no update interrupts or update DMA requests are generated. This is to avoid generating the update interrupt when clearing the counter.

When an update event occurs, all registers are updated and the TIMx_STS.UDITF is set (depending on TIMx_CTRL1.UPRS):

- The auto-reload shadow registers is updated with preload value (TIMx_AR), when TIMx_CTRL1.ARPEN = 1.
- The prescaler shadow register is reloaded with the preload value (TIMx_PSC).

Setting TIMx_CTRL1.UPDIS=1 is to avoid updating the shadow registers when new values are written to the preload registers.

When an update event occurs, the counter will still be cleared and the prescaler counter will also be set to 0 (but the prescaler value will remain unchanged).

The figure below shows some examples of the counter behavior and the update flags for different division factors in the

up-counting mode.

Figure 10-3 Timing Diagram of Up-Counting. The Internal Clock Divider Factor = 2/N

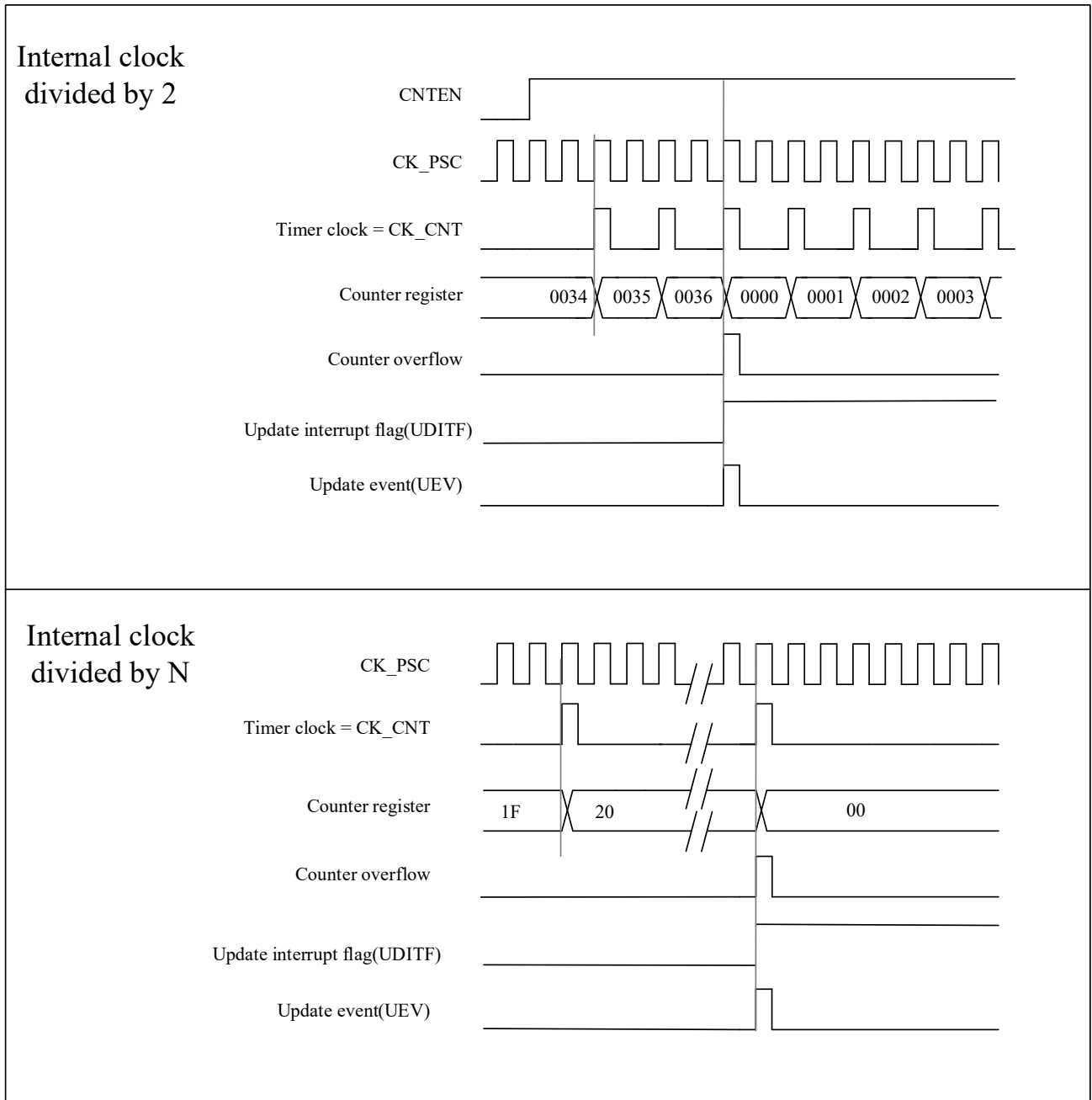
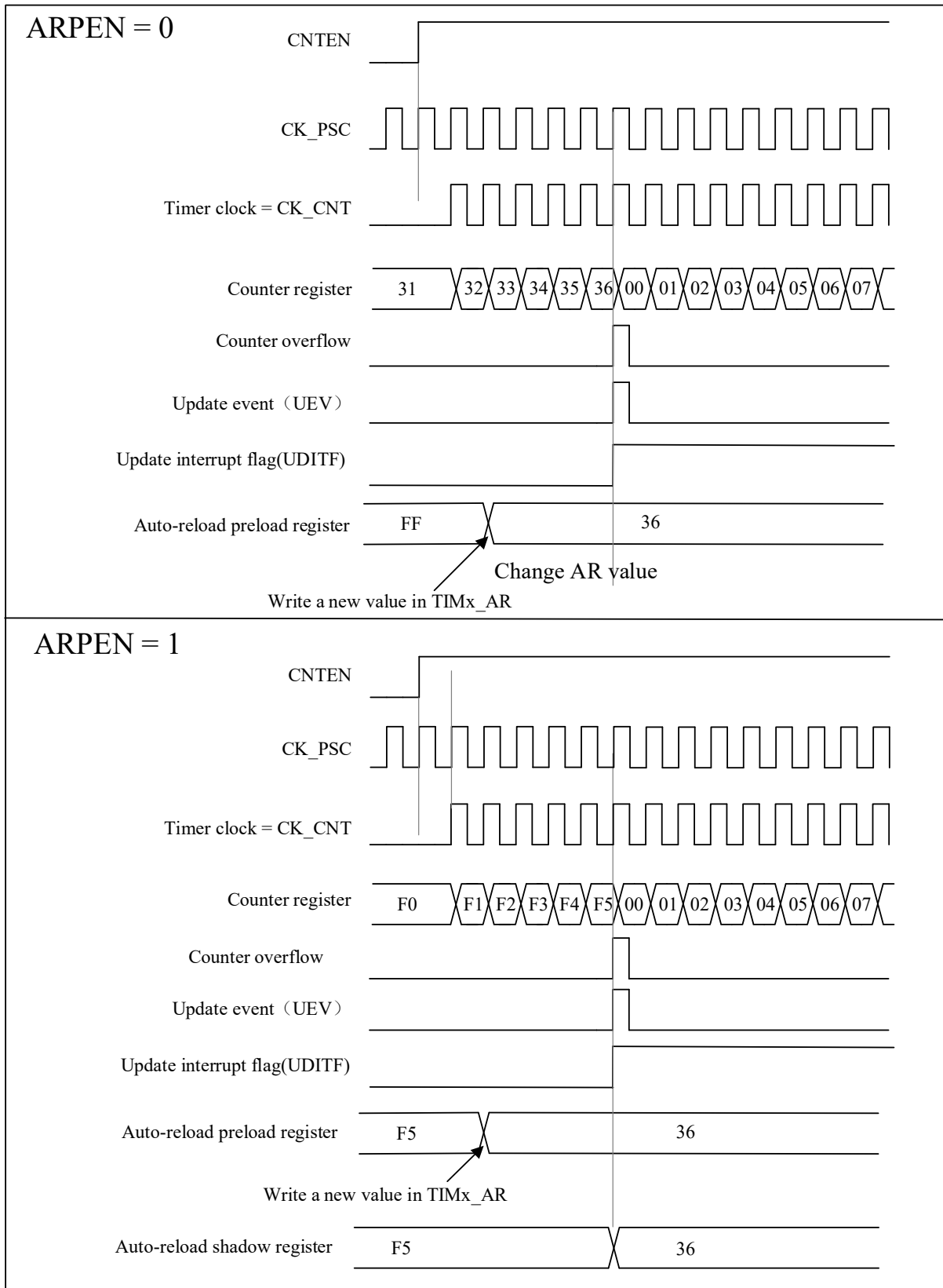


Figure 10-4 Timing Diagram of The Up-Counting, Update Event When ARPEN=0/1



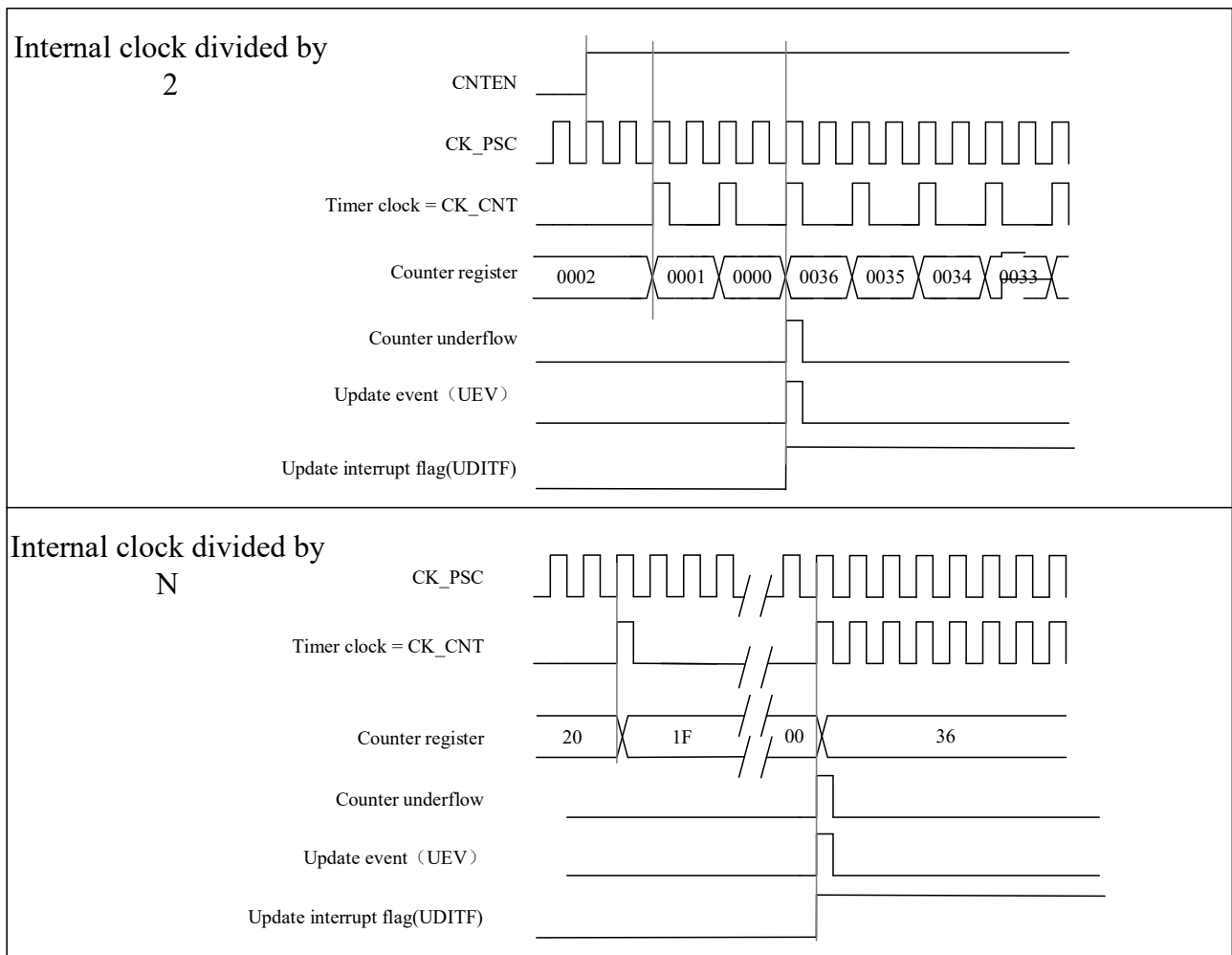
Down-counting mode

In down-counting mode, the counter will decrement from the value of the register TIMx_AR to 0, then restart from the auto-reload value and generate a counter underflow event.

The process of configuring update events and updating registers in down-counting mode is the same as in up-counting mode, refer to Section 0.

The figure below shows some examples of the counter behavior and the update flags for different division factors in the down-counting mode.

Figure 10-5 Timing Diagram of The Down-Counting, Internal Clock Divided Factor = 2/N



Center-aligned mode

In center-aligned mode, the counter increments from 0 to the value (TIMx_AR) – 1, a counter overflow event is generated. Then it counts down from the auto-reload value (TIMx_AR) to 1 and generates a counter underflow event. Then the counter restart from 0 and starts counting up again.

In this mode, the TIMx_CTRL1.DIR direction bits have no effect and the count direction is updated and specified by hardware. Center-aligned mode is active when the TIMx_CTRL1. CAMSEL bit is not equal to "00".

The update events can be generated at each time the counter overflows and at each time the counter underflows. Alternatively, an update event can also be generated by setting the TIMx_EVTGEN.UDGN bit (either by software or using

a slave mode controller). In this case, the counter restarts from 0, as does the prescaler's counter.

Please note: if the update source is a counter overflow, auto-reload update before reloading the counter.

Figure 10-6 Timing Diagram of The Center-Aligned, Internal Clock Divided Factor =2/N

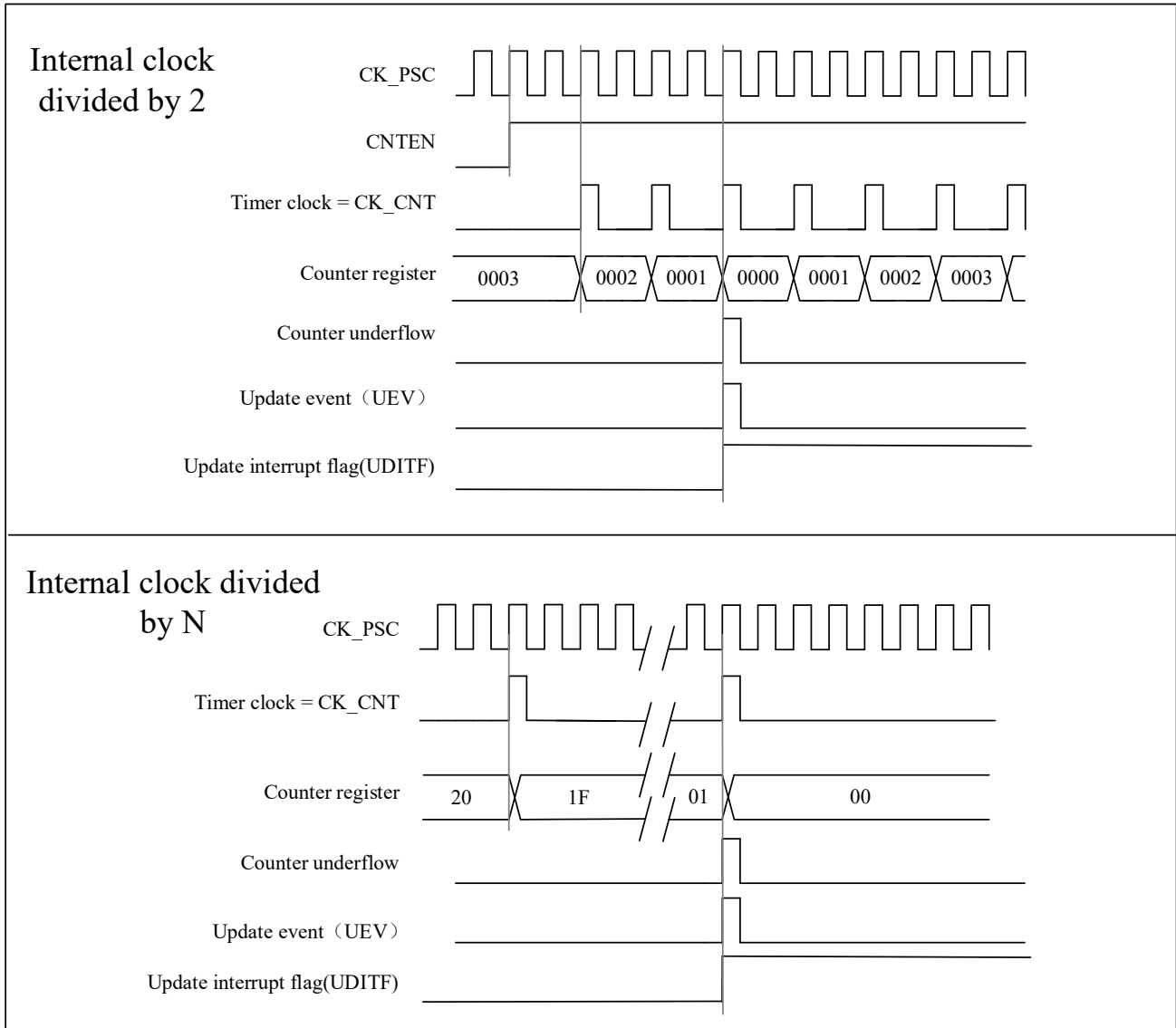
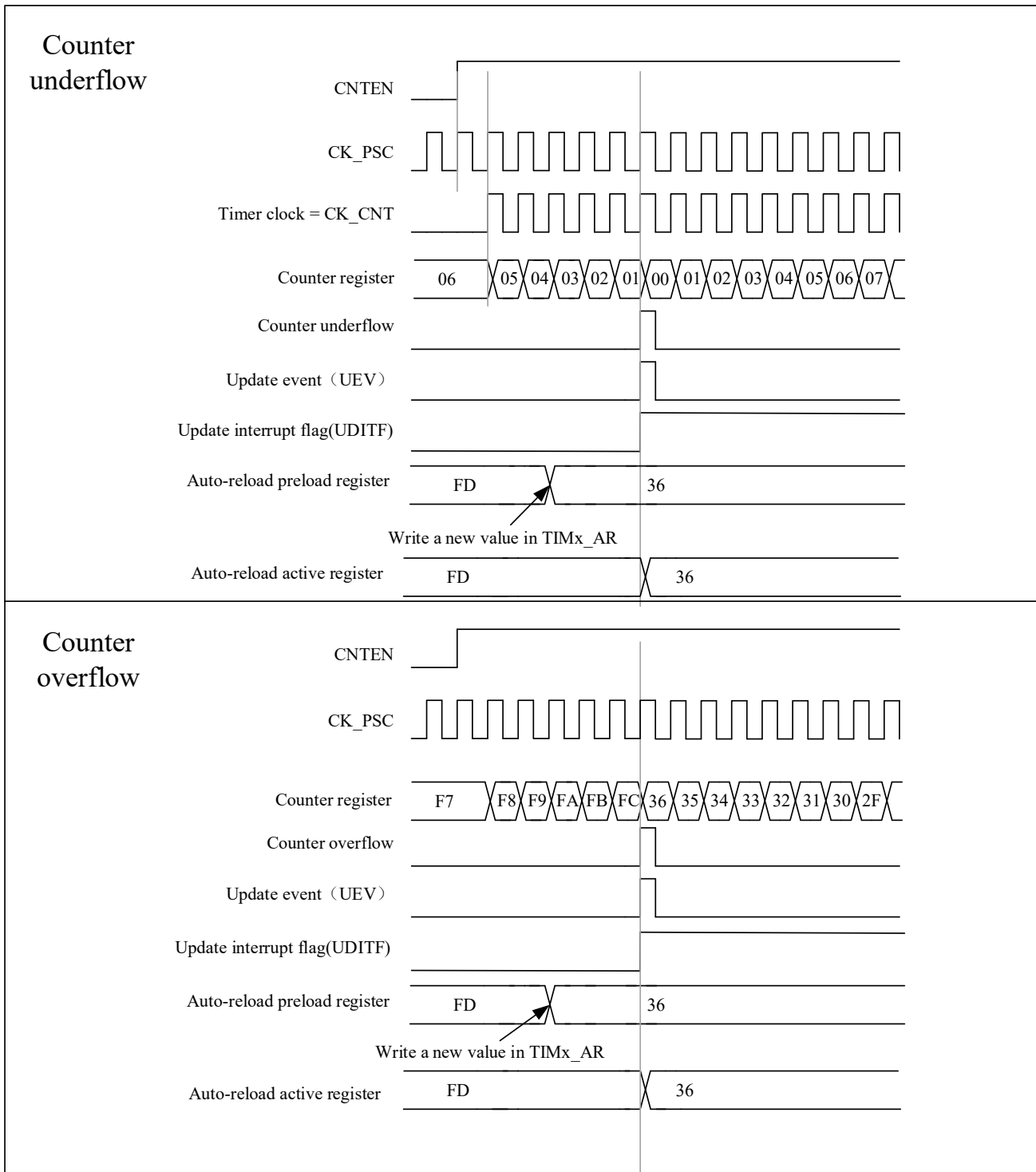


Figure 10-7 A Center-Aligned Sequence Diagram That Includes Counter Overflows and Underflows (ARPEN

= 1)



10.3.3 Clock Selection

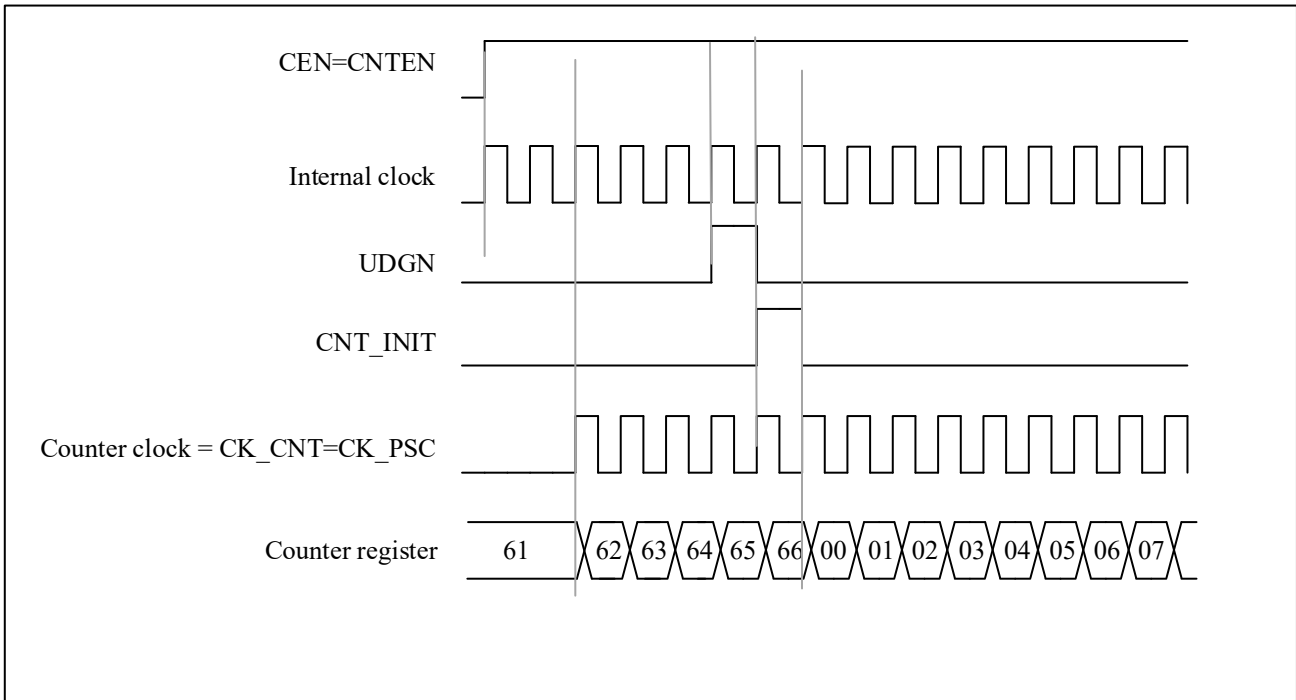
- The internal clock of timers: CK_INT
- Two kinds of external clock mode :
 - External input pin
 - External trigger input ETR

- Internal trigger input (ITRx): one timer is used as a prescaler for another timer.

Internal clock source (CK_INT)

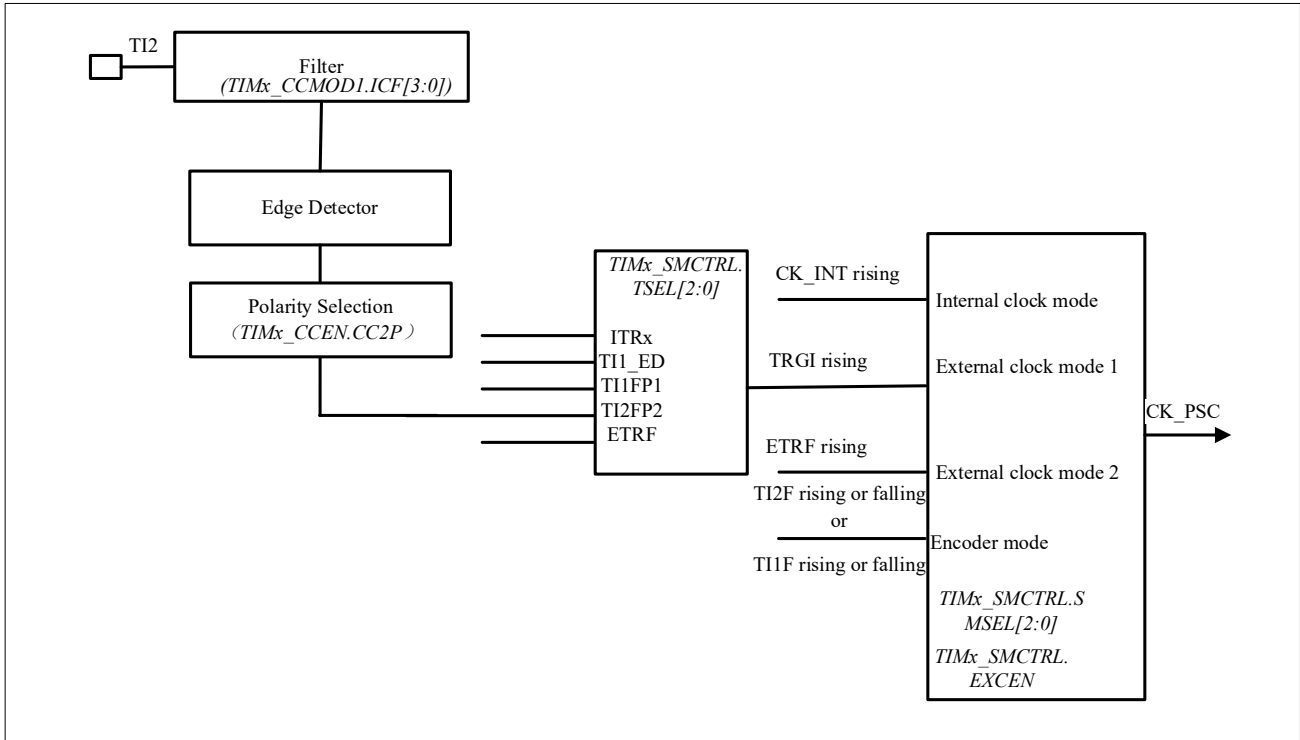
When the TIMx_SMCTRL.SMSEL is equal to “000”, the slave mode controller is disabled. The three control bits (TIMx_CTRL1.CNTEN, TIMx_CTRL1.DIR, TIMx_EVTGEN.UDGN) can only be changed by software (except TIMx_EVTGEN.UDGN, which remains cleared automatically). It is provided that the TIMx_CTRL1.CNTEN bit is written as '1' by software, the clock source of the prescaler is provided by the internal clock CK_INT.

Figure 10-8 Control Circuit in Normal Mode, Internal Clock Divided by 1



External clock source mode 1

Figure 10-9 TI2 External Clock Connection Example



This mode is selected by configuring `TIMx_SMCTRL.SMSEL=111`. The counter can be configured to count on the rising or falling edge of the clock at the selected input.

For example, to configure up-counting mode to count on the rising edge of the clock at the TI2 input, the configuration steps are as follows:

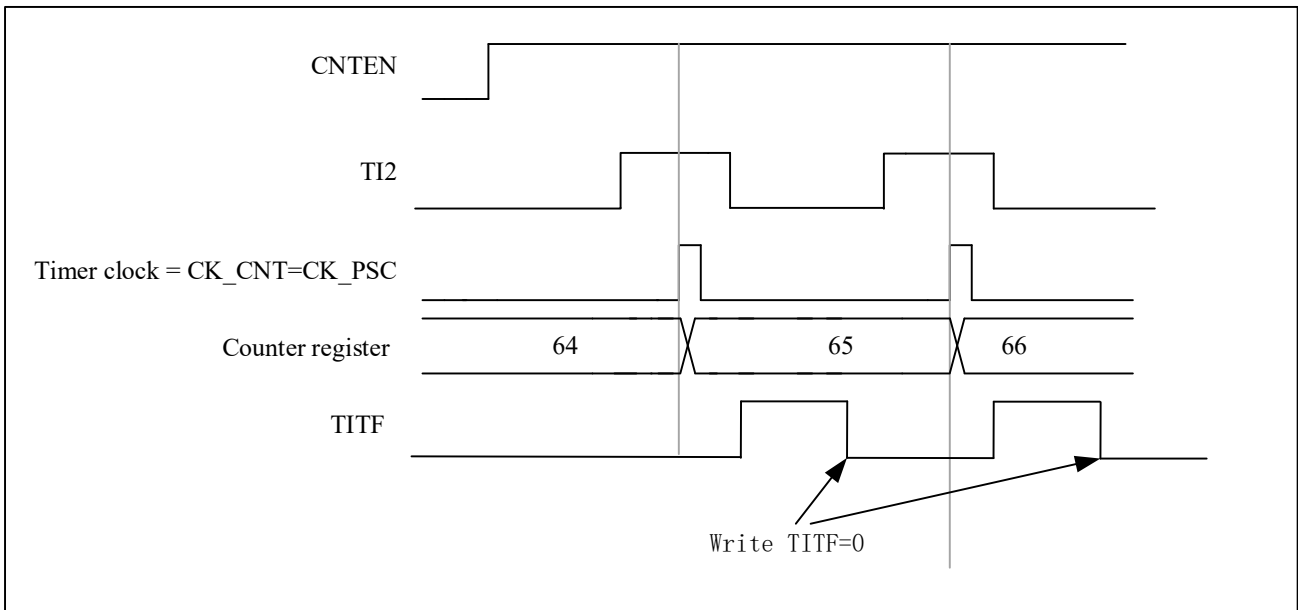
1. Configure `TIMx_CCMOD1.CC2SEL` equal to '01', CC2 channel is configured as input, IC2 is mapped to TI2
2. Configure `TIMx_CCEN.CC2P` equal to '0', select clock rising edge polarity
3. Configure `TIMx_CCMOD1.IC2F[3:0]` to select input filter bandwidth (if filter is not needed, keep IC2F bit at '0000')
4. Configure `TIMx_SMCTRL.SMSEL` equal to '111', select timer external clock mode 1
5. Configure `TIMx_SMCTRL.TSEL` equal to '110', select TI2 as the trigger input source
6. Configure `TIMx_CTRL1.CNTEN` equal to '1' to start the counter

Note: The capture prescaler is not used for triggering, so it does not need to be configured

When the rising edge of the timer clock occurs at `TI2=1`, the counter counts once and the `TIMx_STS .TITF` flag is pulled high.

The delay between the rising edge of TI2 and the actual clock of the counter depends on the resynchronization circuit at the input of TI2.

Figure 10-10 Control Circuit in External Clock Mode 1

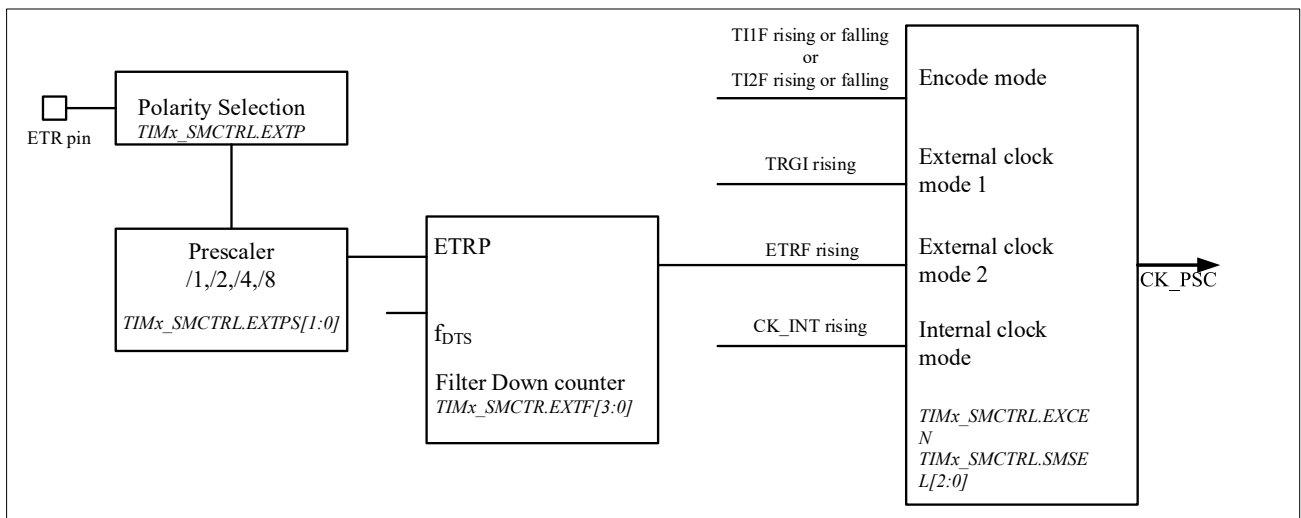


External clock source mode 2

This mode is selected by TIMx_SMCTRL .EXCEN equal to 1. The counter can count on every rising or falling edge of the external trigger input ETR.

The following figure is a schematic diagram of the external trigger input module in external clock source mode 2

Figure 10-11 External Trigger Input Block Diagram

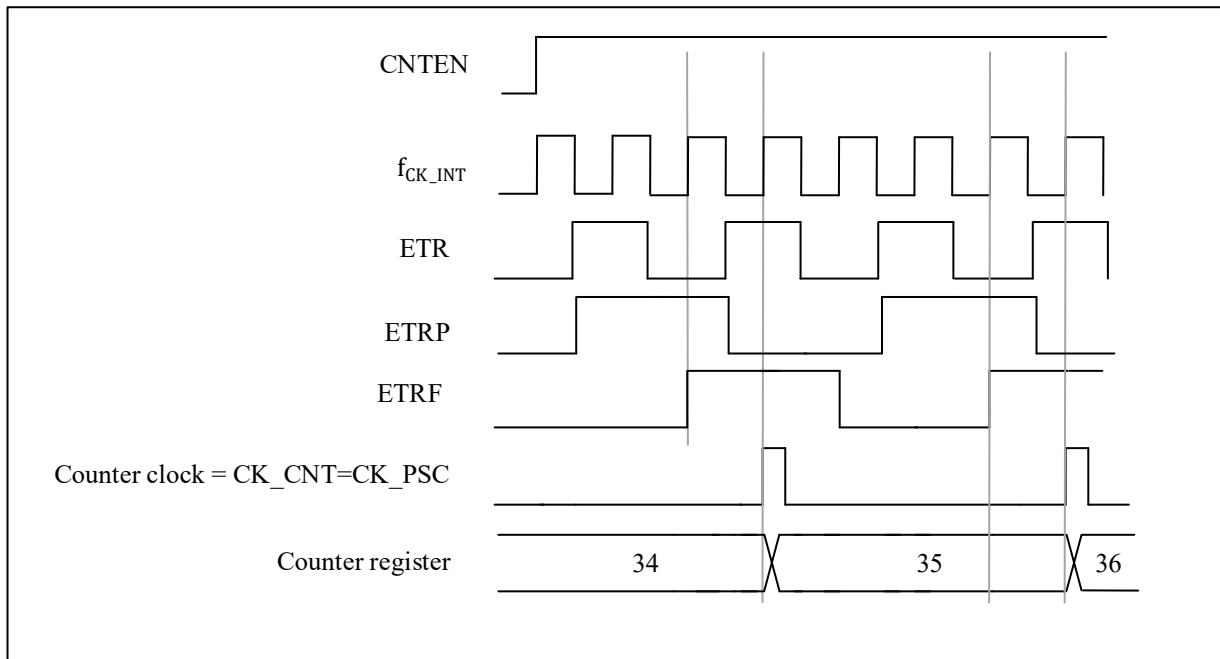


For example, use the following configuration steps to make the up counter count every 2 rising edges on ETR.

1. As no filter is needed in this case, setting TIMx_SMCTRL .EXTF[3:0] equal to '0000'
2. Configure the prescaler by setting TIMx_SMCTRL.EXTPS[1:0] equal to '01'
3. Select the polarity on ETR pin by setting TIMx_SMCTRL.EXTP equal to '0', The rising edge of ETR is valid
4. External clock mode 2 is selected by setting TIMx_SMCTRL .EXCEN equal to '1'
5. Enable the counter by setting TIMx_CTRL1. CNTEN equal to '1'

The counter counts every 2 rising edges of ETR. The delay between the rising edge of ETR and the actual clock of the counter is due to a resynchronization circuit on the ETRP signal.

Figure 10-12 Control Circuit in External Clock Mode 2



10.3.4 Capture/Compare Channels

Capture/compare channels include capture/compare registers and shadow registers. The input section consists of digital filters, multiplexers and prescalers. The output section includes comparators and output controls.

The input signal T_{ix} is sampled and filtered to generate the signal T_{ixF}. Then a signal (T_{ixF}_rising or T_{ixF}_falling) is generated by the edge detector of the polarity select function, the polarity of which is selected by the TIM_x_CCEN.CC_xP bits. This signal can be used as a trigger input for the slave mode controller. At the same time, the signal IC_x is sent to the capture register after prescaler. The following figure shows a block diagram of a capture/compare channel.

Figure 10-13 Capture/Compare Channel (Example: Channel 1 Input Stage)

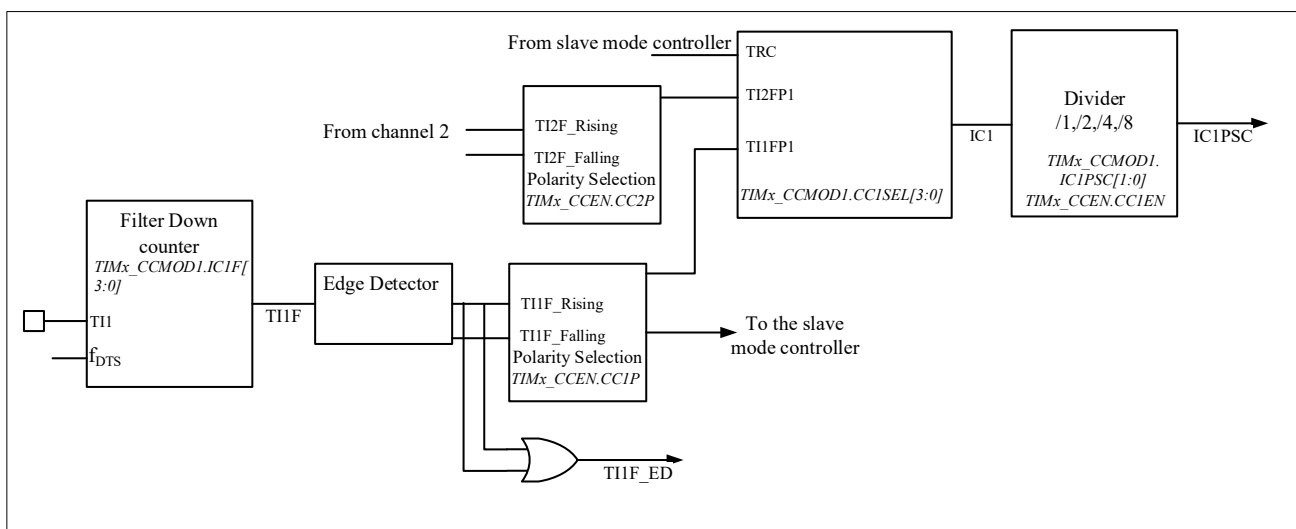
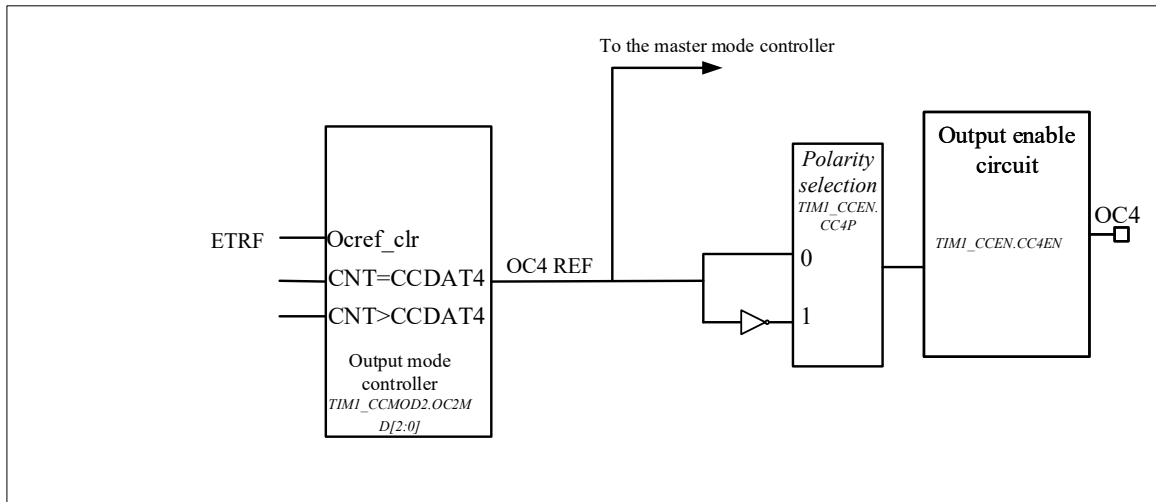


Figure 10-15 Output Part of Channel (X = 1,2,3,4; Take Channel 4 as An Example)



Reads and writes always access preloaded registers when capturing/comparing. The two specific operating processes are as follows:

In capture mode, the capture is actually done in the shadow register, which is copied into the preload register.

In compare mode, as opposed to capture mode, the value of the preload register is copied into the shadow register, which is compared with the counter.

10.3.5 Input Capture Mode

In capture mode, the TIMx_CCDA Tx registers are used to latch the counter value after the ICx signal detects.

There is a capture interrupt flag TIMx_STS.CCxITF, which can trigger an interrupt or DMA request if the corresponding interrupt enable is pulled high.

The TIMx_STS.CCxITF bit is set by hardware when a capture event occurs and is cleared by software or by reading the TIMx_CCDA Tx register.

The overcapture flag TIMx_STS.CCxOCF is set equal to 1 when the counter value is captured in the TIMx_CCDA Tx register and TIMx_STS.CCxITF is already pulled high. Unlike the former, TIMx_STS.CCxOCF is cleared by writing 0 to it.

To achieve a rising edge of the TI1 input to capture the counter value into the TIMx_CCDA T1 register, the configuration flow is as follows:

1. Select the valid input:

Configure TIMx_CCMOD1.CC1SEL to '01'. At this time, the input is the CC1 channel, and IC1 is mapped to TI1.
2. The duration of the input filter required for programming:

Define the sampling frequency of the TI1 input and the length of the digital filter by configuring the TIMx_CCMODx.ICxF bits. Example: If the input signal jitters up to 5 internal clock cycles, we must program a filter duration longer than these 5 clock cycles. When 8 consecutive samples (sampled at f_{DTs} frequency) with the new level are detected, we can validate the transition on TI1. Then configure TIMx_CCMOD1.IC1F to '0011'.
3. Configure TIMx_CCEN.CC1P=0, select the rising edge as the valid transition polarity on the TI1 channel.

4. Configure the input prescaler. In this example, configure `TIMx_CCMOD1.IC1PSC= '00'` to disable the prescaler because we want to capture every valid transition.
5. Enable capture by configuring `TIMx_CCEN. CC1EN = '1'`.

If you want to enable DMA request, you can configure `TIMx_DINTEN.CC1DEN=1`. If you want to enable related interrupt request, you can configure `TIMx_DINTEN.CC1IEN bit=1`

10.3.6 PWM Input Mode

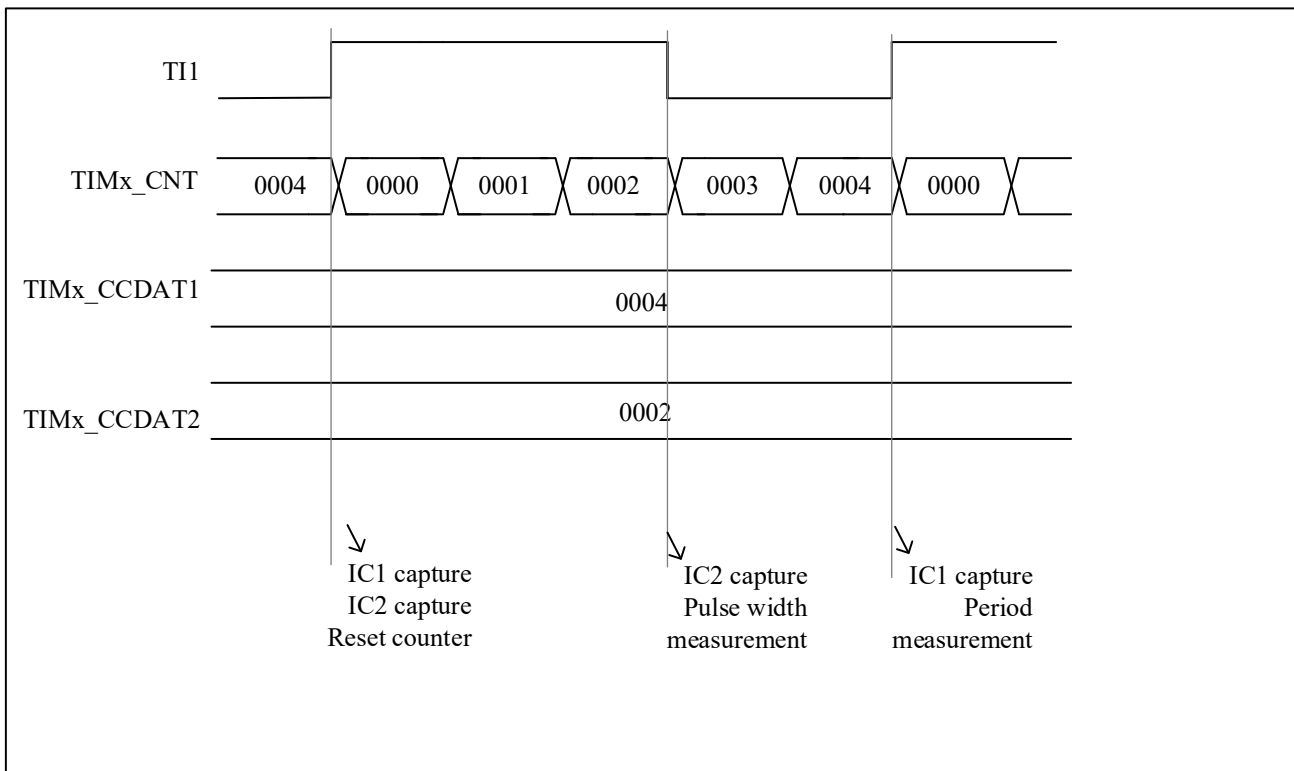
There are some differences between PWM input mode and normal input capture mode, including:

- Two ICx signals are mapped to the same TIx input.
- Two ICx signals are active on edges of opposite polarity.
- Select one of two TIxFP signals as trigger input.
- The slave mode controller is configured in reset mode.

For example, the following configuration flow can be used to know the period and duty cycle of the PWM signal on TI1 (It depends on the frequency of `CK_INT` and the value of the prescaler).

1. Configure `TIMx_CCMOD1.CC1SEL` equal to '01' to select TI1 as valid input for `TIMx_CCDAT1`.
2. Configure `TIMx_CCEN.CC1P` equal to '0' to select the active polarity of filtered timer input 1 (TI1FP1), active at the rising edge.
3. Configure `TIMx_CCMOD1.CC2SEL` equal to '10' to select TI1 as valid input for `TIMx_CCDAT2`.
4. Configure `TIMx_CCEN.CC2P` equal to 1 to select the valid polarity of filtered timer input 2 (TI1FP2), active at the falling edge.
5. Configure `TIMx_SMCTRL.TSEL=101` to select Filtered timer input 1 (TI1FP1) as valid trigger input.
6. Configure `TIMx_SMCTRL.SMSEL=100` to configure the slave mode controller to reset mode.
7. Configure `TIMx_CCEN. CC1EN=1` and `TIMx_CCEN.CC2EN=1` to enable capture.

Figure 10-16 PWM Input Mode Timing



Because of only filter timer input 1 (TI1FP1) and filter timer input 2 (TI2FP2) are connected to the slave mode controller, the PWM input mode can only be used with the TIMx_CH1/TIMx_CH2 signals.

10.3.7 Forced Output Mode

In output mode (TIMx_CCMODx.CCxSEL=00), the output compare signal can be forced to active or inactive level directly by software.

TIMx_CCMODx.OCxMD=101 can be set to force the output compare signal to active level. And the OCxREF will be forced high, OCx get opposite value to CcxP polarity bit. On the other hand, user can set TIMx_CCMODx.OCxMD=100 to force the output compare signal to inactive level.

In this mode, the values of the TIMx_CCDATx shadow register and the counter still comparing with each other.

The comparison between the output compare register TIMx_CCDATx and the counter TIMx_CNT has no effect on OCxREF. And the flag still can be set. Therefore, the interrupt and DMA requests still can be transmitted.

10.3.8 Output Compare Mode

This function is used to control the output waveform, or to indicate that a period of time has elapsed.

When the capture/compare register and the counter have the same value, the output compare function's operations are as follow:

1. TIMx_CCMODx.OCxMD is for output compare mode, and TIMx_CCEN.CCxP is for output polarity. When the compare matches, if set TIMx_CCMODx.OCxMD=000, the output pin will keep its level; if set TIMx_CCMODx.OCxMD=001, the output pin will be set active; if set TIMx_CCMODx.OCxMD=010, the output pin will be set inactive; if set TIMx_CCMODx.OCxMD=011, the output pin will be set to toggle.

2. Set TIMx_STS.CCxITF.
3. If user set TIMx_DINTEN.CCxIEN, a corresponding interrupt will be generated.
4. If user set TIMx_DINTEN.CCxDEN and set TIMx_CTRL2.CCDSEL to select DMA request, and DMA request will be sent.

User can set TIMx_CCMODx.OCxPEN to choose capture/compare shadow register using capture/compare preload registers(TIMx_CCDATx) or not.

The timing resolution is one count of the counter.

In one pulse mode, the output compare mode can also be used to output a single pulse.

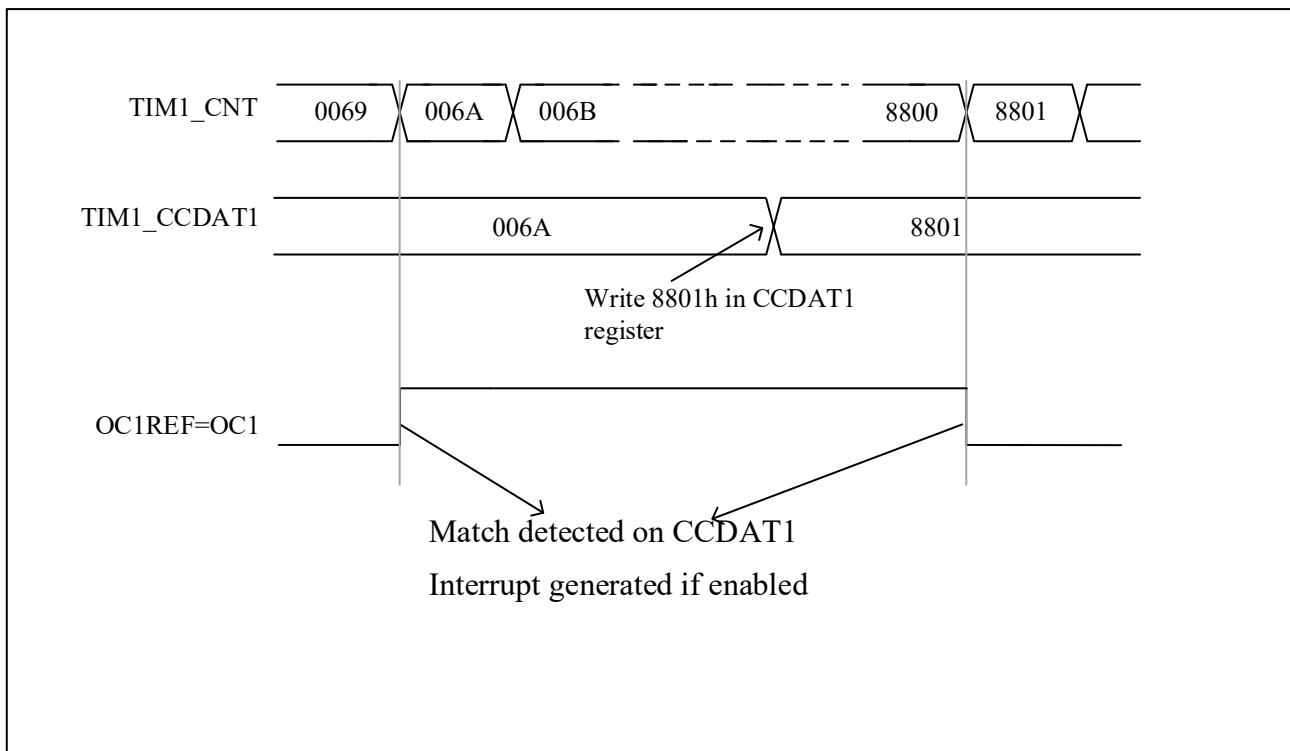
Here are the configuration steps for output compare mode:

1. First of all, user should select the counter clock.
2. Secondly, set TIMx_AR and TIMx_CCDATx with desired data.
3. Set TIMx_DINTEN.CCxIEN if user need to generate an interrupt.
4. Then select the output mode by set TIMx_CCEN.CCxP, TIMx_CCMODx.OCxMD, TIMx_CCEN.CCxEN, etc.
5. At last, set TIMx_CTRL1.CNTEN to enable the counter.

User can update the output waveform by setting TIMx_CCDATx at any time, as long as the preload register is not enabled. Otherwise the TIMx_CCDATx shadow register will be updated at the next update event.

Here is an example.

Figure 10-17 Output Compare Mode, Toggle on OC1



10.3.9 PWM Mode

User can use PWM mode to generate a signal whose duty cycle is determined by the value of the TIMx_CCxDATx register and whose frequency is determined by the value of the TIMx_AR register. And depends on the value of TIMx_CTRL1.CAMSEL, the TIM can generate PWM signal in edge-aligned mode or center-aligned mode.

User can set PWM mode 1 or PWM mode 2 by setting TIMx_CCMODx. OCxMD=110 or setting TIMx_CCMODx. OCxMD=111. To enable preload register, user must set corresponding TIMx_CCMODx.OCxPEN. Then set TIMx_CTRL1.ARPEN to auto-reload preload register.

User can set polarity of OCx by setting TIMx_CCEN.CCxP..

The values of TIMx_CNT and TIMx_CCxDATx are always compared with each other when the TIM is under PWM mode.

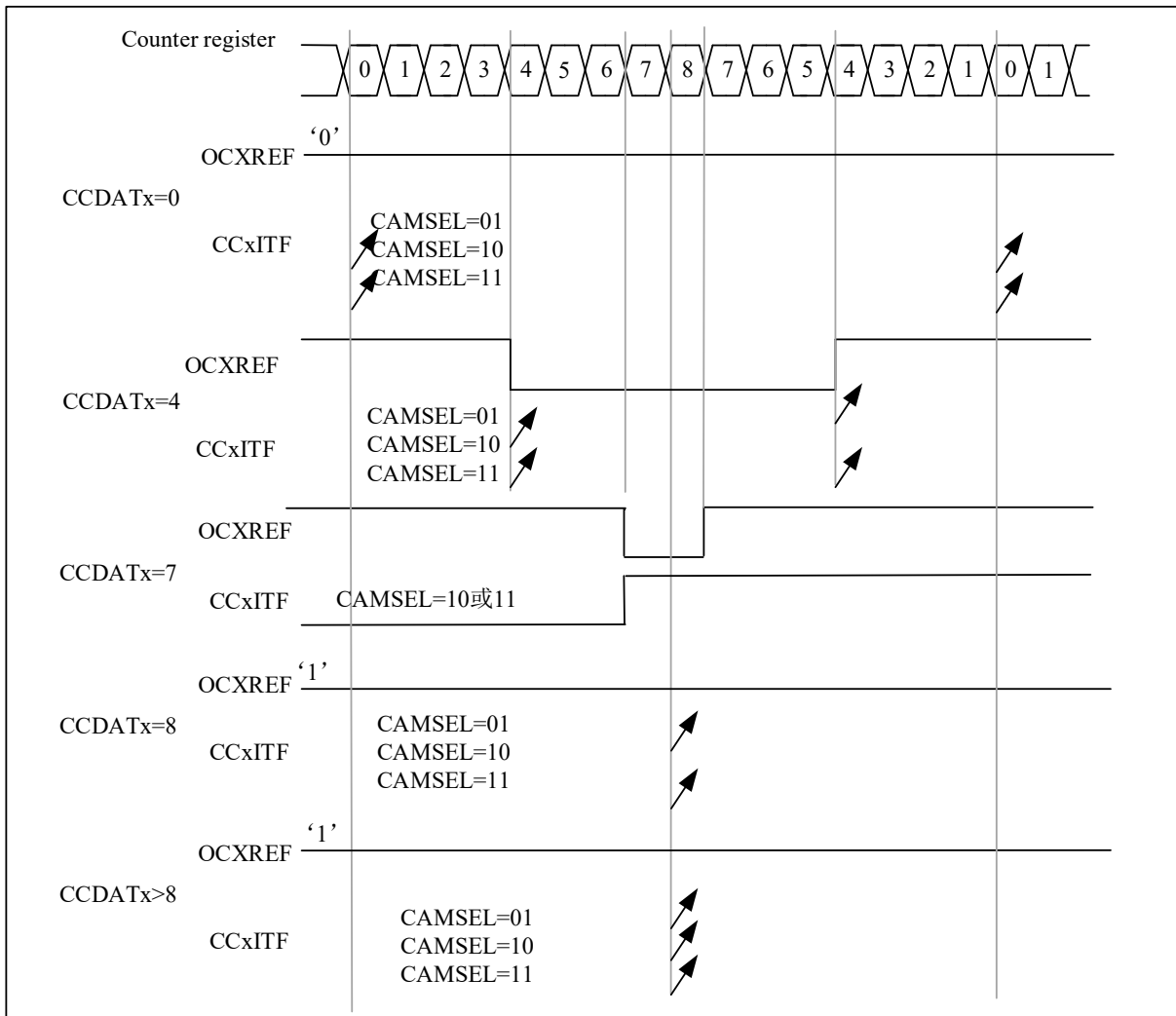
Only if an update event occurs, the preload register will transfer to the shadow register. Therefore user must reset all the registers by setting TIMx_EVTGEN.UDGN before the counter starts counting.

PWM center-aligned mode

The PWM center-aligned mode is active when setting TIMx_CTRL1.CAMSEL equal 01, 10 or 11. The setting of the compare flag depends on the value of TIMx_CTRL1.CAMSEL. There are three kinds of situation that the compare flag is set, only when the counter counts up, only when the counter counts down, or when the counter counts up and counts down. The TIMx_CTRL1.DIR is updated by software and must not be changed by software.

The example of center-aligned PWM waveforms is as follow, and the setting of the waveform are: TIMx_AR=8, PWM mode 1, the compare flag is set when the counter counts down corresponding to TIMx_CTRL1. CAMSEL=01.

Figure 10-18 Center-aligned PWM Waveform (AR=8)



When using center-aligned mode, users should pay attention to the following considerations:

- It depends on the value of TIMx_CTRL1.DIR that the counter counts up or down. Caution that the DIR and CAMSEL bits should not be changed at the same time.
- User should not write the counter while running in center-aligned mode, otherwise it will cause unexpected results. Here are some examples:
 - If the value written into the counter is 0 or is the value of TIMx_AR, the direction will be updated but the update event will not be generated.
 - If the value written into the counter is greater than the value of auto-reload, the direction will not be updated.
- To be on the safe side, user is suggested setting TIMx_EVTGEN.UDGN to generate an update by software before starting the counter, and not writing the counter while it is running.

PWM edge-aligned mode

There are two kinds of configuration in edge-aligned mode, up-counting and down-counting.

- **Up-counting**

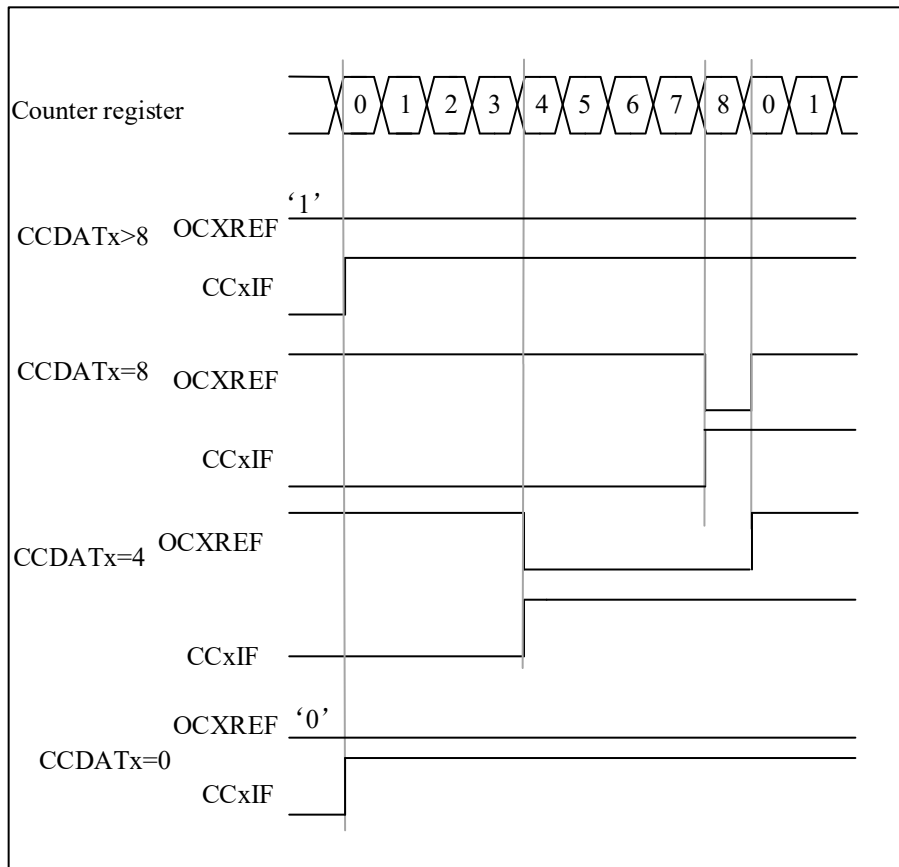
User can set `TIMx_CTRL1.DIR=0` to make counter counts up.

Example for PWM mode1.

When `TIMx_CNT < TIMx_CCDA Tx`, the reference PWM signal `OCxREF` is high. Otherwise it will be low. If the compare value in `TIMx_CCDA Tx` is greater than the auto-reload value, the `OCxREF` will remains 1. Conversely, if the compare value is 0, the `OCxREF` will remains 0.

When `TIMx_AR=8`, the PWM waveforms are as follow.

Figure 10-19 Edge-Aligned PWM Waveform (APR=8)



- Down-counting**

User can set `TIMx_CTRL1.DIR=1` to make counter counts down.

Example for PWM mode1.

When `TIMx_CNT > TIMx_CCDA Tx`, the reference PWM signal `OCxREF` is low. Otherwise it will be high. If the compare value in `TIMx_CCDA Tx` is greater than the auto-reload value, the `OCxREF` will remains 1.

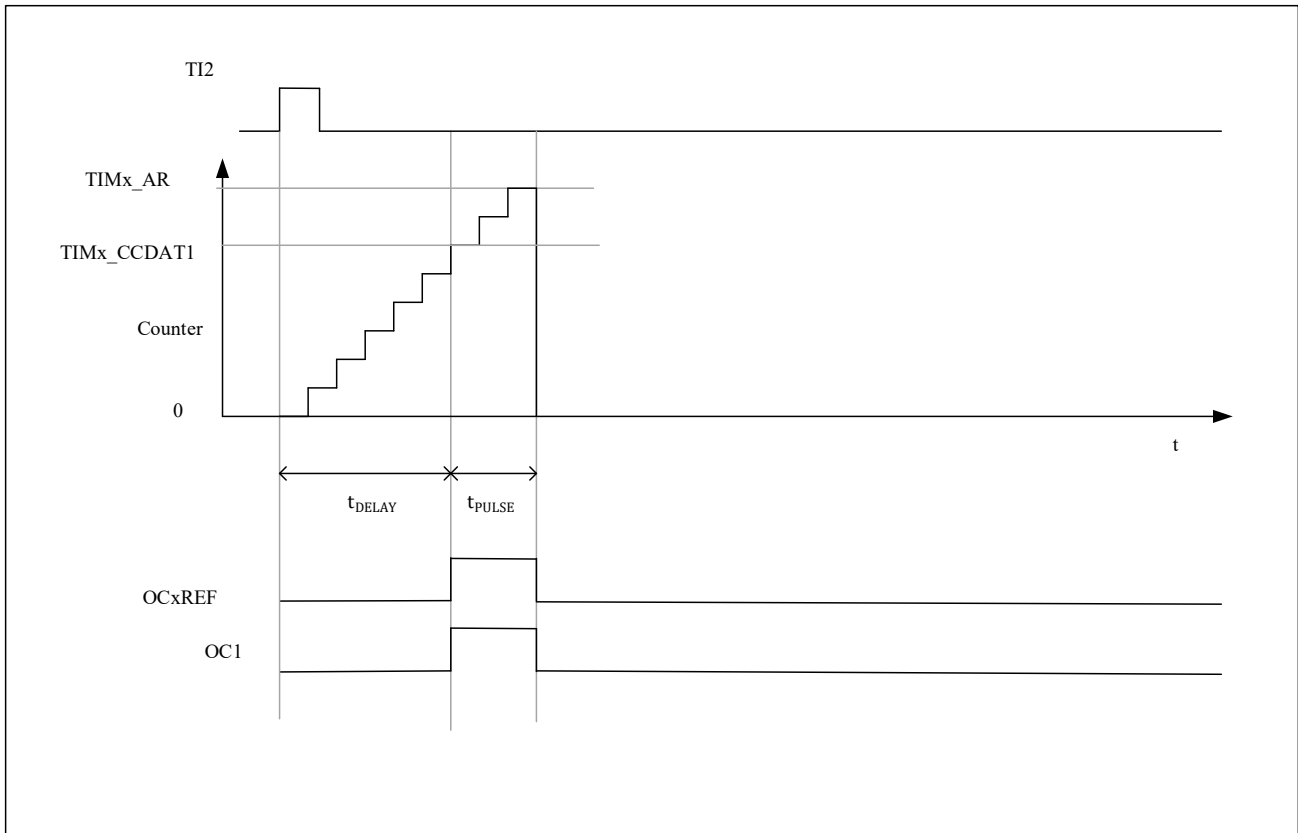
Note: If the nth PWM cycle CCDA Tx shadow register >= AR value, the shadow register value of CCDA Tx in the (n+1)th PWM cycle is 0. At the moment when the counter is 0 in the (n+1)th PWM cycle, although the value of the counter = CCDA Tx shadow register = 0 and OCxREF = '0', no compare event will be generated.

10.3.10 One-pulse Mode

In the one-pulse mode (ONEPM), a trigger signal is received, and a pulse `tPULSE` with a programmable pulse length is generated after a programmable delay `tDELAY`. The output mode needs to be configured as output compare mode or PWM

mode. After selecting one-pulse mode, the counter will stop counting after the update event UEV is generated.

Figure 10-20 Example of One-pulse Mode



The following is an example of the one-pulse mode:

A rising edge trigger is detected from the TI2 input, and a pulse with a length of t_{PULSE} is generated on OC1 after a delay of t_{DELAY} .

1. Counter configuration: count up, counter $TIMx_CNT < TIMx_CCDAT1 \leq TIMx_AR$;
2. TI2FP2 is mapped to TI2, $TIMx_CCMOD1.CC2SEL = '01'$; TI2FP2 is configured for rising edge detection, $TIMx_CCEN.CC2P = '0'$;
3. TI2FP2 acts as the trigger (TRGI) of the slave mode controller and starts the counter, $TIMx_SMCTRL.TSEL = '110'$, $TIMx_SMCTRL.SMSEL = '110'$ (trigger mode);
2. The count value to be delayed (t_{DELAY}) is written to TIMx_CCDAT1, $TIMx_AR - TIMx_CCDAT1$ is the count value of the pulse length t_{PULSE} ;
3. Configure $TIMx_CTRL1.ONEPM = 1$ to enable single pulse mode, configure $TIMx_CCMOD1.OC1MD = '111'$ to select PWM2 mode;
4. Wait for an external trigger event on TI2, and a one pulse waveform will be output on OC1;

Special case: OCx fast enable:

In one-pulse mode, an edge is detected through the TIX input, and triggers the start of the counter to count to the comparison value and then output a pulse. These operations limit the minimum delay t_{DELAY} that can be achieved.

You can set `TIMx_CCMODx.OCxFEN=1` to turn on OCx fast enable, the OCxREF signal will be forced to be converted to the same level as the comparison match occurs immediately after triggering the rising edge, no matter what the comparison result is. OCxFEN fast enable takes effect only when the channel mode is configured for PWM1 and PWM2 modes.

10.3.11 Clearing The Ocxref Signal on An External Event

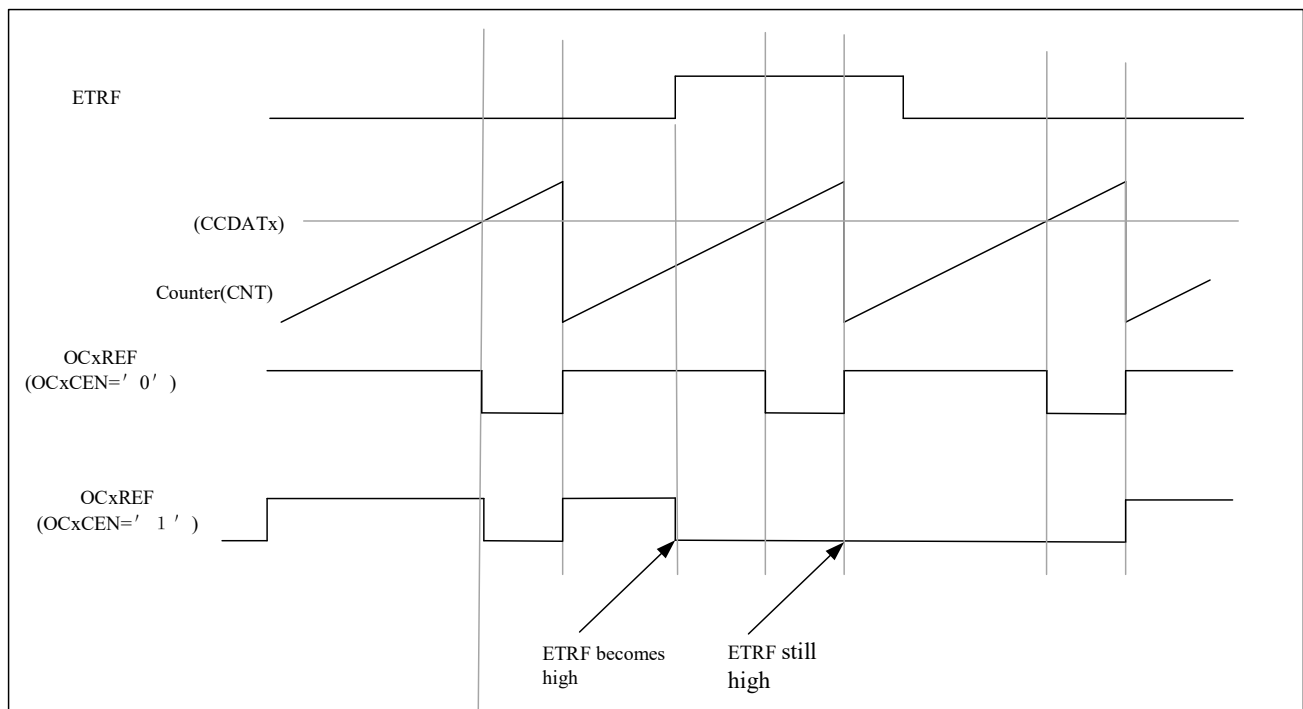
If `TIMx_CCMODx.OCxCEN=1`, high level of ETRF input can be used to driven the OCxREF signal to low, and the OCxREF signal will remains low, until the next UEV happens. Only output compare and PWM modes can use this function. This cannot be used when it is in forced mode.

Here is an example for it. The ETR signal is connected to the output of a comparator to control the current. The operation for ETR should be as follow:

1. Set `TIMx_SMCTRL.EXTPS=00` to disable the external trigger prescaler.
2. Set `TIMx_SMCTRL.EXCEN=0` to disable the external clock mode 2.
3. Set `TIMx_SMCTRL.EXTP` and `TIMx_SMCTRL.EXTF` to configure the external trigger polarity and external trigger filter according to the need.

Here is an example for the case that the behavior of OCxREF signal for different value of OCxCEN when ETRF input becomes high. Timer is set to be in PWM mode in this case.

Figure 10-21 Control Circuit in Reset Mode



10.3.12 Debug Mode

When the microcontroller is in debug mode (the Cortex[®]-M0 core halted), the TIMx counter can either continue to operate normally or stop depending on the `DBG_CTRL.TIMx_STOP` configuration in the PWR module,. For more details, refer to Section 3.3.2.

10.3.13 TIMx and External Trigger Synchronization

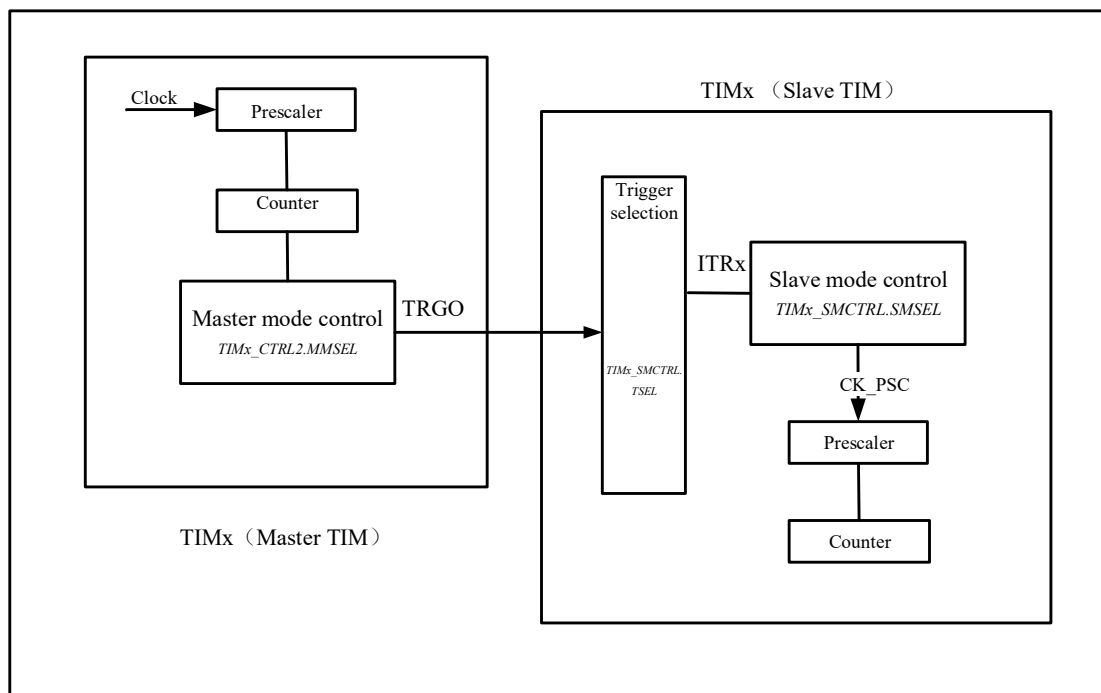
Same with advanced-control timer, refer to Section 9.3.16

10.3.14 Timer Synchronization

All TIMx timers are internally interconnected to each other. This implementation allows a master timer to provide trigger to reset, start, stop or provide a clock for the other slave timers. The master clock is used for internal counter and can be prescaled. Below figure shows a block diagram of timer interconnection.

The synchronization function does not support dynamic change of the connection. User should configure and enable the slave timer before enabling the master timer's trigger or clock.

Figure 10-22 Block Diagram of Timer Interconnection



Master timer as a prescaler for another timer

TIM1 as a prescaler for TIM3. TIM1 is maser timer, TIM3 is slave timer.

User needs to do the following steps for this configuration.

1. Setting TIM1_CTRL2.MMSEL='010' to use the update event of TIM1 as trigger output.
2. Configure TIM3_SMCTRL.TSEL= '000' to connect the TRGO of TIM1 to TIM3.
3. Configure TIM3_SMCTRL.SMSEL = '111', the slave mode controller will be configured in external clock mode 1.
4. Start TIM3 by setting TIM3_CTRL1.CNTEN = '1'.
5. Start TIM1 by setting TIM1_CTRL1.CNTEN = '1'.

Note: If user select OCx as the trigger output of TIM1 by configuring MMSEL = '1xx', OCx rising edge will be used to drive TIM2.

Master timer to enable another timer

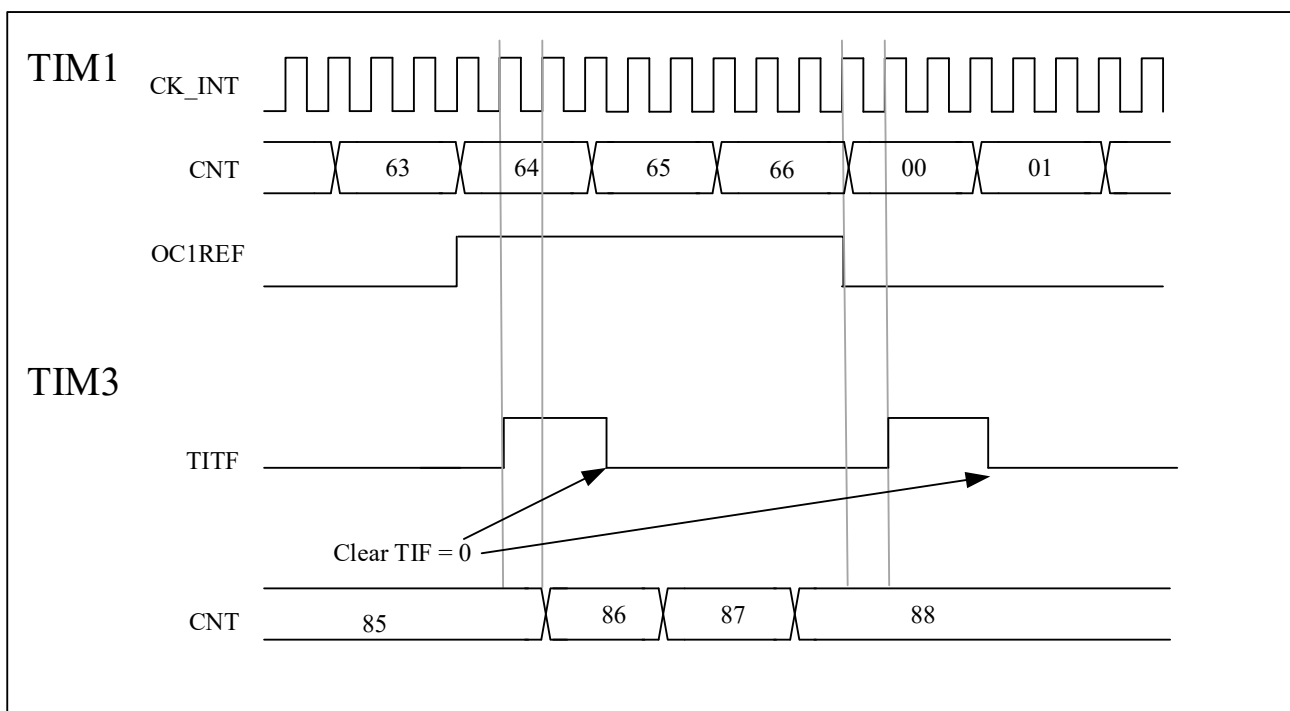
In this example, TIM3 is enabled by the output compare of TIM1. TIM3 counter will start to count after the OC1REF output from TIM1 is high. Both counters are clocked based on CK_INT via a prescaler divide by 3 is performed ($f_{CK_CNT} = f_{CK_INT}/3$).

The configuration steps are shown as below.

1. Setting TIM1_CTRL2.MMSEL='100' to use the OC1REF of TIM1 as trigger output.
2. Configure TIM1_CCMOD1 register to configure the OC1REF output waveform.
3. Setting TIM3_SMCTRL.TSEL = '000' to connect TIM1 trigger output to TIM3.
4. Setting TIM3_SMCTRL.SMSEL= '101' to set TIM3 in gated mode.
5. Setting TIM3_CTRL1.CNTEN= '1' to start TIM3.
6. Setting TIM1_CTRL1.CNTEN= '1' to start TIM1.

Note: The TIM3 clock is not synchronized with the TIM1 clock, this mode only effects the TIM3 counter enable signal.

Figure 10-23 TIM3 Gated by OC1REF Of TIM1



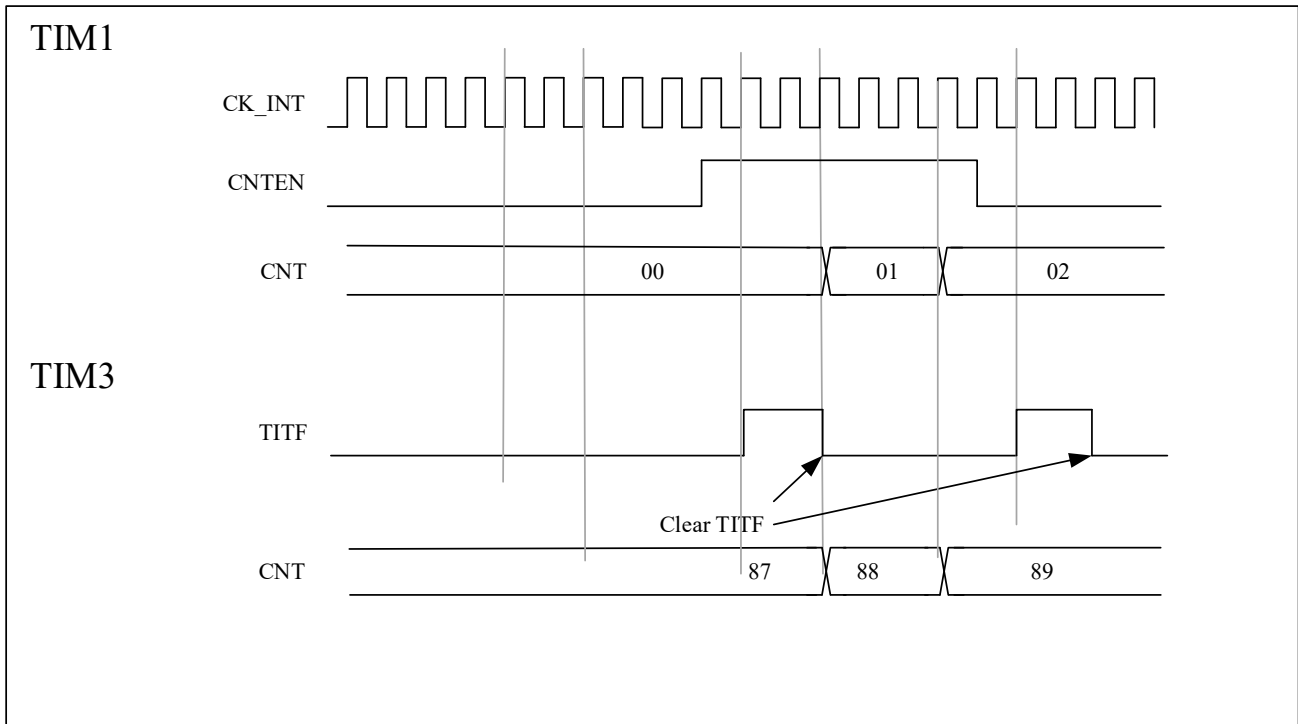
In the next example, The enable signal of TIM1 is used for Gated TIM3, Setting TIM1_CTRL1.CNTEN = '0' to stop TIM1. TIM3 counts on the divided internal clock only when TIM1 is enable. Both counters are clocked based on CK_INT via a prescaler divide by 3 is performed ($f_{CK_CNT} = f_{CK_INT}/3$).

The configuration steps are shown as below

1. Setting TIM1_CTRL2.MMSEL='001' to use the enable signal of TIM1 as trigger output
2. Setting TIM3_SMCTRL.TSEL = '000' to configure TIM3 to get the trigger input from TIM1
3. Setting TIM3_SMCTRL.SMSEL = '101' to configure TIM3 in gated mode.

4. Setting TIM3_CTRL1.CNTEN= '1' to start TIM3.
5. Setting TIM1_CTRL1.CNTEN= '1' to start TIM1.
6. Setting TIM1_CTRL1.CNTEN= '0' to stop TIM1.

Figure 10-24 TIM3 Gated by Enable Signal of TIM1



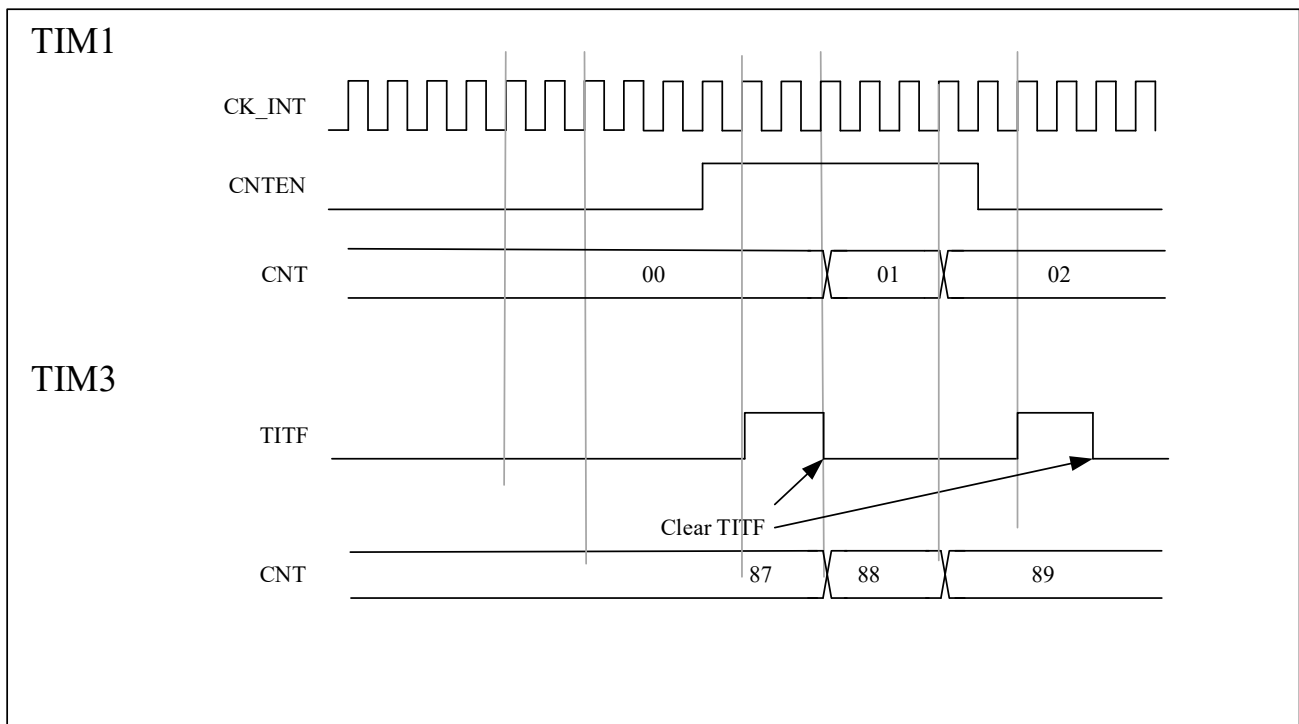
Master timer to start another timer

In this example, we can use update event as trigger source. TIM1 is master timer, TIM3 is slave timer.

The configuration steps are shown as below:

1. Setting TIM1_CTRL2.MMSEL='010' to use the update event of TIM1 as trigger output
2. Configure TIM1_AR register to set the output period.
3. Setting TIM3_SMCTRL.TSEL='000' to connect TIM1 trigger output to TIM3.
4. Setting TIM3_SMCTRL.SMSEL='110' to set TIM3 in trigger mode.
5. Setting TIM1_CTRL1.CNTEN=1 to start TIM1.

Figure 10-25 Trigger TIM3 with An Update of TIM1



Start 2 timers synchronously using an external trigger

In this example, TIM1 is enabled when TIM1's TI1 input rises, and TIM3 is enabled when TIM1 is enabled. To ensure the counters are aligned, TIM1 must be configured in master/slave mode. For TI1, TIM1 is the slave; for TIM3, TIM1 is the master.

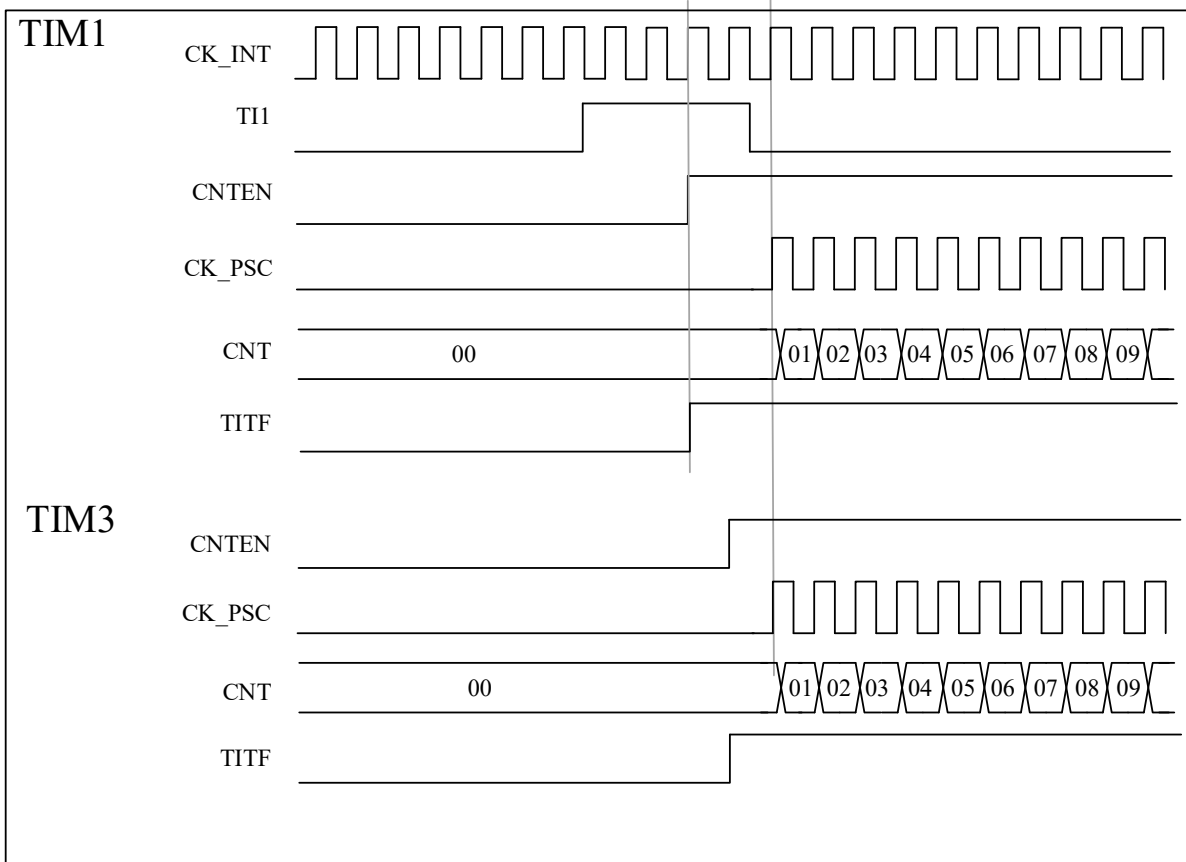
The configuration steps are shown as below:

1. Setting `TIM1.MMSEL = '001'` to use the enable signal as trigger output
2. Setting `TIM1_SMCTRL.TSEL = '100'` to configure the TIM1 in slave mode and receive the trigger input of TI1.
3. Setting `TIM1_SMCTRL.SMSEL = '110'` to configure TIM1 in trigger mode.
4. Setting `TIM1_SMCTRL.MSMD = '1'` to configure TIM1 in master/slave mode.
5. Setting `TIM3_SMCTRL.TSEL = '000'` to connect TIM1 trigger output to TIM3.
6. Setting `TIM3_SMCTRL.SMSEL = '110'` to configure TIM3 in trigger mode.

When TI1 rising edge arrives, both timers start counting synchronously according to the internal clock, and both TITF flags are set simultaneously.

The following figure shows a delay between CNTEN and CK_PSC of TIM1 in master/slave mode.

Figure 10-26 Triggers Timers 1 and 3 Using The TI1 Input of TIM1



10.3.15 Encoder Interface Mode

The encoder uses two inputs TI1 and TI2 as an interface and the counter counts on every edge change on TI1FP1 or TI2FP2. The counting direction is automatically controlled by hardware TIMx_CTRL1.DIR. There are three types of encoder counting modes:

- The counter only counts on the edge of TI1, TIMx_SMCTRL.SMSEL = '001';
- The counter only counts on the edge of TI2, TIMx_SMCTRL.SMSEL = '010';
- The counter counts on the edges of TI1 and TI2 at the same time, TIMx_SMCTRL.SMSEL = '011';

The encoder interface is equivalent to using an external clock with direction selection, and the counter only counts continuously between 0 and the auto-reload value (TIMx_AR.AR [15:0]). Therefore, it is necessary to configure the auto-reload register TIMx_AR in advance.

Note: Encoder mode and external clock mode 2 are not compatible and must not be selected together.

The relationship between the counting direction and the encoder signal is shown in Table 10-1:

Table 10-1 Counting Direction Versus Encoder Signals

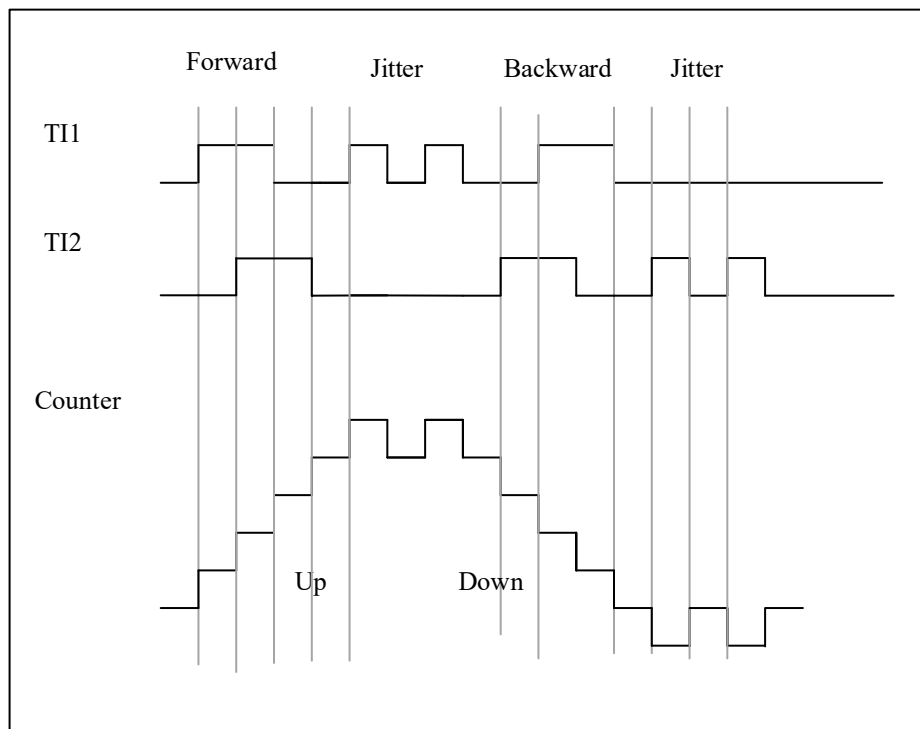
Active Edge	Level On Opposite Signals (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 Signal		TI2FP2 Signal	
		Rising	Falling	Rising	Falling

Counting only at TI1	High	Counting down	Counting up	Don't count	Don't count
	Low	Counting up	Counting down	Don't count	Don't count
Counting only at TI2	High	Don't count	Don't count	Counting up	Counting down
	Low	Don't count	Don't count	Counting down	Counting up
Counting on TI1 and TI2	High	Counting down	Counting up	Counting up	Counting down
	Low	Counting up	Counting down	Counting down	Counting up

Here is an example of an encoder with dual edges triggering selected to suppress input jitter:

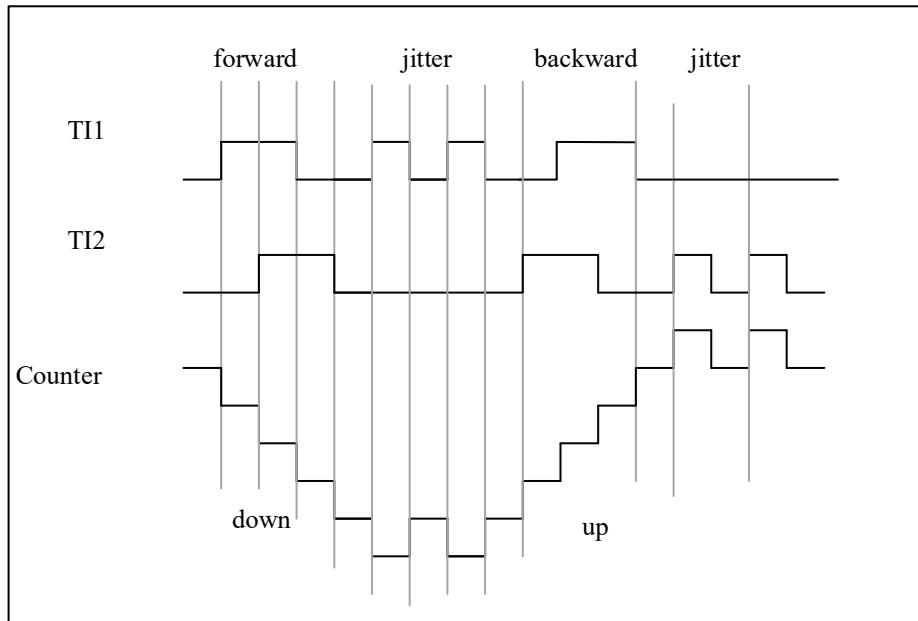
1. IC1FP1 is mapped to TI1 (TIMx_CCMOD1.CC1SEL= '01'), IC1FP1 is not inverted (TIMx_CCEN.CC1P= '0');
2. IC1FP2 is mapped to TI2 (TIMx_CCMOD2.CC2SEL= '01'), IC2FP2 is not inverted (TIMx_CCEN.CC2P= '0');
3. The input is active at both rising and falling edges (TIMx_SMCTRL.SMSEL = '011');
4. Enable counter TIMx_CTRL1.CNTEN= '1';

Figure 10-27 Example of Counter Operation in Encoder Interface Mode



The following figure shows the example of counter behavior when IC1FP1 polarity is inverted (CC1P= '1', other configurations are the same as above)

Figure 9-43 Encoder Interface Mode Example with IC1FP1 Polarity Inverted



10.3.16 Interfacing with Hall Sensor

Please refer to Section 9.3.20

10.4 TIMx Register Description(x=3 And 4)

For abbreviations used in registers, refer to Section 1.1

All register operations must be performed in half word (16-bits) or one word (32-bits).

10.4.1 Register Overview

Table 10-2 Register Overview

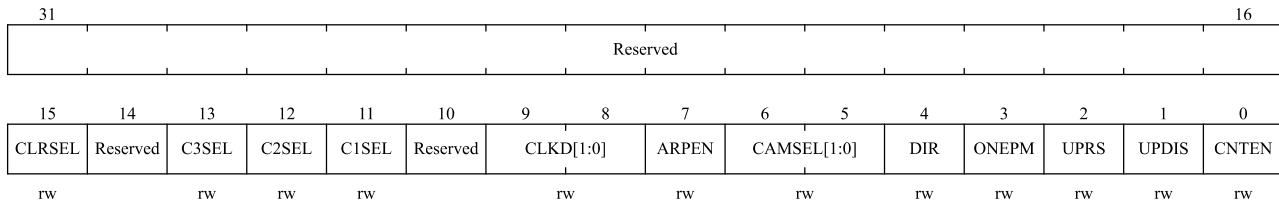
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0											
000h	TIMx_CTRL1	Reserved																CLRSEL	Reserved	C3SEL	C2SEL	C1SEL	Reserved	CLKD[1:0]		ARPEN	CAMSEL[1:0]			DIR	ONEPM	UPRS	UPDIS	CNTEN										
	Reset Value	0																0		0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
004h	TIMx_CTRL2	Reserved																ETRSEL		TI1SEL	MMSEL[2:0]			CCDSEL	Reserved																			
	Reset Value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
008h	TIMx_SMCTRL	Reserved																EXTP	EXCEN	EXTPS[1:0]		EXTIF[3:0]			MSMD	TSEL[2:0]			Reserved	SMSELEL[2:0]														
	Reset Value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00Ch	TIMx_DINTEN	Reserved																TDEN	Reserved	CC4DEN	CC3DEN	CC2DEN	CC1DEN	UDEN	Reserved	TI1EN	Reserved	CC4IEN	CC3IEN	CC2IEN	CC1IEN	UIEN												
	Reset Value	0																0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
010h	TIMx_STS	Reserved																CC4OCF	CC3OCF	CC2OCF	CC1OCF	Reserved			TITF	Reserved			CC4ITF	CC3ITF	CC2ITF	CC1ITF	UDITF
	Reset Value	0																0	0	0	0	0			0	0			0	0	0	0	0
014h	TIMx_EVTGEN	Reserved																			TGN	Reserved			CC4GN	CC3GN	CC2GN	CC1GN	UDGN				
	Reset Value	0																0			0	0			0	0	0	0	0				
018h	TIMx_CCMOD1 Output compare	Reserved																OC2CEN	OC2MD[2:0]		OC2PEN	OC2FEN	CC2SEL[1:0]		OC1CEN	OC1MD[2:0]		OC1PEN	OC1FEN	CC1SEL[1:0]			
	Reset Value	0																0	0	0	0	0		0	0		0	0	0				
	TIMx_CCMOD1 Input capture	Reserved																IC2F[3:0]			IC2PSC[1:0]	CC2SEL[1:0]		IC1F[3:0]			IC1PSC[1:0]	CC1SEL[1:0]					
Reset Value	0																0			0	0		0			0	0						
01Ch	TIMx_CCMOD2 Output compare	Reserved																OC4CEN	OC4MD[2:0]		OC4PEN	OC4FEN	CC4SEL[1:0]		OC3CEN	OC3MD[2:0]		OC3PEN	OC3FEN	CC3SEL[1:0]			
	Reset Value	0																0	0	0	0	0		0	0		0	0	0				
01Ch	TIMx_CCMOD2 Input capture	Reserved																IC4F[3:0]			IC4PSC[1:0]	CC4SEL[1:0]		IC3F[3:0]			IC3PSC[1:0]	CC3SEL[1:0]					
	Reset Value	0																0			0	0		0			0	0					
020h	TIMx_CCEN	Reserved																CC4P	CC4EN	Reserved		CC3P	CC3EN	Reserved		CC2P	CC2EN	Reserved		CC1P	CC1EN		
	Reset Value	0																0	0	0		0	0	0		0	0	0		0	0		
024h	TIMx_CNT	Reserved																CNT[15:0]															
	Reset Value	0																0															
028h	TIMx_PSC	Reserved																PSC[15:0]															
	Reset Value	0																0															
02Ch	TIMx_AR	Reserved																AR[15:0]															
	Reset Value	1																1															
030h	Reserved																																
034h	TIMx_CCDAT1	Reserved																CCDAT1[15:0]															
	Reset Value	0																0															
038h	TIMx_CCDAT2	Reserved																CCDAT2[15:0]															
	Reset Value	0																0															
03Ch	TIMx_CCDAT3	Reserved																CCDAT3[15:0]															
	Reset Value	0																0															
040h	TIMx_CCDAT4	Reserved																CCDAT4[15:0]															
	Reset Value	0																0															
044h	Reserved																																
048h	TIMx_DCTRL	Reserved																DBLEN[4:0]				Reserved			DBADDR[4:0]								
	Reset Value	0																0				0			0								
04Ch	TIMx_DADDR	Reserved																BURST[15:0]															
	Reset Value	0																0															

10.4.2 Control Register 1 (TIMx_CTRL1)

Offset address: 0x00

Reset value: 0x0000 0000



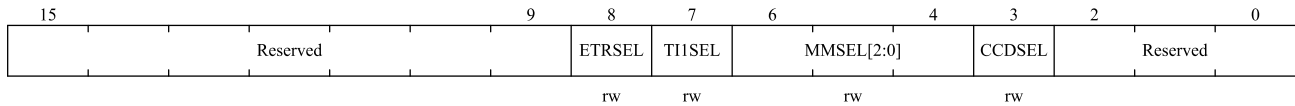
Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained
15	CLRSEL	OCxREF clear selection 0: Select the external OCxREF clear from ETR 1: Select the internal OCxREF clear from COMP(COMP1/COMP2/COMP3)
14	Reserved	Reserved, the reset value must be maintained
13	C3SEL	Channel 3 Selection 0: Select external CH3 (from IOM) signal 1: Select internal CH3 signal from COMP3
12	C2SEL	Channel 2 Selection 0: Select external CH2 (from IOM) signal 1: Select internal CH2 signal from COMP2
11	C1SEL	Channel 1 selection 0: Select external CH1 signal from IOM 1: Select internal CH1 signal from COMP1
10	Reserved	Reserved, the reset value must be maintained
9:8	CLKD[1:0]	Clock division CLKD[1:0] indicates the division ratio between CK_INT (timer clock) and tDTS (clock used for dead-time generator and digital filters (ETR, TIx)) 00: tDTS = tCK_INT 01: tDTS = 2 × tCK_INT 10: tDTS = 4 × tCK_INT 11: Reserved, do not use this configuration
7	ARPEN	ARPEN: Auto-reload preload enable 0: Shadow register disable for TIMx_AR register 1: Shadow register enable for TIMx_AR register
6:5	CAMSEL[1:0]	Center-aligned mode selection 00: Edge-aligned mode. TIMx_CTRL1.DIR specifies up-counting or down-counting. 01: Center-aligned mode 1. The counter counts in center-aligned mode, and the output compare interrupt flag bit is set to 1 when down-counting. 10: Center-aligned mode 2. The counter counts in center-aligned mode, and the output compare interrupt flag bit is set to 1 when up-counting.

Bit Field	Name	Description
		11: Center-aligned mode 3. The counter counts in center-aligned mode, and the output compare interrupt flag bit is set to 1 when up-counting or down-counting. <i>Note: Switching from edge-aligned mode to center-aligned mode is not allowed when the counter is still enabled (TIMx_CTRL1.CNTEN = 1).</i>
4	DIR	Direction 0: Up-counting 1: Down-counting <i>Note: This bit is read-only when the counter is configured in center-aligned mode or encoder mode.</i>
3	ONEPM	One-pulse mode 0: Disable one-pulse mode, the counter counts are not affected when an update event occurs. 1: Enable one-pulse mode, the counter stops counting when the next update event occurs (clearing TIMx_CTRL1.CNTEN bit)
2	UPRS	Update request source This bit is used to select the UEV event sources by software. 0: If update interrupt or DMA request is enabled, any of the following events will generate an update interrupt or DMA request: <ul style="list-style-type: none"> – Counter overflow/underflow – The TIMx_EVTGEN.UDGN bit is set – Update generation from the slave mode controller 1: If update interrupt or DMA request is enabled, only counter overflow/underflow will generate update interrupt or DMA request
1	UPDIS	Update disable This bit is used to enable/disable the Update event (UEV) events generation by software. 0: Enable UEV. And UEV will be generated if one of following condition been fulfilled: <ul style="list-style-type: none"> – Counter overflow/underflow – The TIMx_EVTGEN.UDGN bit is set – Update generation from the slave mode controller Shadow registers will update with preload value. 1: UEV disabled. No update event is generated, and the shadow registers (AR, PSC, and CCDATx) keep their values. If the TIMx_EVTGEN.UDGN bit is set or a hardware reset is issued by the slave mode controller, the counter and prescaler are reinitialized.
0	CNTEN	Counter Enable 0: Disable counter 1: Enable counter <i>Note: external clock, gating mode and encoder mode can only work after TIMx_CTRL1.CNTEN bit is set in the software. Trigger mode can automatically set TIMx_CTRL1.CNTEN bit by hardware.</i>

10.4.3 Control Register 2 (TIMx_CTRL2)

Offset address: 0x04

Reset value: 0x0000

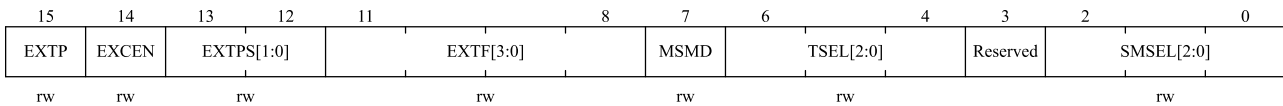


Bit Field	Name	Description
15:9	Reserved	Reserved, the reset value must be maintained
8	ETRSEL	External Triggered Selection storage (ETR Selection) 0: Select external ETR (from IOM) signal; 1: Reserved
7	TI1SEL	TI1 selection 0: TIMx_CH1 pin connected to TI1 input. 1: TIMx_CH1, TIMx_CH2, and TIMx_CH3 pins are XOR connected to the TI1 input.
6:4	MMSEL[2:0]	Master Mode Selection These 3 bits (TIMx_CTRL2.MMSEL [2:0]) are used to select the synchronization information (TRGO) sent to the slave timer in the master mode. Possible combinations are as follows: 000: Reset –When the TIMx_EVTGEN.UDGN is set or a reset is generated by the slave mode controller, a TRGO pulse occurs. And in the latter case, the signal on TRGO is delayed compared to the actual reset. 001: Enable - The TIMx_CTRL1.CNTEN bit is used as the trigger output (TRGO). Sometimes you need to start multiple timers at the same time or enable slave timer for a period of time. The counter enable signal is set when TIMx_CTRL1.CNTEN bit is set or the trigger input in gated mode is high. 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 (refer to the description of the TIMx_SMCTRL.MSMD bit). 010: Update - The update event is selected as the trigger output (TRGO). For example, a master timer clock can be used as a slave timer prescaler. 011: Compare pulse - Triggers the output to send a positive pulse (TRGO) when the TIMx_STS.CC1ITF is to be set (even if it is already high), when a capture or a comparison succeeds. 100: Compare - OC1REF signal is used as the trigger output (TRGO). 101: Compare - OC2REF signal is used as the trigger output (TRGO). 110: Compare - OC3REF signal is used as the trigger output (TRGO). 111: Compare - OC4REF signal is used as the trigger output (TRGO).
3	CCDSEL	Capture/compare DMA selection 0: When a CCx event occurs, a DMA request for CCx is sent. 1: When an update event occurs, a DMA request for CCx is sent.
2:0	Reserved	Reserved, the reset value must be maintained

10.4.4 Slave Mode Control Register (TIMx_SMCTRL)

Offset address: 0x08

Reset value: 0x0000



Bit Field	Name	Description
15	EXTP	<p>External trigger polarity</p> <p>This bit is used to select whether the trigger operation is to use ETR or the inversion of ETR.</p> <p>0: ETR active at high level or rising edge. 1: ETR active at low level or falling edge.</p>
14	EXCEN	<p>External clock enable</p> <p>This bit is used to enable external clock mode 2, and the counter is driven by any active edge on the ETRF signal in this mode.</p> <p>0: External clock mode 2 disable. 1: External clock mode 2 enable.</p> <p><i>Note 1: When external clock mode 1 and external clock mode 2 are enabled at the same time, the input of the external clock is ETRF.</i></p> <p><i>Note 2: The following slave modes can be used simultaneously with external clock mode 2: reset mode, gated mode and trigger mode; However, TRGI cannot connect to ETRF (TIMx_SMCTRL.TSEL ≠ '111').</i></p> <p><i>Note 3: Setting the TIMx_SMCTRL.EXCEN bit has the same effect as selecting external clock mode 1 and connecting TRGI to ETRF (TIMx_SMCTRL.SMSEL = 111 and TIMx_SMCTRL.TSEL = 111).</i></p>
13:12	EXTPS[1:0]	<p>External trigger prescaler</p> <p>The frequency of the external trigger signal ETRP must be at most 1/4 of TIMxCLK frequency. When a faster external clock is input, a prescaler can be used to reduce the frequency of ETRP.</p> <p>00: Prescaler disable 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8</p>
11:8	EXTF[3:0]	<p>External trigger filter</p> <p>These bits are used to define the frequency at which the ETRP signal is sampled and the bandwidth of the ETRP digital filtering. In effect, the digital filter is an event counter that generates a validate output after consecutive N events are recorded.</p> <p>0000: No filter, sampling at f_{DTS} 0001: f_{SAMPLING} = f_{CK_INT}, N = 2 0010: f_{SAMPLING} = f_{CK_INT}, N = 4 0011: f_{SAMPLING} = f_{CK_INT}, N = 8 0100: f_{SAMPLING} = f_{DTS}/2, N = 6 0101: f_{SAMPLING} = f_{DTS}/2, N = 8 0110: f_{SAMPLING} = f_{DTS}/4, N = 6 0111: f_{SAMPLING} = f_{DTS}/4, N = 8 1000: f_{SAMPLING} = f_{DTS}/8, N = 6 1001: f_{SAMPLING} = f_{DTS}/8, N = 8 1010: f_{SAMPLING} = f_{DTS}/16, N = 5</p>

Bit Field	Name	Description
		1011: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16, N = 6$ 1100: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16, N = 8$ 1101: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32, N = 5$ 1110: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32, N = 6$ 1111: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32, N = 8$
7	MSMD	Master/ Slave mode 0: No action 1: Events on the trigger input (TRGI) are delayed to allow a perfect synchronization between the current timer (via TRGO) and its slaves. This is useful when several timers are required to be synchronized to a single external event.
6:4	TSEL[2:0]	Trigger selection These 3 bits are used to select the trigger input of the synchronous counter. 000: Internal trigger 0 (ITR0) 100: TI1 edge detector (TI1F_ED) 001: Internal trigger 1 (ITR1) 101: Filtered timer input 1 (TI1FP1) 010: Internal trigger 2 (ITR2) 110: Filtered timer input 2 (TI2FP2) 011: Internal trigger 3 (ITR3) 111: External triggered Input (ETRF) For more details on ITRx, refer to Table 10-3 below. <i>Note: These bits must be changed only when not in use (e. g. TIMx_SMCTRL.SMSEL=000) to avoid false edge detection at the transition.</i>
3	Reserved	Reserved, the reset value must be maintained
2:0	SMSEL[2:0]	Slave mode selection When an external signal is selected, the active edge of the trigger signal (TRGI) is linked to the selected external input polarity (refer to input control register and control register description) 000: Disable slave mode. If TIMx_CTRL1.CNTEN = 1, the prescaler is driven directly by the internal clock. 001: Encoder mode 1. According to the level of TI2FP2, the counter up-counting or down-counting on the edge of TI1FP1. 010: Encoder mode 2. According to the level of TI1FP1, the counter up-counting or down-counting on the edge of TI2FP2. 011: Encoder mode 3. According to the input level of another signal, the counter up-counting or down-counting on the edges of TI2FP1 and TI2FP2. 100: Reset mode. On the rising edge of the selected trigger input (TRGI), the counter is reinitialized and the shadow register is updated. 101: Gated mode. When the trigger input (TRGI) is high, the clock of the counter is enabled. Once the trigger input becomes low, the counter stops counting, but is not reset. In this mode, the start and stop of the counter are controlled. 110: Trigger mode. When a rising edge occurs on the trigger input (TRGI), the counter is started but not reset. In this mode, only the start of the counter is controlled.

Bit Field	Name	Description
		111: External clock mode 1. The counter is clocked by the rising edge of the selected trigger input (TRGI). <i>Note: Do not use gated mode if TIIF_ED is selected as the trigger input (TIMx_SMCTRL.TSEL=100). This is because TIIF_ED outputs a pulse for each TIIF transition, whereas gated mode checks the level of the triggered input.</i>

Table 10-3 TIMx Internal Trigger Connection

Slave Timer	ITR0 (TSEL = 000)	ITR1 (TSEL = 001)	ITR2 (TSEL = 010)	ITR3 (TSEL = 011)
TIM3	TIM1	NA	NA	TIM4
TIM4	TIM1	NA	TIM3	TIM8

10.4.5 DMA/Interrupt Enable Registers (TIMx_DINTEN)

Offset address: 0x0C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TDEN	Reserved	CC4DEN	CC3DEN	CC2DEN	CC1DEN	UDEN	Reserved	TIEN	Reserved	CC4IEN	CC3IEN	CC2IEN	CC1IEN	UIEN
	rw		rw	rw	rw	rw	rw		rw		rw	rw	rw	rw	rw

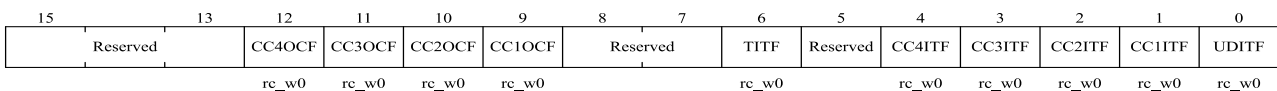
Bit Field	Name	Description
15	Reserved	Reserved, the reset value must be maintained
14	TDEN	Trigger DMA request enable 0: Disable trigger DMA request 1: Enable trigger DMA request
13	Reserved	Reserved, the reset value must be maintained
12	CC4DEN	Capture/Compare 4 DMA request enable 0: Disable capture/compare 4 DMA request 1: Enable capture/compare 4 DMA request
11	CC3DEN	Capture/Compare 3 DMA request enable 0: Disable capture/compare 3 DMA request 1: Enable capture/compare 3 DMA request
10	CC2DEN	Capture/Compare 2 DMA request enable 0: Disable capture/compare 2 DMA request 1: Enable capture/compare 2 DMA request
9	CC1DEN	Capture/Compare 1 DMA request enable 0: Disable capture/compare 1 DMA request 1: Enable capture/compare 1 DMA request
8	UDEN	Update DMA request enable 0: Disable update DMA request 1: Enable update DMA request

Bit Field	Name	Description
7	Reserved	Reserved, the reset value must be maintained
6	TIEN	Trigger interrupt enable 0: Disable trigger interrupt 1: Enable trigger interrupt
5	Reserved	Reserved, the reset value must be maintained
4	CC4IEN	Capture/Compare 4 interrupt enable 0: Disable capture/compare 4 interrupt 1: Enable capture/compare 4 interrupt
3	CC3IEN	Capture/Compare 3 interrupt enable 0: Disable capture/compare 3 interrupt 1: Enable capture/compare 3 interrupts
2	CC2IEN	Capture/Compare 2 interrupt enable 0: Disable capture/compare 2 interrupt 1: Enables capture/compare 2 interrupts
1	CC1IEN	Capture/Compare 1 interrupt enable 0: Disable capture/compare 1 interrupt 1: Enables capture/comparing 1 interrupt
0	UIEN	Update interrupt enable 0: Disable update interrupt 1: Enables update interrupt

10.4.6 Status Registers (TIMx_STS)

Offset address: 0x10

Reset value: 0x0000



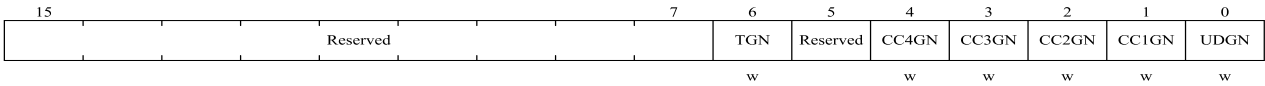
Bit Field	Name	Description
15:13	Reserved	Reserved, the reset value must be maintained
12	CC4OCF	Capture/Compare 4 overcapture flag See TIMx_STS.CC1OCF description.
11	CC3OCF	Capture/Compare 3 overcapture flag See TIMx_STS.CC1OCF description.
10	CC2OCF	Capture/Compare 2 overcapture flags See TIMx_STS.CC1OCF description.
9	CC1OCF	Capture/Compare 1 overcapture flag This bit is set by hardware only when the corresponding channel is configured in input capture mode. Cleared by software writing 0. 0: No overcapture occurred 1: TIMx_STS.CC1ITF was already set when the value of the counter has been captured in the TIMx_CCDAT1 register.

Bit Field	Name	Description
8:7	Reserved	Reserved, the reset value must be maintained
6	TITF	<p>Trigger interrupt flag</p> <p>This bit is set by hardware when an active edge is detected on the TRGI input when the slave mode controller is in a mode other than gated. This bit is set by hardware when any edge in gated mode is detected. This bit is cleared by software.</p> <p>0: No trigger event occurred</p> <p>1: Trigger interrupt occurred</p>
5	Reserved	Reserved, the reset value must be maintained
4	CC4ITF	<p>Capture/Compare 4 interrupt flag</p> <p>See TIMx_STS.CC1ITF description.</p>
3	CC3ITF	<p>Capture/Compare 3 interrupt flag</p> <p>See TIMx_STS.CC1ITF description.</p>
2	CC2ITF	<p>Capture/Compare 2 interrupt flag</p> <p>See TIMx_STS.CC1ITF description.</p>
1	CC1ITF	<p>Capture/Compare 1 interrupt flag</p> <p>When the corresponding channel of CC1 is in output mode:</p> <p>Except in center-aligned mode, this bit is set by hardware when the counter value is the same as the compare value (refer to TIMx_CTRL1.CAMSEL bit description). This bit is cleared by software.</p> <p>0: No match occurred.</p> <p>1: The value of TIMx_CNT is the same as the value of TIMx_CCDAT1.</p> <p>When the value of TIMx_CCDAT1 is greater than the value of TIMx_AR, the TIMx_STS.CC1ITF bit will go high if the counter overflows (in up-counting and up/down-counting modes) and underflows in down-counting mode.</p> <p>When the corresponding channel of CC1 is in input mode:</p> <p>This bit is set by hardware when the capture event occurs. This bit is cleared by software or by reading TIMx_CCDAT1.</p> <p>0: No input capture occurred.</p> <p>1: Input capture occurred. Counter value has captured in the TIMx_CCDAT1. An edge with the same polarity as selected has been detected on IC1.</p>
0	UDITF	<p>Update interrupt flag</p> <p>This bit is set by hardware when an update event occurs under the following conditions:</p> <ul style="list-style-type: none"> – When TIMx_CTRL1.UPDIS = 0, overflow or underflow (An update event is generated). – When TIMx_CTRL1.UPRS = 0, TIMx_CTRL1.UPDIS = 0, and set the TIMx_EVTGEN.UDGN bit by software to reinitialize the CNT. – When TIMx_CTRL1.UPRS = 0, TIMx_CTRL1.UPDIS = 0, and the counter CNT is reinitialized by the trigger event. (See TIMx_SMCTRL Register description) <p>This bit is cleared by software.</p> <p>0: No update event occurred</p> <p>1: Update interrupt occurred</p>

10.4.7 Event Generation Registers (TIMx_EVTGEN)

Offset address: 0x14

Reset values: 0x0000



Bit Field	Name	Description
15: 7	Reserved	Reserved, the reset value must be maintained.
6	TGN	<p>Trigger generation</p> <p>This bit can generate a trigger event when set by software. And at this time TIMx_STS.TITF = 1, if the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated. This bit is automatically cleared by hardware.</p> <p>0: No action</p> <p>1: Generated a trigger event</p>
5	Reserved	Reserved, the reset value must be maintained
4	CC4GN	<p>Capture/Compare 4 generation</p> <p>See TIMx_EVTGEN.CC1GN description.</p>
3	CC3GN	<p>Capture/Compare 3 generation</p> <p>See TIMx_EVTGEN.CC1GN description.</p>
2	CC2GN	<p>Capture/Compare 2 generation</p> <p>See TIMx_EVTGEN.CC1GN description.</p>
1	CC1GN	<p>Capture/Compare 1 generation</p> <p>This bit can generate a capture/compare event when set by software. This bit is automatically cleared by hardware.</p> <p>When the corresponding channel of CC1 is in output mode:</p> <p>The TIMx_STS.CC1ITF flag will be pulled high, if the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated.</p> <p>When the corresponding channel of CC1 is in input mode:</p> <p>TIMx_CCDAT1 will capture the current counter value, and the TIMx_STS.CC1ITF flag will be pulled high, if the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated. If The IMx_STS.CC1ITF is already pulled high, pull TIMx_STS.CC1OCF high.</p> <p>0: No action</p> <p>1: Generated a CC1 capture/compare event</p>
0	UDGN	<p>Update generation</p> <p>This bit can generate an update event when set by software. And at this time the counter will be reinitialized, the prescaler counter will be cleared, the counter will be cleared in center-aligned or up-counting mode, but take TIMx_AR in down-counting mode the value of the register. This bit is automatically cleared by hardware.</p> <p>0: No action</p> <p>1: Generated an update event</p>

10.4.8 Capture/Compare Mode Register 1 (TIMx_CCMOD1)

Offset address: 0x18

Reset value: 0x0000

Channels can be used for input (capture mode) or output (compare mode), and the direction of the channel is defined by the corresponding CCxSEL bit. The other bits of the register act differently in input and output modes. OCx describes the function of a channel in output mode, ICx describes the function of a channel in input mode. Hence, please note that the same bit can have different meanings in output mode and in input mode.

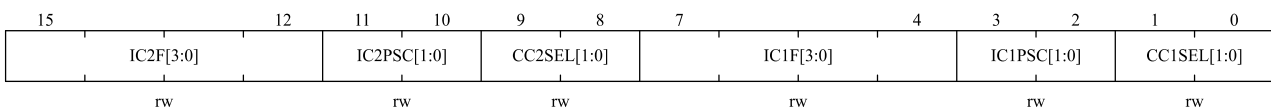
Output compare mode:

15	14	12	11	10	9	8	7	6	4	3	2	1	0
OC2CEN	OC2MD[2:0]		OC2PEN	OC2FEN	CC2SEL[1:0]		OC1CEN	OC1MD[2:0]		OC1PEN	OC1FEN	CC1SEL[1:0]	
rw	rw		rw	rw	rw		rw	rw		rw	rw	rw	

Bit Field	Name	Description
15	OC2CEN	Output Compare 2 clear enable
14:12	OC2MD[2:0]	Output Compare 2 mode
11	OC2PEN	Output Compare 2 preload enable
10	OC2FEN	Output Compare 2 fast enable
9:8	CC2SEL[1:0]	Capture/compare 2 selection These bits are used to select the input/output and input mapping of the channel 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 only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC2SEL is writable only when the channel is off (TIMx_CCEN.CC2EN = 0).</i>
7	OC1CEN	Output Compare 1 clear enable 0: OC1REF is not affected by ETRF input level 1: OC1REF is cleared immediately when the ETRF input level is detected as high
6:4	OC1MD[2:0]	Output Compare 1 mode These bits are used to manage the output reference signal OC1REF, which determines the values of OC1 and OC1N, and is valid at high levels, while the active levels of OC1 and OC1N depend on the TIMx_CCEN.CC1P and TIMx_CCEN.CC1NP bits. 000: Frozen. Comparison between TIMx_CCDAT1 register and counter TIMx_CNT has no effect on OC1REF signal. 001: Set channel 1 to the active level on match. When TIMx_CCDAT1 = TIMx_CNT, OC1REF signal will be forced high. 010: Set channel 1 as inactive level on match. When TIMx_CCDAT1 = TIMx_CNT, OC1REF signal will be forced low. 011: Toggle. When TIMx_CCDAT1 = TIMx_CNT, OC1REF signal will be toggled. 100: Force to inactive level. OC1REF signal is forced low. 101: Force to active level. OC1REF signal is forced high. 110: PWM mode 1 - In up-counting mode, if TIMx_CNT < TIMx_CCDAT1, OC1REF signal of channel 1 is high, otherwise it is low. In down-counting mode, if TIMx_CNT > TIMx_CCDAT1, OC1REF signal of channel 1 is low, otherwise it is high. 111: PWM mode 2 - In up-counting mode, if TIMx_CNT < TIMx_CCDAT1, OC1REF signal of channel 1 is low, otherwise it is high. In down-counting mode, if TIMx_CNT > TIMx_CCDAT1, OC1REF signal of channel 1 is high, otherwise it is low.

Bit Field	Name	Description
		<i>Note: In PWM mode 1 or PWM mode 2, the OC1REF level changes only when the comparison result changes or when the output compare mode is switched from frozen mode to PWM mode.</i>
3	OC1PEN	Output Compare 1 preload enable 0: Disable preload function of TIMx_CCDAT1 register. Supports write operations to TIMx_CCDAT1 register at any time, and the written value is effective immediately. 1: Enable preload function of TIMx_CCDAT1 register. Only read and write operations to preload registers. When an update event occurs, the value of TIMx_CCDAT1 is loaded into the active register. <i>Note: Only when TIMx_CTRL1.ONEPM = 1(In one-pulse mode), PWM mode can be used without verifying the preload register, otherwise no other behavior can be predicted.</i>
2	OC1FEN	Output Compare 1 fast enable This bit is used to speed up the response of the CC output to the trigger input event. 0: CC1 behaves normally depending on the counter and CCDAT1 values, even if the trigger is ON. The minimum delay for activating CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge of the trigger input acts like a comparison match on CC1 output. Therefore, OC is set to the comparison level regardless of the comparison result. The delay time for sampling the trigger input and activating the CC1 output is reduced to 3 clock cycles. OCxFEN only works if the channel is configured in PWM1 or PWM2 mode.
1: 0	CC1SEL[1:0]	Capture/compare 1 selection These bits are used to select the input/output and input mapping of the channel 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 channels are configured as inputs and IC1 is mapped to TRC. This mode is only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC1SEL is writable only when the channel is off (TIMx_CCEN.CCIEN = 0).</i>

Input capture mode:



Bit Field	Name	Description
15:12	IC2F[3:0]	Input Capture 2 Filter
11:10	IC2PSC[1:0]	Input Capture 2 Prescaler

Bit Field	Name	Description
9:8	CC2SEL[1:0]	<p>Capture/Compare 2 selection</p> <p>These bits are used to select the input/output and input mapping of the channel</p> <p>00: CC2 channel is configured as output</p> <p>01: CC2 channel is configured as input, IC2 is mapped on TI2</p> <p>10: CC2 channel is configured as input, IC2 is mapped on TI1</p> <p>11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL.</p> <p><i>Note: CC2SEL is writable only when the channel is off (TIMx_CCEN.CC2EN = 0).</i></p>
7:4	IC1F[3:0]	<p>Input Capture 1 filter</p> <p>These bits are used to define sampling frequency of TI1 input and the length of digital filter. The digital filter is an event counter that generates an output transition after N events are recorded.</p> <p>0000: No filter, sampling at f_{DTS} frequency</p> <p>0001: f_{SAMPLING} = f_{CK_INT}, N = 2</p> <p>0010: f_{SAMPLING} = f_{CK_INT}, N = 4</p> <p>0011: f_{SAMPLING} = f_{CK_INT}, N = 8</p> <p>0100: f_{SAMPLING} = f_{DTS}/2, N = 6</p> <p>0101: f_{SAMPLING} = f_{DTS}/2, N = 8</p> <p>0110: f_{SAMPLING} = f_{DTS}/4, N = 6</p> <p>0111: f_{SAMPLING} = f_{DTS}/4, N = 8</p> <p>1000: f_{SAMPLING} = f_{DTS}/8, N = 6</p> <p>1001: f_{SAMPLING} = f_{DTS}/8, N = 8</p> <p>1010: f_{SAMPLING} = f_{DTS}/16, N = 5</p> <p>1011: f_{SAMPLING} = f_{DTS}/16, N = 6</p> <p>1100: f_{SAMPLING} = f_{DTS}/16, N = 8</p> <p>1101: f_{SAMPLING} = f_{DTS}/32, N = 5</p> <p>1110: f_{SAMPLING} = f_{DTS}/32, N = 6</p> <p>1111: f_{SAMPLING} = f_{DTS}/32, N = 8</p>
3:2	IC1PSC[1:0]	<p>Input Capture 1 prescaler</p> <p>These bits are used to select the ratio of the prescaler for IC1 (CC1 input).</p> <p>When TIMx_CCEN.CC1EN = 0, the prescaler will be reset.</p> <p>00: No prescaler, capture is done each time an edge is detected on the capture input</p> <p>01: Capture is done once every 2 events</p> <p>10: Capture is done once every 4 events</p> <p>11: Capture is done once every 8 events</p>
1:0	CC1SEL[1:0]	<p>Capture/Compare 1 selection</p> <p>These bits are used to select the input/output and input mapping of the channel</p> <p>00: CC1 channel is configured as output</p> <p>01: CC1 channel is configured as input, IC1 is mapped on TI1</p> <p>10: CC1 channel is configured as input, IC1 is mapped on TI2</p> <p>11: CC1 channel is configured as input, IC1 is mapped to TRC. This mode is only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL.</p> <p><i>Note: CC1SEL is writable only when the channel is off (TIMx_CCEN.CC1EN = 0).</i></p>

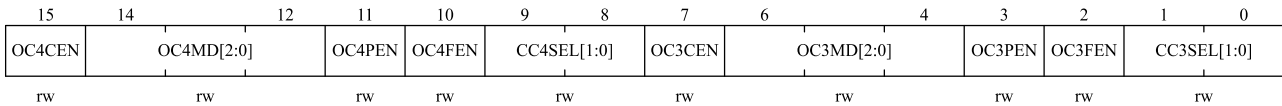
10.4.9 Capture/Compare Mode Register 2 (TIMx_CCMOD2)

Offset address: 0x1C

Reset value: 0x0000

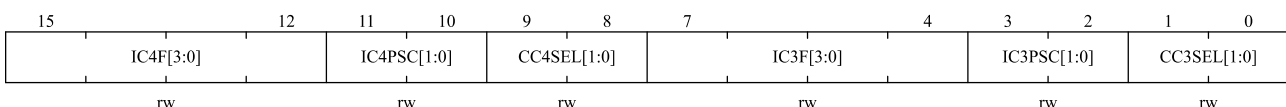
See the description of the CCMOD1 register above

Output comparison mode:



Bit Field	Name	Description
15	OC4CEN	Output compare 4 clear enable
14:12	OC4MD[2:0]	Output compare 4 mode
11	OC4PEN	Output compare 4 preload enable
10	OC4FEN	Output compare 4 fast enable
9:8	CC4SEL[1:0]	Capture/Compare 4 selection These bits are used to select the input/output and input mapping of the channel 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 only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC4SEL is writable only when the channel is off (TIMx_CCEN.CC4EN = 0).</i>
7	OC3CEN	Output compare 3 clear enable
6:4	OC3MD[2:0]	Output compare 3 mode
3	OC3PEN	Output compare 3 preload enable
2	OC3FEN	Output compare 3 fast enable
1:0	CC3SEL[1:0]	Capture/Compare 3 selection These bits are used to select the input/output and input mapping of the channel 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped to TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped to TRC. This mode is only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC3SEL is writable only when the channel is off (TIMx_CCEN.CC3EN = 0).</i>

Input capture mode:



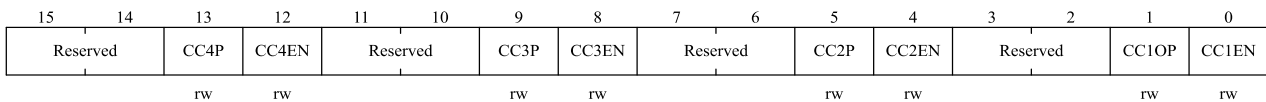
Bit Field	Name	Description
15:12	IC4F[3:0]	Input Capture 4 filter

Bit Field	Name	Description
11:10	IC4PSC[1:0]	Input Capture 4 Prescaler
9:8	CC4SEL[1:0]	Capture/Compare 4 selection These bits are used to select the input/output and input mapping of the channel 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 only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC4SEL is writable only when the channel is off (TIMx_CCEN.CC4EN = 0).</i>
7:4	IC3F[3:0]	Input Capture 3 filter
3:2	IC3PSC[1:0]	Input Capture 3 Prescaler
1:0	CC3SEL[1:0]	Capture/compare 3 selection These bits are used to select the input/output and input mapping of the channel 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped to TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped to TRC. This mode is only active when the internal trigger input is selected by TIMx_SMCTRL.TSEL. <i>Note: CC3SEL is writable only when the channel is off (TIMx_CCEN.CC3EN = 0).</i>

10.4.10 Capture/Compare Enable Registers (TIMx_CCEN)

Offset address: 0x20

Reset value: 0x0000



Bit Field	Name	Description
15:14	Reserved	Reserved, the reset value must be maintained.
13	CC4P	Capture/Compare 4 output polarity See TIMx_CCEN.CC1P description.
12	CC4EN	Capture/Compare 4 output enable See TIMx_CCEN.CC1EN description.
11:10	Reserved	Reserved, the reset value must be maintained
9	CC3P	Capture/Compare 3 output polarity See TIMx_CCEN.CC1P description.
8	CC3EN	Capture/Compare 3 output enable See TIMx_CCEN.CC1EN description.
7:6	Reserved	Reserved, the reset value must be maintained
5	CC2P	Capture/Compare 2 output polarity See TIMx_CCEN.CC1P description.
4	CC2EN	Capture/Compare 2 output enable

Bit Field	Name	Description
		See TIMx_CCEN.CC1EN description.
3:2	Reserved	Reserved, the reset value must be maintained
1	CC1P	<p>Capture/Compare 1 output polarity</p> <p>When the corresponding channel of CC1 is in output mode:</p> <p>0: OC1 active high</p> <p>1: OC1 active low</p> <p>When the corresponding channel of CC1 is in input mode:</p> <p>At this time, this bit is used to select whether IC1 or the inverse signal of IC1 is used as the trigger or capture signal.</p> <p>0: non-inverted: Capture action occurs when IC1 generates a rising edge. When used as external trigger, IC1 is non-inverted.</p> <p>1: inverted: Capture action occurs when IC1 generates a falling edge. When used as external trigger, IC1 is inverted.</p> <p><i>Note: If TIMx_BKDT.LCKCFG = 3 or 2, these bits cannot be modified.</i></p>
0	CC1EN	<p>Capture/Compare 1 output enable</p> <p>When the corresponding channel of CC1 is in output mode:</p> <p>0: Disable - Disable output OC1 signal.</p> <p>1: Enable - Enable output OC1 signal.</p> <p>When the corresponding channel of CC1 is in input mode:</p> <p>At this time, this bit is used to disable/enable the capture function.</p> <p>0: Disable capture</p> <p>1: Enable capture</p>

Table 10-4 Output Control Bits of Standard Ocx Channel

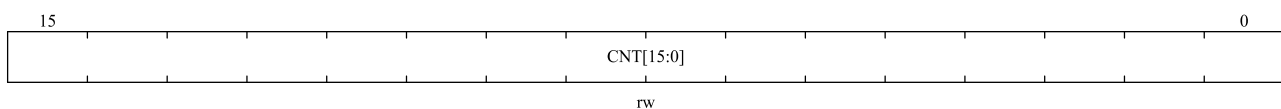
CCxEN	OCx Output Status
0	Disable output (OCx=0)
1	OCx = OCxREF + polarity

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

10.4.11 Counters (TIMx_CNT)

Offset address: 0x24

Reset value: 0x0000

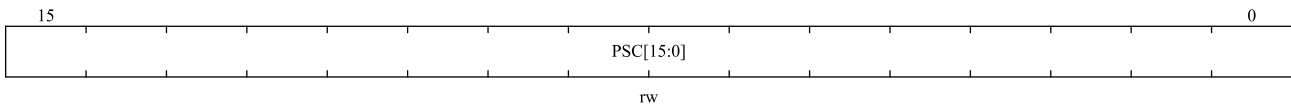


Bit Field	Name	Description
15:0	CNT[15:0]	Counter value

10.4.12 Prescaler (TIMx_PSC)

Offset address: 0x28

Reset value: 0x0000

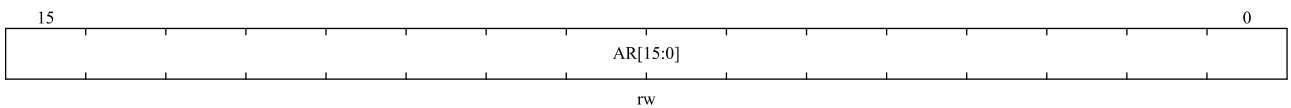


Bit Field	Name	Description
15:0	PSC[15:0]	Prescaler value $\text{Counter clock } f_{CK_CNT} = f_{CK_PSC} / (\text{PSC} [15:0] + 1)$. Each time an update event occurs, the PSC value is loaded into the active prescaler register.

10.4.13 Auto-reload Register (TIMx_AR)

Offset address: 0x2C

Reset values: 0xFFFF



Bit Field	Name	Description
15:0	AR[15:0]	Auto-reload value These bits define the value that will be loaded into the actual auto-reload register. See Section 9.3.1 for more details. When the TIMx_AR.AR [15:0] value is null, the counter does not work.

10.4.14 Capture/Compare Register 1 (TIMx_CCDAT1)

Offset address: 0x34

Reset value: 0x0000



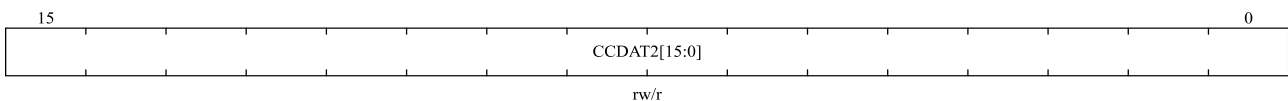
Bit Field	Name	Description
15:0	CCDAT1[15:0]	Capture/Compare 1 value <ul style="list-style-type: none"> CC1 channel is configured as output: CCDAT1 contains the value to be compared to the counter TIMx_CNT, signaling on the OC1 output. If the preload feature is not selected in TIMx_CCMOD1.OC1PEN bit, the written value is immediately transferred to the active register. Otherwise, this preloaded value is transferred to the active register only when an update event occurs.

Bit Field	Name	Description
		<ul style="list-style-type: none"> CC1 channel is configured as input: CCDAT1 contains the counter value transferred by the last input capture 1 event (IC1). When configured as input mode, register CCDAT1 is only readable. When configured as output mode, register CCDAT1 is readable and writable.

10.4.15 Capture/Compare Register 2 (TIMx_CCDAT2)

Offset address: 0x38

Reset value: 0x0000

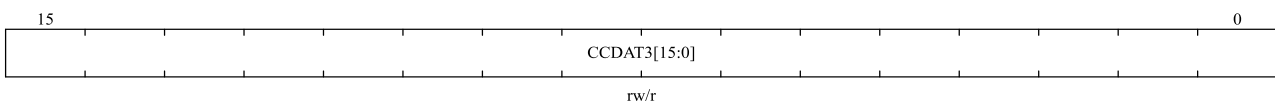


Bit Field	Name	Description
15:0	CCDAT2[15:0]	Capture/Compare 2 values <ul style="list-style-type: none"> CC2 channel is configured as output: CCDAT2 contains the value to be compared to the counter TIMx_CNT, signaling on the OC2 output. If the preload feature is not selected in TIMx_CCMOD1.OC2PEN bit, the written value is immediately transferred to the active register. Otherwise, this preloaded value is transferred to the active register only when an update event occurs. <ul style="list-style-type: none"> CC2 channel is configured as input: CCDAT2 contains the counter value transferred by the last input capture 2 event (IC2). When configured as input mode, register CCDAT2 is only readable. When configured as output mode, register CCDAT2 is readable and writable.

10.4.16 Capture/Compare Register 3 (TIMx_CCDAT3)

Offset address: 0x3C

Reset value: 0x0000



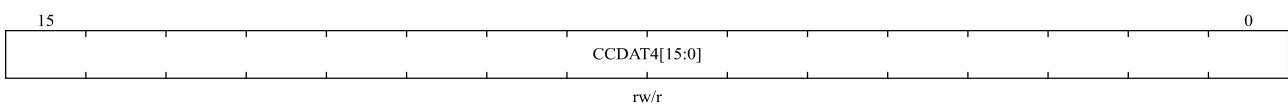
Bit Field	Name	Description
15:0	CCDAT3[15:0]	Capture/Compare 3 value <ul style="list-style-type: none"> CC3 channel is configured as output: CCDAT3 contains the value to be compared to the counter TIMx_CNT, signaling on the OC3 output. If the preload feature is not selected in TIMx_CCMOD2.OC3PEN bit, the written value is immediately transferred to the active register. Otherwise, this preloaded value is transferred to the active register only when an update event occurs.

Bit Field	Name	Description
		<ul style="list-style-type: none"> CC3 channel is configured as input: CCDAT3 contains the counter value transferred by the last input capture 3 event (IC3). When configured as input mode, register CCDAT3 is only readable. When configured as output mode, register CCDAT3 is readable and writable.

10.4.17 Capture/Compare Register 4 (TIMx_CCDA4)

Offset address: 0x40

Reset value: 0x0000

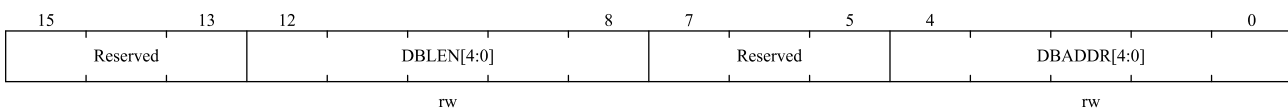


Bit Field	Name	Description
15:0	CCDAT4[15:0]	Capture/Compare 4 value <ul style="list-style-type: none"> CC4 channel is configured as output: CCDAT4 contains the value to be compared to the counter TIMx_CNT, signaling on the OC4 output. If the preload feature is not selected in TIMx_CCMOD2.OC4PEN bit, the written value is immediately transferred to the active register. Otherwise, this preloaded value is transferred to the active register only when an update event occurs. <ul style="list-style-type: none"> CC4 channel is configured as input: CCDAT4 contains the counter value transferred by the last input capture 4 event (IC4). When configured as input mode, register CCDAT4 is only readable. When configured as output mode, register CCDAT4 is readable and writable.

10.4.18 DMA Control Register (TIMx_DCTRL)

Offset address: 0x48

Reset value: 0x0000



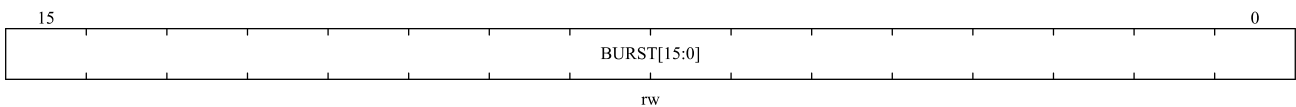
Bit Field	Name	Description
15:13	Reserved	Reserved, the reset value must be maintained
12:8	DBLEN[4:0]	DMA Burst Length This bit field defines the number DMA will accesses (write/read) TIMx_DADDR register. 00000: 1 time transfer 00001: 2 times transfers 00010: 3 times transfers

Bit Field	Name	Description
		... 10001: 18 times transfers
7:5	Reserved	Reserved, the reset value must be maintained.
4:0	DBADDR[4:0]	<p>DMA Base Address</p> <p>This bit field defines the first address where the DMA accesses the TIMx_DADDR register. When access is done through the TIMx_DADDR first time, this bit-field specifies the address you just access. And then the second access to the TIMx_DADDR, you will access the address of “DMA Base Address + 4”</p> <p>00000: TIMx_CTRL1, 00001: TIMx_CTRL2, 00010: TIMx_SMCTRL, ... 01011: TIMx_AR, 01100: Reserved, 01101: TIMx_CC DAT1, 10000: TIMx_CC DAT4, 10001: Reserved, 10010: TIMx_DCTRL</p>

10.4.19 DMA Transfer Buffer Register (TIMx_DADDR)

Offset address: 0x4C

Reset value: 0x0000



Bit Field	Name	Description
15:0	BURST[15:0]	<p>DMA access buffer.</p> <p>When a read or write operation is assigned to this register, the register located at the address range (DMA base address + DMA burst length × 4) will be accessed.</p> <p>DMA base address = The address of TIMx_CTRL1 + TIMx_DCTRL.DBADDR * 4; DMA burst len = TIMx_DCTRL.DBLEN + 1.</p> <p>Example: If TIMx_DCTRL.DBLEN = 0x3(4 transfers), TIMx_DCTRL.DBADDR = 0xD (TIMx_CC DAT1), DMA data length = half word, DMA memory address = buffer address in SRAM, DMA peripheral address = TIMx_DADDR address.</p> <p>When an event occurs, TIMx will send requests to the DMA, and transfer data 4 times. For the first time, DMA access to the TIMx_DADDR register will be mapped to access TIMx_CC DAT1 register; For the second time, DMA access to the TIMx_DADDR register will be mapped to access</p>

Bit Field	Name	Description
		TIMx_CC DAT2 register; For the fourth time, DMA access to the TIMx_DADDR register will be mapped to access TIMx_CC DAT4 register;

11 Basic Timers (TIM6)

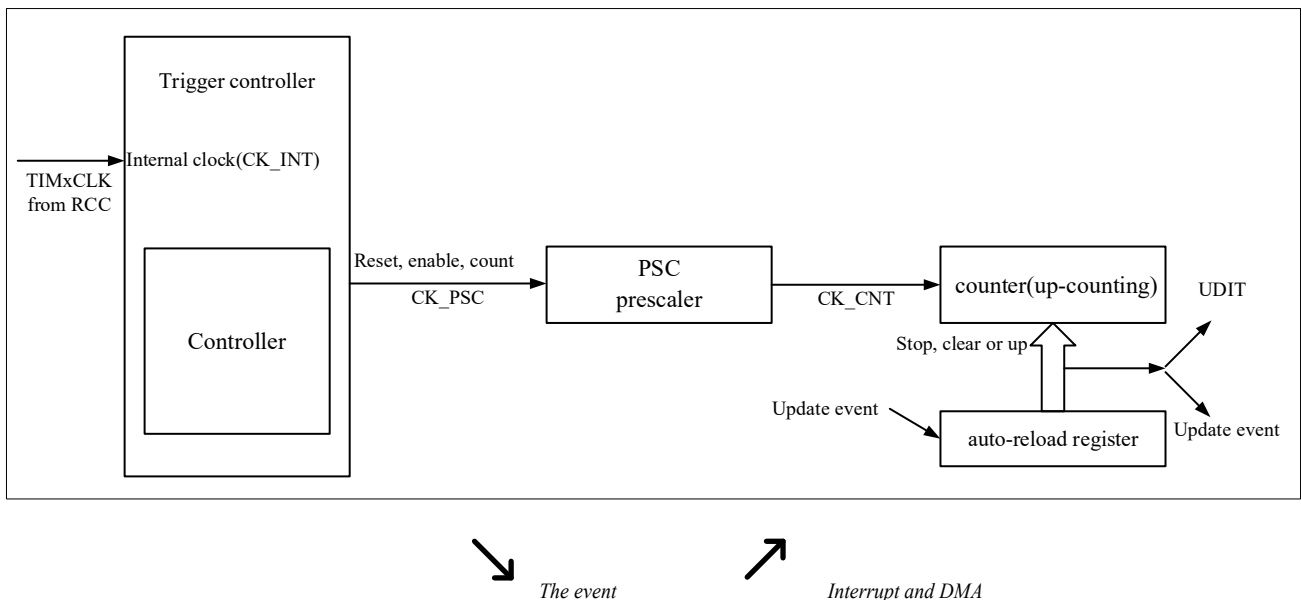
11.1 Basic Timers Introduction

The basic timer contains a 16-bit counter.

11.2 Main Features of Basic Timers

- 16-bit auto-reload up-counting counter.
- 16-bit programmable prescaler. (The prescaler factor can be configured with any value between 1 and 65536)
- Generates interrupt/DMA request on update event (counter overflow)

Figure 11-1 Block Diagram of TIMx (x = 6)



11.3 Basic Timers Description

11.3.1 Time-base Unit

The time-base unit mainly includes: prescaler, counter and auto-reload register. When the time base unit is operating, the software can read and write the corresponding registers (TIMx_PSC, TIMx_CNT and TIMx_AR) at any time.

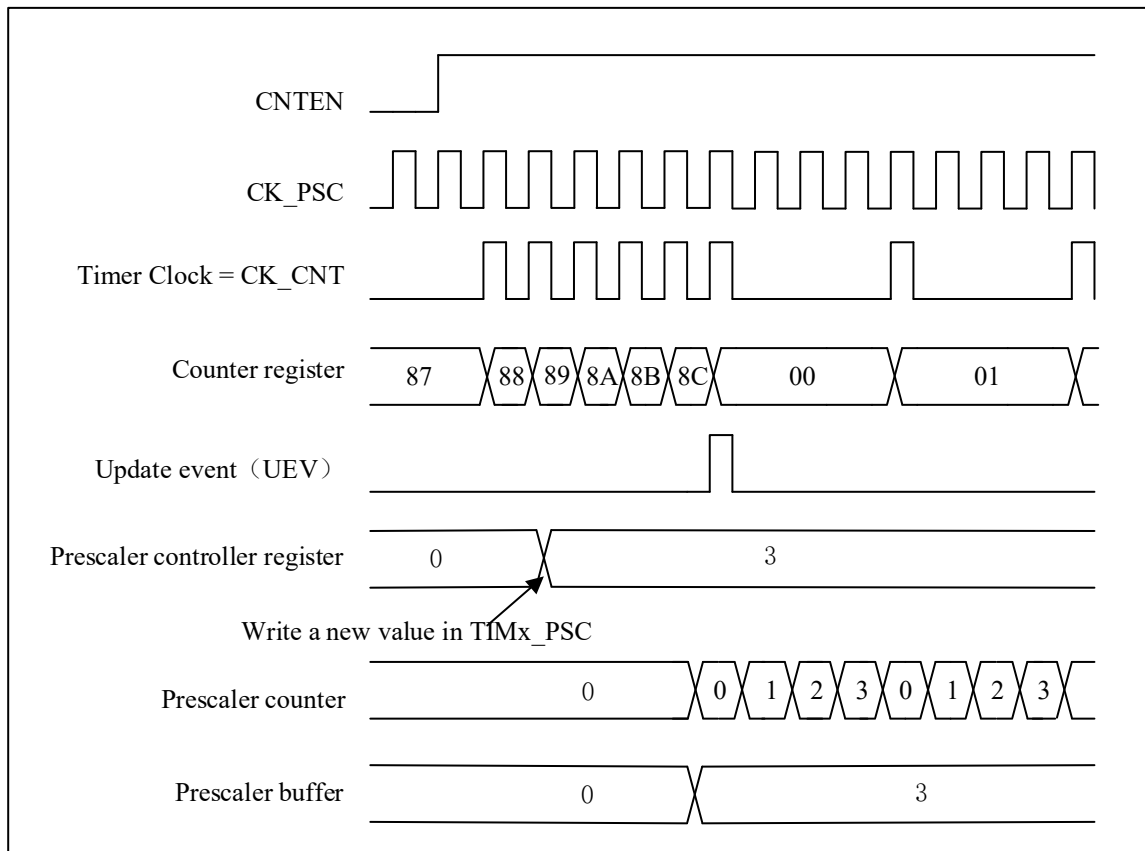
Depending on the setting of the auto-reload preload enable bit (TIMx_CTRL1.ARPEN), the value of the preload register is transferred to the shadow register immediately or at each update event UEV. when TIMx_CTRL1.UPDIS=0, an update event is generated when the counter reaches the overflow condition and it can be generated when the TIMx_EVTGEN.UDGN bit is set by software. The counter CK_CNT is valid only when the TIMx_CTRL1.CNTEN bit is set. The counter starts counting one clock cycle after the TIMx_CTRL1.CNTEN bit is set.

11.3.1.1 Prescaler description

The TIMx_PSC register consists of a 16-bit counter that can be used to divide the counter clock frequency by any factor between 1 and 65536. It can be changed on the fly as it is buffered. The new prescaler ratio is only taken into account at

the next update event.

Figure 11-2 Counter Timing Diagram with Prescaler Division Change from 1 to 4



11.3.2 Counter Mode

11.3.2.1 Up-counting mode

In up-counting mode, the counter will count from 0 to the value of the register TIMx_AR, then restart from 0, and a counter overflow event is generated.

If the TIMx_CTRL1.UPRS bit (select update request) and the TIMx_EVTGEN.UDGN bit are set, an update event (UEV) will generate, but TIMx_STS.UDITF will not be set by hardware. Therefore, no update interrupts or DMA update requests are generated. This setting is used in scenarios where you want to clear the counter but do not want to generate an update interrupt.

When an update event occurs, all registers are updated and the TIMx_STS.UDITF is set (depending on TIMx_CTRL1.UPRS):

- Update auto-reload shadow registers with preload value (TIMx_AR), when TIMx_CTRL1.ARPEN = 1.
- The prescaler shadow register is reloaded with the preload value (TIMx_PSC)

Setting TIMx_CTRL1.UPDIS=1 to avoid updating the shadow registers when new values are written to the preload registers.

In this way, when an update event occurs, the counter and the prescaler counter will be reset to 0 (but the prescaler value will remain unchanged).

The figure below shows some examples of the counter behavior and the update flags for different clock frequencies in the up-counting mode.

Figure 11-3 Timing Diagram of Up-counting. The Internal Clock Divider Factor = 2/N

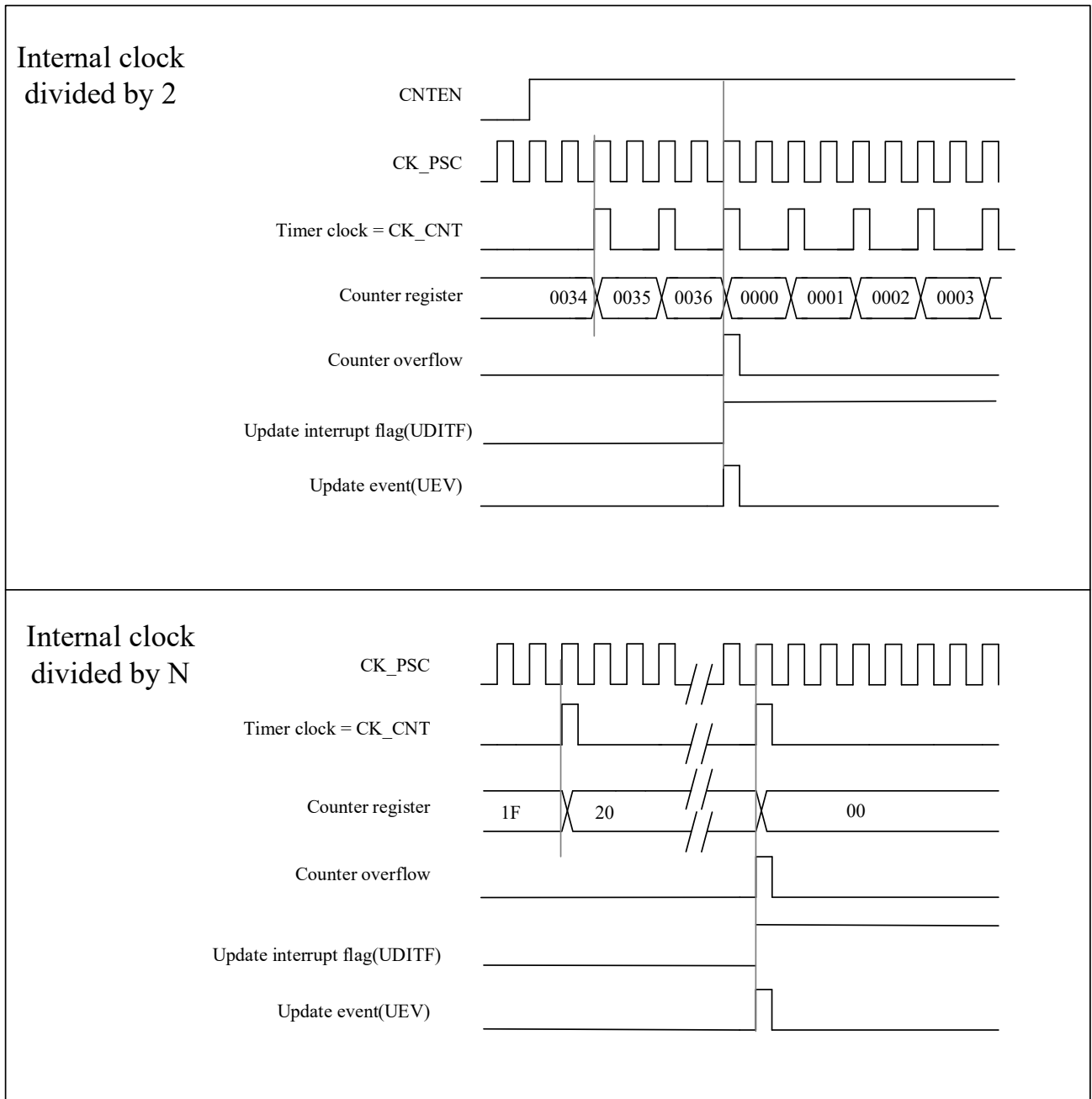
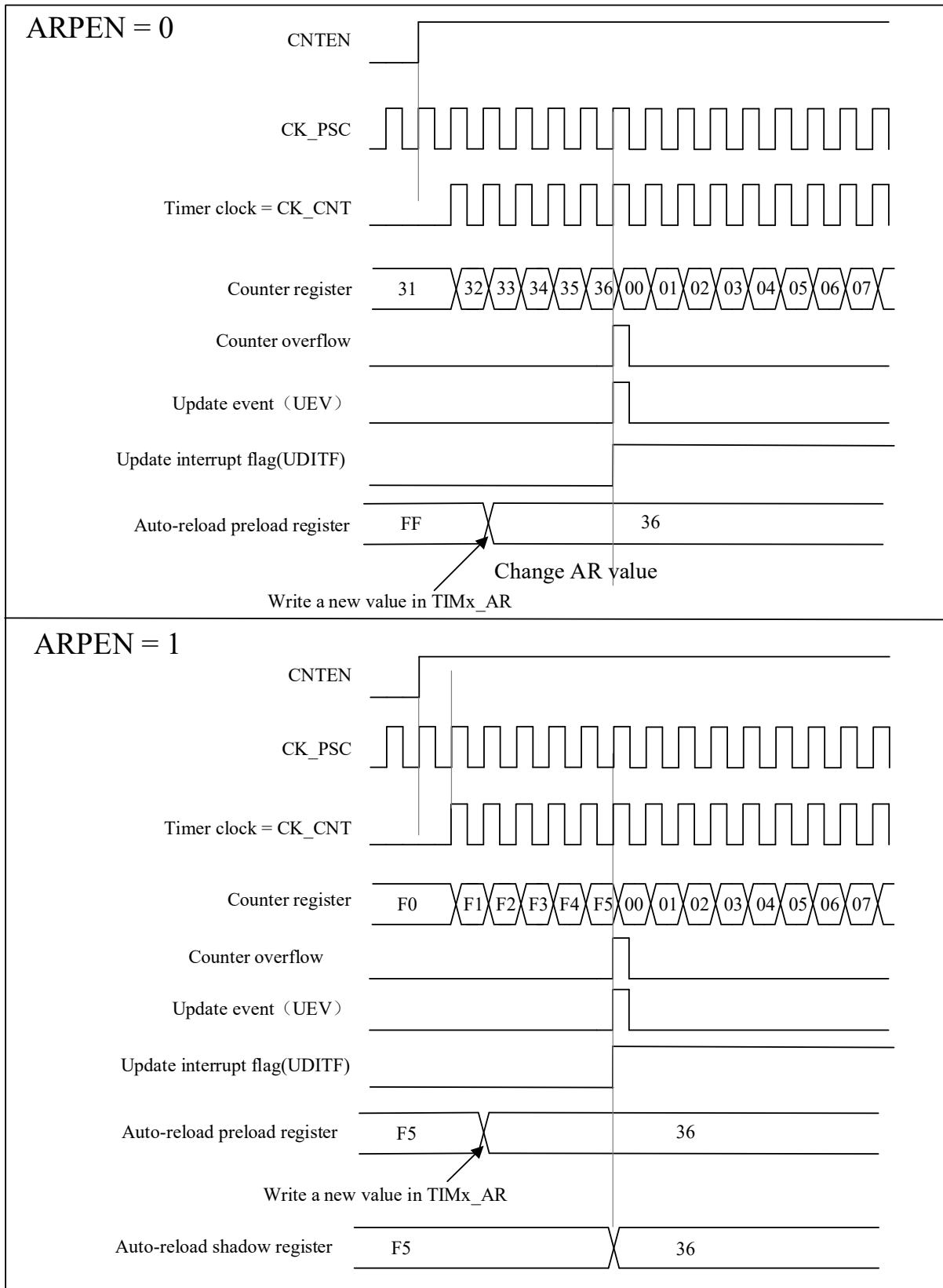


Figure 11-4 Timing Diagram of The Up-Counting, Update Event When ARPEN=0/1



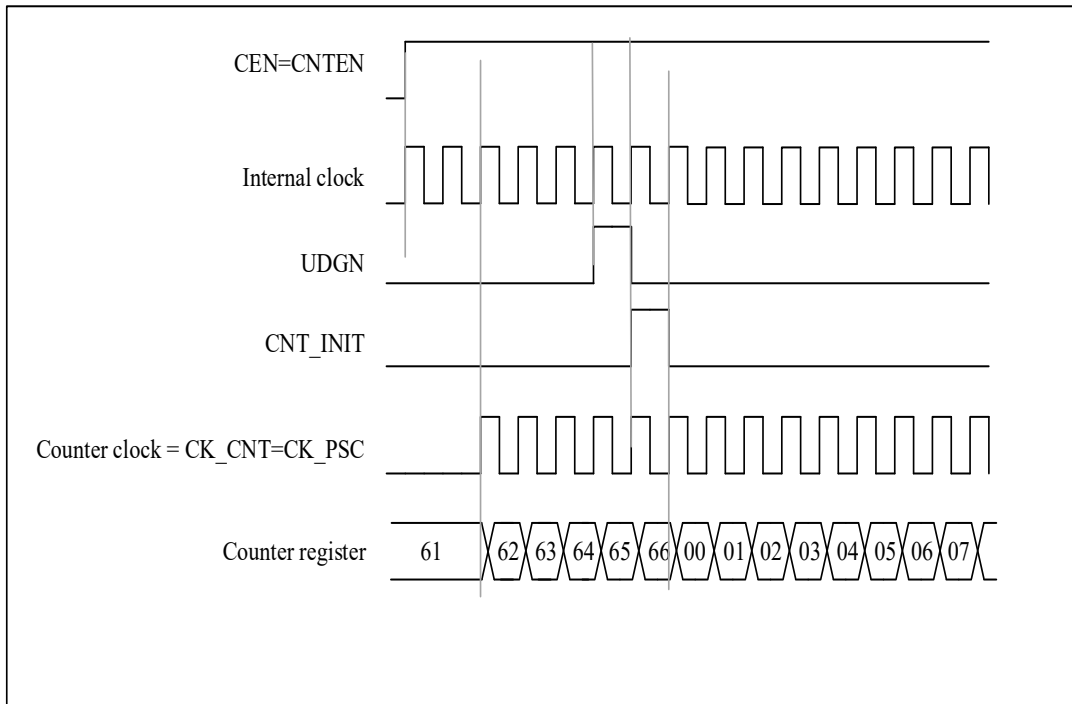
11.3.3 Clock Selection

- The internal clock of timers: CK_INT

11.3.3.1 Internal clock source (CK_INT)

It is provided that the TIMx_CTRL1.CNTEN bit is written as '1' by software, the clock source of the prescaler is provided by the internal clock CK_INT.

Figure 11-5 Control Circuit in Normal Mode, Internal Clock Divided by 1



11.3.4 Debug Mode

When the microcontroller is in debug mode (the Cortex[®]-M0 core halted), the TIMx counter can either continue to operate normally or stop. For more details depending on the DBG_CTRL.TIMx_STOP configuration in the PWR module, refer to Section 3.3.2.

11.4 Timx Register Description(x=6)

For abbreviations used in registers, refer to Section 1.1

All register operations must be performed in half word (16-bits) or one word (32-bits).

11.4.1 Register Overview

Table 11-1 Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
000h	TIMx_CTRL1	Reserved														ARPE	Reserved			ONEP	UPRS	UPDIS	CNTEEN										

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	Reset Value																								0					0	0	0	0	0
004h	Reserved																																	
008h	Reserved																																	
00Ch	TIMx_DINT EN	Reserved																				UDEN		Reserved				UIEN						
	Reset Value																					0						0						
010h	TIMx_STS	Reserved																										UDITF						
	Reset Value																											0						
014h	TIMx_EVTG EN	Reserved																										UDGN						
	Reset Value																											0						
018h	Reserved																																	
01Ch	Reserved																																	
020h	Reserved																																	
024h	TIMx_CNT	Reserved															CNT[15:0]																	
	Reset Value																0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																	
028h	TIMx_PSC	Reserved															PSC[15:0]																	
	Reset Value																0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																	
02Ch	TIMx_AR	Reserved															AR[15:0]																	
	Reset Value																1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1																	

11.4.2 Control Register 1 (TIMx_CTRL1)

Offset address: 0x00

Reset value: 0x0000

15	Reserved						8	7	6	4		3	2	1	0
							ARPEN	Reserved			ONEPM	UPRS	UPDIS	CNTEN	
							rw				rw	rw	rw	rw	

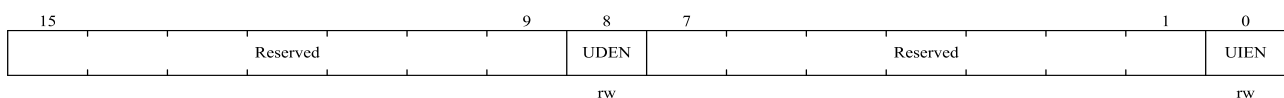
Bit Field	Name	Description
15:8	Reserved	Reserved, the reset value must be maintained

Bit Field	Name	Description
7	ARPEN	ARPEN: Auto-reload preload enable 0: Shadow register disable for TIMx_AR register 1: Shadow register enable for TIMx_AR register
6:4	Reserved	Reserved, the reset value must be maintained
3	ONEPM	One-pulse mode 0: Disable one-pulse mode, the counter counts are not affected when an update event occurs. 1: Enable one-pulse mode, the counter stops counting when the next update event occurs (clearing TIMx_CTRL1.CNTEN bit)
2	UPRS	Update request source This bit is used to select the UEV event sources by software. 0: If update interrupt or DMA request is enabled, any of the following events will generate an update interrupt or DMA request: Counter overflow The TIMx_EVTGEN.UDGN bit is set 1: If update interrupt or DMA request is enabled, only counter overflow will generate update interrupt or DMA request
1	UPDIS	Update disable This bit is used to enable/disable the Update event (UEV) events generation by software. 0: Enable UEV. UEV will be generated if one of following condition been fulfilled: Counter overflow The TIMx_EVTGEN.UDGN bit is set Shadow registers will update with preload value. 1: UEV disabled. No update event is generated, and the shadow registers (AR, PSC) keep their values. If the TIMx_EVTGEN.UDGN bit is set, the counter and prescaler are reinitialized.
0	CNTEN	Counter Enable 0: Disable counter 1: Enable counter

11.4.3 DMA/Interrupt Enable Registers (TIMx_DINTEN)

Offset address: 0x0C

Reset value: 0x0000



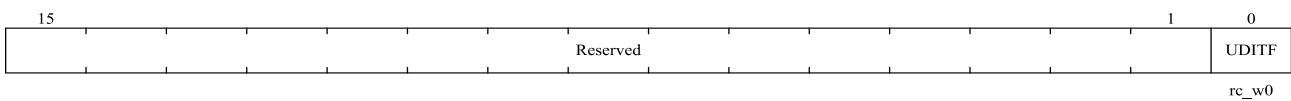
Bit Field	Name	Description
15:9	Reserved	Reserved, the reset value must be maintained
8	UDEN	Update DMA Request enable 0: Disable update DMA request 1: Enable update DMA request

Bit Field	Name	Description
7:1	Reserved	Reserved, the reset value must be maintained
0	UIEN	Update interrupt enable 0: Disable update interrupt 1: Enables update interrupt

11.4.4 Status Registers (TIMx_STS)

Offset address: 0x10

Reset value: 0x0000

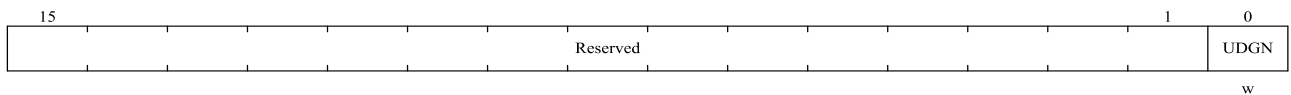


Bit Field	Name	Description
15:1	Reserved	Reserved, the reset value must be maintained
0	UDITF	Update interrupt flag This bit is set by hardware when an update event occurs under the following conditions: When TIMx_CTRL1.UPDIS = 0, and counter value overflow. When TIMx_CTRL1.UPRS = 0, TIMx_CTRL1.UPDIS = 0, and set the TIMx_EVTGEN.UDGN bit by software to reinitialize the CNT. This bit is cleared by software. 0: No update event occurred 1: Update interrupt occurred

11.4.5 Event Generation Registers (TIMx_EVTGEN)

Offset address: 0x14

Reset values: 0 x0000



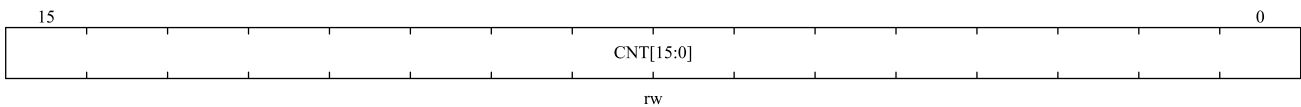
Bit Field	Name	Description
15: 1	Reserved	Reserved, the reset value must be maintained.
0	UDGN	UDGN: Update generation Software can set this bit to update configuration register value and hardware will clear it automatically. 0: No effect.

Bit Field	Name	Description
		1: Timer will restart and all shadow register will be updated. It will restart prescaler counter also.

11.4.6 Counters (TIMx_CNT)

Offset address: 0x24

Reset value: 0x0000

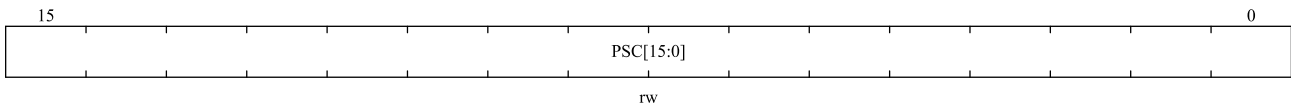


Bit Field	Name	Description
15:0	CNT[15:0]	Counter value

11.4.7 Prescaler (TIMx_PSC)

Offset address: 0x28

Reset value: 0x0000

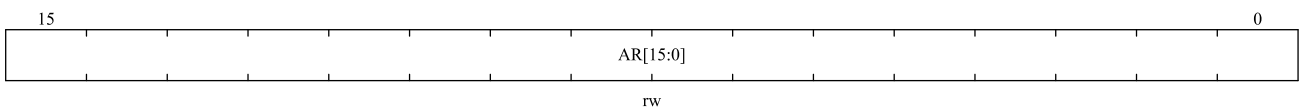


Bit Field	Name	Description
15:0	PSC[15:0]	Prescaler value PSC register value will be updated to prescaler register at update event. Counter clock frequency is input clock frequency divide PSC + 1.

11.4.8 Automatic Reload Register (TIMx_AR)

Offset address: 0x2C

Reset values: 0xFFFF



Bit Field	Name	Description
15:0	AR[15:0]	<p>Auto-reload value</p> <p>These bits define the value that will be loaded into the actual auto-reload register.</p> <p>See 11.3.1 for more details.</p> <p>When the TIMx_AR.AR [15:0] value is null, the counter does not work.</p>

12 Low Power Timer (LPTIM)

12.1 Introduction

The LPTIM is a 16-bit timer with multiple clock sources, it can keep running in all power modes except PD mode. LPTIM can run without internal clock source, it can be used as a “Pulse Counter”. Also, the LPTIM can wake up the system from low-power modes, to realize “Timeout functions” with extremely low power consumption.

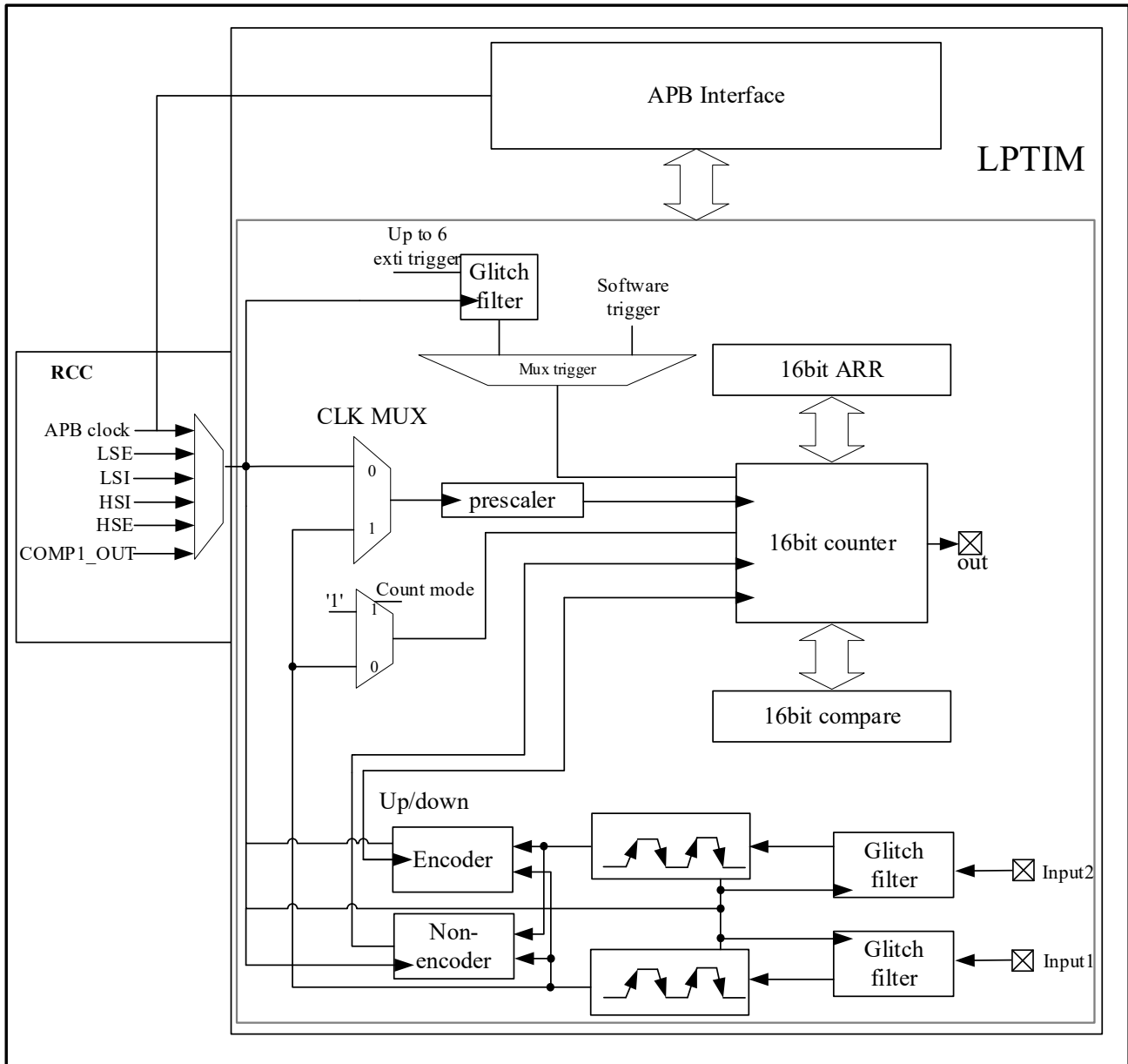
12.2 Main Features

- 16-bit up auto-reload counter
- 3-bit clock prescaler, 8 prescaler factors (1, 2, 4, 8, 16, 32, 64, 128)
- Multiple clock sources
 - Internal: HSI, HSE, LSI, LSE, APB1 and COMP1_OUT clock
 - External: External clock source through LPTIM Input1 (operating without LP oscillator, for pulse counter applications)
- 16-bit auto-reload register
- 16-bit compare register
- Continuous/One-shot trigger mode
- Programmable software and hardware input trigger
- Programmable digital glitch filter
- Configurable output: Single pulse, PWM
- Configurable I/O polarity
- Encoder mode

12.3 Function Description

12.3.1 Block Diagram

Figure 12-1 LPTIM Diagram



12.3.2 LPTIM Reset and Clock

The LPTIM can use an internal clock source or an external clock source. The internal clock source can be selected between APB, LSI, LSE, HSE, HSI or COMP1 by configuring the RCC_CFG2.LPTIMSEL[2:0] bits. The external clock source can be selected from comparator or GPIO. For external clock source, the LPTIM has two configurations:

- The LPTIM uses both external clock and internal clock.
- The LPTIM only use external clock from comparator or input1. This configuration is for LOW POWER application.

LPTIM_CFG.CLKSEL and LPTIM_CFG.CNTMEN bits are used for the clock source configuration. The active clock

edge is configured through LPTIM_CFG.CLKPOL[1:0] bits.

When the LPTIM only uses external clock source. It can only select one active clock edge. LPTIM can select both active clock edges only when it is using internal clock source or both external and internal clock sources.

Note: When using both active edges for external clock, LPTIM needs to use an internal clock to oversample the external clock. The internal clock frequency should be at least 4 times higher than the external clock frequency.

12.3.3 Glitch Filter

LPTIM has glitch filters for inputs to remove glitches and prevent unexpected counts or triggers.

Glitch filter needs an internal clock source to operate. And the clock source should be provided before the glitch filter is enabled. This is necessary to guarantee the proper operation of the filters.

The glitch filters are divided into two groups:

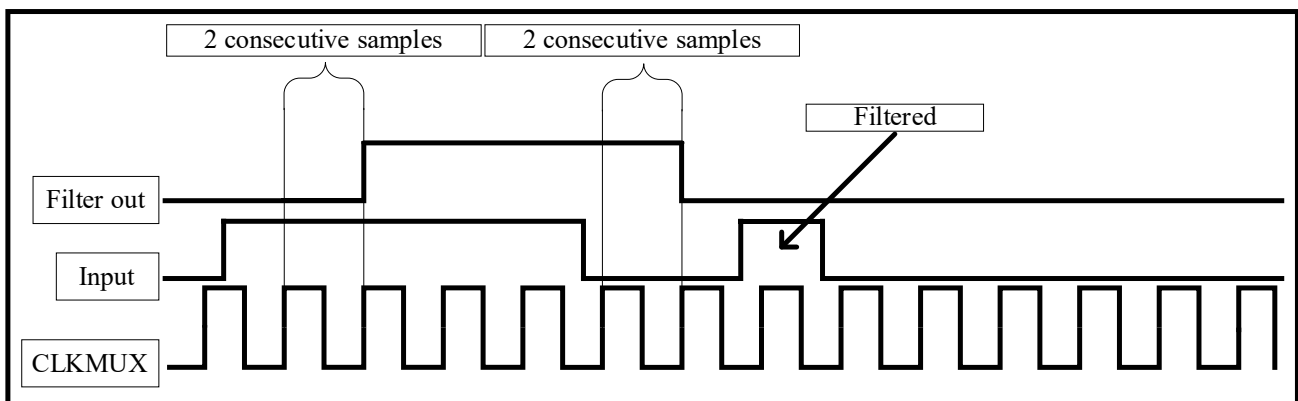
- For the external inputs: The filter sensitivity is configured through the LPTIM_CFG.CLKFLT[1:0] bits.
- For the internal trigger inputs: The filter sensitivity is configured through the LPTIM_CFG.RIGFLT[1:0] bits.

Note: The detection configuration is only applicable for its corresponding inputs.

The filter sensitivity acts on the number of consecutive equal samples that should be detected on one of the LPTIM inputs to consider a signal level change as a valid transition.

Shows an example of glitch filter behavior when detected two consecutive samples.

Figure 12-2 Glitch Filter Timing Diagram



Note: If no internal clock is used, the glitch filter needs to be turned off by clearing LPTIM_CFG.CLKFLT[1:0] and LPTIM_CFG.TRIGFLT[1:0] bits. If glitch filter is not used, user can use digital filter in comparator or external analog filter to remove glitches.

12.3.4 Prescaler

The LPTIM 16-bit counter is preceded by a configurable power-of-2 prescaler. The prescaler ratio is controlled by the LPTIM_CFG.CLKPRE[2:0] field. The table below lists all the possible division ratios:

Table 12-1 Pre-Scaler Division Ratios

Control Bits	The Corresponding Frequency Division Factor
--------------	---------------------------------------------

000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

12.3.5 Trigger Multiplexer

The LPTIM counter can be triggered either by software or by an effective edge on one of the 6 trigger inputs.

The trigger source is configured through LPTIM_CFG.TRGEN[1:0] bits. If LPTIM_CFG.TRGEN[1:0] = '00', the LPTIM start counting can be triggered by setting LPTIM_CTRL.TSTCM or LPTIM_CTRL.SNGMST bit. The other values of LPTIM_CFG.TRGEN[1:0] are for the effective edge configuration of the trigger. The internal counter will start once an effective edge is detected.

LPTIM_CFG.TRGSEL[2:0] is used to selected one of the 6 trigger inputs only when LPTIM_CFG.TRGEN[1:0] is not equal to '00'.

If LPTIM is using external trigger, which will be considered as asynchronous triggers. For asynchronous triggers, the LPTIM needs two counter clock cycles latency for synchronization.

When timeout function is disabled, new trigger event will be ignored if the LPTIM is already started.

Note: Any write to the LPTIM_CTRL.SNGMST/ LPTIM_CTRL.TSTCM bit will be discarded if the LPTIM is not enabled.

Table 12-2 6 Trigger Inputs Corresponding to LPTIM_CFG.TRGSEL[2:0] Bits

Trigger Selection Mode	Corresponding External Trigger Pin
lptim_ext_trig0	PB6,PC3,PA13,PA6 or PA4
lptim_ext_trig1	RTC alarm A
lptim_ext_trig2	RTC alarm B
lptim_ext_trig3	RTC_TAMP1
lptim_ext_trig4	RTC_TAMP2
lptim_ext_trig6	COMP1_OUT,COMP2_OUT or COMP3_OUT

12.3.6 Operating Mode

The LPTIM has two operating modes:

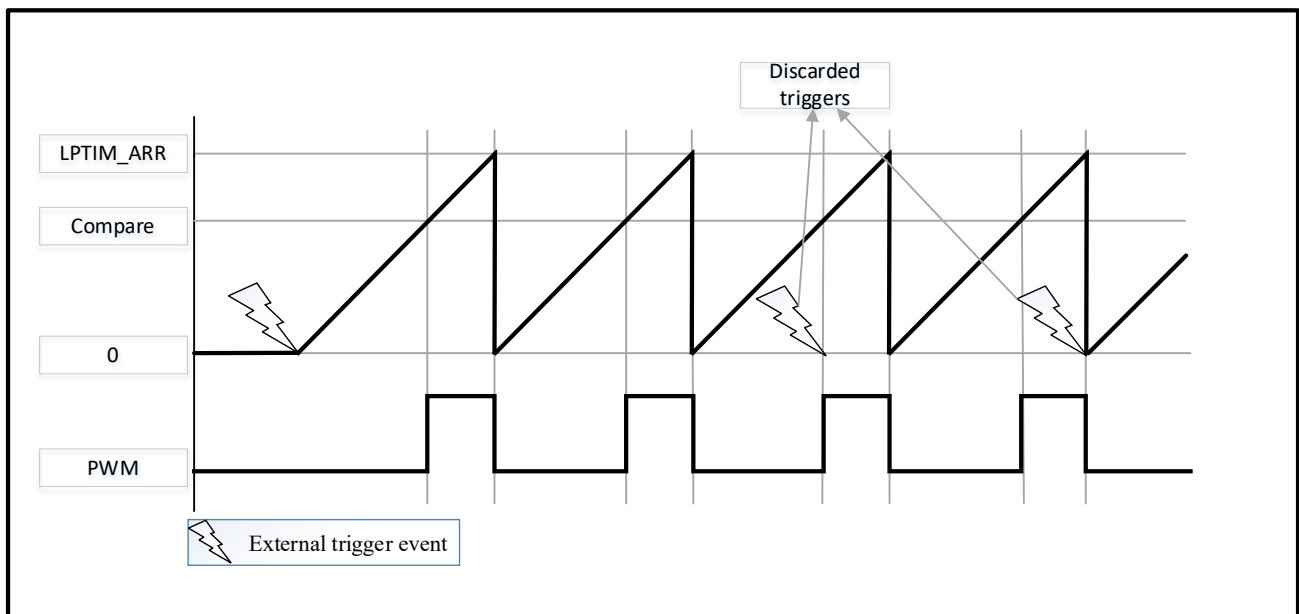
- Continuous mode: A trigger event will start the LPTIM and it will continue running until the user switched off the LPTIM.
- One-shot mode: A trigger event will start the LPTIM and it will stop when the counter value reached LPTIM_ARR.ARRVAL[15:0].

Continuous mode:

LPTIM_CTRL.TSTCM bit must be set to enable the continuous mode. If LPTIM uses external trigger, the internal counter will start when an external trigger event arrives after LPTIM_CTRL.TSTCM bit is set. After the continuous mode starts, hardware will discard any subsequent external trigger event.

If software trigger is used, setting LPTIM_CTRL.TSTCM bit will start the internal counter for continuous mode. Any subsequent external trigger event will be discarded as shown in Figure 12-3.

Figure 12-3 LPTIM Output Waveform, Continuous Counting Mode Configuration



LPTIM_CTRL.SNGMST and LPTIM_CTRL.TSTCM bits can only be set when the timer is enabled (The LPTIM_CTRL.LPTIMEN bit is set to '1').

It is possible to switch from one-shot mode to continuous mode. Setting LPTIM_CTRL.SNGMST bit will switch the LPTIM to one-shot counting mode if continuous counting mode was previously selected. The counter stops as soon as it reaches the LPTIM_ARR register value. If the one-shot counting mode was previously selected, setting LPTIM_CTRL.TSTCM bit to 1 will switch the LPTIM to continuous counting mode. Counter will restart as soon as LPTIM_ARR register value is reached.

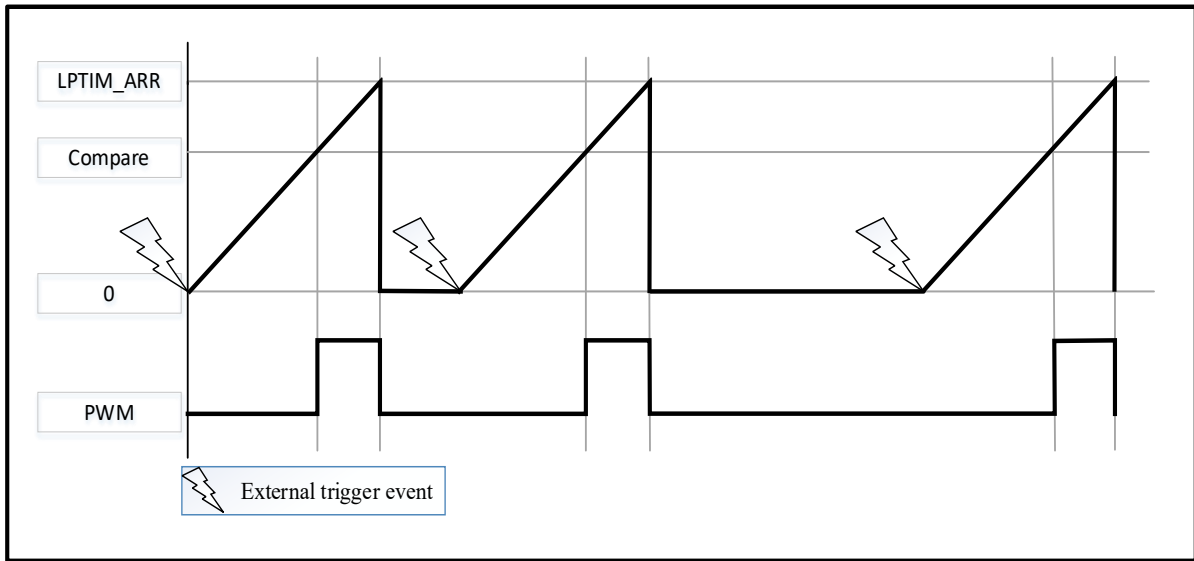
One-shot mode:

LPTIM_CTRL.SNGMST bit must be set to enable the one-shot mode. A trigger event will restart the LPTIM. Hardware will abandon all the trigger events after the internal counter starts and before the counter value equal to LPTIM_ARR.ARRVAL[15:0] value.

If an external trigger is selected, each external trigger event arriving after the LPTIM_CTRL.SNGMST bit is set, and after

the timer register is stopped (containing a zero value), the timer is restarted for a new count cycle, as shown in Figure 12-4.

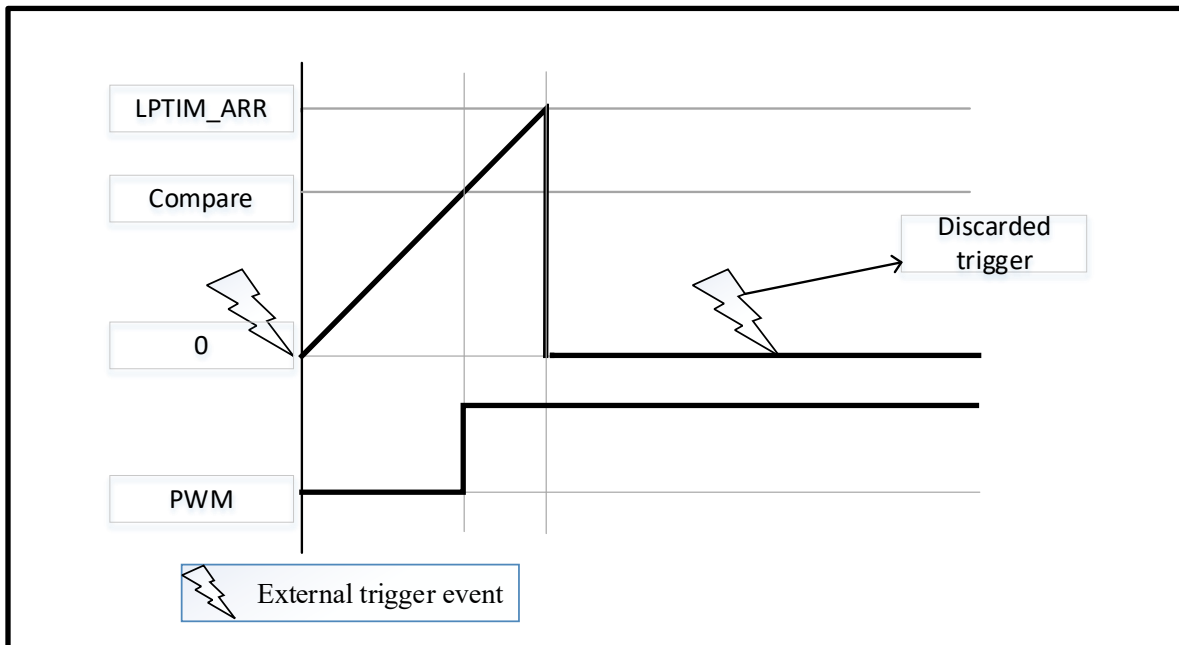
Figure 12-4 PTIM Output Waveform, Single Counting Mode Configuration



Set-once mode activated:

The Set-once mode is used when the LPTIM_CFG.WAVE bit is set. In Set-once mode, the counter is started once when the first trigger event happens, the hardware will discard any subsequent trigger event. As shown Figure 12-5.

Figure 12-5 LPTIM Output Waveform, Single Counting Mode Configuration and Set-Once Mode Activated



In case of software start (LPTIM_CFG.TRGEN[1:0] = '00'), the LPTIM_CTRL.SNGMST setting will start the counter for one-shot counting.

12.3.7 Timeout Function

When the LPTIM_CFG.TIMOUTEN bit is enable, the LPTIM counter will be reset by an effective edge from one selected trigger input.

When timeout function is used, the LPTIM counter will be reset and re-start by a selected trigger input event. If no trigger occurs within the configured time, the compare match event will happen. The waiting time is configured through the timeout value.

12.3.8 Waveform Generation

The LPTIM auto-reload register (LPTIM_ARR) and compare register (LPTIM_COMP) are used for generating LPTIM output waveforms.

The waveforms supported by are shown as below:

- PWM mode: LPTIM output is set when a COMP match event happens. (I.E. the LPTIM_CNT register value matched the LPTIM_COMP register value.) The LPTIM output is reset when an ARR match happens. (I.E. the LPTIM_CNT register value matched the LPTIM_ARR register value.)
- One-pulse mode: The first pulse is triggered same as PWM mode, then the output is permanently reset when the ARR match happens.
- Set-once mode: the output waveform is similar to the One-pulse mode except that the output is kept to the last signal level (depends on the output configured polarity).

Above waveform configuration require that LPTIM_ARR register value must be configured bigger than the LPTIM_COMP register value.

The LPTIM output waveform can be configured through the LPTIM_CFG.WAVE bit as follow:

- Clearing the LPTIM_CFG.WAVE bit will force the LPTIM to generate a PWM waveform or a One-pulse waveform depending on the set bit (LPTIM_CTRL.TSTCM or LPTIM_CTRL.SNGMST).
- LPTIM_CTRL.WAVE bit equals to '1' forces the LPTIM to generate a Set-once mode waveform.

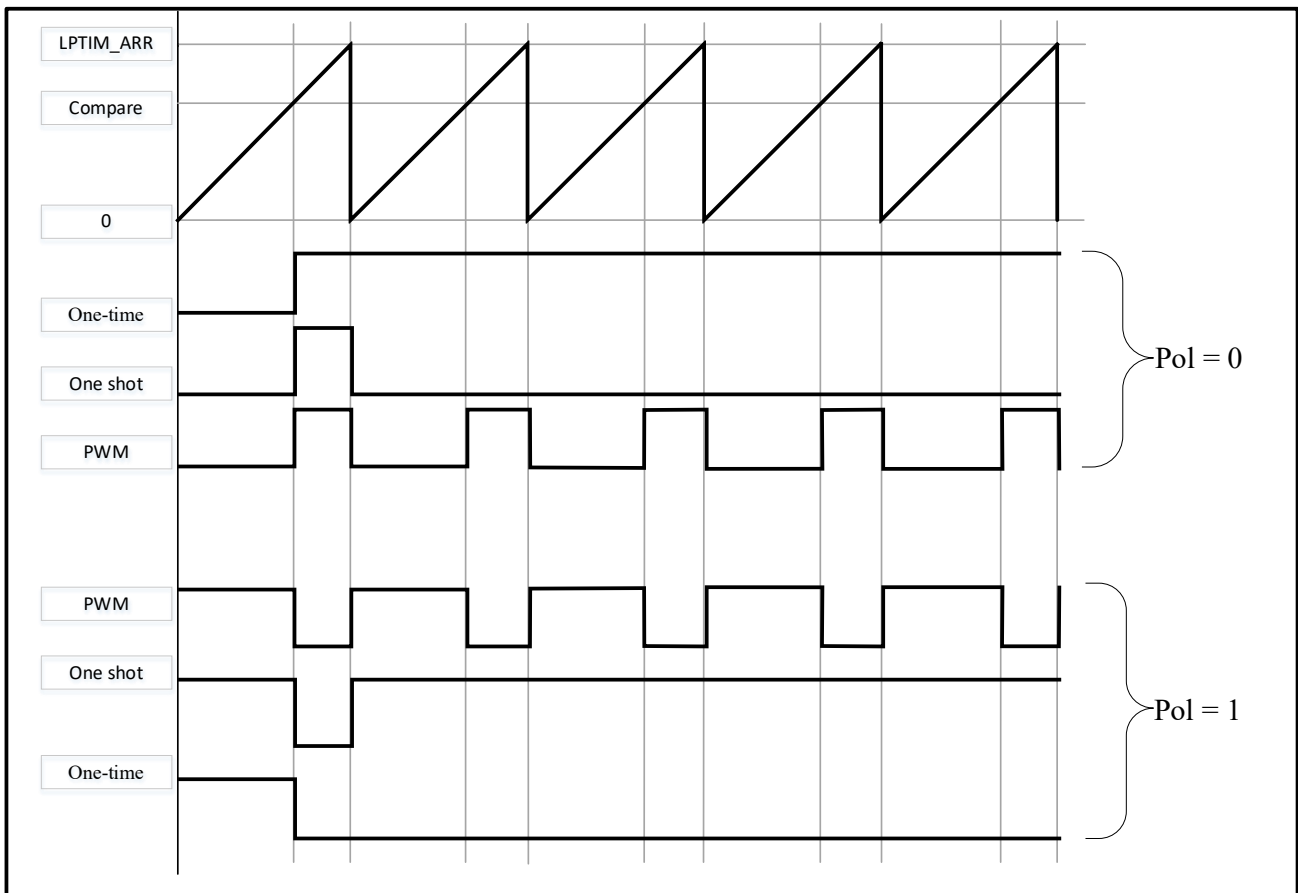
The LPTIM_CFG.WAVEPOL bit controls LPTIM output polarity. The output default value will change immediately after the user configured the polarity, even when the timer is disabled.

Signals with frequencies up to the LPTIM clock frequency divided by 2 can be generated. Only when LPTIM counter counting internal clock effective edge can achieve clock frequency divided by 2.

(I.E. LPTIM_CFG.CLKSEL = 0, LPTIM_CFG.CLKPOL[1:0] = 10, LPTIM_COMP.CMPVAL[15:0] = 'd1 (50% duty cycle)'/d2, LPTIM_ARR.ARRVAL[15:0] = 'd2. d1 and d2 means decimal 1, 2)

Figure 12-6 below shows the three possible waveforms that can be generated on the LPTIM output. Also, it shows the effect of the polarity change using the LPTIM_CFG.WAVEPOL bit.

Figure 12-6 Waveform Generation



12.3.9 Register Update

The LPTIM_ARR register and LPTIM_COMP register can be updated after the software writes. If the LPTIM is started, the LPTIM_ARR register and LPTIM_COMP register can be updated when counter overflow.

The LPTIM APB interface and the LPTIM kernel logic use different clocks, so there is some latency between the software write through APB bus and the moment when these values are available to the kernel logic. During this latency period, any additional write into these registers must be avoided.

The update method of LPTIM_ARR and LPTIM_COMP registers is determined by the LPTIM_CFG.RELOAD bit:

- LPTIM_CFG.RELOAD bit equals to '1': LPTIM_ARR and LPTIM_COMP registers are updated when counter overflow, if the LPTIM already started. When counter overflow, latency = 2~3 APB clock period.
- LPTIM_CFG.RELOAD bit equals to '0': LPTIM_ARR and LPTIM_COMP registers are updated after any software write access. Latency = 2~3 APB clock period + 2~3 LPTIM internal pre-scaled clock period.

The LPTIM_INTSTS.ARRUPD flag and the LPTIM_INTSTS.CMPUPD flag indicate when the write operation is completed to respectively the LPTIM_ARR register and the LPTIM_COMP register.

After a write to the LPTIM_ARR register or the LPTIM_COMP register, any successive write before respectively the LPTIM_INTSTS.ARRUPD flag or the LPTIM_INTSTS.CMPUPD flag be set, will lead to unpredictable results. So a new write operation to the same register can only be performed when the previous write operation is completed.

12.3.10 Counter Mode

The internal counter can count external trigger events from LPTIM input1 or internal clock cycles. This can be configured through LPTIM_CFG.CLKSEL and LPTIM_CFG.CNTMEN bits.

If LPTIM is counting external triggers, user can configure LPTIM_CFG.CLKPOL[1:0] bits to select the effective edge from rising edge, falling edge or both edges.

The count modes below can be selected, depending on LPTIM_CFG.CLKSEL and LPTIM_CFG.CNTMEN bits values:

- LPTIM_CFG.CLKSEL = 0: the LPTIM use an internal clock source to clock.
 - LPTIM_CFG.CNTMEN = 0, The LPTIM is configured to be clocked by an internal clock source and the LPTIM counter is configured to be updated following each internal clock pulse.
 - LPTIM_CFG.CNTMEN = 1, The LPTIM external Input1 is sampled with the internal clock provided to the LPTIM. In order to not miss any event, the frequency of the changes on the external Input1 signal should never exceed the frequency of the internal clock provided to the LPTIM. Also, the internal clock provided to the LPTIM must not be pre-scaled (LPTIM_CFG.CLKPRE[2:0] = 000).
- LPTIM_CFG.CLKSEL = 1: the LPTIM use an external clock source to clock.
 - LPTIM_CFG.CNTMEN bit value is don't care. In this configuration, the LPTIM has no need for an internal clock source (except if the glitch filters are enabled). The signal injected on the LPTIM external Input1 is used as system clock for the LPTIM. This configuration is suitable for operation modes where no embedded oscillator is enabled.
 - For this configuration, the LPTIM counter can be updated either on rising edges or falling edges of the input1 clock signal but not on both rising and falling edges.
 - Since the signal injected on the LPTIM external Input1 is also used to clock the LPTIM kernel logic, there is some initial latency (after the LPTIM is enabled) before the counter is incremented. More precisely, the first two to five active edges on the LPTIM external Input1 (after LPTIM is enable) are lost.

12.3.11 Timer Enable

The LPTIM_CTRL.LPTIMEN bit is used to enable/disable the LPTIM kernel logic. After setting the LPTIM_CTRL.LPTIMEN bit, a delay of two counter clock is needed before the LPTIM is turned on.

The LPTIM_CFG and LPTIM_INTEN registers must be modified only when the LPTIM is turned off.

12.3.12 Encoder Mode

The Encoder mode can handle signals from quadrature encoders which used to detect angular position of rotary elements. The encoder mode allows the counter counts the events within 0 and LPTIM_ARR.ARRVAL[15:0] value. (0 up to LPTIM_ARR.ARRVAL[15:0] or LPTIM_ARR.ARRVAL[15:0] to 0). In this case, user must configure LPTIM_ARR.ARRVAL[15:0] before enable the counter. From external input1 and input2, a clock is generated for the counter. The counting direction depends on the phase between these two input signals.

The Encoder mode is only available when the LPTIM use an internal clock source to clock. The signals frequency on both Input1 and Input2 inputs must not exceed the LPTIM internal clock frequency divided by 4. This is mandatory in order to guarantee a proper operation of the LPTIM.

The change of counting direction is updated by LPTIM_INTSTS.DOWN and LPTIM_INTSTS.UP flags. Also, an interrupt

can be generated for both direction change events if enabled through the LPTIM_INTEN register.

User can enable Encoder mode by setting LPTIM_CFG.ENC bit. And the LPTIM need to be configured in continuous mode first.

When Encoder mode is active, the LPTIM counter is modified automatically following the speed and the direction of the incremental encoder. Therefore, its content always represents the encoder's position. The count direction, signaled by LPTIM_INTSTS.DOWN and LPTIM_INTSTS.UP flags, correspond to the rotation direction of the encoder rotor.

Different counting scenarios may occur base on the different trigger edges configured by the LPTIM_CFG.CLKPOL[1:0] bits. The following table summarizes the possible combinations, assuming that Input1 and Input2 do not switch at the same time.

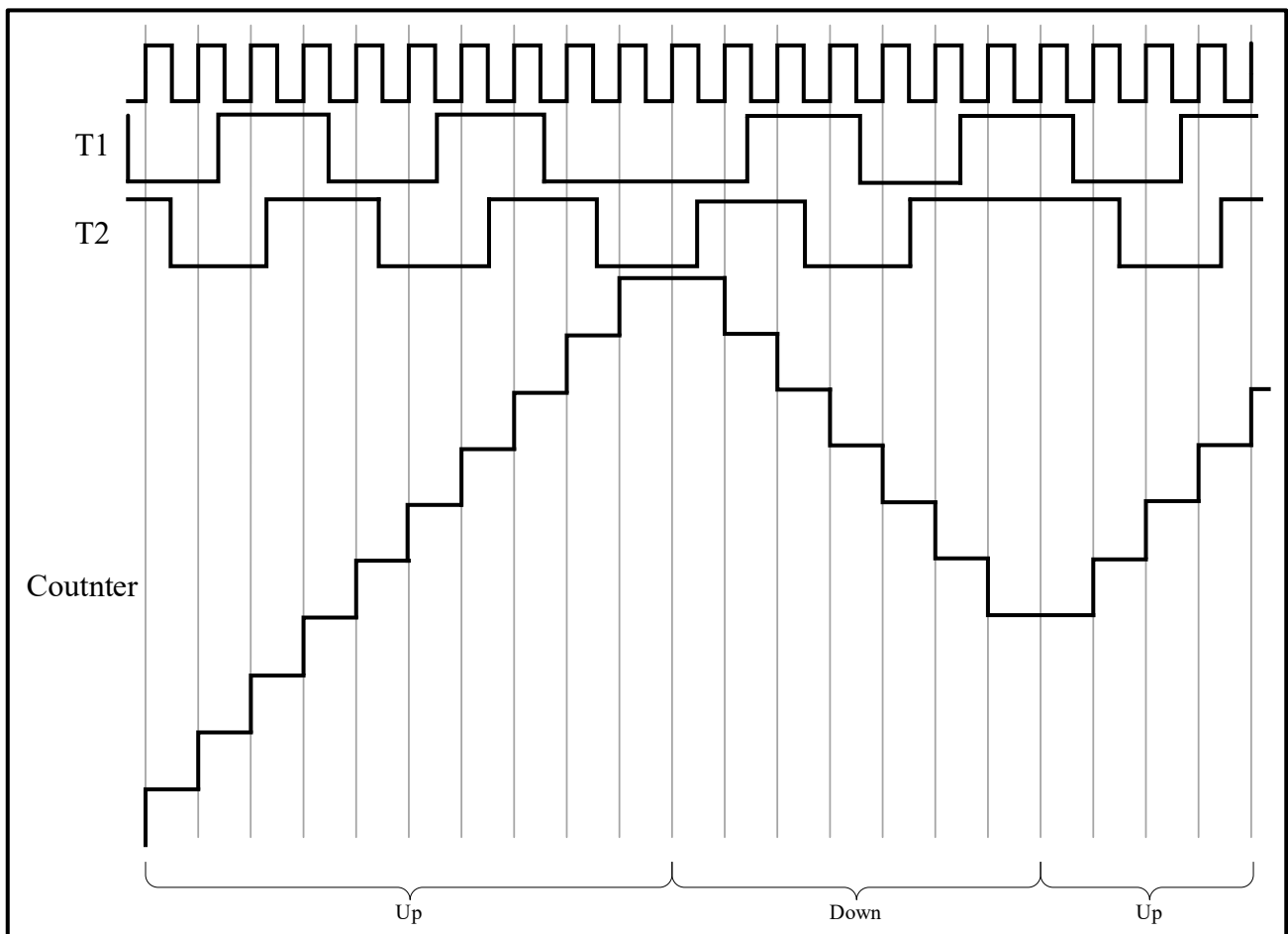
Table 12-3 Encoder Counting Scenarios

Trigger Edge	The Signal Is Opposite (Input1 for Input2, Input2 for Input1)	Input1 Signal		Input2 Signal	
		Rising	Falling	Rising	Falling
Rising Edge	High	Down	No count	Up	No count
	Low	Up	No count	Down	No count
Falling Edge	High	No count	Up	No count	Down
	Low	No count	Down	No count	Up
Both Edges	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

The following figure shows a counting sequence for Encoder mode configured with rising and falling edges triggers.

Caution: In this mode the LPTIM must be clocked by an internal clock source, so the LPTIM_CFG.CLKSEL bit must be maintained to its reset value which is equal to '0'. Also, the prescaler division ratio must be equal to its reset value which is 1 (LPTIM_CFG.CLKPRE[2:0] bits must be '000').

Figure 12-7 Encoder Mode Counting Sequence



12.3.13 Non-quadrature Encoder Mode

This mode allows handling signals from non-quadrature encoders, which is used to detect sub-sequent positive pulses from external interface. Non-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 programmed into the LPTIM_ARR register (0 up to ARR or ARR down to 0 depending on the direction). Therefore you must configure LPTIM_ARR before starting. From the two external input signals, Input1 and Input2, a clock signal is generated to clock the LPTIM counter. The order between those two signals determines the counting direction.

The Non-Encoder mode is only available when the LPTIM is clocked by an internal clock source. The signals frequency on both Input1 and Input2 inputs must not exceed the LPTIM internal clock frequency divided by 4. This is mandatory in order to guarantee a proper operation of the LPTIM.

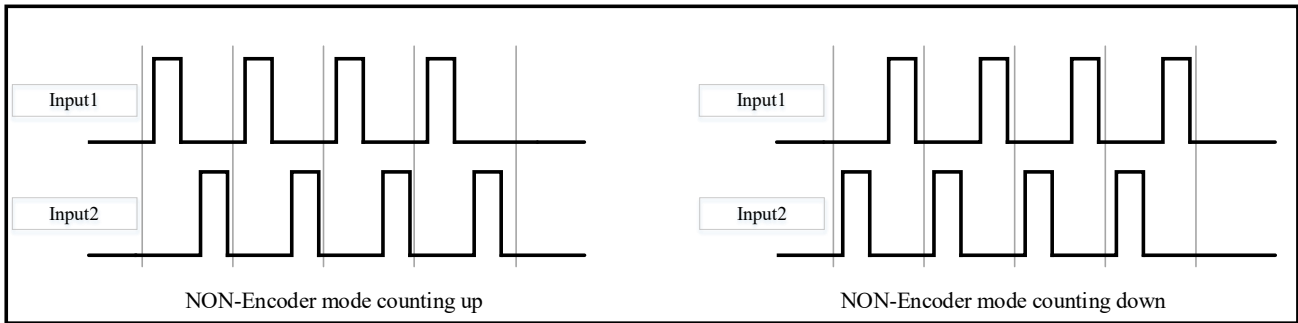
Direction change is signaled by LPTIM_INTSTS.DOWN and LPTIM_INTSTS.UP flags. Also, an interrupt can be generated for both direction change events if enabled through the LPTIM_INTEN register.

To activate the Non-Encoder mode the LPTIM_CFG.NENC bit has to be set to '1'. The LPTIM must first be configured in Continuous mode.

When Non-Encoder mode is active, the LPTIM counter is modified automatically following the speed and the direction of the incremental encoder. Therefore, its content always represents the encoder's position. The count direction, signaled by LPTIM_INTSTS.DOWN and LPTIM_INTSTS.UP flags, correspond to the rotation direction of the encoder rotor.

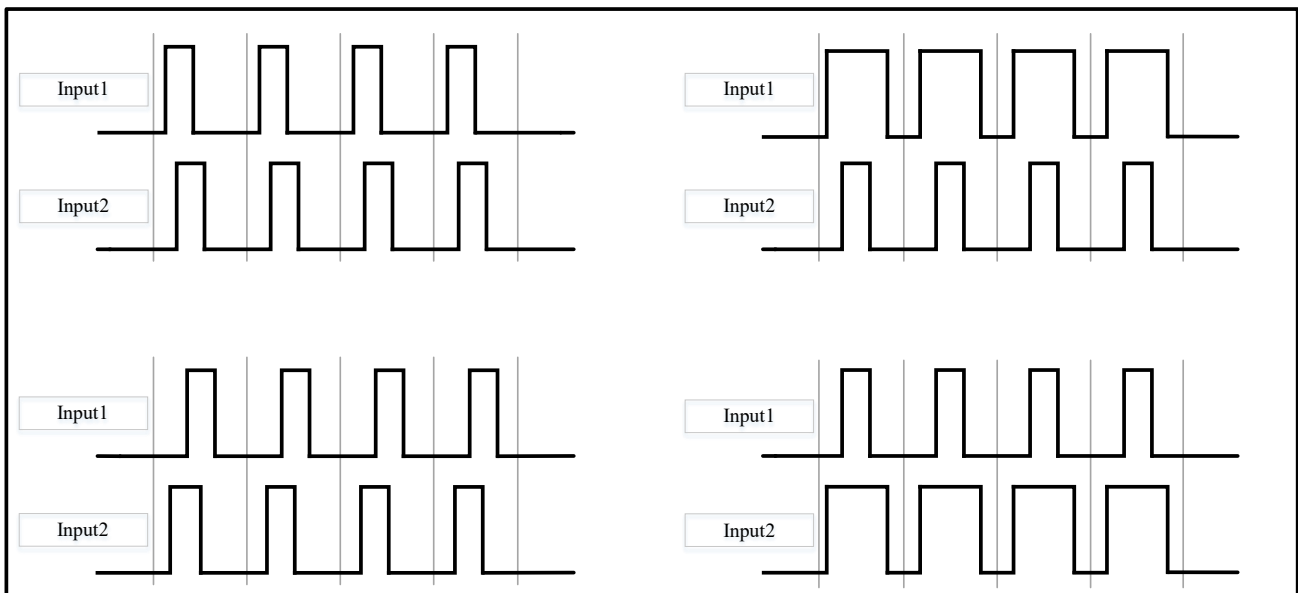
The following two waveforms, the decoder module can operate properly, when there is no case that both Input1 and Input2 are high.

Figure 12-8 Input Waveforms of Input1 and Input2 When The Decoder Module Is Working Normally



If the Input1 and Input2 waveform is as following, the decoder module can't operate properly. The counter will ignore these waveforms and keep the previous value.

Figure 12-9 Input1 and Input2 Input Waveforms When Decoder Module Is Not Working



12.3.14 LPTIM Interrupts

The following events generate an interrupt/wake-up event, if they are enabled through the LPTIM_INTEN register:

- Compare match
- Auto-reload match (whatever the direction if encoder mode)
- External trigger event
- Auto-reload register write completed
- Compare register write completed
- Direction change (encoder mode), programmable(up / down / both).

Note: If any bit in the LPTIM_INTEN register (Interrupt Enable Register) is set after that its corresponding flag in the LPTIM_INTSTS register (Status Register) is set, the interrupt is not asserted. Configuration of EXTI_LINE23 is required

to use LPTIM interrupts.

Table 12-4 Interruption Events

Corresponding Interrupt Event	Describe
Compare match	Interrupt flag is set when LPTIM_CNT (counter register value) = LPTIM_COMP (compare register value).
Auto reload match	Interrupt flag is set when LPTIM_CNT (counter register value) = LPTIM_ARR (auto-reload register value).
External trigger event	Interrupt flag is set when an external trigger event is detected.
Auto-reload register update OK	Interrupt flag is set when the write operation to the LPTIM_ARR register is complete.
Compare register update OK	Interrupt flag is set when the write operation to the LPTIM_COMP register is complete.
Direction change	Used in Encoder mode. Two interrupt flags are embedded to signal direction change: –UP flag indicated that the count direction is changed to count up. –DOWN flag indicated that the count direction is changed to count down.

12.4 LPTIM Registers

12.4.1 LPTIM Register Overview

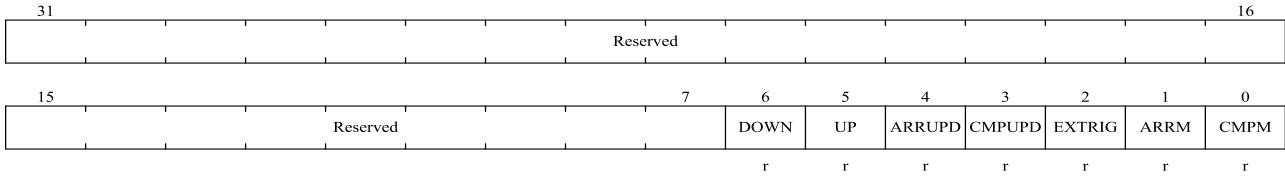
Table 12-5 LPTIM Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																		
000h	LPTIM_INTSTS	Reserved																								DOWN	UP	ARRUPD	CMPUPD	EXTRIG	AREM	CMPM	0																		
	Reset Value	0																								0	0	0	0	0	0	0																			
004h	LPTIM_INTCLR	Reserved																								DOWNCF	UPCF	ARRUPDCF	CMPUPDCF	EXTRIGCF	ARRMCF	CMPMCF	0																		
	Reset Value	0																								0	0	0	0	0	0	0																			
008h	LPTIM_INTEN	Reserved																								DOWNIE	UPIE	ARRUPDIE	CMPUPDIE	EXTRIGIE	ARRMIE	CMPMIE	0																		
	Reset Value	0																								0	0	0	0	0	0	0																			
00Ch	LPTIM_CFG	Reserved							NENC	ENC	CNTMEN	RELOAD	WAVEPOL	WAVE	TIMOUTEN	TRGEN[1:0]	Reserved	TRGSEL[2:0]	Reserved	CLKPRE[2:0]	Reserved	TRIGFLT[1:0]	Reserved	CLKFLT[1:0]	CLKPOL[1:0]	CLKSEL	0	0	0	0	0	0	0	0																	
	Reset Value	0							0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																
010h	LPTIM_CTRL	Reserved																								TSTCM	SNGMST	LPTIMEN	0	0	0																				
	Reset Value	0																								0	0	0																							
014h	LPTIM_CMP	Reserved															CMPVAL[15:0]																																		
	Reset Value	0															0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
018h	LPTIM_ARR	Reserved															ARRVAL[15:0]																																		
	Reset Value	0															0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
01Ch	LPTIM_CNT	Reserved															CNTVAL[15:0]																																		
	Reset Value	0															0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

12.4.2 LPTIM Interrupt and Status Register (LPTIM_INTSTS)

Address offset: 0x00

Reset value: 0x0000 0000

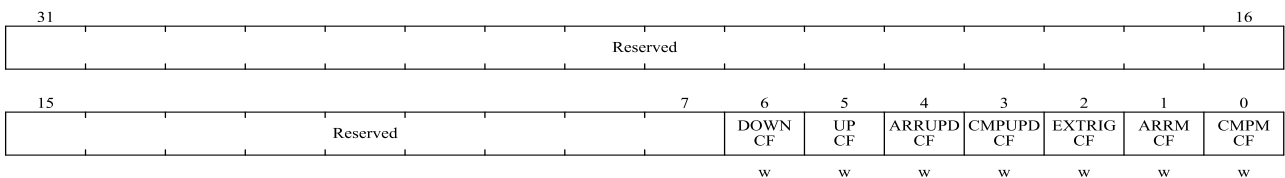


Bit Field	Name	Description
31:7	Reserved	Reserved, the reset value must be maintained.
6	DOWN	Change counter direction to down. In Encoder mode, hardware will set DOWN bit to inform the application the counter direction.
5	UP	Change counter direction to up. In Encoder mode, hardware will set UP bit to inform the application the counter direction.
4	ARRUPD	Auto-reload value updated to register. Hardware sets ARRUPD to inform application that LPTIM_ARR register has been written by the APB1 bus successfully.
3	CMPUPD	Compare value updated to register. Hardware sets COMPUPD to inform application that LPTIM_COMP register has been written by the APB1 bus successfully.
2	EXTRIG	External trigger valid event. Hardware sets EXTRIG to inform application that a valid external trigger edge has occurred. If the trigger is discarded when timer has already started, then this flag is not set.
1	ARRM	Auto-reload match. Hardware set this to inform application that LPTIM_CNT register value reached the LPTIM_ARR register's value.
0	CMPM	Compare match. Hardware set this to inform application that LPTIM_CNT register value reached the LPTIM_COMP register's value.

12.4.3 LPTIM Interrupt Clear Register (LPTIM_INTCLR)

Address offset: 0x04

Reset value: 0x0000 0000



Bit Field	Name	Description
31: 7	Reserved	Reserved, the reset value must be maintained.
6	DOWNCF	Direction change to down Clear Flag Writing 1 to this bit clear the DOWN flag in the LPTIM_INTSTS register

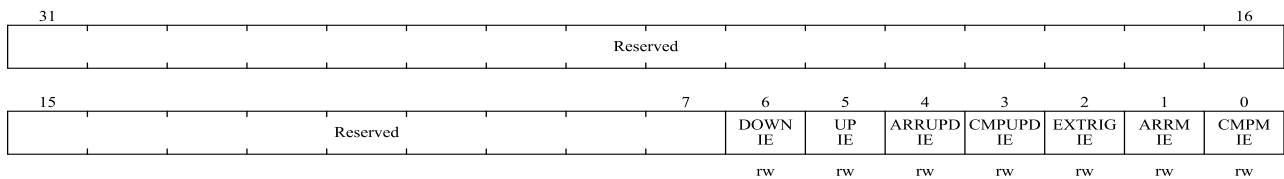
Bit Field	Name	Description
5	UPCF	Direction change to UP Clear Flag Writing 1 to this bit clear the UP flag in the LPTIM_INTSTS register
4	ARRUPDCF	Auto-reload register update OK Clear Flag Writing 1 to this bit clears the ARRUPD flag in the LPTIM_INTSTS register
3	CMPUPDCF	Compare register update OK Clear Flag Writing 1 to this bit clears the CMPUPD flag in the LPTIM_INTSTS register
2	EXTRIGCF	External trigger valid edge Clear Flag Writing 1 to this bit clears the EXTRIG flag in the LPTIM_INTSTS register
1	ARRMCF	Auto-reload match Clear Flag Writing 1 to this bit clears the ARRM flag in the LPTIM_INTSTS register
0	CMPMCF	Compare match Clear Flag Writing 1 to this bit clears the CMPM flag in the LPTIM_INTSTS register

12.4.4 LPTIM Interrupt Enable Register (LPTIM_INTEN)

Address offset: 0x08

Reset value: 0x0000 0000

Note: The LPTIM_INTEN register must only be modified when the LPTIM is disabled (LPTIM_CTRL.LPTIMEN bit reset to '0')



Bit Field	Name	Description
31:7	Reserved	Reserved, the reset value must be maintained.
6	DOWNIE	Direction change to down interrupt enable 0: DOWN interrupt disabled 1: DOWN interrupt enabled
5	UPIE	Direction change to up interrupt enable 0: UP interrupt disabled 1: UP interrupt enabled
4	ARRUPDIE	Auto reload register update succeeded interrupt enable bit. 0: ARRUPD interrupt disable 1: ARRUPD interrupt enable
3	CMPUPDIE	Compare register update OK interrupt enable 0: CMPUPD interrupt disabled 1: CMPUPD interrupt enabled
2	EXTRIGIE	External trigger valid edge interrupt enable 0: EXTRIG interrupt disabled 1: EXTRIG interrupt enabled
1	ARRMIE	Auto reload match interrupt enable bit. 0: ARRM interrupt disabled

Bit Field	Name	Description
		1: ARRM interrupt enabled
0	CMPMIE	Compare match interrupt enable 0: CMPM interrupt disabled 1: CMPM interrupt enabled

12.4.5 LPTIM Configuration Register (LPTIM_CFG)

Address offset: 0x0C

Reset value: 0x0000 0000

Note: The LPTIM_CFG register must only be modified when the LPTIM is disabled (LPTIM_CTRL.LPTIMEN bit reset to '0')

31	Reserved				25	24	23	22	21	20	19	18	17	16
		NENC	ENC	CNTMEN	RELOAD	WAVEPOL	WAVE	TIMOUT EN	TRGEN[1:0]	Reserved				
15	13	12	11	9	8	7	6	5	4	3	2	1	0	
TRGSEL[2:0]		Reserved	CLKPRE[2:0]		Reserved	TRIGFLT[1:0]		Reserved	CLKFLT[1:0]		CLKPOL[1:0]		CLKSEL	
rw			rw			rw			rw		rw		rw	

Bit Field	Name	Description
31:26	Reserved	Reserved, the reset value must be maintained.
25	NENC	Non-Orthogonal mode enable 0: Non-Orthogonal mode disabled 1: Non-Orthogonal mode enabled
24	ENC	Encoder mode enable 0: Encoder mode disabled 1: Encoder mode enabled
23	CNTMEN	Counter mode enabled The CNTMEN bit selects clock source for the LPTIM counter: 0: Counter is incremented following each internal clock pulse 1: Counter is incremented following each valid clock pulse on the LPTIM external Input1
22	RELOAD	Registers update mode The RELOAD bit controls the LPTIM_ARR and the LPTIM_COMP registers update mode 0: Registers are updated after each APB1 bus write access 1: Registers are updated at the end of the current LPTIM period <i>Note: When RELOAD=0, ARRUPD and CMPUPD interrupts cannot be generated, the LPTIM_ARR and LPTIM_COMP registers need to be configured before enabling LPTIM</i>
21	WAVEPOL	Waveform shape polarity The WAVEPOL bit controls the output polarity 0: The LPTIM output reflects the compare results between LPTIM_ARR and LPTIM_COMP registers 1: The LPTIM output reflects the inverse of the compare results between

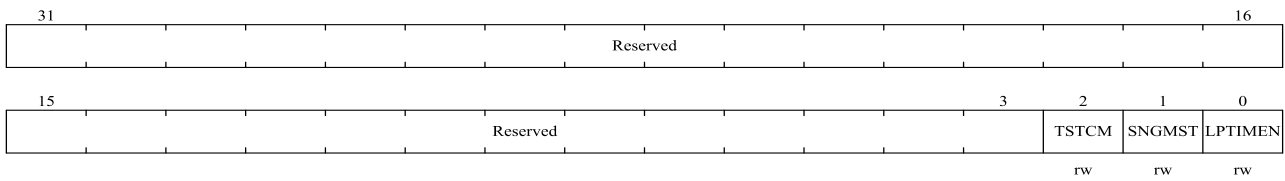
Bit Field	Name	Description
		LPTIM_ARR and LPTIM_COMP registers
20	WAVE	Waveform shape The WAVE bit controls the output shape 0: Deactivate Set-once mode, PWM/One Pulse waveform (depending on LPTIM_CTRL.TSTCM or LPTIM_CTRL.SNGMST bit) 1: Activate the Set-once mode
19	TIMOUTEN	Timeout enable 0: A trigger event arriving when the timer is already started will be ignored 1: A trigger event arriving when the timer is already started will reset and restart the counter
18:17	TRGEN[1:0]	Trigger enable and polarity The TRGEN bits controls whether the LPTIM counter is started by an external trigger or not. If the external trigger option is selected, three configurations are possible for the trigger active edge: 00: Software trigger (counting start is initiated by software) 01: Rising edge is the active edge 02: Falling edge is the active edge 03: Both edges are active edges
16	Reserved	Reserved, the reset value must be maintained.
15:13	TRGSEL[2:0]	Trigger selector The trigger source is selected from the following 6 available sources as the trigger event for LPTIM: 000: PB6 or PA6 001: RTC alarm A 010: RTC alarm B 011: RTC_TAMP1 100: RTC_TAMP2 101: Reserved 110: COMP_OUT 111: Reserved
12	Reserved	Reserved, the reset value must be maintained.
11:9	CLKPRE[2:0]	Clock division factor bit. 000: / 1 001: / 2 010: / 4 011: / 8 100: / 16 101: / 32 110: / 64 111: / 128
8	Reserved	Reserved, the reset value must be maintained.
7:6	TRIGFLT[1:0]	Configure the data filter trigger bit.

Bit Field	Name	Description
		<p>The TRIGFLT value sets the number of consecutive equal samples that should be detected when a level change occurs on an internal trigger before it is considered as a valid level transition. An internal clock source must be present to use this feature</p> <p>00: Any trigger active level change is considered as a valid trigger.</p> <p>01: Trigger active level change must be stable for at least 2 clock periods before it is considered as valid trigger.</p> <p>10: Trigger active level change must be stable for at least 4 clock periods before it is considered as valid trigger.</p> <p>11: Trigger active level change must be stable for at least 8 clock periods before it is considered as valid trigger.</p>
5	Reserved	Reserved, the reset value must be maintained.
4:3	CLKFLT[1:0]	<p>Digital filter external clock input configuration</p> <p>The CLKFLT value sets the number of consecutive equal samples that should be detected when a level change occurs on an external clock signal before it is considered as a valid level transition. An internal clock source must be present to use this feature</p> <p>00: Any external clock signal level change is considered as a valid transition.</p> <p>01: External clock signal level change must be stable for at least 2 clock periods before it is considered as valid transition.</p> <p>10: External clock signal level change must be stable for at least 4 clock periods before it is considered as valid transition.</p> <p>11: External clock signal level change must be stable for at least 8 clock periods before it is considered as valid transition.</p>
2:1	CLKPOL[1:0]	<p>Clock polarity</p> <p>When the LPTIM is clocked by an external clock source, CLKPOL bits is used to configure the active edge or edges used by the counter:</p> <p>00: The rising edge is the active edge used for counting</p> <p>01: The falling edge is the active edge used for counting</p> <p>10: Both edges are active edges. When both external clock signal edges are considered active ones, the LPTIM must also be clocked by an internal clock source with a frequency equal to at least four time the external clock frequency.</p> <p>11: Not allowed</p> <p>If the LPTIM is configured in Encoder mode (LPTIM_CFG.ENC bit is set):</p> <p>00: The encoder rising edge counting mode.</p> <p>01: The encoder falling edge counting mode.</p> <p>02: The encoder both edges counting mode.</p>
0	CLKSEL	<p>Clock selector</p> <p>The CLKSEL bit selects which clock source the LPTIM will use:</p> <p>0: LPTIM is clocked by internal clock source (APB1 clock or any of the embedded oscillators)</p> <p>1: LPTIM is clocked by an external clock source through the LPTIM external Input1</p>

12.4.6 LPTIM Control Register (LPTIM_CTRL)

Address offset: 0x10

Reset value: 0x0000 0000

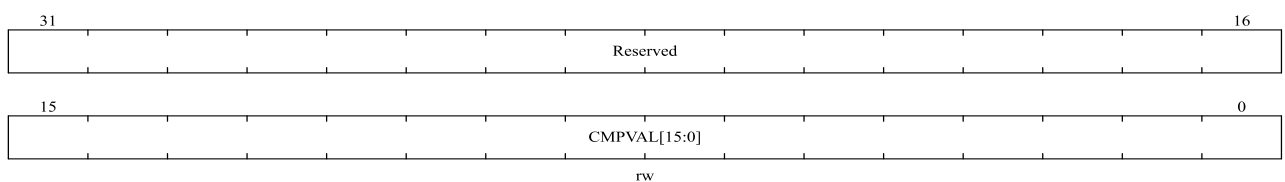


Bit Field	Name	Description
31:3	Reserved	Reserved, the reset value must be maintained.
2	TSTCM	Timer start in Continuous mode This bit is set by software and cleared by hardware. In case of software start (LPTIM_CFG.TRGEN[1:0] = '00'), setting this bit starts the LPTIM in Continuous mode. If the software start is disabled (TRGEN[1:0] ≠ '00'), setting this bit starts the timer in Continuous mode as soon as an external trigger is detected. If this bit is set when a single pulse mode counting is ongoing, then the timer will not stop at the next match between the LPTIM_ARR and LPTIM_CNT registers and the LPTIM counter keeps counting in Continuous mode. This bit can be set only when the LPTIM is enabled. It will be automatically reset by hardware.
1	SNGMST	LPTIM start in Single mode This bit is set by software and cleared by hardware. In case of software start (LPTIM_CFG.TRGEN[1:0] = '00'), setting this bit starts the LPTIM in single pulse mode. If the software start is disabled (LPTIM_CFG.TRGEN[1:0] ≠ '00'), setting this bit starts the LPTIM in single pulse mode as soon as an external trigger is detected. If this bit is set when the LPTIM is in continuous counting mode, then the LPTIM will stop at the following match between LPTIM_ARR and LPTIM_CNT registers. This bit can only be set when the LPTIM is enabled. It will be automatically reset by hardware.
0	LPTIMEN	LPTIM enable The LPTIMEN bit is set and cleared by software. 0: LPTIM is disabled 1: LPTIM is enabled

12.4.7 LPTIM Compare Register (LPTIM_COMP)

Address offset: 0x14

Reset value: 0x0000 0000

Note: The LPTIM_COMP register must only be modified when the LPTIM is enabled (LPTIM_CTRL.LPTIMEN bit reset to '1')


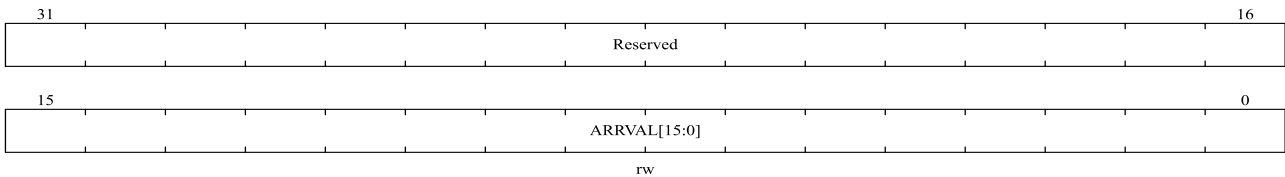
Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained.
15:0	CMPVAL[15:0]	Compare value CMPVAL is the compare value used by the LPTIM.

12.4.8 LPTIM Auto-Reload Register (LPTIM_ARR)

Address offset: 0x18

Reset value: 0x0000 0001

Note: The LPTIM_ARR register must only be modified when the LPTIM is enabled (LPTIM_CTRL.LPTIMEN bit reset to '1')

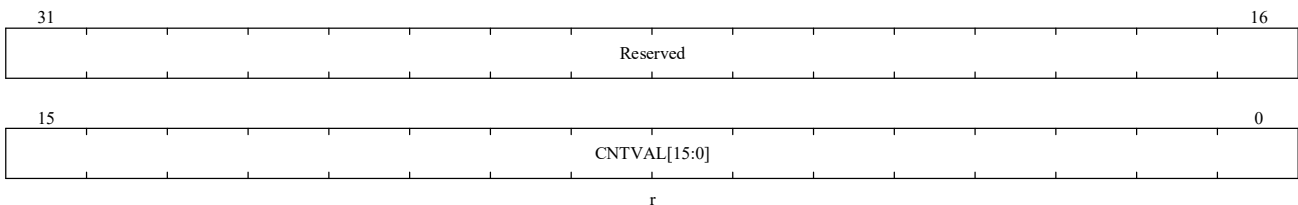


Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained.
15:0	ARRVAL[15:0]	Auto reload value ARRVAL is the auto-reload value for the LPTIM. This value must be strictly greater than the LPTIM_COMP.CMPVAL[15:0] value.

12.4.9 LPTIM Counter Register (LPTIM_CNT)

Address offset: 0x1C

Reset value: 0x0000 0000



Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained.
15:0	CNTVAL[15:0]	Counter value When the LPTIM is running with an asynchronous clock, reading the LPTIM_CNT register may return unreliable values. So in this case it is necessary to perform two consecutive read accesses and verify that the two returned values are identical. If identical, the reading is reliable.

13 Independent Watchdog (IWDG)

13.1 Introduction

The N32G032 has embedded independent watchdog (IWDG) and window watchdog (WWDG) timers to solve the problems caused by software errors. The watchdog timer is very flexible to use, which improves the security of the system and the accuracy of timing control.

The independent watchdog (IWDG) is driving by low-speed internal clock (LSI clock) running at 30 KHz, which will still running event dead loop or MCU stuck is happening. This can provide higher safety level, timing accuracy and flexibility of watchdog. It can reset and resolve system malfunctions due to software failure. The IWDG is best suited for applications that require the watchdog to run as a totally independent process outside the main application, but have lower timing accuracy constraints.

When the power control register PWR_CTRL2.IWDGRSTEN bit is '1' and the IWDG counter reaches 0, a system reset will be generated (if this bit is '0', the IWDG will count but not reset). IWDG reset can also be used for low power wake up.

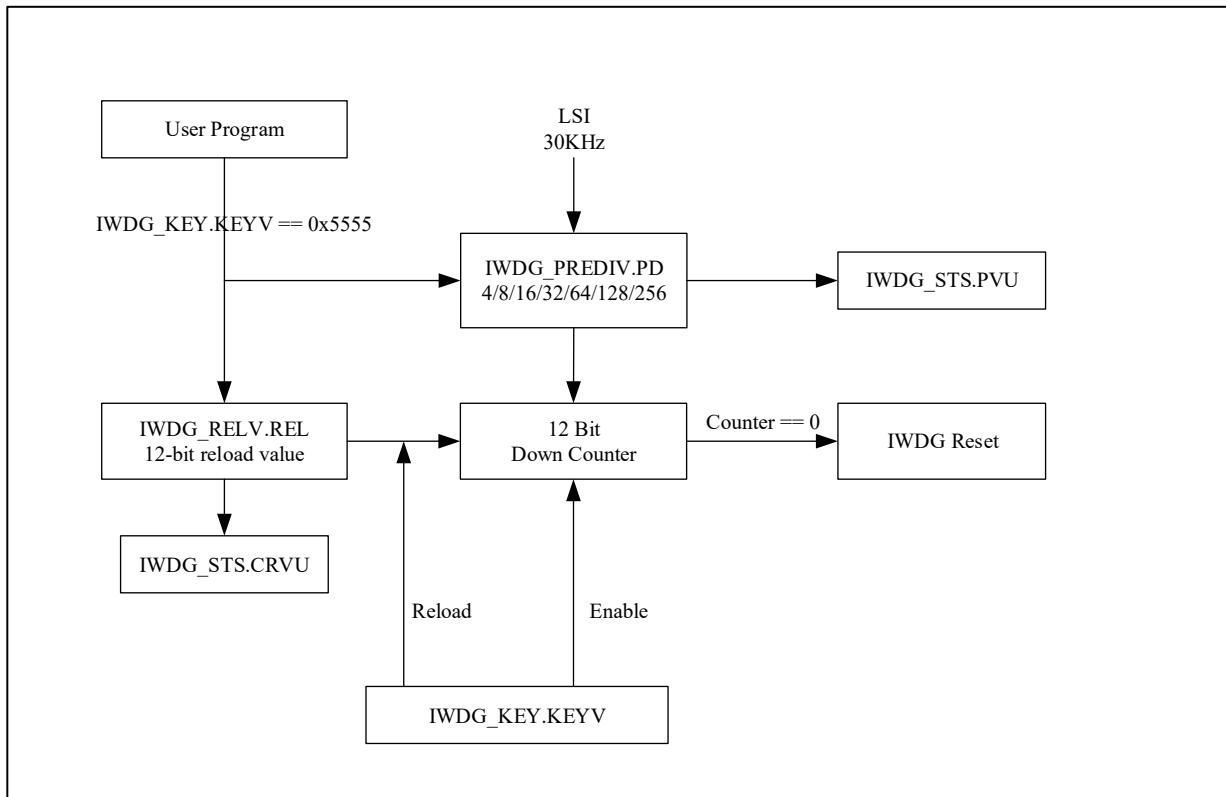
Note: This chapter is based on the system default value IWDGRSTEN = 1.

13.2 Main Features

- A 12-bit down-counter that runs independently
- RC oscillator provides an independent clock source and can operate normally in STOP mode
- Reset and low-power wake-up can be matched.
- A system reset occurs when the down counter reaches 0x0000(if watchdog activated)

13.3 Function Description

Figure 13-1 Functional Block Diagram of The Independent Watchdog Module



To enable IWDG, we need to write 0xCCCC to IWDG_KEY.KEYV[15:0] bits. Counter starts counting down from reset value (0xFFFF). When counter count to 0x000, it generates a reset signal (IWDG_RESET) to MCU. Other than that, as long as 0xAAAA (reload request) is write to IWDG_KEY.KEYV[15:0] bits before reset, the counter value is set to the reload value in the IWDG_RELV.REL[11:0] bits and prevents the watchdog from resetting the entire device.

If the "hardware watchdog timer" function is enabled through the option byte, the watchdog will automatically start running after the system is powered on and will generate a system reset, unless the software reloads the counter before it reaches '0'.

13.3.1 Register Access Protection

IWDG_PREDIV and IWDG_RELV register are write protected. To modify the value of those two register, user needs to write 0x5555 to IWDG_KEY.KEYV[15:0] bits. Writing other value enables write protections again. IWDG_STS.PVU indicates whether the pre-scaler value update is on going. IWDG_STS.CRVU indicates whether the IWDG is updating the reload value. The hardware sets the IWDG_STS.PVU bit and/or IWDG_STS.CRVU bit when the pre-scaler value and/or reload value is updating. After the pre-scaler value and/or reload value update is complete, the hardware clears the IWDG_STS.PVU bit and/or IWDG_STS.CRVU bit.

The reload operation (IWDG_KEY.KEYV[15:0] configured with value of 0xAAAA) will also cause the registers to become write protected again.

13.3.2 Debug Mode

In debug mode (Cortex[®]-M0 core stops), IWDG counter will either continue to operate normally or stops, depending on DBG_CTRL.IWDG_STOP bit in debug module. If this bit is set to '1', the counter stops. The counter operates normally when the bit is '0'. For details, refer to Section 3.3.2 Peripheral Debug Support.

13.4 User Interface

IWDG module user interface contains 4 registers: Key Register (IWDG_KEY), Pre-scale Register (IWDG_PREDIV), Reload Register (IWDG_RELV) and Status Register (IWDG_STS).

13.4.1 Operate Process

When IWDG is enable from reset from software (write 0xAAAA to IWDG_KEY.KEYV[15:0] bits) or hardware (clear WDG_SW bit). It starts counting down from 0xFFFF. Down counting gap is determined by pre-scale LSI clock. Once the counter is reloaded, each new round will start from the value in IWDG_RELV.REL[11:0] instead of 0xFFFF.

When program is running normally, software needs to feed IWDG before counter reaches 0 and start a new round of down counting. When counter reaches 0, this indicates program malfunction. IWDG generates reset signal under this circumstance.

If user wants to configure IWDG prescaler and reload value register, it needs to write 0x5555 to IWDG_KEY.KEYV[15:0] first. Then confirm IWDG_STS.CRVU bit and IWDG_STS.PVU bit. IWDG_STS.CRVU bit indicates reload value update is ongoing, IWDG_STS.PVU indicates prescaler divider ratio is updating. Only when those two bit are 0 then user can update corresponding value. When update is on-going, hardware sets corresponding bit to 1. At this time, reading IWDG_PREDIV.PD[2:0] or IWDG_RELV.REL[11:0] is invalid since data needs to be synchronized to LSI clock domain. The value read from IWDG_PREDIV.PD[2:0] or IWDG_RELV.REL[11:0] will be valid after hardware clears the IWDG_STS.PVU bit or IWDG_STS.CRVU bit.

If the application uses more than one reload value or pre-scaler value, it must wait until the IWDG_STS.CRVU bit is reset before changing the reload value, the same as changing the pre-scaler value. However, after updating the pre-scale and/or the reload value, it is not necessary to wait until IWDG_STS.CRVU bit or IWDG_STS.PVU bit are reset before continuing code execution (even in case of low-power mode entry, the write operation is taken into account and will complete).

Pre-scale register and reload register controls the time that generates reset, as shown in Table 13-1.

Table 13-1 IWDG Counting Maximum and Minimum Reset Time

Pre-scale Factor	PD[2:0]	Min Timeout (ms) REL [11:0]=0x000	Max Timeout (ms) REL [11:0]=0xFFFF
/ 4	000	0.13	546.13
/ 8	001	0.26	1092.27
/ 16	010	0.53	2184.53
/ 32	011	1.07	4369.07
/ 64	100	2.13	8738.13
/ 128	101	4.26	17476.27
/ 256	11x	8.53	34952.53

13.5 IWDG Registers

13.5.1 IWDG Register Overview

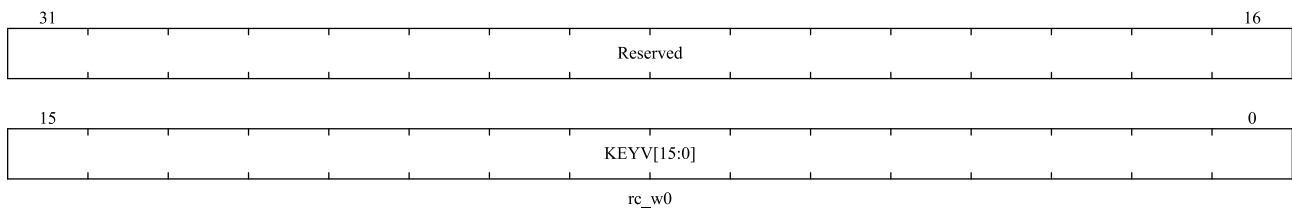
Table 13-2 IWDG Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0														
0x00	IWDG_KEY	Reserved																KEYV[15:0]																													
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	IWDG_PREDIV	Reserved																								PD[2:0]																					
	Reset value																									0	0	0																			
0x08	IWDG_RELV	Reserved																REL[11:0]																													
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x0C	IWDG_STS	Reserved																								CRVU	PVU																				
	Reset value																									0	0																				

13.5.2 IWDG Key Register (IWDG_KEY)

Address offset: 0x00

Reset value: 0x00000000

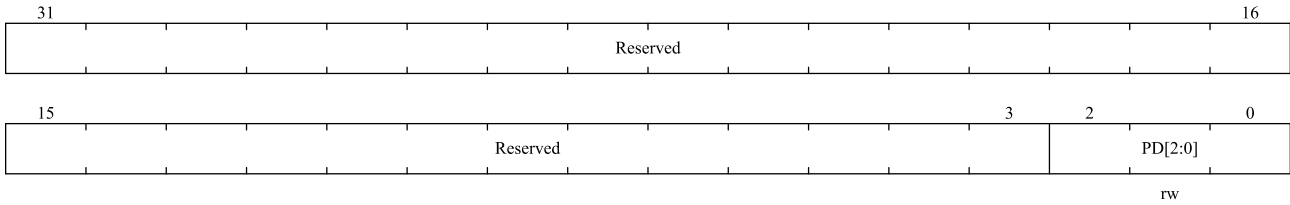


Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained.
15:0	KEYV[15:0]	Key value register: only certain value will serve particular function 0xCCCC: Start watch dog counter, does not have any effect if hardware watchdog is enable, (if hardware watchdog is selected, it is not limited by this command word) 0xAAAA: Reload counter with REL value in IWDG_RELV register to prevent reset. 0x5555: Disable write protection of IWDG_PREDIV and IWDG_RELV register

13.5.3 IWDG Pre-scaler Register (IWDG_PREDIV)

Address offset: 0x04

Reset value: 0x00000000

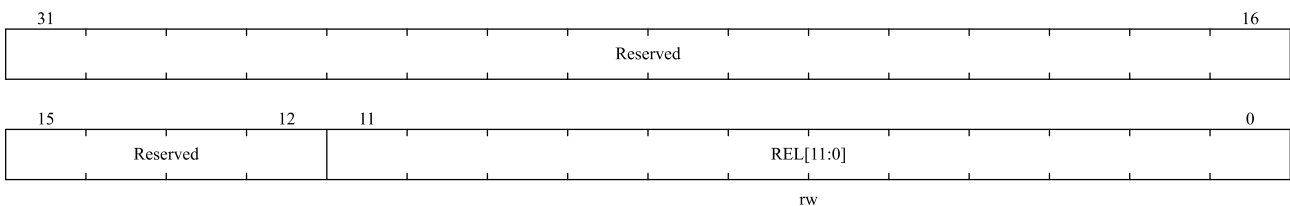


Bit Field	Name	Description
31:3	Reserved	Reserved, the reset value must be maintained.
2:0	PD[2:0]	<p>Pre-frequency division factor</p> <p>Pre-scaler divider: with write access protection when IWDG_KEY.KEYV[15:0] is not 0x5555. The IWDG_STS.PVU bit must be 0 otherwise PD [2:0] value cannot be changed. Divide number is as follow:</p> <p>000: divider /4 001: divider /8 010: divider /16 011: divider /32 100: divider /64 101: divider /128 Other : divider /256</p> <p><i>Note: Reading this register will return the pre-divided value from the V_{DD} voltage domain. If a write operation is in progress, the read-back value may be invalid. Therefore, the read value is valid only when the IWDG_STS.PVU bit is '0'.</i></p>

13.5.4 IWDG Reload Register (IWDG_RELV)

Address offset: 0x08

Reset value: 0x0000FFF



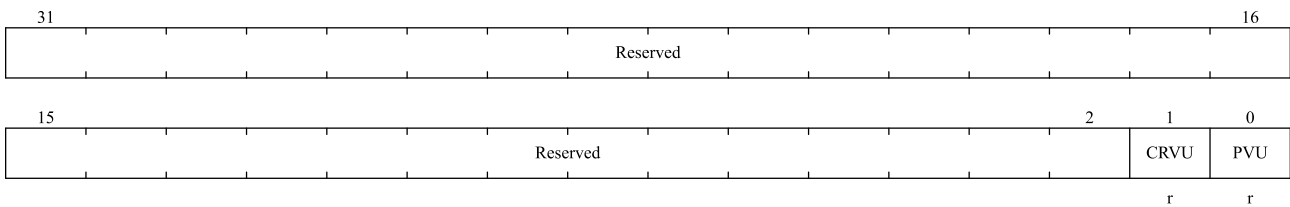
Bit Field	Name	Description
31:12	Reserved	Reserved, the reset value must be maintained.
11:0	REL[11:0]	<p>Watchdog counter reload value.</p> <p>With write protection. Defines the reload value of the watchdog counter, which is loaded to the counter every time 0xAAAA is written to IWDG_KEY.KEYV[15:0] bits. The counter then starts to count down from this value. The watchdog timeout period can be calculated from this reloading value and the clock pre-scaler value, refer to Table 13-1.</p> <p>This register can only be modified when the IWDG_STS.CRVU bit is '0'.</p> <p><i>Note: Reading this register will return the reload value from the V_{DD} voltage domain. If a write operation is in progress, the read-back value may be invalid. Therefore, the read value is valid only</i></p>

Bit Field	Name	Description
		when the IWDG_STS.CRVU bit is '0'.

13.5.5 IWDG Status Register (IWDG_STS)

Address offset: 0x0C

Reset value: 0x00000000



Bit Field	Name	Description
31:2	Reserved	Reserved, the reset value must be maintained.
1	CRVU	Watchdog reload value update Reload Value Update: this bit indicates an update of reload value is ongoing. Set by hardware and clear by hardware. Software can only try to change IWDG_RELV.REL[11:0] value when IWDG_KEY.KEYV[15:0] bits' value is 0x5555 and this bit is 0.
0	PVU	Watchdog pre-scaler value update Pre-scaler Value Update: this bit indicates an update of pre-scaler value is ongoing. Set by hardware and clear by hardware. Software can only try to change IWDG_PREDIV.PD[2:0] value when IWDG_KEY.KEYV[15:0] bits' value is 0x5555 and this bit is 0.

14 Window Watchdog (WWDG)

14.1 Introduction

The clock of the window watchdog (WWDG) is obtained by dividing the APB1 clock frequency by 4096, and it is used to detect abnormal program operation through the configuration of the time window. Therefore, WWDG is suitable for precise timing, and is often used to monitor software failures caused by external disturbances or unforeseen logic conditions which that cause an application to deviate from its normal operating sequence. A system reset is generated when the WWDG down counter value is refreshed before reaching the window register value or after the WWDG_CTRL.T6 bit becomes 0.

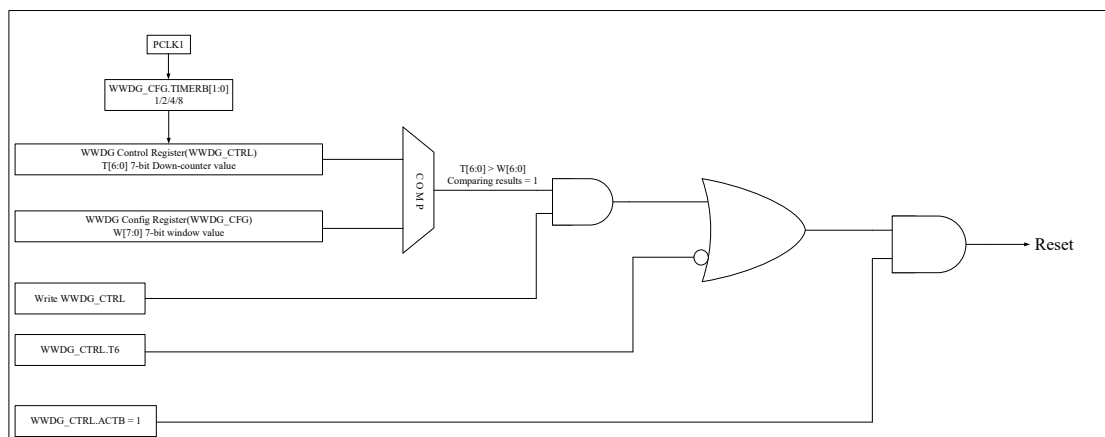
14.2 Main Features

- The clock of WWDG is obtained by dividing the APB1 clock frequency
- Programmable free-running down-counter
- After WWDG is enabled, a reset occurs under the following conditions
 - When the down-counter is less than 0x40
 - When the down-counter is reloaded outside the window
- If the watchdog is enabled and interrupts are allowed, an early wake up interrupt (EWI) occurs when the down-counter equals 0x40, which can be used to reload the counter to avoid WWDG reset

14.3 Function Description

If the watchdog is activated (the WWDG_CTRL.ACTB bit), when the 7-bit (WWDG_CTRL.T[6:0]) down-counter reaches 0x3F (WWDG_CTRL.T6 bit is cleared), or the software reloads the counter when the counter value is greater than the value of the window register, a system reset will be generated. In order to avoid system reset, the software must periodically refresh the counter value in the window during normal operation.

Figure 14-1 Watchdog Block Diagram



Set the WWDG_CTRL.ACTB bit to enable the watchdog, and thereafter, the WWDG will remain on until reset occurs. The 7-bit down-counter runs independently, and the counter keeps counting down whether WWDG is enabled or not. Therefore, before enabling the watchdog, you need to set WWDG_CTRL.T[6] bit to 1, preventing immediate reset after

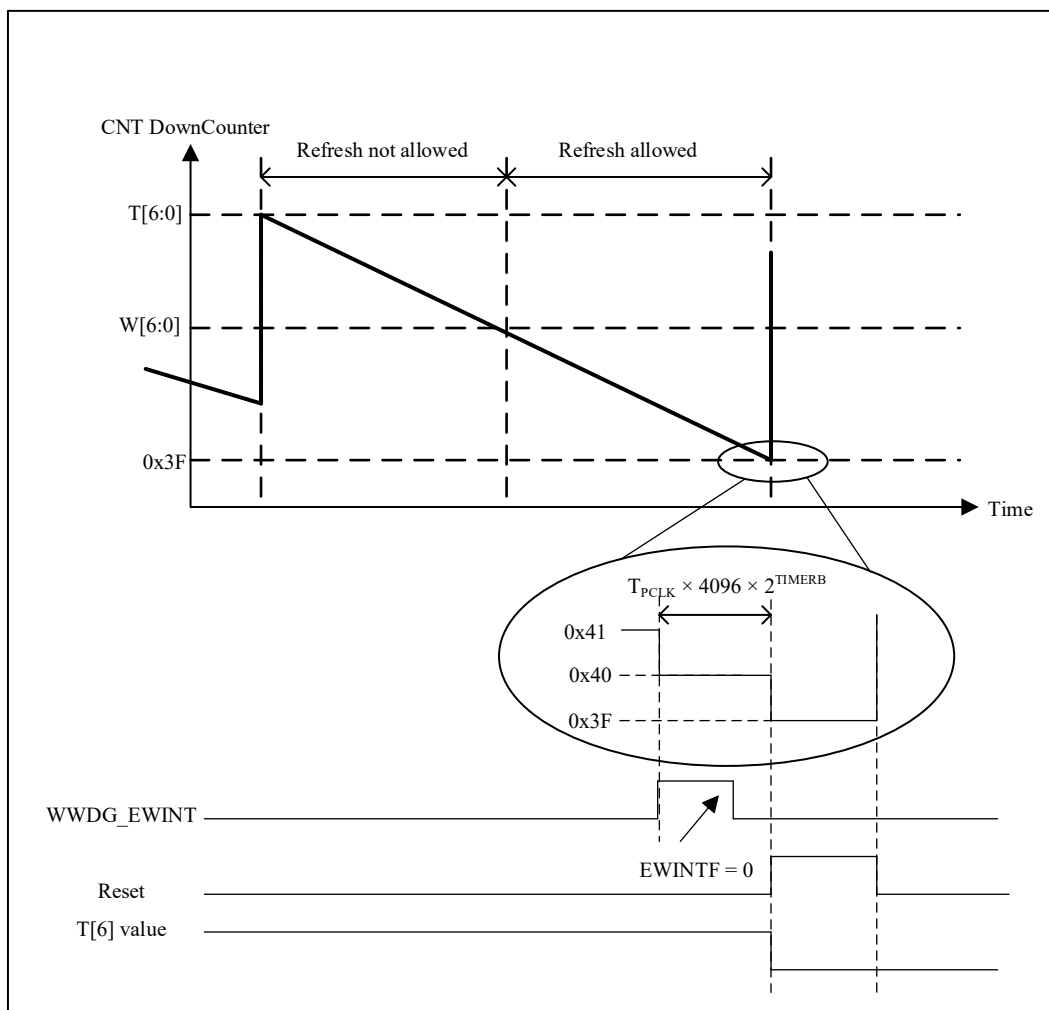
enable. The pre-scaler value set by the clock APB1 and WWDG_CFG.TIMERB[1:0] bits determine the down speed of the counter. WWDG_CFG.W[6:0] bits set the upper limit of the window.

When the down-counter value is refreshed before reaching the window register value or after WWDG_CTRL.T6 bit becomes 0, a system reset will be generated. Figure 14-2 describes the operating process of the window register.

Set the WWDG_CFG.EWINT bit to enable early wake-up interrupt. When the down counter reaches 0x40, an interrupt will be generated. You can analyze the cause of software failure or save important data in the corresponding interrupt service routine (ISR), and reload the counter to prevent WWDG reset. Write '0' to the WWDG_STS.EWINTF bit to clear the interrupt.

14.4 Timing for Refresh Watchdog and Interrupt Generation

Figure 14-2 Refresh Window and Interrupt Timing of WWDG



Watchdog refreshing window is between WWDG_CFG.W[6:0] value (maximum value 0x7F) and 0x3F, refresh outside this window will generates reset request to MCU. Counter count down from 0x7F to 0x3F using scaled APB1 clock, the maximum counting time and minimum counting time is shown in Table 14-1 (assuming APB1 clock 48 MHz) with calculate equation:

$$T_{WWDG} = T_{PCLK1} \times 4096 \times 2^{TIMERB} \times (T[5:0] + 1)$$

In which:

T_{WWDG} : WWDG timeout

T_{PCLK1} : APB1 clock interval in ms

Minimum-maximum timeout value at $PCLK1 = 48MHz$

Table 14-1 Maximum and Minimum Counting Time of WWDG

TIMERB	Min Timeout Value(μs)	Max Timeout Value(ms)
	T[5:0] = 0x00	T[5:0] = 0x3F
0	85.33	5.46
1	170.67	10.92
2	341.33	21.85
3	682.67	43.68

14.5 Debug Mode

In debug mode (Cortex[®]-M0 core stops), WWDG counter will either continue to operate normally or stops, depending on DBG_CTRL.WWDG_STOP bit in debug module. If this bit is set to '1', the counter stops. The counter operates normally when the bit is '0'. For details, refer to Section 3.3.2 Peripheral Debug Support.

14.6 User Interface

14.6.1 WWDG Configuration Process

1. Configure RCC_APB1PCLKEN.WWDGEN[11] bit to enable the clock of WWDG module;
2. Software setting WWDG_CFG.TIMERB[8:7] bits to configure pre-scale factor for WWDG.
3. Software configure WWDG_CTRL.T[6:0] bits, setting starting value of counter. Need to set WWDG_CTRL.T[6] bit to 1, preventing immediate reset after enable.
4. Configure WWDG_CFG.W[6:0] bits to configure upper boundary window value;
5. Setting WWDG_CTRL.ACTB[7] bit to enable WWDG;
6. Software operates WWDG_STS.EWINTF[0] bit to clear wake-up interrupt flag;
7. Configure WWDG_CFG.EWINT[9] bit to enable early wake-up interrupt.

14.7 WWDG Registers

14.7.1 WWDG Register Overview

Table 14-2 WWDG Register Overview

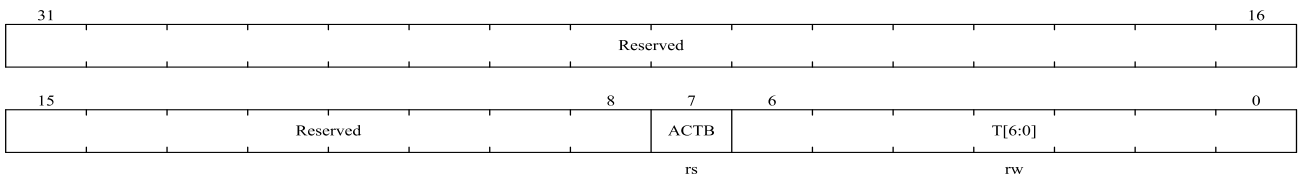
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
000h	WWDG_CTRL	Reserved																							ACTB	T[6:0]							
	Reset Value																									0	1	1	1	1	1	1	1

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
004h	WWDG_CFG	Reserved																						EWINT	TIMERB [1:0]		W[6:0]						
	Reset Value																							0	0	0	1	1	1	1	1	1	1
008h	WWDG_STS	Reserved																										EWINT					
	Reset Value																											0					

14.7.2 WWDG Control Register (WWDG_CTRL)

Address offset : 0x00

Reset value : 0x0000007F

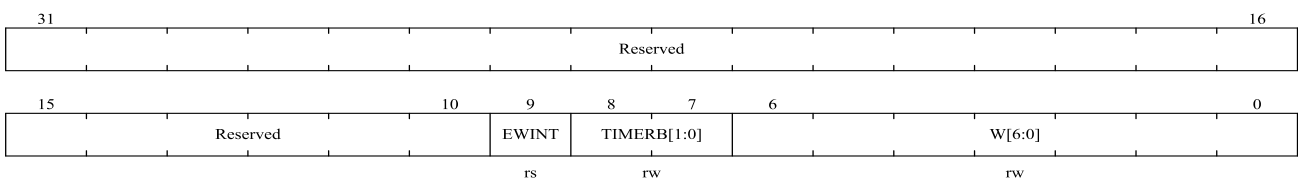


Bit field	Name	Description
31:8	Reserved	Reserved, the reset value must be maintained.
7	ACTB	Activation bit When ACTB=1, the watchdog can generate a reset. This bit is set by software and only cleared by hardware after a reset. When ACTB = 1, the watchdog can generate a reset. 0: Disable watchdog 1: Enable watchdog
6:0	T[6:0]	These bits contain the value of the watchdog counter. It is decremented every $(4096 \times 2^{\text{TIMERB}})$ PCLK1 cycles. A reset is produced when it rolls over from 0x40 to 0x3F (T6 becomes cleared).

14.7.3 WWDG Config Register (WWDG_CFG)

Address offset: 0x04

Reset value : 0x0000007F



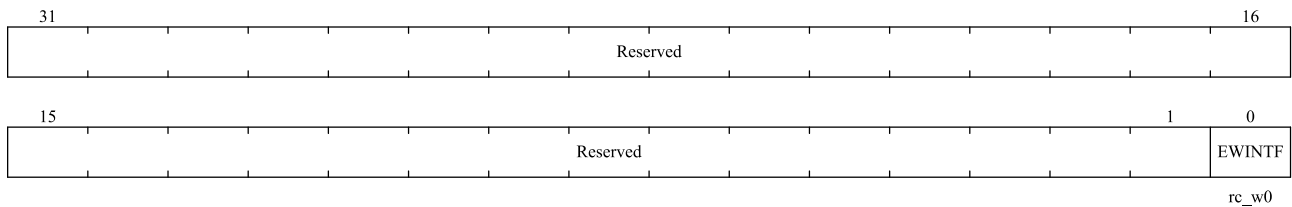
Bit Field	Name	Description
31:10	Reserved	Reserved, the reset value must be maintained.
9	EWINT	Early wake-up interrupt When set, an interrupt occurs whenever the counter reaches the value 0x40. This interrupt is only cleared by hardware after a reset.

Bit Field	Name	Description
8:7	TIMERB[1:0]	Timer base. The time base of the pre-scaler 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	W[6:0]	7-bit window value These bits contain the window value to be compared to the down counter.

14.7.4 WWDG Status Register (WWDG_STS)

Address offset: 0x08

Reset value : 0x0000



Bit Field	Name	Description
31:1	Reserved	Reserved, the reset value must be maintained.
0	EWINTF	Early wake-up interrupt flag This bit is set by hardware when the counter has reached the value 0x40. It must be cleared by software by writing '0'. A write of '1' has no effect. This bit is also set if the interrupt is not enabled.

15 Analog to Digital Converter (ADC)

15.1 Introduction

The 12-bit ADC is a high-speed analog-to-digital converter using successive approximation. It has 19 channels, measuring 16 external and 3 internal signal sources. The A/D conversion of each channel has four execution modes: single, continuous, scan or discontinuous. ADC measurements are stored (left-aligned/ right-aligned) in 16-bit data registers. The application can detect that the input voltage is within user-defined high/low thresholds by analog watchdog and the maximum frequency of the input clock to the ADC is 18MHz.

15.2 Main Features

- Supports 1 ADC, supports 16 single-ended inputs, measuring 16 external and 3 internal signal sources
- 12-bit resolution, the highest sampling rate is 1MSPS
- ADC clock source is divided into operating clock source, sampling clock source and timing clock source
 - AHB_CLK can be configured as the operating clock source, up to 48MHz
 - PLL can be configured as a sampling clock source, up to 18MHz, supports prescaler 1,2,3,4,6,8,10,12,16,32,64,128,256
 - The AHB_CLK can be configured as the sampling clock source, up to 18MHz, supports prescaler 1,2,3,4,6,8,10,12,16,32
 - The timing clock is used for internal timing functions and the frequency must be configured to 1MHz
- Supports timer trigger ADC sampling
- Interrupts when conversion ends, injection conversion ends, and analog watchdog events occur
- Single and continuous conversion modes
- Automatic scan mode from channel 0 to channel N
- Data alignment with embedded data consistency
- Channel_wise programmable sampling time
- Both regular conversions and injection conversions have external triggering options
- Discontinuous mode
- ADC power supply range : 2.4V to 5.5V
- ADC input range: $0 \leq V_{IN} \leq V_{DDA}$
- ADC can use DMA operations, and DMA requests are generated during regular channel conversion.

15.3 Function Description

Figure 15-1 is a functional block diagram of an ADC.

Figure 15-1 Block Diagram of ADC

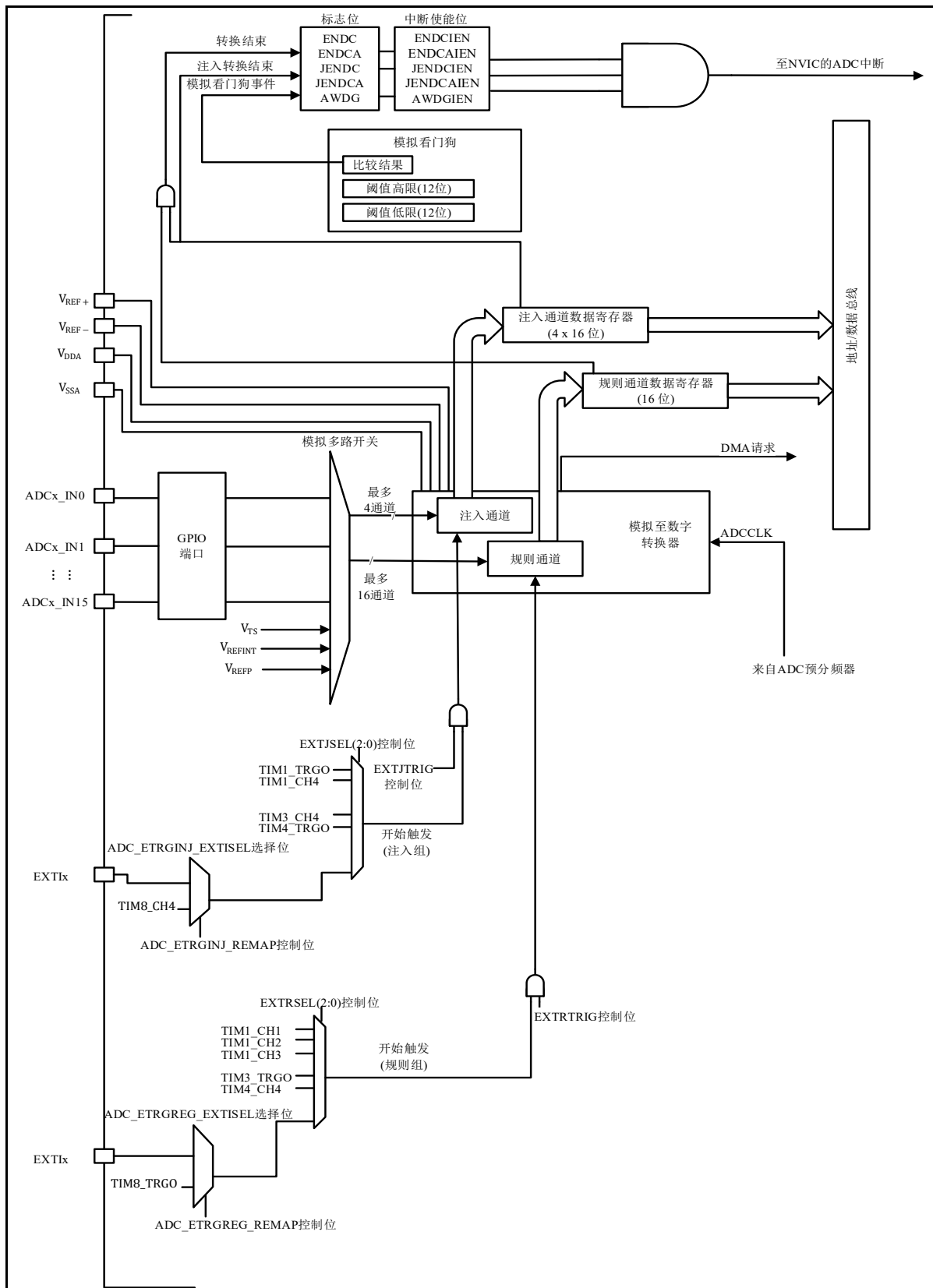


Table 15-1 ADC Pins

Name	Signal Types	Annotations
------	--------------	-------------

V _{REF+}	Input, analog reference positive	Positive voltage reference used by ADC, $2.4V \leq V_{REF+} \leq V_{DDA}$
V _{DDA} ⁽¹⁾	Input, analog power supply	Equivalent to V _{DD} analog power supply and: $1.8V \leq V_{DDA} \leq V_{DD}(5.5V)$
V _{SSA} ⁽¹⁾	Input, analog power supply ground	Equivalent to V _{SS} Analog power supply ground
ADCx_IN[15:0]	Analog input signal	16 analog external input channels

Note: ⁽¹⁾ V_{DDA} and V_{SSA} should be separately connected to V_{DD} and V_{SS}.

15.3.1 ADC Clock

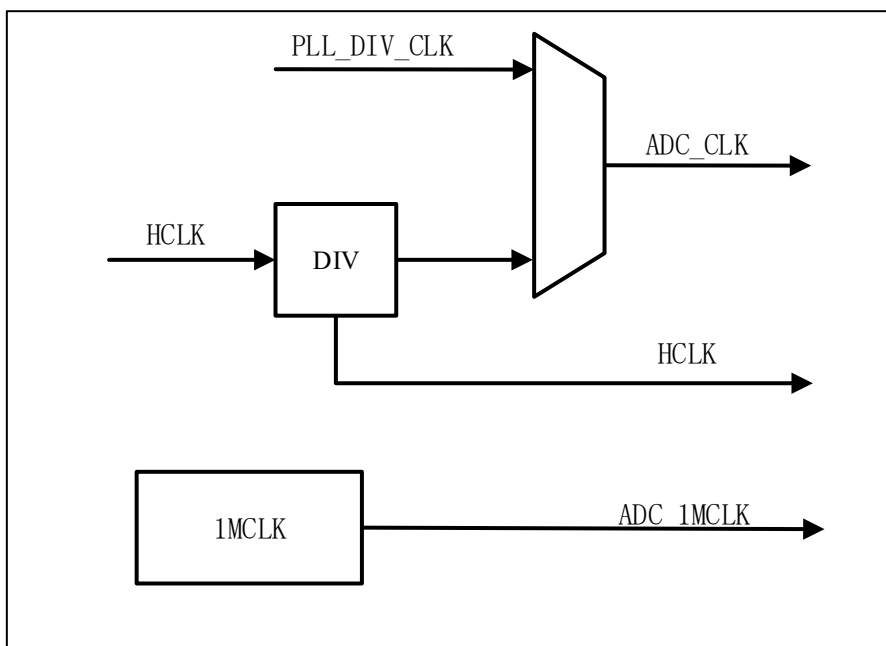
An ADC requires three clocks, HCLK, ADC_CLK and ADC_1MCLK.

- HCLK is used for the register access.
- ADC_CLK is the operating clock of ADC. ADC_CLK has two sources (divided from HCLK or PLL). The HCLK divided clock and system clock are synchronous clock, while the PLL divided clock and system clock are asynchronous clock. The advantage of using a synchronous clock is that there is no uncertainty when triggering the ADC to respond to the trigger. The advantage of using PLL's divider clock is that the ADC's operating clock can be handled independently without affecting other modules attached to the HCLK
- ADC_1MCLK is used for internal timing function, configured in RCC, the frequency must be configured to 1MHz

Notes:

- (1) When configuring PLL as a clock source, the maximum frequency can reach 18MHz, supporting division factors of 1,2,3,4,6,8,10,12,16,32,64,128,256 .
- (2) The AHB_CLK prescaler can be configured as a operating clock up to 18MHz. The AHB_CLK prescaler can be 1,2,3,4,6,8,10,12,16,32.
- (3) When switching the ADC clock source to ADC_1M, it is necessary to ensure that the HSI clock is enabled.

Figure 15-2 ADC Clock



15.3.2 ADC Switch Control

User can proceed to the next step only after the power-up process is complete. User can check if the power-up is complete by accessing the ADC_CTRL3.RDY bit.

User can set the ADC_CTRL2.ON bit to turn on the ADC. When the ADC_CTRL2.ON bit is set for the first time, it wakes up the ADC from the power-off state. After a power-on delay of ADC (t_{STAB}), the conversion begins when the ADC_CTRL2.ON bit is set again.

The conversion can be stopped by clearing the ADC_CTRL2.ON bit, then the ADC in power-off mode. In this mode, the ADC consumes almost no power consumption (just a few μA). Power-down can be checked by accessing the ADC_CTRL3.PDRDY bit.

When the ADC is disabled, the default mode is power-down.

15.3.3 Channel Selection

Each channel can be configured as a regular sequence and an injection sequence.

The injection sequence consists of multiple conversions, up to 4. The ADC_JSEQ register specifies the injection channel and the conversion sequence of the injection channel. The ADC_JSEQ.JLEN[1:0] bits specified injection sequence length.

The regular sequence consists of multiple conversions, up to 16. The ADC_RSEQx registers specify the regular channels and the conversion sequence of the regular channels. The ADC_RSEQ1.LEN[3:0] bits specified regular channel sequence length.

Note: During conversion, changes to the ADC_RSEQx or ADC_JSEQ registers are prohibited; the ADC_RSEQx or ADC_JSEQ registers can only be changed when the ADC is idle.

15.3.4 Internal Channel

- The temperature sensor connects to channel ADC_IN16
- The internal voltage reference V_{REFINT} is connected to the channel ADC_IN17
- Internal voltage reference V_{REFP} is connected to channel ADC_IN18
- ADC_IN6 can be connected to internal OPAMP_OUT besides IO

Internal channels can be converted by injection or regular channels.

Note: V_{REFINT} needs to be enabled by configuring ADC_CTRL3.VREFEN. After enabling, it is necessary to confirm that V_{REFINT} is ready by ADC_CTRL3.VREFRDY.

15.3.5 Single Conversion Mode

The ADC can enter the single conversion mode by configuring ADC_CTRL2.CTU to 0. In this mode, external triggering (for regular channels or injected channels) or setting ADC_CTRL2.ON=1 (for regular channels only) can enable the ADC to start conversion, and the ADC only performs one conversion.

After the conversion starts, when an injected channel conversion is completed, the injected channel conversion end flag (ADC_STS.JENDC) will be set to 1. If the injected channel conversion end interrupt enable (ADC_CTRL1.JENDCIEN) bit is set to 1, an interrupt will be generated at this time, and the converted data will be stored in the ADC_JDATx register.

After the conversion starts, when a regular channel conversion is completed, the regular channel conversion end flag(ADC_STS.ENDC) will be set to 1. If the regular channel conversion end interrupt enable (ADC_CTRL1.ENDCIEN)

bit is set to 1, an interrupt will be generated at this time, and the converted data will be stored in the ADC_DAT register.

After single conversion, the ADC stops.

15.3.6 Continuous Conversion Mode

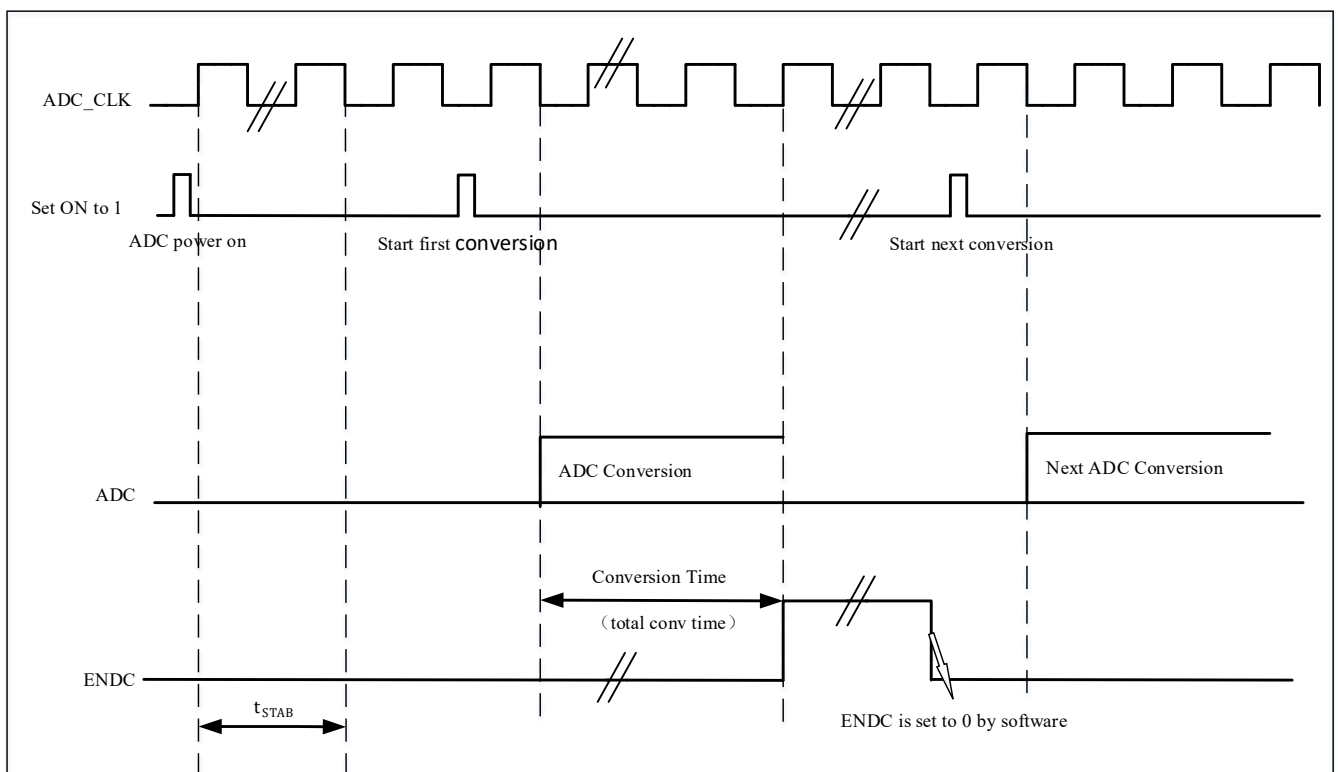
The ADC can enter the continuous conversion mode by configuring ADC_CTRL2.CTU to 1. In this mode, external triggering or setting ADC_CTRL2.ON to 1 can enable the ADC to start conversion, and the ADC will continuously convert the selected channel. Continuous mode is only valid for regular channels, not for injected channels.

After the conversion starts, when a regular channel conversion is completed, the regular channel end of conversion flag bit (ADC_STS.ENDC) will be set to 1. If the regular channel conversion end interrupt enable bit (ADC_CTRL1.ENDCIEN) is set to 1 at this time, an interrupt will be generated. The converted data will be stored in the ADC_DAT register.

15.3.7 Timing Diagram

When ADC_CTRL2.ON is set to 1 for the first time, the ADC is powered on. After the ADC is powered on, the ADC needs a certain time (t_{STAB}) to ensure its stability. After the ADC is stabilized, At this time, set ADC_CTRL2.ON to 1 again through software to start the conversion. The end of conversion flag bit will be set to 1 after the conversion is completed.

Figure 15-3 Timing Diagram



15.3.8 Analog Watchdog

The analog watchdog can be enabled on the regular channel by setting ADC_CTRL1.AWDGERCH to 1, or the analog watchdog on the injection channel can be enabled by setting ADC_CTRL1.AWDGEJCH to 1. The high threshold of the analog watchdog can be set by configuring ADC_WDGHIGH.HTH, and the low threshold of the analog watchdog can be set by configuring ADC_WDGLow.LTH. The threshold of the analog watchdog has nothing to do with the way of data alignment, because the comparison of the ADC's conversion value with the threshold is done before the alignment. When

the value of ADC analog conversion is higher than the high threshold of the analog watchdog or lower than the low threshold of the analog watchdog, the analog watchdog flag (ADC_STS.AWDG) will be set to 1. If ADC_CTRL1.AWDGIEN has been configured to 1, an interrupt will be generated at this time. The analog watchdog can be controlled for one or more channels by configuring ADC_CTRL1.AWSGSGLEN and ADC_CTRL1.AWDGCH[4:0].

Table 15-2 Analog Watchdog Channel Selection

Channel	ADC_CTRL1 Register Control Bit		
	AWDGSLEN	AWDGERCH	AWDGEJCH
There is none	Any value	0	0
All injection channels	0	0	1
All regular channels	0	1	0
All injection and regular channels	0	1	1
A single injection channel	1	0	1
A single regulars of the channel	1	1	0
A single injection or regular channel	1	1	1

15.3.9 Scan Mode

By configuring ADC_CTRL1.SCAMD to 1, the scan conversion mode can be turned on, and the conversion sequence can be selected by configuring the four registers ADC_RSEQ1, ADC_RSEQ2, ADC_RSEQ3, ADC_JSEQ, the ADC will scan and convert all the selected channels. After the conversion is started, the channels will be converted one by one. If ADC_CTRL2.CTU is 1 at this time, the conversion will be restarted from the first channel of the conversion sequence after the conversion of all selected channels is completed. The DMA function can be turned on by setting ADC_CTRL2.ENDMA to 1, and the DMA will transfer the data to the SRAM after the regular channel conversion is completed.

15.3.10 Injection Channel Management

Automatic injection

If ADC_CTRL1.AUTOJC bit is set, then the injected channels are automatically converted following the regular channels mentioned by ADC_RSEQx and ADC_JSEQx. A single trigger can convert up to 16+ 4 channels. Setting ADC_CTRL2.CTU the conversion sequence will be converted continuously.

When this function is turned on, the external trigger of the injection channel needs to be turned off.

This function cannot be used with the discontinuous mode at the same time.

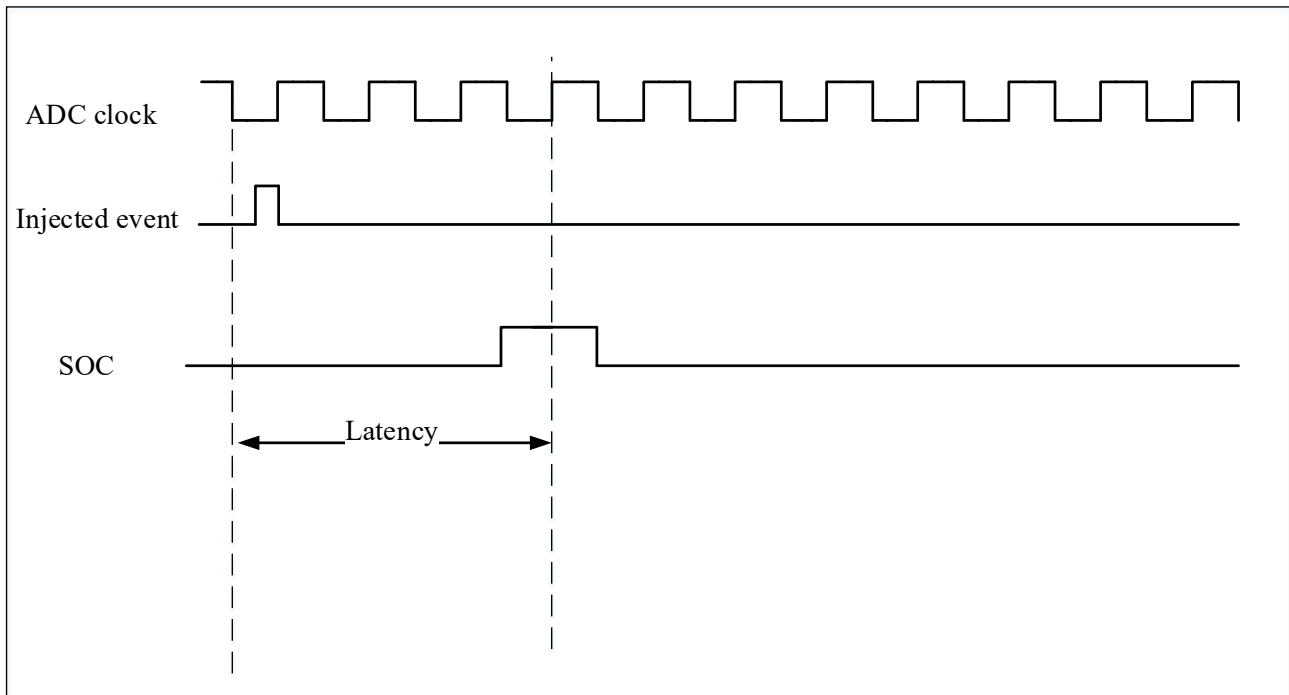
When the ADC clock prescale factor is 2, there is a delay of two ADC clock intervals when the conversion sequence changes from regular to injection or injection to regular. When the ADC clock prescale factor is 4 to 8, there is a delay of one ADC clock intervals when the conversion sequence changes from regular to injection or injection to regular.

Triggered injection

Set ADC_CTRL1.AUTOJC to 0 and ADC_CTRL1.SCAMD to 1 to enable the trigger injection function. In this function, the conversion on regular channels either by setting the ADC_CTRL2.ON or by external trigger in continuous mode. When the regular channel is converted, if an external injection trigger is generated, the current conversion will be suspended, and the injected sequence channel will start conversion. When the injected sequence channel conversion is completed, the interrupted conversion of regular sequence channel will be resumed. If a regular event is generated during the injected conversion, the regular sequence channel will start conversion after the injection sequence channel conversion is completed.

When using this feature, the time interval between injected channel triggers are fired needs to be greater than the time it takes for the injected sequence to complete the transition.

Figure 15-4 Injection Conversion Delay



Note: For the maximum delay value, please refer to the electrical characteristics section in the data manual.

15.3.11 Discontinuous Mode

Regular channels

Configure ADC_CTRL1.DREGCH to 1 to enable the discontinuous mode on the regular channel, obtain the regular sequence by configuring ADC_RSEQ1, ADC_RSEQ2, ADC_RSEQ3, and configure ADC_CTRL1.DCTU[2:0] to control the conversion of n channels each time a trigger signal is generated. The total sequence length is defined by ADC_RSEQ1.LEN[3:0].

When the trigger signal is generated, it will convert n channels of the regular sequence and then stop, until the next trigger signal is generated, it will continue to convert n channels from the point where the previous conversion stopped, until all channels of the regular sequence are converted (If the last trigger occurs and the remaining channels in the conversion sequence are less than n, only the remaining channels will be converted and the conversion will be stopped), and the end of conversion flag bit will also be set to 1. When the conversion of all channels in the conversion sequence is completed, when the next trigger signal occurs, the conversion starts from the first channel of the regular sequence again.

Injected channels

Configure ADC_CTRL1.DJCH to 1 to enable the discontinuous mode on the injection channel, obtain the injection sequence by configuring ADC_JSEQ.

When the trigger signal is generated, it will convert 1 channel of the injection sequence and then stop. Until the next trigger signal is generated, it will continue to convert 1 channel from the point where the previous conversion stopped until all channels of the injection sequence are converted, and the end of conversion flag bit will also be set to 1. When the

conversion of all channels in the conversion sequence is completed, when the next trigger signal occurs, the conversion starts from the first channel of the injection sequence again.

Only one of injection conversion and regular conversion can be set to discontinuous mode at the same time, and the automatic injection function and discontinuous mode cannot be set at the same time.

15.4 Data Aligned

There are two alignment methods for data storage after conversion: left-aligned and right-aligned. The alignment can be set by the ADC_CTRL2.ALIG bit. ADC_CTRL2.ALIG = 0 is right-aligned, as shown in Table 15-3, ADC_CTRL2.ALIG = 1 is left-aligned, as shown in Table 15-4.

For injected sequence, the SYM bit is the extended sign value, and the data stored in the register is the conversion result subtract the user-defined offset in the ADC_JOFFSETx register, so the result can be a negative value; for regular sequence, there is no need to subtract offset value.

Table 15-3 Right-aligned Data

The Injection sequence

SYM	SYM	SYM	SYM	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
-----	-----	-----	-----	-----	-----	----	----	----	----	----	----	----	----	----	----

The regular sequence

0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
---	---	---	---	-----	-----	----	----	----	----	----	----	----	----	----	----

Table 15-4 Left-aligned Data

Injection sequence

SYM	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0
-----	-----	-----	----	----	----	----	----	----	----	----	----	----	---	---	---

The regular sequence

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0
-----	-----	----	----	----	----	----	----	----	----	----	----	---	---	---	---

15.5 Programmable Channel Sampling Time

Specify the number of sampling cycles of ADC in ADC_SAMPT1.SAMPx[2:0] and ADC_SAMPT2.SAMPx[2:0], and then the ADC samples the input voltage in the specified sampling cycle. For different channels, User can select different sampling time. The total conversion time is calculated as follows:

$$T_{CONV} = \text{Sampling time} + 12 \text{ cycles}$$

Example:

ADCCLK=16MHz, the sampling time is 6 cycles and resolution is 12bit, the total conversion time is "6 + 12" ADCCLK Cycles, that is:

$$T_{CONV} = 6 + 12 = 18 \text{ cycle} = 1.125\mu\text{s}$$

15.6 Externally Triggered Conversion

For the regular sequence, software sets the ADC_CTRL2.EXTRTRIG bit to 1, then the regular channel can use the rising edge of the external event to trigger the start conversion, and then the software sets the ADC_CTRL2.EXTRSEL[2:0] bits to select the external trigger source of the regular sequence. The external trigger source selection is shown in the table below. If user select EXTI line 0~15 or TIM8_TRGO as the external trigger source, user can set the AFIO_CFG.ADC_ETRR and AFIO_CFG.EXTI_ETRR[3:0] bits to implement; if you select SWSTRRCH as the external trigger source, you can start the regular channel conversion by setting ADC_CTRL2.SWSTRRCH to 1.

Table 15-5 ADC Is Used for External Triggering of Regular Channels

EXTRSEL[2:0]	Trigger source	Type
000	TIM1_CC1 event	Internal signal from the on-chip timer
001	TIM1_CC2 event	
010	TIM1_CC3 event	
011	N/A	
100	TIM3_TRGO event	
101	TIM4_CC4 event	
110	EXTI line 0~15/TIM8_TRGO event ⁽¹⁾	External pin/internal signal from on-chip timer
111	SWSTRRCH	Software control bit

For the injected sequence, the software sets the ADC_CTRL2.EXTJTRIG bit to 1, then the injected channel can use the rising edge of the external event to trigger the conversion, and the software sets the ADC_CTRL2.EXTJSEL[2:0] bits to select the external trigger source of the injected sequence. The external trigger source selection is shown in the table below. If user select EXTI line 0~15 or TIM8_CC4 as the external trigger source, user can set the AFIO_CFG.ADC_ETRI and AFIO_CFG.EXTI_ETRI[3:0] bits to implement; if user select SWSTRJCH as the external trigger source, you can start the injected channel conversion by setting ADC_CTRL2.SWSTRJCH to 1.

Table 15-6 ADC Is Used for External Triggering of Injection Channels

EXTJSEL[2:0]	Trigger Source	Type
000	TIM1_TRGO event	Internal signal from the on-chip timer
001	TIM1_CC4 event	
010	N/A	
011	N/A	
100	TIM3_CC4 event	
101	TIM4_TRGO event	
110	EXTI line 0~15/TIM8_CC4 event ⁽¹⁾	External pin/internal signal from on-chip timer
111	SWSTRJCH	Software control bit

Note: Injection triggers can interrupt conversion of the regular sequence.

15.7 DMA Requests

To avoid excessive data when converting multiple regular channels, resulting in errors in the regular channel conversion result stored in the ADC_DAT register, user can set the ADC_CTRL2.ENDDMA bit to 1 to use the DMA mode. When the ADC regular channel conversion ends, a DMA request is generated. After receiving the request, the DMA will transfer the

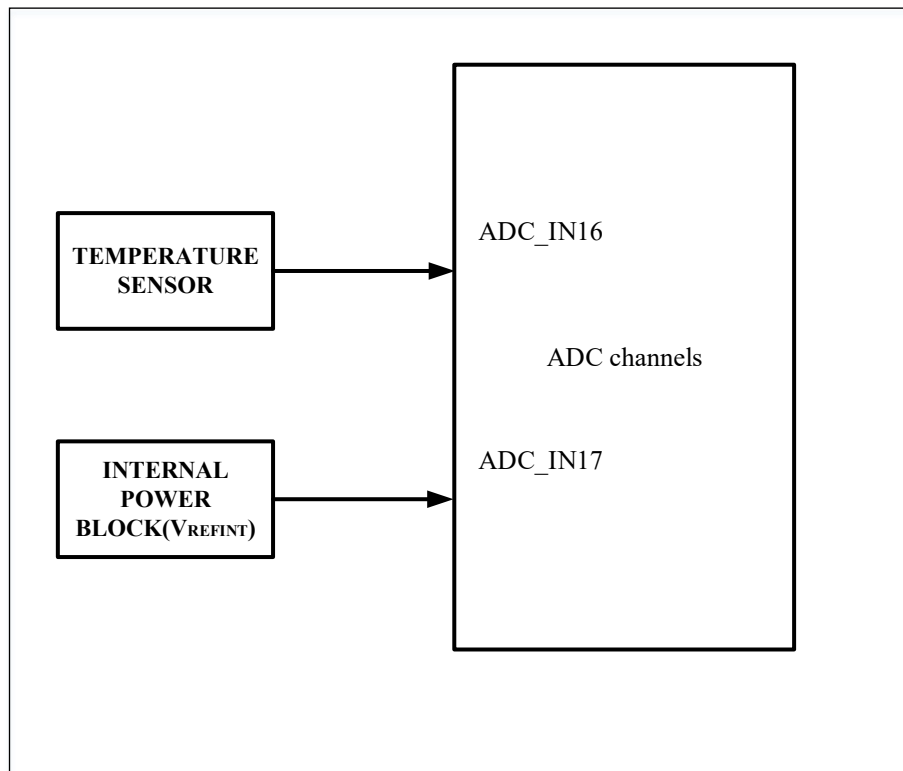
converted data from the ADC_DAT register to the destination address specified by the user.

15.8 Temperature Sensor

Set the ADC_CTRL2.TEMPEN bit to 1, enable the temperature sensor, and use the temperature sensor to detect the ambient temperature when the device is operating. The output voltage sampled by the temperature sensor is converted into a digital value by the ADC_IN16 channel. When the temperature sensor is operating, the recommended sampling time is 17.1us; when the temperature sensor is not operating, the ADC_CTRL2.TEMPEN bit can be cleared by software to turn off the temperature sensor to reduce power consumption. Figure 15-5 is a block diagram of a temperature sensor.

The output voltage of the temperature sensor changes linearly with temperature. Different chips will have different offsets in the temperature curve due to different production processes. Through testing, it is found that the maximum offset is 3°C. This characteristic makes the internal temperature sensor more suitable for detecting temperature changes. Not suitable for measuring absolute temperature. When accurate temperature measurement is required, an external temperature sensor should be used.

Figure 15-5 Temperature Sensor and V_{REFINT} Diagram of The Channel



15.8.1 Temperature Sensor Using Process

1. Configure the channel (ADC_IN16) and sampling time 17.1us
2. Set ADC_CTRL2.TEMPEN bit to 1 to enable temperature sensor
3. Set ADC_CTRL2.ON bit to 1 to start ADC conversion (or through external trigger)
4. Read the temperature data in the ADC data register, and calculate the temperature value by the following formula:

$$\text{Temperature (}^{\circ}\text{C)} = \{(V_{25} - V_{\text{SENSE}}) / \text{Avg_Slope}\} + 25$$

In which:

$V_{25} = V_{\text{SENSE}}$ at 25 degrees Celsius

Avg_Slope = temperature and V_{SENSE} Average slope of a curve (mV/°C or $\mu\text{V}/^\circ\text{C}$)

Refer to the values of V_{25} and Avg_Slope in the electrical characteristics chapter of the datasheet.

Note: There is a settling time before the sensor wakes up from the power-off mode to the correct output of V_{SENSE} ; there is also a settling time after the ADC is powered on, so in order to shorten the delay, the ADC_CTRL2.TEMPEN and ADC_CTRL2.ON bits should be set at the same time.

15.9 ADC Interrupt

ADC interrupts can be from an end of regular or injected sequence conversion, an analog watchdog event when input voltage exceeds the threshold, any end of regular or injected channel conversion. These interrupts have independent interrupt enable bits.

There are 2 status flags in the ADC_STS register: injected sequence channel conversion started (JSTR) and regular sequence channel conversion started (STR). But there are no interrupts associated with these two flags in the ADC.

Table 15-7 ADC Interrupt

Interrupt Event	Event Flags	Enable Control Bit
Regular sequence or injection conversion is complete	ENDC	ENDCIEN
Injection sequence conversion is complete	JENDC	JENDCIEN
Analog watchdog status bit is set	AWDG	AWDGIEN
Any regular channel interruption is enabled	ENDCA	ENDCAIEN
Any injection channel interruption is enabled	JENDCA	JENDCAIEN

15.10 ADC Registers

For abbreviations used in registers, refer to Section 1.1

All register operations must be performed in half word (16-bits) or one word (32-bits).

15.10.1 ADC Register Overview

Table 15-8 ADC Register Overview

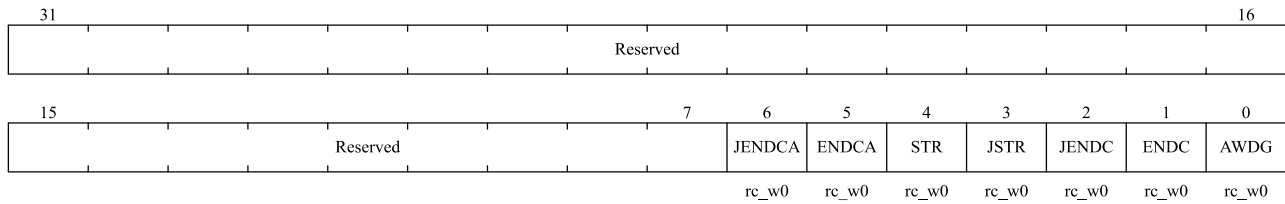
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
000h	ADC_STS	Reserved																								JENDCA	ENDCA	STR	JSTR	JENDC	ENDC	AWDG				
	Reset Value																									0	0	0	0	0	0					
004h	ADC_CTRL1	Reserved										AWDGERCH	AWDGEICH	Reserved						DCTU[2:0]		DJCH	DREGCH	AUTOIC	AWDGSGLN	SCANMD	JENDCIEN	AWDGIEN	ENDCIEN	AWDGCH[4:0]						
	Reset Value											0	0							0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
008h	ADC_CTRL2	Reserved										TEMPEN	SWSTRCH	SWSTRJCH	EXTRIG	EXTRSEL[2:0]			Reserved	EXTRIG	EXTJSEL[2:0]			ALIG	Reserved	ENDMA	Reserved				CTU	ON				
	Reset Value											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																
00Ch	ADC_SAMPT1	Reserved																				SAMP18[3:0]			SAMP17[3:0]			SAMP16[3:0]																					
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
010h	ADC_SAMPT2	SAMP15[3:0]				SAMP14[3:0]				SAMP13[3:0]				SAMP12[3:0]				SAMP11[3:0]				SAMP10[3:0]			SAMP9[3:0]			SAMP8[3:0]																					
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																
014h	ADC_SAMPT3	SAMP7[3:0]				SAMP6[3:0]				SAMP5[3:0]				SAMP4[3:0]				SAMP3[3:0]				SAMP2[3:0]			SAMP1[3:0]																								
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																
018h	ADC_JOFFSET1	Reserved																				OFFSETJCH1[11:0]																											
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
01Ch	ADC_JOFFSET2	Reserved																				OFFSETJCH2[11:0]																											
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
020h	ADC_JOFFSET3	Reserved																				OFFSETJCH3[11:0]																											
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
024h	ADC_JOFFSET4	Reserved																				OFFSETJCH4[11:0]																											
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
028h	ADC_WDGHIGH	Reserved																				HTH[11:0]																											
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
02Ch	ADC_WDGLow	Reserved																				LTH[11:0]																											
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
030h	ADC_RSEQ1	Reserved								LEN[3:0]				SEQ16[4:0]				SEQ15[4:0]				SEQ14[4:0]				SEQ13[4:0]																							
	Reset Value	0								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
034h	ADC_RSEQ2	Reserved	SEQ12[4:0]				SEQ11[4:0]				SEQ10[4:0]				SEQ9[4:0]				SEQ8[4:0]				SEQ7[4:0]																										
	Reset Value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																
038h	ADC_RSEQ3	Reserved	SEQ6[4:0]				SEQ5[4:0]				SEQ4[4:0]				SEQ3[4:0]				SEQ2[4:0]				SEQ1[4:0]																										
	Reset Value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																
03Ch	ADC_JSEQ	Reserved								JLEN[1:0]	JSEQ4[4:0]				JSEQ3[4:0]				JSEQ2[4:0]				JSEQ1[4:0]																										
	Reset Value	0								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0													
040h	ADC_JDAT1	Reserved																				JDAT1[15:0]																											
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
044h	ADC_JDAT2	Reserved																				JDAT2[15:0]																											
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
048h	ADC_JDAT3	Reserved																				JDAT3[15:0]																											
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
04Ch	ADC_JDAT4	Reserved																				JDAT4[15:0]																											
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
050h	ADC_DAT	Reserved																				DAT[15:0]																											
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
054h	ADC_CTRL3	Reserved																				JENDCAIEN	ENDCAIEN	Reserved	PDRDY	RDY	CKMOD	Reserved	VREFRDY	VREFEN	REFSEL																		
	Reset Value	0																				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.10.2 ADC Status Register (ADC_STS)

Address offset: 0x00

Reset value: 0x0000 0000

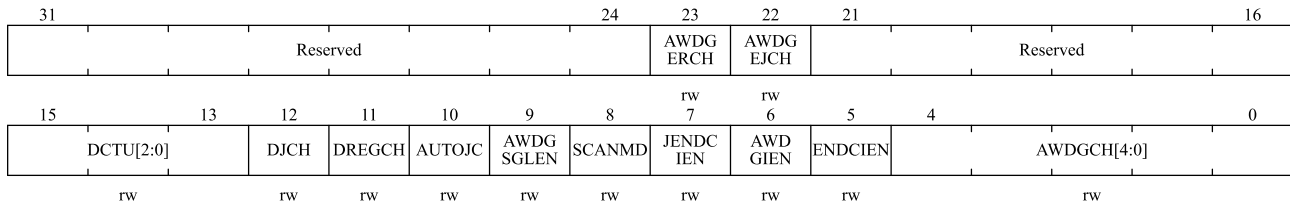


Bit Field	Name	Description
31:7	Reserved	Reserved, the reset value must be maintained
6	JENDCA	Any injected channel end of conversion flag This bit is set by hardware at the end of any injection channel conversion and cleared by software. 0: Conversion is not complete; 1: Conversion is complete.
5	ENDCA	Any regular channel end of conversion flag This bit is set by hardware at the end of any channel (regular or injection) conversion and cleared by software. 0: Conversion is not complete; 1: Conversion is complete.
4	STR	Regular channel start flag This bit is set by hardware at the start of regular channel conversion and cleared by software. 0: Regular channel conversion has not started. 1: Regular channel conversion has started.
3	JSTR	Injected channel start flag This bit is set by hardware at the start of the injection channel conversion and cleared by software. 0: Injection sequence channel conversion has not started. 1: Injection sequence channel conversion has started.
2	JENDC	Injected channel end of conversion This bit is set by hardware at the end of all injection sequence channel conversions and cleared by software 0: Conversion is not complete. 1: Conversion is complete.
1	ENDC	Conversion sequence channel end of conversion This bit is set by hardware at the end of all regular (or injection) sequence channel conversion and cleared by software 0: Conversion is not complete. 1: Conversion is complete.
0	AWDG	Analog watchdog flag This bit is set by hardware and cleared by software when converted voltage values are outside the range defined by the ADC_LTR and ADC_HTR registers 0: Analog watchdog event not occurs; 1: Analog watchdog event occurs.

15.10.3 ADC Control Register 1 (ADC_CTRL1)

Address offset: 0x04

Reset value: 0x0000 0000



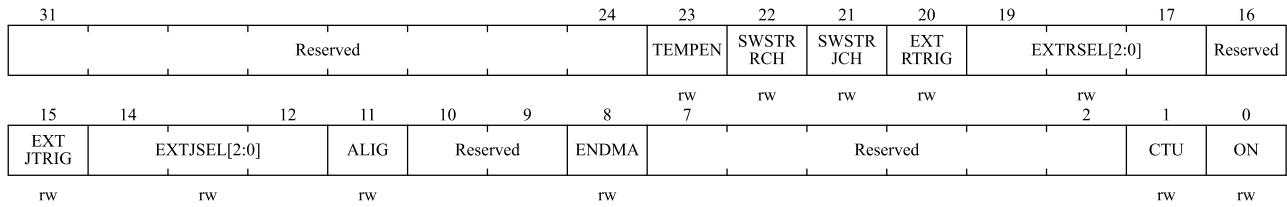
Bit Field	Name	Description
31:24	Reserved	Reserved,the reset value must be maintained
23	AWDGERCH	Analog watchdog enable on regular channels This bit is set and cleared by the software. 0: Disables analog watchdog on regular channel. 1: Use analog watchdog on regular channels.
22	AWDGEJCH	Analog watchdog enable on injected channels This bit is set and cleared by the software. 0: Disables analog watchdog on injection channel. 1: Use analog watchdog on the injection channel.
21:16	Reserved	Reserved,the reset value must be maintained
15:13	DCTU[2:0]	Discontinuous mode channel count The software uses these bits to define the number of channels for converting regulars after receiving an external trigger in intermittent mode 000: 1 channel 001: 2 channels ... 111: 8 channels
12	DJCH	Discontinuous mode on injected channels This bit is set and cleared by the software. It is used to turn on or off discontinuous mode on injected channels. 0: Disable discontinuous mode on injection sequence channel 1: Enable discontinuous mode on injection sequence channel
11	DREGCH	Discontinuous mode is on regular channels. This bit is set and cleared by the software. It is used to turn on or off discontinuous mode on regular channels. 0: Disable discontinuous mode on regular sequence channel 1: Enable discontinuous mode on regular sequence channel
10	AUTOJC	Automatic injected sequence conversion This bit is set and cleared by the software to enable or disable automatic injection sequence channel conversion after regular sequence channel conversion is complete 0: Disable automatic injection channel conversion. 1: Enable automatic injection channel conversion.

Bit Field	Name	Description
9	AWDGSLEN	<p>Enable the watchdog on a single channel in scan mode</p> <p>This bit is set and cleared by software to enable or disable analog watchdog functions on channels specified by ADC_CTRL1.AWDGCH[4:0]</p> <p>0: Use watchdog on all channels. 1: Use watchdog on single channel.</p>
8	SCANMD	<p>Scan mode</p> <p>This bit is set and cleared by the software to enable or disable scan mode. In scan mode, the conversion is made by ADC_RSEQx or the selected channel of the ADC_JSEQ register.</p> <p>0: Disable scan mode. 1: Enable scan mode.</p> <p><i>Note: If the ADC_CTRL1.ENDCIEN or ADC_CTRL1.JENDCIEN bits are set separately, ADC_STS.ENDC or ADC_STS.JENDC interrupts occur only after the last channel has been converted.</i></p>
7	JENDCIEN	<p>Interrupt enable for injected channels</p> <p>This bit is set and cleared by the software to disallow or allow interrupts after all injection channel conversions have finished.</p> <p>0: Disable JENDC interruption. 1: Enable JENDC interruption. An interrupt occurred when hardware set ADC_STS.JENDC bit.</p>
6	AWDGIEN	<p>Analog watchdog interrupt enable</p> <p>This bit is set and cleared by software to disallow or allow interrupt generated by analog watchdog. In scan mode, if the watchdog detects an out-of-range value, the scan is aborted only when that bit is set.</p> <p>0: Disable analog watchdog interruption. 1: Enable analog watchdog interruption.</p>
5	ENDCIEN	<p>Interrupt enable for any channels</p> <p>This bit is set and cleared by the software to disallow or allow interrupts to occur after the regular channel conversion ends.</p> <p>0: Disable ENDC interruption. 1: Enable ENDC interruption.</p>
4:0	AWDGCH[4:0]	<p>Analog watchdog channel select bits</p> <p>These bits are set and cleared by software to select input channels that analog watchdog protection.</p> <p>00000: ADC analog input channel 0 00001: ADC analog input channel 1 ... 01110: ADC analog input channel 14 01111: ADC analog input channel 15 10000: ADC analog input channel 16 10001: ADC analog input channel 17 10010: ADC analog input channel 18 Reserved all other values.</p>

15.10.4 ADC Control Register 2 (ADC_CTRL2)

Address offset: 0x08

Reset value: 0x0000 0000



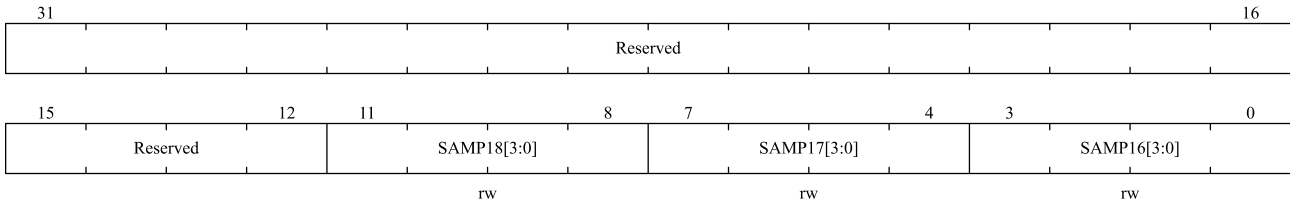
Bit Field	Name	Description
31:24	Reserved	Reserved,the reset value must be maintained
23	TEMPEN	Temperature sensor Enable This bit is set and cleared by the software to enable or disable the temperature sensor channel. 0: Disables the temperature sensor. 1: Enable the temperature sensor.
22	SWSTRRCH	Start conversion of regular channels This bit is set by the software to start the conversion and cleared by the hardware as soon as the conversion begins. If SWSTRRCH is selected as the trigger event in the ADC_CTRL2.EXTRSEL[2:0] bit, which is used to initiate the conversion of a set of regular channels 0: Reset state. 1: Starts converting the regular channel.
21	SWSTRJCH	Start conversion of injected channels This bit is set by the software to initiate the conversion and can be cleared by the software or by the hardware as soon as the conversion begins. If SWSTRJCH is selected as the trigger event in the ADC_CTRL2.EXTJSEL[2:0] bit, which is used to initiate a conversion of a set of injected channels 0: Reset state. 1: Starts converting the injection channel.
20	EXTRTRIG	External trigger conversion mode for regular channels This bit is set and cleared by software to enable or disable external triggering events that can start regular sequence conversion. 0: Start conversion without external events. 1: Use an external event to start the conversion.
19:17	EXTRSEL[2:0]	External event select for regular sequence These bits select external events to start the regular sequence conversion The triggering configuration of ADC is as follows 000: indicates the CC1 event of timer 1 100: indicates the TRGO event of timer 3 001: indicates the CC2 event of timer 1 101: indicates the CC4 event of timer 4 010: indicates the CC3 event of timer 1 110: EXTI line 0~15/TIM8_TRGO event 011: Reserved 111: SWSTRRCH
16	Reserved	Reserved,the reset value must be maintained

Bit Field	Name	Description
15	EXTJTRIG	<p>External trigger conversion mode for injected channels</p> <p>This bit is set and cleared by software to enable or disable external triggering events that can start injection sequence conversion.</p> <p>0: Start conversion without external events. 1: Use an external event to start the conversion.</p>
14:12	EXTJSEL[2:0]	<p>External event select for injected sequence</p> <p>These bits select the External event used to trigger the injected sequence conversion. The triggering configuration of ADC is as follows</p> <p>000: indicates the TRGO event of timer 1 100: indicates the CC4 event of timer 3 001: indicates the CC4 event of timer 1 101: indicates the TRGO event of timer 4 010: Reserved 110: EXTI line 0~15/TIM8_CC4 event 011: Reserved 111: SWSTRJCH</p>
11	ALIG	<p>Data alignment</p> <p>This bit is set and cleared by the software. Refer to Table 15-3 and Table 15-4.</p> <p>0: Right-aligned. 1: Left-aligned.</p>
10:9	Reserved	Reserved, the reset value must be maintained
8	ENDMA	<p>Direct memory access mode</p> <p>This bit is set and cleared by the software. See the DMA Controller chapter for details.</p> <p>0: Do not use DMA mode. 1: Use DMA mode.</p>
7:2	Reserved	Reserved, the reset value must be maintained
1	CTU	<p>Continuous conversion</p> <p>This bit is set and cleared by the software. If this bit is set, the conversion continues until the bit is cleared.</p> <p>0: Single conversion mode. 1: Continuous conversion mode.</p>
0	ON	<p>A/D converter ON/OFF</p> <p>This bit is set and cleared by the software. When the bit is '0', writing '1' will wake the ADC from power-off mode.</p> <p>When the bit is '1', writing '1' starts the conversion. The application should note that there is a delay t_{STAB} between the time the converter is powered on and the time the conversion begins, refer to Figure 15-3.</p> <p>0: Close ADC conversion/calibration and enter power-down mode. 1: Start ADC and start conversion.</p> <p>Note: If there are other bits changed in this register along with ON, the conversion will not be triggered. This is to prevent the wrong conversion from being triggered.</p>

15.10.5 ADC Sampling Time Register 1 (ADC_SAMPT1)

Address offset: 0x0C

Reset value: 0x0000 0000

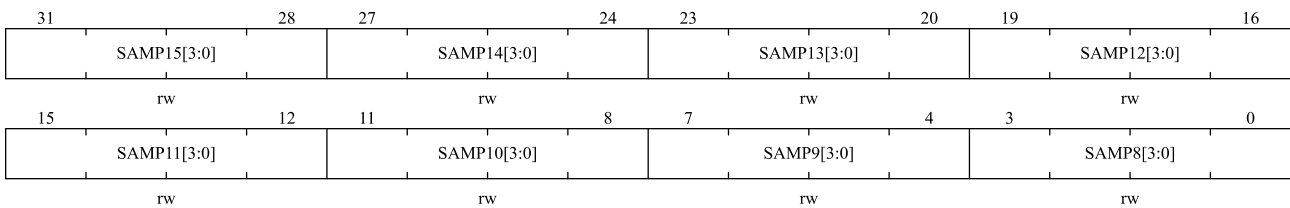


Bit Field	Name	Description																
31:12	Reserved	Reserved, the reset value must be maintained																
11:0	SAMPx[3:0] x=16~18	<p>Channel x sample time selection</p> <p>These bits are used to independently select the sampling time for each channel. The channel selection bit must remain constant during the sampling period.</p> <table border="0"> <tr> <td>0000: 6 cycles</td> <td>1000: 88 cycles</td> </tr> <tr> <td>0001: 8 cycles</td> <td>1001: 120 cycles</td> </tr> <tr> <td>0010: 14 cycles</td> <td>1010: 182 cycles</td> </tr> <tr> <td>0011: 20 cycles</td> <td>1011: 240 cycles</td> </tr> <tr> <td>0100: 29 cycles</td> <td>1100: 300 cycles</td> </tr> <tr> <td>0101: 42 cycles</td> <td>1101: 400 cycles</td> </tr> <tr> <td>0110: 56 cycles</td> <td>1110: 480 cycles</td> </tr> <tr> <td>0111: 72 cycles</td> <td>1111: 600 cycles</td> </tr> </table> <p><i>NOTE: ADC analog channel 16 and channel 17 are internally connected to the temperature sensor and VREFINT, respectively.</i></p>	0000: 6 cycles	1000: 88 cycles	0001: 8 cycles	1001: 120 cycles	0010: 14 cycles	1010: 182 cycles	0011: 20 cycles	1011: 240 cycles	0100: 29 cycles	1100: 300 cycles	0101: 42 cycles	1101: 400 cycles	0110: 56 cycles	1110: 480 cycles	0111: 72 cycles	1111: 600 cycles
0000: 6 cycles	1000: 88 cycles																	
0001: 8 cycles	1001: 120 cycles																	
0010: 14 cycles	1010: 182 cycles																	
0011: 20 cycles	1011: 240 cycles																	
0100: 29 cycles	1100: 300 cycles																	
0101: 42 cycles	1101: 400 cycles																	
0110: 56 cycles	1110: 480 cycles																	
0111: 72 cycles	1111: 600 cycles																	

15.10.6 ADC Sampling Time Register 2 (ADC_SAMPT2)

Address offset: 0x10

Reset value: 0x0000 0000

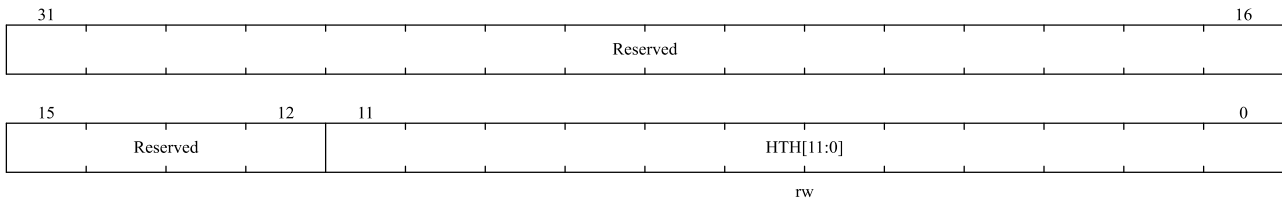


Bit Field	Name	Description														
31:0	SAMPx[3:0] x=8~15	<p>Channel x sample time selection</p> <p>These bits are used to independently select the sampling time for each channel. The channel selection bit must remain constant during the sampling period.</p> <table border="0"> <tr> <td>0000: 6 cycles</td> <td>1000: 88 cycles</td> </tr> <tr> <td>0001: 8 cycles</td> <td>1001: 120 cycles</td> </tr> <tr> <td>0010: 14 cycles</td> <td>1010: 182 cycles</td> </tr> <tr> <td>0011: 20 cycles</td> <td>1011: 240 cycles</td> </tr> <tr> <td>0100: 29 cycles</td> <td>1100: 300 cycles</td> </tr> <tr> <td>0101: 42 cycles</td> <td>1101: 400 cycles</td> </tr> <tr> <td>0110: 56 cycles</td> <td>1110: 480 cycles</td> </tr> </table>	0000: 6 cycles	1000: 88 cycles	0001: 8 cycles	1001: 120 cycles	0010: 14 cycles	1010: 182 cycles	0011: 20 cycles	1011: 240 cycles	0100: 29 cycles	1100: 300 cycles	0101: 42 cycles	1101: 400 cycles	0110: 56 cycles	1110: 480 cycles
0000: 6 cycles	1000: 88 cycles															
0001: 8 cycles	1001: 120 cycles															
0010: 14 cycles	1010: 182 cycles															
0011: 20 cycles	1011: 240 cycles															
0100: 29 cycles	1100: 300 cycles															
0101: 42 cycles	1101: 400 cycles															
0110: 56 cycles	1110: 480 cycles															

15.10.9 ADC Watchdog High Threshold Register (ADC_WDGHIGH)

Address offset: 0x28

Reset value: 0x0000 0FFF

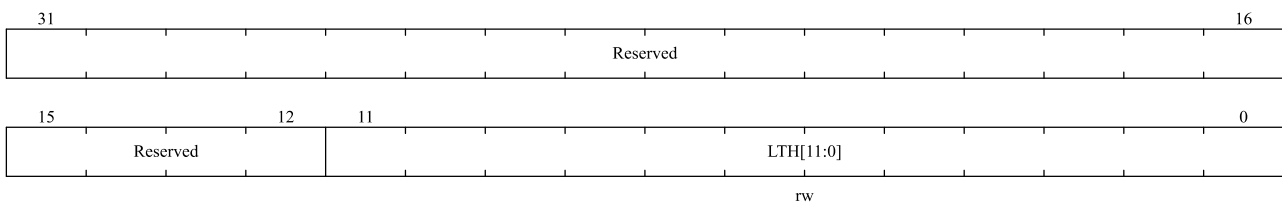


Bit field	Name	Description
31:12	Reserved	Reserved,the reset value must be maintained
11:0	HTH[11:0]	Analog watchdog high threshold These bits define the high thresholds for analog watchdog.

15.10.10 ADC Watchdog Low Threshold Register (ADC_WDGLOW)

Address offset: 0x2C

Reset value: 0x0000 0000

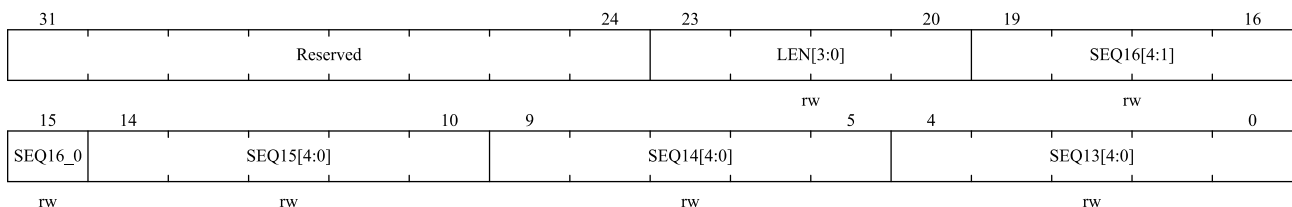


Bit Field	Name	Description
31:12	Reserved	Reserved,the reset value must be maintained
11:0	LTH[11:0]	Analog watchdog low threshold These bits define the low thresholds for analog watchdog.

15.10.11 ADC Regular Sequence Register 1 (ADC_RSEQ1)

Address offset: 0x30

Reset value: 0x0000 0000



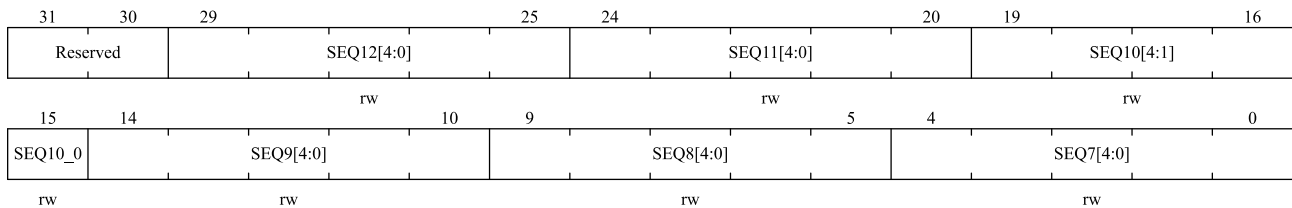
Bit Field	Name	Description
31:24	Reserved	Reserved,the reset value must be maintained

Bit Field	Name	Description
23:20	LEN[3:0]	Regular channel sequence length These bits are software-defined as the number of channels in the regular sequence channel conversion. 0000: 1 conversion 0001: 2 conversions ... 1111: 16 conversions
19:15	SEQ16[4:0]	16th conversion in regular sequence These bits are software-defined as the number (0 to 18) of the 16th conversion channel in the conversion sequence.
14:10	SEQ15[4:0]	15th conversion in regular sequence
9:5	SEQ14[4:0]	14th conversion in regular sequence
4:0	SEQ13[4:0]	13th conversion in regular sequence

15.10.12 ADC Regular Sequence Register 2 (ADC_RSEQ2)

Address offset: 0x34

Reset value: 0x0000 0000

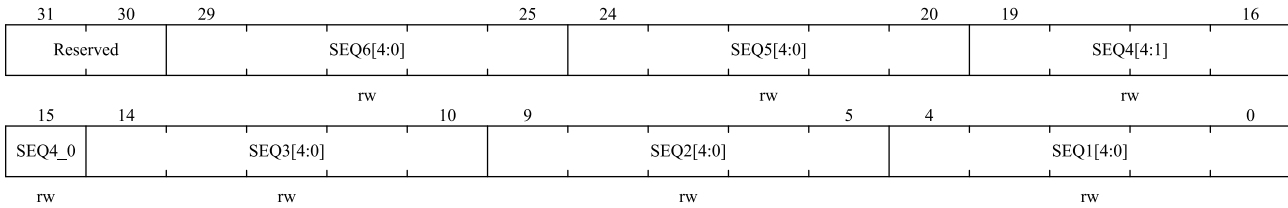


Bit Field	Name	Description
31:30	Reserved	Reserved, the reset value must be maintained
29:25	SEQ12[4:0]	12th conversion in regular sequence These bits are software-defined as the number (0 to 18) of the 12th conversion channel in the conversion sequence.
24:20	SEQ11[4:0]	11th conversion in regular sequence
19:15	SEQ10[4:0]	10th conversion in regular sequence
14:10	SEQ9[4:0]	9th conversion in regular sequence
9:5	SEQ8[4:0]	8th conversion in regular sequence
4:0	SEQ7[4:0]	7th conversion in regular sequence

15.10.13 ADC Regular Sequence Register 3 (ADC_RSEQ3)

Address offset: 0x38

Reset value: 0x0000 0000

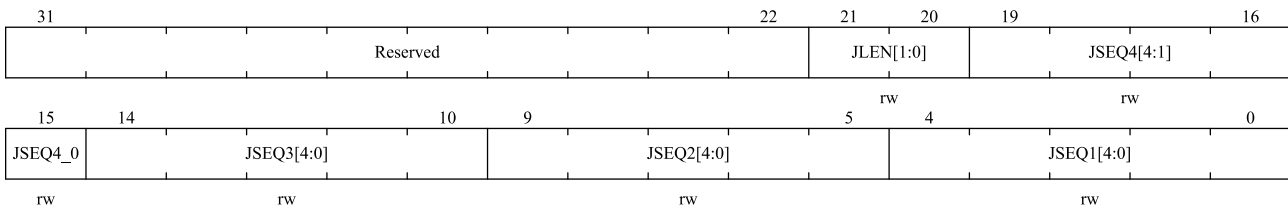


Bit Field	Name	Description
31:30	Reserved	Reserved, the reset value must be maintained
29:25	SEQ6[4:0]	6th conversion in regular sequence These bits are software-defined as the number (0 to 18) of the 6th transition channel in the conversion sequence.
24:20	SEQ5[4:0]	5th conversion in regular sequence
19:15	SEQ4[4:0]	4th conversion in regular sequence
14:10	SEQ3[4:0]	3rd conversion in regular sequence
9:5	SEQ2[4:0]	2nd conversion in regular sequence
4:0	SEQ1[4:0]	1st conversion in regular sequence

15.10.14 ADC Injection Sequence Register (ADC_JSEQ)

Address offset: 0x3C

Reset value: 0x0000 0000



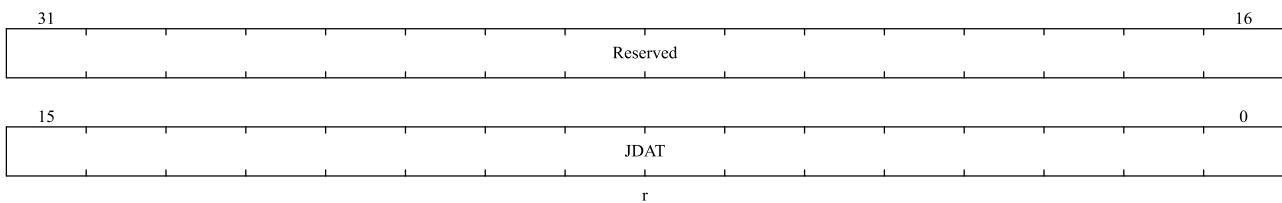
Bit Field	Name	Description
31:22	Reserved	Reserved, the reset value must be maintained
21:20	JLEN[1:0]	Injected sequence length These bits are software-defined as the number of channels in the injected channel conversion sequence. 00: 1 conversion 01: 2 conversions 10: 3 conversions 11: 4 conversions
19:15	JSEQ4[4:0]	This is the 4th conversion in the injected sequence. These bits are software-defined as the number (0 to 18) of the fourth transition channel in the conversion sequence. <i>Note: Different from regular conversion sequences, if the length of ADC_JSEQ.JLEN[1:0] is less than 4, the sequence of conversion starts from (4-JLEN). For example, ADC_JSEQ[21:0] = 100011 00011 00111 00010 means that the scan conversion will be converted in the following channel</i>

Bit Field	Name	Description
		<i>order: 7, 3, 3 instead of 2, 7, 3.</i>
14:10	JSEQ3[4:0]	3rd conversion in injected sequence
9:5	JSEQ2[4:0]	2nd conversion in injected sequence
4:0	JSEQ1[4:0]	1st conversion in injected sequence

15.10.15 ADC Injection Data Register X (ADC_JDATx) (x= 1...4)

Address offset: 0x40 - 0x4C

Reset value: 0x0000 0000

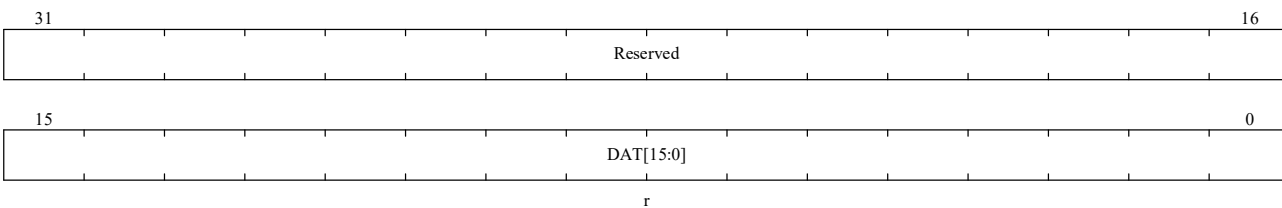


Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained
15:0	JDATx[15:0]	Injected data for conversions These bits are read-only and contain the conversion results of the injected channel. The data is left-aligned or right-aligned

15.10.16 ADC Regulars Data Register (ADC_DAT)

Address offset: 0x50

Reset value: 0x0000 0000

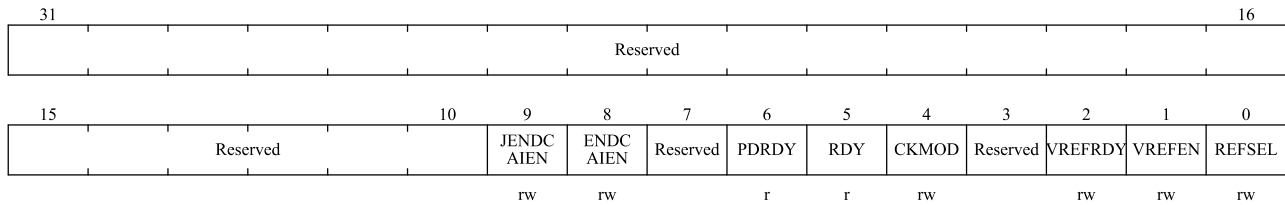


Bit Field	Name	Description
32:16	Reserved	Reserved, the reset value must be maintained
15:0	DAT[15:0]	Regular data for conversion These bits are read-only and contain the conversion results of the regular channel. The data is left-aligned or right-aligned.

15.10.17 ADC Control Register 3 (ADC_CTRL3)

Address offset: 0x54

Reset value: 0x0000 0040



Bit Field	Name	Description
31:10	Reserved	Reserved,the reset value must be maintained
9	JENDCAIEN	Interrupt enable for any injected channels This bit is set and cleared by the software to enable/disable the injection channel conversion end interrupt 0: ADC_STS.JENDCA interrupt is disabled 1: ADC_STS.JENDCA interrupt is enabled
8	ENDCAIEN	Interrupt enable for any channels This bit is set and cleared by the software to enable/disable any channel conversion end the interrupt 0: ADC_STS.ENDCA interrupt is disabled 1: ADC_STS.ENDCA interrupt is enabled
7	Reserved	Reserved, the value is forcibly set to 1.
6	PDRDY	ADC power ready 0: ADC is powered on 1: ADC is powered down
5	RDY	ADC ready 0: Not ready 1: Get ready
4	CKMOD	Clock mode 0: Select AHB for synchronization clock 1: Select PLL for asynchronous clock
3	Reserved	Reserved,the reset value must be maintained
2	VREFRDY	VREFINT_READY ADC internal input buffer ready status, software must check this status bit before measuring VREFINT 0: VREFINT not ready 1: VREFINT is ready
1	VREFEN	VREFINT Enable ADC internal input buffer is enabled, software must enable this bit before measuring VREFINT 0: Disable VREFINT measurement 1: Enable VREFINT measurement
0	REFSEL	ADC reference source setset 0: The reference source is the external reference V _{DD} 1: The reference source is the internal voltage 2.4V

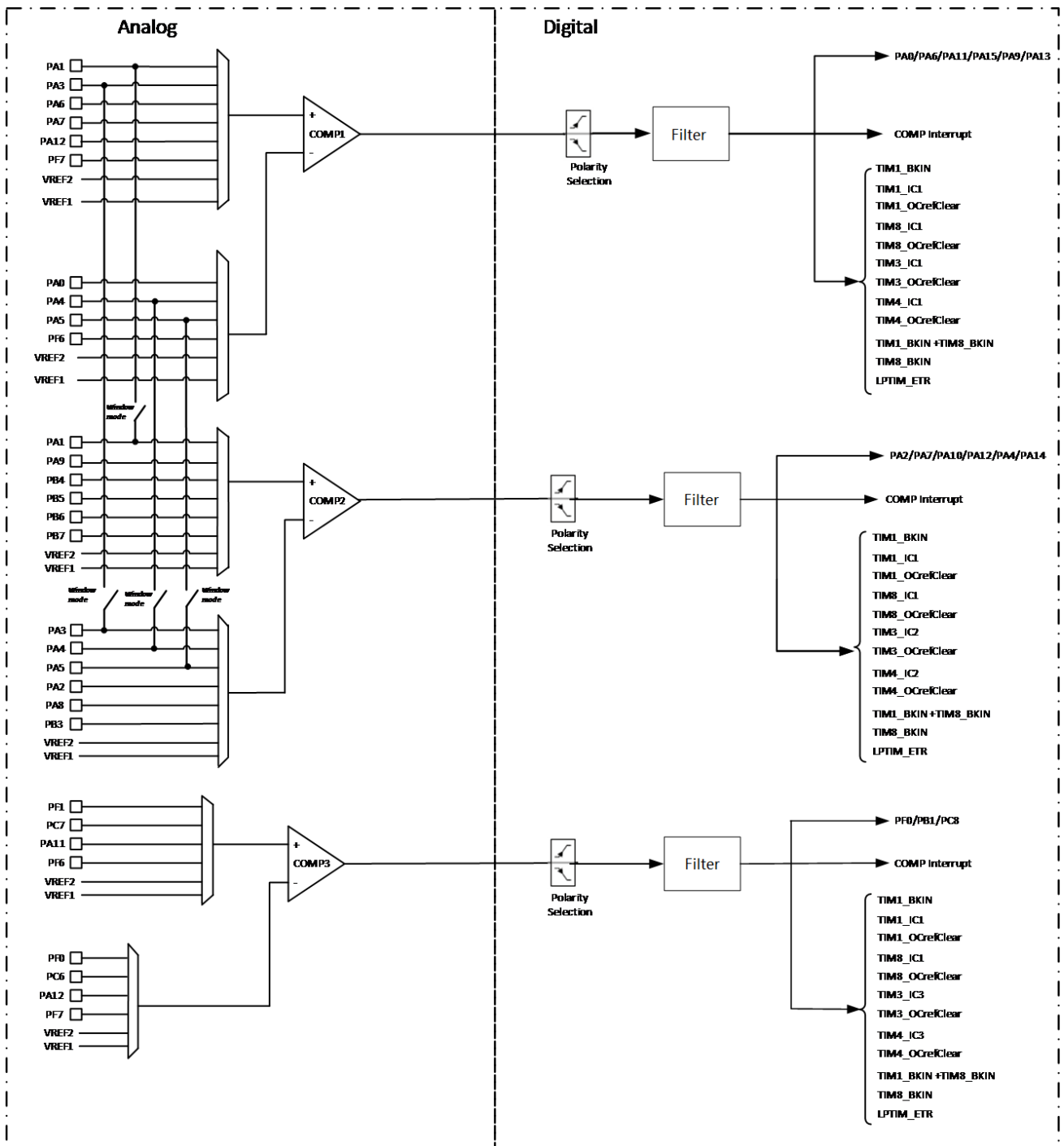
16 Comparator (COMP)

The COMP module is used to compare the magnitude of the two input analog voltages, and output high/low levels according to the comparison result. When the 'INP' input voltage is higher than the 'INM' input voltage, the comparator outputs are high level, and when the "INP" input voltage is lower than the "INM" input voltage, the comparator output is low level.

16.1 COMP System Connection Block Diagram

The COMP module supports up to 3 independent comparator which are uniformly connected to the APB1 bus.

Figure 16-1 Comparator System Connection Diagram



16.2 COMP Features

- 3 independent COMP1/COMP2/COMP3, COMP1 supports low power mode(can work at LPRUN, SLEEP, STOP mode)
- Internal 64-level programmable reference input compare voltage source V_{REF1}/V_{REF2} .
- Support filter clock, filter reset
- Output polarity can be configured to high or low

- Programmable hysteresis can be configured as no hysteresis, low hysteresis, medium hysteresis, and high hysteresis
- The comparator can output to either I/O or timer input for capturing events, OCREF_CLR events, breaking events, triggering event.
- Input channel can select I/O port/VREF1/VREF2
- Can be configured with read-only or read-write, and needs to be reset to unlock when locked
- Support blanking, blanking source can be configured
- COMP1/COMP2 can form a window comparator
- COMP can wake up the system from low power mode by generating an interrupt, and COMP1 can wake up the system from STOP. COMP1 output trigger interrupt by connect to EXTI.
- Configurable filter window size
- Configurable filter threshold size
- Configurable sampling frequency for filtering

16.3 COMP Configuration Procedure

The complete configuration includes the following steps. If the default configuration is used, skip the corresponding configuration items.

1. Configure hysteresis level COMPx_CTRL.HYST[1:0]
2. Configure the output polarity COMPx_CTRL.POL
3. Configure the input selection, comparator non-inverting input COMPx_CTRL.INPSEL[2:0], inverting input COMPx_CTRL.INMSEL [2:0]
4. configure the output selection COMPx_CTRL.OUTSEL[3:0]
5. Configure the blanking source COMPx_CTRL.BLKING[2:0]
6. Configure the filter sampling window COMPx_FILC.SAMPW[4:0]
7. Configure the threshold COMPx_FILC.THRESH[4:0] (threshold should be greater than COMPx_FILC.SAMPW[4:0]/2)
8. Configure the filter sampling frequency (for timer applications, sampling frequency should be greater than 5MHz)
9. Enable the filter COMPx_FILC.FILEN
10. Enable the comparator COMPx_CTRL.EN

Note: For the above steps, you need to enable the filter first, and then the the comparator. The comparator needs to be enabled after the filter (if enabled) configuration and enable is completed. In addition, when the comparator control register is locked, it can only be unlocked by reset.

16.4 COMP Operating Mode

16.4.1 Window Mode

Comparators can be combined as window comparator, as follows:

- Comparators 1 and 2 share PA1, PA3, PA4, and PA5 to form a window comparator.

PA1 (COMP1 non-inverting input, COMP2 non-inverting input)

PA3 (COMP1 non-inverting input, COMP2 inverting input)

PA4 (COMP1 inverting input, COMP2 inverting input)

PA5 (COMP1 inverting input, COMP2 inverting input)

Configure CMP12MD to enable the window mode, supporting 4 different WINMDINSEL configurations. The descriptions of INPSEL and INMSEL under different configurations are shown in the table below:

WINMDINSEL	COMP1		COMP2		COMP_OUT
	INP	INM	INP	INM	
00	PA1(INPSEL=000)	User control	PA1(INPSEL=000)	User control	OUT1 XOR OUT2
01	PA3(INPSEL=001)	User control	User control	PA3(INMSEL=00 0)	OUT1 XNOR OUT2
10	User control	PA4(INMSEL=00 1)	User control	PA4(INMSEL=00 1)	OUT1 XOR OUT2
11	User control	PA5(INMSEL=01 0)	User control	PA5(INMSEL=01 0)	OUT1 XOR OUT2

- In window mode, user can choose the window comparator result as the output of COMP2 by setting WINOUT = 1 in the COMP2_CTRL register. When WINOUT = 0, the COMP2 output comes from the analog comparator.

16.4.2 Independent Comparator

3 comparators can be configured independently to complete the comparator function. The output of the comparator can be output to the IO port, the comparator supports different remapping ports, and the output of the comparator can be selected and connected to the corresponding port through configuration.

The comparator outputs supports triggering events. Example, it can be configured as the brake function of timer 1, timer 8.

Note: Refer to the comparator interconnection for specific configuration.

16.5 Comparator Interconnection

For the interconnection of the comparator output ports, please refer to the AFIO remapping chapter, Remap IO multiplexing function in GPIOx_AFL/AFH.

COMP1_OUT can be mapped to PA0/PA6/PA9/PA11/PA13/PA15

COMP2_OUT can be mapped to PA2/PA4/PA7/PA10/PA12/PA14

COMP3_OUT can be mapped to PF0/PB1/PC8

Comparator INP pins have the following configuration:

INPSEL	COMP1	COMP2	COMP3
000	PA1	PA1	PF1
001	PA3	PA9	PC7
010	PA6	PB4	PA11
011	PA7	PB5	PF6
100	PA12	PB6	--
101	PF7	PB7	--
110	VREF2	VREF2	VREF2
111	VREF1	VREF1	VREF1

The comparator INM pins have the following configuration.

INMSEL	COMP1	COMP2	COMP3
000	PA0	PA3	PF0
001	PA4	PA4	PC6
010	PA5	PA5	PA12
011	PF6	PA2	PF7
100	--	PA8	--
101	--	PB3	--
110	VREF2	VREF2	VREF2
111	VREF1	VREF1	VREF1

Comparator output TRIG signal has the following interconnection.

TRIG	COMP1	COMP2	COMP3
0000	NC	NC	NC
0001	TIM1_BKIN	TIM1_BKIN	TIM1_BKIN
0010	TIM1_IC1	TIM1_IC1	TIM1_IC1
0011	TIM1_OCrefclear	TIM1_OCrefclear	TIM1_OCrefclear
0100	TIM8_IC1	TIM8_IC1	TIM8_IC1
0101	TIM8_OCrefclear	TIM8_OCrefclear	TIM8_OCrefclear
0110	TIM3_IC1	TIM3_IC2	TIM3_IC3
0111	TIM3_OCrefclear	TIM3_OCrefclear	TIM3_OCrefclear
1000	--	--	--
1001	TIM4_IC1	TIM4_IC2	TIM4_IC3
1010	TIM4_OCrefclear	TIM4_OCrefclear	TIM4_OCrefclear
1011	TIM1_BKIN + TIM8_BKIN	TIM1_BKIN + TIM8_BKIN	TIM1_BKIN + TIM8_BKIN
1100	TIM8_BKIN	TIM8_BKIN	TIM8_BKIN
1101	LPTIM_ETR	LPTIM_ETR	LPTIM_ETR
Other	--	--	--

16.6 Interrupt

COMP supports interrupt response. COMP1, COMP2, and COMP3 share one interrupt entry. There are two cases of interrupt generation as follows:

- The COMP_x_CTRL register sets the polarity of the POL bit is not reversed and the interrupt is enabled. When INPSEL > INMSEL, the comparator interrupt will be generated when the OUT bit of the COMP_x_CTRL register is set to 1 by hardware.
- The COMP_x_CTRL register sets the polarity of the POL bit is reversed and the interrupt is enabled. When INPSEL < INMSEL, the comparator interrupt will be generated when the OUT bit of the COMP_x_CTRL register is set to 1 by hardware.

Note: To use the COMP interrupt functions, user must configure EXTI_LINE18 first.

16.7 COMP Register

16.7.1 COMP Register Overview

Table 16-1 COMP Register Overview

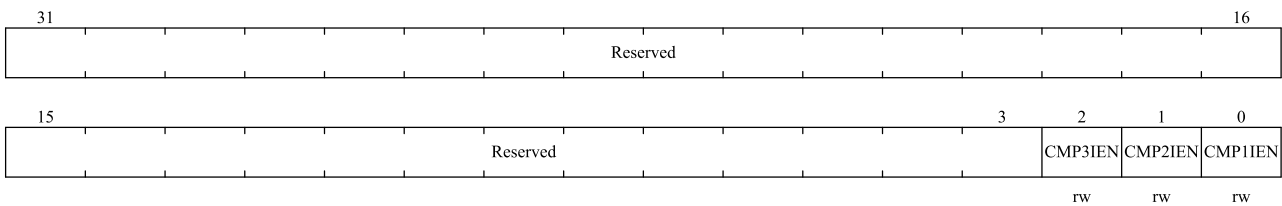
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
000h	COMP_INTEN	Reserved																									CMP3IEN	CMP2IEN	CMP1IEN												
	Reset Value																										0	0	0												
004h	COMP_INTSTS	Reserved																									CMP3IS	CMP2IS	CMP1IS												
	Reset Value																										0	0	0												
008h	COMP_WINMODE	Reserved																									WINMDIN	WINMDIN	CMP12MD												
	Reset Value																										0	0	0												
00Ch	COMP_LOCK	Reserved																									CMP3LK	CMP2LK	CMP1LK												
	Reset Value																										0	0	0												
010h	COMP1_CTRL	Reserved												CLKSEL	PWRMD	Reserved	OUT	BLKING[2:0]	HYST[1:0]	POL	OUTTRG[3:0]	INPSEL[2:0]	INMSEL[2:0]	EN																	
	Reset Value													0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0										
014h	COMP1_FILC	Reserved												SAMPWIN[4:0]				THRESH[4:0]				FILEN																			
	Reset Value													0	0	0	0	0	0	0	0	0	0	0	0	0															
018h	COMP1_FILP	Reserved												CLKPSC[15:0]																											
	Reset Value													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
01Ch	Reserved																																								
020h	COMP2_CTRL	Reserved												WINOUT	OUT	BLKING[2:0]	HYST[1:0]	POL	OUTTRG[3:0]	INPSEL[2:0]	INMSEL[2:0]	EN																			
	Reset Value													0	0	0	0	0	0	0	0	0	0	0	0	0	0														
024h	COMP2_FILC	Reserved												SAMPWIN[4:0]				THRESH[4:0]				FILEN																			
	Reset Value													0	0	0	0	0	0	0	0	0	0	0	0	0															

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																				
	Reset Value	Reserved																						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
028h	COMP2_FILP	Reserved															CLKPSC[15:0]																																				
	Reset Value	0																																																			
02Ch	Reserved																																																				
030h	COMP3_CTRL	Reserved															OUT	BLKING[2:0]		HYST[1:0]		POL	OUTTRG[3:0]			INPSEL[2:0]		INMSEL[2:0]		EN																							
	Reset Value	0															0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								
034h	COMP3_FILC	Reserved																		SAMPWIN[4:0]				THRESH[4:0]				FILEN																									
	Reset Value	0																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
038h	COMP3_FILP	Reserved															CLKPSC[15:0]																																				
	Reset Value	0																																																			
03Ch	Reserved																																																				
040h	COMP_INVREF	Reserved																		VREF2SEL				VREF2EN	VREF1SEL				VREF1EN																								
	Reset Value	0																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					

16.7.2 COMP Interrupt Enable Register (COMP_INTEN)

Address offset : 0x00

Reset value : 0x0000 0000

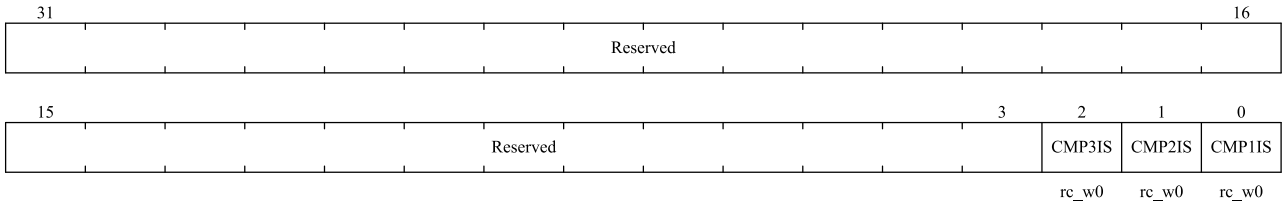


Bit Field	Name	Description
31:3	Reserved	Reserved, the reset value must be maintained
2	CMP3IEN	COMP3 interrupt enable 0: disable 1: enable
1	CMP2IEN	COMP2 interrupt enable 0: disable 1: enable
0	CMP1IEN	COMP1 interrupt enable 0: disable 1: enable

16.7.3 COMP Interrupt Register (COMP_INTSTS)

Address offset : 0x04

Reset value : 0x0000 0000

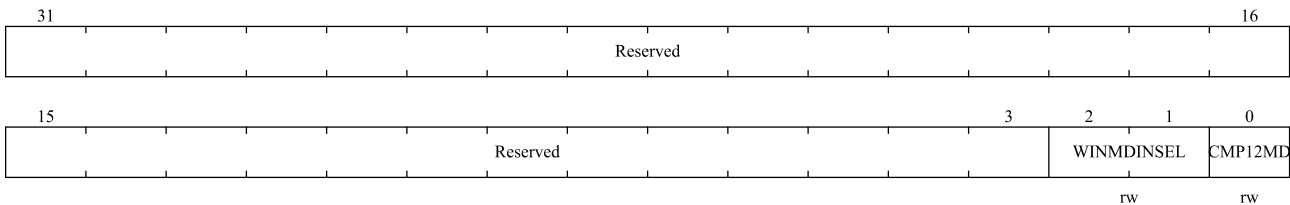


Bit Field	Name	Description
31:3	Reserved	Reserved,the reset value must be maintained
2	CMP3IS	interrupt status of COMP3 Write 0 to clear
1	CMP2IS	interrupt status of COMP2 Write 0 to clear
0	CMP1IS	interrupt status of COMP1 Write 0 to clear

16.7.4 COMP Window Mode Enable Register(COMP_WINMODE)

Address offset : 0x08

Reset value : 0x0000 0000



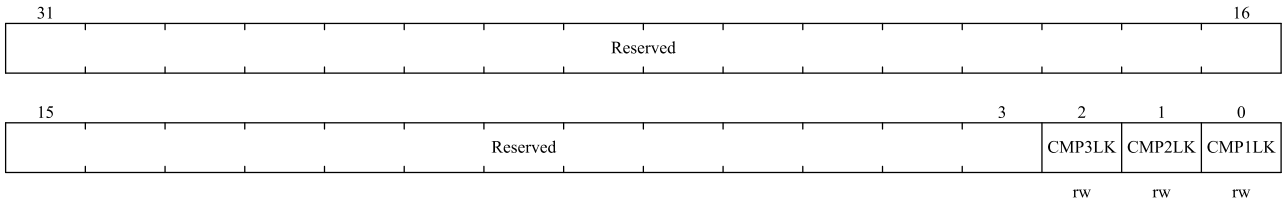
Bit Field	Name	Description
31:3	Reserved	Reserved, the reset value must be maintained
2:1	WINMDINSEL	Comparator 1 and 2 Window Mode Input Selection Only valid when CMP12MD = 1, Comparator 1 and 2 have a shared input. For details, refer to Section 16.4.1 Window Mode. 00: Comparator 1 INPSEL[2:0] = 000, Comparator 2 INPSEL[2:0] = 000, shared input at PA1; 01: Comparator 1 INPSEL[2:0] = 001, Comparator 2 INMSEL[2:0] = 000, shared input at PA3; 10: Comparator 1 INMSEL [2:0] = 001, Comparator 2 INMSEL [2:0] = 001, shared input at PA4; 11: Comparator 1 INMSEL [2:0] = 010, Comparator 2 INMSEL[2:0] = 010, shared input at PA5;
0	CMP12MD	Comparator 1 and Comparator 2 window mode selection bit. 0: Comparators 1 and 2 are not in window mode.

Bit Field	Name	Description
		1: Comparators 1 and 2 are in window mode.

16.7.5 COMP Lock Register (COMP_LOCK)

Address offset : 0x0C

Reset value : 0x0000 0000

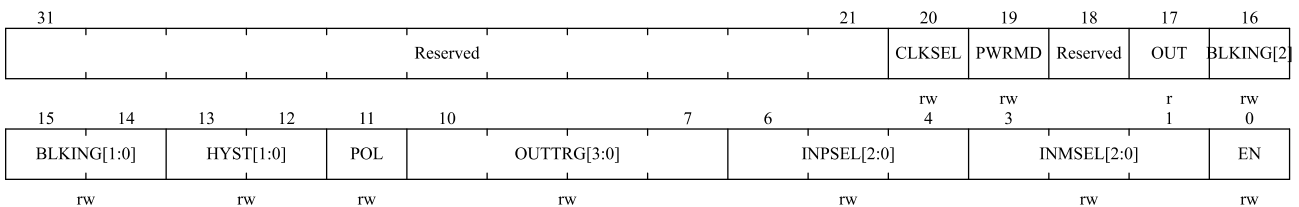


Bit Field	Name	Description
31:3	Reserved	Reserved, the reset value must be maintained
2	CMP3LK	This bit can only be reset then written once by software. If software is set to 1, the COMP3_CTRL register will become a read-only register 0: COMP3_CTRL can be read and written 1: COMP3_CTRL read only
1	CMP2LK	This bit can only be reset then written once by software. If software is set to 1, the COMP2_CTRL register will become a read-only register 0: COMP2_CTRL can be read and written 1: COMP2_CTRL read only
0	CMP1LK	This bit can only be reset then written once by software. If software is set to 1, the COMP1_CTRL register will become a read-only register 0: COMP1_CTRL can be read and written 1: COMP1_CTRL read only

16.7.6 COMP1 Control Register (COMP1_CTRL)

Address offset : 0x10

Reset value : 0x0000 0000



Bit Field	Name	Description
31:21	Reserved	Reserved, the reset value must be maintained
20	CLKSEL	COMP1 operating clock selection 0: System clock (SYSCLK) 1: Low-speed working clock, can work in STOP mode or LPRUN mode.

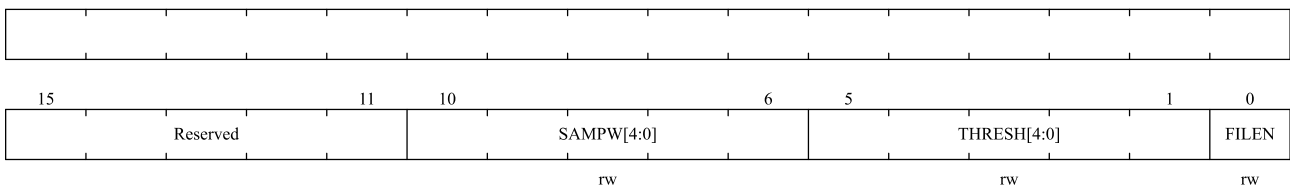
Bit Field	Name	Description
19	PWRMD	COMP1 power select 0: normal mode 1: Low power mode
18	Reserved	Reserved, the reset value must be maintained
17	OUT	This read-only bit is COMP1 output state. 0: Output is low 1: Output is high
16:14	BLKING[2:0]	These bits select which Timer output controls the COMP1 output blanking. 000: No blanking 001: TIM1 OC5 selected as blanking source 010: TIM8 OC5 selected as blanking source Other values: reserved
13:12	HYST[1:0]	These bits control the hysteresis level. 00: No hysteresis 01: Low hysteresis 10: Medium hysteresis 11: High hysteresis
11	POL	This bit is used to invert the COMP1 output. 0: Output is not inverted 1: Output is inverted
10:7	OUTTRG[3:0]	These bits select which Timer input must be connected with the COMP1 output. 0000: Reserved 0001: TIM1_BKIN 0010: TIM1_IC1 0011: TIM1_OCrefclear 0100: TIM8_IC1 0101: TIM8_OCrefclear 0110: TIM3_IC1 0111: TIM3_OCrefclear 1000: Reserved 1001: TIM4_IC1 1010: TIM4_OCrefclear 1011: TIM1_BKIN+TIM8_BKIN 1100: TIM8_BKIN 1101: LPTIM_ETR 1110: Reserved 1111: Reserved
6:4	INPSEL[2:0]	COMP1 Non-inverting input selection 000: PA1 001: PA3 010: PA6 011: PA7 100: PA12

Bit Field	Name	Description
		101: PF7 110: VREF2 111: VREF1 <i>Note: If the window mode is enabled, the selection bits are configured with the WINMDINSEL field of the COMP_WINMODE register.</i>
3:1	INMSEL[2:0]	COMP1 Inverting input selection 000: PA0 001: PA4 010: PA5 011: PF6 100: Reserved 101: Reserved 110: VREF2 111: VREF1 <i>Note: If the window mode is enabled, the selection bits are configured with the WINMDINSEL field of the COMP_WINMODE register.</i>
0	EN	This bit switches COMP1 ON/OFF. 0: Disable 1: Enable

16.7.7 COMP1 Filter Control Register (COMP1_FILC)

Address offset : 0x14

Reset value : 0x0000 0000

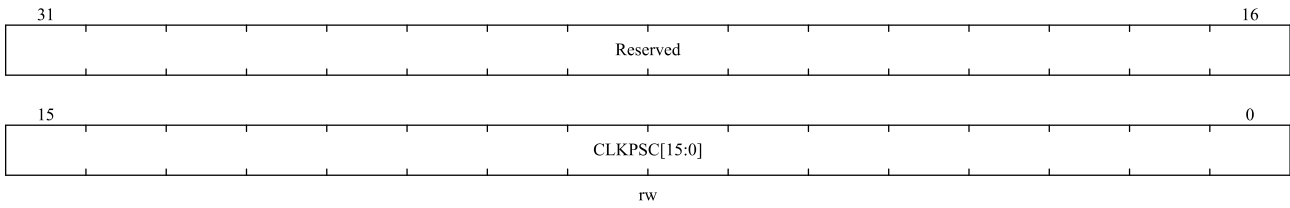


Bit Field	Name	Description
31:11	Reserved	Reserved, the reset value must be maintained
10:6	SAMPW[4:0]	Filter sampling window size, sampling window = SAMPW + 1.
5:1	THRESH[4:0]	The filter threshold is set. At least the sampling threshold of the opposite state in the sample window can change the output state. This value is required to be greater than SAMPW / 2.
0	FILEN	Filter enable. 0: Disable 1: Enable

16.7.8 COMP1 Filter Prescaler Register (COMP1_FILP)

Address offset : 0x18

Reset value : 0x0000 0000

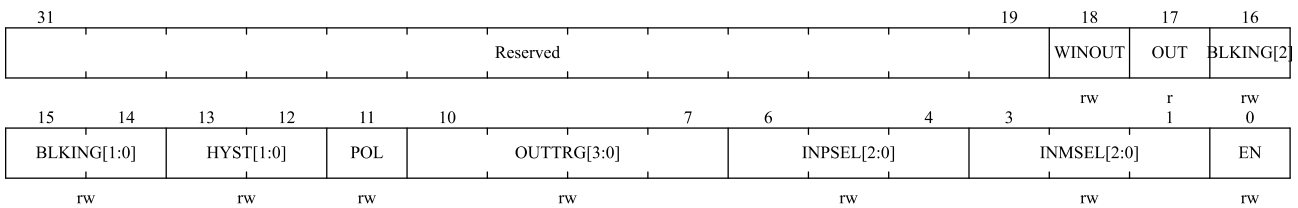


Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained
15:0	CLKPSC[15:0]	Low filter sample clock prescale. System clock divider = CLK_PRE_CYCLE + 1, 0: Every cycle 1: Every 2 cycle 2: Every 3 cycle ...

16.7.9 COMP2 Control Register (COMP2_CTRL)

Address offset : 0x20

Reset value : 0x0000 0000



Bit Field	Name	Description
31:19	Reserved	Reserved, the reset value must be maintained
18	WINOUT	Selecting Comparator 2 Output 0: Normal output of COMP2 1: Output of the COMP in window mode <i>Note: When window mode is enabled, WINOUT must be configured as 1 to use window mode properly.</i>
17	OUT	This read-only bit is COMP2 output state. 0: Output is low 1: Output is high
16:14	BLKING[2:0]	These bits select which Timer output controls the COMP2 output blanking. 000: No blanking 001: TIM1 OC5 selected as blanking source 010: TIM8 OC5 selected as blanking source Other values: reserved
13:12	HYST[1:0]	These bits control the hysteresis level. 00: No hysteresis

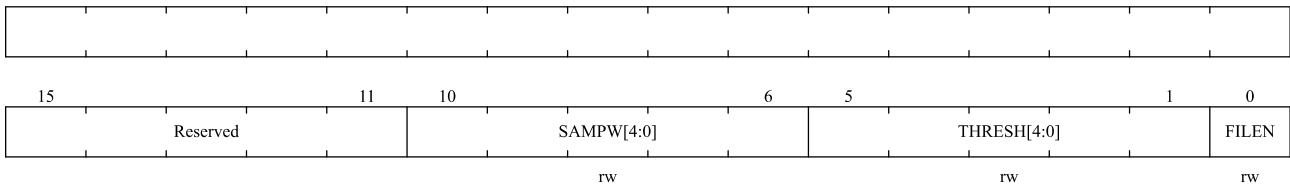
Bit Field	Name	Description
		01: Low hysteresis 10: Medium hysteresis 11: High hysteresis
11	POL	This bit is used to invert the COMP2 output. 0: Output is not inverted 1: Output is inverted
10:7	OUTTRG[3:0]	These bits select which Timer input must be connected with the COMP2 output. 0000: Reserved 0001: TIM1_BKIN 0010: TIM1_IC1 0011: TIM1_OCrefclear 0100: TIM8_IC1 0101: TIM8_OCrefclear 0110: TIM3_IC2 0111: TIM3_OCrefclear 1000: Reserved 1001: TIM4_IC2 1010: TIM4_OCrefclear 1011: TIM1_BKIN + TIM8_BKIN 1100: TIM8_BKIN 1101: LPTIM_ETR others: Reserved
6:4	INPSEL[2:0]	COMP2 Non-inverting input selection 000: PA1 001: PA9 010: PB4 011: PB5 100: PB6 101: PB7 110: VREF2 111: VREF1 <i>Note: If the window mode is enabled, the selection bits are configured with the WINMDINSEL field of the COMP_WINMODE register.</i>
3:1	INMSEL[2:0]	COMP2 Inverting input selection 000: PA3 001: PA4 010: PA5 011: PA2 100: PA8 101: PB3 110: VREF2 111: VREF1 <i>Note: If the window mode is enabled, the selection bits are configured with the</i>

Bit Field	Name	Description
		<i>WINMDINSEL</i> field of the <i>COMP_WINMODE</i> register.
0	EN	This bit switches COMP2 ON/OFF. 0: Disable 1: Enable

16.7.10 COMP2 Filter Control Register (COMP2_FILC)

Address offset : 0x24

Reset value : 0x0000 0000

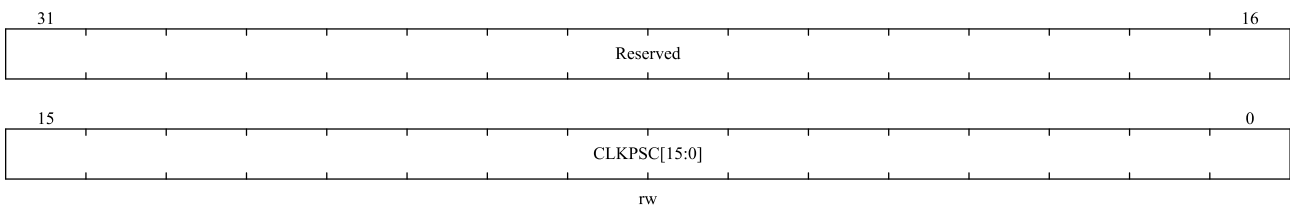


Bit Field	Name	Description
31:11	Reserved	Reserved, the reset value must be maintained
10:6	SAMPW[4:0]	Filter sampling window size, sampling window = SAMPW + 1.
5:1	THRESH[4:0]	The filter threshold is set. At least the sampling threshold of the opposite state in the sample window can change the output state. This value is required to be greater than SAMPW / 2.
0	FILEN	Filter enable. 0: Disable 1: Enable

16.7.11 COMP2 Filter Prescaler Register (COMP2_FILP)

Address offset : 0x28

Reset value : 0x0000 0000



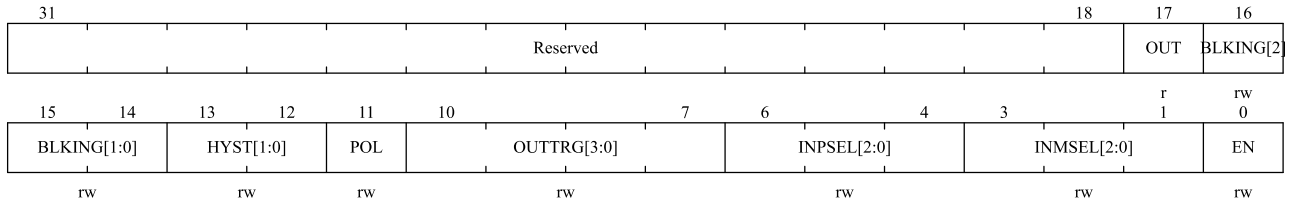
Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained
15:0	CLKPSC[15:0]	Low filter sample clock prescale. System clock divider = CLK_PRE_CYCLE + 1, 0: Every cycle 1: Every 2 cycle 2: Every 3 cycle

Bit Field	Name	Description
		...

16.7.12 COMP3 Control Register (COMP3_CTRL)

Address offset : 0x30

Reset value : 0x0000 0000



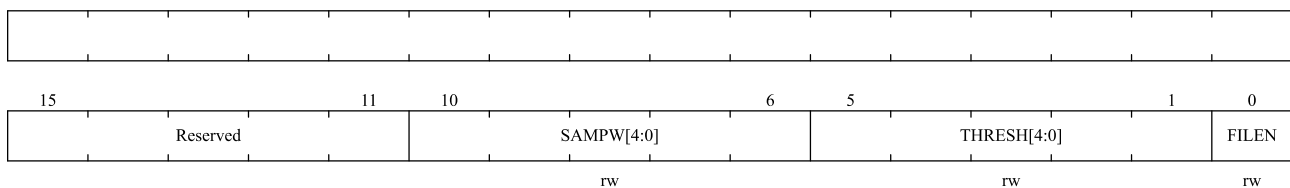
Bit Field	Name	Description
31:18	Reserved	Reserved, the reset value must be maintained
17	OUT	This read-only bit is COMP3 output state. 0: Output is low 1: Output is high
16:14	BLKING[2:0]	These bits select which Timer output controls the COMP3 output blanking. 000: No blanking 001: TIM1 OC5 selected as blanking source 010: TIM8 OC5 selected as blanking source Other values: reserved
13:12	HYST[1:0]	These bits control the hysteresis level. 00: No hysteresis 01: Low hysteresis 10: Medium hysteresis 11: High hysteresis
11	POL	This bit is used to invert the COMP3 output. 0: Output is not inverted 1: Output is inverted
10:7	OUTTRG[3:0]	These bits select which Timer input must be connected with the COMP3 output. 0000: Reserved 0001: TIM1_BKIN 0010: TIM1_IC1 0011: TIM1_OCrefclear 0100: TIM8_IC1 0101: TIM8_OCrefclear 0110: TIM3_IC3 0111: TIM3_OCrefclear 1000: Reserved 1001: TIM4_IC3 1010: TIM4_OCrefclear 1011: TIM1_BKIN + TIM8_BKIN

Bit Field	Name	Description
		1100: TIM8_BKIN 1101: LPTIM_ETR others: Reserved
6:4	INPSEL[2:0]	COMP3 Non-inverting input selection 000: PF1 001: PC7 010: PA11 011: PF6 100: Reserved 101: Reserved 110: VREF2 111: VREF1
3:1	INMSEL[2:0]	COMP3 Inverting input selection 000: PF0 001: PC6 010: PA12 011: PF7 100: Reserved 101: Reserved 110: VREF2 111: VREF1
0	EN	This bit switches COMP3 ON/OFF. 0: Disable 1: Enable

16.7.13 COMP3 Filter Control Register (COMP3_FILC)

Address offset : 0x34

Reset value : 0x0000 0000



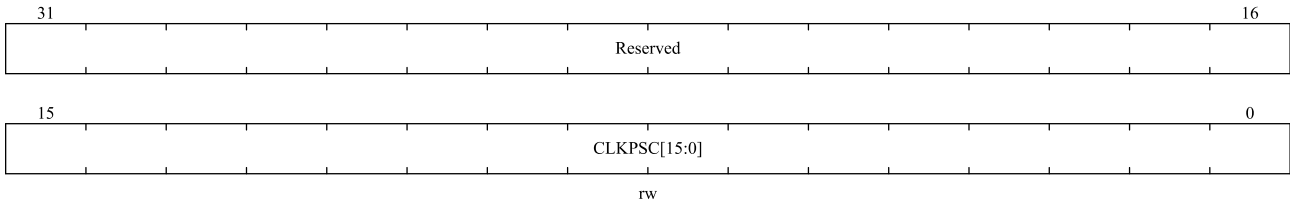
Bit Field	Name	Description
31:11	Reserved	Reserved, the reset value must be maintained
10:6	SAMPW[4:0]	Filter sampling window size, sampling window = SAMPW + 1.
5:1	THRESH[4:0]	The filter threshold is set. At least the sampling threshold of the opposite state in the sample window can change the output state. This value is required to be greater than SAMPW / 2.
0	FILEN	Filter enable. 0: Disable

Bit Field	Name	Description
		1: Enable

16.7.14 COMP3 Filter Prescaler Register (COMP3_FILP)

Address offset : 0x38

Reset value : 0x0000 0000

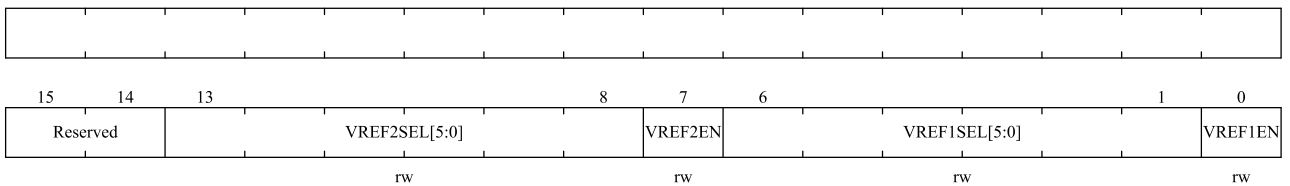


Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained
15:0	CLKPSC[15:0]	Low filter sample clock prescale. System clock divider = CLK_PRE_CYCLE + 1, 0: Every cycle 1: Every 2 cycle 2: Every 3 cycle ...

16.7.15 COMP Reference Input Compare Voltage Register (COMP_INVREF)

Address offset : 0x40

Reset value : 0x0000 0000



Bit Field	Name	Description
31:14	Reserved	Reserved, the reset value must be maintained
13:8	VREF2SEL[5:0]	Comparator reference input comparison voltage V_{REF2} gear selection 0~ V_{DDA} , a total of 64 gears
7	VREF2EN	Comparator reference input compare voltage V_{REF2} enable 0: disable 1: enable
6:1	VREF1SEL[5:0]	Comparator reference input comparison voltage V_{REF1} gear selection 0~ V_{DDA} , a total of 64 gears
0	VREF1EN	Comparator reference input compare voltage V_{REF1} enable 0: disable

Bit Field	Name	Description
		1: enable

17 I²C Interface

17.1 Introduction

The I²C(Inter-Integrated Circuit) bus is a widely used bus structure, it has only two bidirectional lines, the data bus SDA and clock bus SCL. All devices compatible with I²C bus can communicate directly with each other through I²C bus with these two lines.

I²C interface connects microcontroller and serial I²C bus, which can be used for communication between MCU and external I²C devices. It supports standard speed mode and fast mode, with CRC calculation and verification, and supports SMBus (System Management Bus) and PMBus (Power Management Bus), it also provides multi-host function to control all I²C bus specific timing, protocol, arbitration. I²C interface module also supports DMA mode, which can effectively reduce the burden on the CPU.

17.2 Main Features

- This module can be used as master device or slave device;
- I²C master device function:
 - Generate a clock;
 - Generate start and stop signals;
- Function of I²C slave device
 - Programmable address detection;
 - The I²C interface supports 7-bit or 10-bit addressing and dual-slave address response capability in 7-bit slave mode.
 - Stop bit detection;
- Generate and detect 7-bit/10-bit addresses and broadcast calls;
- Support different communication speeds;
 - Standard speed (up to 100 kHz);
 - Fast (up to 400 kHz);
 - Fast + (up to 1MHz);
- Status flags:
 - Transmitter/receiver mode flag;
 - Byte transmit complete flag;
 - I²C bus busy flag;
- Error flags:
 - Arbitration is missing in Master Mode.
 - Acknowledge (ACK) error after address/data transfer;

- Error start or stop condition detected
- Overrun or underrun when disable extend clock function;
- One interrupt vectors:
 - Event interrupt and error interrupt share one interrupt vector
- Optional extend clock function
- DMA of single-byte buffers;
- Generation or verification of configurable PEC(Packet errorchecking)
 - In transmit mode, the PEC value can be transmitted as the last byte
 - PEC error check for the last received byte
- SMBus 2.0 compatible
 - Timeout delay for 25 ms clock low
 - 10 ms accumulates low clock extension time of master device
 - 25 ms accumulates low clock extension time of slave device
 - PEC generation/verification of hardware with ACK control
 - Support address resolution protocol (ARP)
- Compatible with the PMBus
- I²C interface supports dual signal level communication, normal level (signal level matches chip V_{DD}) and low level (chip V_{DD} 3.3V or 5V, signal level 1.8V) two levels can be selected.

Note: not all of the above features are included in all products. Please refer to the relevant data manual to confirm the I²C functions supported by the product.

17.3 Function Description

The I²C interface is connected to I²C bus through data pin (SDA) and clock pin (SCL) to communicate with external devices. It can be connected to standard (up to 100kHz) or fast (up to 400kHz) or fast⁺ (up to 1MHz) I²C bus. I²C module converts data from serial to parallel when receiving, and converts data from parallel to serial when transmitting. It support interrupt mode, users can enable or disable interrupt according to their needs.

17.3.1 SDA And SCL Line Control

I²C module has two interface lines: serial data line (SDA) and serial clock line (SCL). Devices connected to the bus and exchange information through these two wires. Both SDA and SCL are bidirectional lines, connected to positive power supply through a current source or a pull-up resistor. When the bus is idle, both lines are high level. The output of device which is connected to the bus must have open drain or open collector to provide wired-AND functionality. The data on I²C bus can reach 100 kbit/s in standard mode and 1000 kbit/s in fast mode. Since devices of different processors may be connected to the I²C bus, the levels of logic '0' and logic '1' are not fixed and depend on the actual level of V_{DD}.

If the clock stretching is allowed, the SCL line is pulled down to avoided the overload error during reception and the under load error during transmission.

For example, when in the transmission mode, if the transmit data register is empty and the byte transmit end bit is set ($I2C_STS1.TXDATE = 1$, $I2C_STS1.BSF = 1$), the I²C interface keeps the clock line low before transmission to wait for the software to read STS1 and write the data into the data register (both buffer and shift register are empty); when in the receive mode, if the data register is not empty and the byte transmitting end bit is set ($I2C_STS1.RXDATNE = 1$, $I2C_STS1.BSF = 1$), the I²C interface keeps the clock line low after receiving the data byte, waiting for the software to read STS1, and then read the data register (buffer and shift register are full).

If clock stretching is disable in slave mode, when the receive data register is not empty ($I2C_STS1.RXDATNE = 1$) in the receive mode, and the data has not been read before receiving the next byte, an overrun error will be generated and the last word byte will be discarded. In transmit mode, when the transmit data register is empty ($I2C_STS1.TXDATE = 1$), no new data is written into the data register before the next byte must be transmitted, an underrun error will be generated. The same byte will be transmitted repeatedly. In this case, duplicate write conflicts are not controlled.

17.3.2 Software Communication Process

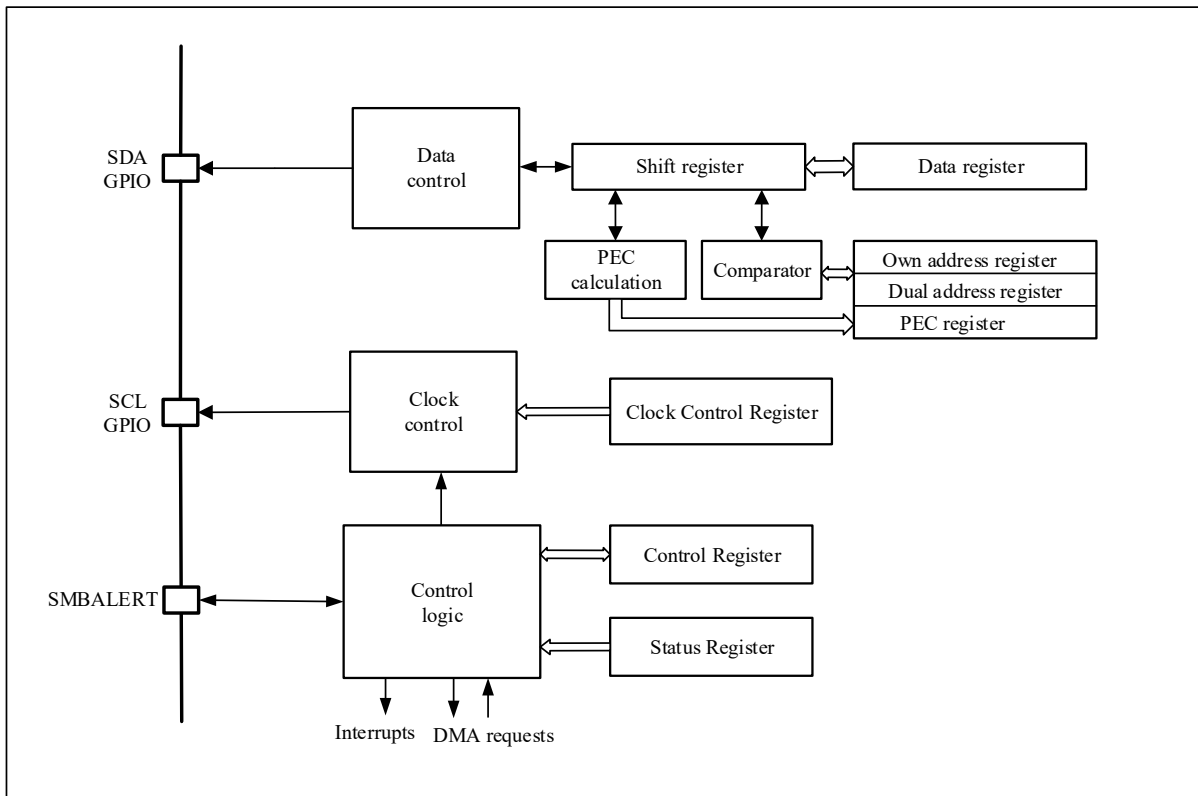
The data transmission of I²C device is divided into master and slave. Master is the device responsible for initializing the transmission of data on the bus and generating clock signal. At this time, any addressed device is a slave. Whether the I²C device is a master or a slave, it can transmit or receive data. Therefore, the I²C interface supports four operation modes:

- Slave transmitter mode
- Slave receiver mode
- Master transmitter mode
- Master receiver mode

After system reset, I²C operates in slave mode by default. The I²C interface is configured by software to transmit a start bit on the bus, then the interface automatically switches from the slave mode to the master mode. When arbitration is lost or a stop signal is generated, the interface will switched to the slave mode from the master mode.

The block diagram of I²C interface is shown in the figure below.

Figure 17-1 I²C Functional Block Diagram

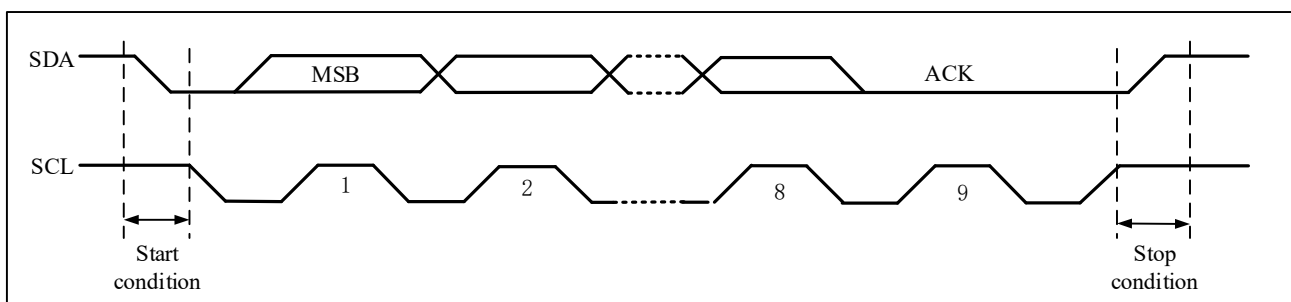


Note: in SMBus mode, SMBALERT is an optional signal. If SMBus is disabled, the signal cannot be used

Start and stop conditions

All data transfers always start with the start bit and end with the stop bit. The start and stop conditions are generated by software in the master mode. Start bit is a level conversion from high to low on SDA line when SCL is high. Stop bit is a level transition from low to high on SDA line when SCL is high. as shown in the figure below.

Figure 17-2 I²C Bus Protocol



Clock synchronization and Arbitration

The I²C module supports multi-master arbitration, which means two masters can transmit I²C start operation concurrently when the bus is idle. Therefore, it is necessary to determine which host obtains control of the bus through some mechanisms, usually achieved through clock synchronization and arbitration.

I²C module has two key features:

- SDA and SCL are open-drain configuration, and the signal wire-AND logic is realized through an external pull-up resistor.
- The SDA and SCL pins not only output signals but also detect the voltage levels on the pins simultaneously to verify if they match the previous output. This provides the hardware basis for clock synchronization and bus arbitration.

The I²C device on the bus is to output logic 0 by grounding the line. Based on the characteristics of the I²C bus, if one device transmits logic 0 and the other transmits logic 1, then only logic 0 is present on the bus, so there is no possibility of level conflicts on the line.

The physical connection of the bus allows the master to read data while writing data to the bus. In this way, when two masters are competing for the bus, the one that transmits logic 0 does not know the occurrence of the competition. Only the one that transmits logic 1 will find the conflict (when writing a logic 1, but read 0) and exit the competition.

Clock synchronization

The high-to-low switching of the SCL line causes the devices to begin counting their low-level periods, and once the device's clock goes low, it keeps the SCL line in this state until the high-level of the clock is reached. However, if another clock is still in the low period, the low-to-high switch of this clock will not change the state of the SCL line. Therefore, the SCL line is kept low by the device with the longest low-level period. A device with a short low-level period will enter a high-level wait state.

When all related devices have counted their low-level periods, the clock line is released and goes high-level, thereafter the clock of the devices and the state of SCL remain consistent, and all devices will begin counting their high-level periods, the device that completes the high-level period first will pull the SCL line low again.

In this way, the low-level period of the generated synchronous SCL clock is determined by the device with the longest low-level clock period, and the high-level period is determined by the device with the shortest high-level clock period.

Arbitration

Arbitration, like synchronization, is to resolve bus control conflicts in multiple masters. The arbitration process has nothing to do with the slave. If the two masters both produce a valid start bit when the bus is idle, in this case, it is necessary to decide which master will complete the data transmission. This is the process of arbitration.

Each master controller does not have the priority level of controlling the bus, which is all determined by arbitration. The bus control is determined and carried out bit by bit. They follow the principle of low level first, that is, whoever transmits the low level first will control the bus. During the arbitration of each bit, when SCL is high, each host checks whether its own SDA level is the same as that transmitted by itself. In theory, if the content transmitted by two hosts is exactly the same, then they can successfully transmit without errors. If a host transmits a high level but detects that the SDA line is low, it considers that it has lost arbitration and shuts down its SDA output driver, while the other host continues to complete its own transmission.

I²C data communication process

Each I²C device is identified by a unique address. According to the device function, they can be either a transmitter or a receiver.

The I²C host is responsible for generating the start bit and the end bit to start and end a transmission. And is responsible for generating the SCL clock.

The I²C module supports 7-bit and 10-bit addresses, and the user can configure the address of the I²C slave through software. After the I²C slave detects the start bit on the I²C bus, it starts to receive the address from the bus, and compares the received

address with its own address. Once the two addresses are matched, the I²C slave will transmit an acknowledgement (ACK) and respond to subsequent commands on the bus: transmit or receive the requested data. In addition, if the software opens a broadcast call, the I²C slave always transmits a confirmation response to a broadcast address (0x00).

Data and address are transmitted in 8-bit width, with the most significant bit first. The 1 or 2 bytes following the start condition is the address (1 byte in 7-bit mode, 2 bytes in 10-bit mode). The address is only transmitted in master mode. During the 9th clock period after 8 clocks of a byte transmission, the receiver must transmit back an acknowledge bit (ACK) to the transmitter, as shown in the Figure 17-2 I2C Bus Protocol.

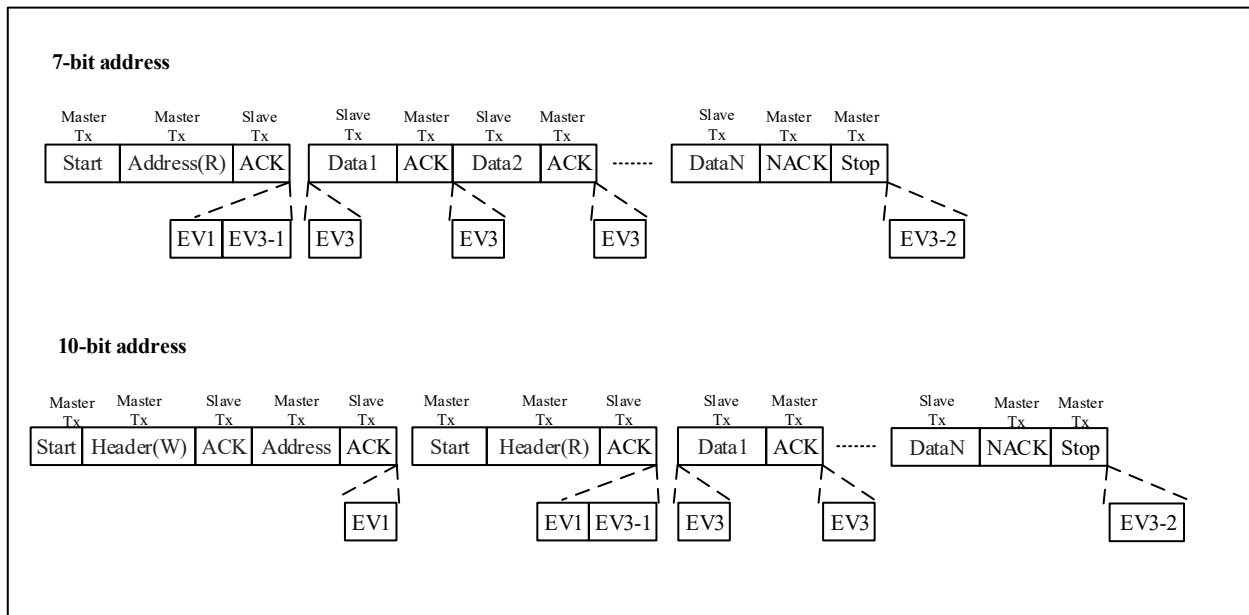
Software can enable or disable acknowledgement (ACK), and can set the I²C interface address (7-bit, 10-bit address or general call address).

I²C slave transmission mode

In slave mode, the transmission reception flag bit (I2C_STS2.TRF) indicates whether it is currently in receiver mode or transmission mode. When transmitting data to I²C bus in transmission mode, the software should follow the following steps:

1. First, enable I²C peripheral clock and configure the clock related register in I2C_CTRL1, ensuring the correct I²C timing. After these two steps are completed, I²C runs in slave mode, waiting for receiving start bit and address.
2. I²C slave receives a start bit first, and then receives a matching 7-bit or 10-bit address. I²C hardware will set the I2C_STS1.ADDRF(received address and matched its own address). The software should monitor this bit regularly or have an interrupt to monitor this bit. After this bit is set, the software reads I2C_STS1 register and then read I2C_STS2 register to clear the I2C_STS1.ADDRF bit. If the address is in 10 bit format, the I²C master should generate a START and transmit an address header to the I²C bus. After detecting START and the following address header, the slave will continue to set I2C_STS1.ADDRF bit. The software continues to read I2C_STS1 register and read I2C_STS2 register to clear the I2C_STS1.ADDRF bit a second time.
3. I²C enters the data transmitting state, and now shift register and data register I2C_DAT are all empty, so the hardware will set the I2C_STS1.TXDATE(transmit data empty). At this time, the software can write the first byte data to I2C_DAT register, however, because the byte of the I2C_DAT register is immediately moved into the internal shift register, the I2C_STS1.TXDATE bit is not cleared to zero. When the shift register is not empty, I²C starts to transmit data to I²C bus.
4. During the transmission of the first byte, the software writes the second byte to I2C_DAT, neither the I2C_DAT register nor the shift register is empty. The I2C_STS1.TXDATE bit is cleared to 0.
5. After the first byte is transmitted, I2C_STS1.TXDATE is set again, and the software writes the third byte to I2C_DAT, the same time, the I2C_STS1.TXDATE bit is cleared. After that, as long as there is still data to be transmitted and I2C_STS1.TXDATE is set to 1, the software can write a byte to I2C_DAT register.
6. During the transmission of the penultimate byte, the software writes the last data to the I2C_DAT register to clear the I2C_STS1.TXDATE flag bit, and then the I2C_STS1.TXDATE status is no longer concerned. The I2C_STS1.TXDATE bit is set after the penultimate byte is transmitted until the stop end bit is detected.
7. According to the I²C protocol, the I²C master will not transmit a ACK to the last byte received. Therefore, after the last byte is transmitted, the I2C_STS1.ACKFAIL bit (acknowledge fail) of the I²C slave will be set to notify the end of transmission to the software. The software writes 0 to the I2C_STS1.ACKFAIL bit to clear this bit.

Figure 17-3 Slave Transmitter Transfer Sequence Diagram



Instructions:

- EV1: I2C_STS1.ADDRF = 1, read STS1 and then read STS2 to clear the event.
- EV3-1: I2C_STS1.TXDATE=1, shift register is empty, data register is empty, write DAT.
- EV3: I2C_STS1.TXDATE=1, shift register is not empty, data register is empty, write DAT will clear the event.
- EV3-2: I2C_STS1.ACKFAIL=1, ACKFAIL bit of STS1 register write "0" to clear the event.

Notes:

- (1) EV1 and EV3_1 event prolongs the low SCL time until the end of the corresponding software sequence.
- (2) The software sequence of EV3 must be completed before the end of the current byte transfer.

I²C slave receiving mode

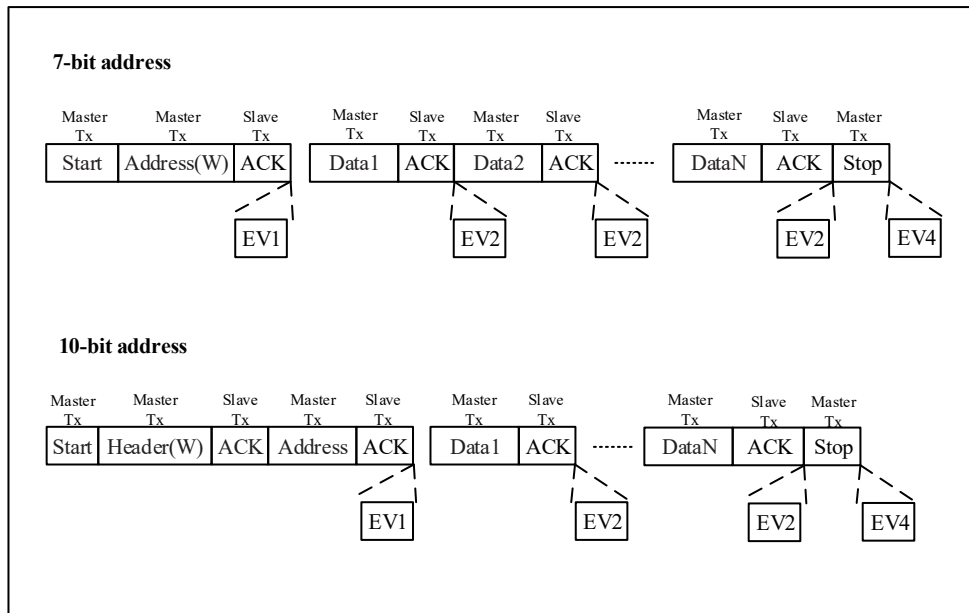
When receiving data in slave mode, the software should follow these steps:

1. First, enable I²C peripheral clock and configure the clock related register in I2C_CTRL1 ensuring the correct I²C timing. After these two steps are completed, I²C runs in slave mode, waiting for receiving start bit and address.
2. After receiving the START condition and the matched 7-bit or 10-bit address, I²C hardware will set I2C_STS1.ADDRF bit(the address received and matched with its own address) to 1. This bit should be detected by software polling or interrupt. After it is found that it is set, the software clears the I2C_STS1.ADDRF bit by first reading I2C_STS1 register and then reading the I2C_STS2 register. Once the I2C_STS1.ADDRF bit is cleared, the I²C slave starts to receive data from the I²C bus.
3. When the first byte is received, the I2C_STS1.RXDATNE bit (the received data is not empty) is set to 1 by hardware. If the I2C_CTRL2.EVTINTEN and I2C_CTRL2.BUFINTEN bits are set, an interrupt is generated. The software should check this bit by polling or interrupt. Once it is found that it is set, the software can read the first byte of I2C_DAT register, and then the I2C_STS1.RXDATNE bit is cleared to 0. Note that if the I2C_CTRL1.ACKEN bit is set, the slave should generate a response pulse after receiving a byte.
4. At any time, as long as the I2C_STS1.RXDATNE bit is set to 1, the software can read a byte from the I2C_DAT

register. When the last byte is received, I2C_STS1.RXDATNE is set to 1 and the software reads the last byte.

- When the slave detects the STOP bit on I2C bus, set I2C_STS1.STOPF to 1, and if the I2C_CTRL2.EVTINTEN bit is set, an interrupt will be generated. The software clears the I2C_STS1.STOPF bit by first reading the I2C_STS1 register and then writing the I2C_CTRL1 register (refer to EV4 in the following figure).

Figure 17-4 Slave Receiver Transfer Sequence Diagram



Instructions:

- EV1: I2C_STS1.ADDRF = 1, read STS1 and then read STS2 to clear the event.
- EV2: I2C_STS1.RXDATNE =1, reading DAT will clear this event.
- EV4: I2C_STS1.STOPF=1, reading STS1 and then writing the CTRL1 register will clear this event.

Notes:

- EV1 event prolongs the time when SCL is low until the end of the corresponding software sequence.
- The software sequence of EV2 must be completed before the end of the current byte transmission.

I²C master transmission mode

In the master mode, the I²C interface starts data transmission and generates a clock signal. Serial data transmission always starts with a start condition and ends with a stop condition. When the START condition is generated on the bus through the start bit, the device enters the master mode.

When transmitting data to I²C bus in master mode, the software should operate as follows:

- First, enable the I²C peripheral clock, and configure the clock-related registers in I2C_CTRL1 to ensure the correct I²C timing. When these two steps are completed, I²C runs in the slave mode by default, waiting for receiving the start bit and address.
- When BUSY=0, I2C_CTRL1.STARTGEN bit set to 1, and the I²C interface will generate a start condition and switch to the master mode (I2C_STS2.MSMODE=1).
- Once the start condition is transmitted, I²C hardware will set I2C_STS1.STARTBF bit (START bit flag) and then enters

the master mode. If the I2C_CTRL2.EVTINTEN bit is set, an interrupt will be generated. Then the software reads the I2C_STS1 register and then writes a 7-bit address bit or a 10-bit address bit with an address header to the I2C_DAT register, in order to clear the I2C_STS1.STARTBF bit. After the I2C_STS1.STARTBF bit is cleared to 0, I²C starts transmitting addresses or address headers to I²C bus.

In 10-bit address mode, transmitting a header sequence will generate the following events:

- I2C_STS1.ADDR10F bit is set by hardware, and if I2C_CTRL2.EVTINTEN bit is set, an interrupt is generated. Then the master reads the STS1 register, and then writes the second address byte into the DAT register.
- I2C_STS1.ADDRF bit is set by hardware, and if I2C_CTRL2.EVTINTEN bit is set, an interrupt is generated. Then the master reads the STS1 register, followed by reading the STS2 register.

Note: In the transmitter mode, the master device first transmits the header byte (11110xx0) and then transmits the lower 8 bits of the slave address. (where xx represents the highest 2 bits of the 10-bit address).

In the 7-bit address mode, only one address byte needs to be transmitted out. Once the address byte is transmitted out:

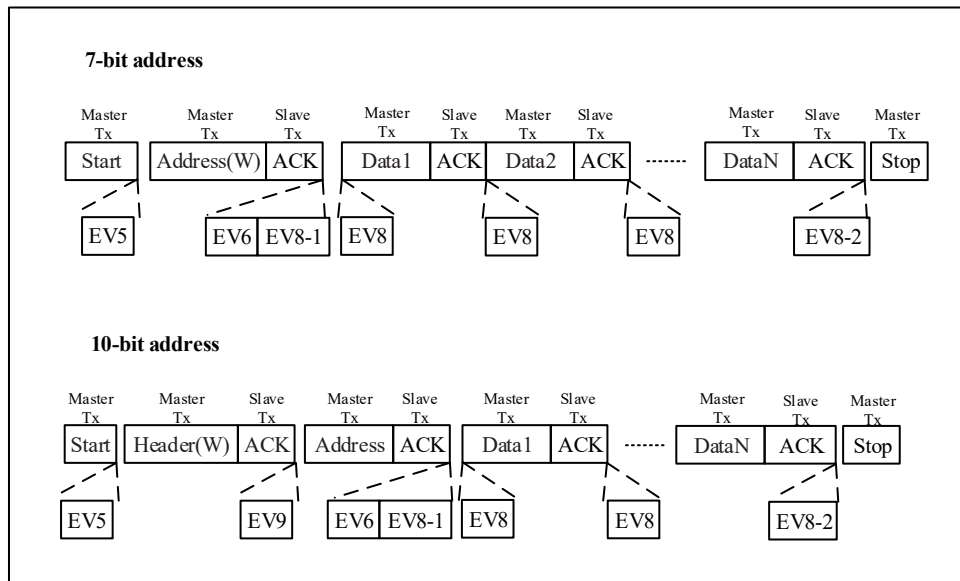
- I2C_STS1.ADDRF bit is set by hardware, and if I2C_CTRL2.EVTINTEN bit is set, an interrupt is generated. Then the master device waits for reading the STS1 register once, followed by reading the STS2 register.

Note: in the transmitter mode, when the master transmits the slave address, set the lowest bit to "0".

Note: In the 7-bit address mode, do not configure the slave address as 0xF0 to prevent inadvertent hardware setting of the I2C_STS1.ADDR10F bit.

4. After the 7-bit or 10-bit address bit is transmitted, the I²C hardware sets the I2C_STS1.ADDRF bit (address has been transmitted) to 1, if the I2C_CTRL2.EVTINTEN bit is set, an interrupt is generated, and the software is cleared by first reading the I2C_STS1 register and then reading the I2C_STS2 register I2C_STS1.ADDRF.
5. I²C enters the data transmission state. Because the shift register and the data register (I2C_DAT) are empty, the hardware sets the I2C_STS1.TXDATE bit (transmission data empty) to 1, and then the software writes the first byte of data to the I2C_DAT register, but because the byte written into the I2C_DAT register is immediately moved into the internal shift register, the I2C_STS1.TXDATE bit will not be cleared at this time. Once the shift register is not empty, I²C starts transmitting data to the bus.
6. During the transmission of the first byte, the software writes the second byte to I2C_DAT, and I2C_STS1.TXDATE is cleared at this time. At any time, as long as there is data waiting to be transmitted and the I2C_STS1.TXDATE bit is set to 1, the software can write a byte to the I2C_DAT register.
7. In the process of transmitting the penultimate byte, the software writes the last byte of data to I2C_DAT to clear the I2C_STS1.TXDATE flag bit. After that, there is no need to care about the status of the I2C_STS1.TXDATE bit. The I2C_STS1.TXDATE bit will be set after the penultimate byte is transmitted, and will be cleared when the stop bit (STOP) is transmitted.
8. After the last byte is transmitted, because the shift register and the I2C_DAT register are empty at this time, the I²C host sets the I2C_STS1.BSF bit (byte transmission end), and the I²C interface will keep SCL low before clearing the I2C_STS1.BSF bit. After reading I2C_STS1, writing to the I2C_DAT register will clear the I2C_STS1.BSF bit. The software sets the I2C_CTRL1.STOPGEN bit at this time to generate a stop condition, and then the I²C interface will automatically return to the slave mode (I2C_STS2.MSMODE bit is cleared).

Figure 17-5 Master Transmitter Transfer Sequence Diagram



Instructions:

- EV5: I2C_STS1.STARTBF = 1, read STS1 and then write the address to the DAT register will clear the event.
- EV6: I2C_STS1.ADDRF = 1, read STS1 and then STS2 to clear the event.
- EV8_1: I2C_STS1.TXDATE = 1, shift register is empty, data register is empty, write DAT register.
- EV8_2: I2C_STS1.TXDATE = 1, shift register is not empty, data register is empty, write to DAT register will clear the event.
- EV8_2: I2C_STS1.TXDATE = 1, I2C_STS1.BSF = 1, request to set stop bit. These two events are cleared by the hardware when a stop condition is generated.
- EV9: I2C_STS1.ADDR10F = 1, read STS1 and then write to DAT register to clear the event.

Notes:

- (1) EV5, EV6, EV9, EV8_1 and EV8_2 event prolonged the low SCL time until the end of the corresponding software sequence.
- (2) The software sequence of EV8 must be completed before the end of the current byte transfer.
- (3) When I2C_STS1.TXDATE or I2C_STS1.BSF bit is set, stop condition should be arranged when EV8_2 occurs.

I2C master receiving mode

In master mode, software receiving data from I²C bus should follow the following steps:

1. First, enable the I²C peripheral clock and configure the clock-related registers in I2C_CTRL1, in order to ensure that the correct I²C timing is output. After enabling and configuring, I2C runs in slave mode by default, waiting to receive the start bit and address.
2. When BUSY=0, set the I2C_CTRL.STARTGEN bit, and the I²C interface will generate a start condition and switch to the master mode (I2C_STS2.MSMODE bit is set to 1).
3. Once the start condition is transmitted, the I²C hardware sets I2C_STS1.STARTBF(start bit flag) and enters the host

mode. If the I2C_CTRL2.EVTINTEN bit is set to 1, an interrupt will be generated. Then the software reads the I2C_STS1 register and then writes a 7-bits address or a 10-bits address with an address header to the I2C_DAT register, in order to clear the I2C_STS1.STARTBF bit. After the I2C_STS1.STARTBF bit is cleared to 0, I²C begins to transmit the address or address header to the I²C bus.

In 10-bits address mode, transmitting a header sequence will generate the following events:

- The I2C_STS1.ADDR10F bit is set to 1 by hardware, and if the I2C_CTRL2.EVTINTEN bit is set to 1, an interrupt will be generated. Then the master device reads the STS1 register, and then writes the second byte of address into the DAT register.
- The I2C_STS1.ADDRF bit is set to 1 by hardware, and if the I2C_CTRL2.EVTINTEN bit is set to 1, an interrupt will be generated. Then the master device reads the STS1 register and read STS2 register.

Note: In the receiver mode, the master device sends the header byte (11110xx0) firstly, then sends the lower 8 bits of the slave address, and then resends a start condition followed by the header byte (11110xx1) (where xx represents the highest 2 digits of the 10-bits address).

In the 7-bits address mode, only one address byte needs to be sent, once the address byte is sent:

- The I2C_STS1.ADDRF bit is set to 1 by hardware, and if the I2C_CTRL2.EVTINTEN bit is set to 1, an interrupt will be generated. Then the master device waits to read the STS1 register, and then reads the STS2 register.

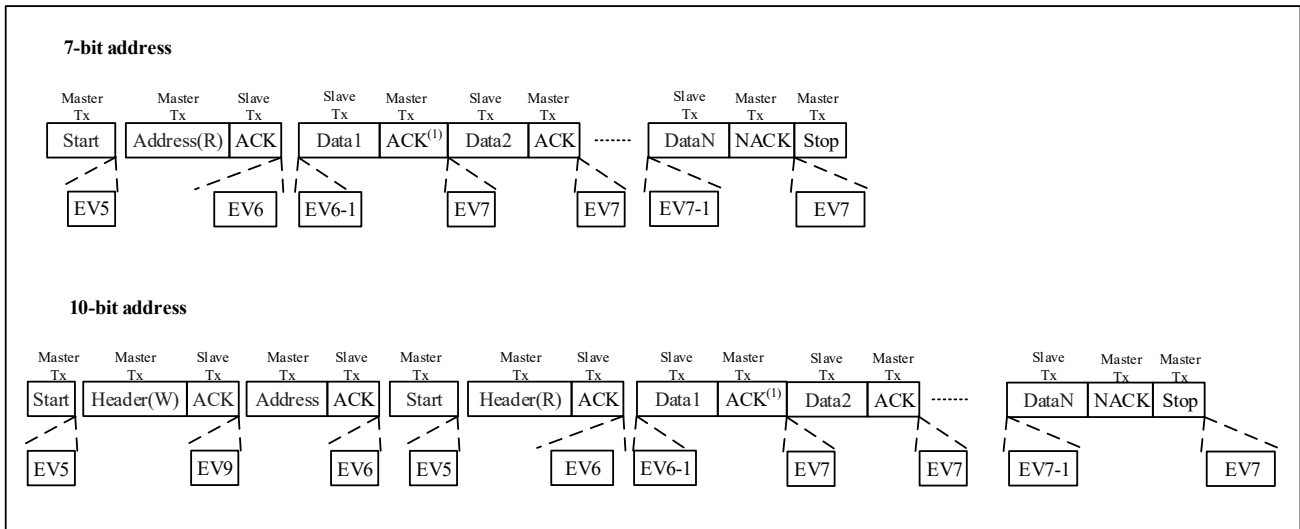
Note: In the receiving mode, the master device sets the lowest bit as '1' when transmitting the slave address.

4. After the 7-bits or 10-bits address is sent, the I²C hardware sets the I2C_STS1.ADDRF bit (address has been sent) to 1. If the I2C_CTRL2.EVTINTEN bit is set to 1, an interrupt will be generated. The software clears the I2C_STS1.ADDRF bit by first reading the I2C_STS1 register and then reading the I2C_STS2 register. If in the 10-bit address mode, software should set the I2C_CTRL1.STARTGEN bit again to regenerate a START. After the START is generated, the I2C_STS1.STARTBF bit will be set. The software should clear the I2C_STS1.STARTBF bit by reading I2C_STS1 firstly and then writing the address header to I2C_DAT, and then the address header is sent to the I²C bus, I2C_STS1.ADDRF is set to 1 again. The software should clear the I2C_STS1.ADDRF bit again by reading I2C_STS1 firstly and then reading I2C_STS2.
5. After transmitting the address and clearing the I2C_STS1.ADDRF bit, the I²C interface enters the host receiving mode. In this mode, the I²C interface receives data bytes from the SDA line and transmits them to the DAT register through the internal shift register. Once the first byte is received, the hardware will set the I2C_STS1.RXDATNE bit (not empty flag bit of received data) to 1, and if the I2C_CTRL1.ACKEN bit is set to 1, an acknowledge pulse will be sent. At this time, the software can read the first byte from the I2C_DAT register, and then the I2C_STS1.RXDATNE bit is cleared to 0. After that, as long as I2C_STS1.RXDATNE is set to 1, the software can read a byte from the I2C_DAT register.
6. The master device transmits a NACK after receiving the last byte from the slave device. After receiving the NACK, the slave device releases the control of SCL and SDA lines; the master device can transmit a stop/restart condition. In order to generate a NACK pulse after receiving the last byte, the software should clear the I2C_CTRL1.ACKEN bit immediately after receiving the penultimate byte (N-1). In order to generate a stop/restart condition, the software must set the I2C_CTRL1.STOPGEN bit or I2C_CTRL1.STARTGEN to 1 after reading the penultimate data byte. This process needs to be completed before the last byte is received to ensure that the NACK is sent for the last byte.
7. After the last byte is received, the I2C_STS1.RXDATNE bit is set to 1, and the software can read the last byte. Since I2C_CTRL1.ACKEN has been cleared to 0 in the previous step, I2C no longer transmits ACK for the last byte, and generates a STOP bit after the last byte is sent.

Note: The above steps require the number of bytes N>1. If N=1, step 6 should be executed after step 4, and it needs to be

completed before the reception of byte is completed.

Figure 17-6 Master Receiver Transfer Sequence Diagram



Instructions:

- EV5: I2C_STS1.STARTBF=1, reading STS1 and then writing the address into the DAT register will clear this event.
- EV6: I2C_STS1.ADDRF=1, reading STS1 and then reading STS2 will clear this event. In the 10-bits master receiving mode, the I2C_CTRL1.STARTGEN should be set to 1 after this event.
- EV6_1: There is no corresponding event flag, only suitable for receiving 1 byte. Just after EV6 (that is after clearing I2C_STS1.ADDRF), the generation bits for acknowledge and stop condition should be cleared.
- EV7: I2C_STS1.RXDATNE=1, read the DAT register to clear this event.
- EV7_1: I2C_STS1.RXDATNE =1, read the DAT register to clear this event. Set I2C_CTRL1.ACKEN=0 and I2C_CTRL1.STOPGEN=1.
- EV9: I2C_STS1.ADDR10F=1, reading STS1 and then writing to the DAT register will clear this event.

Notes:

- (1) If a single byte is received, it is NA.
- (2) EV5, EV6, and EV9 events stretch the low level of SCL until the corresponding software sequence ends.
- (3) The EV7 software sequence shall be completed before the end of the current byte transmission.
- (4) The software sequence of EV6_1 or EV7_1 shall be completed before the ACK pulse of the current transmission byte.

17.3.3 Error Conditions Description

I2C errors mainly include bus error, acknowledge error, arbitration loss, overload\ underload error. These errors may cause communication failure.

Acknowledge Failure(ACKFAIL)

The interface have a acknowledge bit is detected that does not match the expectation, it will occurs acknowledge fail error, I2C_STS1.ACKFAIL bit is set. An interrupt is generated when I2C_CTRL2.ERRINTEN bit is set to 1.

When transmitter receives a NACK, The communication must be reset: Device in slave mode, hardware release the bus; Device in master mode, it must generate a stop condition from software.

Bus Error (BUSERR)

when address or data is being transmitted, I²C interface receive external stop or start condition, it will generate a bus error and I2C_STS1.BUSERR bit is set. An interrupt is generated when I2C_CTRL2.ERRINTEN bit is set to 1.

I²C device as master, the hardware does not release bus, at the same time it does not affect the current status of transfer. The current transfer will be determined by software whether suspend.

I²C device as slave, when data is discarded in transmission and the bus releases by hardware, it will have two situation: If an error start condition is detected, the slave device considers a restart condition and waits for an address or a stop condition. If an error stop condition is detected, the slave device operates as a normal stop condition and the hardware releases the bus.

Arbitration Lost (ARLOST)

The interface has arbitration lost is detected, hardware release the bus, it will generate an arbitration lost error, I2C_STS1.ARLOST bit is set. An interrupt is generated when I2C_CTRL2.ERRINTEN bit is set to 1.

I²C interface will go to slave mode automatically (I2C_STS2.MSMODE bit is cleared). When the I²C interface lost the arbitration, it can not respond to its slave address in the same communication, but it can respond when the master wins the bus retransmits a start signal.

Overrun/Underrun Error (OVERRUN)

In slave mode, if clock stretching is disabled, overrun/underrun Errors can easily occur:

When I²C interface is receiving data (I2C_STS1.RXDATNE=1, data have received in register), and I2C_DAT register still has previous byte which has not been read, it will generate an overrun error. In this situation, the last received data is discarded. And software should clear I2C_STS1.RXDATNE bit, transmitter retransmit last byte.

When I²C interface is transmitting data (I2C_STS1.TXDATE=1, new data has not transmitting to register), and I2C_DAT register still empty, it will generate an underrun error. In this situation, the previous byte in the I2C_DAT register is sent repeatedly. And User make sure that in the event of an underrun error, the receiver discard repeatedly byte, and transmitter should update the I2C_DAT register at the specified time according to the I²C bus standard.

In transmitting the first byte, I2C_DAT register must be written after I2C_STS1.ADDRF bit is cleared and before the first SCL rising edge. If cannot make sure do that, the first byte should be discard by receiver.

17.3.4 DMA Application

The I²C can generate a DMA requests when transfer data register empty or full. DMA can write data to I²C or read data from I²C reduce the CPU overload.

DMA requests must be responded before the current byte transfer ends. If the data transfers set by the corresponding DMA channel is completed, DMA will transmit EOT(End Of Transmission) to I²C, and generates a interrupt when enable interrupt bit.

In the master transmit mode, in EOT interrupt handler, DMA request need to be disable, and set stop condition after waiting for I2C_STS1.BSF event.

In the master receive mode, when the number of data to be received is greater than or equal to 2, DMA will transmit a hardware signal EOT_1 in DMA transmission(byte number-1). If the I2C_CTRL2.DMALAST bit is set, the hardware will automatically transmit a NACK after transmitting the next byte following EOT_1. The user can set a stop condition in the interrupt handler after the DMA transfer is completed if interrupt is allowed.

Transmit process

DMA mode is enabled by setting the I2C_CTRL2.DMAEN bit. When I2C_STS1.TXDATE bit is set, the data will be loaded into the I2C_DAT from a preset memory block by DMA. Configure the DMA channel for I²C transmission, (x is the channel number) the following steps must be operate:

1. In the DMA_PADDRx register set the I2C_DAT register address. Data will be send to this address from memory in every I2C_STS1.TXDATE event.
2. In the DMA_MADDRx register set the memory address. Data will transmit to I2C_DAT from memory in every I2C_STS1.TXDATE event.
3. In the DMA_TXNUMx register set the number of byte need to be transferred. In every I2C_STS1.TXDATE event this number-1 until 0.
4. In the DMA_CHCFGx register set PRIOLVL[1:0] bit to configure the priority of channel.
5. In the DMA_CHCFGx register set DIR bit to configure when generate an interrupt whether send a half data or all completed.
6. In the DMA_CHCFGx register set CHEN bit to enable transfer channel.
7. When DMA transfer data is done, DMA need send a EOT/EOT_1 signal to I²C indicate this transfer is done. If interrupt is enable, DMA generate a interrupt.

Note: if DMA is used for transmission, do not set I2C_CTRL2.BUFINTEN bit.

Receive process

DMA mode is enabled by setting I2C_CTRL2.DMAEN bit. When data byte is received, DMA will transmit I²C data to memory block, set DMA channel for I²C reception. The following steps must be operate:

1. In DMA_PADDRx register set the address of the I2C_DAT register. In every I2C_STS1.RXDATEN event, data will transmit from address to memory block.
2. In DMA_MADDRx register set the memory block address. In every I2C_STS1.RXDATEN event, data will transmit from I2C_DAT register to memory block.
3. In DMA_TXNUMx register set the number of data need to be transferred. In every I2C_STS1.RXDATEN event the number-1 until 0.
4. In DMA_CHCFGx register set PRIOLVL[0:1] to configure the priority of channel.
5. In DMA_CHCFGx register clear DIR to configure when generate a interrupt request whether received half data or all data is received.
6. In the DMA_CHCFGx register set CHEN bit to activate the channle.
7. When DMA tansfer data is done, DMA need to transmit EOT/EOT_1 signal to I²C indicate this transfer is done, if interrupt is enbale, DMA generate a interrupt.

Note: If DMA is used for receiving, do not set I2C_CTRL2.BUFINTEN bit.

17.3.5 Packet Error Check

Enabling the PEC function by setting the I2C_CTRL1.PECEN bit to 1. PEC uses CRC-8 algorithm to calculate all information bytes including address and read/write bits. it can improve the reliability of communication. The CRC-8

polynomial used by the PEC calculator is $C(x) = x^8 + x^2 + x + 1$.

In transmit mode, software sets I2C_CTRL1.PEC transfer bit in the last I2C_STS1.TXDATE event, then PEC will be transferred in the last byte. In receive mode, software sets I2C_CTRL1.PEC transfer bit after the last I2C_STS1.RXDATE event, then receives the PEC byte and compares the received PEC byte to the internally calculated PEC value. If it is not equal to the internally calculated PEC value, the receiver needs to transmit a NACK. If it is in master receiver mode, NACK will be sent after PEC regardless of the calculated result. It should pay attention that I2C_CTRL1.PEC bit has to be set before receiving the ACK pulse of the current byte.

If both DMA and PEC calculator are activated, I²C will automatically transmit or check the PEC value.

In transmit mode, when I²C interface receives EOT signal from DMA controller, it will automatically transmit PEC following the last byte. In receive mode, when I²C interface receives an EOT_1 signal from DMA, it will automatically consider the next byte as PEC and compare it with the internally calculated PEC. It will generate a DMA request after receiving PEC.

To allow intermediate PEC transmission, I2C_CTRL2.DMALAST bit is used to determine whether it is the last DMA transfer. And if it does the last DMA request of the master receiver, NACK will be sent automatically after receiving the last byte.

When arbitration is lost, PEC calculation is invalid.

17.3.6 Smbus

Introduction

The system management bus (SMBus or SMB) is a simple single-ended two-wire bus structure. Using SMBus can communicate with other devices or other parts of the system. SMBus is a derivative of the I²C bus and provides a control bus for system and power management related tasks. If you want browse more information, please refer to the SMBus specification v2.0(<http://smbus.org/specs/>).

SMBus have three types of device standard.

- Master: device transmits commands, generates clocks and stops transmissions;
- Slave: device receives and responds to commands;
- Host: a system have only one host. it provides a master interface with system CPU. host has master-slave capabilities, it supports SMBus alert protocol.

Similarities between SMBus and I²C:

- Both bus protocols contain of 2 wires (a clock wire SCL and a data wire SDA), with an optional SMBus alert wire.
- The data format is similar. SMBus data format is similar to 7-bit address format of I²C(refer to Figure 17-2).
- Both are master-slave communication modes, and the master device provides the clock.
- Both support multi master

Differences between SMBus and I²C:

Table 17-1 Comparison Between Smbus and I²C

SMBus	I ² C
Maximum transmission speed 100kHz	Maximum transmission speed 1MHz
Minimum transmission speed 10kHz	No minimum transmission speed
Low clock timeout 35ms	No clock timeout

SMBus	I ² C
Fixed logic level	V _{DD} determined logic level
Different address types (reserved, dynamic, etc.)	7-bit, 10-bit, and broadcast call slave address types
Different bus protocols (quick command, call handling, etc.)	No bus protocol

SMBus usage

SMBus uses the system management bus to realize lightweight communication requirements. Generally, SMBus is commonly used on the computer motherboard. It is mainly used to transmit ON/OFF instructions for power unit and provide a control bus for system and power management.

Device identification

In SMBus, any device as a slave device only has a address, named slave address.

To distribute address for each devices, it must have a unique device identifier(UDID) to distinguish devices.

Bus protocol

SMBus technical specification includes eight bus protocols. For detailed information and address types regarding SMBus, please refer to the SMBus specification v2.0(<http://smbus.org/specs/>). User can devicmine what protocols are implemented.

Every packet through the SMBus complies with the SMBus protocol predefined format. SMBus is a subset of the data transfer format of I²C specification. As long as an I²C device can be accessed through one of the SMBus protocols, it is considered to be SMBus compliant.

Note: SMBus does not support quick command protocol.

Address resolution protocol (ARP)

The SMBus resolves address conflicts by dynamically assigning a new unique address to each slave device. This is the address resolution protocol(ARP) .

Any master device can connected bus to access all devices.

SMBus physical layer arbitration is used to distribute addresses. When device power on, the device's distribute address is not change, the protocol allows address retain when device power off.

When address is distributed, there is no extra SMBus packaging cost(the cost time that access distribute address device and access fixed address device is same).

Timeout function

SMBus has a timeout feature: if it has taken too long time during the communication, it automatically resets the device. This is the reason why SMBus has a minimum transmission rate limitation to prevent the bus from locking up for a long time after the timeout occurs. I²C bus is essentially a "DC" bus, that is to say, if the slave is executing some subroutines and cannot respond in time while the master is accessing the slave, it can hold the clock. That can remind the host that the slave is busy but does not want to give up the current communication. This session can continue after the current task of the slave is over. I²c doesn't have a maximum limitation for the delay, but it is limited to 35ms in the SMBus system. According to the SMBus protocol, if a session takes too long, it means something is wrong with the bus, and all devices should be reset to eliminate this state. Like this, the slave device is not allowed to pull the clock down for too long. I2C_STS1.TIMOUT bit indicates the status of this feature.

SMBus alter mode

SMBus offers a optional interrupt signal SMBALERT(like SCL and SDA,is a wire-AND signal) that devices uses to extend their control capabilities at expense of a pin. SMBus broadcast call address often combine with SMBALERT. There are 2 bytes message about SMBus.

A device which only has slave function can set I2C_CTRL1.SMBALERT bit to indicate it want to communicate with host. The host handles the interrupt and accesses all SMBALERT devices through the ARA (Alert Response Address, address value 0001100x). Only those devices that pull SMBALERT low can respond to ARA. This state is identified by the I2C_STS1.SMBALERT. The 7-bit device address provided from the transmitting device is placed on the 7 most significant bits of the byte, the eighth bit can be either '0' or '1'.

When more than one device's SMBALERT is low, the highest priority(The smaller the address, the higher the priority) can win bus communication through the standard arbitration during address transmission. If confirming the slave address, device's SMBALERT is no longer pulled low. If message transmitted completely,device's SMBALERT still is low,it mean host will read ARA again. The host can periodically access the ARA when the SMBALERT signal is not used.

SMBus communication process

The communication process on SMBus is similar to that on I2C.To use the SMBus mode, you need to configure SMBus specific registers in the program, respon to SMBus specific flag, and implement the upper-layer protocols described in the SMBus manual.

1. At first, set I2C_CTRL1.SMBMODE bit, and configure I2C_CTRL1.SMBTYPE bit and I2C_CTRL1.ARPEN bit according to the application requirements. If I2C_CTRL1.ARPEN=1 and I2C_CTRL1.SMBTYPE=0, use the default address of the SMB device. If I2C_CTRL1.ARPEN=1 and I2C_CTRL1.SMBTYPE=1, use the SMB master header field.
2. To support ARP (I2C_CTRL1.ARPEN=1), in SMBus host mode (I2C_CTRL1.SMBTYPE=1), software needs to respond to the I2C_STS2.SMBHADDR bit (in SMBus slave mode, respond to I2C_STS2.SMBDADDR bit) and implement the functions according to the ARP protocol.
3. To support the SMBus alert mode, software should respond to the I2C_STS1.SMBALERT bit and implement the corresponding functions.

17.4 Debug Mode

When the microcontroller enters the debug mode (Cortex®-M0 core is in the stop state), configure the DBG_CTRL.I2CxTIMOUT bit in the PWR module, select SMBUS timeout to continue normal operate or stop. Refer to section 3.3.2 for details.

17.5 Interrupt Request

All I²C interrupt requests are listed in the following table.

Table 17-2 I²C Interrupt Request

Interrupt Function	Interrupt Event	Event Flag	Set Control Bit
I2C global interrupt	Start bit sent (master)	STARTBF	EVTINTEN
	Address sent (master) or address matched (slave)	ADDRF	

Interrupt Function	Interrupt Event	Event Flag	Set Control Bit
	10-bit header sent (master)	ADDR10F	
	Received stop (slave)	STOPF	
	Data byte transfer completed.	BSF	
	Receive buffer is not empty.	RXDATNE	EVTINTEN and BUFINTEN
	Transmit buffer is empty.	TXDATE	
	Bus error	BUSERR	ERRINTEN
	Lost arbitration (master)	ARLOST	
	Acknowledge fail	ACKFAIL	
	Overrun/underrun	OVERRUN	
	PEC error	PECERR	
	Timeout /Tlow error	TIMOUT	
	SMBus Alert	SMBALERT	

Notes:

- (1) *STARTBF, ADDR, ADDR10F, STOPF, BSF, RXDATNE and TXDATE* are merged into a interrupt channel through logical OR.
- (2) *BUSERR, ARLOST, ACKFAIL, OVERRUN, PECERR, TIMEOUT and SMBALERT* are merged into a interrupt channel through logical OR.
- (3) *Event interrupts and error interrupts* are logically ORed into the global interrupt channel.

17.6 I²C Registers

All register operations must be performed in half word (16 bits) or word (32 bits)

17.6.1 I²C Register Overview

Table 17-3 I²C Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
000h	I2C_CTRL1	Reserved																SWRESET	Reserved	SMBALERT	PEC	ACKPOS	ACKEN	STOPGEN	STARTGEN	NOEXTEND	GCEN	PECEN	ARPEN	SMBTYPE	Reserved	SMBMODE	EN		
	Reset Value																	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
004h	I2C_CTRL2	Reserved																			DMALAST	DMAEN	BUFINTEN	EVTINTEN	ERRINTEN	Reserved	CLKFREQ[5:0]								
	Reset Value																				0	0	0	0	0		0	0	0	0	0	0	0	0	0
008h	I2C_OADDR1	Reserved																ADDRMODE	Reserved	Reserved						ADDR[9:8]	ADDR[7:1]						ADDR0		
	Reset Value																	0								0	0	0	0	0	0	0	0	0	0
00Ch	I2C_OADDR2	Reserved																Reserved						ADDR2[7:1]						DUALEN					
	Reset Value																							0	0	0	0	0	0	0	0	0	0	0	0
010h	I2C_DAT	Reserved																Reserved						DATA[7:0]											
	Reset Value																							0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
014h	I2C_STS1	Reserved																SMBALERT	TIMOUT	Reserved	PECERR	OVERRUN	ACKFAIL	ARLOST	BUSERR	TXDATE	RXDATNE	Reserved	STOPF	ADDR10F	BSF	ADDRF	STARTBF	0						
	Reset Value																	0	0		0	0	0	0	0	0	0	0		0	0	0	0	0	0	0				
018h	I2C_STS2	Reserved																PECVAL[7:0]							DUALFLAG	SMBHADDR	SMBDADDR	GCALLADDR	Reserved	TRF	BUSY	MSMODE	0							
	Reset Value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
01Ch	I2C_CLKCTRL	Reserved																FSMODE	DUTY	Reserved	CLKCTRL[11:0]											0	0	0	0	0	0	0	0	0
	Reset Value																	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
020h	I2C_TMRISE	Reserved																							TMRISE[5:0]					0	0	0	0	0	0					
	Reset Value																								0	0	0	0	0	0										

17.6.2 I²C Control Register 1 (I2C_CTRL1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SW RESET	Reserved	SMB ALERT	PEC	ACK POS	ACKEN	STOP GEN	START GEN	NO EXTEND	GCEN	PECEN	ARPEN	SMB TYPE	Reserved	SMB MODE	EN
rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw

Bit Field	Name	Description
15	SWRESET	Software reset Make sure the I2C bus is idle before resetting this bit. 0:I2C not reset; 1:I2C reset. <i>Note: This bit can be used when the I2C_STS2.BUSY bit is set to 1 and no stop condition is detected on the bus.</i>
14	Reserved	Reserved, the reset value must be maintained
13	SMBALERT	SMBus alert It can be set or cleared by software. When I2C_CTRL1.EN=0, it will be cleared by hardware. 0: SMBAlert pin go high. The response address header is followed by the NACK signal; 1: SMBAlert pin go low. The response address header is followed by the ACK signal.
12	PEC	Packet error checking It can be set or cleared by software. It will be cleared by hardware when PEC has been transferred, or by start or stop condition, or when I2C_CTRL1.EN=0. 0: No PEC transfer 1: PEC transfer. <i>Note: When arbitration is lost, the calculation of PEC is invalid.</i>
11	ACKPOS	Acknowledge/PEC Position (for data reception) It can be set or cleared by software. Or when I2C_CTRL1.EN=0, it will be cleared by hardware. 0: I2C_CTRL1.ACKEN bit determines whether to send an ACK to the byte currently being received; I2C_CTRL1.PEC bit indicates that the byte in the current shift register is PEC. 1: I2C_CTRL1.ACKEN bit determines whether to send an ACK to the next received byte; I2C_CTRL1.PEC bit indicates that the next byte received in the shift register is PEC.

Bit Field	Name	Description
		<p><i>Note:</i></p> <p><i>ACKPOS bit can only be used in 2-byte receiving configuration and must be configured before receiving data.</i></p> <p><i>For the second byte of NACK, the I2C_CTRL1.ACKEN bit must be cleared after the I2C_STS1.ADDRF bit is cleared.</i></p> <p><i>To detect the PEC of the second byte, the I2C_CTRL1.PEC bit must be set after the ACKPOS bit is configured and when the ADDR event is extended.</i></p>
10	ACKEN	<p>Acknowledge enable</p> <p>It can be set or cleared by software. Or when I2C_CTRL1.EN equals to 0, it will be cleared by hardware.</p> <p>0: No acknowledge send; 1: Send an acknowledge after receiving a byte</p>
9	STOPGEN	<p>Stop generation</p> <p>It can be set or cleared by software. Or it will be cleared by hardware when a stop condition is detected. Or it will be set by hardware when SMBus timeout error is detected,.</p> <p>In the master mode: 0: No stop condition generates; 1: Generate a stop condition.</p> <p>In the slave mode: 0: No stop condition generates; 1: Release SCL and SDA lines after the current byte.</p> <p><i>Note: When the STOPGEN, STARTGEN or PEC bit is set, the software should not take any write operation to I2C_CTRL1 until this bit is cleared by hardware. Otherwise, the STOPGEN, STARTGEN or PEC bits may be set twice.</i></p>
8	STARTGEN	<p>Start generation</p> <p>It can be set or cleared by software. Or it will be cleared by hardware when the start condition is transferred or I2C_CTRL1.EN=0.</p> <p>0: No start condition generates; 1: Generate a start conditions.</p>
7	NOEXTEND	<p>Clock stretching disable (Slave mode)</p> <p>This bit determines whether to pull SCL low when the data is not ready(I2C_STS1.ADDRF or I2C_STS1.BSF flag is set) in slave mode, and is cleared by software reset</p> <p>0: Enable Clock stretching. 1: Disable Clock stretching.</p>
6	GCEN	<p>General call enable</p> <p>0: Disable General call. not respond(NACK) to the address 00h; 1: Enable General call. respond(ACK) the address 00h.</p>
5	PECEN	<p>PEC enable</p> <p>0: Disable PEC module; 1: Enable PEC module.</p>
4	ARPEN	<p>ARP enable</p> <p>0: Disable ARP; 1: Enable ARP.</p>

Bit Field	Name	Description
		If I2C_CTRL1.SMBTYPE=0, the default address of SMBus device is used. If I2C_CTRL1.SMBTYPE=1, the host address of SMBus is used.
3	SMBTYPE	SMBus type 0: Device 1: Host
2	Reserved	Reserved, the reset value must be maintained.
1	SMBMODE	SMBus mode 0: I2C mode; 1: SMBus mode.
0	EN	I2C Peripheral enable 0: Disable I2C module; 1: Enable I2C module <i>Note: If this bit is cleared when communication is in progress, the I2C module is disabled and returns to the idle state after the current communication ends, all bits will be cleared.</i> <i>In master mode, this bit must never be cleared until the communication has ended.</i>

17.6.3 I²C Control Register 2 (I2C_CTRL2)

Address offset: 0x04

Reset value: 0x0000

15	13	12	11	10	9	8	7	6	5	0
Reserved		DMA LAST	DMA EN	BUFINT EN	EVTINT EN	ERRINT EN	Reserved		CLKFREQ[5:0]	
		rw	rw	rw	rw	rw			rw	

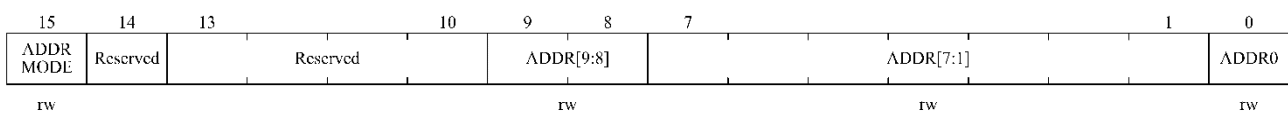
Bit Field	Name	Description
15:13	Reserved	Reserved, the reset value must be maintained.
12	DMALAST	DMA last transfer 0: Next DMA EOT is not the last transfer 1: Next DMA EOT is the last transfer <i>Note: This bit is used in the master receiving mode, so that a NACK can be generated when data is received for the last time.</i>
11	DMAEN	DMA requests enable 0: Disable DMA 1: Enable DMA
10	BUFINTEN	Buffer interrupt enable 0: When I2C_STS1.TXDATE=1 or I2C_STS1.RXDATNE=1, any interrupt is not generated. 1: If I2C_CTRL2.EVTINTEN= 1, When I2C_STS1.TXDATE=1 or I2C_STS1.RXDATNE= 1, interrupt will be generated.
9	EVTINTEN	Event interrupt enable 0: Disable event interrupt; 1: Enable event interrupt This interrupt is generated when: I2C_STS1.STARTBF = 1 (Master)

Bit Field	Name	Description
		I2C_STS1.ADDR F = 1 (Master/Slave) I2C_STS1.ADD10F = 1 (Master) I2C_STS1.STOPF = 1 (Slave) I2C_STS1.BSF = 1 with no I2C_STS1.TXDATE or I2C_STS1.RXDATNE event I2C_STS1.TXDATE = 1 if I2C_CTRL2.BUFINTEN = 1 I2C_STS1.RXDATNE = 1 if I2C_CTRL2.BUFINTEN = 1
8	ERRINTEN	Error interrupt enable 0: Disable error interrupt; 1: Enable error interrupt. This interrupt is generated when: I2C_STS1.BUSERR = 1; I2C_STS1.ARLOST = 1; I2C_STS1.ACKFAIL = 1; I2C_STS1.OVERRUN = 1; I2C_STS1.PECERR = 1; I2C_STS1.TIMOUT = 1; I2C_STS1.SMBALERT = 1.
7:6	Reserved	Reserved, the reset value must be maintained.
5:0	CLKFREQ[5:0]	I2C Peripheral clock frequency CLKFREQ[5:0] should be the APB1 clock frequency to generate the correct timing. 000000: Disable 000001: Disable 000010: 2MHz 000011: 3MHz ... 110000: 48MHz 110001~111111: Disable.

17.6.4 I²C Own Address Register 1 (I2C_OADDR1)

Address offset: 0x08

Reset value: 0x0000



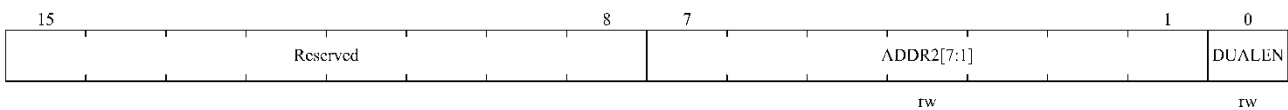
Bit Field	Name	Description
15	ADDRMODE	Addressing mode (slave mode) 0: 7-bit slave address 1: 10-bit slave address
14	Reserved	Must always be kept as '1' by the software.
13:10	Reserved	Reserved, the reset value must be maintained.
9:8	ADDR[9:8]	Interface address 9~8 bits of the address.

Bit Field	Name	Description
		<i>Note: don't care these bits in 7-bit address mode</i>
7:1	ADDR[7:1]	Interface address 7~1 bits of the address.
0	ADDR0	Interface address 0 bit of the address. <i>Note: don't care these bits in 7-bit address mode</i>

17.6.5 I²C Own Address Register 2 (I2C_OADDR2)

Address offset: 0x0C

Reset value: 0x0000

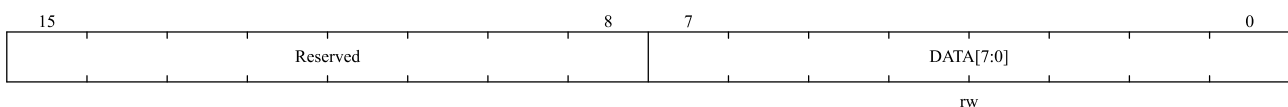


Bit Field	Name	Description
15:8	Reserved	Reserved, the reset value must be maintained.
7:1	ADDR2[7:1]	Interface address 7~1 bits of address in dual address mode.
0	DUALEN	Dual addressing mode enable 0: Disable dual address mode, only OADDR1 is recognized; 1: Enable dual address mode, both OADDR1 and OADDR2 are recognized. <i>Note: Valid only for 7-bit address mode</i>

17.6.6 I²C Data Register (I2C_DAT)

Address offset: 0x10

Reset value: 0x0000



Bit Field	Name	Description
15:8	Reserved	Reserved, the reset value must be maintained.
7:0	DATA[7:0]	8-bit data register Send or receive data buffer. <i>Notes: (1) In the slave mode, the address will not be copied into the data register;</i> <i>(2) If I2C_STS1.TXDATE = 0, data can still be written into the data register;</i> <i>(3) Note: If the ARLOST event occurs when processing the ACK pulse, the received byte will not be</i> <i>(4) Copied into the data register, so it cannot be read.</i>

17.6.7 I²C Status Register 1 (I2C_STS1)

Address offset: 0x14

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMB ALERT	TIM OUT	Reserved	PEC ERR	OVER RUN	ACK FAIL	AR LOST	BUS ERR	TXDATE	RXDAT NE	Reserved	STOPF	ADDR10F	BSF	ADDRF	START BF
rc_w0	rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	r	r		r	r	r	r	r

Bit Field	Name	Description
15	SMBALERT	SMBus alert Writing '0' to this bit by software can clear it, or it is cleared by hardware when I2C_CTRL1.EN=0. 0: No SMBus alert(host mode) or no SMB alert response address header sequence(slave mode); 1: SMBus alert event is generated on the pin(host mode) or receive SMBAAlert response address(slave mode)
14	TIMOUT	Timeout or Tlow error Writing '0' to this bit by software can clear it, or it is cleared by hardware when I2C_CTRL1.EN=0. 0: No Timeout error; 1: A timeout error occurred Error in the following cases: <ul style="list-style-type: none"> SCL has kept low for 25ms (Timeout). Master cumulative clock low extend time more than 10 ms (Tlow:mext). Slave cumulative clock low extend time more than 25 ms (Tlow:sext). Timeout in slave mode: slave device resets the communication and hardware frees the bus. Timeout in master mode: hardware sends the stop condition.
13	Reserved	Reserved, the reset value must be maintained.
12	PECERR	PEC Error in reception Writing '0' to this bit by software can clear it, or it is cleared by hardware when I2C_CTRL1.EN=0. 0: No PEC error 1: PEC error: receiver will returns NACK Whether the I2C_CTRL1.ACKEN bit is enabled
11	OVERRUN	Overrun/Underrun Writing '0' to this bit by software can clear it, or it is cleared by hardware when I2C_CTRL1.EN=0. 0: No Overrun/Underrun 1: Overrun/Underrun Set by hardware in slave mode when I2C_CTRL1.NOEXTEND=1, and when receiving a new byte in receiving mode, if the data within DAT register has not been read yet, over-run occurs,the new received byte will be lost.When transferring a new byte in transfer mode, but there is not new data that has not been written in DAT register, under-run occurs which leads that the same byte will be send twice.
10	ACKFAIL	Acknowledge failure Writing '0' to this bit by software can clear it, or it is cleared by hardware when I2C_CTRL1.EN=0. 0: No acknowledge failed; 1: Acknowledge failed.
9	ARLOST	Arbitration lost (master mode)

Bit Field	Name	Description
		<p>Writing '0' to this bit by software can clear it, or it is cleared by hardware when I2C_CTRL1.EN=0.</p> <p>0: No arbitration lost; 1: Arbitration lost.</p> <p>When the interface loses control of the bus to another host, the hardware will set this bit to '1', and the I2C interface will automatically switch back to slave mode (I2C_STS2.MSMODE=0).</p> <p><i>Note: In SMBUS mode, the arbitration of data in slave mode only occurs in the data stage or the acknowledge transfer interval (excluding the address acknowledge).</i></p>
8	BUSERR	<p>Bus error</p> <p>Writing '0' to this bit by software can clear it, or it is cleared by hardware when I2C_CTRL1.EN=0.</p> <p>0: No start or stop condition error 1: Start or stop condition error</p>
7	TXDATE	<p>Data register empty (transmitters)</p> <p>Writing data to DAT register by software can clear this bit; Or after a start or stop condition occurs, or automatically cleared by hardware when I2C_CTRL1.EN=0.</p> <p>0: Data register is not empty; 1: Data register is empty.</p> <p>When sending data, this bit is set to '1' when the data register is empty, and it is not set at the address sending stage.</p> <p>If a NACK is received, or the next byte to be sent is PEC(I2C_CTRL1.PEC=1), this bit will not be set.</p> <p><i>Note: After the first data to be sent is written, or data is written when BSF is set, the TXDATE bit cannot be cleared, because the data register is still empty.</i></p>
6	RXDATNE	<p>Data register not empty(receivers)</p> <p>This bit is cleared by software reading and writing to the data register, or cleared by hardware when I2C_CTRL1.EN=0.</p> <p>0: Data register is empty; 1: Data register is not empty.</p> <p>During receiving data, this bit is set to '1' when the data register is not empty, and it is not set at the address receiving stage.</p> <p>RXDATNE is not set when the ARLOST event occurs.</p> <p><i>Note: When BSF is set, the RXDATNE bit cannot be cleared when reading data, because the data register is still full.</i></p>
5	Reserved	Reserved, the reset value must be maintained.
4	STOPF	<p>Stop detection (slave mode)</p> <p>After the software reads the STS1 register, the operation of writing to the CTRL1 register will clear this bit, or when I2C_CTRL1.EN=0, the hardware will clear this bit.</p> <p>0: No stop condition is detected; 1: Stop condition is detected.</p> <p>After a ACK, the hardware sets this bit to '1' when the slave device detects a stop condition on the bus.</p> <p><i>Note: I2C_STS1.STOPF bit is not set after receiving NACK.</i></p>

Bit Field	Name	Description
3	ADDR10F	<p>10-bit header sent (Master mode)</p> <p>After the software reads the STS1 register, the operation of writing to the CTRL1 register will clear this bit, or when I2C_CTRL1.EN=0, the hardware will clear this bit.</p> <p>0: No ADDR10F event; 1: Received has sent the first address byte.</p> <p>In 10-bit address mode, when the master device has sent the first byte, the hardware sets this bit to '1'.</p> <p><i>Note: After receiving a NACK, the I2C_STS1.ADDR10F bit is not set.</i></p>
2	BSF	<p>Byte transfer finished</p> <p>After the software reads the STS1 register, reading or writing the data register will clear this bit; Or after sending a start or stop condition in sending mode, or when I2C_CTRL1.EN=0, this bit is cleared by hardware.</p> <p>0: Byte transfer does not finish. 1: Byte transfer finished.</p> <p>When I2C_CTRL1.NOEXTEND =0, the hardware sets this bit to '1' in the following cases: In receiving mode, when a new byte (including ACK pulse) is received and the data register has not been read (I2C_STS1.RXDATNE=1). In sending mode, when a new data is to be transmitted and the data register has not been written with the new data (I2C_STS1.TXDATE=1).</p> <p><i>Note: After receiving a NACK, the BSF bit will not be set.</i></p> <p><i>If the next byte to be transferred is PEC (I2C_STS2.TRF is '1' and I2C_CTRL1.PEC is '1'), the BSF bit will not be set.</i></p>
1	ADDRF	<p>Address sent (master mode) / matched (slave mode)</p> <p>After the STS1 register is read by software, reading the STS2 register will clear this bit, or when I2C_CTRL1.EN=0, it will be cleared by hardware.</p> <p>0: Address mismatch or no address received(slave mode) or Address sending did not end(master mode); 1: Received addresses matched(slave mode) or Address sending ends(master mode)</p> <p>In master mode: In 7-bit address mode, this bit is set to '1' after receiving the ACK of the address. In 10-bit address mode, this bit is set to '1' after receiving the ACK of the second byte of the address.</p> <p>In slave mode: Hardware sets this bit to '1' (when the corresponding setting is enabled) when the received slave address matches the content in the OADDR register, or a general call or SMBus device default address or SMBus host or SMBus alter is recognized.</p> <p><i>Note: After receiving NACK, the I2C_STS1.ADDRF bit will not be set.</i></p>
0	STARTBF	<p>Start bit (Master mode)</p> <p>After the STS1 register is read by software, writing to the data register will clear this bit, or when I2C_CTRL1.EN=0, the hardware will clear this bit.</p> <p>0: Start condition was not sent; 1: Start condition has been sent.</p> <p>This bit is set to '1' when the start condition is sent.</p>

17.6.8 I²C Status Register 2 (I2C_STS2)

Address offset: 0x18

Reset value: 0x0000

15	8	7	6	5	4	3	2	1	0
PECVAL[7:0]		DUAL FLAG	SMBH ADDR	SMBD ADDR	GCALL ADDR	Reserved	TRF	BUSY	MS MODE
r		r	r	r	r		r	r	r

Bit Field	Name	Description
15:8	PECVAL[7:0]	Packet error checking register Stores the internal PEC value When I2C_CTRL1.PECEN =1.
7	DUALFLAG	Dual flag(Slave mode) Hardware clears this bit when a stop condition or a repeated start condition is generated, or when I2C_CTRL1.EN=0. 0: Received address matches the content in OADDR1; 1: Received address matches the content in OADDR2.
6	SMBHADDR	SMBus host header (Slave mode) Hardware clears this bit when a stop condition or a repeated start condition is generated, or when I2C_CTRL1.EN=0. 0: SMBus host address was not received; 1: when I2C_CTRL1.SMBTYPE=1 and I2C_CTRL1.ARPEN=1, the SMBus host address is received.
5	SMBDADDR	SMBus device default address (Slave mode) Hardware clears this bit when a stop condition or a repeated start condition is generated, or when I2C_CTRL1.EN=0. 0: The default address of SMBus device has not been received; 1: when I2C_CTRL1.ARPEN=1, the default address of SMBus device is received.
4	GCALLADDR	General call address(Slave mode) Hardware clears this bit when a stop condition or a repeated start condition is generated, or when I2C_CTRL1.EN=0. 0: No general call address was received; 1: when I2C_CTRL1.GCEN=1, general call address was received.
3	Reserved	Reserved, the reset value must be maintained.
2	TRF	Transmitter/receiver After detecting the stop condition (I2C_STS1.STOPF=1), repeated start condition or bus arbitration loss (I2C_STS1.ARLOST=1), or when I2C_CTRL1.EN=0, the hardware clears it. 0: Data receiving mode 1: Data transmission mode; At the end of the whole address transmission stage, this bit is set according to the R/W bit of the address byte.
1	BUSY	Bus busy Hardware clears this bit when a stop condition is detected. 0: No data communication on the bus; 1: Data communication on the bus.

Bit Field	Name	Description
		When detecting that SDA or SCL is low level, the hardware sets this bit to '1'; <i>Note: This bit indicates the bus communication currently in progress, and this information is still updated when the interface is disabled (I2C_CTRL1.EN=0).</i>
0	MSMODE	Master/slave mode Hardware clears this bit when a stop condition is detected on the bus, arbitration is lost (I2C_STS1.ARLOST=1), or when I2C_CTRL1.EN=0. 0: In slave mode; 1: In master mode. When the interface is in the master mode (I2C_STS1.STARTBF=1), the hardware sets this bit;

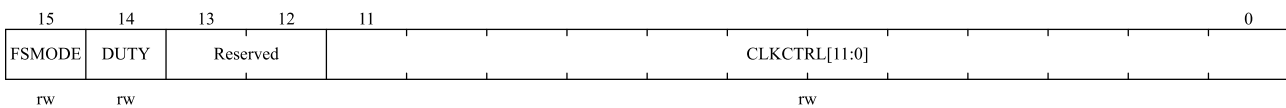
17.6.9 I²C Clock Control Register (I2C_CLKCTRL)

Address offset: 0x1c

Reset value: 0x0000

Note: 1. F_{PCLK1} is required to be an integer multiple of 10 MHz, so that a fast clock of 400KHz can be generated correctly.

2. The CLKCTRL register can only be set when I²C is turned off (I2C_CTRL1.EN=0)



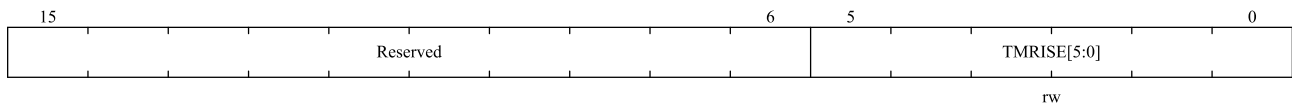
Bit Field	Name	Description
15:14	DUTY	SCL duty cycle 00: Tlow/Thigh = 1; 01: Tlow/Thigh = 1; 10: Tlow/Thigh = 2; 11: Tlow/Thigh = 16/9; <i>Note: 00 or 01 configuration is recommended for SCL 100K or 1M. 10 or 11 configuration recommended for SCL 400K.</i>
13:12	Reserved	Reserved, the reset value must be maintained.
11:0	CLKCTRL[11:0]	Clock control register in Fast/Standard mode (Master mode) This division factor is used to set the SCL clock in the master mode. <ul style="list-style-type: none"> If duty cycle = Tlow/Thigh = 1/1: $CLKCTRL = f_{PCLK1}(Hz)/100000/2$ $Tlow = CLKCTRL \times T_{PCLK1}$ $Thigh = CLKCTRL \times T_{PCLK1}$ If duty cycle = Tlow/Thigh = 2/1: $CLKCTRL = f_{PCLK1}(Hz)/100000/3$ $Tlow = 2 \times CLKCTRL \times T_{PCLK1}$ $Thigh = CLKCTRL \times T_{PCLK1}$ If duty cycle = Tlow/Thigh = 16/9: $CLKCTRL = f_{PCLK1}(Hz)/100000/25$ $Tlow = 16 \times CLKCTRL \times T_{PCLK1}$ $Thigh = 9 \times CLKCTRL \times T_{PCLK1}$

Bit Field	Name	Description
		For example, if $f_{PCLK1}(\text{Hz}) = 8\text{MHz}$, duty cycle = 1/1, $\text{CLKCTRL} = 8000000/100000/2 = 0x28$. Notes: (1) The minimum setting value is 0x04 in standard mode and 0x01 in fast mode; (2) $T_{high} = T_{r(SCL)} + T_{w(SCLH)}$. See the definitions of these parameters in the data sheet for details. (3) $T_{low} = T_{f(SCL)} + T_{w(SCLL)}$, refer to the definitions of these parameters in the data sheet for details; (4) These delays have no filters;

17.6.10 I²C Rise Time Register (I2C_TMRISE)

Address offset: 0x20

Reset value: 0x0002



Bit Field	Name	Description
15:6	Reserved	Reserved, the reset value must be maintained.
5:0	TMRISE[5:0]	Maximum rise time in fast/standard mode (master mode). These bits must be set to the maximum SCL rising time given in the I2C bus specification, and incremented step is 1. For example, the maximum allowable SCL rise time in standard mode is 1000ns. if the value in I2C_CTRL2.CLKFREQ [5:0] is equal to 0x08(8MHz) and $T_{PCLK1}=125\text{ns}$, $09h(1000\text{ns}/125\text{ns} + 1)$ must be written in TMRISE[5:0]. If the result is not an integer, write the integer part to TMRISE[5:0] to ensure the T_{HIGH} parameter. Note: TMRISE[5:0] can only be set when I2C is disabled (EN=0).

18 Universal Synchronous Asynchronous Receiver Transmitter

18.1 Introduction

USART is a full-duplex serial data exchange interface that supports synchronous or asynchronous communication. This interface is a highly flexible serial communication device that can perform full-duplex data exchange with external devices.

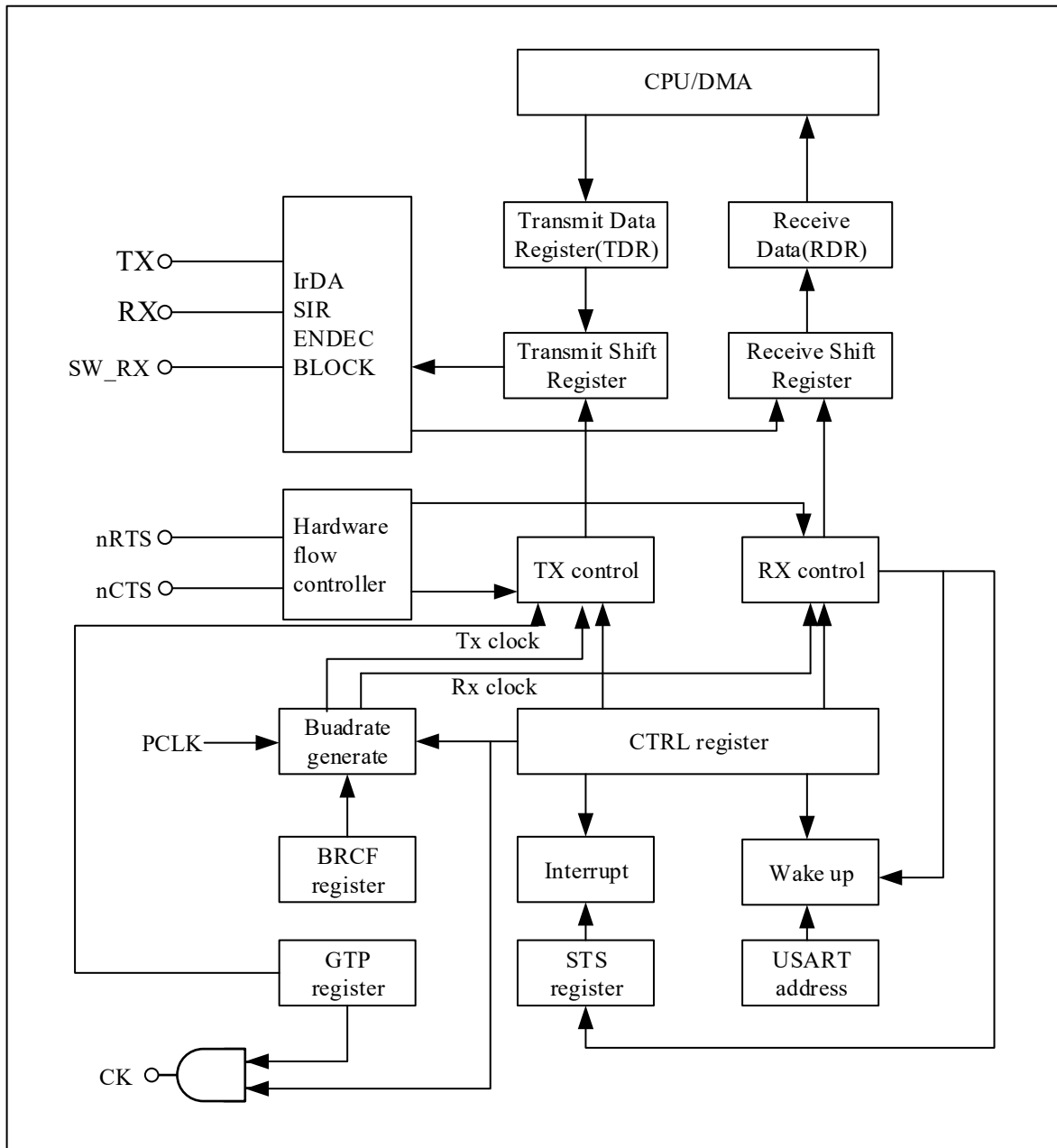
The USART has programmable transmit and receive baud rates and can communicate continuously using DMA. It also supports multiprocessor communication, LIN mode, synchronous mode, single-wire half-duplex communication, smartcard asynchronous protocol, IrDA SIR ENDEC function, and hardware flow control function.

18.2 Main Features

- Full-duplex operation
- Single-wire half-duplex operation
- Baud rate programmable, the highest baud rate can reach 3Mbit/s
- Supports serial data frame structure with 8 or 9 data bits, 1 or 2 stop bits
- Supports generation parity bits by hardware and checking parity bits
- Supports hardware flow control: RTS flow control and CTS flow control
- Supports DMA receiving and transmitting
- Supports multiprocessor communication, enter mute mode if the address dose not match, can wake up by idle detection or address mark detection
- Synchronous mode, allowing users to control bidirectional synchronous serial communication in master mode
- Compliant with ISO7816-3 standard, support smartcard asynchronous protocol
- IrDA SIR encoder and decoder : provide IrDA normal mode and IrDA low power mode
- LIN (Local Area Network) mode
- Supports data overflow error detection, frame error detection, noise error detection, parity error detection
- Interrupt requests include: transmit data register empty, CTS flag, transmit complete, receive data ready to read, data overflow detected, idle line detected, parity error, LIN break frame detection, noise flag/overflow error/frame error in multi-buffer communication

18.3 Functional Block Diagram

Figure 18-1 USART Block Diagram



18.4 Function Description

As shown in the Figure 18-1, the bidirectional communication of USART needs to connecting the RX and TX pins to the external devices. Among them, TX is the output pin for serial data transmission. When the transmitter is active and not transmitting data, the TX pin is pulled high. When the transmitter is inactive, the TX pin reverts to the I/O port configuration. RX is an input pin for serial data reception, data is received by the oversampling technique.

The data packets of serial communication are transmitted from the transmitting device to the RX interface of the receiving device through its own TX interface. The bus is in an idle state when not transmitting or receiving. Frame format is: 1 start

bit + 8 or 9 data bits (least significant bit first) + 1 parity bit (optional) + 0.5,1,1.5 or 2 stop bits.

Using the fractional baud rate generator to configure transmit and receive baud rates .

According to the block diagram, when using the hardware flow control mode, the nRTS output and nCTS input pins are required. When the USART receiver is ready to receive new data, nRTS becomes low level. If nCTS is valid (pulled to low level), the next data is transmitted, otherwise the next frame of data is not transmitted.

When using synchronous mode, the CK pin is required. The CK pin is used for clock output for synchronous transfers. Clock phase and polarity are software programmable. During the start and stop bits, the CK pin does not output clock pulses. The CK pin is also used to provide clock when using smartcard mode.

18.4.1 USART Frame Format

Start bit: 1bit, active low.

Data bits: Configurable USART_CTRL1.WL as 8 or 9 bits, with the LSB first.

Stop bit: Active high.

Idle frame: A complete data frame consisting entirely of '1's, including the start bit. followed by the start bit of a data frame containing the data .

Break frame: A complete data frame consisting entirely of '0's, including the stop bit. at the end of the break frame, the transmitter inserts 1 or 2 stop bits ('1') to acknowledge the start bit.

Figure 18-2 Word Length = 8 Setting

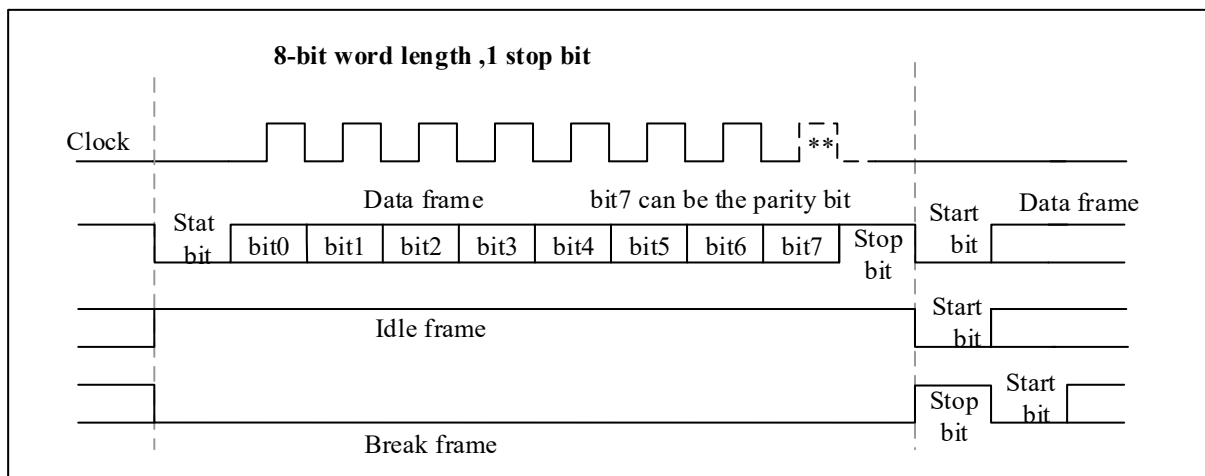
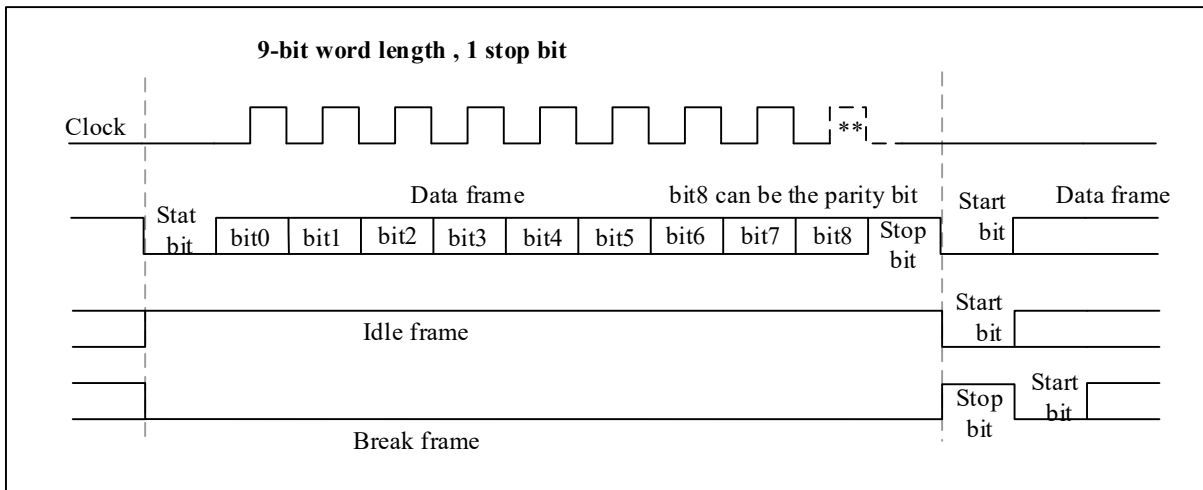


Figure 18-3 Word Length = 9 Setting



18.4.2 Transmitter

After the transmitter is enabled, the data in the transmit shift register is transmitted out through the TX pin.

Idle frame

Setting USART_CTRL1.TXEN will cause the USART to transmit an idle frame before transmitting data.

Character transmission

After the idle frame ends, characters can be transmitted normally. Each character is preceded by a low start bit. The transmitter transmits 8-bit or 9-bit data according to the configuration of the data bit length, with the LSB first. If USART_CTRL1.TXEN is cleared during data transmission,

it will cause the baud rate counter to stop counting and the data being transmitted will be corrupted.

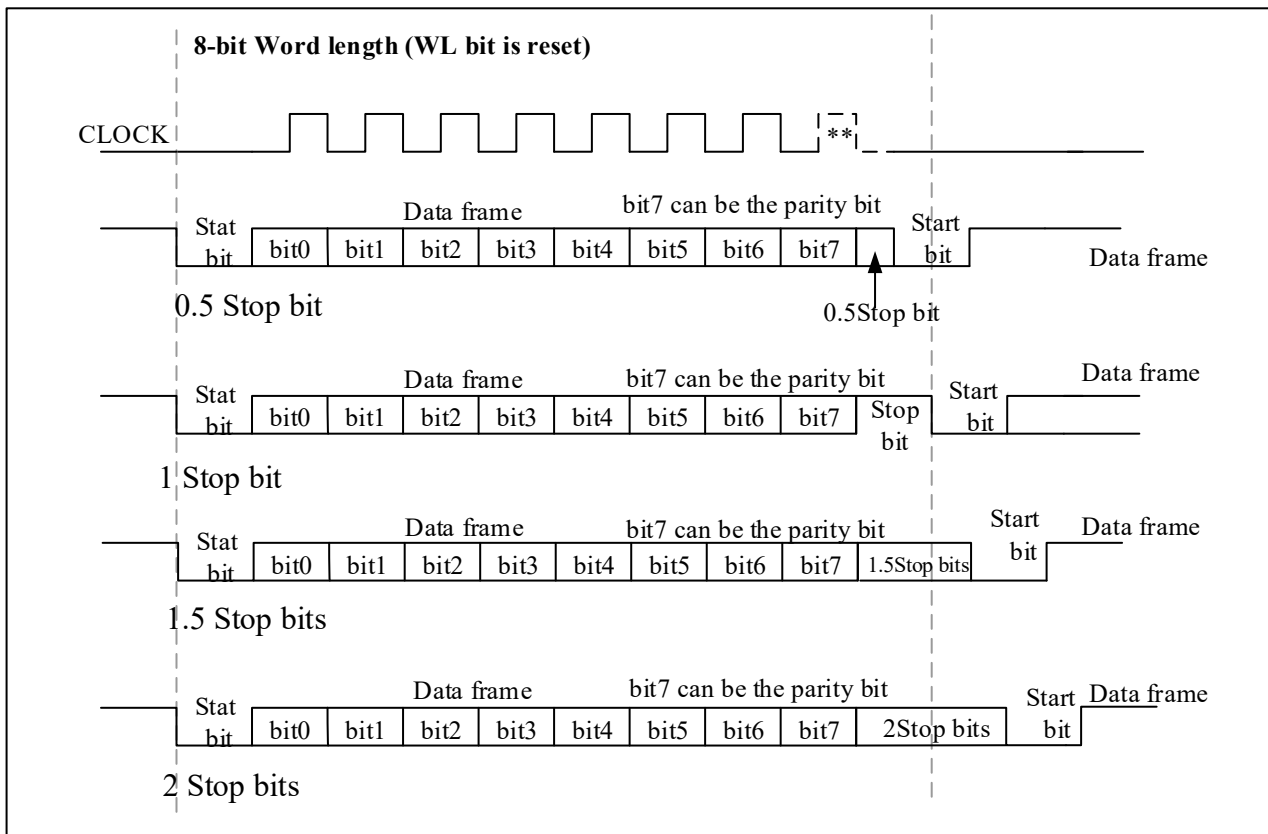
Stop bit

After the character frame ends, the transmitter automatically transmits stop bits. The number of stop bits can be configured by setting USART_CTRL2.STPB[1:0].

Table 18-1 Stop Bit Configuration

USART_CTRL2.STPB[1:0]	Stop Bit Length (bits)	Functional Description
00	1	default
01	0.5	Receiving in Smartcard mode
10	2	General USART mode, single-wire mode and modem mode.
11	1.5	Transmitting and receiving in Smartcard mode

Figure 18-4 Configuration Stop Bit



Break frame

Setting the USART_CTRL1.SDBRK to transmit the break frame. When the data length is 8 bits, the break frame consists of 10 bits with low level, followed by a high level stop bit; when the data length is 9 bits, the break frame consists of 11 bits with low level, followed by a high level stop bit.

After the break frame is transmitted, USART_CTRL1.SDBRK is cleared by hardware, and the stop bit of the break frame is transmitted automatically. Therefore, if user want to continuously transmit break frame, USART_CTRL1.SDBRK should be set after the stop bit of the previous break frame has been transmitted.

If the USART_CTRL1.SDBRK bit is cleared by software before starting to send the break frame, the break frame will not be transmitted.

Transmitter process

1. Enable USART_CTRL1.UEN to activate USART;
2. Configure the transmitter's baud rate, data bit length, parity bit (optional), the number of stop bits and configure relevant DMA ;
3. Activate the transmitter (USART_CTRL1.TXEN);
4. Write the data to be transmitted into the USART_DAT register sequentially through the CPU or DMA, and the write operation to the USART_DAT register will clear USART_STS.TXDE;
5. After writing the last data into the USART_DAT register, wait for USART_STS.TXC =1, which indicates the end of the transmission of the last data frame.

Single byte communication

Writing to the USART_DAT register clears the USART_STS.TXDE bit.

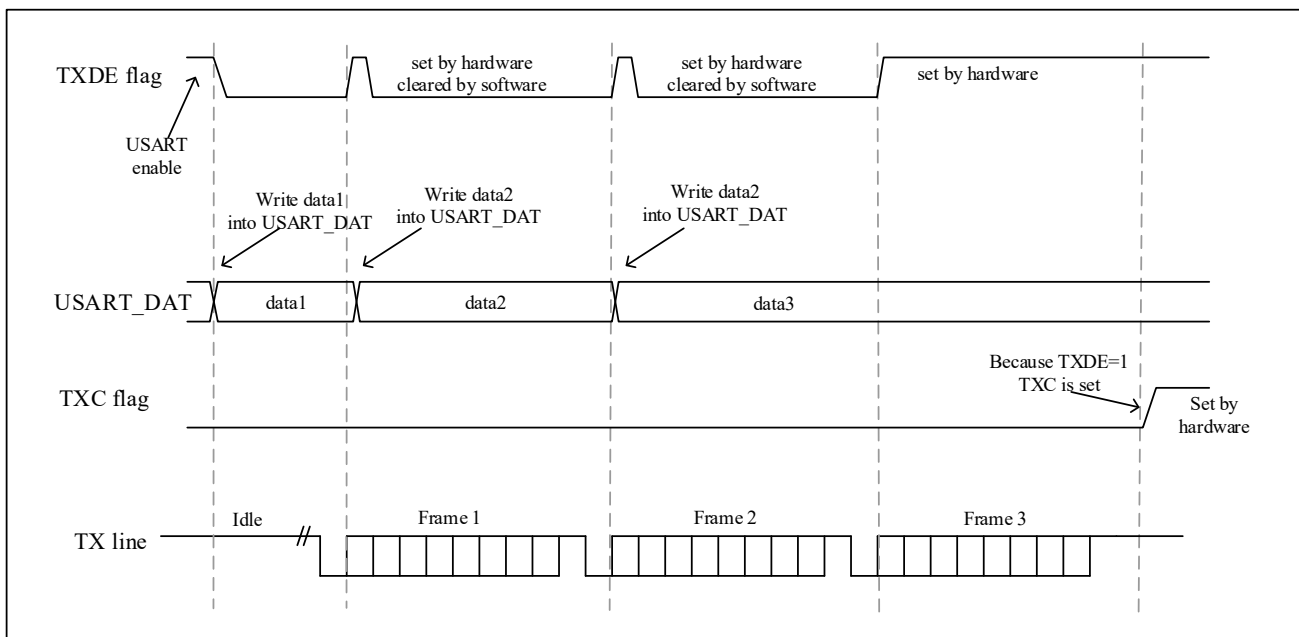
The USART_STS.TXDE bit is set by hardware when the data in the TDR register is transferred to the transmit shift register (indicating that data is being transmitted). An interrupt will be generated if USART_CTRL1.TXDEIEN is set. At this point, the next data can be written into the USART_DAT register because the TDR register has been cleared and will not overwrite the previous data.

Write operation to USART_DAT register:

- When the transmit shift register is not transmitting data and is in an idle state, the data is directly put into the shift register for transmission, and the USART_STS.TXDE bit is set by hardware;
- When the transmit shift register is transmitting data, the data is stored in the TDR register, and after the current transmission is completed, the data is put into the transmit shift register.

When a frame containing data is transmitted and USART_STS.TXDE=1, the USART_STS.TXC bit is set to '1' by hardware. An interrupt is generated if USART_CTRL1.TXCIEN is '1'. The USART_STS.TXC bit is cleared by a software sequence (read USART_STS register first, then write USART_DAT register).

Figure 18-5 TXC/TXDE Changes During Transmission



18.4.3 Receiver

Start bit detection

When the received sampling sequence is: 1 1 1 0 X 0 X 0 X 0 0 0, it is considered that a start bit is detected.

The samples at the 3rd, 5th, and 7th bits, and the samples at the 8th, 9th, and 10th bits are all '0' (that is, 6 '0'), then confirm the receipt of the start bit, the USART_STS.RXDNE flag bit is set and the NEF noise flag is not set. If USART_CTRL1.RXDNEIEN=1, an interruption occurs.

If there are six '0' samples at the 3rd, 5th, 7th bits, and at the 8th, 9th, 10th bits, a start bit is confirmed to have been received, and USART_STS.RXDNE is set to 1, but the NE noise flag will not be set. If USART_CTRL1.RXDNEIEN has

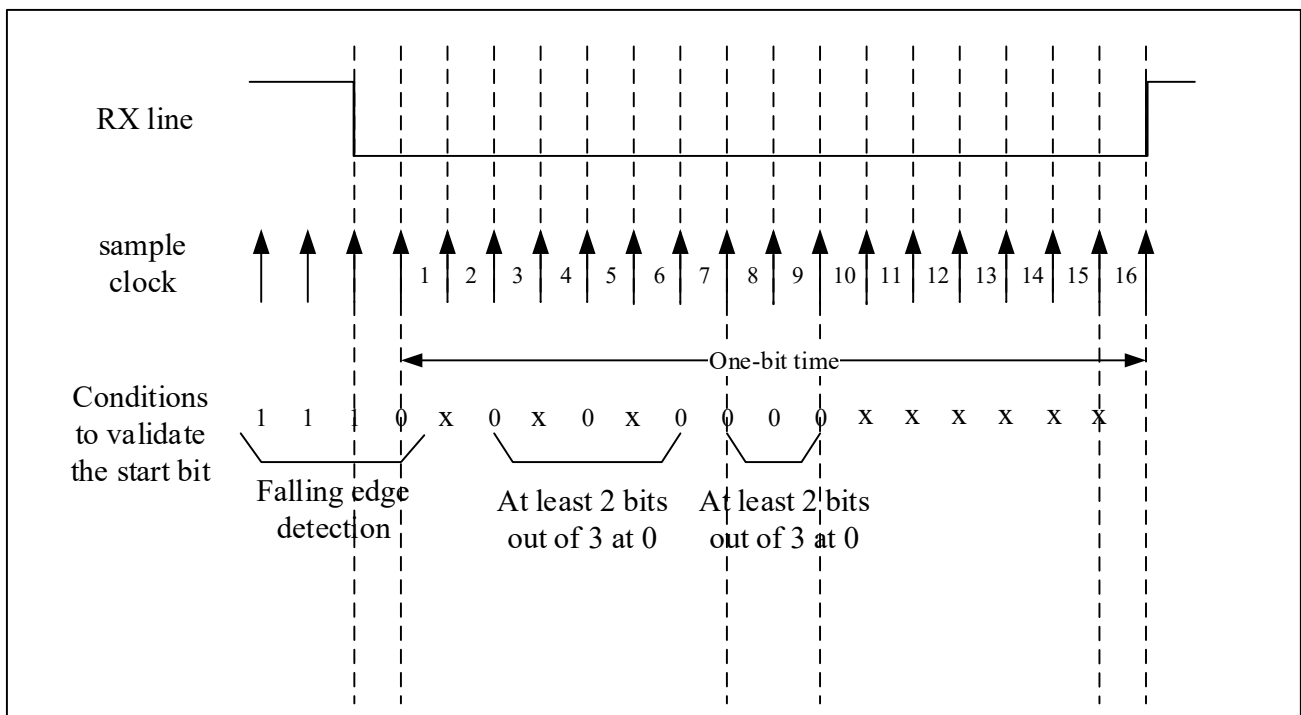
been set to 1, an interrupt will be generated..

If there are two '0' samples at the 3rd, 5th, 7th bits, and at the same time, there are two '0' samples at the 8th, 9th, 10th bits, a start bit is also confirmed to have been received, but the NE noise flag will be set..

If there are three '0' samples at the 3rd, 5th, 7th bits, and at the same time, there are two '0' samples at the 8th, 9th, 10th bits, a start bit is also confirmed to have been received, and the NE noise flag will be set.

If there are two '0' samples at the 3rd, 5th, 7th bits, and at the same time, there are three '0' samples at the 8th, 9th, 10th bits, a start bit is also confirmed to have been received, and the NE noise flag will be set.

Figure 18-6 Start Bit Detection



Stop bit description

The number of stop bits can be configured by the USART_CTRL2.STPB[1:0]. In normal mode, 1 or 2 stop bits can be selected. In smartcard mode, 0.5 or 1.5 stop bits can be selected.

- 0.5 stop bit (receive in smartcard mode): 0.5 stop bit is not sampled. Therefore, if 0.5 stop bits is selected, framing errors and broken frames cannot be detected.
- 1 stop bit: the sampling for one stop bit is done on the 8th, 9th and 10th samples.
- 1.5 stop bits (Smartcard mode): when transmitting in smartcard mode, the device must check whether the data is transmitted correctly. So the receiver function block must be activated (USART_CTRL1.RXEN=1) and sample the signal on the data line during the transmission of the stop bit. If a parity error occurs, the smartcard will pull down the data line when the transmitter samples the NACK signal, that is, within the time corresponding to the stop bit on the bus, indicating that a framing error has occurred. The USART_STS.FEF is set together with the USART_STS.RXDNE at the end of the 1.5th stop bit. The 1.5 stop bits were sampled at points 16th, 17th and 18th. Sampling 1.5 stop bits can be divided into two parts: one is 0.5 clock cycles, the receiver does nothing. This is followed

by the stop bit of 1 clock cycle, which is sampled at the midpoint of this period of time. For details, please refer to Section 18.4.14 Smartcard mode.

- 2 stop bits: the sampling of the 2 stop bits is completed at the 8th, 9th and 10th sampling points of the first stop position. If a frame error is detected during the first stop bit, the frame error flag is set. The USART_STS.RXNE flag will be set at the end of the first stop bit. The second stop bit does not detect framing error.

Receiver process

1. Enable USART_CTRL1.UEN to activate USART;
2. Configure the receiver's baud rate, data bit length, parity bit (optional), the number of stop bit or DMA configuration;
3. Activate the receiver (USART_CTRL1.RXEN) and start detecting the start bit;
4. The receiver receives 8-bit or 9-bit data according to the configuration of the data bit length, and LSB of the data is first shifted from the RX pin into the receive shift register;
5. When the data of the received shift register is moved to the RDR register, USART_STS.RXDNE is set, and the data can be read out. If USART_CTRL1.RXNEIEN is 1, an interrupt will be generated;
6. When an overflow error, noise error, or frame error is detected in the received frame, the corresponding error flag status bit will be set. If USART_CTRL1.RXEN is reset during data transmission, the data being received will be lost;
7. Clear USART_STS.RXDNE by reading USART_DAT :
 - During multi-buffer communication, the data register is cleared by the DMA read operation;
 - During single-buffer communication, it is cleared by software reading the USART_DAT register.

Idle frame detection

The USART_STS.IDLEF is set when an idle frame is detected. An interrupt is generated if USART_CTRL1.IDLEIEN is '1'. USART_STS.IDLEF bit is cleared by a software sequence (read USART_STS register first, then read USART_DAT register).

Break frame detection

The frame error flag(USART_STS.FEF) is set by hardware when the receiver detects a break frame. It can be cleared by a software sequence (read USART_STS register first, then read USART_DAT register).

Framing error

A framing error occurs when the stop bit is not received and recognized at the expected time. At this time, the frame error flag USART_STS.FEF will be set by hardware, and the invalid data will be transferred from the shift register to the USART_DAT register. During single-byte communication, no framing error interrupt will be generated because it occurs with USART_STS.RXDNE and the hardware will generate an interrupt when the USART_STS.RXDNE flag is set. In multi-buffer communication mode, an interrupt will be generated if the USART_CTRL3.ERRIEN bit is set.

Overrun error

When USART_STS.RXDNE is still '1' and the data currently received in the shift register needs to be transmitted to the RDR register, an overflow error will be detected and the hardware will set USART_STS.OREF. When this bit is set, the value in the RDR register is not lost, but the data in the shift register is overwritten. USART_STS.OREF is cleared by a software sequence (read USART_STS register first, then write USART_DAT register).

When an overflow error occurs, an interrupt is generated if USART_STS.RXDNE is '1'. In multi-buffer communication

mode, if the USART_CTRL3.ERRIEN bit is set, an error interrupt will be generated.

Noise error

USART_STS.NEF is set by hardware when noise is detected on a received frame. It is cleared by software sequence (read USART_STS register first, then write USART_DAT register). During single-byte communication, no noise interrupt generated because it occurs with USART_STS.RXDNE and the hardware will generate an interrupt when the USART_STS.RXDNE flag is set. In multi-buffer communication mode, an error interrupt is generated when the USART_STS.NEF flag is set if the USART_CTRL3.ERRIEN bit is set.

Table 18-2 Data Sampling for Noise Detection

Sample Value	NE Status	Received Bits	Data Validity
000	0	0	Effective
001	1	0	be invalid
010	1	0	be invalid
011	1	1	be invalid
100	1	0	be invalid
101	1	1	be invalid
110	1	1	be invalid
111	0	1	Effective

18.4.4 Fractional Baud Rate Calculation

The baud rate of the USART can be configured in the USART_BRCF register. This register defines the integer and fractional parts of the baud rate divider. The baud rate of the transmitter and receiver should be configured to the same value. Be careful not to change the value of the USART_BRCF register during communication, because the baud rate counter will be replaced by the new value of the baud rate register.

$$\text{TX / RX baud rate} = f_{\text{PCLK}} / (16 * \text{USARTDIV})$$

where f_{PCLK} is the clock provided to the peripheral:

- PCLK1 is used for USART2 and UART5, up to 48MHz;
- PCLK2 is used for USART1 and UART6, up to 48 MHz.

USARTDIV is an unsigned fixed-point number.

USARTDIV and USART_BRCF register configuration

Example 1:

If DIV_Integer = 27, DIV_Decimal = 12 (USART_BRCF = 0x1BC), then

$$\text{DIV_Integer(USARTDIV)} = 27$$

$$\text{DIV_Decimal(USARTDIV)} = 12/16 = 0.75$$

So USARTDIV = 27.75

Example 2:

Requirements USARTDIV = 25.62, there are:

$$\text{DIV_Decimal} = 16 * 0.62 = 9.92$$

Closest integer is: 10 = 0x0A

$$\text{DIV_Integer} = \text{DIV_Integer}(25.620) = 25 = 0x19$$

So, USART_BRCF = 0x19A

Example 3:

Requirements USARTDIV = 50.99, there are:

$$\text{DIV_Decimal} = 16 * 0.99 = 15.84$$

Closest integer: 16 = 0x10 => DIV_Decimal[3:0] overrun => carry must be added to the fractional part

$$\text{DIV_Integer} = \text{DIV_Integer}(0d50.990 + \text{carry}) = 51 = 0x33$$

So: USART_BRCF = 0x330, USARTDIV = 0d51.00

Table 18-3 Error Calculation When Setting Baud Rate

Baud Rate		f _{CLK} = 36MHz			f _{CLK} = 48MHz		
serial number	Kbps	Reality	Set value in register	Error(%)	Reality	Set value in register	Error(%)
1	2.4	2.4	937.5	0%	2.4	1250	0%
2	9.6	9.6	234.375	0%	9.6	312.5	0%
3	19.2	19.2	117.1875	0%	19.2	156.25	0%
4	57.6	57.6	39.0625	0%	57.623	52.0625	0.04%
5	115.2	115.384	19.5	0.15%	115.1	26.0625	0.08%
6	230.4	230.769	9.75	0.16%	230.769	13	0.16%
7	460.8	461.538	4.875	0.16%	461.538	6.5	0.16%
8	921.6	923.076	2.4375	0.16%	923.076	3.25	0.16%
9	2250	2250	1	0%	2285.714	1.3125	1.58%
10	3000	impossible	impossible	impossible	3000	1	0%

Note: The lower the clock frequency of the CPU, the lower the error for a particular baud rate.

18.4.5 Receiver's Tolerance Clock Deviation

Transmission errors (including transmitter oscillator variations), receiver baud rate errors, receiver oscillator variations, transmission line variations (usually due to the inconsistency between the low-to-high transition timing of the transceiver

and the high-to-low transition timing of the transceiver) may occur in application, these factors will affect the overall clock system variation. Only when the sum of the above four changes is less than the tolerance of the USART receiver, the USART asynchronous receiver can operate normally.

When receiving data normally, the tolerance of the USART receiver depends on the selection of the data bit length and whether it is generated using a fractional baud rate. The tolerance of the USART receiver is equal to the maximum tolerable variation.

Table 18-4 When DIV_Decimal = 0. Tolerance of USART Receiver

WL Bit	NF Is An Error	NF Is Don't Care
0	3.75%	4.375%
1	3.41%	3.97%

Table 18-5 When DIV_Decimal != 0. Tolerance of USART Receiver

WL bit	NF is an error	NF is don't care
0	3.33%	3.88%
1	3.03%	3.53%

18.4.6 Parity Control

Parity can be enabled by configuring the USART_CTRL1.PCEN bit.

After the parity bit is enabled, a parity bit is generated and transmitted automatically when transmission data. and parity check is performed on reception.

Table 18-6 Frame Format

WL Bit	PCEN Bit	USART Frame
0	0	Start bit 8-bit data Stop bit
0	1	Start bit 7 bits of data Parity bit Stop bit
1	0	Start bit 9-bit data Stop bit
1	1	start bit 8-bit data parity bit stop bit

Even parity

Configure USART_CTRL1.PSEL to 0, and even parity can be selected.

Make the number of '1' in the transmitted data (including parity bit) be an even number. For example: if Data=11000101, there are 4 '1's, then the parity bit will be '0' (4 '1' in total). The receiver confirms the number of '1's in the data. If it is an even number, the check is passed, indicating that no errors occurred during the transmission process. If it is not even, it means that an error has occurred, the USART_STS.PEF flag is set to '1', and an interrupt is generated if USART_CTRL1.PEIE is enabled,.

Odd parity

Configure USART_CTRL1.PSEL to 1, odd parity can be selected.

Make the number of '1' in the transmitted data (including parity bit) be an odd number. For example: if Data=11000101, there are 4 '1's, then the parity bit will be '1' (5 '1' in total). The receiver confirms the number of '1's in the data. If it is an odd number, the check is passed, indicating that no errors occurred during the transmission process. If it is not an odd number, it means that an error has occurred, the USART_STS.PEF flag is set to '1', and an interrupt is generated if USART_CTRL1.PEIE is enabled.

18.4.7 DMA Communication

The USART supports the DMA mode, which can realize high-speed data communication through using multi-buffer configuration.

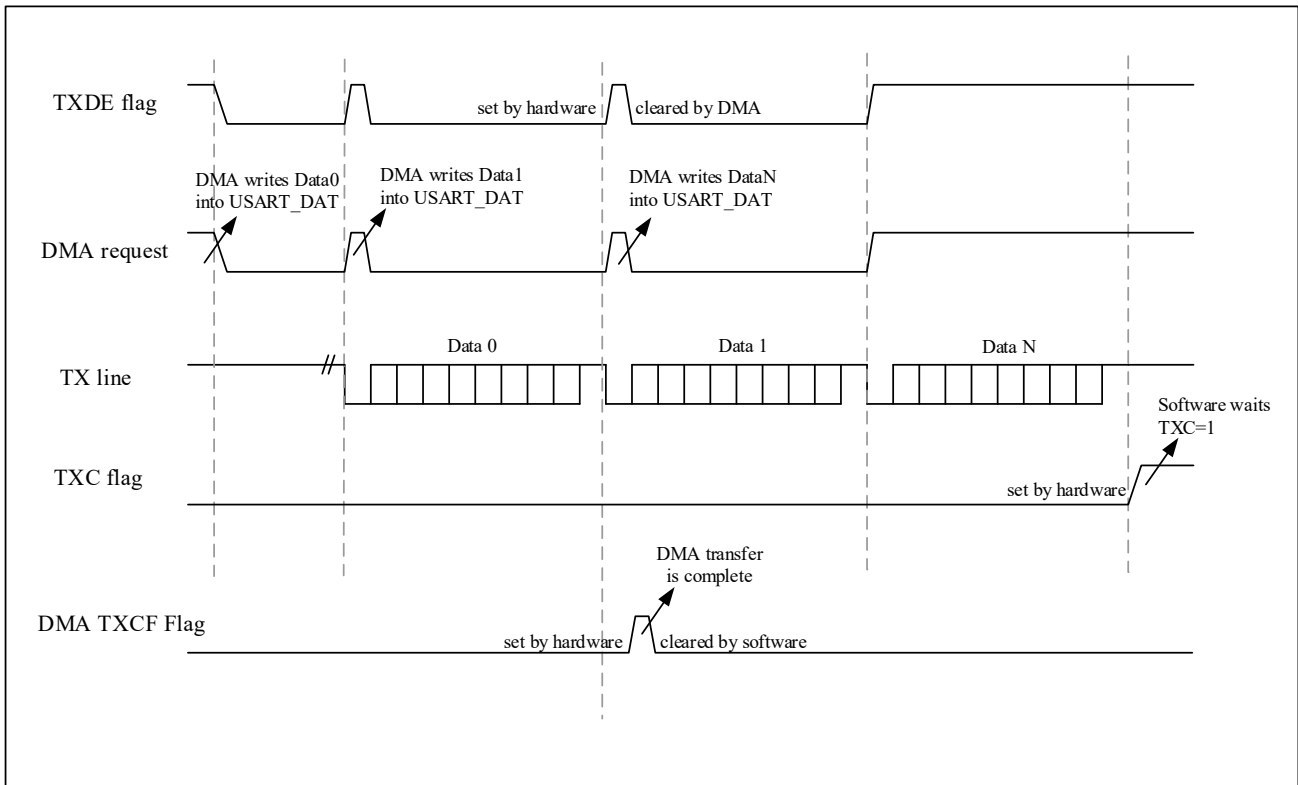
DMA transmission

Set USART_CTRL3.DMATXEN to enable DMA mode for transmission. When the USART's transmit shift register is empty (USART_STS.TXDE=1), the DMA will transmit the data from the SRAM to the USART_DAT register of the USART.

When using DMA transmission, the process of configuring the DMA channel is as follows:

1. Set the source address of DMA to the data memory. When a data transmission request occurs, the transmitted data will be read from this address.
2. Set the destination address of DMA to the USART_DAT register. When a data transmission request occurs, this address will be the destination address of the data transmission.
3. Set the total number of bytes to be transmitted.
4. Set the priority of the channel, the circular mode, the incremental mode of peripherals and memory, the data width of peripherals and memory, the interrupt generated by half of the transmission or the interrupt when the transmission is completed.
5. Start the channel.
6. After the data transmission is completed, the transmission complete flag (DMA_INTSTS.TXCFx) is set to 1.

Figure 18-7 Transmission Using DMA



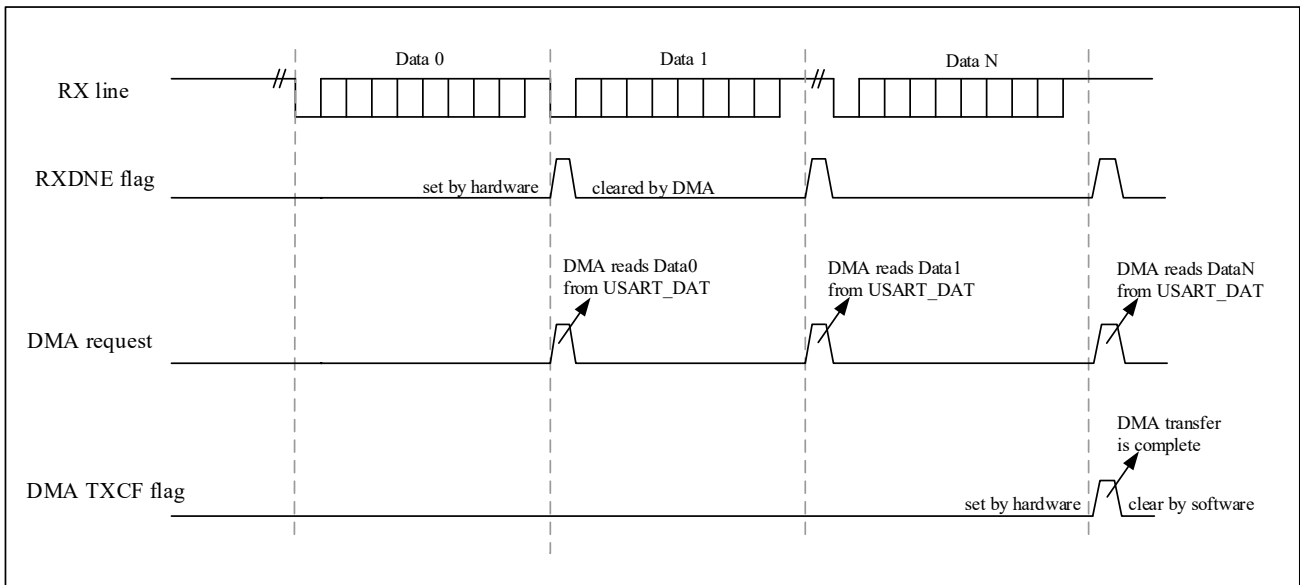
DMA reception

Set USART_CTRL3.DMARXEN to enable DMA mode for reception. When a byte is received (USART_STS.RXDNE=1), the DMA will transmit the data from the USART_DAT register of the USART to the SRAM.

When using DMA reception, the process of configuring the DMA channel is as follows:

1. Set the source address of DMA to the USART_DAT register. When a data transmission request occurs, this address will be the source address of the data transfer.
2. Set the destination address of DMA to the data memory. When a data transmission request occurs, the transmitted data will be written to this address.
3. Set the total number of bytes to be transmitted.
4. Set the priority of the channel, the circular mode, the incremental mode of peripherals and memory, the data width of peripherals and memory, the interrupt generated by half of the transfer or the interrupt when the transmission is completed.
5. Start the current DMA channel.

Figure 18-8 Reception Using DMA

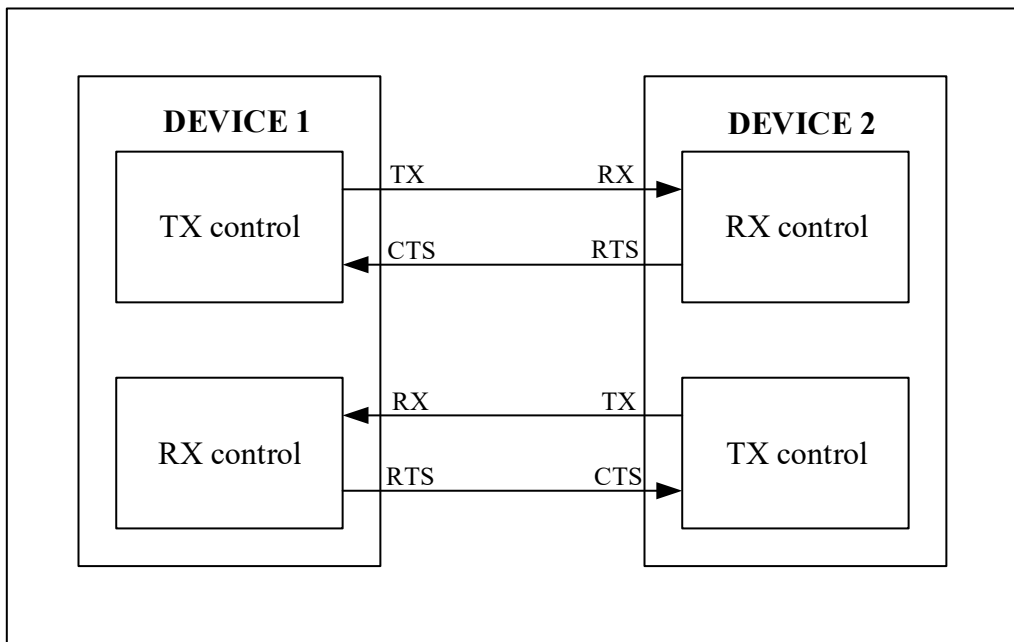


In multi-buffer communication mode, the error flag will be set when there is a frame error, overrun or noise error. An interrupt will be generated if the error interrupt is enabled (USART_CTRL3.ERRIEN=1).

18.4.8 Hardware Flow Control

USART supports hardware flow control. The purpose is to coordinate the timing of the transmitter and receiver to avoid data loss. The connection method is shown in the following figure.

Figure 18-9 Hardware Flow Control Between Two USART

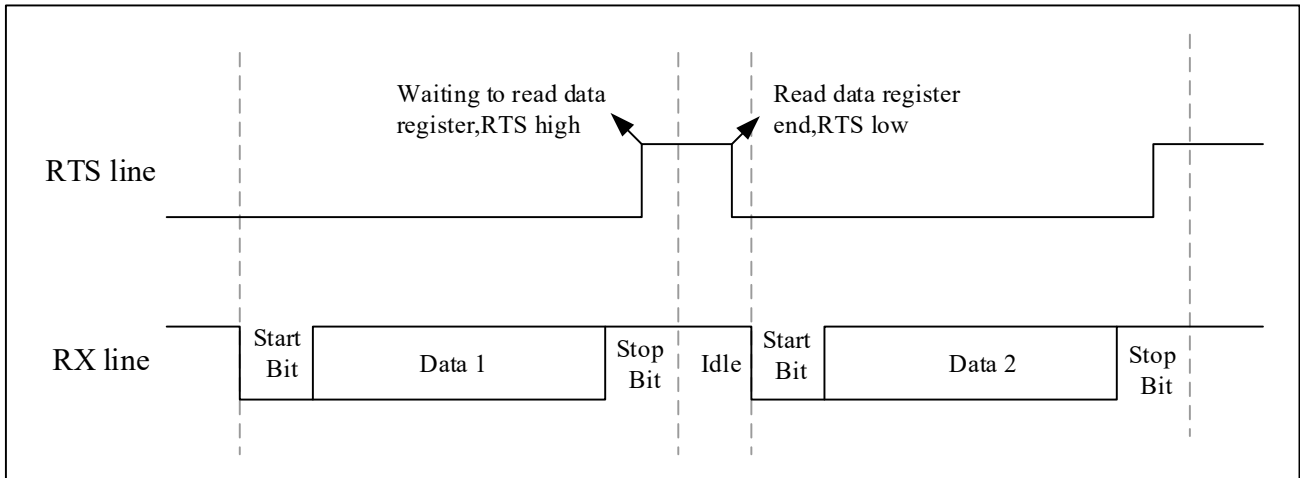


RTS flow control

Set USART_CTRL3.RTSEN to enable RTS. Outputting a low level through nRTS pin indicate that the receiver is ready. When data arrives in RDR, the nRTS outputs a high level, notifying the transmitter to stop data transmission the next

frame of data. If the receiver is ready to receive the next data frame, it will again output a low level through nRTS.

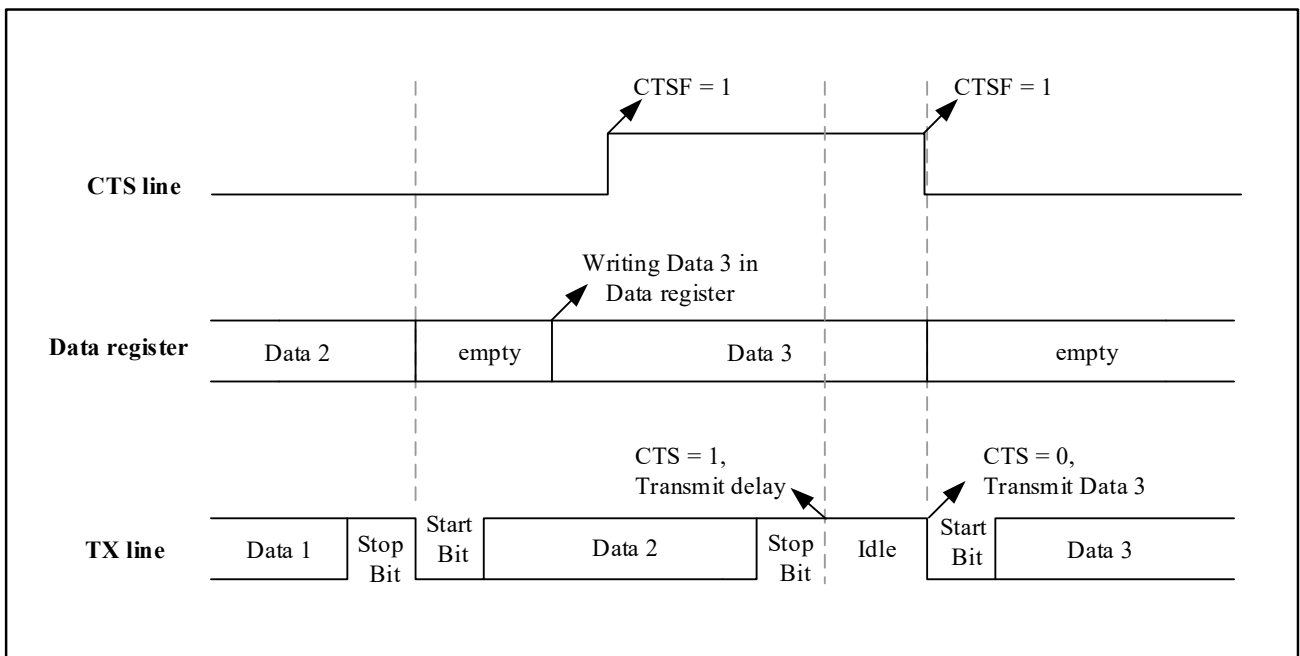
Figure 18-10 RTS Flow Control



CTS flow control

Set USART_CTRL3.CTSEN to enable CTS. nCTS is an input signal, used to judge whether the data can be transmitted to the other device. When the nCTS signal detects a low level, it indicates that the data can be transmitted to the other device. If writing data to the data register when the nCTS signal becomes high level, the data will be held until the nCTS signal becomes low level before transmission starts. If the USART_CTRL3.CTSEN bit is set, the USART_STS.CTSF bit will be set high by hardware when the nCTS input changes state. An interrupt will be generated if USART_CTRL3.CTSIEN is enabled.

Figure 18-11 CTS Flow Controls



18.4.9 Multiprocessor Communication

USART allows multiprocessor communication. When multiple devices are connected to USART for communicate, it is necessary to determine who is the master device, and the remaining processors are all slave devices. The TX output of the

master device is directly connected to the RX port of all slave device. The TX outputs of the slaves are logically AND together and connected to the RX inputs of the master.

When multi-processor communication is performed, the slave devices are all in mute mode, and the host uses a specific method to wake up a slave device to be communicated when needed, so that the slave device is in an active state and transmits data with the master device.

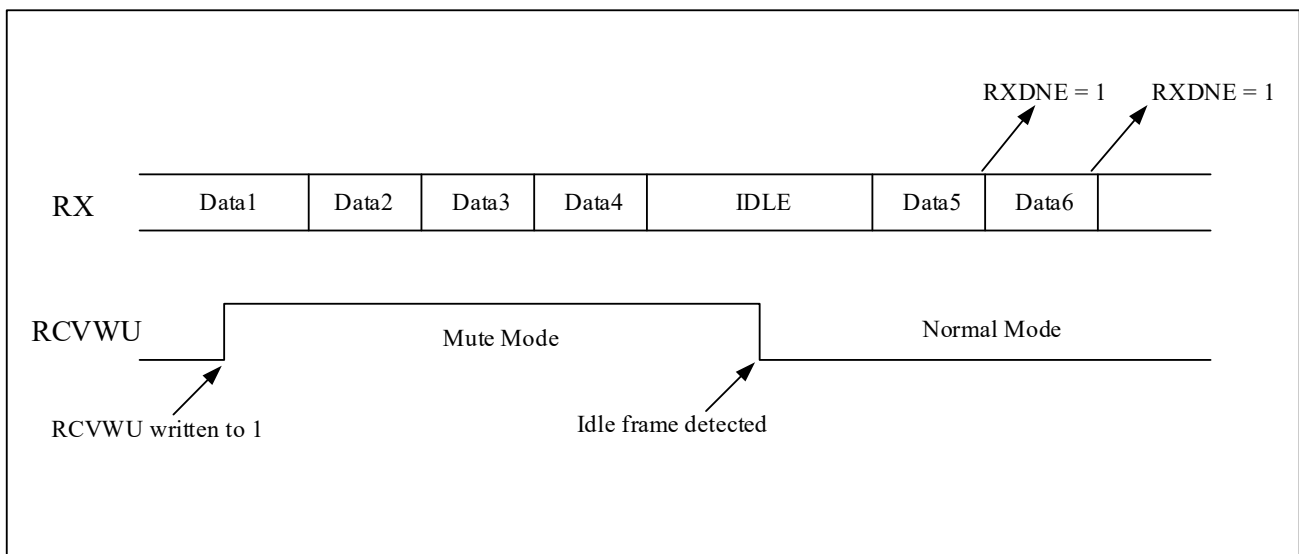
The USART can wake up from mute mode by idle line detection or address mark detection.

Idle line detection

The idle line detection configuration process is as follows:

1. Configure the USART_CTRL1.WUM bit to 0, the USART performs idle line detection;
2. When USART_CTRL1.RCVWU is set (which can be automatically controlled by hardware or written by software under certain conditions), the USART enters mute mode. In mute mode, none of the receive status bits are set, and all receive interrupts are disabled;
3. As shown in Figure 18-12 below, when an idle frame is detected, the USART is woken up, and then USART_CTRL1.RCVWU is cleared by hardware. At this time, USART_STS.IDLEF is not set.

Figure 18-12 Mute Mode Using Idle Line Detection



Address mark detection

The USART performs address mark detection by configuring the USART_CTRL1.WUM bit to 1. The address of the receiver is programmable through the USART_CTRL2.ADDR[3:0] bits. If the MSB is 1, the byte is considered an address, otherwise it is considered data.

In this mode, the USART can enter mute mode by:

- When the receiver is not processing data, USART_CTRL1.RCVWU can be written to 1 by software, and USART enters mute mode;

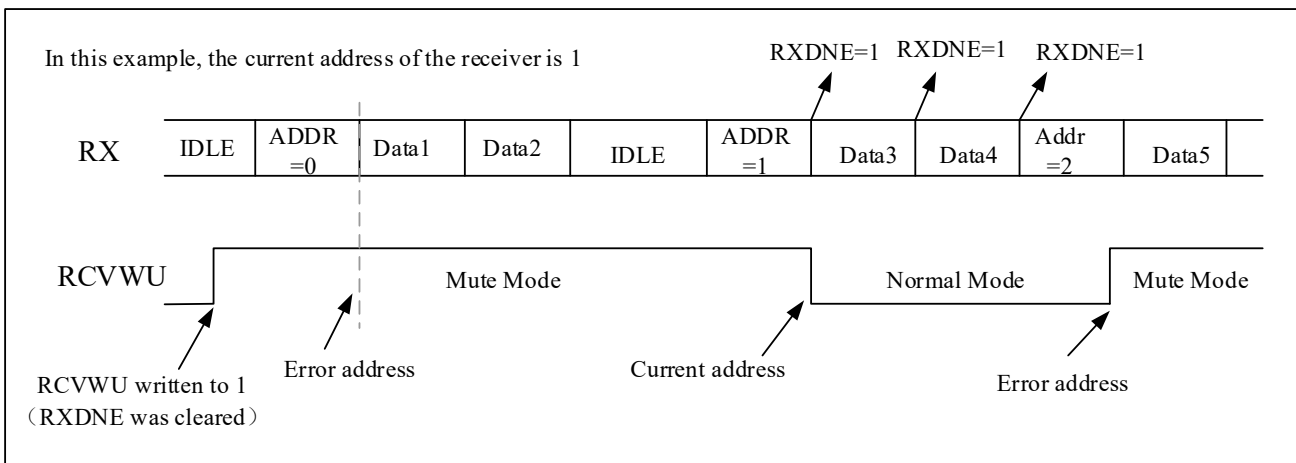
Note: When the receive buffer is empty (RXNE=0 in USART_SR), the USART_CTRL1.RCVWU bit can be written to 0 or 1. Otherwise, the write operation is ignored.

- When the received address does not match the address of the USART_CTRL2.ADDR[3:0] bits, USART_CTRL1.RCVWU is written to 1 by hardware.

In mute mode, none of the receive status bits are set and all receive interrupts are disabled.

When the received address matches the address of the USART_CTRL2.ADDR[3:0] bits, the USART is woken up and USART_CTRL1.RCVWU is cleared. The USART_STS.RXDNE bit will be set. Data can be transmitted normally.

Figure 18-13 Mute Mode Detected Using Address Mark



18.4.10 Synchronous Mode

The USART supports synchronous serial communication. In synchronous mode, the USART only supports the master mode, and cannot use the input clock from other devices to receive and transmit data. Synchronous mode can be enabled by configuring the USART_CTRL2.CLKEN bit.

Note: When using synchronous mode, USART_CTRL2.LINMEN, USART_CTRL3.SCMEN, USART_CTRL3.HDMEN, USART_CTRL3.IRDAMEN, these bits need to be kept clear.

Synchronized clock

The CK pin is the output of synchronous clock. During the bus idle period, before the actual data arrives and when the break symbol is transmitted, the synchronous clock is not output.

Clock phase and polarity are software programmable and need to be configured when both the transmitter and receiver are disabled. When the clock polarity is 0 (USART_CTRL2.CLKPOL=0), the default level of CK is low during the bus idle period; when the clock polarity is 1 (USART_CTRL2.CLKPOL=1), the default level of CLK is high during the bus idle period.

When the phase polarity is 0 (USART_CTRL2.CLKPHA=0), the data is sampled on the first edge of the clock; when the phase polarity is 1 (USART_CTRL2.CLKPHA=1), the data is sampled on the second edge.

During the start and stop bits, the CK pin does not output clock pulses.

The synchronous data cannot be received when no data is transmitted. Because the clock is only available when the transmitter is activated and data is written to the USART_DAT register.

The USART_CTRL2.LBCLK bit controls whether to output the clock pulse corresponding to MSB transmitted on the CK pin. This bit only to be configured when both the transmitter and receiver are disabled. If USART_CTRL2.LBCLK is 1, the clock pulse of the last bit of data will be output from CK. If USART_CTRL2.LBCLK is 0, the clock pulse of the last

bit of data is not output from CK.

Synchronous transmitting

Data transmission in synchronous mode is the same as in asynchronous mode. Data is transmitted out from TX pin and the clock pluses are output from the CK pin.

Synchronous receiving

Data reception in synchronous mode is different from asynchronous mode. Data is sampled at the active clock edge from CK pin, without oversampling. But setup time and hold time (depending on baud rate, 1/16 of a data bit period) must be considered.

Figure 18-14 USART Synchronous Transmission Example

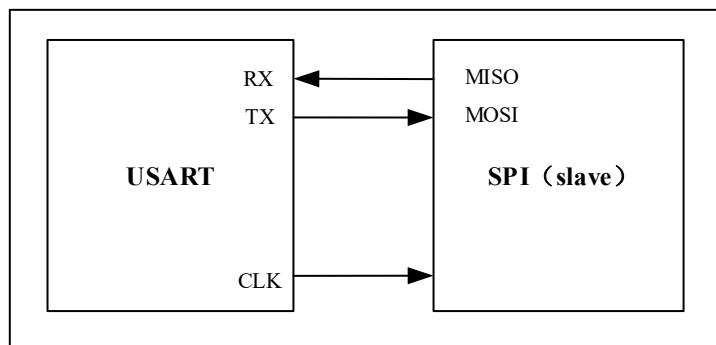


Figure 18-15 USART Data Clock Timing Example (WL=0)

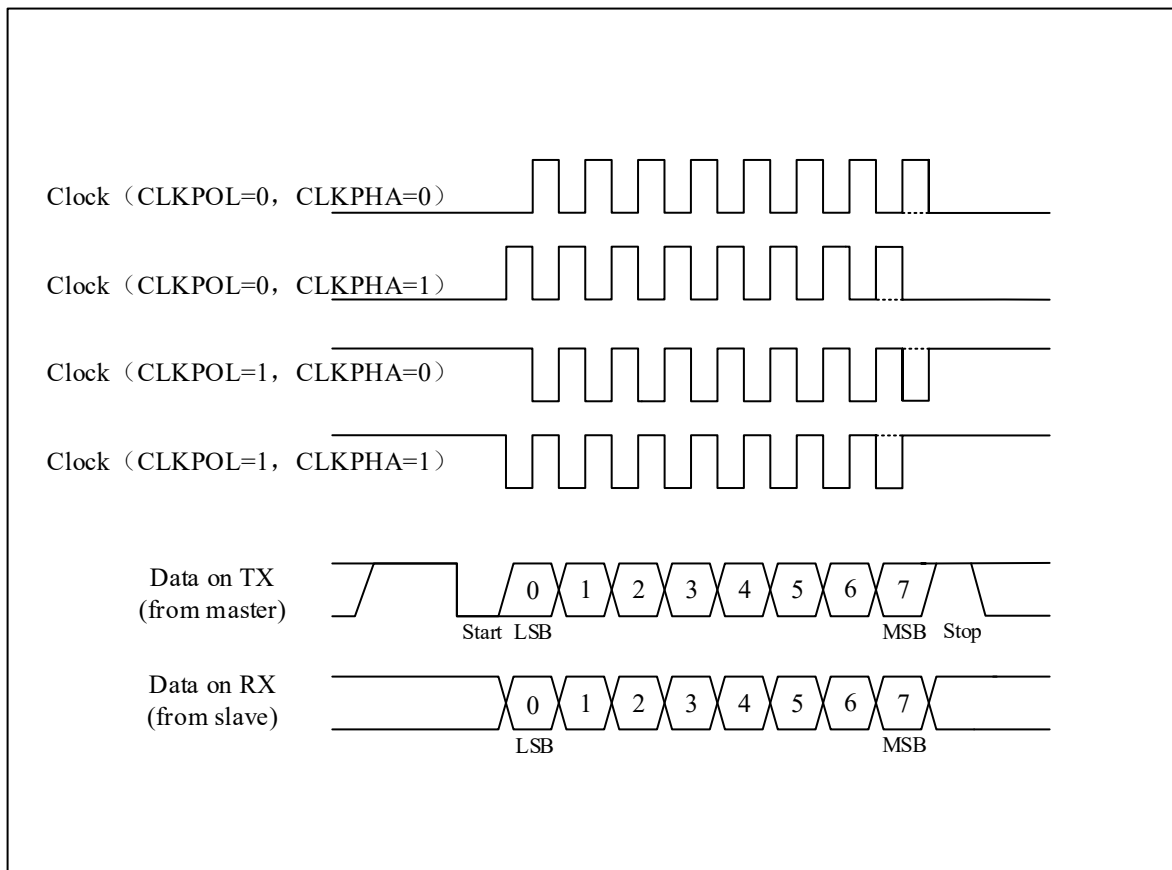


Figure 18-16 USART Data Clock Timing Example (WL=1)

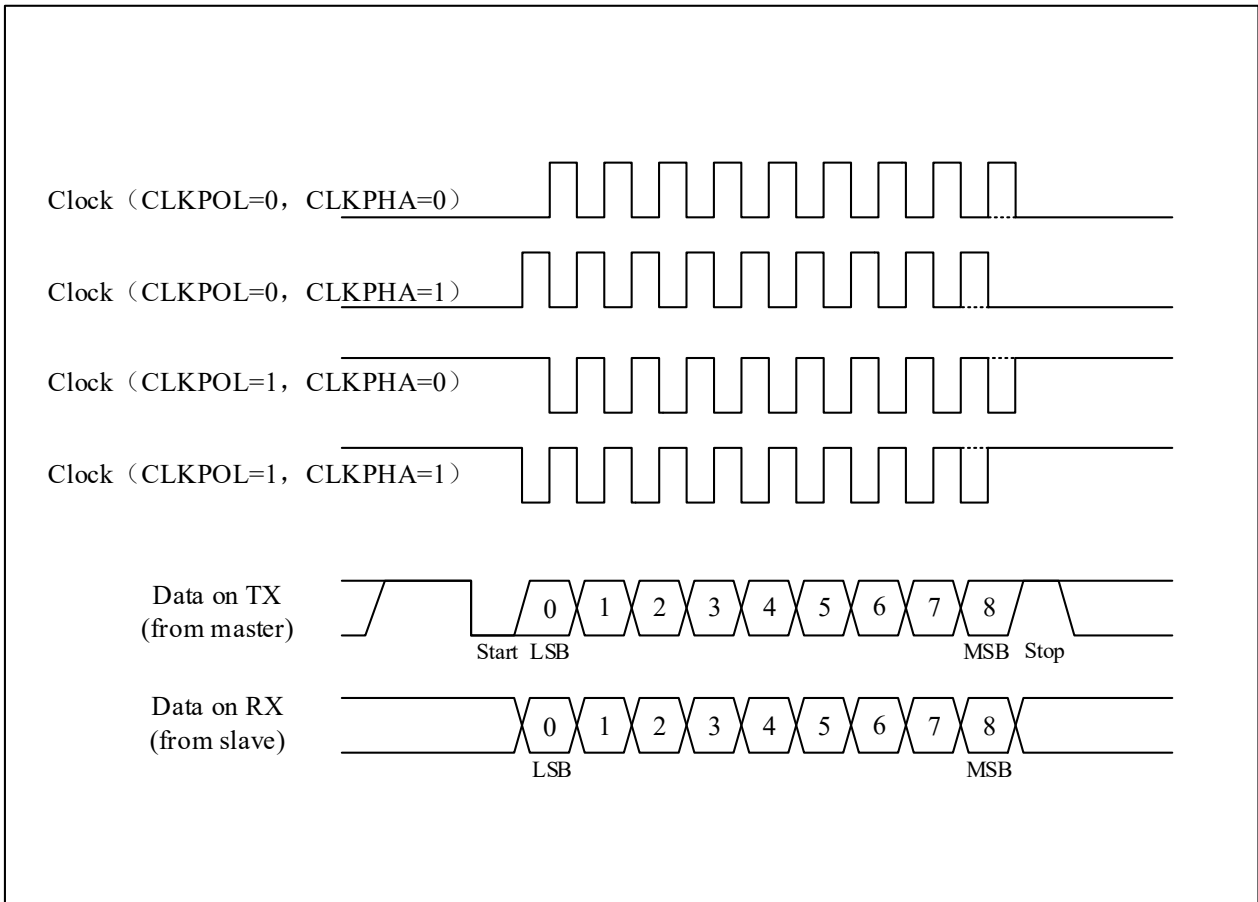
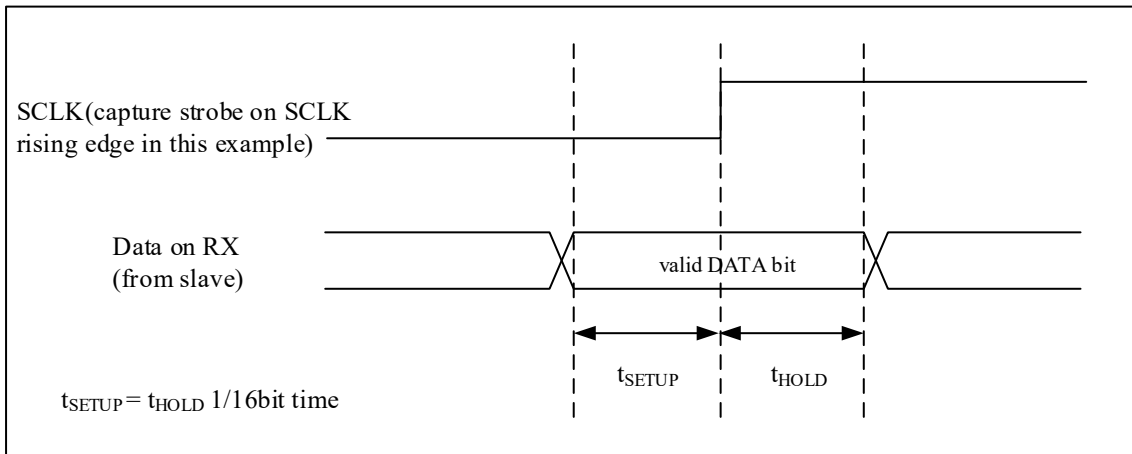


Figure 18-17 RX Data Sampling / Holding Time



Note: the function of CK is different in Smartcard mode, please refer to the Smartcard mode section for details.

18.4.11 Single-wire Half-duplex Mode

USART supports single-wire half-duplex communication, allowing data to be transmitted in both directions, but only one-way data reception or transmission can occur at the same time. Communication conflicts are managed by software.

Choose single-wire half-duplex mode by setting the USART_CTRL3.HDMEN bit. USART_CTRL2.CLKEN,

USART_CTRL2.LINMEN, USART_CTRL3.SCMEN, USART_CTRL3.IRDAMEN, these bits should be kept clear in this mode.

After the half-duplex mode is turned on, the TX pin and the RX pin are internally connected, and the external Rx pin is no longer used. When there is no data to transmit, TX pin is released. Therefore, when TX pin is not driven by the USART, it must be configured as a floating input or an open-drain output high level.

18.4.12 Serial IrDA Interface Encoding/Decoding Mode

USART supports the IrDA (Infrared Data Association). Through the USART_CTRL3.IRDAMEN bit, you can choose whether to enable IrDA mode. When using the IrDA mode, the following bits should be kept clear: USART_CTRL2.CLKEN, USART_CTRL2.STPB[1:0], USART_CTRL2.LINMEN, USART_CTRL3.HDMEN, USART_CTRL3.SCMEN. Through the USART_CTRL3.IRDALP bit, it can be used to select the normal mode or low power mode of IrDA.

IrDA normal mode

When USART_CTRL3.IRDALP=0, the IrDA operates in normal mode.

The IrDA is a half-duplex communication interface, so there must be a minimum delay of 10ms between transmission and reception. The data is modulated using return to zero inverted(RZI), which uses an infrared light pulse to represent a logic '0'. The transmitted pulse width is specified as 3/16 of a bit period in normal mode, as shown in the Figure 18-19. The USART baud rate supports up to 115200bps.

The USART transmits data to the SIR encoder for modulation before outputting it. The modulated data stream is transmitted to an external infrared transmitter for transmission. When receiving, the data is first received and demodulated by the external infrared receiver, then forwarded to the SIR decoder for decoding before being transmitted to the USART.

The transmit encoder output has opposite polarity to the receive decoder input. When idle, SIR transmit is low, while SIR receive is high. The high pulse transmitted by SIR is '0' and the low level is '1', while SIR reception is the opposite.

when the USART is transmitting data to the IrDA transmit encoder, the IrDA receive decoder will ignore all data on the IrDA receive line. when the USART is receiving data transmitted from the SIR receiver decoder, the data transmitted by the USART to the IrDA transmitter encoder will not be encoded.

Pulse width is programmable. The IrDA specification requires pulses to be wider than 1.41us. For pulse widths less than 2 cycles, the receiver will filter them out and lost. PSCV is the prescaler value programmed in the USART_GTP register.

IrDA low power mode

When USART_CTRL3.IRDALP=1, The IrDA operates in low power mode.

When transmitting data in low power mode, the pulse width is 3 times the PSCV. The minimum clock frequency after PSCV division is 1.42MHz, with a typically value of 1.8432MHz (1.42 MHz < PSC < 2.12 MHz).

For the receiver, the requirement for a valid signal is that the duration of the low level signal must be greater than 2 cycles of the IrDA low power baud rate clock.

Figure 18-18 Irdasirendec-block Diagram

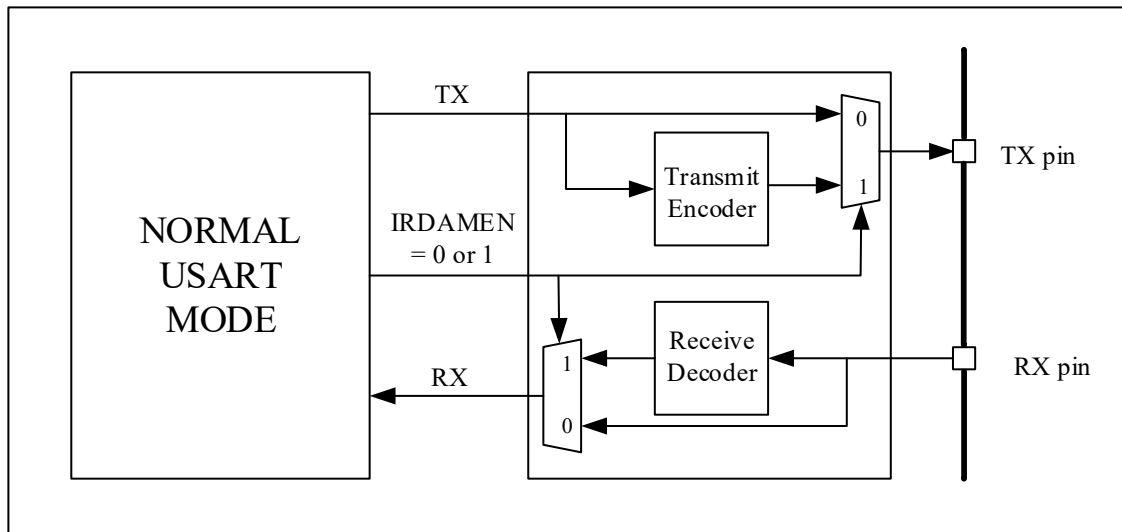
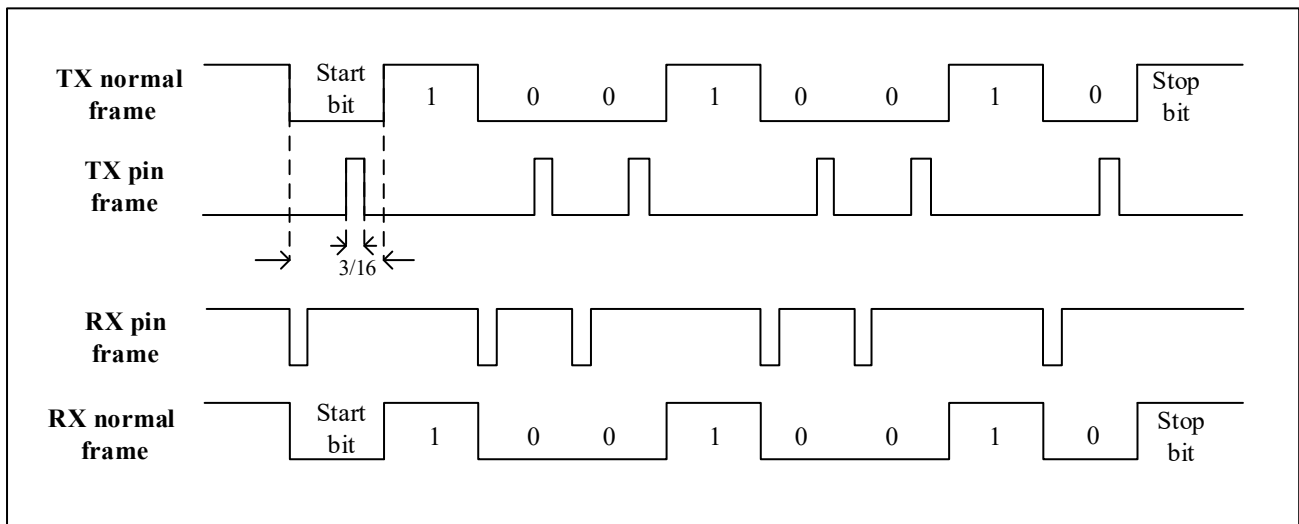


Figure 18-19 Irda Data Modulation (3/16)-normal Mode



18.4.13 LIN Mode

USART supports the ability of a LIN (Local interconnection Network) master to send a synchronization break and the ability of a LIN slave to detect a break. LIN mode can be enabled by configuring the USART_CTRL2.LINMEN bit.

Note: When using LIN mode, USART_CTRL2.STPB[1:0], USART_CTRL2.CLKEN, USART_CTRL3.SCMEN, USART_CTRL3.HDMEN, USART_CTRL3.IRDAMEN, these bits should be kept clear.

LIN transmission

When transmission data in LIN mode, the length of the data bits transmitted can only be 8 bits. By setting USART_CTRL1.SDBRK, a 13-bit '0' will be transmitted as the break frame, and insert a stop bit.

LIN reception

When the bus is idle or during the transmission of a data frame, as long as the break frame appears, it can be detected. The break frame detection is independent of the USART receiver.

By configuring the USART_CTRL2.LINBDL bit, the active low level for break frame detection can be selected as 10-bit

or 11-bit.

When the receiver detects the start bit, the circuit samples each subsequent bit at the 8th, 9th, and 10th oversampling clock points of each bit. if 10th or 11th consecutive bits are detected as '0' and followed by a delimiter, it means that a LIN break frame is detected, and USART_STS.LINBDF is set. Before confirming the break frame, check the delimiter as it means the RX line has gone back to high level. An interrupt is generated if the LIN break frame detection interrupt (USART_CTRL2.LINBDIEN) is enabled.

If a '1' is sampled before the 10th or 11th sample point, the current detection is canceled and the start bit is searched again.

Figure 18-20 Break Detection in LIN Mode (11-Bit Break Length-The LINBDL Bit Is Set)

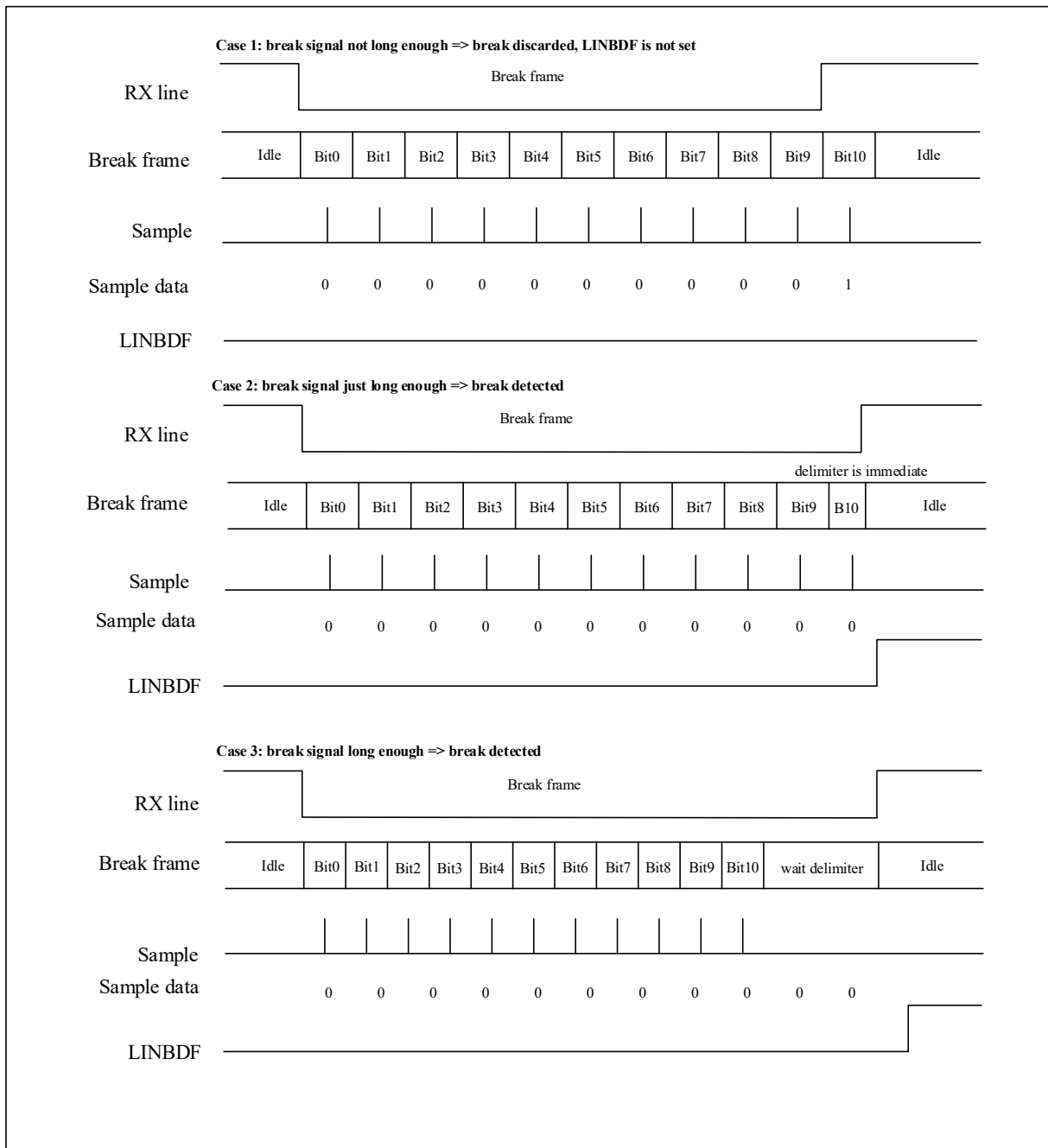
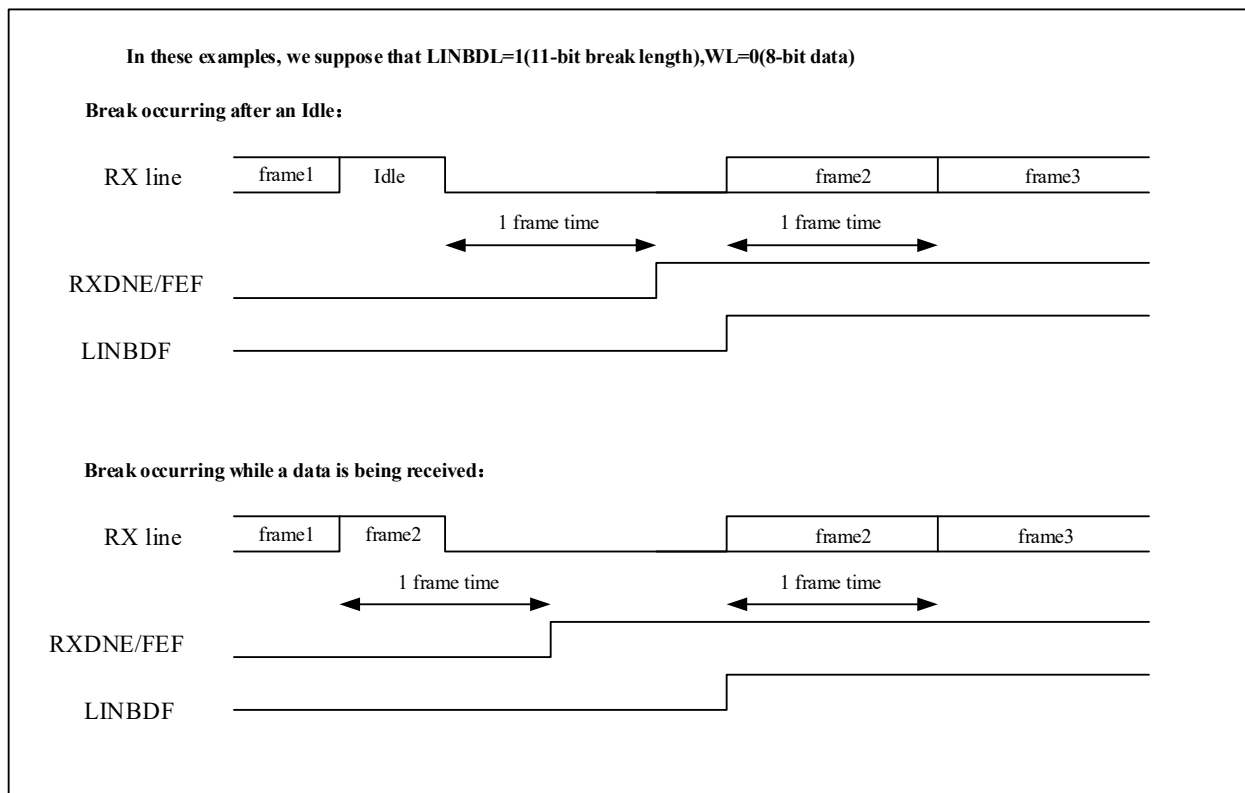


Figure 18-21 Break Detection and Framing Error Detection in LIN Mode



18.4.14 Smartcard Mode (ISO7816)

USART supports smartcard protocol. The smartcard interface supports the asynchronous smartcard protocol defined in the ISO7816-3 standard.

Through the USART_CTRL3.SCMEN bit, you can choose whether to enable smartcard mode. When using smartcard mode, USART_CTRL2.LINMEN, USART_CTRL3.HDMEN, USART_CTRL3.IRDAMEN, these bits should be kept clear.

In smartcard mode, the USART can provide a clock through the CK pin. The system clock is divided by the prescaler register to provide the clock to the smartcard. The CK frequency can be from $f_{CK}/2$ to $f_{CK}/62$, where f_{CK} is the peripheral input clock.

In smartcard mode, 0.5 and 1.5 stop bits can be used when receiving data, and only 1.5 stop bits can be used when transmitting data. So it is recommended to use 1.5 stop bits for both transmission and reception to avoid frequent switching between two stop bit lengths.

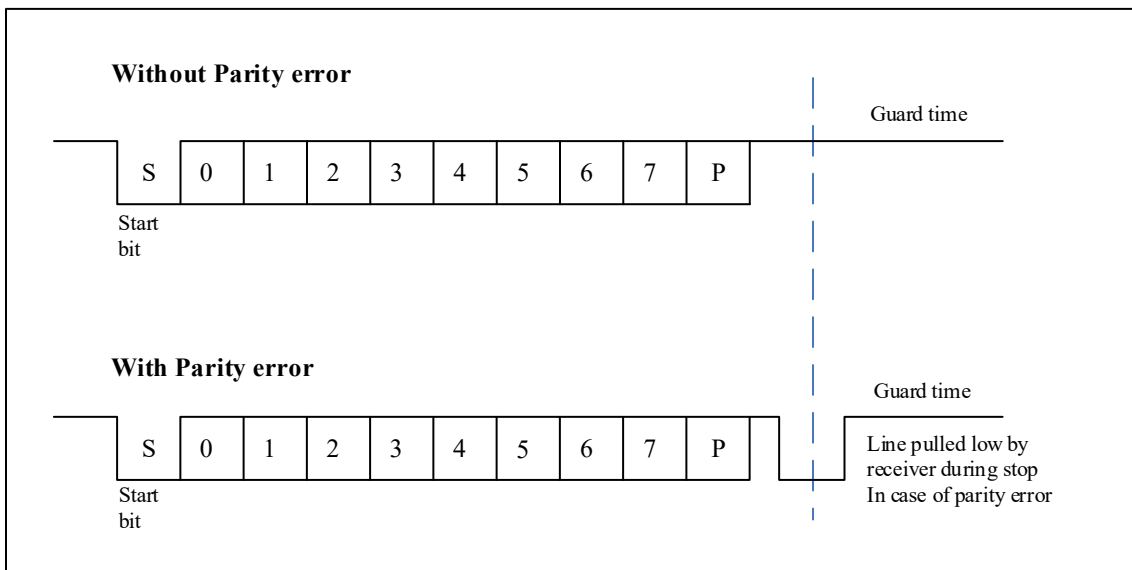
In smartcard mode, the data bits should be configured as 8 bits, and the parity bit need be configured.

When a parity error is detected by receiver, the transmit data line is pulled low for one baud clock cycle at the end of the stop bit as NACK signal (if USART_CTRL3.SCENACK is set). This NACK signal will generate a framing error on the transmitter side (transmitter side is configured with 1.5 stop bits).

When the transmitter receives a NACK signal (framing error) from the receiver, it does not detect the NACK as a start bit (according to the ISO protocol, the duration of the received NACK can be 1 or 2 baud clock cycles).

The example given in the following figure illustrates the signal on the data line with and without parity errors.

Figure 18-22 Iso7816-3 Asynchronous Protocol



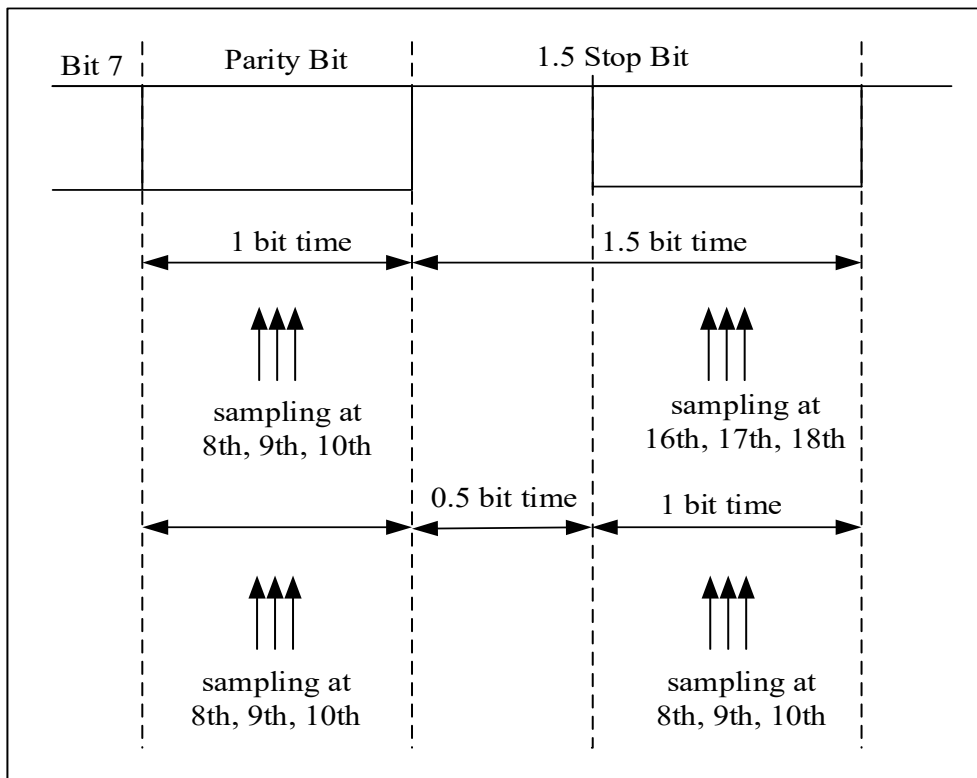
The break frame is not supported in smartcard mode. A 00h data with a framing error will be treated as data instead of a break frame.

In normal mode, data will be shifted out of the transmit shift register on the next baud clock. In smartcard mode is delayed by a minimum of 1/2 baud clock than normal mode.

In normal mode, USART_STS.TXC is set when a frame containing data is transmitted and USART_STS.TXDE=1. In smartcard mode, the transmission completion flag (USART_STS.TXC) is set high when the guard time counter reaches the value (USART_GTP.GTV[7:0]) and the frame containing data is transmitted. The clearing of the USART_STS.TXC flag is not affected by the smartcard mode.

The following figure details how USART samples NACK signals.

Figure 18-23 Use 1.5 Stop Bits to Detect Parity Errors



18.5 Interrupt Request

The various interrupt events of USART are logical OR relations, if the corresponding enable control bit is set, these events can generate their own interrupts, but only one interrupt request can be generated at the same time.

Table 18-7 USART Interrupt Request

Interrupt Function	Interrupt Event	Event Flag	Enable Bit
USART global interrupt	Transmission data register is empty.	TXDE	TXDEIEN
	CTS flag	CTSF	CTSIEN
	Transmission complete	TXC	TXCIEN
	Receive data ready to be read	RXDNE	RXDNEIEN
	Data overrun error detected.	ORERR	
	Idle line detected	IDLEF	IDLEIEN
	Parity error	PEF	PEIEN
	Disconnect flag	LINBDF	LINBDIEN
Noise, overrun error and framing error in multi-buffer communication	NEF/OREF/FEF	ERRIEN ⁽¹⁾	

Note: ⁽¹⁾ This flag bit is used only when DMA is used to receive data(USART_CTRL3.DMARXEN=1).

18.6 Mode Support

Table 18-8 USART Mode Setting ⁽¹⁾

Communication Mode	USART1	USART2
Asynchronous mode	Y	Y
Hardware flow control mode	Y	Y
DMA communication mode	Y	Y
Multiprocessor	Y	Y
Synchronous mode	Y	Y
Smartcard mode	Y	Y
Single-wire half duplex mode	Y	Y
IrDA infrared mode	Y	Y
LIN	Y	Y

Note: ⁽¹⁾ Y = support this mode, N = do not support this mode

18.7 USART Register

18.7.1 USART Register Overview

Table 18-9 USART Register Overview

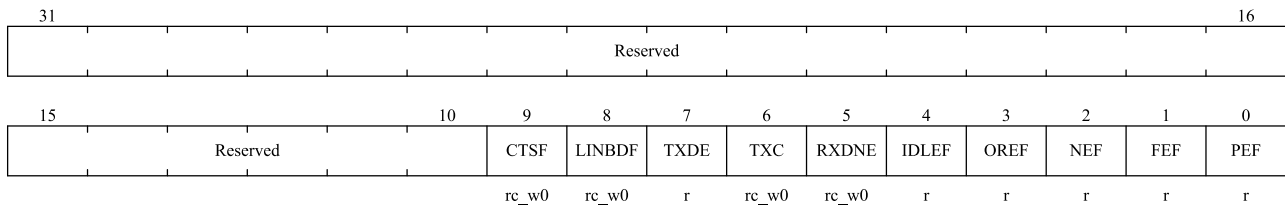
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0													
000h	USART_STS	Reserved																						CTSF	LINBDF	TXDE	TXC	RXDNE	IDLEF	OREF	NEF	FEF	PEF													
	Reset Value																							0	0	1	1	0	0	0	0	0	0													
004h	USART_DAT	Reserved																						DATV[8:0]																						
	Reset Value																							0	0	0	0	0	0	0	0	0	0													
008h	USART_BRCF	Reserved												DIV_Integer[11:0]									DIV_Decimal [3:0]																							
	Reset Value													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								
00Ch	USART_CTRL1	Reserved																		UEN	WL	WUM	PCEN	PSEL	PEIEN	TXDEIEN	TXCIEN	RXDNEIEN	IDLEIEN	TXEN	RXEN	RCVWU	SDBRK													
	Reset Value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
010h	USART_CTRL2	Reserved														LINMEN	STPB [1:0]		CLKEN	CLKPOL	CLKPHA	LBCLK	Reserved	LINBDIEN	LINBDL	Reserved	ADDR[3:0]													
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
014h	USART_CTRL3	Reserved																		CTSIEEN	CTSEN	RTSEN	DMATXEN	DMARXEN	SCMEN	SCNACK	HDMEN	IRDALP	IRDAMEN	ERRIEN										
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0							
018h	USART_GTP	Reserved														GTV[7:0]				PSCV[7:0]																				
	Reset Value	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

18.7.2 USART Status Register (USART_STS)

Address offset : 0x00

Reset value : 0x0000 00C0



Bit Field	Name	Description
31:10	Reserved	Reserved, the reset value must be maintained
9	CTSF	<p>CTS flag</p> <p>If USART_CTRL3.CTSEN bit is set, this bit is set by hardware when the nCTS input changes. If USART_CTRL3.CTSIEN bit is set, an interrupt will be generated.</p> <p>This bit is cleared by software.</p> <p>0:nCTS status line has not changed.</p> <p>1:nCTS status line changes.</p> <p><i>Note: This bit is invalid for UART5/6.</i></p>
8	LINBDF	<p>LIN break detection flag.</p> <p>If USART_CTRL2.LINMEN bit is set, this bit is set by hardware when LIN disconnection is detected. If USART_CTRL2.LINBDIEN bit is set, an interrupt will be generated.</p> <p>This bit is cleared by software.</p> <p>0: LIN break character not detected.</p> <p>1: LIN break character detected.</p>
7	TXDE	<p>The Transmit data register empty.</p> <p>Set to 1 after power-on reset or data to be sent has been sent to the shift register. Setting USART_CTRL1.TXDEIEN will generate an interrupt.</p> <p>This bit is cleared to 0 when the software writes the data to be sent into USART_DAT.</p> <p>0: Transmit data buffer is not empty.</p> <p>1: The transmitting data buffer is empty.</p>
6	TXC	Transmission complete.

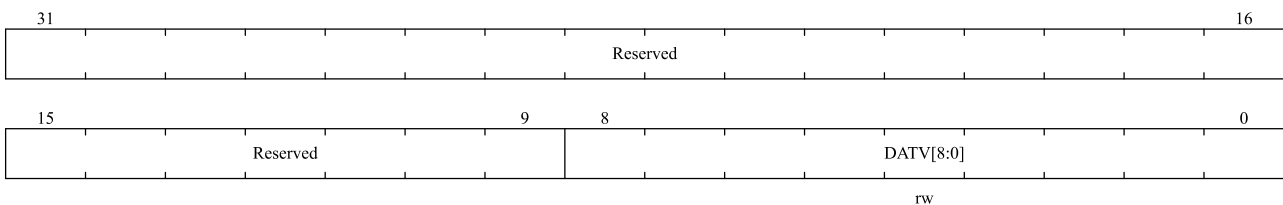
Bit Field	Name	Description
		<p>This bit is set to 1 after power-on reset. If USART_STS.TXDE is set, this bit is set when the current data transmission is completed.</p> <p>Setting USART_CTRL1.TXCIEN bit will generate an interrupt.</p> <p>This bit is cleared by software.</p> <p>0: Transmitting did not complete.</p> <p>1: Send completed.</p>
5	RXDNE	<p>The Read data register not empty.</p> <p>This bit is set when the read data buffer receives data from the shift register. When USART_CTRL1.RXDNEIEN bit is set, an interrupt will be generated.</p> <p>Software can clear this bit by writing 0 to it or reading the USART_DAT register.</p> <p>0: The read data buffer is empty.</p> <p>1: The read data buffer is not empty.</p>
4	IDLEF	<p>IDLE line detected flag.</p> <p>Within one frame time, the idle state is detected at the RX pin, and this bit is set to 1. When USART_CTRL1.IDLEIEN bit is set, an interrupt will be generated.</p> <p>The software can clear this bit by reading USART_STS first and then reading USART_DAT.</p> <p>0: No idle frame detected.</p> <p>1: idle frame detected.</p> <p><i>Note: IDLEF bit will not be set high again until USART_STS.RXDNE bit is set (that is, an idle line is detected again).</i></p>
3	OREF	<p>Overrun error</p> <p>With RXDNE set, this bit is set if the USART_DAT register receives data from the shift register. When USART_CTRL3.ERRIEN bit is set, an interrupt will be generated.</p> <p>The software can clear this bit by reading USART_STS first and then reading USART_DAT.</p> <p>0: No overrun error was detected.</p> <p>1: Overflow error detected.</p>
2	NEF	<p>Noise error flag.</p> <p>When noise is detected in the received frame, this bit is set by hardware. It is cleared by the software sequence (read first USART_STS, read USART_DAT again).</p> <p>0: No noise error detected.</p> <p>1: Noise error detected.</p> <p><i>Note: this bit will not generate an interrupt because it appears with USART_STS.RXDNE, and the hardware will generate an interrupt when setting the USART_STS.RXDNE flag. In the multi-buffer communication mode, if the USART_CTRL3.ERRIEN bit is set, an interrupt will be generated when the NEF flag is set.</i></p>
1	FEF	<p>Framing error.</p> <p>When the data is not synchronized or a large amount of noise is detected, and the stop bit is not received and recognized at the expected time, it will be judged that a framing error has been detected, and this bit will be set to 1. First read USART_STS, then read USART_DAT can cleared this bit.</p>

Bit Field	Name	Description
		0: No framing errors were detected. 1: A framing error or a Break Character is detected. <i>Note: this bit will not generate an interrupt because it appears with USART_STS.RXDNE, and the hardware will generate an interrupt when setting the USART_STS.RXDNE flag. If the currently transmitted data has both framing errors and overload errors, the hardware will continue to transmit the data and only set the USART_STS.OREF flag bit.</i> <i>In the multi-buffer communication mode, if the USART_CTRL3.ERRIEN bit is set, an interrupt will be generated when the FEF flag is set.</i>
0	PEF	Parity error. This bit is set when the parity bit of the received data frame is different from the expected check value. The software can clear this bit by reading USART_STS first and then reading USART_DAT. 0: No parity error was detected. 1: Parity error detected.

18.7.3 USART Data Register (USART_DAT)

Address offset : 0x04

Reset value : undefined (uncertain value)



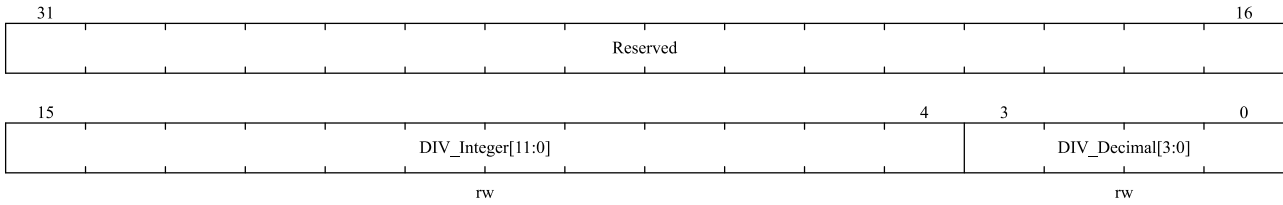
Bit Field	Name	Description
31:9	Reserved	Reserved, the reset value must be maintained
8:0	DATV[8:0]	Data value Contains the data sent or received; Software can change the transmitted data by writing these bits, or read the values of these bits to obtain the received data. If parity is enabled, when the transmitted data is written into the register, the highest bit of the data (the 7th or 8th bit depends on USART_CTRL1.WL bit) will be replaced by the parity bit.

18.7.4 USART Baud Rate Register (USART_BRCF)

Address offset : 0x08

Reset value : 0x0000 0000

Note: When USART_CTRL1.UEN=1, this register cannot be written; The baud counter stops counting if USART_CTRL1.TXEN or USART_CTRL1.RXEN are disabled respectively.

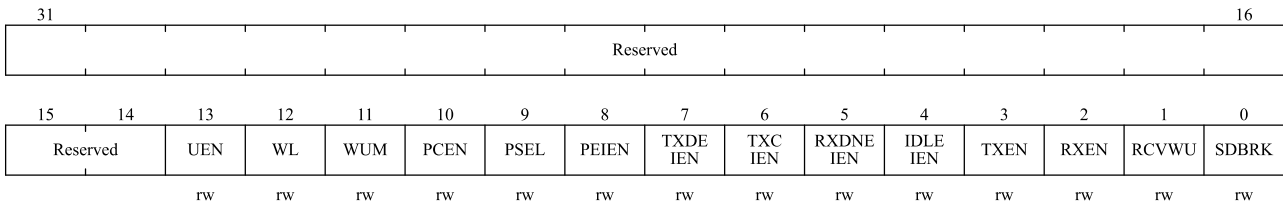


Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained
15:4	DIV_Integer [11:0]	Integer part of baud rate divider.
3:0	DIV_Decimal[3:0]	Fractional part of baud rate divider.

18.7.5 USART Control Register 1 Register (USART_CTRL1)

Address offset : 0x0C

Reset value : 0x0000 0000



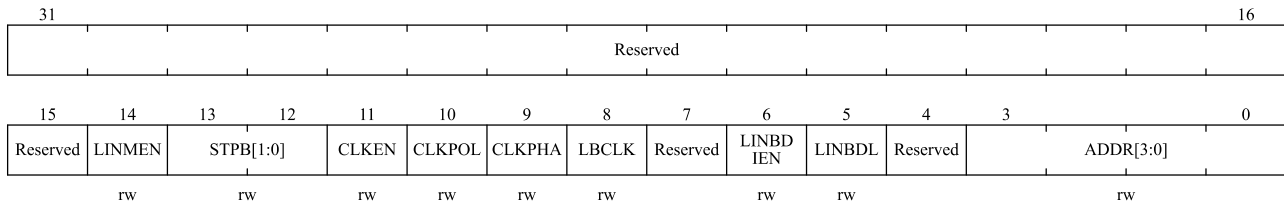
Bit Field	Name	Description
31:14	Reserved	Reserved, the reset value must be maintained
13	UEN	USART enable When this bit is cleared, the divider and output of USART stop working after the current byte transmission is completed to reduce power consumption. Software can set or clear this bit. 0:USART is disabled. 1:USART is enabled.
12	WL	Word length. 0:8 data bits. 1:9 data bits. <i>Note: If data is in transfer, this bit cannot be configured.</i>
11	WUM	Wake up mode from mute mode. 0: Idle frame wake up. 1: Address identifier wake up.
10	PCEN	Parity control enable 0: Parity control is disabled. 1: Parity control is enabled.
9	PSEL	Parity selection. 0: even check. 1: odd check.
8	PEIEN	PE interrupt enable

Bit Field	Name	Description
		<p>If this bit is set to 1, an interrupt is generated when USART_STS.PEF bit is set.</p> <p>0: Parity error interrupt is disabled.</p> <p>1: Parity error interrupt is enabled.</p>
7	TXDEIEN	<p>TXDE interrupt enable</p> <p>If this bit is set to 1, an interrupt is generated when USART_STS.TXDE bit is set.</p> <p>0: Send buffer empty interrupt is disabled.</p> <p>1: Send buffer empty interrupt is enabled.</p>
6	TXCIEN	<p>Transmit complete interrupt enable.</p> <p>If this bit is set to 1, an interrupt is generated when USART_STS.TXC is set.</p> <p>0: Transmission completion interrupt is disabled.</p> <p>1: Transmission completion interrupt is enabled.</p>
5	RXDNEIEN	<p>RXDNE interrupt enable</p> <p>If this bit is set to 1, an interrupt is generated when USART_STS.RXDNE or USART_STS.OREF is set.</p> <p>0: Data buffer non-empty interrupt o and overrun error interrupt are disabled.</p> <p>1: Data buffer non-empty interrupt o and overrun error interrupt are enabled.</p>
4	IDLEIEN	<p>IDLE interrupt enable.</p> <p>If this bit is set to 1, an interrupt is generated when USART_STS.IDLEF is set.</p> <p>0: IDLE line detection interrupt is disabled.</p> <p>1: IDLE line detection interrupt is enabled.</p>
3	TXEN	<p>Transmitter enable.</p> <p>0: The transmitter is disabled.</p> <p>1: the transmitter is enabled.</p>
2	RXEN	<p>Receiver enable</p> <p>0: The receiver is disabled.</p> <p>1: the receiver is enabled.</p>
1	RCVWU	<p>The receiver wakes up</p> <p>Software can set this bit to 1 to make USART enter mute mode, and clear this bit to 0 to wake up USART.</p> <p>In idle frame wake-up mode (USART_CTRL1.WUM=0), this bit is cleared by hardware when an idle frame is detected. In address wake-up mode (USART_CTRL1.WUM=1), when an address matching frame is received, this bit is cleared by hardware. Or when an address mismatch frame is received, it is set to 1 by hardware.</p> <p>0: The receiver is in normal operation mode.</p> <p>1: The receiver is in mute mode.</p>
0	SDBRK	<p>Send Break Character.</p> <p>The software transmits a break character by setting this bit to 1.</p> <p>This bit is cleared by hardware during stop bit of the break frame transmission.</p> <p>0: No break character was sent.</p> <p>1: Send a break character.</p>

18.7.6 USART Control Register 2 Register (USART_CTRL2)

Address offset : 0x10

Reset value : 0x0000 0000



Bit Field	Name	Description
31:15	Reserved	Reserved, the reset value must be maintained
14	LINMEN	LIN mode enable 0:LIN mode is disabled 1:LIN mode enabled
13:12	STPB[1:0]	STOP bits. 00:1 stop bit. 01:0.5 stop bit. 10:2 stop bit. 11:1.5 stop bit. <i>Note: For UART5/6, only one stop bit and two stop bits are valid.</i>
11	CLKEN	Clock enable 0:CK pin is disabled 1:CK pin enabled <i>Note: This bit is invalid for UART5/6.</i>
10	CLKPOL	Clock polarity. This bit is used to set the polarity of CK pin in synchronous mode. 0: CK pin remains low when it is not transmitted to the outside. 1: CK pin remains high when it is not sent to the outside. <i>Note: This bit is invalid for UART5/6.</i>
9	CLKPHA	Clock phase. This bit is used to set the phase of CK pin in synchronous mode. 0: Sample the first data at the first clock edge. 1: Sample the first data at the second clock edge. <i>Note: This bit is invalid for UART5/6.</i>
8	LBCLK	The Last bit clock pulse. This bit is used to set whether the clock pulse corresponding to the last transmitted data byte (MSB) is output on CK pin in synchronous mode. 0: The clock pulse of the last bit of data is not output from CK. 1: The clock pulse of the last bit of data will be output from CK. <i>Note: This bit is invalid for UART5/6.</i>
7	Reserved	Reserved, the reset value must be maintained
6	LINBDIEN	LIN break detection interrupt enable. If this bit is set to 1, an interrupt will be generated when USART_STS.LINBDF bit is set. 0: Disconnect signal detection interrupt is disabled.

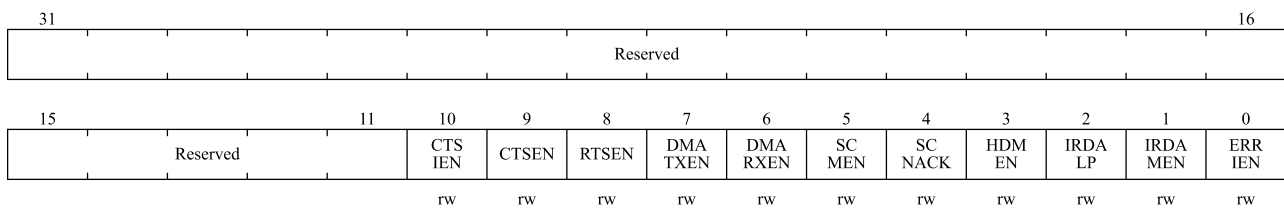
Bit Field	Name	Description
		1: Turn-off signal detection interrupt enabled
5	LINBDL	LIN break detection length. This bit is used to set the length of the break frame. 0:10 bit break detection 1:11 bit break detection <i>Note: LINBDL can be used to control the detection length of Break Characters in LIN mode and other modes, and the detection length is the same as that in LIN mode.</i>
4	Reserved	Reserved, the reset value must be maintained
3:0	ADDR[3:0]	USART address. Used in the mute mode of multiprocessor communication, using address identification to wake up a USART device. In address wake-up mode (USART_CTRL1.WUM=1), if the lower four bits of the received data frame are not equal to the ADDR[3:0] value, USART will enter the mute mode; If the lower four bits of the received data frame are equal to the ADDR[3:0] value, USART will be awakened.

Note: These three bits (USART_CTRL2.CLKPOL, USART_CTRL2.CLKPHA, USART_CTRL2.LBCLK) cannot be overwritten after enabling transmission.

18.7.7 USART Control Register 3 Register (USART_CTRL3)

Address offset : 0x14

Reset value : 0x0000 0000



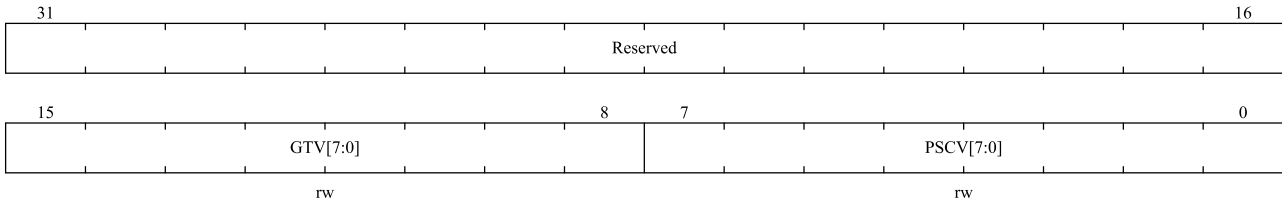
Bit Field	Name	Description
31:11	Reserved	Reserved, the reset value must be maintained
10	CTSIEN	CTS interrupt enable. If this bit is set to 1, an interrupt will be generated when USART_STS.CTSF bit is set. 0:CTS interrupt is disabled. 1:CTS interrupt is enabled. <i>Note: This bit is invalid for UART5/6.</i>
9	CTSEN	CTS enable. This bit is used to enable the CTS hardware flow control function. 0:CTS hardware flow control is disabled. 1:CTS hardware flow control is enabled. <i>Note: This bit is invalid for UART5/6.</i>
8	RTSEN	RTS enable. This bit is used to enable RTS hardware flow control function.

Bit Field	Name	Description
		0:RTS hardware flow control is disabled. 1:RTS hardware flow control is enabled. <i>Note: This bit is invalid for UART5/6.</i>
7	DMATXEN	DMA transmitter enable. 0:DMA transmission mode is disabled. 1:DMA transmission mode is enabled.
6	DMARXEN	DMA receiver enable. 0:DMA receive mode is disabled. 1:DMA receive mode is enabled.
5	SCMEN	Smartcard mode enable. This bit is used to enable Smartcard mode. 0: Smartcard mode is disabled. 1: Smartcard mode is enabled. <i>Note: This bit is invalid for UART5/6.</i>
4	SCNACK	Smartcard NACK enable. This bit is used for Smartcard mode to enable transmitting NACK when parity error occurs. 0: Do not send NACK when there is a parity error. 1: send NACK when there is a parity error. <i>Note: This bit is invalid for UART5/6.</i>
3	HDMEN	Half-duplex mode enable. This bit is used to enable half-duplex mode. 0: Half-duplex mode is disabled. 1: Half-duplex mode is enabled.
2	IRDALP	IrDA low-power mode. This bit is used to select the low power consumption mode for IrDA mode. 0: Normal mode. 1: Low power mode.
1	IRDAMEN	IrDA mode enable. 0:IrDA is disabled. 1:IrDA is enabled.
0	ERRIEN	Error interrupt enable. When DMA receive mode (USART_CTRL3.DMARXEN=1) is enabled, an interrupt will be generated when USART_STS.FEF, USART_STS. OREF or USART_STS. NEF bit is set. 0: Error interrupt is disabled. 1: Error interrupt enabled.

18.7.8 USART Guard Time And Prescaler Register (USART_GTP)

Address offset : 0x18

Reset value : 0x0000 0000



Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained
15:8	GTV[7:0]	<p>Guard time value in Smartcard mode.</p> <p>This bit field specifies the guard time in baud clock. In Smartcard mode, this function is required. The setting time of USART_STS.TXC flag is delayed by GTV[7:0] baud clock cycles.</p> <p><i>Note: This bit is invalid for UART5/6.</i></p>
7:0	PSCV[7:0]	<p>Prescaler value.</p> <p>In IrDA low power consumption mode: these bits are used to set the prescaler coefficient for dividing the peripheral clock (PCLK1/PCLK2) to generate low power consumption frequency.</p> <p>00000000: reserved-do not write this value. 00000001: divide the source clock by 1. ... 11111111: divide the source clock by 255.</p> <p>In IrDA normal mode: PSCV can only be set to 00000001.</p> <p>In Smartcard mode: PSCV[4:0] is used to set the prescaler of Smartcard clock generated by peripheral clock (PCLK1/ PCLK2). Coefficient. The actual prescaler coefficient of is twice the set value of PSCV[4:0]. 0000: reserved-do not write this value. 0001: Divide the source clock by 2. 0010: Divide the source clock by 4. ... 1111: Divide the source clock by 62.</p> <p>In Smartcard mode, PSCV[7:5] is reserved.</p> <p><i>Note: This bit is invalid for UART5/6.</i></p>

19 Low Power Universal Asynchronous Receiver Transmitter

19.1 Introduction

Low power universal asynchronous receiver transmitter (LPUART) is a low power, full duplex, asynchronous serial communication interface. The LPUART can be clocked by HSI, HSE, LSI, LSE, SYSCLK and PCLK1. If 32.768kHz LSE is selected as the clock source, the LPUART can operate in STOP low-power mode with a maximum communications up to 9600bps. The LPUART supports receiving data wake-up. By configuring wake-up events, the CPU in STOP2 mode can be woken up.

At the same time, when MCU operates in RUN mode, The LPUART can also be used as a common asynchronous serial interface. Users can switch the clock source to HSI, HSE, SYSCLK and PCLK1 to obtain higher communication speed.

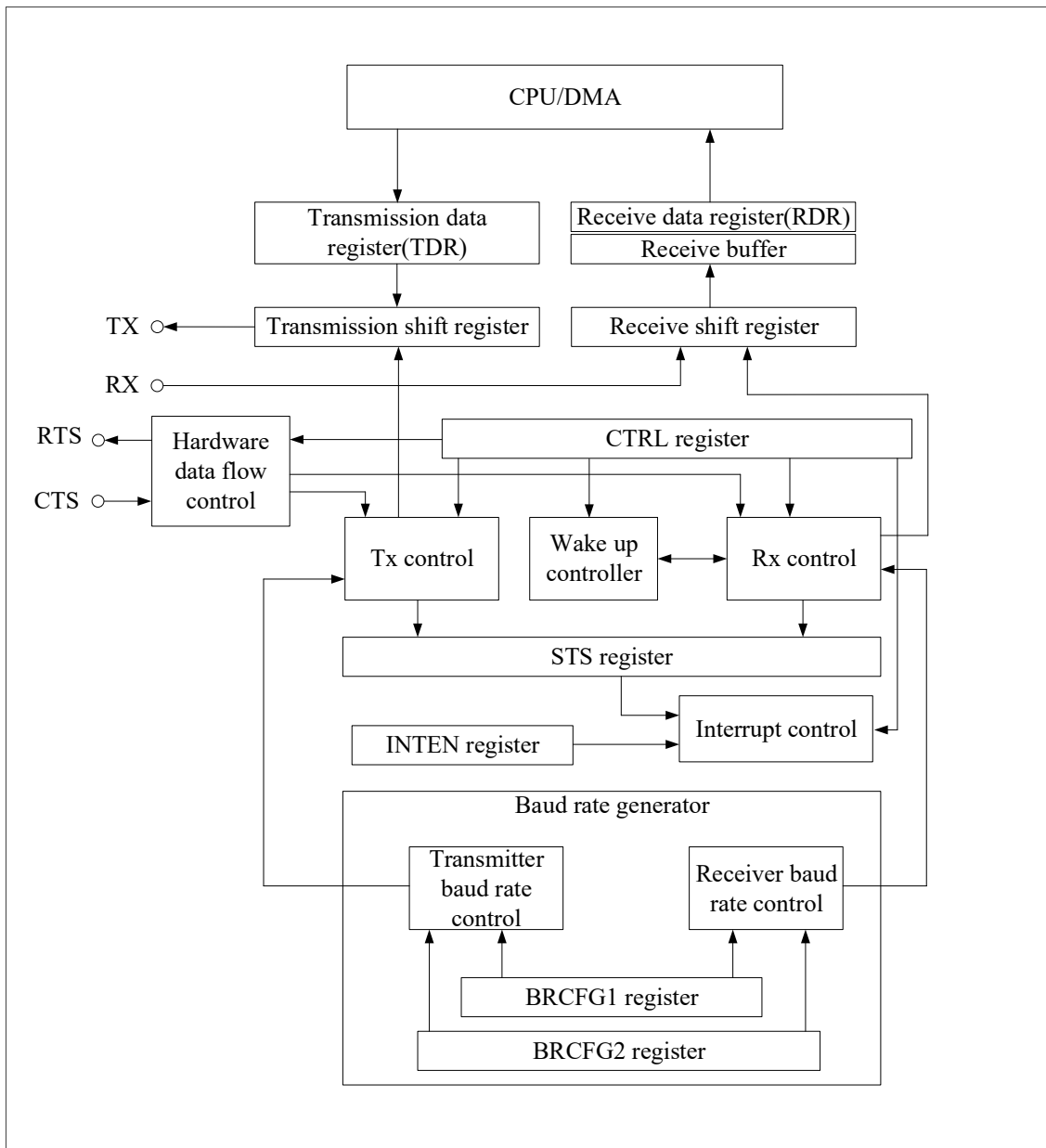
19.2 Main Features

- Full duplex, asynchronous communication;
- NRZ standard format;
- Fractional baud rate generator system, baud rate programmable, used for transmitting and receiving up to 3Mbits/s
- Programmable data word length (8 or 9 bits)
- Configurable stop bit, supporting 1 or 2 stop bits;
- LIN master's ability to send synchronous interrupters and LIN slave's ability to detect interrupters. When USART hardware is configured as LIN, it generates 13 bit interrupters and detects 10/11 bit interrupters
- Output clock for synchronous transmission;
- IrDA SIR encoder decoder, supports 3/16 bit duration in normal mode;
- Smartcard simulation function;
 - The smartcard interface supports the asynchronous smartcard protocol defined in ISO7816-3.
 - 0.5 and 1.5 stop bits for smartcards;
- Single-wire half duplex communication;
- Configurable multi-buffer communication using DMA, receiving/transmitting bytes in SRAM using centralized DMA buffer;
- Independent transmitter and receiver enable bits;
- Test flag
 - Receive buffer is full
 - Send buffer empty
 - End of transmission flag
- Parity control
 - Send parity bit

- Verify the received data
- Four error detection flags;
 - Overrun error
 - Noise error
 - Frame error
 - Parity error
- Ten interrupt sources with flags
 - CTS change
 - LIN disconnect detection
 - Transmit data register is empty
 - Send complete
 - Received data register is full
 - Bus was detected to be idle
 - Overrun error
 - Frame error
 - Noise error
 - Parity error
- Multi-processor communication, if the address does not match, then enter the silent mode;
- Wake up from silent mode (via idle bus detection or address flag detection)
- Two ways to wake up the receiver: address bit (MSB, bit 9), Bus idle

19.3 Functional Block Diagram

Figure 19-1 LPUART Block Diagram



19.4 Function Description

As shown in Figure 19-1, bidirectional LPUART communication requires at least two pins: receiving data input (RX) and transmitting data output (TX).

RX(Receive Data Input): When the number of samples is 3, data and noise can be distinguished.

TX(Transmit Data Output): When transmitting is enabled, the pin defaults to be high level.

The following pins are required in hardware flow control mode:

CTS (Clear To Send): When transmitter detects that CTS is valid (low level), the next data is transmitted.

RTS (Request To Send): When receiver is ready to receive new data, pull the RTS pin low.

LPUART has the following characteristics:

- The bus should be in an Idle state when not transmitting or receiving
- One start bit
- One data word (8 bits) with LSB first
- One stop bit, indicating the end of a data frame
- A status register (LPUART_STS)
- Data register (LPUART_DAT)
- Two baud rate configuration registers (LPUART_BRCFG1 and LPUART_BRCFG2) using fractional baud rate generators: 16-bit integer and 8-bit decimal representations

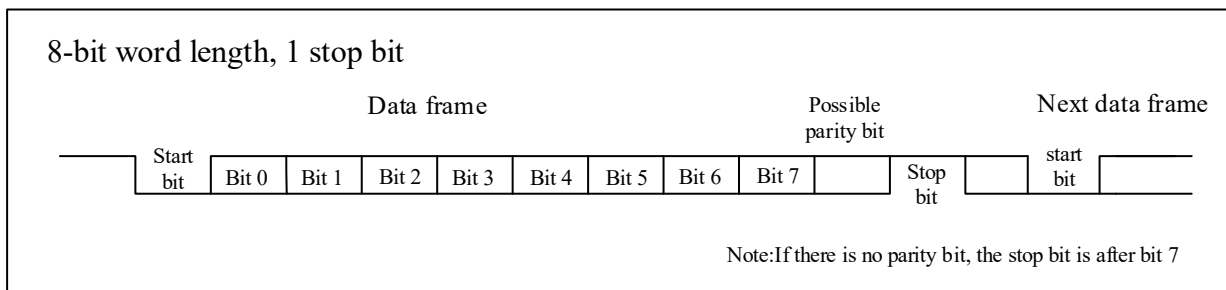
For specific definitions of each bit in the registers above, please refer to Section 19.6 of register Description.

19.4.1 LPUART Frame Format

The LPUART data length is fixed at 8 bits (refer to Figure 19-2). During the start bit, TX pin is at a low level and during the stop bit it is at a high level. The parity bit follows the data when enabled.

Both transmitting and receiving are driven by two different baud clock generators. When the LPUART_CTRL.TXEN of transmitter is set, the corresponding baud clock generator generates baud clock. When the start bit is received, the receiver's corresponding baud clock generator generates the clock.

Figure 19-2 Frame Format



Note: in this chapter, unless special instruction, setting means that a register is set to state '1', and resetting or clearing means that a register is set to state '0'. Hardware or programs may set or clear a register. Please refer to this chapter for details.

19.4.2 Transmitter

When the transmit Enable bit (LPUART_CTRL.TXEN) is set and there is data in the buffer, the transmitter sends 8-bit data. The data in the shift register is output on the TX pin.

Transmission process

During an LPUART transmission, the LSB of the data is shifted out on TX pin. In this mode, the LPUART_DAT register contains a buffer between the internal bus and the transmitter shift register (refer to Figure 19-1).

Each character is preceded by a low level starting bit; and is terminated by a stop bit.

Note: You cannot reset the LPUART_CTRL.TXEN bit during data transmission, otherwise the data on the TX pin will be corrupted because the baud rate counter stops counting. The current data being transmitted will be lost.

The steps for LPUART to transmission data as follows:

1. Configure baud rate, parity check, DMA, flow control, etc.
2. Set the LPUART_CTRL.TXEN bit to enable data transmission.
3. Write data to the LPUART_DAT register.
4. Check if the LPUART_STS.TXC flag is set, it means the transmission is over. If the flag is set, write 1 to the LPUART_STS.TXC bit to clear the flag.
5. Check the LPUART_STS.PEF bit to confirm whether the parity is wrong.
6. Otherwise, go to step 3 and transmit the next data.

Note: Be sure to initialize the LPUART module before using the transmitter.

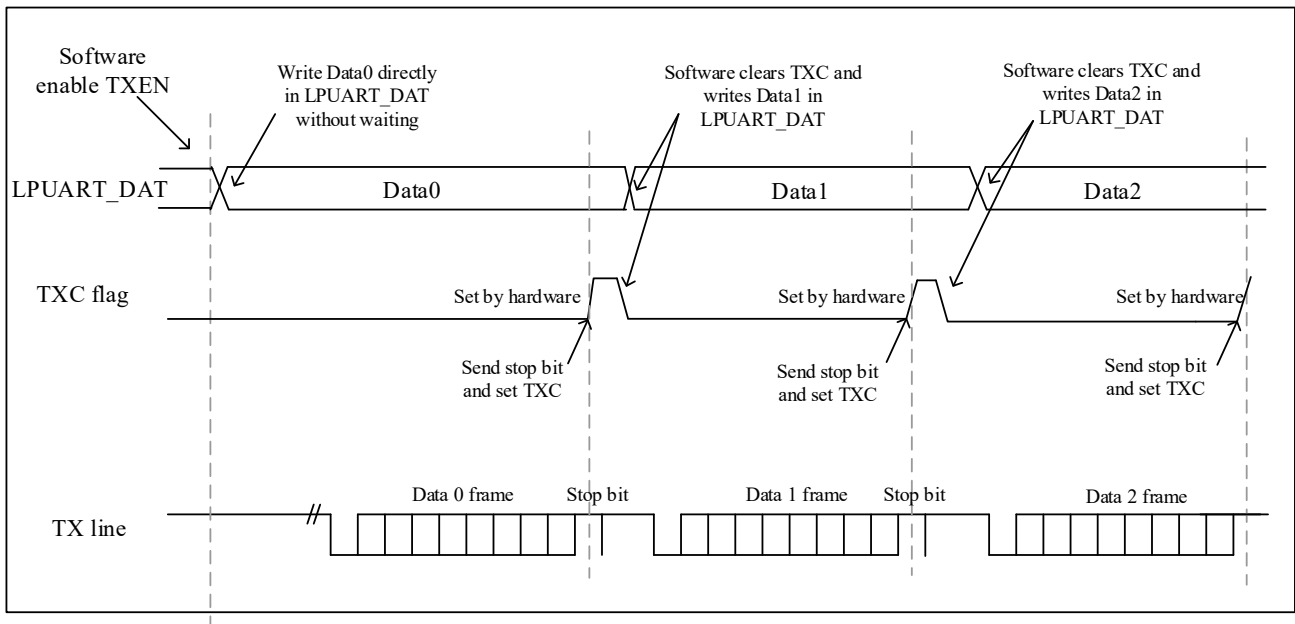
Initialization of LPUART proceeds as follows:

1. Set all flag bits in the LPUART_STS register to clear the interrupt flag.
2. To enable the interrupt function, configure LPUART_INTEN.
3. Set LPUART_CTRL.FLUSH, clear the RX buffer.

When transmit data:

- After configuring the baud rate and setting LPUART_CTRL.TXEN, the CPU can write directly to the LPUART_DAT register to transmit data.
- When a frame transmission is completed (after the stop bit is transmitted), the LPUART_STS.TXC bit is set. If the LPUART_INTEN.TXCIEN bit is set, an interrupt occurs immediately.
- After the last data byte is written to the LPUART_DAT register, you must wait for LPUART_STS.TXC=1 before shutting down the LPUART module or setting the microcontroller into low-power mode.

Figure 19-3 TXC Changes During Transmission



19.4.3 Receiver

Start bit detection

If the LPUART_CTRL.SMPCNT bit is 0, that is, the number of samples is 3, when there are at least two 0s in the three sample numbers, the start bit is valid. Otherwise it will be invalid.

Sampling Values	NF State	Received Bit Value	Start Bit Validity
000	0	0	effective
001	1	0	effective
010	1	0	effective
011	1	1	invalid
100	1	0	effective
101	1	1	invalid
110	1	1	invalid
111	0	1	invalid

Receive process

During LPUART reception, the LSB of data are first moved in from the RX pin. In this mode, the LPUART_DAT register contains a buffer between the internal APB bus and the receive shift register.

The steps for LPUART to receive data are as follows:

1. Configure baud rate, parity check, wake up event, sampling mode, DMA, flow control, etc.
2. Check the interrupt flags of the LPUART_STS register: buffer not empty, buffer half full, buffer full, buffer overrun;

3. Read the data by reading the LPUART_DAT register.
4. Return to step 2 and continue receiving data.

Note: Please be sure to initialize the LPUART module before using the receiver.

When receiving a data frame:

- The LPUART_STS.FIFO_NE bit is set, and the contents of the shift Register are transmitted to the RDR (Receiver Data Register). In other words, the data has been received and can be read (including its associated error flags).
- If the LPUART_INTEN.FIFO_NEIEN bit is set, an interrupt is generated.
- Frame errors (parity detection errors), noise or overrun errors are detected during reception, so the error flag will be set.
- In multi-buffer communication mode, the LPUART_STS.FIFO_NE flag bit is set after each byte received and cleared by DMA's read operation on the data register.
- In single buffer mode, the software can clear LPUART_STS.FIFO_NE bits by reading the LPUART_DAT register or by writing 0. The LPUART_STS.FIFO_NE bit must be cleared before the end of the next frame of data reception to avoid overrun errors.

Overrun error

The LPUART receiving data buffer has a total of 32 bytes. The LPUART_STS.FIFO_FU flag will be set after receiving 32 bytes of data. If the buffer data is not read out in time, causing the LPUART_STS.FIFO_FU not to be reset in time and an overrun error occurs when the next character is received. This character will be discarded by the hardware. Data can only be transmitted from the shift register to the receiving data buffer if the LPUART_STS.FIFO_FU bit is cleared. If the next data has been received or the previous DMA request has not been served, the LPUART_STS.FIFO_FU flag is still set and an overrun error occurs.

When an overrun error occurs:

- The LPUART_STS.FIFO_OV bit is set.
- The receiving data buffer content will not be lost. Reading the LPUART_DAT register still returns the previous data.
- The contents of the shift register will be overwritten. Any subsequent data received will be lost.
- If the LPUART_INTEN.FIFO_OVIE bit is set, an interrupt is generated.
- LPUART_STS.FIFO_OV can be reset by reading the LPUART_DAT register.

Noise error

Noise errors use an over-sampling technique (if the LPUART_CTRL.SMPCNT bit is 0, that is, the number of samples is 3) to recover data by distinguishing valid input data from noise.

Figure 19-4 Data Sampling for Noise Detection

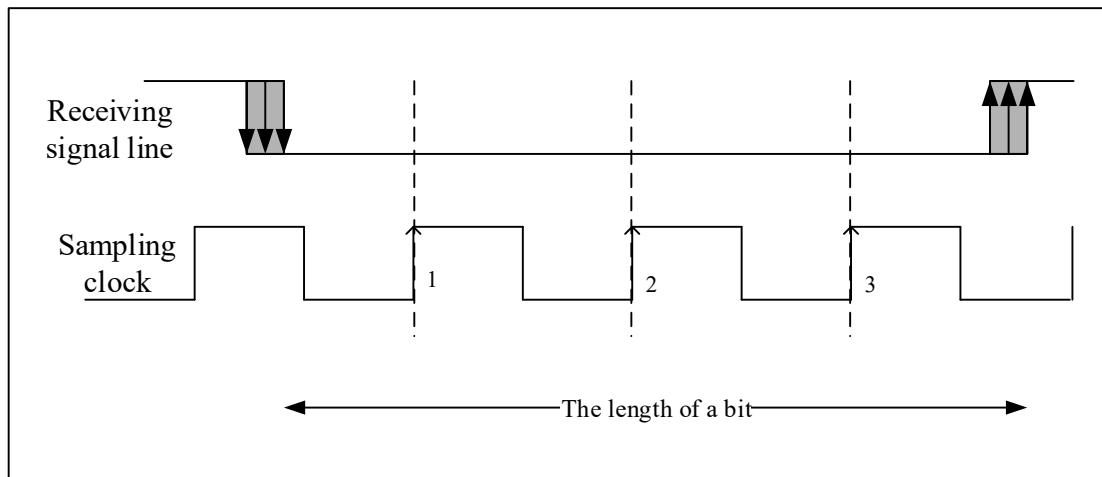


Table 19-1 Data Sampling for Noise Detection

Sampling Values	NF State	Received Bit Value
000	0	0
001	1	0
010	1	0
011	1	1
100	1	0
101	1	1
110	1	1
111	0	1

When noise is detected in a receiving frame, you can do the following:

- If three sample values are inconsistent, set the LPUART_STS.NF flag immediately.
- The received data is transmitted from the shift register to the buffer.
- Software write 1 clears the LPUART_STS.NF flag bit.

19.4.4 Fractional Baud Rate Generation

Baud rate prescaler coefficient is divided into 16-bit integer part and 8-bit decimal part. The baud rate generator uses the value of the combination of these two parts to determine the baud rate. The LPUART can generate standard baud rates due to having a fractional baud rate divider.

Baud rate prescaler coefficient (LPUARTDIV) has the following relationship with system clock (PCLK) :

$$\text{TX/RX baud rate} = f_{CLK}/(\text{LPUARTDIV})$$

Here the f_{CLK} is the clock for LPUART (the clock source of LPUART can be HSI, HSE, LSI, LSE, SYSClk, or PCLK1). The value of LPUARTDIV is set in the baud rate configuration registers LPUART_BRCFG1 and LPUART_BRCFG2

Note: After writing LPUART_BRCFG1 and LPUART_BRCFG2, the baud rate counter is replaced with the new value of the

baud rate register. Therefore, do not change the value of the baud rate register during communication.

Configure baud rates through LPUART_BRCFG1 and LPUART_BRRCFG2

Example, baud rate = 4800bps, clock frequency = 32768Hz.

$LPUARTDIV = 32768/4800 = 6.82667$. LPUART_BRCFG1 = 6 and the value of LPUART_BRCFG2 is calculated by adding fractions in the table below (the value of LPUART_BRCFG2 is 0xEFh).

Decimal Addition	Carry to The Next Integer	Bit Field	Value
$0.82667 + 0.82667 = 1.65333$	YES	DECIMAL0	1
$1.65333 + 0.82667 = 2.48000$	YES	DECIMAL1	1
$2.48000 + 0.82667 = 3.30667$	YES	DECIMAL2	1
$3.30667 + 0.82667 = 4.13333$	YES	DECIMAL3	1
$4.13333 + 0.82667 = 4.96000$	NO	DECIMAL4	0
$4.96000 + 0.82667 = 5.78667$	YES	DECIMAL5	1
$5.78667 + 0.82667 = 6.61333$	YES	DECIMAL6	1
$6.61333 + 0.82667 = 7.44000$	YES	DECIMAL7	1

When LSE clock (32.768KHz) is used, the values of baud rate configuration registers LPUART_BRCFG1 and LPUART_BRCFG2 with different baud rate. Settings are as follows:

Baud Rate	Divisor	LPUART_BRCFG1	LPUART_BRCFG2
300	109.2267	6Dh	88h
600	54.6133	36h	ADh
1200	27.3067	1Bh	24h
2400	13.6533	0Dh	6Dh
4800	6.8267	06h	EFh
9600	3.4133	03h	4Ah

Note: The lower the clock frequency of the CPU, the lower the accuracy of a particular baud rate.

19.4.5 Parity Control

Reset the LPUART_CTRL.PCDIS bit, enable parity control (generate a parity bit when transmitting, parity check when receiving), set or reset the LPUART_CTRL.PSEL bit to select odd or even patity. LPUART frame formats are listed in the table below.

Table 19-2 Parity Frame Format

PCDIS Bit	LPUART Frame
0	Start bit 8-bit data parity bit stop bit
1	Start bit 8 bits data stop bit

Transmission mode: Parity is enabled by resetting the LPUART_CTRL.PCDIS bit. If parity fails, the LPUART_STS.PEF flag is set to '1', and an interrupt occurs if LPUART_INTEN.PEIE is set.

Odd parity: LPUART_CTRL.PSEL = 1.

Make the number of '1' in one frame data (including parity bit) be an odd number. For example: if Data=11000101, there are 4 '1's, then the parity bit will be '1' (5 '1' in total).

Even parity: LPUART_CTRL.PSEL = 0.

Make the number of '1' in one frame data (including parity bit) be an even number. For example: if Data=11000101, there are 4 '1's, then the parity bit will be '0' (4 '1' in total).

19.4.6 DMA Application

The LPUART can access the transmit data register (TDR) and receive buffer respectively through DMA.

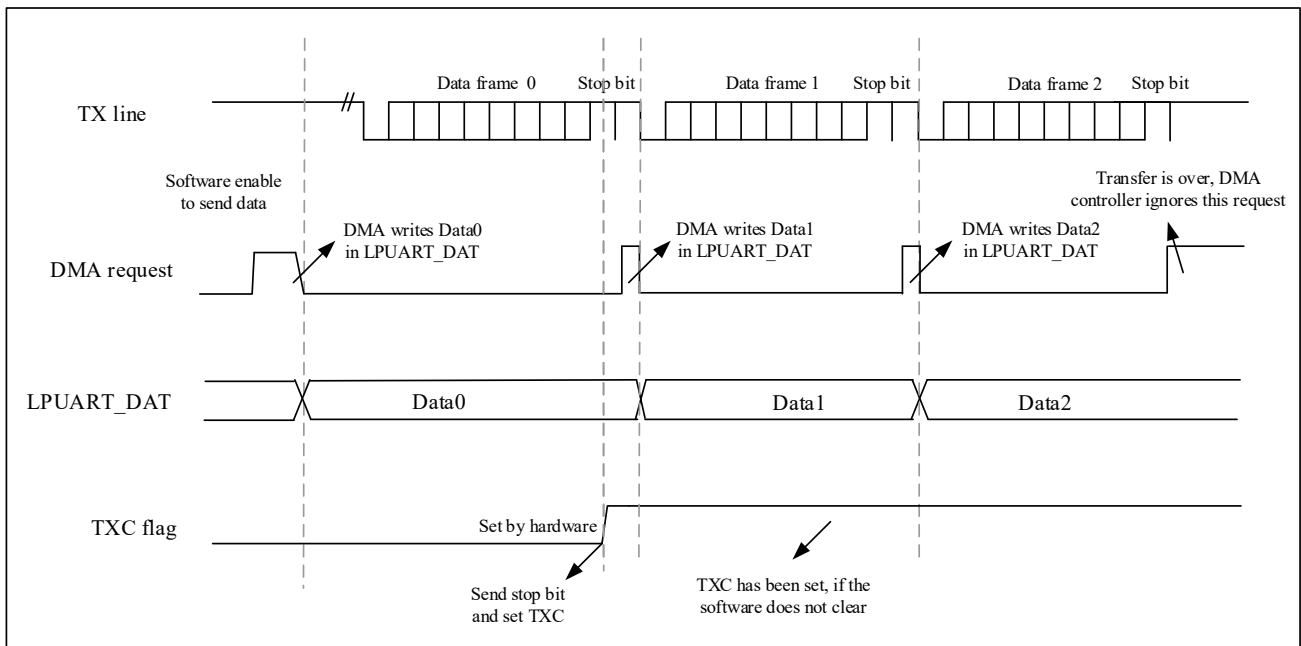
DMA transmission

The steps for assigning a DMA channel to the LPUART transmissions are as follows (x indicates the channel number) :

1. Configure the LPUART_DAT register address as the destination address for DMA transmission, and the memory address as the source address for DMA transmission.
2. Set the total number of bytes to be transmitted.
3. Set the channel priority.
4. Configure to generate DMA interrupts when the transmission is half or all complete.
5. Activate the channel.

Completing a DMA transmission will generate an interrupt on the corresponding DMA channel. In transmission mode, when the DMA has finished the data transmission, the DMA controller sets the DMA_INTSTS.TXCFx flag. The LPUART_STS.TXC flag bit is asserted by the hardware to indicate that the transmission is completed. The software needs wait for LPUART_STS.TXC=1.

Figure 19-5 transmission Using DMA



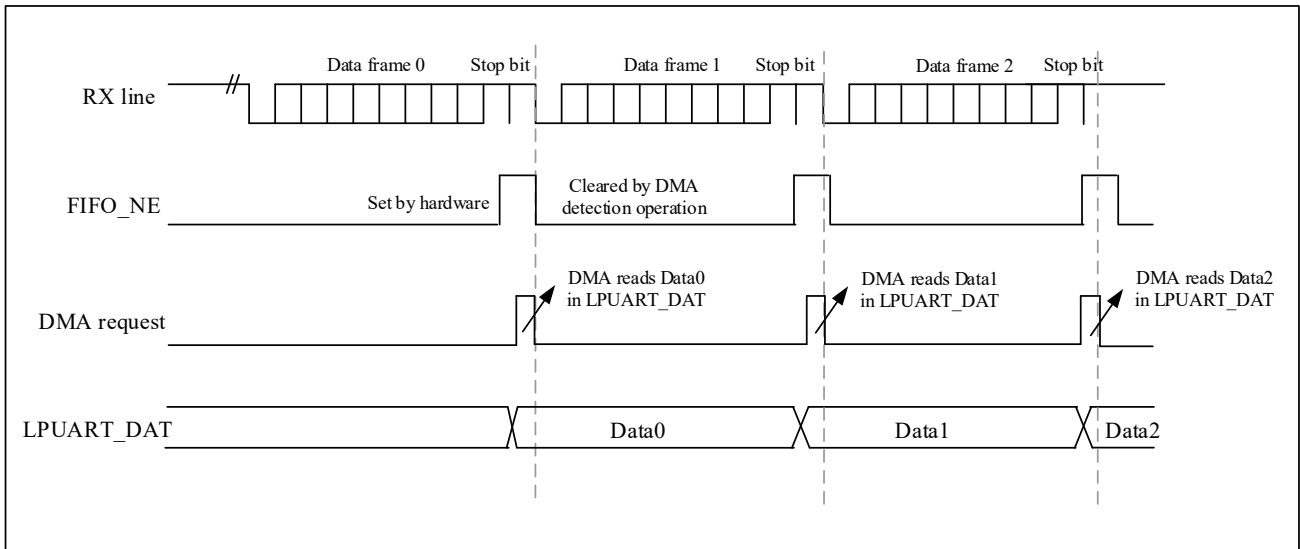
DMA reception

The steps for assigning a DMA channel to the LPUART receiving are as follows (x indicates the channel number) :

1. Configure the LPUART_DAT register address as the source address for transmission and the memory address as the destination address for transmission through the DMA configuration register.
2. Configure the number of DMA bytes to be transmitted.
3. Configure the channel priority on the DMA register for data transmission.
4. Configure interrupts to generate DMA interrupts when the transmission is half or all complete.
5. Activate the channel.

When the DAM controller completes the specified transmission amount for reception, the DMA controller generates an interrupt on the DMA channel's interrupt vector.

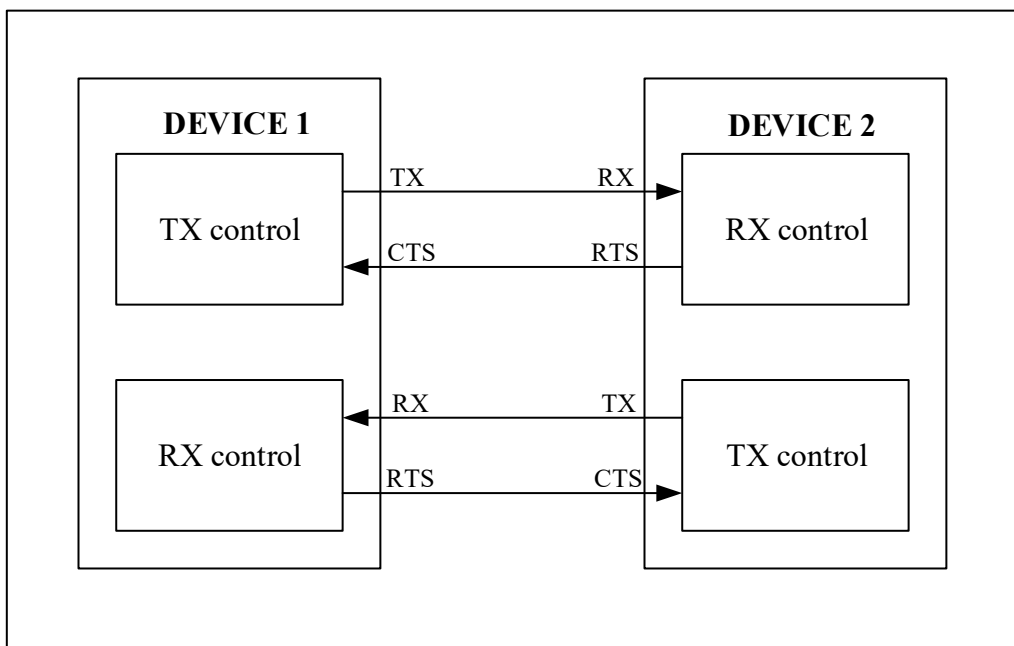
Figure 19-6 Receiving with DMA



19.4.7 Hardware Flow Control

Hardware flow control functions through CTS input and RTS output. The following figure shows how two devices are connected in this mode.

Figure 19-7 Hardware Flow Control Between Two LPUART



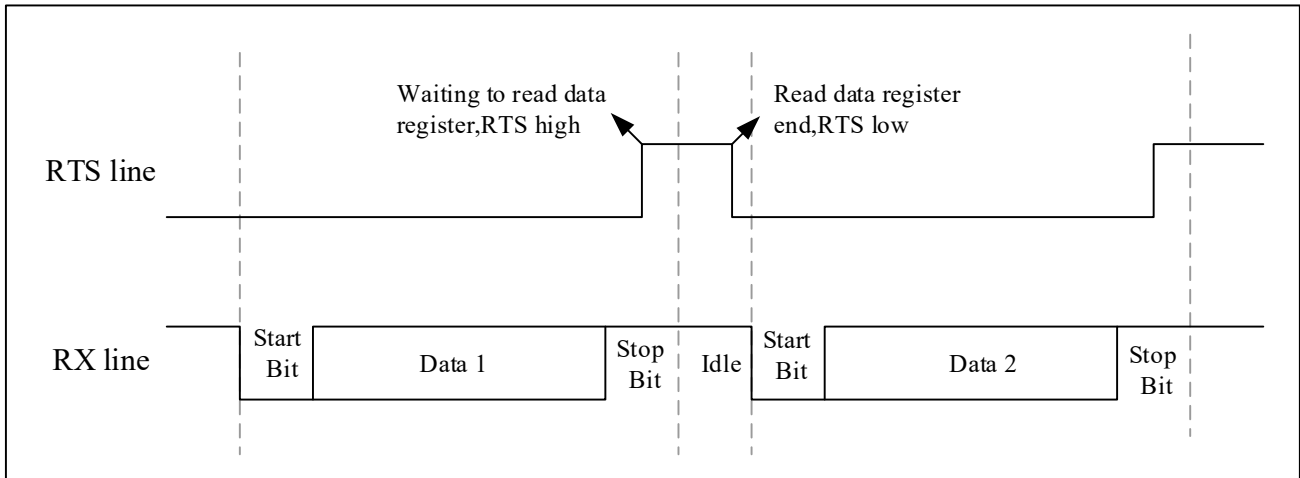
RTS and CTS flow control can be independently enabled by setting LPUART_CTRL.RTSEN and LPUART_CTRL.CTSEN.

RTS flow control

If RTS flow control is enabled (LPUART_CTRL.RTSEN=1), the RTS will be driven high (active) when the RTS threshold condition is achieved, otherwise it will be driven low. The timing of RTS effectiveness can be selected by the LPUART_CTRL.RTS_THSEL[1:0] bits. The RTS threshold can be selected to be effective when the FIFO is half full, 3/4

full, or full. Below is an example of communication with RTS flow control enabled.

Figure 19-8 RTS Flow Control

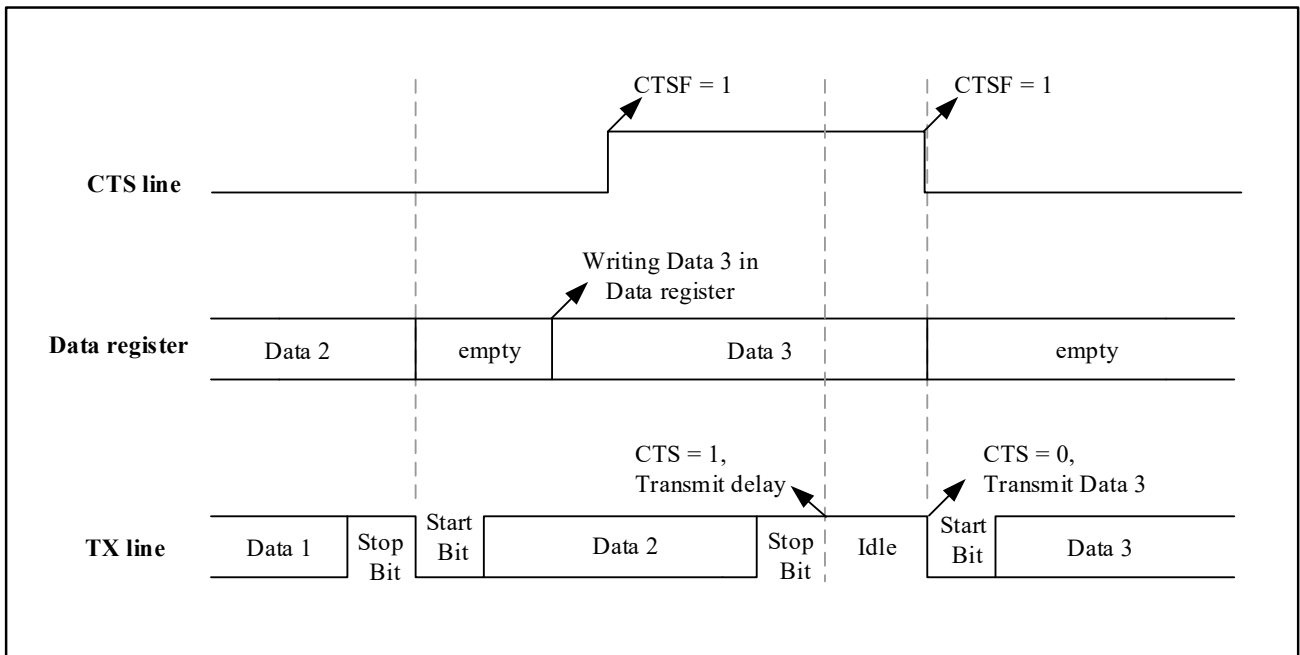


CTS flow control

If CTS flow control is enabled (LPUART_CTRL.CTSEN=1), the transmitter will check the CTS pin to decide whether to transmit data before transmitting the next frame. If the CTS is pulled low (effective), the transmitter transmits data (assuming that data is ready to be transmitted). If the CTS is pulled up during transmission, the transmission of the current data frame is stopped after transmission.

If CTS flow control is enabled (LPUART_CTRL.CTSEN=1), the signal of CTS pin will be changed. Refer to Figure 19-9 for enabling CTS flow control.

Figure 19-9 CTS Flow Control



19.4.8 Low Power Wakeup

The LPUART can operate in STOP mode, if the LPUART_CTRL.WUSTP is set, it can wake up the system on EXTI line 22 when a specific waking up event occurs.

The LPUART waking up event can be handled in the following ways (through the LPUART_CTRL.WUSEL[1:0]) :

- A waking up event is generated when a start bit is detected
- A waking up event is generated when the receive buffer non-empty flag is set
- A waking up event is generated when data is received and the first byte matches LPUART_WUDAT[7:0]
- A waking up event is generated when data is received and four bytes match LPUART_WUDAT[31:0]

When waking up event occurs, the LPUART_STS.WUF bit will be set.

19.5 Interrupt Request

Table 19-3 LPUART Interrupt Requests

Interrupt Function	Interrupt Event	Event Flag	Enable Bit
LPUART global interrupt	Parity check error	PEF	PEIE
	TX complete	TXC	TXCIE
	Receive buffer overrun	FIFO_OV	FIFO_OVIE
	Receive buffer full	FIFO_FU	FIFO_FUIE
	Receive buffer half full	FIFO_HF	FIFO_HFIE
	Receive buffer not empty	FIFO_NE	FIFO_NEIE
	Wake up in STOP mode	WUF	WUFIE

LPUART interrupt events are in a logical OR relationship. If the corresponding enable control bit is set, these events can generate their own interrupt, but only one interrupt request can be generated at the same time.

19.6 LPUART Registers

19.6.1 LPUART Register Overview

Table 19-4 LPUART Register Overview

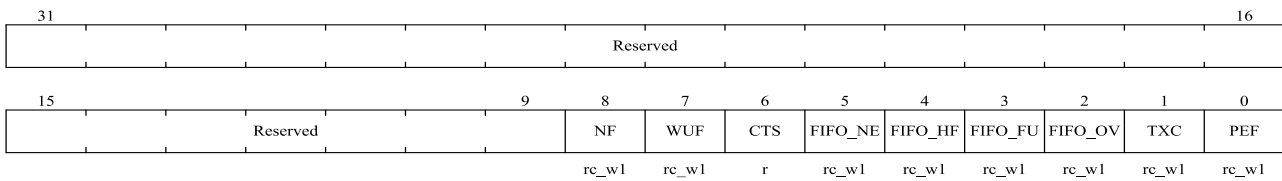
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																					
000h	LPUART_STS	Reserved																							NF	WUF	CTS	FIFO_NE	FIFO_HF	FIFO_FU	FIFO_OV	TXC	PEF																					
	Reset Value																								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
004h	LPUART_INTEN	Reserved																													WUFIE	FIFO_NEIE	FIFO_HFIE	FIFO_FUIE	FIFO_OVIE	TXCIE	PEIE																	
	Reset Value																														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
008h	LPUART_CTRL	Reserved																	SMPCNT	WUSEL [1:0]	RTSEN	CTSEN	RTS_THS EL [1:0]	WUJSTP	DMA_RXEN	DMA_TXEN	LOOPBACK	PCDIS	FLUSH	TXEN	PSEL																							
	Reset Value																		0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
00Ch	LPUART_BRCFG1	Reserved																	INTEGER[15:0]																																			
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
010h	LPUART_DAT	Reserved																								DAT[7:0]							
	Reset Value	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																															
014h	LPUART_BRCFG2	Reserved																								DECIMAL[7:0]							
	Reset Value	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																															
018h	LPUART_WUDAT	WUDAT[31:0]																															
	Reset Value	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																															

19.6.2 LPUART Status Register (LPUART_STS)

Address offset: 0x00

Reset value: 0x0000 0000



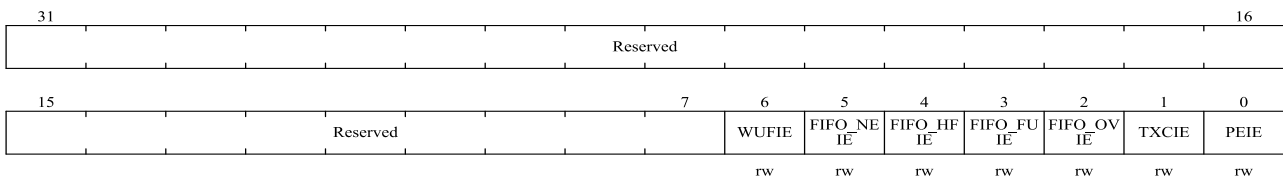
Bit Field	Name	Description
31:9	Reserved	Reserved, the reset value must be maintained.
8	NF	Noise detected flag. When noise is detected in the received frame, this bit is set by hardware. This bit is cleared by the software. 0: No noise is detected. 1: Noise is detected.
7	WUF	Wakeup from STOP mode Flag. 0: No wake up event is detected. 1: A wake up event is detected.
6	CTS	CTS signal (hardware flow control) flag. Once the sender requests to transmit data, it is ready to receive it. 0: CTS line is reset. 1: CTS line is set.
5	FIFO_NE	FIFO non-empty flag. 0: Buffer is empty. 1: Buffer is not empty. RX data is ready to be read
4	FIFO_HF	FIFO half full flag. 0: Buffer is not half full. 1: Buffer is half full. RX data should be read before the buffer is full
3	FIFO_FU	FIFO full flag. 0: Buffer is not full. 1: Buffer is full. RX data should be read out in preparation for receiving new data
2	FIFO_OV	FIFO overrun flag. 0: Buffer did not overrun 1: Buffer overrun.

Bit Field	Name	Description
1	TXC	TX complete flag. 0: TX is disabled or not complete. 1: TX transmission is complete.
0	PEF	Parity check error flag. 0: No parity error detected. 1: Parity error detected

19.6.3 LPUART Interrupt Enable Register (LPUART_INTEN)

Address offset: 0x04

Reset value: 0x0000 0000



Bit Field	Name	Description
31:7	Reserved	Reserved, the reset value must be maintained.
6	WUFIE	Wake up interrupt enable 0: Disable wake up interrupt 1: Enable wake up interrupt
5	FIFO_NEIE	Receive buffer not empty interrupt enable 0: Disable buffer non-empty interrupt 1: Enable buffer non-empty interrupt
4	FOFO_HFIE	Receive buffer half-full interrupt enable 0: Disables buffer half-full interrupt 1: Enables buffer half-full interrupt
3	FOFO_FUIE	Receive buffer full interrupt enable 0: Disables buffer full interrupt 1: Enable buffer full interrupt
2	FIFO_OVIE	Receive buffer overrun interrupt enable 0: Disables buffer overrun interrupt 1: Enable buffer overrun interrupt
1	TXCIE	TX complete interrupt enable 0: Disable TX complete interrupt 1: Enable TX complete interrupt
0	PEIE	Parity check error interrupt enable 0: Disable parity error interrupt 1: Enable parity error interrupt

19.6.4 LPUART Control Register (LPUART_CTRL)

Address offset: 0x08

Reset value: 0x0000 0200

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SMPCNT	WUSEL[1:0]	RTSEN	CTSEN	RTS_THSEL[1:0]	WUSTP	DMA_RXEN	DMA_TXEN	LOOPBACK	PCDIS	FLUSH	TXEN	PSEL		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		

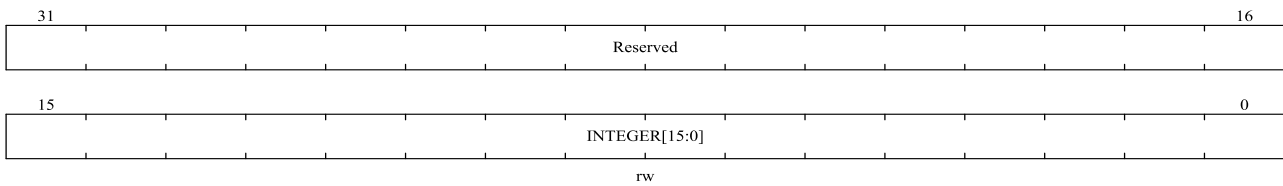
Bit Field	Name	Description
31:15	Reserved	Reserved, the reset value must be maintained.
14	SMPCNT	Specify sampling method 0: 3 sample bits, noise detection is allowed (LPUARTDIV should be large enough, such as greater than 10) 1: 1 sample bits, closed noise detection
13:12	WUSEL[1:0]	Wake up event selection. 00: Start bit detection 01: Non-empty detection of receive buffer 10: A configurable receive byte 11: A programmable 4-byte frame
11	RTSEN	RTS hardware flow control enable 0: Disables RTS hardware flow control 1: Enables RTS hardware flow control
10	CTSEN	CTS hardware flow control enable 0: Disables CTS hardware flow control 1: Enables CTS hardware flow control
9:8	RTS_THSEL[1:0]	RTS threshold selection 00: When FIFO is half full, RTS is effective (pull up) x1: When FIFO is 3/4 full, RTS effective (pull up) 10: When FIFO is full, RTS effective (pull up)
7	WUSTP	LPUART STOP mode wakeup enabled 0: Cannot wake up STOP mode 1: Can wake up the STOP mode
6	DMA_RXEN	DMA RX request enable
5	DMA_TXEN	DMA TX request enable
4	LOOKBACK	Loopback self-test 0: Normal mode 1: Loopback self-test mode
3	PCDIS	Parity control 0: Enables parity bit 1: Disables parity bit
2	FLUSH	Clear receive buffer 0: Disables buffer clear 1: Clear buffer content
1	TXEN	TX enable

Bit Field	Name	Description
		0: Disables TX 1: Enables TX
0	PSEL	Odd parity enable 0: Even parity 1: Odd parity

19.6.5 LPUART Baud Rate Configuration Register 1 (LPUART_BRCFG1)

Address offset: 0x0C

Reset value: 0x0000 0174

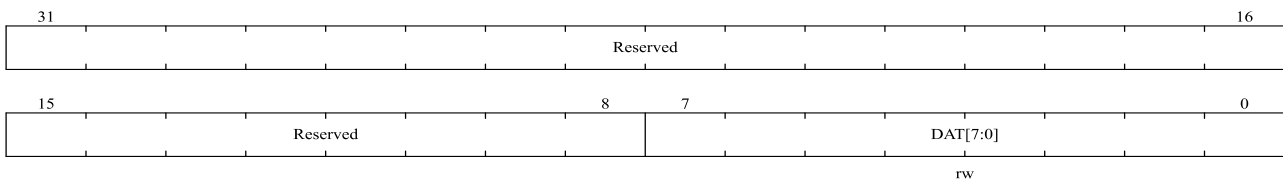


Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained.
15:0	INTEGER[15:0]	Baud rate configuration register 1. The calculation of baud rate configuration register 1 is as follows: If the baud rate is 9600bps and the clock frequency is 32768Hz. $LPUARTDIV = 32768/9600 = 3.4133$ In this case, the integer part of the LPUARTDIV is 3 and the decimal part is 0.4133. LPUART_BRCFG1 = 3. LPUART_BRCFG2 will be used for baud rate error correction. For the 3-bit sampling method with noise detection characteristics, LPUARTDIV is not large enough at this time, so 1-bit sampling method should be adopted to avoid sampling error.

19.6.6 LPUART Data Register (LPUART_DAT)

Address offset: 0x10

Reset value: 0x0000 0000

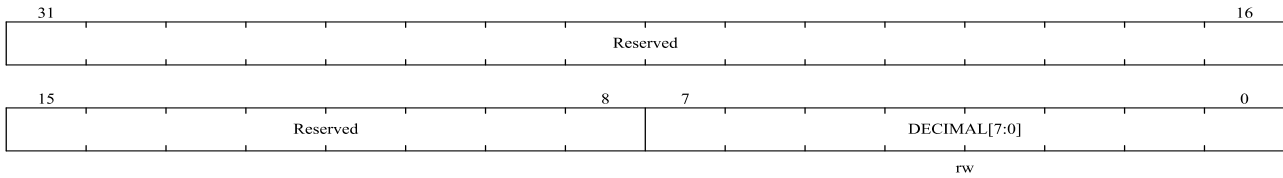


Bit Field	Name	Description
31:8	Reserved	Reserved, the reset value must be maintained.
7:0	DAT[7:0]	Write to the data register when sending Read the data register when receiving

19.6.7 LPUART Baud Rate Configuration Register 2 (LPUART_BRCFG2)

Address offset: 0x14

Reset value: 0x0000 0000

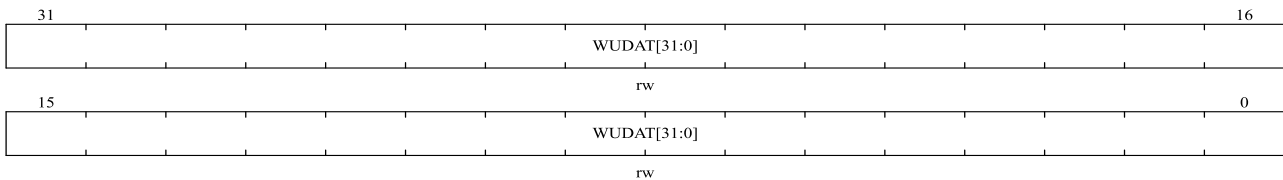


Bit Field	Name	Description
31:8	Reserved	Reserved, the reset value must be maintained.
7:0	DECIMAL[7:0]	Baud rate configuration register 2 is used for baud rate error correction at low frequencies. For example, If the baud rate is 4800bps and the clock frequency is 32768Hz. $LPUARTDIV = 32768/4800 = 6.8266$ $LPUART_BRCFG1 = 6$. In this case, to correct the baud rate error, you should configure register 2 with baud rate. For details on how to configure register 2, refer to the section "Fractional Baud Rate Generation".

19.6.8 LPUART Wake Up Data Register (LPUART_WUDAT)

Address offset: 0x18

Reset value: 0x0000 0000



Bit Field	Name	Description
31:0	WUDAT[31:0]	When LPUART_CTRL.WUSEL[1:0] = 1x, WUDAT[31:0] is used to check whether the conditions for wake up from STOP mode is matched (byte match or frame match): LPUART_CTRL.WUSEL[1:0] = 10 is used to wake up byte matching. In this case, the first byte is valid LPUART_CTRL.WUSEL[1:0] = 11 is used to wake up frame matching. In this case, all 4 bytes are valid

20 Serial Peripheral Interface/Inter-IC Sound (SPI/ I²S)

20.1 SPI Introduction

This module is about SPI/I²S. It operates in SPI mode by default, and users can choose to operate in I²S mode by setting the registers.

Serial peripheral interface (SPI) can operate in master or slave mode, supports full-duplex and half-duplex high-speed communication modes, and have hardware CRC calculation and configurable multi-master mode.

Inter-IC sound (I²S) can operate in master or slave mode in half-duplex communication, and supports four audio standards: Philips I²S standard, MSB alignment standard, LSB alignment standard and PCM standard.

The above two interfaces are synchronous serial communication interfaces

20.2 SPI and I²S Main Features

20.2.1 SPI Features

- Full-duplex synchronous transmission;
- Two-wire simplex synchronous transmission with or without a third bidirectional data wire;
- 8-bit or 16-bit transmission frame format selection;
- Master or slave operations;
- Support multi-master mode;
- Fast communication between master mode and slave mode;
- NSS can be managed by software or hardware in both master and slave modes: dynamic change of master/slave modes;
- Programmable clock polarity and phase;
- Programmable data order, MSB before or LSB before;
- Dedicated send and receive flags that trigger interrupts;
- SPI bus busy flag;
- Hardware CRC for reliable communication;
 - In send mode, the CRC value can be sent as the last byte;
 - In full-duplex mode, CRC is automatically performed on the last byte received.
- Master mode failures, overloads, and CRC error flags that trigger interrupts
- Single-byte send and receive buffer with DMA capability: generates send and receive requests
- Maximum interface speed: 12Mbps

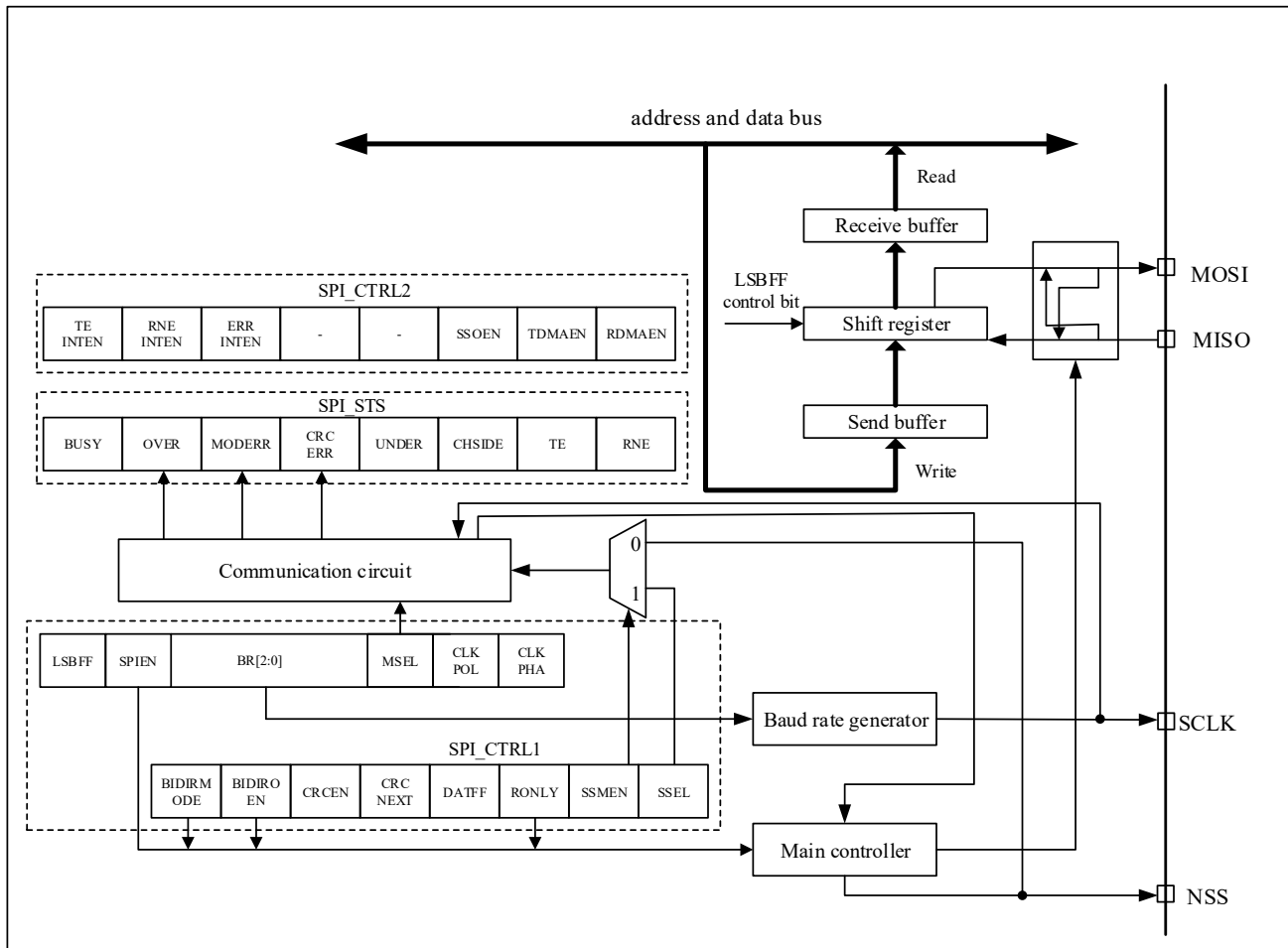
20.2.2 I²S Features

- Half-duplex communication (transmit or receive only);
- Master or slave operations;
- 8-bit linear programmable predivider for accurate audio sampling frequencies (8 KHZ to 96KHZ);
- The data format can be 16, 24, or 32 bits;
- Audio channel fixed packet frame is 16 bit (16 bit data frame) or 32 bit (16, 24 or 32 bit data frame);
- Programmable clock polarity (steady state);
- The overflows flag bit in slave transmitting mode and the overflows flag bit in master/slave receiving mode;
- 16-bit data registers are used for transmitting and receiving, with one register at each end of the channel;
- Supported I²S protocols:
 - I²S Philips standard;
 - MSB alignment standard (left aligned);
 - LSB alignment standard (right aligned);
 - PCM standard (16-bit channel frame with long or short frame synchronization or 16-bit data frame extension to 32-bit channel frame);
- The data direction is always MSB first;
- Both transmit and receive have DMA capability;

20.3 SPI Function Description

20.3.1 General Description

Figure 20-1 SPI Block Diagram



To connected external devices, SPI has four pins, which are as follows:

- **SCLK:** serial clock pin. Serial clock signal is output from the SCLK pin of master device and input to SCLK pin of slave device.
- **MISO:** master input/slave output pin. Data is received from the MISO pin of master device and transmitted by the MISO pin of slave device.
- **MOSI:** master output/slave input pin. Data is transmitted by the MOSI pin of master device and received from the MOSI pin of slave device.
- **NSS:** chip select pin. There are two types of NSS pin, internal pin and external pin. If the internal pin detects a high level, SPI operates in the master mode. Conversely, SPI operates in the slave mode. Users can use a standard I/O pin of the master device to control the NSS pin of the slave device.

Software NSS mode

The software slave device management is enabled when `SPI_CTRL1.SSMEN = 1`

The NSS pin is not used in software NSS mode. In this mode the internal NSS signal level is driven by writing the SPI_CTRL1.SSEL bit (master mode SPI_CTRL1.SSEL = 1, slave mode SPI_CTRL1.SSEL = 0).

Hardware NSS mode

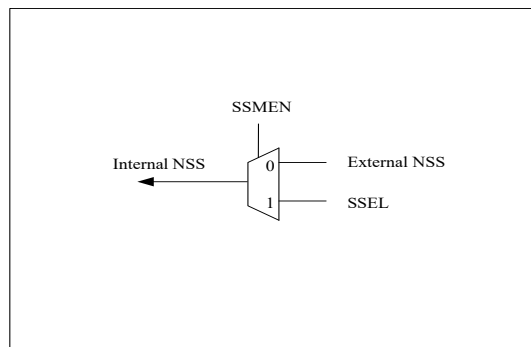
The software slave device management is disabled when SPI_CTRL1.SSMEN = 0.

NSS input mode: The NSS output of the master device is disabled (SPI_CTRL1.MSEL = 1, SPI_CTRL2.SSOEN = 0), allowing operation in multi-master mode. The master should connect NSS pin to the high level and the slave should connect NSS pin to the low level during the entire data frame transmission.

NSS output mode: NSS output of the master device is enable (SPI_CTRL1.MSEL = 1, SPI_CTRL2.SSOEN = 1). SPI as the master device must pull the NSS pin to low level, all device which connected to the master device and set to NSS hardware mode, will detect low level and enter the slave mode automatically. If the master device cannot pull the NSS pin to low level, device will enter the slave mode and generates the master mode failure error.

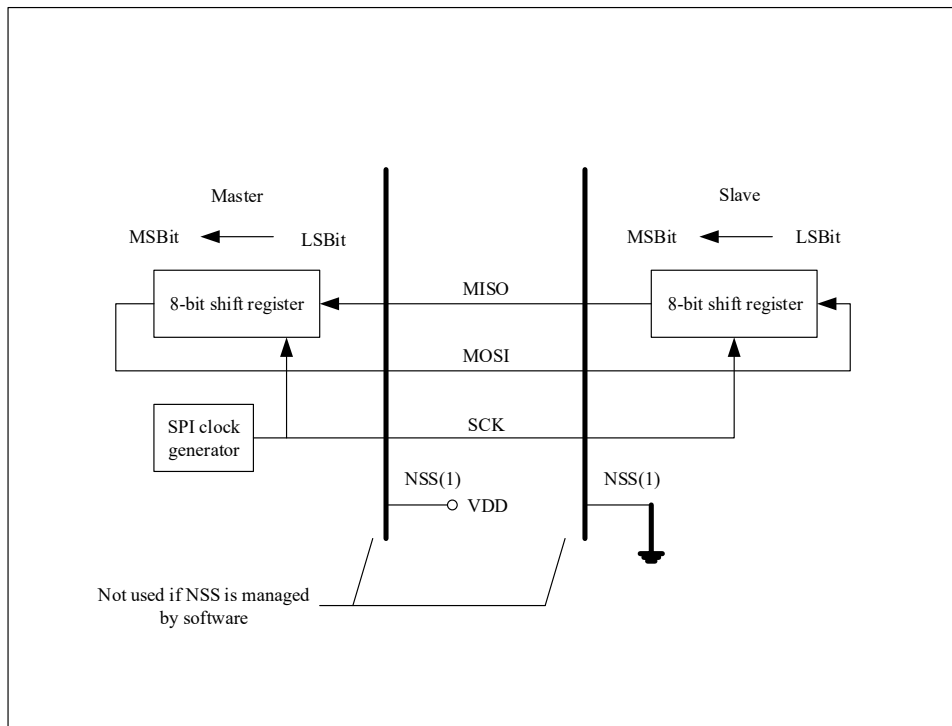
Note: The choice of software mode or hardware mode depends on whether NSS control is needed in the communication protocol. If not, you can choose the software mode, and release a GPIO pin for other purposes.

Figure 20-2 Selective Management of Hardware/Software



The following figure is an example of the interconnection of single master and single slave devices.

Figure 20-3 Master and Slave Applications



Note: NSS pin is set as input

SPI is a ring bus structure. The master device outputs a synchronous clock signal through the SCK pin, the MOSI pin of the master device is connected to the MOSI pin of the slave device, and the MISO pin of the master device is connected to the MISO pin of the slave device, so that data can be transmitted between devices. Continuous data transmission between master and slave, transmitting data to slave through MOSI pin and slave transmitting data to master through MISO pin.

SPI timing mode

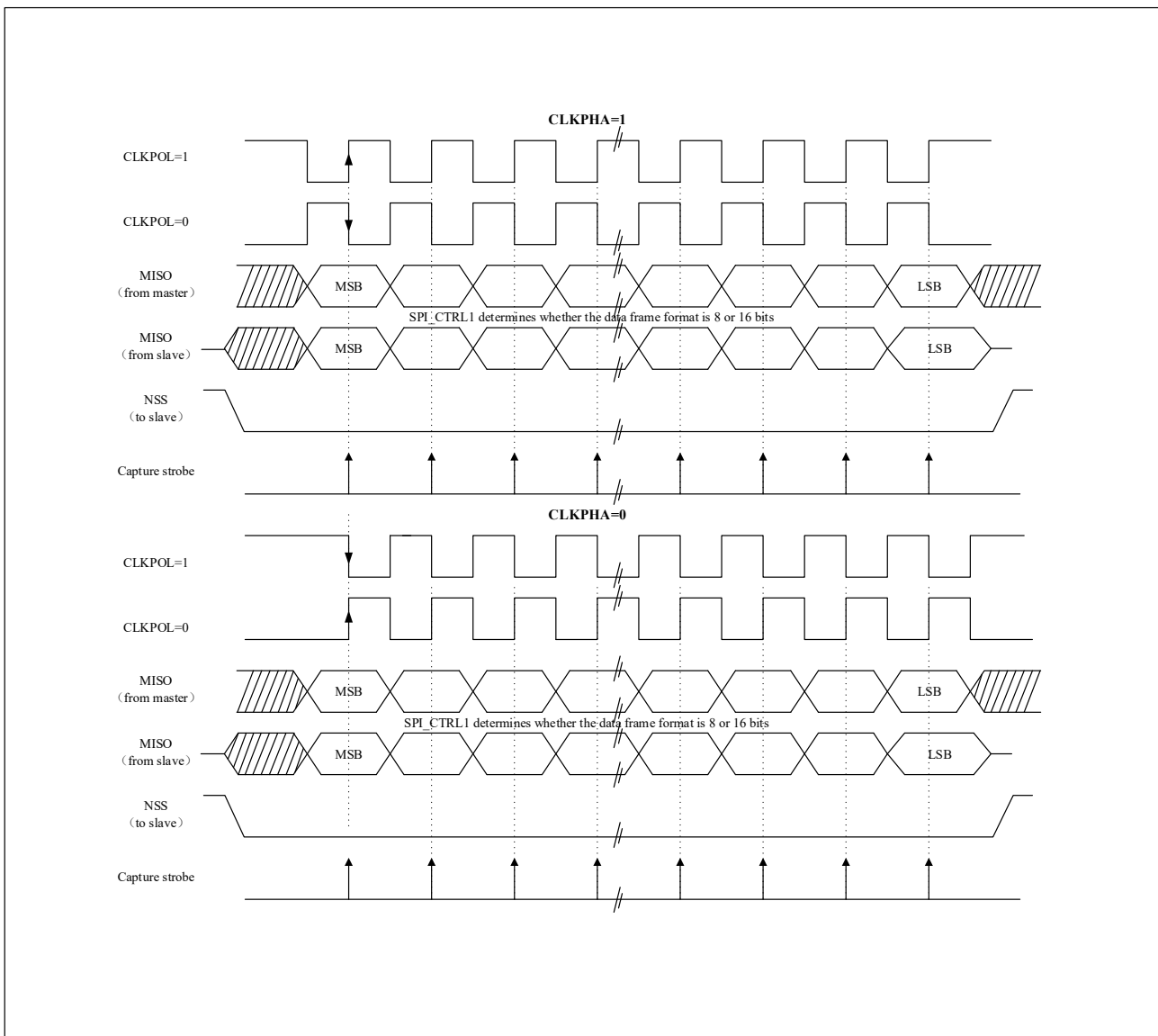
User can select the clock edge of data capture by setting SPI_CTRL1.CLKPOL bit and SPI_CTRL1.CLKPHA bit.

- When CLKPOL = 0, CLKPHA = 0, the SCLK pin will keep low in idle state, and the data will be sampled at the first edge, which is rising edge.
- When CLKPOL = 0, CLKPHA = 1, the SCLK pin will keep low in idle state, and the data will be sampled at the second edge, which is falling edge.
- When CLKPOL = 1, CLKPHA = 0, the SCLK pin will keep high in idle state, and the data will be sampled at the first edge, which is falling edge.
- When CLKPOL = 1, CLKPHA = 1, the SCLK pin will keep high in idle state, and the data will be sampled at the second edge, which is rising edge.

Regardless of the timing mode used, the master and slave configuration must be the same.

Figure 20-4 is the combination timing of four CLKPHA and CLKPOL bits transmitted by SPI when the SPI_CTRL1.LSBFF = 0.

Figure 20-4 Data Clock Timing Diagram



Data format

User can select the data order by setting the SPI_CTRL1.LSBFF bit. When SPI_CTRL1.LSBFF = 0, SPI will transmit the most significant bit (MSB) first; When SPI_CTRL1.LSBFF = 1, SPI will transmit the least significant bit (LSB) first.

User can select the data frame by setting the SPI_CTRL1.DATFF bit.

20.3.2 SPI Operating Mode

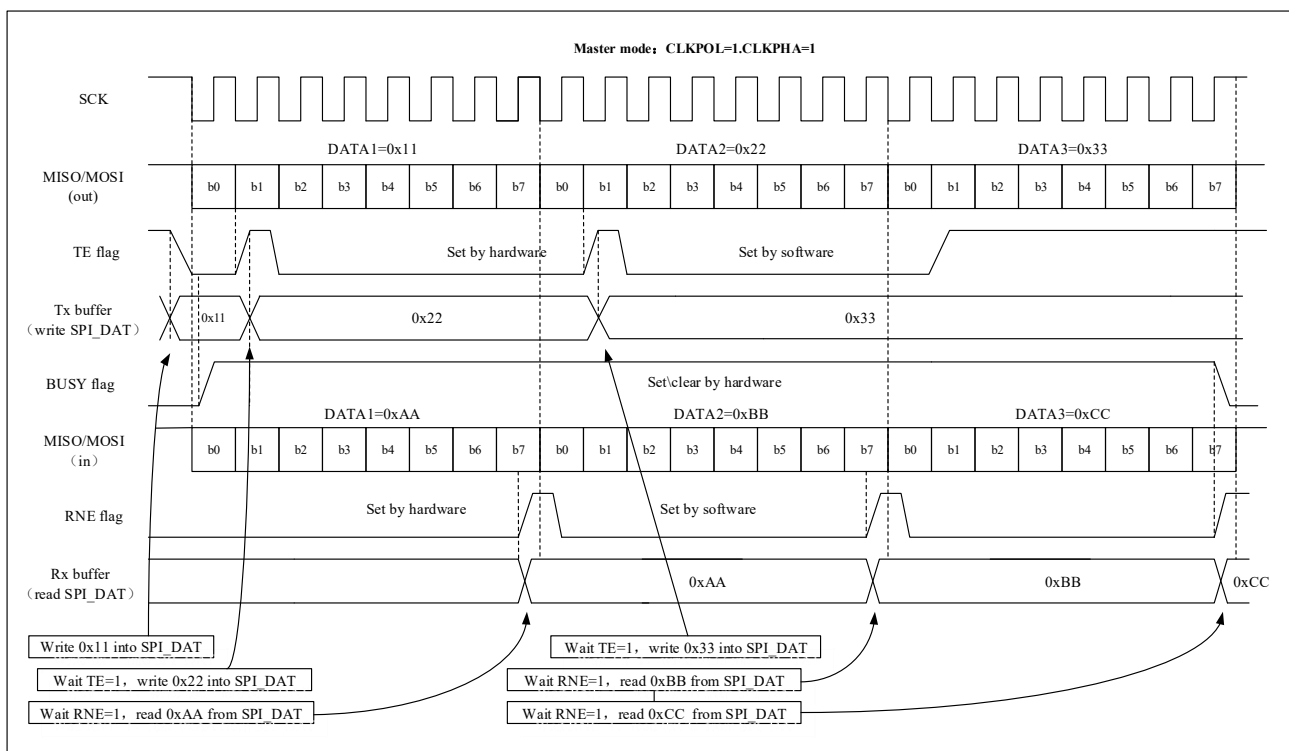
Master full duplex mode

Master full duplex mode (SPI_CTRL1.MSEL = 1, SPI_CTRL1.BIDIRMODE = 0, SPI_CTRL1.ROONLY = 0). After the first data is written to the SPI_DAT register, the transmission will start. When the first bit of the data is transmitted, the data bytes are loaded from the data register into the shift register in parallel, and then according to the configuration of the SPI_CTRL1.LSBFF bit, the data bits follow the MSB or LSB order and are serially shifted to the MOSI pin. At the same time, the data received on the MISO pin is serially shifted into the shift register in the same order and then loaded into the SPI_DAT register in parallel. The software operation process is as follows:

1. Set SPI_CTRL1.SPIEN = 1, Enable SPI module.
2. Write the first data to be transmitted into SPI_DAT register (this operation will clear SPI_STS.TE bit).
3. Wait for SPI_STS.TE bit to be set to '1', and write the second data to be transmitted into SPI_DAT. Wait for SPI_STS.RNE bit to be set to '1', read SPI_DAT to get the first received data, and the SPI_STS.RNE bit will be cleared by hardware while reading SPI_DAT. Repeat the above operation, transmitting subsequent data and receiving n-1 data at the same time;
4. Wait for SPI_STS.RNE bit to be set to '1' to receive the last data;
5. Wait for SPI_STS.TE to be set to '1', then wait for SPI_STS.BUSY bit to be cleared and turn off SPI module.

The process of data transmitting and data receiving can also be implemented in the interrupt handler generated by the rising edge of the SPI_STS.RNE or SPI_STS.TE flag.

Figure 20-5 Change of TE/RNE/BUSY During Continuous Transmission in Master Full Duplex Mode



Master two-wire one-way transmit-only mode

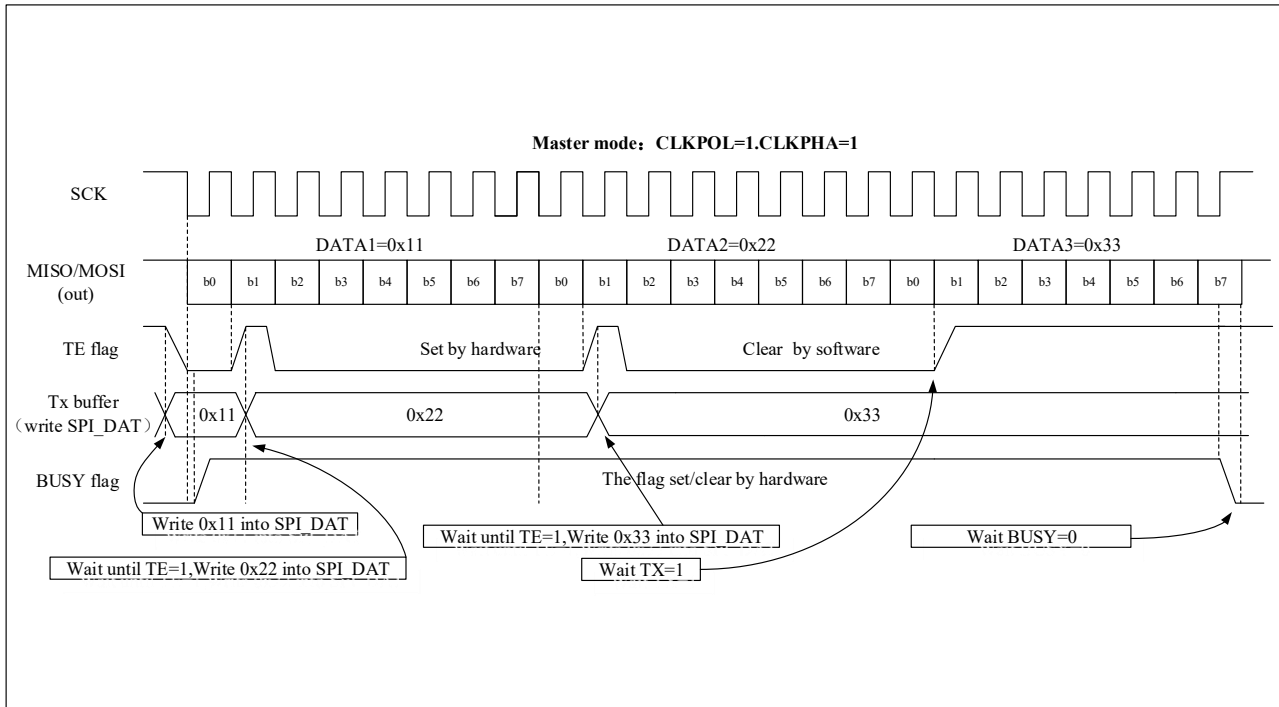
Master two-wire one-way transmit-only mode (SPI_CTRL1.MSEL = 1, SPI_CTRL1.BIDIRMODE = 0, SPI_CTRL1.ROONLY = 0). Master two-wire one-way send-only mode is similar to master full-duplex mode. The difference is that this mode will not read the received data, so the SPI_STS.OVER bit will be set to '1', and the software will ignore it. The software operation process is as follows:

1. Enable SPI module, set SPI_CTRL1.SPIEN = 1.
2. Write the first data to be transmitted into SPI_DAT register (this operation will clear SPI_STS.TE bit).
3. Wait for SPI_STS.TE bit to be set to '1', and write the second data to be transmitted into SPI_DAT. Repeat this operation to transmit subsequent data;

- After writing the last data to SPI_DAT, wait for SPI_STS.TE bit to set '1'; then wait for SPI_STS.BUSY bit to be cleared to complete the transmission of all data.

The process of data transmitting can also be implemented in the interrupt handler generated by the rising edge of the TE flag.

Figure 20-6 Change of TE/BUSY During Host Transmits Continuously in One-Way Only Mode



Master two-wire one-way receive-only mode

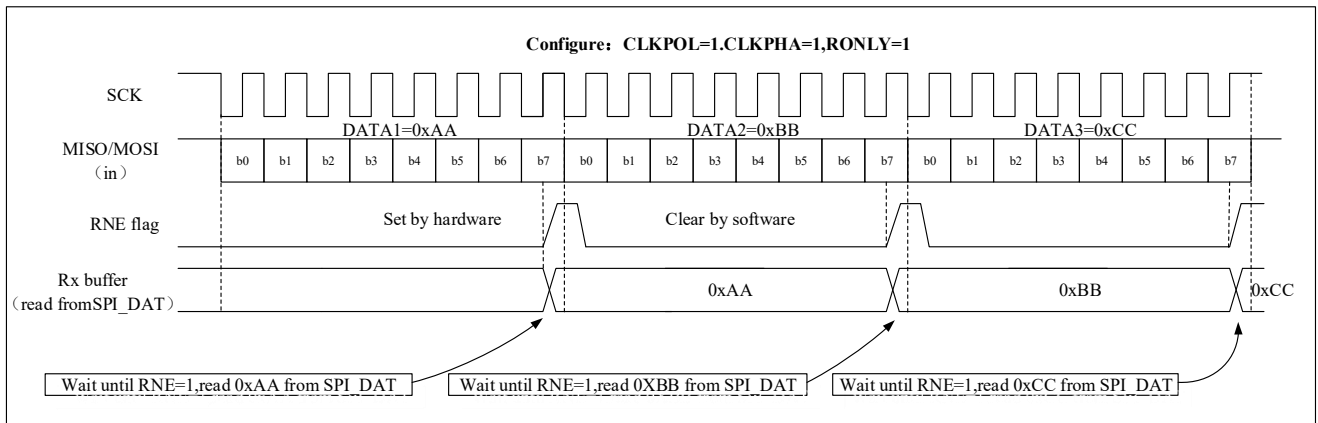
Master two-wire one-way receive-only mode (SPI_CTRL1.MSEL = 1, SPI_CTRL1.BIDIRMODE = 0, SPI_CTRL1.ROONLY = 1). When SPI_CTRL1.SPIEN = 1, the receiving process starts. The data bits from the MISO pin are sequentially shifted into the shift register and then loaded into the SPI_DAT register (receive buffer) in parallel. The software operation process is as follows:

- Enable the receive-only mode (SPI_CTRL1.ROONLY = 1).
- Enable SPI module, set SPI_CTRL1.SPIEN = 1: in master mode, SCLK clock signal is generated immediately, and serial data is continuously received before SPI is turned off (SPI_CTRL1.SPIEN = 0); in slave mode, serial data is continuously received when the SPI master device pulls low the NSS signal and generates SCLK clock.
- Wait for SPI_STS.RNE bit to be set to '1', read the SPI_DAT register to get the received data, and the SPI_STS.RNE bit will be cleared by hardware while reading SPI_DAT register. Repeat this operation to receive all data.

The process of data receiving can also be implemented in the interrupt handler generated by the rising edge of the RNE flag.

Figure 20-7 Change of RNE During Continuous Transmission Occurs in Receive-Only Mode (BIDIRMODE

= 0 And RONLY = 1)



Master one-wire bidirectional transmit mode

Master one-wire bidirectional transmit mode (SPI_CTRL1.MSEL = 1, SPI_CTRL1.BIDIRMODE = 1, SPI_CTRL1.BIDIROEN = 1, SPI_CTRL1.RONLY = 0). After the data is written to the SPI_DAT register (send buffer), the transmission process starts. This mode does not receive data. At the same time as the first data bit is transmitted, the data to be transmitted is loaded into the shift register in parallel, and then according to the configuration of the SPI_CTRL1.LSBFF bit, the SPI serially shifts the data bits to the MOSI pin in MSB or LSB order.

The software operation flow of the master one-wire bidirectional transmit mode is the same as that of the transmit-only mode.

Master one-wire bidirectional receive mode

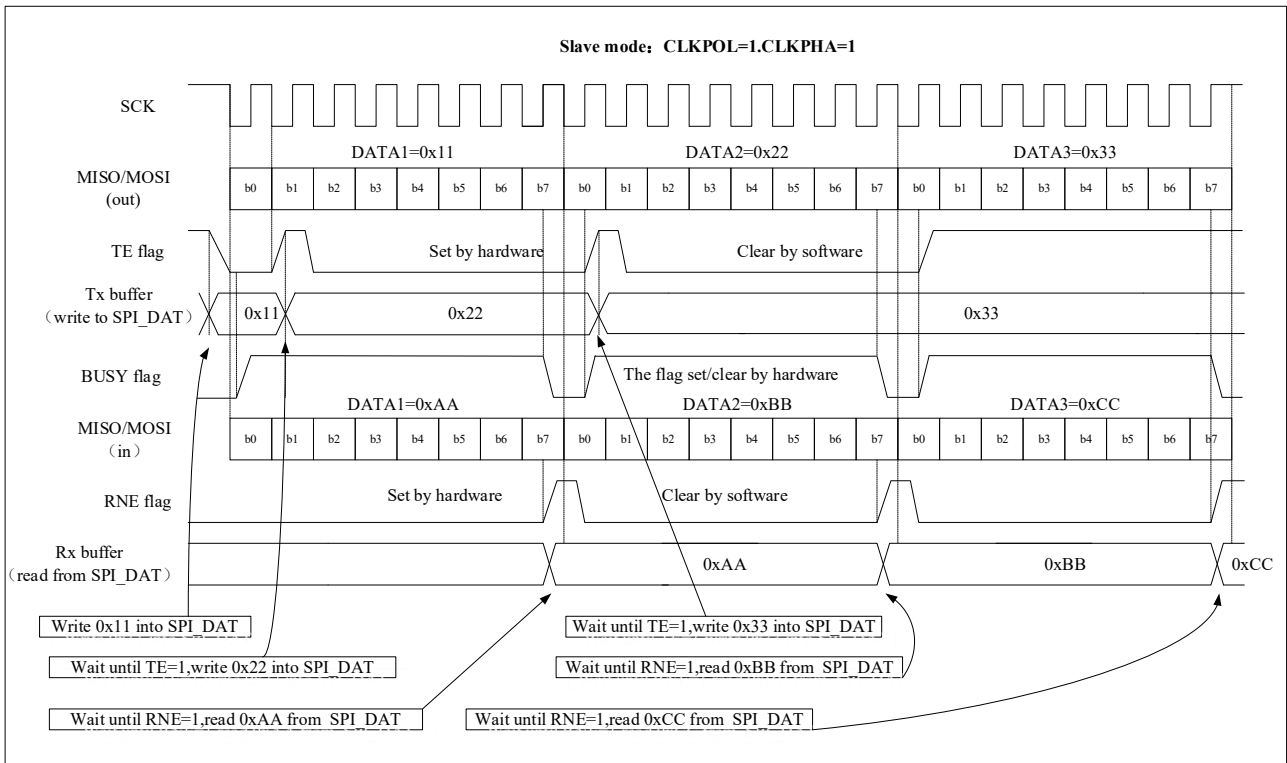
Master one-wire bidirectional receive mode (SPI_CTRL1.MSEL = 1, SPI_CTRL1.BIDIRMODE = 1, SPI_CTRL1.BIDIROEN = 0, SPI_CTRL1.RONLY = 0). When SPI_CTRL1.SPIEN = 1, the receiving process starts. There is no data output in this mode, the received data bits are sequentially and serially shifted into the shift register, and then loaded into the SPI_DAT register (receive buffer) in parallel.

The software operation flow of the master one-wire bidirectional receive mode is the same as that of the receive-only mode.

Slave full duplex mode

Slave full duplex mode (SPI_CTRL1.MSEL = 0, SPI_CTRL1.BIDIRMODE = 0 and SPI_CTRL1.RONLY = 0). The data transmission process begins when the slave device receives the first clock edge. Before the master starts data transmission, software must ensure that the data to be transmitted is written to the SPI_DAT register.

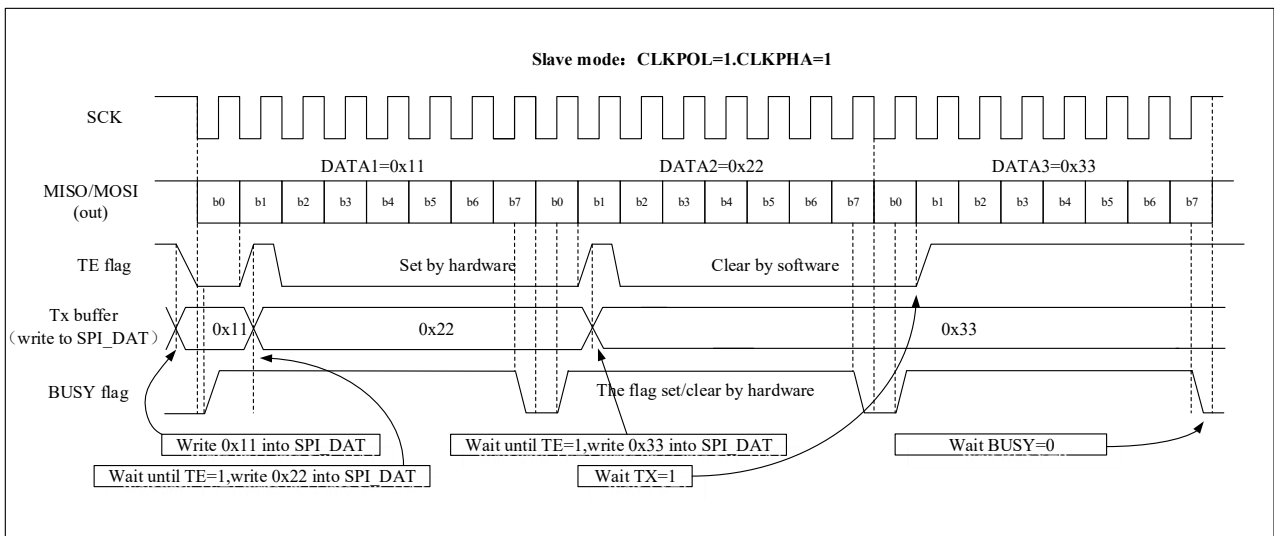
Figure 20-8 Change of TE/RNE/BUSY During The Slave Is Continuously Transmitting in Full Duplex Mode



Slave two-wire one-way transmit-only mode

Slave two-wire one-way transmit-only mode (SPI_CTRL1.MSEL = 0, SPI_CTRL1.BIDIRMODE = 0 and SPI_CTRL1.ROONLY = 0).

Figure 20-9 Change of TE/BUSY During Continuous Transmission in Slave Unidirectional Transmit-Only Mode



Slave two-wire one-way receive-only mode

Slave two-wire one-way receive-only mode (SPI_CTRL1.MSEL = 0, SPI_CTRL1.BIDIRMODE = 0 and SPI_CTRL1.ROONLY = 1).The data receiving process begins when the slave device receives the clock signal and the first data bit from the MOSI pin. The received data bits are sequentially and consecutively shifted serially into an shift register

and then loaded into the SPI_DAT register (receive buffer) in parallel.

Slave one-wire bidirectional transmit mode

Slave one-wire bidirectional transmit mode (SPI_CTRL1.MSEL = 0, SPI_CTRL1.BIDIRMODE = 1 and SPI_CTRL1.BIDIROEN = 1). When the slave device receives the first edge of the clock signal, the transmitting process starts. No data is received in this mode, and the software must ensure that the data to be transmitted has been written in the SPI_DAT register before the SPI slave device starts data transmission.

Slave one-wire bidirectional receive mode

Slave one-wire bidirectional receive mode (SPI_CTRL1.MSEL = 0, SPI_CTRL1.BIDIRMODE = 1 and SPI_CTRL1.BIDIROEN = 0). Data receiving begins when the slave device receives the first clock edge and a data bit from the MOSI pin. There is no data output in this mode, the received data bits are sequentially and consecutively shifted serially into an shift register, and then loaded into the SPI_DAT register (receive buffer) in parallel.

Note: The software operation process of the slave can refer to the master.

SPI initialization process

1. The baud rate of serial clock is defined by the SPI_CTRL1.BR[2:0] bits (this step is ignored if it is working in slave mode).
2. Select the clock priority(SPI_CTRL1.CLKPOL) and phase (SPI_CTRL1.CLKPHA) to define the phase relationship between data transmission and serial clock (refer to Figure 20-4).
3. Set SPI_CTRL1.DATFF bit to define 8-bit or 16-bit data frame format.
4. Configure the SPI_CTRL1.LSBFF bit to define the order of data bit transmission as LSB or MSB.
5. Configure the NSS mode as described above for the NSS function.
6. Operating mode is configured by SPI_CTRL1.MSEL bit, SPI_CTRL1.BIDIRMODE bit, SPI_CTRL1.BIDIROEN bit and SPI_CTRL1.ROONLY bit.
7. Set the SPI_CTRL1.SPIEN = 1 to enable SPI.

Basic transmit and receive process

When SPI transmits a data frame, it firstly loads the data frame from the data buffer into the shift register, then starts to transmit the loaded data. When the data is transferred from the transmission buffer to the shift register, the transmission buffer empty flag is set (SPI_STS.TE = 1), and the next data can be loaded into the transmission buffer; if the TEINTEN bit is set (SPI_CTRL2.TEINTEN = 1), an interrupt will be generated; writing data to the SPI_DAT register will clear the SPI_STS.TE bit.

At the last edge of the sampling clock, when the data is transferred from the shift register to the receive buffer, the receive buffer non-empty flag is set (SPI_STS.RNE = 1), at this time the data is ready and can be read from the SPI_DAT register; if the receive buffer non-empty interrupt is enabled (SPI_CTRL2.RNEINTEN = 1), an interrupt will be generated; the SPI_STS.RNE bit can be cleared by reading the SPI_DAT register data.

In master mode, the transmitting process starts when data is written to the transmission buffer. If the next data has been written into the SPI_DAT register before the current data frame transmitting is completed, the continuous transmission function can be achieved.

In slave mode, the NSS pin is low, and when the first clock edge arrives, the transmission process begins. In order to avoid

accidental data transmission, software must write data to the transmit buffer before data transmission (it is recommended to enable the SPI module before the host transmits the clock).

In some configurations, when the last data is transmitted, the BUSY flag (SPI_STS.BUSY) can be used to wait for the end of the data transmitting.

Continuous and discontinuous transmission

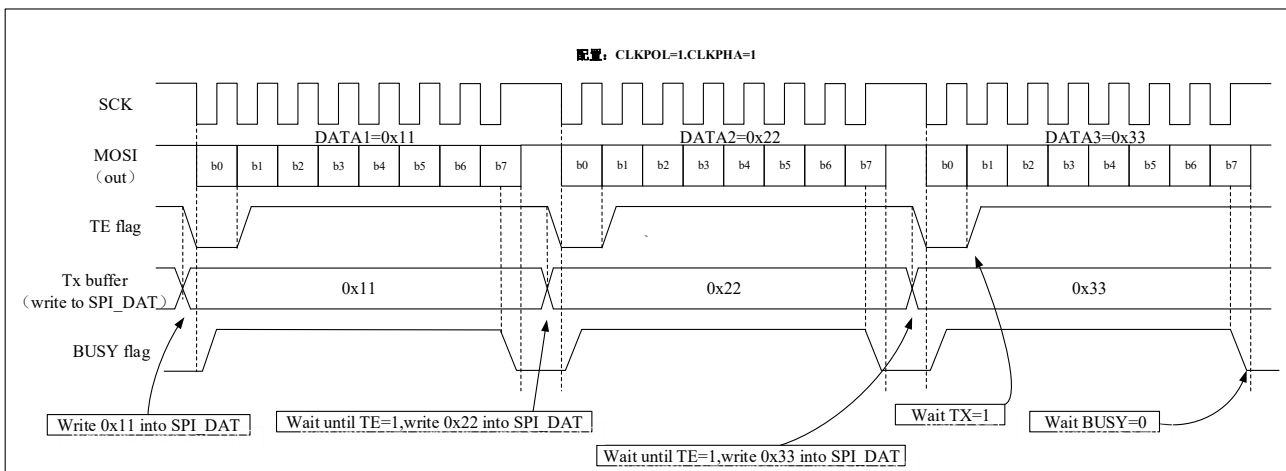
When transmitting data in master mode, if the software is fast enough to detect each TE (SPI_STS.TE) rising edge (or TE interrupt), and the data is written to the SPI_DAT register immediately before the end of the ongoing transmission. At this time, the SPI clock remains continuous between the transmission of data items, and the SPI_STS.BUSY bit will not be cleared, continuous communication can be achieved.

If the software is not fast enough, it will result in discontinuous communication; in this case, the SPI_STS.BUSY bit is cleared between the transmission of each data items (refer to figure below).

In master receive-only mode (SPI_CTRL1.ONLY = 1), communication is always continuous and the BUSY flag (SPI_STS.BUSY) is always high.

In slave mode, the continuity of communication is determined by the SPI master device. However, even if the communication is continuous, the BUSY flag (SPI_STS.BUSY) will be low for at least one SPI clock cycle between each data item (refer to Figure 20-4).

Figure 20-10 Change Of TE/BUSY During BIDIRMODE = 0 and RONLY = 0 Are Transmitted Discontinuously.



20.3.3 Status Flag

The SPI_STS register has 3 flag bits to monitor the status of the SPI:

Send buffer empty flag bit (TE)

When the transmission buffer is empty, the TE flag (SPI_STS.TE) is set to 1, which means that new data can be written into the SPI_DAT register. When the transmission buffer is not empty, the hardware will clear this flag to 0.

Receive buffer non-empty flag bit (RNE)

When the receive buffer is not empty, the RNE flag (SPI_STS.RNE) is set to 1, so the user knows that there is data in the receive buffer. After reading the SPI_DAT register, the hardware will set this flag to 0.

BUSY flag bit (BUSY)

When the transmission starts, the hardware sets the BUSY flag (SPI_STS.BUSY) to 1, and after the transmission ends, the hardware sets the BUSY flag to 0.

Only when the device is in the master one-wire bidirectional receive mode, the BUSY flag (SPI_STS.BUSY) will be set to 0 when the communication is in progress.

The BUSY flag (SPI_STS.BUSY) will be cleared to 0 in the following cases:

- End of transmission (except for continuous communication in master mode);
- Disable the SPI module (SPI_CTRL1.SPIEN = 0);
- The master mode error occurs (SPI_STS.MODERR = 1)

When the communication is discontinuous: the BUSY flag (SPI_STS.BUSY) is cleared to '0' between the transmission of each data item.

When communication is continuous: in master mode, the BUSY flag (SPI_STS.BUSY) remains high during the entire transmission process; In slave mode, the BUSY flag (SPI_STS.BUSY) will be low for 1 SPI clock cycle between each data item transmission. So do not use the BUSY flag to handle the transmitting and receiving of each data item.

20.3.4 Disabling The SPI

To disable the SPI module, different operation modes require different operation steps.

Master or slave full duplex mode

1. Wait for the RNE flag (SPI_STS.RNE) to be set to 1 and the last byte to be received;
2. Wait for the TE flag (SPI_STS.TE) to be set to 1;
3. Wait for the BUSY flag (SPI_STS.BUSY) to be cleared to 0;
4. Disable the SPI module (SPI_CTRL1.SPIEN = 0).

One-way transmit-only mode or bidirectional transmit mode for master or slave

1. After writing the last byte to the SPI_DAT register, wait for the TE flag (SPI_STS.TE) to be set to 1;
2. Wait for the BUSY flag (SPI_STS.BUSY) to be cleared to 0;
3. Disable the SPI module (SPI_CTRL1.SPIEN = 0).

One-way receive-only mode or bidirectional receive mode for master

1. Wait for the penultimate RNE (SPI_STS.RNE) to be set to 1;
2. Before disabling the SPI module (SPI_CTRL1.SPIEN = 0), wait for 1 SPI clock cycle (using software delay);
3. Wait for the last RNE (SPI_STS.RNE) to be set before entering shutdown mode (or turning off the SPI module clock).

One-way receive-only mode or bidirectional receive mode for slave

1. The SPI module can be disabled at any time (SPI_CTRL1.SPIEN = 0), and after the current transmission is over, the SPI module will be disabled;
2. If you want to enter the shutdown mode, you must wait for the BUSY flag (SPI_STS.BUSY) to be set to 0 before entering the shutdown mode (or turn off the SPI module clock).

20.3.5 SPI Communication Using DMA

Users can choose DMA for SPI data transmission, the application program can be released, and the system efficiency can be greatly improved.

When the transmission buffer DMA is enabled (SPI_CTRL2.TDMAEN = 1), each time the TE flag (SPI_STS.TE) bit is 1, a DMA request will be generated, and the DMA will automatically write the data to the SPI_DAT register, which will clear the TE flag (SPI_STS.TE) bit. When the receive buffer DMA is enabled (SPI_CTRL2.RDMAEN = 1), each time the RNE flag (SPI_STS.RNE) bit is set to 1, a DMA request will be generated, and the DMA will automatically read the SPI_DAT register, which will clear the RNE flag (SPI_STS.RNE) bit.

When the SPI is only used for transmitting data, only the transmission DMA channel of the SPI needs to be enabled (SPI_CTRL2.TDMAEN = 1).

When the SPI is only used for receiving data, only the receive DMA channel of the SPI needs to be enabled (SPI_CTRL2.RDMAEN = 1).

In transmit mode, after DMA has transmitted all the data to be transmitted (DMA_INTSTS.TXCF = 1), BUSY flag (SPI_STS.BUSY) can monitor to confirm whether SPI communication is over, which can avoid destroying the transmission of the last data when the SPI is disabled or enters the shutdown mode. Therefore, the software needs to wait for the TE flag (SPI_STS.TE) bit to be set to 1, and wait for the BUSY flag (SPI_STS.BUSY) bit to be set to 0.

Figure 20-11 Transmission Using DMA

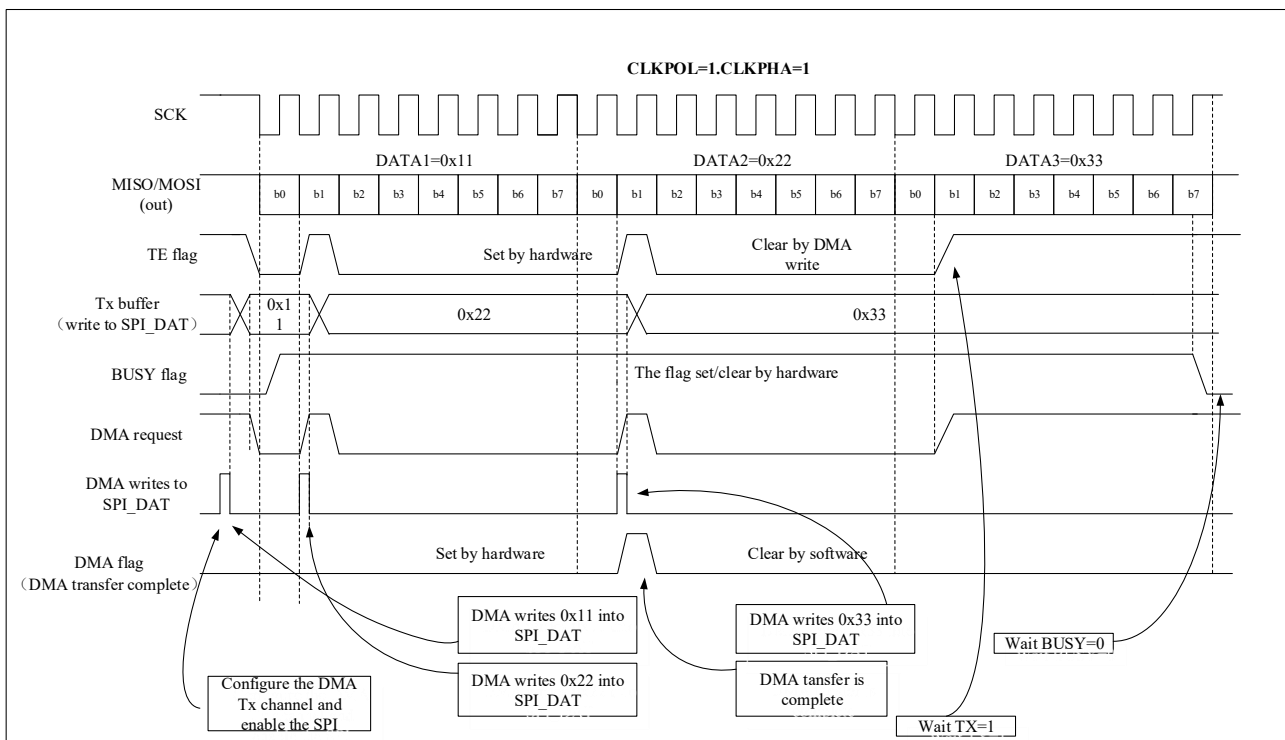
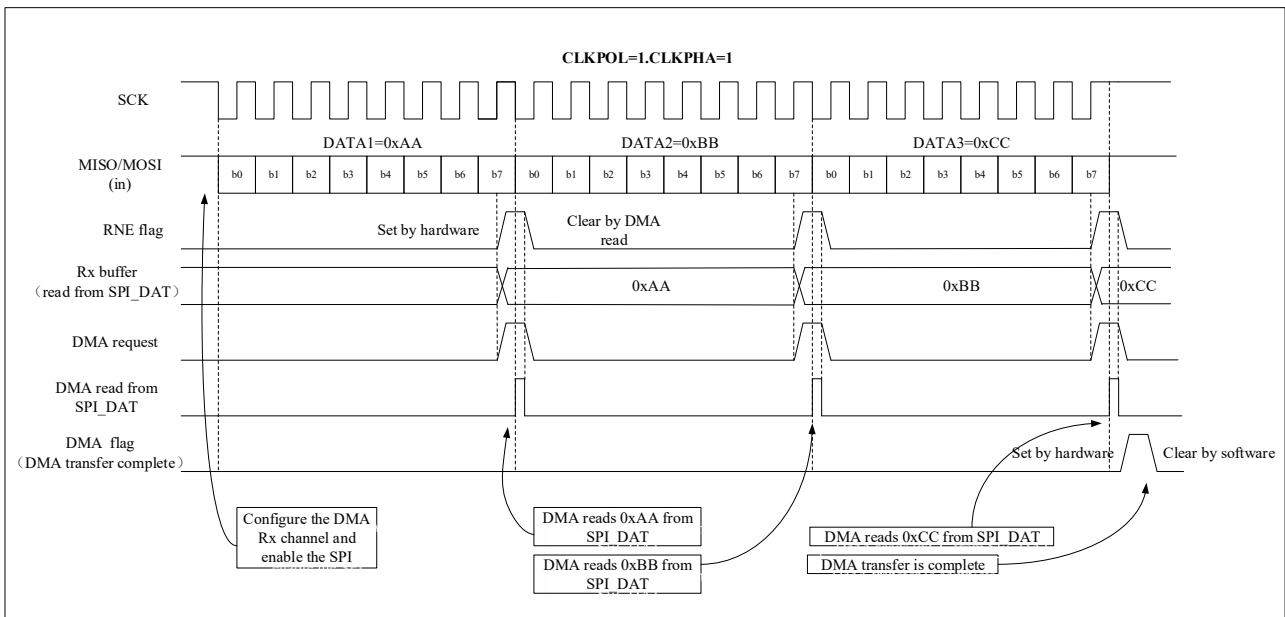


Figure 20-12 Reception Using DMA



20.3.6 CRC Calculation

The SPI contains two independent CRC calculators for data transmitting and data receiving to ensure the correctness of data transmission. According to the transmitting and receiving data frame format, CRC adopts different calculation methods, the 8-bit data frame format adopts CRC8, and the 16-bit data frame format adopts CRC16. The polynomial used in the SPI CRC calculation is set by the SPI_CRCPOLY register, and the user enables the CRC calculation by setting the SPI_CTRL1.CRCEN = 1.

In transmit mode, after the last data is written into the transmission buffer, set the SPI_CTRL1.CRCNEXT = 1, which indicates that the hardware will start transmitting the CRC value (SPI_CRCTDAT value) after transmitting the data. When the CRC is transmitted, the CRC calculation will stop.

In receive mode, after the penultimate data frame is received, set the SPI_CTRL1.CRCNEXT = 1. The received CRC and SPI_CRCDAT values are compared, if they are different, the SPI_STS.CRCERR bit is set to 1. If the SPI_CTRL2.ERRINTEN bit is set to 1, an interrupt will be generated.

In order to keep the synchronization of the next CRC calculation result of the master-slave device, the user should clear the CRC value of the master-slave device. Setting the SPI_CTRL1.CRCEN bit resets the SPI_CRCDAT and SPI_CRCTDAT registers. Take the following steps in order: SPI_CTRL1.SPIEN = 0; SPI_CTRL1.CRCEN = 0; SPI_CTRL1.CRCEN = 1; SPI_CTRL1.SPIEN = 1.

Most importantly, when the SPI is configured in slave mode and CRC is enabled, as long as there is a clock pulse on SCLK pin, the CRC calculation will still be performed even if the NSS pin is high. This situation is common when the master device communicates with multiple slave devices alternately, so it is necessary to avoid CRC misoperation.

When the SPI hardware CRC check is enabled (SPI_CTRL1.CRCEN = 1) and the DMA is enabled, the hardware automatically completes the transmission and reception of CRC bytes when the communication ends.

20.3.7 Error Flag

Master mode failure error (MODERR)

The following two conditions will cause the master mode failure error:

- In NSS pin hardware management mode, the master device NSS pin is pulled low;
- In NSS pin software management mode, the SPI_CTRL1.SSEL bit is set to 0.

When a master mode failure error occurs, the SPI_STS.MODERR bit is set to 1. An interrupt is generated if the user enables the corresponding interrupt (SPI_CTRL2.ERRINTEN=1). The SPI_CTRL1.SPIEN bit and SPI_CTRL1.MSEL bit will be write protected and both are cleared by hardware. SPI is disabled and forced into slave mode

Software performs a read or write operation to the SPI_STS register, then writes to the SPI_CTRL1 register to clear the SPI_STS.MODERR bit (in multi-master mode, the master's NSS pin must be pulled high first).

Normally, the SPI_STS.MODERR bit of the slave cannot be set to 1. However, in a multi-master configuration, the slave's SPI_STS.MODERR bit may be set to 1. In this case, the SPI_STS.MODERR bit indicates that there is a multi-master collision. The interrupt routine can perform a reset or return to the default state to recover from an error state.

Overflow error (OVER)

When the SPI_STS.RNE bit is set to 1, but there is still data transmitted into the receive buffer, an overflow error will occur. At this time, the overflow flag SPI_STS.OVER bit is set to 1. An interrupt is generated if the user enables the corresponding interrupt (SPI_CTRL2.ERRINTEN = 1). All received data is lost and the SPI_DAT register retains only previously unread data.

Reading the SPI_DAT register and the SPI_STS register sequentially can clear the SPI_STS.OVER bit

CRC error (CRCERR)

The CRC error flag is used to verify the validity of the received data. A CRC error occurs when the received CRC value does not match the SPI_CRCRDAT value. At this time, the SPI_STS.CRCERR flag bit is set to '1', and an interrupt will be generated if the user enables the corresponding interrupt (SPI_CTRL2.ERRINTEN = 1).

20.3.8 SPI Interrupt

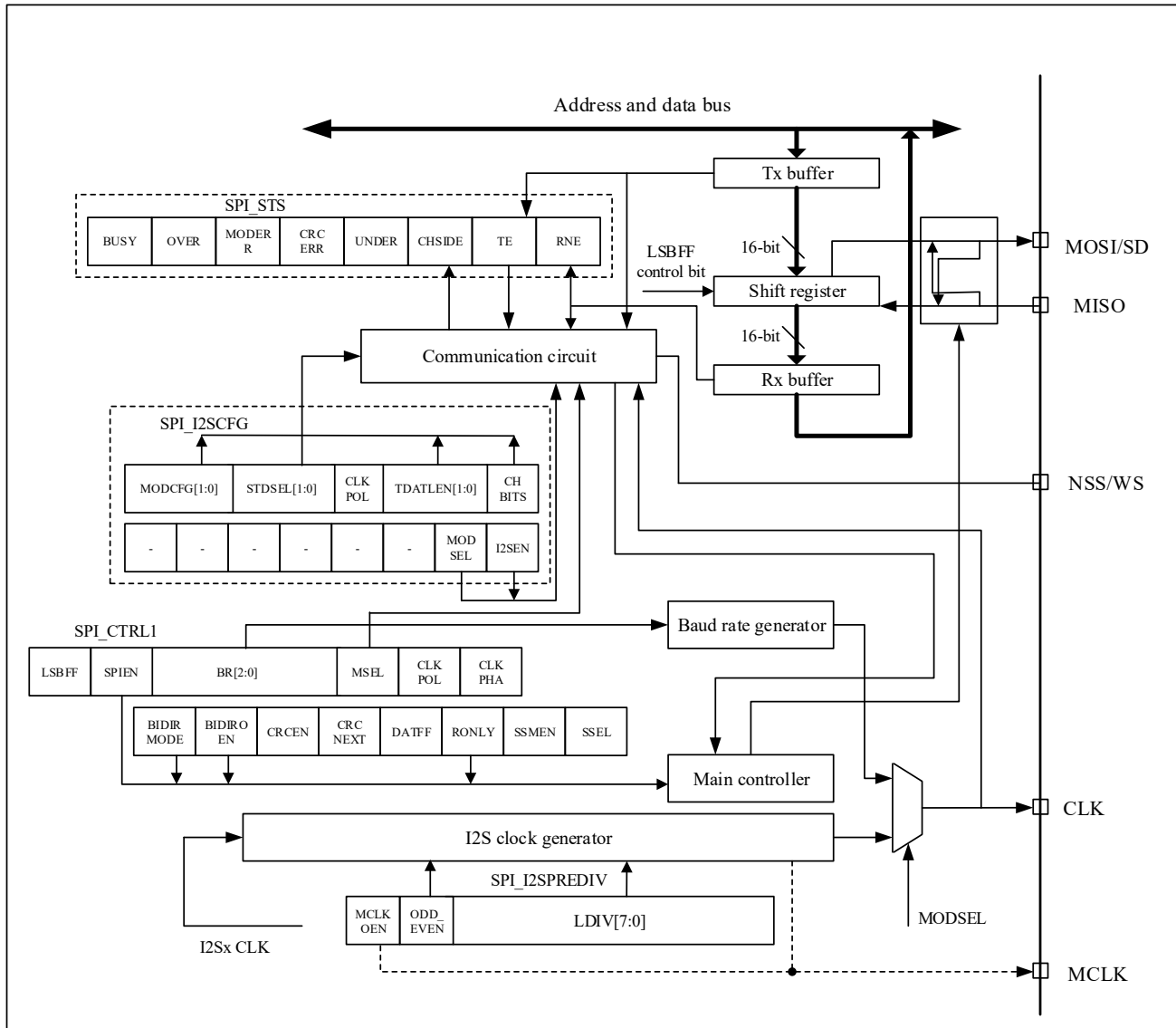
Table 20-1 SPI Interrupt Request

Interrupt Event	Event Flag Bit	Enable Control Bit
Send buffer empty flag	TE	TEINTEN
Receive buffer non empty flag	RNE	RNEINTEN
Master mode failure event	MODERR	ERRINTEN
Overflow error	OVER	
CRC error flag	CRCERR	

20.4 I²S Function Description

The block diagram of I²S is shown in the figure below:

Figure 20-13 I²S Block Diagram



The I²S interface uses the same pins, flags and interrupts as the SPI interface. Setting the SPI_I2SCFG.MODSEL = 1 selects the I²S audio interface.

I²S has a total of 4 pins, 3 of which are shared with SPI:

- CLK: Serial clock (shared with SCLK pin), CLK generates a pulse every time 1-bit audio data is transmitted.
- SD: Serial data (shared with MOSI pin), used for data transmission and receive;
- WS: Channel selection (shared with NSS pin), used as data control signal output in master mode, and used as input in slave mode;
- MCLK: master clock (independent mapping, optional), output $256 \times F_s$ clock signal to ensure better synchronization between systems.

Note: F_s is the sampling frequency of audio signal

In master mode, I²S uses its own clock generator to generate clock signals for communication, and this clock generator is also the clock source of the master clock output (SPI_I2SPREDIV.MCLKOEN = 1, the master clock output is enabled).

20.4.1 Supported Audio Protocols

Four audio standards can be selected by setting the SPI_I2SCFG.STDSEL[1:0] bits:

- I²S Philips standard
- MSB alignment standard
- LSB alignment standard
- PCM standard

The audio data of the left channel and the right channel are usually time-multiplexed, and the left channel always transmits data before the right channel. By checking the SPI_STS.CHSIDE bit, the user can distinguish which channel the received data belongs to. However, in the PCM audio standard, the CHSIDE bit has no meaning.

By setting the SPI_I2SCFG.TDATLEN bits, the user can set the length of the data to be transmitted, and set the data bit width of the channel by setting the SPI_I2SCFG.CHBITS bits. There are 4 data formats for transmitting data as follows:

- 16-bit data is packed into a 16-bit data frame
- 16-bit data is packed into a 32-bit data frame (the first 16 bits are the significant data, and the last 16 bits are set to 0 by hardware)
- 24-bit data is packed into a 32-bit data frame (the first 24 bits data are the significant data, and the latter 8-bit data is set to 0 by hardware)
- 32-bit data is packed into a 32-bit data frame

I²S uses the same SPI_DAT register as SPI to transmit and receive 16-bit wide data. If I²S needs to transmit or receive 24-bit or 32-bit wide data, the CPU needs to read or write the SPI_DAT register twice. On the other hand, when I²S transmits or receives 16-bit wide data, the CPU only needs to read or write the SPI_DAT register once.

Regardless of which data format and communication standard is used, I²S always transmits the the most significant bit (MSB) first.

I²S Philips standard

Using the I²S Philips standard, the device that transmits data changes the data on the falling edge of the clock, and the device that receives data samples the data on the rising edge of the clock. The WS signal should be valid one clock before the first data bit (MSB) is transmitted and will change on the falling edge of the clock signal.

Figure 20-14 I²S Philips Protocol Waveform (16/32-Bit Full Precision, CLKPOL = 0)

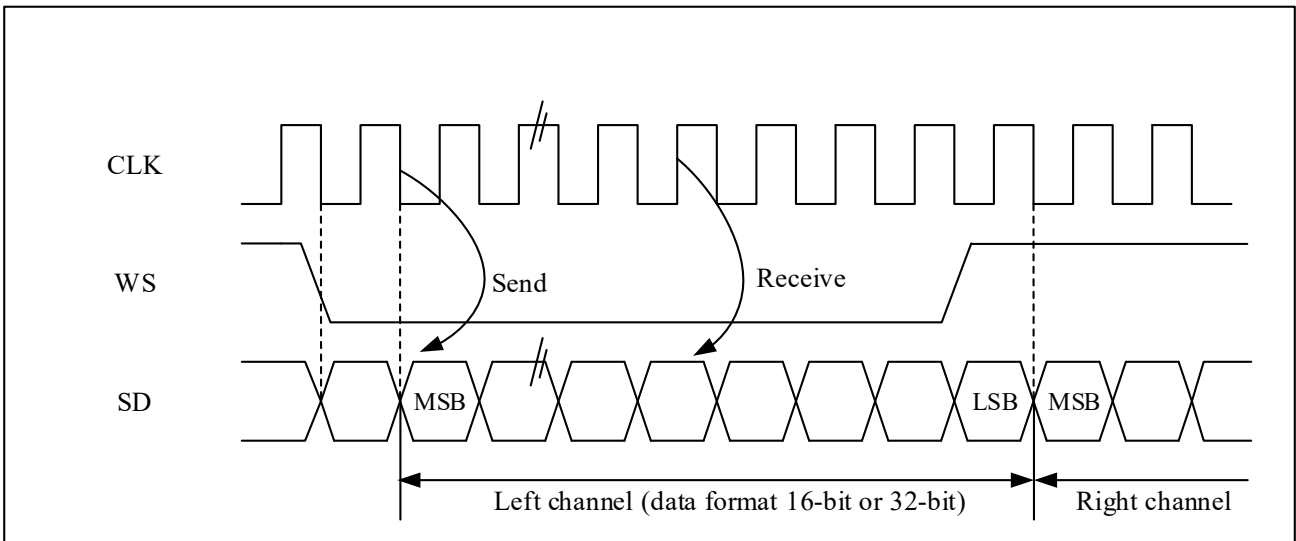
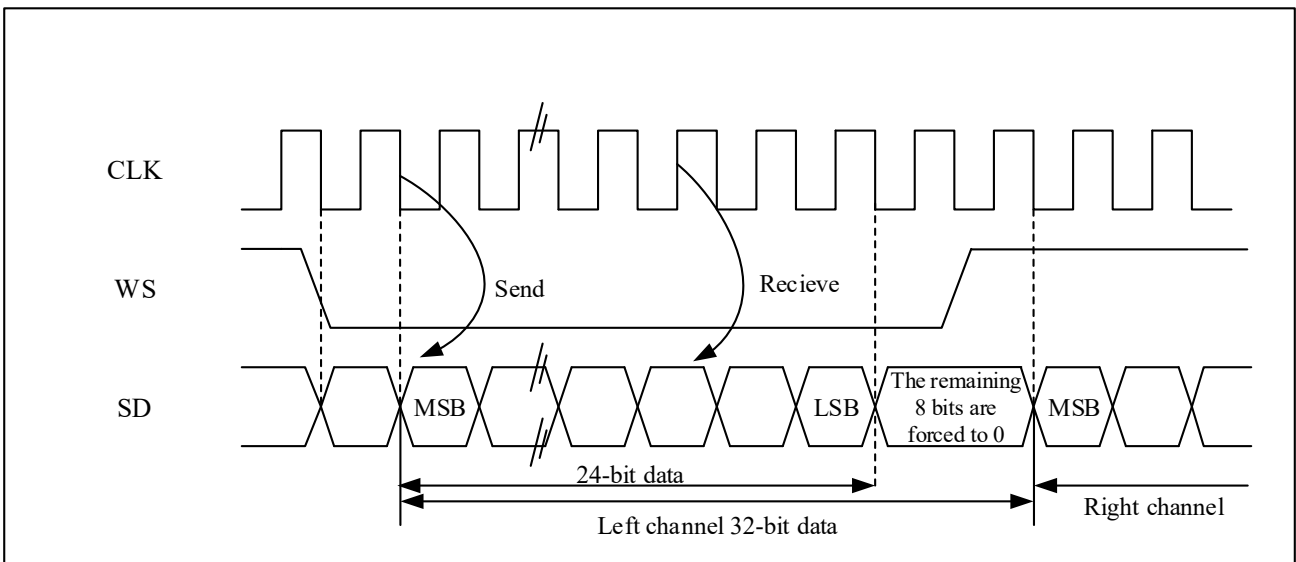


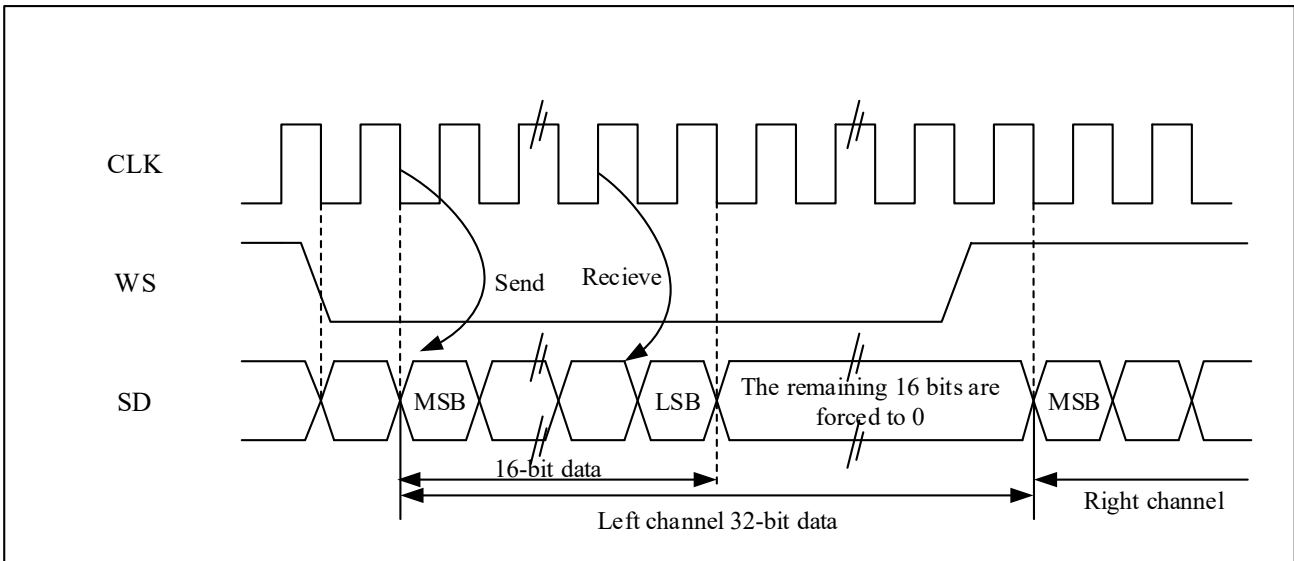
Figure 20-15 I²S Philips Protocol Standard Waveform (24-Bit Frame, CLKPOL = 0)



If the 24-bit data needs to be packaged into 32-bit data frame format, the CPU needs to read or write the SPI_DAT register twice during each frame of data transmission. For example, if the user transmits 24-bit data 0x95AA66, the CPU will first write 0x95AA into the SPI_DAT register, and then write 0x66XX into the SPI_DAT register (only the upper 8-bit data is valid, the lower 8-bit data is meaningless and can be any value); if the user receives 24-bit data 0x95AA66, the CPU will first read the SPI_DAT register to get 0x95AA, and then read the SPI_DAT register to get 0x6600 (only the upper 8-bit data is valid, and the lower 8-bit data is always 0).

Figure 20-16 I²S Philips Protocol Standard Waveform (16-Bit Extended To 32-Bit Packet Frame, CLKPOL =

0)



If 16-bit data needs to be packed into 32-bit data frame format, the CPU only needs to read or write the SPI_DAT register once for each frame of data transmission. The lower 16 bits of data for expansion to 32 bits are always set to 0x0000. For example, if the user transmits or receives 16-bit data 0x89C1 (extended to 32-bit data is 0x89C10000). In the process of transmitting data, the upper 16-bit half word (0x89C1) needs to be written into the SPI_DAT register; the user can write new data until the SPI_STS.TE bit is set. An interrupt is generated if the user enables the corresponding interrupt. The transmitting is performed by hardware, even if the last 16 bits (0x0000) are not transmitted, the hardware will set the TE (SPI_STS.TE) bit to 1 and the corresponding interrupt will be generated. In the process of receiving data, the RNE flag (SPI_STS.RNE) will be set to 1 after each time the device receives the upper 16-bit halfword (0x89C1). An interrupt is generated if the user enables the corresponding interrupt. In this way, the CPU has more time between 2 reads and writes, which can prevent underflow or overflow from happening.

MSB alignment standard

In the MSB alignment standard, the device transmits the data on the falling edge of the clock, and the device receives the data on the rising edge of the clock. The WS signal and the first data bit (MSB) are generated simultaneously.

The standard data receiving and transmitting processing mode is the same as I²S Philips standard.

Figure 20-17 The MSB Is Aligned with 16-Bit or 32-Bit Full Precision, CLKPOL = 0.

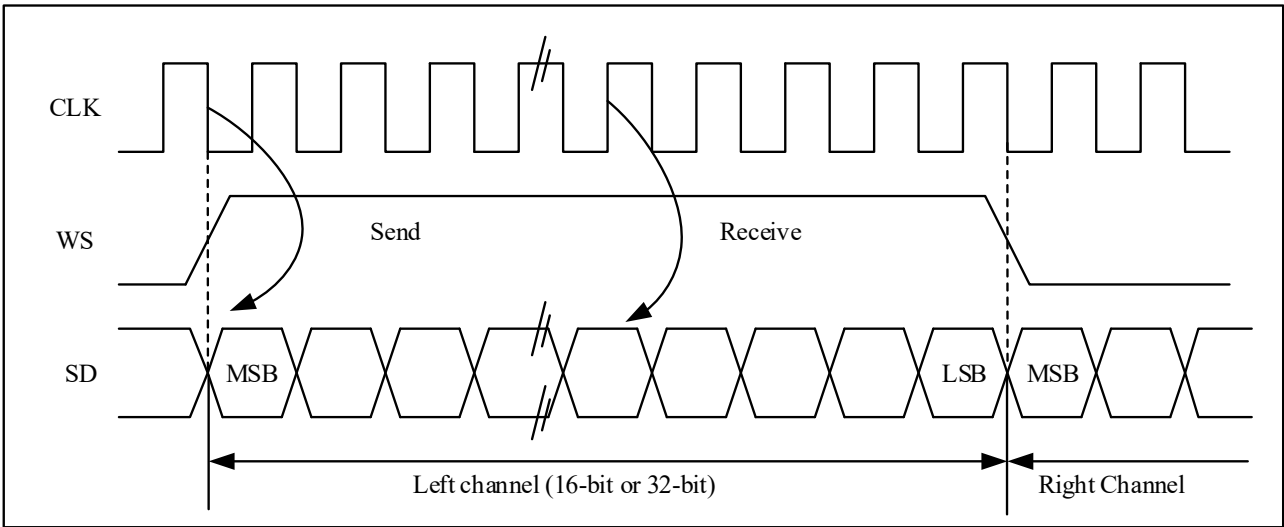


Figure 20-18 MSB Aligns 24-Bit Data, CLKPOL = 0

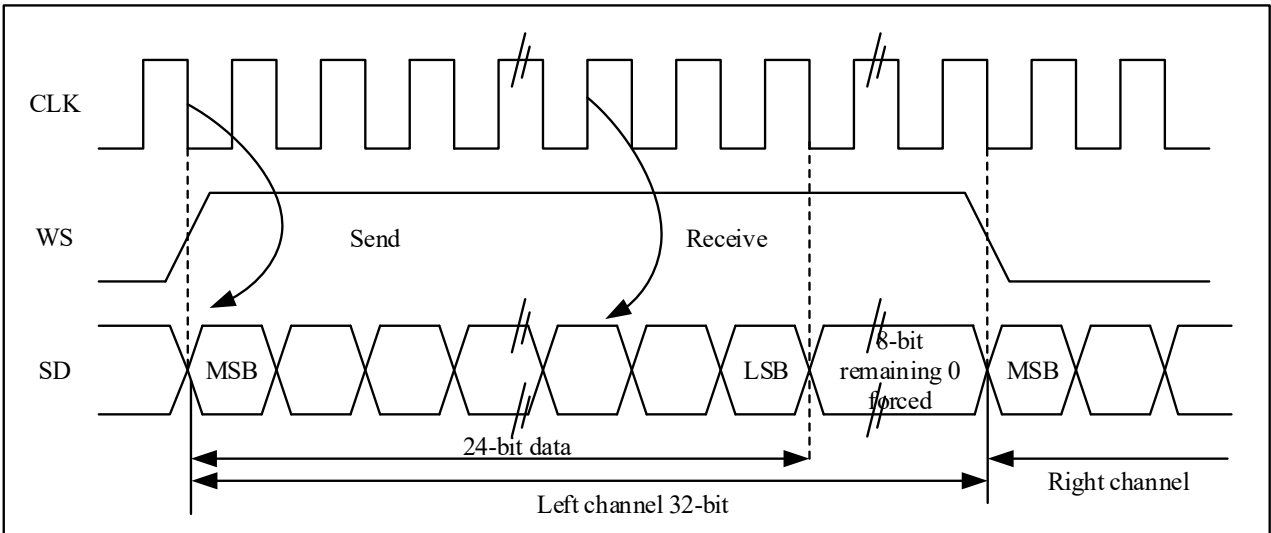
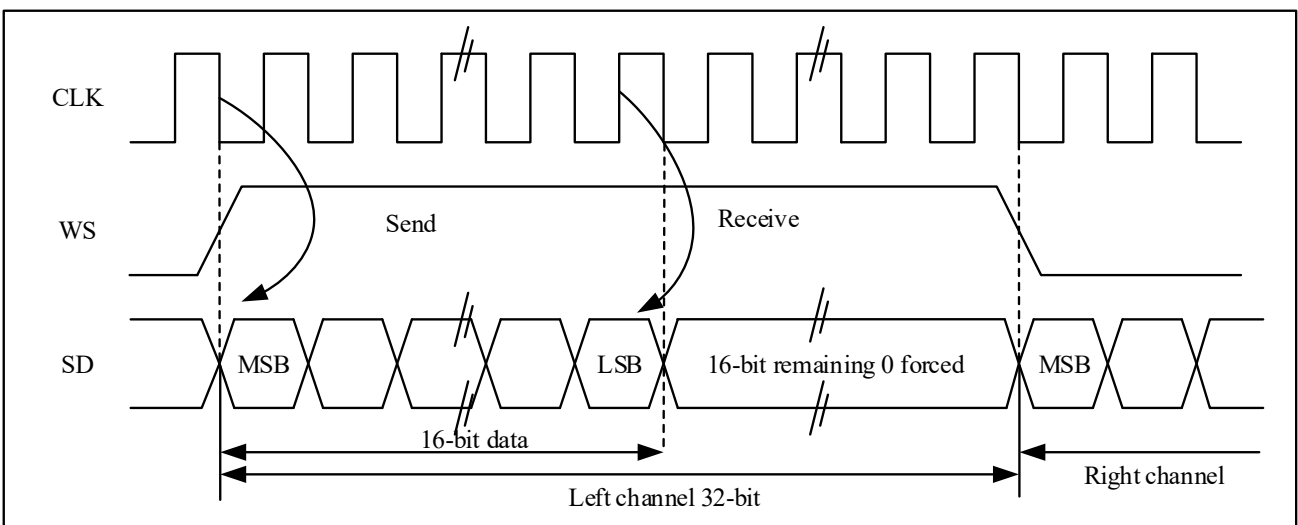


Figure 20-19 MSB-Aligned 16-Bit Data Is Extended to 32-Bit Packet Frame, CLKPOL = 0



LSB alignment standard

In 16-bit or 32-bit full-accuracy frame format, LSB alignment standard is the same as MSB alignment standard.

Figure 20-20 LSB Alignment 16-Bit or 32-Bit Full Precision, CLKPOL = 0

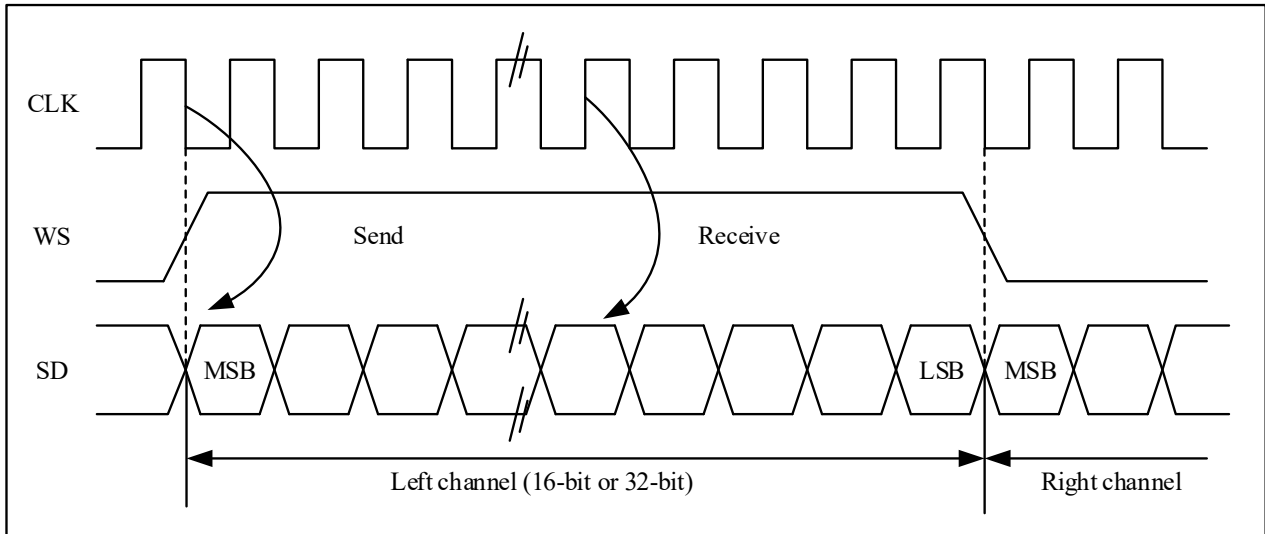
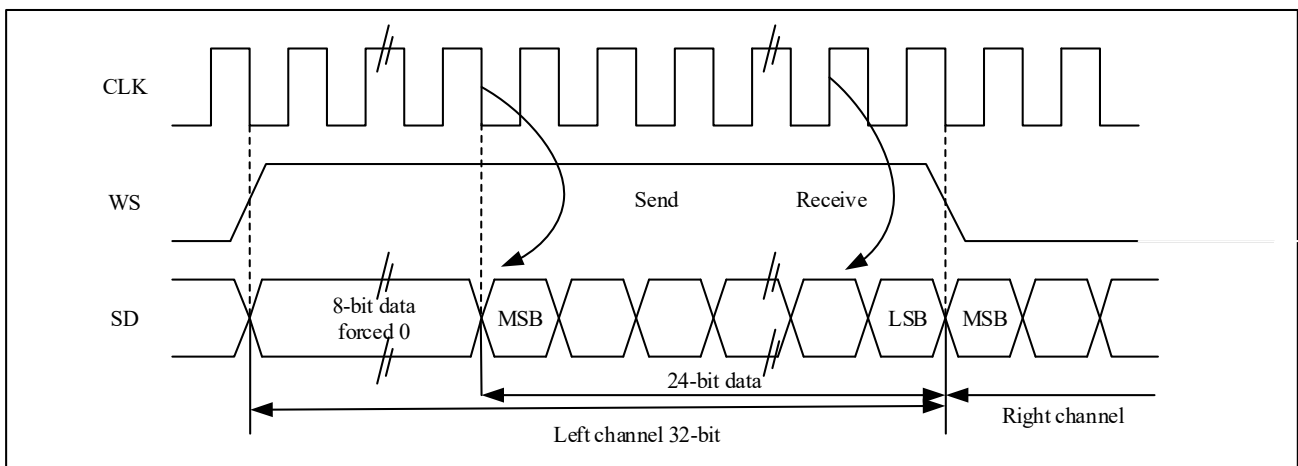
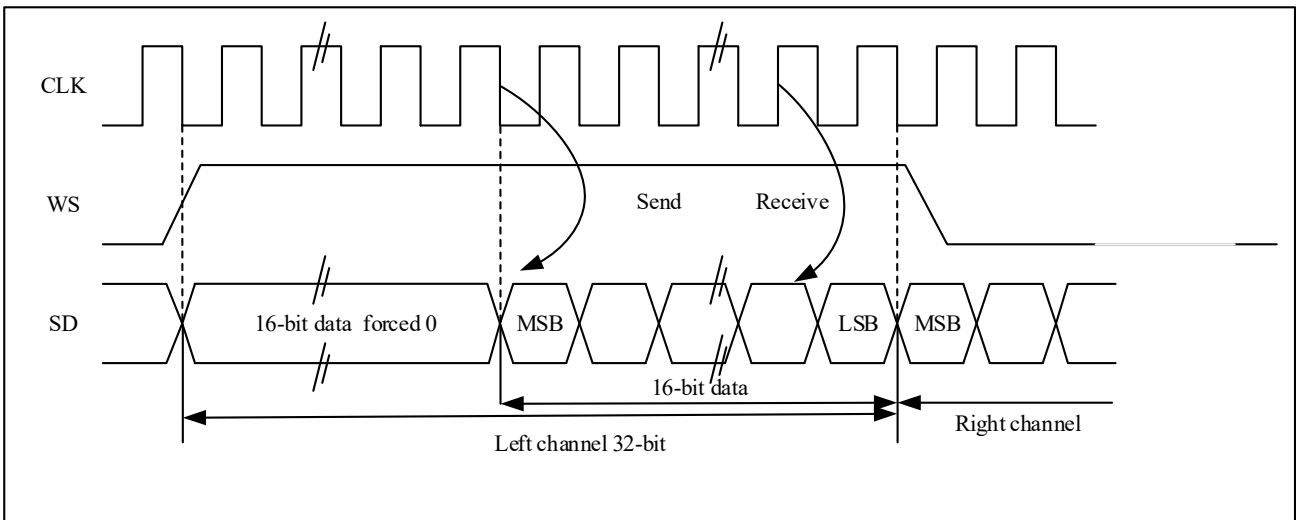


Figure 20-21 LSB Aligns 24-Bit Data, CLKPOL = 0



If the 24-bit data needs to be packed into the 32-bit data frame format, the CPU needs to read or write the SPI_DAT register twice during each frame of data transmission. For example, if the user transmits 24-bit data 0x95AA66, the CPU will first write 0xXX95 (only the lower 8-bit data is valid, the upper 8-bit data is meaningless and can be any value) into the SPI_DAT register, and then write 0xAA66 into the SPI_DAT register. If the user receives 24-bit data 0x95AA66, the CPU will first read the SPI_DAT register to get 0x0095 (only the lower 8 bits are valid, the upper 8 bits are always 0), and then read the SPI_DAT register to get 0xAA66.

Figure 20-22 LSB Aligned 16-Bit Data Is Extended to 32-Bit Packet Frame, CLKPOL = 0



If the 16-bit data needs to be packaged into a 32-bit data frame format, the CPU only needs to read or write the SPI_DAT register once for each frame of data transmission. The upper 16 bits of extended to 32 bits data are set to 0x0000 by hardware, if the user transmits or receives 16-bit data 0x89C1 (extended to 32-bit data is 0x000089C1). In the process of transmitting data, the upper 16-bit half word (0x0000) needs to be written to the SPI_DAT register first; once the valid data starts to be transmit, the next TE (SPI_STS.TE) event will be generated. In the process of receiving data, once the device receives valid data, the RNE (SPI_STS.RNE) event will be generated. In this way, the CPU has more time between 2 reads and writes, which can prevent underflow or overflow from happening.

PCM standard

In the PCM standard, there are two frame structures, short frame and long frame. The user can select the frame structure by setting the SPI_I2SCFG.PCMFSYNC bits. The WS signal indicates frame synchronization information. The WS signal for synchronizing long frames is 13 bits effective; the WS signal length for synchronizing short frames is 1 bit.

The standard data receiving and traning processing mode is the same as I²S Philips standard.

Figure 20-23 PCM Standard Waveform (16 Bits)

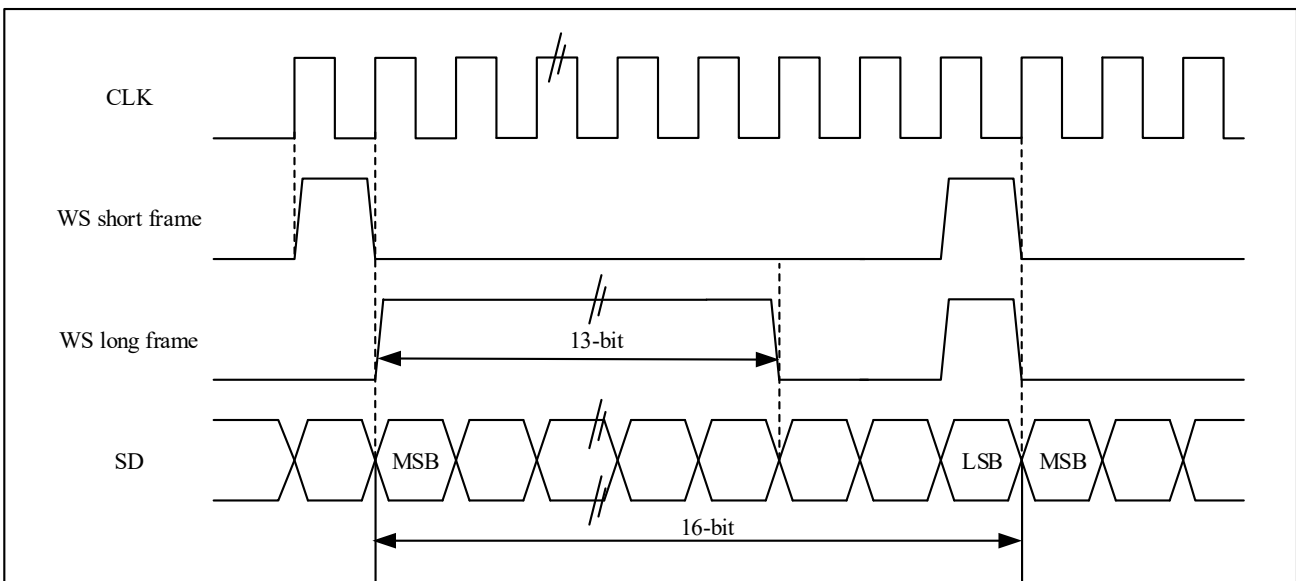
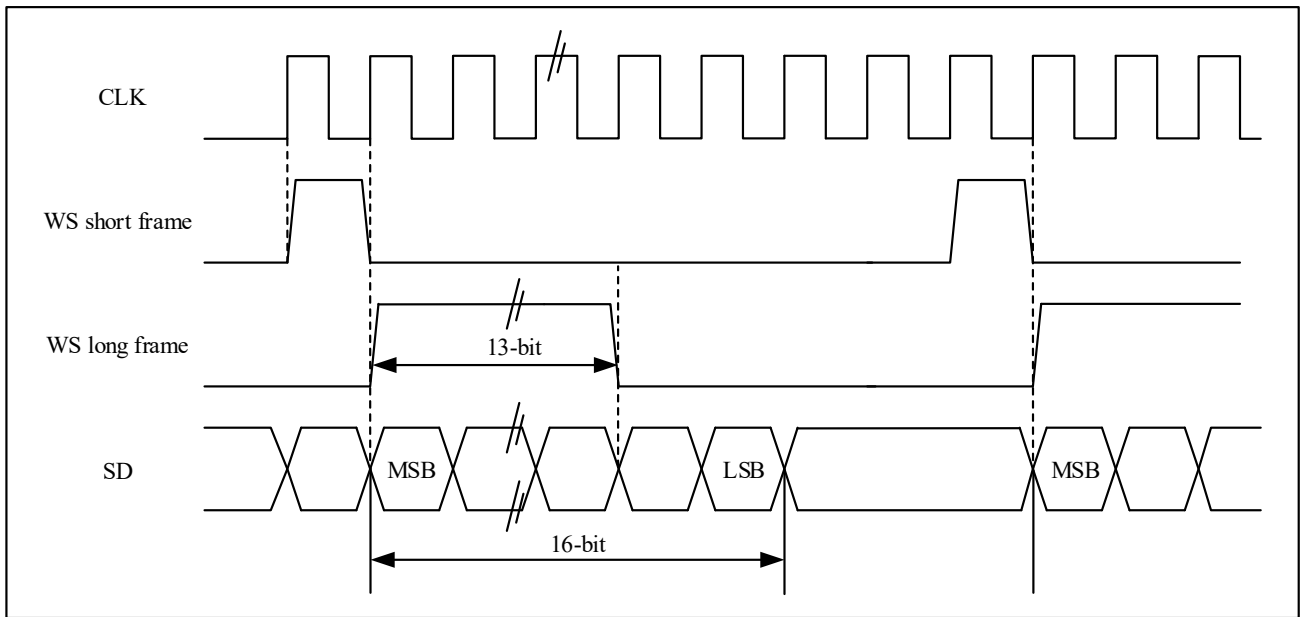


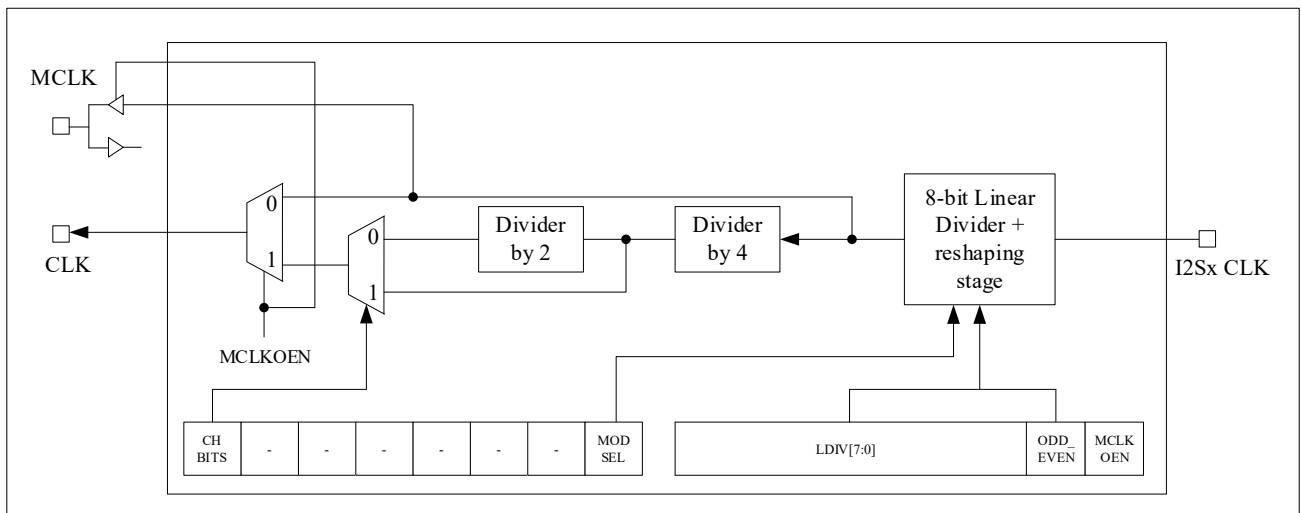
Figure 20-24 PCM Standard Waveform (16-Bit Extended To 32-Bit Packet Frame)



20.4.2 Clock Generator

In the master mode, the linear divider needs to be set correctly in order to obtain the desired audio frequency.

Figure 20-25 I²S Clock Generator Structure



Note: The clock source of I²Sx CLK is HSI, HSE or PLL system clock that drives AHB clock.

The bit rate of I²S determines the data flow on the I²S data line and the frequency of the I²S clock signal.

$$\text{I}^2\text{S bit rate} = \text{number of bits per channel} \times \text{number of channels} \times \text{sampling audio frequency}$$

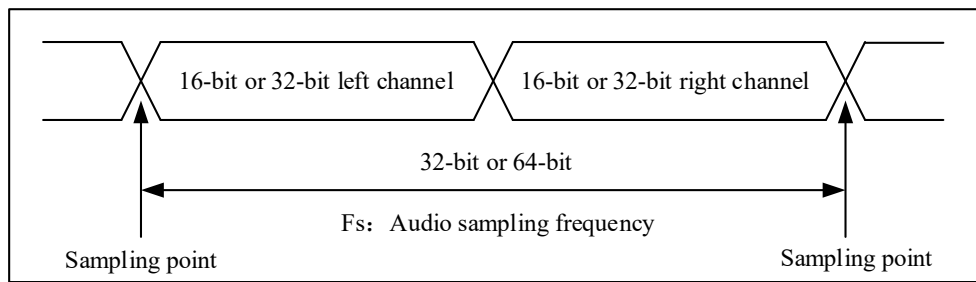
For a signal with left and right channels and 16-bit audio, the I²S bit rate is calculated as:

$$\text{I}^2\text{S bit rate} = 16 \times 2 \times F_s$$

If the packet length is 32 bits, there are:

$$\text{I}^2\text{S bit rate} = 32 \times 2 \times F_s$$

Figure 20-26 Audio Sampling Frequency Definition



The sampling signal frequency of the audio can be set by setting the SPI_I2SPREDIV.ODD_EVEN bit and the SPI_I2SPREDIV.LDIV[7:0] bits. The audio can be sampled at 96kHz, 48kHz, 44.1kHz, 32kHz, 22.05kHz, 16kHz, 11.025kHz, or 8kHz (or any value within this range). Set the linear divider according to the following formula:

When MCLKOEN = 1 and CHBITS= 0, $F_s = I^2Sx CLK / [(16 \times 2) \times ((2 \times LDIV) + ODD_EVEN) \times 8]$

When MCLKOEN = 1 and CHBITS = 1, $F_s = I^2Sx CLK / [(32 \times 2) \times ((2 \times LDIV) + ODD_EVEN) \times 4]$

When MCLKOEN = 0 and CHBITS = 0, $F_s = I^2Sx CLK / [(16 \times 2) \times ((2 \times LDIV) + ODD_EVEN)]$

When MCLKOEN = 0 and CHBITS = 1, $F_s = I^2Sx CLK / [(32 \times 2) \times ((2 \times LDIV) + ODD_EVEN)]$

The exact audio frequency can be obtained by referring to the clock configuration in the table below.

Table 20-2 Use The Standard 8mhz HSE Clock to Get Accurate Audio Frequency.

SYSCLK (MHz)	I ² S_LDIV		I ² S_ODD_EVEN		MCLK	Target Fs(Hz)	Real Fs(Hz)		Error	
	16 bits	32 bits	16 bits	32 bits			16 bits	32 bits	16 bits	32 bits
48	8	4	0	0	without	96000	93750	93750	2.34%	2.34%
48	15	8	1	0	without	48000	48387.1	46875	0.81%	2.34%
48	17	8	0	1	without	44100	44117.65	44117.65	0.04%	0.04%
48	23	11	1	1	without	32000	31914.89	32608.7	0.27%	17.00%
48	34	17	0	0	without	22050	22058.82	22058.82	0.04%	0.04%
48	47	23	0	1	without	16000	15957.45	15957.45	0.27%	0.27%
48	68	34	0	0	without	11025	11029.41	11029.41	0.04%	0.04%
48	94	47	0	0	without	8000	7978.72	7978.72	0.27%	0.27%
48	1	1	0	0	yes	96000	93750	93750	2.34%	2.34%
48	2	2	0	0	yes	48000	46875	46875	2.34%	2.34%
48	2	2	0	0	yes	44100	46875	46875	6.29%	6.29%
48	3	3	0	0	yes	32000	31250	31250	2.34%	2.34%
48	4	4	1	1	yes	22050	20833.33	20833.33	5.51%	5.51%
48	6	6	0	0	yes	16000	15625	15625	2.34%	2.34%
48	8	8	1	1	yes	11025	11029.41	11029.41	0.04%	0.04%
48	11	11	1	1	yes	8000	8152.17	8152.17	1.90%	1.90%

20.4.3 I²S Transmit and Receive Sequence

I²S initialization process

1. The user can set the SPI_I2SPREDIV.LDIV [7:0] bits and SPI_I2SPREDIV.ODD_EVEN bit to configure the related prescaler and serial clock baud rate;

2. If the user needs the master device to provide the main clock MCLK to the external DAC/ADC audio device, set the SPI_I2SPREDIV.MCLKOEN = 1. (Calculate LDIV and ODD_EVEN according to different clock outputs, refer to Section 20.4.2).
3. The user can set the SPI_I2SCFG.CLKPOL bit to define the polarity of the communication clock when idle; the user can set the SPI_I2SCFG.MODSEL = 1 to configure the device to be in I²S mode, and set SPI_I2SCFG.MODCFG[1:0] bits to select the I²S master-slave mode and transmission direction (transmit or receive); set SPI_I2SCFG.STDSEL[1:0] bits to select the corresponding I²S standard (under the PCM standard, set the SPI_I2SCFG.PCMFSYNC bit to select the PCM frame synchronization mode); set SPI_I2SCFG.TDATLEN [1: 0] bits to select length of data to be transmitted, and select the number of data bits of per channel by set the SPI_I2SCFG.CHBITS bit;
4. When user needs to enable interrupt or DMA, the configuration operation is the same as SPI;
5. Finally, set the SPI_I2SCFG.I2SEN = 1 to start I²S communication.

Master mode transmitting process

When I²S operates in master mode, the CLK pin outputs the serial clock, the WS pin generates the channel selection signal, and sets the SPI_I2SPR.MCLKOEN bit to select whether to output the master clock (MCLK).

The transmitting process begins when data is written to the transmission buffer. When the data of the current channel is moved from the transmission buffer to the shift register in parallel, the flag bit TE (SPI_STS.TE) is set to '1'. At this time, the data of the other channel should be written into SPI_DAT. The channel corresponding to the current data to be transmitted is confirmed by the flag bit CHSIDE (SPI_STS.CHSIDE). The value of CHSIDE (SPI_STS.CHSIDE) is updated when TE (SPI_STS.TE) is set to '1'. A complete data frame includes left and right channels, and only part of the data frame cannot be transmitted. When the flag bit TE (SPI_STS.TE) is set to '1', if the SPI_CTRL2.TEINTEN = 1, an interrupt will be generated.

The operation of writing data depends on the selected I²S standard. Refer to chapter 20.4.1 for details.

When the user wants to disable the I²S function, wait for the TE flag (SPI_STS.TE) bit to be 1 and the BUSY flag (SPI_STS.BUSY) bit to be 0, then clear the SPI_I2SCFG.I2SEN bit to 0.

Slave mode transmitting process

The transmitting process of the slave mode is similar to that of the master mode, the difference is as follows:

When I²S operates in slave mode, there is no need to configure the clock, and the CLK pin and WS pin are connected to the corresponding pins of the master device. The transmitting process begins when an external master transmits a clock signal and a WS signal requires data transmission. Only when the slave device is enabled and the data has been written to the I²S data register, the external master device can start communication.

When the first clock edge representing the next data transmission arrives, the new data has not been written into the SPI_DAT register, an underflow occurs, and the SPI_STS.UNDER flag bit is set to 1. If the SPI_CTRL2.ERRINTEN bit is set to 1, an interrupt is generated to indicate that an error has occurred.

The SPI_STS.CHSIDE flag indicates which channel the currently transmitted data corresponds to. Compared with the master mode transmitting process, in the slave mode, the SPI_STS.CHSIDE depends on the WS signal of the external master I²S device (WS signal is 1 means the left channel)

Master mode receiving process

The audio data is always received in 16-bit packets. According to the configured data and channel length, the received audio data will need to be transmitted to the receive buffer once or twice.

When the data is transmitted from the shift register to the receive buffer, the SPI_STS.RNE flag bit is set to 1. At this time, the data is ready and can be read from the SPI_DAT register. If the SPI_CTRL2.RNEINTEN bit is set to 1, an interrupt will be generated. Reading the SPI_DAT register to clear the SPI_STS.RNE flag. If the previously received data is not read and the new data is received again, an overflow occurs and the SPI_STS.OVER flag is set to 1. If the SPI_CTRL2.ERRINTEN bit is set to 1, an interrupt is generated to indicate that an error has occurred.

The channel corresponding to the currently transmitted data can be confirmed by the SPI_STS.CHSIDE bit. When the SPI_STS.RNE flag bit is set to 1, the SPI_STS.CHSIDE value is updated.

The operation of reading data depends on the selected I²S standard. Refer to section 20.4.1 for details.

When I²S function is disabled, different audio standards, data length and channel length adopt different operation steps:

- The data length is 16 bits, the channel length is 32 bits (SPI_I2SCFG.TDATLEN = 00, SPI_I2SCFG.CHBITS = 1), LSB alignment standard (SPI_I2SCFG.STDSEL = 10).
 1. Wait for the penultimate RNE flag (SPI_STS.RNE) bit to be set to '1'.
 2. Software delay, waiting for 17 I²S clock cycles.
 3. Disable I²S (SPI_I2SCFG.I2SEN = 0).
- The data length is 16 bits, the channel length is 32 bits (SPI_I2SCFG.TDATLEN = 00 and SPI_I2SCFG.CHBITS = 1), the MSB alignment standard (SPI_I2SCFG.STDSEL = 01), I²S Philips standard (SPI_I2SCFG.STDSEL = 00) or PCM standard (SPI_I2SCFG.STDSEL = 11)
 1. Wait for the last RNE flag (SPI_STS.RNE) bit to be set to '1'.
 2. Software delay, waiting for 1 I²S clock cycle.
 3. Disable I²S (SPI_I2SCFG.I2SEN = 0).
- Other combinations of SPI_I2SCFG.TDATLEN and SPI_I2SCFG.CHBITS and any audio mode selected by SPI_I2SCFG.STDSEL:
 1. Wait for the penultimate RNE flag (SPI_STS.RNE) bit to be set to '1'.
 2. Software delay, waiting for 1 I²S clock cycle.
 3. Disable I²S (SPI_I2SCFG.I2SEN = 0).

Slave mode receiving process

The receiving process of the slave mode is similar to that of the master mode, with the following differences:

The CHSIDE flag (SPI_STS.CHSIDE) indicates which channel corresponds to the currently transmitted data. Compared with the master mode receiving process, in the slave mode, SPI_STS.CHSIDE depends on the WS signal of the external master device. When the I²S function is disabled, clear the SPI_I2SCFG.I2SEN bit to 0 when the SPI_STS.RNE flag is 1.

20.4.4 Status Flag

There are the following 4 flag bits in the SPI_STS register for monitoring the status of the I²S bus.

TX buffer empty flag (TE)

When the transmission buffer is empty, this flag is set to 1, indicating that the new data can be written into the SPI_DAT register. When the transmission buffer is not empty, this flag is cleared to 0.

RX buffer not empty flag (RNE)

When the receive buffer is not empty, this flag is set to 1, indicating that valid data has been received into the receive buffer. When reading the SPI_DAT register, this flag is set to 0.

BUSY flag (BUSY)

When the transmission starts, the BUSY flag (SPI_STS.BUSY) is set to 1. When the transmission ends, the BUSY flag (SPI_STS.BUSY) is set to 0 by hardware (software operation is invalid).

In master receiving mode (SPI_I2SCFG.MODCFG = 11), the BUSY flag (SPI_STS.BUSY) is set to 0 during receiving. When the I²S module is disabled or the transmission is completed, this flag is set to 0.

In the slave continuous communication mode, between each data item transmission, the BUSY flag (SPI_STS.BUSY) goes low in 1 I²S clock cycle. Therefore, do not use the BUSY flag (SPI_STS.BUSY) to handle the transmitting and receiving of each data item.

Channel flag (CHSIDE)

The CHSIDE (SPI_STS.CHSIDE) bit is used to indicate the channel where the data currently transmitted and received is located. Under the PCM standard, this flag has no meaning.

In transmit mode, the flag is updated when the TE flag (SPI_STS.TE) is set; in receive mode, the flag is updated when the RNE flag (SPI_STS.RNE) is set. In the process of transmitting and receiving, if an overflow (SPI_STS.OVER) or underflow (SPI_STS.UNDER) error occurs, this flag is meaningless, and the I²S needs to be disabled and then enabled again.

20.4.5 Error Flag

The SPI_STS register has 2 error flag bits.

Overflow flag (OVER)

When the RNE flag (SPI_STS.RNE) is set to 1, but data is still being transmitted to the receive buffer, an overflow error will occur. At this time, the OVER flag (SPI_STS.OVER) is set to 1. An interrupt will be generated if the user enables the corresponding interrupt. All data received after this time will be lost, and the SPI_DAT register only retains the previously unread data.

Reading the SPI_DAT register and the SPI_STS register sequentially can clear the SPI_STS.OVER bit.

Underflow flag (UNDER)

In slave transmit mode, when the first clock edge of transmitting data arrives, if the transmission buffer is still empty, the UNDER flag (SPI_STS.UNDER) is set to 1. An interrupt will be generated if the user enables the corresponding interrupt.

Reading the SPI_STS register to clear the SPI_STS.UNDER bit.

20.4.6 I²S Interrupt

The following table lists all I²S interrupts.

Table 20-3 I²S Interrupt Request

Interrupt Event	Event Flag Bit	Enable Control Bit
Send buffer empty flag	TE	TEINTEN
Receive buffer non empty flag	RNE	RNEINTEN
Underflow flag bit	UNDER	ERRINTEN

Interrupt Event	Event Flag Bit	Enable Control Bit
Overflow flag bit	OVER	

20.4.7 DMA Function

Operating in I²S mode, it does not need data transmission protection function, so it does not need to support CRC. Other DMA functions are the same as SPI mode.

20.5 SPI and I²S Register Description

20.5.1 SPI Register Overview

Table 20-4 SPI Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0													
000h	SPI_CTRL1	Reserved																BIDIRMODE	BIDIROEN	CRCEN	CRCNEXT	DATFF	RONLY	SSMEN	SSEL	LSBFF	SPIEN	BR[2:0]			MSEL	CLKPOL	CLKPHA													
	Reset Value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
004h	SPI_CTRL2	Reserved																						TEINTEN	RNEINTEN	ERRINTEN	Reserved			SSOEN	TDMAEN	RDMAEN														
	Reset Value	0																						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
008h	SPI_STS	Reserved																						BUSY	OVER	MODERR	CRCERR	UNDER	CHSIDE	TE	RNE															
	Reset Value	0																						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
00Ch	SPI_DAT	Reserved																DAT[15:0]																												
	Reset Value	0																0																												
010h	SPI_CRCPOLY	Reserved																CRCPOLY[15:0]																												
	Reset Value	0																0																												
014h	SPI_CRCRDAT	Reserved																CRCRDAT[15:0]																												
	Reset Value	0																0																												
018h	SPI_CRCTDAT	Reserved																CRCTDAT[15:0]																												
	Reset Value	0																0																												
01Ch	SPI_I2SCFG	Reserved																MODSEL	I2SEN	MODCFG [1:0]		PCMF5YNC		Reserved		STDSEL [1:0]		CLKPOL	TDATLEN [1:0]		CHBITS															
	Reset Value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								
020h	SPI_I2SPREDIV	Reserved																MCLKOEN		ODD_EVEN		LDIV[7:0]																								
	Reset Value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

20.5.2 SPI Control Register 1 (SPI_CTRL1) (Not Used In I²S Mode)

Address: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	3	2	1	0	
BIDIR MODE	BIDIR OEN	CRCEN	CRC NEXT	DATFF	RONLY	SSMEN	SSEL	LSBFF	SPIEN	BR[2:0]			MSEL	CLKPOL	CLKPHA
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw			rw	rw	rw

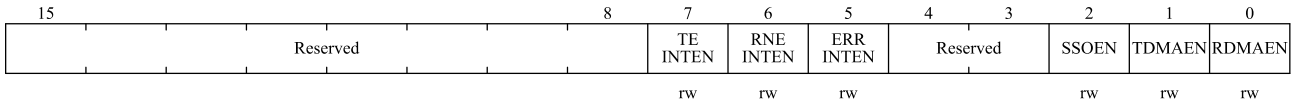
Bit Field	Name	Description
15	BIDIRMODE	<p>Bidirectional data mode enable</p> <p>0: Select the "two-wire one-way" mode.</p> <p>1: Select the "one-wire bidirectional " mode.</p> <p><i>Note: Not used in P²S mode.</i></p>
14	BIDIROEN	<p>Output enable in bidirectional mode</p> <p>0: Output disable (receive-only mode).</p> <p>1: Output enabled (send-only mode).</p> <p>In master mode, the "one-wire" data line is the MOSI pin, and in slave mode, the "one-wire" data line is the MISO pin.</p> <p><i>Note: Not used in P²S mode.</i></p>
13	CRCEN	<p>Hardware CRC check enable</p> <p>0: Disable CRC calculation.</p> <p>1: Enable CRC calculation.</p> <p><i>Note: This bit can only be written when SPI is disabled (SPI_CTRL1.SPIEN = 0), otherwise an error will occur.</i></p> <p>This bit can only be used in full duplex mode.</p> <p><i>Note: Not used in P²S mode.</i></p>
12	CRCNEXT	<p>Send CRC next</p> <p>0: The next sent value comes from the send buffer.</p> <p>1: The next send value comes from the CRC register.</p> <p><i>Note: This bit should be set immediately after the last data is written in SPI_DAT register.</i></p> <p><i>Note: Not used in P²S mode.</i></p>
11	DATFF	<p>Data frame format</p> <p>0: 8-bit data frame format is used for sending/receiving.</p> <p>1: 16-bit data frame format is used for sending/receiving.</p> <p><i>Note: This bit can only be written when SPI is disabled (SPI_CTRL1.SPIEN = 0), otherwise an error will occur.</i></p> <p><i>Note: Not used in P²S mode.</i></p>
10	RONLY	<p>Only receive mode</p> <p>This bit, together with the SPI_CTRL1.BIDIRMODE bit, determines the transfer direction in two-wire one-way mode. In the application scenario of multiple slave devices, this bit is only set to 1 by the accessed slave device, and only the accessed slave device can output, so as to avoid data line conflicts.</p> <p>0: Full duplex (sending mode and receiving mode).</p> <p>1: Disable output (receive-only mode).</p> <p><i>Note: Not used in P²S mode.</i></p>
9	SSMEN	<p>Software slave device management</p> <p>When the SPI_CTRL1.SSMEN bit is set to 1, the NSS pin level is determined by the value of the SPI_CTRL1.SSEL bit.</p> <p>0: Disable software slave device management.</p> <p>1: Enable software slave device management.</p> <p><i>Note: Not used in P²S mode.</i></p>

Bit Field	Name	Description
8	SSEL	Internal slave device selection This bit only has meaning when the SPI_CTRL1.SSMEN bit is set. It determines the NSS level, and I/O operations on the NSS pin have no effect. <i>Note: Not used in P²S mode.</i>
7	LSBFF	Frame format 0: Transmit MSB first. 1: Transmit LSB first. <i>Note: This bit cannot be changed during communication.</i> <i>Note: Not used in P²S mode.</i>
6	SPIEN	SPI enable 0: Disable SPI device. 1: Enable the SPI device. <i>Note: Not used in P²S mode.</i> <i>Note: When turning off the SPI device, please follow paragraph 0 section's procedure operation.</i>
5:3	BR[2:0]	Baud rate control 000: fPCLK/2 001: fPCLK/4 010: fPCLK/8 011: fPCLK/16 100: fPCLK/32 101: fPCLK/64 110: fPCLK/128 111: fPCLK/256 <i>Note: This bit cannot be changed during communication.</i> <i>Note: Not used in P²S mode.</i>
2	MSEL	Master device selection 0: Configure as the slave device. 1: Configure as the master device. <i>Note: This bit cannot be changed during communication.</i> <i>Note: Not used in P²S mode.</i>
1	CLKPOL	Clock polarity 0: In idle state, SCLK remains low. 1: In idle state, SCLK remains high. <i>Note: This bit cannot be changed during communication.</i> <i>Note: Not used in P²S mode.</i>
0	CLKPHA	Clock phase 0: Data sampling starts from the first clock edge. 1: Data sampling starts at the second clock edge. <i>Note: This bit cannot be modified while communication is in progress.</i> <i>Note: Not used in P²S mode.</i>

20.5.3 SPI Control Register 2 (SPI_CTRL2)

Address: 0x04

Reset value: 0x0000

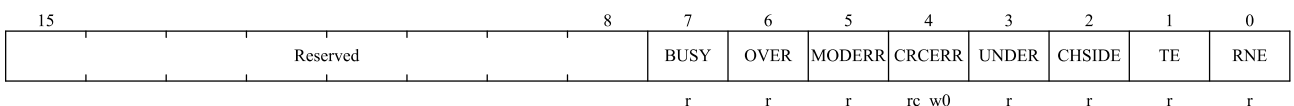


Bit Field	Name	Description
15:8	Reserved	Reserved, the reset value must be maintained
7	TEINTEN	Send buffer empty interrupt enable 0: Disable TE interrupt. 1: Enable TE interrupt, and interrupt request is generated when TE flag (SPI_STS.TE) is set to '1'.
6	RNEINTEN	Receive buffer non-empty interrupt enable 0: Disable RNE interrupt. 1: Enable RNE interrupt, and trigger interrupt request when RNE flag (SPI_STS.RNE) is set to '1'.
5	ERRINTEN	Error interrupt enable When an error (SPI_STS.CRCERR, SPI_STS.OVER, SPI_STS.UNDER, SPI_STS.MODERR) is generated, this bit controls whether an interrupt is generated 0: Disable error interrupt. 1: Enable error interrupt.
4:3	Reserved	Reserved, the reset value must be maintained
2	SSOEN	NSS output enable 0: Disable NSS output in master mode, the device can work in multi-master mode. 1: When the device is turned on, enable NSS output in the master mode, the device cannot work in the multi-master device mode. <i>Note: Not used in P²S mode.</i>
1	TDMAEN	Send buffer DMA enable When this bit is set, a DMA request is issued as soon as the TE flag (SPI_STS.TE) is set 0: Disable send buffer DMA. 1: Enable send buffer DMA.
0	RDMAEN	Receive buffer DMA enable When this bit is set, a DMA request is issued as soon as the RNE flag (SPI_STS.RNE) is set 0: Disable receive buffer DMA. 1: Enable receive buffer DMA.

20.5.4 SPI Status Register (SPI_STS)

Address: 0x08

Reset value: 0x0002

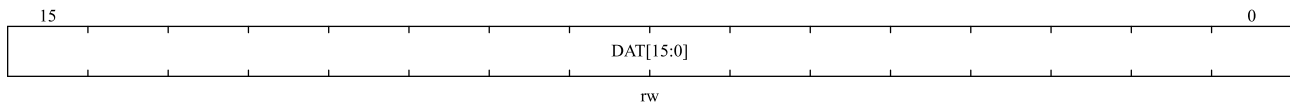


Bit Field	Name	Description
15:8	Reserved	Reserved, the reset value must be maintained
7	BUSY	Busy flag 0: SPI is not busy. 1: SPI is busy communicating or the send buffer is not empty. This bit is set or reset by hardware. <i>Note: Use of this flag requires special attention, refer to Section 20.3.3 and Section 20.3.4 for details..</i>
6	OVER	Overflow flag 0: No overflow error. 1: An overflow error occurred. <i>Note: This bit is set by hardware and cleared according to the sequence of software operations. For more information about software sequences, refer to Section 20.4.5 for details.</i>
5	MODERR	Mode error 0: No mode error. 1: A mode error occurred. <i>Note: This bit is set by hardware and cleared according to the sequence of software operations. For more information about software sequences, refer to Section 20.3.7 for details. Note: Not used in P²S mode.</i>
4	CRCERR	CRC error flag 0: The received CRC value matches the value the SPI_CRCRDAT register value. 1: The received CRC value does not match the SPI_CRCRDAT register value. <i>Note: this bit is set by hardware and cleared by software by writing 0. Note: Not used in P²S mode.</i>
3	UNDER	Underflow flag 0: No underflow occurred. 1: Underflow occurred. <i>Note: This bit is set by hardware and cleared according to the sequence of software operations. For more information about software sequences, refer to 20.4.5 for details. Note: not used in SPI mode.</i>
2	CHSIDE	Channel 0: The left channel needs to be sent or received; 1: The right channel needs to be sent or received. <i>Note: not used in SPI mode. No meaning in PCM mode.</i>
1	TE	The send buffer is empty 0: The send buffer is not empty. 1: The send buffer is empty.
0	RNE	Receive buffer is not empty 0: The receive buffer is empty. 1: The receive buffer is not empty.

20.5.5 SPI Data Register (SPI_DAT)

Address: 0x0C

Reset value: 0x0000

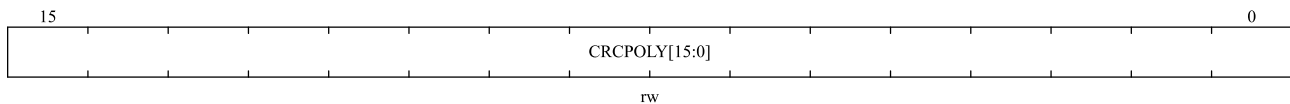


Bit Field	Name	Description
15:0	DAT[15:0]	<p>Data register</p> <p>Data to be sent or received</p> <p>The data register corresponds to two buffers: one for write (send buffer); The other is for read (receive buffer). Write operation writes data to send buffer; The read operation will return the data in the receive buffer.</p> <p>Note on SPI mode: According to the selection of the data frame format by the SPI_CTRL1.DATFF bit, the data sending and receiving can be 8-bit or 16-bit. To ensure correct operation, the data frame format needs to be determined before enabling the SPI.</p> <p>For 8-bit data, the buffer is 8-bit, and only SPI_DAT[7:0] is used when sending and receiving. When receiving, SPI_DAT[15:8] is forced to 0.</p> <p>For 16-bit data, the buffer is 16-bit, and the entire data register is used when sending and receiving, that is, SPI_DAT[15:0].</p>

20.5.6 SPI CRC Polynomial Register (SPI_CRCPOLY) (Not Used In I²S Mode)

Address: 0x10

Reset value: 0x0007

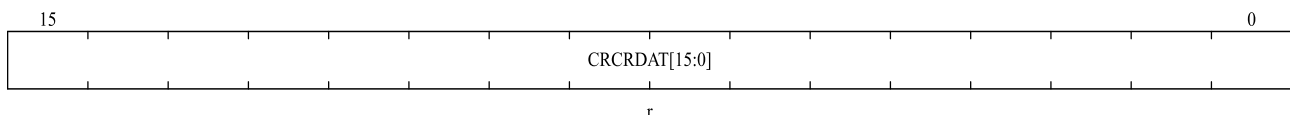


Bit Field	Name	Description
15:0	CRCPOLY [15:0]	<p>CRC polynomial register</p> <p>This register contains the polynomial used for the CRC calculation.</p> <p>The reset value is 0x0007, other values can be set according to the application.</p> <p><i>Note: not used in I²S mode.</i></p>

20.5.7 SPI RX CRC Register (SPI_CRCRDAT) (Not Used In I²S Mode)

Address offset: 0x14

Reset value: 0x0000



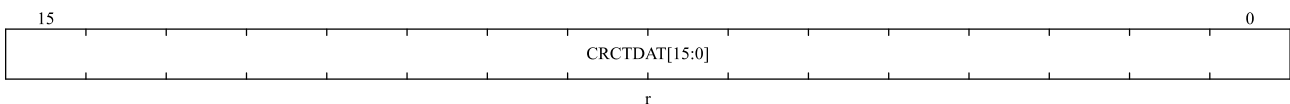
Bit Field	Name	Description
-----------	------	-------------

15:0	CRCRDAT	<p>Receive CRC register</p> <p>When CRC calculation is enabled, CRCRDAT[15:0] will contain the calculated CRC value of subsequent received bytes. This register is reset when '1' is written to the SPI_CTRL1.CRCEN bit. The CRC calculation uses the polynomial in SPI_CRCPOLY.</p> <p>When the data frame format is set to 8 bits, only the lower 8 bits participate in the calculation and follow the CRC8 standard; when the data frame format is 16 bits, all 16 bits in the register participate in the calculation and follow the CRC16 standard.</p> <p><i>Note: reading this register when the BUSY flag (SPI_STS.BUSY) is '1' may read incorrect values.</i></p> <p><i>Note: not used in I²S mode.</i></p>
------	---------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

20.5.8 SPI TX CRC Register (SPI_CRCTDAT)

Address offset: 0x18

Reset value: 0x0000

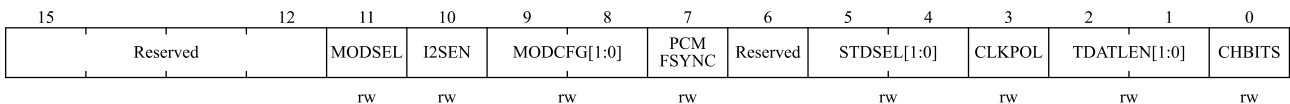


Bit Field	Name	Description
15:0	CRCTDAT	<p>Send CRC register</p> <p>When CRC calculation is enabled, CRCTDAT[15:0] contains the CRC value calculated by the bytes sent subsequently. This register is reset when '1' is written to the SPI_CTRL1.CRCEN bit. The CRC calculation uses the polynomial in SPI_CRCPOLY.</p> <p>When the data frame format is set to 8 bits, only the lower 8 bits participate in the calculation and follow the CRC8 standard; when the data frame format is 16 bits, all 16 bits in the register participate in the calculation and follow the CRC16 standard.</p> <p><i>Note: reading this register when the BUSY flag (SPI_STS.BUSY) is '1' may read incorrect values.</i></p> <p><i>Note: not used in I²S mode.</i></p>

20.5.9 SPI_I²S Configuration Register (SPI_I2SCFG)

Address offset: 0x1c

Reset value: 0x0000



Bit Field	Name	Description
15:12	Reserved	Reserved, the reset value must be maintained
11	MODSEL	<p>I²S mode selection</p> <p>0: Select SPI mode.</p> <p>1: Select I²S mode.</p> <p><i>Note: this bit can only be set when SPI or I²S is turned off.</i></p>

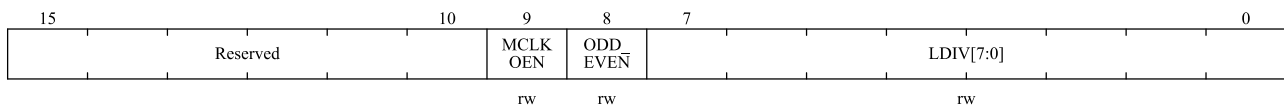
Bit Field	Name	Description
10	I ² SEN	I ² S enable 0: Disable I ² S. 1: Enable I ² S. <i>Note: not used in SPI mode.</i>
9:8	MODCFG	I ² S mode setting 00: Slave device sends. 01: Slave device receives. 10: Master device sends. 11: Master device receives. <i>Note: This bit can only be set when I²S is turned off.</i> <i>Note: not used in SPI mode.</i>
7	PCMFSYNC	PCM frame synchronization 0: Short frame synchronization. 1: Long frame synchronization. <i>Note: This bit is only meaningful when SPI_I2SCFG.STDSEL = 11 (used by the PCM standard).</i> <i>Note: not used in SPI mode.</i>
6	Reserved	Reserved, the reset value must be maintained
5:4	STDSEL	Selection of I ² S standard 00: I ² S Philips standard. 01: High byte alignment standard (left alignment). 10: Low byte alignment standard (right alignment). 11: PCM standard. See for details of I ² S standard on section 20.4.1. <i>Note: For correct operation, this bit can only be set when I²S is turned off.</i> <i>Note: not used in SPI mode.</i>
3	CLKPOL	Static clock polarity 0: I ² S clock static state is low level. 1: I ² S clock static state is high level. <i>Note: For correct operation, this bit can only be set when I²S is turned off.</i> <i>Note: not used in SPI mode.</i>
2:1	TDATLEN	Length of data to be transmitted 00: 16-bit data length. 01: 24-bit data length; 10: 32-bit data length; 11: Not allowed. <i>Note: For correct operation, this bit can only be set when I²S is turned off.</i> <i>Note: not used in SPI mode.</i>
0	CHBITS	Channel length (number of data bits per audio channel) 0: 16 bits wide; 1: 32 bits wide. Writing to this bit is meaningful only when SPI_I2SCFG.TDATLEN = 00, otherwise the channel length is fixed to 32 bits by hardware. <i>Note: For correct operation, this bit can only be set when I²S is turned off.</i>

Bit Field	Name	Description
		<i>Note: not used in SPI mode.</i>

20.5.10 SPI_I²S Prescaler Register (SPI_I2SPREDIV)

Address: 0x20

Reset value: 0x0002



Bit Field	Name	Description
15:10	Reserved	Reserved, the reset value must be maintained
9	MCLK_OEN	Master clock output enable 0: Disable master clock output. 1: Enable master clock output. <i>Note: For correct operation, this bit can only be set when I²S is turned off.</i> <i>Note: not used in SPI mode.</i>
8	ODD_EVEN	Odd coefficient prescaler 0: Actual prescaler factor = LDIV × 2. 1: Actual prescaler factor = (LDIV × 2) + 1. See section 20.4.2 for details. <i>Note: For correct operation, this bit can only be set when I²S is turned off.</i> <i>Note: not used in SPI mode.</i>
7:0	LDIV	I ² S linear prescaler Disable setting LDIV [7:0] = 0 or LDIV [7:0] = 1 See Section 20.4.2 for details. <i>Note: For correct operation, this bit can only be set when I²S is turned off.</i> <i>Note: not used in SPI mode.</i>

21 Real Time Clock (RTC)

21.1 Description

- The real-time clock (RTC) is an independent BCD timer/counter.
- The software supports daylight saving time compensation.
- Programmable periodic automatic wakeup timer.
- Two programmable alarms clocks.
- Two 32-bit registers contain programmable alarms clocks, seconds, minutes, hours, day (day of week), date (day of month), month, and year.
- Two Independent 32-bit register contain programming alarm, sub-seconds value. Digital precision calibration function.
- Reference clock detection: a more precise external source clock (50 or 60 Hz) can be used to improve the calendar precision.
- 2 tamper detection events with configurable filter and internal pull-up.
- Timestamp function.
- Multiple wakeup sources of Interrupt/Event, including alarm A, alarm B, wakeup timer, timestamp, Tamper.
- Automatically perform monthly compensation for 28, 29 (leap year), 30 and 31 day
- After RTC is enabled by the RCC register and voltage remains in the operating range, RTC will not stop timing in RUN mode, LPRUN mode, SLEEP mode and STOP mode.
- Support to wake up MCU from low power consumption mode (SLEEP mode and STOP mode).

21.2 Specification

Table 21-1 RTC Feature Support

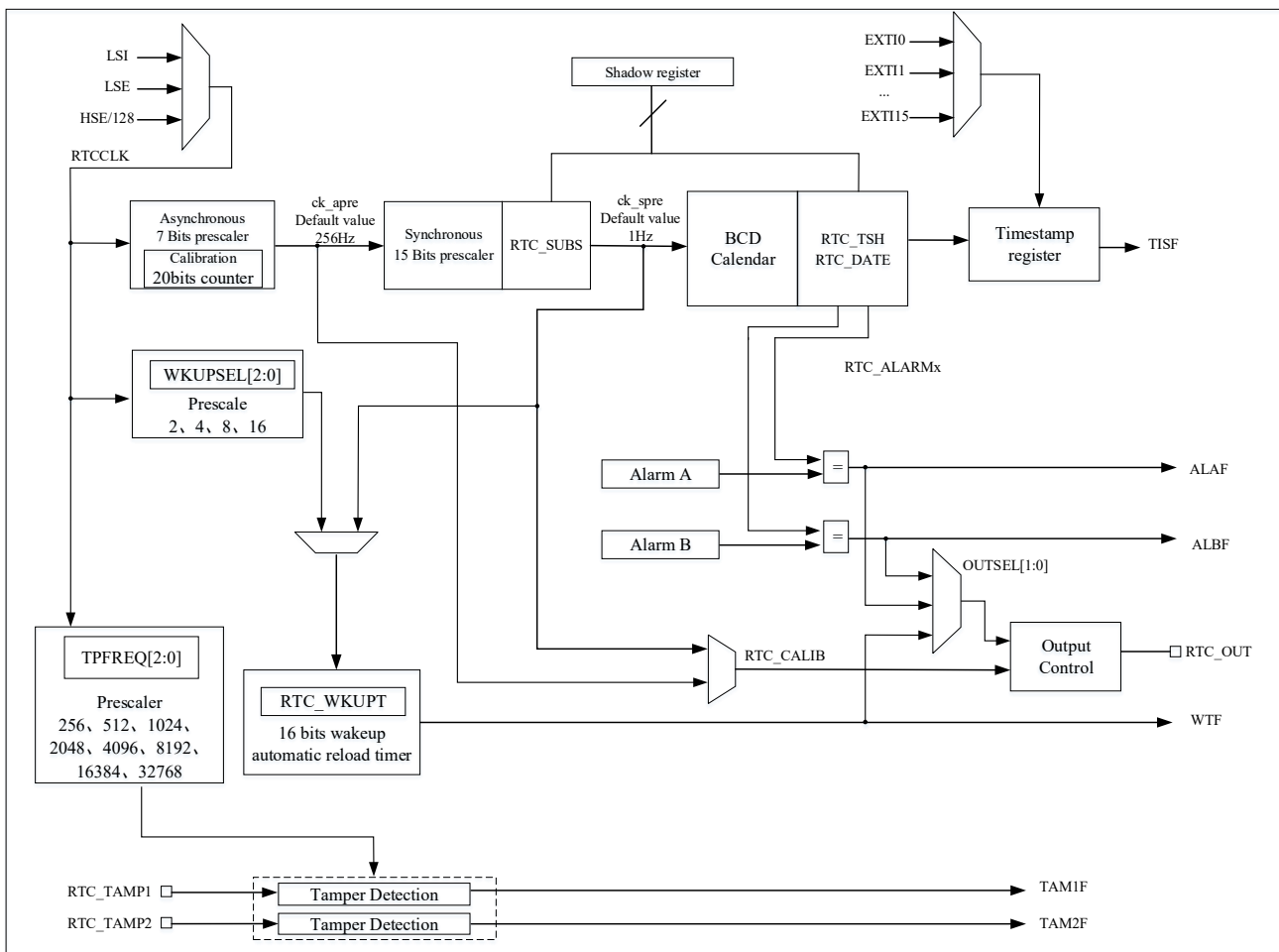
Main Function	Description
Clock	RTC clock can be selected from LSI, LSE and HSE, which are 30KHz, 32.768KHz and HSE / 128 respectively
Calendar	Calendar consists of sub second, second, minute, hour (12 or 24 format), day (day of the week), date, month and year. These data are stored in the shadow register of APB module.
Wakeup Timer	Output “RTC_OUT” can be configured to send wakeup events to GPIO. At the same time, it also can be configured as an interrupt/event to wake up the system from SLEEP, STOP modes.
Alarm	Programmable alarm clock and interrupt function. The alarm can be triggered by any combination of the calendar fields. When the alarm event occurs the alarm flag can be sent to GPIO through “RTC_OUT”, and it also can be used to wake up the CPU or exit from the low power status such as SLEEP, STOP modes.
Tamper	2 Tamper detection logic are a source of system wakeup. If a tamper event happen on one of the input lines. It is also a source of hardware trigger to LP Timer.

Main Function	Description
Timestamp	Time-stamp function for GPIO event saving. It is a source to wakeup system from low power modes. Alternatively a tamper event could be a source of timestamp event.
Interrupts/events	Alarm A/Alarm B interrupt Wakeup interrupt Timestamp interrupt Tamper interrupt

21.3 RTC Function Description

21.3.1 RTC Block Diagram

Figure 21-1 RTC Block Diagram



RTC includes the following functions:

- Alarm A and Alarm B event/interrupt
- Timestamp event/interrupt
- Tamper event/interrupt

- RTC output functions
 - 256Hz or 1Hz clock output (LSE frequency is 32.768 kHz).
 - Alarm clock output (polarity configurable), Alarm A and Alarm B are optional.
 - Auto wakeup output (polarity configurable).
- RTC input functions:
 - Timestamp event detection
 - 50 or 60Hz reference clock input
 - Tamper event detection

21.3.2 GPIOs of RTC

Timestamp input come from IOM (mapped to PC13) or EXTI module, if it is the EXTI module, please refer to the timestamp trigger source selection register (EXTI_TS_SEL) for details.

RTC_OUT (Alarm, Wakeup event or calibration output (256Hz or 1Hz)) is mapped to PC13. Regardless of the PC13 GPIO configuration, the PC13 pin configuration is controlled by the RTC as an output.

PC13 can be used as RTC tamper 1 detection pin, PA0 can be used as RTC tamper 2 detection pin, Controlled by the RTC as a pull-up input.

PA10 or PB15 can be used as RTC_REFCLKIN reference clock input pin.

21.3.3 RTC Register Write Protection

After power-up or reset, all RTC registers except RTC_CTRL, RTC_TMPCFG, RTC_INITSTS[13:8] are write-protected. Writes to the RTC registers are enabled by writing a key to the write protection register RTC_WRP. All write protection RTC registers require the following steps to unlock write protection:

1. Write “0xCA” into RTC_WRP register.
2. Write “0x53” into RTC_WRP register.

After unlocking these registers, it cannot be write protected unless the RTC is software reset or power up again. The unlocking mechanism only checks the write operation to the RTC_WRP register. During, before or after the unlocking process, the write operation to other registers does not affect the unlocking result.

21.3.4 RTC Clock and Prescaler

RTC clock source:

- LSE clock
- LSI clock
- HSE/128 clock

To minimize power consumption, the prescaler is divided into 2 programmable prescalers, they are asynchronous prescaler and synchronous prescaler. If both prescalers are used, it is recommended that the value of the asynchronous perscaler be as large as possible.

- A 7-bit asynchronous prescaler configured by RTC_PRE.DIVA[6:0] bits

- A 15-bit synchronous prescaler configured by RTC_PRE.DIVS[14:0] bits

The formula for f_{ck_apre} and f_{ck_spre} are given below:

$$f_{ck_apre} = \frac{f_{RTCCLK}}{RTC_PRE.DIVA[6:0]+1}$$

$$f_{ck_spre} = \frac{f_{RTCCLK}}{(RTC_PRE.DIVS[14:0]+1)*(RTC_PRE.DIVA[6:0]+1)}$$

The ck_apre clock is used to driven sub-second (RTC_SUBS) down counter. When it reaches 0, RTC_SUBS is reloaded with the value of RTC_PRE.DIVS[14:0].

21.3.5 RTC Calendar

There are three shadow registers, they are RTC_DATE, RTC_TSH and RTC_SUBS. The RTC time and date registers can be accessed through the shadow registers. It is also possible to access them directly to avoid waiting for the synchronization duration. The three shadow registers are as follow:

- RTC_DATE: set and read date
- RTC_TSH: set and read time
- RTC_SUBS: read sub-second

After every two RTCCLK cycles, the current calendar value is updated to the shadow register, and RTC_INITSTS.RSYF bit is set to 1. This process is not performed in low power (STOP) modes. While exiting these modes, the shadow register updates the values after 2 RTCCLK cycles.

By default, when user attempts to access the calendar register, it will accesses the contents of the shadow register. User can access the calendar register directly by setting the RTC_CTRL.BYPS bit.

When RTC_CTRL.BYPS=0, calendar values are from shadow registers. When reading RTC_SUBS, RTC_TSH or RTC_DATE register, it is necessary to make ensure the frequency of APB1 clock (f_{APB1}) is at least 7 times the frequency of RTC clock (f_{RTCCLK}), and APB1 clock frequency lower than RTC clock frequency is not allowed in any case. System reset will reset shadow registers.

Note: If the sub-second matching interrupt is configured, the first sub-second matching interrupt may not be generated, and the subsequent sub-second matching interrupts are normal, and the first sub-second matching interrupt can be ignored.

21.3.6 Calendar Initialization and Configuration

The value of prescaler and calendar can be initialized by the following steps:

- Enter initialization mode by setting “1” to RTC_INITSTS.INITM bit, then wait for RTC_INITSTS.INITF flag to be set 1.
- Set RTC_PRE.DIVS[14:0] and RTC_PRE.DIVA[6:0] value.
- Write the initial calendar values include time and date into the shadow registers (RTC_TSH and RTC_DATE) and configure the time format (12 or 24 hours) by the RTC_CTRL.HFMT bit.
- Exit initialization mode by clearing the RTC_INITSTS.INITM bit.

The values of calendar counter will automatically loaded from shadow registers after 4 RTCCLK clock cycles, then the calendar counter restarts.

Notes:

(1) The minimum time before re-entering the initialization mode above should be 1 second apart, as the internal registers use 1Hz clock to update.

(2) Before entering the RTC initialization mode, ensure that the value of `RTC_SUBS.SS[15:0]` is not less than 2, and read `RTC_DATE` register once.

21.3.7 Calendar Reading

Reading calendar value when `RTC_CTRL.BYPS=0`

Calendar value is read from shadow registers if `RTC_CTRL.BYPS=0`. In order to read RTC calendar registers (`RTC_SUBS`, `RTC_TSH` and `RTC_DATE`) correctly, APB1 clock frequency must be set equal to or greater than 7 times of RTC clock frequency. In any case, APB1 clock frequency must not be less than RTC clock frequency.

If APB1 clock frequency is not equal to or greater than 7 times of RTC clock frequency, refer to the following process to read calendar value.

1. Read the data of `RTC_SUBS`, `RTC_TSH` and `RTC_DATE` twice.
2. Compare the data read twice, if they are equal, the read data can be considered correct; if they are not equal, read the data for the third time.
3. The third time read data can be considered correctly.

The shadow registers (`RTC_SUBS`, `RTC_TSH` and `RTC_DATE`) are updated every two `RTCCLK` cycles. If user want to read calendar value in a short time (less than two `RTCCLK` cycles), `RTC_INITSTS.RSYF` bit must be cleared by software after the first time read.

In some cases, it is necessary to wait until `RTC_INITSTS.RSYF` bit is set to 1 before reading calendar value.

- After waking up from the low power modes (STOP mode), clear `RTC_INITSTS.RSYF` bit, then wait `RTC_INITSTS.RSYF` bit is set again.
- System reset.
- Calendar complete initialization.
- Calendar complete synchronization.

Reading calendar value when `RTC_CTRL.BYPS=1`

Reading the calendar value directly from the calendar counter if `RTC_CTRL.BYPS=1`. The advantage of this configuration is that read calendar value without delay after wakeup from the low power mode, the disadvantage is that these data of `RTC_SUBS`, `RTC_TSH` and `RTC_DATE` may not be from the same time.

To ensure the correctness of read calendar value, it is necessary to read `RTC_SUBS`, `RTC_TSH` and `RTC_DATE` twice, then compare the data read twice, if they are equal, the read data can be considered correct.

Note: After read `RTC_TSH` or `RTC_SUBS` register, it needs to read `RTC_DATE` register once.

21.3.8 Calibration Clock Output

When `RTC_CTRL.COEN` set to 1, PC13 pin will output calibration clock. If `RTC_CTRL.CALOSEL=0` and `RTC_PRE.DIVA[6:0] = 0x7F`, the `RTC_CALIB` frequency is $f_{RTCCLK}/RTC_PRE.DIVA[6:0]$. This is equivalent to a

calibration output of 256 Hz when the RTCCLK frequency is 32.768 kHz. The rising edge is recommended due to slight jitter on the falling edge.

When `RTC_CTRL.CALOSEL=1` and "`RTC_PRE.DIVS[14:0]+1`" is a non-zero integer multiple of 256, the `RTC_CALIB` frequency is given by the formula $f_{\text{RTCCLK}}/(256 * (\text{DIVA}+1))$. This is equivalent to 1Hz calibration output when the RTCCLK frequency is 32.768 kHz and `RTC_PRE.DIVA[6:0] = 0x7F`.

Note: When the `RTC_CALIB` or `RTC_ALARM` output is selected, the `RTC_OUT` pin (PC13) is automatically configured as output.

21.3.9 Programmable Alarms

RTC has 2 programmable alarms: Alarm A and Alarm B.

RTC alarm can be enabled or disabled by `RTC_CTRL.ALxEN` bit. If all the alarm value match the calendar values, the `RTC_INITSTS.ALxF` flag will be set. Each calendar field can be selected to trigger alarm interrupt if `RTC_CTRL.ALxIEN` bit is enabled.

Alarm output: Alarm A and Alarm B can be mapped to `RTC_ALxRM` output when `RTC_CTRL.OUTSEL[1:0]` is selected, and output polarity can be configured by `RTC_CTRL.OPOL` bit.

Note: If the second field is selected (`RTC_ALARMx.MASK1` bit reset), `RTC_PRE.DIVS[14:0]` must be larger than 3 to ensure correct operation.

21.3.10 Alarm Configuration

Alarm A and Alarm B should be configured in the following below:

1. Disable Alarm A/Alarm B by clearing `RTC_CTRL.ALAEN/RTC_CTRL.ALBEN` bit.
2. Configure the Alarm x registers (`RTC_ALRMxSS/RTC_ALARMx`)
3. Enable Alarm A/Alarm B interrupt by set `RTC_CTRL.ALAEN/RTC_CTRL.ALBEN` bit (this step can be selected as needed)
4. Enable Alarm A/Alarm B by setting `RTC_CTRL.ALAEN/RTC_CTRL.ALBEN` bit.

21.3.11 Alarm Output

When `RTC_CTRL.OUTSEL[1:0] != 0`, `RTC_ALARM` alternate function output is enable. There are Alarm A output, Alarm B output and Wakeup output to choose by the value of `RTC_CTRL.OUTSEL[1:0]` bits.

`RTC_CTRL.OPOL` bit control the polarity of the Alarm A, Alarm B or Wakeup output.

When `RTC_CALIB` or `RTC_ALARM` output is selected, the `RTC_OUT` pin (PC13) is automatically configured as output.

21.3.12 Periodic Automatic Wakeup

A 16-bit programmable auto-load down counter can generate periodic wakeup flag. It is also can be extend the range of wakeup timer to 17 bits. Periodic automatic wakeup can be enabled by setting `RTC_CTRL.WTEN`.

There are two wake-up input clock sources can be selected:

- RTC clock (RTCCLK) divided by 2/ 4/8/16.

Assume RTCCLK comes from LSE (32.768KHz), wake-up interrupt period can be configured range from 122us to

32s under the resolution down to 61us.

- Internal clock ck_spre.

Assume ck_spre frequency is 1Hz, the available wake-up time range from 2s to 36h, and the resolution is 1 second.

- When RTC_CTRL.WKUPSEL [2:0] = 10x, the period is range from 2s to 18h.
- When RTC_CTRL.WKUPSEL [2:0] = 11x, the period is range from 18h to 36h.

After RTC_CTRL.WTEN bit is set to 1, the down counter is operating and when it reaches 0, RTC_INITSTS.WTF will be set and the device can exit from low power modes when the periodic wakeup interrupt is enabled by setting the RTC_CTRL.WTIEN bit.

Periodic wakeup output: periodic wakeup can be mapped to RTC_ALxRM output when RTC_CTRL.OUTSEL[1:0] is selected, the RTC_OUT pin(PC13) is automatically configured as output, and output polarity can be configured by RTC_CTRL.OPOL bit.

21.3.13 Wakeup Timer Configuration

The wakeup timer auto-reload value should be configured in the following below:

1. Disable wakeup timer by clearing RTC_CTRL.WTEN bit, then wait for RTC_INITSTS.WTWF flag to be set 1.
2. Select wakeup timer clock by set RTC_CTRL.WKUPSEL[2:0] bits.
3. Configure the wakeup automatic reload value by set RTC_WKUPT.WKUPT[15:0] bits.
4. Enable Wakeup interrupt by set RTC_CTRL.WTIEN bit(this step can be selected as needed)
5. Enable wakeup timer by setting RTC_CTRL.WTEN bit

21.3.14 Timestamp Function

Timestamp can be enabled by setting RTC_CTRL.TSEN bit to 1. When a timestamp event is detected on the RTC_TS pin, the calendar values of the event will be stored in the timestamp register (RTC_TSSS, RTC_TST, RTC_TSD), and RTC_INITSTS.TISF is set to 1. Timestamp event can generate an interrupt if RTC_CTRL.TSIEN is set to 1. If a new timestamp event is detected when RTC_INITSTS.TISF has been set to 1 already, the hardware sets RTC_INITSTS.TISOVF flag to 1, and the timestamp registers (RTC_TST and RTC_TSD) will continue to hold the value of the previous event, which means timestamp registers(RTC_TST and RTC_TSD) data will not change when RTC_INITSTS.TISF=1.

After the timestamp event caused by the synchronization process occurs again, RTC_INITSTS.TISF is set to 1 in 2 RTC_CLK cycles. There is no delay in the generation of RTC_INITSTS.TISOVF. This means that if two timestamp events are very close, this can cause RTC_INITSTS.TISOVF to be "1" and RTC_INITSTS.TISF to be "0". Therefore, after detecting that RTC_INITSTS.TISF is "1", then detect RTC_INITSTS.TISOVF bit.

Tamper event can trigger timestamp event when RTC_TMPCFG.TPTS bit is set to 1.

If timestamp events are enabled, the timestamp will capture the calendar read in the timestamp register. When both tamper events and timestamp events are enabled, tamper events can also result in timestamp capture. Timestamp events can be generated on any of the 16 GPIO ports selected by EXTI. The GPIO pins in each port are selected by setting the corresponding EXTI_TS_SEL.TSSEL[3:0] bits.

21.3.15 Tamper Detection

There are 2 tamper detection pin, RTC_TAMP1 pin is PC13, RTC_TAMP2 pin is PA0. RTC_TAMPx pin can be used as tamper event detection function input pin. There are two detection modes, edge detection mode and level detection mode with configurable filtering function.

Tamper detection initialization

There are 2 tamper detection pins, each of them can be configured independently. User need to configure tamper detection before enable RTC_TMPCFG.TPxEN bit. When the tamper event is detected after tamper detection is enable, tamper event can generate an interrupt and RTC_INITSTS.TAMxF bit will be set 1 if RTC_TMPCFG.TPxINTEN is set to 1.

When RTC_INITSTS.TAMxF is set to 1, a new tamper event on the same pin cannot be detected.

Timestamp on tamper event

Any tamper event can cause a timestamp event when RTC_INITSTS.TPTS is set to 1. The RTC_INITSTS.TISF bit and RTC_INITSTS.TISOVF bit will be set as a normal timestamp event.

Edge detection of tamper input

When RTC_TMPCFG.TPFLT[1:0] bits set to 0, tamper detection is set to edge detection, and one of rising edge or falling edge is controlled by RTC_TMPCFG.TPxTRG bit. The RTC_TAMPx pin will generate a tamper detection event when corresponding edge is detected.

When edge detection is used, to ensure that a valid edge is generated after enabling the tamper event detection, it is recommended to check the tamper pin level by software immediately after enabling it; when RTC_TMPCFG.TPFLT[1:0] = 0 and RTC_TMPCFG.TPxTRG = 0, if the tamper detection is enabled and the tamper input is high before detection, the tamper event can be detected by hardware.

Filtered level detection of RTC_TAMPx input

When RTC_TMPCFG.TPFLT[1:0] bits set to 1/2/3, tamper detection is set to level detection. The value of RTC_TMPCFG.TPFLT[1:0] determines the number of samples.

Precharging can be done before each sampling by using the internal pull-up resistor of tamper pin. The precharge time is controlled by RTC_TMPCFG.TPPRCH[1:0] bits. Precharge will be disabled when RTC_TMPCFG.TPPUDIS set 1.

Using RTC_TMPCFG.TPFREQ[2:0] to determine the sampling frequency of level detection can optimize the best balance between tamper detection delay and pull-up power consumption.

21.3.16 Daylight Saving Time Configuration

Daylight saving time function can be controlled by RTC_CTRL.SU1H, RTC_CTRL.AD1H, and RTC_CTRL.BAKP bits. Calendar will subtract one hour when set RTC_CTRL.SU1H bit to 1, and add one hour when set RTC_CTRL.AD1H to 1. RTC_CTRL.BAKP can be used to record whether this operation was performed.

21.3.17 RTC Sub-Second Register Shift

When the value of calendar has a sub-second deviation compared to the external precision clock, the shift function can be used to improve the precision of calendar.

Calendar can use RTC_SCTRL.AD1S and RTC_SCTRL.SUBF[14:0] bits to control maximum delay or advance 1s. The resolution of the adjustment is $1/(RTC_PRE.DIVS[14:0]+1)$ second, it means the higher value of RTC_PRE.DIVS[14:0], the higher of the resolution. However, to keep the synchronous prescaler output at 1Hz, the higher RTC_PRE.DIVS[14:0]

means the lower RTC_PRE.DIVA[6:0], then more power consuming.

Note: Before starting a shift operation, user must check if RTC_SUBS.SS[15] bit is 0.

Whenever write RTC_SCTRL register, the RTC_INITSTS.SHOPF flag will be set by hardware, which indicate a shift operation is pending. Once this shift operation is complete, the bit is cleared by hardware.

21.3.18 RTC Digital Clock Precision Calibration

Digital precision calibration is achieved by adjusting the number of RTC clock pulses within the calibration period. Digital precision calibration resolution is 0.954 PPM with the range from -487.1 PPM to +488.5 PPM.

When the input frequency is 32768 Hz, calibration period can be configured as $2^{20}/2^{19}/2^{18}$ RTCCLK cycles or 32/16/8 seconds. The precision calibration register (RTC_CALIB) indicates that there has RTC_CALIB.CM[8:0] RTCCLK clock cycles will be reduced during the specified period.

The value of RTC_CALIB.CM[8:0] represents the number of RTCCLK pulses to be reduced during specified period. While RTC_CALIB.CP can be used to increase 488.5 PPM, every 2^{11} RTCCLK cycles will inserts a RTCCLK pulse.

When using RTC_CALIB.CM[8:0] and RTC_CALIB.CP in combination, it can increase cycles range from -511 to +512 RTCCLK cycles, and the calibration range from -487.1 to +488.5 PPM with the resolution is about 0.954 PPM.

The effective calibrated frequency (f_{CAL}) can be calculated by using the formula given below:

$$f_{CAL} = f_{RTCCLK} * \left(1 + \frac{RTC_CALIB.CP * 512 - RTC_CALIB.CM[8:0]}{2^n + RTC_CALIB.CM[8:0] - RTC_CALIB.CP * 512} \right)$$

Note: n=20/19/18

Calibrated when RTC_PRE.DIVA[6:0]<3

When the asynchronous prescaler value (RTC_PRE.DIVA[6:0]) is less than 3. The RTC_CALIB.CP cannot be programmed to 1, and RTC_CALIB.CP value will be ignored if it has been set to 1.

When RTC_PRE.DIVA[6:0]<3, the value of RTC_PRE.DIVS[14:0] should be decrease. Assume RTCCLK frequency is 32768Hz:

- When RTC_PRE.DIVA[6:0] = 2, RTC_PRE.DIVS[14:0] = 8189.
- When RTC_PRE.DIVA[6:0] = 1, RTC_PRE.DIVS[14:0] = 16379.
- When RTC_PRE.DIVA[6:0] = 0, RTC_PRE.DIVS[14:0] = 32759.

The effective calibrated frequency (f_{CAL}) can be calculated by using the formula given below:

$$f_{CAL} = f_{RTCCLK} * \left(1 + \frac{256 - RTC_CALIB.CM[8:0]}{2^n + RTC_CALIB.CM[8:0] - 265} \right)$$

n=20/19/18

Verify RTC calibration

The RTC outputs 1Hz waveform for measuring and verifying RTC precision.

Up to 2 RTCCLK cycles measurement error may occur when measuring the RTC frequency in a limit measurement period. If the measurement period is the same as calibration period, the error can be eliminated.

- The calibration period is 32 seconds (default).

Using an accurate 32-second period to measure the 1Hz calibration output can ensure that the measurement error is within 0.447ppm (0.5 RTCCLK cycles within 32 seconds).

- The calibration period is 16 seconds.

Using an accurate 16-second period to measure the 1Hz calibration output can ensure that the measurement error is within 0.954ppm (0.5 RTCCLK cycles within 16 seconds).

- The calibration period is 8 seconds.

Using an accurate 8-second period to measure the 1Hz calibration output can ensure that the measurement error is within 1.907ppm (0.5 RTCCLK cycles within 8 seconds).

Dynamic recalibration

When RTC_INITSTS.INITF=0, RTC_CALIB register can update by using following steps:

1. Wait RTC_INITSTS.RECPF=0.
2. A new value is written to the RTC_CALIB, then RTC_INITSTS.RECPF is automatically set to 1.
3. The new calibration settings will take effect within 3 ck_apre cycles after a data write to the RTC_CALIB.

21.3.19 RTC Low Power Mode

The operating state of RTC in low power mode.

Lower Power Mode	RTC Working State	Exit Low Power Mode
SLEEP	Normal operate	RTC interrupt
STOP	Normal operate when the clock source of RTC is LSE or LSI	Alarm A, Alarm B, Periodic Wakeup, Tamper event and Timestamp event

21.4 RTC Registers

21.4.1 RTC Register Overview

Table 21-2 RTC Register Overview

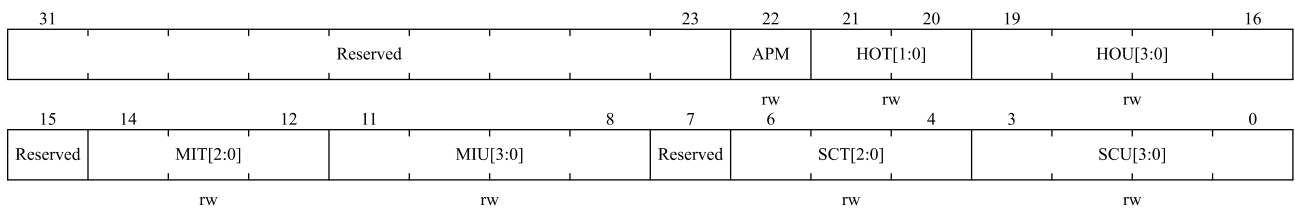
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
000h	RTC_TSH	Reserved										APM	HOT[1:0]			HOU[3:0]			Reserved	MIT[2:0]			MIU[3:0]			Reserved	SCT[2:0]			SCU[3:0]									
	Reset Value											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
004h	RTC_DATE	Reserved										YRT[3:0]				YRU[3:0]				WDU[2:0]		MOT	MOU[3:0]			Reserved	DAT[1:0]			DAU[3:0]									
	Reset Value											0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
008h	RTC_CTRL	Reserved										COEN	OUTSEL[1:0]			OPOE	CALOSEL	BAKP	SUIH	ADJH	TSIEN	WTIEN	ALBIEN	ALAIEN	TSEN	WTEN	ALBEN	ALAIEN	Reserved	HFMT	BYPS	REFCKEN	TEDGE	WKUPSEL[2:0]					
	Reset Value											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00Ch	RTC_INITSTS	Reserved										RECPF	Reserved	TAM2F	TAM1F	TISOVF	TISF	WTF	ALBF	ALAF	INITM	INITF	RSYF	INITSF	SHOPF	WTWF	ALBWF	ALAWF											
	Reset Value											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1								
010h	RTC_PRE	Reserved										DIVA[6:0]						Reserved	DIVS[14:0]																				
	Reset Value											1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																						
014h	RTC_WKUPT	Reserved																WKUPT[15:0]																																					
	Reset Value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1						
01Ch	RTC_ALARMA	MASK4	WKDSEL	DTT[1:0]			DTU[3:0]			MASK3	APM	HOT[1:0]			HOU[3:0]			MASK2	MIT[2:0]		MIU[3:0]			MASK1	SET[2:0]		SEU[3:0]																												
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																				
020h	RTC_ALARMB	MASK4	WKDSEL	DTT[1:0]			DTU[3:0]			MASK3	APM	HOT[1:0]			HOU[3:0]			MASK2	MIT[2:0]		MIU[3:0]			MASK1	SET[2:0]		SEU[3:0]																												
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																					
024h	RTC_WRP	Reserved																								PKEY[7:0]																													
	Reset Value																									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
028h	RTC_SUBS	Reserved																SS[15:0]																																					
	Reset Value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
02Ch	RTC_SCTRL	ADIS	Reserved																SUBF[14:0]																																				
	Reset Value	0																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
030h	RTC_TST	Reserved										APM	HOT[1:0]			HOU[3:0]			Reserved	MIT[2:0]		MIU[3:0]			Reserved	SET[2:0]		SEU[3:0]																											
	Reset Value											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
034h	RTC_TSD	Reserved										YRT[3:0]			YRU[3:0]			WDU[2:0]		MOT	MOU[3:0]			Reserved	DAT[1:0]		DAU[3:0]																												
	Reset Value											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
038h	RTC_TSSS	Reserved																SSE[15:0]																																					
	Reset Value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
03Ch	RTC_CALIB	Reserved																CP	CW8	CW16	Reserved				CM[8:0]																														
	Reset Value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
040h	RTC_TMPCFG	Reserved										TP2MF	TP2NOE	TP2INTEN	TP1MF	TP1NOE	TP1INTEN	TPPUDIS	TPPRCH[1:0]	TPPEL[1:0]	TPFREQ[2:0]			TPPTS	Reserved	TP2TRG	TP2EN	TP1INTEN	TP1TRG	TP1EN																									
	Reset Value											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
044h	RTC_ALRMAS	Reserved			MASKSSA[3:0]			Reserved										SSV[14:0]																																					
	Reset Value				0	0	0	0											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0													
048h	RTC_ALRMBSS	Reserved			MASKSSB[3:0]			Reserved										SSV[14:0]																																					
	Reset Value				0	0	0	0											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0											

21.4.2 RTC Calendar Time Register (RTC_TSH)

Address offset: 0x00

Reset value: 0x0000 0000



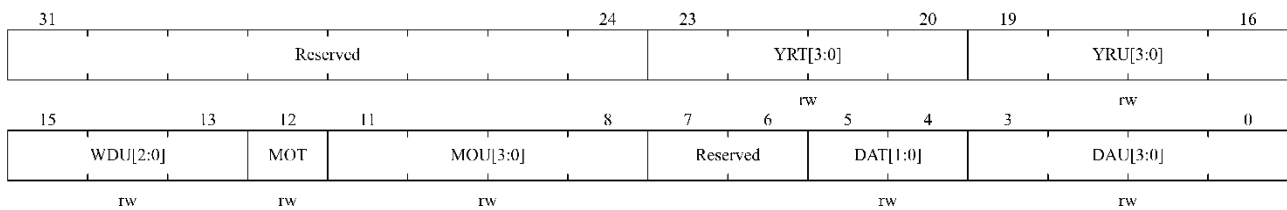
Bit Field	Name	Description
31:23	Reserved	Reserved, the reset value must be maintained.

Bit Field	Name	Description
22	APM	AM/PM format. 0: AM format or 24-hour format 1: PM format
21:20	HOT[1:0]	Describes the hour tens value in BCD format
19:16	HOU[3:0]	Describes the hour units value in BCD format
15	Reserved	Reserved, the reset value must be maintained.
14:12	MIT [2: 0]	Describes the minute tens value in BCD format
11:8	MIU[3:0]	Describes the minute units value in BCD format
7	Reserved	Reserved, the reset value must be maintained.
6:4	SCT[2:0]	Describes the second tens value in BCD format
3:0	SCU[3:0]	Describes the second units value in BCD format

21.4.3 RTC Calendar Date Register (RTC_Date)

Address offset: 0x04

Reset value: 0x0000 2101



Bit Field	Name	Description
31:24	Reserved	Reserved, the reset value must be maintained.
23:20	YRT[3:0]	Describes the year tens value in BCD format
19:16	YRU[3:0]	Describes the year units value in BCD format
15:13	WDU[2:0]	Describes which Week day 000: Forbidden 001: Monday ... 111: Sunday
12	MOT	Describes the month tens value in BCD format
11:8	MOU[3:0]	Describes the month units value in BCD format
7:6	Reserved	Reserved, the reset value must be maintained.
5:4	DAT[1:0]	Describes the date tens value in BCD format
3:0	DAU[3:0]	Describes the date units value in BCD format

21.4.4 RTC Control Register (RTC_Ctrl)

Address offset: 0x08

Reset value: 0x0000 0000

31				24				23	22	21	20	19	18	17	16
Reserved								COEN	OUTSEL[1:0]	OPOL	CALOSEL	BAKP	SUIH	ADIH	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSIEN	WTIEN	ALBIEN	ALAIEN	TSEN	WTEN	ALBEN	ALAEEN	Reserved	HFMT	BYPS	REF CLKEN	TEDGE	WKUPSEL[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	w	w

Bit Field	Name	Description
31:24	Reserved	Reserved, the reset value must be maintained.
23	COEN	Calibration output enable This bit controls RTC_CALIB output 0: Disable calibration output 1: Enable calibration output
22:21	OUTSEL[1:0]	Output selection These bits are used to select the alarm/wakeup output 00: Disable output 01: Enable Alarm A output 10: Enable Alarm B output 11: Enable Wakeup output
20	OPOL	Output polarity bit This bit is used to configure the polarity of output. 0: Outputs high level when the selected output triggers(see OUTSEL[1:0]) 1: Outputs low level when the selected output triggers(see OUTSEL[1:0])
19	CALOSEL	Calibration output selection When RTC_CTRL.COEN=1, RTCCLK = 32.768KHz and prescale at their default value (RTC_PRE.DIVA[6:0]=127 and RTC_PRE.DIVS[14:0]=255). 0: Calibration output is 256 Hz 1: Calibration output is 1 Hz
18	BAKP	Daylight saving time record This bit is written by the user 0: Not record daylight saving time 1: Record daylight saving time
17	SUIH	Subtract 1 hour (winter time change) 1 hour will be subtracted to the calendar time when the current hour value is not 0. This bit is always read as 0. 0: No effect. 1: Subtracts 1 hour to the current time.
16	ADIH	Add 1 hour (summer time change) When this bit is set, 1 hour can be added to the calendar time. This bit is always read as 0. 0: No effect. 1: Adds 1 hour to the current time.
15	TSIEN	Time-stamp interrupt enable 0: Disable time-stamp interrupt. 1: Enable time-stamp interrupt.

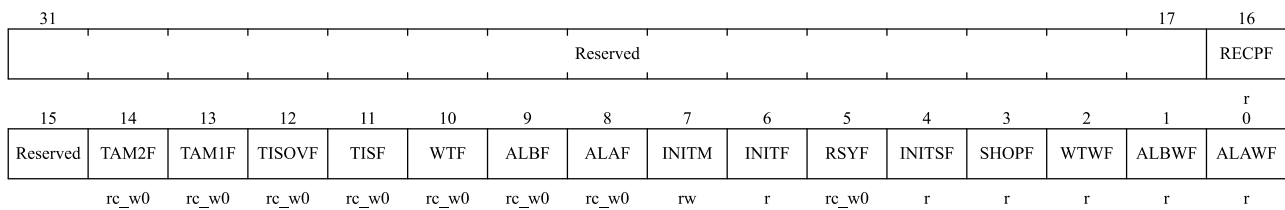
Bit Field	Name	Description
14	WTIEN	Wakeup timer interrupt enable 0: Disable wakeup timer interrupt. 1: Enable wakeup timer interrupt.
13	ALBIEN	Alarm B interrupt enable 0: Disable Alarm B interrupt 1: Enable Alarm B Interrupt
12	ALAIEN	Alarm A interrupt enable 0: Disable Alarm A interrupt 1: Enable Alarm A interrupt
11	TSEN	Timestamp enable 0: Disable timestamp 1: Enable timestamp
10	WTEN	Wakeup timer enable 0: Disable wakeup timer 1: Enable wakeup timer
9	ALBEN	Alarm B enable 0: Disable Alarm B 1: Enable Alarm B
8	ALAEN	Alarm A enable 0: Disable Alarm A 1: Enable Alarm A
7	Reserved	Reserved, the reset value must be maintained.
6	HFMT	Hour format bit 0: 24 hour format 1: Am/PM format
5	BYPS	Bypass values from the shadow registers 0: Calendar values are copied from the shadow registers, which are refreshed every two RTCCLK cycles. 1: Calendar values are copied directly from the calendar counters. <i>Note: If the frequency of the APB1 clock falls below seven times the frequency of RTCCLK, RTC_CTRL.BYPS bit must be set to '1'</i>
4	REFCLKEN	RTC_REFIN reference clock detection enable (50 or 60 Hz) 0: Disable RTC_REFIN detection 1: Enable RTC_REFIN detection <i>Note: RTC_PRE.DIVS must be 0x00FF</i>
3	TEDGE	Time-stamp event active edge 0: Input rising edge creates a timestamp event 1: Input falling edge creates a timestamp event TSEN need to be reset when TEDGE is changed to avoid unwanted RTC_INITSTS.TISF setting.
2:0	WKUPSEL[2:0]	Wakeup clock selection 000: RTC clock is divided by 16 001: RTC clock is divided by 8

Bit Field	Name	Description
		010: RTC clock is divided by 4 011: RTC clock is divided by 2 10x: ck_spre (usually 1Hz) clock is selected 11x: ck_spre (usually 1Hz) clock is selected and 2 ¹⁶ is added to the RTC_WKUPT.WKUPT counter.

21.4.5 RTC Initial Status Register (RTC_Initsts)

Address offset: 0x0C

Reset value: 0x0000 0007



Bit Field	Name	Description
31:17	Reserved	Reserved, the reset value must be maintained.
16	RECPF	Recalibration pending flag The RECPF status flag is automatically set to '1' when software writes to the RTC_CALIB register, indicating that the RTC_CALIB register is blocked. After the new calibration settings are processed, this bit returns to '0'.
15	Reserved	Reserved, the reset value must be maintained.
14	TAM2F	RTC_TAMP2 detection flag This flag is set to '1' by hardware when a tamper event is detected on the RTC_TAMP2 input pin. This flag can be cleared by software writing 0
13	TAM1F	RTC_TAMP1 detection flag This flag is set to '1' by hardware when a tamper event is detected on the RTC_TAMP1 input pin. This flag can be cleared by software writing 0
12	TISOVF	The time-stamp overflow flag This flag is set to '1' by hardware when a time-stamp event happens when TISF bit is set. This flag can be cleared by software writing 0. It is advised to check and clear TISOVF only after clearing the TISF bit. Otherwise, an overflow might not be noticed if a timestamp event occurs immediately before the TISF bit is being cleared.
11	TISF	Time-stamp flag This flag is set to '1' by hardware when a time-stamp event happens. This flag can be cleared by software writing 0
10	WTF	Wake up timer flag This flag is set by hardware when the value of wakeup auto-reload counter reaches 0. This flag is cleared by software by writing 0.

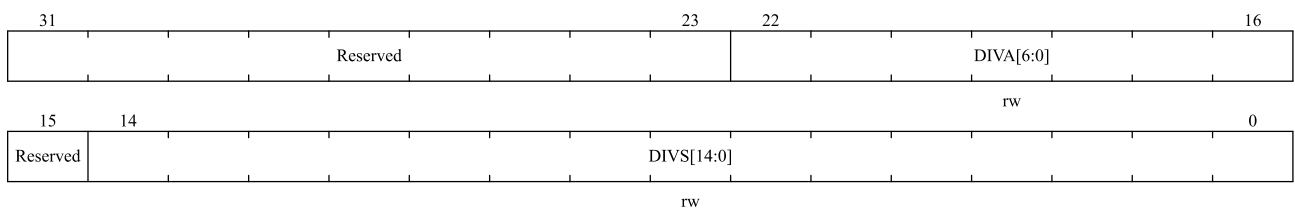
Bit Field	Name	Description
		This flag must be cleared by software at least 1.5 RTCCLK periods before WTF is set again.
9	ALBF	Alarm B flag This flag is set to '1' by hardware when the time/date registers value match the Alarm B register values. This flag can be cleared by software writing 0
8	ALAF	Alarm A flag This flag is set to '1' by hardware when the time/date registers value match the Alarm A register values. This flag can be cleared by software writing 0
7	INITM	Enter initialization mode 0: Free running mode 1: Enter initialization mode and set calendar time value, date value, and prescale value.
6	INITF	Initialization flag RTC is in initialization state when this bit is '1', and calendar time, date and prescale value can be updated. 0: Calendar time, date and prescale value can not be updated 1: Calendar time, date and prescale value can be updated
5	RSYF	Register synchronization flag This flag is set to '1' by hardware when the calendar value are copied into the shadow registers. This bit is cleared by hardware when in initialization mode, while a shift operation is pending (SHOPF=1), or when in bypass shadow register mode (RTC_CTRL.BYPS=1). This bit can also be cleared by software. It is cleared either by software or by hardware in initialization mode. 0: Calendar shadow register not yet synchronized 1: Calendar shadow register synchronized
4	INITSF	Initialization status flag This flag is set to '1' by hardware when the calendar year field is different from 0 (which is the RTC domain reset state). 0: Calendar has not been initialized 1: Calendar has been initialized
3	SHOPF	Shift operation pending flag This flag is set to '1' by hardware as soon as a shift operation is initiated by a write to the RTC_SCTRL register. It is cleared by hardware when the corresponding shift operation has been completed, note that writing to the SHOPF bit has no effect. 0: No shift operation is pending 1: A shift operation is pending
2	WTWF	Wakeup timer write flag 0: Wakeup timer configuration update is not allowed 1: Wakeup timer configuration update is allowed
1	ALBWF	Alarm B write flag This flag is set to '1' by hardware when Alarm B values can be changed, after the

Bit Field	Name	Description
		RTC_CTRL.ALBEN bit has been set to 0. 0: Alarm B update is not allowed 1: Alarm B update is allowed
0	ALAWF	Alarm A write flag. This flag is set to '1' by hardware when Alarm A values can be changed, after the RTC_CTRL.ALAEN bit has been set to 0. 0: Alarm A update is not allowed 1: Alarm A update is allowed

21.4.6 RTC Prescaler Register (RTC_Pre)

Address offset: 0x10

Reset value: 0x007F 00FF

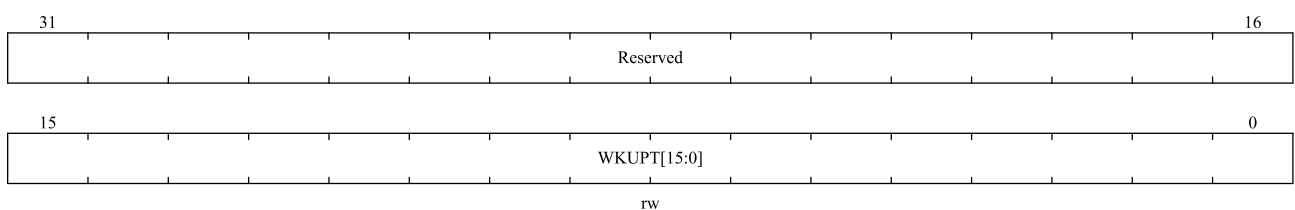


Bit Field	Name	Description
31:23	Reserved	Reserved, the reset value must be maintained.
22:16	DIVA[6:0]	Asynchronous prescaler factor $f_{ck_apre} = RTCCLK / (DIVA[6:0] + 1)$
15	Reserved	Reserved, the reset value must be maintained.
14:0	DIVS[14:0]	Synchronous prescaler factor $f_{ck_spre} = f_{ck_apre} / (DIVS[14:0] + 1)$

21.4.7 RTC Wakeup Timer Register (RTC_WKUPT)

Address offset: 0x14

Reset value: 0x0000 FFFF



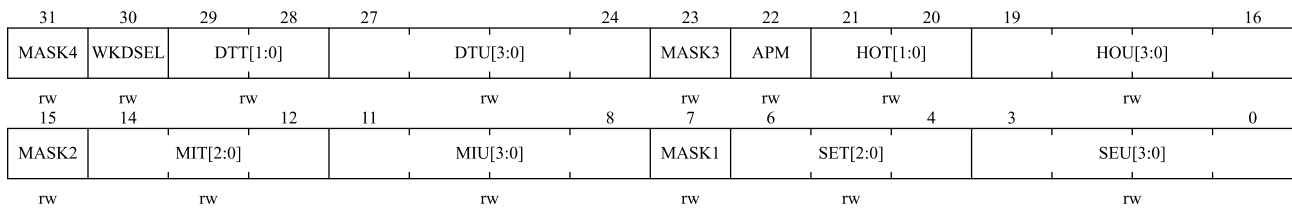
Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained.
15:0	WKUPT[15:0]	Wake up auto-reload value bits The RTC_INITSTS.WTF flag is set every (WKUPT[15:0] + 1) ck_wut cycles when the RTC_CTRL.WTEN=1. The wakeup timer becomes 17-bits When

Bit Field	Name	Description
		RTC_CTRL.WKUPSEL[2]=1. <i>Note:</i> This register change (such as the second setting or later Settings) needs to be changed in the wakeup interrupt, otherwise the changed Settings will not take effect immediately, but will take effect after the next wakeup; In particular, when RTC_CTRL.WKUPSEL[2:0] is set to 010, the modified setting does not take effect immediately, but will take effect after wake up in the next cycle.

21.4.8 RTC Alarm A Register (RTC_ALARM_A)

Address offset: 0x1C

Reset value: 0x0000 0000



Bit Field	Name	Description
31	MASK4	Alarm date mask 0: Date/day match 1: Date/day not match
30	WKDSEL	Week day selection 0: DTU[3:0] represents the date units 1: DTU[3:0] represents week day only. DTT[1:0] is not considered
29:28	DTT[1:0]	Describes the date tens value in BCD format
27:24	DTU[3:0]	Describes the date units value in BCD format
23	MASK3	Alarm hours mask 0: Hours match 1: Hours not match
22	APM	AM/PM notation 0: AM or 24 hours format 1: PM format
21:20	HOT[1:0]	Describes the hour tens value in BCD format
19:16	HOU[3:0]	Describes the hour units value in BCD format
15	MASK2	Alarm minutes mask 0: Minutes match 1: Minutes not match
14:12	MIT[2:0]	Describes the minute tens value in BCD format
11:8	MIU[3:0]	Describes the minute units value in BCD format
7	MASK1	Alarm seconds mask 0: Seconds match

Bit Field	Name	Description
		1: Seconds not match
6:4	SET[2:0]	Describes the second tens value in BCD format
3:0	SEU[3:0]	Describes the second units value in BCD format

21.4.9 RTC Alarm B Register (RTC_ALARM B)

Address offset: 0x20

Reset value: 0x0000 0000

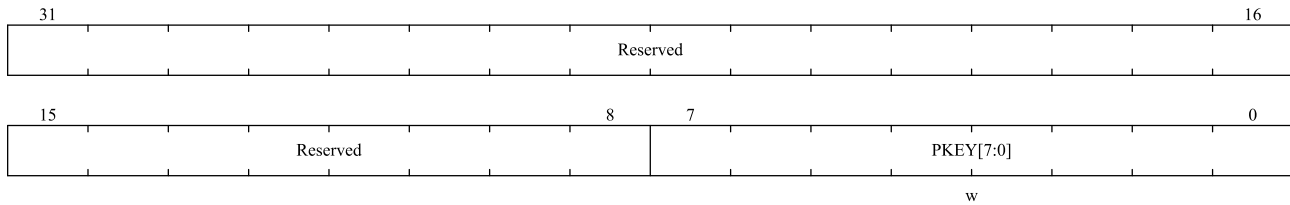
31	30	29	28	27	24	23	22	21	20	19	16	
MASK4	WKDSEL	DTT[1:0]		DTU[3:0]			MASK3	APM	HOT[1:0]		HOU[3:0]	
rw	rw	rw		rw			rw	rw	rw		rw	
15	14	12	11	8	7	6	4	3	0			
MASK2	MIT[2:0]		MIU[3:0]			MASK1	SET[2:0]		SEU[3:0]			
rw	rw		rw			rw	rw		rw			

Bit Field	Name	Description
31	MASK4	Alarm date mask 0: Date/day match 1: Date/day not match
30	WKDSEL	Week day selection 0: DTU[3:0] represents the date units 1: DTU[3:0] represents week day only. DTT[1:0] is not considered
29:28	DTT[1:0]	Describes the date tens value in BCD format
27:24	DTU[3:0]	Describes the date units value in BCD format
23	MASK3	Alarm hours mask 0: Hours match 1: Hours not match
22	APM	AM/PM notation 0: AM or 24 hours format 1: PM format
21:20	HOT[1:0]	Describes the hour tens value in BCD format
19:16	HOU[3:0]	Describes the hour units value in BCD format
15	MASK2	Alarm minutes mask 0: Minutes match 1: Minutes not match
14:12	MIT[2:0]	Describes the minute tens value in BCD format
11:8	MIU[3:0]	Describes the minute units value in BCD format
7	MASK1	Alarm seconds mask 0: Seconds match 1: Seconds not match
6:4	SET[2:0]	Describes the second tens value in BCD format
3:0	SEU[3:0]	Describes the second units value in BCD format

21.4.10 RTC Write Protection Register (RTC_WRP)

Address offset: 0x24

Reset value: 0x0000 0000

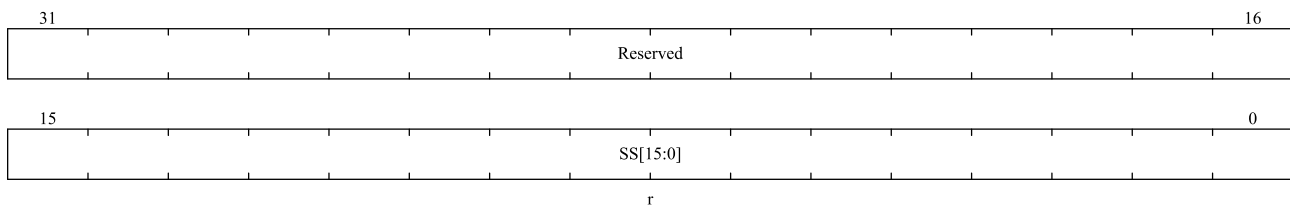


Bit Field	Name	Description
31:8	Reserved	Reserved, the reset value must be maintained.
7:0	PKEY[7:0]	Write protection key Reading this byte always returns 0x00. For detail on how to unlock RTC register write protection, refer to chapter RTC register write protection.

21.4.11 RTC Sub-second Register (RTC_SUBS)

Address offset: 0x28

Reset value: 0x0000 0000

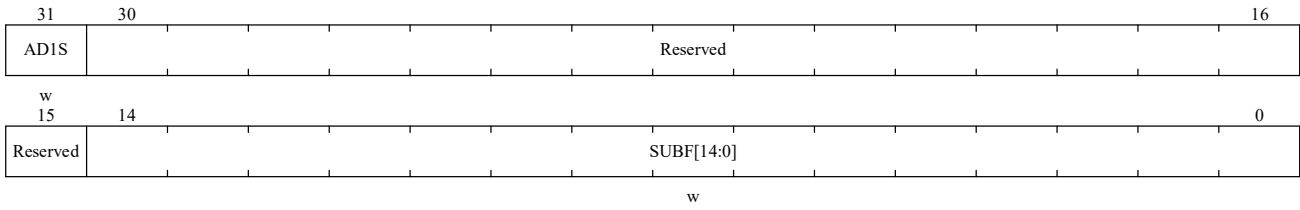


Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained.
15:0	SS[15:0]	Sub-second bit. This value is the value of the synchronous prescaler counter. This sub-second value is calculated by the below formula: Sub-second value = (RTC_PRE.DIVS[14:0]-SS)/(RTC_PRE.DIVS[14:0]+1) <i>Note: SS[15:0] can be larger than RTC_PRE.DIVS[14:0] only after the shift operation is finished. In this case, the correct time/date is one second slower than the time/date indicated by RTC_TSH/RTC_DATE.</i>

21.4.12 RTC Shift Control Register (RTC_SCTRL)

Address offset: 0x2C

Reset value: 0x0000 0000

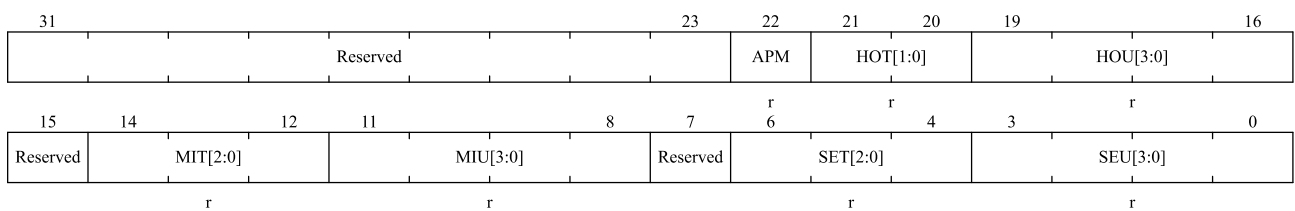


Bit Field	Name	Description
31	ADIS	Add one second 0: No add one second. 1: Add one second to the clock/calendar This bit can only be written and read as zero. Writing to this bit does not have an impact when RTC_INITSTS.SHOPF=1.
30:15	Reserved	Reserved, the reset value must be maintained.
14:0	SUBF[14:0]	Subtract a fraction of a second There bits can only be written and read as zero.. Writing to this bit does not have an impact when RTC_INITSTS.SHOPF=1. The value which is written to SUBF is added to the synchronous prescaler counter, and the clock will delay: Delay (seconds) = (SUBF[14:0]+1) / (DIVS[14:0] + 1) ADIS bit can be used together with the SUBF[14:0]bits: Advance (seconds) = (1 - ((SUBF[14:0] +1)/ (DIVS[14:0] + 1))) <i>Note: RTC_INITSTS.RSYF bit will be cleared when write SUBF[14:0]. When RTC_INITSTS.RSYF=1, the shadow registers have been updated with the shifted time. When ADIS is set to 1, SUBF[14:0] cannot be all 0</i>

21.4.13 RTC Timestamp Time Register (RTC_TST)

Address offset: 0x30

Reset value: 0x0000 0000



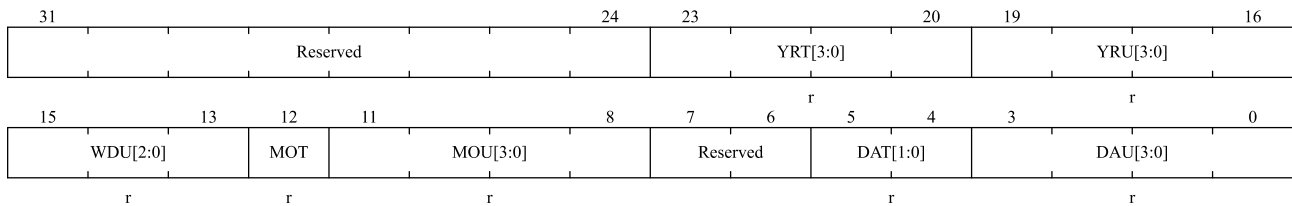
Bit Field	Name	Description
31:23	Reserved	Reserved, the reset value must be maintained.
22	APM	AM/PM notation 0: AM or 24-hour clock 1: PM
21:20	HOT[1:0]	Describes the hour tens value in BCD format
19:16	HOU[3:0]	Describes the hour units value in BCD format
15	Reserved	Reserved. Must be kept at the reset value
14:12	MIT[2:0]	Describes the minute tens value in BCD format

Bit Field	Name	Description
11:8	MIU[3:0]	Describes the minute units value in BCD format
7	Reserved	Reserved, the reset value must be maintained.
6:4	SET[2:0]	Describes the second tens value in BCD format
3:0	SEU[3:0]	Describes the second units value in BCD format

21.4.14 RTC Timestamp Date Register (RTC_TSD)

Address offset: 0x34

Reset value: 0x0000 0000

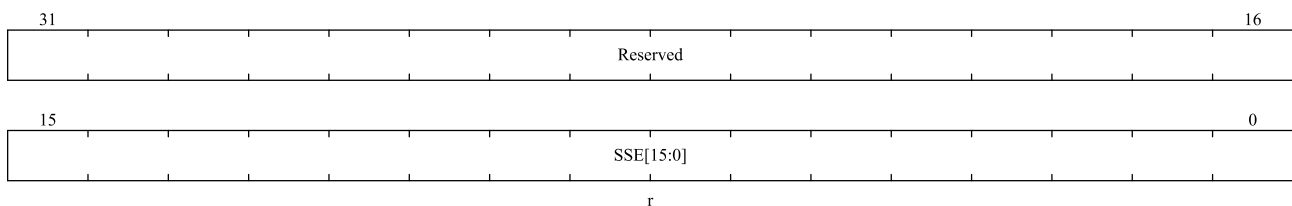


Bit Field	Name	Description
31:24	Reserved	Reserved, the reset value must be maintained.
23:20	YRT[3:0]	Describes the year tens value in BCD format
19:16	YRU[3:0]	Describes the year units value in BCD format
15:13	WDU[2:0]	Describes which Week day 000: Forbidden 001: Monday ... 111: Sunday
12	MOT	Describes the month tens value in BCD format
11:8	MOU[3:0]	Describes the month units value in BCD format
7:6	Reserved	Reserved, the reset value must be maintained.
5:4	DAT[1:0]	Describes the date tens value in BCD format
3:0	DAU[3:0]	Describes the date units value in BCD format

21.4.15 RTC Timestamp Sub-second Register (RTC_TSSS)

Address offset: 0x38

Reset value: 0x0000 0000



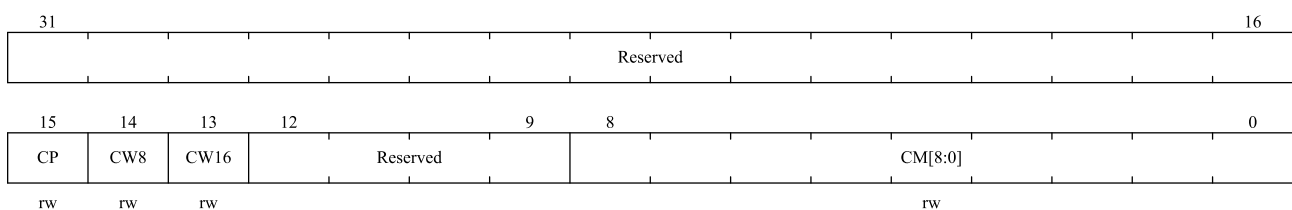
Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained.

Bit Field	Name	Description
15:0	SSE[15:0]	Sub second value SSE[15:0] is the value in the synchronous prescaler counter. The fraction of a second is provided by the formula below: $\text{Second fraction} = (\text{RTC_PRE.DIVS}[14:0] - \text{SSE}) / (\text{RTC_PRE.DIVS}[14:0] + 1)$ Note: SSE can be larger than DIVS only after a shift operation. In that case, the correct time/date is one second less than as indicated by RTC_TSH/RTC_DATE.

21.4.16 RTC Calibration Register (RTC_CALIB)

Address offset: 0x3C

Reset value: 0x0000 0000

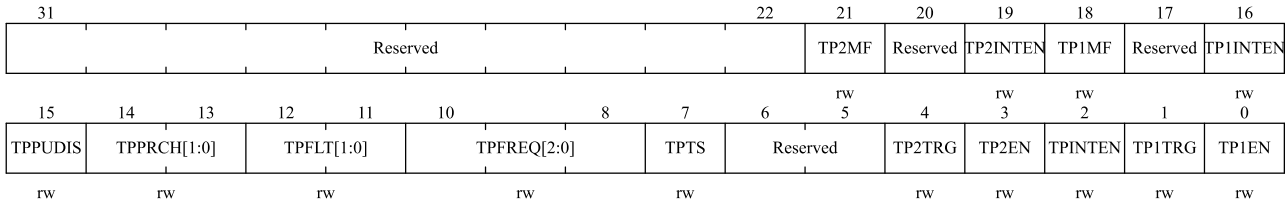


Bit Field	Name	Description
31:16	Reserved	Reserved, the reset value must be maintained.
15	CP	Increase frequency of RTC by 488.5 ppm This feature is intended to be used along with CM[8:0]. When RTCCLK frequency is 32768 Hz, the number of RTCCLK pulses added during a 32-second window is $((512 * CP) - CM[8:0])$. 0: No add pulse. 1: One RTCCLK pulse is inserted every 2^{11} pulses.
14	CW8	Select an 8-second calibration cycle period 0: Not effect. 1: Select an 8-second calibration period. When CW8 is set to '1', the 8-second calibration cycle period is selected. Note: when CW8 = 1, CM[1:0] will always be '00'
13	CW16	To select a 16-second calibration cycle period 0: Not effect. 1: Select a calibration period of 16 seconds. If CW8 = 1, this bit cannot be set to 1. Note: when CW16 = 1, CM[0] will always be '0'
12:9	Reserved	Reserved, the reset value must be maintained.
8:0	CM[8:0]	Negative calibration bits The number of mask pulse out of 2^{20} RTCCLK pulses. This effectively decreases the frequency of the calendar with a resolution of 0.9537 ppm.

21.4.17 RTC Tamper Configuration Register (RTC_TMPCFG)

Address offset: 0x40

Reset value: 0x0000 0000



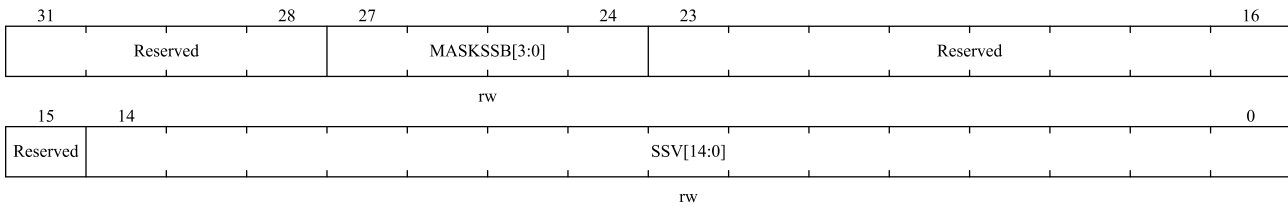
Bit Field	Name	Description
31:22	Reserved	Reserved, the reset value must be maintained.
21	TP2MF	Tamper 2 mask flag 0: Not mask tamper 2 event. 1: Mask tamper 2 event. <i>Note: The Tamper 2 interrupt must not be enabled when TP2MF is set.</i>
20	Reserved	Reserved, the reset value must be maintained.
19	TP2INTEN	Tamper 2 interrupt enable 0: Disable tamper 2 interrupt when TPINTEN = 0. 1: Enabled tamper 2 interrupt
18	TP1MF	Tamper 1 mask flag 0: Not mask tamper 1 event. 1: Mask tamper 1 event. <i>Note: The Tamper 1 interrupt must not be enabled when TP1MF is set.</i>
17	Reserved	Reserved, the reset value must be maintained.
16	TP1INTEN	Tamper 1 interrupt enable 0: Disable tamper 1 interrupt when TPINTEN = 0. 1: Enabled tamper 1 interrupt
15	TPPUDIS	RTC_TAMPx Pull-up disable bit. 0: Enable precharge RTC_TAMPx pins before each sampling. 1: Disable precharge RTC_TAMPx pins
14:13	TPPRCH[1:0]	RTC_TAMPx Precharge duration. These bits determine the the precharge time before each sampling. 0x0: 1 RTCCLK cycles 0x1: 2 RTCCLK cycles 0x2: 4 RTCCLK cycles 0x3: 8 RTCCLK cycles
12:11	TPFLT[1:0]	RTC_TAMPx filter count These bits determine the number of consecutive samples when occur active level. 0x0: Triggers a tamper event at the active level. 0x1: Triggers a tamper event after 2 consecutive samples at the active level. 0x2: Triggers a tamper event after 4 consecutive samples at the active level. 0x3: Triggers a tamper event after 8 consecutive samples at the active level.
10:8	TPFREQ[2:0]	Tamper sampling frequency This bit determines the frequency at the each RTC_TAMPx input is sampled. 0x0: Sampling once every 32768 RTCCLK (1 Hz when RTCCLK = 32.768 KHz). 0x1: Sampling once every 16384 RTCCLK.

Bit Field	Name	Description
		0x2: Sampling once every 8192 RTCCLK. 0x3: Sampling once every 4096 RTCCLK. 0x4: Sampling once every 2048 RTCCLK. 0x5: Sampling once every 1024 RTCCLK. 0x6: Sampling once every 512 RTCCLK. 0x7: Sampling once every 256 RTCCLK.
7	TPTS	Tamper event trigger timestamp 0: Disable tamper event trigger timestamp 1: Enable tamper event trigger timestamp TPTS is valid even if TSEN=0 in the RTC_CTRL register.
6:5	Reserved	Reserved, the reset value must be maintained.
4	TP2TRG	Tamper 2 event trigger edge if TPFLT[1:0] != 00, tamper detection is in level mode: 0: low level trigger a tamper detection event. 1: high level trigger a tamper detection event. if TPFLT = 00, tamper detection is in edge mode: 0: Rising edge trigger a tamper detection event. 1: Falling edge trigger a tamper detection event
3	TP2EN	Tamper 2 detection enable 0: Disable tamper detection 1: Enable tamper detection
2	TPINTEN	Tamper event interrupt enable. 0: Disable tamper interrupt 1: Enable tamper interrupt <i>Note: This bit enables the interrupt of all tamper pins events, regardless of TPxINTEN level. If this bit is cleared, each tamper event interrupt can be individually enabled by setting TPxINTEN.</i>
1	TP1TRG	Tamper 1 event trigger edge if TPFLT[1:0] != 00, tamper detection is in level mode: 0: low level trigger a tamper detection event. 1: high level trigger a tamper detection event. if TPFLT = 00, tamper detection is in edge mode: 0: Rising edge trigger a tamper detection event. 1: Falling edge trigger a tamper detection event
0	TP1EN	Tamper 1 detection enable 0: Disable tamper detection 1: Enable tamper detection

21.4.18 RTC Alarm A Sub-second Register (RTC_ALRMAS)

Address offset: 0x44

Reset value: 0x0000 0000

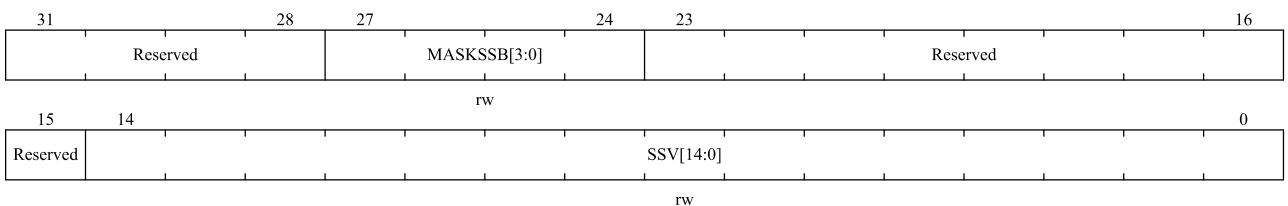


Bit Field	Name	Description
31:28	Reserved	Reserved, the reset value must be maintained.
27:24	MASKSSB[3:0]	Mask the most significant bit from this bits. 0x0: No comparison on sub seconds for Alarm. The alarm is set when the seconds unit is incremented (assuming that the rest of the fields match). 0x1: Only SSV[0] is compared and other bits are not compared. 0x2: Only SSV[1:0] are compared and other bits are not compared. 0x3: Only SSV[2:0] are compared and other bits are not compared. ... 0xC: Only SSV[11:0] are compared and other bits are not compared. 0xD: Only SSV[12:0] are compared and other bits are not compared. 0xE: Only SSV[13:0] are compared and other bits are not compared. 0xF: SSV[14:0] are compared Synchronization counter RTC_SUBS.SS[15] bit is never compared.
23:15	Reserved	Reserved, the reset value must be maintained.
14:0	SSV[14:0]	Sub seconds value This value is compared with the synchronous prescaler counter RTC_SUBS.SS[14:0], and bit number of compared is controlled by MASKSSB[3:0].

21.4.19 RTC Alarm B Sub-Second Register (RTC_ALRMBSS)

Address offset: 0x48

Reset value: 0x0000 0000



Bit Field	Name	Description
31:28	Reserved	Reserved, the reset value must be maintained.
27:24	MASKSSB[3:0]	Mask the most significant bit from this bits. 0x0: No comparison on sub seconds for Alarm. The alarm is set when the seconds unit is incremented (assuming that the rest of the fields match). 0x1: Only SSV[0] is compared and other bits are not compared. 0x2: Only SSV[1:0] are compared and other bits are not compared. 0x3: Only SSV[2:0] are compared and other bits are not compared.

Bit Field	Name	Description
		<p>...</p> <p>0xC: Only SSV[11:0] are compared and other bits are not compared.</p> <p>0xD: Only SSV[12:0] are compared and other bits are not compared.</p> <p>0xE: Only SSV[13:0] are compared and other bits are not compared.</p> <p>0xF: SSV[14:0] are compared.</p> <p>Synchronization counter RTC_SUBS.SS[15] bit is never compared.</p>
23:15	Reserved	Reserved, the reset value must be maintained.
14:0	SSV[14:0]	<p>Sub seconds value</p> <p>This value is compared with the synchronous prescaler counter RTC_SUBS.SS[14:0], and bit number of compared is controlled by MASKSSB[3:0].</p>

22 Operational Amplifier (OPAMP)

The OPAMP module can be flexibly configured. It is suitable for applications such as independent operational programming mplifier (PGA) mode and follower mode. The input range of OPAMP is 0V to V_{DDA} and the output range is 0.1V to $V_{DDA}-0.1V$. The output of the OPAMP is internally connected to the input channel of the ADC for analog signal measurement.

22.1 OPAMP Features

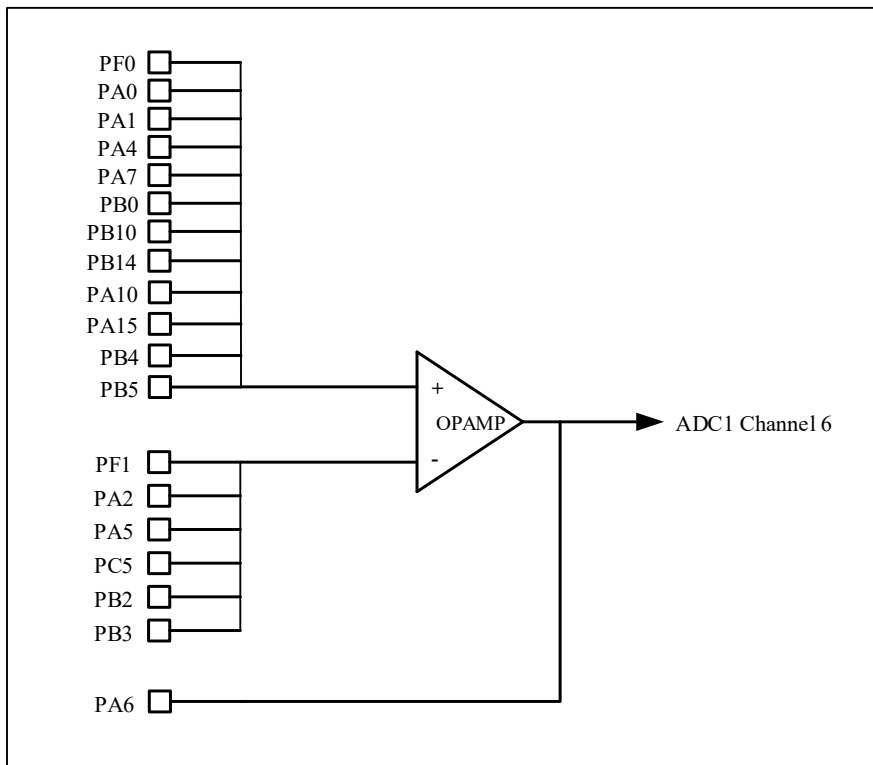
- Support rail-to-rail input
- OPA linear output range 0.4V~ $V_{DDA}-0.4V$
- Can be configured as independent OPAMP and programmable gain OPAMP
- Non-inverted and inverted input multiple selection
- OPAMP operating mode can be configured as:
 - Independent mode (external gain setting)
 - PGA mode, programmable gain is set to 2X, 4X, 8X, 16X, 32X
 - Follower mode
- The internally connected ADC channel is used to measure the output signal of the OPAMP

22.1.1 OPAMP Function Description

The OPAMP can be configured to various PGA modes through register selection, and can also be configured as the OPAMP function of the user using external components. The output of the OPAMP can be used as the input channel to the ADC. The OPAMP outputs are connected to the analog channel of the ADC as follows.

The output of OPAMP is connected to analog input Channel6 of ADC.

Figure 22-1 Block Diagram of OPAMP Connection Diagram

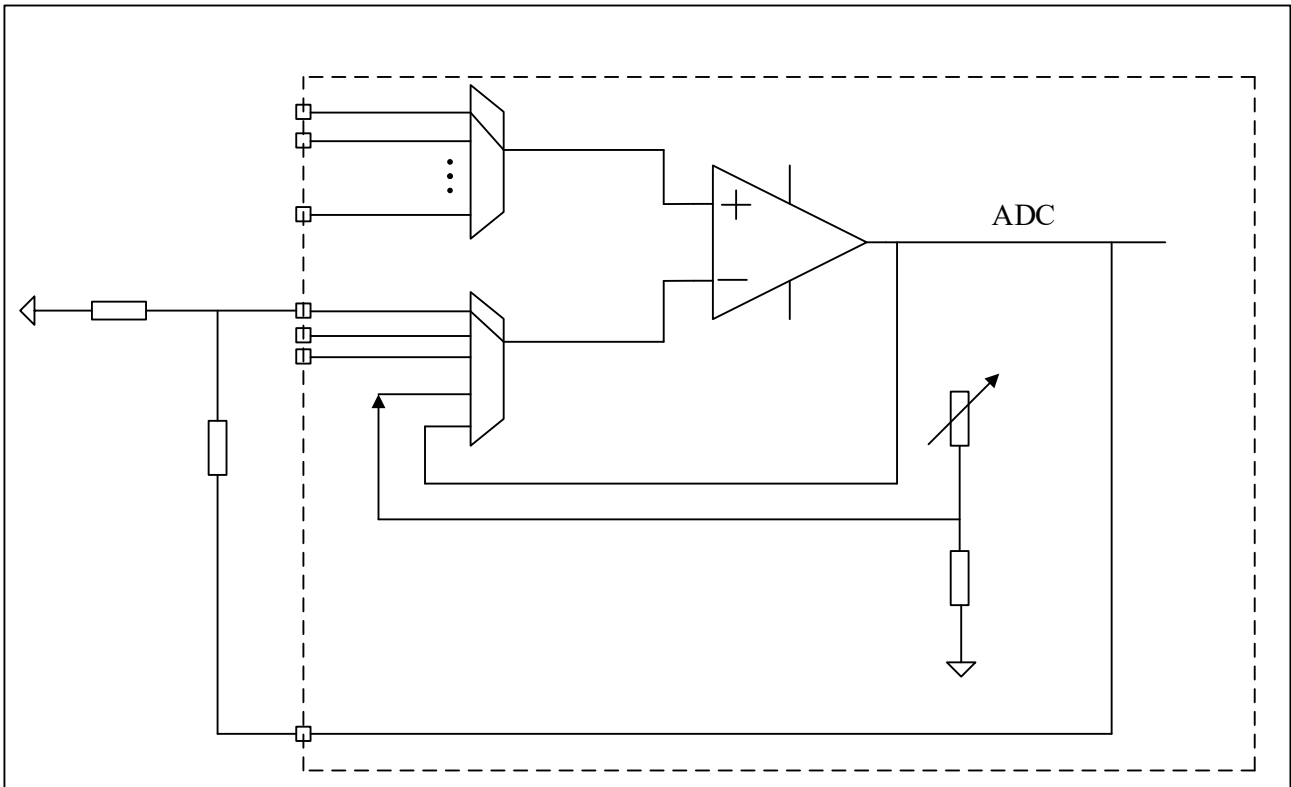


22.2 OPAMP Operating Mode

22.2.1 External Gain Mode

The external gain mode means that the gain factor is determined by the connected resistance and capacitance. When OPAMAP_CS.MOD is set to 2'b00 or 2'b01 for operational amplifier function. The OPAMAP_CS.VPSSEL or OPAMAP_CS.VPSEL selects non-inverted input, and the OPAMAP_CS.VMSSEL or OPAMAP_CS.VMSEL selects inverted input. An external resistor is used to form a closed-loop amplification system. As shown in the figure below, the non inverted terminal, inverted terminal and output terminal of the OPAMP are all connected to the external port. The amplification factor is determined by the external resistance-capacitance network.

Figure 22-2 OPAMP External Gain Mode



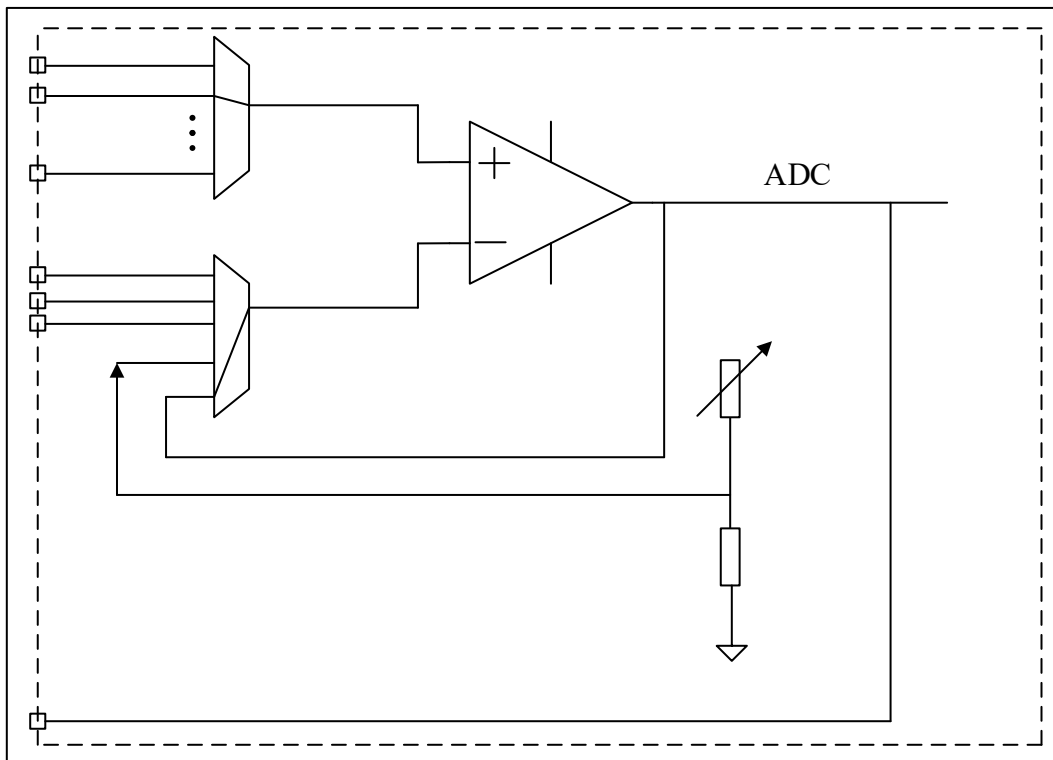
22.2.2 Follower Mode

In follower mode, the output voltage is directly following the input voltage. The VMSEL terminal must be configured to connected to the OPAMP output port.

The OPAMP_CS.MOD is set to 2'b11 for internal follow function, the OPAMAP_CS.VPSSEL or OPAMAP_CS.VPSEL selects non-inverted input, the OPAMAP_CS.VMSSEL or OPAMAP_CS.VMSEL is connected to the output port from inside the chip.

Unused VM pin can be used as other GPIO.

Figure 22-3 Follower Mode



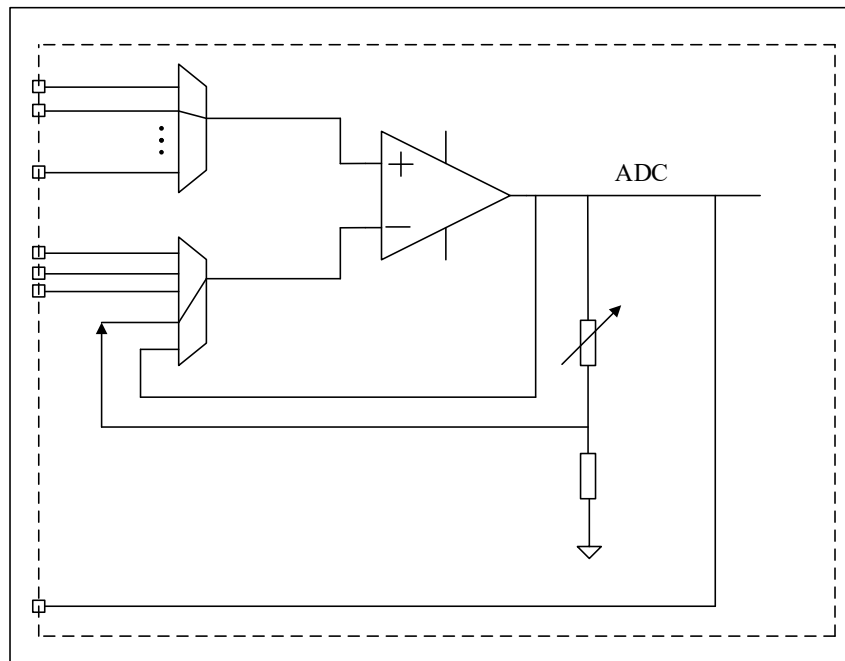
22.2.3 Programming Gain Amplifier (PGA) Mode

The programming gain amplifier (PGA) mode amplifies the input voltage through the built-in resistance feedback network

The OPAMP_CS.MOD is set to 2'b10 for PGA function, supports gain of 2/4/8/16/32, the OPAMP_CS.VMSSEL or OPAMP_CS.VMSEL pins must be set to floating. The OPAMP_CS.VPSSEL or OPAMP_CS.VPSEL selects non-inverted input. The non-inverted input can be connected to an external pin, which can be a resistive network. Set OPAMP_CS.PGAGAN to gain selection. The output of an OPAMP can be connected to a resistive network.

OPAMP's VM input pin can be used as a normal GPIOs.

Figure 22-4 Internal Programmable Gain Mode



22.2.4 Independent Write Protection

The write protection of OPAMP can be set independently by configuring the OPAMP_LOCK register. Once the write protection is set, the software will not be able to write to the corresponding OPAMP register, and the write protection function can be canceled only after the chip is reset.

22.2.5 TIMER Controlled Switching Mode

In some applications, input switching of OPAMP can be done through TIM1_CC6. TIM1_CC6 controls the input switching of the OPAMP.

When TIM1_CC6 is 1, the OPAMP selects the port configured by VPSEL or VMSEL as the input. Otherwise, VPSEL or VMSEL is used.

The OPAMP_CS.TCMEN is set to 1 to enable the automatic input switching. The process of configuring automatic switching is as follows:

1. Enable automatic switching function OPAMP_CS.TCMEN
2. Configure MUX configuration for the two conversion (VPSEL,VMSEL,VPSEL,VMSEL)
3. Start OPAMP and TIM

22.3 OPAMP Registers

22.3.1 OPAMP Registers Overview

Table 22-1 OPAMP Register Overview

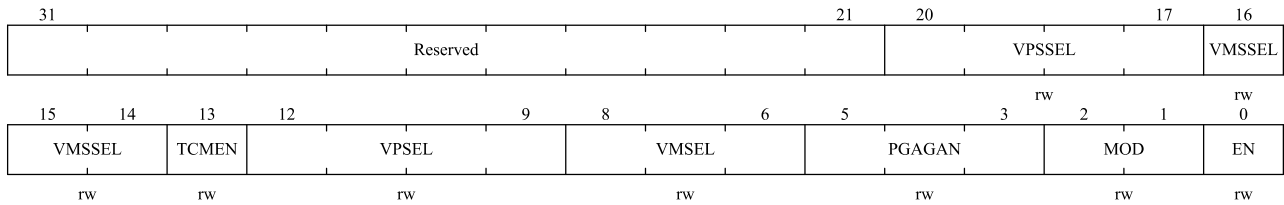
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
--------	----------	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

000h	OPAMP_CS	Reserved	VPSEL[3:0]				VMSEL[2:0]				TCMEN	VPSEL[3:0]				VMSEL[2:0]				PGAGAN[2:0]				MOD[1:0]				EN
	Reset Value		0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
004h	OPA_LOCK	Reserved																								OPAMPLK		
	Reset Value	0																										

22.3.2 OPAMP Control Status Register (OPAMP_CS)

Address offset: 0x00

Reset value: 0x0000 0000



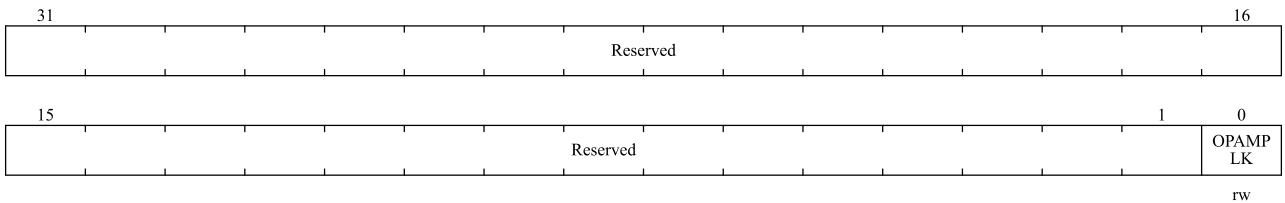
Bit Field	Name	Description																																				
31:21	Reserve	Reserved, the reset value must be maintained																																				
20:17	VPSSSEL[3:0]	OPAMP Non inverted input secondary selection. Same definition as VPSEL																																				
16:14	VMSSSEL[2:0]	OPAMP Inverted input secondary selection Same definition as VMSEL																																				
13	TCMEN	Timer control automatic switching mode enable This bit is set or cleared by software and is used to control the automatic switching of primary and secondary inputs (VPSEL, VMSEL and VPSSSEL, VMSSSEL). TIM1_CC6 automatically switches to OPAMP. 0: Turn off automatic switching 1: Turn on automatic switching																																				
12:9	VPSEL[3:0]	OPAMP Non Inverted input selection <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th>Enumerated value</th> <th>OPAMP</th> <th>Enumerated value</th> <th>OPAMP</th> </tr> </thead> <tbody> <tr> <td>0000</td> <td>PF0</td> <td>1000</td> <td>PA10</td> </tr> <tr> <td>0001</td> <td>PA0</td> <td>1001</td> <td>PA15</td> </tr> <tr> <td>0010</td> <td>PA1</td> <td>1010</td> <td>PB4</td> </tr> <tr> <td>0011</td> <td>PA4</td> <td>1011</td> <td>PB5</td> </tr> <tr> <td>0100</td> <td>PA7</td> <td>1100</td> <td>Reserve</td> </tr> <tr> <td>0101</td> <td>PB0</td> <td>1101</td> <td>Reserve</td> </tr> <tr> <td>0110</td> <td>PB10</td> <td>1110</td> <td>Reserve</td> </tr> <tr> <td>0111</td> <td>PB14</td> <td>1111</td> <td>Reserve</td> </tr> </tbody> </table>	Enumerated value	OPAMP	Enumerated value	OPAMP	0000	PF0	1000	PA10	0001	PA0	1001	PA15	0010	PA1	1010	PB4	0011	PA4	1011	PB5	0100	PA7	1100	Reserve	0101	PB0	1101	Reserve	0110	PB10	1110	Reserve	0111	PB14	1111	Reserve
Enumerated value	OPAMP	Enumerated value	OPAMP																																			
0000	PF0	1000	PA10																																			
0001	PA0	1001	PA15																																			
0010	PA1	1010	PB4																																			
0011	PA4	1011	PB5																																			
0100	PA7	1100	Reserve																																			
0101	PB0	1101	Reserve																																			
0110	PB10	1110	Reserve																																			
0111	PB14	1111	Reserve																																			
8:6	VMSEL[2:0]	OPAMP Inverted input selection <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th>Enumerated value</th> <th>OPAMP</th> <th>Enumerated value</th> <th>OPAMP</th> </tr> </thead> <tbody> <tr> <td>000</td> <td>PF1</td> <td>100</td> <td>PB2</td> </tr> <tr> <td>001</td> <td>PA2</td> <td>101</td> <td>PB3</td> </tr> <tr> <td>010</td> <td>PA5</td> <td>110</td> <td>Reserve</td> </tr> <tr> <td>011</td> <td>PC5</td> <td>111</td> <td>Reserve</td> </tr> </tbody> </table>	Enumerated value	OPAMP	Enumerated value	OPAMP	000	PF1	100	PB2	001	PA2	101	PB3	010	PA5	110	Reserve	011	PC5	111	Reserve																
Enumerated value	OPAMP	Enumerated value	OPAMP																																			
000	PF1	100	PB2																																			
001	PA2	101	PB3																																			
010	PA5	110	Reserve																																			
011	PC5	111	Reserve																																			
5:3	PGAGAN[2:0]	OPAMP Programmable amplifier gain value 000: Internal PGA gain 2																																				

Bit Field	Name	Description
		001: Internal PGA gain 4 010: Internal PGA gain 8 011: Internal PGA gain 16 100: Internal PGA gain 32 Others: Reserve
2:1	MOD[1:0]	OPAMP PGA mode 0x: External zoom mode 10: Internal PGA enable 11: Internal follower mode
0	EN	OPAMP Enable 0: Disable 1: Enable

22.3.3 OPAMP Lock Register (OPAMP_LOCK)

Address offset: 0x04

Reset value: 0x0000 0000



Bit Field	Name	Description
31:1	Reserve	Reserved, the reset value must be maintained
0	OPAMPLK	OPAMP Lock bit After the reset, this bit can be written only once 0: The OPAMP register is readable and writable 1: The OPAMP register is read-only

23 Beeper

23.1 Introduction

The Beeper module (beeper1 and beeper2) supports complementary outputs and can generate periodic signals to drive external passive beepers. It is used to generate alert tones or alarm sounds.

23.2 Function Description

Beeper as an independent module, is mounted on the APB1 bus with a maximum operating frequency of 48MHz. The timbre can be divided into 21 grades, and different timbres can be adjusted by configuring the relevant registers when using. One of the two outputs is usually turned off. If the reverse output is enabled, the two outputs are turned on at the same time and the outputs are complementary.

23.3 Beeper Registers

These peripheral registers must be operated in word (32-bit) mode.

23.3.1 Beeper Register Overview

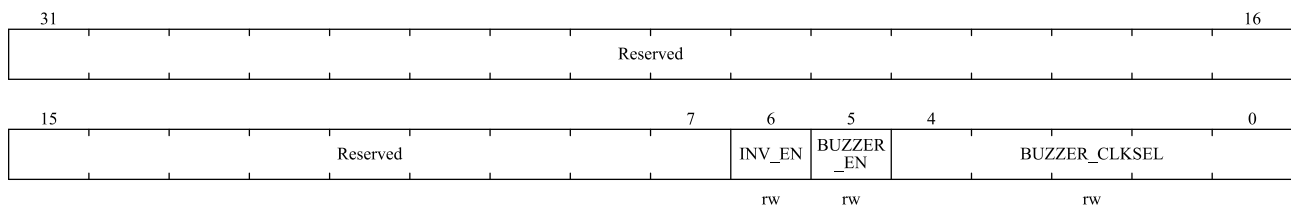
Table 23-1 Beeper Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																					
000h	BEEPER_CTRL	Reserved																								INV_EN	BUZZER_EN	BUZZER_CLKSEL																										
	Reset Value																									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23.3.2 Beeper Control Register (BEEPER_CTRL)

Address offset: 0x00

Reset value: 0x0000 0000



Bit Field	Name	Description
31:7	Reserved	Reserved, the reset value must be maintained.
6	INV_EN	Beeper complementary output enable 0: There is only one output, and the other output is closed. 1: Both outputs are turned on, and the outputs are complementary
5	BUZZER_EN	Beeper enable 0: Beeper disabled 1: Beeper enabled

Bit Field	Name	Description
4:0	BUZZER_CLKSEL[4:0]	<p>Beeper output frequency selection:</p> <p>00001: L1 (Bass 1)</p> <p>00010: L2 (Bass 2)</p> <p>00011: L3 (Bass 3)</p> <p>00100: L4 (Bass 4)</p> <p>00101: L5 (Bass 5)</p> <p>00110: L6 (Bass 6)</p> <p>00111: L7 (Bass 7)</p> <p>01000: M1 (Alto 1)</p> <p>01001: M2 (Alto 2)</p> <p>01010: M3 (Alto 3)</p> <p>01011: M4 (Alto 4)</p> <p>01100: M5 (Alto 5)</p> <p>01101: M6 (Alto 6)</p> <p>01110: M7 (Alto 7)</p> <p>01111: H1 (Treble 1)</p> <p>10000: H2 (Treble 2)</p> <p>10001: H3 (Treble 3)</p> <p>10010: H4 (Treble 4)</p> <p>10011: H5 (Treble 5)</p> <p>10100: H6 (Treble 6)</p> <p>10101: H7 (Treble 7)</p> <p>default: ...</p>

24 Arithmetic Units (HDIV and SQRT)

24.1 HDIV and SQRT Introduction

The divider (HDIV) and square root calculator (SQRT) are mainly used in scenarios with high requirements for computing energy efficiency to partially supplement the lack of calculation of the microcontroller. The divider and square root calculator can perform division or square root calculator of unsigned 32-bit integers.

24.2 HDIV and SQRT Function Description

- Only supports word operation
- 8 clock cycles to complete an unsigned integer division operation
- 32-bit dividend, 32-bit divisor, output 32-bit quotient and 32-bit remainder
- The divisor is a zero warning flag, and the division operation end flag
- 32-bit unsigned square integer, 16-bit squared root output
- Complete an unsigned integer square operation in 8 clock cycles
- Checking whether the calculation is complete can be determined by setting the interrupt enable or querying the relevant register bit

24.3 HDIV Registers

24.3.1 HDIV Register Overview

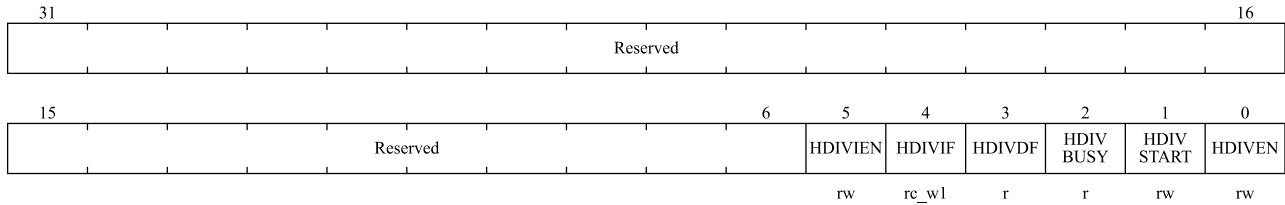
Table 24-1 HDIV Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																				
000h	HDIV_CTRLSTS	Reserved																								HDIVEN	HDIVF	HDIVDF	HDIVBUSY	HDIVSTART	HDIVEN																						
	Reset Value																									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
004h	HDIV_DIVIDEND	DIVIDEND																																																			
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																			
008h	HDIV_DIVISOR	DIVISOR																																																			
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																			
00Ch	HDIV_QUOTIENT	QUOTIENT																																																			
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																			
010h	HDIV_REMAINDER	REMAINDER																																																			
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																			
014h	HDIV_DIVBY0	Reserved																													DIVBY0																						
	Reset Value																														0																						

24.3.2 HDIV Control Status Register (HDIV_CTRLSTS)

Offset address: 0x00

Reset value: 0x0000 0000

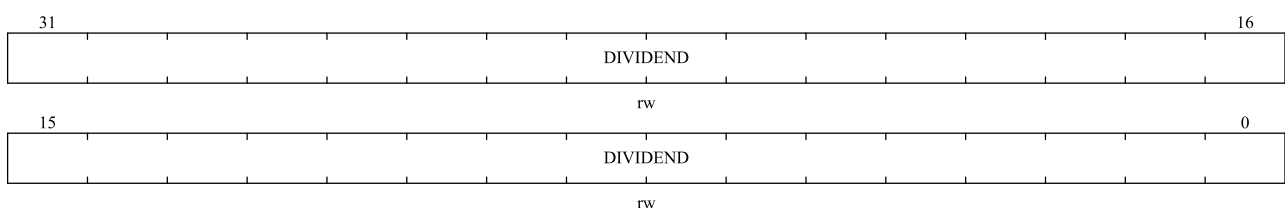


Bit Field	Name	Description
31:6	Reserved	Reserved,the reset value must be maintained
5	HDIVIEN	Divider interrupt enable 0: Disable interrupt 1: Enable interrupt
4	HDIVIF	The divider interrupt flag bit, which generates an interrupt signal when the interrupt enable is turned on 0: No interrupt is generated 1: The division calculation is completed and an interrupt is generated Software writes '1' to clear this status bit.
3	HDIVDF	Completion of calculation flag. 0: The division calculation has not ended or has not started 1: The division calculation ends. Readable quotient and remainder
2	HDIVBUSY	Divider busy flag 0: Divider idle 1: Divider is working
1	HDIVSTART	Divider start bit 0: Wait for the division operation to start 1: Start the division calculation Writing this register bit triggers the start of the division operation
0	HDIVEN	Divider module enable 0: Close the divider module 1: Enable divider module

24.3.3 HDIV Dividend Register (HDIV_DIVIDEND)

Offset address: 0x04

Reset value: 0x0000 0000

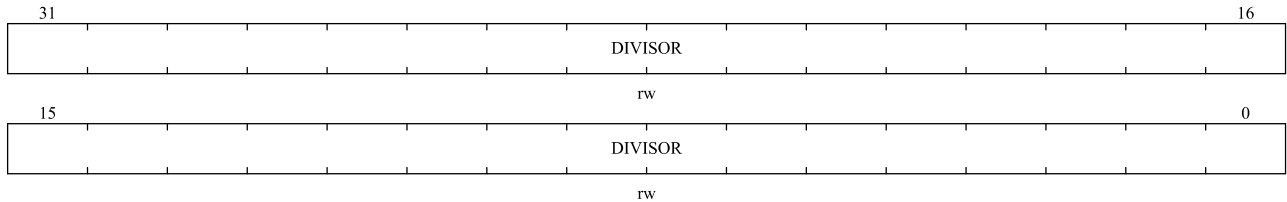


Bit Field	Name	Description
31:0	DIVIDEND	32-bit unsigned integer as dividend

24.3.4 HDIV Divisor Register (HDIV_DIVISOR)

Offset address: 0x08

Reset value: 0x0000 0000

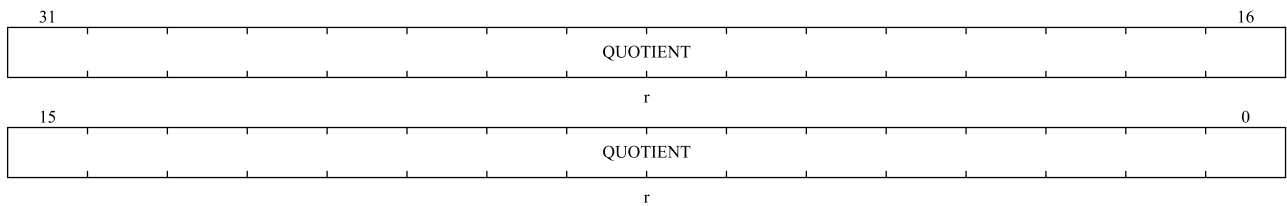


Bit Field	Name	Description
31:0	DIVISOR	32-bit unsigned integer as divisor

24.3.5 HDIV Quotient Register (HDIV_QUOTIENT)

Offset address: 0x0C

Reset value: 0x0000 0000

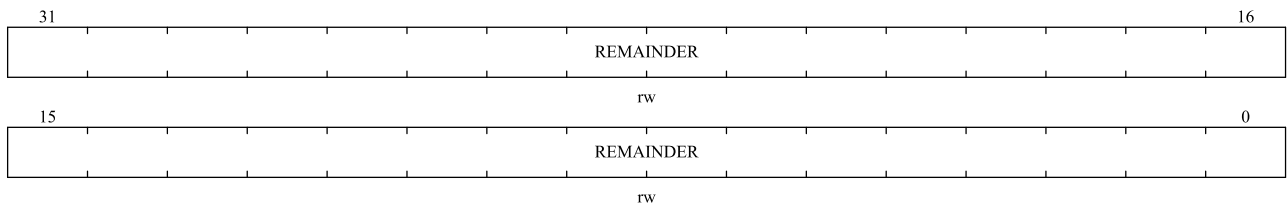


Bit Field	Name	Description
31:0	QUOTIENT	Quotient calculated by the divider

24.3.6 HDIV Remainder Register (HDIV_REMAINDER)

Offset address: 0x10

Reset value: 0x0000 0000

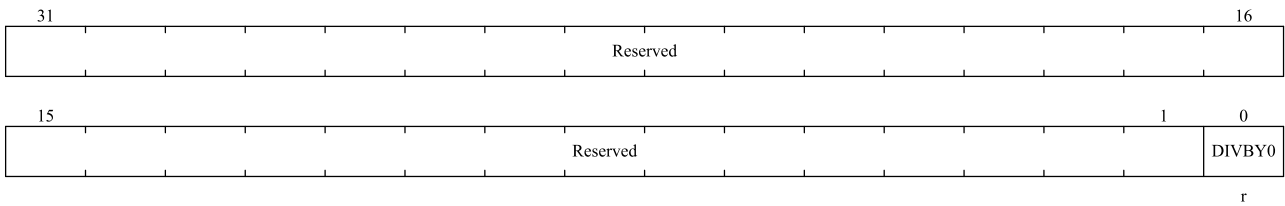


Bit Field	Name	Description
31:0	REMAINDER	Remainder calculated by the divider

24.3.7 HDIV Divide by Zero Register (HDIV_DIVBY0)

Offset address: 0x14

Reset value: 0x0000 0000



Bit Field	Name	Description
31:1	Reserved	Reserved,the reset value must be maintained
0	DIVBY0	Divisor is 0 flag 0: Divisor is not 0 1: Divisor is 0

24.4 Sqrt Registers

24.4.1 Sqrt Register Overview

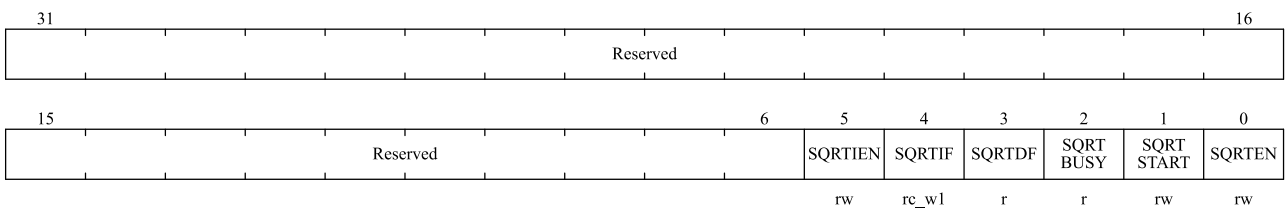
Table 24-2 Sqrt Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																				
000h	SQRT_CTRLSTS	Reserved																								SQRTIEN	SQRTIF	SQRTDF	SQRTBUSY	SQRTSTART	SQRTEN																						
	Reset Value																									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
004h	SQRT_RADICAND	RADICAND																																																			
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																			
008h	SQRT_ROOT	Reserved																ROOT																																			
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																				

24.4.2 Sqrt Control Status Register (SQRT_CTRLSTS)

Offset address: 0x00

Reset value: 0x0000 0000

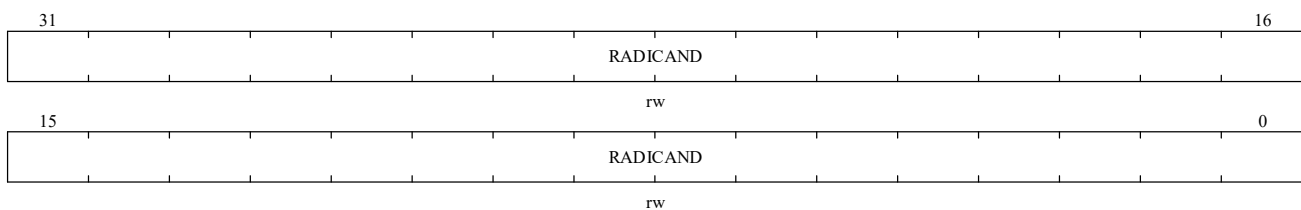


Bit Field	Name	Description
31:6	Reserved	Reserved,the reset value must be maintained
5	SQRTIEN	The square root calculator interrupt enable 0: Disable interrupt 1: Enable interrupt
4	SQRTIF	The square root calculator interrupt flag bit, which generates an interrupt signal when the interrupt enable is turned on 0: No interrupt is generated 1: The square root calculation is completed and an interrupt is generated Software writes '1' to clear the status bit
3	SQRTDF	The square root calculator computes the complete flag bit. 0: The calculation has not ended or not started 1: End of calculation
2	SQRTBUSY	The square root calculator busy flag bit 0: The opening calculator is idle 1: The square root calculator is working
1	SQRTSTART	The square root calculation start bit 0: Waiting for the start of square root calculation 1: Configure 1 to start the square root calculation Write this register bit to trigger the start of the square root operation
0	SQRTEN	The square root calculator module enable: 0: Turn off the square root calculator module 1: Enable the square root calculator module

24.4.3 Sqrt Radicand Register (SQRT_RADICAND)

Offset address: 0x04

Reset value: 0x0000 0000

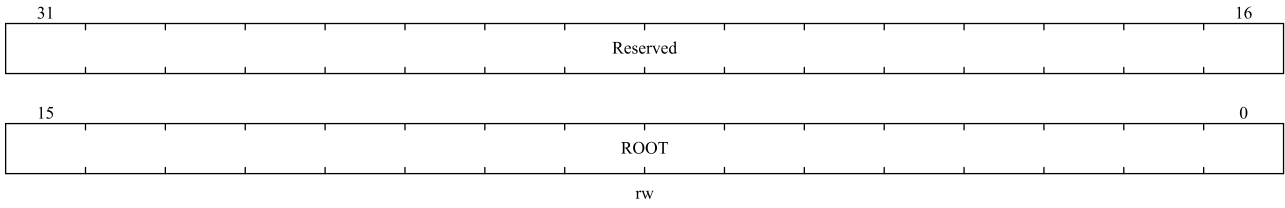


Bit Field	Name	Description
31:0	RADICAND	32-bit unsigned integer as radicand

24.4.4 Sqrt Square Root Register (SQRT_ROOT)

Offset address: 0x08

Reset value: 0x0000 0000



Bit Field	Name	Description
31:16	Reserved	Reserved,the reset value must be maintained
15:0	ROOT	16-bit square root output

25 Controller Area Network (CAN)

25.1 Introduction

As a CAN network interface, the basic extended CAN supports CAN protocols 2.0A and 2.0B. It can efficiently process a large number of received messages and greatly reduce the consumption of CPU resources. The priority characteristics of message transmitting can be configured by software. The hardware function of CAN can support time-triggered communication mode for applications with high security requirements.

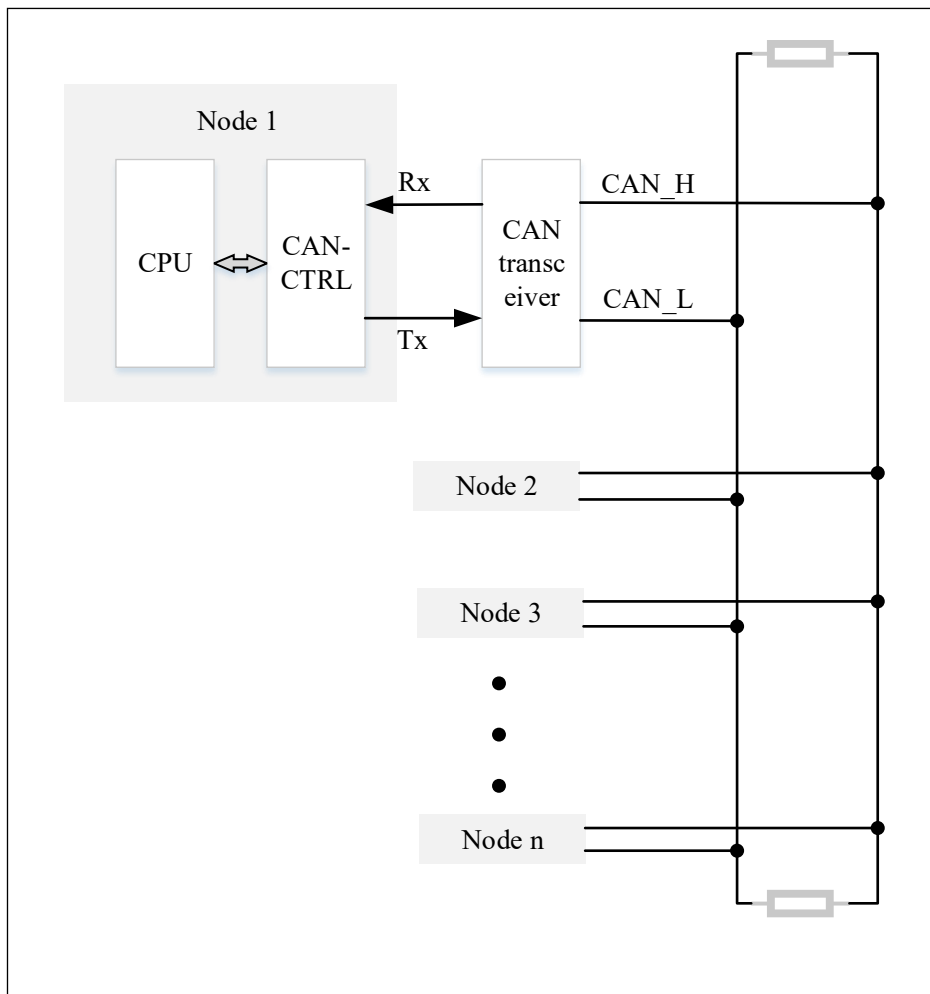
25.2 Main Features

- Support CAN protocol 2.0A and 2.0B active mode
- Baud rate up to 1Mbps
- Supports time-triggered communication
- Transmit :
 - Three transmitting mailboxes
 - The priority of transmitted packets can be configured by software
 - Records the timestamp of the time when the SOF was transmitted
- Receive:
 - Level 3 depth of 2 receiving FIFO
 - Variable filter group
 - There are 14 filter groups
 - Identifier list
 - The FIFO overflow processing mode is configurable
 - Record the time stamp of the receipt of the SOF
- Time-triggered communication mode:
 - Disable automatic retransmission mode
 - 16-bit free run timer
 - Timestamp can be sent in the last 2 bytes of data
- Management:
 - Interrupt masking
 - The mailbox occupies a separate address space to improve software efficiency

25.3 Overview

With the widespread application of CAN, the number of nodes in CAN network is growing rapidly. Multiple CAN nodes are connected through CAN network. With increase number of CAN nodes, the number of messages in CAN network also increase dramatically which will occupy lots of CPU resources. In CAN controller, the addition of receiving FIFOs and filtering mechanisms as hardware support for CPU message processing reduces the real-time response requirement of CAN messages.

Figure 25-1 Topology of CAN Network



25.3.1 CAN Module

The CAN module can automatically receive and transmit CAN messages. It supports standard identifiers (11 bits) and extended identifiers (29 bits).

25.3.2 CAN Operating Mode

Initialization, normal and sleep mode are three main operating modes of CAN. The internal pull-up resistor of CANTX pin is activated after hardware reset. The CAN operates in sleep mode to reduce power consumption.

The software configures CAN to enter initialization or sleep mode by setting CAN_MCTRL.INIRQ and CAN_MCTRL.SLPRQ bits. The software reads values of the CAN_MSTS.INIAK or CAN_MSTS.SLPAK bit to confirm whether the initialization or sleep mode has been entered. The internal pull-up resistor of the CANTX pin is disabled when it has been entered.

CAN is in normal mode when CAN_MSTS.INIAK and CAN_MSTS.SLPAK bits are both '0'. Before entering normal mode, it must synchronize with CAN bus. When 11 consecutive recessive bits are monitored on the CANRX pin, the CAN bus is idle and synchronization is completed.

Normal mode

After the initialization is completed, the software configures CAN to enter normal mode. Clear the CAN_MCTRL.INIRQ

and wait for the hardware to clear the CAN_MSTS.INIRQ bit to confirm that the CAN enters normal mode. Only after the synchronization with CAN bus is completed, the CAN can receive and transmit messages normally.

Setting the bit width and mode of the filter bank must be completed when the filter is in the initialization mode (the CAN_FMC.FINITM bit is 1). Setting the initial value of the filter must be completed when it is inactive (the corresponding CAN_FA1.FAC bit is 0).

Initialization mode

The software can perform initialization configuration only when the CAN is in initialization mode. Setting the CAN_MCTRL.INIRQ bit, clearing CAN_MCTRL.SLPRQ bit and waiting for the hardware to set the CAN_MSTS.INIAK bit to confirm that CAN enters the initialization mode. When entering the initialization mode, the register configuration will not be affected. During the initialization mode, message reception and transmission are prohibited, and the CANTX pin outputs a recessive bit (high level). To exit initialization mode, clearing the CAN_MCTRL.INIRQ bit and waiting for the hardware to clear the CAN_MSTS.INIAK bit to confirm that CAN exits the initialization mode.

To perform initialization configuration for CAN by software, at least the bit time characteristic register (CAN_BTIM) and the control register (CAN_MCTRL) need to be configured. The software needs to set the CAN_FMC.FINITM bit to initialize the filter bank (mode, bit width, FIFO association, activation and filter value) of CAN. Configuring the filter bank of CAN does not necessarily need to be in initialization mode.

Specially, when CAN_FMC.FINITM=1, it is forbidden to receive messages. To modify the value of the corresponding filter, you need to first clear the filter activation bit (in CAN_FA1). It is necessary to keep the unused filter bank inactive (keep its CAN_FA1.FAC bit to '0').

Sleep mode (low power)

To enter the sleep mode, setting the CAN_MCTRL.SLPRQ bit and waiting for the hardware to set the CAN_MSTS.SLPAK bit to confirm that CAN enters the sleep mode. CAN can configure to sleep mode to reduce power consumption when unused. In sleep mode, the clock of CAN stops operating, but the software can still access the transmitting/receiving mailbox register. When CAN is in sleep mode, the CAN_MCTRL.INIRQ bit must be set and the CAN_MCTRL.SLPRQ bit must be clear simultaneously to enter the initialization mode.

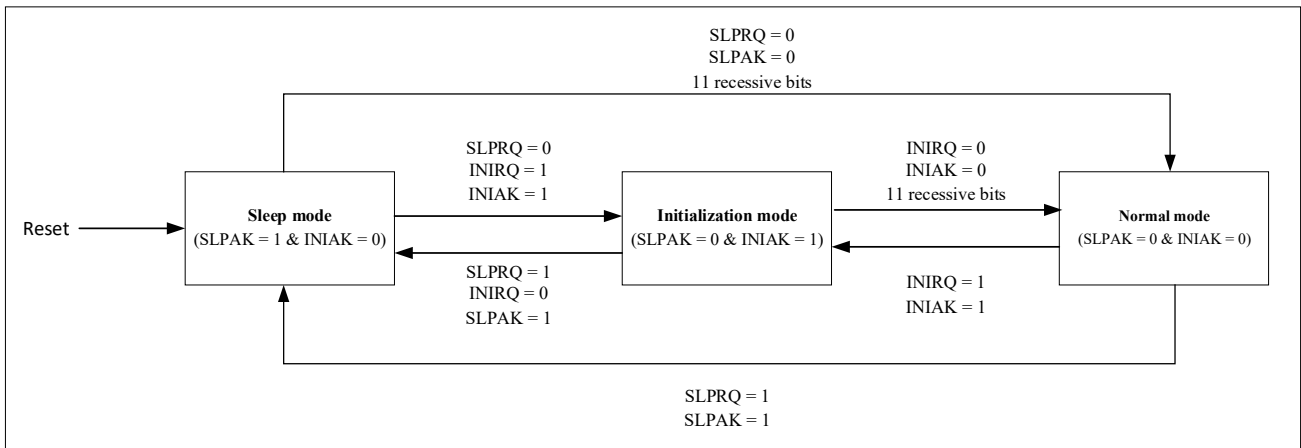
There are two situation to wake up CAN(CAN exits sleep mode):

- When the CAN_MCTRL.AWKUM bit is set(enable hardware wake up automatically), once the activity of the CAN bus is detected, the hardware will automatically clear the CAN_MSTS.SLPRQ bit to wake up CAN.
- When the CAN_MCTRL.AWKUM bit is clear(enable software wake up) and the wake-up interrupt occurred, the software must clear the CAN_MCTRL.SLPRQ bit to exit the sleep mode.

If the wake-up interrupt (set the CAN_INTE.WKUIE bit) is enabled, the wake-up interrupt will be generated once the CAN bus activity is detected, regardless of whether the hardware is enabled to automatically wake up CAN.

Before entering normal mode, the CAN must be synchronized with the CAN bus until the CAN_MSTS.SLPAK bit cleared to confirm the sleep mode has exited. Please refer to Figure 25-2.

Figure 25-2 CAN Operating Mode



Note: The hardware sets the CAN_MSTS.INIAK or CAN_MSTS.SLPK bit to respond to the sleep or initialization request.

25.3.3 Transmit Mailbox

Applications can transmit messages through three transmit mailboxes. The order of transmit three mailbox messages is determined by the transmission scheduler according to the priority of the messages. The priority can be determined by the identifier of the messages or by the order of transmitting requests.

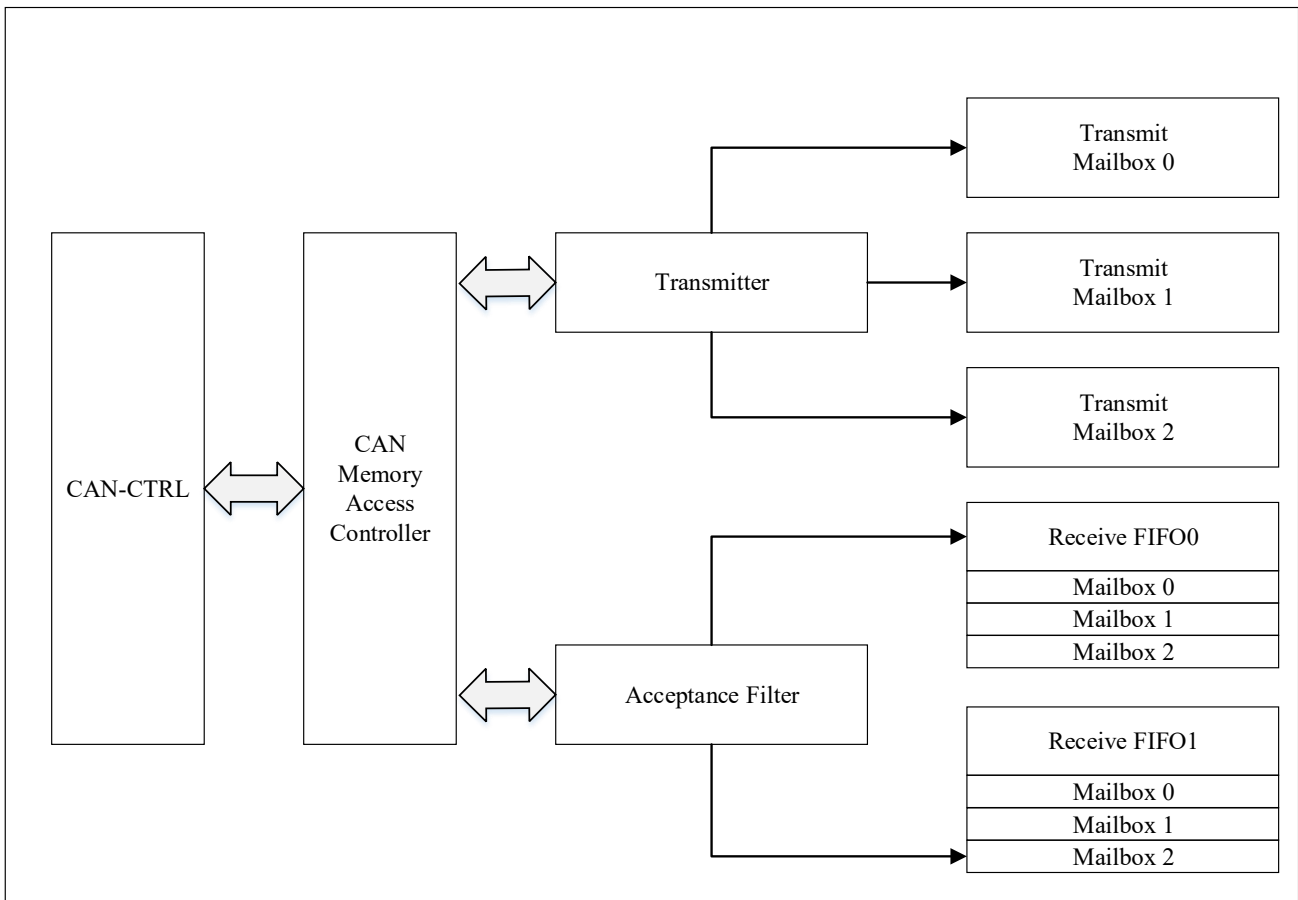
25.3.4 Receive Filter

The CAN has 14 configurable identifier filter banks. After the application configures the identifier filter bank, the receiving mailbox will automatically receive the required messages and discard other messages.

25.3.5 Receive FIFO

CAN has two receiving FIFOs, each of which can store three complete messages. No application program is needed to manage it, and it is managed by hardware.

Figure 25-3 Single CAN Block Diagram



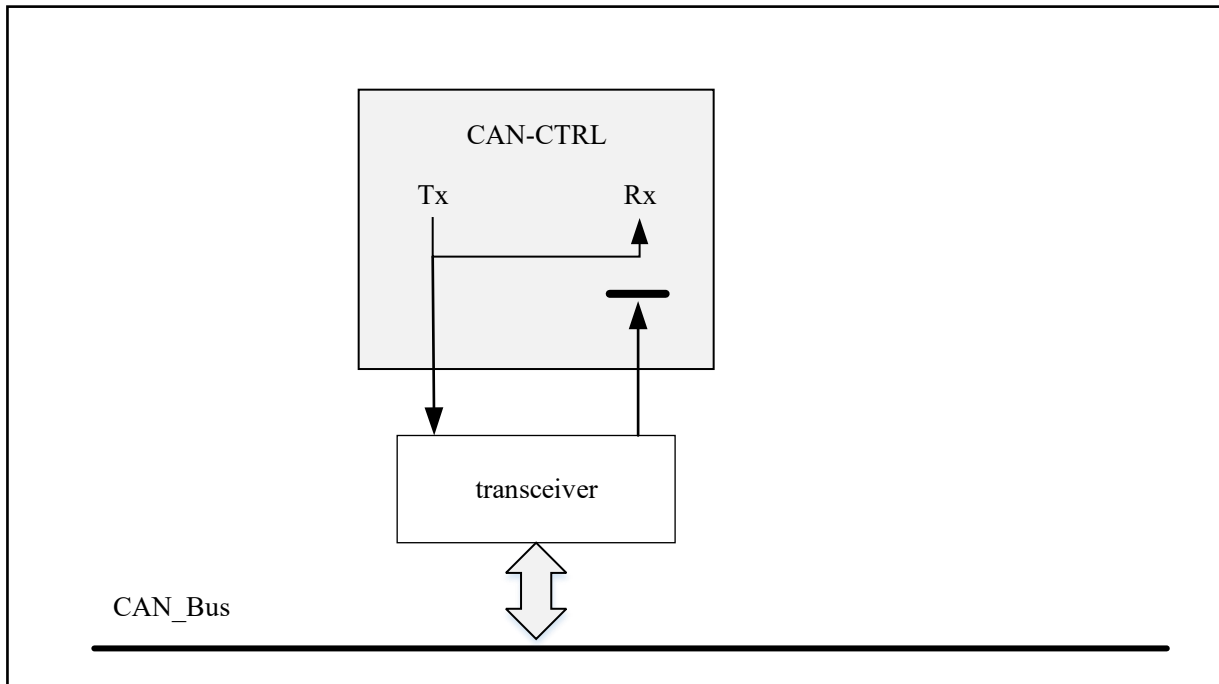
25.3.6 CAN Test Mode

In the initialization mode, the test mode must be selected by combining the CAN_BTIM.SLM bit and CAN_BTIM.LBM bit. After selecting the test mode, the software needs to clear the CAN_MCTRL.INIRQ bit to exit the initialization mode and enter the test mode.

Loopback mode

Loopback mode can be used for self-test. In loopback mode, the CAN stores the transmitted message in the receiving mailbox as received message (if it can be filtered by reception). In loopback mode, CAN internally feeds back the Tx output to the Rx input, completely ignoring the actual state of the CANRX pin. The message transmitted can be detected on the CANTX pin. To avoid external influence, the CAN kernel ignores the acknowledgement error(at the moment of acknowledgement bit of data/remote frame, it does not detect whether there is an dominant bit). To enter loopback mode, the CAN_BTIM.SLM bit should be cleared and the CAN_BTIM.LBM bit should be set.

Figure 25-4 Loopback Mode

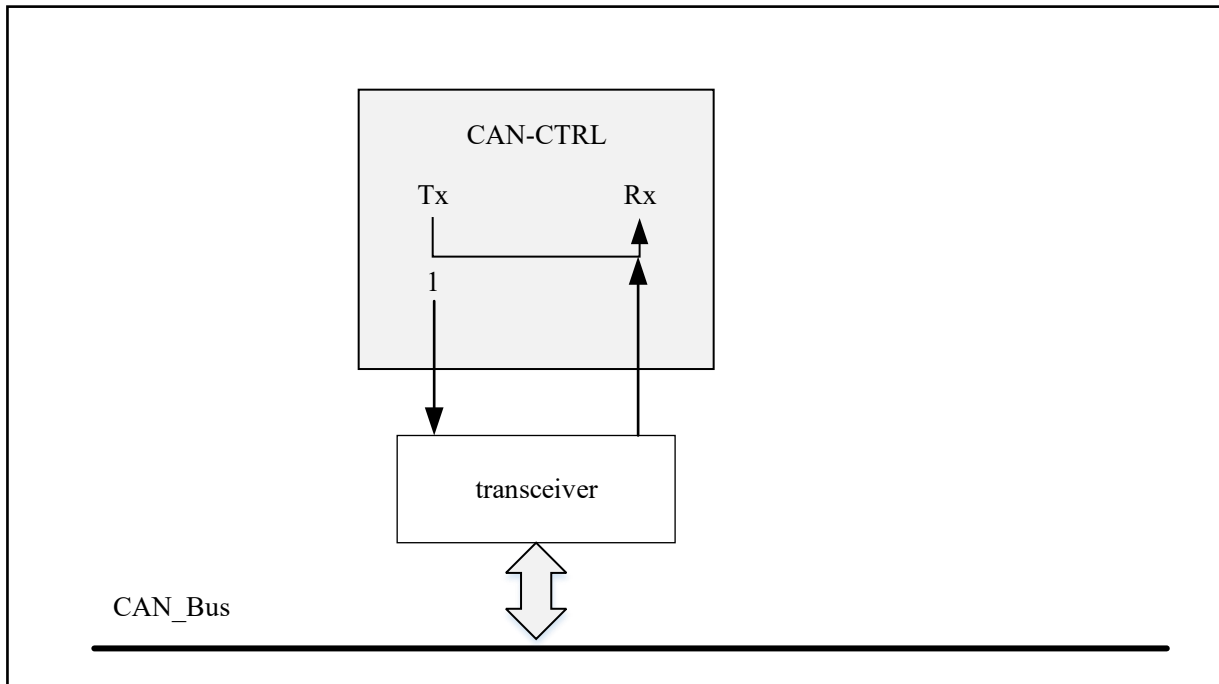


Silent mode

In silent mode, the CAN can normally receive data frames and remote frames, but can only transmit recessive bits, and can't really transmit messages. If CAN needs to transmit overload flag, active error flag or ACK bit (these are dominant bits), such dominant bits are internally connected back to be detected by the CAN core. At the same time, the CAN bus will not be affected and still remain in the recessive bit state. Therefore, the silent mode is usually used to analyze the activity of the CAN bus without affecting the bus, because dominant bits is not actually transmit to the bus.

To enter silent mode, the CAN_BTIM.SLM bit should be set and the CAN_BTIM.LBM bit should be cleared.

Figure 25-5 Silent Mode

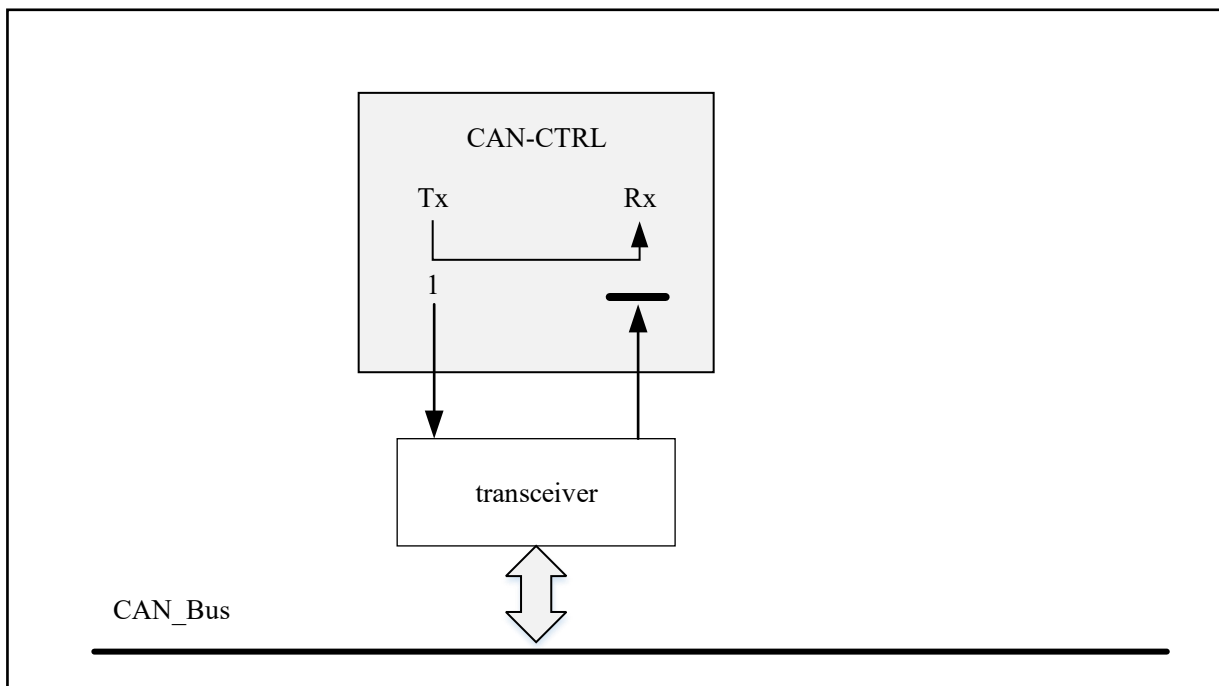


Loopback silence mode

In loopback silent mode, the CANRX pin is disconnected from the CAN bus, while the CANTX pin is driven to the recessive bit state. It can be used for "Hot Selftest", meaning CAN can be tested like in loop-back mode but without affecting the CAN system connected to the CANTX and CANRX pins.

To enter loopback silence mode, both the CAN_BTIM.SLM bit and the CAN_BTIM.LBM bit should be set.

Figure 25-6 Loopback Silent Mode



25.3.7 CAN Debug Mode

The CAN can continue to operate normally or stop operating according to the state of the following configuration bits:

- The DBG_CTRL.CAN_STOP bit of CAN in the debug support(DBG) module. Refer to paragraph 3.3.2 Section: Peripheral debugging support.
- The CAN_MCTRL.DBGF bit refer to paragraph 0 Section: CAN_MCTRL.

When the microcontroller is in debug mode, Cortex[®]-M4F core is in a suspended state.

25.4 CAN Function Description

25.4.1 Transmit Processing

The process of transmitting messages is as follows:

1. The application program selects an empty transmit mailbox;
2. Writes the identifier, data length and data to be transmitted into the transmit mailbox register;
3. Set the CAN_TMIx.TXRQ bit to request transmission(after CAN_TMIx.TXRQ is set, the mailbox is no longer an empty mailbox and the software has no permission to write to the mailbox register);
4. The mailbox enter the pending state and wait to be the highest priority, refer to Section 0 Transmit priority;
5. Changing to the ready status once the mailbox becomes the highest priority mailbox;
6. The messages in the ready mailbox is transmitted as soon as the CAN bus enters the idle state, then enter the transmitting state.
7. Become an empty mailbox when the message in the mailbox is successfully transmitted;
8. Hardware set the RQCPM and TXOKM bits of the corresponding mailbox in CAN_TSTS register to indicate a successful transmission.

However, if the transmission fails, the CAN_TSTS.ALSTM bit will be set to indicate that the failure is caused by arbitration or the CAN_TSTS.TERRM bit will be set to indicates that it is caused by the transmission error (for specific errors, please check the CAN_ESTS.LEC[2:0] error code bits of the error status).

transmit priority

Determined by the order of transmit requests.

Setting the CAN_MCTRL.TXFP bit, and the transmit mailbox can be configured as a transmit FIFO. At this time, the priority of transmission is determined by the order of transmit requests. This mode is useful for segmented transmission.

Determined by the identifier.

According to CAN protocol, the message with the lowest identifier has the highest priority. If the values of identifiers are equal, the message with small mailbox number is transmitted first. When more than one transmit mailbox is registered, the transmit order is determined by the identifier of the message in the mailbox.

Canceling transmitting

Setting the CAN_TSTS.ABRQM bit can abort transmit the request. If the mailbox is ready or pending, the transmitting request will be aborted immediately. If the mailbox is in the transmitting state, aborting the request may lead to two kinds

of results:

- if the message transmission fails in the mailbox, the mailbox becomes ready state, then the transmitting request is aborted, the mailbox becomes an empty mailbox and the CAN_TSTS.TXOKM bit is cleared;
- if the message transmission successfully in the mailbox, the mailbox becomes an empty mailbox, and the CAN_TSTS.TXOKM bit will be set by hardware.

Therefore, when the transmitting mailbox is in the transmitting state, regardless of the transmitting result, the mailbox will become an empty mailbox after the transmitting operation is finished.

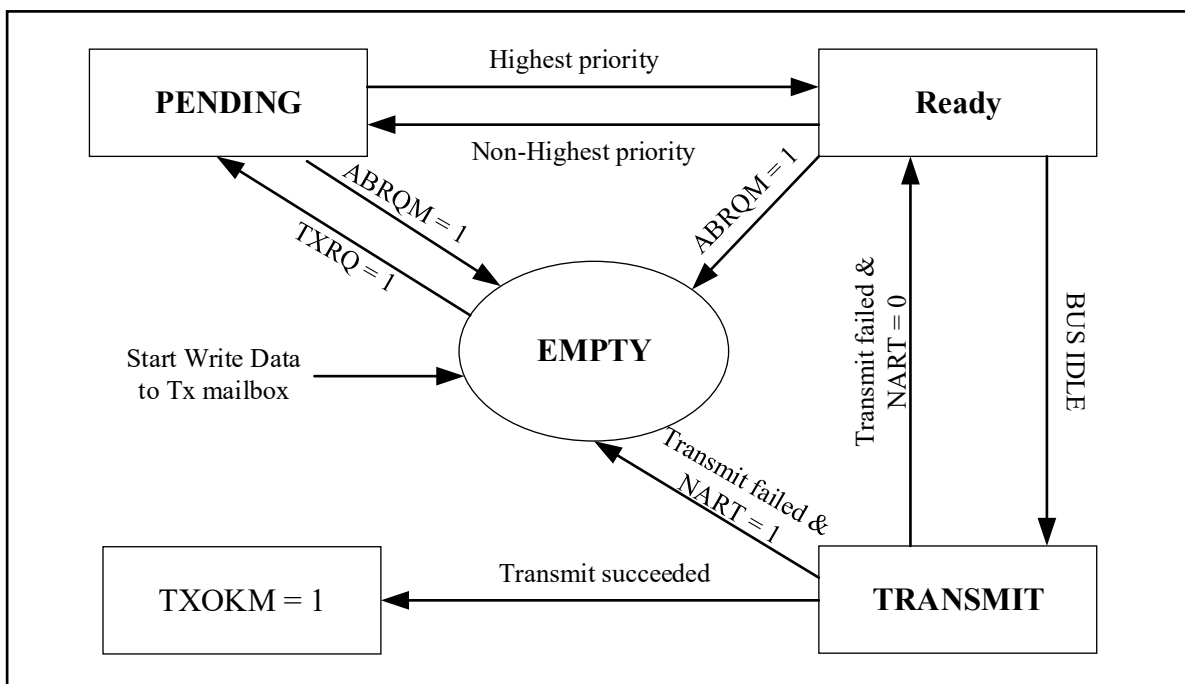
25.4.2 Time Triggered Communication Mode

The internal timer of CAN is activated in time triggered communication mode. It is incremented at each CAN bit time (refer to Section 25.4.7). CAN samples the value of the internal timer at the sampling point position of the received and transmitted frame start bits, and the sampled value is the time stamp of the transmitting and receiving mailboxes. Timestamps generate from the internal timer will be stored in the CAN_RMDTx/CAN_TMDTx registers respectively.

25.4.3 Non-automatic Retransmission Mode

To enable non-automatic retransmission mode, the CAN_MCTRL.NART bit should be set. This mode corresponds to the function of time-triggered communication option in CAN standard. In non-automatic retransmission mode, the transmission will only be executed once. If the transmission fails, due to arbitration loss or error, the hardware will not automatically transmit the message again. At the end of a transmission operation, the hardware judge that the transmission request has been completed and the hardware sets the CAN_TSTS.RQCPM bit. At the same time, the transmission result can query the CAN_TSTS.TXOKM, CAN_TSTS.ALSTM and CAN_TSTS.TERRM bits.

Figure 25-7 Transmit Mailbox Status



25.4.4 Receive Management

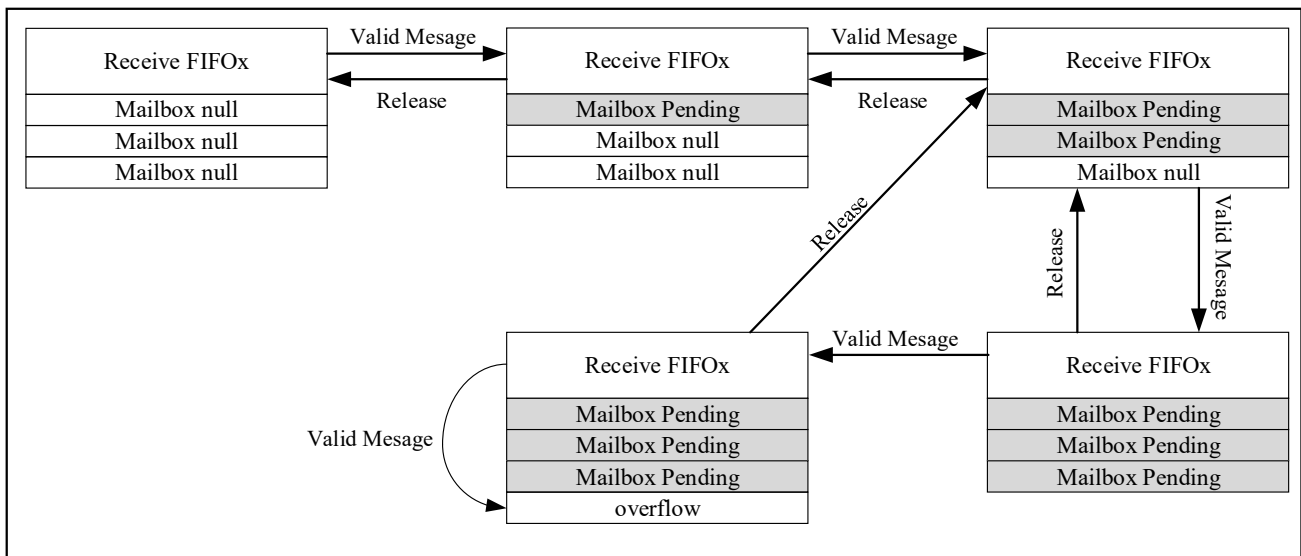
FIFOs with 3 levels depth are used to store received messages. When the application reads the FIFO output mailbox, it

reads the first received message in the FIFO. FIFO is completely managed by hardware, which can simplify the application program, ensure the consistency of data and reduce the processing time of CPU.

Valid messages

A message is considered as valid when the message has been correctly received according to CAN protocol (no errors until the last bit of the EOF field) and it passes through the identifier filterings. Please refer to 25.4.5 Section: identifier filtering.

Figure 25-8 Receive FIFO States



FIFO receive

A FIFO includes mailboxes with three levels of depth, and the initial state is empty. After receiving the first valid message, one of the mailboxes will be suspended and the hardware will set the CAN_RFF.FFMP[1:0] bits to 1 to indicate the receipt of a valid message. A valid message is received again before the mailbox is released. At this time, two mailboxes will be suspended at the same time, and the hardware will set the CAN_RFF.FFMP[1:0] bits to 2 to indicate that two valid messages are pending. As above, the third valid message will suspend all three mailboxes and set the CAN_RFF.FFMP[1:0] bits to 3.

When the three-level mailboxes of the FIFO are all suspended, receiving a valid message again will cause the mailbox to overflow and lose a message. The hardware will set the CAN_RFF.FFOVR bit to 1 to indicate the occurrence of the event. The rules for lost messages depend on the configuration of the FIFO. If the FIFO lock function is disabled (clear the CAN_MCTRL.RFLM bit), then the last message received in the FIFO will be overwritten by the new message. In this way, the latest received message will not be discarded; If the FIFO lock function is enabled (set the CAN_MCTRL.RFLM bit), then the newly received messages will be discarded, and the software can read the first three messages in the FIFO.

FIFO release

The message stored in the FIFO will be read through the corresponding receive mailbox. The software reads the mailbox message and releases the mailbox by setting the CAN_RFR.RFOM bit to 1, and the CAN_RFF.FFMP[1:0] bit is decremented by 1 until it is 0.

Receive related interrupts

The hardware will update the CAN_RFF.FFMP[1:0] bits when a message is stored in the receiving FIFO. If the FIFO message pending interrupt is currently enabled (the CAN_INTE.FMPITE bit is set), the FIFO message pending interrupt

request will be generated.

When the third message is stored, the FIFO becomes full, then the CAN_RFF.FFULL bit will be set. If the FIFO full interrupt is currently enabled (the CAN_INTE.FFITE bit is set), a FIFO full interrupt request will be generated.

When the FIFO overrun, the FFOVR bit will be set. If the FIFO overrun interrupt is currently enabled (the CAN_INTE.FOVITE bit is set), a FIFO overrun interrupt request will be generated.

25.4.5 Identifier Filtering

In the CAN network, when the CAN is in the transmitter state, it broadcasts a message to each node by transmitting a message to the bus; when the CAN is in the receiver state, it determines whether the node needs the message according to the identifier of the message after receiving the message. If message is valid, CAN core copies the message to SRAM(CAN's own SRAM); If it is not needed, the message will be discarded. This process does not require software intervention. Compared with software filtering, hardware filtering reduces the CPU usage. CAN controller provides 14 configurable filter banks (0~13) with variable bit width for application programs to meet this demand. These filter banks are used to receive only those messages needed by software. Each filter bank x contains two 32-bit registers, namely CAN_FxR0 and CAN_FxR1.

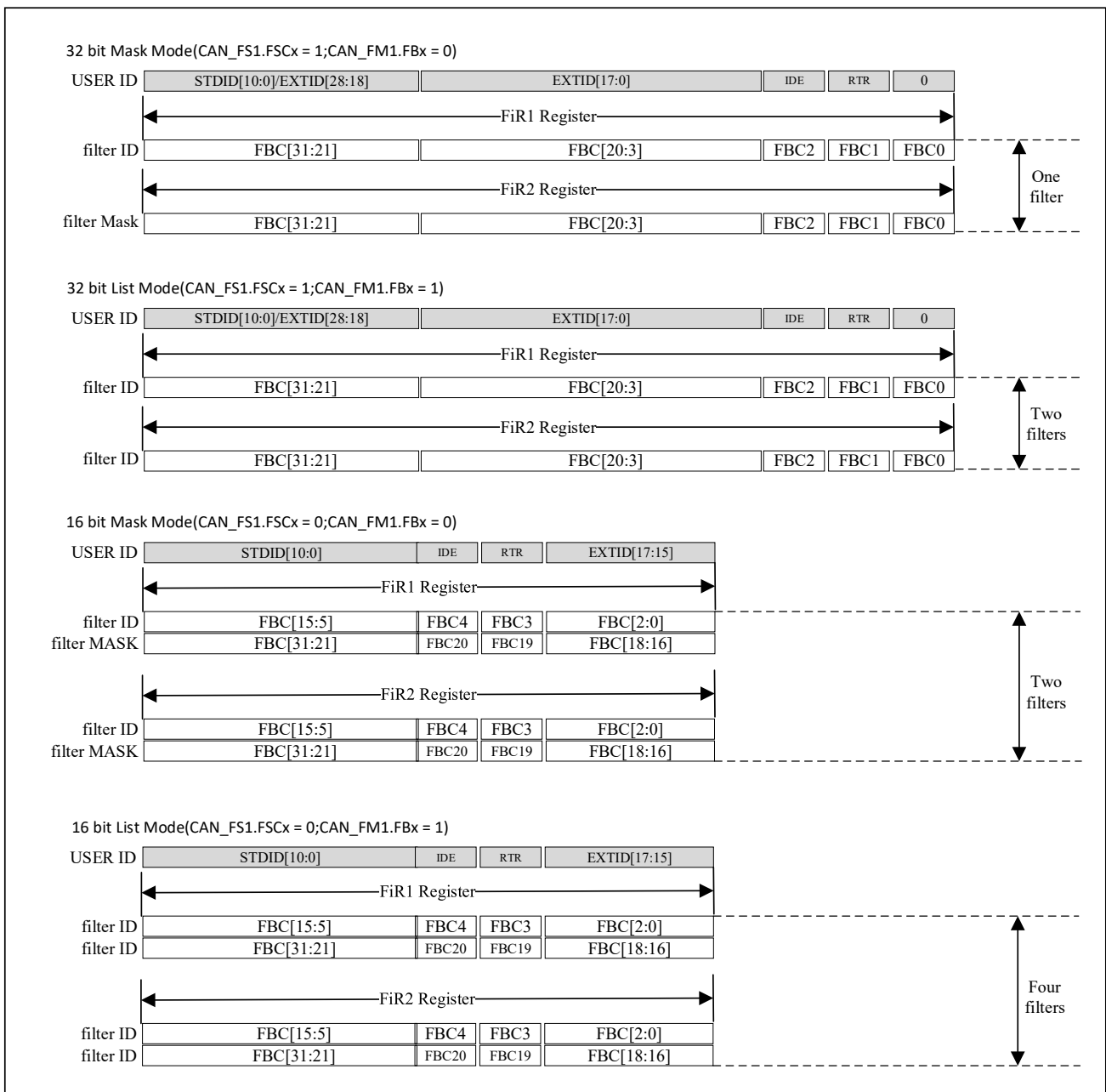
Setting of filter bit width and mode

Each filter in a filter bank is numbered (filter number, from 0 to a certain maximum value) depending on the mode and bit width setting of the filter bank. Refer to the figure below for the filter configuration. The filter bank can be configured through the corresponding CAN_FMC register. Filter banks that are not used by the application should be kept disabled. Before configuring the filter bank, it must be set to the disabled state by clearing the CAN_FA1.FAC bit.

By setting the corresponding CAN_FS1.FSCx bit you can configure the bit width of a filter bank, refer to Figure 25-9.

The corresponding mask/identifier register can be configured to be the identifier list or identifier mask mode by the CAN_FM1.FBx bit. The filter bank should be set to operation in the mask mode in order to filter out a bank of identifiers. And the filter bank should be set to operation in identifier list mode in order to filter out an identifier.

Figure 25-9 Filter Bit Width Setting-register Organization



Variable bit width

The bit width of each filter bank can be independently configured. Each filter bank can be configured to be one 32-bit filter: including STDID[10:0], EXTID[17:0], IDE and RTRQ bits; or two 16-bit filters, including STDID[10:0], IDE, RTRQ and EXTID[17:15] bits, refer to Figure 25-9. In addition, the filter can be configured in two different modes, namely, the mask mode and the identifier list mode.

Mask mode

The filter ID is used to store the identifier format, and the filter mask is used to indicate which bits must be checked and which bits can be ignored.

Identifier list mode

The filter ID is used to store the identifier format. At this time, there is no mask for comparison, and the mask bit can be

used to store one more filter ID. However, the identifier of the message needs to be exactly the same as the filter ID format, otherwise it will fail to pass the filter.

Filter match index

After CAN core receives a valid message it will matching the message ID with filters one by one until there is one filter pass or all filters failed. If this message failed to pass any enabled filter then it will be discarded. Otherwise when CAN core finds the first filter that the ID can pass, it pack filter index with the CAN message and stores inside receive FIFO in SRAM according to filter setting (CAN_FFA1 decides store in which FIFO). User can find filter index in FMI [7:0] bits of CAN_RMDTx register. This filter matching index can help to identify which types of message it is in this receive FIFO.

The filter match index can be used two ways. The first one is comparing the filter match index with a series of expected values. The another is using the filter match index as an index to access the target address. When numbering filters, the activation state of the filter bank is not considered. In addition, each FIFO numbers its associated filter, please refer to the example below.

For the filter in mask mode, the software only needs to compare the mask bits that are needed (bits that must be matched). For the filter in identifier list mode (non-screening filter), the software does not need to directly compare with the identifier.

Table 25-1 Examples of Filter Numbering

Point to FIFOx	Filter Group	Filter Mode	FIFO0 Filter Number
FIFO	0	32 bit mask mode	0
	2	16 bit mask mode	1/2
	5	32 bit list mode	3/4
	7	16 bit list mode	5/6/7/8
	9	32 bit list mode	9/10
	11	16 bit list mode	11/12/13/14
	13	32 bit mask mode	15
Point to FIFOx	Filter Group	Filter Mode	FIFO1 Filter Number
FIFO1	1	32 bit list mode	0/1
	3	16 bit list mode	2/3/4/5
	4	32 bit mask mode	6
	6	16 bit mask mode	7/8
	8	32 bit mask mode	9
	10	16 bit mask mode	10/11

	12	32 bit list mode	12/13
--	----	------------------	-------

Filter priority rules

According to different configurations of filters, it is possible that a message identifier can be filtered by multiple filters; In this case, the filter matching index stored in the receiving mailbox is first determined according to bit width, 32-bit-wide filters have higher priority than 16-bit-wide filters. For filters with the same bit width, the identifier list mode takes precedence over the mask mode. If filters have the same bit width and mode, the priority is determined by the filter bank number, the filter bank with lower number has the higher priority. Within a filter bank, the lower the filter number, the higher the priority.

Figure 25-10 Examples of Filter Mechanisms

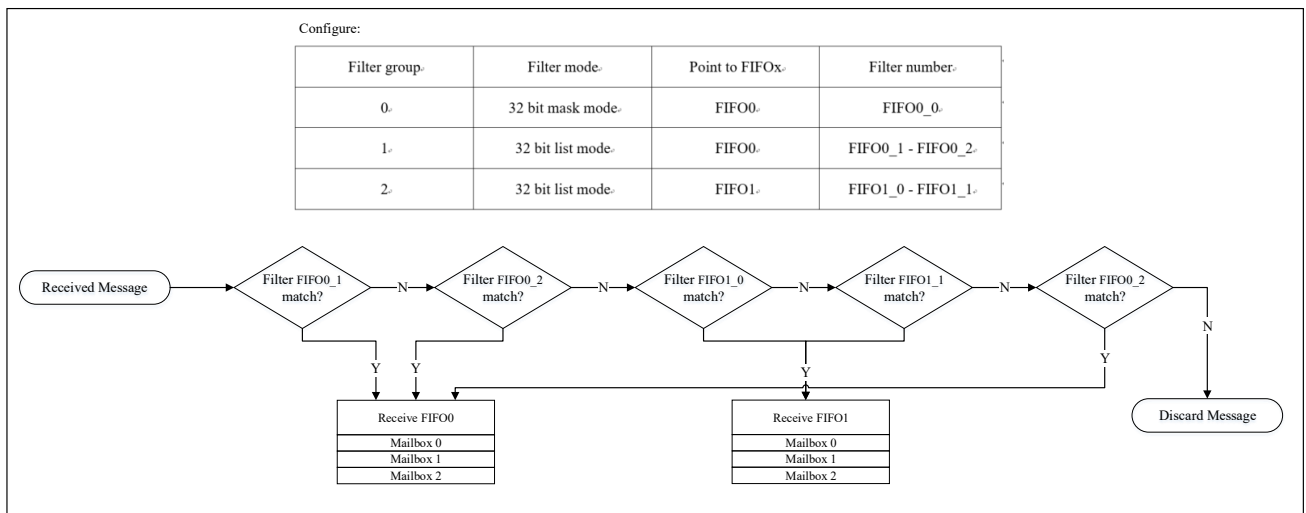


Figure 25-10 illustrates the filter rules of CAN. When receiving a message, its identifier is first compared with the filter configured in identifier list mode. If there is a match, the message will be stored in the associated FIFO, and the index of the matched filter will be stored in the filter matching index.

If there is no match, the message identifier is compared with the filter configured in the mask mode. And the hardware will automatically discard the message without software intervention if the message identifier does not match any identifier in the filter.

25.4.6 Message Storage

A mailbox contains all information related to a message: identifier, data, control, status and time stamp information. The mailbox is the interface between software and hardware for transmission messages.

Transmit mailbox

Message should be written into an empty transmit mailbox by software before enabling the transmission request. You can query the transmission status through the CAN_TSTS register.

Table 25-2 Transmit Mailbox Register List

Offset from The Base Address of The Sending Mailbox	Register Name
0	CAN_TMIx
4	CAN_TMDTx

8	CAN_TMDLx
12	CAN_TMDHx

Receive mailbox (FIFO)

The CAN_RMDTx.FMI[7:0] field can store the filter matching index and CAN_RMDTx.MTIM[15:0] field can store the 16-bit timestamp. The software can access the output mailbox of the receiving FIFO to read the received message. Once the software has processed the message such as reading it out, the software should set the CAN_RFFx.RFFOM bit to release the corresponding receive FIFO to reserve storage space for later messages.

Table 25-3 Receive Mailbox Register List

Offset from The Base Address of The Receiving Mailbox	Register Name
0	CAN_RMIx
4	CAN_RMDTx
8	CAN_RMDLx
12	CAN_RMDHx

25.4.7 Bit Timing

The bit timing logic monitors the serial CAN bus by sampling, and adjusts its sampling point by synchronizing with the edge of the frame start bit and resynchronizing with the following edge. To avoid programming errors in software, setting the bit timing register (CAN_BTIM) can only be done when the CAN is in initialization mode.

Its operation can be simply understood as dividing each bit timing into three segments: synchronization segment (SYNC_SEG), time period 1(BS1) and time period 2(BS2).

Usually, the change of bits will occur in SYNC_SEG. Its value is fixed to 1 time unit ($1 \times t_{CAN}$).

BS1 defines the position of the sample point. It includes the PROP_SEG and PHASE_SEG1 of the CAN standard. Its duration is programmable between 1 and 16 time quanta but may be automatically lengthened to compensate for positive phase drifts due to differences in the frequency of the various nodes of the network

In BS2, it defines the location of the transmit point. It represents the PHASE_SEG2 of CAN standard. Its duration is programmable between 1 and 8 time quanta but may also be automatically shortened to compensate for negative phase drifts

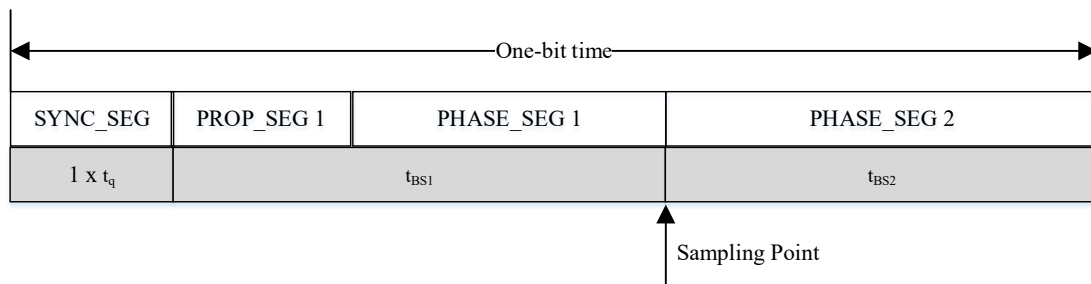
If a valid transition is detected in BS1 but not in SYNC_SEG, BS1 is extended by up to RSJW so that the sample point is delayed. On the contrary, if a valid transition is detected in BS2 but not in SYNC_SEG, BS2 is shortened by up to RSJW so that the sample point is moved earlier.

In the above description, RSJW (the resynchronization hop width) defines the upper limit of how many time quantas can be extended or shortened in each bit. Its value can be programmed into 1 to 4 time quanta. The effective transition is defined as the first transition from the dominant bit to the recessive bit when CAN itself does not transmit the recessive bit.

Note:

- (1) *The bit timing characteristic and resynchronization mechanism of CAN bits are details in ISO11898 standard.*
- (2) *In order to improve the CAN bit time accuracy, it is not recommended to use HSI as the clock source.*

Figure 25-11 Bit Timing



Notes:

$$t_{PCLK} = \text{time period of the APB1 clock,}$$

$$t_q = (CAN_BTIM.BRTP[9:0] + 1) \times t_{PCLK},$$

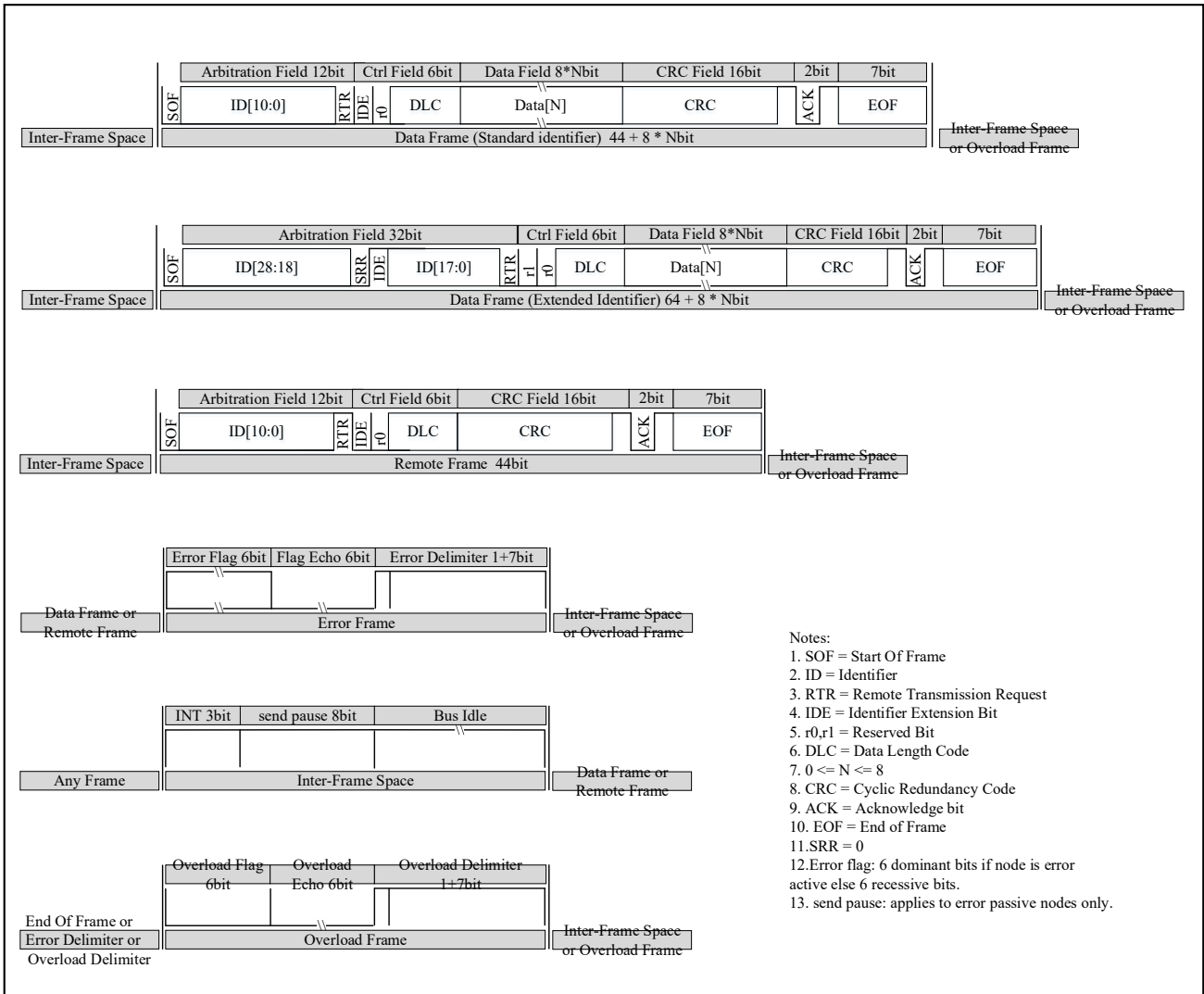
$$t_{BS1} = t_q \times (CAN_BTIM.TBS1[3:0] + 1),$$

$$t_{BS2} = t_q \times (CAN_BTIM.TBS2[2:0] + 1),$$

$$\text{One-bit time} = 1 \times t_q + t_{BS1} + t_{BS2}$$

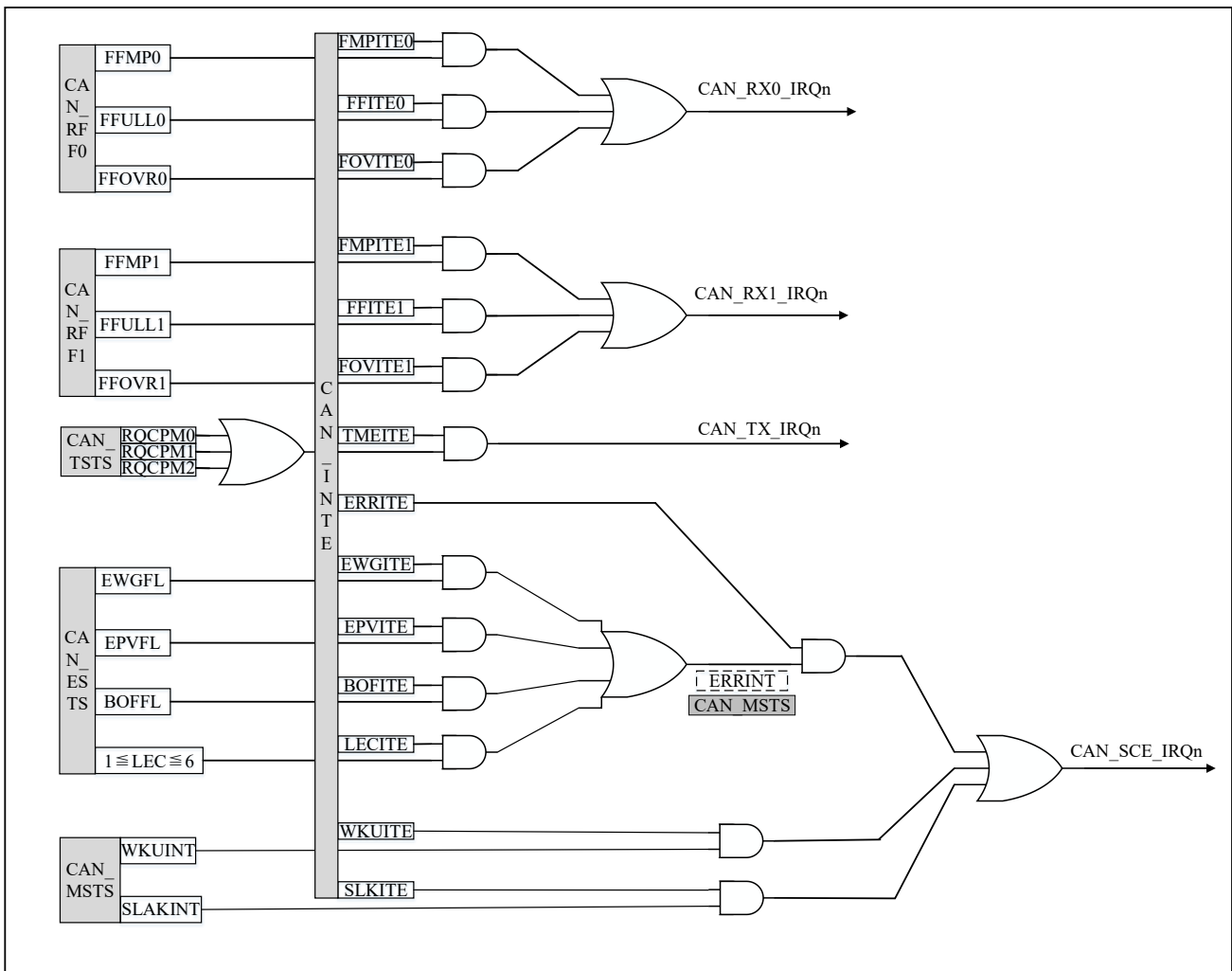
$$\text{BaudRate} = \frac{1}{\text{One-bit time}}$$

Figure 25-12 Various CAN Frames



25.5 CAN Interrupt

Figure 25-13 Event Flag and Interrupt Generation



CAN has four interrupt vectors. By setting the CAN interrupt enable register (CAN_INTE), you can individually enable or disable each interrupt source. The following are the events that can generate each interrupt.

- FIFO0 interrupt(CAN_RX0_IRQn):

FIFO0 receives a new message, and the CAN_RFF0.FFMP0 bit is not '00' ;

When FIFO0 becomes full, and the CAN_RFF0.FFULL0 bit is set;

When FIFO0 overruns, and the CAN_RFF0.FFOVR0 bit is set.
- FIFO1 interrupt(CAN_RX1_IRQn):

FIFO1 receive a new message, and the CAN_RFF1.FFMP1 bit is not '00'.

When FIFO1 becomes full, and the CAN_RFF1.FFULL1 bit is set.

When FIFO1 overruns, and the CAN_RFF1.FFOVR1 bit is set.
- Transmit interrupts(CAN_TX_IRQn):

Transmit mailbox x becomes empty, and the corresponding CAN_TSTS.RQCPMx bit is set(x=1/2/3).

- Error and status change interrupt(CAN_SCE_IRQn):

CAN enters sleep mode;

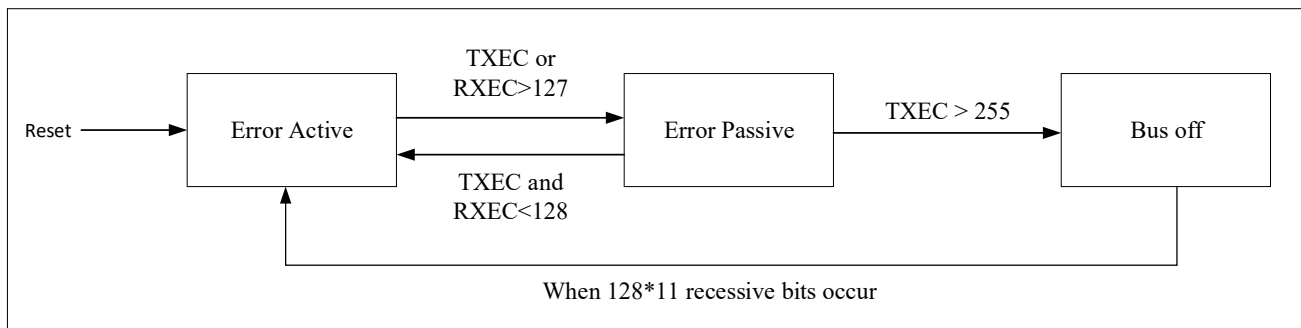
Wake-up condition, the start of frame bit (SOF) is monitored on the CAN receiving pin.

Error condition, please refer to the CAN error status register (CAN_ESTS) for details of the error.

25.5.1 Error Management

As described in CAN protocol, the error management is completely handled by hardware through transmitting error counter (CAN_ESTS.TXEC field) and reception error counter (CAN_ESTS.RXEC field). The counter value will increase or decrease according to the error situation. Please refer to CAN standard if you want to know more detailed information about CAN_ESTS.TXEC and CAN_ESTS.RXEC management.

Figure 25-14 CAN Error State Diagram



Software can read out the value of the transmission/reception error counter to judge the stability of CAN network, and CAN_ESTS.LEC[2:0] bits can be read to get the detailed information of the current error status. What’s more, by setting the CAN_INTE register (such as CAN_INTE.LECITE bit), the software can flexibly control the generation interrupts when an error is detected.

25.5.2 Bus-off Recovery

When TXEC is greater than 255, the CAN_ESTS.BOFFL bit is set indicating that CAN goes bus-off. at this time, CAN can't receive and transmit messages.

In normal mode, according to the CAN_MCTRL.ABOM bit, CAN recover from bus-off state and change to error active state either automatically or on software request. If the CAN_MCTRL.ABOM bit is set, the recovery process will be started automatically after it has entered bus-off state. Otherwise, the recovery process will be started after software must request CAN to enter and then exit initialization mode. In both cases, CAN must wait for a recovery process described of CAN standard (128*11 consecutive recessive bits are detected on CAN RX pin).

In initialization mode, CAN will not monitor the state of CAN RX pin, so the recovery process cannot be completed.

25.6 Can Configuration Process

This chapter will introduce common configuration procedure of CAN, other details like functions of each mode and register bits are revealed in other part of this manual. CAN configuration process can divided into several phases. Some of the configurations can be changed anytime as long as prior requirements are satisfied (e.g., filter value).

- Preparation Stage:
 1. Configure RCC to enable CAN clock
 2. Configure RCC to enable AFIO and GPIO clock
 3. Write GPIO registers to map CAN TX and CAN RX signals to desired GPIO pins.
- Basic Configuration Stage:
 1. CAN device starts with sleep mode after reset.
 2. Exit sleep mode by clearing CAN_MCTRL.SLPRQ bit.
 3. Enter initialization mode by setting CAN_MCTRL.INIRQ bit.
 4. Wait for CAN_MSTS.INIAK bit become 1 (enter initialization mode).
 5. Configure bit timing for CAN by writing value to CAN_BTIM.BSJW, CAN_BTIM.TBS2, CAN_BTIM.TBS1 and CAN_BTIM.BRTP bits. Baud rate of CAN bus is defined by the formula below:

$$BaudRate = \frac{1}{(1 + (TBS1 + 1) + (TBS2 + 1)) * (BRTP * t_{clk})}$$

6. Configure operation mode options for CAN by writing to CAN_BTIM.SLM (silent) or CAN_BTIM.LBM in register.
 7. Configure CAN behavior(TTCM,ABOM,AWKUM,NART,RFLM,TXFP) through CAN_MCTRL. Most of the configuration in this register can be changed on the fly but its advised not to do so. Otherwise during few cycles, CAN behavior will become unpredictable.
 8. Exit Initialization mode by clearing CAN_MCTRL.INIRQ bit.
 9. Wait for CAN_MSTS.INIAK bit become 0 (exit initialization mode).
 10. User can use filters to filter the messages they want to receive. To configure filter, user needs to write '1' to CAN_FMC.FINITM bit to request the filters to enter initialization mode. When filter is in initialization mode, CAN stops reception.
 11. Configure the operating mode (CAN_FM1), filter scale (CAN_FS1) and filter assignment (CAN_FFA1) for each filter. User can also change filter value (CAN_FiRx) during this time. After completing filter configuration, clear CAN_FMC.FINITM bit to exit initialization for filters.
- For transmission:
 1. Enable the necessary transmit related interrupt CAN_INTE.TMEITE bit.
 2. Check status bits of each mailbox in CAN_TSTS. If any mailbox with TMEIx (x = 0~2) is '1', user can write the message which is waiting for transmission into the corresponding mailbox address. CAN_TMIx.TXRQ(x = 0~2) bit must be written to '1' after this mailbox is programmed.
 3. After some time or after waiting for transmit interrupts, come back to check transmit status in CAN_TSTS. Repeat step 2~3 for new message transmission.
 - For Reception:
 1. User can also change the filter value (CAN_FiRx) when the corresponding filter is deactivated. To deactivate certain filter, user needs to write '0' to the corresponding bit in CAN_FA1 register.

2. Configure reception related interrupts in CAN_INTE register.
3. Once CAN has received message and stored them inside reception FIFO, user needs to read the corresponding FIFO on time and release reception mailbox by writing '1' to RFFOMx in register CAN_RFFx (x = 0,1).

25.7 Can Register File

All register operations must be performed in words (32 bits).

25.7.1 Register Description

Register access protection

When a CAN node is operating normally, incorrect access/modification of some configuration registers may cause hardware errors in the node, and temporarily interfere with the entire CAN network. Therefore, modification of the CAN_BTIM register is only allowed when the CAN core is in initialization mode.

Only when the transmit mailbox status bit CAN_TSTS.TMEM = 1, the user can modify data to the transmit mailbox.

Control and status registers

By configuring these registers, user can configure CAN parameters, such as operating mode and baud rate; start message transmitting; handling message reception; interrupt setting; read diagnostic information.

Mailbox Register Description

The transmitting and receiving mailboxes are basically the same except that the receiving mailbox is read-only and contains the CAN_RMDTx.FMI field. The transmitting mailbox is writable when it is empty.

Note: the corresponding CAN_TSTS.TMEM bit is set, which means that the transmitting mailbox is empty.

There are 3 transmitting mailboxes and 2 receiving FIFO. Each receiving FIFO has three mailboxes, and only the first received message in the FIFO can be accessed.

Each mailbox contains 4 registers:

FIFO0 contains CAN_RMI0, CAN_RMDT0, CAN_RMDL0, CAN_RMDH0;

FIFO1 contains CAN_RMI1, CAN_RMDT1, CAN_RMDL1, CAN_RMDH1;

Transmit mailbox (x) contains CAN_TMIx, CAN_TMDTx, CAN_TMDLx, CAN_TMDHx; x = 0,1,2.

Filter Register Description

The value of the filter can be modified only when the corresponding filter bank is closed or the CAN_FMC.FINITM bit is set. In addition, only when the whole filter is set to the initialization mode (that is, CAN_FMC.FINITM=1), the settings of the filter can be modified, that is, the CAN_FM1,CAN_FS1 and CAN_FFA1 registers can be modified.

25.7.2 CAN Register Address Overview

Table 25-4 CAN Register Overview

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
000h	CAN_MCTRL	Reserved																DBGF	MIRST	Reserved										TTCM	ABOM	AWKUM	NART	RFLM	TXFP	SLPRQ	INIRQ
	Reset Value																	1	0											0	0	0	0	0	0	0	1

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
004h	CAN_MSTS	Reserved																				RXS	LSMP	RXMD	TXMD	Reserved			SLAKINT	WKUJINT	ERRINT	SLPAK	INI/AK	
	Reset Value																					1	1	0	0				0	0	0	1	0	
008h	CAN_TSTS	LOWM[2:0]		TMEEM[2:0]			CODE[1:0]		ABROM2	Reserved				TERRM2	ALSTM2	TXOKM2	RQCPM2	ABRQM1	Reserved				TERRM1	ALSTM1	TXOKM1	RQCPM1	ABROM0	Reserved			TERRM0	ALSTM0	TXOKM0	RQCPM0
	Reset Value	0	0	0	1	1	1	0	0	0					0	0	0	0	0					0	0	0	0	0				0	0	0
00Ch	CAN_RFF0	Reserved																				RFFOM0		FFOVR0	FFULL0	Reserved		FFMP0[1:0]						
	Reset Value																					0	0	0			0	0						
010h	CAN_RFF1	Reserved																				RFFOM1	FFOVR1	FFULL1	Reserved		FFMP1[1:0]							
	Reset Value																					0	0	0			0	0						
014h	CAN_INTE	Reserved												SLKITE	WKUITE	ERRITE	Reserved			LECITE	BOFITE	EPVITE	EWGITE	Reserved		FOVITE1	FFITE1	EMPITE1	FOVITE0	FFITE0	EMPITE0	TMEITE		
	Reset Value													0	0	0				0	0	0	0			0	0	0	0	0	0	0	0	
018h	CAN_ESTS	RXEC[7:0]						TXEC[7:0]						Reserved						LEC[2:0]			Reserved		BOFPL	EPVFL	EWGFL							
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0							0	0	0			0	0	0				
01Ch	CAN_BTIM	SLM	LBM	Reserved				RSJW[1:0]	Reserved		TBS2[2:0]		TBS1[3:0]		Reserved				BRTP[9:0]															
	Reset Value	0	0					0	0			0	1	0	0	0	1	1																
020h - 17Fh	Reserved																																	
180h	CAN_TMI0	STDID[10:0]/EXTID[28:18]												EXTID[17:0]																	IDE	RTRQ	TXRQ	
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	
184h	CAN_TMDT0	MTIM[15:0]												Reserved						TGT	Reserved			DLC[3:0]										
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
188h	CAN_TMDL0	DATA3[7:0]						DATA2[7:0]						DATA1[7:0]						DATA0[7:0]														
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
18Ch	CAN_TMDH0	DATA7[7:0]						DATA6[7:0]						DATA5[7:0]						DATA4[7:0]														
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
190h	CAN_TMI1	STDID[10:0]/EXTID[28:18]												EXTID[17:0]																	IDE	RTRQ	TXRQ	
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	
194h	CAN_TMDT1	MTIM[15:0]												Reserved						TGT	Reserved			DLC[3:0]										
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
198h	CAN_TMDL1	DATA3[7:0]						DATA2[7:0]						DATA1[7:0]						DATA0[7:0]														
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
19Ch	CAN_TMDH1	DATA7[7:0]							DATA6[7:0]							DATA5[7:0]							DATA4[7:0]											
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
1A0h	CAN_TM2	STDID[10:0]/EXTID[28:18]											EXTID[17:0]															IDE	RTRQ	TXRQ				
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0		
1A4h	CAN_TMDT2	MTIM[15:0]															Reserved							TGT	Reserved							DLC[3:0]		
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
1A8h	CAN_TMDL2	DATA3[7:0]							DATA2[7:0]							DATA1[7:0]							DATA0[7:0]											
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
1ACh	CAN_TMDH2	DATA7[7:0]							DATA6[7:0]							DATA5[7:0]							DATA4[7:0]											
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
1B0h	CAN_RMI0	STDID[10:0]/EXTID[28:18]											EXTID[17:0]															IDE	RTRQ	Reserved				
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
1B4h	CAN_RMDT0	MTIM[15:0]															FMI[7:0]							Reserved							DLC[3:0]			
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
1B8h	CAN_RMDL0	DATA3[7:0]							DATA2[7:0]							DATA1[7:0]							DATA0[7:0]											
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
1BCh	CAN_RMDH0	DATA7[7:0]							DATA6[7:0]							DATA5[7:0]							DATA4[7:0]											
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
1C0h	CAN_RMI1	STDID[10:0]/EXTID[28:18]											EXTID[17:0]															IDE	RTRQ	Reserved				
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
1C4h	CAN_RMDT1	MTIM[15:0]															FMI[7:0]							Reserved							DLC[3:0]			
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
1C8h	CAN_RMDL1	DATA3[7:0]							DATA2[7:0]							DATA1[7:0]							DATA0[7:0]											
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
1CCh	CAN_RMDH1	DATA7[7:0]							DATA6[7:0]							DATA5[7:0]							DATA4[7:0]											
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
1D0h - 1FFh	Reserved																																	
200h	CAN_FMC	Reserved																																
	Reset Value	1																																

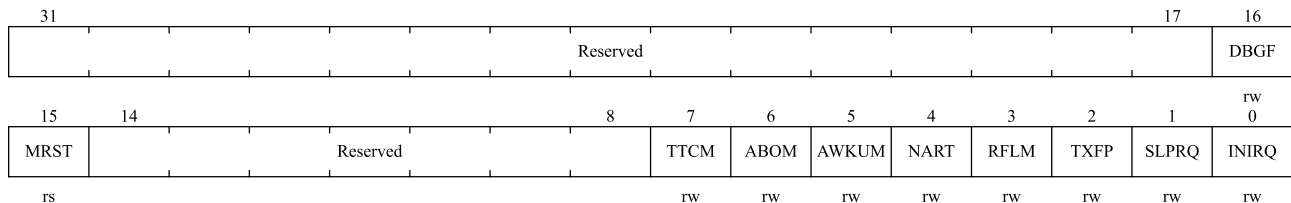
25.7.3 CAN Control and Status Register

Abbreviations used in register descriptions, please refer to Section 1.1.

CAN master control register (CAN_MCTRL)

Address offset: 0x00

Reset value: 0x0001 0002



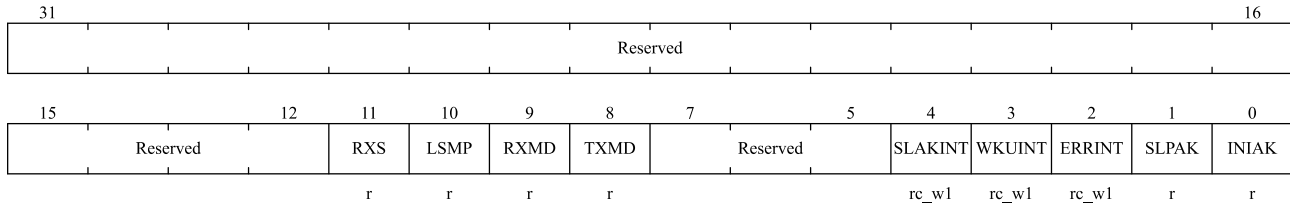
Bit Field	Name	Description
31:17	Reserved	Reserved, the reset value must be maintained.
16	DBGF	Debug freeze 0: During debugging, CAN works as usual. 1: Freeze the reception/transmission of CAN during debugging. The receiving FIFO can still be read, written and controlled normally. <i>Note: DBG_CTRL.CAN_STOP bit must be set when CAN is frozen, please refer to Section 25.3.7: Can Debug Mode.</i>
15	MRST	CAN software master reset 0: This peripheral works normally; 1: enforce reset CAN, after which CAN enters sleep mode and CAN_RFFx.FFMP bit and CAN_MCTRL register are initialized to their reset values. After that, the hardware automatically clears this bit.
14:8	Reserved	Reserved, the reset value must be maintained.
7	TTCM	Time triggered communication mode 0: disable time triggered communication mode; 1: enable time trigger communication mode. <i>Note: For more information about time-triggered communication mode, please refer to Section 25.4.2: Time-triggered communication mode.</i>
6	ABOM	Automatic bus-off management This bit determines the conditions under which the CAN hardware can exit the bus-off state. 0: The process of exiting the bus-off state is that after the software sets the CAN_MCTRL.INIRQ bit and then clears it, once the hardware detects 128 consecutive 11-bit recessive bits, it exits the bus-off state; 1: Once the hardware detects 128 consecutive 11-bit recessive bits, it will automatically exit the bus-off state. <i>Note: For more information about bus-off status, please refer to Section 25.5.1: Error management.</i>
5	AWKUM	Automatic wake up mode This bit determines whether CAN is awakened by hardware or software when it is

Bit Field	Name	Description
		<p>in sleep mode.</p> <p>0: The sleep mode is awakened by the software by clearing the CAN_MCTRL.SLPRQ bit;</p> <p>1: Sleep mode is automatically awakened by hardware by detecting CAN messages. At the same time of wake-up, the hardware automatically clears the CAN_MSTS.SLPRQ and CAN_MSTS.SLPAK bits.</p>
4	NART	<p>No automatic retransmission</p> <p>0: According to the CAN standard, when the CAN hardware fails to send a message, it will automatically retransmit it until it is successfully sent;</p> <p>1: CAN message is only sent once, regardless of the sending result (success, error or arbitration loss).</p>
3	RFLM	<p>Receive FIFO locked mode.</p> <p>0: the FIFO is not locked when receiving overflows, and when the message of the receiving FIFO is not read out, the next received message will overwrite the last message;</p> <p>1: FIFO is locked when receiving overflow. When the message of receiving FIFO is not read out, the next received message will be discarded.</p>
2	TXFP	<p>Transmit FIFO priority</p> <p>When there are multiple messages waiting to be sent at the same time, this bit determines the sending order of these messages.</p> <p>0: Priority is determined by the identifier of the message;</p> <p>1: Priority is determined by the order in which requests are sent.</p>
1	SLPRQ	<p>Sleep mode request</p> <p>The software can request the CAN to enter the sleep mode by setting this bit, and once the current CAN activity (sending or receiving messages) ends, the CAN will enter the sleep mode.</p> <p>Clear by software to make CAN exit sleep mode.</p> <p>When the CAN_MCTRL.AWKUM bit is set and the SOF bit is detected in the CAN Rx signal, the hardware clears this bit.</p> <p>This bit is set after reset, that is, CAN is in sleep mode after reset.</p>
0	INIRQ	<p>Initialization request</p> <p>clear this bit by software can make CAN exit from initialization mode: when CAN leaves Initialization mode and entering normal mode, it needs to detect 11 consecutive recessive bits at the receiving pin, CAN will be synchronized and ready for receiving and sending data. To this end, the hardware correspondingly the CAN_MSTS.INIAK bit is cleared.</p> <p>Setting this bit by software enables CAN to enter initialization mode from normal operation mode: once the current CAN activity (sending or receiving) is over, the hardware sets the CAN_MSTS.INIAK bit and CAN enters initialization mode.</p>

CAN master status register (CAN_MSTS)

Address offset: 0x04

Reset value: 0x0000c02



Bit Field	Name	Description
31:12	Reserved	Reserved, the reset value must be maintained.
11	RXS	CAN Rx signal This bit reflects the actual level of the CAN receive pin (CAN_RX).
10	LSMP	Last sample point The last sampled value of the CAN receive pin (corresponding to the value of the current receive bit).
9	RXMD	Receive mode if this bit equals to 1 indicates CAN is currently the receiver.
8	TXMD	Transmit mode if this bit equals to 1 indicates CAN is currently the transmitter.
7:5	Reserved	Reserved, the reset value must be maintained.
4	SLAKINT	Sleep acknowledge interrupt When CAN_INTE.SLKITE=1, once CAN enters sleep mode, hardware will set this bit, and then the corresponding interrupt will be triggered. When this bit is set, if the CAN_INTE.SLKITE bit is set, a state change interrupt will be generated. Software can clear this bit, and hardware also clears this bit when CAN_MSTS.SLPAK bit is cleared. <i>Note: When CAN_INTE.SLKITE=0, this bit should not be queried, but the CAN_MSTS.SLPAK bit should be queried to know the sleep state.</i>
3	WKUINT	Wakeup interrupt When CAN is in sleep state, once the start of frame bit (SOF) is detected, the hardware will set this bit; And if the CAN_INTE.WKUIITE bit is set, a state change interrupt is generated. This bit is cleared by software.
2	ERRINT	Error interrupt When an error is detected, a bit of the CAN_ESTS register will be set, and if the corresponding interrupt enable bit of the CAN_INTE register is also set, the hardware will set this bit; If the CAN_INTE.ERRITE bit is set, a state change interrupt is generated. This bit is cleared by software.
1	SLPAK	Sleep acknowledge This bit is set by hardware, indicating that the software CAN module is in sleep mode. This bit is the confirmation of the software request to enter sleep mode (the CAN_MCTRL.SLPRQ bit is set). Hardware clears this bit when CAN exits sleep mode (CAN leaves Sleep mode and entering normal mode,it needs to be synchronized with CAN bus).

Bit Field	Name	Description
		Synchronization with CAN bus here means that the hardware needs to detect 11 consecutive recessive bits on the RX pin of CAN. <i>Note: clearing CAN_MCTRL.SLPRQ bit by software or hardware will start the process of exiting sleep mode. See the description of CAN_MCTRL.AWKUM bit for details of clearing CAN_MCTRL.SLPRQ bit.</i>
0	INIAK	Initialization acknowledge This bit is set by hardware, indicating that the software CAN module is in initialization mode. This bit is the confirmation of the software request to enter the initialization mode (the CAN_MCTRL.INIRQ bit is set). Hardware clears this bit when CAN exits initialization mode (CAN leaves Initialization mode and entering normal mode, it needs to be synchronized with CAN bus). Synchronization with CAN bus here means that the hardware needs to detect 11 consecutive recessive bits on the RX pin of CAN.

CAN transmit status register (CAN_TSTS)

Address offset: 0x08

Reset value: 0x1C00 0000

31	30	29	28	27	26	25	24	23	22	20	19	18	17	16	
LOWM2	LOWM1	LOWM0	TMEM2	TMEM1	TMEM0	CODE		ABRQM2		Reserved		TERRM2	ALSTM2	TXOKM2	RQCPM2
r	r	r	r	r	r	r	r	rs				rc_wl	rc_wl	rc_wl	rc_wl
15	14		12	11	10	9	8	7	6	4	3	2	1	0	
ABRQM1		Reserved		TERRM1	ALSTM1	TXOKM1	RQCPM1	ABRQM0		Reserved		TERRM0	ALSTM0	TXOKM0	RQCPM0
rs				rc_wl	rc_wl	rc_wl	rc_wl	rs				rc_wl	rc_wl	rc_wl	rc_wl

Bit Field	Name	Description
31	LOWM2	Lowest priority flag for mailbox 2 When multiple mailboxes are waiting to send messages, and the priority of mailbox 2 is the lowest, hardware sets this bit.
30	LOWM1	Lowest priority flag for mailbox 1 When multiple mailboxes are waiting to send messages, and the priority of mailbox 1 is the lowest, hardware sets this bit.
29	LOWM0	Lowest priority flag for mailbox 0 When multiple mailboxes are waiting to send messages, and the priority of mailbox 0 is the lowest, hardware sets this bit. <i>Note: If there is only one mailbox waiting, CAN_TSTS.LOW[2:0] is cleared</i>
28	TMEM2	Transmit mailbox 2 empty When there is no message waiting to be sent in mailbox 2, hardware sets this bit.
27	TMEM1	Transmit mailbox 1 empty When there is no message waiting to be sent in mailbox 1, hardware sets this bit.
26	TMEM0	Transmit mailbox 0 empty When there is no message waiting to be sent in mailbox 0, hardware sets this bit .
25:24	CODE[1:0]	Mailbox code When at least one sending mailbox is empty, these two bits represent the next

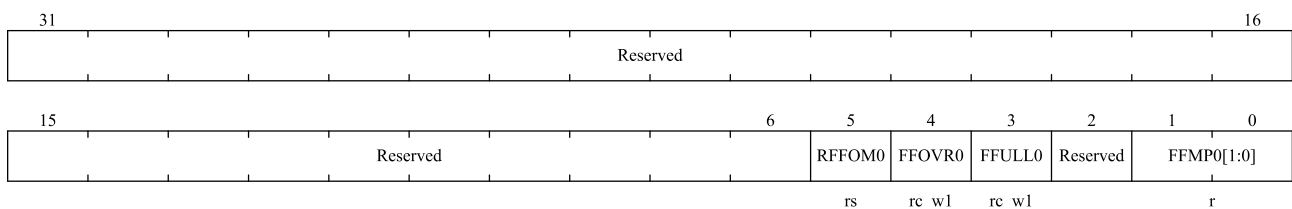
Bit Field	Name	Description
		empty sending mailbox number. When all sending mailboxes are empty, these two bits represent the sending mailbox number with the lowest priority.
23	ABRQM2	Abort request for mailbox 2 Set this bit, software can stop the sending request of mailbox 2, and hardware clears this bit when the sending message of mailbox 2 is idle. If there is no message waiting to be sent in mailbox 2, it will have no effect to set this bit.
22:20	Reserved	Reserved, hardware force is 0.
19	TERRM2	Transmission error of mailbox 2 failed. When the mailbox 2 fails to send due to an error, set this bit.
18	ALSTM2	Arbitration lost for mailbox 2 When the mailbox 2 fails to send due to the loss of arbitration, set this bit.
17	TXOKM2	Transmission OK of mailbox 2 The hardware updates this bit after each sending attempt of mailbox 2: 0: The last sending attempt is not yet successful; 1: The last sending attempt was successful. When the sending request of mailbox 2 is successfully completed, hardware sets this bit. See Figure 25-7.
16	RQCPM2	Request completed mailbox 2 When the last request (send or abort) for mailbox 2 is completed, the hardware will set this bit. Writing '1' to this bit by software can clear it; When the hardware receives the send request, it can also clear this bit (the CAN_TMI2.TXRQ bit is set). When this bit is cleared, other sending status bits (CAN_TSTS.TXOKM2, CAN_TSTS.ALSTM2 and CAN_TSTS.TERRM2 bits) of mailbox 2 are also cleared.
15	ABRQM1	Abort request for mailbox 1 Set this bit, the software can stop the sending request of mailbox 1, and the hardware clears this bit when the sending message of mailbox 1 is idle. If there is no message waiting to be sent in mailbox 1, it will have no effect to set this bit.
14:12	Reserved	Reserved, the reset value must be maintained.
11	TERRM1	Transmission error of mailbox 1 When the mailbox 1 fails to send due to an error, set this bit.
10	ALSTM1	Arbitration lost for mailbox 1 When the mailbox 1 fails to send due to the loss of arbitration, set this bit
9	TXOKM1	Transmission OK of mailbox 1 The hardware updates this bit after each sending attempt of mailbox 1: 0: The last sending attempt is not yet successful; 1: The last sending attempt was successful. When the sending request of mailbox 1 is successfully completed, the hardware sets this bit. See Figure 25-7.
8	RQCPM1	Request completed mailbox 1 When the last request (send or abort) for mailbox 1 is completed, the hardware

Bit Field	Name	Description
		sets this bit. Writing '1' to this bit by software can clear it; When the hardware receives the send request, it can also clears this bit (the CAN_TMI1.TXRQ bit is set). When this bit is cleared, other sending status bits (CAN_TSTS.TXOKM1, CAN_TSTS.ALSTM1 and CAN_TSTS.TERRM1 bits) of mailbox 1 are also cleared.
7	ABRQM0	Abort request for mailbox 0 The software can stop the sending request of mailbox 0 by setting this bit , and the hardware clears this bit when the sending message of mailbox 0 is idle. If there is no message waiting to be sent in mailbox 0, it will have no effect to set this bit.
6:4	Reserved	Reserved, the reset value must be maintained.
3	TERRM0	Transmission error of mailbox 0 When the mailbox 0 fails to send due to an error, set this bit.
2	ALSTM0	Arbitration lost for mailbox 0 When the mailbox 0 fails to send due to the loss of arbitration, set this bit
1	TXOKM0	Transmission OK of mailbox 0 The hardware updates this bit after each attempt to send mailbox 0: 0: The last send attempt is not yet successful; 1: The last sending attempt was successful. When the sending request of mailbox 0 is successfully completed, the hardware sets this bit. See Figure 25-7.
0	RQCPM0	Request completed mailbox 0 When the last request (send or abort) for mailbox 0 was completed, the hardware sets this bit. Writing '1' to this bit by software can clear it; When the hardware receives the send request, it can also clears this bit (the CAN_TMI0.TXRQ bit is set). When this bit is cleared, other sending status bits (CAN_TSTS.TXOKM0, CAN_TSTS.ALSTM0 and CAN_TSTS.TERRM0 bits) of mailbox 0 are also cleared.

CAN receive FIFO 0 register (CAN_RFF0)

Address offset: 0x0c

Reset value: 0x0000 0000



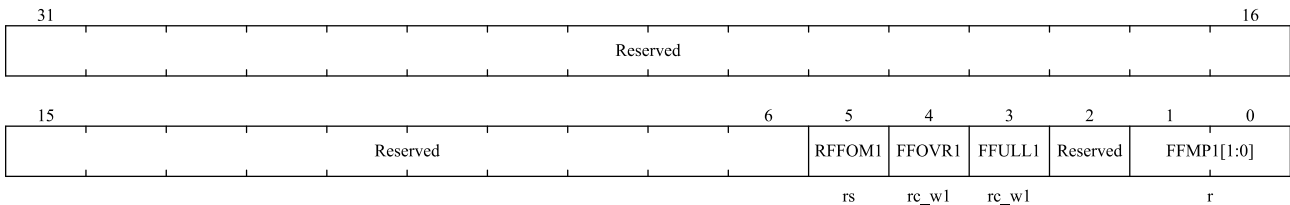
Bit Field	Name	Description
31:6	Reserved	Reserved, the reset value must be maintained.
5	RFFOM0	Release FIFO 0 output mailbox.

Bit Field	Name	Description
		The software releases the output mailbox of the receive FIFO by setting this bit. If the receiving FIFO is empty, it will have no effect on setting this bit, that is, it will be meaningful to set this bit only when there is a message in the FIFO. If there are more than two messages in FIFO, because of the characteristics of FIFO, the software needs to release the output mailbox to access the second message. When the output mailbox is released, the hardware clears this bit.
4	FFOVR0	FIFO 0 overrun When FIFO 0 is full, a new message is received and the message meets the filtering conditions, the hardware sets this bit. This bit is cleared by software.
3	FFULL0	FIFO 0 full When there are 3 messages in FIFO 0, the hardware sets this bit. This bit is cleared by software.
2	Reserved	Reserved, the reset value must be maintained.
1:0	FFMP0[1:0]	FIFO 0 message pending Number of FIFO messages 0 These two bits reflect the number of messages currently stored in the receiving FIFO 0. Every time a new message is stored in the receiving FIFO 0, the hardware adds 1 to the CAN_RFF0.FFMP0. Every time the software writes '1' to the CAN_RFF0.RFFOM0 bit to release the output mailbox, CAN_RFF0.FFMP0 is decremented by 1 until it is 0.

CAN receive FIFO 1 register (CAN_RFF1)

Address offset: 0x10

Reset value: 0x0000 0000



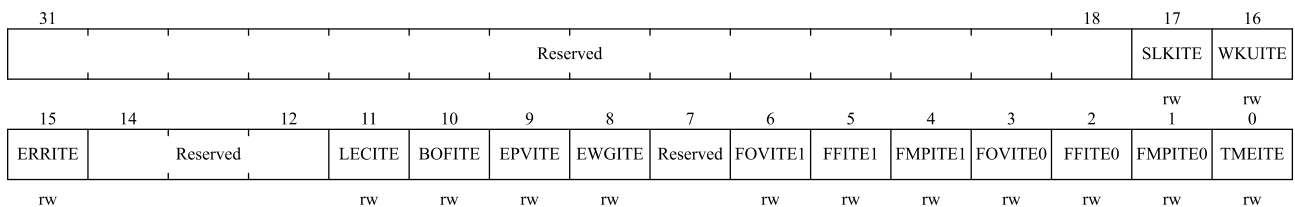
Bit Field	Name	Description
31:6	Reserved	Reserved, the reset value must be maintained.
5	RFFOM1	Release FIFO 1 output mailbox. The software releases the output mailbox of the receive FIFO by setting this bit. If the receiving FIFO is empty, it will have no effect on setting this bit, that is, it will be meaningful to set this bit only when there is a message in the FIFO. If there are more than two messages in FIFO, because of the characteristics of FIFO, the software needs to release the output mailbox to access the second message. When the output mailbox is released, the hardware clears this bit.
4	FFOVR1	FIFO 1 overrun When FIFO 1 is full, a new message is received and the message meets the filtering conditions, the hardware sets this bit. This bit is cleared by software.
3	FFULL1	FIFO 1 full

Bit Field	Name	Description
		When there are 3 messages in FIFO 1, the hardware sets this bit. This bit is cleared by software.
2	Reserved	Reserved, the reset value must be maintained.
1:0	FFMP1[1:0]	FIFO 1 message pending Number of messages in FIFO 1 These two bits reflect the number of messages stored in the current receiving FIFO 1. Every time a new message is stored in receiving FIFO 1, the hardware adds 1 to CAN_RFF1.FFMP1. Every time the software releases the output mailbox by writing '1' to CAN_RFF1.RFFOM1 bit, CAN_RFF1.FFMP1 is decremented by 1 until it is 0.

CAN interrupt enable register (CAN_INTE)

Address offset: 0x14

Reset value: 0x0000 0000



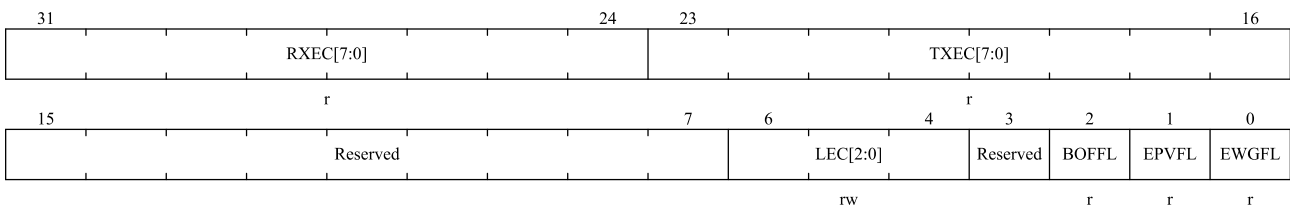
Bit Field	Name	Description
31:18	Reserved	Reserved, the reset value must be maintained.
17	SLKITE	Sleep interrupt enable 0: when the CAN_MSTS.SLAKINT bit is set, no interrupt is generated; 1: when the CAN_MSTS.SLAKINT bit is set, an interrupt is generated.
16	WKUITE	Wakeup interrupt enable 0: when CAN_MSTS.WKUINT bit is set, no interrupt is generated; 1: when CAN_MSTS.WKUINT bit is set, an interrupt is generated.
15	ERRITE	Error interrupt enable 0: When there is an error registration in the CAN_ESTS register, no interrupt is generated; 1: When there is an error registration in the CAN_ESTS register, an interrupt is generated.
14:12	Reserved	Reserved, the reset value must be maintained.
11	LECITE	Last error code interrupt enable 0: When an error is detected, when the hardware sets CAN_ESTS.LEC[2:0], the CAN_MSTS.ERRINT bit is not set; 1: When an error is detected, when the hardware sets CAN_ESTS.LEC[2:0], the CAN_MSTS.ERRINT bit is set.
10	BOFITE	Bus-off interrupt enable 0: When CAN_ESTS.BOFFL bit is set, CAN_MSTS.ERRINT bit is not set; 1: When the CAN_ESTS.BOFFL bit is set, set the CAN_MSTS.ERRINT bit.
9	EPVITE	Error passive interrupt enable

Bit Field	Name	Description
		0: when CAN_ESTS.EPVFL bit is set, CAN_MSTS.ERRINT bit is not set; 1: when CAN_ESTS.EPVFL bit is set, set the CAN_MSTS.ERRINT bit.
8	EWGITE	Error warning interrupt enable 0: When CAN_ESTS.EWGFL bit is set, CAN_MSTS.ERRINT bit is not set; 1: when the CAN_ESTS.EWGFL bit is set, set the CAN_MSTS.ERRINT bit.
7	Reserved	Reserved, the reset value must be maintained.
6	FOVITE1	FIFO 1 overflow interrupt enable 0: When CAN_RFF1.FFOVR bit is set, no interrupt is generated; 1: When CAN_RFF1.FFOVR bit is set, an interrupt is generated.
5	FFITE1	FIFO 1 full interrupt enable 0: When CAN_RFF1.FFULL bit is set, no interrupt is generated; 1: When CAN_RFF1.FFULL bit is set, an interrupt is generated.
4	FMPITE1	FIFO 1 message pending interrupt enable 0: When CAN_RFF1.FFMP[1:0] bits are non-0, no interrupt is generated; 1: When CAN_RFF1.FFMP[1:0] bits are not 0, an interrupt is generated.
3	FOVITE0	FIFO 0 overflow interrupt enable 0: When CAN_RFF0.FFOVR bit is set, no interrupt is generated; 1: When CAN_RFF0.FFOVR bit is set, an interrupt is generated.
2	FFITE0	FIFO 0 full interrupt enable 0: When CAN_RFF0.FFULL bit is set, no interrupt is generated; 1: When CAN_RFF0.FFULL bit is set, an interrupt is generated.
1	FMPITE0	FIFO 0 message pending interrupt enable 0: When CAN_RFF0.FFMP[1:0] bits are non-0, no interrupt is generated; 1: When CAN_RFF0.FFMP[1:0] bits are not 0, an interrupt is generated.
0	TMEITE	Transmit mailbox empty interrupt enable. 0: When CAN_TSTS.RQCPMx bit is set, no interrupt is generated; 1: When CAN_TSTS.RQCPMx bit is set, an interrupt is generated. <i>Note: Please refer to 25.5 Section CAN interrupt.</i>

CAN error status register (CAN_ESTS)

Address offset: 0x18

Reset value: 0x0000 0000



Bit Field	Name	Description
31:24	RXEC[7:0]	Receive error counter This counter is implemented according to the receiving part of the fault definition mechanism of CAN protocol. According to the standard of CAN, when receiving

Bit Field	Name	Description
		error, the counter is incremented by 1 or incremented by 8 according to the error condition; After each successful reception, the counter is decremented by 1, or when the value of the counter is greater than 127, its value is set to 120. When the value of this counter exceeds 127, CAN enters the error passive state.
23:16	TXEC[7:0]	Least significant byte of the 9-bit transmit error counter Similar to the above, this counter is implemented according to the sending part of the fault definition mechanism of CAN protocol.
15:7	Reserved	Reserved, the reset value must be maintained.
6:4	LEC[2:0]	The Last error code. When an error is detected on the CAN bus, the hardware is set according to the error situation. When the message is sent or received correctly, the hardware clears its value to '0'. The hardware does not use error code 7, and the software can set this value, so that the code update can be detected. 000: there is no error; 001: Bit padding error; 010: wrong Format (form); 011: Acknowledgment (ack) error; 100: recessive dislocation (CAN transmits recessive but detect dominant on the bus) ; 101: dominant dislocation(CAN transmits dominant but detect recessive on the bus); 110: CRC error; 111: Set by software.
3	Reserved	Reserved, the reset value must be maintained.
2	BOFFL	Bus-off flag When going bus-off, hardware sets this bit. When the transmission error counter CAN_TSTS.TXEC overflows, that is, it is greater than 255, CAN goes bus-off. Please refer to 25.5.1 section.
1	EPVFL	Error passive flag When the number of errors reaches the threshold of error passivity, the hardware sets the bit. (The value of the receiving error counter or the sending error counter is > 127).
0	EWGFL	Error warning flag When the number of errors reaches the warning threshold, the hardware sets the bit. (The value of the receiving error counter or the sending error counter is ≥96).

CAN bit timing register (CAN_BTIM)

Address offset: 0x1C

Reset value: 0x0023 0000

Note: This register CAN only be accessed by software when CAN is in initialization mode.



Bit Field	Name	Description
31	SLM	Silent mode (debug) 0: Normal state; 1: Silent mode.
30	LBM	Loop back mode (debug) 0: Loopback mode is prohibited; 1: Loopback mode is allowed.
29:26	Reserved	Reserved, the reset value must be maintained.
25:24	RSJW[1:0]	Resynchronization jump width For resynchronization, this bit field defines the upper limit of how many time units CAN be extended or shortened by CAN hardware in each bit. $t_{RJW} = t_{CAN} \times (RSJW[1:0] + 1)$.
23	Reserved	Reserved, the reset value must be maintained.
22:20	TBS2[2:0]	Time segment 2 This bit field defines how many time units time period 2 occupies. $t_{BS2} = t_{CAN} \times (TBS2[2:0] + 1)$.
19:16	TBS1[3:0]	Time segment 1 This bit field defines how many time units time period 1 occupies. $t_{BS1} = t_{CAN} \times (TBS1[3:0] + 1)$ For details of bit time characteristics, please refer to Section 25.4.7: bit time characteristics.
15:10	Reserved	Reserved, the reset value must be maintained.
9:0	BRTP[9:0]	Baud rate prescaler This bit field defines the time length of the time unit (t_q) $t_q = (BRTP[9:0] + 1) \times t_{PCLK}$

25.7.4 CAN Mailbox Register

This section describes the sending and receiving mailbox registers. For more information about register mapping, refer to Section 25.4.6: message storage.

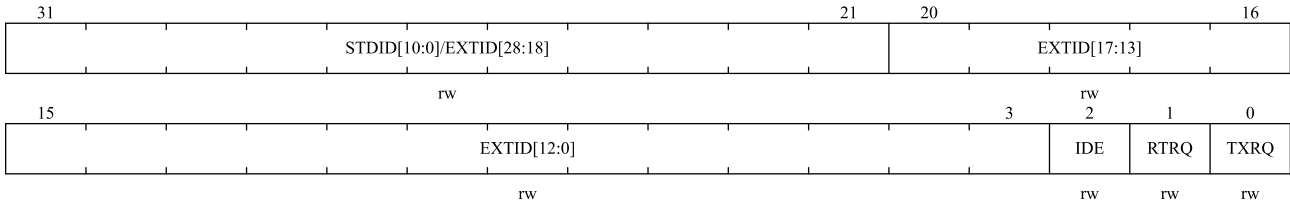
Tx mailbox identifier register(CAN_TMIx)(x=0..2)

Address offset: 0x180, 0x190, 0x1A0

Reset value: 0xXXXX XXXX,X= undefined bit (except bit 0, TXRQ=0 at reset)

Notes:

- (1) This register is write-protected when the mailbox to which it belongs is waiting to be sent;
- (2) This register implements the send request control function (bit 0)-the reset value is 0.



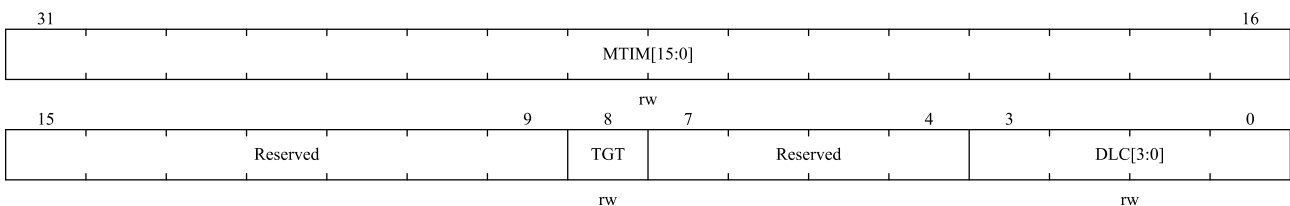
Bit Field	Name	Description
31:21	STDID[10:0]/EXTID[28:18]	Standard identifier or extended identifier Depending on the content of CAN_TMIx.IDE bits, these bits are either standard identifiers or high bytes of extended identity.
20:3	EXTID[17:0]	Extended identifier Low byte of extended identity.
2	IDE	Identifier extension. This bit determines the type of identifier used for sending messages in the mailbox. 0: Use standard identifier; 1: Use extended identifiers.
1	RTRQ	The Remote transmission request 0: data frame; 1: Remote frame.
0	TXRQ	Transmit mailbox request It is set by the software to request to send the data of the mailbox. When the data transmission is completed and the mailbox is empty, hardware clears it.

Tx mailbox data length and time stamp register (CAN_TMDTx)(x=0..2)

When the mailbox is not empty, all bits in this register are write-protected.

Address offset: 0x184, 0x194, 0x1A4

Reset value: undefined



Bit Field	Name	Description
31:16	MTIM[15:0]	Message time stamp This field contains the value of the 16-bit timer at the time of sending the message SOF.
15:9	Reserved	Reserved, the reset value must be maintained.

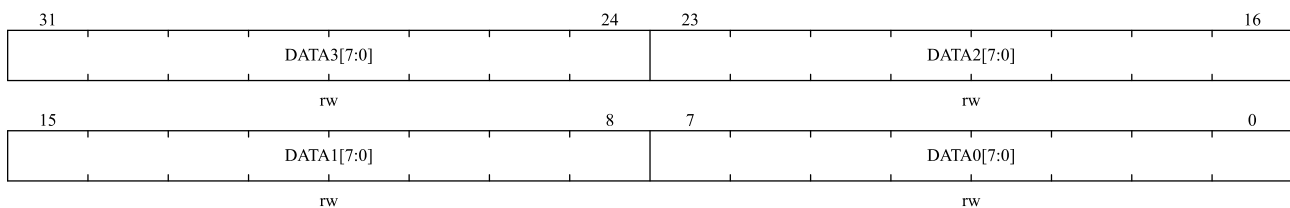
Bit Field	Name	Description
8	TGT	<p>Transmit global time</p> <p>This bit is valid only when the CAN is in time-triggered communication mode, that is, the CAN_MCTRL.TTCM bit is set.</p> <p>0: Do not send the time stamp CAN_TMDTx.MTIM[15:0];</p> <p>1: send the time stamp CAN_TMDTx.MTIM[15:0]. In a message of length 8, the time stamp CAN_TMDTx.MTIM[15:0] is the last two bytes sent:</p> <p>CAN_TMDTx.MTIM[7:0] is the seventh byte and CAN_TMDTx.MTIM[15:8] is the eighth byte. They replace the data written in CAN_TMDHx[31:16] (CAN_TMDLx.DATA6[7:0] and CAN_TMDLx.DATA7[7:0]). In order to send the 2 bytes of the timestamp, DLC must be programmed to 8.</p>
7:4	Reserved	Reserved, the reset value must be maintained.
3:0	DLC[3:0]	<p>Data length code</p> <p>This field specifies the data length of the data message or the data length requested by the remote frame. One message contains 0 to 8 bytes of data, which is determined by DLC.</p>

Tx mailbox low byte data register(CAN_TMDLx) (x=0..2)

When the mailbox is not empty, all bits in this register are write-protected.

Address offset: 0x188, 0x198, 0x1A8

Reset value: undefined



Bit Field	Name	Description
31:24	DATA3[7:0]	<p>Data byte 3</p> <p>Data byte 3 of the message.</p>
23:16	DATA2[7:0]	<p>Data byte 2</p> <p>Data byte 2 of the message.</p>
15:8	DATA1[7:0]	Data byte 1

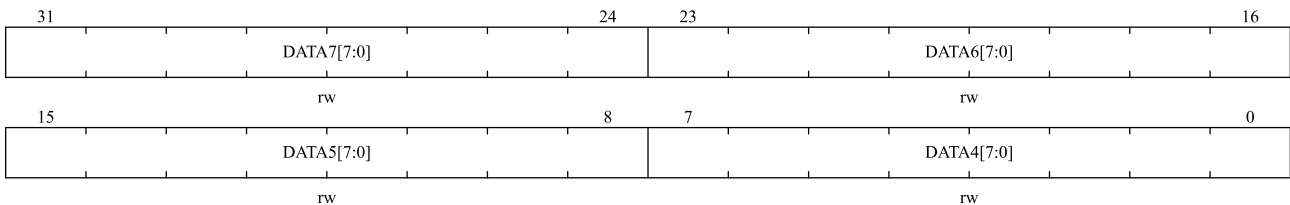
Bit Field	Name	Description
		Data byte 1 of the message.
7:0	DATA0[7:0]	Data byte 0 Data byte 0 of the message. <i>Note: the message contains 0 to 8 bytes of data, starting from byte 0.</i>

Tx mailbox high byte data register(CAN_TMDHx) (x=0..2)

When the mailbox is not empty, all bits in this register are write protected.

Address offset: 0x18c, 0x19c, 0x1ac

Reset value: undefined



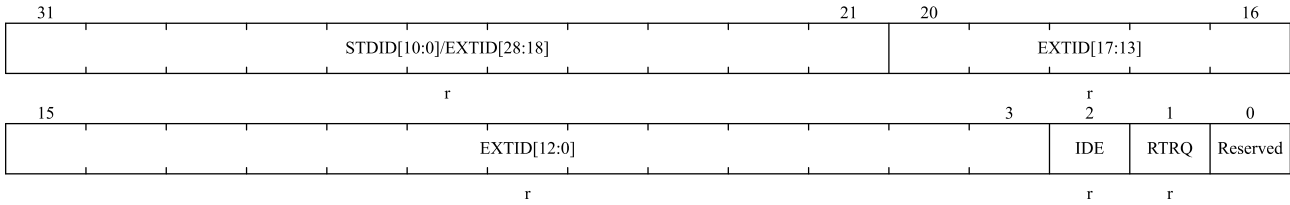
Bit Field	Name	Description
31:24	DATA7[7:0]	Data byte 7 Data byte 7 of the message. <i>Note: If the CAN_MCTRL.TTCM bit is '1' and the CAN_TMDTx.TGT bit of this mailbox is also '1', then DATA7 and DATA6 will be replaced by TIME stamps.</i>
23:16	DATA6[7:0]	Data byte 6 Data byte 6 of the message.
15:8	DATA5[7:0]	Data byte 5 Data byte 5 of the message.
7:0	DATA4[7:0]	Data byte 4 Data byte 4 of the message.

Receive FIFO mailbox identifier register (CAN_RMIx) (x=0..1)

Address offset: 0x1B0, 0x1C0

Reset value: undefined

Note: All receiving mailbox registers are read-only.



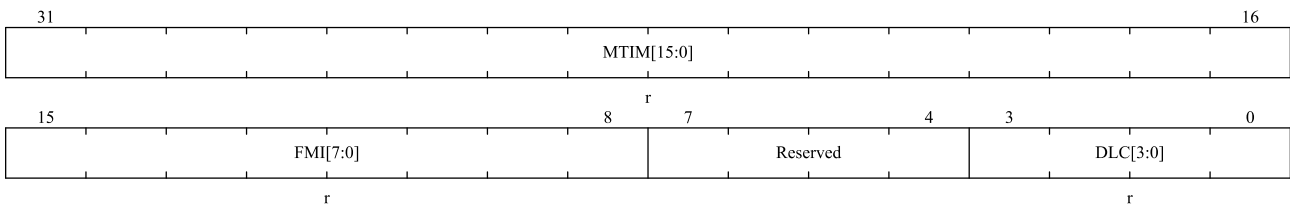
Bit Field	Name	Description
31:21	STDID[10:0]/EXTID[28:18]	Standard identifier or extended identifier Depending on the content of CAN_RMIx.IDE bits, these bits are either standard identifiers or high bytes of extended identity.
20:3	EXTID[17:0]	Extended identifier Low byte of extended identity.
2	IDE	Identifier extension. This bit determines the type of identifier used for sending messages in the mailbox. 0: Use standard identifier; 1: Use extended identifiers.
1	RTRQ	Remote transmission request 0: data frame; 1: Remote frame.
0	Reserved	Reserved, the reset value must be maintained.

Receive FIFO mailbox data length and time stamp register(CAN_RMDTx)(x=0..1)

Address offset: 0x1B4, 0x1C4

Reset value: undefined

Note: All receiving mailbox registers are read-only.



Bit Field	Name	Description
31:16	MTIM[15:0]	Message time stamp

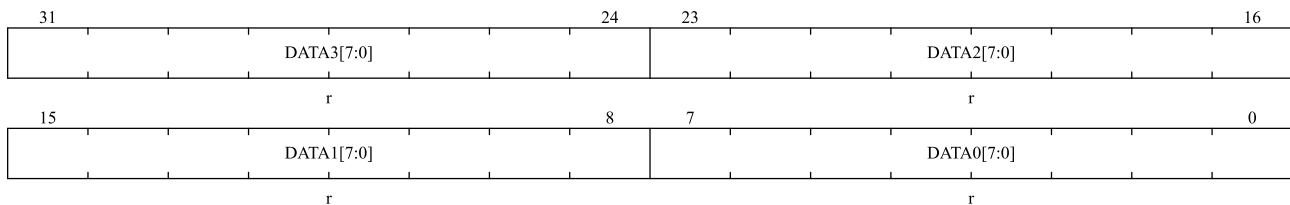
Bit Field	Name	Description
		This field contains the value of the 16-bit timer at the time of sending the message SOF.
15:8	FMI[7:0]	Filter match index Here is the filter serial number of the information transfer stored in the mailbox. For details of identifier filtering, please refer to 25.4.5 Section: Identifier filtering.
7:4	Reserved	Reserved, the reset value must be maintained.
3:0	DLC[3:0]	Data length code This field indicates the data length (0~8) of the received data frame. For remote frame requests, the data length DLC is always 0.

Receive FIFO mailbox low byte data register(CAN_RMDLx)(x=0..1)

Address offset: 0x1B8, 0x1C8

Reset value: undefined

Note: All receiving mailbox registers are read-only.

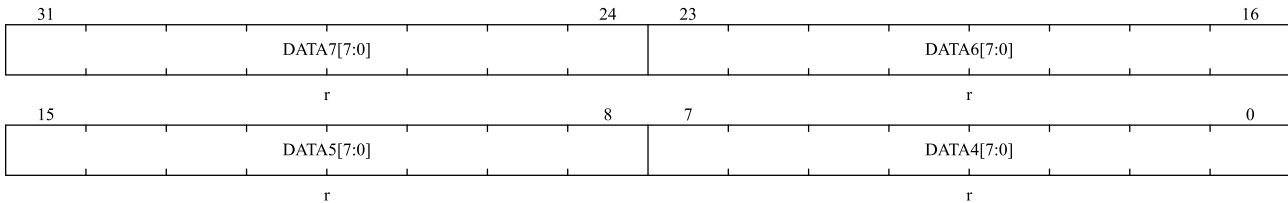


Bit Field	Name	Description
31:24	DATA3[7:0]	Data byte 3 Data byte 3 of the message.
23:16	DATA2[7:0]	Data byte 2 Data byte 2 of the message.
15:8	DATA1[7:0]	Data byte 1 Data byte 1 of the message.
7:0	DATA0[7:0]	Data byte 0 Data byte 0 of the message. <i>Note: the message contains 0 to 8 bytes of data, starting from byte 0.</i>

Receive FIFO mailbox high byte data register(CAN_RMDHx) (x=0..1)

Address offset: 0x1BC, 0x1CC

Reset value: undefined

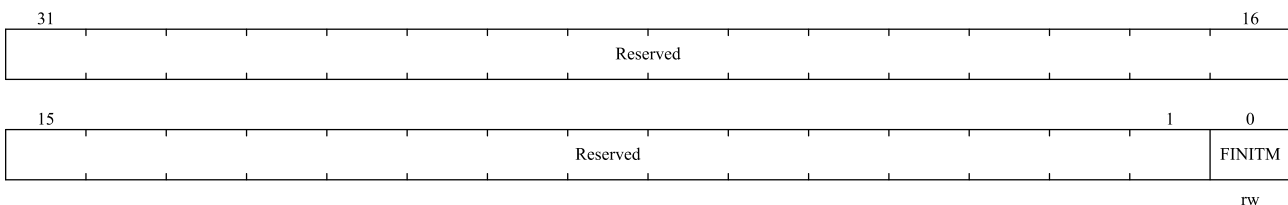
Note: All receiving mailbox registers are read-only.


Bit Field	Name	Description
31:24	DATA7[7:0]	Data byte 7 Data byte 7 of the message.
23:16	DATA6[7:0]	Data byte 6 Data byte 6 of the message.
15:8	DATA5[7:0]	Data byte 5 Data byte 5 of the message.
7:0	DATA4[7:0]	Data byte 4 Data byte 4 of the message.

25.7.5 CAN Filter Register
CAN filter master control register (CAN_FMC)

Address offset: 0x200

Reset value: 0x2A1C 0E01

Note: The unreserved bits of this register are completely controlled by software.


Bit Field	Name	Description
31:1	Reserved	Reserved, the reset value must be maintained.

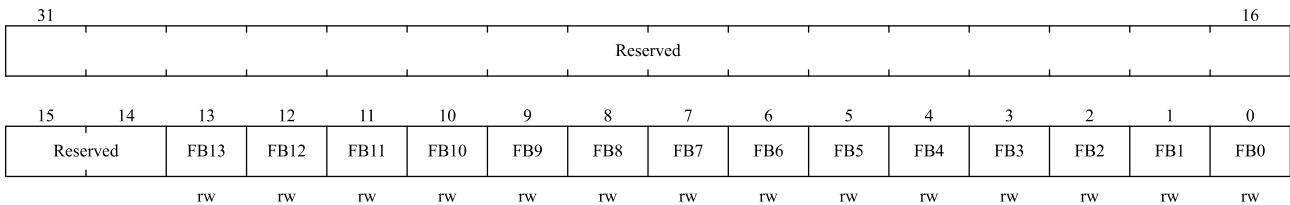
Bit Field	Name	Description
0	FINITM	Filter init mode Initialization mode settings for all filter groups. 0: The filter group works in normal mode; 1: The filter group works in initialization mode.

CAN filter mode register (CAN_FM1)

Address offset: 0x204

Reset value: 0x0000 0000

Note: You can only write to this register when you set CAN_FMC.FINITM bit and put the filter in initialization mode.



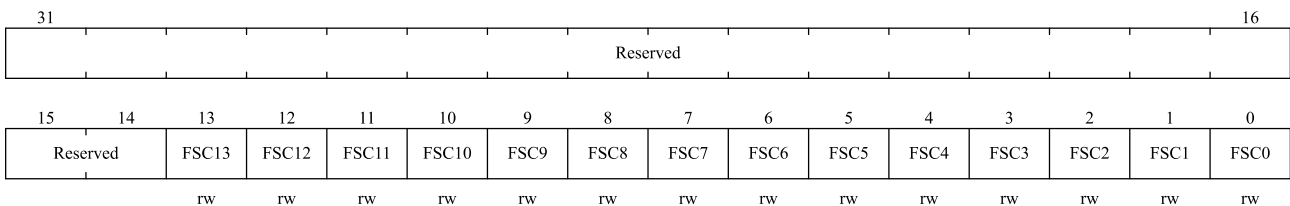
Bit Field	Name	Description
31:14	Reserved	Reserved, the reset value must be maintained.
13:0	FBx	Filter mode Working mode of filter group X 0: Two 32-bit registers of CAN_FiRx work in identifier mask mode; 1: Two 32-bit registers of CAN_FiRx work in identifier list mode.

CAN filter scale register (CAN_FS1)

Address offset: 0x20C

Reset value: 0x0000 0000

Note: You can only write to this register when you set CAN_FMC.FINITM bit and put the filter in initialization mode.



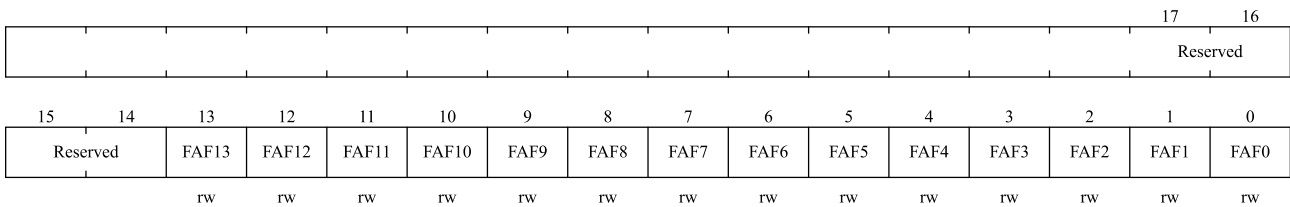
Bit Field	Name	Description
31:14	Reserved	Reserved, the reset value must be maintained.
13:0	FSCx	Filter scale configuration Bit width of filter group x (13~0). 0: The filter bit width is 2 16-bits; 1: The filter bit width is a single 32-bit.

CAN filter FIFO assignment register (CAN_FFA1)

Address offset: 0x214

Reset value: 0x0000 0000

Note: You can only write to this register when you set CAN_FMC.FINITM bit and put the filter in initialization mode.

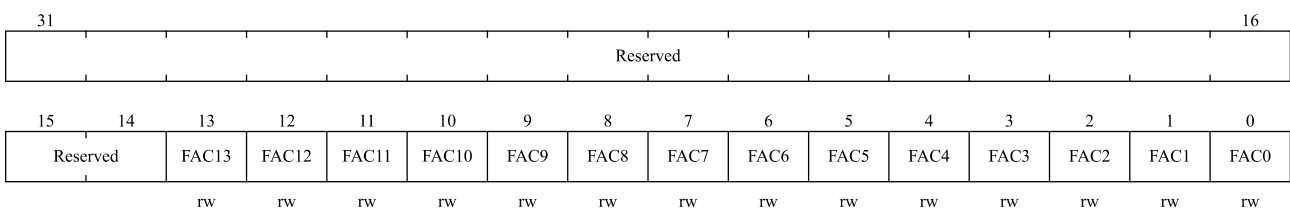


Bit Field	Name	Description
31:14	Reserved	Reserved, the reset value must be maintained.
13:0	FAFx	Filter FIFO assignment for filter x After the message is filtered by a certain filter, it will be stored in its associated FIFO. 0: the filter is associated to FIFO0; 1: the filter is associated to FIFO1.

CAN filter activation register (CAN_FA1)

Address offset: 0x21C

Reset value: 0x0000 0000



Bit Field	Name	Description
31:14	Reserved	Reserved, the reset value must be maintained.
13:0	FACx	Filter active The software sets '1' for someone to activate the corresponding filter. The corresponding filter register i(CAN_FiR[2:1]) can only be modified after the CAN_FA1.FACx bit is cleared or the CAN_FMC.FINITM bit is set. 0: The filter is disabled; 1: The filter is activated.

CAN filter i register x (CAN_FiRx) (i=0..13;x=1..2)

Address offset: 0x240h, 0x31C

Reset value: undefined

Note: 14 groups of filters: i = 0 ... 13. Each group of filters consists of two 32-bit registers, CAN_FiR[2:1]. Only when the corresponding CAN_FA1.FACx bit is cleared or the CAN_FMC.FINIT bit is set, the corresponding filter register can be modified.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FBC31	FBC30	FBC29	FBC28	FBC27	FBC26	FBC25	FBC24	FBC23	FBC22	FBC21	FBC20	FBC19	FBC18	FBC17	FBC16
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
FBC15	FBC14	FBC13	FBC12	FBC11	FBC10	FBC9	FBC8	FBC7	FBC6	FBC5	FBC4	FBC3	FBC2	FBC1	FBC0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit Field	Name	Description
31:0	FBC[31:0]	Filter bits Identifier pattern Each bit of the register corresponds to the level of the corresponding bit of the expected identifier. 0: the corresponding bit is expected to be dominant; 1: The corresponding bit is expected to be recessive. Mask bit pattern Each bit of the register indicates whether the corresponding identifier register bit must be consistent with the corresponding bit of the expected identifier. 0: Don't care, this bit is not used for comparison; 1: Must match, and the incoming identifier bit must be consistent with the identifier register bit corresponding to the filter.

Notes:

- (1) *According to the different settings of filter bit width and mode, the functions of the two registers in the filter bank are different. For the mapping of filters, function description and association of mask registers, refer to 25.4.5 Section: identifier filtering.*
- (2) *Mask/identifier register in mask mode has the same definition as register bit in identifier list mode.*
- (3) *See for the address of the filter bank register Table 25-4.*

26 Secure Algorithm Co-processor (SAC)

Secure algorithm co-processor supports a variety of algorithms hash cryptographic algorithm acceleration, which can greatly improve encryption and decryption speed compared with pure software algorithms.

The hardware-supported algorithms are as follows:

- Supports the AES symmetric algorithm
 - Supports 128 bits, 192 bits, and 256 bits keys
 - Supports CBC, ECB, and CTR modes
- Supports true random number generation
- Supports SM4 algorithm

Note: For information regarding the performance and use of cryptographic algorithms, please contact the sales representatives of Nsing Technology.

27 Debug Support (DBG)

27.1 Overview

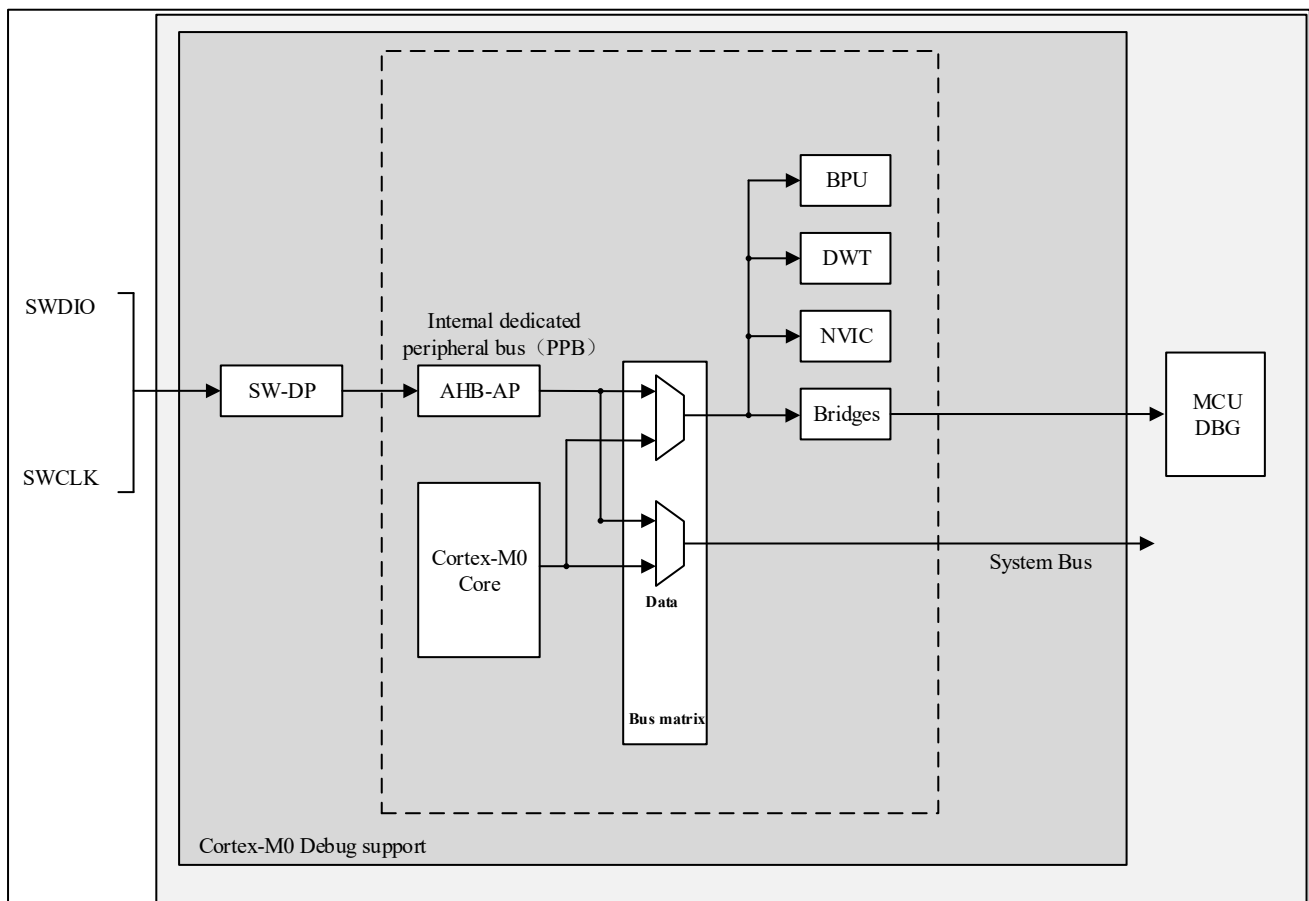
N32G032 uses Cortex[®]-M0 core, which integrates hardware debugging module. Supporting instruction breakpoint (stop when instruction fetches value) and data breakpoint (stop when data access). When the core is stopped, the user can view the internal state of the core and the external state of the system. After the user's query operation is completed, the core and peripherals can be restored, and the corresponding program can continue to be executed.

The hardware debugging module of the N32G032 core can be used when it is connected to the debugger (when it is not disabled).

N32G032 supports the following debugging interfaces:

- Serial interface

Figure 27-1 Block Diagram of N32G032 Level and Cortex[®]-M0 Level Debugging



The ARM Cortex[®]-M0 core hardware debugging module can provide the following debugging functions:

- SW-DP: Serial debugging port
- AHP-AP: AHB access port
- BPU: Breakpoint generation
- DWT: Data watchpoint trigger

Reference:

- Cortex[®]-M0 Technical Reference Manual (TRM)
- ARM debug interface V5
- ARM CoreSight design kit revision r1p0 version technical reference manual

27.2 SWD Function

The debugging tool can call the debugging function through the above-mentioned SWD debugging interface.

27.2.1 Pin Assignment

SWD (serial debug) interface consists of two pins: SWCLK (clock pin) and SWDIO (data input and output pin) to provide two-pin interface.

The pin assignment of SWD debug interface is shown in the following table:

Table 27-1 Debug Port Pin

Debug Port	Pin Assignment
SWDIO	PA13
SWCLK	PA14

28 Unique Device Serial Number (UID)

28.1 Introduction

MCU series products have two built-in unique device serial numbers with different lengths, namely 96-bit UID (Unique device ID) and 128-bit UCID (Unique Customer ID). These two device serial numbers are stored in the system configuration block of the Flash memory, and the information is programmed during manufacture, and any MCU microcontroller is guaranteed to be unique under any circumstances. It can be read by user applications or external devices through CPU or SWD interface and cannot be modified.

UID is 96 bits, which is usually used as serial number or password. When writing Flash memory, this unique identifier is combined with software encryption and decryption algorithm to further improve the security of code in Flash memory, and it can also be used to activate secure bootloader with security function.

UCID is 128 bits and complies with the definition of the Nations Technologies chip serial number. It contains information about chip production and version.

In addition to the above two device serial numbers, there is also a 32-bit DBGMCU_ID, which contains the chip version number, chip model, and Flash/SRAM capacity information.

28.2 UID Register

Start address: 0x1FFF_F4FC, 96 bits in length.

28.3 UCID Register

Start address: 0x1FFF_F4D0, 128 bits in length.

28.4 DBGMCU_ID Register

Start address: 0x1FFF_F508, 32 bits in length. For different bytes, the low byte comes first and the high byte follows; for the same byte, the high byte comes first and the low byte follows.

Table 28-1 DBGMCU_ID Bit Description

Description	Size	Remark
Chip version number	4bit	The lower 4 bits of the chip version number.
	4bit	The upper 4 bits of the chip version number.
Chip model	4bit	The upper 4 bits of the device model number. The device model consists of 12 high, middle and low bits, representing the model of the chip.
	4bit	The middle 4 bits of the device model number.
	4bit	The lower 4 bits of the device model number.
Flash capacity	4bit	Flash capacity indicator. 16KB as unit, Flash size = N * 16KB
SRAM capacity	4bit	SRAM capacity indicator. 4KB as unit, SRAM size = N * 4KB

Reserved	4bit	Keep it all 1.
----------	------	----------------

29 Version History

Version	Date	Changes
V2.3.0	2023.8.23	Initial version

30 Disclaimer

This document is the exclusive property of NSING TECHNOLOGIES PTE. LTD.(Hereinafter referred to as NSING).

This document, and the product of NSING described herein (Hereinafter referred to as the Product) are owned by NSING under the laws and treaties of Republic of Singapore and other applicable jurisdictions worldwide. The intellectual properties of the product belong to Nations Technologies Inc. and Nations Technologies Inc. does not grant any third party any license under its patents, copyrights, trademarks, or other intellectual property rights. Names and brands of third party may be mentioned or referred thereto (if any) for identification purposes only. NSING reserves the right to make changes, corrections, enhancements, modifications, and improvements to this document at any time without notice. Please contact NSING and obtain the latest version of this document before placing orders.

Although NATIONS has attempted to provide accurate and reliable information, NATIONS assumes no responsibility for the accuracy and reliability of this document. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. In no event shall NATIONS be liable for any direct, indirect, incidental, special, exemplary, or consequential damages arising in any way out of the use of this document or the Product.

NATIONS Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, 'Insecure Usage'. Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, all types of safety devices, and other applications intended to supporter sustain life. All Insecure Usage shall be made at user's risk. User shall indemnify NATIONS and hold NATIONS harmless from and against all claims, costs, damages, and other liabilities, arising from or related to any customer's Insecure Usage Any express or implied warranty with regard to this document or the Product, including, but not limited to. The warranties of merch antability, fitness for a particular purpose and non-infringement are disclaimed to the fullest extent permitted by law. Unless otherwise explicitly permitted by NATIONS, anyone may not use, duplicate, modify, transcribe or otherwise distribute this document for any purposes, in whole or in part.