

User Guide

megawin

MG32L003
User Guide

Version 1.0
Date 2023/4/19

Contents

Contents	3
Introduction.....	1
1 Document convention	2
1.1 The list of abbreviations used in the register description table	2
2 Memory and bus architecture	3
2.1 system structure	3
2.2 Storage Address.....	4
3 Embedded Flash (Flash)	6
3.1 Embedded Flash Controller	6
3.2 Functional description	6
3.2.1 Erase operation.....	6
3.2.2 Write operation.....	7
3.2.3 Read operation	7
3.2.4 Erasure time.....	7
3.2.5 Write protection.....	7
3.2.6 PC address erasure protection.....	7
3.2.7 Register write protection.....	8
3.3 System Boot Address Mapping.....	8
3.3.1 Program boot.....	8
3.4 Flash Registers.....	9
3.5 Flash Registers Description.....	9
3.5.1 Flash control register(Flash_CR).....	9
3.5.2 Flash Interrupt Flag Register(Flash_IFR)	10
3.5.3 Flash Interrupt Flag Clear Register(Flash_ICLR).....	10
3.5.4 Flash BYPASS Register(Flash_BYPASS)	11
3.5.5 Flash Sector erase protection register 0(Flash_SLOCK0)	11
3.5.6 Flash Sector erase protection register 1(Flash_SLOCK1)	12
3.5.7 Flash ISPCON Configuration Register(Flash_ISPCON).....	14
4 Cyclic Redundancy Check Unit(CRC)	15
4.1 CRC introduction.....	15

4.2	CRC functional description.....	15
4.2.1	CRC Encoding mode	15
4.2.2	CRC Check mode	15
4.3	CRC Registers	16
4.4	CRC Registers Description	16
4.4.1	CRC Result Register(CRC_RESULT)	16
4.4.2	CRC Data Register((CRC_DATA).....	17
5	Operating Modes and Power Management.....	18
5.1	Active Mode.....	18
5.2	Sleep Mode.....	19
5.2.1	How to enter sleep mode:	19
5.2.2	How to exit sleep mode :	19
5.2.3	Note :	20
5.3	Deep Sleep Mode.....	20
5.3.1	How to enter deep sleep mode :	21
5.3.2	How to exit deep sleep mode :	21
5.4	M0+ Kernel system control register(SCR)	22
6	Reset and Clock(RCC)	23
6.1	Reset 23	
6.1.1	Reset Controller Introduction	23
6.1.2	Reset source POR Reset.....	23
6.2	System clock.....	24
6.2.1	System clock tree.....	25
6.2.2	Internal High Speed RC Clock HSI.....	25
6.2.3	Internal Low Speed RC Clock LSI	26
6.2.4	External High Speed Crystal Clock HSE	26
6.2.5	External Low Speed Crystal Clock LSE.....	26
6.2.6	System clock startup process	26
6.2.7	System Clock Switching.....	27
6.2.8	The specific process of switching from internal high speed to external low speed	27
6.2.9	The specific process of switching from internal high speed to external high speed.....	27
6.2.10	The specific process of switching from internal low speed to external high speed	27
6.2.11	System clock output.....	27
6.2.12	System Clock Security Control.....	28

6.2.13	IWDG Clock	28
6.2.14	RTC Clock.....	28
6.2.15	AWK Clock	28
6.2.16	Low power consumption mode	28
6.3	RCC Registers	29
6.4	Register Description	29
6.4.1	AHB Clock Frequency Division Register(RCC_HCLKDIV)	29
6.4.2	APB Clock Frequency Division Register(RCC_PCLKDIV).....	30
6.4.3	AHB Peripheral module clock enable Register(RCC_HCLKEN).....	30
6.4.4	APB Peripheral module clock Enable register(RCC_PCLKEN)	31
6.4.5	Clock output control register(RCC_MCOCR).....	32
6.4.6	System reset control register(RCC_RSTCR)	33
6.4.7	System resets status register(RCC_RSTSR)	33
6.4.8	System Clock Source Configuration(RCC_SYSCCLKCR).....	34
6.4.9	System Clock Source Selection Register(RCC_SYSCCLKSEL).....	35
6.4.10	Internal High Speed RC Oscillator Control Register(RCC_HSICR)	36
6.4.11	External high speed crystal oscillator control register(RCC_HSECR)	37
6.4.12	Internal low speed RC oscillator control register(RCC_LSICR)	38
6.4.13	External low speed crystal oscillator control register(RCC_LSECR).....	38
6.4.14	M0+ IRQ delay control register(RCC_IRQLATENCY)	40
6.4.15	SysTick Timer Register(RCC_STICKCR)	40
6.4.16	SWDIO port control register(RCC_SWDIOCR)	41
6.4.17	Peripheral module resets the control register(RCC_PERIRST)	41
6.4.18	RTC Reset control register(RCC_RTCRST)	43
6.4.19	Register Write protection control register(RCC_UNLOCK)	44
6.4.20	Internal high speed RC oscillator control register 2(RCC_HSITC)	45
6.4.21	Internal high speed RC oscillator control register 2(RCC_HSITC)	45
6.4.22	Internal Low speed RC oscillator control register 2(RCC_LSITRIM).....	46
7	System Control(SYSCON).....	48
7.1	SYS Registers	48
7.2	Register Description	48
7.2.1	System configuration register 0(SYSCON_CFGR0)	48
7.2.2	Terminal Deep Sleep Interrupt Mode Control Register(SYSCON_PORTINTCR).....	49
7.2.3	Terminal control register(SYSCON_PORTCR).....	49
7.2.4	PCA Capture channel control register(SYSCON_PCACR)	52
7.2.5	TIM1 channel input source selection(SYSCON_TIM1CR).....	53

7.2.6	TIM2 channel input source selection(SYSCON_TIM2CR)	57
7.2.7	Syscon register write protection(SYSCON_UNLOCK)	61
8	Interrupt Controller(NVIC)	62
8.1	Introduction	62
8.2	Features	62
8.3	Interrupt priority	62
8.4	Interrupt vector table	62
8.5	Interrupt wake-up control WIC	64
8.5.1	NVIC wakes up from deep-sleep mode to interrupt ISR setting	64
8.5.2	NVIC wakes up from Deep Sleep mode setting and does not execute interrupt ISR	64
8.5.3	Use the exit sleep feature	64
8.6	Software	65
8.6.1	External interrupt enable	65
8.6.2	NVIC interrupt enable and clear enable	65
8.6.3	NVIC interrupt pending and clear pending	65
8.6.4	NVIC Interrupt Priority	65
8.6.5	NVIC interrupt mask	66
8.7	NVIC Registers	67
8.8	Register Description	68
8.8.1	Interrupt enable set register(NVIC_ISER)	68
8.8.2	Interrupt enables clear register(NVIC_ICER)	68
8.8.3	Interrupt suspends the setting register(NVIC_ISPR)	69
8.8.4	Interrupt suspends the clear register(NVIC_ICPR)	69
8.8.5	Interrupt priority control register 0(NVIC_IPR0)	70
8.8.6	Interrupt priority control register 1(NVIC_IPR1)	70
8.8.7	Interrupt priority control register 2(NVIC_IPR2)	71
8.8.8	Interrupt priority control register 3(NVIC_IPR3)	71
8.8.9	Interrupt priority control register 4(NVIC_IPR4)	72
8.8.10	Interrupt priority control register 5(NVIC_IPR5)	72
8.8.11	Interrupt priority control register 6(NVIC_IPR6)	73
8.8.12	Interrupt priority control register 7(NVIC_IPR7)	73
9	General purpose input and output(GPIO)	74
9.1	GPIO Introduction	74
9.2	GPIO main features	74
9.3	GPIO Function description	74
9.3.1	General purpose I/O(GPIO)	76

9.3.2	I/O port control register.....	77
9.3.3	I/O port data register.....	77
9.3.4	I/O data bit processing.....	77
9.3.5	Input configuration	77
9.3.6	Output configuration	78
9.3.7	External interrupt/wake up line	79
9.3.8	I/O pin alternate function and remapping	79
9.3.9	General input/output (GPIO)	81
9.3.10	Analog configuration	82
9.3.11	HSE or LSE pins are used as GPIO.....	83
9.4	GPIO Registers.....	84
9.5	GPIO Register Description.....	85
9.5.1	GPIO port direction register(GPIOx_DIRCR)(x = A..D)	85
9.5.2	GPIO Port output type register(GPIOx_OTYPER)(x = A..D).....	85
9.5.3	GPIO Port output data register(GPIOx_ODR)(x = A..D)	86
9.5.4	GPIO Port input data register(GPIOx_IDR)(x = A..D).....	87
9.5.5	GPIO Port interrupt enable register(GPIOx_INTEN)(x = A..D).....	88
9.5.6	GPIO Port interrupt original status register(GPIOx_RAWINTSR)(x = A..D).....	89
9.5.7	GPIO Port interrupt status register(GPIOx_MSKINTSR)(x = A..D)	90
9.5.8	GPIO Port interrupt clear register(GPIOx_INTCLR)(x = A..D).....	91
9.5.9	GPIO Port interrupt type register(GPIOx_INTTYPCR)(x = A..D).....	92
9.5.10	GPIO Interrupt type value register(GPIOx_INTPOLCR)(x = A..D)	93
9.5.11	GPIO port arbitrary edge trigger interrupt register(GPIOx_INTANY)(x = A..D).....	94
9.5.12	GPIO Port output set register(GPIOx_ODSET)(x = A..D)	95
9.5.13	GPIO Port output clear register(GPIOx_ODCLR)(x = A..D)	96
9.5.14	GPIO Port input de jitter register(GPIOx_INDBEN)(x = A..D)	97
9.5.15	GPIO Port input de jitter clock configuration register(GPIOx_DBCLKCR)(x = A..D)	97
9.5.16	GPIO Port pull-up/pull-down register(GPIOx_PUPDR)(x = A..D).....	98
9.5.17	GPIO Port voltage conversion rate configuration(GPIOx_SLEWCR)(x = A..D)	100
9.5.18	GPIO Port drive intensity configuration register(GPIOx_DRVCR)(x = A..D).....	101
9.5.19	GPIO Port alternate function register(GPIOx_AFR)(x = A..D).....	102
10	Advanced-control timers(TIM1)	117
10.1	TIM1 introduction.....	117
10.1.1	TIM1 main features	117
10.1.2	TIM1 functional description.....	118
10.1.3	Counter modes	120

10.1.4	Repetition counter	129
10.1.5	Clock selection	130
10.1.6	Capture/compare channels.....	133
10.1.7	Input capture mode.....	135
10.1.8	PWM input mode.....	136
10.1.9	Forced output mode.....	137
10.1.10	Output compare mode.....	137
10.1.11	PWM mode.....	138
10.1.12	Complementary outputs and dead-time insertion	141
10.1.13	Re-directing OCxREF to OCx or OCxN	142
10.1.14	Using the break function	142
10.1.15	Clearing the OCxREF signal on an external event	144
10.1.16	6-step PWM generation	145
10.1.17	One-pulse mode	146
10.1.18	Encoder interface mode.....	148
10.1.19	Timer input XOR function.....	150
10.1.20	Interfacing with Hall sensors.....	150
10.1.21	TIM1 and external trigger synchronization	152
10.1.22	Slave mode: Reset mode.....	152
10.1.23	Slave mode: Gated mode	153
10.1.24	Slave mode: Trigger mode.....	154
10.1.25	Slave mode: external clock mode 2 + trigger mode	155
10.1.26	Timer synchronization.....	155
10.1.27	Debug mode	156
10.2	TIM1 Registers	156
10.3	TIM1 register description	157
10.3.1	TIM1 control register 1(TIM1_CR1)	157
10.3.2	TIM1 control register 2(TIM1_CR2)	158
10.3.3	TIM1 slave mode control register(TIM1_SMCR)	161
10.3.4	TIM1 interrupt enable register(TIM1_DIER)	163
10.3.5	TIM1 status register (TIM1_SR).....	164
10.3.6	TIM1 event generation register(TIM1_EGR)	167
10.3.7	TIM1 capture/compare mode register 1(TIM1_CCMR1)	168
10.3.8	TIM1 capture/compare mode register 2(TIM1_CCMR2)	171
10.3.9	TIM1_CCMR2 capture/compare mode register 2	172
10.3.10	TIM1 capture/compare enable register(TIM1_CCER)	174
10.3.11	TIM1 counter(TIM1_CNT).....	176

10.3.12	TIM1 prescaler(TIM1_PSC)	177
10.3.13	TIM1 auto-reload register(TIM1_ARR).....	177
10.3.14	TIM1 repetition counter register (TIM1_RCR).....	178
10.3.15	TIM1 capture/compare register 1(TIM1_CCR1).....	179
10.3.16	TIM1 capture/compare register2(TIM1_CCR2)	179
10.3.17	TIM1 capture/compare register 3 (TIM1_CCR3).....	180
10.3.18	TIM1 capture/compare register 4(TIM1_CCR4).....	180
10.3.19	TIM1 break and dead-time register(TIM1_BDTR)	182
11	General-purpose(TIM2).....	185
11.1	TIM2 introduction.....	185
11.2	TIM2 main features.....	185
11.3	TIM2 functional description.....	186
11.3.1	Time-base unit.....	186
11.3.2	Prescaler description	186
11.3.3	Counter modes	187
11.3.4	Clock selection	195
11.3.5	PWM input mode.....	200
11.3.6	Forced output mode	201
11.3.7	Output compare mode.....	202
11.3.8	PWM mode	203
11.3.9	One-pulse mode.....	206
11.3.10	Particular case: OCx fast enable:.....	207
11.3.11	Clearing the OCxREF signal on an external event:	207
11.3.12	Encoder interface mode	208
11.3.13	Timer input XOR function.....	210
11.3.14	Timer and external trigger synchronization	211
11.3.15	Slave mode: Reset mode	211
11.3.16	Slave mode: Gated mode	211
11.3.17	Slave mode: Trigger mode.....	212
11.3.18	Slave mode: external clock mode 2 + trigger mode	213
11.3.19	Master/Slave timer example.....	214
11.3.20	Using one timer as prescaler for another timer.....	214
11.3.21	Using one timer to enable another timer.....	215
11.3.22	Using one timer to start another timer.....	216
11.3.23	2 timers synchronously in response to an external trigger.....	217
11.3.24	Debug mode	218

11.4 TIM2 Register	219
11.5 TIM2 Register Description	220
11.5.1 TIM2 control register 1(TIM2_CR1)	220
11.5.2 TIM2 control register 2(TIM2_CR2)	222
11.5.3 TIM2 slave mode control register(TIM2_SMCR)	224
11.5.4 TIM2 interrupt enable register(TIM2_DIER)	227
11.5.5 TIM2 status register(TIM2_SR).....	228
11.5.6 TIM2 event generation register(TIM2_EGR)	230
11.5.7 TIM2 capture/compare mode register 1 (TIM2_CCMR1).....	231
11.5.8 TIM2 capture/compare mode register 2(TIM2_CCMR2)	234
11.5.9 TIM2 capture/compare enable register(TIM2_CCER).....	236
11.5.10 TIM2 counter(TIM2_CNT)	238
11.5.11 TIM2 prescaler(TIM2_PSC)	238
11.5.12 TIM2 auto-reload register(TIM2_ARR).....	239
11.5.13 TIM2 capture/compare register 1(TIM2_CCR1).....	239
11.5.14 TIM2 capture/compare register 2(TIM2_CCR2).....	239
11.5.15 TIM2 capture/compare register 3(TIM2_CCR3).....	240
11.5.16 TIM2 capture/compare register 4(TIM2_CCR4).....	240
12 Programmable Counter Array(PCA).....	242
12.1 PCA introduction.....	242
12.2 PCA functional description	243
12.2.1 PCA Timer/Counter.....	243
12.2.2 Capture function	245
12.2.3 PCA comparison function.....	246
12.3 PCA Module interconnection and control with other modules	249
12.3.1 ECI Interconnection.....	249
12.3.2 PCACAP0	250
12.3.3 PCACAP1/2/3/4	250
12.4 PCA registers	251
12.5 PCA registers description	251
12.5.1 PCA control register (PCA_CR)	251
12.5.2 PCA mode register (PCA_MOD)	253
12.5.3 PCA counter register (PCA_CNT)	253
12.5.4 PCA interrupt clear register (PCA_INTCLR)	254
12.5.5 PCA capture/compare register (PCA_CCAPM0~4)	255
12.5.6 8 MSB bits register of PCA capture/compare (PCA_CCAP0~4H)	256

12.5.7	8 LSB bits register of PCA capture/compare (PCA_CCAP0~4L).....	256
12.5.8	Compare Capture 16 Bit Register (PCA_CCAP0~4).....	257
12.5.9	Compare High Speed Output Flag Register (PCA_CCAPO).....	257
12.5.10	PCA Port output control register (PCA_POOCR)	258
13	Base timer(TIM10/TIM11).....	260
13.1	Base Timer introduction	260
13.2	Base Timer description.....	260
13.2.1	Mode 1 free counting	261
13.2.2	Mode 2 reload	261
13.2.3	Counting function	262
13.2.4	Timing function.....	263
13.2.5	Buzzer Features.....	264
13.3	Base Timer interconnection.....	264
13.3.1	GATE Interconnection	264
13.3.2	Toggle Output Interconnection	265
13.4	Base Timer registers	265
13.5	Base Timer registers description.....	265
13.5.1	Control Register(TIMx_CR)	265
13.5.2	Immediate Reload Register(TIMx_LOAD).....	266
13.5.3	Counter Register(TIMx_CNT).....	267
13.5.4	Raw interrupt status register(TIMx_RAWINTSR).....	267
13.5.5	Interrupt flag register(TIMx_MSKINTSR)	267
13.5.6	Interrupt clear register(TIMx_INTCLR)	268
13.5.7	BackGround Cycle reload register(TIMx_BGLOAD).....	268
14	Low Power Timer(LPTIM).....	270
14.1	LPTIM description.....	270
14.1.1	Counting function	272
14.1.2	Timing function.....	272
14.2	LPTIM Interconnection	272
14.2.1	GATE Interconnection	272
14.2.2	EXT Interconnection	272
14.2.3	TOGGLE Output Interconnection	272
14.3	LPTIM Registers.....	272
14.4	LPTIM Registers description	273
14.4.1	LPTIM Count Value read-only Register (LPTIM_CNTVAL)	273
14.4.2	LPTIM Control Register (LPTIM_CR).....	273

14.4.3 LPTIM Immediate reload register(LPTIM_LOAD)	274
14.4.4 LPTIM Interrupt Register(LPTIM_INTSR)	275
14.4.5 LPTIM Interrupt Clear Register(LPTIM_INTCLR)	275
14.4.6 LPTIM Cycle reload register(LPTIM_BGLOAD).....	275
15 Auto-wake-up Timer	277
15.1 AWK Registers	277
15.2 Register Description	277
15.2.1 AWK timer control register(AWK_CR).....	277
15.2.2 AWK timer reload data register(AWK_RLOAD)	278
15.2.3 AWK timer state register(AWK_SR)	279
15.2.4 AWK timer Interrupt Clear register(AWK_INTCLR)	279
16 BEEP	280
16.1 BEEP introduction	280
16.2 BEEP Functional description.....	280
16.2.1 Buzzer operation.....	280
16.2.2 Buzzer Calibration.....	280
16.3 BEEP Registers.....	280
16.4 Registers description	281
16.4.1 BEEP control register(BEEP_CSR).....	281
17 Independent watchdog (IWDG)	282
17.1 IWDG introduction	282
17.2 IWDG functional description.....	282
17.2.1 Timeout period	282
17.2.2 IWDG Interrupt after overflow.....	283
17.2.3 IWDG reset after overflow	283
17.3 IWDG registers	283
17.4 Registers description	284
17.4.1 IWDG Control Command Register (IWDG_CMDCR)	284
17.4.2 IWDG Configuration Register (IWDG_CFGR)	284
17.4.3 IWDG Counter Reload Register(IWDG_RLOAD)	285
17.4.4 IWDG Counter Value Register(IWDG_CNTVAL).....	285
17.4.5 IWDG Interrupt Status Register(IWDG_SR)	286
17.4.6 IWDG Interrupt Clear register(IWDG_INTCLR)	286
17.4.7 IWDG protection register(IWDG_UNLOCK).....	287
18 Window watchdog (WWDG)	288

18.1 WWDG introduction.....	288
18.2 WWDG main features.....	288
18.3 WWDG BLOCK Diagram	288
18.4 WWDG Basic configuration.....	288
18.5 WWDG functional description	289
18.5.1 WWDG counts	289
18.5.2 WWDG comparison interrupt.....	290
18.5.3 WWDG reset system	290
18.5.4 Window setting limit of WWDG.....	290
18.6 WWDG compare with IWDG	290
18.6.1 Reset Conditions and Reset Delays.....	290
18.6.2 Wake-up function	290
18.7 WWDG Registers	290
18.8 Register description.....	291
18.8.1 WWDG Reload Count Register(WWDG_RLOAD).....	291
18.8.2 WWDG Control Register (WWDG_CR).....	291
18.8.3 WWDG Interrupt Enable Register(WWDG_INTEN).....	292
18.8.4 WWDG Status Register (WWDG_SR)	292
18.8.5 WWDG Interrupt Clear Register(WWDG_INTCLR)	293
18.8.6 WWDG Counter Value Register(WWDG_CNTVAL).....	293
19 UART1/UART2.....	295
19.1 UART introduction	295
19.2 UART Block Diagram	295
19.3 UART Operating Modes	295
19.3.1 Mode 0(synchronous mode, half duplex)	295
19.3.2 Mode 1(asynchronous mode, full duplex)	296
19.3.3 Mode 2(asynchronous mode, full duplex)	297
19.3.4 Mode 3(asynchronous mode, full duplex)	298
19.3.5 Baud rate programming.....	298
19.3.6 Frame error detection	299
19.3.7 Multi-machine communication	299
19.3.8 Automatic Address Recognition	299
19.3.9 Given address.....	300
19.3.10 Transceiver cache	300
19.4 Infrared data association(IrDA)	301
19.4.1 IrDA low power consumption mode.....	302

19.4.2 Transmitter	302
19.4.3 Receiver	302
19.5 Different baud rate frequency division Settings	303
19.6 UART Registers.....	306
19.7 Register Description	306
19.7.1 UART Control Register(UART_SCON)	306
19.7.2 UART Data Register(UARTx_SBUF)	307
19.7.3 UART Address Register(UARTx_SADDR)	308
19.7.4 UART Address Mask Register(UARTx_SADEN).....	308
19.7.5 UART Interrupt Flag Register(UART_INTSR).....	309
19.7.6 UART Interrupt Flag Clear Register(UART_INTCLR).....	309
19.7.7 UART Baud Rate Control Register(UART_BAUDCR)	310
19.7.8 UART IrDA Control Register(UART_IRDACR)	310
20 LPUART	312
20.1 LPUART Introduction.....	312
20.2 LPUART Diagram.....	312
20.3 Operating Modes	313
20.3.1 Mode 0(synchronous mode, half duplex)	313
20.3.2 Mode 1(asynchronous mode, full duplex)	314
20.3.3 Mode 2(asynchronous mode, full duplex)	315
20.3.4 Mode 3(asynchronous mode, full duplex)	316
20.4 Baud rate programming.....	316
20.4.1 Mode 0	316
20.4.2 Mode 1/3	316
20.4.3 Mode 2	317
20.5 Frame error detection	317
20.6 Multi-machine communication.....	317
20.7 Automatic Address Recognition	317
20.8 Given address	318
20.9 Broadcast addresses.....	318
20.9.1 Examples of given address and broadcast address	318
20.10 Transceiver buffer	318
20.10.1 Receive buffer.....	318
20.10.2 Send buffer	319
20.11 Registers.....	319
20.12 Register Description	319

20.12.1	LPUART Data Register(LPUART_SBUF)	319
20.12.2	LPUART Control Register(LPUART_SCON)	320
20.12.3	LPUART Address Register(LPUART_SADDR)	321
20.12.4	LPUART Address Mask Register(LPUART_SADEN).....	322
20.12.5	LPUART Interrupt Flag Register(LPUART_INTSR).....	322
20.12.6	LPUART Interrupt Flag Clear Register(LPUART_INTCLR).....	323
20.12.7	LPUART Baud Rate Control Register(LPUART_BAUDCR).....	323

21 I2C Interface..... 325

21.1	I2C Introduction	325
21.2	I2C main features	325
21.3	I2C Protocol description	325
21.3.1	Data transfer on the I2C bus.....	326
21.3.2	Start bit or Repeated start signal	326
21.3.3	Slave Address Transfer	327
21.3.4	Data transmission	327
21.4	I2C description.....	328
21.5	I2C Operating mode	329
21.5.1	Arbitration and Synchronization Logic.....	330
21.5.2	serial clock generator.....	330
21.5.3	Input filter	331
21.5.4	Address comparator	331
21.5.5	Interrupt generator	331
21.5.6	I2C Master Transmit Mode	331
21.5.7	I2C Master Receive Mode	335
21.5.8	I2C Slave Receive Mode	338
21.5.9	I2C Slave Transmitter Mode.....	342
21.5.10	I2C Miscellaneous Status	344
21.5.11	I2C_SR = 0xF8	345
21.5.12	I2C_SR = 0x00	345
21.6	Operating mode.....	345
21.6.1	initializer	345
21.6.2	Port configuration program	345
21.6.3	Start the master sending function.....	345
21.6.4	Start the master receiving function	346
21.6.5	I2C Interrupt Routine	346
21.6.6	no mode specified.....	346

21.6.7 Master sending status.....	347
21.6.8 Master Receive Status.....	347
21.6.9 Slave receiving state.....	348
21.6.10 Slave sending status.....	350
21.7 I2C Registers.....	351
21.8 I2C Register description	351
21.8.1 I2C Configuration Register(I2C_CR)	351
21.8.2 I2C Data Register(I2C_DATA)	352
21.8.3 I2C Address Register(I2C_ADDR)	352
21.8.4 I2C Status Register(I2C_SR)	352
21.8.5 I2C Baud Rate Counter Enable(I2C_TIMRUN).....	353
21.8.6 I2C Baud Rate Counter Configuration Register(I2C_BAUDCR).....	353
22 Serial peripheral interface(SPI)	355
22.1 SPI introduction	355
22.2 SPI main features	355
22.3 SPI functional description	355
22.3.1 SPI master mode	355
22.3.2 SPI slave mode.....	356
22.4 SPI interrupts.....	358
22.5 Multi-Master/Multi-Slave Mode.....	358
22.6 SPI Registers.....	360
22.7 SPI register description	360
22.7.1 SPI control register(SPI_CR).....	360
22.7.2 SPI slice selection configuration register (SPI_SSN).....	361
22.7.3 SPI state register (SPI_SR).....	361
22.7.4 SPI data register (SPI_DATA)	362
23 One-Wire Interface (OWIRE).....	363
23.1 One-Wire Protocol(One-Wire).....	363
23.1.1 Features.....	363
23.1.2 Advantage.....	363
23.2 Single bus communication process.....	363
23.2.1 Initialization	363
23.2.2 Write gap.....	363
23.2.3 Read gap	364
23.3 Configuration description.....	364
23.3.1 Initial configuration instructions	364

23.3.2	Write data configuration instructions	365
23.4	Registers.....	366
23.5	Registers description	366
23.5.1	One-Wire Module Control Register(OWIRE_CR)	366
23.5.2	One-Wire Input terminal filter control register(OWIRE_NFCR).....	368
23.5.3	One-Wire RESET Control Register(OWIRE_RSTCNT).....	369
23.5.4	One-Wire Presence PuLXT counter register(OWIRE_PRESCNT).....	370
23.5.5	One-Wire Bit rate Design counter(OWIRE_BITRATECNT)	371
23.5.6	One-Wire Master device Read/write time of PULL0 drive (OWIRE_DRVCNT).....	372
23.5.7	One-Wire Master device read sampling time setting (OWIRE_RDSMPCNT)	373
23.5.8	One-Wire Recover Time Counting interval value (OWIRE_REC CNT)	374
23.5.9	One-Wire Data Register (OWIRE_DATA)	375
23.5.10	One-Wire Bus operation command register (OWIRE_CMD).....	376
23.5.11	One-Wire Interrupt enable (OWIRE_INTEN)	377
23.5.12	One-Wire status register(OWIRE_SR).....	378
23.5.13	One-Wire Status clear register(OWIRE_INTCLR)	379
24	Clock Trim/Monitoring module(CLKTRIM)	380
24.1	CLKTRIM introduction	380
24.2	CLKTRIM main features.....	380
24.3	CLKTRIM functional description.....	380
24.3.1	CLKTRIM calibration mode.....	380
24.3.2	CLKTRIM monitoring mode	381
24.4	CLKTRIM registers	382
24.5	CLKTRIM registers description	382
24.5.1	Configuration Register(CLKTRIM_CR)	382
24.5.2	Reference Counter Disposition Configuration Register(CLKTRIM_REFCON).....	383
24.5.3	Reference Counter Value Register(CLKTRIM_REFCNT)	383
24.5.4	Calibration Counter Value Register(CLKTRIM_CALCNT)	383
24.5.5	Interrupt Flag Register(CLKTRIM_IFR).....	384
24.5.6	Interrupt Flag Clear Register(CLKTRIM_ICLR).....	384
24.5.7	Calibration Counter Overflow Value Configuration Register(CLKTRIM_CALCON)	385
25	Real-time clock (RTC)	386
25.1	RTC introduction.....	386
25.2	RTC main features	386
25.3	RTC functional description	386
25.3.1	RTC block diagram.....	386

25.3.2	RTC clock	387
25.3.3	Reset process	387
25.3.4	Register write protection	387
25.3.5	Calendar initialization and configuration	388
25.3.6	Read count register	388
25.3.7	Write count register	389
25.3.8	Alarm clock setting	389
25.3.9	Calibrated 1Hz output	389
25.3.10	RTC clock calibration	389
25.4	RTC interrupt	390
25.4.1	RTC alarm interrupt	390
25.4.2	RTC cycle interrupt	390
25.5	RTC Registers	391
25.6	RTC Registers Description	392
25.6.1	RTC Control Register(RTC_CR)	392
25.6.2	RTC Clock Control Register(RTC_CLKCR)	393
25.6.3	RTC Time Register(RTC_TIME)	394
25.6.4	RTC Date Register (RTC_DATE)	394
25.6.5	RTC Time Alarm Register(RTC_ALM1TIME)	395
25.6.6	RTC Date Alarm Register (RTC_ALM1DATE)	396
25.6.7	RTC Periodic Alarm Clock Register (RTC_ALM2PRD)	397
25.6.8	RTC Clock Trim Register (RTC_CLKTRIM)	397
25.6.9	RTC Initialize and Status Registers (RTC_ISR)	398
25.6.10	RTC Status Clear Register(RTC_INTCLR)	399
25.6.11	RTC Write Protection Register(RTC_WPR)	400
26	Analog-to-digital converter (ADC)	401
26.1	ADC introduction	401
26.2	ADC Block Diagram	401
26.3	Conversion timing and speed	402
26.4	Single Conversion Mode	402
26.5	Continuous Conversion Mode	403
26.6	Continuous transformation cumulative mode	404
26.7	Comparison of ADC conversion results	405
26.8	ADC interrupt	406
26.9	ADC Register	407
26.9.1	ADC Configuration Register(ADC_CR0)	408

26.9.2	ADC Configuration Register 1((ADC_CR1).....	410
26.9.3	ADC Configuration Register 2(ADC_CR2).....	416
26.9.4	ADC channel 0 conversion result(ADC_RESULT0).....	418
26.9.5	ADC channel 1 conversion result(ADC_RESULT1).....	418
26.9.6	ADC channel 2 conversion result(ADC_RESULT2).....	419
26.9.7	ADC channel 3 conversion result(ADC_RESULT3).....	419
26.9.8	ADC channel 4 conversion result(ADC_RESULT4).....	419
26.9.9	ADC channel 5 conversion result(ADC_RESULT5).....	421
26.9.10	ADC channel 6 conversion result(ADC_RESULT6).....	421
26.9.11	ADC channel 7 conversion result(ADC_RESULT7).....	423
26.9.12	ADC conversion result(ADC_RESULT)	423
26.9.13	ADC Conversion result accumulation value(ADC_RESULT_ACC).....	423
26.9.14	ADC comparison high threshold register(ADC_HT).....	425
26.9.15	ADC comparison low threshold register(ADC_LT).....	425
26.9.16	ADC Interrupt enable register(ADC_INTEN).....	426
26.9.17	ADC interrupt clear register(ADC_INTCLR).....	427
26.9.18	ADC Pre-mask interrupt status register(ADC_RAWINTSR).....	429
26.9.19	ADC Post-mask interrupt status register(ADC_MSKINTSR).....	430
26.9.20	ADC channel 8 conversion result(ADC_RESULT8).....	432
26.9.21	ADC channel 9 conversion result(ADC_RESULT9).....	433
26.9.22	ADC channel 10 conversion result(ADC_RESULT10)	433
26.9.23	ADC channel 11 conversion result(ADC_RESULT11)	434
26.9.24	ADC channel 12 conversion result(ADC_RESULT12)	434
26.9.25	ADC channel 13 conversion result(ADC_RESULT13)	434
26.9.26	ADC channel 14 conversion result(ADC_RESULT14)	435
26.9.27	ADC channel 15 conversion result(ADC_RESULT15)	435
27	Low Voltage Detector (LVD)	437
27.1	LVD introduction	437
27.2	LVD Block Diagram	437
27.3	Digital filtering	437
27.4	Configuration example	438
27.4.1	LVD configured as low voltage reset.....	438
27.4.2	LVD configured as voltage change interrupt.....	438
27.5	LVD Registers	439
27.6	LVD Registers Description	439
27.6.1	LVD Control Register(LVD_CR)	439

27.6.2 LVD Status Register(LVD_SR).....	440
28 Voltage Comparator(VCMP)	442
28.1 VCMP introduction.....	442
28.2 VCMP Block Diagram.....	442
28.3 Digital Filtering.....	442
28.4 Configuration example	443
28.5 VCMP Registers	443
28.6 VCMP Registers Description.....	443
28.6.1 Voltage comparator control register(VCMP_CR0)	443
28.6.2 Voltage comparator control register(VCMP_CR1)	444
28.6.3 VCMP Output Configuration Register(VCMP_OUTCFG)	445
28.6.4 VCMP Status Register(VCMP_SR).....	447
29 Option Bytes	448
29.1 Introduction.....	448
29.2 Registers.....	448
29.2.1 User Configuration Register1(USERCFG1)	448
29.2.2 User Configuration Register2(USERCFG2)	448
30 Debug(DBG)	450
30.1 SWD Describes the debugging interface	450
30.1.1 SWD Pin assignment for debug interface	450
30.1.2 Internal pull-up and pull-down for SWD pins.....	450
30.2 Working principle of SWD protected bit	450
30.3 Use SWDS in low power mode	451
30.3.1 Using SWD in sleep mode(Sleep Mode).....	451
30.3.2 Use SWD in Deep Sleep Mode	451
30.4 DBG Registers.....	451
30.5 Debug Mode Control Register(DBG_APBZ).....	451
31 SysTick timer(SYST)	454
31.1 SysTick Introduction	454
31.2 Set SysTick.....	454
31.3 SysTick Register.....	454
31.4 SysTick Register Description	455
31.4.1 SysTickTimer Control and Status Register(SYST_CSR).....	455
31.4.2 SysTickTimer Reload Value Register(SYST_RVR)	455
31.4.3 SysTickTimer current value register(SYST_CVR).....	455

31.4.4 SysTickTimer calibration value register(SYST_CALIB)	455
--	-----

32 Version.....	456
------------------------	------------

Introduction

This reference manual provides detailed information on how to use memory and peripherals of MG32L003xx series microcontroller (MG32L003), including internal structure of each function module, description of all possible functions, usage of various working modes and register configuration to help users solve application problems

MG32L003xx series microcontrollers have different memory capacities, packages and peripheral configurations. The configurations mentioned in this reference manual are only the highest of the series.

For the memory capacity, package and peripheral configuration, electrical and physical performance parameters of the MG32L003xx series, please refer to the data manual of the specific model of the series.

Download the relevant information:

<http://www.megawin.com.tw/>

This document is only used to help users understand and use the product, and does not assume any loss or damage caused by the use of any information in this document or any incorrect use of the product!

1 Document convention

1.1 The list of abbreviations used in the register description table

The following abbreviations are used in the description of the register:

Tab 1.1-1 Abbreviated list

Abbreviation	Full name	Description
R	read-only	Software can only read this bit
W	write-only	Software can only write this bit
RW	read/write	Software can read and write this bit
RC_W0	read/clear write0	Software can read this bit or clear it by writing '0'. Writing '1' has no effect on this bit
RC_W1	read/clear write1	Software can read this bit or clear it by writing '1'. Writing '0' has no effect on this bit
RC_R	read/clear by read	Software can read this bit. When this bit is read, it will be cleared automatically. Writing has no effect on this bit
RS_R	read/set by read	Software can read this bit. When this bit is read, it is automatically set. Writing has no effect on this bit.
RS	read/set	Software can read or set this bit, write '0' has no effect on this bit
S	set	The software can only set this bit. Writing '0' has no effect on this bit
T	toggle	The software can only flip this bit by writing '1'. Writing '0' has no effect on this bit
Res.	Reserved	Reserved bit.

2 Memory and bus architecture

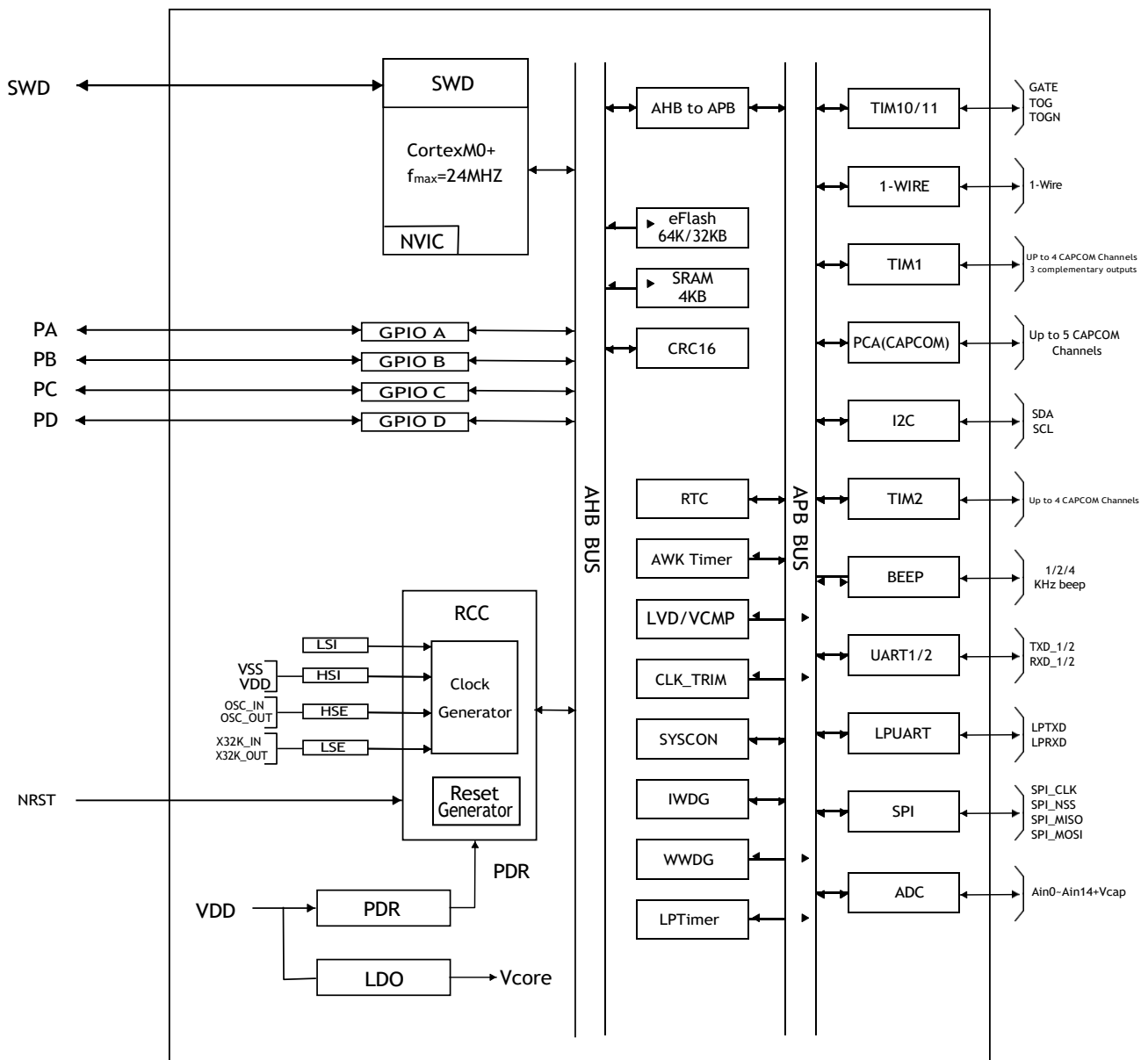
2.1 system structure

The system mainly consists of the following parts:

- 1 AHB Bus master
 - Cortex M0+Core
- 6 AHB slave units:
 - Internal SRAM
 - internal flash
 - AHB to APB bridge to connect all APB peripherals
 - GPIOs
 - Clock and Reset Control
 - CRC module

These are interconnected through a multi-layer bus architecture as follows 2.1-1:

Fig 2.1-1 System architecture diagram



2.2 Storage Address

MG32L003xx embed 64/32KB Flash memory for storing user application code and data. The kernel runs on 24MHz and below,Flash do not need wait cycle.

MG32L003xx embeds 4KB of SRA Mmemory without wait cycles.

Tab 2.2-1 **Storage Address**

Bus	Addresses	Module	Size (bytes)
	0xE000_0000 - 0xE00F_FFFF	M0+ Peripheral	1MB
	0x4003_0000 - 0xDFFF_FFFF	Reserved	
AHB	0x4002_1000 - 0x4002_1FFF	GPIOB	1K
	0x4002_1000 - 0x4002_1BFF	GPIOC	1K
	0x4002_1000 - 0x4002_17FF	GPIOB	1K
	0x4002_1000 - 0x4002_13FF	GPIOA	1K
	0x4002_0C00 - 0x4002_0FFF	Reserved	1K
	0x4002_0800 - 0x4002_0BFF	CRC16	1K
	0x4002_0400 - 0x4002_07FF	FMC	1K
	0x4002_0000 - 0x4002_03FF	RCC	1K
		0x4000_5400 - 0x4001_FFFF	Reserved
APB	0x4000_5000 - 0x4000_53FF	LPUART	1K
	0x4000_4C00 - 0x4000_4FFF	DEBUG	1K
	0x4000_4800 - 0x4000_4BFF	BEEP	1K
	0x4000_4400 - 0x4000_47FF	LPTIM	1K
	0x4000_4000 - 0x4000_43FF	LVD/VCMP	1K
	0x4000_3C00 - 0x4000_3FFF	TIM2	1K
	0x4000_3800 - 0x4000_3BFF	OWIER	1K
	0x4000_3400 - 0x4000_37FF	CLKTRIM	1K
	0x4000_3000 - 0x4000_33FF	RTC	1K
	0x4000_2C00 - 0x4000_2FFF	ADC	1K
	0x4000_2800 - 0x4000_2BFF	AWK	1K
	0x4000_2400 - 0x4000_27FF	IWDT	1K
	0x4000_2000 - 0x4000_23FF	WWDT	1K
	0x4000_1C00 - 0x4000_1FFF	SYSCON	1K
	0x4000_1800 - 0x4000_1BFF	TIM10/11	1K
	0x4000_1400 - 0x4000_17FF	PCA	1K
	0x4000_1000 - 0x4000_13FF	TIM1	1K
	0x4000_0C00 - 0x4000_0FFF	I2C	1K
	0x4000_0800 - 0x4000_0BFF	SPI	1K
	0x4000_0400 - 0x4000_07FF	UART2	1K
	0x4000_0000 - 0x4000_03FF	UART1	1K
AHB	0x2000_1000 - 0x3FFF_FFFF	Reserved	
	0x2000_0000 - 0x2000_0FFF	SRAM	4K
	0x1800_0100 - 0x1FFF_FFFF	Reserved	
	0x1800_0000 - 0x1800_00FF	System configuration	256
	0x0800_0200 - 0x17FF_FFFF	Reserved	
	0x0800_0000 - 0x0800_01FF	Byte option	512
	0x0001_0000 - 0x07FF_FFFF	Reserved	
	0x0000_0000 - 0x0000_FFFF	Main flash memory	64K

3 Embedded Flash (Flash)

3.1 Embedded Flash Controller

MG32L003 contains 1 64K/32K embedded Flash with 128/64 sector of main storage, 5 sectors of NVR area. Each sector has a capacity of 512 bytes. The main storage area of Flash is for users to store programs and data developed by users. In the NVR area, one sector is used to store system configuration values, one sector is used to store user configuration values, and the remaining 2 sectors are used to store system ISP program. This module supports the erase, program and read operations of Flash memory. In addition, this module supports the protection of Flash memory erase and write, and the write protection of control registers. The main memory area is used to store user's code.

NVR sector 0 1 These 2 sectors are used to store the system ISP (programmable in the system) code, the user can download the developed application program through the ISP code, the user program cannot read or erase the ISP code.

3.2 Functional description

MG32L003 supports three bit width mode FLASH reading and writing: Byte (8bits), Half-word (16bits), and Word (32bits). Note that the address of Byte operation must be aligned with Byte and the target address of Half-word operation must be aligned with Half-word. (the lowest bit of the address is 0), the address of the Word operation must be aligned with Word (the lowest two bits of the address are 0). If the address of the read and write operation is not aligned according to the above method, the system will enter the Hard Fault error interrupt.

3.2.1 Erase operation

Sector erase can erase one sector specified by the user (Sector) at a time. After the erase operation is completed, the data in the sector (Sector) is 0xFF. If the erase operation is performed from FLASH, the system will stop fetching instructions and wait for the operation to complete automatically (FLASH_CR. BUSY becomes 0); if the erase operation is performed from RAM, the system will not stop fetching instructions, the user should wait for the operation to complete (FLASH_CR. BUSY becomes 0).

Sector (Sector) erase operation steps are as follows:

1. Configure FLASH_CR. OP to 2, and set Flash operation mode to Sector Erase.
2. Write any data to any address in Sector to be erased, and trigger Sector erase.
3. Wait for FLASH_CR. BUSY to change to 0, and Sector erase operation is completed.

Chip erase can erase all sectors at one time (Sector). After the erase operation is completed, the data in all sectors (Sector) are 0xFF. If the erase operation is performed from FLASH, the system will stop fetching instructions and wait for the operation to complete automatically (FLASH_CR. BUSY becomes 0); if the erase operation is performed from RAM, the system will not stop fetching instructions, and the user should wait for the The operation to complete (FLASH_CR. BUSY becomes 0).

The full chip erase operation steps are as follows:

1. Configure FLASH_CR. OP to 3, and set Flash operation mode to chip erase.

2. Write any address in the chip to be erased to trigger chip erase.
3. Wait for FLASH_CR. BUSY to change to 0, and the chip erase operation is completed.

3.2.2 Write operation

The data of FLASH can only be written from 1 to 0, so before writing data, make sure that the data to be written is 0xFF. It supports writing three data lengths: Byte(8bits), Half-word(16bits), and Word(32bits).The written data is stored in FLASH in Little endian mode, that is,the low byte of the data is stored at the low address.If the write operation is performed from within FLASH,the system will stop fetching instructions and wait for the operation to complete automatically(FLASH_CR.BUSY becomes 0); if the write operation is performed from RAM, the system will not stop fetching instructions, and the user should wait for the operation to complete(FLASH_CR.BUSY becomes 0).

The Byte write operation steps are as follows:

1. Configure FLASH_CR. OP to 1, and set Flash operation mode to write.
2. Perform a Byte write operation on target address to be written to trigger the write operation.
3. Wait for FLASH_CR. BUSY to change to 0, and the write operation is completed.

3.2.3 Read operation

The maximum readout speed of the on-chip Flash is 40ns.

3.2.4 Erasure time

FLASH memory has strict time requirements for the control signals of the erasure and programming operations, and unqualified timing of the control signals will cause the erasure and programming operations to fail. The registers related to the erasure timing parameters are: FLASH_TPROG , FLASH_TSERASE , FLASH_TMERASE .

3.2.5 Write protection

The entire 64K byte Flash memory is divided into 128 sectors,and each 2 Sectors share one erase write protection bit. Protection bit register Flash_SLOCK.SLOCK0/1[31:0] default value is“0000_0000”, that is, erasing and writing are not allowed. Only by modifying the corresponding protection bit to "1", the sector can be erased and written. When Sector is protected, the erasing and writing operations on Sector are invalid and an alarm flag bit and interrupt signal will be generated. When any Sector in the FLASH memory is protected, the Chip erasure of the FLASH is invalid, and an alarm flag bit and interrupt signal will be generated. If chip needs to be erased and written, the protection bit register Flash_SLOCK.SLOCK0/1[31:0] must be modified to “0xFFFFFFFF”.

3.2.6 PC address erasure protection

When CPU runs the program in Flash, if the current running PC pointer happens to fall within the sector address range of software to erase the Flash, the erasing and writing operation will also be automatically shielded by the controller, and an alarm flag bit and an interrupt signal will be generated.

3.2.7 Register write protection

In order to prevent accidental Flash erasing and writing operations from changing the contents of Flash during application, the write operation to the Flash controller register and the write operation to Flash during erasing operation must use the write sequence method to modify.

The registers that can be changed by writing sequence are as follows:

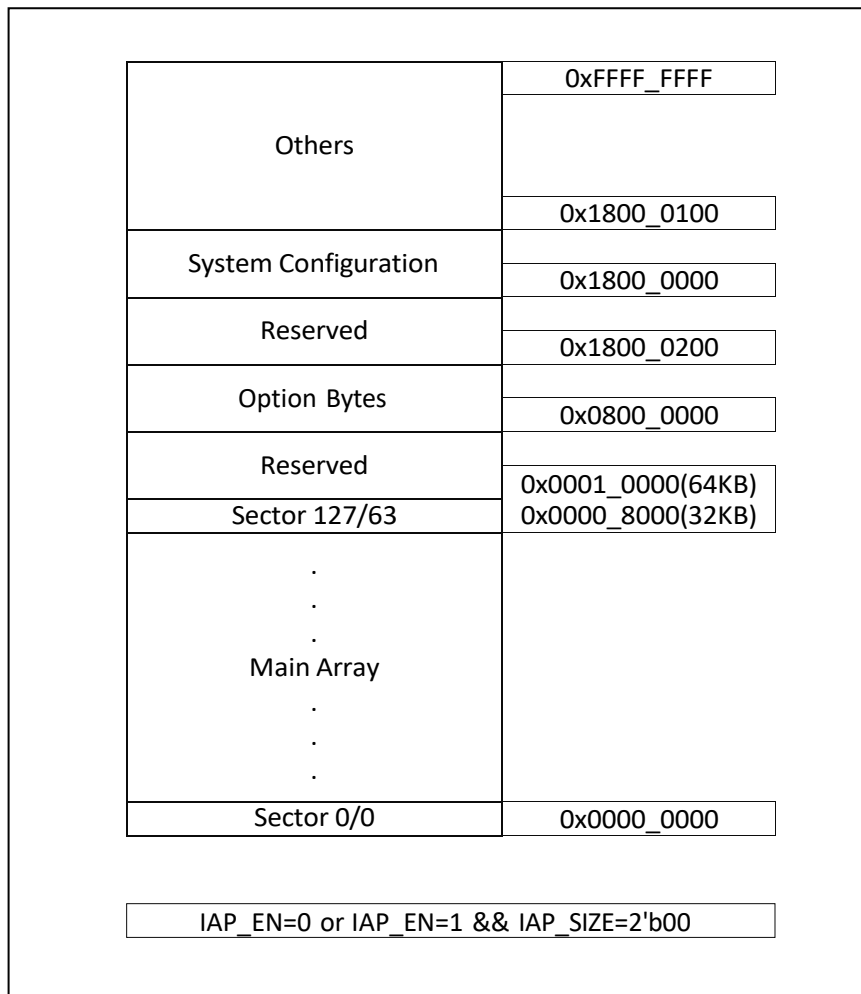
1. Write 0x5A5A to the FLASH_BYPASS register.
2. Write 0xA5A5 to FLASH_BYPASS register.
3. Write the target value to the register to be modified.
4. Write Flash

Note : No write operation can be inserted between the two steps of write 0x5a5a and write 0xa5a5, and cannot be interrupted by interrupts, otherwise the Bypass sequence will be invalid, and the 0x5a5a-0xa5a5 sequence needs to be rewritten.

3.3 System Boot Address Mapping

3.3.1 Program boot

Fig 3.3-1 Boot area address mapping



The boot state of the program area is the initial state. At this time, the main storage area is mapped to the 64K/32K Byte area starting from the logical address 0x0000_0000, and the option byte area is mapped to the 512 Byte area starting from the logical address 0x0800_0000. The low 256 Byte of the system configuration area is mapped to the 256 Byte area starting from 0x1800_0000, the program can only read 256 Bytes.

3.4 Flash Registers

Tab 3.4-1 Flash Registers

Address offset	Register	Reset value	Description
Flash base address:0x4002_0400			
0x00	Flash_CR	0x0000_0000	Flash control register(Flash_CR)
0x04	Flash_IFR	0x0000_0000	Flash Interrupt Flag Register(Flash_IFR)
0x08	Flash_ICLR	0x0000_0000	Flash Interrupt Flag Clear Register(Flash_ICLR)
0x0C	Flash_BYPASS	0x0000_0000	Flash BYPASS Register(Flash_BYPASS)
0x10	Flash_SLOCK0	0x0000_0000	Flash Sector erase protection register 0(Flash_SLOCK0)
0x14	Flash_SLOCK1	0x0000_0000	Flash Sector erase protection register 1(Flash_SLOCK1)
0x18	Flash_ISPCON	0x0000_0001	Flash ISPCON Configuration Register(Flash_ISPCON)

Note: All Flash registers can only be read and written in word mode

3.5 Flash Registers Description

3.5.1 Flash control register(Flash_CR)

Register	Address offset	Access	Reset value	Description
Flash_CR	0x00	RW	0x0000_0000	Flash control register(Flash_CR)

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

Flash control register(Flash_CR)

Bit	Access	Description
[31:5]	-	Reserved, always read as 0.
[4:3]	RW	IE[1]: Flash Erase and write protected address interrupt enable; 0: disable 1: Enable IE[0]: Flash Erase PC Value interrupt enable 0: disable 1: Enable

[2]	R	BUSY: Free/Busy flag; 0: idle; 1: Busy
[1:0]	RW	OP[1:0]: Flash Operation mode register. 2' b00: Read 2' b01: Write 2' b10: Sector Erase 2' b11: Page Erase

3.5.2 Flash Interrupt Flag Register(Flash_IFR)

Register	Address offset	Access	Reset value	Description
Flash_IFR	0x04	R	0x0000_0000	Flash Interrupt Flag Register(Flash_IFR)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						IF1	IF0

Flash Interrupt Flag Register(Flash_IFR)

Bit	Access	Description
[31:2]	-	Reserved, always read as 0.
[1]	R	IF1: Erase and write protection alarm interrupt flag bit
[0]	R	IF0: Erase and write PC Address alarm interrupt flag

3.5.3 Flash Interrupt Flag Clear Register(Flash_ICLR)

Register	Address offset	Access	Reset value	Description
Flash_ICLR	0x08	W	0x0000_0000	Flash Interrupt Flag Clear Register(Flash_ICLR)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						ICLR1	ICLR0

Flash Interrupt Flag Clear Register(Flash_ICLR)

Bit	Access	Description
[31:2]	-	Reserved, always read as 0.

[1]	W	ICLR1: Clear protection alarm interrupt flag: 0 :clear; 1: invalid
[0]	W	ICLR0: Clear PC Address alarm interrupt flag: 0:clear; 1:invalid

3.5.4 Flash BYPASS Register(Flash_BYPASS)

Register	Address offset	Access	Reset value	Description
Flash_BYPASS	0x0C	W	0x0000_0000	Flash BYPASS Register(Flash_BYPASS)

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

Flash BYPASS Register(Flash_BYPASS)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0.
[15:0]	W	BYPASSEQ[15:0]: Before modifying this module, the 0x5A5A, 0xA5A5 sequence must be written to the BYPASSEQ[15:0] register. Only one write to the register is allowed after a correct write to the sequence. If need to modify the register again, must enter it again correct BYPASS sequence.

3.5.5 Flash Sector erase protection register 0(Flash_SLOCK0)

Register	Address offset	Access	Reset value	Description
Flash_SLOCK0	0x10	RW	0x0000_0000	Flash Sector erase protection register 0(Flash_SLOCK0)

31	30	29	28	27	26	25	24
SLOCK0[31:124]							
23	22	21	20	19	18	17	16
SLOCK0[23:16]							
15	14	13	12	11	10	9	8
SLOCK0[15:8]							
7	6	5	4	3	2	1	0
SLOCK0[7:0]							

Flash Sector erase protection register 0(Flash_SLOCK0)

Bit	Access	Description
-----	--------	-------------

[31]	RW	SLOCK0[31]: Sector 62-63 (same below) 0: Erase is not allowed. 1: Erase is allowed.
[30]	RW	SLOCK0[30]: Sector 60-61
[29]	RW	SLOCK0[29]: Sector 58-59
[28]	RW	SLOCK0[28]: Sector 56-57
[27]	RW	SLOCK0[27]: Sector 54-55
[26]	RW	SLOCK0[26]: Sector 52-53
[25]	RW	SLOCK0[25]: Sector 50-51
[24]	RW	SLOCK0[24]: Sector 48-49
[23]	RW	SLOCK0[23]: Sector 46-47
[22]	RW	SLOCK0[22]: Sector 44-45
[21]	RW	SLOCK0[21]: Sector 42-43
[20]	RW	SLOCK0[20]: Sector 40-41
[19]	RW	SLOCK0[19]: Sector 38-39
[18]	RW	SLOCK0[18]: Sector 36-37
[17]	RW	SLOCK0[17]: Sector 34-35
[16]	RW	SLOCK0[16]: Sector 32-33
[15]	RW	SLOCK0[15]: Sector 30-31
[14]	RW	SLOCK0[14]: Sector 28-29
[13]	RW	SLOCK0[13]: Sector 26-27
[12]	RW	SLOCK0[12]: Sector 24-25
[11]	RW	SLOCK0[11]: Sector 22-23
[10]	RW	SLOCK0[10]: Sector 20-21
[9]	RW	SLOCK0[9]: Sector 18-19
[8]	RW	SLOCK0[8]: Sector 16-17
[7]	RW	SLOCK0[7]: Sector 14-15
[6]	RW	SLOCK0[6]: Sector 12-13
[5]	RW	SLOCK0[5]: Sector 10-11
[4]	RW	SLOCK0[4]: Sector 8-9
[3]	RW	SLOCK0[3]: Sector 6-7
[2]	RW	SLOCK0[2]: Sector 4-5
[1]	RW	SLOCK0[1]: Sector 2-3
[0]	RW	SLOCK0[0]: Sector 1-0

3.5.6 Flash Sector erase protection register 1(Flash_SLOCK1)

Register	Address offset	Access	Reset value	Description
Flash_SLOCK1	0x14	RW	0x0000_0000	Flash Sector erase protection register 1(Flash_SLOCK1)

31	30	29	28	27	26	25	24
SLOCK1[31:124]							
23	22	21	20	19	18	17	16
SLOCK1[23:16]							
15	14	13	12	11	10	9	8
SLOCK1[15:8]							
7	6	5	4	3	2	1	0
SLOCK1[7:0]							

Flash Sector erase protection register 1(Flash_SLOCK1)

Bit	Access	Description
[31]	RW	SLOCK1[31]: Sector 126-127 (same below) 0: Erase is not allowed. 1: Erase is allowed.
[30]	RW	SLOCK1[30]: Sector 124-125
[29]	RW	SLOCK1[29]: Sector 122-123
[28]	RW	SLOCK1[28]: Sector 120-121
[27]	RW	SLOCK1[27]: Sector 118-119
[26]	RW	SLOCK1[26]: Sector 116-117
[25]	RW	SLOCK1[25]: Sector 114-115
[24]	RW	SLOCK1[24]: Sector 112-113
[23]	RW	SLOCK1[23]: Sector 110-111
[22]	RW	SLOCK1[22]: Sector 108-109
[21]	RW	SLOCK1[21]: Sector 106-107
[20]	RW	SLOCK1[20]: Sector 104-105
[19]	RW	SLOCK1[19]: Sector 102-103
[18]	RW	SLOCK1[18]: Sector 100-101
[17]	RW	SLOCK1[17]: Sector 98-99
[16]	RW	SLOCK1[16]: Sector 96-97
[15]	RW	SLOCK1[15]: Sector 94-95
[14]	RW	SLOCK1[14]: Sector 92-93
[13]	RW	SLOCK1[13]: Sector 90-91
[12]	RW	SLOCK1[12]: Sector 88-89
[11]	RW	SLOCK1[11]: Sector 86-87
[10]	RW	SLOCK1[10]: Sector 84-85
[9]	RW	SLOCK1[9]: Sector 82-83
[8]	RW	SLOCK1[8]: Sector 80-81
[7]	RW	SLOCK1[7]: Sector 78-79
[6]	RW	SLOCK1[6]: Sector 76-77
[5]	RW	SLOCK1[5]: Sector 74-75
[4]	RW	SLOCK1[4]: Sector 72-73
[3]	RW	SLOCK1[3]: Sector 70-71
[2]	RW	SLOCK1[2]: Sector 68-69
[1]	RW	SLOCK1[1]: Sector 66-67
[0]	RW	SLOCK1[0]: Sector 64-65

Note : If Flash_FULLIRCZE is set to 0 and the Flash capacity is set to 32K Byte, the Flash_SLOCK1 register is invalid

3.5.7 Flash ISPCON Configuration Register(Flash_ISPCON)

Register	Address offset	Access	Reset value	Description
Flash_ISPCON	0x18	RW	0x0000_0001	Flash ISPCON Configuration Register(Flash_ISPCON)

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

Flash ISPCON Configuration Register(Flash_ISPCON)

Bit	Access	Description
[31:1]	-	Reserved, always read as 0.
[0]	RW	<p>ISP_CON: ISP function configuration bit;</p> <p>0: Enable ISP function in BOOTLOAD.</p> <p>1: Skip ISP function switch to application in BOOTLOAD;</p> <p>ISP_CON is only reset by MCURST, CPURST cannot be reset by ISP_CON. After the user modifies ISP_CON, the settings will take effect through CPURST.</p>

Note : When writing ISP_CON of the Flash_ISPCON register, the data of the upper 16 bits must be 16'h5A69.

4 Cyclic Redundancy Check Unit(CRC)

4.1 CRC introduction

The cyclic redundancy check (CRC) computing unit takes the data stream or data module as the input and generates an output number under the control of generating polynomial. This output number is often used to verify the correctness and integrity of data transmission or storage. This module supports the calculation and verification of CRC values.

4.2 CRC functional description

The algorithm of this module complies with the definition of ISO/IEC13239, adopts 16-bit CRC, and the calculation polynomial is:

$$x^{16} + x^{12} + x^5 + x$$

The calculated initial value is 0xFFFF.

The functions of this module include:

1. CRC code and CRC verification
2. Three bit width access modes:8bits,16bits,32bits
3. Examples of input data under 8-bit are 0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77
4. Examples of input data under 16-bit are 0x1100, 0x3322, 0x5544, 0x7766
5. Examples of input data under 32-bit are 0x33221100, 0x77665544

4.2.1 CRC Encoding mode

Encoding mode can encode the raw data to calculate its CRC value, and the operation process is as follows:

1. Write 0xFFFF to CRC_RESULT.RESULT, and initialize CRC calculation.
2. Write the original data to be encoded into the CRC_DATA register. sequentially, according to the organization of 8bits/ 16bits/ 32bits
3. Read CRC_RESULT.RESULT, which is CRC value.

4.2.2 CRC Check mode

The verification mode can verify whether the encoded data has been tampered with, and the operation process is as follows:

1. Write 0xFFFF to CRC_RESULT.RESULT, and initialize CRC calculation.
2. Write the encoded data into the CRC_DATA register sequentially according to the organization of 8bits/16bits/32bits.
3. Read CRC_RESULT.FLAG to determine whether the CRC verification is successful.

Note: When writing the CRC value to the CRC_DATA register according to the 8-digit organization, the lower 8 bits should be written first, and then the higher 8 bits.

4.3 CRC Registers

Tab 4.3-1 CRC Registers

Address offset	Register	Reset value	Description
CRC base address: CRC = 0x4002_0800			
0x04	CRC_RESULT	0x0000_0000	CRC Result Register(CRC_RESULT)
0x80-0xFF	CRC_DATA	0x0000_0000	CRC Data Register((CRC_DATA)

4.4 CRC Registers Description

4.4.1 CRC Result Register(CRC_RESULT)

Register	Address offset	Access	Reset value	Description
CRC_RESULT	0x04	RW	0x0000_0000	CRC Result Register(CRC_RESULT)

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

CRC Result Register(CRC_RESULT)

Bit	Access	Description
[31:17]	-	Reserved, always read as 0.
[16]	R	FLAG: Verification result flag; 0: Verification error. 1: verification correct. Registers[16] are read-only bits and writing to them has no effect. When performing CRC check, read this bit after all data and 16 bits CRC code input data register, it is 1 indicates that the verification is achievement
[15:0]	RW	RESULT: Used to update and save CRC calculation results. Afte roperation,16 bit CRC encoding result will be obtained. According to the standard, after the operation is completed,CRC encoding value of 16 bits is the operation register inversion result, the reading of this register[15:0] will get the inversion of the current[15:0]of this register.

4.4.2 CRC Data Register((CRC_DATA))

Register	Address offset	Access	Reset value	Description
CRC_DATA	0x80-0xFF	RW	0x0000_0000	CRC Data Register((CRC_DATA))

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

CRC Data Register((CRC_DATA))

Bit	Access	Description
[31:0]	RW	CRC_DATA[31:0]: It is used to input the data to be calculated. Address is a range(0x80-0xFF), any address operation in this range will be considered as an operation on this register. The purpose of is to facilitate the software to perform continuous 32 bit data write operations on this register to speed up the operation. supports 8/16/32bit input.

5 Operating Modes and Power Management

The power management module of MG32L003xx is responsible for managing the switch between various working modes of this product, and controlling the working status of each functional module in each working mode. The operating voltage(VDD)of this product is 2.5 ~ 5.5V. This product has the following working modes:

1. Active Mode:The core is running, and the peripheral function modules are running
2. Sleep Mode:The core stops running, and the peripheral function modules run.
3. Deep Sleep Mode:The core stops running, and the high-speed clock stops running.

From active mode, other low-power modes can be entered by executing a software program. From various other low-power modes, it is possible to return to active mode through an interrupt trigger.

Tab 5.0-1 System Reset

Serial number	reset source	operating mode	Sleep mode wakeup	Deep Sleep mode wakeup
1	Power-on/Power-down reset	Y	Y	Y
2	External Reset PinReset	Y	Y	Y
3	IWDG Reset	Y	Y	Y
4	WWDG Reset	Y	Y	N
5	System Software Reset	Y	N	N
6	Low Voltage Detector(LVD)Reset	Y	Y	Y
7	LOCKUP Reset	Y	N	N
8	CPURST Register Reset	Y	N	N
9	MCURST Register Reset	Y	N	N

5.1 Active Mode

The microcontroller MCU is in the running state after the system is powered on and reset, or after waking up from various low-power consumption modes. When the CPU is not required to continue to run, various low-power consumption modes can be utilized to save energy. Users need to select an optimal low-power consumption mode based on the minimum energy consumption, the fastest startup time, and the available wake-up source and other conditions.

Tab 5.1-1 Active Mode Optional module

Active Mode		
Cortex-M0+	TIM1	WWDG
SWD	CRC	AWK
FLASH	Internal low speed clock	1-WIRE
SRAM	Internal high speed clock	LVD
SYSCON	External low speed clock	VCMP
TIM10/11	External high speed clock	ADC
LPTIM	IWDG	UART1/2
PCA	RTC	LPUART
TIM2	BEEP	SPI
I2C	GPIO	CLK_TRIM
PIN	Reset POR/PDR	

Several ways to reduce chip power consumption:

- In the Active Mode, the speed of any system clock(HCLK, PCLK)can be slowed down by programming the prescaler registers(RCC_HCLKDIV, RCC_PCLKDIV). The prescaler can also be used to reduce the clock of peripheral devices before entering sleep mode.
- In the Active Mode, turn off the clock(RCC_HCLKEN, RCC_PCLKEN) that does not use peripherals to reduce power consumption.
- Use the deep sleep mode instead of the sleep mode, because the wake-up time of this product is extremely short(20us), which can also meet the real-time response requirements of the system.

5.2 Sleep Mode

Use WFI instruction to enter the sleep mode. In the sleep mode,CPU stops running, but the system clock,NVIC interrupt processing and non-HCLK-driven peripheral function modules can still work. System enters the Sleep Mode, the port state will not be changed. Before entering sleep mode, please change the state of IO to the state in sleep mode as required.

5.2.1 How to enter sleep mode:

Enter the sleep state by executing the WFI command. Depending on the value of the SLEEPONEXIT bit in the M0+ System Control Register, there are two options for selecting the sleep mode entry mechanism:

1. SLEEP-NOW: if SLEEPONEXIT = 0, when WFI or WFE is executed, the microcontroller enters sleep mode immediately
2. SLEEP-ON-EXIT: If SLEEPONEXIT=1, the microcontroller enters sleep mode as soon as the system exits from the interrupt handler with the lowest priority.

5.2.2 How to exit sleep mode :

If WFI instruction is executed to enter the sleep mode, any peripheral interrupt responded by any high-priority nested vector interrupt controller can wake up the system from the sleep mode.

5.2.3 Note :

1. SLEEP-ON-EXIT=1, automatically enter Sleep after executing the interrupt, and the program does not need to write wfi();
2. SLEEP-ON-EXIT=0,main() executes wfi() and enters Sleep. After the interrupt is triggered and the interrupt program returns to main () after executing the WFI command, it enters Sleep. Wait for a subsequent interrupt to trigger.
3. The SLEEP-ON-EXIT bit does not affect the execution of wfi() instruction. SLEEP-ON-EXIT=0:main() execute wfi() then enter Sleep. After the interrupt is triggered and the interrupt program returns to main() after execution,and then continue to execute;
4. If you enter Sleep during an interrupt, only interrupts with a priority higher than this interrupt can be awakened. Firstly execute the high priority and then the low priority; and interrupts with priority lower than or equal to the current interrupt cannot be wakened.

Tab 5.2-1 Optional Module diagram in Sleep mode

Sleep Mode		
Cortex-M0+	TIM1	WWDG
SWD	CRC	AWK
FLASH	Internal low speed clock	1-WIRE
SRAM	Internal high speed clock	LVD
SYSCON	External low speed clock	VCMP
TIM10/11	External high speed clock	ADC
LPTIM	IWDG	UART1/2
PCA	RTC	LPUART
TIM2	BEEP	SPI
I2C	GPIO	CLK_TRIM
PIN	Reset POR/PDR	

Note : *The gray box is the module that does not work in sleep mode*

5.3 Deep Sleep Mode

Use SLEEPDEEP and WFI instructions to enter the deep sleep mode. In the deep sleep mode,CPU stops running, the high-speed clock is turned off, and low-speed clock can be configured to run or not. Some low-power peripheral modules can be configured to run or not,and NVIC interrupt processing can still work.

1. The system enters the deep sleep mode from the high-speed clock, the high-speed clock is automatically turned off, and the low-speed clock remains in the state before entering deep sleep.
2. The system enters the deep sleep mode from the low-speed clock, and the low-speed clock keeps running. Except for the low-power consumption module, other modules can be closed automatically.
3. When the system clock is switched, all clocks will not be automatically turned off, and the corresponding clocks need to be turned off and turned on by software according to the power consumption and system requirements.

4. When the system enters deep sleep state, the port state will not be changed. Before entering deep sleep, change the state of IO to the state in the deep sleep mode as required.

5.3.1 How to enter deep sleep mode :

First, set the SLEEPDEEP bit in the M0+system control register, and enter the deep sleep state by executing the WFI instruction. Depending on the value of the SLEEPONEXIT bit in the M0+System Control Register, there are two options for selecting the deep-sleep mode entry mechanism: SLEEP-NOW: If SLEEPONEXIT=0, the microcontroller enters sleep mode when WFI or WFE is executed. SLEEP-ON-EXIT: If SLEEPONEXIT=1, the microcontroller enters sleep mode as soon as the system exits from interrupt handler with the lowest priority.

5.3.2 How to exit deep sleep mode :

If executing the WFI instruction to enter the sleep mode, any peripheral interrupt(peripheral module interrupt that can run in Deep Sleep mode)responded by nested vector interrupt controller (NVIC) can wake up the system from sleep mode.

Tab 5.3-1 Optional module in Deep Sleep mode

Deep Sleep Mode		
Cortex-M0+	TIM1	WWDG
SWD	CRC	AWK
FLASH	Internal low speed clock	1-WIRE
SRAM	Internal high speed clock	LVD
SYSCON	External low speed clock	VCMP
TIM10/11	External high speed clock	ADC
LPTIM	IWDG	UART1/2
PCA	RTC	LPUART
TIM2	BEEP	SPI
I2C	GPIO	CLK_TRIM
PIN	Reset POR/PDR	

5.4 M0+ Kernel system control register(SCR)

Register	Address	Access	Reset value	Description
SCR	0xE000 ED10	RW	0x0000_0000	M0+ Kernel system control register(SCR)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			SEVONPEND	RESERVED	SLEEPDEEP	SLEEPONEXI	RESERVED

M0+ Kernel system control register(SCR)

Bit	Access	Description
[31:5]	-	Reserved
[4]	RW	SEVONPEND: 1: each new interrupt suspension will generate an event if used The WFE sleep can be used to wake up the processor
[3]	-	Reserved
[2]	RW	SLEEPDEEP: 1 : WFI() to enter Deep Sleep mode 0: WFI() to enter the Sleep/Idle mode
[1]	RW	SLEEPONEXIT: 1 : The handler enters automatically when it exits exception handling and returns to the program thread Sleep mode (WFI) 0: The feature is disabled
[0]	-	Reserved

After entering deep sleep, there are two options for the system clock after waking up. The clock entering deep sleep is used by default, after the configuration register `RCC_SYSCLKCR.WKBYHSI` is 1, no matter what clock is before entering deep sleep, the internal high-speed clock HSI will be used after waking up. Use external crystal oscillation before entering deep sleep. This setting can speed up the wake-up.

6 Reset and Clock(RCC)

6.1 Reset

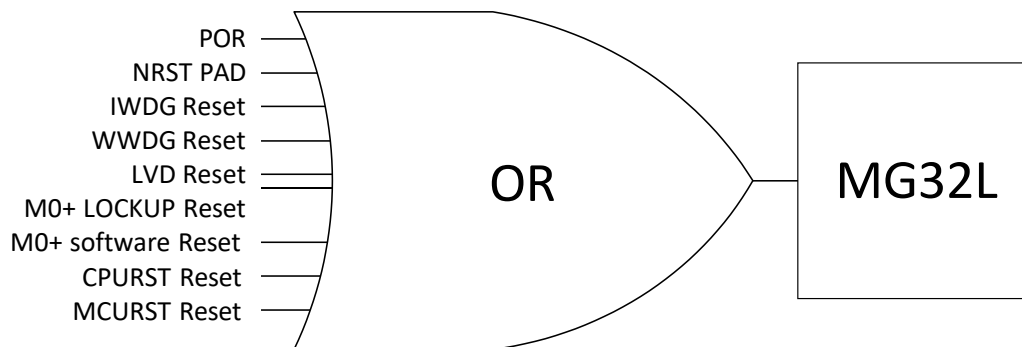
6.1.1 Reset Controller Introduction

MG32L003xx has 9 reset signal sources, each reset signal can make the CPU run again. Most of the registers will be reset to the reset value.

1. POR reset
2. External Reset pin low level reset(NRST PAD)
3. IWDG reset
4. WWDG reset
5. Low Voltage reset(Low Voltage Detector)
6. Software reset(M0+ SYSRESETREQ)
7. M0+ LOCKUP hardware reset
8. CPURST reset
9. MCURST reset

Each reset source has a corresponding reset flag bit to indicate, which is set to "1" by hardware and needs to be cleared by user software. In addition to the POR reset flag bit in the digital area, other reset flag bits can be cleared by the POR in the digital area.

Fig 6.1-1 Reset source diagram



6.1.2 Reset source POR Reset

MG32L003 has two voltage regions, VDD region and Vcore region, so there are two PORs: the POR in VDD region and the POR in Vcore region. When the chip is powered on or powered off, if the power supply voltage is lower than the threshold voltage (typical value is 2.5V), a POR signal will be generated. When the power supply voltage is higher than this threshold voltage, the POR signal is released. The POR signal will reset the registers and control signals of the chip.

External Reset pin reset

When the external reset pin detects low power level, a system reset will be generated. The reset pin has a built-in pull-up resistor and a burr filter circuit. The burr filter circuit will filter the burr signal less than 10us (typical value). Therefore, the low-level signal added to the reset pin must be greater than 10us to ensure the reliable reset of the chip.

M0+Software reset

A software reset can be achieved by setting SYSRESETREQ in M0+ System Control Register.

Register reset

The chip can be reset by writing to the RCC_RSTCR register.

M0+ LOCKUP hardware reset

When M0+ encounters a serious exceptions, it stops its PC pointer at the current address, locks itself, and resets the entire Kernel area after several clock cycles delay.

6.2 System clock

The clock module mainly controls the system clock and the peripheral clock. Different clock sources can be configured as the system clock, different system clock frequency divisions can be configured, and the peripheral clock can be enabled or disabled. In addition, in order to ensure high precision, the internal clock has a calibration function.

This product supports the following four different clock sources as the system clock:

1. High-speed internalRC Clock HSI(4MHz)(Default Dominant Frequency)
2. Low-speed external crystal oscillator clock LSE(32.768KHz)
3. Low-speed internalRC Clock LSI (38.4KHz and 32.768KHz Configurable)
4. High-speed external crystal clock HSE(4MHz~24MHz)

LSE, HSE can be input from outside through terminals PB5,PA1. When using an external oscillator input, the corresponding oscillator needs to be enabled. Select external oscillator control selection in RCC_SYCLKCR, RCC_LSECR registers.

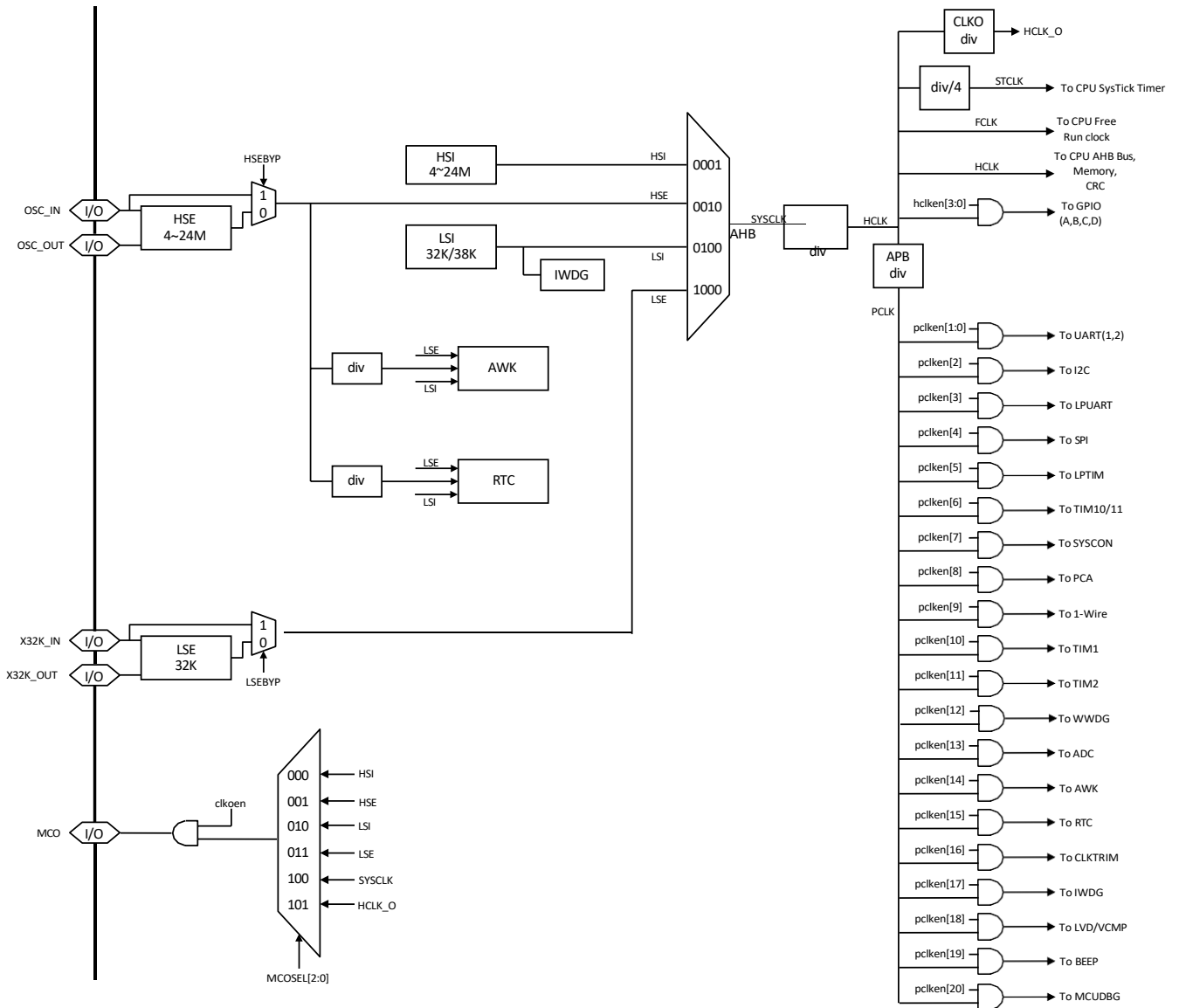
Each clock source can be turned on or off independently, and when they are not in use, they can be turned off to reduce power consumption.

There are multiple dividers available to configure the AHB and APB clock domains, AHB and APB domains with a maximum clock frequency of 24MHz. M0+ SysTick Timer driven by AHB clock, which can be directly driven by AHB/4 or AHB clock frequency(Configure via SYST_CSR.CLKSOURCE). RCC can use 4 frequency division of AHB clock(HCLK) as external clock for SysTick timer, The above clocks or Cortex clocks(HCLK) can be selected as SysTick Clock.

When switching the clock source of the system clock, please strictly follow the operation steps to switch.

6.2.1 System clock tree

Fig 6.2-1 System clock tree



6.2.2 Internal High Speed RC Clock HSI

The default clock source after the chip is powered on or reset is the internal high-speed clock with a frequency of 4MHz.

MG32L003 F8 : The output frequency of HSI can be adjusted by changing the values of register **RCC_HSICR** [11:0] and **RCC_HSI TC**[3:0].

MG32L003 K8 : The output frequency of HSI can be adjusted by changing the values of register **RCC_HSICR** [11:0] and **RCC_HSI TC**[7:0].

Five frequencies of 4MHz, 8MHz, 16MHz, 22.12MHz and 24MHz have been adjusted before delivery ; The internal high-speed clock only needs 20us from startup to stabilization. In order to respond to interrupts quickly in the deep sleep mode, it is recommended to switch the system clock to HSI before entering the deep sleep mode.

6.2.3 Internal Low Speed RC Clock LSI

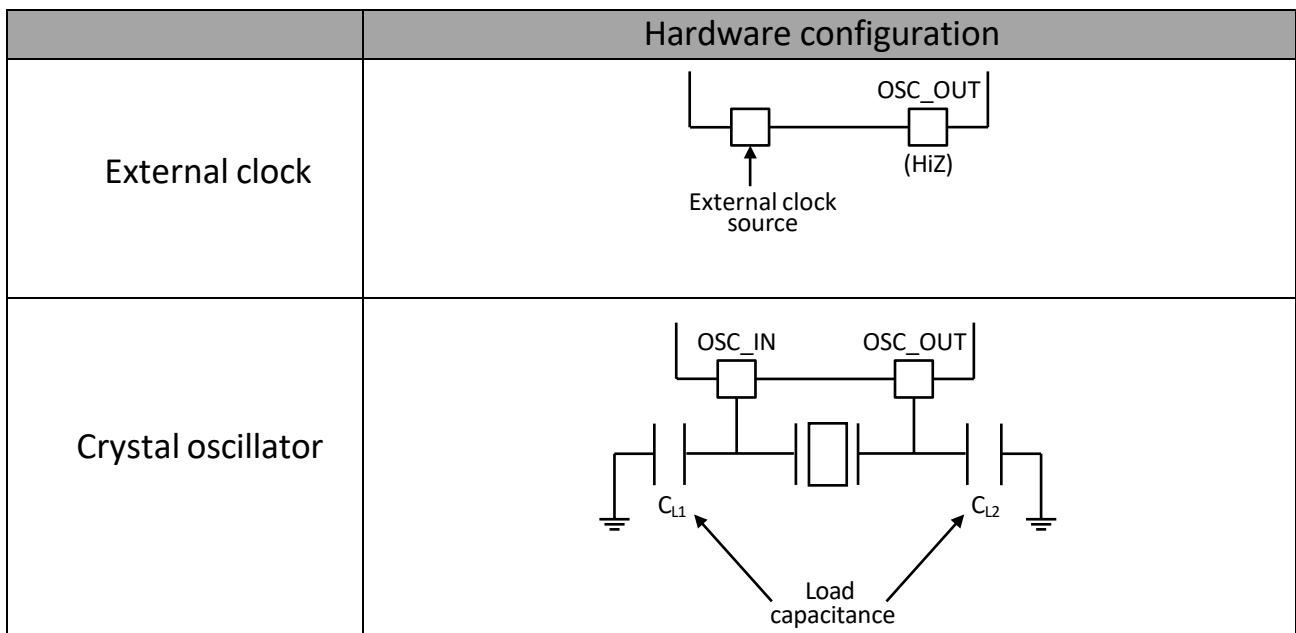
The internal low-speed RC clock frequency can be configured as 32.768/38.4KHz, and this clock source can be used as the system clock in low-speed or low-precision application environment.

6.2.4 External High Speed Crystal Clock HSE

The external high-speed crystal oscillator clock can be connected with a 4MHz~24MHz high-speed crystal oscillator, or input clock signal.

To reduce clock distortion and start-up time, the resonator and load capacitor need to be as close as possible to the oscillator pin. The load capacitance value must be adjusted according to the selected oscillator.

Fig 6.2-2 HSE/LSE



6.2.5 External Low Speed Crystal Clock LSE

The external low-speed crystal oscillator clock needs an external 32.768KHz low-power crystal oscillator, or input clock signal.

6.2.6 System clock startup process

The above four clock sources all have a start-up stabilization time. After the clock source is enabled, it will wait for a period of stabilization time before sending the clock to the system for use.

6.2.7 System Clock Switching

The clock source of the system clock can be switched between four clock sources via `RCC_SYSCLKSEL[3:0]`. The clock switch operation must be carried out according to the standard clock switching process, otherwise abnormalities may occur.

6.2.8 The specific process of switching from internal high speed to external low speed

The specific process is as follows :

1. When clock switch to LSE, the pin that LSE will use is configured as an analog port using the `RCC_LSECR.LSEPORT` bit or configured as analog function for the pin with `GPIOx_AFR = 0x0F`.
2. Write `RCC_LSECR.LSEEN` to enable LSE,
3. Wait for register `RCC_LSECR.LSERDY` bit to become "1",
4. Write register `RCC_SYSCLKSEL.CLKSW[3:0]` to switch clock
5. Turn off HSI clock as required

6.2.9 The specific process of switching from internal high speed to external high speed

The specific process is as follows :

1. When clock switch to HSE, the pin that HSE will use is configured as an analog port using the `RCC_HSECR.HSEPORT` bit or configured as analog function for the pin with `GPIOx_AFR = 0x0F`.
2. Write register `RCC_SYSCLKCR.HSEEN` enable HSE,
3. Wait for register `RCC_HSECR.HSERDY` bit to become "1",
4. Write register `RCC_SYSCLKSEL.CLKSW[3:0]` to switch the clock,
5. Turn off HSI clock as required

When using an external high-speed 24M crystal oscillator, `RCC_HSECR.HSESTARTUP` stabilization time control bit is set to 0x3, use the default configuration 0x2 stability time will not be enough

6.2.10 The specific process of switching from internal low speed to external high speed

The specific process is as follows :

1. When clock switch to HSE, the pin that HSE will use is configured as an analog port using the `RCC_HSECR.HSEPORT` bit or configured as analog function for the pin with `GPIOx_AFR = 0x0F`.
2. Write register `RCC_SYSCLKCR.HSEEN` enable HSE,
3. Wait for register `RCC_HSECR.HSERDY` bit to become "1",
4. Write register `RCC_SYSCLKSEL.CLKSW[3:0]` to switch the clock
5. Turn off LSI clock as required

6.2.11 System clock output

Allow output clock signal to external MCO pin. There are following 6 signals that can be selected as MCO clock output:

- LSE

- HSE
- HSI
- LSI
- SYSCLK
- FCLK and its frequency division output

The MCO clock selection is determined by the MCOSEL[2:0] bit of the control register(RCC_MCOCR) .

6.2.12 System Clock Security Control

After CLKFAILLEN is set to be effective and CLKTRIM clock monitoring function is enabled, when HSE clock or LSE clock stops, the system clock will switch to the internal high-speed clock.(HSI)

6.2.13 IWDG Clock

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

6.2.14 RTC Clock

RTCCLK clock source can be provided by HSE frequency division ,LSE or LSI clock.

6.2.15 AWK Clock

AWKCLK clock source can be provided by HSE frequency division,LSE or LSI clock.

6.2.16 Low power consumption mode

The APB peripheral clock and some AHB peripheral clocks can be disabled by software. The CPU clock is stopped in sleep mode, and the memory interface clock(Flash and RAM interface) is stopped in CPU sleep.

When SYSCON_CFGR0.DBGDLSP_DIS is configured, then CPU can also have debugging function in the corresponding deep sleep mode.

6.3 RCC Registers

Tab 6.3-1 RCC Registers map

Offset	Register	Reset value	Description
RCC base address:0x4002_0000			
0x00	RCC_HCLKDIV	0x0000_0000	AHB Clock Frequency Division Register
0x04	RCC_PCLKDIV	0x0000_0000	APB Clock Frequency Division Register
0x08	RCC_HCKEN	0x0000_0100	AHB Peripheral module clock enable Register
0x0C	RCC_PCKEN	0x0000_0000	APB Peripheral module clock Enable register
0x10	RCC_MCOCR	0x0000_0000	Clock output control register
0x18	RCC_RSTCR	0x0000_0000	System reset control register
0x1C	RCC_RSTSR	0x0000_00A0	System resets status register
0x20	RCC_SYSCLKCR	0x0000_0001	System Clock Source Configuration
0x24	RCC_SYSCLKSEL	0x0000_0001	System Clock Source Selection Register
0x28	RCC_HSICR	0x0000_1312	Internal High Speed RC Oscillator Control Register
0x2C	RCC_HSECR	0x0000_0027	External high speed crystal oscillator control register
0x30	RCC_LSICR	0x0000_007F	Internal low speed RC oscillator control register
0x34	RCC_LSECR	0x0000_042F	External low speed crystal oscillator control register
0x38	RCC_IRQLATENCY	0x0000_0000	M0+ IRQ delay control register
0x3C	RCC_STICKCR	0x0100_9C3F	SysTick Timer Register
0x40	RCC_SWDIOCR	0x0000_0001	SWDIO port control register
0x44	RCC_PERIRST	0x0000_0000	Peripheral module resets the control register
0x48	RCC_RTCRST	0x0000_0000	RTC Reset control register
0x60	RCC_UNLOCK	0x0000_0000	Register Write protection control register
0x394	RCC_HSITC (MG32L003 F8)	0x0000_0009	Internal high speed RC oscillator control register 2
0x394	RCC_HSITC (MG32L003 K8)	0x0000_00FF	Internal high speed RC oscillator control register 2
0x39c	RCC_LSITRIM (MG32L003 K8)	0x0000_00ff	Internal Low speed RC oscillator control register 2

6.4 Register Description

6.4.1 AHB Clock Frequency Division Register(RCC_HCLKDIV)

Register	Address offset	Access	Reset value	Description
RCC_HCLKDIV	0x00	RW	0x0000_0000	AHB Clock Frequency Division Register

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

AHB Clock Frequency Division Register(RCC_HCLKDIV)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	AHBCKDIV[7:0]: System HCLK Clock Divider 0 : HCLK=SYSCLK 1~255 : HCLK = SYSCLK/(2×AHBCKDIV)

6.4.2 APB Clock Frequency Division Register(RCC_PCLKDIV)

Register	Address offset	Access	Reset value	Description
RCC_PCLKDIV	0x04	RW	0x0000_0000	APB Clock frequency division register

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

APB Clock frequency division register(RCC_PCLKDIV)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	APBCKDIV[7:0]: System PCLK Clock frequency division(max1/16) 0 : PCLK = HCLK 1~255 : PCLK = HCLK/(2×APBCKDIV)

6.4.3 AHB Peripheral module clock enable Register(RCC_HCLKEN)

Register	Address offset	Access	Reset value	Description
RCC_HCLKEN	0x08	RW	0x0000_0100	AHB Peripheral module clock enable Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							FlashCKE
7	6	5	4	3	2	1	0
Reserved			CRCCKE	GPIODCKE	GPIOCCKE	GPIOBCKE	GPIOACKE

AHB Peripheral module clock enable Register(RCC_HCLKEN)

Bit	Access	Description
[31:9]	-	Reserved

[8]	RW	FlashCKE: Flash control module clock enabled After closing the Flash configuration register cannot be written, the program in Flash can run. 0:Off , 1 :enabled
[7:5]	-	Reserved
[4]	RW	CRCCKE: Enabling CRC clock (the same as below) 0: Clock is off 1: Clock is enabled
[3]	RW	GPIODCKE: Enabling GPIOD clock
[2]	RW	GPIOCCKE: Enabling GPIOC clock
[1]	RW	GPIOBCKE: Enabling GPIOB clock
[0]	RW	GPIOACKE: Enabling GPIOA clock

6.4.4 APB Peripheral module clock Enable register(RCC_PCLKEN)

Register	Address offset	Access	Reset value	Description
RCC_PCLKEN	0x0C	RW	0x0000_0000	APB Peripheral module clock Enable register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			DBGCKEN	BEEPCKEN	LVDVCCKEN	IWDGCKEN	CLKTRIMCKEN
15	14	13	12	11	10	9	8
RTCCKEN	AWKCKEN	ADCCKEN	WWDGCKEN	TIM2CKEN	TIM1CKEN	OWIRECKEN	PCACKEN
7	6	5	4	3	2	1	0
SYSCON CKEN	BASETIM CKEN	LPTIM CKEN	SPI CKEN	LPUART CKEN	I2CC KEN	UART1CKEN	UART1CKEN

APB Peripheral module clock Enable register(RCC_PCLKEN)

Bit	Access	Description
[31:21]	-	Reserved
[20]	RW	DBGCKEN: Debug PCLK clock enable (same below) 0: Clock is off 1: Clock is enabled
[19]	RW	BEEPCKEN: Enabling BEEP PCLK clock
[18]	RW	LVDVCCKEN: Enabling LVD/VCMP PCLK clock
[17]	RW	IWDGCKEN: Enabling IWDG PCLK clock
[16]	RW	CLKTRIMCKEN: Enabling CLKTRIM PCLK clock
[15]	RW	RTCCKEN: Enabling RTC PCLK clock
[14]	RW	AWKCKEN: Enabling AWK PCLK clock
[13]	RW	ADCCKEN: Enabling ADC PCLK clock
[12]	RW	WWDGCKEN: Enabling WWDG PCLK clock
[11]	RW	TIM2CKEN: Enabling TIM2 PCLK clock
[10]	RW	TIM1CKEN: Enabling TIM1 PCLK clock
[9]	RW	OWIRECKEN: Enabling 1-WIRE PCLK clock
[8]	RW	PCACKEN: Enabling PCA PCLK clock
[7]	RW	SYSCONCKEN: Enabling SYSCON PCLK clock

[6]	RW	BASETIMCKEN: Enabling TIM10/11 PCLK clock
[5]	RW	LPTIMCKEN: Enabling Low Power Timer PCLK clock
[4]	RW	SPICKEN: Enabling SPI PCLK clock
[3]	RW	LPUARTCKEN: Enabling Low Power UART PCLK clock
[2]	RW	I2CCKEN: Enabling I2C PCLK clock
[1]	RW	UART2CKEN: Enabling UART2 PCLK clock
[0]	RW	UART1CKEN: Enabling UART1 PCLK clock

6.4.5 Clock output control register(RCC_MCOCR)

Register	Address offset	Access	Reset value	Description
RCC_MCOCR	0x10	RW	0x0000_0000	Clock output control register

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

Clock output control register(RCC_MCOCR)

Bit	Access	Description
[31:13]	-	Reserved
[12]	RW	<p>MCOEN: MCO output enable</p> <p>Write:</p> <p>0: MCO output is disabled</p> <p>1: MCO output is enabled</p> <p>Read:</p> <p>0: The MCO does not start output</p> <p>1: The MCO starts to output</p> <p>This bit is read after the output enable signal is synchronized through the output clock</p>
[11]	-	Reserved
[10:8]	RW	<p>MCOSEL[2:0]: Select the clock output source</p> <p>000: HSI</p> <p>001: HSE</p> <p>010: LSI</p> <p>011: LSE</p> <p>100: SYSCLK</p> <p>101: HCLK and its frequency division output</p> <p>110,111: Reserved</p>
[7:0]	RW	<p>MCO DIV[7:0]: HCLK clock frequency division coefficient</p> <p>0: HCLK</p> <p>1~ 255: HCLK_O = HCLK/(2 x DIV)</p>

6.4.6 System reset control register(RCC_RSTCR)

Register	Address offset	Access	Reset value	Description
RCC_RSTCR	0x18	RW	0x0000_0000	System reset control register

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

System reset control register(RCC_RSTCR)

Bit	Access	Description
[31:2]	W	RSTKEY To be valid, 0x156A99A6 (0x55AA6699»2) must be written to RCC_RSTCR[1:0]. Other values are invalid.
[1]	RW	CPURST : CPU register reset 0: Normal , 1: Reset CPU
[0]	RW	MCURST : MCU register reset 0: Normal , 1: Reset MCU

Note: Reset the product by writing 0x55AA6699 to RCC_RSTCR

The RCC_UNLOCK register can be written only after its protection is released.

6.4.7 System resets status register(RCC_RSTSR)

Register	Address offset	Access	Reset value	Description
RCC_RSTSR	0x1C	RW	0x0000_00A0	System resets status register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							SFTRST
7	6	5	4	3	2	1	0
PADRST	LOCKUPRST	PORRST	LVDRST	IWDGRST	WWDGRST	CPURST	MCURST

System resets status register(RCC_RSTSR)

Bit	Access	Description
[31:9]	-	Reserved
[8]	RW	SFTRST : M0+ CPU software reset flag, software clear 0: None M0+ CPU software reset occurred 1: The M0+ CPU software resets
[7]	RW	PADRST : RESET Indicates that the port can be reset by the POR 0: no port reset occurs 1: port reset occurs

[6]	RW	LOCKUPRST: M0+ CPU Lockup Reset flag 0: no M0+ CPU Lockup reset occurs 1: M0+ CPU Lockup reset occurs
[5]	RW	PORRST: Vcore Domain POR Reset Flag 0: None Vcore Domain POR Reset occurs 1:Vcore Domain POR Reset occurs
[4]	RW	LVDRST: LVD Reset Flag 0:LVD No reset 1:LVD reset occurs
[3]	RW	IWDGRST: IWDG Reset Flag 0:IWDG No reset 1:IWDG Reset occurs
[2]	RW	WWDGRST: WWDG Reset Flag 0:WWDG No reset 1:WWDG Reset occurs
[1]	RW	CPURST: Register CPU Reset flag 0: Register CPURST No reset 1: Register CPURST with reset
[0]	RW	MCURST: Register MCUR eset flag 0: Register MCURST No reset 1: Register MCURST with reset

Note : Controlled by POR, it can only be set "1" by hardware, and cleared by software "0".

6.4.8 System Clock Source Configuration(RCC_SYSCCLKCR)

Register	Address offset	Access	Reset value	Description
RCC_SYSCCLKCR	0x20	RW	0x0000_0001	System Clock Source Configuration

31	30	29	28	27	26	25	24
KEY							
23	22	21	20	19	18	17	16
KEY							
15	14	13	12	11	10	9	8
WKBYHSI	Reserved						CLKFAILEN
7	6	5	4	3	2	1	0
Reserved	HSEPORT	HSEBYP	Reserved		LSIEN	HSEEN	HRCEN

System Clock Source Configuration(RCC_SYSCCLKCR)

Bit	Access	Description
[31:16]	W	KEY : High bit writes 0x5A69 to configure this register to be valid, other values are invalid.
[15]	RW	WKBYHSI: 0: wake up from Deep Sleep,system clock source enter same as out 1: Wake up from Deep Sleep is to use HSI to wake up, the hardware automatically enable HSI, and system clock automatically switch to HSI, original clock Keep on.

[14:9]	-	Reserved
[8]	RW	CLKFAILEN: Clock fail detect enable control 0: Clock fail detection disabled 1: Clock failure detection enable, when the clock failure is detected, automatically switch the system clock to HSI
[7]	-	Reserved
[6]	RW	HSEPORT: OSCIN/OSCOUTT erminal Configuration 0:GPIO function mode(digital function). 1:HSE terminal mode(analog function).
[5]	RW	HSEBYP: External high-speed clock input selection 0:HSE Internal oscillator module not bypassed, connected to OSC_IN/OSC_OUT 1:HSE internal oscillation module is bypassed,HSE input directly from terminal OSCIN
[4:3]	RW	Reserved
[2]	RW	LSIEN: Internal low-speed clock LSI Enable signal. 0: Off 1: Enable When the system clock selects this clock, it cannot be turned off
[1]	RW	HSEEN: External 4M 24M Crystal oscillator HSEOSC Enable signal. 0: Off 1: Enable Note: 1. When the system enters Deep Sleep, the high-speed clock will be turned off automatically. Remark: When is used, the two external ports connected to this crystal must be set as analog ports (configure RCC_SYSCCLKCR.HSEPORT register). 2. When HSE stop detection, this bit will be cleared by hardware 0 When the system clock is selected The clock cannot be turned off.
[0]	RW	HSIEN: Internal high-speed clock HSI Enable signal. 0: Off 1: Enable Note: 1, When the system entersdeep sleep, the high-speed clock will be turned off automatically. 2, When HSE stop detection, if the system clock selection is HSE, and CLKFAIL_EN is enabled,HSI_EN will be automatically set 1 by hardware. 3, When the system clock selects this clock, it cannot be turned off

Note : This register can only be written after the RCC_UNLOCK register is unprotected

6.4.9 System Clock Source Selection Register(RCC_SYSCCLKSEL)

Register	Address offset	Access	Reset value	Description
RCC_SYSCCLKSEL	0x24	RW	0x0000_0001	System Clock Source Selection Register

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

System Clock Source Selection Register(RCC_SYSCLOCKSEL)

Bit	Access	Description
[31:16]	W	KEY : High bit writes 0x5A69 to configure this register to be valid, other values are invalid.
[15:4]	-	Reserved
[3:0]	RW	CLKSW[3:0] :System clock source selection. 0001:HSI 0010:HSE 0100:LSI 1000:LSE When HSE stops detection, if the system clock is selected asHSE, CLKFAIL_EN is enabled,HSI_EN will be set 1 by hardware. System clock auto selection HSI.

Note : This bit is write protected by RCC_UNLOCK

6.4.10 Internal High Speed RC Oscillator Control Register(RCC_HSICR)

Register	Address offset	Access	Reset value	Description
RCC_HSICR	0x28	RW	0x0000 1312	Internal High Speed RC Oscillator Control Register

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

Internal High Speed RC Oscillator Control Register(RCC_HSICR)

Bit	Access	Description
[31:16]	W	KEY : High bit writes 0x5A69 to configure this register to be valid, other values are invalid.
[15:13]	-	Reserved
[12]	R	HSIRDY : HSI Clock stable flag. 0:HSI is not stable and cannot be used by internal circuits. 1:HSI has stabilized and can be used by the internal circuit.

[11:0]	RW	<p>HSITC[11:0] :Master frequency clock frequency calibration</p> <p>When leaving the factory, the frequency calibration value is saved in Flash, the user needs to write the value of Flash into RCC_HSICR.HSITC can configure the main frequency clock.</p> <ul style="list-style-type: none"> • 24M calibration address:0x1800_00A0 • 22.12M calibration address:0x1800_00A2 • 16M calibration address:0x1800_00A4 • 8M calibration address:0x1800_00A6 • 4M calibration address:0x1800_00A8
--------	----	--

Note :

1. This register is not controlled by M0+ software reset.
2. This register can only be written after theRCC_UNLOCKregister is unprotected.

6.4.11 External high speed crystal oscillator control register(RCC_HSECR)

Register	Address offset	Access	Reset value	Description
RCC_HSECR	0x2C	RW	0x0000 0027	External high speed crystal oscillator control register

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

External high speed crystal oscillator control register(RCC_HSECR)

Bit	Access	Description
[31:16]	W	KEY : High bit writes 0x5A69 to configure this register to be valid, other values are invalid.
[15:7]	-	Reserved
[6]	R	<p>HSECRDY: External 4M ~ 24M Crystal stabilization flag</p> <p>0: The external high-speed crystal oscillator clock is not stable and cannot be used by the internal circuit.</p> <p>1: The external high speed crystal oscillator clock has been stabilized and can be used by the internal circuit.</p>
[5:4]	RW	<p>HSESTARTUP [1:0] :Selection of stability time of external 4M ~24M crystal oscillator</p> <p>00:1024 cycles</p> <p>01:2048 cycles</p> <p>10:4096 cycles</p> <p>11:16384 cycles</p> <p>If the high-speed crystal oscillator clock is used, the stability time must be set to 11. Otherwise, the stability time is insufficient, and the system will be unstable when the system clock is switched over or when the high-speed crystal oscillator clock is in deep sleep.</p>
[3]	-	Reserved

[2:0]	RW	HSEDRV[2:0]: External 4M 24M crystal vibration drive selection 000: minimum drive 111: Maximum drive (recommended)
-------	----	---

Note : The `RCC_UNLOCK` register can be written only after its protection is released

6.4.12 Internal low speed RC oscillator control register(RCC_LSICR)

Register	Address offset	Access	Reset value	Description
RCC_LSICR	0x30	RW	0x0000 007F	Internal low speed RC oscillator control register

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

Internal low speed RC oscillator control register(RCC_LSICR)

Bit	Access	Description
[31:16]	W	KEY : High bit writes 0x5A69 to configure this register to be valid, other values are invalid.
[15:13]	-	Reserved
[12]	R	LSIRDY: Internal low speed clock stability flag 0: The internal low speed is not stable and cannot be used by the internal circuit. 1: The internal low speed has been stabilized and can be used by the internal circuit.
[11 : 10]	RW	LSISTARTUP[1:0] : Internal low speed clock stable time selection 00:4 cycles 01:16 Cycles 10:64 cycles 11:256 cycles
[9]	-	Reserved
[8:0]	RW	LSITRIM[8:0]: Internal low speed clock frequency adjustment. Before delivery, the frequency adjustment value is stored in the Flash. You need to write the Flash value to <code>RCL_CR.TRIM</code> to configure the 38.4KHz/32.768KHz internal low speed clock. 32.768KHz Calibration address: 0x1800_00B0 38.4KHz Calibration address: 0x1800_00B4

Note : 1. Reset control other than LVD and M0+ software reset

2. The `RCC_UNLOCK` register can be written only after its protection is released.

6.4.13 External low speed crystal oscillator control register(RCC_LSECR)

Register	Address offset	Access	Reset value	Description
RCC_LSECR	0x34	RW	0x0000 042F	External low speed crystal oscillator control register

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

External low speed crystal oscillator control register(RCC_LSECR)

Bit	Access	Description
[31:16]	W	KEY : High bit writes 0x5A69 to configure this register to be valid, other values are invalid.
[15:12]	-	Reserved
[11]	RW	LSEPORT : X32K_IN / X32K_OUT function selection 0: GPIO function (digital function) 1: X32K terminal mode (analog function) Note: This bit is protected by RCC_UNLOCK
[10]	RW	LSEAON :LSE enables only disable control 0: LSE_EN Allows or disables control 1: LSE_EN Enables or disables only the control
[9]	RW	LSEBYP : Set and cleared by the software, this bit is written only when the external 32KHz oscillator is off. 0: The LSE oscillator is not bypassed 1: The LSE oscillator is bypassed Note: The enable bit LSE_EN for low speed crystal oscillation is required when using external low speed oscillation
[8]	RW	LSEEN : External 32K crystal oscillator LSE enable signal 0: disable 1: enables the function If this clock is selected for the system clock, it cannot be disabled
[7]	-	Reserved
[6]	R	LSERDY : External 32K crystal oscillator stability marker 0: The external 32K crystal oscillator clock is not stable and cannot be used by the internal circuit. 1: The external 32K crystal oscillator clock has been stabilized and can be used by the internal circuit.
[5:4]	RW	LSESTARTUP[1:0] : External 32.768KHz crystal vibration stability time selection 00:1024 cycles 01:2048 cycles 10:4096 cycles 11:16384 cycles If the low-speed crystal oscillator clock is used, the stability time must be set to 11. Otherwise, the stability time is insufficient, and the system may be unstable when the system clock is switched over or when the low-speed crystal oscillator clock is deep sleep

[3:0]	RW	LSEDRV[3:0]: Select an external low-speed clock driver 0000: minimum drive 1111: maximum drive
-------	----	---

- Note :** 1. This register is in the RTC domain. Only POR can reset this register
2. The RCC_UNLOCK register can be written only after its protection is released.

6.4.14 M0+ IRQ delay control register(RCC_IRQLATENCY)

Register	Address offset	Access	Reset value	Description
RCC_IRQLATENCY	0x38	RW	0x0000 0000	M0+ IRQ delay control register

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

M0+ IRQ delay control register(RCC_IRQLATENCY)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	IRQLATENCY[7:0]:

- Note :** The RCC_UNLOCK register can be written only after its protection is released.

6.4.15 SysTick Timer Register(RCC_STICKCR)

Register	Address offset	Access	Reset value	Description
RCC_STICKCR	0x3C	RW	0x0100 9C3F	SysTick Timer Register

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

SysTick Timer Register(RCC_STICKCR)

Bit	Access	Description
[31:26]	-	Reserved

[25]	RW	NOREF: SysTick Timer Specifies whether to use an external reference clock 0: HCLK/4 1: Using the Core clock (HCLK) Note : 1. If either of the SYST_CSR.CLKSOURCE registers is set to 1, use the Core clock (HCLK). 2. If the SysTick clock is HCLK/4, the reference clock frequency cannot be higher than the system clock frequency
[24]	RW	SKEW: 10ms Check whether the STCALIB value is accurate 0: accurate 1: Not accurate
[23:0]	RW	STCALIB[23:0]: SysTick 10ms calibration value, which is the 10ms calibration value for the external reference clock HCLK/4(4MHz).

6.4.16 SWDIO port control register(RCC_SWDIOCR)

Register	Address offset	Access	Reset value	Description
RCC_SWDIOCR	0x40	RW	0x0000 0001	SWDIO port control register

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

SysTick Timer Register(RCC_STICKCR)

Bit	Access	Description
[31:16]	W	This register is valid only when configured with high level write 0x5A69. Other values are invalid
[15:1]	-	Reserved
[0]	RW	SWDPORT: Configure the terminal function mode for PC7 and PD1 0: peripheral module function mode 1: SWD terminal function

Note : The write bit is protected by RCC_UNLOCK

6.4.17 Peripheral module resets the control register(RCC_PERIRST)

Register	Address offset	Access	Reset value	Description
RCC_PERIRST	0x44	RW	0x0000 0000	Peripheral module resets the control register

31	30	29	28	27	26	25	24
Reserved			CRCRST	GPIODRST	GPIOCRST	GPIOBRST	GPIOARST
23	22	21	20	19	18	17	16
Reserved			DBGRST	BEEPRST	LVDVCRST	Reserved	CLKTRIMRST
15	14	13	12	11	10	9	8
Reserved	AWKRST	ADCRST	WWDGRST	TIM2RST	TIM1RST	OWRIERST	PCARST
7	6	5	4	3	2	1	0
SYSCONRST	BASETIMRST	LPTIMRST	SPIRST	LPUARTRST	I2CRST	UART2RST	UART1RST

Peripheral module resets the control register(RCC_PERIRST)

Bit	Access	Description
[31:29]	-	Reserved
[28]	RW	CRCRST: CRC module reset 0: normal 1: Reset
[27]	RW	GPIODRST: GPIOD module reset 0: normal 1: Reset
[26]	RW	GPIOCRST: GPIOC module reset 0: normal 1: Reset
[25]	RW	GPIOBRST: GPIOB module reset 0: normal 1: Reset
[24]	RW	GPIOARST: GPIOA module reset 0: normal 1: Reset
[23:21]	-	Reserved
[20]	RW	DBGRST: MCU DEBUG module reset 0: normal 1: Reset
[19]	RW	BEEPRST: BEEP module reset 0: normal 1: Reset
[18]	RW	LVDVCRST: LVD module reset 0: normal 1: Reset
[17]	-	Reserved
[16]	RW	CLKTRIMRST: Clock TRIM module reset 0: normal 1: Reset
[15]	-	Reserved
[14]	RW	AWKRST: AWK module reset 0: normal 1: Reset
[13]	RW	ADCRST: ADC module reset 0: normal 1: Reset

[12]	RW	WWDGRST: WWDG module reset 0: normal 1: Reset
[11]	RW	TIM2RST: TIM2 module reset 0: normal 1: Reset
[10]	RW	TIM1RST: TIM1 module reset 0: normal 1: Reset
[9]	RW	OWIREST: 1-Wire module reset 0: normal 1: Reset
[8]	RW	PCARST: PCA module reset 0: normal 1: Reset
[7]	RW	SYSCONRST: SYSCON module reset 0: normal 1: Reset
[6]	RW	BASETIMRST: Base Timer10/11 reset 0: normal 1: Reset
[5]	RW	LPTIMRST: Low Power Timer0/1 reset 0: normal 1: Reset
[4]	RW	SPI: SPI module reset 0: normal 1: Reset
[3]	RW	LPUARTRST: LPUART module reset 0: normal 1: Reset
[2]	RW	I2CRST: I2C module reset 0: normal 1: Reset
[1]	RW	UART2RST: UART2 module reset 0: normal 1: Reset
[0]	RW	UART1RST: UART1 module reset 0: normal 1: Reset

Note : The `RCC_UNLOCK` register can be written only after its protection is released.

6.4.18 RTC Reset control register(RCC_RTCRST)

Register	Address offset	Access	Reset value	Description
RCC_RTCRST	0x48	RW	0x0000 0000	RTC Reset control register

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

RTC Reset control register(RCC_RTCRST)

Bit	Access	Description
[31:16]	W	This register is valid only when configured with high level write 0x5A69. Other values are invalid
[15:1]	-	Reserved, Hold reset value
[0]	RW	RTCST: The RTC module causes a reset Set 1 or clear 0 by software 0: Reset inactive 1: Reset the RTC

Note : The `RCC_UNLOCK` register can be written only after its protection is released

6.4.19 Register Write protection control register(RCC_UNLOCK)

Register	Address offset	Access	Reset value	Description
RCC_UNLOCK	0x60	RW	0x0000 0000	Register Write protection control register

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

Register Write protection control register(RCC_UNLOCK)

Bit	Access	Description
[31:1]	W	Key[31:1]: This register is valid only when configured with high level 0x2AD5334C. Other values are invalid.
[0]	RW	UNLOCK: 0: register write protection is enabled 1: Register write protection is disabled

Note : Write 0x55AA6699 to remove protection

6.4.20 Internal high speed RC oscillator control register 2(RCC_HSITC)**Note :** Only MG32L003 F8

Register	Address offset	Access	Reset value	Description
RCC_HSITC	0x394	RW	0x0000 0009	(Only MG32L003 F8) Internal high speed RC oscillator control register 2

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

Internal high speed RC oscillator control register 2(RCC_HSITC)

Bit	Access	Description
[31:16]	W	Key: This register is valid only when configured with high level 0x5A69. Other values are invalid.
[15:4]	-	Reserved
[3:0]	RW	<p>HSITC[3:0]: Master frequency clock frequency calibration</p> <p>Before delivery, the frequency calibration value is stored in Flash. Users need to write the Flash value in RCC_HSITC to configure the master frequency clock.</p> <ul style="list-style-type: none"> • 24M Calibration address: 0x1800_0090 • 22.12M Calibration address: 0x1800_0092 • 16M Calibration address: 0x1800_0094 • 8M Calibration address: 0x1800_0096 • 4M Calibration address: 0x1800_0098

Note :

1: This register is not controlled by M0+ software reset

2: The RCC_UNLOCK register can be written only after its protection is released

6.4.21 Internal high speed RC oscillator control register 2(RCC_HSITC)**Note :** Only MG32L003 K8

Register	Address offset	Access	Reset value	Description
RCC_HSITC	0x394	RW	0x0000 00FF	(Only MG32L003 K8) Internal high speed RC oscillator control register 2

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

Internal high speed RC oscillator control register 2(RCC_HSITC)

Bit	Access	Description
[31:16]	W	Key: This register is valid only when configured with high level 0x5A69. Other values are invalid.
[15:8]	-	Reserved
[7:0]	RW	<p>HSITC[7:0]: Master frequency clock frequency calibration</p> <p>Before delivery, the frequency calibration value is stored in Flash. Users need to write the Flash value in RCC_HSITC to configure the master frequency clock.</p> <ul style="list-style-type: none"> • 24M Calibration address: 0x1800_0090 • 22.12M Calibration address: 0x1800_0092 • 16M Calibration address: 0x1800_0094 • 8M Calibration address: 0x1800_0096 • 4M Calibration address: 0x1800_0098

Note :

1: This register is not controlled by M0+ software reset

2: The RCC_UNLOCK register can be written only after its protection is released

6.4.22 Internal Low speed RC oscillator control register 2(RCC_LSITRIM)

Note : Only MG32L003 K8

Register	Address offset	Access	Reset value	Description
RCC_LSITRIM	0x39c	RW	0x0000 00ff	(Only MG32L003 K8) Internal low speed RC oscillator control register 2

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

Internal low speed RC oscillator control register 2(RCC_LSITRIM)

Bit	Access	Description
[31:16]	W	Key: This register is valid only when configured with high level 0x5A69. Other values are invalid.

[15:8]	-	Reserved
[7:0]	RW	LSITRIM[7:0]: Internal low speed clock frequency adjustment Before delivery, the frequency calibration value is stored in Flash. Users need to write the Flash value in LSITRIM to configure the 38.4KHz/32.768KHz internal low clock. <ul style="list-style-type: none">• 32.768KHz Calibration address: 0x1800_00B8• 38.4KHz Calibration address: 0x1800_00BC

Note :

1: This register is not controlled by MO+ software reset

2: The RCC_UNLOCK register can be written only after its protection is released

7 System Control(SYSCON)

MG32L003 has a set of system configuration registers. The main purposes of the system configuration controller are as follows:

- Configure the interrupt generation mode of GPIO terminal
- Remap the input trigger sources of TIM10/11,PCA,TIM1 and TIM2
- CS input remapping setting in SPI slave mode
- System-level Deep Sleep debugging and Lockup reset control settings

7.1 SYS Registers

Tab 7.1-1 SYS register map

Offset	Register	Reset value	Description
SYS base address: 0x4000 1C00			
0x00	SYSCON_CFGR0	0x0000 0000	System configuration register 0
0x04	SYSCON_PORTINTCR	0x0000 0000	Terminal Deep Sleep Interrupt Mode Control Register
0x08	SYSCON_PORTCR	0x0000 0000	Terminal control register
0x0C	SYSCON_PCACR	0x0000 0000	PCA Capture channel control register
0x10	SYSCON_TIM1CR	0x0000 0000	TIM1 channel input source selection
0x14	SYSCON_TIM2CR	0x0000 0000	TIM2 channel input source selection
0x50	SYSCON_UNLOCK	0x0000 0000	Syscon register write protection

7.2 Register Description

7.2.1 System configuration register 0(SYSCON_CFGR0)

Register	Address offset	Access	Reset value	Description
SYS_CFGR0	0x00	RW	0x0000 0000	System configuration register 0

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

System configuration register 0(SYSCON_CFGR0)

Bit	Access	Description
[31:16]	W	Key: High bit writes 0x5A69 to configure this register to be valid, other values are invalid.
[15:2]	-	Reserved
[1]	RW	DBGDLSP_DIS: Debugmode, enterDeep Sleepmode disable control bit 0: Allow enteringDeep Sleep inDebugmode 1: Do not allow enteringDeep Sleep inDebugmode

[0]	RW	LOCKUPEN: M0+ LookUpFunction Enable 0: Off 1: Enable Note: MCU will reset when M0+ read wrong command.
-----	----	---

Note: This register can only be written after the SYSICON_UNLOCK register has been unprotected.

7.2.2 Terminal Deep Sleep Interrupt Mode Control Register(SYSICON_PORTINTCR)

Register	Address offset	Access	Reset value	Description
SYS_PORTINTCR	0x04	RW	0x0000 0000	Terminal Deep Sleep Interrupt Mode Control Register

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

Terminal Deep Sleep Interrupt Mode Control Register(SYSICON_CFGR0)

Bit	Access	Description
[31:16]	W	Key: High bit writes 0x5A69 to configure this register to be valid, other values are invalid.
[15:2]	-	Reserved
[1]	RW	PADDLSPCON: 0: After entering Deep Sleep, the interrupt generation mode of PAD is automatically switched to Deep Sleep Interrupt Generation Mode(None Debounce Function) 1: After entering Deep Sleep, the interrupt generation mode of PAD will not switch automatically, and the interrupt generation mode is determined by the SYSICON_PORTINTCR.PADINTSEL bit.
[0]	RW	PADINTSEL: Port Interrupt Mode Selection 0: ACTIVE/Sleep Interrupt generation mode 1: Deep Sleep Interrupt generation mode

Note:

- while Deep Sleep interrupt mode is generated, Debounce function of the GPIO terminal is disabled, resulting in poor anti-interference capability. Therefore, you are advised to use the Deep Sleep mode only when you need to wake up deep sleep mode with terminal interrupt.
- This register can only be written after the SYSICON_UNLOCK register has been unprotected.

7.2.3 Terminal control register(SYSICON_PORTCR)

Register	Address offset	Access	Reset value	Description
SYS_PORTCR	0x08	RW	0x0000 0000	Terminal control register

MG32L003 F8

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						LPTIM_GATE_SEL[1:0]	
7	6	5	4	3	2	1	0
TIM11_GATE_SEL[1:0]		TIM10_GATE_SEL[1:0]		SPINCS_SEL[3:0]			

Terminal control register(SYSICON_PORTCR)

Bit	Access	Description																																
[31:10]	-	Reserved																																
[9:8]	RW	LPTIM_GATE_SEL[1:0]: Low Power Timer Gated input signal source selection 00 : LPTIM_GATE 01 : UART1_RXD 10 : UART2_RXD 11 : LPUART_RXD																																
[7:6]	RW	TIM11_GATE_SEL[1:0] Timer11 Gated input signal source selection 00 : TIM11_GATE 01 : UART1_RXD 10 : UART2_RXD 11 : LPUART_RXD																																
[5:4]	RW	TIM10_GATE_SEL[1:0]: Timer10 Gated input signal source selection 00 : TIM10_GATE 01 : UART1_RXD 10 : UART2_RXD 11 : LPUART_RXD																																
[3:0]	RW	SPINCS_SEL[3:0]: SPI Slave mode NCS signal source selection <table border="1"> <tr> <td>0000</td><td>Fixed high voltage levels</td></tr> <tr> <td>0001</td><td>PA1</td></tr> <tr> <td>0010</td><td>PA2</td></tr> <tr> <td>0011</td><td>PA3</td></tr> <tr> <td>0100</td><td>PB4</td></tr> <tr> <td>0101</td><td>PB5</td></tr> <tr> <td>0110</td><td>PC3</td></tr> <tr> <td>0111</td><td>PC4</td></tr> <tr> <td>1000</td><td>PC5</td></tr> <tr> <td>1001</td><td>PC6</td></tr> <tr> <td>1010</td><td>PC7</td></tr> <tr> <td>1011</td><td>PD1</td></tr> <tr> <td>1100</td><td>PD2</td></tr> <tr> <td>1101</td><td>PD3</td></tr> <tr> <td>1110</td><td>PD4</td></tr> <tr> <td>1111</td><td>PD6</td></tr> </table>	0000	Fixed high voltage levels	0001	PA1	0010	PA2	0011	PA3	0100	PB4	0101	PB5	0110	PC3	0111	PC4	1000	PC5	1001	PC6	1010	PC7	1011	PD1	1100	PD2	1101	PD3	1110	PD4	1111	PD6
0000	Fixed high voltage levels																																	
0001	PA1																																	
0010	PA2																																	
0011	PA3																																	
0100	PB4																																	
0101	PB5																																	
0110	PC3																																	
0111	PC4																																	
1000	PC5																																	
1001	PC6																																	
1010	PC7																																	
1011	PD1																																	
1100	PD2																																	
1101	PD3																																	
1110	PD4																																	
1111	PD6																																	

MG32L003 K8

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					SPINCS_SEL[4]	LPTIM_GATE_SEL[1:0]	
7	6	5	4	3	2	1	0
TIM11_GATE_SEL[1:0]		TIM10_GATE_SEL[1:0]		SPINCS_SEL[3:0]			

Terminal control register(SYSICON_PORTCR)

Bit	Access	Description
[31:11]	-	Reserved
[10]	RW	SPINCS_SEL[4]
[9:8]	RW	LPTIM_GATE_SEL[1:0]: Low Power Timer Gated input signal source selection 00 : LPTIM_GATE 01 : UART1_RXD 10 : UART2_RXD 11 : LPUART_RXD
[7:6]	RW	TIM11_GATE_SEL[1:0] Timer11 Gated input signal source selection 00 : TIM11_GATE 01 : UART1_RXD 10 : UART2_RXD 11 : LPUART_RXD
[5:4]	RW	TIM10_GATE_SEL[1:0]: Timer10 Gated input signal source selection 00 : TIM10_GATE 01 : UART1_RXD 10 : UART2_RXD 11 : LPUART_RXD

[3:0]	RW	SPI Slave mode NCS signal source selection,SPINCS_SEL[4:0]:	
		00000	Fixed high voltage levels
		00001	PA1
		00010	PA2
		00011	PA3
		00100	PB4
		00101	PB5
		00110	PC3
		00111	PC4
		01000	PC5
		01001	PC6
		01010	PC7
		01011	PD1
		01100	PD2
		01101	PD3
		01110	PD4
		01111	PD6
		10001	PA4
		10010	PB0
		10011	PB1
		10100	PB2
		10101	PB3
		10110	PB6
		10111	PB7
11000	PC0		
11001	PC1		
11010	PC2		
11011	PD0		
11100	PD7		

7.2.4 PCA Capture channel control register(SYSICON_PCACR)

Register	Address offset	Access	Reset value	Description
SYS_PCACR	0x0C	RW	0x0000 0000	PCA Capture channel control register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						PCA_CAP4_SEL[1:0]	
7	6	5	4	3	2	1	0
PCA_CAP3_SEL[1:0]		PCA_CAP2_SEL[1:0]		PCA_CAP1_SEL[1:0]		PCA_CAP0_SEL[1:0]	

PCA Capture channel control register(SYSICON_PCACR)

Bit	Access	Description
[31:10]	-	Reserved

[9:8]	RW	PCA_CAP4_SEL[1:0]: PCA capture channel 4 signal source selection 00 : PCA_CH4 01 : UART1_RXD 10 : UART2_RXD 11 : LPUART_RXD
[7:6]	RW	PCA_CAP3_SEL[1:0]: PCA capture channel 3 signal source selection 00 : PCA_CH3 01 : UART1_RXD 10 : UART2_RXD 11 : LPUART_RXD
[5:4]	RW	PCA_CAP2_SEL[1:0]: PCA capture channel 2 signal source selection 00 : PCA_CH2 01 : UART1_RXD 10 : UART2_RXD 11 : LPUART_RXD
[3:2]	RW	PCA_CAP1_SEL[1:0]: PCA capture channel 1 signal source selection 00 : PCA_CH1 01 : UART1_RXD 10 : UART2_RXD 11 : LPUART_RXD
[1:0]	RW	PCA_CAP0_SEL[1:0]: PCA capture channel 0 signal source selection 00 : PCA_CH0 01 : UART1_RXD 10 : UART2_RXD 11 : LPUART_RXD

7.2.5 TIM1 channel input source selection(SYSICON_TIM1CR)

Register	Address offset	Access	Reset value	Description
SYSICON_TIM1CR	0x10	RW	0x0000 0000	TIM1 channel input source selection

MG32L003 F8

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	CLKFAILBRKEN	DSLPRBRKEN	TIM1BRKOUTCFG	TIM1ETR_SEL[3:0]			
15	14	13	12	11	10	9	8
Reserved	TIM1CH4IN_SEL[2:0]			Reserved	TIM1CH3IN_SEL[2:0]		
7	6	5	4	3	2	1	0
Reserved	TIM1CH2IN_SEL[2:0]			Reserved	TIM1CH1IN_SEL[2:0]		

TIM1 channel input source selection(SYSICON_TIM1CR)

Bit	Access	Description
[31:23]	-	Reserved
[22]	RW	CLKFAILBRKEN: The system clock stops checking the TIM1 Braek enable 0: invalid 1: enable

[21]	RW	DSLPRKEN: The Deep Sleep mode TIM1 Break was enabled 0: invalid 1: enable																																
[20]	RW	TIM1BRKOUTCFG: 0: ocxp/ocxnp output in break mode is controlled by TIM1 1: ocxp/ocxnp outputs 0 at the same time in break mode																																
[19:16]	RW	TIM1ETR_SEL[3:0]: <table border="1" style="margin-left: 20px;"> <tr> <td>0000</td> <td>Fixed low voltage levels</td> </tr> <tr> <td>0001</td> <td>PA1</td> </tr> <tr> <td>0010</td> <td>PA2</td> </tr> <tr> <td>0011</td> <td>PA3</td> </tr> <tr> <td>0100</td> <td>PB4</td> </tr> <tr> <td>0101</td> <td>PB5</td> </tr> <tr> <td>0110</td> <td>PC3</td> </tr> <tr> <td>0111</td> <td>PC4</td> </tr> <tr> <td>1000</td> <td>PC5</td> </tr> <tr> <td>1001</td> <td>PC6</td> </tr> <tr> <td>1010</td> <td>PC7</td> </tr> <tr> <td>1011</td> <td>PD1</td> </tr> <tr> <td>1100</td> <td>PD2</td> </tr> <tr> <td>1101</td> <td>PD3</td> </tr> <tr> <td>1110</td> <td>PD4</td> </tr> <tr> <td>1111</td> <td>PD6</td> </tr> </table>	0000	Fixed low voltage levels	0001	PA1	0010	PA2	0011	PA3	0100	PB4	0101	PB5	0110	PC3	0111	PC4	1000	PC5	1001	PC6	1010	PC7	1011	PD1	1100	PD2	1101	PD3	1110	PD4	1111	PD6
0000	Fixed low voltage levels																																	
0001	PA1																																	
0010	PA2																																	
0011	PA3																																	
0100	PB4																																	
0101	PB5																																	
0110	PC3																																	
0111	PC4																																	
1000	PC5																																	
1001	PC6																																	
1010	PC7																																	
1011	PD1																																	
1100	PD2																																	
1101	PD3																																	
1110	PD4																																	
1111	PD6																																	
[15]	-	Reserved																																
[14:12]	RW	TIM1CH4IN_SEL[2:0]: TIM1 CH4 Input channel signal source Selection 000: TIM1_CH4 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved																																
[11]	-	Reserved																																
[10:8]	RW	TIM1CH3IN_SEL[2:0]: TIM1 CH3 Input channel signal source Selection 000: TIM1_CH3 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved																																
[7]	-	Reserved																																
[6:4]	RW	TIM1CH2IN_SEL[2:0]: TIM1 CH2 Input channel signal source Selection 000: TIM1_CH2 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved																																

[3]	-	Reserved
[2:0]	RW	TIM1CH1IN_SEL[2:0]: TIM1 CH1 Input channel signal source Selection 000: TIM_CH1 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved

MG32L003 K8

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TIM1ETR_SEL[4]	CLKFAILBRKEN	DSLPRKEN	TIM1BRKOUTCFG	TIM1ETR_SEL[3:0]			
15	14	13	12	11	10	9	8
Reserved	TIM1CH4IN_SEL[2:0]			Reserved	TIM1CH3IN_SEL[2:0]		
7	6	5	4	3	2	1	0
Reserved	TIM1CH2IN_SEL[2:0]			Reserved	TIM1CH1IN_SEL[2:0]		

TIM1 channel input source selection(SYSICON_TIM1CR)

Bit	Access	Description
[31:22]	-	Reserved
[23]	RW	TIM1ETR_SEL[4]
[22]	RW	CLKFAILBRKEN: The system clock stops checking the TIM1 Braek enable 0: invalid 1: enable
[21]	RW	DSLPRKEN: The Deep Sleep mode TIM1 Break was enabled 0: invalid 1: enable
[20]	RW	TIM1BRKOUTCFG: 0: ocxp/ocxnp output in break mode is controlled by TIM1 1: ocxp/ocxnp outputs 0 at the same time in break mode

[19:16]	RW	Signal source selection, TIM1ETR_SEL[4:0]:	
		00000	Fixed low voltage levels
		00001	PA1
		00010	PA2
		00011	PA3
		00100	PB4
		00101	PB5
		00110	PC3
		00111	PC4
		01000	PC5
		01001	PC6
		01010	PC7
		01011	PD1
		01100	PD2
		01101	PD3
		01110	PD4
		01111	PD6
		10000	Fixed low voltage levels
		10001	PA4
		10010	PB0
		10011	PB1
		10100	PB2
		10101	PB3
		10110	PB6
10111	PB7		
11000	PC0		
11001	PC1		
11010	PC2		
11011	PD0		
11100	PD7		
[15]	-	Reserved	
[14:12]	RW	TIM1CH4IN_SEL[2:0]: TIM1 CH4 Input channel signal source Selection 000: TIM1_CH4 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved	
[11]	-	Reserved	
[10:8]	RW	TIM1CH3IN_SEL[2:0]: TIM1 CH3 Input channel signal source Selection 000: TIM1_CH3 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved	
[7]	-	Reserved	

[6:4]	RW	TIM1CH2IN_SEL[2:0]: TIM1 CH2 Input channel signal source Selection 000: TIM1_CH2 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved
[3]	-	Reserved
[2:0]	RW	TIM1CH1IN_SEL[2:0]: TIM1 CH1 Input channel signal source Selection 000: TIM_CH1 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved

7.2.6 TIM2 channel input source selection(SYSICON_TIM2CR)

Register	Address offset	Access	Reset value	Description
SYSICON_TIM2CR	0x14	RW	0x0000 0000	TIM2 channel input source selection

MG32L003 F8

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				TIM2ETR_SEL[3:0]			
15	14	13	12	11	10	9	8
Reserved	TIM2CH4IN_SEL[2:0]			Reserved	TIM2CH3IN_SEL[2:0]		
7	6	5	4	3	2	1	0
Reserved	TIM2CH2IN_SEL[2:0]			Reserved	TIM2CH1IN_SEL[2:0]		

TIM2 channel input source selection(SYSICON_TIM2CR)

Bit	Access	Description
[31:20]	-	Reserved

[19:16]	RW	TIM2ETR_SEL[3:0]: TIM2 ETR signal source selection	
		0000	Fixed low voltage levels
		0001	PA1
		0010	PA2
		0011	PA3
		0100	PB4
		0101	PB5
		0110	PC3
		0111	PC4
		1000	PC5
		1001	PC6
		1010	PC7
		1011	PD1
		1100	PD2
[15]	-	Reserved	
[14:12]	RW	TIM2CH4IN_SEL[2:0]: TIM2 CH4 Input channel signal source Selection 000: TIM2_CH4 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved	
[11]	-	Reserved	
[10:8]	RW	TIM2CH3IN_SEL[2:0]: TIM2 CH3 Input channel signal source Selection 000: TIM2_CH3 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved	
[7]	-	Reserved	
[6:4]	RW	TIM2CH2IN_SEL[2:0]: TIM2 CH2 Input channel signal source Selection 000: TIM2_CH2 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved	
[3]	-	Reserved	

[2:0]	RW	TIM2CH1IN_SEL[2:0]: TIM2 CH1 Input channel signal source Selection 000: TIM_CH1 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved
-------	----	---

MG32L003 K8

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TIM2ETR_SEL[4]	Reserved			TIM2ETR_SEL[3:0]			
15	14	13	12	11	10	9	8
Reserved	TIM2CH4IN_SEL[2:0]			Reserved	TIM2CH3IN_SEL[2:0]		
7	6	5	4	3	2	1	0
Reserved	TIM2CH2IN_SEL[2:0]			Reserved	TIM2CH1IN_SEL[2:0]		

TIM2 channel input source selection(SYSICON_TIM2CR)

Bit	Access	Description
[31:24]	-	Reserved
[23]	RW	TIM2ETR_SEL[4]
[22:20]	-	Reserved

[19:16]	RW	AATIM2 ETR signal source selection,TIM2ETR_SEL[4:0] :	
		00000	Fixed low voltage levels
		00001	PA1
		00010	PA2
		00011	PA3
		00100	PB4
		00101	PB5
		00110	PC3
		00111	PC4
		01000	PC5
		01001	PC6
		01010	PC7
		01011	PD1
		01100	PD2
		01101	PD3
		01110	PD4
		01111	PD6
		10000	Fixed low voltage levels
		10001	PA4
		10010	PB0
		10011	PB1
		10100	PB2
		10101	PB3
		10110	PB6
10111	PB7		
11000	PC0		
11001	PC1		
11010	PC2		
11011	PD0		
11100	PD7		
[15]	-	Reserved	
[14:12]	RW	TIM2CH4IN_SEL[2:0]: TIM2 CH4 Input channel signal source Selection 000: TIM2_CH4 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved	
[11]	-	Reserved	
[10:8]	RW	TIM2CH3IN_SEL[2:0]: TIM2 CH3 Input channel signal source Selection 000: TIM2_CH3 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved	
[7]	-	Reserved	

[6:4]	RW	TIM2CH2IN_SEL[2:0]: TIM2 CH2 Input channel signal source Selection 000: TIM2_CH2 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved
[3]	-	Reserved
[2:0]	RW	TIM2CH1IN_SEL[2:0]: TIM2 CH1 Input channel signal source Selection 000: TIM_CH1 001: UART1_RXD 010: UART2_RXD 011: LPUART_RXD 100: LSI Other: reserved

7.2.7 Syscon register write protection(SYSCON_UNLOCK)

Register	Address offset	Access	Reset value	Description
SYSCON_UNLOCK	0x50	RW	0x0000 0000	Syscon register write protection

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

Syscon register write protection(SYSCON_UNLOCK)

Bit	Access	Description
[31:1]	W	Key : This register is valid only when configured with high level 0x2AD5334C. Other values are invalid
[0]	RW	UNLOCK: 0: The protection is effective 1: The protection is invalid

Note : Write 0x55AA6699 to disable protection.

8 Interrupt Controller(NVIC)

8.1 Introduction

The M0+ processor has a built-in nested vector interrupt controller (NVIC), supporting up to 32 interrupt request (IRQ) inputs and 1 non-maskable interrupt (NMI) input. In addition, the processor also supports several internal exceptions.

Each exception source has a separate exception number, and each exception type has a corresponding priority. Some exceptions have a fixed priority, while others are programmable.

8.2 Features

- 32 maskable interrupt channels (excluding 16 interrupt lines for M0+)
- 4 programmable priority levels (2-bit interrupt priority level is used)
- Low-latency exception and interrupt handling
- Power Management Control
- Implementation of System Control Register
- Support nesting and vector interrupt
- Dynamic change of priority

8.3 Interrupt priority

The software can set 4 levels of priority for external interrupts. The highest priority is "0", the lowest priority is "3", and the default value for all user-configurable priorities is "0".

If the processor is running an interrupt processing and the priority of the new interrupt is higher than that of the running one, preemption will occur. Running interrupt processing will be suspended and a new interrupt will be executed instead. This process is usually called interrupt nesting. After the execution of the new interrupt, the previous interrupt processing will continue to execute and return to the program thread after its completion.

If the processor is running interrupt processing with the same or higher priority, the new interrupt will wait and enter the suspended state. Suspended interrupts will wait until the current interrupt level changes. For example, after the current running interrupt processing is completed and returned, the current priority will be reduced to smaller than the suspended interrupt.

If two interrupts occur at the same time and they have the same priority, the interrupt with the smaller interrupt number will be executed first. For example, if interrupt 0 and interrupt 1 are enabled and have the same priority, when they are triggered at the same time, interrupt 0 will be executed first.

8.4 Interrupt vector table

M0+ responds to an interrupt, the processor automatically extracts the start address of the interrupt service routine(ISR) from the interrupt vector table of the memory. The vector table includes the initial value of the reset stack(MSP) after reset and the entry address of all exception handling. The interrupt number indicates the order in which exceptions are handled. Among them, the storage order of the interrupt vector is consistent with the interrupt number. Since each vector is 1 word(4 bytes), the address of the interrupt vector is the interrupt number multiplied by 4, and each interrupt vector is

the starting address of the interrupt processing

Tab 8.4-1 Interrupt vector

Interrupt	Memory address	External interrupt (IRQ#)	Priority	Priority type	Interrupt source
0	0x0000 0000	-	-	-	MSP Initial value
1	0x0000 0004	-	3(highest)	fixed	Reset vector
2	0x0000 0008		2	fixed	NMI vector
3	0x0000 000C		1	fixed	HardFault
4-10	0x0000 000C - 0x0000 002B				Reserved
11	0x0000 002C			programmable	SVC
12-13	0x0000 0030 - 0x0000 0037				Reserved
14	0x0000 0038			programmable	PendSV vector
15	0x0000 003C			programmable	SysTick vector
16	0x0000 0040	0		programmable	GPIO_PA
17	0x0000 0044	1		programmable	GPIO_PB
18	0x0000 0048	2		programmable	GPIO_PC
19	0x0000 004C	3		programmable	GPIO_PD
20	0x0000 0050	4		programmable	Flash
21	0x0000 0054	5		programmable	Reserved
22	0x0000 0058	6		programmable	UART1
23	0x0000 005C	7		programmable	UART2
24	0x0000 0060	8		programmable	LPUART
25	0x0000 0064	9		programmable	Reserved
26	0x0000 0068	10		programmable	SPI
27	0x0000 006C	11		programmable	Reserved
28	0x0000 0070	12		programmable	I2C
29	0x0000 0074	13		programmable	Reserved
30	0x0000 0078	14		programmable	TIM10
31	0x0000 007C	15		programmable	TIM11
32	0x0000 0080	16		programmable	LPTIM
33	0x0000 0084	17		programmable	Reserved
34	0x0000 0088	18		programmable	TIM1
35	0x0000 008C	19		programmable	TIM2
36	0x0000 0090	20		programmable	Reserved
37	0x0000 0094	21		programmable	PCA
38	0x0000 0098	22		programmable	WWDG
39	0x0000 009C	23		programmable	IWDG
40	0x0000 00A0	24		programmable	ADC
41	0x0000 00A4	25		programmable	LVD
42	0x0000 00A8	26		programmable	VCMP
43	0x0000 00AC	27		programmable	Reserved
44	0x0000 00B0	28		programmable	AWK
45	0x0000 00B4	29		programmable	ONEWIRE
46	0x0000 00B8	30		programmable	RTC
47	0x0000 00BC	31		programmable	CLKTRIM

8.5 Interrupt wake-up control WIC

When the processor uses the SLEEP-ON-EXIT feature or executes the WFI instruction to enter sleep, the instruction execution will be stopped, and when an interrupt request(higher priority) occurs and needs to be processed, the processor will be awakened.

WFI behavior	Wake up	ISR execute
PRIMASK 0 : IRQ priority > Current level	Y	Y
PRIMASK 0 : IRQ priority <= Current level	N	N
PRIMASK 1 : IRQ priority > Current level	Y	N
PRIMASK 1 : IRQ priority <= Current level	N	N

8.5.1 NVIC wakes up from deep-sleep mode to interrupt ISR setting

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set SCR.DEEPSLEEP to 1
4. Use WFI command to enter deep sleep mode
5. The system enters the deep sleep mode and waits for the interrupt to wake up, and executes the interrupt service program after waking up

8.5.2 NVIC wakes up from Deep Sleep mode setting and does not execute interrupt ISR

1. Enable NVIC corresponding to the module that needs to wake up the processor
2. Use PRIMASK register to mask interrupt
3. Enable module interrupt enable
4. Set SCR.DEEPSLEEP to 1
5. Enter Deep Sleep mode using the WFI command
6. The system enters the deep sleep mode and waits for the interrupt to wake up, and executes the next instruction after waking up
7. Clear interrupt flag, and clear interrupt pending status

8.5.3 Use the exit sleep feature

The Exit Sleep(SLEEP-ON-EXIT) feature is very suitable for interrupt-driven applications. When this feature is enabled, the processor enters sleep mode as long as the exception processing is completed and returned to the thread mode. With the exit-from-sleep feature, the processor can be in sleep mode as much as possible. M0+Use the(SLEEP-ON-EXIT) feature to enter sleep mode, which is similar to the effect of executing WFI immediately after executing the exception exit. However, in order to avoid the need to push the stack when entering an exception next time, the processor will not perform the stack pushing operation.

1. Enable interrupts for wakeup from deep sleepNVIC
2. Enable module interrupt enable
3. Set SCR.DEEPSLEEP to 1

4. Set SCR.SLEEPONEXIT to 1
5. Enter Deep Sleep mode using the WFI command
6. The system enters the deep sleep mode and waits for the interrupt to wake up, and executes the interrupt service subroutine after waking up
7. Automatically enter sleep mode routine when exiting interrupt service

8.6 Software

8.6.1 External interrupt enable

Each peripheral module has its own interrupt enable register. When an interrupt operation is required, the interrupt enable of the peripheral must be turned on.

8.6.2 NVIC interrupt enable and clear enable

The M0+ processor supports up to 32 interrupt sources, each interrupt source corresponds to an interrupt enable bit and a reset bit. Thus, there are 32-bit interrupt enable set register and 32-bit interrupt enable reset register. If you want to enable an interrupt, set 1 to the corresponding position of the interrupt enable setting register. If you want to clear an interrupt enable, set 1 to the corresponding position of the interrupt enable reset register.

The interrupt enable mentioned here is only for the processor NVIC. Whether each peripheral interrupt is generated or not is determined by the interrupt control register of the peripheral, and has nothing to do with the interrupt enable set/reset register.

8.6.3 NVIC interrupt pending and clear pending

If an interrupt occurs but cannot be handled immediately, the interrupt request will be pending. The pending state is stored in a register and will remain valid as long as the processor's current priority has not been lowered enough to handle the pending request and the pending state has not been cleared manually.

When the processor starts to enter the interrupt processing, the hardware will automatically clear the corresponding interrupt pending status.

Interrupt pending status can be accessed or modified by manipulating the Interrupt Set Pending NVIC_ISPR and Interrupt Clear Pending NVIC_ICPR registers. The Interrupt Pending Status Register allows software to trigger an interrupt.

8.6.4 NVIC Interrupt Priority

Setting the NVIC_IPR0-NVIC_IPR7 register determines the priority of IRQ0-IRQ32. The interrupt priority register should be programmed before the interrupt is enabled, which is usually completed at the beginning of the program. You should avoid changing the interrupt priority after the interrupt is enabled, and the result of this situation is unpredictable and is not supported by the M0+processor.

8.6.5 NVIC interrupt mask

Some time-sensitive applications need to prohibit all interrupts in a short period of time, which can be realized by using the interrupt mask register PRIMASK. PRIMASK is only valid for 1 bit and defaults to 0 after reset. When the register is 0, all interrupts and exceptions are allowed; and when it is set to 1, only NMI (not supported by this system) and hardware error exceptions are enabled. In fact, when PRIMASK is set to 1, the current priority of the processor is reduced to 0.

The PRIMASK register can be programmed in several ways, using assembly language, and using the CPSIEi and CPSIDi instructions to set and clear the PRIMASK register. If using C language and CMSIS device driver library, users can use the following functions to set and clear PRIMASK.

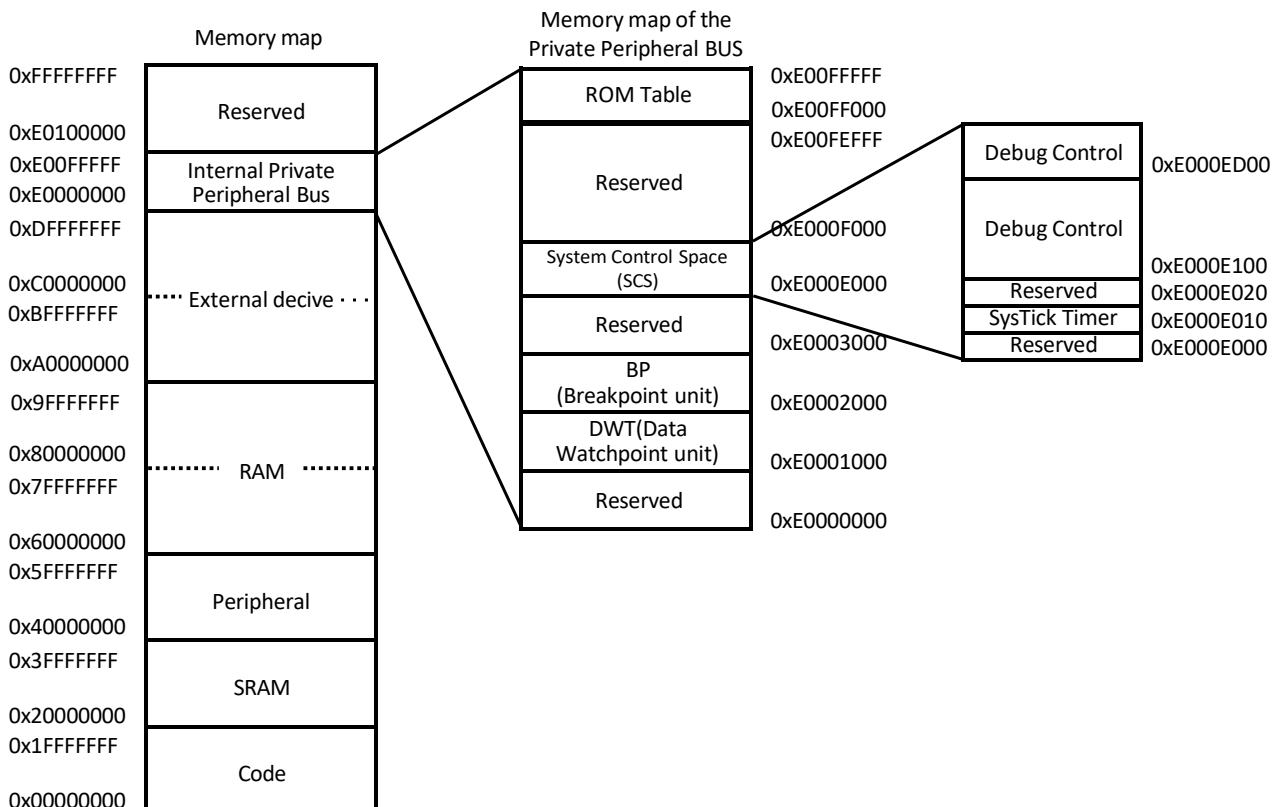
```
__disable_irq(); __enable_irq();
```


8.7 NVIC Registers

Tab 8.7-1 RCC Registers map

Offset	Register	Reset value	Description
NVIC bass address: 0xE000 E000			
0x100	NVIC_ISER	0x0000_0000	AHB Clock Frequency Division Register
0x180	NVIC_ICER	0x0000_0000	APB Clock Frequency Division Register
0x200	NVIC_ISPR	0x0000_0000	AHB Peripheral module clock enable Register
0x280	NVIC_ICPR	0x0000_0000	APB Peripheral module clock Enable register
0x400	NVIC_IPR0	0x0000_0000	Clock output control register
0x404	NVIC_IPR1	0x0000_0000	System reset control register
0x408	NVIC_IPR2	0x0000_0000	System resets status register
0x40C	NVIC_IPR3	0x0000_0000	System Clock Source Configuration
0x410	NVIC_IPR4	0x0000_0000	System Clock Source Selection Register
0x414	NVIC_IPR5	0x0000_0000	Internal High Speed RC Oscillator Control Register
0x418	NVIC_IPR6	0x0000_0000	External high speed crystal oscillator control register
0x41C	NVIC_IPR7	0x0000_0000	Internal low speed RC oscillator control register

The NVIC register is a part of the SCS register. The SCS base address is 0xE000E000, as shown in the following figure:



8.8 Register Description

8.8.1 Interrupt enable set register(NVIC_ISER)

Register	Address offset	Access	Reset value	Description
NVIC_ISER	0x100	RW	0x0000_0000	Interrupt enable set register

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

Interrupt enable set register(NVIC_ISER)

Bit	Access	Description
[31:0]	RW	SETENA: If external interrupt 0 to 31 is enabled, 1 is set and 0 is invalid [0] : IRQ0 [1] : IRQ1 [2] : IRQ2 [31] : IRQ31

8.8.2 Interrupt enables clear register(NVIC_ICER)

Register	Address offset	Access	Reset value	Description
NVIC_ICER	0x180	RW	0x0000_0000	Interrupt enables clear register

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

Interrupt enable set register(NVIC_ICER)

Bit	Access	Description
[31:0]	RW	CLRENA: Clear Enable external interrupt 0 to 31; "1" is cleared, "0" is invalid [0] : IRQ0 [1] : IRQ1 [2] : IRQ2 [31] : IRQ31

8.8.3 Interrupt suspends the setting register(NVIC_ISPR)

Register	Address offset	Access	Reset value	Description
NVIC_ISPR	0x200	RW	0x0000_0000	Interrupt suspends the setting register

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

Interrupt suspends the setting register(NVIC_ISPR)

Bit	Access	Description
[31:0]	RW	SETPEND: Set the pending state of external interrupts from 0 to 31. "1" is set and "0" is invalid [0] : IRQ0 [1] : IRQ1 [2] : IRQ2 [31] : IRQ31

8.8.4 Interrupt suspends the clear register(NVIC_ICPR)

Register	Address offset	Access	Reset value	Description
NVIC_ICPR	0x280	RW	0x0000_0000	Interrupt suspends the clear register

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

Interrupt suspends the setting register(NVIC_ICPR)

Bit	Access	Description
[31:0]	RW	CLRPEND: Clear the pending state of external interrupts from 0 to 31. "1" is cleared and "0" is invalid [0] : IRQ0 [1] : IRQ1 [2] : IRQ2 [31] : IRQ31

8.8.5 Interrupt priority control register 0(NVIC_IPR0)

Register	Address offset	Access	Reset value	Description
NVIC_IPR0	0x400	RW	0x0000_0000	Interrupt priority control register 0

31	30	29	28	27	26	25	24
IPR0[31:24]							
23	22	21	20	19	18	17	16
IPR0[23:16]							
15	14	13	12	11	10	9	8
IPR0[15:8]							
7	6	5	4	3	2	1	0
IPR0[7:0]							

Interrupt priority control register 0(NVIC_IPR0)

Bit	Access	Description
[31:0]	RW	<p>IPR0:External interrupts have a priority of 0 to 3;</p> <p>[31:30] : Indicates the priority of interrupt 3</p> <p>[23:22] : Indicates the priority of interrupt 2</p> <p>[15:14] : The priority of interrupt 1</p> <p>[7:6] : The priority of interrupt 0</p> <p>00 has the highest priority and 11 has the lowest priority</p>

8.8.6 Interrupt priority control register 1(NVIC_IPR1)

Register	Address offset	Access	Reset value	Description
NVIC_IPR1	0x404	RW	0x0000_0000	Interrupt priority control register 1

31	30	29	28	27	26	25	24
IPR1[31:24]							
23	22	21	20	19	18	17	16
IPR1[23:16]							
15	14	13	12	11	10	9	8
IPR1[15:8]							
7	6	5	4	3	2	1	0
IPR1[7:0]							

Interrupt priority control register1(NVIC_IPR1)

Bit	Access	Description
[31:0]	RW	<p>IPR1:External interrupts have a priority of 4 to 7;</p> <p>[31:30] : Indicates the priority of interrupt 7</p> <p>[23:22] : Indicates the priority of interrupt 6</p> <p>[15:14] : The priority of interrupt 5</p> <p>[7:6] : The priority of interrupt 4</p> <p>00 has the highest priority and 11 has the lowest priority</p>

8.8.7 Interrupt priority control register 2(NVIC_IPR2)

Register	Address offset	Access	Reset value	Description
NVIC_IPR2	0x408	RW	0x0000_0000	Interrupt priority control register 2

31	30	29	28	27	26	25	24
IPR2[31:24]							
23	22	21	20	19	18	17	16
IPR2[23:16]							
15	14	13	12	11	10	9	8
IPR2[15:8]							
7	6	5	4	3	2	1	0
IPR2[7:0]							

Interrupt priority control register2(NVIC_IPR2)

Bit	Access	Description
[31:0]	RW	IPR2: External interrupts have a priority of 8 to 11; [31:30] : Indicates the priority of interrupt 11 [23:22] : Indicates the priority of interrupt 10 [15:14] : The priority of interrupt 9 [7:6] : The priority of interrupt 8 00 has the highest priority and 11 has the lowest priority

8.8.8 Interrupt priority control register 3(NVIC_IPR3)

Register	Address offset	Access	Reset value	Description
NVIC_IPR3	0x40C	RW	0x0000_0000	Interrupt priority control register 3

31	30	29	28	27	26	25	24
IPR3[31:24]							
23	22	21	20	19	18	17	16
IPR3[23:16]							
15	14	13	12	11	10	9	8
IPR3[15:8]							
7	6	5	4	3	2	1	0
IPR3[7:0]							

Interrupt priority control register 3(NVIC_IPR3)

Bit	Access	Description
[31:0]	RW	IPR3: External interrupts have a priority of 12 to 15; [31:30] : Indicates the priority of interrupt 15 [23:22] : Indicates the priority of interrupt 14 [15:14] : The priority of interrupt 13 [7:6] : The priority of interrupt 12 00 has the highest priority and 11 has the lowest priority

8.8.9 Interrupt priority control register 4(NVIC_IPR4)

Register	Address offset	Access	Reset value	Description
NVIC_IPR4	0x410	RW	0x0000_0000	Interrupt priority control register 4

31	30	29	28	27	26	25	24
IPR4[31:24]							
23	22	21	20	19	18	17	16
IPR4[23:16]							
15	14	13	12	11	10	9	8
IPR4[15:8]							
7	6	5	4	3	2	1	0
IPR4[7:0]							

Interrupt priority control register 4(NVIC_IPR4)

Bit	Access	Description
[31:0]	RW	<p>IPR4:External interrupts have a priority of 16 to 19;</p> <p>[31:30] : Indicates the priority of interrupt 19</p> <p>[23:22] : Indicates the priority of interrupt 18</p> <p>[15:14] : The priority of interrupt 17</p> <p>[7:6] : The priority of interrupt 16</p> <p>00 has the highest priority and 11 has the lowest priority</p>

8.8.10 Interrupt priority control register 5(NVIC_IPR5)

Register	Address offset	Access	Reset value	Description
NVIC_IPR5	0x414	RW	0x0000_0000	Interrupt priority control register 5

31	30	29	28	27	26	25	24
IPR5[31:24]							
23	22	21	20	19	18	17	16
IPR5[23:16]							
15	14	13	12	11	10	9	8
IPR5[15:8]							
7	6	5	4	3	2	1	0
IPR5[7:0]							

Interrupt priority control register 5(NVIC_IPR5)

Bit	Access	Description
[31:0]	RW	<p>IPR5:External interrupts have a priority of 20 to 23;</p> <p>[31:30] : Indicates the priority of interrupt 23</p> <p>[23:22] : Indicates the priority of interrupt 22</p> <p>[15:14] : The priority of interrupt 21</p> <p>[7:6] : The priority of interrupt 20</p> <p>00 has the highest priority and 11 has the lowest priority</p>

8.8.11 Interrupt priority control register 6(NVIC_IPR6)

Register	Address offset	Access	Reset value	Description
NVIC_IPR6	0x418	RW	0x0000_0000	Interrupt priority control register 6

31	30	29	28	27	26	25	24
IPR6[31:24]							
23	22	21	20	19	18	17	16
IPR6[23:16]							
15	14	13	12	11	10	9	8
IPR6[15:8]							
7	6	5	4	3	2	1	0
IPR6[7:0]							

Interrupt priority control register 6(NVIC_IPR6)

Bit	Access	Description
[31:0]	RW	IPR6: External interrupts have a priority of 24 to 27; [31:30] : Indicates the priority of interrupt 27 [23:22] : Indicates the priority of interrupt 26 [15:14] : The priority of interrupt 25 [7:6] : The priority of interrupt 24 00 has the highest priority and 11 has the lowest priority

8.8.12 Interrupt priority control register 7(NVIC_IPR7)

Register	Address offset	Access	Reset value	Description
NVIC_IPR7	0x41C	RW	0x0000_0000	Interrupt priority control register 7

31	30	29	28	27	26	25	24
IPR7[31:24]							
23	22	21	20	19	18	17	16
IPR7[23:16]							
15	14	13	12	11	10	9	8
IPR7[15:8]							
7	6	5	4	3	2	1	0
IPR7[7:0]							

Interrupt priority control register 7(NVIC_IPR7)

Bit	Access	Description
[31:0]	RW	IPR7: External interrupts have a priority of 28 to 31; [31:30] : Indicates the priority of interrupt 31 [23:22] : Indicates the priority of interrupt 30 [15:14] : The priority of interrupt 29 [7:6] : The priority of interrupt 28 00 has the highest priority and 11 has the lowest priority

9 General purpose input and output(GPIO)

9.1 GPIO Introduction

The general input/output port is used for chip and external data transmission, there are four groups of GPIO:GPIOA,GPIOB,GPIOC and GPIOD. The functions are basically the same. You can map GPIO to the corresponding chip pins through configuration. Each pin can be independently configured as a digital input or output port, or can be configured as an analog input. In addition, it can also be configured as an external interrupt, on-chip peripheral input/output and other alternate functions. A pin can only be mapped to one alternate function at the same time, which is configured through the port alternate function register(GPIOx_AFR).

Each general purpose I/O port has five configuration registers(GPIOx_DIRCR, GPIOx_OTYPER, GPIOx_PUPDR, GPIOx_SLEWCR and GPIOx_DRVCR),two data registers(GPIOx_IDR and GPIOx_ODR),one output set register(GPIOx_ODSET),one output reset register(GPIOx_ODCLR)and one alternate function register(GPIOx_AFR).

Each port can be configured as an internal pull-up(pull up) /pull-down(pull down)input/output, high-impedance input(floating input), push-pull output(push-pull output), open-drain output(open drain output), to enhance the output drive capability. After the chip is reset, the port is reset to a high-impedance input. The purpose is to prevent the abnormal action of external device due to chip's abnormal reset. In order to avoid the leakage caused by high-impedance input, the user should configure the port(configured as internal pull-up input or output) after the chip is started. After the port is configured as an analog port, the digital function is isolated and the number "1" and "0" cannot be output.The result of CPU reading the port is "0".

All ports can provide external interrupts, and each interrupt can be configured as high-level trigger, low-level trigger, rising edge trigger, falling edge trigger or any edge trigger, and supports input debounce in edge trigger mode. Support interrupt generation in working mode/sleep mode/deep sleep mode.

9.2 GPIO main features

- Output status: push-pull output or open-drain output with pull-up or pull-down
- Output data from data register(GPIOx_ODR) or peripheral(alternate function output)
- Configurable speed of each I/O port
- Input status: floating, pull-up/pull-down, analog input
- Slave Data Register(GPIOx_IDR) or Peripheral Input Data(Alternate Function Output)
- Output Set/Reset Register(GPIOx_ODSET,GPIOx_ODCLR)provides bit write capability for GPIOx_ODR register
- Analog function pin/Debug pin/Digital general purpose pin/Digital function pin alternate
- Allow high flexible alternate functions of GPIO port and peripheral pin

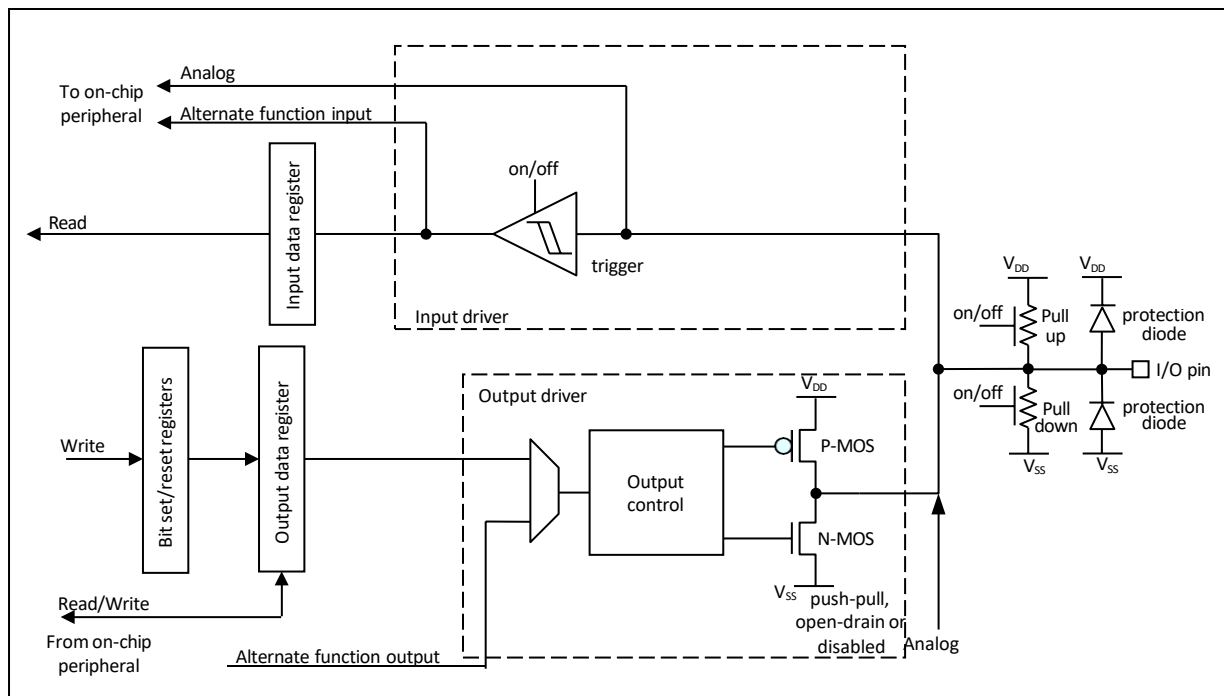
9.3 GPIO Function description

Depending on the specific hardware characteristics of each I/O port listed in the data manual, each bit of a GPIO port can be configured separately by software into a variety of modes:

- Floating input
- Pull-up input
- Drop down to enter
- analog input
- Open drain output with pull-up or pull-down
- Push-pull output with pull-up or pull-down
- Alternate push-pull output with pull-up or pull-down
- Alternate open-drain output with pull-up or pull-down

Each I/O port bit can be freely programmed, and the I/O port register can be accessed as a 32-bit word. GPIOx_ODSET, GPIOx_ODCLR register allows atomic read/modify operations on GPIOx_ODR. In this case, there is no risk that an interruption occurs between read and modify access.

Fig 9.3-1 **GPIO diagram**



Tab 9.3-1 **GPIO configuration**

AFRi[3:0]	DIRi	OTYPi	DRVi	PUPDi		IO Configuration	
0000	1	0	DRVi	0	0	GPIO output	PP
		0		0	1	GPIO output	PP+PU
		0		1	0	GPIO output	PP+PD
		0		1	Reserved		
		1		0	0	GPIO output	OD
		1		0	1	GPIO output	OD+PU
		1		0	0	GPIO output	OD+PD
		1		1	Reserved(output OD)		
	0	x	x	0	0	input	Floating
		x	x	0	1	input	PU
		x	x	1	0	input	PD
		x	x	1	1	Reserved(input floating)	
0001~1110	x	0	DRVi	0	0	AF	PP
		0		0	1	AF	PP+PU
		0		1	0	AF	PP+PD
		0		1	1	Reserved	
		1		0	0	AF	OD
		1		0	1	AF	OD+PU
		1		0	0	AF	OD+PD
		1		1	Reserved		
1111	x	x	x	0	0	input/output	Analog
		x	x	0	1	forbid	
		x	x	1	0		
		x	x	1	1		

i = 0~7

GP=General purpose, PP=Push-pull output, PU=pull-up, PD=pull-down, OD=open-drain , AF=alternate-function

9.3.1 General purpose I/O(GPIO)

During and after reset, multiplexing is not enabled, and all I/O ports except debug pins are configured to float input mode.

After the reset, the debug pin is set to alternate mode with pull-up/pull-down :

- PD1 : SWDCLK pull-down mode
- PC7 : SWDIO pull-up mode

When configured as output mode, the value written to the output data register (GPIOx_ODR) is output to the corresponding pin. Output can be in push-pull mode or open-drain mode (only low levels are driven, high levels represent high resistance).

The input data register (GPIOx_IDR) captures data on the I/O pins at each AHB clock cycle.

All GPIO pins have an internal weak pull-up and pull-down resistor, which are activated or disconnected depending on the value of the GPIOx_PUPD register

9.3.2 I/O port control register

Each general-purpose I/O port has five control registers to configure I/O. The GPIOx_DIRCR register is used to select the direction (input/output). The GPIOx_OTYPER, GPIOx_SLEWCR, and GPIOx_DRVCR registers are used to select the output type (push-pull or open-drain), voltage conversion rate, and drive strength. The GPIOx_PUPDR register is used to select the pull-up or pull-down mode.

9.3.3 I/O port data register

Each GPIO port has two data registers: the input and output data registers (GPIOx_IDR and GPIOx_ODR). GPIOx_ODR is used to store output data, which can be read/write accessed. The input data from the I/O line is stored in the (GPIOx_IDR) register, which is a read-only register

9.3.4 I/O data bit processing

The output setting register (GPIOx_ODSET) and output clearing register (GPIOx_ODCLR) allow applications to set and clear each bit of the output data register (GPIOx_ODR).

When the parity GPIOx_ODSET[i] is written to 1, the corresponding ODR[i] bit is set. When 1 is written to GPIOx_ODCLR[i], the corresponding ODR[i] bit is cleared.

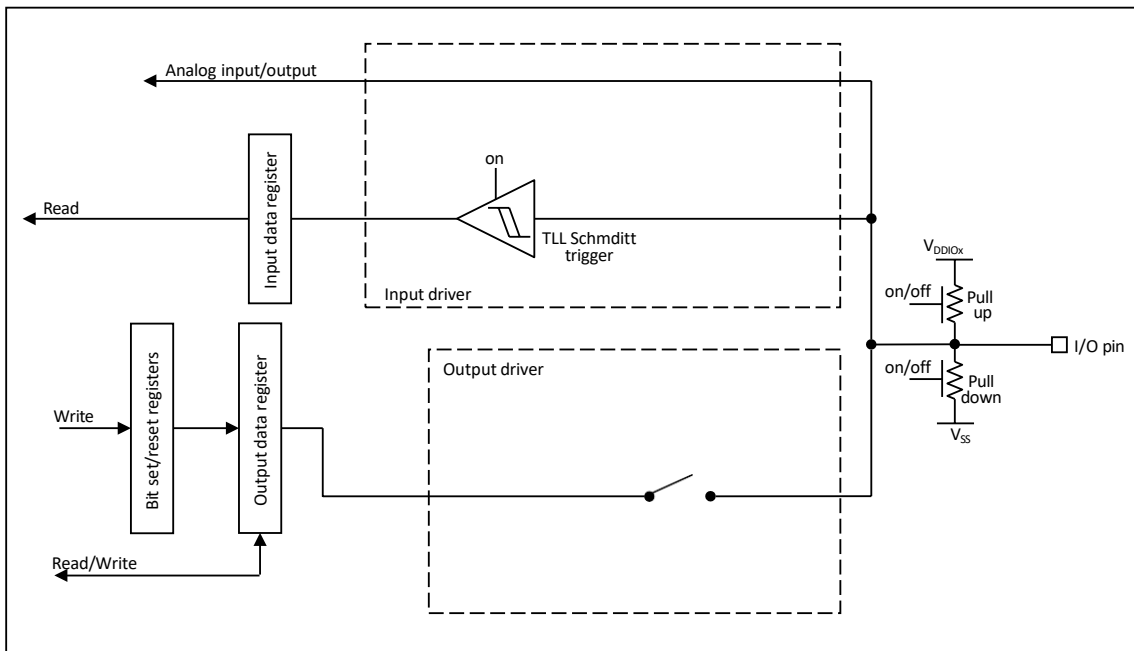
For GPIOx_ODSET, any bit of GPIOx_ODCLR written to 0 does not affect the value of the GPIOx_ODR register. GPIOx_ODSET and GPIOx_ODCLR registers can be used to modify the corresponding bit of GPIOx_ODR. You can also access GPIOx_ODR register directly to change the output. GPIOx_ODSET, GPIOx_ODCLR register provides atomic bit operations on the GPIOx_OD Register.

Set or clear GPIOx_ODR with GPIOx_ODSET, GPIOx_ODCLR, does not require software to turn off interrupts to access GPIOx_ODR register: It is possible to modify 1 or more bits of data in an AHB write access cycle.

9.3.5 Input configuration

When I/O configured as input:

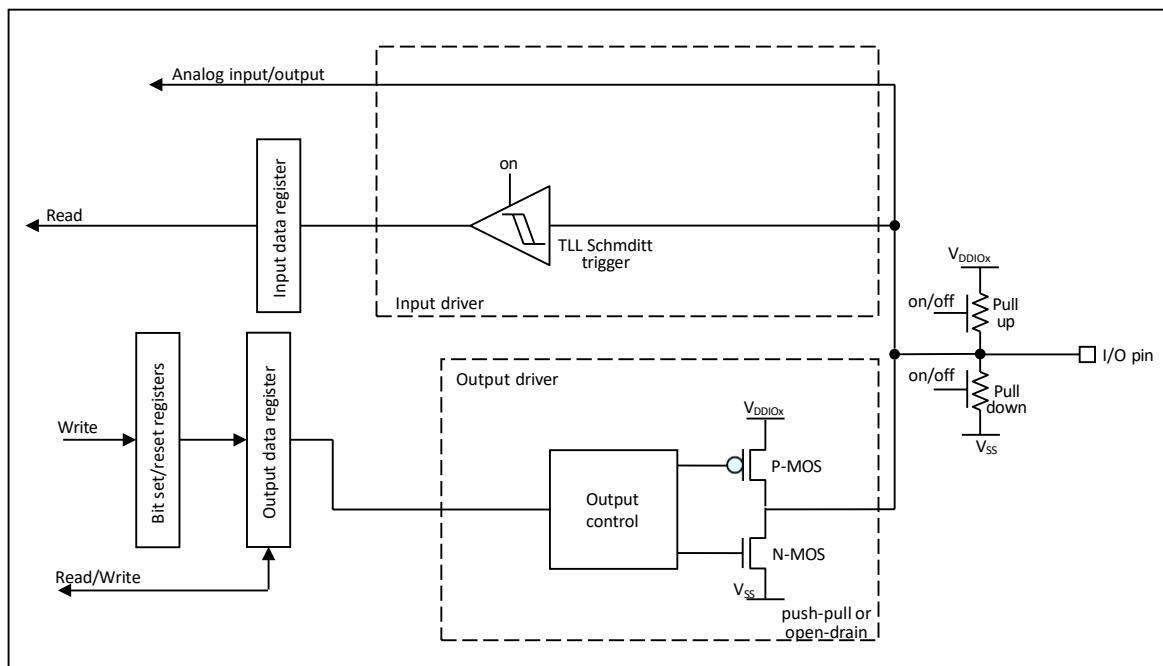
- The output buffer for this port is disabled
- The pull-up and pull-down resistors are activated by the value of the GPIOx_PUPD register
- The data on each AHB clock cycle I/O pin is sampled into the input data register.
- Uses read access to the input data register to get the synchronized input data

Fig 9.3-2 **GPIO input configuration**

9.3.6 Output configuration

The I/O port is configured as output:

- The output buffer is on:
 - Leak-open mode: '0' on the output register activates N-MOS, while '1' on the output register puts the port in a high-impedance state (P-MOS is never activated).
 - Push-pull mode: '0' on the output register activates the N-MOS, while '1' on the output register activates the P-MOS.
- Schmidt triggers input is activated
- Whether the weak pull-up and weak pull-down resistors are activated depends on the value of the GPIOx_PUPDR register
- The data at the I/O pin at every AHB clock cycle is sampled into the input data register
- Uses read access to the output data register to get the synchronized output data

Fig 9.3-3 **GPIO output configuration**

9.3.7 External interrupt/wake up line

All ports have interrupt generation and wake up capabilities. After entering the Sleep or Deep Sleep mode, the WIC module will wake up the chip automatically if an interruption is detected.

Each digital universal port can be interrupted by an external signal source, which can be high level/low level/rising edge/falling edge of four types of signals. When the interrupt is triggered, it can be determined which port has triggered the interrupt by inquiring the interrupt status register, and the corresponding interrupt status flag bit can be cleared by setting the interrupt clearing register.

9.3.8 I/O pin alternate function and remapping

Device I/O is connected to an embedded peripheral module via a multiplexer. Microscopically, only one pin of the peripheral's alternate function is allowed to connect to an I/O interface at a time. Therefore, there can be no conflicting peripheral pin assignments on the same wire. Each I/O pin has an alternate function multiplexer, which can be implemented by configuring the GPIOx_AFR register (from pin 0 to pin 7).

After the reset, all I/O ports are connected to the GPIO function. Each peripheral also has alternate function mapped to different I/O pins, an approach used to optimize more available peripherals on small package devices.

Refer to the specific alternate function of each pin in 9.3.9. In order to use a given I/O port configuration, you must follow the following principles:

- Debugging functions: After each device is reset, these pins are configured to alternate functions immediately to support calls .
- GPIO: Configure the required I/O as output/input in the GPIOx_DIRCR register.
- Alternate functions of peripherals:

- Connect I/O to the desired AFRx, defined in the GPIOx_AFR register;
- The pull-up/pull-down, output speed and output capacity of the corresponding pins are configured for the GPIOx_PUPDR, GPIOx_SLEWCR and GPIOx_DRVCR registers;
- The output type is configured with the GPIOx_OTYPER register: 0 : push-pull ;1 : open-drain
- Additional features:
 - For ADC/VCMP, set GPIOx_AFR to 0x0F, set the required I/O port to analog mode, and configure the required function in the ADC or VCMP register.
 - For additional function oscillators, configure the required function in the associated RCC register

9.3.9 General input/output (GPIO)

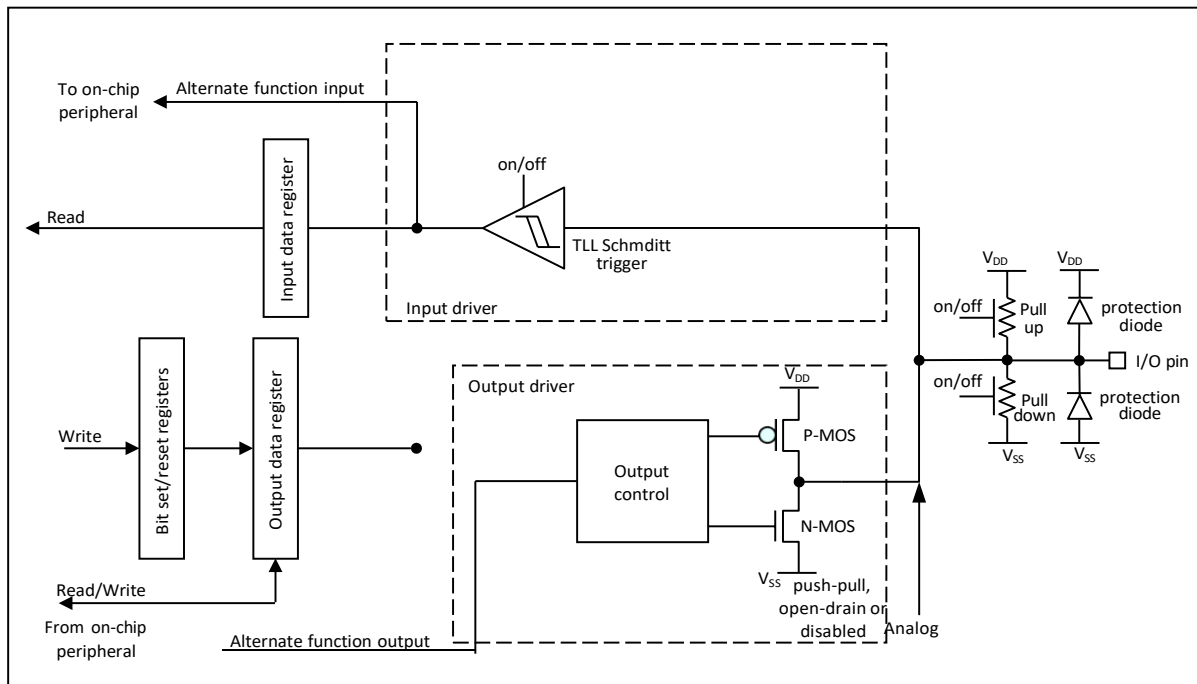
Tab 9.3-2 GPIO multiplexing

Package			GPIO Multiplexing											
LOFP32/QFN32	TSSOP/SOP20	QFPN20	name	0	1	2	3	4	5	6	7	8	9	F
			29	1	18	PD4		TIM1_CH1	PCA_CH0	RTC_1HZ	TIM10_TOG	UART1_TXD	TIM10_EXT	BEEP
30	2	19	PD5		TIM1_CH1N	PCA_CH4	SPI_MISO	I2C_SCL	UART2_TXD	TIM10_GATE	UART1_TXD	TIM2_CH4		AIN5
31	3	20	PD6		TIM1_CH2	PCA_CH3	SPI_MOSI	I2C_SDA	UART2_RXD	LPTIM_EXT	UART1_RXD	TIM2_CH2		AIN6
1	4	1	NRST											
2	5	2	PA1		TIM1_CH2N		SPI_CLK	I2C_SDA	UART1_RXD	TIM10_TOG	UART2_RXD			OSC_IN
3	6	3	PA2		TIM1_CH3		SPI_NSS	I2C_SCL	UART1_TXD	TIM10_TOGN	UART2_TXD	TIM2_CH2		OSC_OUT
4	7	4	VSS											
5	8	5	VCAP											
6	9	6	VDD											
7	10	7	PA3		TIM1_CH3N	PCA_CH2	SPI_NSS	RTC_1HZ	LPUART_RXD	PCA_ECI	VCMP0_OUT	TIM2_CH3	UART2_TXD	
11	11	8	PB5		TIM1_BKIN	PCA_CH4	SPI_CLK	I2C_SDA	UART1_RXD	TIM11_TOG	LVD_OUT	TIM2_CH1		X32K_IN
12	12	9	PB4		LPTIM_GATE	PCA_ECI	SPI_NSS	I2C_SCL	UART1_TXD	TIM11_TOGN				X32K_OUT
20	13	10	PC3		TIM1_CH3	TIM1_CH1N		I2C_SDA	UART2_TXD	PCA_CH1	1-WIRE	TIM2_CH3		AIN1
21	14	11	PC4		TIM1_CH4	TIM1_CH2N		I2C_SCL	UART2_RXD	PCA_CH0	CLK_MCO	TIM2_CH4		AIN2
22	15	12	PC5		TIM1_BKIN	PCA_CH0	SPI_CLK		LPUART_TXD	TIM11_GATE	LVD_OUT	TIM2_CH1		VCMPIN1
23	16	13	PC6		TIM1_CH1	PCA_CH3	SPI_MOSI		LPUART_RXD	TIM11_EXT	CLK_MCO	TIM2_CH4		AIN0
24	17	14	SWDIO	PC7	TIM1_CH2	PCA_CH4	SPI_MISO		UART2_RXD	LSI_OUT	LSE_OUT			
26	18	15	SWDCLK	PD1		PCA_ECI			UART2_TXD	HSE_OUT	VCMP0_OUT			
27	19	16		PD2	TIM1_CH2	PCA_CH2	SPI_MISO	RTC_1HZ	LPUART_TXD	LPTIM_TOG	1-WIRE	TIM2_CH3		AIN3/ VCMPIN0
28	20	17		PD3	TIM1_CH3N	PCA_CH1	SPI_MOSI	LSE_OUT	UART1_RXD	LPTIM_TOGN		TIM2_CH2		AIN4
8			PA4		TIM1_CH2N		SPI_CLK	I2C_SDA	UART1_RXD	TIM10_TOG	UART2_RXD			AIN14
9			PB7		TIM1_CH3N	PCA_CH2	SPI_NSS	RTC_1HZ	LPUART_RXD	PCA_ECI	VCMP0_OUT	TIM2_CH3		AIN8
10			PB6		TIM1_CH1N	PCA_CH4	SPI_MISO	I2C_SCL	UART2_TXD	TIM10_GATE	UART1_TXD	TIM2_CH4		AIN9
13			PB3		TIM1_CH2	PCA_CH3	SPI_MOSI	I2C_SDA	UART2_RXD	LPTIM_EXT	UART1_RXD	TIM2_CH2		AIN10
14			PB2		TIM1_CH3N	PCA_CH1	SPI_MOSI	LSE_OUT	UART1_RXD	LPTIM_TOGN		TIM2_CH2		AIN11
15			PB1		TIM1_CH1	PCA_CH0	RTC_1HZ	TIM10_TOG	UART1_TXD	TIM10_EXT	BEEP	TIM2_CH1	TIM1_CH2N	AIN12
16			PB0		TIM1_CH3	TIM1_CH1N		I2C_SDA	UART2_TXD	PCA_CH1	1-WIRE	TIM2_CH3		AIN13
17			PC0		TIM1_CH3		SPI_NSS	I2C_SCL	UART1_TXD	TIM10_TOGN	UART2_TXD	TIM2_CH2	TIM1_CH1N	AIN15
18			PC1		TIM1_CH1	PCA_CH3	SPI_MOSI		LPUART_RXD	TIM11_EXT	CLK_MCO	TIM2_CH4	TIM1_CH2N	
19			PC2		TIM1_CH2	PCA_CH2	SPI_MISO	RTC_1HZ	LPUART_TXD	LPTIM_TOG	1-WIRE	TIM2_CH3	TIM1_CH3N	
25			PD0		TIM1_BKIN	PCA_CH0	SPI_CLK		LPUART_TXD	TIM11_GATE	LVD_OUT	TIM2_CH1		
32					PD7	TIM1_CH4	TIM1_CH2N	I2C_SCL		UART2_RXD	PCA_CH0	CLK_MCO	TIM2_CH4	

When I/O ports are configured for alternate function :

- The output buffer can be configured in open-drain or push-pull mode
- The peripheral's signal drives the output buffer
- The Schmidt trigger input is activated
- Whether the weak pull-up and weak pull-down resistors are activated depends on the value of the GPIOx_PUPDR register
- The data on each AHB clock cycle I/O pin is sampled into the input data register
- Uses read access to the input data register to get the synchronized input data

Fig 9.3-4 **GPIO multiplexing configuration**

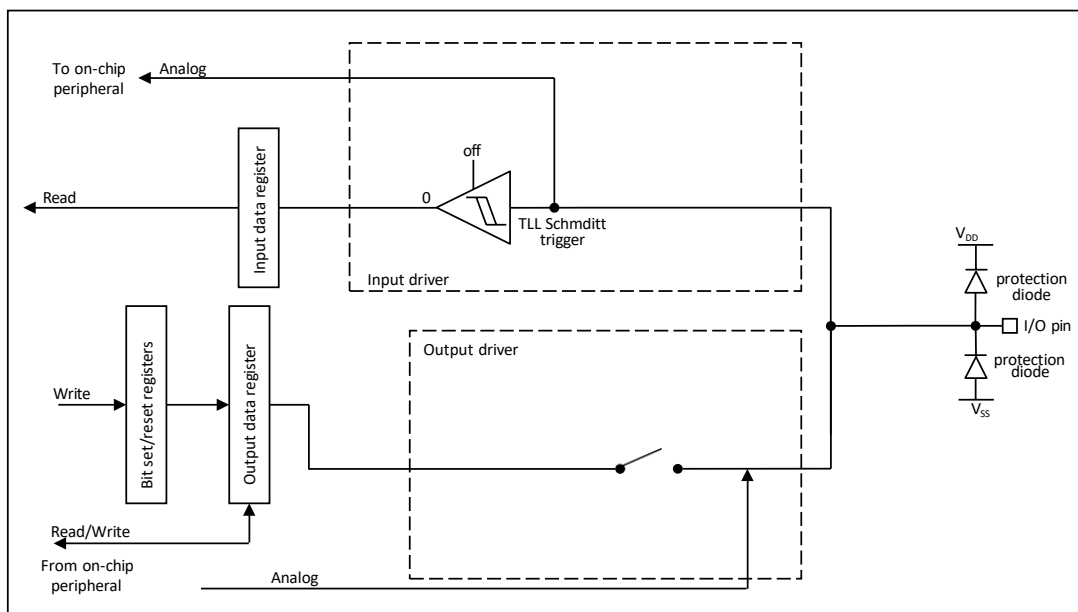


9.3.10 Analog configuration

When the I/O port is configured as analog mode (GPIOx_AFR.AFR=0x0F) :

- The output buffer is closed
- Disables Schmitt trigger input, achieving zero consumption on each analog I/O pin. Schmitt trigger output is forcibly set to '0'
- Weak pull-up and pull-down resistors are prohibited
- Reads a value of 0 from the input data register

Fig 9.3-5 **GPIO high impedance mode**



9.3.11 HSE or LSE pins are used as GPIO

When the HSE or LSE oscillator is turned off (default after reset), the associated oscillator pin can be used as a regular GPIO port. When the HSE or LSE oscillator is turned on (RCC_SYSCCLKCR.HSEEN or RCC_LSECR.LSEEN bit is set to turn on), the corresponding pin must be configured for analog function, the oscillator controls its associated pin and the GPIO configuration of the associated pin is invalid. Method of configuring the oscillator pin for analog function:

- Set RCC_SYSCCLKCR.HSEPORT = 1 or RCC_LSECR.LSEPORT = 1
- Configured by setting GPIOx_AFR = 0xF for the corresponding pin

When the oscillator is configured with the user's external clock input mode (RCC_SYSCCLKCR.HSEBYP=1 or RCC_LSECR.LSEBYP=1), it is not necessary to configure the corresponding pins as analog function, and only OSC_IN or X32K_IN pins are used for clock input processing. The OSC_OUT or X32K_OUT pins can still be configured as normal GPIO pins.

9.4 GPIO Registers

Tab 9.4-1 GPIO Registers map

Offset	Register	Description
GPIOx address: 0x40021000 (x = A, B, C, D)		
0x000	GPIOA	GPIOA Offset address
0x400	GPIOB	GPIOB Offset address
0x800	GPIOC	GPIOC Offset address
0xC00	GPIOD	GPIOD Offset address

Offset	Register	Reset value	Description
0x00	GPIOx_DIRCR	0x0000_0000	I/O mode register
0x04	GPIOx_OTYPER	0x0000_0000	Output type register
0x08	GPIOx_ODR	0x0000_0000	Output data register
0x0C	GPIOx_IDR	0x0000_00XX	Input data register
0x10	GPIOx_INTEN	0x0000_0000	Interrupt enable register
0x14	GPIOx_RAWINTSR	0x0000_0000	Interrupt the original status register, read only. The interrupt status can be read whether the interrupt is enabled or not
0x18	GPIOx_MSKINTCR	0x0000_0000	Interrupt status register, read only
0x1C	GPIOx_INTCLR	0x0000_0000	Interrupt clear register
0x20	GPIOx_INTTYPCR	0x0000_0000	Interrupt type register
0x24	GPIOx_INTPOLCR	0x0000_0000	Interrupt type value register
0x28	GPIOx_INTANY	0x0000_0000	Arbitrary edge trigger interrupt register
0x2C	GPIOx_ODSET	0x0000_0000	Output set register
0x30	GPIOx_ODCLR	0x0000_0000	Output clear register
0x34	GPIOx_INDBEN	0x0000_0000	Input de jitter and sync enable registers
0x38	GPIOx_DBCLKCR	0x0000_0000	Input de jitter clock configuration register
0x3C	GPIOx_PUPDR	0x0100_0000	Pull up/drop down register
0x40	GPIOA_SLEWCR	0x0000 000E	Voltage conversion rate control
0x40	GPIOB_SLEWCR	0x0000 0030	Voltage conversion rate control
0x40	GPIOC_SLEWCR	0x0000 00F8	Voltage conversion rate control
0x40	GPIOD_SLEWCR	0x0000 007E	Voltage conversion rate control
0x44	GPIOx_DRVCR	0x0000 0000	Drive strength configuration
0x48	GPIOx_AFR	0x0000_0000	Alternate function register

0: indicates the logical value 0

1: indicates the logical value X:

uncertainty

9.5 GPIO Register Description

9.5.1 GPIO port direction register(GPIOx_DIRCR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_DIRCR	0x00	R/W	0x0000_0000	I/O mode register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxDIR7	PxDIR6	PxDIR5	PxDIR4	PxDIR3	PxDIR2	PxDIR1	PxDIR0

GPIO port direction register(GPIOx_DIRCR)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	RW	PxDIR7: Select the input/output direction of the terminal Px7 0: input 1: Output
[6]	RW	PxDIR6: Select the input/output direction of the terminal Px6 0: input 1: Output
[5]	RW	PxDIR5: Select the input/output direction of the terminal Px5 0: input 1: Output
[4]	RW	PxDIR4: Select the input/output direction of the terminal Px4 0: input 1: Output
[3]	RW	PxDIR3: Select the input/output direction of the terminal Px3 0: input 1: Output
[2]	RW	PxDIR2: Select the input/output direction of the terminal Px2 0: input 1: Output
[1]	RW	PxDIR1: Select the input/output direction of the terminal Px1 0: input 1: Output
[0]	RW	PxDIR0: Select the input/output direction of the terminal Px0 0: input 1: Output

9.5.2 GPIO Port output type register(GPIOx_OTYPER)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_OTYPER	0x04	R/W	0x0000_0000	Output type register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxOTYP7	PxOTYP6	PxOTYP5	PxOTYP4	PxOTYP3	PxOTYP2	PxOTYP1	PxOTYP0

GPIO Port output type register(GPIOx_OTYPER)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	RW	PxOTYP7: Port Px7 output type control 0: push-pull output (state after reset) 1: Open leakage output
[6]	RW	PxOTYP6: Port Px6 output type control 0: push-pull output (state after reset) 1: Open leakage output
[5]	RW	PxOTYP5: Port Px5 output type control 0: push-pull output (state after reset) 1: Open leakage output
[4]	RW	PxOTYP4: Port Px4 output type control 0: push-pull output (state after reset) 1: Open leakage output
[3]	RW	PxOTYP3: Port Px3 output type control 0: push-pull output (state after reset) 1: Open leakage output
[2]	RW	PxOTYP2: Port Px2 output type control 0: push-pull output (state after reset) 1: Open leakage output
[1]	RW	PxOTYP1: Port Px1 output type control 0: push-pull output (state after reset) 1: Open leakage output
[0]	RW	PxOTYP0: Port Px0 output type control 0: push-pull output (state after reset) 1: Open leakage output

9.5.3 GPIO Port output data register(GPIOx_ODR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_ODR	0x08	R/W	0x0000_0000	Output data register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxOD7	PxOD6	PxOD5	PxOD4	PxOD3	PxOD2	PxOD1	PxOD0

GPIO Port output data register(GPIOx_ODR)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	RW	PxOD7: Configure the output value of the port Px7 0: indicates the output low level 1: High output level. If it is open and leaky output, external pull-up resistor is required
[6]	RW	PxOD6: Configure the output value of the port Px6 0: indicates the output low level 1: High output level. If it is open and leaky output, external pull-up resistor is required
[5]	RW	PxOD5: Configure the output value of the port Px5 0: indicates the output low level 1: High output level. If it is open and leaky output, external pull-up resistor is required
[4]	RW	PxOD4: Configure the output value of the port Px4 0: indicates the output low level 1: High output level. If it is open and leaky output, external pull-up resistor is required
[3]	RW	PxOD3: Configure the output value of the port Px3 0: indicates the output low level 1: High output level. If it is open and leaky output, external pull-up resistor is required
[2]	RW	PxOD2: Configure the output value of the port Px2 0: indicates the output low level 1: High output level. If it is open and leaky output, external pull-up resistor is required
[1]	RW	PxOD1: Configure the output value of the port Px1 0: indicates the output low level 1: High output level. If it is open and leaky output, external pull-up resistor is required
[0]	RW	PxOD0: Configure the output value of the port Px0 0: indicates the output low level 1: High output level. If it is open and leaky output, external pull-up resistor is required

Note : For atom bit setting/clearing, GPIOx_ODSET, GPIOx_ODCLR(x= A.. D) register operation to achieve.

9.5.4 GPIO Port input data register(GPIOx_IDR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_IDR	0x0C	R	0x0000_00xx	Input data register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxID7	PxID6	PxID5	PxID4	PxID3	PxID2	PxID1	PxID0

GPIO Port input data register(GPIOx_IDR)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	R	PxID7: Input value of port Px7 0: input low level 1: Input high level
[6]	R	PxID6: Input value of port Px6 0: input low level 1: Input high level
[5]	R	PxID5: Input value of port Px5 0: input low level 1: Input high level
[4]	R	PxID4: Input value of port Px4 0: input low level 1: Input high level
[3]	R	PxID3: Input value of port Px3 0: input low level 1: Input high level
[2]	R	PxID2: Input value of port Px2 0: input low level 1: Input high level
[1]	R	PxID1: Input value of port Px1 0: input low level 1: Input high level
[0]	R	PxID0: Input value of port Px0 0: input low level 1: Input high level

Note : x: uncertainty

9.5.5 GPIO Port interrupt enable register(GPIOx_INTEN)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_INTEN	0x10	R/W	0x0000_0000	Interrupt enable register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxIEN7	PxIEN6	PxIEN5	PxIEN4	PxIEN3	PxIEN2	PxIEN1	PxIEN0

GPIO Port interrupt enable register(GPIOx_INTEN)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	RW	PxIEN7: Port Px7 interrupt mask is lifted 0: The Px7 terminal interrupt is masked 1: The Px7 terminal interrupt is enabled
[6]	RW	PxIEN6: Port Px6 interrupt mask is lifted 0: The Px6 terminal interrupt is masked 1: The Px6 terminal interrupt is enabled
[5]	RW	PxIEN5: Port Px5 interrupt mask is lifted 0: The Px5 terminal interrupt is masked 1: The Px5 terminal interrupt is enabled
[4]	RW	PxIEN4: Port Px4 interrupt mask is lifted 0: The Px4 terminal interrupt is masked 1: The Px4 terminal interrupt is enabled
[3]	RW	PxIEN3: Port Px3 interrupt mask is lifted 0: The Px3 terminal interrupt is masked 1: The Px3 terminal interrupt is enabled
[2]	RW	PxIEN2: Port Px2 interrupt mask is lifted 0: The Px2 terminal interrupt is masked 1: The Px2 terminal interrupt is enabled
[1]	RW	PxIEN1: Port Px1 interrupt mask is lifted 0: The Px1 terminal interrupt is masked 1: The Px1 terminal interrupt is enabled
[0]	RW	PxIEN0: Port Px0 interrupt mask is lifted 0: The Px0 terminal interrupt is masked 1: The Px0 terminal interrupt is enabled

9.5.6 GPIO Port interrupt original status register(GPIOx_RAWINTSR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_RAWINTSR	0x14	R	0x0000_0000	Interrupt the original status register, read only. The interrupt status can be read whether the interrupt is enabled or not

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

GPIO Port interrupt original status register(GPIOx_RAWINTSR)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	R	PxRIS7: Port Px7 interrupts the original state 0: no interruption 1: An interrupt occurs
[6]	R	PxRIS6: Port Px6 interrupts the original state 0: no interruption 1: An interrupt occurs
[5]	R	PxRIS5: Port Px5 interrupts the original state 0: no interruption 1: An interrupt occurs
[4]	R	PxRIS4: Port Px4 interrupts the original state 0: no interruption 1: An interrupt occurs
[3]	R	PxRIS3: Port Px3 interrupts the original state 0: no interruption 1: An interrupt occurs
[2]	R	PxRIS2: Port Px2 interrupts the original state 0: no interruption 1: An interrupt occurs
[1]	R	PxRIS1: Port Px1 interrupts the original state 0: no interruption 1: An interrupt occurs
[0]	R	PxRIS0: Port Px0 interrupts the original state 0: no interruption 1: An interrupt occurs

9.5.7 GPIO Port interrupt status register(GPIOx_MSKINTSR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_MSKINTSR	0x18	R	0x0000_0000	Interrupt status register, read only

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxMIS7	PxMIS6	PxMIS5	PxMIS4	PxMIS3	PxMIS2	PxMIS1	PxMIS0

GPIO Port interrupt status register(GPIOx_MSKINTSR)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	R	PxMIS7: Interrupt status of the masked port Px7 0: no interruption 1: An interrupt occurs
[6]	R	PxMIS6: Interrupt status of the masked port Px6 0: no interruption 1: An interrupt occurs
[5]	R	PxMIS5: Interrupt status of the masked port Px5 0: no interruption 1: An interrupt occurs
[4]	R	PxMIS4: Interrupt status of the masked port Px4 0: no interruption 1: An interrupt occurs
[3]	R	PxMIS3: Interrupt status of the masked port Px3 0: no interruption 1: An interrupt occurs
[2]	R	PxMIS2: Interrupt status of the masked port Px2 0: no interruption 1: An interrupt occurs
[1]	R	PxMIS1: Interrupt status of the masked port Px1 0: no interruption 1: An interrupt occurs
[0]	R	PxMIS0: Interrupt status of the masked port Px0 0: no interruption 1: An interrupt occurs

Note : These bits are read-only and set by the hardware. The interrupt status can be read only when the interrupt status is enabled.

9.5.8 GPIO Port interrupt clear register(GPIOx_INTCLR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_INTCLR	0x1C	W	0x0000_0000	Interrupt clear register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxICLR7	PxICLR6	PxICLR5	PxICLR4	PxICLR3	PxICLR2	PxICLR1	PxICLR0

GPIO Port interrupt clear register(GPIOx_INTCLR)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	W	PxICLR7: Interrupt status of port Px7 0: The interrupt flag bit is reserved 1: clears the corresponding interrupt flag bit
[6]	W	PxICLR6: Interrupt status of port Px6 0: The interrupt flag bit is reserved 1: clears the corresponding interrupt flag bit
[5]	W	PxICLR5: Interrupt status of port Px5 0: The interrupt flag bit is reserved 1: clears the corresponding interrupt flag bit
[4]	W	PxICLR4: Interrupt status of port Px4 0: The interrupt flag bit is reserved 1: clears the corresponding interrupt flag bit
[3]	W	PxICLR3: Interrupt status of port Px3 0: The interrupt flag bit is reserved 1: clears the corresponding interrupt flag bit
[2]	W	PxICLR2: Interrupt status of port Px2 0: The interrupt flag bit is reserved 1: clears the corresponding interrupt flag bit
[1]	W	PxICLR1: Interrupt status of port Px1 0: The interrupt flag bit is reserved 1: clears the corresponding interrupt flag bit
[0]	W	PxICLR0: Interrupt status of port Px0 0: The interrupt flag bit is reserved 1: clears the corresponding interrupt flag bit

Note : These bits are written only and are used by the software to clear interrupts

9.5.9 GPIO Port interrupt type register(GPIOx_INTTYPCR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_INTTYPCR	0x20	RW	0x0000_0000	Interrupt type register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8

Reserved							
7	6	5	4	3	2	1	0
PxIYPE7	PxIYPE6	PxIYPE5	PxIYPE4	PxIYPE3	PxIYPE2	PxIYPE1	PxIYPE0

GPIO Port interrupt type register(GPIOx_INTTYPCR)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	RW	PxIYPE7: Set the interrupt type of the port Px7 0: edge triggers interrupt 1: level triggered interrupt
[6]	RW	PxIYPE6: Set the interrupt type of the port Px6 0: edge triggers interrupt 1: level triggered interrupt
[5]	RW	PxIYPE5: Set the interrupt type of the port Px5 0: edge triggers interrupt 1: level triggered interrupt
[4]	RW	PxIYPE4: Set the interrupt type of the port Px4 0: edge triggers interrupt 1: level triggered interrupt
[3]	RW	PxIYPE3: Set the interrupt type of the port Px3 0: edge triggers interrupt 1: level triggered interrupt
[2]	RW	PxIYPE2: Set the interrupt type of the port Px2 0: edge triggers interrupt 1: level triggered interrupt
[1]	RW	PxIYPE1: Set the interrupt type of the port Px1 0: edge triggers interrupt 1: level triggered interrupt
[0]	RW	PxIYPE0: Set the interrupt type of the port Px0 0: edge triggers interrupt 1: level triggered interrupt

9.5.10 GPIO Interrupt type value register(GPIOx_INTPOLCR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_INTPOLCR	0x24	RW	0x0000_0000	Interrupt type value register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxITPxIVAL7	PxITPxIVAL6	PxITPxIVAL5	PxITPxIVAL4	PxITPxIVAL3	PxITPxIVAL2	PxITPxIVAL1	PxITPxIVAL0

GPIO Interrupt type value register(GPIOx_INTPOLCR)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved

[7]	RW	PxIVAL7: Interrupt polarity configuration of terminal Px7 0: Low level or falling edge trigger 1: High level or rising edge trigger
[6]	RW	PxIVAL6: Interrupt polarity configuration of terminal Px6 0: Low level or falling edge trigger 1: High level or rising edge trigger
[5]	RW	PxIVAL5: Interrupt polarity configuration of terminal Px5 0: Low level or falling edge trigger 1: High level or rising edge trigger
[4]	RW	PxIVAL4: Interrupt polarity configuration of terminal Px4 0: Low level or falling edge trigger 1: High level or rising edge trigger
[3]	RW	PxIVAL3: Interrupt polarity configuration of terminal Px3 0: Low level or falling edge trigger 1: High level or rising edge trigger
[2]	RW	PxIVAL2: Interrupt polarity configuration of terminal Px2 0: Low level or falling edge trigger 1: High level or rising edge trigger
[1]	RW	PxIVAL1: Interrupt polarity configuration of terminal Px1 0: Low level or falling edge trigger 1: High level or rising edge trigger
[0]	RW	PxIVAL0: Interrupt polarity configuration of terminal Px0 0: Low level or falling edge trigger 1: High level or rising edge trigger

9.5.11 GPIO port arbitrary edge trigger interrupt register(GPIOx_INTANY)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_INTANY	0x28	RW	0x0000_0000	Arbitrary edge trigger interrupt register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxIANY7	PxIANY6	PxIANY5	PxIANY4	PxIANY3	PxIANY2	PxIANY1	PxIANY0

GPIO port arbitrary edge trigger interrupt register(GPIOx_INTANY)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	RW	PxIANY7: Port Px7 is configured to trigger an interrupt along arbitrary edge 0: The trigger edge is determined by PxIVAL7 1: Both rising and falling edges are triggered
[6]	RW	PxIANY6: Port Px6 is configured to trigger an interrupt along arbitrary edge 0: The trigger edge is determined by PxIVAL6 1: Both rising and falling edges are triggered

[5]	RW	PxIANY5: Port Px5 is configured to trigger an interrupt along arbitrary edge 0: The trigger edge is determined by PxIVAL5 1: Both rising and falling edges are triggered
[4]	RW	PxIANY4: Port Px4 is configured to trigger an interrupt along arbitrary edge 0: The trigger edge is determined by PxIVAL4 1: Both rising and falling edges are triggered
[3]	RW	PxIANY3: Port Px3 is configured to trigger an interrupt along arbitrary edge 0: The trigger edge is determined by PxIVAL3 1: Both rising and falling edges are triggered
[2]	RW	PxIANY2: Port Px2 is configured to trigger an interrupt along arbitrary edge 0: The trigger edge is determined by PxIVAL2 1: Both rising and falling edges are triggered
[1]	RW	PxIANY1: Port Px1 is configured to trigger an interrupt along arbitrary edge 0: The trigger edge is determined by PxIVAL1 1: Both rising and falling edges are triggered
[0]	RW	PxIANY0: Port Px0 is configured to trigger an interrupt along arbitrary edge 0: The trigger edge is determined by PxIVAL0 1: Both rising and falling edges are triggered

9.5.12 GPIO Port output set register(GPIOx_ODSET)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_ODSET	0x2C	W	0x0000_0000	Output set register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxODSET7	PxODSET6	PxODSET5	PxODSET4	PxODSET3	PxODSET2	PxODSET1	PxODSET0

GPIO Port output set register(GPIOx_ODSET)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	W	PxODSET7: The output of port Px7 is controlled by 1 0: output hold 1: output is high
[6]	W	PxODSET6: The output of port Px6 is controlled by 1 0: output hold 1: output is high
[5]	W	PxODSET5: The output of port Px5 is controlled by 1 0: output hold 1: output is high
[4]	W	PxODSET4: The output of port Px4 is controlled by 1 0: output hold 1: output is high

[3]	W	PxODSET3: The output of port Px3 is controlled by 1 0: output hold 1: output is high
[2]	W	PxODSET2: The output of port Px2 is controlled by 1 0: output hold 1: output is high
[1]	W	PxODSET1: The output of port Px1 is controlled by 1 0: output hold 1: output is high
[0]	W	PxODSET0: The output of port Px0 is controlled by 1 0: output hold 1: output is high

9.5.13 GPIO Port output clear register(GPIOx_ODCLR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_ODCLR	0x30	W	0x0000_0000	Output clear register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxODCLR7	PxODCLR6	PxODCLR5	PxODCLR4	PxODCLR3	PxODCLR2	PxODCLR1	PxODCLR0

GPIO Port output clear register(GPIOx_ODCLR)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	W	PxODCLR7: Terminal Px7 output clear 0 control 0: output hold 1: Reset the corresponding ODRx
[6]	W	PxODCLR6: Terminal Px6 output clear 0 control 0: output hold 1: Reset the corresponding ODRx
[5]	W	PxODCLR5: Terminal Px5 output clear 0 control 0: output hold 1: Reset the corresponding ODRx
[4]	W	PxODCLR4: Terminal Px4 output clear 0 control 0: output hold 1: Reset the corresponding ODRx
[3]	W	PxODCLR3: Terminal Px3 output clear 0 control 0: output hold 1: Reset the corresponding ODRx
[2]	W	PxODCLR2: Terminal Px2 output clear 0 control 0: output hold 1: Reset the corresponding ODRx

[1]	W	PxODCLR1: Terminal Px1 output clear 0 control 0: output hold 1: Reset the corresponding ODRx
[0]	W	PxODCLR0: Terminal Px0 output clear 0 control 0: output hold 1: Reset the corresponding ODRx

Note : If both ODSETx and ODCLR_x are set, ODSETx takes precedence

9.5.14 GPIO Port input dejitter register(GPIOx_INDBEN)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_INDBEN	0x34	RW	0x0000_0000	Input dejitter and sync enable registers

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

GPIO Port input dejitter register(GPIOx_INDBEN)(x = A..D)

Bit	Access	Description
[31:9]	-	Reserved
[8]	RW	SYNC_EN: Jitter Elimination When not enabled, this register is used to configure the input to use level 2 synchronization to eliminate metastable state (valid only for interrupts). 0: two-level synchronization is not used 1: Two-level synchronization is used
[7:0]	W	PxDIDBnN=(0 7): Port Pxn(n=0 to 7) jitter elimination enable configuration bit. If the input cannot be sampled for two consecutive buffering sampling periods, the input signal is regarded as jitter and no interruption is triggered. It is only used for edge touch "edge-trigger" interrupts and cannot be used for level trigger interrupts. In level mode, it is a two-stage synchronous input 0: disables the port jitter elimination function 1: Enables the port jitter elimination function

9.5.15 GPIO Port input dejitter clock configuration register(GPIOx_DBCLKCR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_DBCLKCR	0x38	RW	0x0000_0000	Input dejitter clock configuration register

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

GPIO Port input dejitter clock configuration register(GPIOx_DBCLKCR)(x = A..D)

Bit	Access	Description																																		
[31:5]	-	Reserved																																		
[4]	RW	DBCLKEN: Whether to enable clock dejitter 0: Clock dejitter is disabled 1: enables clock dejitter																																		
[3:0]	RW	<p>DBCLK_DIV: Selection of shake off sampling period</p> <table border="1"> <thead> <tr> <th>DBCLK_DIV</th><th>description</th></tr> </thead> <tbody> <tr><td>0x0</td><td>dejitter sampling 1 HCLK cycle 1 time</td></tr> <tr><td>0x1</td><td>dejitter sampling 2 HCLK cycle 1 time</td></tr> <tr><td>0x2</td><td>dejitter sampling 4 HCLK cycle 1 time</td></tr> <tr><td>0x3</td><td>dejitter sampling 8 HCLK cycle 1 time</td></tr> <tr><td>0x4</td><td>dejitter sampling 16 HCLK cycle 1 time</td></tr> <tr><td>0x5</td><td>dejitter sampling 32 HCLK cycle 1 time</td></tr> <tr><td>0x6</td><td>dejitter sampling 64 HCLK cycle 1 time</td></tr> <tr><td>0x7</td><td>dejitter sampling 128 HCLK cycle 1 time</td></tr> <tr><td>0x8</td><td>dejitter sampling 256 HCLK cycle 1 time</td></tr> <tr><td>0x9</td><td>dejitter sampling 512 HCLK cycle 1 time</td></tr> <tr><td>0xA</td><td>dejitter sampling 1024 HCLK cycle 1 time</td></tr> <tr><td>0xB</td><td>dejitter sampling 2*1024 HCLK cycle 1 time</td></tr> <tr><td>0xC</td><td>dejitter sampling 4*1024 HCLK cycle 1 time</td></tr> <tr><td>0xD</td><td>dejitter sampling 8*1024 HCLK cycle 1 time</td></tr> <tr><td>0xE</td><td>dejitter sampling 16*1024 HCLK cycle 1 time</td></tr> <tr><td>0xF</td><td>dejitter sampling 32*1024 HCLK cycle 1 time</td></tr> </tbody> </table> <p>The Debounce clock is hclk ($2^{dbclk_div[3:0]}$)frequency division</p>	DBCLK_DIV	description	0x0	dejitter sampling 1 HCLK cycle 1 time	0x1	dejitter sampling 2 HCLK cycle 1 time	0x2	dejitter sampling 4 HCLK cycle 1 time	0x3	dejitter sampling 8 HCLK cycle 1 time	0x4	dejitter sampling 16 HCLK cycle 1 time	0x5	dejitter sampling 32 HCLK cycle 1 time	0x6	dejitter sampling 64 HCLK cycle 1 time	0x7	dejitter sampling 128 HCLK cycle 1 time	0x8	dejitter sampling 256 HCLK cycle 1 time	0x9	dejitter sampling 512 HCLK cycle 1 time	0xA	dejitter sampling 1024 HCLK cycle 1 time	0xB	dejitter sampling 2*1024 HCLK cycle 1 time	0xC	dejitter sampling 4*1024 HCLK cycle 1 time	0xD	dejitter sampling 8*1024 HCLK cycle 1 time	0xE	dejitter sampling 16*1024 HCLK cycle 1 time	0xF	dejitter sampling 32*1024 HCLK cycle 1 time
DBCLK_DIV	description																																			
0x0	dejitter sampling 1 HCLK cycle 1 time																																			
0x1	dejitter sampling 2 HCLK cycle 1 time																																			
0x2	dejitter sampling 4 HCLK cycle 1 time																																			
0x3	dejitter sampling 8 HCLK cycle 1 time																																			
0x4	dejitter sampling 16 HCLK cycle 1 time																																			
0x5	dejitter sampling 32 HCLK cycle 1 time																																			
0x6	dejitter sampling 64 HCLK cycle 1 time																																			
0x7	dejitter sampling 128 HCLK cycle 1 time																																			
0x8	dejitter sampling 256 HCLK cycle 1 time																																			
0x9	dejitter sampling 512 HCLK cycle 1 time																																			
0xA	dejitter sampling 1024 HCLK cycle 1 time																																			
0xB	dejitter sampling 2*1024 HCLK cycle 1 time																																			
0xC	dejitter sampling 4*1024 HCLK cycle 1 time																																			
0xD	dejitter sampling 8*1024 HCLK cycle 1 time																																			
0xE	dejitter sampling 16*1024 HCLK cycle 1 time																																			
0xF	dejitter sampling 32*1024 HCLK cycle 1 time																																			

9.5.16 GPIO Port pull-up/pull-down register(GPIOx_PUPDR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_PUPDR	0x3C	RW	0x0000_0000	Pull up/drop down register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PxPUPD7[1:0]		PxPUPD6[1:0]		PxPUPD5[1:0]		PxPUPD4[1:0]	
7	6	5	4	3	2	1	0
PxPUPD3[1:0]		PxPUPD2[1:0]		PxPUPD1[1:0]		PxPUPD0[1:0]	

GPIO Port pull-up/pull-down register(GPIOx_PUPDR)(x = A..D)

Bit	Access	Description
[31:16]	-	Reserved
[15: 14]	RW	PxPUPD7[1:0]: Port Px7 pull-up or pull-down configuration control 00: Pull up, do not pull down 01: Enable the pull-up function 10: Enable the drop-down function 11: Reserve
[13: 12]	RW	PxPUPD6[1:0]: Port Px7 pull-up or pull-down configuration control 00: Pull up, do not pull down 01: Enable the pull-up function 10: Enable the drop-down function 11: Reserve
[11: 10]	RW	PxPUPD5[1:0]: Port Px7 pull-up or pull-down configuration control 00: Pull up, do not pull down 01: Enable the pull-up function 10: Enable the drop-down function 11: Reserve
[9: 8]	RW	PxPUPD4[1:0]: Port Px7 pull-up or pull-down configuration control 00: Pull up, do not pull down 01: Enable the pull-up function 10: Enable the drop-down function 11: Reserve
[7: 6]	RW	PxPUPD3[1:0]: Port Px7 pull-up or pull-down configuration control 00: Pull up, do not pull down 01: Enable the pull-up function 10: Enable the drop-down function 11: Reserve
[5: 4]	RW	PxPUPD2[1:0]: Port Px7 pull-up or pull-down configuration control 00: Pull up, do not pull down 01: Enable the pull-up function 10: Enable the drop-down function 11: Reserve

[3: 2]	RW	PxPUPD1[1:0]: Port Px7 pull-up or pull-down configuration control 00: Pull up, do not pull down 01: Enable the pull-up function 10: Enable the drop-down function 11: Reserve
[1: 0]	RW	PxPUPD0[1:0]: Port Px7 pull-up or pull-down configuration control 00: Pull up, do not pull down 01: Enable the pull-up function 10: Enable the drop-down function 11: Reserve

9.5.17 GPIO Port voltage conversion rate configuration(GPIOx_SLEWCR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_SLEWCR	0x40	RW	0x0000_0000	Voltage conversion rate control

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxSR7	PxSR6	PxSR5	PxSR4	PxSR3	PxSR2	PxSR1	PxSR0

GPIO Port voltage conversion rate configuration(GPIOx_SLEWCR)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	RW	PxSR7: Port Px7 voltage conversion rate configuration control 0: high voltage conversion 1: Low voltage conversion
[6]	RW	PxSR6: Port Px6 voltage conversion rate configuration control 0: high voltage conversion 1: Low voltage conversion
[5]	RW	PxSR5: Port Px5 voltage conversion rate configuration control 0: high voltage conversion 1: Low voltage conversion
[4]	RW	PxSR4: Port Px4 voltage conversion rate configuration control 0: high voltage conversion 1: Low voltage conversion
[3]	RW	PxSR3: Port Px3 voltage conversion rate configuration control 0: high voltage conversion 1: Low voltage conversion
[2]	RW	PxSR2: Port Px2 voltage conversion rate configuration control 0: high voltage conversion 1: Low voltage conversion

[1]	RW	PxSR1: Port Px1 voltage conversion rate configuration control 0: high voltage conversion 1: Low voltage conversion
[0]	RW	PxSR0: Port Px0 voltage conversion rate configuration control 0: high voltage conversion 1: Low voltage conversion

9.5.18 GPIO Port drive intensity configuration register(GPIOx_DRVCR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_DRVCR	0x44	RW	0x0000_0000	Drive strength configuration

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PxDRV7	PxDRV6	PxDRV5	PxDRV4	PxDRV3	PxDRV2	PxDRV1	PxDRV0

GPIO Port drive intensity configuration register(GPIOx_DRVCR)(x = A..D)

Bit	Access	Description
[31:8]	-	Reserved
[7]	RW	PxDRV7: Port Px7 drive strength selection control 0: high drive 1: Low drive
[6]	RW	PxDRV6: Port Px6 drive strength selection control 0: high drive 1: Low drive
[5]	RW	PxDRV5: Port Px5 drive strength selection control 0: high drive 1: Low drive
[4]	RW	PxDRV4: Port Px4 drive strength selection control 0: high drive 1: Low drive
[3]	RW	PxDRV3: Port Px3 drive strength selection control 0: high drive 1: Low drive
[2]	RW	PxDRV2: Port Px2 drive strength selection control 0: high drive 1: Low drive
[1]	RW	PxDRV1: Port Px1 drive strength selection control 0: high drive 1: Low drive
[0]	RW	PxDRV0: Port Px0 drive strength selection control 0: high drive 1: Low drive

9.5.19 GPIO Port alternate function register(GPIOx_AFR)(x = A..D)

Register	Address offset	Access	Reset value	Description
GPIOx_AFR	0x48	RW	0x0000_0000	alternate function register

31	30	29	28	27	26	25	24
PxAFR7[3:0]				PxAFR6[3:0]			
23	22	21	20	19	18	17	16
PxAFR5[3:0]				PxAFR4[3:0]			
15	14	13	12	11	10	9	8
PxAFR3[3:0]				PxAFR2[3:0]			
7	6	5	4	3	2	1	0
PxAFR1[3:0]				PxAFR0[3:0]			

GPIOA_AFR (MG32L003 F8)

Bit	Access	Description
[31:16]	-	Reserved
[15:12]	RW	PAAFR3: Select the port PA3 function 0000: PA3 0001: TIM1_CH3N 0010: PCA_CH2 0011: SPI_NSS 0100: RTC_1HZ 0101: LPUART_RXD 0110: PCA_ECI 0111: VCMP0_OUT 1000: TIM2_CH3 1001 to 1111: Reserved
[11:8]	RW	PAAFR2: Select the port PA2 function 0000: PA2 0001: TIM1_CH3 0010: Reserved 0011: SPI_NSS 0100: I2C_SCL 0101: UART1_TXD 0110: TIM10_TOGN 0111: UART2_TXD 1000: TIM2_CH2 1111: OSC_OUT Other: reserved

[7:4]	RW	PAAFR1: Select the port PA1 function 0000: PA1 0001: TIM1_CH2N 0010: Reserved 0011: SPI_CLK 0100: I2C_SDA 0101: UART1_RXD 0110: TIM10_TOG 0111: UART2_RXD 1111: OSC_IN Other: reserved
[3:0]	-	Reserved

GPIOA_AFR (MG32L003 K8)

Bit	Access	Description
[31:20]	-	Reserved
[19:16]	RW	PAAFR4: Select the port PA4 function 0000: PA4 0001: TIM1_CH2N 0010: Reserved 0011: SPI_Clk 0100: I2C_SDA 0101: UART1_RXD 0110: TIM10_TOG 0111: UART2_RXD 1111: AIN14 others: Reserved
[15:12]	RW	PAAFR3: Select the port PA3 function 0000: PA3 0001: TIM1_CH3N 0010: PCA_CH2 0011: SPI_NSS 0100: RTC_1HZ 0101: LPUART_RXD 0110: PCA_ECI 0111: VCMP0_OUT 1000: TIM2_CH3 1001: UART2_TXD Others: Reserved

[11:8]	RW	PAAFR2: Select the port PA2 function 0000: PA2 0001: TIM1_CH3 0010: Reserved 0011: SPI_NSS 0100: I2C_SCL 0101: UART1_TXD 0110: TIM10_TOGN 0111: UART2_TXD 1000: TIM2_CH2 1111: OSC_OUT Other: reserved
[7:4]	RW	PAAFR1: Select the port PA1 function 0000: PA1 0001: TIM1_CH2N 0010: Reserved 0011: SPI_CLK 0100: I2C_SDA 0101: UART1_RXD 0110: TIM10_TOG 0111: UART2_RXD 1111: OSC_IN Other: reserved
[3:0]	-	Reserved

GPIOB_AFR (MG32L003 F8)

Bit	Access	Description
[31:24]	-	Reserved
[23:20]	RW	PBAFR5: Select the port PB5 function 0000: PB5 0001: TIM1_BKIN 0010: PCA_CH4 0011: SPI_CLK 0100: I2C_SDA 0101: UART1_RXD 0110: TIM11_TOG 0111: LVD_OUT 1000: TIM2_CH1 1111: X32K_IN Other: reserved

[19:16]	RW	PBAFR4: Select the port PB4 function 0000: PB4 0001: LPTIM_GATE 0010: PCA_ECI 0011: SPI_NSS 0100: I2C_SCL 0101: UART1_TXD 0110: TIM11_TOGN 1111: X32K_OUT Other: reserved
[15:0]	-	Reserved

GPIOB_AFR (MG32L003 K8)

Bit	Access	Description
[31:28]	RW	PBAFR7: Select the port PB7 function 0000: PB7 0001: TIM1_CH3N 0010: PCA_CH2 0011: SPI_NSS 0100: RTC_1HZ 0101: LAURT_RXD 0110: PCA_ECI 0111: VCMP0_OUT 1000: TIM2_CH3 1001: UART2_TXD 1010 to 1110 : reserved 1111: AIN8
[27:24]	RW	PBAFR6: Select the port PB6 function 0000: PB6 0001: TIM1_CH1N 0010: PCA_CH4 0011: SPI_MISO 0100: I2C_SCL 0101: UART2_TXD 0110: TIM10_GATE 0111: UART1_TXD 1000: TIM2_CH4 1001 to 1110: reserved 1111: AIN9

[23:20]	RW	<p>PBAFR5: Select the port PB5 function</p> <p>0000: PB5 0001: TIM1_BKIN 0010: PCA_CH4 0011: SPI_CLK 0100: I2C_SDA 0101: UART1_RXD 0110: TIM11_TOG 0111: LVD_OUT 1000: TIM2_CH1 1111: X32K_IN Other: reserved</p>
[19:16]	RW	<p>PBAFR4: Select the port PB4 function</p> <p>0000: PB4 0001: LPTIM_GATE 0010: PCA_ECI 0011: SPI_NSS 0100: I2C_SCL 0101: UART1_TXD 0110: TIM11_TOGN 1111: X32K_OUT Other: reserved</p>
[15:12]	RW	<p>PBAFR3: Select the port PB3 function</p> <p>0000: PB3 0001: TIM1_CH2 0010: PCA_CH3 0011: SPI_MOSI 0100: I2C_SDA 0101: UART2_RXD 0110: TIM2_CH2 1001 to 1110: reserved 1111: AIN10</p>
[11:8]	RW	<p>PBAFR2: Select the port PB2 function</p> <p>0000: PB2 0001: TIM1_CH3N 0010: PCA_CH1 0011: SPI_MOSI 0100: HSE_OUT 0101: UART1_RXD 0110: LPTIM_TOGN 0111: reserved 1000: TIM2_CH2 1001 to 1110: reserved 1111: AIN11</p>

[7:4]	RW	<p>PBAFR1: Select the port PB1 function 0000: PB1 0001: TIM1_CH1 0010: PCA_CH0 0011: RTC_1HZ 0100: TIM10_TOG 0101: UART1_TXD 0110: TIM10_EXT 0111: BEEP 1000: TIM2_CH1 1001: TIM1_CH2N 1010 to 1110: reserved 1111: AIN12</p>
[3:0]	RW	<p>PBAFR0: Select the port PB0 function 0000: PB0 0001: TIM1_CH3 0010: TIM1_CH1N 0011: reserved 0100: I2C_SDA 0101: UART2_TXD 0110: PCA_CH1 0111: 1-WIRE 1000: TIM2_CH3 1001 to 1110: reserved 1111: AIN13</p>

GPIOC_AFR (MG32L003 F8)

Bit	Access	Description
[31:28]	RW	<p>PCAFR7: Select the port PC7 function 0000: PC7 0001: TIM1_CH2 0010: PCA_CH4 0011: SPI_MISO 0100: reserved 0101: UART2_RXD 0110: LSI_OUT 0111: LSE_OUT 1000: reserved 1001 to 1111: Reserved</p>

[27:24]	RW	<p>PCAFR6: Select the port PC6 function</p> <p>0000: PC6 0001: TIM1_CH1 0010: PCA_CH3 0011: SPI_MOSI 0100: reserved 0101: LPUART_RXD 0110: TIM11_EXT 0111: CLK_MCO 1000: TIM2_CH4 1001 to 1110: Reserved 1111: AIN0</p>
[23:20]	RW	<p>PCAFR5: Select the port PC5 function</p> <p>0000: PC5 0001: TIM1_BKIN 0010: PCA_CH0 0011: SPI_CLK 0100: reserved 0101: LPUART_TXD 0110: TIM11_GATE 0111: LVD_OUT 1000: TIM2_CH1 1001 to 1110: Reserved 1111: VCMPIN1</p>
[19:16]	RW	<p>PCAFR4: Select the port PC4 function</p> <p>0000: PC4 0001: TIM1_CH4 0010: TIM1_CH2N 0011: Reserved 0100: I2C_SCL 0101: UART2_RXD 0110: PCA_CH0 0111: CLK_MCO 1000: TIM2_CH4 1001 to 1110: Reserved 1111: AIN2</p>

[15:12]	RW	PCAFR3: Select the port PC3 function 0000: PC3 0001: TIM1_CH3 0010: TIM1_CH1N 0011: Reserved 0100: I2C_SDA 0101: UART2_TXD 0110: PCA_CH1 0111: 1-WIRE 1000: TIM2_CH3 1001 to 1110: Reserved 1111: AIN1
[11:0]	-	Reserved

GPIOC_AFR (MG32L003 K8)

Bit	Access	Description
[31:28]	RW	PCAFR7: Select the port PC7 function 0000: PC7 0001: TIM1_CH2 0010: PCA_CH4 0011: SPI_MISO 0100: reserved 0101: UART2_RXD 0110: LSI_OUT 0111: LSE_OUT 1000: reserved 1001 to 1111: Reserved
[27:24]	RW	PCAFR6: Select the port PC6 function 0000: PC6 0001: TIM1_CH1 0010: PCA_CH3 0011: SPI_MOSI 0100: reserved 0101: LPUART_RXD 0110: TIM11_EXT 0111: CLK_MCO 1000: TIM2_CH4 1001 to 1110: Reserved 1111: AIN0

[23:20]	RW	<p>PCAFR5: Select the port PC5 function 0000: PC5 0001: TIM1_BKIN 0010: PCA_CH0 0011: SPI_CLK 0100: reserved 0101: LPUART_TXD 0110: TIM11_GATE 0111: LVD_OUT 1000: TIM2_CH1 1001 to 1110: Reserved 1111: VCMPIN1</p>
[19:16]	RW	<p>PCAFR4: Select the port PC4 function 0000: PC4 0001: TIM1_CH4 0010: TIM1_CH2N 0011: Reserved 0100: I2C_SCL 0101: UART2_RXD 0110: PCA_CH0 0111: CLK_MCO 1000: TIM2_CH4 1001 to 1110: Reserved 1111: AIN2</p>
[15:12]	RW	<p>PCAFR3: Select the port PC3 function 0000: PC3 0001: TIM1_CH3 0010: TIM1_CH1N 0011: Reserved 0100: I2C_SDA 0101: UART2_TXD 0110: PCA_CH1 0111:1-WIRE 1000: TIM2_CH3 1001 to 1110: Reserved 1111: AIN1</p>

[11: 8]	RW	<p>PCAFR2: Select the port PC2 function 0000: PC2 0001: TIM1_CH2 0010: PCA_CH2 0011: SPI_MISO 0100: RTC_1HZ 0101: LPUART_TXD 0110: LPTIM_TOG 0111: 1-WIRE 1000: TIM2_CH3 1001: TIM1_CH3N 1010 to 1111: Reserved</p>
[7: 4]	RW	<p>PCAFR1: Select the port PC1 function 0000: PC1 0001: TIM1_CH1 0010: PCA_CH3 0011: SPI_MOSI 0100: Reserved 0101: LPUART_RXD 0110: TIM11_EXT 0111: CLK_MCO 1000: TIM2_CH4 1001: TIM1_CH2N 1010 to 1111: Reserved</p>
[3: 0]	RW	<p>PCAFR0: Select the port PC0 function 0000: PC0 0001: TIM1_CH3 0010: Reserved 0011: SPI_NSS 0100: I2C_SCL 0101: UART1_TXD 0110: TIM10_TOGN 0111: UART2_TXD 1000: TIM2_CH2 1001: TIM1_CH1N 1111: OSC_OUT other: Reserved</p>

GPIOD_AFR (MG32L003 F8)

Bit	Access	Description
[31:28]	-	Reserved

[27:24]	RW	<p>PDAFR6: Port PD6 function Select</p> <p>0000: PD6 0001: TIM1_CH2 0010: PCA_CH3 0011: SPI_MOSI 0100: I2C_SDA 0101: UART2_RXD 0110: LPTIM_EXT 0111: UART1_RXD 1000: TIM2_CH2 1001 to 1110: Reserved 1111: AIN6</p>
[23:20]	RW	<p>PDAFR5: Port PD5 function selection</p> <p>0000: PD5 0001: TIM1_CH1N 0010: PCA_CH4 0011: SPI_MISO 0100: I2C_SCL 0101: UART2_TXD 0110: TIM10_GATE 0111: UART1_TXD 1000: TIM2_CH4 1001 to 1110: Reserved 1111: AIN5</p>
[19:16]	RW	<p>PDAFR4: Port PD4 function Select</p> <p>0000: PD4 0001: TIM1_CH1 0010: PCA_CH0 0011: RTC_1HZ 0100: TIM10_TOG 0101: UART1_TXD 0110: TIM10_EXT 0111: BEEP 1000: TIM2_CH1 1001 to 1110: Reserved 1111: VCMPIN2</p>

[15:12]	RW	<p>PDAFR3: Port PD3 function Select 0000: PD3 0001: TIM1_CH3N 0010: PCA_CH1 0011: SPI_MOSI 0100: HSE_OUT 0101: UART1_RXD 0110: LPTIM_TOGN 0111: reserved 1000: TIM2_CH2 1001 to 1110: Reserved 1111: AIN4</p>
[11:8]	RW	<p>PDAFR2: Port PD2 function Select 0000: PD2 0001: TIM1_CH2 0010: PCA_CH2 0011: SPI_MISO 0100: RTC_1HZ 0101: LPUART_TXD 0110: LPTIM_TOG 0111:1-WIRE 1000: TIM2_CH3 1001 to 1110: Reserved 1111: VCMPIN0/AIN3</p>
[7:4]	RW	<p>PDAFR1: Port PD1 function Select 0000: PD1 0001: Reserved 0010: PCA_ECI 0011: Reserved 0100: reserved 0101: UART2_TXD 0110: HSI_OUT 0111: VCMP0_OUT 1000: reserved 1001 to 1110: Reserved 1111: reserved</p>
[3:0]	-	Reserved

GPIOD_AFR (MG32L003 K8)

Bit	Access	Description
-----	--------	-------------

[31:28]	RW	<p>PDAFR7: Port PD7 function Select</p> <p>0000: PD7 0001: TIM1_CH4 0010: TIM1_CH2N 0011: Reserved 0100: I2C_SCL 0101: UART2_RXD 0110: PCA_CH0 0111: CLK_MCO 1000: TIM2_CH4 1001 to 1110: Reserved 1111: AIN2</p>
[27:24]	RW	<p>PDAFR6: Port PD6 function Select</p> <p>0000: PD6 0001: TIM1_CH2 0010: PCA_CH3 0011: SPI_MOSI 0100: I2C_SDA 0101: UART2_RXD 0110: LPTIM_EXT 0111: UART1_RXD 1000: TIM2_CH2 1001 to 1110: Reserved 1111: AIN6</p>
[23:20]	RW	<p>PDAFR5: Port PD5 function selection</p> <p>0000: PD5 0001: TIM1_CH1N 0010: PCA_CH4 0011: SPI_MISO 0100: I2C_SCL 0101: UART2_TXD 0110: TIM10_GATE 0111: UART1_TXD 1000: TIM2_CH4 1001 to 1110: Reserved 1111: AIN5</p>

[19:16]	RW	<p>PDAFR4: Port PD4 function Select</p> <p>0000: PD4 0001: TIM1_CH1 0010: PCA_CH0 0011: RTC_1HZ 0100: TIM10_TOG 0101: UART1_TXD 0110: TIM10_EXT 0111: BEEP 1000: TIM2_CH1 1001 to 1110: Reserved 1111: VCMPIN2</p>
[15:12]	RW	<p>PDAFR3: Port PD3 function Select</p> <p>0000: PD3 0001: TIM1_CH3N 0010: PCA_CH1 0011: SPI_MOSI 0100: HSE_OUT 0101: UART1_RXD 0110: LPTIM_TOGN 0111: reserved 1000: TIM2_CH2 1001 to 1110: Reserved 1111: AIN4/VCMPIN0</p>
[11:8]	RW	<p>PDAFR2: Port PD2 function Select</p> <p>0000: PD2 0001: TIM1_CH2 0010: PCA_CH2 0011: SPI_MISO 0100: RTC_1HZ 0101: LPUART_TXD 0110: LPTIM_TOG 0111:1-WIRE 1000: TIM2_CH3 1001 to 1110: Reserved 1111: VCMPIN0/AIN3</p>

[7:4]	RW	<p>PDAFR1: Port PD1 function Select</p> <p>0000: PD1 0001: Reserved 0010: PCA_ECI 0011: Reserved 0100: reserved 0101: UART2_TXD 0110: HSI_OUT 0111: VCMP0_OUT 1000: reserved 1001 to 1110: Reserved 1111: reserved</p>
[3:0]	RW	<p>PDAFR0: Port PD0 function Select</p> <p>0000: PD0 0001: TIM1_BKIN 0010: PCA_CH0 0011: SPI_CLK 0100: reserved 0101: LPUART_TXD 0110: TIM11_GATE 0111: LVD_OUT 1000: TIM2_CH1 1001 to 1111: reserved</p>

10 Advanced-control timers(TIM1)

10.1 TIM1 introduction

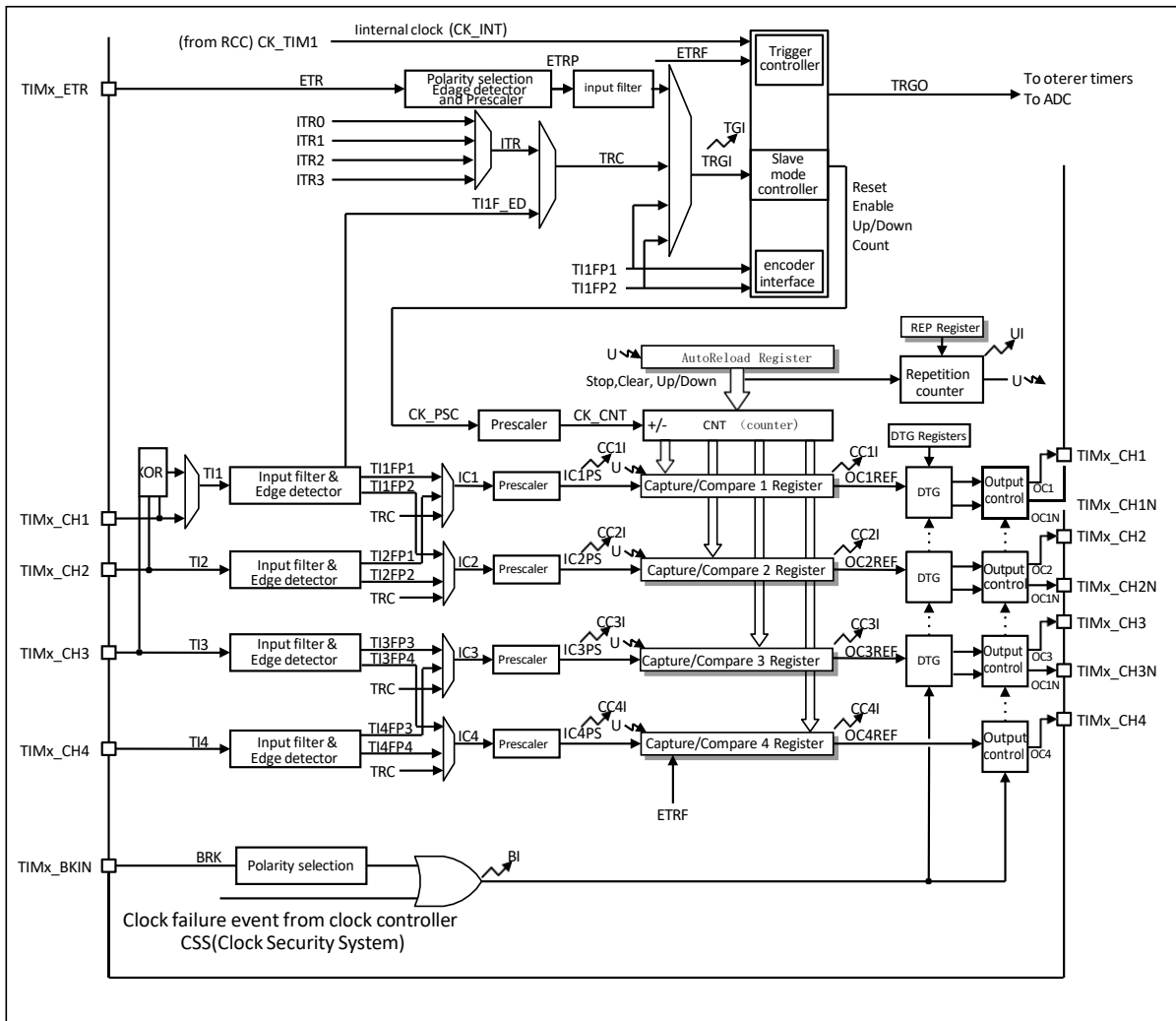
The advanced-control timers (TIM1) consist of a 16-bit auto-reload counter driven by a programmable prescaler. It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare, PWM,complementary PWM with dead-time insertion).

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




10.1.1 TIM1 main features

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

Fig 10.1-1 Advanced-control timer block diagram



NOTE :

-  Preload registers transferred to active registers on U event according to control bit
-  Event
-  Interrupt

10.1.2 TIM1 functional description

Time-base unit

The main module of the programmable advanced-control timer is a 16-bit counter with its related auto-reload register. The counter can count up, down or both up and down. The counter clock can be divided by a prescaler. The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIM1_CNT)

- Prescaler register (TIM1_PSC)
- Auto-reload register (TIM1_ARR)
- Repetition counter register (TIM1_RCR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIM1_CR1 register. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIM1_CR1 register. It can also be generated by software. The generation of the update event is described in detailed for each configuration.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIM1_CR1 register is set (refer also to the slave mode controller description to get more details on counter enabling).

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIM1_CR1 register.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIM1_PSC register).It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

Figure 10.1-2 and Figure 10.1-3give some examples of the counter behavior when the prescaler ratio is changed on the fly:

Fig 10.1-2 Counter timing diagram with prescaler division change from 1 to 2

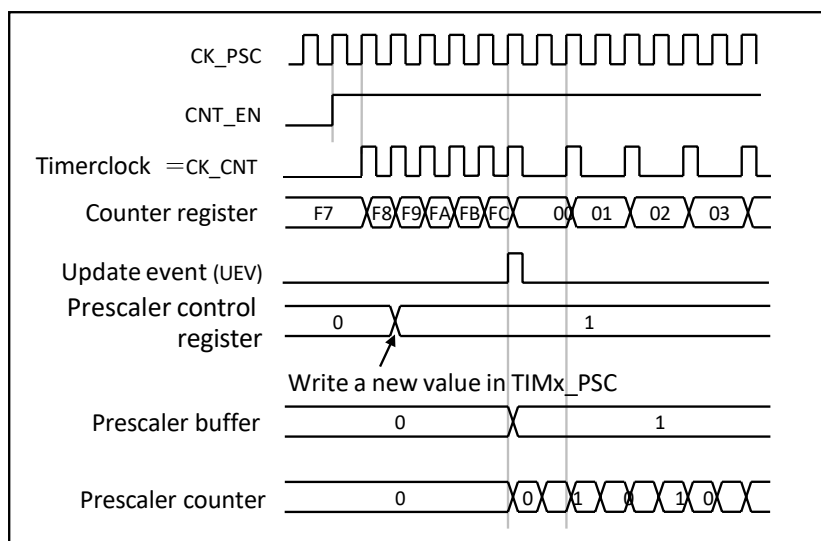
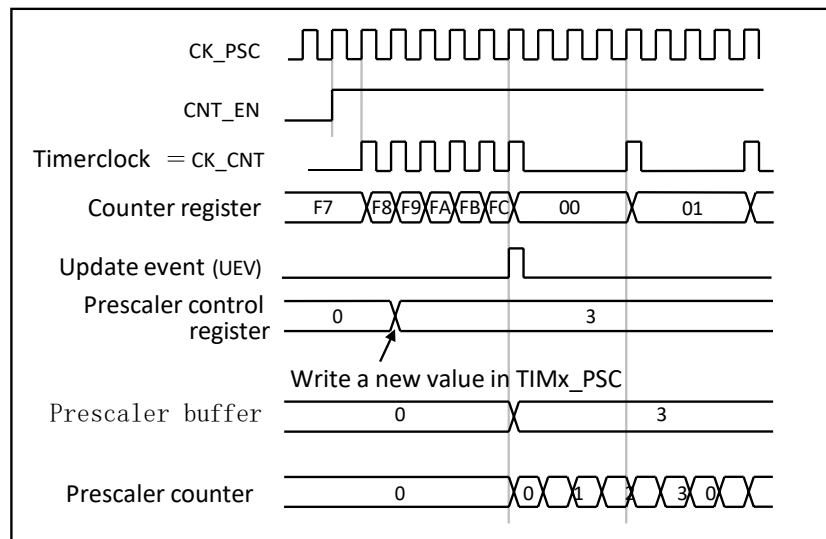


Fig 10.1-3 Counter timing diagram with prescaler division change from 1 to 4



10.1.3 Counter modes

10.1.3.1 Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the TIM1_ARR register), then restarts from 0 and generates a counter overflow event.

If the repetition counter is used, the update event (UEV) is generated after upcounting is repeated for the number of times programmed in the repetition counter register plus one (TIM1_RCR+1). Else the update event is generated at each counter overflow.

Setting the UG bit in the TIM1_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIM1_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIM1_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag. This is to avoid generating both update and capture interrupts when clearing the counter on the capture event. When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM1_SR register) is set (depending on the URS bit):

- The repetition counter is reloaded with the content of TIM1_RCR register,
- The auto-reload shadow register is updated with the preload value (TIM1_ARR),
- The buffer of the prescaler is reloaded with the preload value (content of the TIM1_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIM1_ARR=0x36

Fig 10.1-4 Counter timing diagram, internal clock divided by 1

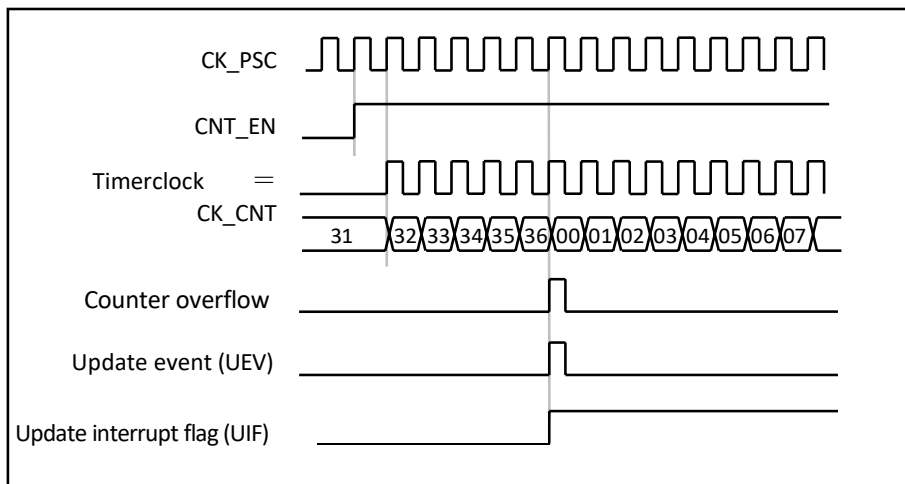


Fig 10.1-5 Counter timing diagram, internal clock divided by 2

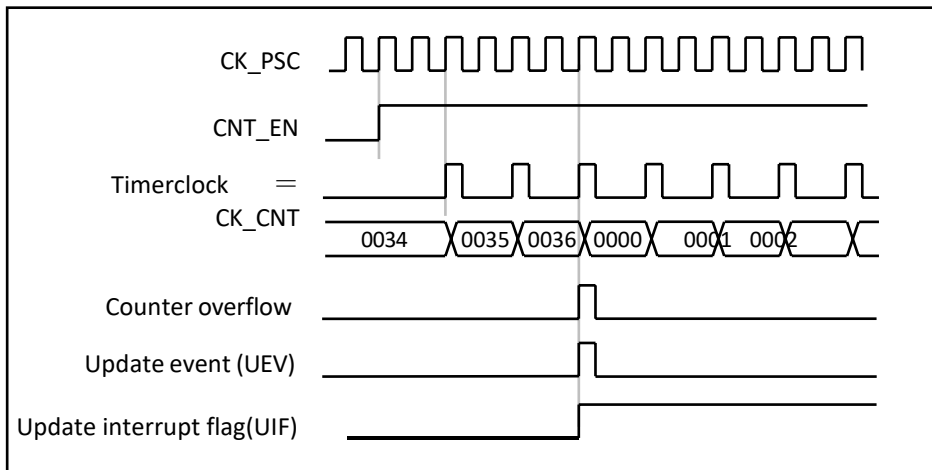


Fig 10.1-6 Counter timing diagram, internal clock divided by 4

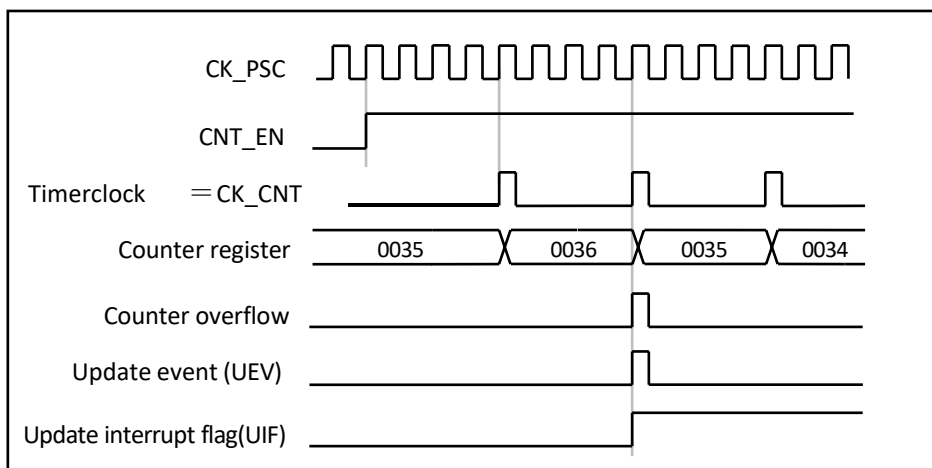


Fig 10.1-7 Counter timing diagram, internal clock divided by N

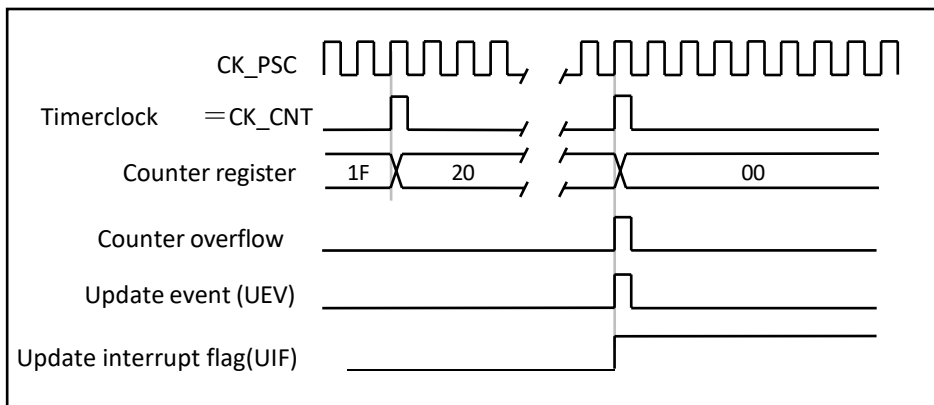


Fig 10.1-8 Counter timing diagram, update event when ARPE=0 (TIM1_ARR not preloaded)

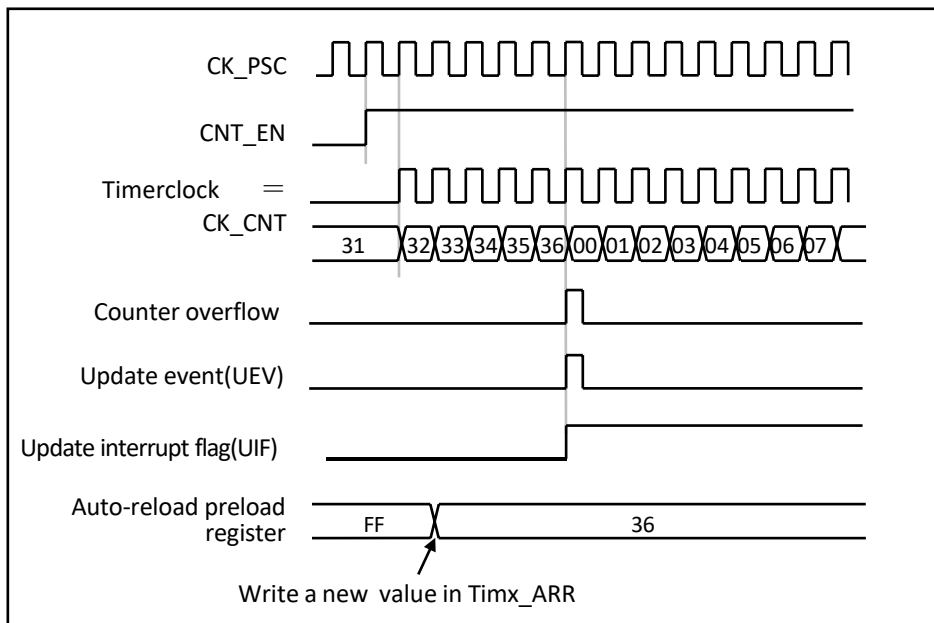
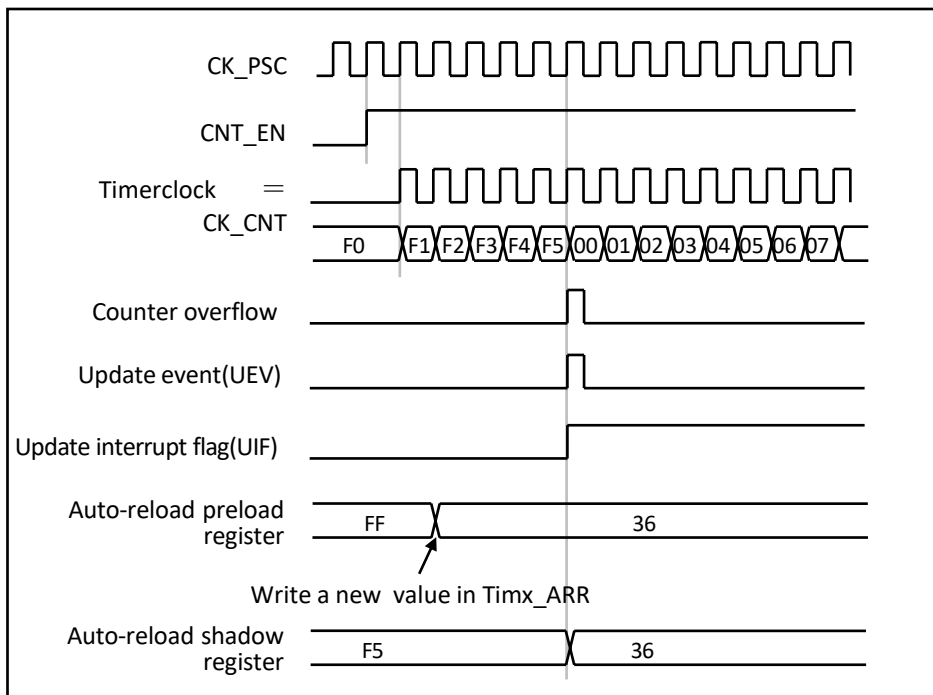


Fig 10.1-9 Counter timing diagram, update event when ARPE=1 (TIM1_ARR preloaded)



10.1.3.2 Downcounting mode

In downcounting mode, the counter counts from the auto-reload value (content of the TIM1_ARR register) down to 0, then restarts from the auto-reload value and generates a counter underflow event.

If the repetition counter is used, the update event (UEV) is generated after downcounting is repeated for the number of times programmed in the repetition counter register plus one (TIM1_RCR+1). Else the update event is generated at each counter underflow.

Setting the UG bit in the TIM1_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV update event can be disabled by software by setting the UDIS bit in TIM1_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescale rate doesn't change).

In addition, if the URS bit (update request selection) in TIM1_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag. This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM1_SR register) is set (depending on the URS bit):

- The repetition counter is reloaded with the content of TIM1_RCR register
- The buffer of the prescaler is reloaded with the preload value (content of the TIM1_PSC register)
- The auto-reload active register is updated with the preload value (content of the TIM1_ARR reg-

ister).

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

The following figures show some examples of the counter behavior for different clock frequencies when TIM1_ARR=0x36.

Fig 10.1-10 Counter timing diagram, internal clock divided by 1

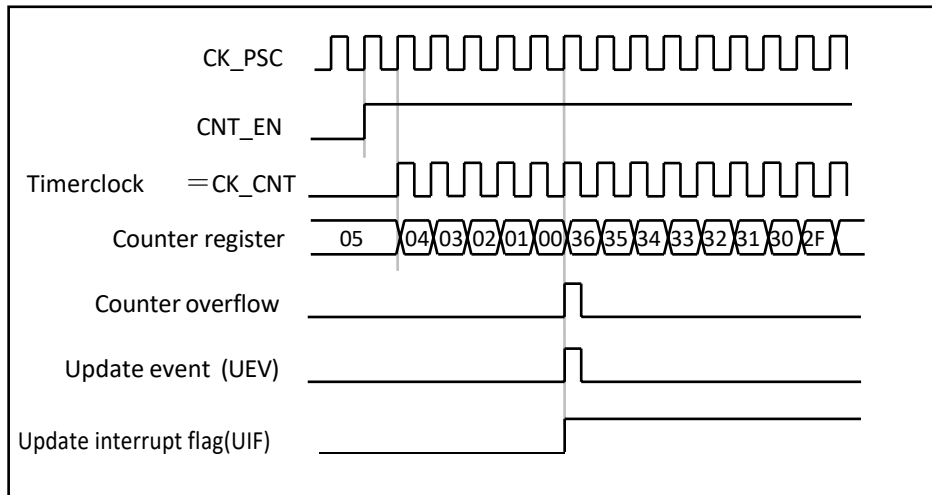


Fig 10.1-11 Counter timing diagram, internal clock divided by 2

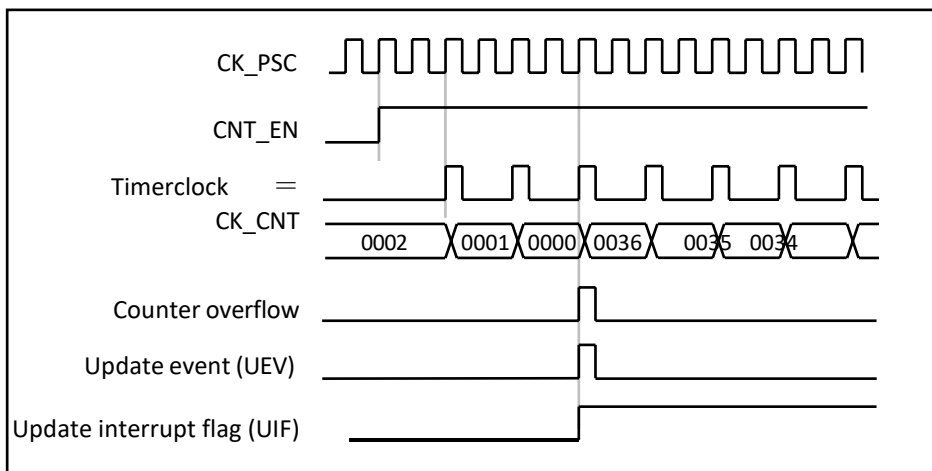


Fig 10.1-12 Counter timing diagram, internal clock divided by 4

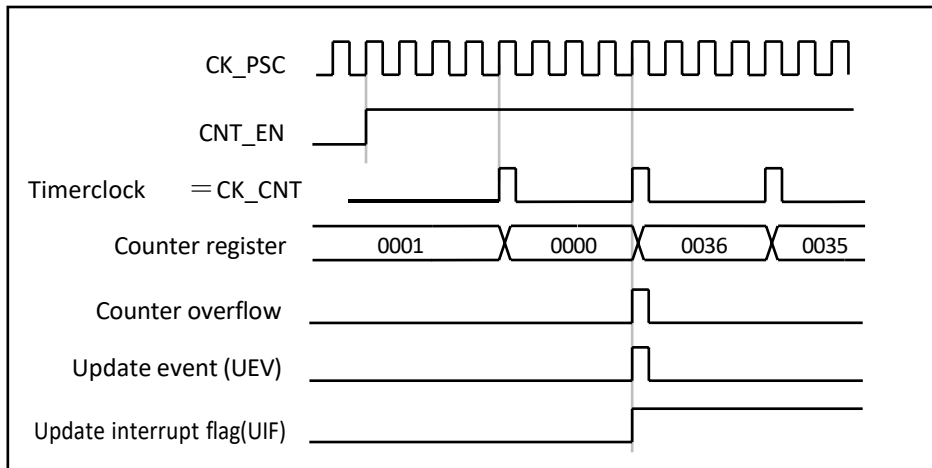


Fig 10.1-13 Counter timing diagram, internal clock divided by N

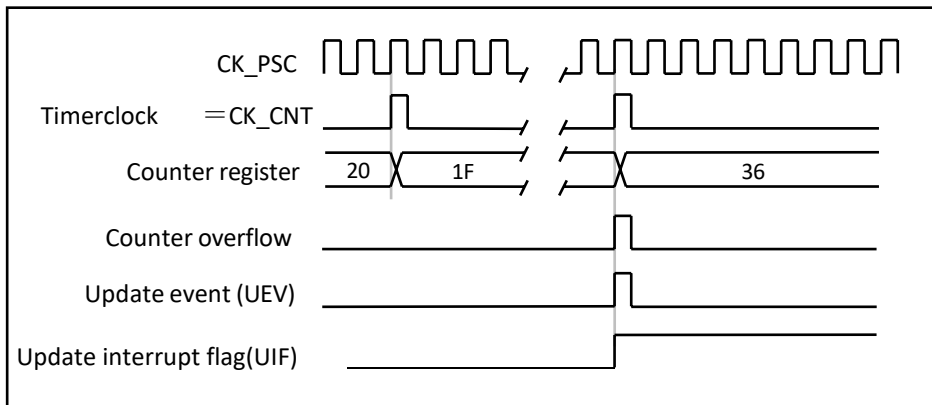
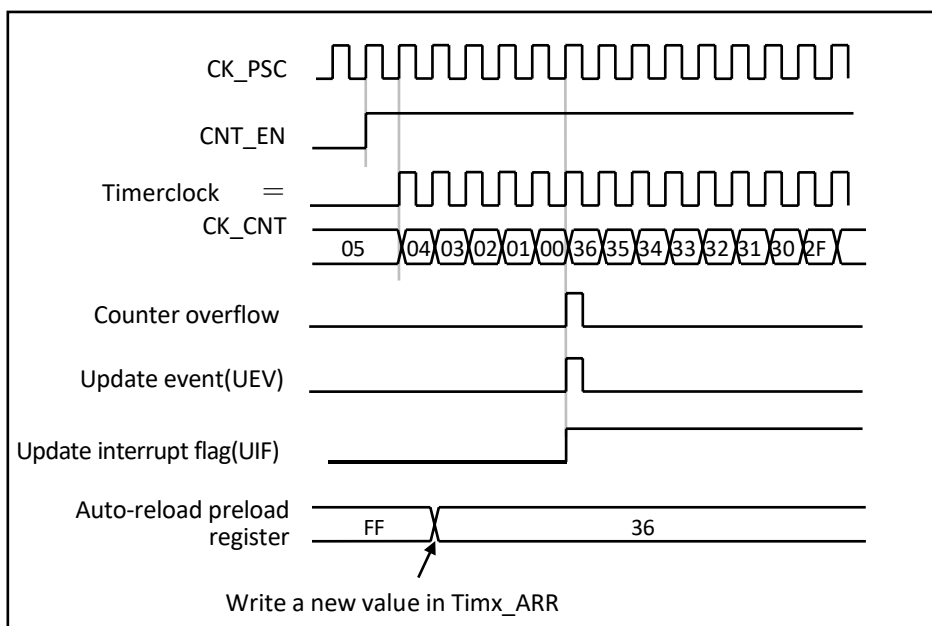


Fig 10.1-14 Counter timing diagram, update event when repetition counter is not used



10.1.3.3 Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the TIM1_ARR register) –1, generates a counter overflow event, then counts from the auto-reload value down to 1 and generates a counter underflow event. Then it restarts counting from 0.

In this mode, the DIR direction bit in the TIM1_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter. The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIM1_EGR register (by software or by using the slave mode controller) also generates an update event. In this case, the counter restarts counting from 0, as well as the counter of the prescaler.

The UEV update event can be disabled by software by setting the UDIS bit in the TIM1_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter continues counting up and down, based on the current auto-reload value.

In addition, if the URS bit (update request selection) in TIM1_CR1 register is set, setting the UG bit generates an UEV update event but without setting the UIF flag. This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM1_SR register) is set (depending on the URS bit):

- The repetition counter is reloaded with the content of TIM1_RCR register
- The buffer of the prescaler is reloaded with the preload value (content of the TIM1_PSC register)
- The auto-reload active register is updated with the preload value (content of the TIM1_ARR register).

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

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

Fig 10.1-15 Counter timing diagram, internal clock divided by 1, TIM1_ARR = 0x6

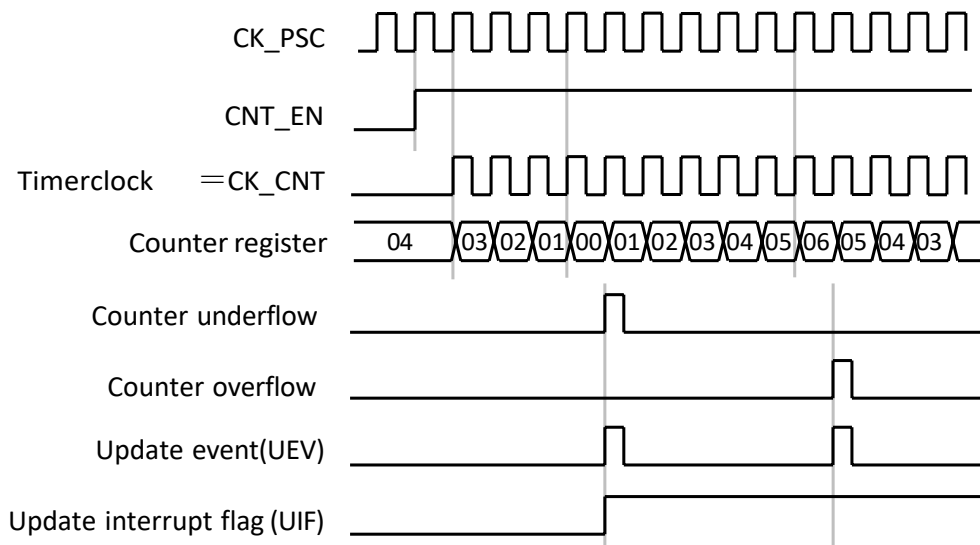


Fig 10.1-16 Counter timing diagram, internal clock divided by 2

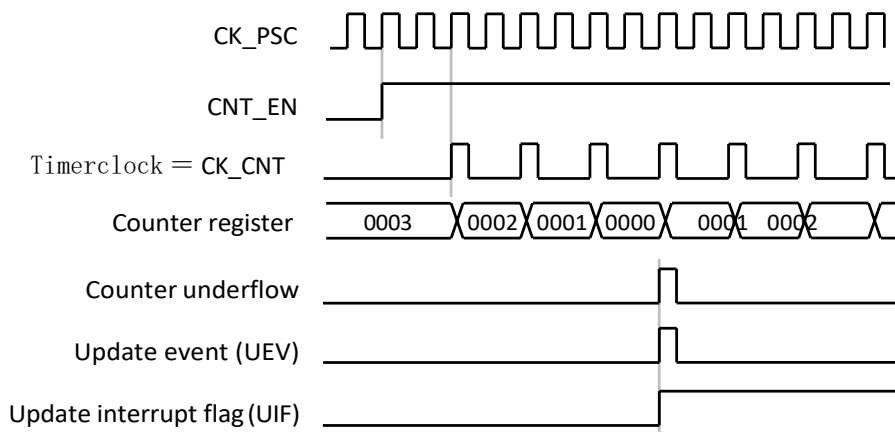
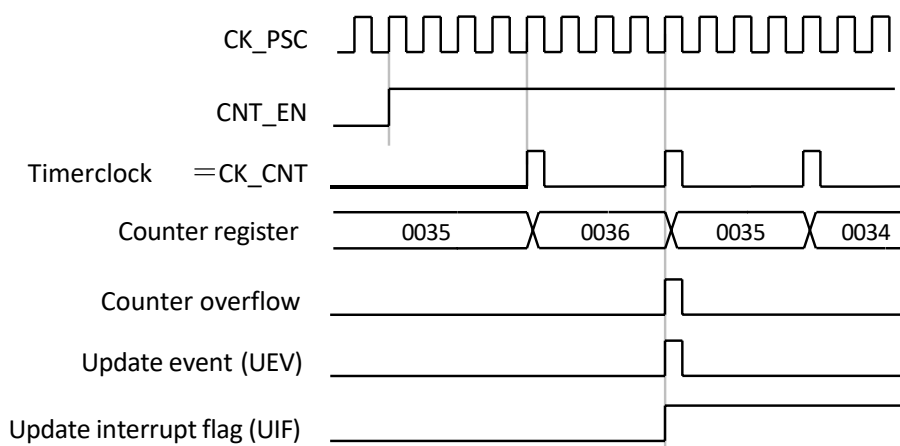


Fig 10.1-17 Counter timing diagram, internal clock divided by 4, TIM1_ARR = 0x36



Center-aligned mode 2 or 3 is used with an UIF on overflow.

Fig 10.1-18 Counter timing diagram, internal clock divided by N

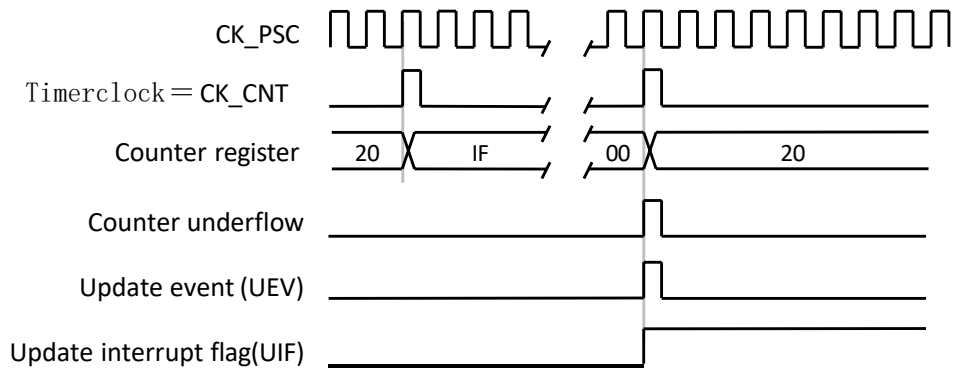


Fig 10.1-19 Counter timing diagram, update event with ARPE=1 (counter underflow)

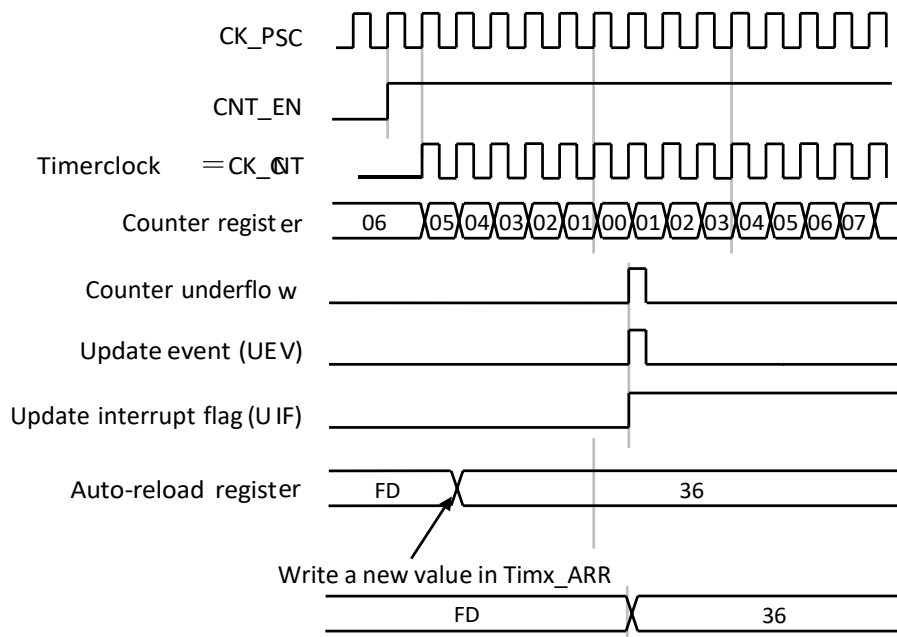
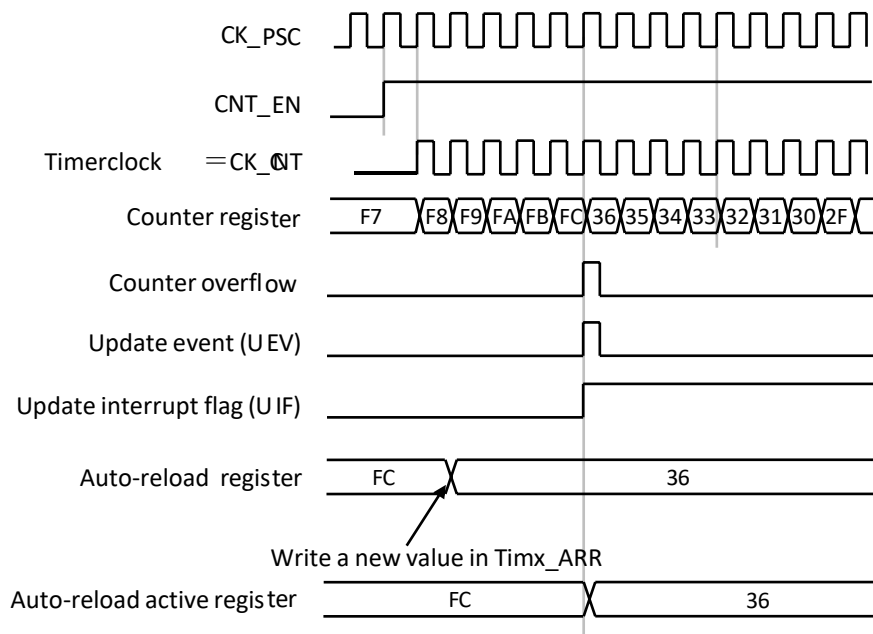


Fig 10.1-20 Counter timing diagram, Update event with ARPE=1 (counter overflow)



10.1.4 Repetition counter

It is actually generated only when the repetition counter has reached zero. This can be useful when generating PWM signals.

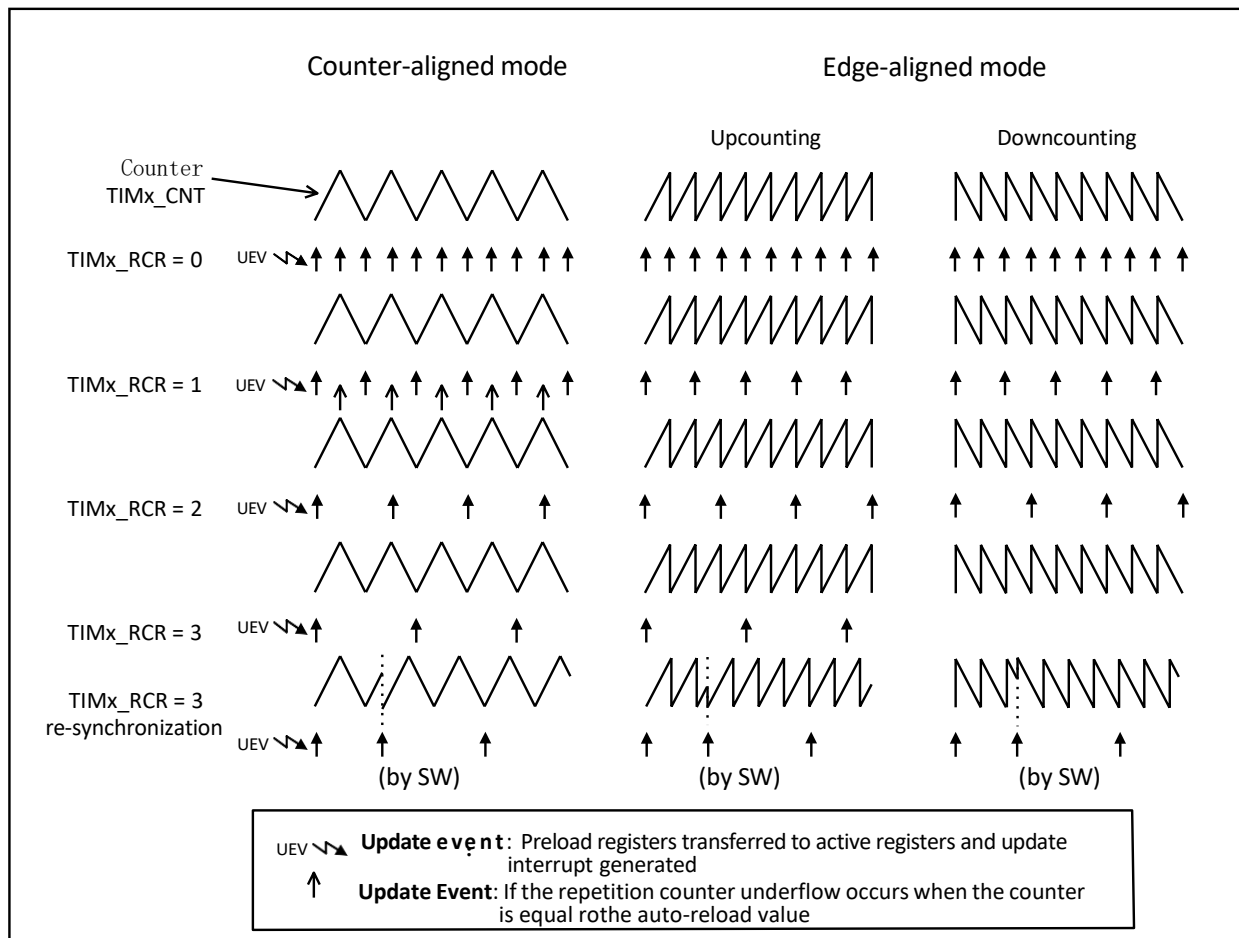
This means that data are transferred from the preload registers to the shadow registers (TIM1_ARR auto-reload register, TIM1_PSC prescaler register, but also TIM1_CCRx capture/compare registers in compare mode) every N+1 counter overflows or underflows, where N is the value in the TIM1_RCR repetition counter register.

The repetition counter is decremented:

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

The repetition counter is an auto-reload type; the repetition rate is maintained as defined by the TIM1_RCR register value (refer to Figure 10.1-21). When the update event is generated by software (by setting the UG bit in TIM1_EGR register) or by hardware through the slave mode controller, it occurs immediately whatever the value of the repetition counter is and the repetition counter is reloaded with the content of the TIM1_RCR register.

Fig 10.1-21 Update rate examples depending on mode and TIM1_RCR register settings



10.1.5 Clock selection

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

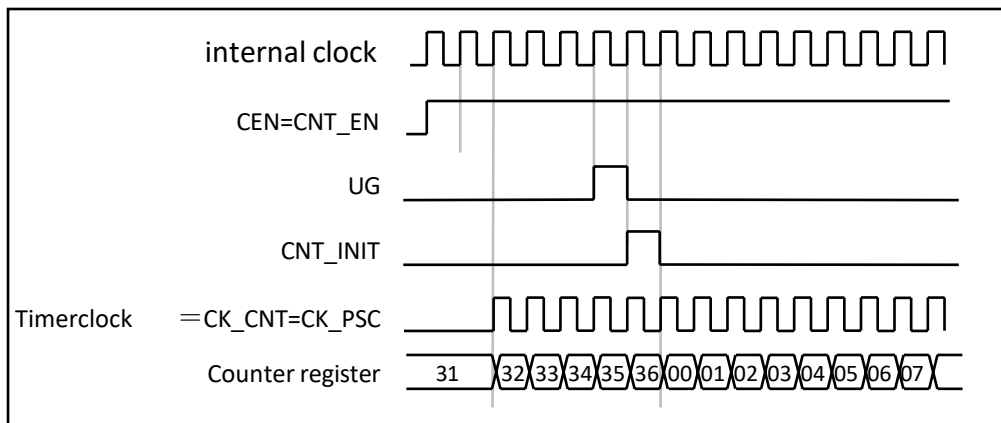
- Internal clock (CK_INT)
- External clock mode1: external input pin
- External clock mode2: external trigger input ETR
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, the user can configure Timer 1 to act as a prescaler for Timer 2.

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS=000), then the CEN, DIR (in the TIM1_CR1 register) and UG bits (in the TIM1_EGR register) are actual control bits and can be changed only by software (except UG which remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

Figure 10.1-22 shows the behavior of the control circuit and the upcounter in normal mode, without prescaler.

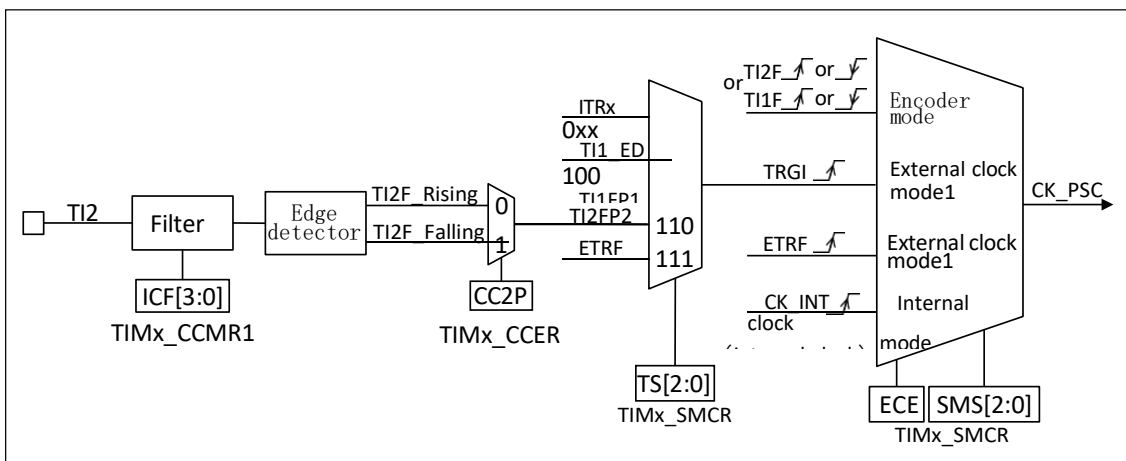
Fig 10.1-22 Control circuit in normal mode, internal clock divided by 1



External clock source mode 1

This mode is selected when SMS=111 in the TIM1_SMCR register. The counter can count at each rising or falling edge on a selected input.

Fig 10.1-23 TI2 external clock connection example



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

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = '01' in the TIM1_CCMR1 register.
2. Configure the input filter duration by writing the IC2F[3:0] bits in the TIM1_CCMR1. register (if no filter is needed, keep IC2F=0000).
3. Select rising edge polarity by writing CC2P=0 in the TIM1_CCER register.
4. Configure the timer in external clock mode 1 by writing SMS=111 in the TIM1_SMCR register.
5. Select TI2 as the trigger input source by writing TS=110 in the TIM1_SMCR register.
6. Enable the counter by writing CEN=1 in the TIM1_CR1 register.

Note:

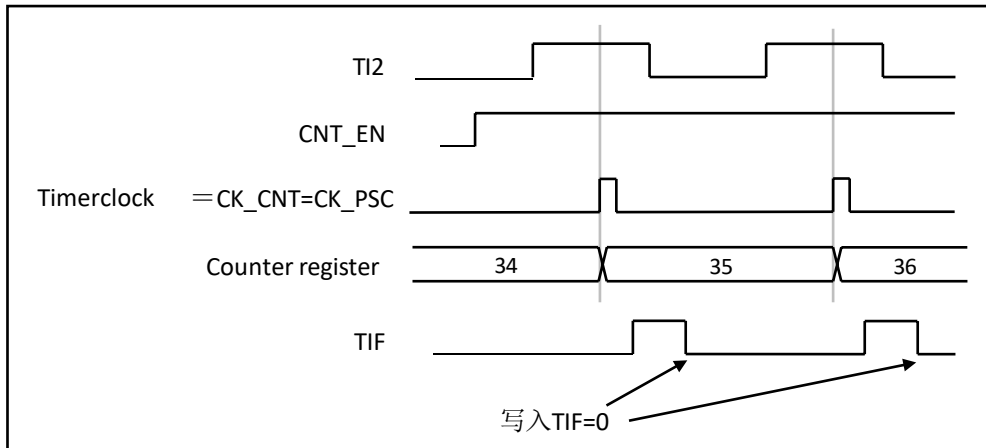
The capture prescaler is not used for triggering, so the user does not need to configure it.

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge on TI2 and the actual clock of the counter is due to the resynchronization circuit on TI2

input.

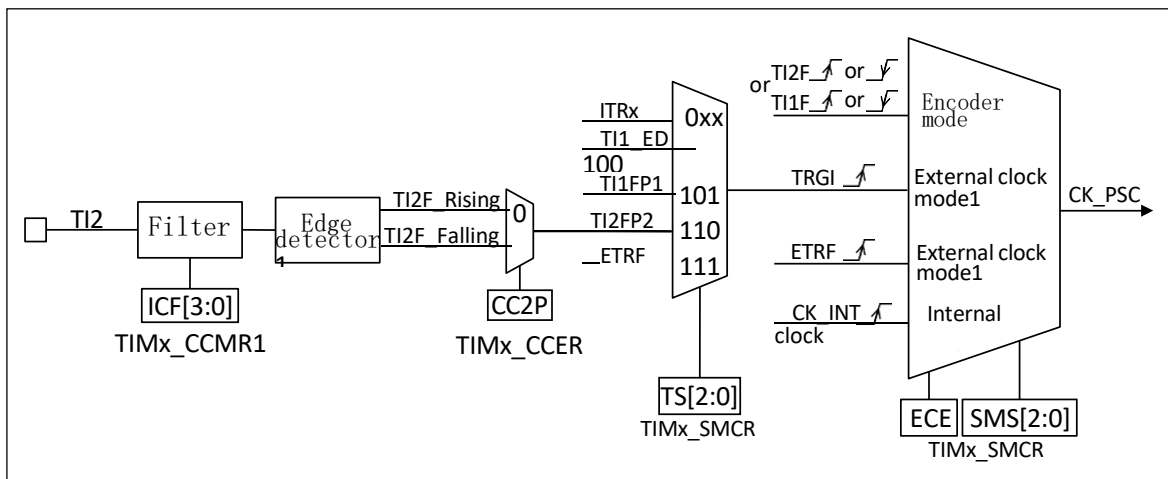
Fig 10.1-24 Control circuit in external clock mode 1



External clock source mode 2

This mode is selected by writing ECE=1 in the TIM1_SMCR register. The counter can count at each rising or falling edge on the external trigger input ETR. The following gives an overview of the external trigger input module.

Fig 10.1-25 External trigger input module



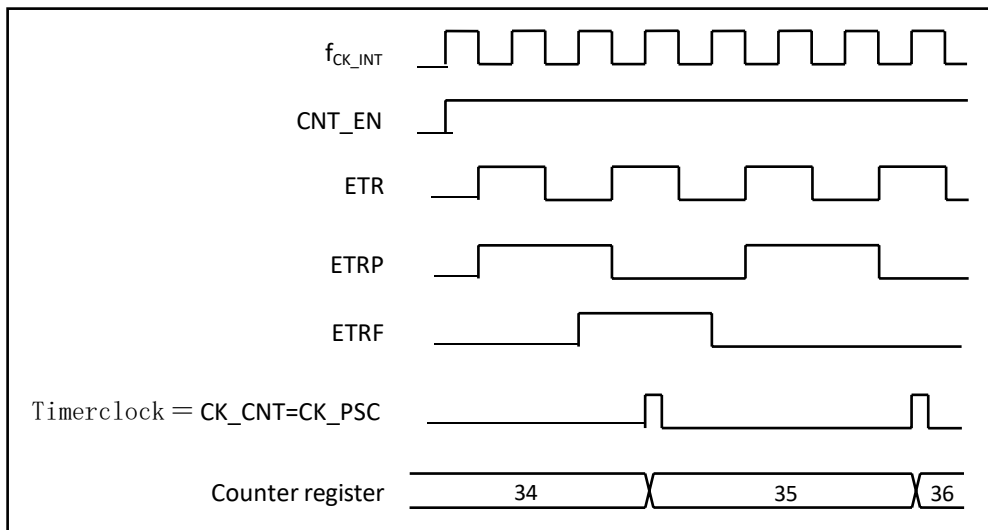
For example, to configure the upcounter to count each 2 rising edges on ETR, use the following procedure:

1. As no filter is needed in this example, write **ETF[3:0]=0000** in the **TIM1_SMCR** register.
2. Set the prescaler by writing **ETPS[1:0]=01** in the **TIM1_SMCR** register.
3. Select rising edge detection on the ETR pin by writing **ETP=0** in the **TIM1_SMCR** register.
4. Enable external clock mode 2 by writing **ECE=1** in the **TIM1_SMCR** register.
5. Enable the counter by writing **CEN=1** in the **TIM1_CR1** register.

The counter counts once each 2 ETR rising edges.

The delay between the rising edge on ETR and the actual clock of the counter is due to the resynchronization circuit on the ETRP signal.

Fig 10.1-26 Control circuit in external clock mode 2

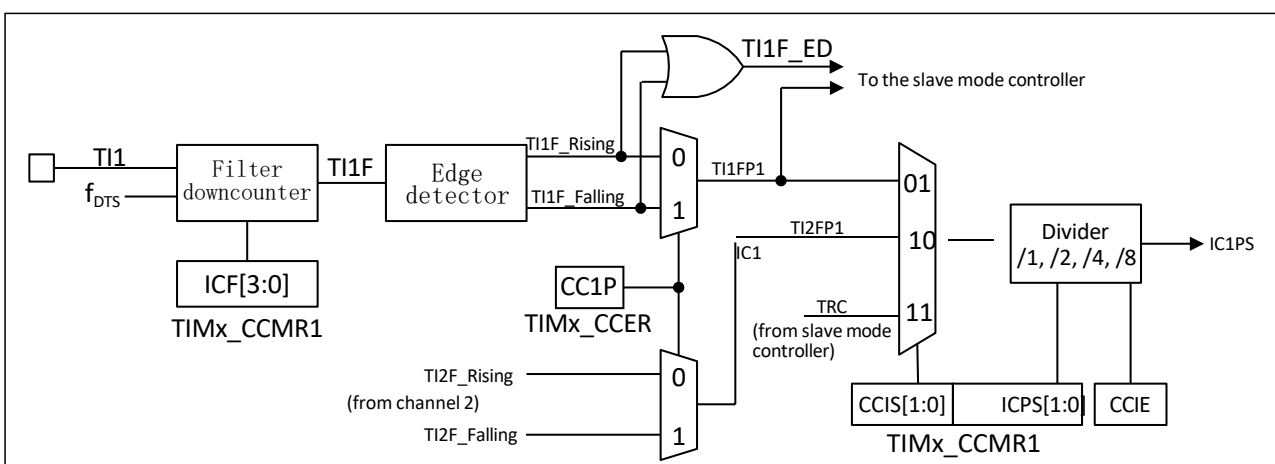


10.1.6 Capture/compare channels

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

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

Fig 10.1-27 Capture/compare channel (example: channel 1 input stage)



The output stage generates an intermediate waveform that is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

Fig 10.1-28 Capture/compare channel 1 main circuit

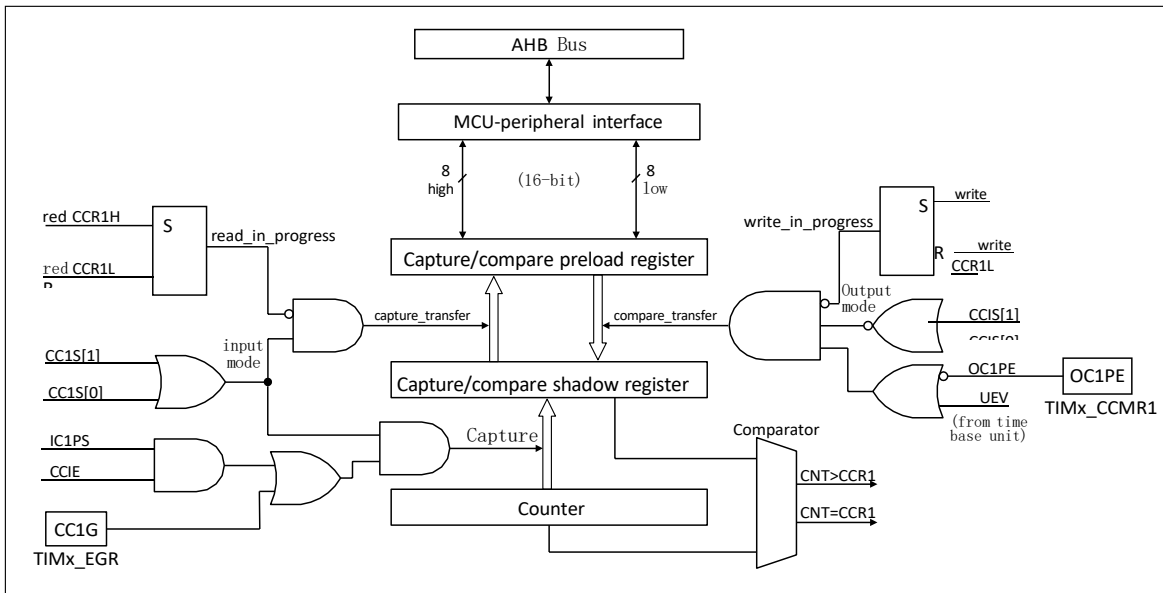


Fig 10.1-29 Output stage of capture/compare channel (channel 1 to 3)

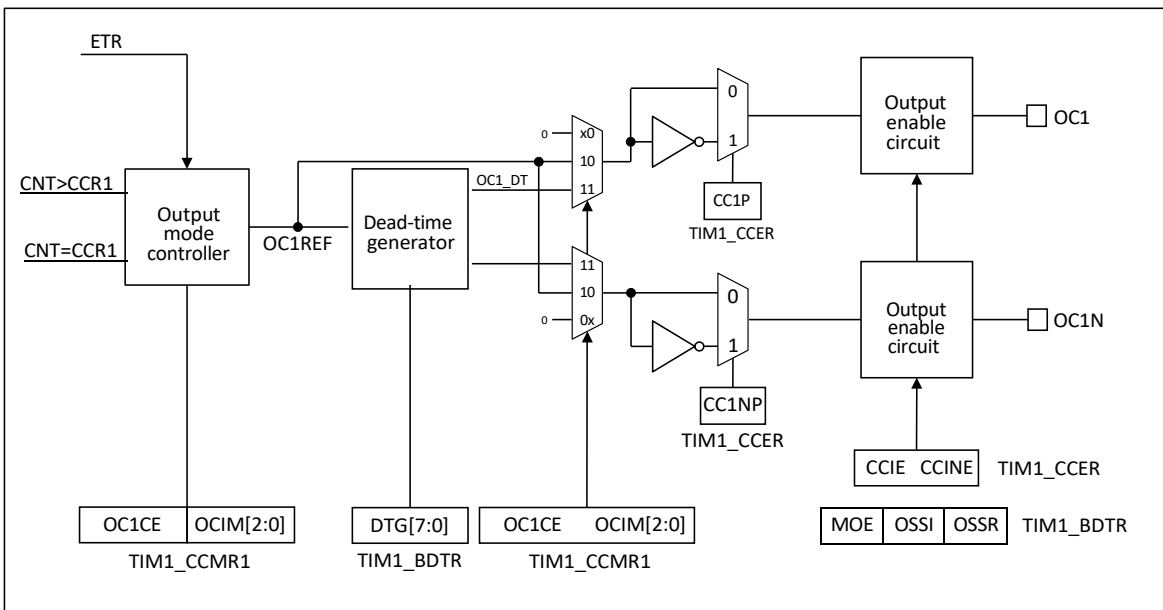
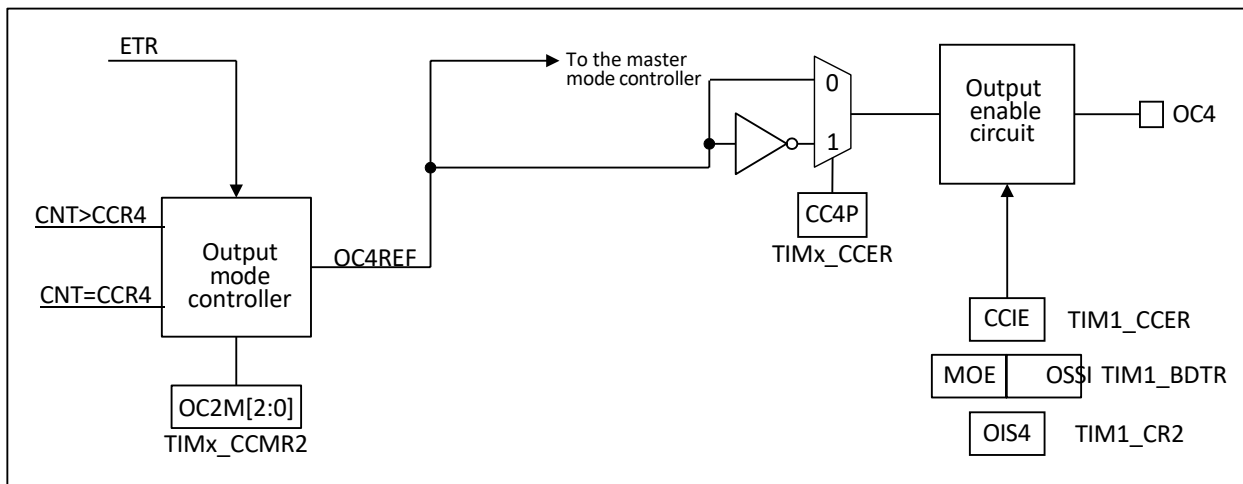


Fig 10.1-30 Output stage of capture/compare channel (channel 4)



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

10.1.7 Input capture mode

In Input capture mode, the Capture/Compare Registers (TIM1_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIM1_SR register) is set and an interrupt can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIM1_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIM1_CCRx register. CCxOF is cleared when written to '0'.

The following example shows how to capture the counter value in TIM1_CCR1 when TI1 input rises. To do this, use the following procedure:

- Select the active input: TIM1_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIM1_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIM1_CCR1 register becomes read-only.
- Program the needed input filter duration with respect to the signal connected to the timer (by programming ICxF bits in the TIM1_CCMRx register if the input is a TIx input). Let's imagine that, when toggling, the input signal is not stable during at most five internal clock cycles. We must program a filter duration longer than these five clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at f DTS frequency). Then write IC1F bits to 0011 in the TIM1_CCMR1 register.
- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIM1_CCER register (rising edge in this case).
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIM1_CCMR1 register).

- Enable capture from the counter into the capture register by setting the CC1E bit in the TIM1_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIM1_DIER register, by setting the CC1DE bit in the TIM1_DIER register.

When an input capture occurs:

- The TIM1_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.

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

Note : IC interrupt can be generated by software by setting the corresponding CCxG bit in the TIM1_EGR Register.

10.1.8 PWM input mode

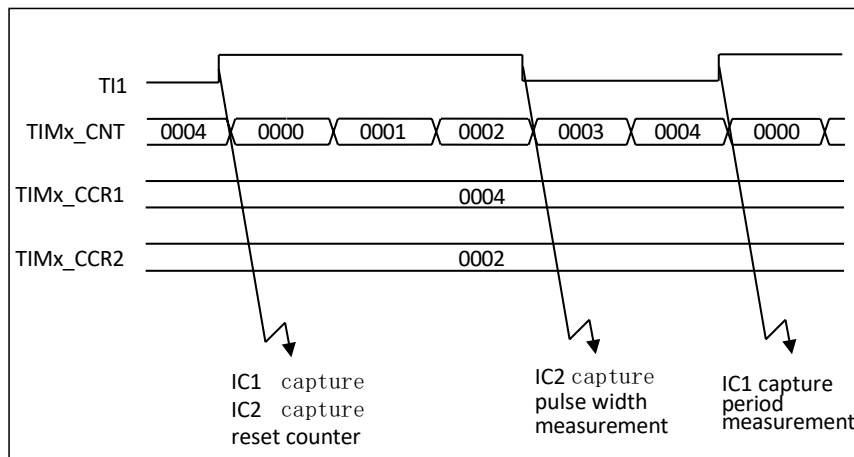
This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on the same T1x input.
- These 2 ICx signals are active on edges with opposite polarity.
- One of the two T1xFP signals is selected as trigger input and the slave mode controller is configured in reset mode.

For example, user can measure the period (in TIM1_CCR1 register) and the duty cycle (in TIM1_CCR2 register) of the PWM applied on T11 using the following procedure (depending on CK_INT frequency and prescaler value):

1. Select the active input for TIM1_CCR1: write the CC1S bits to 01 in the TIM1_CCMR1 register (T11 selected).
2. Select the active polarity for T11FP1 (used both for capture in TIM1_CCR1 and counter clear): write the CC1P bit to '0'(active on rising edge).
3. Select the active input for TIM1_CCR2: write the CC2S bits to 10 in the TIM1_CCMR1 register (T11 selected).
4. Select the active polarity for T11FP2 (used for capture in TIM1_CCR2): write the CC2P bit to '1'(active on falling edge).
5. Select the valid trigger input: write the TS bits to 101 in the TIM1_SMCR register (T11FP1 selected).
6. Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIM1_SMCR register.
7. Enable the captures: write the CC1E and CC2E bits to '1'in the TIM1_CCER register.

Fig 10.1-31 PWM input mode timing



Note: The PWM input mode can be used only with the TIM1_CH1/TIM1_CH2 signals due to the fact that only TI1FP1 and TI2FP2 are connected to the slave mode controller.

10.1.9 Forced output mode

In output mode (CCxS bits = 00 in the TIM1_CCMRx register), each output compare signal (OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, the user just needs to write 101 in the OCxM bits in the corresponding TIM1_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

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

The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIM1_CCMRx register.

Anyway, the comparison between the TIM1_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt can be sent accordingly. This is described in the output compare mode section below.

10.1.10 Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed. When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIM1_CCMRx register) and the output polarity (CCxP bit in the TIM1_CCER register). The output pin can keep its level (OCxM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIM1_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIM1_DIER

register).

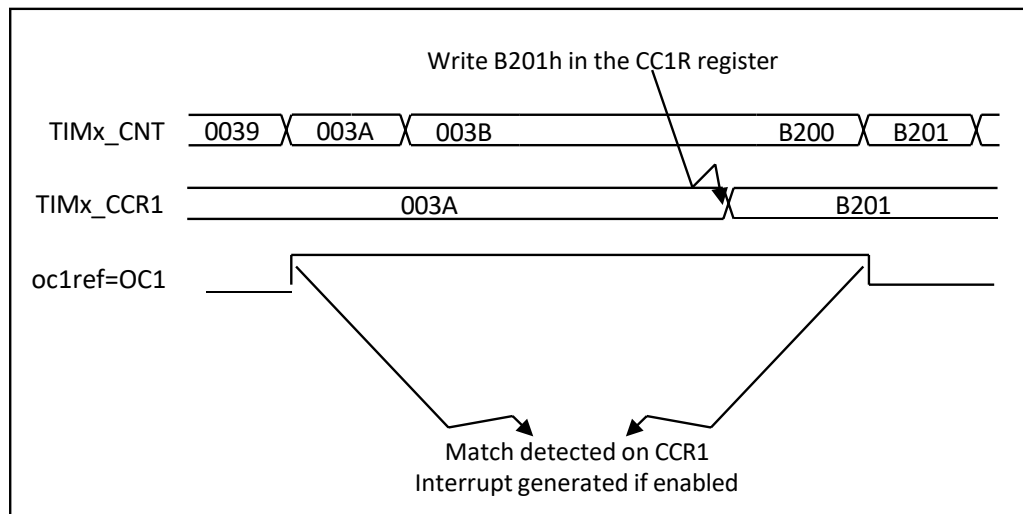
The TIM1_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIM1_CCMRx register. In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIM1_ARR and TIM1_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:
 - Write OCxM = 011 to toggle OCx output pin when CNT matches CCRx
 - Write OCxPE = 0 to disable preload register
 - Write CCxP = 0 to select active high polarity
 - Write CCxE = 1 to enable the output
5. Enable the counter by setting the CEN bit in the TIM1_CR1 register.

The TIM1_CCRx register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE='0', else TIM1_CCRx shadow register is updated only at the next update event UEV).

Fig 10.1-32 Output compare mode, toggle on OC1



10.1.11 PWM mode

Pulse Width Modulation mode allows generating a signal with a frequency determined by the value of the TIM1_ARR register and a duty cycle determined by the value of the TIM1_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bits in the TIM1_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIM1_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIM1_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, the user must initialize all the registers by setting the UG bit in the TIM1_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIM1_CCER register. It can be programmed as active high or active low. OCx output is enabled by a combination of the CCxE, CCxNE, MOE, OSSI and OSSR bits (TIM1_CCER and TIM1_BDTR registers). Refer to the TIM1_CCER register description for more details.

In PWM mode (1 or 2), TIM1_CNT and TIM1_CCRx are always compared to determine whether $TIM1_CCRx \leq TIM1_CNT$ or $TIM1_CNT \leq TIM1_CCRx$ (depending on the direction of the counter).

The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIM1_CR1 register.

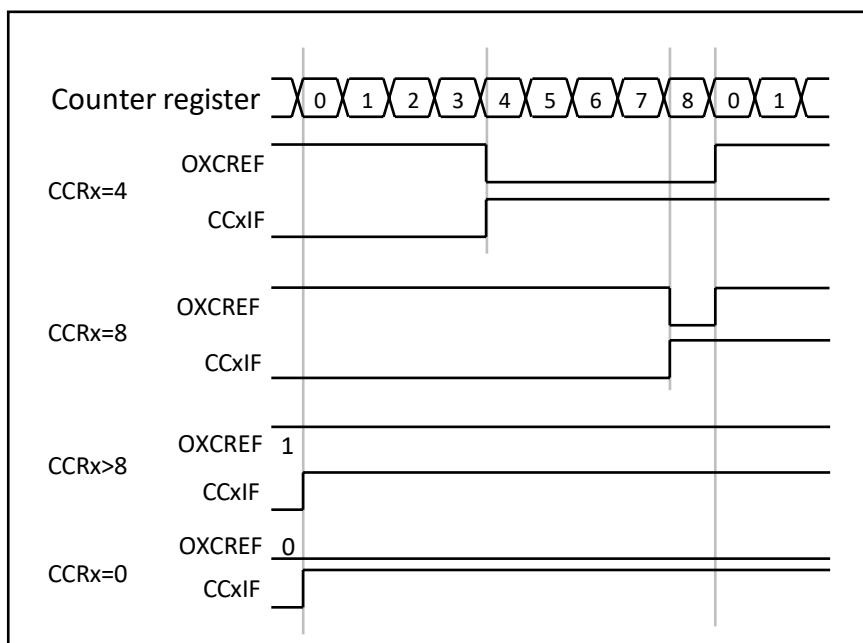
PWM edge-aligned mode

Upcounting configuration

Upcounting is active when the DIR bit in the TIM1_CR1 register is low.

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as $TIM1_CNT < TIM1_CCRx$ else it becomes low. If the compare value in TIM1_CCRx is greater than the auto-reload value (in TIM1_ARR) then OCxREF is held at '1'. If the compare value is 0 then OCxRef is held at '0'. Figure 10.1-33 shows some edge-aligned PWM waveforms in an example where $TIM1_ARR=8$.

Fig 10.1-33 Edge-aligned PWM waveforms (ARR=8)



Downcounting configuration

Downcounting is active when DIR bit in TIM1_CR1 register is high

In PWM mode 1, the reference signal OCxRef is low as long as TIM1_CNT > TIM1_CCRx else it becomes high. If the compare value in TIM1_CCRx is greater than the auto-reload value in TIM1_ARR, then OCxREF is held at '1'. 0 is not possible in this mode.

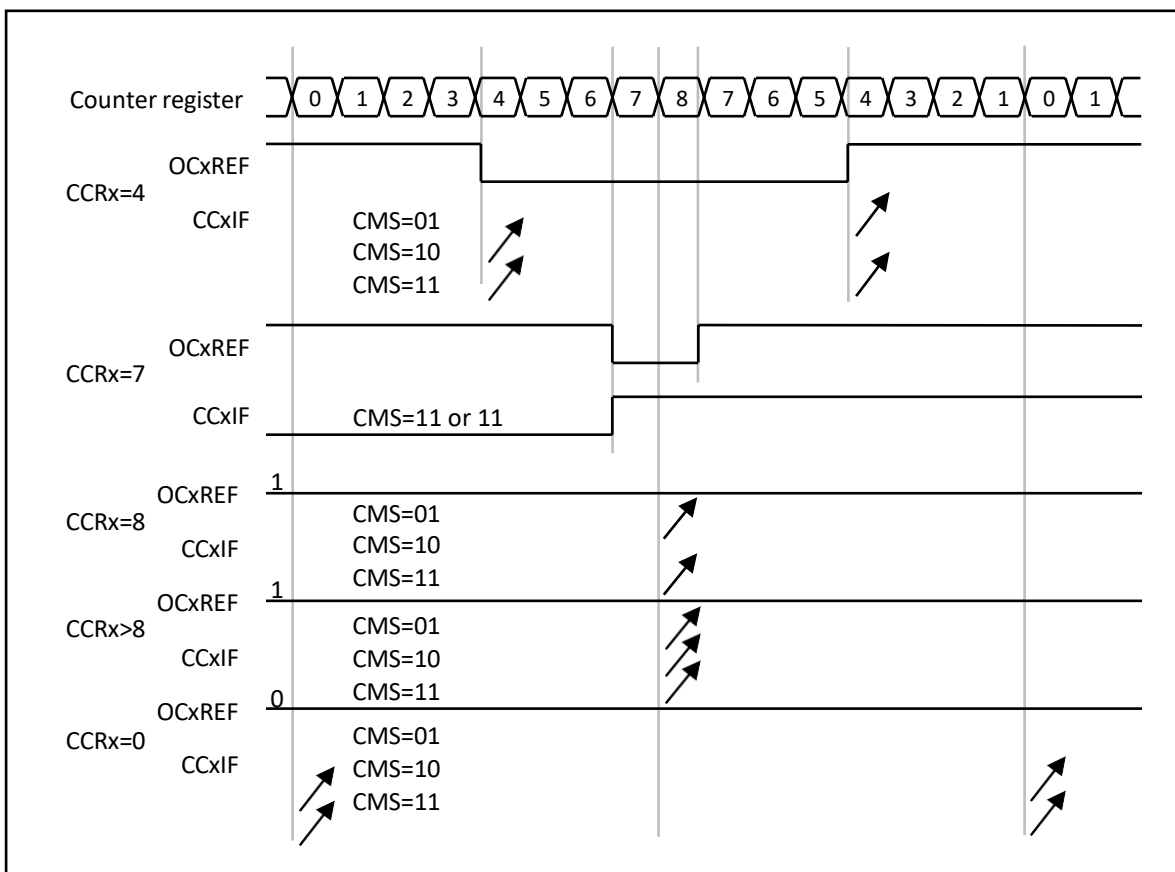
PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIM1_CR1 register are different from '00' (all the remaining configurations having the same effect on the OCxRef/OCx signals). The compare flag is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the TIM1_CR1 register is updated by hardware and must not be changed by software. 10.1.3.3

11.3-31 shows some center-aligned PWM waveforms in an example where:

- TIM1_ARR=8,
- PWM mode is the PWM mode 1,
- The flag is set when the counter counts down corresponding to the center-aligned mode 1 selected for CMS=01 in TIM1_CR1 register.

Fig 10.1-34 Center-aligned PWM waveforms (ARR=8)



Hints on using center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIM1_CR1

register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.

- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:
 - The direction is not updated if the user writes a value in the counter greater than the auto-reload value ($TIM1_CNT > TIM1_ARR$). For example, if the counter was counting up, it will continue to count up.
 - The direction is updated if the user writes 0 or write the $TIM1_ARR$ value in the counter but no Update Event UEV is generated.
- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the $TIM1_EGR$ register) just before starting the counter and not to write the counter while it is running.

10.1.12 Complementary outputs and dead-time insertion

The advanced-control timers (TIM1 and TIM8) can output two complementary signals and manage the switching-off and the switching-on instants of the outputs.

This time is generally known as dead-time and it has to be adjust it depending on the devices connected to the outputs and their characteristics (intrinsic delays of level-shifters, delays due to power switches...)

User can select the polarity of the outputs (main output OCx or complementary OCxN) independently for each output. This is done by writing to the CCxP and CCxNP bits in the $TIM1_CCER$ register.

The complementary signals OCx and OCxN are activated by a combination of several control bits: the CCxE and CCxNE bits in the $TIM1_CCER$ register and the MOE, OISx, OISxN, OSSI and OSSR bits in the $TIM1_BDTR$ and $TIM1_CR2$ registers. Refer to Table 10.1-32 for more details. In particular, the dead-time is activated when switching to the IDLE state(MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. DTG[7:0] bits of the $TIM1_BDTR$ register are used to control the dead-time generation for all channels. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

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

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

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

Fig 10.1-35 Complementary output with dead-time insertion.

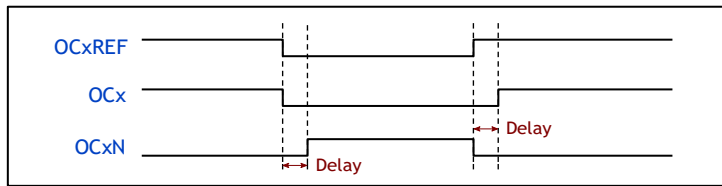


Fig 10.1-36 Dead-time waveforms with delay greater than the negative pulse.

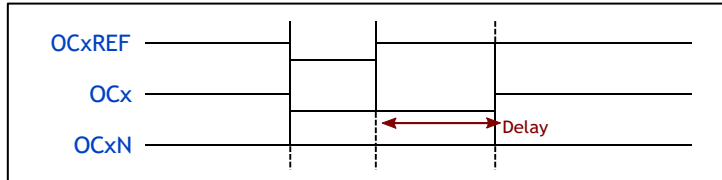
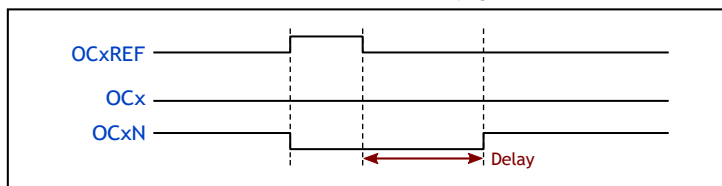


Fig 10.1-37 Dead-time waveforms with delay greater than the positive pulse.



The dead-time delay is the same for each of the channels and is programmable with the DTG bits in the TIM1_BDTR register.

10.1.13 Re-directing OCxREF to OCx or OCxN

In output mode (forced, output compare or PWM), OCxREF can be re-directed to the OCx output or to OCxN output by configuring the CCxE and CCxNE bits in the TIM1_CCER register.

This allows the user to send a specific waveform (such as PWM or static active level) on one output while the complementary remains at its inactive level. Other possibilities are to have both outputs at inactive level or both outputs active and complementary with dead-time. **Note:** When only OCxN is enabled (CCxE=0, CCxNE=1), it is not complemented and becomes active as soon as OCxREF is high. For example, if CCxNP=0 then OCxN=OCxRef. On the other hand, when both OCx and OCxN are enabled (CCxE=CCxNE=1) OCx becomes active when OCxREF is high whereas OCxN is complemented and becomes active when OCxREF is low.

10.1.14 Using the break function

When using the break function, the output enable signals and inactive levels are modified according to additional control bits (MOE, OSS1 and OSSR bits in the TIM1_BDTR register, OISx and OISxN bits in the TIM1_CR2 register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time. Refer to 10.3.10 for more details.

The break source can be either the break input pin or a clock failure event, generated by the Clock Security System (CSS), from the Reset Clock Controller. For further information on the Clock Security System, refer to 10.1-38

When exiting from reset, the break circuit is disabled and the MOE bit is low. User can enable the break

function by setting the BKE bit in the TIM1_BDTR register. The break input polarity can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time. When the BKE and BKP bits are written, a delay of 1 APB clock cycle is applied before the writing is effective. Consequently, it is necessary to wait 1 APB clock period to correctly read back the bit after the write operation.

Because MOE falling edge can be asynchronous, a resynchronization circuit has been inserted between the actual signal (acting on the outputs) and the synchronous control bit (accessed in the TIM1_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if MOE is written to 1 whereas it was low, a delay (dummy instruction) must be inserted before reading it correctly. This is because the user writes an asynchronous signal, but reads a synchronous signal

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

- The MOE bit is cleared asynchronously, putting the outputs in inactive state, idle state or in reset state (selected by the OSS1 bit). This feature functions even if the MCU oscillator is off.
- Each output channel is driven with the level programmed in the OISx bit in the TIM1_CR2 register as soon as MOE=0. If OSS1=0 then the timer releases the enable output else the enable output remains high.
- When complementary outputs are used:
 - The outputs are first put in reset state inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
 - If the timer clock is still present, then the dead-time generator is reactivated in order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that because of the resynchronization on MOE, the dead-time duration is a bit longer than usual (around 2 ck_tim clock cycles).
 - If OSS1=0 then the timer releases the enable outputs else the enable outputs remain or become high as soon as one of the CCxE or CCxNE bits is high.
- The break status flag (BIF bit in the TIM1_SR register) is set. An interrupt can be generated if the BIE bit in the TIM1_DIER register is set.
- If the AOE bit in the TIM1_BDTR register is set, the MOE bit is automatically set again at the next update event UEV. This can be used to perform a regulation, for instance. Else, MOE remains low until it is written to '1' again. In this case, it can be used for security and the break input can be connected to an alarm from power drivers, thermal sensors or any security components

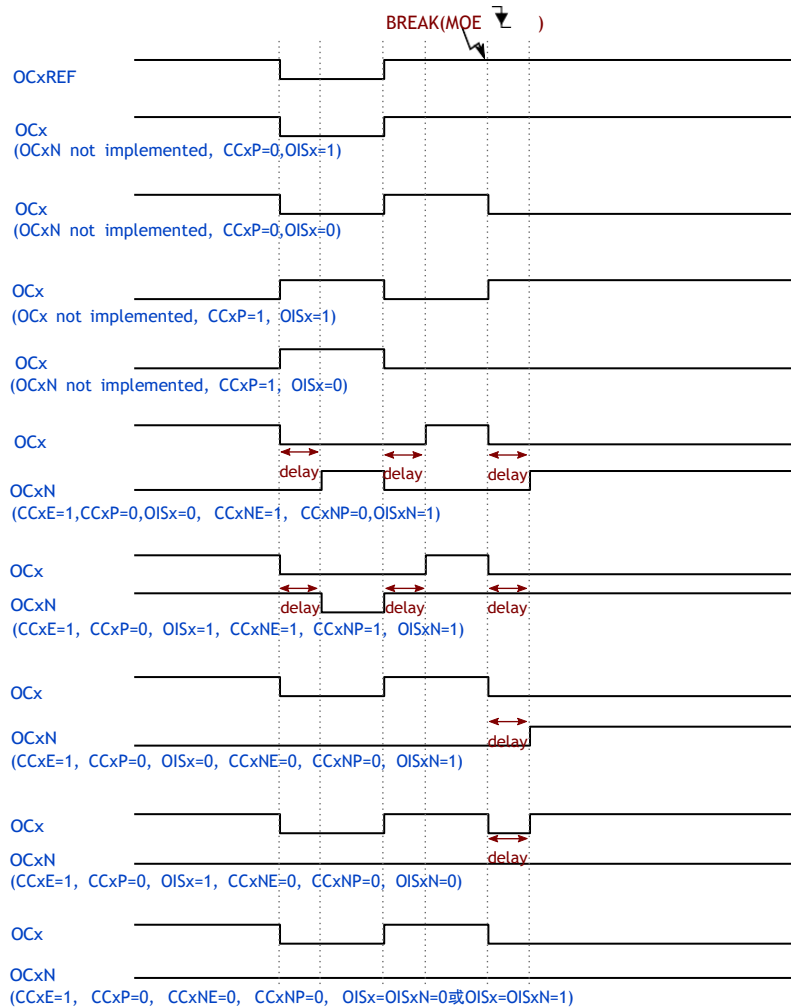
Note : *The break inputs is acting on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software). In the meantime, the status flag BIF cannot be cleared.*

The break can be generated by the BRK input which has a programmable polarity and an enable bit BKE in the TIM1_BDTR Register.

In addition to the break input and the output management, a write protection has been implemented inside the break circuit to safeguard the application. It allows freezing the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations,

break enable and polarity). The user can choose from three levels of protection selected by the LOCK bits in the TIM1_BDTR register. Refer to (TIM1_BDTR). The LOCK bits can be written only once after an MCU reset. Shows an example of behavior of the outputs in response to a break:

Fig 10.1-38 Output behavior in response to a break



10.1.15 Clearing the OCxREF signal on an external event

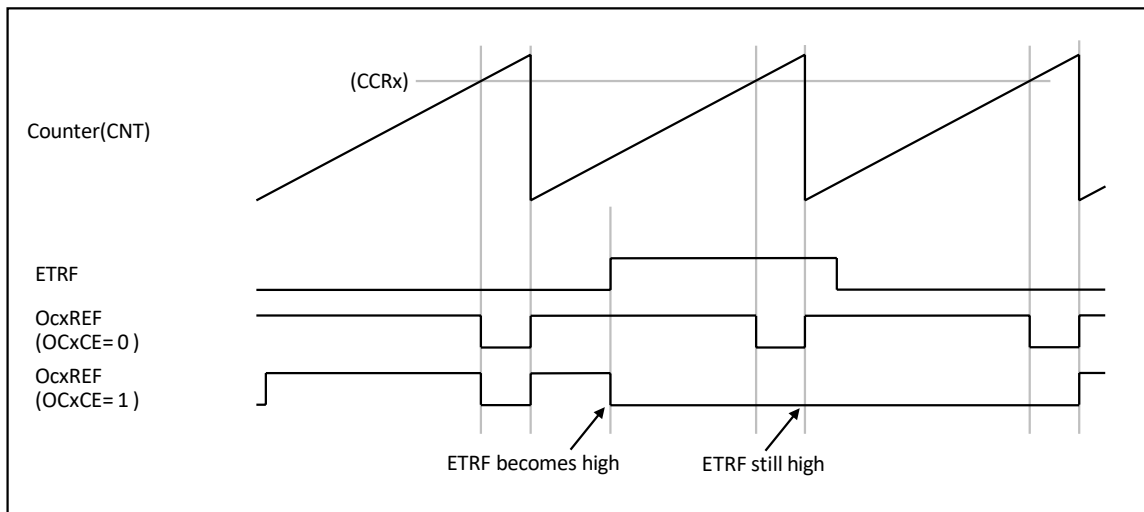
The OCxREF signal for a given channel can be driven Low by applying a High level to the ETRF input (OCxCE enable bit of the corresponding TIM1_CCMRx register set to '1'). The OCxREF signal remains Low until the next update event, UEV, occurs. This function can only be used in output compare and PWM modes, and does not work in forced mode

For example, the ETR signal can be connected to the output of a comparator to be used for current handling. In this case, the ETR must be configured as follow:

- The External Trigger Prescaler should be kept off: bits ETPS[1:0] of the TIM1_SMCR register set to '00'.
- The external clock mode 2 must be disabled: bit ECE of the TIM1_SMCR register set to '0'.
- The External Trigger Polarity (ETP) and the External Trigger Filter (ETF) can be configured according to the user needs.

Fig10.1-39 shows the behavior of the OCxREF signal when the ETRF Input becomes High, for both values of the enable bit OCxCE. In this example, the timer TIMx is programmed in PWM mode.

Fig 10.1-39 Clearing TIMx OCxREF



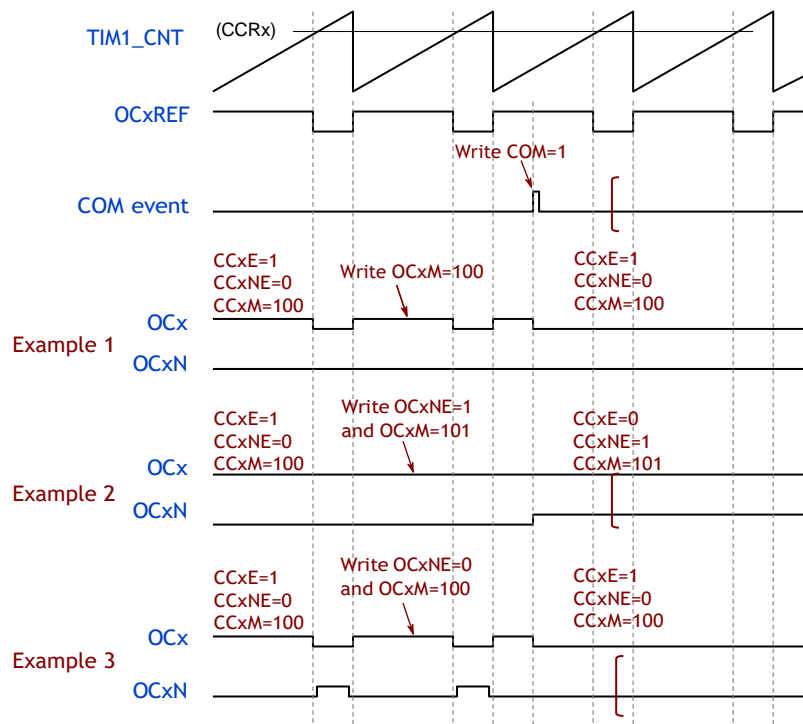
10.1.16 6-step PWM generation

When complementary outputs are used on a channel, preload bits are available on the OCxM, CCxE and CCxNE bits. The preload bits are transferred to the shadow bits at the COM commutation event. The user can thus program in advance the configuration for the next step and change the configuration of all the channels at the same time. COM can be generated by software by setting the COM bit in the TIM1_EGR register or by hardware (on TRGI rising edge).

A flag is set when the COM event occurs (COMIF bit in the TIM1_SR register), which can generate an interrupt (if the COMIE bit is set in the TIM1_DIER register).

Fig10.1-40 describes the behavior of the OCx and OCxN outputs when a COM event occurs, in 3 different examples of programmed configurations

Fig 10.1-40 6-step generation, COM example (OSSR=1)



10.1.17 One-pulse mode

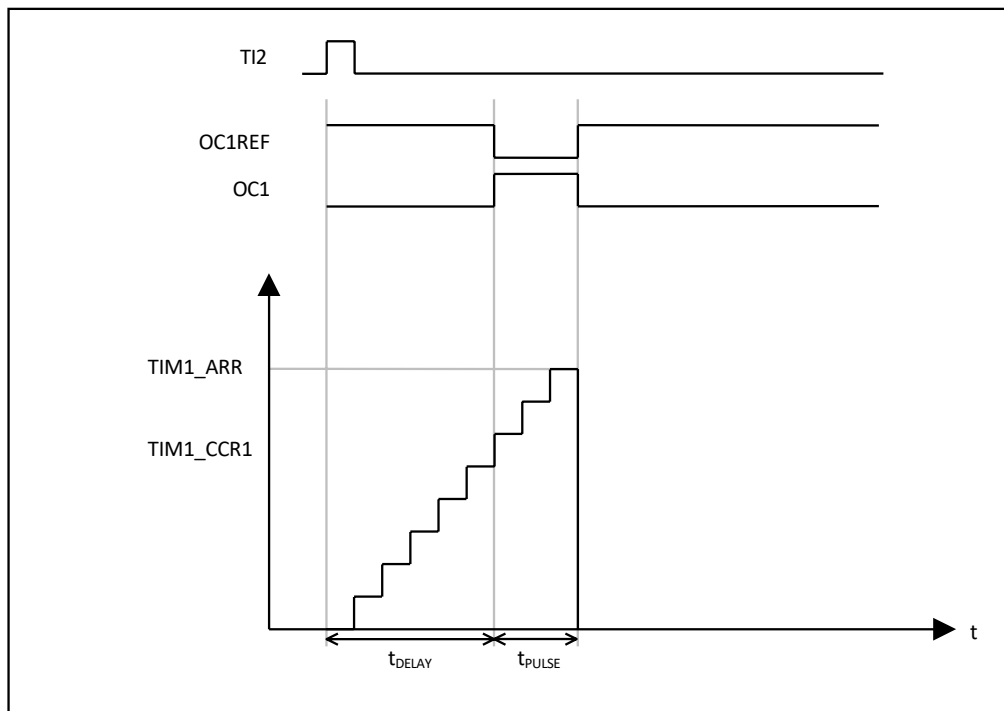
One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. Select One-pulse mode by setting the OPM bit in the TIM1_CR1 register. This makes the counter stop automatically at the next update event UEV.

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

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

Fig 10.1-41 Example of one pulse mode.



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

Let's use TI2FP2 as trigger 1:

- Map TI2FP2 to TI2 by writing $CC2S='01'$ in the TIM1_CCMR1 register.
- TI2FP2 must detect a rising edge, write $CC2P='0'$ in the TIM1_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing $TS='110'$ in the TIM1_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to '110' in the TIM1_SMCR register (trigger mode).

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

- The t_{DELAY} is defined by the value written in the TIM1_CCR1 register.
- The t_{PULXT} is defined by the difference between the auto-reload value and the compare value (TIM1_ARR - TIM1_CCR1).
- Let us say the user wants to build a waveform with a transition from '0' to '1' when a compare match occurs and a transition from '1' to '0' when the counter reaches the auto-reload value. To do this, enable PWM mode 2 by writing $OC1M=111$ in the TIM1_CCMR1 register. The user can optionally enable the preload registers by writing $OC1PE='1'$ in the TIM1_CCMR1 register and ARPE in the TIM1_CR1 register. In this case the compare value must be written in the TIM1_CCR1 register, the auto-reload value in the TIM1_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. $CC1P$ is written to '0' in this example.

In our example, the DIR and CMS bits in the TIM1_CR1 register should be low. The user only wants one pulse (Single mode), so '1' must be written in the OPM bit in the TIM1_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0). When OPM bit in the TIM1_CR1 register is set to '0', so the Repetitive Mode is selected.

Particular case: OCx fast enable:

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

If the user wants to output a waveform with the minimum delay, the OCxFE bit in the TIM1_CCMRx register must be set. Then OCxRef (and OCx) are forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2 mode.

10.1.18 Encoder interface mode

To select Encoder Interface mode write SMS = 001 in the TIM1_SMCR register if the counter is counting on TI2 edges only, SMS='010' if it is counting on TI1 edges only and SMS='011' if it is counting on both TI1 and TI2 edges.

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIM1_CCER register. When needed, the user can program the input filter as well.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Refer to Table 10.1-1. The counter is clocked by each valid transition on TI1FP1 or TI2FP2 (TI1 and TI2 after input filter and polarity selection, TI1FP1=TI1 if not filtered and not inverted, TI2FP2=TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit in TIM1_CR1 register written to '1'). The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the TIM1_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIM1_ARR register (0 to ARR or ARR down to 0 depending on the direction). So user must configure TIM1_ARR before starting. In the same way, the capture, compare, prescaler, repetition counter, trigger output features continue to work as normal. Encoder mode and External clock mode 2 are not compatible and must not be selected together.

In this mode, the counter is modified automatically following the speed and the direction of the incremental encoder and its content, therefore, always represents the encoder's position. The count direction correspond to the rotation direction of the connected sensor.

Tab 10.1-1 Counting direction versus encoder signals

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

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally be used to convert the encoder’s differential outputs to digital signals. This greatly increases noise immunity. The third encoder output which indicate the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

Fig 10.1-42gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points. For this example we assume that the configuration is the following:

- CC1S= ‘01’(TIM1_CCMR1 register, TI1FP1 mapped on TI1)
- CC2S= ‘01’(TIM1_CCMR2 register, TI2FP2 mapped on TI2)
- CC1P=‘0’, CC1NP =‘0’, IC1F =‘0000’(TIM1_CCER register, TI1FP1 noninverted, TI1FP1=TI1)
- CC2P=‘0’, CC2NP =‘0’, IC2F =‘0000’(TIM1_CCER register, TI2FP2 noninverted, TI2FP2=TI2)
- SMS= ‘011’(TIM1_SMCR register, both inputs are active on both rising and falling edges)
- CEN = 1 (TIM1_CR1 register, Counter is enabled)

Fig 10.1-42 Example of counter operation in encoder interface mode

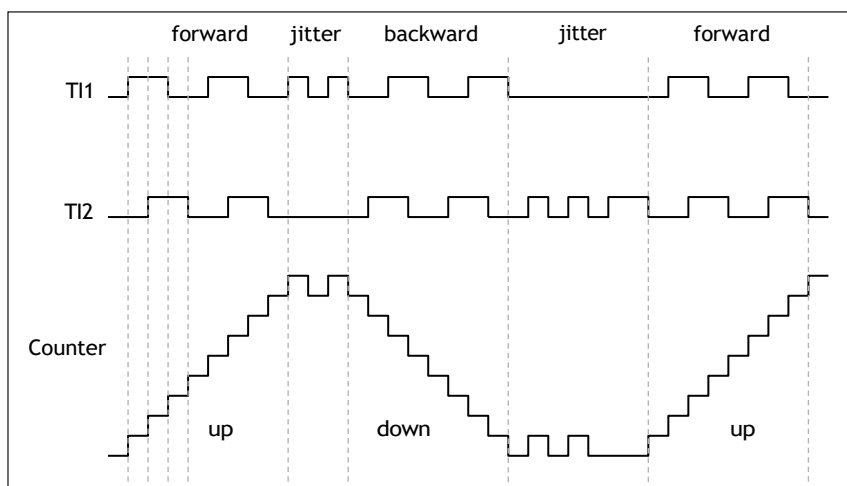
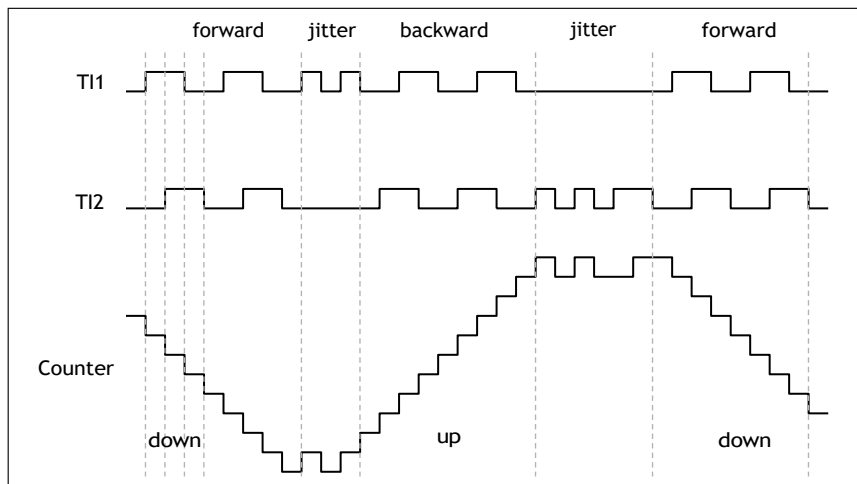


Fig 10.1-43gives an example of counter behavior when TI1FP1 polarity is inverted (sameconfiguration as above except CC1P=1).

Fig 10.1-43 Example of encoder interface mode with TI1FP1 polarity inverted



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

10.1.19 Timer input XOR function

The TI1S bit in the TIM1_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIM1_CH1, TIM1_CH2 and TIM1_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture. An example of this feature used to interface Hall sensors is given in 10.1.20

10.1.20 Interfacing with Hall sensors

This is done using the advanced-control timers (TIM1 or TIM8) to generate PWM signals to drive the motor and another timer TIMx (TIM2, TIM3, TIM4 or TIM5) referred to as "interfacing timer" in Figure 10.1-44. The "interfacing timer" captures the 3 timer input pins (TIM1_CH1, TIM1_CH2, and TIM1_CH3) connected through a XOR to the TI1 input channel (selected by setting the TI1S bit in the TIM1_CR2 register).

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

On the "interfacing timer", capture/compare channel 1 is configured in capture mode, capture signal is TRC. Figure 10.1-27. The captured value, which corresponds to the time elapsed between 2 changes on the inputs, gives information about motor speed.

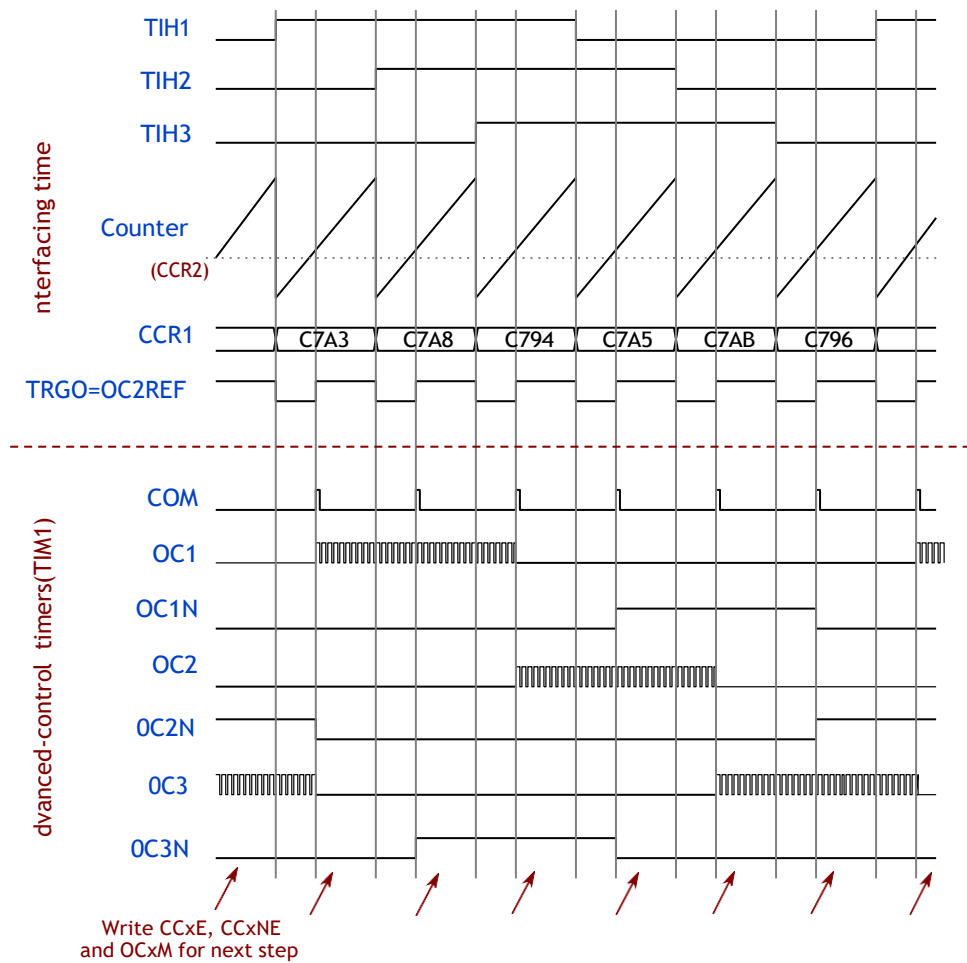
The “interfacing timer” can be used in output mode to generate a pulse which changes the configuration of the channels of the advanced-control timer (TIM1 or TIM8) (by triggering a COM event). The TIM1 timer is used to generate PWM signals to drive the motor. To do this, the interfacing timer channel must be programmed so that a positive pulse is generated after a programmed delay (in output compare or PWM mode). This pulse is sent to the advanced-control timer (TIM1 or TIM8) through the TRGO output.

Example: the user wants to change the PWM configuration of the advanced-control timer TIM1 after a programmed delay each time a change occurs on the Hall inputs connected to one of the TIMx timers.

- Configure 3 timer inputs ORed to the TI1 input channel by writing the TI1S bit in the TIM1_CR2 register to ‘1’,
- Program the time base: write the TIM1_ARR to the max value (the counter must be cleared by the TI1 change. Set the prescaler to get a maximum counter period longer than the time between 2 changes on the sensors,
- Program channel 1 in capture mode (TRC selected): write the CC1S bits in the TIM1_CCMR1 register to ‘11’. The user can also program the digital filter if needed,
- Program channel 2 in PWM 2 mode with the desired delay: write the OC2M bits to ‘111’ and the CC2S bits to ‘00’ in the TIM1_CCMR1 register,
- Select OC2REF as trigger output on TRGO: write the MMS bits in the TIM1_CR2 register to ‘101’

In the advanced-control timer TIM1, the right ITR input must be selected as trigger input, the timer is programmed to generate PWM signals, the capture/compare control signals are preloaded (CCPC=1 in the TIM1_CR2 register) and the COM event is controlled by the trigger input (CCUS=1 in the TIM1_CR2 register). The PWM control bits (CCxE, OCxM) are written after a COM event for the next step (this can be done in an interrupt subroutine generated by the rising edge of OC2REF).

Fig 10.1-44 Example of Hall sensor interface



10.1.21 TIM1 and external trigger synchronization

The TIM1 timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

10.1.22 Slave mode: Reset mode

The counter and its prescaler can be reinitialized in response to an event on a trigger input. Moreover, if the URS bit from the TIM1_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIM1_ARR, TIM1_CCRx) are updated. In the following example, the upcounter is cleared in response to a rising edge on TI1 input:

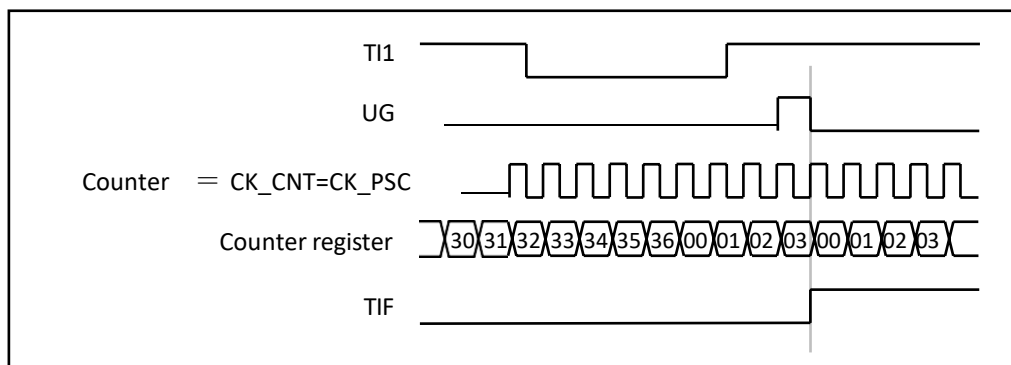
- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC1S bits select the input capture source only, CC1S = 01 in the TIM1_CCMR1 register. Write CC1P=0 in TIM1_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS=100 in TIM1_SMCR register. Select TI1 as the input source by writing TS=101 in TIM1_SMCR register.

- Start the counter by writing CEN=1 in the TIM1_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. In the meantime, the trigger flag is set (TIF bit in the TIM1_SR register) and an interrupt request can be sent if enabled (depending on the TIE and TDE bits in TIM1_DIER register).

The following figure shows this behavior when the auto-reload register TIM1_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

Fig 10.1-45 Control circuit in reset mode



10.1.23 Slave mode: Gated mode

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

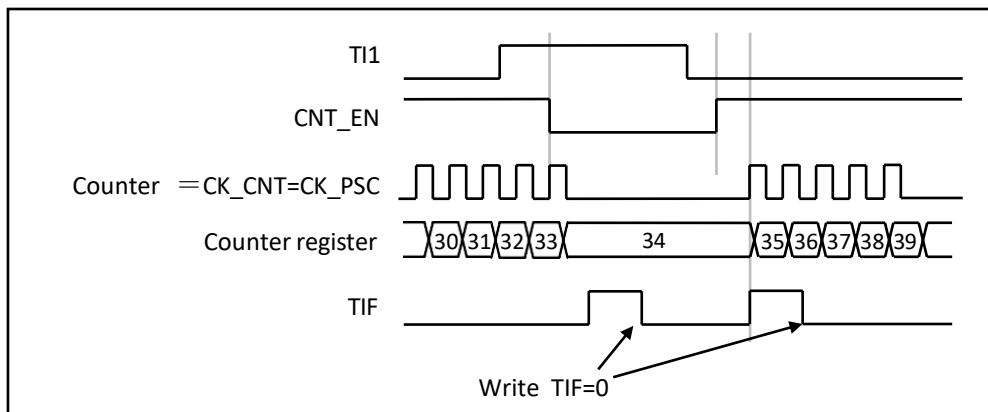
In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so the user does not need to configure it. The CC1S bits select the input capture source only, CC1S=01 in TIM1_CCMR1 register. Write CC1P=1 in TIM1_CCER register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS=101 in TIM1_SMCR register. Select TI1 as the input source by writing TS=101 in TIM1_SMCR register.
- Enable the counter by writing CEN=1 in the TIM1_CR1 register (in gated mode, the counter doesn't start if CEN=0, whatever is the trigger input level).

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIM1_SR register is set both when the counter starts or stops.

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

Fig 10.1-46 Control circuit in gated mode



10.1.24 Slave mode: Trigger mode

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

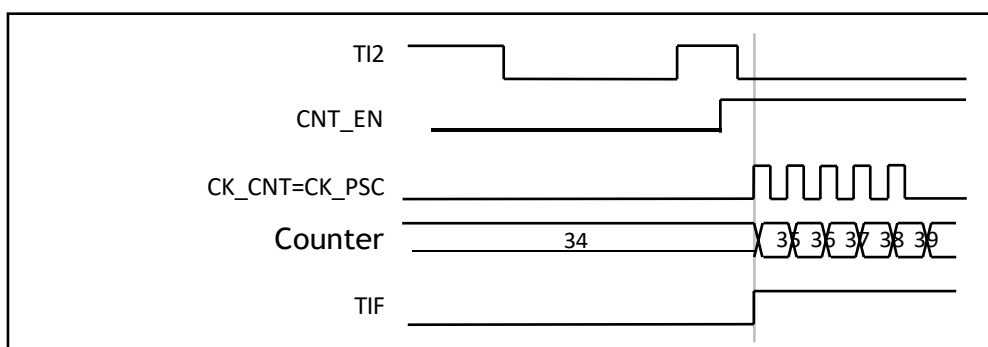
In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we don't need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC2S bits are configured to select the input capture source only, CC2S=01 in TIM1_CCMR1 register. Write CC2P=1 in TIM1_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS=110 in TIM1_SMCR register. Select TI2 as the input source by writing TS=110 in TIM1_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set.

The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

Fig 10.1-47 Control circuit in trigger mode



10.1.25 Slave mode: external clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input (in reset mode, gated mode or trigger mode). It is recommended not to select ETR as TRGI through the TS bits of TIM1_SMCR register.

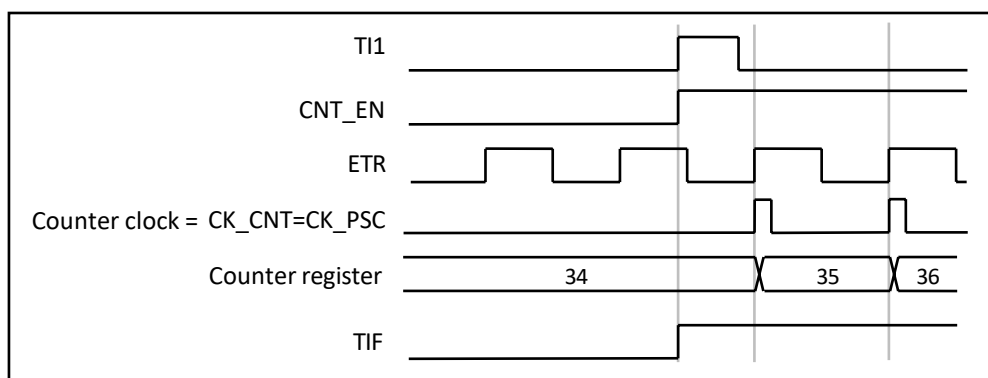
In the following example, the upcounter is incremented at each rising edge of the ETR signal as soon as a rising edge of TI1 occurs:

1. Configure the external trigger input circuit by programming the TIM1_SMCR register as follows:
 - ETF = 0000: no filter
 - ETPS = 00: prescaler disabled
 - ETP = 0: detection of rising edges on ETR and ECE=1 to enable the external clock mode 2.
2. Configure the channel 1 as follows, to detect rising edges on TI:
 - IC1F=0000: no filter.
 - The capture prescaler is not used for triggering and does not need to be configured.
 - CC1S=01 in TIM1_CCMR1 register to select only the input capture source
 - CC1P=0 in TIM1_CCER register to validate the polarity (and detect rising edge only).
3. Configure the timer in trigger mode by writing SMS=110 in TIM1_SMCR register. Select TI1 as the input source by writing TS=101 in TIM1_SMCR register.

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges.

The delay between the rising edge of the ETR signal and the actual reset of the counter is due to the resynchronization circuit on ETRP input.

Fig 10.1-48 Control circuit in external clock mode 2 + trigger mode



10.1.26 Timer synchronization

The TIM timers are linked together internally for timer synchronization or chaining

10.1.27 Debug mode

When the microcontroller enters debug mode (Cortex-M0+ core halted), the TIMx counter either continues to work normally or stops

10.2 TIM1 Registers

Tab 10.2-1 TIM1 register mapping

offset	Register	Reset value	Description
TIM1 base address: 0x4000_1000(The peripheral registers can be accessed by half-words words (32-bit))			
0x00	TIM1_CR1	0x0000_0000	TIM1 control register 1
0x04	TIM1_CR2	0x0000_0000	TIM1 control register 2
0x08	TIM1_SMCR	0x0000_0000	TIM1 slave mode control register
0x0C	TIM1_DIER	0x0000_0000	TIM1 interrupt enable register
0x10	TIM1_SR	0x0000_0000	TIM1 status register
0x14	TIM1_EGR	0x0000_0000	TIM1 event generation register
0x18	TIM1_CCMR1	0x0000_0000	TIM1 capture/compare mode register 1
0x1C	TIM1_CCMR2	0x0000_0000	TIM1 capture/compare mode register 2
0x20	TIM1_CCER	0x0000_0000	TIM1 capture/compare enable register
0x24	TIM1_CNT	0x0000_0000	TIM1 counter register
0x28	TIM1_PSC	0x0000_0000	TIM1 prescaler
0x2C	TIM1_ARR	0x0000_0000	TIM1 auto-reload register
0x30	TIM1_RCR	0x0000_0000	TIM1 repetition counter register
0x34	TIM1_CCR1	0x0000_0000	TIM1 capture/compare register 1
0x38	TIM1_CCR2	0x0000_0000	TIM1 capture/compare register 2
0x3C	TIM1_CCR3	0x0000_0000	TIM1 capture/compare register 3
0x40	TIM1_CCR4	0x0000_0000	TIM1 capture/compare register 4
0x44	TIM1_BDTR	0x0000_0000	TIM1 break and dead-time register

10.3 TIM1 register description

10.3.1 TIM1 control register 1(TIM1_CR1)

Register	Address offset	Access	Reset value	Description
TIM1_CR1	0x00	RW	0x0000_0000	TIM1 control register 1

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						CKD[1:0]	
7	6	5	4	3	2	1	0
ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN

TIM1 control register 1(TIM1_CR1)

Bit	Access	Description
[31:10]	-	Reserved, Always read 0 .
[9:8]	RW	<p>CKD[1:0]: Clock division</p> <p>These two bits are defined in the timer clock (CK_INT) frequency, dead-time and by a dead-time generator with a digital filter The frequency division ratio between sampling clocks used in a wave apparatus (ETR, Tlx).</p> <p>00: $t_{DTS} = t_{CK_INT}$</p> <p>01: $t_{DTS} = 2 * t_{CK_INT}$</p> <p>10: $t_{DTS} = 4 * t_{CK_INT}$</p> <p>11: Reserved, do not program this value</p>
[7]	RW	<p>ARPE: Auto-reload preload enable</p> <p>0: TIM1_ARR register is not buffered</p> <p>1: TIM1_ARR register is buffered</p>
[6:5]	RW	<p>CMS[1:0]: Center-aligned mode selection</p> <p>00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR).</p> <p>01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIM1_CCMRx register) are set only when the counter is counting down.</p> <p>10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIM1_CCMRx register) are set only when the counter is counting up.</p> <p>11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIM1_CCMRx register) are set both when the counter is counting up or down.</p> <p>Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1)</p>

[4]	RW	<p>DIR: Direction 0: Counter used as upcounter 1: Counter used as downcounter</p> <p>Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.</p>
[3]	RW	<p>OPM: One pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)</p>
[2]	RW	<p>URS: Update request source This bit is set and cleared by software to select the UEV event sources.</p> <p>1. If 0 is set, one of the following events generates an update interrupt:</p> <ul style="list-style-type: none"> • Counter overflow/underflow • Setting the UG bit • Update generation through the slave mode controller <p>2. If set to 1, only counter overflow/underflow causes update interrupts</p>
[1]	RW	<p>UDIS: Update disable This bit is set and cleared by software to select the UEV event sources.</p> <p>0: UEV enabled. The Update (UEV) event is generated by one of the following events:</p> <ul style="list-style-type: none"> • Counter overflow/underflow • Setting the UG bit • Update generation through the slave mode controller <p>Buffered registers are then loaded with their preload values.</p> <p>1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.</p>
[0]	RW	<p>CEN: Counter enable 0: Counter disabled 1: Counter enabled</p> <p>Note: External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware.</p>

10.3.2 TIM1 control register 2(TIM1_CR2)

Register	Address offset	Access	Reset value	Description
TIM1_CR2	0x04	RW	0x0000_0000	TIM1 control register 2

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1
7	6	5	4	3	2	1	0
T11S	MMS[1:0]			Reserved	CCUS	Reserved	CCPC

TIM1 control register 2(TIM1_CR2)

Bit	Access	Description
[31:15]	-	Reserved, Always read as 0
[14]	RW	OIS4: Output Idle state 4 (OC4 output) refer to OIS1 bit
[13]	RW	OIS3N: Output Idle state 3 (OC3N output) refer to OIS1N bit
[12]	RW	OIS3: Output Idle state 3 (OC3 output) refer to OIS1 bit
[11]	RW	OIS2N: Output Idle state 2 (OC2N output) refer to OIS1N bit
[10]	RW	OIS2: Output Idle state 2 (OC2 output) refer to OIS1 bit
[9]	RW	OIS1N: Output Idle state 1 (OC1N output) 0: OC1N=0 after a dead-time when MOE=0 1: OC1N=1 after a dead-time when MOE=0 Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register).
[8]	RW	OIS1: Output Idle state 1 (OC1 output) 0: OC1=0 (after a dead-time if OC1N is implemented) when MOE=0 1: OC1=1 (after a dead-time if OC1N is implemented) when MOE=0 Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register).
[7]	RW	TI1S: TI1 selection 0: The TIM1_CH1 pin is connected to TI1 input 1: The TIM_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)

[6:4]	RW	<p>MMS[2:0]: Master mode selection</p> <p>These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows:</p> <table border="1" data-bbox="421 327 1425 1200"> <tr> <td data-bbox="421 327 501 479">000:</td> <td data-bbox="501 327 708 479">Reset</td> <td data-bbox="708 327 1425 479">the UG bit from the TIM1_EGR register is used as trigger out- put (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset.</td> </tr> <tr> <td data-bbox="421 479 501 815">001:</td> <td data-bbox="501 479 708 815">Enable</td> <td data-bbox="708 479 1425 815">the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIM1_SMCR register).</td> </tr> <tr> <td data-bbox="421 815 501 927">010:</td> <td data-bbox="501 815 708 927">Update</td> <td data-bbox="708 815 1425 927">The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer.</td> </tr> <tr> <td data-bbox="421 927 501 1039">011:</td> <td data-bbox="501 927 708 1039">Compare Pulse</td> <td data-bbox="708 927 1425 1039">The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. (TRGO).</td> </tr> <tr> <td data-bbox="421 1039 501 1077">100:</td> <td data-bbox="501 1039 708 1077">Compare</td> <td data-bbox="708 1039 1425 1077">OC1REF signal is used as trigger output (TRGO)</td> </tr> <tr> <td data-bbox="421 1077 501 1115">101:</td> <td data-bbox="501 1077 708 1115">Compare</td> <td data-bbox="708 1077 1425 1115">OC2REF signal is used as trigger output (TRGO)</td> </tr> <tr> <td data-bbox="421 1115 501 1153">110:</td> <td data-bbox="501 1115 708 1153">Compare</td> <td data-bbox="708 1115 1425 1153">OC3REF signal is used as trigger output (TRGO)</td> </tr> <tr> <td data-bbox="421 1153 501 1191">111:</td> <td data-bbox="501 1153 708 1191">Compare</td> <td data-bbox="708 1153 1425 1191">OC4REF signal is used as trigger output (TRGO)</td> </tr> </table>	000:	Reset	the UG bit from the TIM1_EGR register is used as trigger out- put (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset.	001:	Enable	the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIM1_SMCR register).	010:	Update	The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer.	011:	Compare Pulse	The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. (TRGO).	100:	Compare	OC1REF signal is used as trigger output (TRGO)	101:	Compare	OC2REF signal is used as trigger output (TRGO)	110:	Compare	OC3REF signal is used as trigger output (TRGO)	111:	Compare	OC4REF signal is used as trigger output (TRGO)
000:	Reset	the UG bit from the TIM1_EGR register is used as trigger out- put (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset.																								
001:	Enable	the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIM1_SMCR register).																								
010:	Update	The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer.																								
011:	Compare Pulse	The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. (TRGO).																								
100:	Compare	OC1REF signal is used as trigger output (TRGO)																								
101:	Compare	OC2REF signal is used as trigger output (TRGO)																								
110:	Compare	OC3REF signal is used as trigger output (TRGO)																								
111:	Compare	OC4REF signal is used as trigger output (TRGO)																								
[3]	-	Reserved,Always read as 0																								
[2]	RW	<p>CCUS: Capture/compare control update selection</p> <p>0: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COMG bit only</p> <p>1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COMG bit or when an rising edge occurs on TRGI</p> <p>Note: <i>This bit acts only on channels that have a complementary output.</i></p>																								
[1]	-	Reserved,Always read as 0																								
[0]	RW	<p>CCPC: Capture/compare preloaded control</p> <p>0: CCxE, CCxNE and OCxM bits are not preloaded</p> <p>1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when a commutation event (COM) occurs (COMG bit set or rising edge detected on TRGI, depending on the CCUS bit).</p> <p>Note: <i>This bit acts only on channels that have a complementary output.</i></p>																								

10.3.3 TIM1 slave mode control register(TIM1_SMCR)

Register	Address offset	Access	Reset value	Description
TIM1_SMCR	0x08	RW	0x0000_0000	TIM1 slave mode control register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ETP	ECE	ETPS[1:0]		ETF[3:0]			
7	6	5	4	3	2	1	0
MSM	TS[2:0]			Reserved	SMS[2:0]		

TIM1 slave mode control register

Bit	Access	Description
[31:16]	-	Reserved, Always read as 0
[15]	RW	<p>ETP: External trigger polarity This bit selects whether ETR or \overline{ETR} is used for trigger operations 0: ETR is non-inverted, active at high level or rising edge. 1: ETR is inverted, active at low level or falling edge.</p>
[14]	RW	<p>ECE: External clock enable This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal.</p> <p>Note :</p> <ol style="list-style-type: none"> Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=111). It is possible to simultaneously use external clock mode 2 with the following slave modes: reset mode, gated mode and trigger mode. Nevertheless, TRGI must not be connected to ETRF in this case (TS bits must not be 111). If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
[13:12]	RW	<p>ETPS[1:0]: External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of TIMxCLK frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8</p>

[11:8]	RW	<p>ETF[3:0]: External trigger filter</p> <p>This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output:</p> <table border="0" style="width: 100%;"> <tr> <td>0000: No filter, sampling is done at f_{DTS}</td> <td>1000: $f_{SAMPLING} = f_{DTS}/8$, N=6</td> </tr> <tr> <td>0001: $f_{SAMPLING} = f_{CK_INT}$, N = 2</td> <td>1001: $f_{SAMPLING} = f_{DTS}/8$, N = 8</td> </tr> <tr> <td>0010: $f_{SAMPLING} = f_{CK_INT}/2$, N = 4</td> <td>1010: $f_{SAMPLING} = f_{DTS}/16$, N = 5</td> </tr> <tr> <td>0011: $f_{SAMPLING} = f_{CK_INT}/2$, N = 8</td> <td>1011: $f_{SAMPLING} = f_{DTS}/16$, N = 6</td> </tr> <tr> <td>0100: $f_{SAMPLING} = f_{DTS}/2$, N = 6</td> <td>1100: $f_{SAMPLING} = f_{DTS}/16$, N = 8</td> </tr> <tr> <td>0101: $f_{SAMPLING} = f_{DTS}/2$, N = 8</td> <td>1101: $f_{SAMPLING} = f_{DTS}/32$, N = 5</td> </tr> <tr> <td>0110: $f_{SAMPLING} = f_{DTS}/4$, N = 6</td> <td>1110: $f_{SAMPLING} = f_{DTS}/32$, N = 6</td> </tr> <tr> <td>0111: $f_{SAMPLING} = f_{DTS}/4$, N = 8</td> <td>1111: $f_{SAMPLING} = f_{DTS}/32$, N = 8</td> </tr> </table>	0000: No filter, sampling is done at f_{DTS}	1000: $f_{SAMPLING} = f_{DTS}/8$, N=6	0001: $f_{SAMPLING} = f_{CK_INT}$, N = 2	1001: $f_{SAMPLING} = f_{DTS}/8$, N = 8	0010: $f_{SAMPLING} = f_{CK_INT}/2$, N = 4	1010: $f_{SAMPLING} = f_{DTS}/16$, N = 5	0011: $f_{SAMPLING} = f_{CK_INT}/2$, N = 8	1011: $f_{SAMPLING} = f_{DTS}/16$, N = 6	0100: $f_{SAMPLING} = f_{DTS}/2$, N = 6	1100: $f_{SAMPLING} = f_{DTS}/16$, N = 8	0101: $f_{SAMPLING} = f_{DTS}/2$, N = 8	1101: $f_{SAMPLING} = f_{DTS}/32$, N = 5	0110: $f_{SAMPLING} = f_{DTS}/4$, N = 6	1110: $f_{SAMPLING} = f_{DTS}/32$, N = 6	0111: $f_{SAMPLING} = f_{DTS}/4$, N = 8	1111: $f_{SAMPLING} = f_{DTS}/32$, N = 8
0000: No filter, sampling is done at f_{DTS}	1000: $f_{SAMPLING} = f_{DTS}/8$, N=6																	
0001: $f_{SAMPLING} = f_{CK_INT}$, N = 2	1001: $f_{SAMPLING} = f_{DTS}/8$, N = 8																	
0010: $f_{SAMPLING} = f_{CK_INT}/2$, N = 4	1010: $f_{SAMPLING} = f_{DTS}/16$, N = 5																	
0011: $f_{SAMPLING} = f_{CK_INT}/2$, N = 8	1011: $f_{SAMPLING} = f_{DTS}/16$, N = 6																	
0100: $f_{SAMPLING} = f_{DTS}/2$, N = 6	1100: $f_{SAMPLING} = f_{DTS}/16$, N = 8																	
0101: $f_{SAMPLING} = f_{DTS}/2$, N = 8	1101: $f_{SAMPLING} = f_{DTS}/32$, N = 5																	
0110: $f_{SAMPLING} = f_{DTS}/4$, N = 6	1110: $f_{SAMPLING} = f_{DTS}/32$, N = 6																	
0111: $f_{SAMPLING} = f_{DTS}/4$, N = 8	1111: $f_{SAMPLING} = f_{DTS}/32$, N = 8																	
[7]	RW	<p>MSM: Master/slave mode</p> <p>0: No action</p> <p>1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.</p>																
[6:4]	RW	<p>TS[2:0]: Trigger selection</p> <p>This bit-field selects the trigger input to be used to synchronize the counter.</p> <table border="0" style="width: 100%;"> <tr> <td>000: Internal Trigger 0 (ITR0)</td> <td>001: Internal Trigger 1 (ITR1)</td> </tr> <tr> <td>010: Internal Trigger 2 (ITR2)</td> <td>011: Internal Trigger 3 (ITR3)</td> </tr> <tr> <td>100: TI1 Edge Detector (TI1F_ED)</td> <td>101: Filtered Timer Input 1 (TI1FP1)</td> </tr> <tr> <td>110: Filtered Timer Input 2 (TI2FP2)</td> <td>111: External Trigger input (ETRF)</td> </tr> </table> <p>See Table 10.3.3 for more details on ITRx meaning for each Timer.</p> <p>Note: <i>These bits must be changed only when they are not used (e.g. when SMS=000) to avoid wrong edge detections at the transition.</i></p>	000: Internal Trigger 0 (ITR0)	001: Internal Trigger 1 (ITR1)	010: Internal Trigger 2 (ITR2)	011: Internal Trigger 3 (ITR3)	100: TI1 Edge Detector (TI1F_ED)	101: Filtered Timer Input 1 (TI1FP1)	110: Filtered Timer Input 2 (TI2FP2)	111: External Trigger input (ETRF)								
000: Internal Trigger 0 (ITR0)	001: Internal Trigger 1 (ITR1)																	
010: Internal Trigger 2 (ITR2)	011: Internal Trigger 3 (ITR3)																	
100: TI1 Edge Detector (TI1F_ED)	101: Filtered Timer Input 1 (TI1FP1)																	
110: Filtered Timer Input 2 (TI2FP2)	111: External Trigger input (ETRF)																	
[3]	-	Reserved, Always read as 0																

[2:0]	RW	<p>SMS[2:0]: Slave mode selection</p> <p>When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control Register description).</p> <table border="1"> <thead> <tr> <th>SMS</th> <th>Mode</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>000:</td> <td>Slave mode disabled</td> <td>if CEN = '1' then the prescaler is clocked directly by the internal clock.</td> </tr> <tr> <td>001:</td> <td>Encoder mode 1</td> <td>Counter counts up/down on TI2FP1 edge depending on TI1FP2 level.</td> </tr> <tr> <td>010:</td> <td>Encoder mode 2</td> <td>Counter counts up/down on TI1FP2 edge depending on TI2FP1 level.</td> </tr> <tr> <td>011:</td> <td>Encoder mode 3</td> <td>Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input.</td> </tr> <tr> <td>100:</td> <td>Reset Mode</td> <td>Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers.</td> </tr> <tr> <td>101:</td> <td>Gated Mode</td> <td>The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled.</td> </tr> <tr> <td>110:</td> <td>Trigger Mode</td> <td>The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled.</td> </tr> <tr> <td>111:</td> <td>External Clock Mode 1</td> <td>Rising edges of the selected trigger (TRGI) clock the counter</td> </tr> </tbody> </table> <p>Note : The gated mode must not be used if TI1F_ED is selected as the trigger input (TS='100'). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.</p>	SMS	Mode	Description	000:	Slave mode disabled	if CEN = '1' then the prescaler is clocked directly by the internal clock.	001:	Encoder mode 1	Counter counts up/down on TI2FP1 edge depending on TI1FP2 level.	010:	Encoder mode 2	Counter counts up/down on TI1FP2 edge depending on TI2FP1 level.	011:	Encoder mode 3	Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input.	100:	Reset Mode	Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers.	101:	Gated Mode	The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled.	110:	Trigger Mode	The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled.	111:	External Clock Mode 1	Rising edges of the selected trigger (TRGI) clock the counter
SMS	Mode	Description																											
000:	Slave mode disabled	if CEN = '1' then the prescaler is clocked directly by the internal clock.																											
001:	Encoder mode 1	Counter counts up/down on TI2FP1 edge depending on TI1FP2 level.																											
010:	Encoder mode 2	Counter counts up/down on TI1FP2 edge depending on TI2FP1 level.																											
011:	Encoder mode 3	Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input.																											
100:	Reset Mode	Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers.																											
101:	Gated Mode	The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled.																											
110:	Trigger Mode	The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled.																											
111:	External Clock Mode 1	Rising edges of the selected trigger (TRGI) clock the counter																											

Slave timer	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
TIM1	Tim2_trgo	irq_timer10	irq_timer11	irq_pca

10.3.4 TIM1 interrupt enable register(TIM1_DIER)

Register	Address offset	Access	Reset value	Description
TIM1_DIER	0x0C	RW	0x0000_0000	TIM1 interrupt enable register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE

TIM1 interrupt enable register(TIM1_DIER)

Bit	Access	Description
-----	--------	-------------

[31:8]	-	Reserved,Always read as 0
[7]	RW	BIE: Break interrupt enable 0: Break interrupt disabled 1: Break interrupt enabled
[6]	RW	TIE: Trigger interrupt enable 0: Trigger interrupt disabled 1: Trigger interrupt enabled
[5]	RW	COMIE: COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
[4]	RW	CC4IE: Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled 1: CC4 interrupt enabled
[3]	RW	CC3IE: Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled 1: CC3 interrupt enabled
[2]	RW	CC2IE: Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled 1: CC2 interrupt enabled
[1]	RW	CC1IE: Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
[0]	RW	UIE: Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

10.3.5 TIM1 status register (TIM1_SR)

Register	Address offset	Access	Reset value	Description
TIM1_SR	0x10	RC_W0	0x0000_0000	TIM1 status register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			CC4OF	CC3OF	CC2OF	CC1OF	Reserved
7	6	5	4	3	2	1	0
BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF

TIM1 status register (TIM1_SR)

Bit	Access	Description
[31:13]	-	Reserved,Always read as 0
[12]	RC_W0	CC4OF: Capture/Compare 4 overcapture flag refer to CC1OF description
[11]	RC_W0	CC3OF: Capture/Compare 3 overcapture flag refer to CC1OF description

[10]	RC_W0	CC20F: Capture/Compare 2 overcapture flag refer to CC10F description
[9]	RC_W0	CC10F: Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The counter value has been captured in TIM1_CCR1 register while CC1F flag was already set
[8]	-	Reserved, Always read as 0
[7]	RC_W0	BIF: Break interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input
[6]	RC_W0	TIF: Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode, both edges in case gated mode is selected). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
[5]	RC_W0	COMIF: COM interrupt flag This flag is set by hardware on COM event (when Capture/compare Control bits - CCxE, CCxNE, OCxM - have been updated). It is cleared by software. 0: No COM event occurred. 1: COM interrupt pending.
[4]	RC_W0	CC4IF: Capture/Compare 4 interrupt flag refer to CC1IF description
[3]	RC_W0	CC3IF: Capture/Compare 3 interrupt flag refer to CC1IF description
[2]	RC_W0	CC2IF: Capture/Compare 2 interrupt flag refer to CC1IF description
[1]	RC_W0	CC1IF: Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIM1_CR1 register description). It is cleared by software. 0: No match. 1: The content of the counter TIM1_CNT matches the content of the TIM1_CCR1 register. When the contents of TIM1_CCR1 are greater than the contents of TIM1_ARR, the CC1IF bit goes high on the counter overflow (in upcounting and up/down-counting modes) or under-flow (in downcounting mode) If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIM1_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIM1_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)

[0]	RC_W0	<p>UIF: Update interrupt flag</p> <p>This bit is set by hardware on an update event. It is cleared by software.</p> <ol style="list-style-type: none">1. No update occurred.2. Update interrupt pending. This bit is set by hardware when the registers are updated:<ul style="list-style-type: none">• At overflow or underflow regarding the repetition counter value (update if repetition counter = 0) and if the UDIS=0 in the TIM1_CR1 register.• When CNT is reinitialized by software using the UG bit in TIM1_EGR register, if URS=0 and UDIS=0 in the TIM1_CR1 register.• When CNT is reinitialized by a trigger event 10.3.3, if URS=0 and UDIS=0 in the TIM1_CR1 register.
-----	-------	--

10.3.6 TIM1 event generation register(TIM1_EGR)

Register	Address offset	Access	Reset value	Description
TIM1_EGR	0x14	W	0x0000_0000	TIM1 event generation register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
BG	TG	COMG	COMIF	CC4G	CC3G	CC1G	UG

TIM1 event generation register(TIM1_EGR)

Bit	Access	Description
[31:8]	-	Reserved, Always read as 0
[7]	W	BG: Break generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt can occur if enabled.
[6]	W	TG: Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in TIM1_SR register. Related interrupt can occur if enabled.
[5]	W	COMIF: Capture/Compare control update generation This bit can be set by software, it is automatically cleared by hardware 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note : <i>This bit acts only on channels having a complementary output</i>
[4]	W	CC4G: Capture/Compare 4 generation refer to CC1G description
[3]	W	CC3G: Capture/Compare 3 generation refer to CC1G description
[2]	W	CC2G: Capture/Compare 2 generation refer to CC1G description
[1]	W	CC1G: Capture/Compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel: If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIM1_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.

[0]	W	<p>UG: Update generation</p> <p>This bit can be set by software, it is automatically cleared by hardware. 0: No action</p> <p>1: Reinitialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), else it takes the auto-reload value (TIM1_ARR) if DIR=1 (downcounting).</p>
-----	---	---

10.3.7 TIM1 capture/compare mode register 1(TIM1_CCMR1)

Register	Address offset	Access	Reset value	Description
TIM1_CCMR1	0x18	RW	0x0000_0000	TIM1 capture/compare mode register 1

31	30	29	28	27	26	25	24
Reserved							
Reserved							
23	22	21	20	19	18	17	16
Reserved							
Reserved							
15	14	13	12	11	10	9	8
IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]	
OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]	
7	6	5	4	3	2	1	0
IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	

Input capture mode

TIM1 capture/compare mode register 1(TIM1_CCMR1)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15:12]	RW	IC2F[3:0]: Input capture 2 filter
[11:10]	RW	IC2PSC[1:0]: Input capture 2 prescaler
[9:8]	RW	<p>CC2S[1:0]: Capture/Compare 2 selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output</p> <p>01: CC2 channel is configured as input, IC2 is mapped on TI2 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 working only if an internal trigger input is selected through TS bit (TIM1_SMCR register)</p> <p>Note : CC2S bits are writable only when the channel is OFF (CC2E = '0' in TIM1_CCER).</p>

[7:4]	RW	<p>IC1F[3:0]: Input capture 1 filter</p> <p>This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done 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 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: $f_{SAMPLING} = f_{DTS}/8$, N = 8</p>
[3:2]	RW	<p>IC1PSC[1:0]: Input capture 1 prescaler</p> <p>This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E='0'(TIM1_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events</p>
[1:0]	RW	<p>CC1S[1:0]: Capture/Compare 1 Selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM1_SMCR register)</p> <p>Note : CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIM1_CCER).</p>

TIM1_CCMR1 Output compare mode :

TIM1 Output compare mode (TIM1_CCMR1)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15]	RW	OC2CE: Output Compare 2 clear enable
[14:12]	RW	OC2M[2:0]: Output Compare 2 mode
[11]	RW	OC2PE: Output Compare 2 preload enable
[10]	RW	OC2FE: Output Compare 2 fast enable

[9:8]	RW	<p>CC2S[1:0]: Capture/Compare 2 selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output</p> <p>01: CC2 channel is configured as input, IC2 is mapped on TI2 10:</p> <p>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 working only if an internal trigger input is selected through the TS bit (TIM1_SMCR register)</p> <p>Note :</p> <p><i>CC2S bits are writable only when the channel is OFF (CC2E = '0' in TIM1_CCER)</i></p>
[7]	RW	<p>OC1CE: Output Compare 1 clear enable</p> <p>OC1CE: Output Compare 1 Clear Enable</p> <p>0: OC1Ref is not affected by the ETRF Input</p> <p>1: OC1Ref is cleared as soon as a High level is detected on ETRF input</p>
[6:4]	RW	<p>OC1M[2:0]: Output Compare 1 mode</p> <p>These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits.</p> <p>000: Frozen - The comparison between the output compare register TIM1_CCR1 and the counter TIM1_CNT has no effect on the outputs.(this mode is used to generate a timing base).</p> <p>001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIM1_CNT matches the capture/compare register 1 (TIM1_CCR1).</p> <p>010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIM1_CNT matches the capture/compare register 1 (TIM1_CCR1).</p> <p>011: Toggle - OC1REF toggles when TIM1_CNT=TIM1_CCR1.</p> <p>100: Force inactive level - OC1REF is forced low.</p> <p>101: Force active level - OC1REF is forced high.</p> <p>110: PWM mode 1 - In upcounting, channel 1 is active as long as TIM1_CNT<TIM1_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF='0') as long as TIM1_CNT>TIM1_CCR1 else active (OC1REF='1').</p> <p>111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIM1_CNT<TIM1_CCR1 else active. In downcounting, channel 1 is active as long as TIM1_CNT>TIM1_CCR1 else inactive</p> <p>Note :</p> <ol style="list-style-type: none"> <i>These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIM1_BDTR register) and CC1S='00' (the channel is configured in output).</i> <i>In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from "frozen" mode to "PWM" mode.</i>

[3]	RW	<p>OC1PE: Output Compare 1 preload enable</p> <p>0: Preload register on TIM1_CCR1 disabled. TIM1_CCR1 can be written at anytime, the new value is taken in account immediately.</p> <p>1: Preload register on TIM1_CCR1 enabled. Read/Write operations access the preload register. TIM1_CCR1 preload value is loaded in the active register at each update event.</p> <p>Note :</p> <ol style="list-style-type: none"> 1. These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIM1_BDTR register) and CC1S='00' (the channel is configured in output). 2. The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIM1_CR1 register). Else the behavior is not guaranteed.
[2]	RW	<p>OC1FE: Output Compare 1 fast enable</p> <p>This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles.</p> <p>1: An active edge on the trigger input acts like a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OC1FE acts only if the channel is configured in PWM1 or PWM2 mode</p>
[1:0]	RW	<p>CC1S[1:0]: Capture/Compare 1 selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 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 on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM1_SMCR register)</p> <p>Note :</p> <p>CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIM1_CCER)</p>

10.3.8 TIM1 capture/compare mode register 2(TIM1_CCMR2)

Register	Address offset	Access	Reset value	Description
TIM1_CCMR2	0x1C	RW	0x0000_0000	TIM1 capture/compare mode register 2

31	30	29	28	27	26	25	24
Reserved							
Reserved							
23	22	21	20	19	18	17	16
Reserved							
Reserved							
15	14	13	12	11	10	9	8
IC4F[3:0]				IC4PSC[1:0]		CC4S[1:0]	
OC4CE	OC2M[2:0]			OC4PE	OC4FE	CC4S[1:0]	
7	6	5	4	3	2	1	0
IC3F[3:0]				IC3PSC[1:0]		CC3S[1:0]	
OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	

10.3.9 TIM1_CCMR2 capture/compare mode register 2**TIM1_CCMR2 Input capture mode :**

capture/compare mode register 2(TIM1_CCMR2)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15:12]	RW	IC4F[3:0]: Input capture 4 filter.
[11:10]	RW	IC4PSC[1:0]: Input capture 4 prescaler
[9:8]	RW	CC4S[1:0]: Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM1_SMCR register) Note : <i>CC4S bits are writable only when the channel is OFF (CC4E = '0' in TIM1_CCER).</i>
[7:4]	RW	IC3F[3:0]: Input capture 3 filter
[3:2]	RW	IC3PSC[1:0]: Input capture 3 prescaler
[1:0]	RW	CC3S[1:0]: Capture/compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM1_SMCR register) Note : <i>CC3S bits are writable only when the channel is OFF (CC3E = '0' in TIM1_CCER)</i>

TIM1_CCMR2 Output compare mode :

capture/compare mode register 2(TIM1_CCMR2)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15]	RW	OC4CE: Output compare 4 clear enable
[14:12]	RW	OC4M[2:0]: Output compare 4 mode
[11]	RW	OC4PE: Output compare 4 preload enable
[10]	RW	OC4FE: Output compare 4 fast enable

[9:8]	RW	<p>CC4S[1:0]: Capture/Compare 4 selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output</p> <p>01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3</p> <p>11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM1_SMCR register)</p> <p>Note : CC4S bits are writable only when the channel is OFF (CC4E = '0' in TIM1_CCER)</p>
[7]	RW	OC3CE: Output compare 3 clear enable
[6:4]	RW	OC3M[2:0]: Output compare 3 mode
[3]	RW	OC3PE: Output compare 3 preload enable
[2]	RW	OC3FE: Output compare 3 fast enable
[1:0]	RW	<p>CC3S[1:0]: Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input.</p> <p>00: CC3 channel is configured as output</p> <p>01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4</p> <p>11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM1_SMCR register)</p>

10.3.10 TIM1 capture/compare enable register(TIM1_CCER)

Register	Address offset	Access	Reset value	Description
TIM1_CCER	0x20	RW	0x0000_0000	TIM1 capture/compare enable register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E
7	6	5	4	3	2	1	0
CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E

TIM1 capture/compare enable register(TIM1_CCER)

Bit	Access	Description
[31:14]	-	Reserved, always read as 0
[13]	RW	CC4P: Capture/Compare 4 output polarity refer to CC1P description
[12]	RW	CC4E: Capture/Compare 4 output enable refer to CC1E description
[11]	RW	CC3NP: Capture/Compare 3 complementary output polarity refer to CC1NP description
[10]	RW	CC3NE: Capture/Compare 3 complementary output enable refer to CC1NE description
[9]	RW	CC3P: Capture/Compare 3 output polarity refer to CC1P description
[8]	RW	CC3E: Capture/Compare 3 output enable refer to CC1E description
[7]	RW	CC2NP: Capture/Compare 2 complementary output polarity refer to CC1NP description
[6]	RW	CC2NE: Capture/Compare 2 complementary output enable refer to CC1NE description
[5]	RW	CC2P: Capture/Compare 2 output polarity refer to CC1P description
[4]	RW	CC2E: Capture/Compare 2 output enable refer to CC1E description
[3]	RW	CC1NP: Capture/Compare 1 complementary output polarity 0: OC1N active high. 1: OC1N active low. Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register) and CC1S="00"(the channel is configured in output).
[2]	RW	CC1NE: Capture/Compare 1 complementary output enable 0: Off - OC1N is not active. OC1N level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. 1: On - OC1N signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits.

[1]	RW	<p>CC1P: Capture/Compare 1 output polarity</p> <p>CC1 channel configured as output: 0: OC1 active high 1: OC1 active low</p> <p>CC1 channel configured as input: This bit selects whether IC1 or IC1 is used for trigger or capture operations. 0: non- inverted: capture is done on a rising edge of IC1. When used as external trigger, IC1 is non-inverted. 1: inverted: capture is done on a falling edge of IC1. When used as external trigger, IC1 is inverted.</p> <p>Note : <i>This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register).</i></p>
[0]	RW	<p>CC1E: Capture/Compare 1 output enable</p> <p>CC1 channel configured as output: 0: Off - OC1 is not active. OC1 level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. 1: On - OC1 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits.</p> <p>CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIM_CCR1) or not. 0: Capture disabled. 1: Capture enabled.</p>

Tab 10.3-12 Output control bits for complementary OCx and OCxN channels with break feature

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

10.3.11 TIM1 counter(TIM1_CNT)

Register	Address offset	Access	Reset value	Description
TIM1_CNT	0x24	RW	0x0000_0000	TIM1 counter

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

TIM1 counter (TIM1_CNT)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15:0]	RW	CNT[15:0]: Counter value.

10.3.12 TIM1 prescaler(TIM1_PSC)

Register	Address offset	Access	Reset value	Description
TIM1_PSC	0x28	RW	0x0000_0000	TIM1 prescaler

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

TIM1 prescaler(TIM1_PSC)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15:0]	RW	PSC[15:0]: Prescaler value The counter clock frequency (CK_CNT) is equal to $f_{CK_PSC} / (PSC[15:0] + 1)$. PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of TIM_EGR register or through trigger controller when configured in “reset mode”).

10.3.13 TIM1 auto-reload register(TIM1_ARR)

Register	Address offset	Access	Reset value	Description
TIM1_ARR	0x2C	RW	0x0000_0000	TIM1 auto-reload register

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

TIM1 auto-reload register(TIM1_ARR)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15:0]	RW	ARR[15:0]: Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to Section 10.1.2: Time-base unit for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

10.3.14 TIM1 repetition counter register (TIM1_RCR)

Register	Address offset	Access	Reset value	Description
TIM1_RCR	0x30	RW	0x0000_0000	TIM1 repetition counter register

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

TIM1 repetition counter register (TIM1_RCR)

Bit	Access	Description
[31:8]	-	Reserved, always read as 0
[7:0]	RW	RFP[7:0]: Repetition counter value These bits allow the user to set-up the update rate of the compare registers (i.e. periodic transfers from preload to active registers) when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNT related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event U_RC, any write to the TIM1_RCR register is not taken in account until the next repetition update event. It means in PWM mode (REP+1) corresponds to: <ul style="list-style-type: none"> • the number of PWM periods in edge-aligned mode • the number of half PWM period in center-aligned mode.

10.3.15 TIM1 capture/compare register 1(TIM1_CCR1)

Register	Address offset	Access	Reset value	Description			
TIM1_CCR1	0x34	RW	0x0000_0000	TIM1 capture/compare register 1			
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CCR1[15:8]							
7	6	5	4	3	2	1	0
CCR1[7:0]							

TIM1 capture/compare register 1

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15:0]	RW	<p>CCR1[15:0]: Capture/Compare 1 value</p> <p>If channel CC1 is configured as output:</p> <p>CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs.</p> <p>The active capture/compare register contains the value to be compared to the counter TIM1_CNT and signaled on OC1 output.</p> <p>If channel CC1 is configured as input:</p> <p>CCR1 is the counter value transferred by the last input capture 1 event (IC1).</p>

10.3.16 TIM1 capture/compare register2(TIM1_CCR2)

Register	Address offset	Access	Reset value	Description			
TIM1_CCR2	0x38	RW	0x0000_0000	TIM1 capture/compare register 2			
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CCR2[15:8]							
7	6	5	4	3	2	1	0
CCR2[7:0]							

TIM1 capture/compare register2(TIM1_CCR2)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0

[15:0]	RW	<p>CCR2[15:0]: Capture/Compare 2 value</p> <p>If channel CC2 is configured as output: CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR2 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs.</p> <p>The active capture/compare register contains the value to be compared to the counter TIM1_CNT and signalled on OC2 output.</p> <p>If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2).</p>
--------	----	---

10.3.17 TIM1 capture/compare register 3 (TIM1_CCR3)

Register	Address offset	Access	Reset value	Description			
TIM1_CCR3	0x3C	RW	0x0000_0000	TIM1 capture/compare register 3			
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CCR3[15:8]							
7	6	5	4	3	2	1	0
CCR3[7:0]							

TIM1 capture/compare register 3 (TIM1_CCR3)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15:0]	RW	<p>CCR3[15:0]: Capture/Compare value</p> <p>If channel CC3 is configured as output: CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR3 register (bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs.</p> <p>The active capture/compare register contains the value to be compared to the counter TIM1_CNT and signalled on OC3 output.</p> <p>If channel CC3 is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3).</p>

10.3.18 TIM1 capture/compare register 4(TIM1_CCR4)

Register	Address offset	Access	Reset value	Description
TIM1_CCR4	0x40	RW	0x0000_0000	TIM1 capture/compare register 4

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CCR4[15:8]							
7	6	5	4	3	2	1	0
CCR4[7:0]							

TIM1 capture/compare register 4(TIM1_CCR4)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15:0]	RW	<p>CCR4[15:0]: Capture/Compare value</p> <p>If channel CC4 is configured as output: CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR4 register (bit OC4PE). Else the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM1_CNT and signalled on OC4 output.</p> <p>If channel CC4 is configured as input: CCR4 is the counter value transferred by the last input capture 4 event (IC4).</p>

10.3.19 TIM1 break and dead-time register(TIM1_BDTR)

Register	Address offset	Access	Reset value	Description
TIM1_BDTR	0x44	RW	0x0000_0000	TIM1 break and dead-time register

Note : As the bits AOE, BKP, BKE, OSSI, OSSR and DTG[7:0] can be write-locked depending on the LOCK configuration, it can be necessary to configure all of them during the first write access to the TIM1_BDTR register.

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

TIM1 break and dead-time register(TIM1_BDTR)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15]	RW	<p>MOE: Main output enable This bit is cleared asynchronously by hardware as soon as the break input is active. It is set by software or automatically depending on the AOE bit. It is acting only on the channels which are configured in output.</p> <p>0: OC and OCN outputs are disabled or forced to idle state. 1: OC and OCN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in TIM1_CCER register).</p> <p>See OC/OCN enable description for more details10.3.10</p>
[14]	RW	<p>AOE: Automatic output enable 0: MOE can be set only by software 1: MOE can be set by software or automatically at the next update event (if the break input is not be active)</p> <p>Note : <i>This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIM1_BDTR register)</i></p>
[13]	RW	<p>BKP: Break polarity 0: Break input BRK is active low 1: Break input BRK is active high</p> <p>Note : 1. <i>This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIM1_BDTR register).</i> 2. <i>Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.</i></p>

[12]	RW	<p>BKE: Break enable 0: Break inputs (BRK and CSS clock failure event) disabled 1; Break inputs (BRK and CSS clock failure event) enabled</p> <p>Note : 1. This bit cannot be modified when LOCK level 1 has been programmed (LOCK bits in TIM1_BDTR register). 2. Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.</p>
[11]	RW	<p>OSSR: Off-state selection for Run mode This bit is used when MOE=1 on channels having a complementary output which are configured as outputs. OSSR is not implemented if no complementary output is implemented in the timer. See OC/OCN enable description for more details 10.3.10 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal=0). 1: When inactive, OC/OCN outputs are enabled with their inactive level as soon as CCxE=1 or CCxNE=1. Then, OC/OCN enable output signal=1</p> <p>Note : This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIM1_BDTR register)</p>
[10]	RW	<p>OSSI: Off-state selection for Idle mode This bit is used when MOE=0 on channels configured as outputs. See OC/OCN enable description for more details 10.3.10 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal=0). 1: When inactive, OC/OCN outputs are forced first with their idle level as soon as CCxE=1 or CCxNE=1. OC/OCN enable output signal=1)</p> <p>Note : This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIM1_BDTR register).</p>
[9:8]	RW	<p>LOOK[1:0]: Lock Configuration These bits offer a write protection against software errors. 00: LOCK OFF - No bit is write protected. 01: LOCK Level 1 = DTG bits in TIM1_BDTR register, OISx and OISxN bits in TIM1_CR2 register and BKE/BKP/AOE bits in TIM1_BDTR register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIM1_CCER register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in TIM1_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written.</p> <p>Note : The LOCK bits can be written only once after the reset. Once the TIM1_BDTR register has been written, their content is frozen until the next reset.</p>

[7:0]	RW	<p>DTG[7:0]: Dead-time generator setup</p> <p>This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration.</p> <p>DTG[7:5]=0xx => DT=DTG[7:0]x T_dtg with T_dtg = T_DTS.</p> <p>DTG[7:5]=10x => DT=(64+DTG[5:0])xT_dtg with T_dtg =2xT_DTS.</p> <p>DTG[7:5]=110 => DT=(32+DTG[4:0])xT_dtg with T_dtg =8xT_DTS.</p> <p>DTG[7:5]=111 => DT=(32+DTG[4:0])xT_dtg with T_dtg =16xT_DTS.</p> <p>Example if T DTS =125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps, 16 us to 31750 ns by 250 ns steps, 32 us to 63us by 1 us steps, 64 us to 126 us by 2 us steps</p> <p>Note : <i>This bit-field can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register).</i></p>
-------	----	--

11 General-purpose(TIM2)

11.1 TIM2 introduction

The general-purpose timers consist of a 16-bit auto-reload counter driven by a programmable prescaler.

They may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare and PWM).

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

The timers are completely independent, and do not share any resources. They can be synchronized together as described in section:11.3.14.

11.2 TIM2 main features

- 6-bit up, down, up/down auto-reload counter.
- 16-bit programmable prescaler used to divide (also “on the fly”) the counter clock frequency by any factor between 1 and 65536.
- Up to 4 independent channels for:
 - Input capture
 - Output compare
 - PWM generation (Edge- and Center-aligned modes)
 - One-pulse mode output
- Synchronization circuit to control the timer with external signals and to interconnect several timers.
- Interrupt generation on the following events:
 - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
 - Trigger event (counter start, stop, initialization or count by internal/external trigger)
 - Input capture
 - Output compare
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes
- Trigger input for external clock or cycle-by-cycle current management

11.3 TIM2 functional description

11.3.1 Time-base unit

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

- Counter Register (TIM2_CNT)
- Prescaler Register (TIM2_PSC):
- Auto-Reload Register (TIM2_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIM2_CR1 register. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIM2_CR1 register. It can also be generated by software. The generation of the update event is described in detail for each configuration.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIM2_CR1 register is set (refer also to the slave mode controller description to get more details on counter enabling).

Note that the actual counter enable signal CNT_EN is set 1 clock cycle after CEN.

11.3.2 Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIM2_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

Sse Figure 11.3-1and Figure11.3-2,

Fig 11.3-1 Counter timing diagram with prescaler division change from 1 to 2

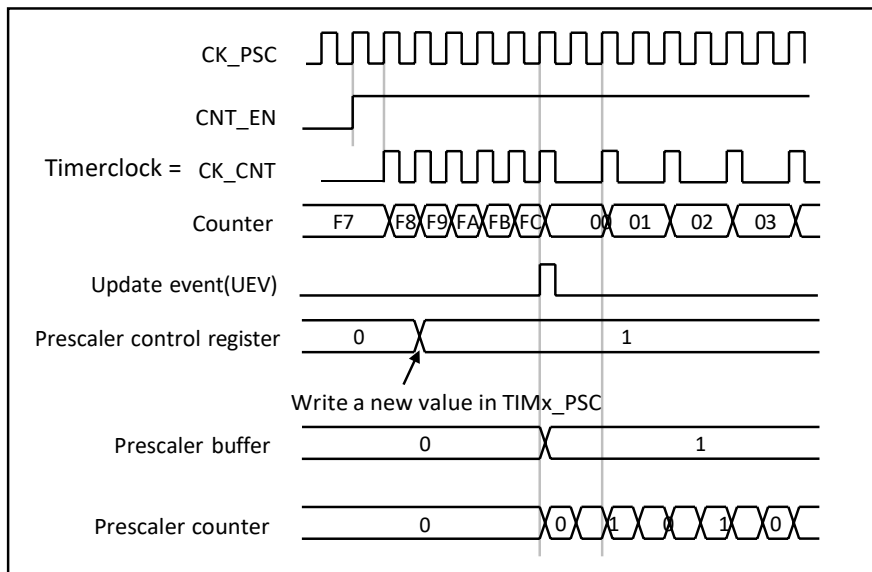
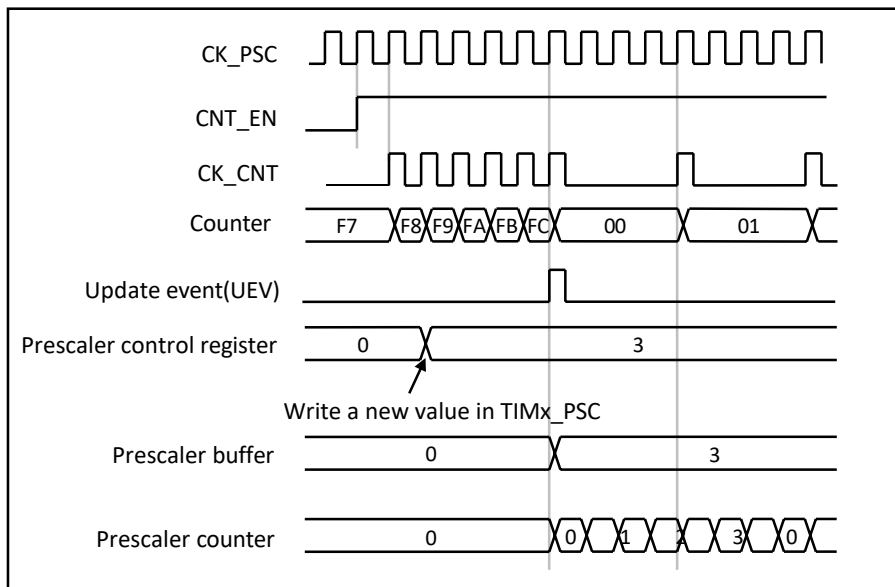


Fig 11.3-2 Counter timing diagram with prescaler division change from 1 to 4



11.3.3 Counter modes

11.3.3.1 Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the TIM2_ARR register), then restarts from 0 and generates a counter overflow event.

An Update event can be generated at each counter overflow or by setting the UG bit in the TIM2_EGR register (by software or by using the slave mode controller).

The UEV event can be disabled by software by setting the UDIS bit in TIM2_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update

event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIM2_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag . This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM2_SR register) is set (depending on the URS bit):

- The buffer of the prescaler is reloaded with the preload value (content of the TIM2_PSC register)
- The auto-reload shadow register is updated with the preload value (TIM2_ARR)

The following figures show some examples of the counter behavior for different clock frequencies when TIM2_ARR=0x36.

Fig 11.3-3 Counter timing diagram, internal clock divided by 1

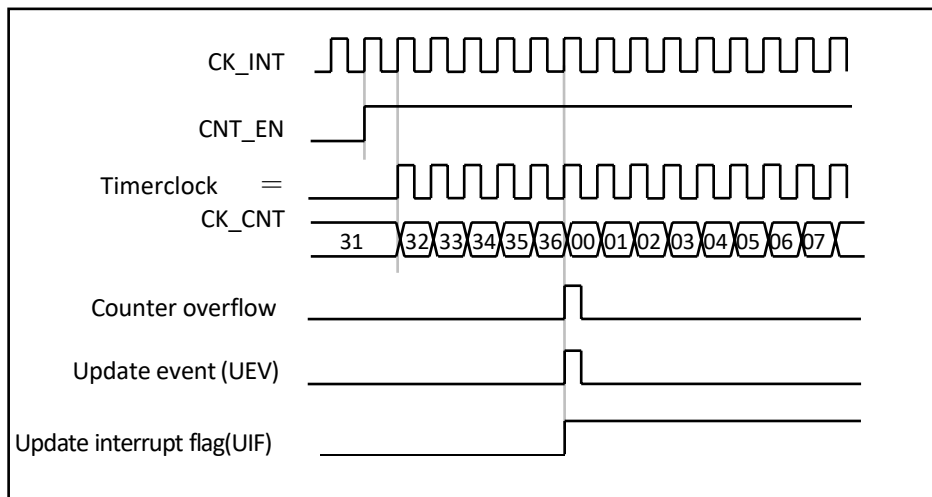


Fig 11.3-4 Counter timing diagram, internal clock divided by 2

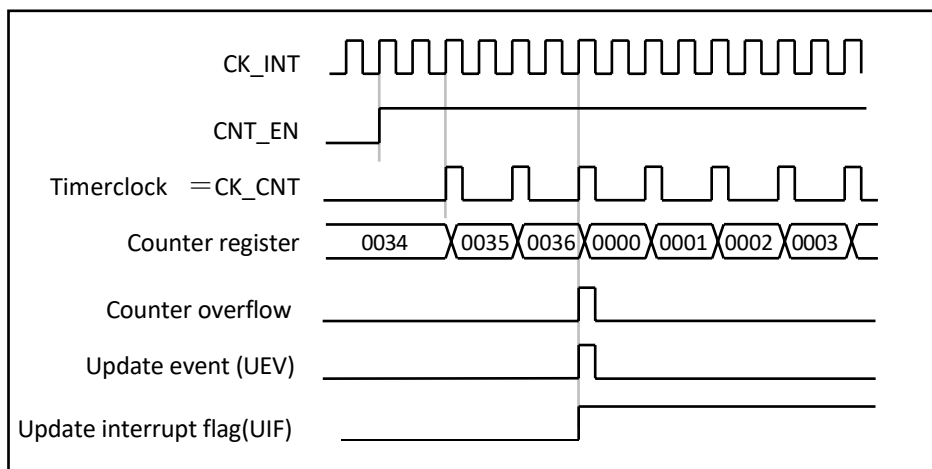


Fig 11.3-5 Counter timing diagram, internal clock divided by 4

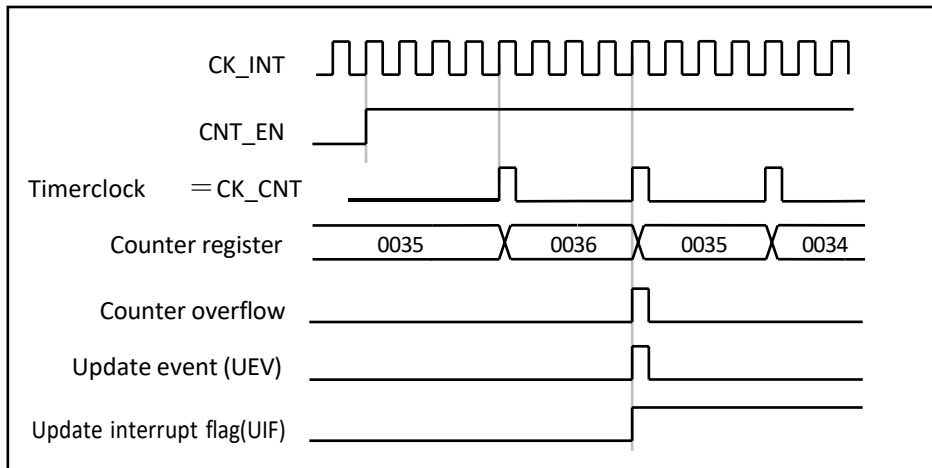


Fig 11.3-6 Counter timing diagram, internal clock divided by N

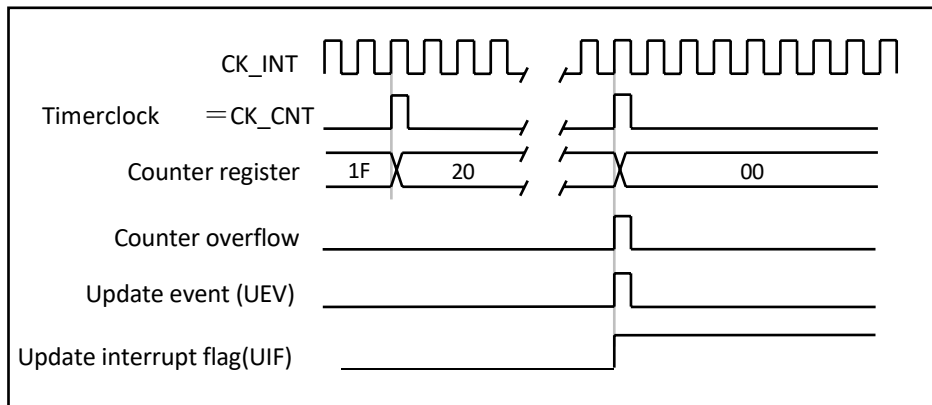


Fig 11.3-7 Counter timing diagram, Update event when ARPE=0 (TIM2_ARR not preloaded)

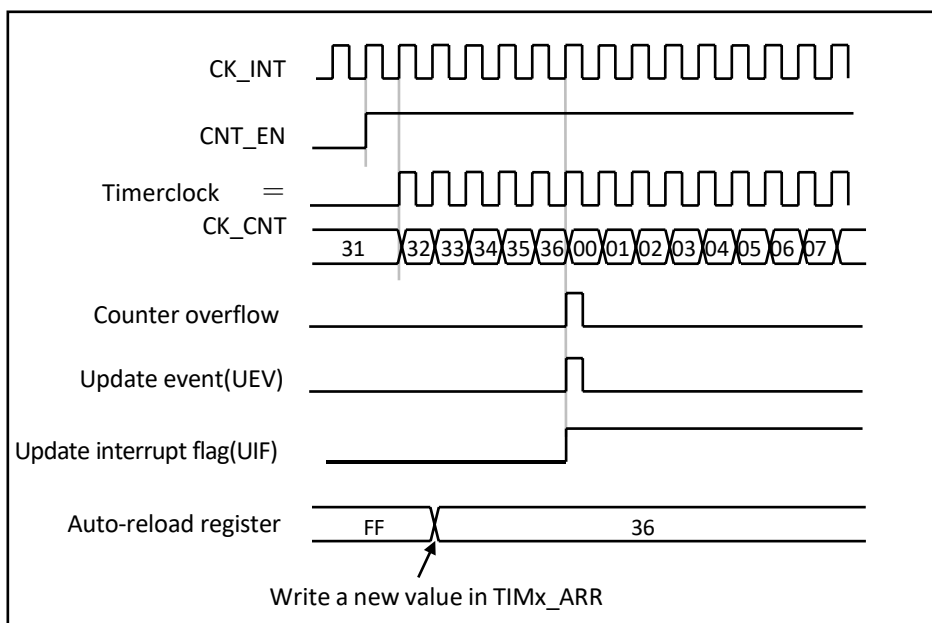
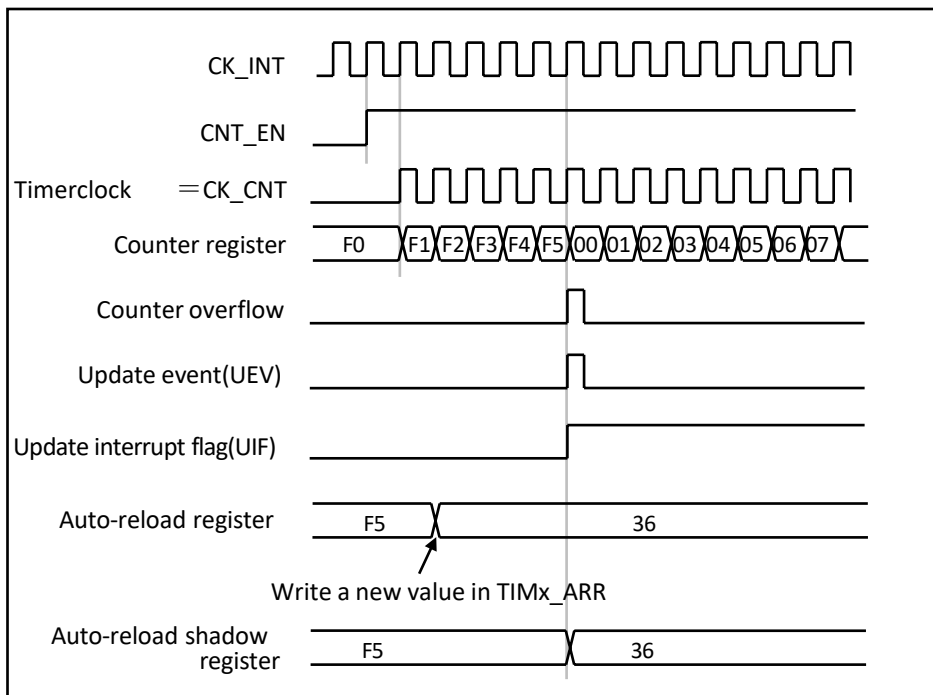


Fig 11.3-8 Counter timing diagram, Update event when ARPE=1 (TIM2_ARR preloaded)



11.3.3.2 Downcounting mode

In downcounting mode, the counter counts from the auto-reload value (content of the TIM2_ARR register) down to 0, then restarts from the auto-reload value and generates a counter underflow event.

An Update event can be generated at each counter underflow or by setting the UG bit in the TIM2_EGR register (by software or by using the slave mode controller)

The UEV update event can be disabled by software by setting the UDIS bit in TIM2_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescale rate doesn't change).

In addition, if the URS bit (update request selection) in TIM2_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag. This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM2_SR register) is set (depending on the URS bit):

- The buffer of the prescaler is reloaded with the preload value (content of the TIM2_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIM2_ARR register). Note that the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

The following figures show some examples of the counter behavior for different clock frequencies when TIM2_ARR=0x36.

Fig 11.3-9 Counter timing diagram, internal clock divided by 1

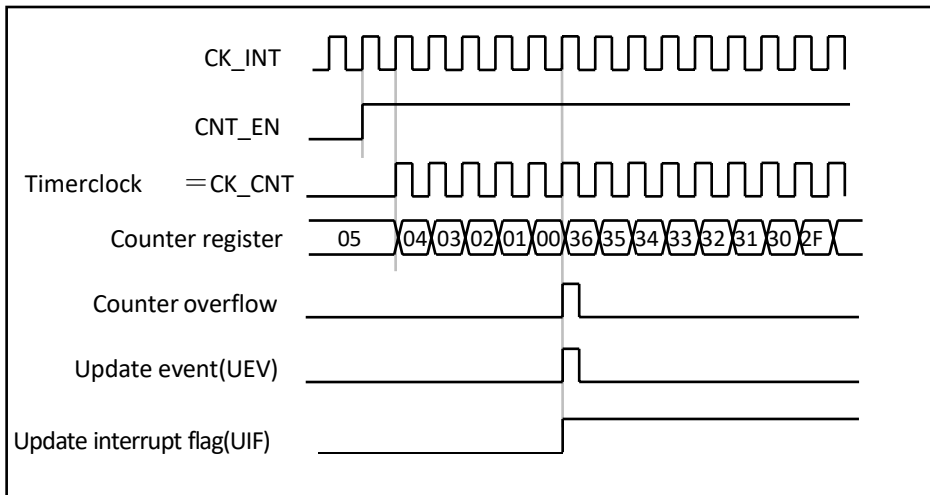


Fig 11.3-10 Counter timing diagram, internal clock divided by 2

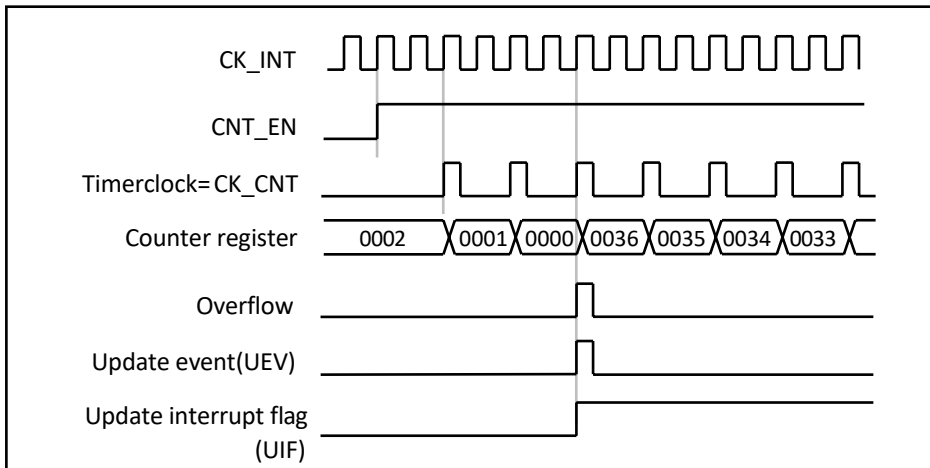


Fig 11.3-11 Counter timing diagram, internal clock divided by 4

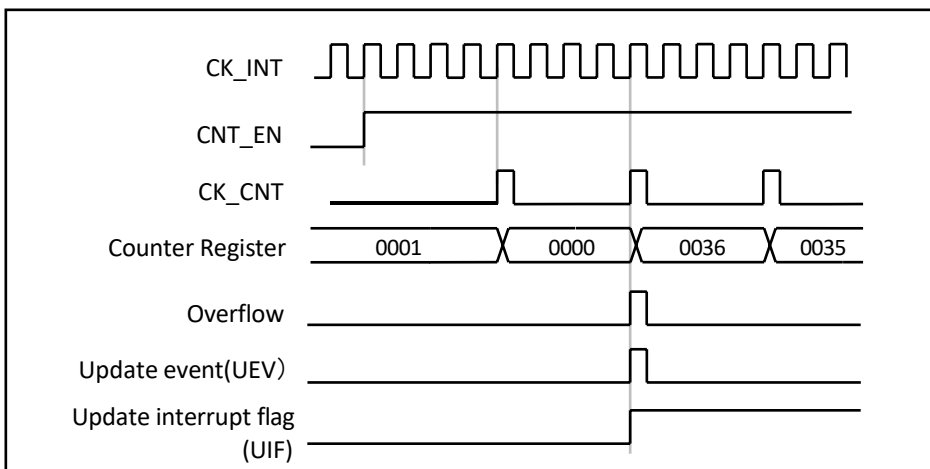


Fig 11.3-12 Counter timing diagram, internal clock divided by N

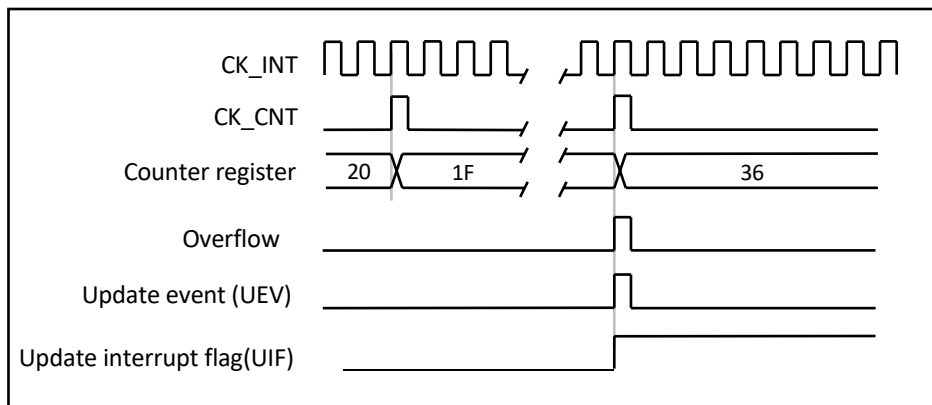
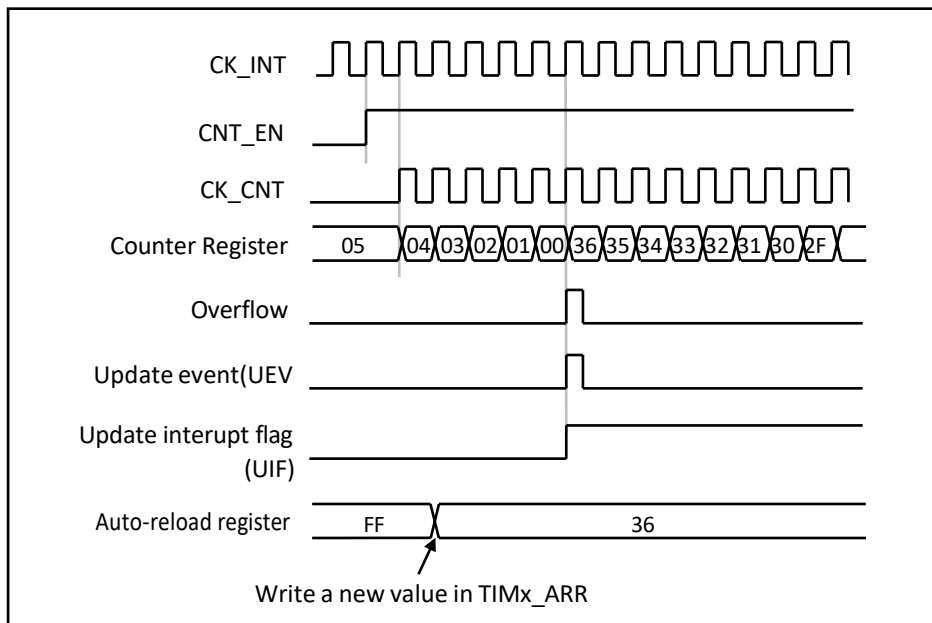


Fig 11.3-13 Counter timing diagram, Update event



11.3.3.3 Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the TIM2_ARR register) –1, generates a counter overflow event, then counts from the auto-reload value down to 1 and generates a counter underflow event. Then it restarts counting from 0.

In this mode, the direction bit (DIR from TIM2_CR1 register) cannot be written. It is updated by hardware and gives the current direction of the counter. The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIM2_EGR register (by software or by using the slave mode controller) also generates an update event. In this case, the counter restarts counting from 0, as well as the counter of the prescaler.

The UEV update event can be disabled by software by setting the UDIS bit in TIM2_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter continues

counting up and down, based on the current auto-reload value. In addition, if the URS bit (update request selection) in TIM2_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag . This is to avoid generating both update and capture interrupt when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM2_SR register) is set (depending on the URS bit):

- The buffer of the prescaler is reloaded with the preload value (content of the TIM2_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIM2_ARR register).

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

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

Fig 11.3-14 Counter timing diagram, internal clock divided by 1, TIM2_ARR=0x6

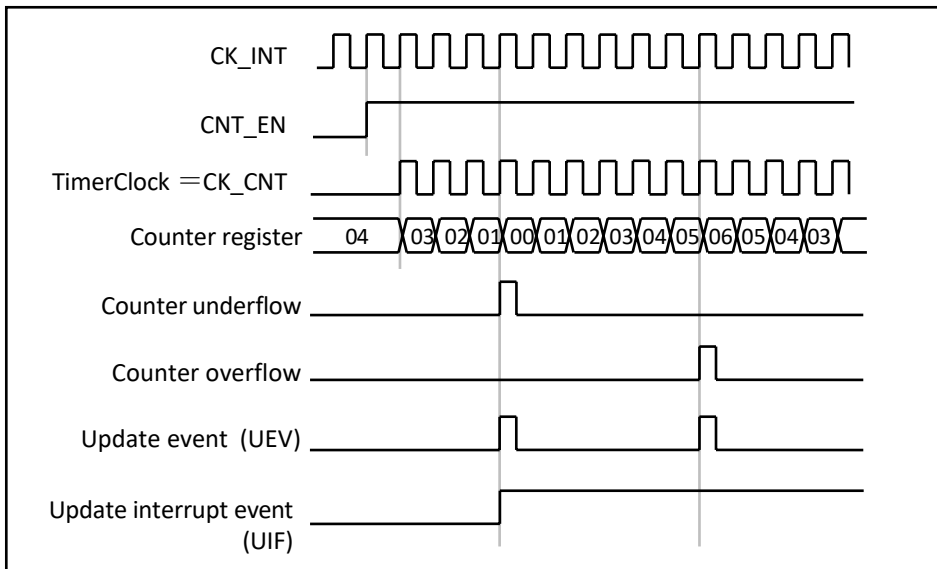


Fig 11.3-15 Counter timing diagram, internal clock divided by 2

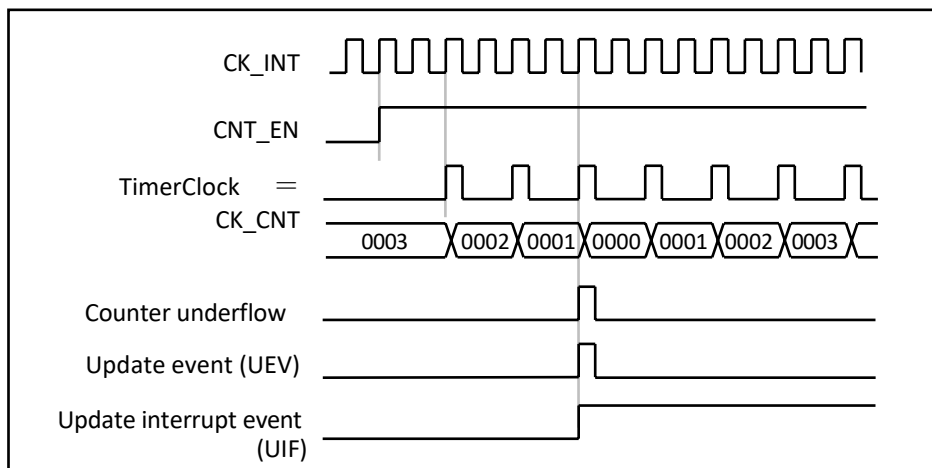


Fig 11.3-16 Counter timing diagram, internal clock divided by 4, TIM2_ARR=0x36

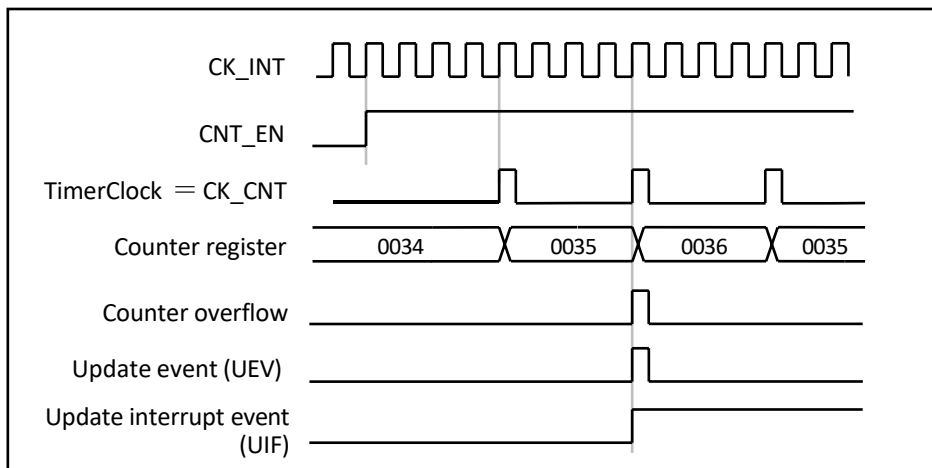


Fig 11.3-17 Counter timing diagram, internal clock divided by N

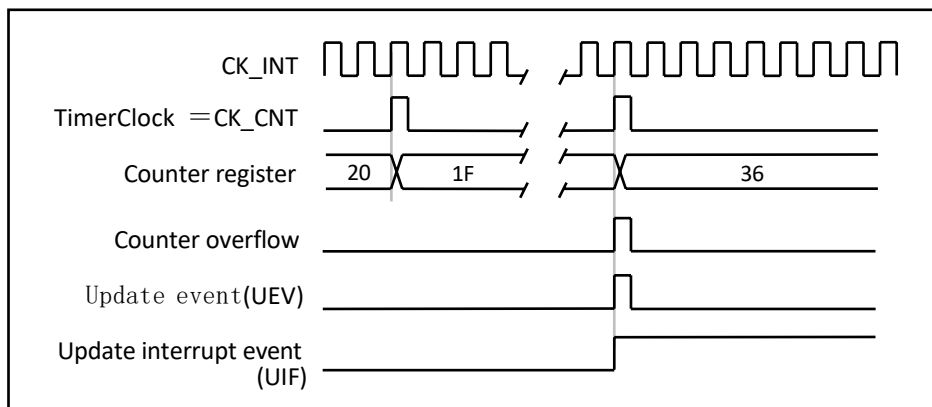


Fig 11.3-18 Counter timing diagram, Update event with ARPE=1 (counter underflow)

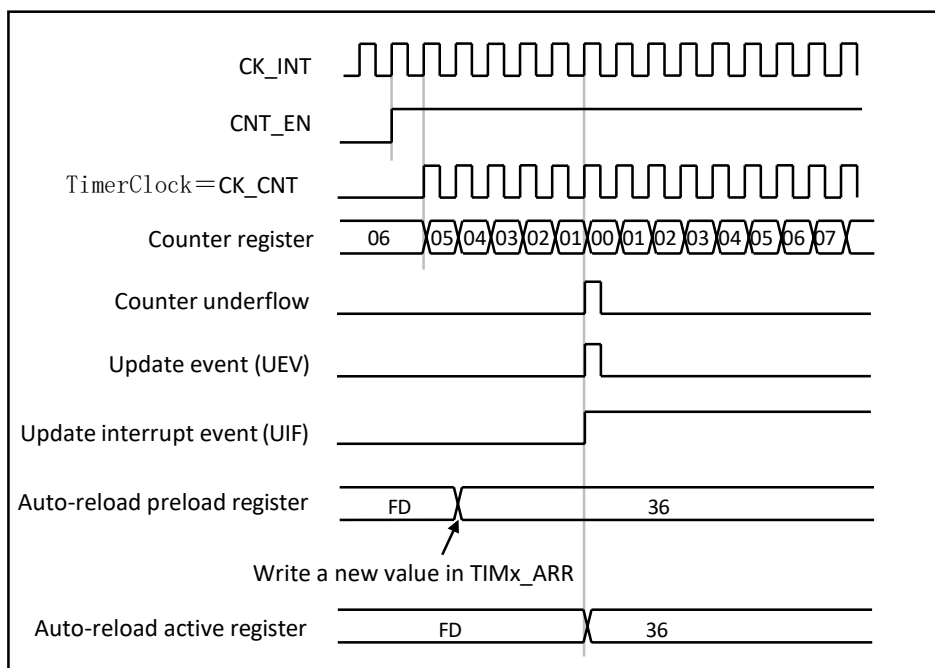
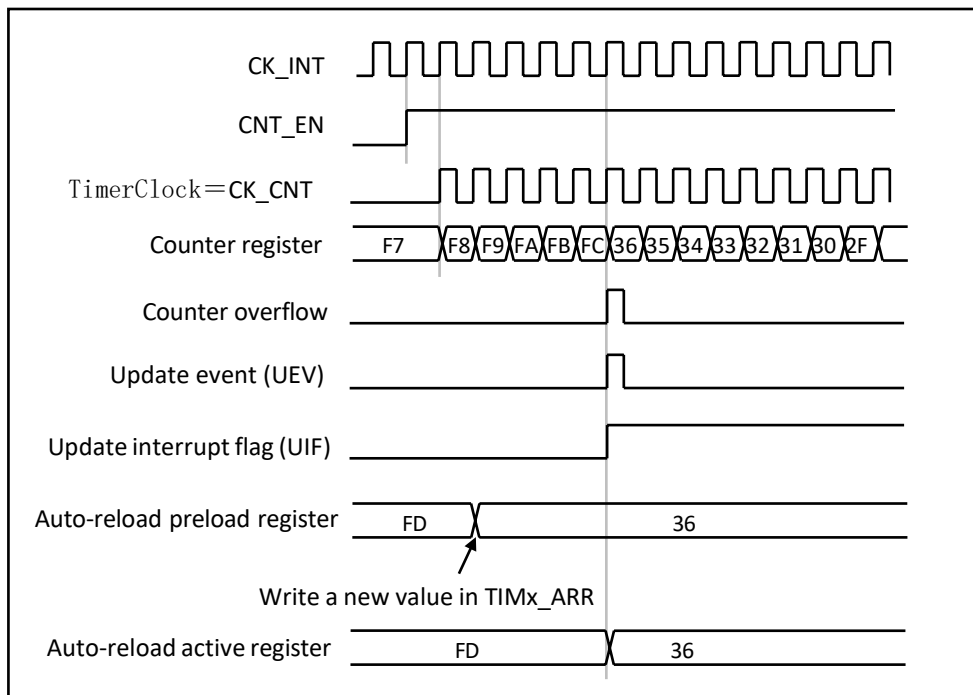


Fig 11.3-19 Counter timing diagram, Update event with ARPE=1 (counter overflow)



11.3.4 Clock selection

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

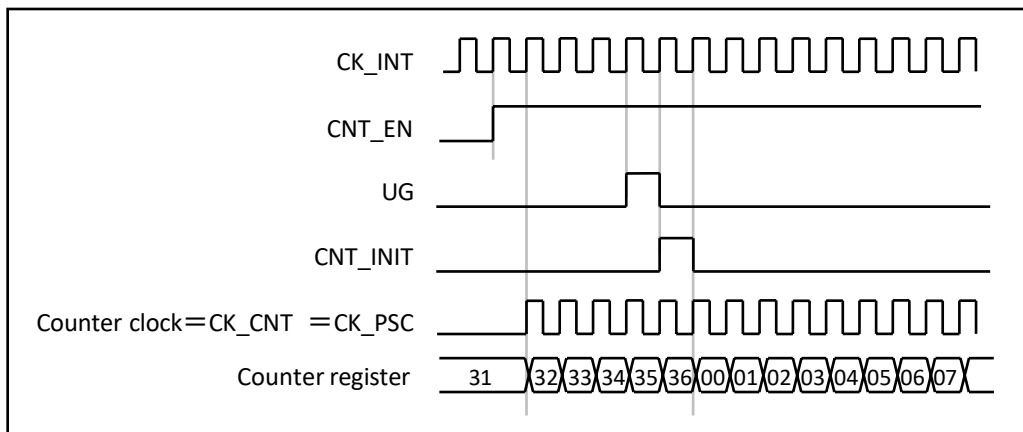
- Internal clock (CK_INT)
- External clock mode1: external input pin (TIx)
- External clock mode2: external trigger input (ETR).
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, Timer1 can be configured to act as a prescaler for Timer 2. Refer to Using one timer as prescaler for another timer for more details.

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS=000 in the TIM2_SMCR register), then the CEN, DIR (in the TIM2_CR1 register) and UG bits (in the TIM2_EGR register) are actual control bits and can be changed only by software (except UG which remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

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

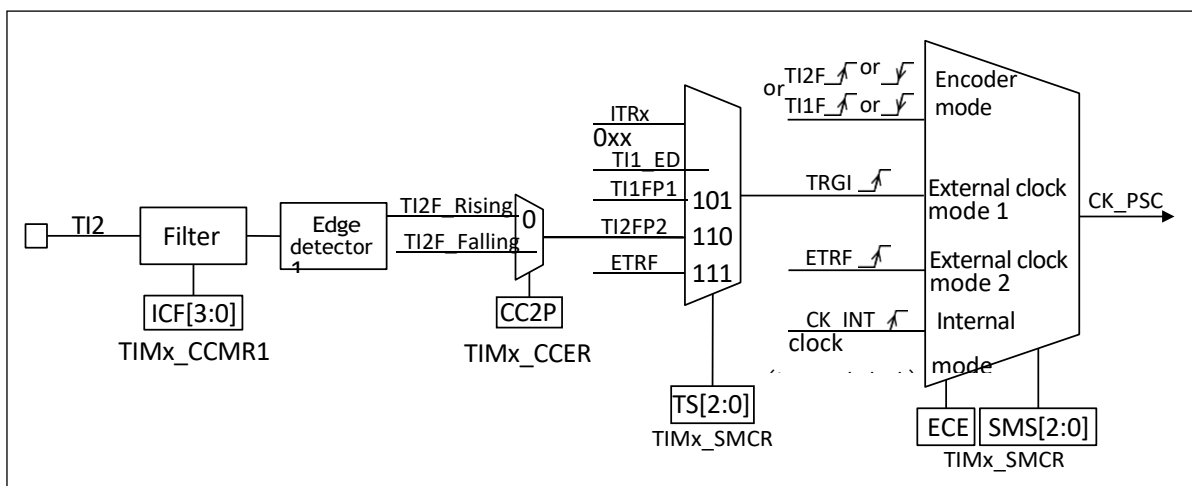
Fig 11.3-20 Control circuit in normal mode, internal clock divided by 1



External clock source mode 1

This mode is selected when SMS=111 in the TIM2_SMCR register. The counter can count at each rising or falling edge on a selected input.

Fig 11.3-21 T12 external clock connection example



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

- Configure channel 2 to detect rising edges on the TI2 input by writing CC2S= ‘01 in the TIM2_CCMR1 register.
- Configure the input filter duration by writing the IC2F[3:0] bits in the TIM2_CCMR1 register (if no filter is needed, keep IC2F=0000).

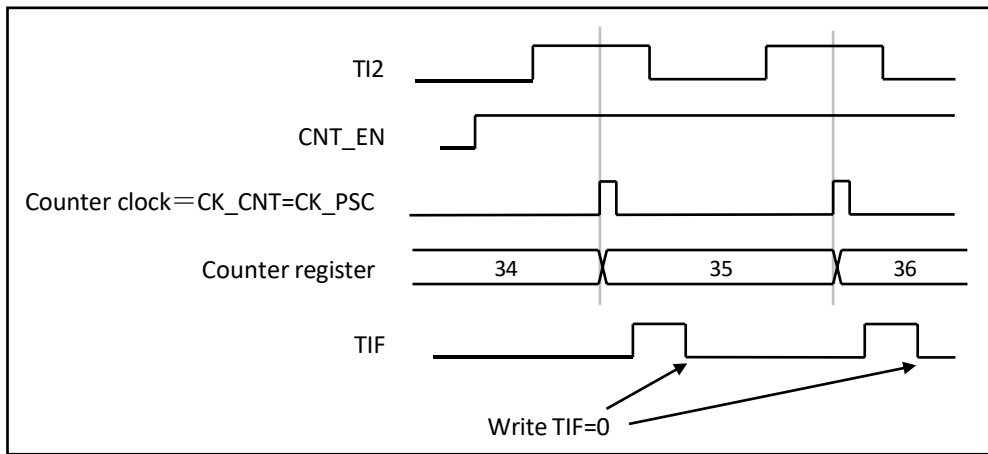
Note: The capture prescaler is not used for triggering, so there’s no need to configure it.

- Select rising edge polarity by writing CC2P=0 in the TIM2_CCER register.
- Configure the timer in external clock mode 1 by writing SMS=111 in the TIM2_SMCR register.
- Select TI2 as the input source by writing TS=110 in the TIM2_SMCR register.
- Enable the counter by writing CEN=1 in the TIM2_CR1 register.

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge on TI2 and the actual clock of the counter is due to the resynchronization circuit on TI2 input.

Fig 11.3-22 T12 control circuit in external clock mode 1

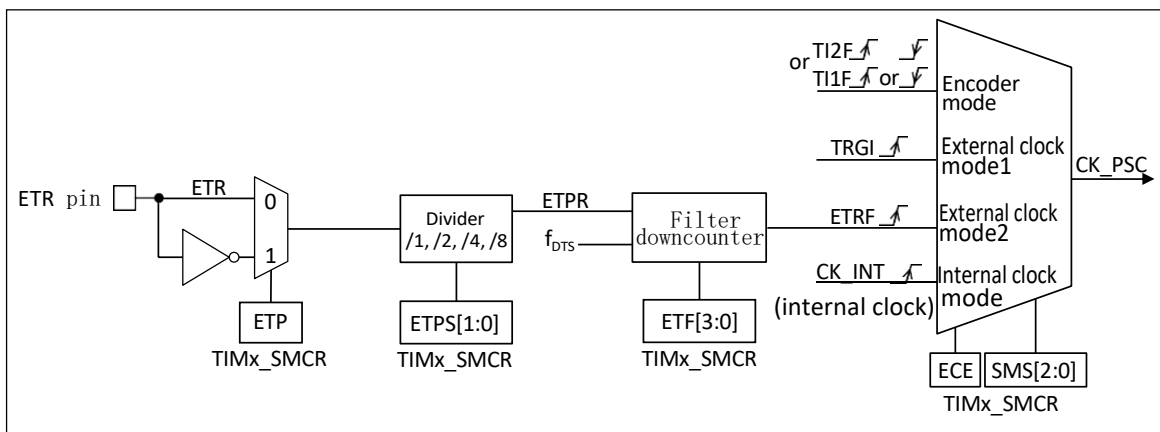


External clock source mode 2

This mode is selected by writing ECE=1 in the TIM2_SMCR register.

The counter can count at each rising or falling edge on the external trigger input ETR. 11.3-23 gives an overview of the external trigger input module:

Fig 11.3-23 T12 external trigger input module



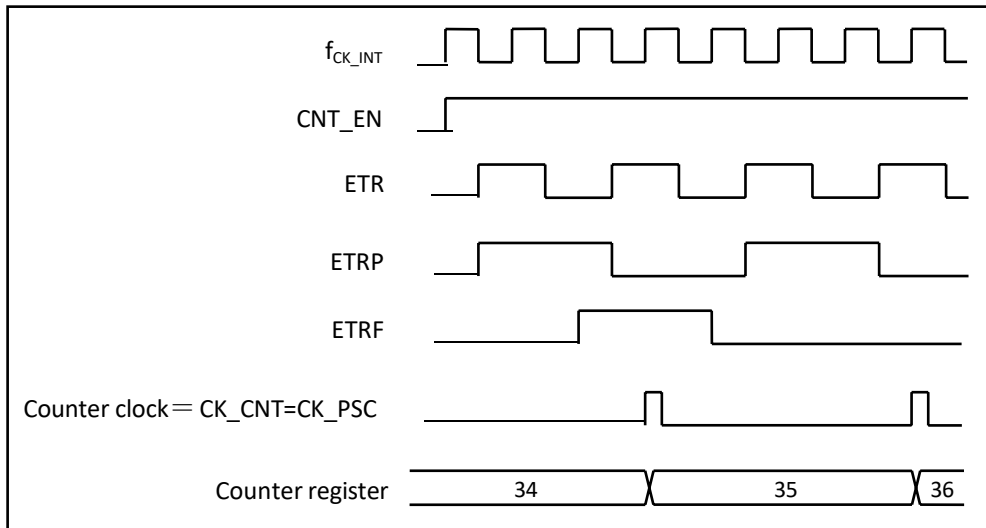
For example, to configure the upcounter to count each 2 rising edges on ETR, use the following procedure:

- As no filter is needed in this example, write ETF[3:0]=0000 in the TIM2_SMCR register.
- Set the prescaler by writing ETPS[1:0]=01 in the TIM2_SMCR register
- Select rising edge detection on the ETR pin by writing ETP=0 in the TIM2_SMCR register
- Enable external clock mode 2 by writing ECE=1 in the TIM2_SMCR register.
- Enable the counter by writing CEN=1 in the TIM2_CR1 register.

The counter counts once each 2 ETR rising edges.

The delay between the rising edge on ETR and the actual clock of the counter is due to the resynchronization circuit on the ETRP signal

Fig 11.3-24 Control circuit in external clock mode 2



Capture/compare channels

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

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

Fig 11.3-25 Capture/compare channel (example: channel 1 input stage)

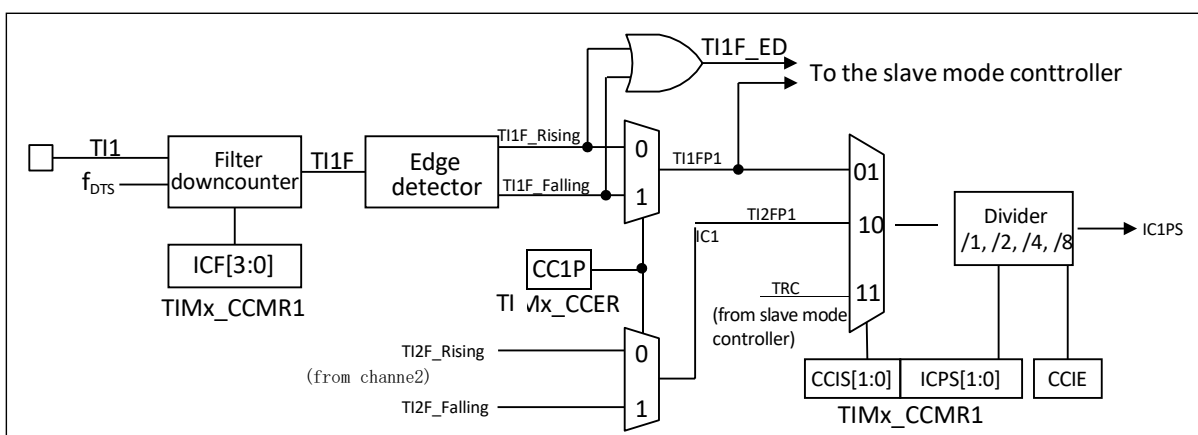
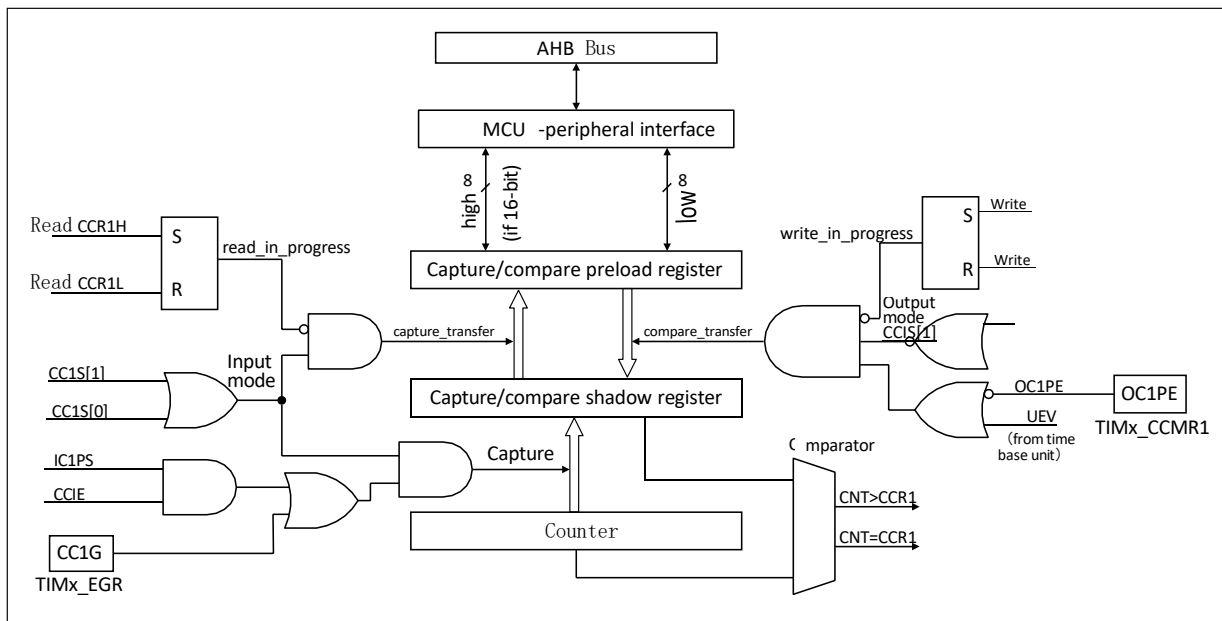
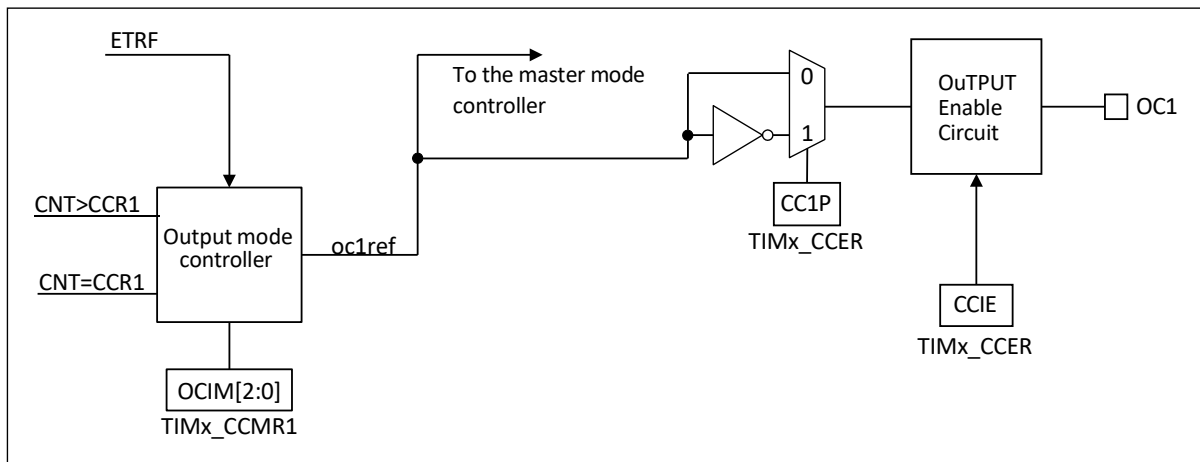


Fig 11.3-26 Capture/compare channel (example: channel 1 input stage)



The output stage generates an intermediate waveform which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

Fig 11.3-27 Capture/compare channel (example: channel 1 input stage)



The capture/compare module is made of one preload register and one shadow register. Write and read always access the preload register.

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

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

Input capture mode

In Input capture mode, the Capture/Compare Registers (TIM2_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCXIF flag (TIM2_SR register) is set and an interrupt can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIM2_SR register) is set. CCxIF can be cleared by software by writing it to 0 or by reading the captured data stored in the TIM2_CCRx register. CCxOF is cleared when written to 0.

The following example shows how to capture the counter value in TIM2_CCR1 when TI1 input rises. To do this, use the following procedure:

1. Select the active input: TIM2_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIM2_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIM2_CCR1 register becomes read-only.
2. Program the needed input filter duration with respect to the signal connected to the timer (by programming the ICxF bits in the TIM2_CCMRx register if the input is one of the TIx inputs). Let's imagine that, when toggling, the input signal is not stable during at most five internal clock cycles. We must program a filter duration longer than these five clock cycles. We can validate a transition on TI1 when eight consecutive samples with the new level have been detected (sampled at f_{DTS} frequency). Then write IC1F bits to 0011 in the TIM2_CCMR1 register.
3. Select the edge of the active transition on the TI1 channel by writing the CC1P bit to 0 in the TIM2_CCER register (rising edge in this case).
4. Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to 00 in the TIM2_CCMR1 register).
5. Enable capture from the counter into the capture register by setting the CC1E bit in the TIM2_CCER register.
6. If needed, enable the related interrupt request by setting the CC1IE bit in the TIM2_DIER register.

When an input capture occurs:

- The TIM2_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.

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

Note : IC interrupt can be generated by software by setting the corresponding CCxG bit in the TIM2_EGR Register.

11.3.5 PWM input mode

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

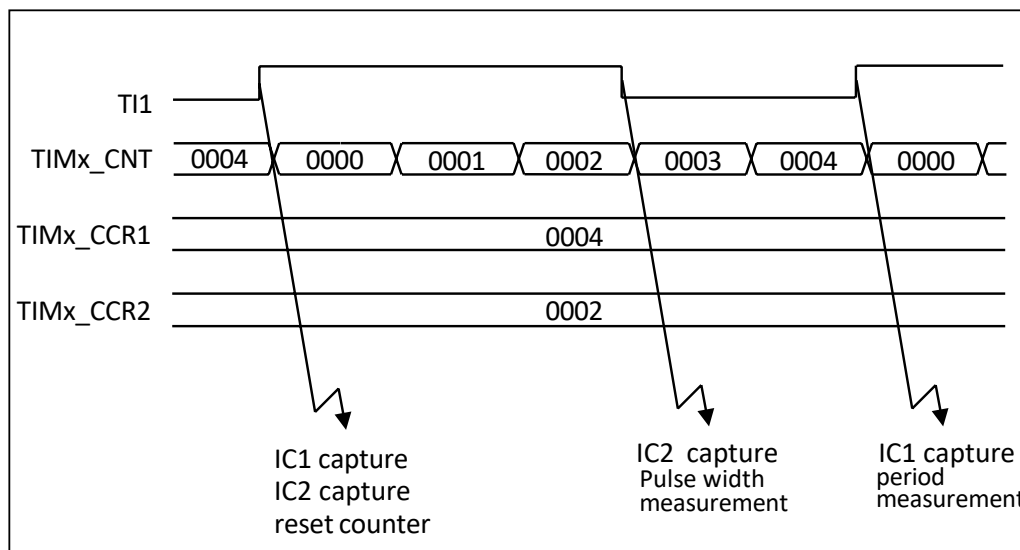
1. Two ICx signals are mapped on the same TIx input.
2. These 2 ICx signals are active on edges with opposite polarity.

3. One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode

For example, the user can measure the period (in TIM2_CCR1 register) and the duty cycle (in TIM2_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):

1. Select the active input for TIM2_CCR1: write the CC1S bits to 01 in the TIM2_CCMR1 register (TI1 selected).
2. Select the active polarity for TI1FP1 (used both for capture in TIM2_CCR1 and counter clear): write the CC1P to '0' (active on rising edge).
3. Select the active input for TIM2_CCR2: write the CC2S bits to 10 in the TIM2_CCMR1 register (TI1 selected).
4. Select the active polarity for TI1FP2 (used for capture in TIM2_CCR2): write the CC2P bit to '1' (active on falling edge).
5. Select the valid trigger input: write the TS bits to 101 in the TIM2_SMCR register (TI1FP1 selected).
6. Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIM2_SMCR register.
7. Enable the captures: write the CC1E and CC2E bits to '1' in the TIM2_CCER register.

Fig 11.3-28 PWM input mode timing



Note: The PWM input mode can be used only with the TIM2_CH1/TIM2_CH2 signals due to the fact that only TI1FP1 and TI2FP2 are connected to the slave mode controller.

11.3.6 Forced output mode

In output mode (CCxS bits = 00 in the TIM2_CCMRx register), each output compare signal (OCxREF and then OCx) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (ocxref/OCx) to its active level, the user just needs to write 101

in the OCxM bits in the corresponding TIM2_CCMRx register. Thus ocxref is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit

e.g.: CCxP=0 (OCx active high) => OCx is forced to high level.

ocxref signal can be forced low by writing the OCxM bits to 100 in the TIM2_CCMRx register.

Anyway, the comparison between the TIM2_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt can be sent accordingly. This is described in the 11.3.7 Output Compare Mode section.

11.3.7 Output compare mode

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

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

1. Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIM2_CCMRx register) and the output polarity (CCxP bit in the TIM2_CCER register). The output pin can keep its level (OCxM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
2. Sets a flag in the interrupt status register (CCxIF bit in the TIM2_SR register).
3. Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIM2_DIER register).

The TIM2_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIM2_CCMRx register.

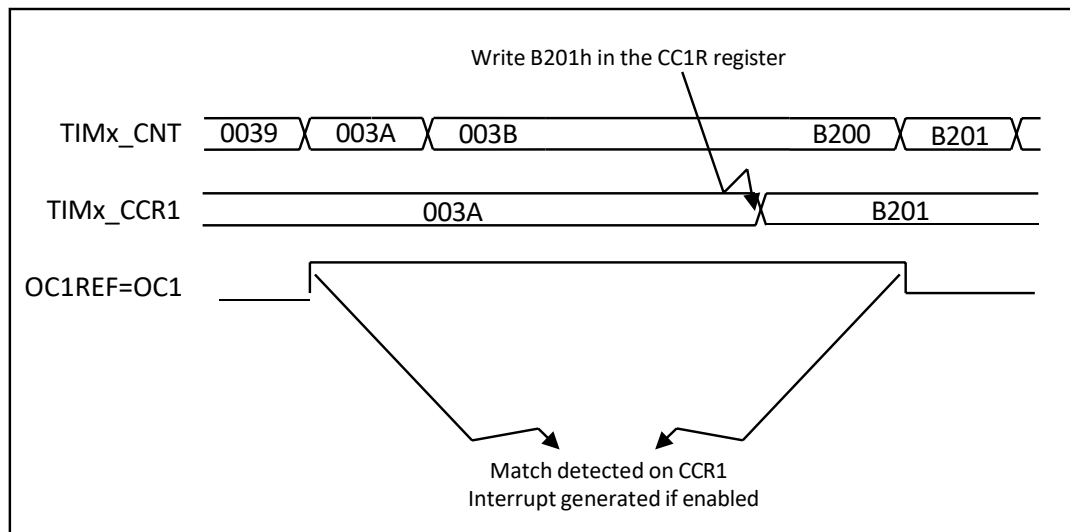
In output compare mode, the update event UEV has no effect on ocxref and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One-pulse mode).

Procedure:

- Select the counter clock (internal, external, prescaler).
- Write the desired data in the TIM2_ARR and TIM2_CCRx registers.
- Set the CCxIE and/or CCxDE bits if an interrupt request is to be generated.
- Select the output mode. For example, the user must write OCxM=011, OCxPE=0, CCxP=0 and CCxE=1 to toggle OCx output pin when CNT matches CCRx, CCRx preload is not used, OCx is enabled and active high.
- Enable the counter by setting the CEN bit in the TIM2_CR1 register.

The TIM2_CCRx register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE=0, else TIM2_CCRx shadow register is updated only at the next update event UEV). An example is given in 11.3-29.

Fig 11.3-29 Output compare mode, toggle on OC1



11.3.8 PWM mode

Pulse width modulation mode allows generating a signal with a frequency determined by the value of the TIM2_ARR register and a duty cycle determined by the value of the TIM2_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing 110 (PWM mode 1) or '111 (PWM mode 2) in the OCxM bits in the TIM2_CCMRx register. The user must enable the corresponding preload register by setting the OCxPE bit in the TIM2_CCMRx register, and eventually the auto-reload preload register by setting the ARPE bit in the TIM2_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, the user has to initialize all the registers by setting the UG bit in the TIM2_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIM2_CCER register. It can be programmed as active high or active low. OCx output is enabled by the CCxE bit in the TIM2_CCER register. Refer to the TIM2_CCERx register description for more details.

In PWM mode (1 or 2), TIM2_CNT and TIM2_CCRx are always compared to determine whether $TIM2_CCR_x \leq TIM2_CNT$ or $TIM2_CNT \leq TIM2_CCR_x$ (depending on the direction of the counter). However, to comply with the ETRF (OCREF can be cleared by an external event through the ETR signal until the next PWM period), the OCREF signal is asserted only:

1. When the result of the comparison changes
2. When the output compare mode (OCxM bits in TIM2_CCMRx register) switches from the "frozen" configuration (no comparison, OCxM='000) to one of the PWM modes (OCxM='110 or '111).

This forces the PWM by software while the timer is running.

The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIM2_CR1 register.

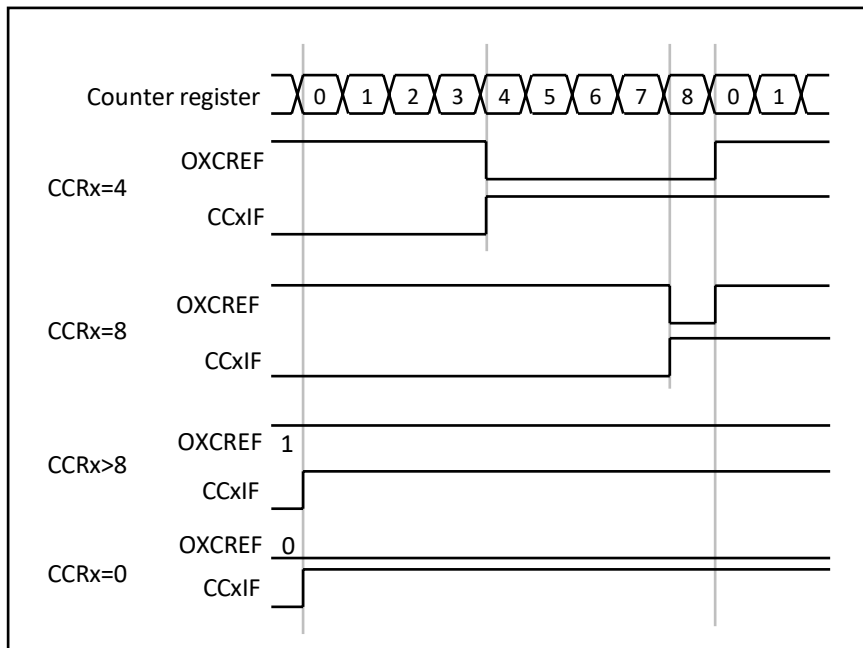
PWM edge-aligned mode

Upcounting configuration

Upcounting is active when the DIR bit in the TIM2_CR1 register is low.

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as $TIM2_CNT < TIM2_CCRx$ else it becomes low. If the compare value in $TIM2_CCRx$ is greater than the auto-reload value (in $TIM2_ARR$) then OCxREF is held at '1. If the compare value is 0 then OCxREF is held at '0. Figure 11.3-30 shows some edge-aligned PWM waveforms in an example where $TIM2_ARR=8$.

Fig 11.3-30 Edge-aligned PWM waveforms (ARR=8)



Downcounting configuration

Downcounting is active when DIR bit in TIM2_CR1 register is high.

In PWM mode 1, the reference signal ocxref is low as long as $TIM2_CNT > TIM2_CCRx$ else it becomes high. If the compare value in $TIM2_CCRx$ is greater than the auto-reload value in $TIM2_ARR$, then ocxref is held at '1. 0% PWM is not possible in this mode.

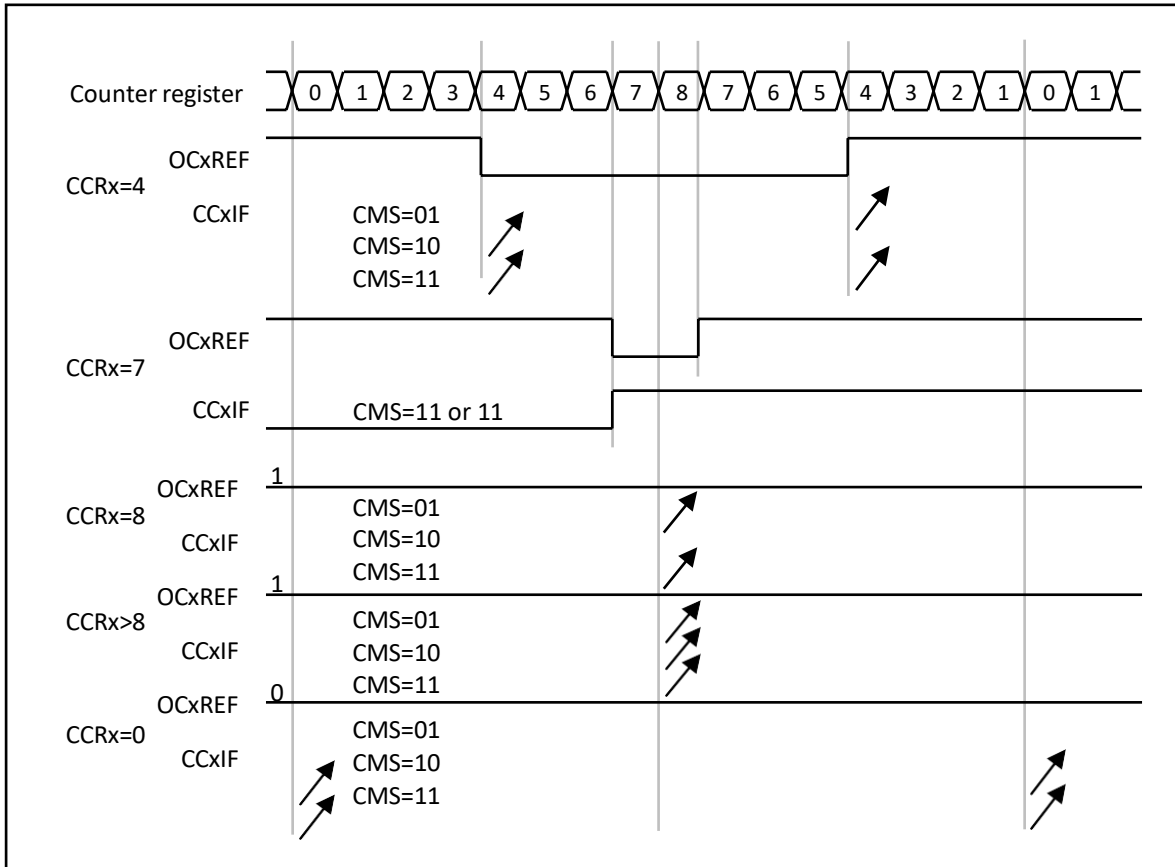
PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIM2_CR1 register are different from '00 (all the remaining configurations having the same effect on the ocxref/OCx signals). The compare flag is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the TIM2_CR1 register is updated by hardware and must not be changed by software.

Figure 11.3-31 shows some center-aligned PWM waveforms in an example where:

1. TIM2_ARR=8,
2. PWM mode is the PWM mode 1,
3. The flag is set when the counter counts down corresponding to the center-aligned mode 1 selected for CMS=01 in TIM2_CR1 register.

Fig 11.3-31 Center-aligned PWM waveforms (ARR=8)



Hints on using center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIM2_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:
 - The direction is not updated if the user writes a value in the counter that is greater than the auto-reload value (TIM2_CNT>TIM2_ARR). For example, if the counter was counting up, it continues to count up.
 - The direction is updated if the user writes 0 or write the TIM2_ARR value in the counter but no Update Event UEV is generated.
- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIM2_EGR register) just before starting the counter and not to write the counter while it is running.

11.3.9 One-pulse mode

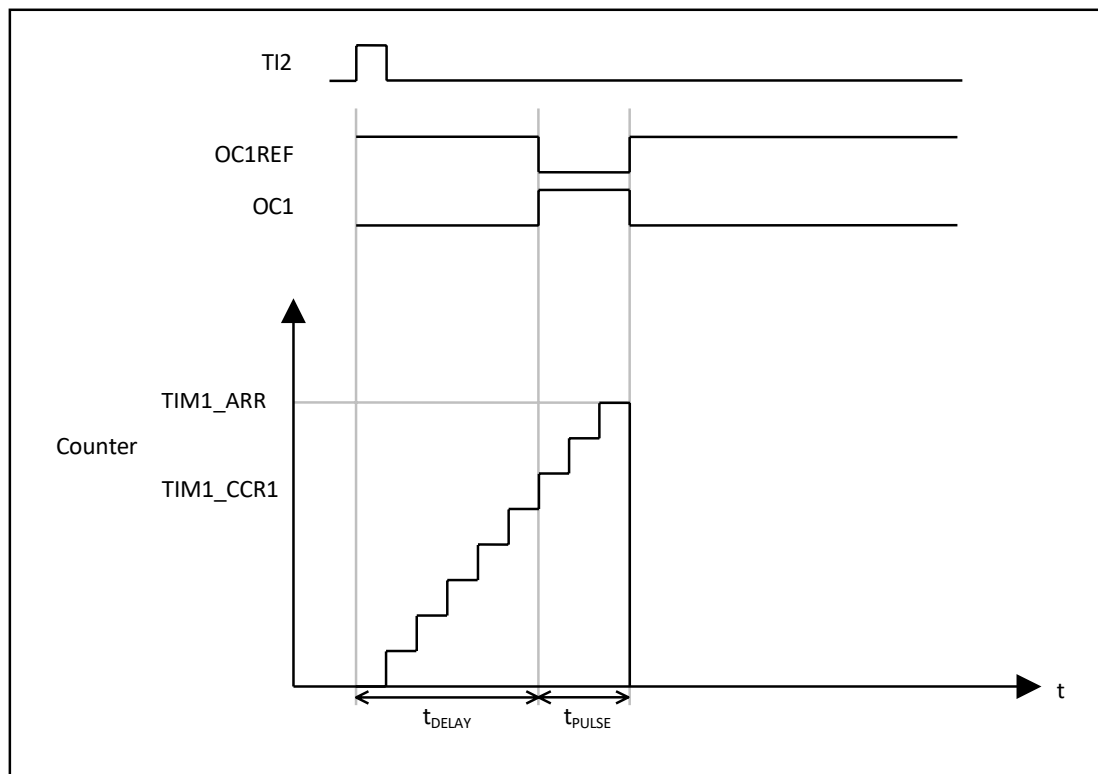
One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. Select One-pulse mode by setting the OPM bit in the TIM2_CR1 register. This makes the counter stop automatically at the next update event UEV.

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

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

Fig 11.3-32 Example of one pulse mode



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

Let's use TI2FP2 as trigger 1:

- Map TI2FP2 to TI2 by writing $CC2S='01'$ in the TIM2_CCMR1 register.
- TI2FP2 must detect a rising edge, write $CC2P='0'$ in the TIM2_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing $TS='110'$ in the TIM2_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to '110' in the TIM2_SMCR register (trigger

mode).

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

- The t_{DELAY} is defined by the value written in the TIM2_CCR1 register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (TIM2_ARR - TIM2_CCR1).
- Let us say the user wants to build a waveform with a transition from '0' to '1' when a compare match occurs and a transition from '1' to '0' when the counter reaches the auto-reload value. To do this, enable PWM mode 2 by writing OC1M=111 in the TIM2_CCMR1 register. The user can optionally enable the preload registers by writing OC1PE='1' in the TIM2_CCMR1 register and ARPE in the TIM2_CR1 register. In this case the compare value must be written in the TIM2_CCR1 register, the auto-reload value in the TIM2_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to '0' in this example.

In our example, the DIR and CMS bits in the TIM2_CR1 register should be low.

The user only wants one pulse (Single mode), so '1' must be written in the OPM bit in the TIM2_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0). When OPM bit in the TIM2_CR1 register is set to '0', so the Repetitive Mode is selected.

11.3.10 Particular case: OCx fast enable:

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

If the user wants to output a waveform with the minimum delay, the OCxFE bit in the TIM2_CCMRx register must be set. Then OCxRef (and OCx) are forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2

11.3.11 Clearing the OCxREF signal on an external event:

The OCxREF signal for a given channel can be driven Low by applying a High level to the ETRF input (OCxCE enable bit of the corresponding TIM2_CCMRx register set to '1'). The OCxREF signal remains Low until the next update event, UEV, occurs.

This function can only be used in output compare and PWM modes, and does not work in forced mode.

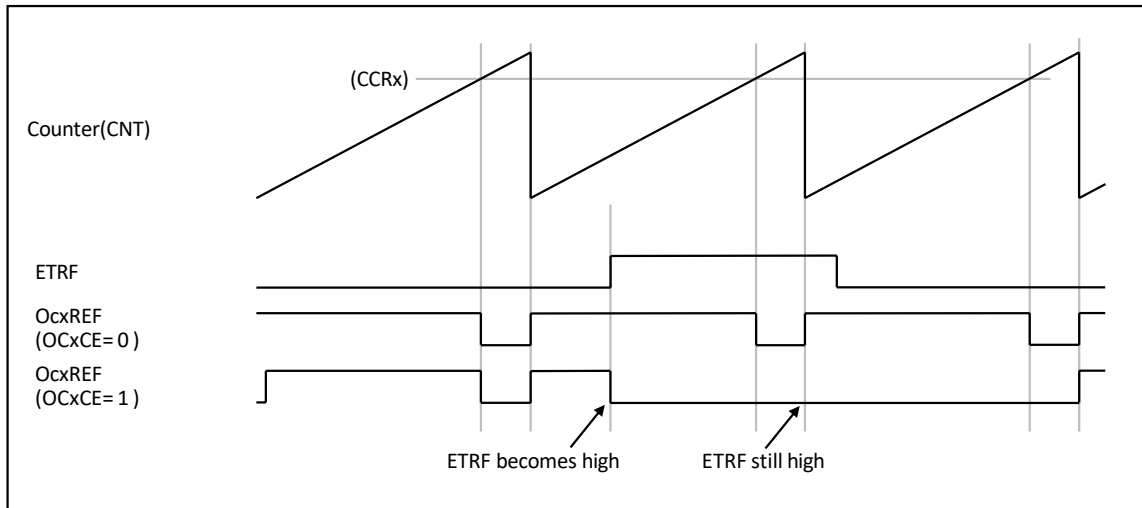
For example, the ETR signal can be connected to the output of a comparator to be used for current handling. In this case, the ETR must be configured as follow:

- The External Trigger Prescaler should be kept off: bits ETPS[1:0] of the TIM2_SMCR register set to '00'.

- The external clock mode 2 must be disabled: bit ECE of the TIM2_SMCR register set to '0'.
- The External Trigger Polarity (ETP) and the External Trigger Filter (ETF) can be configured according to the user needs.

Figure 11.3-33 shows the behavior of the OCxREF signal when the ETRF Input becomes High, for both values of the enable bit OCxCE. In this example, the timer TIM2 is programmed in PWM mode.

Fig 11.3-33 Clearing TIM2 OCxREF



11.3.12 Encoder interface mode

To select Encoder Interface mode write $SMS = 001$ in the TIM2_SMCR register if the counter is counting on TI2 edges only, $SMS = 010$ if it is counting on TI1 edges only and $SMS = 011$ if it is counting on both TI1 and TI2 edges.

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIM2_CCER register. When needed, the user can program the input filter as well.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Refer to Table 11.3-1 The counter is clocked by each valid transition on TI1FP1 or TI2FP2 (TI1 and TI2 after input filter and polarity selection, TI1FP1=TI1 if not filtered and not inverted, TI2FP2=TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit in TIM2_CR1 register written to '1'). The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the TIM2_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIM2_ARR register (0 to ARR or ARR down to 0 depending on the direction). So user must configure TIM2_ARR before starting. In the same way, the capture, compare, prescaler, repetition counter, trigger output features continue to work as normal. Encoder mode and External clock mode 2 are not compatible and must not be selected together.

In this mode, the counter is modified automatically following the speed and the direction of the incremental encoder and its content, therefore, always represents the encoder's position. The count direction correspond to the rotation direction of the connected sensor.

Table 11.3-1 summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Tab 11.3-1 Counting direction versus encoder signals

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

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally be used to convert the encoder's differential outputs to digital signals. This greatly increases noise immunity. The third encoder output which indicate the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

Figure 11.3-34 gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points. For this example we assume that the configuration is the following:

- CC1S='01'(TIM2_CCMR1 register, TI1FP1 mapped on TI1).
- CC2S='01'(TIM2_CCMR2 register, TI1FP2 mapped on TI2).
- CC1P='0', and IC1F = '0000'(TIM2_CCER register, TI1FP1 non-inverted, TI1FP1=TI1).
- CC2P='0', and IC2F = '0000'(TIM2_CCER register, TI1FP2 non-inverted, TI1FP2= TI2).
- SMS='011'(TIM2_SMCR register, both inputs are active on both rising and falling edges).
- CEN='1'(TIM2_CR1 register, Counter enabled).

Fig 11.3-34 Example of counter operation in encoder interface mode

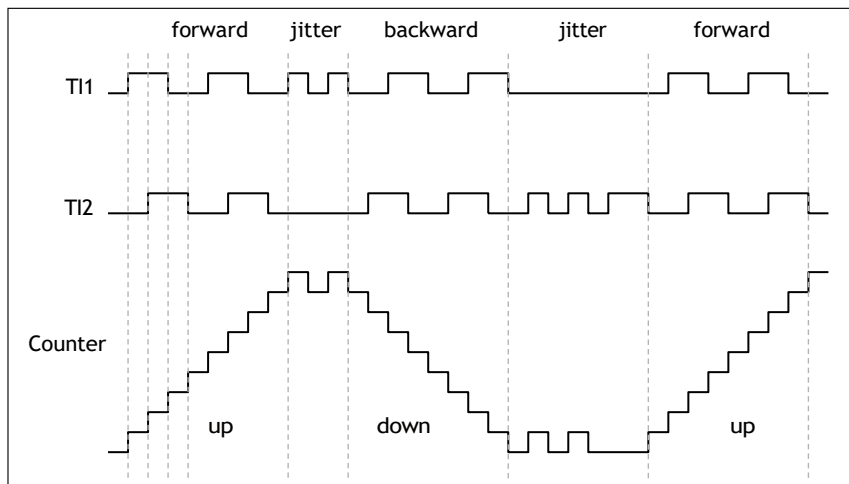
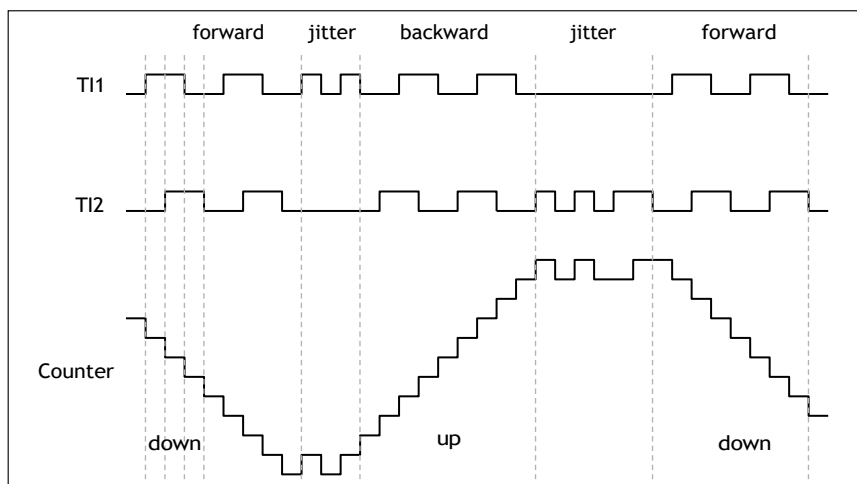


Fig 11.3-35 gives an example of counter behavior when TI1FP1 polarity is inverted (same configuration as above except CC1P='1').

Fig 11.3-35 Example of encoder interface mode with TI1FP1 polarity inverted.



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

11.3.13 Timer input XOR function

The TI1S bit in the TIM2_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIM2_CH1, TIM2_CH2 and TIM2_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture.

11.3.14 Timer and external trigger synchronization

The TIM2 timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

11.3.15 Slave mode: Reset mode

The counter and its prescaler can be reinitialized in response to an event on a trigger input. Moreover, if the URS bit from the TIM2_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIM2_ARR, TIM2_CCRx) are updated.

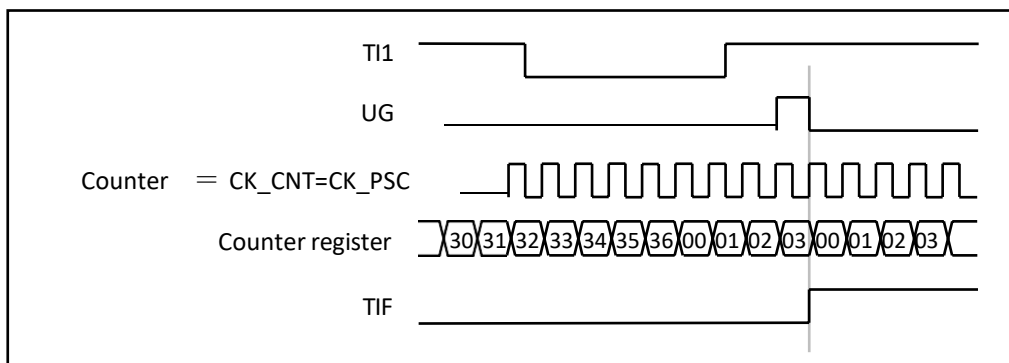
In the following example, the upcounter is cleared in response to a rising edge on TI1 input:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC1S bits select the input capture source only, CC1S = 01 in the TIM2_CCMR1 register. Write CC1P=0 in TIM2_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS=100 in TIM2_SMCR register. Select TI1 as the input source by writing TS=101 in TIM2_SMCR register.
- Start the counter by writing CEN=1 in the TIM2_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. In the meantime, the trigger flag is set (TIF bit in the TIM2_SR register) and an interrupt request can be sent if enabled (depending on the TIE and TDE bits in TIM2_DIER register).

The following figure shows this behavior when the auto-reload register TIM2_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

Fig 11.3-36 Control circuit in reset mode



11.3.16 Slave mode: Gated mode

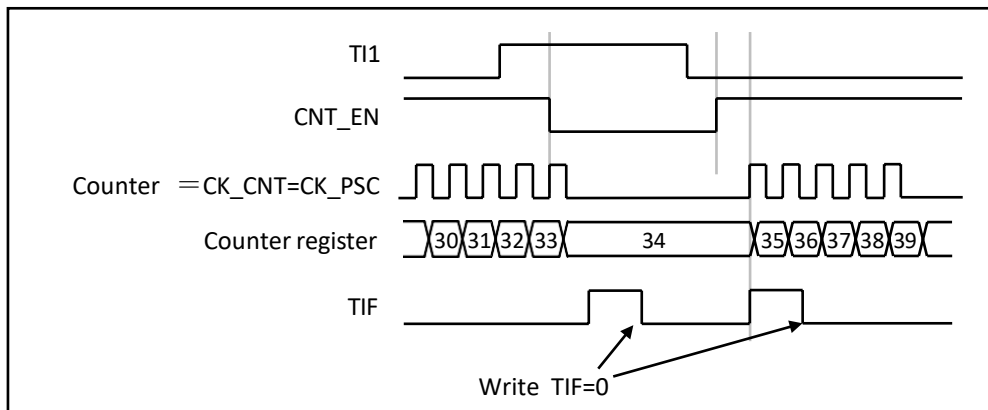
The counter can be enabled depending on the level of a selected input. In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so the user does not need to configure it. The CC1S bits select the input capture source only, CC1S=01 in TIM2_CCMR1 register. Write CC1P=1 in TIM2_CCER register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS=101 in TIM2_SMCR register. Select TI1 as the input source by writing TS=101 in TIM2_SMCR register.
- Enable the counter by writing CEN=1 in the TIM2_CR1 register (in gated mode, the counter doesn't start if CEN=0, whatever is the trigger input level).

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIM2_SR register is set both when the counter starts or stops.

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

Fig 11.3-37 Control circuit in gated mode



11.3.17 Slave mode: Trigger mode

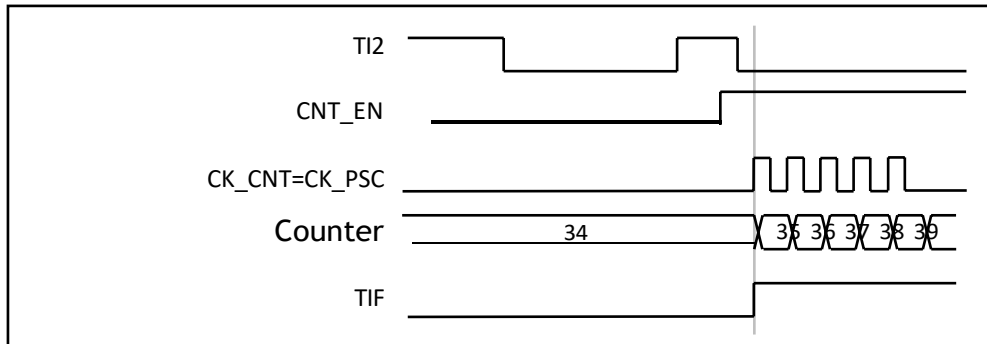
The counter can start in response to an event on a selected input. In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we don't need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC2S bits are configured to select the input capture source only, CC2S=01 in TIM2_CCMR1 register. Write CC2P=1 in TIM2_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS=110 in TIM2_SMCR register. Select TI2 as the input source by writing TS=110 in TIM2_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set.

The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

Fig 11.3-38 Control circuit in trigger mode



11.3.18 Slave mode: external clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input (in reset mode, gated mode or trigger mode). It is recommended not to select ETR as TRGI through the TS bits of TIM2_SMCR register.

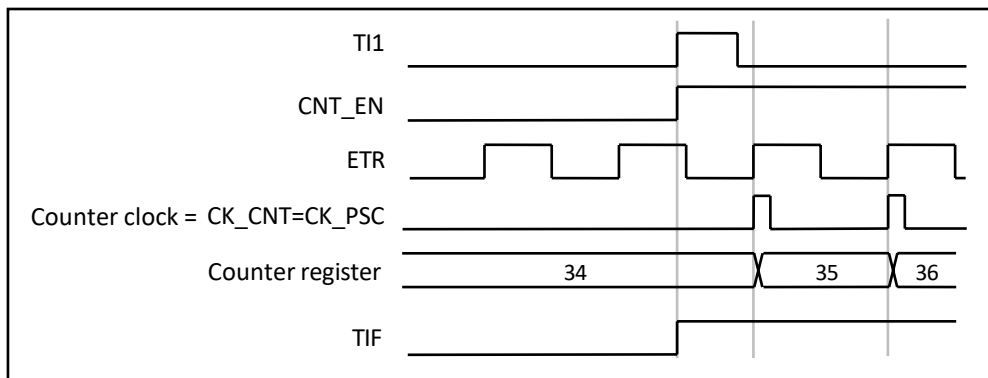
In the following example, the upcounter is incremented at each rising edge of the ETR signal as soon as a rising edge of TI1 occurs:

1. Configure the external trigger input circuit by programming the TIM2_SMCR register as follows:
 - ETF = 0000: no filter
 - ETPS = 00: prescaler disabled
 - ETP = 0: detection of rising edges on ETR and ECE=1 to enable the external clock mode 2.
2. Configure the channel 1 as follows, to detect rising edges on TI:
 - IC1F=0000: no filter.
 - The capture prescaler is not used for triggering and does not need to be configured.
 - CC1S=01 in TIM2_CCMR1 register to select only the input capture source
 - CC1P=0 in TIM2_CCER register to validate the polarity (and detect rising edge only).
3. Configure the timer in trigger mode by writing SMS=110 in TIM2_SMCR register. Select TI1 as the input source by writing TS=101 in TIM2_SMCR register.

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges.

The delay between the rising edge of the ETR signal and the actual reset of the counter is due to the resynchronization circuit on ETRP input.

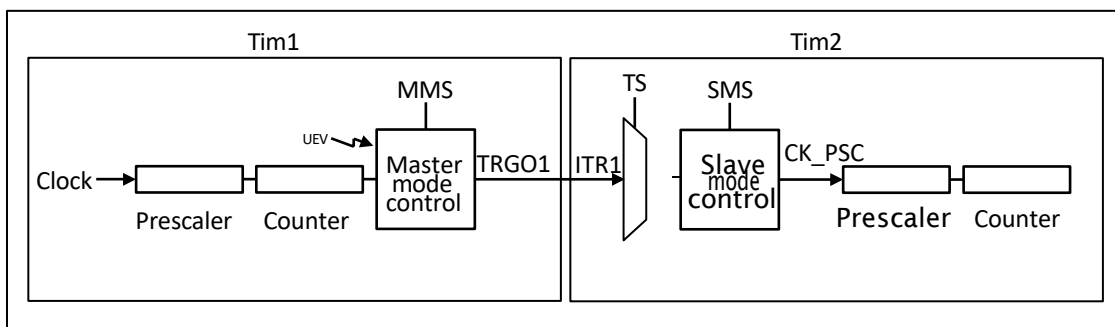
Fig 11.3-39 Control circuit in external clock mode 2 + trigger mode



11.3.19 Master/Slave timer example

The TIM2 timers are linked together internally for timer synchronization or chaining. When one Timer is configured in Master Mode, it can reset, start, stop or clock the counter of another Timer configured in Slave Mode.

Fig 11.3-40 Master/Slave timer example



11.3.20 Using one timer as prescaler for another timer

For example, the user can configure Timer 1 to act as a prescaler for Timer 2 (see Figure 11.3-40). To do this:

- Configure Timer 1 in master mode so that it outputs a periodic trigger signal on each update event UEV. If you write MMS=010 in the TIM1_CR2 register, a rising edge is output on TRGO1 each time an update event is generated.
- To connect the TRGO1 output of Timer 1 to Timer 2, Timer 2 must be configured in slave mode using ITR0 as internal trigger. You select this through the TS bits in the TIM2_SMCR register (writing TS=000).
- Then you put the slave mode controller in external clock mode 1 (write SMS=111 in the TIM2_SMCR register). This causes Timer 2 to be clocked by the rising edge of the periodic Timer 1 trigger signal (which correspond to the timer 1 counter overflow).
- Finally both timers must be enabled by setting their respective CEN bits (TIM2_CR1 register).

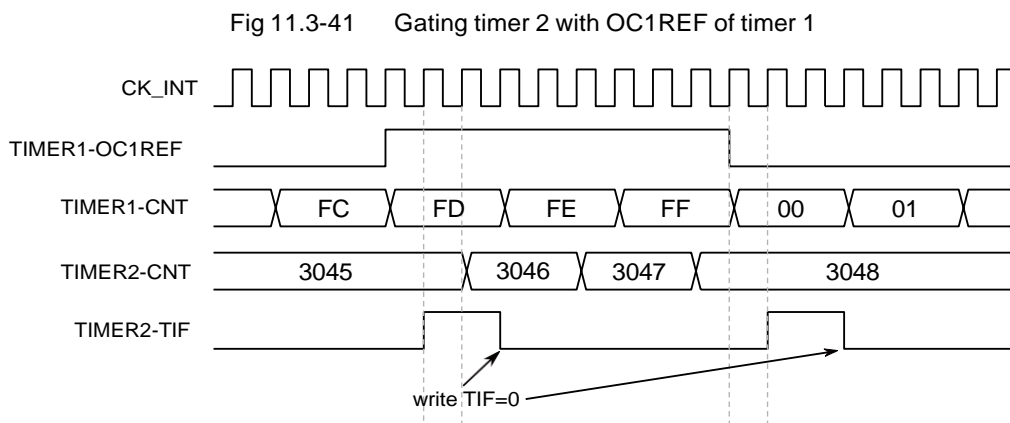
Note: If OCx is selected on Timer 1 as trigger output (MMS=1xx), its rising edge is used to clock the counter of timer 2.

11.3.21 Using one timer to enable another timer

In this example, we control the enable of Timer 2 with the output compare 1 of Timer 1. Refer to Figure 11.3-40 for connections. Timer 2 counts on the divided internal clock only when OC1REF of Timer 1 is high. Both counter clock frequencies are divided by 3 by the prescaler compared to CK_INT ($f_{CK_CNT} = f_{CK_INT} / 3$).

- Configure Timer 1 master mode to send its Output Compare 1 Reference (OC1REF) signal as trigger output (MMS=100 in the TIM1_CR2 register).
- Configure the Timer 1 OC1REF waveform (TIM1_CCMR1 register).
- Configure Timer 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in gated mode (SMS=101 in TIM2_SMCR register).
- Enable Timer 2 by writing '1' in the CEN bit (TIM2_CR1 register).
- Start Timer 1 by writing '1' in the CEN bit (TIM1_CR1 register).

Note: The counter 2 clock is not synchronized with counter 1, this mode only affects the Timer 2 counter enable signal.



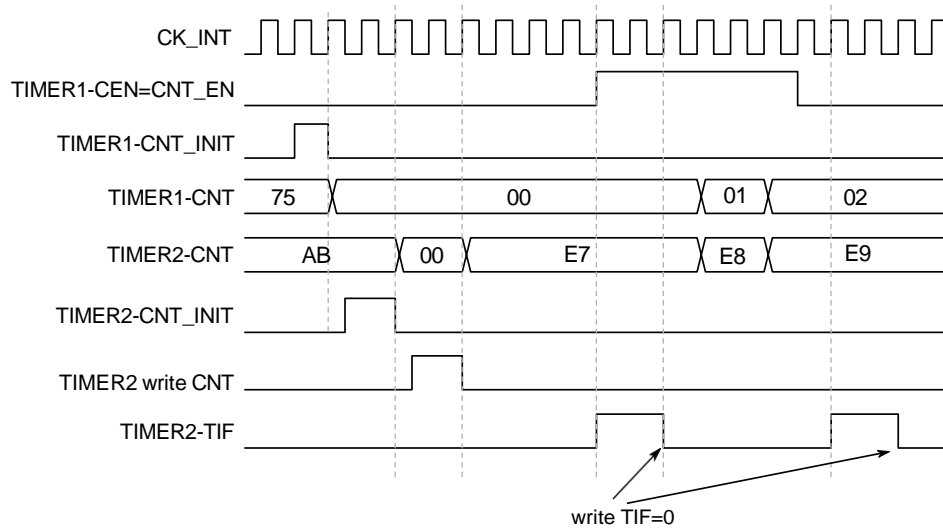
The Timer 2 counter and prescaler are not initialized before being started. So they start counting from their current value. It is possible to start from a given value by resetting both timers before starting Timer 1. You can then write any value you want in the timer counters. The timers can easily be reset by software using the UG bit in the TIM2_EGR registers.

In the next example, we synchronize Timer 1 and Timer 2. Timer 1 is the master and starts from 0. Timer 2 is the slave and starts from 0xE7. The prescaler ratio is the same for both timers. Timer 2 stops when Timer 1 is disabled by writing '0' to the CEN bit in the TIM1_CR1 register.

- Configure Timer 1 master mode to send its Output Compare 1 Reference (OC1REF) signal as trigger output (MMS=100 in the TIM1_CR2 register).
- Configure the Timer 1 OC1REF waveform (TIM1_CCMR1 register).
- Configure Timer 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in gated mode (SMS=101 in TIM2_SMCR register).
- Reset Timer 1 by writing '1' in UG bit (TIM1_EGR register).
- Reset Timer 2 by writing '1' in UG bit (TIM2_EGR register).
- Initialize Timer 2 to 0xE7 by writing '0xE7' in the timer 2 counter (TIM2_CNT).
- Enable Timer 2 by writing '1' in the CEN bit (TIM2_CR1 register).

- Start Timer 1 by writing '1 in the CEN bit (TIM1_CR1 register).
- Stop Timer 1 by writing '0 in the CEN bit (TIM1_CR1 register).

Fig 11.3-42 Gating timer 2 with Enable of timer 1

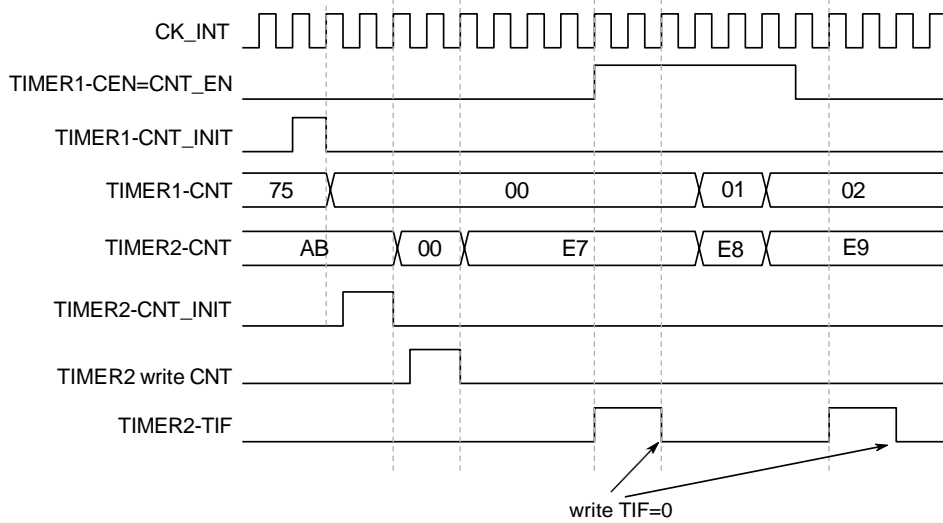


11.3.22 Using one timer to start another timer

In this example, we set the enable of Timer 2 with the update event of Timer 1. Refer to Figure 11.3-40 for connections. Timer 2 starts counting from its current value (which can be nonzero) on the divided internal clock as soon as the update event is generated by Timer 1. When Timer 2 receives the trigger signal its CEN bit is automatically set and the counter counts until we write '0 to the CEN bit in the TIM2_CR1 register. Both counter clock frequencies are divided by 3 by the prescaler compared to CK_INT ($f_{CK_CNT} = f_{CK_INT}/3$)

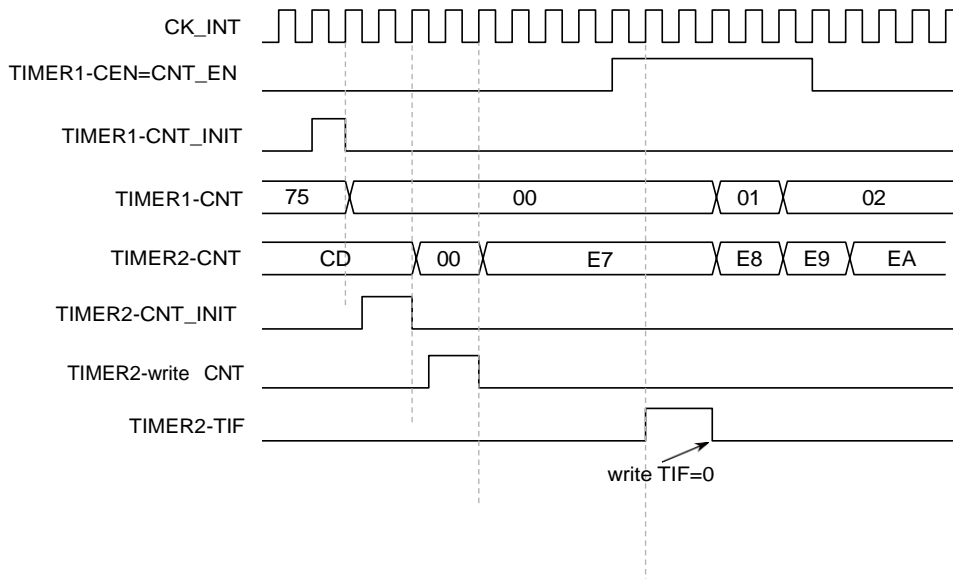
- Configure Timer 1 master mode to send its Update Event (UEV) as trigger output (MMS=010 in the TIM1_CR2 register).
- Configure the Timer 1 period (TIM1_ARR registers).
- Configure Timer 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in trigger mode (SMS=110 in TIM2_SMCR register).
- Start Timer 1 by writing '1 in the CEN bit (TIM1_CR1 register).

Fig 11.3-43 Triggering timer 2 with update of timer 1



As in the previous example, the user can initialize both counters before starting counting.

Fig 11.3-44 Triggering timer 2 with Enable of timer 1



11.3.23 2 timers synchronously in response to an external trigger

In this example, we set the enable of timer 1 when its TI1 input rises, and the enable of Timer 2 with the enable of Timer 1. Refer to Figure 11.3-40 for connections. To ensure the counters are aligned, Timer 1 must be configured in Master/Slave mode (slave with respect to TI1, master with respect to Timer 2):

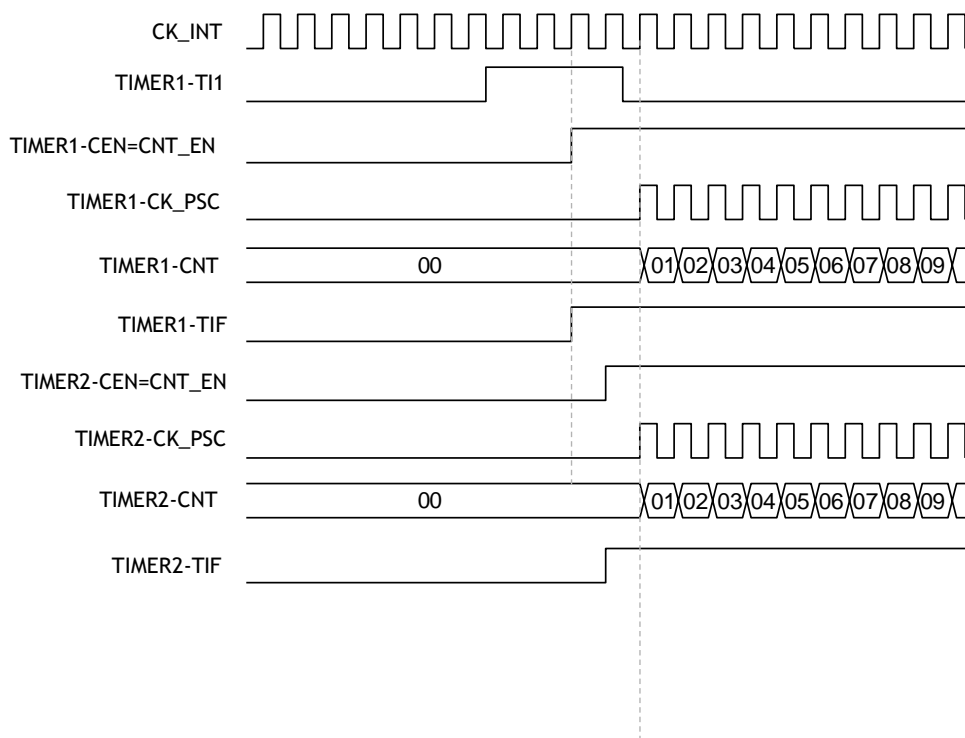
- Configure Timer 1 master mode to send its Enable as trigger output (MMS=001 in the TIM1_CR2 register).
- Configure Timer 1 slave mode to get the input trigger from TI1 (TS=100 in the TIM1_SMCR register).

- Configure Timer 1 in trigger mode (SMS=110 in the TIM1_SMCR register).
- Configure the Timer 1 in Master/Slave mode by writing MSM=1 (TIM1_SMCR register).
- Configure Timer 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in trigger mode (SMS=110 in the TIM2_SMCR register).

When a rising edge occurs on TI1 (Timer 1), both counters starts counting synchronously on the internal clock and both TIF flags are set.

Note: In this example both timers are initialized before starting (by setting their respective UG bits). Both counters starts from 0, but you can easily insert an offset between them by writing any of the counter registers (TIM2_CNT). You can see that the master/slave mode insert a delay between CNT_EN and CK_PSC on timer 1.

Fig 11.3-45 Triggering timer 1 and 2 with timer 1 TI1 input



11.3.24 Debug mode

When the microcontroller enters debug mode (Cortex ® -M0+ core - halted), the TIM2 counter either continues to work normally or stops, depending on DBG_TIM2_STOP configuration bit in DBGMCU module. For more details, refer to 30

11.4 TIM2 Register

Tab 11.4-1 TIM2 register map

Address offset	Register	Reset value	Description
0x00	TIM2_CR1	0x0000_0000	TIM2 register 1
0x04	TIM2_CR2	0x0000_0000	TIM2 register 2
0x08	TIM2_SMCR	0x0000_0000	TIM2 slave mode register
0x0C	TIM2_DIER	0x0000_0000	TIM2 interrupt enable register
0x10	TIM2_SR	0x0000_0000	TIM2 status register
0x14	TIM2_EGR	0x0000_0000	TIM2 event generation register
0x18	TIM2_CCMR1	0x0000_0000	TIM2 capture/compare mode register 1
0x1C	TIM2_CCMR2	0x0000_0000	TIM2 capture/compare mode register 2
0x20	TIM2_CCER	0x0000_0000	TIM2 capture/compare enable register
0x24	TIM2_CNT	0x0000_0000	TIM2 counter register
0x28	TIM2_PSC	0x0000_0000	TIM2 prescaler register
0x2C	TIM2_ARR	0x0000_0000	TIM2 auto-reload register
0x30	Reserved		
0x34	TIM2_CCR1	0x0000_0000	TIM2 capture/compare register 1
0x38	TIM2_CCR2	0x0000_0000	TIM2 capture/compare register 2
0x3C	TIM2_CCR3	0x0000_0000	TIM2 capture/compare register 3
0x40	TIM2_CCR4	0x0000_0000	TIM2 capture/compare register 4

11.5 TIM2 Register Description

11.5.1 TIM2 control register 1(TIM2_CR1)

Register	Address offset	Access	Reset value	Description
TIM2_CR1	0x00	RW	0x0000_0000	TIM2 control register 1

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						CKD[1:0]	
7	6	5	4	3	2	1	0
ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN

TIM2 control register 1(TIM2_CR1)bit description

Bit	Access	description
[31:10]	-	Reserved, must be kept at reset value.
[9:8]	RW	<p>CKD[1:0]: Clock division</p> <p>This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock (t_{DTS}) used by the dead-time generators and the digital filters (ETR, TIX).</p> <p>00: $t_{DTS} = t_{CK_INT}$</p> <p>01: $t_{DTS} = 2 * t_{CK_INT}$</p> <p>10: $t_{DTS} = 4 * t_{CK_INT}$</p> <p>11: Reserved, do not program this value</p>
[7]	RW	<p>ARPE: Auto-reload preload enable</p> <p>0: TIM2_ARR register is not buffered</p> <p>1: TIM2_ARR register is buffered</p>
[6:5]	RW	<p>CMS[1:0]: Center-aligned mode selection</p> <p>00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR).</p> <p>01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIM2_CCMRx register) are set only when the counter is counting down.</p> <p>10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIM2_CCMRx register) are set only when the counter is counting up.</p> <p>11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIM2_CCMRx register) are set both when the counter is counting up or down.</p> <p>Note : It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1)</p>

[4]	RW	<p>DIR: Direction 0: Counter used as upcounter 1: Counter used as downcounter</p> <p>Note : <i>This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.</i></p>
[3]	RW	<p>OPM: One pulse mode 0: 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)</p>
[2]	RW	<p>URS: Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt if enabled. These events can be:</p> <ul style="list-style-type: none"> - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller <p>1: Only counter overflow/underflow generates an update interrupt if enabled.</p>
[1]	RW	<p>UDIS: Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events:</p> <ul style="list-style-type: none"> - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller <p>Buffered registers are then loaded with their preload values. 1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.</p>
[0]	RW	<p>CEN: Counter enable 0: Counter disabled 1: Counter enabled</p> <p>Note : <i>External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware..</i></p>

11.5.2 TIM2 control register 2(TIM2_CR2)

Register	Address offset	Access	Reset value	Description
TIM2_CR2	0x04	RW	0x0000_0000	TIM2 control register 2

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1
7	6	5	4	3	2	1	0
TI1S	MMS[1:0]			Reserved	CCUS	Reserved	CCPC

TIM2 control register 2(TIM2_CR2)bit description

BIT	Access	Description
[31:15]	-	Reserved, must be kept at reset value.
[14]	RW	OIS4: Output Idle state 4 (OC4 output) refer to OIS1 bit
[13]	RW	OIS3N: Output Idle state 3 (OC3N output) refer to OIS1N bit
[12]	RW	OIS3: Output Idle state 3 (OC3 output) refer to OIS1 bit
[11]	RW	OIS2N: Output Idle state 2 (OC2N output) refer to OIS1N bit
[10]	RW	OIS2: Output Idle state 2 (OC2 output) refer to OIS1 bit
[9]	RW	OIS1N: Output Idle state 1 (OC1N output) 0: OC1N=0 after a dead-time when MOE=0 1: OC1N=1 after a dead-time when MOE=0 Note : This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM2_BDTR register)
[8]	RW	OIS1: Output Idle state 1 (OC1 output). 0: OC1=0 (after a dead-time if OC1N is implemented) when MOE=0 1: OC1=1 (after a dead-time if OC1N is implemented) when MOE=0 Note : This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM2_BDTR register).
[7]	RW	TI1S: TI1 selection 0: The TIM2_CH1 pin is connected to TI1 input 1: The TIM2_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)

[6:4]	RW	<p>MMS[2:0]: Master mode selection</p> <p>These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows:</p> <table border="1" data-bbox="422 331 1457 1160"> <thead> <tr> <th data-bbox="422 331 513 365">MMS</th> <th data-bbox="513 331 673 365">Mode</th> <th data-bbox="673 331 1457 365">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="422 365 513 517">000</td> <td data-bbox="513 365 673 517">Reset</td> <td data-bbox="673 365 1457 517">the UG bit from the TIM2_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset.</td> </tr> <tr> <td data-bbox="422 517 513 819">001</td> <td data-bbox="513 517 673 819">Enable</td> <td data-bbox="673 517 1457 819">the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIM2_SMCR register).</td> </tr> <tr> <td data-bbox="422 819 513 898">010</td> <td data-bbox="513 819 673 898">Update</td> <td data-bbox="673 819 1457 898">The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer.</td> </tr> <tr> <td data-bbox="422 898 513 1010">011</td> <td data-bbox="513 898 673 1010">Compare Pulse</td> <td data-bbox="673 898 1457 1010">The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. (TRGO).</td> </tr> <tr> <td data-bbox="422 1010 513 1043">100</td> <td data-bbox="513 1010 673 1043">Compare 1</td> <td data-bbox="673 1010 1457 1043">OC1REF signal is used as trigger output (TRGO)</td> </tr> <tr> <td data-bbox="422 1043 513 1077">101</td> <td data-bbox="513 1043 673 1077">Compare 2</td> <td data-bbox="673 1043 1457 1077">OC2REF signal is used as trigger output (TRGO)</td> </tr> <tr> <td data-bbox="422 1077 513 1111">110</td> <td data-bbox="513 1077 673 1111">Compare 3</td> <td data-bbox="673 1077 1457 1111">OC3REF signal is used as trigger output (TRGO)</td> </tr> <tr> <td data-bbox="422 1111 513 1160">111</td> <td data-bbox="513 1111 673 1160">Compare 4</td> <td data-bbox="673 1111 1457 1160">OC4REF signal is used as trigger output (TRGO)</td> </tr> </tbody> </table>	MMS	Mode	Description	000	Reset	the UG bit from the TIM2_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset.	001	Enable	the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIM2_SMCR register).	010	Update	The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer.	011	Compare Pulse	The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. (TRGO).	100	Compare 1	OC1REF signal is used as trigger output (TRGO)	101	Compare 2	OC2REF signal is used as trigger output (TRGO)	110	Compare 3	OC3REF signal is used as trigger output (TRGO)	111	Compare 4	OC4REF signal is used as trigger output (TRGO)
MMS	Mode	Description																											
000	Reset	the UG bit from the TIM2_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset.																											
001	Enable	the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIM2_SMCR register).																											
010	Update	The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer.																											
011	Compare Pulse	The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. (TRGO).																											
100	Compare 1	OC1REF signal is used as trigger output (TRGO)																											
101	Compare 2	OC2REF signal is used as trigger output (TRGO)																											
110	Compare 3	OC3REF signal is used as trigger output (TRGO)																											
111	Compare 4	OC4REF signal is used as trigger output (TRGO)																											
[3]	-	Reserved, must be kept at reset value.																											
[2]	RW	<p>CCUS: Capture/compare control update selection</p> <p>0: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COMG bit only</p> <p>1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COMG bit or when an rising edge occurs on TRGI</p> <p>Note: <i>This bit acts only on channels that have a complementary output.</i></p>																											
[1]	-	Reserved, must be kept at reset value.																											
[0]	RW	<p>CCPC: Capture/compare preloaded control.</p> <p>0: CCxE, CCxNE and OCxM bits are not preloaded</p> <p>1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when a commutation event (COM) occurs (COMG bit set or rising edge detected on TRGI, depending on the CCUS bit).</p> <p>Note: <i>This bit acts only on channels that have a complementary output.</i></p>																											

11.5.3 TIM2 slave mode control register(TIM2_SMCR)

Register	Address offset	Access	Reset value	Description
TIM2_SMCR	0x08	RW	0x0000_0000	TIM2 slave mode control register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ETP	ECE	ETPS[1:0]		ETF[3:0]			
7	6	5	4	3	2	1	0
MSM	TS[2:0]			Reserved	SMS[2:0]		

TIM2 slave mode control register(TIM2_SMCR)bit description

Bit	Access	Description
[31:16]	-	Reserved, must be kept at reset value.
[15]	RW	<p>ETP: External trigger polarity</p> <p>This bit selects whether ETR or \overline{ETR} is used for trigger operations 0: ETR is non-inverted, active at high level or rising edge. 1: ETR is inverted, active at low level or falling edge.</p>
[14]	RW	<p>ECE: External clock enable</p> <p>This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal.</p> <p>Note :</p> <ol style="list-style-type: none"> Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=111). It is possible to simultaneously use external clock mode 2 with the following slave modes: reset mode, gated mode and trigger mode. Nevertheless, TRGI must not be connected to ETRF in this case (TS bits must not be 111). If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
[13:12]	RW	<p>ETPS[1:0]: External trigger prescaler</p> <p>External trigger signal ETRP frequency must be at most 1/4 of TIMxCLK frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8</p>

[11:8]	RW	<p>ETF[3:0]: External trigger filter</p> <p>This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output :</p> <p>0000: No filter, $f_{SAMPLING}=f_{DTS}$ 1000: $f_{SAMPLING} = f_{DTS}/8, N = 6$ 0001: $f_{SAMPLING} = f_{CK_INT}, N = 2$ 1001: $f_{SAMPLING} = f_{DTS}/8, N = 8$ 0010: $f_{SAMPLING} = f_{CK_INT}, N = 4$ 1010: $f_{SAMPLING} = f_{DTS}/16, N = 5$ 0011: $f_{SAMPLING} = f_{CK_INT}, N = 8$ 1011: $f_{SAMPLING} = f_{DTS}/16, N = 6$ 0100: $f_{SAMPLING} = f_{DTS}/2, N = 6$ 1100: $f_{SAMPLING} = f_{DTS}/16, N = 8$ 0101: $f_{SAMPLING} = f_{DTS}/2, N = 8$ 1101: $f_{SAMPLING} = f_{DTS}/32, N = 5$ 0110: $f_{SAMPLING} = f_{DTS}/4, N = 6$ 1110: $f_{SAMPLING} = f_{DTS}/32, N = 6$ 0111: $f_{SAMPLING} = f_{DTS}/4, N = 8$ 1111: $f_{SAMPLING} = f_{DTS}/32, N = 8$</p>										
[7]	RW	<p>MSM: Master/slave mode</p> <p>0: No action</p> <p>1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.</p>										
[6:4]	RW	<p>TS[2:0]: Trigger selection</p> <p>This bit-field selects the trigger input to be used to synchronize the counter.</p> <p>000: Internal Trigger 0 (ITR0) 001: Internal Trigger 1 (ITR1) 010: Internal Trigger 2 (ITR2) 011: Internal Trigger 3 (ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) 111: External Trigger input (ETRF)</p> <hr/> <table border="0" style="width: 100%; border-collapse: collapse;"> <tr> <td style="border-bottom: 1px solid black; width: 15%;"></td> <td style="border-bottom: 1px solid black; width: 15%;">ITR0(TS=000)</td> <td style="border-bottom: 1px solid black; width: 15%;">ITR1(TS=001)</td> <td style="border-bottom: 1px solid black; width: 15%;">ITR2(TS=010)</td> <td style="border-bottom: 1px solid black; width: 15%;">ITR3(TS=011)</td> </tr> <tr> <td>TIM2</td> <td>tim1_trgo</td> <td>irq_timer10</td> <td>irq_timer11</td> <td>irq_pca</td> </tr> </table> <p>Note : These bits must be changed only when they are not used (e.g. when SMS=000) to avoid wrong edge detections at the transition.</p>		ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)	TIM2	tim1_trgo	irq_timer10	irq_timer11	irq_pca
	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)								
TIM2	tim1_trgo	irq_timer10	irq_timer11	irq_pca								
[3]	R	Reserved, must be kept at reset value.										

[2:0]	RW	<p>SMS[2:0]: Slave mode selection</p> <p>When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control Register description).</p> <table border="1" data-bbox="422 369 1457 1086"> <thead> <tr> <th data-bbox="422 369 502 403">SMS</th> <th data-bbox="502 369 726 403">Mode</th> <th data-bbox="726 369 1457 403">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="422 403 502 481">000:</td> <td data-bbox="502 403 726 481">Slave mode disabled</td> <td data-bbox="726 403 1457 481">if CEN = 1 then the prescaler is clocked directly by the internal clock.</td> </tr> <tr> <td data-bbox="422 481 502 560">001:</td> <td data-bbox="502 481 726 560">Encoder mode 1</td> <td data-bbox="726 481 1457 560">Counter counts up/down on TI1FP1 edge depending on TI1FP2 level.</td> </tr> <tr> <td data-bbox="422 560 502 638">010:</td> <td data-bbox="502 560 726 638">Encoder mode 2</td> <td data-bbox="726 560 1457 638">Counter counts up/down on TI1FP2 edge depending on TI2FP1 level.</td> </tr> <tr> <td data-bbox="422 638 502 716">011:</td> <td data-bbox="502 638 726 716">Encoder mode 3</td> <td data-bbox="726 638 1457 716">Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input.</td> </tr> <tr> <td data-bbox="422 716 502 795">100:</td> <td data-bbox="502 716 726 795">Reset Mode</td> <td data-bbox="726 716 1457 795">Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers.</td> </tr> <tr> <td data-bbox="422 795 502 940">101:</td> <td data-bbox="502 795 726 940">Gated Mode</td> <td data-bbox="726 795 1457 940">The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled.</td> </tr> <tr> <td data-bbox="422 940 502 1019">110:</td> <td data-bbox="502 940 726 1019">Trigger Mode</td> <td data-bbox="726 940 1457 1019">The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled.</td> </tr> <tr> <td data-bbox="422 1019 502 1086">111:</td> <td data-bbox="502 1019 726 1086">External Clock Mode 1</td> <td data-bbox="726 1019 1457 1086">Rising edges of the selected trigger (TRGI) clock the counter.</td> </tr> </tbody> </table>	SMS	Mode	Description	000:	Slave mode disabled	if CEN = 1 then the prescaler is clocked directly by the internal clock.	001:	Encoder mode 1	Counter counts up/down on TI1FP1 edge depending on TI1FP2 level.	010:	Encoder mode 2	Counter counts up/down on TI1FP2 edge depending on TI2FP1 level.	011:	Encoder mode 3	Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input.	100:	Reset Mode	Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers.	101:	Gated Mode	The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled.	110:	Trigger Mode	The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled.	111:	External Clock Mode 1	Rising edges of the selected trigger (TRGI) clock the counter.
SMS	Mode	Description																											
000:	Slave mode disabled	if CEN = 1 then the prescaler is clocked directly by the internal clock.																											
001:	Encoder mode 1	Counter counts up/down on TI1FP1 edge depending on TI1FP2 level.																											
010:	Encoder mode 2	Counter counts up/down on TI1FP2 edge depending on TI2FP1 level.																											
011:	Encoder mode 3	Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input.																											
100:	Reset Mode	Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers.																											
101:	Gated Mode	The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled.																											
110:	Trigger Mode	The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled.																											
111:	External Clock Mode 1	Rising edges of the selected trigger (TRGI) clock the counter.																											

11.5.4 TIM2 interrupt enable register(TIM2_DIER)

Register	Address offset	Access	Reset value	Description
TIM2_DIER	0x0C	RW	0x0000_0000	TIM2 interrupt enable register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	TIE	Reserved	CC4IE	CC3IE	CC2IE	CC1IE	UIE

TIM2 interrupt enable register(TIM2_DIER)

Bit	Access	Description
[31:7]	-	Reserved, must be kept at reset value.
[6]	RW	TIE: Trigger interrupt enable 0: Trigger interrupt disabled 1: Trigger interrupt enabled
[5]	-	Reserved, must be kept at reset value.。
[4]	RW	CC4IE: Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled 1: CC4 interrupt enabled
[3]	RW	CC3IE: Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled 1: CC3 interrupt enabled
[2]	RW	CC2IE: Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled 1: CC2 interrupt enabled
[1]	RW	CC1IE: Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
[0]	RW	UIE: Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

11.5.5 TIM2 status register(TIM2_SR)

Register	Address offset	Access	Reset value	Description
TIM2_SR	0x10	RC_W0	0x0000_0000	TIM2 status register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			CC4OF	CC3OF	CC2OF	CC1OF	Reserved
7	6	5	4	3	2	1	0
Reserved	TIF	Reserved	CC4IF	CC3IF	CC2IF	CC1IF	UIF

TIM2 status register(TIM2_SR)bit description

Bit	Access	Description
[31:13]	-	Reserved, must be kept at reset value.
[12]	RC_W0	CC4OF: Capture/Compare 4 overcapture flag refer to CC1OF description
[11]	RC_W0	CC3OF: Capture/Compare 3 overcapture flag refer to CC1OF description
[10]	RC_W0	CC2OF: Capture/Compare 2 overcapture flag refer to CC1OF description
[9]	RC_W0	CC1OF: Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The counter value has been captured in TIM2_CCR1 register while CC1IF flag was already set .
[8:7]	-	Reserved, must be kept at reset value.
[6]	RC_W0	TIF: Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode, both edges in case gated mode is selected). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
[5]	-	Reserved, must be kept at reset value.
[4]	RC_W0	CC4IF: Capture/Compare 4 interrupt flag refer to CC1IF description.
[3]	RC_W0	CC3IF: Capture/Compare 3 interrupt flag refer to CC1IF description.
[2]	RC_W0	CC2IF: Capture/Compare 2 interrupt flag refer to CC1IF description.

[1]	RC_W0	<p>CC1IF: Capture/Compare 1 interrupt flag</p> <p>If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIM2_CR1 register description). It is cleared by software. 0: No match. 1: The content of the counter TIM2_CNT has matched the content of the TIM2_CCR1 register.</p> <p>If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIM2_CCR1 register. 0: No input capture occurred. 1: The counter value has been captured in TIM2_CCR1 register (An edge has been detected on IC1 which matches the selected polarity).</p>
[0]	RC_W0	<p>UIF: Update interrupt flag</p> <ul style="list-style-type: none"> - This bit is set by hardware on an update event. It is cleared by software. <p>0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated:</p> <ul style="list-style-type: none"> - At overflow or underflow and if the UDIS=0 in the TIM2_CR1 register. - When CNT is reinitialized by software using the UG bit in TIM2_EGR register, if URS=0 and UDIS=0 in the TIM2_CR1 register. - When CNT is reinitialized by a trigger event (refer to the synchro control register description), if URS=0 and UDIS=0 in the TIM2_CR1 register.11.5.3

11.5.6 TIM2 event generation register(TIM2_EGR)

Register	Address offset	Access	Reset value	Description
TIM2_EGR	0x14	W	0x0000_0000	TIM2 event generation register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	TG	Reserved	CC4G	CC3G	CC2G	CC1G	UG

TIM2 event generation register(TIM2_EGR)bit description

Bit	Access	Description
[31:7]	-	Reserved, must be kept at reset value.
[6]	W	TG: Trigger interrupt generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in TIM2_SR register. Related interrupt can occur if enabled.
[5]	-	COMIF: Reserved, must be kept at reset value.
[4]	W	CC4G: Capture/Compare 4 interrupt generation refer to CC1G description
[3]	W	CC3G: Capture/Compare 3 interrupt generation refer to CC1G description
[2]	W	CC2G: Capture/Compare 2 interrupt generation refer to CC1G description
[1]	W	CC1G: Capture/Compare 1 interrupt generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel 1 If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIM2_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
[0]	W	UG: Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Re-initialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), else it takes the auto-reload value (TIM2_ARR) if DIR=1 (downcounting).

11.5.7 TIM2 capture/compare mode register 1 (TIM2_CCMR1)

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

Register	Address offset	Access	Reset value	Description
TIM2_CCMR1	0x18	RW	0x0000_0000	TIM2 capture/compare mode register 1

31	30	29	28	27	26	25	24
Reserved							
Reserved							
23	22	21	20	19	18	17	16
Reserved							
Reserved							
15	14	13	12	11	10	9	8
IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]	
OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]	
7	6	5	4	3	2	1	0
IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	

TIM2 Input capture mode bit description :

TIM2 capture register(TIM2_CCMR1)bit description

Bit	Access	description
[31:16]	-	Reserved, must be kept at reset value.
[15:12]	RW	IC2F[3:0]: Input capture 2 filter
[11:10]	RW	IC2PSC[1:0]: Input capture 2 prescaler
[9:8]	RW	<p>CC2S[1:0]: Capture/Compare 2 selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 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 working only if an internal trigger input is selected through TS bit (TIM2_SMCR register)</p> <p>Note: CC2S bits are writable only when the channel is OFF (CC2E = 0 in TIM2_CCER).</p>

[7:4]	RW	<p>IC1F[3:0]: Input capture 1 filter</p> <p>This bit-field defines the frequency used to sample T11 input and the length of the digital filter applied to T11. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output:</p> <p>0000: No filter, $f_{SAMPLING} = f_{DTS}$ 1000: $f_{SAMPLING} = f_{DTS}/8$, N = 6 0001: $f_{SAMPLING} = f_{CK_INT}$, N = 2 1001: $f_{SAMPLING} = f_{DTS}/8$, N = 8 0010: $f_{SAMPLING} = f_{CK_INT}$, N = 4 1010: $f_{SAMPLING} = f_{DTS}/16$, N = 5 0011: $f_{SAMPLING} = f_{CK_INT}$, N = 8 1011: $f_{SAMPLING} = f_{DTS}/16$, N = 6 0100: $f_{SAMPLING} = f_{DTS}/2$, N = 6 1100: $f_{SAMPLING} = f_{DTS}/16$, N = 8 0101: $f_{SAMPLING} = f_{DTS}/2$, N = 8 1101: $f_{SAMPLING} = f_{DTS}/32$, N = 5 0110: $f_{SAMPLING} = f_{DTS}/4$, N = 6 1110: $f_{SAMPLING} = f_{DTS}/32$, N = 6 0111: $f_{SAMPLING} = f_{DTS}/4$, N = 8 1111: $f_{SAMPLING} = f_{DTS}/32$, N = 8</p>
[3:2]	RW	<p>IC1PSC[1:0]: Input capture 1 prescaler</p> <p>This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E=0 (TIM2_CCER register). 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 10: capture is done once every 4 events 11: capture is done once every 8 events</p>
[1:0]	RW	<p>CC1S[1:0]: Capture/Compare 1 selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output</p> <p>01: CC1 channel is configured as input, IC1 is mapped on T11 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register)</p> <p>Note: <i>CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIM2_CCER).</i></p>

TIM2_CCMR1 Output compare mode bit description :

TIM2 compare register (TIM2_CCMR1)bit description

Bit	Access	description
[31:16]	-	Reserved, must be kept at reset value.
[15]	RW	OC2CE: Output compare 2 clear enable)
[14:12]	RW	OC2M[2:0]: Output Compare 2 mode
[11]	RW	OC2PE: Output compare 2 preload enable
[10]	RW	OC2FE: Output compare 2 fast enable
[9:8]	RW	<p>CC2S[1:0]: Capture/Compare 2 selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output</p> <p>01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIM2_SMCR register)</p> <p>Note: <i>CC2S bits are writable only when the channel is OFF (CC2E = 0 in TIM2_CCER).</i></p>

[7]	RW	<p>OC1CE: Output compare 1 clear enable 0: OC1Ref is not affected by the ETRF input 1: OC1Ref is cleared as soon as a High level is detected on ETRF input</p>
[6:4]	RW	<p>OC1M[2:0]: Output Compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits.</p> <p>001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIM2_CNT matches the capture/compare register 1 (TIM2_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIM2_CNT matches the capture/compare register 1 (TIM2_CCR1). 011: Toggle - OC1REF toggles when TIM2_CNT=TIM2_CCR1. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as TIM2_CNT<TIM2_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF= ‘0) as long as TIM2_CNT>TIM2_CCR1 else active (OC1REF=1). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIM2_CNT<TIM2_CCR1 else active. In downcounting, channel 1 is active as long as TIM2_CNT>TIM2_CCR1 else inactive.</p> <p>Note: <i>In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.</i></p>
[3]	RW	<p>OC1PE: Output compare 1 preload enable 0: Preload register on TIM2_CCR1 disabled. TIM2_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIM2_CCR1 enabled. Read/Write operations access the preload register. TIM2_CCR1 preload value is loaded in the active register at each update event.</p> <p>Note : 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIM2_BDTR register) and CC1S=00 (the channel is configured in output). 2: The PWM mode can be used without validating the preload register only in one- pulse mode (OPM bit set in TIM2_CR1 register). Else the behavior is not guaranteed.</p>
[2]	RW	<p>OC1FE: Output compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.</p>

[1:0]	RW	<p>CC1S[1:0]: Capture/Compare 2 selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 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 on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register)</p> <p>Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIM2_CCER).</p>
-------	----	---

11.5.8 TIM2 capture/compare mode register 2(TIM2_CCMR2)

Register	Address offset	Access	Reset value	Description
TIM2_CCMR1	0x1C	RW	0x0000_0000	TIM2 capture/compare mode register 2

31	30	29	28	27	26	25	24
Reserved							
Reserved							
23	22	21	20	19	18	17	16
Reserved							
Reserved							
15	14	13	12	11	10	9	8
IC4F[3:0]				IC4PSC[1:0]		CC4S[1:0]	
OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]	
7	6	5	4	3	2	1	0
IC3F[3:0]				IC3PSC[1:0]		CC3S[1:0]	
OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	

TIM2_CCMR2 Input capture mode :

TIM2 capture/compare mode register 2(TIM2_CCMR2)bit description

Bit	Access	Description
[31:16]	-	Reserved, must be kept at reset value.
[15:12]	RW	IC4F[3:0]: Input capture 4 filter
[11:10]	RW	IC4PSC[1:0]: Input capture 4 prescaler
[9:8]	RW	<p>CC4S[1:0]: Capture/Compare 4 selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output</p> <p>01: CC4 channel is configured as input, IC4 is mapped on TI4</p> <p>10: CC4 channel is configured as input, IC4 is mapped on TI3</p> <p>11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register)</p> <p>Note: CC4S bits are writable only when the channel is OFF (CC4E = 0 in TIM2_CCER).</p>
[7:4]	RW	IC3F[3:0]: Input capture 3 filter
[3:2]	RW	IC3PSC[1:0]: Input capture 3 prescaler

[1:0]	RW	<p>CC3S[1:0]: Capture/Compare 3 selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output</p> <p>01: CC3 channel is configured as input, IC3 is mapped on TI3</p> <p>10: CC3 channel is configured as input, IC3 is mapped on TI4</p> <p>11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register)</p> <p>Note: CC3S bits are writable only when the channel is OFF (CC3E = 0 in TIM2_CCER).</p>
-------	----	--

TIM2_CCMR2 Output compare mode :

TIM2 compare register 2(TIM2_CCMR2)bit description

Bit	Access	Description
[31:16]	-	Reserved, must be kept at reset value.
[15]	RW	OC4CE: Output compare 4 clear enable
[14:12]	RW	OC4M[2:0]: Output Compare 4 mode
[11]	RW	OC4PE: Output compare 4 preload enable)
[10]	RW	OC4FE: Output compare 4 fast enable
[9:8]	RW	<p>CC4S[1:0]: Capture/Compare 4 selection</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output</p> <p>01: CC4 channel is configured as input, IC4 is mapped on TI4</p> <p>10: CC4 channel is configured as input, IC4 is mapped on TI3</p> <p>11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register)</p> <p>Note: CC4S bits are writable only when the channel is OFF (CC4E = 0 in TIM2_CCER).</p>
[7]	RW	OC3CE: Output compare 3 clear enable
[6:4]	RW	OC3M[2:0]: Output Compare 3 mode
[3]	RW	OC3PE: Output compare 3 preload enable
[2]	RW	OC3FE: Output compare 3 fast enable)
[1:0]	RW	<p>CC3S[1:0]: Capture/Compare 2 selection)</p> <p>This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output</p> <p>01: CC3 channel is configured as input, IC3 is mapped on TI3</p> <p>10: CC3 channel is configured as input, IC3 is mapped on TI4</p> <p>11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register)</p> <p>Note: CC3S bits are writable only when the channel is OFF (CC3E = 0 in TIM2_CCER).</p>

11.5.9 TIM2 capture/compare enable register(TIM2_CCER)

Register	Address offset	Access	Reset value	Description
TIM2_CCER	0x20	RW	0x0000_0000	TIM2 capture/compare enable register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CC4P	CC4E	Reserved		CC3P	CC3E
7	6	5	4	3	2	1	0
Reserved		CC2P	CC2E	Reserved		CC1P	CC1E

TIM2 capture/compare enable register(TIM2_CCER)bit description

Bit	Access	Description
[31:14]	-	Reserved, must be kept at reset value.
[13]	RW	CC4P: Capture/Compare 4 output polarity refer to CC1P description
[12]	RW	CC4E: Capture/Compare 4 output enable refer to CC1E description
[11:10]	-	Reserved, must be kept at reset value.
[9]	RW	CC3P: Capture/Compare 3 output polarity refer to CC1P description
[8]	RW	CC3E: Capture/Compare 3 output enable refer to CC1E description
[7:6]	-	Reserved, must be kept at reset value.
[5]	RW	CC2P: Capture/Compare 2 output polarity refer to CC1P description.
[4]	RW	CC2E: Capture/Compare 2 output enable refer to CC1E description.
[3:2]	-	Reserved, must be kept at reset value.
[1]	RW	CC1P: Capture/Compare 1 output polarity CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: This bit selects whether IC1 or IC1 is used for trigger or capture operations. 0: non-inverted: capture is done on a rising edge of IC1. When used as external trigger, IC1 is non-inverted. 1: inverted: capture is done on a falling edge of IC1. When used as external trigger, IC1 is inverted..

[0]	RW	<p>CC1E: Capture/Compare 1 output enable</p> <p>CC1 channel configured as output: 0: Off - OC1 is not active. 1: On - OC1 signal is output on the corresponding output pin.</p> <p>CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIM2_CCR1) or not. 0: Capture disabled. 1: Capture enabled.</p>
-----	----	---

Tab 11.5-12 Output control bit for standard OCx channels

CCxE bit	OCx output state
0	Output Disabled (OCx=0, OCx_EN=0)
1	OCx=OCxREF + Polarity, OCx_EN=1

Note: The state of the external IO pins connected to the standard OCx channels depends on the OCx channel state and the GPIO and AFIO registers.

11.5.10 TIM2 counter(TIM2_CNT)

Register	Address offset	Access	Reset value	Description
TIM2_CNT	0x24	RW	0x0000_0000	TIM2 counter

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

TIM2 Counter(TIM2_CNT)bit description

Bit	Access	Description
[31:20]	-	Reserved, must be kept at reset value.
[15:0]	RW	CNT[15:0]: Counter value

11.5.11 TIM2 prescaler(TIM2_PSC)

Register	Address offset	Access	Reset value	Description
TIM2_PSC	0x28	RW	0x0000_0000	TIM2 prescaler

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

TIM2 Prescaler (TIM2_PSC)bit description

Bit	Access	Description
[31:16]	-	Reserved, must be kept at reset value.
[15:0]	RW	PSC[15:0]: Prescaler value The counter clock frequency CK_CNT is equal to / (PSC[15:0] + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of TIM2_EGR register or through trigger controller when configured in “reset mode”).

11.5.12 TIM2 auto-reload register(TIM2_ARR)

Register	Address offset	Access	Reset value	Description
TIM2_ARR	0x2C	RW	0x0000_0000	TIM2 auto-reload register

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

TIM2 auto-reload register(TIM2_ARR)bit description

Bit	Access	Description
[31:16]	-	Reserved, must be kept at reset value.
[15:0]	RW	ARR[15:0]: Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to the Section 11.3.3 : Time-base unit for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

11.5.13 TIM2 capture/compare register 1(TIM2_CCR1)

Register	Address offset	Access	Reset value	Description
TIM2_CCR1	0x34	RW	0x0000_0000	TIM2 capture/compare register 1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															

TIM2 capture/compare register 1(TIM2_CCR1)bit description

Bit	Access	Description
[31:16]	-	Reserved, must be kept at reset value.
[15:0]	RW	CCR1[15:0]: Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM2_CCMR1 register(bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM2_CNT and signaled on OC1 output. If channel CC1is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

11.5.14 TIM2 capture/compare register 2(TIM2_CCR2)

Register	Address offset	Access	Reset value	Description
TIM2_CCR2	0x38	RW	0x0000_0000	TIM2 capture/compare register 2

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															

TIM2 capture/compare register 2(TIM2_CCR2)bit description

Bit	Access	Description
[31:16]	-	Reserved, must be kept at reset value.
[15:0]	RW	<p>CCR2[15:0]: Capture/Compare 2 value</p> <p>If channel CC2 is configured as output:</p> <p>CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM2_CCMR2 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs.</p> <p>The active capture/compare register contains the value to be compared to the counter TIM2_CNT and signalled on OC2 output.</p> <p>If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2)</p>

11.5.15 TIM2 capture/compare register 3(TIM2_CCR3)

Register	Address offset	Access	Reset value	Description
TIM2_CCR3	0x3C	RW	0x0000_0000	TIM2 capture/compare register 3

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															

TIM2 capture/compare register 3(TIM2_CCR3)bit description

Bit	Access	Description
[31:16]	-	Reserved, must be kept at reset value.
[15:0]	RW	<p>CCR3[15:0]: Capture/Compare 3 value</p> <p>If channel CC3 is configured as output:</p> <p>CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM2_CCMR3 register(bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs.</p> <p>The active capture/compare register contains the value to be compared to the counter TIM2_CNT and signaled on OC3 output.</p> <p>If channel CC3 is configured as input:</p> <p>CCR3 is the counter value transferred by the last input capture 3 event (IC3).</p>

11.5.16 TIM2 capture/compare register 4(TIM2_CCR4)

Register	Address offset	Access	Reset value	Description
TIM2_CCR4	0x40	RW	0x0000_0000	TIM2 capture/compare register 4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4[15:0]															

TIM2 capture/compare register 4

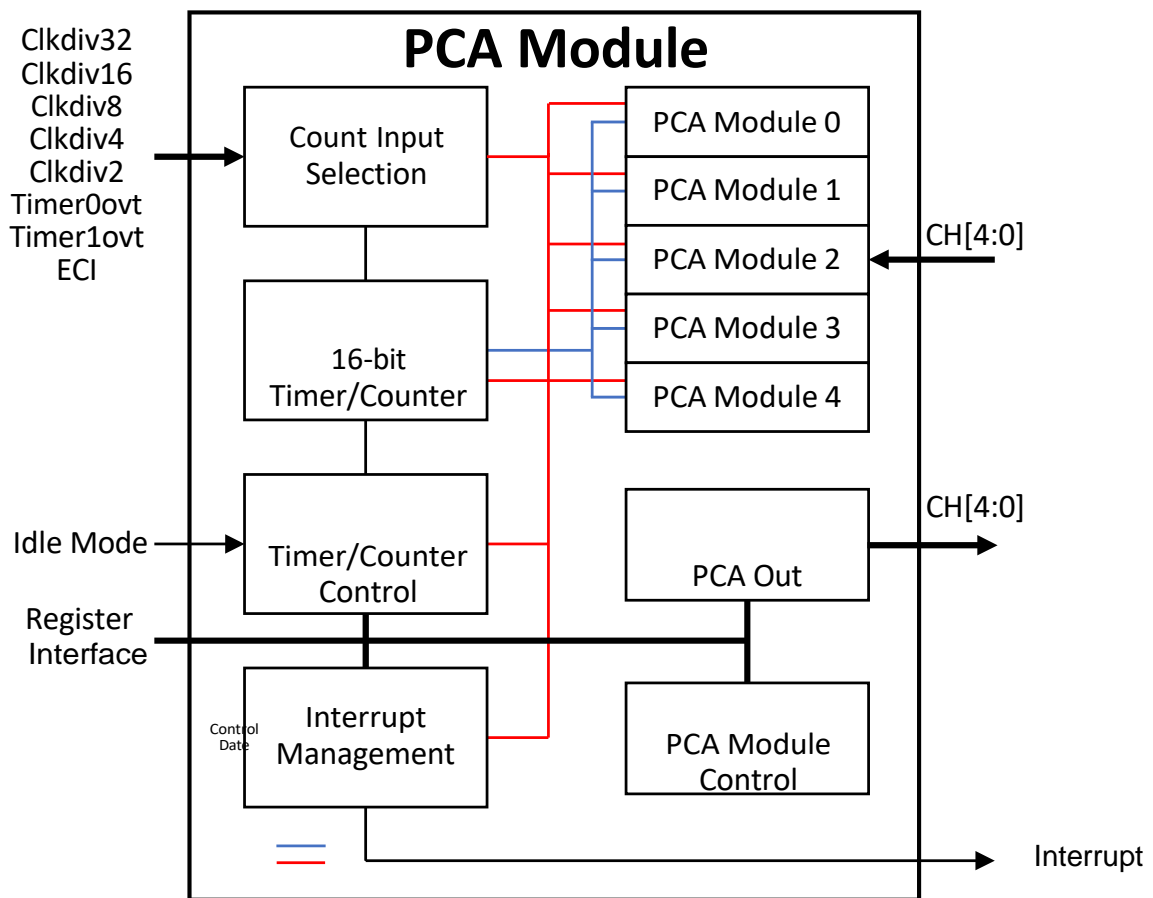
Bit	Access	Description
[31:16]	-	Reserved, must be kept at reset value.
[15:0]	RW	<p>CCR4[15:0]: Capture/Compare 4 value</p> <p>If CC4 channel is configured as output:</p> <p>CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM2_CCMR4 register (bit OC4PE). Else the preload value is copied in the active capture/compare 4 register when an update event occurs.</p> <p>The active capture/compare register contains the value to be compared to the counter TIM2_CNT and signalled on OC4 output.</p> <p>If CC4 channel is configured as input</p> <p>CCR4 is the counter value transferred by the last input capture 4 event (IC4)</p>

12 Programmable Counter Array(PCA)

12.1 PCA introduction

PCA (Programmable Counter Array, Programmable Counter Array) supports up to five 16-bit capture/comparison modules. The timer/counter can be used as a general-purpose clock counting/event counter with capture/comparison functions. Each module of the PCA can be independently programmed to provide input capture, output comparison or pulse width modulation. Additionally module 4 has an additional watchdog timer mode.

Fig 12.1-1 PCA Block Diagram



12.2 PCA functional description

Each module can be configured to work independently, and there are three working modes: edge trigger capture, output comparison, and 8-bit pulse width modulation. Each module has its own function registers in the system controller, and these registers are used to configure the working mode of the module and exchange data with the module. Each compare/capture module is composed of a compare/capture register set (CCAPx), a 16-bit comparator and various logic gate control components. The register group is used to store the time or number of times for external trigger capture conditions, or internal trigger comparison conditions. In PWM mode, the register (CCAPxL) is used to control the duty cycle of the output waveform. Each module can be independently programmed to operate in any of the following modes:

- 16 Rising edge, falling edge or any edge trigger of bit capture mode.
- Comparison mode: 16-bit software timer, 16-bit high-speed output or 8-bit pulse width modulation.
- Not started.

Compare/Capture Module Mode Register (CCAPMx) to determine the corresponding operating mode. When programming the compare/capture module, they are based on a common time count. The opening and closing of the timer/counter can control the operation of the PCA timer/counter through the CR.CR bit. If the corresponding enable bit (CCAPMx.CCIE) is set, when a match or capture occurs, the compare/capture flag (CR.CCFx) is set and a PCA interrupt request is generated. The CPU can read and write the CCAPx register at any time.

12.2.1 PCA Timer/Counter

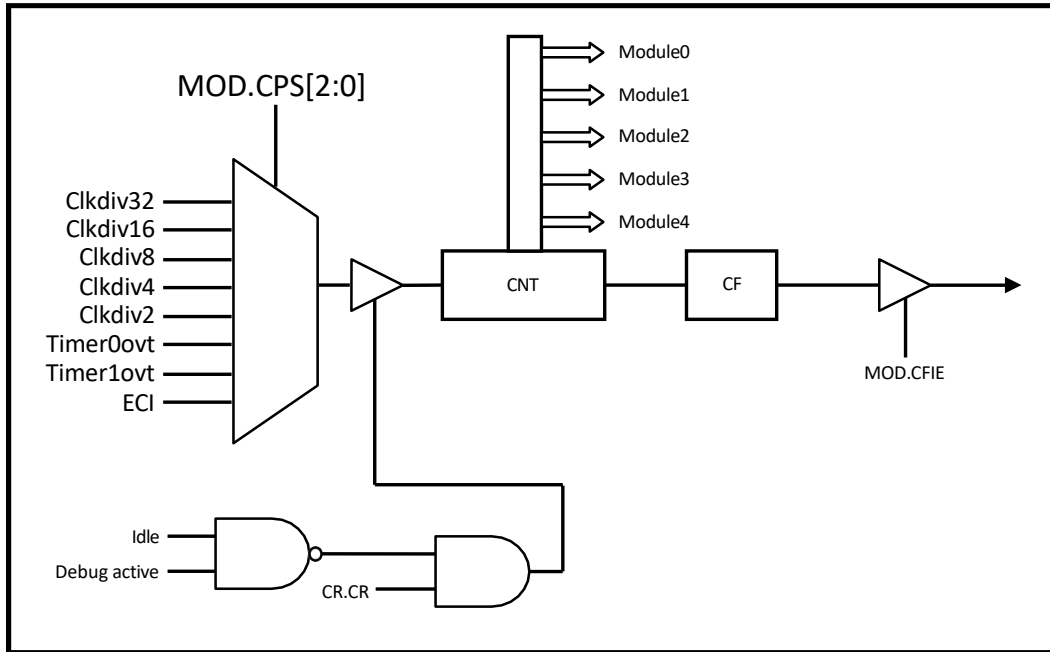
This set of special function registers of CNT can be used as a 16-bit timing/counter. This is a 16-bit counter that counts up. If the MOD.CFIE bit is set to "1", when the CNT overflows, the hardware automatically sets the PCA overflow flag (CR.CF) and generates a PCA interrupt request. MOD.CPS[2:0] select eight signals to input to the timer/counter.

- System clock PCLK divided by 32
- System clock PCLK divided by 16
- System clock PCLK divided by 8
- System clock PCLK divided by 4
- System clock PCLK divided by 2
- Timer 0 overflow (overflow): Each time the timer 0 counts overflow, the CNT will increase, thus providing PCA's variable programming frequency input.
- Timer 1 overflow (overflow): Each time the timer 1 counts overflow, the CNT will increase, thus providing PCA's variable programming frequency input.
- ECI: The CPU samples PCA ECI every 4 PCLK clock cycles. When each sampling result changes from high to low, CNT_L (CNT low 8 bit) automatically adds 1, so the highest ECI input frequency cannot be higher than 1/8 of the system clock PCLK to meet the sampling requirements.

Set the operation controller (CR.CR) to start PCA timing/counter. When MOD.CIDL is set to "1", PCA timer/counter can continue to run in idle mode. The CPU can read the value of CNT at any time, but

when counting starts(CR.CR=1), in order to prevent counting errors,CNT is forbidden to write.

Fig 12.2-1 PCA Counter Block Diagram



12.2.2 Capture function

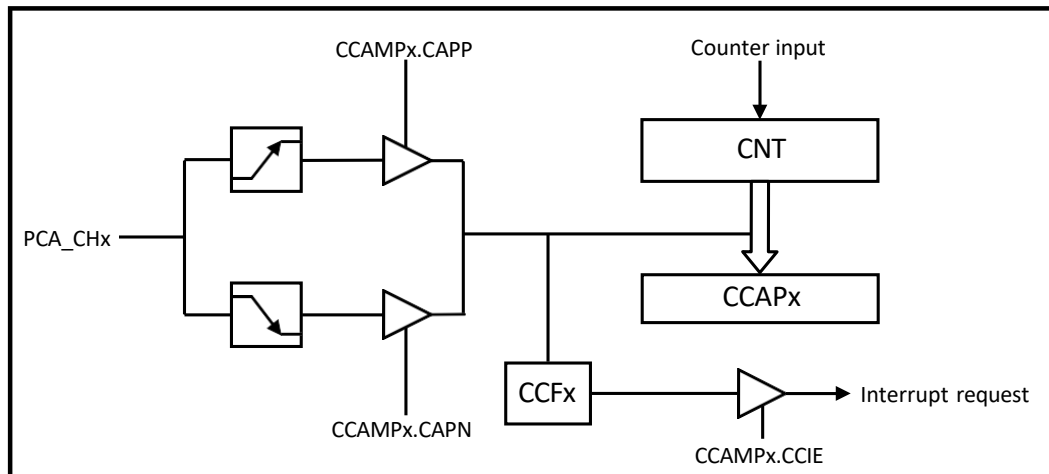
PCA capture mode provides the function of 5-channel PCA to measure pulse period, pulse width, duty cycle and phase difference. The level jump on the pin causes the PCA to capture the value of the PCA counter/timer and load it into the 16-bit capture/compare register (CCAPx) of the corresponding module. CCAPMx.CAPP and CCAPMx.CAPN bits are used to select the level change type of trigger capture: low level to high level (positive edge), high level to low level (negative edge) or any change (positive edge or negative edge). When capture occurs, the capture/comparison flag (CCFx) in CR is set to logic '1' and an interrupt request is generated (if CCF interrupt is allowed). When the CPU turns to the interrupt service program, the CR.CCFx bit cannot be automatically cleared by the hardware. The user software writes the INTCLR.CCFx register to clear this flag bit. If both CCAPMx.CAPP and CCAPMx.CAPN bits are set to logic '1', you can determine whether this capture is triggered by the rising edge or by the falling edge by directly reading the status of the corresponding port pin. The resolution is equal to the clock of the timer/counter. The input signal must maintain at least 2 clock cycles during high or low level to ensure that the input signal can be recognized by hardware.

The CPU can read or write the CCAPx registers at any time. Capture settings:

- When capturing at the external rising edge is required,CCAPMx.CAPP = "1"and CCAPMx.CAPN = "0"
- When capturing at the external falling edge is required,CCAPMx.CAPP = "0"and CCAPMx.CAPN = "1"
- When it is necessary to capture at the external rising and falling edges, CCAPMx.CAPP = "1" and CCAPMx.CAPN = "1"

Note *Then the captured values of the same module will overwrite the existing captured values. In order to maintain the captured values, save it in RAM in the interrupt service program. This operation must be completed before the next event occurs, otherwise the previous captured sampling value will be lost.*

Fig 12.2-2 PCA Capture Block Diagram



12.2.3 PCA comparison function

PCA comparison function provides the following functions: timer, event counter, pulse width modulation.

PCA comparison function can provide three modes: 16-bit software timer mode, high-speed output mode and PWM mode. In the first two modes, compare/capture module compares 16-bit PCA timer/counter value with the 16-bit value pre-loaded into the `CCAPx` register of the module. In PWM mode, the PCA module continuously compares the PCA Timer/Counter Low Byte Register (`CNT`) with an 8-bit value in the `CCAPxL` register. Compare every 4 clock cycles, i.e. match with clock with the fastest PCA timer/counter rates. Set the `CCAPMx.ECOM` bit to select the comparison function of the module. To use the module in comparison mode correctly, please follow the general procedures below:

- Select the operation mode of PCA module
- Select the input signal of PCA timer/counter
- The comparison value is loaded into the module's comparison/capture register pair
- Set PCA Timer/Counter operation control bit
- After matching, interrupt is generated to clear the comparison/capture flag of the module

16-bit software counter mode

To set a compare/capture module to work in 16-bit software timer mode, you need to set the `CCAPMx.ECOM` and `CCAPMx.MAT` bits. Once a match occurs between the PCA timer/counter and the compare/capture register (`CCAPx`), this will set the compare/capture flag (`CR.CCFx`) of the module. This will generate an interrupt request if the corresponding interrupt enable bit (`CCAPMx.CCIE`) is set. Since the hardware does not clear the compare/capture flag (`CR.CCFx`), the user must clear the software flag. In the interrupt service program, a new 16-bit comparison value can be written into the compare/capture register (`CCAPx`). Note: When updating these registers, in order to prevent invalid matching, user software should write `CCAPxL` first and then `CCAPxH`. Once the `CCAPxL` is written, the `ECOMx` bit of the disabled comparison function will be cleared, while the `ECOMx` bit set by the `CCAPxH`

will be written at the same time, and the comparison function will be re-enabled. That is, when writing a 16-bit value to the capture/compare register of PCA0, the lower byte should be written first.

High-speed output mode

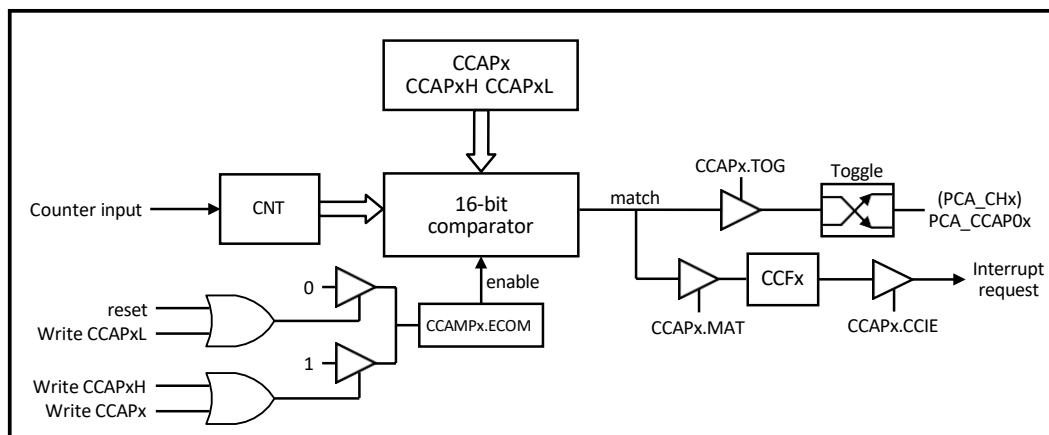
In the high-speed output mode, whenever the value in PCA counter matches the 16-bit capture/compare register(CCAPx)of the module , the value of the PCA_CCAP0x register (corresponding to the PCA_CHx pin) of the module PCA will flip. This can provide higher precision than switching IO output, because this high-speed output will not respond to interrupts and affect the output frequency. If the CPU switches the IO to generate PWM output, the power consumption and precision will be worsen.

To set the high-speed output mode of a compare/capture module, set the CCAPMx.ECOM,CCAPMx.MAT and CCAPMx.TOG bits. Match between PCA timer/counter and compare/capture register (CCAPx) flips the value of PCA_CCAP0x register and set the compare/capture flag of the module (CR.CCFx).

Users can also choose to generate an interrupt request by setting the corresponding interrupt enable bit(CCAPMx.CCIE)when a match occurs, an interrupt request can be generated. Since the compare/capture flag cannot be cleared by hardware, the user must clear this flag in the software. If the user does not change the comparison/capture register in the interrupt program,the PCA will count again, and the next flip will occur if it matches. In the interrupt service program, a new 16-bit comparison value can be written into the compare/capture register(CCAPx).

Note : To prevent invalid matching while updating these registers, user software should write CCAPxL first and then CCAPxH. Write to CCAPxL to clear and disable the ECOM bit of the comparison function, and write to the ECOM bit set by CCAPxH and re-enables the compare function.

Fig 12.2-3 PCA Compare Block Diagram

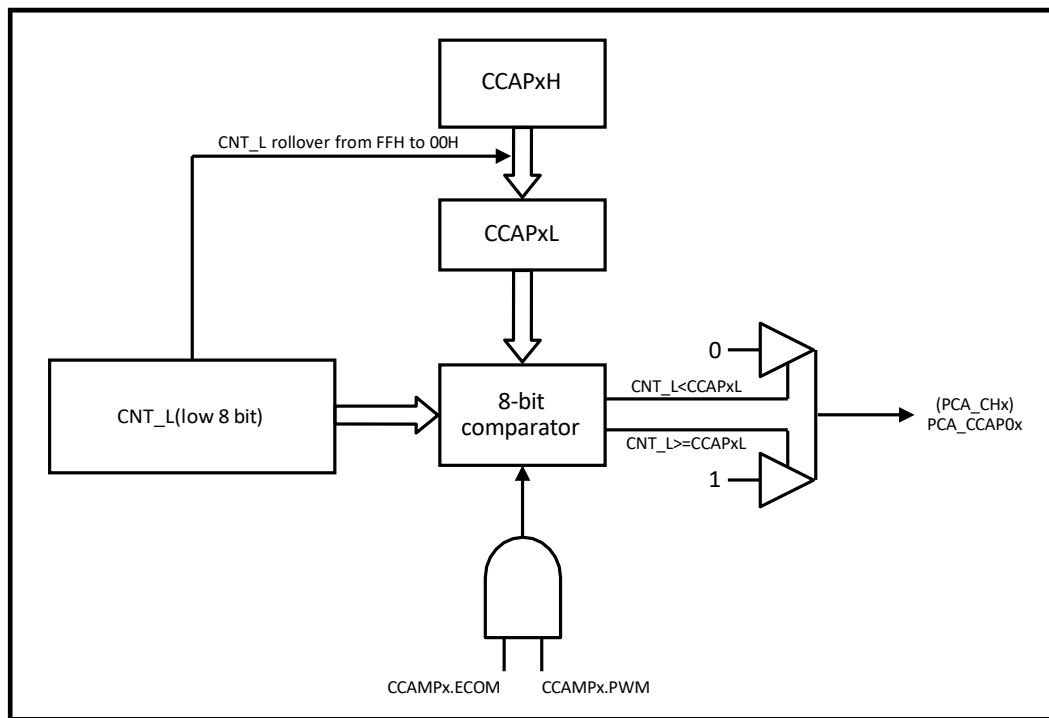


8-Bit Pulse Width Modulation(PWM) Function

Pulse width modulation is a technology that uses a program to control the duty cycle, period and phase of the waveform. All five PCA modules can be independently used in PCA_CHx corresponding to PCA pin generates a pulse width modulation (PWM) output with a pulse width of 8 bits resolution. The frequency of PWM output depends on the time base of PCA counter/timer. Use the capture/compare register CCAPxL of the module to change the duty cycle of the PWM output signal. When the low byte (CNT_L) of PCA counter/timer is equal to the value in CCAPxL, the pin of PCA_CHx is set to "1"; When the count value in CNT_L overflows, PCA_CHx output is reset to "0". When the low byte CNT_L of the

counter/timer overflows (from 0xFF to 0x00), the value saved in CCAPxH is automatically loaded into CCAPxL without software intervention.

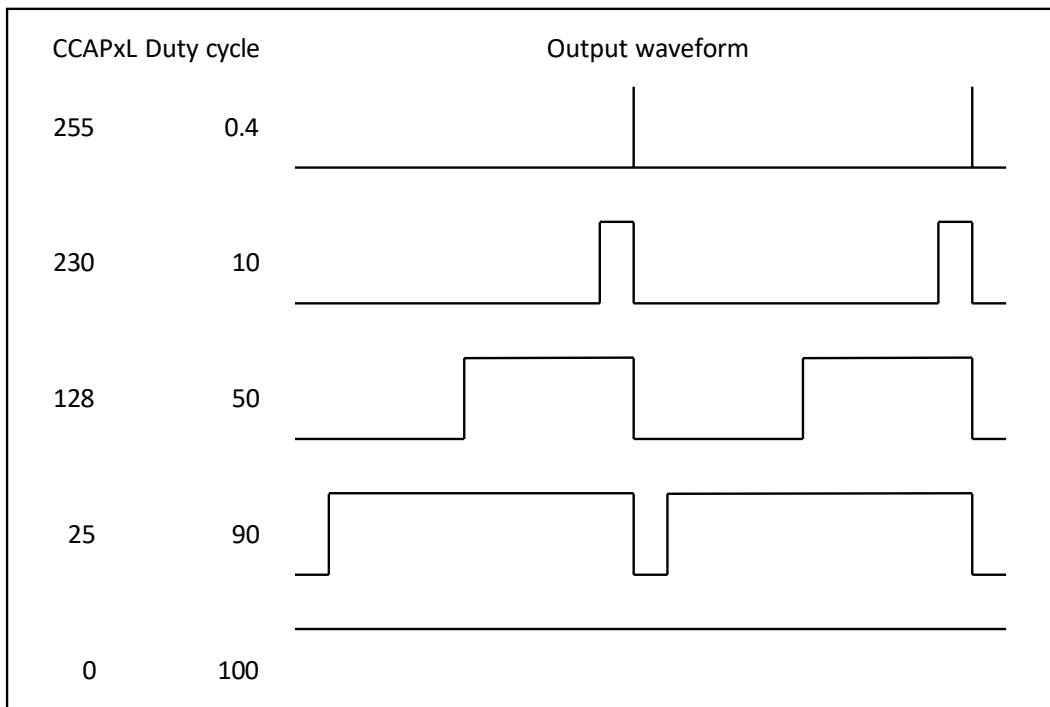
Fig 12.2-4 PCA PWM Functional Diagram



In this mode, PCA timer/counter CNT_L value is constantly compared with the value (CCAPxL) of the low byte compare/capture register. When $CNT_L < CCAPxL$, the output waveform is low. When both match ($CNT_L = CCAPxL$), output waveform goes to high until CNT_L overflow from FFH to 00H, and is still high at the end. In case of overflow, the value of CCAPxH is automatically loaded into CCAPxL, and a new cycle starts.

The value of CCAPxL determines the duty cycle of the current waveform. Determine the duty cycle of the next waveform at the value of CCAPxH. Pulse width modulation can be changed by changing the value in CCAPxL. As shown in the figure, 8-bit values in CCAPxL can be from 0 (100% duty cycle) to 255 (0.4% duty cycle). To change the CCAPxL value without burring, you need to write a new value in the high byte register (CCAPxH). When CL exceeds 0xFF and scrolls to 0x00, this value is automatically loaded into CCAPxL by hardware.

Fig 12.2-5 PCA PWM Output Pulse



To set a compare/capture module to work in PWM mode, you need to set CCAPMx.ECOM and CCAPMx.PWM bit. In addition, the PCA timer/counter can select the input count signal frequency by programming MOD.CSP[2:0]. Enter an 8-bit value in CCAPxL to specify the duty cycle of the first PWM waveform. Entering an 8-bit value in CCAPxH specifies the duty cycle of the second PWM waveform. Set the timer/counter operation control bit(CR.CR) to start the PCA timer/counter.

Tab 12.2-1 Registers

ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	Working mode
X	1	0	0	0	0	X	CCPn rising edge make capture
X	0	1	0	0	0	X	CCPn falling edge make capture
X	1	1	0	0	0	X	CCPn valid edge make capture
1	0	0	1	0	0	X	Software timer
1	0	0	1	1	0	X	High speed output
1	0	0	X	0	1	0	8 bit Pulse Width Modulation

12.3 PCA Module interconnection and control with other modules

12.3.1 ECI Interconnection

ECI input can be the external input port selected by IO MUX(multiplexer), or the filtered output of internal VCMP comparison output.

VCMP output control register in VCMP control module.

12.3.2 PCACAP0

The capture input for channel 0 can be:

- Input port of external IO MUX, or MUX input of external UART RX
- The filtered output of internal VCMP comparison output. the Uart RX channel selected is in PCA capture channel control register (SYSCON_PCACR), the VCMP output control register is in the VCMP control module.

12.3.3 PCACAP1/2/3/4

The capture input of channel 1/2/3/4 can be:

- Input port of external IO MUX, MUX input of external UART RX

the Uart RX channel selected is in PCA capture channel control register (SYSCON_PCACR).

12.4 PCA registers

Address offset	Name	Description	Reset value
0x000	PCA_CR	PCA control register	0x0000 0000
0x004	PCA_MOD	PCA mode register	0x0000 0000
0x008	PCA_CNT	PCA counter register	0x0000 0000
0x00C	PCA_INTCLR	PCA interrupt clear register	0x0000 009F
0x010	PCA_CCAPM0	PCA capture/compare 0 register	0x0000 0000
0x014	PCA_CCAPM1	PCA capture/compare 1 register	0x0000 0000
0x018	PCA_CCAPM2	PCA capture/compare 2 register	0x0000 0000
0x01C	PCA_CCAPM3	PCA capture/compare 3 register	0x0000 0000
0x020	PCA_CCAPM4	PCA capture/compare 4 register	0x0000 0000
0x030	PCA_CCAP0L	8 LSB bits register of PCA capture/compare 0	0x0000 0000
0x034	PCA_CCAP0H	8 MSB bit register of PCA capture/compare 0	0x0000 0000
0x038	PCA_CCAP1L	8 LSB bits register of PCA capture/compare 1	0x0000 0000
0x03C	PCA_CCAP1H	8 MSB bit register of PCA capture/compare 1	0x0000 0000
0x040	PCA_CCAP2L	8 LSB bits register of PCA capture/compare 2	0x0000 0000
0x044	PCA_CCAP2H	8 MSB bit register of PCA capture/compare 2	0x0000 0000
0x048	PCA_CCAP3L	8 LSB bits register of PCA capture/compare 3	0x0000 0000
0x04c	PCA_CCAP3H	8 MSB bit register of PCA capture/compare 3	0x0000 0000
0x050	PCA_CCAP4L	8 LSB bits register of PCA capture/compare 4	0x0000 0000
0x054	PCA_CCAP4H	8 MSB bit register of PCA capture/compare 4	0x0000 0000
0x058	PCA_CCAPO	PCA PWM high speed output register	0x0000 0000
0x05C	PCA_POCR	PCA Port output control register	0x0000 0000
0x060	PCA_CCAP0	16 bit register of PCA capture/compare 0	0x0000 0000
0x064	PCA_CCAP1	16 bit register of PCA capture/compare 1	0x0000 0000
0x068	PCA_CCAP2	16 bit register of PCA capture/compare 2	0x0000 0000
0x06C	PCA_CCAP3	16 bit register of PCA capture/compare 3	0x0000 0000
0x070	PCA_CCAP4	16 bit register of PCA capture/compare 4	0x0000 0000

12.5 PCA registers description**12.5.1 PCA control register (PCA_CR)**

Register	Address offset	Access	Reset value	Description
PCA_CR	0x000	RW	0x0000_0000	PCA control register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CF	CR	Reserved	CCF4	CCF3	CCF2	CCF1	CCF0

PCA control register (PCA_CR)

Bit	Access	Description
[31:8]	-	Reserved
[7]	R	CF: PCA Counter overflow flag 0: no overflow 1: Counter overflow occurred PCA count overflows, CF is set by hardware When the CFIE bit of the MOD register is 1, the CF flag can generate an interrupt
[6]	RW	CR: PCA Counter run control bit 0: Off PCA Counter count 1: Start PCA Counter count
[5]	-	Reserved
[4]	R	CCF4: PCA Counter Module 4 Compare/Capture Flag Bit: This bit is set by hardware when a match or capture occurs This flag generates a PCA interrupt when CCAPM4.CCIE is set 0: no match or capture 1: match or capture occurred
[3]	R	CCF3: PCA Counter Module 3 Compare/Capture Flag Bit: This bit is set by hardware when a match or capture occurs This flag generates a PCA interrupt when CCAPM3.CCIE is set 0: no match or capture 1: match or capture occurred
[2]	R	CCF2: PCA Counter Module 2 Compare/Capture Flag Bit: This bit is set by hardware when a match or capture occurs This flag generates a PCA interrupt when CCAPM2.CCIE is set 0: no match or capture 1: match or capture occurred
[1]	R	CCF1: PCA Counter Module 1 Compare/Capture Flag Bit: This bit is set by hardware when a match or capture occurs This flag generates a PCA interrupt when CCAPM1.CCIE is set 0: no match or capture 1: match or capture occurred
[0]	R	CCF0: PCA Counter Module 0 Compare/Capture Flag Bit: This bit is set by hardware when a match or capture occurs This flag generates a PCA interrupt when CCAPM0.CCIE is set 0: no match or capture 1: match or capture occurred

12.5.2 PCA mode register (PCA_MOD)

Register	Address offset	Access	Reset value	Description
PCA_MOD	0x004	RW	0x0000_0000	PCA mode register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CIDL	Reserved			CPS			CFIE

PCA mode register (PCA_MOD)

Bit	Access	Description
[31:8]	-	Reserved
[7]	RW	CIDL: :Idle mode IDLE,PCA stop working 0: In sleep mode(Sleep),PCA continues to work 1: In sleep mode(Sleep),PCA stops working
[6:4]	-	Reserved
[3:1]	RW	CPS: Clock frequency division selection and clock source selection 000:PCLK/32 001:PCLK/16 010:PCLK/8 011:PCLK/4 100:PCLK/2 101:Timer0 overflow 110:Timer1 overflow 111:ECIExternal clock, clockPCLKFour-frequency sampling
[0]	RW	CFIE: PCA Counter interrupt enable control signal 0: disable interrupt 1: Enable interrupt

12.5.3 PCA counter register (PCA_CNT)

Register	Address offset	Access	Reset value	Description
PCA_CNT	0x008	RW	0x0000_0000	PCA counter register

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

PCA counter register (PCA_CNT)

Bit	Access	Description
-----	--------	-------------

[31:8]	-	Reserved
[15:0]	RW	CNT: :The value of the timer counter: Only when PCA is stopped,CNT can be written, otherwise writing is invalid

12.5.4 PCA interrupt clear register (PCA_INTCLR)

Register	Address offset	Access	Reset value	Description
PCA_INTCLR	0x00C	W	0x0000_0000	PCA interrupt clear register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CF	Reserved		CCF4	CCF3	CCF2	CCF1	CCF0

PCA interrupt clear register (PCA_INTCLR)

Bit	Access	Description
[31:8]	-	Reserved
[7]	W	CF: :PCA Counter overflow flag clear 0: write 0 invalid 1: software writes 1 clears the corresponding flag
[6:5]	W	CF: :Reserved
[4]	W	CCF4: :PCA Counter Module 4 Compare/Capture flag clear 0: write 0 invalid 1: software writes 1 clears the corresponding flag
[3]	W	CCF3: :PCA Counter Module 3 Compare/Capture flag clear 0: write 0 invalid 1: software writes 1 clears the corresponding flag
[2]	W	CCF2: :PCA Counter Module 2 Compare/Capture flag clear 0: write 0 invalid 1: software writes 1 clears the corresponding flag
[1]	W	CCF1: :PCA Counter Module 1 Compare/Capture flag clear 0: write 0 invalid 1: software writes 1 clears the corresponding flag
[0]	W	CCF0: :PCA Counter Module 0 Compare/Capture flag clear 0: write 0 invalid 1: software writes 1 clears the corresponding flag

12.5.5 PCA capture/compare register (PCA_CCAPM0~4)

Register	Address offset	Access	Reset value	Description
PCA_CCAPM0	0x010	RW	0x0000_0000	PCA capture/compare 0 register
PCA_CCAPM1	0x014	RW	0x0000_0000	PCA capture/compare 1 register
PCA_CCAPM2	0x018	RW	0x0000_0000	PCA capture/compare 2 register
PCA_CCAPM3	0x01C	RW	0x0000_0000	PCA capture/compare 3 register
PCA_CCAPM4	0x020	RW	0x0000_0000	PCA capture/compare 4 register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	ECOM	CAPP	CAPN	MAT	TOG	PWM	CCIE

PCA capture/compare register (PCA_CCAPM0~4)

Bit	Access	Description
[31:7]	-	Reserved
[6]	RW	ECOM: :Enable comparator function control bits 0: disable compare strong function 1: Enable comparator function Set ECOM when PCA is used for software counter, high speed output and PWM mode; When writing CCAMPx or CCAMPx register will automatically set ECOM; when writing The CCAMPx register automatically clears the ECOM bit;
[5]	RW	CAPP: :Positive edge capture control bit 0: Disable rising edge capture 1: enable rising edge capture
[4]	RW	CAPN: :negative edge capture control bit 0: Disable falling edge capture 1: enable falling edge capture
[3]	RW	MAT: :Allow match control bits 0: disable matching function; 1: Once the PCA count matches the value of the module (compare/capture register), the interrupt flag CCFx of the CR register is set
[2]	RW	TOG: :Toggle control bits 0: Toggle function disabled 1: Once the PCA counter value matches the module (compare/capture register) value, the PCA_CHx pin flips
[1]	RW	PWM: :PWM control bits: 0: Disable PWM Pulse width modulation function 1: Enable PC_CHx pin as PWM output PWM function is valid only when CCAPMx[6:0]=100 0010'b
[0]	RW	CCIE: :PCAEnable interrupt: 0:PCA Compare/Capture function interrupt disable 1: Enable compare/capture interrupt

12.5.6 8 MSB bits register of PCA capture/compare (PCA_CCAP0~4H)

Register	Address offset	Access	Reset value	Description
PCA_CCAP0H	0x034	RW	0x0000_0000	8 MSB bit register of PCA capture/compare 0
PCA_CCAP1H	0x03C	RW	0x0000_0000	8 MSB bit register of PCA capture/compare 1
PCA_CCAP2H	0x044	RW	0x0000_0000	8 MSB bit register of PCA capture/compare 2
PCA_CCAP3H	0x04C	RW	0x0000_0000	8 MSB bit register of PCA capture/compare 3
PCA_CCAP4H	0x054	RW	0x0000_0000	8 MSB bit register of PCA capture/compare 4

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

High 8 bit register of PCA capture/compare (PCA_CCAP0~4H)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	<p>CCAPx[7:0] :compare/capture mode high 8 bit register</p> <p>8 MSB bits used to hold 16 bit capture count value when PCA mode is used to compare/capture mode , writing to the CCAPxH register will automatically set the ECOM bit in the CCAPMx register.</p> <p>Used to control output duty load register when PCA mode is used in PWM mode register, when counter 8 LSB bits overflow, load register is automatically updated to PWM compare register</p>

12.5.7 8 LSB bits register of PCA capture/compare (PCA_CCAP0~4L)

Register	Address offset	Access	Reset value	Description
PCA_CCAP0L	0x030	RW	0x0000_0000	8 LSB bit register of PCA capture/compare 0
PCA_CCAP1L	0x038	RW	0x0000_0000	8 LSB bit register of PCA capture/compare 1
PCA_CCAP2L	0x040	RW	0x0000_0000	8 LSB bit register of PCA capture/compare 2
PCA_CCAP3L	0x048	RW	0x0000_0000	8 LSB bit register of PCA capture/compare 3
PCA_CCAP4L	0x050	RW	0x0000_0000	8 LSB bit register of PCA capture/compare 4

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

low 8 bit register of PCA capture/compare (PCA_CCAP0~4L)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	CCAPx[7:0] : :compare/capture mode low 8 bit register When PCA mode is used to compare/capture mode, low 8 bits used to hold 16 bit capture count value bit; writing CCAPxH register automatically clears ECOM bit of register CCAPMx.Used to control the output duty compare register when PCA mode is used in PWM mode, in PWM mode, the value of the low 8 bits of the counter is less than the value of CCAPx[7:0] PWM output low level, otherwise PWM output high level.

12.5.8 Compare Capture 16 Bit Register (PCA_CCAP0~4)

Register	Address offset	Access	Reset value	Description
PCA_CCAP0	0x060	RW	0x0000_0000	16 bit register of PCA capture/compare 0
PCA_CCAP1	0x064	RW	0x0000_0000	16 bit register of PCA capture/compare 1
PCA_CCAP2	0x068	RW	0x0000_0000	16 bit register of PCA capture/compare 2
PCA_CCAP3	0x06C	RW	0x0000_0000	16 bit register of PCA capture/compare 3
PCA_CCAP4	0x070	RW	0x0000_0000	16 bit register of PCA capture/compare 4

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

Capture/Compare 16 Bit Register(PCA_CCAP0~4)

Bit	Access	Description
[31:16]	-	Reserved
[15:0]	RW	CCAPx : :compare/capture mode 16 bit register Used to hold 16 bit capture count value when PCA is used to compare/capture mode; writing CCAPx register will Set bit ECOM of register CCAPMx. writing CCAPx register is equivalent to writing CCAPxL and CCAPxH two 8 bit registers. In compare/capture mode, this register can be read and written directly;in PWM mode, use CCAPxL and CCAPxH register.

12.5.9 Compare High Speed Output Flag Register (PCA_CCAPO)

Register	Address offset	Access	Reset value	Description
PCA_CCAPO	0x058	RW	0x0000_0000	Compare High Speed Output Flag Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			CCAPO4	CCAPO3	CCAPO2	CCAPO1	CCAPO0

Compare High Speed Output Flag Register(PCA_CCAPO)

Bit	Access	Description
[31:5]	-	Reserved
[4]	RW	CCAPO4 : :High-speed mode comparison module 4 Output value of 0: Output0 1: output1
[3]	RW	CCAPO3 : :High-speed mode comparison module 3 Output value of 0: Output0 1: output1
[2]	RW	CCAPO2 : :High-speed mode comparison module 2 Output value of 0: Output0 1: output1
[1]	RW	CCAPO1 : :High-speed mode comparison module 1 Output value of 0: Output0 1: output1
[0]	RW	CCAPO0 : :High-speed mode comparison module 0 Output value of 0: Output0 1: output1

12.5.10 PCA Port output control register (PCA_POCR)

Register	Address offset	Access	Reset value	Description
PCA_POCR	0x05C	RW	0x0000_0000	PCA Port output control register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			POINV4	POINV3	POINV2	POINV1	POINV0
7	6	5	4	3	2	1	0
Reserved			POE4	POE3	POE2	POE1	POE0

PCA Port output control register (PCA_POCR)

Bit	Access	Description
[31:13]	-	Reserved
[12]	RW	POINV4 : :compare channel 4 output polarity inversion 0 =Disable PWM0 Output Polarity Inversion 1 =Enable PWM0 Output Polarity Inversion

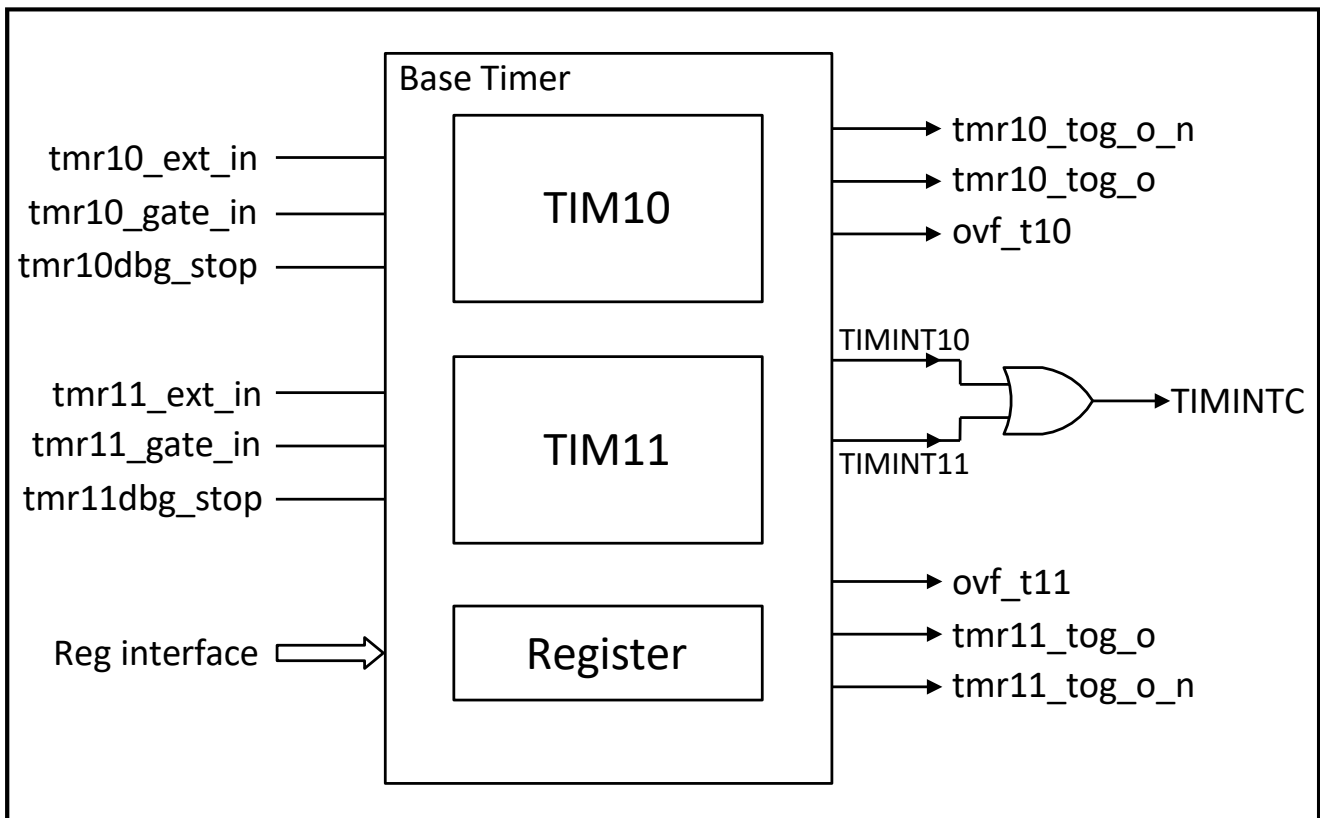
[11]	RW	POINV3: :compare channel 3 output polarity inversion 0 =Disable PWM0 Output Polarity Inversion 1 =Enable PWM0 Output Polarity Inversion
[10]	RW	POINV2: :compare channel 2 output polarity inversion 0 =Disable PWM0 Output Polarity Inversion 1 =Enable PWM0 Output Polarity Inversion
[9]	RW	POINV1: :compare channel 1 output polarity inversion 0 =Disable PWM0 Output Polarity Inversion 1 =Enable PWM0 Output Polarity Inversion
[8]	RW	POINV0: :compare channel 0 output polarity inversion 0 =Disable PWM0 Output Polarity Inversion 1 =Enable PWM0 Output Polarity Inversion
[7:5]	-	Reserved
[4]	RW	POE4: :Compare channel 4 Output enable 0: Output disabled 1: Output enable
[3]	RW	POE3: :Compare channel 3 Output enable 0: Output disabled 1: Output enable
[2]	RW	POE2: :Compare channel 2 Output enable 0: Output disabled 1: Output enable
[1]	RW	POE1: :Compare channel 1 Output enable 0: Output disabled 1: Output enable
[0]	RW	POE0: :Compare channel 0 Output enable 0: Output disabled 1: Output enable

13 Base timer(TIM10/TIM11)

Base Timer contains two timers TIM10/11. TIM10/11 has the same function. TIM10/11 is a synchronous timing/counter, which can be used as the timing/counter of the 16/32 bit automatic reload function, or as the timing/counter of the 16/32 bit non-reload function. TIM10/11 can count external pulses or realize system timing.

13.1 Base Timer introduction

Fig 13.1-1 Base Timer Block Diagram



13.2 Base Timer description

Each TIM10/11 timer/counter has independent control start signal, external input clock and gate control signal.

When TIM10/11 uses EXT and GATE to perform counting function, EXT is used for the external input clock signal of the counter, and GATE is used for effective level count enable signal. When the gate control function is enabled, the counter will count only when the external input GATE level is valid, otherwise the counter is in the hold state. Gate control enable is controlled using TIMx_CR.GATE_EN. By default, the gate control function is off. Gate level selection is controlled using TIMx_CR.GATE_P. The default high level is the effective level of gate control; after setting TIMx_CR.GATE_P to 1, the gate low level is the effective level.

When TIM10/11 uses PCLK and GATE for timing function, PCLK is used for the internal input clock signal of the timer, and GATE can be used for the effective level timing enable signal. When the gate control power is enabled, the timer will count only when and only when the external input GATE level is valid, otherwise the timer is in the stop state of the timer. Gate control enable TIMx_CR.GATE_EN control. The default gate control function is off. The gate control level selects to use TIMx_CR.GATE_P control. The default high level is the effective level of gate control; after setting to 1, the effective level of rear gate control is low. The timing function can be configured with pre-frequency division. TIMx_CR.TMR_PRE controls the frequency division ratio.

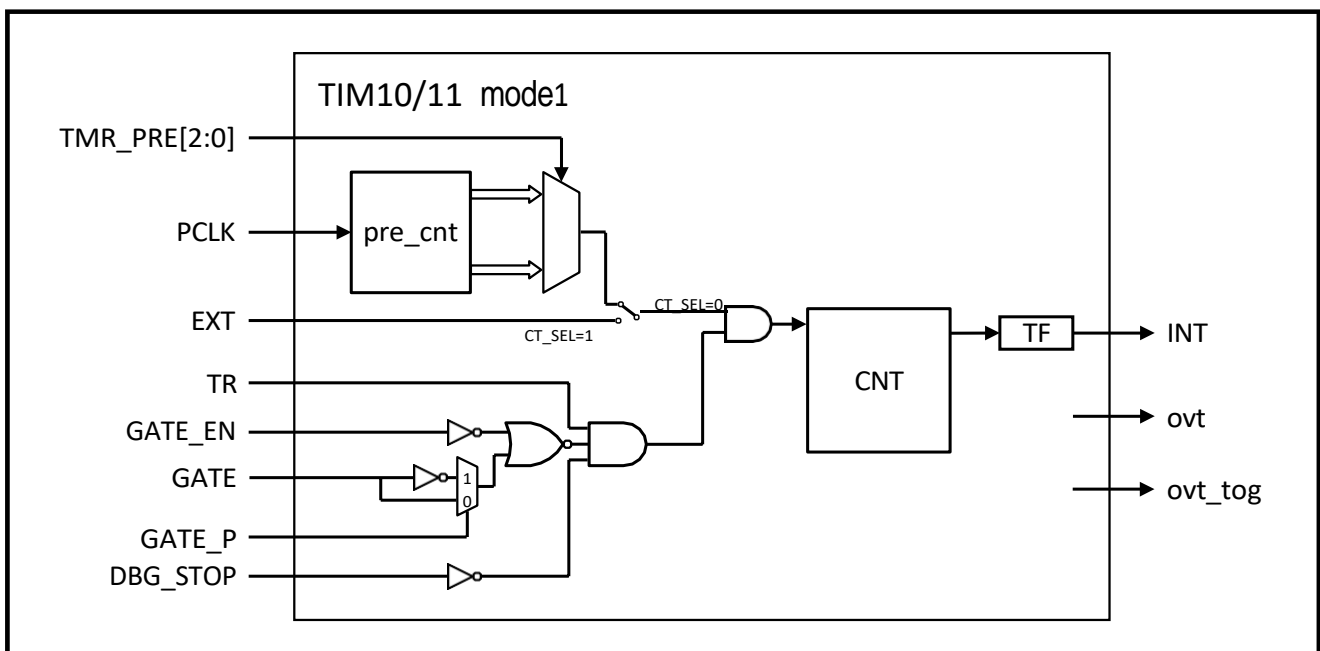
TMR_PRE [2:0]	000	001	010	011	100	101	110	111
Frequency division ratio	1	2	4	8	16	32	64	128

TIM10/11 supports two functions of timing/counter, which can be configured by setting CT_SEL in the timer control register(TIMx_CR). Each function supports 2 modes, mode 1 is 16/32-bit free count mode, mode 2 is 16 /32-bit reload mode.

13.2.1 Mode 1 free counting

When the count reaches the maximum value (16-bit Max=0xFFFF, 32-bit Max=0xFFFFFFFF), an interrupt will be generated after overflow, and the timer/counter will be cleared, and then the count will continue

Fig 13.2-1 Base Timer Mode 1 Diagram

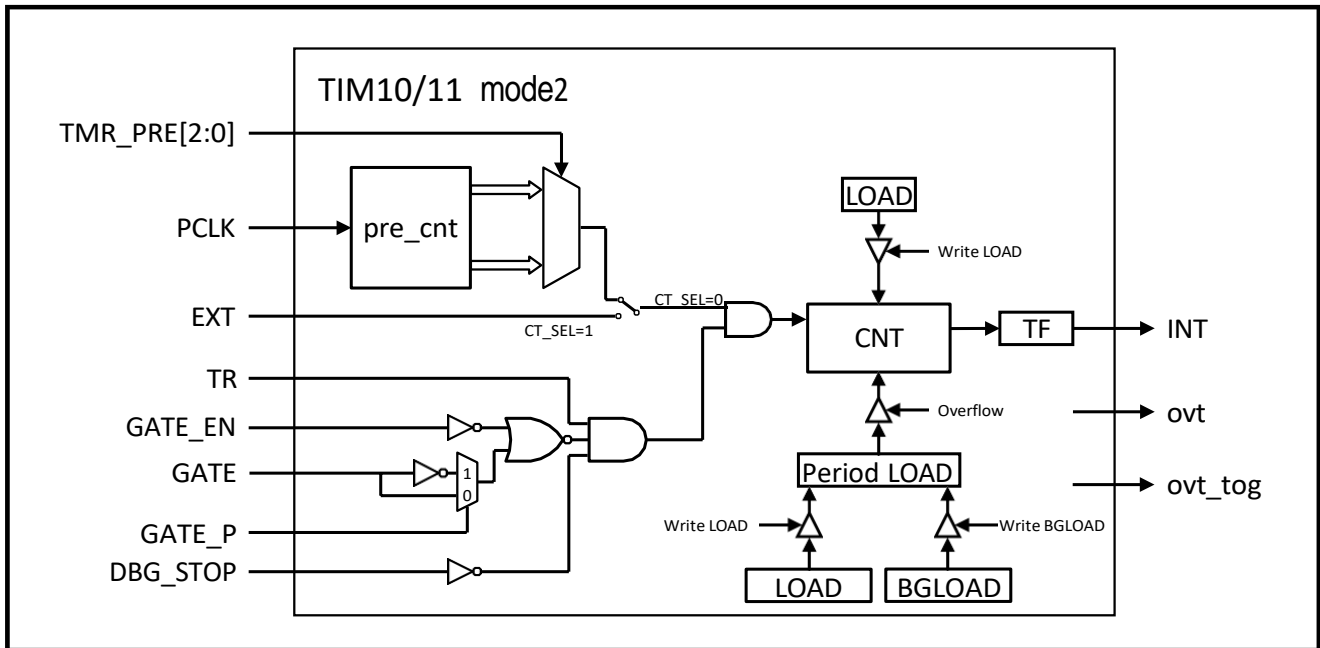


13.2.2 Mode 2 reload

In reload mode, overflow occurs after the count reaches the maximum value, and an interrupt is generated. The value of the timer/counter is loaded as the value of BGLOAD, and then continue to

count up. In the reload mode, the software processing speed needs to be considered when the timing time is set to be small, otherwise the interrupt will not be processed in time and cause the interrupt loss.

Fig 13.2-2 Base Timer Mode 2 Diagram

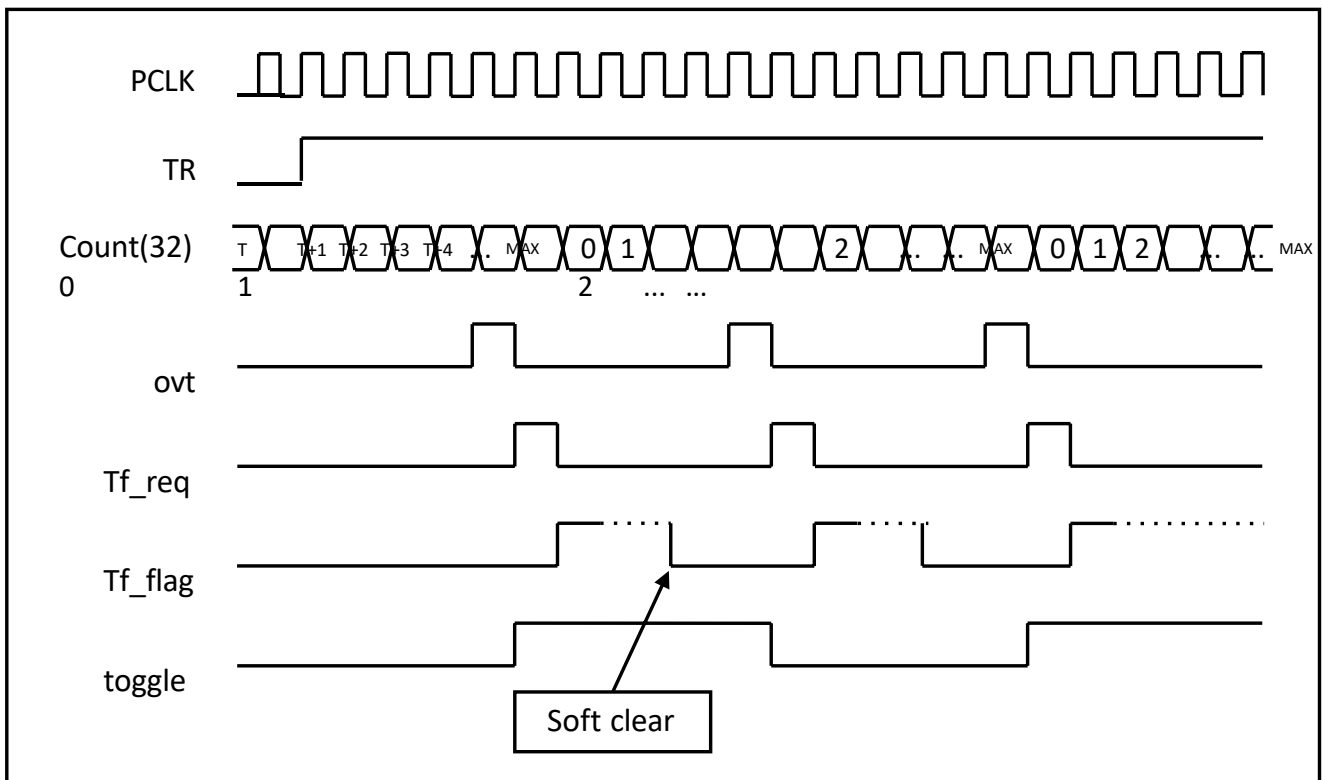


When the corresponding timer TIMx_CR.TR is set to 1, the timer starts to run. In mode 1, counting starts from the initial value set by the register after startup, an overflow interrupt will be generated after counting to the maximum, and then counting continues from 0. in Mode2, count up from the initial value of the register CNT after startup, and after counting to the maximum value, generate interrupt and reload the value of the reload register BGLOAD to the counter CNT, and continue to count up. Regardless of the free counting mode or the reload mode, as long as the value of LOAD is written, the value of the timer/counter will be updated immediately, and then continue to count up.

13.2.3 Counting function

The counting function is used to measure the number of times an event occurs. In the counting function, the counter is accumulated once at the falling edge of each corresponding input clock(EXT). The input signal is sampled by the internal PCLK, so the external input clock frequency cannot exceed the system’s PCLK clock. Counting to the maximum value will overflow and cause an interrupt. The interrupt flag needs to be cleared by software.

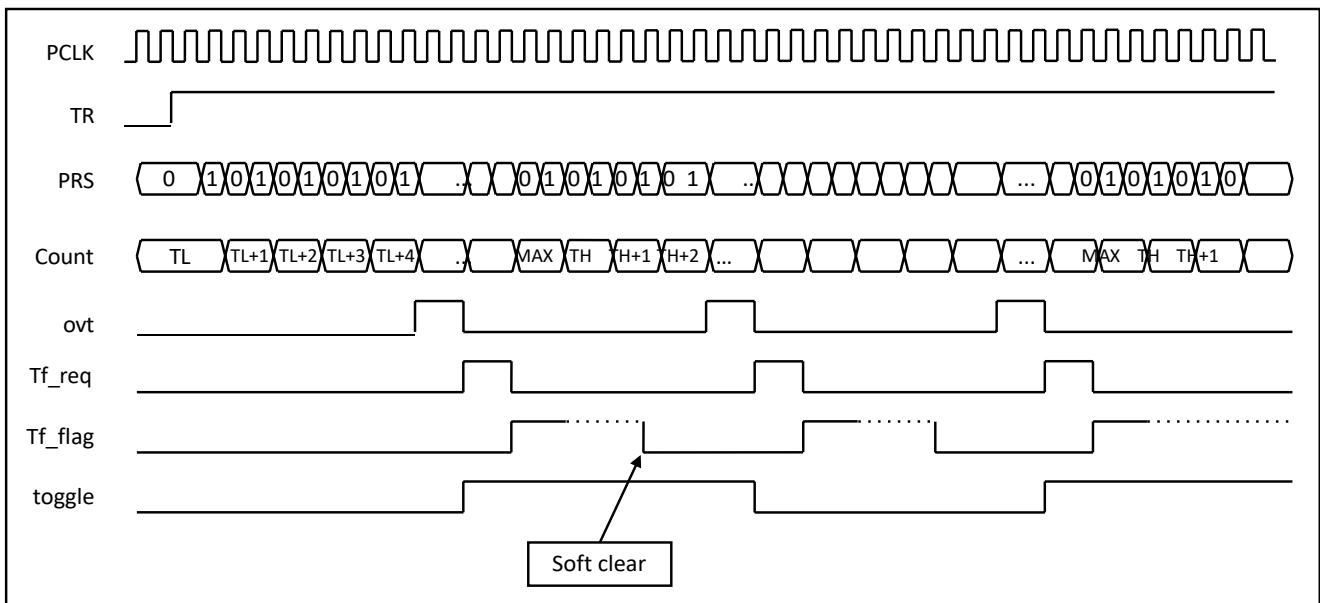
Fig 13.2-3 32 bit mode 1 Timing diagram(max=0xFFFF FFFF)



13.2.4 Timing function

The timing function is used to generate interval timing. In the timing function, the timer has a pre-frequency division, and the timer accumulates one clock based on PCLK clock after pre-division. When counting to the maximum value, it will overflow and generate an interrupt. The interrupt flag needs to be cleared by software.

Fig 13.2-4 32 bit mode 2 Timing diagram



13.2.5 Buzzer Features

The function of driving Buzzer can be realized through the flip output function of the timer. When TIMx_CR.TOG_EN is 1, TOG and TOGN outputs are inverted. Setting TIMx_CR.TOG_EN to 0 can set port TOG, TOGN output to 0 at the same time. When the counting clock is 4M, Buzzer output Timer reload mode with different frequency configuration as follows (16 bits Max= 0xFFFF):

Tab 13.2-1 Output frequency

Buzzer frequency	Counting period	Count value	Reload	CNT initial	LOAD reload
1KHz	0.5ms	2000	63536	0xF830	0xF830
2KHz	0.25ms	1000	64536	0xFC18	0xFC18
4KHz	0.125ms	500	65036	0xFE0C	0xFE0C

13.3 Base Timer interconnection

13.3.1 GATE Interconnection

GATE input can be input directly from the port, or input the RX signal of UART/LPUART; the output of VCMP can also be configured as GATE signal. GATE of TIM10/11 can be configured.

Through the internal interconnection configuration, the automatic identification of UART baud rate can be realized, the pulse width of VCMP comparison output can be measured, and the external control counting can be realized.

RX input configuration selection is in SYSCON_PORTCR register control, VCMP control is controlled in VCMP_OUTCFG register. When selecting the port, only one of UART/LPUART input selection and VCMP output selection can be selected as gate control input. VCMP's output selection has the highest priority.

13.3.2 Toggle Output Interconnection

Toggle output of TIM10tmr10_tog_o to internal module UART1, control the baud rate of UART1;Flip output tmr11 of TIM11_tog_O to the internal module UART2, control the baud rate of UART2;The flip output of TIM10/11 is also output to the port, which can drive Buzzer to control the buzzer.

13.4 Base Timer registers

	Offset address	Description
TIM10	0x00	TIM10 Offset address
TIM11	0x100	TIM11 Offset address

Tab 13.4-1 Base Timer register mapping

offset	Register	Reset value	Description
TIM10/TIM11 base address : 0x 4000 1800			
0x00	TIMx_CR	0x0000_0000	control register
0x04	TIMx_LOAD	0x0000_0000	32 bit Immediate reload register
0x08	TIMx_CNT	0x0000_0000	Counter Register
0x0C	TIMx_RAWINTSR	0x0000_0000	Raw interrupt status register
0x10	TIMx_MSKINTSR	0x0000_0000	Interrupt flag register
0x14	TIMx_INTCLR	0x0000_0000	Interrupt clear register
0x18	TIMx_BGLOAD	0x0000_0000	32-bit cycle reload register

Note: x= 10/11

13.5 Base Timer registers description

13.5.1 Control Register(TIMx_CR)

Register	Address offset	Access	Reset value	Description			
TIMx_CR	0x00	RW	0x0000_0000	Control Register			
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				GATE_P	GATE_EN	TOG_EN	CT_SEL
7	6	5	4	3	2	1	0
TR	MODE	INTEN	TMR_SIZE	ONESHOT	TMR_PRSC[2:0]		

TIM1 auto-reload register(TIM1_ARR)

Bit	Access	Description
[31:12]	-	Reserved, read as 0
[11]	RW	GATE_P: port GATE polarity control, default high level GATE is valid, set to 1 after low level is valid 0: active high 1: active low

[10]	RW	GATE_EN: Timer Gating 0: No gate control,Timer works when TR = 1; 1: work only if port GATE is valid and TR = 1
[9]	RW	TOG_EN: TOG Output Enable 0:TOG,TOGN simultaneously output 0 1:TOG and TOGN output signals of opposite phase. Available for use by buzzer
[8]	RW	CT_SEL: Counter/Timer function selection 0: Timer function, the timer is counted by PCLK. 1: Counter function, the counter counts by the falling edge of the external input. external input is sampled by PCLK, the external input clock frequency is lower than 1/2 sampling clock.
[7]	RW	TR: Timer run control 0: Timer stop 1: Timer running
[6]	RW	MODE: Timer working mode 0: Mode 1 Counter/Timer 1: Mode 2 Auto-reload counter/Timer
[5]	RW	INTEN: Interrupt enable control, enable interrupt after writing 1
[4]	RW	TMR_SIZE: TimerSize = 0 : max count value = 0xFFFF; TimerSize = 1 : max count value = 0xFFFFFFFF;
[3]	RW	ONESHOT: Counter run once enable 0: repeat mode 1: oneshot mode
[2:0]	RW	TMR_PRSC[2:0]: TIM Prescaler selection. 000: Frequency divider 1; 001: frequency division 2; 010: frequency division 4; 011: frequency division 8; 100: frequency division 16; 101: frequency division 32; 110: frequency division 64; 111: frequency division 128

13.5.2 Immediate Reload Register(TIMx_LOAD)

Register	Address offset	Access	Reset value	Description			
TIMx_LOAD	0x04	RW	0x0000_0000	32 bit Immediate reload register			
31	30	29	28	27	26	25	24
LOAD[31:24]							
23	22	21	20	19	18	17	16
LOAD[23:16]							
15	14	13	12	11	10	9	8
LOAD[15:8]							
7	6	5	4	3	2	1	0
LOAD[7:0]							

Immediate Reload Register(TIMx_LOAD)

Bit	Access	Description
[31:0]	RW	<p>LOAD[31:0]: immediate reload register Writing to this register immediately updates the value of the counter register CNT</p> <p>Note: readTIMx_LOAD and TIMx_BGLOAD, you can read the latest updated LOAD or BGLOAD register value</p>

13.5.3 Counter Register(TIMx_CNT)

Register	Address offset	Access	Reset value	Description			
TIMx_CNT	0x08	R	0x0000_0000	Counter Register			
31	30	29	28	27	26	25	24
CNT[31:24]							
23	22	21	20	19	18	17	16
CNT[23:16]							
15	14	13	12	11	10	9	8
CNT[15:8]							
7	6	5	4	3	2	1	0
CNT[7:0]							

Counter Register(TIMx_CNT)

Bit	Access	Description
[31:0]	R	CNT[31:0]: Counter Register

13.5.4 Raw interrupt status register(TIMx_RAWINTSR)

Register	Address offset	Access	Reset value	Description			
TIMx_RAWINTSR	0x0c	R	0x0000_0000	Raw interrupt status register			
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							RIS[0]

Raw interrupt status register(TIMx_RAWINTSR)

Bit	Access	Description
[31:1]	-	Reserved, read as 0
[0]	R	RIS[0]: Interrupt flag, hardware setting; IntEnable = 0/1 (read interrupts)

13.5.5 Interrupt flag register(TIMx_MSKINTSR)

Register	Address offset	Access	Reset value	Description
TIMx_MSKINTSR	0x10	R	0x0000_0000	Interrupt flag register

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

Interrupt flag register(TIMx_MSKINTSR)

Bit	Access	Description
[31:1]	-	Reserved, read as 0
[0]	R	TF: If INTEN=1, the interrupt register can be read

13.5.6 Interrupt clear register(TIMx_INTCLR)

Register	Address offset	Access	Reset value	Description
TIMx_INTCLR	0x14	W	0x0000_0000	Interrupt clear register

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

Interrupt clear register(TIMx_INTCLR)

Bit	Access	Description
[31:1]	-	Reserved, read as 0
[0]	W	INTCLR: Interrupt flag cleared, write 1 cleared, write 0 invalid

13.5.7 BackGround Cycle reload register(TIMx_BGLOAD)

Register	Address offset	Access	Reset value	Description
TIMx_BGLOAD	0x18	RW	0x0000_0000	32-bit cycle reload register

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

BackGround Cycle reload register(TIMx_BGLOAD)

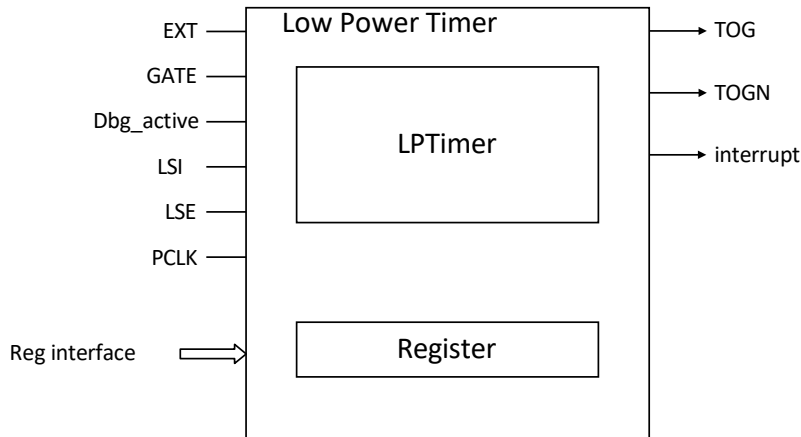
Bit	Access	Description
-----	--------	-------------

[31:0]	RW	<p>BGLOAD[31:0]: BackGround Cycle reload register writing to this register does not update the value of counter register CNT immediately. will reload BGLOAD value into CNT register only when CNT value overflows.</p> <p>Note read <i>TIMx_LOAD</i> and <i>TIMx_BGLOAD</i>, you can read the latest updated <i>LOAD</i> or <i>BGLOAD</i> Register value</p>
--------	----	---

14 Low Power Timer(LPTIM)

LPTIM is an asynchronous 16-bit timing/counter. It can time/count through the internal low speed RC clock (LSI) or the external low speed crystal oscillation clock(LSE). After the system clock is turned off,the system can still be waken up in the low power consumption mode by interrupt.

Fig 14.0-1 LPTIMER diagram



14.1 LPTIM description

LPTIM has independent control start signal, external input clock and gate control signal.

LPTIM uses EXT and GATE for counting function, EXT is used for external input clock signal of counter, and GATE is effective level counting enable signal.

The timer of LPTIM supports two working modes, select the working mode by setting MODE in the timer control register(LPTIM_CR).

- Mode1 is 16-bit free counting mode, and the counter starts counting from the value set by LPTIM_LOAD, and the counting value starts from 0x0000 after overflow.
- Mode2 is 16-bit reload mode, LPTIM will automatically load the value of reload register LPTIM_LOAD to the counter when starting. After overflow, it will automatically load the value of Period LOAD into the counter. When the user writes LPTIM_LOAD or LPTIM_BGLOAD, the value of "Period LOAD" is updated. The value of "Period LOAD" is the value of "LPTIM_LOAD" or "LPTIM_BGLOAD" last updated by the user.

LPTIM can select three kinds of clocks as timer clock, which can be selected through the control register LPTIM_CR.TCK_SEL. Default selection PCLK. The clock selection is shown in the table below:

TCK_SEL	00	01	10	11
timer clock	PCLK	PCLK	LSE	LSI
Read timer count value	read synchronized	no sync	read synchronized	read synchronized

When selecting the corresponding clock source, and then setting TCK_EN to 1, the counting clock source of the counter can be turned on.

After the clock source is selected and turned on, set TIM_RUN of the corresponding timer to 1 and then the timer starts running.

For mode 1 and mode 2, if the LOAD value is set, the count value will be immediately updated to the LOAD value at any time, and the counter will count again from the LOAD value. The priority of setting the LOAD value is higher than the priority of other counters being updated.

In mode 2, set the value of BGLOAD. The set value will be updated to the counter only after the counter overflows.

Mode 1: If the LOAD value is not set, the counter starts counting from 0, and generates an overflow interrupt after counting to the maximum 0xFFFF. After the counter counts to the maximum 0xFFFF, the counter starts counting from 0 again.

Mode 2: If the LOAD value is not set, the counter starts counting from 0, and generates an overflow interrupt after counting to the maximum 0xFFFF. After the counter counts to the maximum 0xFFFF, the count value will be updated to the value of Period LOAD, and then counted up.

Fig 14.1-1 LPTIMER mode 1

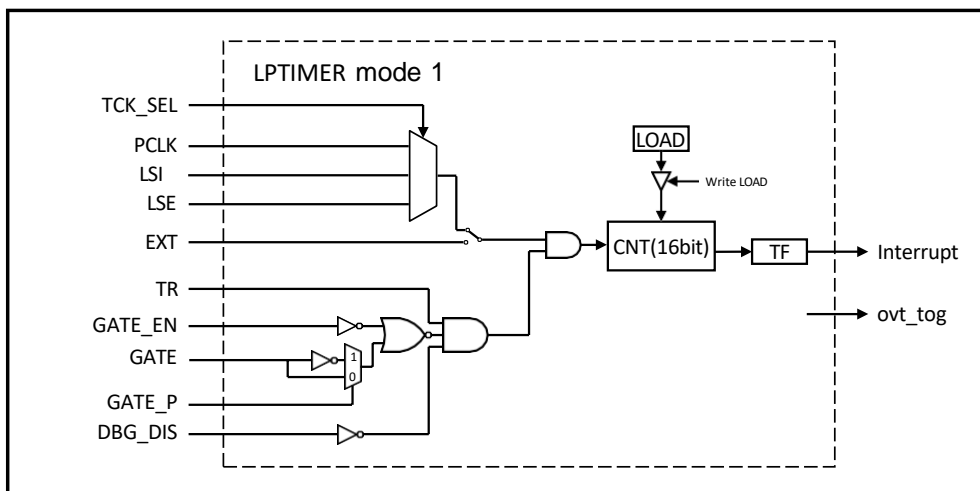
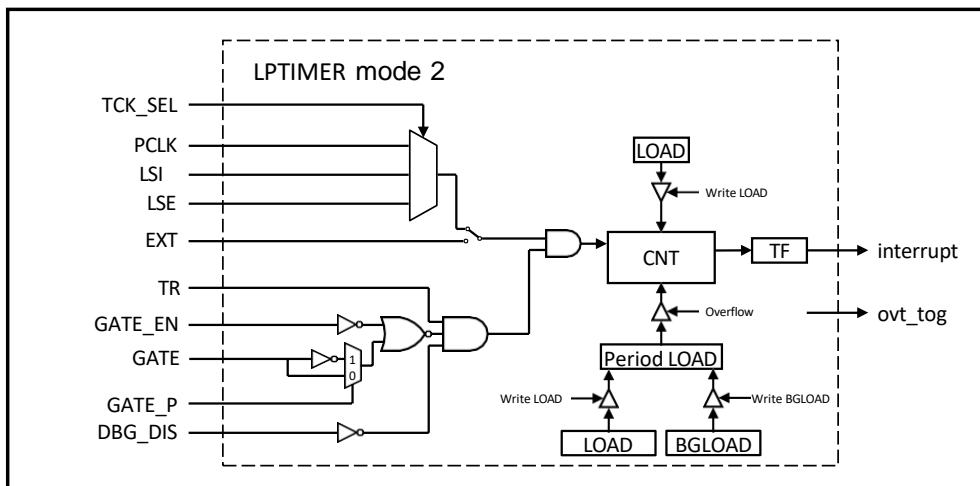


Fig 14.1-2 LPTIMER mode 2



14.1.1 Counting function

The counting function is used to measure the number of times an event occurs. In the counting function, the counter is accumulated once at the rising edge of each corresponding input clock. The input signal is sampled by the internal count clock, so the external input clock frequency cannot exceed the system count clock. Counting to the maximum value will overflow and cause an interrupt.

14.1.2 Timing function

The timing function is used to generate interval timing. In the timing function, the timer accumulates once per clock, and the maximum value will overflow and generate an interrupt.

14.2 LPTIM Interconnection

14.2.1 GATE Interconnection

GATE input can be input directly from the port, or input RX signal of UART;

Through the internal interconnection configuration, the automatic identification of UART baud rate can be realized, the pulse width of VCMP comparison output can be measured, and the external control counting can be realized.

RX input selection configuration is controlled in SYSCON_PORTCR register, and VCMP control is controlled in VCMP control register.

14.2.2 EXT Interconnection

The EXT input can be input directly from the port, also can be configured as the icomparison outcome of VCMP module.

VCMP pulse count can be measured through internal interconnection configuration. VCMP output control register is in VCMP control module.

14.2.3 TOGGLE Output Interconnection

The flip output of LPTIM is output to the port, which can drive BUZZER to control buzzer.

14.3 LPTIM Registers

Tab 14.3-1 LPTIM register mapping

offset	Register	Reset value	Description
LPTIM base address : 0x 4000 4400			
0x00	LPTIM_CNTVAL	0x0000_0000	LPTIM Count Value read-only Register
0x04	LPTIM_CR	0x0000_0000	LPTIM Control Register
0x08	LPTIM_LOAD	0x0000_0000	LPTIM Immediate reload register
0x0C	LPTIM_INTSR	0x0000_0000	LPTIM Interrupt Register
0x10	LPTIM_INTCLR	0x0000_0000	LPTIM Interrupt Clear Register
0x14	LPTIM_BGLOAD	0x0000_0000	LPTIM Cycle reload register

14.4 LPTIM Registers description

14.4.1 LPTIM Count Value read-only Register (LPTIM_CNTVAL)

Register	Address offset	Access	Reset value	Description
LPTIM_CNTVAL	0x00	R	0x0000_0000	LPTIM Count value read-only register

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

LPTIM Count Value read-only Register (LPTIM_CNTVAL)

Bit	Access	Description
[31:16]	-	Reserved
[15:0]	R	LPT_CNT[15:0] : :Count value read-only register

14.4.2 LPTIM Control Register (LPTIM_CR)

Register	Address offset	Access	Reset value	Description
LPTIM_CR	0x04	RW	0x0000_0000	LPTIM Control Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							WT_FLAG
15	14	13	12	11	10	9	8
Reserved						TCK_EN	INT_EN
7	6	5	4	3	2	1	0
GATE_POLE	GATE	TCK_SEL[1:0]		TOG_EN	CT_SEL	MODE	TIMER_RUN

LPTIM Count Value read-only Register (LPTIM_CNTVAL)

Bit	Access	Description
[31:17]	-	Reserved
[16]	R	WT_FLAG : :WT Write synchronization flag 0: Synchronization complete, now can change LOAD/BGLOAD 1: synchronizing, write LOAD/BGLOAD invalid
[15:10]	-	Reserved
[9]	RW	TCK_EN : :LPTIM Count clock enable 0:LPTIM Count clock off 1:LPTIM Count clock enable The configuration of LOAD/BGLOAD can only be performed after the count clock is enabled
[8]	RW	INT_EN : :Interrupt enable control, enable interrupt after writing 1
[7]	RW	GATE_P : :Input Valid polarity of GATE Default high level GATE is valid, after setting 1 low level is valid

[6]	RW	GATE_EN :Timer Gating 0: no gate 1: Gated
[5:4]	RW	TCK_SEL[1:0] : :LPTIM Clock Select 00:PCLK; 10:LSE; 11:LSI
[3]	RW	TOG_EN : :TOG Output Enable 0:TOG, TOGN simultaneously output 0 1:TOG, TOGN Output signals with opposite phases. available for BUZZER.
[2]	RW	CT_SEL : :Counter/Timer function selection 0: Timer function, the timer uses the clock selected byTCK_SEL to count. 1: Counter func- tion, the counter uses the falling edge of the external input to count. sampling clock uses the clock selected by TCK_SEL, and the external input clock is lower than 1/2 sampling clock.
[1]	RW	MODE : :Timer working mode 0: Mode 1 No reload mode16Bit counter/Timer 1: Mode 2 auto-reload 16 Bit counter/Timer
[0]	RW	TIM_RUN : :Timer run control bit 0: Timer stop 1: Timer running

14.4.3 LPTIM Immediate reload register(LPTIM_LOAD)

Register	Address offset	Access	Reset value	Description
LPTIM_LOAD	0x08	RW	0x0000_0000	LPTIM Immediate reload register

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

LPTIM Immediate reload register(LPTIM_LOAD)

Bit	Access	Description
[31:16]	-	Reserved
[15:0]	RW	LOAD : immediate reload register Independent of whether Timer is running or not, regardless of MODE. Needs to read LPTIM_CR.WT_FLAG before writing LOAD, if and only when WT_FLAG is 0, data can be written . Write LOAD register completed WT_FLAG will go low. Writing to this register immediately updates the counter value. Reading this register returns the value last updated to LOAD or BGLOAD

14.4.4 LPTIM Interrupt Register(LPTIM_INTSR)

Register	Address offset	Access	Reset value	Description
LPTIM_INTSR	0x0C	R	0x0000_0000	LPTIM Interrupt Register

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

LPTIM Interrupt Register(LPTIM_INTSR)

Bit	Access	Description
[31:1]	-	Reserved
[0]	R	INTF: Interrupt flags 0: Interrupt did not occur 1: Overflow interrupt occurred

14.4.5 LPTIM Interrupt Clear Register(LPTIM_INTCLR)

Register	Address offset	Access	Reset value	Description
LPTIM_INTCLR	0x10	W	0x0000_0000	LPTIM Interrupt Clear Register

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

LPTIM Interrupt Clear Register(LPTIM_INTCLR)

Bit	Access	Description
[31:1]	-	Reserved
[0]	W	ICLR: write 1 clear interrupt flag, write 0 have no effect

14.4.6 LPTIM Cycle reload register(LPTIM_BGLOAD)

Register	Address offset	Access	Reset value	Description
LPTIM_BGLOAD	0x14	RW	0x0000_0000	LPTIM Cycle reload register

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

LPTIM Cycle reload register(LPTIM_BGLOAD)

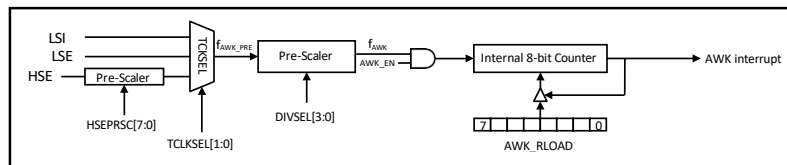
Bit	Access	Description
[31:16]	-	Reserved
[15:0]	RW	<p>BGLOAD[15:0]: BackGround Cycle reload Register</p> <p>needs to read LPTIM_CR.WT_FLAG before writing BGLOAD, if and only when WT_FLAG is 0, data can be written . Write BGLOAD register is completed WT_FLAG will go low.</p> <p>When this register is written, the set value is updated to the counter when the counter overflows.</p> <p>Reading this register returns the value last updated to LOAD or BGLOAD.</p>

15 Auto-wake-up Timer

MG32L003 has a special auto-wake-up timer (AWK), which provides a wake-up event benchmark for the chip in low power consumption mode. AWK keeps counting in Sleep or Deep Sleep mode. When AWK is used as a wake-up timer, AWK should be turned on before entering the power-saving mode. AWK can configure internal low-speed clock source LSI, external low-speed clock source LSE, and external high-speed clock HSE after frequency division. Note that the system clock frequency must be more than twice that of AWK clock. If AWK starts counting, the selected clock source will also keep working when the device enters Sleep or Deep Sleep mode. Note that the selected AWK clock source will not be automatically enabled together with the AWK configuration. The user should manually enable the selected clock source and wait for it to stabilize to ensure the success of the operation.

AWK is equipped with a simple 8-bit automatic reload count-up timer. Its prescaler can be selected from 1/2 to 1/65536 through AWK_CR.DIVSEL[3:0]. The user fills in the reload value to AWK_RLOAD register to determine its overflow rate. After AWK_CR.AWKEN is set, when CPU enters Sleep/Deep Sleep mode, load AWK_RLOAD register value to the internal 8-bit counter and start counting. When the counter overflows, AWK_SR.AWUF is set to 1 to wake up CPU.

Fig 15.0-1 AWK block diagram



15.1 AWK Registers

Tab 15.1-1 AWK register mapping

offset	Register	Reset value	Description
AWK base address : 0x 4000 1C00			
0x000	AWK_CR	0x0000_BC00	AWK timer control register
0x004	AWK_RLOAD	0x0000_0000	AWK timer reload data register
0x008	AWK_SR	0x0000_0000	AWK timer state register
0x00C	AWK_INTCLR	0x0000_0000	AWK timer Interrupt Clear register

15.2 Register Description

15.2.1 AWK timer control register(AWK_CR)

Register	Address offset	Access	Reset value	Description
AWK_CR	0x00	RW	0x0000_BC00	AWK timer control register

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

LPTIM Cycle reload register(LPTIM_BGLOAD)

Bit	Access	Description
[31:16]	-	Reserved
[15:8]	RW	HSEPRSC[7:0]: HSE Clock division factor $f_{HSE}/(N+1)$
[7]	-	Reserved
[6:5]	RW	TCLKSEL[1:0]: AWK Count clock source selection 00: Stop 01: LSI Clock 10: HSE divided Clock 11: LSE Clock
[4]	RW	AWKEN: AWK enable 1: Enable 0: Disable
[3:0]	RW	DIVSEL[3:0]: Counter clock source selection bit 0000: $F_{AWK_PRE}/2^1$ 0001: $F_{AWK_PRE}/2^2$ 0010: $F_{AWK_PRE}/2^3$... 1111: $F_{AWK_PRE}/2^{16}$

15.2.2 AWK timer reload data register(AWK_RLOAD)

Register	Address offset	Access	Reset value	Description
AWK_RLOAD	0x04	RW	0x0000_0000	AWK timer reload data register

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

AWK timer reload data register(AWK_RLOAD)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	RLDVAL[7:0]: Counter

15.2.3 AWK timer state register(AWK_SR)

Register	Address offset	Access	Reset value	Description
AWK_SR	0x08	R	0x0000_0000	AWK timer state register

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

AWK timer state register(AWK_SR)

Bit	Access	Description
[31:1]	-	Reserved
[0]	R	AWUF: Auto-wake-up occurs, hardware sets 1, software clears 0: no auto-wake-up occurs 1: Counter overflow, auto-wake-up occurs

15.2.4 AWK timer Interrupt Clear register(AWK_INTCLR)

Register	Address offset	Access	Reset value	Description
AWK_INTCLR	0x0C	W	0x0000_0000	AWK timer Interrupt Clear register

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

AWK timer Interrupt Clear register(AWK_INTCLR)

Bit	Access	Description
[31:1]	-	Reserved
[0]	W	INTCLR: Auto-wake-up interrupt clear 0: no effect 1: Clear auto-wake-up interrupt

16 BEEP

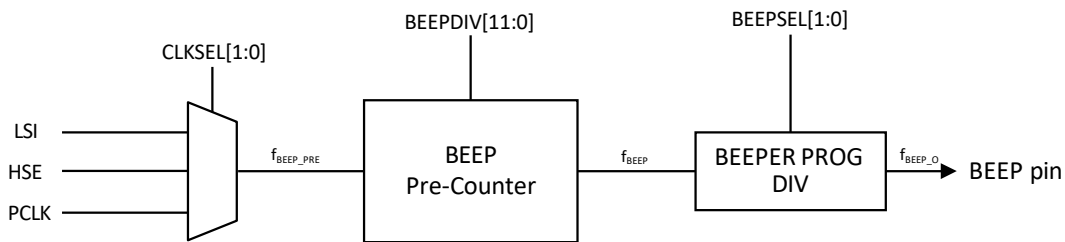
16.1 BEEP introduction

Select the LSI clock, HSE clock or PCLK clock to generate a variety of frequency buzzing signals through the frequency division setting.

BEEP_CSR.CLKSEL[1:0] bit to select the fBEEP_PRE clock, the buzzer clock fBEEP_PRE is divided by setting BEEPDIV[11:0] to get fBEEP.

By setting BEEPSEL[1:0], fBEEP_O buzzing signal is obtained.

Fig 16.1-1 Beep Function Diagram



16.2 BEEP Functional description

16.2.1 Buzzer operation

In order to use the beep function, perform the following steps in order:

1. Determine the value of BEEPDIV[11:0] according to the required buzzer frequency output value;
2. Select the output frequency of fBEEP_O by writing BEEPSEL[1:0] bits of BEEP_CSR;
3. Set BEEPEN bit of fBEEP_CSR to enable the clock source;
 - The prescaler calculator starts running only when the value of BEEPDIV[11:0] is different from the reset value 0x0FFF.
 - The value of BEEPDIV[11:0] should be kept unchanged during operation of the buzzer.

16.2.2 Buzzer Calibration

This step can be used to calibrate the LSI clock to achieve a more standard fBEEP_O(1KHz,2KHz or 4KHz) frequency output.

1. TIM1,TIM2 or CLKTRIM module to measure the clock frequency of the internal low-speed clock(LSI)
2. Use the following method to calculate the value of BEEPDIV, where A and x are the integer and fractional values of fBEEP_PRE/fBEEP: when x is less than or equal to A/(1+2*A),BEEPDIV = A-1; otherwise BEEPDIV = A
3. Write BEEPDIV value to BEEPDIV[11:0] bit of BEEP_CSR

16.3 BEEP Registers

Tab 16.3-1 BEEP register mapping

offset	Register	Reset value	Description
BEEP base address : 0x 4000 4800			
0x000	BEEP_CSR	0x0000_0FFF	BEEP control register

16.4 Registers description

16.4.1 BEEP control register(BEEP_CSR)

Register	Address offset	Access	Reset value	Description
BEEP_CSR	0x00	RW	0x0000_0FFF	BEEP control register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		CLKSEL		Reserved	BEEPEN	BEEPSEL	
15	14	13	12	11	10	9	8
Reserved				BEEPDIV			
7	6	5	4	3	2	1	0
BEEPDIV							

BEEP control register(BEEP_CSR)

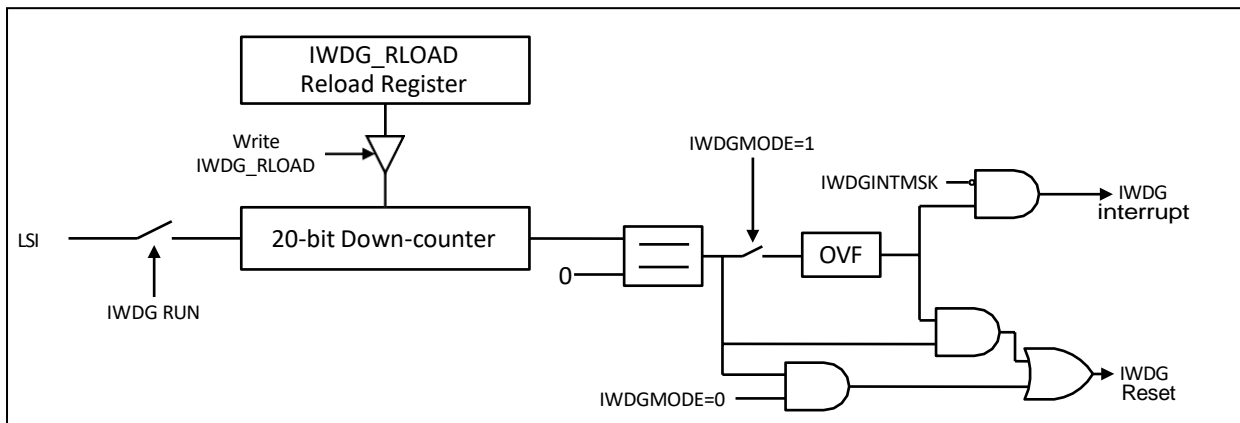
Bit	Access	Description
[31:22]	-	Reserved
[21:20]	RW	AWUF: Clock selection. 00: Stop 01: LSI 10: HSE 11: PCLK
[19]	-	Reserved
[18]	RW	BEEPEN: enable Beep
[17:16]	RW	BEEPSEL: Buzzer output BEEP_OF requency selection bit 00: fBEEP_PRE/8 01: fBEEP_PRE/4 1x:fBEEP_PRE/2
[15:12]	-	Reserved
[11:0]	RW	BEEPDIV: BEEP prescaler divides fBEEP_PRE to get 8KHz period signal frequency division factor is BEEPDIV + 1 fBEEP= fBEEP_PRE/(BEEPDIV + 1)

17 Independent watchdog (IWDG)

17.1 IWDG introduction

The function of IWDG is to prevent the software system from working abnormally or crashing due to program execution errors under abnormal conditions. And IWDG reset can help the system recover automatically. The working principle is that when the software system makes an error, a reset or interrupt is generated at a fixed time (this time can be configured) to allow the program to be re-executed or executed according to the interrupt service program without system breakdown. This thereby increases the security performance of the software system.

Fig 17.1-1 IWDG diagram



17.2 IWDG functional description

- IWDG module is a 20-bit counter, each LSI clock (assuming 38.4KHz) counts and accumulates once, and the counting time can be configured as 26us-27s;
- Internal low-speed clock (LSI) count;
- When the count overflows, it supports interrupt and reset 2 methods;
- Configurable count overflow time;
- Start and clear IWDG has operation sequence requirements to increase safety performance;
- Some registers are write-protected to prevent accidental operation of the program.

17.2.1 Timeout period

IWDG_RLOAD[19:0] register value	Timeout period
0x00000	26us
...	...
0x003FF	26.6ms
...	...
0xFFFFF	27s

17.2.2 IWDG Interrupt after overflow

In this mode, IWDG will periodically generate an interrupt according to the set time. The IWDG overflow flag needs to be cleared in the interrupt service program.

The configuration method is as follows:

1. Write 0x55AA6699 to IWDG_UNLOCK to unlock IWDG register write protection. If IWDG_UNLOCK.IWDGREN is 1, this step can be omitted.
2. Configure IWDG_CFGR.IWDGMODE to 1 to select the interrupt mode.
3. Select configuration IWDG_CFGR.IWDGINTMSK as 0, then CPU responds to the interrupt signal of IWDG; configured as 1, the interrupt is masked, and CPU can only read the IWDG_SR.IWDGOVF to determine whether the count overflows.
4. Configure IWDG_RLOAD register. Select IWDG count overflow time.
5. Write any value other than 0x55AA6699 to IWDG_UNLOCK to enable register write protection of IWDG.
6. Write 0x55 to IWDG_CMDCR, start IWDG.
7. If an interrupt occurs, first remove the IWDG register protection in the interrupt service program, and then IWDG_INTCLR writing 1 clears the interrupt flag.

17.2.3 IWDG reset after overflow

In this mode, the reset signal will be generated after the IWDG counter overflows, and the signal will reset the MCU. Users need to reload the IWDG counter before the IWDG overflow to avoid IWDG reset.

The configuration method is as follows:

1. Write 0x55AA6699 to IWDG_UNLOCK to unlock IWDG register write protection. If WDG_UNLOCK.IWDGREN is 1, this step can be omitted.
2. Configure IWDG_CFGR.IWDGMODE to 0 to select the reset mode.
3. Configure IWDG_RLOAD register. Select IWDG count overflow time.
4. Write any value other than 0x55AA6699 to IWDG_UNLOCK to enable register write protection of IWDG.
5. Write 0x55 to IWDG_CMDCR, start IWDG.
6. Write 0xAA to IWDG_CMDCR before count overflow, refresh IWDG counter.

17.3 IWDG registers

Tab 17.3-1 IWDG register mapping

offset	Register	Reset value	Description
IWDG base address : 0x 4000 2400			
0x00	IWDG_CMDCR	0x0000 0000	control register
0x04	IWDG_CFGR	0x0000 0000	IWDG Configuration register
0x08	IWDG_RLOAD	0x000F FFFF	IWDG counter reload register
0x0c	IWDG_CNTVAL	0x000F FFFF	IWDG counter value
0x10	IWDG_SR	0x0000 0000	IWDG status register
0x14	IWDG_INTCLR	0x0000 0000	IWDG Interrupt Clear register

0x18	IWDG_UNLOCK	0x0000 0000	IWDG unlock
------	-------------	-------------	-------------

17.4 Registers description

17.4.1 IWDG Control Command Register (IWDG_CMDCR)

Register	Address offset	Access	Reset value	Description
IWDG_CMDCR	0x00	W	0x0000_0000	IWDG Control Command Register

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

IWDG Control Command Register (IWDG_CMDCR)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	CMD[7:0]: 0x55:IWDG start command 0xAA:IWDG Reload refresh command

Note: Reload commands can only be written when IWDG is running

17.4.2 IWDG Configuration Register (IWDG_CFGR)

Register	Address offset	Access	Reset value	Description
IWDG_CFGR	0x04	RW	0x0000_0000	IWDG Configuration Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					IWDGRUNF	IWDGINTMS	IWDGMODE

IWDG Configuration Register (IWDG_CFGR)

Bit	Access	Description
[31:3]	-	Reserved
[2]	R	IWDGRUNF: IWDG Running flag 0:IWDG Stop 1:IWDG running(is counting)

[1]	RW	IWDGINTMSK: IWDG Interrupt Mask 0: interrupt not masked(notified to CPU) 1: Interrupts are masked
[0]	RW	IWDGMODE: IWDG Count overflow mode selection bit 0: Reset method 1: Interrupt mode, generate an interrupt signal, then restart the counter, if the interrupt is not cleared before the second timeout occurs, then generate a system reset signal. Note: IWDG generates a reset which resets the entire system

Note: protected by IWDG_UNLOCK

17.4.3 IWDG Counter Reload Register(IWDG_RLOAD)

Register	Address offset	Access	Reset value	Description
IWDG_RLOAD	0x08	RW	0x000F_FFFF	IWDG Counter Reload Register

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

IWDG Counter Reload Register(IWDG_RLOAD)

Bit	Access	Description
[31:20]	-	Reserved
[19:0]	RW	IWDGRUNF: IWDG Reload Register

Note: protected by IWDG_UNLOCK

17.4.4 IWDG Counter Value Register(IWDG_CNTVAL)

Register	Address offset	Access	Reset value	Description
IWDG_CNTVAL	0x0C	R	0x000F_FFFF	IWDG Counter Value Register

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

IWDG Counter Value Register(IWDG_CNTVAL)

Bit	Access	Description
[31:20]	-	Reserved
[19:0]	R	IWDGCNT: IWDG Counter Value Register

17.4.5 IWDG Interrupt Status Register(IWDG_SR)

Register	Address offset	Access	Reset value	Description
IWDG_SR	0x10	R	0x0000_0000	IWDG Interrupt Status Register

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

IWDG Interrupt Status Register(IWDG_SR)

Bit	Access	Description
[31:1]	-	Reserved
[0]	R	<p>IWDGOVF: IWDG Overflow interrupt flag</p> <p>0:IWDG No overflow interrupt occurs</p> <p>1:IWDG overflow interrupt occurs</p> <p>Note:</p> <ol style="list-style-type: none"> When IWDG is configured as reset mode, regardless of IWDG counter overflow, None of this will be set high. When IWDG is configured as interrupt mode, no matter whether IWDGINTMSK is set high or not, as long as IWDG counter overflows, this bit will be set high.

17.4.6 IWDG Interrupt Clear register(IWDG_INTCLR)

Register	Address offset	Access	Reset value	Description
IWDG_INTCLR	0x14	W	0x0000_0000	IWDG Interrupt Clear register

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

IWDG Interrupt Clear register(IWDG_INTCLR)

Bit	Access	Description
-----	--------	-------------

[31:1]	-	Reserved
[0]	W	IWDGINTCLR: IWDG Interrupt clearing 0:Write 0, no action 1:Write 1 to clear the IWDG interrupt flag

Note: Protected by IWDG_UNLOCK

17.4.7 IWDG protection register(IWDG_UNLOCK)

Register	Address offset	Access	Reset value	Description
IWDG_UNLOCK	0x18	RW	0x0000_0000	IWDG unlock

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

IWDG protection register(IWDG_UNLOCK)

Bit	Access	Description
[31:1]	-	Reserved
[0]	RW	IWDGINTCLR: IWDG Interrupt clearing 0:The associated registers of the IWDG cannot be changed 1:The associated registers of the IWDG can be changed

Note: Write 0x55AA6699 to IWDG_UNLOCK to unlock the IWDG register write protection, Write any value other than 0x55AA6699 to IWDG_UNLOCK to enable IWDG register write protection.(IWDG_CFGR、IWDG_RLOAD、IWDG_INTCLR)

- The feed dog command to the watchdog timer being updated requires a delay of two watchdog count clock source clocks
- When the system operates the watchdog, at least three watchdog clocks should be allowed between feeding sessions

18 Window watchdog (WWDG)

18.1 WWDG introduction

The purpose of the Window Watchdog Timer(WWDG) is to perform system reset within a specified window cycle, preventing the software from entering an uncontrollable state under any unpredictable condition.

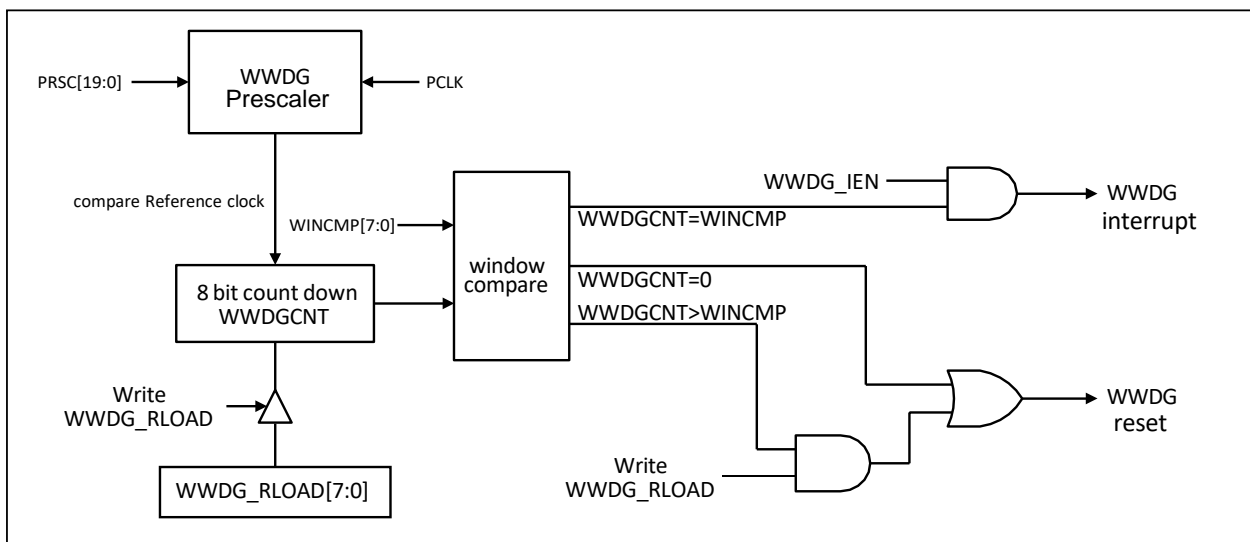
18.2 WWDG main features

- An 8-bit down counter(WWDG_CNT) and an 8-bit comparison value(WINCMP) make WWDG time-out window cycle adjustable
- Support 20-bit value(PRSC) to select watchdog prescale value
- Support window count value comparison interrupt, count overflow, and reset when loading count value error

18.3 WWDG BLOCK Diagram

The block diagram of the window watchdog timer is as follows:

Fig 18.3-1 WWDG BLOCK Diagram



18.4 WWDG Basic configuration

WWDG peripheral clock source is enabled by `RCC_PCLKEN.WWDGCKEN`.

1. Configure window watchdog count initial value via `WWDG_RLOAD[19:0]`
2. Configure count clock prescaler via `WWDG_CR.PRSC[19:0]`
3. Configure window comparison value via `WWDG_CR.WINCMP[7:0]`
4. Configure `WWDG_INTEN.WWDGIEN` according to whether enabling interrupt is required
5. Write 1 via `WWDG_CR.WWDGEN` to enable window watchdog

18.5 WWDG functional description

Window watchdog timer(WWDG)is an 8-bit down counter with a selectable prescaler value. Different prescaler values correspond to different watchdog timing overflow time. The clock source of the 8-bit window watchdog timer is the clock after the frequency division of the PCLK clock. The watchdog clock source has an optional 20-bit prescaler value, the value can be set and selected through WWDG_CR.PRSC[19:0]bit, the corresponding prescaler value is as follows.

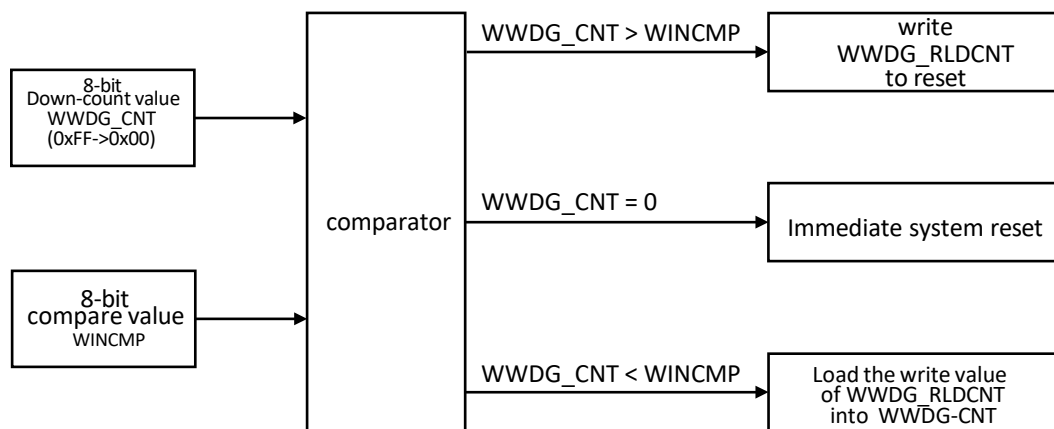
Tab 18.5-1 WWDG Pre-division frequency value selection

PRSC[19:0]	Prescaler value	Timed overflow period	Timed overflow interval PCLK = 24MHz
0x00000	1	$T_{PCLK} * 1$	41.7ns
0x00001	2	$T_{PCLK} * 2$	83.4ns
0x00002	3	$T_{PCLK} * 3$	125.1ns
0x00003	4	$T_{PCLK} * 4$	166.8ns
0x00004	5	$T_{PCLK} * 5$	208.5ns
..
0x80000	524289	$T_{PCLK} * 524289$	21.9ms
..
0xFFFFF	1048576	$T_{PCLK} * 1048576$	43.8ms

18.5.1 WWDG counts

When WWDG_CR.WWDGEN bit is enabled, the window watchdog down counter will count down from WWDG_CNT[7:0] to 0, and cannot be turned off by software. In order to prevent the program from closing the window watchdog timer at a position not specified by the user, WWDGEN of the window watchdog timer control register can only be written once after the chip is powered on or reset. When the WWDG_CR.WWDGEN bit is enabled by software, the user cannot disable the window watchdog timer WWDG_CR.WWDGEN, modify the counter prescaler period WWDG_CR.PRSC[19 :0], or modify the window comparison value WWDG_CR.WINCMP[7:0], unless the chip is reset. The window watchdog timer will stop counting when CPU enters the Sleep mode or the Deep Sleep mode, and the CPU will resume normal operation after being awakened.

Fig 18.5-1 WWDG Reset and reload



18.5.2 WWDG comparison interrupt

During the down-counting process of the window watchdog timer, when the count value of the window watchdog timer WWDG_CNT[7:0] is equal to the window comparison value WWDG_CR.WINCMP[7:0], WWDG_SR.WWDGIF will be set to 1 and WWDG_SR.WWDGIF can be cleared by software. If WWDG_INTEN.WWDGIEN bit is enabled, when WWDG_SR.WWDGIF bit is set 1 by hardware, the window watchdog comparison match interrupt will be generated.

18.5.3 WWDG reset system

When the value of WWDG counter counts to 0, RCC_RSTSR.WWDGRST will be set to 1. Before the WWDG counter is counted down to 0, the user must reload by writing a value to WWDG_RLOAD, thus preventing WWDG reset from occurring. reloading can only be performed when the value of the counter is less than or equal to WINCMP value. If the current value of the WWDG counter is greater than the value of WINCMP, the user writes to the WWDG_RLOAD register, the window watchdog timer reset system signal will be generated immediately, and cause the chip reset

18.5.4 Window setting limit of WWDG

When the user writes the value of reloaded WWDG to the WWDG_RLOAD register,
 $TPCLK = THCLK * (2 * RCC_PCLKDIV.APBCKDIV[7:0])$

Set time interval:

$$T = TPCLK * (WWDG_CR.WWDG_PRSC[19:0] + 1) * (WWDG_RLOAD.WWDG_RLOAD[7:0] + 1)$$

Users can configure the frequency division register WWDG_PRSC[19:0] and WWDG_RLOAD[7:0] according to the needs to achieve the desired time interval. To ensure normal operation, the value of WWDG_RLOAD[7:0] must be greater than or equal to 1.

18.6 WWDG compare with IWDG

18.6.1 Reset Conditions and Reset Delays

IWDG and WWDG are usually used to reset the system after the system has run into an uncontrollable state. IWDG has only one condition to trigger the reset signal, and WWDG has two conditions to trigger WWDG to generate the reset signal:

WWDGCNT = 0;

Write to WWDG_RLOAD when WWDGCNT is greater than WINCMP.

Once WWDGRST is set to 1, WWDG will reset the system immediately.

18.6.2 Wake-up function

IWDG supports wake-up function and continues to work in Deep Sleep mode. In contrast, WWDG does not support wake-up function and the counter of WWDG will stop counting in Deep Sleep mode.

18.7 WWDG Registers

Tab 18.7-1 WWDG Registers

Address offset	Identifier	Reset value	Description
WWDG :0x4000 2000			
0x00	WWDG_RLOAD	0x0000_00FF	WWDG Reload Count Register
0x04	WWDG_CR	0x0800_00FF	WWDG Control Register
0x08	WWDG_INTEN	0x0000_0000	WWDG Interrupt Enable Register
0x0C	WWDG_SR	0x0000_0000	WWDG Status Register
0x10	WWDG_INTCLR	0x0000_0000	WWDG Interrupt Clear Register
0x14	WWDG_CNTVAL	0x0000_00FF	WWDG Counter Value Register

18.8 Register description

18.8.1 WWDG Reload Count Register(WWDG_RLOAD)

Register	Address offset	Access	Reset value	Description
WWDG_RLOAD	0x00	W	0x0000_00FF	WWDG Reload Count Register

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

WWDG Reload Count Register (WWDG_RLOAD)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	W	WWDG_RLOAD[7:0]: WWDG Reload Count Register, write value greater than 0

18.8.2 WWDG Control Register (WWDG_CR)

Register	Address offset	Access	Reset value	Description
WWDG_CR	0x04	RW	0x0800_00FF	WWDG Control Register

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

WWDG Control Register (WWDG_CR)

Bit	Access	Description
[31:29]	-	Reserved
[28]	RW	WWDGEN: Window watchdog enable bit Setting this bit enables the window watchdog timer 0: Disable window watchdog timer function 1: Enable window watchdog timer function
[27:8]	RW	PRSC[19:0]: WWDG Prescaler : $F_{PCLK}/(PRSC+1)$
[7:0]	RW	WINCMP[7:0]: WWDG Window Compare Register Setting this register adjusts the effective reload window. Note : Software can write only if WWDG counter value is between 0 and WINCMP WWDG_LOAD. When WWDG counter value is greater than WINCMP, if software writes WWDG_LOAD, WWDG will generate a reset signal

Note : When WWDGEN is set to 1, the software configuration

18.8.3 WWDG Interrupt Enable Register(WWDG_INTEN)

Register	Address offset	Access	Reset value	Description
WWDG_INTEN	0x08	RW	0x0000_0000	WWDG Interrupt Enable Register

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

WWDG Interrupt Enable Register (WWDG_INTEN)

Bit	Access	Description
[31:1]	-	Reserved
[0]	RW	WWDGIEN: WWDG Interrupt Enable Bit Setting this bit enables the windowed watchdog timer interrupt function. 0: Disable window watchdog timer interrupt function 1: Enable window watchdog timer interrupt function

18.8.4 WWDG Status Register (WWDG_SR)

Register	Address offset	Access	Reset value	Description
WWDG_SR	0x08	R	0x0000_0000	WWDG Status Register

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

WWDG Status Register (WWDG_SR)

Bit	Access	Description
[31:1]	-	Reserved
[0]	R	<p>WWDGIF: WWDG Compare match interrupt flag</p> <p>0: No window watchdog timer interrupt</p> <p>1: Windowed watchdog timer interrupt</p> <p>When WINCMP and WWDG counters match, the bit is 1, software is on WWDG_INTCLR.INTCLR Write 1 Clear 0 this bit.</p>

18.8.5 WWDG Interrupt Clear Register(WWDG_INTCLR)

Register	Address offset	Access	Reset value	Description
WWDG_INTCLR	0x10	W	0x0000_0000	WWDG Interrupt Clear Register

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

WWDG Interrupt Clear Register(WWDG_INTCLR)

Bit	Access	Description
[31:1]	-	Reserved
[0]	W	<p>WWDGIF: WWDG Compare match interrupt flag clear</p> <p>software write 1 clear 0 corresponding interrupt flag WWDG_SR.WWDGIF.</p>

18.8.6 WWDG Counter Value Register(WWDG_CNTVAL)

Register	Address offset	Access	Reset value	Description
WWDG_CNTVAL	0x14	R	0x0000_00FF	WWDG Counter Value Register

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

WWDG Counter Value Register(WWDG_CNTVAL)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	R	WWDGCNT[7:0] : WWDG Counter value This register represents the current value of the window watchdog counter, this register is read-only

19 UART1/UART2

UART1/UART2: Universal asynchronous receiver / transmitter 1/2

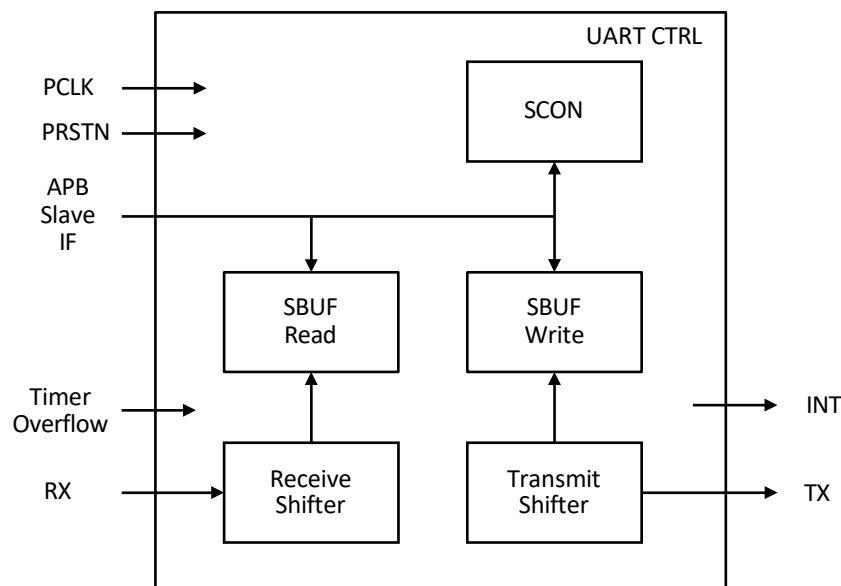
19.1 UART introduction

This product has 2 general-purpose UART modules(UART1/2), supporting half-duplex and full-duplex transmission; support 8 bit and 9 bit data formats; support four different transmission modes of Mode0/1/2/3;the baud rate of UART1 can be generated by TIM10 or automatic baud rate generator, and the baud rate of UART2 can be generated by TIM11 or automatic baud rate generator; support multi-machine communication mode; support automatic address recognition; support the given address and broadcast address.

Universal UART(UART1/2) has only one clock input from PCLK,and the register configuration logic and data transmission logic work in this clock domain.

19.2 UART Block Diagram

Fig 19.2-1 UART Block Diagram



19.3 UART Operating Modes

19.3.1 Mode 0(synchronous mode, half duplex)

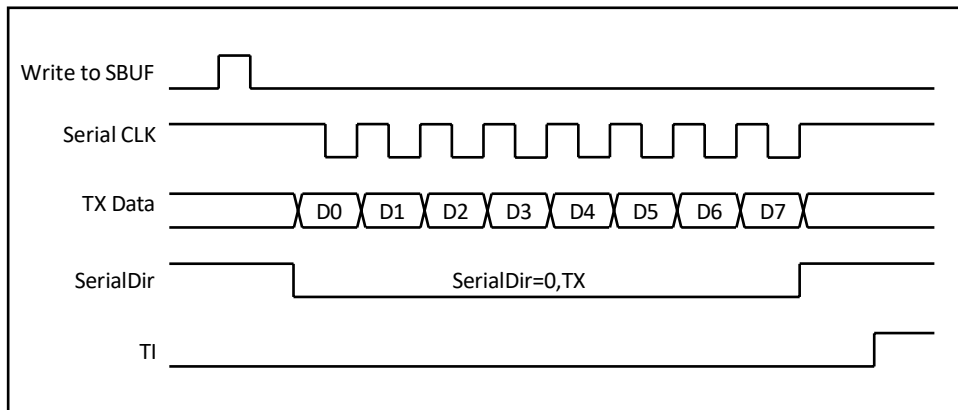
When working in Mode 0, UART works in synchronous mode, and its baud rate is 1/12 of the fixed PCLK clock. UART receiving data is input by RXD, UART sending data is output by TXD, and TXD is the input and output port at this time. The UART synchronous shift clock is output by TXD, which is output port at this time. Note that this mode can only be used as the master to send the synchronous shift clock, and cannot be used as the slave to receive the shift clock from the outside. In this mode, the data bit width transmitted can only be 8 bits, without start bit and stop bit.

Clear UARTx_SCON.SM0 and UARTx_SCON.SM1 to zero to enter Mode0 operation mode.

Send data

When sending data, clear the UARTx_SCON.REN bit and write the data to the UARTx_SBUF register. At this point, sending data will output from RXD(low bit first, high bit last), synchronous shift clock is output from TXD.

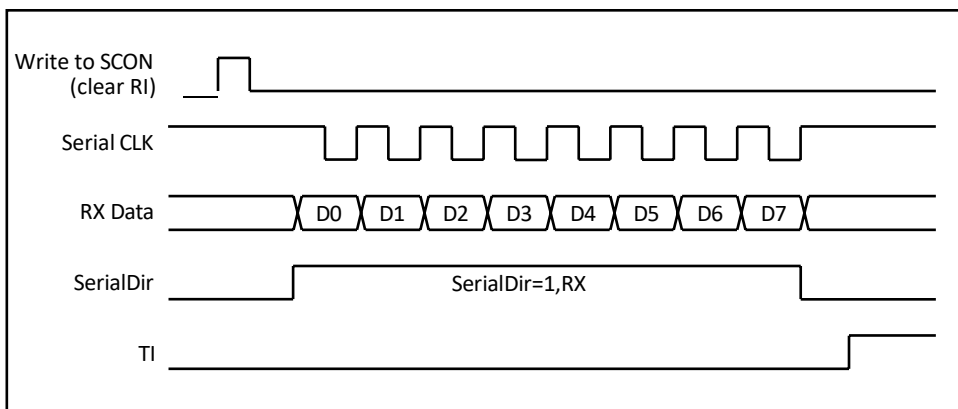
Fig 19.3-1 UART mode0 output



Receive data

When receiving data, set UARTx_SCON.REN bit to 1 and clear UARTx_INTSR.RI bit. When receiving is finished, the data can be read from UARTx_SBUF register. At this time, the received data is input from RXD(low bit first, high bit last), and the synchronous shift clock is output from TXD.

Fig 19.3-2 UART mode0 receive



19.3.2 Mode 1(asynchronous mode, full duplex)

When working in Mode1, the sending data is sent through TXD, and the receiving data is received through RXD. The data consists of 10 bits: Start bit "0", followed by 8 data bit(low bit first, high bit after), and finally the stop bit "1". In this mode, the baud rate can be generated by the programmable timer module or by the automatic baud rate generator inside the module. When UARTx_BAUDCR.SELF_BRG is 0, when the baud rate is selected to be generated by the timer,the baud rate of UART1 is generated

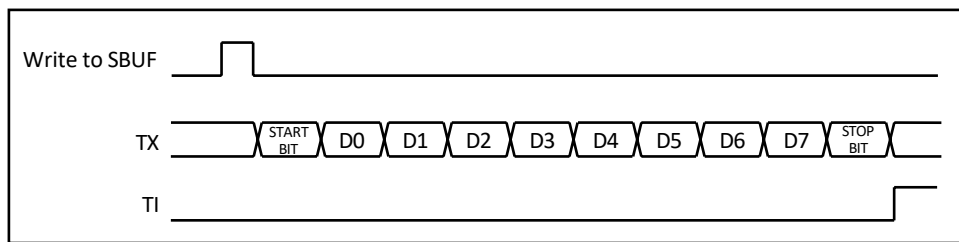
by TIM10, the baud rate of UART2 is generated by TIM11;the baud rates of UART1 and UART2 are generated by their own internal automatic baud rate generator. For baud rate generation formula, refer to Mode1/3 in Section 19.3.5.

Set UARTx_SCON.SM0 to 0 and UARTx_SCON.SM1 to 1 to enter Mode1 working mode.

Send data

When sending data, regardless of the value of UARTx_SCON.REN, write the sent data to UARTx_SBUF register, the data will be moved out of the TXD (low bit first, high bit last)

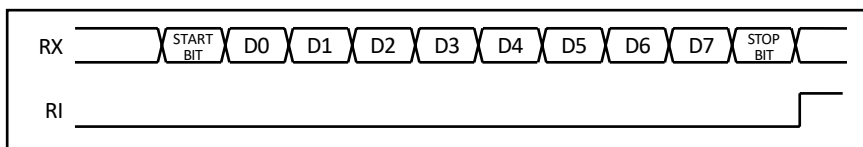
Fig 19.3-3 UART mode1 output



Receive data

When receiving data, set the UARTx_SCON.REN position to 1 and clear the UARTx_INTSR.RI bit to 0. Start to receive RXD data (low first, high last), after receiving, can be read from the UARTx_SBUF register.

Fig 19.3-4 UART mode1 receive



19.3.3 Mode 2(asynchronous mode, full duplex)

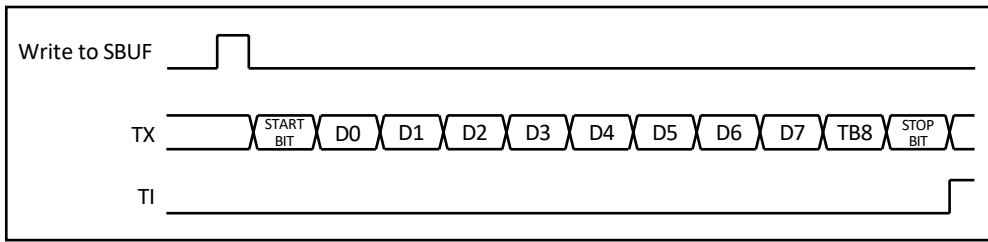
When working in Mode2, the sending data is sent through TXD, and the receiving data is received through RXD. The data consists of 11 bits:starting bit "0", followed by 8 data bits,one TB8 bit and stop bit. The additional TB8 bit is used in the multi-machine communication environment. When TB8 = 1, it indicates that the address frame is received; when TB8 = 0, it indicates that the data frame is received. When multi-machine communication is not required, this bit can also be used as a parity check bit. In this mode, the baud rate can be generated independently without an external timer module.

Set UARTx_SCON.SM0 to 1 and clear UARTx_SCON.SM1 bit to 0 to enter Mode 2 working mode.

Send data

When sending data, regardless of the value of UARTx_SCON.REN, write the sent data to UARTx_SBUF.In the register, the data will be moved out of TXD(low bit first, high bit last).

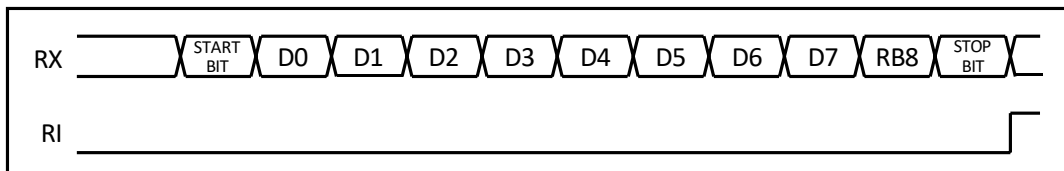
Fig 19.3-5 UART mode2 output



Receive data

When receiving data, set the UARTx_SCON.REN position to 1 and clear the UARTx_INTSR.RI bit to 0. Start to receive RXD data (low first, high last), after receiving, can be read from the UARTx_SBUF register.

Fig 19.3-6 UART mode2 receive



19.3.4 Mode 3(asynchronous mode, full duplex)

The data format, transmission timing and operation mode of Mode3 are the same as Mode2. The only difference is that the selection of Mode3 baud rate is generated by the programmable timer or the internal automatic baud rate generator, instead of Mode2, which is only generated by the device itself. The baud rate of Mode3 is programmable and the baud rate is generated in the same way as Mode1.

Set UARTx_SCON.SM0 to 1,UARTx_SCON.SM1 to 1 to enter Mode3 working mode.

19.3.5 Baud rate programming

Mode 0

When working in Mode0, the baud rate is fixed at 1/12 of PCLK, and the support of Timer is not required.

Mode 1/3

When working in Mode1 or Mode3, the baud rate generation formula is shown in the figure below:

UARTx_BAUDCR.SELF_BRG=0, use Timer baud rate mode:

$$\text{BaudRate} = \frac{(\text{UARTx_SCON.DBAUD} + 1) * F_{\text{PCLK}}}{32 * (2^{16} - \text{TIMx_BGLOAD}[15 : 0])} \tag{19.1}$$

UARTx_BAUDCR.SELF_BRG=1, use its own baud rate generation mode:

$$\text{BaudRate} = \frac{(\text{UARTx_SCON.DBAUD} + 1) * F_{\text{PCLK}}}{32 * (\text{UARTx_BAUDCR.BRG}[15 : 0] + 1)} \tag{19.2}$$

Where, `UARTx_SCON.DBAUD` means double baud rate, `FPCLK` means PCLK clock frequency, `TIMx_BGLOAD` is the periodic load count value of Timer.

Note: that Timer must be configured as 16-bit automatic reload mode. In this mode, the immediate reload register (`TIMx_LOAD`) and the cycle reload register (`TIMx_BGLOAD`) should be written with the same initial value.

Mode 2

When operating in Mode 2, the baud rate is fixed at the value obtained by the following formula:

$$\text{BaudRate} = \frac{(\text{UARTx_SCON.DBAUD} + 1) * F_{\text{PCLK}}}{64} \quad (19.3)$$

Where, `UARTx_SCON.DBAUD` means double baud rate, and `FPCLK` means PCLK clock frequency.

19.3.6 Frame error detection

Mode 1/2/3 has a frame error detection function. The hardware will automatically detect whether the received frame data has a valid Stop bit. If no valid Stop bit is received, `UARTx_INTSR.FE` is set to 1. `UARTx_INTSR.FE` bit is set by hardware and cleared by software. If the software fails to clear 0 in time, and even if the subsequent received data has a valid Stop bit, it will not clear `UARTx_INTSR.FE` flag 0.

19.3.7 Multi-machine communication

Mode 2/3 has multi-machine communication function, so 1 bit `TB8/RB8` is added in its frame format. Set `UARTx_SCON.SM2` to "1" to enable the multi-machine communication bit. When the multi-machine communication bit is enabled, when sending data, the master can use `UARTx_SCON.TB8` to distinguish whether the current frame is an address frame (`UARTx_SCON.TB8=1`) or a data frame (`UARTx_SCON.TB8=0`). When receiving data, the slave ignores current Receive frame with `RB8` bit (9th bit) of "0". When the `RB8` bit (9th bit) of the received frame is "1", it indicates that it is an address frame, and the opportunity continues to judge whether the received address is equal to its own address. If it matches, the `UARTx_SCON.RB8` is set to "1" and `UARTx_INTSR.RI` is set to "1" to indicate that the frame is an address frame and the address has been matched. After seeing `UARTx_SCON.RB8=1` and `UARTx_INTSR.RI=1` from machine software, the slave software first clears `UARTx_SCON.SM2` to "0", then prepares to accept data frames given to it. If the addresses are not equal, it indicates that the master does not address the slave, and the slave hardware maintains `UARTx_SCON.RB8` and `UARTx_INTSR.RI` as "0", and the software keeps `UARTx_SCON.SM2` bit as "1", and the slave continues to be in the address monitoring state.

19.3.8 Automatic Address Recognition

When the multi-device communication bit is enabled (`UARTx_SCON.SM2` is set to "1"), the automatic address recognition function will also be enabled. This function is implemented by hardware, so that the slave can detect and receive each address frame. If the address matches the address of the slave, the receiver will give `UARTx_INTSR.RI` receive flag. If the addresses do not match, the receiving end will not give any acceptance flag.

If necessary, the multi-machine communication bit can also be turned on under Mode 1. At this time, `TB8` bit is replaced by Stop bit. When the slave receives the matched address frame and valid

Stop bit, UARTx_INTSR.RI will be set to “1”. In order to support automatic address recognition, the concepts of broadcast address and given address are defined.

19.3.9 Given address

The UARTx_SADDR register of the UART device is used to represent the given address of your device, and the UARTx_SADEN register is an address mask, which can be used to define the irrelevant bits in the address. When a certain bit of UARTx_SADEN is “0”, it means that the bit address is an unrelated bit, which means in the process of address matching, the address of this bit does not participate in address matching. These independent bits increase the flexibility of addressing, so that the master can address one or more slave devices at the same time.

Note: that the UARTx_SADEN register must be set to 0xFF if a unique matching address needs to be given.

$$\text{GivenAddr} = \text{UARTx_SADDR} \ \& \ \text{UARTx_SADEN} \quad (19.4)$$

Broadcast address

The broadcast address is used to address all slave devices at the same time, and the general broadcast address is 0xFF.

$$\text{BoardCastAddr} = \text{UARTx_SADDR} \ || \ \text{UARTx_SADEN} \quad (19.5)$$

Examples of given address and broadcast address

Assume that UARTx_SADDR and UARTx_SADEN of a slave device are configured as follows:

UARTx_SADDR : 0b01101001

UARTx_SADEN : 0b11111011

Then its given address and broadcast address are as follows:

Given : 0b01101x01

Broadcast : 0b11111x11

It can be seen that the master can address address the local slave with four addresses:

0b01101001 and 0b01101101 (given address)

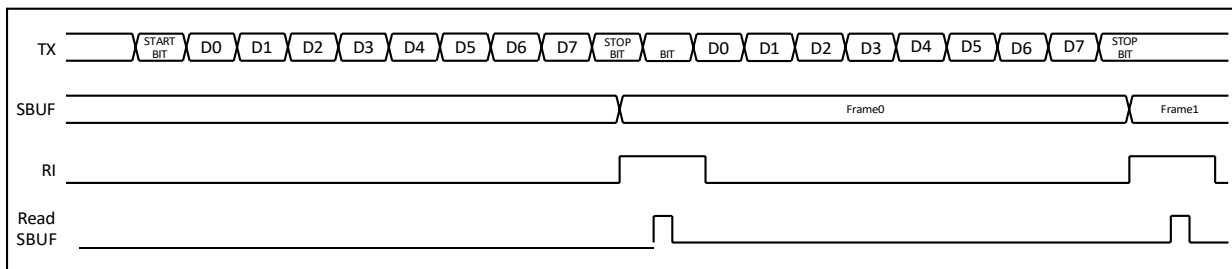
0b11111011 and 0b11111111 (broadcast address)

19.3.10 Transceiver cache

Receive cache

The universal UART(UART1/2) receiving end has a receiving cache with a frame length(8/9 bits). That is, after receiving a frame of data, the data in the receiving cache will be maintained until the Stop bit of the next frame of data is received, and the receiving cache will be updated to a new frame of data.

Fig 19.3-7 Receive cache



Send cache

The universal UART(UART1/2) sending end does not support transmit caching. If UARTx_SBUF register is filled in during data transmission, it will destroy the data currently being sent. Software should avoid this operation.

19.4 Infrared data association(IrDA)

The IrDA SIR physical layer specifies the use of an inverted return-to-zero modulation scheme(RZI), which uses an infrared light pulse to represent logic'0', see Figure 19.4-1 for the IrDA block diagram. The SIR transmission encoder modulates the NRZ (Non Return to Zero)bit stream output from the UART. The output pulse stream is transmitted to an external output driver and infrared LED. For SIRENDEC applications,UART only supports up to 115.2Kbps rate. In normal mode, the pulse width is specified as 3/16 of a bit cycle. The SIR receiver decoder demodulates return-to-zero bit stream from the infrared receiver and outputs the received NRZ serial bit stream to the UART. In the idle state, the decoder input is usually high(flag state). The polarity of the output of the transmitting encoder is opposite to the input of the decoder. A start bit is detected when the decoder input is low.

- IrDA is a half-duplex communication protocol. If the transmitter is busy(that is,UART is sending data to the IrDA encoder), any data on the IrDA receiving line will be ignored by the IrDA decoder. If the receiver is busy(that is, UART is receiving decoded data from the IrDA decoder), data from the TX of UART to IrDA will not be encoded by IrDA. When receiving data, sending should be avoided because the data to be sent may be corrupted.
- The SIR transmitting logic sends '0' as a high pulse and '1' as a low level; or reverse transmission. The width of the pulse is specified as 3/16 of the normal mode bit cycle, as shown in Figure19.4-2 IrDA transceiver pulses.
- The SIR decoder converts the received IrDA signal into a bit stream and sends it to the UART.
- The SIR receiving logic interprets the high level state as '0' and the low pulse as '1'; or reverse reception.
- The transmission encoder output has opposite polarity to the decoder input. When idle, the SIR output is in a low state.
- The IrDA specification requires the pulse width be greater than 1.41us. The pulse width is programmable. The spike pulse detection circuit at the receiver end filters pulses with a width less than 2 PSC cycles(PSC is a pre-frequency division value programmed in UARTx_IRDACR). Pulses with a width of less than 1 PSC cycle must be filtered out, but those with a width of more than 1 but less than 2 PSC cycles may be received or filtered out. Pulses with a width of more than 2

cycles will be considered as valid pulses.

- The IrDA receiver can communicate with another IrDA low-power transmitter.

19.4.1 IrDA low power consumption mode

IrDA can operate in normal mode or low power consumption mode. Selecting the low-power consumption mode requires setting the UART_IRDACR.IRLPMODE register to 1.

19.4.2 Transmitter

In low-power consumption mode, the pulse width no longer lasts for 3/16 bit cycles. Instead, the pulse width is 3 times the clock period of the low-power baud rate, which can have a minimum frequency of 1.42MHz. Typically, this value is 1.8432MHz (1.42MHz < PSC < 2.12MHz) A low-power mode programmable frequency divider divides the system clock to achieve this value.

19.4.3 Receiver

Reception in low power consumption mode is similar to reception in normal mode. To filter out spikes, UART should filter out pulses with a width shorter than 1 cycle. Only low-level signals of an IrDA low- power baud rate clock(PSC in UART_IRDACR) with a duration greater than 2 cycles are accepted as valid signals.

Note:

1. Pulses with a width less than 2 and greater than 1 PSC cycle may or may not be filtered out.
2. The setup time of the receiver should be managed by software. The IrDA physical layer specification specifies a minimum delay of 10ms between transmission and reception(IrDA is a half-duplex protocol).

Fig 19.4-1 IrDA Diagram

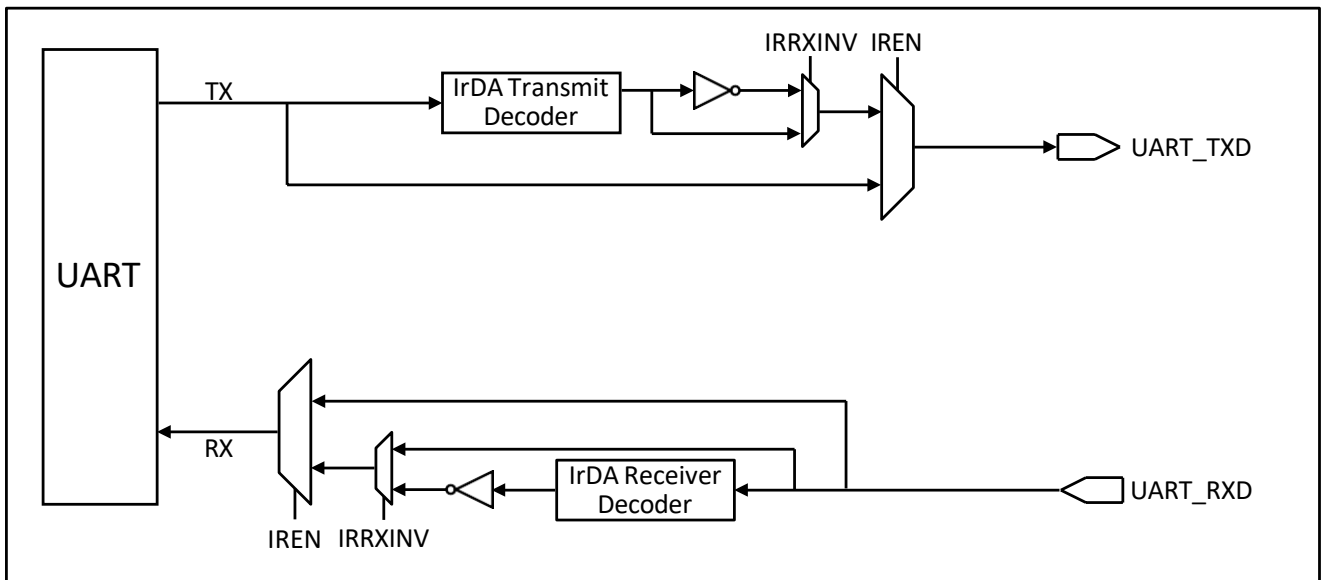
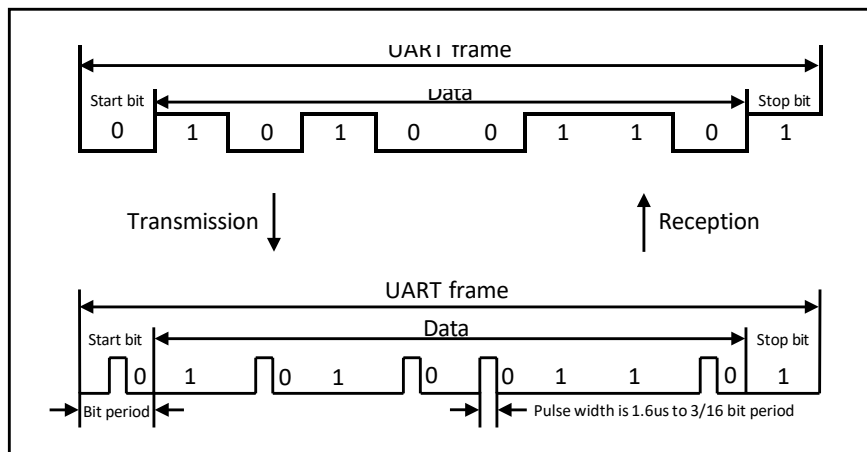


Fig 19.4-2 IrDA Send/Receive Pulse



19.5 Different baud rate frequency division Settings

PCLK = 1 MHz						
Baud Rate	Double baud rate			Single baud rate		
	CNT	Actual Baud Rate	Error Rang (%)	CNT	Actual Baud Rate	Error Rang (%)
2400	26	2403.85	0.16%	13	2403.85	0.16%
4800	13	4807.69	0.16%	7	4464.29	-6.99%
9600	7	8928.57	-6.99%	3	10416.67	8.51%
19200	3	20833.33	8.51%	2	15625.00	-18.62%
38400	2	31250.00	-18.62%	1	31250.00	-18.62%
57600	1	62500.00	8.51%	1	31250.00	-45.75%
76800	1	62500.00	-18.62%	0	-	-
115200	1	62500.00	-45.75%	0	-	-

PCLK = 4 MHz						
Baud Rate	Double baud rate			Single baud rate		
	CNT	Actual Baud Rate	Error Rang (%)	CNT	Actual Baud Rate	Error Rang (%)
2400	104	2403.85	0.16%	52	2403.85	0.16%
4800	52	4807.69	0.16%	26	4807.69	0.16%
9600	26	9615.38	0.16%	13	9615.38	0.16%
19200	13	19230.77	0.16%	7	17857.14	-6.99%
38400	7	35714.29	-6.99%	3	41666.67	8.51%
57600	4	62500.00	8.51%	2	62500.00	8.51%
76800	3	83333.33	8.51%	2	62500.00	-18.62%
115200	2	125000.00	8.51%	1	125000.00	8.51%

PCLK = 10 MHz						
Baud Rate	Double baud rate			Single baud rate		
	CNT	Actual Baud Rate	Error Rang (%)	CNT	Actual Baud Rate	Error Rang (%)
2400	260	2403.85	0.16%	130	2403.85	0.16%
4800	130	4807.69	0.16%	65	4807.69	0.16%
9600	65	9615.38	0.16%	33	9469.70	-1.36%
19200	33	18939.39	-1.36%	16	19531.25	1.73%
38400	16	39062.50	1.73%	8	39062.50	1.73%
57600	11	56818.18	-1.36%	5	62500.00	8.51%
76800	8	78125.00	1.73%	4	78125.00	1.73%
115200	5	125000.00	8.51%	3	104166.67	-9.58%

PCLK = 14 MHz						
Baud Rate	Double baud rate			Single baud rate		
	CNT	Actual Baud Rate	Error Rang (%)	CNT	Actual Baud Rate	Error Rang (%)
2400	365	2397.26	-0.11%	182	2403.85	0.16%
4800	182	4807.69	0.16%	91	4807.69	0.16%
9600	91	9615.38	0.16%	46	9510.87	-0.93%
19200	46	19021.74	-0.93%	23	19021.74	-0.93%
38400	23	38043.48	-0.93%	11	39772.73	3.57%
57600	15	58333.33	1.27%	8	54687.50	-5.06%
76800	11	79545.45	3.57%	6	72916.67	-5.06%
115200	8	109375.00	-5.06%	4	109375.00	-5.06%

PCLK = 20 MHz						
Baud Rate	Double baud rate			Single baud rate		
	CNT	Actual Baud Rate	Error Rang (%)	CNT	Actual Baud Rate	Error Rang (%)
2400	521	2399.23	-0.03%	260	2403.85	0.16%
4800	260	4807.69	0.16%	130	4807.69	0.16%
9600	130	9615.38	0.16%	65	9615.38	0.16%
19200	65	19230.77	0.16%	33	18939.39	-1.36%
38400	33	37878.79	-1.36%	16	39062.50	1.73%
57600	22	56818.18	-1.36%	11	56818.18	-1.36%
76800	16	78125.00	1.73%	8	78125.00	1.73%
115200	11	113636.36	-1.36%	5	125000.00	8.51%

PCLK = 24 MHz						
Baud Rate	Double baud rate			Single baud rate		
	CNT	Actual Baud Rate	Error Rang (%)	CNT	Actual Baud Rate	Error Rang (%)
2400	625	2400.00	0.00%	313	2396.17	-0.16%
4800	313	4792.33	-0.16%	156	4807.69	0.16%
9600	156	9615.38	0.16%	78	9615.38	0.16%
19200	78	19230.77	0.16%	39	19230.77	0.16%
38400	39	38461.54	0.16%	20	37500.00	-2.34%
57600	26	57692.31	0.16%	13	57692.31	0.16%
76800	20	75000.00	-2.34%	10	75000.00	-2.34%
115200	13	115384.62	0.16%	7	107142.86	-6.99%

PCLK = 2 MHz						
Baud Rate	Double baud rate			Single baud rate		
	CNT	Actual Baud Rate	Error Rang (%)	CNT	Actual Baud Rate	Error Rang (%)
2400	52	2403.85	0.16%	26	2403.85	0.16%
4800	26	4807.69	0.16%	13	4807.69	0.16%
9600	13	9615.38	0.16%	7	8928.57	-6.99%
19200	7	17857.14	-6.99%	3	20833.33	8.51%
38400	3	41666.67	8.51%	2	31250.00	-18.62%
57600	2	62500.00	8.51%	1	62500.00	8.51%
76800	2	62500.00	-18.62%	1	62500.00	-18.62%
115200	1	125000.00	8.51%	1	62500.00	-45.75%

PCLK = 8 MHz						
Baud Rate	Double baud rate			Single baud rate		
	CNT	Actual Baud Rate	Error Rang (%)	CNT	Actual Baud Rate	Error Rang (%)
2400	208	2403.85	0.16%	104	2403.85	0.16%
4800	104	4807.69	0.16%	52	4807.69	0.16%
9600	52	9615.38	0.16%	26	9615.38	0.16%
19200	26	19230.77	0.16%	13	19230.77	0.16%
38400	13	38461.54	0.16%	7	35714.29	-6.99%
57600	9	55555.56	-3.55%	4	62500.00	8.51%
76800	7	71428.57	-6.99%	3	83333.33	8.51%
115200	4	125000.00	8.51%	2	125000.00	8.51%

PCLK = 11.0592 MHz						
Baud Rate	Double baud rate			Single baud rate		
	CNT	Actual Baud Rate	Error Rang (%)	CNT	Actual Baud Rate	Error Rang (%)
2400	288	2400.00	0.00%	144	2400.00	0.00%
4800	144	4800.00	0.00%	72	4800.00	0.00%
9600	72	9600.00	0.00%	36	9600.00	0.00%
19200	36	19200.00	0.00%	18	19200.00	0.00%
38400	18	38400.00	0.00%	9	38400.00	0.00%
57600	12	57600.00	0.00%	6	57600.00	0.00%
76800	9	76800.00	0.00%	5	69120.00	-10.00%
115200	6	115200.00	0.00%	3	115200.00	0.00%

PCLK = 16 MHz						
Baud Rate	Double baud rate			Single baud rate		
	CNT	Actual Baud Rate	Error Rang (%)	CNT	Actual Baud Rate	Error Rang (%)
2400	417	2398.08	-0.08%	208	2403.85	0.16%
4800	208	4807.69	0.16%	104	4807.69	0.16%
9600	104	9615.38	0.16%	52	9615.38	0.16%
19200	52	19230.77	0.16%	26	19230.77	0.16%
38400	26	38461.54	0.16%	13	38461.54	0.16%
57600	17	58823.53	2.12%	9	55555.56	-3.55%
76800	13	76923.08	0.16%	7	71428.57	-6.99%
115200	9	111111.11	-3.55%	4	125000.00	8.51%

PCLK = 22.12 MHz						
Baud Rate	Double baud rate			Single baud rate		
	CNT	Actual Baud Rate	Error Rang (%)	CNT	Actual Baud Rate	Error Rang (%)
2400	576	2400.17	0.01%	288	2400.17	0.01%
4800	288	4800.35	0.01%	144	4800.35	0.01%
9600	144	9600.69	0.01%	72	9600.69	0.01%
19200	72	19201.39	0.01%	36	19201.39	0.01%
38400	36	38402.78	0.01%	18	38402.78	0.01%
57600	24	57604.17	0.01%	12	57604.17	0.01%
76800	18	76805.56	0.01%	9	76805.56	0.01%
115200	12	115208.33	0.01%	6	115208.33	0.01%

19.6 UART Registers

Tab 19.6-1 UART Registers

offset	Register	Reset value	Description
UART1 Base address : 0x4000 0000			
UART2 Base address : 0x4000 0400			
0x00	UARTx_SCON	0x0000_0000	Control Register
0x04	UARTx_SBUF	0x0000_0000	Data Register
0x08	UARTx_SADDR	0x0000_0000	Address Register
0x0C	UARTx_SADEN	0x0000_0000	Address Mask Register
0x10	UARTx_INTSR	0x0000_0000	Interrupt Flag Register
0x14	UARTx_INTCLR	0x0000_0000	Interrupt Flag Clear Register
0x18	UARTx_BAUDCR	0x0000_0000	Baud Rate Control Register
0x1C	UARTx_IRDACR	0x0000_0000	IrDA Control Register

19.7 Register Description

19.7.1 UART Control Register(UART_SCON)

Register	Address offset	Access	Reset value	Description
UART_SCON	0x00	RW	0x0000_0000	UART Control Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						DBA UD	FEEN
7	6	5	4	3	2	1	0
SM01[1:0]		SM2	REN	TB8	RB8	TIEN	RIEN

UART Control register(UART_SCON)bit description

Bit	Access	Description
[31:10]	-	Reserved

[9]	RW	DBAUD : Double baud rate 0:Double baud rates 1:Single baud rates																									
[8]	RW	FEEN : Receiving frame error to interrupt enable																									
[7 : 6]	RW	<p>SM0:SM1 worke mode</p> <table border="1"> <thead> <tr> <th>SM0</th> <th>SM1</th> <th>MODE</th> <th>Description</th> <th>Bit Rates</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>offset register</td> <td>CLK/12</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>serial transmission as 8-bit</td> <td>Variable baud rates</td> </tr> <tr> <td>1</td> <td>0</td> <td>2</td> <td>serial transmission as 9-bit</td> <td>CLK/32, CLK/64</td> </tr> <tr> <td>1</td> <td>1</td> <td>3</td> <td>serial transmission as 9-bit</td> <td>Variable baud rates</td> </tr> </tbody> </table> <p>00: Mode 0;01: Mode 1; 10: Mode 2; 11: Mode 3</p>	SM0	SM1	MODE	Description	Bit Rates	0	0	0	offset register	CLK/12	0	1	1	serial transmission as 8-bit	Variable baud rates	1	0	2	serial transmission as 9-bit	CLK/32, CLK/64	1	1	3	serial transmission as 9-bit	Variable baud rates
SM0	SM1	MODE	Description	Bit Rates																							
0	0	0	offset register	CLK/12																							
0	1	1	serial transmission as 8-bit	Variable baud rates																							
1	0	2	serial transmission as 9-bit	CLK/32, CLK/64																							
1	1	3	serial transmission as 9-bit	Variable baud rates																							
[5]	RW	<p>SM2 Multi-master communicate 0:Disable , 1:Enable SM2:Software configuration multi-machine communication and automatic address match- ing mode. 1: Start multi-slave communication and automatic address matching 0: Disable multi-slave communication and address auto-matching Mode 2 /Mode 3</p> <ul style="list-style-type: none"> • If SM2 = 1 and REN = 1, the receiver is in the address frame monitoring mode, and the received ninth of RB8 can be used for address filtering. RB8=1 is the address frame, communication data can enter SBUF, set RI, and enter the interrupt service program for address comparison;RB8=0 are data frames, the receiver ignores these data frames and keeps RI=0. • If SM2 = 0 and REN = 1, the receiver does not use the address monitoring mode, regardless of the received RB8 is 0 or 1, both receive directly and enter SUBF, setRI,RB8 is checksum in this mode bit. 																									
[4]	RW	<p>REN receive enable Mode0: 0: send,1: receive other: 0: send,1: receive/send</p>																									
[3]	RW	TB8 send TB8 bit																									
[2]	RW	RB8 Receive RB8 Bit																									
[1]	RW	<p>TIEN Transmit complete interrupt enable 0:Disable ,1:Enable</p>																									
[0]	RW	<p>RIEN Receive complete interrupt enable 0:Disable ,1:Enable</p>																									

19.7.2 UART Data Register(UARTx_SBUF)

Register	Address offset	Access	Reset value	Description
UARTx_SBUF	0x04	RW	0x0000_0000	UART Data Register

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

UART Data Register(UARTx_SBUF)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	SBUF[7:0]: When sending data, write the sending data into this register; when receiving data, read it from this register after receiving the data. Note: The value read to this register is actually the value in RXBuffer, and the value written to this register is actually written to TXShifter.

19.7.3 UART Address Register(UARTx_SADDR)

Register	Address offset	Access	Reset value	Description
UARTx_SADDR	0x08	RW	0x0000_0000	UART Address Register

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

UART Address Register(UARTx_SADDR)bit description

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	SADDR[7:0]: Slave Device Address Register

19.7.4 UART Address Mask Register(UARTx_SADEN)

Register	Address offset	Access	Reset value	Description
UART_SADEN	0x0C	RW	0x0000_0000	UART Address Mask Register

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

UART Address Mask Register

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	SADEN[7:0] Slave Device Address Mask Register

19.7.5 UART Interrupt Flag Register(UART_INTSR)

Register	Address offset	Access	Reset value	Description
UART_INTSR	0x10	R	0x0000_0000	UART Interrupt Flag Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					FE	TI	RI

UART Interrupt Flag Register

Bit	Access	Description
[31:3]	-	Reserved
[2]	R	FE: Receive frame error flag, set by hardware, cleared by software 0:FE Interrupt invalid 1:FE Interrupt valid
[1]	R	TI: Transmit complete interrupt flag, set by hardware, cleared by software 0:TI Interrupt invalid 1:TI interrupt valid
[0]	R	RI: Receive completion interrupt flag, set by hardware, cleared by software 0:RIinterrupt invalid 1:RIinterrupt valid

19.7.6 UART Interrupt Flag Clear Register(UART_INTCLR)

Register	Address offset	Access	Reset value	Description
UART_INTCLR	0x14	W	0x0000_0000	UART Interrupt Flag Clear Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					FECLR	TICLR	RICLR

UART Interrupt Flag Clear Register

Bit	Access	Description
[31:3]	-	Reserved
[2]	W	FECLR: Clear receive frame error flag; write 1 clear, write 0 invalid
[1]	W	TICLR: Clear the sending completion interrupt flag; write 1 clear, write 0 invalid
[0]	W	RICLR: Clear the receive completion interrupt flag; write 1 clear, write 0 invalid

19.7.7 UART Baud Rate Control Register(UART_BAUDCR)

Register	Address offset	Access	Reset value	Description
UART_BAUDCR	0x18	RW	0x0000_0000	UART Baud Rate Control Register

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

UART Baud Rate Control Register(UART_BAUDCR)Bit Description

Bit	Access	Description
[31:17]	-	Reserved
[16]	RW	SELF_BRG: UART Baud rate selection bits: 0:UART baud rate is generated by timer 1: The baud rate of UART is determined by $(DBAUD+1)*F_{PCLK}/(32*(BRG[15:0]+1))$ generate
[15:0]	RW	BRG[15:0]: UART Auto-Baud Generation Configuration Bits: Baud rate= $(DBAUD+1)*F_{PCLK}/(32*(BRG[15:0]+1))$

19.7.8 UART IrDA Control Register(UART_IRDACR)

Register	Address offset	Access	Reset value	Description
UART_IRDACR	0x1C	RW	0x0000_0000	UART IrDA Control Register

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

UART IrDA Control Register

Bit	Access	Description
[31:12]	-	Reserved
[11]	RW	IRLPMODE: Ir Low Power Mode 0: Ir Normal mode 1: Ir Low Power Mode
[10]	RW	IRRXINV: IrRXD Data Inversion Bit 0: Do not invert 1: Invert output
[9]	RW	IRTXINV: IrTXD Data Toggle Bit 0: Do not invert 1: Invert output
[8]	RW	IREN: IrDA Enable bit 0: invalid 1: Enable
[7:0]	RW	PSC[7:0]: Infrared mode sending, receiving mode filter frequency division Divide the system clock frequency to achieve low power consumption

20 LPUART

20.1 LPUART Introduction

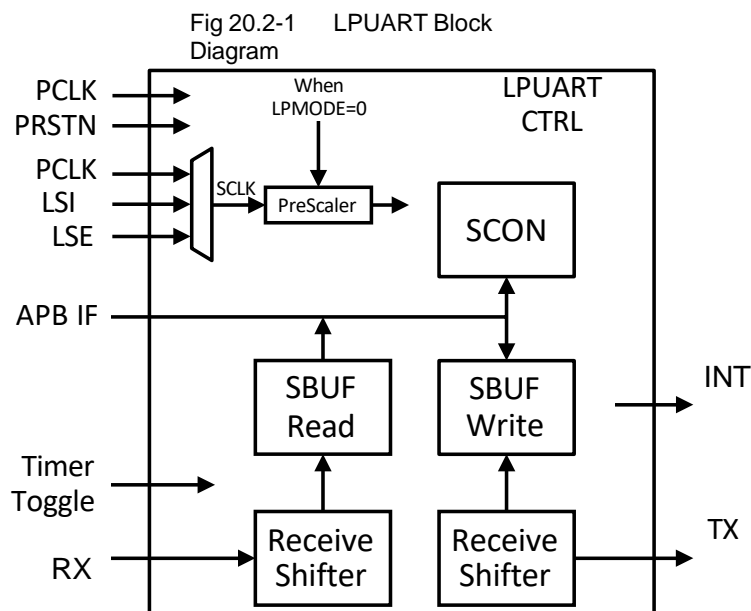
MG32L003 has a LPUART module(Low Power Universal Asynchronous Receiver/Transceiver), which supports half-duplex and full-duplex transmission; support 8BIT and 9BIT data formats; support four different transmission modes of Mode 0/1/2/3. The baud rate of LPUART is generated by LPTIM, and it can also be generated by an internal automatic baud rate generator; support multi-machine communication mode; support automatic address recognition; support a given address and broadcast address, and support low power consumption mode

To support low-power consumption applications, LPUART adds a SCLK clock in addition to the original PCLK clock, and can control the working state of LPUART. The internal register configuration logic of the LPUART module works in the PCLK clock domain, and the data transmission logic works in the SCLK clock domain. When the system enters the low power consumption mode and is in the LPUART operating state, turn off the high-frequency PCLK clock, and turn on the low-frequency SCLK clock, and the LPUART can still conduct normal data transmission and reception. Turn off the working state to stop the generation of the baud rate

SCLK clock sources can be selected:PCLK, external low-speed clock(LSE), and internal low-speed clock(LSI). When LPMODE=1, the SCLK clock also supports 1/2/4/8/16/32/64/128 times prescaler.

Note : that when LPMODE=0,LPUART receives the TOGGLE output signal of the LPTIM clock instead of the overflow signal, so it is necessary to enable the TOGGLE output of the LPTIM

20.2 LPUART Diagram



20.3 Operating Modes

LPUART adds a LPMODE control bit compared to general UART(UART1/1). When the position is "1", only Mode 1/3 working mode is supported, and the baud rate generation method will also change. For detailed descriptions, please refer to the following sections.

20.3.1 Mode 0(synchronous mode, half duplex)

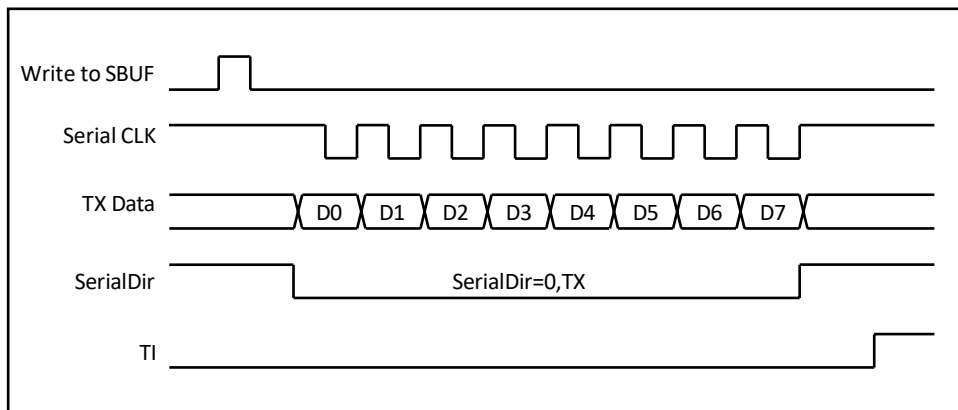
When working in Mode 0,the UART operates in synchronous mode with a baud rate of 1/12 of the fixed SCLK clock. UART receives data that is input by RXD,and UART sends data that is output by RXD. RXD is the input and output port at this time. The UART synchronous shift clock is output by TXD, which is the output port at this time. Note that this mode can only be used as a master to send a synchronous shift clock, and cannot be used as a slave to receive the clock from the outside. In this mode, the bit width of the transmitted data can only be 8 bits, with no start and stop bits.

Clear LPUARAT_SCON.SM0 and LPUART_SCON.SM1 to enter Mode 0 operation mode. When LPMODE = 1,Mode 0 working mode is not supported.

Send data

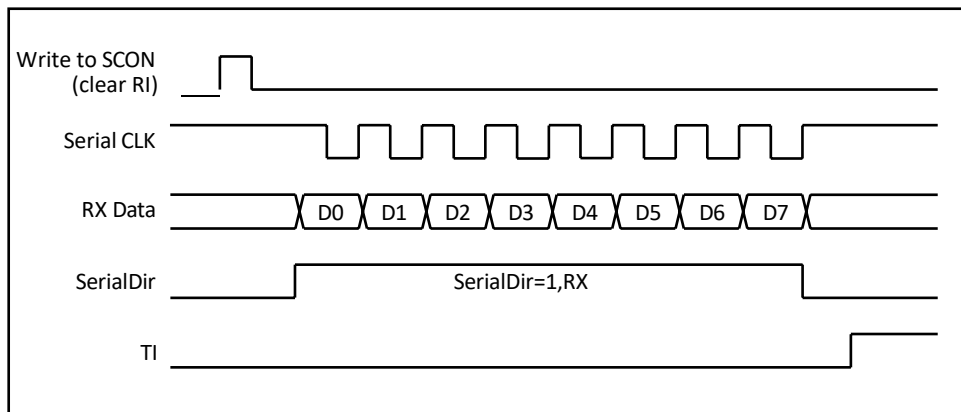
When sending data, clear the LPUART_SCON.REN bit and write the data into the LPUART_SBUF register. At this time, the transmission data will be output from RXD(low bit first, high bit last), and the synchronous shift clock will be output from TXD.

Fig 20.3-1 LPUART Output mode0



Receive data

When receiving data, set LPUART_SCON.REN bit to 1 and clear LPUART_INTSR.RI bit. When the reception ends, the data can be read out from LPUART_SBUF register. At this time, the received data are input from RXD(low bit first, high bit last), and the synchronous shift clock is output from TXD.



20.3.2 Mode 1(asynchronous mode, full duplex)

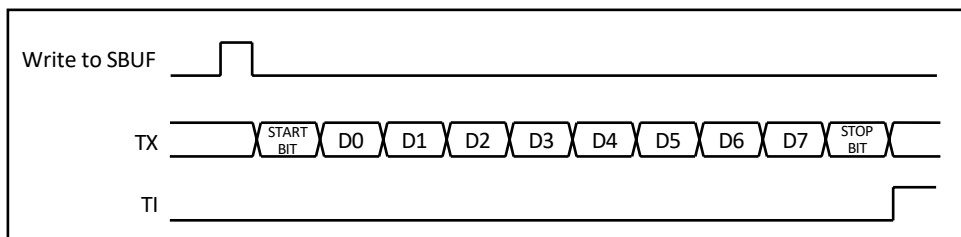
When operating in Mode 1, the transmission data is transmitted through TXD, and the received data is received through RXD. The data consists of 10 bits: the starting bit "0" starts, followed by 8 data bits(low bit first, high bit last), and finally the ending bit "1".

Clear LPUART_SCON.SM0 to 0 and set LPUART_SCON.SM1 to 1 to enter Mode 1 working mode. In this mode, when LPMODE = 0, the baud rate of LPUART can be selected to be generated by an automatic baud rate generator or timer LPTIM module, and it is programmable. When LPMODE = 1, the baud rate calculation method will change.

Send data

When sending data, regardless of the value of LPUART_SCON.REN, write the sent data into LPUART_SBUF register,and the data will be moved out of TXD(low bit first, high bit last).

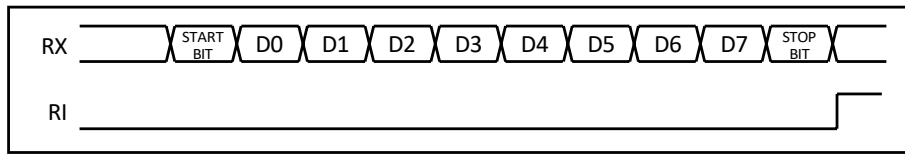
Fig 20.3-3 LPUART Output mode1



Receive data

When receiving data, set LPUART_SCON.REN bit to 1 and clear LPUART_INTSR.RI bit 0. Start receiving RXD data(low bit first, high bit last). When the reception is completed, it can be read out from LPUART_SBUF register.

Fig 20.3-4 LPUART Receive mode1



20.3.3 Mode 2(asynchronous mode, full duplex)

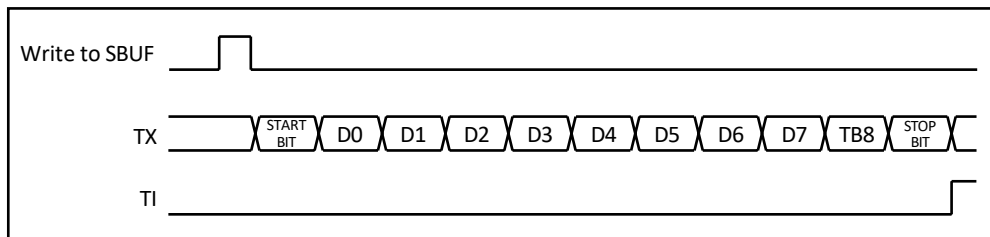
When working in Mode 2, the transmission data is transmitted through TXD, and the received data is received through RXD. The data consists of 11 bits: the start bit "0" starts, followed by 8 data bits, 1 TB8 bit and the stop bit. The additional TB8 bits are used in the multi-machine communication loop. When TB8=1, it indicates that the address frame is received; when TB8=0, it indicates that the received frame is a data frame. When multi-machine communication is not required, this bit can also be used as a parity check bit.

Set LPUART_SCON.SM0 to 1 and clear LPUART_SCON.SM1 to 0 to enter Mode 2 operation mode. In this mode, the baud rate can be generated independently without the need for an external Timer to be generated. When LPMODE=1, Mode 2 working mode is not supported.

Send data

When sending data, it has nothing to do with the value of LPUART_SCON.REN, and write the sent data into LPUART_SBUF register, the data will be moved out of TXD(low bit first, high bit last).

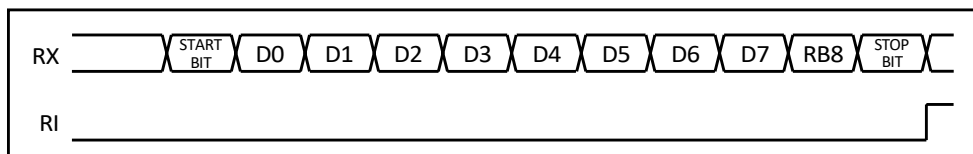
Fig 20.3-5 LPUART Output mode2



Receive data

When receiving data, set LPUART_SCON.REN bit to 1 and clear LPUART_INTSR.RI bit 0. Start receiving RXD data(low bit first, high bit last). When the reception is completed, it can be read out from LPUART_SBUF register.

Fig 20.3-6 LPUART Receive mode2



20.3.4 Mode 3(asynchronous mode, full duplex)

The data format, transmission timing and operation mode of Mode 3 are the same as those of Mode 2. The only difference is that the baud rate of Mode 3 is generated by LPTIM or an internal automatic baud rate generator, rather than be generated independently by the device itself as in Mode 2. The baud rate of Mode 3 is programmable and the baud rate is generated in the same way as Mode 1. Set LPUART_SCON.SM0 to 1 and LPUART_SCON.SM1 to 1 to enter Mode 3 working mode. When LPMODE=1, Mode 3 working mode is supported.

20.4 Baud rate programming

20.4.1 Mode 0

LPMODE = 0

When working in Mode 0, the baud rate is fixed at 1/12 of PCLK, and LPTIM support is not required.

LPMODE = 1

This mode is not supported when LPMODE = 1.

20.4.2 Mode 1/3

LPMODE = 0

When working in Mode 1 or Mode 3, the baud rate can be determined by the overflow time of LPTIM.

$$\text{BAUDRATE} = \frac{(\text{LPUART_SCON.DBAUD} + 1) * \text{F}_{\text{SCLK}}}{32 * (2^{16} - \text{LPTIM_BGLOAD}[15 : 0])} \quad (20.1)$$

LPUART_SCON.DBAUD represents double baud rate, and F_{SCLK} is the SCLK clock frequency, LPTIM_BGLOAD is the periodic load count value of LPTIM

Note: LPTIM must be configured for a 16-bit automatic reload mode, and the immediate reload register(LPTIM_LOAD) and the periodic reload register(LPTIM_BGLOAD) must write the same initial value.

It is also possible to use the own baud rate generation mode:

$$\text{BAUDRATE} = \frac{(\text{LPUART_SCON.DBAUD} + 1) * \text{F}_{\text{SCLK}}}{32 * (\text{LPUART_BAUDCR.BRG} + 1)} \quad (20.2)$$

UARTX_SCON.DBAUD means double baud rate, and F_{SCLK} represents the SCLK clock frequency.

LPMODE = 1

When LPMODE is set to “1”, the baud calculation formula is different from the above formula

$$\text{BAUDRATE} = \frac{\text{F}_{\text{SCLK}}}{4 * \text{LPUART_SCON.PRSC}} \quad (20.3)$$

F_{SCLK} is the clock frequency of SCLK, and LPUART_SCON.PRSC is the prescaler coefficient

20.4.3 Mode 2

LPMODE = 0

When working in Mode 2, the transmission clock can only select PCLK, and the baud rate is fixed at the value obtained by the following formula:

$$\text{BAUDRATE} = \frac{(\text{LPUART_SCON.DBAUD} + 1) * \text{F}_{\text{PCLK}}}{64} \quad (20.4)$$

LPUART_SCON.DBAUD represents double baud rate, and FPCLK represents PCLK clock frequency.

LPMODE = 1

This mode is not supported when LPMODE = 1.

20.5 Frame error detection

Mode 1/2/3 has a frame error detection function, and the hardware will automatically detect whether the received frame data has a valid STOP bit. If no valid STOP bit is received, LPUART_INTSR.FE is set to 1. LPUART_INTSR.FE bit is set to 1 by the hardware, and cleared to 0 by the software. If the software fails to clear 0 in time, even if the subsequent received data has a valid STOP bit, it will not clear LPUART_INTSR.FE flag to 0.

20.6 Multi-machine communication

Mode 2/3 has multi-machine communication function, so 1 bit TB8/RB8 is added to its frame format. Set LPUART_SCON.SM2 to "1" to enable the multi-machine communication bit. When the multi-machine communication bit is turned on, when sending data, the master can use LPUART_SCON.TB8 to distinguish whether the current frame is an address frame (LPUART_SCON.TB8=1) or a data frame (LPUART_SCON.TB8=0). When receiving data, the slave ignores the current received frame with RB8 bit (9th bit) being "0". When the RB8 bit (9th bit) of the received frame is "1", it indicates that it is an address frame, the slave will continue to judge whether the received address is equal to its own address from the opportunity. If it matches, the slave will set LPUART_SCON.RB8 to "1" and LPUART_INTSR.RI to "1", so as to indicate that the frame is an address frame and the address has already been matched. After the slave software views LPUART_SCON.RB8=1 and LPUART_INTSR.RI=1, first clear LPUART_SCON.SM2 to "0" and then prepare to accept the data frame addressed to it. If the addresses are not equal, it indicates that the master is not addressing the slave, and the slave hardware maintains LPUART_SCON.RB8 and LPUART_INTSR.RI as "0", and the software maintains LPUART_SCON.SM2 bit as "1", and the slave continues to be in the monitoring state.

20.7 Automatic Address Recognition

When the multi-machine communication bit is enabled (LPUART_SCON.SM2 is set to "1"), the automatic address recognition function will also be enabled. This function is implemented by hardware, enabling the slave to detect each address frame received. If the address matches the address of the slave, the receiver will give LPUART_INTSR.RI receive flag. If the address does not match, the receiving end will not give any acceptance flags.

If necessary, the multi-machine communication bit can also be enabled in Mode 1, where the TB8 bit is replaced by the STOP bit. When the slave receives a matching address frame and a valid STOP bit, LPUART_INTSR.RI will be set to "1". To support automatic address recognition, the concepts of broadcast address and given address are defined.

20.8 Given address

The LPUART_SADDR register of the LPUART device is used to represent the given address of its own device, and the LPUART_SADEN register is an address mask, which can be used to define the irrelevant bits in an address. When a certain bit of LPUART_SADEN is "0", it indicates that the address of this bit is irrelevant, which means that the bit address does not participate in address matching in the process of address matching,. These unrelated bits increase the flexibility of addressing, allowing the master to address one or more slave devices simultaneously.

Note : that the LPUART_SADEN register must be set to 0xFF if a unique matching address needs to be given

$$\text{GIVENADDR} = \text{SADDR} \ \& \ \text{SADEN} \quad (20.5)$$

20.9 Broadcast addresses

The broadcast address is used to address all slave devices simultaneously. The general broadcast address is 0xHFF

$$\text{BoardCastAddr} = \text{SADDR} \ || \ \text{SADEN} \quad (20.6)$$

20.9.1 Examples of given address and broadcast address

Assume that LPUART_SADDR and LPUART_SADEN of a slave are configured as follows:

SADDR: 0b01101001

SADEN: 0b11111011

Then the given address and broadcast address are as follows:

Given Address: 0b01101x01

BroadCast Address: 0b11111x11

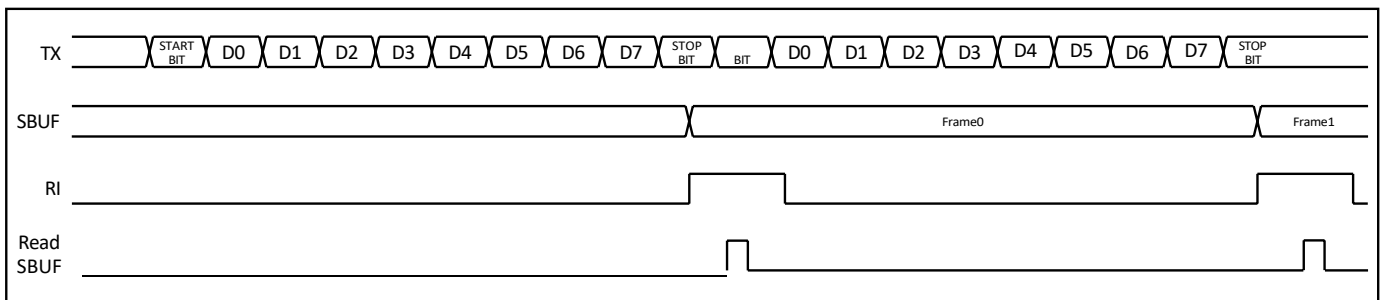
It can be seen that the master can address the local slave with four addresses, namely: 0b01101001 and 0b01101101(given address), 0b11111011 and 0b11111111(broadcast address).

20.10 Transceiver buffer

20.10.1 Receive buffer

The LPUART receiver has a receiving buffer with a frame length of 8/9 BITS. After receiving data of one frame, the data in the receiving buffer is kept until the STOP bit of the next frame is received. The receiving buffer is updated with the data of the next frame.

Fig 20.10-1 Receive buffer



20.10.2 Send buffer

The LPUART sender does not support send caching. If the LPUART_SBUF register is filled in during data transmission, the write operation will be masked. Software should avoid this operation

20.11 Registers

Tab 20.11-1 LPUART Registers

offset	Register	Reset value	Description
LPUART Base address : 0x4000 5000			
0x00	LPART_SBUF	0x0000_0000	Data Register
0x04	LPART_SCON	0x0000_E000	Control Register
0x08	LPART_SADDR	0x0000_0000	Address Register
0x0C	LPART_SADEN	0x0000_0000	Address Mask Register
0x10	LPART_INTSR	0x0000_0000	Interrupt Flag Register
0x14	LPART_INTCLR	0x0000_0000	Interrupt Flag Clear Register
0x18	LPART_BAUDCR	0x0000_0000	Baud Rate Control Register

20.12 Register Description

20.12.1 LPUART Data Register(LPUART_SBUF)

Register	Address offset	Access	Reset value	Description
LPUART_SBUF	0x00	RW	0x0000_0000	LPUART Data Register

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

LPUART Data Register(LPUART_SBUF)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	SBUF[7:0] : When sending data, when the sending data is written into this register; when receiving data, the data has been received After that, read from this register

20.12.2 LPUART Control Register(LPUART_SCON)

Register	Address offset	Access	Reset value	Description
LPUART_SCON	0x04	RW	0x0000_E000	LPUART Control Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							EN
15	14	13	12	11	10	9	8
PRSC[2:0]			SCLKSEL		LPMODE	DBAUD	TEEN
7	6	5	4	3	2	1	0
SM0:SM1		SM2	REN	TB8	RB8	TIEN	RIEN

LPUART Control register(LPUART_SCON)bit description

Bit	Access	Description
[31:17]	-	Reserved
[16]	RW	EN : Low-Power UART Operation Enable 0:Low-Power UART off, not receiving/sending data 1:Low-Power UART enable, must be set to 1 before data transmission
[15:13]	RW	PRSC[2:0] : Transmission clock SCLK Prescaler selection 000 : DIV128 ; 001 : DIV64 ; 010 : DIV32 ; 011 : 16 ; 100 : 8 ; 101 : 4 ; 110 : DIV2 ; 111 : DIV1 PRSC[2:0] is valid only when LPMODE=1; when LPMODE=0, PRS[2:0] will not prescale SCLK.
[12:11]	RW	SCLKSEL[1:0] Transmit Clock SCLK Select 00/01:PCLK 10:LSE 11:LSI
[10]	RW	LPMODE : low power mode 0: Normal working mode 1: Low Power Operation Mode
[9]	RW	DBAUD : double baud rate 0: Single baud rate 1: double baud rate
[8]	RW	TEEN :The send buffer affects writing SUBF 0:Sending buffering does not affect writing SBUF 1:SBUF can only be written if the send buffer is empty

[7:6]	RW	SM0:SM1 work mode; 00: Mode 0; 01: Mode 1; 10: Mode 2; 11: Mode 3				
		SM0	SM1	MODE	Description	baud rate
		0	0	0	offset register	PCLK/12
		0	1	1	serial transmission as 8-bit	Variable baud rates
		1	0	2	serial transmission as 9-bit	PCLK/32, PCLK/64
1	1	3	serial transmission as 9-bit	Variable baud rates		
[5]	RW	<p>SM2 Multi-master communicate 0: Disable , 1: Enable SM2: Software configuration multi-machine communication and automatic address matching mode. 1: Start multi-slave communication and automatic address matching 0: Disable multi-slave communication and address auto-matching Mode 2 /Mode 3</p> <ul style="list-style-type: none"> If SM2 = 1 and REN = 1, the receiver is in the address frame monitoring mode, and the received ninth of RB8 can be used for address filtering. RB8=1 is the address frame, communication data can enter SBUF, set RI, and enter the interrupt service program for address comparison; RB8=0 are data frames, the receiver ignores these data frames and keeps RI=0. If SM2 = 0 and REN = 1, the receiver does not use the address monitoring mode, regardless of the received RB8 is 0 or 1, both receive directly and enter SBUF, set RI, RB8 is checksum in this mode bit. 				
[4]	RW	<p>REN receive enable Mode0: 0: send, 1: receive other: 0: send, 1: receive/send</p>				
[3]	RW	TB8 send TB8 bit				
[2]	RW	RB8 Receive RB8 Bit				
[1]	RW	<p>TIEN Transmit complete interrupt enable 0: Disable , 1: Enable</p>				
[0]	RW	<p>RIEN Receive complete interrupt enable 0: Disable , 1: Enable</p>				

20.12.3 LPUART Address Register(LPUART_SADDR)

Register	Address offset	Access	Reset value	Description
LPUART_SADDR	0x08	RW	0x0000_0000	LPUART Address Register

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

LPUART Address Register(LPUART_SADDR)bit description

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	SADDR[7:0]: Slave Device Address Register

20.12.4 LPUART Address Mask Register(LPUART_SADEN)

Register	Address offset	Access	Reset value	Description
LPUART_SADEN	0x0C	RW	0x0000_0000	LPUART Address Mask Register

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

UART Address Mask Register

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	SADEN[7:0] Slave Device Address Mask Register

20.12.5 LPUART Interrupt Flag Register(LPUART_INTSR)

Register	Address offset	Access	Reset value	Description
LPUART_INTSR	0x10	R	0x0000_0000	LPUART Interrupt Flag Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					FE	TI	RI

LPUART Interrupt Flag Register

Bit	Access	Description
[31:3]	-	Reserved
[2]	R	FE: Receive frame error flag, set by hardware, cleared by software 0:FE Interrupt invalid 1:FE Interrupt valid
[1]	R	TI: Transmit complete interrupt flag, set by hardware, cleared by software 0:TI Interrupt invalid 1:TI interrupt valid
[0]	R	RI: Receive completion interrupt flag, set by hardware, cleared by software

20.12.6 LPUART Interrupt Flag Clear Register(LPUART_INTCLR)

Register	Address offset	Access	Reset value	Description
LPUART_INTCLR	0x14	W	0x0000_0000	LPUART Interrupt Flag Clear Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					FECLR	TICLR	RICLR

LPUART Interrupt Flag Clear Register

Bit	Access	Description
[31:3]	-	Reserved
[2]	W	FECLR: Clear receive frame error flag; write 1 clear, write 0 invalid
[1]	W	TICLR: Clear the sending completion interrupt flag; write 1 clear, write 0 invalid
[0]	W	RICLR: Clear the receive completion interrupt flag; write 1 clear, write 0 invalid

20.12.7 LPUART Baud Rate Control Register(LPUART_BAUDCR)

Register	Address offset	Access	Reset value	Description
LPUART_BAUDCR	0x18	RW	0x0000_0000	LPUART Baud Rate Control Register

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

LPUART Baud Rate Control Register(LPUART_BAUDCR)Bit Description

Bit	Access	Description
[31:17]	-	Reserved
[16]	RW	SELF_BRG: LPUART Baud rate selection bits: 0:LPUART baud rate is generated by LPTIMER 1: The baud rate of LPUART is determined by $(DBAUD+1)*F_{PCLK}/(32*(BRG[15:0]+1))$ generate
[15:0]	RW	BRG[15:0]: LPUART Auto-Baud Generation Configuration Bits: Baud rate= $(DBAUD+1)*F_{PCLK}/(32*(BRG[15:0]+1))$

21 I2C Interface

21.1 I2C Introduction

I2C is a two-wire bidirectional serial bus that provides a simple and efficient method for data exchange between devices. I2C standard a real multi-master bus with collision detection mechanism and arbitration mechanism. It prevents data collisions when two or more masters request control of the bus at the same time.

The I2C bus controller can meet various specifications of the I2C bus and support all transmission modes communicating with the I2C bus.

The I2C bus uses "SCL" (serial clock bus) and "SDA" (serial data bus) to connect devices to transfer information. The data is controlled by the SCL clock line between the master and the slave to realize a byte-by-byte synchronous transmission on the SDA data line, each byte is 8 bits long, and one SCL clock pulse transmits one data bit, and the data is determined by the highest bit MSB to start the transfer, and each transferred byte is followed by With an acknowledge bit, each bit is sampled while SCL is high; therefore, the SDA line can only change while SCL is low, and SDA remains stable while SCL is high. When SCL is high, a transition on the SDA line is interpreted as a command interrupt (START or STOP), and the I2C logic can handle byte transfers autonomously. It keeps track of serial transfers, and also has a status register (I2C_SR) that reflects the status of the I2C bus controller and the I2C bus.

21.2 I2C main features

The I2C controller supports the following features:

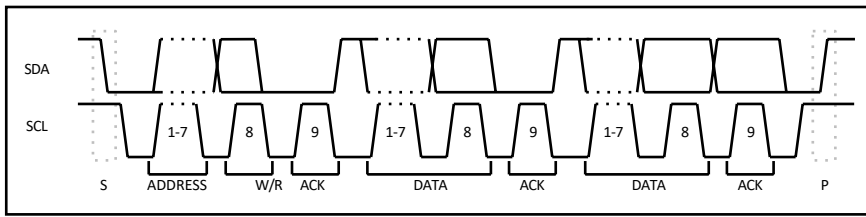
- Support four working modes of master sending/receiving and slave sending/receiving
- Support standard (100Kbps)/fast (400Kbps)/high-speed (1Mbps) three working speeds
- Support 7-bit addressing function
- Support noise filtering function
- Support broadcast address
- Support interrupt status query function

21.3 I2C Protocol description

Usually the standard I2C transmission protocol consists of four parts:

1. Start signal or repeated start signal
2. Slave address transfer and R/W bit transfer
3. Data transfer
4. Stop signal

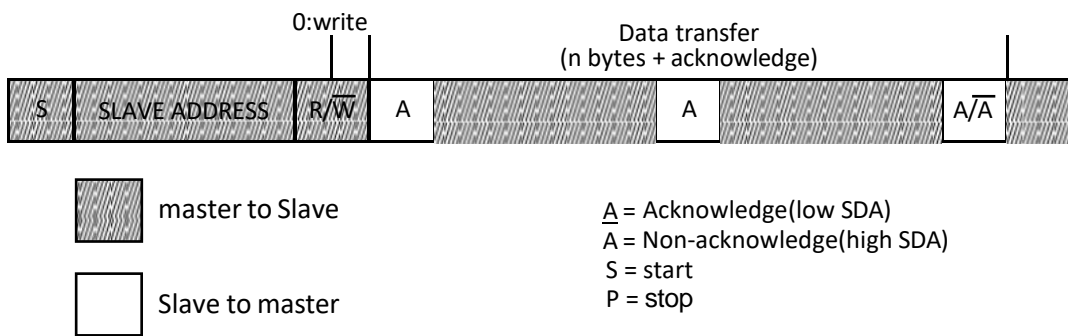
Fig 21.3-1 I2C Protocol description



21.3.1 Data transfer on the I2C bus

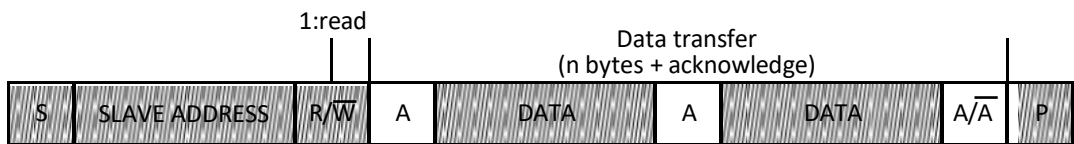
The master sends a slave to receive a 7-bit address (one byte), and the transfer direction remains unchanged.

Fig 21.3-2 I2C Transmission



After the first byte, the master reads the data from the slave (the content is the address of the slave), and the transmission direction changes.

Fig 21.3-3 I2C master reads the address from the slave



21.3.2 Start bit or Repeated start signal

When the bus is in an idle state, it means that there is no master to initiate a transfer request to the bus (the SCL and SDA lines are high at the same time), and the master can initiate a transfer request by sending a START signal.

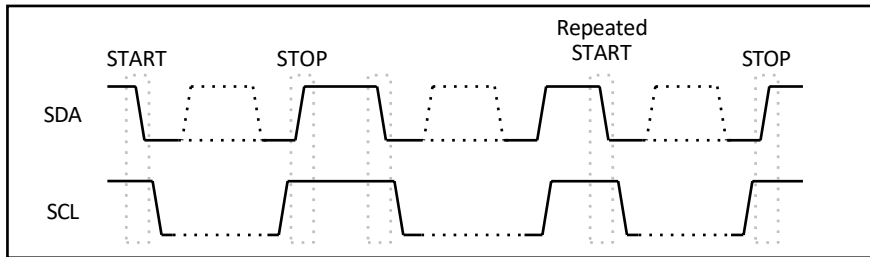
Start signal: usually expressed as S-bit, when the SCL line is high, the signal on the SDA line changes from high to low, indicating that a start signal is generated on the bus and a new transmission starts.

Repeated start signal (Sr): there is no STOP signal between two START signals. This method is used by a master to communicate with another slave or the same slave in a different transfer direction (for example: from writing to a device to reading from a slave) without releasing the bus.

STOP signal: The master sends a stop signal to the bus to end data transmission. Stop signal, usually represented by P-bit, when the SCL line is high, a signal from low to high appears on the SDA line,

which is defined as a stop signal.

Fig 21.3-4 I2C start/stop



21.3.3 Slave Address Transfer

When the START signal is the slave address, the master immediately transmits the first bit of data. This is a 7-bit call address followed by an RW bit, the RW bit controls the signal transmission direction of the slave. No two slaves in the system have the same address, and only the slave addressed by the master responds by setting SDA to low on the 9th SCL clock cycle.

21.3.4 Data transmission

When the slave address is successfully identified, it can start byte-by-byte data transmission according to the direction determined by RW, each transmission byte ends with a response signal on the 9th clock cycle. If no response signal (NACK) is generated from the slave machine, the master can either generate a stop signal to exit the data transmission, or generate a repeat start signal to start a new round of data transmission.

When the master is used as a receiving device, if there is no response signal (NACK), the slave releases the SDA line, so that the master generates a stop signal or a repeated start signal.

Fig 21.3-5 I2C bus transmission

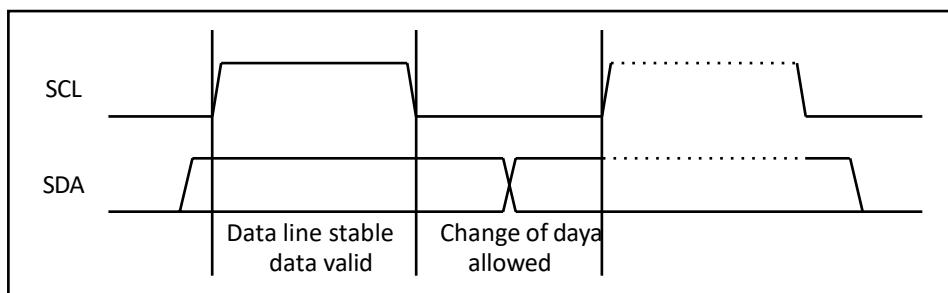
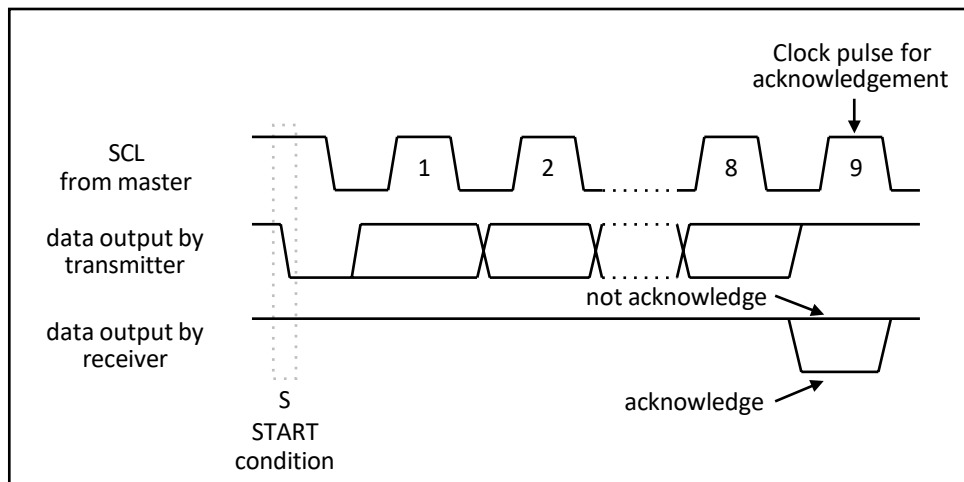


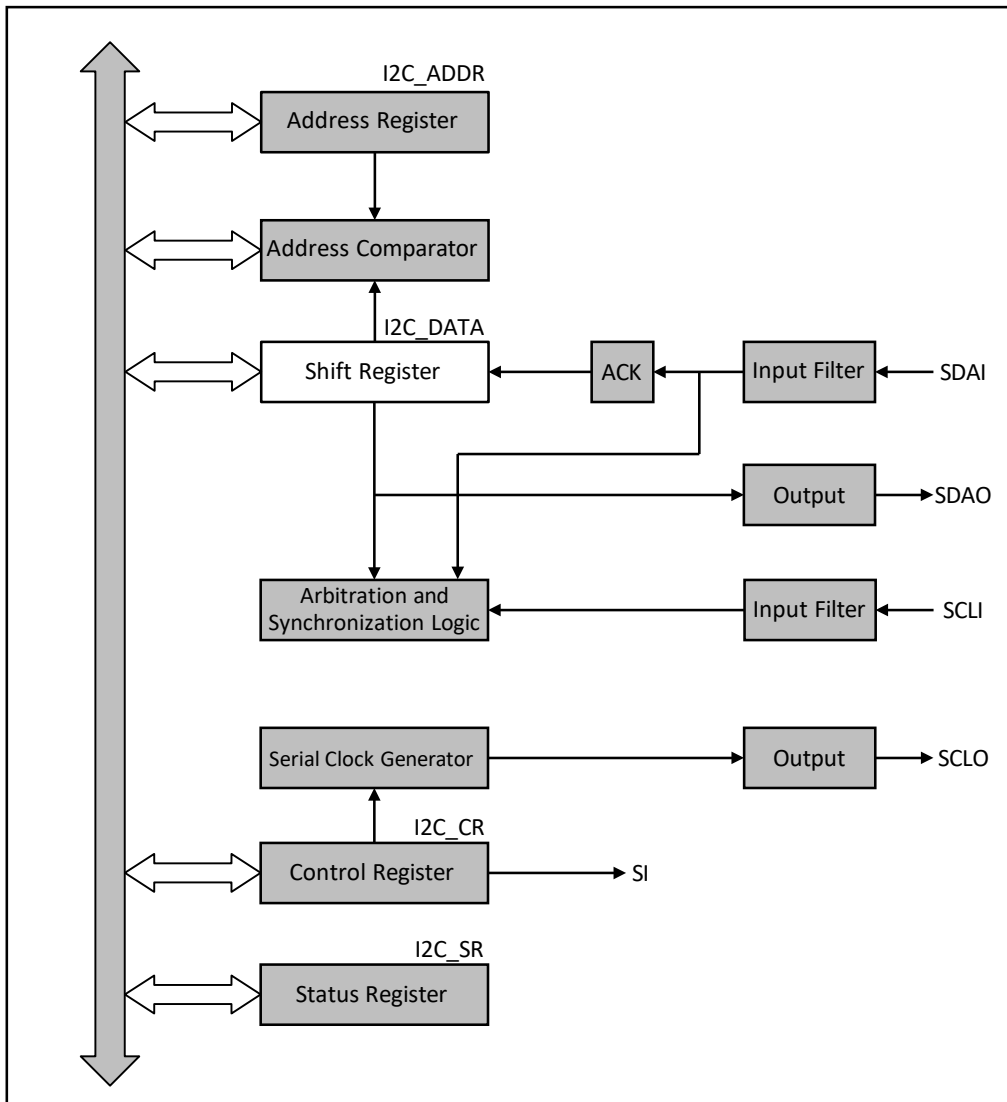
Fig 21.3-6 I2C bus bits



21.4 I2C description

The I2C bus uses two wires to transfer information between devices connected to the bus "SCL" (Serial Clock Line) and "SDA" (Serial Data Line). Since there are only non-directional ports, the I2C component requires the use of open-drain buffers to the pins. Every device connected to the bus can use the Software is addressed by a specific address. The I2C standard is a true multi-master bus with a collision detection mechanism and an arbitration mechanism. It prevents data conflicts when two or more masters start transmitting data at the same time. Filtering logic can filter burrs on the data bus to protect data integrity.

Fig 21.4-1 I2c Block Diagram



21.5 I2C Operating mode

I2C module can realize 8-bit bidirectional data transmission, the transmission rate can reach 100Kbits/s in standard mode, 400Kbits/s in high-speed mode, and 1Mbits/s in ultra-high-speed mode, and can be used in the following four modes:

1. master sending mode: When "SCL" outputs serial clock signal, "SDA" outputs serial data.
2. master receiving mode: Serial data is received through "SDA" when "SCL" outputs serial clock signal.
3. Slave receiving mode: Serial data and serial clock are received through "SDA" and "SCL" respectively.
4. Slave sending mode: When the serial clock is input from the "SCL" port, the serial data is sent through the "SDA" port.

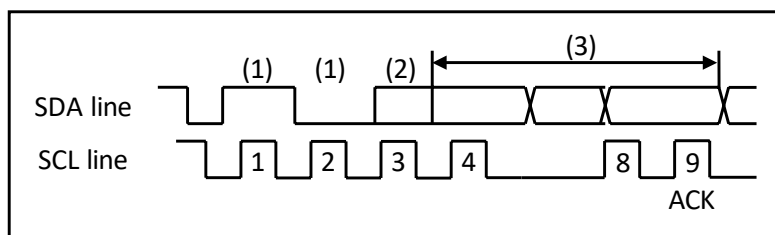
21.5.1 Arbitration and Synchronization Logic

In master transmit mode, the arbitration logic checks that each transmitted logic 1 is actually present on the bus. If another device on the bus deasserts a logic 1 and pulls the SDA line low, arbitration is lost and the I2C module immediately changes from a master transmitter to a slave receiver.

The I2C module will continue to output clock pulses (on SCL) until the current serial byte has been sent.

Arbitration may also be lost in master receive mode. This condition only occurs when the I2C module is returning a "not acknowledge(logic 1)" to the bus. Arbitration is lost when another device on the bus pulls the signal low. Since it only occurs at the end of serial bytes, the I2C module no longer generates clock pulses.

Fig 21.5-1 Mediation on I2C bus



1. The other device sends serial data;
2. The other device first undoes a logical 1 sent by the I2C master by pulling down the SDA (dotted line). Arbitration is lost and I2C enters slave receive mode
3. At this time, the I2C is in slave receive mode, but still generates clock pulses until the current byte is sent. I2C will not generate a clock pulse for the next byte transmission. Once arbitration is won, the data transfer on SDA is initiated by the new master

Synchronization logic synchronizes the serial clock generator to clock pulses on the SCL line of another device. If 2 or more master devices generate clock pulses, the high period is determined by the device producing the shortest high time; the low period is determined by the device producing the longest low time.

21.5.2 serial clock generator

The serial clock generator uses an 8-bit counter as the baud rate generator, and the frequency relationship between the SCL signal and the PCLK signal is $F_{SCL} = F_{PCLK} / 8 * (N + 1)$

table below shows the frequency value of SCL signal when PCLK is various frequencies and the frequency division factor is 1-7.

Tab 21.5-1 I2C Clock Baud Rate

Frequency(KHz)	1	2	3	4	5	6	7
1000	62	41	31	25	20	17	15
2000	125	83	62	50	41	35	31
4000	250	166	125	100	83	71	62
6000	375	250	187	150	125	107	93
8000	500	333	250	200	166	142	125
10000	625	416	312	250	208	178	156
12000	750	500	375	300	250	214	187
14000	875	583	437	350	291	250	218
16000	1000	666	500	400	333	285	250

21.5.3 Input filter

The input signal is synchronized with the clock signal (clk), and the spike pulse signal lower than 3 clock cycles will be filtered out. Each filter consists of 3 flip-flop. The first flipflop is used to directly latch the input signal and load the data into the shift register formed by the other two. When the states of the second and third flipflop are "11" or "00", the internal filter signal is set to 1 or 0 respectively.

21.5.4 Address comparator

The I2C comparator compares its own slave address with the received 7-bit slave address. It can program its own slave address using the "I2C_ADDR" register. And it will be compared with the first received 8-bit or with the general call address (0x00) according to the "GC" bit of the "I2C_ADDR" register. If any one is the same, the "SI" bit of the "I2C_CR" register will be set to 1. If the same, the "SI" bit of the "I2C_CR" register will be set to 1 and generate an interrupt request.

21.5.5 Interrupt generator

When all four modes of the I2C module are used, there are 26 possible bus states. When I2C enters 25 of the 26 states, the "SI" flag in the "I2C_CR" register will be set to 1 by hardware. The only state in which the SI bit is not set to 1 is 0xF8, which indicates that there is no valid status information. The "SI" flag must be cleared by software. In order to clear the "SI" bit, a 0 must be written into this position. Writing 1 in "SI" will not change the value of "SI". In order to determine the actual source of the interrupt, the interrupt service routine before clears the "SI" flag, the I2C status register will be queried.

21.5.6 I2C Master Transmit Mode

ENS must be set to "1" to enable the I2C module. If the AA bit is reset, the I2C module will not acknowledge its own slave address or general call address while another device is becoming the bus master. In other words, if the AA bit is reset, the I2C interface cannot enter slave mode. STA, STO and SI must be reset.

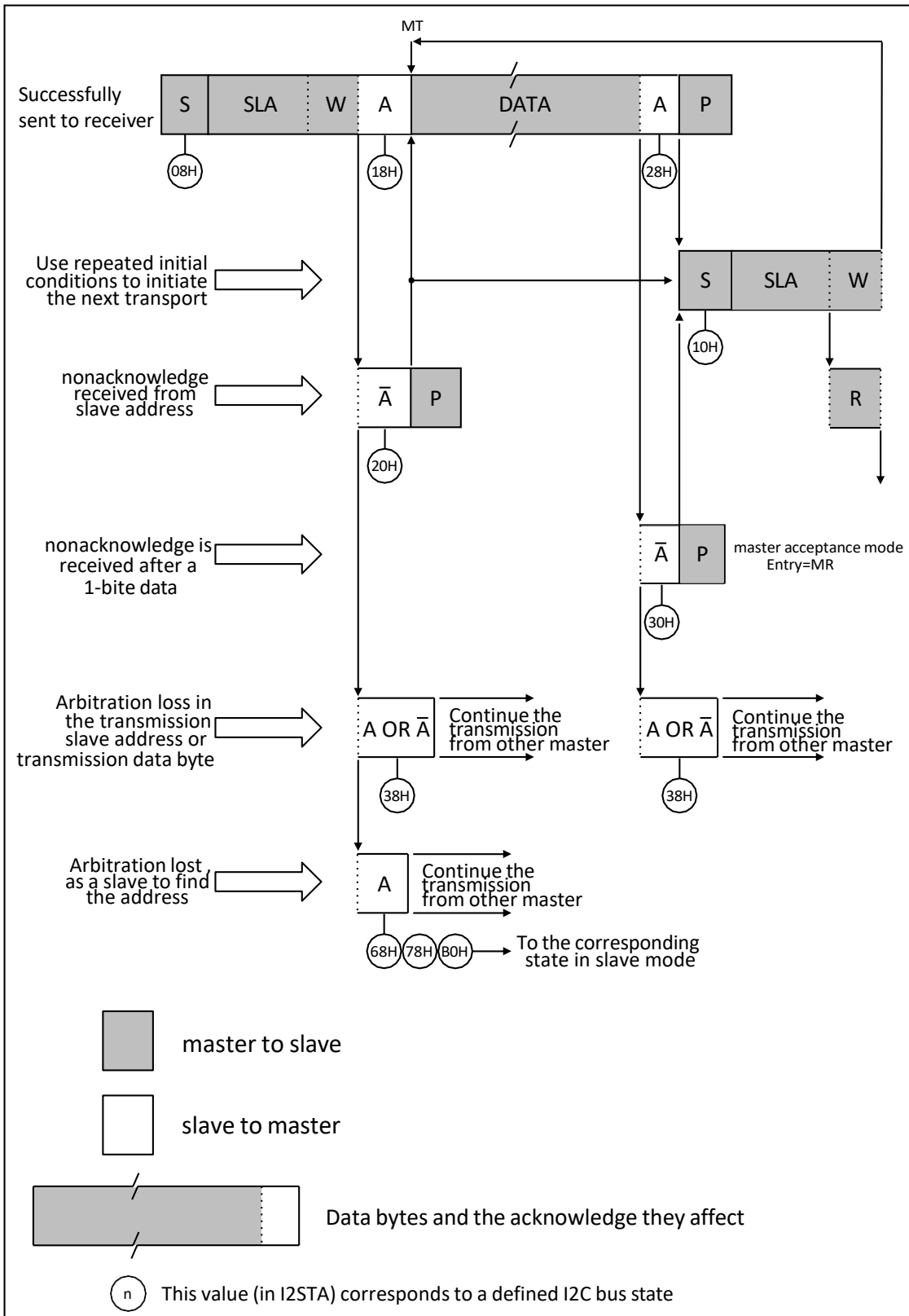
At this point, the master transmit mode can be entered by setting the STA bit. Once the bus is free, the I2C logic will immediately test the I2C bus and generate a starting condition. When a start condition

is sent, the serial interrupt flag (SI) is set, and the status code in the status register (I2C_SR) is 0x08. The interrupt service routine uses this status code to enter the corresponding status service routine, and loads the slave address and data direction bit (SLA+W) into I2C_DATA. The SI bit of I2C_CR must be reset before serial transmission can continue. When the slave address and direction bits have been sent and an acknowledge bit has been received, the Serial Interrupt Flag (SI) is set again, possibly a series of different status codes in the I2C_SR. It is 0x18, 0x20 or 0x38 in master mode, and 0x68, 0x78 or 0xB0 in slave mode (AA=1). The actions for each status code are detailed in the table below 21.5-2. After sending a repeated start condition (state 0x10), the I2C module switches to master receive mode by loading SLA+R into I2C_DATA

Tab 21.5-2 Master Transmit Mode Status

Status Code	I2C bus and hardware status	Application software response					I2C next action performed by the hardware
		R/W I2C_DATA	write I2C_CR				
			STA	STO	SI	AA	
08H	Start condition has been sent	Load SLA+W	X	0	0	X	Will send SLA+W, receive ACK
10H	Have sent repeated starting conditions	Load SLA+W	X	0	0	X	Will send SLA+W, receive ACK
		input SLA+R	X	0	0	X	Will send SLA+R, I2C Auto-switch to master receive mode
18H	Sent SLA+W received ACK	Load data byte	0	0	0	X	will send Data bytes, Will receive ACK
		None I2C_DATA action	1	0	0	X	Repeated start condition will send
		None I2C_DATA action	0	1	0	X	will send stop condition and the STO reset
		None I2C_DATA action	1	1	0	X	The stop condition is sent first, followed by the start condition, the STO reset
20H	SLA+W has been sent , non-ACK received	Load data byte	0	0	0	X	Data bytes will send, Will receive ACK
		None I2C_DATA action	1	0	0	X	duplicate start condition will send
		None I2C_DATA action	0	1	0	X	will send stop condition and the STO reset
		None I2C_DATA action	1	1	0	X	The stop condition is sent first, followed by the start condition, and the STO reset
28H	Data in I2C_DATA has been sent, ACK received	Load data byte	0	0	0	X	Data bytes will send, Will receive ACK
		None I2C_DATA action	1	0	0	X	duplicate start condition will send
		None I2C_DATA action	0	1	0	X	will send stop condition and the STO reset
		None I2C_DATA action	1	1	0	X	The stop condition is sent first, followed by the start condition, and the STO reset
30H	Data in I2C_DATA has been sent	Load data byte	0	0	0	X	Data bytes will send, Will receive ACK
		None I2C_DATA action	1	0	0	X	duplicate start condition will send
		None I2C_DATA action	0	1	0	X	will send stop condition and the STO reset
		None I2C_DATA action	1	1	0	X	The stop condition is sent first, followed by the start condition, and the STO reset
38H	Lost arbitration when SLA+R/W or writing data bytes	None I2C_DATA action	0	0	0	X	I2C Bus release, Enter unaddressable slave mode
		None I2C_DATA action	1	0	0	X	Send the start condition when the I2C bus is idle

Fig 21.5-2 I2C Functional block diagram



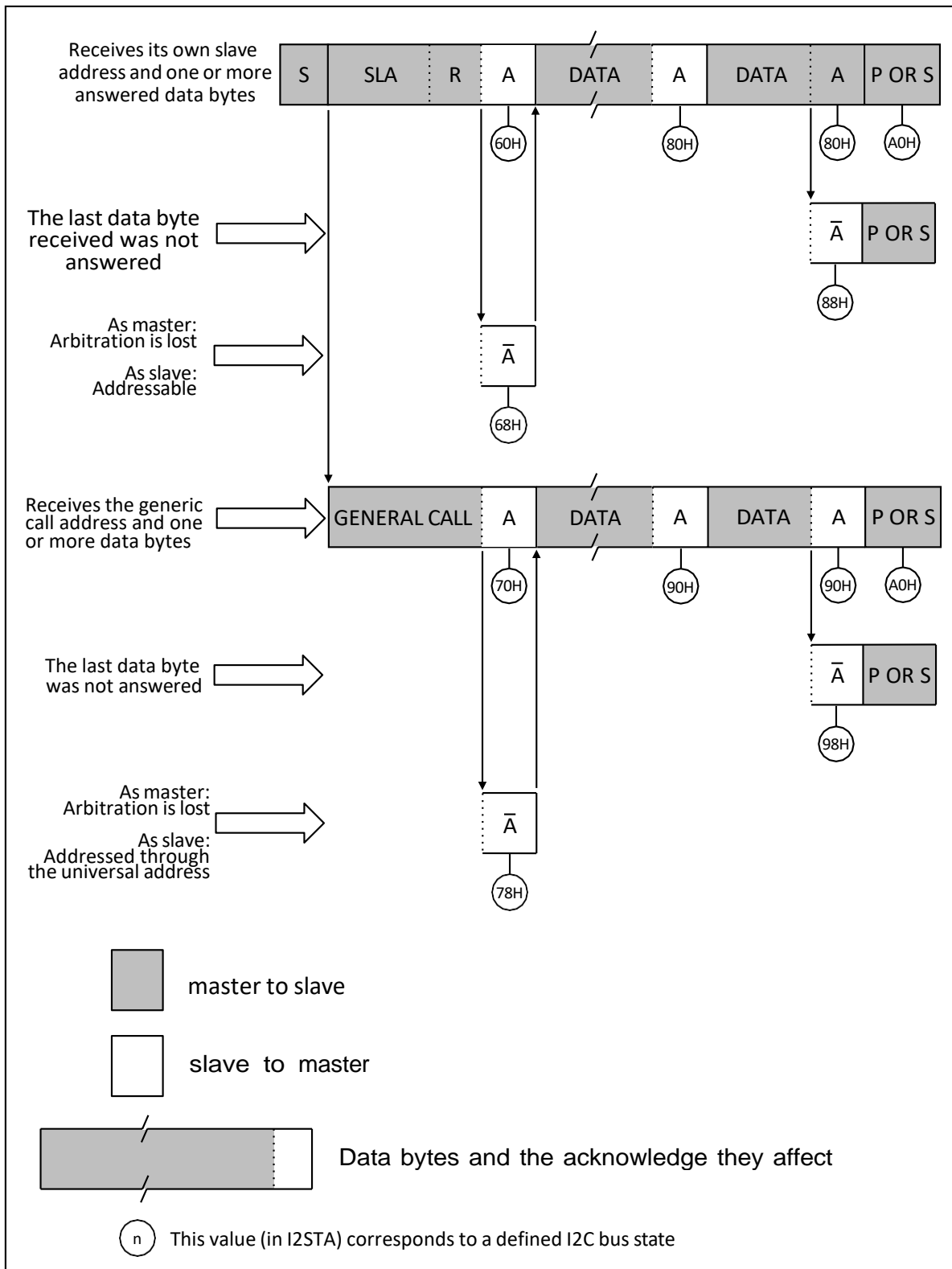
21.5.7 I2C Master Receive Mode

In master receive mode, the data bytes received by the master come from the slave transmitter. Initiate transfer as in master send mode. After sending the start condition, the interrupt service routine must load the 7-bit slave address and data direction bit (SLA+R) I2C_DATA. The SI bit in I2C_CR must be cleared before proceeding with the serial transfer. When the slave address and data direction bits are sent and an acknowledge bit is received, the serial interrupt flag SI is set again. At this time, a series of different status codes may be in the I2C_SR. It is 0x40, 0x48 or 0x38 in master mode, and 0x68, 0x78 or 0xB0 in slave mode (AA=1). The operation corresponding to each status code is detailed in the table below 21.5-3. After sending the Repeated START condition (state 0x10), the I2C module switches to Master Transmit mode by loading SLA+W into I2C_DATA

Tab 21.5-3 Master Receive Mode Status

Status Code	I2C bus and hardware status	Application software response					I2C next action performed by the hardware
		R/W I2C_DATA	write I2C_CR				
			STA	STO	SI	AA	
08H	The start condition was sent	Load SLA+R	X	0	0	X	Will send SLA+R, receiveACK
10H	Repeated start condition has been sent	Load SLA+R	X	0	0	X	Will send SLA+R, receiveACK
		Load SLA+W	X	0	0	X	Will send SLA+W, The I2C auto-switch to the master send mode
38H	Arbitration is lost in a non-ACK	None I2C_DATA action	0	0	0	X	The I2C bus will be released; Entry slave mode
		None I2C_DATA action	1	0	0	X	Initiates the start condition when the bus is idle
40H	sent SLA+R , ACK received	None I2C_DATA action	0	0	0	0	Data bytes will be received, Will returnnon-ACK
		None I2C_DATA action	0	0	0	1	Data bytes will be received, Will return ACK
48H	sent SLA+R, non-ACK received	None I2C_DATA action	1	0	0	X	duplicate start condition will send
		None I2C_DATA action	0	1	0	X	will send stop condition and the STO reset
		None I2C_DATA action	1	1	0	X	The stop condition is sent first, followed by the start condition, and the STO reset
50H	Data bytes received, ACK returned	Read data byte	0	0	0	0	Data bytes will be received, Will returnnon-ACK
		Read data byte	0	0	0	1	Data bytes will be received, Will return ACK
58H	Data bytes received, non-ACK returned	Read data byte	1	0	0	X	duplicate start condition will send
		Read data byte	0	1	0	X	will send stop condition and the STO reset

Fig 21.5-3 I2C master receiving status graph



21.5.8 I2C Slave Receive Mode

In slave receive mode, the slave receives data bytes from the master transmitter. The upper 7 bits are the address that the I2C module responds to when addressing by the master. If the LSB (GC) is set, the I2C module will respond to the general call address (0x00); otherwise ignore the general call address.

The setting of the I2C bus speed does not affect the I2C module in slave mode. ENS must be set to enable the I2C module. AA bit must be set to enable the I2C module to acknowledge its own slave address or general call address. STA, STO and SI must be reset. After I2C_ADDR and I2C_CR are initialized, the I2C module waits until it is addressed by the slave address, and then it is addressed by the data direction bit. In order to work in the slave receiving mode, the data direction bit must be "0"

(W). After receiving its own slave address and W bit, the serial interrupt flag (SI) is set, and a valid status code can be read from I2C_SR. This status code is used as a vector for the status service program. The corresponding action for each status code is shown in the table below 21.5-4. If the I2C module loses arbitration in master mode, it can also enter slave receive mode (please refer to the description of status 0x68 and 0x78).

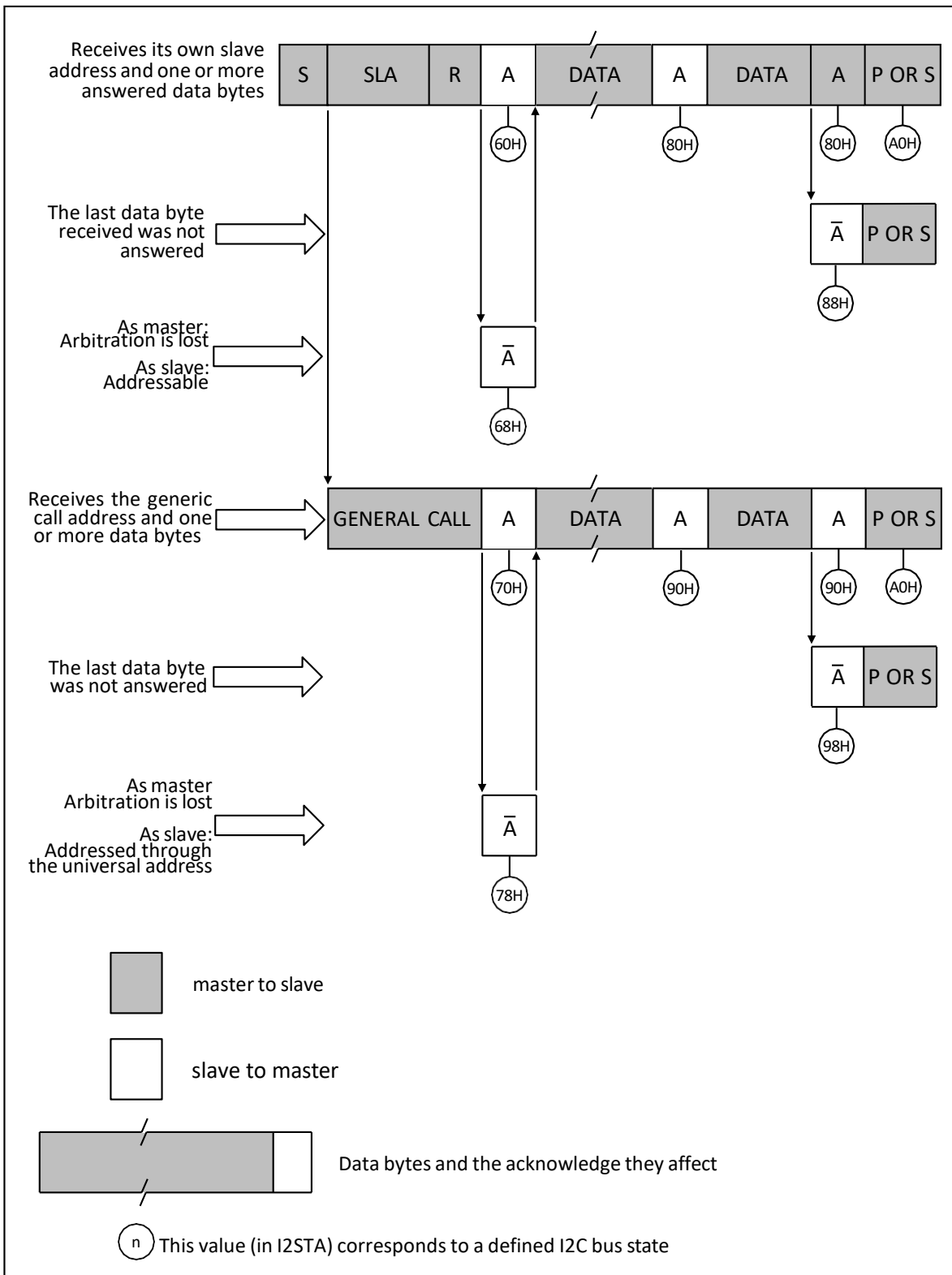
If the AA bit is reset during transmission, the I2C module will return a non-acknowledgement to SDA after receiving the next data byte (logic 1). When AA is reset, the I2C module does not respond to its own slave address or general call address. However, the I2C bus is still monitored and address recognition can be resumed at any time by setting bit AA. This means that the AA bit temporarily detaches the I2C module from the I2C bus.

Tab 21.5-4 I2C Slave Receive Mode Status

Status Code	I2C bus and hardware status	Application software response					I2C next action performed by the hardware
		R/W I2C_DATA	write I2C_CR				
			STA	STO	SI	AA	
60H	Has received its own SLA+W received ACK	None I2C_DATA action	X	0	0	0	Data bytes will be received, Will return non-ACK
		None I2C_DATA action	X	0	0	1	Data bytes will be received, Will return ACK
68H	when in control, Arbitration is lost in SLA+R/W; Has received its own SLA+W; returned ACK;	None I2C_DATA action	X	0	0	0	Data bytes will be received, Will return non-ACK
		None I2C_DATA action	X	0	0	1	Data bytes will be received, Will return ACK
70H	Received pass call address (0x00); returned ACK;	None I2C_DATA action	X	0	0	0	Data bytes will be received, Will return non-ACK
		None I2C_DATA action	X	0	0	1	Data bytes will be received, Will return ACK
78H	when in control, Arbitration is lost in SLA+R/W; Received pass call address (0x00); returned ACK;	None I2C_DATA action	X	0	0	0	Data bytes will be received, Will return non-ACK
		None I2C_DATA action	X	0	0	1	Data bytes will be received, Will return ACK
80H	The previous addressing used the self slave address; Data bytes received; returned ACK;	None I2C_DATA action	X	0	0	0	Data bytes will be received, Will return non-ACK
		None I2C_DATA action	X	1	0	1	Data bytes will be received, Will return ACK
88H	The previous addressing used the self slave address; Data bytes received; returned non-ACK;	Read data byte	0	0	0	0	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address;
		Read data byte	0	0	0	1	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address;
		Read data byte	1	0	0	0	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address
		Read data byte	1	0	0	1	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address; The start condition is sent when the bus is idle;
90H	The previous addressing used the generic call address; Data bytes received; returned ACK	Read data byte	1	0	0	X	Data bytes will be received, Will return non-ACK
		Read data byte	0	1	0	X	Data bytes will be received, Will return ACK

Status Code	I2C bus and hardware status	Application software response					I2C next action performed by the hardware
		R/W I2C_DATA	write I2C_CR				
			STA	STO	SI	AA	
98H	The previous addressing used the generic call address Data bytes received; returned non-ACK;	Read data byte	0	0	0	0	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address;
		Read data byte	0	0	0	1	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address;
		Read data byte	1	0	0	0	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address; The start condition is sent when the bus is idle;
		Read data byte	1	0	0	1	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address; The start condition is sent when the bus is idle;
A0H	The stop condition or restart start condition is received when the enable is addressed statically in slave receive/send mode	None I2C_DATA action	0	0	0	0	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address;
		None I2C_DATA action	0	0	0	1	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address;
		None I2C_DATA action	1	0	0	0	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address; The start condition is sent when the bus is idle;
		None I2C_DATA action	1	0	0	1	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address; The start condition is sent when the bus is idle;

Fig 21.5-4 I2C Slave receive state diagram



21.5.9 I2C Slave Transmitter Mode

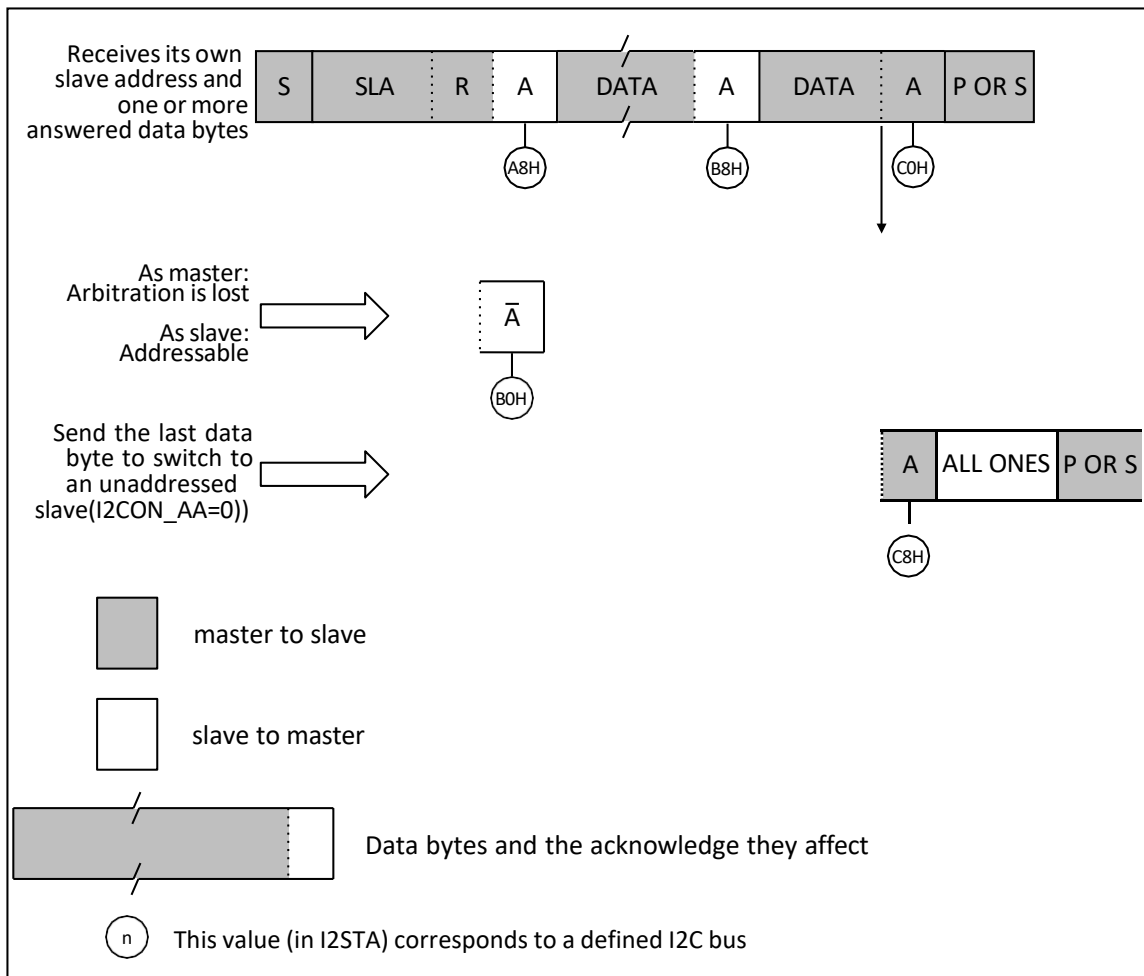
In slave transmit mode, data bytes are transmitted to the master receiver. Data transfers are initiated as in slave receive mode. After initializing I2C_ADDR and I2C_CR, the I2C module waits until it is addressed by its own slave address, followed by the data direction bit, which must be "1" (R), so that the I2C module works in the slave sending mode. After receiving its own slave address and R bit, the serial interrupt flag (SI) is set and a valid status code can be read from I2C_SR. The status code is used as the vector of the status service program, and the corresponding operation of each status code is shown in the table below. If arbitration is lost while the I2C module is in master mode, it can enter slave transmit mode (see state 0xB0) 21.5-5.

If the AA bit is reset during a transfer, the I2C module will send the last byte and go to state 0xC0 or 0xC8. The I2C module switches to "unaddressed" slave mode, and it ignores the master receiver if it continues to transmit. So the master receiver receives all 1's as serial data. When AA is reset, the I2C module does not respond to its own slave address or general call address. However, I2C The bus is still monitored and address recognition can be resumed at any time by setting bit AA. This means that the AA bit can be used to temporarily disconnect the I2C module from the I2C bus

Tab 21.5-5 Slave Transmit Mode Status

Status Code	I2C bus and hardware status	Application software response					I2C next action performed by the hardware
		R/W I2C_DATA	write I2C_CR				
			STA	STO	SI	AA	
A8H	Has received its own SLA+R; returned ACK	Load data byte	X	0	0	0	The last data byte will be sent; Will receive ACK ;
		Load data byte	X	0	0	1	will send 1 byte; Will receive ACK;
B0H	when in control, Arbitration is lost in SLA+R/W; Has received its own SLA+R; returned ACK;	Load data byte	X	0	0	0	The last data byte will be sent; Will receive ACK;
		Load data byte	X	0	0	1	will send 1 byte; Will receive ACK
B8H	Data bytes were sent; ACK received;	Load data byte	X	0	0	0	The last data byte will be sent; receive ACK ;
		Load data byte	X	0	0	1	will send 1 byte; Will receive ACK;
C0H	Data bytes were sent; non-ACK received;	None I2C_DATA action	0	0	0	0	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address
		None I2C_DATA action	0	0	0	1	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address;
		None I2C_DATA action	1	0	0	0	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address; The start condition is sent when the bus is idle;
		None I2C_DATA action	1	0	0	1	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address; The start condition is sent when the bus is idle;
C8H	The loaded data byte has been sent; ACK received;	None I2C_DATA action	0	0	0	0	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address;
		None I2C_DATA action	0	0	0	1	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address;
		None I2C_DATA action	1	0	0	0	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address; The start condition is sent when the bus is idle;
		None I2C_DATA action	1	0	0	1	Switch to non-addressable slave mode; Does not recognize its own secondary or universal address; The start condition is sent when the bus is idle;

Fig 21.5-5 I2C Slave sent state diagram



21.5.10 I2C Miscellaneous Status

Tab 21.5-6 Other

Status Code	I2C bus and hardware status	Application software response				I2C next action performed by the hardware	
		R/W I2C_DATA	write I2C_CR				
			STA	STO	SI		AA
F8H	No relevant status information is available; SI=0;	None I2C_DATA action	None I2C_DATA action				Wait for or execute the current transfer
00H	A bus error will occur on the master or selected slave due to an illegal start or stop condition; Also occurs I2C 0x00 when external interference causes I2C to enter an undefined state	None I2C_DATA action	0	1	0	X	Only in the master or addressable slave mode is the internal hardware affected. Normally, the bus is released and the I2C module switches to non-addressing slave mode. STO reset.

21.5.11 I2C_SR = 0xF8

This status code indicates that no relevant information is available because the serial interrupt flag SI has not been set. This condition occurs between other states and the I2C module has not yet started performing a serial transfer.

21.5.12 I2C_SR = 0x00

This status code indicates that a bus error occurred during an I2C serial transfer. A bus error occurs when a START or STOP condition occurs at an illegal position in a format frame. These illegal locations refer to address bytes, data bytes, or acknowledge bits during serial transfers. Bus errors can also occur when external disturbances affect the internal I2C module signals. SI set when bus error occurs. To recover from the bus error, the STO flag must be set and the SI must be cleared. This causes the I2C module to enter "unaddressed" slave mode (defined state) and clear the STO flag (other bits in I2C_CR are not affected). SDA and SCL lines are released (no stop condition is sent).

21.6 Operating mode**21.6.1 initializer**

Example of using the I2C interface initialization as a slave and/or master.

1. Load its own slave address into I2C_ADDR, enable general call identification (if necessary);
2. Enable I2C interrupt;
3. Write 0x44 to register I2C_CR to set ENS and AA bits and enable slave function. For master function, write 0x40 to register I2C_CR.

21.6.2 Port configuration program

Example of I2C interface signals SCL, SDA mapped to chip pins PB4, PB5.

1. Configure PB4, PB5 as open-drain output mode: P2OD[6], P2OD[5] are configured as 0x1
2. Configure PB4, PB5 function configuration registers: PBAFR4, PBAFR5 are configured as 0x4
3. Configure the pull-up enable configuration registers of PB4 and PB5: PBPPUPD4 and PBPPUD5 are configured as 0x1

21.6.3 Start the master sending function

A master send operation is performed by setting up buffers, pointers, and data counts and then asserting a start condition.

1. Initialize the master data counter;
2. Set up the slave address where the data will be sent, and add the write bit;
3. Write 0x20 to I2C_CR to set the STA bit;
4. Establish the slave address to be sent in the master transmit buffer data;
5. Initialize the master data counter to match the length of the message being sent;
6. Exit.

21.6.4 Start the master receiving function

A master receive operation is performed by setting up buffers, pointers, and data counts and then asserting a start condition.

1. Initialize the master data counter;
2. Set up the slave address where the data will be sent, and add the read bit;
3. Write 0x20 to I2C_CR to set the STA bit;
4. Set up the data to be sent in the master receive buffer;
5. Initialize the master data counter to match the length of the message being sent;
6. Exit.

21.6.5 I2C Interrupt Routine

Determines the status of the I2C and the status routine that handles that status

1. Read the state of I2C from I2C_SR;
2. Use the state value to jump to one of the 26 possible state routines.

21.6.6 no mode specified

- **Status: 0x00** Bus error. Enters "unaddressed" slave mode and releases the bus
 1. Write 0x14 to I2C_CR to set STO and AA bits
 2. Write 0xF7 to I2C_CR to clear SI flag
 3. Exit.
- **master Status** States 08 and 10 apply to master transmit mode and master receive mode. The R/W bit determines whether the next state is in master transmit mode or in master receive mode.
- **Status: 0x08** A start condition has been sent. About to send slave address + R/W bit and receive ACK bit.
 1. Write the slave address and R/W bit to I2C_DATA;
 2. Write 0x04 to I2C_CR to set the AA bit;
 3. Write 0xF7 to I2C_CR to clear the SI flag;
 4. Establish the master transmission mode data buffer;
 5. Establish master receive mode data buffer;
 6. Initialize master data counter;
 7. Exit.
- **Status: 0x10** A repeated start condition has been sent. About to send slave address + R/W bit and receive ACK bit.
 1. Write the slave address and R/W bit to I2C_DATA;
 2. Write 0x04 to I2C_CR to set the AA bit;
 3. Write 0xF7 to I2C_CR to clear the SI flag;
 4. Establish the master transmission mode data buffer;
 5. Establish master receive mode data buffer;
 6. Initialize master data counter;
 7. Exit.

21.6.7 Master sending status

- **Status: 0x18** A previous state of 8 or 10 indicates that the slave address and write bits were sent and an acknowledgment was received. The first data byte is about to be sent and the ACK bit is received.
 1. Load the first data byte of the master transmit buffer into I2C_DATA;
 2. Write 0x04 to I2C_CR to set the AA bit;
 3. Write 0xF7 to I2C_CR to clear the SI flag;
 4. Add 1 to the master send buffer pointer;
 5. Exit.
- **Status: 0x20** Slave address and write operation bit sent and not acknowledge received. A STOP condition is about to be sent
 1. Write 0x14 to I2C_CR to set STO and AA bits;
 2. Write 0xF7 to I2C_CR to clear SI flag;
 3. Exit.
- **Status: 0x28** Data has been sent and ACK has been received. A STOP condition is sent if the last data byte is sent, otherwise the next data byte is sent
 1. The master data counter is decremented by 1, If not the last byte is sent, skip to step 5
 2. Write 0x14 to I2C_CR to set the STO and AA bits;
 3. Write 0xF7 to I2C_CR to clear SI flag;
 4. Exit;
 5. Load the next data byte of the master transmit buffer into I2C_DATA;
 6. Write 0x04 to I2C_CR to set the AA bit;
 7. Write 0xF7 to I2C_CR to clear the SI flag;
 8. The master sends the buffer pointer plus 1;
 9. Exit.
- **Status: 0x30** Data has been sent and a non-acknowledgement has been received. A STOP condition is about to be sent;
 1. Write 0x14 to I2C_CR to set the STO and AA bits;
 2. Write 0xF7 to I2C_CR to clear the SI flag;
 3. Exit
- **Status: 0x38** Arbitration has been lost during the transmission of the slave address and write operation bits or data. The bus has been released and the Enter non-addressed slave mode. A new START condition will be sent when the bus becomes free again
 1. Write 0x24 to I2C_CR to set STA and AA bits;
 2. Write 0xF7 to I2C_CR to clear SI flag
 3. Exit

21.6.8 Master Receive Status

- **Status: 0x40** A preceding state of 08 or 10 indicates that the slave address and read operation bits were sent and ACK was received. will receive data and return ACK.
 1. Write 0x04 to I2C_CR to set AA bit;
 2. Write 0xF7 to I2C_CR to clear SI flag;

3. Exit.
- **Status: 0x48** Slave Address and Read Action bit sent and Not Acknowledged received. A STOP condition will be sent
 1. Write 0x14 to I2C_CR to set STO and AA bits;
 2. Write 0xF7 to I2C_CR to clear SI flag;
 3. Exit.
 - **Status: 0x50** Data has been received and ACK is returned. Data will be read from I2C_DATA. Additional data will be received. If this is the last data byte, return NOT acknowledgment, otherwise return ACK
 1. Read the data byte in I2C_DATA and store it in the master receiving buffer;
 2. Decrement the master data counter by 1, if it is not the last data byte, skip to step 5;
 3. Write 0xF3 to I2C_CR to clear SI flag and AA bit;
 4. Exit;
 5. Write 0x04 to I2C_CR to set AA bit;
 6. Write 0xF7 to I2C_CR to clear SI flag;
 7. master receive buffer pointer plus 1;
 8. Exit.
 - **Status: 0x58** Data has been received, a non-acknowledgement has been returned. Data will be read from I2C_DATA and a STOP condition will be sent
 1. Read the data bytes in I2C_DATA and store them in the master receiving buffer;
 2. Write 0x14 to I2C_CR to set the STO and AA bits;
 3. Write 0xF7 to I2C_CR to clear the SI flag;
 4. Exit.

21.6.9 Slave receiving state

- **Status: 0x60** Own slave address and write operation bits have been received, ACK has been returned. Will receive data and return ACK.
 1. Write 0x04 to I2C_CR to set the AA bit;
 2. Write 0xF7 to I2C_CR to clear the SI flag;
 3. Create slave receive mode data buffer;
 4. Initialize slave data counter;
 5. Exit.
- **Status: 0x68** Arbitration has been lost while transmitting the slave address and R/W bit when acting as a bus master. Own slave address and write operation bits have been received and ACK has been returned. Will receive data and return ACK. Set STA to restart the master mode when the bus is free again mode
 1. Write 0x24 to I2C_CR to set the STA and AA bits;
 2. Write 0xF7 to I2C_CR to clear the SI flag;
 3. Set up slave receive mode data buffer;
 4. Initialize slave data counter;
 5. Exit.
- **Status: 0x70** A generic call has been received and an ACK is returned. Will receive data and

return ACK.

1. Write 0x04 to I2C_CR to set the AA bit;
2. Write 0xF7 to I2C_CR to clear the SI flag;
3. Create slave receive mode data buffer;
4. Initialize slave data counter;
5. Exit

- **Status: 0x78** Arbitration lost while transmitting slave address and R/W bit when acting as a bus master. A generic call has been received and an ACK is returned. Will receive data and return ACK. Set STA to restart master mode when the bus becomes free again.

1. Write 0x24 to I2C_CR to set the STA and AA bits;
2. Write 0xF7 to I2C_CR to clear the SI flag;
3. Create the slave receive mode data buffer;
4. Initialize the slave data counter;
5. Exit.

- **Status: 0x80** Previously addressed its own slave address. Data has been received and ACK is returned. Additional data will be read

1. Read the data bytes of I2C_DATA and store them in the slave receive buffer. The slave data counter is decremented by 1 ,If it is not the last byte, skip to step 5
2. Write 0xF3 to I2C_CR to clear SI flag and AA bit;
3. Exit;
4. Write 0x04 to I2C_CR to set the AA bit;
5. Write 0xF7 to I2C_CR to clear the SI flag;
6. Add 1 to the slave receive buffer pointer;
7. Exit.

- **Status: 0x88** Previously addressed its own slave address. Data has been received and a non-acknowledgement has been returned. Received data will not be saved. Enter "unaddressed" slave mode.

1. Write 0x04 to I2C_CR to set AA bit;
2. Write 0xF7 to I2C_CR to clear SI flag;
3. Exit.

- **Status: 0x90** Before addressing the general call address. Data has been received and ACK is returned. The received data will be saved. Only the first data byte is received and ACK is returned. A non-acknowledgement is returned after receiving additional data bytes.

1. Read the data byte of I2C_DATA and put it into the slave receiver buffer;
2. Write 0xF3 to I2C_CR to clear the SI flag and AA bit;
3. Exit.

- **Status: 0x98** Before addressing the general call address. Data has been received and a non-acknowledgement has been returned. Received data will not be saved. Enter "unaddressed" slave mode

1. Write 0x04 to I2C_CR to set AA bit;
2. Write 0xF7 to I2C_CR to clear SI flag;
3. Exit.

- **Status: 0xA0** A Stop condition or Repeated Start condition has been received, but is still being

addressed as a slave. Received data is not saved. into "unaddressed" slave mode.

1. Write 0x04 to I2C_CR to set AA bit;
2. Write 0xF7 to I2C_CR to clear SI flag;
3. Exit.

21.6.10 Slave sending status

- **Status: 0xA8** Received its own slave address and read operation bit and returned ACK. Data will be sent and ACK bit will be received
 1. Load the first data byte of the slave transmit buffer into I2C_DATA;
 2. Write 0x04 to I2C_CR to set the AA bit;
 3. Write 0xF7 to I2C_CR to clear the SI flag;
 4. Establish slave transmit mode Data buffer;
 5. Add 1 to the slave sending buffer pointer;
 6. Exit.
- **Status: 0xB0** When used as a bus master, mediation is lost when transferring slave addresses and R/W bits. Received its slave address and read operation bit and returned ACK. Data is sent and ACK bits are received. When the bus is idle again, STA is set to restart master mode.
 1. Load the first data byte of slave transmit buffer into I2C_DATA;
 2. Write 0x24 to I2C_CR to set STA and AA bits;
 3. Write 0xF7 to I2C_CR to clear SI flag;
 4. Establish slave send mode data buffer;
 5. Add 1 to the send buffer pointer of the slave;
 6. Exit
- **Status: 0xB8** Data has been sent and ACK has been received. Data will be sent and ACK bit will be received.
 1. Load the data bytes of the slave sending buffer into I2C_DATA;
 2. Write 0x04 to I2C_CR to set the AA bit;
 3. Write 0xF7 to I2C_CR to clear the SI flag;
 4. Add the slave sending buffer pointer to 1;
 5. Exit.
- **Status: 0xC0** Data has been sent and a non-acknowledgement has been received. Enter "unaddressed" slave mode.
 1. Write 0x04 to I2C_CR to set AA bit;
 2. Write 0xF7 to I2C_CR to clear SI flag;
 3. Exit.
- **Status: 0xC8** Last data byte sent and ACK received. Enter "unaddressed" slave mode.
 1. Write 0x04 to I2C_CR to set AA bit;
 2. Write 0xF7 to I2C_CR to clear SI flag;
 3. Exit.

21.7 I2C Registers

Tab 21.7-1 LPUART Registers

offset	Register	Reset value	Description
I2C Base Address: 0x4000_0C00			
0x00	I2C_CR	0x0000_0000	I2C Configuration Register
0x04	I2C_DATA	0x0000_0000	I2C Data Register
0x08	I2C_ADDR	0x0000_0000	I2C Address Register
0x0c	I2C_SR	0x0000_00F8	I2C Status Register
0x10	I2C_TIMRUN	0x0000_0000	I2C Baud Rate Counter Enable
0x14	I2C_BAUDCR	0x0000_0000	I2C Baud Rate Counter Configuration Register

21.8 I2C Register description

21.8.1 I2C Configuration Register(I2C_CR)

Register	Address offset	Access	Reset value	Description
I2C_CR	0x00	RW	0x0000_0000	I2C Configuration Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	ENS	STA	STO	SI	AA	Reserved	H1M

I2C Configuration Register(I2C_CR)

Bit	Access	Description
[31:7]	-	Reserved
[6]	RW	ENS: I2C module was enabled 0:DISABLE 1:ENABLE
[5]	RW	STA: Start flag is enabled 0:DISABLE 1:ENABLE
[4]	RW	STO: Stop flag enabled 0:DISABLE 1:ENABLE
[3]	RW	SI: Interrupt flag bit
[2]	RW	AA: Acknowledge flag is enabled 0:DISABLE 1:ENABLE
[1]	-	Reserved
[0]	RW	H1M: I2C High speed 1 Mbps mode enable 0:DISABLE 1:ENABLE

21.8.2 I2C Data Register(I2C_DATA)

Register	Address offset	Access	Reset value	Description
I2C_DATA	0x04	RW	0x0000_0000	I2C Data Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	I2CDAT[7:0]						

I2C Data Register(I2C_DATA)

Bit	Access	Description
[31:8]	-	Reserved
[7: 0]	RW	I2CDAT[7:0]: I2C Data Register I2C transmit mode, write transmit data to this register. In I2C receive mode, receive data from this register.

21.8.3 I2C Address Register(I2C_ADDR)

Register	Address offset	Access	Reset value	Description
I2C_ADDR	0x08	RW	0x0000_0000	I2C Address Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	I2CADR[6:0]						GC

I2C Address Register(I2C_ADDR)

Bit	Access	Description
[31:8]	-	Reserved
[7: 1]	RW	I2CADR[6:0]: I2C Slave mode address
[0]	RW	GC: Broadcast address acknowledge enable 0: disable 1: Enable

21.8.4 I2C Status Register(I2C_SR)

Register	Address offset	Access	Reset value	Description
I2C_SR	0x0C	R	0x0000_00F8	I2C Status Register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
I2CSTA[7:0]							

I2C Status Register(I2C_SR)

Bit	Access	Description
[31:8]	-	Reserved
[7: 0]	R	I2CSTA[7:0]: I2C Status Register

21.8.5 I2C Baud Rate Counter Enable(I2C_TIMRUN)

Register	Address offset	Access	Reset value	Description
I2C_TIMRUN	0x10	RW	0x0000_0000	I2C Baud Rate Counter Enable

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

I2C Baud Rate Counter Enable(I2C_TIMRUN)

Bit	Access	Description
[31:1]	-	Reserved
[0]	RW	TME: I2C Baud Rate Counter Enable 0: disable 1: Enable

21.8.6 I2C Baud Rate Counter Configuration Register(I2C_BAUDCR)

Register	Address offset	Access	Reset value	Description
I2C_BAUDCR	0x14	RW	0x0000_0000	I2C Baud Rate Counter Configuration Register(I2C_BAUDCR)

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

I2C Baud Rate Counter Configuration Register(I2C_BAUDCR)

Bit	Access	Description
[31:8]	-	Reserved
[0]	RW	TM[7:0] Baud rate counter configuration value Fo=Fi/[8*(N+1)], N=TM, N>0

22 Serial peripheral interface(SPI)

22.1 SPI introduction

The SPI (Serial Peripheral Interface) bus is a synchronous serial peripheral interface, which enables the MCU to exchange information with various peripheral devices in a serial direction. The SPI interface uses 4 lines: serial clock line (SCLK), master output/slave input line (MOSI), master input/slave output line (MISO), active low slave select line (SSN)

22.2 SPI main features

The SPI controller supports the following features:

- Can be configured as a master or a slave through programming
- Full-duplex communication
- 7 baud rates can be configured
- 4-wire transmission mode
- The maximum baud rate of the master mode is 1/2 of the system clock
- The maximum baud rate of the slave mode is 1/4 of the system clock
- Configurable serial clock polarity and phase
- Support interrupt mode
- 8-bit data transmission first transmits the high bit and then the low bit

22.3 SPI functional description

22.3.1 SPI master mode

All data transfers on the SPI bus are initiated by the SPI master, which puts the SPI in master mode by setting the master/slave control bit SPI_CR. MSTR to 1". When SPI is in master mode, enable SPI (set SPI_CR. SPEN to "1") and write a byte to SPI data register SPI_DATA at the same time, data transmission starts. The SPI master device immediately serially shifts out data on MOSI line while providing serial clock on SCK, and the SPI_SR. SPIF interrupt flag is set to "1" after the transfer is completed. If interrupts are enabled, an interrupt request will be generated. In full-duplex applications, while the SPI master is sending data to the slave, the addressed SPI slave can send data to the master on the MISO line. Therefore, the SPIF flag is used both as a sending completion flag and as a ready flag for receiving data. The processor gets the received data by reading the SPI_DATA register.

Operation process:

1. Port configuration: configure the port controller, map the SCK, MISO, MOSI signals to the correct pins, and set them as positive correct input/output status.
2. In master mode, the level of the chip select signal SPI_CS is determined by the value of the register SPI_SSN.SSN
3. SPI baud rate configuration: set SPI_CR.SPR2, SPI_CR.SPR1, SPI_CR.SPR0
4. Serial clock configuration: set SPI clock polarity SPI_CR.CPOL, clock phase SPI_CR.CPHA. See SPI_CR Register for details

5. master configuration: SPI_CR.MSTR = 1
6. SPI enable: SPI_CR.SPEN = 1
7. Slave selection: Configure SPI_SSN.SSN = 0;
8. Start sending data: The data to be sent to the slave is written to the SPI Data Register(SPI_DATA)
9. Wait for the completion of sending/receiving data, and prepare to send/receive the next data The following is the interrupt service routine:
10. Read the SPI_DATA register, that is, receive the data sent from the device.

Note:

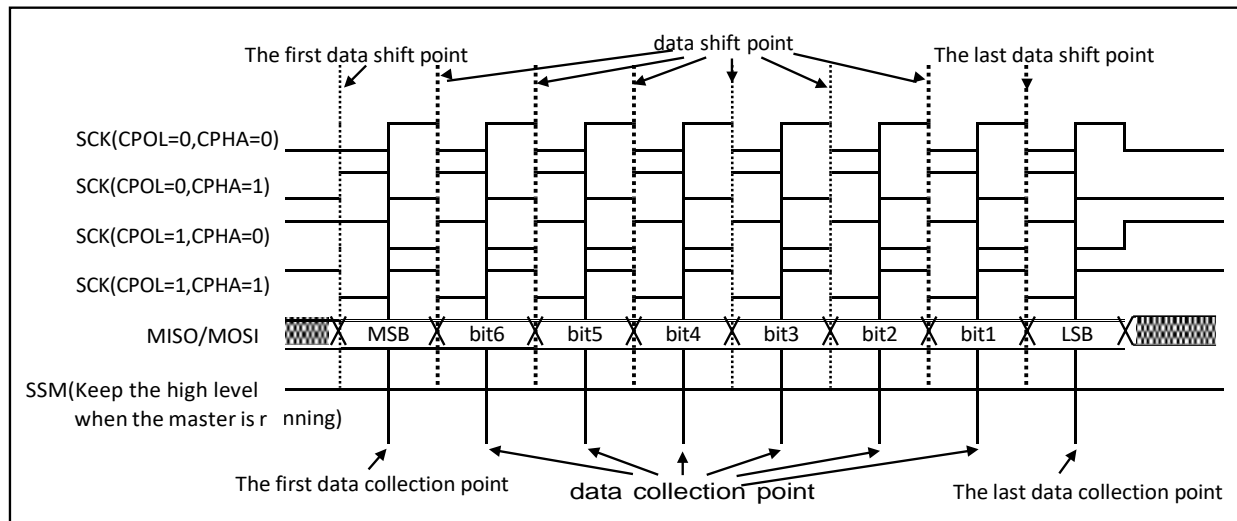
- After step 10 is completed, the SPIF interrupt is cleared to "0"; if you want to send/receive data continuously, repeat steps 8 and 9.
- In multi-machine communication, the SSN pin can be replaced by GPIO.

timing sequence

The serial clock can be selected in 1 of 4 combinations using the clock polarity CPOL and phase CPHA in the SPI Control Register(SPI_CR). SPI_CR.CPOL is whether SCK is high or low when SPI is idle. SPI_CR.CPHA is to select one of two clock phases (edges used to sample data). Master and slave must be configured to use the same clock phase and polarity. The baud rate setting is only valid for the master, and the baud rate setting for the slave will be ignored.

The main mode data/clock timing is as follows:

Fig 22.3-1 master mode data/clock timing diagram

**22.3.2 SPI slave mode**

When the SPI is enabled and not configured as a master, it will operate in SPI slave mode. Put the SPI in slave mode by setting the master/slave control bit SPI_CR.MSTR to "0". When the SPI is in slave mode, the master controls the serial clock(SCK), shifting in data from MOSI. The counter in the SPI logic counts the SCK edges and the SPIF flag is set to "1" when the 8-bit data shift is complete. Received data is obtained by reading SPI_DATA. The slave device cannot start the function of sending data. By writing SPI_DATA to preload the data to be sent to the master device, under the action of the master

device SCK, it moves data bit by bit to the MISO line to start sending to the master device. Operation process :

1. Port configuration: configure the port controller, map the SCK, MISO, MOSI signals to the correct pins, and set them as positive correct input/output status.
2. In slave mode, a GPIO is selected by the port control register as the source of the chip select signal, See 7.2.3 Terminal Control Register (SYSCON_PORTCR)
3. SPI baud rate configuration: set SPI_CR.SPR2, SPI_CR.SPR1, SPI_CR.SPR0
4. Serial clock configuration: set SPI clock polarity SPI_CR.CPOL, clock phase SPI_CR.CPHA. See 22.7.1 SPI Control Register (SPI_CR)
5. Slave mode configuration: SPI_CR.MSTR=0
6. SPI enable: SPI_CR.SPEN=1
7. The data to be sent to the master is written to the SPI Data Register(SPI_DATA)
8. Wait for the completion of sending/receiving data and prepare to send /Receive the next data The following is the interrupt service routine:
9. Read the SPI_DATA register, that is, receive the data sent by the master device

NOTE:

1. *SPIF interrupt is cleared to "0" after step 9 is completed*
2. *When the slave clock phase SPI_CR. CPHA is configured as 0, each time the master pulls down the SPI_CS signal, only one byte of data can be transmitted to the slave. If the master pulls down the SPI_CS signal every time, when it wants to continuously transmit data to the slave, it needs to wait at least 1/2 SCK cycle after the transmission completion interrupt occurs before the next SPI_DATA write operation*
3. *Repeat steps 7 and 8 if you want to send/receive data continuously.*

The slave mode data/clock timing is as follows:

Fig 22.3-2 Slave mode data/clock timing diagram (CPHA=0)

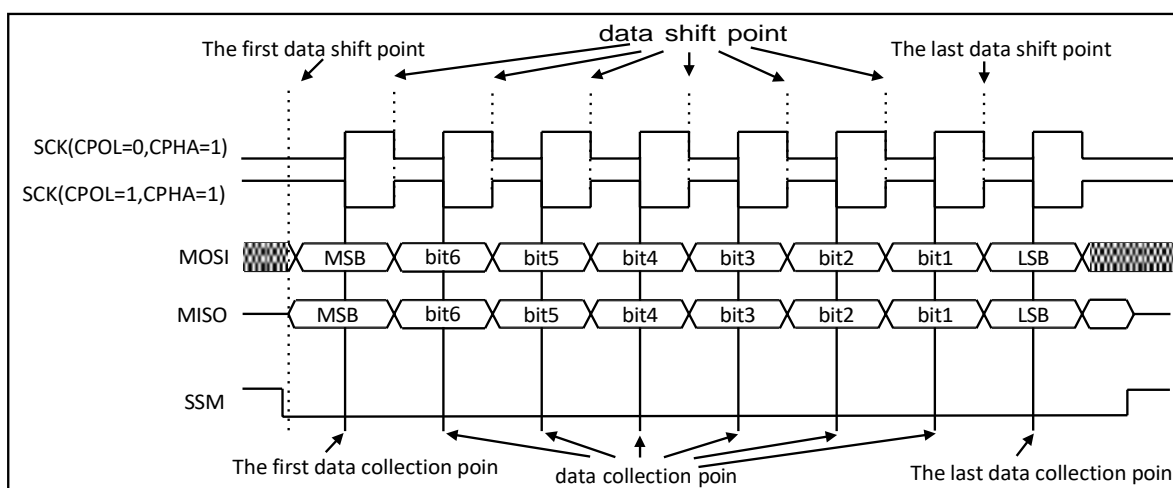
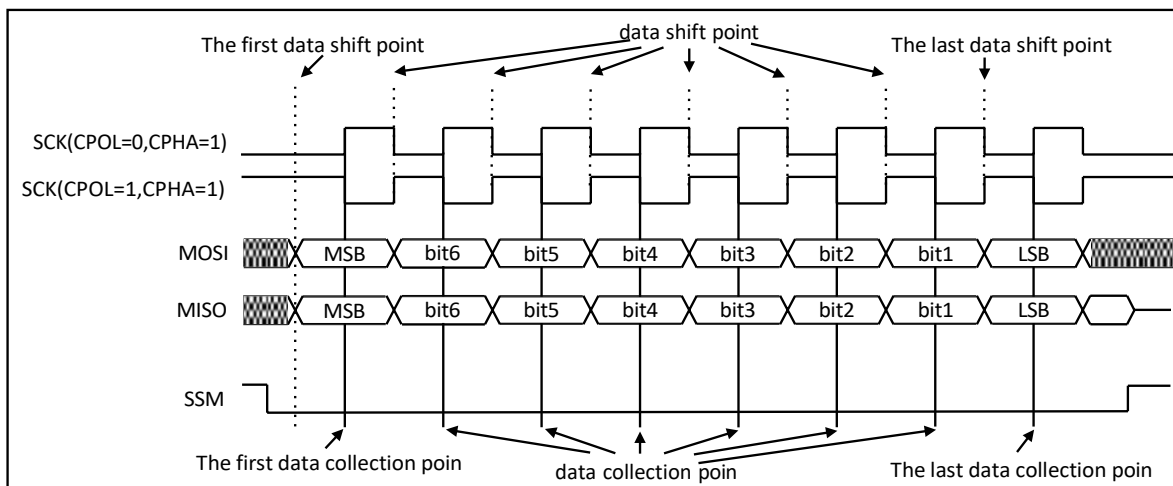


Fig 22.3-3 Slave mode data/clock timing diagram (CPHA=1)



22.4 SPI interrupts

If the SPI interrupt is enabled, an interrupt will be generated when the SPI transfer complete interrupt flag bit SPI_SR.SPIF is set to "1"; or the SPI master mode error interrupt flag bit SPI_SR.MDF is set to "1" and an interrupt will also be generated.

At the end of each byte transfer, the SPI transfer completion interrupt flag bit SPI_SR.SPIF will be automatically set to "1" by hardware.

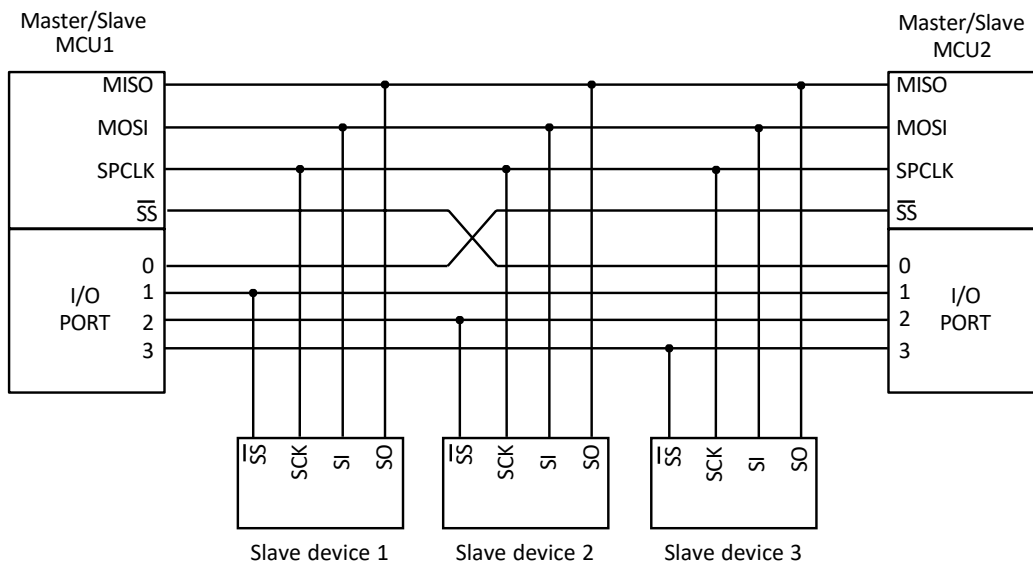
When the SPI is configured as the master mode, the external SSN input is low level. At this time, the SSN input level conflicts with the SPI working mode, and the SPI master mode error interrupt flag SPI_SR.MDF is automatically set to "1" by hardware.

22.5 Multi-Master/Multi-Slave Mode

When this product is used as the master in the SPI single-master single-slave system, the SPI chip select configuration register SPI_SSN can be configured to output high/low level signals to the SPI_CS pin. As a slave, you can configure 7.2.3 Terminal Control Register(SYSCON_PORTCR) to select a GPIO pin as the source of SPI_SSN. However, in a multi-master and multi-slave system, as shown in the figure below 22.5-1, it must be configured according to the corresponding process

When the system is a single master with multiple slaves, the SPI_CS pin can be used as the chip select signal of slave 1, and the chip select signals of other slaves are connected through GPIO pins. When the system is multi-master and multi-slave, all slave chip select signals are connected through GPIO pins, and the master must also be connected with SPI_CS signals of other masters through GPIO pins to monitor whether the bus is occupied.

Fig 22.5-1 Multi-master/multi-slave mode



22.6 SPI Registers

Tab 22.6-1 SPI Registers

offset	Register	Reset value	Description
SPI base address : 0x4000_0800			
0x00	SPI_CR	0x0000_0014	SPI control register
0x04	SPI_SSN	0x0000_0001	SPI slice selection configuration register
0x08	SPI_SR	0x0000_0000	SPI state register
0x0C	SPI_DATA	0x0000_0000	SPI data register

22.7 SPI register description

22.7.1 SPI control register(SPI_CR)

Register	Address offset	Access	Reset value	Description
SPI_CR	0x00	RW	0x0000_0014	SPI control register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
SPR2	SPEN	Reserved	MSTR	CPOL	CPHA	SPR1	SPR0

SPI control register(SPIM_CR)

Bit	Access	Description
[31:8]	-	Reserved
[7]	RW	SPR2: Baud rate selection bit2. refer to 22.7-2
[6]	Rw	SPEN: SPI Module Enable Register 0: disable 1: Enable
[5]	-	Reserved
[4]	RW	MSTR: Master/Slave mode selection 0: Slave 1: master
[3]	RW	CPOL: Clock Polarity Select Register 0: Low 1: high
[2]	RW	CPHA: Clock Phase Select Register 0: First edge 1: Second edge
[1]	RW	SPR1: Baud rate selection bit1, refer to 22.7-2
[0]	RW	SPR0: Baud rate selection bit0, refer to 22.7-2

Tab 22.7-2 Baud rate configuration

SPR2	SPR1	SPR0	SPI_CLK Rate
0	0	0	$F_{sys}/2$
0	0	1	$F_{sys}/4$
0	1	0	$F_{sys}/8$
0	1	1	$F_{sys}/16$
1	0	0	$F_{sys}/32$
1	0	1	$F_{sys}/64$
1	1	0	$F_{sys}/128$

22.7.2 SPI slice selection configuration register (SPI_SSN)

Register	Address offset	Access	Reset value	Description
SPI_SSN	0x04	RW	0x0000_0001	SPI slice selection configuration register

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

SPI slice selection configuration register (SPI_SSN)

Bit	Access	Description
[31:1]	-	Reserved
[0]	RW	SSN: Output value, in master mode, software configuration SSN Value to control SPI_CS Port power Level

22.7.3 SPI state register (SPI_SR)

Register	Address offset	Access	Reset value	Description
SPI_SR	0x08	R	0x0000_0000	SPI state register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
SPIF	WCOL	SSERR	MDF	Reserved			

SPI state register (SPI_SR)

Bit	Access	Description
-----	--------	-------------

[31:8]	-	Reserved
[7]	R	SPIF: transfer end interrupt flag
[6]	R	WCOL: write conflict flag
[5]	R	SSERR: Slave mode SSN Error flag
[4]	R	MDF: master mode error flag
[3:0]	-	Reserved

22.7.4 SPI data register (SPI_DATA)

Register	Address offset	Access	Reset value	Description
SPI_DATA	0x00	RW	0x0000_0000	SPI data register

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

SPI data register (SPI_DATA)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	SPDATA[7:0]: The data register in the sending mode, write the sending value to this register in the receiving mode, Read the received value from this register

23 One-Wire Interface (OWIRE)

23.1 One-Wire Protocol(One-Wire)

The master computer and the slave computer communicate through one wire, and the number of slave devices that can be attached to one bus is almost unlimited.

23.1.1 Features

It uses a single signal line, which can transmit both clock and data, and the data transmission is bidirectional.

23.1.2 Advantage

Single-bus technology has the advantages of simple wiring, less hardware overhead, low cost, and easy bus expansion and maintenance.

23.2 Single bus communication process

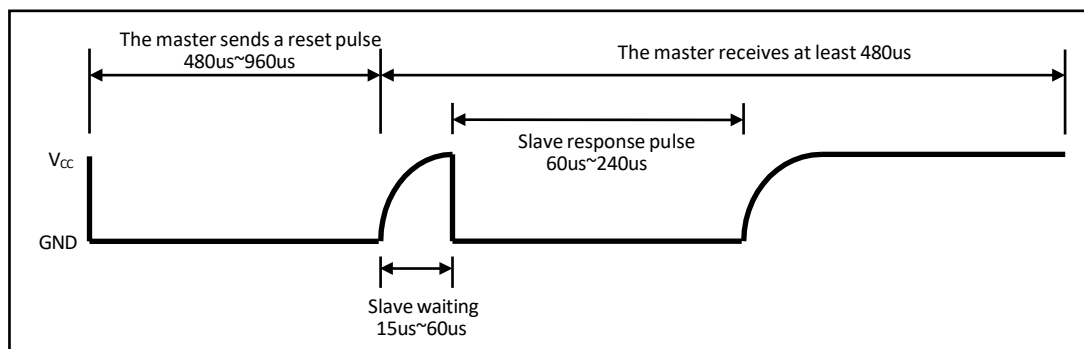
23.2.1 Initialization

Initialization process = reset pulse + slave acknowledge pulse

The master generates a reset pulse by pulling down the single bus for 480 ~ 960 us, and then releases the bus to enter the receiving mode. When the master releases the bus, a rising edge from low level to high level will be generated. After the single-bus device detects the rising edge, there will be a delay of 15 ~ 60us, and the single-bus device will pull down the bus for 60 ~ 240us to generate a response pulse. The master receives the response pulse from the slave, indicating that the single-bus device is ready and the initialization process is completed.

The initialization timing diagram is shown in Figure 23.2-1 :

Fig 23.2-1 Reset and response pulses during initialization

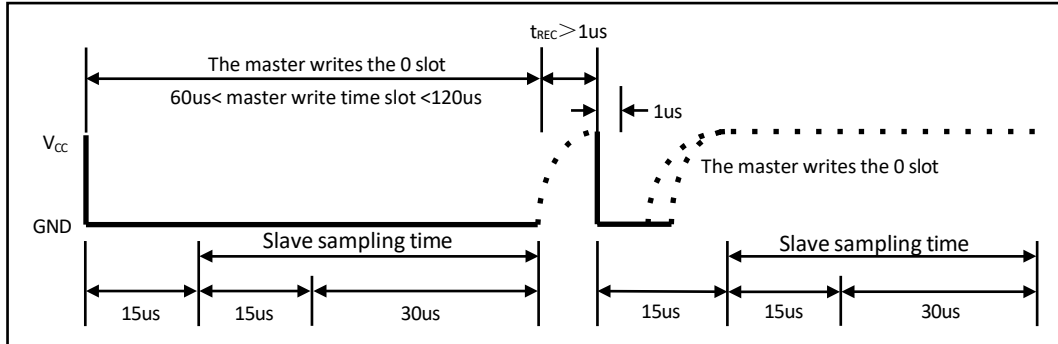


23.2.2 Write gap

Two types of write time slots, including the time slot for writing 0 and the time slot for writing 1. When the data line is pulled low, the data line is sampled within the time window of 15 ~ 60us. If the

data line is low, it is to write 0, and if the data line is high, it is to write 1. To generate a write 1 time gap, the master must pull the data line low, and allow the data line to be pulled high within 15us after the start of the write time gap. To generate a write 0 time gap, the master must pull the data line low and hold for 60us.

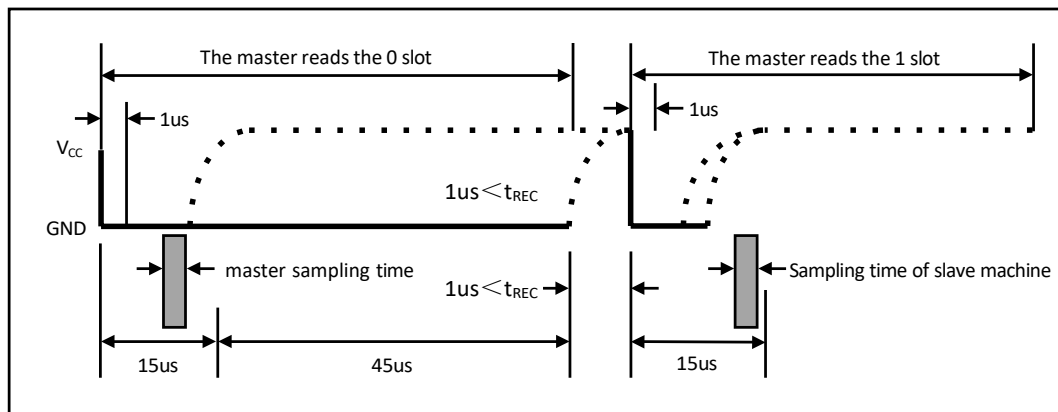
Fig 23.2-2 Write time gap sequence diagram in a single - bus communication protocol



23.2.3 Read gap

When the master pulls the bus low and keeps it for at least 1us and then releases the bus, the data must be read within 15us. The time sequence diagram of read time gap is shown in Figure 23.2-3 :

Fig 23.2-3 Read time gap sequence diagram in a single - bus communication protocol



23.3 Configuration description

23.3.1 Initial configuration instructions

1. Configure the corresponding GPIO pins to multiplex One-Wire pins;
2. Configure the OWIRE_CR.CLKDIV register to set the One-Wire mode clock selection;
3. Configure the One-Wire Reset width control register(OWIRE_RSTCNT) to set the reset time of One-Wire master send (480us ~ 960us);
4. Configure the One-Wire response width count register (OWIRE_PRESCNT), and Set One-Wire slave response setting count value (60us ~ 240us);
5. Configure the One-Wire interrupt enable register (OWIRE_INTEN.INITEN) to enable the initialization completion interrupt;

6. Configure the OWIRE_CR.EN register to enable the One-Wire module;
7. Configure the One-Wire bus operation command register(OWIRE_CMD), set the Initial command;
8. The system enters the interrupt subroutine, configure the status clear register (OWIRE_INTCLR) in the interrupt subroutine, and clear the corresponding interrupt flag;
9. Read data configuration instructions
10. Configure One-Wire Bit rate counter OWIRE_BITRATECNT, set 1 Bit data width (15us ~ 60us);
11. Configure One-Wire master read/write PULL0 drive time OWIRE_DRVCNT, set drive time width (0us ~ 15us);
12. Configure the One-Wire master read sampling time setting OWIRE_RDSMPCNT, set the read sampling time (1us ~ 15us)
13. Configure the one-Wire Recover Time count interval value OWIRE_REC CNT, set the RECOVER time to ($T_{REC} > 1\mu s$);
14. Configure the One-Wire interrupt enable register OWIRE_INTEN.RXDONEEN to enable the receive completion interrupt;
15. Configure the One-Wire bus operation command register (OWIRE_CMD) to set the RX command;
16. The system enters the interrupt subroutine, and the configuration status in the interrupt subroutine is cleared register(OWIRE_INTCLR) , Clear Receive Done Interrupt FLAG and read the One-Wire data register(OWIRE_DATA);

23.3.2 Write data configuration instructions

1 Configure One-Wire Bit rate counter OWIRE_BITRATECNT, set 1 Bit data width (15us ~ 60us); 2. Configure One-Wire master read/write PULL0 drive time OWIRE_DRVCNT, set drive time width (0us ~ 15us); 3 Configure the one-Wire Recover Time count interval value OWIRE_REC CNT, set the RECOVER time to ($T_{REC} > 1\mu s$); 4 Configure the One-Wire interrupt enable register(OWIRE_INTEN.TXDONEEN) to enable the transmission completion interrupt; 5 Write data to the One-Wire data register (OWIRE_DATA); 6. Configure the One-Wire bus operation command register (OWIRE_CMD), and set the TX command; 7. Data After sending, the system enters the interrupt subroutine, and interrupt subroutine configuration status clear register(OWIRE_INTCLR), clear TX complete interrupt FLAG;

23.4 Registers

Tab 23.4-1 OWIRE Registers

offset	Register	Reset value	Description
OWIRE base address : 0x4000_3800			
0x00	OWIRE_CR	0x0000_0000	One-Wire Module Control Register
0x04	OWIRE_NFCR	0x0000_0000	One-Wire Input terminal filter control register
0x08	OWIRE_RSTCNT	0x0000_0000	One-Wire RESET Control Register
0x0C	OWIRE_PRESCNT	0x0000_0000	One-Wire Presence PuLXT counter register
0x10	OWIRE_BITRATECNT	0x0000_0000	One-Wire Bit rate Design counter
0x14	OWIRE_DRVCNT	0x0000_0000	One-Wire master device Read/write time of PULL0 drive
0x18	OWIRE_RDSPMCNT	0x0000_0000	One-Wire Master device read sampling time setting
0x1C	OWIRE_RECNCNT	0x0000_0000	One-Wire Recover Time Counting interval value
0x20	OWIRE_DATA	0x0000_0000	One-Wire Data Register
0x24	OWIRE_CMD	0x0000_0000	One-Wire Bus operation command register
0x28	OWIRE_INTEN	0x0000_0000	One-Wire Interrupt enable
0x2C	OWIRE_SR	0x0000_0000	One-Wire status register
0x30	OWIRE_INTCLR	0x0000_0000	One-Wire Status clear register

23.5 Registers description

23.5.1 One-Wire Module Control Register(OWIRE_CR)

Register	Address offset	Access	Reset value	Description
OWIRE_CR	0x00	RW	0x0000_0000	One-Wire Module Control Register

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

One-Wire Module Control Register(OWIRE_CR)

Bit	Access	Description
[31:8]	-	Reserved
[7]	RW	RDMODE: 0: Normal mode 1: Write 0/Read 0 Equal gap
[6]	RW	MSBFIRST: bit mode setting for byte transmission 0: LSB(bit0) send/receive first 1: MSB(bit7) send/receive first When OWIRE_CR.SIZE=0, this bit should be set to 0

[5]	RW	EN: One-Wiremodule enable control bit 0:One-Wire module stop 1:One-Wire module enable
[4]	RW	SIZE: Data processing bit control bit 0: Single processing 1 bit(Bit mode) 1: Single processing 8 bit(Byte mode)
[3:2]	-	Reserved
[1:0]	RW	CLKDIV[1:0]: Counter clock source selection bit 00: F_{PCLK} 01: $F_{PCLK}/2$ 10: $F_{PCLK}/4$ 11: $F_{PCLK}/16$

23.5.2 One-Wire Input terminal filter control register(OWIRE_NFCR)

Register	Address offset	Access	Reset value	Description
OWIRE_NFCR	0x04	RW	0x0000_0000	One-Wire Input terminal filter control register

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

One-Wire Input terminal filter control register(OWIRE_NFCR)

Bit	Access	Description
[31:5]	-	Reserved
[4]	RW	NFEN: Input terminal filter enable control bit 0: Filter function invalid 1: Filter function valid
[3:2]	-	Reserved
[1:0]	RW	NFDIV[1:0]: 00:F _{PCLK} 01:F _{PCLK} /2 10:F _{PCLK} /4 11:F _{PCLK} /8

23.5.3 One-Wire RESET Control Register(OWIRE_RSTCNT)

Register	Address offset	Access	Reset value	Description
OWIRE_RSTCNT	0x08	RW	0x0000_0000	One-Wire RESET Control register(OWIRE_RSTCNT)

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

One-Wire RESET Control Register(OWIRE_RSTCNT)

Bit	Access	Description
[31:16]	-	Reserved
[15:0]	RW	RSTCNT[15:0]: Master sends reset time setting count value

23.5.4 One-Wire Presence PuLXT counter register(OWIRE_PRESCNT)

Register	Address offset	Access	Reset value	Description
OWIRE_PRESCNT	0x0C	RW	0x0000_0000	One-Wire Presence PuLXT counter register

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

One-Wire Presence PuLXT counter register(OWIRE_PRESCNT)

Bit	Access	Description
[31:13]	-	Reserved
[12:0]	RW	PRESCNT[12:0]: Set count value from response time

23.5.5 One-Wire Bit rate Design counter(OWIRE_BITRATECNT)

Register	Address offset	Access	Reset value	Description
OWIRE_BITRATECNT	0x10	RW	0x0000_0000	One-Wire Bit rate Design counter(OWIRE_BITRATECNT)

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

One-Wire Bit rate Design counter(OWIRE_BITRATECNT)

Bit	Access	Description
[31:12]	-	Reserved
[11:0]	RW	BITRATECNT[11:0]: Bit Rate Time setting count value

23.5.6 One-Wire Master device Read/write time of PULL0 drive (OWIRE_DRVCNT)

Register	Address offset	Access	Reset value	Description
OWIRE_DRVCNT	0x14	RW	0x0000_0000	One-Wire master device Read/write time of PULL0 drive (OWIRE_DRVCNT)

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

One-Wire master device Read/write time of PULL0 drive (OWIRE_DRVCNT)

Bit	Access	Description
[31:9]	-	Reserved
[8:0]	RW	DRVCNT[8:0] : master device read/write PULL0 drive time setting meter value

23.5.7 One-Wire Master device read sampling time setting (OWIRE_RDSMPCNT)

Register	Address offset	Access	Reset value	Description
OWIRE_RDSMPCNT	0x18	RW	0x0000_0000	One-Wire Master device read sampling time setting (OWIRE_RDSMPCNT)

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

One-Wire Master device read sampling time setting (OWIRE_RDSMPCNT)

Bit	Access	Description
[31:9]	-	Reserved
[8:0]	RW	RDSMPCNT[8:0] : master device read sampling time setting meter value

23.5.8 One-Wire Recover Time Counting interval value (OWIRE_RECNT)

Register	Address offset	Access	Reset value	Description
OWIRE_RECNT	0x01C	RW	0x0000_0000	One-Wire Recover Time Counting interval value (OWIRE_RECNT)

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

One-Wire Recover Time Counting interval value (OWIRE_RECNT)

Bit	Access	Description
[31:11]	-	Reserved
[10:0]	RW	RECNT[10:0] : Recover Time counting interval value

23.5.9 One-Wire Data Register (OWIRE_DATA)

Register	Address offset	Access	Reset value	Description
OWIRE_DATA	0x20	RW	0x0000_0000	One-Wire Data Register (OWIRE_DATA)

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

One-Wire Data Register (OWIRE_DATA)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	RW	DRVCNT[7:0]: 1 bit mode(Bit mode): can only send and receive bit 0 8 bit mode(Byte mode): can send and receive all 8 bits

23.5.10 One-Wire Bus operation command register (OWIRE_CMD)

Register	Address offset	Access	Reset value	Description
OWIRE_CMD	0x24	RW	0x0000_0000	One-Wire Bus operation command register (OWIRE_CMD)

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

One-Wire Bus operation command register (OWIRE_CMD)

Bit	Access	Description
[31:2]	-	Reserved
[1:0]	RW	CMD[1:0]: 00: Reserved 01 : Initial 10: TX 11: RX

23.5.11 One-Wire Interrupt enable (OWIRE_INTEN)

Register	Address offset	Access	Reset value	Description
OWIRE_INTEN	0x28	RW	0x0000_0000	One-Wire Interrupt enable (OWIRE_INTEN)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				RXDONEEN	TXDONEEN	INITDONE	ACKERREN

One-Wire Interrupt enable (OWIRE_INTEN)

Bit	Access	Description
[31:4]	-	Reserved
[3]	RW	RXDONEEN: Receive complete interrupt enable 0: disable 1: Enable
[2]	RW	TXDONEEN: Transmit complete interrupt enable 0: disable 1: Enable
[1]	RW	INITEN: Initialization complete interrupt enable 0: disable 1: Enable
[0]	RW	ACKERREN: Slave acknowledge error interrupt enable 0: disable 1: Enable

23.5.12 One-Wire status register(OWIRE_SR)

Register	Address offset	Access	Reset value	Description
OWIRE_SR	0x2C	R	0x0000_0000	One-Wire status register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				RXDONE	TXDONE	INITDONE	ACKERR

One-Wire status register(OWIRE_SR)

Bit	Access	Description
[31:4]	-	Reserved
[3]	R	RXDONE: Receive complete interrupt status 0: Receive completion did not occur 1: Receive complete
[2]	R	TXDONE: Transmit Complete Interrupt Flag 0: Send completion did not occur 1: Send completed
[1]	R	INITDONE: Initialization complete flag 0: Initialization completion did not occur 1: Initialization completed
[0]	R	ACKERR: Slave response error interrupt flag 0: Slave response error did not occur 1: Slave response error occurred

23.5.13 One-Wire Status clear register(OWIRE_INTCLR)

Register	Address offset	Access	Reset value	Description
OWIRE_INTCLR	0x30	W	0x0000_0000	One-Wire Status clear register

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				RXDONECLR	TXDONECLR	INITDONECLR	ACKERRCLR

One-Wire Status clear register(OWIRE_INTCLR)

Bit	Access	Description
[31:4]	-	Reserved
[3]	W	RXDONECLR: Receive complete interrupt clear 0: no effect 1: Clear transmit complete interrupt
[2]	W	TXDONECLR: Transmit Complete Interrupt clear 0: no effect 1: Clear receive completion interrupt
[1]	W	INITDONECLR: Initialization complete flag clear 0: no effect 1: Clear initialization complete interrupt
[0]	W	ACKERRCLR: Slave response error interrupt flag clear 0: no effect 1: Clear slave acknowledge error interrupt

24 Clock Trim/Monitoring module(CLKTRIM)

24.1 CLKTRIM introduction

The CLKTRIM (Clock Trim) module is a circuit specially used to calibrate/monitor the clock. In the calibration mode, select an accurate clock source to calibrate the inaccurate clock source, repeat the calibration, and adjust the parameters of the inaccurate clock source until the frequency of the calibrated clock source meets the accuracy requirements. In the calibration mode, the count value will have a certain error, but it is within the allowable precision error range. In the monitoring mode, select a stable clock source to monitor the system working clock. Under the set monitoring period, monitor whether the system working clock fails and generate an interrupt. In calibration mode and monitor mode, the required clock sources must be initialized and enabled.

24.2 CLKTRIM main features

CLKTRIM supports the following features:

- Calibration Mode
- Monitor Mode
- 32-bit reference clock counter can load initial value
- 32-bit counter to be calibrated can be configured overflow value
- 6 kinds of reference clock sources
- 4 kinds of clock sources to be calibrated
- support interrupt mode

24.3 CLKTRIM functional description

24.3.1 CLKTRIM calibration mode

The calibration mode is mainly used to select an accurate clock source as a reference clock to calibrate an inaccurate clock source to be calibrated. The software repeatedly calibrates according to the following operation process, and adjusts the parameters of the clock source to be calibrated until the clock source to be calibrated meets the frequency accuracy requirements.

1. Set the CLKTRIM_CR.REFCLK_SEL register to select the reference clock.
2. Set the CLKTRIM_CR.CALCLK_SEL register to select the clock to be calibrated.
3. Set CLKTRIM_CR.CLKEN to enable trim and reference clock.
4. Set the CLKTRIM_REFCON.RCNTVAL register to the calibration time.
5. Set the CLKTRIM_CR.IE register to enable the interrupt.
6. Set the CLKTRIM_CR.TRIM_START register to start the trim.
7. The reference clock counter and the clock to be calibrated counter start counting.
8. When the reference clock counter counts down from the initial value to 0, CLKTRIM_IFR.STOP is set to 1 and an interrupt is triggered.
9. The interrupt service subroutine judges that CLKTRIM_IFR.STOP is 1, reads the values of the registers CLKTRIM_REFCNT and CLKTRIM_CALCNT, and clears the CLKTRIM_CR.TRIM_START register to

end the calibration.

Note: In calibration mode, the clock counter to be calibrated may overflow before CLKTRIM_IFR.STOP is set to 1 because the calibration time is set too long. CLKTRIM_IFR.CALCNT_OVF sets 1 to trigger an interrupt.

When the interrupt service subroutine finds that CLKTRIM_IFR.CALCNT_OVF is set to 1, clear the CLKTRIM_CR.TRIM_START register to end the calibration. In this case, the calibration cannot be performed correctly, and the calibration time must be adjusted and re-calibrated. The specific steps are:

1. Set the CLKTRIM_REFCON.RCNTVAL register to adjust the calibration time.
2. Set the CLKTRIM_CR.TRIM_START register to restart the calibration.

24.3.2 CLKTRIM monitoring mode

The monitoring mode is mainly used to select a stable clock source as the reference clock, and monitor the abnormal state of the system working clock in the set time period. In monitoring mode, only external HSE clock or external LSE clock can be selected as the monitored clock.

Operation flow:

1. Set the CLKTRIM_CR.REFCLK_SEL register to select the reference clock.
2. Set the CLKTRIM_CR.CALCLK_SEL register to select the monitored clock.
3. Set CLKTRIM_CR.CLKEN to enable the monitored and reference clock.
4. Set the CLKTRIM_REFCON.RCNTVAL register to monitor interval time.
5. Set the CLKTRIM_CALCON.CALOVCNT register to the monitored clock counter overflow time.
6. Set the CLKTRIM_CR.MON_EN register to enable the monitor function.
7. Set the CLKTRIM_CR.IE register to enable the interrupt.
8. Set the CLKTRIM_CR.TRIM_START register to start monitoring.
9. The reference clock counter and the monitored clock counter start counting.
10. When the counting of the reference clock counter reaches the monitoring interval time, judge whether the monitored clock counter overflows. If overflow Indicates that the monitored clock works normally. If there is no overflow, it means that the monitored clock fails, CLKTRIM_IFR.HSE_FAULT or CLKTRIM_IFR.LSE_FAULT is set to 1, and an interrupt is triggered.
11. If RCC_SYSCSKCR.CLKFAILEN is set to 1, after an interrupt occurs, it will automatically switch the system clock source to the internal high-speed RC clock (HSI), and process the interrupt service subroutine. Clear the interrupt flag bit CLKTRIM_IFR.HSE_FAULT or CLKTRIM_IFR.LSE_FAULT, clear the CLKTRIM_CR.TRIM_START register to end monitoring.

24.4 CLKTRIM registers

Tab 24.4-1 CLKTRIM Registers

offset	Register	Reset value	Description
CLKTRIM base address : 0x4000_3400			
0x00	CLKTRIM_CR	0x0000_0000	Configuration Register
0x04	CLKTRIM_REFCON	0xFFFF_FFFF	Reference Counter Disposition Configuration Register
0x08	CLKTRIM_REFCNT	0xFFFF_FFFF	Reference Counter Value Register
0x0C	CLKTRIM_CALCNT	0x0000_0000	Calibration Counter Value Register
0x10	CLKTRIM_IFR	0x0000_0000	Interrupt Flag Register
0x14	CLKTRIM_ICLR	0x0000_0000	Interrupt Flag Clear Register
0x18	CLKTRIM_CALCON	0x0000_FFFF	Calibration Counter Overflow Value Configuration Register

24.5 CLKTRIM registers description

24.5.1 Configuration Register(CLKTRIM_CR)

Register	Address offset	Access	Reset value	Description
CLKTRIM_CR	0x00	RW	0x0000_0000	Configuration Register(CLKTRIM_CR)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							CLKEN
7	6	5	4	3	2	1	0
IE	MON_EN	CALCLK_SEL[1:0]		REFCLK_SEL[2:0]			TRIM_START

1-Wire Module Control Register(OWIRE_CR)

Bit	Access	Description
[31:9]	-	Reserved
[8]	RW	CLKEN: Reference clock and calibration clock enable 0: disable 1: Clock enable
[7]	RW	IE: interrupt enable register 0: disable 1: Enable
[6]	RW	MON_EN: Monitor Mode Enable Register 0: disable 1: Enable
[5:4]	RW	CALCLK_SEL[1:0]: To be calibrated/Monitoring clock selection register 00:HSI 01:HSE 10:LSI 11:LSE

[3:1]	RW	REFCLK_SEL[2:0]: Reference clock select register 000:HSI 001:HSE 010:LSI 011:LSE 100:HSE bypass clock
[0]	RW	TRIM_START: Calibration/Monitoring start register 0: Stop 1: Start

24.5.2 Reference Counter Disposition Configuration Register(CLKTRIM_REFCON)

Register	Address offset	Access	Reset value	Description
CLKTRIM_REFCON	0x04	RW	0xFFFF_FFF F	Reference Counter Disposition Configuration Register(CLKTRIM_REFCON)

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

Reference Counter Disposition Configuration Register(CLKTRIM_REFCON)

Bit	Access	Description
[31:0]	RW	RCNTVAL[31:0]: Reference counter initial value

24.5.3 Reference Counter Value Register(CLKTRIM_REFCNT)

Register	Address offset	Access	Reset value	Description
CLKTRIM_REFCNT	0x08	R	0xFFFF_FFF F	Reference counter value register(CLKTRIM_REFCNT)

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

Reference counter value register(CLKTRIM_REFCNT)

Bit	Access	Description
[31:0]	R	REFCNT[31:0]: Reference counter value To read this register, you need to enable the clock first. When TRIM_START is valid, the initial value written will be updated to this register

24.5.4 Calibration Counter Value Register(CLKTRIM_CALCNT)

Register	Address offset	Access	Reset value	Description
CLKTRIM_CALCNT	0x0C	R	0x0000_0000	Calibration Counter Value Register(CLKTRIM_CALCNT)

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

Calibration Counter Value Register(CLKTRIM_CALCNT)

Bit	Access	Description
[31:0]	R	CALCNT[31:0]: Calibration counter value

24.5.5 Interrupt Flag Register(CLKTRIM_IFR)

Register	Address offset	Access	Reset value	Description
CLKTRIM_IFR	0x10	R	0x0000_0000	Interrupt Flag Register(CLKTRIM_IFR)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				HSE_FAULT	LSE_FAULT	CALCNT_OVF	STOP

Interrupt Flag Register(CLKTRIM_IFR)

Bit	Access	Description
[31:4]	-	Reserved
[3]	R	HSE_FAULT : HSE Fail flag 1:HSE invalid 0:HSE effective
[2]	R	LSE_FAULT : LSE Fail Flag 1:LSE invalid 0:LSE effective
[2]	R	CALCNT_OVF :Calibration counter overflow flag CLKTRIM_CR.TRIM_START Write 0 to clear this flag
[0]	R	STOP : Reference counter stop flag CLKTRIM_CR.TRIM_START Write 0 to clear this flag

24.5.6 Interrupt Flag Clear Register(CLKTRIM_ICLR)

Register	Address offset	Access	Reset value	Description
CLKTRIM_ICLR	0x14	W	0x0000_0000	Interrupt Flag Clear Register(CLKTRIM_ICLR)

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

Interrupt Flag Clear Register(CLKTRIM_ICLR)

Bit	Access	Description
[31:4]	-	Reserved
[3]	W	HSE_FAULT_CLR : clear HSE invalid flag, write 1 clear
[2]	W	LSE_FAULT_CLR : clear LSE invalid flag, write 1 clear
[1:0]	-	Reserved

24.5.7 Calibration Counter Overflow Value Configuration Register(CLKTRIM_CALCON)

Register	Address offset	Access	Reset value	Description
CLKTRIM_CALCON	0x18	W	0x0000_FFFF	Calibration Counter Overflow Value Configuration Register(CLKTRIM_CALCON)

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

Calibration Counter Overflow Value Configuration Register(CLKTRIM_CALCON)

Bit	Access	Description
[31:16]	-	Reserved
[15: 0]	RW	CALOVCNT[15 : 0] : Calibration counter overflow value. In monitor mode, if the monitored clock counts to the CALOVCNT value during the monitor period , it means that the monitored clock is working normally; if it is less than the set value, it means that the monitored clock stops, and the disabled bit of the monitored clock will be set.

25 Real-time clock (RTC)

25.1 RTC introduction

The Real Time Clock (RTC) is an independent BCD timer/counter that provides seconds, minutes, hours (12/24 hour format), week, day, month and year information.

The RTC module has an automatic wake-up function to manage all low-power modes.

Two 32-bit registers store seconds, minutes, hours (12/24 hour clock), week, day, month, and year in BCD format.

The RTC has an automatic month day compensation function, and the number of days in a month and the number of days in a leap year can be automatically adjusted.

Two 32-bit registers are used to store programmable alarm information including seconds, minutes, hours, weeks, days, months, and years.

For any error caused by the frequency deviation, temperature drift and other reasons of the crystal itself, the digital calibration function of the RTC can be used to correct it.

After a power-on reset, all RTC registers are disabled to prevent accidental writes. After a power-on reset, all RTC registers are disabled to prevent accidental writes.

25.2 RTC main features

- Calendar function showing seconds, minutes, hours(12/24 hours), week, day, month and year.
- Automatic leap year adjustment is possible.
- It has functions of alarm interrupt and period interrupt.
- Digital calibration circuit (regular correction of the counter): from a calibration window of a few seconds.
- Clock source optional: external low-speed clock (LSE), internal low-speed clock (LSI), external high-speed clock (HSE)
- 1Hz square wave output

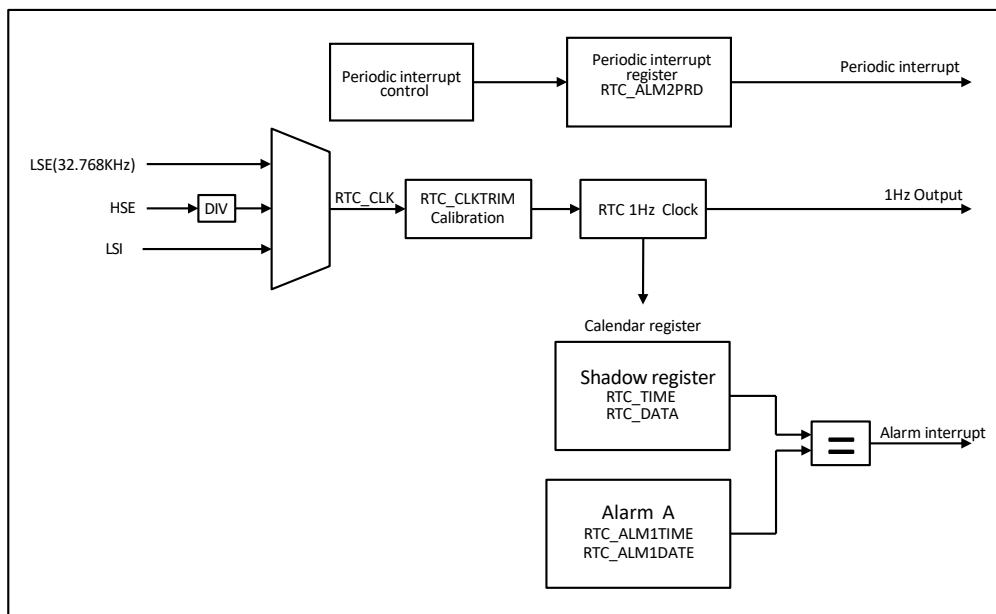
25.3 RTC functional description

25.3.1 RTC block diagram

RTC modules include:

- An alarm clock alarm interruption
- One-period interrupt
- calibrated 1Hz clock output perpetual
- calendar register

Fig 25.3-1 RTC Block Diagram



25.3.2 RTC clock

The RTC clock source (RTC_CLK) is selected by the clock controller from the following 3 clocks:

1. LSE clock as RTC clock;
2. LSI clock as RTC clock;
3. HSE clock as RTC clock.

25.3.3 Reset process

Any available system reset source will cause the calendar shadow register and the RTC initialization and status register (RTC_ISR) to be reset to default values. However, the reset of the following registers has nothing to do with a system reset, but only with a power-on reset (POR reset):

RTC control register register (RTC_CR), RTC clock control register (RTC_CLKCR), RTC trim register (RTC_CLKTRIM), ALM register (RTC_ALM1DATE/RTC_ALM1TIME/RTC_ALM2PRD), RTC current calendar register

The RTC will remain running on any system reset except power-on reset. After a power-on reset occurs, the RTC stops running and all RTC registers are reset to default values.

25.3.4 Register write protection

After power-on reset, all RTC registers will be write-protected. Write access to the RTC register is enabled by writing the specified keyword to the write protection register RTC_WPR.

Release the write protection of all RTC registers by the following operations.

1. Write '0xCA' to the RTC_WPR register;
2. Write '0x53' to the RTC_WPR register.

Note: After the protection is released, any further write to this register will reactivate the write protection.

25.3.5 Calendar initialization and configuration

Initialization of the time and date values, including configuration of the time format and prescaler, is done in the following sequence:

1. Select the RTC timing clock source through `RTC_CLKCR.CKSEL`. If you choose the HSE clock, you must first set the prescaler.
2. `RTC_CLKCR.RTCCKEN` to enable RTC timing clock.
3. Set the `WAIT` bit of the `RTC_ISR` register to "1" to enter initialization mode.
4. Wait for the `WAITF` bit of the `RTC_ISR` register to be "1" to ensure that the initialization mode has been formally entered. Due to clock synchronization delay, this process takes about 2 `RTC_CLK` clock cycles. In this mode the calendar counter is suspended and the time and date counters can be updated.
5. Set the time format (12-hour/24-hour) by the `FMT` bit of the `RTC_CR` register.
6. Load the initial time and date values into the time registers (`RTC_TIME` and `RTC_DATE`).
7. When clock error compensation is required, set the clock compensation register `RTC_CLKTRIM`.
8. Clear the value of the `RTC_ISR.WAIT` bit to exit initialization mode. The actual value of the calendar counter will be loaded automatically and restart after 4 `RTCCLK` clock cycles.

After the above series of initialization operations are completed, the calendar will start timing.

25.3.6 Read count register

When the `BYPSHAD` control bit of the `RTC_CR` register is cleared:

To ensure that the RTC calendar registers (`RTC_TIME` and `RTC_DATE`) can be read normally under the safe synchronization mechanism, the APB clock frequency (`fPCLK`) should be at least 7 times the RTC clock frequency (`fRTCCLK`). When the APB clock frequency is lower than 7 times the RTC clock frequency, the software must read the calendar time and date registers twice. If the value read for the second time is the same as the value read for the first time, indicates the returned value is correct. Otherwise, read the value again. In any case, the APB clock frequency must be greater than the RTC clock frequency. When the contents of the calendar register are copied into the `RTC_TIME` and `RTC_DATE` shadow registers, the `RSF` bit of the `RTC_ISR` register is set. The copy operation is performed every two `RTC_CLK` cycles. The copy operation is performed every two `RTC_CLK` cycles. To ensure that the values are the same, the hardware locks the value of the `RTC_DATE` shadow register when `RTC_TIME` is read until the value of `RTC_DATE` is read.

To avoid software accessing the calendar multiple times when the time interval is less than 2 `RTC_CLK` cycles, the `RSF` bit should be cleared by software every time the calendar is read, and the software must wait for the `RSF` bit to be set before reading the `RTC_TIME` and `RTC_DATE` registers. After waking up from low-power mode, the `RSF` bit should be cleared by software, software must wait for the `RSF` bit to be set again before reading `RTC_TIME` and `RTC_DATE` register. The `RSF` bit should be cleared after wake-up, not before entering low-power mode. After a system reset, software must wait for the `RSF` bit to be set before reading the `RTC_TIME` and `RTC_DATE` registers. In fact a system reset will cause the shadow registers to reset to their default values.

When the BYPSHAD control bit of the RTC_CR register is set (no need to consider the shadow register): Read the calendar register to get the value directly from the calendar counter without waiting for the RSF bit to be set. This feature is useful right after exiting low-power mode because shadow registers are not automatically updated in low-power mode.

When BYPSHAD is "1", if an RTC_CLK edge occurs between two register reads, the results in different registers may be independent of each other. Also, if an RTC_CLK edge is encountered during a read operation, the value of a register may be incorrect. Software must read all registers twice and compare the results of the two reads; or by comparing the results of two sets of least significant calendar registers to check whether the data is correct and relevant.

25.3.7 Write count register

1. Set RTC_ISR.WAIT = 1, stop calendar register counting, and enter write mode;
2. Query until RTC_ISR.WAITF=1;
3. Write seconds, minutes, hours, weeks, days, months, year counting register values;
4. Set RTC_ISR.WAIT=0, the counter restarts. Note that all write operations must be completed within 1 second;
5. Query until RTC_ISR.WAITF = 0.

25.3.8 Alarm clock setting

Set the ALM1EN bit of the RTC_CR register to start the alarm clock function. When the second, minute, hour, week, day, month, and year in the calendar match the values set in the alarm registers RTC_ALM1TIME and RTC_ALM1DATE, RTC_ISR.ALM1_F is set to 1 by hardware. All calendar fields can be selected as alarm sources by the ALMxEN bits in the RTC_ALM1DATE register. Setting the ALM1_INTEN bit in the RTC_CR register will generate an alarm interrupt.

25.3.9 Calibrated 1Hz output

The RTC can optionally output a calibrated 1Hz clock. Set the output enable through RTC_CR.RTC1HZOE, through RTC_CLKTRIM to set the calibration value

25.3.10 RTC clock calibration

The RTC module compensates the frequency of the RTC clock by masking the specified number of RTC clock cycles every fixed time period, use RTC_CLKTRIM.MODE[1:0] to select the adjustment interval: 0b00: calibrate every 60 seconds (SEC=00)

0b01: Calibrate every 30 seconds (SEC=00, 30)

0b10: Calibrate every 15 seconds (SEC=00, 15, 30, 45)

0b11: Calibrate every 6 seconds (SEC=00, 06, 12, 18, 24, 30, 36, 42, 48, 54) The masked RTC clock cycle number is specified by RTC_CLKTRIM.TRIM[7:0].

Note: that the value of RTC_CLKTRIM.TRIM[7:0] is a signed integer with a range of -128 ~ +127.

25.4 RTC interrupt

The RTC supports two types of interrupts. Alarm clock interruption, regular period interruption. The alarm clock interrupt and the periodic interrupt share the same interrupt signal, and the interrupt source is distinguished by the flag register bit.

25.4.1 RTC alarm interrupt

1. Set `RTC_CR.ALM1EN = 0` to disable the alarm function;
2. Set the time alarm register `RTC_ALM1TIME` and date alarm register `RTC_ALM1DATE`;
3. Set `RTC_CR.ALM1EN=1` to enable the alarm clock;
4. Clear the interrupt flag bit `RTC_ISR.ALM1_F` ;
5. Set `RTC_CR.ALM1_INTEN = 1`, the alarm clock interrupt is allowed, if the current calendar time is equal to the alarm clock register, the alarm clock will be triggered broken
6. Wait for an interrupt to occur;

25.4.2 RTC cycle interrupt

When `RTC_CR.ALM2_INTEN = 1` in the control register `RTC_CR`, after the selected cycle occurs, the fixed cycle wake-up interrupt is triggered. Since the alarm clock and the fixed cycle share the interrupt, it is distinguished by the flag register bit.

1. Set `RTC_CR.ALM2_INTEN=0`, disable the periodic interrupt function;
2. Set the periodic alarm register `RTC_ALM2PRD`;
3. Clear the interrupt flag `RTC_ISR.ALM2_F`;
4. Set `RTC_CR.ALM2_INTEN=1`, the periodic interrupt is enabled, selected After the cycle occurs, trigger a periodic wake-up interrupt;
5. Wait for the interrupt to occur;

25.5 RTC Registers

Tab 25.5-1 RTC Registers

offset	Register	Reset value	Description
RTC base address: 0x4000_3000			
0x00	RTC_CR	0x0000_0000	RTC Control Register(RTC_CR)
0x04	RTC_CLKCR	0x0000_0000	RTC Clock Control Register(RTC_CLKCR)
0x08	RTC_TIME	0x0000_0000	RTC Time register(RTC_TIME)
0x0C	RTC_DATE	0x0000_0000	RTC Date Register (RTC_DATE)
0x10	RTC_ALM1TIME	0x0000_0000	RTC Time Alarm Register(RTC_ALM1TIME)
0x14	RTC_ALM1DATE	0x0000_0000	RTC Date Alarm Register (RTC_ALM1DATE)
0x18	RTC_ALM2PRD	0x0000_0000	RTC Periodic Alarm Clock Register (RTC_ALM2PRD)
0x1C	RTC_CLKTRIM	0x0000_0000	RTC Clock Trim Register (RTC_CLKTRIM)
0x20	RTC_ISR	0x0000_0000	RTC Initialize and Status Registers (RTC_ISR)
0x24	RTC_INTCLR	0x0000_0000	RTC Status Clear Register(RTC_INTCLR)
0x28	RTC_WPR	0x0000_0000	RTC Write Protection Register(RTC_WPR)

25.6 RTC Registers Description

25.6.1 RTC Control Register(RTC_CR)

Register	Address offset	Access	Reset value	Description
RTC_CR	0x00	RW	0x0000_0000	RTC Control Register(RTC_CR)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							START
7	6	5	4	3	2	1	0
Reserved	ALM1EN	ALM2_INTEN	ALM1_INTEN	Reserved	FMT	RTC1HZOE	BYPSHAD

RTC Control Register(RTC_CR)

Bit	Access	Description
[31:9]	-	Reserved
[8]	RW	START: 0: Stop RTC Counter 1: Enable RTC Counter
[7]	-	Reserved
[6]	RW	ALM1EN: ALM1 Alarm function enable 0: Disable ALM1 Alarm function 1: Enable ALM1 Alarm function Note: During calendar(RTC_CLKCR.RTCCKEN=1) counting and with the ALM1 alarm interrupt permission enabled , (ALM1_INTEN=1)When enabling ALM1EN, to prevent malfunction, please disable the system interrupt. Afteris enabled, please clear ALM1_F flag bit.
[5]	RW	ALM2_INTEN : ALM2 Periodic interrupt enable 0: Disable ALM2 Periodic interrupt 1: Enable ALM2 Periodic interrupt
[4]	RW	ALM1_INTEN : ALM1 Alarm interrupt enable 0: Disable ALM1 Alarm Interrupt 1: Enable ALM1 Alarm Interrupt
[3]	-	Reserved
[2]	RW	FMT : Time format. 0:12 hour format(AM/PM time format) 1:24 hour format
[1]	RW	RTC1HZOE : RTC 1Hz output enable. 0: disable 1: Enable

[0]	RW	<p>BYPSHAD : Bypass shadow registers</p> <p>0: read calendar value from shadow register, shadow register every two RTC_CLK Periodic update 1: read the calendar value directly from the calendar counter</p> <p>Note: If the APB clock frequency is less than 7 times the RTCCLK frequency, BYPSHAD must be set to "1"</p>
-----	----	--

25.6.2 RTC Clock Control Register(RTC_CLKCR)

Register	Address offset	Access	Reset value	Description
RTC_CLKCR	0x04	RW	0x0000_0000	RTC Clock Control Register(RTC_CLKCR)

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

RTC Clock Control Register(RTC_CLKCR)

Bit	Access	Description
[31:21]	-	Reserved
[20]	RW	<p>START: RTC Counting clock enable Set 1 or clear 0 by software</p> <p>Write: 0:RTC clock off 1:RTC clock on</p> <p>Read: 0:RTC clock off 1:RTC clock on</p>
[19:18]	-	Reserved
[17:16]	RW	<p>RTCKSEL[1:0]: RTC Clock source selection</p> <p>Select RTC clock source by software setting. Once RTC clock source is selected, these bit values cannot be changed unless RTC is reset. The RTC field can be reset by setting the RCC_RTCRST.RTCRST bit.</p> <p>00:LSE oscillator as RTC clock 01:LSI oscillator as RTC clock 10:FHSE/(HSEDIV[9:0]) 11: Reserved</p>
[15:10]	-	Reserved
[9:0]	RW	<p>HSEDIV[9:0]: External high-speed crystal oscillator clock frequency division</p> <p>0: Stop others:F= F_{HSE}/(HSEDIV[9:0])</p>

25.6.3 RTC Time Register(RTC_TIME)

Register	Address offset	Access	Reset value	Description
RTC_TIME	0x08	RW	0x0000_0000	RTC Time register(RTC_TIME)

31	30	29	28	27	26	25	24
Reserved					WEEK[2:0]		
23	22	21	20	19	18	17	16
Reserved		H20_PA	HOUR19[4:0]				
15	14	13	12	11	10	9	8
Reserved	MIN[6:0]						
7	6	5	4	3	2	1	0
Reserved	SEC[6:0]						

RTC Time register(RTC_TIME)

Bit	Access	Description
[31:27]	-	Reserved
[26:24]	RW	WEEK[2:0]: Week counter. week counter value is binary counting, the counting interval is from 0 to 6 (7 is not used unless week count is not used). The correspondence between the week and the week counter value is defined by the user. (such as Sunday= 0, Monday= 1 ...Saturday= 6)
[23:22]	-	Reserved
[21]	RW	H20_PA : Hour counter. The time format is determined by the clock system. 12 hour format : H20_PA means morning or afternoon.when counts from[1,11](11PM) to [0,12](12AM), the date counter increases by one day 24 hour format : H20_PA determines whether the ten digit of the counter is 2. when counts from[1,3](23H) to [0,0](0H), the date counter increases by one day
[20: 16]	RW	HOUR19[4:0]: Hour counter. The value of HOUR19 is BCD coded. The time format is determined by the clock system. 12 hour format : when counts from[1,11](11PM) to [0,12](12AM), the date counter increases by one day. 24 hour format : when counts from[1,3](23H) to [0,0](0H), the date counter increases by one day
[15]	-	Reserved
[14:8]	RW	MIN[6:0] : Minute counter. Minute counter value BCD coded, counting range from 0 to 59. When the minute counter counts from 59 to 0, the hour counter increments by 1. When the counter is written, the time less than one second will be ignored.
[7]	-	Reserved
[6:0]	RW	SEC[6:0]: Seconds counter. The value of the second counter is BCD coded, the counting interval is from 0 to 59. When the seconds counter counts from 59 to 0, the minute counter increments by 1. when this counter is written, times less than one second are ignored

25.6.4 RTC Date Register (RTC_DATE)

Register	Address offset	Access	Reset value	Description
RTC_DATE	0x0C	RW	0x0000_0000	RTC Date register (RTC_DATE)

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

RTC Date register (RTC_DATE)

Bit	Access	Description
[31:24]	-	Reserved
[23: 16]	RW	YEAR[7:0]: Year counter. The year counter represents the tens and ones of the decimal year. The year counter is BCD coded and counts from 00 to 99. When the year counter counts from 99 to 00, the century counter increments by 1. The century counter is 0 : 04,08,... 92,96 are leap years. The century counter is 1: 00,04,08,... 92,96 are leap year.
[15]	RW	CEN Century counter. 0 for 20 century,1 for 21 century
[14]	-	Reserved
[12:8]	RW	MONTH[4:0]: Month counter. The month counter is BCD coded and counts from 01 to 12. When the counts from 12 to 01, the year counter increments by 1.
[7:6]	-	Reserved
[5:0]	RW	DAY[5:0]: Day counter. The day counter is BCD coded and the counting intervals are as follows: 01 to 31: January, March, May, July, August, October, December; 01 to 30: April, June, September, November; 01 to 29: February of leap year 01 to 28: February of a non-leap year When the year counter counts from 99 to 00, the century counter increments by 1.

25.6.5 RTC Time Alarm Register(RTC_ALM1TIME)

Register	Address offset	Access	Reset value	Description
RTC_ALM1TIME	0x10	RW	0x0000_0000	RTC Time alarm register(RTC_ALM1TIME)

31	30	29	28	27	26	25	24
Reserved					ALWEEK[2:0]		
23	22	21	20	19	18	17	16
Reserved		ALH20_PA	ALHOUR19[4:0]				
15	14	13	12	11	10	9	8
Reserved		ALMIN[6:0]					
7	6	5	4	3	2	1	0
Reserved		ALXTC[6:0]					

RTC Time alarm register(RTC_ALM1TIME)

Bit	Access	Description
[31:27]	-	Reserved
[26: 24]	RW	ALWEEK[2:0]: Alarm week setting.
[23: 22]	-	Reserved
[21]	RW	ALH20_PA: Alarm hour setting
[20:16]	RW	ALHOUR19[4:0]: Alarm hour setting
[15]	-	Reserved
[14:8]	RW	ALMIN[6:0]: Alarm minute setting
[7]	-	Reserved
[6:0]	RW	ALXTC[6:0]: Alarm seconds setting

25.6.6 RTC Date Alarm Register (RTC_ALM1DATE)

Register	Address offset	Access	Reset value	Description
RTC_ALM1DATE	0x14	RW	0x0000_0000	RTC Date alarm register (RTC_ALM1DATE)

31	30	29	28	27	26	25	24
Reserved	ALMYEAREN	ALMMONEN	ALMDAYEN	ALMWEEKEN	ALMHOUREN	ALMMINEN	ALMSECEN
23	22	21	20	19	18	17	16
ALYEAR[7:0]							
15	14	13	12	11	10	9	8
ALCEN	Reserved		ALMONTH[4:0]				
7	6	5	4	3	2	1	0
Reserved		ALDAY[5:0]					

RTC Date alarm register (RTC_ALM1DATE)

Bit	Access	Description
[31]	-	Reserved
[30]	RW	ALMYEAREN: Alarm year setting enabled.
[29]	RW	ALMMONEN: Alarm month setting enable.
[28]	RW	ALMDAYEN: Alarm day setting enabled.
[27]	RW	ALMWEEKEN: Alarm week setting enable.
[26]	RW	ALMHOUREN: Alarm hour setting enable.
[25]	RW	ALMMINEN: Alarm minute setting enable
[24]	RW	ALMSECEN: Alarm clock second setting enable
[23:16]	RW	ALYEAR[7:0]: Alarm year setting
[15]	RW	ALCEN: Alarm clock century setting
[14:13]	-	Reserved
[12:8]	RW	ALMONTH[4:0]: Alarm month setting
[7:6]	-	Reserved
[5:0]	RW	ALDAY[5:0]: Alarm day setting

25.6.7 RTC Periodic Alarm Clock Register (RTC_ALM2PRD)

Register	Address offset	Access	Reset value	Description
RTC_ALM2PRD	0x18	RW	0x0000_0000	RTC periodic alarm clock register (RTC_ALM2PRD)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				ALM2PR_CNT[3:0]			

RTC periodic alarm clock register (RTC_ALM2PRD)

Bit	Access	Description
[31:4]	-	Reserved
[3:0]	RW	ALM2PR_CNT[3:0]: Periodic alarm clock 2 Counting period setting. 0x0:Turn off periodic alarm 2 0x1: 1 seconds 0x2: 1/2 seconds 0x3: 1/4 second 0x4: 1/8 second 0x5: 1/16 second 0x6: 1/32 second 0x7: 1/64 second 0x8: 1/128 second 0x9: 10 seconds 0xA: 30 seconds 0xB: 1 minutes 0xC: 30 minutes 0xD: 60 minutes 0xE: 12 hours 0xF: 24 hours

25.6.8 RTC Clock Trim Register (RTC_CLKTRIM)

Register	Address offset	Access	Reset value	Description
RTC_CLKTRIM	0x1C	RW	0x0000_0000	RTC Clock Trim Register (RTC_CLKTRIM)

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

RTC Clock Trim Register (RTC_CLKTRIM)

Bit	Access	Description
[31:10]	-	Reserved
[9:8]	RW	MODE[1:0]: Clock Adjustment Register. Determines the frequency of clock adjustments. 0x0: every 60 seconds(SEC=00) 0x1: every 30 seconds(SEC=00, 30) 0x2: every 15 seconds(SEC=00, 15, 30, 45) 0x3: every 6 seconds(SEC=00, 06, 12, 18, 24, 30, 36, 42, 48, 54)
[7:0]	RW	TRIM[7:0]: Clock compensation time register. This register is a signed integer. (-128 +127)

25.6.9 RTC Initialize and Status Registers (RTC_ISR)

Register	Address offset	Access	Reset value	Description
RTC_ISR	0x20	RW	0x0000_0000	RTC Initialize and Status Registers (RTC_ISR)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		ALM2_F	ALM1_F	Reserved	RSF	WAITF	WAIT

RTC Initialize and Status Registers (RTC_ISR)

Bit	Access	Description
[31:6]	-	Reserved
[5]	R	ALM2_F: Periodic alarm clock 2 Interrupt initial state register. When this register is read, the status value is returned: 0: Periodic alarm clock 2 Interrupt is not activated. 1: Periodic alarm clock 2 Interrupt is activated.
[4]	R	ALM1_F: Alarm interrupt initial status register. When this register is read, the status value is returned: 0: Alarm clock interrupt is not activated. 1: Alarm clock interrupt is activated
[3]	-	Reserved

[2]	RW	<p>RSF: Register Synchronization Flag</p> <p>This bit is set by hardware whenever the content of the calendar register is copied into the shadow register(RTC_TIME,RTC_DATE). when in ignore shadow register mode(BYPHAD=1),this bit is cleared by hardware in initialization mode or cleared by software. In initialization mode, this bit can be cleared by hardware/software. 0: Calendar shadow register not yet synchronized; 1: Calendar shadow register has been synchronized.</p> <p>Note : That software cannot write 1</p>
[1]	RW	<p>WAITF: 0: Non-write/Read status 1: Write/Read status</p> <p>Note : WAITF is WAIT bit setting valid flag. Please confirm before writing/reading Whether the bit is "1". During the counting process, During the counting process, the WAIT bit clears "0" after the write is completed.</p>
[0]	RW	<p>WAIT:</p> <p>0: Normal counting mode 1: Write/Read mode</p> <p>Note : When writing/reading, please set the position "1",Since the counter is counting continuously, please complete the write/read operation within 1 second and clear the bit "0".</p>

25.6.10 RTC Status Clear Register(RTC_INTCLR)

Register	Address offset	Access	Reset value	Description
RTC_INTCLR	0x24	W	0x0000_0000	RTC Status Clear Register(RTC_INTCLR)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		ALM2_CLR	ALM1_CLR	Reserved			

RTC Status Clear Register(RTC_INTCLR)

Bit	Access	Description
[31:6]	-	Reserved
[5]	W	<p>ALM2_CLR: Periodic alarm clock 2 Interrupt initial state clear register.</p> <p>When this register is written, the interrupt initial status is required to be cleared: 0: No operation. 1: Periodic alarm clock 2 Interrupt initial status is cleared.</p>
[4]	W	<p>ALM1_CLR: Alarm interrupt initial status clear register.</p> <p>When this register is written, the interrupt initial status is required to be cleared: 0: No operation. 1: Alarm interrupt initial state is cleared.</p>
[3:0]	-	Reserved

25.6.11 RTC Write Protection Register(RTC_WPR)

Register	Address offset	Access	Reset value	Description
RTC_WPR	0x28	W	0x0000_0000	RTC Write Protection Register(RTC_WPR)

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

RTC Write Protection Register(RTC_WPR)

Bit	Access	Description
[31:8]	-	Reserved
[7:0]	W	<p>WPR[7:0]: Write the specified keyword to enable RTC register write permission. 1. Write '0xCA' to RTC_WPR register; 2. Write '0x53' to RTC_WPR register.</p> <p>Note : After the protection is released, any further write to this register will reactivate the write protection.</p>

26 Analog-to-digital converter (ADC)

26.1 ADC introduction

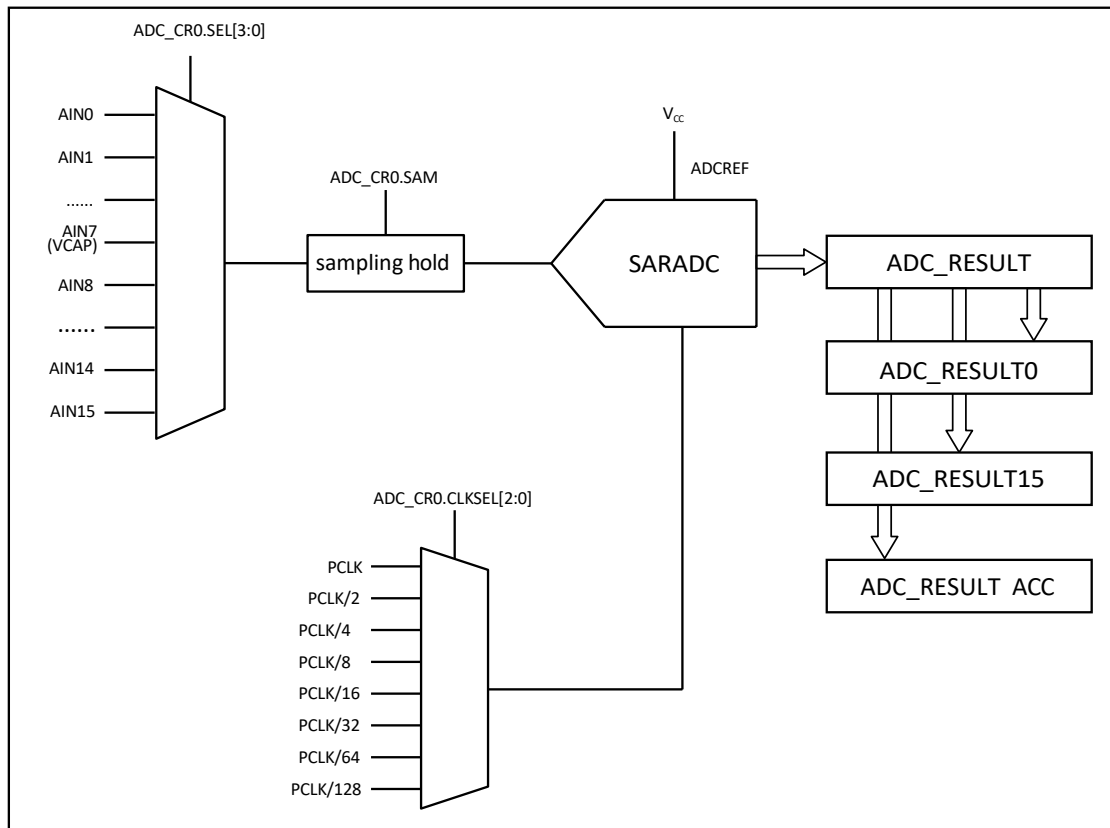
The chip integrates a 12 high precision, high conversion rate successive approximation(SAR) analog- to-digital converter(ADC)module.

Has the following properties:

- 12 digits conversion precision
- 1MSPS conversion speed
- Up to 16 channels,includes 1 VCAP calibration channel
- Reference Voltage is the supply voltage
- ADC Voltage input range for : 0-VREF
- 3 conversion modes: single conversion, continuous conversion, cumulative conversion
- ADC conversion rate software configurable
- Support on-chip and peripheral interrupts to automatically trigger ADC conversion start, effectively reducing chip power consumption and improving real-time conversion

26.2 ADC Block Diagram

Fig 26.2-1 ADC block diagram

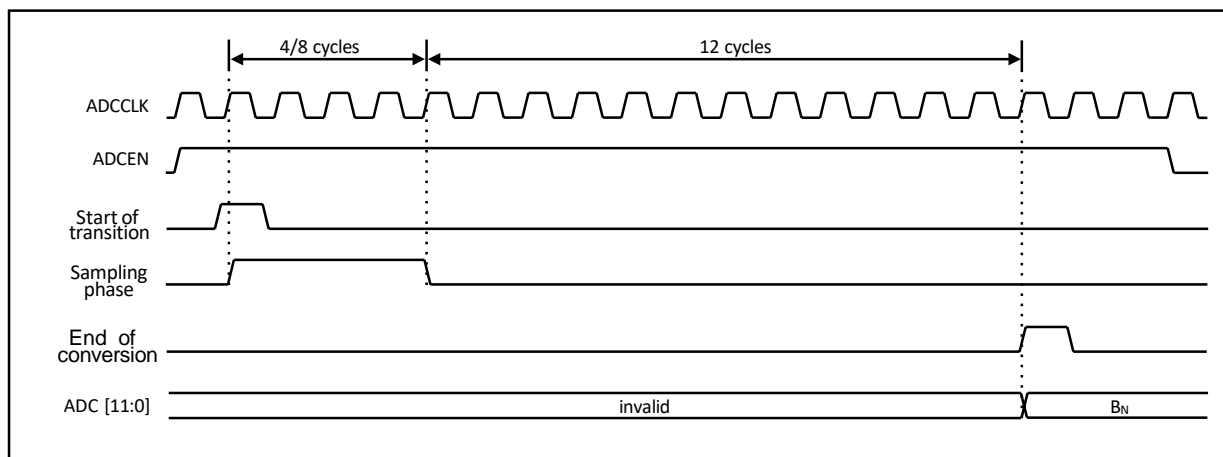


26.3 Conversion timing and speed

The conversion timing of ADC is shown in the figure below: a complete ADC conversion consists of a sampling process and a successive comparison process. The sampling process requires 4 ~ 8 ADC clocks, configured by ADC_CR0.SAM; the successive comparison process requires 12 ADC Clock. So, one ADC conversion needs 16 ~ 20 ADC clocks in total.

The unit of ADC conversion speed is sps(samples per second), which is how many ADC conversions per second. ADC conversion speed is calculated as : ADC clock frequency/The number of ADC clocks required for an ADC conversion

Fig 26.3-1 ADC Transition sequence diagram



26.4 Single Conversion Mode

In single-conversion mode, ADC performs only one conversion after startup, which can convert all 16 channels ADC channels. This mode can be started either by setting the ADC_CR0.START bit or an external trigger by setting ADC_CR1[9:0]. Once the ADC conversion of the selected channel is completed, the ADC_CR0.START bit is automatically cleared and the conversion result is stored in the ADC_RESULT register.

Start ADC single conversion operation flow through START bit:

1. Configure the ADC channel to be converted as an analog port , by configuring the corresponding GPIO based on the pin.
2. Configure ADC_CR2.CIRCLE_MODE to 0 to select acyclic mode.
3. Configure ADC_CR1.CT to 0 to select single conversion mode.
4. Configure ADC_CR0.SAM and ADC_CR0.CLKSEL, set the conversion speed of ADC.
5. Configure ADC_CR0.SEL, select the channel to be converted(Note that it needs to be consistent with step1).
6. Configure ADC_CR0.ADCEN to 1, enableADCmodule.
7. Configure ADC_CR0.START to 1, startADCsingle conversion.
8. Wait for ADC_CR0.START is 0, read ADC_RESULT register to get ADC conversion result.
9. To convert other channels, repeat steps4 ~ 7.
10. Configure ADC_CR0.ADCEN, turn off ADC module.

Note: The flow configuration of the ADC conversion triggered by the internal signal is similar, and the option of trigger needs to be added.

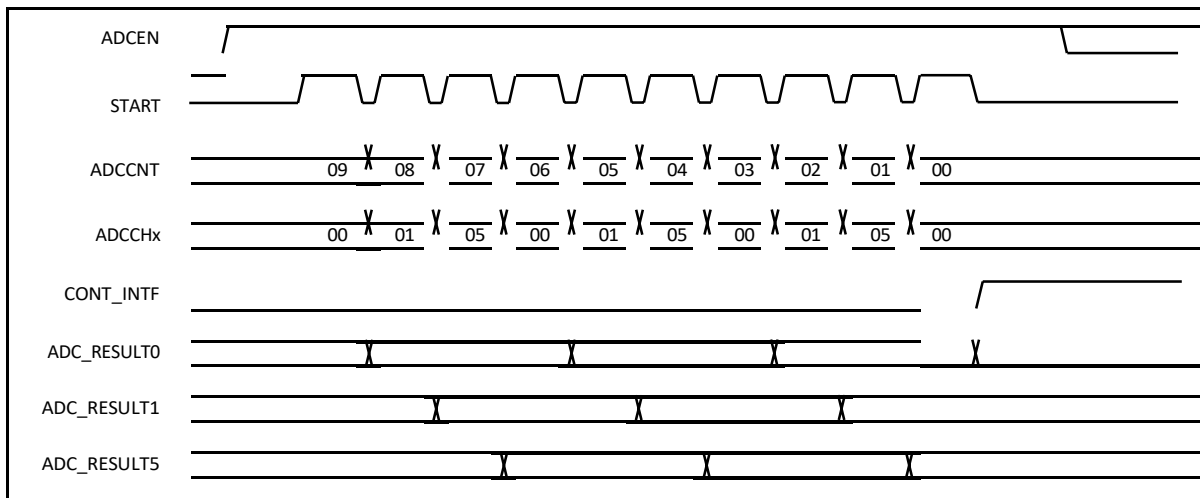
26.5 Continuous Conversion Mode

In the continuous conversion mode, start ADC once to convert multiple channels sequentially; the convertible ADC channels are AIN0 ~ AIN15. The total number of ADC conversions is configured by ADC_CR2.ADCNT[7:0]; The channel to be converted is configured by ADC_CR2.CHEN[15:0]. This mode can be started either by setting the ADC_CR0.START bit or by setting the external trigger of ADC_CR1[9:0]. After starting continuous conversion, the AD Cmodule converts the channels to be converted in AIN0 ~ AIN15 in turn until the total conversion times are completed. After the ADC module completes the total number of conversions, the ADC_RAWINTSR.CONT_INTF bit will be automatically set to 1, and the conversion result is saved in ADC_RESULT0 ~ ADC_RESULT15 corresponding to the conversion channel register.

If the total number of conversions is greater than the number of ADC channels to be converted, ADC_RESULT0 ~ ADC_RESULT15 registers only save the last conversion result.

The figure below demonstrates 10 consecutive conversions of AIN0, AIN1, AIN5. Set START to 1 through the register, the state opportunity within the ADC converts AIN0, AIN1, and AIN5 in turn until the count value of ADCNT reaches 0.

Fig 26.5-1 Adc continuous conversion



Configuration steps:

1. Configure the ADC channel to be converted as an analog port, by configuring the corresponding GPIO based on the PIN.
2. Configure ADC_CR2.CIRCLE_MODE to 0 to select acyclic mode.
3. Configure ADC_CR1.CT to 1 to select continuous conversion mode.
4. Configure ADC_CR2.ADCNT[7:0] to select the total number of transitions for successive transitions.
5. Configure ADC_CR0.SAM and ADC_CR0.CLKSEL, set the conversion speed of ADC.
6. Configure ADC_CR2.CHEN[15:0] to enable the channel to be converted.

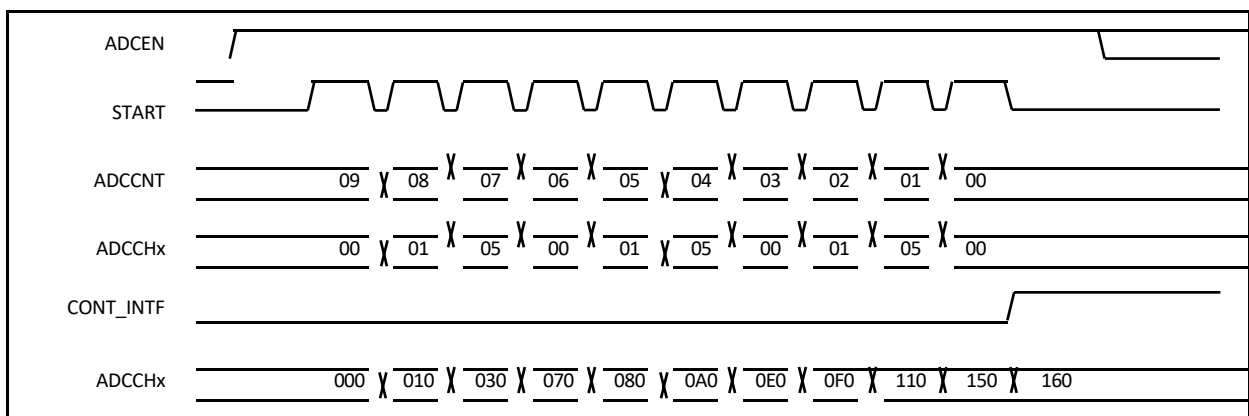
7. Configure ADC_INTCLR.CONT_INTC to 1, clear ADC_RAWINTSR.CONT_INTF flag.
8. Configure ADC_INTEN.CONT_IEN to 1 to enable continuous conversion complete interrupt mask enable.
9. Configure ADC_CR0.STATERST to 1 to reset the continuous conversion state. Then set ADC_CR0.STATERST to 0.
10. Configure ADC_CR0.ADCEN to 1, enable ADC module.
11. Configure ADC_CR0.START to 1, start ADC continuous conversion.
12. Wait for ADC_RAWINTSR.CONT_INTF to change to 1, read ADC_RESULT0 ~ ADC_RESULT15 register to get the conversion result for the corresponding channel.
13. To convert other channels, repeat steps 6 ~ 11.
14. Configure ADC_CR0.ADCEN, turn off ADC module.

26.6 Continuous transformation cumulative mode

In the continuous conversion accumulation mode, ADC can be started once to convert multiple channels and the result of each conversion can be accumulated. The conversion ADC channels are AIN0 ~ AIN15. The total number of ADC conversions is configured by ADC_CR2.ADCCNT[7:0]. The channel to be converted is configured by ADC_CR2.CHEN[15:0]. This mode can be started either by setting the ADC_CR0.START bit or by setting the external trigger of ADC_CR1[9:0]. After continuous conversion is started, ADC module converts channels in AIN0 ~ AIN15 to be converted successively until the total number of conversion is completed. After the ADC module completes the total number of conversions, the ADC_RAWINTSR.CONT_INTF bit is automatically set to 1, and the accumulated value of the conversion result is saved in the ADC_RESULT_ACC register.

The figure below demonstrates the accumulation of 10 consecutive conversions for AIN0,AIN1,AIN5. After START is set to 1 through the register, the internal state machine of ADC performs AIN0,AIN1,AIN5 performs conversion until the count value of ADCCNT becomes 0. The ADC_RESULT_ACC registers are automatically accumulated each time a conversion is complete. The conversion results of AIN0,AIN1,AIN5 given in the figure are 0x010,0x020,0x040.

Fig 26.6-1 Continuous transformation cumulative mode



Configuration steps:

1. Configure the ADC channel to be converted as an analog port , by configuring the corresponding

GPIO based on the pin.

2. Configure ADC_CR2.CIRCLE_MODE to 0 to select acyclic mode.
3. Configure ADC_CR1.CT to 1 to select continuous conversion mode.
4. Configure ADC_CR1.RACC_EN to 1, select ADC conversion automatic accumulation function.
5. Configure ADC_CR2.ADCCNT[7:0] to select the total number of conversions for consecutive conversions.
6. Configure ADC_CR0.SAMandAD_CR0.CLKSEL, set the conversion speed of ADC.
7. Configure ADC_CR2.CHEN[15:0] to select the channel to be converted.
8. Configure ADC_INTCLR.CONT_INTC to 1, clear ADC_RAWINTSR.CONT_INTF flag.
9. Configure ADC_INTEN.CONT_IEN to 1 to enable continuous conversion complete interrupt mask enable.
10. Set ADC_CR1.RACC_CLR to 1, clear ADC_RESULT_ACC register.
11. Configure ADC_CR0.STATERST to 1 to reset the continuous conversion state. Then set ADC_CR0.STATERST to 0.
12. Configure ADC_CR0.ADCEN to 1, enable ADC module.
13. Configure ADC_CR0.START to 1, start ADC continuous conversion.
14. Wait for ADC_RAWINTSR.CONT_INTF to become 1, read ADC_RESULT_ACC register to get continuous conversion accumulation result.
15. To convert other channels, repeat steps 6 ~ 11.
16. Configure ADC_CR0.ADCEN, turn off ADC module.

26.7 Comparison of ADC conversion results

When ADC conversion is completed, ADC conversion result can be compared with the threshold set by the user, which supports high threshold comparison, low threshold comparison, and interval value comparison. This function needs to set the corresponding control bits ADC_CR1.HTCMP, ADC_CR1.LTCMP, ADC_CR1.HTCMP, ADC_CR1.LTCMP, ADC_CR1.HTCMP, ADC_CR1.LTCMP to 1. This function can realize the automatic monitoring of analog quantity, until the ADC conversion results meet the user's expectations, the interrupt application user program boundary will be generated.

High threshold comparison:

when ADC conversion result is in [ADC_HT, 4095] interval, then ADC_RAWINTSR.HHT_INTF set 1; write 1 to ADC_INTCLR.HHT_INTC clear ADC_RAWINTSR.HHT_INTF.

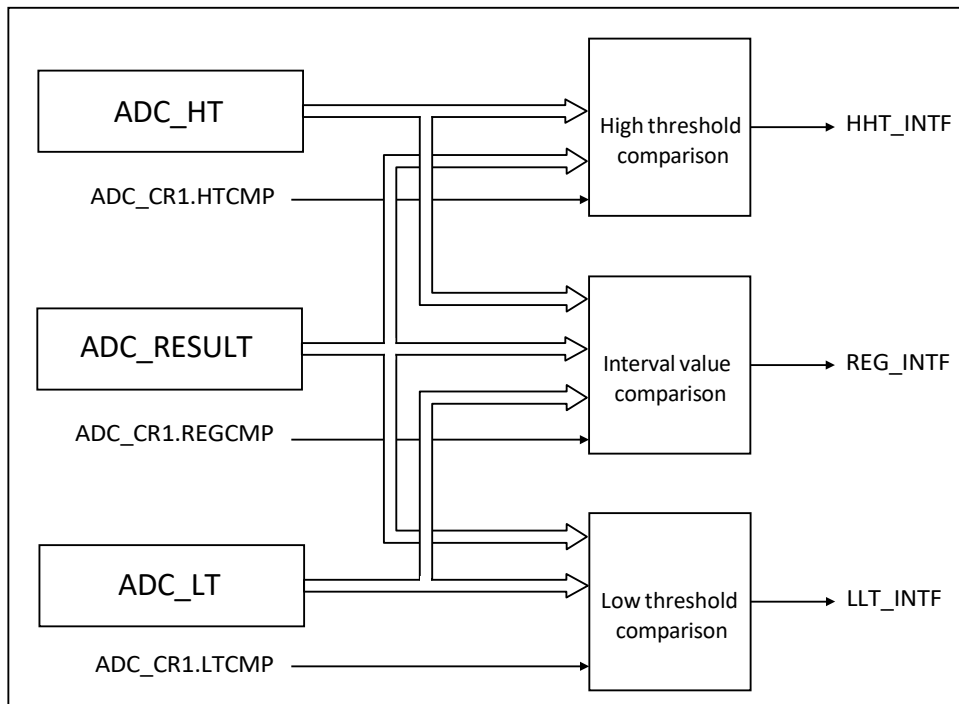
Low threshold comparison:

when ADC conversion result is in [0, ADC_LT] interval, then ADC_RAWINTSR.LLT_INTF set 1; write 1 to ADC_INTCLR.LLT_INTC clear ADC_RAWINTSR.LLT_INTF.

Interval value comparison:

when ADC conversion result is in [ADC_LT, ADC_HT] interval, then ADC_RAWINTSR.REG_INTF set 1; write 1 to ADC_INTCLR.REG_INTC clear ADC_RAWINTSR.REG_INTF.

Fig 26.7-1 Comparison of ADC conversion results



26.8 ADC interrupt

ADC interrupt requests are shown in the table below:

Tab 26.8-1 ADC interrupt request

interrupt source	interrupt flag	interrupt enable mask
ADC Continuous Conversion Done	ADC_MSKINTSR.CONT_MIF	ADC_INTEN.CONT_IEN
ADC conversion result is in interval value area	ADC_MSKINTSR.REG_MIF	ADC_INTEN.REG_IEN
ADC conversion result is in high threshold area	ADC_MSKINTSR.HHT_MIF	ADC_INTEN.HHT_IEN
ADC conversion result is in low threshold area	ADC_MSKINTSR.LLT_MIF	ADC_INTEN.LLT_IEN

26.9 ADC Register

Tab 26.9-1 ADC Registers

offset	Register	Reset value	Description
ADC base address:0x4000_2C00			
0x00	ADC_CR0	0x0000_0000	ADC Configuration Register
0x04	ADC_CR1	0x0000_7000	ADC Configuration Register 1
0x08	ADC_CR2	0x0000_0000	ADC Configuration Register 2
0x0C	ADC_RESULT0	0x0000_0000	ADC channel 0 conversion result
0x10	ADC_RESULT1	0x0000_0000	ADC channel 1 conversion result
0x14	ADC_RESULT2	0x0000_0000	ADC channel 2 conversion result
0x18	ADC_RESULT3	0x0000_0000	ADC channel 3 conversion result
0x1C	ADC_RESULT4	0x0000_0000	ADC channel 4 conversion result
0x20	ADC_RESULT5	0x0000_0000	ADC channel 5 conversion result
0x24	ADC_RESULT6	0x0000_0000	ADC channel 6 conversion result
0x28	ADC_RESULT7	0x0000_0000	ADC channel 7 conversion result
0x2C	ADC_RESULT	0x0000_0000	ADC conversion result
0x30	ADC_RESULT_ACC	0x0000_0000	ADC Conversion result accumulation value
0x34	ADC_HT	0x0000_0FFF	ADC comparison high threshold register
0x38	ADC_LT	0x0000_0000	ADC comparison low threshold register
0x44	ADC_INTEN	0x0000_0000	ADC Interrupt enable register
0x48	ADC_INTCLR	0x0000_0000	ADC interrupt clear register
0x4C	ADC_RAWINTSR	0x0000_0000	ADC Pre-mask interrupt status register
0x50	ADC_MSKINTSR	0x0000_0000	ADC Post-mask interrupt status register
0x60	ADC_RESULT8 (MG32L003 K8)	0x0000_0000	ADC channel 8 conversion result
0x64	ADC_RESULT9 (MG32L003 K8)	0x0000_0000	ADC channel 9 conversion result
0x68	ADC_RESULT10 (MG32L003 K8)	0x0000_0000	ADC channel 10 conversion result
0x6C	ADC_RESULT11 (MG32L003 K8)	0x0000_0000	ADC channel 11 conversion result
0x70	ADC_RESULT12 (MG32L003 K8)	0x0000_0000	ADC channel 12 conversion result
0x74	ADC_RESULT13 (MG32L003 K8)	0x0000_0000	ADC channel 13 conversion result
0x78	ADC_RESULT14 (MG32L003 K8)	0x0000_0000	ADC channel 14 conversion result
0x7C	ADC_RESULT15 (MG32L003 K8)	0x0000_0000	ADC channel 15 conversion result

26.9.1 ADC Configuration Register(ADC_CR0)

Register	Address offset	Access	Reset value	Description
ADC_CR0	0x00	RW	0x0000_0000	ADC Configuration Register

MG32L003 F8

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
STATERST	Reserved			SAM	SEL[2:0]		
7	6	5	4	3	2	1	0
Reserved	CLKSEL[2:0]			Reserved		START	ADCEN

ADC Configuration Register(ADC_CR0)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15]	RW	STATERST: ADC Continuous Conversion Status Control 0: invalid 1: Reset ADC Continuous Conversion Status
[14:12]	-	EMPF: Reserved, always read as 0
[11]	RW	SAM: ADC Sampling period selection 0:4 sampling periods 1:8 sampling periods
[10:8]	RW	SEL[2:0]: ADC Conversion channel selection(single conversion mode) 000: Select channel 0 001: Select channel 1 010: Select channel 2 011: Select channel 3 100: Select channel 4 101: Select channel 5 110: Select channel 6 111: Select channel 7(VCAP)
[7]	-	Reserved, always read as 0
[6:4]	RW	CLKSEL[2:0]: ADC Clock Select 000:PCLK clock 001:PCLK clock 2 frequency division 010:PCLK clock 4 frequency division 011:PCLK clock 8 frequency division 100:PCLK clock 16 frequency division 101:PCLK clock 32 frequency division 110:PCLK clock 64 frequency division 111:PCLK clock 128 frequency division
[3:2]	-	Reserved, always read as 0
[1]	RW	START: ADC Conversion Control 0: Stop ADC conversion 1: Start ADC conversion Note: This bit field software writes 1, hardware clears 0

[0]	RW	ADCEN: ADC Enable Control 0: Disable ADC 1: Enable ADC
-----	----	---

MG32L003 K8

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
STATERST	Reserved			SAM	SEL[2:0]		
7	6	5	4	3	2	1	0
SEL[3]	CLKSEL[2:0]			Reserved		START	ADCEN

ADC Configuration Register(ADC_CR0)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15]	RW	STATERST: ADC Continuous Conversion Status Control 0: invalid 1: Reset ADC Continuous Conversion Status
[14:12]	-	EMPF: Reserved, always read as 0
[11]	RW	SAM: ADC Sampling period selection 0:4 sampling periods 1:8 sampling periods
[10:8]	RW	SEL[2:0]: ADC Conversion channel selection (single conversion mode) ADC Conversion channel selection SEL[3:0] as follow: 0000: Select channel 0 0001: Select channel 1 0010: Select channel 2 0011: Select channel 3 0100: Select channel 4 0101: Select channel 5 0110: Select channel 6 0111: Select channel 7(VCAP) 1000: Select channel 8 1001: Select channel 9 1010: Select channel 10 1011: Select channel 11 1100: Select channel 12 1101: Select channel 13 1110: Select channel 14 1111: Select channel 15
[7]	RW	SEL[3] :ADC Conversion channel selection

[6:4]	RW	CLKSEL[2:0]: ADC Clock Select 000:PCLK clock 001:PCLK clock 2 frequency division 010:PCLK clock 4 frequency division 011:PCLK clock 8 frequency division 100:PCLK clock 16 frequency division 101:PCLK clock 32 frequency division 110:PCLK clock 64 frequency division 111:PCLK clock 128 frequency division
[3:2]	-	Reserved, always read as 0
[1]	RW	START: ADC Conversion Control 0: Stop ADC conversion 1: Start ADC conversion Note: <i>This bit field software writes 1, hardware clears 0</i>
[0]	RW	ADCEN: ADC Enable Control 0: Disable ADC 1: Enable ADC

26.9.2 ADC Configuration Register 1((ADC_CR1)

Register	Address offset	Access	Reset value	Description
ADC_CR1	0x04	RW	0x0000_7000	ADC Configuration Register 1

MG32L003 F8

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RACC_CLR	REGCMP	HTCMP	LTCMP	RACC_EN	CT	TRIGS1[4:3]	
7	6	5	4	3	2	1	0
TRIGS1[2:0]				TRIGS0[4:0]			

ADC Configuration Register 1(ADC_CR1)

Bit	Access	Description
[31:16]	-	Reserved, always read as 0
[15]	RW	RACC_CLR: ADC clear conversion result accumulation register 0: no effect; 1:ADC conversion result accumulation register(ADC_RESULT_ACC) cleared. Note: The bit is read as 0, so special attention should be paid to the value of the bit when operating this register to prevent malfunction.
[14]	RW	REGCMP: ADC Interval comparison control 0: Disable interval comparison 1: Enable interval comparison
[13]	RW	HTCMP: ADC High Threshold Compare Control 0: Disable high threshold comparison 1: Enable High Threshold Compare

[12]	RW	<p>LTCMP: ADC Low Threshold Compare Control 0: Disable low threshold comparison 1: Enable Low Threshold Compare</p>
[11]	RW	<p>RACC_EN: ADC conversion result automatic accumulation control 0: Disable ADC conversion result automatic accumulation function 1: Enable ADC conversion result automatic accumulation function</p>
[10]	RW	<p>CT: ADC conversion mode selection 0: single conversion mode 1: Continuous conversion mode</p>
[9:5]	RW	<p>TRIGS1[4:0]: ADC conversion automatically triggers Option 1: 00000: Disables automatic ADC conversion 00001: Timer 10 interrupt automatically triggers ADC conversion 00010: Timer 11 interrupt automatically triggers ADC conversion 00011: TIM1 interrupt automatically triggers ADC conversion 00100: LPTIM interrupt automatically triggers ADC conversion 00101: TIM1 TRGO automatically triggers ADC conversion 00110: TIM2 TRGO automatically triggers ADC conversion 00111: TIM2 interrupt automatically triggers ADC conversion 01000: UART1 interrupt automatically triggers ADC conversion 01001: UART2 interrupt automatically triggers ADC conversion 01010: LPUART interrupt automatically triggers ADC conversion 01011: VCMP0 interrupt automatically triggers ADC conversion 01100: NC 01101: RTC interrupt automatically triggers ADC conversion 01110: PCA interrupt automatically triggers ADC conversion 01111: SPI interrupt automatically triggers ADC conversion 10000: PA1 interrupt automatically triggers ADC conversion 10001: PA2 interrupt automatically triggers ADC conversion 10010: PA3 interrupt automatically triggers ADC conversion 10011: PB4 interrupt automatically triggers ADC conversion 10100: PB5 interrupt automatically triggers ADC conversion 10101: PC3 interrupt automatically triggers ADC conversion 10110: PC4 interrupt automatically triggers ADC conversion 10111: PC5 interrupt automatically triggers ADC conversion 11000: PC6 interrupt automatically triggers ADC conversion 11001: PC7 interrupt automatically triggers ADC conversion 11010: PD1 interrupt automatically triggers ADC conversion 11011: PD2 interrupt automatically triggers ADC conversion 11100: PD3 interrupt automatically triggers ADC conversion 11101: PD4 interrupt automatically triggers ADC conversion 11110: PD5 interrupt automatically triggers ADC conversion 11111: PD6 interrupt automatically triggers ADC conversion</p> <p>Note: <i>The ADC is triggered using the rising edge of each interrupt flag bit. If need re- trigger, The interrupt flag needs to be cleared. If not need to enter the interrupt service routine, do not enable it NVIC interrupt enabled.</i></p>

[4:0]	RW	<p>TRIGS1[4:0]: ADC conversion automatically triggers Option 1: 00000: Disables automatic ADC conversion 00001: Timer10 interrupt automatically triggers ADC conversion 00010: Timer11 interrupt automatically triggers ADC conversion 00011: TIM1 interrupt automatically triggers ADC conversion 00100: LPTIM interrupt automatically triggers ADC conversion 00101: TIM1 TRGO automatically triggers ADC conversion 00110: TIM2 TRGO automatically triggers ADC conversion 00111: TIM2 interrupt automatically triggers ADC conversion 01000: UART1 interrupt automatically triggers ADC conversion 01001: UART2 interrupt automatically triggers ADC conversion 01010: LPUART interrupt automatically triggers ADC conversion 01011: VCMPO interrupt automatically triggers ADC conversion 01100: NC 01101: RTC interrupt automatically triggers ADC conversion 01110: PCA interrupt automatically triggers ADC conversion 01111: SPI interrupt automatically triggers ADC conversion 10000: PA1 interrupt automatically triggers ADC conversion 10001: PA2 interrupt automatically triggers ADC conversion 10010: PA3 interrupt automatically triggers ADC conversion 10011: PB4 interrupt automatically triggers ADC conversion 10100: PB5 interrupt automatically triggers ADC conversion 10101: PC3 interrupt automatically triggers ADC conversion 10110: PC4 interrupt automatically triggers ADC conversion 10111: PC5 interrupt automatically triggers ADC conversion 11000: PC6 interrupt automatically triggers ADC conversion 11001: PC7 interrupt automatically triggers ADC conversion 11010: PD1 interrupt automatically triggers ADC conversion 11011: PD2 interrupt automatically triggers ADC conversion 11100: PD3 interrupt automatically triggers ADC conversion 11101: PD4 interrupt automatically triggers ADC conversion 11110: PD5 interrupt automatically triggers ADC conversion 11111: PD6 interrupt automatically triggers ADC conversion</p> <p>Note: <i>The ADC is triggered using the rising edge of each interrupt flag bit. If need re- trigger, The interrupt flag needs to be cleared. If not need to enter the interrupt service routine, do not enable it NVIC interrupt enabled.</i></p>
-------	----	---

MG32L003 K8

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						TRIGS1[5]	TRIGS0[5]
15	14	13	12	11	10	9	8
RACC_CLR	REGCMP	HTCMP	LTCMP	RACC_EN	CT	TRIGS1[4:3]	
7	6	5	4	3	2	1	0
TRIGS1[2:0]				TRIGS0[4:0]			

ADC Configuration Register 1(ADC_CR1)

Bit	Access	Description
[31:18]	-	Reserved, always read as 0
[17]	RW	TRIGS1[5]
[16]	RW	TRIGS0[5]
[15]	RW	<p>RACC_CLR: ADC clear conversion result accumulation register 0: no effect; 1:ADC conversion result accumulation register(ADC_RESULT_ACC) cleared.</p> <p>Note: The bit is read as 0, so special attention should be paid to the value of the bit when operating this register to prevent malfunction.</p>
[14]	RW	<p>REGCMP: ADC Interval comparison control 0: Disable interval comparison 1: Enable interval comparison</p>
[13]	RW	<p>HTCMP: ADC High Threshold Compare Control 0: Disable high threshold comparison 1: Enable High Threshold Compare</p>
[12]	RW	<p>LTCMP: ADC Low Threshold Compare Control 0: Disable low threshold comparison 1: Enable Low Threshold Compare</p>
[11]	RW	<p>RACC_EN: ADC conversion result automatic accumulation control 0: Disable ADC conversion result automatic accumulation function 1: Enable ADC conversion result automatic accumulation function</p>
[10]	RW	<p>CT: ADC conversion mode selection 0: single conversion mode 1: Continuous conversion mode</p>

[9:5]	RW	<p>TRIGS1[4:0]: ADC conversion automatically triggers Option 1: TRIGS1[5:0] as follow 000000: Disables automatic ADC conversion 000001: Timer 10 interrupt automatically triggers ADC conversion 000010: Timer 11 interrupt automatically triggers ADC conversion 000011: TIM1 interrupt automatically triggers ADC conversion 000100: LPTIM interrupt automatically triggers ADC conversion 000101: TIM1 TRGO automatically triggers ADC conversion 000110: TIM2 TRGO automatically triggers ADC conversion 000111: TIM2 interrupt automatically triggers ADC conversion 001000: UART1 interrupt automatically triggers ADC conversion 001001: UART2 interrupt automatically triggers ADC conversion 001010: LPUART interrupt automatically triggers ADC conversion 001011: VCMP0 interrupt automatically triggers ADC conversion 001100: NC 001101: RTC interrupt automatically triggers ADC conversion 001110: PCA interrupt automatically triggers ADC conversion 001111: SPI interrupt automatically triggers ADC conversion 010000: PA1 interrupt automatically triggers ADC conversion 010001: PA2 interrupt automatically triggers ADC conversion 010010: PA3 interrupt automatically triggers ADC conversion 010011: PB4 interrupt automatically triggers ADC conversion 010100: PB5 interrupt automatically triggers ADC conversion 010101: PC3 interrupt automatically triggers ADC conversion 010110: PC4 interrupt automatically triggers ADC conversion 010111: PC5 interrupt automatically triggers ADC conversion 011000: PC6 interrupt automatically triggers ADC conversion 011001: PC7 interrupt automatically triggers ADC conversion 011010: PD1 interrupt automatically triggers ADC conversion 011011: PD2 interrupt automatically triggers ADC conversion 011100: PD3 interrupt automatically triggers ADC conversion 011101: PD4 interrupt automatically triggers ADC conversion 011110: PD5 interrupt automatically triggers ADC conversion 011111: PD6 interrupt automatically triggers ADC conversion 100001: PA4 interrupt automatically triggers ADC conversion 100010: PB0 interrupt automatically triggers ADC conversion 100011: PB1 interrupt automatically triggers ADC conversion 100100: PB2 interrupt automatically triggers ADC conversion 100101: PB3 interrupt automatically triggers ADC conversion 100110: PB6 interrupt automatically triggers ADC conversion 100111: PB7 interrupt automatically triggers ADC conversion 101000: PC0 interrupt automatically triggers ADC conversion 101001: PC1 interrupt automatically triggers ADC conversion 101010: PC2 interrupt automatically triggers ADC conversion 101011: PD0 interrupt automatically triggers ADC conversion 101100: PD7 interrupt automatically triggers ADC conversion</p> <p>Note: The ADC is triggered using the rising edge of each interrupt flag bit. If need re- trigger, The interrupt flag needs to be cleared. If not need to enter the interrupt service routine, do not enable it NVIC interrupt enabled.</p>
-------	----	--

[4:0]	RW	<p>TRIGS0[4:0]: ADC conversion automatically triggers Option 1: TRIGS0[5:0] as follow 000000: Disables automatic ADC conversion 000001: Timer10 interrupt automatically triggers ADC conversion 000010: Timer11 interrupt automatically triggers ADC conversion 000011: TIM1 interrupt automatically triggers ADC conversion 000100: LPTIM interrupt automatically triggers ADC conversion 000101: TIM1 TRGO automatically triggers ADC conversion 000110: TIM2 TRGO automatically triggers ADC conversion 000111: TIM2 interrupt automatically triggers ADC conversion 001000: UART1 interrupt automatically triggers ADC conversion 001001: UART2 interrupt automatically triggers ADC conversion 001010: LPUART interrupt automatically triggers ADC conversion 001011: VCMP0 interrupt automatically triggers ADC conversion 001100: NC 001101: RTC interrupt automatically triggers ADC conversion 001110: PCA interrupt automatically triggers ADC conversion 001111: SPI interrupt automatically triggers ADC conversion 010000: PA1 interrupt automatically triggers ADC conversion 010001: PA2 interrupt automatically triggers ADC conversion 010010: PA3 interrupt automatically triggers ADC conversion 010011: PB4 interrupt automatically triggers ADC conversion 010100: PB5 interrupt automatically triggers ADC conversion 010101: PC3 interrupt automatically triggers ADC conversion 010110: PC4 interrupt automatically triggers ADC conversion 010111: PC5 interrupt automatically triggers ADC conversion 011000: PC6 interrupt automatically triggers ADC conversion 011001: PC7 interrupt automatically triggers ADC conversion 011010: PD1 interrupt automatically triggers ADC conversion 011011: PD2 interrupt automatically triggers ADC conversion 011100: PD3 interrupt automatically triggers ADC conversion 011101: PD4 interrupt automatically triggers ADC conversion 011110: PD5 interrupt automatically triggers ADC conversion 011111: PD6 interrupt automatically triggers ADC conversion 100001: PA4 interrupt automatically triggers ADC conversion 100010: PB0 interrupt automatically triggers ADC conversion 100011: PB1 interrupt automatically triggers ADC conversion 100100: PB2 interrupt automatically triggers ADC conversion 100101: PB3 interrupt automatically triggers ADC conversion 100110: PB6 interrupt automatically triggers ADC conversion 100111: PB7 interrupt automatically triggers ADC conversion 101000: PC0 interrupt automatically triggers ADC conversion 101001: PC1 interrupt automatically triggers ADC conversion 101010: PC2 interrupt automatically triggers ADC conversion 101011: PD0 interrupt automatically triggers ADC conversion 101100: PD7 interrupt automatically triggers ADC conversion</p> <p>Note: The ADC is triggered using the rising edge of each interrupt flag bit. If need re- trigger, The interrupt flag needs to be cleared. If not need to enter the interrupt service routine, do not enable it NVIC interrupt enabled.</p>
-------	----	--

26.9.3 ADC Configuration Register 2(ADC_CR2)

Register	Address offset	Access	Reset value	Description
ADC_CR2	0x08	RW	0x0000_0000	ADC Configuration Register 1

MG32L003 F8

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

ADC Configuration Register 2(ADC_CR2)

Bit	Access	Description
[31:17]	-	Reserved, always read as 0
[16]	RW	CIRCLE_MODE: ADC Conversion cycle mode selection 0: Acyclic mode 1: Cycle mode
[15:8]	RW	ADCCNT[7:0]: ADC Consecutive conversion count configuration 0: Continuous conversion 1 times 1: Consecutive conversion 2 times ... 255: Continuous conversion 256 times
[7:0]	RW	CHEN[7:0]: ADC Continuous conversion channel 7~0 Enable 0: disable 1: Enable

MG32L003 K8

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

ADC Configuration Register 2(ADC_CR2)

Bit	Access	Description
[31:28]	-	Reserved, always read as 0

[27:20]	RW	CHEN[15:8] ADC Continuous conversion channel 15~8 Enable 0: disable 1: Enable
---------	----	--

[19:17]	-	Reserved, always read as 0
[16]	RW	CIRCLE_MODE: ADC Conversion cycle mode selection 0: Acyclic mode 1: Cycle mode
[15:8]	RW	ADCCNT[7:0]: ADC Consecutive conversion count configuration 0: Continuous conversion 1 times 1: Consecutive conversion 2 times ... 255: Continuous conversion 256 times
[7:0]	RW	CHEN[7:0]: ADC Continuous conversion channel 7~0 Enable 0: disable 1: Enable

26.9.4 ADC channel 0 conversion result(ADC_RESULT0)

Register	Address offset	Access	Reset value	Description
ADC_RESULT_0	0x0C	R	0x0000_0000	ADC channel 0 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT0[11:8]			
7	6	5	4	3	2	1	0
RESULT0[7:0]							

ADC channel 0 conversion result(ADC_RESULT0)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT0[11:0]: ADC Channel 0 Conversion result

26.9.5 ADC channel 1 conversion result(ADC_RESULT1)

Register	Address offset	Access	Reset value	Description
ADC_RESULT_1	0x10	R	0x0000_0000	ADC channel 1 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT1[11:8]			
7	6	5	4	3	2	1	0
RESULT1[7:0]							

ADC channel 1 conversion result(ADC_RESULT1)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT1[11:0]: ADC Channel 1 Conversion result

26.9.6 ADC channel 2 conversion result(ADC_RESULT2)

Register	Address offset	Access	Reset value	Description
ADC_RESULT_2	0x14	R	0x0000_0000	ADC channel 2 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT2[11:8]			
7	6	5	4	3	2	1	0
RESULT2[7:0]							

ADC channel 2 conversion result(ADC_RESULT2)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT2[11:0]: ADC Channel 2 Conversion result

26.9.7 ADC channel 3 conversion result(ADC_RESULT3)

Register	Address offset	Access	Reset value	Description
ADC_RESULT_3	0x18	R	0x0000_0000	ADC channel 3 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT3[11:8]			
7	6	5	4	3	2	1	0
RESULT3[7:0]							

ADC channel 3 conversion result(ADC_RESULT3)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT3[11:0]: ADC Channel 3 Conversion result

26.9.8 ADC channel 4 conversion result(ADC_RESULT4)

Register	Address offset	Access	Reset value	Description
ADC_RESULT 4	0x1C	R	0x0000_0000	ADC channel 4 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT4[11:8]			
7	6	5	4	3	2	1	0
RESULT4[7:0]							

ADC channel 4 conversion result(ADC_RESULT4)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT4[11:0]: ADC Channel 4 Conversion result

26.9.9 ADC channel 5 conversion result(ADC_RESULT5)

Register	Address offset	Access	Reset value	Description
ADC_RESULT5	0x20	R	0x0000_0000	ADC channel 5 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT5[11:8]			
7	6	5	4	3	2	1	0
RESULT5[7:0]							

ADC channel 5 conversion result(ADC_RESULT5)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT5[11:0]: ADC Channel 5 Conversion result

26.9.10 ADC channel 6 conversion result(ADC_RESULT6)

Register	Address offset	Access	Reset value	Description
ADC_RESULT6	0x24	R	0x0000_0000	ADC channel 6 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT6[11:8]			
7	6	5	4	3	2	1	0

RESULT6[7:0]

ADC channel 6 conversion result(ADC_RESULT6)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT6[11:0]: ADC Channel 6 Conversion result

26.9.11 ADC channel 7 conversion result(ADC_RESULT7)

Register	Address offset	Access	Reset value	Description
ADC_RESULT7	0x28	R	0x0000_0000	ADC channel 7 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT7[11:8]			
7	6	5	4	3	2	1	0
RESULT7[7:0]							

ADC channel 7 conversion result(ADC_RESULT7)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT7[11:0]: ADC Channel 7 Conversion result

26.9.12 ADC conversion result(ADC_RESULT)

Register	Address offset	Access	Reset value	Description
ADC_RESULT	0x2C	R	0x0000_0000	ADC conversion result

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

ADC conversion result(ADC_RESULT)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT[11:0]: ADC Conversion result

26.9.13 ADC Conversion result accumulation value(ADC_RESULT_ACC)

Register	Address offset	Access	Reset value	Description
ADC_RESULT_AC C	0x30	R	0x0000_0000	ADC Conversion result accumulation value

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

ADC Conversion result accumulation value(ADC_RESULT_ACC)

Bit	Access	Description
[31:20]	-	Reserved, always read as 0
[19:0]	R	RESULT_ACC[19:0]: ADC Conversion result accumulation value

26.9.14 ADC comparison high threshold register(ADC_HT)

Register	Address offset	Access	Reset value	Description
ADC_ht	0x34	RW	0x0000_0FFF	ADC comparison high threshold register

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

ADC comparison high threshold register(ADC_HT)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	RW	HT[11:0]: ADC conversion results are compared to the high threshold

26.9.15 ADC comparison low threshold register(ADC_LT)

Register	Address offset	Access	Reset value	Description
ADC_LT	0x38	RW	0x0000_0000	ADC comparison low threshold register

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

ADC comparison low threshold register(ADC_LT)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	RW	LT[11:0]: ADC conversion results are compared to lower thresholds

26.9.16 ADC Interrupt enable register(ADC_INTEN)

Register	Address offset	Access	Reset value	Description
ADC_INTEN	0x44	RW	0x0000_0000	ADC Interrupt enable register

MG32L003 F8

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

ADC Interrupt enable register(ADC_INTEN)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11]	RW	CONT_IEN: Continuous conversion completes interrupt mask configuration 0: Disable interrupt 1: Enable interrupt
[10]	RW	REG_IEN: ADC Conversion result comparison interval interrupt mask configuration 0: disable 1: Enable
[9]	RW	HHT_IEN: ADC Conversion result compare high threshold interrupt mask configuration 0: disable 1: Enable
[8]	RW	LLT_IEN: ADC Conversion result compare lower threshold interrupt mask configuration 0: disable 1: Enable
[7:0]	RW	ADCXIEN[7:0]: ADC Channel 7~0 interrupt mask configuration 0: disable 1: Enable

MG32L003 K8

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

ADC Interrupt enable register(ADC_INTEN)

Bit	Access	Description
[31:20]	-	Reserved, always read as 0
[19:12]	RW	ADCXIEN[15:8]: ADC Channel 15~8 interrupt mask configuration 0: disable 1: Enable
[11]	RW	CONT_IEN: Continuous conversion completes interrupt mask configuration 0: Disable interrupt 1: Enable interrupt
[10]	RW	REG_IEN: ADC Conversion result comparison interval interrupt mask configuration 0: disable 1: Enable
[9]	RW	HHT_IEN: ADC Conversion result compare high threshold interrupt mask configuration 0: disable 1: Enable
[8]	RW	LLT_IEN: ADC Conversion result compare lower threshold interrupt mask configuration 0: disable 1: Enable
[7:0]	RW	ADCXIEN[7:0]: ADC Channel 7~0 interrupt mask configuration 0: disable 1: Enable

26.9.17 ADC interrupt clear register(ADC_INTCLR)

Register	Address offset	Access	Reset value	Description
ADC_INTCLR	0x48	W	0x0000_0000	ADC interrupt clear register

MG32L003 F8

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

ADC interrupt clear register(ADC_INTCLR)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11]	W	CONT_INTC: Write 1 Clear continuous conversion complete flag write 0 no effect
[10]	W	REG_INTC: Write 1 Clear ADC Conversion result comparison interval flag write 0 no effect
[9]	W	HHT_INTC: write 1 clear ADC conversion result comparison high threshold write 0 no effect
[8]	W	LLT_INTC: Write 1 Clear ADC Conversion result comparison lower threshold flag write 0 no effect
[7:0]	W	ADCICLR[7:0]: ADC Channel 7~ 0 interrupt mask configuration write 1 clear ADC channel 7~0 interrupt status write 0 no effect

MG32L003 K8

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

ADC interrupt clear register(ADC_INTCLR)

Bit	Access	Description
[31:20]	-	Reserved, always read as 0
[19:12]	W	ADCICLR[15:8]: ADC Channel 15~8 interrupt mask configuration write 1 clear ADC channel 15~8 interrupt status write 0 no effect
[11]	W	CONT_INTC: Write 1 Clear continuous conversion complete flag write 0 no effect
[10]	W	REG_INTC: Write 1 Clear ADC Conversion result comparison interval flag write 0 no effect
[9]	W	HHT_INTC: write 1 clear ADC conversion result comparison high threshold write 0 no effect
[8]	W	LLT_INTC: Write 1 Clear ADC Conversion result comparison lower threshold flag write 0 no effect

[7:0]	W	ADCICLR[7:0]: ADC Channel 7~0 interrupt mask configuration write 1 clear ADC channel 7~0 interrupt status write 0 no effect
-------	---	--

26.9.18 ADC Pre-mask interrupt status register(ADC_RAWINTSR)

Register	Address offset	Access	Reset value	Description
ADC_RAWINTSR	0x4C	R	0x0000_0000	ADC Pre-mask interrupt status register

MG32L003 F8

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

ADC Pre-mask interrupt status register(ADC_RAWINTSR)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11]	R	CONT_INTF: Continuous Conversion Complete Flag 0:ADC continuous conversion incomplete 1:ADC continuous conversion complete
[10]	R	REG_INTF: ADC conversion result comparison interval flag 0:ADC conversion result is outside[ADC_LT,ADC_HT) interval 1:ADC conversion result is within[ADC_LT,ADC_HT) interval
[9]	R	HHT_INTF: ADC conversion result compare the high threshold flag 0:ADC conversion result is outside[ADC_HT,4095] interval 1:ADC conversion result is within[ADC_HT,4095] interval
[8]	R	LLT_INTF: ADC conversion result compare the lower threshold flag 0:ADC conversion result is outside[0,ADC_LT) interval 1:ADC conversion result is within[0,ADC_LT) interval
[7:0]	R	ADCRIS[7:0]: ADC channel 7~0 conversion complete interrupt status (before mask)

MG32L003 K8

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

ADC Pre-mask interrupt status register(ADC_RAWINTSR)

Bit	Access	Description
[31:20]	-	Reserved, always read as 0
[19:12]	R	ADCRIS[15:8]: ADC channel 15~8 conversion complete interrupt status (before mask)
[11]	R	CONT_INTF: Continuous Conversion Complete Flag 0:ADC continuous conversion incomplete 1:ADC continuous conversion complete
[10]	R	REG_INTF: ADC conversion result comparison interval flag 0:ADC conversion result is outside[ADC_LT,ADC_HT) interval 1:ADC conversion result is within[ADC_LT,ADC_HT) interval
[9]	R	HHT_INTF: ADC conversion result compare the high threshold flag 0:ADC conversion result is outside[ADC_HT,4095] interval 1:ADC conversion result is within[ADC_HT,4095] interval
[8]	R	LLT_INTF: ADC conversion result compare the lower threshold flag 0:ADC conversion result is outside[0,ADC_LT) interval 1:ADC conversion result is within[0,ADC_LT) interval
[7:0]	R	ADCRIS[7:0]: ADC channel 7~0 conversion complete interrupt status (before mask)

26.9.19 ADC Post-mask interrupt status register(ADC_MSKINTSR)

Register	Address offset	Access	Reset value	Description
ADC_MSKINTSR	0x50	R	0x0000_0000	ADC Post-mask interrupt status register

MG32L003 F8

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

ADC Post-mask interrupt status register(ADC_MSKINTSR)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11]	R	CONT_MIF: Interrupt after continuous conversion completes mask 0:ADC continuous conversion incomplete 1:ADC continuous conversion complete
[10]	R	REG_MIF: ADC Interrupt after conversion result comparison interval mask 0:ADC conversion result is outside[ADC_LT,ADC_HT) interval 1:ADC conversion result is within[ADC_LT,ADC_HT) interval
[9]	R	HHT_MIF: ADC Interrupt after conversion result comparison high threshold mask 0:ADC conversion result is outside[ADC_HT,4095] interval 1:ADC conversion result is within[ADC_HT,4095] interval

[8]	R	LLT_MIF: ADC Interrupt after conversion result comparison low threshold mask 0:ADC conversion result is outside[0,ADC_LT) interval 1:ADC conversion result is within[0,ADC_LT) interval
-----	---	---

[7:0]	R	ADCRIS[7:0]: ADC channel 7~0 conversion complete interrupt status (after mask)
-------	---	---

MG32L003 K8

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

ADC Post-mask interrupt status register(ADC_MSKINTSR)

Bit	Access	Description
[31:20]	-	Reserved, always read as 0
[19:12] [11]	R R	ADCRIS[15:8]: ADC channel 15~8 conversion complete interrupt status (after mask) CONT_MIF: Interrupt after continuous conversion completes mask 0:ADC continuous conversion incomplete 1:ADC continuous conversion complete
[10]	R	REG_MIF: ADC Interrupt after conversion result comparison interval mask 0:ADC conversion result is outside[ADC_LT,ADC_HT) interval 1:ADC conversion result is within[ADC_LT,ADC_HT) interval
[9]	R	HHT_MIF: ADC Interrupt after conversion result comparison high threshold mask 0:ADC conversion result is outside[ADC_HT,4095] interval 1:ADC conversion result is within[ADC_HT,4095] interval
[8]	R	LLT_MIF: ADC Interrupt after conversion result comparison lower threshold mask 0:ADC conversion result is outside[0,ADC_LT) interval 1:ADC conversion result is within[0,ADC_LT) interval
[7:0]	R	ADCRIS[7:0]: ADC channel 7~0 conversion complete interrupt status (after mask)

26.9.20 ADC channel 8 conversion result(ADC_RESULT8)**NOTE :** Only MG32L003 K8

Register	Address offset	Access	Reset value	Description
ADC_RESULT8	0x60	R	0x0000_0000	ADC channel 8 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT8[11:8]			
7	6	5	4	3	2	1	0
RESULT8[7:0]							

ADC channel 8 conversion result(ADC_RESULT8)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT8[11:0]: ADC Channel 8 Conversion result

26.9.21 ADC channel 9 conversion result(ADC_RESULT9)

NOTE : Only MG32L003 K8

Register	Address offset	Access	Reset value	Description
ADC_RESULT9	0x64	R	0x0000_0000	ADC channel 9 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT9[11:8]			
7	6	5	4	3	2	1	0
RESULT9[7:0]							

ADC channel 9 conversion result(ADC_RESULT9)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT9[11:0]: ADC Channel 9 Conversion result

26.9.22 ADC channel 10 conversion result(ADC_RESULT10)

NOTE : Only MG32L003 K8

Register	Address offset	Access	Reset value	Description
ADC_RESULT10	0x68	R	0x0000_0000	ADC channel 10 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT10[11:8]			
7	6	5	4	3	2	1	0
RESULT10[7:0]							

ADC channel 10 conversion result(ADC_RESULT10)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT10[11:0]: ADC Channel 10 Conversion result

26.9.23 ADC channel 11 conversion result(ADC_RESULT11)**NOTE :** Only MG32L003 K8

Register	Address offset	Access	Reset value	Description
ADC_RESULT1 1	0x6C	R	0x0000_0000	ADC channel 11 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT11[11:8]			
7	6	5	4	3	2	1	0
RESULT11[7:0]							

ADC channel 11 conversion result(ADC_RESULT11)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT11[11:0]: ADC Channel 11 Conversion result

26.9.24 ADC channel 12 conversion result(ADC_RESULT12)**NOTE :** Only MG32L003 K8

Register	Address offset	Access	Reset value	Description
ADC_RESULT1 2	0x70	R	0x0000_0000	ADC channel 12 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT12[11:8]			
7	6	5	4	3	2	1	0
RESULT12[7:0]							

ADC channel 12 conversion result(ADC_RESULT12)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT12[11:0]: ADC Channel 12 Conversion result

26.9.25 ADC channel 13 conversion result(ADC_RESULT13)**NOTE :** Only MG32L003 K8

Register	Address offset	Access	Reset value	Description
ADC_RESULT1 3	0x74	R	0x0000_0000	ADC channel 13 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT13[11:8]			
7	6	5	4	3	2	1	0
RESULT13[7:0]							

ADC channel 13 conversion result(ADC_RESULT13)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT13[11:0]: ADC Channel 13 Conversion result

26.9.26 ADC channel 14 conversion result(ADC_RESULT14)

NOTE : Only MG32L003 K8

Register	Address offset	Access	Reset value	Description
ADC_RESULT1 4	0x78	R	0x0000_0000	ADC channel 14 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT14[11:8]			
7	6	5	4	3	2	1	0
RESULT14[7:0]							

ADC channel 14 conversion result(ADC_RESULT14)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT14[11:0]: ADC Channel 14 Conversion result

26.9.27 ADC channel 15 conversion result(ADC_RESULT15)

NOTE : Only MG32L003 K8

Register	Address offset	Access	Reset value	Description
ADC_RESULT1 5	0x7C	R	0x0000_0000	ADC channel 15 conversion result

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				RESULT15[11:8]			
7	6	5	4	3	2	1	0
RESULT15[7:0]							

ADC channel 15 conversion result(ADC_RESULT15)

Bit	Access	Description
[31:12]	-	Reserved, always read as 0
[11:0]	R	RESULT15[11:0]: ADC Channel 15 Conversion result

27 Low Voltage Detector (LVD)

27.1 LVD introduction

LVD can be used to monitor the operating voltage, when the comparison result of the monitored voltage and LVD threshold meets the trigger condition, LVD will generate an interrupt or reset signal. An interrupt or reset signal can only be cleared by an interrupt or reset clear signal. Only after the interrupt or reset signal is cleared, the interrupt or reset signal will be generated again under the trigger condition.

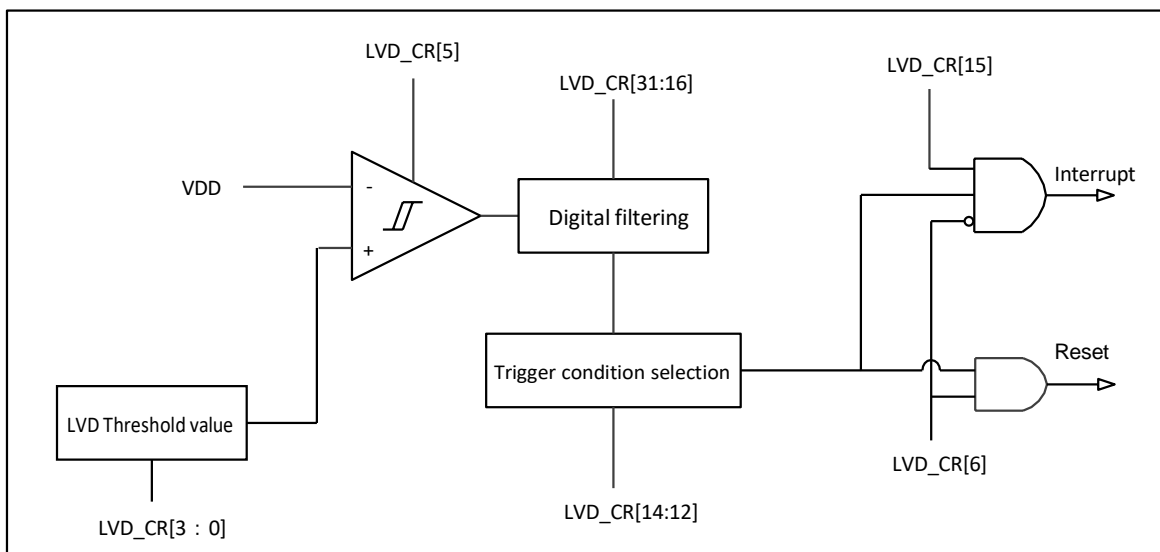
The sampling filter clock is configurable and can be configured as APB clock or LSI. The filter count value is configurable. When the number of sampling reaches the filter count value, the results are consistent and output.

2 trigger conditions: high level, rising edge, falling edge combination.

2 trigger results: interrupt, reset signal (prohibits selection of reset signal when filter clock selection PCLK).
Interrupt and reset signals cannot be generated at the same time.

27.2 LVD Block Diagram

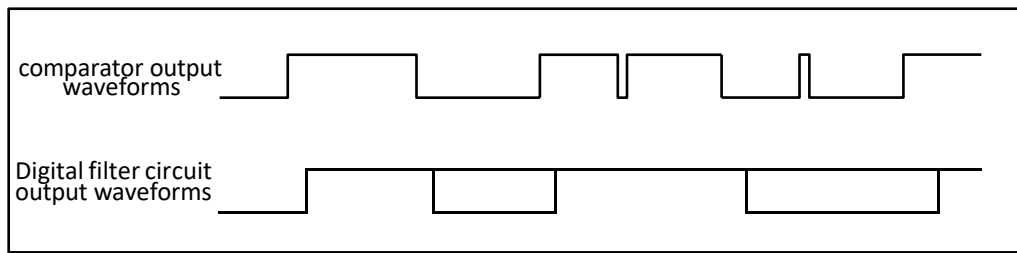
Fig 27.2-1 LVD block diagram



27.3 Digital filtering

If the working environment of the chip is bad, the output of the hysteresis comparator will appear noise signal. If the digital filter module is enabled, the noise signal whose pulse width is less than the LVD_CR.FLT_NUM[15:0] setting time in the output waveform of the hysteresis comparator can be filtered out. If the digital filter module is disabled, the input and output signals of the digital filter module are the same. Enable the digital filtering module, and the filtering diagram is as follows:

Fig 27.3-1 LVD filtered output



27.4 Configuration example

27.4.1 LVD configured as low voltage reset

In this mode, reset MCU when the monitored voltage is lower than the threshold voltage. The configuration method is as follows:

1. Configure LVD_CR.DIV_SEL to select the voltage division to be monitored.
2. Configure LVD_CR.FLT_NUM, select LVD filter time.
3. Configure LVD_CR.FLTCLK_SEL to select the filter clock.
4. Configure LVD_CR.FLTEN, enable LVD filter.
5. Set LVD_CR.HIGHINTEN to 1, select high level trigger LVD action.
6. Set LVD_CR.ACTto1, select LVD to reset the action.
7. Set LVD_CR.LVDEN to 1, enable LVD.

27.4.2 LVD configured as voltage change interrupt

In this mode, an interrupt is generated when the monitored voltage goes above or below the threshold voltage.

The configuration method is as follows:

1. Configure LVD_CR.DIV_SEL Select the voltage source to be monitored.
2. Configure LVD_CR.FLT_NUM, select LVD filter time.
3. Configure LVD_CR.FLTCLK_SEL to select the filter clock.
4. Configure LVD_CR.FLTEN, enable LVD filter.
5. Set LVD_CR.RISEINTEN to 1, or LVD_CR.FALLINTEN to 1, or both for 1, select level change trigger LVD action.
6. Set LVD_CR.ACT to 0, select LVD as interrupt.
7. Set LVD_CR.INT_EN to 1, enable LVD interrupt.
8. Set LVD_CR.LVDEN to 1, enable LVD.
9. Write 0 to LVD_SR.INTF in the interrupt service routine to clear the interrupt flag.

27.5 LVD Registers

Tab 27.5-1 LVD Registers

offset	Register	Reset value	Description
lvd base address:0x4000_4000			
0x00	LVD_CR	0x0000_0007	LVD Control Register(LVD_CR)
0x04	LVD_SR	0x0000_0000	LVD Status Register(LVD_SR)

27.6 LVD Registers Description

27.6.1 LVD Control Register(LVD_CR)

Register	Address offset	Access	Reset value	Description
LVD_CR	0x00	RW	0x0000_0007	LVD Control register(LVD_CR)

31	30	29	28	27	26	25	24
FLT_NUM[15:8]							
23	22	21	20	19	18	17	16
FLT_NUM[7:0]							
15	14	13	12	11	10	9	8
INT_EN	HIGHINTEN	RISEINTEN	FALLINTEN	Reserved		FLTCLK_SEL[1:0]	
7	6	5	4	3	2	1	0
FLTEN	ACT	LV DEN	Reserved	DIV_SEL[3:0]			

LVD Control register(LVD_CR)

Bit	Access	Description
[31:16]	RW	FLT_NUM[15:0]: LVD Sample filter count value Sampling clock is ABP clock or LSI. The filter count value is configurable. When the number of Sampling times reaches the filter count value, the results are consistent and output. Sample count period=FLT_NUM[15:0]
[15]	RW	INT_EN: LVD Interrupt Enable 0: disable 1: Enable
[14]	RW	HIGHINTEN: High level trigger enable(VDD is lower than threshold voltage) 0: Disabled 1: Enable
[13]	RW	RISEINTEN: Rising edge trigger enable(VDD changes from above threshold voltage to be- low threshold voltage) 0: Disable 1: Enable
[12]	RW	FALLINTEN: Falling edge trigger enable(VDD changes from below threshold voltage to above threshold voltage) 0: Disable 1: Enable
[11: 10]	-	Reserved

[9: 8]	RW	FLTCLK_SEL[1:0]: Filter Clock Selection 00: Filter clock invalid 01: filter clock selection is PCLK (Configure only the interrupt mode) 10: Filter clock selection is LSI 11: Reserved																																		
[7]	RW	FLTEN: Digital filter function configuration 0: Disable digital filter 1: Enable digital filtering																																		
[6]	RW	ACT: LVD Interrupt reset select bit 0: Generate Interrupt 1: Generate reset																																		
[5]	RW	LVDEN: LVD Enable 0: Disable LVD 1: Enable LVD																																		
[4]	-	Reserved																																		
[3:0]	[3]:W [2:0]:RW	DIV_SEL[3:0]: LVD Divider configuration <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>MG32L003 F8</th> <th>MG32L003 K8</th> </tr> </thead> <tbody> <tr><td>0000 = 4.4V</td><td>0000 = 4.6V</td></tr> <tr><td>0001 = 4.0V</td><td>0001 = 4.4V</td></tr> <tr><td>0010 = 3.6V</td><td>0010 = 4.2V</td></tr> <tr><td>0011 = 3.3V</td><td>0011 = 4.0V</td></tr> <tr><td>0100 = 3.1V</td><td>0100 = 3.8V</td></tr> <tr><td>0101 = 2.9V</td><td>0101 = 3.6V</td></tr> <tr><td>0110 = 2.7V</td><td>0110 = 3.4V</td></tr> <tr><td>0111 = 2.5V</td><td>0111 = 3.2V</td></tr> <tr><td></td><td>1000 = 3.0V</td></tr> <tr><td></td><td>1001 = 2.8V</td></tr> <tr><td></td><td>1010 = 2.6V</td></tr> <tr><td></td><td>1011 = Reserved</td></tr> <tr><td></td><td>1100 = Reserved</td></tr> <tr><td></td><td>1101 = Reserved</td></tr> <tr><td></td><td>1110 = Reserved</td></tr> <tr><td></td><td>1111 = Reserved</td></tr> </tbody> </table>	MG32L003 F8	MG32L003 K8	0000 = 4.4V	0000 = 4.6V	0001 = 4.0V	0001 = 4.4V	0010 = 3.6V	0010 = 4.2V	0011 = 3.3V	0011 = 4.0V	0100 = 3.1V	0100 = 3.8V	0101 = 2.9V	0101 = 3.6V	0110 = 2.7V	0110 = 3.4V	0111 = 2.5V	0111 = 3.2V		1000 = 3.0V		1001 = 2.8V		1010 = 2.6V		1011 = Reserved		1100 = Reserved		1101 = Reserved		1110 = Reserved		1111 = Reserved
MG32L003 F8	MG32L003 K8																																			
0000 = 4.4V	0000 = 4.6V																																			
0001 = 4.0V	0001 = 4.4V																																			
0010 = 3.6V	0010 = 4.2V																																			
0011 = 3.3V	0011 = 4.0V																																			
0100 = 3.1V	0100 = 3.8V																																			
0101 = 2.9V	0101 = 3.6V																																			
0110 = 2.7V	0110 = 3.4V																																			
0111 = 2.5V	0111 = 3.2V																																			
	1000 = 3.0V																																			
	1001 = 2.8V																																			
	1010 = 2.6V																																			
	1011 = Reserved																																			
	1100 = Reserved																																			
	1101 = Reserved																																			
	1110 = Reserved																																			
	1111 = Reserved																																			

27.6.2 LVD Status Register(LVD_SR)

Register	Address offset	Access	Reset value	Description
LVD_SR	0x04	RC_WO	0x0000_0000	LVD Status Register(LVD_SR)

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

LVD Status Register(LVD_SR)

Bit	Access	Description
[31:1]	-	Reserved
[0]	RC_W0	INTF: LVD Interrupt flag: 0: No LVD Interrupt occurred 1: Occurs LVD Interrupt writing 0 clears the interrupt flag, writing 1 has no effect.

28 Voltage Comparator(VCMP)

28.1 VCMP introduction

Analog voltage comparator VCMP is used to compare the magnitude of two input analog voltages, and output high/low level according to the comparison result. When "+" input voltage is higher than "-" input voltage, the voltage comparator outputs high level; when "+" input voltage is lower than "-" input terminal voltage, the voltage comparator outputs low level.

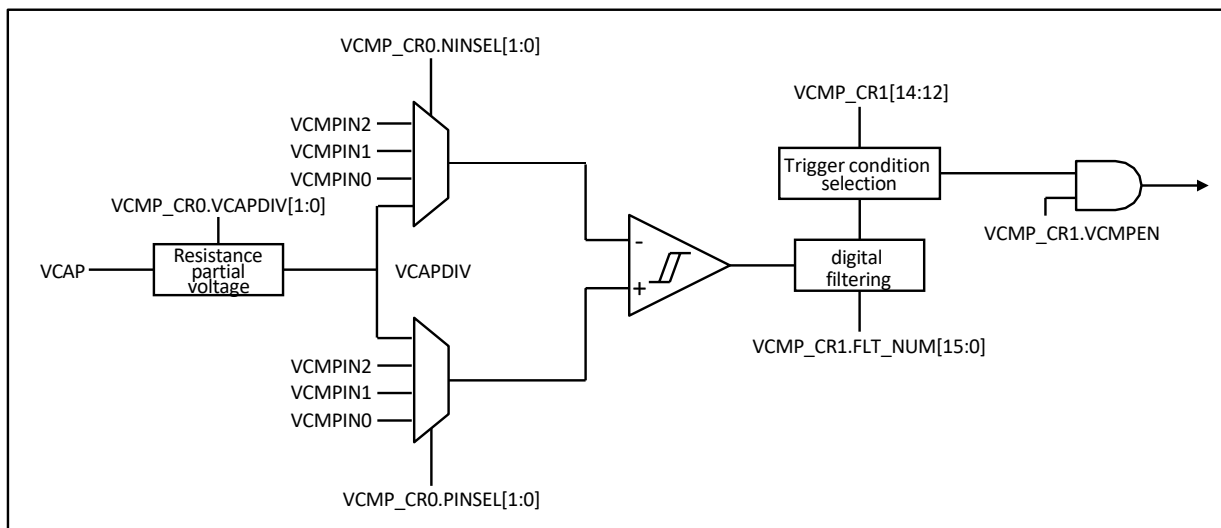
Both "+" input and "-" input support 4 power input options.

3 trigger conditions: high level, rising edge, falling edge combination.

2 trigger results: interrupt, reset signal. Interrupt and reset signals cannot be generated at the same time.

28.2 VCMP Block Diagram

Fig 28.2-1 VCMP block diagram

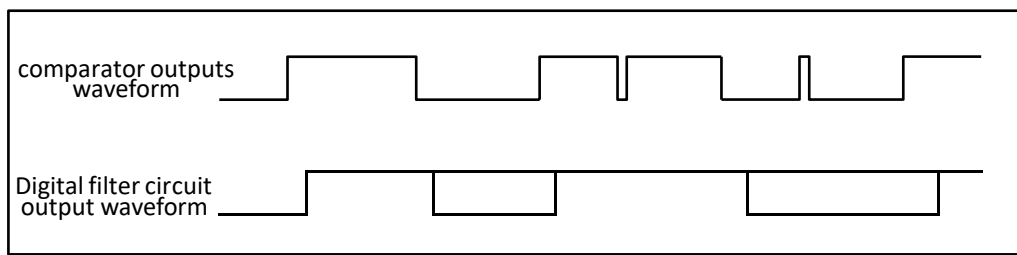


NOTE: $VCAP = 1.57V (MG32L003 K8) / 2.5V (MG32L003 F8)$

28.3 Digital Filtering

If the chip working environment is bad, the output of hysteresis comparator will appear noise signal. If the digital filter module is enabled, the noise signals whose pulse width is less than the set time of **VCMP_CR1.FLT_NUM[15:0]** in the output waveform of hysteresis comparator can be filtered. If the digital filter module is disabled, the input and output signals of the digital filter module are the same. The digital filtering module is enabled. The following figure shows the filtering

Fig 28.3-1 VCMP digital filtering



28.4 Configuration example

In this mode, an interrupt is generated when the monitored voltage goes above or below the threshold voltage. The configuration method is as follows:

1. Configure VCMP_CR0.VCAPDIV_EN to enable voltage division.
2. Configure VCMP_CR0.VCAPDIV and set the voltage division coefficient.
3. Configure VCMP_CR0.NINSEL, and select the voltage source to be monitored at the "-" terminal.
4. Configure VCMP_CR0.PINSEL, and select the voltage source to be monitored at the "+" terminal.
5. Configure VCMP_CR1.FLT_NUM[15:0] to select VCMP filter time.
6. Configure VCMP_CR1.FLTCLK_SEL to select the filter clock.
7. Configure VCMP_CR1.FLTEN to enable VCMP filtering.
8. Set the HIGHINTEN, RISEINTEN, and FALLINTEN of VCMP_CR1, and select the trigger mode
9. Set VCMP_CR1.INT_EN to 1 to enable VCMP interrupt.
10. Write 0 to VCMP_SR.INTF in the interrupt service routine to clear the interrupt flag.

NOTE: $VCAP = 1.57V(MG32L003 K8) / 2.5V (MG32L003 F8)$

28.5 VCMP Registers

Tab 28.5-1 VCMP Registers

offset	Register	Reset value	Description
VCMP base address:0x4000_4000			
0x080	VCMP_CR0	0x0000_0000	VCMP control register0
0x084	VCMP_CR1	0x0000_0000	VCMP control register1
0x088	VCMP_OUTCFG	0x0000_0000	VCMP Output Configuration Register
0x08C	VCMP_SR	0x0000_0000	VCMP Status Register

28.6 VCMP Registers Description

28.6.1 Voltage comparator control register(VCMP_CR0)

Register	Address offset	Access	Reset value	Description
VCMP_CR0	0x00	RW	0x0000_0000	Voltage comparator control register(VCMP_CR0)

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

Voltage comparator control register(VCMP_CR0)

Bit	Access	Description
[31:7]	-	Reserved
[6]	RW	VCAPDIV_EN : VCMP VCAP Divider enable control 0: disable 1: Enable
[5:4]	RW	VCAPDIV[1:0] : VCMP Voltage divider control. 00:1/4 VCAP 01:2/4 VCAP 10:3/4 VCAP 11:VCAP
[3:2]	RW	NINSEL[1:0] : "Input voltage selection. 00:VCMPIN[0] 01:VCMPIN[1] 10:VCMPIN[2] 11:VCAPDIV
[1: 0]	RW	PINSEL[1:0] : "Input terminal Voltage selection 00:VCMPIN[0] 01:VCMPIN[1] 10:VCMPIN[2] 11:VCAPDIV

NOTE: $VCAP = 1.57V(MG32L003 K8) / 2.5V (MG32L003 F8)$

28.6.2 Voltage comparator control register(VCMP_CR1)

Register	Address offset	Access	Reset value	Description
VCMP_CR1	0x04	RW	0x0000_0000	Voltage comparator control register(VCMP_CR1)

31	30	29	28	27	26	25	24
FLT_NUM[15:8]							
23	22	21	20	19	18	17	16
FLT_NUM[7:0]							
15	14	13	12	11	10	9	8
INT_EN	HIGHINTEN	RISEINTEN	FALLINTEN	Reserved			FLTEN
7	6	5	4	3	2	1	0
Reserved				VCMP_FLTCLK_SEL [1:0]		Reserved	VCMPEN

Voltage comparator control register(VCMP_CR1)

Bit	Access	Description
[31:16]	RW	FLT_NUM[15:0]: VCMP Sample filter count number Sample clock is either APB clock or LSI. The filter count value is configurable. When the number of sampling times reaches the filter count value, the results are consistent and output. Sample count period=FLT_NUM[15:0]
[15]	RW	INT_EN: VCMP Interrupt Enable 0: disable 1: Enable
[14]	RW	HIGHINTEN: VCMP Output signal high level trigger enable 0: disable 1: Enable
[13]	RW	RISEINTEN: VCMP Output signal rising edge trigger enable 0: disable 1: Enable
[12]	RW	FALLINTEN: VCMP Output signal falling edge trigger enable 0: disable 1: Enable
[11:9]	-	Reserved
[8]	RW	FLTEN: Digital filter function configuration 0: Disable digital filter 1: Enable digital filtering
[7:4]	-	Reserved
[3:2]	RW	VCMP_FLTCLK_SEL[1:0]: VCMP Filter Clock Selection 00: Filter clock invalid 01: filter clock selection is PCLK 10: Filter clock selection is LSI 11: Reserved for
[1]	-	Reserved
[0]	RW	VCMPEN: VCMP Enable 0: Voltage comparison function disabled 1: Voltage comparison function enable

28.6.3 VCMP Output Configuration Register(VCMP_OUTCFG)

Register	Address offset	Access	Reset value	Description
VCMP_OUTCFG	0x08	RW	0x0000_0000	VCMP Output Configuration Register(VCMP_OUTCFG)

Note : The reverse output function needs to be used with set of enable-bit and reverse-bit. If you just set the reverse-bit, VCMP can not output the reverse comparison result to other peripherals. For example, to output the filter-out comparison result in reverse to TIM1 capture channel 4, we need to set TIM1CH4_EN(bit16) and INV_TIM1_CH4(bit15) at the same time.

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					INV_PAD	TM1BKE	TM1CH3_EN
15	14	13	12	11	10	9	8
INV_TM1CH4	TM1CH3_EN	INV_TM1CH3	TM1CH2_EN	INV_TM1CH2	TM1CH1_EN	INV_TM1CH1	PCAECI_EN
7	6	5	4	3	2	1	0
PCACAP0_EN	NV_PCA	LPTIMEXT_EN	LPTIM_EN	Reserved	TIM11_EN	TIM10_EN	INV_TIMX

VCMP Output Configuration Register(VCMP_OUTCFG)

Bit	Access	Description
[31:19]	-	Reserved
[18]	RW	INV_PAD: The VCMP filter results are reversed and output to the VCMP_OUT channel 0: disable; 1: enable
[17]	RW	TM1BKE: VCMP Interrupt as TIM1 Brake Control 0: disable; 1: enable
[16]	RW	TM1CH4_EN: VCMP filter Result output to TIM1 CH4 input capture channel enabled 0: disable;1: enable
[15]	RW	INV_TM1CH4: VCMP filter results are reversed and output to the TIM1 CH4 input capture channel 0: disable;1: enable
[14]	RW	TM1CH3_EN: VCMP filter Result output to TIM1 CH3 input capture channel enabled 0: disable;1: enable
[13]	RW	INV_TM1CH3: VCMP filter results are reversed and output to the TIM1 CH3 input capture channel 0: disable;1: enable
[12]	RW	TM1CH2_EN: VCMP filter Result output to TIM1 CH2 input capture channel enabled 0: disable;1: enable
[11]	RW	INV_TM1CH2: VCMP filter results are reversed and output to the TIM1 CH2 input capture channel 0: disable;1: enable
[10]	RW	TM1CH1_EN: VCMP filter Result output to TIM1 CH1 input capture channel enabled 0: disable;1: enable
[9]	RW	INV_TM1CH1: VCMP filter results are reversed and output to the TIM1 CH1 input capture channel 0: disable;1: enable
[8]	RW	PCAECI_EN: VCMP filter output to PCA external clock enable 0: disable;1: enable
[7]	RW	PCACAP0_EN: VCMP filter Output to PCA Capture0 Enable 0: disable;1: enable
[6]	RW	INV_PCA: VCMP filter results are reversed and output to pca capture channel 0 0: disable;1: enable
[5]	RW	LPTIMEXT_EN: VCMP filter Result output to LPTIM External clock enable control 0: disable;1: enable
[4]	RW	LPTIM_EN: VCMP filter Output to LPTIM Gate enable 0: disable;1: enable
[3]	-	Reserved

[2]	RW	TIM11_EN: VCMP filter Output to TIM11 Gate enable 0: disable;1: enable
[1]	RW	TIM10_EN: VCMP filter Output to TIM10 Gate enable 0: disable;1: enable
[0]	RW	INV_TIMX: VCMP filter reverse output to TIM10,TIM11,LPTIM gated 0: disable;1: enable

28.6.4 VCMP Status Register(VCMP_SR)

Register	Address offset	Access	Reset value	Description
VCMP_SR	0x0C	RW	0x0000_0000	VCMP Status Register(VCMP_SR)

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

VCMP Status Register(VCMP_SR)

Bit	Access	Description
[31:2]	-	Reserved
[1]	R	VCMP_FLOUT: The Status after VCMP Filter 0:VCMP filter result is 0 1:VCMP filter result is 1
[0]	RC_W0	INTF: VCMP Interrupt Flag 0: No VCMP interrupt occurred 1: Occurs VCMP interrupt write 0 clear interrupt flag, write 1 have no effect

29 Option Bytes

29.1 Introduction

Each configuration field in the option bytes is used for the user to realize the configuration of some system functions

Address	name	description
Option Bytes address:00x0800 0000- 0x0800 01FF		
0x0800 0004	USERCFG 1	SWD Protection Bits Configuration
0x0800 0008	USERCFG 2	IWDGCNT[19:0]、IWDGMODE、IWDGINTMASK、IWDGON Configuration

29.2 Registers

29.2.1 User Configuration Register1(USERCFG1)

Register	Address offset	Access	Reset value	Description
USERCFG1	0x0800 0004	RW	FFFF FFFF	User Configuration Register1(USERCFG1)

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

User Configuration Register1(USERCFG1)

Bit	Access	Description
[31:1]	-	Reserved
[0]	RW	<p>SWDP: SWD protection bit. 0:SWD interface configured as protected; 1:SWD interface is configured as unprotected.</p> <p>Note: After changing this bit, the user must reset other than CPURST, then the function of this bit will be valid.</p>

29.2.2 User Configuration Register2(USERCFG2)

Register	Address offset	Access	Reset value	Description
USERCFG2	0x0800 0008	RW	FFFF FFFF	User Configuration Register2(USERCFG2)

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

User Configuration Register2(USERCFG2)

Bit	Access	Description
[31:1]	RW	IWDGON: IWDGON Configuration bit. 0: Hardware Non-automatic start IWDG 1: Hardware automatic start IWDG
[30]	-	Reserved
[29]	RW	IWDGINTMASK: IWDG Interrupt mask bit. 0:IWDG do not mask interrupt 1:IWDG mask interrupt
[28]	RW	IWDGMODE: IWDG Mode configuration bit. 0: Reset method 1: Interrupt method
[27:20]	-	Reserved
[19:0]	RW	IWDGCNT[19:0]: IWDG Count value register

Note: Since the rewriting of the option byte area needs to conform to the erasing sequence of Flash, when the user rewrites the parameters of the option byte area, it is necessary to read and backup the data in the entire option byte area, and erase the entire option After the byte area, rewrite the fields that need to be configured, and then write them all back to the option byte area.

30 Debug(DBG)

The JTAG/SWD of the general-purpose MCU is a non-encrypted traditional JTAG/SWD debug interface, which brings big hidden danger to the confidentiality of the client program and the security of the system. In order to protect user programs and improve system security, this chip integrates a security authorization circuit on the SWD port. When the chip leaves the factory, the ports are configured with SWD debugging interface by default. With the host computer software or customer program of this product, write 0 value, SWD debug ports are automatically disconnected after reset or next power-on. The debug interface SWD cannot be opened by writing 0xFF through security byte, if you want to open the SWD interface, you need to erase chip first

NOTE :

1. When the user does not set security bit, PC7 and PD1 are automatically configured as SWD debug port (PC7 Pull up, PD1 pull down); users can also configure the debug interface as GPIO by configuring RCC_SWDIOCR.SWDPORT register.
2. When the user sets security bit to 0, PC7 and P31 port and SWD debug port are automatically Disconnected, that is, the SWD debugging function cannot be used. RCC_SWDIOCR.SWDPORT cannot be written to 1.

30.1 SWD Describes the debugging interface

30.1.1 SWD Pin assignment for debug interface

2 GPIO of this chip can be used as SWD interface pins. These pins are present on all packages.

SWD Interface pin	SWD Interface Type	SWD Interface function	Pin assignment
SWDIO	input/output	Serial data input/Output	PC7
SWDCLK	enter	serial clock	PD1

30.1.2 Internal pull-up and pull-down for SWD pins

It is necessary to ensure that the SWD pins are not floating because they are directly connected to D flip-flops control the debug mode. Special attention must be paid to the SWDCLK pin, as it is directly connected to the clock terminal of some D flip-flops.

To avoid any uncontrolled I/O levels, this chip embeds internal pull-up and pull-down resistors on the SWD pins.

- SWDIO: Internal pull-up
- SWCLK: Input with pull-down

The software can also configure these I/O ports as GPIO

30.2 Working principle of SWD protected bit

1. The chip received by the customer is a blank chip, and the value of security bit is 1, so SWD is enabled by default.

2. Customers use Keil for software development. After the development is completed, they can be downloaded directly through Keil or downloaded through the burner.
3. After the customer downloads the program, download the secret key through the host computer and turn on the protection bit.
4. After the MCUs reset, the key will take effect immediately.
5. If the customer needs to perform SWD debugging again, the entire chip needs to be erased to re-open the SWD channel.

30.3 Use SWDS in low power mode

30.3.1 Using SWD in sleep mode(Sleep Mode)

In sleep mode, the system clock keeps working and the SWD connection is not interrupted

30.3.2 Use SWD in Deep Sleep Mode

1. When SYSCON_CFGR0.DBGDLSP_DIS = 0 (default value), enter deep sleep mode in Debug mode system clock will be stop, SWD connection will be broken; when SYSCON_CFGR0.DBGDLSP_DIS=1, enter deep sleep mode in Debug mode system clock will not stop, The SWD connection will not be interrupted.
2. After entering Deep Sleep Mode, the chip can be woken up through SWD port by power-on reset.

30.4 DBG Registers

Tab 30.4-1 VCMP Registers

Offset	Register	Reset value	Description
DBG base address: 0x4000_4C00			
0x00	DBG_APBZ	0x0000_0000	Debug mode control register

30.5 Debug Mode Control Register(DBG_APBZ)

Register	Address offset	Access	Reset value	Description
DBG_APBZ	0x00	RW	0x0000_0000	Debug Mode Control Register(DBG_APBZ)

31	30	29	28	27	26	25	24
Key							
23	22	21	20	19	18	17	16
Key							
15	14	13	12	11	10	9	8
Reserved				TIM2DBGSTOP	WWDGDBGSTOP	IWDGDBGSTOP	BEEPDBGSTOP
7	6	5	4	3	2	1	0
Reserved	RTCDBGSTOP	TIM1DBGSTOP	PCADBGSTOP	Reserved	LPTIMDBGSTOP	TIM11DBGSTOP	TIM10DBGSTOP

Debug Mode Control Register(DBG_APBZ)

Bit	Access	Description
[31:15]	W	Key: This register is only valid when the high bit is written 0x5A69

[14:12]	-	Reserved
---------	---	----------

[11]	RW	TIM2DBGSTOP: TIM2 debug mode stopped working 0: Debug mode counter still works 1: Debug mode counter stops working
[10]	RW	WWDGDBGSTOP: WWDG Debug mode stopped working 0: Debug mode counter still works 1: Debug mode counter stops working
[9]	RW	IWDGDBGSTOP: IWDG debug mode stopped working 0: Debug mode counter still works 1: Debug mode counter stops working
[8]	RW	BEEPDBGSTOP: BEEP debug mode stopped working 0: Debug mode counter still works 1: Debug mode counter stops working
[7]	-	Reserved
[6]	RW	RTCDBGSTOP: RTC debug mode stopped working 0: Debug mode counter still works 1: Debug mode counter stops working
[5]	RW	TIM1DBGSTOP: TIM1 debug mode stopped working 0: Debug mode counter still works 1: Debug mode counter stops working
[4]	RW	PCADBGSTOP: PCA debug mode stopped working 0: Debug mode counter still works 1: Debug mode counter stops working
[3]	-	Reserved
[2]	RW	LPTIMDBGSTOP: Low Power Timer debug mode stopped working 0: Debug mode counter still works 1: Debug mode counter stops working
[1]	RW	TIM11DBGSTOP: TIM11 debug mode stopped working 0: Debug mode counter still works 1: Debug mode counter stops working
[0]	RW	TIM10DBGSTOP: TIM10 debug mode stopped working 0: Debug mode counter still works 1: Debug mode counter stops working

31 SysTick timer(SYST)

31.1 SysTick Introduction

For the OS to support multitasking, it requires periodic context switching, which requires hardware resources such as timers to interrupt program execution. When the timer interrupt occurs, the processor will maintain OS and in exception handling perform OS task scheduling. The Cortex-M0+ processor has a simple timer called SysTick, which is used to generate periodic interrupt requests.

SysTick is a 24-bit timer and counts down. After the timer's count is reduced to 0, a programmable value is reloaded and a SysTick interrupt (interrupt number 15) is generated at the same time. This exception event causes the execution of the SysTick interrupt handling, which is part of the OS.

For systems that do not require an OS, SysTick timers can also be used for other purposes, such as timing, or providing an interrupt source for tasks that require periodic execution. SysTick interrupt generation is controllable. If interrupts are disabled, SysTick timers can still be used in polling ways, such as checking the current count or polling for overflow flags

31.2 Set SysTick

Since both the overloaded value and the current value of the SysTick timer are undefined at reset. procedure for configuring SysTick to prevent abnormal results:

1. Set SYST_CSR.ENABLE to 0 and disable SysTick.
2. Configure SYST_CSR.CLKSOURCE and select the SysTick clock source.
3. Configure SYST_RVR and select SysTick overflow period.
4. Write any value to SYST_CVR and clear SYST_CVR and SYST_CSR.COUNTFLAG.
5. Set SYST_CSR.TICKINT to 1 to enable SysTick interrupt.
6. Set SYST_CSR.ENABLE to 1, enable SysTick, and start counting.
7. Read SYST_CSR in the interrupt service program to clear the overflow flag.

NOTE: The SysTick timer overflow time is for SYST_RVR+1 SysTick clock period. The following is an example:

SysTickClock Source	SysTickClock period	SYST_RVR	SysTickTimer overflow time
Core Clock HCLK(is set to 4 MHz)	0.25us	3999	1ms = 0.25us * (3999+1)

31.3 SysTick Register

Address	Identifier	Description	reset value	CMSIS Symbol
0xE000 E010	SYST_CSR	SysTickTimer Control and Status Register	0x0000 0000	SysTick->CTRL
0xE000 E014	SYST_RVR	SysTickTimer Reload Value Register	TBD	SysTick->LOAD
0xE000 E018	SYST_CVR	SysTickTimer current value register	TBD	SysTick->VAL
0xE000 E01C	SYST_CALIB	SysTickTimer calibration value register	0x4000 9C3F	SysTick->CALIB

Note: SYST_CALIB register has corresponding relationship with RCC_STICKCR, please refer to SysTick Timercontrol register (RCC_STICKCR).

31.4 SysTick Register Description

31.4.1 SysTickTimer Control and Status Register(SYST_CSR)

Bits	Access	Reset value	Description
[31:17]	-	-	Reserved
[16]	R	0	COUNTFLAG: SysTickTimer overflow flag 1: SysTickTimer underflow occurs. 0: SysTickTimer overflow has not occurred. Read this register to clear COUNTFLAG flag
[15:3]	-	-	Reserved
[2]	RW	0	CLKSOURCE: SysTickClock source selection 1: use core clock (HCLK) 0: HCLK/4
[1]	RW	0	TICKINT: SysTickInterrupt enable 1: Enable interrupt 0: Disable interrupt
[0]	RW	0	ENABLE: SysTickTimer enable 1: EnableSysTick 0: DisableSysTick

31.4.2 SysTickTimer Reload Value Register(SYST_RVR)

Bits	Access	Reset value	Description
[31:24]	-	-	Reserved
[23:0]	RW	TBD	RELOAD: SysTickTimer reload value

31.4.3 SysTickTimer current value register(SYST_CVR)

Bits	Access	Reset value	Description
[31:24]	-	-	Reserved
[23:0]	RW	TBD	CURRENT: Read this register, get the current count value of SysTicktimer; write any value to this register, clear this register and COUNTFLAG.

31.4.4 SysTickTimer calibration value register(SYST_CALIB)

Bits	Access	Reset value	Description
[31]	R	0	NOREF: SysTickWhether the timer uses an external reference clock 0: HCLK/4 1: use core clock(HCLK)
[30]	R	1	SKEW: 10ms TENMS is the value accurate 0: Exact 1: inaccurate
[29:24]	-	-	Reserved
[23:0]	R	0x009C3F	TENMS[23:0]: SysTick 10ms Calibration value, this value uses external reference clock 10ms calibration value of HCLK/4(4MHZ).

32 Version

Version	Revision date	Summary of revisions
0.0	2023/1/07	draft version
1.0	2023/3/1	Add LQFP32 QFN32